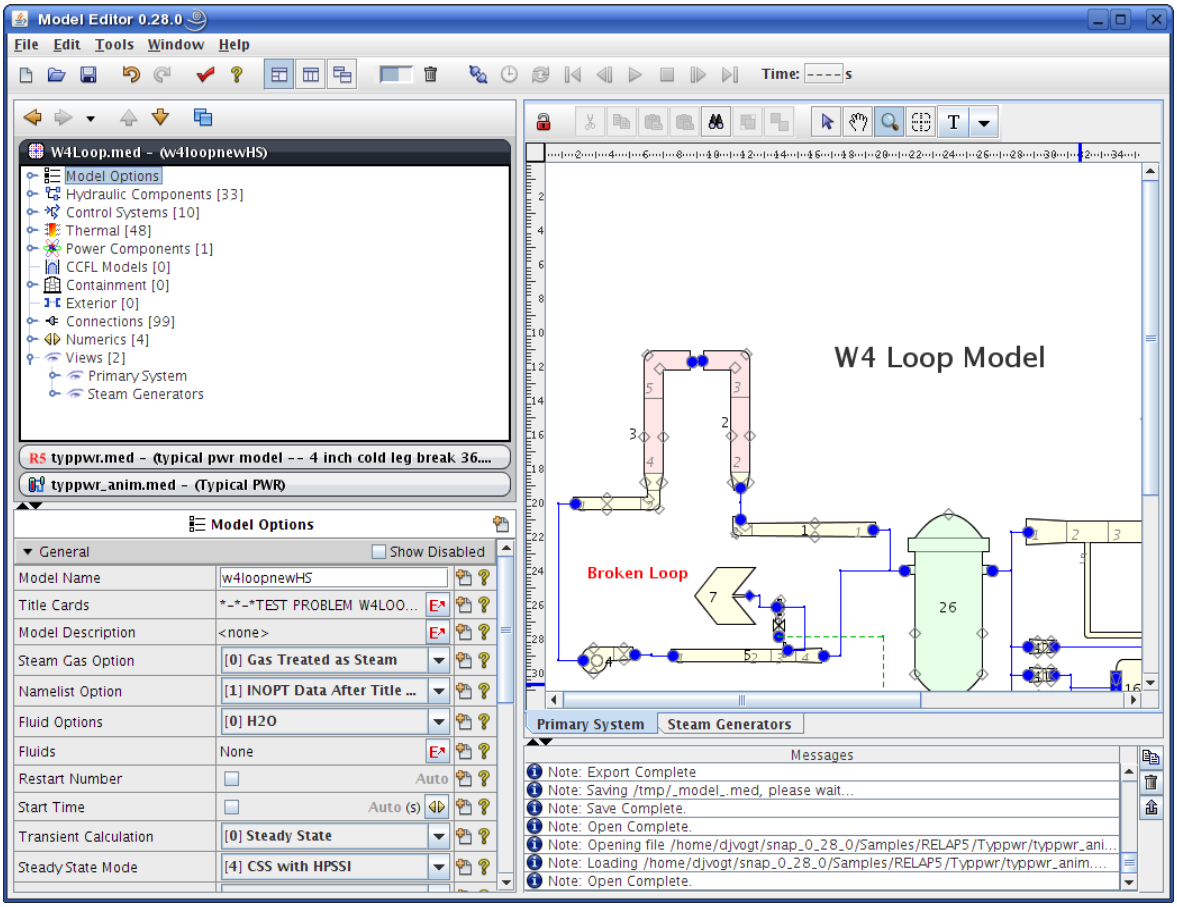
SNAP 0.28.0 User interface Improvements

A number of user-interface improvements have been completed for the 0.28.0 release of the Model Editor. The following image depicts the ModelEditor in the 0.28 series.



Experienced SNAP users will notice a number of user-interface changes. These include the following, along with other changes that may not be immediately apparent:

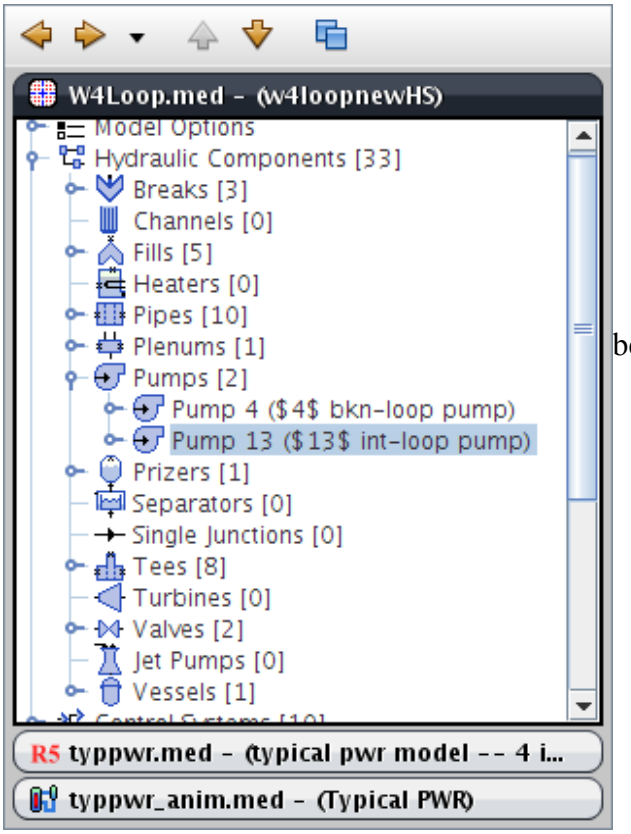
- The Navigator has been redesigned to eliminate the "*current model*" ambiguity. With the new Navigator, only those views associated with the current model are displayed.
- Navigational history is has been added to a new navigation window tool bar.
- A shortcut key / popup menu is available for switching between models eases model selection.
- A Component Differencing utility allows comparing the ASCII output of two component selections.
- Existing windowing modes have been revised and a new window mode introduced. Switching between different window arrangements is now fast and more intuitive.
- The view-lock button is now a model-wide lock on views. The performance of view-locking has also been greatly improved.

- Various cosmetic adjustments to the SNAP user-interface and its icons have been made to reduce clutter and make SNAP easier to use.
- New user-defined numerics allow additional data types and a wider range of values. The new numerics are currently only available in certain plug-ins.
- Components displayed in the Navigator can now be added to a 2D View via "Drag-n-Drop" of their Navigator nodes.
- The ModelEditor Undo/Redo system was updated to handle each open model separately. The undo/redo buttons will now only undo and redo changes made to the contents of the model currently selected in the Navigator.
- Support was added for Attribute Level Documentation references. This means that each attribute in a model can refer to any number of model notes. Note that though the CAFEAN core now has this support, each plug-in must also be modified before attribute level references will be available for that plug-in.

These UI improvements are discussed in more detail below:

New 'Accordion' Navigator

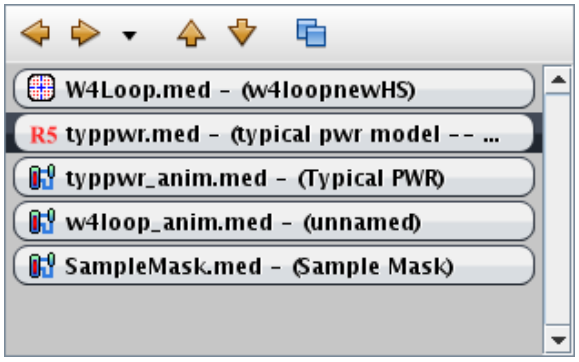
The new Navigator, is designed to minimize any ambiguity as to which open model is the *current model*. This is achieved with an *accordion view*. Each model is represented as a node in the accordion. When a model is selected, the accordion expands the node to display its contents. The expanded contents include the familiar tree-view, sans plug-in and model nodes. Only one model can be expanded at a time: this is always the current model.



In the adjacent image, three models are visible: W4Loop.med, typpwr.med, and typpwr_anim.med. These are not necessarily the only open models. To conserve space, the accordion displays at most two nodes above and two nodes below the expanded model. To view all open models, the expanded node can be collapsed by double clicking it. The expanding and contracting behavior of this component is where the name “accordion” originates.

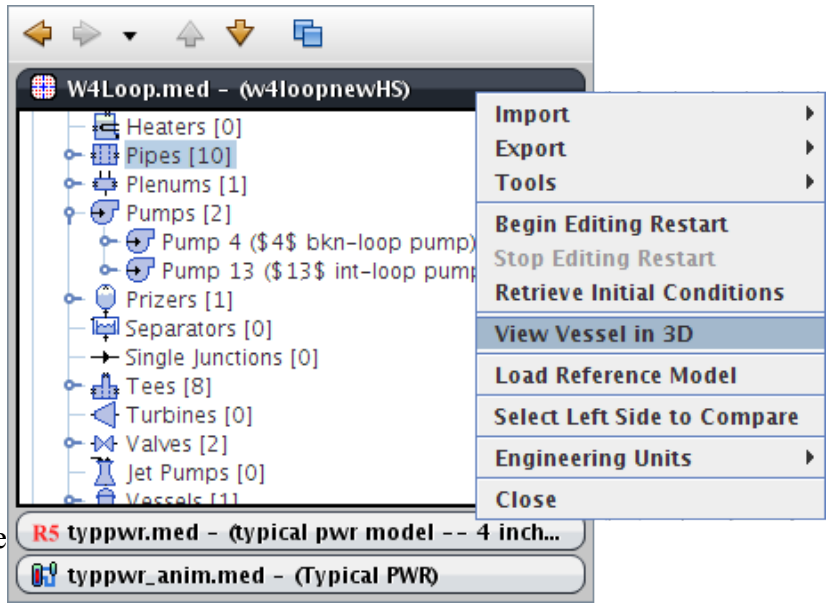
Even when contracted, one model is highlighted as the *current model*. The contracted model remains active as the *current model* until another model is expanded.

Like model nodes in the original, tree-based Navigator, accordion model nodes can be right-clicked to display a pop-up menu of model-wide actions.



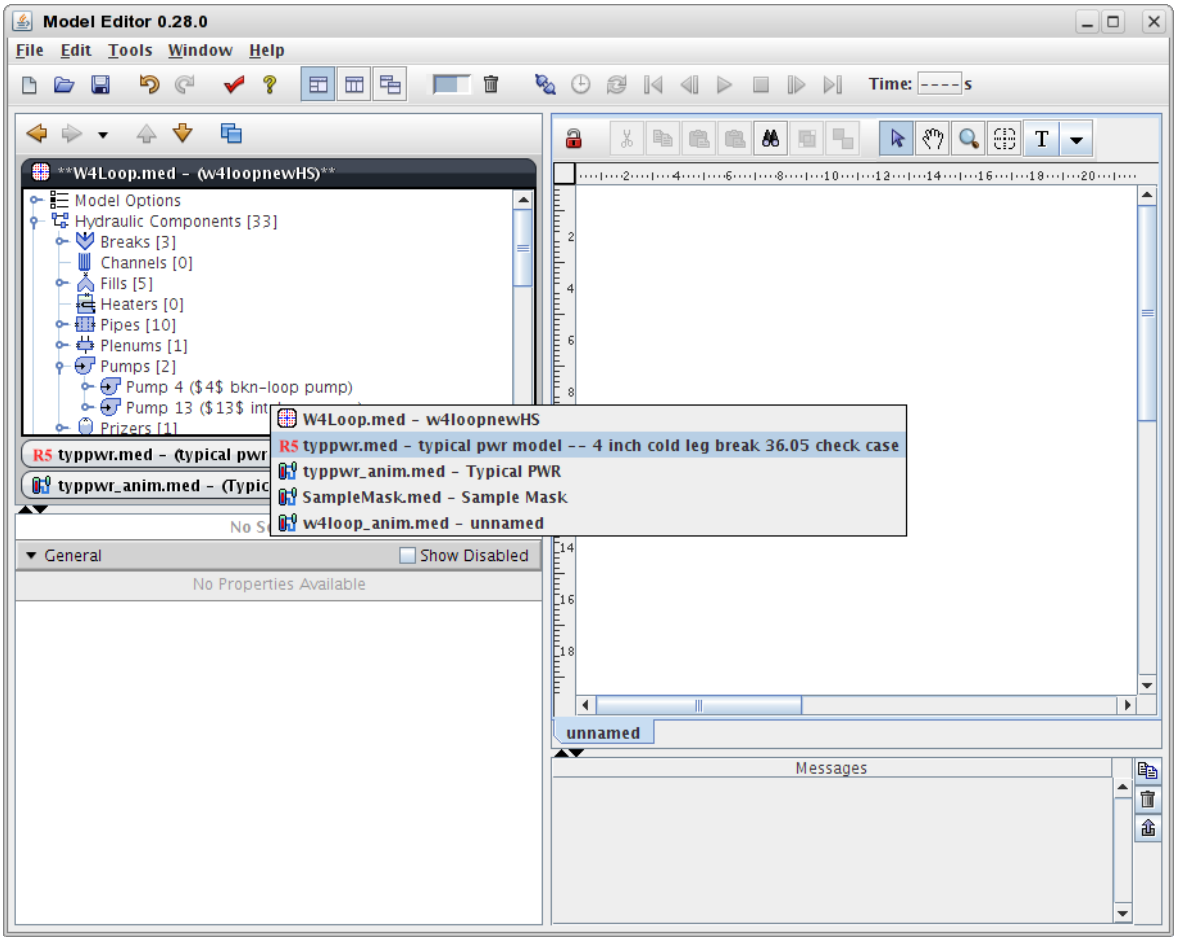
The new Navigator also provides a toolbar for quick access to several useful actions. From left to right, these are:

1. *Back*. Select the previous selection of model contents. Every time the selection changes in the expanded model, the previous selection is stored in a selection history. Pressing the back button restores the first selection in the history, pushing the current selection into the front of a similar forward-selection history.
2. *Forward*. Restores selections placed onto the forward-selection history by restoring old selections.
3. *Selection History*. Allows selecting a specific step in the selection history.
4. *Up* and *Down*. Expands the first model node above or below the current model in the Navigator accordion.
5. *Model List*. Displays a list of available models. Selecting an entry expands that model in the Navigator.



Switching Models

The new Navigator places greater emphasis on the current model by hiding the contents of other models. Therefore, switching between models needs to be fast and simple. This is already aided through the single-click to expand behavior of model nodes and various Navigator toolbar actions. To ease model selection even more, a CTRL+TAB shortcut has been added to the model editor. By pressing and holding CTRL, the pressing TAB, a model selection pop-up is displayed over the ModelEditor.



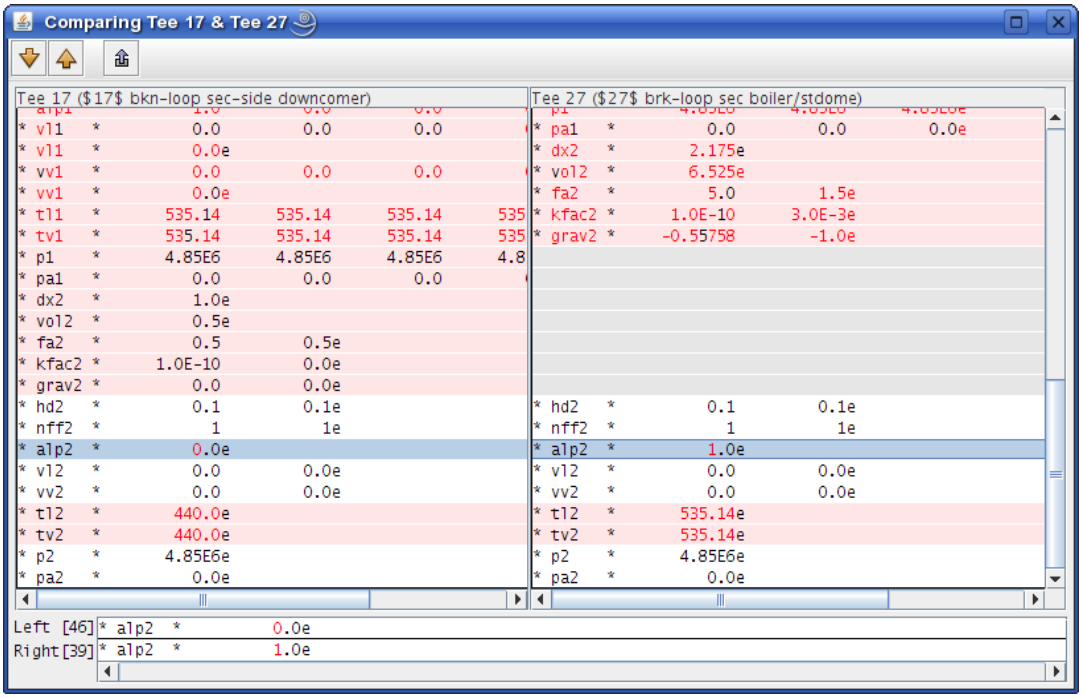
Holding or pressing TAB repeatedly will cycle the highlighted model in the list. Once the desired model is selected, releasing the CTRL key will close the list and expand the chosen model in the Navigator.

This pop-up list displays models in the order they were last selected, with the current model on top. When the pop-up is first displayed, the second entry in the list is highlighted. This allows the two most recent models to be quickly toggled with the CTRL+TAB shortcut.

Component Differencing Utility

The component differencing utility provides a means of comparing the ASCII output of two component selections. Component selections can vary in size, ranging from a single component, to entire models, where all components of two models can be compared side-by-side.

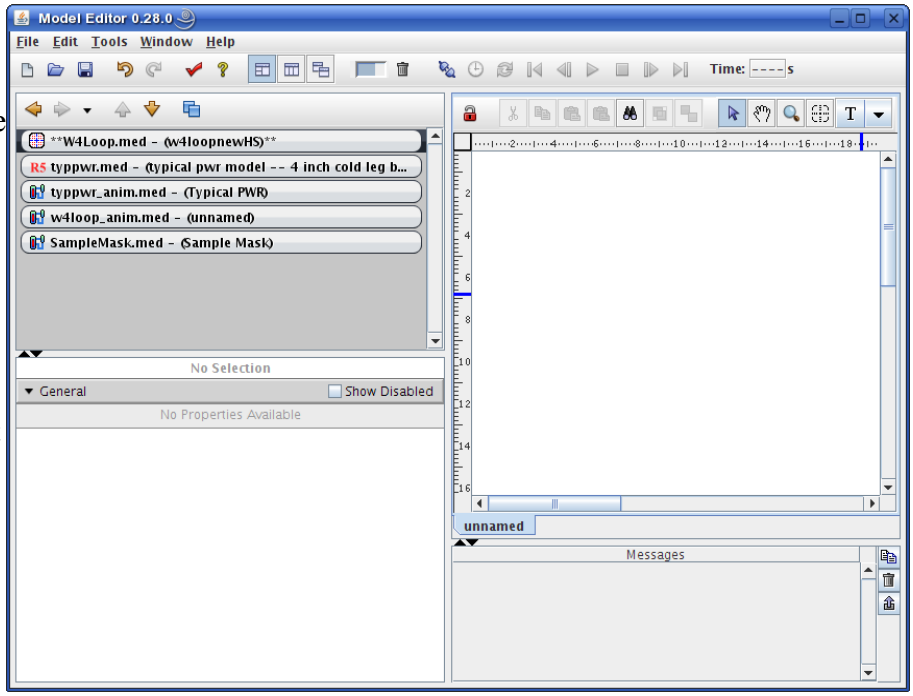
The component differencing utility provides two methods for comparing selections of single components; the target components view menu, and the right-click menu of a component's navigator node. Choosing the *Select Left Side to Compare* menu item adds the selected component to the left side of a difference viewer. If a component has already been added to the left side of a comparison, the *Compare to...* menu item will add the currently selected component to the right side of a difference viewer. The differences between the ASCII output of the components will be calculated, and a new Component Difference Viewer will be displayed.



Components selected on the right side of the component differencing tool implement SNAP's component listener interface so that modifications to the component will be automatically reflected in the tool. This feature can be used to monitor the changes to the ASCII output of a component as it is being modified.

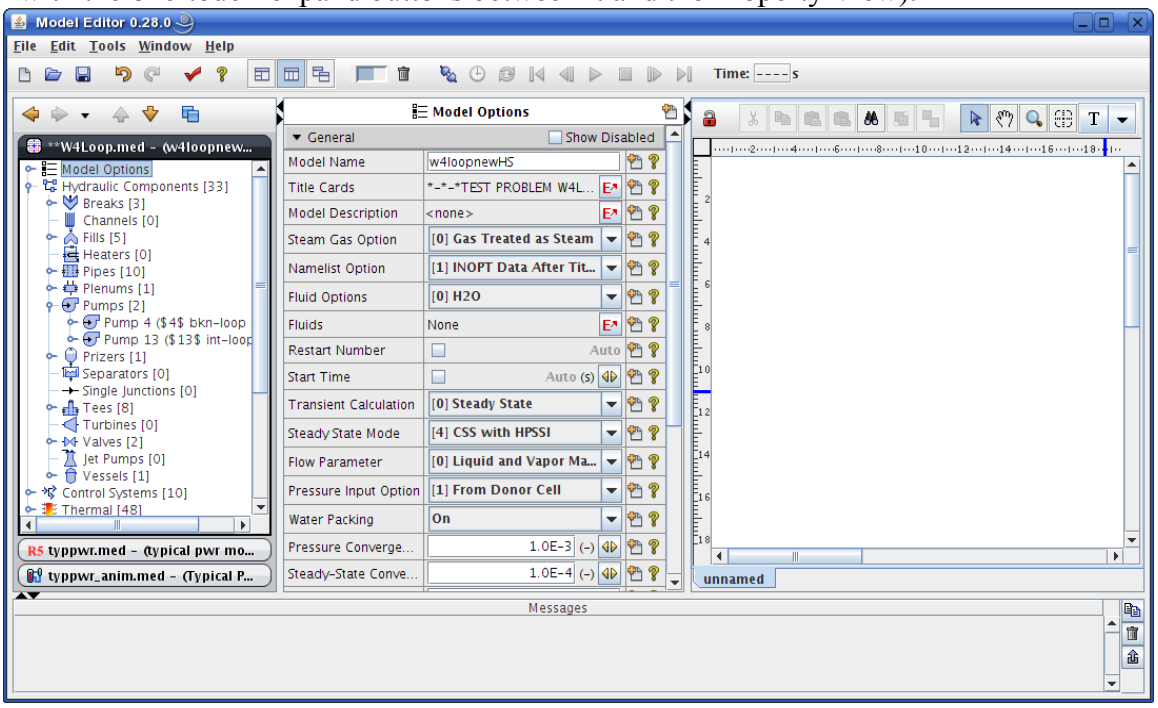
Window Modes

The window modes previously available through the ModelEditor preferences have been revised. Now, three toggle buttons in the main toolbar can be used to select one of the three available window modes.

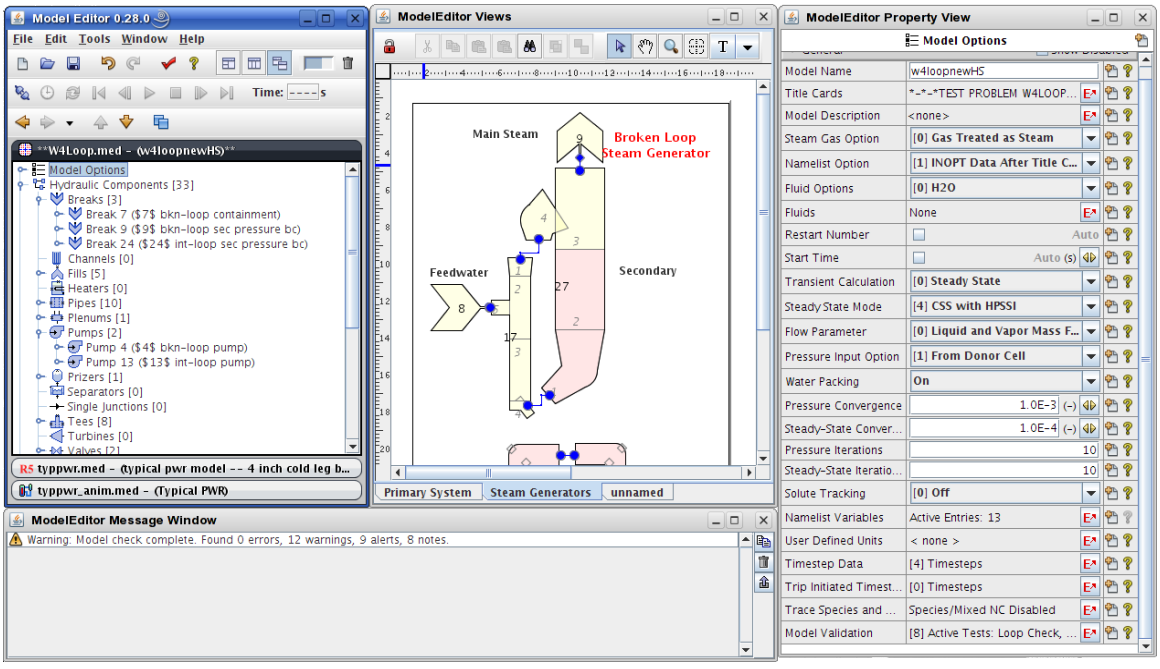


The first window mode places the Navigator above the Property View. This is similar the previous default ModelEditor layout, except that the Property View extends past the Message Window. This mode is ideal for standard 4:3 monitors and those that prefer the legacy layout.

The second mode places the Navigator, Property View, and View area in columns above the Message Window. This mode is ideal for widescreen monitors and plug-ins without views (the View area can be hidden with the one-touch-expand buttons between it and the Property View).

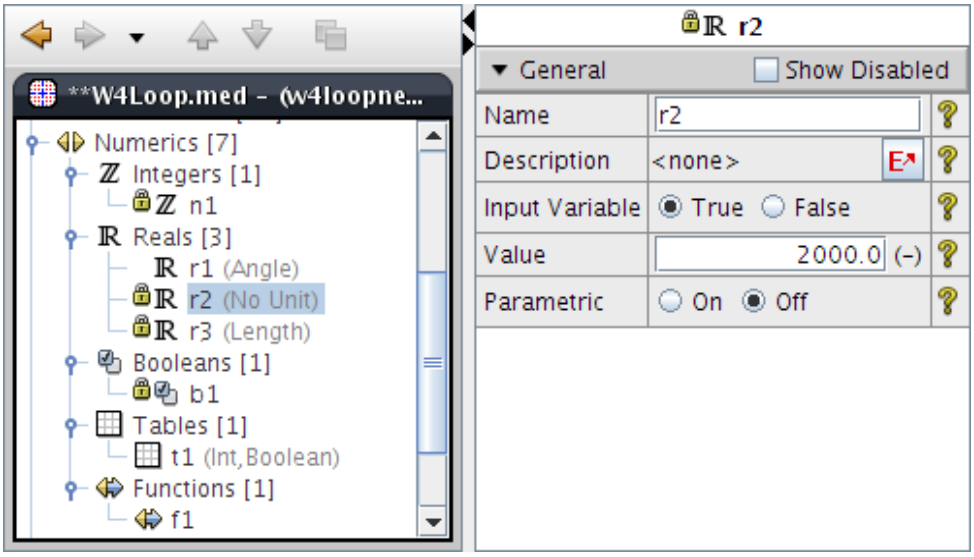


The third mode places each major component in a separate window to be arranged as seen fit by the user. This replaces the Multi-Window mode from previous versions of the ModelEditor. Unlike the original multi-window mode, views stay docked in a central View area unless manually undocked. Furthermore, saving a model after docking or undocking a window will store the view's docked status, which will be restored the next time the model is opened.



New Numerics

The user-defined numerics have been significantly overhauled to provide greater functionality. In addition to real numbers, numerics can be defined for integers, booleans, and tables. Of special note is the lack of limitation on the value of real-number numerics: a change in the internal implementation of reals allows any number to be specified, including values below $-1e^{30}$. Previously, numbers below $-1e^{30}$ were reserved for use as encoded references to user defined numerics.



The new Table numeric is defined by a set of column types. Once set, it may be populated with rows of data. As specific tables are identified in SNAP plug-ins as being reference safe, they will modified to allow referencing table numerics as their contents.

Reals, booleans, and integers are all *single-value* numerics: they can be set as *parametric* for use in parametric export and submit. For reals and integers, parametric functions identically to User Defined Constants in older SNAP versions. For booleans, this means that each export or submit occurs twice: one with a true, one with a false.

Unlike the old system, numerics are not broken down into constant and variable categories. Instead, single-value numerics can set an Input Variable flag. When set to *true*, the value cannot be modified in a Python function. When set to *false*, the numeric's *Initial Value* field is copied into the *Value* field at the start of Python evaluation, and then may be modified by functions as needed. *Input variables* are marked by a small lock icon in the Navigator.

Functions also have new semantics. Like the older numerics, functions are a mix of Python and custom functions written for the ModelEditor. Those functions are listed below:

- *SetVariable(name, value)* - sets the *value* field of a variable real, integer, or boolean numeric. This method raises an error if the name cannot be resolved or the value is marked as an *input variable*.
- *GetVariable(name)* - returns the *value* field of a real, integer, or boolean numeric. This method raises an error if the name cannot be resolved.
- *GetTable(name)* - returns a table object representing table numeric with the given name. This

method raises an error if the name cannot be resolved. Table objects have the following methods:

- *GetColumnCount()* - returns the number of columns in the table.
- *GetRowCount()* - returns the number of rows in the table.
- *GetValueAt(row, col)* - returns the value at the given coordinate. This method raises an error if the coordinate is invalid.
- *SetValueAt(row, col, value)* - sets the value at the given coordinate. This method raises an error if the coordinate is invalid. Otherwise, it makes its best attempt to assign a value based on the type of the column and the type of the value. For reals and integers, this typically means coercing the data type to a larger or smaller type. For booleans, this means setting the value to true if it equals 1, and false for all other values.