How to: IDEAS Building model

1. Create own package where the whole project is located [Ctrl+Shift+P or right click on a package in Package Browser -> New-> Package]
   - Tick "save as one file" if small, otherwise as separate files (better for version tracking)
   - All names cannot contain spaces (or special characters, etc.)
2. Create new **MyBuilding** model in your package [Ctrl+Shift+M or right click on the package -> New-> Model]. Extend from `IDEAS.Interfaces.Building`.
   - For any model you make, it is useful to often check for syntax or connection errors [Edit-> Check-> Normal]. A check 'with simulation' will also check if the model works with the specified inputs, etc.
3. In your `MyBuilding` model **redeclare** the `Medium` you want for the air in your building (can be one from Modelica library, the IDEAS library, or one of your own predefined media) [in the `MyBuilding` model, the Modelica text write: `extends IDEAS.Interfaces.Building(redeclare package Medium = Lib.Example_IDEAS.Data.Medium)`]
4. To select the specific models you want for the building (structure), ventilationSystem, heatingSystem, occupant and inHomeGrid you can use the drop down menus from the diagram [right click on the component->change class…->All matching choices or Browse to your model] or write directly in the Modelica text
   [e.g. `extends IDEAS.Interfaces.Building(redeclare package Medium = Lib.Example_IDEAS.Data.Medium, redeclare IDEAS.VentilationSystems.None ventilationSystem(redeclare package Medium = Medium), redeclare IDEAS.HeatingSystems.Heating_Embedded heatingSystem, redeclare Structure_FH building(AZones={11.88,11.88}, VZones=building.AZones .* 3.3, redeclare package Medium = Medium), redeclare Occupant_ISO13790 occupant(AZones=building.AZones, nLoads=0));`]
   - You can change parameters of the models through the diagram interface [right click->parameters] or in the text [writing the assignment of values or redeclarations in parentheses in the respective component as in the example above `occupant(AZones=building.AZones)`]
   - Don't forget to assign the same medium to your building and ventilationSystem [`(redeclare package Medium = Medium)`]. Here `Medium` is the parameter of the overall `MyBuilding` model, and its value is passed to the parameter `Medium` in the building model and the ventilationSystem model.
5. Models that represent the building structure have to be extended from the `IDEAS.Interfaces.BaseClasses.Structure` interface in order to be compatible and also to appear in the drop-down menu. Similar rules apply for the other components and the respective interfaces in `IDEAS.Interfaces.BaseClasses`.
6. Details on the ventilationSystem, heatingSystem, occupant and inHomeGrid models are not given here. Example models can be found in the IDEAS library [`IDEAS.VentilationSystems, IDEAS.HeatingSystems, IDEAS.Occupants`]
7. Create a building **MyStructure** model extending from the needed interface and give the overall parameters [`extends IDEAS.Interfaces.BaseClasses.Structure(final Q_design={north.Q_design,south.Q_design}, final nZones=2, final nEmb=0);`]
   - `final` means that these parameters will no longer be able to change from higher models, e.g. changing their values in `MyBuilding`.
   - `nZones` is the number of thermal zones in this structure and `nEmb` is the number of embedded ports, which is how many components will have an active thermal layer (e.g. floor heating)
   - `Q_design` is the design heating load for each zone of the structure and has to be manually assigned from the zone components (depends on the name one gives to his/her zones).

8. Drag and drop the **zone** models `IDEAS.Buildings.Components.Zone`. Change the needed parameters and redeclare the `Medium` parameter of the zone to be equal to the `Medium` of `MyStructure` (this latter takes its value from `MyBuilding`, see earlier)
   [`IDEAS.Buildings.Components.Zone south(V=39.2, nSurf=6,redeclare package Medium = Medium) "south zone of office area";` , or through the parameter window of the zone component: Edit Text-> type `Medium`]
   - `nSurf` is the number of surfaces (walls, windows…) that will be connected to the zone
9. In a similar way, drag and drop the **wall and window components** according to your needs from `IDEAS.Buildings.Components`. You can rotate and flip the components using the right click menu. For similar components (same construction details, but different orientation, size) only one object needs to be drag & dropped. Then it is transformed into an array in the text, and the properties are given also as arrays
   ```
   [ IDEAS.Buildings.Components.Window windows[2]( A={2,1.5},
   azi={IDEAS.Constants.South,IDEAS.Constants.North}, frac={0.15,0.2},
   inc={IDEAS.Constants.Wall,IDEAS.Constants.Wall},
   redeclare Lib.Example_IDEAS.Data.Materials.Glazing glazing,
   redeclare Lib.Example_IDEAS.Data.Materials.Frame fraType,
   redeclare  IDEAS.Buildings.Components.Shading.None shaType,) "Windows of day
   zone" ].
   ```
   - Window : select geometric parameters, glazing and frame, as well as shading device and optional control.
   - OuterWall : select geometric parameters, construction type (insulation type and thickness if you're using them in your construction component). Used for components in contact with outdoor conditions.
   - InternalWall : (") Used for internal walls and floors. Mind the order of the material layers in your Construction type and the propsBus you will connect to each zone! More detail in Construction types below.
   - SlabOnGround : (") Used for floor constructions in contact with the soil.
   - BoundaryWall : (") Used for walls with known boundary conditions (known temperature or heat flux from the other side). If both use_T_in and use_Q_in are false, it works as an adiabatic wall.
10. Create records of **Materials** (also Insulation material, frame and glazing materials) from `IDEAS.Buildings.Data.Interfaces`. See example materials in `IDEAS.Buildings.Data`.
11. Create records of **Construction types** from `IDEAS.Buildings.Data.Interfaces.Construction` by putting the right materials in order. See example constructions in `IDEAS.Buildings.Data`.
   - Be careful with the order of the materials. In the current implementation the 1st layer corresponds to the outer surface (propsBus_b) of building components, and the last layer to the inner surface (propsBus_a).
   - For thermally activated components the position of the location of the embedded system has to be specified with `locGain` (e.g. `locGain=2` : surface between 2nd and 3rd layer).
12. All **connections** can be implemented manually on the diagram, but can also be written in the Modelica text in the `equation` part. The latter is especially useful when each component is an array of more elements. In case of floor heating or TABS where heat is directly injected in one component, the component in question must be connected to the corresponding `HeatPortEmb`.
13. In the **`SimInfoManager`** of your highest level model you can specify the climate data that you want (to be read from files). By default it is for Uccle, BE. The sub-models will get the required data if they contain a `SimInfoManager` which is specified as `outer` (takes data from outer `SimInfoManager`).
   - `IDEAS.Occupants.Extern.StrobeInfoManager` works similarly, but reads data related to occupant behaviour (to be used with `IDEAS.Occupants.Extern.StROBe` as occupant model). Inputs for this are not generally included (see https://github.com/open-ideas/StROBe), and should be added to the specified folder where Inputs are read (…IDEAS/Inputs).
14. To simulate your model go to the **Simulation** tab. In Setup choose your output file name, time step, solver and other simulation parameters. Make sure you simulate in the directory you want [File-> Change Directory…]. Simulate model! If you're lucky, it will work. Not the first time…