# SIMPACK — Tips and Tricks

## SIMPACK Scripting: Re-using Scripts, New Widgets and Signals

**SIMPACK's JavaScript based scripting is a powerful tool for adapting and expanding the SIMPACK PostProcessor as well as for automation of simulation workflows. Besides the base capabilities of JavaScript, SIMPACK scripting offers powerful features such as a framework for generating custom GUI dialogs, for accessing files and directories, and for launching and controlling processes.**

*"...SIMPACK scripting offers powerful features such as a framework for generating custom GUI dialogs..."*

**Another valuable but less well known feature of SIMPACK scripting is its capability to reuse existing script code. This is particularly useful if a set of scripts or script functions exist that are being re-used in different scripting projects. This feature allows the sharing of scripts between different users throughout a company. This is ideal for situations where one user creates and maintains a set of scripts that should be used but not changed by other users. The `Script.include()` method serves for re-using scripts.**

### TWO WAYS TO INCLUDE SCRIPTS

Script code can be loaded into a script either from an external script file or as a string.

### INCLUDING SCRIPTS FROM FILES

Loading scripts from a file is done by using the `Script.include()` method where a filename is given as a parameter. The scripting engine then searches for the respective script file in the script search path. Once the file is found, it is loaded and the definitions and functions in that script file can be accessed as with any other script code. The scripts are searched for in a set of search paths.
These paths can be set and manipulated with the scripting methods: `addSearchDir()`, `insertSearchDir()`, `removeSearchDir()` and `ResetSearchPath()`.
The default search path points to the directories $SIMPACK/run/scripts and $SIMPACK_MODEL/dat/scripts.
The following statement shows how to include the script file "myFile.qs" from the script search path:

```
Script.include("myFile.qs");
```

After this state-ment, all methods and functions defined in "myFile.qs" can be accessed. Please note that any global script code, i.e. code that is not part of a method or function, is executed immediately once the script is included.

### INCLUDING SCRIPTS FROM A STRING

Loading script code from a string is particularly useful if you want to create script code on the fly that needs to be executed by the script currently running. The following shows how to include script code from a string:

```
Script.include("function myName()
  {print("Hello SIMPACK");}",
  "myContext", true);
```

Once this command is executed, a new function `myName()` is known to the scripting engine and can be called from scripts by executing `myName()`.

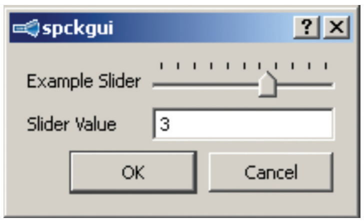
**New widgets (GUI elements) and signals have arrived in the SIMPACK Scripting engine. These serve for creating more powerful dialogs.**

### NEW WIDGET SLIDER FOR ENTERING AND CHANGING VALUES

With SIMPACK 8902 a new GUI widget has arrived in the scripting engine, the Slider.
The Slider is ideally suited for letting the user dynamically change a number value and immediately see the results (see Fig. 1).

### SIGNALS ADD IMMEDIATE CONTROL

Not only was the slider added but also new Signals were made available. Signals are sent out by the scripting engine when certain events occur in a scripted GUI. An example for such an event would be the dragging of a slider to a new position. To make the scripted GUIs more dynamic, Signals were added to the Slider widget, the LineEdit and the NumberEdit. With these new Signals, it is now possible to find out about and immediately react to input of the user on any of these widgets. In this way, highly dynamic GUIs can be created. Previously such signals were only available for the PushButton widget.
Please see the SIMPACK Scripting documentation for details and examples about these functionalities.



```
function sliderDemo()
{
  var myDiag=new Dialog();
  var mySlider=new Slider();

  myDiag.add(mySlider);

  mySlider.label="Example Slider"
  // set min, max and current value
  mySlider.min=-10;
  mySlider.max= 10;
  mySlider.value=3;
  // enable ticks
  mySlider.tickPosition=1;
  mySlider.tickIntervall=2;

  // add NumberEdit to dialog to
  // display current value of slider
  var myNumberEdit=new NumberEdit();
  myNumberEdit.label="Slider Value";
  myNumberEdit.value=mySlider.value;
  myDiag.add(myNumberEdit);

  // connect the sliderChanged
  // signal with the NumberEdit
  connect(mySlider , "valueChanged()",
  mySliderFunction);

  // show dialog with slider
  myDiag.exec()

  // the function that gets called
  // each time the slider is changed
  function mySliderFunction()
  {
    myNumberEdit.value=mySlider.value;
  }
}
```

*Fig. 1: Slider widget and NumberEdit connected via valueChanged signal*

**For more 'SIMPACK — Tips and Tricks' please see our website under: www.simpack.com/simpack_news_by_industrialsector.html**