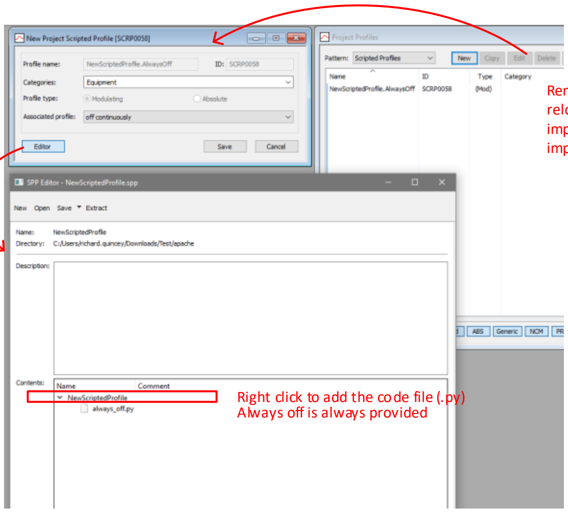
Python in the VE

Scripted profiles – daytime & night purge ventilation

What? This scripted profile can be used for controlling Macroflo openings (or an Air exchange); it provides day time summer temperature operation and tests for a room condition of > 23°C at approx. 11pm in order to switch on night purge ventilation

Why? We need to use a scripted profile in this instance because of the need to test at a specific time of day and to persist this state over the time-steps that constitute the night-time period. Scripted profiles offer a means to extend profile functionality when more complex logic or external data is involved



Scripted profiles can be created in any Python IDE or text editor.

The code is packaged-up when the scripted profile is created in ApPro. It is saved in a .spp file in the Model Apache folder.

Any code edits will require the code file to be replaced and the package re-created.

We recommend testing & debugging any algorithms as a straight script before inserting into a profile class.

We also recommend adding comments to all scripts … use # or """ … """ for blocks of comment to aid maintenance.

Errors for scripted profiles will appear in the Task manager when you simulate.

Also refer to scripted profile help.

Script:

```python
# Daytime & secure night cool Window & door opening scripted profile
# Night-time algorithm uses first reporting point after 11pm based on reporting timestep
# and operates at max all night

import apache

class Occupied_Day_Secure_Night (apache.Profile):
    # Window opening 8am - 11pm & occupied, Ta 22 to 26 degC, ramp opening 0-1
    # Window opening 11pm - 8am open if Ta > 23 degC at 11pm
    category = apache.Modulating

    def setup(self):
        # Initialise a night purge flag that will persist over timesteps
        self.night_purge = 0
        # Initialise opening output
        self.opening = 0
        # Get reporting interval and determine first data point in an hour in minutes past the hour
        interval = self.context.simulation.reporting_interval_mins
        if interval == 6:
            self.mins = 3
        elif interval == 10:
            self.mins = 5
        elif interval == 30:
            self.mins = 15
        elif interval == 60:
            self.mins = 30

    def evaluate(self):
        # Night time purge flag set each evening
        # Use first datapoint after 11pm for test; so for 10 min reporting 11:05
        if self.context.simulation.hour == 23 and self.context.simulation.minute == self.mins and self.context.surface.space.ta > 23:
            self.night_purge = 1

        # Night time purge flag reset each morning
        if self.context.simulation.hour == 8 and self.context.simulation.minute == self.mins:
            self.night_purge = 0

        # Daytime opening control
        if self.context.simulation.hour >= 8 and self.context.simulation.hour < 23 and self.context.surface.space.occ > 0:
            # Daytime hours occupied ramp control
            # Set opening value for values outside ramp limits
            if self.context.surface.space.ta <= 22:
                self.opening = 0
            if self.context.surface.space.ta > 26:
                self.opening = 1
            # Set opening value for values inside ramp  limits
            if self.context.surface.space.ta > 22 and self.context.surface.space.ta <= 26:
              self.opening = (self.context.surface.space.ta - 22) / 4
        else:
            # Outside of occupancy and daytime hours then closed
            self.opening = 0

        # Night purge opening control
        if (self.context.simulation.hour >= 23 or self.context.simulation.hour < 8) and self.night_purge == 1:
            # Night time hours night purge control
            self.opening = 1

        return self.opening
```

Import apache module; this is required to access IESVE scripted profile functionality

A scripted profile is defined by a Python class; one or more classes can be in each script (.py) file

Set the profile type

The setup function initializes data; this is done once. Here we initialise the night purge flag, interval & output variable. Not the use of self; this references the current instance of the class

APS file output is never on the hour; so we check the reporting interval and set the minutes after the hour for the first reporting interval possible

The evaluate function is called each simulation time-step and a value returned to the profile

The .context variables are specific to the Apache module; we check to see at the specified time the room temperature is > 23 degC. If it is we set the night purge flag to 1. We also reset it to 0 before morning occupancy

The daytime opening control produces a ramp output of 0-1 during occupied daytime hours

If the night purge flag is set we output to 1 for all night-time hours

The class returns the scripted profile value each time-step

Sample output:

Night cool operation when >23°C test is met

Summer daytime operation during occupancy