

EP3 User Manual

EnergyPlus Unleashed

A working guide to modeling with EP3

September 2026

Contents

1. What Is EP3? 3

2. The EP3 Interface and Model Settings 10

3. The Model Wizard 24

4. Weather, Climate, and Design Conditions 36

5. Creating Building Geometry 49

6. Spaces and Zones 79

7. Constructions and Materials 82

8. Windows, Shading, and Daylighting 95

9. Internal Loads, Space Types, and Zone Types 108

10. Schedules 119

11. Heating and Cooling Loads: From Design Day to Annual Profile 129

12. Air-Side System Fundamentals 135

13. The HVAC Outliner and Creating HVAC Systems 145

14. Zone-Level HVAC Equipment and Zone Sizing 154

15. Air Loops, Branches, and Nodes 166

16. Central Plant Equipment and Plant Loops 193

17. Performance Curves: Part-Load and Efficiency Behavior 218

18. Service Hot Water and Solar Water Heating 224

19. Fans and Fan Power 236

20. Controllers and Availability Managers 249

21. Outdoor Air, Ventilation, and Energy Recovery 255

22. Infiltration, Natural Ventilation and Airflow Networks 265

23. Energy Efficiency Measures (EEMs) 275

24. Model Troubleshooting and Quality Assurance 291

25. Running Simulations 301

26. Viewing and Interpreting Results 309

27. Utility Costs **321**

29. Multi-Version EnergyPlus Support **327**

31. Importing EnergyPlus Files **330**

33. The EP3 AI Assistant **339**

35. Modeling Workflows **347**

This manual is a work in progress. The chapters below are complete and ready to use; several advanced/reference chapters are still being written and will be added in future revisions of this document.

1. What Is EP3?

1. What EP3 is

EP3 is a SketchUp extension that serves as a user interface for EnergyPlus. It does not implement its own HVAC, geometry, or simulation engine — every simulation EP3 runs is an EnergyPlus simulation, built from EnergyPlus objects (`Zone`, `AirLoopHVAC`, `ZoneHVAC:FourPipeFanCoil`, and so on), using EnergyPlus's own fields, defaults, and restrictions. Learning EP3 means learning how those objects are exposed, connected, and controlled through EP3's interface — not learning a separate modeling language.

Exposing EnergyPlus objects directly does not mean exposing them raw. EP3 does a substantial amount of work to keep a model well-formed as it grows: preventing object name collisions, managing node connections between components, building and validating `AirLoopHVAC`/`PlantLoop` branch structures, and grouping related objects into composite object types (a single `ZoneHVAC:PackagedTerminalHeatPump`, for example, is really a container coordinating several EnergyPlus sub-objects — a fan, a coil, and more — as one unit). Building geometry — walls, floors, roofs — is created with EP3's own dedicated tools rather than by hand-assembling raw SketchUp faces into thermal zones. EP3's job is to make these connections correctly so the reader can focus on the engineering decisions.

EP3 is designed to be a one-stop shop for an EnergyPlus-based project: geometry creation *(see "Creating Building Geometry")*, thermal envelope specification *(see "Constructions and Materials")*, internal loads definition *(see "Internal Loads, Space Types, and Zone Types")*, HVAC system assembly *(see "Creating HVAC Systems: Templates vs. Building from Components")*, running simulations *(see "Running Simulations")*, comparing model alternatives *(see "Energy Efficiency Measures (EEMs)")*, and viewing and analyzing results *(see "Viewing and Interpreting Results")* all happen inside EP3.

2. One model, updated in place

An EP3 project uses a single model. That model is built once and then updated as the project develops — through schematic design, design development, construction documents, and (where a project's scope calls for it) post-construction measurement and verification — rather than being rebuilt from scratch at each phase.

Design alternatives and efficiency measures are handled the same way: an Energy Efficiency Measure (EEM) captures a set of changes against the model's baseline definition, evaluated by rerunning the same model rather than by duplicating it into a separate file per alternative. A single model can carry an unlimited number of EEMs. *(see "Energy Efficiency Measures (EEMs)")* covers how EEMs are created and scoped; *(see "Viewing and Interpreting Results")* covers comparing their results side by side.

3. How EP3 relates to EnergyPlus

EP3 configures and runs the actual EnergyPlus engine — it does not reinterpret or approximate EnergyPlus's calculations. Two capabilities follow directly from that relationship:

Multiple EnergyPlus versions. EP3 tracks several recent major EnergyPlus versions at once, rather than being tied to a single release. *(see "Multi-Version EnergyPlus Support")* covers how a model's target version is set and what changes when moving a model between versions.

EnergyPlus file import. EP3 can import EnergyPlus IDF/epJSON files. Beyond bringing an existing EnergyPlus model into EP3, import is the mechanism behind EP3's reusable libraries — internal loads sets, construction/material assemblies, and HVAC system templates are built by creating the objects once in EP3, saving the template, and reusing the result across projects, rather than re-entering the same inputs every time. Templates are shareable and allow standardization across projects, modelers and teams. *(see "Import, Export, and Interoperability")* covers the mechanics; the library chapters for internal loads, constructions, and HVAC each cover their own template workflow.

EP3 also includes a built-in AI assistant that can act on a model directly. *(see "The EP3 AI Assistant")* covers what it can do and how to use it.

4. How EP3 relates to SketchUp

Working in EP3 requires only a small set of SketchUp's own tools: **Select**, **Move**, **Paint**, **Erase**, **Rectangle**, and **Stretch**. Beyond that basic tool set, the reader does not need broader SketchUp experience — EP3 does not expect familiarity with SketchUp's groups, components or layers as SketchUp concepts. EP3 uses SketchUp's 3D modeling environment for building geometry, and for displaying HVAC system diagrams, but everything specific to an EnergyPlus model is navigated through EP3's own outliners and dialogs, not through SketchUp's native panels.

The **Building Geometry Outliner** and the **HVAC Outliner** are the two primary examples, each loosely modeled on SketchUp's own Outliner but scoped to show the structure of one part of the model:

- The Building Geometry Outliner shows the building broken down into floors, the spaces contained on each floor, exterior and interior walls, roofs, and any shading objects — the full geometric structure of the model in one tree.
- The HVAC Outliner (accessible from the HVAC dialog) is organized by loop — air loops and plant loops — and shows each loop's setpoint managers, branches, and the objects placed on each branch.

(see "The EP3 Interface and Model Settings") covers the full dialog and toolbar layout in detail; this chapter only establishes that these outliners, not SketchUp's native tools, are where day-to-day navigation happens.

5. The modeling workflow, at a glance

Most projects begin in the **Wizard**, which assembles a complete, simulation-ready starting point in one step: templates for constructions and the materials they reference, internal loads and the schedules they reference, zone sizing objects and thermostats, and HVAC systems together with the curves and schedules they depend on. Starting from the Wizard means the project's supporting objects — everything a simulation requires — don't need to be assembled piece by piece. *(see "The Model Wizard")* walks through this path hands-on, with a first worked example that carries the Wizard's output through to a running simulation.

For most projects, the modeler still builds the building's actual geometry by hand, in **detailed mode** — the Wizard's optional basic rectangular core-and-shell zoning is an option for quick models, not the required path. Detailed mode is also where a model gets refined beyond its starting templates: envelope, internal loads and schedules, zoning, and custom HVAC components are all expanded or fine-tuned here. Importing a template is not limited to the Wizard's initial pass, either

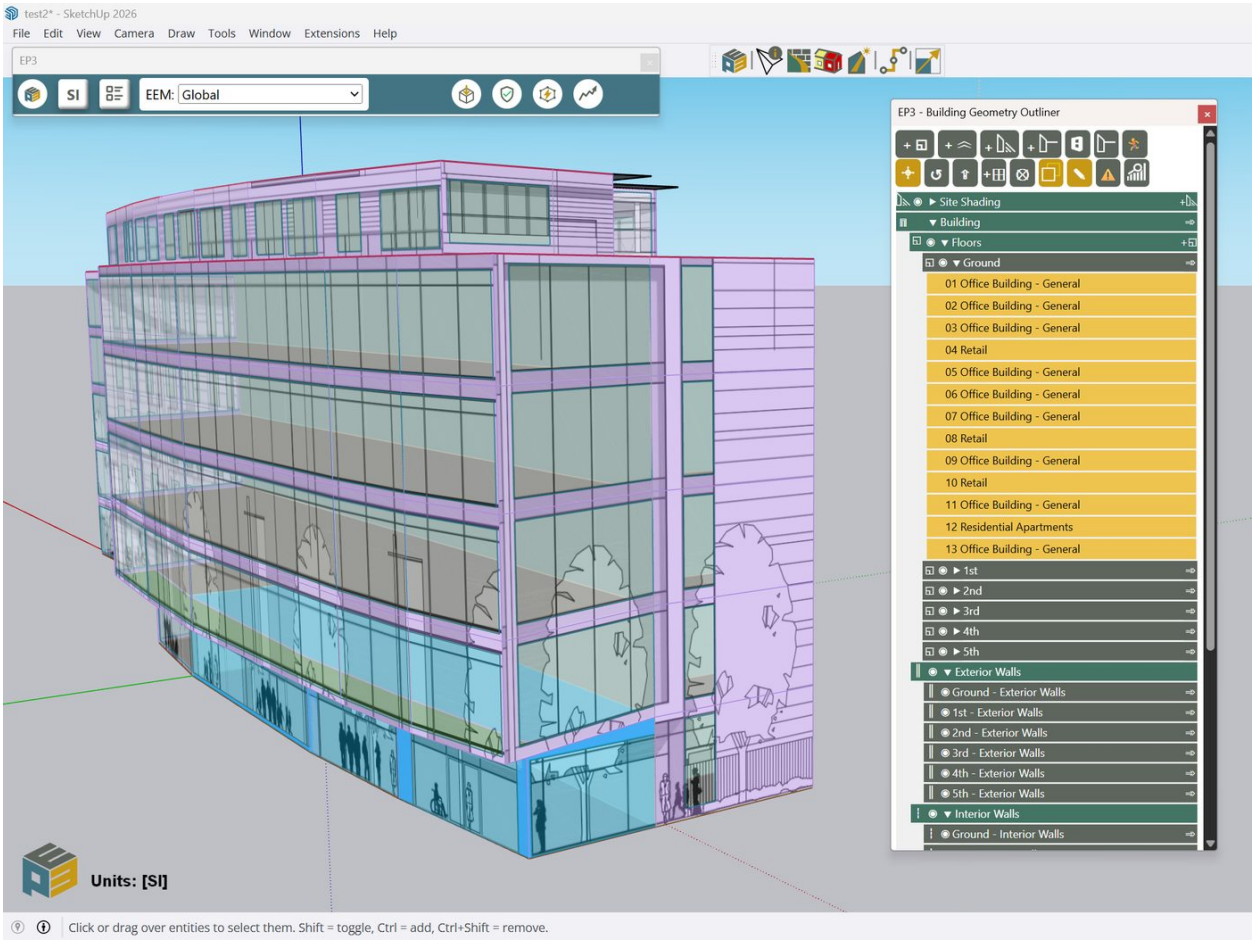


Figure 1: SketchUp's 3D viewport alongside the Building Geometry Outliner, with its tree expanded to show the building broken down into floors, the spaces on each floor, and exterior and interior walls.

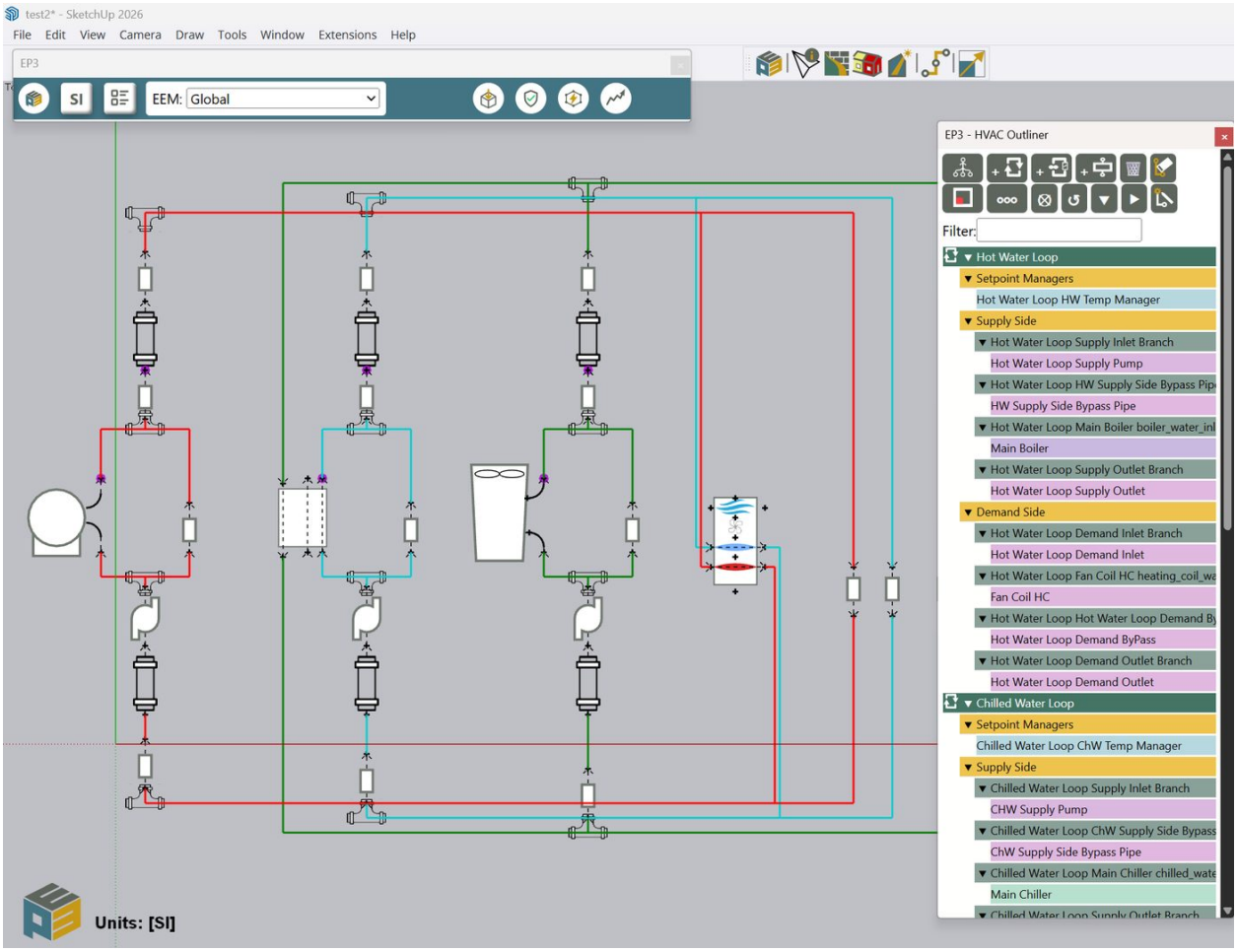


Figure 2: The HVAC Outliner showing a Hot Water Loop and a Chilled Water Loop, each with its setpoint managers and the sequence of objects on its supply and demand branches, alongside the plant loop's schematic diagram in the SketchUp viewport.

— a space type overlooked on that first pass, or an HVAC system needed later to evaluate an EEM, can be imported into the model at any stage. A model is revisited this way throughout a project rather than finished once, per §2:

- **Weather and design conditions** — assigned in the Wizard or independently. *(see "Weather, Climate, and Design Conditions")*
- **Building geometry** — typically custom-built in detailed mode; the Wizard's basic core-and-shell zoning and geometry import are the alternatives. *(see "Creating Building Geometry")* / *(see "Importing and Cleaning Up Geometry")*
- **Zones** — drawn to match the building's geometry, with zone types assigned. *(see "Internal Loads, Space Types, and Zone Types")*
- **Envelope** — constructions and materials brought in complete from a Wizard template, then adjusted as needed. *(see "Constructions and Materials")*
- **Internal loads and schedules** — brought in complete from a Wizard template, then adjusted as needed. *(see "Internal Loads, Space Types, and Zone Types")* / *(see "Schedules")*
- **HVAC systems** — brought in complete from a Wizard template, or assembled from individual components in detailed mode. *(see "The HVAC Outliner and Creating HVAC Systems")*

Equipment sizing is not a separate modeling step performed before running a simulation — it happens automatically as part of the EnergyPlus simulation itself. Any field set to Autosize is sized from the model's design conditions and the relevant sizing object: `Sizing:System` for an `AirLoopHVAC`, `Sizing:Zone` for `ZoneHVAC` equipment, `Sizing:Plant` for a `PlantLoop`. *(see "Sizing HVAC Systems and Zones")* covers these objects; *(see "Heating and Cooling Loads: From Design Day to Annual Profile")* covers reading the load profile that drives them.

Once the model runs:

- **Simulation** — run EnergyPlus against the finished model, sizing included. *(see "Running Simulations")*
- **Results and EEM comparison** — review output and compare EEMs against the baseline. *(see "Viewing and Interpreting Results")*

6. How to use this handbook

This manual assumes the reader already has working knowledge of their engineering domain — HVAC systems, building envelopes, lighting and equipment loads, and general building energy modeling practice. It does not explain what a fan is or what a wall assembly does; it explains how that concept is represented, controlled, and configured specifically in EP3.

Prior experience with EnergyPlus itself is useful but not required. A reader who already knows EnergyPlus's object model will recognize the objects EP3 exposes; a reader who does not can still build a working model using EP3's template-based workflows (system templates, internal loads libraries, construction libraries) without first learning EnergyPlus's raw input structure. Where a topic benefits from EnergyPlus's own documentation — exhaustive field-by-field detail on an object type, or the derivation of a calculation — this manual points to it rather than repeating it:

- EnergyPlus Input/Output Reference for complete field-by-field documentation of any EnergyPlus object.
- EnergyPlus Engineering Reference for the calculations and algorithms behind EnergyPlus's objects.

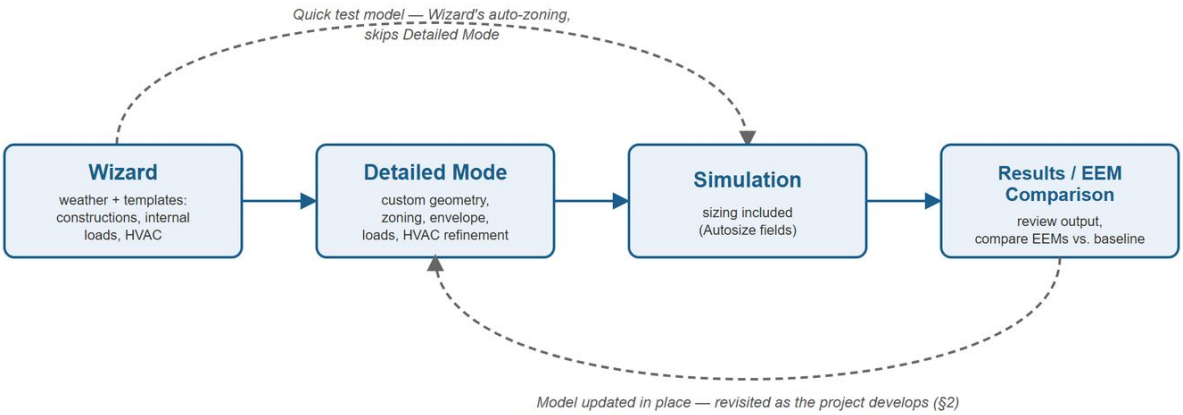


Figure 3: The EP3 modeling workflow: Wizard feeding into Detailed Mode, into Simulation, into Results / EEM Comparison, with a dashed loop back from Results to Detailed Mode for the model's ongoing revisions, and a dashed shortcut from Wizard directly to Simulation for quick test models using the Wizard's auto-zoning.

Reading order. The manual is organized into parts, running introductory to advanced:

- **Parts I–IV** are the core spine — geometry, envelope, internal loads and schedules, and HVAC systems — and are meant to be read roughly in order, since later chapters assume earlier ones (Sizing and Performance Curves in particular are shared foundations several HVAC chapters assume).
- **Part V** covers EEMs, running simulations, and results/analysis — relevant once a working model exists.
- **Part VI** covers compliance workflows (ASHRAE 90.1 Appendix G baseline generation, multi-version support, regional add-ons) and can be read once the core spine is solid.
- **Part VII** is reference material — troubleshooting, import/export, documentation/reporting — usable incrementally, independent of reading order.
- **Part VIII** is a single capstone chapter walking one model through the entire manual end to end.

Conventions used throughout:

- A **bold field name** is always the exact label shown in EP3's dialog for that field.
- An `EnergyPlus:ObjectType` shown in backticks and styled as a badge is the EnergyPlus IDD object type — the same name EnergyPlus's own documentation uses.
- Numeric examples are worked in whichever unit system (IP or SI) the source data is given in, and checked against the other unit system where useful; EP3 itself displays and accepts either.
- Cross-references to other chapters and links to external EnergyPlus documentation are given at the point they're relevant, rather than collected in one place.

End of draft chapter. Outstanding items for the next pass: resolve the XREF and EXTLINK placeholders once their target chapters/pages exist; confirm the Building Geometry Outliner's exact accessible-from location (toolbar/menu) once Ch.2 nails the full dialog inventory.

2. The EP3 Interface and Model Settings

This chapter maps the EP3 interface: the toolbar that launches it, the Main Dialog and the other windows reachable from that toolbar, and the two settings groups — Application Settings and Model Settings — that live inside the Main Dialog. It covers where each piece is and what it's for, not the full mechanics of any one workflow; later chapters go deep on importing, validating, simulating, and viewing results.

1. Starting EP3

EP3 adds a single toolbar to SketchUp, named **EP3**. Its icons, left to right, are:

- **EP3 Tools** — opens the Main Dialog (§2).
- **View Info for Selected Object** — opens the Selected Object Dialog (§3).
- **View Constructions Dialog** — opens the Constructions Dialog (§4).
- **View Building Geometry + Outliner** — opens the Building Geometry View and Outliner (§5).
- **Convert Geometry to Sketchup Analysis Hub Compatible Geometry** — converts the model's geometry into a form SketchUp's built-in Analysis Hub extension can use for daylighting studies. It doesn't open an EP3 window; it runs once against the current model. *(see "Windows, Shading, and Daylighting", §7)*
- **View HVAC + Outliner** — opens the HVAC Systems View and Outliner (§6).
- **Scale Object** — rescales the selected SketchUp object. Like the Analysis Hub icon, this runs once rather than opening a window. *(see "Importing and Cleaning Up Geometry")*

(EP3 also adds a second, smaller toolbar, **EP3 → eQuest**, with a single **Allow eQuest Merge** icon. That workflow isn't covered in this manual yet.)

In a SketchUp model that hasn't been set up for EP3 yet, clicking any of these icons opens the **Model Wizard** instead of that icon's usual window. *(see "The Model Wizard")* Once the Wizard has initialized the model, every icon opens its own window as described below.



Figure 4: The EP3 toolbar in SketchUp.

2. The Main Dialog

The **EP3 Tools** icon opens the Main Dialog, titled **EP3**. Below a control strip that runs across its top (§2.1), it's a three-column browser:

- **Column 1** lists object types, arranged in groups (Simulation Parameters, Location and Climate, Internal Loads, HVAC Design Objects, Schedules, and so on).

- **Column 2** lists the instances of whichever object type is selected in Column 1.
- **Column 3** shows the fields for whichever instance is selected in Column 2.

All three columns can be resized or collapsed.

Column 1 does **not** list every EnergyPlus object type in the model — only object types with no dedicated place to edit them elsewhere in EP3. An `AirLoopHVAC` or `PlantLoop`, for example, isn't in Column 1 at all; it's managed entirely through the HVAC Systems View and Outliner (§6). A `Zone`, by contrast, is drawn as SketchUp geometry and browsable through the Building Geometry Outliner (§5) — but that Outliner is for navigating and selecting geometry, not for editing fields, so `Zone`'s own EnergyPlus fields (Direction of Relative North, Ceiling Height, Floor Area, and so on) are still edited through Column 1, the same as any schedule, internal loads definition, or other object type with no other dedicated editor. Column 1 also holds two pseudo-groups that aren't EnergyPlus data at all — **Application Settings** and **Model Settings**, covered in §7 and §8. EP3 marks that distinction visually: Application Settings and Model Settings display with a gold group header, while every group backed by real EnergyPlus data displays with the standard teal header.

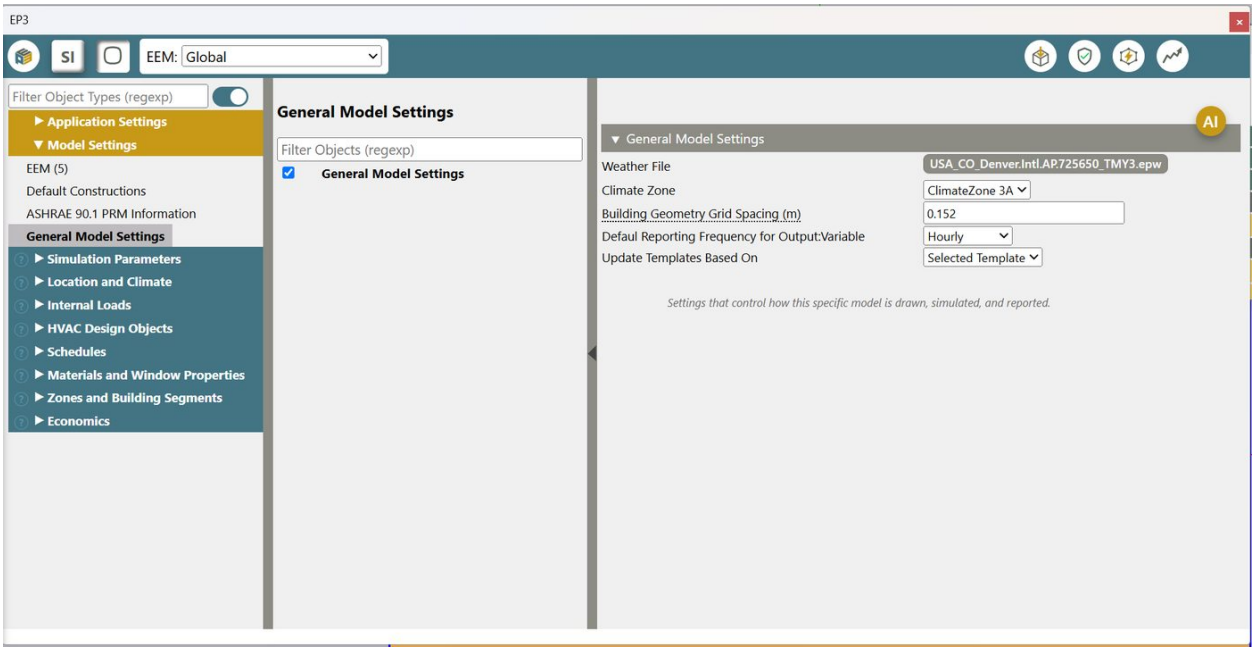


Figure 5: The Main Dialog with the Model Settings group expanded in Column 1, its General Model Settings entry selected in Column 2, and that entry's fields shown in Column 3.

2.1 The Top Control Strip

A control strip runs across the top of the Main Dialog, above the three columns. Left to right:

- **Visit EP3 website** — opens the EP3 website in your system browser, for tutorials and help.
- **Units toggle** — switches every field in the dialog between SI and IP units.
- **Simplified/complex view toggle** — switches the dialog between a reduced view (simplified) and the full view (complex). Simplified view can hide entire groups and object types from Column 1 (not just individual fields), hide some object types' custom Column 2/3 buttons, and hide some fields and checkboxes within Column 3 — §2.4 shows a worked before/after

example. Some settings and object types, including ASHRAE 90.1 PRM Information and the report categories in §8, are hidden entirely in simplified view. You may toggle back and forth between simplified and complex view as much as you like.

- **EEM dropdown** — selects which Energy Efficiency Measure is currently active for editing. *(see "Energy Efficiency Measures (EEMs)")*
- **Import** — imports EnergyPlus files or templates into the model. *(see "Import, Export, and Interoperability").* When in simplified view mode, clicking **Import** will bring you directly to the library importer, and will skip the text paste importer and file importer.
- **Validate Model** — checks the model for common issues (space enclosure, missing constructions, and similar). *(see "Model Troubleshooting and Quality Assurance")*
- **Simulate Model** — runs the model through EnergyPlus. *(see "Running Simulations")*
- **View Results** — opens the results viewer. *(see "Viewing and Interpreting Results")*
- **PRM Messages** — opens the ASHRAE 90.1 Appendix G baseline generation log; only visible on a project using that workflow. *(see "ASHRAE 90.1 Appendix G Baseline Generation")*

Clicking anywhere on the control strip itself, rather than one of its buttons, collapses the Main Dialog down to just this strip, hiding Columns 1–3 and shrinking the window to match; clicking it again restores the full three-column view. This is a quick way to get the dialog out of the way while keeping its controls (units, EEM, Simulate, and so on) reachable.

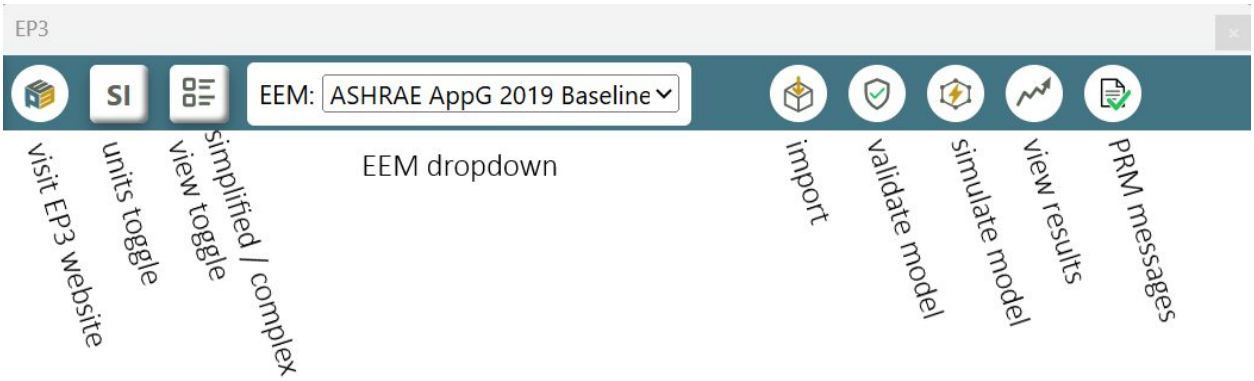


Figure 6: The Top Control Strip, labeled left to right: visit EP3 website, units toggle, simplified/complex view toggle, EEM dropdown, import, validate model, simulate model, view results, PRM messages.

2.2 Column 1: finding an object type

Above the group list is a filter box, labeled to accept a regular expression, that narrows Column 1 to matching group and object-type names as you type. **Ctrl+H** jumps straight to this box.

A toggle to the right of this box switches between showing every object type available within this dialog, and showing only the object types that already have at least one instance in the current model. **Ctrl+L** toggles the same switch.

Groups are listed in a fixed order that roughly follows the EnergyPlus Input/Output Reference's own group order, not alphabetically, with a handful of groups deliberately moved (Templates near the top; Availability Managers, Setpoint Managers, and Refrigeration near the bottom). Each group

header has a **?** icon that opens that group's page in the EnergyPlus Input/Output Reference. Not every group has a published I/O Reference page; groups without one show a blank space instead of the icon.

Expanding a group lists every object type it contains. An object type is shown in black if the model already has at least one instance of it, and in red if it doesn't — a quick way to see which object types are in use within a group without opening each one.

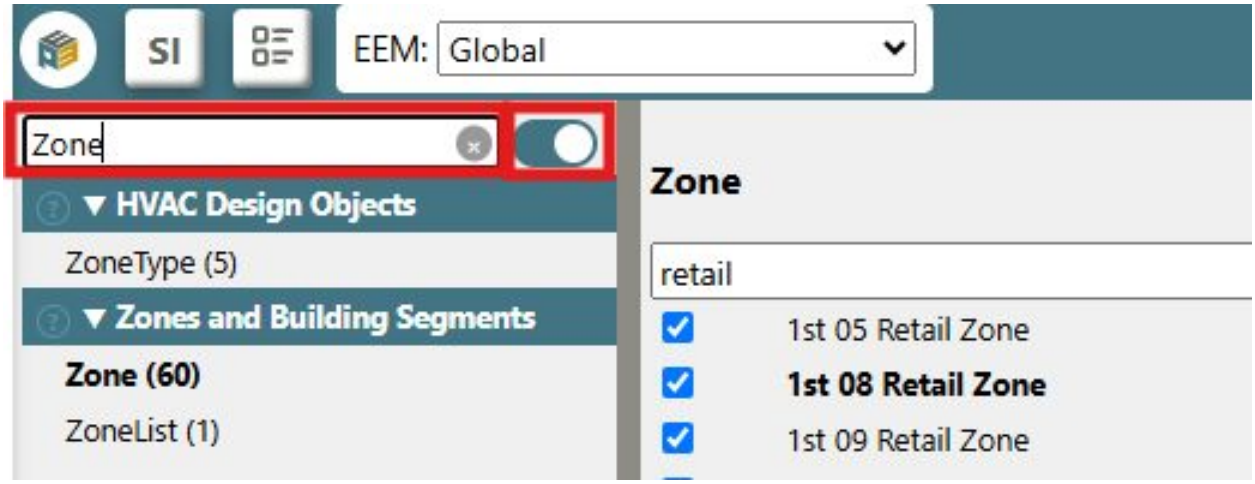


Figure 7: The Column 1 filter box (typing "Zone" to narrow the list) and, to its right, the all-object-types/in-model-only toggle.

2.3 Column 2: working with instances

Once a group and object type are selected in Column 1, Column 2 lists its instances and, below the list, a row of buttons: **Add Object**, **Delete Selected**, **Select All**, **Select None**, **Copy Selected**, and **Include Selected in EEM / Exclude Selected in EEM**. *(see "Energy Efficiency Measures (EEMs)")* covers EEM inclusion in full. Some object types also show a **Select Unused Objects** button, which selects instances nothing else in the model currently references, or their own extra buttons beyond this common set — §2.4 covers one such example.

Object types that can't have more than one instance — `SimulationControl` and the heat-balance algorithm objects, for example — don't get an **Add Object** button once an instance already exists; Column 2 shows only **Delete** until that instance is removed.

Column 2 has its own filter box, separate from Column 1's, for narrowing the instance list by name. Filtering first and then clicking **Select All** selects only the instances currently visible in the filtered list, not every instance of that type. Be aware that any objects that were **previously** selected and hidden will remain selected.

Selecting an object in Column 2, loads that single instance into Column 3 and allows inspection and editing of the fields for that particular object. To edit a field across several objects at once, select those objects' geometry in the SketchUp viewport itself (not their Column 2 checkboxes) before editing the field in Column 3; EP3 detects that more than one selected object shares the field being edited and asks whether to apply the change to all of them.

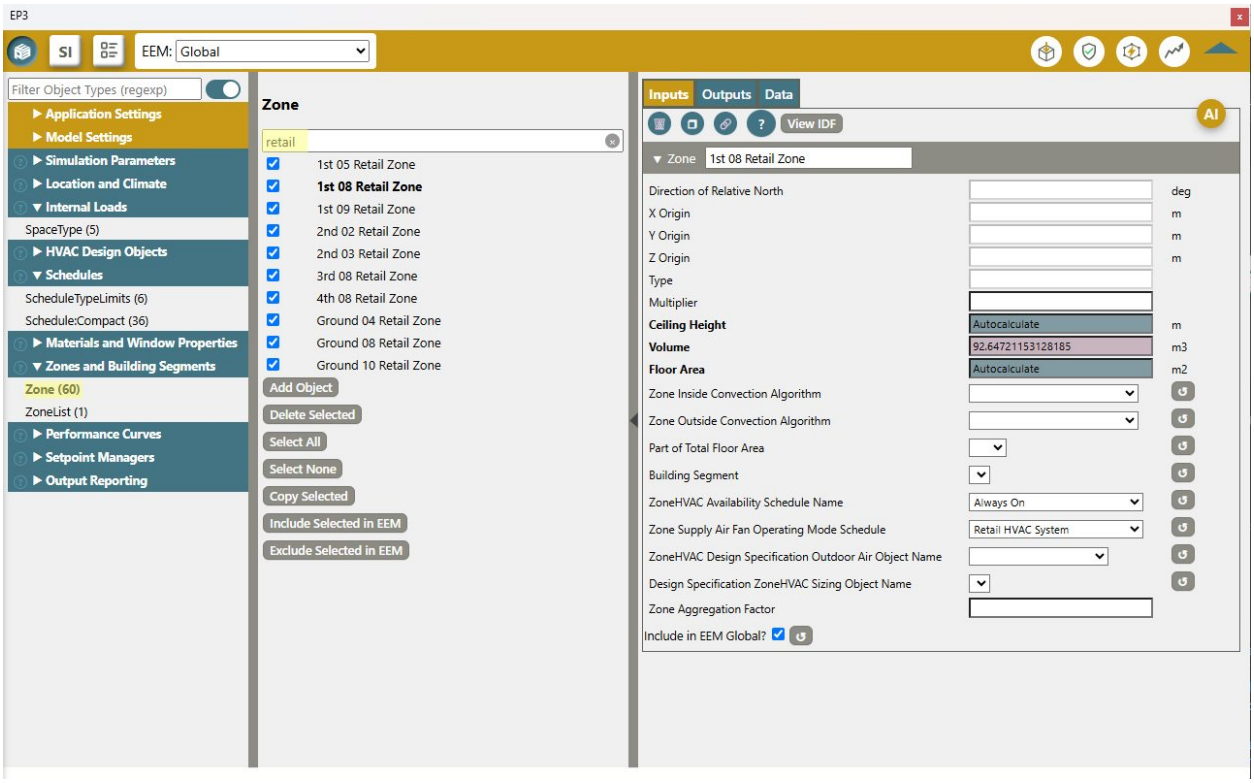


Figure 8: Column 2 for the Zone object type, filtered to "retail," with the full button row (Add Object, Delete Selected, Select All, Select None, Copy Selected, Include/Exclude Selected in EEM) below the instance filter box.

2.4 Column 3: editing an object

Column 3 shows the selected instance's fields, styled per the conventions in this manual, under an **Inputs** tab; an adjacent **Data** tab always appears, and an **Outputs** tab appears for object types that have their own EnergyPlus output variables — neither is covered in this chapter. A checkbox near the bottom of the field list, labeled **Include in EEM ?**, controls whether that instance participates in the currently active EEM; this checkbox is one of the elements simplified view can hide. *(see "Energy Efficiency Measures (EEMs)")*

Above the fields, every object type has the same standard row of icon buttons — **Delete**, **Copy**, and **View/Replace References** (which finds every other object in the model referencing this one) — plus a **?** Help button, which opens that specific object's page in the EnergyPlus Input/Output Reference (the same embedded page viewer used by Column 1's group-level **?** icon, but scoped to the object rather than its whole group), and a **View IDF** button, which opens a preview of the instance's EnergyPlus IDF text representation. Some objects also get specialized buttons here - in this case the SpaceType gets a + button, for adding a new sub-object.

A gold **AI** icon in the top-right corner of Column 3 opens EP3's built-in AI assistant, scoped to whatever's currently shown there. *(see "The EP3 AI Assistant")* covers what it can do.

A worked example: SpaceType. `SpaceType` is one of the most feature-complete object types in Column 3, since it's a container for several other object types at once. Beyond its own Inputs, Column 3 shows one tab per sub-object it holds — **Lights**, **People**, **ElectricEquipment**, **Infiltration** — each with its own field set and its own **Include in EEM** checkbox, addable via the + button and removable via the **X** on its tab. Two more tabs are specific to this kind of container: **Spaces** lists every space in the model currently assigned this SpaceType, and **ASHRAE Space Types** holds information the ASHRAE 90.1 Appendix G baseline generator uses. *(see "Internal Loads, Space Types, and Zone Types")* covers `SpaceType` itself; *(see "ASHRAE 90.1 Appendix G Baseline Generation")* covers the ASHRAE Space Types tab.

The two screenshots below show this same SpaceType and its Lights tab twice — once in complex view, once in simplified view — as an example of the Top Control Strip's simplified/complex view toggle (§2.1, boxed in red in the second screenshot). Toggling it hides many object types and the rarely-used input fields across EP3. The exact settings for simplified view are configurable for your team, or for individual users. Contact EP3 for more information on how to configure your own preferences for simplified view.

3. The Selected Object Dialog



The **View Info for Selected Object** icon opens a small floating window, titled **EP3 - Selected Object**. It shows only the fields for whichever SketchUp entity is currently selected, without the group/instance browsing of Columns 1 and 2. It's meant to stay open alongside the SketchUp viewport while modeling, showing each object's fields as you click through the model. This dialog will reduce it's size to give you the maximum available screen space when not needed.

When multiple objects of the same type are selected, making changes to an object field in the selected object dialog will result in a prompt asking if you would like to make the update for all selected objects.

A refresh icon at the top right forces this dialog to re-sync with SketchUp's current selection. Normally the dialog only updates when the selection actually changes; if it's showing stale fields for

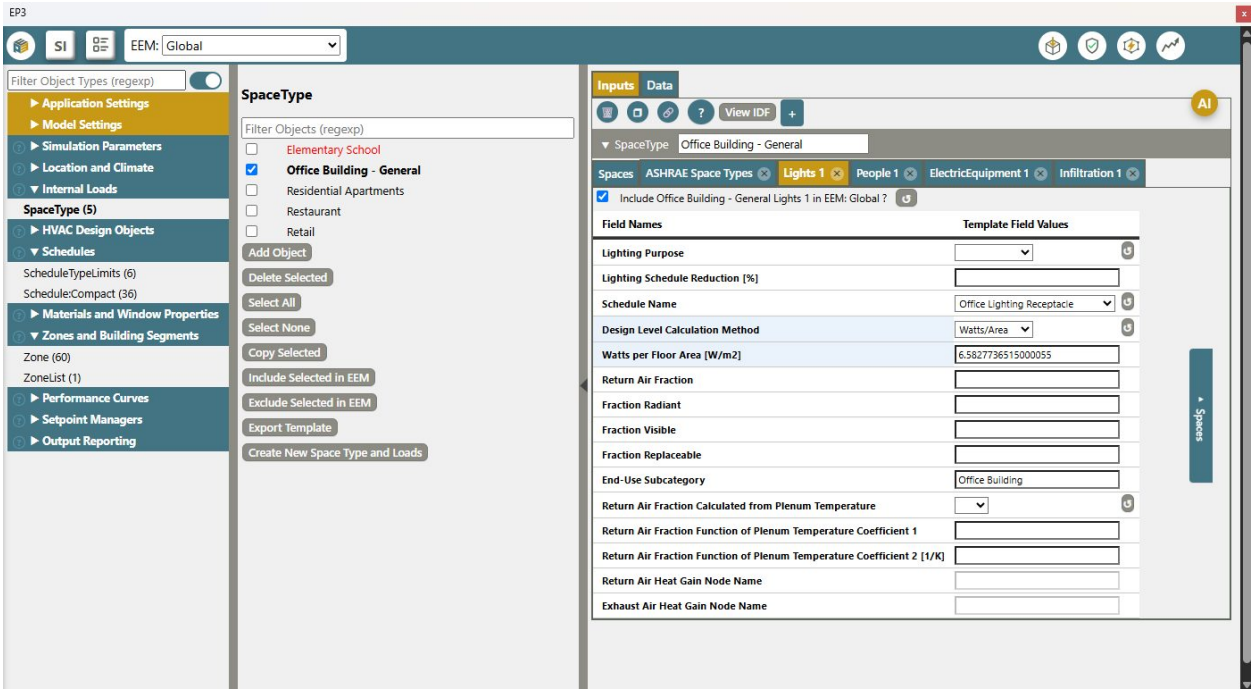


Figure 9: SpaceType in complex view: its Lights sub-object tab selected, the Include in EEM checkbox for that sub-object, its full field list, and the standard button row (Delete, Copy, View/Replace References, Help, View IDF, and the + to add another sub-object) plus Column 2's SpaceType-specific Export Template and Create New Space Type and Loads buttons.

some other reason — after an undo, for example, or a change made outside the dialog — clicking refresh re-reads the current selection (or, if nothing is selected, whatever group or component you're currently editing inside) and redraws Column 3 from it. Next to it, the gold **AI** icon opens the same built-in AI assistant available from the Main Dialog. *(see "The EP3 AI Assistant")* covers what it can do.

4. The Constructions Dialog

The **View Constructions Dialog** icon opens a window titled **EP3 - Constructions Dialog**, scoped to constructions and materials. *(see "Constructions and Materials")* covers its fields and the defaults library in full.

5. The Building Geometry View and Outliner

The **View Building Geometry + Outliner** icon opens a window titled **EP3 - Building Geometry Outliner**. As introduced in Chapter 1, it's a tree — loosely modeled on SketchUp's own Outliner panel — that shows the geometric structure of the whole model: Site Shading, then the Building broken down into Floors (with the spaces on each floor nested underneath), Exterior Walls, Interior Walls, Roofs, and Building Shading, plus a separate Floor Plans/Drawings section for any imported reference drawings. *(see "Creating Building Geometry")* and *(see "Internal Loads, Space Types, and Zone Types")* cover the objects it displays.

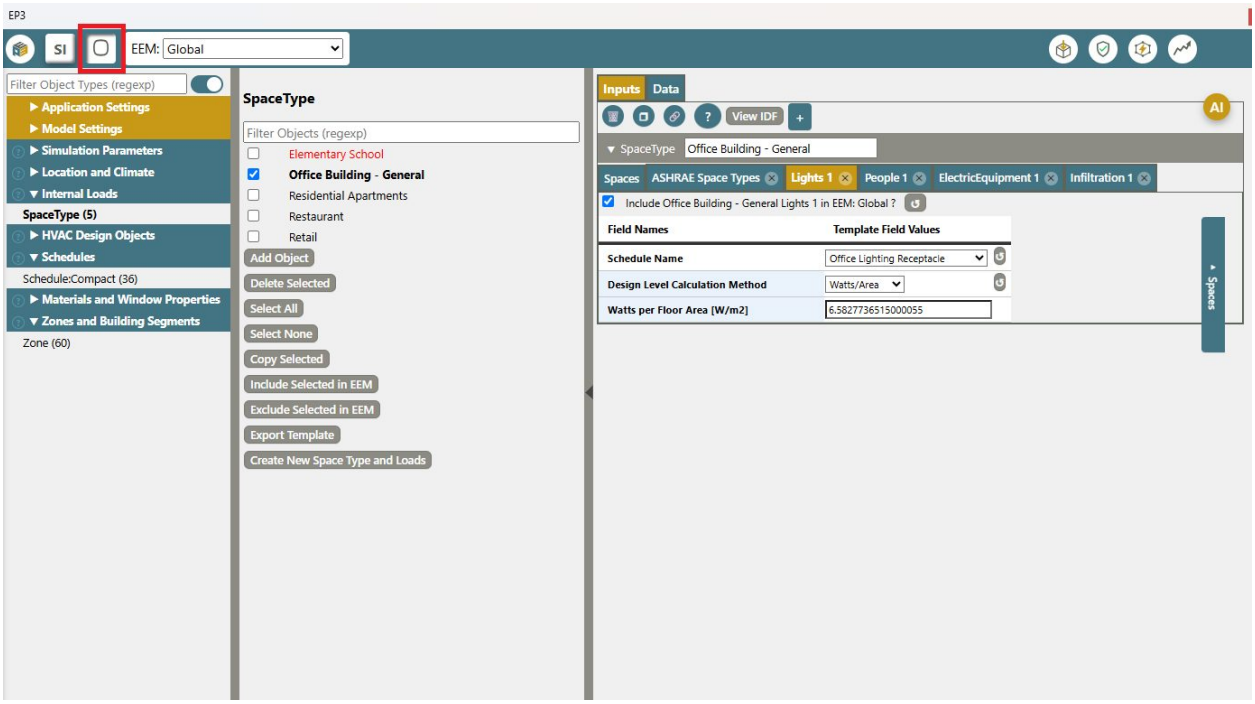


Figure 10: The same SpaceType and Lights tab in simplified view (the simplified/complex view toggle boxed in red): the ASHRAE Space Types, People, ElectricEquipment, and Infiltration tabs, the Include in EEM checkbox, several fields, and Column 2's two SpaceType-specific buttons are all hidden.

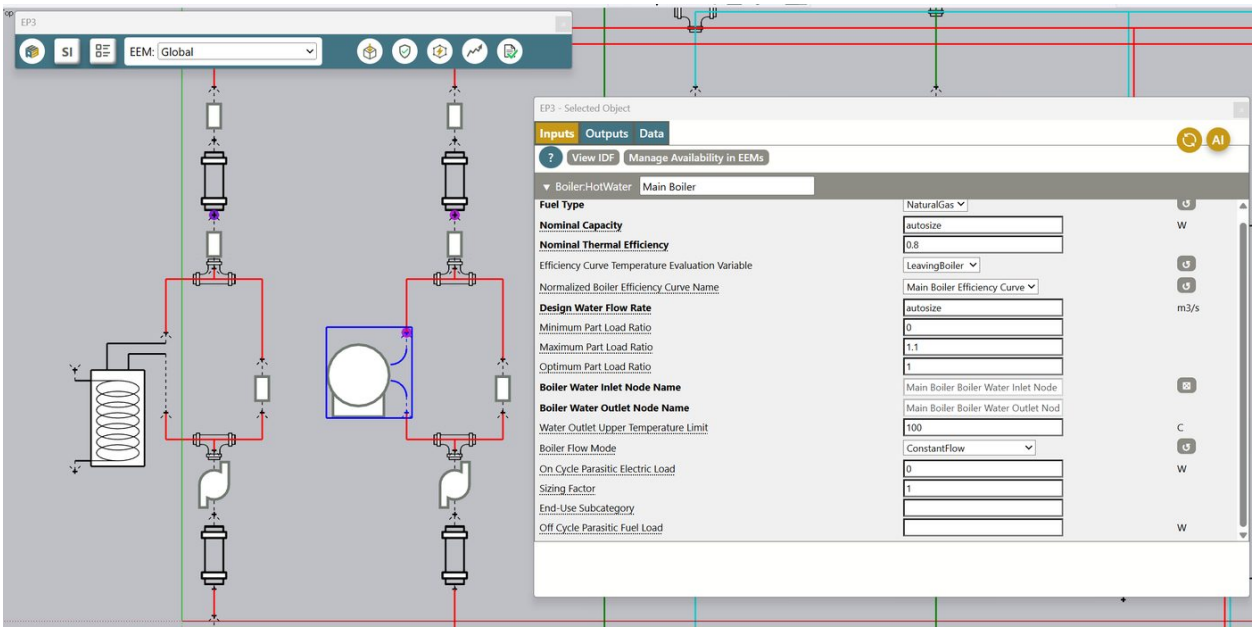


Figure 11: The Selected Object Dialog, floating over the SketchUp viewport, showing a Boiler:HotWater object's fields; the refresh icon and the AI icon sit at the top right of the tab row.

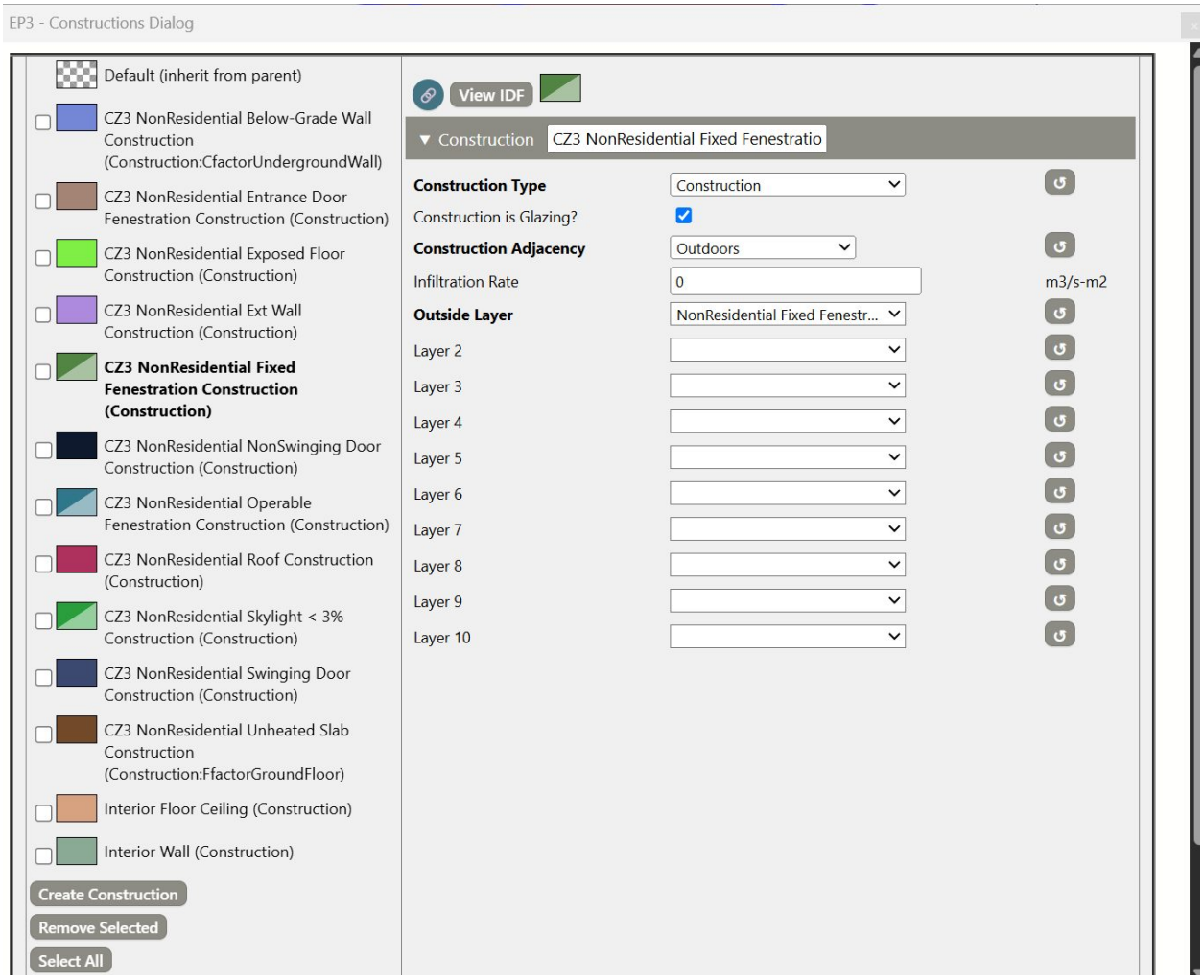


Figure 12: The Constructions Dialog: the construction list on the left, and the selected construction's fields (Construction Type, Construction is Glazing?, Construction Adjacency, Infiltration Rate, and its material layers) on the right.

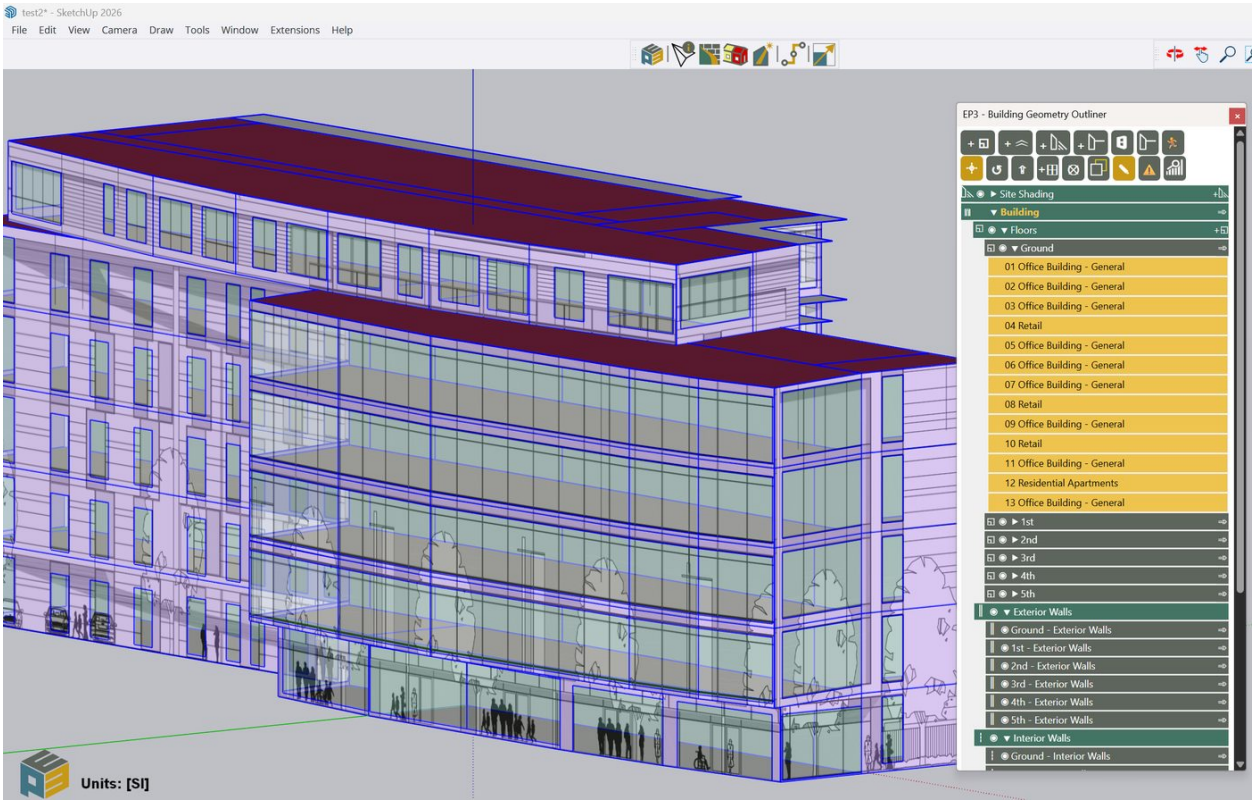


Figure 13: The Building Geometry Outliner, docked next to the SketchUp viewport, with Site Shading, Building, Floors, and the Ground floor's spaces expanded, and Exterior Walls and Interior Walls listed one entry per floor.

6. The HVAC Systems View and Outliner



The **View HVAC + Outliner** icon opens a window titled **EP3 - HVAC Outliner**. As introduced in Chapter 1, it's organized by loop — one entry per `AirLoopHVAC` and one per `PlantLoop` — and each loop expands to show its Setpoint Managers, then its Supply Side and Demand Side, each broken down into branches and the components placed on each branch. Non-loop equipment (VRF systems, zone HVAC equipment, water heaters, and unconnected objects) is listed separately, below the loops. *(see "Air Loops, Branches, and Nodes")*, *(see "Creating HVAC Systems: Templates vs. Building from Components")*, and *(see "Central Plant Equipment")* cover the objects it displays.

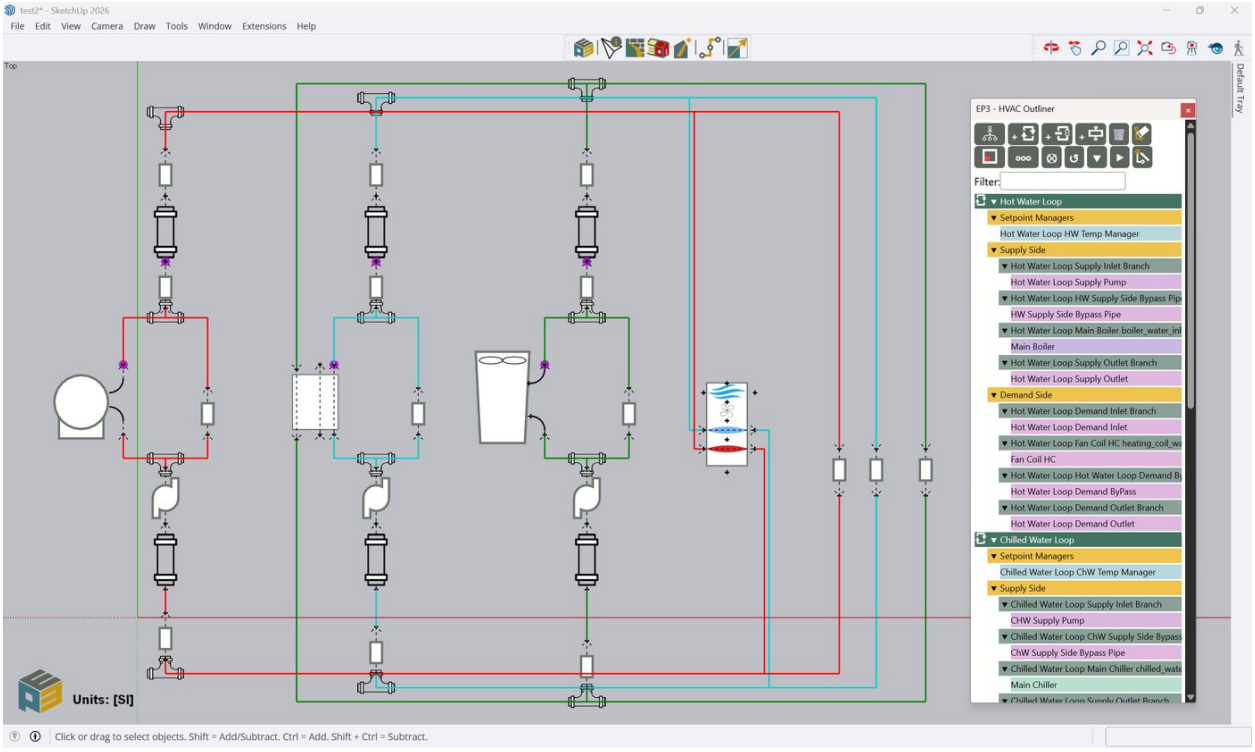


Figure 14: The HVAC Outliner, docked next to the SketchUp viewport showing a plant loop schematic, with a Hot Water Loop and Chilled Water Loop each expanded to show their Setpoint Managers and Supply Side/Demand Side branches.

7. Application Settings

Application Settings, a gold-header group in the Main Dialog's Column 1, holds settings scoped to your installation rather than to any one model — they carry over between projects. Column 2 lists four entries, covered in §7.1–§7.4.

> *Figure placeholder — the Main Dialog with Application Settings selected in Column 1, showing all four Column 2 entries, and Library Locations open in Column 3 with at least two folders listed for one category*

7.1 User Info

Energy Modeler Name, **Contact Email**, **Phone**, **Company Name** — contact details, not consumed by any simulation or export logic in this manual's scope.

7.2 EP3 License

Shows your current license key, or "Trial" if none is active.

7.3 EnergyPlus Installation

Shows the detected EnergyPlus install folder, or "Not Found." EP3 stores a separate install-folder path per EnergyPlus version, so upgrading or maintaining multiple EnergyPlus versions doesn't require reconfiguring this each time you switch. Click to open a folder picker and choose the location; EP3 checks the selected folder's `Energy+.schema.epJSON` against the version it expects and warns if they don't match. This must be set correctly before a model can be simulated. *(see "Running Simulations")*

7.4 Library Locations

The folders EP3 searches when browsing or importing reusable templates, one folder list per template category: Internal Loads *(see "Internal Loads, Space Types, and Zone Types")*, HVAC Systems *(see "Creating HVAC Systems: Templates vs. Building from Components")*, Service Water Heating *(see "Central Plant Equipment")*, Constructions *(see "Constructions and Materials")*, Weather Files *(see "Weather, Climate, and Design Conditions")*, and AI Report Generation Instructions *(see "The EP3 AI Assistant")*. Each list can hold multiple folders, reordered with Up/Down buttons; the first folder listed is where EP3 saves a new template of that type by default.

Referencing and selecting from a library. For Internal Loads, Constructions, Service Water Heating, and HVAC Systems, this is one shared mechanism, not four separate ones. In simplified view, the Top Control Strip's Import button (§2.1) opens a window titled "Select Templates / Objects to import," with one tab per category. Each tab lists the available files in two groups: **User Library** — one sub-group per folder you've configured here, labeled by that folder's name, so files from different folders stay distinguishable rather than merging into one flat list — and **EP3 Library**, the templates that ship with EP3 itself. Check whatever you want from any tab and click **Import selected** once, at the bottom, to bring all of it into the model together. In complex view, the same Import button instead opens the full Importer dialog, which also supports IDF/epJSON file import and pasting text directly. *(see "Import, Export, and Interoperability")* covers that dialog in full.

Weather is the exception: as of this writing, the Weather File Library Locations folders configured here aren't read by any browsing UI — there's no library picker for weather files. The **Weather File** field in General Model Settings (§8.3) opens a plain system file browser filtered to `.epw` files, independent of this folder list; it defaults to wherever you last picked a weather file from, not to one of these configured folders.

8. Model Settings

Model Settings, the other gold-header group, holds settings stored on the current model rather than on your installation — they're saved with the file and don't carry over to a new project. Column 2 lists five entries, covered in §8.1–§8.5.

8.1 Default Constructions

Assigns a construction to each of eight surface types — Exterior Walls, Interior Walls, Roof, Glazing, Interior Floor & Ceiling, Underground Floor, Underground Walls, Exposed Floor — chosen from the constructions already in the model. This only affects *new* geometry: a wall, roof, or floor picks up its type's default construction at the moment it's created, and changing a default afterward never repaints surfaces that already exist. To change an existing surface's construction, select it and edit its own Construction field, or repaint it. *(see "Constructions and Materials")*

This is unrelated to the ASHRAE 90.1 Appendix G baseline generator's own construction assignment, which independently assigns code-minimum constructions by climate zone when building a baseline model. *(see "ASHRAE 90.1 Appendix G Baseline Generation")*

Deleting a material currently assigned as a default will leave the default empty - this means that new geometry created **will not be given a construction at all** care must be taken to ensure that your model maintains valid and appropriate default constructions.

8.2 ASHRAE 90.1 PRM Information

Fields feeding the Appendix G baseline workflow and LEED submittals: **Building Tested Infiltration Rate**, **LEED Project Name**, **LEED Project ID**, **PRM Report Units**, **Energy Model Based On** (90% Construction Documents / 100% Construction Documents / As-built Construction Documents), and **Number of Floors in the Building**. Hidden in simplified view. *(see "ASHRAE 90.1 Appendix G Baseline Generation")* covers how these are used.

8.3 General Model Settings

- **Weather File** and **Climate Zone** — briefly, here, since *(see "Weather, Climate, and Design Conditions")* and *(see "ASHRAE 90.1 Appendix G Baseline Generation")* cover them in full.
- **Building Geometry Grid Spacing** — the spacing new points snap to while drawing with EP3's own geometry tools. It only takes effect if snap-to-grid itself is turned on elsewhere (a separate toggle, not in this dialog), and only for points picked from that moment forward — it has no effect on geometry that already exists. *(see "Creating Building Geometry")*
- **Default Reporting Frequency for Output:Variable** — the default frequency used by `Output:Variable` for reporting. Each `Output:Variable` can still be set to its own frequency individually. Changing this default only affects variables that have not had a frequency manually selected by the modeler. *(see "Viewing and Interpreting Results")*
- **Update Templates Based On** (**Name** / **Selected Template**) — controls how a space or zone gets matched to its Space Type or Zone Type template. In **Name** mode, the match is automatic: a template claims any space or zone whose name matches its own naming pattern, and there's no manual template field on the space/zone at all — renaming a space or zone can silently change, or clear, which template it's matched to. In **Selected Template** mode, each space and zone instead carries an explicit, manually-assigned template, independent of its name; a **Selected Template** field appears on the space/zone dialog, and a matching assignment control appears on the template itself. Switching from Name to Selected Template offers to snapshot every space's and zone's current name-based match as an explicit assignment, so nothing is lost in the transition; switching back to Name does not reverse that snapshot — it simply resumes matching by name going forward. *(see "Internal Loads, Space Types, and Zone Types")* covers Space Types and Zone Types themselves. Name matching

is live-refreshed each time a Space or Zone Type is accessed; in large models this can create some slowdown, and you may want to switch modes.

8.4 Import Details, Import Errors Summary, and Export (EP3 → EnergyPlus) Reports

Each lists the model's past import or export runs in Column 2; selecting one reopens that run's report. Hidden in simplified view. *(see "Import, Export, and Interoperability")* and *(see "Model Troubleshooting and Quality Assurance")* cover what these reports contain and how to act on them.

8.5 EEM

The model's defined Energy Efficiency Measures, also reachable from the EEM dropdown in the Top Control Strip (§2.1). *(see "Energy Efficiency Measures (EEMs)")*

3. The Model Wizard

1. Overview

The Model Wizard is the dialog titled **EP3 - Model Wizard**. It opens automatically the first time you interact with a brand-new SketchUp file — clicking any EP3 toolbar icon before the file has been set up for EP3 opens the Wizard instead of that icon's usual window. *(see "The EP3 Interface and Model Settings")*

The Wizard is a single dialog with one tab per category — Model Settings, Internal Loads, Constructions, Service Hot Water, HVAC Systems, Building Geometry Properties, and Utility Rates. You can move between tabs in any order, filling in as many or as few as you want, before finishing. This chapter covers each tab in the order that they appear in the dialog.

A single button at the top of the dialog, reading **"Wizard Step 1 - Select the elements to include in your model"** above a **"Next → Select ZoneHVAC Assignments"** button, ends the Wizard and imports everything selected across all tabs. Clicking it immediately opens a second, separate dialog — **EP3 - Select Zone Conditioning Equipment** — for assigning whatever HVAC equipment you selected to your zones. §9 covers that dialog in more detail.

On every tab with a checkbox tree (Internal Loads, Constructions, Service Hot Water, HVAC Systems), you can check as many templates as you like — these aren't exclusive, single-pick choices. Selecting more than one space type, construction set, or HVAC system in the same category adds all of them to the model side by side.

Every dimensional field in the Wizard (building dimensions, perimeter depth, and so on) displays in whatever unit system — IP or SI — the current SketchUp model is already using. The wizard will detect the units used by your Sketchup model, so you may want to start your journey in EP3 by selecting a Sketchup template that uses your preferred units. Once in detailed mode, you can toggle back and forth between IP and SI units for all EP3 dialogs *(see "The EP3 Interface and Model Settings")*.

---

2. Model Settings

The Model Settings tab collects the handful of settings every model needs regardless of what else you select:

- **Select EnergyPlus Version** — the EnergyPlus version this file will target. Changing it after you've already made other selections elsewhere in the Wizard restarts the dialog and discards every selection made so far; a confirmation prompt warns you before this happens.
- **Select Weather File** — opens the same weather-file picker covered in *(see "Weather, Climate, and Design Conditions")*.
- **Select Climate Zone** — used later by the ASHRAE 90.1 Appendix G baseline generator. *(see "ASHRAE 90.1 Appendix G Baseline Generation")*
- **Run Period** — Begin Month, Begin Day of Month, Begin Year, End Month, and End Day of Month, generated by the same dialog logic `RunPeriod` itself uses elsewhere in EP3. *(see "Running Simulations")* covers Run Period and other simulation settings in full.

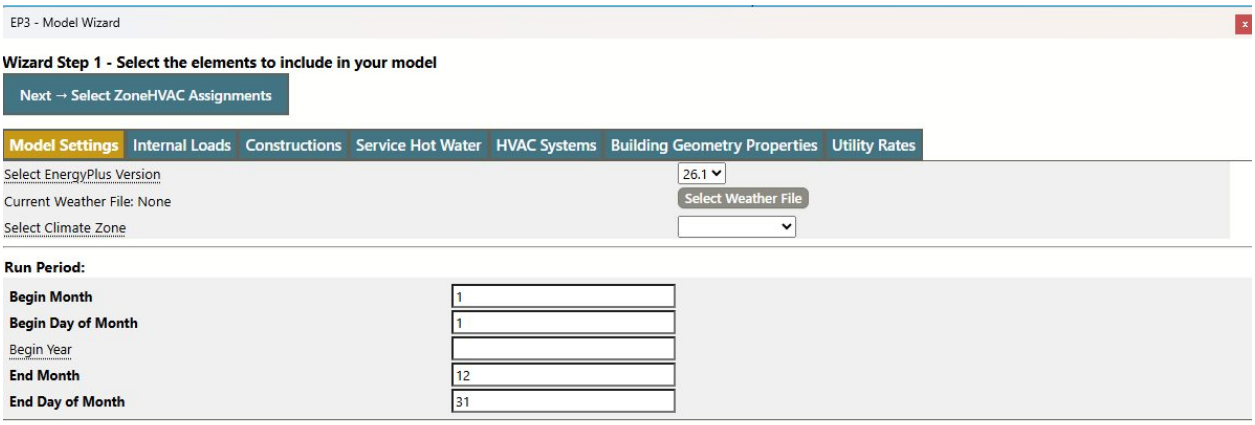


Figure 15: The Model Settings tab of the Model Wizard.

3. Internal Loads

Use this tab to select the types of spaces in your model. A space type loaded from the EP3 library includes appropriate schedules, lighting, occupancy, and internal loads for that space type — but it also includes matching zone sizing and thermostat setup, since every `SpaceType` template here is paired with a same-named `ZoneType` template: checking "Residential Apartments," for example, brings in both a "Residential Apartments" `SpaceType` (schedules, lighting, occupancy, other internal loads) and a "Residential Apartments" `ZoneType` (its `Sizing:Zone` object and thermostat setup). Rather than building all of that one piece at a time, you check off which space types your building contains and each one arrives as a single bundle of already-connected objects — internal loads and zone sizing together, in the same step.

Many of these templates also include `WaterUse:Equipment` objects — domestic water use (sinks, showers, and similar fixtures appropriate to the space type), separate from the space's heating and cooling loads. This water use shows up in results whether or not you select a Service Hot Water system on the tab that follows (§5) — see that section for what happens to it if you don't.

The library, browsable as a checkbox tree, is organized into folders:

- **NECB**
- **Space-by-Space Occupancy Types**
- **Whole Building Occupancy Types**

alongside any folders you've added yourself under **Library Locations** in Application Settings. *(see "The EP3 Interface and Model Settings")*

Whatever you select here, the Wizard also creates (if it doesn't already exist) a dynamic `ZoneList` named **"All"** that automatically keeps itself updated with every zone in the model — a default grouping independent of whichever space-type templates you pick.

Fill in this tab before Building Geometry Properties (§7): the Perimeter Space Type and Core Space Type dropdowns on that tab are populated from whatever ZoneList/space-type names you've checked here, so those dropdowns are empty until you've made selections on this tab.

(see "Internal Loads, Space Types, and Zone Types") covers creating your own reusable, linked `SpaceType`/`ZoneType` templates and the full field set they carry, including the cascading-defaults pattern these bundles participate in; *(see "Schedules")* covers the schedules bundled in; *(see "Sizing HVAC Systems and Zones")* covers the `Sizing:Zone` fields that come along with each template.

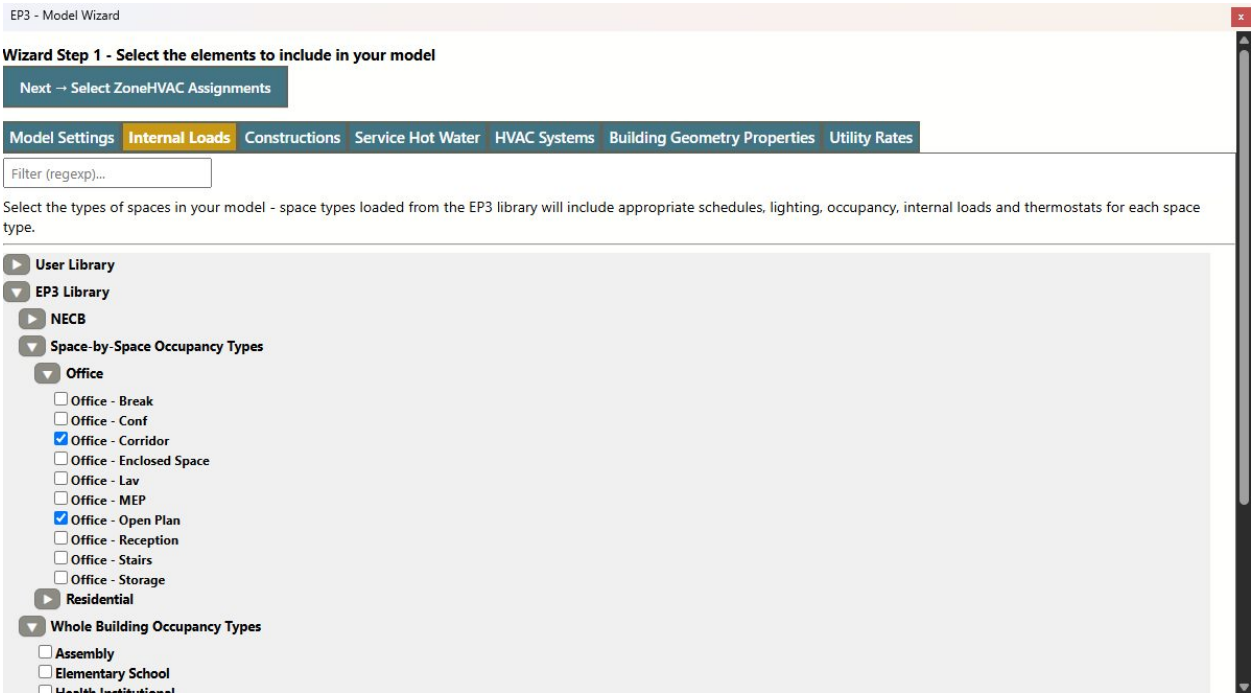


Figure 16: The Internal Loads tab, showing the Space-by-Space Occupancy Types > Office folder expanded.

---

4. Constructions

Use this tab to select constructions to use in this project. The EP3 library is organized by code baseline:

- **ASHRAE90_1 2022**
- **ASHRAE90_1 AppG Baseline**

each split into climate-zone subfolders (CZ0 through CZ8).

If you have any user-libraries for constructions, these will be listed under **User Library**.

If you don't select any constructions here, EP3 still creates one for each of the model's required surface types — but these are incomplete. They carry no material layers at all, and the model cannot be simulated until you fill in real materials for each one, whether by returning to this tab's library later (§10) or building them out directly. *(see "The EP3 Interface and Model Settings")* covers the Default Constructions surface-type list (Exterior Walls, Interior Walls, Roof,

Glazing, Interior Floor & Ceiling, Underground Floor, Underground Walls, Exposed Floor) that these incomplete defaults fill in.

(see "Constructions and Materials") covers building your own reusable construction/material templates and the full assembly/material field set.

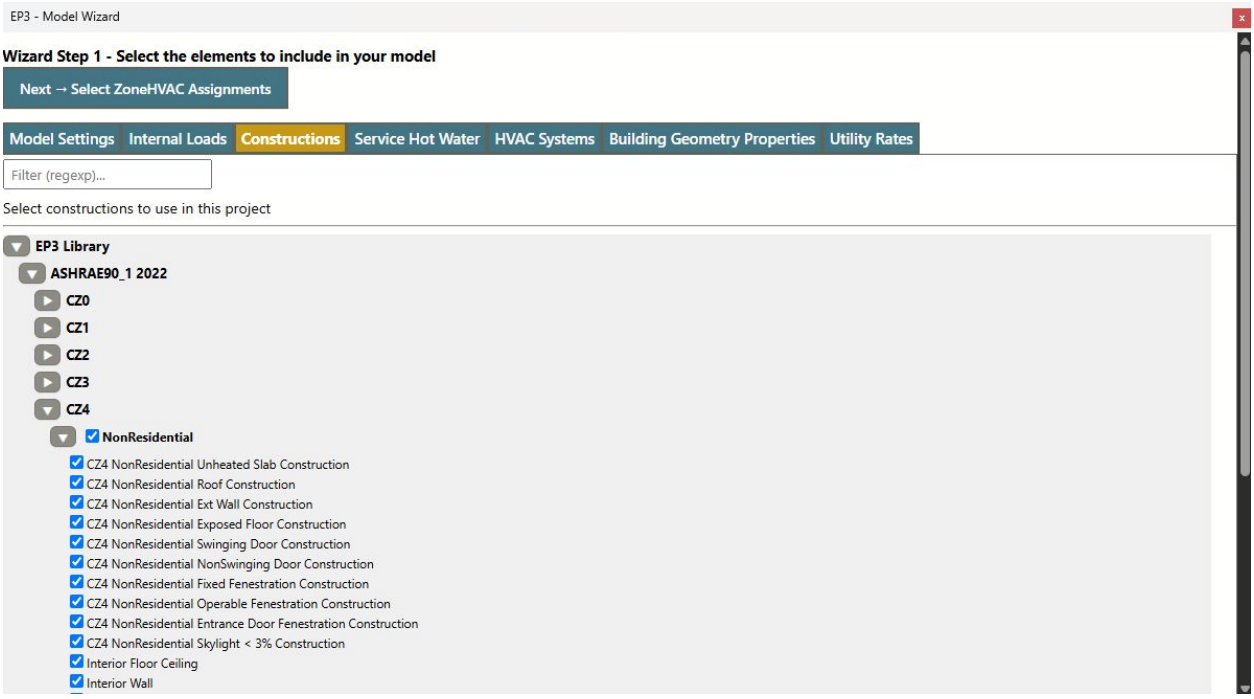


Figure 17: The Constructions tab, showing ASHRAE90_1 2022 > CZ4 > NonResidential selected.

The next two tabs, Service Hot Water and HVAC Systems, only appear if your EP3 license includes HVAC. Without it, the Wizard ends after Constructions.

5. Service Hot Water

Use this tab to select a Service Water Heater for the building. Its library, `default_swh_systems`, currently offers one complete system — **"WaterHeater with Detailed Connections"** — with its piping already built.

Each system you import becomes its own Energy Efficiency Measure ("Alternative"), tagged as a Service Water Heating measure, sitting on the model's Global EEM. *(see "Energy Efficiency Measures (EEMs)")* covers what an EEM is and how measures are organized.

Skipping this tab doesn't remove water use from your model — it removes what heats it. The `WaterUse:Equipment` objects bundled into your space-type templates (§3) generate domestic water use regardless of what you select here. Without a water heater object to serve it, that water's heating load simply goes unmet — you'll still see a water end use in results, but it's the raw domestic water volume, not hot water; none of it has actually been heated.

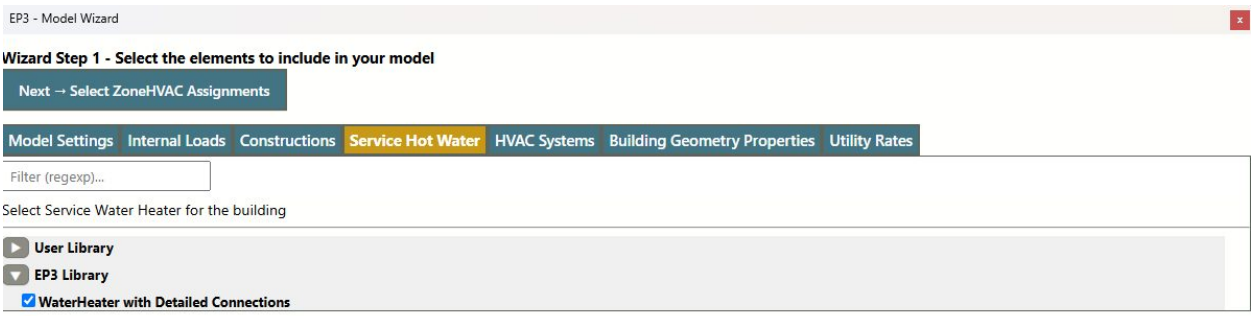


Figure 18: The Service Hot Water tab, showing WaterHeater with Detailed Connections selected.

6. HVAC Systems

Use this tab to select HVAC systems for the building. Its library, `default_hvac_systems`, offers a range of complete prototype systems — packaged rooftop units (PSZ-AC, PSZ-HP), PTAC/PTHP, VRV, Fan Coil, Packaged VAV (with PFP or reheat), DOAS with water coils, Geothermal Zone Heat Pump, HW Baseboard, ZoneHVAC Water to Air Heat Pumps, an airhandler with gas furnace, and AirLoopHVAC Water Coils VAV Reheat — plus an `ashrae_baseline` subfolder holding the ASHRAE 90.1 Appendix G baseline system templates (SYS-1 through SYS-13).

Every performance curve and schedule needed by a system template — `Curve:Biquadratic`, `Curve:Quadratic`, `Schedule:Compact`, is included with the equipment template. *(see "Performance Curves: Part-Load and Efficiency Behavior")* covers how those curves work once imported.

Every HVAC system you select is added to the model, not swapped in for a previous selection. Like Service Hot Water systems, each one becomes its own EEM, tagged as an HVAC measure — so selecting three different system types across a few Wizard sessions leaves all three living in the model side by side, each as its own EEM, ready to be compared against one another, or combined for more complex HVAC system architecture. Each HVAC EEM this tab creates is also automatically paired with a Service Water Heating EEM, so an HVAC-vs-HVAC comparison carries its own SWH comparison along with it. *(see "Energy Efficiency Measures (EEMs)")* covers how to combine an HVAC EEM with other EEMs into a single model for simulation.

(see "The HVAC Outliner and Creating HVAC Systems") covers prototype/template systems versus building from individual components in full, including system-type selection guidance; *(see "Zone-Level HVAC Equipment")* covers the zone-level equipment objects these templates bring in.

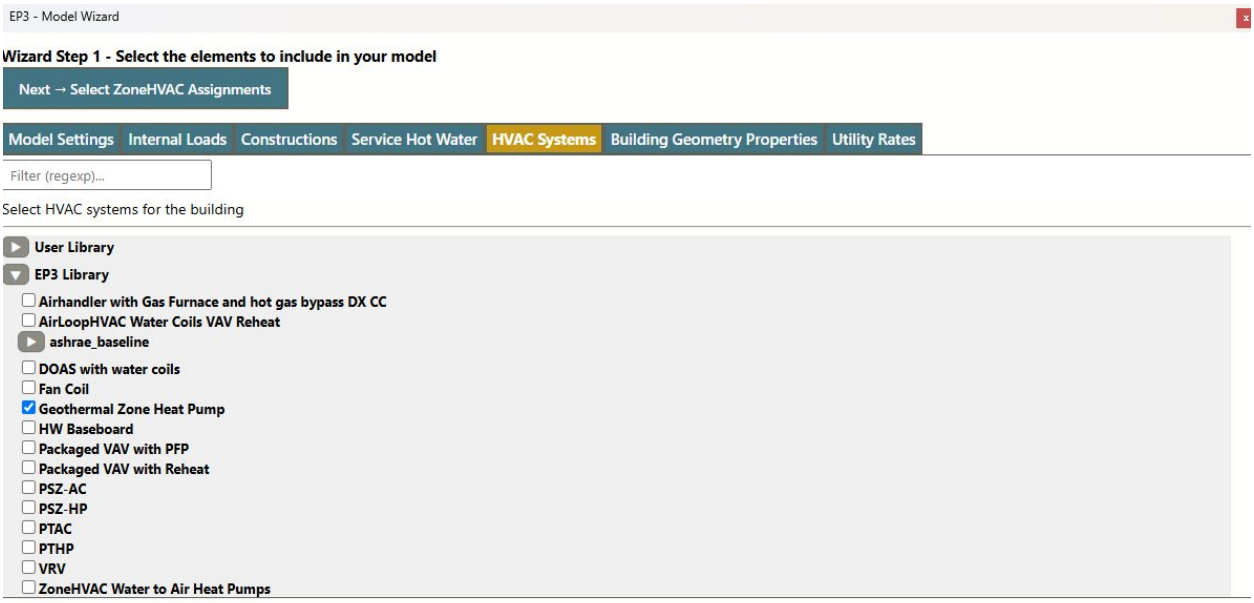


Figure 19: The HVAC Systems tab, showing the ashrae_baseline folder alongside the standard system checkboxes.

7. Building Geometry Properties

This tab offers two ways to get your building's geometry into the model, selected under **"Choose Your Building Drawing Method:"**:

- **"Let EP3 draw a rectangular building for me"**
- **"I want to draw my own building geometry"**

Choosing the second option skips the rest of this tab entirely; the geometry-drawing fields below only apply to the first.

When the automatic option is useful. Because a construction, internal-loads, or HVAC template only needs to be built once and can be reused across any number of models, the automatic rectangular-building option lets you test a system or object you haven't used before without first building geometry from scratch by hand. A simple box model — a perimeter/core split is enough to be representative of a real building — combined with the templates already selected in the tabs above, gets you to a working, simulate-able model in a couple of minutes.

7.1 Fields

- **Building Width**, **Building Depth** — the building's East-West and North-South dimensions.
- **Number of Floors** — 0 creates no geometry at all.
- **Floor to Floor Height**, **Plenum Height**, **Elevation of Lowest Floor**, **Building North Axis**
- **Roof Height Above Top Floor**, **Roof Overhang**, **Roof Pitch**
- **Window to Wall Ratio**

- **Perimeter Space Depth** — see §7.2.
- **Perimeter Space Type**, **Core Space Type** — populated from whatever space-type/ZoneList names you selected on the Internal Loads tab (§3).

7.2 The perimeter/core split

Entering a **Perimeter Space Depth** greater than 0, and less than half the smaller of Building Width and Building Depth, splits each floor into five spaces: one perimeter space along each of the four sides of the building (each assigned the Perimeter Space Type) and one core space in the middle (assigned the Core Space Type) — the "core and shell" split. A depth of 0, or a depth at or beyond that half-width threshold, produces no split at all: one space per floor, using the Perimeter Space Type.

After the floors are created, the Wizard also builds a roof (flat or gabled, based on Roof Pitch and Roof Overhang), grows exterior walls to enclose the building, and adds windows according to the Window to Wall Ratio.

(see "Creating Building Geometry") and *(see "Spaces and Zones")* cover building geometry and thermal-block strategy for models built by hand, beyond what this automatic path produces.

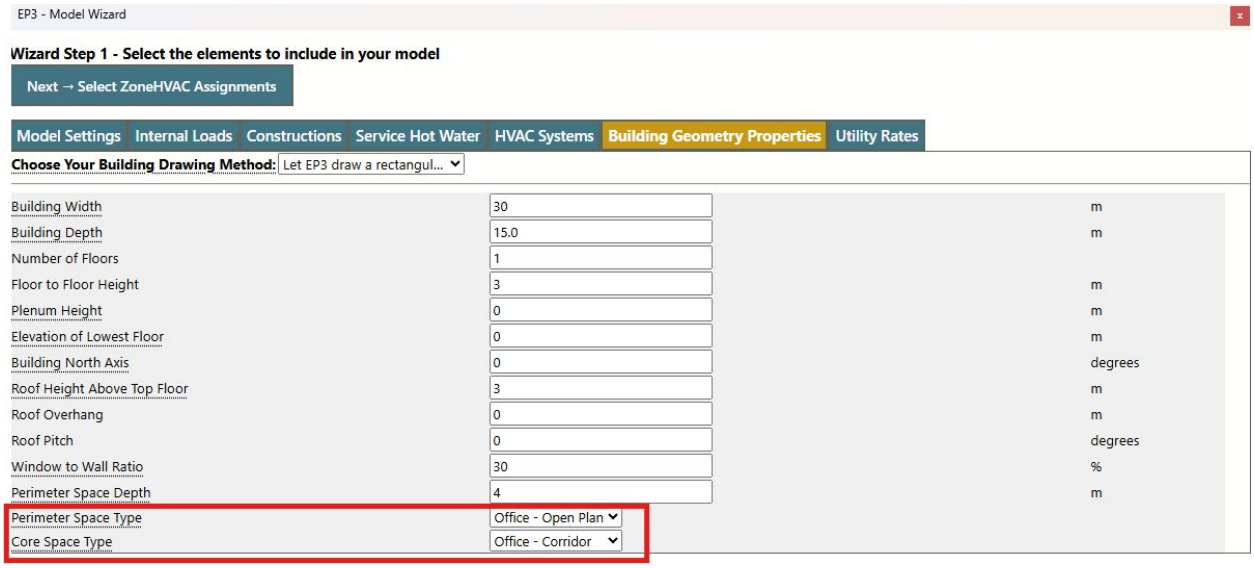


Figure 20: The Building Geometry Properties tab with "Let EP3 draw a rectangular building for me" selected, Perimeter Space Type and Core Space Type fields highlighted.

---

8. Utility Rates

The Utility Rates tab lets you set a Monetary Unit, then a flat Monthly Charge, per-unit energy charge, and per-unit demand charge for each of Electric Rate, Gas Rate, and Water Rate. Whatever you enter here creates a `UtilityCost:Tariff` object per meter — the same underlying object the always-available Utility Cost Manager uses, just reached through a faster path at model-creation time. Leaving all three charge fields at 0 for a meter skips creating a tariff for it. *(see "Utility Costs")* covers the full Utility Cost Manager and every field a tariff supports beyond this flat-rate shortcut.

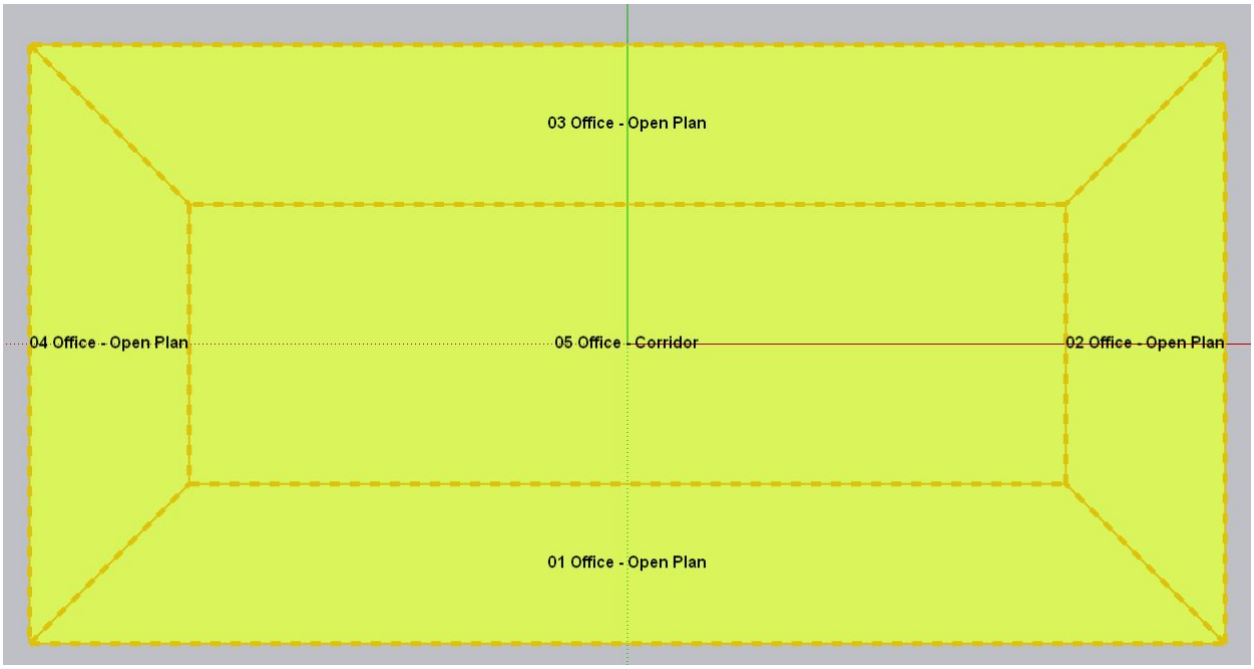


Figure 21: The resulting floor plate: four perimeter spaces (one per side, all "Office - Open Plan") wrapped around one core space ("Office - Corridor").

EP3 - Model Wizard

Wizard Step 1 - Select the elements to include in your model

Next → Select ZoneHVAC Assignments

Model Settings | Internal Loads | Constructions | Service Hot Water | HVAC Systems | Building Geometry Properties | Utility Rates

Monetary Uinit	EUR
Electric Rate	
Monthly Electricity Charge	20.0
Electricity Charge per kWh	0.2
Electricity Charge per kW	15.0
Gas Rate	
Monthly Gas Charge	0
Gas Charge per MJ	0
Gas Charge per MJ/h	0
Water Rate	
Monthly Water Charge	0
Water Charge per m3	0
Water Charge per m3/h	0

Figure 22: The Utility Rates tab, with an Electric Rate entered and Gas/Water Rates left at 0.

---

9. After the Wizard: Assigning Zone Conditioning Equipment

Clicking **"Next → Select ZoneHVAC Assignments"** opens **EP3 - Select Zone Conditioning Equipment**, where whatever `ZoneHVAC` equipment you selected gets assigned to actual zones. It has two tabs:

- **Assign HVAC Systems by Template** — one row per `Sizing:Zone` object (that is, per space-type template from §3), for assigning equipment to every zone that template covers at once.
- **Assign HVAC Systems by Zone** — one row per individual zone, for the occasional zone that needs to deviate from its template's normal equipment assignment.

Only zones with a thermostat appear in either tab; zones without one are listed separately as informational text rather than given a row. The dialog also flags two conditions as likely fatal EnergyPlus errors if left unresolved: a zone with a thermostat but no `Sizing:Zone` object, and a zone covered by more than one `Sizing:Zone` object (usually from being assigned to more than one ZoneList).

9.1 Equipment and load distribution

Each row shows the equipment assigned, alongside a **Load Distribution Scheme** — `SequentialLoad`, `UniformLoad`, `UniformPLR`, or `SequentialUniformPLR` — controlling how a zone's remaining load is shared across its equipment whenever more than one piece is assigned to the same zone or template.

Where a zone or template has more than one piece of equipment, order matters: EnergyPlus works down the list, calling on each successive piece only to make up whatever load the equipment ahead of it didn't meet. Order equipment from least controllable to most controllable, so the equipment best able to modulate its own output is the one left to true up whatever the simpler equipment ahead of it couldn't fully address. An **"Update Heating Order"** button manages this same priority specifically for heating, independent of the general (cooling) order shown on the row.

(see "Zone-Level HVAC Equipment") covers the individual equipment objects assigned here; *(see "Sizing HVAC Systems and Zones")* covers `Sizing:Zone`; *(see "Model Troubleshooting and Quality Assurance")* covers resolving the fatal-error conditions this dialog flags.

---

10. Reusing the Wizard's Template Picker on an Existing Model

The same dialog also opens, in a reduced form, to import additional templates into a model you've already set up — titled **"Select Templates / Objects to import"** with an **"Import selected"** button in place of the Model Wizard's own title and Next button. This is the same checkbox-tree library browsing described in §3–§6, without the Model Settings, Building Geometry, or Utility Rates tabs. *(see "Import, Export, and Interoperability")* covers reaching this from the Main Dialog's Import button.

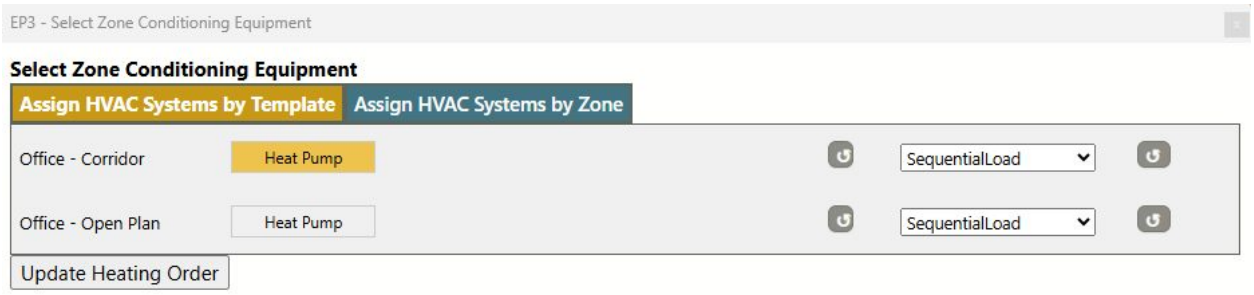


Figure 23: The Select Zone Conditioning Equipment dialog, "Assign HVAC Systems by Template" tab, showing one row per space-type template with its equipment, Load Distribution Scheme, and the Update Heating Order button.

11. A Worked Example — From the Wizard to a Running Simulation

This example carries a model from a blank SketchUp file through to a finished simulation using only the Wizard — no hand-built geometry, no detailed-mode refinement afterward. It's deliberately small, just enough to show every tab from §§2–9 connecting to the next. *(see "A Complete Worked Example, End to End")* picks up a larger, more realistic building and works through detailed mode in full; this example's only job is to make the Wizard path concrete, once, with real selections.

11.1 The example building

A single-story, 60 ft × 40 ft rectangular office building, with a 15 ft deep perimeter zone wrapped around a core, one packaged rooftop unit conditioning each zone, in Climate Zone 4A.

Note - this example uses IP units, select a Sketchup template that uses feet and inches in order to follow this example exactly

11.2 Filling in the Wizard

1. **Internal Loads (§3)** — under Whole Building Occupancy Types, check **"Office Building - General."** It's the only space type this building uses, so it'll be the only option available for both Perimeter Space Type and Core Space Type once you reach Building Geometry Properties.
2. **Constructions (§4)** — under ASHRAE90_1 2022 → CZ4, check **"NonResidential."**
3. **HVAC Systems (§6)** — check **"PSZ-AC"** (Packaged Single Zone Air Conditioner) — a simple system that puts one packaged unit on each zone, a natural fit for a small building like this one. Its curves and schedules come along automatically. No Service Hot Water system is selected in this example, so the resulting HVAC EEM parents directly to Global rather than to a paired SWH EEM (§5, §6).
4. **Model Settings (§2)** — select a weather file for Climate Zone 4A, and set Select Climate Zone to match.
5. **Building Geometry Properties (§7)** — choose **"Let EP3 draw a rectangular build-**

ing for me," then set: Building Width 60 ft, Building Depth 40 ft, Number of Floors 1, Perimeter Space Depth 15 ft (well under half the 40 ft depth, so the perimeter/core split applies — §7.2), and both Perimeter Space Type and Core Space Type set to **"Office Building - General."** Leave every other field at its default.

6. **Utility Rates (§8)** — left blank; this example isn't tracking utility cost.

With those tabs filled in, click **"Next → Select ZoneHVAC Assignments."**

11.3 Assigning equipment and simulating

Because this example has only one space-type template and one HVAC system, the Select Zone Conditioning Equipment dialog (§9) has one thing left to do: assign the PSZ-AC units to the zones on the **"Assign HVAC Systems by Template"** tab, rather than working zone by zone. With that assignment made, the model is complete enough to run.

From here, running the simulation and reviewing what it produced work exactly as they would for any other model. *(see "Running Simulations")* and *(see "Viewing and Interpreting Results")* cover both in full. For this example, a completed run confirms the whole chain end to end: the five zones created in §7 (01–04 the perimeter, 05 the core, all "Office Building - General") carry the loads and schedules selected in §3, conditioned by the PSZ-AC equipment selected in §6, run against the weather selected in §2. The Annual Building Utility Performance Summary's End Uses table, below, shows the result under the **"PSZ-AC"** EEM this HVAC selection created (§6): cooling, heating, lighting, and equipment electricity and gas broken out by end use.

The table also shows a **Water Systems** end use, even though this example never selected a Service Hot Water system. That's expected, not an error: the `WaterUse:Equipment` objects bundled into "Office Building - General" (§3) generate domestic water use regardless, and with no water heater in the model to serve it (§5), that figure is the raw domestic water volume rather than heated water — none of it has actually been heated.

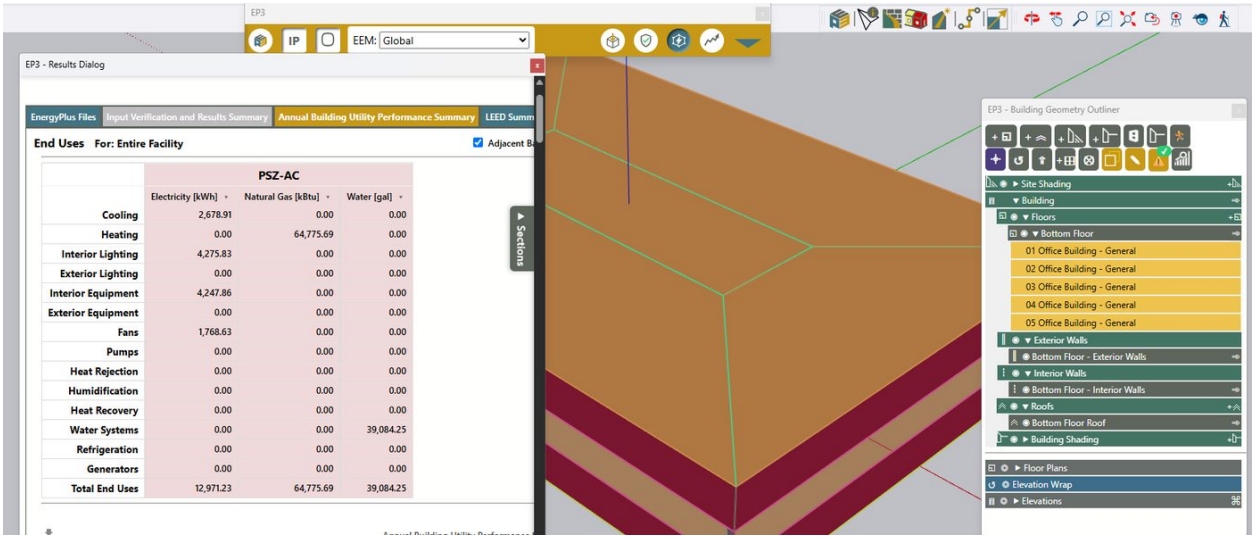


Figure 24: The finished example: five "Office Building - General" spaces in the Building Geometry Outliner, and the resulting Annual Building Utility Performance Summary End Uses table for the PSZ-AC EEM.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF placeholders once their target sections/pages exist.

4. Weather, Climate, and Design Conditions

1. Overview

Every EP3 model draws on climate data in two distinct ways: an annual simulation runs against a full year (or a chosen slice of one) of hourly weather from an assigned `.epw` file, while sizing calculations run against a handful of `SizingPeriod` objects representing extreme or statistically-defined single days. They are configured through different objects and serve different purposes — the annual run predicts energy use and comfort over time; the design-day runs size equipment to handle worst-case conditions without materially oversizing for conditions that occur only a few hours a year.

This chapter covers assigning a weather file, entering `Site:Location` data, building the `SizingPeriod` objects that establish real sizing conditions, and the `RunPeriod` object that bounds the annual simulation. For what `Sizing:Zone`, `Sizing:System` and `Sizing:Plant` do with the results, see *(see "Sizing HVAC Systems and Zones")*; for how a full annual load profile is built from these same weather inputs, see *(see "Heating and Cooling Loads: From Design Day to Annual Profile")*.

---

2. Assigning a Weather File

2.1 The per-model weather file

Every model has exactly one primary weather file, set from **Model Settings → General Model Settings → Weather File**. The button shows the current file's name (or "No Weather File Selected for the Project" if none is set); clicking it opens a standard file browser filtered to `.epw` files. EP3 stores the full path on the model, not just the filename — if you later move or rename the `.epw` file, or open the model on a different machine where it lives at a different path, you'll need to reassign it.

EP3 remembers the last folder you browsed for a weather file and reopens the picker there next time. This is separate from the Weather File Library Locations setting described next.

2.2 Weather File Library Locations

Application Settings has a **Weather File Library Locations** entry — an editable list of folders, with Add/Remove/reorder controls, alongside similar location lists for materials, internal loads, HVAC templates, and reports. It also carries a **Download Weather File** button that opens the EnergyPlus weather-file map (`energyplus.net/weather`) in-app so you can find and download a `.epw` for your project's location.

Unlike the Materials, HVAC, and Internal Loads library locations — which EP3 searches by name when resolving a template file — the Weather File Library Locations list isn't searched for candidate files by name; a weather file is still picked by browsing, not looked up automatically. What it does do: the Weather File picker (§2.1, and the per-EEM override picker, §2.4) opens starting in the first folder on this list, if one is configured, rather than wherever you last happened to browse. Add the folder(s) where you keep `.epw` files here so the picker starts somewhere useful.

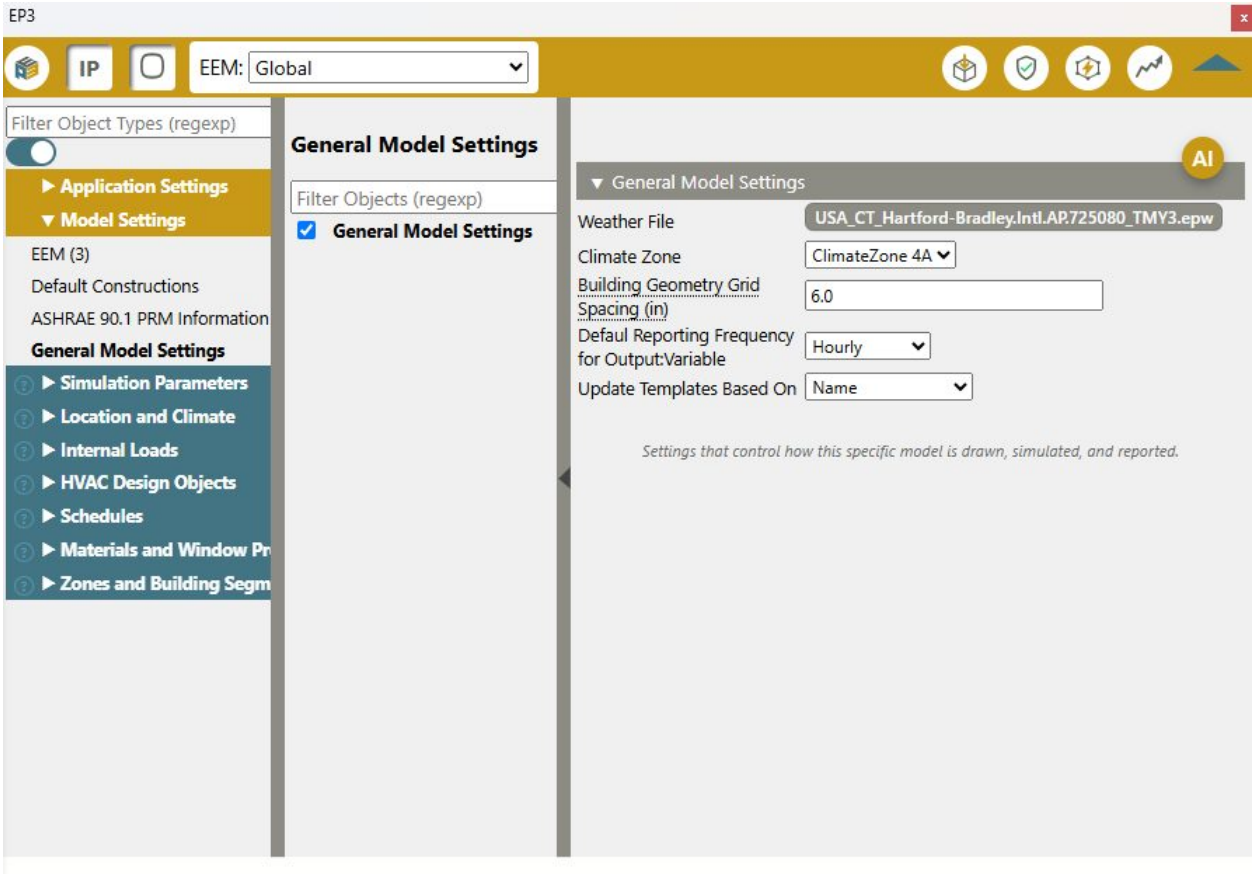


Figure 25: The General Model Settings panel, with the assigned Weather File and the Climate Zone dropdown just below it.

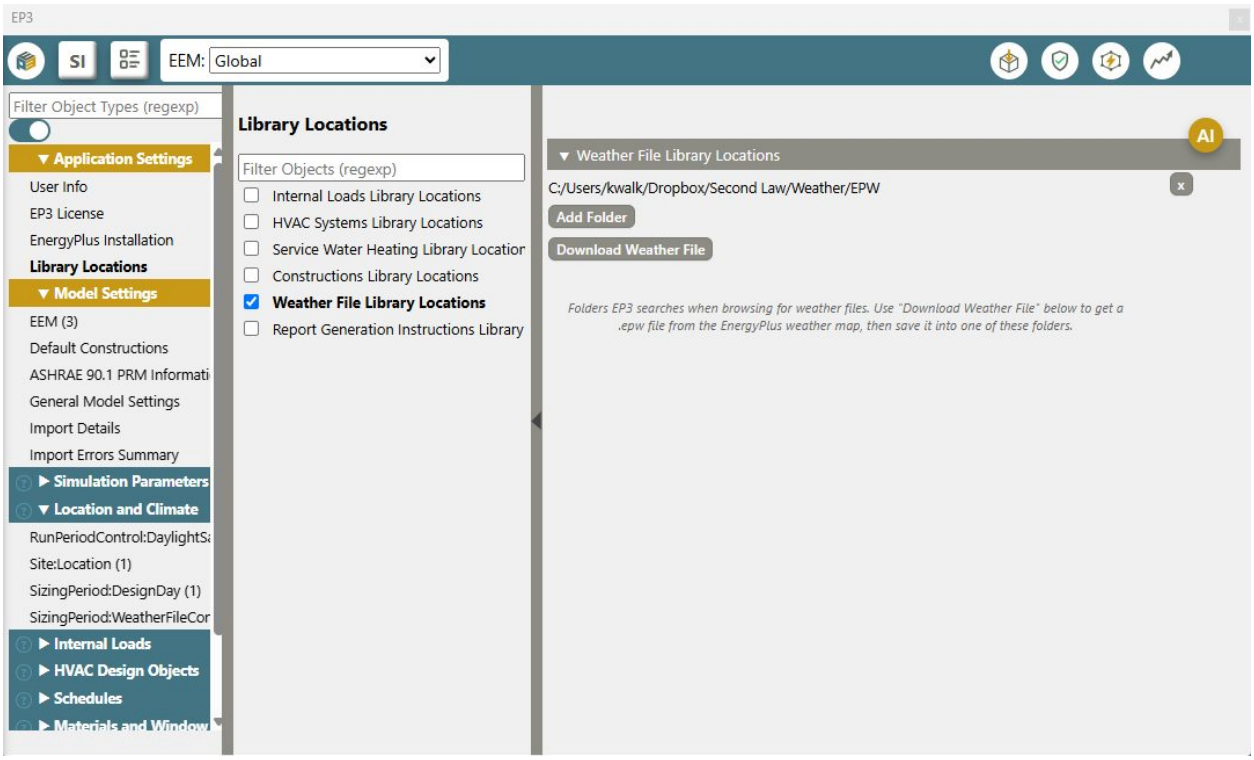


Figure 26: The Weather File Library Locations panel in Application Settings, showing one configured folder and the Download Weather File button.

2.3 Weather file required before running

If you attempt to export/run a model with no weather file assigned, EP3 forces the Weather File picker to appear before the run can proceed — you cannot export without one.

2.4 Per-EEM weather overrides

An individual EEM can override the model's global weather file with its own — useful for comparing a project against multiple climate scenarios, or for a proposed/baseline pair that must reference different weather data. An EEM without its own override inherits the weather file from its base chain, ultimately falling back to the model's global assignment if nothing in the chain overrides it. See *(see "Energy Efficiency Measures (EEMs)")* for how EEM scenarios and base chains work.

2.5 Weather file assignment also updates the model's shadow/geolocation

The first time SketchUp's geometry is accessed after a weather file is assigned or changed, EP3 reads the `.epw` file's header (its `LOCATION` line: city, country, latitude, longitude) and, if it differs meaningfully from the model's current setting, updates SketchUp's own shadow-study geolocation to match — the same georeferencing SketchUp uses to compute sun position and cast shadows in the 3D view. This is independent of the EnergyPlus `Site:Location` object covered next; it affects only SketchUp's own solar/shadow rendering, not the simulation.

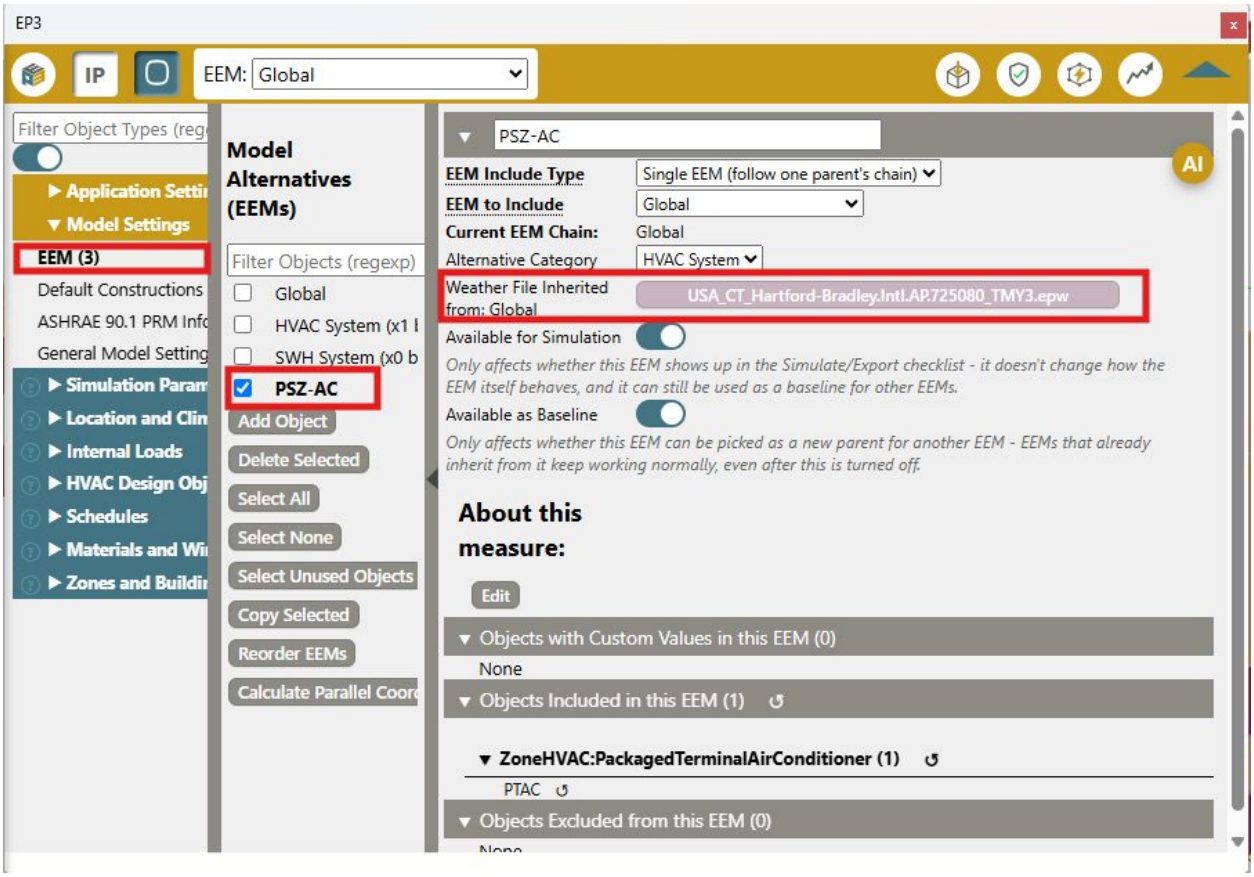


Figure 27: An EEM's Weather File field, here showing "Weather File Inherited from: Global" — this EEM has no override of its own and is using the model's global weather file.

3. Site Location Data (`Site:Location`)

`Site:Location` carries the latitude, longitude, time zone, and elevation EnergyPlus uses for solar position. EP3 does not create this object by default: if no `Site:Location` object exists in the model, EnergyPlus uses the assigned weather file's own location data automatically — no action needed. Only create a `Site:Location` object if you have a specific reason to override the weather file's location (e.g., a design point genuinely different from the weather station the `.epw` represents).

Field	Notes
Latitude	+ is North, - is South
Longitude	- is West, + is East
Time Zone	Relative to GMT
Elevation	
Keep Site Location Information	See below

If you do create the object, also set **Keep Site Location Information** to **Yes** — left at its default of **No**, EnergyPlus overwrites whatever you've entered with the weather file's own header data at simulation time anyway, silently defeating the override.

> **Caution:** EP3 does not keep a `Site:Location` object in sync with later changes to the model's weather file. If you create one to override the weather file's location and later reassign the model to a different weather file, the `Site:Location` object still holds the old location — producing a mismatch between the site data EnergyPlus uses for solar position and the climate actually being simulated. EnergyPlus will flag the discrepancy in the error file, but EP3 does not at this time. Delete or update the object yourself whenever the weather file changes, if you're using this override.

Worked example — the `.epw`/`.ddy` weather-file set for Burlington International Airport, VT, carries this header in its companion `.ddy` file:

```
Site:Location,
 Burlington International Ap_VT_USA Design_Conditions,
     44.47,     !- Latitude {N+ S-}
    -73.15,     !- Longitude {W- E+}
     -5.00,     !- Time Zone Relative to GMT {GMT+/-}
    101.00;     !- Elevation {m}
```

If you had reason to override the weather file's location with these values explicitly — Latitude 44.47, Longitude -73.15, Time Zone -5.00, Elevation 101.00 m (~ 331.4 ft, for reference if displaying in IP units) — create a `Site:Location` object, enter them, and set Keep Site Location Information to Yes. Absent a reason to override, leave no `Site:Location` object in the model at all and let the weather file govern.

`Site:VariableLocation` is a related, advanced object for modeling a building whose location changes during the simulation (e.g., a mobile or marine application) via schedules rather than a fixed point; it's out of scope here as a rare use case. Using this object would also require a custom weather file weather matching the location as the variable location changes.

---

4. Sizing Periods and Design Days

4.1 The three `SizingPeriod:*` object types

EP3 exposes all three EnergyPlus `SizingPeriod:*` object types, grouped together with `Site:Location` under **Location and Climate**:

- **`SizingPeriod:DesignDay`** — a single day fully defined by explicit values (dry-bulb, humidity, wind, solar, etc.) — the object used for standard ASHRAE design conditions.
- **`SizingPeriod:WeatherFileConditionType`** — a period EnergyPlus derives statistically from whichever `.epw` is assigned at runtime (e.g., "Extreme Summer," "Summer Typical").
- **`SizingPeriod:WeatherFileDays`** — a literal date range pulled directly from the assigned weather file's own recorded data.

All three, along with `RunPeriodControl:SpecialDays` and `RunPeriodControl:DaylightSavingTime`, are edited through the same generic object-list dialog for the "Location and Climate" group — there's no dedicated design-day screen.

4.2 What's there by default

Every new model automatically gets two `SizingPeriod:WeatherFileConditionType` objects — "Extreme Summer Weather Period for Design" (Period Selection = `SummerExtreme`) and "Extreme Winter Weather Period for Design" (Period Selection = `WinterExtreme`) — created without any action on your part. These tell EnergyPlus to derive an extreme summer/winter period statistically from whichever `.epw` ends up assigned; they are not fixed values and update automatically if you change the weather file.

`SummerExtreme`/`WinterExtreme` is one of several period-selection options EnergyPlus offers for `SizingPeriod:WeatherFileConditionType`; others include `SummerTypical`/`WinterTypical`, `WetSeason`/`DrySeason`, and `TropicalHot`/`TropicalCold`. A project in a hot-humid or tropical climate, where the governing load is driven more by humidity/wet-season conditions than by a dry-bulb temperature extreme, may be better represented by `WetSeason` or `TropicalHot` than by the default `SummerExtreme` — change the Period Selection field on the default object, or add another `SizingPeriod:WeatherFileConditionType` object with the period selection that fits the project's climate.

ASHRAE's 99.6% heating and 0.4%/1%/2% cooling design conditions (§4.3) are a separate, standard basis, entered as `SizingPeriod:DesignDay` objects. Which basis a given project's sizing should use is a design decision — add `SizingPeriod:DesignDay` objects per §4.3 whenever a project calls for ASHRAE design-condition sizing alongside or instead of the defaults.

4.3 Getting real ASHRAE design conditions into your model

ASHRAE Handbook — Fundamentals, Chapter 14 ("Climate Design Data"), publishes standard design conditions — most commonly the 99.6% heating dry-bulb and the 0.4%/1%/2% cooling dry-bulb (with mean coincident wet-bulb) — for thousands of weather stations worldwide. ASHRAE Handbook—Fundamentals, Climate Design Data There are two ways to get these into an EP3 model as `SizingPeriod:DesignDay` objects.

4.3.1 Manual entry, worked example Suppose your project's location is Burlington, VT, and your source table (or the `.ddy` file described in §4.3.2) gives:

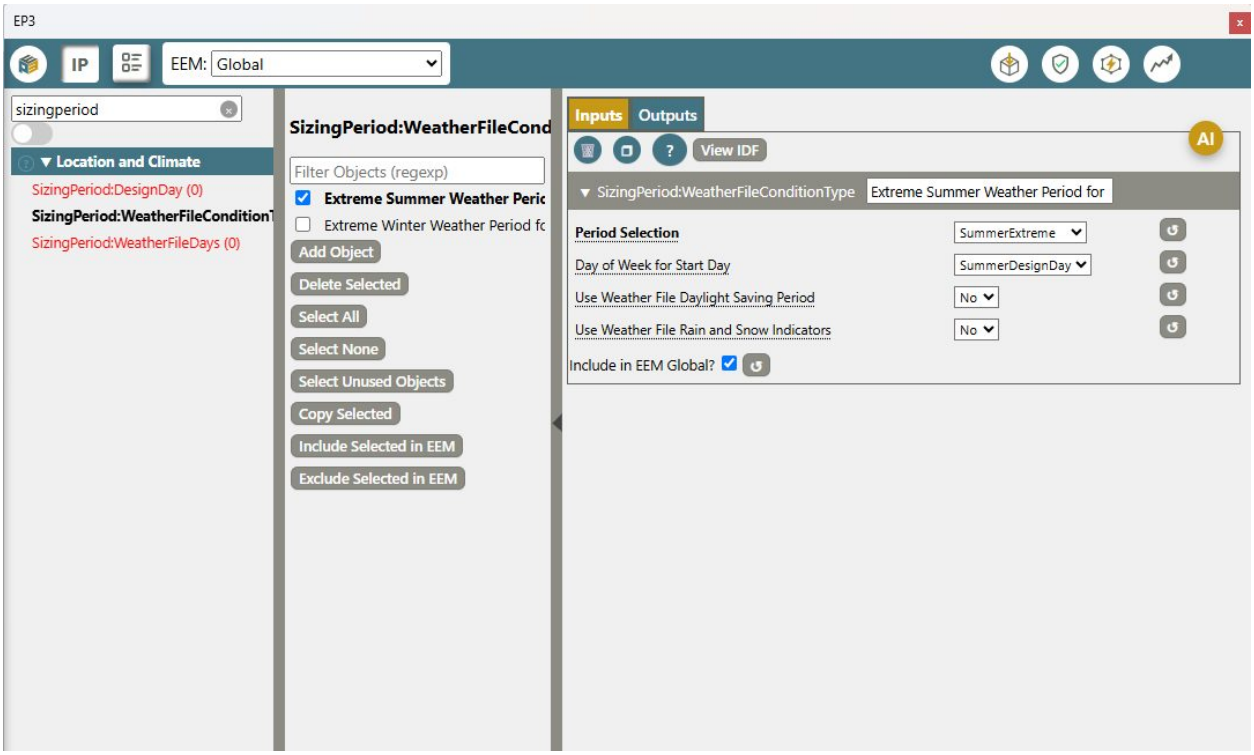


Figure 28: The "Location and Climate" object list, filtered to "sizingperiod," showing the two default SizingPeriod:WeatherFileConditionType objects alongside the (empty) SizingPeriod:DesignDay and SizingPeriod:WeatherFileDays types. The selected "Extreme Summer Weather Period" object shows its Period Selection field set to SummerExtreme.

Condition	Value (SI)	Value (IP)
99.6% Heating Design Dry-Bulb	-22.4°C	-8.3°F
0.4% Cooling Design Dry-Bulb / Mean Coincident Wet-Bulb	31.3°C / 21.7°C	88.3°F / 71.1°F
Daily cooling dry-bulb range (mean coincident)	10.7°C	19.3°F

(°F = °C × 1.8 + 32 for absolute temperatures; a range/delta converts by ×1.8 alone, with no offset — e.g. 10.7°C × 1.8 = 19.3°F.)

Create two `SizingPeriod:DesignDay` objects and populate them as follows. Fields not listed keep their defaults (Rain Indicator/Snow Indicator = No, Solar Model Indicator = `ASHRAEClearSky`, etc.) unless your source data says otherwise.

Winter design day (named "Burlington VT Htg 99.6% Condns DB" below):

Field	Value
Month / Day of Month	1 / 21 *(the source table's associated date — ASHRAE design days are conventionally dated the 21st of the coldest/hottest month)*
Day Type	`WinterDesignDay`
Maximum Dry-Bulb Temperature	-22.4°C
Daily Dry-Bulb Temperature Range	0.0°C *(winter design days conventionally assume no diurnal swing)*
Humidity Condition Type	`WetBulb`
Wetbulb or DewPoint at Maximum Dry-Bulb	-23.7°C *(the coincident wet-bulb value for this station's heating design condition — check your own source table rather than assuming it always equals the dry-bulb value)*
Barometric Pressure	97,272 Pa *(the station's elevation-adjusted standard pressure — see §4.4)*
Wind Speed / Wind Direction	2 m/s / 320° *(this project's assumed design wind — use your own source table's coincident wind data)*

Summer design day (named "Burlington VT Clg 0.4% Conds" below):

Field	Value
Month / Day of Month	7 / 21
Day Type	`SummerDesignDay`
Maximum Dry-Bulb Temperature	31.3°C
Daily Dry-Bulb Temperature Range	10.7°C
Humidity Condition Type	`WetBulb`
Wetbulb or DewPoint at Maximum Dry-Bulb	21.7°C *(the mean coincident wet-bulb)*

Field	Value
Barometric Pressure	left blank *(EnergyPlus calculates the standard pressure from `Site:Location`'s elevation when this field is blank — see §4.4; contrast with the winter design day above, where a specific value was entered)*
Wind Speed / Wind Direction	2 m/s / 180° *(this project's assumed design wind)*

4.3.2 Importing a `.ddy` file Most published EnergyPlus weather-file sets include a companion `.ddy` file — plain IDF text containing pre-built `SizingPeriod:DesignDay` objects with real ASHRAE design-condition values for that station, alongside a matching `Site:Location` object. It shares the weather file's base name (e.g. `USA_VT_Burlington.Intl.AP.726170_TMY3.epw` pairs with `USA_VT_Burlington.Intl.AP.726170_TMY3.ddy`) and is typically downloaded from the same source as the `.epw` itself.

Use **Import EnergyPlus File** and select the `.ddy` file directly — it now appears alongside `.idf`/`.epJSON` files in the file picker. The objects it contains import as ordinary `SizingPeriod:DesignDay` objects, no different from ones you create by hand. A typical `.ddy` file bundles several design-day variants per season (99.6%/99% heating; 0.4%/1%/2% cooling defined against dry-bulb, wet-bulb, dew-point, and enthalpy respectively) — after importing, delete whichever variants you don't need and keep the ones matching your project's design basis.

EnergyPlus-published `.ddy` files name objects predictably enough to identify the standard ones at a glance, e.g.:

- `<Station> Ann Htg 99.6% Condns DB` — the 99.6% heating design day
- `<Station> Ann Clg .4% Condns DB=>MWB` — the 0.4% cooling design day, defined dry-bulb-to-mean-coincident-wet-bulb

4.4 `SizingPeriod:DesignDay` field reference

Field	Purpose
Month, Day of Month, Day Type	Calendar placement and which zone/equipment schedules apply on this design day (`WinterDesignDay`/`SummerDesignDay`, or a weekday for a schedule-driven design day)
Maximum Dry-Bulb Temperature, Daily Dry-Bulb Temperature Range	The design condition and its 24-hour swing — see §4.3.1
Humidity Condition Type + its companion value field	Which second variable pins down humidity — wet-bulb, dew-point, humidity ratio, enthalpy, or a schedule
Barometric Pressure	Checked against the standard pressure computed from `Site:Location`'s elevation; out-of-range or blank values are replaced automatically

Field	Purpose
Wind Speed, Wind Direction	Design-day wind envelope — mainly affects infiltration-driven loads
Solar Model Indicator (+ ASHRAE Clear Sky Optical Depth / Sky Clearness / schedule fields, as applicable)	How solar radiation is generated for the design day — `ASHRAEClearSky` by default; `ASHRAETau`/`ASHRAETau2017` if you have taub/taud coefficients for the location
Rain Indicator, Snow Indicator, Daylight Saving Time Indicator	Day-long on/off flags

Full field-by-field detail, including every enumerated option, is in the EnergyPlus I/O Reference. SizingPeriod:DesignDay

---

5. `RunPeriod` — the Annual Simulation Window

`RunPeriod` bounds the date range of the annual simulation. It lives in the "Location and Climate" group alongside `Site:Location` and the `SizingPeriod:*` objects, but is controlled separately at the engine level via `SimulationControl` (§6). It doesn't reference the weather file by name — whichever `.epw` is passed to EnergyPlus at run time supplies the actual hourly data that the `RunPeriod` date range slices.

Field	Default	Notes
Begin Month / Begin Day of Month	—	Required
End Month / End Day of Month	—	Required
Begin Year / End Year	blank	If left blank, the day-of-week is used to select a matching year rather than pinning one
Day of Week for Start Day	—	Defaults to Sunday if no year is specified
Use Weather File Holidays and Special Days	Yes	
Use Weather File Daylight Saving Period	Yes	
Apply Weekend Holiday Rule	No	
Use Weather File Rain Indicators / Use Weather File Snow Indicators	Yes / Yes	
Treat Weather as Actual	No	
First Hour Interpolation Starting Values	Hour24	Governs interpolation when the weather data timestep is coarser than the simulation timestep

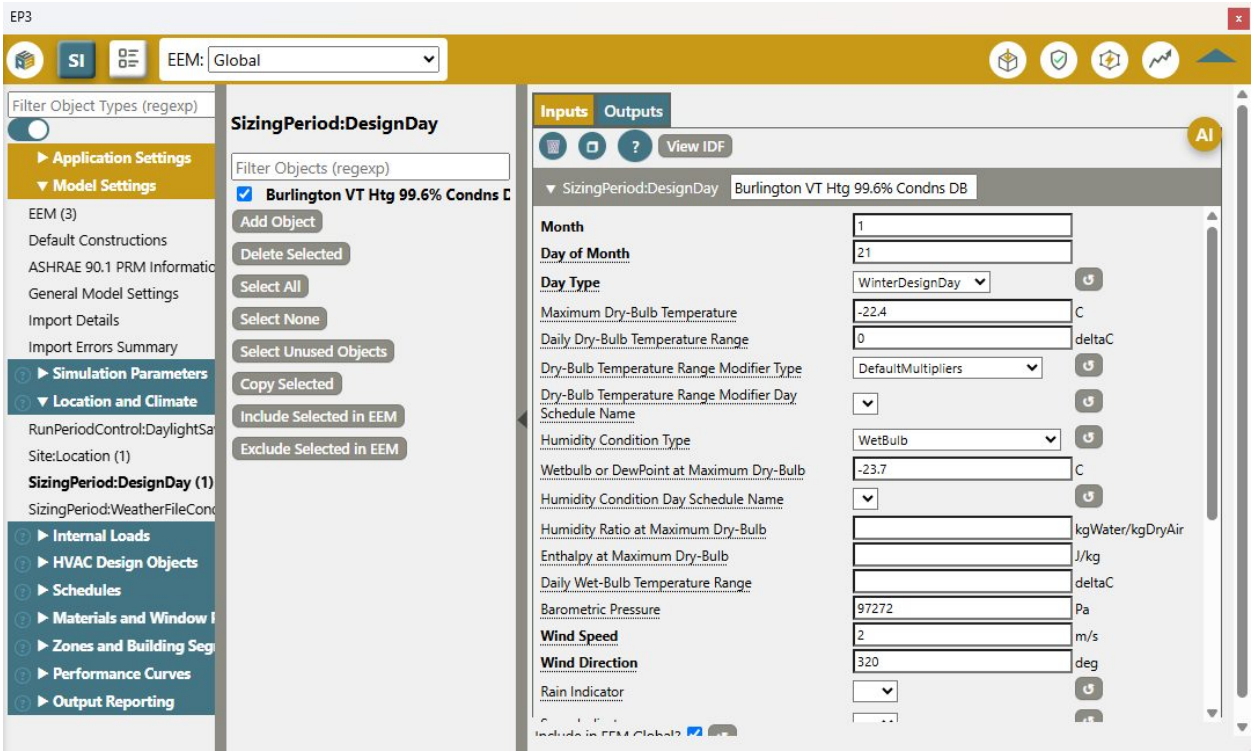


Figure 29: The winter design day above, entered as a `SizingPeriod:DesignDay` object.

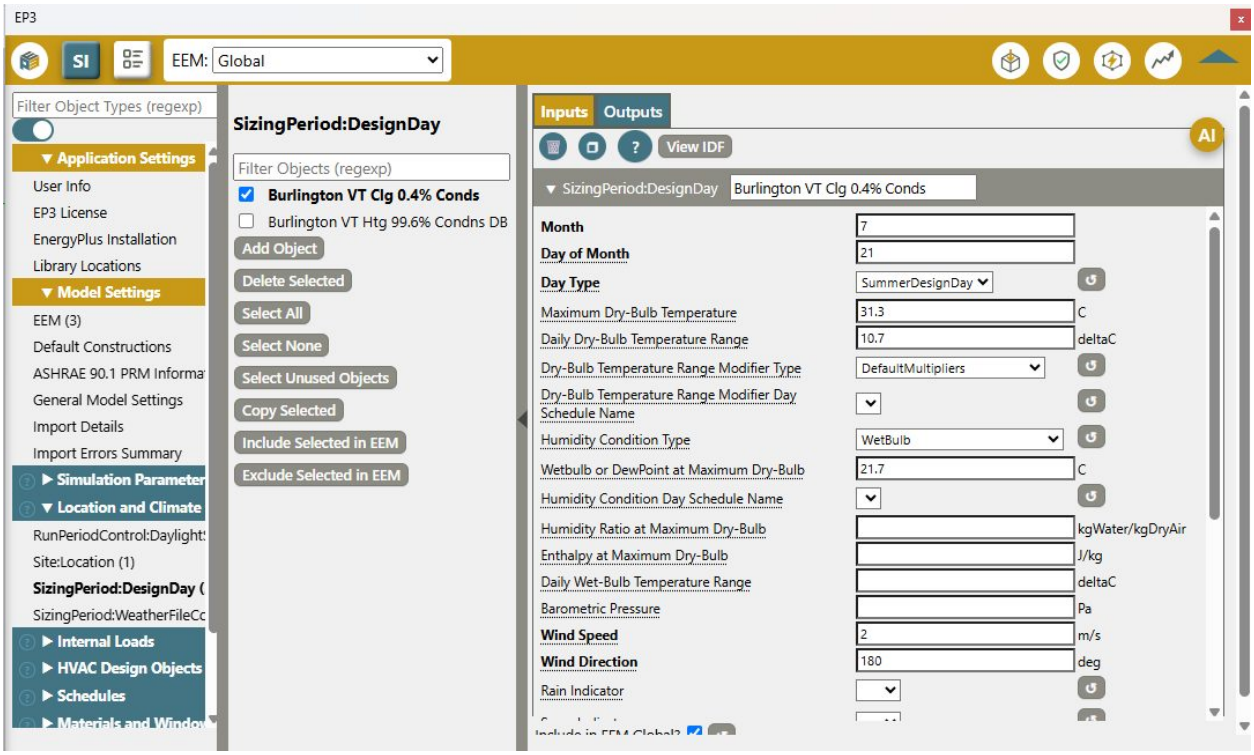


Figure 30: The summer design day above, entered as a `SizingPeriod:DesignDay` object.

More than one `RunPeriod` object is allowed, but is uncommon — the Model Wizard's Model Settings tab only ever renders the first one. Where the Wizard offers a quick way to set the common begin/end dates during guided setup, this is the full reference for every field.

Full field detail: RunPeriod

---

6. How Design Conditions Feed Sizing

Whether the objects covered above actually run, and what they're used for, is governed by `SimulationControl`:

Field	Default	Effect
Do Zone Sizing Calculation	No	Gates whether `Sizing:Zone` autosizing runs
Do System Sizing Calculation	No	Gates whether `Sizing:System` autosizing runs (requires Zone Sizing = Yes)
Do Plant Sizing Calculation	No	Gates whether `Sizing:Plant` autosizing runs
Run Simulation for Sizing Periods	Yes	Whether `SizingPeriod:*` objects execute at all
Run Simulation for Weather File Run Periods	Yes	Whether `RunPeriod:*` objects (the annual run) execute at all
Do HVAC Sizing Simulation for Sizing Periods	No	Advanced/coincident sizing, tied to `Sizing:Plant`

New EP3 models default `SimulationControl` to Yes for zone, system, and plant sizing, and Yes for both "run simulation for…" flags — sizing is on by default. There is no per-object linkage between a specific `Sizing:Zone`/`Sizing:System` and a specific design day; EnergyPlus sizes against whichever `SizingPeriod:*` objects are present and enabled in the model as a whole. The dependency chain is: weather file assigned → `SizingPeriod:*` objects present (defaults or your own, §4) → `SimulationControl` flags turn sizing on → `Sizing:Zone`/`Sizing:System` autosize using whichever design days ran.

Full `Sizing:Zone`/`Sizing:System` field detail is covered in *(see "Sizing HVAC Systems and Zones")*; how the annual `RunPeriod` results build into a load profile is covered in *(see "Heating and Cooling Loads: From Design Day to Annual Profile")*.

---

7. Climate Zone

The **Climate Zone** dropdown (General Model Settings, and also shown in the Model Wizard) is a manually-selected ASHRAE 90.1 climate zone (0A through 8) for the project's location — it is not derived from `Site:Location` or the assigned weather file. As currently used in EP3, it exists solely as a required input to the ASHRAE 90.1 Appendix G baseline generator. See *(see "ASHRAE 90.1 Appendix G Baseline Generation")*.

8. Common Pitfalls

Being surprised by a per-EEM weather override. If a scenario/EEM has its own weather-file override set (§2.4), its results will diverge from the model's global weather assignment in a way that isn't visible unless you check that EEM's own weather setting. Before comparing EEM results side-by-side, confirm whether any of them carry an override rather than inheriting the global file.

Creating a `Site:Location` object without setting Keep Site Location Information to Yes. With that field at its default of No, EnergyPlus overwrites whatever you've typed into `Site:Location` with the assigned weather file's own header data at simulation time (§3) — the override silently does nothing.

Changing the weather file after creating a `Site:Location` override. EP3 does not update or remove an existing `Site:Location` object when you reassign the model's weather file (§3) — if you're using this override, check it whenever you change weather files, or you'll simulate one weather file's climate against another location's coordinates without any warning.

End of draft chapter. Outstanding items for the next pass: resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

5. Creating Building Geometry

1. Overview

EP3 builds a building's simulated geometry through a small pipeline, one floor at a time: you draw a Floor, its enclosed footprint areas become `Space` objects automatically, and a separate step extrudes those Space boundaries into real wall and roof surfaces. Every wall, floor, and roof face that results is a `BuildingSurface:Detailed` object; every glazed or door opening is a `FenestrationSurface:Detailed` object — both derived automatically from the geometry you draw.

EP3 creates one `Zone` per `Space` automatically, so a model is simulate-able the moment its geometry and constructions are complete — but you can override this and combine multiple Spaces into a single Zone, or otherwise customize the assignment, at any time. *(see "Spaces and Zones")* covers that assignment step and thermal-block strategy in full; this chapter stops at Space creation.

Everything above describes drawing geometry by hand. EP3 can also bring in an existing building's surfaces by importing an EnergyPlus IDF or epJSON file — including one produced by a third-party BIM-to-EnergyPlus tool such as Pollination, which outputs a native EnergyPlus file for EP3 to import rather than integrating with EP3 directly. *(see "Import, Export, and Interoperability")* covers that path in full. One consequence worth knowing here: an imported surface's boundary condition is set explicitly by the importer rather than resolved automatically the way hand-drawn geometry is (§4, §13), so imported geometry typically shows up flagged in the Unexpected Adjacency highlighter (§13) and is worth checking there before relying on it.

Jumping into an object's context. Every row in the Building Geometry Outliner — Building, a Floor, its Walls, a Roof, a Shading group — has a right-arrow (→) that jumps you directly into that object's editing context, the same as double-clicking it in the 3D view. A **View Scene** button on the Building row and on each Floor row switches the camera to a saved view framing that object, a fast way to jump between floors without manually orbiting and zooming.

Show/hide toggles cascade. The toggle next to a Floor hides that floor along with its walls and any background drawings attached to it. The toggle next to that floor's own Floor Plans (drawings) section has the same cascading effect from the other direction, hiding the drawing, the floor, and its walls together. Toggling an individual drawing's own row hides just that drawing; toggling an individual wall group's row hides just that wall group. Reach for the more specific toggle when you only want to hide one piece rather than everything tied to that floor.

Toggle Exterior Geometry Visibility hides everything except whatever you're currently editing — useful for seeing through the rest of the building while working on one piece. It's easy to trip over right after starting a new, empty Building or Site Shading group (§8): with this toggle on, an empty group has nothing to show, so the whole building appears to have vanished. Turn the toggle off (or just start drawing) rather than assuming the model was lost.

---

2. Building Orientation (North Axis)

North Axis (`Building` object, group "Simulation Parameters," default 0°, degrees from true north) is set from the Building object's own properties, wherever the Building node appears in the Building Geometry Outliner. The Model Wizard's **Building North Axis** field on its Building Geometry Properties tab writes into this same field.

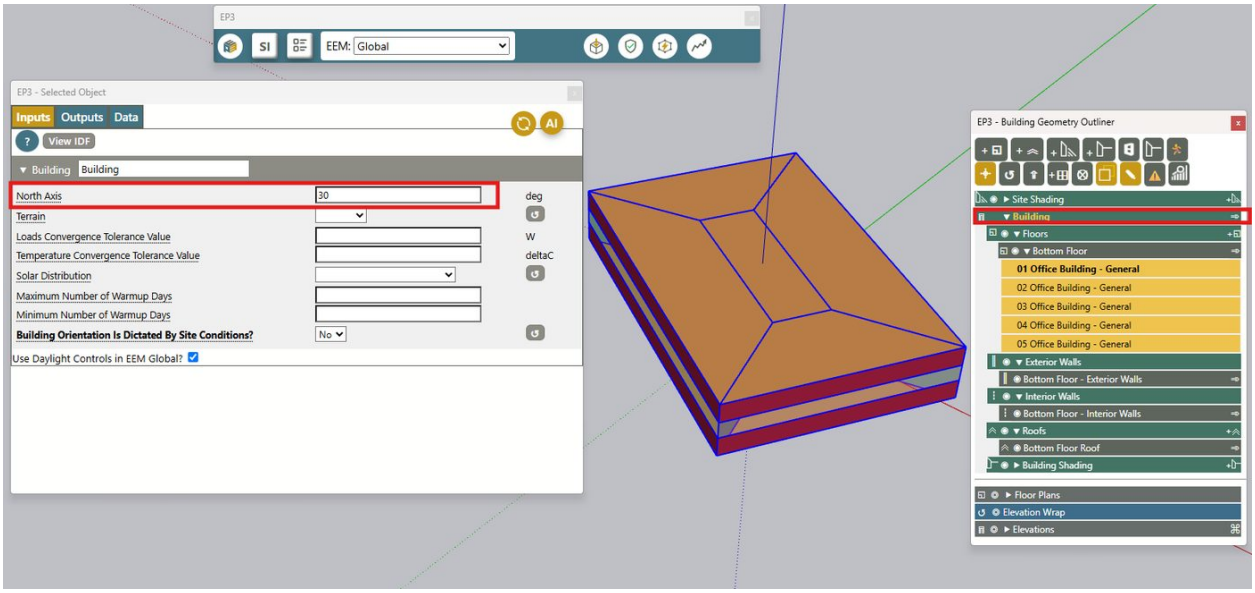


Figure 31: The Building object's properties panel with North Axis set to 30°, and the resulting rotated geometry in the 3D view.

Setting North Axis does two things at once: it sets EnergyPlus's own `North Axis` field for solar-position calculations, and it physically rotates the Building — every floor, wall, roof, and window already drawn — about the origin. There's no need to redraw anything to reorient a building; the whole model spins to the new angle.

You can set or change this whenever you like — the drawing tools handle re-orientation cleanly regardless of when it happens. Because North Axis is a real rotation of the geometry rather than just a field EnergyPlus reads internally, SketchUp's native daylighting and shadow tools, and any sun-path extension, read the building's true orientation directly and work correctly against EP3 geometry without extra setup. North Axis is also a natural EEM parameter for comparing orientation alternatives on the same building. *(see "Energy Efficiency Measures (EEMs)")*

Local vs. site coordinates. Because North Axis rotates the Building rather than individual entities, a rotated building's floors and walls live in a different coordinate frame than the Site group around it. Background floor-plan and elevation drawings (§10–11) are positioned in the Building's local (rotated) coordinates, not the Site's. EP3 warns about this directly: attempting to set an elevation while North Axis is non-zero shows *"The building north axis is not equal to zero. This can cause some confusion while assigning elevations because elevations are assigned based on the building's local coordinates, not site coordinates." To avoid confusion, you may want to draw your initial building geometry, and assign elevations with the building rotation set to 0.*

---

3. Adding and Configuring a Floor

Click **"+ Floor"** in the Building Geometry Outliner to add a floor. The dialog prompts for a **New Floor Name** (default "Floor"), a **New Floor Elevation**, and — once at least one floor already exists — an option to copy an existing floor's geometry into the new one, translated to the new elevation (the same underlying mechanism as the "Copy Floor" command described in §3.2).

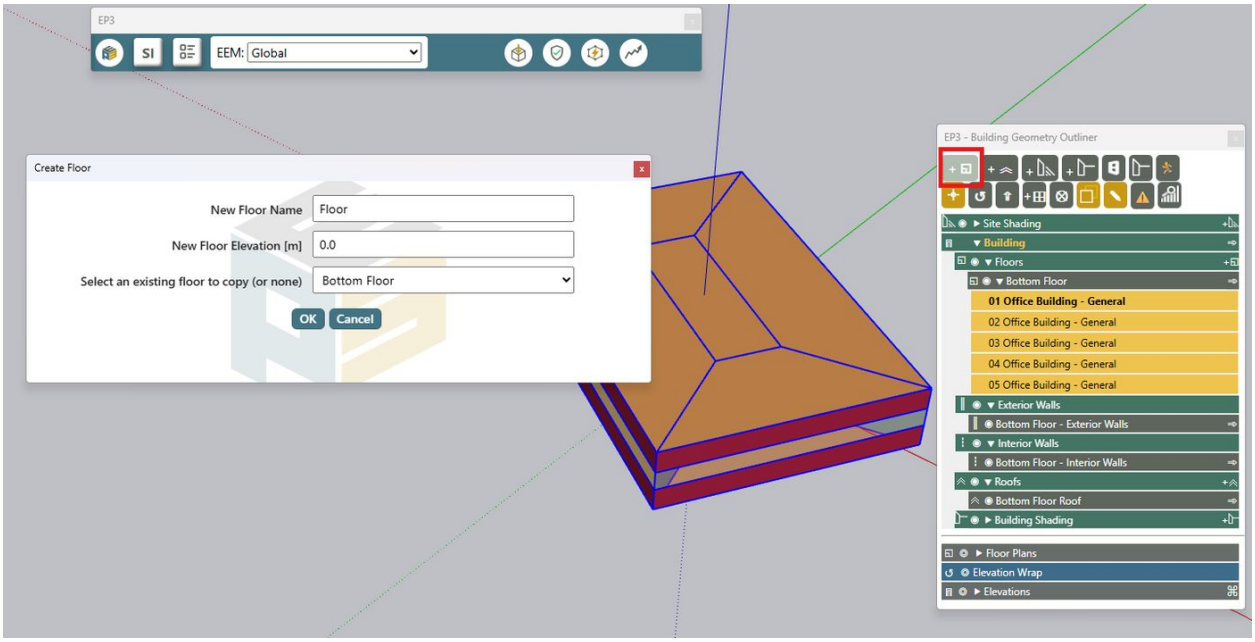


Figure 32: The "+ Floor" button and its Create Floor dialog: New Floor Name, New Floor Elevation, and the option to copy an existing floor.

3.1 Floor fields

- **Floor Height Calculation Method** — `Floor or Roof Above` (default) or `Specify Ceiling Height`.
- **Floor Or Roof That Determines Floor Height** — populated only when the method above is set to "Floor or Roof Above"; a dropdown of every floor or roof at least roughly 2 m above this one.
- **Ceiling Height** — the height used when "Specify Ceiling Height" is selected.
- **Floor Multiplier** — default 1.0; see §3.2.
- **Elevation** — this floor's height above the model origin. Changing it moves the floor's own geometry, its wall groups, and its linked background floor-plan image together.
- **Construction** — the default construction applied to every surface on the floor.

Whichever method you choose, wall generation (§5) always works the same way: a floor's exterior and interior walls extrude upward from the floor until they hit the next object above — another floor's slab, or a roof — **or** the height computed by these fields, whichever comes first. Leaving the default "Floor or Roof Above" is correct for nearly every floor; the two situations that call for "Specify Ceiling Height" are a building's top floor (there's nothing above it to stop at) and a split-level or atrium floor, where you want the walls to keep rising past where a normal floor slab would otherwise cut them off. Raising a floor's ceiling height past the floor (or roof) that would normally sit above it turns that area into a double-height space instead of a separate story, at no extra modeling cost beyond simply not drawing a floor above it. §14's worked example uses exactly this to build a two-story atrium.

3.2 Floor Multiplier vs. Copy Floor

These are two different multi-floor mechanisms, easy to conflate:

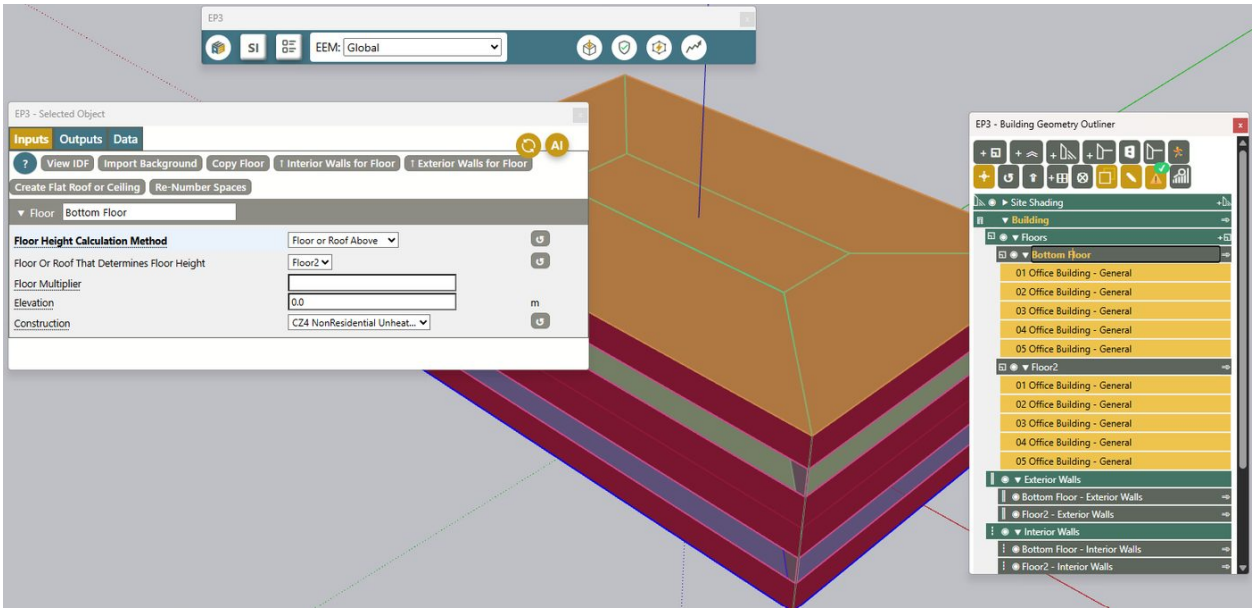


Figure 33: A Floor's properties panel, showing its per-floor action buttons and the fields listed above — here set to Floor or Roof Above, pointing at Floor2.

- **Floor Multiplier** (a Floor field, §3.1) is EnergyPlus's own repeat-simulation shortcut: set it above 1.0 and every `Zone` on that floor gets multiplied by it — EnergyPlus simulates the floor once and multiplies the result, with **no additional 3D geometry created.** If a Zone also carries its own multiplier, the two combine: the effective multiplier is the Zone's own multiplier × the Floor Multiplier. Use this for a tower's identical repeating floors.
- **Copy Floor** (the per-floor "Copy Floor" button, or the option offered when adding a new floor) performs an actual geometric duplication: a new Floor, including a deep copy of every nested Space's own data — space names and per-Space settings all survive the. Use this when a floor is only *similar* to another, not identical, and needs to diverge from there. Good modeling practice is to start with the **most common** floor in the building, and then copy it up, or down to create subsequent floors. This ensures that exterior walls and common interior walls are lined up.

---

4. Drawing Space Boundaries

Double-click into a Floor to edit it. Inside a Floor's editing context, drawing is constrained to that floor's own elevation plane — you can't pull a point out of plane the way you can when editing ordinary SketchUp geometry; a vertex that does end up off-plane (from an accidental 3D nudge, for example) will be corrected by EP3.

The Line, Rectangle, Move, Eraser, Select, and Paint Bucket tools all work the way you'd expect from SketchUp, with a few EP3-specific behaviors worth knowing:

- **The Paint Bucket paints both sides of a face.** This holds everywhere a `Sketchup::Face` is interpreted as a `BuildingSurface:Detailed` — inside a Floor, an exterior/interior wall group, or a Roof alike — so there's never ambiguity about which construction got assigned to a surface.

- **Paint is a SketchUp-level fact, not an EEM-scoped one.** A surface keeps whatever construction it's painted with across every EEM; there's no way to repaint it per EEM. To model a different assembly in an EEM, change that construction's own parameters instead of repainting — see *(see "Constructions and Materials", §1.1)*.
- Faces are never erased directly, only edges. Erasing an edge that separates two rooms merges them into a single Space the next time boundaries are recalculated.
- Drawing snaps to the grid spacing set in the Model Settings, to nearby vertices and edges, and as a continuation of an existing edge.
- The **Right/Left arrow keys** toggle a persistent lock to the model's local X or Y axis while drawing, the same way the arrow keys work in native SketchUp. **Shift** locks onto whatever direction is currently being inferred — an axis, or the "extend this edge" direction when the cursor is near the line of an edge already drawn — so you can move the cursor freely afterward without losing that lock. Press the same key again to release the lock.

Every closed loop of edges you draw becomes a `Space` automatically — there's no separate "create room" step. Redrawing or nudging a boundary preserves the Space's existing identity (name, schedules, thermostat, and so on) rather than creating a new one, provided the redrawn area still overlaps the old one; when a redraw is ambiguous, whichever candidate area overlaps most wins. Deleting all of a Space's remaining faces deletes the Space itself.

Right-click a selected Space (or its faces) for **"EP3 - Rename Spaces"** and **"EP3 - Assign Space Template"** — a quick way to rename or template several Spaces at once instead of opening each one's own properties panel individually. Renaming prompts for a new **Space Name**; assigning a template prompts for a **Space Template** from the model's library, with "DO NOT CHANGE" and "NONE" options available when the selection spans multiple Spaces. *(see "Internal Loads, Space Types, and Zone Types")* covers what a Space Template actually sets.

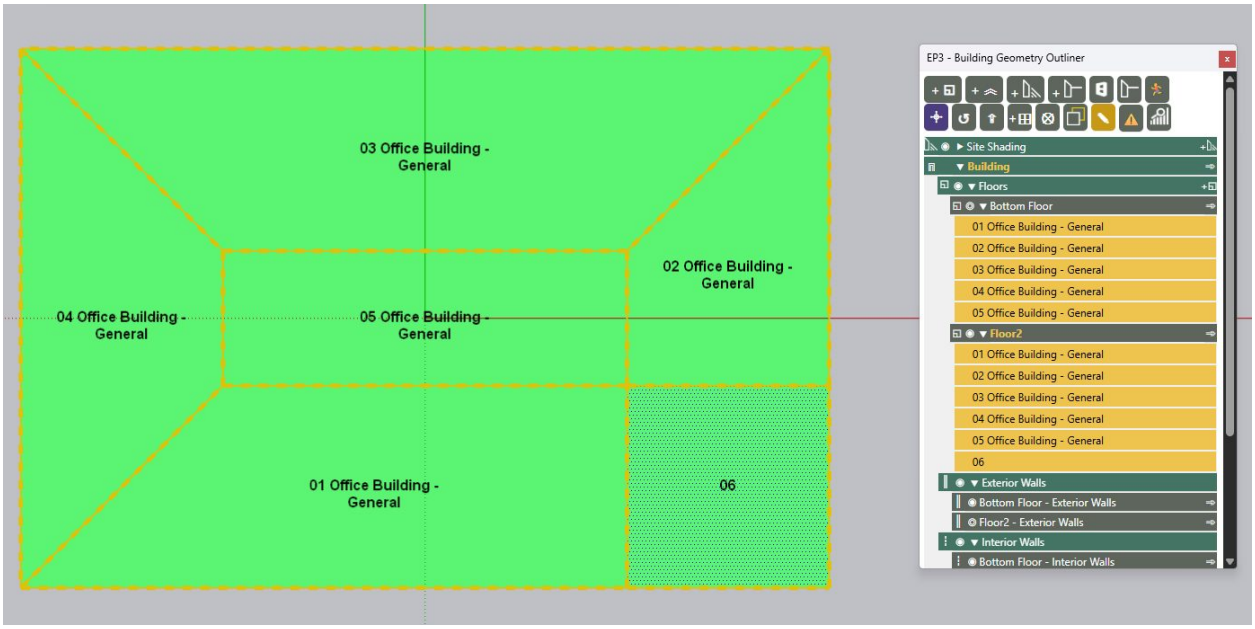


Figure 34: A floor plan with five templated Spaces ("Office Building - General") and a sixth, newly drawn Space still unassigned — shown here before running "Assign Space Template."

Selecting or hovering over a Space — or a Zone, ZoneList, SpaceList, SpaceType, or ZoneType

from wherever those are managed — highlights every surface belonging to the Space(s) associated with it directly in the 3D view. This is a fast way to confirm what a name actually corresponds to geometrically once a model has enough Spaces that names alone stop being enough to place them.

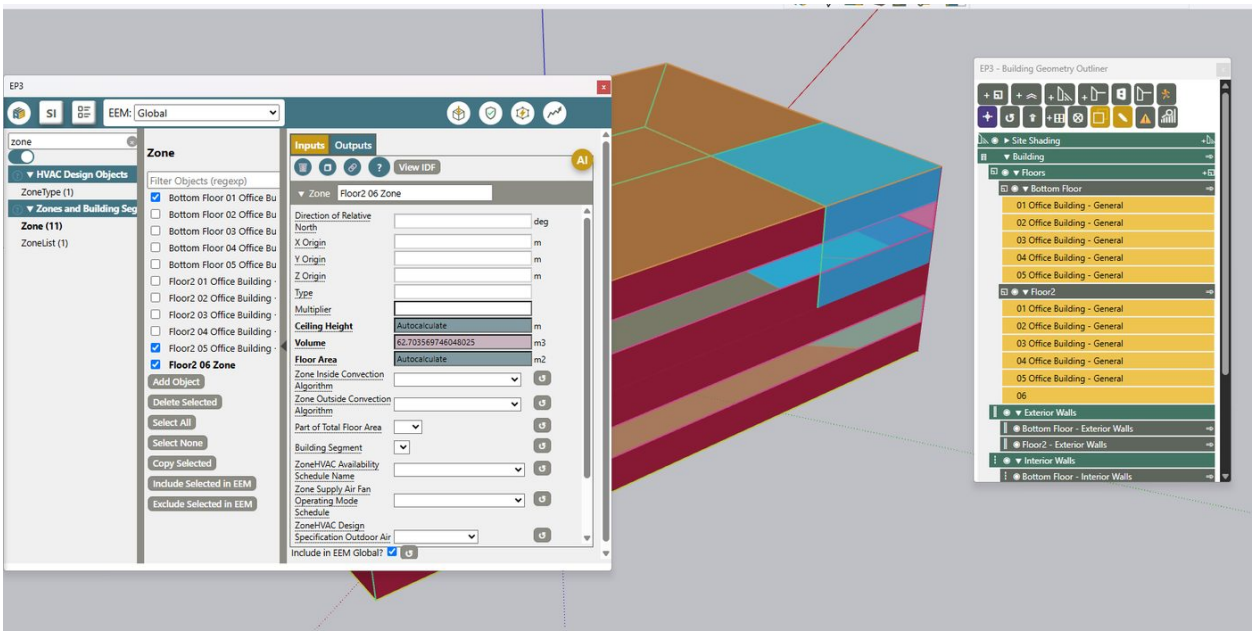


Figure 35: Checking Zones in the Zone object list (left) highlights each checked Zone's surfaces directly in the 3D view — here two zones on Floor2, shown in blue and pink.

---

5. Generating Walls

Once a floor's Space boundaries are drawn, generate walls either per floor — the **"↑ Interior Walls for Floor"** / **"↑ Exterior Walls for Floor"** buttons in that floor's own panel — or for the whole model at once, using the Building Geometry Outliner's **"+ Walls"** button (tooltip: "Generate, or re-generate walls"), which prompts:

- **Align Walls Between Floors?** — nudges wall vertices between adjacent floors closer together (up to twice the grid spacing) to avoid sliver surfaces where floors don't quite line up. Running this repeatedly can drift walls further with each pass rather than converging on an aligned result — align once and check the result, rather than running it repeatedly to "settle" a model.
- **Generate Walls** — Exterior, Interior, or All.
- **Generate Flat Roofs** — automatically adds a flat roof over any floor whose height can be resolved to a number (§6).

Either path re-extrudes the floor's space-boundary edges upward to the height computed in §3.1, intersects the result against the rest of the model so window openings, floor slabs, and roof lines cut correctly, and re-matches every resulting face to whichever Space(s) it borders. If a floor's walls already exist, this replaces the faces in that same group rather than starting over, preserving whatever materials and attributes are already assigned.

To regenerate only part of a model — say, after editing one floor's footprint — go to that

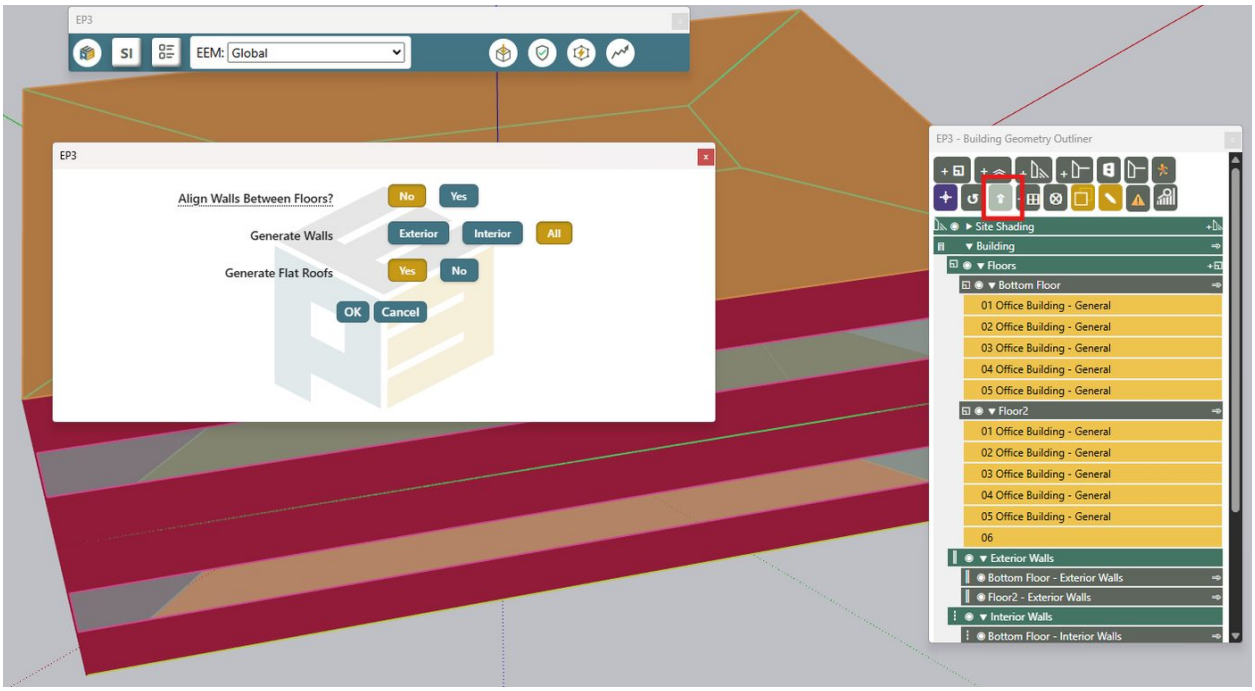


Figure 36: The wall-generation prompt: Align Walls Between Floors?, Generate Walls (Exterior/Interior/All), and Generate Flat Roofs.

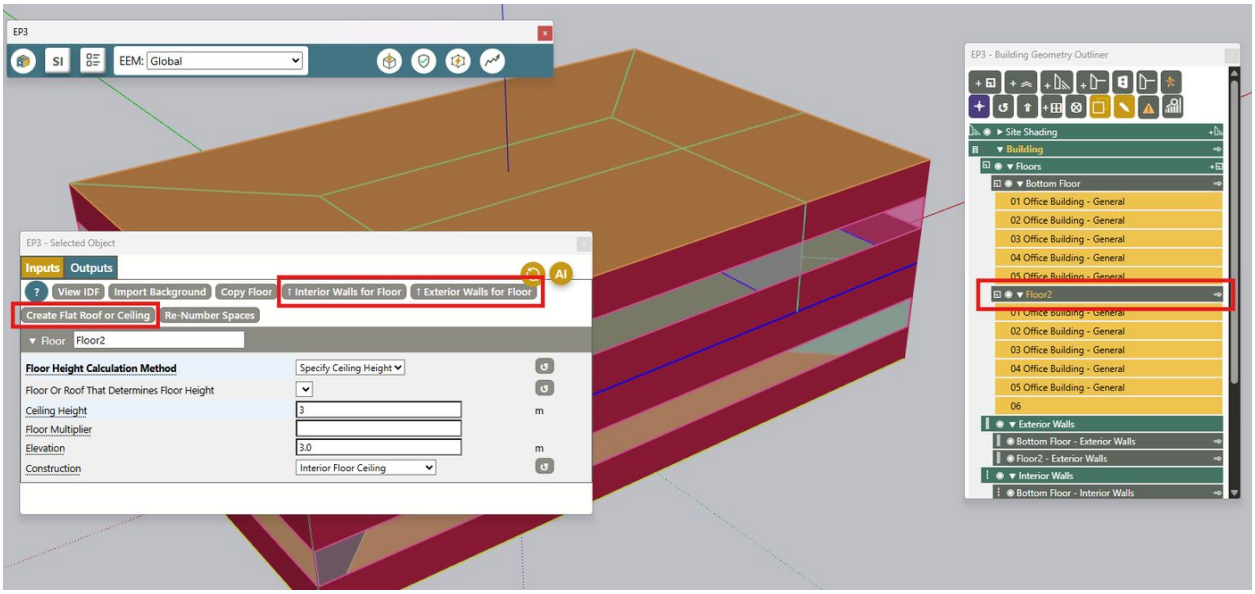


Figure 37: The per-floor “Interior Walls for Floor” / “Exterior Walls for Floor” / “Create Flat Roof or Ceiling” buttons, highlighted on Floor2’s properties panel.

floor's own panel and use its individual "Interior Walls"/"Exterior Walls"/roof buttons rather than running the model-wide "+ Walls" command across the whole building.

---

6. Generating Roofs

EP3 builds two different kinds of roof, and they don't coexist on the same floor:

- **Flat roofs** are generated automatically — either from a floor's own "Create Flat Roof or Ceiling" button, or across the whole model via "+ Walls" 's "Generate Flat Roofs" option (§5).
- **Pitched roofs** are created explicitly with **"+ Roof."**

A floor with a pitched roof associated with it does not also get a flat roof generated — not unless that pitched roof is deleted first.

"+ Roof" prompts differently depending on whether any floors already exist: - **No floors yet: Enter Roof Name**, **Enter Roof Area**, **Enter Roof Elevation** — builds a blank placeholder roof of that area (a plain square) for you to edit by hand. - **Floors exist: Enter Roof Name**, **Copy Floor** (which floor's footprint to trace), **Enter Roof Elevation**, **Enter Roof Overhang**, **Enter Roof Angle** — copies the chosen floor's footprint at the given elevation, offsets it outward by the overhang, and, if Roof Angle is greater than 0°, converts the flat copy into a pitched roof at that angle.

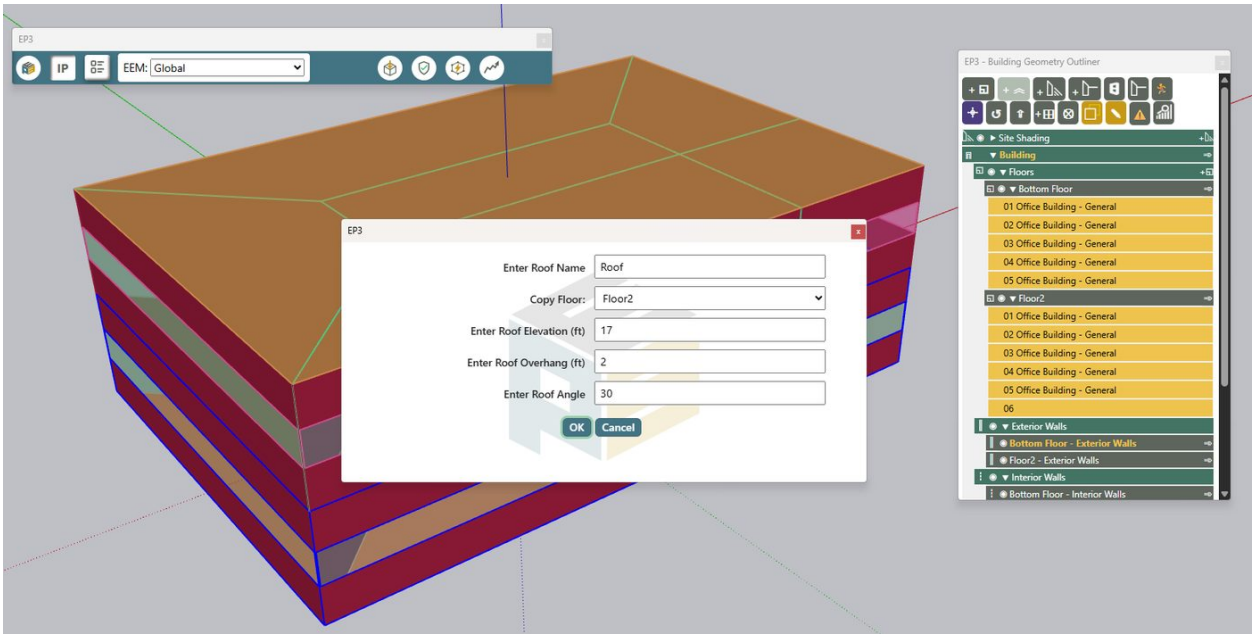


Figure 38: The "+ Roof" dialog in its floor-exists mode: Roof Name, Copy Floor, Roof Elevation, Roof Overhang, Roof Angle.

Applied to a rectangular footprint, Roof Angle produces a hip-style roof — all four sides sloping up to a single center ridge point, not just two opposing gable ends.

Make sure the floor's walls are tall enough to actually reach the roof. A pitched roof is separate geometry sitting above the floor, not something the wall-extrusion step automatically

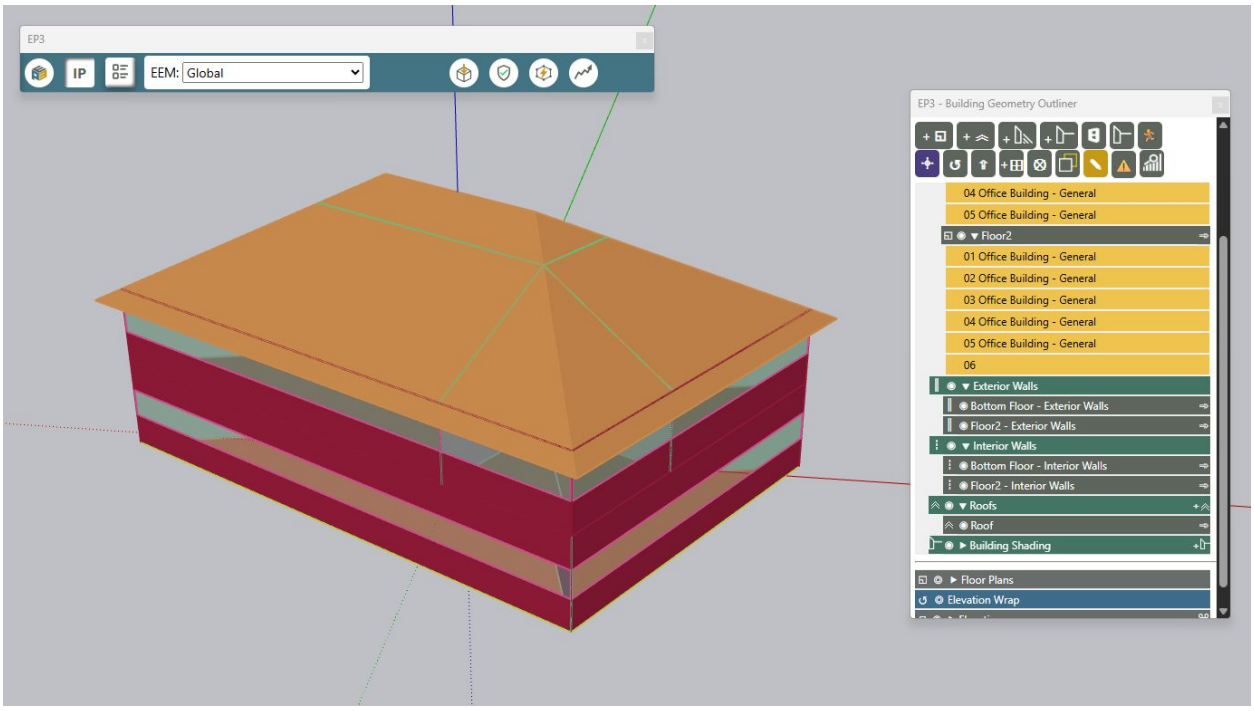


Figure 39: The resulting pitched roof: a hip roof rising from the footprint's overhang line to a single center peak.

reads the shape of — set the floor's Ceiling Height (or whichever floor/roof it's set to key off, §3.1) tall enough that its walls extend all the way up to the underside of the roof, including at the roof's peak. Walls that stop short leave a gap between the top of the wall and the sloped roof deck instead of a clean intersection.

Once you edit a roof's own geometry into a genuinely three-dimensional shape — reshaping a pitched roof by hand, for instance — that roof is no longer touched by later wall/roof regeneration; EP3 treats hand-edited 3D roof geometry as finished rather than something to recreate on the next "+ Walls" pass.

Beyond that, a roof is just ordinary SketchUp geometry — reshape it, add dormers, split it into multiple faces, or otherwise edit it by hand as freely as any other group. What matters for a roof face to be picked up as a real building surface is whether it can "see" a Space below it: EP3 matches a roof face to whatever Space sits underneath by checking straight down from the face, so a roof face with a clear line of sight to a Space's floor area gets matched to it automatically, the same as walls being matched by adjacency. A roof face that can't see any Space below — because it overhangs past the building's footprint, or sits over a gap, for instance — doesn't get silently dropped; it becomes shading instead (`Shading:Building:Detailed`), the same fallback described in §8. That fallback shows up in two places: the Unexpected Adjacency highlighter (§13, green — unmatched to any Space) and the outliner's own **"Highlight Building Shading"** toggle, which is worth checking after reshaping a roof to confirm you got the building-surface/shading split you actually intended.

6.1 Worked example: overhang and pitch to ridge height

"Enter Roof Angle" is measured from horizontal and applied uniformly around the copied footprint's edges, so a roof's rise above its eave follows simple trigonometry: for a given horizontal run — half the footprint's shorter dimension, for the rise to a center ridge —

$$\text{rise} = \text{run} \times \tan(\text{angle})$$

For a 10 ft-wide footprint with a 35° Roof Angle, the run from eave to ridge is half the width, 5 ft:

$$\text{rise} = 5 \text{ ft} \times \tan(35°) = 5 \times 0.7002 = 3.50 \text{ ft}$$

so a roof copied at an eave elevation of 20 ft reaches a ridge around 23.5 ft. Checked in SI: run = 5 ft × 0.3048 = 1.524 m; rise = 1.524 m × tan(35°) = 1.067 m = 3.50 ft — the two paths agree. §14 uses this relationship for the worked example's atrium roof.

---

7. Creating Windows

7.1 Add Windows By Exposure

The **"Add Windows"** toolbar button (tooltip: "Add Windows By Exposure") prompts:

- **Select Orientation** — All Orientations (default), North, South, East, or West.
- **Enter Window to Wall Ratio (%)** — 0–100, default 25%; entering more than 99% is rejected.
- **Erase or turn off existing windows** — "Turn Off" means that any current windows found on the selected elevation(s) will be toggled to "off" in the current EEM. "Erase" will erase any windows that are currently on.
- **Glass Construction** — every semi-transparent, EP3-recognized glazing construction already in the model, plus the model's default.

This places windows on every exterior wall face whose orientation falls in the requested range, corrected for the Building's North Axis (§2) rather than SketchUp's raw axes — so "South" always means the building's actual south, however it's currently rotated. Rectangular wall faces get a scaled instance of the standard window component, centered on the face; non-rectangular faces get a custom triangulated inset sized to hit the target ratio, with a warning that the result may not be fully contained by the wall and should be checked. Existing doors on a wall are detected and preserved rather than overwritten. Every window this creates is assigned an Opening Type sized to match the requested Window to Wall Ratio — the same template mechanism described at the end of §7.2.

Windows (or doors) turned off rather than erased — by the "Erase or turn off existing windows" option above, or by turning an object off in an EEM directly — are hidden from the 3D view by default. The Building Geometry Outliner's **"Show / Hide Objects that are Off in this EEM"** button toggles their visibility back on; the HVAC outliner has the same toggle for HVAC equipment that's off in the current EEM.

7.2 Painting a custom window

Painting any wall or roof face with a semi-transparent material converts it into a real, glued window or door the moment you exit that wall or roof group's editing context — there's no separate "convert

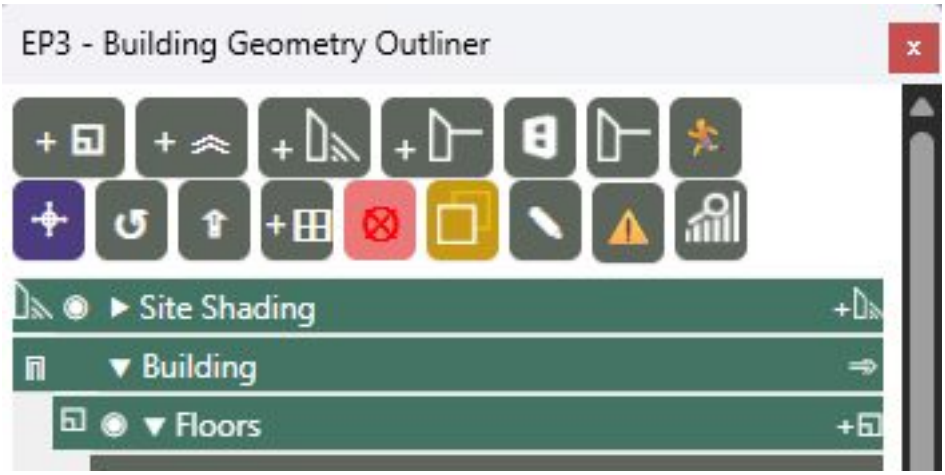


Figure 40: The Building Geometry Outliner's "Show / Hide Objects that are Off in this EEM" toggle (highlighted).

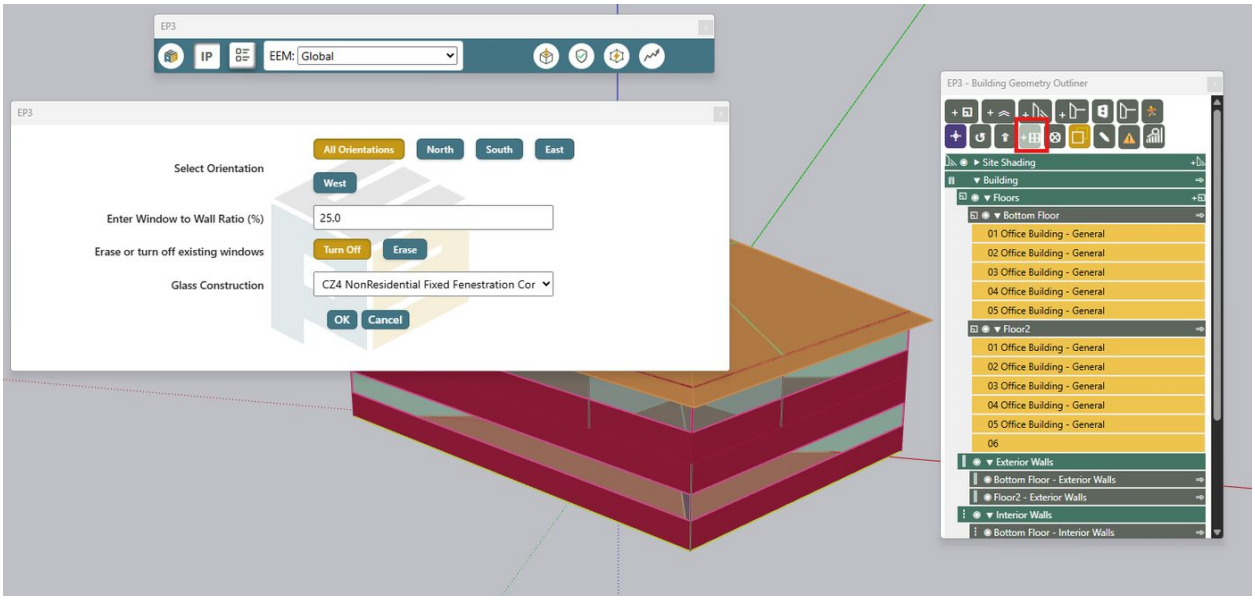


Figure 41: The "Add Windows By Exposure" dialog: Select Orientation, Window to Wall Ratio, existing-window handling, Glass Construction.

to window" command. A rectangular painted area becomes a scaled instance of the standard window component; any other shape becomes a one-off custom component built directly from the painted outline.

Windows can also be placed by right-clicking and selecting **EP3 - Add Window** - windows created this way can be resized using the Sketchup Stretch tool (**S** shortcut on the keyboard).

Automatic conversion ONLY happens for glazing-type construction. If the intent is to create a door, use **EP3 - Add Window** and then paint the object with the desired construction type.

Use **"Copy Windows from Floor"** (on an exterior wall group) to replicate every window from another floor's exterior wall group onto this one, adjusted for the elevation difference — it prompts for what to do with any windows already present (**Erase Them**, **Turn Them Off**, or **Maintain Them**).

"Divide Windows Across Walls" (right-click a window/door) splits it into multiple window instances wherever its geometry now spans more than one sub-face — useful after editing a wall in a way that introduces a new edge under an existing window. **"Select All Windows and Doors"** selects every window/door in whatever group you're currently editing.

(see "Windows, Shading, and Daylighting") covers window templates (which control size and sill-height calculation, construction, and frame/divider assignment) and the full field set once a window exists.

---

8. Creating Building Shading and Site Shading

"+ Building Shading" and **"+ Site Shading"** each create an empty group and drop you into it to draw, exactly like starting a new Floor — but unlike a Floor, there's no plane restriction here: draw with SketchUp's ordinary 3D tools, since shading massing (overhangs, fins, neighboring structures) is often genuinely three-dimensional.

The type of shading object is important - every face inside a Building Shading group becomes `Shading:Building:Detailed`; every face inside a Site Shading group becomes `Shading:Site:Detailed`. Building Shading groups **rotate** with the building. This is an appropriate choice for representing architectural elements such as shading fins, balconies or other structures attached to the building; Site Shading groups do not rotate with the building, and are appropriate for modeling adjacent structures, large, legacy trees or terrain features big enough to affect the site's solar access.

Within site and building shading groups, only Sketchup faces and edges are allowed. Any other geometry you create inside the group will be erased or exploded.

---

9. Importing Adjacent Site Context via Geolocation

SketchUp's own Geo-location feature (Add Location / Toggle Terrain — not an EP3 tool) can bring in an aerial snapshot of a real site and, coverage permitting, extruded massing for nearby buildings and/or a terrain mesh. EP3 doesn't drive that import; it only offers a conversion step afterward.

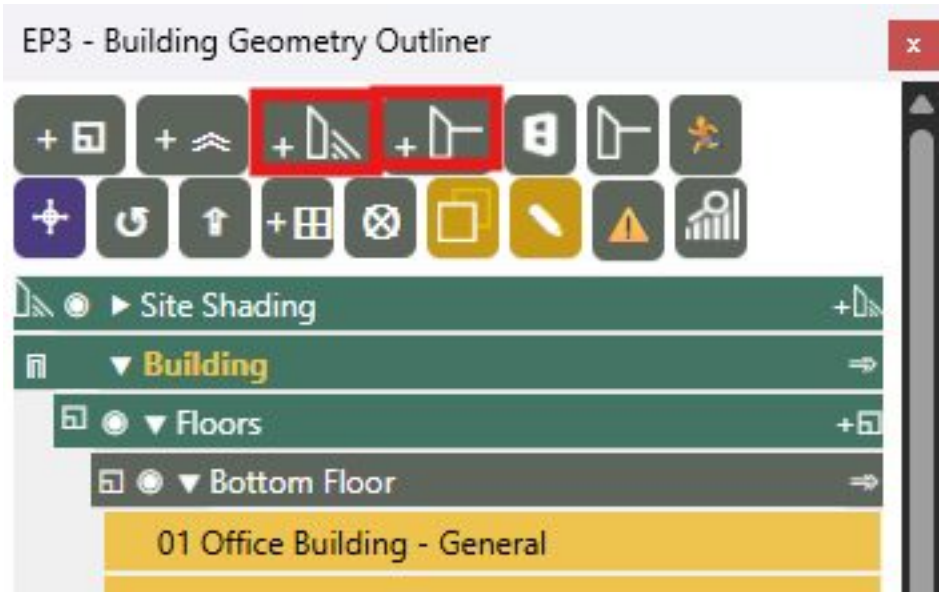


Figure 42: The "+ Building Shading" and "+ Site Shading" toolbar buttons in the Building Geometry Outliner.

To start it, click the Geolocation icon in the lower-left of the SketchUp window, choose **"Add Location..."**, and select your site in the dialog that opens. Set **True North offset** to 0° — EP3's own North Axis field (§2) is what handles any actual building rotation, and it assumes the model's Y-axis already points true north, so leaving SketchUp's True North offset at 0° keeps the geolocated site aligned with that convention rather than introducing a second, independent rotation.

From there, click **"Add Site Context"** and drag out the area you want to bring in. The panel that opens lets you choose **Import Type** (Flat Site or Elevated Site) and which **Context** layers to include — Map Texture, Terrain Mesh, Building footprints, and **3D Buildings** (the one that matters for the conversion workflow below); SketchUp previews which buildings fall inside your selection before you click **"Import Site Context"** to commit.

Add only the minimum footprint actually needed to capture real shading impacts on your building: every additional surface pulled in becomes its own `Shading:Site:Detailed` object once converted (below), and more surfaces in an EnergyPlus model means longer simulation times, so a wide site-context grab you don't need is a real, avoidable cost rather than a harmless default.

Right-click the top-level group Geo-location created — offered only when nothing is currently being edited, exactly one object is selected, and that object is unmistakably a geolocation result — and choose **"EP3 - Convert Site Context to EP3 Shading Objects."** Depending on what the geolocation import actually contains, it may prompt:

- **Convert buildings from:** — `Elevated Site` or `Flat Site` (only asked if both are present).
- **Convert 3D terrain to Site Shading?** — `No` (default) or `Yes`.

Each building (and the terrain, if included) becomes its own new group under Site geometry, wrapped as Site Shading — **always Site Shading, regardless of which building-massing type it came from.** The original geolocation snapshot is hidden, not deleted, so it stays available as a visual reference. Because the imported massing is converted face-for-face with no simplification, dense photorealistic building data from a well-covered urban site can produce a correspondingly

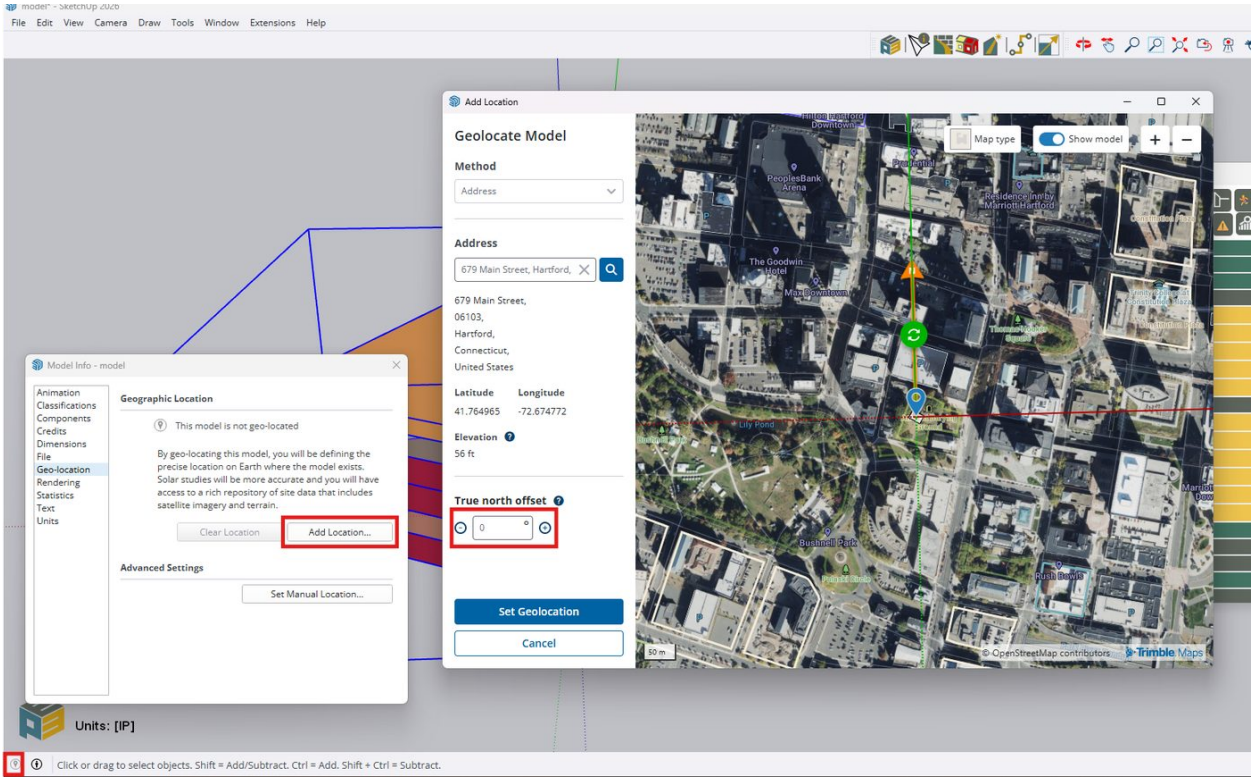


Figure 43: SketchUp's Add Location dialog, with True North offset set to 0°.

large number of `Shading:Site:Detailed` surfaces — worth checking before running a simulation.

North Axis caution. EP3 keeps the SketchUp model's own geolocation coordinates synced to whichever weather file is assigned to the project, but it always writes a `GeoReferenceNorthAngle` of 0°, independent of whatever the Building's `North Axis` field (§2) is set to. The two "north" concepts are tracked independently, so importing/converting site context and rotating North Axis don't automatically stay consistent with each other. Treat the two as needing separate verification: after doing both to the same model, visually confirm the converted site context still lines up with your building the way you expect, rather than assuming one operation accounts for the other.

---

10. Importing and Tracing a Background Floor Plan

A floor's **"Import Background"** button accepts an image (BMP/JPG/PNG/TGA/TIFF/PSD), a PDF, or a CAD file (DWG/DXF) as a non-destructive reference layer for that floor — it never creates EP3 geometry on its own.

After import, an interactive tool walks you through positioning it: an optional scale-from-known-length step (click two points on the drawing, then type the real-world distance between them), then position (click a point on the drawing, then the matching point on your building), then rotation, then scale — each step skippable. **"Position Background"** re-opens this same tool later if the drawing needs adjusting.

The recommended workflow from here is to trace over the positioned drawing by hand with the Line

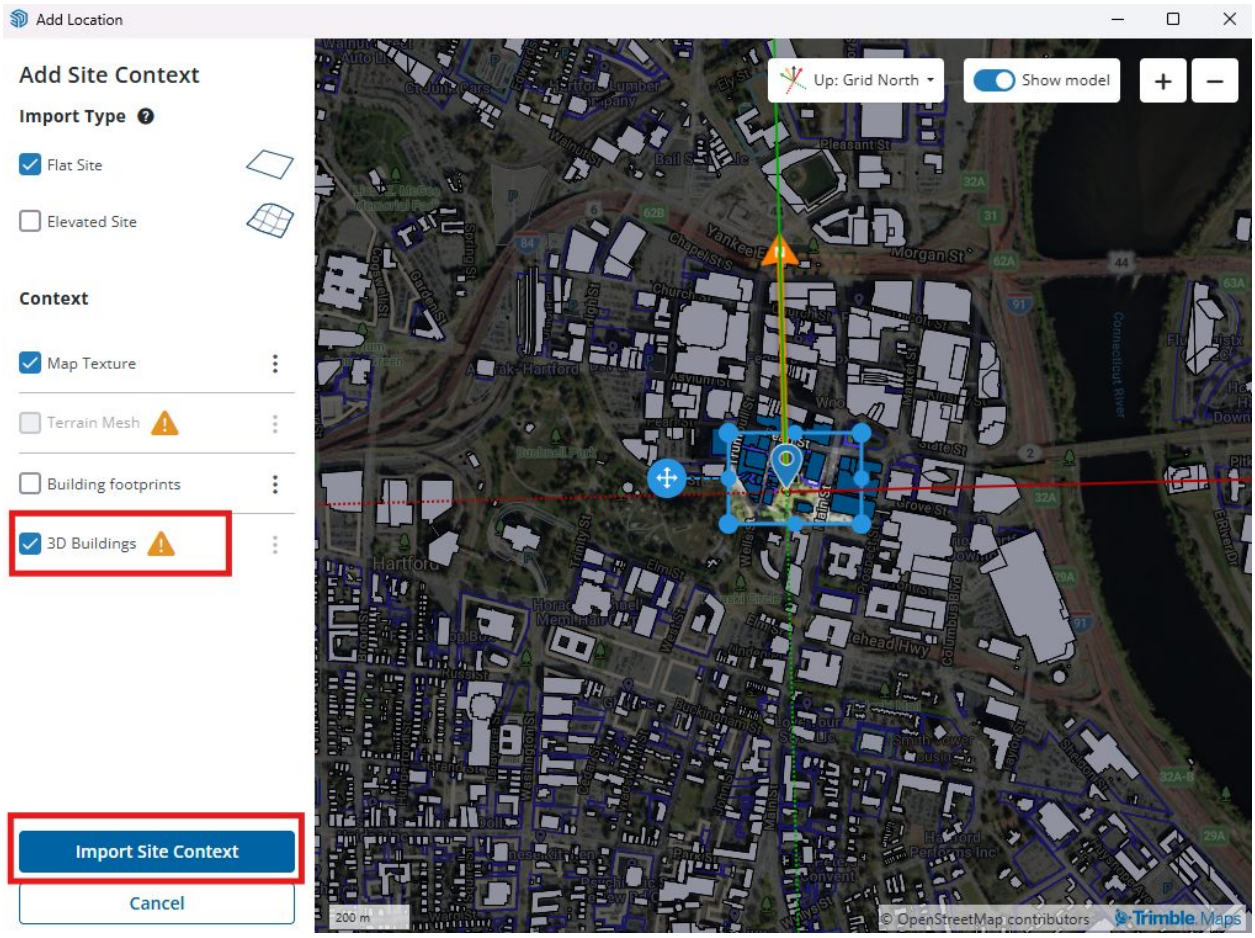


Figure 44: SketchUp's Add Site Context panel: Import Type, Context layers (3D Buildings checked), the previewed selection area, and the Import Site Context button.

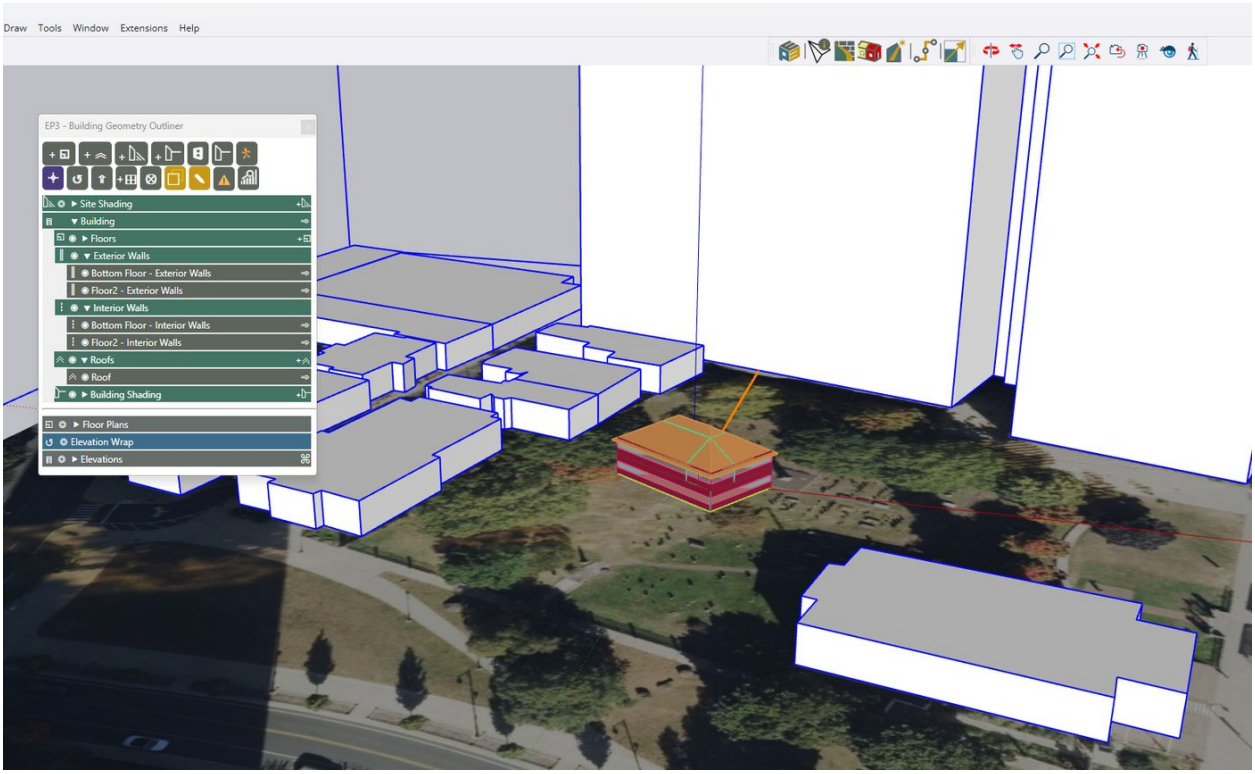


Figure 45: The imported geolocation content — aerial imagery and neighboring 3D building massing — sitting around the EP3 building before conversion.

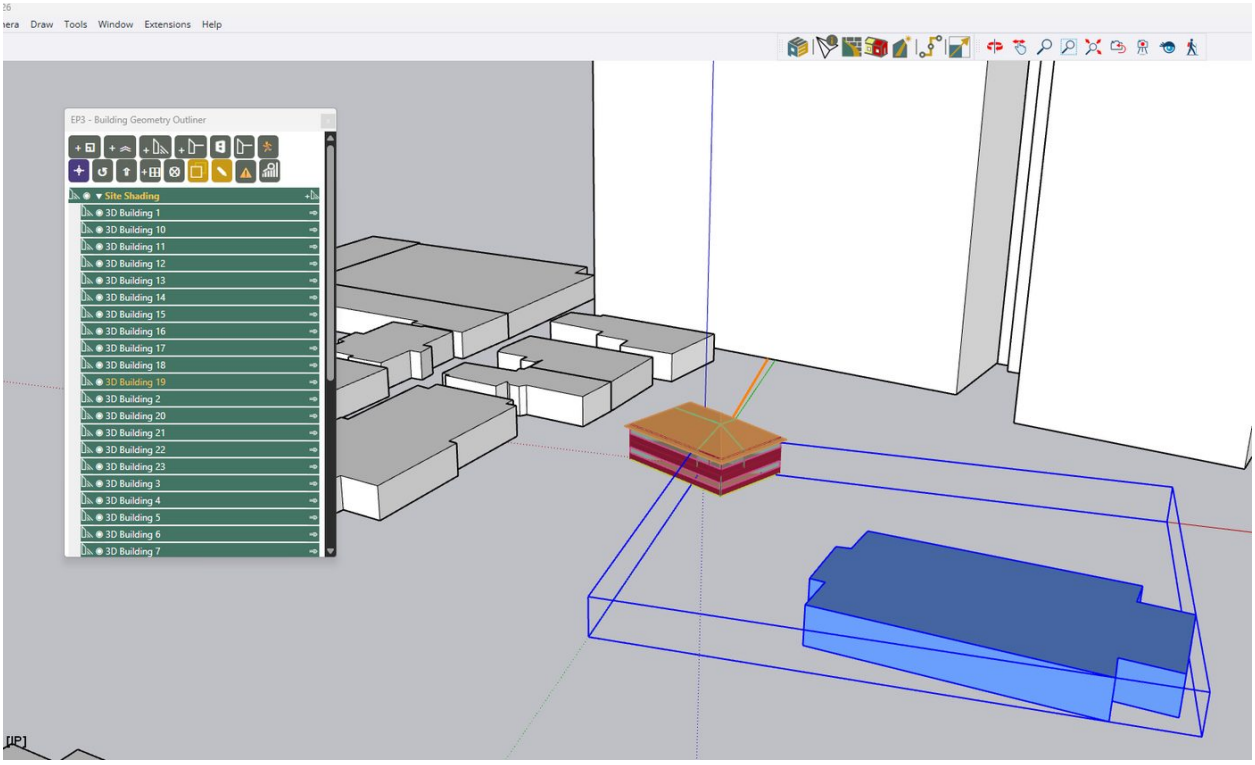


Figure 46: Converted neighboring buildings listed as individual Site Shading objects in the outliner — clicking one ("3D Building 19") highlights its mass in the 3D view, the same selection-highlighting behavior described in §4.

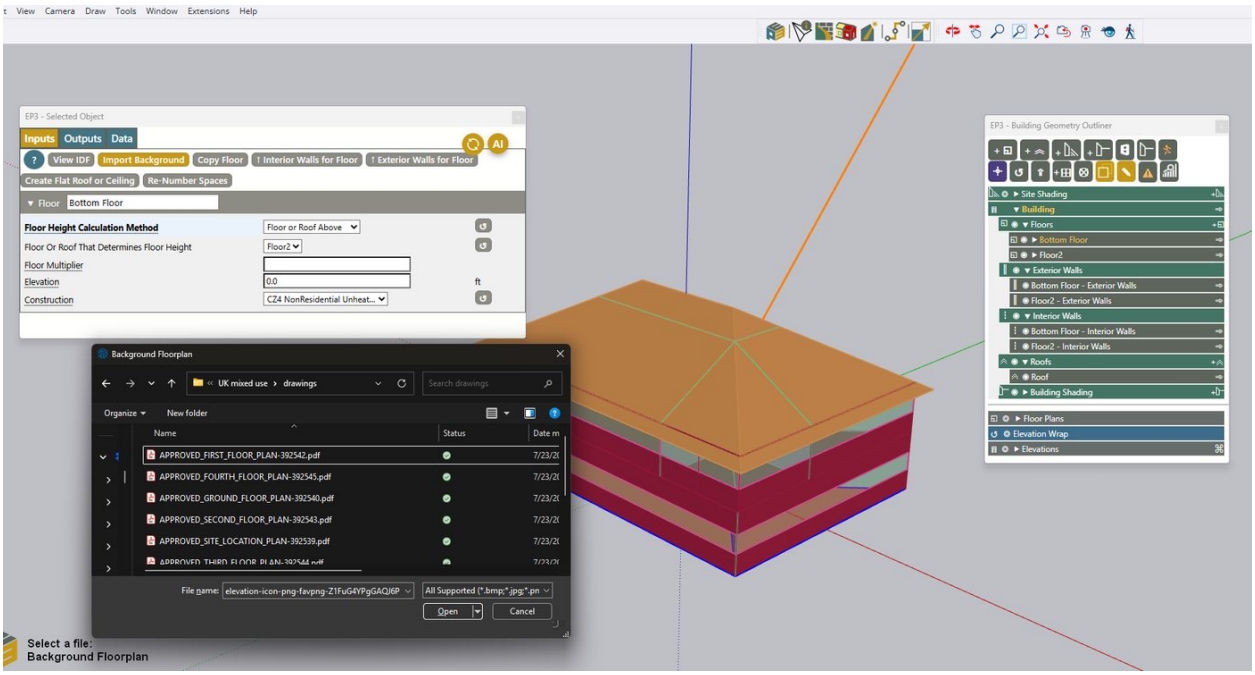


Figure 47: A floor's "Import Background" button and the file picker showing the supported formats.

and Rectangle tools (§4) — closing loops directly on top of the reference image produces exactly the Space boundaries you want.

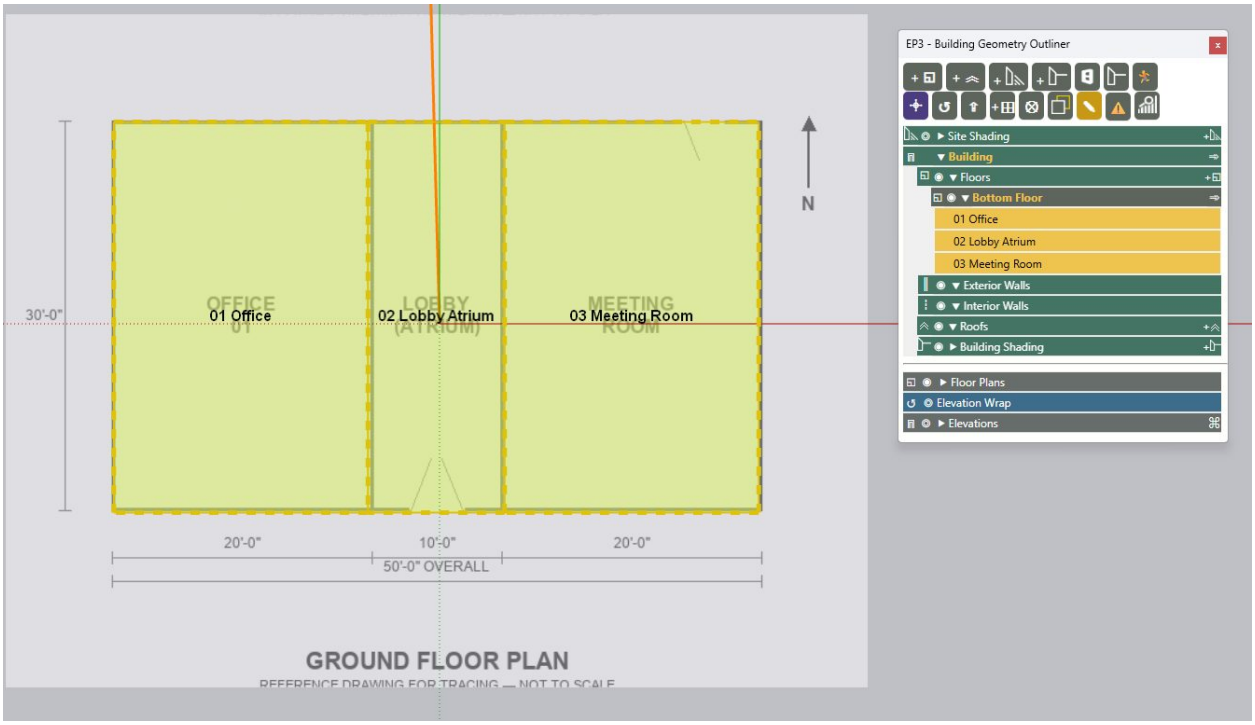


Figure 48: The worked example's ground floor plan positioned as a background drawing, with its three Space boundaries hand-traced directly on top of it.

An automatic alternative, **"EP3 - Generate Floor from Drawing"** Right-click on a CAD drawing that has been imported into the general Sketchup model space - EP3, vectorizes it into a floor directly — bridging small door-sized gaps, snapping to the grid, and merging away enclosed areas below roughly 97 sq ft as noise. This function depends on clean input geometry to produce a usable result and isn't a substitute for tracing when the source drawing is messy or low-resolution; hand-tracing with EP3's own tools over a positioned background remains the most reliable option.

---

11. Importing and Orienting Elevation Drawings

Start an elevation import from the clover-shaped **"+"** button next to **Elevations** in the outliner.

Elevation drawings use the same import formats and the same alignment tool as floor plans (§10), but are placed vertically instead: after import, you choose an **Orientation** (North, South, East, West, or a custom angle from north) and whether to **Mirror** the drawing, and EP3 tips it upright to face that direction and files it under a matching North/South/East/West layer.

To reposition a drawing already in the model, use the same tool that placed it: a floor plan is repositioned from that floor's own **"Import Background"** panel, via **"Position Background"** (§10); an elevation is repositioned by selecting it in the outliner and clicking the clover button again. The native SketchUp Move tool works fine for small nudges to either kind of drawing's position, but changing which cardinal direction an elevation is aligned with should always go through EP3's

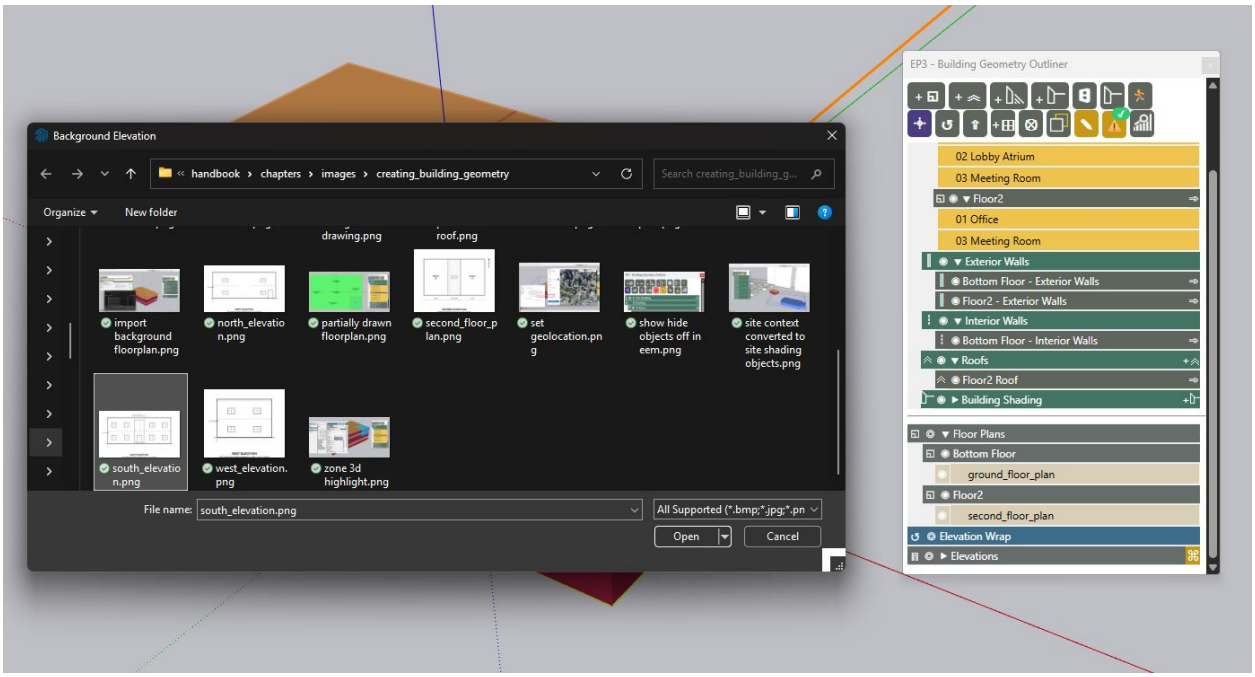


Figure 49: Importing an elevation drawing from the clover-shaped "+" button next to Elevations.

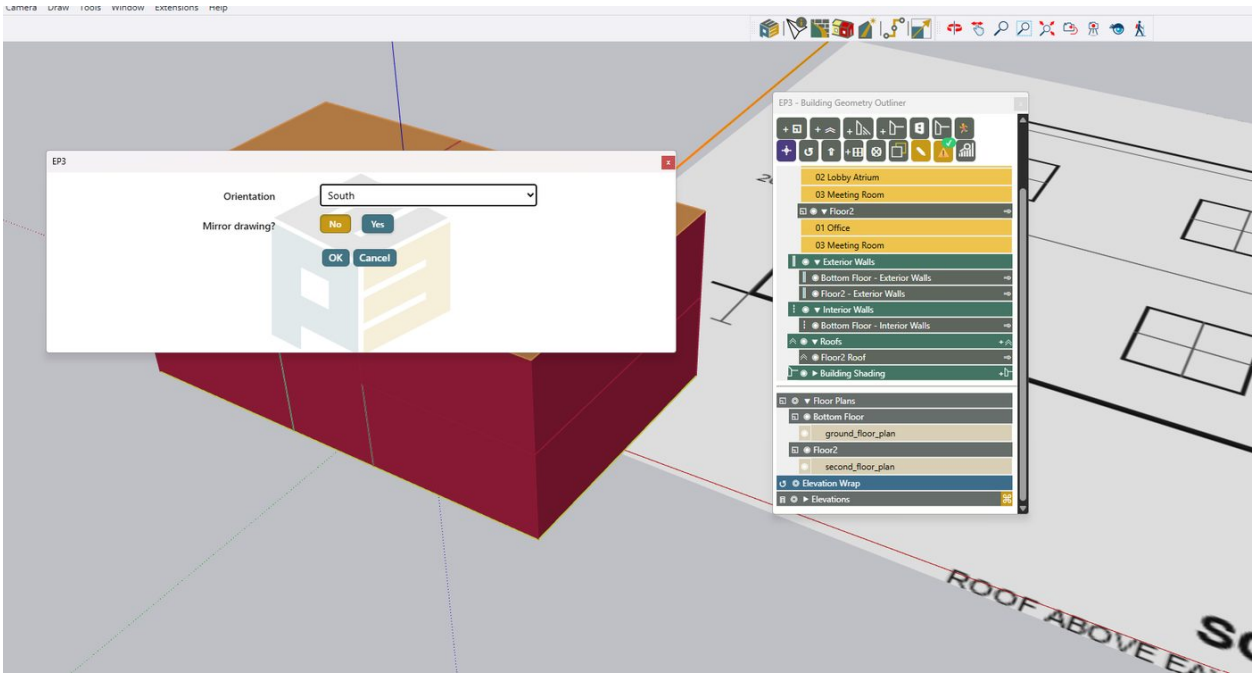


Figure 50: The elevation orientation prompt: Orientation and Mirror drawing?

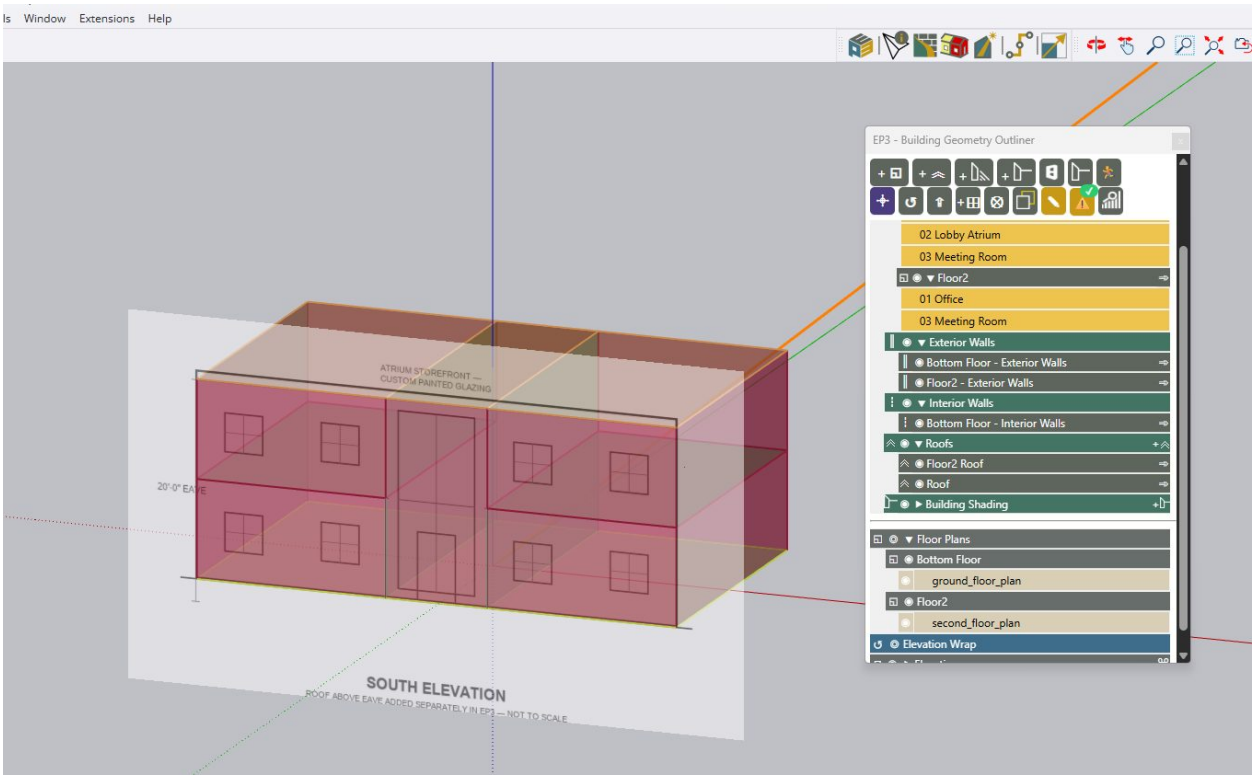


Figure 51: A positioned south elevation drawing standing against the model, semi-transparent, ready to be oriented and aligned.

own positioning tools rather than manually rotating it with Move — a manually rotated elevation keeps whatever direction it was originally assigned, so its recorded orientation and its actual facing fall out of sync.

"EP3 - Snap Elevations to Face" (right-click a wall or roof face) is deprecated — use Elevation Wrap (§12) instead to align an elevation drawing's window and door layout with the real wall geometry.

---

12. Elevation Wrap

Elevation Wrap projects your positioned elevation drawings onto the matching exterior walls and roof as a temporary, semi-transparent preview — a placement guide for windows and materials, not a change to the model. Per EP3's own description: *"Nothing here changes the model - it is a temporary, view-only preview… The wrap is selective - it will only wrap elevation drawings that are currently visible."*

It requires two things to already exist: the elevation drawings themselves (§11, currently visible), and the exterior wall/roof geometry they're projected onto (§5–6) — there's nothing to wrap onto before walls and roofs have been generated.

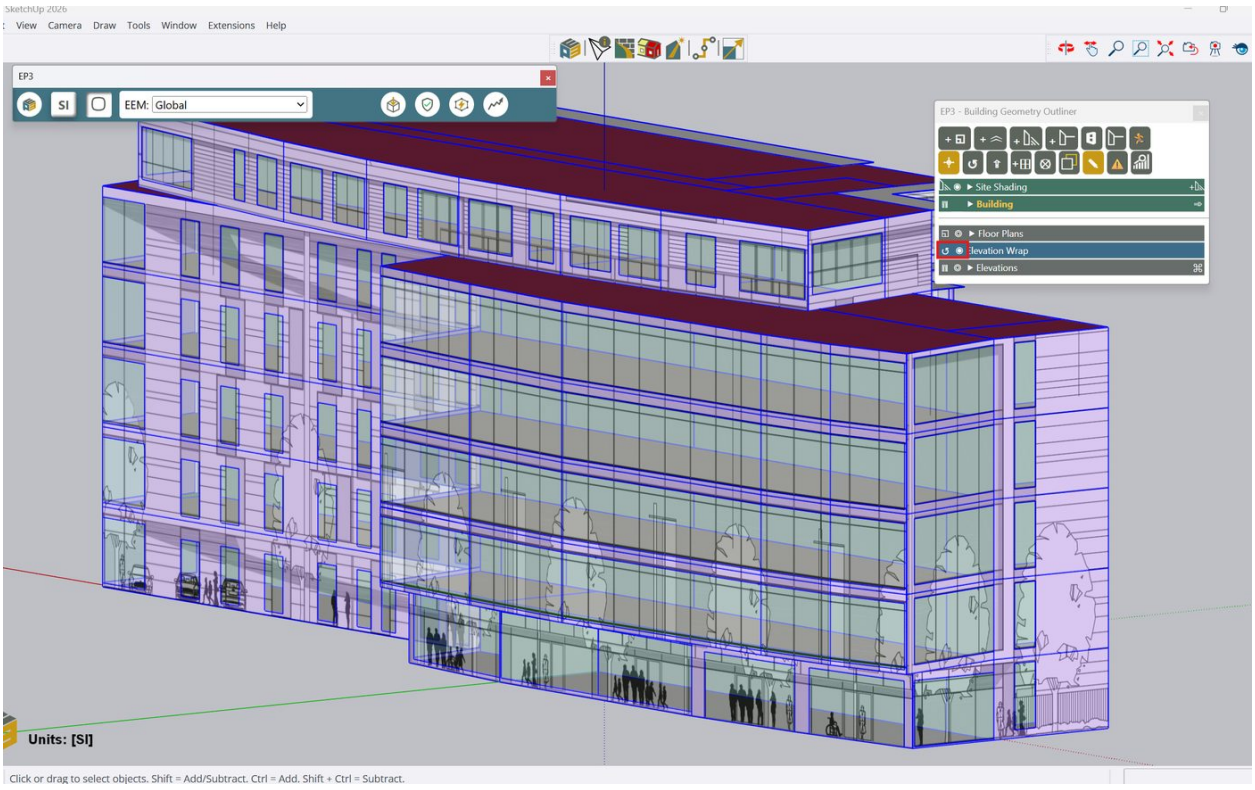


Figure 52: The Elevation Wrap row in the Building Geometry Outliner, with its refresh button (highlighted) and show/hide toggle.

Two controls handle it: a refresh button (recomputes the projection) and a show/hide toggle. Once wrapped, an elevation drawing's window and door layout sits directly on the real 3D wall, making it straightforward to paint a custom window (§7.2) in exactly the right place and at the right size.

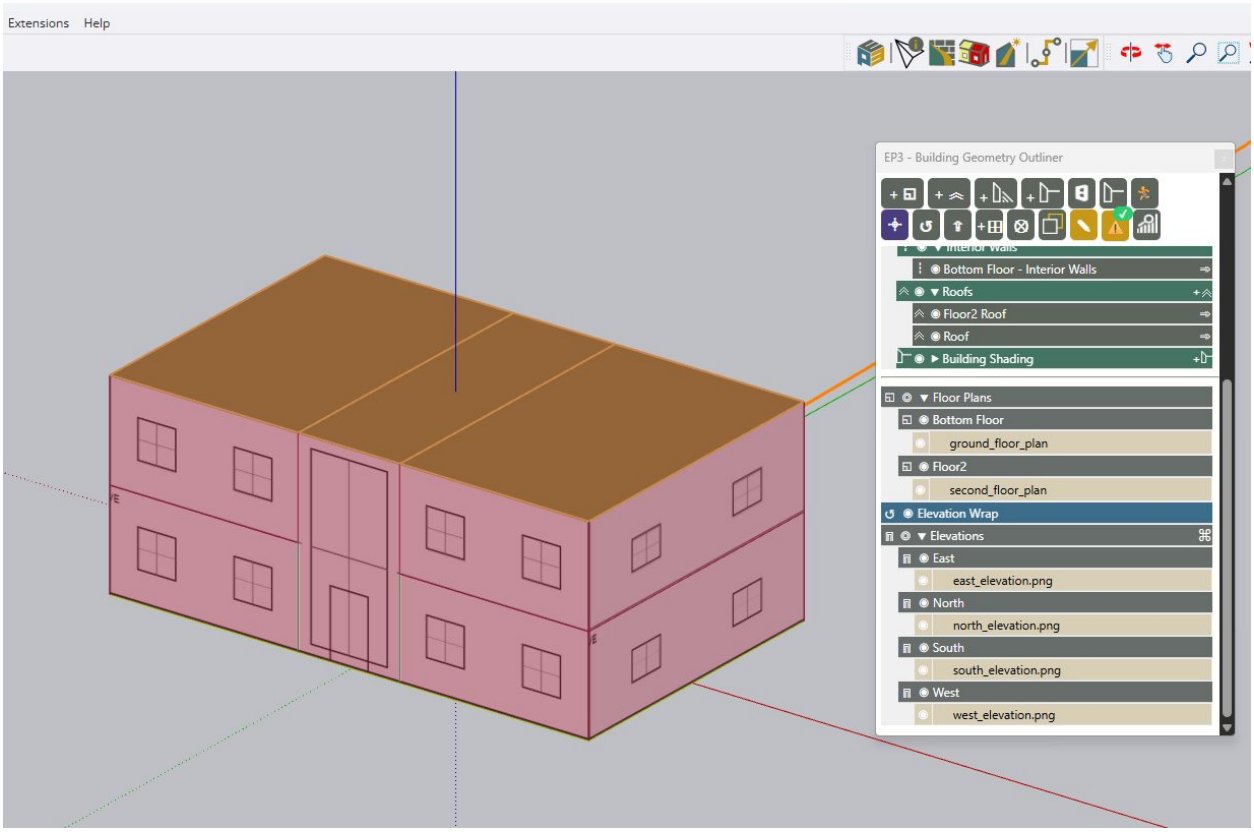


Figure 53: Elevation Wrap active on the worked example's building: the elevation drawings' window and door layouts projected directly onto the real 3D wall surfaces.

13. Diagnosing Geometry Problems: Unexpected Adjacency

The Building Geometry Outliner's adjacency-highlight toggle color-codes every surface with a potential problem:

- **Red** — no construction assigned.
- **Orange** — a boundary-condition mismatch: an exterior wall detected as adjacent to two spaces, or an interior wall detected as adjacent to only one.
- **Green** — the surface isn't matched to any Space at all, so EP3 is treating it as shading rather than a real building surface.
- **Blue** — the surface's boundary condition has been manually overridden.

Repainting a surface (§4) automatically refreshes this highlight for the repainted face, so the color updates immediately rather than waiting for a full rematch.

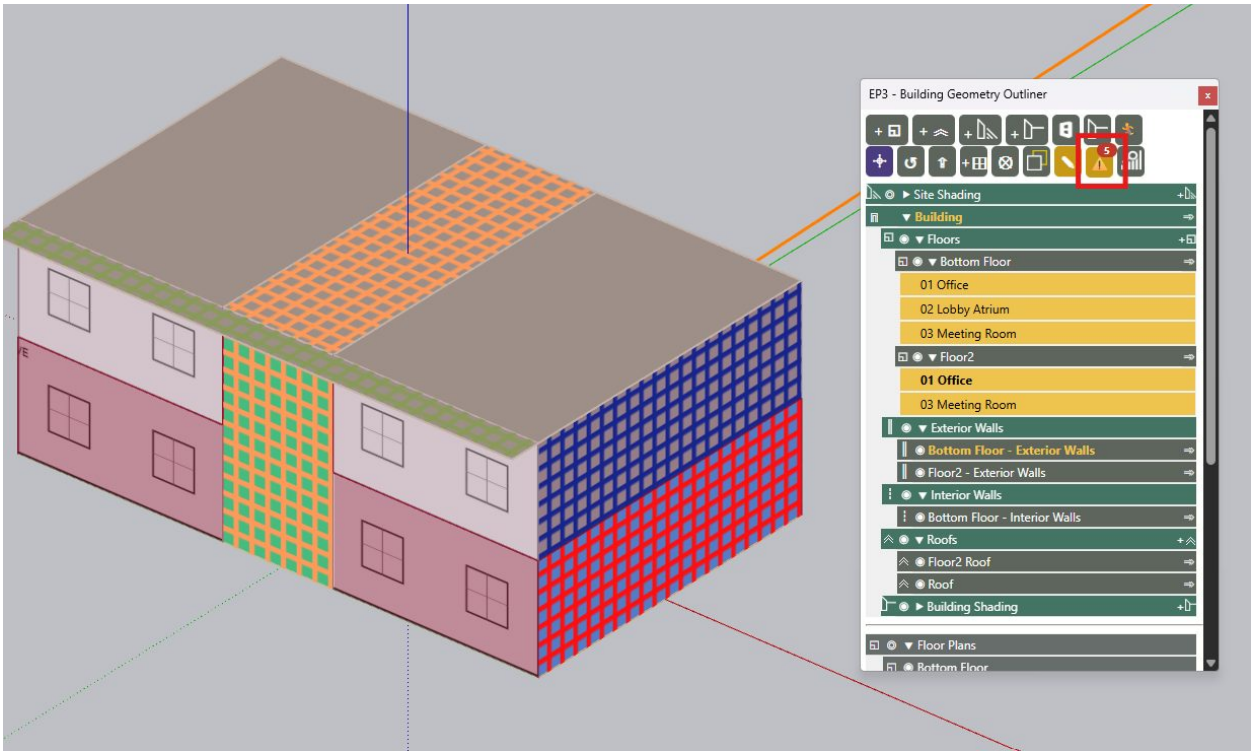


Figure 54: The unexpected-adjacency highlighter active on the worked example's building, with a counter badge on its toolbar button showing how many surfaces are flagged.

14. A Worked Example — A Two-Story Building with an Atrium

This example starts from the same Wizard selections as Chapter 3's worked example — Office Building - General internal loads, ASHRAE90_1 2022 CZ4 NonResidential constructions, PSZ-AC HVAC, a Climate Zone 4A weather file — but on the Building Geometry Properties tab, choose

"I want to draw my own building geometry" instead of the automatic rectangular option. From here, everything is built with the tools from this chapter.

14.1 The building

A 50 ft (E-W) × 30 ft (N-S) footprint: a 20 ft **Office** wing to the west and a 20 ft **Meeting Room** wing to the east, each two stories, flanking a 10 ft-wide double-height **Lobby** in the center. The second floor covers only the two wings — **Office 2** above the Office, **Office 3** above the Meeting Room — leaving the center bay open, so the Lobby reads as a full atrium from the ground floor to a single 20 ft flat roof spanning the whole building.

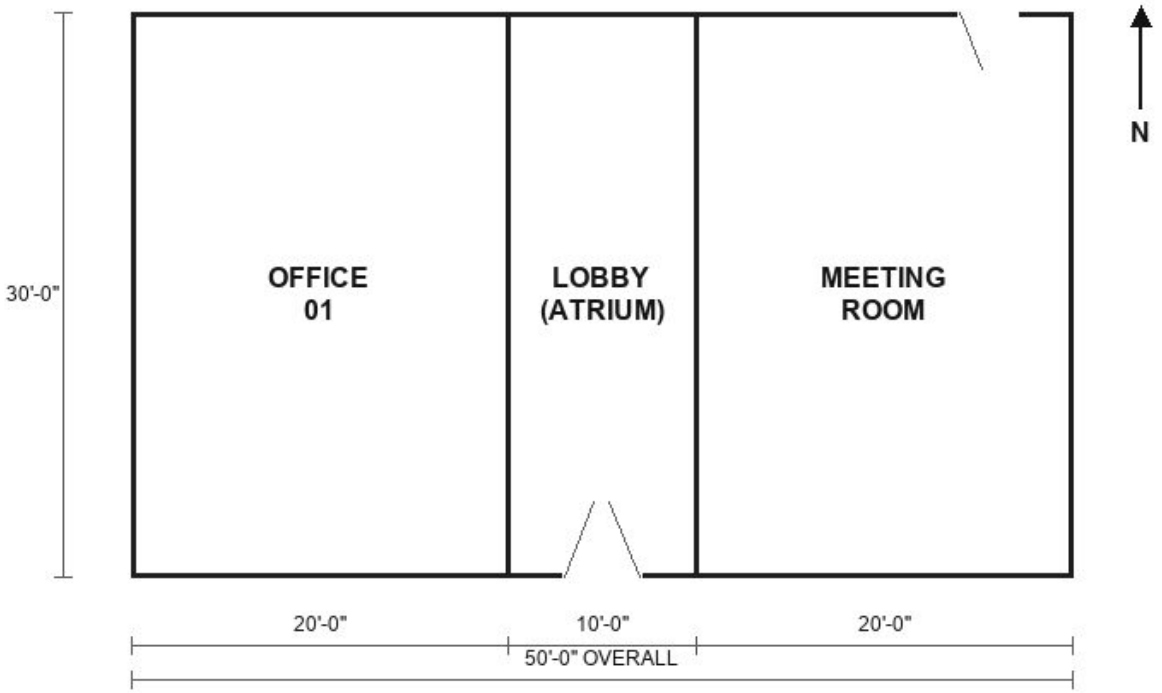


Figure 55: Ground floor plan for the worked example: a 20 ft Office wing, a 10 ft Lobby/atrium bay, and a 20 ft Meeting Room wing, each spanning the full 30 ft depth.

14.2 Floors, walls, and the atrium

1. Add the ground floor at elevation 0. Import the ground floor plan drawing above as a background (§10), position it, and trace its three room boundaries with the Line and Rectangle tools (§4) — Office, Lobby, and Meeting Room become three Spaces.
2. Add a second floor at a 10 ft floor-to-floor elevation. Import and trace the second floor plan drawing — but this time trace only the two wing footprints, leaving the center bay empty.

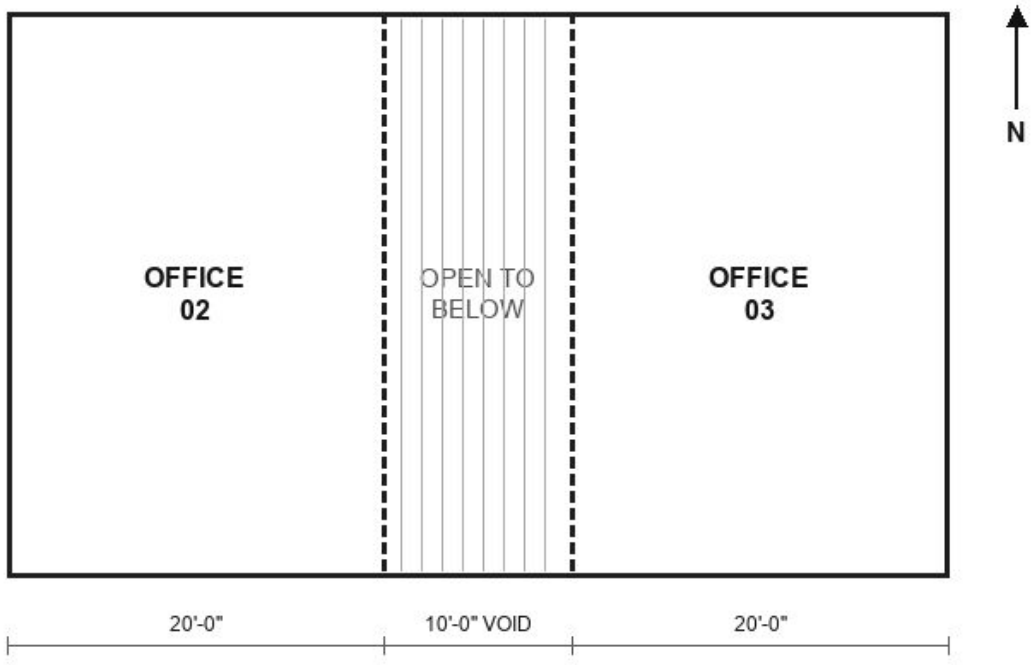


Figure 56: Second floor plan for the worked example: Office 2 and Office 3 occupy the same two wings; the 10 ft center bay is left open, shown here as the void above the Lobby.

Office 2 and Office 3 become two Spaces; nothing is drawn over the atrium.

3. On the ground floor (Bottom Floor as a whole), set **Floor Height Calculation Method** to **Specify Ceiling Height** and enter 20 ft. Leave the second floor at the default "Floor or Roof Above," so its walls stop naturally where its own roof forms.
4. Generate exterior and interior walls for both floors (§5), leaving "Generate Flat Roofs" at its default of Yes. The Office and Meeting Room walls extrude to 20 ft but get cut short at 10 ft where they hit the second floor's slab; the Lobby's walls, with nothing above them to stop at, run the full 20 ft in one piece — a two-story atrium, without doing anything beyond setting one field and not drawing a floor over that bay. This same step also generates the roofs: a flat roof for the second floor over the two wings, and the ground floor's own flat roof over the Lobby, both at 20 ft — together reading as one continuous flat roofline across the whole building, matching the elevation drawings above.

With two floors and their background drawings in play, the outliner navigation tools from §1 earn their keep: use each floor's **View Scene** button to jump the camera between the ground floor and second floor without re-orbiting by hand, and hide a floor's Exterior/Interior Walls rows while tracing so the background drawing and space boundaries stay visible without wall geometry in the way. Once both floors are traced, hide the second floor entirely — which cascades to hide its walls and drawing too — to check the ground floor's atrium opening from directly above without the second floor cluttering the view.

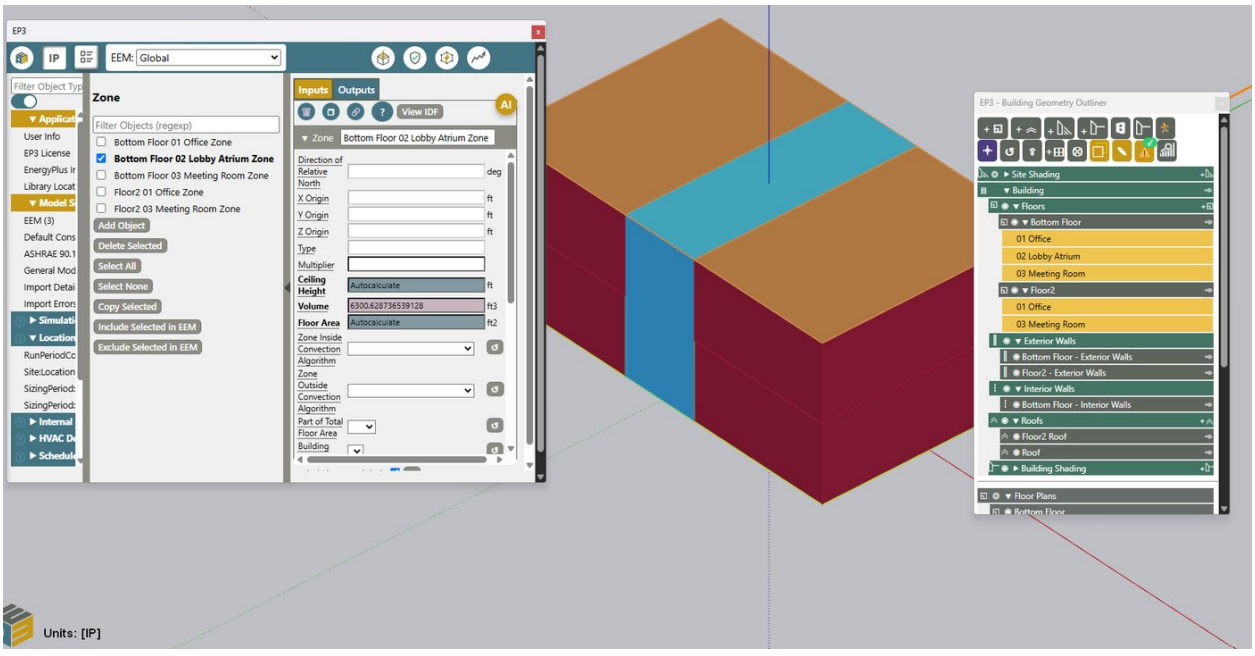


Figure 57: The building after §14.2: two-story wings flanking the Lobby's double-height atrium wall, with the flat roof already generated.

14.3 Windows, shading, and the atrium storefront

Import and orient all four elevation drawings following the instructions in (§11) - having all four in place lets you check the wrap against every facade.

Use Elevation Wrap (§12) to project the elevation drawings onto all four elevations, and use the

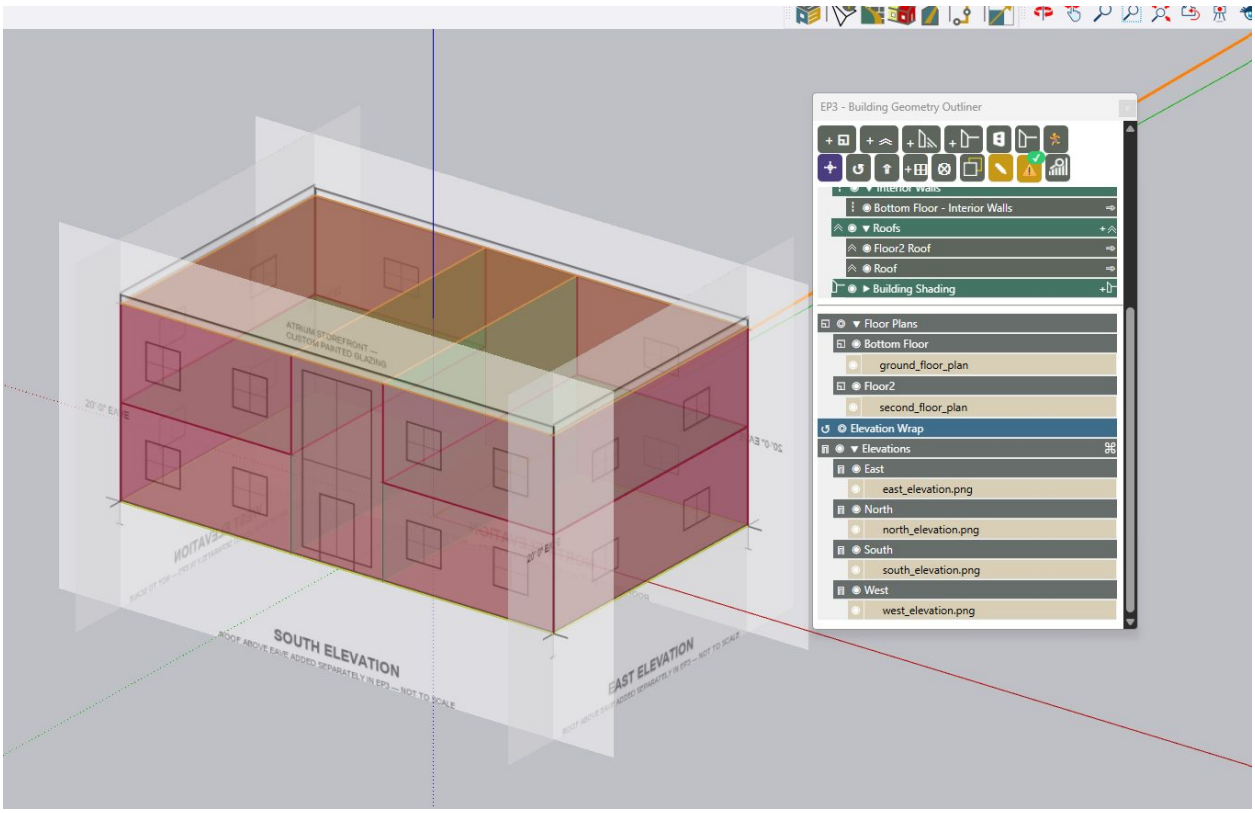


Figure 58: All four elevation drawings imported and oriented around the worked example's building, before wrapping.

resulting images as a placement guide for painting windows (§7.2) onto the real walls.

Paint the Office and Meeting Room's ground-floor windows onto the Bottom Floor's exterior walls first, matching the small square windows shown in the wrapped elevation guide. Since these are painted individually rather than placed by the WWR tool (§7.1), create an Opening:Type — a 4 ft × 4 ft type with a 3 ft sill height, named something like "sq win" — and assign it to each one via the window's own **Opening Type** field, so the repeated wing windows share one named size and construction instead of each being an unlinked one-off shape. Updating the **Opening Type** definition will update the window size and position of all windows assigned to this type. This is useful for making quick changes to many windows in your model.

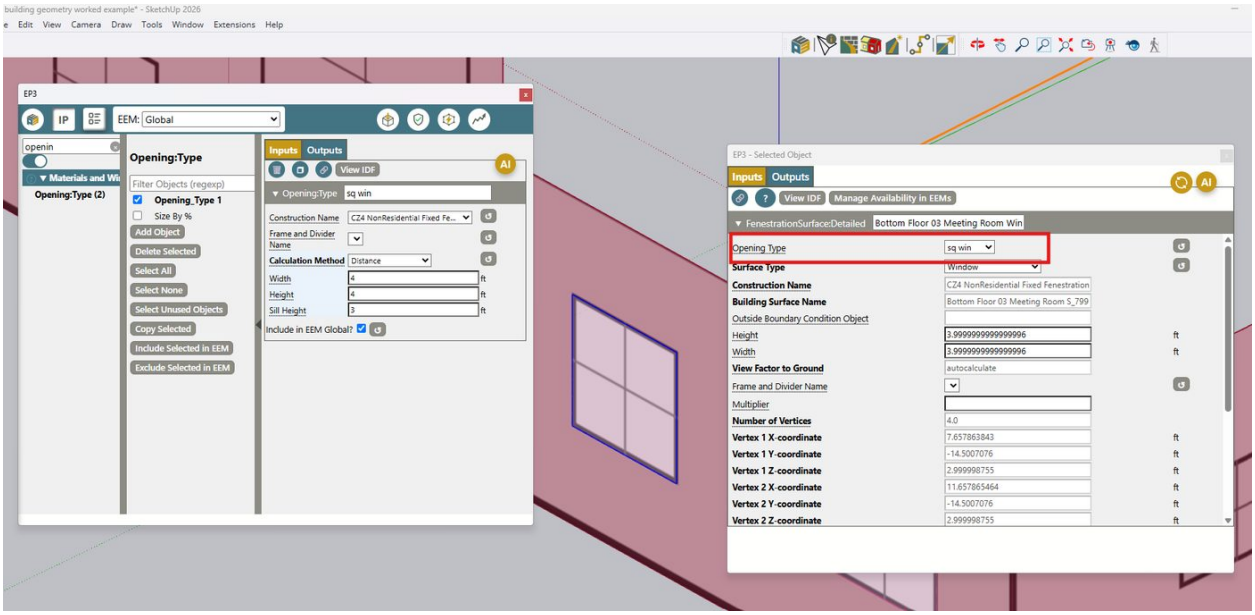


Figure 59: Creating an Opening:Type ("sq win") and assigning it to a painted window via that window's own Opening Type field.

Rather than repainting the same windows on Office 2 and the second-floor Meeting Room, use **"Copy Windows from Floor"** (§7.2) on Floor2's exterior wall group, copying from the Bottom Floor's — it replicates every ground-floor window upward at the correct elevation in one step.

The Lobby's south wall gets the full two-story storefront instead: paint its glazed area to match the wrapped south elevation (§7.2) — the glazing becomes a custom window from its painted outline. The drawing shows the entry doors as part of that same continuous glass wall rather than a solid punched opening, so create them as glass doors instead of opaque ones: right-click the wall and use **"EP3 - Add Window"** to drop in a window component, resize it with SketchUp's Stretch tool (**S**) to match the door opening, and set its `Surface Type` to `GlassDoor` rather than `Door` — an opaque door here would read, and simulate, as a solid patch in the middle of the glazed wall.

For an actual opaque door, look to the service door on the **north** elevation instead: place or paint that opening the same way, but leave its `Surface Type` as `Door`. Unlike the south storefront, the north wall has no surrounding glazing, so a plain opaque door there is exactly what the drawing shows.

Add a small **Building Shading** canopy (§8) near the top of the storefront's two-story glazing — at roughly the 20 ft roofline, and one simple neighboring low building as **Site Shading** (§8),

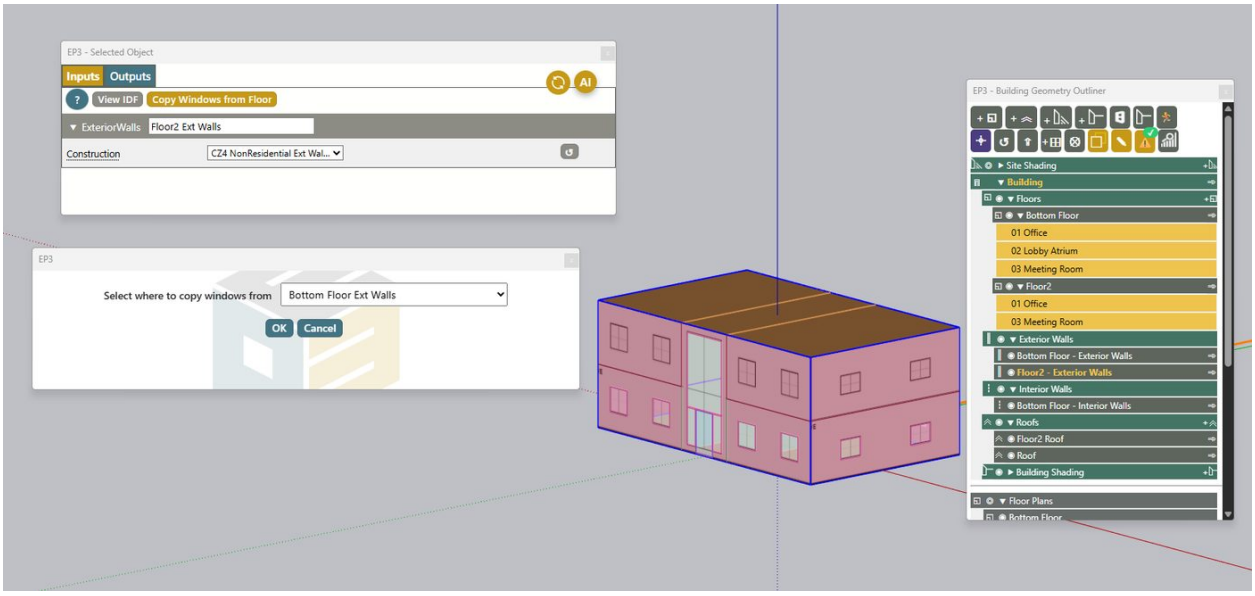


Figure 60: Using "Copy Windows from Floor" on Floor2's Exterior Walls to replicate the ground floor's windows upward, and the finished building with the wing windows and atrium storefront in place.

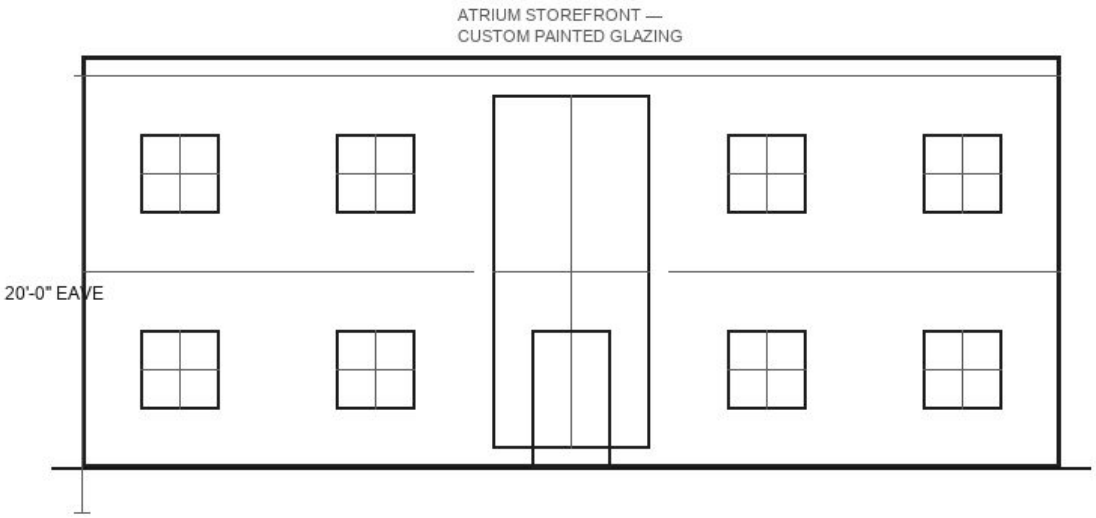


Figure 61: South elevation reference drawing for the worked example: standard punched windows on both wings, and the Lobby's full-height storefront glazing with entry doors below.

hand-drawn as a plain massing block to the east. §9 covers pulling in real neighboring-building context via geolocation instead of hand-drawing a placeholder block.

14.4 Checking the result

Turn on the unexpected-adjacency highlighter (§13) and resolve anything still shown in red (a surface with no construction assigned) before moving on. Everything else should read either unmarked (a normal, fully-resolved surface) or green at the Site Shading block, which is expected — shading massing normally doesn't match to any Space.

From here, the model is complete enough to assign HVAC equipment through the same Select Zone Conditioning Equipment dialog the Wizard opens, run a simulation, and review results exactly as in Chapter 3's worked example. *(see "Running Simulations")* and *(see "Viewing and Interpreting Results")* cover both in full.

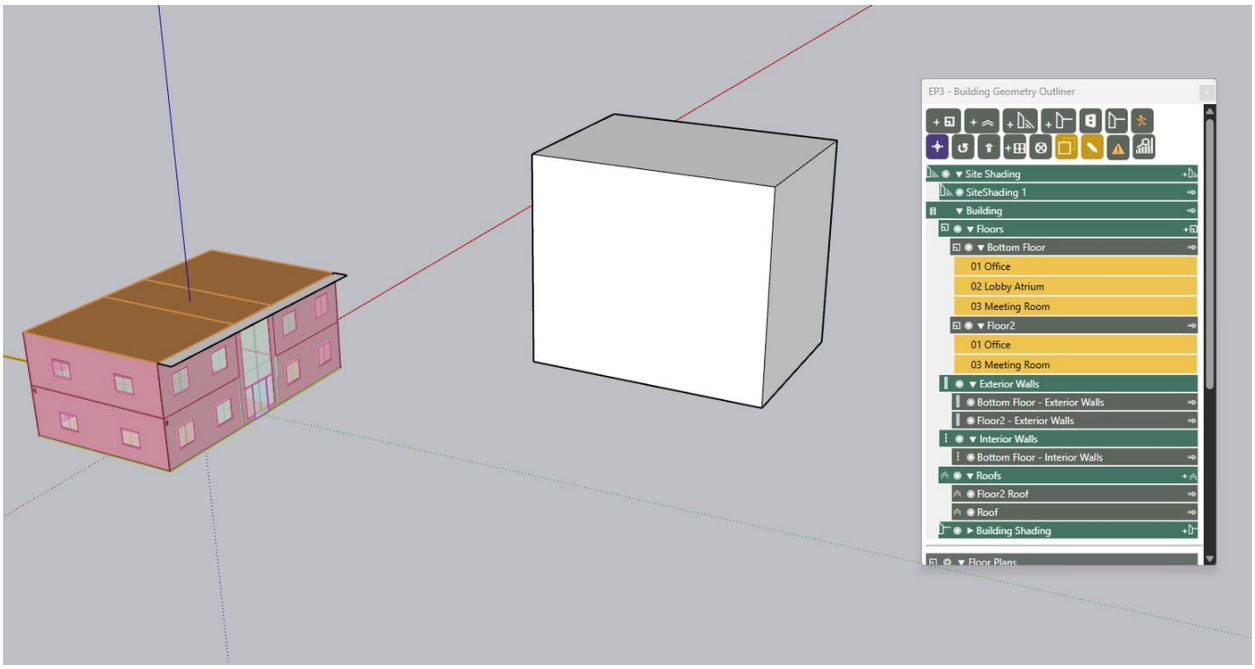


Figure 62: The finished worked example in the Building Geometry Outliner and 3D viewport: two-story wings flanking the double-height Lobby atrium, a single flat roof over the whole building, and the neighboring Site Shading block.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF placeholders once their target sections/pages exist; confirm the `use_shading_properties/ShadingProperty_Reflectance` *field's current status (it appears to short-circuit in the current code) before deciding whether it belongs in a future pass of §8 or the fields chapter.*

6. Spaces and Zones

1. Overview

EP3 supports both `Space` and `Zone`, EnergyPlus's two levels of building subdivision: a `Space` sits inside exactly one `Zone`, and a `Zone` can contain one or several Spaces. Supporting both gives a model two independent levels of resolution instead of one — the two objects answer different questions:

- A **Space** is where you assign internal loads and occupancy — people, lighting, equipment — and where floor area is tracked.
- A **Zone** is where HVAC equipment and thermostats are assigned.

When creating building geometry, users draw Spaces (*(see "Creating Building Geometry")*). EP3 automatically creates one Zone for each Space created. Most models don't need to change this relationship. This chapter covers that default, and the mechanics and reasons for deviating from it — grouping several Spaces into one shared Zone.

For information on `SpaceType` (the reusable bundle of internal-loads defaults assigned to a Space) and `ZoneType` (the reusable bundle of HVAC/sizing defaults assigned to a Zone), see *(see "Internal Loads, Space Types, and Zone Types")*. Those govern the internal loads, thermostat, and sizing a Space or Zone *inherits*; this chapter covers the structural relationship between the two objects themselves.

---

2. The default: one Zone per Space

A newly created Space is automatically given its own Zone — there's no separate step to create a Zone for the ordinary case. The Space's "Zone Name" field simply points at that automatically created Zone from the start.

This means a model with, say, twelve Spaces has twelve Zones by default.

---

3. Merging Spaces into a shared Zone

3.1 Reasons to merge Spaces into a shared Zone

Finer control over internal loads and occupancy within one thermal zone. Internal loads are usually specified as a density — people per unit area, watts per unit area — applied uniformly across a Space. If one area of a room has meaningfully different occupancy or equipment density than another, splitting it into two Spaces lets you assign each its own density (a high-density huddle area and a low-density open desk area, for example) while still simulating both as a single Zone thermally. This is the most common reason to have more than one Space feeding a single Zone.

Reducing zone count for cost. There are also legitimate cases where Spaces that would ideally each get their own Zone — e.g. different envelope exposure or occupancy type — get grouped into a shared Zone. This is a common cost-cutting measure. Modeling the individual spaces and combined zone allows analysis of thermal comfort parameters in each space and a more accurate understanding of the human and comfort tradeoffs of the cost-cutting measure.

3.2 Mechanics

Select the Space you want to reassign. Its properties show a **"Zone Name"** field, which lists every existing Zone in the model. Choosing a different Zone from that list moves the Space into it — the Space's own internal-loads and occupancy assignments are unaffected; only which Zone simulates it thermally changes.

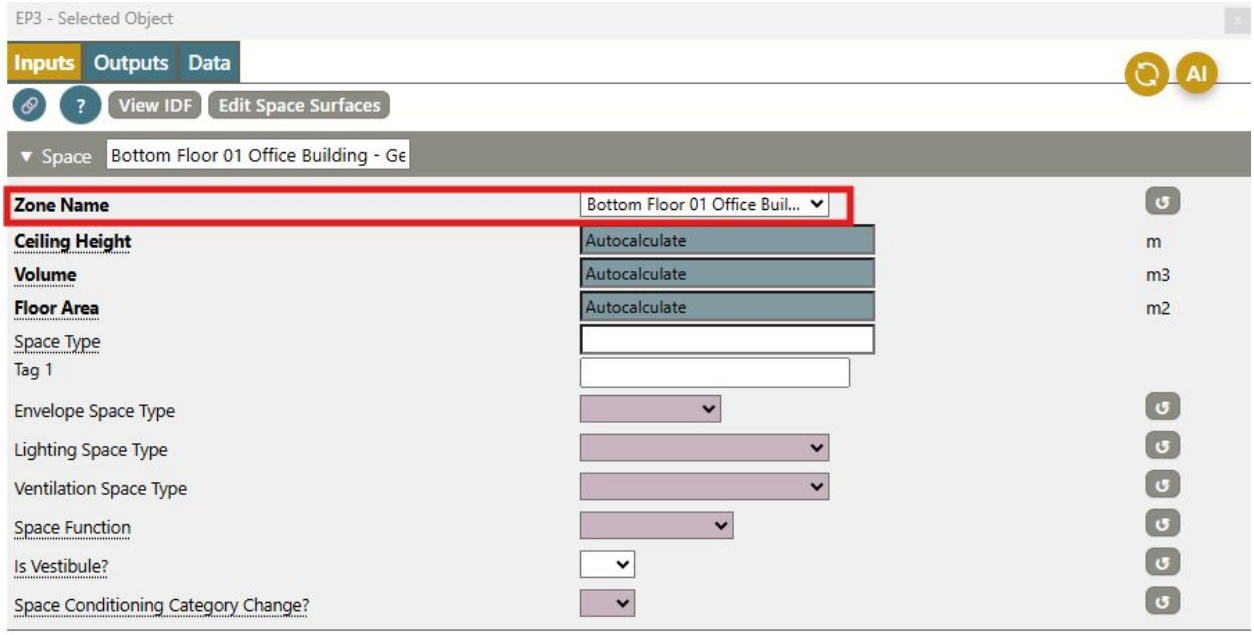


Figure 63: A selected Space's properties dialog, with the "Zone Name" field open showing a drop-down list of existing Zone names in the model.

A Zone left with no Spaces assigned to it (because every Space that used to point to it was moved elsewhere) isn't flagged as a model error — it's simply excluded from the exported EnergyPlus model.

---

4. Zone `multiplier`, not the same thing

EnergyPlus Zones have their own aggregation mechanism independent of this chapter's topic: a Zone's **Multiplier** field represents a number of *identical, unmodeled* copies of that one Zone — used when a building has many physically identical zones (repeated floors or repeated units) and modeling each individually would be redundant.

Don't confuse this with a shared Zone containing several Spaces (§3): Multiplier scales up one modeled Zone to stand in for several unmodeled duplicates of itself, while a shared Zone combines several distinct, individually-modeled Spaces into one thermal simulation unit. See *(see "Sizing HVAC Systems and Zones")* for Multiplier in full.

End of draft chapter. Outstanding items for the next pass: supply the screenshot marked above;

resolve the XREF placeholders once their target chapters exist; confirm final document format (HTML vs. PDF vs. other).

7. Constructions and Materials

1. Overview

Every opaque or glazed surface in an EP3 model — walls, roofs, floors, doors, windows — carries a **construction**: the layered assembly (or, for a few specialized cases, a single calculated value) that determines its thermal performance. In EP3, a construction is a Sketchup material. You assign it to geometry the same way you'd assign any material — by painting — and its EnergyPlus behavior comes from attributes stored on that material, editable through the Constructions dialog.

A construction's **Construction Type** determines which EnergyPlus object it becomes and which fields it exposes:

Construction Type	What it represents
`Construction`	The general case: a stack of up to 10 material layers, outside to inside. Used for both opaque assemblies and simple glazing, depending on whether "Construction is Glazing?" is checked.
`Construction:FfactorGroundFloor`	A slab-on-grade or other ground-contact floor, characterized by a single F-Factor rather than a layer stack.
`Construction:CfactorUndergroundWall`	A below-grade wall, characterized by a single C-Factor rather than a layer stack.
`Construction:ComplexFenestrationState`	A BSDF-based fenestration construction for complex shading/glazing systems.
`Construction:WindowDataFile`	A construction pulled from an external Window5/Window6 data file rather than built from layers in EP3.
`Construction:AirBoundary`	An adjacency between two zones with no thermal resistance, optionally mixing air between them.

This chapter covers building and assigning each of these, the material layers that make up a `Construction`, the EP3-specific perimeter workflow for F-Factor/C-Factor constructions, and how to package constructions into a reusable library. Model-wide default constructions — the "whatever construction gets auto-painted when you draw new geometry" setting, one per surface category — are covered in *(see "The EP3 Interface and Model Settings", §8.1)*.

Window and door *openings* — placing them in geometry, and the shading/daylighting controls attached to them — are covered in *(see "Windows, Shading, and Daylighting")*. This chapter covers the glazing constructions an opening's own Construction Name field points at, not the opening object itself.

1.1 Paint is persistent across EEMs — construction parameters are what change

A quick primer, covered in full in *(see "Energy Efficiency Measures", §1)* later in this manual: every EP3 model has a general model space, called **Global**, plus any number of named **Energy Efficiency Measures** (**EEMs**) layered on top of it. Switching the active EEM (from the EEM dropdown in the Top Control Strip) changes the model context - objects can be included / exculded

in an **EEM**, and field values can be changed. This is how one model can represent both a baseline design and one or more proposed designs without keeping a separate copy of the building for each.

However, which construction is painted onto a given surface, is *not* something that can be changed between **EEMs**: a wall painted with "Ext Wall Construction" stays painted with that same construction in every EEM. There's no per-EEM repaint.

To represent a different assembly in an EEM, switch to that EEM and change the *parameters* of the construction object instead: edit its layer stack (a layer's Thickness, Conductivity, or Thermal Resistance — §3), swap one layer's material for another, change its F-Factor/C-Factor/SimpleGlazingSystem fields (§5, §6) directly, or even change its Construction Type itself (§2.1). Because every surface sharing that construction is painted with the identical Sketchup material, an EEM-level override on any of these fields applies to every surface that construction touches at once — you don't need to find and repaint each one.

For example, an EEM modeling added wall insulation doesn't repaint the exterior walls with a new "Insulated Wall Construction" — it switches to that EEM and edits the existing exterior wall construction's insulation layer's Thermal Resistance field directly. *(see "Energy Efficiency Measures", §4.1)* (later in this manual) works through exactly this case as part of a larger EEM-chaining example: a "Wall `Material:NoMass` R-value → R-19" change, layered together with an unrelated lighting-power-density change to show how two EEMs combine.

If an EEM's intent is really "these specific surfaces get a different assembly than the rest of the exterior walls" rather than "the exterior wall assembly changes everywhere it's used" — paint a distinct construction for those surfaces in Global, and then change the parameters of that construction in the target EEM.

---

2. Creating a construction

From the Constructions dialog, **Create Construction** prompts for a name, a Construction Type, and a boundary condition, then creates a new Sketchup material with those attributes attached. Painting that material onto geometry assigns the construction.

Construction Adjacency (the boundary condition field) accepts `Outdoors`, `Interior`, `Ground`, `Adiabatic`, or `GroundFCfactorMethod`. EP3 suggests a starting value based on the construction's name — a name containing "ext" or containing "roof" suggests Outdoors, a name containing "int" suggests Interior, a name containing "ground" suggests Ground — but this is only a starting suggestion, not an inferred fact; confirm it matches the surface's actual condition. **Construction Adjacency** is also used by EP3 to confirm that the automatic space adjacency detection is correct — when a surface that is flagged as `Interior` finds only one adjacent `Space`, EP3 flags it via the unexpected-adjacency highlighter so the modeler can inspect and resolve the issue. See *(see "Creating Building Geometry", §13)* for that highlighter's full color-coding and workflow.

2.1 Changing Construction Type without repainting

Construction Type is a dropdown on the construction itself, not a one-time choice made at creation — because a construction is a wrapper around a single Sketchup material, changing its type reclassifies that same material in place, without needing to repaint any surface it's already assigned to.

For example, suppose you built an underground wall as a plain `Construction`, with a boundary condition of `Ground`, a hand-built layer stack modeling the wall assembly and soil resistance — and later decide you'd rather let EnergyPlus's C-Factor method handle the ground-coupling calculation instead. Changing that construction's Construction Type to `Construction:CfactorUndergroundWall` swaps its fields to C-Factor and Height (§4) in place; every surface already painted with it keeps that assignment automatically. (EP3 also forces the boundary condition to `GroundFCfactorMethod` as part of this switch — see §4.)

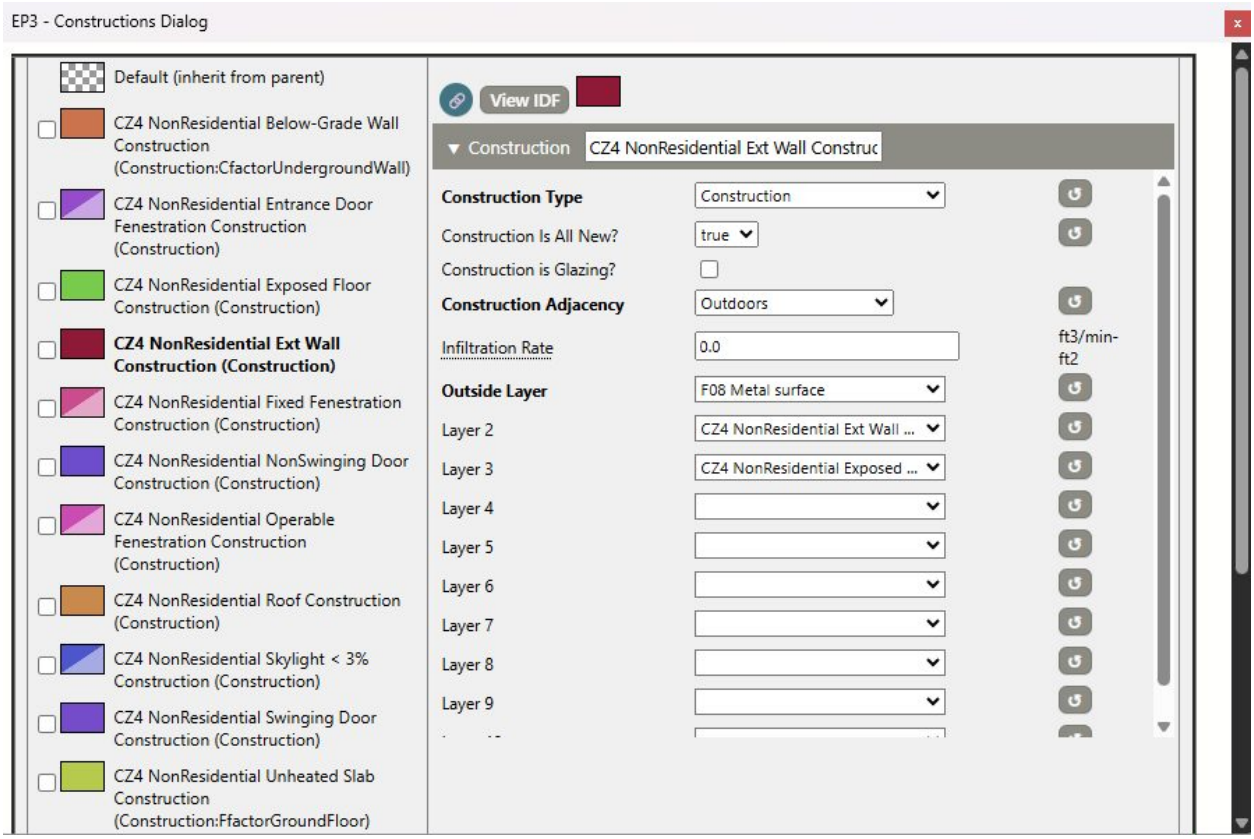


Figure 64: The Constructions dialog for an exterior wall Construction, with Construction Type and Construction Adjacency visible above its material layers.

---

3. Material layers

A `Construction`'s layer stack runs from **Layer: Outside Layer** through up to nine additional layers, outside to inside. Each slot can hold one of:

- `Material` — the general case: **Thickness**, **Conductivity**, **Density**, **Specific Heat**, plus **Roughness**, **Thermal Absorptance**, **Solar Absorptance**, and **Visible Absorptance** (the last three default to 0.9, 0.7, and 0.7 and rarely need changing). Resistance is derived from thickness and conductivity. Material
- `Material:NoMass` — for a layer you want to characterize by resistance alone (e.g., a manufacturer-rated batt insulation) rather than by thickness and conductivity: just **Thermal Resistance**, plus the same Roughness/Absorptance fields. Material:NoMass

- `Material:AirGap` — an unventilated air space, specified by **Thermal Resistance** alone. Material:AirGap
- `WindowMaterial:*` — for glazing-mode layers only: `WindowMaterial:SimpleGlazingSystem` (§5.1) WindowMaterial:SimpleGlazingSystem, or a physically-detailed pane-by-pane stack built from `WindowMaterial:Glazing` WindowMaterial:Glazing, `WindowMaterial:Gas` WindowMaterial:Gas / `WindowMaterial:GasMixture` WindowMaterial:GasMixture, and `WindowMaterial:Gap` WindowMaterial:Gap. Shade, blind, and screen `WindowMaterial:*` types support Complex Fenestration State assemblies (§5.2) rather than an ordinary glazing `Construction`.

Full field-by-field detail for each — valid ranges, exactly what Roughness affects, etc. — is in the EnergyPlus Input/Output Reference (linked above per type); EP3 doesn't add behavior on top of these fields beyond what's described here.

EP3 enforces layer-type consistency with how the construction's alpha (glazing) flag is set: an opaque construction (alpha off) can't contain a `WindowMaterial:*` layer, and a glazing construction (alpha on) can only contain `WindowMaterial:*` layers — the Constructions dialog's material-picker for each layer slot only offers the layer types valid for the construction's current glazing state, and the model-wide Validation check (§4) flags any construction where a layer type doesn't match.

Every construction — regardless of type — also carries an **Infiltration Rate** field (entered per unit area), an advanced option most projects won't need. Setting infiltration at the SpaceType level (*(see "Internal Loads, Space Types, and Zone Types")*) is sufficient for the large majority of projects. Construction-level infiltration exists for the case a single SpaceType-wide rate can't represent. For example, a building with exterior walls of noticeably different construction integrity across the same space, a project that's gone through multiple phases of construction and renovation — where infiltration should scale with the exposed area of each construction type rather than apply uniformly across the space. Entering a value here doesn't do anything on its own; it only takes effect once a `ComponentInfiltration` object is also created, which is what actually aggregates these per-construction rates into the zone's infiltration calculation (*(see "Infiltration, Natural Ventilation and Airflow Networks")*).

3.1 Where layer materials come from

Unlike a construction, a bare `Material`/`Material:NoMass`/`Material:AirGap` object has no "Create New" button of its own in the Constructions dialog — a layer slot's dropdown only offers materials that already exist somewhere in the model. You get them into the model one of two ways: importing a materials library, which brings in a construction's referenced materials along with it (§8), or pasting EnergyPlus IDF text for the object directly through EP3's general Paste dialog. In practice, most modelers get their layer materials as part of a library import rather than typing one in from scratch.

Once a layer slot is populated, clicking directly on that field's own label (not the dropdown) in the Constructions dialog navigates straight to the referenced material, so you can edit its Thickness/Conductivity/Resistance/etc. fields in the same dialog without hunting for a separate materials list.

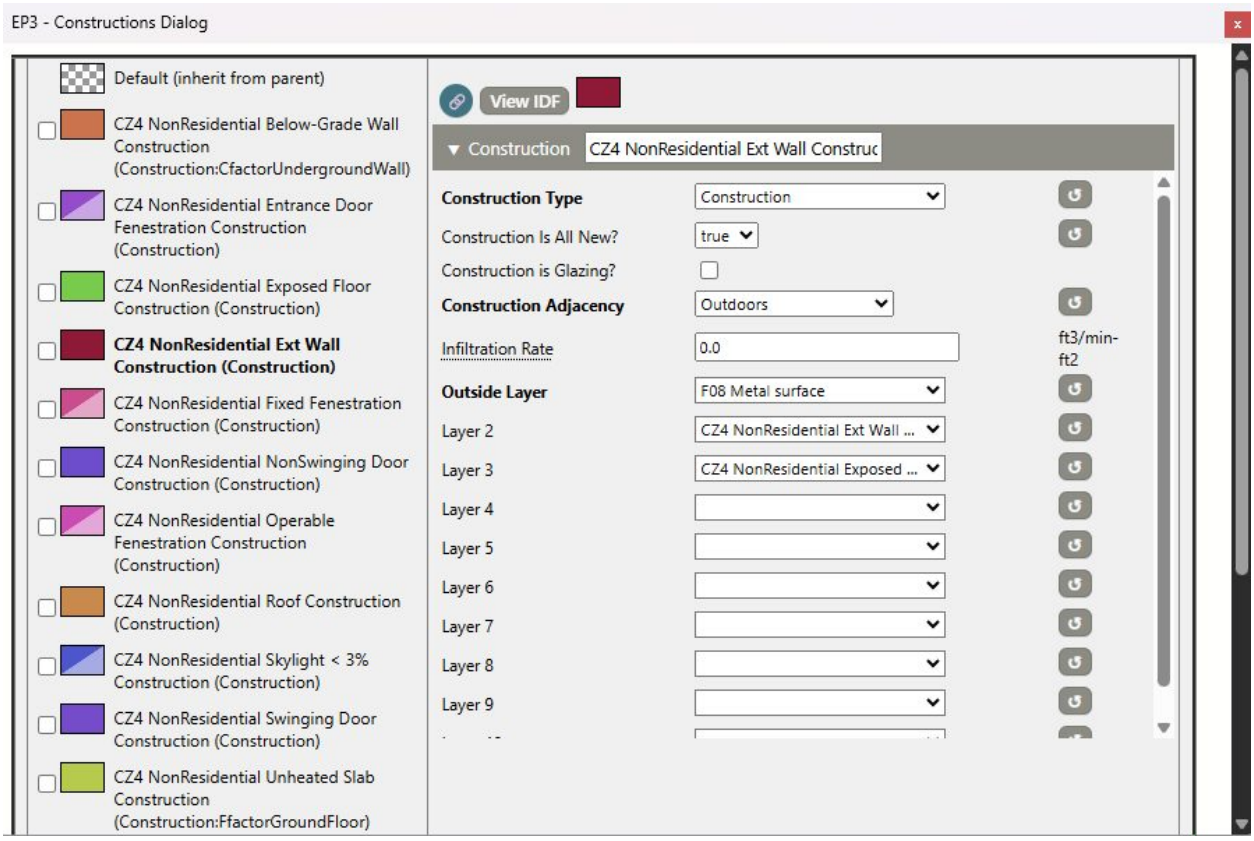


Figure 65: The Constructions dialog for an exterior wall Construction, showing its populated material layers.

4. Opaque construction performance, worked example

EP3 calculates a total R-value for every opaque construction (one without the glazing flag set) and surfaces it in the Validation dialog's **Opaque Construction Performance** check, so you can confirm a layer stack produces the assembly performance you intended without leaving EP3. The calculation sums each layer's resistance — thickness ÷ conductivity for a `Material` layer, the entered value directly for `Material:NoMass` or `Material:AirGap` — and reports "performance could not be calculated" for any layer type it doesn't recognize (anything other than those three).

Worked example. Suppose you build a wall from three illustrative layers:

Layer	Type	Values	Resistance
Outside Layer — wood siding	`Material`	Thickness 0.019 m, Conductivity 0.109 W/m-K	0.019 ÷ 0.109 ~ **0.174 m² · K/W**
Layer 2 — batt insulation	`Material:NoMass`	Thermal Resistance entered directly	**2.290 m² · K/W** (an IP-rated "R-13" batt, 13 ÷ 5.678 ~ 2.29 in SI)
Layer 3 — gypsum board	`Material`	Thickness 0.0127 m, Conductivity 0.160 W/m-K	0.0127 ÷ 0.160 ~ **0.079 m² · K/W**

Total R-value = 0.174 + 2.290 + 0.079 ~ 2.54 m² · K/W

Source: this mirrors the calculation EP3 itself performs (`ConstructionWrap.calculate_opaque_r_value`) — summing thickness/conductivity for `Material` layers and reading resistance directly for `Material:NoMass`/`Material:AirGap` layers.

Check, converted to IP: 1 $m^2 \cdot K/W$ ~ 5.678 $ft^2 \cdot °F \cdot hr/Btu$, so 2.54 × 5.678 ~ **14.4 ft² · °F · hr/Btu (R-14.4)** — whichever unit system your model displays in, EP3's Opaque Construction Performance table reports the same underlying value converted to match.

---

5. Glazing constructions

Checking a construction's **Construction is Glazing?** box switches it into glazing mode: Sketchup renders it semi-transparent, and its layer slots restrict to `WindowMaterial:*` types.

5.1 `WindowMaterial:SimpleGlazingSystem`

The simple-glazing path takes performance data directly rather than deriving it from physical layer properties — three fields, matching what a manufacturer's NFRC label typically reports:

- **U-Factor** — entered *including film coefficients* (i.e., the whole-assembly U-factor, not a center-of-glass value).
- **Solar Heat Gain Coefficient** — SHGC at normal incidence.
- **Visible Transmittance** — VT at normal incidence; optional.

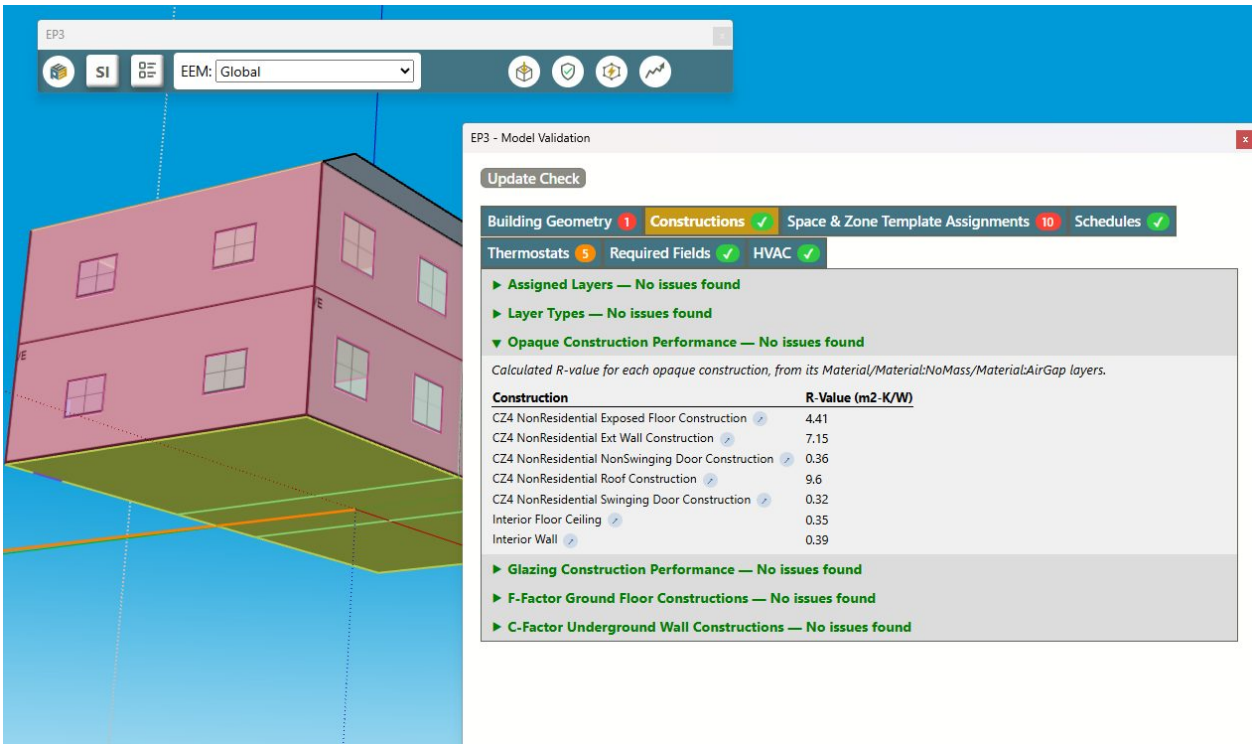


Figure 66: The Validation dialog's Opaque Construction Performance table, showing calculated R-values.

Full field definitions and applicable ranges are in WindowMaterial:SimpleGlazingSystem. As with the opaque materials in §3.1, a `WindowMaterial:SimpleGlazingSystem` object isn't created from a blank form — it arrives via a materials library import or the Paste dialog (§8), then gets picked in a glazing construction's layer dropdown like any other layer material.

A single `WindowMaterial:SimpleGlazingSystem` layer is normally the construction's only layer. Because you enter U-Factor, SHGC, and VT directly rather than building them up from other inputs, no derivation is needed — if your source data is in IP units (a typical NFRC label reports U-Factor in Btu/h · ft² · °F), enter it in whichever unit system EP3 is displaying; EP3 converts and stores it in SI internally.

EP3's Validation dialog reports the same U-Factor/SHGC/VT for each glazing construction in its **Glazing Construction Performance** table, useful for scanning many glazing constructions at once. This table currently only evaluates a construction built from a single `WindowMaterial:SimpleGlazingSystem` layer — anything more complex is flagged as not yet calculable rather than evaluated incorrectly.

5.2 `Construction:ComplexFenestrationState` and `Construction:WindowDataFile`

For fenestration systems the simple-glazing model can't represent — e.g., a shading-integrated glazing unit characterized by a full bidirectional scattering distribution function (BSDF) — `Construction:ComplexFenestrationState` takes a matrix-based description instead of a layer stack. Its full field set is extensive; see Construction:ComplexFenestrationState for the complete input specification. `Construction:WindowDataFile` is narrower: it points at an external

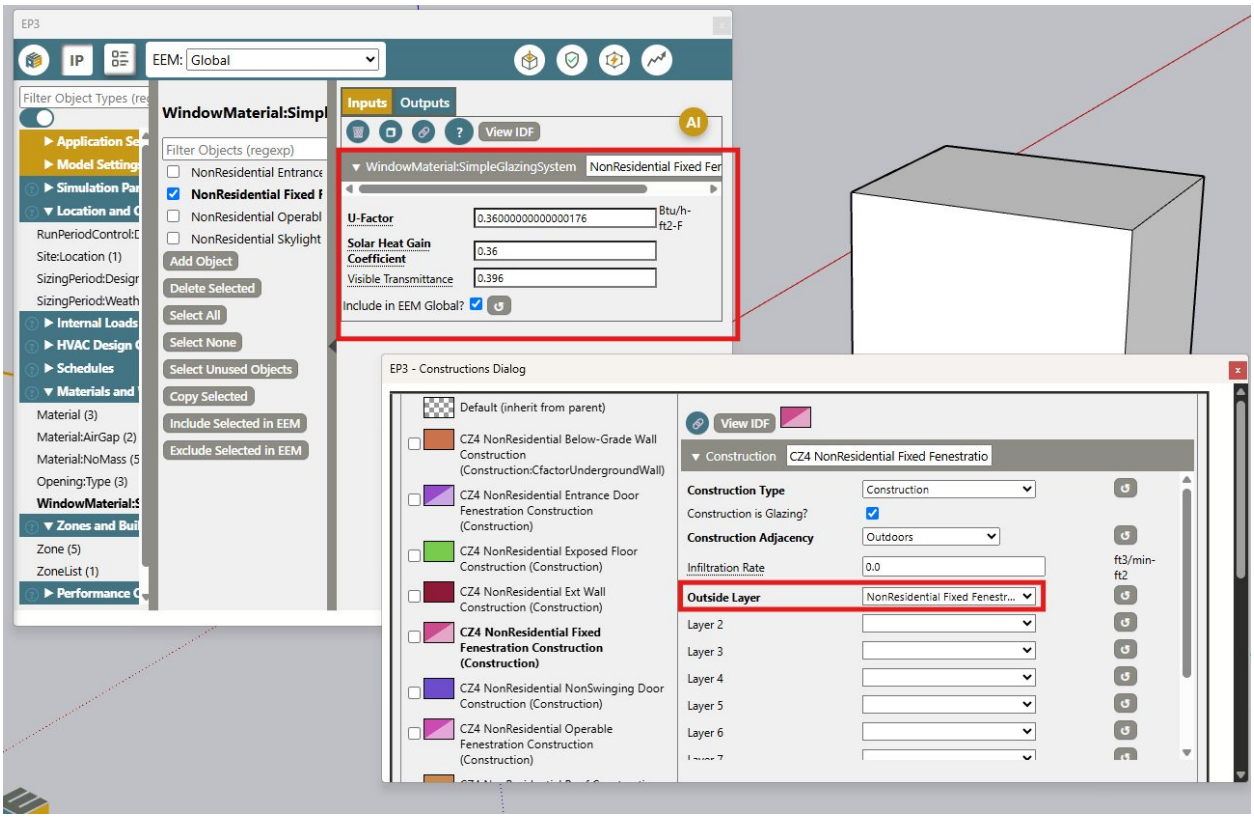


Figure 67: A glazing Construction in the Constructions dialog, with its WindowMaterial:SimpleGlazingSystem layer shown in the background.

Window5/Window6 data file (**File Name**, defaulting to `Window5DataFile.dat`) rather than describing the assembly in EP3 at all.

Both are available as Construction Type choices from the same Constructions dialog as every other construction type, but neither is built up from EP3 material layers the way `Construction` is.

5.3 How a window picks its glazing construction

A window gets its construction the same way any other surface does — by painting. Painting the window's own component directly assigns it, exactly like a wall or roof; the **Construction Name** shown on a window's own properties is read-only, a live reflection of whatever's currently painted rather than an independently editable field. In practice a window's paint is usually set one of three ways: hand-painting it directly, choosing a Glass Construction when placing it with Add Windows By Exposure, or assigning it an Opening Type, which repaints every window referencing that type automatically. Placing openings, and the shading/daylighting controls and Opening Type mechanism that attach to them, are covered in *(see "Windows, Shading, and Daylighting")*.

---

6. Ground-contact constructions: F-Factor and C-Factor

`Construction:FfactorGroundFloor` and `Construction:CfactorUndergroundWall` replace a layer stack with a single EnergyPlus-calculated ground-coupling method.

`Construction:FfactorGroundFloor` fields: **F-Factor** (entered without film coefficients or soil resistance included), **Area**, and **PerimeterExposed**. `Construction:CfactorUndergroundWall` fields: **C-Factor** (also without film coefficients or soil) and **Height**.

Area and PerimeterExposed aren't hand-entered in EP3 — both fields are locked and calculated automatically from geometry, using the same perimeter-tracking mechanism described next.

6.1 Assigning the exposed perimeter

Because PerimeterExposed depends on which edges of a slab actually border the outdoors (as opposed to another heated space), EP3 provides a dedicated workflow, available whenever a `Construction:FfactorGroundFloor` construction is selected in the dialog:

- **Automatically Assign Perimeter** — scans every face painted with this construction and marks each edge that borders only one face (i.e., an outer boundary edge) as an exposed-perimeter edge.
- **Add Perimeter Edges** — marks the currently selected Sketchup edges as exposed perimeter.
- **Remove Perimeter Edges** — clears the exposed-perimeter marking from the currently selected edges.

Exposed-perimeter edges are drawn in red while a `Construction:FfactorGroundFloor` construction is active in the dialog, so you can visually confirm the assignment against your actual slab-edge condition before relying on it.

EP3's Validation dialog cross-checks this assignment: for each F-Factor construction, it compares the assigned perimeter length against the length it calculates from geometry and flags a mismatch greater than 10%. This is an advisory warning intended to inform the modeler that there may

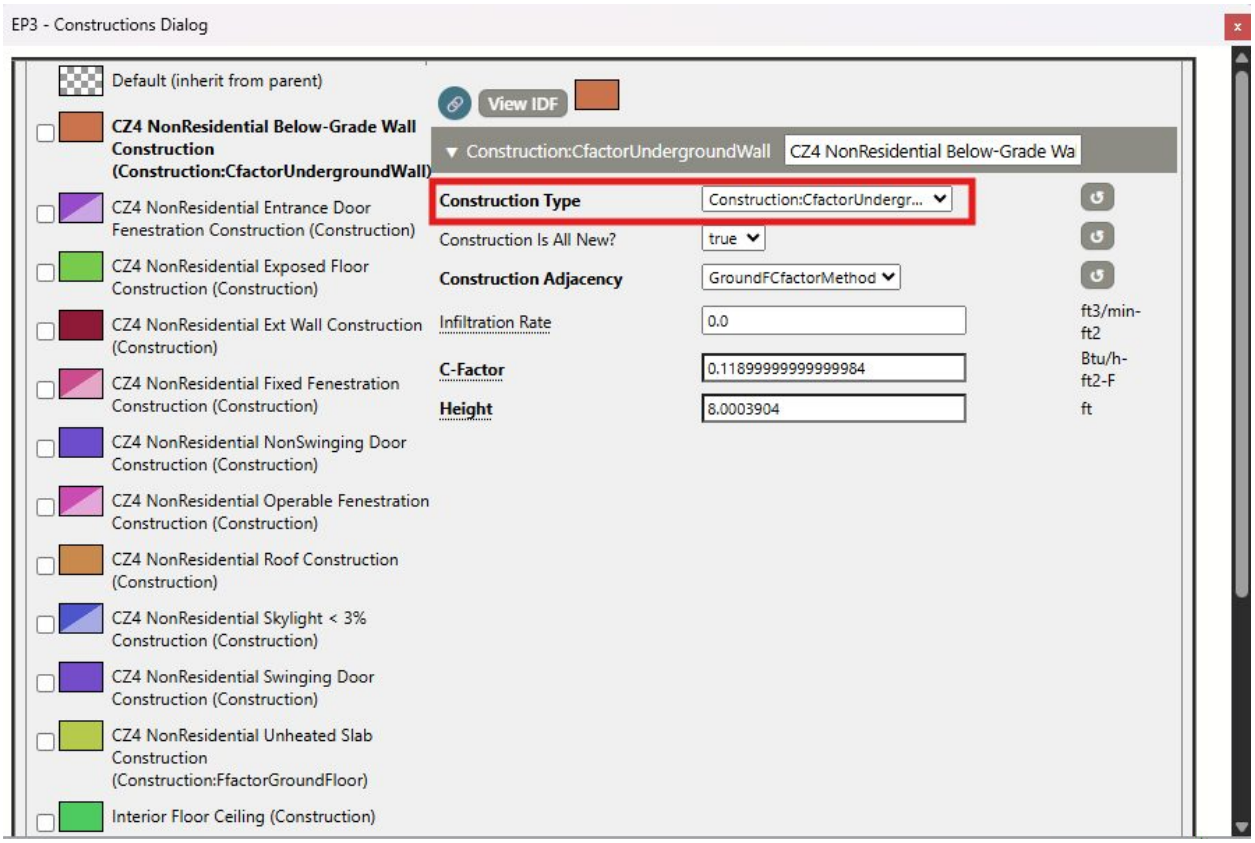


Figure 68: The Constructions dialog for a Construction:CfactorUndergroundWall, showing its C-Factor and Height fields.

be missig (or additional) perimeter edges in the model - it will not prevent the simulation from running or result in simulation errors.

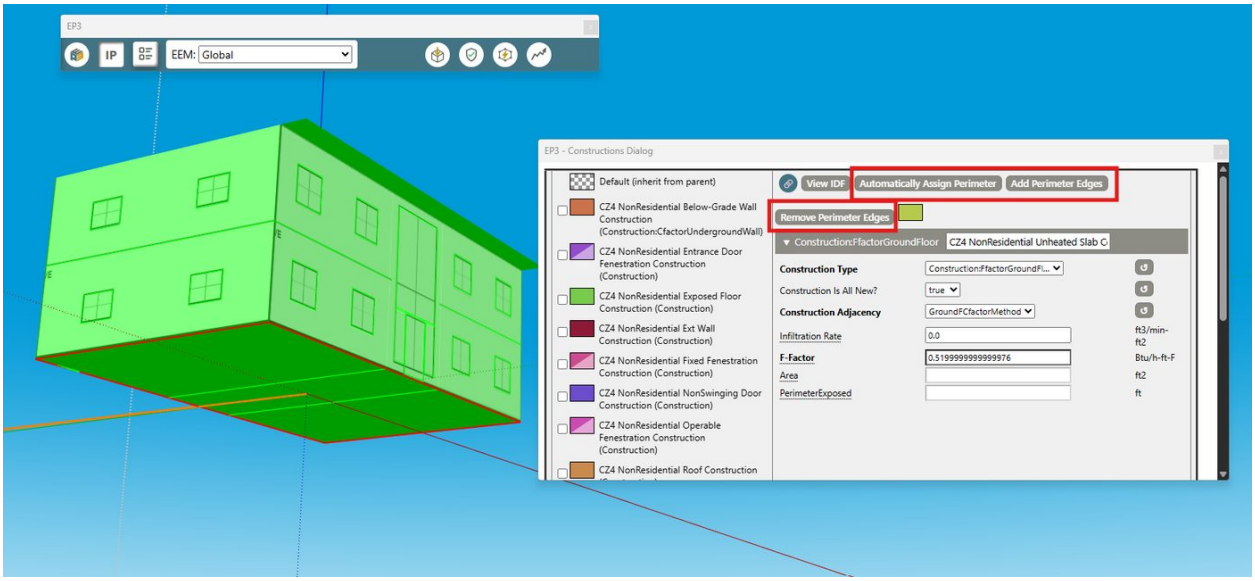


Figure 69: A Construction:FfactorGroundFloor active in the Constructions dialog, with the building's exposed-perimeter edges highlighted in red.

6.2 Boundary condition is locked to GroundFCfactorMethod

Selecting either F-Factor or C-Factor as a construction's type forces its Construction Adjacency to `GroundFCfactorMethod`, regardless of what it was set to before — this is the EnergyPlus-required boundary condition for both methods, and EP3 sets it automatically rather than leaving it to the modeler.

6.3 Relationship to ASHRAE 90.1 baseline generation

F-Factor and C-Factor constructions, and the "Roof Aged Test Data Available?" field EP3 adds to `Material`/`Material:NoMass` layers, are most consequential in the context of ASHRAE 90.1 Appendix G baseline generation, where EP3 assigns code-minimum ground-contact and roof performance by climate zone. That baseline-specific behavior — along with the related **Construction Is All New?** flag used to distinguish existing construction from new construction in an addition/alteration project — is covered in *(see "ASHRAE 90.1 Appendix G Baseline Generation").*

---

7. `Construction:AirBoundary`

`Construction:AirBoundary` models an adjacency between two zones with no thermal resistance — the surface exists for zoning/geometry purposes but doesn't restrict heat transfer the way a real wall would. Its **Air Exchange Method** field (`None` or `SimpleMixing`) optionally adds bulk air mixing across the boundary, driven by a **Simple Mixing Air Changes per Hour** value (using the smaller of the two zones' volumes as the basis) and an optional **Simple Mixing Schedule Name**. If an AirflowNetwork simulation is active in the model, this simple-mixing behavior is ignored in

favor of the network's own airflow calculation — see *(see "Infiltration, Natural Ventilation and Airflow Networks").*

---

8. Building a reusable construction/material template

Once you've built a set of constructions, you may want to save them as a template for re-use in the EP3 wizard. To create the re-usable template, select all of the constructions you wish to save in the Constructions dialog and click **Create Constructions Library for Selected**. EP3 gathers the selected constructions and everything they reference (materials, glazing layers, etc.), and saves the file in your configured library folder (set via the library-path prompt the first time you save one).

> **Simulate before you save the template.** As with any template creation in EP3, run a simulation using these constructions in an actual model *before* saving them as a library — not after. A construction that exports and simulates without error is confirmed usable; one saved straight from the Constructions dialog without that check can carry a problem (an invalid layer combination, a missing required field) that only surfaces the next time someone tries to use the template, in a different project.

To reuse a saved library, either select the constructions from the wizard when you create a new model, or open the Importer and go to its Materials import step — it lists your own saved libraries under **User Library** (one sub-entry per configured folder) alongside EP3's own built-in constructions under **EP3 Library**. Selecting an entry pastes its constructions into the current model.

8.1 Naming your constructions so EP3's default-picker finds them

(see "The EP3 Interface and Model Settings", §8.1) lets you set a default construction per surface category (exterior walls, roof, glass, etc.) for new geometry. When no default is set yet, EP3 tries to find a plausible one automatically by matching substrings in your constructions' names against each category — so naming your library's constructions with these substrings in mind means the picker finds them without you having to assign every default by hand after importing a library:

Category	Substrings EP3 looks for
Exterior walls	contains both "ext" and "wall"
Interior walls	contains both "int" and "wall"
Roof	contains "roof"
Glass	contains "glazing" or "fenestration" (and does **not** contain "door"); failing that, contains "glass" or "win"
Interior floor/ceiling	contains "int" and "floor", or "int" and "ceil"
Underground floor	contains "underground floor", "slab", "f factor", or "f-factor"
Underground wall	contains "below", "grade", and "wall", or contains "c factor"
Exposed floor	contains "exposed" and "floor"

The "does not contain 'door'" rule on the glass category is exact, not a general exclusion of anything door-related elsewhere: a construction named "Entrance Door Fenestration Construction" contains "fenestration" but is excluded from auto-detection as a default glass construction specifically because it also contains "door" — intentional, since an entrance door's glazing shouldn't silently become the model's general-purpose default glass construction.

EP3's own built-in library follows this convention throughout — for example, its ASHRAE 90.1 climate-zone libraries name constructions like "CZ4 NonResidential Ext Wall Construction," "CZ4 NonResidential Roof Construction," and "CZ4 NonResidential Below-Grade Wall Construction," each picking up the matching substrings for its category. Following the same pattern in your own libraries — `<something> Ext Wall Construction`, `<something> Roof Construction`, and so on — gets you the same automatic behavior after import.

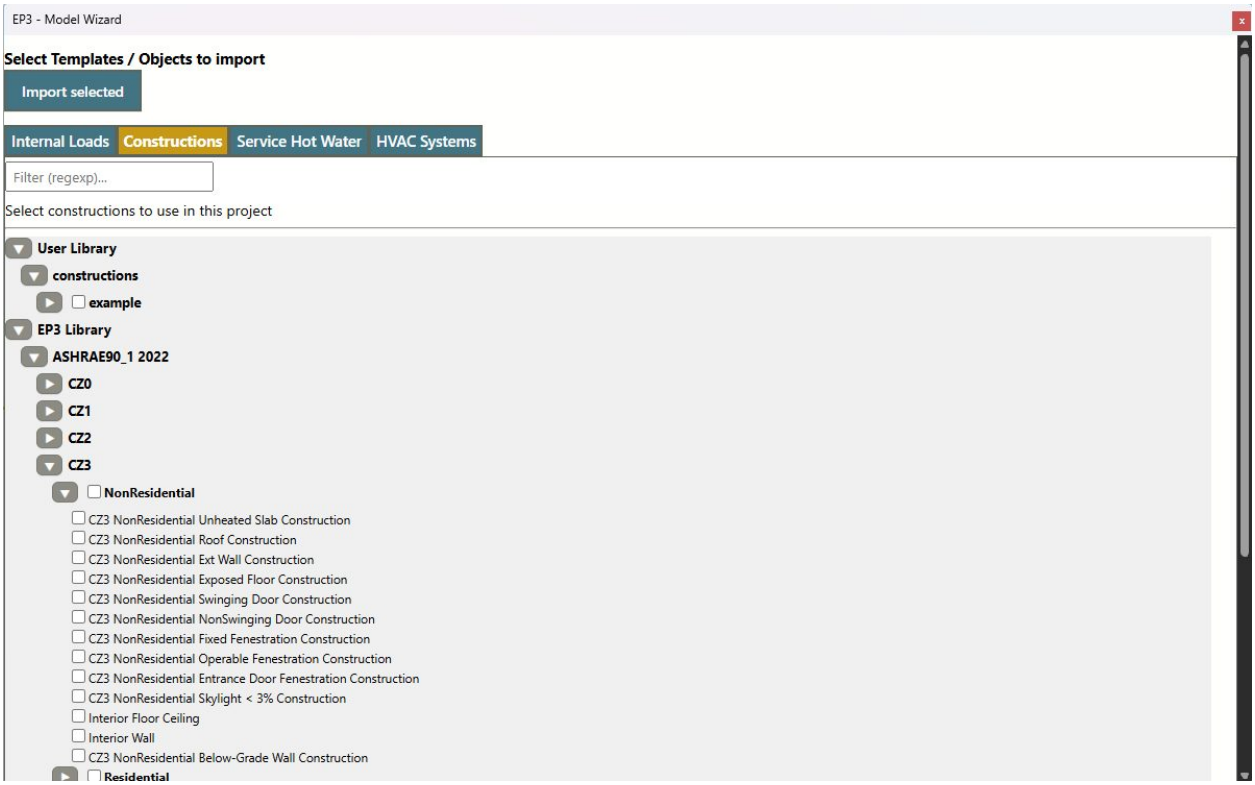


Figure 70: The Importer's Materials import step, showing the User Library / EP3 Library grouping.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target sections/pages exist (infiltration-natural-ventilation-and-airflow-networks is not yet drafted); confirm final document format (HTML vs. PDF vs. other).

8. Windows, Shading, and Daylighting

1. Overview

(see "Creating Building Geometry", §7) covers placing windows and doors in the model — Add Windows By Exposure, painting, and copying between floors. Once an opening exists, this chapter covers everything that shapes its behavior beyond that placement: standardizing its size and construction through an Opening Type, giving it fixed external shading (overhangs, fins, inset reveals), controlling movable shading and switchable glazing, adding a tubular daylighting device, placing the daylighting reference points that drive daylight-responsive lighting controls, and converting the model for a radiance-based study in SketchUp's Analysis Hub extension.

The glazing construction an opening's Construction Name field points at — layer buildup, `WindowMaterial:SimpleGlazingSystem`, and the rest — is covered in *(see "Constructions and Materials")*. The physical shading materials referenced from a `WindowShadingControl` (`WindowMaterial:Shade`, `WindowMaterial:Blind`, `WindowMaterial:Screen`) are covered directly in §4 below, since that's where they're used.

---

2. Standardizing Window and Door Properties (Opening Type)

An **Opening Type** is a reusable template a window or door references from its own **Opening Type** field: construction, frame and divider, and size/sill height. Assign the same Opening Type to every window that's supposed to match. Changing the Opening Type's fields updates every window referencing it at once.

2.1 Fields

Field	Purpose
Construction Name	The glazing (or door) construction painted onto every window using this type.
Frame and Divider Name	A `WindowProperty:FrameAndDivider` object (frame width, projection, conductance) — rectangular exterior windows and glass doors only; unused for triangular windows.
Calculation Method	`Distance` (default) or `Percent of Wall Size` — see §2.2.
Width / Height / Sill Height	Used when Calculation Method is `Distance`. Sill Height cannot be used for windows in roofs or on angled surfaces, and will be ignored for windows on such surfaces.
% of Wall Width / % of Wall Height / Sill Height as % of Wall Height	Used when Calculation Method is `Percent of Wall Size`; best reserved for rectangular walls. Sill Height as % of Wall Height defaults to **Autocalculate**, which centers the window vertically on the wall rather than measuring from the wall's bottom.

None of these fields are required — an Opening Type can leave any of them blank, in which case that particular property of the window is left alone.

2.2 Worked example: Percent of Wall Size

A wall face is 4 m wide and 3 m tall. An Opening Type set to `Percent of Wall Size` with % of Wall Width = 60 and % of Wall Height = 40, Sill Height as % of Wall Height left at Autocalculate, produces:

- Width = 4 m × 0.60 = **2.4 m**
- Height = 3 m × 0.40 = **1.2 m**
- Sill Height (autocalculated, centered) = (3 m − 1.2 m) ÷ 2 = **0.9 m**

Every window on a different wall referencing the same Opening Type is sized off *that* wall's own width and height — the percentages are shared, the resulting dimensions are not.

2.3 Opening Type overrides the window, not the other way around

Assigning (or editing) an Opening Type **overwrites** the window's own Width, Height, Sill Height, and Construction Name outright — it does not merely supply a fallback for whichever of those fields the window hasn't set. Change the Opening Type's Height field and every window referencing it is resized immediately, even a window whose height you'd previously set by hand.

This override isn't limited to editing the Opening Type's own fields — it re-fires on ordinary actions that have nothing to do with Opening Type at all. Say you stretch a window taller by hand with SketchUp's Stretch tool. That resize sticks visually until the next time the model regenerates that window — which happens far more often than "someone edits this Opening Type": regenerating walls or roofs for the building (*(see "Creating Building Geometry", §5–6)*) re-applies every affected window's Opening Type, and so does switching the model's active EEM. Either one snaps the hand-stretched window straight back to whatever its Opening Type specifies, silently undoing the manual resize.

This is the reverse of how inheritance works everywhere else in EP3: a SpaceType, ZoneType, or HVAC template supplies a *default* for the object that references it — the sub-object's own explicit value always overrides the template, not the other way around. Opening Type is the **one exception** in the entire application: the template overrides the instance. Don't assign an Opening Type to a window whose size or construction you need to hold fixed independent of other windows sharing that type, and don't rely on a hand-stretched, Opening-Type-linked window keeping its stretched size across a wall/roof regeneration or an EEM switch.

2.4 Custom (non-rectangular) windows don't use Opening Type

Opening Type only applies to the standard rectangular window/door component. A window built by painting a non-rectangular face (*(see "Creating Building Geometry", §7.2)*) doesn't expose an Opening Type field at all — its size and shape come directly from the painted geometry, and it's edited individually.

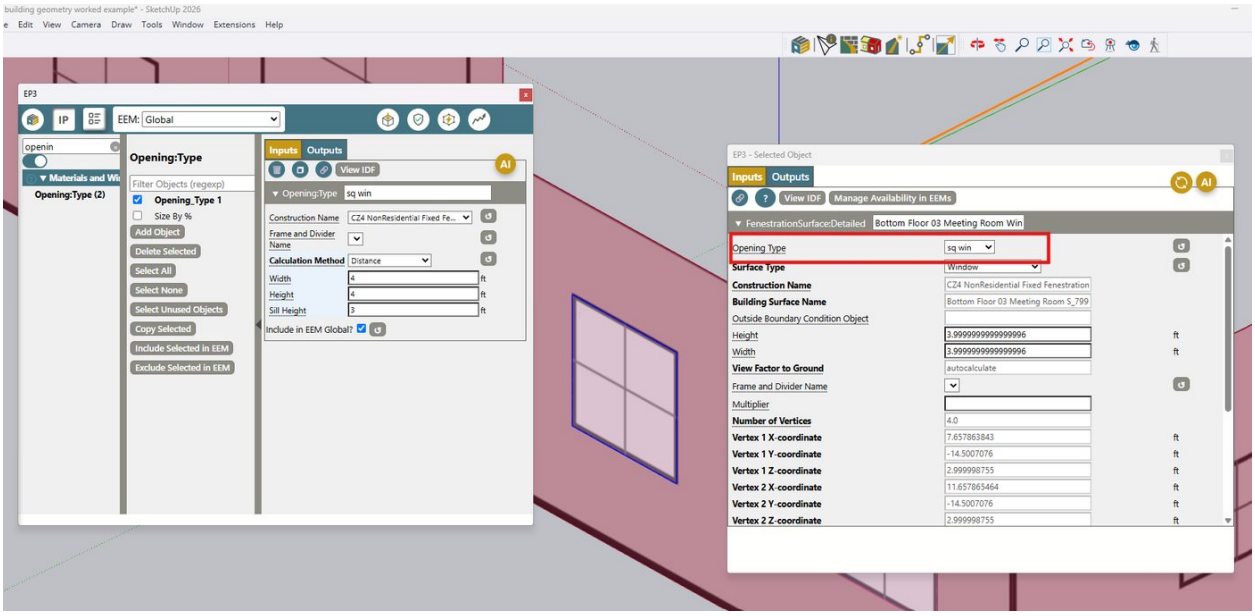


Figure 71: The Opening Type editor (Column 3) for an Opening Type named "sq win" — Calculation Method Distance, Width 4 ft, Height 4 ft, Sill Height 3 ft — alongside the referencing window's own Column 3, its Opening Type field (highlighted) pointing at "sq win" and its own Height/Width fields reading the same 4 ft the Opening Type specifies.

3. Fixed External Shading

Three shading geometries can be attached to any window or door, each toggled on from that opening's own properties and convered to EnergyPlus as `Shading:Building:Detailed`: an **overhang**, **fins** (left and / or right, independent of each other), and an **inset** (a recessed reveal). All three only exist — and only appear in the model — once their toggle is switched on.

3.1 Overhangs

Set **Use Shading Overhang?** to Yes to expose:

Field	Meaning
Height above Window or Door	Vertical gap between the top of the opening and the overhang. Default 0 — overhang sits directly at the head.
Tilt Angle from Window/Door	0–180°, default 90° (horizontal, projecting straight out).
Left extension from Window/Door Width / Right extension from Window/Door Width	How far the overhang extends past each side of the opening. Default 0 on each.
Depth	How far the overhang projects outward. Default 0.
Transmittance Schedule Name	Defaults to always-opaque if left blank.

Worked example — sizing an overhang to shade a window at solar noon on the summer solstice. At solar noon on the summer solstice, solar altitude at a given latitude L is approximately:

altitude ~ $90° - L + 23.45°$

For a building at 41.9°N (Chicago):

altitude ~ $90° - 41.9° + 23.45° =$ **71.55°**

Mount the overhang directly at the window head (Height above Window or Door = 0) and size Depth so its shadow just reaches the sill of a 1.5 m-tall window at that sun angle:

Depth = window height ÷ tan(altitude) = 1.5 m ÷ tan(71.55°) = 1.5 ÷ 2.988 ~ **0.50 m**

Checked in IP: 1.5 m = 4.92 ft, tan(71.55°) is unitless, so Depth = 4.92 ÷ 2.988 ~ 1.65 ft ~ 0.50 m — the two paths agree. At the winter solstice the sun sits much lower (altitude ~ $90° - L - 23.45°$ ~ 24.65° for the same latitude), so the same overhang casts a shadow far below the sill and admits full sun.

Source: solar altitude formula from ASHRAE Fundamentals' solar position chapter; Height above Window or Door and Depth are the same two quantities EnergyPlus's sunlit-fraction calculation uses for ***Shading:Overhang*** *— see* Shading:Overhang *for the full geometric derivation.*

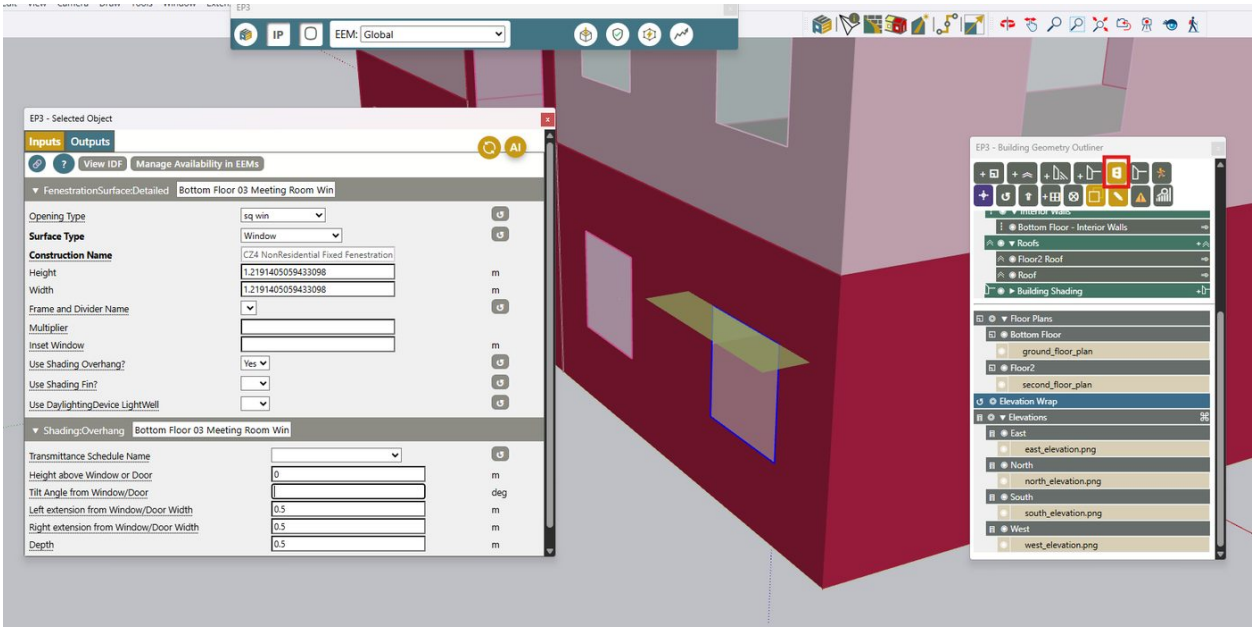


Figure 72: A window's own dialog with Use Shading Overhang? set to Yes and its Shading:Overhang sub-object's fields below it, the resulting overhang geometry visible in the 3D view, and the Building Geometry Outliner's "Show window insets and overhangs" toggle (highlighted) switched on — without it, neither the overhang nor an inset (§3.3) is visible in the viewport even though both still export to EnergyPlus.

The Building Geometry Outliner's "Show window insets and overhangs" toggle controls whether these two shading geometries are visible in the 3D view at all. Turn it on before checking a model's overhangs or insets by eye — with it off, the geometry is still there and still exports, it's just hidden.

3.2 Fins

Set **Use Shading Fin?** to Yes to expose independent left and right fin geometry: Extension from Window/Door, Distance Above Top of Window, Distance Below Bottom of Window, Tilt Angle from Window/Door (default 90°), and Depth for each side, plus a Left and Right Transmittance Schedule Name. All extension/distance/depth defaults are 0, so a fin contributes no geometry on whichever side is left at defaults.

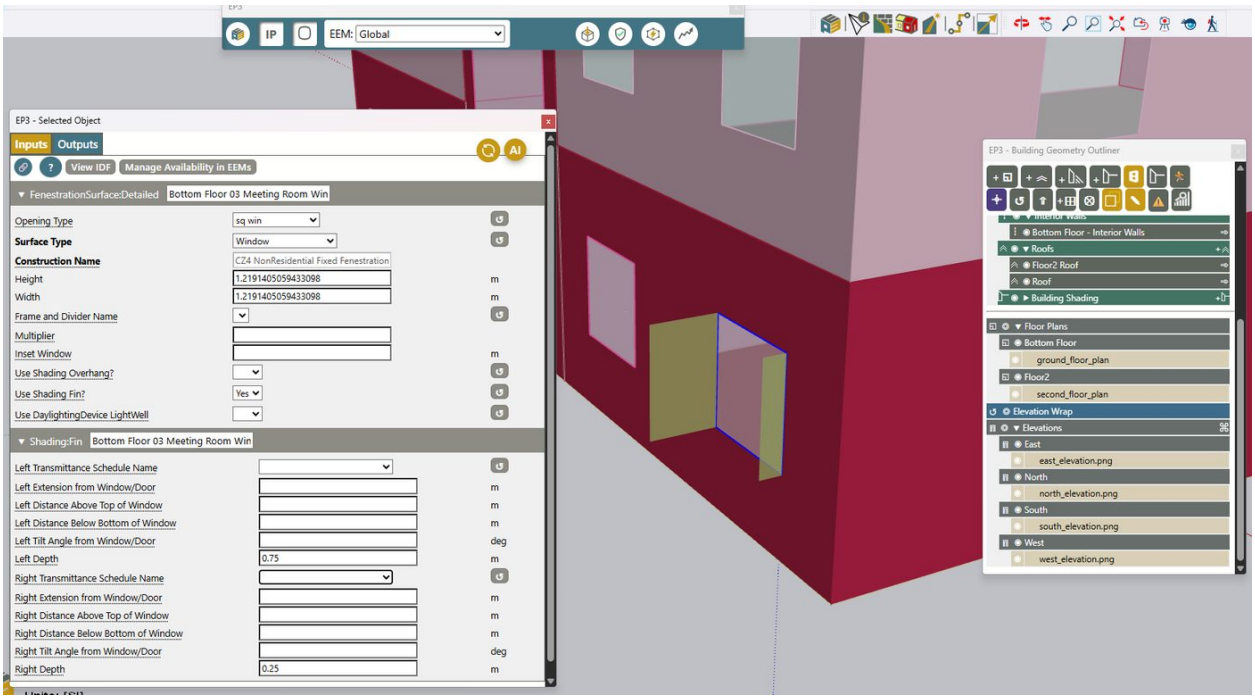


Figure 73: A window's Shading:Fin sub-object with Left Depth set to 0.75 m and Right Depth set to 0.25 m — independent of each other — producing a visibly deeper fin on the left than the right in the 3D view.

3.3 Inset windows

The **Inset Window** field lives on the window itself, not on a sub-object — enter a depth and EP3 builds the four reveal surfaces (top, bottom, left, right) connecting the opening to a plane recessed by that depth along the wall's outward normal. Its sub-object exposes only a **Transmittance Schedule Name**, same default-to-opaque behavior as the other two.

---

4. Movable Shading and Switchable Glazing

`WindowShadingControl` governs shading that responds to a schedule or a triggering condition — blinds, shades, screens, or switchable (thermochromatic) glazing — rather than fixed geometry. It's created like any other object, through the Main Dialog's three columns: select `WindowShadingControl` in Column 1, **Add Object** in Column 2, edit its fields in Column 3 (*(see "The EP3 Interface and Model Settings", §2.1–§2.4)*).

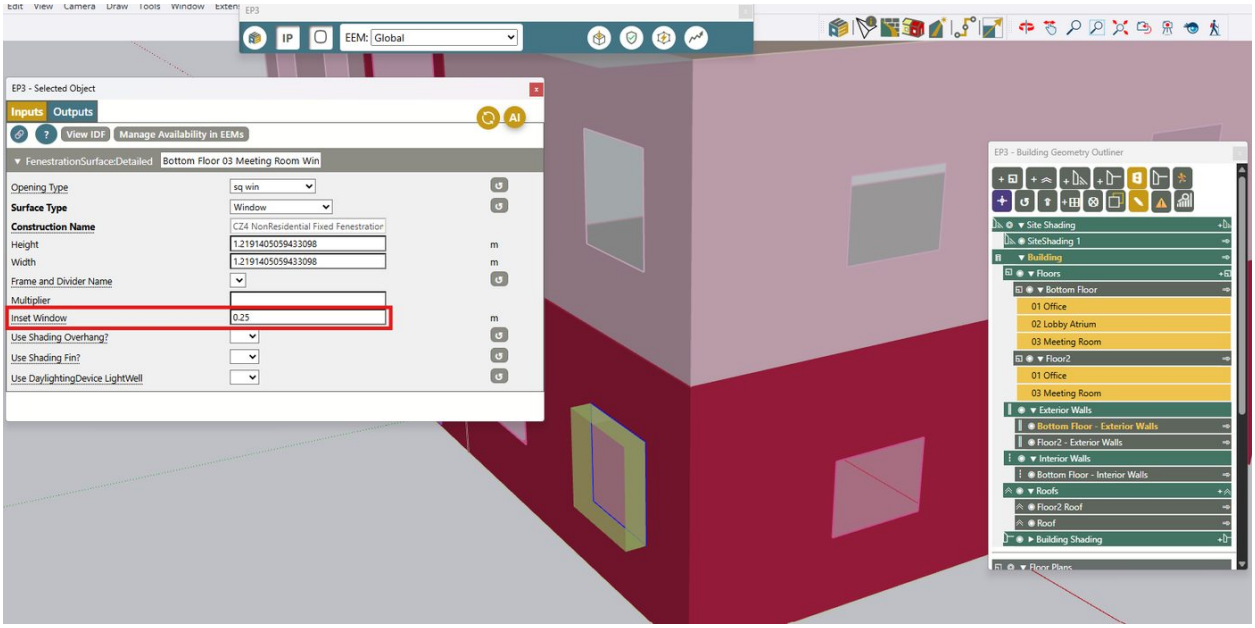


Figure 74: A window's own Inset Window field (highlighted) set to 0.25 m, and the resulting recessed reveal connecting the glazing to the wall's outer face in the 3D view.

4.1 What it applies to

A `WindowShadingControl` is scoped to a Zone (EP3 also accepts a ZoneList in this field). Its **Fenestration Surfaces** list lets you name specific windows within that zone/space — but if you leave that list empty, EP3 applies the control to **every** fenestration surface in the space instead of exporting an invalid, windowless control.

4.2 Shading Type: what's being controlled

Shading Type	Referenced material
InteriorShade, ExteriorShade, BetweenGlassShade	`WindowMaterial:Shade`
InteriorBlind, ExteriorBlind, BetweenGlassBlind	`WindowMaterial:Blind`
ExteriorScreen	`WindowMaterial:Screen`
SwitchableGlazing	A second, switched-state glazing construction (thermochromatic or electrochromic glass)

Point **Construction with Shading Name** at a full glazing construction that already incorporates the shade/blind/screen/switched layer, or point **Shading Device Material Name** directly at the shading material and let EP3 pair it with the window's existing construction — if both are entered, Construction with Shading Name takes precedence. SwitchableGlazing requires Construction with Shading Name (there's no separate "material" for a switched state — it's a whole alternate construction, built the same way as any other glazing construction — *(see "Constructions and Materials")*).

`WindowMaterial:Shade`, `WindowMaterial:Blind`, and `WindowMaterial:Screen` are created the same way as any other EP3 object — Column 1/2/3 of the Main Dialog — and are documented here rather than in the Constructions chapter, since a shading device only ever gets referenced from a `WindowShadingControl`, not painted onto anything.

`WindowMaterial:Shade` A shade is treated as a single uniform layer: **Solar Transmittance**, **Solar Reflectance**, **Visible Transmittance**, **Visible Reflectance**, **Infrared Hemispherical Emissivity**, and **Infrared Transmittance** are all required, with no default — they come from manufacturer data or physical measurement, same as **Thickness** and **Conductivity**. **Shade to Glass Distance** defaults to 0.05 m, and **Top/Bottom/Left-Side/Right-Side Opening Multiplier** (default 0.5 each) set how much air can circulate around the shade's edges against the glass — distinct from **Airflow Permeability** (default 0), which sets how much air passes straight through the fabric itself.

`WindowMaterial:Blind` A blind adds slat geometry on top of the same kind of optical properties: **Slat Orientation** (Horizontal, the default, or Vertical), **Slat Width** and **Slat Separation** (both required), **Slat Thickness**, and **Slat Conductivity** (default 221 W/m-K, aluminum). Front and back **beam** and **diffuse** solar and visible transmittance/reflectance are all required and independent of each other — a slat's surface finish typically scatters more than it transmits directly, so these aren't expected to match the simpler single-value fields a `WindowMaterial:Shade` uses. **Slat Angle** (default 45°) is the field that interacts directly with §4.4's **Type of Slat Angle Control for Blinds**: it's the fixed angle under `FixedSlatAngle`, the resting angle when beam solar isn't currently being blocked under `BlockBeamSolar`, and unused under `ScheduledSlatAngle`; **Minimum Slat Angle**/**Maximum Slat Angle** (0°/180° defaults) then bound whatever angle EnergyPlus is allowed to choose under `ScheduledSlatAngle` or `BlockBeamSolar`. **Blind to Glass Distance** defaults to 0.05 m; **Blind Top/Bottom/Left-Side/Right-Side Opening Multiplier** default to 0.5/0/0.5/0.5 — asymmetric, since a blind's bottom edge is normally down against the sill rather than open to airflow the way its top and sides are.

`WindowMaterial:Screen` **Reflected Beam Transmittance Accounting Method** (default `ModelAsDiffuse`; `DoNotModel` or `ModelAsDirectBeam` are the alternatives) governs how EnergyPlus treats beam solar that reflects off the screen's woven material rather than passing through it. **Diffuse Solar Reflectance** and **Diffuse Visible Reflectance** are required; **Thermal Hemispherical Emissivity** and **Conductivity** default to 0.9 and 221 W/m-K (aluminum), matching a blind's defaults. **Screen Material Spacing** and **Screen Material Diameter** are both required — together they set the weave's actual openness rather than there being a separate "openness factor" field to enter directly. **Screen to Glass Distance** defaults to 0.025 m, and the Top/Bottom/Left-Side/Right-Side Opening Multipliers all default to **0** here — unlike a shade's or blind's 0.5 defaults — since a screen is normally mounted flush against the frame with no edge gap for air to circulate through.

4.3 Shading Control Type: what triggers it

Purpose	Representative control types
Always on/off, or schedule-only	`AlwaysOn`, `AlwaysOff`, `OnIfScheduleAllows`

Purpose	Representative control types
Reduce cooling load from solar gain	`OnIfHighSolarOnWindow`, `OnIfHighHorizontalSolar`, `OnIfHighOutdoorAirTemperature`, `OnIfHighZoneCooling`, and their combined-condition variants
Reduce heating load (movable insulation)	`OnNightIfHeatingAndOffDay`, `OnNightIfLowOutdoorTempAndOffDay`, `OnNightIfLowInsideTempAndOffDay`
Reduce both heating and cooling load	`OnNightIfHeatingAndOnDayIfCooling`, `OnNightAndOnDayIfCoolingAndHighSolarOnWindow`
Glare and daylight-linked	`OnIfHighGlare`, `MeetDaylightIlluminanceSetpoint`, `OnIfHighSolarOrHighLuminanceTillSunset` (and its TillMidnight/TillNextMorning variants)

Most setpoint-driven types need **Setpoint** (W/m² for solar, W for cooling/heating, °C for temperature); the two-condition types also need **Setpoint 2**. `OnIfScheduleAllows` requires **Shading Control Is Scheduled** = Yes and a **Schedule Name** — a schedule value of 0 forces shading off regardless of what the control type would otherwise decide; a value of 1 lets the control type's own test decide. `OnIfHighGlare` requires **Glare Control Is Active** = Yes. WindowShadingControl documents every control type's exact trigger condition.

4.4 Slat angle control for blinds

For any Blind shading type, **Type of Slat Angle Control for Blinds** chooses between `FixedSlatAngle` (default), `ScheduledSlatAngle` (paired with a **Slat Angle Schedule Name**, values in degrees, 0–180), or `BlockBeamSolar` (EnergyPlus calculates the angle needed to block direct beam solar at each timestep).

4.5 Daylighting Control Object Name is automatic

This field isn't user-editable in EP3 — it's assigned automatically whenever the space it applies to has daylighting controls (§6). You don't need to wire shading and daylighting controls together by hand.

4.6 Controlling more than one window in a space

If a space needs more than one `WindowShadingControl` (different shading on different windows, or windows that deploy in a particular order), **Shading Control Sequence Number** sets that order — the first control a space needs is 1, the next 2, and so on. **Multiple Surface Control Type** then decides whether the windows listed in one control deploy together (`Group`, the setting under which glare control deploys the whole group at once) or individually in sequence (`Sequential`, the default).

5. Tubular Daylighting Devices

A window becomes a tubular daylighting device (TDD) by setting its **Surface Type** to `TUBULARDaylightDome` — there's no separate object to place. Once set, EP3 automatically creates the matching dome, tube (`DaylightingDevice:Tubular`), and diffuser objects and adds a **Tubular Daylight Content** section to that window's own dialog.

5.1 Fields

Dome Name and **Diffuser Name** are generated automatically from the window's own name. **Construction Name** is the tube shaft's own construction — a highly reflective pipe material, not a glazing — while the window's own Construction Name field (unaffected by this section) still governs the dome's exterior glazing, and the separately-added **Diffuser Construction Name** governs the interior diffuser's glazing. **Diameter** is auto-calculated from the dome opening's glazed area the first time the device is created — leave it as-is unless you need to override it.

Total Length and **Transition Zone Name/Transition Zone Length** are not auto-calculated and must be entered by hand: Total Length is the tube's full physical length from dome to diffuser, and if that run passes through one or more unconditioned zones in between (an attic or plenum, most commonly) rather than going straight from roof to ceiling in a single conditioned zone, add a Transition Zone Name/Transition Zone Length pair for each such zone the tube passes through — the exterior-exposed length EnergyPlus uses in its heat-loss calculation is Total Length minus the sum of whatever transition-zone lengths are entered. **Effective Thermal Resistance** (default 0.28 $m^2 \cdot K/W$) is the R-value between the dome and the diffuser.

Worked example — recovering Diameter from dome area. A circular dome opening 15 in (0.381 m) in diameter has a glazed area of pi × 7.5^2 ~ 176.7 in^2. EP3 recovers the diameter from that area as:

radius = √(area ÷ pi) = √(176.7 ÷ pi) = √56.25 = 7.5 in diameter = 2 × 7.5 in × 0.0254 m/in ~ **0.381 m**

— matching the physical dome exactly, because the opening is a true circle. If the glazed opening isn't circular (a square skylight used as a TDD dome, for instance), this calculation still runs and produces an *equivalent-area* diameter rather than the shape's actual diameter — worth checking manually in that case, since EnergyPlus's tubular-daylighting-device calculation assumes a circular cross-section.

---

6. Daylighting Reference Points and Controls

A **daylighting reference point** is a component you place and glue to the floor at the location where you want illuminance evaluated — right-click in the space and choose **"EP3 - Add Daylight Sensor"**, then position and glue it like any other floor-glued component. Its Sensor Height, Fraction of Lights Controlled by Reference Point, and Illuminance Setpoint fields are edited from Column 3 like any other object.

Once a space contains one or more daylighting reference points, a **Daylighting** section becomes visible and editable directly on that Space's own dialog (*(see "Spaces and Zones")*) — there's no separate `Daylighting:Controls` object to create or manage by hand.

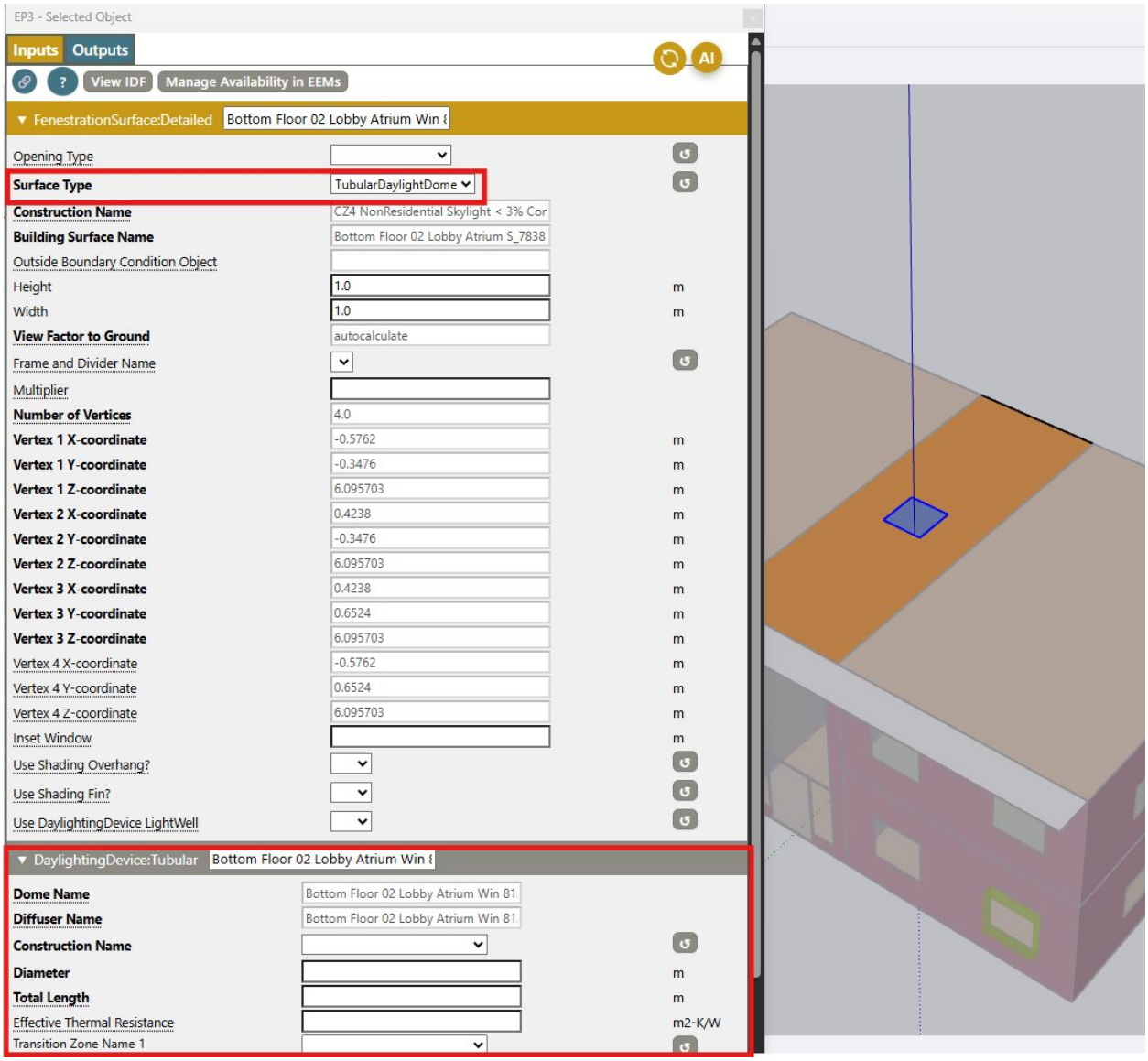


Figure 75: A window with Surface Type set to TubularDaylightDome (highlighted), and the added DaylightingDevice:Tubular section below it showing Dome Name, Diffuser Name, Construction Name, Diameter, Total Length, Effective Thermal Resistance, and Transition Zone Name — with the dome visible in the 3D view as a small diamond shape on the roof.



Figure 76: A Daylighting:ReferencePoint's own Column 3 (Zone or Space Name, X/Y/Z-Coordinate, Sensor Height, Fraction of Lights Controlled, Illuminance Setpoint) alongside its sensor icon placed in the floor plan, inside the Lobby Atrium space.

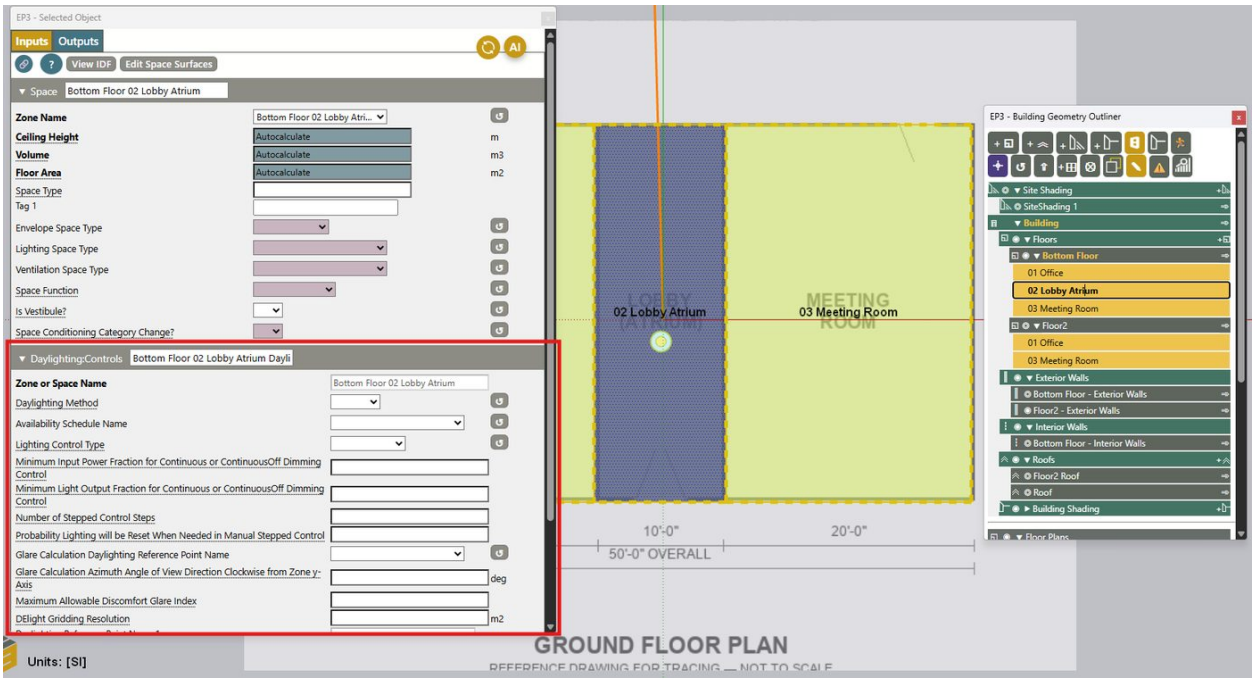


Figure 77: The Lobby Atrium space's own Column 3, with the added Daylighting:Controls section (highlighted) below the Space's ordinary fields — Daylighting Method, Availability Schedule Name, Lighting Control Type, and the rest — visible only because this space now has a daylighting reference point.

7. SketchUp Analysis Hub Integration

Analysis Hub is SketchUp's extension for radiance-based daylighting studies and renderings — a separate program from EnergyPlus, installed independently through the Extension Warehouse. EP3 doesn't run Analysis Hub itself; it prepares a copy of the model's geometry that Analysis Hub can read, then hands off to Analysis Hub directly. The command that does this — **Convert Geometry to SketchUp Analysis Hub Compatible Geometry** — lives on the EP3 toolbar rather than opening an EP3 window (*(see "The EP3 Interface and Model Settings", §2.1–§2.4)*).



Figure 78: Convert Geometry to Sketchup Analysis Hub Compatible Geometry icon

7.1 What the conversion produces

Running the command builds a self-contained copy of the current EEM's geometry on a separate layer, leaving the live model untouched:

- Window and door component instances are exploded into plain glazing faces, since Analysis Hub reads raw geometry rather than EP3's parametric window objects.
- Every face is recolored to a neutral grey scaled to the visible reflectance of the construction layer it represents — taken from that construction layer's Visible Absorptance field, not the material's SketchUp display color or texture — so a radiance-based study run against the copy uses the same optical properties as the EnergyPlus model. A construction layer with no Visible Absorptance entered falls back to a default reflectance.
- Each EP3 space becomes an IFC-classified space volume carrying that space's name, so Analysis Hub results can be tied back to the same space names used throughout EP3.
- The building's exterior massing is included as shading geometry, so self-shading from the building's form carries into the study.

Because this is a static copy, edits made to the live model after generating it — moving a wall, resizing a window — aren't reflected until the command is run again.

The Convert Geometry to SketchUp Analysis Hub Compatible Geometry toolbar icon, and the resulting geometry copy — exploded window glazing and grey IFC space volumes — immediately before Analysis Hub's dialog opens.

Figure 79: The Convert Geometry to SketchUp Analysis Hub Compatible Geometry toolbar icon, and the resulting geometry copy — exploded window glazing and grey IFC space volumes — immediately before Analysis Hub's dialog opens.

7.2 Running the conversion

The conversion is scoped to whichever EEM is active when the command runs, and the resulting geometry is tagged to that EEM. Running the command again while that same EEM is active finds the previously generated geometry and asks whether to reuse it or generate a new copy, instead of regenerating every time.

Once the copy is built, EP3 opens Analysis Hub's dialog directly, handing off to Analysis Hub for the rest of the workflow. If Analysis Hub isn't installed, EP3 prompts to install it from the Extension Warehouse; if an older version is installed that doesn't support this handoff, EP3 prompts to update it through the Extension Manager instead.

7.3 Relationship to EnergyPlus's daylighting calculation

This conversion is independent of the Daylighting:ReferencePoint and Daylighting:Controls objects covered in §6. Those drive EnergyPlus's illuminance calculation and the lighting-control energy savings that show up in simulation results; Analysis Hub's radiance-based study runs separately in SketchUp and doesn't feed back into the EnergyPlus simulation. Use §6 for the daylighting behavior that feeds the EnergyPlus simulation, and Analysis Hub for renderings or radiance-based metrics outside it.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

9. Internal Loads, Space Types, and Zone Types

1. Overview

Almost every project gets its people, lighting, and equipment loads — along with the thermostat and zone-sizing defaults that go with them — from a `SpaceType`/`ZoneType` template pair. This chapter calls the mechanism that brings such a pair into a model the **Wizard/Library path**, and treats it as the default way of working: the Model Wizard's Internal Loads tab *(see "The Model Wizard", §3)* is where most projects pick their templates the first time, and reopening the Wizard later in its **Library** mode is how a template gets added at any later point in the project. It's also the *only* way a `SpaceType`/`ZoneType` template gets reused, whether it comes from EP3's own built-in library or one created from another project (§7): EP3's general EnergyPlus file import *(see "Import, Export, and Interoperability")* brings in plain EnergyPlus objects, and doesn't reconstruct a linked template pair the way the Wizard/Library path does.

A model can also carry `People`, `Lights`, `ElectricEquipment`, `OtherEquipment`, infiltration, `Sizing:Zone`, `WaterUse:Equipment`, and `Thermostat` objects with no `SpaceType` or `ZoneType` behind them at all — tied directly to a `Zone`, a `Space`, a `ZoneList`, or a `SpaceList`. This is EnergyPlus's native approach: point each object at the `Zone`/`Space`/`ZoneList`/`SpaceList` it applies to. The modeler is responsible for keeping each list's membership current, and mutually exclusive of every other list, by hand. It's how every one of these objects looks in a model built outside EP3 and then imported — nothing constructs a matching `SpaceType`/`ZoneType` pair for you on import, so the loads simply attach to whatever the source file pointed them at (§6 covers reading and working with loads in this form). The Wizard/Library path exists to remove that hand-maintenance: a `SpaceType`/`ZoneType`'s own `SpaceList`/`ZoneList` keeps its membership current automatically (§5), and every field that would otherwise need setting object-by-object across a whole occupancy or use type is set once, on the template.

A loose object isn't exclusive to an untemplated model, though. Adding one directly to a single Space or Zone that already carries a `SpaceType`/`ZoneType` is the normal way to model something the template doesn't represent at all — special display lighting in one particular space, say, alongside the `Lights` its `SpaceType` already provides. A `SpaceType`/`ZoneType`'s own fields can also be overridden for a single assigned Space or Zone (§5.3): enter a value directly on that one field, for that one Space or Zone, and it takes precedence over the template's value from then on, while every other field — and every other Space or Zone sharing the template — is unaffected. This is the tool to reach for when the template is right for most spaces or most fields, but not quite all of them: for example, an occupant density that varies room to room within the same space type, or a lighting power density taken room-by-room from a lighting design rather than the type's own default. Both stay on the `SpaceType` or `ZoneTypes`'s own `People`/`Lights`/etc fields, just overridden per space or zone, rather than requiring a separate loose object.

---

2. Worked example: creating a linked SpaceType and ZoneType

The "Create New Space Type and Loads" dialog builds a matched pair from a single set of prompts — lighting, equipment, occupancy, infiltration, and thermostat inputs together — rather than assembling five separate objects by hand. It's reached from the **Create New Space Type and Loads** button on the Column 2 list for `SpaceType` or `ZoneType`.

Suppose this is being filled in for a typical open-plan office space, named to match: **New Space**

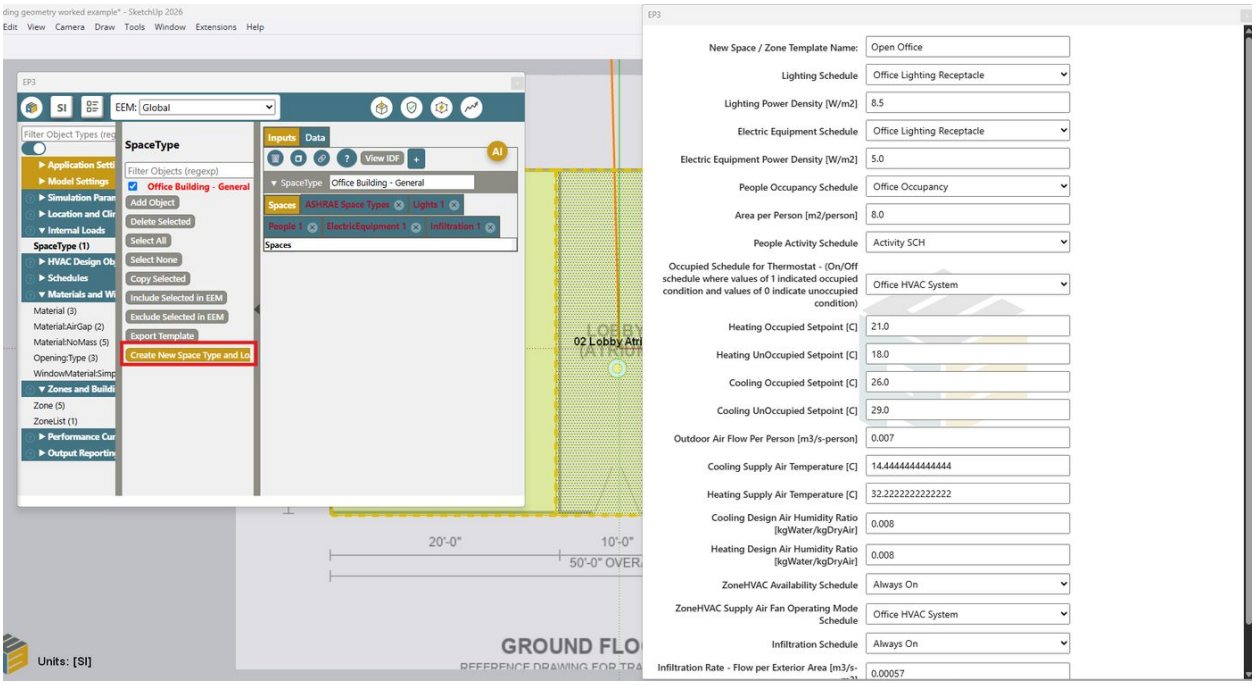


Figure 80: The "Create New Space Type and Loads" dialog filled in for the Open Office example below, with the triggering **Create New Space Type and Loads** button boxed in red on the SpaceType Column 2 list behind it.

/ Zone Template Name = `Open Office`, **Lighting Power Density** = 8.5 W/m², **Electric Equipment Power Density** = 5.0 W/m², **Area per Person** = 8.0 m²/person, **Infiltration Rate** = 0.00057 m³/s-m², **Occupied/Un-Occupied Heating Setpoint** = 21°C/18°C, **Occupied/Un-Occupied Cooling Setpoint** = 26°C/29°C, and **Outdoor Air Flow Per Person** = 0.007 m³/s-person, with schedules picked to match: Office Lighting Receptacle for both the lighting and equipment schedules, Office Occupancy and Activity SCH for the people schedules, Office HVAC System for the thermostat's occupied/unoccupied schedule and the ZoneHVAC supply air fan operating mode schedule, and Always On for the ZoneHVAC availability and infiltration schedules. Confirming creates, in one step:

- A `SpaceType` named **Open Office**, holding:
  - `Lights`, sized by **Design Level Calculation Method** = `Watts/Area`, at 8.5 W/m² on the **Lighting Schedule** entered
  - `ElectricEquipment`, sized the same way at 5.0 W/m² on the **Electric Equipment Schedule** entered
  - `People`, sized by **Number of People Calculation Method** = `Area/Person`, at 8.0 m²/person, using the **People Occupancy Schedule** and **People Activity Schedule** entered
  - `ZoneInfiltration:DesignFlowRate`, sized by **Design Flow Rate Calculation Method** = `Flow/ExteriorArea`, at 0.00057 m³/s-m² on the **Infiltration Schedule** entered
- A `ZoneType` named **Open Office**, holding:
  - A `SimpleThermostat` (§4.1), with occupied/unoccupied heating setpoints of 21°C/18°C and occupied/unoccupied cooling setpoints of 26°C/29°C

- A `Sizing:Zone`, with its supply-air-temperature input method set to `SupplyAirTemperature` from the design supply air temperature and humidity ratio fields
- A `DesignSpecification:OutdoorAir`, sized `Flow/Person` at 0.007 m^3/s-person, wired into the `Sizing:Zone` above
- The **ZoneHVAC Availability Schedule Name** and **Zone Supply Air Fan Operating Mode Schedule** cascading defaults (§4.2), if entered

The two objects are name-paired the same way a library import pairs them, ready to assign to Spaces and Zones (§5).

Every field in this dialog is a shortcut into fields covered in full in §3 and §4 below — nothing it sets is exclusive to this dialog, and every value it enters can be edited afterward on the `SpaceType`/`ZoneType` themselves like any other template.

---

3. The SpaceType object

A `SpaceType` is a container: its own fields are limited to a handful of ASHRAE 90.1 classification values (§3.6), and the actual loads live in sub-objects added to it — `Lights`, `People`, `ElectricEquipment`, `OtherEquipment`, and `ZoneInfiltration:DesignFlowRate`. Column 3 shows one tab per sub-object present *(see "The EP3 Interface and Model Settings")*; a `SpaceType` doesn't need to carry all five — add or remove a sub-object with the tab row's **+** and **X** controls. Assignment to Spaces happens at the `SpaceType` level (§5), not per sub-object.

3.1 People

Field	Purpose
Number of People Schedule Name	Fraction (0.0–1.0) applied to the calculated maximum occupant count, by time.
Number of People Calculation Method	`People` (enter a headcount directly), `People/Area` (headcount ÷ floor area), or `Area/Person` (floor area ÷ headcount) — whichever method is chosen determines which of the next three fields is used.
Number of People / People per Floor Area / Floor Area per Person	Only one is used, per the calculation method above.
Activity Level Schedule Name	Metabolic rate schedule (W/person) driving the sensible/latent split.
Fraction Radiant	Radiant fraction of sensible heat from people. Defaults to 0.30.
Sensible Heat Fraction	Overrides the program-calculated sensible/latent split if entered; defaults to `Autocalculate`.
Carbon Dioxide Generation Rate	Defaults from ASHRAE 62.1's 0.0084 cfm/met/person figure for the general adult population.

Field	Purpose
Enable ASHRAE 55 Comfort Warnings, Mean Radiant Temperature Calculation Type, and the thermal-comfort-model fields	Only relevant when running EnergyPlus's thermal comfort reporting — People covers these in full.

3.2 Lights

Field	Purpose
Schedule Name	Fraction (0.0–1.0) applied to the design level, by time.
Design Level Calculation Method	`LightingLevel` (enter watts directly), `Watts/Area` (× floor area), or `Watts/Person` (× occupant count).
Lighting Level / Watts per Floor Area / Watts per Person	Only one is used, per the calculation method above.
Return Air Fraction	Fraction of heat picked up by return air, used only if return-air-fraction coefficients are also specified. Defaults to 0.0.
Fraction Radiant / Fraction Visible	Split of the remaining (non-return-air) heat between radiant and short-wavelength (visible) gain. Both default to 0.0.
Fraction Replaceable	For `Daylighting:Controls` compatibility, must be 0 or 1 — 1 means the fixture can be dimmed. Defaults to 1.0.
End-Use Subcategory	Free text, groups this object's energy under a named row in the ABUPS End Uses by Subcategory table. Defaults to "General"; the quick-create dialog (§2) sets it to the template's own name.

3.3 ElectricEquipment

Field	Purpose
Schedule Name	Fraction (0.0–1.0) applied to the design level, by time.
Design Level Calculation Method	`EquipmentLevel` (enter watts directly), `Watts/Area`, or `Watts/Person`.
Design Level / Watts per Floor Area / Watts per Person	Only one is used, per the calculation method above.
Fraction Latent / Fraction Radiant / Fraction Lost	Split of the design level's heat between latent, radiant, and lost (removed from the space entirely — e.g. exhausted). All default to 0.0; whatever fraction remains after these three is convective.

Field	Purpose
End-Use Subcategory	Same purpose as on `Lights` above.

3.4 OtherEquipment

Structurally identical to `ElectricEquipment`, with one addition: a **Fuel Type** field (`Electricity`, `NaturalGas`, `DistrictHeatingWater`, `None`, and others) that lets `OtherEquipment` represent a load billed to a different meter than the electric one — process loads or unmetered/estimated gains are common uses. Entering a negative value in **Design Level** (or the per-area/per-person equivalent) models a heat *loss* rather than a gain. `GasEquipment`, `HotWaterEquipment`, and `SteamEquipment` are fuel-specific equivalents that exist as EnergyPlus objects but aren't managed by the `SpaceType` template mechanism — add one directly to a Space or Zone if needed (§6), the same way it would be entered in raw EnergyPlus. OtherEquipment

3.5 Infiltration (ZoneInfiltration:DesignFlowRate)

Field	Purpose
Schedule Name	Fraction (0.0–1.0) applied to the design flow rate, by time. Blank defaults to always 1.0.
Design Flow Rate Calculation Method	`Flow/Zone`, `Flow/Area` (× zone floor area), `Flow/ExteriorArea`, `Flow/ExteriorWallArea`, or `AirChanges/Hour` (× zone volume).
Design Flow Rate / Flow Rate per Floor Area / Flow Rate per Exterior Surface Area / Air Changes per Hour	Only one is used, per the calculation method above.
Constant Term Coefficient / Temperature Term Coefficient / Velocity Term Coefficient / Velocity Squared Term Coefficient	The A/B/C/D coefficients EnergyPlus applies to the design flow rate as a function of indoor-outdoor temperature difference and wind speed. Constant Term Coefficient defaults to 1.0 (no adjustment); the other three default to 0.0.

This is the object a `SpaceType`'s infiltration rate is stored on, but choosing a rate and understanding how it interacts with mechanical ventilation and `AirflowNetwork` is covered in *(see "Infiltration, Natural Ventilation and Airflow Networks").*

3.6 The ASHRAE Space Types tab

A `SpaceType` also carries five fields — Envelope Space Type, Lighting Space Type, Ventilation Space Type, Space Function, and Space Conditioning Category Change — shown on a separate **ASHRAE Space Types** tab in Column 3. These aren't loads; they classify the `SpaceType` for the ASHRAE 90.1 Appendix G baseline generator and have no effect on a simulation run outside that process. *(see "ASHRAE 90.1 Appendix G Baseline Generation")* covers them.

4. The ZoneType object

A `ZoneType` bundles the HVAC- and sizing-side defaults that pair with a `SpaceType`'s loads: thermostat configuration, a `Sizing:Zone`, `WaterUse:Equipment`, and a set of cascading defaults that ZoneHVAC equipment assigned to the zone can draw on.

4.1 Thermostat Configuration

The **Thermostat Configuration** field on the `ZoneType` itself selects one of `None`, `Simple`, `Detailed`, `StagedDualSetpoint`, or `ThermalComfort`. Choosing anything but `None` adds the matching thermostat sub-object to the `ZoneType` (and a `Sizing:Zone`, if one isn't already present).

`Simple` is what the quick-create dialog (§2) produces, and covers the common case of a single occupied/unoccupied setpoint schedule:

Field	Purpose
Zone or ZoneList Name	Set automatically to the `ZoneType`'s own `ZoneList`.
Setback Schedule Name	An on/off schedule: a value of 0 means the unoccupied setpoints apply, any value greater than 0 (usually 1) means the occupied setpoints apply. Left blank, this defaults to the `ZoneType`'s own ZoneHVAC Availability Schedule Name (§4.2).
Occupied Heating Setpoint / Un-Occupied Heating Setpoint	Default to 21°C / 18°C.
Occupied Cooling Setpoint / Un-Occupied Cooling Setpoint	Default to 26°C / 29°C.

Behind the scenes, `Simple` is EP3's own construction: on translation to EnergyPlus, it generates the two `Schedule:Compact` setpoint schedules, a `ZoneControl:Thermostat`, and a `ThermostatSetpoint:DualSetpoint` directly from the Setback Schedule Name and the four setpoints above — there's no separate `ZoneControl:Thermostat` object to open or edit while `Simple` is selected. `Detailed` exposes a real `ZoneControl:Thermostat` object instead, with the full generality (and field count) of the EnergyPlus object itself; `StagedDualSetpoint` and `ThermalComfort` are its staged- and comfort-based counterparts. ZoneControl:Thermostat / ThermostatSetpoint:DualSetpoint cover all three field-by-field.

4.2 Cascading ZoneHVAC defaults

Field	Purpose
ZoneHVAC Availability Schedule Name	Default availability schedule for ZoneHVAC equipment assigned to these zones.
Zone Supply Air Fan Operating Mode Schedule	Default fan operating mode schedule for ZoneHVAC fans connected to these zones.

Field	Purpose
ZoneHVAC Design Specification Outdoor Air Object Name	Default `DesignSpecification:OutdoorAir` (or DSOA SpaceList) for ZoneHVAC equipment assigned to these zones.
Design Specification ZoneHVAC Sizing Object Name	Default `DesignSpecification:ZoneHVAC:Sizing` object for the same equipment.

These same four fields also appear directly on the `Zone` object's own dialog, alongside a **Zone Aggregation Factor** field that has no `ZoneType` equivalent. A piece of ZoneHVAC equipment's own field always wins if set explicitly; otherwise EP3 falls back first to the value entered on the `Zone` itself, then to the value on the `Zone`'s assigned `ZoneType`, and only then to a hardcoded EP3 default (for example, an always-off schedule for availability). This is what "cascading" means here: a value set once on a `ZoneType` reaches every piece of equipment across every zone using that template, without needing to be entered per zone or per device, but a single zone or a single device can still opt out with its own value. Which equipment fields actually consume this cascade is covered in *(see "Zone-Level HVAC Equipment").*

4.3 Sizing:Zone

Every `ZoneType` with a thermostat configuration other than `None` carries a `Sizing:Zone`. Its full field set — supply air temperatures and humidity ratios, sizing factors, outdoor air method — is covered in *(see "Sizing HVAC Systems and Zones").*

4.4 WaterUse:Equipment

Field	Purpose
Peak Flow Rate	The fixture's maximum water draw.
Flow Rate Fraction Schedule Name	Fraction (0.0–1.0) applied to Peak Flow Rate, by time. Blank defaults to always 1.0.
Target Temperature Schedule Name	Desired mixed-water temperature at the fixture. Blank defaults to the hot water supply temperature.
Hot Water Supply Temperature Schedule Name / Cold Water Supply Temperature Schedule Name	Supply temperatures at the fixture; both blank by default (hot defaults to cold, cold defaults to `Site:WaterMainsTemperature`).
Sensible Fraction Schedule Name / Latent Fraction Schedule Name	Fraction of the water heating energy released to the zone as sensible or latent gain, rather than leaving with the drain. Both blank default to always 0.0.
End-Use Subcategory	Same purpose as on the internal loads objects in §3.

A `WaterUse:Equipment` added to a `ZoneType` is assigned the same way the loads objects in §3 are — at the `ZoneType` level (§5), not on the sub-object itself. It represents fixture demand — sinks,

showers, and similar draws appropriate to the zone — separate from whatever water heater actually satisfies that demand; the equipment and plant side of service hot water is covered in *(see "Central Plant Equipment").*

---

5. Assigning Spaces and Zones, and per-item overrides

5.1 Name mode vs. Selected Template mode

Which Spaces and Zones a `SpaceType`/`ZoneType` pair applies to depends on the model's **Update Templates Based On** setting — `Name` or `Selected Template` — covered as a model setting in *(see "The EP3 Interface and Model Settings").* That chapter covers the setting and switching between modes; this section covers what each mode means specifically for `SpaceType`/`ZoneType` assignment.

In `Name` mode — the default — a Space and a `SpaceType` are matched purely by name: a Space is claimed by whichever `SpaceType`'s naming pattern its own name contains, and a Zone is separately claimed by whichever `ZoneType`'s naming pattern its own name contains. These two matches happen independently of each other — nothing ties a Zone's `ZoneType` to its Spaces' `SpaceType` beyond the modeler having named things consistently.

In `Selected Template` mode, assignment instead happens through explicit fields — a **Space Template** field on `Space`, and a **Zone Template** field on `Zone` — and, unlike Name mode, a Zone's `ZoneType` assignment can inherit from its Spaces' `SpaceType` assignment, covered next.

5.2 Assigning in Selected Template mode

A Space's `SpaceType` is set directly, on that one Space's own **Space Template** field, on several Spaces at once with the right-click **"EP3 - Assign Space Template"** command *(see "Spaces and Zones")*, or from the `SpaceType` itself: a **Choose Spaces** button appears on a `SpaceType`'s own Column 3 (only in `Selected Template` mode) and opens a list of every Space in the model, one row each, with a choice between that `SpaceType` and Default/None — a bulk way to build up or edit one `SpaceType`'s full assignment at once, from the template's side rather than each Space's.

A Zone's `ZoneType` can be set the same three ways — its own **Zone Template** field, a bulk assignment command, or a matching **Choose Zones** button on the `ZoneType` itself — but has a fourth option besides: if the Zone's own **Zone Template** field is left blank, EP3 looks at every Space the Zone contains, and if they all carry the same `SpaceType`, the Zone inherits whichever `ZoneType` shares that `SpaceType`'s name — its paired companion template (§1) — automatically, with nothing to set on the Zone itself. Setting the Zone's own **Zone Template** field explicitly always takes precedence over that inherited companion, even if it names a different `ZoneType` than the one its Spaces would otherwise imply. If a Zone's Spaces don't all agree on one `SpaceType` — mixed assignments, or none at all — there's no single companion to inherit, and that Zone needs an explicit assignment of its own to get a `ZoneType` at all.

5.3 Per-item overrides

Once a Space or Zone is assigned, every field on its template's sub-objects applies to it by default. Any single field can still be overridden for one assigned Space or Zone without breaking the assignment or affecting any other Space or Zone sharing the same template — entering a value directly

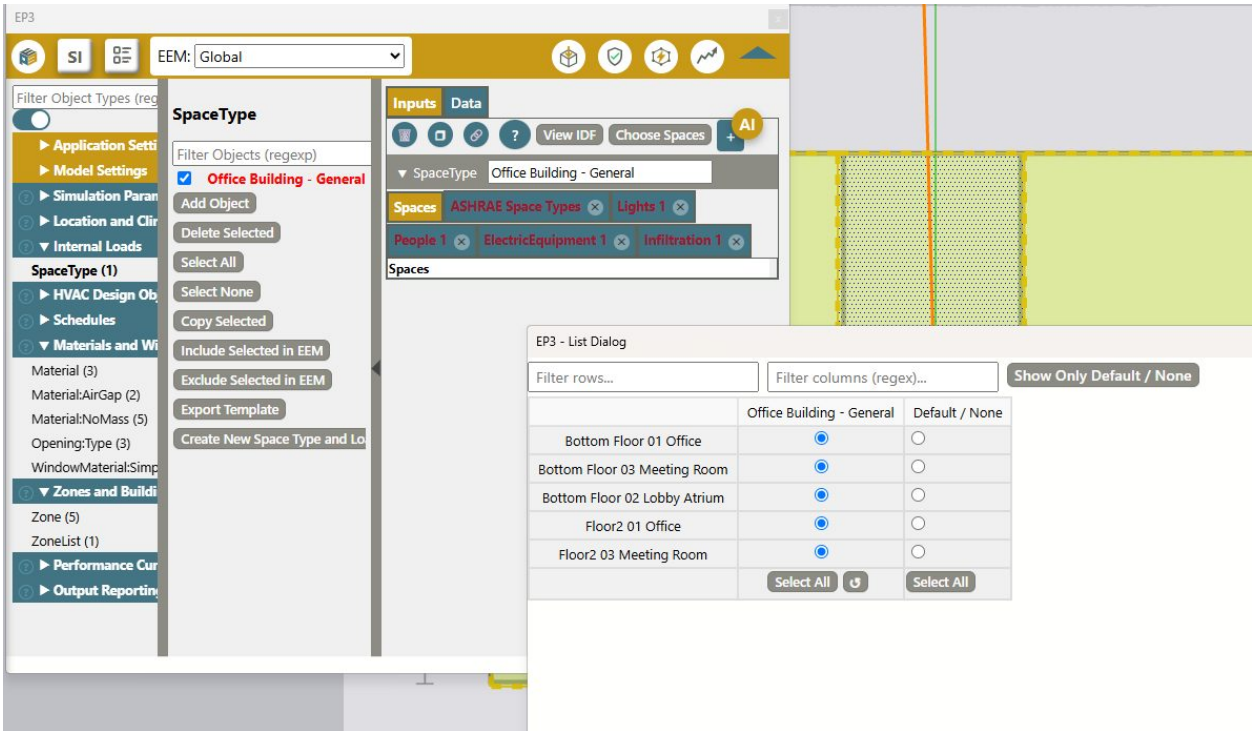


Figure 81: The "Choose Spaces" list dialog for the "Office Building - General" SpaceType, with each Space in the model as a row and a choice between that SpaceType and Default/None per row, plus Select All controls for each column.

on that field, for that one item, takes precedence over the template's value from then on. This is the mechanism to reach for on a templated project instead of detaching a Space or Zone into loose objects: a single conference room needing its own Lighting Power Density, for instance, doesn't require pulling it out of its `SpaceType` — override just that one field on that one Space.

The **Data** tab, alongside every sub-object's **Inputs** tab, lays every assigned Space or Zone out as its own column next to a **Template Field Values** column, one field per row — the place to compare and edit overrides across an entire `SpaceType`/`ZoneType` at once rather than one Space or Zone at a time. A field left alone displays with a rose-tinted background, matching the Template Field Values column exactly; entering a value of its own for that one Space or Zone clears the tint, marking it as that item's own override rather than an inherited value.

---

6. Loose internal loads without a template

A model imported into EP3 from outside — an IDF or epJSON file that never went through EP3's own `SpaceType`/`ZoneType` mechanism — carries its `People`, `Lights`, `ElectricEquipment`, `OtherEquipment`, and infiltration objects as loose objects, each pointing at a `Zone`, `Space`, `ZoneList`, or `SpaceList` directly through its own **Zone or ZoneList or Space or SpaceList Name** field (the same field that a `SpaceType`'s sub-objects have set automatically to their `SpaceList`, per §3). A `ZoneList` or `SpaceList` here is nothing more than a named, ordered group of Zones or Spaces — useful for pointing one loads object at several zones at once, with no template or cascading-default behavior attached.

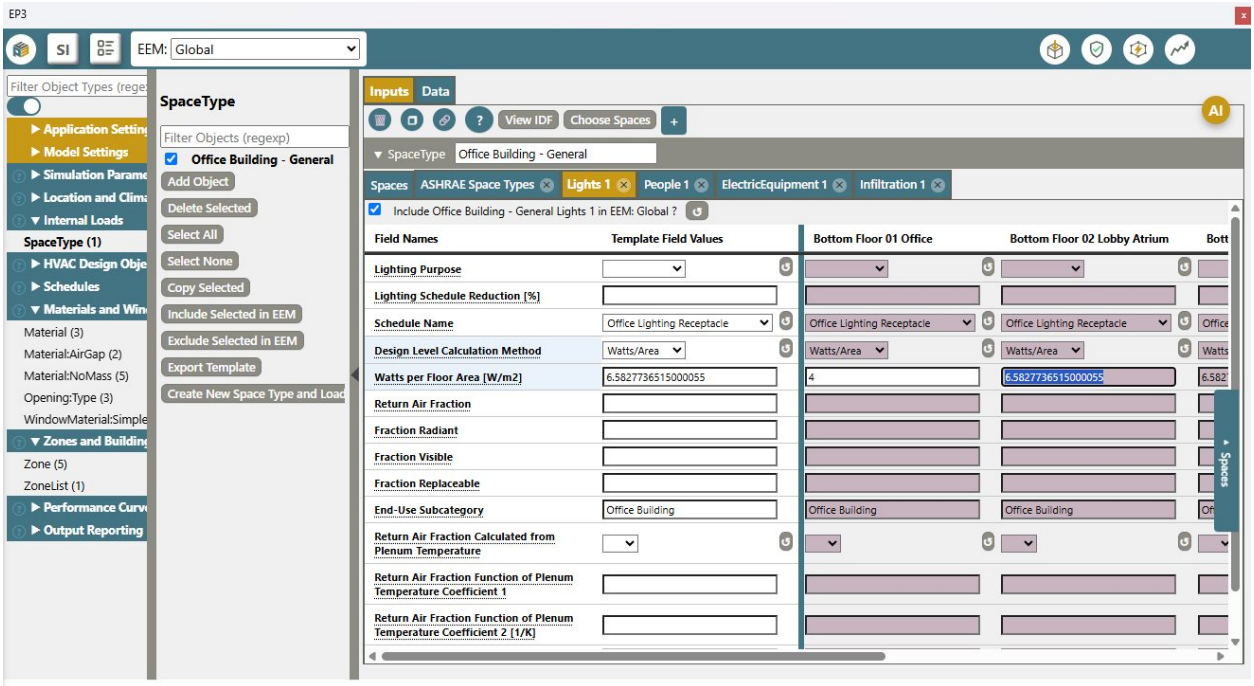


Figure 82: The Data tab for the "Office Building - General" SpaceType's Lights sub-object. Watts per Floor Area has been overridden on Bottom Floor 01 Office (plain background, entered as 4) while Bottom Floor 02 Lobby Atrium still shows the Template Field Values column's own inherited value, 6.5827736515000055, with the matching rose-tinted background.

Reading and editing a loose object this way is no different from working with the equivalent object in raw EnergyPlus — every field, default, and calculation method is identical to what §3 and §4 describe, just without a `SpaceType/ZoneType` bundling them together or providing per-item overrides. People, Lights, ElectricEquipment, and OtherEquipment cover field-by-field detail for anyone working this way directly.

Building a new project entirely in loose objects, bypassing `SpaceType/ZoneType` altogether, is possible but not the intended workflow — everything in §2 through §5 exists specifically to avoid it.

---

7. Reusing a template across projects

A `SpaceType/ZoneType` pair built for one project — whether from the quick-create dialog, the built-in library, or by hand — can be saved and reused in another project through the **Export Template** button on the Column 2 list, next to **Create New Space Type and Loads** (§2).

Export Template doesn't produce a file for EP3's general EnergyPlus file import — it writes the selected `SpaceType/ZoneType` pair directly into one of the folders registered under **Library Locations** for Internal Loads *(see "The EP3 Interface and Model Settings", §7.4)*, prompting to choose or confirm which one. The file it writes is an epJSON template, in the same format (and often the same folder) as the built-in library entries described in §1, carrying EP3-specific information beyond the plain EnergyPlus objects: the link between the two templates, any per-Space or per-

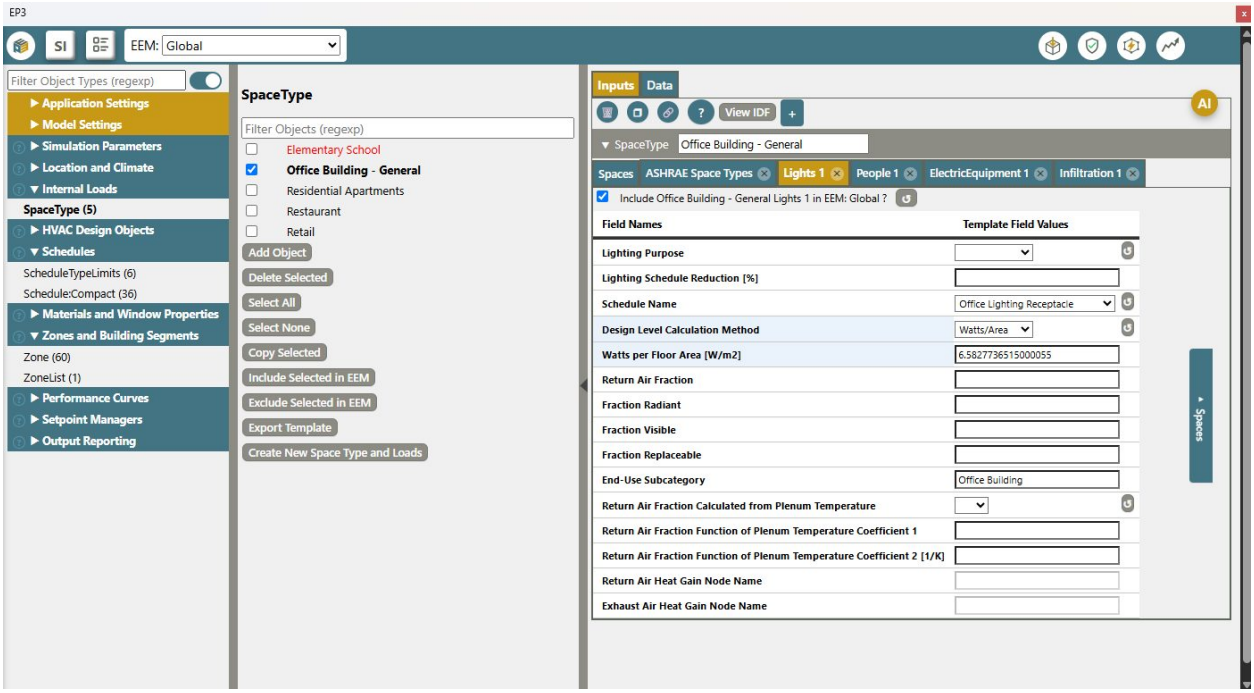


Figure 83: SpaceType in Column 3 with its Lights tab selected, alongside Column 2's SpaceType-specific Export Template and Create New Space Type and Loads buttons.

Zone overrides (§5.3), the thermostat configuration, and any `WaterUse:Equipment`. Once written, the new template shows up in the Wizard's Internal Loads tab the next time it's opened, alongside the built-in library, and is pulled into a model the same way (§1). Reusing a template is exclusively a Wizard/Library-path operation — there is no separate template-import dialog. This is the supported mechanism for building a firm- or project-standard set of `SpaceType`/`ZoneType` templates once and reusing them across projects.

10. Schedules

1. Overview

A schedule is EnergyPlus's mechanism for a value that varies over time. Dozens of other objects reference one by name wherever they need a time-varying input instead of a single fixed number: occupancy and lighting fractions, equipment availability, thermostat setpoints, fan operating mode, an outdoor-air-reset target, and many more.

This chapter covers the schedule objects themselves — the family of object types EP3 exposes, how to build a full year of data with each, and where EP3 gives you tools beyond typing raw EnergyPlus text by hand. It does not cover what a schedule *means* to a particular object that consumes it — a fan's Availability Schedule Name versus its Operating Mode Schedule, for instance, or the cascading zone-type-level schedule defaults for `ZoneHVAC` equipment. Those live in the chapter for the object that defines that meaning; see §8 for where to find them.

Every schedule other than `Schedule:Constant` ultimately resolves to one value per hour (or per sub-hourly timestep) for every hour of the year. EP3 gives you four ways to arrive at that:

- **`Schedule:Compact`** (§3) — the practical default, and the only one with a graphical editor.
- **`Schedule:Year`**, assembled from `Schedule:Week:Daily`/`Schedule:Week:Compact` and `Schedule:Day:*` building blocks (§4).
- **`Schedule:Constant`** (§5) — a single value, unchanging all year.
- **`Schedule:File`** (§6) — values read from an external file, computed however you like before the simulation runs.

---

2. Schedule Type Limits

Every schedule — including the `Schedule:Day:*` and `Schedule:Week:*` building blocks — points at a `ScheduleTypeLimits` object through its **Schedule Type Limits Name** field. This object holds none of the schedule's actual data; it defines the valid range and meaning of whatever data the schedule contains:

- **Lower Limit Value / Upper Limit Value** — the valid numeric range.
- **Numeric Type** — `Continuous` (any value in range) or `Discrete` (integers only).
- **Unit Type** — a category describing what the numbers mean: `Temperature`, `DeltaTemperature`, `Percent`, `Power`, `Capacity`, `Velocity`, `ActivityLevel`, `Angle`, `Availability`, `Control`, `ConvectionCoefficient`, `Mode`, `PrecipitationRate`, or the default, `Dimensionless`.

EP3 creates five `ScheduleTypeLimits` automatically the first time a model needs them, rather than requiring you to build them from scratch:

Name	Lower Limit	Upper Limit	Numeric Type	Unit Type	Typical use
Fraction	0	1	Continuous	Dimensionless	Occupancy/lighting/equipme operating fraction; most availability and multiplier schedules
Temperature	-60	200	Continuous	Temperature	Setpoint schedules, outdoor-air-reset targets
On Off	0	1	Discrete	Dimensionless	Binary on/off schedules
Control Type	0	4	Discrete	Dimensionless	Zone Control Type schedules driving `ThermostatSetpoint:*` selection (0–4: Uncontrolled/Heating/Coolin Cool/Dual-Setpoint)
Any Number	*(none)*	*(none)*	*(none)*	*(none)*	Values that shouldn't be range-checked at all — e.g. an Activity Level schedule in W/person

Like every other numeric field in EP3, Lower/Upper Limit Value are stored internally in SI regardless of which unit system your model displays. Only two Unit Type categories are ever auto-converted for display, though: **Temperature** and **DeltaTemperature**. Give a schedule a Unit Type of `Power`, `Velocity`, `Capacity`, or any of the other categories, and EP3 stores and displays whatever number you type as-is — switching your model between IP and SI display does not convert it. Unit Type still drives the graphical editor's y-axis and the range/discreteness checks described in §7; for anything other than Temperature/DeltaTemperature, it doesn't drive a unit conversion.

---

3. Schedule:Compact — the practical default

Almost every schedule you build directly in EP3 is a `Schedule:Compact`. Its underlying structure is EnergyPlus's own compact schedule grammar: one or more `Through:` blocks in chronological order (each covering the calendar up through a given date), each containing one or more `For:` lines

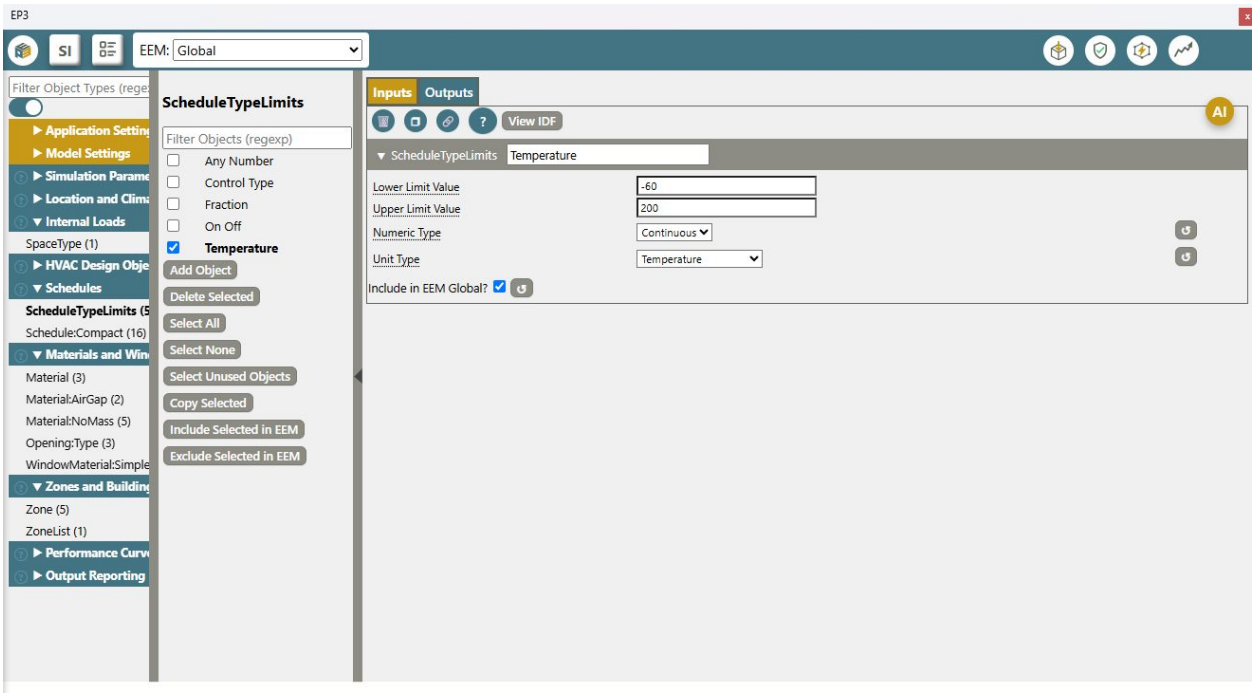


Figure 84: The "Temperature" ScheduleTypeLimits object's dialog, with Lower Limit Value, Upper Limit Value, Numeric Type, and Unit Type set.

naming which day types that section's values apply to, each followed by one or more `Until:` lines giving a value that holds from the previous transition up to the stated time.

3.1 Structure

A `Through:` block must be present for every day of the year — EP3 validates that the last block reaches `12/31` (§7 covers this and the rest of EP3's schedule validation). Within a block, every day type EnergyPlus recognizes must be covered exactly once, either individually (`Monday`, `Tuesday`, … `SummerDesignDay`, `WinterDesignDay`, `CustomDay1`, `CustomDay2`, `Holiday`) or through a group shorthand (`Weekdays`, `Weekends`, `AllDays`, `AllOtherDays`).

`Schedule:Compact`'s own dialog doesn't expose this as individual fields — the object stores it as a single free-text **Data** block. Typing or pasting valid compact-schedule text (comma-, semicolon-, or newline-separated) directly into that field works, but EP3's parser only accepts `Until:` transitions on the hour — a sub-hourly `Until: 7:30`, for instance, raises an error rather than being interpreted. For anything beyond a schedule you're pasting in from an existing source, the graphical editor (§3.2) is the intended way to build and edit `Schedule:Compact` data.

3.2 The graphical editor

Every `Schedule:Compact` object's dialog has an **"Edit Graphically"** button. Clicking it opens a chart-based editor with one panel per `Through:` block, in chronological order, each showing that block's day-type groups as separate step-line series across the 24 hours of a day.

Within a panel:

- **Editing a value** — drag a series' point at a given hour up or down (the chart shows the value

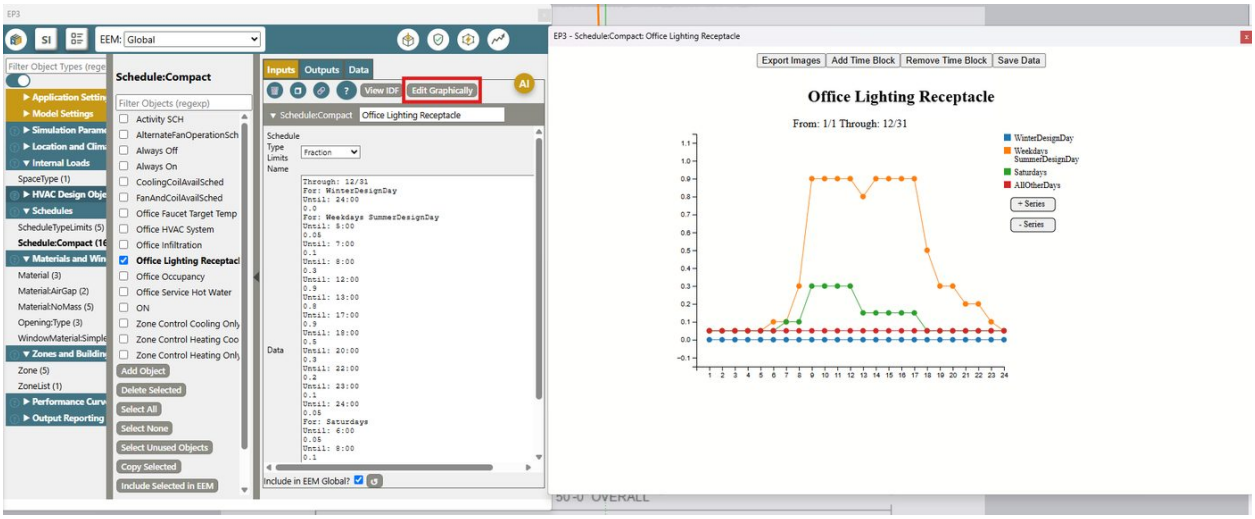


Figure 85: The "Edit Graphically" button on a Schedule:Compact object's dialog, and the resulting graphical editor open beside it, showing one Through: block with four day-type series.

live while dragging, clamped to the schedule's `ScheduleTypeLimits` range), or right-click a point to type an exact number directly.

- **Adding a series** — the **"+ Series"** button either copies an existing series (from this block or any other) or opens a day-type picker to build a new one from scratch.
- **Editing which day types a series covers** — click the series' label in the legend to reopen that same day-type picker, pre-checked with whichever day types it currently covers. Either way, the picker only lists day types not already claimed by another series in the same block — a block's day types must partition without gaps or overlap, so once, say, `WinterDesignDay` and `AllOtherDays` belong to other series, they no longer appear as choices.
- **Removing a series** — the **"- Series"** button; it has no effect if the block has only one series, since something has to keep covering every day type.
- **Adding or removing a whole `Through:` block** — **"Add Time Block"** opens a calendar to pick where a new split falls; **"Remove Time Block"** deletes an existing one outright. Either way EP3 sorts the remaining blocks into chronological order, and a block's own title bar (`From: ... Through: ...`) is clickable to reopen the calendar and adjust its date range.

Dragging, typing, or adding/removing a series updates the chart immediately but doesn't commit anything to the `Schedule:Compact` object until you click **"Save Data"**; closing the editor without having clicked it prompts you to save or discard what changed. An **"Export Images"** button saves every visible chart panel as an image for documentation.

This same editor also appears in the Model Wizard's schedule-review step, letting you check and adjust every `Schedule:Compact` the templates you picked brought in before finishing setup — *(see "The Model Wizard").*

3.3 EP3's built-in default schedules

Alongside the five default `ScheduleTypeLimits` (§2), EP3 creates a small set of default `Schedule:Compact` objects the first time a model needs them:

- **Always On / Always Off** — constant 1 / constant 0, `Fraction` type.

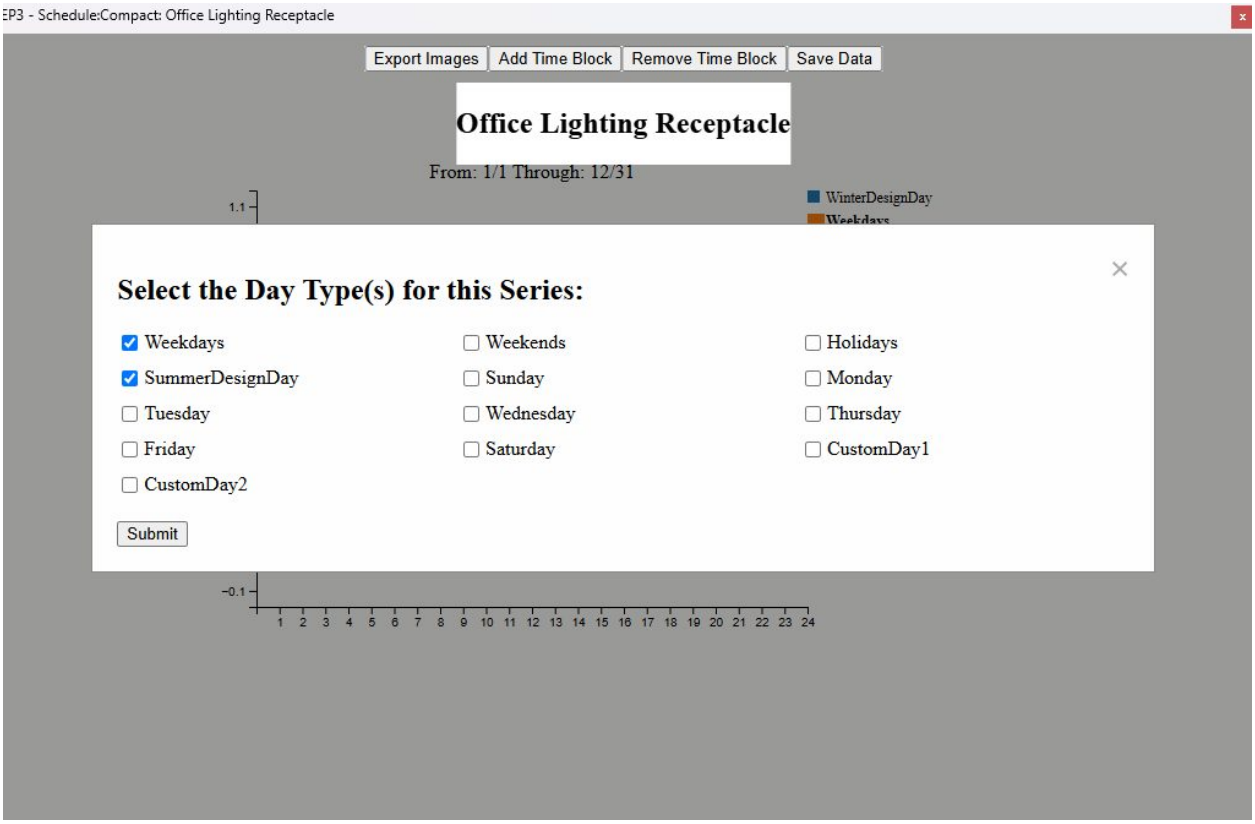


Figure 86: The "Select the Day Type(s) for this Series" modal, listing only the day types not already covered by another series in this block.

- **Zone Control Heating Cooling Type Schedule / Zone Control Heating Only Type Schedule / Zone Control Cooling Only Type Schedule** — constant `Control Type` values (4, 1, and 2 respectively) for the three corresponding thermostat control modes.
- **Activity SCH** — a constant value under the `Any Number` type, used as a default People Activity Level schedule.

These exist so that objects needing a simple constant schedule (a thermostat's control-type schedule, an always-available piece of equipment) have one ready-made rather than requiring you to build a trivial one-value `Schedule:Compact` yourself every time.

---

4. Schedule:Year, Schedule:Week:Daily, Schedule:Week:Compact, and Schedule:Day:*

`Schedule:Year` builds a full year out of smaller, reusable pieces instead of one continuous block of text:

- A `Schedule:Year`'s **Schedule Weeks** field is a table of rows, each pairing a **Schedule Week Name** with a **Start Month/Start Day/End Month/End Day** date range.
- Each row points at either a **`Schedule:Week:Daily`** — which names one `Schedule:Day:*` object for each of 12 required day-type fields (Sunday through Saturday, Holiday, SummerDesignDay, WinterDesignDay, CustomDay1, CustomDay2) — or a **`Schedule:Week:Compact`**, whose **Data** field instead takes a table of `Daytype List/Schedule Day Name` pairs, grouping day types the same way a `Schedule:Compact`'s `For:` line does rather than requiring a separate field per individual day.
- Each `Schedule:Day:*` object supplies one day's worth of values, in one of three forms: **`Schedule:Day:Hourly`** (24 explicit Hour 1–Hour 24 fields), **`Schedule:Day:Interval`** (a table of `Time/Value Until Time` pairs, the same Until-time structure as `Schedule:Compact` but as literal rows instead of free text), or **`Schedule:Day:List`** (a fixed-size **Minutes per Item** interval, giving one value every N minutes rather than at named times).

This path has no graphical editor — you build and edit it entirely through each object's own field-by-field dialog: the `Schedule:Year`'s Schedule Weeks table, each `Schedule:Week:*` object it references, and each `Schedule:Day:*` object those in turn reference.

Reach for `Schedule:Year` when you're reconstructing or preserving a schedule that's already expressed this way — from an imported EnergyPlus file, a schedule library, or a reference standard — rather than flattening it into a single `Schedule:Compact`, or when the same weekly pattern recurs across multiple non-contiguous date ranges and you'd rather reference one `Schedule:Week:*` object from several rows than retype the same `For:/Until:` block into every seasonal period. For a schedule you're building from scratch, `Schedule:Compact` and its graphical editor is the more direct path.

---

5. Schedule:Constant

`Schedule:Constant` holds a single **Hourly Value**, applied for every hour of every day of the year — no `Through:/For:/Until:` structure at all. Use it for a value that genuinely doesn't vary, rather than building a one-line `Schedule:Compact` just to express a single number.

6. Schedule:File

`Schedule:File` reads its values from an external file instead of storing them in the object itself:

- **File Name** — path to the file.
- **Column Number** — which column holds the values.
- **Rows to Skip at Top** — header rows to ignore.
- **Number of Hours of Data** — 8760, or 8784 for a run period spanning a leap year.
- **Column Separator** — Comma, Semicolon, Space, or Tab.
- **Interpolate to Timestep** — how to reconcile the file's resolution with the simulation timestep (No / Average / Linear).
- **Minutes per Item** — the file's own resolution (60 for hourly data).
- **Adjust Schedule for Daylight Savings** — whether to shift the file's values for daylight savings, or use them exactly as given.

The technique `Schedule:File` enables: precompute a value for every hour (or sub-hourly timestep) of the year outside EnergyPlus — in a spreadsheet or script — whenever the value is a deterministic function of something already known before the simulation runs, rather than something that only exists once the simulation is running. The weather file's own hourly data (dry-bulb temperature, for instance) is known in advance, since it's read from a fixed file; a zone's actual temperature or a piece of equipment's actual output is not, since it depends on the simulation as it unfolds. The same reasoning extends past pure weather-driven cases: any pattern you can fully define ahead of time and want held identical across separate runs — a detailed, hour-by-hour occupancy pattern for a building where occupancy is irregular but should be reproducible run to run (a hotel, where which rooms are occupied on which night varies but shouldn't reshuffle between a baseline run and an EEM alternative), for example — can be generated once, saved to a file, and referenced the same way.

EP3 does not touch the file's contents. Unlike data typed directly into a `Schedule:Compact` or `Schedule:Constant` dialog — which EP3 converts between your model's displayed unit system and its internal SI storage for Temperature/DeltaTemperature-type schedules — a `Schedule:File`'s values are read by EnergyPlus directly from the file at simulation time. Every value in the file must already be in the units EnergyPlus expects for that schedule's Unit Type (Celsius for a Temperature-type schedule) regardless of which unit system your model is currently displaying.

Schedule:File covers the object's fields in full.

Worked example: an outdoor-air-reset hot water supply temperature schedule

Suppose a heating hot water loop's supply temperature should reset with outdoor air temperature along a four-point curve — more breakpoints than a single `SetpointManager:OutdoorAirReset` object's two-segment reset can express for one setpoint node list (SetpointManager:OutdoorAirReset):

Outdoor Air Temperature	Hot Water Supply Setpoint
<= 10°F	180°F
30°F	160°F
50°F	140°F
>= 65°F	120°F

with linear interpolation between adjacent breakpoints.

Building the file:

1. Pull the hourly dry-bulb temperature column from the same weather file your model already uses — it's fixed and known before the simulation runs. *(see "Weather, Climate, and Design Conditions")* covers where that file is assigned.
2. For every hour, apply the reset curve to that hour's temperature (interpolating between the two nearest breakpoints, clamping at the two ends) to get that hour's setpoint.
3. Convert every resulting value from °F to °C — the units a Temperature-type schedule needs internally, independent of which units your model displays in (see the limitation above).
4. Save one column of 8760 values (8784 for a leap year) as a CSV, with as many header rows as you'd like `Rows to Skip at Top` to skip.
5. Create the `Schedule:File` object: File Name pointing at the CSV, Column Number = the column holding your computed values, Rows to Skip at Top matching your header, Number of Hours of Data = 8760, Column Separator = Comma, Minutes per Item = 60, Interpolate to Timestep = Linear or Average if your simulation timestep is finer than hourly, Adjust Schedule for Daylight Savings per your project's convention.
6. Reference the `Schedule:File`'s name wherever a Schedule Name field wants it — a `SetpointManager:Scheduled`'s Schedule Name, in place of `SetpointManager:OutdoorAirReset`.

Checking one hour's value: take an hour where the weather file reports 15°F outdoor air temperature, between the 10°F/180°F and 30°F/160°F breakpoints:

fraction = (15 − 10) / (30 − 10) = 0.25 Setpoint = 180 − 0.25 × (180 − 160) = **175°F**

Converted to the °C the file actually stores: (175 − 32) × 5/9 = **79.4°C**.

Check via the same interpolation done in °C directly: 10°F = −12.2°C, 30°F = −1.1°C, 180°F = 82.2°C, 160°F = 71.1°C, and 15°F = −9.4°C:

fraction = (−9.4 − (−12.2)) / (−1.1 − (−12.2)) = 2.8 / 11.1 = 0.25 Setpoint = 82.2 − 0.25 × (82.2 − 71.1) = **79.4°C** — converting back, (79.4 × 9/5) + 32 = 175.0°F, matching the first path.

---

7. Design day profiles

A schedule's `SummerDesignDay` and `WinterDesignDay` values feed directly into `SizingPeriod:DesignDay` sizing runs, and what those values *should* be depends on what the schedule is the primary schedule for. EP3 checks this automatically, under the Schedules tab of the Validation dialog (*(see "Model Troubleshooting and Quality Assurance")*) — one collapsible subsection per schedule, expanded and flagged only when it finds an issue.

SummerDesignDay, regardless of what the schedule drives, should equal the schedule's own typical annual peak value — cooling design sizing assumes the coincident peak condition, so a design-day value lower than the schedule's normal peak understates the cooling load the equipment needs to handle.

WinterDesignDay differs depending on what the schedule is the primary schedule for:

- **Infiltration** (`ZoneInfiltration:DesignFlowRate`, `ZoneInfiltration:EffectiveLeakageArea`, `ZoneInfiltration:FlowCoefficient`) — WinterDesignDay should equal that same annual

peak value. Infiltration doesn't stop because it's a design day; using anything less than the typical peak understates the heating load the equipment needs to overcome.

- **Internal loads** (`People`, `Lights`, `ElectricEquipment`, `GasEquipment`, `OtherEquipment`) — WinterDesignDay should be zero for every hour. Heating design sizing conventionally assumes no internal gains at all, so the equipment is sized for the worst case rather than one that credits gains that may not be there when they're needed most.

If the same schedule is the primary schedule for both an infiltration object and an internal-loads object, EP3 flags this directly — the two uses need contradictory WinterDesignDay values, and the fix is a second, dedicated schedule for one of the two uses rather than trying to satisfy both from one schedule.

This check only looks at a schedule's role as an object's *primary* schedule field (a `People` object's Number of People Schedule Name, for instance) — a schedule referenced elsewhere, such as a `People` object's Activity Schedule, isn't evaluated against these design-day rules, and an empty result here doesn't mean the schedule is unused.

---

8. What's covered elsewhere

- **Availability Schedule Name and Supply Air Fan Operating Mode Schedule Name**, including the zone-type-level cascading defaults for `ZoneHVAC` equipment — *(see "Fans and Fan Power").*
- **`SpaceType/ZoneType`'s own cascading default schedules** (thermostat schedules and the rest of the internal-loads/HVAC defaults bundle) — *(see "Internal Loads, Space Types, and Zone Types").*
- **The Validation dialog** this chapter's design-day-profile check runs inside of, and everything else it checks — *(see "Model Troubleshooting and Quality Assurance").*

End of draft chapter. Outstanding items for the next pass: resolve the XREF/EXTLINK placeholders once their targets exist; confirm final document format (HTML vs. PDF vs. other).

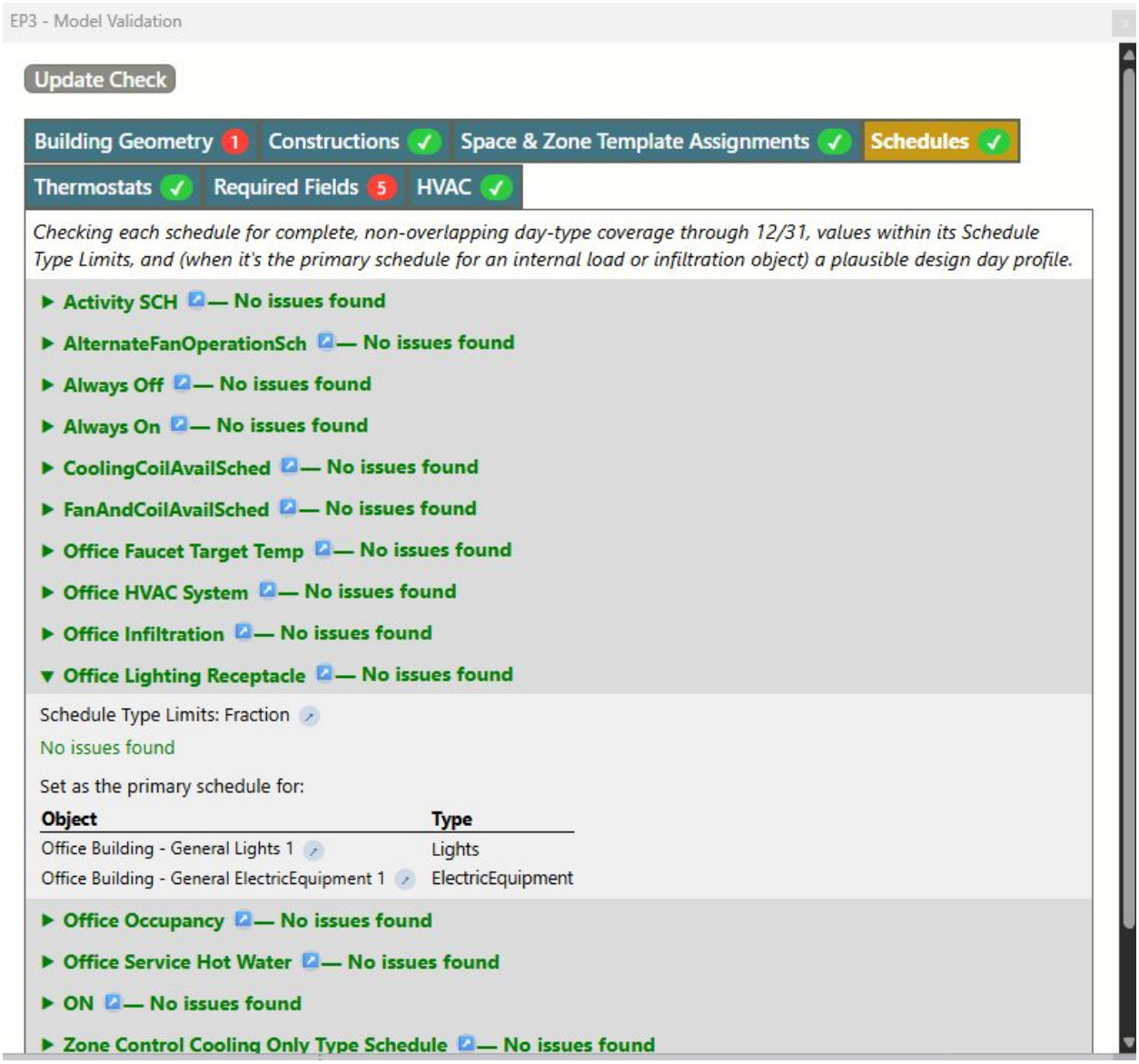


Figure 87: The Validation dialog's Schedules tab, with "Office Lighting Receptacle" expanded to show its Schedule Type Limits reference and the objects it's the primary schedule for.

11. Heating and Cooling Loads: From Design Day to Annual Profile

1. Overview

Geometry and constructions define how a building exchanges heat with the outdoors *(see "Creating Building Geometry") (see "Constructions and Materials")*, windows and shading define how solar radiation enters it *(see "Windows, Shading, and Daylighting")*, internal loads and their schedules define what generates heat inside it *(see "Internal Loads, Space Types, and Zone Types") (see "Schedules")*, and design days establish the conditions to size against *(see "Weather, Climate, and Design Conditions")*. This chapter covers what EnergyPlus does with all of that together: how it turns those inputs into a heating or cooling load for a zone, at a single peak hour and across a full year.

This is a conceptual chapter rather than an object-by-object one — it explains the heat balance mechanism itself, which is standard EnergyPlus/building-physics content rather than something EP3 configures. Two boundaries worth stating up front:

- This chapter stops at the zone's load — a sensible and latent quantity, in watts or Btu/h, at a point in time. What equipment actually does with that load — the psychrometric process of a cooling coil, sensible heat ratio, dehumidification — is *(see "Air-Side System Fundamentals")*.
- This chapter doesn't cover `Sizing:Zone`, `Sizing:System`, or `Sizing:Plant` field-by-field. It covers what those objects are consuming when they autosize. Their fields are, respectively, *(see "Zone-Level HVAC Equipment and Zone Sizing")*, *(see "Air Loops, Branches, and Nodes")*, and *(see "Plant Loops and Setpoint Control")*.

---

2. The Zone Heat Balance

EnergyPlus does not calculate a cooling or heating load with a single formula evaluated once. It solves a heat balance — one equation for each surface, one for the zone's air — at every simulation timestep, whether that timestep belongs to a `SizingPeriod:DesignDay` or to the annual `RunPeriod`. A zone's load at any given moment is the net result of everything below acting on that heat balance simultaneously. EnergyPlus Engineering Reference, "Heat Balance Model Overview" covers the full derivation; ASHRAE Handbook—Fundamentals' load calculation chapter covers the same heat balance method from a load-calculation practitioner's perspective. ASHRAE Handbook—Fundamentals, "Nonresidential Cooling and Heating Load Calculations"

Conduction through walls, roofs, and floors is driven by the constructions assigned to them. By default EnergyPlus solves this with Conduction Transfer Functions (CTFs), which account for a construction's thermal mass — a heavyweight wall doesn't transmit an outdoor temperature swing to its interior surface instantly, but delayed and damped, over hours. This is why a wall's conductive contribution to a zone's load doesn't track outdoor temperature in real time. EnergyPlus Engineering Reference, "Conduction Transfer Function Module"

Solar radiation enters directly through glazing (transmitted solar, tied to window properties covered in Ch.8) and lands on both interior and exterior opaque surfaces, where it's absorbed and re-emitted as the surface itself heats. How EnergyPlus resolves *where* that solar radiation lands inside a zone is controlled by the `Building` object's **Solar Distribution** field (group "Simulation Parameters," alongside **North Axis** — *(see "Creating Building Geometry", §2)*) — options range

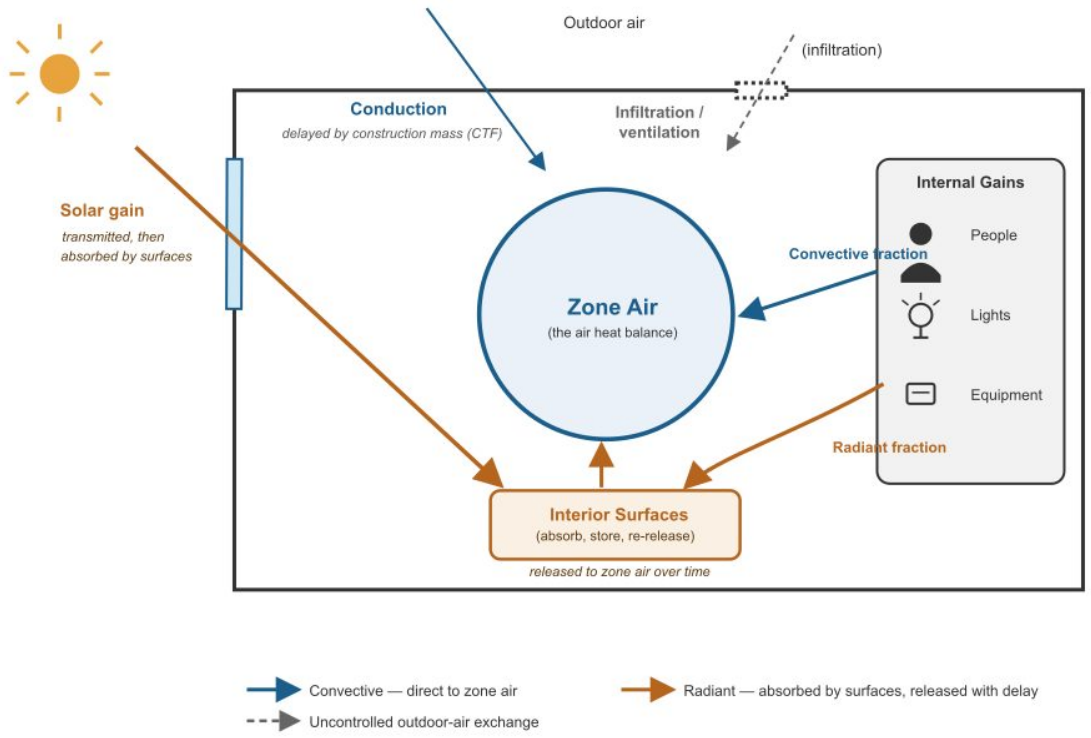


Figure 88: A single zone's heat balance: conduction (through the exterior wall) and infiltration feed the zone air node directly; solar gain is absorbed by interior surfaces and released with delay; internal gains (people, lights, equipment) split between a convective fraction reaching zone air immediately and a radiant fraction absorbed by interior surfaces first.

from crediting all interior solar to the floor (`MinimalShadowing`) to actually ray-tracing it onto whichever interior surfaces it would strike (`FullInteriorAndExterior`). Most projects can leave this at its default without a specific reason to change it. Building

Internal gains — `People`, `Lights`, `ElectricEquipment`, `OtherEquipment` — each split their output between convective heat (delivered to the zone air immediately) and radiant heat (absorbed by surfaces first, then released into the zone air over time, the same way absorbed solar is). The split is set per object: `People` defaults **Fraction Radiant** to 0.30; `Lights` defaults both **Fraction Radiant** and **Fraction Visible** to 0.0 (fully convective, unless changed); `ElectricEquipment`/`OtherEquipment` default **Fraction Latent**, **Fraction Radiant**, and **Fraction Lost** all to 0.0, meaning a plug load is fully convective by default. `People` additionally splits its output between sensible and latent heat (respiration and perspiration) — driven by its **Activity Level Schedule Name**, with **Sensible Heat Fraction** available to override the program's own calculated split. *(see "Internal Loads, Space Types, and Zone Types", §3.1–3.4)* covers these fields directly.

Infiltration and ventilation exchange outdoor air with the zone, carrying both a sensible load (proportional to the indoor-outdoor temperature difference) and, in humid or dry climates, a latent one (proportional to the indoor-outdoor humidity difference). Infiltration is covered in *(see "Internal Loads, Space Types, and Zone Types", §3.5)* for its object fields and *(see "Infiltration, Natural Ventilation and Airflow Networks")* for how its rate is chosen and how it interacts with mechanical ventilation.

The radiant/convective split above, combined with each surface's own thermal mass, is why a zone's *load* rarely peaks at the same moment as its strongest single driver — a west-facing zone's peak cooling hour is shaped as much by how quickly stored radiant heat re-releases into the air as by when solar gain itself is highest.

Two more "Simulation Parameters"-group objects govern the heat balance calculation method itself: `HeatBalanceAlgorithm` (default `ConductionTransferFunction`; alternatives like `CondFD` or `CombinedHeatAndMoistureFiniteElement` exist for projects specifically studying moisture transport through assemblies) and `ZoneAirHeatBalanceAlgorithm` (governs how the zone air node itself is solved, including whether Spaces sharing a Zone are summed as a `Coincident` or `NonCoincident` load — §5). Both are edited through the same generic object-list dialog as any other object in their group; neither needs to be touched for a typical project.

---

3. Cooling Loads

Cooling load is the more dynamic of the two, because nearly every driver in §2 is actively working to add heat: conduction (whenever outdoor temperature exceeds indoor), solar gain, internal gains, and often infiltration too if outdoor air is warmer or more humid than the zone. None of these arrive at the same moment or with the same time lag, so a zone's peak cooling hour is a genuine result of the heat balance rather than something read directly off the design day's peak dry-bulb temperature. A heavyweight, west-facing zone can peak well after the design day's hottest hour, once solar heat stored earlier in the afternoon finishes working its way back out of the walls; a lightweight, east-facing zone with large internal gains might instead peak close to midday, driven more by occupancy and equipment than by envelope.

Sensible cooling load is the sum of every sensible contribution in §2: conductive gain, transmitted

and absorbed solar, the convective and released-radiant fractions of internal gains, and sensible infiltration/ventilation gain.

Latent cooling load is tracked separately, and comes from a narrower set of sources: `People` (respiration and perspiration, per their Activity Level Schedule Name and Sensible Heat Fraction), infiltration and ventilation air that's more humid than the zone, and any `OtherEquipment`/`ElectricEquipment` object with a nonzero **Fraction Latent** (a process load releasing moisture, for instance). This sensible/latent split belongs to the building side of the model — it's a property of the zone's own load, independent of whatever equipment eventually serves it. How a cooling coil actually responds to a given sensible/latent combination — its own sensible heat ratio, and what happens when the load's latent fraction exceeds what the coil naturally removes — is *(see "Air-Side System Fundamentals").*

Which design day and hour actually governs a zone's peak cooling load is not a fixed answer — it depends on the interaction of orientation, mass, and internal-gain schedule described above, evaluated across whichever `SizingPeriod:DesignDay`/`SizingPeriod:WeatherFileConditionType` objects exist in the model *(see "Weather, Climate, and Design Conditions", §4).* Reading which design day and hour EnergyPlus actually selected for a given zone is part of `Sizing:Zone`'s own output, covered in *(see "Sizing HVAC Systems and Zones").*

---

4. Heating Loads

A heating design day still runs the full dynamic heat balance from §2 — EnergyPlus does not switch to a simplified steady-state calculation for heating the way a manual hand-calculation method might. What changes is the design day itself and the schedules run against it, both built specifically to represent a conservative, near-worst-case condition rather than a typical operating day:

- A `SizingPeriod:DesignDay` for heating conventionally sets **Daily Dry-Bulb Temperature Range** to 0.0°C *(see "Weather, Climate, and Design Conditions", §4.3.1)* — the whole day sits at the same minimum design temperature, rather than the diurnal swing a real winter day would have. This removes any benefit the building's thermal mass might otherwise bank against a milder overnight low.
- Solar gain contributes essentially nothing on a heating design day, both because the design condition is winter and because — unlike cooling, where the design day's date is chosen to catch the sun near its most direct — a heating design day carries no expectation of favorable solar timing.
- Internal gains are excluded by the schedules themselves, not by any special heating-specific calculation. EP3's built-in internal-loads schedule library sets `People` and `Lights`/`ElectricEquipment` schedules to a fraction of 0.0 for every hour of `WinterDesignDay`, while grouping `SummerDesignDay` with the normal weekday operating profile. This reflects a standard heating-design convention: a heating system has to be able to recover the zone from a cold snap even if it coincides with an unoccupied period — nights, weekends, or a holiday shutdown — so no credit is taken for occupant or equipment heat that can't be relied on. Nothing in EnergyPlus applies this automatically; it exists because the schedule's `For: WinterDesignDay` rule says so. A custom schedule built without an explicit `WinterDesignDay` rule of its own won't get this treatment for free — see the pitfall in §7.

With solar and internal gains effectively removed, a heating design load is dominated by conduction

and infiltration — both driven directly by the indoor-outdoor temperature difference at the design condition, with no offsetting gains left to reduce them. Heating load is also, barring a project that specifically models humidification, purely sensible: there's no equivalent to cooling's separately-tracked latent component, since no significant moisture source remains active once occupancy and equipment schedules are zeroed.

---

5. From Peak Load to Annual Profile

The annual `RunPeriod` solves the exact same heat balance described in §2, at every timestep across the simulated year — the difference from a design day is only which weather and which schedules drive it: real recorded weather-file conditions instead of a fixed design condition, and each schedule's actual weekday/weekend/holiday values instead of a zeroed or peak-occupancy design-day override. The result is a continuous sensible and latent load for each zone across the year, rather than a single peak value — the annual load profile.

A building-level or system-level peak is not simply the sum of every zone's individual peak. Zones facing different directions, or running different occupancy schedules, peak at different hours — some at 9am, some at 3pm — so the load actually seen by equipment serving multiple zones together is smaller than the arithmetic sum of each zone's own worst hour. This diversity is why `ZoneAirHeatBalanceAlgorithm`'s **Type of Space Sum to Use** offers both `Coincident` (sum whatever each Space's load actually is at the same simultaneous hour) and `NonCoincident` (sum each Space's own peak regardless of timing, the more conservative choice) for a Zone composed of multiple Spaces *(see "Spaces and Zones")* — and the same coincident-vs-non-coincident question recurs one level up, at `Sizing:System`, when combining multiple zones onto one air system. Both are `Sizing:Zone`/`Sizing:System` fields, covered in *(see "Sizing HVAC Systems and Zones")*.

Actually viewing or charting the annual load profile — output variables, the results viewer, EUI-level summaries — is covered in *(see "Viewing and Interpreting Results")*.

---

6. How This Feeds HVAC Sizing

`SimulationControl`'s sizing flags (§6 of *(see "Weather, Climate, and Design Conditions")*) gate whether `Sizing:Zone`, `Sizing:System`, and `Sizing:Plant` run at all. When they do, what they consume is exactly what this chapter describes: the governing coincident sensible and latent peak the heat balance produces for each zone, and the hour and design day it occurred on — nothing about the load calculation itself is configured on `Sizing:Zone`; its own fields (supply air conditions, sizing factors, outdoor air method) determine how that peak load is converted into a design airflow and capacity, which is *(see "Sizing HVAC Systems and Zones")*'s territory.

A zone's calculated peak load and the installed capacity of the equipment ultimately serving it are related but not identical figures — sizing factors, equipment selection increments, and safety margins mean installed capacity is normally somewhat larger than the bare calculated load. Don't expect an equipment output variable to reproduce the zone sizing peak exactly.

---

7. Common Pitfalls

Building a custom internal-loads schedule without an explicit `WinterDesignDay` rule. EP3's built-in library schedules (§4) zero out occupancy and equipment loads for the heating design day deliberately, through an explicit schedule rule — not through any automatic EnergyPlus behavior. A hand-built or imported schedule that never sets a `WinterDesignDay` value falls through to whatever its catch-all rule provides, which may credit a full day of occupied-hours internal gains against the heating design load — understating it for exactly the unoccupied-cold-snap scenario the convention exists to protect against.

Assuming the design day with the highest cooling dry-bulb temperature automatically produces every zone's peak cooling load. Orientation, thermal mass, and internal-gain timing can shift a given zone's actual peak to a different hour, or even a different design day, than dry-bulb alone would suggest (§3).

Treating "zone load" and "equipment capacity" as the same number. The load calculated here is an input to sizing, not the installed capacity that results from it (§6) — safety factors and real equipment selection mean they won't match exactly.

Discounting infiltration because mechanical ventilation is already modeled. Infiltration is a separate, uncontrolled outdoor-air path that contributes to both heating and cooling loads regardless of whatever mechanical ventilation is also present — see *(see "Infiltration, Natural Ventilation and Airflow Networks")* for how it's distinguished from mechanical ventilation.

End of draft chapter. Outstanding items for the next pass: resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

12. Air-Side System Fundamentals

1. Overview

Every chapter in Part IV assumes a shared vocabulary: sensible and latent load, humidity ratio, supply air condition, economizer, dedicated outdoor air. EnergyPlus's air-side objects are built directly on these terms, and EP3 exposes them as fields without redefining them — a "Zone Cooling Design Supply Air Humidity Ratio" field or a "Cooling Coil Gross Rated Sensible Heat Ratio" field means exactly what the underlying psychrometrics says it means. This chapter builds that vocabulary once, so later chapters can use it without re-deriving it.

Unlike most chapters in this manual, this one leans on general HVAC engineering and building physics more than on EP3's own code or dialogs. EP3 does not compute psychrometric properties itself — it collects the input values (design temperatures, humidity ratios, sensible heat ratios, economizer limits) that EnergyPlus's own psychrometric routines consume at simulation time. Where a concept below does correspond to a real EP3 field, that connection is called out explicitly and verified against the field labels EP3 actually uses; the underlying physics is presented as background knowledge.

This chapter does not walk through any EP3 dialog step by step. Each concept introduced here is picked up again, in EP3-specific mechanical detail, in the chapter that owns it.

---

2. Psychrometrics — properties and the chart

Conditioning air means controlling two things at once: its temperature and the amount of moisture it carries. The two don't move together — a process can change one while holding the other fixed, or change both — and an air-side system has to account for both separately because they draw on different physical mechanisms and different amounts of equipment capacity to control. A cooling coil's load splits into a sensible part (removing heat, dropping dry-bulb temperature) and a latent part (condensing moisture out of the airstream), and a coil sized only for the sensible part will under-dehumidify no matter how much sensible capacity it has (§3). An economizer's "is outdoor air favorable" decision can come out differently depending on whether temperature or total heat content — which includes the energy carried by moisture — is what's being compared (§6). None of this is visible from dry-bulb temperature alone: describing an air sample's condition completely, and predicting what happens when two airstreams mix or a coil pulls moisture out of one, requires tracking temperature and moisture content together. That's what the properties below, and the chart that relates them, exist to do.

Moist air's condition is fully described by two independent properties (at a given barometric pressure); everything else can be derived from them. The properties that come up repeatedly in EnergyPlus fields:

- **Dry-bulb temperature (T)** — what an ordinary thermometer reads. The default sense of "temperature" everywhere else in this manual.
- **Humidity ratio (W)** — mass of water vapor carried per unit mass of dry air (kg/kg). This is the property EnergyPlus's sizing and coil objects almost always use to represent moisture content — not relative humidity — because it doesn't change when dry-bulb temperature changes at constant moisture content.

- **Relative humidity (RH)** — the ratio of actual water vapor pressure to the saturation vapor pressure at the same dry-bulb temperature. Familiar and intuitive, but temperature-dependent: the same air parcel's RH changes as it's heated or cooled even though it hasn't gained or lost moisture. EnergyPlus humidistat setpoints are expressed in RH; most sizing and coil-performance fields are not.
- **Wet-bulb temperature (Twb)** — the temperature air reaches if cooled by evaporating water into it, at constant total heat content, until it saturates. Used directly by evaporative-cooling equipment and by weather-file design conditions.
- **Dew point (Tdp)** — the temperature at which an air sample becomes saturated if cooled at constant humidity ratio, with no other process. Appears directly in fixed-dew-point economizer limits and design-day weather data.
- **Specific enthalpy (h)** — total heat content of the moist air, per unit mass of dry air (kJ/kg). Combines sensible and latent heat into one number, which is exactly why enthalpy-based economizer logic behaves differently from dry-bulb-only logic (§6).

A **psychrometric chart** plots any two of these against each other (conventionally dry-bulb temperature on the horizontal axis, humidity ratio on the vertical) with curves of constant RH, wet-bulb temperature, and enthalpy overlaid, so that any one state point fixes all the others. Three process types recur constantly in air-side HVAC and read as specific moves on the chart:

- **Sensible heating or cooling** — a horizontal move (constant humidity ratio, dry-bulb temperature changes). No moisture is added or removed.
- **Cooling with dehumidification** — a move down and to the left, with both dry-bulb temperature and humidity ratio decreasing. This is what a cooling coil operating below the entering air's dew point does: water vapor condenses out of the airstream onto the coil surface and drains away, which is what actually removes the moisture behind the drop in humidity ratio. §3 covers the sensible/latent split this process represents.
- **Humidification** — a move that adds moisture: nearly straight up (adiabatic/evaporative humidification, moving along a constant-wet-bulb line, since evaporating water into air cools it as it humidifies) or up and to the right (steam/isothermal humidification, adding moisture without a significant temperature change).

Mixing two airstreams — combining, for example, a return air stream and an outdoor air stream — produces a state point on the straight line connecting the two original points, located in proportion to their mass flow rates:

Tmix = (m.1T1 + m.2T2) ÷ (m.1 + m.2), and the same weighted-average form for humidity ratio.

Enthalpy mixes the same way to a very close approximation (both terms are linear in temperature and humidity ratio), which is why a mixed-air enthalpy can be computed either directly from the mixed T and W, or as the mass-weighted average of the two entering enthalpies, with the two paths agreeing. §6 uses this to work a full numeric mixing example in the context of economizer control, where it matters most.

---

3. Sensible and Latent Heat — the split that drives coil behavior

A cooling or heating process changes moist air's dry-bulb temperature (**sensible** heat transfer) and, whenever a cooling coil's surface runs cold enough to fall below the entering air's dew point, also

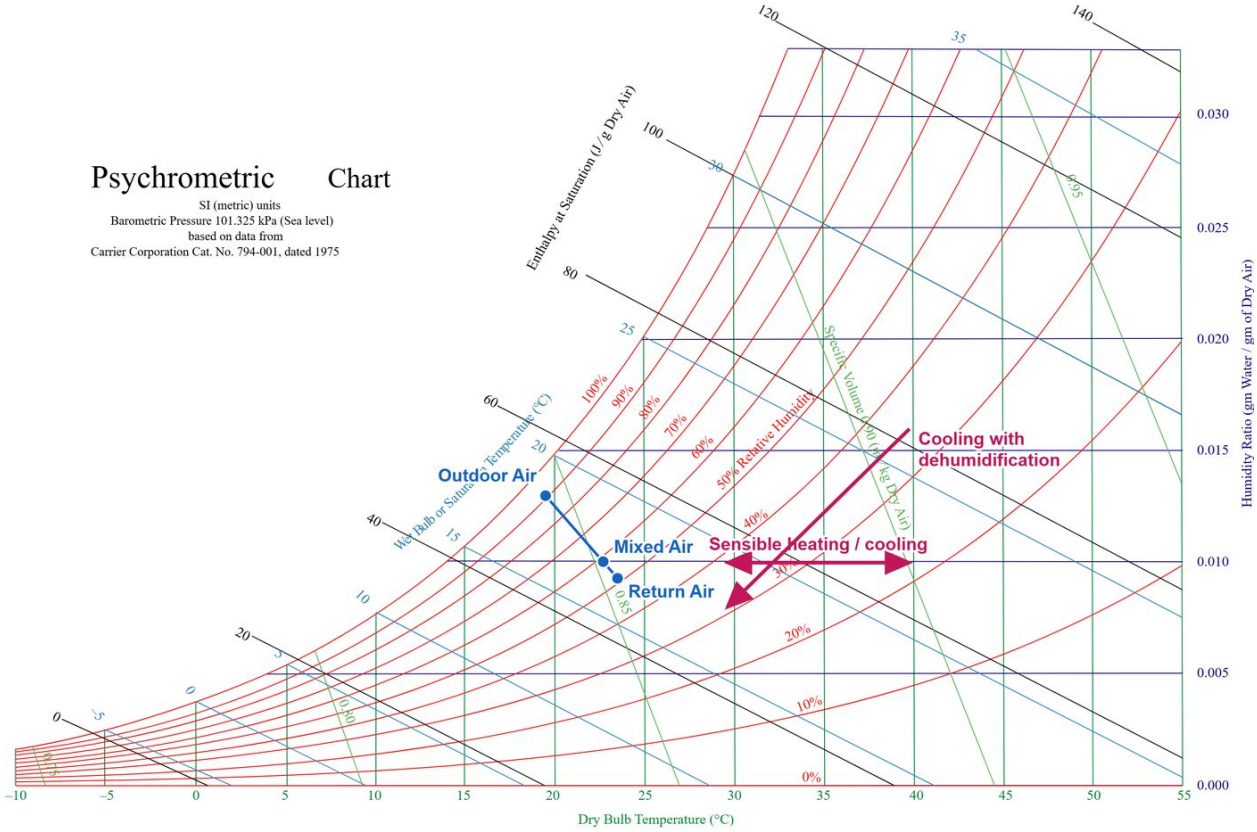


Figure 89: Sensible heating/cooling and cooling-with-dehumidification as chart moves, and two airstreams mixing along the straight line between their state points. Base chart: ArthurOgawa/Wikimedia Commons, CC BY-SA 3.0, annotated for this manual.

removes moisture (**latent** heat transfer). The **Sensible Heat Ratio (SHR)** is the fraction of the total load that is sensible:

SHR = Qsensible ÷ Qtotal = Qsensible ÷ (Qsensible + Qlatent)

A comfort-cooling zone commonly sits around SHR ~ 0.75–0.85; a space with unusually high occupant density or process moisture generation runs lower (more latent-dominated); a space with mostly envelope/solar/lighting load and few occupants runs higher (more sensible-dominated, sometimes approaching 1.0).

3.1 Worked example — from zone loads to required supply air condition

Take a zone with a design sensible cooling load of **10.0 kW** and a design latent cooling load of **2.5 kW** — figures the zone heat balance produces, per *(see "Heating and Cooling Loads: From Design Day to Annual Profile")*, and taken as given here — (SHR = 10.0 ÷ 12.5 = **0.80**), held at a design room condition of **24°C dry-bulb, 50% RH** — a state point with humidity ratio Wroom ~ 0.00930 kg/kg and enthalpy hroom ~ 47.8 kJ/kg (read from a psychrometric chart; this state point recurs in §6).

The sensible heat equation for moist air at standard conditions (derived from mass flow × specific heat × ΔT, using standard air density rho ~ 1.2 kg/m³ and moist-air specific heat cp ~ 1.006 kJ/kg · K):

Qs [W] = V [L/s] × rho × cp × ΔT ~ 1.2 × V [L/s] × ΔT [°C]

(1.2 kg/m³ × 1,006 J/kg · K × 0.001 m³/L ~ 1.21, commonly rounded to 1.2 — the standard SI sensible heat factor. Source: ASHRAE Fundamentals, Psychrometrics chapter.)

Choosing a design supply air temperature difference of **ΔT = 11°C** (a typical VAV cooling design point — this is exactly the quantity EP3's Sizing:Zone asks for as "Zone Cooling Design Supply Air Temperature Difference," §4) gives the required supply airflow:

V = Qs ÷ (1.2 × ΔT) = 10,000 ÷ (1.2 × 11) = 10,000 ÷ 13.2 ~ 758 L/s (~ 0.758 m³/s)

The latent heat equation, similarly derived using the latent heat of vaporization of water hfg ~ 2,501 kJ/kg (the constant term in the moist-air enthalpy relationship used below):

Ql [W] = V [L/s] × rho × hfg × ΔW ~ 3,001 × V [L/s] × ΔW [kg/kg]

(1.2 kg/m³ × 2,501,000 J/kg × 0.001 m³/L ~ 3,001. Source: ASHRAE Fundamentals.)

Solving for the required humidity ratio drop at the airflow found above:

ΔW = Ql ÷ (3,001 × V) = 2,500 ÷ (3,001 × 758) ~ 0.0011 kg/kg

So the required supply air humidity ratio is **Wsupply = Wroom − ΔW ~ 0.0093 − 0.0011 ~ 0.0082 kg/kg** — again, exactly the quantity EP3's Sizing:Zone asks for as "Zone Cooling Design Supply Air Humidity Ratio."

Check, via total heat: the enthalpy drop corresponding to this process is approximately Δh ~ cp × ΔT + hfg × ΔW = 1.006 × 11 + 2,501 × 0.0011 ~ 11.07 + 2.75 ~ 13.8 kJ/kg. The total-heat equation, **Qt [W] ~ 1.2 × V [L/s] × Δh [kJ/kg]**, gives Qt ~ 1.2 × 758 × 13.8 ~ 12,552 W — matching the target 12,500 W (10,000 + 2,500) to within rounding.

These hand constants (1.2, 3,001) assume standard sea-level air density and are for manual sizing checks only — EnergyPlus's own sizing calculations use actual air properties at simulation conditions, not fixed constants. *(see "Zone-Level HVAC Equipment and Zone Sizing")* covers how these two numbers — a design supply air temperature (or ΔT) and a design supply air humidity ratio — actually get entered.

3.2 Rated SHR vs. operating SHR — and why bypass factor mostly stays invisible

The SHR in §3.1 is the zone's *load* SHR — a target the equipment has to meet. A DX cooling coil's own **"Cooling Coil Gross Rated Sensible Heat Ratio"** field (present on EP3's `Coil:Cooling:DX:*` composites) is a different number: the coil's SHR *at a specific rating condition* (a standard entering air temperature and humidity ratio set by the coil's rating standard), used to characterize the coil's physical performance. The coil's actual operating SHR at any other entering condition and airflow shifts away from this rated value — that shift is what the coil's performance curves (*(see "Performance Curves: Part-Load and Efficiency Behavior")*) and its sizing inputs (*(see "Zone-Level HVAC Equipment")* / *(see "Central Plant Equipment")*) are there to capture.

Physically, a real cooling coil doesn't bring 100% of the entering air down to one uniform leaving condition — some fraction of the airstream passes close enough to the fins that it barely contacts a cold surface (approximately unchanged), while the rest approaches the coil's **apparatus dew point (ADP)**, the fully-saturated condition the coil surface itself sits at. The **bypass factor** is the fraction idealized as passing through unchanged; the leaving air condition used in load calculations is a mass-weighted blend of the ADP and the entering condition. EnergyPlus's simple DX coil model derives this behavior internally from the rated SHR curve set rather than asking for a bypass factor or ADP as a direct input — those stay conceptual for ordinary DX and chilled-water coils. The one exception found in EP3 is VRF outdoor units, which do expose bypass factor directly ("Outdoor Unit Evaporator Rated Bypass Factor" / "Outdoor Unit Condenser Rated Bypass Factor"), covered in *(see "Central Plant Equipment")*.

Extending the §3.1 condition line from the room point through the coil leaving (supply) air point out to the saturation curve gives an apparatus dew point of roughly **11°C** for that example — and since the coil leaving point sits close to that ADP, the implied bypass factor (measured along the temperature axis: (Tleaving − TADP) ÷ (Troom − TADP) = (13 − 11) ÷ (24 − 11) ~ 0.15) falls in the typical 0.05–0.20 range for a reasonably selected coil, another internal consistency check on the example.

---

4. From Zone Load to Supply Air Condition — sizing groundwork

§3.1 already showed the mechanism: a zone's sensible and latent design loads, combined with a chosen supply air temperature (or ΔT), determine both the required supply airflow and the required supply air humidity ratio, via the same two constants (1.2 for sensible, 3,001 for latent) derived there. This is the exact relationship EnergyPlus's zone and system sizing calculations run in reverse — starting from design loads and a chosen supply air condition, to determine the equipment capacity and airflow needed.

Where this actually lands in EP3: `Sizing:Zone` takes a **"Zone Cooling Design Supply Air Temperature"** (or **Temperature Difference**) and a **"Zone Cooling Design Supply Air Hu-**

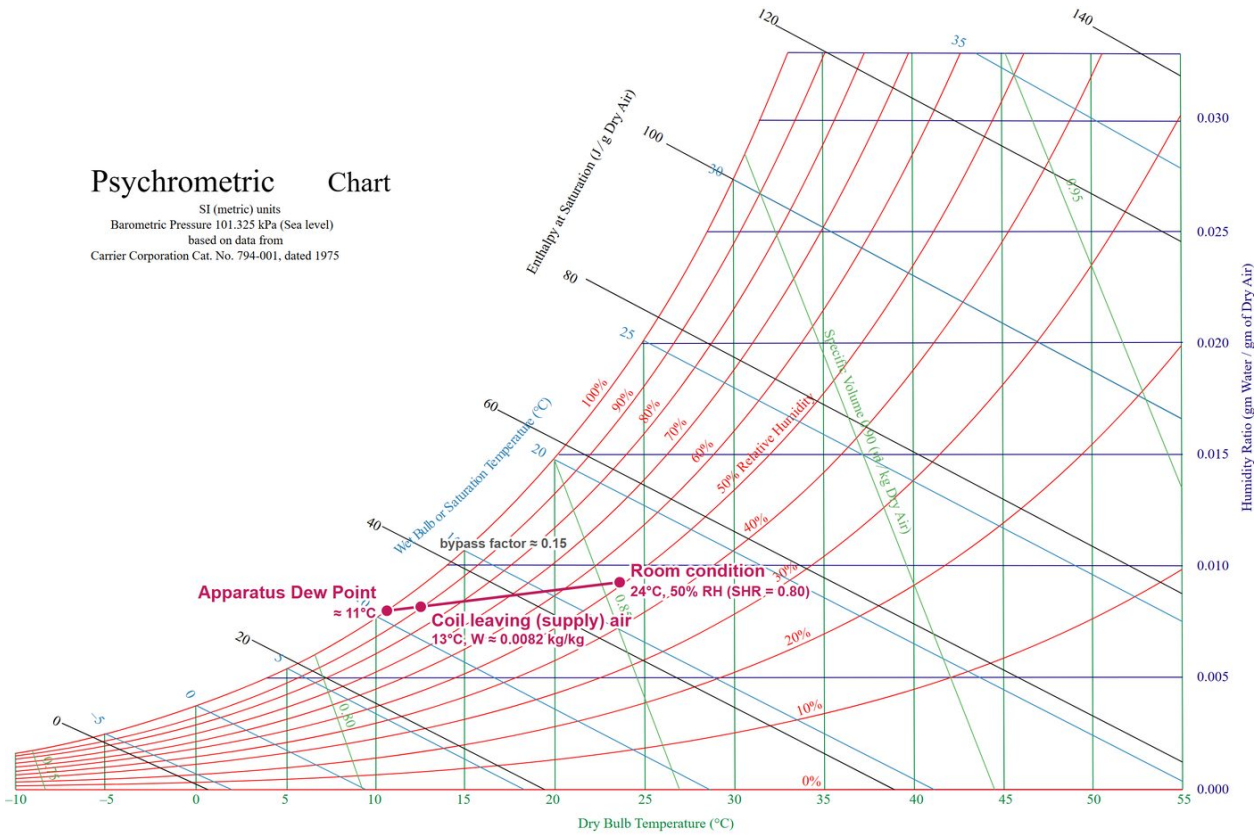


Figure 90: The §3.1 room condition and coil leaving (supply) air point, on the condition line extended to the apparatus dew point on the saturation curve; the coil leaving point sits close to the ADP, consistent with the bypass factor of about 0.13 this construction implies. Base chart: ArthurOgawa/Wikimedia Commons, CC BY-SA 3.0, annotated for this manual.

midity Ratio" (with heating-side equivalents) — the mechanics of that dialog and how it interacts with autosizing belong to *(see "Zone-Level HVAC Equipment and Zone Sizing")*. `Sizing:System` takes the equivalent central-system fields — **"Central Cooling Design Supply Air Temperature"**, **"Central Heating Design Supply Air Temperature"** — plus preheat/precool design temperature and humidity ratio fields for systems that condition outdoor air ahead of the main coil; that object, and how a system-level sizing run reconciles the zones it serves, belong to *(see "Air Loops, Branches, and Nodes")*. This chapter's job is only to establish why those numbers relate to the zone's load the way they do.

---

5. Air-Side System Architecture — a conceptual map

Before meeting the specific objects that build an air-side system in EP3, it helps to have a map of how the pieces fit together in general. Each item below gets its own detailed treatment later — this section exists so that map can be drawn once.

- **Single-zone vs. multi-zone (built-up) systems.** A single-zone system serves one zone from one air handler sized and controlled for that zone alone. A multi-zone (built-up) system shares one central air handler across many zones through a branching duct distribution, with zone-level devices reconciling the difference between what the central system delivers and what each individual zone needs.
- **Constant volume (CV) vs. variable air volume (VAV).** A CV system holds supply airflow fixed and modulates capacity some other way (coil cycling, reheat, multi-speed fan/compressor stages). A VAV system instead throttles supply airflow to each zone to track its load, holding supply air temperature comparatively fixed. VAV's part-load airflow reduction is exactly why the SHR/latent-control issue in §3 gets harder at part load: less airflow moving across the coil at a similar supply temperature shifts the coil's actual operating SHR, and a system whose coil was only sized to handle a design-condition sensible/latent split can under-dehumidify as it throttles down.
- **The mixed-air path.** Return air from the zones and outdoor air brought in for ventilation combine at a mixing point — conceptually the "mixing" process from §2 — before reaching the main coils, with a relief/exhaust path venting the return air not recirculated. An economizer (§6) controls how much outdoor air enters that mix. The objects that build this path — `OutdoorAir:Mixer`, `Controller:OutdoorAir`, and the airloop branch order around them — belong to *(see "Air Loops, Branches, and Nodes")* and *(see "Outdoor Air, Ventilation, and Energy Recovery")*.
- **Dedicated outdoor air systems (DOAS).** Rather than mixing outdoor air into each system's central airstream, a DOAS conditions and delivers ventilation air through a dedicated path, separate from the equipment handling each zone's sensible load. In EP3 this is ordinarily a standalone `AirLoopHVAC` configured for 100% outdoor air rather than a distinct object type — covered in *(see "Air Loops, Branches, and Nodes")* and *(see "Outdoor Air, Ventilation, and Energy Recovery", §5)*.
- **Terminal units.** Where a VAV system's central airflow gets reconciled with each zone's actual demand — VAV boxes, with or without reheat — is airloop-side plumbing, not zone equipment; it belongs alongside branches and nodes in *(see "Air Loops, Branches, and Nodes")* rather than in the zone-level equipment chapter.
- **Central vs. zone-level equipment.** Cooling and heating can happen centrally, on the airloop itself (a chilled-water or DX coil serving the whole system), or locally, in packaged zone

equipment (fan coils, PTACs, VRF terminals) that each condition only their own zone. This split is exactly why *(see "Zone-Level HVAC Equipment")* and *(see "Central Plant Equipment")* are separate chapters — the same sensible/latent and psychrometric relationships from this chapter apply to both, but the objects and controls around them differ.

Figure placeholder — a schematic diagram of a generic multi-zone VAV air handler: outdoor air intake and relief/exhaust meeting return air at a mixing box (with an economizer damper called out), through heating and cooling coils, through a supply fan, out to a duct distribution feeding two or three zones each through a VAV terminal box, with one zone shown served by a separate DOAS ventilation connection.

---

6. Economizers, Conceptually

An economizer uses outdoor air, when conditions favor it, to meet some or all of a system's cooling need without running mechanical cooling — "free cooling." Whether outdoor air actually is favorable depends on what property is being compared, which is why EnergyPlus's `Controller:OutdoorAir` offers several economizer types built on different psychrometric properties (**"Economizer Type"**: DifferentialDryBulb, DifferentialEnthalpy, DifferentialDryBulbAndEnthalpy, FixedDryBulb, FixedEnthalpy, FixedDewPointAndDryBulb, ElectronicEnthalpy, or NoEconomizer), alongside upper/lower temperature and enthalpy limit fields.

- A **dry-bulb economizer** compares outdoor and return air temperature only. Simple, and adequate in dry climates, but it can be fooled by cool, humid air — outdoor air that reads as favorable by temperature but is actually carrying more total heat than the return air once its moisture content is accounted for.
- An **enthalpy economizer** compares total heat content instead, correctly rejecting outdoor air that's cool but humid.
- A **fixed** economizer type compares against an absolute limit rather than the return air condition — useful where a fixed threshold, not a differential, is the design intent.

6.1 Worked example — dry-bulb vs. enthalpy economizer decisions

Using the same room/return air condition as §3.1 — return air at **24°C, W = 0.0093 kg/kg, h ~ 47.8 kJ/kg** (computed from h ~ cpT + W(2,501 + 1.86T) = 1.006(24) + 0.0093(2,501 + 1.86×24) = 24.14 + 0.0093×2,545.6 ~ 47.8 kJ/kg, confirming the chart-read value used earlier) — take an outdoor air condition of **20°C, W = 0.0130 kg/kg** (a cool but humid, overcast day):

hoa = 1.006(20) + 0.0130(2,501 + 1.86×20) = 20.12 + 0.0130×2,538.2 ~ 53.1 kJ/kg

By dry-bulb temperature, outdoor air (20°C) reads as cooler than return air (24°C) — a dry-bulb economizer would call this favorable and open the outdoor air damper toward its maximum. By enthalpy, outdoor air (53.1 kJ/kg) is actually carrying *more* total heat than return air (47.8 kJ/kg) — an enthalpy economizer would correctly reject it, since bringing in more of this air increases the coil's total load even though it feels cooler. This is the standard justification for requiring enthalpy-based economizer control in humid climates.

Mixed air condition, at 20% outdoor air / 80% return air (a typical minimum-outdoor-air fraction, using the mixing relationship from §2):

Tmix = 0.2(20) + 0.8(24) = 4.0 + 19.2 = 23.2°C

Wmix = 0.2(0.0130) + 0.8(0.0093) = 0.0026 + 0.00744 ~ 0.0100 kg/kg

Check, two paths to the same mixed enthalpy: directly from the mixed condition, hmix = 1.006(23.2) + 0.0100(2,501 + 1.86×23.2) ~ 23.34 + 0.0100×2,544.2 ~ 48.9 kJ/kg. As a mass-weighted average of the two entering enthalpies instead: hmix = 0.2(53.1) + 0.8(47.8) = 10.62 + 38.24 = 48.9 kJ/kg — the two paths agree, confirming enthalpy mixes linearly to the precision needed here.

The mechanics of setting economizer limits, high-limit lockout, and how this interacts with freeze protection on the mixed-air stream belong to *(see "Outdoor Air, Ventilation, and Energy Recovery")* — this section's job was only to make the physical reasoning behind the "Economizer Type" choice checkable by hand.

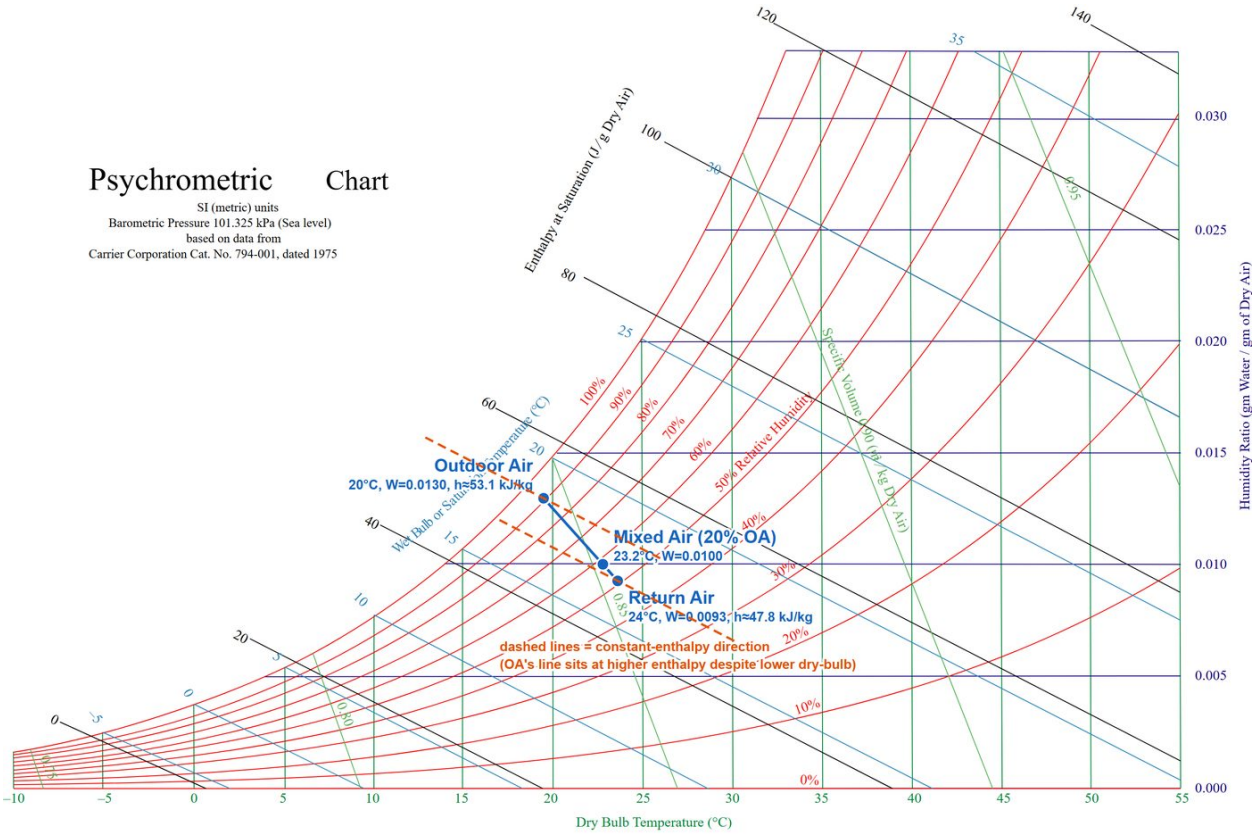


Figure 91: The §6.1 return air, outdoor air, and mixed air points, with constant-enthalpy reference lines through the return-air and outdoor-air points showing the outdoor air's higher total heat content despite its lower dry-bulb temperature. Base chart: ArthurOgawa/Wikimedia Commons, CC BY-SA 3.0, annotated for this manual.

---

7. Vocabulary recap

Term	What it means	Where it becomes an EP3 field
Dry-bulb temperature	Ordinary air temperature	Supply air temperature fields throughout Sizing:Zone/Sizing:System (§4)
Humidity ratio (W)	Mass of water vapor per mass of dry air	Supply air humidity ratio fields in Sizing:Zone/Sizing:System (§4); preheat/precool humidity ratio fields
Relative humidity (RH)	Vapor pressure ratio to saturation, at a given temperature	ZoneControl:Humidistat setpoint schedules
Wet-bulb temperature	Temperature reached by adiabatic saturation	Evaporative cooler effectiveness fields; design-day weather data
Dew point	Saturation temperature at constant humidity ratio	Fixed-dew-point economizer limit field (§6)
Specific enthalpy (h)	Total (sensible + latent) heat content per unit mass of dry air	Enthalpy-based economizer type and limit fields (§6)
Sensible Heat Ratio (SHR)	Sensible ÷ total heat	DX coil "…Gross Rated Sensible Heat Ratio" fields (§3.2)
Bypass factor / apparatus dew point	Coil-model idealization of unconditioned bypass air blended with saturated coil-surface air	Exposed directly only for VRF outdoor units (§3.2); otherwise internal to the coil model
Economizer	Using favorable outdoor air instead of mechanical cooling	Controller:OutdoorAir "Economizer Type" and limit fields (§6)
DOAS	Ventilation air conditioned/delivered separately from zone sensible conditioning	`AirLoopHVAC:DedicatedOutdoorAirSystem` (§5)

End of draft chapter. Outstanding items for the next pass: resolve the XREF placeholders once their target chapters exist; confirm the psychrometric chart-read values in §3.1/§6.1 against a calculator or chart rather than hand-derived approximations, if higher precision is wanted.

13. The HVAC Outliner and Creating HVAC Systems

1. Overview

This chapter has two jobs: it's a tour of the **HVAC Outliner**, the dialog you use to see and navigate any HVAC system once it exists in your model, and it's a guide to the options for creating HVAC systems within EP3 — picking a complete prototype system as a template, or assembling one from individual components.

What this chapter does not cover. It doesn't teach airloop/branch topology in the mechanical detail needed to build one from scratch — that's *(see "Air Loops, Branches, and Nodes")*. It doesn't cover the details of any particular piece of equipment — zone-level equipment is *(see "Zone-Level HVAC Equipment")*, central plant equipment is *(see "Central Plant Equipment")*, and setpoint-manager control strategy is covered in *(see "Air Loops, Branches, and Nodes")* and *(see "Plant Loops and Setpoint Control")*. And it doesn't re-explain the Model Wizard's HVAC Systems tab or the post-Wizard zone-assignment dialog — both already covered in *(see "The Model Wizard", §6, §9)* — beyond what's needed to connect that tab's choices to what you'll actually see afterward, in this chapter's Outliner.

---

2. The HVAC Outliner

 The **View HVAC + Outliner** toolbar icon opens the dialog titled **EP3 - HVAC Outliner** and, at the same time, makes the model's HVAC group active in the SketchUp viewport.

2.1 Tree structure

The Outliner lists every HVAC object in the model, organized into sections. Two EnergyPlus concepts run through this tree and the rest of Part IV: a **node** is a single point in the system where EnergyPlus tracks air or water conditions — temperature, flow rate, humidity — as it moves between components, and a **branch** is an ordered sequence of one or more components connected in series between two nodes. A loop's supply and demand sides are each built from one or more branches — component after component, inlet node to outlet node — joined at splitters and mixers where a loop divides into parallel branches and rejoins. Branch covers the object, and the node/branch/connector system it belongs to, in EnergyPlus's own terms; *(see "Air Loops, Branches, and Nodes")* covers EP3's implementation, with a focus on Air Loops; *(see "Plant Loops and Setpoint Control")* covers the specifics of plant-side branches and connections.

- **Air Loops** and **Plant Loops** — one entry per `AirLoopHVAC` or `PlantLoop` object, each expandable into:
  - **Setpoint Managers** — every setpoint manager connected anywhere on that loop, listed by name.
  - **Supply Side** and **Demand Side** — each broken into its inlet branch, any parallel branches, and its outlet branch, in that order. Expanding a branch lists the components on it (fans, coils, pumps, and so on) by name.
- **VRV** — one entry per `AirConditioner:VariableRefrigerantFlow` condensing unit (§5). VRF systems don't sit on an `AirLoopHVAC` or `PlantLoop`'s branch structure at all, which is why they get their own section rather than appearing under Air Loops or Plant Loops.

- **ZoneHVAC** — a flat list of zone-level equipment (fan coils, PTAC/PTHP, VRF terminal units, baseboards, water-to-air heat pumps, and similar) that lives in the model's HVAC systems. ZoneHVAC objects may appear also appear on a branch. §4 covers the behavior of this equipment family.
- **WaterHeaters** — every `WaterHeater:*` object.
- **Unconnected Objects** — anything else in the HVAC group that doesn't fall into one of the sections above, typically a component you've dragged in (§6) but haven't yet wired into a loop.
- **Outdoor Air Nodes** — every `OutdoorAir:Node`.

Hovering over a loop, a branch, or an individual component highlights the corresponding geometry in the diagram, so you can trace what a given tree entry actually is without leaving the Outliner.

2.2 Toolbar actions

Across the top of the Outliner:

- **Consolidate** — for `ZoneHVAC` equipment, `AirLoopHVAC`s, or both (your choice), finds separate equipment instances that are functionally identical except for which zone each one serves, and merges them back down into a single shared object serving all of those zones. Useful after assigning the same equipment to many zones individually (§3) leaves the model with more separate objects than necessary.
- **Add Air Loop** / **Add Plant Loop** — creates a new, empty `AirLoopHVAC` or `PlantLoop` group.
- **Add Component** — opens the Components dialog (§6) for dragging an individual piece of equipment into the model.
- **View Zone Assignment** — opens the same **EP3 - Select Zone Conditioning Equipment** dialog the Wizard opens after its own HVAC selections. *(see "The Model Wizard", §9)*
- **Condense** — spatially reorganizes the HVAC group's components in the 3D view, without changing any connections.
- **Refresh** — re-catalogs and re-traces the HVAC group's contents, for when the tree falls out of sync with the model.
- **Expand All** / **Collapse All**

Condense, Refresh, Expand All, and Collapse All have no dedicated icon file in EP3 itself — the images above reproduce their actual in-app button glyphs and coloring at legible size.

2.3 The branch-connection tools

Two further toolbar actions push a 3D-viewport tool rather than opening a dialog:

- **Erase Branch** — the reverse of Draw Branch: click a connection to remove it.
- **Draw Branch** — activates a click-to-connect tool. Eligible connection points appear in the 3D view as small colored markers, colored by the fluid they carry (air, chilled water, hot water, condenser water, steam, and so on); click a first node, then a second, to connect them. *(see "Air Loops, Branches, and Nodes")* covers how branches, nodes, and this connection process fit into an `AirLoopHVAC`'s structure; *(see "Plant Loops and Setpoint Control")* covers the same for a `PlantLoop`.

Shift-clicking a node already on a branch, as your first pick, switches Draw Branch

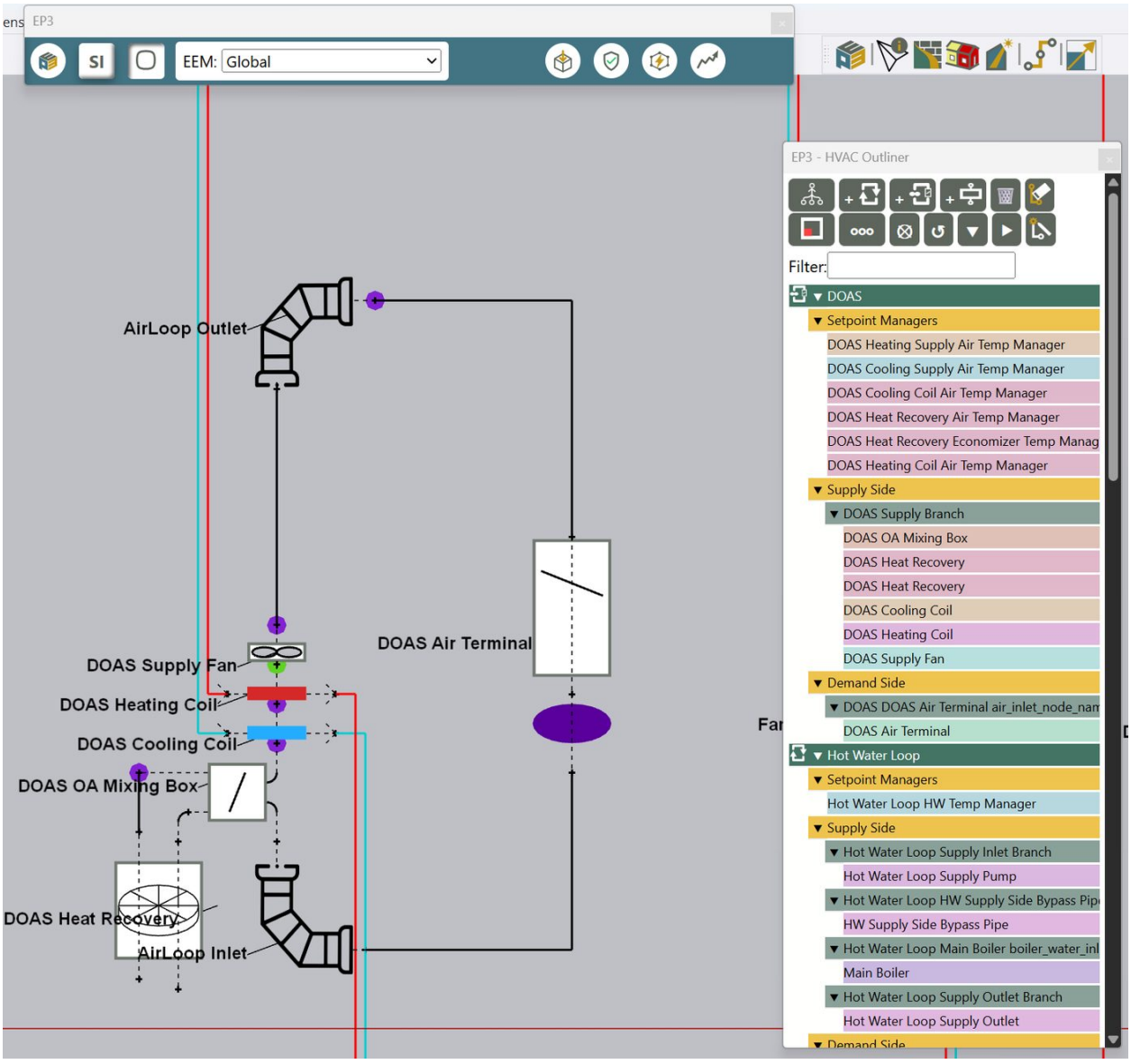


Figure 92: The EP3 - HVAC Outliner, with a DOAS air loop expanded down through Setpoint Managers and Supply Side (its branch expanded to show the OA mixing box, heat recovery, cooling coil, heating coil, and supply fan) and Demand Side, alongside a Hot Water Loop entry below it.

into Add Mode. Instead of connecting to something that already exists, it opens a **"Select Component to Connect"** picker, listing only the object types valid at that node. Choosing a type (and, if that type has more than one connectable nodes, which node to attach) creates a new instance of it and splices it directly onto the branch at that point in a single step, without a separate trip through the Components dialog first. A plain (non-shift) first click only works from an eligible node — either one already on a branch or one from a component already sitting unconnected in the model (§2.1's Unconnected Objects) — Draw Branch doesn't create new objects on its own.

---

3. Creating an HVAC System from an EP3 Template

The library of complete prototype HVAC systems — `default_hvac_systems` *(see "The Model Wizard", §6)* — is the normal way most systems get into a model. To recap: the Wizard's HVAC Systems tab lists complete systems. EP3's built-in library includes standard system types (PSZ-AC, PSZ-HP, PTAC/PTHP, VRV, Fan Coil, Packaged VAV, DOAS, and others, plus the `ashrae_baseline` subfolder). The user-library contains user-created HVAC system libraries — §7 covers how to build one of your own. Each HVAC system template comes complete and ready to simulate with every object, curve, and schedule needed.

Once imported, the template HVAC system is shown in the diagram and in the HVAC outliner in full detail. A template built around an `AirLoopHVAC` or `PlantLoop` appears under Air Loops/Plant Loops with its branches and setpoint managers already wired; a template built from packaged zone equipment appears under ZoneHVAC instead. Zone assignment — connecting the equipment a template brought in to the actual zones it should serve — happens in the **Select Zone Conditioning Equipment** dialog, reachable either from the Wizard directly after import or from the Outliner's **View Zone Assignment** button (§2.2) at any later point. *(see "The Model Wizard", §9)*

---

4. Choosing a System Type: ZoneHVAC vs. AirLoopHVAC

Every system in the Wizard's library, and every piece of equipment in the Components dialog (§6), belongs to one of two families, and the Outliner's own section structure (§2.1) reflects the split directly: packaged zone equipment that lists flatly under **ZoneHVAC**, or airloop-based equipment that builds out the **Supply Side**/**Demand Side** branch structure under **Air Loops**.

ZoneHVAC equipment — PTAC, PTHP, fan coils, water-to-air heat pumps, VRF terminal units, and similar — condition one zone at a time. These systems simulate faster than equivalent airloop-based systems, since there's no central mixed-air path or branch network for EnergyPlus to solve. In exchange, you give up direct control over each unit's fan, coil, and flow-rate operation: a ZoneHVAC object runs its own built-in control scheme, usually with little room to adjust it. Outdoor air is the sharpest example — most ZoneHVAC equipment supplies its design-level outdoor airflow whenever the fan runs, with no way to cut it back at night or drive it with demand-controlled ventilation. And because there's no shared, centrally-managed airstream to act on, options that depend on one — an economizer, ventilation coordinated across zones, heat recovery on a common exhaust stream — aren't available.

AirLoopHVAC-based equipment — packaged rooftop units, built-up VAV systems, DOAS — routes zones through a shared central airstream, mixing return and outdoor air at a controlled

point before it reaches the main coils. That central mixed-air path is exactly what makes economizers, coordinated ventilation control, and central heat recovery possible (*(see "Air-Side System Fundamentals", §5)* covers the mixed-air path conceptually). The cost is simulation overhead and modeling complexity: more nodes, more branches, more control objects for EnergyPlus to solve.

Neither family is categorically better — the choice tracks what the building actually needs. A building with modest, zone-independent loads and no meaningful economizer opportunity can often be accurately represened by ZoneHVAC equipment; a building where ventilation-driven energy use or economizer savings matter needs the central airflow control that only an airloop-based system provides. Among the Wizard's own library, PTAC/PTHP/Fan Coil/VRV are ZoneHVAC systems; the Packaged VAV variants, and DOAS are AirLoopHVAC-based — so the tab you'd check in the Wizard already reflects this choice.

---

5. VRF Systems

A VRF system is built around one or more `AirConditioner:VariableRefrigerantFlow` condensing units, each serving a set of `ZoneHVAC:TerminalUnit:VariableRefrigerantFlow` (or, if terminal units ride an airloop branch instead, `AirLoopHVAC:TerminalUnit:VariableRefrigerantFlow`) terminal units. Like other systems, VRF is available both as a complete Wizard template (**"VRV"**, under `default_hvac_systems`) and as individual pieces in the Components dialog's VRV/AirLoop/ZoneHVAC folders (§6) — but either way, it needs one extra step no other system type requires.

Connecting terminal units to a condensing unit is a separate, explicit action. A VRF condensing unit's terminal units aren't inferred from geometry or from zone assignment — they're maintained as their own list, connected through a drag-and-drop dialog reached by selecting the condensing unit and clicking **Connect Zone Terminals to VRVs**. Importing the Wizard's VRV template wires this automatically for the zones the template covered; adding terminal units afterward — by hand through the Components dialog, or by adding zones later — does not, and needs this same drag-list step run again. Leaving a terminal unit unconnected doesn't fail silently: it's flagged as a validation error (*(see "Model Troubleshooting and Quality Assurance")*), since EnergyPlus itself errors on a VRF terminal unit with no condensing unit serving it.

Because a VRF condensing unit isn't a branch on an `AirLoopHVAC` or `PlantLoop`, it gets its own **VRV** section in the Outliner (§2.1) rather than appearing nested under Air Loops or Plant Loops — its refrigerant-side connections are represented through the terminal-unit list, not through the branch/node structure §2.3's Draw Branch tool operates on. AirConditioner:VariableRefrigerantFlow covers the object's full field set; the engineering detail behind VRF's bypass-factor-based coil model belongs to *(see "Central Plant Equipment")*.

---

6. Building a System from Individual Components

Rather than importing a complete template, you can assemble a system piece by piece from the **Components** dialog — reached from the Outliner's **Add Component** button (§2.2), or titled **EP3 - HVAC Components** directly. It's organized into folders by system area — Air Distribution, AirLoop, Plant, Service Hot Water, VRV, ZoneHVAC — each listing the individual composite objects available in that area (fans, coils, terminal units, pumps, chillers, boilers, cooling

towers, and so on). Selecting one and confirming drops it into the model's HVAC group, ready to place and connect.

This chapter's job ends at knowing this path exists and where to find it — actually wiring components together into a working airloop or plant loop, in the topology and node-connection detail that takes, is *(see "Air Loops, Branches, and Nodes")*'s job for an `AirLoopHVAC` and *(see "Plant Loops and Setpoint Control")*'s for a `PlantLoop`; the individual objects themselves are covered field-by-field in *(see "Zone-Level HVAC Equipment")* and *(see "Central Plant Equipment")*.

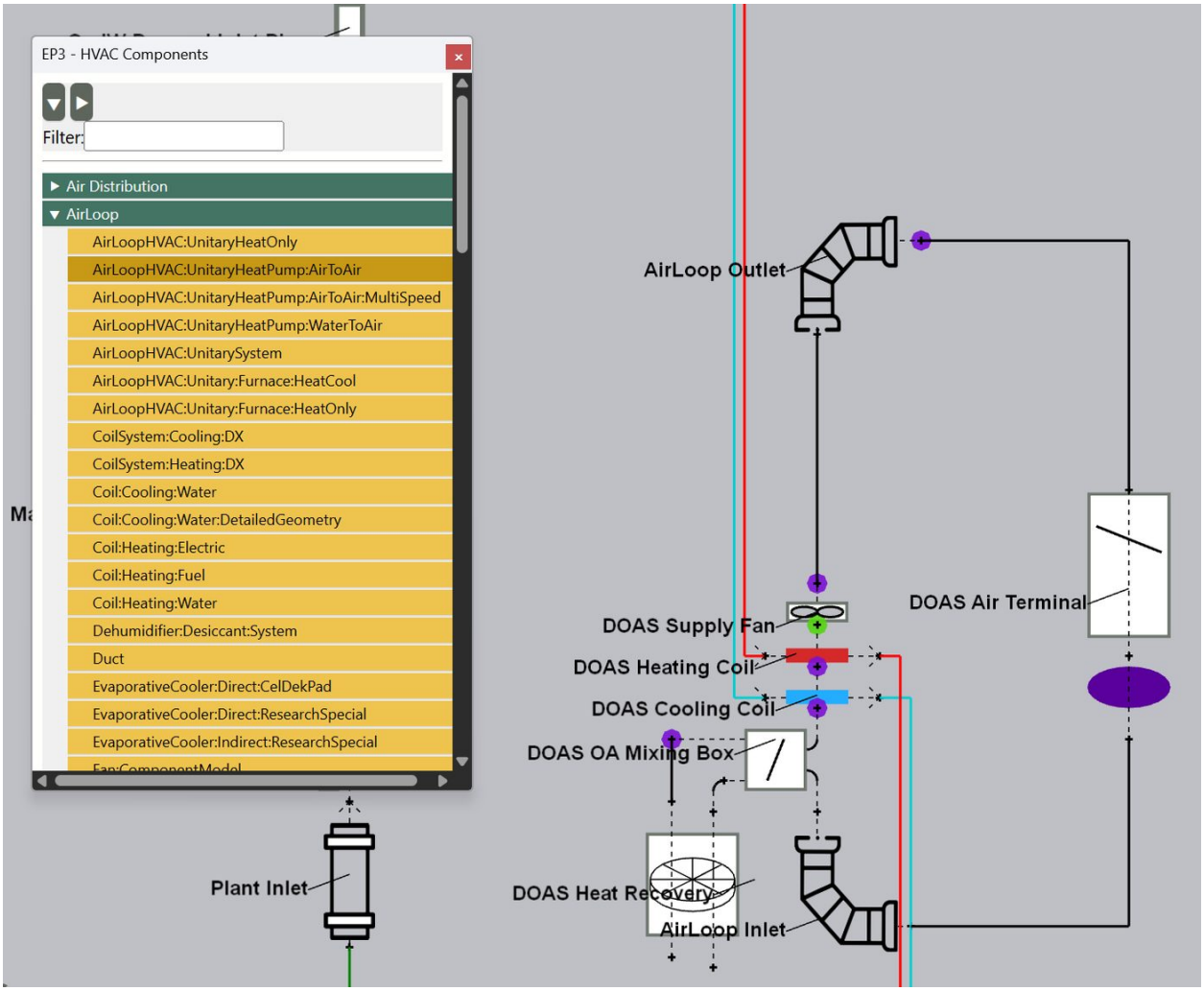


Figure 93: The EP3 - HVAC Components dialog with the AirLoop folder expanded, listing individual composite objects like AirLoopHVAC:UnitaryHeatOnly, AirLoopHVAC:UnitaryHeatPump:AirToAir, and Coil:Cooling:Water, available to drag into the model.

---

7. Building Your Own Reusable EP3 Template

A system you've built — whether by customizing a Wizard template, or by assembling it from individual components (§6) — can be saved as a reusable template the same as a construction set or a `SpaceType`/`ZoneType`. *(see "Constructions and Materials", §8)*, *(see "Internal Loads, Space*

Types, and Zone Types", §7)

7.1 Recommended workflow

Building an HVAC template reliably follows the same four steps regardless of how the system itself was built:

1. **Create the system** — from a Wizard template you've customized, or assembled by hand from components (§6).
2. **Test the system** — simulate it in the model you built it in, and confirm the results are what you expect. The EP3 team builds templates in a simple 5-zone core & shell model (usually a wizard generated model). The fast simulation times allow the team to fully test the system, confirm expected operation and make sure that unmet hours fall within acceptable ranges.
3. **Create the template** — save it, following §7.2 below.
4. **Test the template** — import it into a new project, usually a simple Wizard-built model, and simulate again.

Construction and internal-loads templates carry a handful of object types each, simple enough that step 2's simulation is normally assurance enough on its own. An HVAC system template is a different scale of problem: EnergyPlus's HVAC object family runs to hundreds of object types with an open-ended range of valid connections between them, and EP3's template save/reuse mechanism (§7.2) has to correctly capture and rebuild all of it. Step 4 exists because step 2 only confirms the system works in the model you built it in — it doesn't confirm that everything the system needs got captured when it was saved, or that it reconstructs correctly on import.

If a system simulates correctly in step 2 but the reimported template doesn't reproduce the same result in step 4, report it — that gap points to something EP3's template mechanism missed. Please contact the EP3 team for assistance - we want to know about any gaps so we can fill them!

7.2 Saving and reusing a template

Saving a template. While the HVAC view is active (§2), select the objects you want to save (double check that everything is selected - often loops themselves are missed when selecting), right-click, and choose **"EP3 - Create HVAC Template from selected objects."** This gathers the selection along with everything it references and opens a dialog showing the generated IDF text, a notes field, and a **"Save as HVAC template"** button. Confirming opens a standard save dialog, defaulting to your configured HVAC template library folder (a warning appears if you save somewhere else); the saved file carries your notes and is written in a format valid across every EnergyPlus version EP3 supports.

Reusing a saved template. A template saved this way reappears the next time you open the Wizard's HVAC Systems tab, or its reduced-mode "Select Templates / Objects to import" reuse path, listed under **User Library** alongside EP3's own built-in systems under **EP3 Library** — (covered in §3).

---

8. An Alternate Path: EnergyPlus's Own HVAC Templates

EnergyPlus provides its own simplified template objects — `HVACTemplate:Zone:*` (for example `HVACTemplate:Zone:VAV`, `HVACTemplate:Zone:FanCoil`, `HVACTemplate:Zone:PTAC`), `HVACTemplate:System:*`, `HVACTemplate:Plant:*`, and `HVACTemplate:Thermostat` — as a shorthand way of specifying an HVAC system, meant to be expanded into full EnergyPlus objects before simulation. HVACTemplate covers the full object family.

EP3 exposes these objects for editing, but they are a genuinely separate, standalone path from everything else in this chapter. Find them in the Main Dialog's Column 1, under the **HVAC Templates** group — since a new model has no instances of any of them, you'll need to enable **Show Empty Classes** to see the group at all the first time. *(see "The EP3 Interface and Model Settings", §2.2)* Unlike every other object this chapter covers, an `HVACTemplate:*` object you create here has no SketchUp entity behind it: it never appears in the HVAC Outliner (§2), in any of its Air Loop/Plant Loop/ZoneHVAC/VRV sections, and it can't be connected to anything with the Draw Branch tool (§2.3). It exists only as a set of fields, editable in Column 3 like any other object, until EnergyPlus itself expands it at simulation time.

That standalone nature is exactly what makes this path useful for a user already comfortable working directly in EnergyPlus's own template objects: build a system this way, let EnergyPlus expand it during a simulation, then bring the *expanded*, fully-realized objects back into EP3 — at which point they behave exactly like anything else in this chapter, including being savable as a normal EP3 template (§7).

8.1 Workflow: building an EP3 template from EnergyPlus's HVAC templates

1. **Build a simple test model with no HVAC system.** Run the Wizard as usual — geometry, internal loads, constructions — but leave the HVAC Systems tab empty. *(see "The Model Wizard", §6)*
2. **Build the system using EnergyPlus's own template objects.** In the Main Dialog, find the **HVAC Templates** group in Column 1 (enable Show Empty Classes if it isn't visible), and create the `HVACTemplate:Zone:*`, `HVACTemplate:System:*`, `HVACTemplate:Plant:*`, and `HVACTemplate:Thermostat` objects your system needs, filling in their fields in Column 3.
3. **Simulate the file.** *(see "Running Simulations")* EnergyPlus's own working files for the run, including the expanded IDF it generates internally from your `HVACTemplate:*` objects, are cleaned up once the run completes — but the exported `.idf` EP3 handed to EnergyPlus is not; it persists in the model's results folder alongside the run's other output.
4. **Expand the template objects using EP-Launch.** Open that exported `.idf` in **EP-Launch** (installed alongside EnergyPlus, outside EP3) and run it from there, pointed at the same weather file. EP-Launch runs the **ExpandObjects** preprocessor automatically and saves the result as `<filename>.expidf` next to its other outputs — a complete IDF with your `HVACTemplate:*` objects replaced by the real, non-template HVAC objects EnergyPlus generated from them. Auxiliary EnergyPlus Programs: ExpandObjects
5. **Start a NEW EP3 project and import the expanded file.** Open a fresh instance of Sketchup with a blank model. Click on the EP3 logo to start the wizard, but close the wizard (X). Then use the Importer's general file-import path to bring in the `.expidf` file (you can accept the defaults on import). *(see "Import, Export, and Interoperability")* This builds ordinary EP3 HVAC objects from what EnergyPlus expanded your template objects into — these now appear normally in the HVAC Outliner (§2), on real branches, connectable like any

other component.

6. **Save the result as an EP3 template.** With real composite objects now in the model, select them and follow §7 to save them as a reusable EP3 HVAC template, for normal reuse through the Wizard from then on.
7. **Test** Don't forget to test your saved template in a simple, wizard model.

14. Zone-Level HVAC Equipment and Zone Sizing

1. Overview

This chapter covers zone conditioning equipment (**ZoneHVAC** objects) *(see "The HVAC Outliner and Creating HVAC Systems", §2.1)*. This is equipment that is assigned to each zone, and conditions one zone at a time, rather than routing zones through a shared central airstream. This chapter also covers how `ZoneHVAC` objects are sized — and the object that governs the sizing: `Sizing:Zone`.

This chapter assumes you already know how to get `ZoneHVAC` equipment into a model (the Components dialog, or an imported template — *(see "The HVAC Outliner and Creating HVAC Systems", §6)*) and why you'd choose `ZoneHVAC` equipment over an `AirLoopHVAC`-based system in the first place (*(see "The HVAC Outliner and Creating HVAC Systems", §4)*). It also assumes the VRF condensing-unit-to-terminal-unit connection mechanism, which is Ch.13's job (*(see "The HVAC Outliner and Creating HVAC Systems", §5)*).

This is not a field-by-field reference for any of these objects — refer to the the EnergyPlus Input/Output Reference for field-by-field descriptions. This chapter covers: the equipment mechanisms EP3 makes available and how to tell them apart, how equipment gets assigned to a zone once it exists, what `Sizing:Zone` is and why it's required, and how a zone's sizing assumptions cascade into equipment, system, and plant sizing, and where a mismatch in those assumptions shows up as unmet hours or unexpected operation.

Also out of scope here: air terminal units (VAV/CV boxes) and `AirTerminal:SingleDuct:Mixer` participate in the same equipment-list and load-distribution machinery this chapter describes, but they are covered in *(see "Air Loops, Branches, and Nodes")* because they are connected to AirLoops.

---

2. Zone-Level Equipment, by Mechanism

Many, but not all, ZoneHVAC objects in EP3 are built as **composite objects**: a parent object with a handful of interchangeable sub-component slots — typically a fan, one or two coils, and an outdoor-air mixer — each exposed as an Object Type/Name field pair on the parent (the same pattern the Fans chapter describes in full — *(see "Fans and Fan Power", §3.2)*). PTAC, PTHP, Four-Pipe Fan Coil, the VRF terminal unit, and several others are composites in this sense. Others — Ideal Loads Air System, the Baseboard family, `Fan:ZoneExhaust`, and most of §2.6 — are plain objects instead: no sub-components, no interchangeable coil or fan slots, just one object acting on the zone directly. Where an object is a composite, what distinguishes it from another composite is both which sub-component types its slots accept *and* how the wrapper object controls those sub-components as a package — staging, capacity control method, cycling versus continuous operation, outdoor air supply and controls. Two composites can reference the exact same coil and fan types and still behave completely differently, because it's the wrapper's control logic that governs how the package operates.

2.1 Self-contained packaged equipment

`ZoneHVAC:PackagedTerminalAirConditioner`, `ZoneHVAC:PackagedTerminalHeatPump`, and `ZoneHVAC:WindowAirConditioner` are composite objects built around a self-contained DX cooling coil — cooling capacity is entirely local to the unit, with no central plant or condensing unit

needed. Heating is a separate choice, and not automatically self-contained: PTAC's heating-coil slot accepts a fuel, electric, hot-water, or steam coil, and PTHP's DX heating coil (the same refrigerant loop reversed) can be backed by an optional supplementary coil — fuel, electric, hot-water, or steam — to cover capacity below the heat pump's balance point. Choosing a hot-water or steam coil for either object's heating side pulls in the same `PlantLoop` dependency §2.2 describes; choosing fuel or electric heating keeps the whole unit self-contained. Window AC has no heating function at all.

2.2 Equipment served by a central plant, or its electric-resistance equivalent

`ZoneHVAC:FourPipeFanCoil`, `ZoneHVAC:LowTemperatureRadiant:VariableFlow`, `ZoneHVAC:WaterToAirHeatPu` the `ZoneHVAC:Baseboard:*` family (radiant/convective, water/steam/electric), and `ZoneHVAC:LowTemperatureRa` all draw their heating and/or cooling capacity from somewhere other than themselves — but they aren't all composites. Four-Pipe Fan Coil, Low-Temp Radiant Variable Flow, and Water-to-Air Heat Pump are composite objects, with a coil (and, for the fan coil and heat pump, fan) sub-component slot. The Baseboard family and Low-Temp Radiant Constant Flow/Electric are plain objects — capacity and flow rate are entered directly on the object itself, with no swappable coil, though the water/steam Baseboard variants do carry a separate design/sizing sub-object alongside their main fields.

Four-Pipe Fan Coil and the water/steam baseboard and radiant variants need a `PlantLoop` supplying hot and/or chilled water — Four-Pipe Fan Coil's coil slots only accept hydronic coils (`Coil:Cooling:Water`/`Coil:Cooling:Water:DetailedGeometry` and `Coil:Heating:Water`/`Coil:Heating:Elec` no DX option). Water-to-Air Heat Pump draws instead from a shared condenser-water loop rather than a hot/chilled-water loop — a water-loop heat-pump (WLHP) configuration. Every plant-connected object here has an electric-resistance sibling (Baseboard:Electric, LowTemperatureRadiant:Electric) that needs no loop at all, for the same physical equipment type without a central water system behind it. None of these objects will simulate correctly without whatever loop they depend on already built and connected — *(see "Plant Loops and Setpoint Control")* and *(see "Central Plant Equipment")* cover that side.

2.3 VRF terminal units

`ZoneHVAC:TerminalUnit:VariableRefrigerantFlow` is a composite object that draws its capacity from a central `AirConditioner:VariableRefrigerantFlow` condensing unit over refrigerant piping rather than an airside or waterside loop; connecting a terminal unit to its condensing unit is a separate, explicit step covered in full at *(see "The HVAC Outliner and Creating HVAC Systems", §5)*. The terminal unit's coil slots — VRF-specific DX cooling and heating coils, plus an optional supplementary heating coil for capacity below the system's low-ambient limit — carry the terminal-level rated flow and capacity fractions; the condensing unit itself is where overall system capacity is set.

2.4 Zone-level dedicated ventilation equipment

`ZoneHVAC:EnergyRecoveryVentilator` and `ZoneHVAC:UnitVentilator` are both composite objects that manage outdoor air intake directly, rather than depending on a central airloop's mixed-air path the way the equipment above typically does. The ERV pairs a heat-exchanger sub-object with a supply/exhaust fan pair to recover energy from exhaust air; Unit Ventilator exposes minimum/maximum outdoor-air-fraction and OA-control-schedule fields directly, since — unlike the

packaged and plant-connected equipment above — modulating outdoor air intake in response to conditions is part of what this object is for.

2.5 Ideal Loads Air System

`ZoneHVAC:IdealLoadsAirSystem` is a plain object, not a composite — it has no real fan or coil subcomponents at all. It idealizes a zone's HVAC entirely: EnergyPlus supplies exactly the airflow, temperature, and (optionally) humidity needed to hold the zone at setpoint, with outdoor air, economizer, and heat recovery behavior available as simple fields on the object rather than modeled hardware. Use it to isolate the heating/cooling impact of an architectural or envelope decision before any real HVAC system has been designed — it produces a complete, runnable simulation without committing to equipment selection. It carries the same `Sizing:Zone` requirement as every other piece of zone equipment (§4.1) — an idealized system still needs somewhere to draw its design-day sizing assumptions from.

2.6 Other zone-level objects

A handful of other zone-level objects exist for narrower cases and aren't detailed further here: `ZoneHVAC:Dehumidifier:DX` (standalone DX dehumidification), `ZoneHVAC:VentilatedSlab`, `ZoneHVAC:HighTemperatureRadiant`, `ZoneHVAC:ForcedAir:UserDefined` (an EMS-driven custom equipment shell), `ZoneHVAC:RefrigerationChillerSet`, and `Fan:ZoneExhaust` (a standalone exhaust fan not part of any supply-air path — placing one triggers a one-time, model-wide setup choice for how its exhausted air gets made up; *(see "Infiltration, Natural Ventilation and Airflow Networks", §2)* covers it in full). None of these are composite objects — each is available from the Components dialog's ZoneHVAC folder as a single plain object; consult the EnergyPlus Input/Output Reference for their field sets. ZoneHVAC:Dehumidifier:DX ZoneHVAC:VentilatedSlab ZoneHVAC:HighTemperatureRadiant

---

3. Assigning Equipment to a Zone

Placing a piece of equipment from the Components dialog, or importing a template that includes one, adds it to the model and to the HVAC systems diagram, but does not connect it to any zones. A zone's equipment set, and the order in which multiple pieces of equipment share a zone's load, are configured separately, in the **"Assign HVAC Systems by Template"** / **"Assign HVAC Systems by Zone"** dialog — reached from the Outliner's **View Zone Assignment** button (*(see "The HVAC Outliner and Creating HVAC Systems", §2.2)*).

3.1 Ordering Equipment by Load-Following Ability

Order equipment from least able to meet the zone's load to most able. Under the default SequentialLoad dispatch (§3.4), the first piece of equipment in priority order runs first and hands whatever load it can't meet to the next piece in line — so equipment with a fixed, uncontrolled output that can't track the zone's load belongs first, and equipment that can modulate to meet whatever's left belongs after it. A DOAS air terminal serving a zone alongside a fan coil is the clearest case: the terminal supplies a largely fixed airflow at a tempered — sometimes barely conditioned — supply condition, which can't meet the zone's load on its own and can even add to it rather than reduce it. Placing the DOAS terminal first lets EnergyPlus apply its fixed contribution and pass the remainder to the fan coil. Reversed, the fan coil sizes its response against the zone's

load **before** the DOAS contribution is accounted for, and the DOAS terminal is then handed a share of the load it has no ability to meet — a mistake that shows up as unmet heating and cooling hours.

3.2 `ZoneType` Defaults and Per-Zone Overrides

The dialog has two tabs sharing the same mechanism, differing only in scope:

- **Assign HVAC Systems by Template** — one row per `Sizing:Zone` object or `ZoneType`, setting the *default* equipment set and priority order for every zone that inherits from it.
- **Assign HVAC Systems by Zone** — the same mechanism scoped to one conditioned zone at a time, for a zone that needs to deviate from its default.

Each row shows a drag-sortable list of the equipment currently active for that zone/type, plus every other piece of `ZoneHVAC` equipment available in the model. Clicking an inactive (greyed-out) entry adds it to the zone's active set; dragging within the list reorders load-serving priority among equipment already active, per §3.1's principle. **This dialog only assigns and reorders equipment that already exists somewhere in the model — it does not create new equipment objects.** Place the equipment first (§2, or via a template import), then use this dialog to assign it to a zone.

3.3 Cooling Order vs. Heating Order

A zone's equipment set is the same list whether it's operating in heating or cooling mode — there's no separate "heating equipment" and "cooling equipment" to assign. What differs is priority: the dialog lets you set a distinct load-serving order for heating versus cooling, because a zone's real equipment can legitimately need to be dispatched in a different sequence depending on which load it's meeting.

In practice, the two orders are almost always identical, since §3.1's reasoning rarely depends on which load is being met — a DOAS terminal is no better at tracking a heating load than a cooling one. Setting a heating-specific order is the exception, reserved for the case where dispatch priority genuinely needs to differ by mode. The cooling-order list, in the main dialog, is where equipment is added to or removed from the zone; heating order is set separately, in its own dialog, opened with the **Update Heating Order** button and scoped to whichever zone is currently selected. That heating-order dialog only reorders equipment already active for the zone — it has no equivalent add/remove control of its own, since a zone's equipment set is defined once, on the cooling side.

3.4 Load Distribution Scheme and Sequential Load Fractions

Where more than one piece of equipment is active for a zone, EnergyPlus's **Load Distribution Scheme** field determines how the zone's load is divided among them, on top of the cooling/heating priority order §3.3 sets. Four schemes are available:

- **SequentialLoad** (the default) — equipment runs in priority order; the highest-priority unit is asked to meet as much of the load as it can, and whatever remains is passed to the next unit in sequence, and so on. This is the behavior §3.1's DOAS-before-fan-coil example assumes.
- **UniformLoad** — the total load is divided evenly across every unit currently scheduled available, regardless of priority order or relative capacity; each unit is asked to meet an equal share rather than whatever's left over from the unit ahead of it.

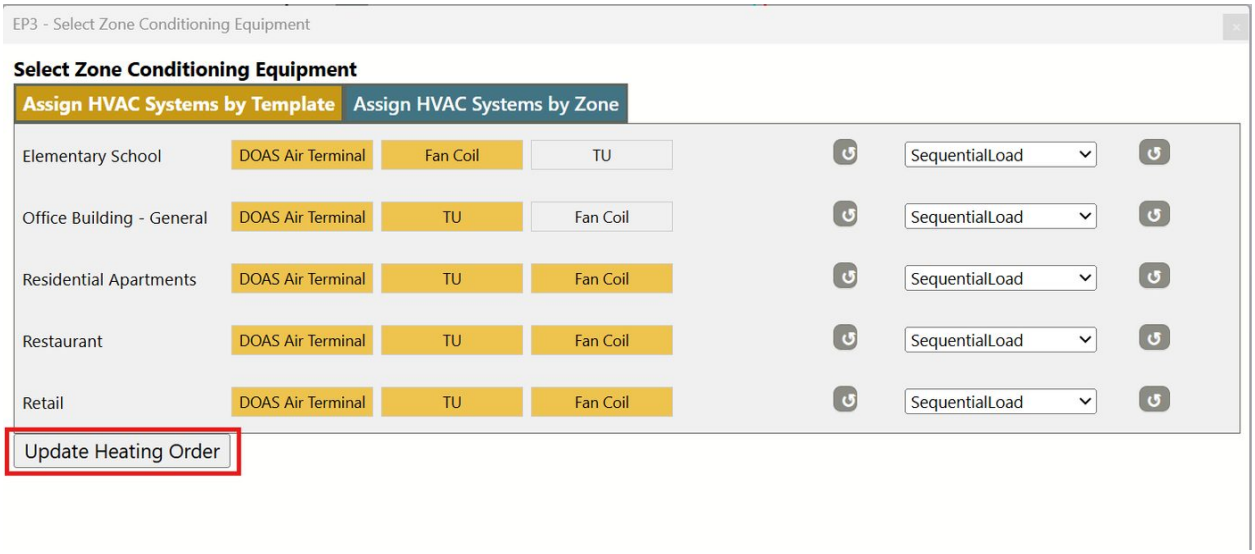


Figure 94: The "Select Zone Conditioning Equipment" dialog for a zone, with the cooling-order sortable list and the Update Heating Order button that opens the separate heating-order dialog.

- **SequentialUniformPLR** — equipment still runs in priority order as under SequentialLoad, but every operating unit is held to the same part-load ratio as the others, rather than each modulating independently to meet its own assigned share.
- **UniformPLR** — every unit scheduled available runs simultaneously, all at the same part-load ratio needed collectively to meet the total load, with no sequential dispatch by priority.

ZoneHVAC:EquipmentList covers the exact dispatch algorithm behind each scheme; EnergyPlus applies whichever one is selected the same way regardless of which equipment types are involved.

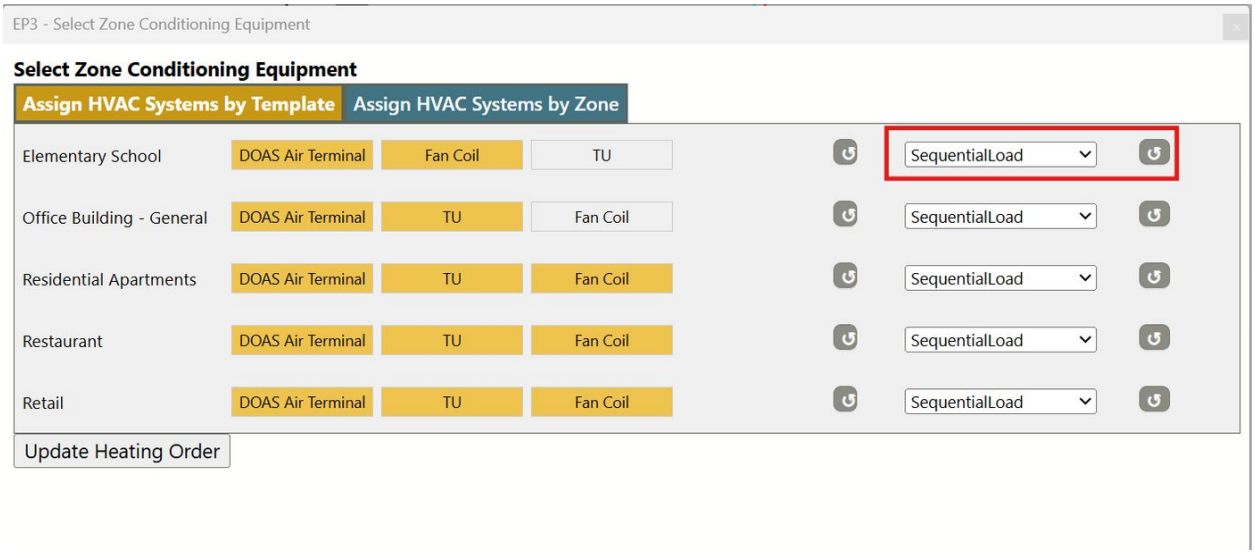


Figure 95: The Load Distribution Scheme field on a zone's equipment list.

Independent of which scheme is selected, each active piece of equipment also carries a **Sequential Cooling Fraction Schedule Name** and a **Sequential Heating Fraction Schedule Name** —

a time-varying multiplier on whatever fraction of the load that unit would otherwise be assigned, so a unit's share can change by hour or season rather than staying fixed at an even split or a fixed sequential remainder. These two fields appear as the last two fields on every `ZoneHVAC:*` and air terminal object's dialog in EP3.

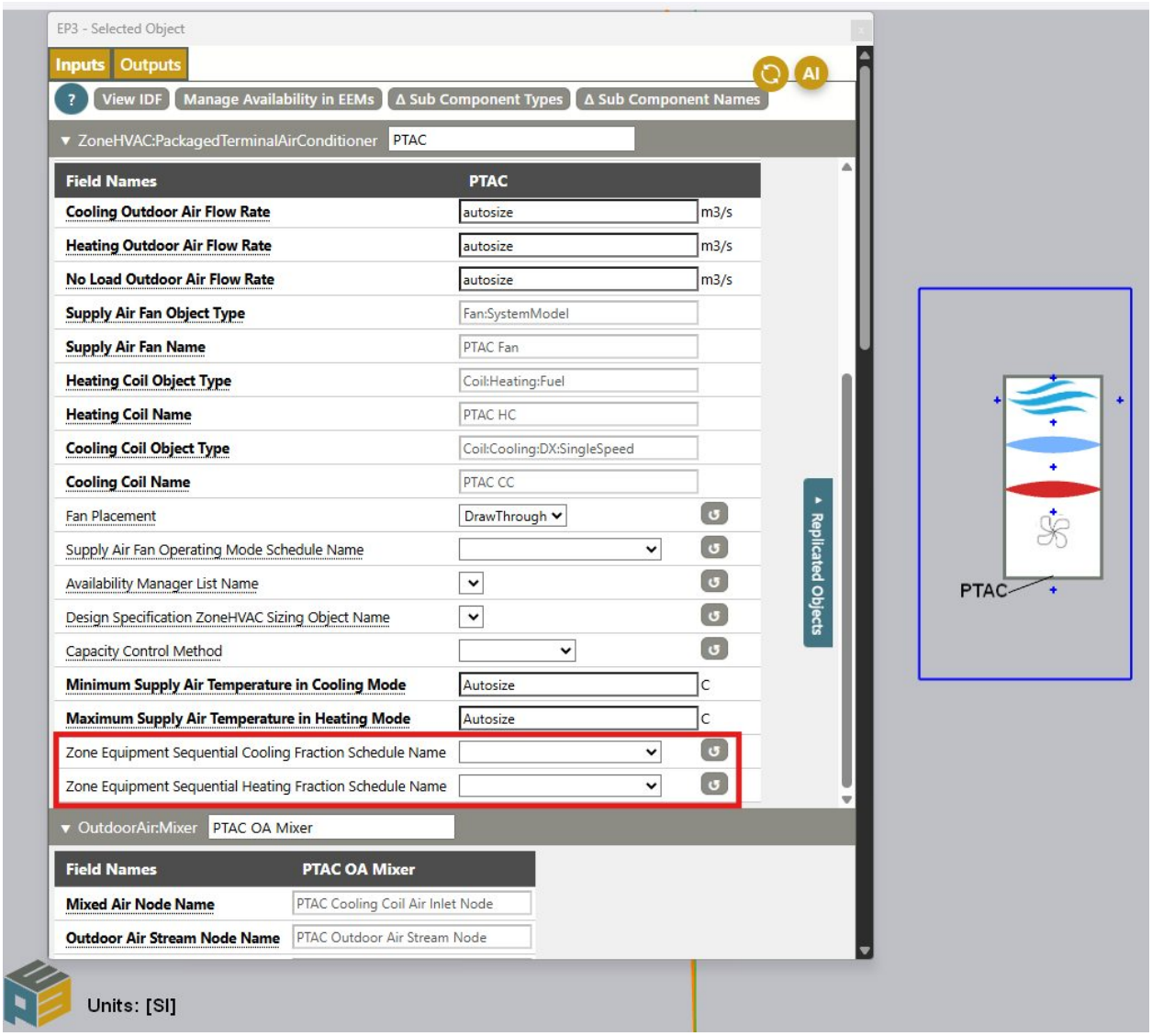


Figure 96: The Sequential Cooling Fraction Schedule Name and Sequential Heating Fraction Schedule Name fields at the bottom of a ZoneHVAC object's dialog.

3.5 Interaction with Energy Efficiency Measures

Assigning equipment to a zone (§3.2) and controlling which Energy Efficiency Measure that equipment participates in are two separate mechanisms. Most objects in an EP3 model, ZoneHVAC equipment included, can be switched on or off per EEM through the **Manage Availability in EEMs** command, reached by clicking the **Manage Availability in EEMs** button from that object's dialog. *(see "Energy Efficiency Measures (EEMs)")* covers EEMs generally. A piece of equipment switched off for a given EEM is absent from that EEM's export entirely — not assigned

to any zone, and not listed on that zone's `ZoneHVAC:EquipmentList` — regardless of what the assignment dialog in §3.2 shows.

Equipment can be switched off in the Global EEM, but the recommended practice is to leave every user-created object on in Globnal. The reasons are connectivity and comprehension: branch and node connections between components can only be made while working in the Global model; and it's helpful to have everything visible in Global, in order to have one, complete view of all **potential** systems. The recommended workflow is to assign every piece of equipment a zone might use to that zone in Global, then use **Manage Availability in EEMs** to exclude whichever pieces a given comparison measure doesn't include. A zone's effective equipment set for any one EEM is the intersection of what's assigned to it in the dialog and what's switched on for that EEM. Reference *(see "Energy Efficiency Measures (EEMs)")* for more specifics on efficiency measures.

---

4. Sizing:Zone

4.1 What `Sizing:Zone` Is, and Why Every Conditioned Zone Needs One

A conditioned zone needs both a thermostat and a `Sizing:Zone` object attached. The thermostat determines the setpoints the zone's equipment controls to (*(see "Internal Loads, Space Types, and Zone Types", §4.1)*); `Sizing:Zone` sets the assumptions EnergyPlus uses to calculate that zone's design heating and cooling loads and airflow on the design day — the numbers every autosized field on the zone's equipment, and everything attached to it (AirLoops, PlantLoops and plant equipment) rely on.that thermostat.

EP3 flags a missing or duplicate `Sizing:Zone` directly, in the "Assign HVAC Systems by Template" tab:

- a zone with a thermostat and no `Sizing:Zone` at all ("this will likely cause a fatal error in EnergyPlus"), and
- a zone attached to more than one `Sizing:Zone` object ("this will cause a fatal error in EnergyPlus").

A `Sizing:Zone` reaches a zone one of two ways: automatically, as part of a `ZoneType` template whenever its Thermostat Configuration is set to anything but `None` (*(see "Internal Loads, Space Types, and Zone Types", §4.3)*); or added independently, as a loose object pointed at a Zone or ZoneList by hand — the path a model built in a more traditional, EnergyPlus-native style is likely to use.

4.2 What Sizing:Zone Controls: Design Air Flow Method, Outdoor Air, and Sizing Factors

`Sizing:Zone` has a large field set, and Sizing:Zone is the place for exhaustive field-by-field detail. We discuss a few fields in more detail here because they're the ones most likely to cause confusion, or to need a deliberate choice, rather than being left at their default:

Cooling Design Air Flow Method and Heating Design Air Flow Method (each defaulting to `DesignDay`) set how the zone's design airflow is derived. `DesignDay` — the normal case — takes the airflow straight from EnergyPlus's own design-day heat-balance calculation: whatever airflow the zone's peak sensible load and supply air condition (§4.4) imply. `Flow/Zone` bypasses that calculation entirely: the zone's design airflow becomes whatever value is entered directly

in **Cooling/Heating Design Air Flow Rate**, unaffected by what the load calculation would otherwise produce.

`DesignDayWithLimit` applies to both fields, but clamps in opposite directions depending on which one it's set on. On the cooling side, it floors the design-day-calculated airflow at a **Cooling Minimum Air Flow** (or **...per Zone Floor Area**, or **...Fraction**) — a convention tied to VAV-terminal-with-reheat sizing, where a VAV box's minimum position needs a floor so it isn't sized to an unrealistically small cooling airflow. On the heating side, the same option instead caps the calculated airflow at a **Heating Maximum Air Flow** (or **...per Zone Floor Area**) — a ceiling, not a floor. Because the heating-side limit only ever caps, it can't be used to guarantee a zone a minimum heating airflow or capacity the way the cooling side's minimum can — a zone whose heating load would otherwise call for more airflow than the design-day calculation produced gets no equivalent floor to raise it.

Design Specification Outdoor Air Object Name sets which ventilation-air requirement the zone-load calculation credits when it computes the zone's design airflow. It accepts either a single `DesignSpecification:OutdoorAir` object — one ventilation rate for the whole zone — or a `DesignSpecification:OutdoorAir:SpaceList`, which lets Spaces within the zone carry different outdoor-air requirements that still roll up into one zone-level calculation. This is the same field the cascading-defaults mechanism (§4.3) can supply a default for; what it does mechanically, independent of where its value comes from, is tell `Sizing:Zone` how much outdoor air the zone's design load needs to account for.

Account for Dedicated Outdoor Air System (default `No`) tells the calculation whether to credit a DOAS's sensible contribution before sizing the rest of the zone's equipment. Set to `Yes`, EnergyPlus computes an idealized DOAS sensible heat gain or loss — proportional to the zone's design ventilation airflow and the difference between an assumed DOAS supply air temperature and the zone temperature (EnergyPlus Engineering Reference, Zone Design Loads and Air Flow Rates) — and adds or subtracts that from the zone's total load before sizing the rest of its equipment, so that equipment isn't sized to condition ventilation air the DOAS has already handled. The assumed supply air temperature comes from **Dedicated Outdoor Air System Control Strategy**, which also sets the default **Dedicated Outdoor Air Low/High Setpoint Temperature for Design** (both `Autosize` unless overridden):

- `NeutralSupplyAir` (the default) — holds the DOAS supply air between the low and high setpoints (21.1°C/23.9°C, 70°F/75°F by default), close to room-neutral, so the DOAS imposes little sensible load either way and exists mainly for ventilation.
- `NeutralDehumidifiedSupplyAir` — cools and dehumidifies the outdoor air, then reheats it back up to the high setpoint (14.4°C/22.2°C, 58°F/72°F by default) so it arrives sensibly neutral: the DOAS does latent/dehumidification work without contributing to the zone's sensible load calculation.
- `ColdSupplyAir` — delivers cold, dehumidified ventilation air (12.2°C/14.4°C, 54°F/58°F by default) that actively offsets part of the zone's sensible cooling load, so the zone's own equipment is sized smaller to reflect that contribution.

The credit this produces only ever reflects what the control strategy *assumes* the DOAS delivers — §5.2 returns to what happens when a zone's actual equipment configuration doesn't match that assumption.

Zone Heating Sizing Factor and Zone Cooling Sizing Factor apply a safety margin on top of the calculated peak load — a factor of 1.15, for instance, sizes the zone's design load and airflow

15% above what the design-day calculation alone produced, to absorb load-calculation uncertainty rather than sizing equipment to the calculated peak exactly. Left blank or zero, each falls back to the matching global factor on `Sizing:Parameters` (§4.6) instead of defaulting to no margin at all.

4.3 The Cascading Equipment-Default Fields

When `Sizing:Zone` is used **without** `ZoneType`, it carries the four EP3-only fields — availability schedule, supply-air-fan-operating-mode schedule, design-specification-outdoor-air object, and design-specification-ZoneHVAC-sizing object — that supply cascading defaults to ZoneHVAC equipment. The full three-tier lookup (the equipment's field → the zone's override → the template default) is covered in *(see "Internal Loads, Space Types, and Zone Types", §4.2)* and works identically no matter how the `Sizing:Zone` got attached — a loose, independently-added `Sizing:Zone` cascades to its zone's equipment exactly the same way a `ZoneType`'s does.

Which of an equipment object's fields pick up that cascade depends on the object. Most ZoneHVAC equipment's **Availability Schedule Name**, **Supply Air Fan Operating Mode Schedule Name**, and **Design Specification Outdoor Air Object Name** fields match the cascade's three concepts by name directly. A few don't, and fall back on the nearest available field instead: the VRF terminal unit's availability field, named **Terminal Unit Availability Schedule**, uses the same availability-schedule default despite the different name. Four-Pipe Fan Coil has no outdoor-air-object field at all — its ventilation is governed by an **Outdoor Air Schedule Name** that modulates outdoor air as a fraction of its maximum flow rather than pointing at a sizing object — so that field cascades from the fan-operating-mode default instead, on the reasoning that an OA-modulation schedule tracks a zone's hours of operation more closely than an always-available schedule does.

4.4 What Sizing:Zone Calculates: Supply Air Temperature and Airflow, Worked

Beyond §4.2's fields, `Sizing:Zone` also sets the zone load sizing method, whether to credit a dedicated outdoor air system separately, and — where a zone contains more than one Space — whether those Spaces' individual loads sum coincidentally or non-coincidentally (**Type of Space Sum to Use**). Sizing:Zone covers these exhaustively too; the relationship worth walking through here is Design Supply Air Temperature's direct effect on the calculated design airflow, since it's the assumption most likely to disagree with what real equipment can deliver.

EnergyPlus derives a zone's design cooling airflow from its calculated sensible load and the temperature difference between the zone and the assumed supply air (EnergyPlus Engineering Reference, Zone Sizing):

Airflow [cfm] = Sensible Load [Btu/h] ÷ (1.08 × ΔT [°F])

Suppose a zone's calculated design sensible cooling load is 12,500 Btu/h, its cooling setpoint is 75°F, and its **Zone Cooling Design Supply Air Temperature** is set to 55°F (ΔT = 20°F):

`Airflow = 12,500 ÷ (1.08 × 20) ~ 579 cfm`

In SI: the same load is 3,663 W, the same ΔT is 11.1°C, and the equivalent relationship is Airflow [L/s] = Sensible Load [W] ÷ (1.2 × ΔT [°C]):

`Airflow = 3,663 ÷ (1.2 × 11.1) ~ 275 L/s ~ 583 cfm` — matching the IP result within rounding.

(1.08 and 1.2 are the standard air sensible-heat-transfer constants at typical sea-level air density and specific heat, per ASHRAE Fundamentals.)

Lowering the assumed supply air temperature (52°F instead of 55°F, say, to shrink the calculated airflow and duct sizes) raises ΔT and lowers the calculated design airflow accordingly — `12,500 ÷ (1.08 × 23) ~ 503 cfm` in this example. That's a legitimate design choice, but only if the equipment eventually selected can deliver air at that leaving-coil temperature at the entering conditions the zone will see. If it can't, the zone's design airflow was calculated against a supply air temperature the real coil never reaches — see §5.2 for the consequence.

> Whenever one of `Sizing:Zone`'s numeric fields is left blank, it resolves to Autosize (or Autocalculate) rather than a hardcoded default — this is an ordinary numeric field, not a separate toggle, and EnergyPlus itself computes the value during the sizing calculation. There's nothing EP3-specific about this mechanism; it applies identically to every autosizable field on every object in the codebase, not just `Sizing:Zone`'s.

4.5 DesignSpecification:ZoneHVAC:Sizing and DesignSpecification:AirTerminal:Sizing

`Sizing:Zone`'s **Design Specification ZoneHVAC Sizing Object Name** field can point at a `DesignSpecification:ZoneHVAC:Sizing` object, which lets a zone apply separate scaling factors and flow-rate limits to its ZoneHVAC equipment's sizing independent of the airloop-served portion of the same zone's load. It's a freestanding object, added and configured the same way any other loose object is; nothing about attaching it beyond pointing `Sizing:Zone`'s field at it requires further steps.

`DesignSpecification:AirTerminal:Sizing` attaches to an air terminal through a **Design Specification Air Terminal Sizing Object Name** field EP3 adds directly on the air terminal's own dialog — the EnergyPlus field this feeds actually belongs to `ZoneHVAC:AirDistributionUnit`, which EP3 doesn't expose as an editable object at all (it's assembled automatically at export time from whichever air terminal is connected), so EP3 surfaces the reference on the terminal itself and threads it through into that generated object. Every air terminal type — VAV and constant-volume, with or without reheat, dual-duct, series/parallel fan-powered, and `AirTerminal:SingleDuct:Mixer` — carries this field. *(see "Air Loops, Branches, and Nodes")* covers air terminals themselves in full.

4.6 A note on Sizing:Parameters

`Sizing:Parameters` sets global heating and cooling sizing factors that `Sizing:Zone`'s sizing-factor fields fall back to when left blank or zero. This fallback is EnergyPlus's runtime behavior. EP3 doesn't compute or surface it, and has no dialog or wizard step for it — it's reachable only as a loose object from the generic HVAC Design Objects list, and only one should exist in a model.

---

5. How Sizing Propagates, and Where It Breaks

5.1 From zone to system to plant

A zone's `Sizing:Zone` output feeds two places: directly into the autosizing of whatever ZoneHVAC equipment serves that zone, and — where the zone is instead (or additionally) served by an airloop — into that airloop's `Sizing:System` calculation, which sums its served zones' design

loads into a system-level design airflow and capacity. `Sizing:System` resolves the same coincident-versus-non-coincident question `Sizing:Zone`'s **Type of Space Sum to Use** does, one level up, through its **Type of Zone Sum to Use** field — *(see "Air Loops, Branches, and Nodes")* covers `Sizing:System` in full. From there, plant-side equipment sizing follows the same pattern again through `Sizing:Plant` — *(see "Plant Loops and Setpoint Control").*

5.2 Where a mismatch shows up

`Sizing:Zone` calculates loads and airflow from a set of assumptions — a supply air temperature, an outdoor-air accounting method, a sizing factor — and equipment is then autosized to meet exactly what that calculation produced. Nothing checks, while you're building the model, that the equipment you assign can meet those assumptions in practice:

- **An unreachable supply air temperature.** §4.4's worked example: a `Sizing:Zone` calculated against a 52°F supply air temperature sizes the zone's equipment for a smaller design airflow than a 55°F assumption would. If the fan coil or coil selection eventually used can't reach 52°F leaving-air temperature at real entering conditions, the zone runs at less airflow, and therefore less sensible capacity, than its design load requires — showing up as unmet cooling hours rather than as an error.
- **A dedicated-outdoor-air assumption that doesn't match reality.** §4.2's **Account for Dedicated Outdoor Air System** credit is only as good as its assumption. If that field is set to `Yes` while the zone's actual equipment configuration draws unconditioned or partially-conditioned outdoor air directly — no DOAS air terminal serving the zone, or a control strategy assuming colder or more neutral supply air than what's physically delivered — the equipment's calculated capacity was never sized to handle the ventilation load it now carries.
- **Zero or duplicate `Sizing:Zone` objects** (§4.1) — the two fatal-error conditions EP3 already flags directly.
- **A zone with equipment but no active thermostat/`Sizing:Zone` pairing**, or a VRF terminal unit never connected to a condensing unit (*(see "The HVAC Outliner and Creating HVAC Systems", §5)*) — both fail outright rather than producing subtly wrong results, and both are checked automatically (§5.3).

None of these are exotic failure modes — they follow directly from `Sizing:Zone`'s assumptions and a zone's equipment configuration being edited somewhat independently of each other over the course of building a model. Whenever a zone shows persistent unmet hours, or capacity that doesn't track the load you'd expect, check whether `Sizing:Zone`'s assumptions still match what's connected to that zone before assuming the equipment is undersized.

5.3 What's checked automatically, and what isn't

EP3's "Validate Model" dialog's HVAC checks catch the two structural failures above outright: a zone with a thermostat and no equipment, or no `Sizing:Zone`, fails validation directly, as does a VRF terminal unit with no condensing unit assigned. **What EP3 does not check** is whether `Sizing:Zone`'s assumptions are internally consistent with the equipment assigned — an unreachable supply air temperature, a DOAS-accounting mismatch, or an autosized capacity that turns out too small in practice all surface only as EnergyPlus's runtime unmet-hours output or severe/fatal errors, discovered by running the simulation rather than flagged ahead of it.

Figure placeholder — the "Validate Model" dialog's HVAC category, showing a zone

flagged for having a thermostat but no Sizing:Zone object, alongside the warning text EP3 displays for that condition

---

6. Where to Look Next

- *(see "The HVAC Outliner and Creating HVAC Systems")* — the Outliner, Components dialog, ZoneHVAC-vs-AirLoopHVAC choice, and VRF terminal-unit connection.
- *(see "Air Loops, Branches, and Nodes")* — air terminal units, `Sizing:System`, and airloop topology.
- *(see "Plant Loops and Setpoint Control")* — `Sizing:Plant` and the loops this chapter's plant-connected equipment depends on.
- *(see "Central Plant Equipment")* — the coils, boilers, and chillers a plant-connected zone object ultimately draws from.
- *(see "Internal Loads, Space Types, and Zone Types", §4.2–4.3)* — the `ZoneType`/`Zone`/`Sizing:Zone` cascade mechanism in full.
- *(see "Fans and Fan Power")* — the composite sub-component pattern, and fan-specific scheduling detail.
- *(see "Model Troubleshooting and Quality Assurance")* — reading "Validate Model" output generally.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; confirm the exact UI label text for Sizing:Zone's cascading-default fields against a live dialog (this draft follows the labels already used in Ch.9); the `sizing-hvac-systems-and-zones` XREF slug used in Ch.9 §4.3 and Ch.12 predates this chapter and should be repointed to this chapter's slug once the reorg is final.

15. Air Loops, Branches, and Nodes

1. Overview

(see "The HVAC Outliner and Creating HVAC Systems") already covers the HVAC Outliner's tree structure, the Components dialog, the Draw Branch/Erase Branch tools' existence, and how a system is created — either by importing a template or built from individual pieces. This chapter goes deeper into the specifics of `AirLoopHVAC`'s branch structure, what a node connection is (and isn't), how a terminal unit connects an airloop to zone(s), how the loop's sizing object works, how its controls attach, and how one airloop definition can stand in for many physical airloops through replication.

(see "Zone-Level HVAC Equipment and Zone Sizing") is also assumed reading here, specifically for the composite-object/sub-component pattern (a Reference class plus one or more swappable sub-component slots).

What this chapter defers:

- Detail on boilers, chillers, pumps, and cooling towers — *(see "Central Plant Equipment")*.
- Performance curves generically — *(see "Performance Curves: Part-Load and Efficiency Behavior")*.
- `PlantLoop` topology and `Sizing:Plant` field-by-field detail — *(see "Central Plant Equipment")*.
- `AvailabilityManager` control-strategy detail beyond §8.2/§8.3, and the `Controller:WaterCoil/Controlle` control layer — *(see "Controllers and Availability Managers")*. Common airside `SetpointManager` types are covered directly in §8.1 below; `PlantLoop`-specific types are *(see "Central Plant Equipment")*'s job.
- Outdoor Air System details — *(see "Outdoor Air, Ventilation, and Energy Recovery")*.
- Viewing and interpreating simulation results — *(see "Viewing and Interpreting Results")*.
- Validation, unmet hours, `NodeReport` in full, and reading the EnergyPlus `.err` file — *(see "Model Troubleshooting and Quality Assurance")*.

---

2. `AirLoopHVAC` Branch Anatomy

> *Figure placeholder — an annotated diagram of a typical single-branch AirLoopHVAC, labeling the supply side (outdoor air system, heating coil, cooling coil, supply fan) and the demand side (zone splitter, terminal units, zones, zone mixer, return path) as two distinct halves of one loop*

2.1 Supply vs. demand: what each side actually does

In EnergyPlus, supply-side and demand-side equipment are distinguished by function, not just by position in the loop. Demand-side equipment *demands* a load — under the hood, it requests a flow rate and lets upstream equipment produce whatever condition results. Supply-side equipment *meets* a load — it's driven not just by a requested flow, but by a setpoint it's expected to hold.

That supply-side setpoint is usually established by a `SetpointManager` (§8) sitting on the loop, but not always. An airloop can legitimately have no `SetpointManager` at all — an uncontrolled DOAS system for example, where the supply air condition floats, or a system where an

`AirLoopHVAC:Unitary*` object sits directly on the supply branch and controls itself against zone conditions, rather than the loop's outlet node being driven to a setpoint directly.

2.2 Supply-side branch shapes

Within the supply side, EP3 supports four structural shapes, but in practice you should default to the first one:

- **Single branch** — one continuous run of equipment from the supply inlet to the supply outlet. This is the default in the current EP3 templates and in the default in the **Create Air Loop** dialog.
- **Inlet / Parallel Branches / Outlet** — a splitter divides the supply air across two or more parallel equipment trains, which a mixer then recombines into one supply outlet before the demand side. This mirrors how a `PlantLoop` stages multiple boilers or chillers in parallel. It's available as a Branch Configuration choice, but only when Number of Outlets is set to 1 — and it's rarely what an air-side system needs, since air-handling equipment is seldom built as two full redundant fan/coil trains reconverging into one duct the way plant equipment routinely is.
- **Two- or three-outlet, no mixer** — the supply side splits into two or three genuinely separate downstream duct runs, each with a distinct outlet node and a distinct `AirLoopHVAC:SupplyPath`, and nothing recombines them. This is the shape a dual-duct (hot-deck/cold-deck) system needs. EP3's Create Air Loop dialog doesn't support building one directly — its air-terminal-type list excludes every `AirTerminal:DualDuct:*` type — so a dual-duct system has to be assembled and wired by hand rather than through the guided path.

Recommendation: build single-branch airloops unless your system specifically needs separate duct runs from one air handler (dual duct) or a genuinely redundant, fully parallel equipment train. Both multi-branch shapes exist and are supported, but neither is auto-wired for you the way the single-branch path is (§11), and both are uncommon enough in real designs that treating single-branch as the default is the right instinct.

2.3 Demand-side architecture at a glance

The demand side is built from two kinds of objects: air terminals, and optionally, supply/return plenums or `AirLoopHVAC:ZoneSplitter`/`AirLoopHVAC:ZoneMixer`. For most day-to-day modeling, air terminals (§5) are the only demand-side objects you'll need to configure. EP3 handles all of the default required EnergyPlus `AirLoopHVAC:ZoneSplitter`/`AirLoopHVAC:ZoneMixer` objects for you. The zone splitter and mixer are placeable components too (from the Components dialog, wired with the Draw Branch tool like any other equipment, §3.1), if you would like to model more than one on the demand side of your `AirLoop` but EP3 supports only a single level of fan-out: one zone splitter feeding a set of terminal units directly, not a splitter feeding another splitter or any other cascading distribution arrangement. §6 covers plenums, the other demand-side object type you might place.

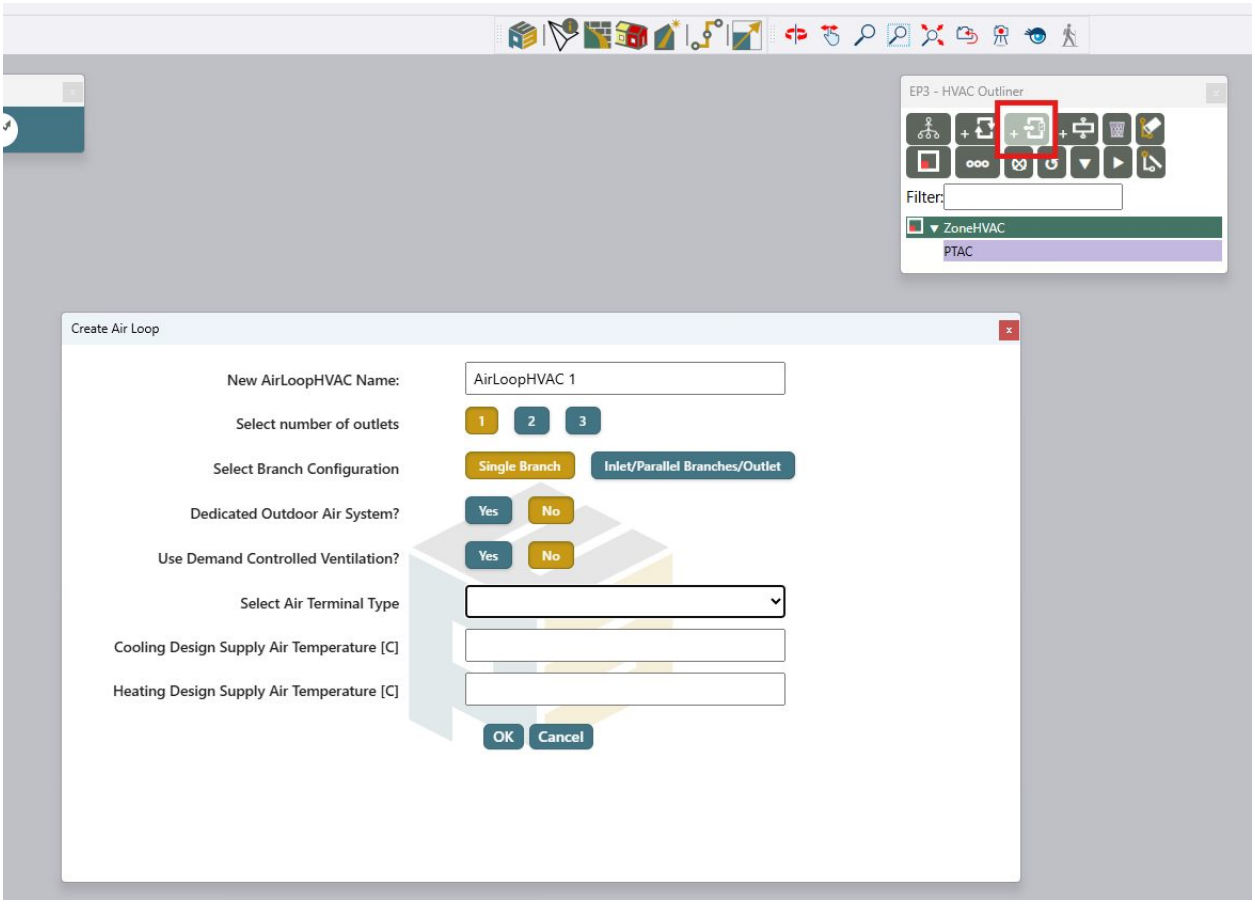


Figure 97: The "Create Air Loop" dialog, showing the Number of Outlets and Branch Configuration fields.

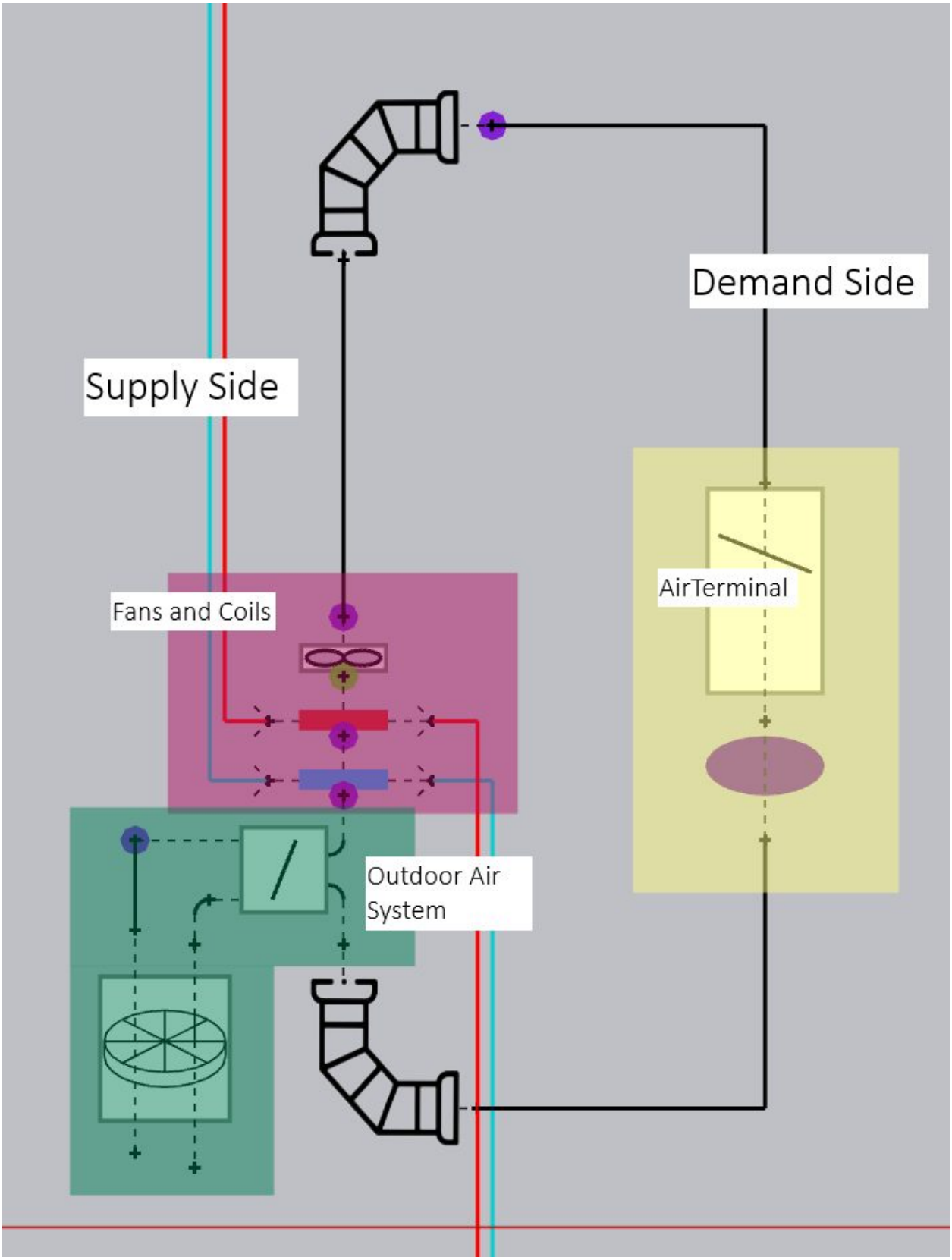


Figure 98: A single-branch AirLoopHVAC's supply and demand side, with terminal units wired directly to the demand side and no explicit ZoneSplitter/ZoneMixer components shown.

3. How Branch Connections Map onto an `AirLoopHVAC`

3.1 The main branch and terminal units

Every node on an `AirLoopHVAC`'s branches carries air, so the Draw Branch tool (*(see "The HVAC Outliner and Creating HVAC Systems", §2.3)*) lets you connect any two nodes on the same side of the loop freely. As a general rule, if the tool offers a connection, it belongs there — the tool itself won't offer one that doesn't — and EP3 generates whatever supporting IDF objects (splitters, mixers, paths, node lists) that connection needs behind the scenes, without requiring you to build or verify them yourself. In practice, that means placing a fan or coil's air-inlet/air-outlet nodes in series along the supply branch, in whatever order you want to build them (draw-through vs. blow-through, covered with a worked branch order in *(see "Fans and Fan Power", §3.1)*).

A terminal unit sits on the demand side. Only its air-inlet node is offered as a connection point. If the terminal unit includes any sub-components (fans or coils), those node connections are wired automatically by EP3.

3.2 Return air is a zone assignment, not a branch connection

A `PlantLoop` is a closed loop: the same fluid leaves the supply side, passes through demand-side equipment, and returns, so tracing branch connections tells you the whole story of where flow goes. An `AirLoopHVAC`'s return pathworks differently. Supply air is delivered into a zone through a terminal unit, but a zone is not a closed system. Air can be supplied by `ZoneHVAC:*` equipment, other `AirLoops`, introduced or lost via infilttration or mixed with other zones.

Which zones' return air comes back to a given `AirLoopHVAC` is instead a separate, explicit setting on the loop: a checkbox list of every conditioned zone in the model. By default, every zone the loop supplies with a terminal unit is checked, and a **Reset All** button restores that default at any time. You can also check a zone the loop doesn't supply at all, or leave the same zone checked for more than one loop — EP3 doesn't restrict either. A zone providing return air to more than one `AirLoopHVAC` is a legitimate configuration.

Because this list is a manual setting rather than something derived from branch geometry, it's easy for it to quietly drift from what you intended — after adding a terminal unit, adding a zone, or copying part of a model.

An `AirLoopHVAC:ReturnPlenum` or a relief-outlet path placed on the demand side claims specific zones the same way, at that object's level rather than the loop's — §6 covers plenums.

---

4. `Sizing:System` — The Loop's Sizing Object

4.1 A convenience object, included by default

A `Sizing:System` object is provided within the `AirLoopHVAC`'s dialog for convenience. EP3 creates it automatically — to find it, scroll down in the **Selected Object Dialog** with the `AirLoopHVAC` selected, where it appears inline under a checkbox labeled **"Include Sizing Object in EEM [name]?"** that lets you decide whether the loop uses it.

Unchecking that box doesn't delete the object; it excludes it — and its export — from the current EEM. Doing so means every autosizable field on every fan, coil, `AirLoopHVAC:Unitary*` object, and sub-component on the loop has to be hard-sized instead, since nothing remains to drive EnergyPlus's

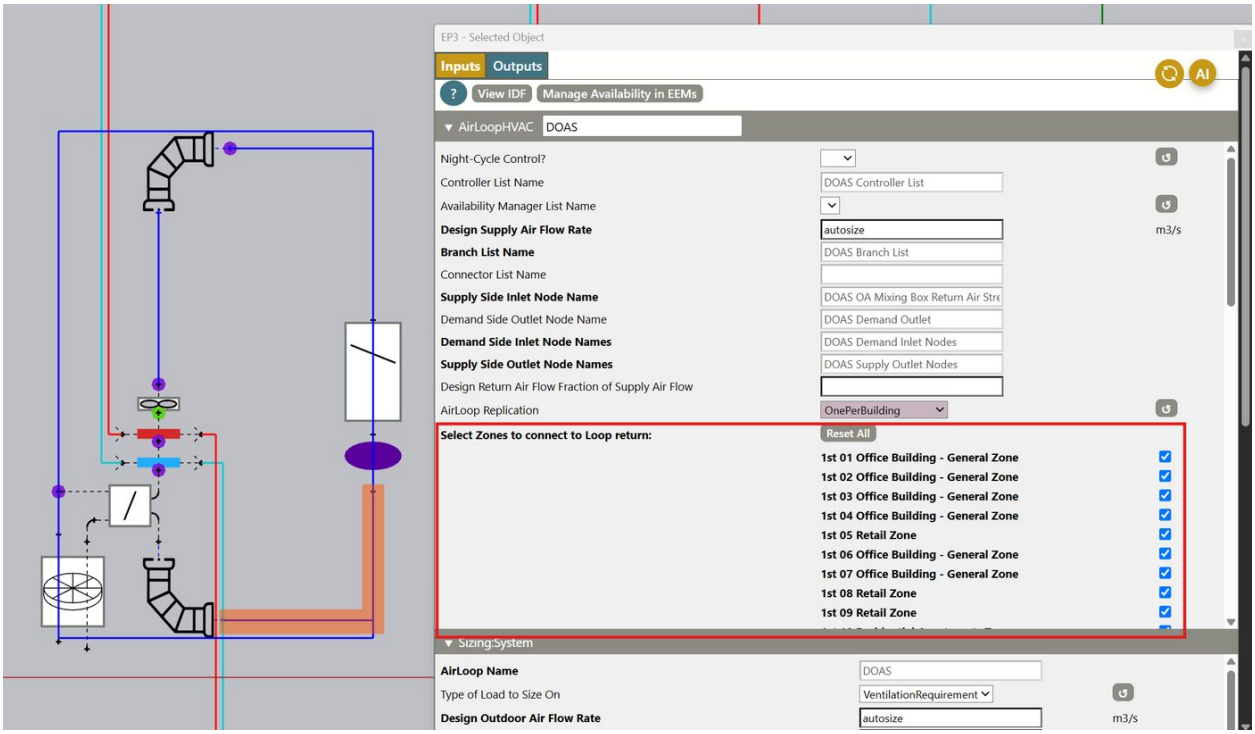


Figure 99: The demand return path drawn from a terminal unit's zone back to the AirLoopHVAC, reflecting that zone's return-air assignment.

sizing calculation. This is extremely uncommon in practice — leave the checkbox checked unless you have a specific reason to size the entire loop by hand.

4.2 Field Set

Field	Default	Notes
Type of Load to Size On	Sensible	`Latent`, `Sensible`, `Total`, or `VentilationRequirement`
Design Outdoor Air Flow Rate	Autosize	
Preheat/Precool Design Temperature & Humidity Ratio	(required; seeded by the wizard — see below)	Sizes any preconditioning of outdoor air ahead of the main coils
Central Cooling/Heating Design Supply Air Temperature	(required)	Drives coil capacity sizing — §4.4
Central Cooling/Heating Design Supply Air Humidity Ratio	0.008	
Type of Zone Sum to Use	NonCoincident	`Coincident` or `NonCoincident` — §4.3
100% Outdoor Air in Cooling / Heating	No	Set automatically for a DOAS system — see below

Field	Default	Notes
Cooling/Heating Supply Air Flow Rate Method	DesignDay	Alternatives include a flat flow rate, a per-floor-area value, or a fraction of the autosized flow
System Outdoor Air Method	ZoneSum	`Standard62.1SimplifiedProcedure`, `Standard62.1VentilationRateProcedure`, or `ZoneSum`
Zone Maximum Outdoor Air Fraction	1.0	
Cooling/Heating Design Capacity Method	CoolingDesignCapacity / HeatingDesignCapacity	Alternatives include a flat capacity, per-floor-area, or a fraction of the autosized capacity
Central Cooling Capacity Control Method	OnOff	`Bypass`, `OnOff`, `VAV`, `VT`
Occupant Diversity	Autosize	Used by the two Standard 62.1 outdoor air methods — see §4.5 for why this field is exempt from EP3's usual Autosize-forces-required rule
Heating Coil Sizing Method	None	`CoolingCapacity`, `GreaterOfHeatingOrCooling`, `HeatingCapacity`, `None`
Maximum Heating Capacity To Cooling Capacity Sizing Ratio	1.0	

Sizing:System covers every field exhaustively. The Dedicated-Outdoor-Air-System *accounting* fields (Account for Dedicated Outdoor Air System, its control strategy, and its design setpoint temperatures) live on `Sizing:Zone` rather than here; the system-level analog for a DOAS airloop is the `VentilationRequirement`/100%-outdoor-air combination described below.

What the Create Air Loop dialog seeds. The Create Air Loop dialog (§2) prompts for a Name, Number of Outlets, Branch Configuration, **"Dedicated Outdoor Air System?"**, **"Use Demand Controlled Ventilation?"**, an air terminal type, and **Cooling/Heating Design Supply Air Temperature**. Selecting different combinations changes what the initial values given to the new `AirLoopHVAC's Sizing:System`:

- **Preheat Design Temperature** and **Preheat Design Humidity Ratio**, and **Precool Design Temperature** and **Precool Design Humidity Ratio**, are always set — to 7°C/0.008 and 12.8°C/0.008, respectively — regardless of your climate or design conditions. Review and edit these as needed.
- If you answer **Yes** to Dedicated Outdoor Air System, **Type of Load to Size On** is set to `VentilationRequirement` and both **100% Outdoor Air in Cooling** and **100% Outdoor Air in Heating** are set to Yes. Answering Yes to Use Demand Controlled Ventilation doesn't touch `Sizing:System` at all — it configures the airloop's `Controller:MechanicalVentilation` instead, covered in *(see "Outdoor Air, Ventilation,*

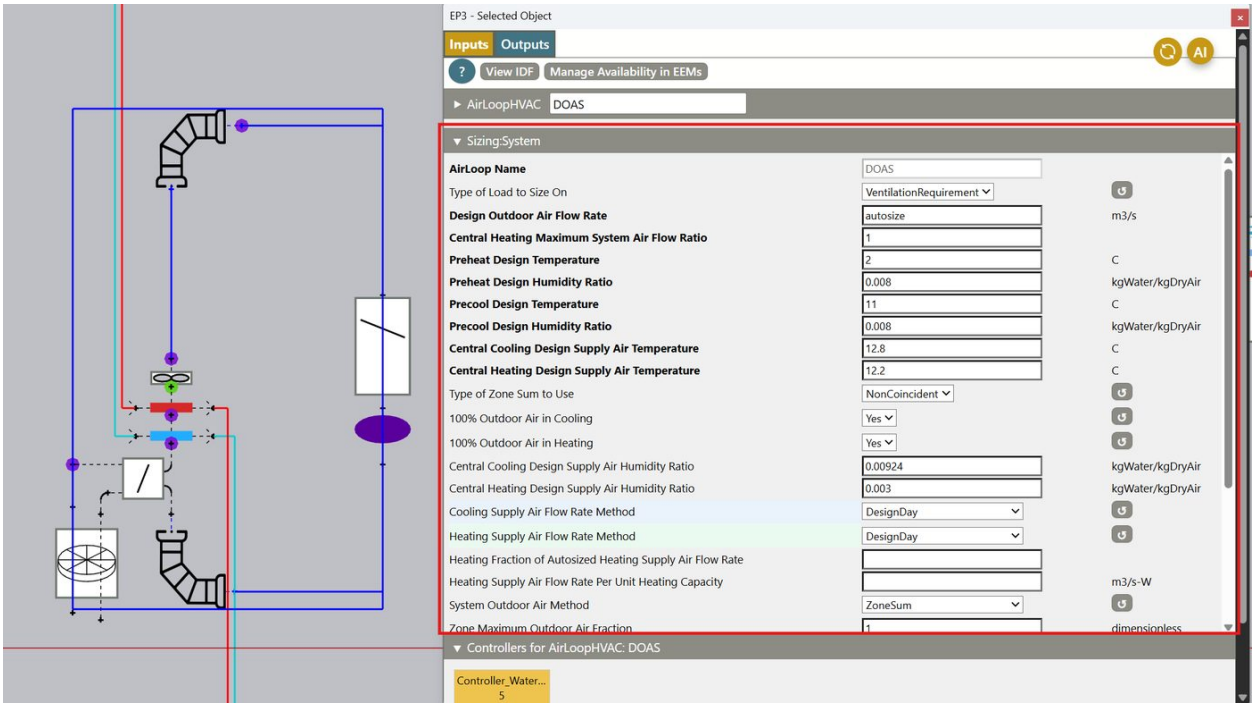


Figure 100: The Sizing:System object embedded inline in the AirLoopHVAC's Selected Object Dialog, under the "Include Sizing Object in EEM" checkbox.

and Energy Recovery").

- **Central Cooling Design Supply Air Temperature** and **Central Heating Design Supply Air Temperature** are only written if you actually typed a value into the corresponding prompt — leave one blank and the field stays blank, ready for you (or EnergyPlus's default) to resolve later.

4.3 Coincident vs. non-coincident system sizing

Type of Zone Sum to Use decides how the individual design airflows `Sizing:Zone` computes for served zone get combined into one system-level design airflow:

- **NonCoincident** (the default) simply adds up each zone's peak design airflow, even though individual zones may peak at different times. This is the conservative assumption — it treats every zone as if it peaks (or could peak) simultaneously, even when they don't.
- **Coincident** instead uses the system's peak — the airflow (summed across all zones) at whichever single hour the combined total is highest across the design day. Because zones rarely all peak at exactly the same hour, this coincident peak is generally lower than the simple non-coincident sum.

Illustrative example (the exact coincident value depends on each zone's full hourly profile, not only its individual peak, so this is conceptual rather than a formula you can hand-calculate): suppose three zones served by one loop have individual peak cooling design airflows of 500 cfm (a west-facing zone, peaking around 2pm from solar gain), 350 cfm (an east-facing zone, peaking around 10am), and 420 cfm (an interior zone, peaking near noon). NonCoincident sizing gives a system design airflow of 500 + 350 + 420 = **1,270 cfm**, the arithmetic sum regardless of timing.

Coincident sizing instead looks at the three zones' combined airflow at whichever hour that sum is highest across the day — since the three peaks don't align, that hour's combined total will be some value below 1,270 cfm, producing a smaller central fan/coil design. Which choice is right for a given project depends on whether the diversity NonCoincident ignores is real for the building — a conservative default exists precisely because assuming diversity that doesn't exist under-sizes the system.

`Sizing:Zone`'s Type of Space Sum to Use field (default `Coincident`) asks the equivalent question one at the zone level, when a single zone is built from multiple `Space` objects — *(see "Zone-Level HVAC Equipment and Zone Sizing")* covers it in full.

4.4 How Coil Capacity and Water-Side Flow Get Sized Together

Central Cooling/Heating Design Supply Air Temperature, combined with the system's design airflow (§4.3) and the mixed-air (return + outdoor air) condition assumed on the design day, sets the air-side ΔT a downstream coil is sized against. A colder cooling supply air temperature (a bigger ΔT across the coil) increases the sized cooling coil capacity for the same airflow; the same relationship holds on the heating side.

Once a water coil's capacity is fixed that way, EnergyPlus sizes that same coil's **water-side flow rate** from its `PlantLoop`'s `Sizing:Plant` **Design Loop Exit Temperature** and **Loop Design Temperature Difference** — solving $Q = m. \cdot cp \cdot \Delta T$ for m. using the plant's design ΔT and the capacity the air-side pass already produced. EnergyPlus's own multi-pass sizing engine handles this directly: an air-loop sizing pass establishes the coil's capacity, then a plant-loop sizing pass uses that capacity to size the water-side flow. This is what lets `Sizing:System` and `Sizing:Plant` be entered and maintained as two independent objects, each describing its own side of the coil, without cross-checking one against the other by hand. Setting a `Sizing:Plant` Loop Design Temperature Difference that doesn't match what a coil's air-side sizing implies produces an oversized or undersized water flow rate — and, downstream, pump and pipe sizing — rather than a capacity error, since capacity is fixed by the air side first. *(see "Central Plant Equipment")* covers `Sizing:Plant` field-by-field.

`Coil:Cooling:Water` and `Coil:Heating:Water` each carry their own water-temperature fields — Design Inlet Water Temperature on the cooling coil, Rated Inlet Water Temperature and Rated Outlet Water Temperature on the heating coil — and today these aren't cross-checked against `Sizing:Plant`'s Design Loop Exit Temperature and Loop Design Temperature Difference. `Coil:Cooling:Water`'s temperature fields default to Autosize, and left that way, EnergyPlus derives them from `Sizing:Plant` through the same reconciliation described above. `Coil:Heating:Water`'s Rated Inlet Water Temperature and Rated Outlet Water Temperature, by contrast, default to fixed values (82.2°C/71.1°C, a legacy high-temperature hot-water rating convention) regardless of what the coil's plant loop is actually sized for — if a hot-water loop runs at a different supply temperature or ΔT, update these two fields to match by hand. Coordinating a coil's rated/design values with its plant loop's sizing inputs automatically is a candidate for a future EP3 enhancement; for now, it's a manual step.

One asymmetry worth knowing if you're choosing between Coincident and NonCoincident: `Sizing:System`'s Type of Zone Sum to Use works as described with no additional setup. `Sizing:Plant`'s equivalent **Sizing Option** field, when set to Coincident, also requires `SimulationControl`'s **"Do HVAC Sizing Simulation for Sizing Periods"** to be turned on — per that field's description — because plant-level coincident sizing draws on EnergyPlus's iterative

HVAC Sizing Simulation feature in a way system-level coincident sizing does not.

4.5 Autosizing

Many `Sizing:System` fields — and many fields elsewhere in EP3, on fans, coils, chillers, and pumps alike — default to the literal string **Autosize** (or, for a small number of fields, **Autocalculate**). For a field with that default, leaving it blank is what tells EnergyPlus to size that value itself during the sizing simulation, rather than using a value you typed; it's an ordinary numeric field, not a separate mode you toggle. This behavior belongs specifically to fields whose default is Autosize/Autocalculate — leaving a different, non-autosizable field blank doesn't trigger it.

4.6 Checking your work

Two ways to confirm `Sizing:System`'s values are taking effect as intended, both covered in full in *(see "Model Troubleshooting and Quality Assurance")*:

- The Validation dialog's HVAC tab compares each loop's design temperature(s) here against its `SetpointManager` setpoint(s) (§8) and flags a mismatch — for example, if you sized the system for a 12.8°C central cooling supply air temperature, but a setpoint manager on the same loop is driving the supply node to 15°C.
- EP3 already configures every new model to request the full EnergyPlus sizing-report set (Component Sizing Summary, Coil Sizing Details, HVAC Sizing Summary, and others) and defaults the results viewer to display the Component Sizing Summary and the "Time Setpoint Not Met" table as soon as results are opened — *(see "Viewing and Interpreting Results")* covers reading them.

---

5. Terminal Units: From Airloop Branch to Zone

`AirTerminal:SingleDuct:VAV:Reheat` and `AirTerminal:SingleDuct:VAV:NoReheat` are the two most common terminal units on a single-duct VAV airloop. Both expose exactly two connectable nodes — an air-inlet and an air-outlet — which is what you wire to the demand-side branch structure (§3.1). Everything else about the object lives behind those two nodes.

Terminal-based demand-controlled ventilation. Both terminal types carry a **Design Specification Outdoor Air Object Name** field referencing a `DesignSpecification:OutdoorAir` object, as do `AirTerminal:SingleDuct:ConstantVolume:NoReheat`, `AirTerminal:SingleDuct:ConstantVolume:C` `AirTerminal:DualDuct:VAV`, and `AirTerminal:DualDuct:VAV:OutdoorAir`. `Sizing:Zone` references the same object type to feed its design-day sizing calculation (*(see "Internal Loads, Space Types, and Zone Types", §4)*); the terminal's reference does something different — at every simulation timestep, EnergyPlus computes the zone's current outdoor air requirement from the referenced object and opens the terminal's damper past whatever the thermal load alone calls for, whenever needed to deliver the minimum outdoor air flowrate. This lets ventilation track occupancy zone by zone through each terminal's damper position, as an alternative to a single system-level `Controller:MechanicalVentilation` managing one outdoor air fraction for the whole airloop. Field-by-field detail, including how the two mechanisms interact when both are present on the same system, is *(see "Outdoor Air, Ventilation, and Energy Recovery", §4)*.

The reheat coil on a VAV Reheat terminal is a swappable sub-component, using the same composite/sub-component pattern *(see "Zone-Level HVAC Equipment and Zone Sizing")* covers

in full: the terminal's damper is the Reference class, and the reheat coil (`Coil:Heating:Water`, `Coil:Heating:Electric`, `Coil:Heating:Fuel`, or `Coil:Heating:Steam`, depending on what you pick) is a sub-component slot. The coil's inlet is wired internally to the damper's outlet, and the coil's outlet is aliased to the terminal's air-outlet node — neither of those internal connections is ever offered as something you can click or rewire; only the coil's water/steam-side nodes (for a hydronic reheat coil) are externally connectable, into a hot-water or steam `PlantLoop`.

The `ZoneHVAC:AirDistributionUnit` wrapper every terminal unit needs in EnergyPlus is generated automatically the moment you assign the terminal to a zone (below) — you never see or edit it directly.

Zone assignment itself — connecting the terminal you've just wired on the airloop side to the zone it should serve — happens through the Select Zone Conditioning Equipment dialog, already covered in *(see "The Model Wizard", §9)* and *(see "The HVAC Outliner and Creating HVAC Systems", §2.2)*. This chapter's job stops at noting that this dialog is where the airloop side and the zone side of a terminal unit come together — the mechanics of the dialog itself aren't repeated here.

5.1 Advanced: `AirTerminal:SingleDuct:Mixer`

`AirTerminal:SingleDuct:Mixer` combines a **primary** airstream — treated ventilation air fed centrally from an `AirLoopHVAC` branch, typically through a zone splitter or supply plenum — with a **secondary** airstream: recirculated air conditioned by a separate zone-level unit. It's the object behind a "DOAS plus local unit" design: a zone gets its ventilation air centrally, but its heating/cooling from a local fan coil, PTAC/PTHP, unit ventilator, water-to-air heat pump, or VRF terminal.

Rather than a standalone object you place from the Components dialog, you enable it on one of six supported `ZoneHVAC` composite types (`ZoneHVAC:FourPipeFanCoil`, `ZoneHVAC:PackagedTerminalAirConditione` `ZoneHVAC:PackagedTerminalHeatPump`, `ZoneHVAC:UnitVentilator`, `ZoneHVAC:WaterToAirHeatPump`, and `ZoneHVAC:TerminalUnit:VariableRefrigerantFlow`): with the host unit selected, choose **"Change Sub Component Types"** in its Selected Object Dialog, then pick `AirTerminal:SingleDuct:Mixer` from the pop-up. Doing so replaces that unit's outdoor-air mixer sub-component — the host unit no longer draws separate outdoor air once a primary DOAS airstream supplies it instead.

Mixer Connection Type decides where the two airstreams combine:

- **InletSide** (EP3's default) mixes the primary and secondary airstreams together first, then feeds the combined stream into the host `ZoneHVAC` unit for conditioning.
- **SupplySide** has the host unit condition the secondary (recirculated) airstream alone, and only mixes in the primary (DOAS) airstream afterward, immediately before the combined supply reaches the zone.

Because it's a sub-component rather than an independent piece of equipment, it never appears as a separate row in the Select Zone Conditioning Equipment dialog — it rides along with its host unit as a single logical piece of zone equipment. Its primary air inlet still needs to be wired to an `AirLoopHVAC` branch like any other node; leaving it unconnected keeps EP3 from exporting it at all, and produces an explicit warning rather than a silent failure.

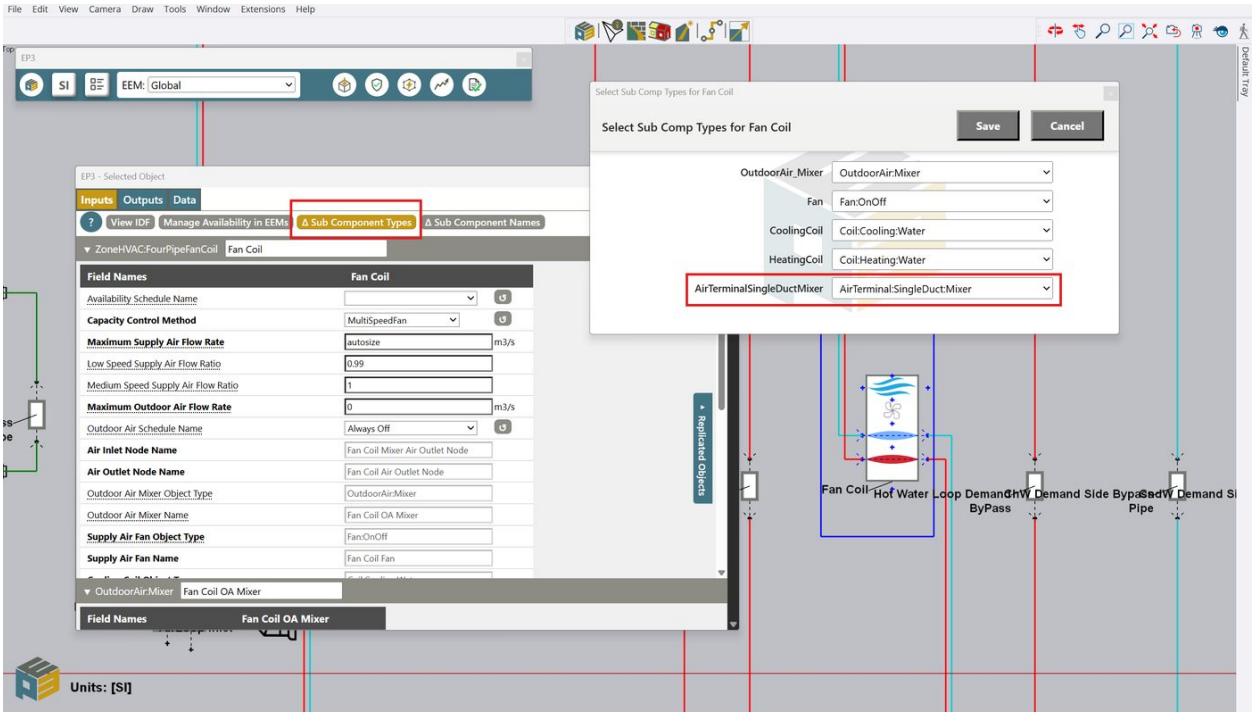


Figure 101: Selecting AirTerminal:SingleDuct:Mixer from the "Change Sub Component Types" pop-up on a supported ZoneHVAC composite.

6. Plenums and Supply/Return Paths

`AirLoopHVAC:SupplyPlenum` and `AirLoopHVAC:ReturnPlenum` are ordinary placeable components — pick one from the Components dialog (*(see "The HVAC Outliner and Creating HVAC Systems", §6)*) and wire it onto the demand side with the Draw Branch tool like any other equipment.

A **Supply Plenum** sits between a supply path and one or more terminal units, standing in for a physical plenum space (typically a ceiling return-air plenum used on the supply side, or a raised-floor supply plenum) that's doing the air distribution instead of ordinary ductwork. Its **Zone Name** field — the zone representing that physical plenum space — only offers unconditioned zones, so model the plenum space itself as an unconditioned zone before assigning it here.

A **Return Plenum** works the same way on the return side, with the same unconditioned-zone requirement for its Zone Name field, and — like the loop itself (§3.2) — carries an explicit list of which zones' return air routes through it specifically, rather than that list being inferred from anything physically wired to it.

This manual doesn't go further into plenum modeling than this; no other chapter currently covers the topic in more depth.

---

7. The Outdoor Air System on the Branch

An `AirLoopHVAC:OutdoorAirSystem` — built around an `OutdoorAir:Mixer` — typically sits early in the branch order, mixing outdoor air with return air before the main coils see either. *(see "Air-Side System Fundamentals", §5)* covers the mixed-air path conceptually, and *(see "Fans and Fan*

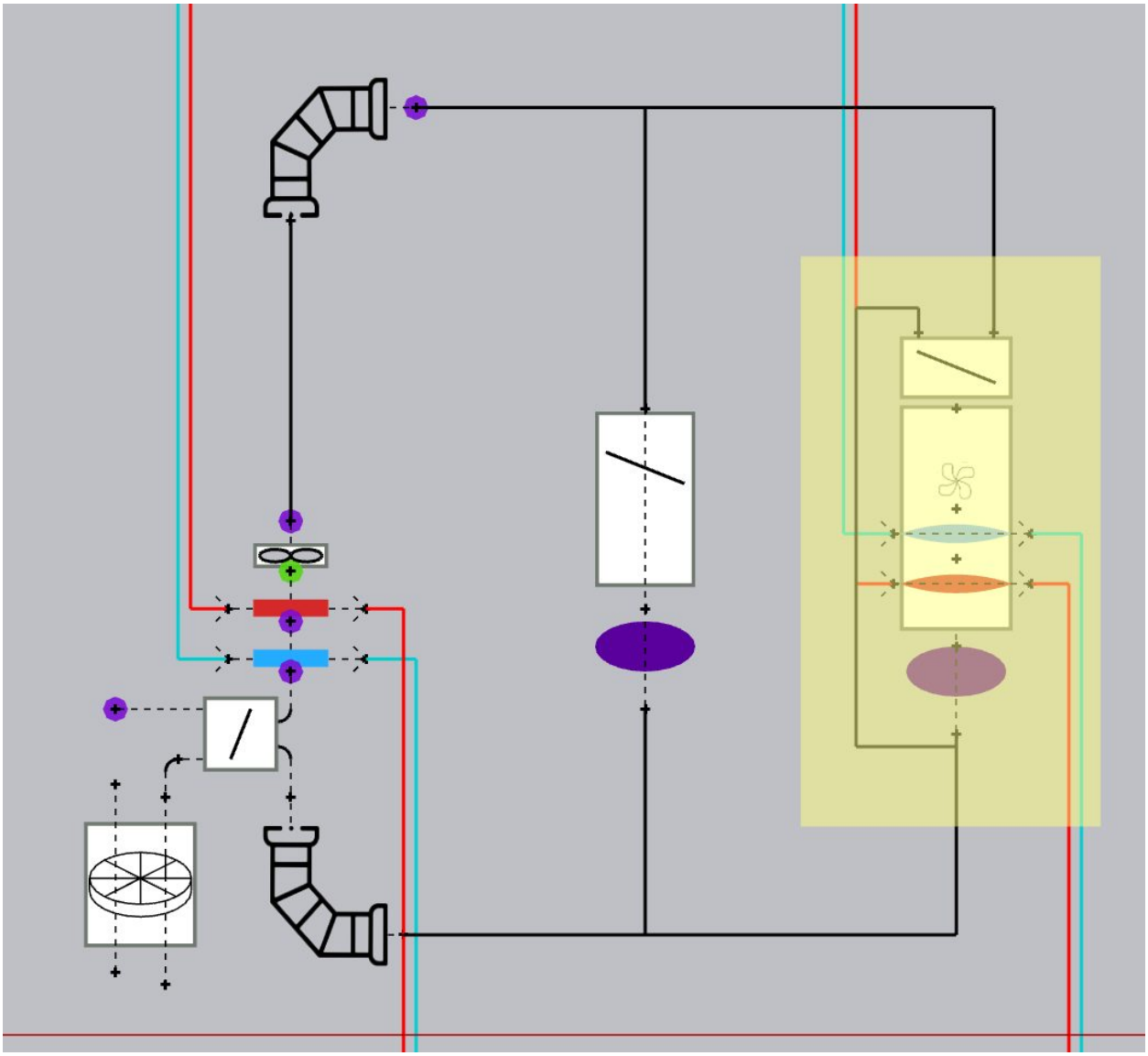


Figure 102: InletSide Mixer Connection Type: the primary and secondary airstreams mix before entering the host ZoneHVAC unit.

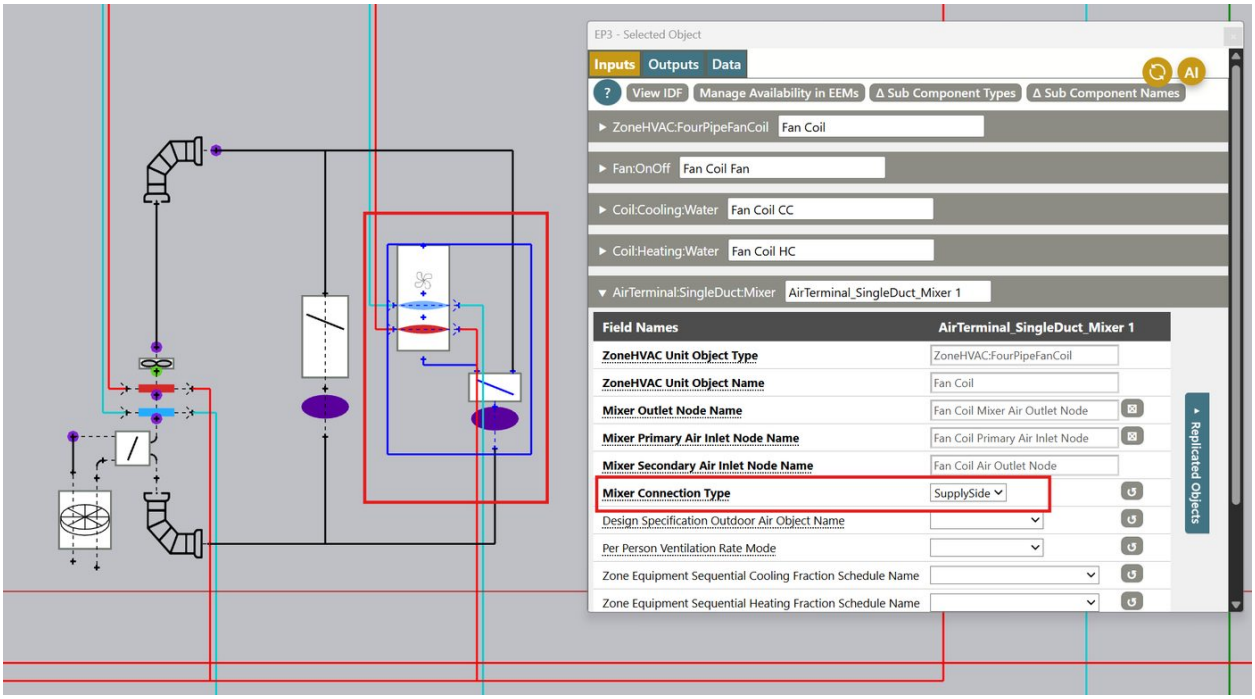


Figure 103: SupplySide Mixer Connection Type: the host ZoneHVAC unit conditions the secondary airstream alone, and the primary airstream mixes in afterward.

Power", §3.1) shows a concrete branch order (return fan, then the outdoor air system, then heating and cooling coils, then a draw-through supply fan).

The outdoor air system maintains a distinct internal supply/exhaust branch structure — apart from the loop's main branch hash — with a dedicated controller. Its fields, its controller's fields, and outdoor-air-specific control strategies (economizer, demand-controlled ventilation, energy recovery) are all covered in *(see "Outdoor Air, Ventilation, and Energy Recovery")*; this chapter only establishes that it exists as a branch component like any other.

---

8. Setpoint Managers and Availability Managers: How They Attach to a Loop

EnergyPlus generally doesn't let you tell a piece of equipment directly, "hold 55°F." Instead, a target value gets assigned to a specific point in the airflow or waterflow path — a node — and whatever equipment can affect the condition at that point works to bring it there. A `SetpointManager` is the object that decides what that target should be and writes it onto the node: at the start of each simulation timestep, it looks at some source of information — a fixed schedule, what a zone currently needs, the outdoor air temperature, or the condition at another node entirely — and calculates a setpoint (usually a temperature, though some types work in humidity ratio) for one or more nodes elsewhere in the loop. EnergyPlus Input Output Reference, Group – Setpoint Managers covers the full mechanism and every type in exhaustive detail; this section only builds the concept far enough to support the rest of this chapter.

This is what makes §2.1's "supply side meets a load" concrete. A node with no `SetpointManager` targeting it simply has no target — nothing tells nearby equipment what condition to aim for. A

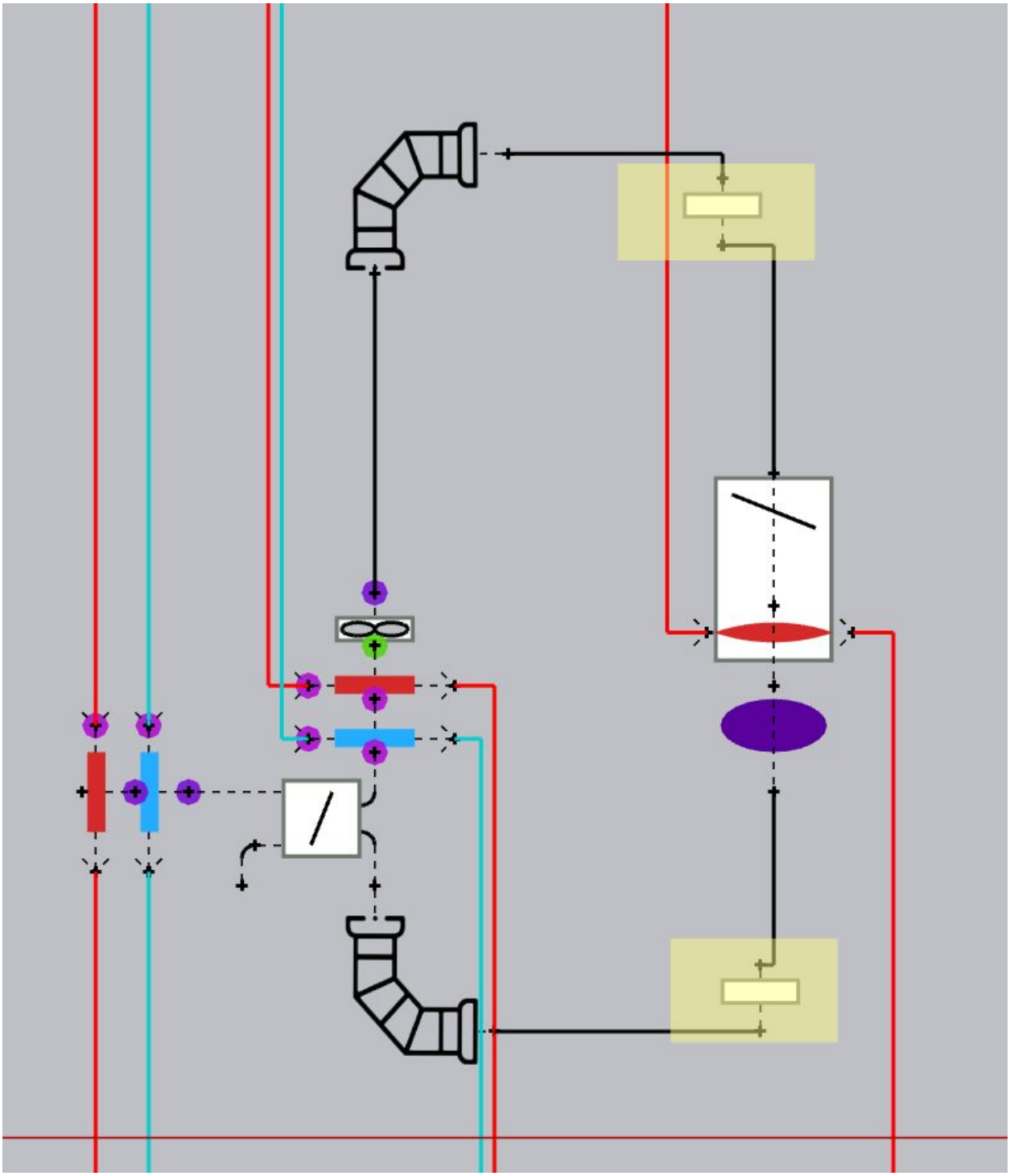


Figure 104: An AirLoopHVAC with a Supply Plenum between the supply path and its terminal units, and a Return Plenum between the zones and the return path.

node that does have one gives whatever equipment sits there something to work toward: a cooling coil, for example, modulates its own water flow to bring its outlet node to whatever temperature a `SetpointManager` has assigned there. Placing the right equipment on a branch (§3.1) gets air or water flowing through it; a `SetpointManager` is what tells that equipment what condition to actually produce.

A `SetpointManager` or `AvailabilityManager` attaches to a loop differently than a fan or a coil does. Rather than being placed and wired with the Draw Branch tool, both are created through the generic object-library "Add" mechanism also used for schedules and curves, then attached to a loop by referencing a node — or, for the convenience path below, a zone/terminal pair — rather than by sitting on a branch.

8.1 Setpoint managers

Some `SetpointManager` types only make sense on an `AirLoopHVAC` — `SetpointManager:SingleZone:Heating/Co` `SetpointManager:MixedAir`, `SetpointManager:Warmest/Coldest`, and the multizone humidity/temperature-average variants among them — while a few others (`SetpointManager:OutdoorAirReset`, `SetpointManager:CondenserEnteringReset`, and the return-temperature variants) only make sense on a `PlantLoop`. `SetpointManager:Scheduled` and a handful of others work on either. *(see "Central Plant Equipment")* covers the plant-only types; the airloop-only and either-loop types are this chapter's job, covered below.

Among the airloop-only types, two families answer different questions. The single-zone types — `SetpointManager:SingleZone:Heating/Cooling/Reheat` and the `SetpointManager:SingleZone:Humidity:Mi` pair — control to the needs of one specific zone and terminal unit you pick (the Select Control Terminal convenience below is built for exactly these). `SetpointManager:Warmest` and `SetpointManager:Coldest`, along with the `SetpointManager:MultiZone:*:Average` and `SetpointManager:MultiZone:Humidity:*` variants, instead respond to every zone the loop serves at once: `Warmest/Coldest` reset the central supply air temperature to whatever a VAV system's neediest zone currently requires, without you designating any single zone as the control point.

Any piece of equipment with active temperature control needs one of two things: a `SetpointManager` targeting it, or a wrapper — an `AirLoopHVAC:Unitary*` object — that provides its own internal controls instead of relying on the loop's (§2.1). `Coil:Heating:Water` and `Coil:Cooling:Water`, placed as standalone branch components, are common examples of equipment that need a `SetpointManager` telling it what temperature to hold; neither coil can provide conditioning without one.

`SetpointManager:MixedAir` is a useful example of why a setpoint sometimes gets set at one node based on conditions at an entirely different, downstream node. It lets you set a setpoint upstream of a fan — typically at a cooling coil's outlet node, immediately before a draw-through fan — based on the setpoint you actually want held further downstream, at its Reference Setpoint Node Name. Using the fan's own inlet and outlet nodes, it backs out the temperature rise the fan itself adds to the airstream, so the coil is driven to a slightly colder condition than the final target — exactly cold enough that once the fan's heat is added, the air downstream lands on the intended supply air temperature. Without this adjustment, a fan's own heat addition would leave the final supply air warmer than intended, even with a correctly performing coil.

`SetpointManager:Scheduled` is the simplest and most common type on either loop: it drives a node straight to whatever value a schedule provides, at every timestep, with no zone or outdoor-air

feedback involved. It's often what actually supplies the "reference" setpoint that a downstream-looking type like `SetpointManager:MixedAir` reads and adjusts for upstream.

`SetpointManager:OutdoorAirReset` is a common energy-saving strategy, also usable on either loop type: it resets a setpoint linearly between two points you define — **Setpoint at Outdoor Low Temperature** at **Outdoor Low Temperature**, and **Setpoint at Outdoor High Temperature** at **Outdoor High Temperature**. A heating supply air temperature set up this way runs warmer on the coldest days and cooler on milder ones, cutting reheat energy on days that don't need the full design temperature rise.

Attaching most setpoint manager types means clicking a node's name directly — hovering over it swaps the cursor to a bullseye icon, the same picker interaction used elsewhere in EP3 for picking a point in the 3D model. The single-zone types add a friendlier **"Select Control Terminal"** button instead, letting you pick a zone and its terminal unit from a dropdown rather than clicking raw nodes — EP3 derives the correct control-zone and zone-node fields from that pick automatically.

Once a node already has a setpoint manager attached, hovering over it in the 3D view shows that manager's name and the node's type — a quick way to confirm what's controlling a given point without opening any dialog.

8.2 Availability managers

An `AvailabilityManager` attaches to a loop indirectly: you create one or more `AvailabilityManager` objects, bundle the ones you want (in order) into an `AvailabilityManagerAssignmentList`, and then pick that list in a dropdown field on the loop's dialog. There's no "add availability manager to this loop" button — the loop just references whichever list you've built. *(see "Controllers and Availability Managers")* covers building that list and each availability manager type in depth.

8.3 EP3's Night-Cycle Control toggle

`AvailabilityManager:NightCycle` is a standard availability manager type — buildable through §8.2's standard tools like any other. Because it's common enough to want on nearly every airloop, EP3 also builds a shortcut directly into the `AirLoopHVAC` dialog: a **"Night-Cycle Control?"** Yes/No field that doesn't exist on the real EnergyPlus object at all. Setting it to Yes:

- Auto-creates an `AvailabilityManager:NightCycle` object, embedded and edited inline in the same `AirLoopHVAC` dialog rather than in the separate availability-manager library — pre-filled with an Always On applicability schedule, Control Type set to `CycleOnAny`, and a Thermostat Tolerance of 0.2.
- Is mutually exclusive with manually picking an `AvailabilityManagerAssignmentList` (§8.2) — the toggle only appears when no list has already been chosen by hand.
- Auto-detects your supply fan's Availability Schedule Name and uses it as the Night-Cycle object's Fan Schedule Name, unless you override that field.
- Exports as an ordinary `AvailabilityManager:NightCycle` object plus a synthesized `AvailabilityManagerAssignmentList` — the resulting IDF looks exactly like the manual path in §8.2, even though you never built either object by hand.

The auto-detected fan schedule assumes one supply fan to detect a schedule from. For an airloop with multiple fans running different Availability Schedule Names, build the `AvailabilityManager:NightCycle` object yourself through §8.2's standard tools instead, so

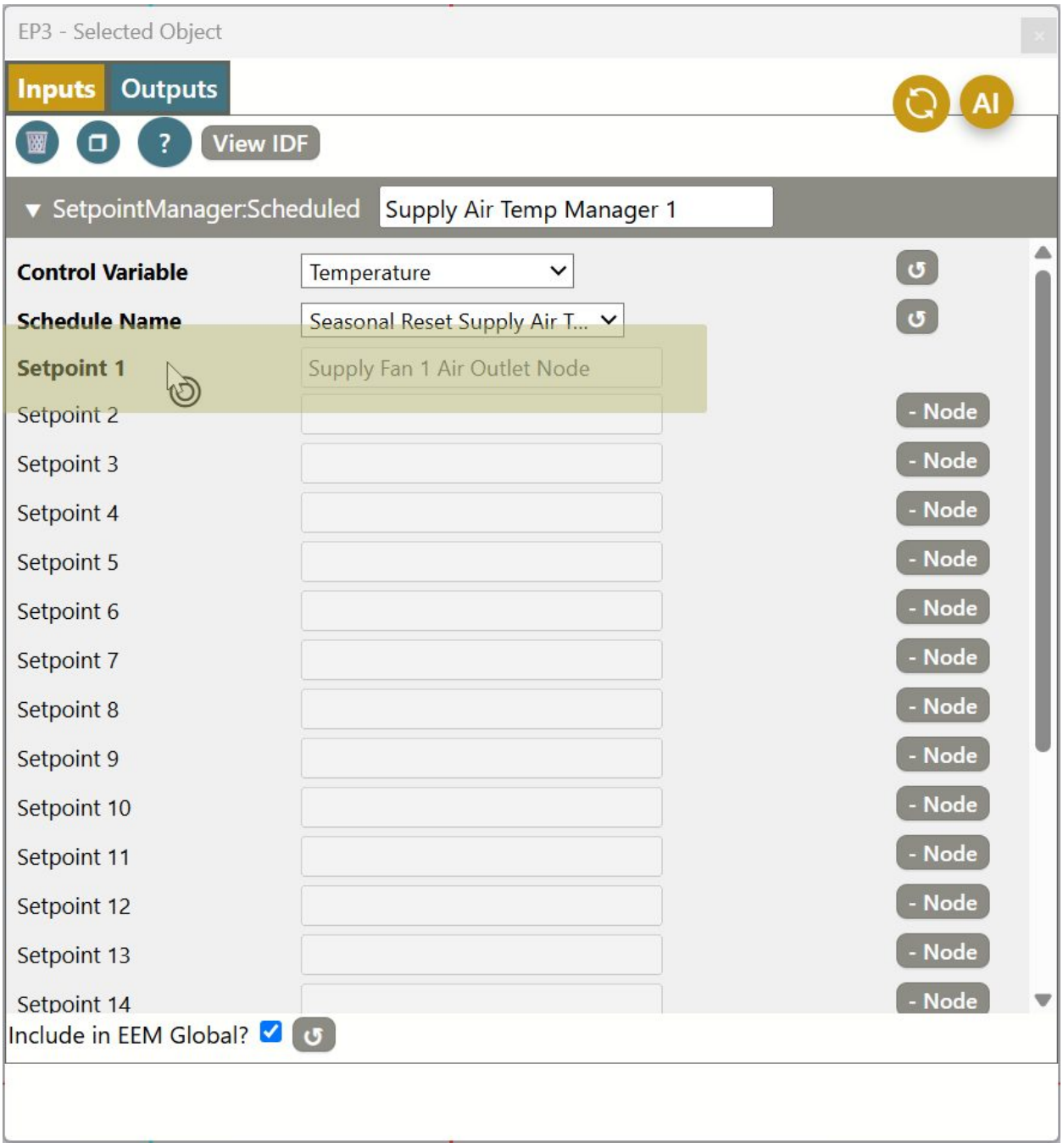


Figure 105: The bullseye cursor, shown whenever a node is available to pick.

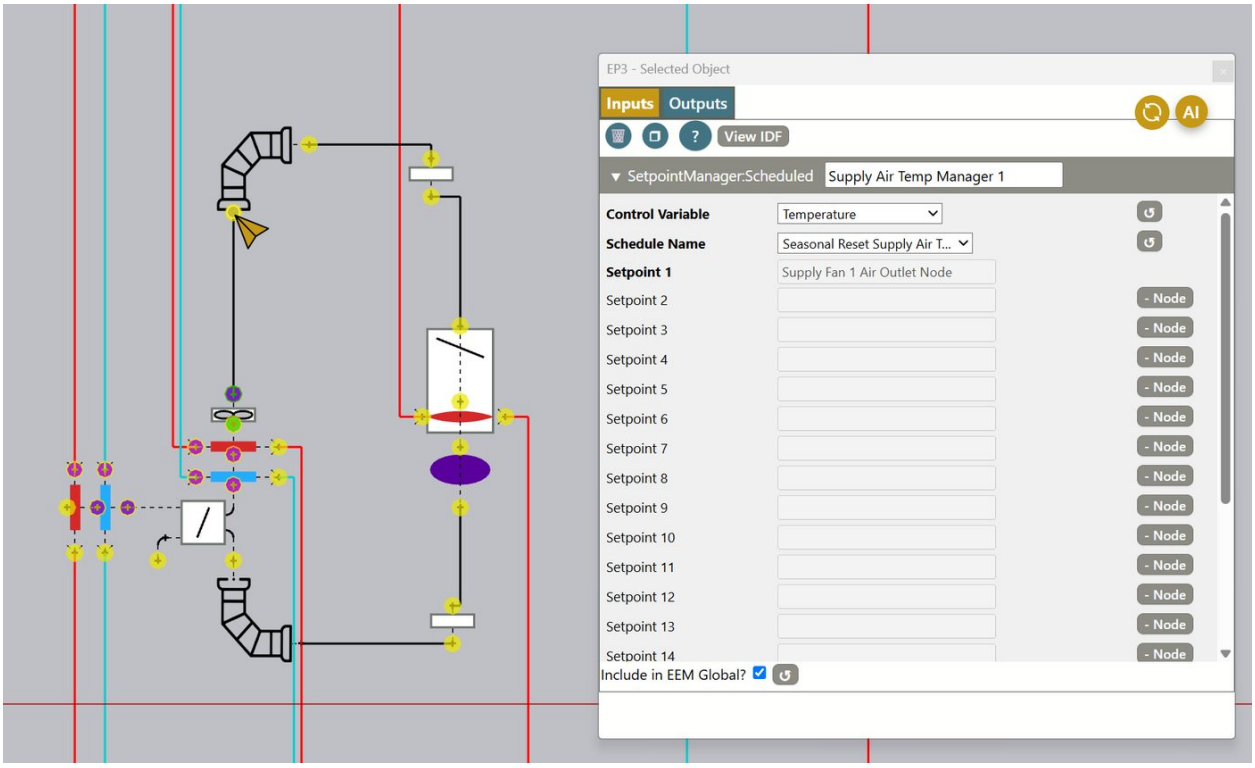


Figure 106: A SetpointManager's dialog, listing the node fields available to click and assign.

you can set its Fan Schedule Name to whichever schedule actually reflects the loop's intended behavior.

8.4 Checking your work

The Validation dialog includes checks specific to `AirLoopHVAC` configuration, including the setpoint-vs-design-temperature comparison introduced in §4.6.

It also flags any node controlled by more than one `SetpointManager`. This isn't automatically an error — EnergyPlus doesn't simply pick one setpoint manager per node and ignore the rest, so two managers sharing a node is a legitimate configuration as often as it's a mistake. Severity escalates to **fail** when a humidity setpoint manager (`SetpointManager:SingleZone:Humidity:Minimum/Maximum`, the `MultiZone:Humidity:*` family) is one of the two or more sharing the node — EnergyPlus is more restrictive about humidity setpoint managers sharing a node than temperature ones — or when a fixed-value `SetpointManager:Scheduled` is paired with another manager on the same node, the case most likely to actively fight whatever else is there. Any other combination of multiple setpoint managers on a shared node is a **warning**, flagged so you check that they don't conflict, without assuming they do.

A related check is specific to `SetpointManager:MixedAir` (§8.1): since it has no setpoint value of its own and simply copies whatever setpoint already exists at its Reference Setpoint Node, a reference node with no setpoint manager of its own controlling it — nothing for `MixedAir` to copy at runtime — is flagged as a **fail**, a broken control chain rather than a cosmetic gap.

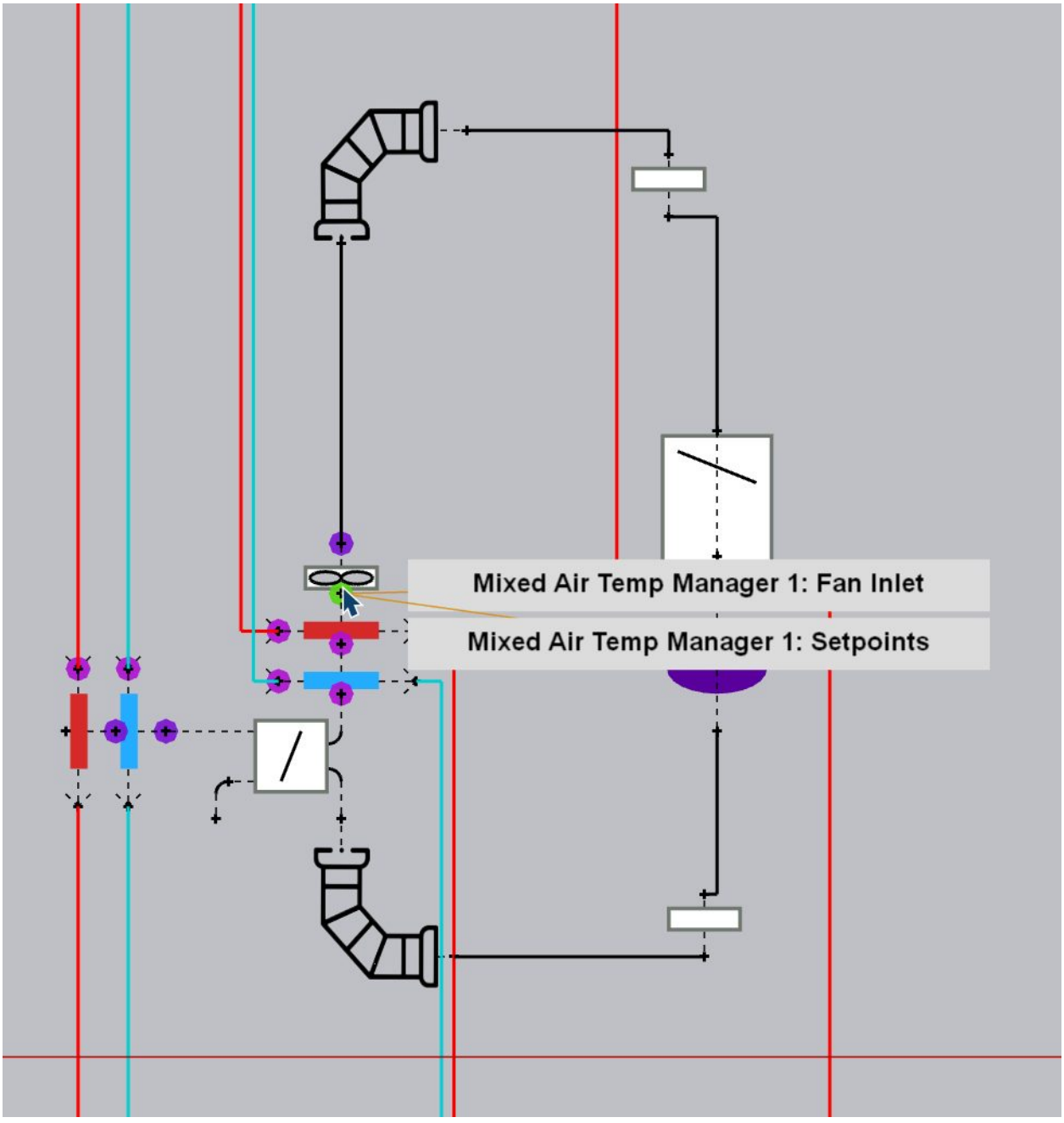


Figure 107: Hovering over a node with a setpoint manager already attached shows that manager's name and the node's type.

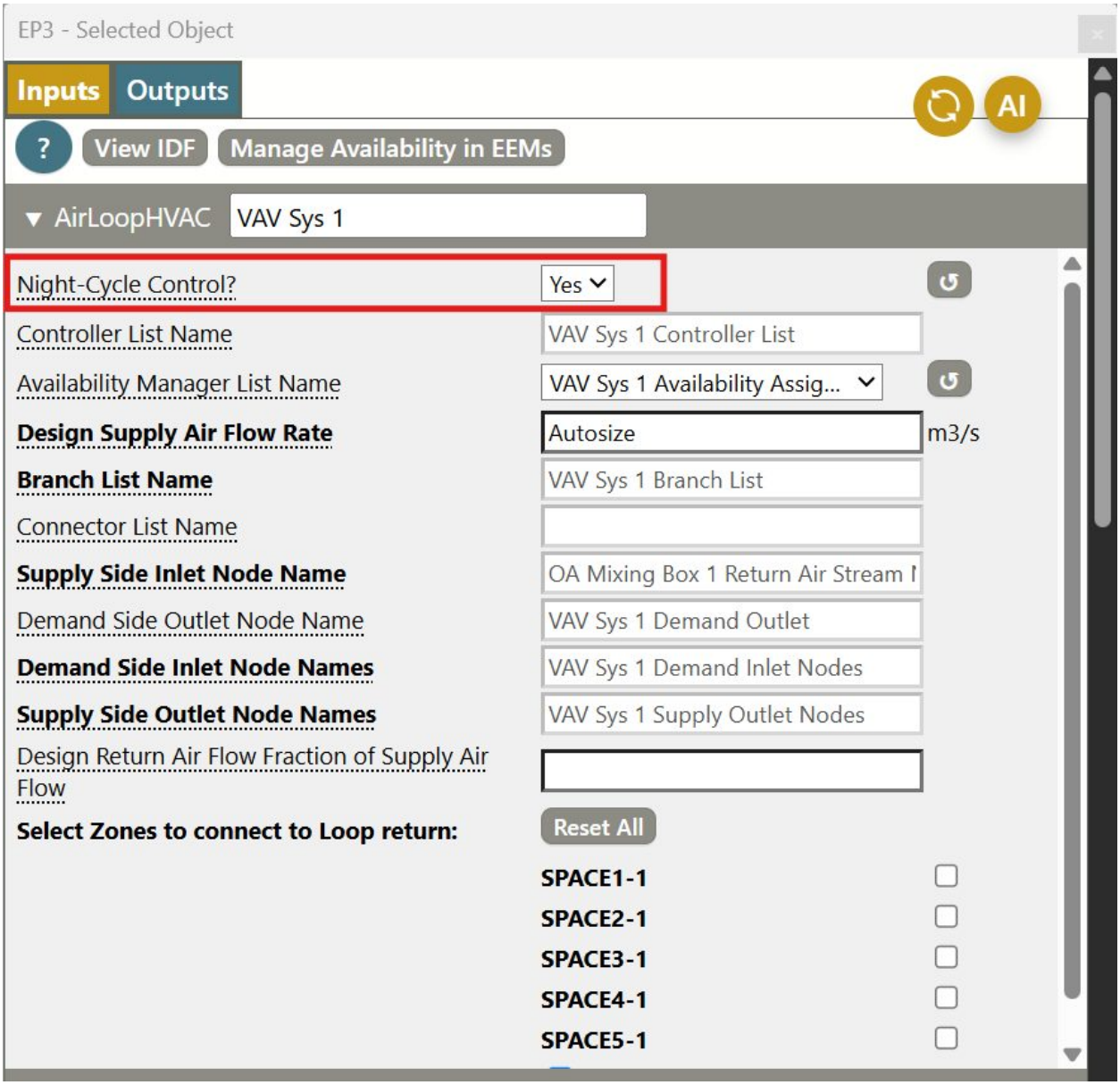


Figure 108: The AirLoopHVAC dialog's "Night-Cycle Control?" field set to Yes.

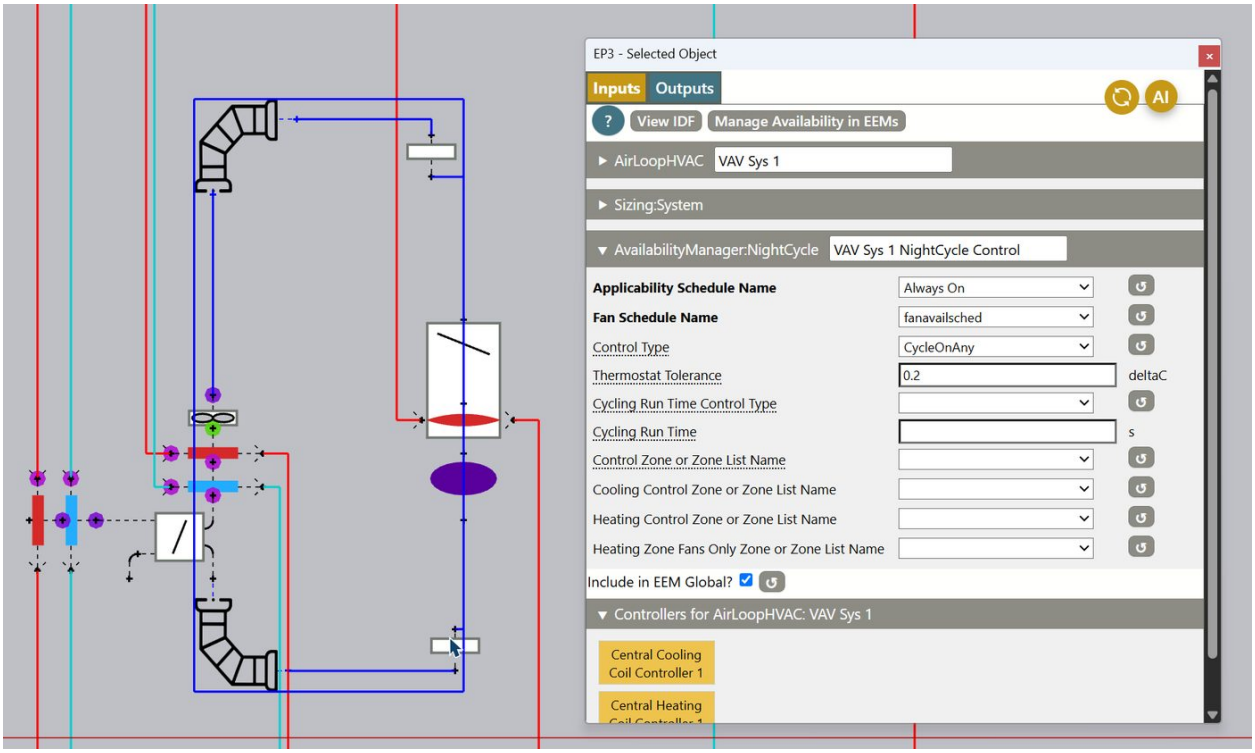


Figure 109: The embedded AvailabilityManager:NightCycle fields shown once Night-Cycle Control is enabled.

Full treatment: *(see "Model Troubleshooting and Quality Assurance").*

---

9. Checking Your Work: `NodeReport`

`NodeReport` applies equally to `AirLoopHVAC`, `PlantLoop`, and `ZoneHVAC` equipment rather than being specific to any one of them, so it's covered in full in *(see "Model Troubleshooting and Quality Assurance")* rather than here. It's worth introducing briefly in this chapter, because so much of what this chapter covers is easiest to confirm by directly inspecting a node rather than trusting the dialogs alone.

Mechanically: create a `NodeReport` object the same way you'd create a setpoint manager, then click a node's name in its dialog — the bullseye cursor (§8.1) shows it's clickable — and select the node you want from the ones available in the 3D view, the same mechanism setpoint managers and sensor-based availability managers use. Choose which System Node variable to report — temperature, mass flow rate, humidity ratio, setpoint temperature, and several others are available — and at what frequency. The result is an ordinary `Output:Variable` (with an optional companion `EnergyManagementSystem:Sensor`, if you name one) that shows up in your results the same way any other output variable would — *(see "Viewing and Interpreting Results")* covers viewing that data once you've simulated.

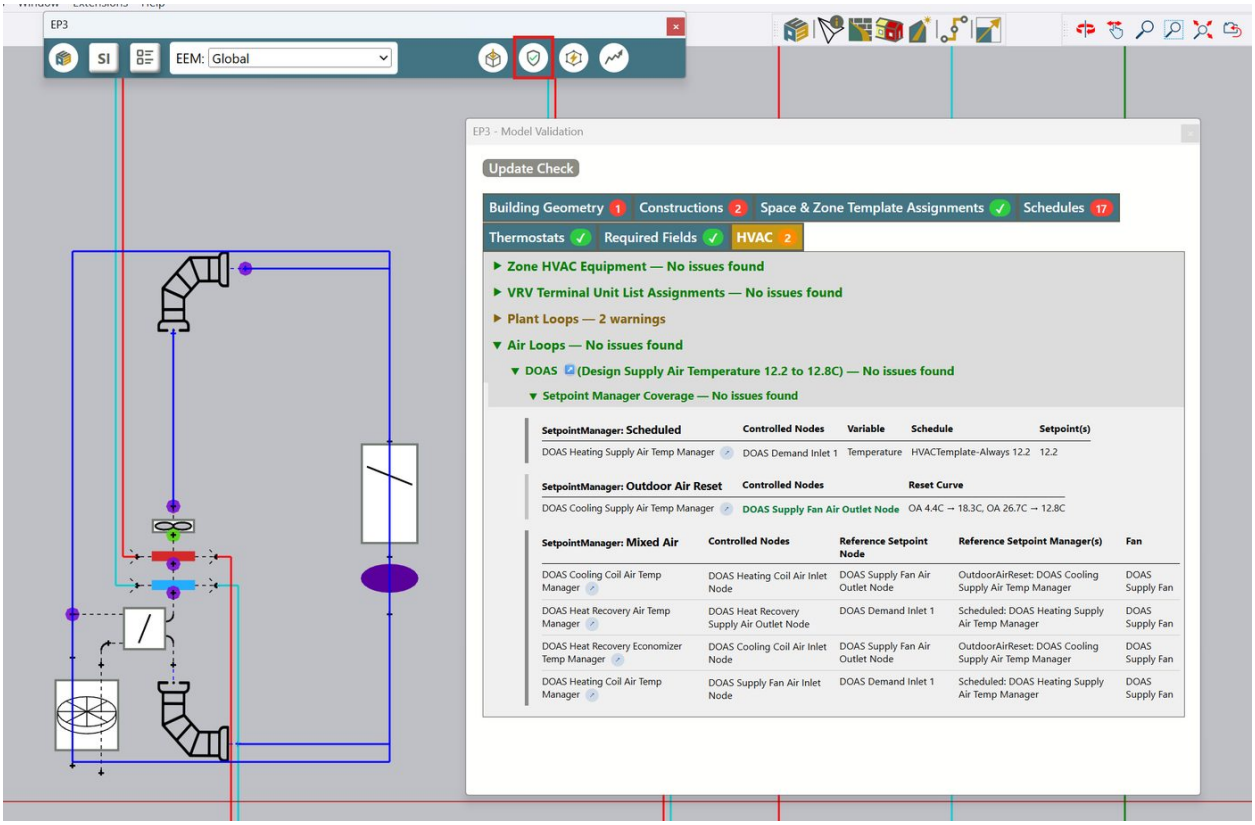


Figure 110: The model Validation dialog, showing checks specific to AirLoopHVAC configuration.

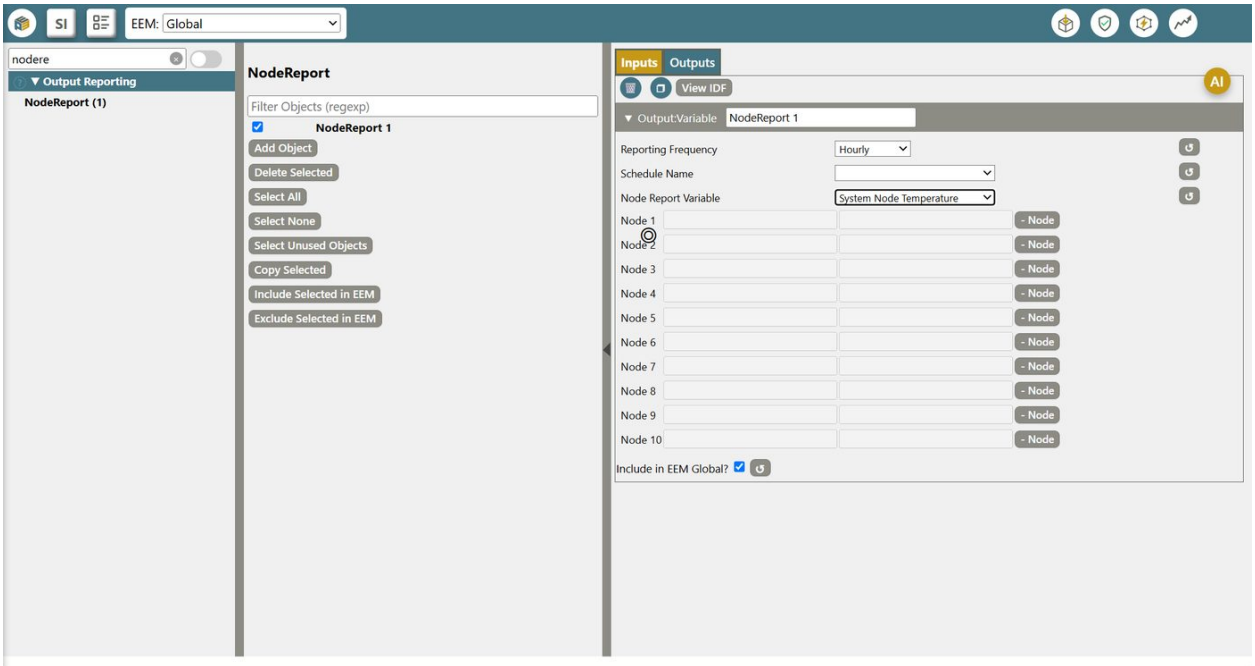


Figure 111: The NodeReport dialog, where clicking a node's name lets you pick its location the same way a setpoint manager's sensor node is picked.

10. Replicating an `AirLoopHVAC`

A single `AirLoopHVAC` you've built once in the 3D model can stand in for many physical airloops at export time, through the **AirLoop Replication** field on the loop's dialog (also prompted immediately after importing a template system, via the Wizard). Four methods are available:

- **One Per Building** (the default) — no replication; the loop exports exactly once.
- **One Per Zone** — one exported instance per zone the loop serves.
- **One Per Floor** — one exported instance per building floor, intersecting each floor's zones against the zones this loop serves; a floor with none of this loop's zones simply doesn't get an instance.
- **One Per Group of Zones** — one exported instance per user-created `GroupOfZones` object (built the same way you'd build a `ZoneList`) that overlaps the loop's zones.

Changing this field doesn't refresh the dialog on its own — after picking a new replication method, refresh (or reopen) the `AirLoopHVAC`'s dialog to see the resulting per-replicate controls.

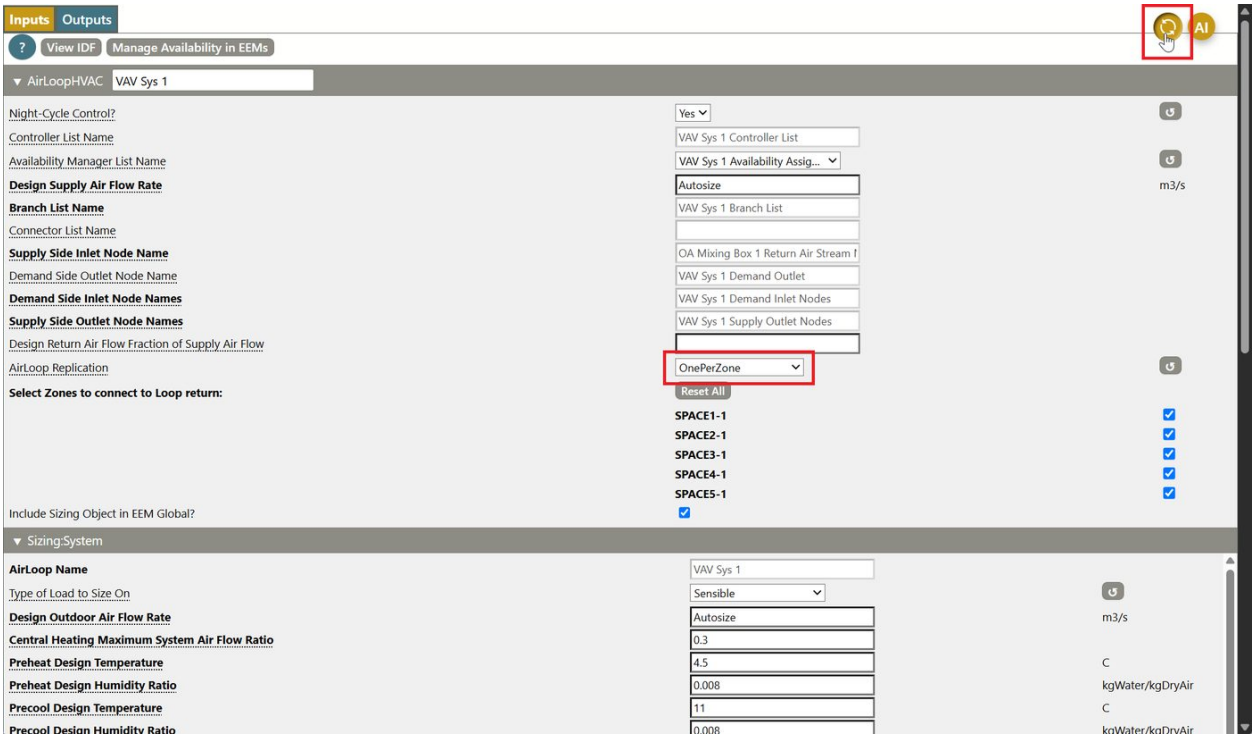


Figure 112: The AirLoop Replication field set to One Per Zone.

Nothing is physically duplicated. There's still exactly one drawn `AirLoopHVAC` and one branch structure in the 3D model. A replicate method only changes how many times, and under what names, EP3 re-serializes that same structure to IDF — once per replica, each with a distinct name prefix and reference id, computed entirely at export time.

Which zones get replicated is decided by §3.1's wiring, rather than by the replicate method — the replicate method only decides how that already-determined zone set gets *partitioned*. One Per Zone splits it one zone at a time; One Per Floor and One Per Group of Zones split it by intersecting it against a `FloorPlate`'s or a `GroupOfZones`' membership. If the airloop doesn't serve any zones on a particular floor (or in a particular group), no replicate is created for that floor or

group.

Eligibility. A loop can only be replicated if it's single-branch (§2) with no supply or return plenum (§6), has exactly one terminal unit on its demand side, and hasn't had its return-zone list manually overridden (§3.2) — outside those conditions, the field is not offered.

Per-replicate overrides. Any field on the loop or its branch components can be given a different value for one specific replicate — one floor's terminal running a different reheat setpoint than the rest, for instance — while every other replicate inherits the shared value, the same inheritance mechanism EEMs use elsewhere in EP3.

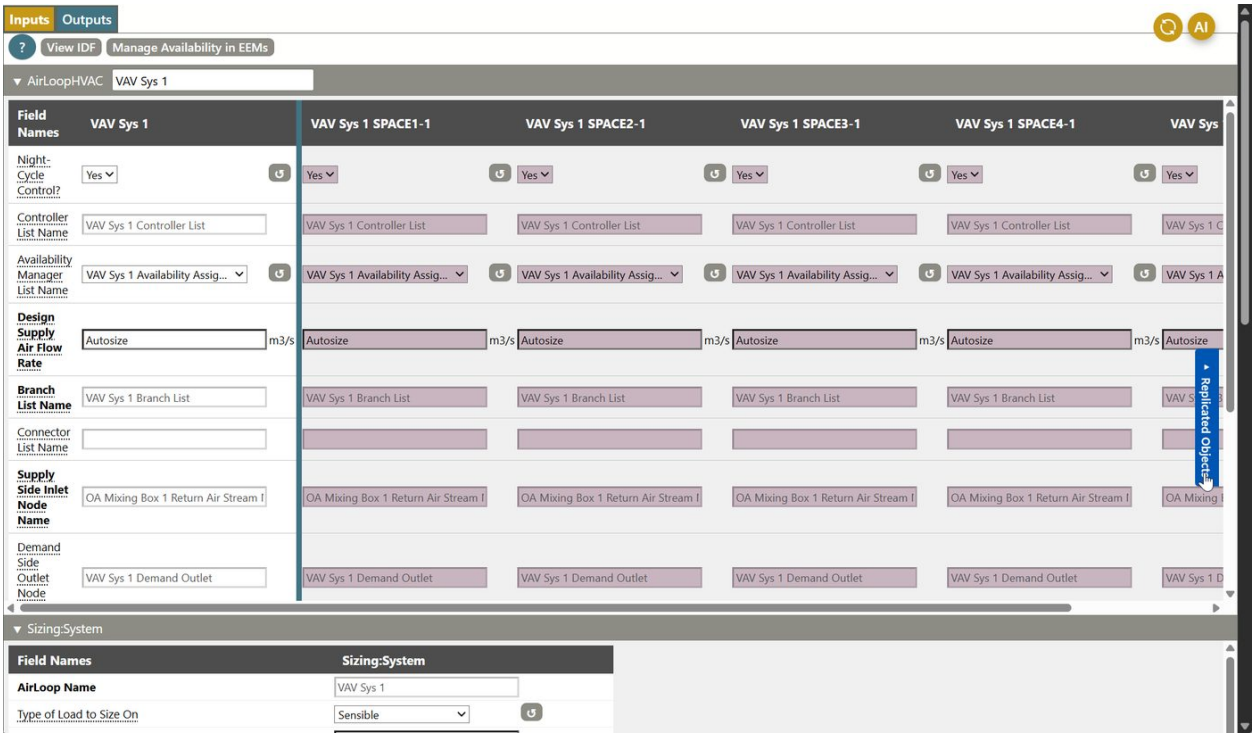


Figure 113: A replicated object's fields, with a pull tab that appears on mouse hover for setting a per-replicate override.

Naming. Each replicate's exported name takes the form `"<replicate name> <object's name>"` — for example, `"Floor 2 VAV-1"` for a One-Per-Floor loop named `VAV-1` — and that same prefix propagates to every node, branch, and node-list name generated for that replicate's export pass. Because names are computed at export time rather than stored, there's no separate dialog listing every replica individually; inspect the result via the View IDF option.

Setpoint and availability managers replicate correctly along with the loop — each replicate resolves its own node reference correctly, and the auto-detected fan schedule (§8.3) picks up the specific fan serving that replicate rather than a single shared value.

---

11. Worked Example: Building a Single-Zone VAV Airloop

1. **Open the HVAC Outliner** (*(see "The HVAC Outliner and Creating HVAC Systems", §2)*) and choose **Add Air Loop**.

2. **Fill in the Create Air Loop prompts**: Name "VAV-1"; Number of Outlets 1; Branch Configuration Single Branch; Dedicated Outdoor Air System? No; Use Demand Controlled Ventilation? No; Select Air Terminal Type → `AirTerminal:SingleDuct:VAV:Reheat`; Cooling Design Supply Air Temperature 12.8°C; Heating Design Supply Air Temperature left blank.
3. **Confirm.** EP3 creates the loop, auto-imports and wires the VAV Reheat terminal and an `OutdoorAir:Mixer` onto the single branch, and seeds `Sizing:System` per §4.2: preheat/precool design temperature and humidity ratio are set to their fixed defaults; the DOAS-specific fields are untouched (you answered No); Central Cooling Design Supply Air Temperature is set to 12.8°C (you typed a value); Central Heating Design Supply Air Temperature is left blank (you didn't).
4. **Add the remaining branch equipment** — a heating coil, a cooling coil, and a supply fan — from the Components dialog, and wire each onto the branch with Draw Branch (§3.1), in a draw-through order: outdoor air system, heating coil, cooling coil, supply fan (*(see "Fans and Fan Power", §3.1)*).
5. **Assign the terminal unit to a zone** through Select Zone Conditioning Equipment (*(see "The Model Wizard", §9)*).
6. **Confirm the return-zone checkbox** (§3.2) already shows your zone checked — this is the default, so no action is needed unless you want to route return air differently.
7. **Add a `SetpointManager:SingleZone:Reheat`** using the Select Control Terminal convenience (§8.1), pointing it at the zone/terminal pair you just assigned.
8. **Leave AirLoop Replication at One Per Building.** With only one zone on this loop, switching to One Per Zone wouldn't change anything — replication only matters once a loop serves more than one instance of whatever you're partitioning by.
9. **Add a `NodeReport`** (§9) on the supply outlet node, reporting System Node Temperature at Timestep frequency, as a quick post-simulation sanity check.
10. **Run the Validation dialog's HVAC tab** (*(see "Model Troubleshooting and Quality Assurance")*) before simulating, to confirm the setpoint manager and the `Sizing:System` design temperature agree.

---

12. Summary and Where to Go Next

This chapter covered the mechanics beneath an `AirLoopHVAC`: branch anatomy and why single-branch should be your default, what a node connection does and doesn't establish (especially on the return side), the loop's `Sizing:System` object and how it interacts with sizing elsewhere in the model, terminal units as the airloop-to-zone boundary, plenums, the outdoor air system's place on the branch, how controls attach to a loop, and replication.

From here: *(see "Central Plant Equipment")* and *(see "Performance Curves: Part-Load and Efficiency Behavior")* cover the individual pieces of equipment this chapter treated only as branch components, the water side these coils connect into (including `Sizing:Plant` and `PlantLoop`-specific `SetpointManager` types); *(see "Controllers and Availability Managers")* covers `AvailabilityManager` control strategy and the `Controller:*` layer in field-by-field depth; *(see "Outdoor Air, Ventilation, and Energy Recovery")* covers the outdoor air system itself; and *(see "Model Troubleshooting and Quality Assurance")* covers verifying all of it — `NodeReport` in full, unmet hours, the Validation dialog, and reading the EnergyPlus `.err` file — before you trust a simulation's results.

End of draft chapter. Outstanding items for the next pass: supply the screenshots/diagrams marked above; resolve the XREF/EXTLINK placeholders once their target chapters exist; confirm the worked example (§11) against a live run.

16. Central Plant Equipment and Plant Loops

1. Overview

This chapter covers `PlantLoop` topology and the equipment that sits on one: boilers, chillers, water-to-water heat pumps, cooling towers and fluid coolers, pumps, water-side coils (from the plant-loop side), `HeatExchanger:FluidToFluid`, `Sizing:Plant`, plant equipment dispatch, and the `SetpointManager` types specific to a `PlantLoop`.

This chapter assumes you already know how to get HVAC objects into a model and wire them together — the Components dialog, the Draw Branch tool, and the branch-connection fluid-type check — from *(see "The HVAC Outliner and Creating HVAC Systems")*, and the composite/sub-component pattern (a Reference class plus swappable sub-component slots) from *(see "Zone-Level HVAC Equipment and Zone Sizing")*. Performance curves and tables are covered generically in *(see "Performance Curves: Part-Load and Efficiency Behavior")*.

Out of scope:

- Water heaters, thermal storage tanks, and solar water heating — *(see "Service Hot Water and Solar Water Heating")*. Those are organizationally separate from central-plant equipment in EP3 (their own system-template pathway), even though a water heater is technically a `PlantLoop`-connectable object.
- Common-pipe / two-way-common-pipe simulation mode is deliberately **not** covered. While EP3 allows the common pipe field and structure to be used, our team has found that using two loops connected by a `HeatExchanger:FluidToFluid` (§10) is more reliable and controllable — common pipe is an older EnergyPlus modeling convention this manual doesn't recommend reaching for.
- Steam loops and `Boiler:Steam` are not covered — not supported in EP3 at this time.
- Field-by-field detail for any object — refer to the EnergyPlus Input/Output Reference. This chapter covers what EP3 exposes, how equipment connects, and where its behavior differs from a plain EnergyPlus object.

---

2. `PlantLoop` Topology: Loop Types and the Select Loop Options Dialog

Creating a `PlantLoop` uses the same Add-from-Outliner mechanism as any other HVAC object — the **Add Plant Loop** button (*(see "The HVAC Outliner and Creating HVAC Systems")*) — opening a dialog titled **Select Loop Options**. It prompts for:

- A **Loop Name** and a **Loop Type** (Heating/Cooling/Condenser, defaulting to Heating).
- A **Design Loop Exit Temperature [C]** and **Loop Design Temperature Difference [deltaC]** — both write directly into the loop's `Sizing:Plant` object (§11); Design Loop Exit Temperature defaults differently depending on the Loop Type you picked (82°C for Heating, 7°C for Cooling, 29°C for Condenser).
- A **SetpointManager Type** dropdown (defaulting to `SetpointManager:Scheduled`) — this seeds the loop's initial setpoint manager immediately, rather than leaving you to add one separately afterward (§13).
- **Operation Scheme Load Range 1 Lower/Upper Limit [W]** (0/1,000,000 defaults) and an **Operation Scheme Operating Schedule** ("Always On" default) — these seed the loop's initial `PlantEquipmentOperation` scheme (§12.1).

- A **Select Supply Side Equipment** dropdown, scoped to whatever equipment types are valid for the Loop Type you picked (e.g. only `CoolingTower:SingleSpeed` and its siblings for a Condenser loop), plus a **Set Quantity** field for how many instances to place immediately.

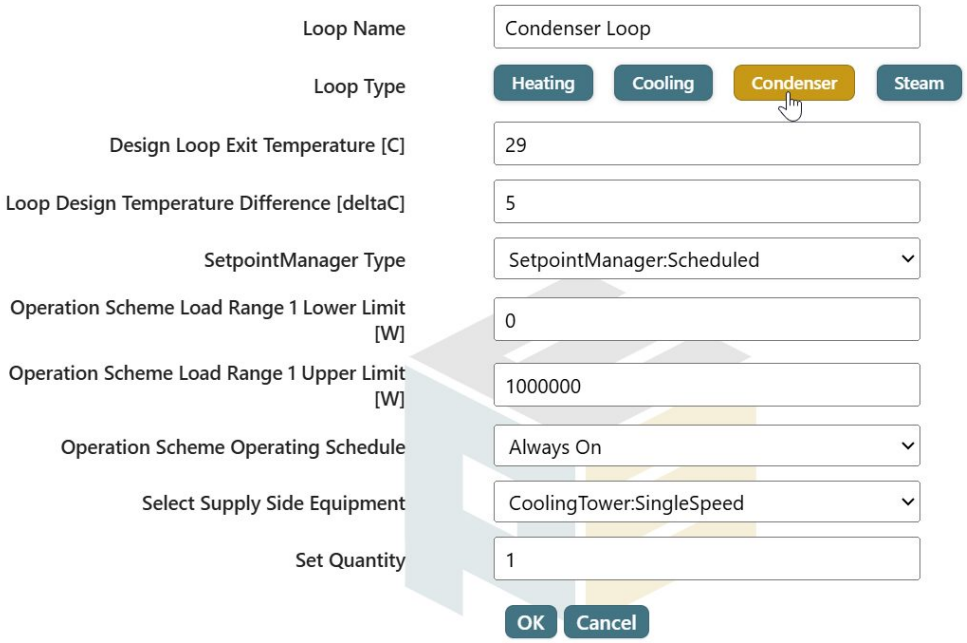


Figure 114: The Select Loop Options dialog for a new Condenser loop: Loop Name "Condenser Loop", Loop Type set to Condenser, Design Loop Exit Temperature 29°C and Loop Design Temperature Difference 5Δ°C (both matching Condenser's defaults), SetpointManager Type set to SetpointManager:Scheduled, Operation Scheme Load Range 1 set to 0–1,000,000 W with an Always On operating schedule, and Select Supply Side Equipment set to CoolingTower:SingleSpeed with a quantity of 1.

2.1 Two different "Loop Type" fields, easy to conflate

The **Loop Type** prompted above is `Sizing:Plant`'s field that feeds EnergyPlus's sizing calculation.

Separately, the `PlantLoop` object carries an EP3-only field, **Loop Conditioning Type** (`All/Heating/Cooling/Condenser`), that determines which equipment types EP3 lets you connect to the loop. This field gates the Draw Branch tool's connection options. This will prevent you from connecting a chiller onto a loop whose Loop Conditioning Type is `Heating`, for example.

The two fields interact rather than duplicate each other. Loop Conditioning Type defaults to

being **derived** from `Sizing:Plant`'s Loop Type whenever you haven't set it explicitly — so a freshly-created Cooling loop behaves as cooling-restricted by default, without you ever touching Loop Conditioning Type directly. The standard workflow is to leave the default. However, for advanced modelers you may want more loop connection flexibility. Setting the Loop Conditioning Type to `All` lifts the restrictions entirely. The restriction only applies to the current selection - changing the loop conditioning type **does not** change or remove what objects are connected to the loop, nor does changing `Sizing:Plant`'s Loop Type update an explicitly-set Loop Conditioning Type. If equipment is already connected to the loop when a selection is made, the loop's connected equipment can end up out of sync with what the Loop Type claims.

Setting Loop Conditioning Type to `All` should be a deliberate, advanced-user choice — it means EP3 will let you wire any HVAC equipment type onto that loop, correct or not, without guardrails.

2.2 Condenser-loop pairing

EnergyPlus's IDD still defines a legacy `CondenserLoop` object, but it's functionally identical to `PlantLoop` under the hood — EP3 only ever works with `PlantLoop`. A "condenser loop" in EP3 is simply an ordinary `PlantLoop` whose `Sizing:Plant` **Loop Type** happens to be set to `Condenser` (§11); there's no separate object type, dialog, or mechanism dedicated to condenser loops specifically. For imported models, `CondenserLoops` are changed to `PlantLoops` on import.

---

3. Branch Architecture: Splitters, Mixers, and Filler Objects

Unlike an `AirLoopHVAC`, where a splitter/mixer pair only appears for a multi-outlet or parallel-branch configuration (*(see "Air Loops, Branches, and Nodes", §2)*), every `PlantLoop` gets a Supply Splitter, Supply Mixer, Demand Splitter, and Demand Mixer unconditionally, the moment the loop is created, regardless of how simple the loop is. The minimum branch structure on **both** the supply and demand side is therefore: inlet branch → splitter → at least one parallel branch → mixer → outlet branch. A single-piece-of-equipment `PlantLoop` still has this full six-branch shape (three on each side) — there's no simpler topology to fall back to.

3.1 Every branch needs something on it

EP3 requires every branch to hold at least one object; a branch with no components fails validation, since EnergyPlus itself requires the same. For a branch that only needs to pass fluid through without adding or removing load, or requesting flow — an unused bypass, for instance — the correct object is `Pipe:Adiabatic`. This is the object type EP3 places automatically on an inlet or outlet branch with no pump or real equipment, and it's also what makes a branch recognizable as a **bypass branch** in the first place: EP3 identifies a branch as a bypass by checking whether its only component is a passive pipe, rather than treating "bypass" as a slot with restricted contents. Place active equipment on that branch instead, and it becomes an ordinary parallel branch rather than a bypass.

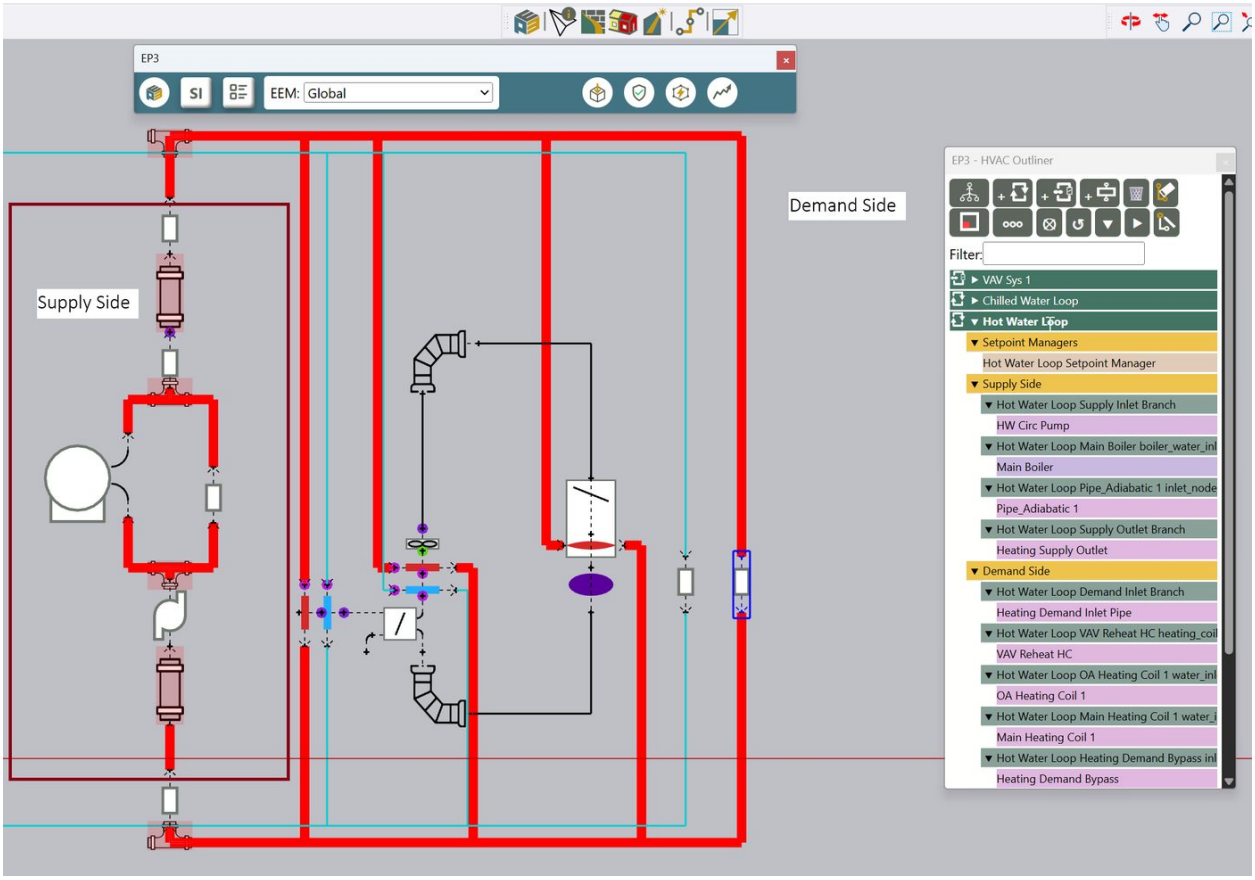
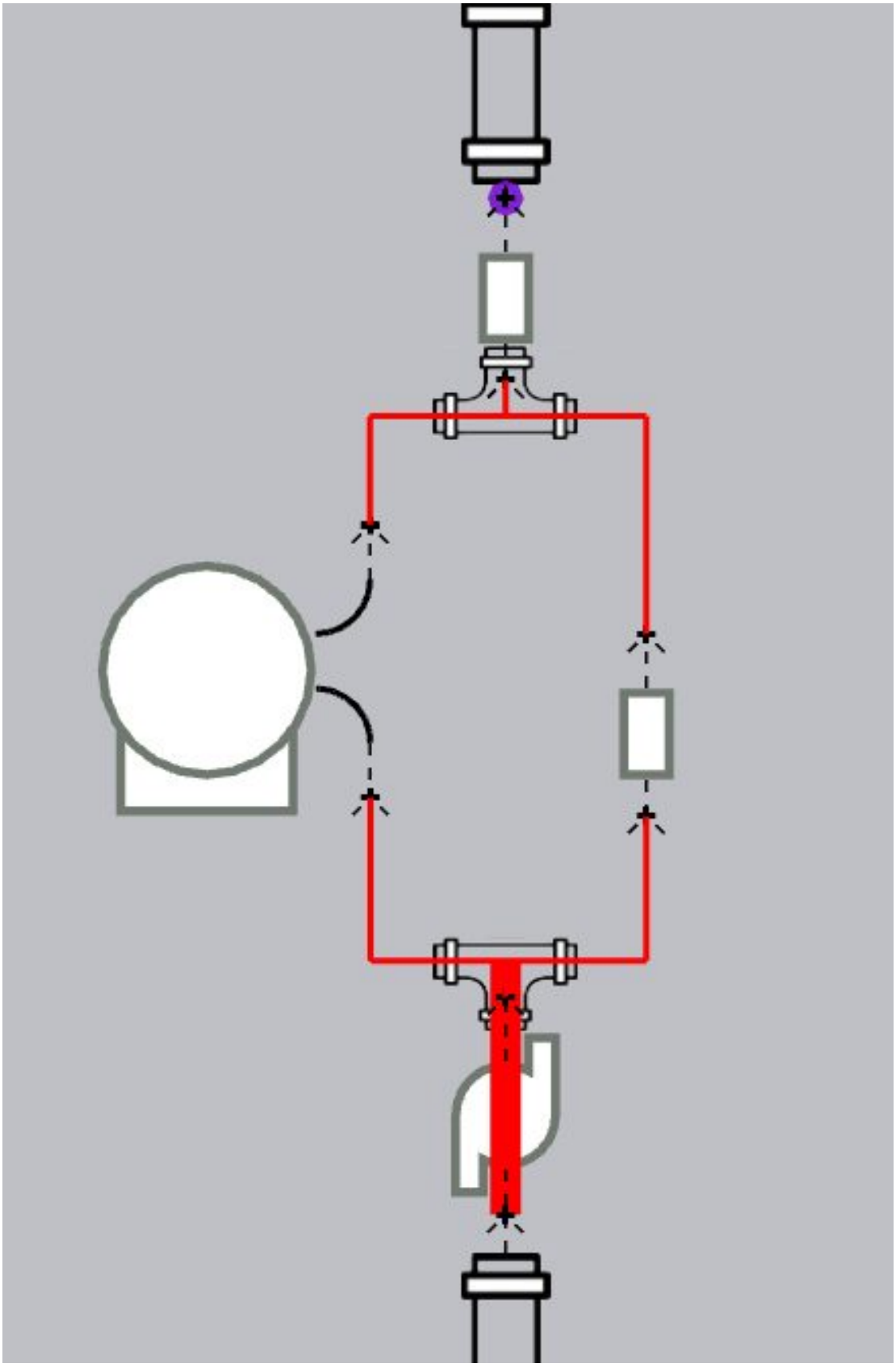
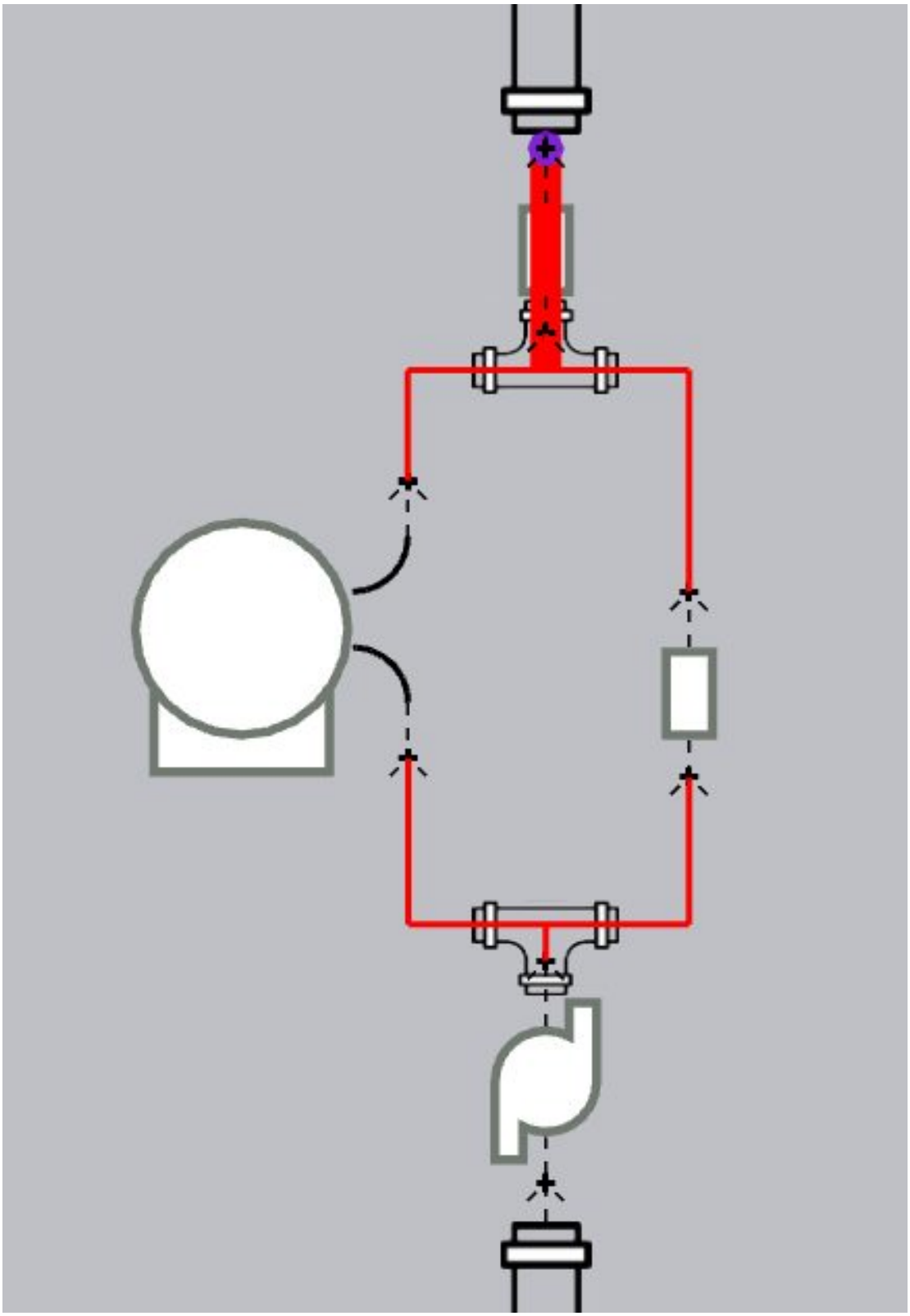


Figure 115: A Hot Water Loop's Supply Side and Demand Side, shown in the 3D viewport alongside the HVAC Outliner tree confirming each branch by name: Supply Inlet Branch (HW Circ Pump), Main Boiler branch, Pipe_Adiabatic 1 branch, and Supply Outlet Branch on the supply side; Demand Inlet Branch, VAV Reheat HC, OA Heating Coil 1, Main Heating Coil 1, and Heating Demand Bypass on the demand side.



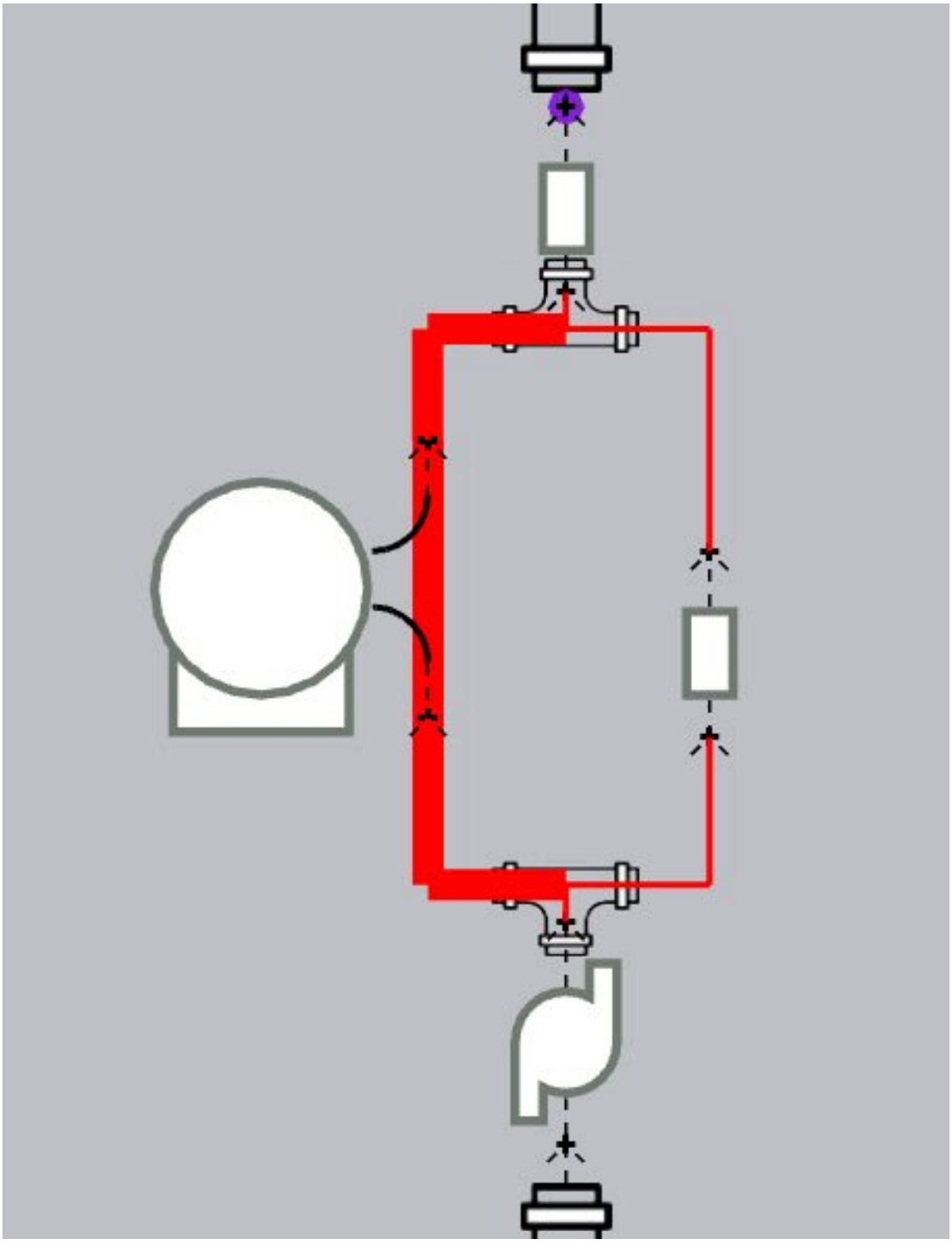


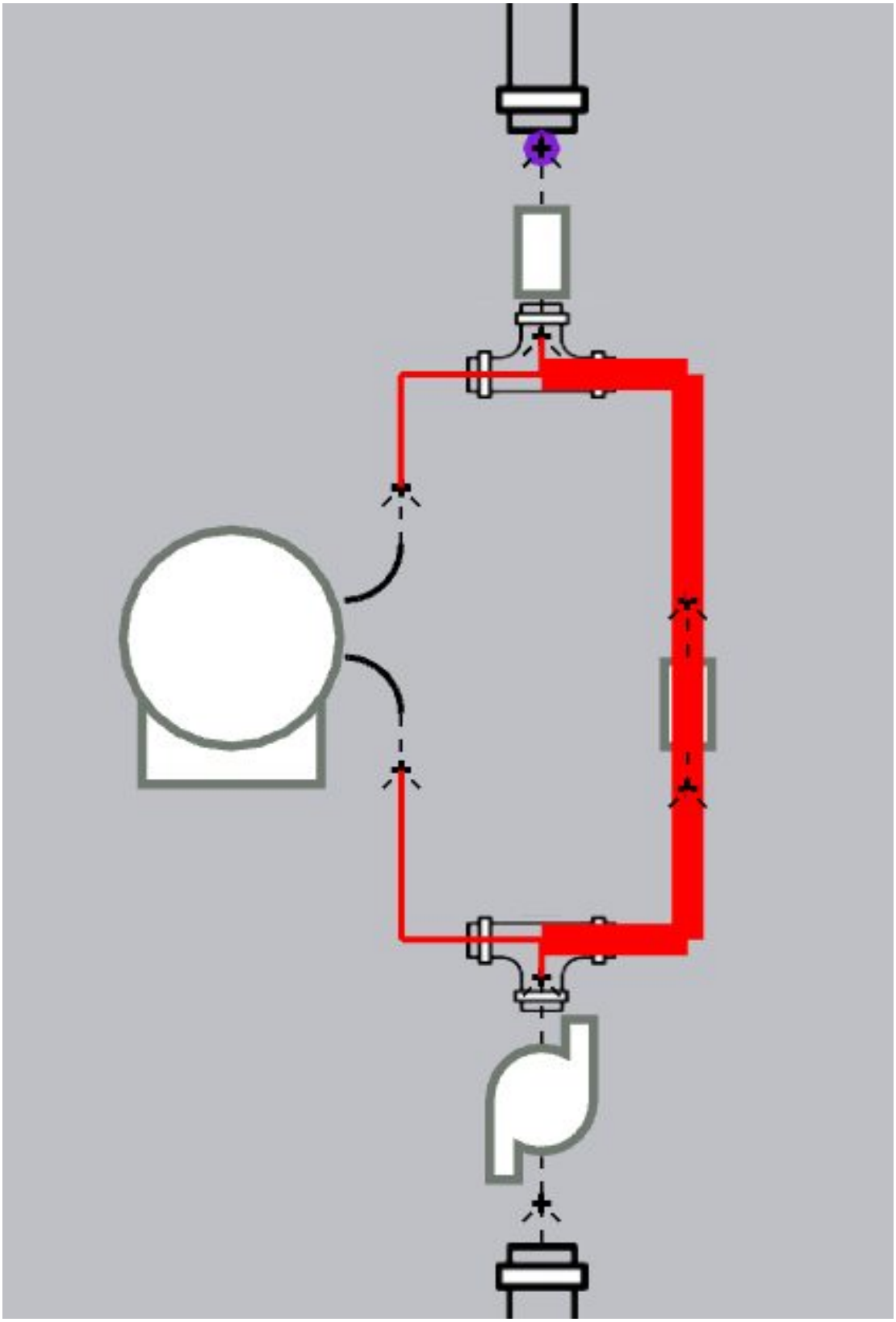
3.2 Bypass branches

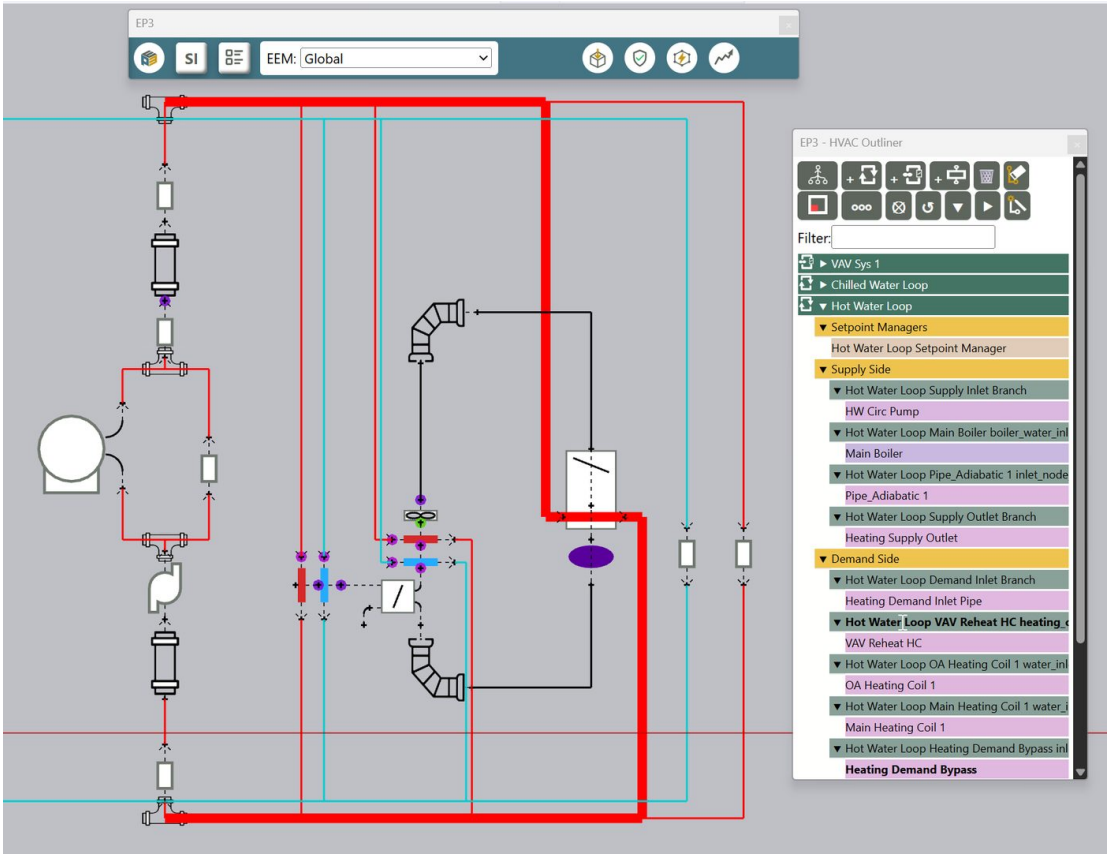
A bypass branch runs in parallel with the branches that hold active equipment, connecting the splitter directly back to the mixer with nothing but a `Pipe:Adiabatic` in between. It serves two purposes:

- It gives the loop somewhere for excess flow to go when the loop's flow rate is greater than what the active equipment on the other parallel branches is requesting — without a bypass, that surplus flow would have nowhere to go.
- It gives EnergyPlus's flow-resolution algorithm somewhere to put small rounding-error flows that don't perfectly balance across the other branches, rather than failing to converge.

A branch is only recognized as a bypass because of what it holds, not the reverse — so "the demand-side bypass branch" always means whichever demand-side branch carries nothing but a passive pipe.







---

4. Boilers

`Boiler:HotWater` is EP3's boiler type — no other boiler type is available in EP3.

A boiler sits on a parallel supply-side branch of a Heating-type `PlantLoop` (§3), serving whatever hot-water demand equipment — baseboards, fan coils, reheat coils, air-handler heating coils — is wired to that loop's demand side. Multiple boilers can share a loop, staged by an Operation Scheme (§12) so a small boiler covers light loads and a second boiler joins in above some threshold.

Field	Default	Notes
Fuel Type	— (required)	Coal/Diesel/Electricity/FuelOilNo1/FuelOilNo
Nominal Capacity	Autosize	
Nominal Thermal Efficiency	— (required)	Based on the fuel's higher heating value
Normalized Boiler Efficiency Curve Name	—	References a curve (XREF Ch.17); dependence on Entering or Leaving Boiler temperature set by Efficiency Curve Temperature Evaluation Variable
Design Water Flow Rate	Autosize	

Field	Default	Notes
Minimum / Maximum / Optimum Part Load Ratio	0.0 / 1.0 / 1.0	
Water Outlet Upper Temperature Limit	99.9°C	
Boiler Flow Mode	NotModulated	
On Cycle Parasitic Electric Load	0 W	
Sizing Factor	1.0	
End-Use Subcategory	"General"	

Boiler:HotWater covers every field exhaustively.

---

5. Chillers

EP3 supports 4 of EnergyPlus's 9+ chiller and chiller-heater types: `Chiller:Electric`, `Chiller:Electric:EIR`, `Chiller:Electric:ReformulatedEIR`, and `Chiller:ConstantCOP`. Absorption chillers, engine-driven and combustion-turbine chillers, `Chiller:Electric:ASHRAE205`, and chiller-heaters exist in EnergyPlus but aren't available in EP3; the plant-loop/water-to-water heat pump families that model similar territory (§6) are available instead.

If you need an unsupported chiller type, request it from the EP3 team — turnaround on adding a new EnergyPlus object type to EP3 is typically a few days.

`Chiller:Electric:EIR` is the type you'll reach for most often. Like a boiler, it sits on a parallel supply-side branch — this time of a Cooling-type `PlantLoop` — and multiple chillers can share a loop under the same Operation Scheme staging (§12).

Field	Default	Notes
Reference Capacity	Autosize	
Reference COP	— (required)	
Reference Leaving Chilled Water Temperature	6.67°C	
Reference Entering Condenser Fluid Temperature	29.4°C	
Condenser Type	WaterCooled	Alternatives: AirCooled, EvaporativelyCooled — basin-heater fields only apply to the evaporative case
Chiller Flow Mode	NotModulated	
Condenser Flow Control	ConstantFlow	
Design Heat Recovery Water Flow Rate	0.0	Only valid for WaterCooled
End-Use Subcategory		

Five fields reference curves (all XREF Ch.17):

- **Cooling Capacity Function of Temperature Curve Name** and **Electric Input to Cooling Output Ratio Function of Temperature Curve Name** — both bivariate.
- **Electric Input to Cooling Output Ratio Function of Part Load Ratio Curve Name** — univariate.
- **Condenser Loop Flow Rate Fraction Function of Loop Part Load Ratio Curve Name** — univariate, used only when Condenser Flow Control is `ModulatedLoopPLR`.
- **Thermosiphon Capacity Fraction Curve Name** — univariate, optional; leaving it blank disables the thermosiphon (free-cooling) feature entirely.

`Chiller:ConstantCOP` is the simplest of the four — no capacity/EIR curves at all, just a flat COP applied regardless of operating condition. `Chiller:Electric` (the "DOE-2 formulation," not to be confused with `Chiller:Electric:EIR`) and `Chiller:Electric:ReformulatedEIR` sit between these two in complexity; Chiller:Electric and Chiller:Electric:ReformulatedEIR cover each one's field set.

---

6. Water-to-Water Heat Pumps

Three separate EnergyPlus object families model a water-to-water heat pump in EP3, each answering the same modeling need — a heat pump moving heat between two water loops — at a different level of detail.

`HeatPump:PlantLoop:EIR:Cooling/:Heating`

Its **Condenser Type** field can be set to `AirSource` or `WaterSource` per instance: choosing `AirSource` gives the object an outdoor-air node pair on its source side instead of requiring a condenser `PlantLoop` connection. Changing Condenser Type after the object is already wired to a source-side `PlantLoop` is blocked outside the Global EEM: EP3 requires you to either erase the branch connection or turn off the PlantLoop in that EEM before the change is allowed, rather than leaving a dangling branch. A **Companion Heat Pump Name** field links a Cooling instance to its Heating counterpart so the two coordinate sizing as a pair.

`HeatPump:WaterToWater:EquationFit:Cooling/:Heating`

Always water-source on both sides — Source Side and Load Side node pairs, no Condenser Type choice at all. Performance is entirely curve-fit: capacity and power come from **Cooling Capacity Curve Name** and **Cooling Compressor Power Curve Name**, both quadvariate curves (XREF Ch.17) — a four-variable curve family, distinct from the bivariate curves a chiller uses, fit against both sides' entering conditions and flow fractions at once.

Field	Default	Notes
Reference Load Side Flow Rate	Autosize	
Reference Source Side Flow Rate	Autosize	
Reference Cooling Capacity	Autosize	
Reference Cooling Power Consumption	Autosize	

Field	Default	Notes
Reference Coefficient of Performance	8.0	Optional; used to autosize Reference Cooling Power Consumption when given
Sizing Factor	1.0	
Companion Heating Heat Pump Name	—	Links a Cooling instance to its Heating counterpart

`HeatPump:WaterToWater:ParameterEstimation:Cooling/:Heating`

A different modeling approach from the two families above: instead of curve-fit performance data, this object describes the heat pump's actual reciprocating compressor — fields include **Piston Displacement**, **Compressor Clearance Factor**, and **Load Side/Source Side Heat Transfer Coefficient** — alongside **Nominal COP** and **Nominal Capacity**, and EnergyPlus calculates performance from that physical description rather than from a curve. EP3 provides no default field values the way the two curve-driven families have. This is an advanced object type for a very specific use. The EP3 team recommends choosing one of the other chiller types.

---

7. Cooling Towers and Fluid Coolers

A fluid-cooler alternative to a cooling tower also exists — `FluidCooler:SingleSpeed/:TwoSpeed` and `EvaporativeFluidCooler:SingleSpeed/:TwoSpeed`. Reach for a fluid cooler instead of a cooling tower when the condenser loop needs closed-circuit (dry or evaporatively-assisted) heat rejection rather than the open, direct-contact-with-air heat rejection a cooling tower provides.

Approach is the difference between the tower's leaving water temperature and the entering air wet-bulb temperature — the theoretical limit a tower can never fully close; Range is the temperature drop the water experiences across the tower. A tighter (smaller) Approach means a larger, more expensive tower for the same load.

`CoolingTower:SingleSpeed`'s fields:

Field	Default	Category
Design Water Flow Rate	Autosize	Sizing
Design Air Flow Rate	Autosize	Sizing
Design Fan Power	Autosize	Sizing
Performance Input Method	UFactorTimesAreaAndDesignWaterFlowRate	Sizing — alternative: NominalCapacity
Design U-Factor Times Area Value / Nominal Capacity	Autosize	Sizing — whichever the Performance Input Method requires
Heat Rejection Capacity and Nominal Capacity Sizing Ratio	1.25	Sizing
Design Inlet Air Dry-Bulb Temperature	35°C	Design conditions

Field	Default	Category
Design Inlet Air Wet-Bulb Temperature	25.6°C	Design conditions
Design Approach Temperature	Autosize (→ 3.9Δ°C)	Design conditions
Design Range Temperature	Autosize (→ 5.5Δ°C)	Design conditions
Basin Heater Capacity / Setpoint Temperature (2°C min) / Operating Schedule		Water management
Evaporation Loss Mode	SaturatedExit	Water management — alternative: LossFactor
Evaporation Loss Factor	0.2%/K	Water management
Drift Loss Percent	0.008%	Water management
Blowdown Calculation Mode	ConcentrationRatio	Water management
Blowdown Concentration Ratio	3.0	Water management
Supply Water Storage Tank Name		Water management
Capacity Control	FanCycling	Operation — alternative: FluidBypass
Number of Cells		Operation
Cell Control	MaximalCell	Operation — alternative: MinimalCell
Cell Minimum / Maximum Water Flow Rate Fraction	0.33 / 2.5	Operation

CoolingTower:SingleSpeed covers every field exhaustively; `:TwoSpeed` and `:VariableSpeed` add low/high-speed or continuously-variable fan control on top of the same core field set.

---

8. Pumps

Every new `PlantLoop`, regardless of Loop Type, gets a `Pump:VariableSpeed` placed on its supply inlet branch automatically — there's no per-loop-type pump choice in the Select Loop Options dialog at all. Swapping to `Pump:ConstantSpeed` or a headered-pump arrangement afterward is a manual replacement; there's no dedicated "change pump type" action.

8.1 Fields, and two different kinds of curve

`Pump:VariableSpeed`'s fields:

Field	Default	Notes
Design Flow Rate	Autosize	
Design Power Consumption	Autosize	
Design Pump Head	179,352 Pa (60 ft)	
Motor Efficiency	0.9	

Field	Default	Notes
Fraction of Motor Inefficiencies to Fluid Stream		
Pump Control Type	Continuous	Alternative: Intermittent
Design Power Sizing Method	PowerPerFlowPerPressure	Alternative: PowerPerFlow
Design Electric Power per Unit Flow Rate	348,701.1 W/(m³/s)	Used by the PowerPerFlow method
Design Shaft Power per Unit Flow Rate per Unit Head	1.282051282	Used by the PowerPerFlowPerPressure method — reciprocal of an implied ~78% pump efficiency
Zone Name		Optional; pump heat rejected as an internal gain
End-Use Subcategory		

Two different curve-like mechanisms appear on the same object:

- Four **Part-Load Performance Curve** coefficient fields (default 0/1/0/0) are literal polynomial coefficients typed directly onto the pump object — not a reference to any `Curve` object at all.
- **Pump Curve Name** is a separate, optional field that references a real `Curve` object (univariate — XREF Ch.17), describing the pump's pressure-vs-flow relationship for cases needing that level of detail rather than the simpler head/flow-rate fields above.

`Pump:ConstantSpeed` carries the same node/flow/head/efficiency fields but drops the part-load curve coefficients, the Minimum Flow Rate field, and the VFD-specific fields entirely — consistent with running at one fixed speed, there's no part-load behavior to describe.

`HeaderedPumps:ConstantSpeed`/`:VariableSpeed` model a bank of identical pumps behind one pair of nodes, with a Number of Pumps in Bank field and a Flow Sequencing Control Scheme deciding how EnergyPlus stages them on as flow demand rises. Not covered further here.

8.2 Worked example: sizing Design Power Consumption from Design Flow Rate and Head

For the default **PowerPerFlowPerPressure** sizing method, EnergyPlus derives Design Power Consumption from the pump's Design Flow Rate and Design Pump Head, using the Design Shaft Power per Unit Flow Rate per Unit Head coefficient and Motor Efficiency:

Shaft Power [W] = Flow Rate [m³/s] × Pump Head [Pa] × (Design Shaft Power per Unit Flow Rate per Unit Head) Electric Power [W] = Shaft Power ÷ Motor Efficiency

Take a pump autosized to a 0.02 m³/s (~317 gpm) design flow rate, left at the default 179,352 Pa (60 ft) head, the default 1.282051282 shaft-power coefficient (~78% implied pump efficiency), and the default 0.9 motor efficiency:

`Shaft Power = 0.02 × 179,352 × 1.282051282 ~ 4,599 W  Electric Power = 4,599 ÷ 0.9 ~ 5,110 W`

Checked against the IP rule-of-thumb brake-horsepower formula (BHP = GPM × Head[ft] ÷ (3,960 × pump efficiency)):

`BHP = (317 × 60) ÷ (3,960 × 0.78) ~ 6.16 hp ~ 4,595 W` (shaft power, matching within rounding) `Electric Power = 6.16 ÷ 0.9 ~ 6.84 hp ~ 5,103 W` — matching the SI calculation within rounding.

(3,960 is the standard hydraulic constant relating GPM, feet of head, and horsepower for water at typical density; per any standard pump-sizing reference, e.g. ASHRAE Fundamentals.)

---

9. Water-Side Coils, as Plant-Loop Demand

`Coil:Cooling:Water`, `Coil:Cooling:Water:DetailedGeometry`, and `Coil:Heating:Water` appear either as standalone objects wired directly onto a `PlantLoop` branch, or as a sub-component slot inside a larger piece of equipment — a Four-Pipe Fan Coil's or VAV Reheat terminal's coil slot, for instance (*(see "Zone-Level HVAC Equipment and Zone Sizing")*, *(see "Air Loops, Branches, and Nodes", §5)*). Either way, a matching `Controller:WaterCoil` object appears alongside it automatically.

9.1 What a water coil looks like from the plant-loop side

Regardless of whether a given coil is standalone or riding along as a sub-component, its **water-side** nodes are always externally connectable into a `PlantLoop` demand branch. From the plant loop's perspective, every water coil in the model is demand and creates a load on the loop.

`Coil_Cooling_Water`'s fields:

Field	Default	Notes
Design Water Flow Rate	Autosize	
Design Air Flow Rate	Autosize	
Design Inlet / Outlet Air Temperature	Autosize	
Design Inlet / Outlet Air Humidity Ratio	Autosize	
Type of Analysis	SimpleAnalysis	Alternative: DetailedAnalysis
Heat Exchanger Configuration	CounterFlow	Alternative: CrossFlow
Design Water Temperature Difference	—	Optional override for sizing the Design Water Flow Rate; when left blank, EnergyPlus uses the `Sizing:Plant` Loop Design Temperature Difference instead (*(see "Air Loops, Branches, and Nodes", §4.5)*)

`Coil_Heating_Water` carries the same override and fallback behavior, plus its own capacity-side fields: **U-Factor Times Area Value**, **Performance Input Method**, **Rated Capacity**, and rated inlet/outlet water and air temperatures.

The paired `Controller:WaterCoil` varies the coil's water flow during a simulation to hit whatever setpoint a `SetpointManager` (§13) has placed downstream. The coil object describes physical design flow and capacity only, not runtime control behavior.

9.2 Worked example: coil water flow rate from capacity and Loop Design Temperature Difference

Solving Q = m. · cp · ΔT for volumetric water flow gives the standard water-side sizing relationship:

Flow [gpm] = Capacity [Btu/h] ÷ (500 × ΔT [°F])

(500 = 8.33 lb/gal × 60 min/hr × 1 Btu/lb · °F, water's specific heat — the water-side analog of the 1.08 air-side constant (see "Zone-Level HVAC Equipment and Zone Sizing", §4.3)* uses for airflow.)*

Suppose a cooling coil's calculated design capacity is 120,000 Btu/h (10 tons), and its `PlantLoop`'s `Sizing:Plant` **Loop Design Temperature Difference** is 10°F — the coil's **Design Water Temperature Difference** field was left blank, so it inherits this value directly, per §9.1:

`Flow = 120,000 ÷ (500 × 10) = 24 gpm`

In SI: the same load is 35,169 W, the same ΔT is 5.56°C, and the equivalent relationship is Flow [L/s] = Capacity [W] ÷ (4.186 × ΔT [°C] × 1000):

`Flow = 35,169 ÷ (4,186 × 5.56) ~ 1.51 L/s ~ 24.0 gpm` — matching the IP result within rounding.

Tightening the loop's Design Temperature Difference (say, to 14°F instead of 10°F, to shrink pipe sizes and pump energy) lowers the calculated design flow for the same capacity — `120,000 ÷ (500 × 14) ~ 17 gpm` — the same design tradeoff *(see "Air Loops, Branches, and Nodes", §4.3)* describes for a system's design supply air temperature.

---

10. `HeatExchanger:FluidToFluid` — Water-to-Water Heat Exchange

`HeatExchanger:FluidToFluid` is EP3's preferred mechanism for primary-secondary hydronic configurations and other loop-to-loop energy sharing.

10.1 Two node pairs, one on each loop

The object carries two independent node pairs, each wired onto a different `PlantLoop`:

- **Loop Demand Side Inlet/Outlet Node Name** — wired onto the *primary* loop's demand branch. From that loop's perspective, the heat exchanger is just another piece of demand equipment, drawing chilled or hot water the same way a coil would.
- **Loop Supply Side Inlet/Outlet Node Name** — wired onto the *secondary* loop's supply branch. From that loop's perspective, the heat exchanger stands in for a chiller or boiler — it's the secondary loop's actual heat source or heat sink.

Each side has a separate **Design Flow Rate**, which is autosizeable.

10.2 Fields that control how and when it operates

Heat Exchange Model Type (`Ideal` default — assumes perfect heat transfer, no UA needed; alternatives `CounterFlow`, `ParallelFlow`, and three `CrossFlow` variants, all of which use **Heat Exchanger U-Factor Times Area Value**, autosize) sets the physical heat-transfer assumption.

Control Type decides how the exchanger activates and modulates, and determines whether this configuration behaves like a true primary/secondary loop:

- `UncontrolledOn` (default) — runs continuously whenever both loops are calling for flow, with no setpoint logic at all.
- `CoolingDifferentialOnOff` — activates purely based on which loop is colder than the other, the mechanism behind a "free cooling" configuration with no explicit setpoint.
- The Setpoint-driven family — `CoolingSetpointModulated/OnOff`, `HeatingSetpointModulated/OnOff`, `DualDeadbandSetpointModulated/OnOff`, `CoolingSetpointOnOffWithComponentOverride` — watches a **Heat Exchanger Setpoint Node Name** and throttles (Modulated) or cycles (OnOff) the exchanger to hold that node at setpoint. The `WithComponentOverride` variant additionally lets **Component Override Cooling Control Temperature Mode** substitute a dry-bulb, wet-bulb, or generic loop-temperature signal for the control decision instead.
- `OperationSchemeModulated/OnOff` — hands control to a `PlantEquipmentOperation` scheme (§12) instead of managing it internally.

Minimum Temperature Difference to Activate Heat Exchanger (0.01°C default) is a tolerance band preventing the exchanger from cycling on for a negligible temperature difference. **Operation Minimum/Maximum Temperature Limit** are separate, hard cutoffs — the exchanger won't run at all if either inlet temperature falls outside this range, independent of whatever Control Type would otherwise call for. **Heat Transfer Metering End Use Type** (`LoopToLoop` default; alternatives `FreeCooling`, `HeatRecoveryForCooling`, `HeatRecoveryForHeating`, `HeatRejection`) determines which output-meter end-use category the exchanger's energy shows up under. Set it deliberately if you want free-cooling savings isolated in your results rather than lumped into generic loop-to-loop transfer.

HeatExchanger:FluidToFluid covers every field exhaustively.

---

11. `Sizing:Plant`

Every `PlantLoop` owns exactly one `Sizing:Plant` object, created and managed the same lazy-create/embedded-in-the-loop-dialog way `Sizing:System` is for an `AirLoopHVAC` (*(see "Air Loops, Branches, and Nodes", §4.1)*) — you never add or remove one independently.

Field	Default	Notes
Plant or Condenser Loop Name	—	Mirrors the parent loop; not independently editable

Field	Default	Notes
Loop Type	— (required)	`Condenser/Cooling/Heating` — set by the Select Loop Options dialog at creation, §2.1
Design Loop Exit Temperature	— (required)	
Loop Design Temperature Difference	— (required)	The field §9.2's worked example depends on
Sizing Option	NonCoincident	Alternative: `Coincident`, which requires `SimulationControl`'s "Do HVAC Sizing Simulation for Sizing Periods" to be turned on — the same requirement `Sizing:System`'s Coincident option has (*(see "Air Loops, Branches, and Nodes", §4.5)*)
Zone Timesteps in Averaging Window	—	Used only by the coincident sizing algorithm's running-average calculation
Coincident Sizing Factor Mode	—	`GlobalCoolingSizingFactor/GlobalHeating`

11.1 Where these values feed equipment elsewhere in this chapter

Loop Design Temperature Difference feeds `Coil:Cooling:Water/Coil:Heating:Water` directly: when a coil's own **Design Water Temperature Difference** field is left blank, EnergyPlus sizes that coil's Design Water Flow Rate from this loop-level value instead (§9.1, worked example in §9.2).

Design Loop Exit Temperature sets the supply temperature the loop's equipment is actually expected to hit — and several pieces of equipment elsewhere in this chapter carry their own rated or reference temperature fields that should agree with it, though nothing in EP3 enforces that agreement:

- `Coil:Heating:Water`'s **Rated Inlet Water Temperature** defaults to 82.2°C — matching a Heating loop's Design Loop Exit Temperature default of 82°C (§2) almost exactly; both reflect the same standard 180°F hot-water design convention. Change one without the other and the coil's rated performance ends up characterized against a water temperature the loop was never actually designed to deliver.
- `Chiller:Electric:EIR`'s **Reference Leaving Chilled Water Temperature** defaults to 6.67°C — close to, but not identical to, a Cooling loop's own 7°C default (§2), a reminder that EnergyPlus's stock defaults for a piece of equipment and for the loop it sits on aren't guaranteed to already agree with each other; check both when you change either.
- `CoolingTower:SingleSpeed`'s **Design Approach Temperature** combines with the entering air wet-bulb design condition to produce a leaving water temperature that should, in turn, match a Condenser loop's Design Loop Exit Temperature (29°C default, §2).

Nothing in the Validation dialog's HVAC checks extends to this — only the setpoint-vs-design-

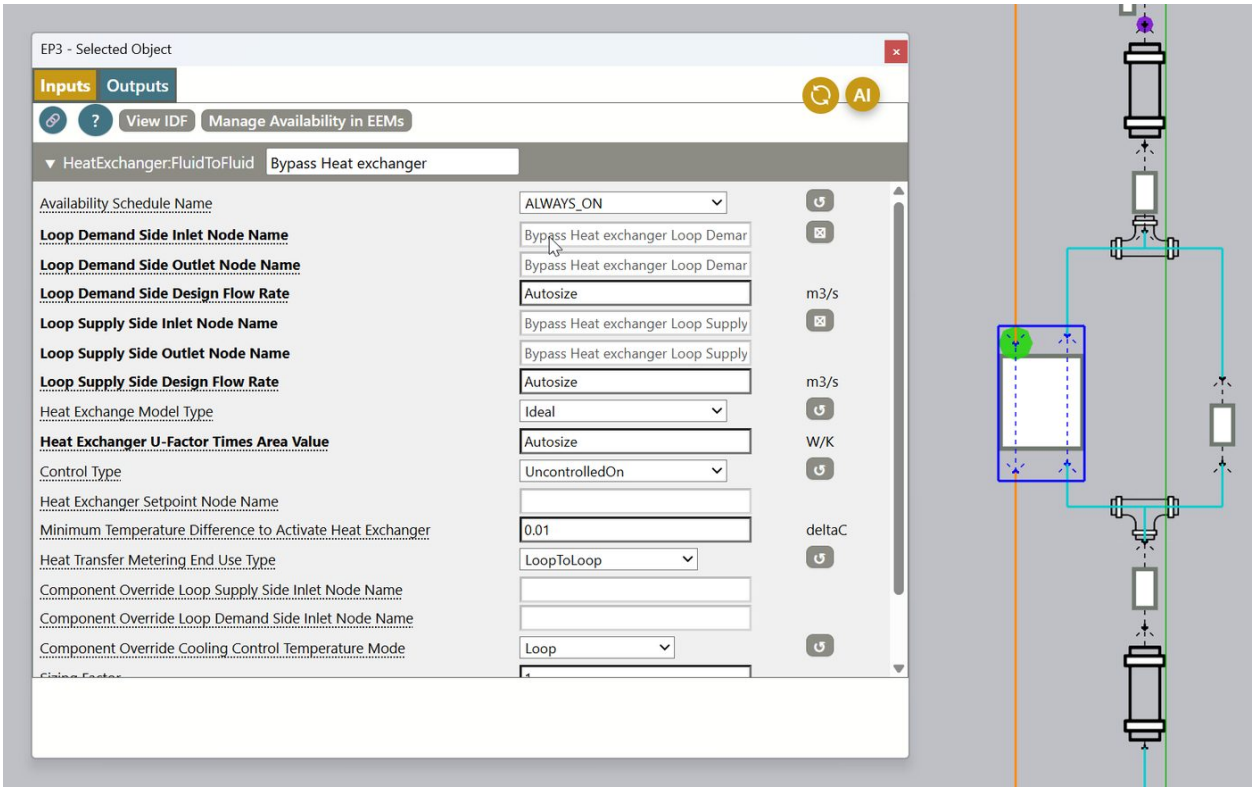


Figure 116: A HeatExchanger:FluidToFluid named "Bypass Heat exchanger," its Selected Object dialog showing the full field set — Loop Demand/Supply Side Inlet/Outlet Node Names, Design Flow Rates, Heat Exchange Model Type (Ideal), Control Type (UncontrolledOn), Minimum Temperature Difference to Activate Heat Exchanger (0.01 deltaC), Heat Transfer Metering End Use Type (LoopToLoop) — alongside its 3D diagram connection, with the orange loop as the primary/demand-side loop and the blue loop as the secondary/supply-side loop, its demand inlet node connection highlighted.

temperature relationship (§13.6) is checked automatically. A chiller's or coil's own rated/reference temperature quietly disagreeing with its loop's Design Loop Exit Temperature isn't flagged — the same kind of unchecked drift *(see "Zone-Level HVAC Equipment and Zone Sizing", §5.3)* describes between `Sizing:Zone`'s assumptions and the equipment assigned to a zone.

Sizing:Plant covers every field exhaustively.

---

12. Dispatching Plant Equipment

Where more than one piece of equipment sits on a `PlantLoop`'s supply side, EP3 offers two layers of dispatch structure to decide which equipment runs, and when: equipment lists (§12.1) and operation schemes (§12.1–12.2), which can be stacked and reordered rather than reduced to a single distribution setting.

12.1 Equipment Lists and Operation Schemes

A **`PlantEquipmentList`** (or, for condenser-side equipment, a **`CondenserEquipmentList`**) is just an ordered array of Equipment Object Type/Name pairs — the equipment available to a given operation scheme, in dispatch-priority order.

A **`PlantEquipmentOperation`** scheme decides *when* and *which* equipment list is active. EP3 offers 11 of EnergyPlus's ~14 scheme types: `Uncontrolled`, `CoolingLoad`, `HeatingLoad`, `OutdoorDryBulb`, `OutdoorWetBulb`, `OutdoorRelativeHumidity`, `OutdoorDewpoint`, `ComponentSetpoint`, `OutdoorDryBulbDifferer` `OutdoorWetBulbDifference`, `OutdoorDewpointDifference` (missing: `ThermalEnergyStorage`, `ChillerHeaterChangeover`, `UserDefined` — as with an unsupported chiller type, §5, request one from the EP3 team if you need it). Multiple schemes on the same loop are then ordered via a drag-sortable **"Set Simulation order for Operation Schemes"** list, which writes the export-time evaluation order directly.

`PlantEquipmentOperation:CoolingLoad/:HeatingLoad` — the two most common scheme types — implement EnergyPlus's load-range-bracket structure: up to several **Load Range N Lower/Upper Limit** pairs (in W), each paired with a **Range N Equipment List Name**, so a small load might dispatch only a single small chiller while a load above some threshold brings a second chiller's equipment list into play. Creating a new `PlantLoop` auto-creates one such scheme by default (a `HeatingLoad` scheme for a Heating loop, a `CoolingLoad` scheme for a Cooling loop), pre-filled with a single load range of 0–1,000,000 W and an "Always On" operating schedule if one exists — you extend it with additional ranges yourself as needed.

12.2 `ComponentSetpoint` — dispatch by equipment, not by loop

Instead of dispatching equipment based on the loop's overall load or an outdoor condition, `PlantEquipmentOperation:ComponentSetpoint` assigns a setpoint **directly to each piece of equipment's outlet node** — up to several Equipment N Object Type/Name pairs, each with a **Demand Calculation N Node Name**, **Setpoint N Node Name**, **Component N Flow Rate**, and an **Operation N Type** (`Cooling`/`Heating`/`Dual`). This is the mechanism behind, for instance, staging multiple chillers, each to a fixed leaving-water-temperature target, independent of whatever a loop-level `SetpointManager` (§13) might otherwise be driving toward — see §13.5 for how the two control layers relate.

13. Setpoint and Availability Managers for `PlantLoops`

13.1 How they attach

A `SetpointManager` or `AvailabilityManager` is created through the generic object-library "Add" mechanism — the same path used for schedules and curves — rather than placed and wired with the Draw Branch tool. Once created, you attach it to a `PlantLoop` by clicking a reference or setpoint node, using the same bullseye-picker interaction used everywhere else in EP3 for picking a point in the 3D model.

For most setpoint manager types, the node choices the picker offers are the same regardless of whether you're attaching to a `PlantLoop` or an `AirLoopHVAC`, however there are some types of `SetpointManager` that are intended specifically for a particular loop type (§13.2–13.3).

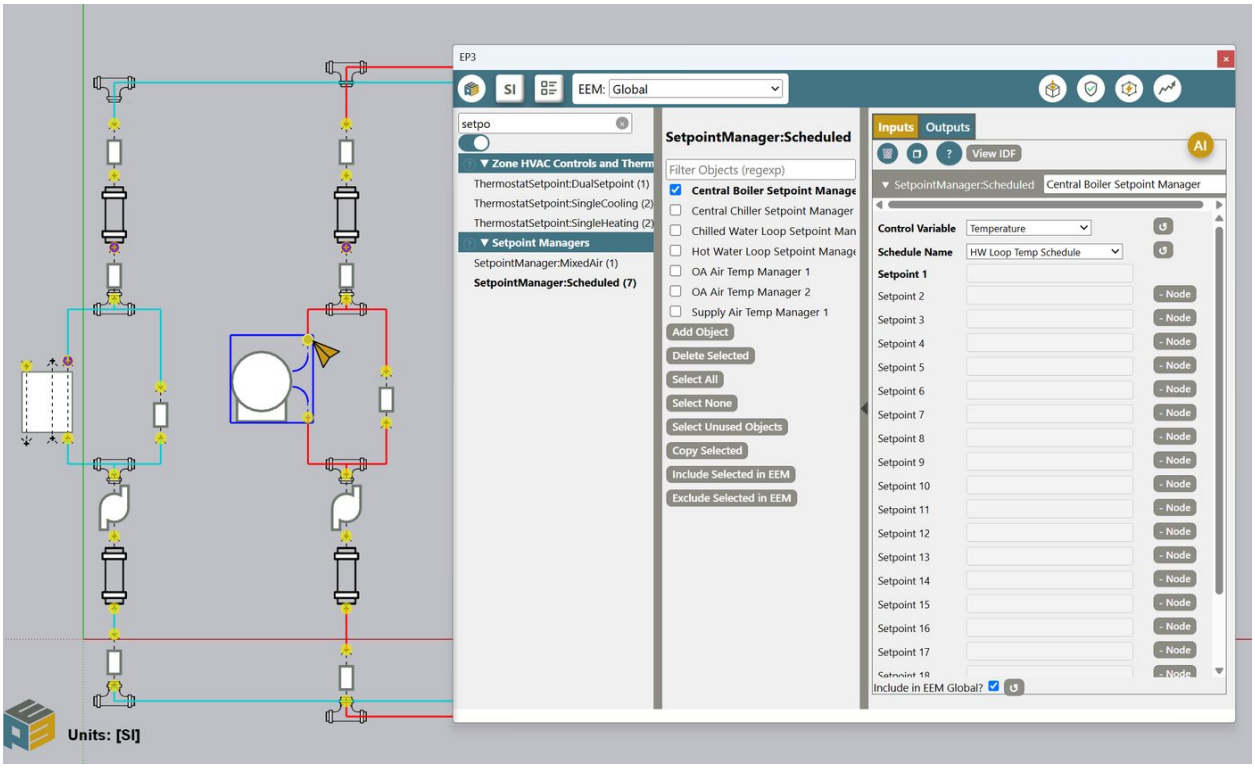


Figure 117: Creating a SetpointManager:Scheduled named "Central Boiler Setpoint Manager": the generic object list (filtered to "setpo") on the left, its Inputs panel on the right showing the Setpoint 1 field's node-picker button, and the bullseye cursor in the 3D viewport selecting the boiler's outlet node to attach Setpoint 1 to.

13.2 Plant-only, shared, and airloop-only setpoint manager types

Some `SetpointManager` types are typically applied only to a `PlantLoop`, some only to an `AirLoopHVAC`, and some work on either — a distinction in what each type actually controls, not a restriction EP3 enforces on which types you can pick. The types typically applied only to a `PlantLoop`

are `SetpointManager:ReturnTemperature:ChilledWater`, `:ReturnTemperature:HotWater`, `:CondenserEnteringReset`, and `:CondenserEnteringReset:Ideal` (§13.3). The generic types that work on either loop include `SetpointManager:Scheduled`, `:OutdoorAirReset`, and the `FollowGroundTemperature/FollowSystemNodeTemperature/FollowOutdoorAirTemperature` family.

`SetpointManager:Warmest/:Coldest`, the `SingleZone:*` family, `:MixedAir`, and the `MultiZone:*` average/humidity variants are typically applied only to an `AirLoopHVAC`, not a `PlantLoop`. These types reset an `AirLoopHVAC`'s supply air setpoint based on the zones it serves, and are covered in full at *(see "Air Loops, Branches, and Nodes", §8.1)*.

13.3 The plant-only setpoint manager types, briefly

- **`SetpointManager:ReturnTemperature:ChilledWater/:HotWater`** control the *supply* temperature setpoint indirectly, by targeting a desired *return* temperature instead — a different control paradigm from anything on the air side. Fields: **Plant Loop Supply Outlet/Inlet Node**, **Minimum/Maximum Supply Temperature Setpoint** (5°C/10°C default for chilled water; 77°C/82°C for hot water), and **Return Temperature Setpoint Input Type** (`Constant/Scheduled/ReturnTemperatureSetpoint`) deciding whether the target return temperature is a fixed value, a schedule, or supplied by yet another setpoint manager placed on the supply inlet node.
- **`SetpointManager:CondenserEnteringReset`** resets a condenser loop's entering water temperature setpoint using three curves (**Minimum Design Wetbulb Temperature**, **Minimum Outside Air Wetbulb Temperature**, **Optimized Condenser Entering Water Temperature** — all quadvariate) plus a **Minimum Lift** (11.1Δ°C default) and **Maximum Condenser Entering Water Temperature** (32°C default) limit, falling back to a **Default Condenser Entering Water Temperature Schedule** whenever the optimized calculation doesn't fall within those bounds. **`:CondenserEnteringReset:Ideal`** is the simplified version — same Minimum Lift and Maximum Condenser Entering Water Temperature limits, but no curves and no fallback schedule.

SetpointManager:ReturnTemperature:ChilledWater and the sibling types cover every field exhaustively.

13.4 Availability managers for `PlantLoops`

An `AvailabilityManager` attaches to a `PlantLoop` the same indirect way it attaches to an `AirLoopHVAC` — bundle one or more `AvailabilityManager` objects, in order, into an `AvailabilityManagerAssignmentList`, then pick that list on the loop's dialog (*(see "Air Loops, Branches, and Nodes", §8.2)*). EP3's `AirLoopHVAC:NightCycle` shortcut (*(see "Air Loops, Branches, and Nodes", §8.3)*) has **no `PlantLoop` equivalent** — there's no analogous embedded, auto-created availability manager for plant loops; building one means creating the object yourself, the plain way.

The availability manager types particularly relevant to a `PlantLoop`:

- **`AvailabilityManager:DifferentialThermostat`** turns the loop on and off based on the temperature difference between two sensor nodes — **Hot Node Name**, **Cold Node Name**, a **Temperature Difference On Limit** (required), and a **Temperature Difference Off Limit** (defaults to the On Limit if left blank, giving simple on/off rather than a deadband).

It's a generic two-node differential control usable on any `PlantLoop`, and the mechanism behind a solar-collector loop or a heat-recovery/waterside-economizer loop enabling flow only once one side is warmer or cooler than the other.

- `AvailabilityManager:HighTemperatureTurnOff/:HighTemperatureTurnOn` and `:LowTemperatureTurnOf` are simpler single-sensor-node, fixed-threshold pairs (**Sensor Node Name**, **Temperature**) — the classic economizer-lockout mechanism: turn a heating loop's availability off once outdoor (or some other sensed) temperature climbs above a threshold, or turn a cooling loop on only once it drops below one. `LowTemperatureTurnOff` additionally accepts an optional **Applicability Schedule Name**, defaulting to always-active if left blank.

AvailabilityManager:DifferentialThermostat covers every field exhaustively.

13.5 Where the loop expects a setpoint, vs. where equipment does

A loop-level `SetpointManager` writes its setpoint onto a single node — typically the loop's supply outlet — that supply-side equipment then controls toward as a group, coordinated by whichever `PlantEquipmentOperation` scheme is active (§12.1). `PlantEquipmentOperation:ComponentSetpoint` (§12.2) works differently and separately: it places a setpoint directly on each piece of equipment's node, bypassing any loop-level `SetpointManager` for those specific components entirely. A loop can use one mechanism, the other, or — where different equipment on the same loop is controlled different ways — both at once.

13.6 Checking your work

The Validation dialog's HVAC tab compares a loop's `SetpointManager` setpoint values against its `Sizing:Plant` design temperatures (§11) and flags a mismatch — for example, if `Sizing:Plant` was set up for a 6.67°C leaving chilled-water temperature but a `SetpointManager:Scheduled` on the same loop is driving the supply outlet node to 7.2°C. This is the same check *(see "Air Loops, Branches, and Nodes", §4.7)* describes for `Sizing:System` against an `AirLoopHVAC`'s setpoint managers. Full treatment of the Validation dialog and `NodeReport` (the general-purpose way to confirm what value a node holds at runtime, rather than trusting a dialog) is *(see "Model Troubleshooting and Quality Assurance")*'s job.

Separately, the Validation dialog flags any node controlled by more than one `SetpointManager`. This isn't automatically an error — EnergyPlus doesn't simply pick one setpoint manager per node and ignore the rest, so two managers sharing a node is a legitimate configuration as often as it's a mistake. What actually happens at runtime depends on which two are involved: a fixed `SetpointManager:Scheduled` value can end up fighting a load-based type on the same node (a scheduled 7°C chilled-water setpoint against a `SetpointManager:ReturnTemperature:ChilledWater` trying to reset it upward, for instance), and either could win depending on evaluation order.

The check reflects that ambiguity in its severity:

- **Fail** — a `SetpointManager:Scheduled` is paired with another manager on the same node — the fixed-value case most likely to actively fight whatever else is there.
- **Warning** — any other combination of multiple setpoint managers on the same node, flagged so you check that they don't conflict, without assuming they do.

Confirming which setpoint manager actually wins on a shared node — rather than guessing from the Validation dialog's warning alone — is a job for `NodeReport` (*(see "Model Troubleshooting and*

Quality Assurance")), reading the node's actual setpoint value at runtime.

---

14. Where to Look Next

- *(see "The HVAC Outliner and Creating HVAC Systems")* — the Outliner, Components dialog, and Draw Branch mechanics this chapter assumes.
- *(see "Zone-Level HVAC Equipment and Zone Sizing")* — the composite/sub-component pattern, and `Sizing:Zone`'s cascading-defaults mechanism.
- *(see "Air Loops, Branches, and Nodes")* — `Sizing:System`, the air-side setpoint/availability manager types this chapter's §13.1 contrasts against, and the airside terminal units a water coil might also be feeding.
- *(see "Performance Curves: Part-Load and Efficiency Behavior")* — every curve/table attachment point this chapter flagged but didn't explain.
- *(see "Service Hot Water and Solar Water Heating")* — water heaters, thermal storage, and solar collectors, deliberately excluded here.
- *(see "Model Troubleshooting and Quality Assurance")* — the Validation dialog's loop setpoint-vs-sizing-design-temperature check, and `NodeReport` for confirming what a node's actual runtime value is rather than trusting a dialog.

End of draft chapter. Outstanding items for the next pass: supply the screenshots/diagrams marked above; verify `HeatPump:WaterToWater:ParameterEstimation`'s field set and companion-pairing mechanism directly against its live dialog rather than this draft's flagged-uncertain description; confirm the exact UI label and location of the Loop Conditioning Type field against a live PlantLoop dialog, since it was found in code (§2.1) but not confirmed as user-visible or reachable in the dialog.

17. Performance Curves: Part-Load and Efficiency Behavior

1. Overview

Many equipment types in EnergyPlus — coils, fans, chillers, pumps — have a rated performance point and a set of **performance curves** that describe how that performance actually changes as operating conditions change. A DX coil's rated capacity means little on its own; the `Curve:Biquadratic` behind its "Total Cooling Capacity Modifier Function of Temperature" field is what tells the simulation how that capacity rises and falls with indoor and outdoor temperature at every timestep of the run.

Using a curve in EP3 is mechanically no different from using one in raw EnergyPlus — EP3 doesn't reinterpret or simplify curve math, it just exposes the same `Curve:*`/`Table:*` objects EnergyPlus already defines. This chapter is deliberately conceptual rather than a field-by-field how-to: it covers what a curve or table represents, and — most importantly — the pitfalls that come from one being valid only over the range it was actually built from. For field-by-field input detail on any specific `Curve:*`/`Table:*` object, and for guidance on generating a new curve or table from manufacturer data, see the EnergyPlus Input/Output Reference — this manual doesn't duplicate that material. §8 covers where curves and tables live in EP3's own interface.

---

2. What a Performance Curve Is

A performance curve is a closed-form equation — a low-order polynomial, in most cases — that lets one performance quantity (capacity, energy input ratio, part-load fraction, pressure rise, and so on) be expressed as a continuous function of one or more operating conditions (temperature, flow fraction, part-load ratio) instead of requiring a distinct input for every condition the simulation might step through.

In EP3, a curve is not owned by, or bundled inside, the equipment that uses it. It's a freestanding object, created like any other through the Main Dialog's three columns — select `Curve:*`/`Table:*` in Column 1, **Add Object** in Column 2, edit its fields in Column 3 (*(see "The EP3 Interface and Model Settings", §2.1–§2.4)*); §8 below covers where curves and tables specifically show up in that list. Equipment refers to a curve purely by name, through a field whose label always ends in "…Curve Name" (e.g. "Total Cooling Capacity Modifier Function of Temperature Curve Name"). Because the reference is just a name, the same curve object can be reused across any number of pieces of equipment, and pointing a field at a different curve doesn't touch anything else on the equipment it's attached to.

> **The curve object itself never tells you what its inputs physically mean.** A `Curve:Biquadratic`'s two inputs are always just "x" and "y" — the object only tells you their *units* (via Input Unit Type for X/Y: Temperature, Dimensionless, MassFlow, etc.), never *which* temperature or *which* flow fraction they represent. That mapping is defined entirely by whichever equipment field references the curve. Before trusting a curve's shape, check that specific field's entry in the EnergyPlus Input/Output Reference — the same curve object plugged into two different fields (say, a "Function of Temperature" field and a "Function of Flow Fraction" field it was never meant for) will evaluate without error and produce a number that means nothing.

---

3. Curve Shapes at a Glance

This is a map to help you recognize a curve's shape from its coefficient fields, not an exhaustive catalog. Field labels spell the polynomial term directly ("Coefficient3 x**2", "Coefficient6 x*y"), so once the general pattern is familiar you can read any curve's shape straight off its own field list.

Curve Type	Equation	Independent Variables	Typical Use
`Curve:Linear`	$C1 + C2 \cdot x$	1	Simple proportional relationships
`Curve:Quadratic`	$C1 + C2 \cdot x + C3 \cdot x^2$	1	Part-load fraction vs. part-load ratio (PLF-PLR) curves
`Curve:Cubic`	$C1 + C2 \cdot x + C3 \cdot x^2 + C4 \cdot x^3$	1	Higher-order single-variable fits
`Curve:Biquadratic`	$C1 + C2 \cdot x + C3 \cdot x^2 + C4 \cdot y + C5 \cdot y^2 + C6 \cdot xy$	2	The most common curve in an HVAC model — DX coil capacity/EIR as a function of two temperatures
`Curve:Bicubic`	Biquadratic terms, plus $C7 \cdot x^3 + C8 \cdot y^3 + C9 \cdot x^2y + C10 \cdot xy^2$	2	Higher-order two-variable fits (e.g. some chiller performance curves)
`Curve:QuadraticLinear`	$C1 + C2 \cdot x + C3 \cdot x^2 + C4 \cdot y + C5 \cdot xy (+ C6 \cdot x^2y)$	2	Mixed-order two-variable fits

Beyond these, EnergyPlus defines a number of specialty forms — `Curve:Exponent`, `Curve:Sigmoid`, `Curve:FanPressureRise`, `Curve:RectangularHyperbola1/2`, `Curve:ExponentialDecay/DoubleExponentialDec` `Curve:QuadLinear/QuintLinear`, and more — used by particular objects (fan pressure-rise modeling, absorption chillers, EMS). EnergyPlus Engineering Reference: Performance Curves has the complete list and every equation form.

---

4. The Valid Input Range — and What Happens Outside It

Every curve carries **Minimum Value of x / Maximum Value of x** (and the same pair for y, on two-variable forms). These fields define the range of conditions the coefficients were actually fit to — typically a handful of manufacturer rating points or a published reference curve — and the equation has no established validity outside that range.

EnergyPlus doesn't refuse to evaluate a curve outside that domain, and it doesn't extrapolate the polynomial either. Per EnergyPlus Engineering Reference: Performance Curves, each independent variable is clamped to its Minimum/Maximum Value before being plugged into the equation — a simulation timestep with a condition outside the fitted range is evaluated *at the boundary value* instead. That clamp keeps the polynomial from being extrapolated, but it is not the same thing as

the equipment being able to operate at those conditions at all. A clamped curve keeps reporting whatever performance it calculated at the boundary, indefinitely, no matter how far past that boundary the actual operating condition goes — as far as the curve itself is concerned, nothing changes once the boundary is reached. Whether the equipment actually shuts down beyond some condition is a separate question, controlled by separate fields on the equipment itself — a DX coil, for example, has its own **Minimum Outdoor Dry-Bulb Temperature for Compressor Operation** field, which locks the compressor out below a set temperature independent of anything its performance curves say. Leaving that kind of field unset, or set too low, while relying on curve clamping alone can make a unit appear to keep delivering rated-adjacent performance at outdoor conditions it would never actually run at.

The clamp also depends entirely on the Minimum/Maximum fields reflecting where the underlying data actually stops. Two ways that breaks in practice:

- **The fields are widened past the actual fitted range** — e.g. out of a mistaken instinct that a wider domain is "safer" or more general, or because coefficients were copied from one source without also copying that source's fit range. The curve now gets evaluated (not clamped) at conditions the original data never covered.
- **The fields are left blank on a hand-entered curve.** A blank Minimum/Maximum Value field is treated as no limit at all — every condition the simulation ever produces gets run through the raw polynomial.

Either way, the result is a smooth, plausible-looking number with no physical grounding — and nothing in EP3's interface will visually flag it (see §8).

Worked example

One of EP3's default packaged-terminal-heat-pump template curves — "Total Cooling Capacity Modifier Function of Temperature" — is a `Curve:Biquadratic` with:

C1 = 2.015235142, C2 = −0.120393178, C3 = 0.004278319, C4 = −0.000285454, C5 = −0.0000485775, C6 = −0.000345086

fit over x (entering coil wet-bulb temperature) from 13.89 to 22.22°C (57–72°F), and y (outdoor dry-bulb temperature) from 23.89 to 46.11°C (75–115°F). (What x and y physically represent here comes from the referencing coil object's own I/O Reference entry, not from the curve — see the callout in §2.)

At the AHRI standard rating point (x = 19.4°C / 67°F, y = 35°C / 95°F — well inside the fitted domain):

Curve = 2.015235 − 0.120393(19.4) + 0.004278(19.4²) − 0.000285(35) − 0.0000486(35²) − 0.000345(19.4)(35) ~ **0.986**

A capacity modifier of ~0.99 at the rating point is exactly what you'd expect — the coil delivers essentially its rated capacity there.

Now push the outdoor temperature to 50°C (122°F) — a plausible design-day condition in a hot climate, but beyond the curve's fitted maximum of 46.11°C:

- **Correctly clamped** (what EnergyPlus actually does): y is capped at 46.11°C before evaluating → curve ~ **0.865**.

- **If the Maximum Value of y field had instead been widened to, say, 60°C**, the curve would evaluate at the raw y = 50°C, unclamped → curve ~ **0.759** — about 12% lower than the properly-clamped result, from a curve that was never actually validated past 46.11°C in the first place.

The Minimum/Maximum Value fields aren't a formality — set them to match whatever domain the source data (a manufacturer curve-fit report, a DOE prototype curve, etc.) actually covered, not to whatever range seems generously wide.

> *Figure placeholder — the Curve:Biquadratic dialog for the example curve above, with Minimum/Maximum Value of x and y fields circled, and a small inset plot showing the curve surface with the fitted domain shaded and the region beyond it left unshaded, to visually distinguish "fit" from "extrapolated-and-clamped."*

---

5. Minimum/Maximum Curve Output — a Separate Clamp

Minimum Curve Output and **Maximum Curve Output** clamp the *result* of the polynomial, after evaluation — a distinct mechanism from the input-domain clamp in §4, and not automatically related to it. A pair of inputs can sit correctly inside a curve's Minimum/Maximum Value domain and still produce an output that's outside what's physically sensible, simply because the fitted data didn't densely sample every combination inside that domain box.

This matters most for curves whose output has a hard physical floor or ceiling that the fit itself doesn't guarantee. A part-load-fraction (PLF-PLR) curve is the common case: EnergyPlus expects PLF to never fall below the part-load ratio itself (a coil can't appear to run for less time than the energy it draws implies), but a low-order quadratic fit through a handful of PLR data points can dip below that line at a PLR the original data barely touched. EnergyPlus Engineering Reference: DX Cooling Coil Models documents the specific runtime check EnergyPlus applies to this curve type; Minimum Curve Output is the field-level tool for enforcing the same floor directly, independent of whatever the fit alone would produce.

---

6. Tracing a Curve's Usage — View/Replace References

Because a curve is a freestanding, named object rather than something bundled inside its equipment, finding *which equipment currently uses a given curve* isn't something you can see from the curve's own fields. EP3's generic object-library interface (the same one used for Schedules, curves, and tables) surfaces this directly: any object that's referenced anywhere in the current model shows a **"View/Replace References"** button alongside its Delete/Copy buttons.

Clicking it lists every field across the model currently pointing at that specific curve, grouped by which EEM each occurrence belongs to. Occurrences in the EEM you're currently viewing let you pick a replacement curve directly from that list — swap one curve for another without navigating to the referencing equipment's own dialog at all. Occurrences in other EEMs are shown for awareness only; editing them requires switching to that EEM first.

This is also the mechanism that fires automatically if you try to delete a curve that's still referenced — you're asked to pick a replacement (or clear back to the referencing field's default) for every occurrence before the delete goes through.

For field-by-field detail on which curve fields a given piece of equipment carries, see that object's entry in the EnergyPlus Input/Output Reference — EnergyPlus Input/Output Reference.

> *Figure placeholder — the View/Replace References dialog opened from a curve row, showing a list of referencing fields grouped by EEM, with a replacement dropdown next to the current-EEM occurrences.*

---

7. Performance Tables — Tabulated Data Instead of an Equation

A **performance table** (`Table:Lookup`, built from a `Table:IndependentVariable` for each axis, bundled into a `Table:IndependentVariableList`) is an alternative to a curve for representing the same kind of relationship — used when the underlying data doesn't fit a low-order polynomial well, or when you'd rather use raw catalog data directly instead of pre-fitting it to an equation.

The same domain concern from §4 applies, but tables give you an explicit choice EnergyPlus makes for curves automatically. Each `Table:IndependentVariable` carries its own **Interpolation Method** (Linear or Cubic, between data points) and **Extrapolation Method** (Constant or Linear, beyond the last data point) — Constant behaves like a curve's clamp-to-boundary, but Linear extrapolation deliberately projects the fitted trend *past* your last real data point. Choosing Linear extrapolation means you are knowingly extending behavior beyond validated data, not falling into it by accident — a meaningfully different risk than a curve, where clamping is forced and can't be turned off.

Data for a table can be entered directly as inline values, or sourced from an external file (a column in a CSV, referenced by column number and starting row) — useful for bringing in a full manufacturer performance map without retyping it. A table also supports its own **Normalization Method/Normalization Divisor**, letting raw catalog values be scaled to a fraction-of-rated basis without altering the source data itself.

This chapter doesn't walk through wiring the three table objects together field-by-field — EnergyPlus Input/Output Reference: Table:Lookup covers that mechanically. The concept to take away is: same valid-range discipline as a curve, but with an extrapolation choice that's explicit rather than automatic.

---

8. Editing a Curve or Table in EP3

Curves and tables live in the Main Dialog's Column 1 under the "Performance Curves" and "Performance Tables" groups respectively, alongside Schedules and every other freestanding object type (*(see "The EP3 Interface and Model Settings", §2.1–§2.4)*) — not in the Components dialog used for HVAC parts.

The curve/table dialog itself has no graphing or plotting built in — coefficients and data points are entered and reviewed as bare numbers, with no chart shown alongside them. EP3's AI assistant fills part of this gap on request: it can plot a curve's shape, or the shape of multiple curves - for example: every curve attached to a given piece of equipment, without you having to build the chart by hand. What it won't do is help you determine a curve's coefficients in the first place — see §9. Absent asking the assistant, the arithmetic check in §4, or plotting a curve yourself in a spreadsheet

before typing its coefficients in, is the way to confirm a curve behaves as expected across its stated domain.

---

9. Why EP3's AI Assistant Won't Generate Curves for You

EP3's built-in AI assistant is explicitly blocked from creating `Curve:*` or `Table:*` objects, even though it can plot and analyze existing ones (§8). That distinction is deliberate, not an oversight: AI-generated curve coefficients haven't been reliably sound, and a bad curve is exactly the kind of error §4 describes — smooth, plausible-looking, and invisible without independently checking the math. *(see "The EP3 AI Assistant")* covers the assistant generally.

---

10. Looking Ahead

Today, EP3's built-in HVAC templates ship with generic curves sourced from DOE prototype/reference-building data — not manufacturer-specific performance, and turning your own manufacturer catalog data into a new curve or table is still a manual, outside-of-EP3 exercise (§9). Planned future work includes building out accurate manufacturer curve libraries, and curve-generation tools that would let you produce a correctly-bounded curve or table directly from your own data inside EP3.

18. Service Hot Water and Solar Water Heating

1. Overview

This chapter covers water heaters, heat pump water heaters, chilled/hot-water and ice thermal storage tanks, the `WaterUse:Equipment`/`WaterUse:Connections` pair that draws load onto a tank, and solar water heating.

All of these are `PlantLoop`-connectable objects, so this chapter assumes that you are already familiar with the branch/loop mechanics *(see "Central Plant Equipment and Plant Loops")* covers: how a branch connection is made with the Draw Branch tool, why every branch needs at least one object (or a `Pipe:Adiabatic` filler), and how Loop Conditioning Type gates which equipment a given `PlantLoop` will accept. Performance curves are covered generically in *(see "Performance Curves: Part-Load and Efficiency Behavior")*; this chapter only flags the one curve-shaped field a solar collector performance object carries (§8).

Out of scope:

- General central-plant equipment (boilers, chillers, pumps, cooling towers) — *(see "Central Plant Equipment and Plant Loops")*. A water heater is technically just another `PlantLoop`-connectable device, but EP3 gives service hot water a dedicated system-template pathway (§6), organizationally separate from central-plant equipment.
- How `WaterUse:Equipment` objects get into a model — that's covered in *(see "Internal Loads, Space Types, and Zone Types")*; this chapter picks up once `WaterUse:Equipment` already exists and needs to be drawn onto a loop.
- EEM structure and how an HVAC EEM and a Service Water Heating EEM get combined for a single simulation — *(see "Energy Efficiency Measures (EEMs)")*.
- Field-by-field detail for any object here — refer to the EnergyPlus Input/Output Reference. This chapter covers what EP3 exposes and where its behavior is EP3-specific rather than a plain IDD passthrough.

---

2. Water Heaters: `WaterHeater:Mixed` and `WaterHeater:Stratified`

A water heater is a storage tank with two independent fluid connections. The **use side** is the domestic hot water draw — it's demand-side equipment, wired to whatever consumes hot water (§7's `WaterUse:Connections`, or a plant branch feeding fixtures directly). The **source side** is optional: a second, independent node pair a tank can use to accept heat from another source (a loop, or a heat pump when packedged inside a heat pump water heater) instead of, or in addition to, its internal heating element. Both sides can be left unconnected — a tank with no source-side connection heats itself internally; a tank with no use-side connection falls back to its `Peak Use Flow Rate`/`Use Flow Rate Fraction Schedule Name`/`Cold Water Supply Temperature Schedule Name` fields to describe a draw profile without an explicit upstream object.

`WaterHeater:Mixed` models a single, well-mixed tank temperature with one heating element — a single Heater Maximum/Minimum Capacity pair, Heater Fuel Type, and Heater Thermal Efficiency. `WaterHeater:Stratified` instead models the tank as a vertical stack of temperature nodes, with up to two independently-scheduled heating elements (Heater 1 and Heater 2, each with its own setpoint schedule, deadband, capacity, and placement height), coordinated by **Heater Priority Control** (`MasterSlave` default: Heater 2 only fires once Heater 1 can't keep up; alternative `Simultaneous`,

which runs both at once). Both tank types are placeable standalone, and both are valid Tank sub-component choices for §3's `WaterHeater:HeatPump:PumpedCondenser` (§3.1).

`WaterHeater:Stratified`'s **Number of Nodes** and its twelve **Node N Additional Loss Coefficient** are editable number fields, describing how many vertical temperature nodes the tank is divided into and each node's heat loss.

WaterHeater:Mixed and WaterHeater:Stratified cover the full field set for each, including tank geometry, fuel/parasitic losses, and the use-side/source-side flow-control fields.

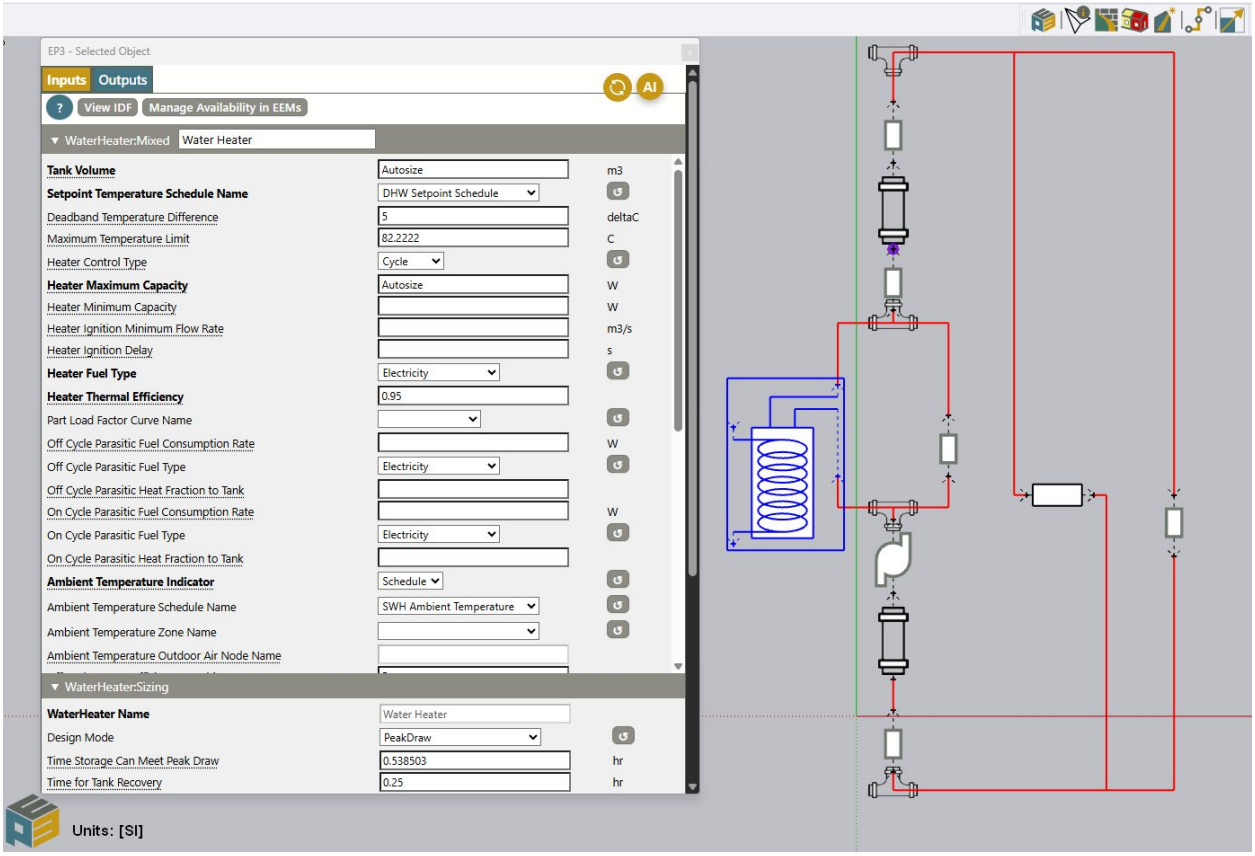


Figure 118: The WaterHeater:Mixed dialog with its auto-created WaterHeater:Sizing panel visible beneath it, and the tank wired onto a central plant loop in the HVAC Outliner at right.

---

3. Heat Pump Water Heaters

Two `WaterHeater:HeatPump:*` object families exist in EnergyPlus. One is supported in EP3's UI.

3.1 `WaterHeater:HeatPump:PumpedCondenser`

This is the heat pump water heater type EP3 exposes. It's a composite — a Reference object with three swappable sub-component slots, the same pattern *(see "Zone-Level HVAC Equipment and Zone Sizing")* describes for zone equipment:

Sub-Component	Type(s)	Notes
Tank	`WaterHeater:Mixed` or `WaterHeater:Stratified`	Swappable between the two — both are also placeable standalone (§2)
DX Coil	`Coil:WaterHeating:AirToWaterHeatPump:Wrapped`	The only valid choice EnergyPlus itself allows for this parent object
Fan	`Fan:OnOff` or `Fan:SystemModel`	

Inlet Air Configuration (required, no default) decides where the heat pump's evaporator draws its air from, and which node fields apply

Inlet Air Configuration	Requires
`OutdoorAirOnly`	Outdoor Air Node Name, Exhaust Air Node Name — no Air Inlet/Outlet Node Name pair
`ZoneAirOnly`	Air Inlet/Outlet Node Name, Inlet Air Zone Name
`Schedule`	Air Inlet/Outlet Node Name, Inlet Air Temperature Schedule Name, Inlet Air Humidity Schedule Name
`ZoneAndOutdoorAir`	Air Inlet/Outlet Node Name, Inlet Air Zone Name, Outdoor Air Node Name, Exhaust Air Node Name, Inlet Air Mixer Node Name, Outlet Air Splitter Node Name, Inlet Air Mixer Schedule Name (0 = all zone air, 1 = all outdoor air)

Compressor Setpoint Temperature Schedule Name and Dead Band Temperature Difference (5°C default) together define the cut-in/cut-out band the compressor cycles on — the setpoint schedule's note is explicit in that it should always sit above the tank's internal-element setpoint. Minimum/Maximum Inlet Air Temperature for Compressor Operation (10°C/48.9°C defaults) is a separate hard lockout independent of that band. Compressor Location and Parasitic Heat Rejection Location each independently decide whether the compressor's ambient condition and its parasitic heat affect a zone's air heat balance — `Zone` for either one requires Inlet Air Configuration to already be `ZoneAirOnly` or `ZoneAndOutdoorAir`.

Tank Element Control Logic (`Simultaneous` default; alternative `MutuallyExclusive`) decides whether the tank's internal heating element and the heat pump compressor can run at once during recovery, or whether the element only kicks in once the compressor is shut down. When the Tank sub-component is `WaterHeater:Stratified`, **Control Sensor 1/2 Height in Stratified Tank** and their weights let the heat pump's control logic sample tank temperature at different heights than the tank's own Heater 1/2 elements — with a `WaterHeater:Mixed` tank, there's only one tank temperature to sample, so these fields go unused.

WaterHeater:HeatPump:PumpedCondenser covers every field exhaustively.

3.2 `WaterHeater:HeatPump:WrappedCondenser`

This is a separate EnergyPlus object type — wrapping the condenser around the tank's exterior rather than immersing it via a stratified tank's source-side nodes — it is not implemented in EP3 at this time. If you need it, please request it from the EP3 team. We can typically add such objects to EP3 within a few days.

---

4. The Sizing-Object Opt-In Mechanism

Every tank type in this chapter can opt in to include a sizing object by clicking the checkbox titled: **"Include Sizing Object in EEM [current EEM name]?"**. EP3 will choose the correct sizing object for the waterheater or thermal storage tank selected:

- **`WaterHeater:Mixed`, `WaterHeater:Stratified`, and both `ThermalStorage:ChilledWater:*`/`ThermalSt` tanks (§5)** all get a **`WaterHeater:Sizing`**, which appears in the `WaterHeater:*` or `ThermalStorage:*` dialog.
- **`ThermalStorage:Ice:Simple`/`:Detailed` (§5.2)** get a **`ThermalStorage:Sizing`**, which appears in the `ThermalStorage:*` dialog.

4.1 `WaterHeater:Sizing` and its Design Modes

`WaterHeater:Sizing`'s **Design Mode** field picks from six sizing strategies, each activating a different subset of the remaining fields:

Design Mode	Fields Used
`PeakDraw`	Time Storage Can Meet Peak Draw, Time for Tank Recovery, and — only relevant if the tank also has autosized plant-connection flow rates — Nominal Tank Volume for Autosizing Plant Connections
`PerPerson`	Storage Capacity per Person, Recovery Capacity per Person
`PerFloorArea`	Storage Capacity per Floor Area, Recovery Capacity per Floor Area
`PerUnit`	Number of Units, Storage Capacity per Unit, Recovery Capacity per Unit
`ResidentialHUD-FHAMinimum`	Number of Bedrooms, Number of Bathrooms — a fixed lookup table baked into EnergyPlus in place of a rate you supply
`PerSolarCollectorArea`	Sizes storage directly off the aperture area of whatever `SolarCollector:FlatPlate:Water` feeds the tank (§8)

4.2 Worked example: `PerPerson` design mode

`PerPerson` mode sizes the tank directly from occupancy: multiply a per-person rate by the number of people the tank serves. Suppose a design guide specifies 0.75 gal/person storage capacity and

0.25 gal/hr-person recovery capacity for a given occupancy type, and the tank serves 220 occupants (via whichever `WaterUse:Equipment` §7 wires onto it):

Storage Capacity per Person = 0.75 gal/person → Tank Volume target = 0.75 × 220 = 165 gal Recovery Capacity per Person = 0.25 gal/hr-person → Recovery Capacity target = 0.25 × 220 = 55 gal/hr

In SI, the same inputs (2.84 L/person, 0.95 L/hr-person) give 2.84 × 220 ~ 625 L ~ 165 gal and 0.95 × 220 ~ 209 L/hr ~ 55 gal/hr — matching within rounding. EnergyPlus uses these two targets, together with the tank's Setpoint Temperature and a fixed 14.4°C cold-water reference, to autosize **Tank Volume** and **Heater Maximum Capacity** on the parent tank — the exact autosizing calculation beyond this point is documented in WaterHeater:Sizing's Engineering Reference entry, not reproduced here.

4.3 `ThermalStorage:Sizing`

Ice thermal storage sizing is driven by a demand-shifting schedule: **On Peak Period Start Time** and **On Peak Period End Time** (both required, hours) bound the on-peak window the storage system is meant to serve without drawing on the primary plant equipment, and **Sizing Factor** (1.0 default) applies a multiplier on top. It's a narrower field set than `WaterHeater:Sizing`'s six design modes because it answers one question only — how much on-peak load has to be shifted — rather than several different ways to estimate a domestic draw profile.

WaterHeater:Sizing and ThermalStorage:Sizing cover every field exhaustively.

---

5. Thermal Storage Tanks

5.1 Chilled-Water and Hot-Water Tanks

`ThermalStorage:ChilledWater:Mixed`, `ThermalStorage:ChilledWater:Stratified`, and `ThermalStorage:HotWater:Stratified` are structurally the water-heater family from §2 repurposed as storage rather than active heating — same use-side/source-side node pair, same §4.1 sizing mechanism. The differences are in which capacity and loss fields apply:

Object	Capacity Field	Additional Fields	Recovery Field
`ThermalStorage:ChilledWater:Mixed`	Nominal Cooling Capacity (replaces Heater Maximum/Minimum Capacity and the fuel/efficiency fields)	Use/Source Side Heat Transfer Effectiveness (explicit fields here, not folded into a generic "effectiveness" the way §2 describes); independent Use/Source Side Availability Schedule Name fields	Tank Recovery Time (4.0 hr default) — time to *lower* the tank from 14.4°C to 9.0°C, the opposite direction of §2's Indirect Water Heating Recovery Time

Object	Capacity Field	Additional Fields	Recovery Field
`ThermalStorage:ChilledWater:Stratified`	Same as `ChilledWater:Mixed`	Adds the same vertical-noding fields §2 describes for `WaterHeater:Stratified`	Same as `ChilledWater:Mixed`
`ThermalStorage:HotWater:Stratified`	Nominal Capacity (autosize)	Top/Bottom Setpoint Temperature Schedule Name in place of a single setpoint; same Tank Height/Shape/node-loss fields as any other stratified tank	—

5.2 Ice Storage

`ThermalStorage:Ice:Simple` and `ThermalStorage:Ice:Detailed` store cooling as a phase change — energy goes in and out by freezing and melting the water in the tank, rather than by heating or cooling a fluid volume the way the tanks in §5.1 do. Both types have a single **Inlet Node Name/Outlet Node Name** pair, and that same pair handles both directions: EnergyPlus decides whether the tank is charging (freezing) or discharging (melting) at each timestep based on the water's inlet temperature and the setpoint on the tank's outlet node — there's no separate charge/discharge field or connection to configure.

Worked example, from EnergyPlus's own `5ZoneIceStorage_autosized.idf` example file: the ice tank sits on the chilled water loop's supply-side outlet branch, positioned so that everything the loop's chiller (or its bypass) produces passes through the tank before leaving the loop. Two dispatch schedules alternate which equipment the loop actually runs, using the `PlantEquipmentOperation:ComponentSetpoint` scheme *(see "Central Plant Equipment and Plant Loops", §12.2)* already introduces:

- **Overnight and early morning (10 pm–9 am):** both the chiller and the ice tank are dispatched. The chiller's own outlet setpoint schedule drives it to -5°C — far colder than the loop actually needs — while the ice tank's own outlet setpoint stays at its normal 7.22°C target. That gap is what charges the tank: the very cold water gives up cooling to freeze the tank's water reservoir, and leaves the tank warmed back toward 7.22°C.
- **Daytime, on-peak hours (9 am–10 pm):** only the ice tank is dispatched — the chiller doesn't run at all in this period. With no active cooling source ahead of it, the tank meets its own 7.22°C outlet setpoint by melting stored ice into the passing water instead.

The ice tank's own outlet setpoint never changes between these two periods. Only which equipment the loop dispatches, and what the chiller is asked to do while it's running, changes — that's the entire mechanism: one node pair, one constant target temperature, and the loop's schedule deciding whether the tank ends up freezing or melting to hit it.

ThermalStorage:Ice:Simple covers the full field set — Ice Storage Type (`IceOnCoilInternal`/`IceOnCoilExternal`, describing how the ice-charging coil sits relative to the tank), Capacity, and the `ThermalStorage:Sizing` link from §4.3 — for both `ThermalStorage:Ice:Simple` and `ThermalStorage:Ice:Detailed`.

ThermalStorage:ChilledWater:Mixed, ThermalStorage:ChilledWater:Stratified, and ThermalStorage:HotWater:Stratified cover §5.1's tanks exhaustively.

---

6. The SWH System Template Pathway

Service hot water has its own section in the EP3 **Wizard / Library**, and users have the same ability to create Service Hot Water templates as described in *(see "The HVAC Outliner and Creating HVAC Systems")*. For service hot water systems, templates should be saved to a separate library folder.

This library is reached through the *(see "The Model Wizard")*'s Service Hot Water tab. Each system created by selecting it here becomes a separate EEM, tagged as a Service Water Heating measure. Each wizard-generated HVAC system EEM will include a Service Water Heating system, so the HVAC system EEMs represent complete (HVAC and SWH) mechanical system coverage. *(see "Energy Efficiency Measures (EEMs)")* covers this EEM-pairing/chaining mechanism and how to combine EEMs for a single simulation.

---

7. `WaterUse:Equipment` and `WaterUse:Connections`

7.1 `WaterUse:Equipment` recap

`WaterUse:Equipment` is a generalized object for simulating all water end-uses. Commonly, building simulations are concerned with energy consumption and are most concerned with **hot water** use. In EP3, `WaterUse:Equipment` is included within the `ZoneType` structure, and EP3 templates will include a `WaterUse:Equipment` covering typical hot water use for the `ZoneType`. See: *(see "Internal Loads, Space Types, and Zone Types")* for more information on `Zone` and `Space Types`.

EP3 adds three custom fields, which allow the flows in `WaterUse:Equipment` to scale with occupancy or area: **Design Level Calculation Method** (`FlowRate`/`FlowRatePerPerson`/`FlowRatePerArea`), **Peak Flow Rate per Person**, and **Peak Flow Rate per Area**. When a `WaterUse:Equipment` is assigned to multiple zones, rather than a single `Zone` (as it is when it's included in within a `ZoneType` object), its Peak Flow Rate is treated as the *total* across every zone on the list, split per zone by that zone's share of the list's total floor area.

7.2 `WaterUse:Connections` — the demand-side draw

A `WaterUse:Equipment` object cannot be connected to a loop or a `WaterHeater:*` — it simply describes a load. `WaterUse:Connections` is the object that connects a `WaterUse:Equipment` to a `PlantLoop`, and to a `WaterHeater:*`.

Field	Default	Notes
Supply Water Storage Tank Name	Mains water	Used if left blank or the tank is empty
Reclamation Water Storage Tank Name		
Hot Water Supply Temperature Schedule Name	Cold Water Supply Temperature Schedule Name	

Field	Default	Notes
Cold Water Supply Temperature Schedule Name	`Site:WaterMainsTemperature`	
Drain Water Heat Exchanger Type	None	Alternatives: CounterFlow, CrossFlow, Ideal — for greywater heat recovery
Drain Water Heat Exchanger Destination	Plant	Alternatives: Equipment, PlantAndEquipment
Drain Water Heat Exchanger U-Factor Times Area		
Connections		Array of `WaterUse:Equipment Name` entries — the explicit list of which loads this `WaterUse:Connections` serves

7.3 Worked walkthrough: the unassigned-equipment auto-pickup

If a `WaterUse:Connections` object's Connections array is empty at export time, it auto-adopts every `WaterUse:Equipment` in the model not already explicitly claimed by some other `WaterUse:Connections`. When this occurs, you may see the following warning in the export errors dialog:

> *"WARNING - [name] does not have any WaterUse:Equipment assigned. All previously unassigned WaterUse:Equipment ([names]) will be assigned to this WaterUse:Connections."*

This is EP3 functionality, EnergyPlus's `WaterUse:Connections` definition has no such fallback. It means a single-tank, single-`WaterUse:Connections` model works with zero explicit assignment — every `WaterUse:Equipment` the zone-type templates generated gets picked up automatically, which is the common case for the **Wizard**'s SWH systems (*(see "The Model Wizard", §5)*: *"Skipping this tab doesn't remove water use from your model — it removes what heats it"* assumes there's a `WaterUse:Connections` ready to adopt whatever `WaterUse:Equipment` already exists).

This creates a real ambiguity in models with **more than one** `WaterUse:Connections` object — separate service hot water systems for separate building wings, for instance. If two or more `WaterUse:Connections` objects each have an empty Connections array, EP3 has no way to know which system a given piece of unassigned `WaterUse:Equipment` actually belongs to.

EP3 resolves that ambiguity the same way every time rather than leaving it to chance: only one `WaterUse:Connections` — picked consistently, by name — is ever allowed to adopt the unclaimed equipment. Every other empty-Connections system in that situation exports with **no** equipment assigned instead, and logs an export error naming which system won the adoption. An empty system is obvious and gets caught during troubleshooting; equipment duplicated onto two systems at once would not have been. The **Validation dialog**'s Service Water Heating tab surfaces the same situation before you even export — it only appears when a model has two or more `WaterUse:Connections` objects competing for the same unclaimed pool, lists each one with a link to open it, and names which one will win the adoption.

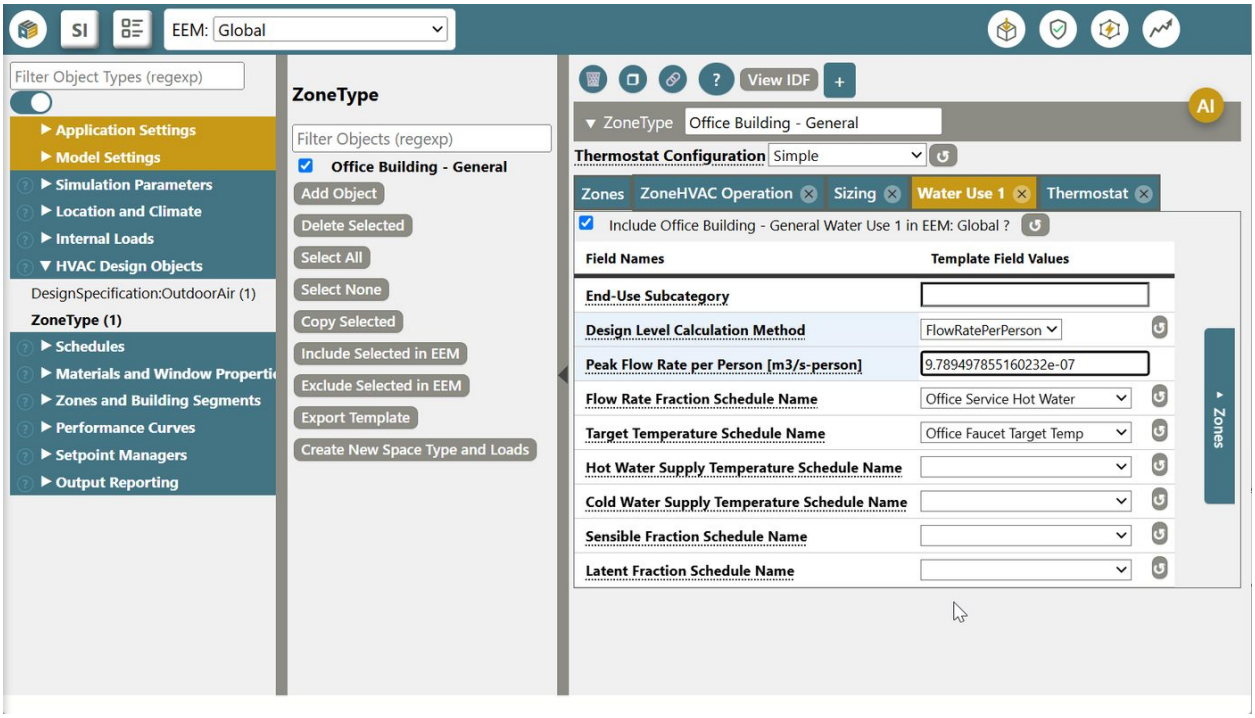


Figure 119: The Water Use tab in a ZoneType template's dialog, showing the Design Level Calculation Method, Peak Flow Rate per Person, and schedule fields EP3 bundles into the template.

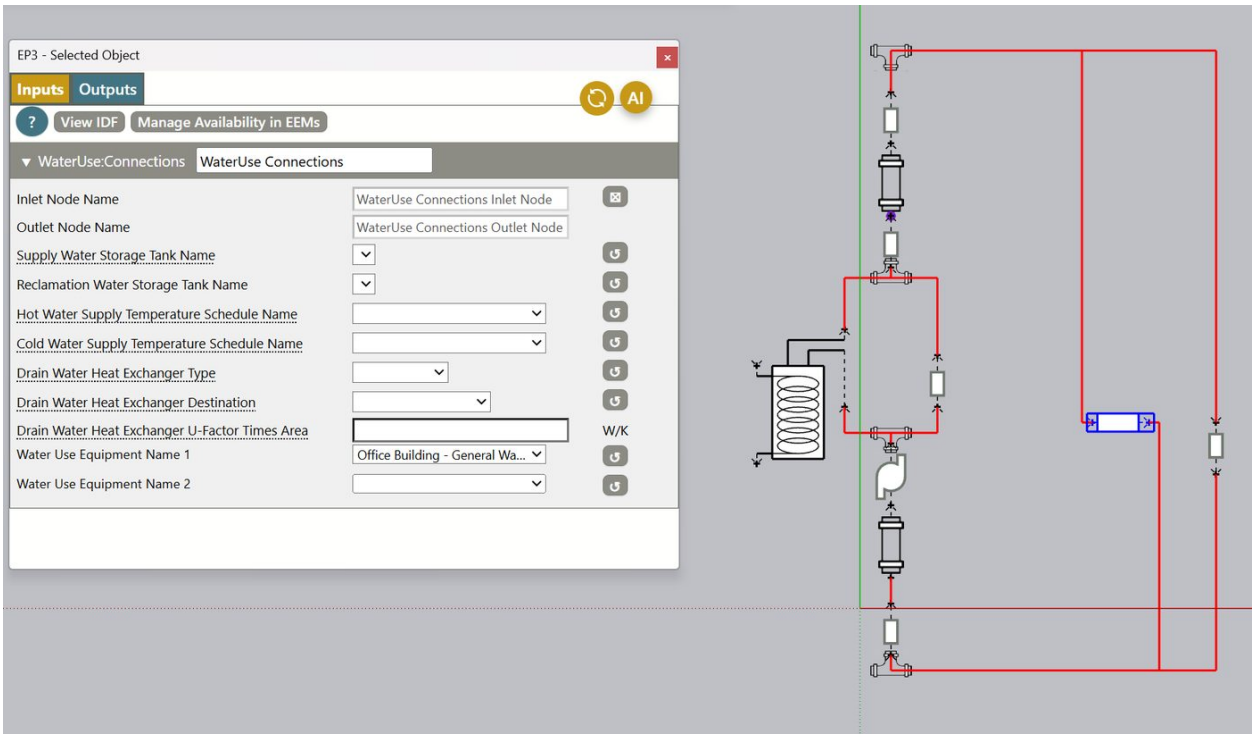


Figure 120: The WaterUse:Connections dialog, and its wiring downstream of a water heater's use side in the HVAC Outliner at right.

None of this actually decides which equipment belongs to which system — that's a modeling decision only you can make. The validation warning and export error exist to make the ambiguity visible, not to resolve it: the fix, in every case, is to give each `WaterUse:Connections` its own explicit Connections list.

---

8. Solar Water Heating

EP3 currently supports `SolarCollector:FlatPlate:Water`. EnergyPlus offers a few other `SolarCollector:*` models - please contact EP3 if you need access to one of these other models - the EP3 team can typically implement such objects within a few days. This section covers use of `SolarCollector:FlatPlate:Water` within EP3.

8.1 Why are Solar Collectors Placed on the Demand Side of a PlantLoop?

A `SolarCollector:FlatPlate:Water` has one inlet/outlet node pair and no setpoint field of any kind. EP3 enforces its demand-only role directly — the Draw Branch tool won't let you wire it onto a loop's supply side at all, the same way it blocks any other invalid connection. This follows directly from Ch.16 §2's supply/demand distinction: supply-side equipment is driven to hold a setpoint, and a solar collector can't be — it delivers whatever heat the available irradiance and flow happen to produce, demand-side behavior even though the collector is what's adding heat to the loop.

In practice, a solar collector is usually wired on the **demand return side** of its loop — after the load has already drawn what it needs, on the water's way back toward the tank or an active heat source — so it picks up whatever solar gain is available at the coldest point in the loop, before that water reaches equipment that has to make up the rest of the temperature rise. This placement reflects hydronic design practice; EP3 itself enforces only the demand-only rule above, nothing about which specific branch.

> *Figure placeholder — a screenshot from an imported EnergyPlus example file showing a SolarCollector:FlatPlate:Water wired on a loop's demand return branch, from the HVAC Outliner*

8.2 `SolarCollectorPerformance:FlatPlate` and its efficiency equation

`SolarCollector:FlatPlate:Water`'s fields are minimal — the performance behavior lives entirely on the paired `SolarCollectorPerformance:FlatPlate` object:

`SolarCollector:FlatPlate:Water` Field	Notes
SolarCollectorPerformance Name	References the paired `SolarCollectorPerformance:FlatPlate` object
Surface Name	The placeable geometry this object is mounted to
Inlet/Outlet Node Name	
Maximum Flow Rate	

`SolarCollectorPerformance:FlatPlate` Field	Notes
Gross Area	
Test Fluid	`Water` — the only choice
Test Flow Rate	
Test Correlation Type	`Average/Inlet/Outlet`
Coefficient 1/2/3 of Efficiency Equation	Used in the efficiency equation below
Coefficient 2/3 of Incident Angle Modifier	

Unlike the curve-attachment points Ch.16 and Ch.17 describe elsewhere, these coefficients are always literal numbers typed directly onto the object — there's no alternative `Curve:*` object reference to reach for here.

The efficiency equation itself, per the standard flat-plate collector test method (ASHRAE 93 / SRCC-style rating) these coefficients come from:

eta = C1 + C2 · (Tin − Tamb)/G + C3 · (Tin − Tamb)²/G

where Tin is the collector inlet fluid temperature, Tamb the ambient (outdoor) dry-bulb temperature, and G the incident solar radiation on the collector plane (W/m²). Suppose a rated collector reports C1 = 0.70 (dimensionless y-intercept), C2 = −3.5 W/m² · K (1st-order loss term), C3 = 0 (2nd-order term omitted, a common simplification), tested at Tin = 40°C, Tamb = 20°C, and G = 800 W/m²:

`eta = 0.70 + (-3.5) × (40 - 20)/800 + 0 = 0.70 - 0.0875 = 0.6125`

Useful heat delivered by a collector of **Gross Area** 4 m²: `Q = eta × G × Area = 0.6125 × 800 × 4 ~ 1,960 W`.

Raising the inlet temperature (a fuller, hotter tank, or a poorly-placed collector on the supply side if that were allowed) drives (Tin − Tamb) up and eta down — the same efficiency-falls-as-operating-temperature-rises behavior that makes indirect solar loops (a cool, dedicated collector loop feeding the tank's source side rather than the collector sitting directly on the domestic hot water path) a common design choice, though EP3 places no restriction on wiring a collector either way as long as it stays on a demand branch. SolarCollectorPerformance:FlatPlate covers the incident angle modifier terms and every other field exhaustively; the ASHRAE 93 test standard is the source for the equation form itself.

Tank sizing can be driven directly off this collector's area — §4.1's `WaterHeater:Sizing` **`PerSolarCollectorArea`** Design Mode sizes tank storage as a function of whatever `SolarCollector:FlatPlate:Wa` feeds it, rather than an occupancy- or floor-area-based rate.

---

9. Where to Look Next

- *(see "Central Plant Equipment and Plant Loops")* — `PlantLoop` topology, branch/splitter/mixer mechanics, and the Loop Conditioning Type connection gate this chapter assumes throughout.
- *(see "The HVAC Outliner and Creating HVAC Systems")* — the Draw Branch tool and the EnergyPlus-file-import template mechanism §6's SWH pathway reuses.

- *(see "The Model Wizard")* — the Service Hot Water tab, and how SWH/HVAC EEM pairing shows up during initial model setup.
- *(see "Energy Efficiency Measures (EEMs)")* — full treatment of EEM structure and the SWH/HVAC companion-chaining mechanism §6 only flags.
- *(see "Internal Loads, Space Types, and Zone Types")* — how `WaterUse:Equipment` enters a model via `SpaceType` templates, before §7's `WaterUse:Connections` ever draws it onto a loop.
- *(see "Performance Curves: Part-Load and Efficiency Behavior")* — the general curve/table mechanism, contrasted against §8.2's always-literal solar collector coefficients.
- *(see "Model Troubleshooting and Quality Assurance")* — the Validation dialog in full; its Service Water Heating tab (§7.3) is the check that catches competing `WaterUse:Connections` auto-pickups before export.

End of draft chapter. Outstanding items for the next pass: supply the §8.1 screenshot (pending your EnergyPlus example-file import).

19. Fans and Fan Power

1. Overview

Fan behavior in EP3 is governed by whichever object owns the fan: a standalone fan placed directly on an `AirLoopHVAC` branch responds to its own Availability Schedule Name and to the loop's own request for supply air flow; a fan wrapped inside an `AirLoopHVAC:Unitary*` object or a `ZoneHVAC:*` object is instead controlled by that wrapper object's own logic. Fan power, meanwhile, has to be entered in EnergyPlus's own terms — efficiency, pressure rise, or electric power — rather than the terms that typically appear on an equipment schedule or nameplate.

This chapter covers both: identifying what governs a given fan's operation, and translating fan power data into EP3's input fields. For how HVAC systems get assembled in the first place — prototype/template systems versus building from individual components — see *(see "The HVAC Outliner and Creating HVAC Systems")*. This chapter assumes a fan already exists somewhere in your model.

---

2. Fan operation — fan control depends on where a fan is housed

Fan operation and control depends as much on *where the fan lives*, as on what type of fan is chosen. There are two governing patterns to where fans live in EP3 models.

2.1 Standalone fans on an `AirLoopHVAC`

A fan placed directly on the main branch of an `AirLoopHVAC` — not wrapped inside a unitary or zone object — responds to two things:

- its own **Availability Schedule Name**, and
- the airloop's own request for supply air flow, driven by the loop's control strategy (zone demand, setpoint managers, etc.).

A fan placed on the outdoor-air-intake or relief/exhaust stream of an `AirLoopHVAC:OutdoorAirSystem`'s `OutdoorAir:Mixer` is a subcategory of this same pattern — it also responds to its own availability schedule and to the OA system's flow request. This placement is only available when the mixer sits at the airloop level; the `OutdoorAir:Mixer` built into a `ZoneHVAC:*` object doesn't expose a separate fan slot on its streams.

On its own, a standalone fan's Availability Schedule Name can force the fan off for a scheduled period. Paired with an `AvailabilityManager:NightCycle` object, the same fan can also be given "on during scheduled hours, cycling during off hours" behavior — conceptually similar to what a Supply Air Fan Operating Mode Schedule Name achieves for packaged equipment (§2.2), but built from different objects. `AvailabilityManager:NightCycle`, not a schedule field on the fan itself, is the mechanism for that behavior on a standalone fan. Its mechanics — including EP3's built-in "Night-Cycle Control?" shortcut, which auto-detects the fan's own Availability Schedule Name — are covered in *(see "Air Loops, Branches, and Nodes", §8.3)*.

2.2 Fans inside packaged equipment (`AirLoopHVAC:Unitary*` / `ZoneHVAC:*`)

When a fan is wrapped inside an `AirLoopHVAC:Unitary*` object or a `ZoneHVAC:*` object, that wrapper object takes over fan control through its own **Supply Air Fan Operating Mode Schedule**

Name field. The field's values are straightforward:

- **Schedule value 0** → the fan cycles with the coil, running only as needed to meet the thermostat setpoint.
- **Schedule value greater than 0** (typically modeled as 1) → the fan runs continuously whenever the unit is active. Any ventilation/outdoor air the unit provides is delivered throughout these hours.
- Blank defaults to cycling.

For example, a schedule with value 1 during occupied hours and 0 during unoccupied hours produces continuous fan operation (and continuous ventilation delivery) while occupied, and cycling operation while unoccupied. The field does not apply this pattern automatically; the schedule must be built to produce it.

> **Caveat — variable airflow during continuous operation:** some wrapper objects allow the fan to modulate airflow even while operating continuously (for example, variable-flow configurations of `ZoneHVAC:FourPipeFanCoil`). If your unit is one of these, don't assume continuous operation guarantees full design ventilation airflow at all times — confirm the outdoor air amount you need is actually delivered at whatever part-load flow the unit settles on.

Not every wrapper object provides outdoor air. If a unit's only job is heating and cooling — no ventilation function — modeling the fan as cycling continuously (not just during unoccupied hours) is a legitimate, and often energy-saving, choice: the fan only runs when it's actually delivering conditioning, rather than running continuously to move ventilation air the unit was never providing in the first place.

> *Figure placeholder — a simple two-panel schedule chart: left panel shows an occupied/unoccupied fraction schedule (e.g., 1.0 from 7am–6pm, 0 otherwise); right panel shows the resulting fan behavior over the same 24 hours when that schedule is used as the Supply Air Fan Operating Mode Schedule (continuous block during occupied hours, intermittent on/off spikes during unoccupied hours). Caption: "One possible use of the Supply Air Fan Operating Mode Schedule — tying it to occupancy."*
>
> *Figure placeholder — the parent composite object's dialog (e.g. AirLoopHVAC:UnitarySystem or ZoneHVAC:PackagedTerminalHeatPump) with the "Supply Air Fan Operating Mode Schedule Name" field circled/highlighted, positioned near — but visibly distinct from — the fan sub-component's own fields, to reinforce that these live on different objects.*

---

3. Where fans live in an EP3 model

3.1 Standalone fans on an `AirLoopHVAC` branch

A fan can sit directly on the main supply-air branch of an `AirLoopHVAC`, alongside coils and the outdoor air system. Where it sits relative to those other components matters:

- **Draw-through**: fan placed *after* the coils (downstream) — the fan pulls air through the coils.

- **Blow-through**: fan placed *before* the coils (upstream) — the fan pushes air through the coils.
- A **return fan** can also be placed at the very start of the branch, upstream of the outdoor air mixing box, pulling return air in from the zones before it's mixed with outdoor air.

A typical branch order, using a draw-through supply fan and a return fan, looks like:

Return Fan → Outdoor Air System (mixing box) → Heating Coil → Cooling Coil → Supply Fan (draw-through)

The fan object type on a given branch is chosen to match the system: a constant-volume system typically uses `Fan:ConstantVolume` for its supply (and return, if present) fan; a VAV system typically uses `Fan:VariableVolume` for both, which carries part-load power curve coefficients so fan energy tracks reduced flow rather than staying fixed.

3.2 Fans embedded inside composite (wrapper) objects

Most fans in a typical EP3 model aren't placed as standalone objects at all — they come bundled inside a composite object as a "Fan" sub-component, and the composite exposes a Fan Object Type / Fan Name pair of fields (naming varies slightly by object — some use "Supply Air Fan Object Type"/"Supply Air Fan Name," others "Supply Fan Object Type"/"Supply Fan Name") that point at the actual fan.

Composite objects that carry a fan sub-component this way include, among others: `ZoneHVAC:PackagedTerminalA` `ZoneHVAC:PackagedTerminalHeatPump`, `ZoneHVAC:FourPipeFanCoil`, `ZoneHVAC:UnitHeater`, `ZoneHVAC:UnitVentilator`, `ZoneHVAC:WaterToAirHeatPump`, `ZoneHVAC:TerminalUnit:VariableRefrigerantF` `AirLoopHVAC:UnitarySystem` (and its more specific relatives — furnace, heat-pump, and multi-speed variants), and the series/parallel fan-powered terminal units (`AirTerminal:SingleDuct:SeriesPIU:Reheat` `AirTerminal:SingleDuct:ParallelPIU:Reheat`). Which fan object types each of these accepts varies — some allow `Fan:OnOff`, `Fan:ConstantVolume`, `Fan:VariableVolume`, and `Fan:SystemModel`; others allow a narrower subset.

3.3 Fan object types, briefly

You don't need to memorize every field on every fan type to work in EP3 — the EnergyPlus Input/Output Reference is the right place for exhaustive detail. As a quick orientation, with a placeholder link to that reference for each object:

Object	Typical role	I/O Reference
`Fan:ConstantVolume`	Fixed airflow whenever running. Common supply/return fan for single-zone and constant-volume systems, and a common choice inside many composite objects.	Fan:ConstantVolume

Object	Typical role	I/O Reference
`Fan:VariableVolume`	Airflow modulates continuously; carries part-load power curve coefficients (*(see "Performance Curves: Part-Load and Efficiency Behavior")*) so fan energy drops off at reduced flow. Used for VAV supply/return fans and inside composites that support variable airflow.	Fan:VariableVolume
`Fan:OnOff`	Built around cycling operation, with a part-load power/efficiency curve for that mode. **Restricted to use as a sub-component fan inside `AirLoopHVAC:Unitary*` or `ZoneHVAC:*` equipment — not valid as a standalone `AirLoopHVAC` branch fan.**	Fan:OnOff
`Fan:SystemModel`	The modern, more flexible fan object — can represent constant-volume, discrete multi-speed, or continuous variable-speed behavior, with several ways to specify power (efficiency + pressure, or power directly per unit flow). Generally the recommended choice for new work where it's accepted.	Fan:SystemModel
`Fan:ComponentModel`	A detailed physical model (motor, drive, housing sub-models) for higher-fidelity fan analysis. More input-intensive than the others; used less often in typical prototype-system modeling.	Fan:ComponentModel
`Fan:ZoneExhaust`	A standalone zone exhaust fan, not part of the main supply-air path — used for dedicated zone exhaust rather than central air handling.	Fan:ZoneExhaust

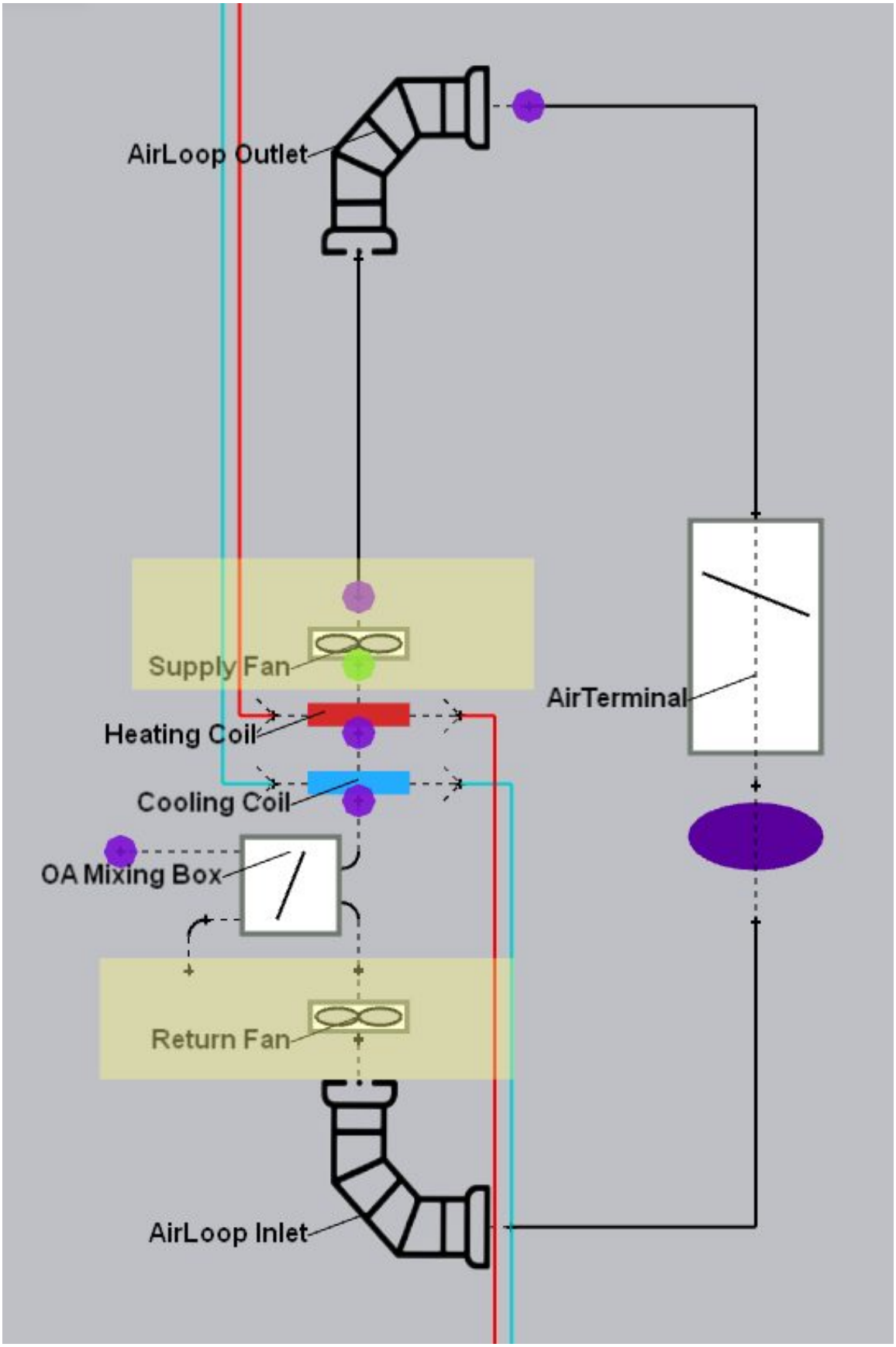


Figure 121: An `AirLoopHVAC` branch with a draw-through supply fan and a return fan.

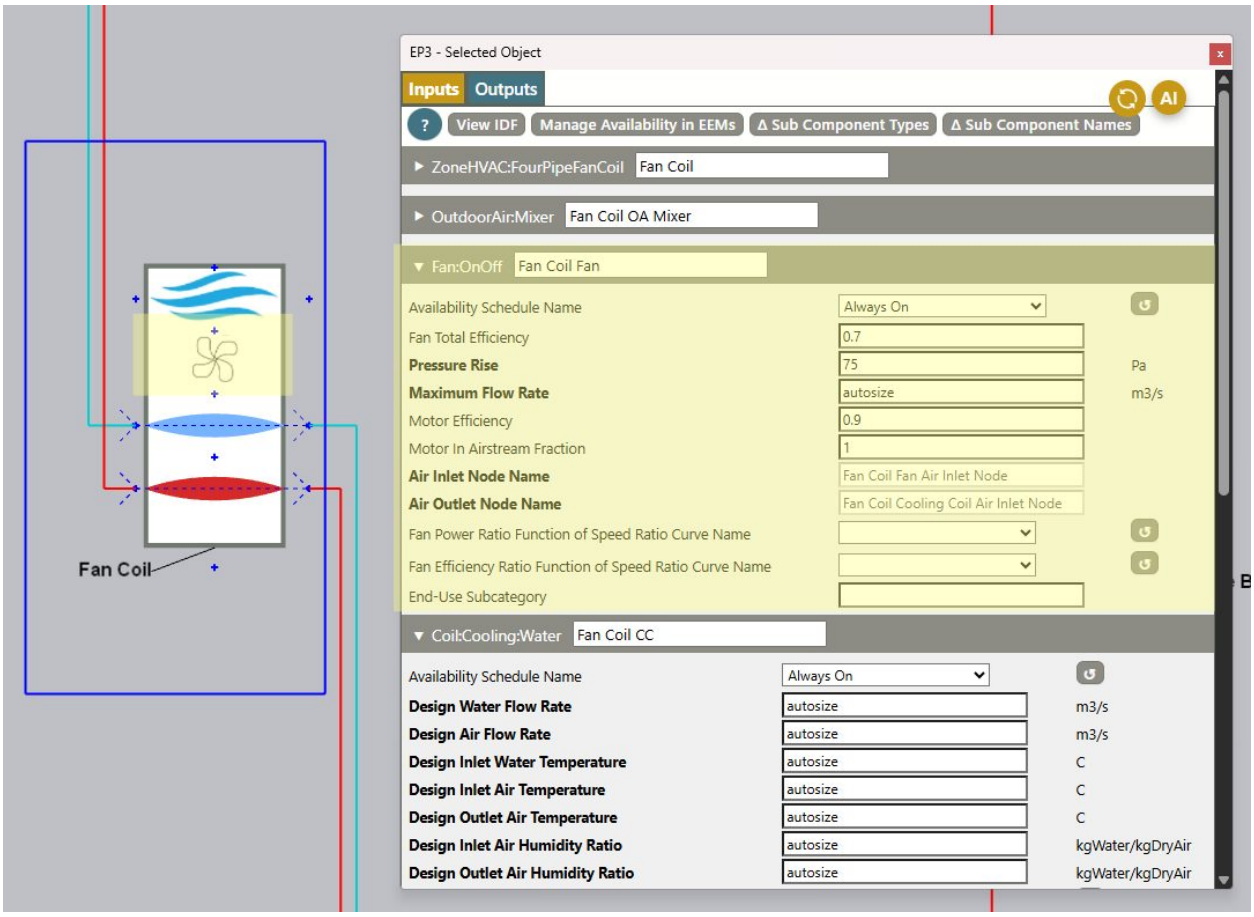


Figure 122: A `ZoneHVAC:FourPipeFanCoil` object with its `Fan:OnOff` sub-component expanded.

4. Availability Schedule Name — mechanics, a common pitfall, and zone-type defaults

4.1 Availability Schedule Name (on the fan itself)

Every fan object carries its own **Availability Schedule Name**. This is a hard gate: while the schedule evaluates to 0, the fan cannot run, regardless of what any operating-mode schedule, thermostat, or airloop flow request wants. There is one exception to this *cannot run* rule - and that is when the `Fan` sits within a system that is controlled by an `AvailabilityManager`: *(see "Controllers and Availability Managers", §3.1, §3.3)* covers where the list is built and attached, and how its status overrides the fan.

The field's description is direct: *"Availability schedule name for this system. Schedule value > 0 means the system is available. If this field is blank, the system is always available."* Leaving the field blank defaults to always-available.

An Availability Schedule Name can be set on almost any piece of equipment — fans, coils, and beyond — not only fans. Pointing it at the wrong schedule (an occupancy schedule, for example, rather than a genuine availability schedule) makes the equipment fully unavailable during "off" hours: no heating, no cooling, and no ventilation air. Unmet heating/cooling hours and reduced ventilation results often trace back to an availability schedule set this way.

Recommendation for fans within Composite Systems: unless a fan genuinely needs to be *incapable* of running for some period (a planned outage, staged commissioning, equipment not yet installed in phased construction), leave its Availability Schedule Name blank or point it at an "Always On" schedule. Let occupancy-driven behavior — cycling during unoccupied hours, running continuously while occupied — happen through the Supply Air Fan Operating Mode Schedule Name instead (§2.2, for packaged equipment) or `AvailabilityManager:NightCycle` (for standalone airloop fans, §2.1), not through the Availability Schedule.

For a standalone fan on an `AirLoopHVAC` branch, Availability Schedule Name is the *only* schedule field the fan itself has — there's no separate operating-mode field to fall back on. Used alone, it's a straightforward on/off gate. Paired with an `AvailabilityManager:NightCycle` object on the airloop, it becomes the basis for "on during scheduled hours, cycling to meet setpoint otherwise" behavior — see *(see "Air Loops, Branches, and Nodes", §8.3)*.

4.2 Setting defaults at the Zone Type level (`ZoneHVAC` equipment only)

This mechanism applies to `ZoneHVAC:*` equipment; it has no equivalent for standalone `AirLoopHVAC` fans or `AirLoopHVAC:Unitary*` objects.

Rather than setting an operating schedule on every individual piece of `ZoneHVAC` equipment, EP3 lets you set a *default* once per zone type, under **ZoneType → ZoneHVAC Operation**:

- **"ZoneHVAC Availability Schedule Name"**
- **"ZoneHVAC Supply Air Fan Operating Mode Schedule Name"**

These work like the equipment-level fields described above, but they apply as the default for *any* `ZoneHVAC` equipment placed on zones carrying that zone type. That means if you assign, say, `ZoneHVAC:FourPipeFanCoil` to zones belonging to three different zone types (Office, Conference, Storage), each zone type's default schedule flows automatically to the fan coils serving its zones, without editing each individual unit. The same two fields — along with two more, unrelated to fan

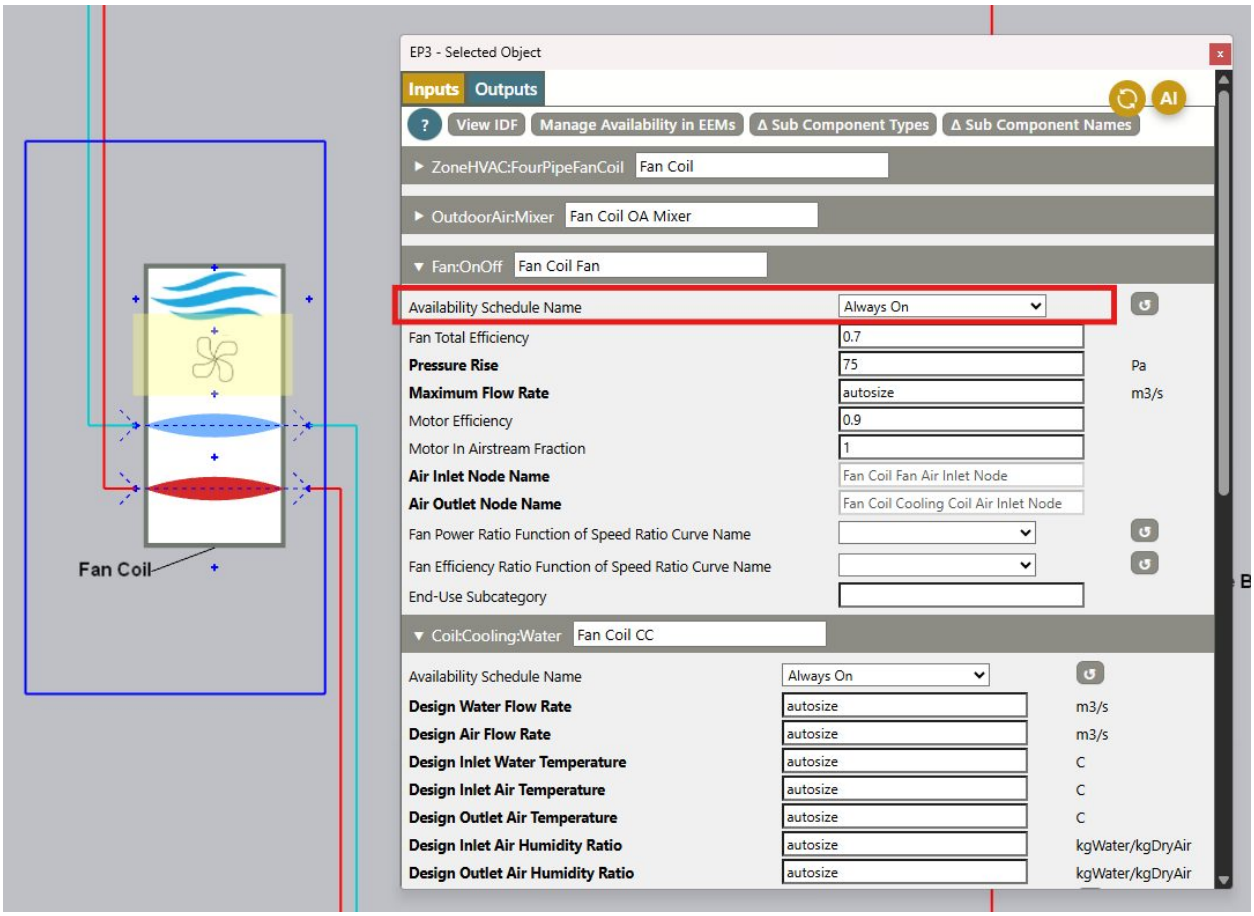


Figure 123: The fan's own Availability Schedule Name field, set to "Always On".

control, that default a zone's outdoor-air and zone-sizing objects — live directly on the `ZoneType` object itself; *(see "Internal Loads, Space Types, and Zone Types", §4.2)* covers the full field set.

The `Zone` object's own dialog carries the same fields, and a value entered there is the per-zone override beneath the zone-type default — for the occasional zone that needs to deviate from its zone type's normal schedule without creating a whole new zone type just for that one exception. In practice, think of it as three levels, checked in order of specificity: an explicit setting directly on a piece of equipment overrides everything; short of that, the `Zone`'s own override applies if one is set; short of that, the zone type's default applies.

When to use which: set the schedule at the zone type level as the default whenever a group of zones share the same operating pattern (the normal case); this reduces the number of settings to maintain and the risk of them diverging. Reserve a per-zone or per-equipment override for the exception that does not fit its zone type's normal pattern.

Not every piece of `ZoneHVAC` equipment picks up the cascade through a field named exactly "Availability Schedule Name" or "Supply Air Fan Operating Mode Schedule Name" — the VRF terminal unit's availability field, for one, is named **Terminal Unit Availability Schedule** instead. *(see "Zone-Level HVAC Equipment and Zone Sizing", §4.3)* covers which field on which equipment object actually consumes each cascade default, exceptions included.

---

5. Translating engineering fan power into EP3 inputs

EnergyPlus's simple fan objects (`Fan:ConstantVolume`, `Fan:VariableVolume`, `Fan:OnOff`) relate electric input power to airflow and pressure rise through a single field, **Fan Total Efficiency**:

Electric Power [W] = (Airflow [m³/s] × Pressure Rise [Pa]) ÷ Fan Total Efficiency

Source: this is the fan power relationship EnergyPlus documents for its simple fan objects (EnergyPlus Engineering Reference, Fans).

You don't need to hand-convert units yourself before entering values — EP3 converts whatever unit system you're displaying (IP or SI) to and from the SI values it stores internally. What actually needs translating is the *quantity itself*: many standards and specifications state fan power as a power density — Watts per unit airflow — rather than as a Pressure Rise and Fan Total Efficiency pair. The worked examples below start from that common case, once in IP units and once in SI, so you can check the two against each other.

5.1 Worked example, IP units — from a fan power allowance in W/cfm

Suppose a specification or standard sets a fan power allowance of **0.75 W/cfm** (illustrative — §6 gives real code-derived values), and you're assuming a **Fan Total Efficiency of 0.65** for the fan you'll select.

The classic IP fan-power relationship (documented in ASHRAE Fundamentals and standard fan-industry references) relates brake/air horsepower to airflow and static pressure through a single constant:

Air Power [hp] = (Airflow [cfm] × Pressure [in. w.c.]) ÷ 6,356

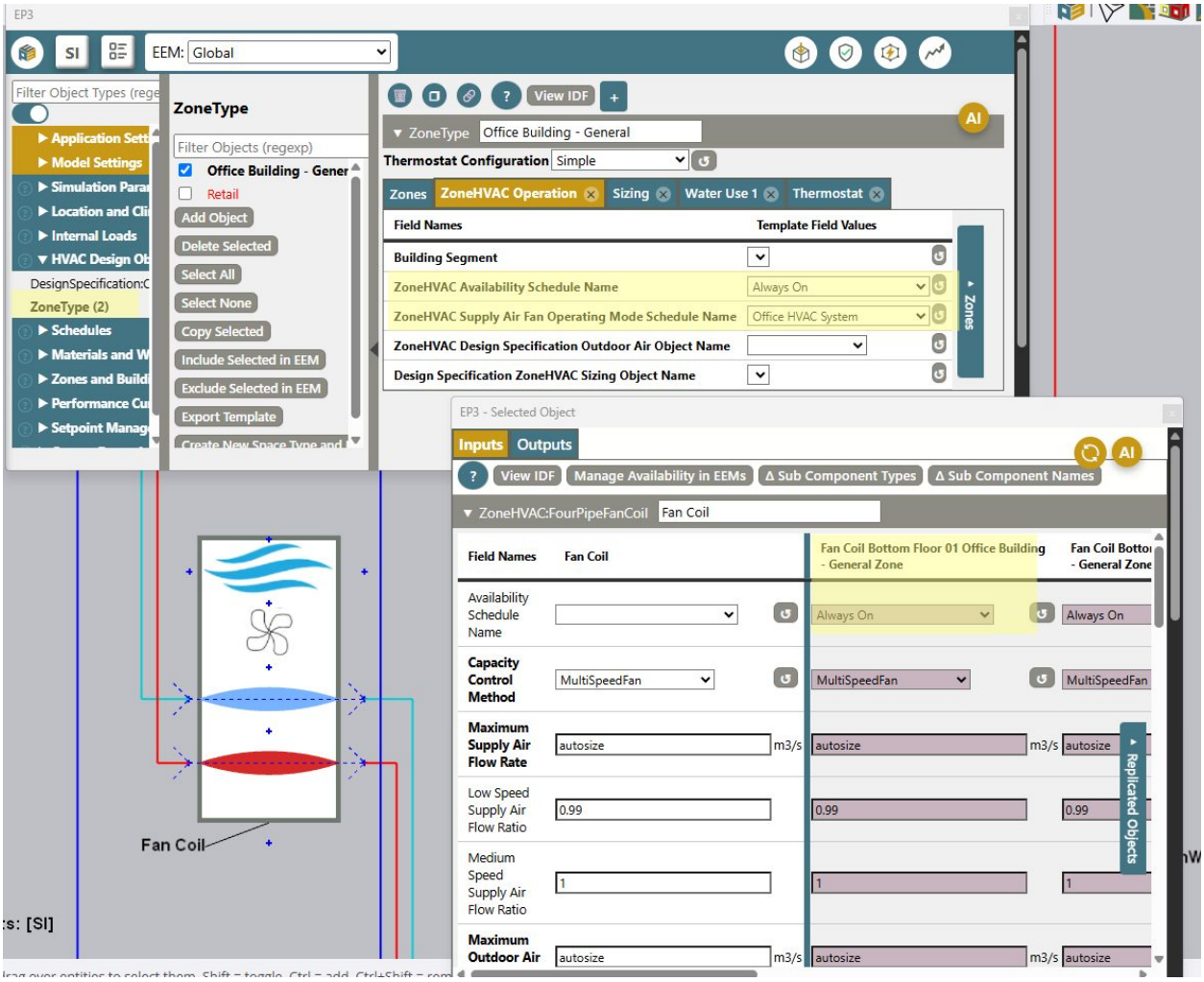


Figure 124: The ZoneType dialog's "ZoneHVAC Operation" tab, with both default schedule fields visible together.

The constant 6,356 comes from converting cfm × in. w.c. into ft · lbf/min, then into horsepower: 1 in. w.c. ~ 5.192 lbf/ft², and 1 hp = 33,000 ft · lbf/min, so 33,000 ÷ 5.192 ~ 6,356.7.

Combining this with the electric-power/efficiency relationship above (in Watts, using 1 hp = 745.7 W) gives:

Electric Power [W] = (Airflow [cfm] × Pressure [in. w.c.] × 745.7) ÷ (6,356 × Fan Total Efficiency)

Because Electric Power = (W/cfm allowance) × Airflow, the airflow term cancels — a power-density allowance converts directly to a Pressure Rise value independent of the airflow you eventually size the fan for:

Pressure [in. w.c.] = (W/cfm allowance × 6,356 × Fan Total Efficiency) ÷ 745.7

Plugging in:

`Pressure = (0.75 × 6,356 × 0.65) ÷ 745.7 = 3,098.6 ÷ 745.7 ~ 4.16 in. w.c.`

So a 0.75 W/cfm allowance at 0.65 Fan Total Efficiency corresponds to a **Pressure Rise of about 4.16 in. w.c.** — enter that (or its Pa equivalent, next section) together with **Fan Total Efficiency = 0.65**, and EnergyPlus will reproduce the target power density at whatever design airflow you size the fan for.

Check, at a specific airflow: for a 2,000 cfm fan, this allowance implies 0.75 × 2,000 = **1,500 W**. Converting 4.16 in. w.c. to Pa (§5.2) and airflow to m³/s and plugging into the base SI equation gives 1,500 × 0.65 ÷ 0.9439 ~ 1,033 Pa — matching, within rounding, the SI-side result below.

5.2 Worked example, SI units — the same allowance in W/(m³/s)

The SI equivalent of a W/cfm allowance is commonly expressed as W per L/s or W per m³/s. Converting the same 0.75 W/cfm allowance (1 cfm ~ 0.0004719 m³/s) gives:

`0.75 W/cfm ÷ 0.0004719 m³/s per cfm ~ 1,589 W/(m³/s)` (equivalently, ~ 1.59 W/(L/s))

In SI, the airflow term cancels the same way it did in IP, so Pressure Rise follows directly from the power density and the assumed efficiency, with no separate unit-constant needed:

Pressure Rise [Pa] = Power density [W/(m³/s)] × Fan Total Efficiency

`Pressure Rise = 1,589 × 0.65 ~ 1,033 Pa`

Converting back to IP for comparison: `1,033 Pa ÷ 249.089 Pa per in. w.c. ~ 4.15 in. w.c.` — matching §5.1's 4.16 in. w.c. result to within rounding, confirming the two paths agree.

Unit conversions used above: 1 hp = 745.7 W; 1 in. w.c. ~ 249.089 Pa; 1 cfm ~ 0.0004719 m³/s ~ 0.4719 L/s.

5.3 Alternate starting point — a manufacturer's brake horsepower and static pressure

If instead you're working from an equipment submittal that states brake horsepower (BHP) directly, rather than a power-density allowance, the path is similar but starts one step earlier — from shaft power rather than electric power:

Quantity (example submittal)	Value
Design airflow	2,000 cfm (0.9439 m^3/s)
Total static pressure (TSP)	3.5 in. w.c. (~ 872 Pa)
Fan brake horsepower (BHP)	1.35 hp
Motor efficiency	0.87

Convert TSP to Pa directly for your **Pressure Rise** field (3.5 × 249.089 ~ 872 Pa), then convert BHP to electric input power, accounting for motor losses:

`Electric Power = 1.35 hp × 745.7 W/hp ÷ 0.87 ~ 1,157 W`

Solve the base equation (§5, top) for the **Fan Total Efficiency** that reproduces this electric power at the given pressure and airflow:

`Fan Total Efficiency = (872 Pa × 0.9439 m³/s) ÷ 1,157 W ~ 0.71`

Enter **Pressure Rise = 872 Pa** and **Fan Total Efficiency ~ 0.71**. A result in roughly the 0.5–0.8 range for a reasonably well-selected fan is a good internal-consistency check — §6 gives reference points to compare against.

---

6. Reference values

6.1 ASHRAE 90.1 baseline fan power (Table G3.1.2.9)

ASHRAE 90.1 Appendix G's baseline fan power method (Table G3.1.2.9) gives a code-grounded range of expected fan power by system type. The columns below apply the §5 relationships to convert each system category's power allowance into the two "gut-check" quantities that actually go into a `Fan:ConstantVolume`/`Fan:VariableVolume`-style fan object — **Pressure Rise** and **Fan Total Efficiency** — in both unit systems.

Assumptions used for this table, held constant across all rows so the comparison isolates the code's power-density differences by system type:

- Fan Total Efficiency assumed = **0.65**
- Motor Efficiency assumed = **0.90** (rows based on brake horsepower only). In EP3's actual baseline-generator implementation, motor efficiency is looked up by motor size from ASHRAE 90.1 Table G3.9.1, not held at a flat value — a smaller or larger motor than assumed here will shift these rows slightly.

The coefficients themselves reflect how EP3's own ASHRAE 90.1-2019 baseline generator implements Table G3.1.2.9 (`ASHRAE901PRM2019/hvac.rb`); consult the current published standard directly for the authoritative method and any additional adders (e.g., for MERV filtration or energy recovery) that may apply to your project.

System category	Power basis	Pressure Rise (Pa)	Pressure Rise (in. w.c.)
Simple constant-volume systems (System Types 1, 2, 9, 10)	0.3 W/cfm (flat)	413	1.66
Constant-volume systems, more complex (System Types 3–8, 12, 13)	BHP = 0.00094 × cfm	1,073	4.31
VAV systems (System Types 3–8, 12, 13)	BHP = 0.0013 × cfm	1,483	5.96
System Type 11	BHP = 0.00062 × cfm	708	2.84

(All rows: Fan Total Efficiency = 0.65. BHP-based rows additionally assume Motor Efficiency = 0.90, per the note above.)

6.2 Sanity-checking your own inputs

Whichever direction you're solving (§5.1–5.3), a derived Fan Total Efficiency in roughly the 0.5–0.8 range is typical for a reasonably well-selected fan, and the Pressure Rise values in §6.1 give a sense of the range real baseline systems span — from under 2 in. w.c. for a simple constant-volume system up to nearly 6 in. w.c. for a VAV system with more complex distribution. A manufacturer's stated BHP, a fan power allowance, and the airflow/pressure you paired it with should all be internally consistent before you move on.

---

7. Where to Look Next

- *(see "The HVAC Outliner and Creating HVAC Systems")* — assembling the airloops and zone equipment a fan lives on in the first place.
- *(see "Air Loops, Branches, and Nodes", §8.3)* — `AvailabilityManager:NightCycle` and EP3's "Night-Cycle Control?" shortcut in full.
- *(see "Internal Loads, Space Types, and Zone Types", §4.2)* — the `ZoneType`'s full cascading-defaults field set, including the two fields (outdoor air, zone sizing) unrelated to fan control.
- *(see "Zone-Level HVAC Equipment and Zone Sizing", §4.3)* — which field on which specific `ZoneHVAC` equipment object consumes the cascading Availability/Operating Mode Schedule defaults.
- *(see "Performance Curves: Part-Load and Efficiency Behavior")* — the part-load power curves behind `Fan:VariableVolume` and similar fan types.
- *(see "Controllers and Availability Managers")* — `AvailabilityManager` control-strategy detail beyond the Night-Cycle shortcut above.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

20. Controllers and Availability Managers

1. Overview

This chapter covers two fairly different controls mechanisms: `Controller:WaterCoil`, and `AvailabilityManager`.

`Controller:WaterCoil` drives a water coil's flow rate to hit a setpoint value that a `SetpointManager` places at the coil's sensor node. Building and attaching a `SetpointManager` at the outlet node of a water coil is covered in *(see "Air Loops, Branches, and Nodes", §8.1)*; this chapter covers the controller side of that pairing.

`AvailabilityManager` objects decide whether a piece of equipment can run at all, independent of any setpoint. A user builds one or more `AvailabilityManager` objects and bundles them into an `AvailabilityManagerAssignmentList`, which a loop (or other equipment) then references. Building that list and attaching it to a loop is covered in *(see "Air Loops, Branches, and Nodes", §8.2)*; this chapter covers what each manager type does once it is in the list. `AvailabilityManagers` are not required for simulation, and many models won't need them at all, but they are incredibly useful for modeling more advanced operation and controlls.

`Controller:OutdoorAir` and `Controller:MechanicalVentilation` are also `Controller:*` objects, but their fields are entirely about outdoor-air and ventilation strategy — economizer operation, demand-controlled ventilation — so they are covered in *(see "Outdoor Air, Ventilation, and Energy Recovery")* instead.

---

2. Controller:WaterCoil

2.1 Purpose

A controller reads a value at one node (the sensor node) and adjusts an actuator at another node (the actuator node) until the sensor node reaches a target value. For `Controller:WaterCoil`, the actuator is always the coil's water inlet node, and the actutor control variable is `Flow`. The target comes from a `SetpointManager` placed at the sensor node (typically the coil's air outlet node). There is no setpoint value entered directly on the controller; it inherits whatever value the setpoint manager places at that node and adjusts flow each iteration until the sensor node matches it.

EnergyPlus requires a controller to operate any `Coil:Cooling:Water`, `Coil:Heating:Water`, or `Coil:Cooling:Water:DetailedGeometry` placed directly on an AirLoopHVAC branch, outside a composite object — without one, EnergyPlus has no way to determine how much water the coil should pass, and the simulation cannot run. You will see a `Controller:WaterCoil` in the object dialog for any water coil that sits directly on an AirLoopHVAC branch. EP3 populates the Sensor Node Name and Actuator Node Name fields automatically.

Water coils used as a sub-component inside a composite object — a Four-Pipe Fan Coil's cooling coil, or a VAV Reheat terminal's hot water coil, for example — are controlled through that composite's mechanism instead, and do not need a `Controller:WaterCoil`. *(see "Zone-Level HVAC Equipment and Zone Sizing")* covers composite-specific coil control.

2.2 Fields

- **Control Variable** — what the sensor node value represents: `Temperature`, `HumidityRatio`, or `TemperatureAndHumidityRatio`. EP3 defaults this to `Temperature`, this is the case that applies to an ordinary heating or cooling coil paired with a temperature-based `SetpointManager`. `HumidityRatio` and `TemperatureAndHumidityRatio` require a paired `ZoneControl:Humidistat` and a humidity-capable `SetpointManager` (`SetpointManager:SingleZone:Humidity:Maximum` or one of the multizone humidity variants) — set up that pairing before switching Control Variable away from the default.
- **Action** — `Normal` or `Reverse`, describing which direction increasing the actuator moves the sensor value. Left blank, EP3 sets this automatically from the coil type when the coil is prepared for export: `Normal` for a heating coil, `Reverse` for a cooling coil. Override it only when a specific system configuration needs the opposite convention.
- **Actuator Variable** — fixed to `Flow`.
- **Sensor Node Name / Actuator Node Name** — auto-populated per §2.1, not editable.
- **Controller Convergence Tolerance** — EP3's default for this field is blank; EnergyPlus calculates a tolerance directly from the coil's design conditions when the field is left blank.
- **Maximum Actuated Flow / Minimum Actuated Flow** — the water flow range the controller is allowed to command. Maximum Actuated Flow defaults to `Autosize`, tying it to the coil's design flow rate; Minimum Actuated Flow defaults to 0.

  Figure placeholder — the Controller:WaterCoil dialog for a coil placed directly on an AirLoopHVAC branch, with the Sensor Node Name and Actuator Node Name fields visibly grayed out / non-editable, and the Control Variable, Action, Controller Convergence Tolerance, and Maximum/Minimum Actuated Flow fields visible and editable.

---

3. Availability Managers — the pattern

3.1 What is an AvailabilityManagerAssignmentList?

An `AvailabilityManagerAssignmentList` is an ordered bundle of one or more `AvailabilityManager` objects. Each `AvailabilityManager` object is built individually through the generic object-library Add mechanism — the same one used for schedules and curves. A loop, pump or other object references exactly one list, through a dropdown field on its dialog; *(see "Air Loops, Branches, and Nodes", §8.2)* covers building and attaching a list to an `AirLoopHVAC`, and the same mechanism attaches to a `PlantLoop` (*(see "Central Plant Equipment and Plant Loops")*).

The same list mechanism also attaches directly to certain zone equipment: `ZoneHVAC:TerminalUnit:VariableRefr` carries an Availability Manager List Name field independent of any airloop or plant loop. *(see "Zone-Level HVAC Equipment and Zone Sizing")* covers where that fits alongside the equipment-level Availability Schedule fields most `ZoneHVAC` objects use instead.

Each `AvailabilityManager` object can appear in a list only once.

3.2 Multiple managers in one list

Each `AvailabilityManager` in a list resolves, at a given timestep, to one of a small set of status values — most commonly `ForceOff`, `CycleOn` (or, for an `AirLoopHVAC` only, `CycleOnZoneFansOnly`),

or `NoAction`, meaning it defers to the rest of the list. How EnergyPlus combines these into one decision depends on what the list is attached to:

- **`AirLoopHVAC` and zone components** evaluate every manager in the list and combine their statuses by a fixed precedence, regardless of the order the managers appear in: `ForceOff` wins if any manager returns it; short of that, `CycleOnZoneFansOnly` wins if any manager returns it; short of that, `CycleOn` wins if any manager returns it; only if every manager returns `NoAction` does the loop default to off. List order does not change this outcome.
- **`PlantLoop`** works differently: managers are evaluated in the order they appear in the list, and the first one to return anything other than `NoAction` sets the loop's status — managers listed after it are not consulted. List order is significant here in a way it is not for `AirLoopHVAC`.

AvailabilityManagerAssignmentList in the EnergyPlus Input Output Reference documents this precedence in full.

3.3 How the resolved status turns equipment on or off

The status an `AvailabilityManagerAssignmentList` resolves to (§3.2) is not itself an on/off command. It is checked by the loop's prime mover — a fan for `AirLoopHVAC`, a pump for `PlantLoop` — against that object's Availability Schedule Name. A `ForceOff` status turns the fan or pump off even when its Availability Schedule Name says on; a `CycleOn` status turns it on even when the schedule says off. The availability managers overrule the fan or pump's on/off schedule in both directions; only a `NoAction` status leaves the schedule's setting in effect. System Availability Managers in the EnergyPlus Engineering Reference documents this interaction directly.

3.4 Building a list with more than one manager

Two situations call for it:

- **Combining `AvailabilityManager:NightCycle` with a different manager type on the same loop** — for example, night-cycle operation together with a scheduled shutdown or a temperature-based lockout. EP3's built-in Night-Cycle Control shortcut (*(see "Air Loops, Branches, and Nodes", §8.3)*) only ever produces a single `AvailabilityManager:NightCycle`, and is unavailable once a list has been picked by hand — so this combination requires building the list manually.
- **An `AirLoopHVAC` with more than one fan running different Availability Schedule Names.** The Night-Cycle Control shortcut auto-detects one supply fan's schedule; an airloop with several fans on different schedules needs one `AvailabilityManager:NightCycle` object per fan/schedule pairing, each built and added to the list by hand. This case is uncommon.

---

4. Availability manager types

EP3 exposes twelve `AvailabilityManager:*` types. What decides each manager's on/off behavior varies by type — a schedule value, a temperature crossing a threshold, zone thermostat demand, a planned start time, or outdoor conditions favorable for ventilation.

4.1 Schedule-driven: Scheduled, ScheduledOn, ScheduledOff

- **`AvailabilityManager:Scheduled`** (AvailabilityManager:Scheduled) — a single Schedule Name field. A schedule value greater than 0 sets `CycleOn`; a value of 0 or less sets `ForceOff`. Because `ForceOff` wins over `CycleOn` (§3.2), this manager can force a system off unilaterally, but a `CycleOn` result from it can still be overridden by a `ForceOff` from another manager in the same list.
- **`AvailabilityManager:ScheduledOn`** (AvailabilityManager:ScheduledOn) — a schedule value greater than 0 sets `CycleOn`; any other value sets `NoAction`, deferring entirely to the rest of the list.
- **`AvailabilityManager:ScheduledOff`** (AvailabilityManager:ScheduledOff) — a schedule value of exactly 0 sets `ForceOff`; any other value sets `NoAction`, deferring entirely to the rest of the list.

`ScheduledOn` and `ScheduledOff` each assert a decision in only one direction, setting `NoAction` the rest of the time — the building blocks for combining a forced-on or forced-off period with another manager's normal operation in the same list. `Scheduled` is the right choice whenever a single schedule should determine availability alone, since it is the only one of the three that asserts a decision in both directions.

4.2 Temperature-threshold-driven: DifferentialThermostat, HighTemperatureTurnOff/On, LowTemperatureTurnOff/On

These five types compare a temperature at one or two nodes against a threshold and turn the system on or off when that threshold is crossed.

- **`AvailabilityManager:DifferentialThermostat`** (AvailabilityManager:DifferentialThermostat) compares a Hot Node and a Cold Node. The system turns on once the hot node's temperature minus the cold node's temperature exceeds the Temperature Difference On Limit, and off once that difference drops below the Temperature Difference Off Limit (which defaults to the On Limit if left blank). This fits a solar or waterside heat-exchange loop, where operation should depend on which side of the loop is warmer rather than on a fixed setpoint.
- **`AvailabilityManager:HighTemperatureTurnOff` / `:HighTemperatureTurnOn`** and **`:LowTemperatureTurnOff` / `:LowTemperatureTurnOn`** (AvailabilityManager:HighTemperatureTurnOff) each compare a single Sensor Node Name against a Temperature field, turning the system off or on once that node crosses the threshold, in the direction the object's name describes.

For any of these, the Sensor Node Name, Hot Node Name, or Cold Node Name can point at a physical model node, or — using the Outdoor Air Node and Outdoor Air Node Is fields EP3 adds to these objects — at outdoor conditions directly, without first locating an outdoor air node in the model. For the single-sensor types, a "Zone Node for Sensor Location" button on the object's dialog offers a further option: pointing the sensor at a zone's virtual air, inlet, return, or exhaust node rather than a node on the HVAC side of the system.

> *Figure placeholder — the AvailabilityManager:LowTemperatureTurnOff dialog, showing the Sensor Node Name field, the Temperature field, and the Outdoor Air Node / Outdoor Air Node Is fields together.*

4.3 Load-driven: NightCycle

`AvailabilityManager:NightCycle` (AvailabilityManager:NightCycle) brings a system on to satisfy zone thermostat demand during hours it would otherwise be off — distinct from the ventilation-driven types in §4.5, which run a system for outdoor-air cooling regardless of thermostat demand.

(see "Air Loops, Branches, and Nodes", §8.3) covers EP3's Night-Cycle Control shortcut — a Yes/No field on the `AirLoopHVAC` dialog that builds and auto-configures a single `AvailabilityManager:NightCycle` object without a manual placement step. Building the object directly, through the generic object-library Add mechanism, exposes its full field set and is the path required for either scenario in §3.2:

- **Applicability Schedule Name** — the schedule that determines when night-cycle logic is active at all; a value of 0 disables the manager entirely for that hour, regardless of Control Type or thermostat demand.
- **Fan Schedule Name** — the Availability Schedule Name of the fan(s) this manager should cycle. The shortcut auto-detects this from the airloop's supply fan; building the object manually requires setting it explicitly, which is what makes the multiple-fans-multiple-schedules case in §3.2 possible.
- **Control Type** — which zones' demand the manager checks, and what it does with fans while checking: `CycleOnAny`, `CycleOnAnyCoolingOrHeatingZone`, `CycleOnAnyCoolingZone`, `CycleOnAnyHeatingZone`, `CycleOnAnyHeatingZoneFansOnly`, `CycleOnAnyZoneFansOnly`, `CycleOnControlZone`, or `StayOff`.
- **Control Zone or Zone List Name** (plus the Cooling/Heating/FansOnly variants of the same field) — the zone or zone list Control Type checks against, when Control Type requires one.
- **Thermostat Tolerance** — how far outside setpoint a zone's temperature must be before the manager treats it as calling for conditioning.
- **Cycling Run Time Control Type** and **Cycling Run Time** — `FixedRunTime`, `Thermostat`, or `ThermostatWithMinimumRunTime`, controlling how long the system keeps running once a cycle starts.

4.4 Start-time-driven: OptimumStart

`AvailabilityManager:OptimumStart` brings a system on early enough to reach setpoint by a scheduled occupancy time, rather than at a fixed number of hours before occupancy every day. Applicability Schedule Name and Fan Schedule Name work as in §4.3; Control Type (`ControlZone`, `MaximumofZoneList`, or `StayOff`) and the matching Control Zone Name or Zone List Name field pick which zone(s) determine the required start time, and Maximum Value for Optimum Start Time caps how many hours before occupancy the system can start.

Control Algorithm selects how EnergyPlus estimates the needed lead time: `ConstantStartTime` (a fixed lead time, set directly in Constant Start Time), `ConstantTemperatureGradient` (a fixed rate of temperature change per hour, set in the Constant Temperature Gradient during Heating/Cooling fields), `AdaptiveTemperatureGradient` (a rate calculated from the Number of Previous Days of simulated results), or `AdaptiveASHRAE` (EnergyPlus's ASHRAE-based adaptive method). AvailabilityManager:OptimumStart documents the calculation behind each algorithm.

4.5 Ventilation-driven: NightVentilation, HybridVentilation

- `AvailabilityManager:NightVentilation` (AvailabilityManager:NightVentilation) enables outdoor-air cooling outside occupied hours when conditions favor it: a zone's temperature must be above the Ventilation Temperature Schedule Name's value, the outdoor temperature must exceed the control zone's temperature by more than the Ventilation Temperature Difference, and the outdoor temperature must stay above the Ventilation Temperature Low Limit. When active, airflow is set to the Night Venting Flow Fraction of system design flow, evaluated against the zone named in Control Zone Name.
- `AvailabilityManager:HybridVentilation` (AvailabilityManager:HybridVentilation) coordinates natural ventilation — `AirflowNetwork` openings or simple `ZoneVentilation` objects — with mechanical system operation, switching between them by a Ventilation Control Mode Schedule and a set of weather-based cutouts (rain, wind speed, outdoor temperature/enthalpy/dewpoint limits). Its field set is built around the `AirflowNetwork` mechanism specifically; *(see "Infiltration, Natural Ventilation and Airflow Networks", §5)* covers how it coordinates with `AirflowNetwork`'s opening/schedule mechanics.

---

5. Where to Look Next

- *(see "Air Loops, Branches, and Nodes", §8.1)* — building and attaching a `SetpointManager`, the object `Controller:WaterCoil` depends on for its target value.
- *(see "Air Loops, Branches, and Nodes", §8.2–8.3)* — building an `AvailabilityManagerAssignmentList` and attaching it to an `AirLoopHVAC`, plus the Night-Cycle Control shortcut.
- *(see "Central Plant Equipment and Plant Loops")* — `PlantLoop`-specific `SetpointManager` types, and attaching an `AvailabilityManagerAssignmentList` to a `PlantLoop`.
- *(see "Zone-Level HVAC Equipment and Zone Sizing")* — control for a water coil packaged inside a composite object, and the VRF terminal unit's Availability Manager List Name field.
- *(see "Outdoor Air, Ventilation, and Energy Recovery")* — `Controller:OutdoorAir` and `Controller:MechanicalVentilation`.
- *(see "Infiltration, Natural Ventilation and Airflow Networks", §5)* — how `AvailabilityManager:HybridVent` (field set above, §4.5) coordinates with `AirflowNetwork`'s opening/schedule mechanics.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

21. Outdoor Air, Ventilation, and Energy Recovery

1. Overview

This chapter covers the objects that bring outdoor air into an `AirLoopHVAC`, control how much of it enters at any given moment, and recover energy from the exhaust stream before it leaves the building: `OutdoorAir:Mixer`, `Controller:OutdoorAir`, `Controller:MechanicalVentilation`, `DesignSpecification:OutdoorAir` and its companion objects, dedicated outdoor air system configurations, zone-level ventilation equipment, and air-to-air heat exchangers.

What this chapter does not cover:

- Infiltration, `AirflowNetwork`, and natural ventilation, including `Fan:ZoneExhaust`'s make-up-air mechanics — *(see "Infiltration, Natural Ventilation and Airflow Networks").* Mechanical ventilation (this chapter) and infiltration are separate, independently-modeled outdoor-air paths; a zone can carry both at once.
- `AvailabilityManager` control strategy generally, including the ventilation-driven types (`AvailabilityManager:NightVentilation`, `AvailabilityManager:HybridVentilation`) — *(see "Controllers and Availability Managers", §4.5).*
- How `AirTerminal:SingleDuct:Mixer` is wired onto a host `ZoneHVAC` composite to combine a DOAS primary airstream with a locally-conditioned secondary airstream — *(see "Air Loops, Branches, and Nodes", §5.1).* This chapter covers the ventilation-control objects that govern how much air moves through that connection, not the wiring itself.
- `Sizing:Zone` and `Sizing:System` field-by-field detail — *(see "Zone-Level HVAC Equipment and Zone Sizing", §4)* and *(see "Air Loops, Branches, and Nodes", §4).* This chapter references specific outdoor-air-related fields on both objects where they interact with the objects covered here, without re-teaching either object in full.
- Reading output variables, unmet hours, and other means of confirming a model is behaving as configured — *(see "Viewing and Interpreting Results")* and *(see "Model Troubleshooting and Quality Assurance").*

---

2. The Outdoor Air System on an Air Loop

`OutdoorAir:Mixer` combines return air with outdoor air brought, and vents whatever return air isn't recirculated through a relief path. It exposes four nodes: **Mixed Air Node Name** (its outlet, feeding the main coils), **Outdoor Air Stream Node Name** and **Relief Air Stream Node Name** (its separate intake/exhaust branch pair, §2.1), and **Return Air Stream Node Name** (its inlet from the loop's main branch).

When an `OutdoorAir:Mixer` is on an `AirLoopHVAC` branch, you will see the input fields for the associated controller - `Controller:OutdoorAir` - in the `OutdoorAir:Mixer` dialog (§3).

2.1 The intake and exhaust branch pair

`OutdoorAir:Mixer` carries a supply and exhaust branch structure, separate from the loop's main branch. Equipment placed on the **supply** side sits between outdoor air and the Mixed Air Node. Objects placed in this location might include a fan, a preheat coil, and/or one side of an air-to-air heat exchanger (§7). Equipment placed on the **exhaust** side sits between the Relief Air Stream Node and the building exterior. Objects placed in this location might include exhaust an exhaust fan

and/or the other side of that same heat exchanger, since energy recovery needs both the incoming outdoor airstream and the outgoing exhaust airstream to pass through it. Draw Branch (*(see "The HVAC Outliner and Creating HVAC Systems", §2.3)*) works on this branch pair the same way it works on the loop's main branch.

No equipment is required on the **supply** and **exhaust** sides of an `OutdoorAir:Mixer` - components are added only as needed.

2.2 The equipment list and controller list

For those familiar with EnergyPlus, EP3 creates an `AirLoopHVAC:OutdoorAirSystem` for you on translation from EP3 to EnergyPlus - there is nothing for you to set up: its **Controller List Name** and **Outdoor Air Equipment List Name** fields, and the `AirLoopHVAC:ControllerList` and `AirLoopHVAC:OutdoorAirSystem:EquipmentList` objects they point to, are built from whatever controllers and equipment are present — every component on the intake/exhaust branch pair, plus `Controller:OutdoorAir` and, if enabled, `Controller:MechanicalVentilation` (§4.1). There's nothing to edit here beyond placing the equipment and controllers that should appear on these lists.

The HVAC Outliner with a DOAS air loop's Outdoor Air System expanded, showing its Supply and Exhaust branches with an OA mixing box and a heat recovery component.

Figure 125: The HVAC Outliner with a DOAS air loop's Outdoor Air System expanded, showing its Supply and Exhaust branches with an OA mixing box and a heat recovery component.

---

3. `Controller:OutdoorAir`

`Controller:OutdoorAir` decides how much outdoor air the mixer draws at each timestep, between a configured minimum and maximum, and whether conditions favor using outdoor air for free cooling. You will see a section for `Controller:OutdoorAir` for every `OutdoorAir:Mixer` located on an `AirLoopHVAC` branch (§2).

Minimum Outdoor Air Flow Rate and **Maximum Outdoor Air Flow Rate** bound how much outdoor air the controller will draw, both autosizable. The minimum is a hard floor: if `Controller:MechanicalVentilation` (§4.1) is also present and calls for more, the larger value governs, but this field's value is never undercut by demand-controlled ventilation reducing it further.

Any economizer type other than `NoEconomizer` (§3.1) needs a `SetpointManager` — typically `SetpointManager:MixedAir` — targeting the Mixed Air Node Name (*(see "Air Loops, Branches, and Nodes", §8.1)*). The setpoint manager is what establishes the condition `Controller:OutdoorAir` modulates the outdoor air damper toward; the economizer fields decide when and how far to open the damper, but they need a target at that node to modulate against. Where a fan sits downstream of the mixed air point, the setpoint at this node needs to account for the temperature rise that fan adds - in this case a `SetpointManager:MixedAir` is a good choice for the mixed air node.

3.1 Economizer control

Economizer Control Type selects which outdoor-air condition, if any, the controller checks before deciding whether to bring in outdoor air beyond the minimum for free cooling:

Economizer Control Type	Compares
`NoEconomizer` (default)	No economizer logic; outdoor air stays at the minimum unless `Controller:MechanicalVentilation` calls for more.
`FixedDryBulb`	Outdoor dry-bulb temperature against a fixed limit.
`FixedEnthalpy`	Outdoor enthalpy against a fixed limit.
`FixedDewPointAndDryBulb`	Outdoor dry-bulb and dew point temperature against two fixed limits.
`DifferentialDryBulb`	Outdoor dry-bulb temperature against return air dry-bulb temperature.
`DifferentialEnthalpy`	Outdoor enthalpy against return air enthalpy.
`ElectronicEnthalpy`	Outdoor humidity ratio against a curve-defined limit that varies with dry-bulb temperature.

Each type that checks against a fixed limit has a paired value field: **Economizer Maximum Limit Dry-Bulb Temperature** (`FixedDryBulb`, `FixedDewPointAndDryBulb`), **Economizer Maximum Limit Enthalpy** (`FixedEnthalpy`), **Economizer Maximum Limit Dewpoint Temperature** (`FixedDewPointAndDryBulb`), and **Electronic Enthalpy Limit Curve Name** (`ElectronicEnthalpy`, referencing a quadratic or cubic curve — *(see "Performance Curves: Part-Load and Efficiency Behavior")*). **Economizer Minimum Limit Dry-Bulb Temperature** applies regardless of type — a low-temperature cutoff below which the economizer won't operate even if the high-side comparison would otherwise allow it.

Economizer Control Action Type sets how the controller responds once the economizer is active: `ModulateFlow` (the default) lets outdoor air flow rise smoothly above the minimum toward the maximum as conditions allow, while `MinimumFlowWithBypass` holds the total supply airflow fixed and only changes the outdoor-air/return-air split.

Lockout Type disables the economizer under conditions where free cooling would work against the equipment it's paired with: `LockoutWithCompressor` disables it whenever a DX compressor is running (since the compressor's cooling would fight an economizer already bringing in cool air), and `LockoutWithHeating` disables it whenever a heating coil is active. `NoLockout` (the default) applies no such restriction. **Minimum Limit Type** controls how the minimum outdoor air flow rate itself behaves at the boundary of the economizer's operating range — consult Controller:OutdoorAir for its exact enumeration and effect.

`Controller:OutdoorAir` also carries a **High Humidity Control** field and several related fields not covered in this draft pending verification of their actual operation — see the outstanding items note at the end of this chapter.

4. Demand-Controlled Ventilation

EP3 exposes two independent mechanisms for scaling outdoor air delivery to occupancy rather than a outdoor air flow rate: a system-level controller that adjusts the airloop's shared outdoor air intake, and a per-terminal field that lets an individual VAV terminal adjust its damper position. Both read from the same `DesignSpecification:OutdoorAir` object type; what each does with it differs.

4.1 System-level: `Controller:MechanicalVentilation`

Checking **Use Controller:MechanicalVentilation?** on the `OutdoorAir:Mixer`'s dialog creates a `Controller:MechanicalVentilation` object and adds it to the outdoor air system's controller list (§2.2). **Availability Schedule Name** left blank makes it defer entirely to `Controller:OutdoorAir`'s values; a schedule with a value greater than 0 enables it for those hours.

Demand Controlled Ventilation (Yes/No) is the master DCV switch for this controller. With it set to Yes, the controller recalculates the system's outdoor air requirement using current zone population rather than the design occupancy baked into each zone's sizing — this is also the flag that VAV terminal-level ventilation control (§4.2) reads to decide whether to track current or design occupancy.

System Outdoor Air Method on `Controller:MechanicalVentilation` shares a name with a field of the same name on `Sizing:System` (*(see "Air Loops, Branches, and Nodes", §4.2)*), but the two are unrelated fields on different objects. `Sizing:System`'s version feeds the design-day sizing calculation and offers three choices (`ZoneSum`, `Standard62.1SimplifiedProcedure`, `Standard62.1VentilationRateProcedure`). `Controller:MechanicalVentilation`'s version drives real-time operation and offers a fuller set: `ZoneSum`, `Standard62.1VentilationRateProcedure`, `Standard62.1VentilationRateProcedureWithLimit`, `ProportionalControlBasedOnDesignOccupancy`, `ProportionalControlBasedOnDesignOARate`, `ProportionalControlBasedOnOccupancySchedule`, `IndoorAirQualityProcedure`, `IndoorAirQualityProcedureCombined`, and `IndoorAirQualityProcedureGeneri` Setting one does not set the other — check both if a project's design requires a specific calculation method for both sizing and operation.

Zone Maximum Outdoor Air Fraction caps how much of the system's total supply airflow to any one zone can be outdoor air, regardless of what that zone's ventilation requirement alone would call for.

The per-zone list. `Controller:MechanicalVentilation` carries an extensible list — one row per zone — where each row references a `DesignSpecification:OutdoorAir` (or `DesignSpecification:OutdoorAir:SpaceList`) object and a `DesignSpecification:ZoneAirDistribution` object for that zone. This list, not `Sizing:Zone`'s outdoor-air reference (*(see "Internal Loads, Space Types, and Zone Types", §4)*), is what the system-level ventilation calculation reads at runtime — every zone the controller should account for needs an entry here. EP3 presents this as a table, one row per zone, with checkboxes for selecting multiple zones at once (a **Filter Zones** regex box plus **Select Matching/Deselect Matching** narrow the selection on a system with many zones) and a popup for assigning a `DesignSpecification:OutdoorAir` and `DesignSpecification:ZoneAirDistribution` to every selected zone in one step, rather than entering each zone's row individually.

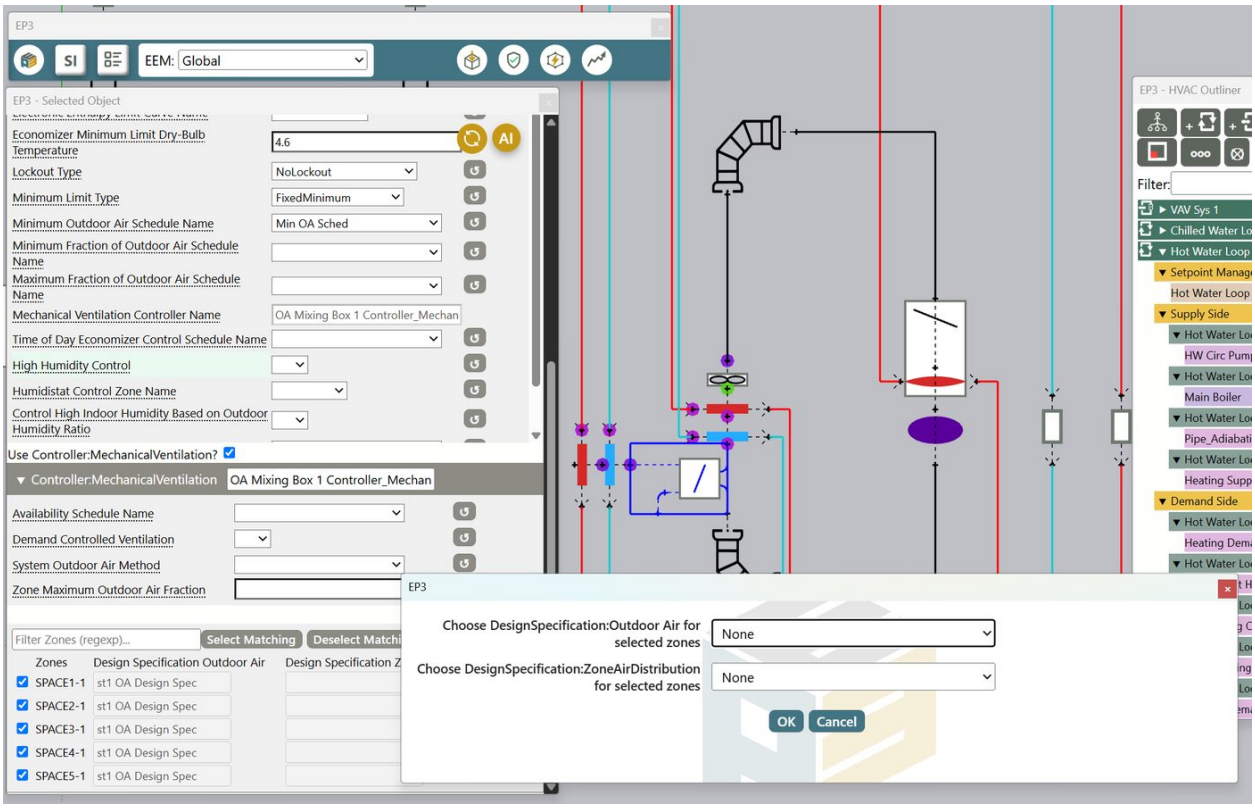


Figure 126: The OutdoorAir:Mixer dialog with Use Controller:MechanicalVentilation? checked, showing Controller:MechanicalVentilation's fields and its per-zone table, with the popup for batch-assigning a DesignSpecification:OutdoorAir and DesignSpecification:ZoneAirDistribution to selected zones.

`DesignSpecification:OutdoorAir` Six calculation methods, set on **Outdoor Air Method**:

Method	Design Flow Rate
`Flow/Person` (default)	Outdoor Air Flow per Person × occupancy
`Flow/Area`	Outdoor Air Flow per Zone Floor Area × zone floor area
`Flow/Zone`	Outdoor Air Flow per Zone, taken directly
`AirChanges/Hour`	Outdoor Air Flow Air Changes per Hour × zone volume, converted to a flow rate
`Sum`	The four fields above, added together
`Maximum`	The larger of the four fields above at any given time

`IndoorAirQualityProcedure`, `ProportionalControlBasedOnDesignOccupancy`, and `ProportionalControlBase` are also valid entries here, corresponding to matching `Controller:MechanicalVentilation` system outdoor air methods; they don't use any of the four flow-rate fields above.

Outdoor Air Flow per Person defaults to 0.00944 m^3/s-person (20 cfm/person). Whether the calculation using this field tracks current or design occupancy depends on which object is reading it: `AirTerminal:SingleDuct:VAV:NoReheat` and `:VAV:Reheat` follow the DCV flag on `Controller:MechanicalVentilation` (§4.2); `AirTerminal:DualDuct:VAV:OutdoorAir` and `ZoneHVAC:IdealLoadsAirSystem` carry an independent occupancy-mode input; `ZoneHVAC:FourPipeFanCoil` always tracks current occupancy regardless of any DCV setting elsewhere.

Under `Standard62.1VentilationRateProcedure` and `Standard62.1VentilationRateProcedureWithLimit`, the EnergyPlus Engineering Reference gives each zone's breathing-zone outdoor airflow as Vbz = Rp×P + Ra×A — the per-person rate (Outdoor Air Flow per Person) times occupant count, plus the per-area rate (Outdoor Air Flow per Zone Floor Area) times zone floor area.

> **Caution — unconfirmed interaction.** Whether this calculation reads both fields unconditionally or only the ones implied by the zone's `DesignSpecification:OutdoorAir` Outdoor Air Method setting isn't confirmed here — the Engineering Reference states the formula without saying whether Outdoor Air Method gates it, and this wasn't checked against EnergyPlus's simulation source code. If a project's ventilation strategy depends on which term is included under VRP, verify the specific configuration with a test run rather than assuming — *(see "Viewing and Interpreting Results")* covers the output variables that would show it.

Outdoor Air Schedule Name multiplies the calculated flow rate by a 0–1 schedule value at every timestep it's read — every object above that consumes this `DesignSpecification:OutdoorAir` object reads this schedule the same way, including `Controller:MechanicalVentilation` regardless of which System Outdoor Air Method is chosen. It's ignored during sizing.

Proportional Control Minimum Outdoor Air Flow Rate Schedule Name applies only when `Controller:MechanicalVentilation`'s System Outdoor Air Method is `ProportionalControlBasedOnDesignOARate`.

`DesignSpecification:ZoneAirDistribution` Accounts for how effectively a zone's supply air distribution delivers outdoor air to the breathing zone, per ASHRAE 62.1's ventilation rate pro-

cedure math. **Zone Air Distribution Effectiveness in Cooling Mode** and **...in Heating Mode** each default to 1.0 (fully effective); **Zone Air Distribution Effectiveness Schedule Name** can replace both with a time-varying value. **Zone Secondary Recirculation Fraction** and **Minimum Zone Ventilation Efficiency** feed the multi-zone system-level ventilation efficiency calculation that `Standard62.1VentilationRateProcedure` performs across all the zones a system serves.

`DesignSpecification:OutdoorAir:SpaceList` Where a zone contains multiple Spaces with different ventilation requirements, this object lists each Space by name alongside a `DesignSpecification:OutdoorAir` reference, and is entered wherever a zone-level outdoor air field (on `Sizing:Zone`, a VAV terminal, or `Controller:MechanicalVentilation`'s per-zone list) would otherwise take a single `DesignSpecification:OutdoorAir` object directly.

4.2 Terminal-level: per-terminal outdoor air control

`AirTerminal:SingleDuct:VAV:Reheat`, `:VAV:NoReheat`, `:ConstantVolume:NoReheat`, `:ConstantVolume:Cooled`
`AirTerminal:DualDuct:VAV`, and `:DualDuct:VAV:OutdoorAir` each carry a **Design Specification Outdoor Air Object Name** field, referencing a `DesignSpecification:OutdoorAir` object directly on the terminal rather than through `Controller:MechanicalVentilation`'s per-zone list.

This reference drives a different kind of calculation than the same object type referenced from `Sizing:Zone`. `Sizing:Zone`'s reference feeds a one-time design-day calculation used for autosizing. A terminal-level reference is read at every simulation timestep: the terminal first sizes its airflow to the zone's thermal load as usual, then compares that against the outdoor air requirement computed from the referenced object, and opens its damper further whenever the load-driven flow alone wouldn't deliver enough outdoor air. Leaving this field blank on a terminal that supports it means that terminal's damper responds to zone load only, with no minimum-outdoor-air enforcement beyond that.

How this interacts with `Controller:MechanicalVentilation` varies by terminal type:

- On `AirTerminal:SingleDuct:VAV:Reheat` and `:VAV:NoReheat`, whether the per-person portion of the calculation uses current occupancy or design occupancy is governed by `Controller:MechanicalVentilation`'s **Demand Controlled Ventilation** flag (§4.1) — these two terminal types carry no occupancy-mode field.
- `AirTerminal:ConstantVolume:CooledBeam` falls back to whatever `DesignSpecification:OutdoorAir` object is referenced on `Sizing:Zone` if that field is left blank, rather than disabling outdoor air control entirely.
- `AirTerminal:DualDuct:VAV:OutdoorAir` carries an occupancy-mode field, independent of `Controller:MechanicalVentilation`'s DCV flag, and this field is required rather than optional: supplying exactly the zone's outdoor air requirement is this terminal type's purpose.

  Caution — unconfirmed interaction. How `AirTerminal:SingleDuct:ConstantVolume:NoReheat` and `AirTerminal:DualDuct:VAV` behave when both this field and a system-level `Controller:MechanicalVentilation` are active on the same system isn't confirmed here. Before relying on both together on a real project, verify actual delivered outdoor air airflow with a test run — *(see "Viewing and Interpreting Results")* covers the output variables that would show it.

5. Dedicated Outdoor Air Systems (DOAS)

5.1 The common pattern: a single 100%-outdoor-air `AirLoopHVAC`

The great majority of DOAS systems in EP3 are `AirLoopHVAC` configured to deliver only outdoor air. Answering **Yes** to **Dedicated Outdoor Air System?** in the Create Air Loop dialog sets `Sizing:System`'s Type of Load to Size On to `VentilationRequirement` and both 100% Outdoor Air in Cooling and 100% Outdoor Air in Heating to Yes (*(see "Air Loops, Branches, and Nodes", §4.2)*). No distinct EnergyPlus object type is involved, just sized and controlled to pass 100% outdoor air rather than a mixed-air fraction.

Delivery to each zone then follows one of two paths: a standalone airloop feeding ordinary terminal units when the DOAS is the only air-side system a zone has, or `AirTerminal:SingleDuct:Mixer` (*(see "Air Loops, Branches, and Nodes", §5.1)*) combining the DOAS airstream with a locally-conditioned airstream from a zone-level unit. §4 covers what decides how much air moves through either path at a given moment.

`Sizing:Zone`'s Account for Dedicated Outdoor Air System field and its associated control strategy and design supply air temperature fields (`NeutralSupplyAir`, `NeutralDehumidifiedSupplyAir`, `ColdSupplyAir`) govern how a DOAS's contribution is credited against a zone's equipment sizing — *(see "Zone-Level HVAC Equipment and Zone Sizing", §2.4)* covers these in full.

5.2 `AirLoopHVAC:DedicatedOutdoorAirSystem`

EnergyPlus also defines a distinct `AirLoopHVAC:DedicatedOutdoorAirSystem` object, which ties one shared `AirLoopHVAC:OutdoorAirSystem` to an `AirLoopHVAC:Mixer`/`AirLoopHVAC:Splitter` pair serving multiple separate `AirLoopHVAC`s at once — a single central dedicated OA source feeding several independent airloops, rather than one airloop delivering 100% outdoor air by itself. This object isn't yet supported in EP3. **If a project needs it, request it from the EP3 team.**

---

6. Outdoor Air at Zone-Level Equipment

Most `ZoneHVAC` composite objects can draw outdoor air directly, independent of a central airloop's mixed-air path — *(see "Zone-Level HVAC Equipment and Zone Sizing")* covers this equipment generally and how each type is assigned to a zone. A few object types expose more outdoor-air-specific control than a simple minimum/maximum flow rate — `ZoneHVAC:EnergyRecoveryVentilator`, paired with an energy recovery heat exchanger (§7.1), and `ZoneHVAC:UnitVentilator`, with economizer-style outdoor air control modes, among them. Consult ZoneHVAC:EnergyRecoveryVentilator and ZoneHVAC:UnitVentilator for their field sets.

Whether a `ZoneHVAC` object's outdoor air is actually delivered at any given moment also depends on its fan's operating mode — *(see "Fans and Fan Power", §2.2)* covers the Supply Air Fan Operating Mode Schedule Name mechanics, and interactions with outdoor air supply for this object types.

---

7. Energy Recovery

An air-to-air heat exchanger transfers energy between the outdoor air intake and the relief/exhaust airstream, placed on the intake/exhaust branch pair of an `OutdoorAir:Mixer` (§2.1) or as the heat exchanger sub-component inside `ZoneHVAC:EnergyRecoveryVentilator` (§6).

7.1 `HeatExchanger:AirToAir:SensibleAndLatent`

Sized by **Nominal Supply Air Flow Rate** (autosizable), with performance entered directly as effectiveness at each of four combinations of mode and airflow fraction: **Sensible Effectiveness at 100% Heating Air Flow**, **Latent Effectiveness at 100% Heating Air Flow**, **Sensible Effectiveness at 100% Cooling Air Flow**, and **Latent Effectiveness at 100% Cooling Air Flow**, each a 0–1 fraction. **Heat Exchanger Type** (`Plate` or `Rotary`) affects the frost-control mechanics below but not the effectiveness calculation itself. **Supply Air Outlet Temperature Control** (Yes/No) lets the device modulate its effectiveness to hold a downstream setpoint rather than running at its full rated effectiveness continuously.

Frost Control Type — `None`, `ExhaustAirRecirculation`, `ExhaustOnly`, or `MinimumExhaustTemperature` — governs how the device protects itself against frost formation at low outdoor temperatures. `ExhaustAirRecirculation` and `ExhaustOnly` use **Threshold Temperature** (default 1.7°C) as the supply air inlet temperature below which frost control activates, **Initial Defrost Time Fraction** (default 0.083) as the fraction of operating time spent in frost control right at the threshold, and **Rate of Defrost Time Fraction Increase** (default 0.012 per K) as how much further that fraction grows for every degree below the threshold. `MinimumExhaustTemperature` instead reads Threshold Temperature as an exhaust air outlet temperature limit.

Economizer Lockout (default Yes) disables the heat exchanger whenever `Controller:OutdoorAir`'s economizer or high humidity control is active on the same airloop — recovering energy from an airstream an economizer is deliberately using for free cooling would work against the economizer's purpose.

Worked example — entering manufacturer effectiveness data. A manufacturer's submittal lists an energy recovery wheel's performance at two design conditions: at the winter (heating) design condition, 100% airflow, sensible effectiveness of 72% and latent effectiveness of 61%; at the summer (cooling) design condition, 100% airflow, sensible effectiveness of 75% and latent effectiveness of 63%. These map directly onto the four effectiveness fields, entered as fractions:

Field	Value
Sensible Effectiveness at 100% Heating Air Flow	0.72
Latent Effectiveness at 100% Heating Air Flow	0.61
Sensible Effectiveness at 100% Cooling Air Flow	0.75
Latent Effectiveness at 100% Cooling Air Flow	0.63

No unit conversion is needed here — the source data and the field are both dimensionless fractions of the same quantity. The conversion work in a case like this is entirely in identifying which of the manufacturer's four reported numbers (heating vs. cooling, sensible vs. latent) belongs in which of the four fields.

7.2 `HeatExchanger:AirToAir:FlatPlate`

A simpler alternative, sized from a single nominal operating condition rather than four effectiveness values directly. **Flow Arrangement Type** (`CounterFlow`, `CrossFlowBothUnmixed`, or `ParallelFlow`) and **Ratio of Supply to Secondary hA Values** (the ratio of the supply side's $h \cdot A$ — heat transfer coefficient times area — to the exhaust side's) describe the exchanger's physical construction. **Nominal Supply Air Flow Rate**, **Nominal Secondary Air Flow Rate** (both autosizable), and the paired **Nominal Supply/Secondary Air Inlet/Outlet Temperature** fields describe one reference operating point EnergyPlus uses to back out the exchanger's UA and predict its performance at other conditions during the simulation, rather than accepting rated effectiveness values directly the way `HeatExchanger:AirToAir:SensibleAndLatent` does. **Economizer Lockout** works the same as its `SensibleAndLatent` counterpart above; this object has no frost control fields.

`HeatExchanger:Desiccant:BalancedFlow`, EnergyPlus's desiccant wheel object, is available in EP3's schema but isn't detailed here — HeatExchanger:Desiccant:BalancedFlow.

---

8. Summary

This chapter covered how outdoor air enters an `AirLoopHVAC` (`OutdoorAir:Mixer`, `AirLoopHVAC:OutdoorAirSyste` how much of it enters at a given moment (`Controller:OutdoorAir`'s economizer, `Controller:MechanicalVentila` system-level demand-controlled ventilation, and per-terminal outdoor air control on individual VAV terminals), how a dedicated outdoor air system is built in EP3 versus the distinct EnergyPlus object of a similar name, zone-level equipment that manages ventilation air directly, and the two air-to-air heat exchanger objects that recover energy between the intake and exhaust airstreams.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target chapters exist; confirm how `Controller:MechanicalVentilation` and terminal-level outdoor air control interact on `AirTerminal:SingleDuct:ConstantVolume:NoReheat` and `AirTerminal:DualDuct:VAV` specifically (§4.2) and update that section once verified; confirm `Controller:OutdoorAir`'s Minimum Limit Type enumeration and effect against the current EnergyPlus I/O Reference before removing the EXTLINK placeholder in §3.1; write up `Controller:OutdoorAir`'s High Humidity Control field and related fields in §3 once their operation is confirmed in EnergyPlus directly; confirm whether `Controller:MechanicalVentilation`'s Standard 62.1 Ventilation Rate Procedure calculation (§4.1) reads both Outdoor Air Flow per Person and Outdoor Air Flow per Zone Floor Area unconditionally or only per the zone's Outdoor Air Method setting, by reading EnergyPlus source code or testing directly, and update that section once verified.

22. Infiltration, Natural Ventilation and Airflow Networks

1. Overview

This chapter covers the outdoor-air paths driven by wind, buoyancy (stack effect), and pressure differences across the envelope rather than by a fan under active control: infiltration through unintentional envelope leakage, natural ventilation through intentionally opened windows or vents, and `AirflowNetwork`'s detailed multizone pressure model, which can represent either of the first two in more physical detail. Mechanical, fan-driven ventilation — economizers, demand-controlled ventilation, dedicated outdoor air systems, energy recovery — is covered in *(see "Outdoor Air, Ventilation, and Energy Recovery")*. A zone can carry both mechanical and natural/infiltration airflow paths at once; they're modeled independently and EnergyPlus sums their effects on the zone heat balance.

Two objects that sit right at the boundary between the two chapters are covered here rather than there: `Fan:ZoneExhaust` (§2.4) is HVAC equipment, but the mechanism EP3 uses to keep its exhausted air from leaving a zone under-pressurized is built entirely from this chapter's infiltration objects; `Construction:AirBoundary` and `AvailabilityManager:HybridVentilation` (§5) are each covered in full in their respective chapters, but both change behavior when `AirflowNetwork` is active — that interaction is explained here.

---

2. Infiltration

2.1 Three paths to model infiltration

EP3 offers three ways to put infiltration into a model, at increasing levels of manual control:

- **Standard — a `SpaceType`'s `ZoneInfiltration:DesignFlowRate` sub-object.** This is what the "Create New Space Type and Loads" dialog creates automatically *(see "Internal Loads, Space Types, and Zone Types", §2)*, and it's the object EP3's built-in `SpaceType` templates ship with. Its full field table lives in *(see "Internal Loads, Space Types, and Zone Types", §3.5)* — this is the path almost every project should use, and the rest of this section exists for the cases where it isn't enough by itself.
- **Legacy — a loose `ZoneInfiltration:DesignFlowRate`/`:EffectiveLeakageArea`/`:FlowCoefficient` assigned directly.** The same object type as the standard path, built and assigned the same generic way any loose object is *(see "The EP3 Interface and Model Settings")*, but pointed at a Zone, Space, ZoneList, or SpaceList directly through the **Zone or ZoneList or Space or SpaceList Name** field instead of inheriting from a `SpaceType`. This is EnergyPlus's native approach, useful for a model with no `SpaceType` structure at all, or for one Space/Zone that needs an infiltration rate independent of its `SpaceType`'s default.
- **Advanced — `ComponentInfiltration`** (§2.3), for the specific case of a building whose exterior construction assemblies vary in air leakage within the same Space.

Only one of these should apply to a given Space at a time — §2.3 covers what happens when more than one does.

2.2 Choosing an infiltration rate — worked example

`ZoneInfiltration:DesignFlowRate`'s **Design Flow Rate** (or its per-area/per-exterior-area/ACH equivalents) is scaled at every timestep by up to four coefficients — **Constant Term Coefficient**, **Temperature Term Coefficient**, **Velocity Term Coefficient**, and **Velocity Squared Term Coefficient** — applied to the indoor-outdoor temperature difference and wind speed. The "Create New Space Type and Loads" dialog's default (Constant Term Coefficient = 1.0, the other three = 0.0) makes infiltration a flat rate, modulated only by its Schedule — the simplest case, and the right default for most projects. Wind-driven modulation (Constant Term Coefficient = 0.0, Velocity Term Coefficient > 0.0) is a deliberate alternative for when a tested leakage rate should scale with actual wind speed rather than hold constant.

Worked example — converting a tested envelope leakage rate to a wind-driven design flow rate, following the method ASHRAE 90.1 Appendix G §G3.1.1.4 uses for baseline/proposed model infiltration — the same method EP3's baseline generator (`ASHRAE901PRM2019/envelope.rb`) implements:

Suppose a whole-building air-leakage test (per ASTM E779 or E1827) measures **0.35 cfm/ft² at 75 Pa** — a moderately tight envelope, below the 2019 code-minimum default of 1.0 cfm/ft². The test pressure (75 Pa) is far above the pressures that drive infiltration under normal wind and stack conditions, so the tested rate needs scaling down before it's a usable **Flow Rate per Exterior Surface Area** value:

`Flow Rate per Exterior Surface Area [m³/(s·m²)] = tested rate [cfm/ft²] × 0.00508 × 0.112`

The 0.00508 factor converts cfm/ft² to m³/(s · m²) (1 cfm/ft² = 0.3048 m / 60 s ~ 0.00508 m³/(s · m²)); the 0.112 factor is the normalization ASHRAE 90.1 Appendix G §G3.1.1.4 applies to a 75 Pa test result to reach the reference infiltration rate EnergyPlus's wind-driven equation expects.

`Flow Rate per Exterior Surface Area = 0.35 × 0.00508 × 0.112 ~ 0.000199 m³/(s·m²)`

Check, in IP terms: 0.35 cfm/ft² × 0.112 = 0.0392 cfm/ft² — the same result, since the unit-conversion factor cancels when checking in the original units. Entering 0.000199 m³/(s · m²) (or its 0.0392 cfm/ft² equivalent) as **Flow Rate per Exterior Surface Area**, together with **Constant Term Coefficient = 0.0**, **Temperature Term Coefficient = 0.0**, **Velocity Term Coefficient = 0.224**, and **Velocity Squared Term Coefficient = 0.0** (the coefficient set §G3.1.1.4 pairs with this conversion), reproduces the tested leakage rate's contribution to infiltration scaled by wind speed rather than held flat.

Sanity check at a specific wind speed: at a 4 m/s wind speed, this produces an infiltration flow density of 0.000199 × 0.224 × 4 ~ 0.000178 m³/(s · m²) — roughly 90% of the base rate at a modest wind speed, rising above it as wind speed increases further, and falling toward zero in calm conditions.

This conversion is specific to the wind-driven coefficient convention above; it isn't the right factor to apply if you intend to enter a flat, schedule-only rate (Constant Term Coefficient = 1.0) instead — translating a tested rate into a flat convention is a different calculation, and isn't attempted here.

Sources: ASHRAE 90.1-2019 Appendix G §G3.1.1.4; `ASHRAE901PRM2019/envelope.rb` (`CFM_FT2_TO_M3_SM2`, `G3_1_1_4_FACTOR`, and `assign_infiltration`), which document both the conversion factor and the paired coefficient set.

2.3 `ComponentInfiltration` — the advanced aggregation path

`ComponentInfiltration` exists for a narrower case than §2.1's two paths: a building where exterior construction assemblies within the same Space have meaningfully different air leakage, most often because the building was constructed or renovated in phases and different-generation wall assemblies now share a Space. Neither a `SpaceType`'s single infiltration rate nor a legacy loose object pointed at the whole Space can represent that variation — both apply one rate uniformly across the Space regardless of which wall it's leaking through.

Setup. Every `Construction`, regardless of type, carries an **Infiltration Rate** field, entered per unit area *(see "Constructions and Materials", §3)* — but entering a value there does nothing by itself. A `ComponentInfiltration` object is what reads those per-construction rates and turns them into a real `ZoneInfiltration:DesignFlowRate`: assigned to a Space or SpaceList through the **Space or SpaceList Name** field, it walks every face and window in each assigned Space, multiplies each one's area by its construction's Infiltration Rate, and sums the result into one `ZoneInfiltration:DesignFlowRate` per Space, using **Flow/Zone** as the calculation method with the aggregated value as **Design Flow Rate**. The **Schedule Name** field (blank defaults to always 1.0) and the **Constant/Temperature/Velocity/Velocity Squared Term Coefficient** fields (Constant defaults to 1.0, the rest to 0.0 — the same flat-rate default as §2.1's standard path) work exactly as they do on an ordinary `ZoneInfiltration:DesignFlowRate`.

> *Figure placeholder — a Construction dialog showing the "Infiltration Rate" field for two different wall constructions with visibly different values, alongside a `ComponentInfiltration` object's dialog showing the Space or SpaceList Name it's assigned to.*

Pitfalls. Two mistakes are specific to this path, and EP3 doesn't currently catch either:

- **Double-counting.** A Space keeps whatever `ZoneInfiltration:DesignFlowRate` it already had — from its `SpaceType` or from a legacy loose object — even after a `ComponentInfiltration` is added on top of it. Both produce real infiltration for the same Space; nothing removes the first automatically. Remove or unassign the Space's `SpaceType`-inherited or legacy infiltration before relying on `ComponentInfiltration` for it.
- **Silent coverage gaps.** Adopting this approach for a project doesn't extend to every Space automatically — a Space with no `ComponentInfiltration` assigned gets zero modeled infiltration, with nothing to flag it.

No model-validation check today catches either case — *(see "Model Troubleshooting and Quality Assurance")* is the intended home for one, but as of this writing it doesn't exist; treat both pitfalls above as something to check by hand.

2.4 `Fan:ZoneExhaust` make-up air

An exhaust fan removes air from a zone; EnergyPlus doesn't supply anything to replace it unless something else in the model is set up to. If the zone is served by an `AirLoopHVAC`, that loop's outdoor air system can draw in extra outdoor air and reduce return flow to cover the exhaust — but only while that airloop's fan is running. A zone with only `ZoneHVAC` equipment — most of what *(see "Zone-Level HVAC Equipment and Zone Sizing")* covers — has no automatic make-up air source at all, and EnergyPlus doesn't warn about the resulting imbalance.

The setup choice. Placing a `Fan:ZoneExhaust` from the Components dialog triggers a one-

time prompt, skipped once it's been answered and skipped entirely if the model already has `AirflowNetwork` active (§4 covers why — AFN computes zone-level airflow directly, which supersedes what this mechanism does):

- **"Automatically balance with infiltration (recommended)"** adds a single, model-wide `ZoneAirMassFlowConservation` object (or updates it if one already exists), configured with **Adjust Zone Mixing and Return For Air Mass Flow Balance = AdjustReturnOnly**, **Infiltration Balancing Method = AddInfiltrationFlow**, and **Infiltration Balancing Zones = AllZones**. `ZoneAirMassFlowConservation` only affects a zone that already has a `ZoneMixing` or `ZoneInfiltration` object, and most exhaust-fan zones have neither — so EP3 also emits a synthetic, zero-flow `ZoneInfiltration:DesignFlowRate` at export time, assigned to every zone in the model. This synthetic object is raw exported IDF text only, generated from within `ZoneAirMassFlowConservation` itself — it never appears as a real object in EP3's model tree or object list, and won't reappear as a separate object on import. Its only purpose is giving the balancing calculation something to act on in a zone that otherwise has nothing.
- **"I'll model this manually"** leaves the choice to you: either a `ZoneVentilation:DesignFlowRate` object (§3) with **Ventilation Type = Exhaust** or **Balanced**, sized to the fan's flow rate for outdoor-air-balanced, per-fan control, or a real `ZoneMixing` object that deliberately draws air from one named adjacent zone instead of from outdoors.

Scope caveat. `ZoneAirMassFlowConservation` is a single, model-wide object — there is no per-zone or per-fan variation. Choosing "manual" for one exhaust fan doesn't turn off an already-active automatic backstop for every other exhaust fan in the model, and the automatic setup's `AdjustReturnOnly` choice is deliberately conservative — it leaves any real `ZoneMixing` objects elsewhere in the model untouched rather than overriding them.

Finding and reviewing it later. `ZoneAirMassFlowConservation` doesn't get a dedicated dialog — like `HeatBalanceAlgorithm` and other model-wide simulation settings, it's edited through the generic object list under the **Simulation Parameters** group. Its field names and labels don't explain what it does, so look there directly rather than expecting a dedicated dialog to surface it.

Figure placeholder — the one-time "Fan:ZoneExhaust Make-Up Air" picker dialog, showing both option labels and their explanatory notes.

Model Validation. A model-wide (not per-zone) check flags this under the Validation dialog's HVAC category: an error if any `Fan:ZoneExhaust` exists with no `ZoneAirMassFlowConservation` at all, a warning if one exists but isn't configured for the infiltration backstop (relying on `ZoneMixing`/`None` instead, which EP3 has no way to verify covers every exhaust-fan zone). Both offer a one-click fix that applies the same recommended configuration as the creation-time picker.

Figure placeholder — the Validation dialog's "Exhaust Fan Make-Up Air" section, showing the warning state (relying on ZoneMixing) with its "Switch to Automatic Infiltration Backstop" fix button.

---

3. Natural ventilation (simple method)

`ZoneVentilation:DesignFlowRate` and `ZoneVentilation:WindandStackOpenArea` model intentional, typically controllable outdoor-air exchange — an operable window opened on a schedule, or

a vent sized to a physical opening — without requiring `AirflowNetwork`. Both are loose objects, built and assigned the same generic way as any other loose object *(see "The EP3 Interface and Model Settings")*; neither is managed by a `SpaceType` template the way infiltration is.

The distinction from infiltration (§2) comes down to intent: infiltration is unintentional leakage through the envelope assembly, always present whenever there's a pressure difference; `ZoneVentilation:*` represents an opening a person or a control system chooses to open, typically scheduled and often shut off outside a favorable temperature range.

3.1 `ZoneVentilation:DesignFlowRate`

Structurally close to `ZoneInfiltration:DesignFlowRate` (§2.1) — the same **Zone or ZoneList or Space or SpaceList Name** assignment field, the same **Design Flow Rate Calculation Method** pattern (`Flow/Zone`, `Flow/Area`, `Flow/Person`, or `AirChanges/Hour`), and the same four term coefficients — with two additions:

Field	Purpose
Ventilation Type	`Natural` (the default — wind/stack driven, no fan), or `Exhaust`/`Intake`/`Balanced` for a small fan-assisted flow, sized via the two fields below. This is the object §2.4's "manual" make-up-air option points to for a per-fan alternative to the automatic infiltration backstop.
Fan Pressure Rise / Fan Total Efficiency	Only meaningful for a non-`Natural` Ventilation Type — sizes the fan power associated with the flow, the same relationship *(see "Fans and Fan Power", §5)* covers for real fan objects.
Minimum/Maximum Indoor Temperature, Delta Temperature, Minimum/Maximum Outdoor Temperature, Maximum Wind Speed (each with an optional schedule variant)	Cutoffs that shut ventilation off outside the conditions it's meant to represent — for example, a Delta Temperature that stops crediting "ventilation" once the zone is no longer warmer than outdoors.

ZoneVentilation:DesignFlowRate documents the full field set, including every schedule-variant cutoff field.

3.2 `ZoneVentilation:WindandStackOpenArea`

Models flow through a specific, sized opening — an operable window's clear opening area — rather than an arbitrary design flow rate, combining EnergyPlus's wind and stack-effect correlations. Assigned directly to one Zone or Space (no ZoneList/SpaceList option, unlike §3.1).

Field	Purpose
Opening Area / Opening Area Fraction Schedule Name	The physical opening's area, and the fraction of it open over time (e.g., a window cracked partway).

Field	Purpose
Opening Effectiveness / Effective Angle	The wind-driven flow coefficient (Cw) — `Autocalculate` derives it from the opening's angle relative to prevailing wind, interpolating between 0.55 (facing the wind) and 0.3 (180° from it).
Height Difference	The stack-effect driving height (ΔH) — the vertical distance between the opening's midpoint and the zone's neutral pressure level.
Discharge Coefficient for Opening	The stack-effect flow coefficient (Cd) — `Autocalculate` derives it from the indoor-outdoor temperature difference.
Minimum/Maximum Indoor Temperature, Delta Temperature, Minimum/Maximum Outdoor Temperature, Maximum Wind Speed (each with an optional schedule variant)	The same cutoff pattern as §3.1.

ZoneVentilation:WindandStackOpenArea documents the full field set and the underlying wind/stack equations.

---

4. AirflowNetwork

4.1 General mechanics

`AirflowNetwork` solves a pressure-driven mass balance across the zones and exterior nodes it's told about, rather than assuming a fixed design flow rate the way §2 and §3's objects do. Three driving forces feed the calculation:

- **Wind pressure**, from `P = Pref + 0.5 × rho × V² × Cp`, where Cp (the wind pressure coefficient, typically −0.8 to +0.8) depends on a surface's orientation relative to the wind.
- **Buoyancy (stack effect)**, from indoor-outdoor temperature (density) differences.
- **HVAC-driven pressure** — not modeled in EP3 (§4.4).

Two ways to get Cp values:

Approach	How it works	Trade-off
`SurfaceAverageCalculation`	EnergyPlus derives Cp automatically from empirical correlations based on overall building shape. EP3's default.	Minimal setup; adequate for simple, roughly rectangular massing; less accurate for complex or shielded geometry.

Approach	How it works	Trade-off
`Input`	Cp is entered by hand per surface, per wind direction (commonly 36 directions), via `AirflowNetwork:MultiZone:WindPressureCoefficientArray` and requires `AirflowNetwork:MultiZone:ExternalNode` objects to represent outdoor conditions.	Full control, and the only way to bring in wind-tunnel or CFD-derived data; `AirflowNetwork:MultiZone:WindPressureCoefficientValues` good as the data behind it.

Both are set on `AirflowNetwork:SimulationControl`'s **Wind Pressure Coefficient Type** field. When using `SurfaceAverageCalculation`, EP3 auto-populates **Azimuth Angle of Long Axis of Building** and **Ratio of Building Width Along Short Axis to Width Along Long Axis** from the modeled floor plate's geometry whenever those fields are left blank — they don't need to be measured or entered by hand for a model EP3 already has geometry for.

Source: general AirflowNetwork mechanics per the EnergyPlus Engineering Reference; EP3's field behavior per `airflownetwork.rb`.

4.2 How EP3 presents it

Turning it on. `AirflowNetwork` participation is a Zone-level choice, not a per-surface one. Adding an `AirflowNetwork:MultiZone:Zone` sub-object to a Zone — directly, or cascaded from its `ZoneType` the same way `Sizing:Zone` is (*(see "Zone-Level HVAC Equipment and Zone Sizing", §4.1)* covers that cascading pattern) — is what makes a zone part of the network, alongside an `AirflowNetwork:SimulationControl` object existing somewhere in the model.

Surface participation is automatic, and lives on the Construction, not the surface. Once a Zone participates, every exterior surface belonging to it (excluding ground-contact and adiabatic surfaces) picks up an `AirflowNetwork:MultiZone:Surface` sub-component automatically — nothing needs to be placed by hand, surface by surface. That sub-component is attached to the surface's **Construction**, not the surface itself: assign a leakage or opening component once, on a construction, and every surface built from that construction — across the whole model — carries it automatically. Overriding a default means changing the **Leakage Component Name** field on the construction's `AirflowNetwork:MultiZone:Surface` sub-object, not hunting through individual surfaces.

Choosing a leakage/opening component. The **Leakage Component Name** field points at one of:

Component	Represents
`AirflowNetwork:MultiZone:Surface:EffectiveLeakageArea`	One leakage path/crack/unintended air leakage, entered as a percentage of surface area — recalculated per surface at export, so one component correctly serves surfaces of different sizes that share the same construction. This is the object to use for the common case (§4.3's worked example uses it).
`AirflowNetwork:MultiZone:Surface:EffectiveLeakageArea`	A fixed leakage area in m², applied identically regardless of the surface's area — appropriate for one specific, individually-measured leakage path (a known damper or vent), not for ordinary wall leakage shared across differently-sized surfaces.
`AirflowNetwork:MultiZone:Surface:Crack`	A flow-coefficient/exponent pair, for a leakage path better characterized directly by a crack-flow test than by an area.
`AirflowNetwork:MultiZone:Component:SimpleOpening` / `:DetailedOpening`	An openable opening (a window or door) rather than leakage — `SimpleOpening` as an orifice-flow approximation, `DetailedOpening` for a more detailed two-way flow model through a large opening.

`AirflowNetwork:MultiZone:ExternalNode` and the `WindPressureCoefficientArray/Values` pair are only needed with the `Input` Cp method (§4.1).

> *Figure placeholder — a Construction dialog's AirflowNetwork tab, showing the Leakage Component Name field pointing at an EffectiveLeakagePercentArea object, to illustrate that this setting lives on the construction rather than on any individual surface.*

4.3 Worked example — Effective Leakage Area

`AirflowNetwork:MultiZone:Surface:EffectiveLeakageArea`'s **Effective Leakage Area** field is referenced to EnergyPlus's default conditions — **Reference Pressure Difference = 4.0 Pa**, **Discharge Coefficient = 1.0** — a different convention from §2.2's 75 Pa infiltration test pressure, so the two worked examples don't share a conversion factor.

Suppose a blower-door test reports **250 L/s at 50 Pa** on a test section with **40 m² of exterior wall/window area**, and no multi-point data is available, so a typical whole-building flow exponent of **n = 0.65** is assumed (a multi-point test would give a measured exponent instead of this assumption).

Step 1 — scale the tested flow down to EnergyPlus's 4 Pa reference, using the power-law relationship between flow and pressure difference:

`Q(ΔP) = Q(ΔP_test) × (ΔP / ΔP_test)<sup>n</sup>`

`Q(4 Pa) = 0.250 m³/s × (4 / 50)^0.65 = 0.250 × 0.194 ~ 0.0485 m³/s`

Step 2 — convert that flow to an effective leakage area, using the standard ELA relationship (ASHRAE Fundamentals, Chapter 16, "Ventilation and Infiltration"):

`ELA = Q(ΔP_ref) / (Cd × √(2 × ΔP_ref / rho))`

With Cd = 1.0 (matching the object's default) and rho ~ 1.2 kg/m³ (standard air density):

`√(2 × 4 / 1.2) = √6.67 ~ 2.58 m/s`

`ELA = 0.0485 / 2.58 ~ 0.0188 m² (~ 188 cm²)`

Step 3 — express it as a percentage of the tested surface's area, since EP3's per-construction auto-attachment (§4.2) means the practical object to enter this on is `EffectiveLeakagePercentArea`, not the fixed-area object directly:

`Effective Leakage As Percent of Surface Area = 0.0188 m² ÷ 40 m² × 100 ~ 0.047%`

Entered on the wall construction's `EffectiveLeakagePercentArea` component — alongside **Discharge Coefficient = 1.0**, **Reference Pressure Difference = 4.0 Pa**, and **Air Mass Flow Exponent = 0.65** (matching the exponent assumed in Step 1; entering a different value here would make the object's flow calculation inconsistent with the leakage rate just derived) — this reproduces the tested leakage rate on every surface built from that construction, scaled to each surface's area.

If a test instead covers only a single, specific opening (a known operable vent rather than distributed wall leakage), skip Step 3 and enter the Step 2 result directly on a plain `EffectiveLeakageArea` object instead — its fixed-area assumption is the correct one for a single measured opening, rather than an approximation to work around.

Source: power-law and effective leakage area relationships per ASHRAE Fundamentals, Chapter 16 ("Ventilation and Infiltration"); object field defaults per `json_key_array26p1.rb`.

4.4 Envelope-only

EP3's `AirflowNetwork` implementation covers building-envelope airflow only. No `AirflowNetwork:Distribution*` object, and no HVAC-side AFN component, exists in any EnergyPlus version's class or field-schema files EP3 uses. `AirflowNetwork:SimulationControl`'s **AirflowNetwork Control** field reflects this directly: it's fixed to `MultizoneWithoutDistribution` and isn't editable. A duct leakage or fan-interaction study using `AirflowNetwork`'s distribution-system modeling isn't possible in EP3 today.

---

5. Coordinating natural and mechanical ventilation

- **`Construction:AirBoundary`**, covered fully in *(see "Constructions and Materials", §7)*, models a zone-to-zone adjacency with no thermal resistance and an optional simple-mixing airflow between the two zones. When `AirflowNetwork` is active in the model, that simple-mixing behavior is ignored in favor of the network's airflow calculation across that same boundary — `Construction:AirBoundary`'s mixing fields stay in the model but stop doing anything once AFN takes over.
- **`AvailabilityManager:HybridVentilation`**, whose field set is covered in full in *(see "Controllers and Availability Managers", §4.5)*, coordinates natural ventilation with mechanical

system operation — switching between them on a schedule, subject to weather-based cutouts (rain, wind speed, temperature/enthalpy/dewpoint limits). Its natural-ventilation side is built around `AirflowNetwork` openings specifically (or, in a simpler model, `ZoneVentilation` objects from §3) — when it switches natural ventilation "on," what happens is governed by whichever of this chapter's mechanisms the model is using, not by anything on the availability manager itself.

- **`ZoneMixing`/`ZoneCrossMixing`** move air directly from one zone to another at a fixed or scheduled rate, independent of any pressure calculation. They're valid loose objects in EP3 but see limited use in practice, since `Construction:AirBoundary` (above) covers the same zone-to-zone-mixing need more directly for an adjacency that's already being modeled as a boundary rather than a wall. Reach for `ZoneMixing`/`ZoneCrossMixing` directly when the two zones are separated by a real wall/door rather than an air boundary, or when the transfer needs a schedule independent of the boundary's.

---

6. Where to Look Next

- *(see "Internal Loads, Space Types, and Zone Types", §3.5)* — the full `ZoneInfiltration:DesignFlowRate` field table.
- *(see "Constructions and Materials", §3, §7)* — the per-construction Infiltration Rate field `ComponentInfiltration` reads, and `Construction:AirBoundary`.
- *(see "Zone-Level HVAC Equipment and Zone Sizing")* — `Fan:ZoneExhaust`'s place among other zone-level equipment, and the `ZoneType` cascading-default pattern `AirflowNetwork:MultiZone:Zone` reuses.
- *(see "Outdoor Air, Ventilation, and Energy Recovery")* — mechanical, fan-driven ventilation and energy recovery.
- *(see "Controllers and Availability Managers", §4.5)* — `AvailabilityManager:HybridVentilation`'s full field set.
- *(see "Fans and Fan Power", §5)* — the fan power relationship behind `ZoneVentilation:DesignFlowRate`'s Fan Pressure Rise/Fan Total Efficiency fields.
- *(see "Model Troubleshooting and Quality Assurance")* — not yet drafted; intended home for the infiltration double-counting/coverage-gap validation check flagged in §2.3, which doesn't exist yet.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

23. Energy Efficiency Measures (EEMs)

1. Overview

Energy modeling is a practice of asking "what if" questions — testing a hypothesis about a design and comparing the results. A single project routinely produces many simulated options over the course of a project.

Some energy modeling software handles this by saving a separate copy of the model for each option, then managing a whole set of files as the project moves forward. EP3's approach is different: one project, one model, one file. **Efficiency Measures (EEMs)** are the mechanism for collecting, storing, displaying, and combining the data for each alternative within that single file.

Global is the foundation model within each file — the building geometry, constructions, schedules, loads, and HVAC systems built up through the rest of this manual. An **EEM** is a named alternative to Global: it changes a handful of things — a construction, a lighting power density, an entire HVAC system — while everything it doesn't touch still comes from Global, or from another EEM it's built on.

Efficiency Measures can be used for several distinct kinds of work, all built on the same mechanism:

- **Comparing a baseline against a proposed design.** One EEM (or Global itself) holds the baseline configuration; another holds the proposed changes. Simulating both and comparing results is the standard way a design's savings get quantified.
- **Comparing individual options against each other.** A single decision — which chiller type, which insulation level, which control sequence — modeled as a handful of EEMs that each change just that one thing, to see the effect in isolation.
- **Assembling a complete model from several independent changes.** A lighting change, an envelope change, and an HVAC change can each be developed as a separate EEM, then combined into one final model, rather than building a single EEM that changes everything at once (§4).
- **Running a parametric sweep.** A Branching EEM holds several variants of one change — several lighting power densities, several chiller efficiencies — and every combination of selected variants simulates as a separate run, without building a separate EEM for each combination (§6).

Which EEM is currently active for editing is chosen from the EEM dropdown in the Top Control Strip *(see "The EP3 Interface and Model Settings", §2.1)*.

The full EEM list lives in Column 1 under **Model Alternatives (EEMs)**, with each EEM as a row in Column 2 and Global always present as the entry every EEM ultimately falls back to.

The rest of this chapter covers creating and configuring an EEM (§2), what changing a value in an EEM does and how to keep an EEM easy to reason about (§3), combining EEMs (§4), assembling a complete HVAC or Service Water Heating system alternative as an EEM (§5), Branching EEMs (§6), what gets simulated (§7), and comparing results across EEMs (§8). For how a complete HVAC or SWH system becomes an EEM — see *(see "The Model Wizard")* and *(see "The HVAC Outliner and Creating HVAC Systems")*.

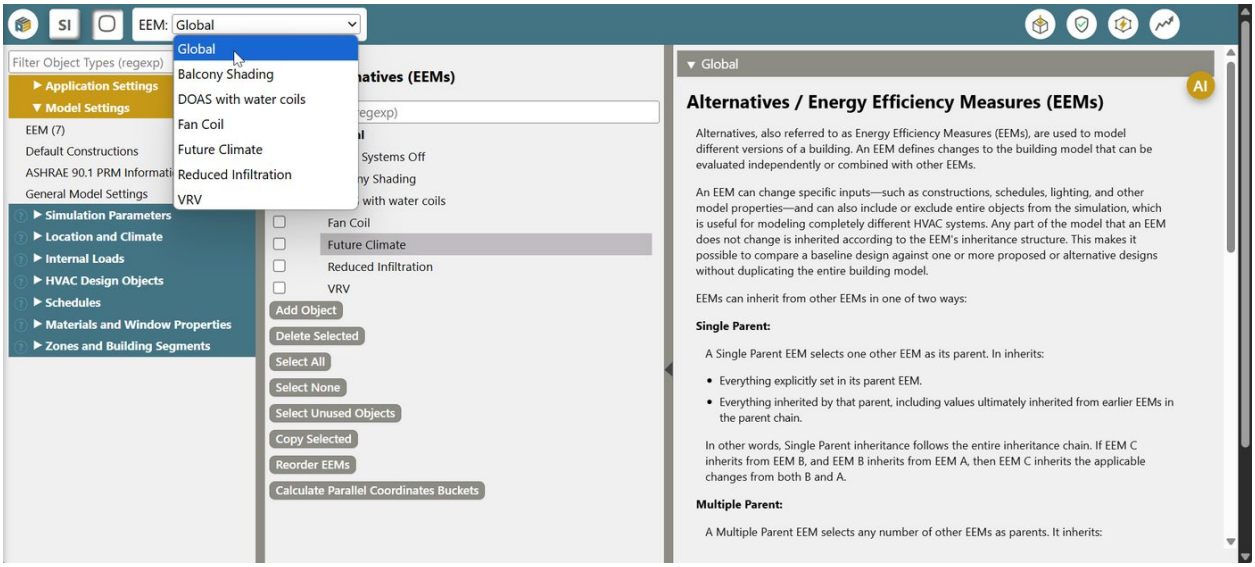


Figure 127: The EEM dropdown in the Top Control Strip, open to show the model's EEMs.

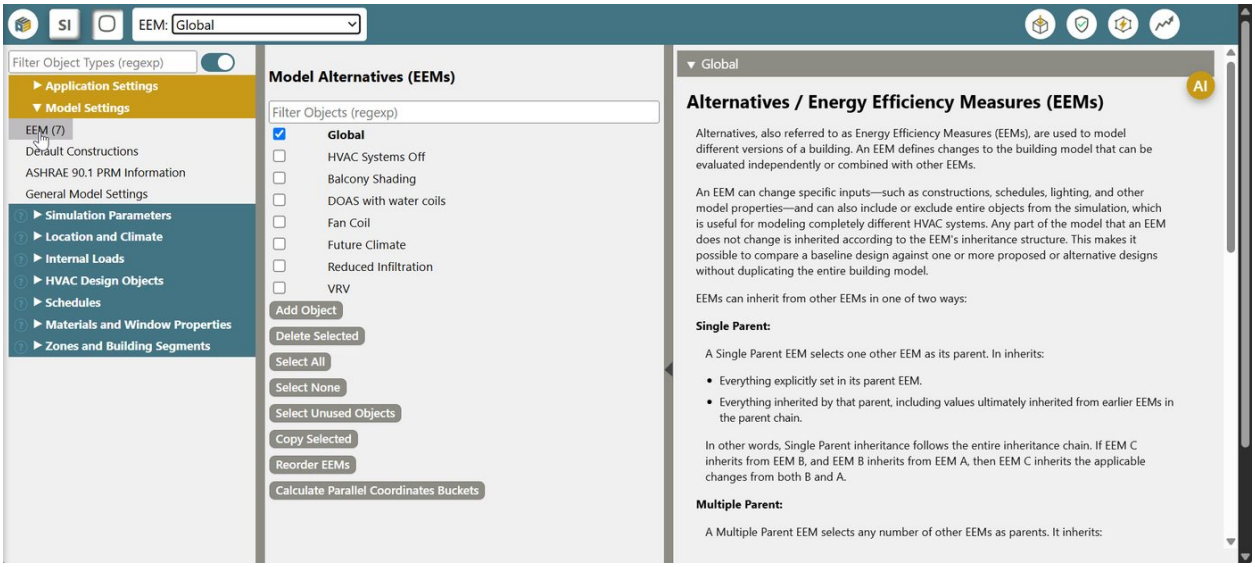


Figure 128: Column 1's "EEM (7)" entry under Model Settings, Column 2 listing all seven EEMs with Global selected, and Column 3 showing Global's description.

2. Creating and Configuring an EEM

A new EEM is added the same way as any other object — **Add Object** in Column 2 under **Model Alternatives (EEMs)**. Its name is editable directly in its Column 3 header.

An EEM carries a few settings independent of what it changes in the model:

- **Available for Simulation** — whether this EEM is offered as a choice in the Simulate/Export checklist *(see "Running Simulations")*. Turning this off simply reduces noise in the Simulate/Export view if there are EEMs that are not intended to be simulated on their own. This doesn't change how the EEM behaves or prevent other EEMs from building on it — it only removes it from that checklist.
- **Available as Baseline** — whether this EEM is offered as a choice when another EEM picks what to inherit from (§4). Turning this off doesn't disturb any EEM that already inherits from it; it only stops the EEM from being offered as a new parent going forward. Like **Available for Simulation**, this option simply reduces the visible EEMs in the select baseline dialog and dropdown.
- **Weather File** — an EEM can carry a weather file, overriding whatever it would otherwise inherit. Left unset, it shows which EEM (or Global) the weather file is coming from. A model comparing performance across climates, rather than across building changes, is the main reason to set this per EEM rather than only once at the model level.
- **About this measure** — a free-text description field, useful for recording what an EEM represents and why, especially once a model accumulates many of them.

Some EEMs are created by EP3 itself rather than by you, such as the auto-generated "HVAC System" and "SWH System" choice EEMs (§5) — and appear read-only: their name, inheritance, and values can't be edited, though they can still be selected as something to inherit from or inspected in Column 3.

---

3. Changing What an EEM Affects

3.1 Do most of your work in Global

Switching to an EEM and editing a field there stores that value specifically for that EEM, layered on top of whatever it inherits. This is powerful, but an EEM that accumulates changes unrelated to what it's meant to represent becomes hard to reason about and hard to compare against its siblings.

Build and refine the building in Global as much as possible, and switch to an EEM only to make the specific changes that define that alternative. The Custom Values report described in §3.3 is the tool for confirming an EEM has stayed scoped to what it's supposed to represent — periodically checking it against what you intended an EEM to change is good practice, especially before running a comparison.

3.2 Recognizing an inherited value

A field showing a rose-tinted background is displaying a value inherited from somewhere else rather than one set specifically here — the same convention used throughout EP3 wherever one thing can fall back to another's value: a Space or Zone inheriting a field from its `SpaceType`/`ZoneType` *(see "Internal Loads, Space Types, and Zone Types", §5.3)*, a composite object's per-reference field

falling back to a shared default, and an EEM inheriting a field from Global or from another EEM. Entering a value while an EEM is active clears that tint for this EEM specifically — it becomes this EEM's override, and every other EEM that doesn't set that field still inherits from wherever it did before.

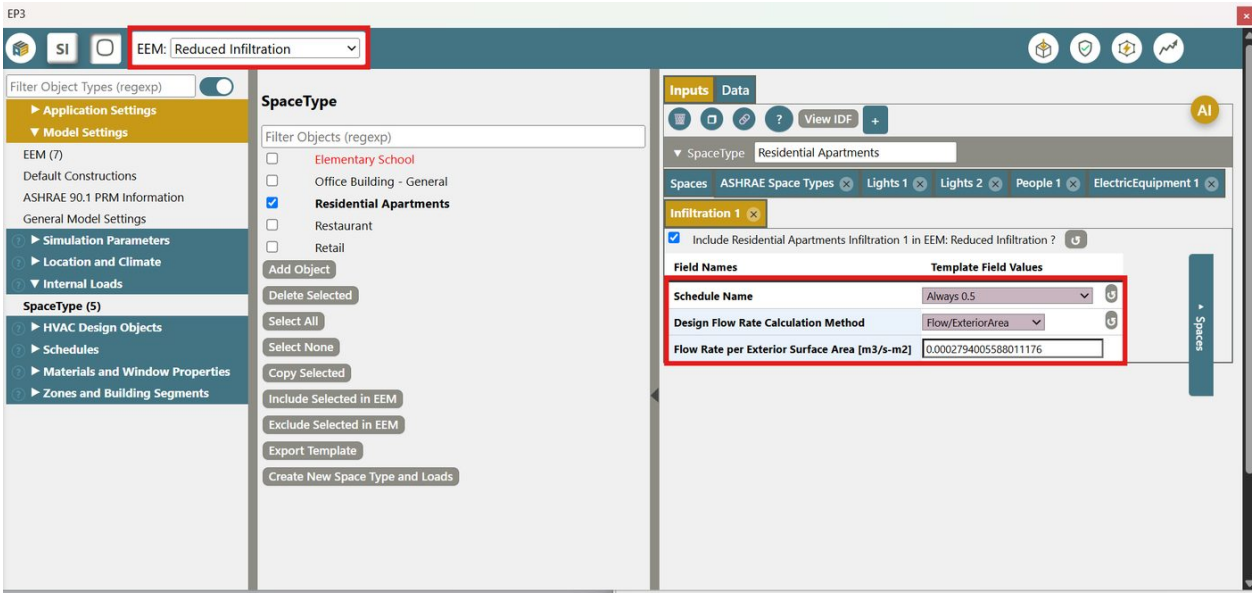


Figure 129: With the "Reduced Infiltration" EEM active, the Residential Apartments SpaceType's Infiltration sub-object still shows Schedule Name, Design Flow Rate Calculation Method, and Flow Rate per Exterior Surface Area with a rose-tinted background — this EEM hasn't overridden them, so they're still coming from the Template Field Values.

3.3 Including and excluding objects

An object can also be included or excluded entirely from an EEM, rather than having one of its fields overridden — the mechanism for building an EEM that represents a different HVAC system, for instance, by excluding one system's equipment and including another's.

Which control is available depends on the kind of object:

- A `SpaceType`/`ZoneType` sub-object such as Lights, People, or Infiltration, and a Zone itself, carry an **Include in EEM [name]?** checkbox directly in their dialog *(see "The EP3 Interface and Model Settings", §2.4)*. With the target EEM active, checking or unchecking it sets that object's state for that EEM directly; Column 2's **Include Selected in EEM** / **Exclude Selected in EEM** buttons apply the same setting to several selected objects at once.
- HVAC equipment, windows and doors, and Site or Building Shading — objects with a physical representation in the model, typically opened through the Selected Object Dialog *(see "The EP3 Interface and Model Settings", §3)* rather than by browsing Columns 1 and 2 — carry a **Manage Availability in EEMs** button in their dialog instead, opening a list of every EEM in the model with an On/Off/Inherit choice for the selected object or objects. Choices made in this list only take effect once **Save** is clicked; **Cancel** discards them.

Individual building surfaces (walls, roofs, floors) can't be included or excluded either way. Zones can — the mechanism for simulating a building with and without an entire zone, such as an addition

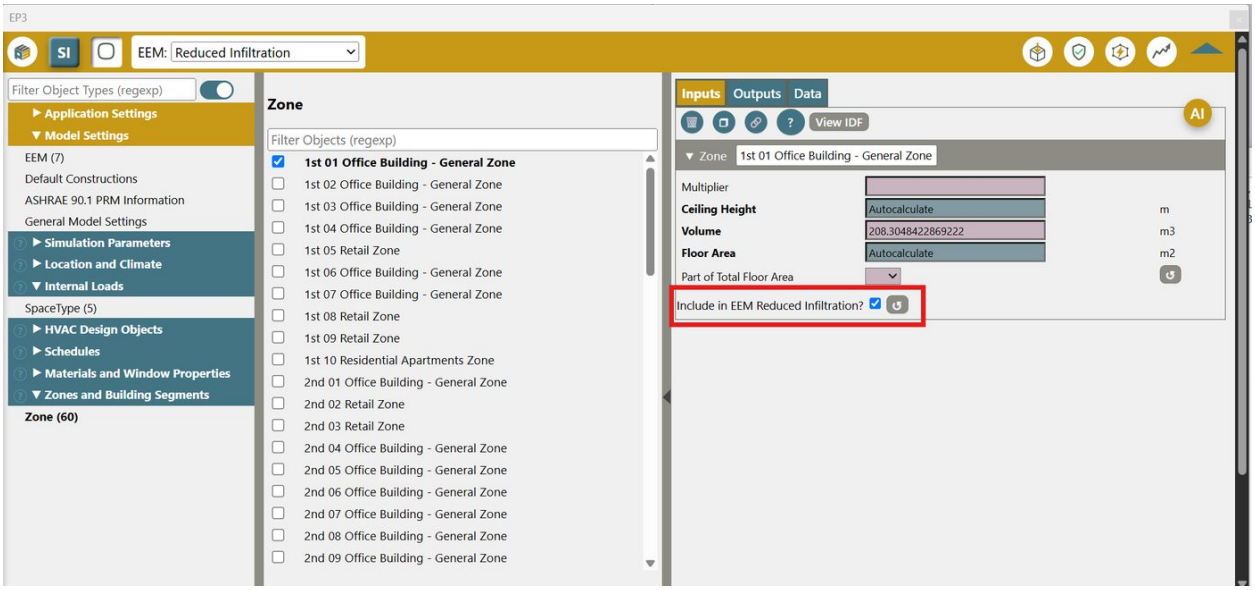


Figure 130: A Zone's dialog with the "Include in EEM Reduced Infiltration?" checkbox checked, and its reset control next to it.

or an atrium, as a geometry alternative.

To see what's currently on or off while you work rather than only after the fact, the Building Geometry Outliner's **Show / Hide Objects that are Off in this EEM** toggle and the HVAC Outliner's matching toggle bring hidden objects back into view without changing their inclusion state *(see "Creating Building Geometry") (see "The HVAC Outliner and Creating HVAC Systems")*.

3.4 Reviewing and resetting an EEM's changes

Every EEM lists exactly what it changes, in three sections:

- **Objects with Custom Values in this EEM** — every field this EEM overrides, grouped by object type, with the object, the field, and the value.
- **Objects Included in this EEM / Objects Excluded from this EEM** — every object this EEM turns on or off, grouped by object type.

Each entry, each object, each object type, and each section as a whole carries a small reset control that clears the corresponding value or values back to inherited. Resetting a whole section or type is a bulk action and asks for confirmation first; resetting a single field or object applies immediately.

---

4. Combining EEMs: Single Parent vs. Multiple Parent

Every EEM chooses one of two inheritance types, set in its **EEM Include Type** field:

- **Single EEM** (`single_parent`) — pick one other EEM to inherit from. This EEM automatically includes everything that EEM includes, and everything *that* EEM in turn includes, all the way back to Global — plus this EEM's changes on top. Choosing a parent this way extends an existing chain.

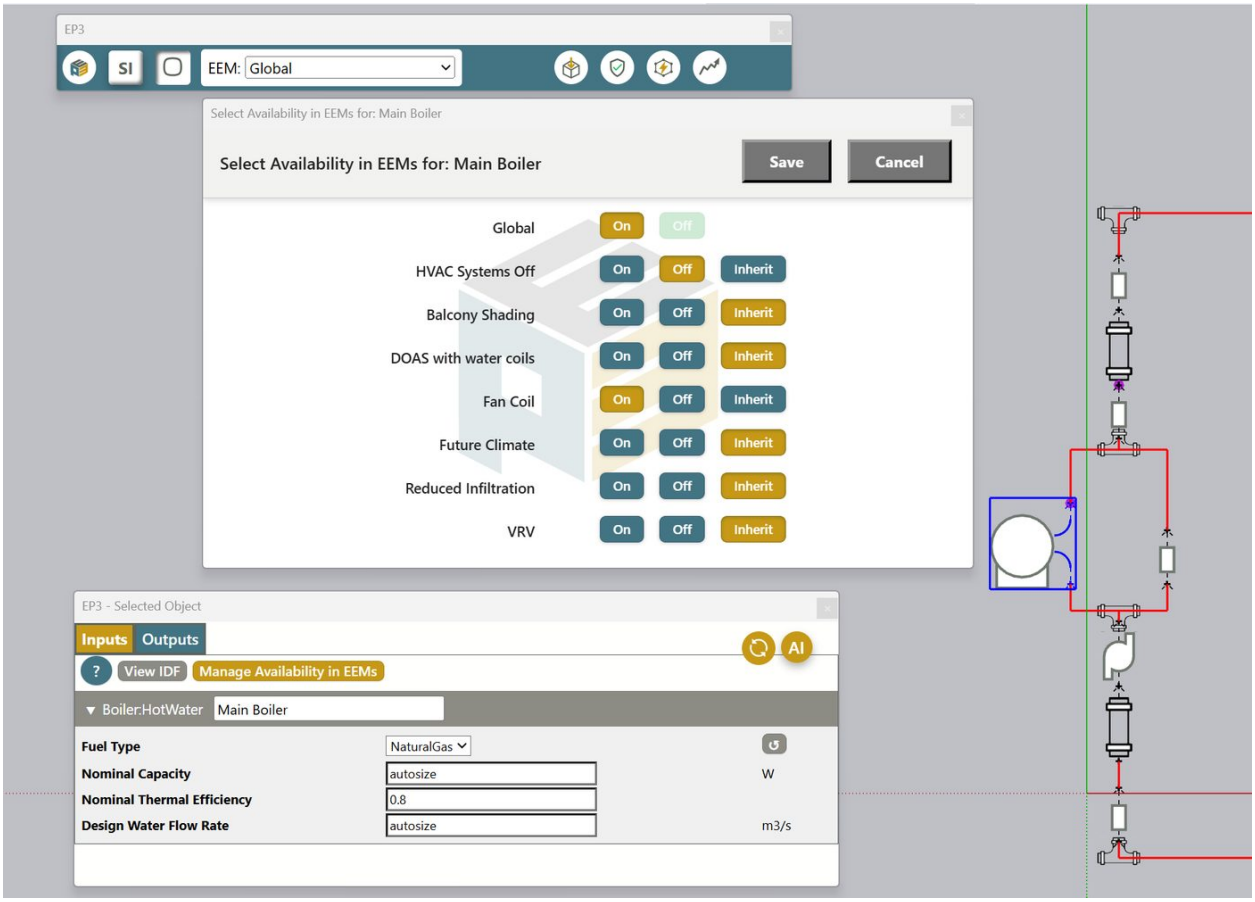


Figure 131: The "Select Availability in EEMs" list for a Boiler:HotWater object named "Main Boiler," opened from its Selected Object Dialog's "Manage Availability in EEMs" button, with an On/Off/Inherit choice per EEM and Save/Cancel buttons at the top.

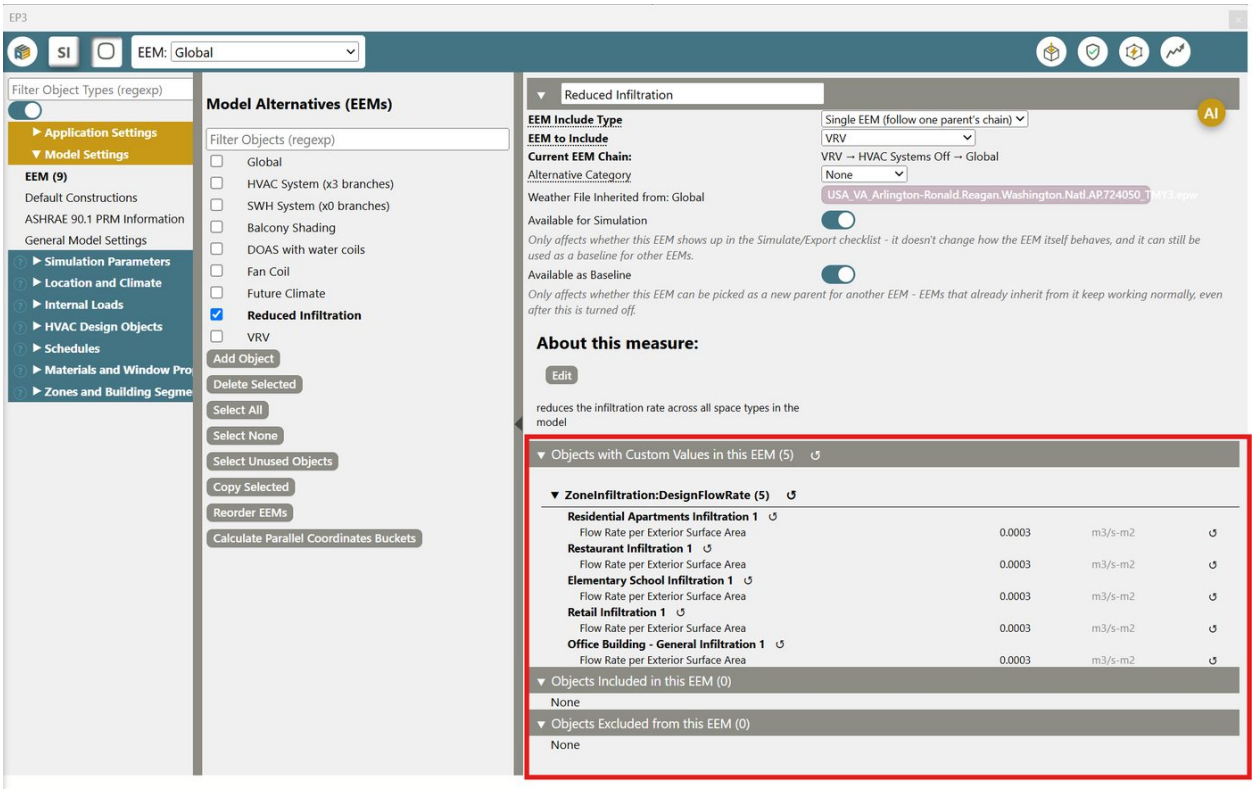


Figure 132: The "Reduced Infiltration" EEM's dialog, with "Objects with Custom Values in this EEM" expanded to show its ZoneInfiltration:DesignFlowRate group and the reset control next to the section header, the group header, and each individual value, and the empty "Objects Included in this EEM" / "Objects Excluded from this EEM" sections below it.

- **Multiple EEMs** (`bases`) — pick an ordered list of EEMs to inherit from. This EEM includes only what each listed EEM changes directly, not what those EEMs inherit in turn. If two listed EEMs change the same field, whichever is listed first takes precedence.

This distinction matters most when an EEM built with Single EEM inheritance is later included as one of several parents of a Multiple EEM. The chain it built up along the way does not come with it — only its direct changes do.

4.1 Worked example

Three EEMs:

EEM	Include Type	Inherits From	Direct Change
LPD	Single EEM	Global	Lighting Power Density → 0.7 W/ft²
Insulation	Single EEM	LPD	Wall `Material:NoMass` R-value → R-19
Proposed	Multiple EEMs	Insulation	*(none)*

Insulation inherits from LPD using Single EEM, so its effective chain is Insulation → LPD → Global. Switching to Insulation shows both the R-19 wall and the 0.7 W/ft² lighting power density, even though Insulation never touched lighting.

Proposed lists Insulation as its one included EEM, using Multiple EEMs. Multiple EEMs pulls in only what Insulation changes directly — its R-19 wall override — not what Insulation inherited from further up its chain. LPD is never part of Proposed's effective chain at all. Switching to Proposed shows the R-19 wall, but lighting power density is still Global's original value:

EEM	Lighting Power Density	Wall R-value
Global	(original value)	(original value)
LPD	0.7 W/ft²	(original value)
Insulation	0.7 W/ft²	R-19
Proposed	**(original value) — LPD change not included**	R-19

To bring the lighting change into Proposed as well, add LPD to Proposed's Multiple EEMs list directly (Insulation, LPD, in whichever order matters if they ever overlap) — Proposed can't reach it by way of Insulation, since Multiple EEMs never follows a listed EEM's chain.

4.2 Checking what an EEM includes

Regardless of which inheritance type is selected, an EEM's dialog always shows its full resolved chain — labeled **Current EEM Chain** for Single EEM, or shown on the resolved-chain button next to **Which EEMs to Include** for Multiple EEMs. Either way, it's the place to confirm an EEM includes what you intended before relying on it for a comparison or a final model.

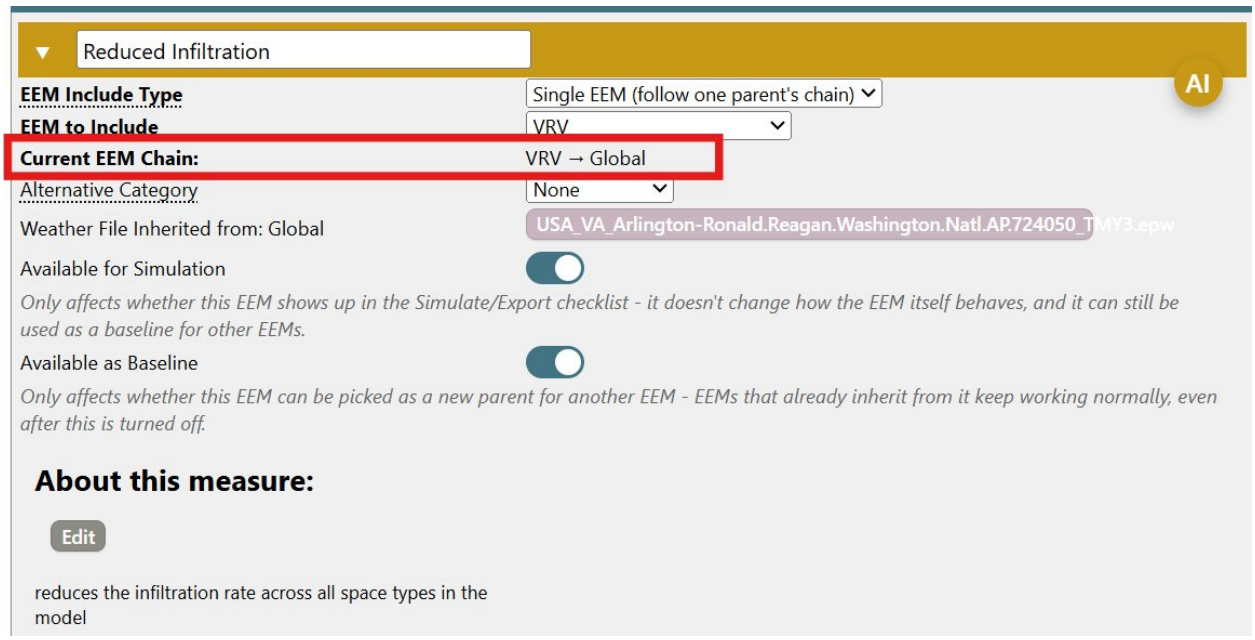


Figure 133: The "Reduced Infiltration" EEM, set to Single EEM and including VRV, with "Current EEM Chain: VRV → HVAC Systems Off → Global" displayed below it.

5. Building a Complete System Alternative: HVAC and SWH System Choice

Comparing whole HVAC or Service Water Heating systems against each other — rather than individual field changes — uses the same Single/Multiple EEM mechanism above, plus one additional piece: **Alternative Category**, a field on an EEM's Column 3 that tags it as "HVAC System" or "SWH System."

Tagging an EEM this way does two things:

- It identifies the EEM as a complete, self-contained system alternative — only the HVAC (or SWH) equipment **explicitly** included in that EEM and its chain get simulated.
- It adds the EEM to an automatically maintained list of options under a model-generated EEM named **HVAC System** (or **SWH System**), which collects every EEM tagged with that category so they can be compared, and combined, as a single named choice.

5.1 HVAC and SWH EEMs are created automatically by the model wizard

Each HVAC system template imported through the Model Wizard becomes an EEM, tagged **HVAC System** automatically *(see "The Model Wizard").* If a Service Water Heating system tagged **SWH System** already exists in the model, the newly imported HVAC EEM is automatically chained to it as a Single EEM parent — so choosing an HVAC system also brings its paired water heating system along, without combining the two by hand.

5.2 Combining HVAC systems that serve different zones

A building commonly needs more than one HVAC system — a VRV system for the perimeter offices and a packaged system for a server room, for instance. The Model Wizard imports each one as a separate EEM (§5.1), configured independently, with no relationship to which zones it should serve

in the final design.

Two steps turn these separate system alternatives into one working design:

- Assign each system's equipment to the zones it serves, using the HVAC Outliner's **View Zone Assignment** dialog *(see "The HVAC Outliner and Creating HVAC Systems")*, so each system conditions only its intended zones rather than the whole building.
- Create a new EEM — a "Proposed Systems" EEM, for instance — that includes both HVAC system EEMs using Multiple EEM inheritance (§4), combining their changes into the one EEM that gets simulated as the final design. Because Multiple EEM only pulls in what each listed EEM changes directly, not what it inherited in turn, remember to also select a paired SWH EEM, and any other EEMs that include objects or information your proposed design expects.

An EEM built this way, to combine several system alternatives, should not itself be tagged with an Alternative Category (§5) — that tag identifies a single, discrete system alternative, not a combination of several serving different parts of the building.

---

6. Branching EEMs: Many Variants from One EEM

A **Branching EEM** makes several specific variants of the same change — different lighting power densities, different boiler efficiencies — available from one place, instead of building a separate EEM for each variant by hand. It's scoped narrowly on purpose: one branching EEM targets fields on a single object type, applied to all objects of that type or to a chosen subset, varying one or more of that type's fields across its iterations. Any EEM that inherits from a branching EEM can then choose which of its iterations to include (or all).

A Branching EEM works differently from an ordinary EEM in one important way: it can't be selected in the EEM dropdown to switch to and edit live. Everything about it — which object type and fields it varies, and the value each iteration sets — is authored directly in the EEM's dialog, described next.

6.1 Enabling and authoring

An EEM becomes eligible for branching when it has no changes — before anything has been overridden, included, or excluded on it. Its dialog then offers an **Enable Branching** button; enabling branching cannot be undone.

Once enabled, an EEM's dialog shows a **Branching Iterations** section instead of the ordinary field-editing view:

1. Choose one **Object Type** the whole EEM will vary — every iteration targets this same type.
2. Optionally narrow the objects affected from "all objects of this type" down to a specific subset.
3. Choose one or more **fields** iterations are allowed to set.
4. Add iterations with **+ Add Iteration**, each setting values for the chosen fields (or leaving a field blank to inherit it as usual). An iteration is named automatically from its values, or can be given a custom name instead — this name is what identifies that iteration on the Parallel Coordinates plot (§8).

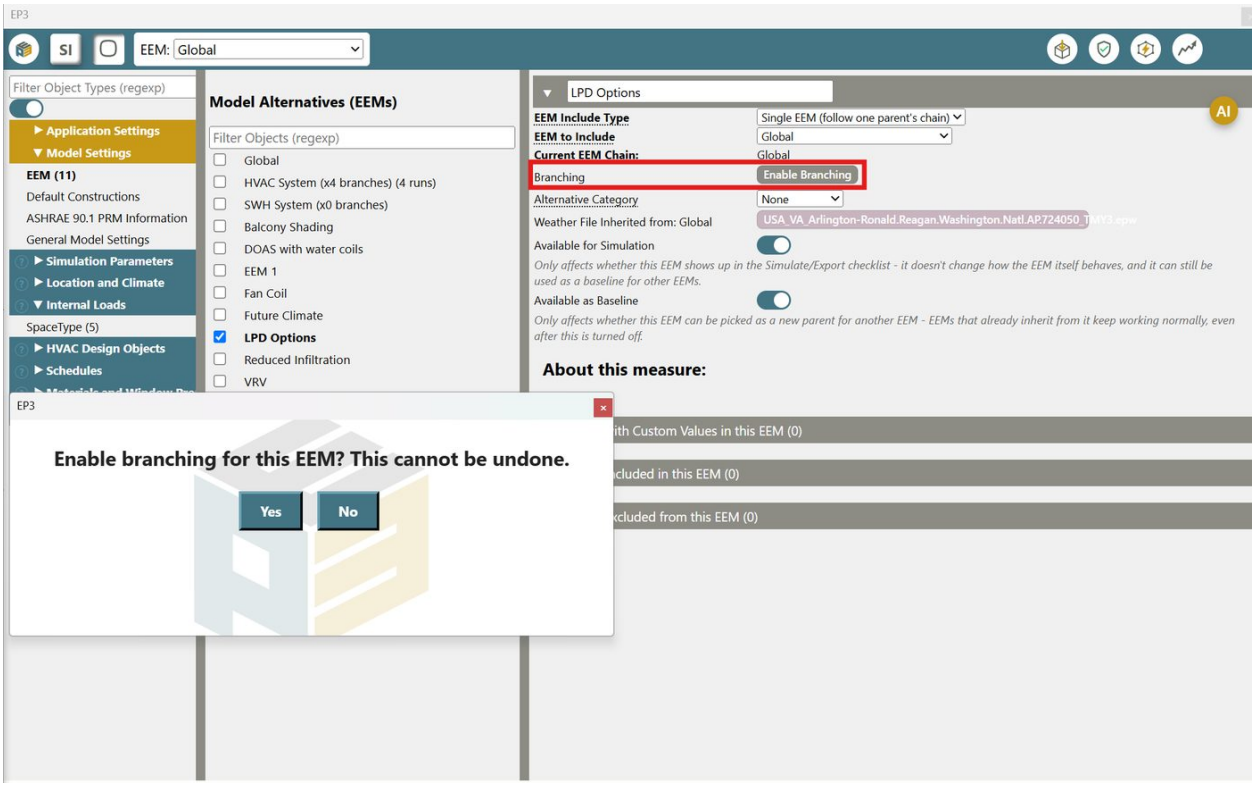


Figure 134: The "LPD Options" EEM's dialog before branching is enabled, with the "Enable Branching" button and the confirmation prompt that appears after clicking it.

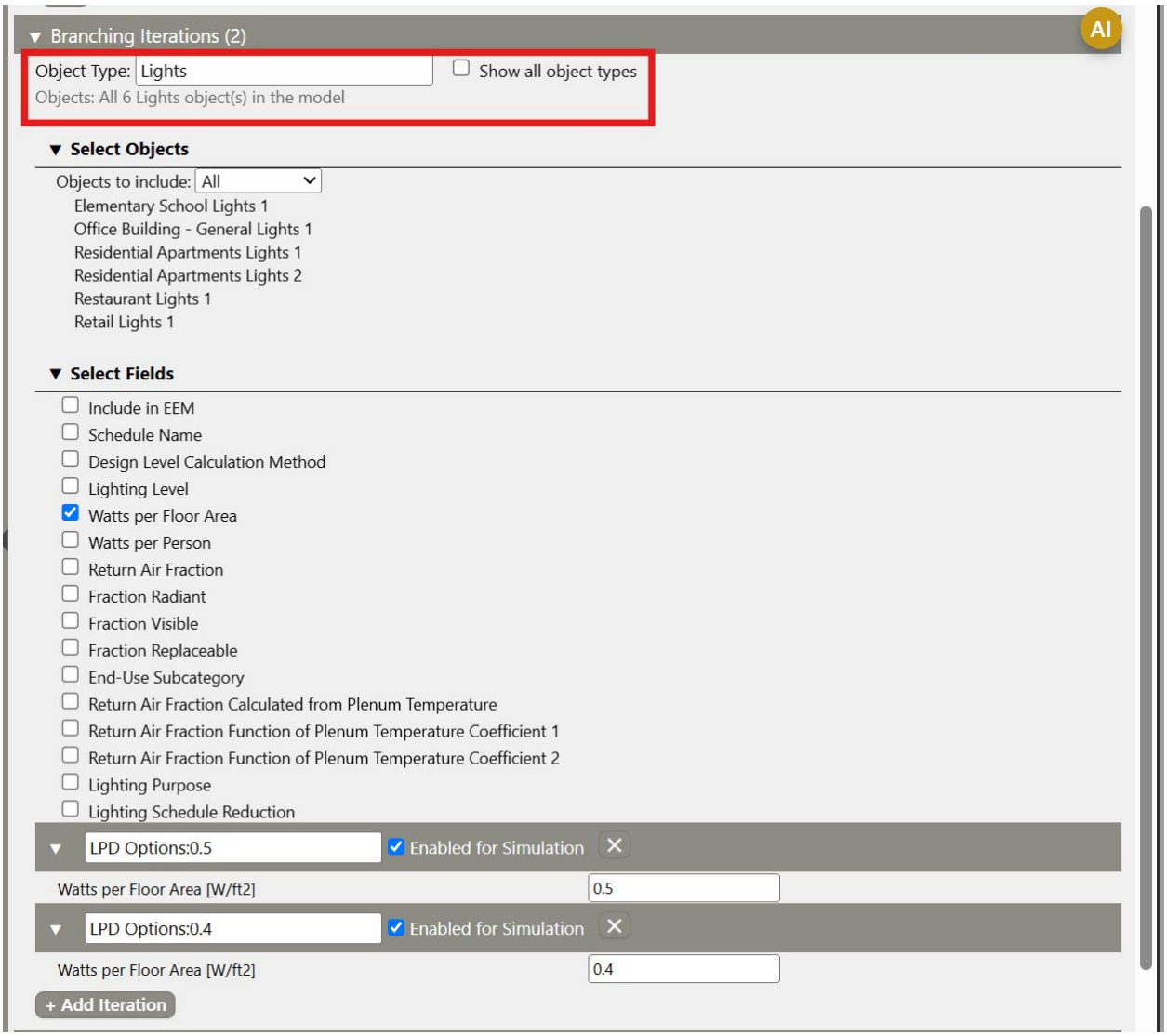


Figure 135: The "Branching Iterations" section for an EEM varying Lights, with Object Type set to "Lights," all 6 Lights objects in the model included, only "Watts per Floor Area" selected under Select Fields, and two iterations — "LPD Options:0.5" and "LPD Options:0.4" — each setting that one field.

6.2 Selecting iterations downstream

Any EEM with a branching EEM somewhere in its chain shows a **Simulation Branches from Parent EEMs** section, with one group per branching ancestor and a three-position switch per iteration: **Include**, **Exclude**, or **Inherit** (defer to whatever an ancestor further up the chain already chose, or Exclude if nothing ever set it). This is the same tri-state mechanism the automatic HVAC/SWH System Choice EEMs (§5) use for their auto-discovered options.

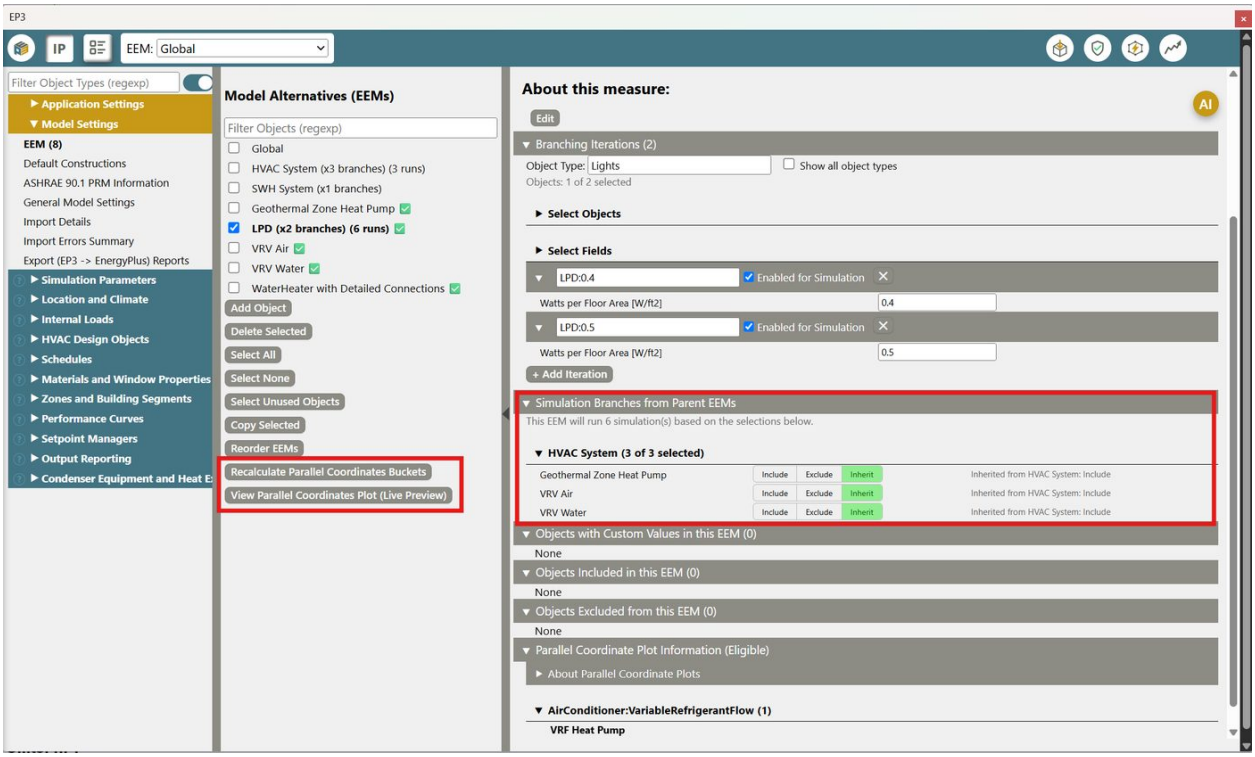


Figure 136: The "Proposed" EEM's "Simulation Branches from Parent EEMs" section, showing the "LPD Options" group with both of its iterations — "LPD Options:0.5" and "LPD Options:0.4" — set to Inherit, each noting it's inherited from LPD Options itself as Include; the note above states this EEM will run 2 simulations based on these selections.

6.3 Combinations

Selecting more than one iteration — on one branching EEM, or across several — produces one simulation run per combination. Two independent branching choices with three options each run all nine combinations. An EEM's dialog reports how many runs it will produce when more than one combination is selected.

---

7. Choosing What Gets Simulated

An EEM's **Available for Simulation** flag (§2) controls whether the EEM is offered in the Simulate/Export checklist. For an EEM whose chain includes any branching selections, the number of combinations (§6.3) determines how many runs that one checklist entry produces. The checklist and export process themselves are covered in *(see "Running Simulations").*

8. Comparing Alternatives: The Parallel Coordinates Plot

A Parallel Coordinates plot compares model alternatives side by side — one line per simulation run, one axis per tracked change — so a change made in one part of the model can be read against every alternative's results at once, without maintaining a separate comparison spreadsheet.

8.1 How axes are generated

Every axis comes directly from what an EEM's chain changes — the same data shown in the Custom Values/Included/Excluded report (§3.4) — rather than a separately maintained list:

- A field changed to the same value on every object of one type, across every EEM being compared, becomes one shared axis (Lighting Power Density, for example).
- A field changed to different values on different objects of the same type becomes one axis per object instead, so every EEM compared reads the same kind of axis.
- An EEM tagged with an Alternative Category (§5) contributes one shared "HVAC System" or "SWH System" axis rather than being tracked object by object.
- Any other object turned on or off becomes an axis tracking whether it's included or excluded.
- Simulation results — end-use energy consumption, currently, from the Annual Building Utility Performance Summary's End Uses table — are available as axes alongside the input axes above, so a plot can show inputs and outputs together in one view.

Every change an EEM makes has to land on exactly one axis. If a change can't be attributed to a tracked axis, that EEM is excluded from the plot by default, with the specific unresolved change identified rather than left unexplained.

8.2 Checking an EEM's eligibility while you build it

From the main dialog's list of **Model Alternatives (EEMs)**, **Calculate Parallel Coordinates Buckets** (or **Recalculate...**, once buckets already exist) checks every EEM against the tracked axes and marks each row with a badge: eligible, eligible with some values unresolved, or not yet eligible. Opening an EEM whose badge is showing shows a **Parallel Coordinate Plot Information** section in Column 3, explaining exactly which axis each of its changes maps to, or exactly which change is blocking it. This check has to be run explicitly — it isn't recalculated automatically on every edit — and clears itself out whenever the model changes enough to make its results stale.

Once buckets have been calculated, **View Parallel Coordinates Plot (Live Preview)** opens a chart of the current *inputs* in a separate window — check specific EEMs first to plot only those, or leave nothing checked to plot every eligible EEM.

> *Figure placeholder — Column 2's EEM list with eligibility badges visible next to several EEM names, and the "Calculate Parallel Coordinates Buckets" / "View Parallel Coordinates Plot (Live Preview)" buttons above the list.*

8.3 The Parallel Coordinates tab in Results

Once at least one EEM has been simulated, the Results dialog's **Parallel Coordinates** tab plots every simulated run together, reading each run's recorded snapshot of what is in the simulation, rather than the live model — so a later edit to an EEM doesn't alter how an already-simulated run

is displayed. A drawer alongside the chart lists every available axis, letting you show or hide axes and drag them into whatever order is most useful; that arrangement is remembered the next time the tab is opened. Dragging along any axis directly on the chart — brushing — selects a range of values on that axis and filters the plot down to only the runs that fall within it, letting you narrow the comparison to a subset of EEMs without leaving the chart. **Save as HTML...** freezes the plot exactly as currently arranged into a standalone file that opens on its own, with no access to the model required — suited to sharing a comparison with someone who doesn't have EP3.

A banner above the chart flags anything that affects how trustworthy the comparison is: a run excluded because one of its changes doesn't map to a tracked axis, a run whose results file may no longer match the snapshot it was plotted from, or two runs sharing an EEM name whose content differs between them. These are advisory — a flagged run still simulated and still exists — except for a run excluded outright for an unmapped change, which is left out of the plot until that change is resolved (§8.2).

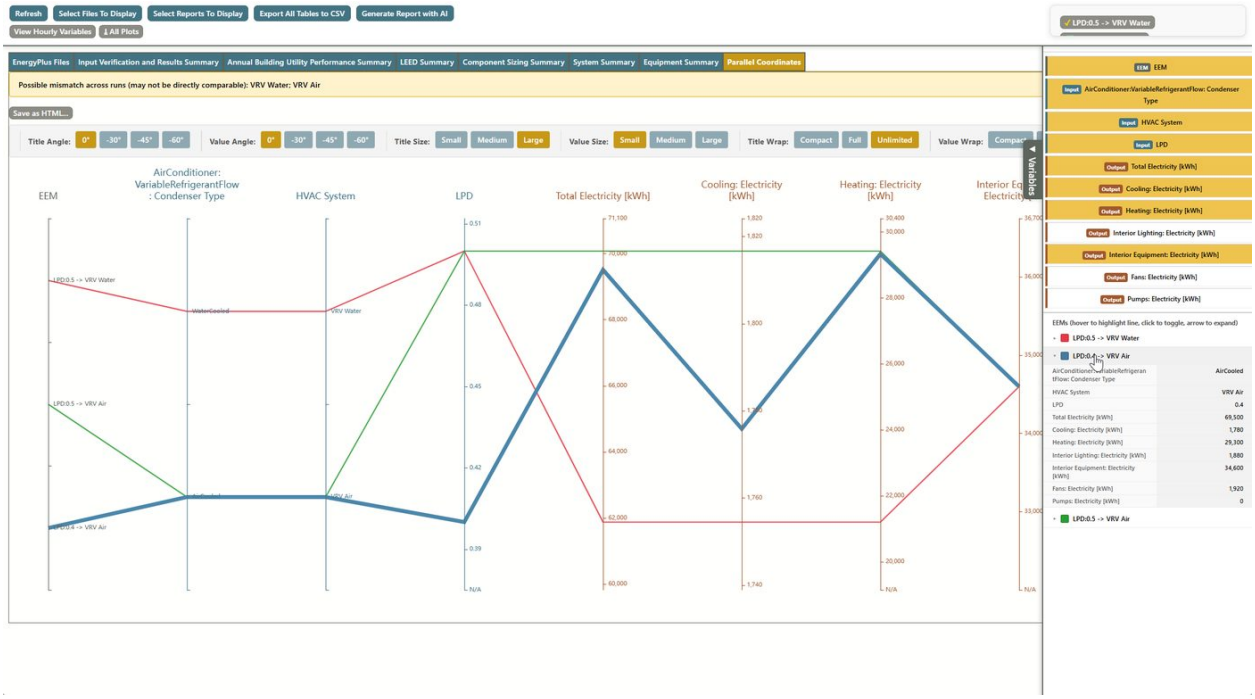


Figure 137: The Results dialog's Parallel Coordinates tab, with a label-conflict banner for "VRV Water"/"VRV Air," the chart plotting three runs across the AirConditioner:VariableRefrigerantFlow Condenser Type, HVAC System, LPD, and several energy-outcome axes, the Variables drawer expanded on the right with Input/Output candidates and the per-run EEM list below it, and "Save as HTML..." above the chart.

---

9. Where to Look Next

- *(see "The Model Wizard")* — how an HVAC or SWH system template import becomes a tagged EEM in the first place.
- *(see "The HVAC Outliner and Creating HVAC Systems")* — assembling HVAC systems, and the outliner's show/hide-when-off-in-this-EEM toggle.

- *(see "Internal Loads, Space Types, and Zone Types", §5.3)* — the rose-tinted inherited-value convention in its `SpaceType`/`ZoneType` context.
- *(see "Running Simulations")* — the Simulate/Export checklist and what happens to an EEM with multiple Branching combinations there.
- *(see "Viewing and Interpreting Results")* — the rest of the Results dialog beyond the Parallel Coordinates tab.
- *(see "ASHRAE 90.1 Appendix G Baseline Generation")* — the additional EEM tags the baseline generator uses to identify Baseline/Proposed pairs and building vintage, not covered here.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF placeholders once their target sections/pages exist (§9's Ch.24/25/27 targets in particular are not yet drafted); confirm final document format (HTML vs. PDF vs. other).

24. Model Troubleshooting and Quality Assurance

1. Overview

EP3 doesn't force a fixed modeling sequence — geometry, loads, and HVAC can be built in whatever order suits the project, and a Space or Zone can go without a Type assigned when there's nothing for one to configure. That flexibility is deliberate: EP3 is built to give experienced modelers real freedom and advanced capability without leaving them to work out what's missing unaided. The Validation dialog is the other half of that design — a standing check on the model that works regardless of the order the model was constructed, so the freedom to work the way you want doesn't come at the cost of guidance.

The Validation dialog serves three purposes:

- **Guidance** — showing what's missing or incomplete in the model, so there's always a concrete next step no matter where you started or what's been built so far.
- **Pre-empting EnergyPlus errors** — catching, before you ever simulate, the kind of missing or malformed setup that would otherwise surface as an EnergyPlus simulation error.
- **Catching quiet failures** — flagging conditions that wouldn't stop a simulation but would quietly produce results that don't represent what you intended, such as a setpoint manager unlikely to reach its loop's design temperature.

The rest of this chapter works outward from there: §2 is a full reference for the Validation dialog's checks, and §3 and §4 cover two things EnergyPlus itself produces that are worth reading closely once a simulation has run — unmet hours and the `.err` file — including how EP3's AI Assistant can help trace a specific cause. See *(see "The EP3 AI Assistant")*.

The Validation dialog checks the currently active EEM only. An object excluded from the active EEM (by an EEM-specific override, or because something it depends on is excluded) is invisible to Validation, the same way it would be invisible on export. If you're about to simulate several EEMs, run Validation once per EEM — switch the active EEM, then re-run — rather than trusting a single pass to have covered all of them. See *(see "Energy Efficiency Measures (EEMs)")* for switching the active EEM.

---

2. The Validation dialog

The Validation dialog opens from the **Validate Model** button on the Top Control Strip. *(see "The EP3 Interface and Model Settings", §2.1)*



Figure 138: The Validate Model icon.

2.1 How it works

Opening the Validation dialog runs every check below against the current EEM and renders the results as a row of tabs, one per category, each carrying a badge: a green checkmark for a category

with no issues, or a colored count (red for errors, gold for warnings) for one that found something. The tab that opens first is whichever category has the earliest issue, in the order the tabs are listed below — if nothing is wrong anywhere, the first tab opens instead.

A few things about how it behaves:

- **Checks run on open or explicit refresh, not live.** - the validation dialog checks are thorogouh and take enough time to evaluate that EP3 waits for the user to refresh the dialog instead of re-running on every user edit
- **A category can disappear entirely** rather than show as clean. If a check doesn't apply to this model at all — for example, if VRV equipment exists, there's nothing for the VRV terminal-list check, that category will not be rendered at all. Alternately, a green checkmark means the check ran and found nothing wrong; if there's a category that you're used to seeing that isn't visible, this means that check didn't apply.
- **Failing entries are usually clickable.** An object named in a check's results (a Space, a Loop, a Construction) carries a small arrow button that opens that object's dialog directly, so you don't have to go find it in order to edit and resolve the issue.
- **A few checks offer a one-click fix** — visible as a button under the check's description (e.g. Add Floor, Grow Walls, Switch to Automatic Infiltration Backstop). Most checks don't; you fix those by opening the flagged object and correcting it directly.

The rest of this section is a reference for what each tab checks.

2.2 Building Geometry

Check	What it's verifying
Floors	At least one Floor exists.
Spaces	At least one Space exists.
Exterior & Interior Walls	Every Floor has had its exterior and interior walls generated.
Roofs	At least one Roof exists.
Space Enclosure	Every Space is fully enclosed by surfaces (no gaps that would leave it without a real volume).
Assigned Constructions	Every surface has a construction assigned.
Unexpected Construction Types	A surface's assigned construction matches its boundary condition — an interior surface uses an interior-type construction, an exterior surface an exterior-type construction, and so on.

These run roughly in dependency order — there's no point flagging space enclosure or construction assignment on a model with no geometry yet, so the more basic checks come first.

2.3 Constructions

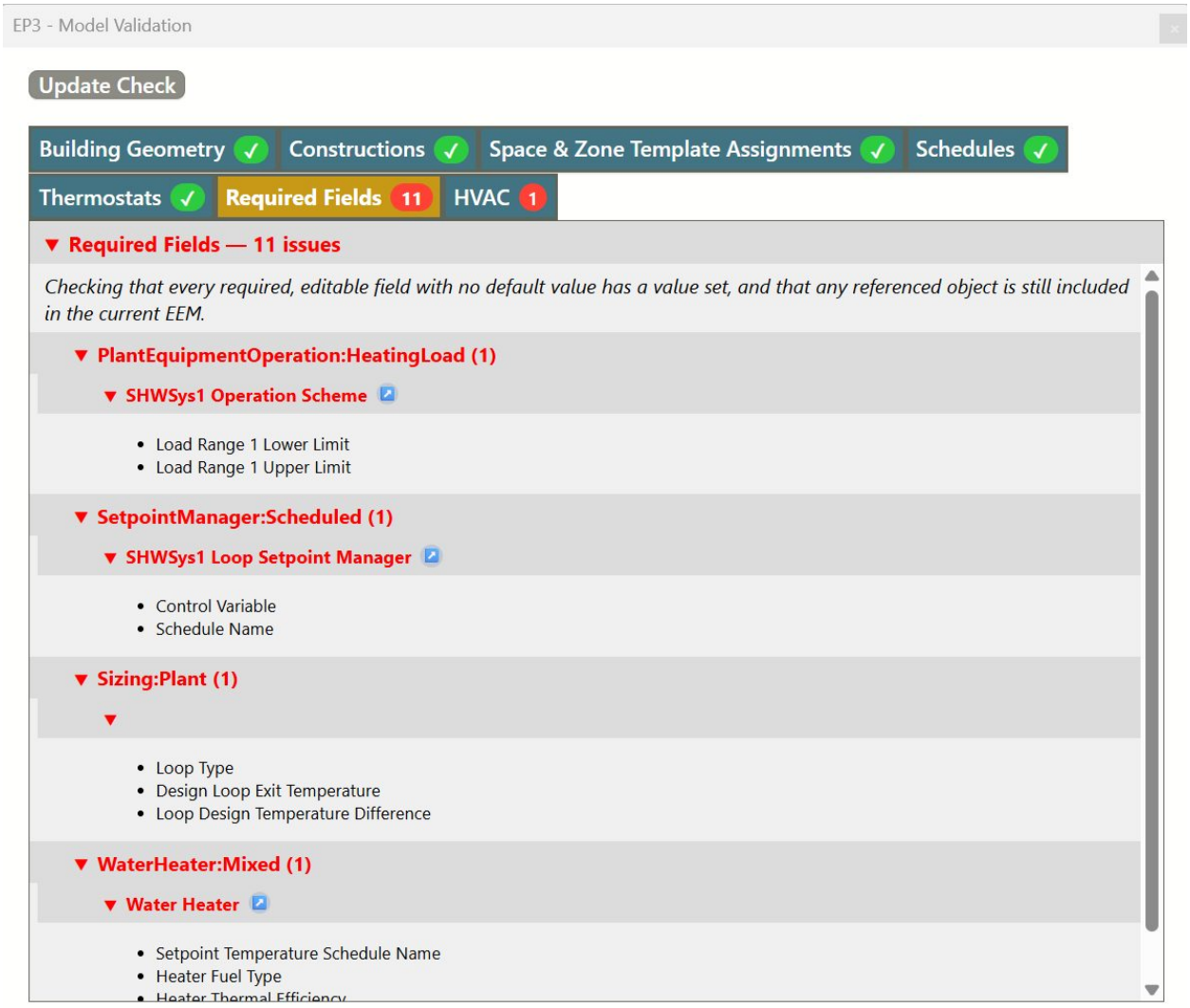


Figure 139: The Validation dialog's tab bar, most tabs showing a green checkmark, with the gold-badged Required Fields tab open to show several objects (PlantEquipmentOperation:HeatingLoad, SetpointManager:Scheduled, Sizing:Plant, WaterHeater:Mixed) each listing its missing fields under a clickable object link.

Check	What it's verifying
Assigned Layers	Every construction has at least one material layer.
Layer Types	An opaque construction contains only regular material layers (no `WindowMaterial:*` layers), and a glazing construction contains only `WindowMaterial:*` layers.
Opaque Construction Performance	Calculated R-value for each opaque construction, from its `Material`/`Material:NoMass`/`Material:AirGap` layers. A layer type outside that set (e.g. `Material:AirWall`) can't be calculated and is flagged rather than silently skipped.
Glazing Construction Performance	Calculated U-factor, SHGC, and VLT for each glazing construction — currently only supported for a construction built from a single `WindowMaterial:SimpleGlazingSystem` layer; a more complex glazing assembly is flagged as not yet calculable rather than evaluated incorrectly.
F-Factor Ground Floor Constructions	Calculated F-Factor for each `Construction:FfactorGroundFloor`, plus whether the exposed-perimeter length assigned on the construction is within 10% of the perimeter length calculated from the model geometry.
C-Factor Underground Wall Constructions	Calculated C-Factor and wall height for each `Construction:CfactorUndergroundWall`.

A ground-floor construction's perimeter is often set from an early design assumption, and the modeled geometry's perimeter can drift from it as the building evolves. A mismatch beyond 10% means the construction's assigned perimeter and the geometry no longer agree, changing the heat loss EnergyPlus calculates relative to what was intended.

See *(see "Constructions and Materials")* for how these fields are set.

2.4 Space & Zone Template Assignments

Check	What it's verifying
Space Template Assignments	Every Space has a `SpaceType` assigned.
Zone Template Assignments	Every Zone has a `ZoneType` assigned.

Both checks are skipped entirely — not run as a pass/fail check at all — for a model that clearly isn't using EP3's template structure in the first place (no `SpaceType`/`ZoneType` objects exist anywhere, but internal-loads or zone-sizing objects exist directly on Spaces/Zones instead, as is typical for a

model imported from outside EP3). Flagging every Space as "missing a template" on a model that was never built around templates would be noise, not a finding.

In a model that does use templates, an individual Space or Zone can still show as failing this check without needing a template added. A Space with no internal loads, or a Zone with no thermostat and no HVAC equipment, has nothing a template would configure — the fail is accurate, but not necessarily something to fix. See *(see "Internal Loads, Space Types, and Zone Types")*.

2.5 Schedules

Each `Schedule:Compact` is checked for complete, non-overlapping day-type coverage through 12/31; values that fall within its `ScheduleTypeLimits`; and, when it's the primary schedule driving an internal load or infiltration object, a plausible design-day profile. See *(see "Schedules")* for the day-type and design-day-profile conventions these checks assume.

2.6 Thermostats

One row per zone, listing its assigned thermostat(s):

Result	Meaning
Exactly one thermostat	Pass.
No thermostat	Warning — some zones (e.g. unconditioned space) legitimately have none.
More than one thermostat	Error — EnergyPlus resolves multiple `ZoneControl:Thermostat` assignments to the same zone in ways that are rarely what was intended.

2.7 Required Fields

This tab works differently from the others: instead of a fixed list of named checks, it sweeps every object in the model for any required, editable field that has no default value and isn't computed automatically — exactly the category of field a modeler could simply have left blank without EP3 catching it any other way. Results are grouped by object type, then by the specific object, listing which fields on it are missing.

A field referencing another object (an ID field pointing at a Schedule, a Construction, etc.) is checked further than "not blank": the check also confirms the referenced object still resolves to something included in the current EEM. A field that names a real object that happens to be excluded from this EEM is treated the same as a blank field, since it would export the same way — as nothing.

2.8 HVAC

This tab only appears once HVAC objects can exist in the model at all. It covers zone-level equipment assignment, VRV-specific terminal connections, and — in the most detailed part of the Validation dialog — a per-loop card for every Plant Loop and Air Loop in the model.

2.8.1 Zone-level checks

Check	What it's verifying
Zone HVAC Equipment	Every zone with a thermostat has at least one ZoneHVAC object or AirTerminal assigned. A thermostat with nothing to satisfy it causes a fatal EnergyPlus export error.
VRV Terminal Unit List Assignments	Every VRV zone terminal is connected to a VRV outdoor unit (only appears when VRV equipment exists in the model).

2.8.2 Per-loop checks (Plant Loops / Air Loops) Every Plant Loop and every Air Loop included in the EEM gets a collapsible card, headed by its name and a one-line sizing summary (loop type and design supply temperature, or design supply air temperature range, when `Sizing:Plant`/`Sizing:System` is set up). Inside each card:

Setpoint Manager Coverage — the most involved check here. It looks at every setpoint manager connected anywhere on the loop and asks several questions at once:

- Does something control the loop's supply outlet node(s)? An uncontrolled outlet is a fail — the loop has no way to know what temperature to deliver.
- Where a setpoint manager does control the outlet, does its setpoint value overlap the loop's design supply temperature (from `Sizing:Plant`/`Sizing:System`)? A setpoint that never reaches the design condition is a warning — the loop was sized for a condition its controls will never produce.
- For a `SetpointManager:OutdoorAirReset` on a PlantLoop specifically: even when the reset range does overlap the design temperature, a reset spread wider than 3°F/1.7°C on the off-design end (lower for a Heating loop, higher for a Cooling loop) is flagged as a warning worth a second look at unmet hours — see §3. A wide reset is not inherently wrong (outdoor-air reset is usually configured deliberately), but the loop runs at reduced capacity across that range.
- Does a `SetpointManager:MixedAir` have something upstream setting the reference node it copies its value from? `SetpointManager:MixedAir` has no setpoint value; it copies whatever the reference node's setpoint manager already produced. If nothing controls that reference node, the copy has nothing to copy, and the mixed-air/economizer control chain is broken. This is a fail, not a warning.
- Do multiple setpoint managers share the same node? EnergyPlus doesn't pick a winner among setpoint managers on the same node — it's genuinely possible for two to fight (a fixed `SetpointManager:Scheduled` value against a load-based one, for instance). Any shared node is flagged; it's escalated to a fail specifically when a humidity setpoint manager is involved, or when a `SetpointManager:Scheduled` is one of the pair, since those are the combinations most likely to actively conflict rather than coexist harmlessly.

Branch Components — every branch (inlet, outlet, and each parallel branch) needs at least one component in the EEM, or EnergyPlus fails on export. A branch that's only meant to pass fluid through without adding load or requesting flow — an unused bypass, for instance — needs a `Pipe:Adiabatic` placed on it rather than being left empty. This also checks that at least one

populated parallel branch survives between each side's splitter and mixer; an emptied-out side has no path across the loop at all.

Supply Side Pump *(PlantLoop only)* — flags a PlantLoop with no `Pump:VariableSpeed`/`Pump:ConstantSpeed`/H anywhere on its supply side. Without one, nothing circulates fluid through the loop.

Supply Branch Fan Type *(AirLoopHVAC only)* — flags a `Fan:OnOff` placed directly on an AirLoopHVAC's supply branch. EnergyPlus only allows a cycling fan (`Fan:OnOff`) as zone equipment; a continuous fan (`Fan:VariableVolume`, `Fan:ConstantVolume`, or `Fan:SystemModel`) is required on the branch itself. Unlike the other checks here, this one renders nothing at all on a compliant loop — it would be pure noise to show a green checkmark on every single air loop for a restriction only a handful of them could ever violate.

See *(see "Air Loops, Branches, and Nodes")* and *(see "Central Plant Equipment and Plant Loops")* for the underlying loop, branch, and setpoint manager mechanics these checks assume.

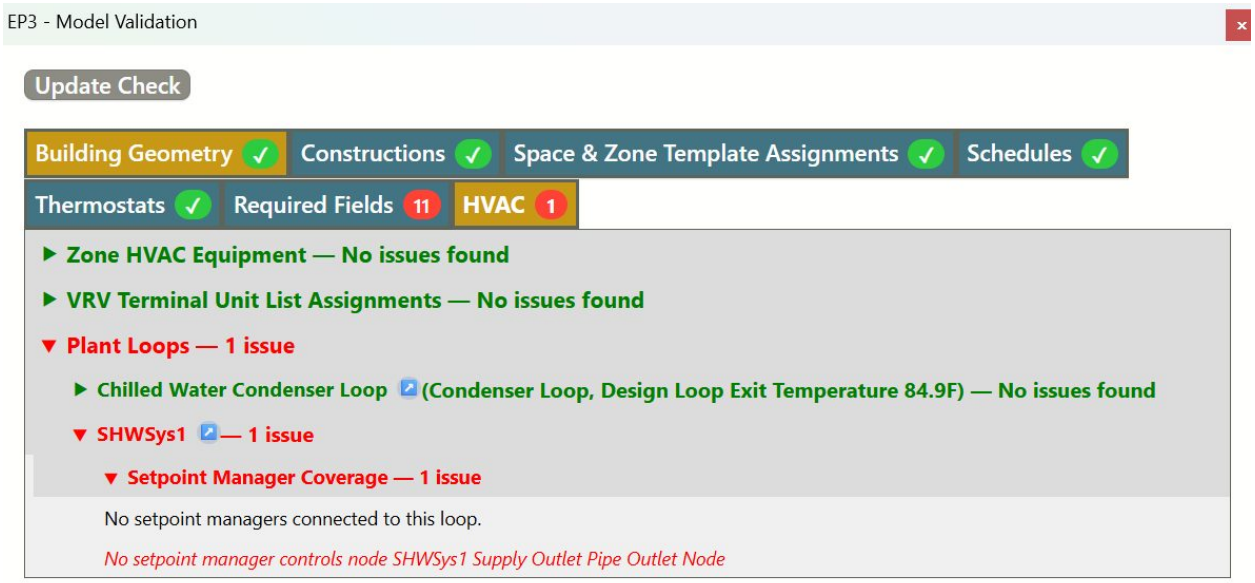


Figure 140: The HVAC tab's Plant Loops section: the Chilled Water Condenser Loop card showing no issues, and the SHWSys1 card flagged with one issue — its Setpoint Manager Coverage subsection expanded to show that no setpoint manager controls the loop's supply outlet node.

2.8.3 Exhaust Fan Make-Up Air A model-wide check, not per-zone: it looks for any `Fan:ZoneExhaust` anywhere in the model and confirms a source of make-up outdoor air is configured for it.

Result	Meaning
No make-up air source configured at all	Fail — every `Fan:ZoneExhaust` in a zone without an always-running AirLoopHVAC will pull an unbalanced, unphysical amount of air out of that zone.
Make-up air relies on `ZoneMixing` rather than automatic infiltration	Warning — EP3 can't verify from here that every exhaust-fan zone is covered by a `ZoneMixing` source zone or an always-running AirLoopHVAC; that needs to be confirmed by hand.
Automatic infiltration backstop configured for all zones	Pass.

This fires whenever any `Fan:ZoneExhaust` exists at all, even in a zone already served by an AirLoopHVAC — an airloop only induces make-up air while its fan is running, so a zone with intermittent airloop operation still benefits from the backstop. Both the fail and warning states offer a one-click fix button.

2.9 Service Water Heating

Appears once any `WaterUse:Connections` object exists, and only when more than one does. It flags any `WaterUse:Connections` with an empty `WaterUse:Equipment` list as a warning: at export, only one such "empty" Connections object silently adopts every `WaterUse:Equipment` left unassigned elsewhere in the model, and every other empty one exports with nothing connected to it. In a model with more than one water heating system, that adoption behavior is rarely what was intended — assign each system's equipment to its Connections list explicitly. See *(see "Service Hot Water and Solar Water Heating").*

---

3. Diagnosing unmet hours

Time Setpoint Not Met (System Summary), one of the tables shown by default in the Results dialog (*(see "Viewing and Interpreting Results")* §3.1), is the primary signal that a model isn't delivering the conditions it claims to. It reports, separately for heating and cooling, the number of occupied hours a zone's temperature fell outside its thermostat deadband.

A small amount of unmet hours are expected in any simulation - a count that's small and concentrated at the extremes of the design conditions is expected; a count that's large or spread broadly across the run period points to a problem. The most common causes tie back to checks and mechanics covered elsewhere in this manual, roughly in the order worth checking:

- **Equipment scheduled off during occupied hours** — an Availability Schedule Name (or another on/off schedule the equipment depends on) that's off, or zero, for hours the zone needs conditioning; the simplest cause, and the first one worth ruling out (*(see "Schedules")*, *(see "Fans and Fan Power")*).
- **Undersized or missing zone equipment/airflow** — `Sizing:Zone`'s design airflow/capacity assumptions not matching what's installed (*(see "Zone-Level HVAC Equipment and Zone Sizing")*).

- **A loop setpoint that doesn't reach its design condition, or an outdoor-air-reset range that drifts too far off-design** — exactly what §2.8.2's Setpoint Manager Coverage check flags; if that check passed clean, this usually isn't the cause.
- **Insufficient central plant capacity** relative to what the airloops/zones connected to it are asking for (*(see "Central Plant Equipment and Plant Loops")*).
- **A narrower schedule mismatch** — equipment availability or fan operating-mode schedules that run during some, but not all, of the hours the zone is occupied and expected to be conditioned (*(see "Schedules")*, *(see "Fans and Fan Power")*).
- **Genuinely extreme design-day/weather conditions** relative to what the equipment was sized for (*(see "Weather, Climate, and Design Conditions")*).

Once you've narrowed to a specific zone and time period, the Timestep Viewer is where you look at what the equipment did during an unmet-hours period — supply air temperature, flow rate, and setpoint at the relevant node, hour by hour. That only works for a node whose data was requested before you simulated; NodeReport is how you request it at a specific physical node when the equipment's Outputs tab doesn't already cover it. See *(see "Viewing and Interpreting Results")* §5.

EP3's AI Assistant can trace a specific unmet-hours result back through a model's connections faster than tracing it by hand in a large model. See *(see "The EP3 AI Assistant")*.

---

4. Reading the `.err` file

Every simulation produces an `.err` file alongside its other output, reachable from the Results dialog's file panel (*(see "Viewing and Interpreting Results")* §8). EnergyPlus writes it as a flat, chronological log, organized loosely into the order the simulation proceeded — input processing, then each sizing pass, then the environments simulated — rather than grouped by severity or topic. Every message in it is tagged with one of a small set of literal markers:

- `** Fatal **` — the run stopped. A fatal error is almost always preceded, a few lines above it, by the real underlying message explaining why; the fatal line itself is often just "Program terminated" or similar.
- `** Severe **` — doesn't stop the simulation, but represents something that will affect the result and needs to be investigated.
- `** Warning **` — worth a skim; a warning is often EnergyPlus falling back to a default or benign assumption, but not always.
- `** ~~~ **` — a continuation of the message immediately above it, not a new message.

EP3's file-status indicator (the green/gold/red checkmark shown per file in the Results dialog, and the fatal/severe/warning counts in its tooltip) is built by scanning the `.err` file's text for exactly these markers and counting them — there is no separate EP3-side interpretation of what any individual message means. Reading the file is the way to find out.

Given a run with a fatal error, start at the top of the file and read forward rather than jumping straight to the fatal line — the cause is usually a severe or warning message from earlier in the same run, sometimes from a much earlier stage (input processing or a sizing pass) than where the fatal one appears. Output Details and Examples documents the `.err` file's structure and message conventions in full; use it alongside the Input/Output Reference for what a specific message text means for the object it names.

EP3's AI Assistant can read an `.err` file directly and walk through what caused a fatal error or a cluster of severe messages, tracing the underlying issue back to the specific model objects involved rather than requiring a line-by-line manual read. See *(see "The EP3 AI Assistant").*

---

5. Where to Look Next

- *(see "Viewing and Interpreting Results")* — interpreting substantive output once a model has passed the checks in this chapter, and requesting node-level data (via NodeReport or an object's Outputs tab) ahead of a simulation.
- *(see "Energy Efficiency Measures (EEMs)")* — switching the active EEM, which the Validation dialog's checks are always scoped to.
- *(see "Air Loops, Branches, and Nodes")* / *(see "Central Plant Equipment and Plant Loops")* — the loop, branch, and setpoint manager mechanics behind §2.8.2's checks.
- *(see "Zone-Level HVAC Equipment and Zone Sizing")* — zone-level sizing assumptions, a common root cause of unmet hours.
- *(see "The EP3 AI Assistant")* — automated unmet-hours cause-tracing and `.err` file diagnosis.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

25. Running Simulations

1. Overview

A model is ready to simulate once it has geometry, constructions, internal loads, and an HVAC system in place, and a weather file assigned to it (*(see "Weather, Climate, and Design Conditions")*). This chapter covers the settings that control how EnergyPlus runs a simulation, the Simulate/Export dialog used to launch one, and what happens between clicking **Simulate** and a completed set of output files.

EP3 allows a model to be assembled in enough different ways that no fixed set of checks can guarantee every model will simulate successfully in EnergyPlus. The Validation dialog is where that gets resolved: it reports schema violations, missing required fields, and cross-object conflicts, so you can address a problem or decide it's acceptable for your case without EP3 forcing one particular solution. *(see "Model Troubleshooting and Quality Assurance")* covers the Validation dialog in full; running it before simulating is the most reliable way to catch a problem before simulating, and can minimize EnergyPlus simulation errors or unmet hours.

(see "The EP3 Interface and Model Settings", §7.3) covers configuring EP3's EnergyPlus installation path, required before any simulation can run. *(see "Energy Efficiency Measures (EEMs)")* covers what an EEM is and how to build one; this chapter assumes at least one EEM already exists and focuses on running it.

What this chapter defers:

- Interpreting simulation output — EUI, charts, plant performance indicators, the timestep viewer — *(see "Viewing and Interpreting Results")*.
- Reading the EnergyPlus `.err` file, unmet hours, and other troubleshooting — *(see "Model Troubleshooting and Quality Assurance")*.
- Combining and comparing EEM alternatives — *(see "Energy Efficiency Measures (EEMs)")*.

---

2. Simulation Settings

Simulation-control objects live in the Main Dialog's Column 1 under the **Simulation Parameters** group (*(see "The EP3 Interface and Model Settings", §5)* covers the three-panel Main Dialog mechanism generally, including the black/red convention used below). `SimulationControl` and `RunPeriod` are the only two object types in this group required for a standard annual simulation, and EP3 creates one instance of each automatically in a new model.

2.1 `SimulationControl`

EP3 creates this object automatically with Zone, System, and Plant sizing all enabled and both "Run Simulation for…" fields set to Yes.

Field	Notes
Do Zone Sizing Calculation	Runs zone sizing from `Sizing:Zone` objects and autosized fields.
Do System Sizing Calculation	Requires Do Zone Sizing Calculation to also be Yes.

Field	Notes
Do Plant Sizing Calculation	Runs plant sizing from `Sizing:Plant` objects and autosized fields.
Run Simulation for Sizing Periods	Executes `SizingPeriod:*` objects and reports their results.
Run Simulation for Weather File Run Periods	Executes `RunPeriod` objects and reports their results.
Do HVAC Sizing Simulation for Sizing Periods	Repeats sizing periods for advanced coincident plant sizing.
Maximum Number of HVAC Sizing Simulation Passes	Only relevant when the previous field is Yes.

Turning off Run Simulation for Weather File Run Periods and running only sizing periods is a fast way to check that a model sizes and exports cleanly before committing to a full annual run.

Maximum HVAC Iterations lives on `ConvergenceLimits` (§2.3), not on this object.

2.2 `RunPeriod`

`RunPeriod` sets the annual (or partial-year) simulation window. EP3 creates one instance automatically in a new model, named YEAR, spanning the full calendar year (Begin Month/Day 1/1, End Month/Day 12/31).

Field	Notes
Begin Month / Begin Day of Month / Begin Year	Begin Year is optional; if blank, EP3 picks a year matching Day of Week for Start Day.
End Month / End Day of Month / End Year	
Day of Week for Start Day	Defaults to Sunday if no year is given.
Use Weather File Holidays and Special Days	Default Yes.
Use Weather File Daylight Saving Period	Default Yes.
Apply Weekend Holiday Rule	Default No. If Yes, a single-day holiday falling on a weekend moves to the following Monday.
Use Weather File Rain Indicators / Use Weather File Snow Indicators	Both default Yes.
Treat Weather as Actual	Default No.
First Hour Interpolation Starting Values	Default Hour24. Controls which values from the first day are used to interpolate the first simulation hour, when the weather file's timestep is longer than the simulation timestep.

A model can have more than one `RunPeriod`; each is run independently.

2.3 `Timestep` and `ConvergenceLimits`

Unlike `SimulationControl` and `RunPeriod`, EP3 doesn't create these objects automatically. If a model has no `Timestep` or `ConvergenceLimits` instance, EnergyPlus runs with the defaults below.

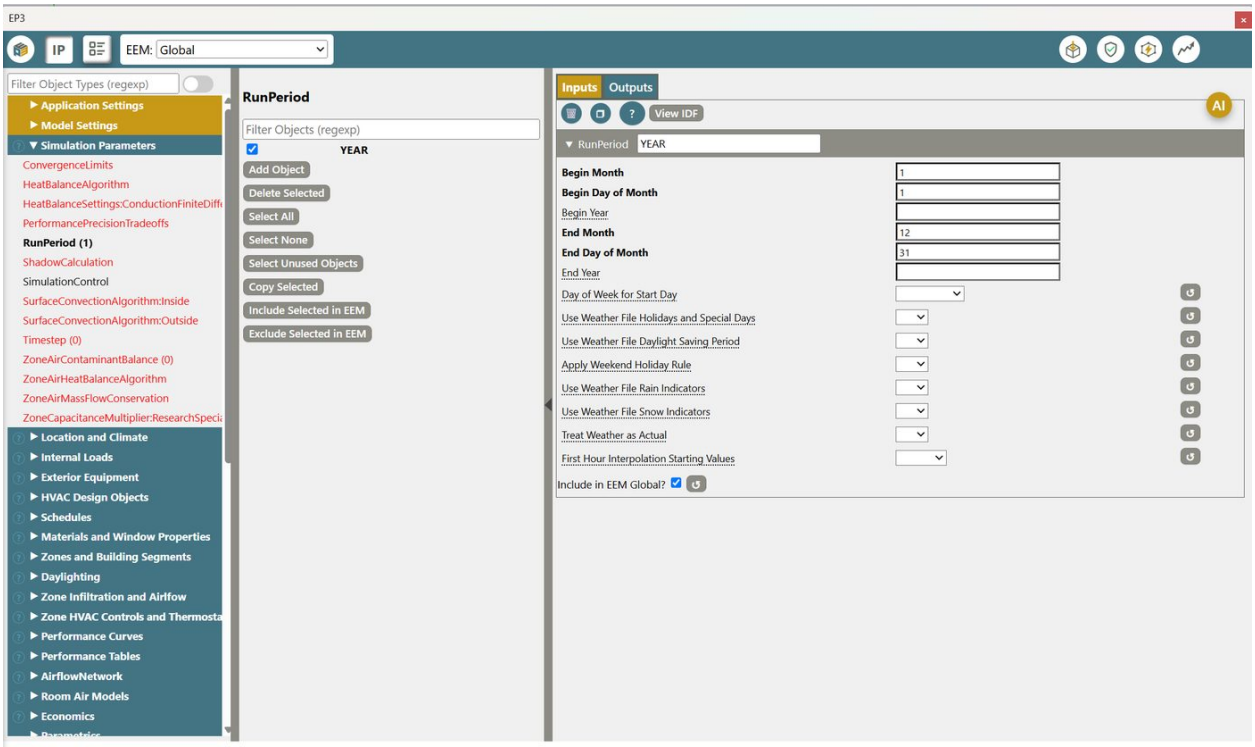


Figure 141: The Simulation Parameters group expanded in Column 1: SimulationControl and RunPeriod appear in black, since this model has an instance of each; every other object type in the group appears in red, since none are included in this model.

Field	Default	Notes
Number of Timesteps per Hour	6	Must divide evenly into 60. EnergyPlus suggests a minimum of 20 for `ConductionFiniteDifference/CombinedHeat` surface heat balance algorithms, and 12 or higher when a `Material:RoofVegetation` is present.
Minimum System Timestep	—	In minutes; 0 sets it to the zone timestep.
Maximum HVAC Iterations	20	
Minimum Plant Iterations	2	Higher values increase runtime but can improve accuracy on complex plants.
Maximum Plant Iterations	8	Lower values can reduce runtime at the cost of accuracy on complex plants.

2.4 Other Simulation Parameters

The Simulation Parameters group also holds a set of objects that control simulation physics and numerical precision rather than what gets run: `ShadowCalculation`, `HeatBalanceAlgorithm`, `SurfaceConvectionAlgorithm:Inside/:Outside`, `ZoneAirHeatBalanceAlgorithm`, `ZoneAirContaminantBalanc` `ZoneAirMassFlowConservation`, `ZoneCapacitanceMultiplier:ResearchSpecial`, `HeatBalanceSettings:Condu` `PerformancePrecisionTradeoffs`, and `HVACSystemRootFindingAlgorithm`. Each is reachable in the same Column 1 group and browsable the same way as the objects above.

Field-by-field coverage of this group is out of scope for this chapter. For most models the defaults are appropriate; consult the EnergyPlus Input/Output Reference for guidance on when to change them. ShadowCalculation HeatBalanceAlgorithm

---

3. The Simulate/Export Dialog

Clicking **Simulate Model** on the main toolbar opens the **EP3 - Simulate / Export** dialog.

The dialog lists every EEM in the model available for simulation. A branching EEM's row is labeled with the number of parameter combinations it expands into, for example "LPD (6 runs)." Checking a box selects that EEM (and, for a branching EEM, every combination) to include in the batch. **Select All** and **Select None** apply to the whole list at once.

Four actions are available, depending on what's enabled for the model:

- **Simulate** — writes an IDF for each checked EEM and launches EnergyPlus on it.
- **Export IDF Only** — writes the IDF files without running EnergyPlus.
- **Simulate Existing Files** — bypasses EEM selection entirely and re-runs a single, previously exported `.idf` file you pick from disk, using the model's default Weather File (§7).

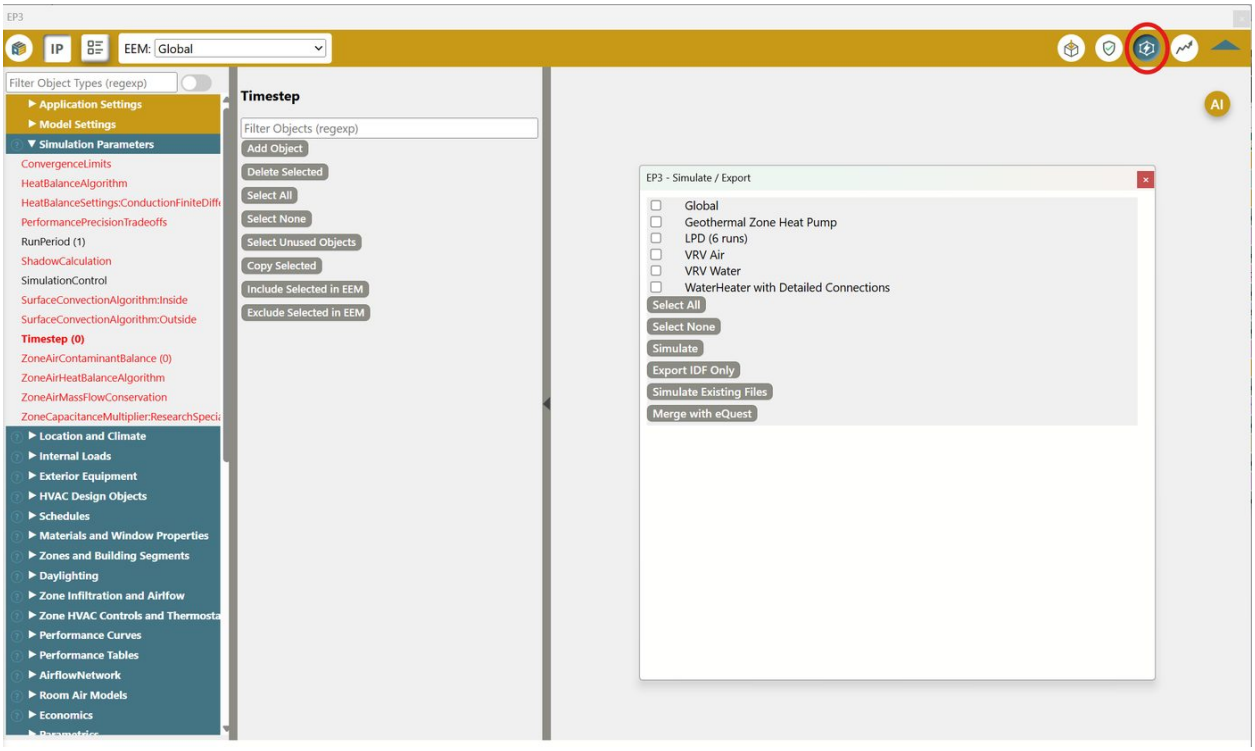


Figure 142: The EP3 - Simulate / Export dialog, listing Global, Geothermal Zone Heat Pump, LPD (6 runs), VRV Air, VRV Water, and WaterHeater with Detailed Connections as checkbox rows, with Select All, Select None, Simulate, Export IDF Only, Simulate Existing Files, and Merge with eQuest below.

- **Merge with eQuest** — available only when eQuest merge is enabled for the installation; combines the checked EEMs into an eQuest INP file rather than an EnergyPlus run. Out of scope for this manual.

Before writing any files, EP3 validates the model and prompts for a save location if one isn't already set for the current phase. *(see "Model Troubleshooting and Quality Assurance")* covers what model validation checks.

---

4. Running Multiple EEMs at Once

Checking more than one EEM (or a mix of ordinary and branching EEMs) runs the whole set in one batch from a single click of **Simulate**. EnergyPlus processes for the batch run concurrently, up to roughly 75% of the machine's CPU cores at a time — each EnergyPlus process occupies one core for the duration of its run, so the throttle keeps a large batch from oversubscribing the machine. Runs beyond that limit queue and start automatically as earlier ones finish; nothing further is required from you once the batch is launched.

If a batch would produce more than eight IDF files, EP3 shows a confirmation before starting, stating the file count and the machine's core count.

A branching EEM's combinations are written as separate IDF files. Rather than a readable filename, each combination file is named from a hash of its specific chain of settings; the readable name (the EEM chain, joined with `->`) is shown in progress messages and in the Results Viewer instead of the filename itself.

---

5. What Happens During a Run

For each checked EEM (or combination), in order:

1. EP3 resolves a weather file: an EEM can carry a weather-file override; if it doesn't, the model's default Weather File (*(see "The EP3 Interface and Model Settings", §8.3)*) is used. If neither is set, Simulate cannot proceed.
2. EP3 switches the model to that EEM, updates HVAC connections for it, and writes the IDF file. EP3 also configures the model's `Output:JSON` object to produce JSON results (with tabular data in kWh) regardless of what that object currently has stored, so a simulated run always has JSON output available for the Results Viewer.
3. EnergyPlus is launched against that IDF in a temporary output subdirectory, with a visible console window for the duration of the run. IDF preprocessing (equivalent to running EnergyPlus's ExpandObjects utility, needed for objects like `HVACTemplate:*` and compact-schedule macros) and the readvars postprocessing step (which produces the CSV outputs alongside JSON) both run automatically as part of the same launch.
4. A status overlay in the lower corner of the SketchUp viewport names the current EEM and step for the duration of the run (for example, "VRV Air: Simulating").

---

6. Completion, Output Files, and Cleanup

6.1 Determining success or failure

Each run's outcome is determined by EnergyPlus's process exit code:

- **Exit code 0** — reported as Completed.
- **Nonzero exit code** — reported as Failed, with the tail of the run's log file included in the status text.
- **Process did not exit cleanly** — reported as Crashed.

A finer-grained breakdown of Fatal/Severe/Warning counts from each run's `.err` file appears later, in the Results Viewer. *(see "Model Troubleshooting and Quality Assurance")*

6.2 Output files

Each run's native EnergyPlus output files are renamed out of its temporary subdirectory into the model's output folder, using the run's filename as a prefix:

EnergyPlus output	EP3 output
`eplustbl.htm`	`{name}Table.html`
`eplusmtr.csv`	`{name}Meter.csv`
`eplusssz.csv`	`{name}Ssz.csv`
`epluszsz.csv`	`{name}Zsz.csv`
`eplusout.err`, `.eio`, `.bnd`, etc.	`{name}.err`, `{name}.eio`, `{name}.bnd`, etc.
the summary JSON	`{name}Output.json`
hourly/monthly/etc. JSON variants	`{name}Output_{suffix}.json`

Before writing a new run over an existing file, EP3 deletes the previous run's output artifacts for that filename — audit, BND, CSV, EIO, END, ERR, ESO, MDD, MTD, RVAUDIT, SHD files, the output JSON files, and the tabular HTML/CSV/Zsz files. The `.rdd` file (available output variables) is deliberately left in place across re-runs, since it remains useful for troubleshooting even after a new run starts.

6.3 End-of-batch summary

Once every run in the batch has finished, a single **Simulation Status** dialog lists Completed/Failed/Crashed for each file, and the Results Viewer opens automatically if at least one run completed. *(see "Viewing and Interpreting Results")*

---

7. Export IDF Only and Simulate Existing Files

Export IDF Only runs the same batch process as Simulate — EEM selection, weather resolution, model validation — but stops after writing the IDF files, without launching EnergyPlus. Use it to produce IDF files for manual editing, for use outside EP3, or to stage a batch to run later with **Simulate Existing Files**.

Simulate Existing Files re-runs a single, already-exported `.idf` file you pick from disk, using the model's default Weather File (*(see "The EP3 Interface and Model Settings", §8.3)*) regardless of what weather file the IDF was originally exported with.

26. Viewing and Interpreting Results

1. Overview

EP3 offers three separate ways to look at a completed simulation, matched to three different questions:

- The **Results dialog** answers "what were the annual or summary numbers" — EnergyPlus's tabular reports, with charts, side-by-side comparison across EEMs, and a built-in EUI table.
- The **Output Variables / Timestep Viewer** answers "how did this value change over time" — an hourly or sub-hourly line chart of any output variable, across one or more result files.
- The **Data Visualization** heat map answers "where in the building did this happen" — the same time-series data, painted directly onto building surfaces and zones in the 3D view.

All three read from files EnergyPlus writes after a simulation completes; running the simulation is covered in *(see "Running Simulations")*. This chapter assumes at least one EEM has already been simulated.

---

2. Choosing which results to view

The Results dialog opens from the **View Results** icon on the Main Dialog's Top Control Strip (*(see "The EP3 Interface and Model Settings", §2.1)*).

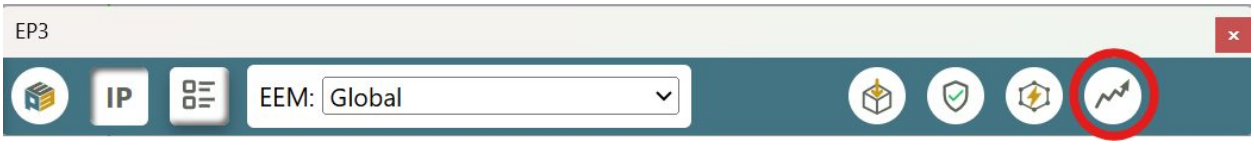


Figure 143: The View Results icon on the Top Control Strip opens the Results dialog.

It opens showing whichever result files it last displayed. **Select Files To Display** switches that set: it scans a chosen directory, including its subfolders, for `.idf` files with matching output, and lets you pick which ones populate the dialog. This is also how results from a different model are compared against the current one — point it at that model's output folder instead.

Within that comparison set, an EEM's tabular data is read from its `Output.json` file (EnergyPlus's JSON-format tabular report output). A file with no matching `Output.json` shows no report content, even if its `.idf`/`.err` are still reachable from the file-status panel described in §8.

---

3. Getting a big-picture summary

3.1 Default reports and Select Reports To Display

Opening the Results dialog shows a curated set of tables rather than every report EnergyPlus produces: General (Input Verification and Results Summary), End Uses (Annual Building Utility Performance Summary), Time Setpoint Not Met (System Summary), every table in Component Sizing Summary, and three LEED Summary tables — Advisory Messages, Energy Cost Summary, and Performance Rating Method Compliance.

Select Reports To Display expands that set. It lists every report/table pair present across the currently loaded result files, grouped by report, with a checkbox per table. **Update** applies the change; **Reset To Default** returns to the curated set above. The selection is saved per model, not per session.

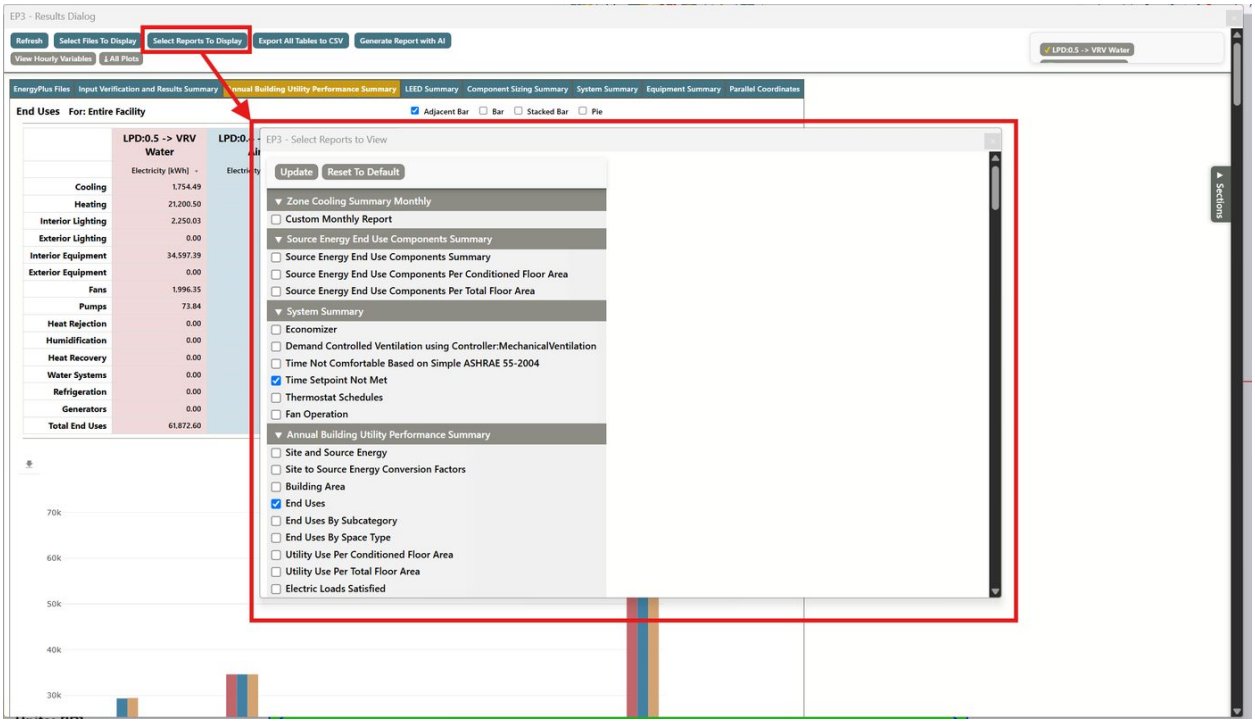


Figure 144: Select Reports To Display, with the System Summary and Annual Building Utility Performance Summary groups expanded — Time Setpoint Not Met and End Uses are checked, matching the default set in §3.1.

3.2 Charts, column visibility, and units

Every table in the Results dialog can render as a chart — Adjacent Bar, Bar, Stacked Bar, or Pie — with the chart type chosen per table and remembered afterward. Individual data columns can be hidden per table, a saved preference rather than a one-time filter, which keeps a wide comparison table readable when only a few of many EEMs matter for a given question.

Tables display in whichever unit system the model is currently set to display, SI or IP; switching that setting converts both the column headers (e.g. "Electricity [kWh]" becomes the IP equivalent where one applies) and the values shown. Electricity usage and demand stay in kWh/kW even in an IP-displayed model, matching how utility bills are read; other quantities convert to their usual IP units.

A collapsible **Sections** drawer at the right edge of each report tab jumps between that report's tables, useful once several are showing on the same tab.

3.3 Getting an EUI

Annual Building Utility Performance Summary carries a **Utility Use Per Conditioned Floor Area** table with site and source EUI already computed by EnergyPlus, broken out by fuel type. It

is not part of the default table set (§3.1) — add it through Select Reports To Display.

Annual Building Utility Performance Summary documents exactly how this table's EUI figures are computed.

3.4 Exporting results

Export All Tables to CSV downloads every currently-displayed table as CSV. **Export All Plots** saves every currently-rendered chart, and every currently-displayed table rendered as an image, as PNG files to a folder you choose; each individual chart also carries a save-image icon for exporting one chart at a time. **Generate Report with AI** produces an AI-written summary of the currently loaded results — see *(see "The EP3 AI Assistant")* and *(see "Model Documentation and Reporting")* for building a fuller report package around any of these.

---

4. Digging into a specific tabular report

A few more reports give equipment- and sizing-level detail. None of these appear by default (§3.1) — add them through Select Reports To Display.

Report	What it covers
Equipment Summary	Central plant and zone equipment nameplate data — capacities, efficiencies, rated flow — one row per piece of equipment.
Coil Sizing Details	Coil-by-coil design conditions and capacities, including the entering/leaving air conditions EnergyPlus assumed.
HVAC Sizing Summary	Zone- and system-level design airflow and load results in one place.
Standard 62.1 Summary	Ventilation rate procedure results, when Standard 62.1 reporting is enabled.
LEED Summary (additional tables)	LEED-specific compliance and cost tables beyond the three shown by default.

Each report tab behaves the same way described in §3.2 — chartable, column-hideable tables — so a plant-performance question such as which chiller carries the most load, or which pump draws the most power, is usually a matter of adding the right report through Select Reports To Display rather than reaching for the per-timestep detail in §6.

---

5. Getting the data you'll need before you simulate

The Results dialog's tabular reports come from EnergyPlus's summary and sizing reporting and need no special setup. The Output Variables/Timestep Viewer (§6) and Data Visualization heat map (§7) both read `Output:Variable` time-series data instead, and that data exists only for variables EnergyPlus was asked to report.

EP3 gets `Output:Variable` requests into a model four ways.

5.1 Built-in defaults

Some variables are requested automatically, with no setup needed:

- Every `Zone` defaults to reporting Zone Mean Air Temperature and both thermostat setpoints (heating and cooling).
- The building/site object defaults to reporting Site Outdoor Air Drybulb and Wetbulb Temperature, and Facility Total Purchased Electricity Rate.
- `AvailabilityManager:NightCycle` defaults to reporting its control status.
- Setpoint-managed nodes get a `NodeReport` object placed automatically at that node's location, covering temperature, flow rate, and setpoint temperature.

Any of these can be turned off the same way a variable is turned on elsewhere in this section — through the object's Outputs tab (§5.2), or, for node data, the `NodeReport` object itself. Automatic `NodeReport` placement at a setpoint-managed node follows the same pattern as the setpoint manager itself, covered in *(see "Air Loops, Branches, and Nodes").*

5.2 An object's Outputs tab

Most object dialogs carry an **Outputs** tab: a Reporting Frequency selector and a checkbox per variable that object type can report, with an optional EMS Sensor Name field next to each one for tying an EMS sensor to that same variable (EnergyManagementSystem:Sensor). A **View Output Variables IDF** button previews the exact `Output:Variable` lines that checked state will export.

> *Figure placeholder — an object's Outputs tab, showing the Reporting Frequency selector, a few checked/unchecked variable rows, and one EMS Sensor Name field filled in.*

5.3 View Building Data, for the heat map specifically

The Building Geometry Outliner's **View Building Data** button opens a picker scoped to the variables the Data Visualization heat map (§7) can display — every potential `BuildingSurface:Detailed`, `FenestrationSurface:Detailed`, `Zone`, and `Space` variable, filterable by name, with a toggle to show only variables that already have results. Checking a variable here has the same effect as checking it on that object type's Outputs tab: it requests the variable for the next simulation. A variable left unchecked before simulating will not appear as an option in the heat map afterward, regardless of how relevant it turns out to be.

5.4 A loose `Output:Variable` object

`Output:Variable` (Output:Variable) also exists as an ordinary object — Key Value, Variable Name, and Reporting Frequency — addable from the generic object list like any other EnergyPlus object. This is the direct EnergyPlus-native path, useful for a variable name copied straight from the Input/Output Reference or an `.rdd` file.

Importing an IDF whose `Output:Variable` targets a system node converts it into a `NodeReport` object at that node's location instead of staying a loose object. Both represent the same underlying request; `NodeReport` is how EP3 represents it once the node's location is resolved.

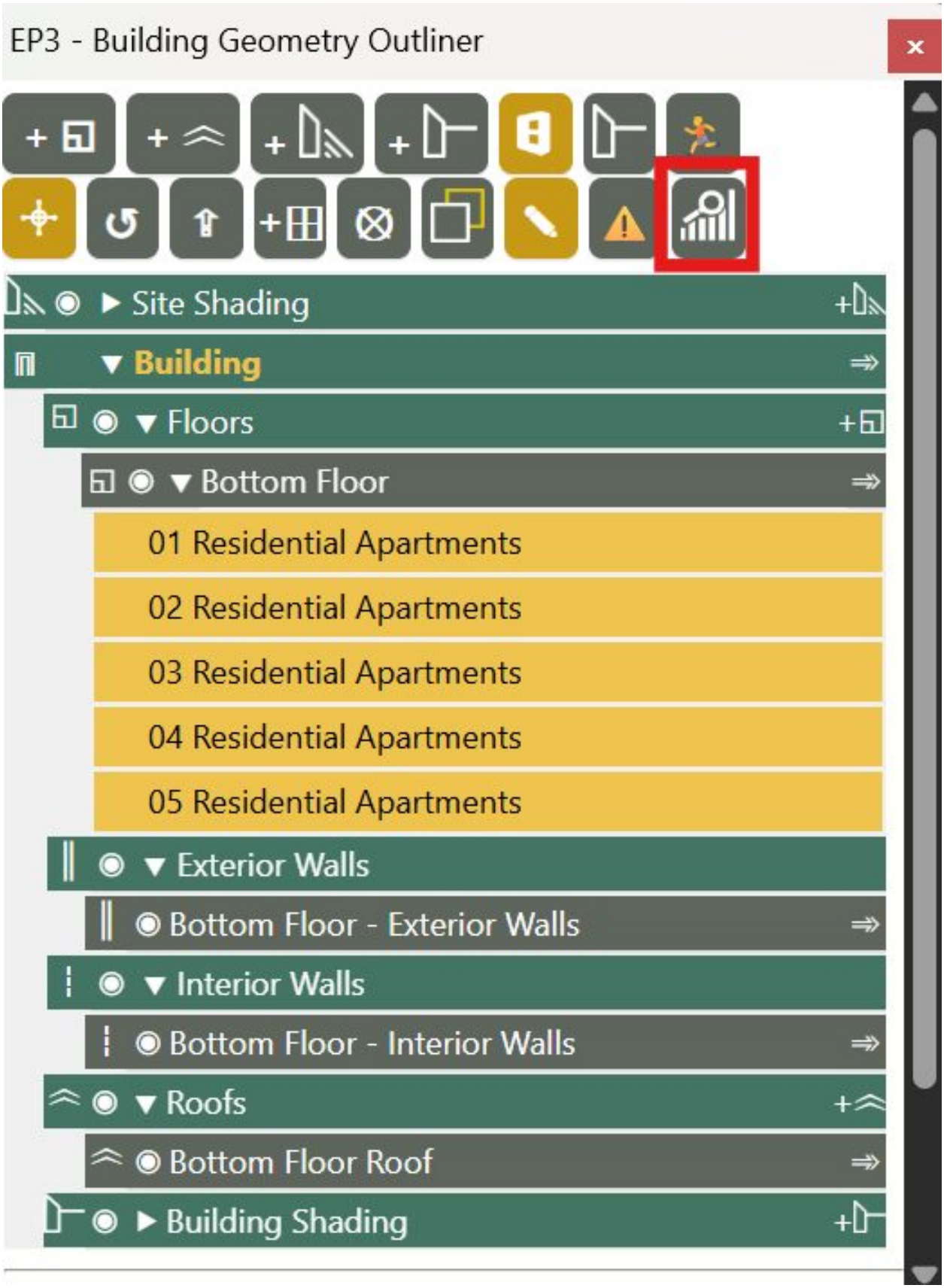


Figure 145: The Building Geometry Outliner's View Building Data icon opens the picker for requesting variables and, once results exist, the Data Visualization heat map.

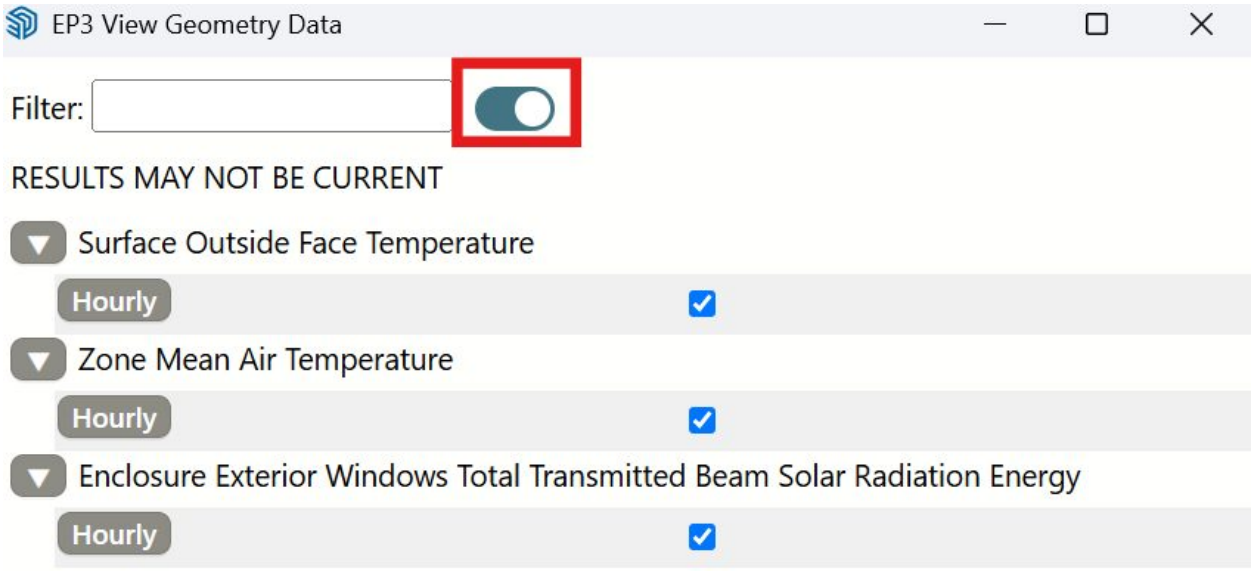


Figure 146: View Geometry Data, with the "only show variables with results" toggle highlighted, a Filter field, and three checked variables each with Hourly results available. The banner appears when the model has changed since the loaded results were generated.

6. Seeing how a value changes over time

6.1 Opening the Timestep Viewer

Whenever simulated files include Hourly or Timestep output, the Results dialog shows a **View Hourly Variables / View Timestep Variables** button for each frequency present. Opening one launches a chart across every currently-loaded result file that shares the same simulated date range as the first. **Export to CSV** and **Export Images** save the chart's underlying data and a snapshot image, independent of the Results dialog's export buttons (§3.4).

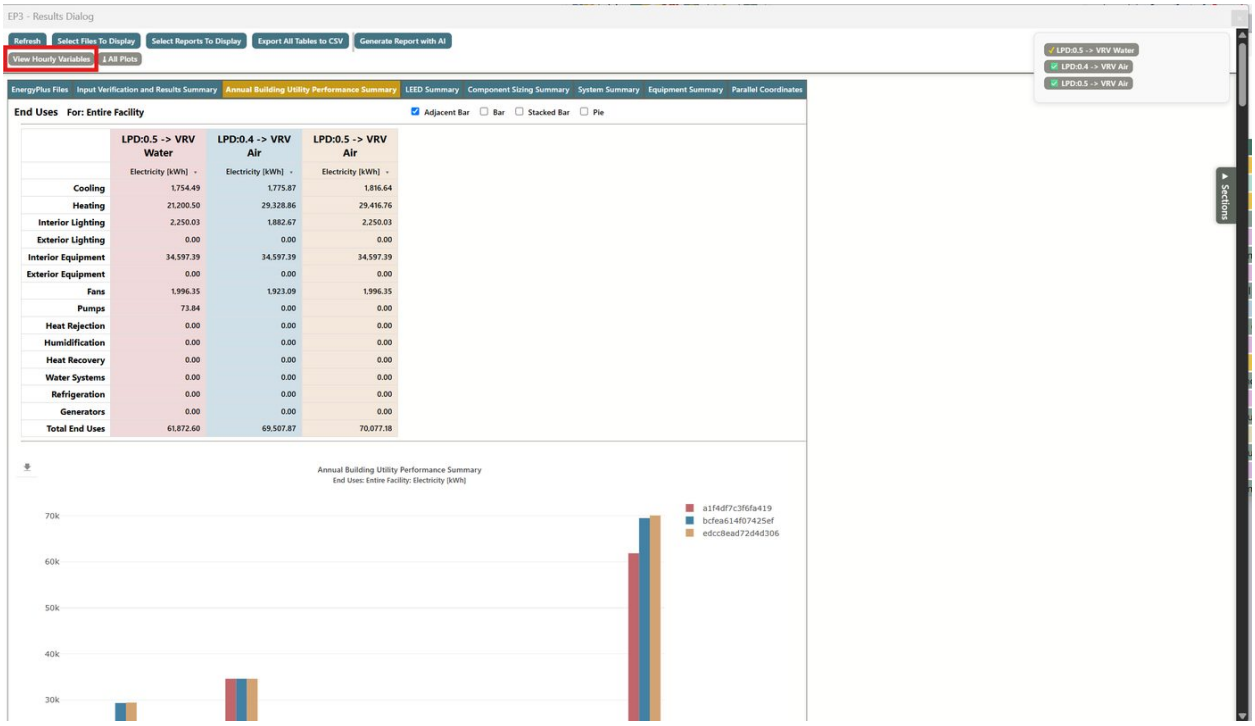


Figure 147: The Results dialog, with View Hourly Variables highlighted — opens the Timestep Viewer for every currently loaded result file with Hourly output.

6.2 Picking and organizing variables

A **Select Variables** panel lists every variable available across the loaded files, filterable by name, grouped under a collapsible header per loaded file, with **Select All/Deselect All** to toggle the whole list at once. A checked variable can be:

- **Aliased** — given a short display name, in an "alias (optional)" field directly beneath it, in place of its full EnergyPlus variable name.
- Assigned a **Sum/Average/Minimum/Maximum** override for aggregation (§6.3), in place of whichever convention EnergyPlus's reporting would otherwise use for that variable.

A **Calculated Variables** section offers several blank rows, each defining one custom variable from a Name, a Formula (e.g. `VarA + VarB`) referencing other variables by their alias, and Units. A calculated variable charts and aggregates like a native EnergyPlus variable, except it carries no EnergyPlus Sum/Average convention — it defaults to Sum unless given an explicit override.

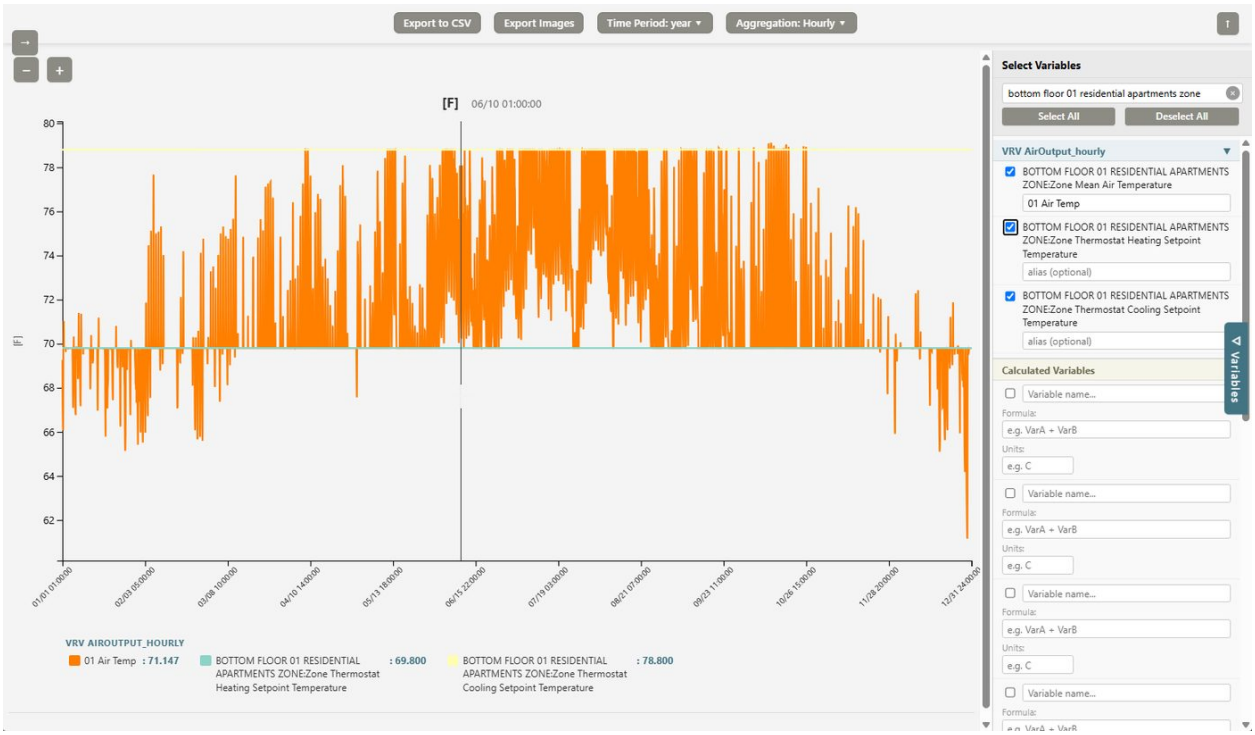


Figure 148: The Timestep Viewer, filtered to one zone, with three variables checked (two aliased) and the Calculated Variables section below.

6.3 Date range and aggregation

A **Time Period** control narrows the chart to a specific run period, sizing period, or design day, matching the ranges the model simulated. An **Aggregation** control rolls the chart up from its native resolution (Hourly or Timestep) to Daily, Monthly, or Annual; each variable aggregates by Sum or Average according to EnergyPlus's reporting convention for that variable, unless overridden per-variable as described in §6.2.

6.4 Node data and the node state overlay

When exactly one file is loaded, a separate **Node states to display in HVAC diagram** panel lists every system-node variable available in that file, with a filter field and **Select All**/**Select None**, independent of the Select Variables panel and its checked/charted state (§6.2). Each checked node variable overlays as a text label at that node's location in the 3D view, updating live as the chart's time slider moves — a quick way to see, for example, an entering coil temperature and its setpoint side by side without adding either to the chart itself.

This is one way of reaching node-level data; the same information is also available directly, without a chart, through the standalone Node Report troubleshooting tool — see *(see "Model Troubleshooting and Quality Assurance")*.



Figure 149: The Time Period control open on an Hourly chart, showing the full simulated year checked alongside the extreme summer and winter design-day periods available to switch to.

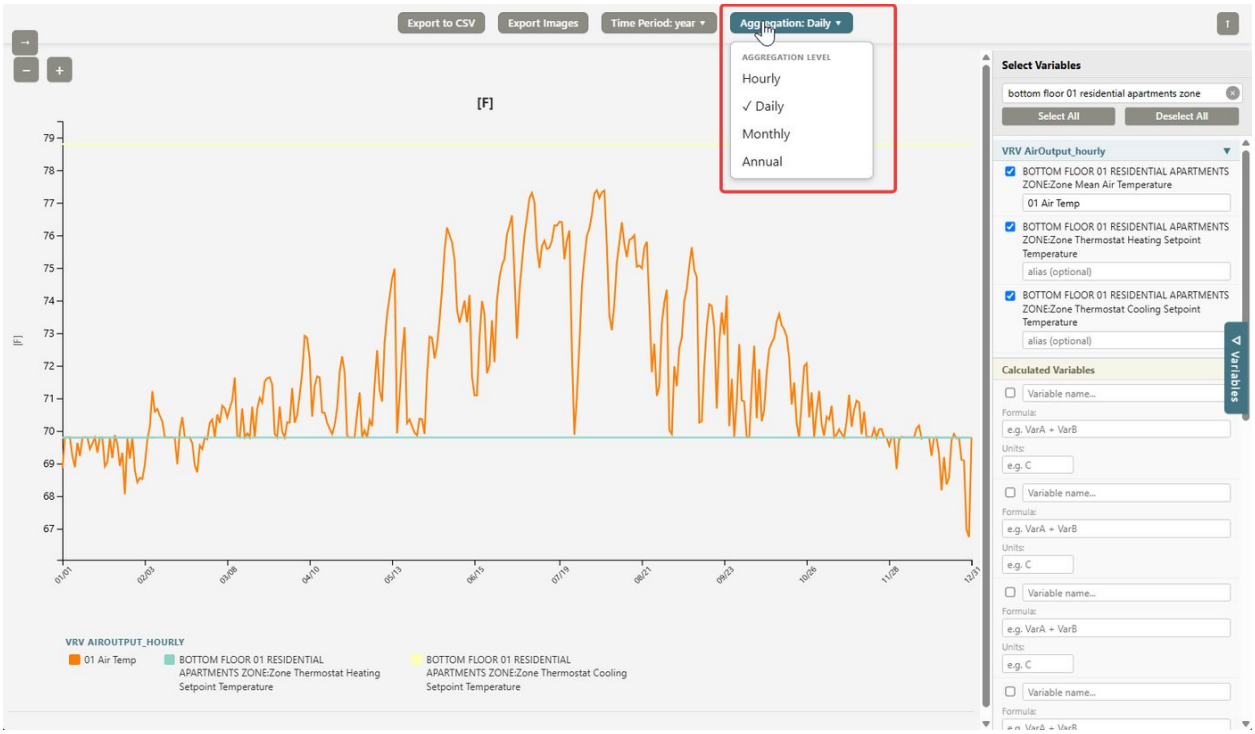


Figure 150: The Aggregation control open, switching from Hourly to Daily — the chart's native hourly resolution rolls up to one point per day.

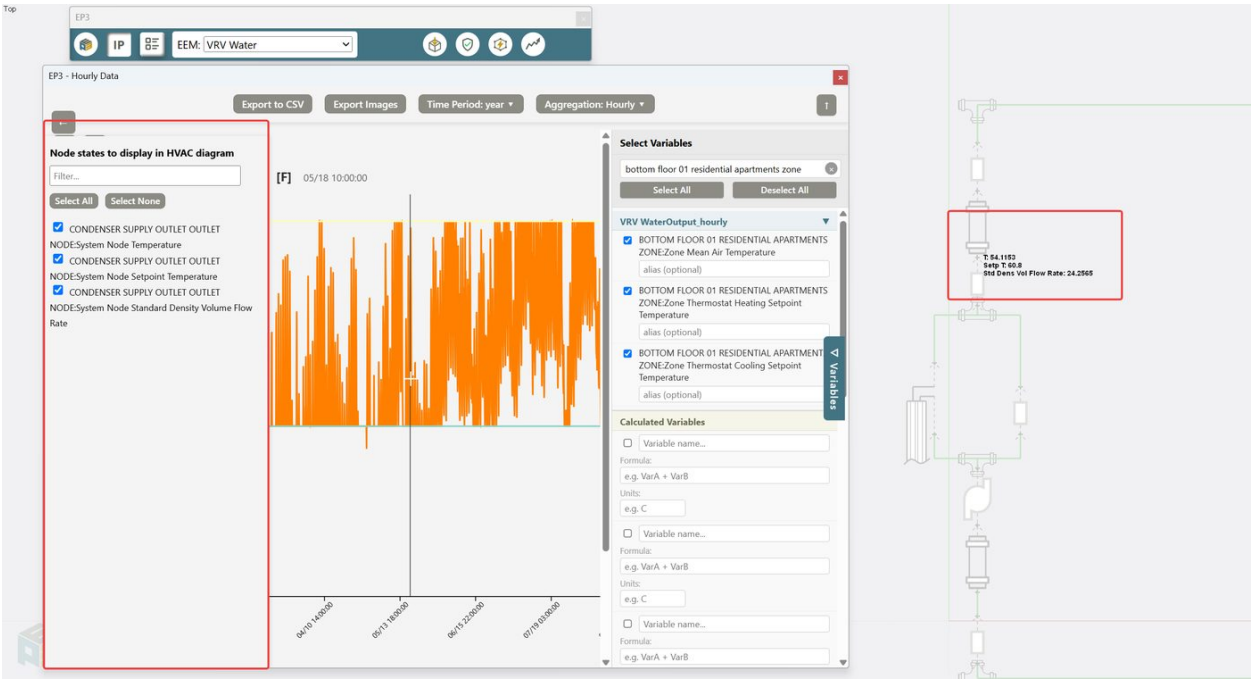


Figure 151: The Node states to display in HVAC diagram panel (left) and its live overlay on one piece of equipment in the 3D view (right), reading temperature, setpoint temperature, and flow rate at the scrubbed time.

7. Seeing where in the building a value occurs

7.1 Opening Data Visualization

The Building Geometry Outliner's **View Building Data** button — the same entry point used to request variables before simulation, §5.3 — opens onto whichever variables already have simulated results, each with a **View** button per available reporting frequency. Selecting one colors every relevant surface, zone, or space directly in the 3D view according to that variable's value, with a legend showing the color scale's current minimum and maximum.

Results shown here belong to whichever EEM is currently active — set the EEM dropdown to the EEM you simulated before opening View Building Data, not merely the EEM you want compared in the Results dialog's file list (§2). A banner warns when the model has changed since those results were generated.

7.2 Time, aggregation, and normalization

A time scrubber moves through the simulated period, recoloring the model as the value changes. **Time Period** and **Aggregation** work the same way described in §6.3 — narrowing to a run period or design day, or rolling up to a coarser resolution, with the same per-variable Sum/Average convention.

A **Normalize by Area** toggle divides each surface, zone, or space's value by its area before coloring and scaling the legend — useful when a variable is only meaningful per unit area, such as comparing heat gain across zones of different sizes.

Site-level variables that do not tie to a specific surface, such as outdoor air conditions, display as

text alongside the model rather than as a color, since there is no single surface to paint them onto.

Figure placeholder — the 3D view with a heat map applied to a multi-zone floor, showing the color legend, the time scrubber, and the Normalize by Area toggle.

---

8. Raw file artifacts

8.1 The EnergyPlus Files tab

The **EnergyPlus Files** tab lists, for each loaded file, a row of buttons — **Errors** (`.err`), **IDF** (`.idf`), **Tables** (`Table.html`), **RDD** (`.rdd`), **Output Variables** (`.csv`), **ESO** (`.eso`) — each opening that raw file in whatever application handles it on the system, for a reader who wants EnergyPlus's output directly rather than through EP3's rendering of it. The `.rdd` is the authoritative list of every variable available for that run — useful when a variable expected in the Timestep Viewer or Data Visualization picker does not appear.

8.2 The file status panel and AI-assisted error resolution

Separately from the EnergyPlus Files tab, the list of simulated files shown alongside the Results dialog carries one status button per file: a green checkmark, a gold checkmark for severe-but-non-fatal errors, or a red X for a fatal error, with fatal/severe/warning counts in its tooltip. Clicking that button opens the file's `.err` content.

When EP3's AI assistant is available (a valid API key configured — *(see "The EP3 AI Assistant")*), opening an error file this way starts an AI conversation seeded with the error text, and switches the model to the EEM that produced it, so the assistant reads and suggests changes against the correct EEM rather than whichever one happens to be active. If the user switches to a different EEM while the assistant is working, it pauses rather than acting on the wrong EEM, and resumes once the original EEM is active again. Without a configured AI assistant, the same button opens the error text in a plain viewer instead.

Detailed interpretation of `.err` content, and unmet-hours diagnosis generally, is covered in *(see "Model Troubleshooting and Quality Assurance")*.

---

9. Where to Look Next

- *(see "The EP3 Interface and Model Settings", §2.1)* — the Top Control Strip this chapter's View Results icon lives on.
- *(see "Running Simulations")* — getting a model to the point this chapter starts from.
- *(see "Air Loops, Branches, and Nodes")* — automatic `NodeReport` placement at setpoint-managed nodes.
- *(see "Energy Efficiency Measures (EEMs)")* — the Parallel Coordinates comparison tab, a different way of comparing EEMs than the side-by-side tables in this chapter.
- *(see "Model Troubleshooting and Quality Assurance")* — unmet hours, `.err` interpretation in depth, and the standalone Node Report tool.
- *(see "Utility Costs")* — the Utility Cost Manager, for cost rather than energy/demand results.
- *(see "Model Documentation and Reporting")* — building a report package around exported tables and images.

- *(see "The EP3 AI Assistant")* — the Generate Report with AI feature in more depth.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF and EXTLINK placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

27. Utility Costs

1. Overview

Utility costs are modeled with `UtilityCost:Tariff` objects, along with the charge, eligibility, ratchet, and variable objects that attach to a tariff. These are found under the **Economics** group in Column 1 of the Main Dialog.

The Model Wizard's Utility Rates tab (*(see "The Model Wizard", §8)*) builds the same `UtilityCost:Tariff` object through a nine-field shortcut: a monthly service charge and a flat per-unit energy and demand rate, for each of Electricity, Natural Gas, and Water. That shortcut covers a flat rate with no time-of-use periods, no tiered pricing, and no charges beyond the three it asks for. This chapter covers the full tariff, including everything the Wizard's shortcut does not build.

---

2. Currency

One **Currency Type** object, listed under Economics in Column 1 of the Main Dialog, sets the **Monetary Unit** for every tariff in the model — a three-letter ISO 4217 currency code (USD, EUR, GBP, and so on).

Leaving Monetary Unit blank defaults to USD. Common codes (USD, EUR, GBP, JPY, and similar) display with their usual symbol wherever a rate or charge appears; other codes display as the code itself, for example "PHP 12.50."

---

3. Adding a Tariff

Clicking **Add Object** on the `UtilityCost:Tariff` type in Column 1 opens a form:

- **Meter** — Electricity, Natural Gas, Water, or Other. Choosing Other reveals a **Custom Meter Name** field, for billing against a sub-meter or another named `Output:Meter`.
- **Tariff Name** — leave blank for a default such as "Electricity Tariff."
- **Group Name** — optional; covered in §4.

If a tariff already exists on the meter selected, the form shows a warning describing the conflict and how to resolve it (§4).

Once added, the tariff appears in Column 2 labeled with its meter, and opens directly in Column 3. Column 3 opens with a field table for values that apply to the tariff as a whole:

- **Conversion Factor Choice** — the units charges are expressed in (kWh, Therm, MJ, gal, m^3, and so on). Defaults follow the meter and the model's unit system (IP or SI) if left blank — for example, Natural Gas defaults to Therm in IP and MJ in SI.
- **Demand Window Length** — the averaging window a peak demand charge is measured over (QuarterHour, HalfHour, FullHour, Day, or Week). Set this to match the utility's demand metering convention.
- **Monthly Charge or Variable Name** and **Minimum Monthly Charge or Variable Name** — a flat monthly service charge, and a minimum total bill, if either applies to this account.

Below the field table are Energy Charges, Demand Charges, Other Charges, and an Advanced section, each covered in the rest of this chapter.

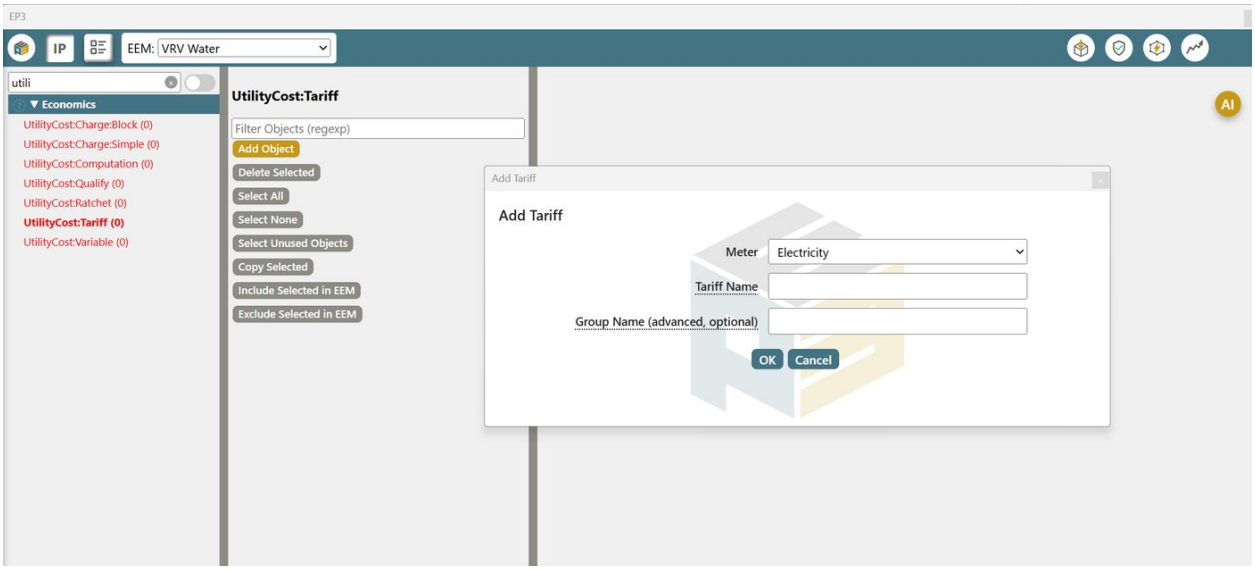


Figure 152: The Add Tariff form, with Meter set to Electricity and Tariff Name and Group Name left blank.

---

4. Comparing or Combining Multiple Tariffs on the Same Meter

EnergyPlus bills only one tariff per meter each month: if two tariffs both reference the same meter, it uses whichever produces the lower cost that month and drops the other entirely. Column 3 of the Main Dialog shows a banner on a tariff whenever this applies, and Column 2 marks the same tariffs with a badge.

Two situations produce this banner, with different implications:

- **No Group Name set on either tariff.** This usually happens when two charges meant to combine into one bill — for example, a distribution charge alongside an energy charge — get built as separate tariffs. Move the second tariff's charges onto the first as additional charges.
- **Both tariffs share the same Group Name.** EnergyPlus selects the cheaper of two genuine alternate rate plans automatically, within a single simulation run.

For most rate comparisons, leave Group Name blank on every tariff and include or exclude each tariff per Energy Efficiency Measure, comparing cost across separate simulation runs. *(see "Energy Efficiency Measures (EEMs)")* covers EEM inclusion in full. Reserve Group Name for cases where the cheaper-of-two selection should happen automatically within one run.

---

5. Modeling a Flat Rate

A flat energy or demand rate is one number that applies to all usage, all the time. Under **Energy Charges** or **Demand Charges** in a tariff's dialog, clicking **+ Add Flat Rate** opens a form:

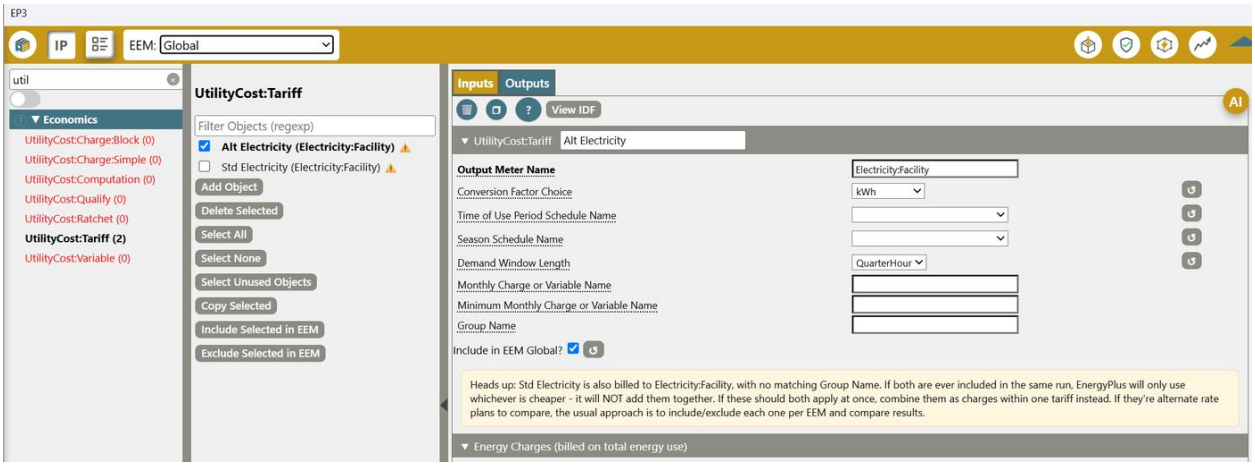


Figure 153: The warning banner on "Alt Electricity," reading that "Std Electricity" is also billed to Electricity:Facility with no matching Group Name.

- **Charge Name** — leave blank for an auto-generated name.
- **Rate** — the price per unit (per kWh, per Therm, per kW, and so on, matching the tariff's Conversion Factor Choice).
- **Billing Period** and **Season** — shown only if the tariff has a Time of Use Period Schedule Name or Season Schedule Name set (§6). With neither set, a flat rate applies to all hours of every month.

Each added charge appears as a row, with its rate and an **Edit**/**Remove** button, under the section it was added to.

Worked example. The rest of this chapter builds one tariff for a mid-size commercial electric account, adding to it as each mechanism is introduced. Start with the account's flat monthly service charge, entered directly on the Tariff's **Monthly Charge or Variable Name** field (§3):

> Monthly Charge or Variable Name: **$25.00**

---

6. Time-of-Use and Seasonal Charges

Two schedules on the tariff, set once, unlock additional options on every charge form afterward:

- **Time of Use Period Schedule Name** — a schedule whose values mark each hour as 1 (Peak), 2 (Shoulder), 3 (Off-Peak), or 4 (Mid-Peak). Once set, every Add Flat Rate and Add Tiered/Block Rate form gains a **Billing Period** dropdown, with entries for each period individually plus combined options such as "Peak + Shoulder," and, for demand charges, comparison options such as "Peak Exceeds Off-Peak."
- **Season Schedule Name** — a schedule whose values mark each month as 1 (Winter), 2 (Spring), 3 (Summer), or 4 (Autumn). Once set, every charge form gains a **Season** dropdown.

Both schedules are ordinary `Schedule:Compact` (or similar) objects, built the same way as any other schedule in the model — *(see "Schedules")* covers building one. Scoping by billing period and scoping by season are independent: a charge can use either, both, or neither. A charge with no

Season selected applies to all months; a charge with no Billing Period selected applies to all hours within whatever months it does apply to.

Worked example, continued. With a Time of Use Period Schedule Name and Season Schedule Name both set on the tariff, add three flat energy charges:

Season	Billing Period	Rate
Summer	Peak	$0.16/kWh
Summer	Off-Peak	$0.09/kWh
Winter	All Hours	$0.10/kWh

Figure placeholder — the Add Flat Rate form with Season set to Summer and Billing Period set to Peak

---

7. Modeling a Tiered or Block Rate

A tiered (block) rate charges different rates for different slices of usage — for example, a lower rate for the first block of demand and a higher rate for everything past it. Clicking **+ Add Tiered / Block Rate** under Energy Charges or Demand Charges opens a form with the same Charge Name, Billing Period, and Season fields as a flat rate, plus:

- **Number of Tiers** — from 1 to 15.
- For each tier: a **size** (a number, or the word "remaining" for the last tier) and a **rate** per unit within that tier.

Editing a charge down to fewer tiers clears the values in the tiers no longer used.

Worked example, continued. Add a tiered demand charge, with no Season or Billing Period selected (applies to all months, all hours):

Tier	Size	Rate
1	50 kW	$9.00/kW
2	remaining	$13.50/kW

Figure placeholder — the Add Tiered / Block Rate form with Number of Tiers set to 2, Tier 1 size 50 and rate 9.00, Tier 2 size "remaining" and rate 13.50

---

8. Other Charges: Service Fees, Taxes, Surcharges, and Adjustments

Beyond energy and demand, a tariff computes its total through a fixed sequence of running subtotals:

1. **Basis** = Energy Charges + Demand Charges + Service Charges
2. **Subtotal** = Basis + Adjustments + Surcharges
3. **Total** = Subtotal + Taxes

UtilityCost:Tariff documents this calculation in full.

Clicking **+ Add Custom Charge** under Other Charges opens a form for any charge in the Service Charge, Tax, Surcharge, or Adjustment categories:

- **Charge Name** — leave blank for a name generated from the type.
- **Type** — Service Charge, Tax, Surcharge, or Adjustment.
- **Based On** — what the charge is calculated from: Total Energy Use, Total Demand, the time-of-use breakdowns if in use, any of the running subtotals above, or any other named charge already on the tariff. A tax is typically based on Subtotal, so it applies after adjustments and surcharges without compounding on itself.
- **Rate** — entered as a fraction (a 5% tax is 0.05), or a fixed amount depending on what it is based on.

Worked example, continued. Add a 6% tax based on Subtotal:

> Type: **Tax**, Based On: **Subtotal (Basis + Adjustments + Surcharges)**, Rate: **0.06**

With this tariff now complete, a July bill with 42,000 kWh of total energy use (30,000 kWh during peak hours, 12,000 kWh off-peak) and 80 kW of peak demand computes as:

Item	Calculation	Amount
Energy Charges	(30,000 × \$0.16) + (12,000 × \$0.09)	\$5,880.00
Demand Charges	(50 × \$9.00) + (30 × \$13.50)	\$855.00
Service Charges	Monthly Charge	\$25.00
Basis	sum of the above	**\$6,760.00**
Subtotal	Basis (no adjustments or surcharges on this tariff)	**\$6,760.00**
Taxes	6% of Subtotal	\$405.60
Total	Subtotal + Taxes	**\$7,165.60**

---

9. Eligibility Rules, Demand Ratchets, and Custom Variables

A tariff's Advanced section holds three less common mechanisms, along with the **Month Schedule Name** and **Buy or Sell** fields (a non-standard billing-cycle schedule, and whether the tariff represents purchases, sales, or net metering — defaulting to Buy From Utility).

Eligibility rules, added with **+ Add Eligibility Rule**, disqualify a tariff for a given month if some variable fails a threshold — for example, a tariff only available to accounts whose demand stays under a certain limit. The form asks for the variable being tested, whether the threshold is a minimum or maximum, the threshold value, whether the test counts total months or consecutive months, and how many months.

Demand ratchets and **custom variables** are added with **+ Add Demand Ratchet** and **+ Add Custom Variable**, but have no form — clicking either creates the object and opens it directly for editing. A demand ratchet holds a current month's demand to some fraction of a prior peak (a common clause is "billed demand is the greater of the current month's demand

or 90% of the summer peak"); a custom variable holds twelve hand-entered monthly values for use in another charge's Based On field. Both are documented in full in UtilityCost:Ratchet and UtilityCost:Variable.

---

10. Mechanisms Without a Form

Two mechanisms exist on the EnergyPlus objects behind a tariff with no form:

- **Real-time pricing**, via a Real Time Pricing Charge Schedule Name and Customer Baseline Load Schedule Name on the Tariff.
- **`UtilityCost:Computation`**, an object that scripts the tariff's calculation steps directly, for cases the standard charge categories can't express.

Both are reachable through each object's generic field editor. UtilityCost:Computation documents the computation scripting language in full.

29. Multi-Version EnergyPlus Support

1. Overview

Every EP3 model targets one EnergyPlus version — currently 24.1, 24.2, 25.1, 25.2, or 26.1 — set once and shared by every EEM in that model. Field labels, defaults, and which object types exist at all can differ from one version to the next, so EP3 keeps each supported version's schema available and tracks which one a given model targets.

This is a different sense of "version" than two other things elsewhere in EP3 that use the same word: EP3's internal plugin/schema version, which updates automatically with no model-level setting, and the ASHRAE 90.1 standard-year (2010/2016/2019) chosen for Appendix G baseline generation — *(see "ASHRAE 90.1 Appendix G Baseline Generation").* This chapter covers only the EnergyPlus engine/IDD version.

What this chapter does not cover:

- Adding a new EnergyPlus release's schema to EP3 itself — an EP3 development task, not a modeling workflow.
- ASHRAE 90.1 baseline-year selection — see the XREF above.
- Field-by-field detail on any specific EnergyPlus object — the EnergyPlus Input/Output Reference covers that; this chapter is about moving a model between versions, not the science of any one object.

---

2. Choosing a Version for a New Model

The Model Wizard's Model Settings tab includes an **EnergyPlus Version** field, defaulting to the newest version EP3 supports. Changing it restarts the Wizard, since every downstream Wizard tab depends on version-specific field definitions.

Use the newest available version unless a project has a specific reason not to. The recurring reason to pin an older one is reproducibility: reopening a model built years ago, at whatever EnergyPlus version was current then, reproduces the same simulation it always has. EP3 keeps every supported version's schema and behavior intact rather than forcing every model forward — a model's EnergyPlus version is a deliberate anchor, not a default to outgrow.

> *Figure placeholder — the Model Wizard's Model Settings tab with the EnergyPlus Version dropdown open, showing the list of available versions.*

---

3. Opening an Existing Model

When a model opens, EP3 compares the EnergyPlus version stored in the file against what the installed copy of EP3 supports.

- **The file's version is newer than the installed EP3 supports.** The model doesn't open. EP3 reports that the file was created by a newer EP3 and needs an update before it can be opened.
- **The file's version is older than the installed EP3 supports.** EP3 offers to upgrade the model, listing every supported version between the file's current one and the newest available.

The default choice is to leave the model at its current version — nothing changes unless a newer one is actively selected.

Declining the upgrade keeps the model exactly as it was: the same field set, defaults, and simulation behavior as when it was last saved. Accepting an upgrade runs the migration described in §4, one version step at a time, up to whichever version was chosen.

---

4. What Happens During an Upgrade

EP3 walks an upgraded model forward one version step at a time (24.1→24.2→25.1→25.2→26.1) rather than converting it in a single jump, applying whatever field- or object-level change each specific step requires. This is EP3's own migration logic, built from the specific renames and restructurings each EnergyPlus release actually introduced — it does not call EnergyPlus's own IDF-transition utility.

Most of the currently-supported steps need no changes at all. The one step with real content is 24.2→25.1, which renames and restructures a handful of fields on variable-speed coil objects and one comfort-related object:

- `ComfortViewFactorAngles` stored a separately-numbered `angle_factor_#`/`surface_#_name` field pair for each surface before 25.1; from 25.1 on, the same information is a single list of angle/surface pairs.
- `Coil:Cooling:DX:VariableSpeed` renames each higher speed's condenser air flow rate field to match the naming used by the lower speeds.
- `Coil:Heating:WaterToAirHeatPump:VariableSpeedEquationFit` renames its Speed 1 rated air flow field to include "Rate" for consistency with the other speeds.
- `Coil:WaterHeating:AirToWaterHeatPump:VariableSpeed` renames its per-speed rated capacity, COP, and sensible heat ratio fields to a consistent `Speed N ...` naming pattern.

EP3 applies each of these automatically as part of the upgrade — nothing to do by hand, and nothing that changes the value of any field, only where that value is stored.

Upgrading only ever moves forward. There's no path back to an older EnergyPlus version once a model has been upgraded — choose a version deliberately at the outset (§2) rather than planning to reverse the choice later.

---

5. Templates and Library Content Across Versions

HVAC system templates (*(see "The HVAC Outliner and Creating HVAC Systems", §7)*), construction/material templates (*(see "Constructions and Materials", §8)*), and space/zone type templates (*(see "Internal Loads, Space Types, and Zone Types", §7)*) carry the same forward-only migration built in: a template exported from a model at one version carries its content forward to every newer version automatically, using the same step-by-step logic as §4.

The same limitation runs in the other direction: a template exported from a model at a newer EnergyPlus version isn't available in a model at an older version. Migration only ever runs forward, so there's nothing to convert a newer template's content back down to an older schema — the older-version model simply won't offer that template in its library picker.

If a template needs to be usable on an older-version project, build and export it from a model at that version (or an older one), rather than from whatever version happens to be current at the time.

---

6. Out of Scope

- **Adding a new EnergyPlus version's schema to EP3.** Generating the field definitions for a release EP3 doesn't yet support is a task for the EP3 development team, not something done from within a model.
- **EP3's internal plugin/schema version.** Separately from the EnergyPlus version this chapter covers, EP3 tracks its own build/schema version to run internal housekeeping — renaming internal object types, migrating storage formats — when a model made with an older copy of EP3 opens in a newer one. This runs automatically, has no version picker of its own, and is unaffected by a model's EnergyPlus version choice.

---

7. Summary

An EP3 model targets exactly one EnergyPlus version, chosen once in the Model Wizard and changed only through an explicit, forward-only upgrade prompt when reopening an older model. That upgrade applies field- and object-level migrations built for each specific version step, most of which currently require no changes at all. The same forward-only migration governs exported templates, with the consequence that a template built at a newer version isn't available to a model at an older one.

End of draft chapter. Outstanding items for the next pass: supply the screenshot marked above; resolve the XREF placeholders once their target chapters exist, and confirm the section numbers cited for Ch.7 §8, Ch.9 §7, and Ch.13 §7 still match once those chapters are finalized; confirm with the EP3 team whether any field-level changes are planned for the 25.1→25.2 and 25.2→26.1 steps before stating outright that those steps require no migration.

31. Importing EnergyPlus Files

1. Overview

The **EP3 - Importer** dialog is a single dialog with three options, each a distinct, purpose-built way to bring EnergyPlus content into a model:

- **Import EnergyPlus File** — brings in a complete external file: a model built entirely outside EP3, or a file produced by EP3 itself. A separate **Select EnergyPlus Example File to Import** dropdown, further down the same dialog, offers a shortcut straight to EnergyPlus's own bundled example files, without browsing to their location on disk. §2 covers this path.
- **Import Templates** — reuses an HVAC system, construction set, or SpaceType/ZoneType pair saved earlier as a reusable EP3 template. §3 covers how this relates to the rest of the chapter; the template dialogs themselves are covered where each object type already lives — *(see "The Model Wizard", §3)*, *(see "Constructions and Materials", §8)*, *(see "Internal Loads, Space Types, and Zone Types", §7)*, *(see "The HVAC Outliner and Creating HVAC Systems", §7)*.
- **Paste Elements** — takes raw IDF text typed or pasted directly into the dialog, without a file on disk at all. §4 covers this path.

All three run through the same import engine, differing only in their starting material and default option choices. §2's mechanics — file types, options, collision handling — apply to all three.

Real-world reasons to use the Import EnergyPlus File path — porting a project built in another EnergyPlus-based tool (OpenStudio, DesignBuilder, or current versions of Carrier HAP and Trane TRACE 3D Plus, both of which now build on OpenStudio), opening one of EnergyPlus's example files to see a system type modeled directly in EnergyPlus objects, inspecting or QC'ing a file someone else produced — a colleague, another platform, or an AI tool — before trusting its content, or gathering starting material to turn into a reusable EP3 template (§3).

This chapter does not cover Importing `SizingPeriod:DesignDay` data from a `.ddy` file — *(see "Weather, Climate, and Design Conditions", §4.3.2)* covers this specifically, including a worked example.

---

2. Importing a Complete File

2.1 Supported File Types

The Import EnergyPlus File picker accepts `.idf`, `.epJSON`, and `.expidf` (an EnergyPlus-expanded file, produced by running the ExpandObjects preprocessor on a file containing `HVACTemplate:*` objects — *(see "The HVAC Outliner and Creating HVAC Systems", §8.1)* walks through that specific round trip end to end).

EnergyPlus needs to be installed and located (*(see "The EP3 Interface and Model Settings", §7.3)*) before any of these files can be imported.

2.2 The Import EnergyPlus File Dialog

Clicking the **Import EnergyPlus files or templates** toolbar icon opens the EP3 - Importer dialog (in Simplified View, it jumps straight to the Wizard's library browser instead — §3). Choosing

Import EnergyPlus File and picking a file prompts for the following, in order:

Prompt	Choices	Default	What it does
Erase All Existing EP3 objects?	Yes / No	Yes	**Yes deletes every existing geometry and non-geometry object in the model before importing anything.** Leave at **No** when adding a file's content into a model you're already building. Change to **Yes** when the intent is to import a complete model into EP3.
Import Geometry?	Yes / No	Yes	Builds Spaces, Zones, and surfaces from the file's geometry objects.
Import Non-Geometry Objects (schedules, loads, etc)?	Yes / No / Only Those Serving HVAC	Yes	**Only Those Serving HVAC** is for pulling just an HVAC system out of a file — pair it with Import Geometry set to No. Internal loads, thermostats, and other non-geometry objects the file defines are skipped, but the schedules, curves, and everything else the imported HVAC system needs to operate still come in.
Import Shading?	Yes / No	Yes	Builds shading geometry from the file.
Import Unused Constructions?	Yes / No	No	With the default No, a `Construction` the file defines but never assigns to an imported surface isn't created at all.
If objects with the same name are found:	see §2.3	Skip duplicate objects	Governs what happens on a naming collision — §2.3.

Prompt	Choices	Default	What it does
Consolidate Internal Loads if possible?	Yes / No	Yes	Merges loose `People`/`Lights`/`ElectricEquipment`/ objects and simple schedules that turn out to be field-for-field identical except for name, rather than keeping one separate object per zone.
Import HVAC Systems?	Yes / No	Yes	Brings in `AirLoopHVAC`, `PlantLoop`, `ZoneHVAC`, and related objects from the file. Only shown if HVAC import is available in this installation.
Consolidate Zone HVAC Objects if possible?	Yes / No	Yes	Only shown when Import HVAC Systems is Yes. Collapses the file's separate, per-zone HVAC equipment into single shared objects, tracking each zone's differing field values as a per-zone override rather than keeping one independent object per zone — the same kind of consolidation the HVAC Outliner's **Consolidate** button performs on demand (*(see "The HVAC Outliner and Creating HVAC Systems", §2.2)*), run automatically here as part of the import. §2.5 shows this happening to a set of VAV terminal units.

2.3 What Happens When Names Collide

Whether importing a file into a model that already has content causes problems depends on what's in both files — there's no rule that guarantees a clean merge, and none that guarantees a mess. Two files that both build on common EnergyPlus-example naming conventions — a `Construction` called "R-13 Wall," a `Schedule:Compact` called "Always On" — can collide on several objects even though neither file was built with the other in mind. A file built with deliberately distinct naming can merge in with none.

EP3 decides whether an incoming object matches something already in the model in two steps:

1. **By the id the source file stamped on the object.** This only ever succeeds for a file EP3 itself produced — an exported template, or an EP3-generated `*.idf` file. A file from OpenStudio, DesignBuilder, or any other outside source never matches this way.
2. **By name, case-insensitive, within the same object type.** For an externally-authored file, this is the only matching that happens. A `Construction` named "R-13 Wall" in the incoming file matches an existing `Construction` named "R-13 Wall" or "r-13 wall" — nothing else.

If neither check finds a match, the object is always created new, regardless of which overwrite choice you picked — overwrite mode only decides what happens once a same-named object of the same type already exists.

The **"If objects with the same name are found"** prompt then chooses what happens to each match:

Choice	Effect
Skip duplicate objects	The incoming object is dropped. The existing object is left exactly as it was.
Create a new object with a default name	The incoming object is still created, but under that object type's generic default name — the same placeholder name a brand-new object of that type would get — rather than the name from the file. The model ends up with both the original and the newly created object.
Overwrite inputs in the Global model	The existing object is kept (same name, same id, same connections to anything else in the model) and its field values are replaced with the incoming file's values.
Overwrite inputs in the *[EEM name]* EEM	Only offered when an EEM other than the Global model is active at import time. Instead of writing into the Global model, the incoming values are written as that EEM's override — *(see "Energy Efficiency Measures (EEMs)")* covers EEM override layers generally.

A file from recognized software is not guaranteed correct. Files brought in this way have shown up with duplicate or extra surfaces, and interior walls tagged with an outdoor boundary

condition instead of an interzone one. Check what an import produced before trusting it: review the Import Report (§2.4) for what was created, and run the Validation dialog (*(see "Model Troubleshooting and Quality Assurance")*) afterward regardless of how the source file was produced.

2.4 Reviewing What Happened

Every import — whether a complete file, a template, or a pasted snippet — produces two dialogs:

- **Import Report** — lists every object found in the source, one row per object: its type, its original name, whether it was imported, and the name it ended up with in EP3. Filterable by type and by imported/not-imported, and exportable to CSV. Every past import is listed under **Import Details**, in the Main Dialog's Column 1 Model Settings group, and can be reopened from there at any time — reading it doesn't require importing anything again.
- **Import Errors** — a plain list of warnings and errors from the import, shown only when something went wrong. Listed the same way, under **Import Errors Summary** in the same Column 1 group.

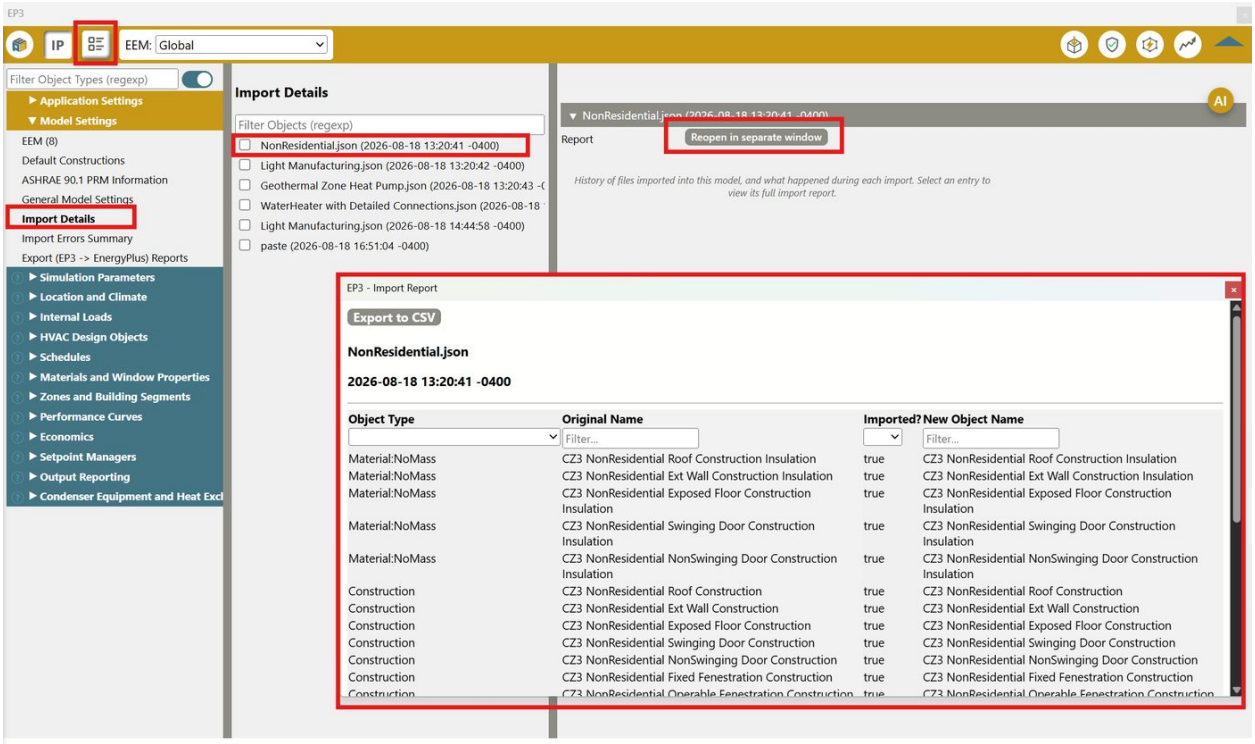


Figure 154: The Import Details view in Column 1, listing past imports by file name and timestamp, with one selected to show its Import Report — Object Type, Original Name, Imported?, and New Object Name columns, and an Export to CSV button.

2.5 Worked Example: Importing `5ZoneAirCooled.idf`

`5ZoneAirCooled.idf` ships with every EnergyPlus install, in its `ExampleFiles` folder. EnergyPlus Example Files It's a single `AirLoopHVAC` ("VAV Sys 1") delivering air through `AirTerminal:SingleDuct:VAV:Reheat` terminals to five zones (SPACE1-1 through SPACE5-1) and a return plenum, with a `Boiler:HotWater` ("Central Boiler") serving the reheat coils and a `Chiller:Electric` ("Central Chiller") on the cooling side.

1. Start a new EP3 project by clicking on the EP3 logo. Close the Wizard without selecting any templates.
2. Click the **Import EnergyPlus files or templates** icon in the main dialog's top toolbar.



Figure 155: The main dialog's top toolbar, with the "Import EnergyPlus files or templates" icon circled.

3. In the EP3 - Importer dialog, select `5ZoneAirCooled` from the **Select EnergyPlus Example File to Import:** drop-down.

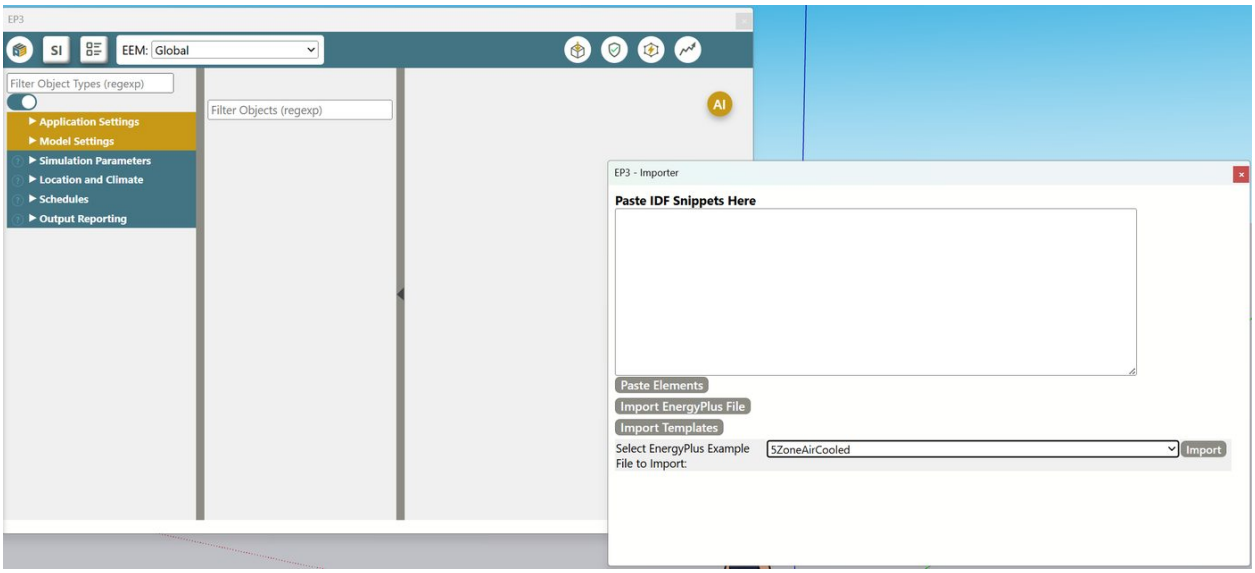


Figure 156: The EP3 - Importer dialog, with 5ZoneAirCooled selected in the Select EnergyPlus Example File to Import drop-down and the Import button beside it.

4. Click **Import** to import the `5ZoneAirCooled.idf` example file into the current model.
5. At the prompts: Import Geometry — Yes (the file's five zones need to become Spaces); Import Non-Geometry Objects — Yes; Import Unused Constructions — No (default; nothing built for a surface no imported geometry uses); Import HVAC Systems — Yes (brings in the airloop, boiler, and chiller); everything else at its default. The overwrite prompt has no effect here — there's nothing in the model to collide with.
6. Confirm the import.
7. Once the import is complete, explore the model. The Building Geometry Outliner shows the file's five zones grouped under one floor, with the return plenum on a floor of its own:

The HVAC Outliner shows VAV Sys 1 with its full supply-side branch (OA mixing box, OA and main heating/cooling coils, supply fan) and its Chilled Water Loop. The file's five separate `AirTerminal:SingleDuct:VAV:Reheat` objects — one per zone — collapse into a single shared "VAV Reheat" terminal on the Demand Side, with each zone's differing values (Availability Schedule Name, Maximum Air Flow Rate, and the rest) tracked as a per-zone override instead of five

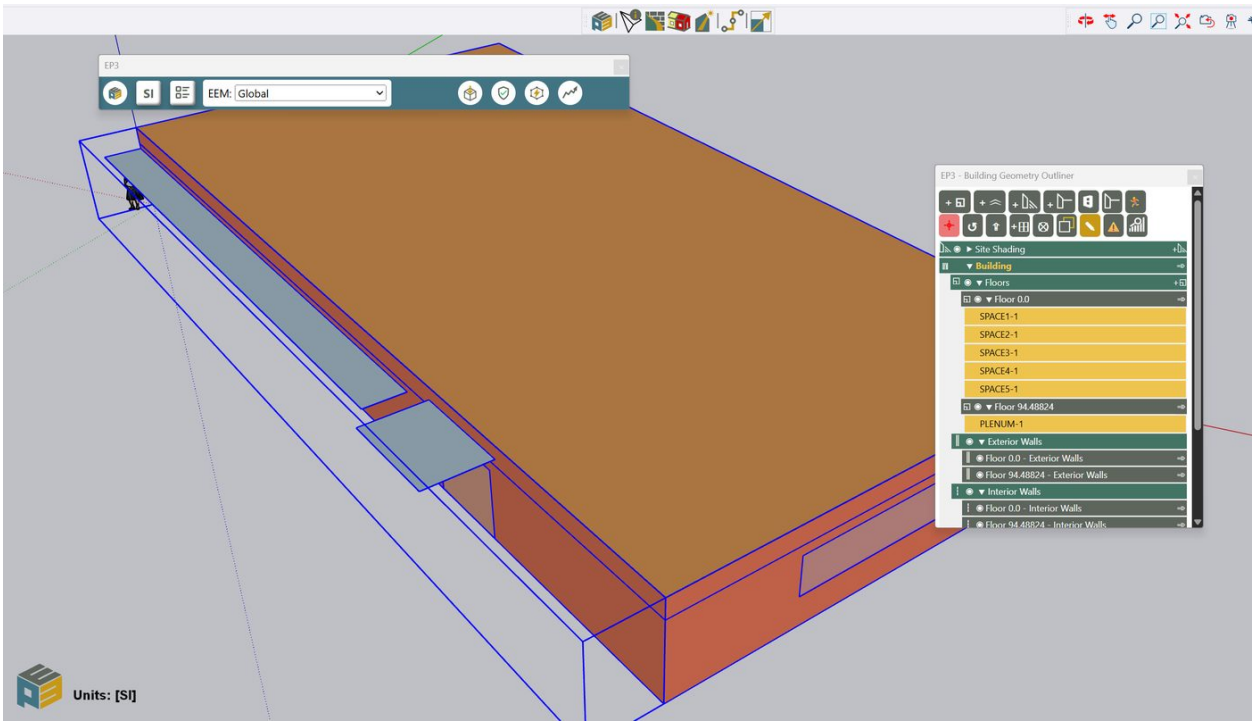


Figure 157: The Building Geometry Outliner after import: Floor 0.0 containing SPACE1-1 through SPACE5-1, and a separate floor containing PLENUM-1.

independent objects — the **Consolidate Zone HVAC Objects if possible?** prompt (§2.2) at work:

---

3. Importing a Template

Clicking **Import Templates** on the EP3 - Importer dialog (or, in Simplified View, the toolbar icon directly) opens the Model Wizard's library browser — the same dialog used during initial model setup. *(see "The Model Wizard", §3)* covers browsing and selecting from it. Under the hood, choosing a template runs the same import engine §2 describes, just with option defaults suited to what's being imported (an HVAC template import, for example, defaults to skipping geometry entirely, since a template file carries no geometry to bring in).

Because it's the same mechanism, anything brought in through §2's "complete file" path can become source material for a template: import a file completely, confirm in the model that it's built and behaves the way you expect, then follow the relevant domain chapter's "save as template" steps — *(see "Constructions and Materials", §8)*, *(see "Internal Loads, Space Types, and Zone Types", §7)*, *(see "The HVAC Outliner and Creating HVAC Systems", §7)* — on the pieces you want to keep for reuse.

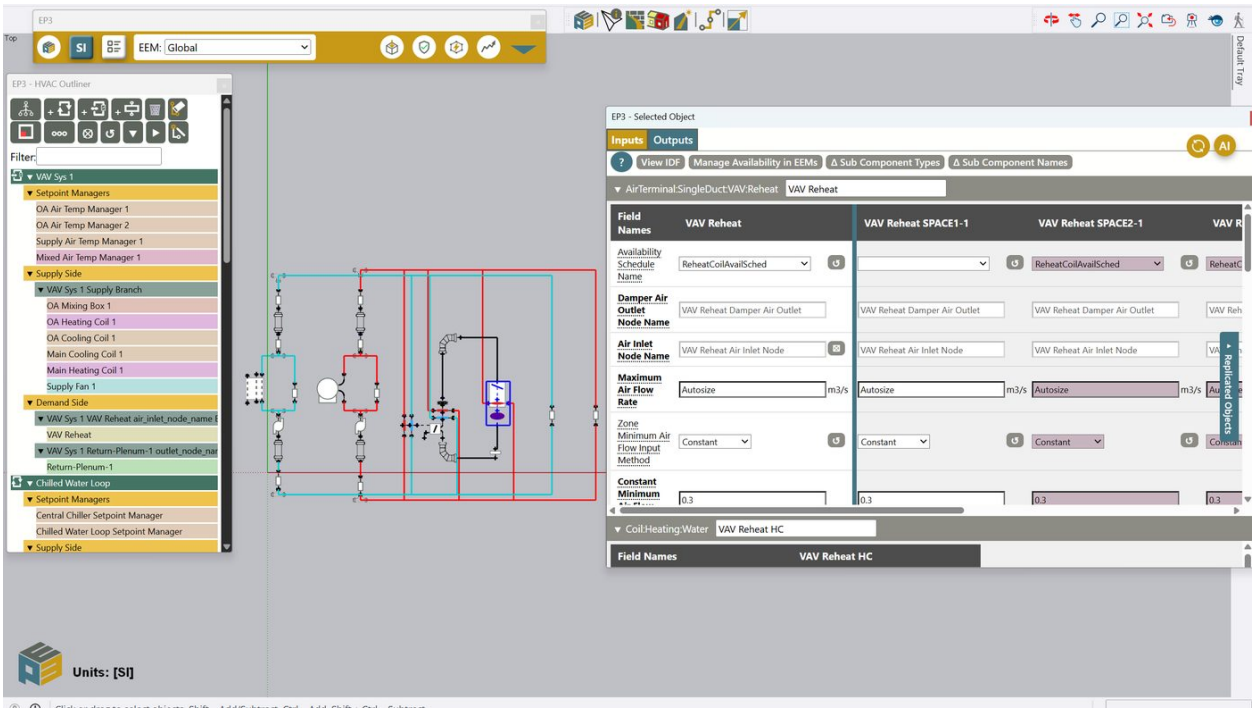


Figure 158: The HVAC Outliner after import, with VAV Sys 1 expanded and the consolidated VAV Reheat terminal's field table open, showing per-zone override columns (VAV Reheat, VAV Reheat SPACE1-1, VAV Reheat SPACE2-1, …) and the Replicated Objects tab.

4. Pasting a Snippet

Paste Elements, the third button on the EP3 - Importer dialog, takes IDF text typed or pasted directly into its text box — one object, or a handful — without saving anything to disk first. It's the fastest way to try a short piece of IDF copied from a reference file, the EnergyPlus Input/Output Reference's example syntax, a forum post, or a snippet an AI tool produced, when writing it into a file just to import it would be unnecessary overhead.

It differs mechanically from §2's complete-file import in two ways:

- **There's only one prompt** — the same overwrite-mode choice §2.3 covers. Every other option (Import Geometry, Import Shading, and the rest) doesn't apply; a pasted snippet always attempts to create everything it contains.
- **The pasted text has to already be valid, comma-and-semicolon-delimited IDF syntax.** EP3 doesn't correct or reformat it first.

Example. Pasting this single line:

```
Material:NoMass,CP01,Rough,0.3670000,0.9000000,0.7500000,0.7500000;
```

and confirming the overwrite prompt creates one ordinary `Material:NoMass` object named "CP01," which then behaves exactly like any object built by hand in the Main Dialog.

---

5. Export: Where Each Mechanism Lives

EP3 has no single "export the whole model" action separate from what other chapters already cover:

- **A full-model IDF, tied to preparing for or running a simulation** — *(see "Running Simulations", §7)* covers **Export IDF Only** and **Simulate Existing Files**.
- **Saving a hand-picked selection as a reusable template** — *(see "Constructions and Materials", §8)*, *(see "Internal Loads, Space Types, and Zone Types", §7)*, and *(see "The HVAC Outliner and Creating HVAC Systems", §7)* each cover the save mechanism for its object family.
- **Viewing an object's IDF text without saving anything** — right-clicking a single object and choosing **View IDF** (or, in the HVAC Outliner, viewing IDF for a whole selection at once) opens a read-only preview of its EnergyPlus representation. There's no save button on this dialog; it's for reading or copying text out — to paste into another tool, or to compare against a reference — not for producing a file. Saving a selection as a reusable file is what the template-export mechanisms above are for.

33. The EP3 AI Assistant

1. Overview

EP3 includes a built-in AI assistant, Atlas, built on Anthropic's Claude models. Atlas comes in two forms: an interactive chat, scoped to whatever object, selection, or simulation error you open it from, and a separate one-shot **Generate Report with AI** tool that writes a comparison report across a set of EEM results. This chapter covers both.

Atlas never changes the model without your approval. Anything it proposes — new or modified EnergyPlus objects, objects to remove — appears in the chat as a preview you explicitly accept or reject; nothing is written until you click Integrate (§5). EP3 runs every proposal through an automatic check first, but that check confirms the proposal is structurally valid EnergyPlus input — real fields, real object types, references that resolve — not that it behaves correctly once combined with everything else already in the model. Save a copy of the model before accepting a suggestion, particularly anything beyond a small, targeted fix, and verify an integrated change the same way you'd verify a change you made by hand (*(see "Model Troubleshooting and Quality Assurance")*).

Atlas is most useful for two things: tracing down what's causing an EnergyPlus error or a set of unmet hours (§6), and making small, targeted changes to objects already in the model. It's a weaker fit for assembling a system from scratch or making sweeping changes across many objects at once — a large proposal is harder to verify by reading it, and Atlas's checks only catch structural problems, not modeling ones.

Atlas's knowledge of EP3 is limited. It knows general EnergyPlus and HVAC modeling from training, but nothing about EP3's specific dialogs, workflows, or terminology beyond a short set of built-in instructions and whatever it looks up using the tools in §4 and §7. If you ask how to do something in the EP3 interface, rather than what a specific object or result means, verify the answer against this handbook or by checking the dialog directly before relying on it.

---

2. Getting started

Atlas requires an Anthropic API key, tied to an account you create directly with Anthropic rather than through EP3. Anthropic's guide to creating an API key covers signing up for a Claude Console account, adding billing, and generating a key. The first time you open Atlas without one configured, EP3 prompts you to enter it; once entered, it's encrypted and stored locally so you aren't asked again.

Contracting directly with Anthropic means your organization controls that relationship: you can review Anthropic's data-usage terms, monitor usage, and set billing limits from the Anthropic Console. EP3 does not currently offer that level of billing and account management — usage is billed by Anthropic directly to your account, and EP3 doesn't mark up or resell it.

Every session opens with a beta disclaimer: Atlas is provided as-is, you're responsible for verifying anything it produces before relying on it, and sending a message to Atlas sends information about your model — the objects, fields, or error text involved — to Anthropic's Claude API over the network. Nothing is transmitted until you send that first message. Opening the chat window and reading what it displays before typing anything, including an error file's contents echoed back to you, is safe on a confidential project; don't send a message on a project where that transmission isn't acceptable.

A model dropdown at the top of the chat window lets you choose which Claude model answers your questions. The list of available models is fetched live from Anthropic, so it changes over time as models are added or retired. Different models are given different sets of tools by default — a lighter, faster model may not have every tool attached until a question calls for it — this is handled automatically and isn't something you need to manage.

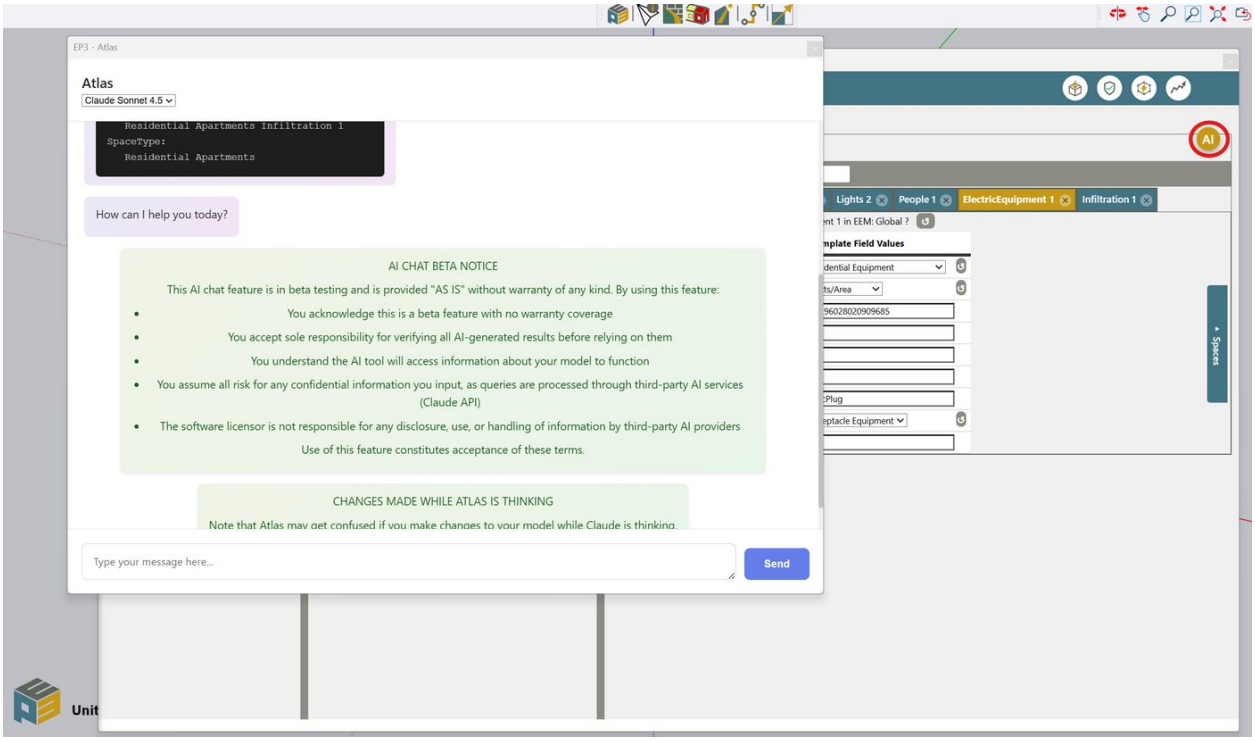


Figure 159: The Atlas chat window opening from a SpaceType's AI icon: the model dropdown, the SpaceType's IDF echoed back to confirm scope, "How can I help you today?", and the AI Chat Beta Notice below it.

---

3. Starting a conversation

Atlas is scoped by where you open it from:

- **From an object.** The gold **AI** icon in the top-right of Column 3 in the Main Dialog, or the same icon in the Selected Object dialog, starts a conversation seeded with the EnergyPlus representation of whatever's currently shown — a single object, or multiple selected objects (*(see "The EP3 Interface and Model Settings")* §2.4, §3).
- **From a simulation error.** Clicking a file's status button in the Results dialog opens the `.err` file. If Atlas is configured, this instead starts a conversation seeded with the error text and switches the model to the EEM that produced it, so Atlas reads and proposes changes against the correct EEM rather than whichever one happens to be active (*(see "Viewing and Interpreting Results")* §8.2). Nothing in the error text reaches Anthropic until you send a message (§2), so opening it this way is safe even on a confidential project. Without a configured API key, the same button opens the error text in a plain viewer instead.
- **General.** Opened with nothing selected or pinned to a particular EEM, for broad questions

about EP3 or EnergyPlus. Atlas can still request specific model or results information as needed using its tools (§4, §7).

If you switch EEMs mid-conversation. A conversation seeded from an error file is pinned to that error's EEM. If you switch the model to a different EEM while Atlas is composing a response, or before you click Integrate on a suggestion, Atlas pauses rather than reading or writing the wrong EEM's data, and shows a message asking you to switch back. A **Resume** button appears once you have; clicking it before switching back re-checks and asks again.

---

4. Asking Atlas about the model

Atlas can look up, without proposing any change:

- The names of every object of a given EnergyPlus type in the model.
- The full field set for one or more named objects (shown to you as IDF).
- The area of a wall, window, space, zone, or list of either, in the units EnergyPlus works in.
- Which spaces sit on a given floor plate.
- Whether the model assigns Space/Zone templates by name-matching or by manual selection (*(see "Internal Loads, Space Types, and Zone Types")*).
- The field-by-field schema for EP3-specific object types it wouldn't otherwise know (`SimpleThermostat`, `SpaceType`, `ZoneType`, `FloorPlate`), so it isn't guessing at what fields those support.

These are read-only lookups — asking a question never puts anything in front of you to accept or reject.

---

5. Getting suggested model changes

When a question calls for changing the model, Atlas proposes a set of EnergyPlus objects to add or modify, and, if something needs to come out to make room for them, a matching list of objects to remove. Both appear together in the chat as a decision box with **Integrate** and **Reject** buttons, alongside a preview of what's being proposed, rendered as IDF — native EnergyPlus format.

5.1 What EP3 checks before you see it

Before a proposal reaches the decision box, EP3 validates it against the EnergyPlus schema. A field Atlas invented that doesn't exist on that object type, or a reference to an object that doesn't exist at all, is caught automatically and sent back to Atlas for correction, so only a corrected version reaches you.

`Curve:*` and `Table:*` objects are blocked outright — Atlas cannot create or modify a performance curve or table under any circumstance. AI-generated curve coefficients have not been reliably sound, and a bad one is exactly the kind of error that looks plausible but causes simulation errors or results in wildly incorrect outputs (*(see "Performance Curves, Part-Load, and Efficiency Behavior")*).

This check confirms the proposal is structurally valid EnergyPlus input. It does not confirm the proposal behaves correctly once combined with everything else already in the model. Verify an integrated change the same way you would verify any other manual edit.

Two behaviors Atlas is instructed to follow consistently when a proposal touches `SpaceType` or `ZoneType`: it creates a new `SpaceType`/`ZoneType` pair together, never one without its companion (*(see "Internal Loads, Space Types, and Zone Types")*), and it will not propose changes to `WaterUse:Equipment` inside a `ZoneType` — that has to be edited by hand.

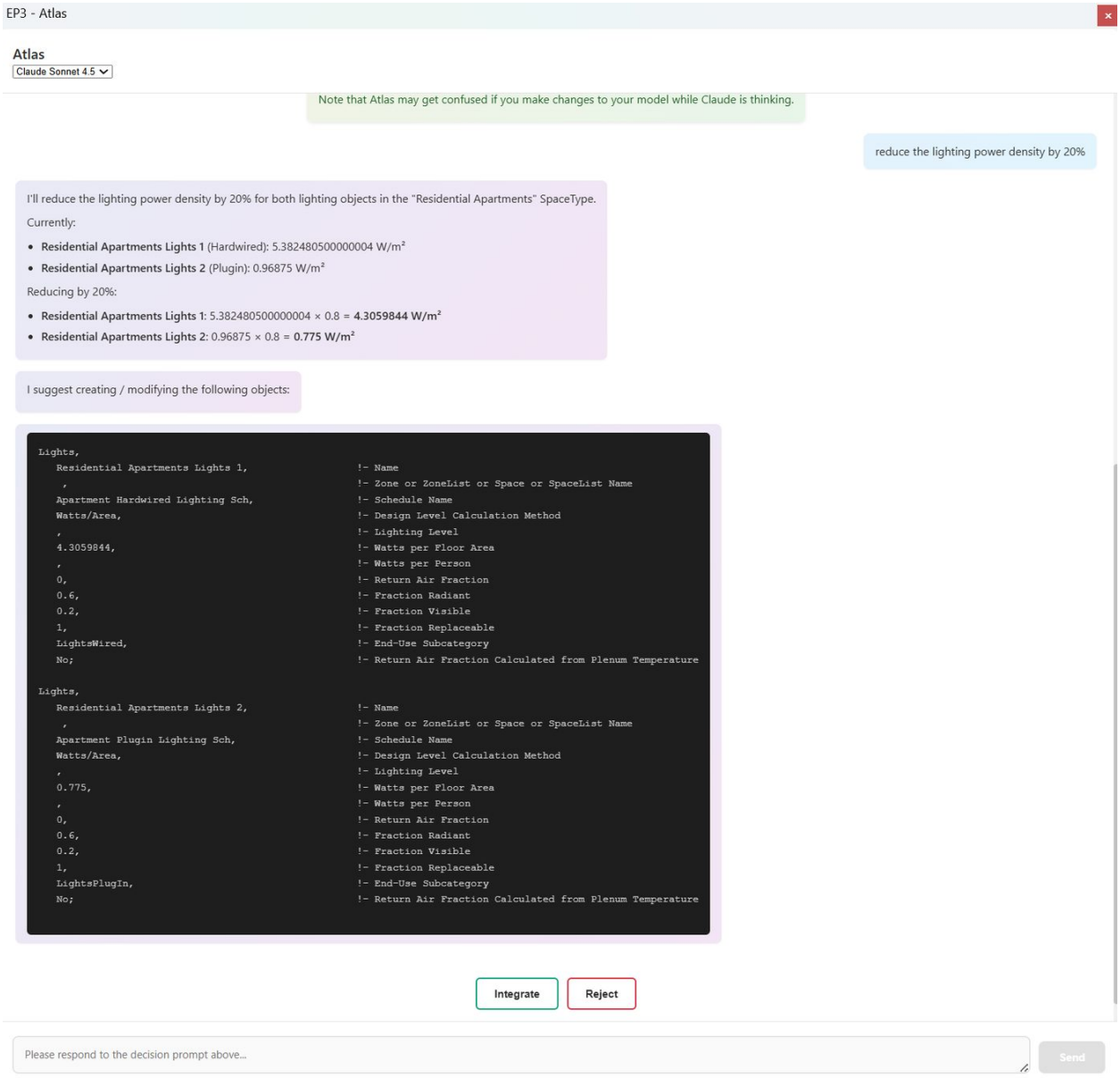


Figure 160: An Atlas decision box after asking to reduce lighting power density by 20%: Atlas's worked math for the SpaceType's two Lights objects, the resulting IDF preview with both Watts per Floor Area values updated, and the Integrate/Reject buttons.

5.2 Integrating or rejecting

Reject discards the suggestion and tells Atlas it was rejected, so it can offer an alternative if you ask. **Integrate** writes the proposal into the model through the same import path used for pasting IDF or epJSON text directly. EP3 creates the same **Import Details** and **Import Errors**

Summary reports for Atlas suggestions as it does for every other import. You can review your model import history, including Atlas suggestions under **Model Settings**.

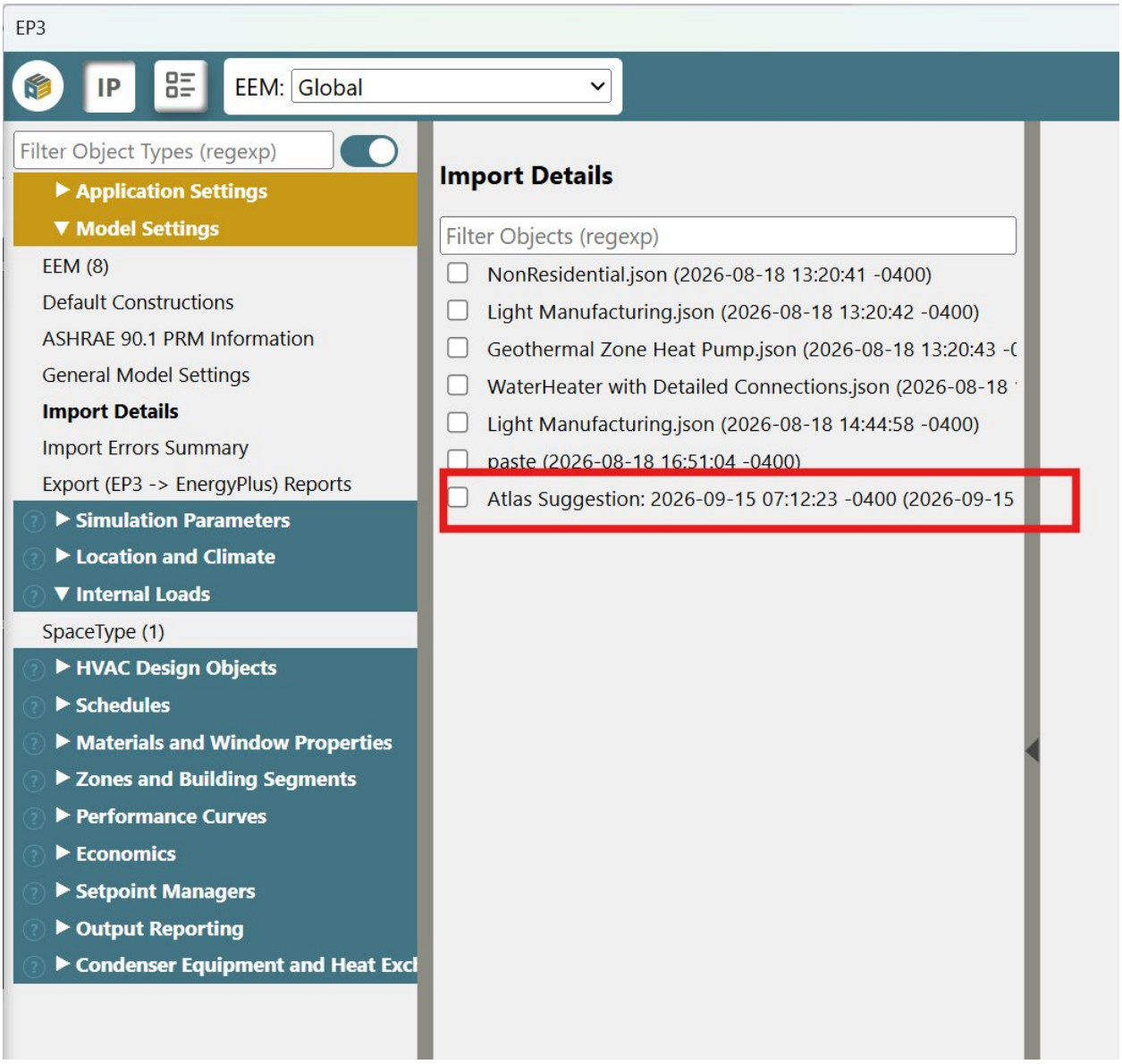


Figure 161: Model Settings > Import Details, listing an integrated Atlas suggestion alongside ordinary file and pasted-text imports, timestamped and named "Atlas Suggestion."

---

6. Diagnosing errors and unmet hours

This is the most common way Atlas gets used day to day: tracing a specific EnergyPlus error, or a specific set of unmet heating/cooling hours, back to its cause.

Starting from a `.err` file (§3) seeds the conversation with the file's text and the EEM that produced it, so Atlas can request the specific objects a message names rather than requiring you to trace through the file by hand.

Asked about unmet hours in any conversation, Atlas is instructed to work through the same checklist, roughly in this order:

1. Coil capacity against system and plant sizing.
2. Availability schedules against the hours the load goes unmet.
3. Heating/cooling sizing ratios set too low relative to the peak load.
4. Zone sizing assumptions — design supply air temperature, outdoor air fraction — against what the system delivers.
5. Plant loop supply temperatures against the design temperatures coils were sized for.
6. Setpoint manager conflicts between plant control and equipment control.
7. Autosized and hard-sized components mismatched on the same system.
8. DOAS-related zone sizing fields against the model's air-side topology.
9. Plant loop convergence warnings in the `.err` file.

Several of these tie directly to mechanics covered elsewhere: zone sizing (*(see "Zone-Level HVAC Equipment and Zone Sizing")*), system and plant sizing (*(see "Air Loops, Branches, and Nodes")*, *(see "Central Plant Equipment and Plant Loops")*), and the Validation dialog's own setpoint and sizing checks (*(see "Model Troubleshooting and Quality Assurance")* §2.8.2, §4).

---

7. Working with results and charts

Within a conversation, Atlas can pull the names of available results tables and their full contents for the current EEM, list which output variables were reported and over what date ranges, and work with hourly, timestep, or detailed time-series data — either by building a bar, line, scatter, pie, or histogram chart directly from it (rendered in the chat, without the underlying data ever passing through the conversation itself) or, when a calculation needs the raw numbers, by pulling a bounded slice of the time series to work with directly.

---

8. Generating a written report

Generate Report with AI, a button in the Results dialog, is a separate, one-shot workflow rather than an open-ended conversation: you describe the report you want section by section, and Atlas writes the whole thing at once, comparing whichever EEM result files are currently loaded in the Results dialog when you click it.

8.1 Defining the report

The Report Generator dialog collects:

- **Writing Style** — audience, tone, and any formatting preferences, in plain language.
- **CSS Style** — custom CSS rules applied to the finished report.
- **Relevant Project Data** — project name, address, client, preparer, phase, or anything else you want available as context; this also feeds an automatically generated report header.
- **Sections** — added one at a time via **Add Section**, which asks whether the section will have sub-sections (**Content** vs. **Parent**). A parent section holds one or more sub-sections, each added with **+ Add Sub-section**; both section and sub-section types get a free-text

description of what should go there. Sections can be reordered with the up/down arrows and removed with **Delete**.

Each section or sub-section carries a **Write last with context** checkbox. Checked, that section is written after every other section has been generated, using their finished content as additional context — useful for a results/conclusion section meant to reflect the whole report rather than being written in isolation. This only affects generation order, not where the section appears in the finished document; placement is controlled by the section list's order instead.

The dialog opens with a default three-section structure — Introduction; Input Summary, with sub-sections for Building Envelope, Internal Loads, HVAC Systems, Utility Rates, and Unmet Hours; and Results and Conclusion — as a starting point to edit rather than a fixed template.

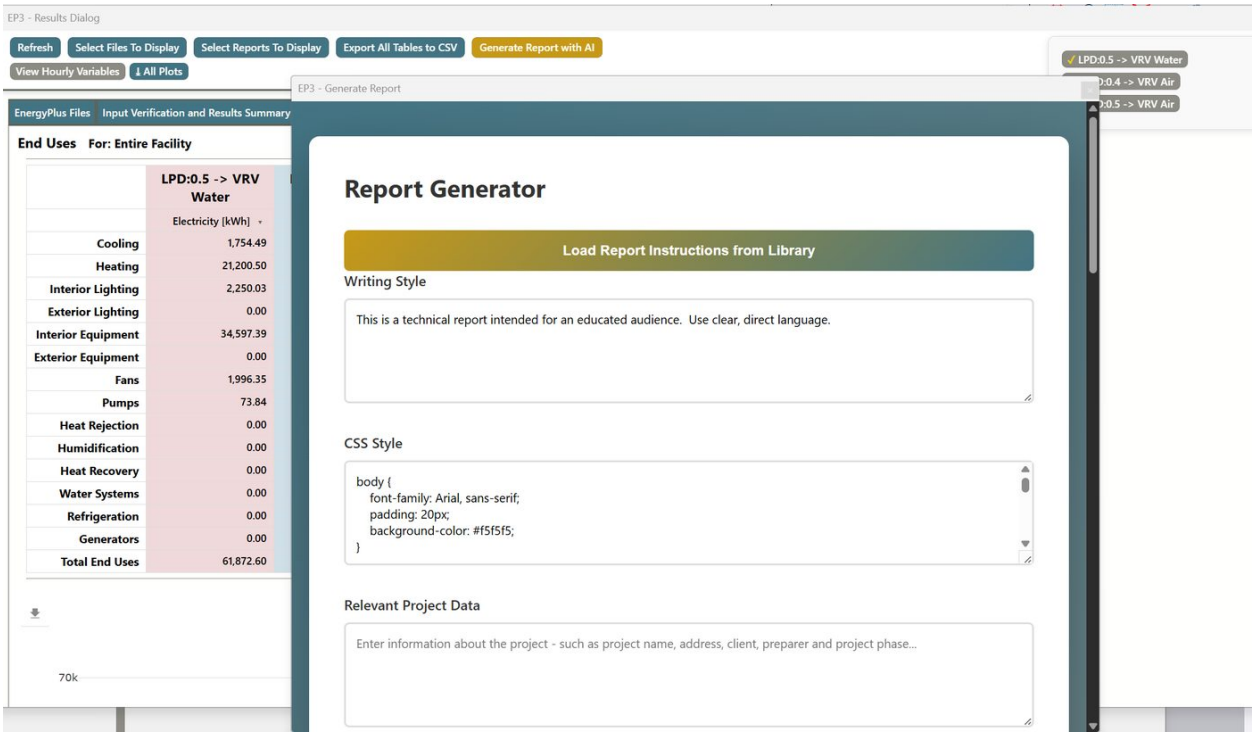


Figure 162: The Report Generator dialog opened from the Results dialog's Generate Report with AI button, showing Load Report Instructions from Library and the default Writing Style, CSS Style, and Relevant Project Data fields.

8.2 Reusing a report structure

Load Report Instructions from Library opens a picker rooted at your configured AI Report Generation Instructions library folder (*(see "The EP3 Interface and Model Settings")* §7.4) and loads a previously saved section/style structure into the dialog. **Save Report Structure for Later** writes the current structure back to that library as a reusable file, independent of any particular project. Separately, the model remembers the last complete report definition generated from it, and reopens the dialog pre-filled with that the next time you generate a report for the same model.

8.3 Generating and reading the output

Generate Report shows the same beta disclaimer as the chat assistant, then prompts you for where to save the finished report as an HTML file. While it works, a chat-style progress window lists each section as Atlas writes it. The report always includes an isometric snapshot image of the building model, placed automatically near the top.

The output is a single, self-contained HTML file with its charts embedded — EP3 saves it to disk but doesn't reopen it for you, so open it in a browser once generation finishes. Report generation uses a fixed Claude model chosen for speed and cost rather than the model you'd pick for the interactive chat; it isn't user-selectable here.

---

9. Where to Look Next

- *(see "Model Troubleshooting and Quality Assurance")* — verifying a model before and after accepting an Atlas suggestion; unmet hours and `.err` file interpretation in depth.
- *(see "Viewing and Interpreting Results")* — the file status panel Atlas's error-diagnosis entry point opens from, and the results tables/output variables Atlas queries in §7.
- *(see "Performance Curves, Part-Load, and Efficiency Behavior")* — why curve and table creation is off-limits to Atlas specifically.
- *(see "Internal Loads, Space Types, and Zone Types")* — the `SpaceType`/`ZoneType` pairing Atlas maintains when it proposes either.
- *(see "The EP3 Interface and Model Settings")* — the AI icon's locations, and the AI Report Generation Instructions library folder.

End of draft chapter. Outstanding items for the next pass: supply the screenshots marked above; resolve the XREF placeholders once their target sections/pages exist; confirm final document format (HTML vs. PDF vs. other).

35. Modeling Workflows

1. Overview

The rest of this manual is organized by object domain — geometry, envelope, HVAC, and so on. This chapter is organized around a different question: given a project you're about to start, which combination of EP3's tools fits? Each section below is a sequence of decision points built entirely from tools covered in full elsewhere, gathered here by project purpose rather than by domain.

The technique spectrum. Every workflow below draws on the same three layers of modeling technique, in increasing order of manual effort and precision:

- **Wizard-only** — complete templates for internal loads, constructions, and HVAC systems, optionally paired with the Wizard's auto-zoned rectangular geometry, producing a simulation-ready model in one pass. *(see "The Model Wizard")*
- **Wizard + hand-built (detailed mode)** — Wizard templates as a starting point, refined with hand-built geometry, zoning, envelope, loads, and HVAC assembly. *(see "Creating Building Geometry")* and the chapters through Part IV.
- **Advanced (template/component-level)** — building or customizing reusable templates, assembling HVAC from individual components rather than a prototype system, or round-tripping through EnergyPlus's native objects. *(see "The HVAC Outliner and Creating HVAC Systems", §§6-8)*

Which layer, or combination of layers, a given project purpose calls for is the question each section answers. *(see "A Complete Worked Example, End to End")* carries one project through the standard detailed-design workflow (§3) in full depth; this chapter stays at the level of which steps to take and in what order, not how each step works.

---

2. Quick Feasibility / Concept Study

When it fits. Early design, before real geometry exists — comparing a handful of HVAC system types or envelope strategies at an order-of-magnitude level, where an approximate read on which direction performs better is the goal.

The recipe:

1. Run the Wizard using its automatic rectangular-building option rather than drawing geometry by hand — geometric fidelity isn't the point at this stage. *(see "The Model Wizard", §7)*
2. Select whichever internal-loads and construction templates are a reasonable match for the building type. Precision matters less here than arriving at a complete, simulate-able model quickly.
3. On the HVAC Systems tab, check every system type worth comparing — each one becomes a separate EEM automatically, ready to compare side by side. *(see "The Model Wizard", §6)*
4. Leave Utility Rates blank unless cost is already part of the question you're answering.
5. Simulate every HVAC EEM and compare results directly, or, once more than a couple of options are in play, on the Parallel Coordinates plot. *(see "Running Simulations")*, *(see "Energy Efficiency Measures (EEMs)", §8)*

(see "The Model Wizard", §11) works through this exact sequence as a concrete worked example, end to end — start there for the hands-on version of this recipe.

When to graduate out of this workflow: once the project needs real geometry, a specific system rather than a menu of prototypes to choose among, or code-compliance quantification — move to §3 or §5.

---

3. Standard Detailed-Design Model

When it fits. The normal case: a real building, a specific design intent, and a project that will be revisited repeatedly as design development and construction documents progress — not a one-off comparison.

The recipe:

1. Run the Wizard for a fast starting point on constructions, internal loads, an HVAC system to customize, and (optionally) rectangular building geometry rather than building every supporting object by hand. *(see "The Model Wizard")*
2. Replace the Wizard's placeholder geometry with hand-built geometry matching the design. *(see "Creating Building Geometry")*
3. Assign zones and a thermal-block strategy that reflect the design's zoning intent. *(see "Spaces and Zones")*
4. Refine constructions and materials, internal loads, and schedules beyond the Wizard's generic templates wherever the design departs from them. *(see "Constructions and Materials")*, *(see "Internal Loads, Space Types, and Zone Types")*, *(see "Schedules")*
5. Build the HVAC system: keep the Wizard's template and customize it, or assemble one from individual components if the design doesn't match any available template. *(see "The HVAC Outliner and Creating HVAC Systems")*
6. Run the model through the Validation dialog before simulating. *(see "Model Troubleshooting and Quality Assurance")*
7. Simulate and review results. *(see "Running Simulations")*, *(see "Viewing and Interpreting Results")*
8. Revisit the same model as the design develops rather than rebuilding it — the "one model, updated in place" pattern this whole manual assumes. *(see "What Is EP3?", §2)* Capture design alternatives explored along the way as EEMs rather than duplicating the file. *(see "Energy Efficiency Measures (EEMs)")*

---

4. Parametric / Sensitivity Study

When it fits. You already have a working model — typically from §2 or §3 — and want to know which of several parameters moves the results, or need to present a range of outcomes across a design variable rather than a single point answer.

The recipe:

1. Start from a working base model. This is the parent EEM (or Global) every variant will inherit from. *(see "Energy Efficiency Measures (EEMs)", §1-3)*
2. For each parameter to test, create a Branching EEM scoped to the object type and field(s) that parameter lives on, with one iteration per value you want to test. *(see "Energy Efficiency Measures (EEMs)", §6.1)*

3. On whichever EEM should run the sweep, include the specific iterations you want (or all of them) under **Simulation Branches from Parent EEMs**. *(see "Energy Efficiency Measures (EEMs)", §6.2)*

Testing more than one parameter at once multiplies out to a run per combination — a 3-value branching EEM and a 4-value branching EEM together produce 12 runs, not 7. *(see "Energy Efficiency Measures (EEMs)", §6.3)* Scope each branching EEM deliberately before enabling it — enabling branching on an EEM can't be undone — and check the reported run count before committing a large combination to a batch simulation.

4. Simulate the whole batch at once. *(see "Running Simulations")*
5. Compare the swept parameter(s) against the results together on the Parallel Coordinates plot, rather than reading each run's output table individually. *(see "Energy Efficiency Measures (EEMs)", §8)*

When not to reach for a Branching EEM: if what you're comparing isn't a clean sweep of one object type's fields — three fundamentally different HVAC system topologies, for example, rather than several efficiency values of the same equipment type — ordinary side-by-side EEMs are the more natural fit. *(see "Energy Efficiency Measures (EEMs)", §1)* already draws this distinction.

---

5. Code-Compliance Model (ASHRAE 90.1 Appendix G Baseline)

When it fits. Performance-path code compliance, or a certification pathway that requires quantifying savings against a code-mandated baseline rather than a self-defined comparison.

The proposed design is built the same way as any standard detailed-design model (§3). Climate Zone, set on the Wizard's Model Settings tab (or independently, later), feeds directly into baseline system selection. *(see "The Model Wizard", §2)*

EP3 includes an ASHRAE 90.1 Appendix G baseline generator that produces the corresponding code baseline for comparison against the proposed design. *(see "ASHRAE 90.1 Appendix G Baseline Generation")* covers the generator's inputs, outputs, and comparison mechanics in full. Build the proposed model per §3, then hand it to the baseline generator.

---

6. Choosing Among These

Project purpose	Typical technique layer	Start here
Quick feasibility / concept study	Wizard-only, auto-zoned	§2; *(see "The Model Wizard", §11)*
Standard detailed-design model	Wizard + hand-built detailed mode	§3
Parametric / sensitivity study	An existing model, extended with Branching EEMs	§4; *(see "Energy Efficiency Measures (EEMs)", §6)*
Code-compliance model	Standard detailed-design model + baseline generator	§5; *(see "ASHRAE 90.1 Appendix G Baseline Generation")*

These aren't mutually exclusive over a project's life. A quick feasibility study (§2) commonly becomes the seed of a standard detailed-design model (§3) once a direction is chosen; a detailed-design model routinely picks up a parametric study (§4) partway through to settle an open question, and a code-compliance requirement (§5) on top of that — all within the same file, per *(see "What Is EP3?", §2)*.

---

7. Where to Look Next

- *(see "The Model Wizard")* — the Wizard in full, including the worked example §2 and §5 both build on.
- *(see "The HVAC Outliner and Creating HVAC Systems")* — template, hand-assembled, and EnergyPlus-native paths to building an HVAC system.
- *(see "Energy Efficiency Measures (EEMs)")* — EEM mechanics, Branching EEMs, and the Parallel Coordinates plot behind §4.
- *(see "Model Troubleshooting and Quality Assurance")* — validating a model before trusting any of these workflows' results.
- *(see "ASHRAE 90.1 Appendix G Baseline Generation")* — the baseline generator behind §5.
- *(see "A Complete Worked Example, End to End")* — the standard detailed-design workflow (§3) carried through in full depth.

End of draft chapter. Outstanding items for the next pass: §5 stays intentionally high-level until Ch.27 (ASHRAE 90.1 Appendix G Baseline Generation) is researched and drafted — revisit this chapter once that one exists, both to verify §5's claims and to fill in its XREF targets; resolve the remaining XREF placeholders once their target chapters exist; final placement of this chapter (Part IX vs. an early Part I chapter) is still undecided per the outline.