Actions Tech. Note 5

Draft 1.0

Custom Slave Functions

Many of the Freeway actions that are supplied with Freeway can be combined so that they work as Triggers and Slaves. This method of communication is open and as an action writer you can write actions that are custom slave functions - that is you can write your own slaves that will work with Freeway's Trigger functions.

How Slaves Work

If you apply a slave image to a graphic JavaScript of the following pattern is generated

```
<script language="javascript"><!--

function FWImage_Indigo(num,msg)
{
  if (num==1)
    FWHitSwap('img1',msg,'Resources/untitl1a.gif','Resources/untitl1.gif');
}

window.FT_Indigo=new Object;
window.FT_Indigo['0']=FWImage_Indigo;

//-->
</script>
```

Code of this pattern is generated for all slave objects. The code falls into two parts

a) **Custom Slave Function.** In this case it is `FWImage_Indigo`.
b) **Slave Group List**. In this example it is `window.FT_Indigo`.

The way that the slaves work is that there is a slave group list for each group of all the functions that are to be called when a particular slave group is triggered. These functions will be called one after the other.

Your Custom Slave Function should have two parameters:

`num` - the slave number that is triggered, and the other is for the
`msg` - what the trigger is 1 is an activate (typically a mouse entering or clicking) 0 is a deactivate (typically a mouse leaving)

It is up to you how you respond to the values in those parameters.

If you wanted to write a trivial slave that would post an alert whenever a slave is triggered you would have to write a new Custom Slave Function and add it to the Slave Group List. You would require the following HTML to be generated:

```
<script language="javascript"><!--

function FWImage_Indigo(num,msg)
{
  if (num==1)
    FWHitSwap('img1',msg,'Resources/untitl1a.gif','Resources/untitl1.gif');
}

function MyCustomFunction(num,msg)
{
  alert("Hello");
}

window.FT_Indigo=new Object;
window.FT_Indigo['0']=FWImage_Indigo;
window.FT_Indigo['1']=MyCustomFunction;

//-->
</script>
```

How to Generate the Code

Generating the Custom Slave Function is pretty much up to you, the actions writer. You will have to add the code to the page in the normal way using `fwAddRaw`, `fwAddRawOpt` etc. The following fragment shows a simple way that you might do this:

```
<action-markup custom name="MyCustomFunction">
function MyCustomFunction(num,msg)
{
  alert("Hello");
}
</action-markup>

<action-javascript>

  function fwBeforeEndBody()
  {
    var bodyTag = fwDocument.fwTags.fwFind("body");
    if (bodyTag)
    {
      var javascript = bodyTag.fwAddJavaScript();
      javascript.fwAddRawOpt(fwMarkups["MyCustomFunction"]);
    }
  }

</action-javascript>
```

Generating the Slave Group List is more complex it falls into two parts.

a) Constructing a list of all the Custom Slave Functions that will be in the Slave Group List.
b) Generating the Slave Group List in the output.

Many Actions may want to add items to the slave group list so there is a protocol by which this is managed.

The convention there is an internal object attached to the page for each slave group list. (for the slave group indigo it is `fwPage.fwIndigo`).

In the `fwBeforeEndBody` phase of code generation all the actions that want to have function that they wish to be added to the Slave Group List add their function to the list. So during the `fwBeforeEndBody` phase of the page an internal list of all the Custom Slave Functions is generated. The following function will do this for you:

```
// This adds a function call to the SlaveGroupList so that it will be
// called in response to messages on that group
// call this in fwBeforeEndBody
function AddFunctionToSlaveGroupList(slaveGroup, functionName)
{
  if (!fwPage["fw"+slaveGroup])
    fwPage["fw"+slaveGroup] = new Array;
  fwPage["fw"+slaveGroup][functionName]=functionName;
}
```

In the `fwAfterEndBody` phase of the code generation the list is generated. The question is "who generates the list" If you are the only slave the answer is easy - you do. But if there is another slave on the page who should do it then. The way that this is managed is that everyone attempts to generate the list. But if you are the action that does generate the list you clear the internal list so that nobody else can do it. It is important to bear this in mind when you do the testing of your action.

The following function will generate code

```
// This generates the slave group function list for particular
// slave group & append it to the bodyTag
// call this in fwAfterEndBody
function GenerateSlaveGroupList(bodyTag, slaveGroup)
{
  // get the javascript area
  var javascript = bodyTag.fwAddJavaScript();

  // is there a slaveGroup list?
  var theGroupList = fwPage["fw"+slaveGroup];
  if (theGroupList)
  {

    // declare the slaveGroup list object - this is not an array in order to
    // make it compatable with vanilla versions of the browser

    javascript.fwAddRawOpt("window.FT_",slaveGroup,"=new Object;");
    var itemNumber = 0;
    for (var i in theGroupList)
    {
      javascript.fwAddRawOpt("window.FT_",slaveGroup,"['",itemNumber,"']=",
        theGroupList[i],";");
      itemNumber++;
    }

    // clear the slaveGroup list so that no other action can write it
    fwPage["fw"+slaveGroup] = null;
  }
}
```

So the following Action will generate an alert that will post an alert each time a slave gets a message on the slave group "Indigo":

```
<action! ------------------------------- Test Slave/>
<page-action name="Test Slave">

<action-markup custom name="MyCustomFunction">
function MyCustomFunction(num,msg)
{
  alert("Hello");
}
</action-markup>
```

```
<action-javascript>

  // This adds a function call to the SlaveGroupList so that it will be
  // called in response to messages on that group
  // call this in fwBeforeEndBody
  function AddFunctionToSlaveGroupList(slaveGroup, functionName)
  {
    if (!fwPage["fw"+slaveGroup])
      fwPage["fw"+slaveGroup] = new Array;
    fwPage["fw"+slaveGroup][functionName]=functionName;
  }

  // This generates the slave group function list for particular
  // slave group & append it to the bodyTag
  // call this in fwAfterEndBody
  function GenerateSlaveGroupList(bodyTag, slaveGroup)
  {
    // get the javascript area
    var javascript = bodyTag.fwAddJavaScript();

    // is there a slaveGroup list?
    var theGroupList = fwPage["fw"+slaveGroup];
    if (theGroupList)
    {

      // declare the slaveGroup list object - this is not an array in order to
      // make it compatable with vanilla versions of the browser
      javascript.fwAddRawOpt("window.FT_",slaveGroup,"=new Object;");
      var itemNumber = 0;
      for (var i in theGroupList)
      {
        javascript.fwAddRawOpt("window.FT_",slaveGroup,"['",itemNumber,"']=",
          theGroupList[i],";");
        itemNumber++;
      }

      // clear the slaveGroup list so that no other action can write it
      fwPage["fw"+slaveGroup] = null;
    }
  }

  function fwBeforeEndBody()
  {
    var bodyTag = fwDocument.fwTags.fwFind("body");
    if (bodyTag)
    {
      var javascript = bodyTag.fwAddJavaScript();
      javascript.fwAddRawOpt(fwMarkups["MyCustomFunction"]);
      AddFunctionToSlaveGroupList("Indigo", "MyCustomFunction");
    }
  }

  function fwAfterEndBody()
  {
    var bodyTag = fwDocument.fwTags.fwFind("body");
    if (bodyTag)
      GenerateSlaveGroupList(bodyTag, "Indigo");
  }

</action-javascript>

</page-action>
```