Creating Custom Locators for Validations and Extractions

This topic explains how the CreateXPath hook allows you to create custom locators that can be used to identify elements when recording web scenarios tests. These custom locators can then be used for validations and extractions.

Sections include:

About Hooks

Customized hooks can be used to record or modify the values passed at specific points during execution.

Hooks are defined and customized in scripts using custom methods. The same file can define multiple hooks. If you add more than one method to a hook, all methods defined for that hook will be executed when the hook is called.

You can create, apply, and invoke scripts that define hooks in the same way that you create, apply, and invoke any other script in SOAtest: upon startup (only for JavaScript and Jython scripts), by creating and applying an Extension tool, and by adding scripts to a specific path node. You can invoke hooks at different times to elicit the desired functionality. For example, if you want to use a script’s hook functionality for all projects and sites, you could add the JavaScript or Jython script that defines and uses that hook to the `<install_dir>/plugins/com.parasoft.xtest.libs.web_<version>/root/startup` directory; then, any time the program calls the hook, the associated user-defined methods will be executed. The methods will be executed until you call clear () on the hook.

About the CreateXPath Hook

When recording web scenarios, SOAtest identifies the elements that you interacts with by certain attributes of those elements. For example, SOAtest will use the ID of an element, or the text content of an element to locate that element. If you need to extend the attributes used to identify elements during functional testing, you can use the CreateXPath hook.

The CreateXPath hook takes two arguments. The first is an instance of org.w3c.dom. Node, and is the node that the locator will be built for. The second is an instance of org.w3c.dom.Document, and is the document where the node was found. The CreateXPath function should return an XPath that identifies the node in the document, or null if no XPath can be created.

For example, if you want to identify elements by their "class" attribute, you could use the CreateXPath hook to create a custom identifier that checks elements for a "class" attribute and, if it finds one, returns an XPath that identifies the element by the value of the "class" attribute.

When selecting an element to validate, or clicking an element during recording, SOAtest would then use this custom hook. The element would be passed to the custom hook, which would then check the element for a "class" attribute. If it found one, it would return an XPath using the "class" attribute value to identify the element.

Here is a sample script that identifies elements using the "class" attribute and an index (when more than one node has the same "class").

```
app = Packages.com.parasoft.api.Application;
settings = Packages.webtool.browsertest.XPathCreator;

// establish the locator builder priority order
// we use our custom locator after using the id, name and attribute
locators // and before using the position xpath locator
settings.locatorBuildersOrder[0] = settings.ID_LOCATOR;
settings.locatorBuildersOrder[1] = settings.NAME_LOCATOR;
settings.locatorBuildersOrder[2] = settings.ATTRIBUTE_XPATH_LOCATOR;
settings.locatorBuildersOrder[3] = settings.HOOK_LOCATOR;
settings.locatorBuildersOrder[4] = settings.POSITION_XPATH_LOCATOR;

// gets the index of the node out of a list of nodes returned using xpath.
function getIndex(xpath, node, doc) {
  index = -1;
   if (doc) {
    try {
         list = Packages.org.apache.xpath.XPathAPI.selectNodeList(doc,
xpath);
         count = 0;
         while (index == -1 && count < list.getLength()) {
             candidate = list.item(count);
             if (candidate.equals(node)) {
                index = count;
             } else {
                  ++count;
             }
        }
    } catch (e) {}
  }
  return index;
}

// the hook function to link to the CreateXPath hook.
// looks for the class attribute and uses it (along with an index) to
create
// a locator for the given node.
function createXPathHook(node, doc) {
  nodeName = node.getNodeName();
  attributes = node.getAttributes();
  classValue = attributes.getNamedItem("class");
  if (classValue) {
    xpath = "//" + nodeName + "[@"+ classValue + "]";
    index = getIndex(xpath, node, doc);
    if (index != -1) {
          return xpath + "[" + index + "]";
    }
  }
  return null;
}

// sets up the hook
function setHook() {
  hook = app.getHook("CreateXPath");
  hook.set(createXPathHook);
}
```