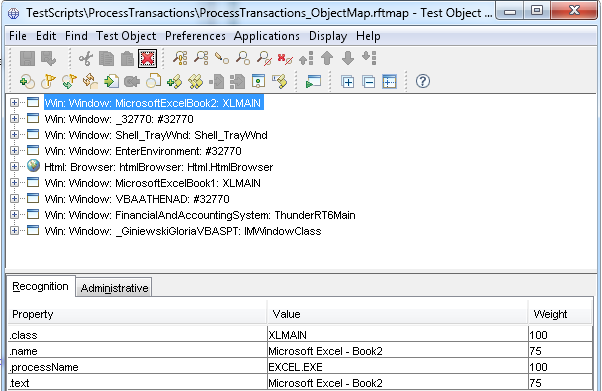
Beaucoup notes on FAS scripting

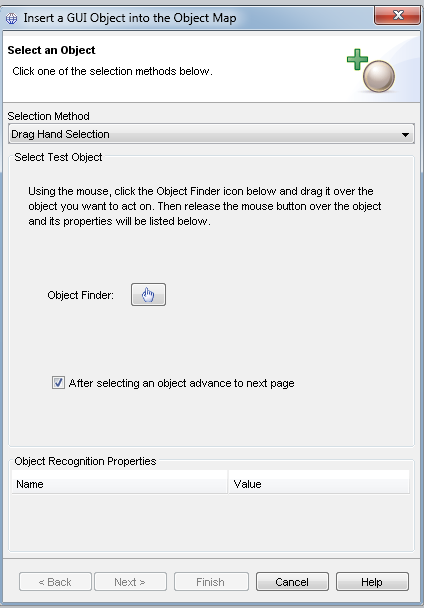
Procedure that seems to work best is one IBM created about 10 years ago, the QSE framework. Here’s the basics:

## Step 1: Import test objects into test map

Double-click the test object map



Click Test Object->Insert Objects…



Click the object finder and highlight the wanted test object

Select “Just the selected object” and click Finish

Back in the Test Object Map, select the Add to Script option from the toolbar

Then exit the object map and say yes to save it.

### Step 2: Creating subroutines

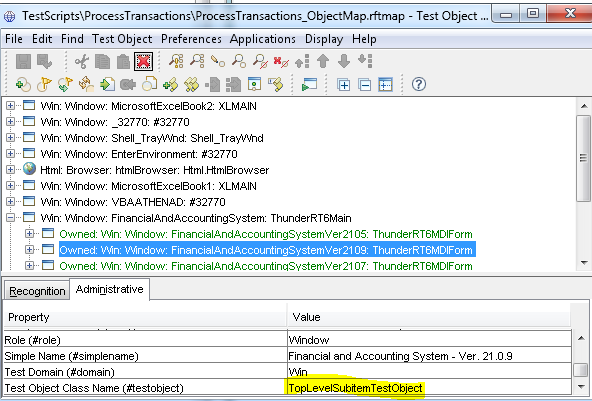
Works best if separate subs are created, although the treeView20WndClasstree() control works better inline.

Example:

public TopLevelGuiSubitemTestObject fas() {

TestObject to=financialAndAccountingSystemVe(ANY,NO\_STATE);

return new TopLevelGuiSubitemTestObject(to); }

The return type comes from looking at the TestObject’s properties in the Administrative tab: 

financialAndAccountingSystemVe is the inserted item

### Step 3: Instantiate program

public static void main[args()] {

SomeAwefullyLongFASProgramName myProg=new SomeAwefullyLongFASProgramName();

### Step 4: using subroutines

Under the instantiation,

myProg.fas().inputChars(“{ENTER}”);

## Other notes

financialAndAccountingSystemVe() can control most of the buttons that are underlined via

fas().inputKeys(“%n”);

financialAndAccountingSystemwi() for pushbuttons

#### Find method

A little slower than most methods but necessary on some screens

RootTestObject root=getRootTestObject();

TestObject[] to=root.find(atDescendant*(“.id”, “desc”),* false);

*((ObjType)* to[0]).*method()*;

Where *.id*  and *desc* come from the TestObject’s properties in Recognition tab.

Can use 2 sets of id and desc with atDescendant. If need more, use atList(atDescendant(“a”,”b”,”c”,”d”), atDescendant(“e”,”f”,”g”,”h”), atDescendant(“j”,”k”,”m”,”n”), atDescendant(“p”,”q”,”r”,”s”));

Can also set up as

RootTestObject root=getRootTestObject();

Property[] props=new Property[4];

props[0]=new Property(“a”,”b”);

props[1]=new Property(“c”,”d”);

props[2]=new Property(“e”,”f”);

props[3]=new Property(“g”,”h”);

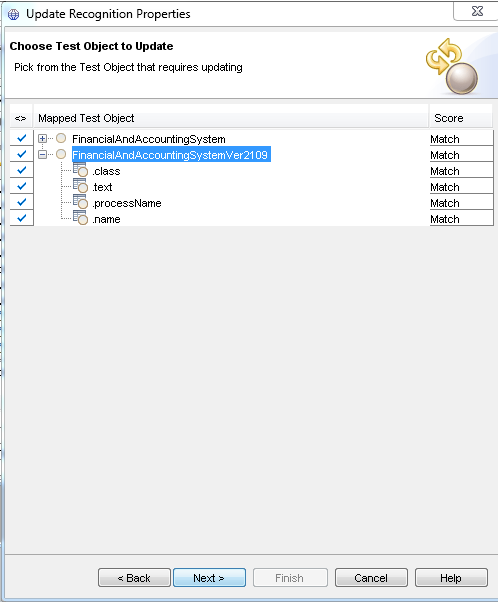
TestObject[] to=root.find(atDescendant(props),false);

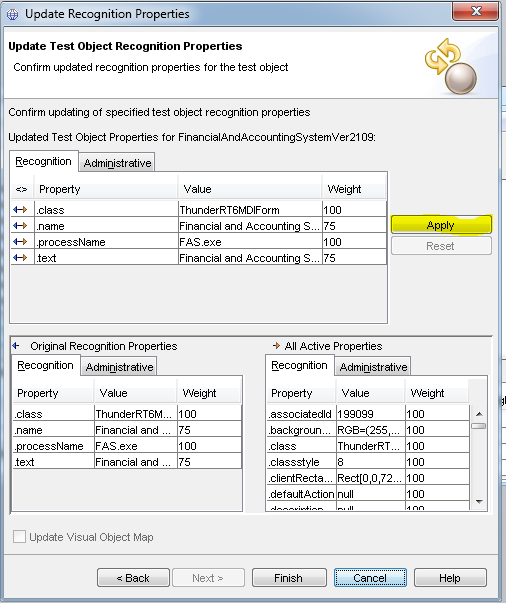
*((ObjType)* to[0]).*method()*;

Where ObjType is the Test Object Class Name (#testobject) in Administrative tab of object map.

Method is chosen from among methods available, usually of the form .click() or .setText()

# Updating items

Usually when they do a version upgrade, scripts using objects with version numbers like financialAndAccountingSystemVe run much slower. To update, double click the object in the object map, then right click it and select update recognition properties. Make sure the item is visible in FAS when you do so. The item you are updating will briefly have a red box around it, then you will get something like the following: 

Click Next, then select Apply

Click finish, then exit out of the box and say yes.

The above is also the best way to see all properties of object.

#### Other stuff

Visual Basic scripts: Runtime.getRuntime.exec(“wscript…”);

In above need to escape all back slashes (http:// becomes [http:////](NULL))

#### Work with dialogs

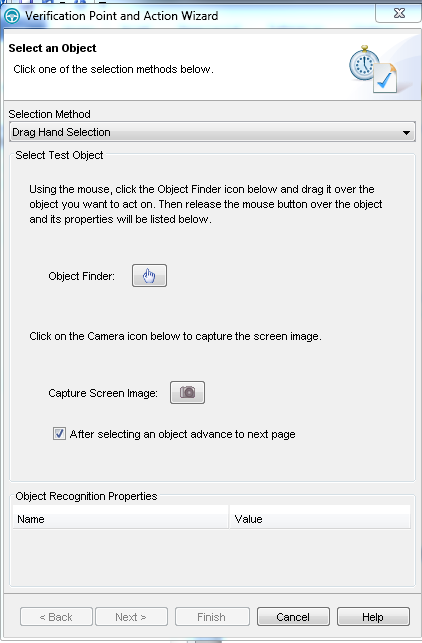
IWindow dlg=*getScreen*().getActiveWindow();

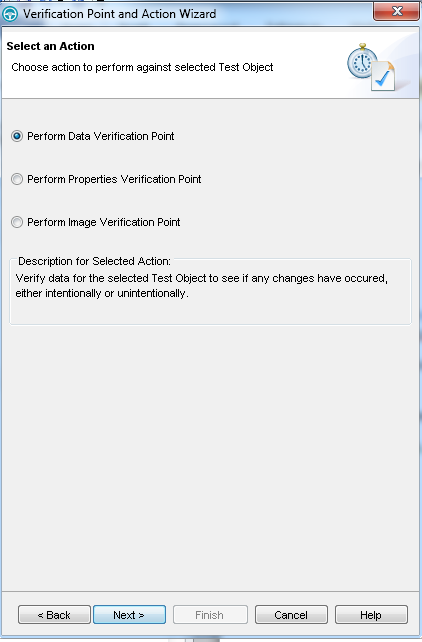
Can then do things like dlg.inputKeys(…);

Or this: *getScreen*().inputKeys("{Num9}{Num9}{Num9}{Num5}{Num5}{Num8}{Num8}{Num8}{Num8}");

May need to be combined with other methods as getScreen does not navigate.

#### Verification Points

At point in script where Verification Point (VP) is needed, ensure first that FAS is in appropriate screen with appropriate items selected, etc. Then in top menu in RFT, click Script->Insert Verification Point: 

Click Object finder and select object (here using the Enter Payments) 

Select appropriate item, next and finish. Data verification is good when verifying selection of items, image verification for some of the obscure items like msflexgrid that RFT cannot “see”.