WS Functional Automation
8.11 Web Server Commands

Purpose

List of the commands that are specific to the WebUI interface.

Most of the commands included in this reference library can be used on multiple interfaces. However, the following commands can only be used with Liquid UI's Web Server and WebUI interfaces. These commands are as follows:

- htmlnode
- usecss
- usejs

We will explain these three commands in the following sections of this document.

htmlnode

The htmlnode command is used to read or to display HTML on SAP screens in WS. This can be very useful when users want to view images in Web Server or implement a Google map, to name two very common uses of the Htmlnode command. The syntax to display or read HTML in WS is as follows. The parameter '101' in the syntax example refers to the type of control - in this case it references an entry field.

```
htmlnode('101',[row,col],"html string");
```

Note: This command is very similar to the view command. However, the htmlnode command produces read-only results, while the view command can produce user-editable files and forms.

Options

The htmlnode command does not take any options.

Examples

To view an image in SAP with htmlnode:

1. Select an image to display. The supported file types are JPGs and GIFs.
2. Go to the Easy Access screen in SAP.
3. Open the 'SAPLSMTR_NAVIGATION.E0100.sjs' script file, and enter the following code. Create the file if it does not already exist.

   ```
   htmlnode(200,[5,70],[10,90],"<image src='./demo_img/branding-
   image.gif' width='301' height='245'>");
   ```

In this example, we will display our image in a window beginning on Row 5, Column 70 and ending on Row 10, Column 90. The image area will be 301 pixels wide and 245 pixels high.

4. Save the changes and refresh the SAP screen. The relevant image will now display on the screen

To view Google maps with htmlnode:

1. Go to the Easy Access screen in SAP.
2. Open the 'SAPLSMTR_NAVIGATION.E0100.sjs' script file, and enter the following code to create a prototype string function. Create the file if it does not already exist.

```
string.prototype.trim = function ( ) {
  return this.replace(/^\s+|\s+$/g,"");
}
```

This function uses regular expressions to strip spaces from any strings.

3. In the 'SAPLSMTR_NAVIGATION.E0100.sjs' script file, create variables to store the values in the SAP fields. An example is shown below:

```
var street;
var district;
var city;
var state;
var country;
var zipcode;
var customer;
```

4. In the 'SAPLSMTR_NAVIGATION.E0100.sjs' script file, use the set command to copy the values from the SAP fields to the GuiXT variables we created in the previous step. An example is shown below:

```
set("V[street]", "&F[Street/House Number]");
set("V[customer]","&F[Customer.text]");
set("V[district]","&F[District]");
set("V[city]","&F[ADDR1_DATA-CITY1]");
set("V[state]","&F[ARRD_DATA-REGION]");
set("V[country]","&F[ADDR1_DATA-COUNTRY]");
set("V[zipcode]","&F[ADDR1_DATA-POST_CODE1]");
```

5. In the 'SAPLSMTR_NAVIGATION.E0100.sjs' script file, generate the iFrame that will go into the HTML node. An example is shown below:

```
var z_sold_to_address = "<iframe scrolling=no width=100%
height=360 frameborder=0 src=\"http://www.agawarehouse.com/tools
/map/?address="
+encodeURIComponent(street.trim( ))+"%20"+encodeURIComponent(dis
trict.trim( ))
+ "%20"+encodeURIComponent(city.trim( ))+ "%20"+encodeURICompone
nt(state.trim( ))
+ "%20"+encodeURIComponent(state.trim( ))+ "%20"+encodeURICompon
ent(country.trim( ))
+ "%20"+encodeURIComponent(zipcode.trim( ))+"\"></iframe>
```

6. In the 'SAPLSMTR_NAVIGATION.E0100.sjs' script file, create the htmlnode. An example is shown below:

   ```
   htmlnode(200,[5,85],[50,140],z_sold_to_address);
   ```

7. Save the changes and refresh the SAP screen. It will now display as shown below:

usecss

Modifying Web Server css files.

The usecss command is used to modify the included CSS format in the WS Web

Server. These modifications are usually for personalizing the WS platform to match a customer's corporate environment, but can be for a variety of purposes. Some possible uses might be to add background image or to specify an external location for a CSS file. In this section we will demonstrate how to use the usecss command.

Options

The usecss command does not take any options.

Adding a background image

To add a background image to the CSS, use the following syntax:

```
usecss("string");
```

The string in the preceding syntax would include the specifications for the background image. For example, if we wished to use the image file 'Synactive.jpg', we would need to add the following code to the config.sjs file in the C: \Program Files\Synactive Inc\GuiXTFuzion directory:

```
usecss("body.urBdyStd {background-
image: url('.demo_images/synactive.jpg'); No-repeat;}");
```

This code will result in a background image being added to the server. CSS files are also maintained in the server.

Specifying an external CSS file

You can also specify an external file to use as the CSS file with GuiXT Web Server. The syntax is as follows:

```
usecss("C:\\Program Files\\Synactive Inc\\GuiXTFuzion\\test.css");
```

GuiXT Web Server will now use the specified file as the source for the CSS.

usejs

Including JavaScript or a file within a script in Web Server.

The usejs command is utilized to include Javascript or a file within a script in WS Web Server. Usejs functions very similarly to the usecss command and is placed in the same config.sjs configuration file as well. The syntax is as follows:

We will explain these three commands in the following sections of this document.

Options

The usejs command does not take any options.

Including a script

The string in the command could include a number of parameters or possibly a short script that would then be incorporated into the WS script. For example, if a user wanted to include code to print out 'Hello World', the following would need to be included in the config.sjs configuration file in the C:\Program Files\Synactive Inc\GuiXTFuzion\ directory.

```
usejs("println('Hello, World')");
```

The body of the action to be performed will be included within the double-quotes. In this case, the included line is for a simple 'println' operation. However, this string could be more complex as well.

Including a file

You can also call a separate Webscript function or even a complete script from an external file. To do this, the syntax would be as follows:

```
usejs("filename");
```

For example, if you want to include a script or function that would be housed externally in the file C:\ExternalScripts\TestFunction.js, you must also include the following code in the config.sjs configuration file:

```
usejs("TestFunction.js");
```

For files housed in the same directory as the scripts, you do not need to include the true or absolute path - the code can find the relevant file as long as it is in the same directory. If the file is housed in a different directory, then it would be necessary to

include the full path to the file, as in the following example:

```
usejs("C:\ScriptFiles\TestFunction.js");
```

This file can contain any content from a single function to a complete script. Once called, the file will run, and then the remainder of the original code will continue from the point where the file left off.

Unique solution ID: #1105
Author: Punil Shah
Last update: 2018-10-29 08:31