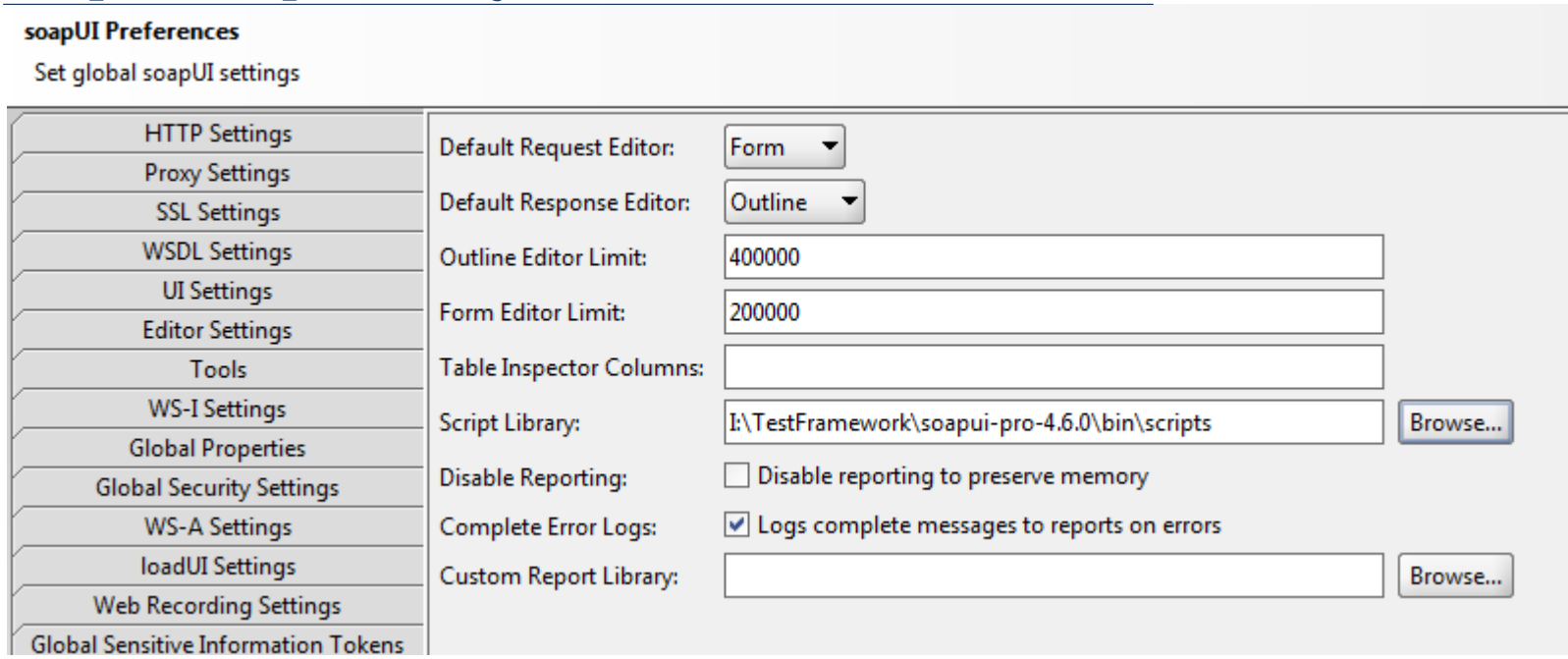
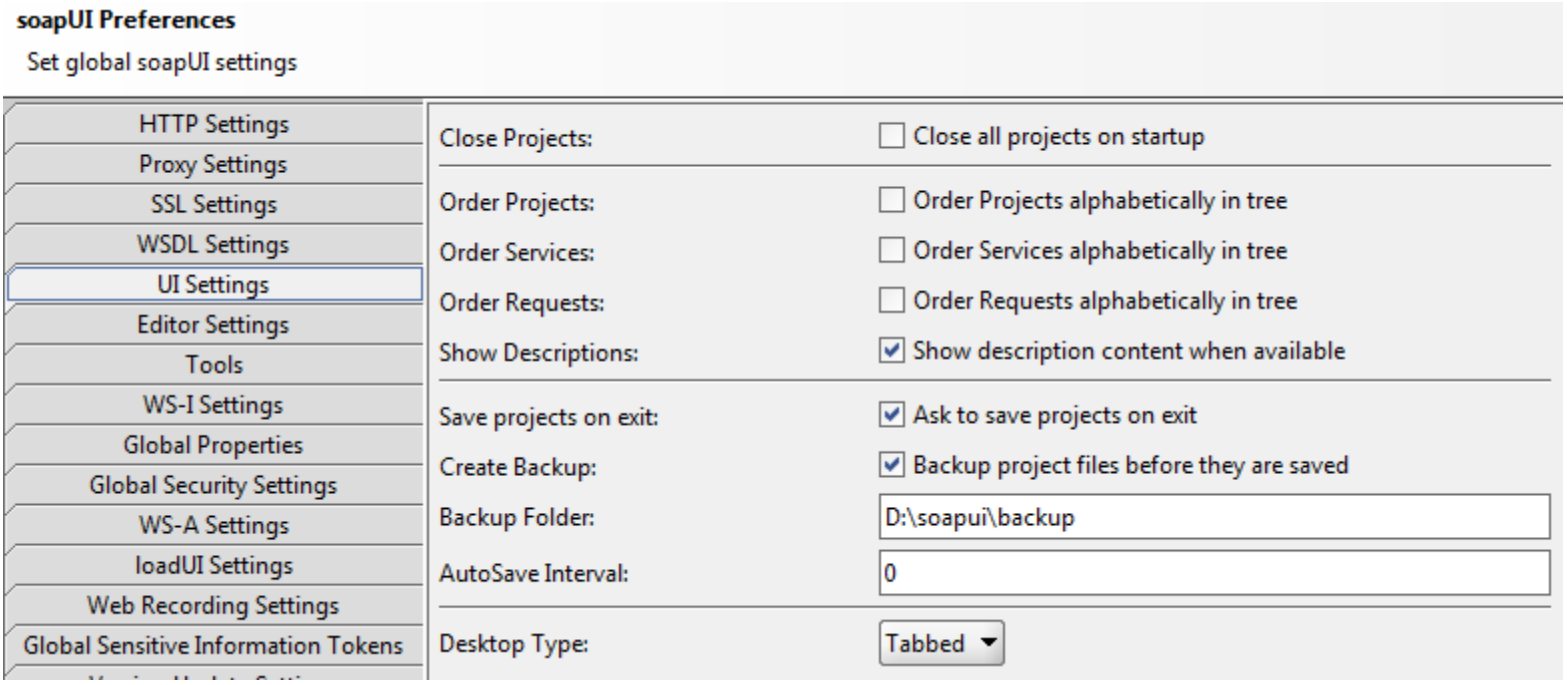
Test Framework Guide

Setup

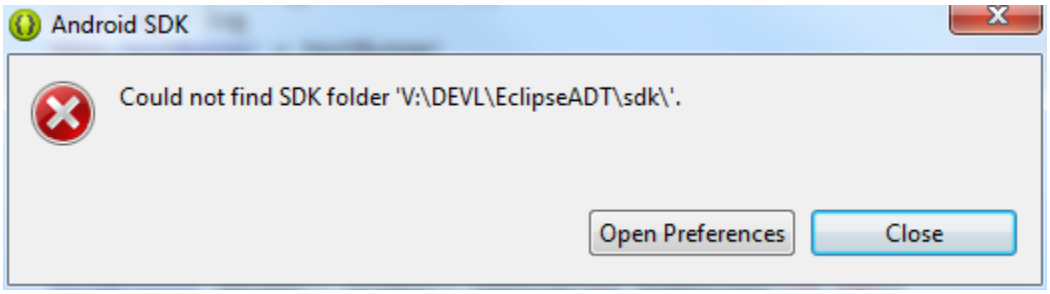
SoapUI Script Library

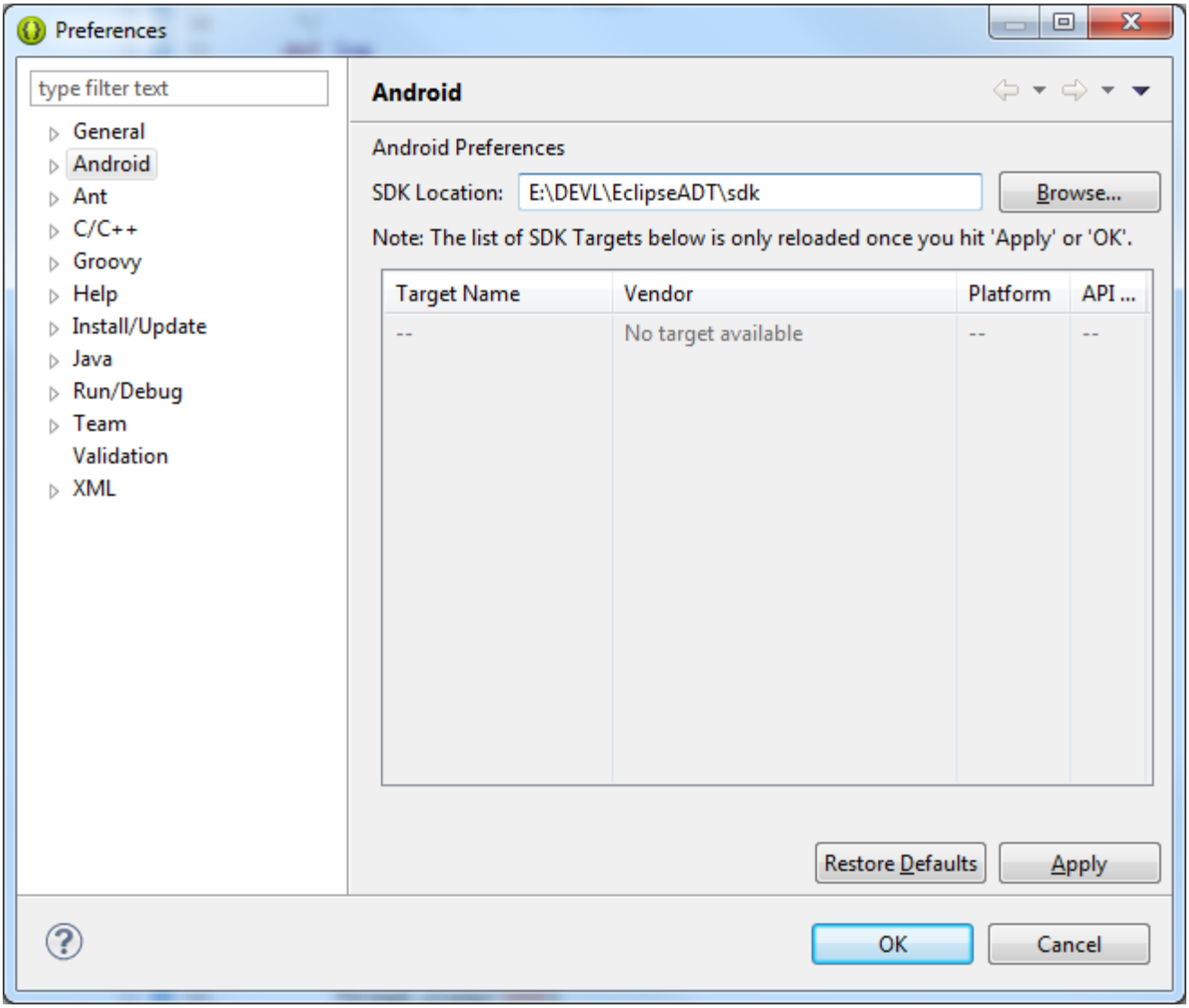


Backup Folder



Eclipse ADT for Android Support



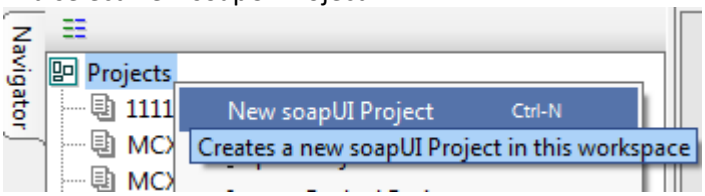


Android SDK/ADB

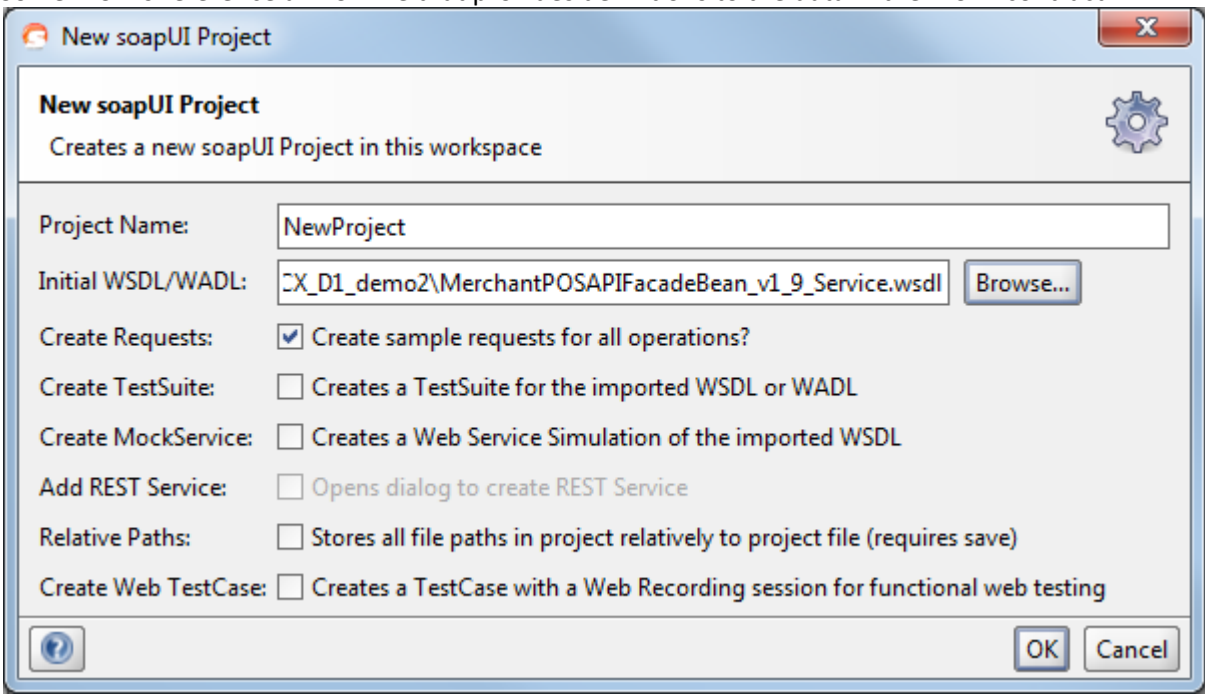
SOAPUI Walkthrough

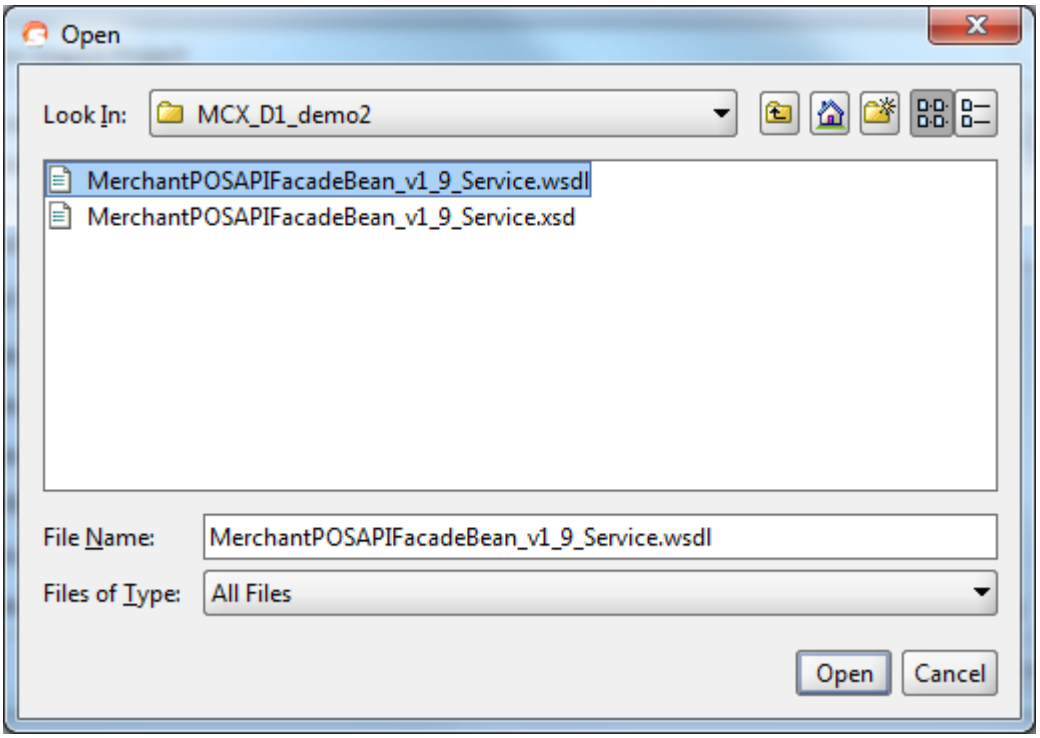
Create a SOAPUI project from WSDL

Go to Projects in the tree
And select New soapUI Project

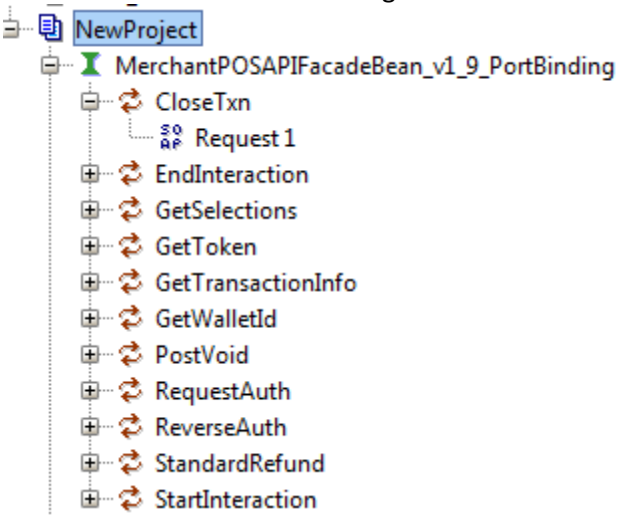


Give the project a name, then click 'Browse' and select the appropriate WSDL file.
Some WSDLs reference an XSD file that provides definitions to the data in the WSDL contract.

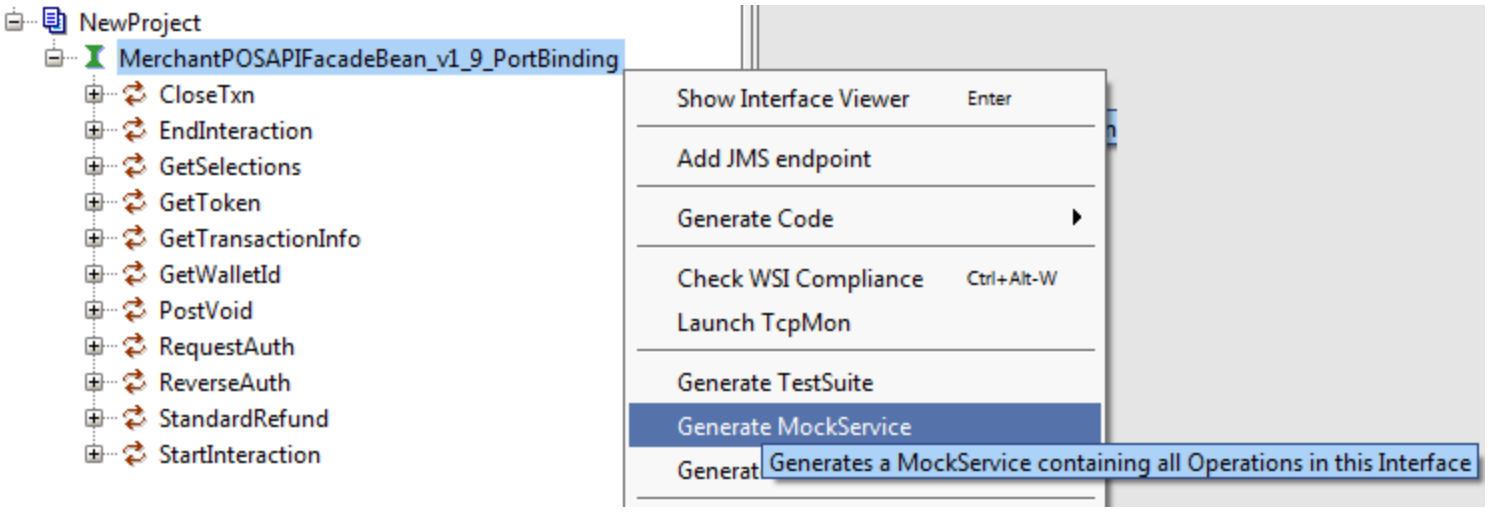




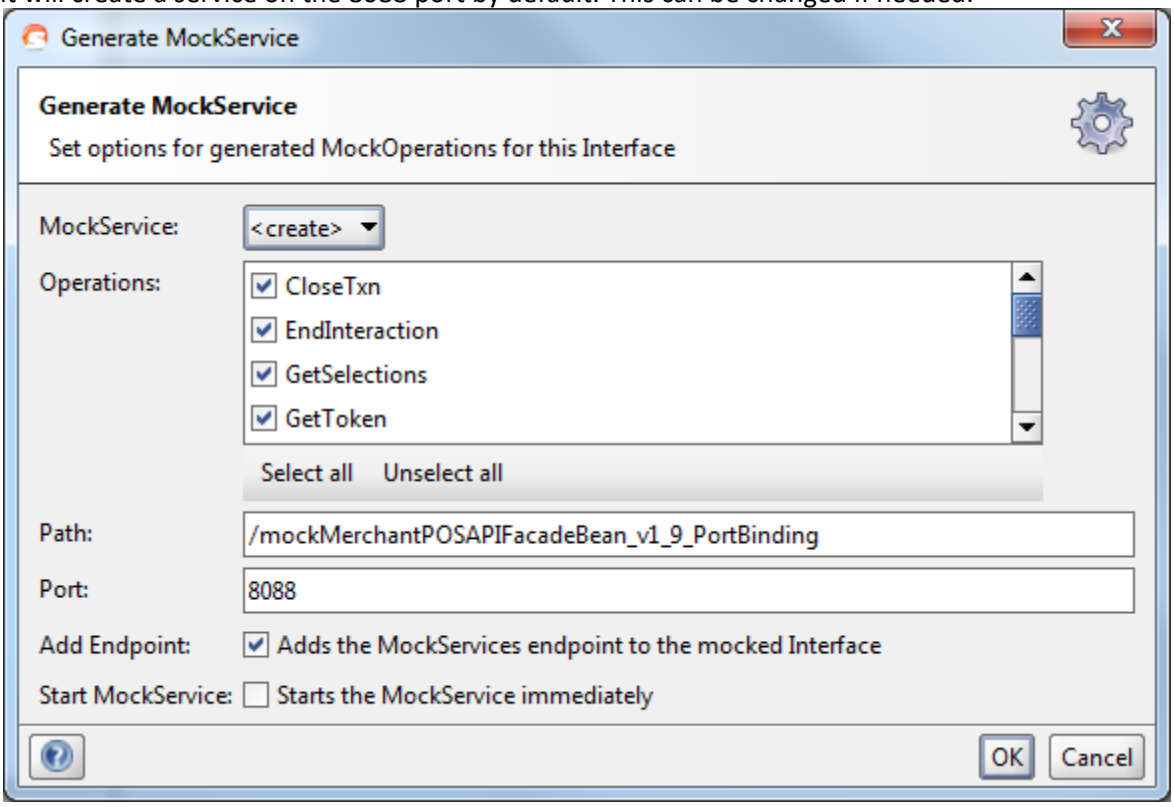
SOAPUI will create a 'PortBinding' branch on the tree with all the requests in the WSDL contract



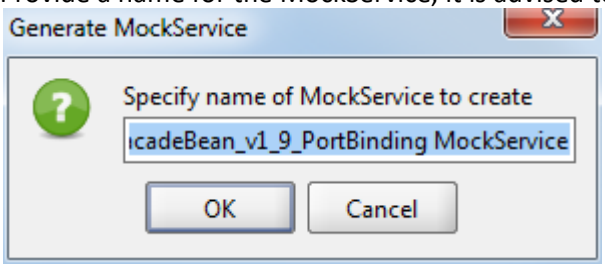
Create and Use a Mock Service

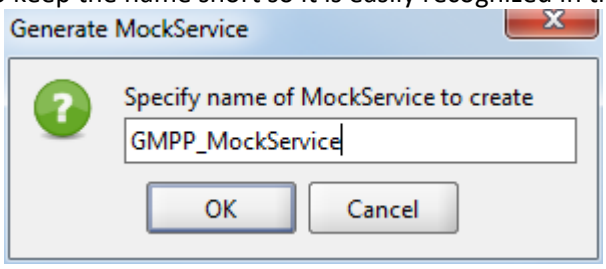


A MockService will be created that contains all the responses in the WSDL contract/PortBinding
It will create a service on the 8088 port by default. This can be changed if needed.

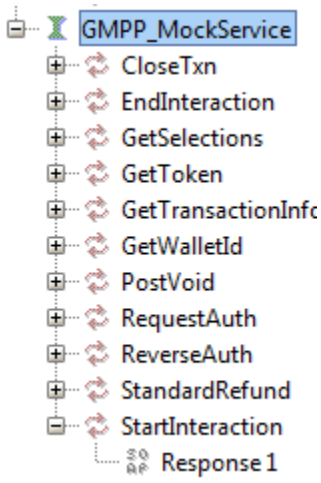


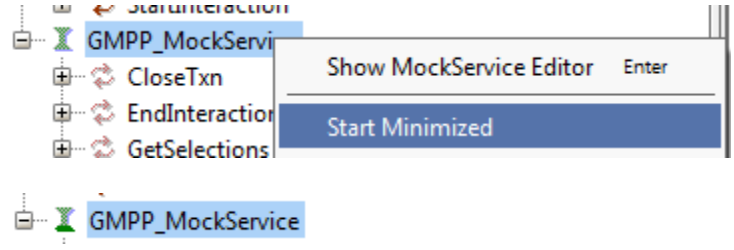
Provide a name for the MockService, it is advised to keep the name short so it is easily recognized in the program





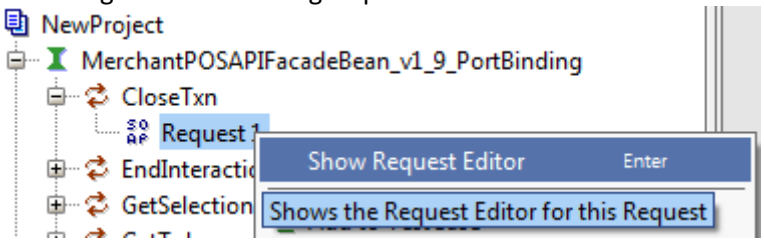
MockService is created but must be started to provide responses

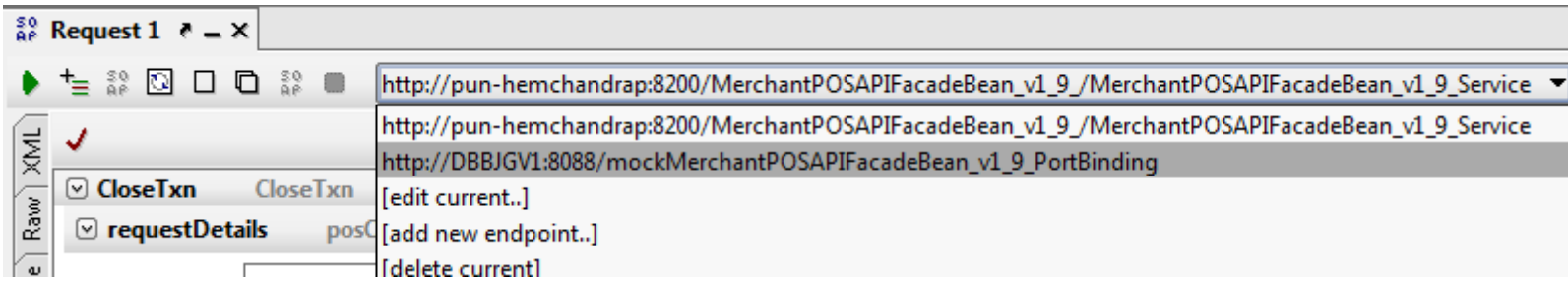




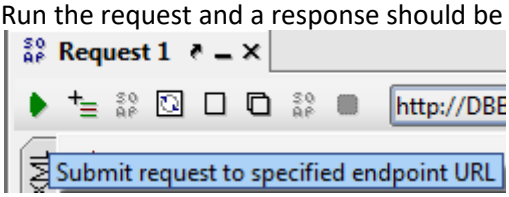
The hour glass will move indicating the service is running

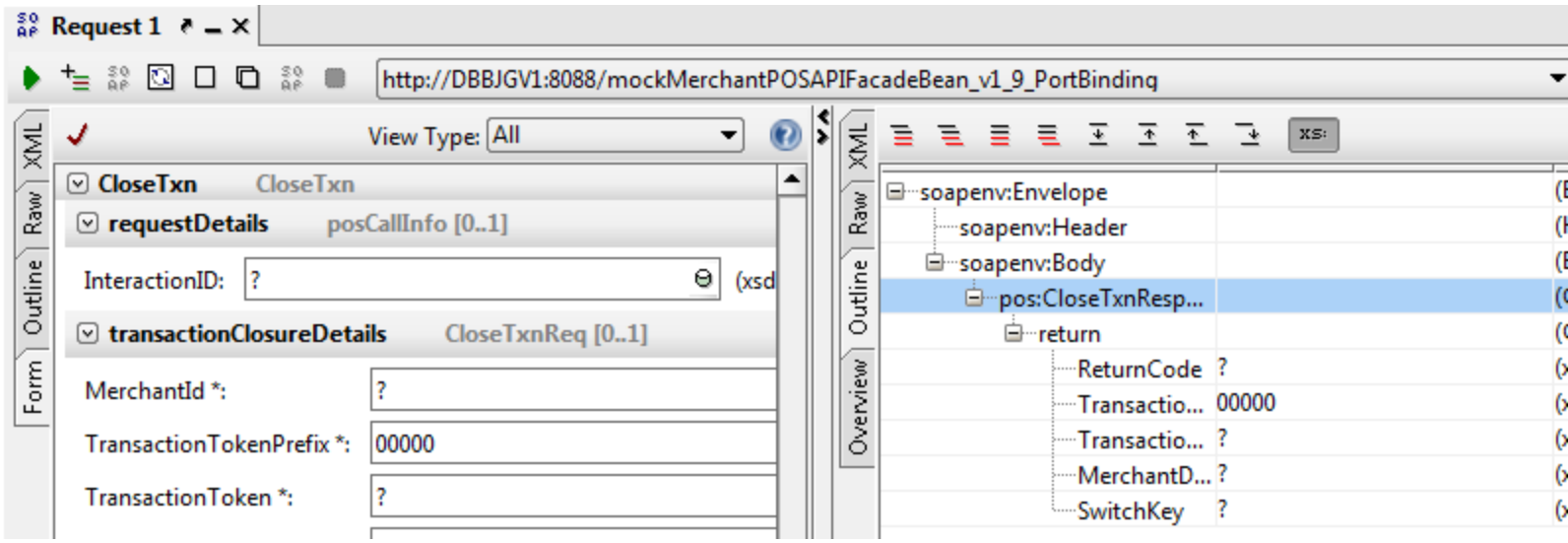
Next assign the PortBinding request to the MockService instead of the URL in the WSDL contract





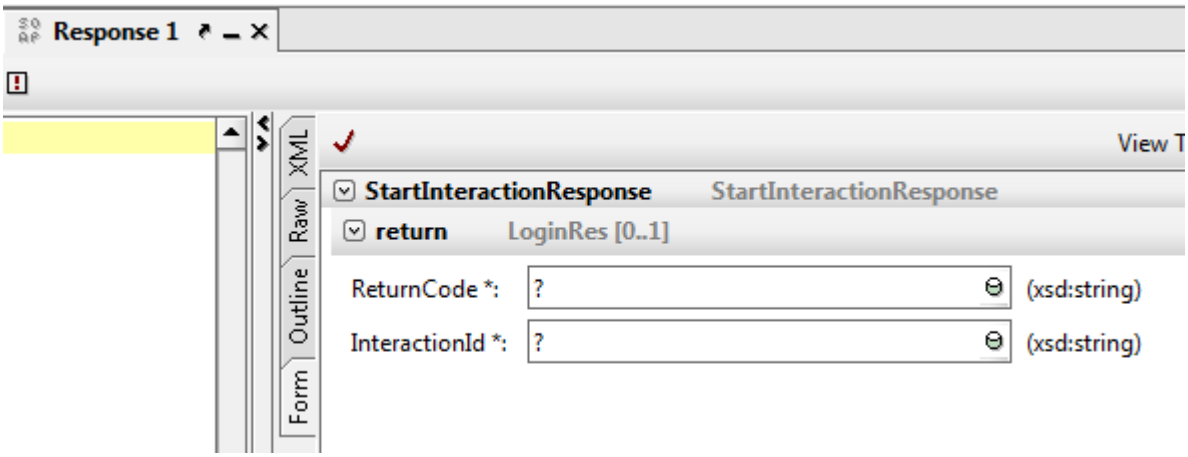
Run the request and a response should be sent from the MockService



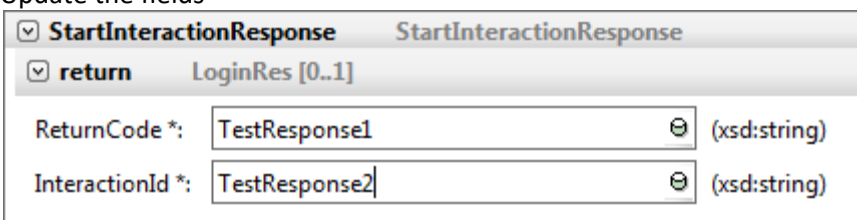


Editing a MockService Response

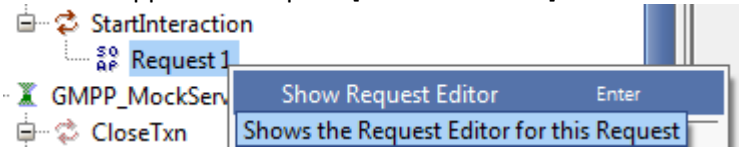
Open a Response under the MockService



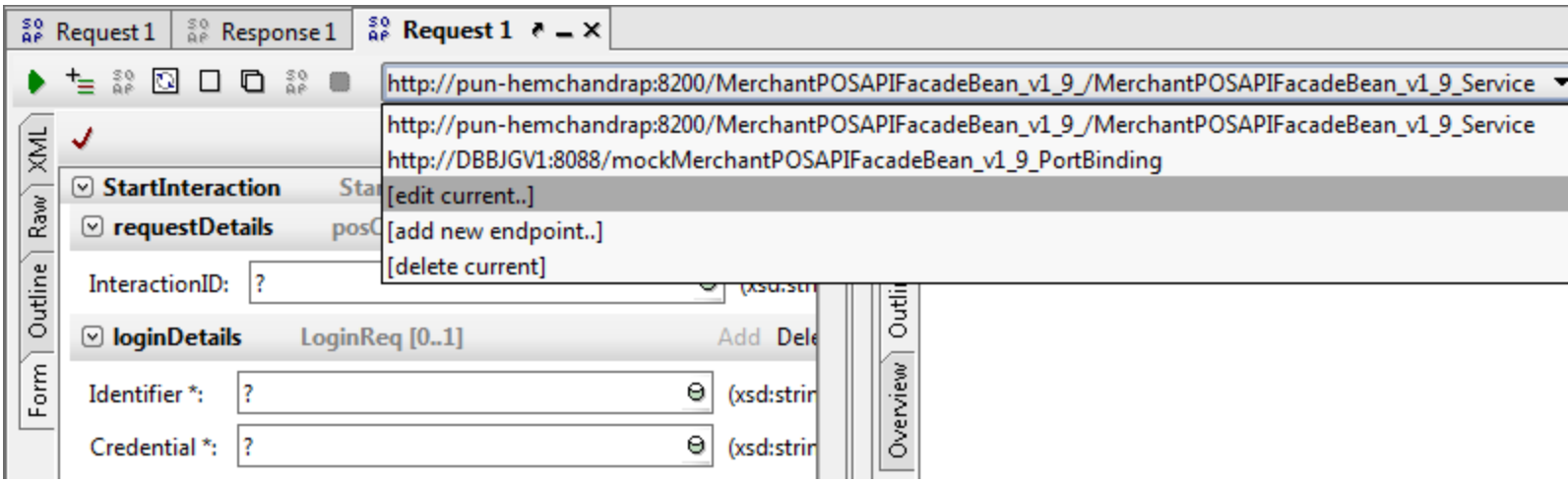
Update the fields



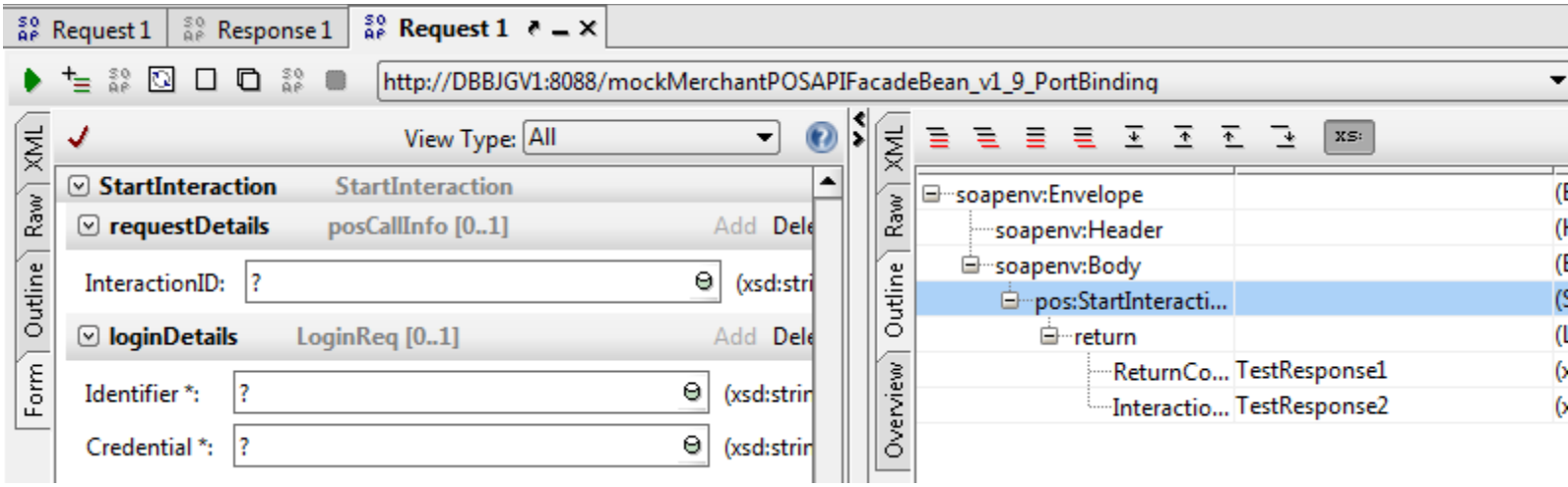
Go to the applicable Request [StartInteraction]



Make sure to point the request to the MockService endpoint!!!

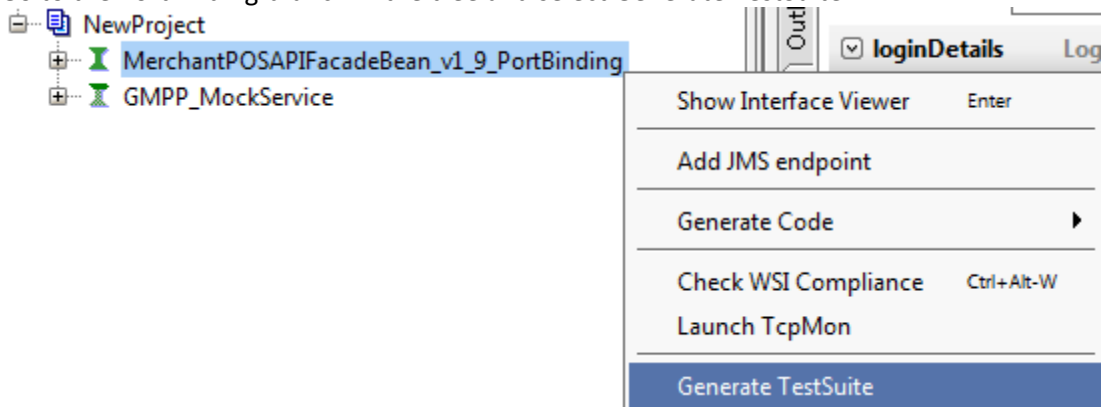


Run the request to see the updated Response data

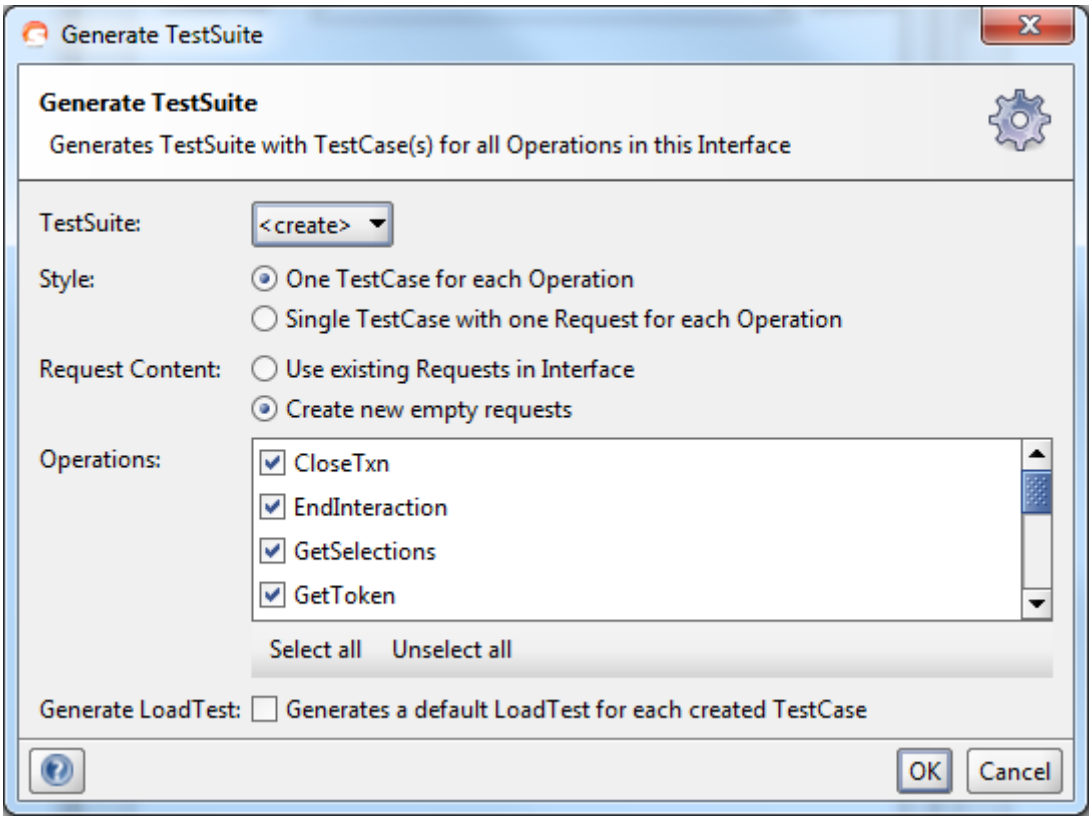


Create Test Suite

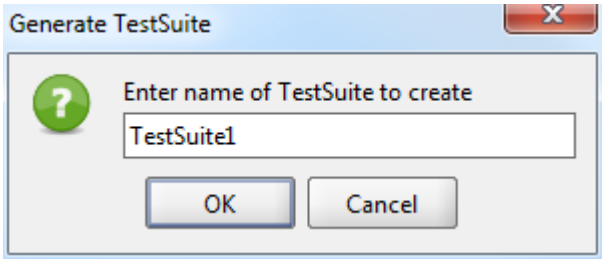
Go to the PortBinding branch in the tree and select Generate TestSuite



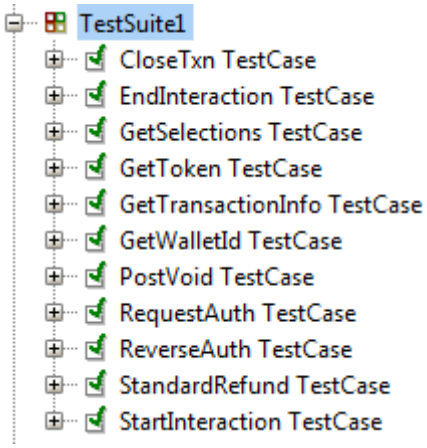
While we really don't need to create empty requests, leave them checked for learning purposes



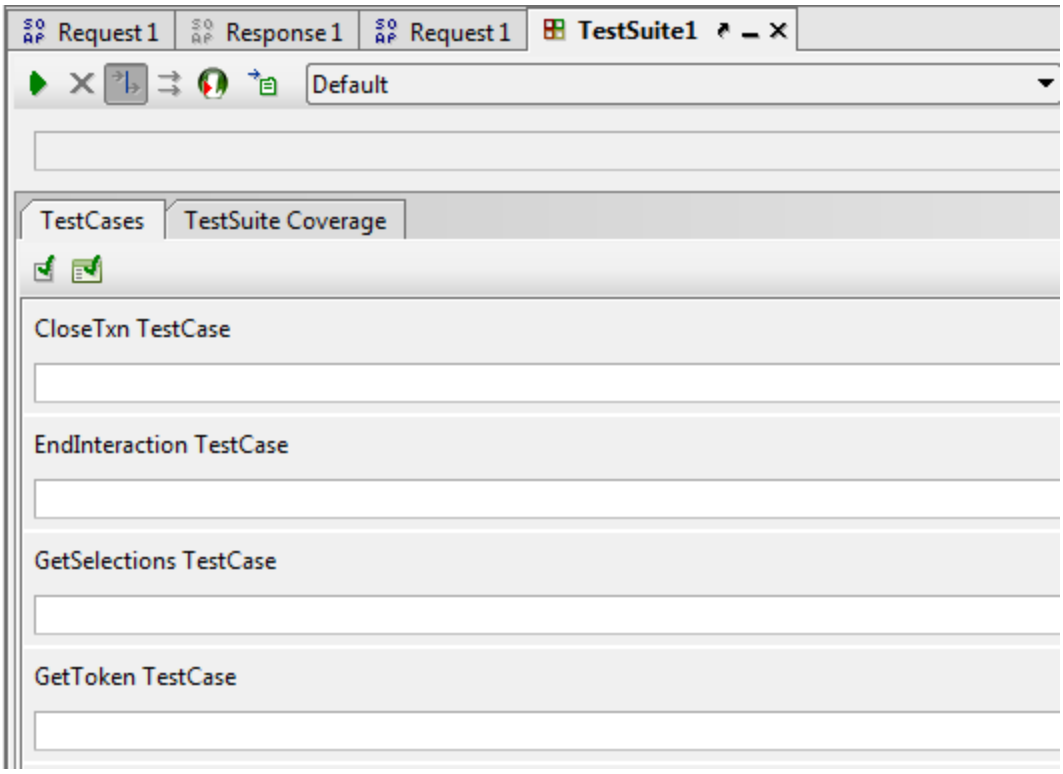
Give it a name



The test suite will be created and added to the project tree

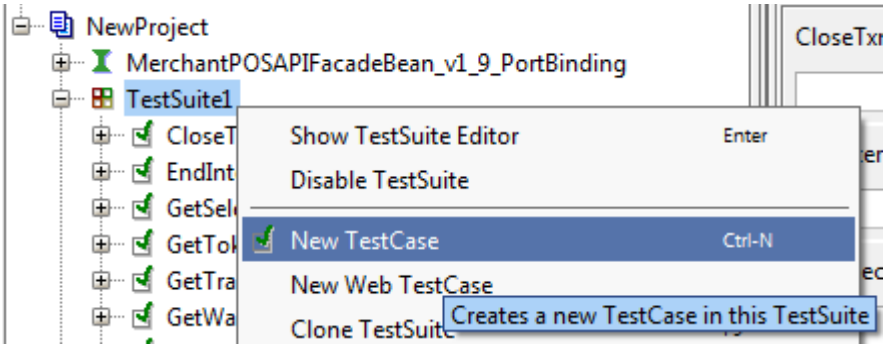


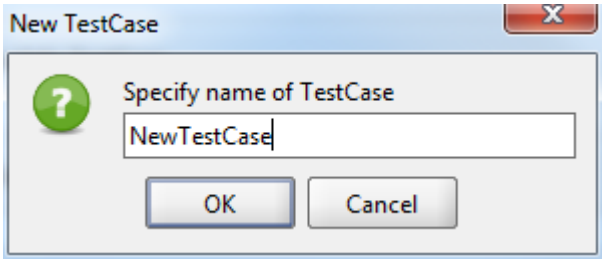
Open the TestSuite and each test case will be listed



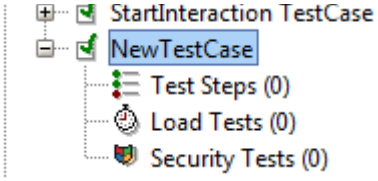
Create a Test Case

Go to the Test Suite in the tree and select New Test Case



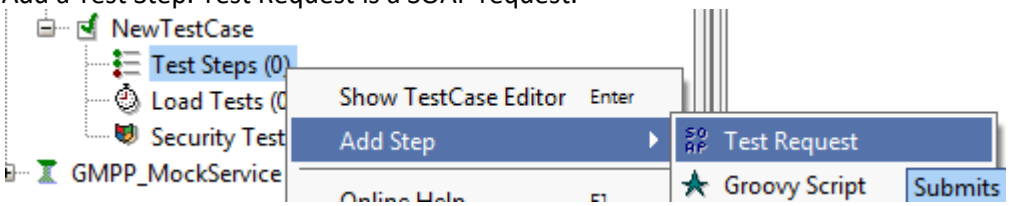


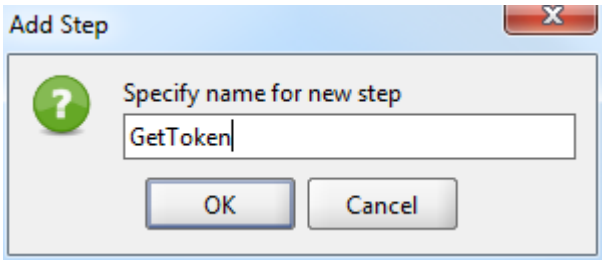
The Test Case will be created

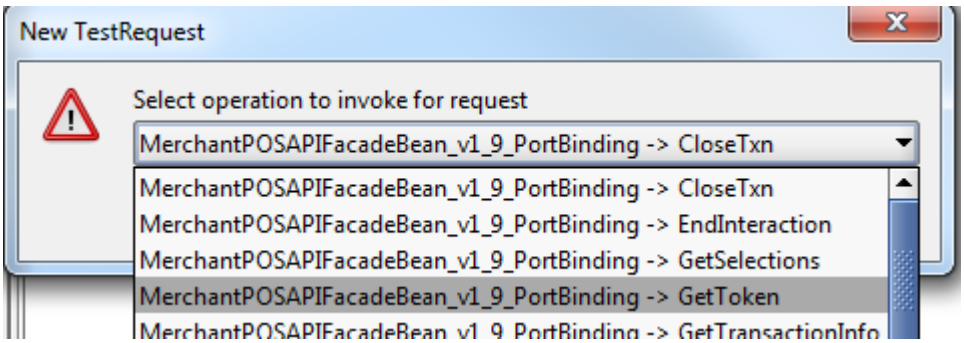


Add Test Step

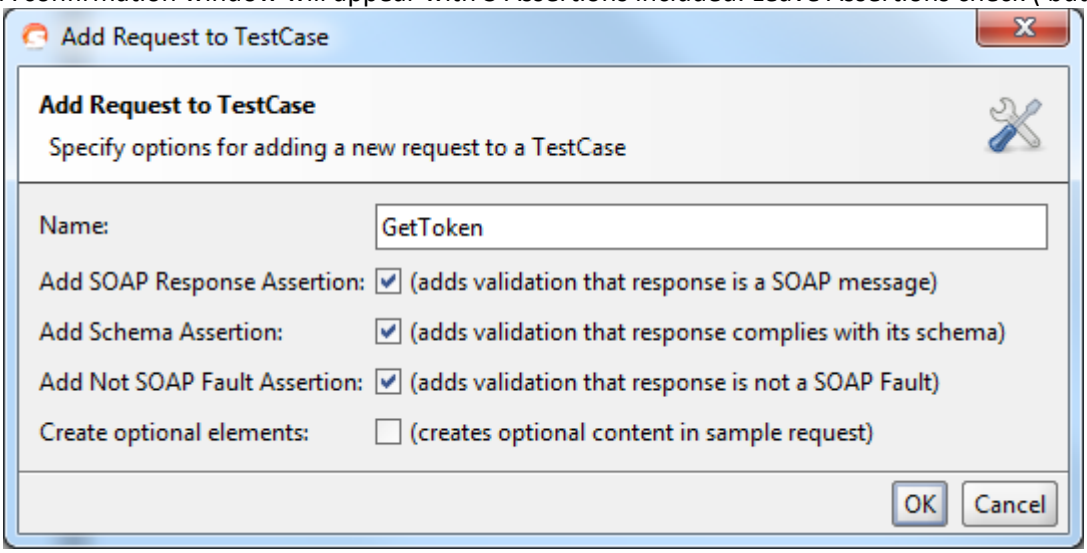
Add a Test Step. Test Request is a SOAP request.



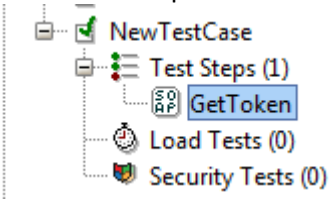




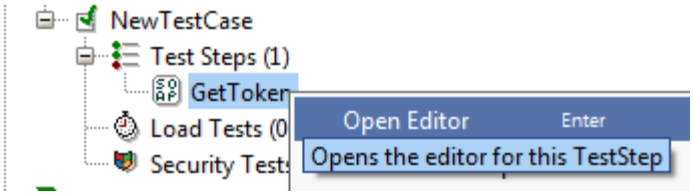
A confirmation window will appear with 3 Assertions included. Leave Assertions check (but you can add them later also)



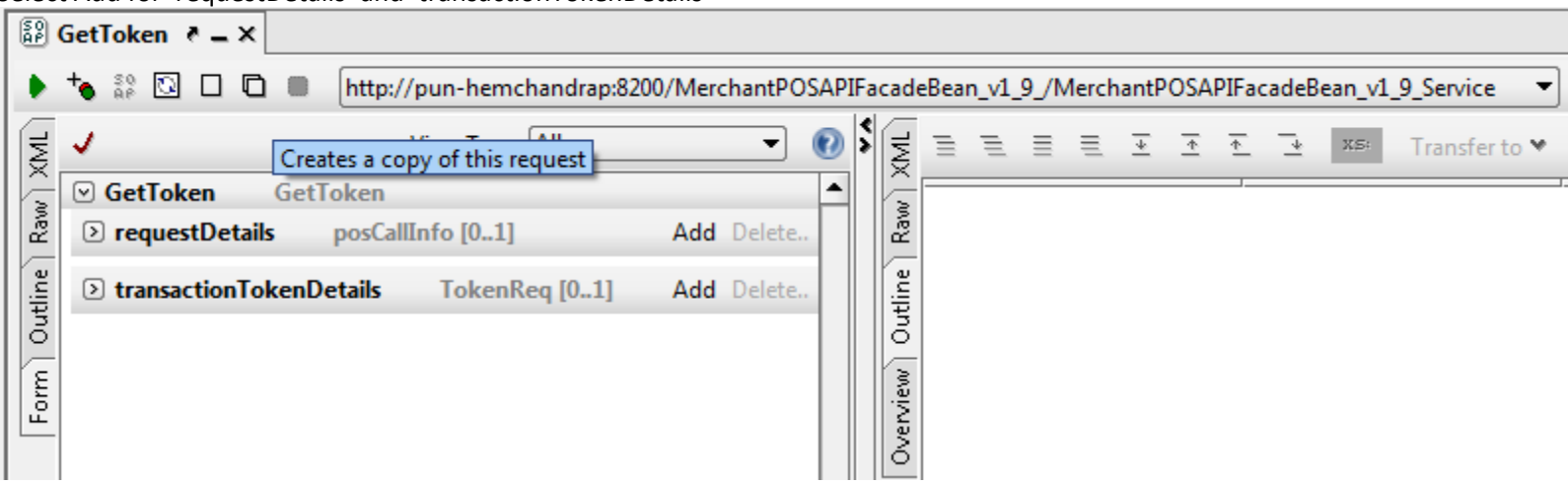
The new Test Step will be added to the tree

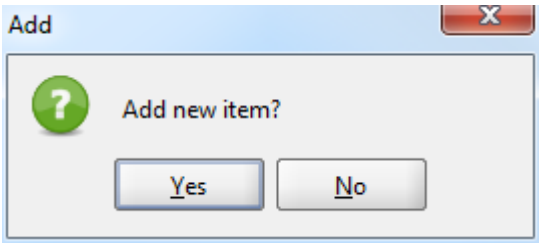


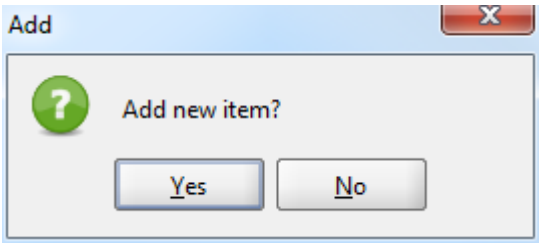
Open the Test Step editor



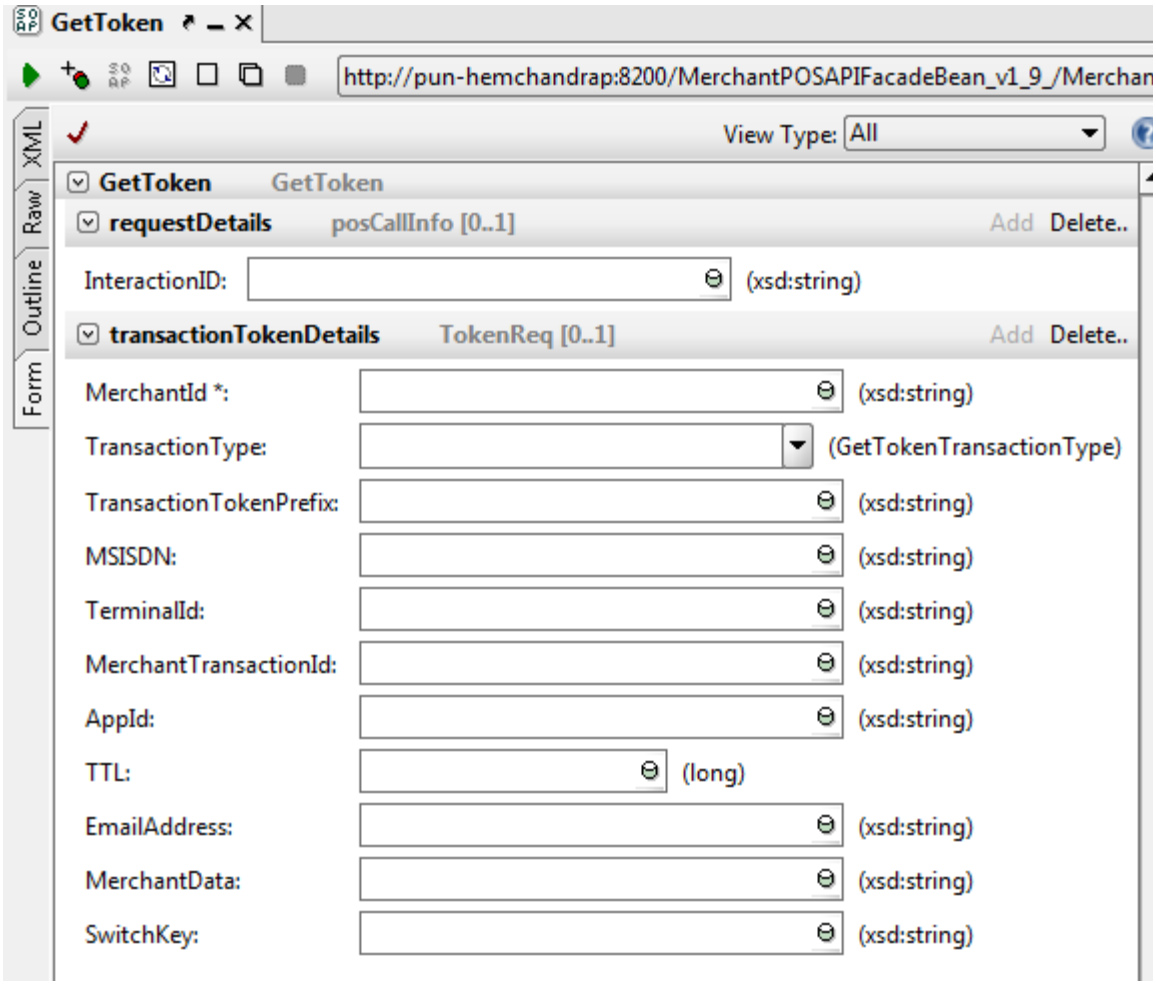
Select Add for ‘requestDetails’ and ‘transactionTokenDetails’



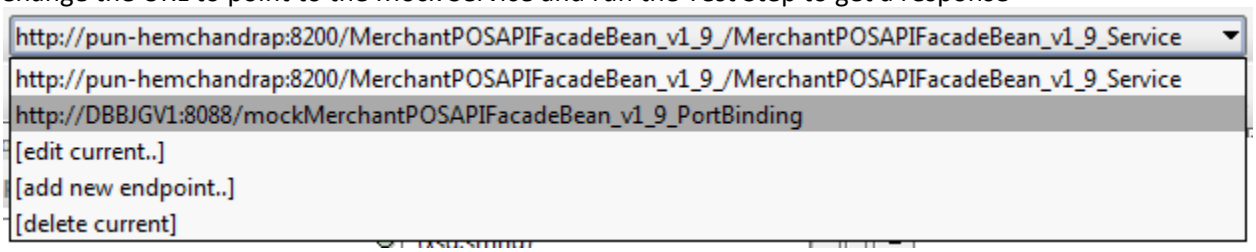




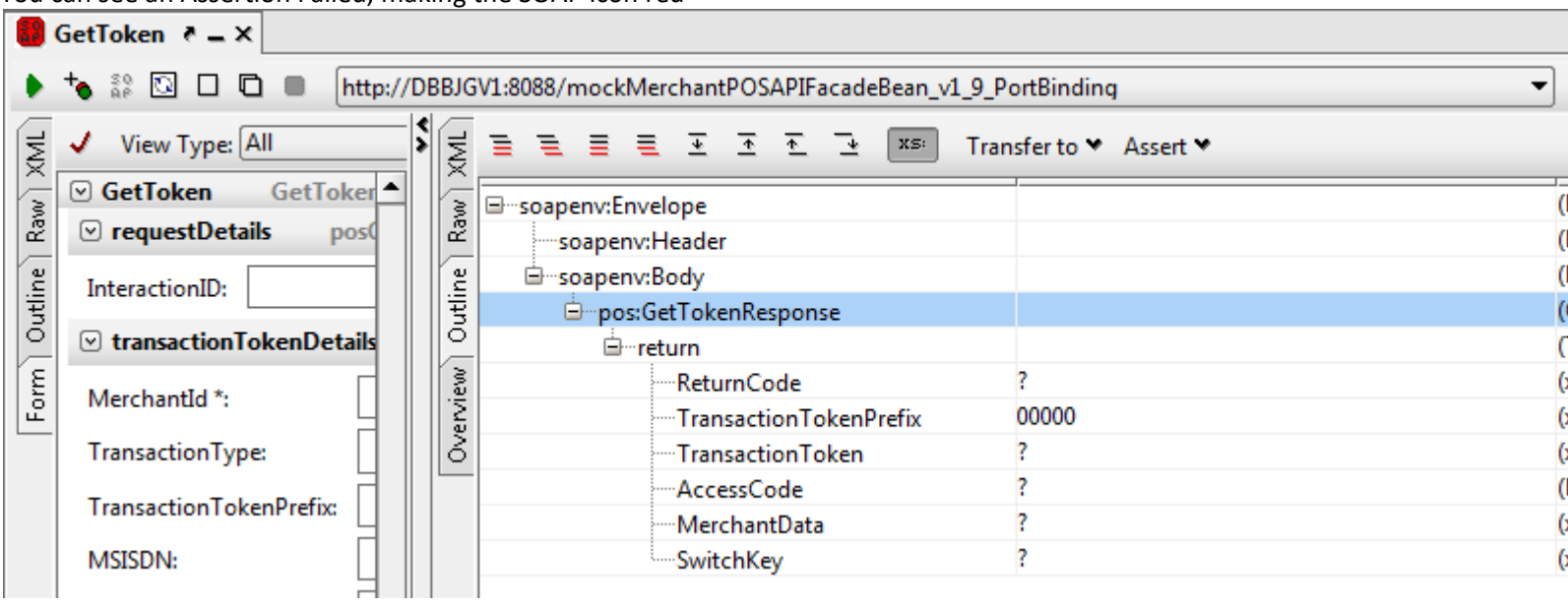
Now you will see all the field in the request



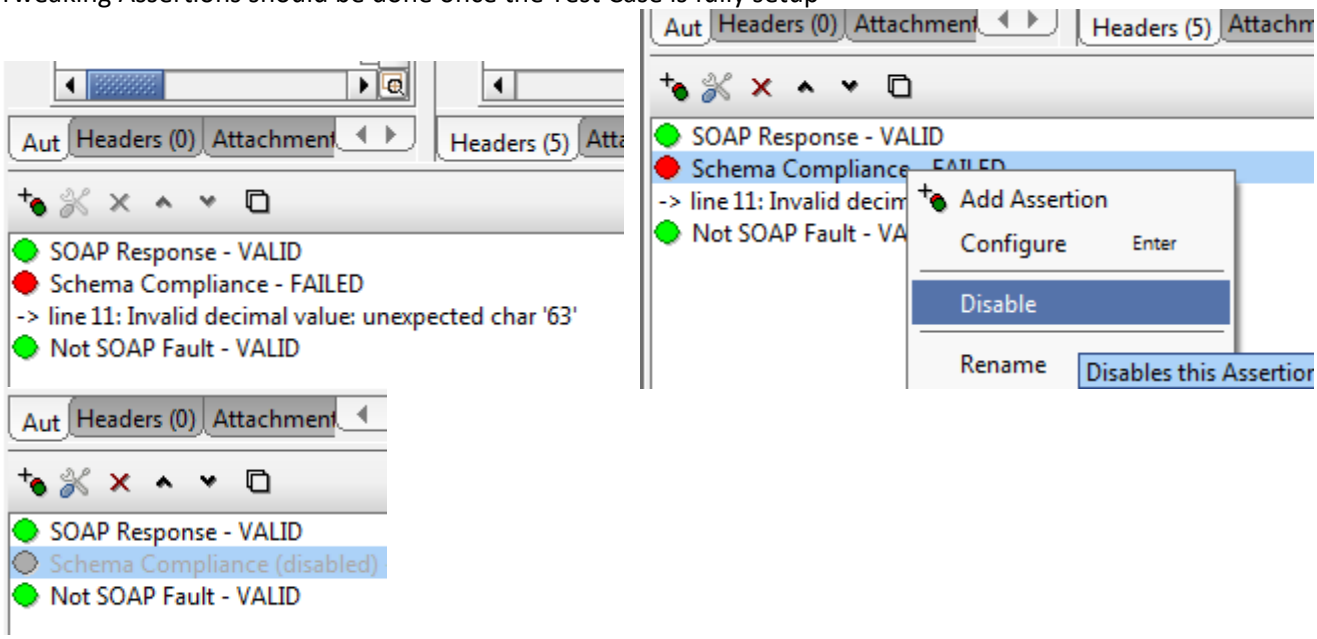
Change the URL to point to the Mock Service and run the Test Step to get a response



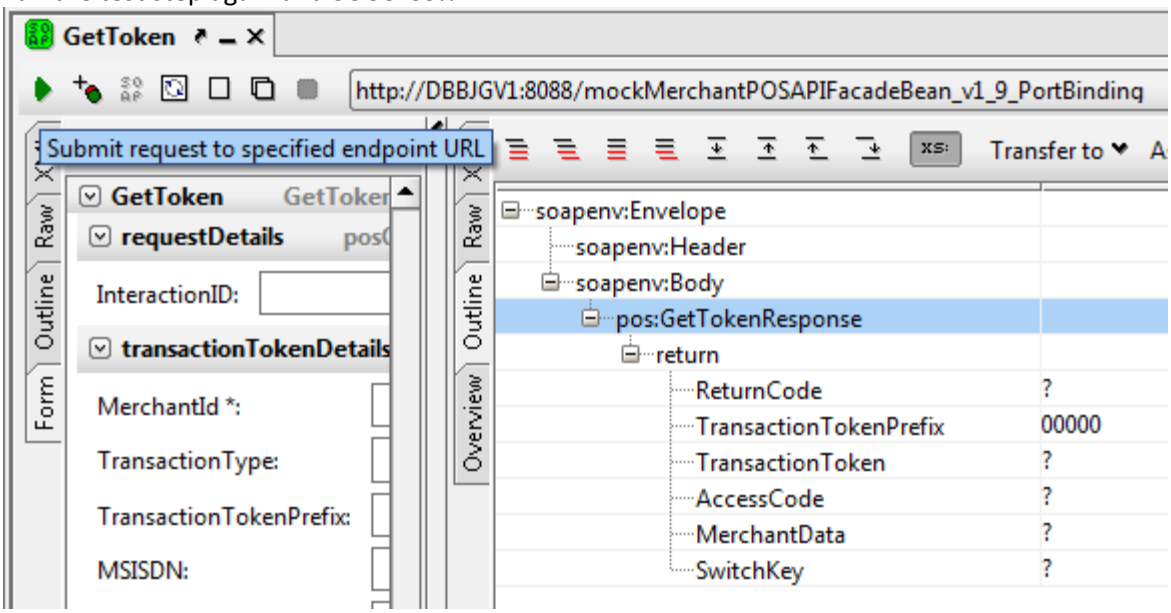
You can see an Assertion Failed, making the SOAP icon red



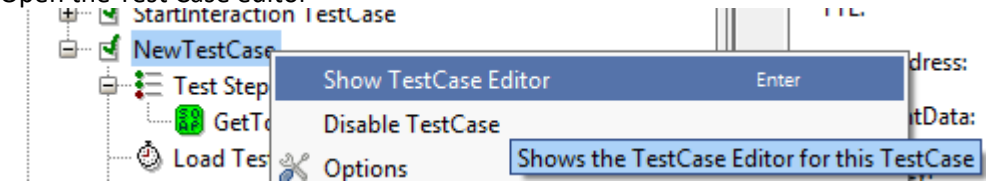
At the bottom of the Test Case form, we can see the ‘Schema Compliance – FAILED’
At this point just disable the Assertion so the test will execute successfully.
Tweaking Assertions should be done once the Test Case is fully setup



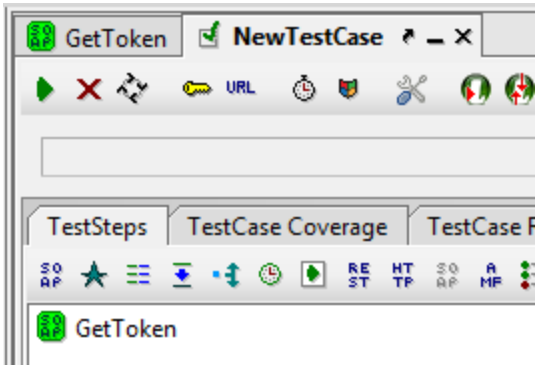
Run the test step again and SUCCESS!!



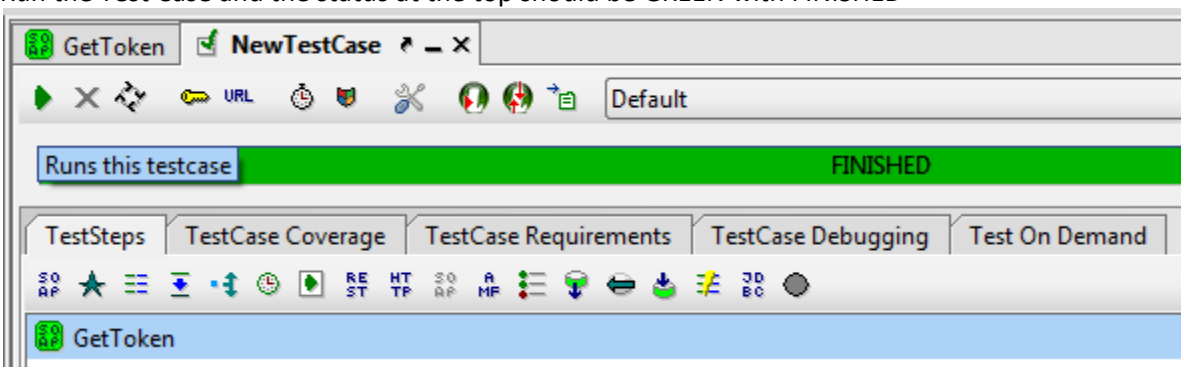
Open the Test Case editor



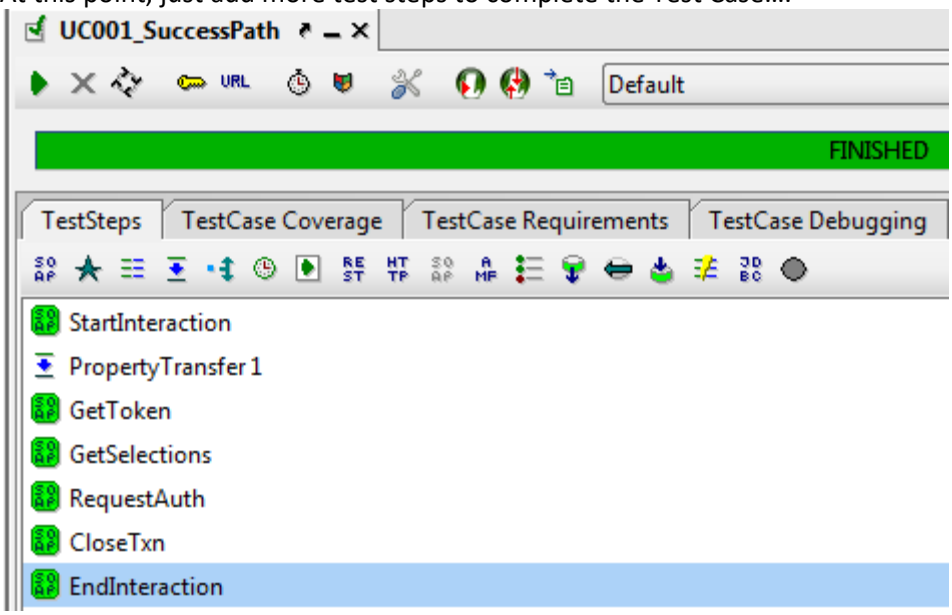
The new Test Step is now present with green icon



Run the Test Case and the status at the top should be GREEN with FINISHED

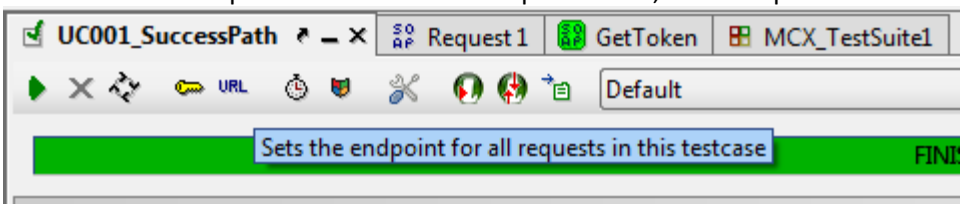


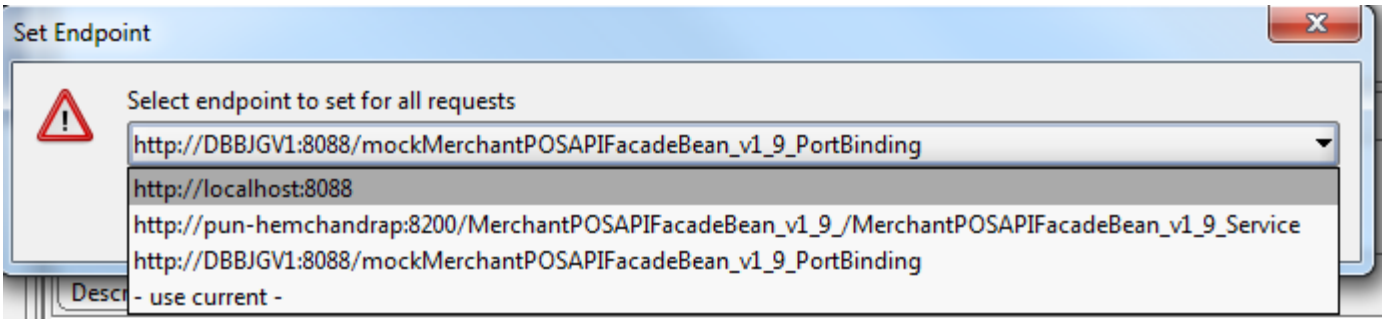
At this point, just add more test steps to complete the Test Case….

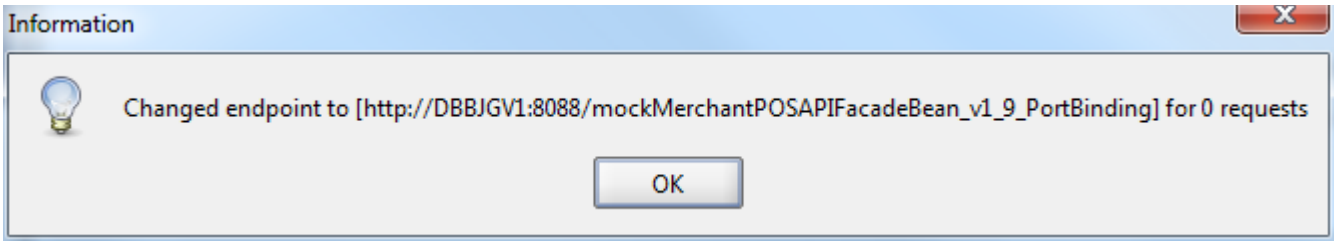


Set Test Case Endpoint (For ALL Test Steps)

While each test Step can be directed to a separate URL, it is best practice to set it one time at the Test Case level.

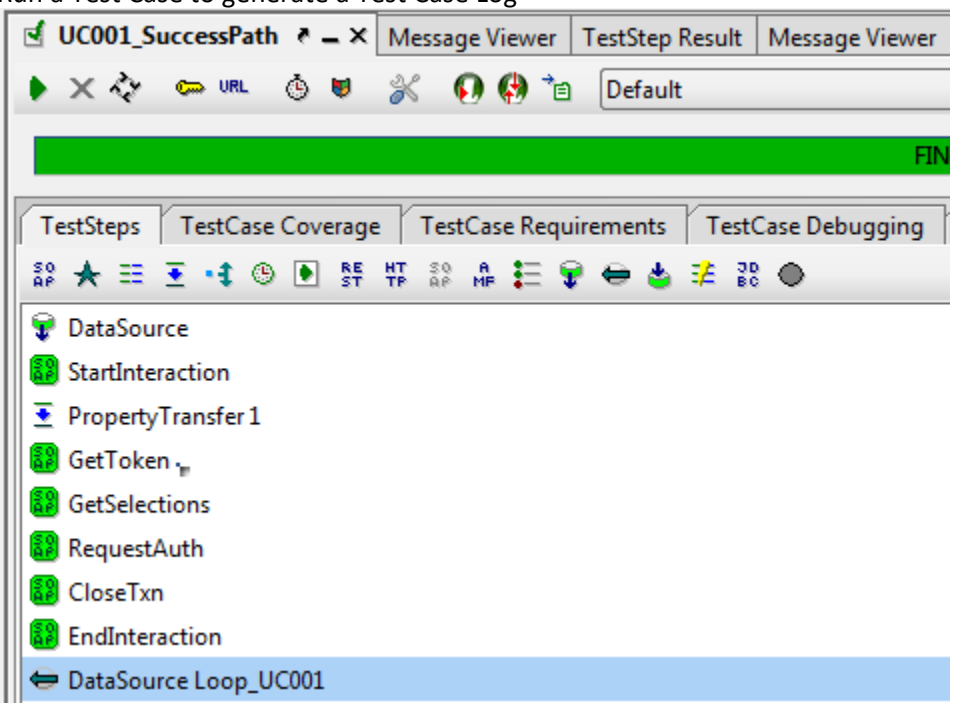



Set Endpoint
Select endpoint to set for all requests
http://DBBJGV1:8088/mockMerchantPOSAPIFacadeBean_v1_9_PortBinding
http://localhost:8088
http://pun-hemchandrap:8200/MerchantPOSAPIFacadeBean_v1_9_/MerchantPOSAPIFacadeBean_v1_9_Service
http://DBBJGV1:8088/mockMerchantPOSAPIFacadeBean_v1_9_PortBinding
- use current -

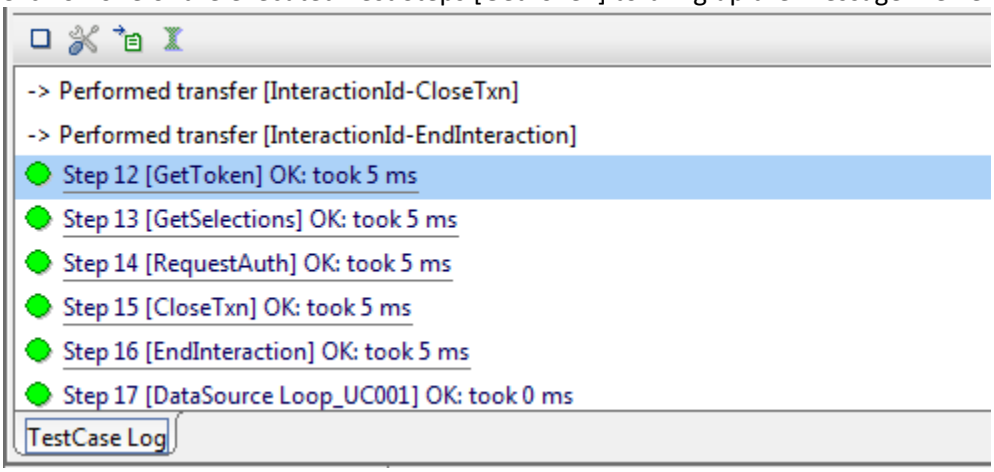

Information
Changed endpoint to [http://DBBJGV1:8088/mockMerchantPOSAPIFacadeBean_v1_9_PortBinding] for 0 requests
OK

Reviewing Test Case Requests and Responses

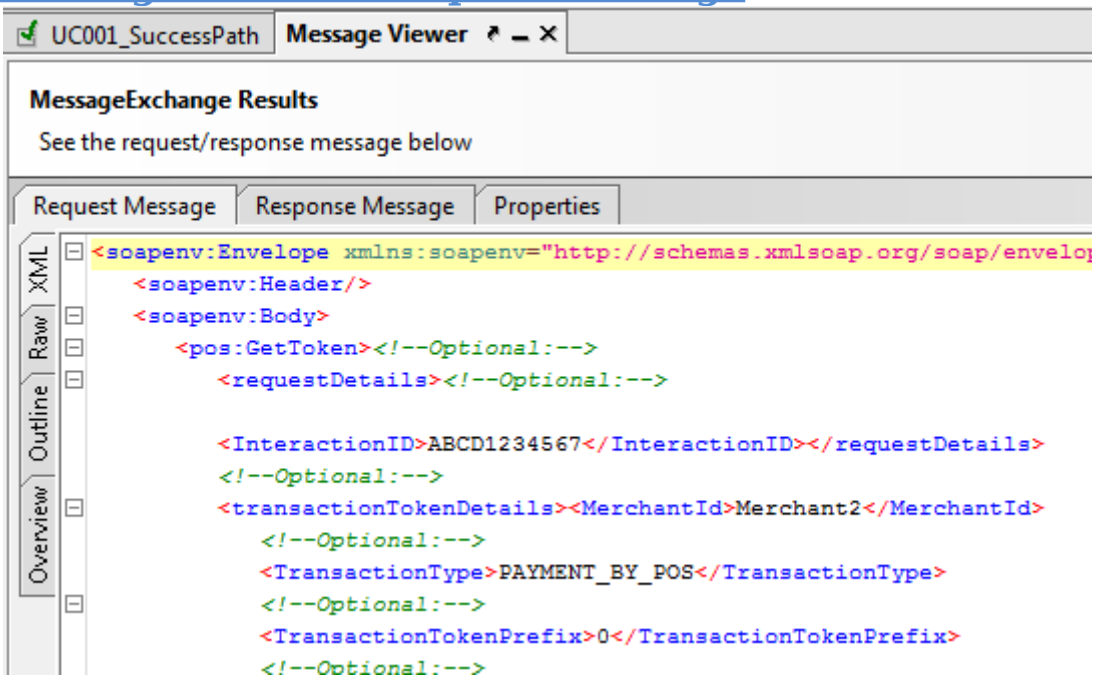
Run a Test Case to generate a Test Case Log



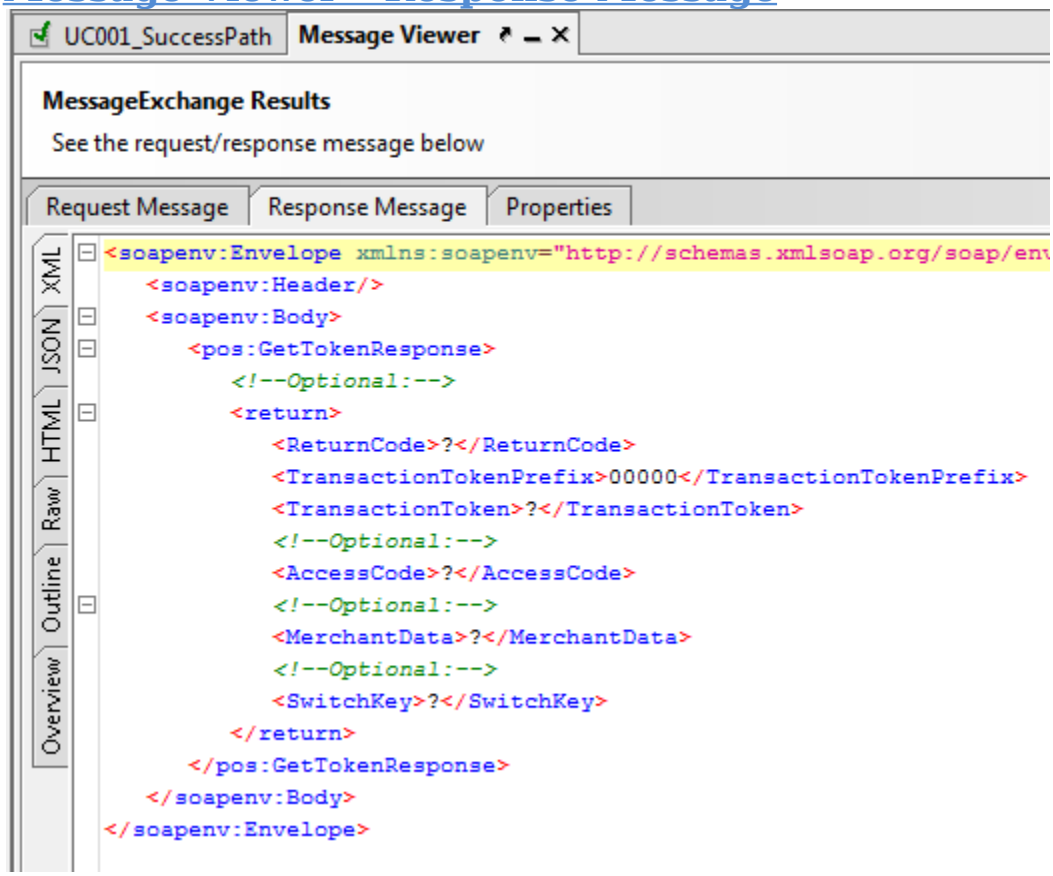
Click on one of the executed Test Steps [GetToken] to bring up the Message Viewer



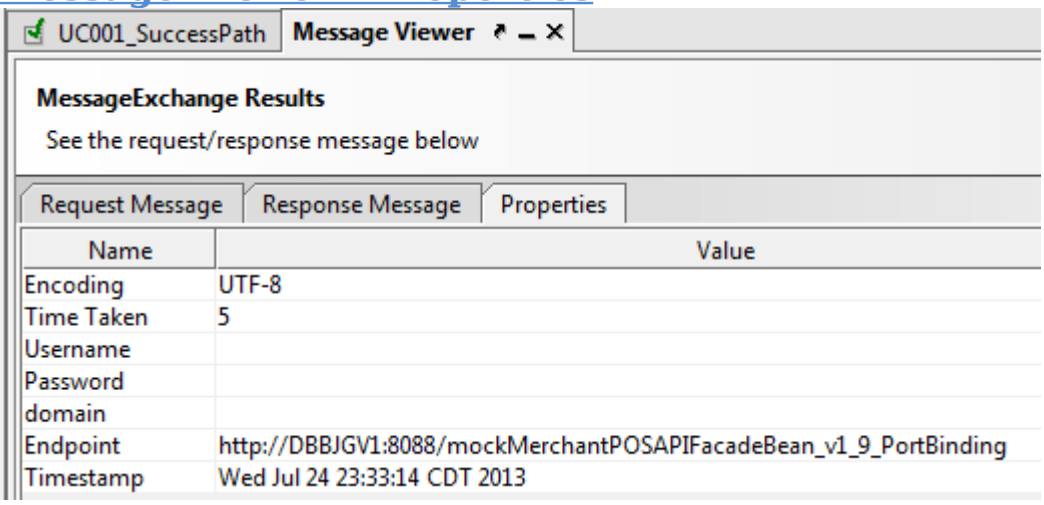
Message Viewer - Request Message



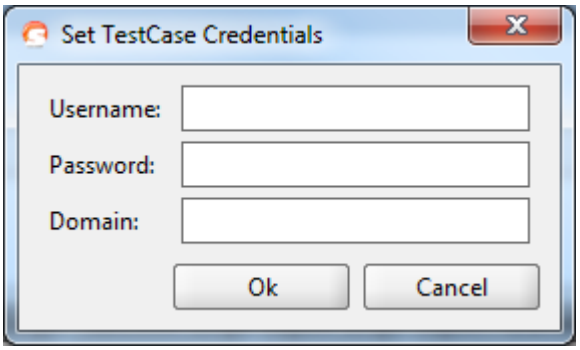
Message Viewer - Response Message



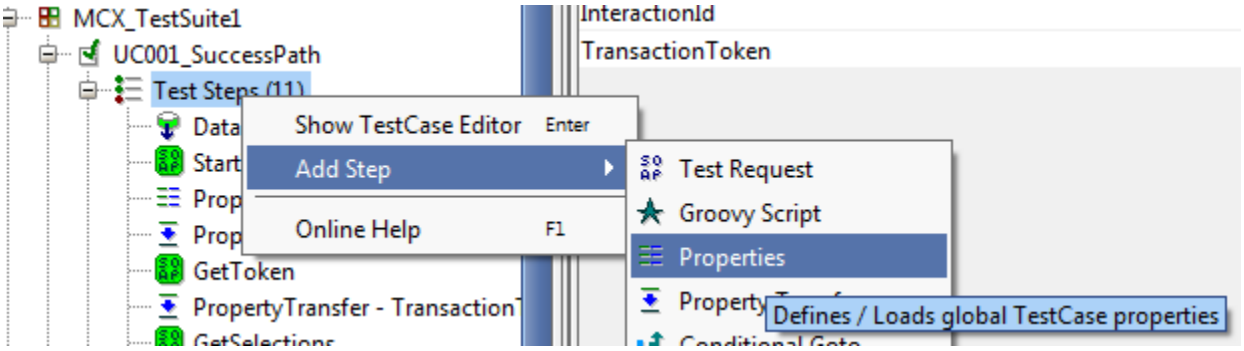
Message Viewer - Properties

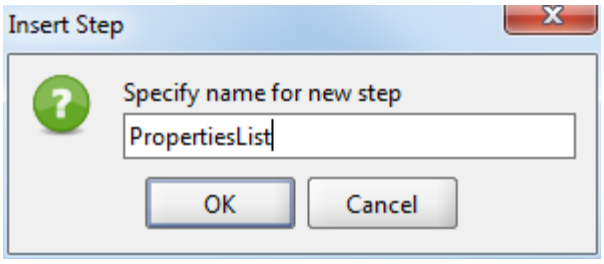


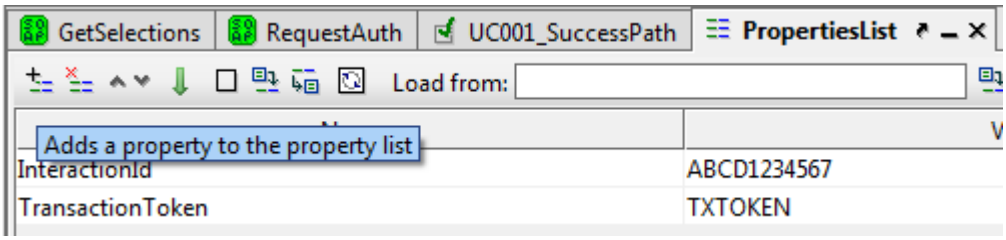
Set Credentials for Test Case URL

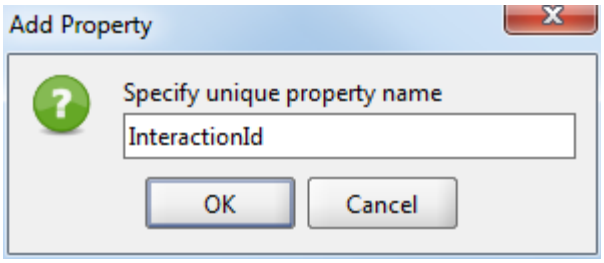


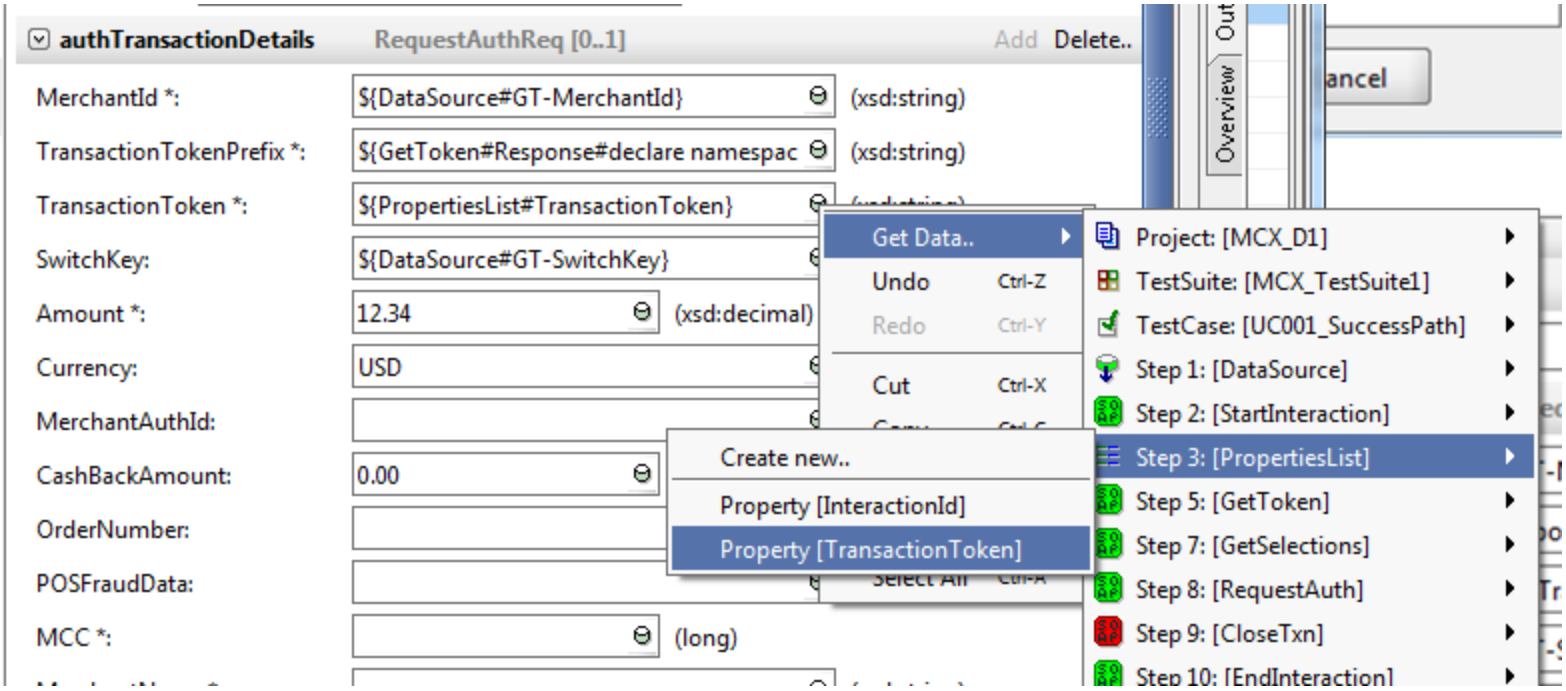
Properties List



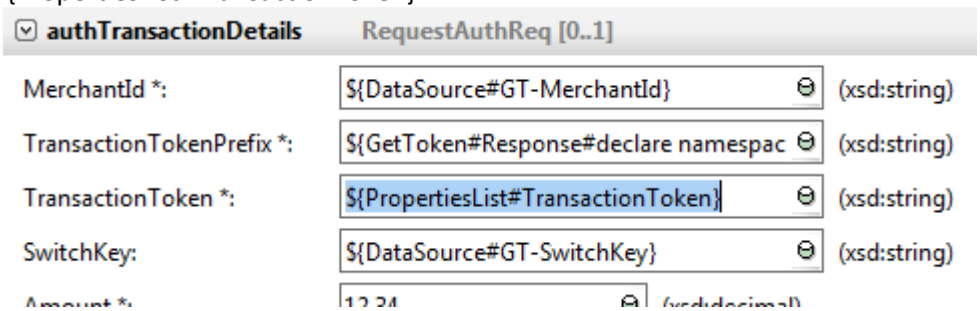






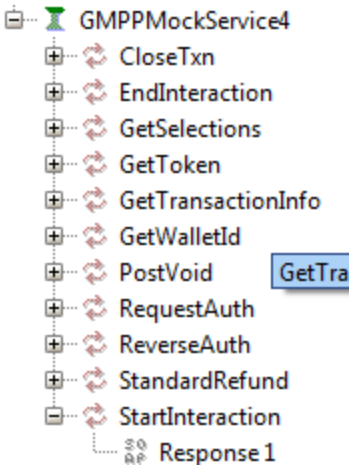


${PropertiesList#TransactionToken}

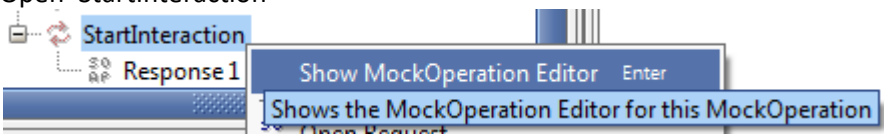


Transfer Properties

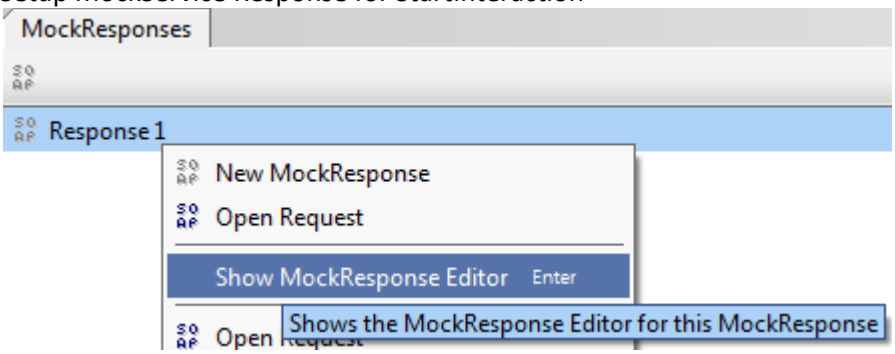
Go to the MockService Responses in the tree



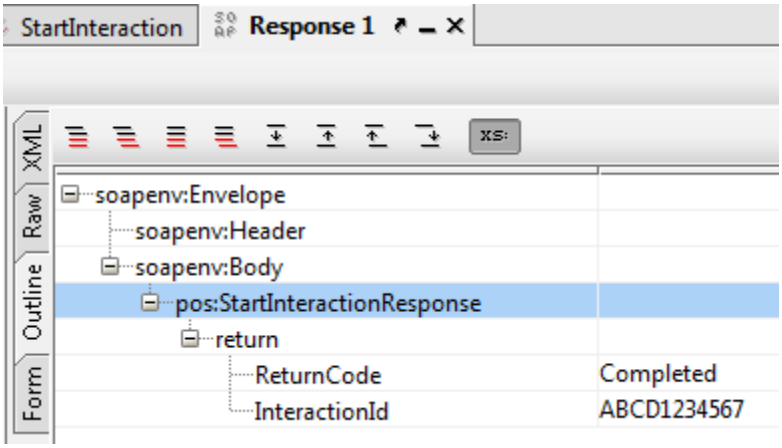
Open 'StartInteraction '



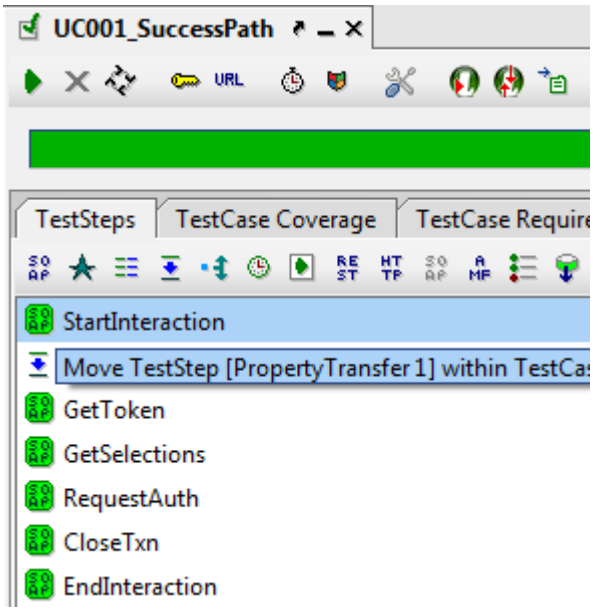
Setup MockService Response for StartInteraction



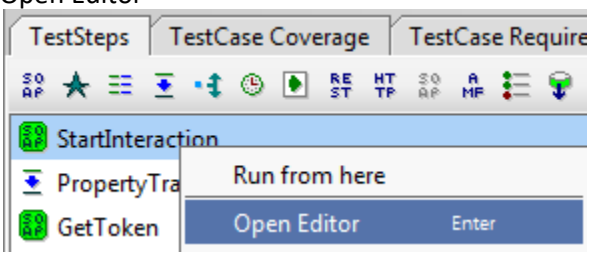
We'll set it to ABCD1234567..

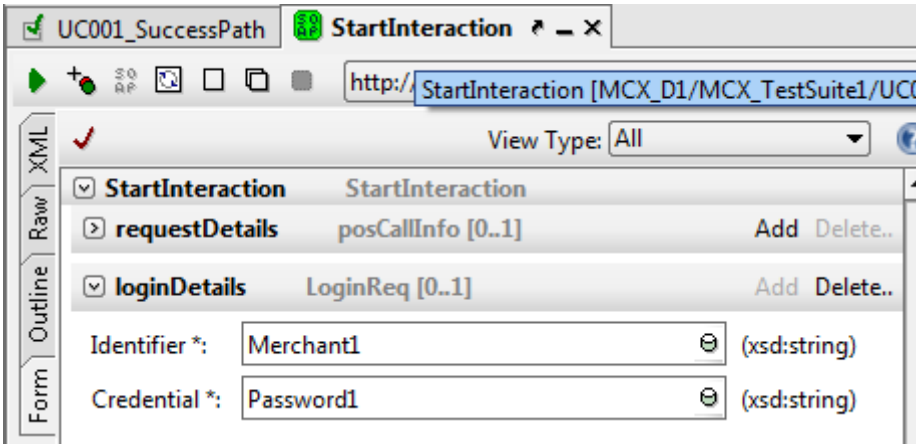


Open the Test Case that has multiple Test Steps

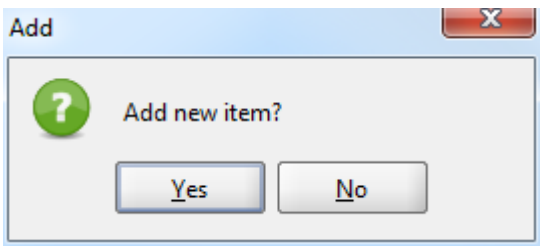


Open Editor

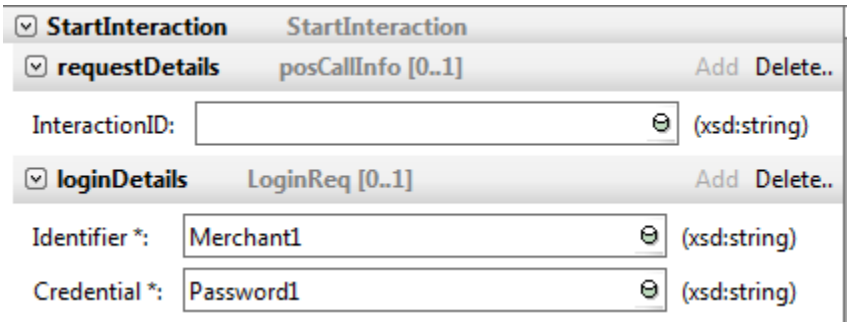




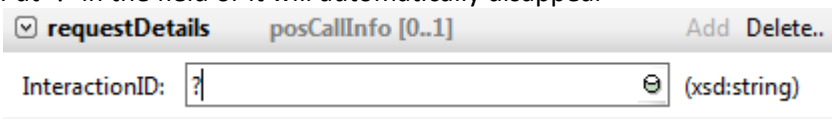
Add the StartInteraction, requestDetails, InteractionID field



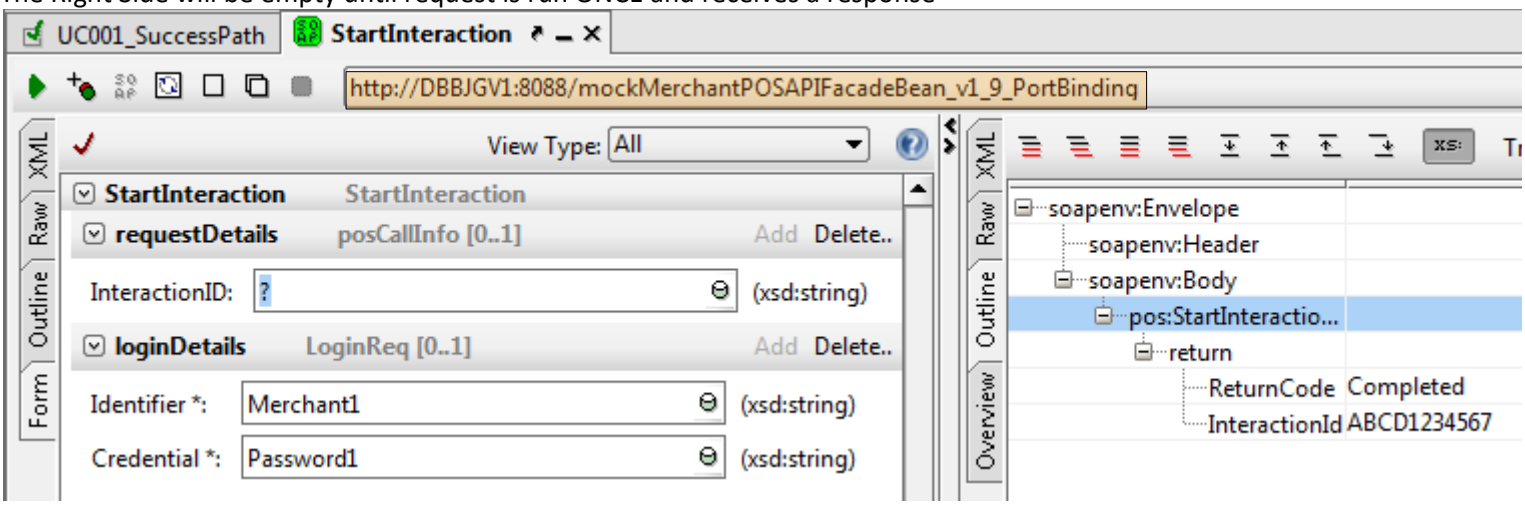
It now shows….



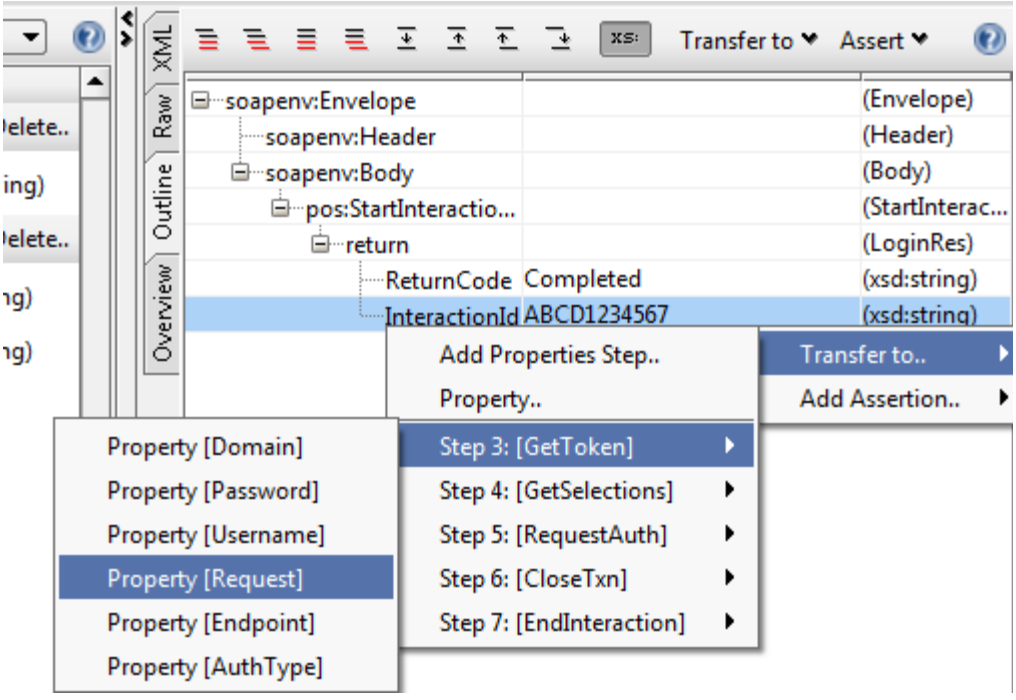
Put '?' in the field or it will automatically disappear



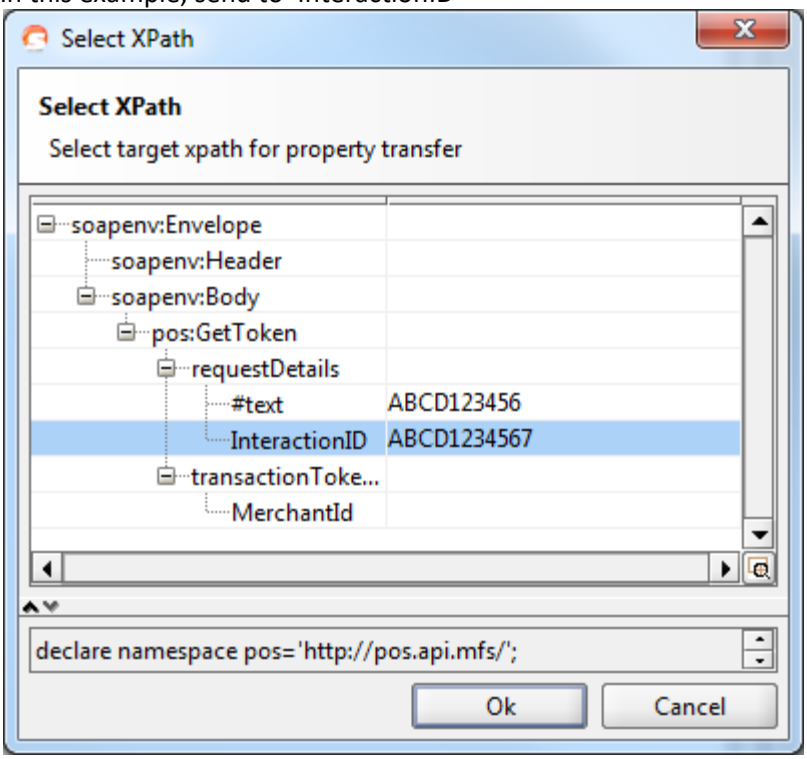
Make sure the endpoint is set to the MockService and Run the request to get a response from the MockService
The Right Side will be empty until request is run ONCE and receives a response



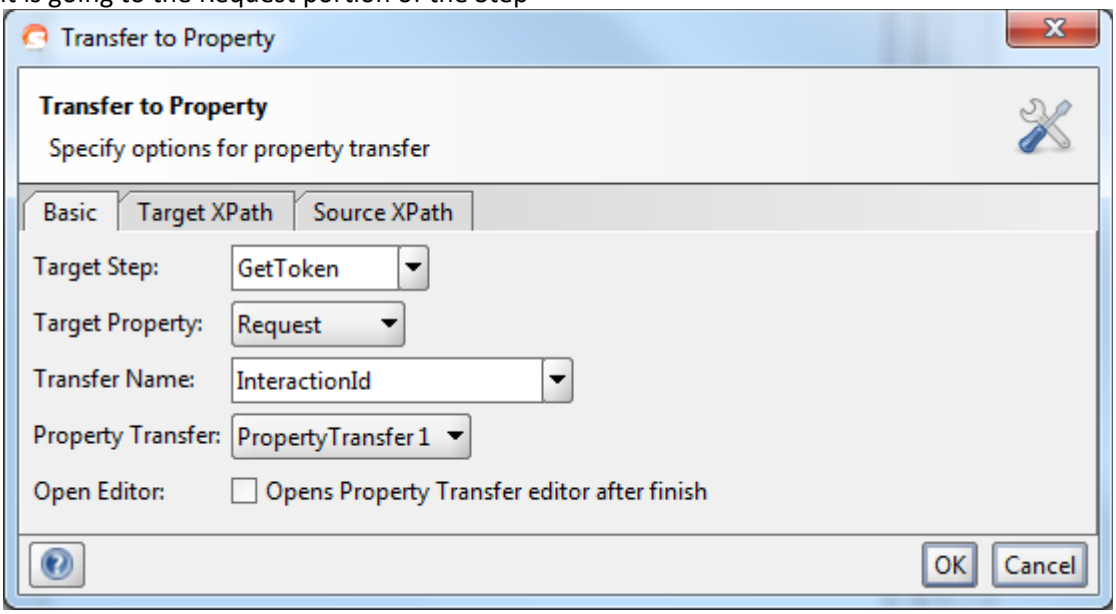
Now that we have a response, we tell SOAPUI to transfer a field from the Response to the next request.
In this case we want to Transfer 'InteractionId' to GetToken request 'InteractionID' (note the different cases of 'Id' and 'ID'



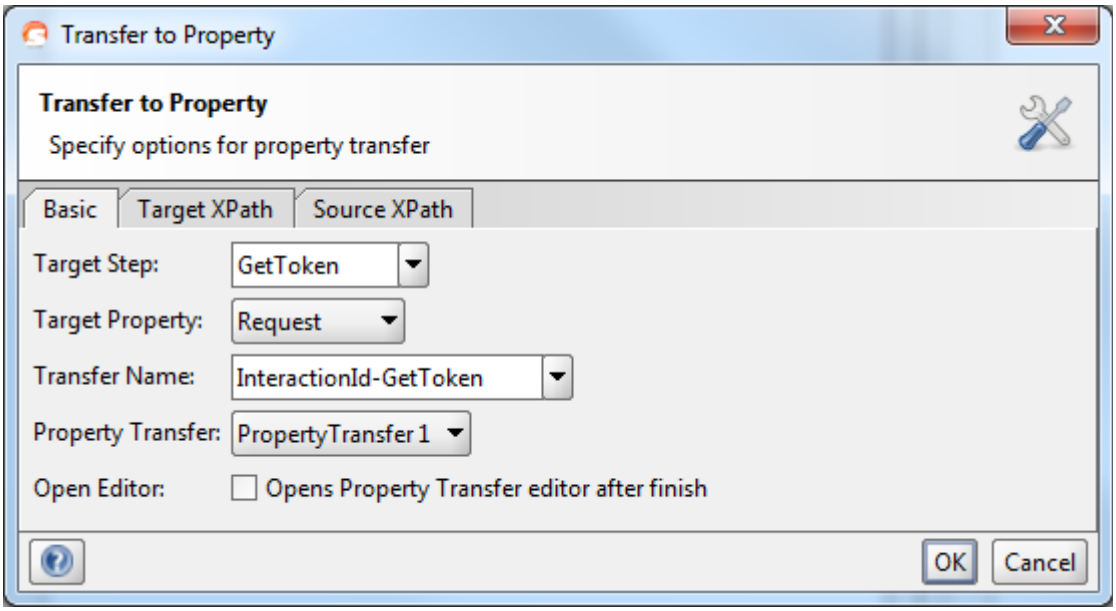
SOAPUI will ask for more information; where to SEND the Transferred data
In this example, send to ‘InteractionID’



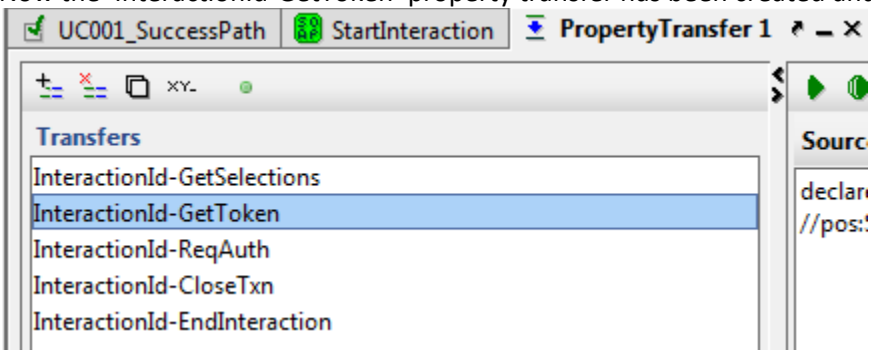
SOAPUI will ask for more information and/or confirm the setting before creating the Property Transfer
We’re sending it to the ‘GetToken’ Step/Request
It is going to the Request portion of the Step



Update the name to ‘Field’-‘RequestName’ so it can be easily identified

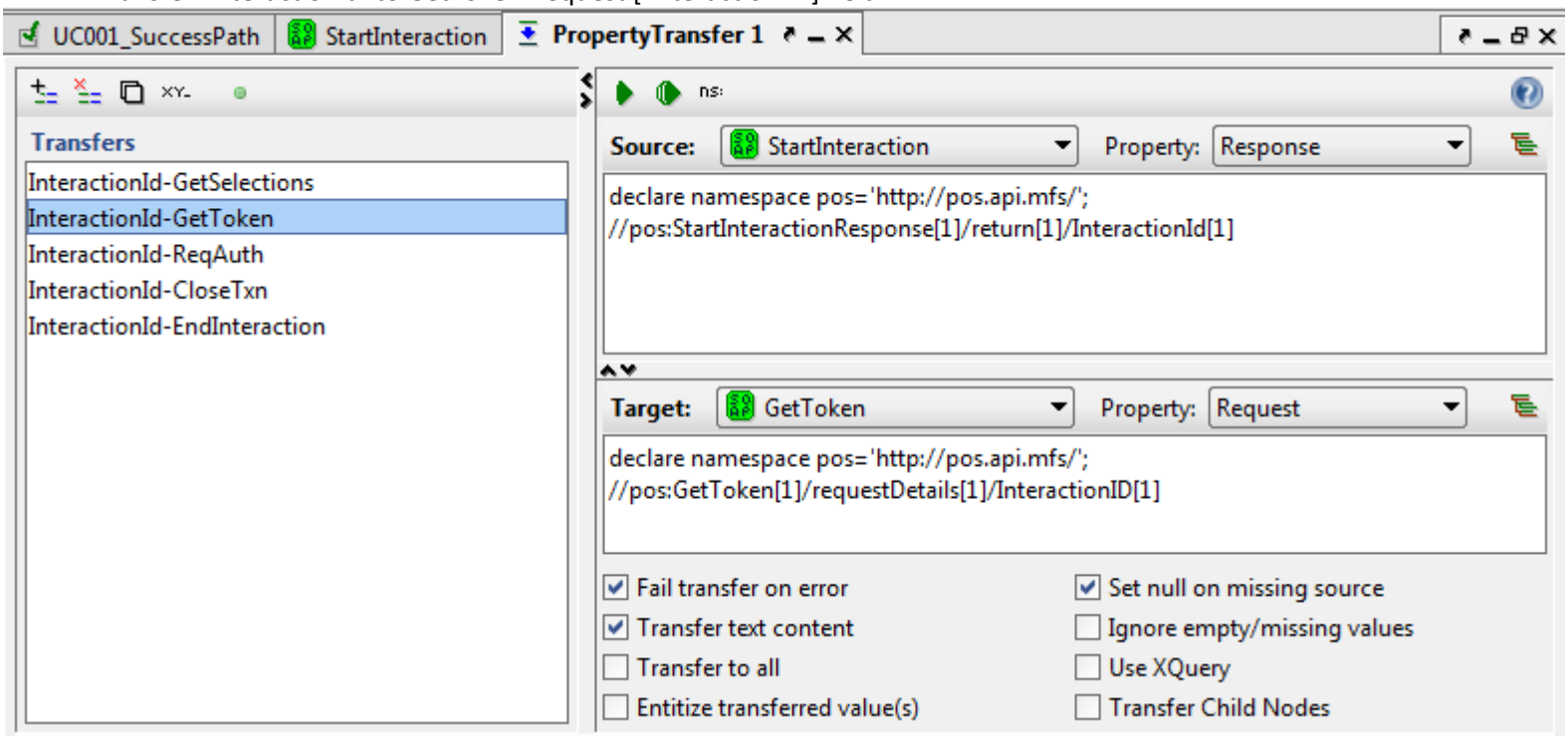


Now the 'InteractionId-GetToken' property transfer has been created and added to already existing transfers

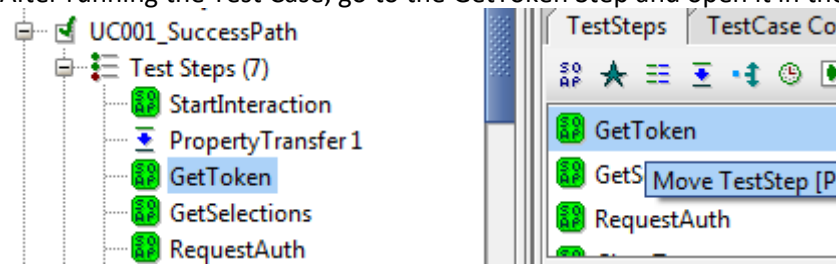


The complete tab
SOAPUI uses XPATH to identify items within the XML Response AND the target XML Request (GetToken)
For the example:

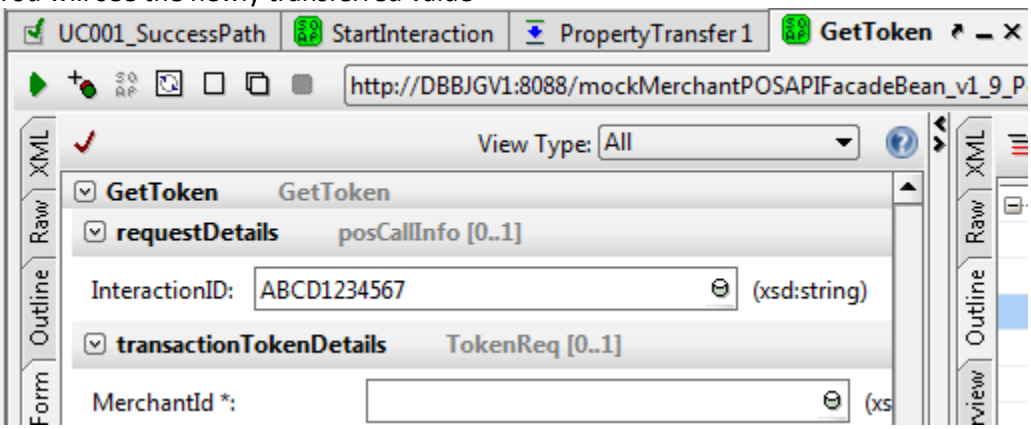
1. Run StartInteraction Request and get 'InteractionId'
2. Transfer 'InteractionId' to GetToken Request [InteractionID] field



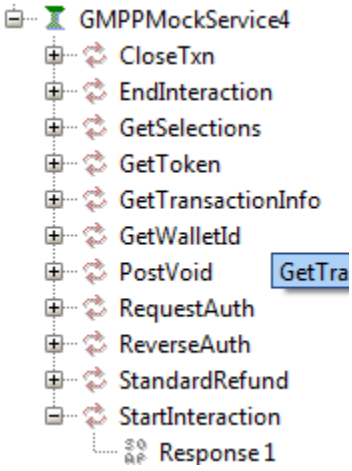
After running the Test Case, go to the GetToken Step and open it in the editor



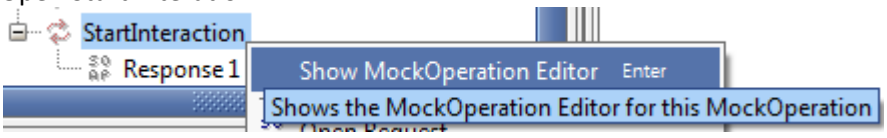
You will see the newly transferred value



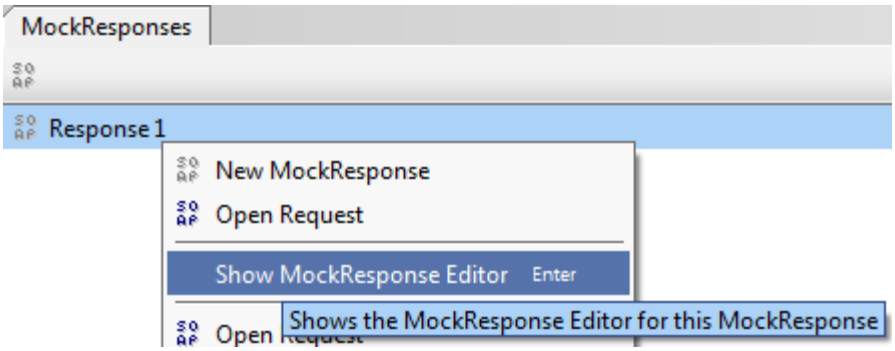
Goto MockService

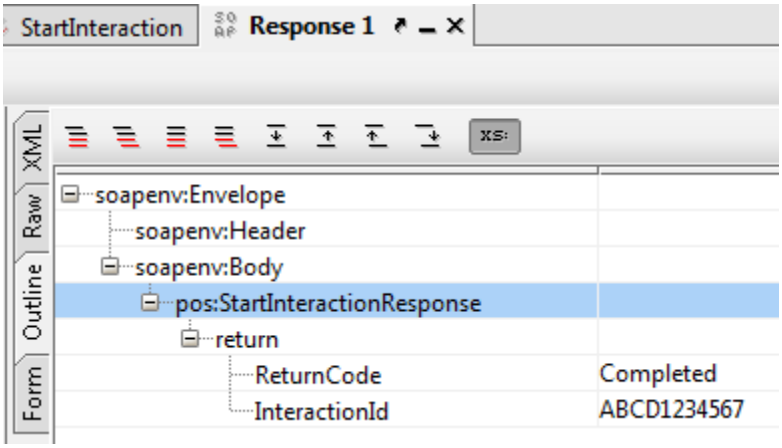


Open Start Interation



Setup MockService Response for StartInteraction

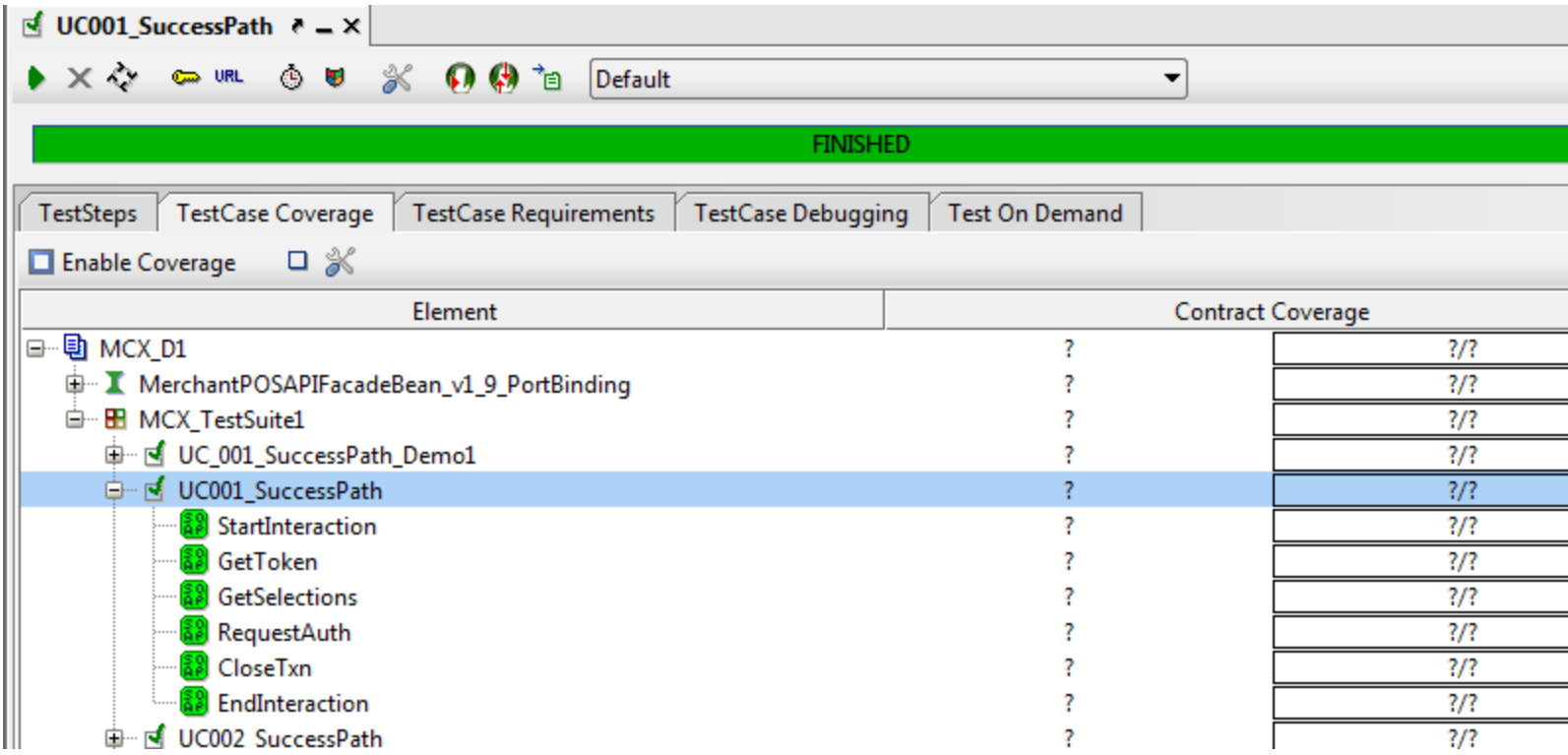




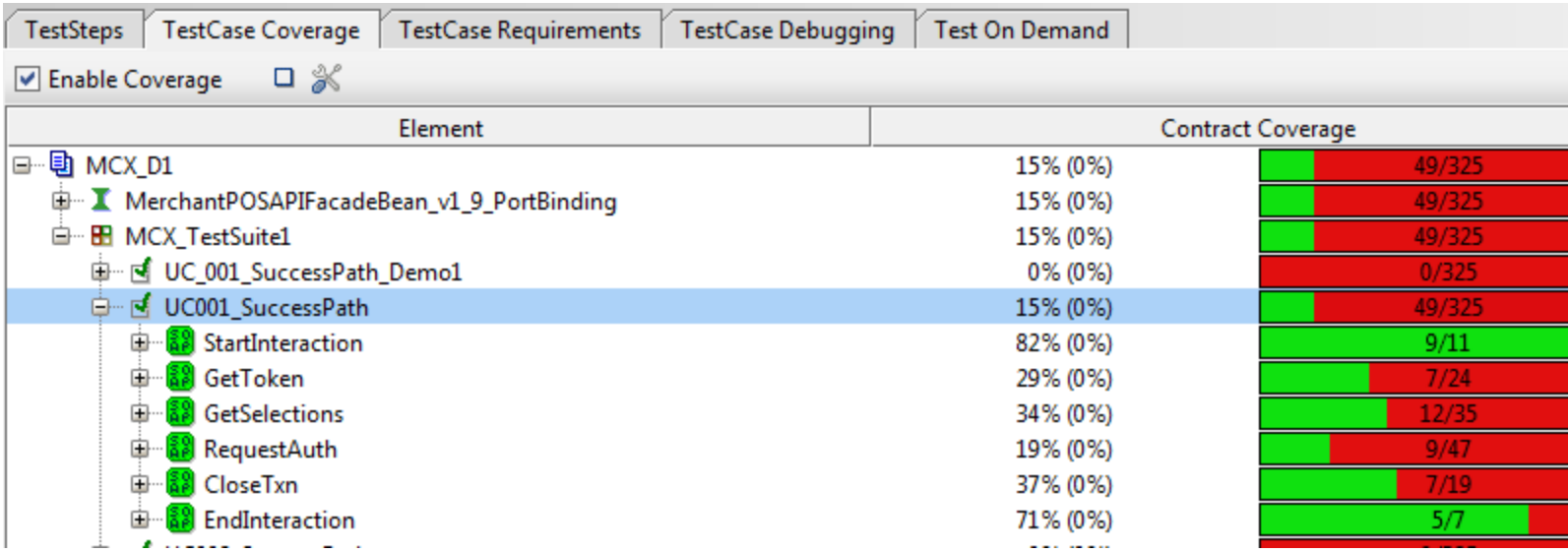
Test Coverage: WSDL Contract

This feature is helpful, but may not be required

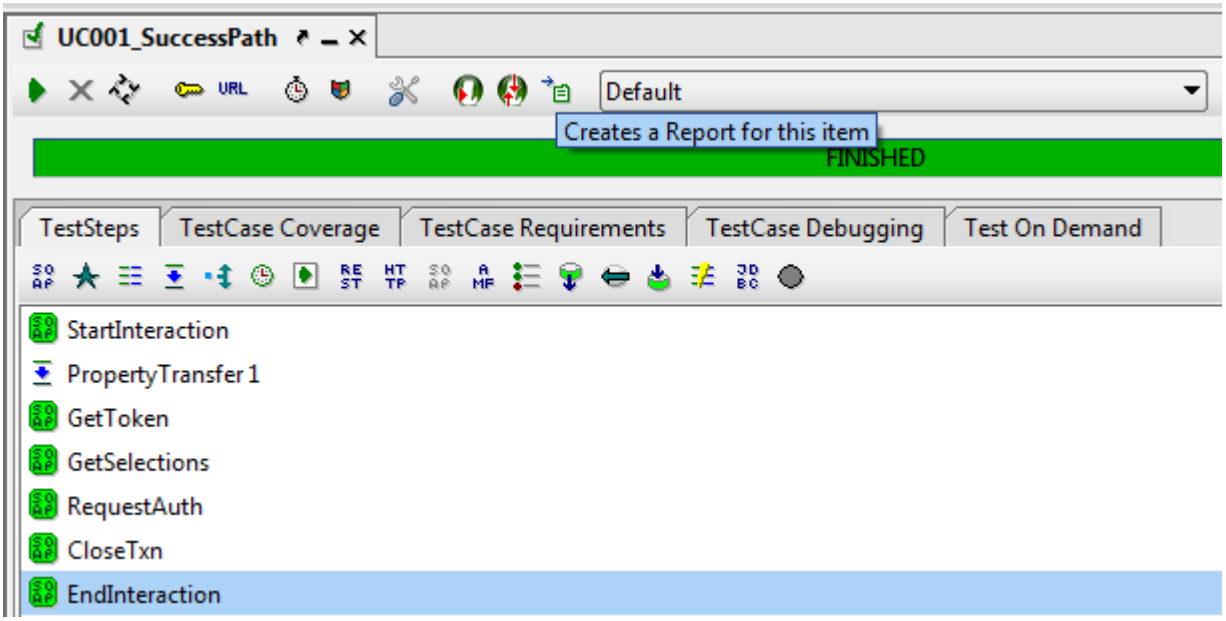
Each request is created from the WSDL contract and it is helpful to know how much of the contract is used in test case



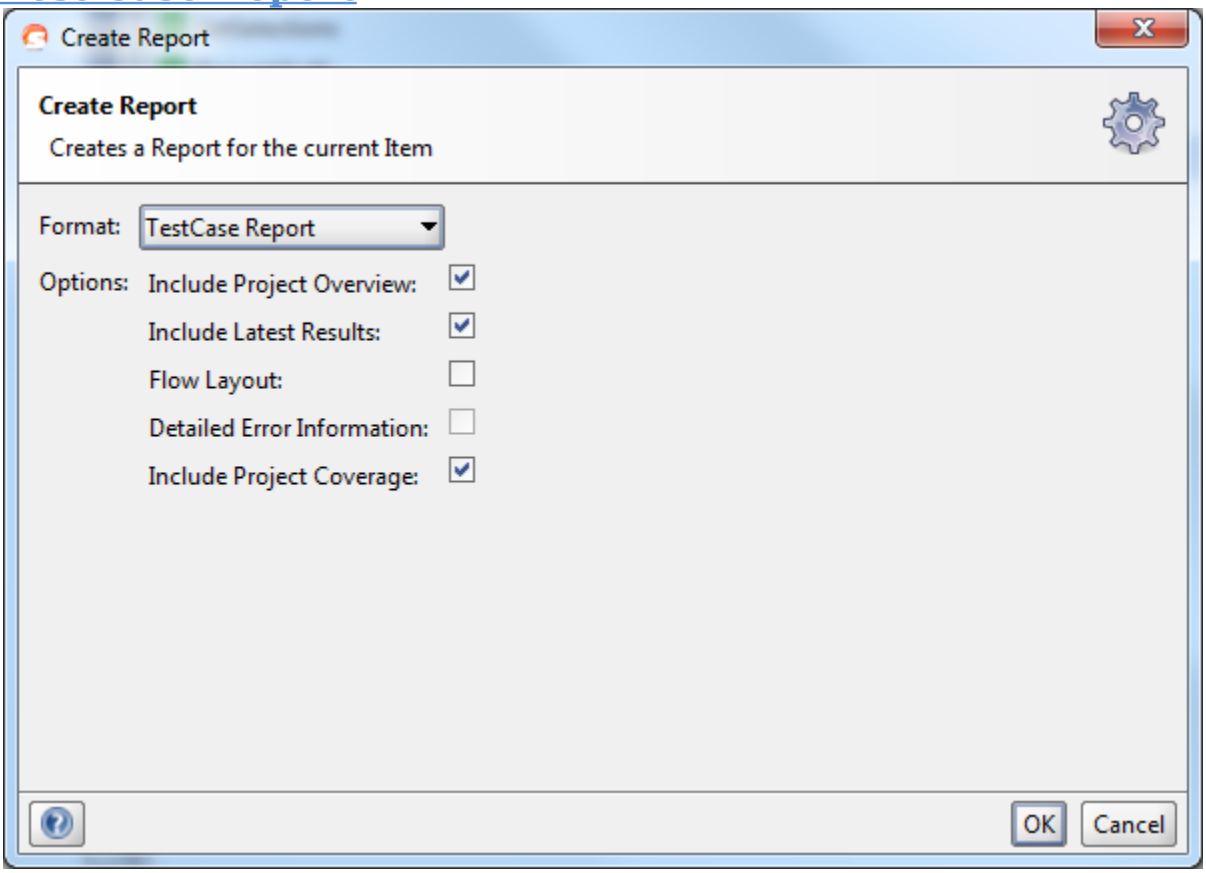
Select Enable Coverage and run the Test Case again

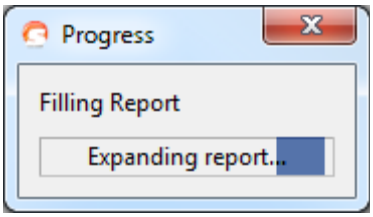


SOAPUI Test Reports



Test Case Report





TestCase Results Report for UC001_SuccessPath

TestCase Metrics

Overview

Project	MCX_D1
TestSuite	MCX_TestSuite1
TestCase name	UC001_SuccessPath
Description	

Base Metrics

Number of TestSteps	7
Number of LoadTests	0
Number of Assertions	18

TestSteps

TestStep Name	TestStep Description
StartInteraction	
PropertyTransfer 1	
GetToken	
GetSelections	
RequestAuth	
CloseTxn	
EndInteraction	

TestCase Properties

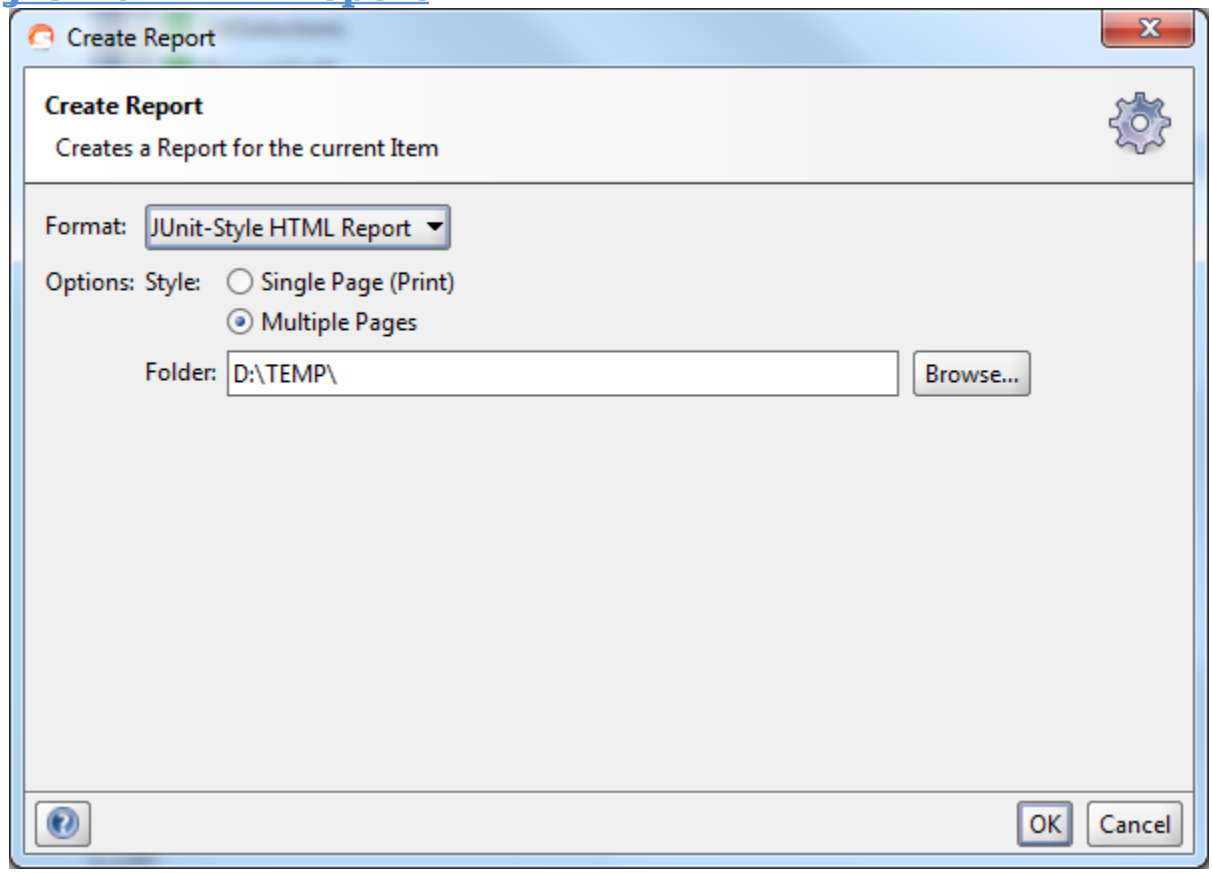
Name	Value

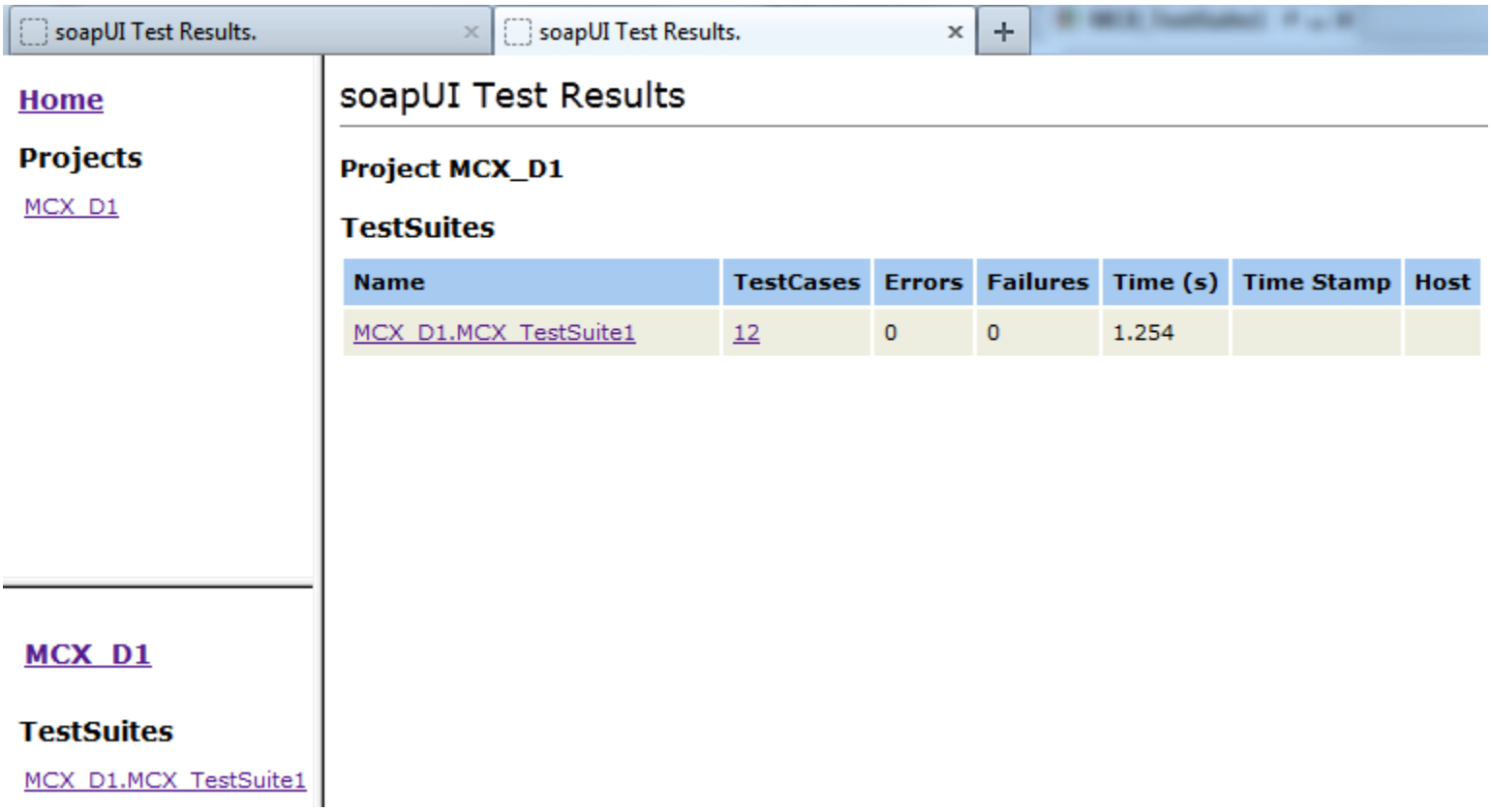
Result Metrics

Result Metrics

Start Time	Wed Jul 24 15:20:48 CDT 2013

J-Unit HTML Report





soapUI Test Results. | soapUI Test Results.

Home

Projects

MCX_D1

TestSuites

MCX_D1.MCX_TestSuite1

soapUI Test Results

TestSuite MCX_D1.MCX_D1.MCX_TestSuite1

Name	TestCases	Errors	Failures	Time (s)	Time Stamp	Host
MCX_D1.MCX_TestSuite1	12	0	0	1.254		

TestCases

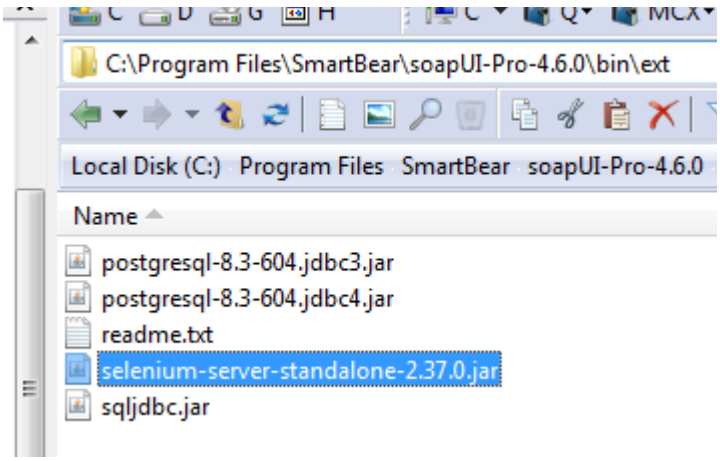
Name	Status	Type	Time (s)
UC001_SuccessPath	Success		0.167
UC002_SuccessPath	Success		0.209
UC003_SuccessPath	Success		0.173
UC004_SuccessPath	Success		0.097
UC005_SuccessPath	Success		0.360
UC006_SuccessPath	Success		0.073
UC007_SuccessPath	Success		0.043
UC008_SuccessPath	Success		0.035
UC009_SuccessPath	Success		0.060
UC010_SuccessPath	Success		0.037
UC011_SuccessPath_WalletUIONLY	Success		0.000
UC012_SuccessPath_WalletUIONLY	Success		0.000

report.zip

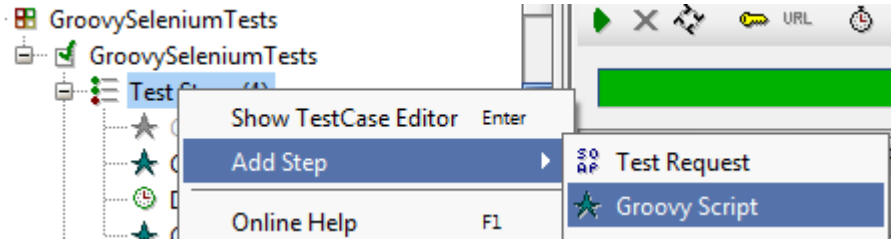
Integrating Selenium

Configuring SoapUI for using Selenium

Download the latest Selenium FireFox Standalone server from seleniumhq.org
Place the JAR file under <soapUI>\bin\ext
Start/Re-Start SoapUI to pick up the file



Create a new Groovy Test Step



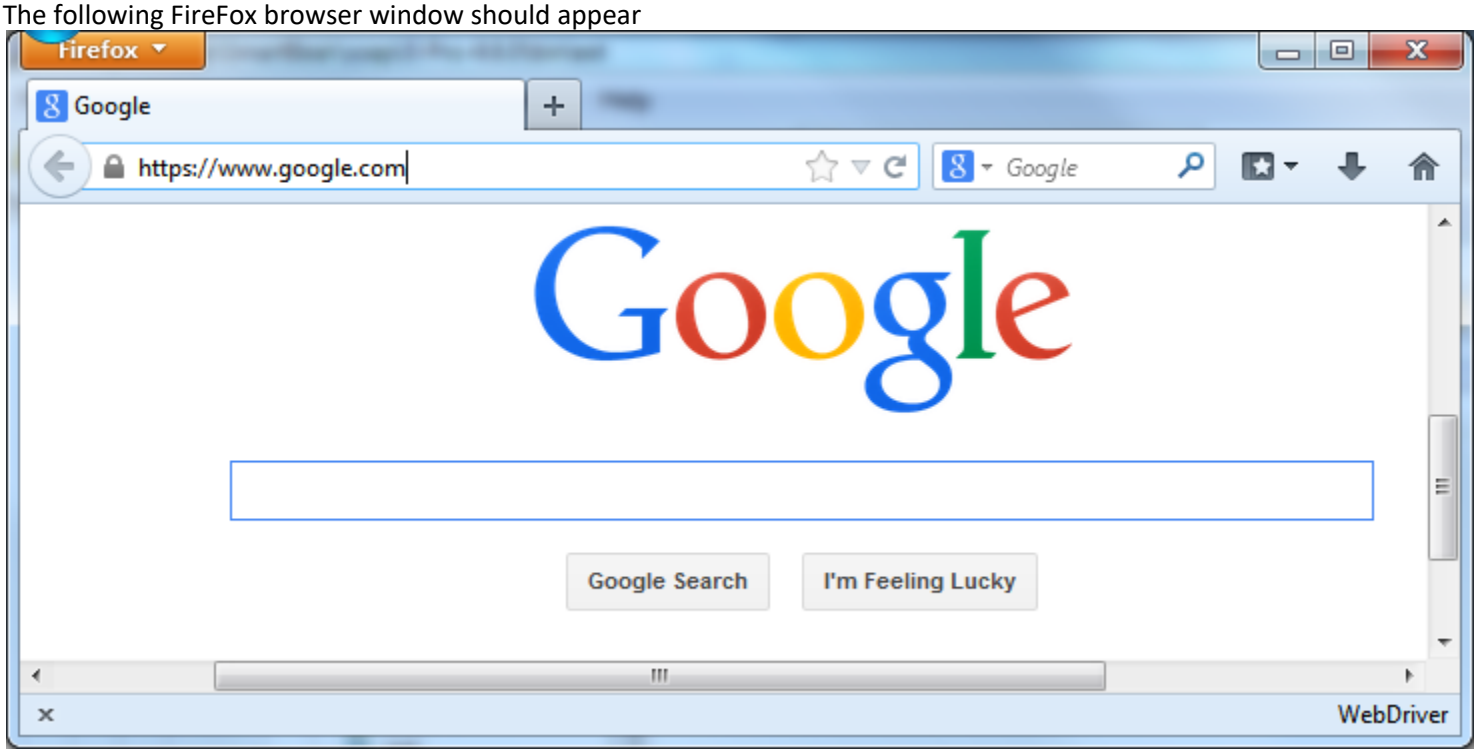
Paste the following code block into the script to test if SoapUI finds the Selenium JAR file

```
import org.openqa.selenium.By
import org.openqa.selenium.WebDriver
import org.openqa.selenium.WebElement
import org.openqa.selenium.firefox.FirefoxDriver
import org.openqa.selenium.support.ui.ExpectedCondition
import org.openqa.selenium.support.ui.WebDriverWait

// Create a new instance of the Firefox driver
// Notice that the remainder of the code relies on the interface,
// not the implementation.
WebDriver driver = new FirefoxDriver()

driver.get("http://www.google.com")
//driver.quit()
```

The following FireFox browser window should appear



You have successfully called selenium commands from SoapUI!!
Note: Remove the '//' from the //driver.quit() line if you want the browser to close when done

WebDriver Status

When the browser is executing commands, the 'WebDriver' text will show in RED

Interacting with Selenium browser elements

Create another Groovy Test Step and paste in the following code to suspend a Merchant
Note: requires the Test Suite Property 'MerchantId' with a valid merchant ID value

```
import org.openqa.selenium.By
import org.openqa.selenium.WebDriver
import org.openqa.selenium.WebElement
import org.openqa.selenium.firefox.FirefoxDriver
import org.openqa.selenium.support.ui.ExpectedCondition
import org.openqa.selenium.support.ui.WebDriverWait

// Create a new instance of the Firefox driver
// Notice that the remainder of the code relies on the interface,
// not the implementation.
WebDriver driver = new FirefoxDriver()

driver.get("https://127.0.0.1:18201/AdminApp/rest/login")
WebElement element = driver.findElement(By.name("identifier"))
element.sendKeys("Admin")
element = driver.findElement(By.name("credential"))
element.sendKeys("9491iI3")
element = driver.findElement(By.id("loginBtn"))
element.click()

def thisMerch     = testRunner.testCase.testSuite.getPropertyValue("MerchantId")
//driver.get("https://127.0.0.1:18201/AdminApp/page/merchant/310HQQ13VBB4/merchant-
details.xhtml")
driver.get("https://127.0.0.1:18201/AdminApp/page/merchant/${thisMerch}/merchant-
details.xhtml")
element = driver.findElement(By.id("suspend"))
element.click()


//Close the browser
//driver.quit()
```

Example of Selenium ScreenShot

Selenium Screenshot requires extra imports:

```
//For ScreenCap
```

```
import java.io.File
import org.apache.commons.io.FileUtils
import org.openqa.selenium.TakesScreenshot
import org.openqa.selenium.OutputType
import java.util.concurrent.TimeUnit
```

Syntax for driver.getScreenShotAs

```
//Get TimeStamp and Save Screenshot
Thread.sleep(3000)
startDate = new Date()
thisDate = startDate.format("yyyy_MMdd'_'HHmmss_SSS")
screen = driver.getScreenshotAs(OutputType.FILE)
FileUtils.copyFile(screen, new File("d:/temp/${thisDate}-screen1.png"))
```

Example usage

```
import org.openqa.selenium.By
import org.openqa.selenium.*
import org.openqa.selenium.WebDriver
import org.openqa.selenium.WebElement
import org.openqa.selenium.firefox.FirefoxDriver
import org.openqa.selenium.support.ui.ExpectedCondition
import org.openqa.selenium.support.ui.WebDriverWait

//For ScreenCap
import java.io.File
import org.apache.commons.io.FileUtils
import org.openqa.selenium.TakesScreenshot
import org.openqa.selenium.OutputType
import java.util.concurrent.TimeUnit

def filePath = testRunner.testCase.testSuite.project.getPropertyValue("logfilePath")
def thisMerch = testRunner.testCase.testSuite.getPropertyValue("MerchantId")

// Create a new instance of the Firefox driver
// Notice that the remainder of the code relies on the interface,
// not the implementation.
WebDriver driver = new FirefoxDriver()
driver.manage().timeouts().implicitlyWait(60, TimeUnit.SECONDS)

// Open Admin App and Login
driver.get("https://127.0.0.1:18201/AdminApp/rest/login")

// Login
WebElement element = driver.findElement(By.name("identifier"))
element.sendKeys("Admin")
element = driver.findElement(By.name("credential"))
element.sendKeys("9491iI3")
element = driver.findElement(By.id("loginBtn"))
element.click()

// Merchant Details By MerchantId
driver.get("https://127.0.0.1:18201/AdminApp/page/merchant/${thisMerch}/merchant-
details.xhtml")


// Check and Release Suspension
```

```
element = driver.findElement(By.id("releaseSuspension"))
if (element.isDisplayed() ) {
      log.info "Merchant is Suspended"
      } else {
      element = driver.findElement(By.id("suspend"))
      if (element.isDisplayed() ) {
            log.info "Merchant is Not Suspended"
            element.click()
      }
}

//Get TimeStamp and Save Screenshot
Thread.sleep(3000)
startDate = new Date()
thisDate = startDate.format("yyyy_MMdd'_'HHmmss_SSS")
screen = driver.getScreenshotAs(OutputType.FILE)
FileUtils.copyFile(screen, new File("d:/temp/${thisDate}-screen1.png"))

// Check and Suspend
element = driver.findElement(By.id("releaseSuspension"))
if (element.isDisplayed() ) {
      log.info "Merchant is already Suspended"
      } else {
      element = driver.findElement(By.id("suspend"))
      if (element.isDisplayed() ) {
            log.info "Merchant is Not Suspended, Suspending...."
            element.click()
      }
}

//Get TimeStamp and Save Screenshot
Thread.sleep(3000)
def startDate = new Date()
thisDate = startDate.format("yyyy_MMdd'_'HHmmss_SSS")
def screen = driver.getScreenshotAs(OutputType.FILE)
FileUtils.copyFile(screen, new File("d:/temp/${thisDate}-screen2.png"))

// Check and Release Suspension
element = driver.findElement(By.id("releaseSuspension"))
if (element.isDisplayed() ) {
      log.info "Merchant is Suspended"
      element.click()
      } else {
      element = driver.findElement(By.id("suspend"))
      if (element.isDisplayed() ) {
            log.info "Merchant is Not Suspended"
      }
}

//Get TimeStamp and Save Screenshot
Thread.sleep(3000)
startDate = new Date()
thisDate = startDate.format("yyyy_MMdd'_'HHmmss_SSS")
screen = driver.getScreenshotAs(OutputType.FILE)
FileUtils.copyFile(screen, new File("d:/temp/${thisDate}-screen3.png"))


//Close the browser
//driver.quit()
```

Finding a table and Iterating its rows (for Payment Pockets)

Example

```
// Pass values to an array, THEN use array to get links...

//Detect Pockets and push to array
for (int i=1; i<rowCount+1; i++){
      thisCustPockets = driver.findElement(By.xpath("//table[@id='default-pocket-
table']/tbody/tr[${i}]"))
      thisCustPocket = thisCustPockets.getAttribute('pocketid')
      log.info "Current Pocket: " + thisCustPocket
      pocketList.add(thisCustPocket)
}

// Suspend via Iteration
for (j in pocketList){
      driver.get("https://127.0.0.1:18201/AdminApp/page/customer/${thisCust}/pocket/$
{j}.xhtml")
      log.info j + ": Checking visible links"
      Thread.sleep(3000)
      element = driver.findElement(By.id("releaseSuspension"))
      if (element.isDisplayed() ) {
            // RELEASE
            log.info "Customer Pocket -${j}- is already Suspended"
      } else {
            // SUSPEND
            element = driver.findElement(By.id("suspend"))
            if (element.isDisplayed() ) {
                  log.info "Customer Pocket -${j}- is Not Suspended"
                  element.click()
                  log.info "Customer Pocket -${j}- is NOW Suspended"
            }
      }
}
```

Obtaining and Formatting Date/Time

Example

```
startDate = new Date()
thisDate = startDate.format("yyyy_MMdd'_'HHmmss_SSS")
screen = driver.getScreenshotAs(OutputType.FILE)
FileUtils.copyFile(screen, new File("d:/temp/${thisDate}-screen2.png"))
```

Integrating Eclipse Android ADT with SoapUI

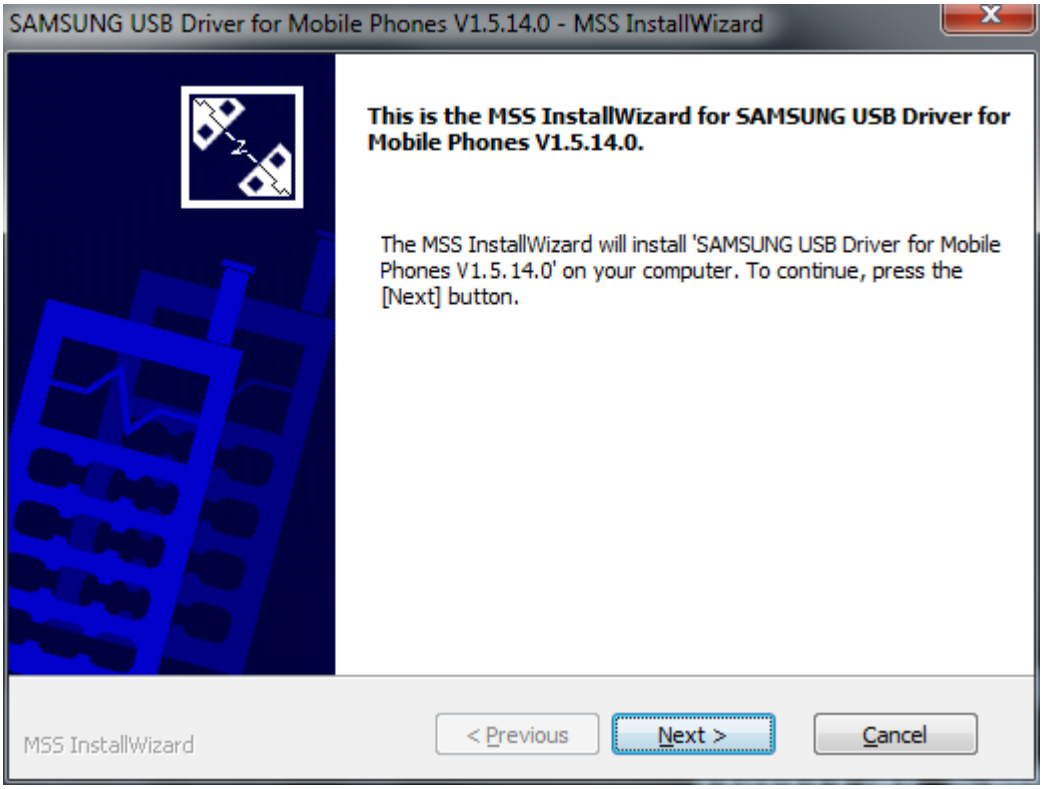
Download:

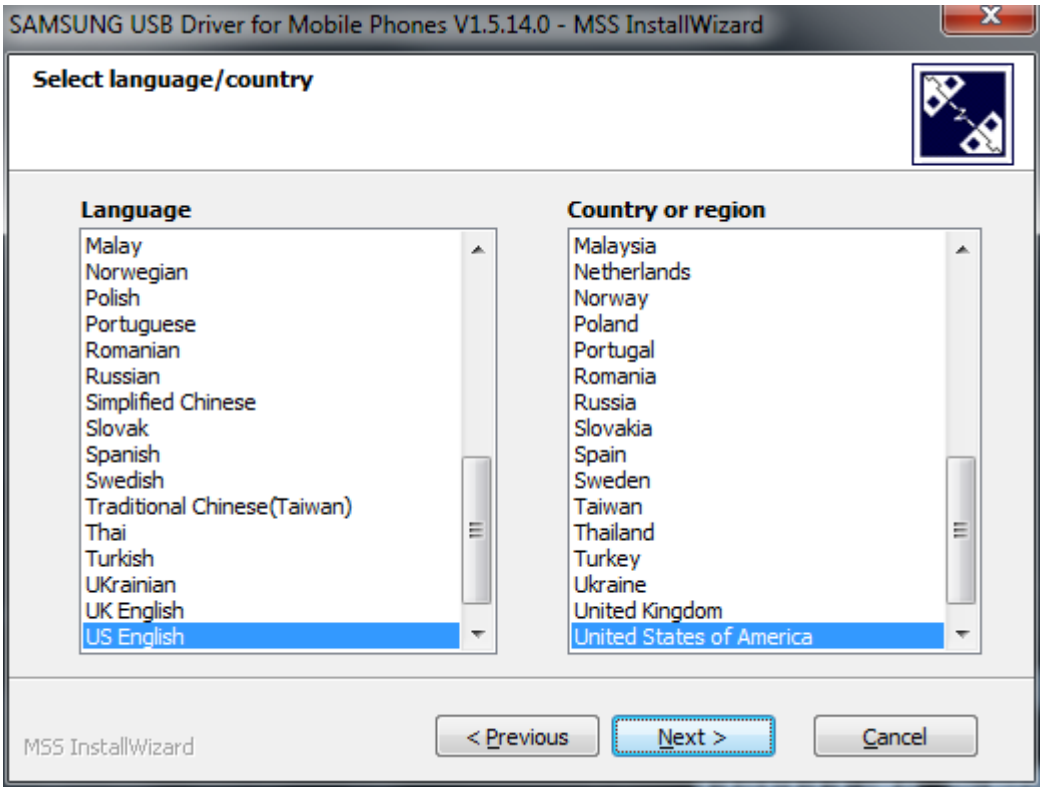
- Android ADT/Eclipse: (adt-bundle-windows-x86-20130729)

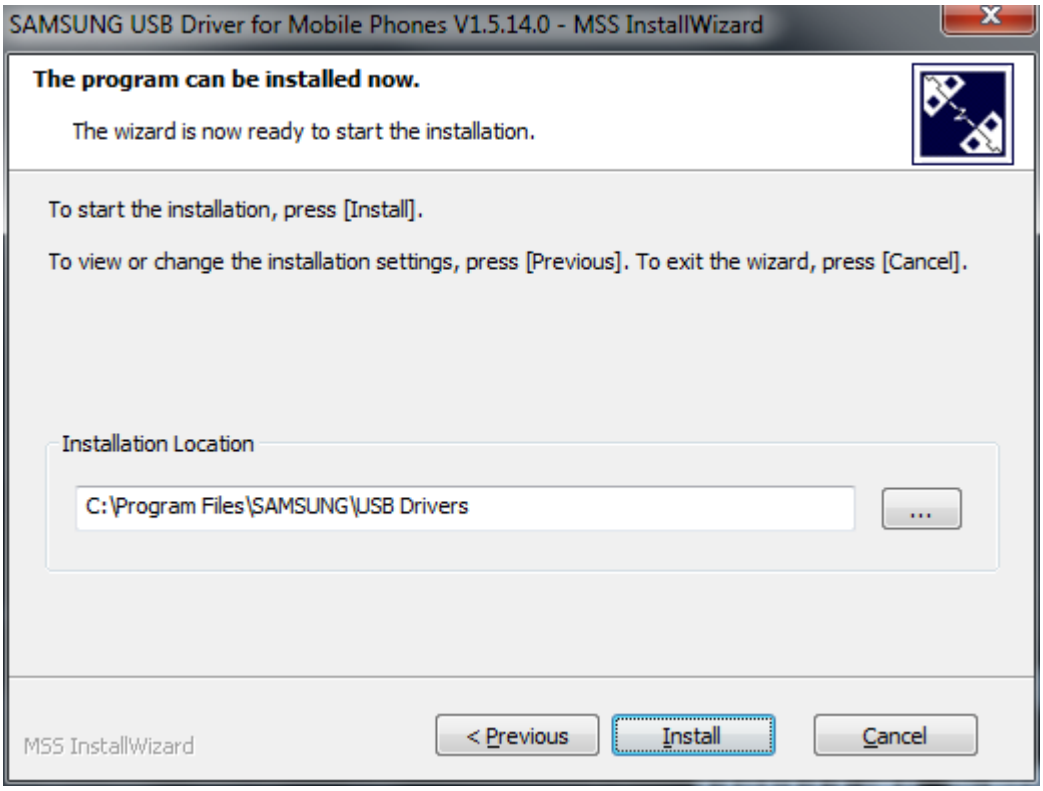
- Android SDK
- Samsung USB Drivers (Samsung_USB_Driver_for_Mobile_Phones_v1.5.14.0.exe)
    - http://www.samsung.com/us/support/downloads
-

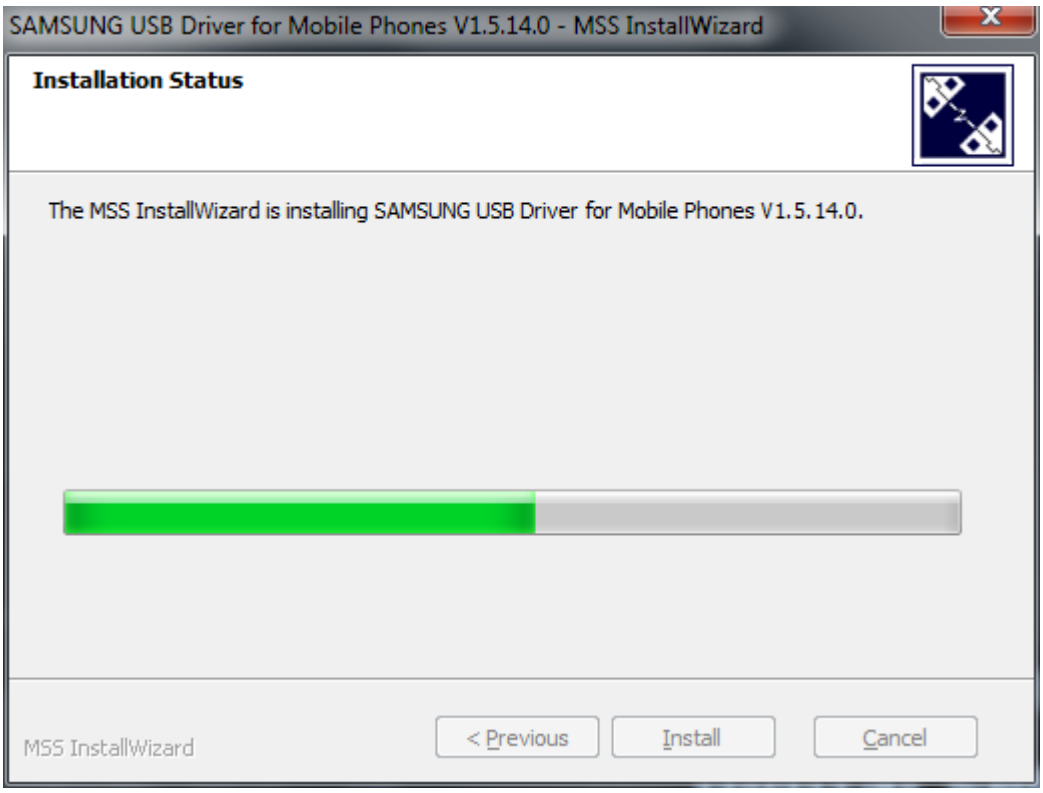
In Eclipse:
Get Groovy http://dist.springsource.org/release/GRECLIPSE/e4.3/

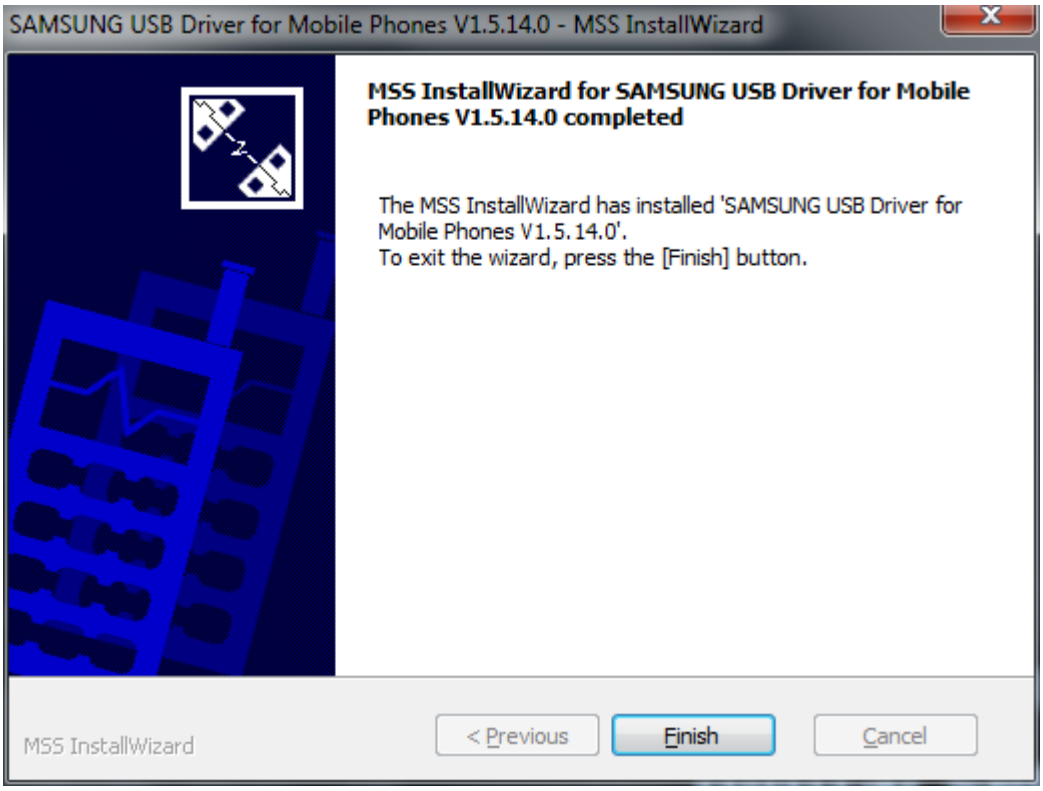
Install Samsung USB Drivers

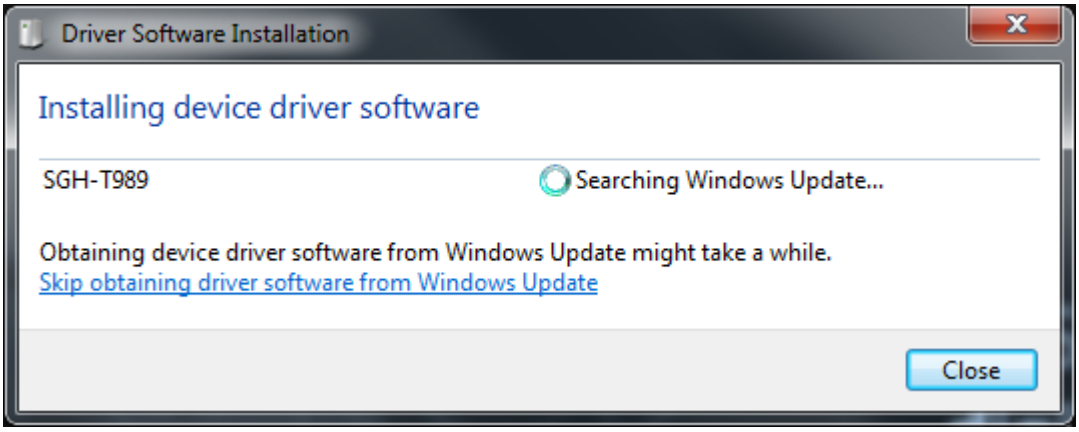


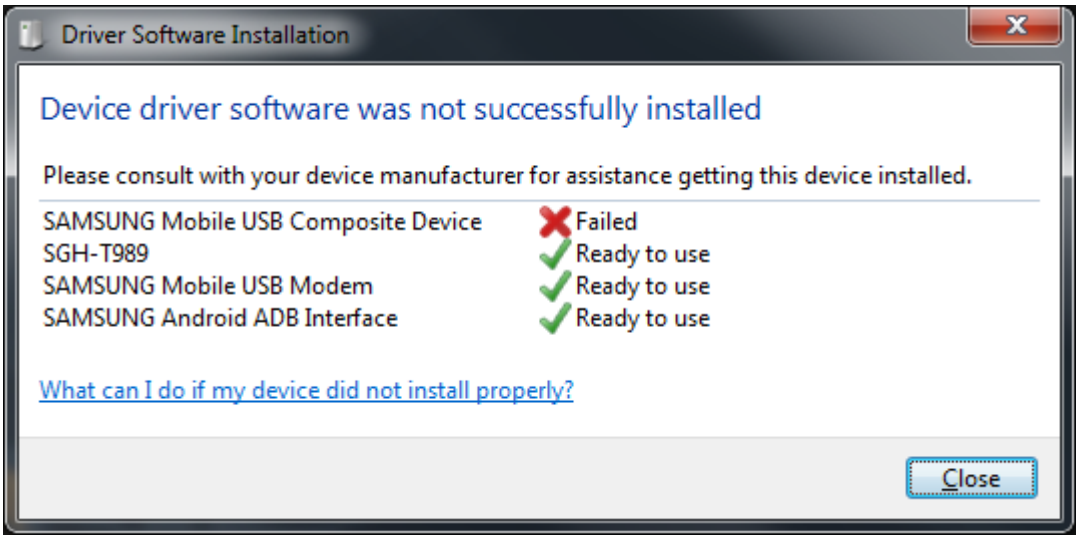


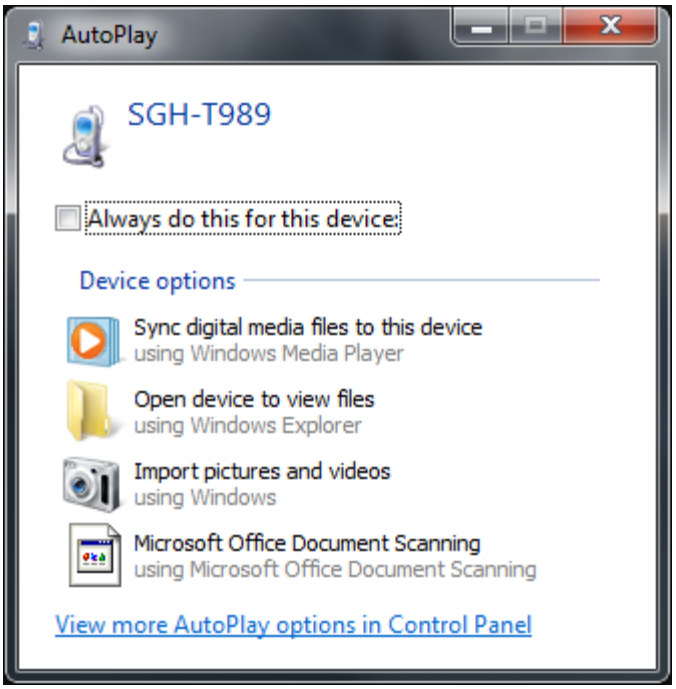

SAMSUNG USB Driver for Mobile Phones V1.5.14.0 - MSS InstallWizard
The program can be installed now.
The wizard is now ready to start the installation.
To start the installation, press [Install].
To view or change the installation settings, press [Previous]. To exit the wizard, press [Cancel].
Installation Location
C:\Program Files\SAMSUNG\USB Drivers
...
MSS InstallWizard
< Previous
Install
Cancel


SAMSUNG USB Driver for Mobile Phones V1.5.14.0 - MSS InstallWizard
Installation Status
The MSS InstallWizard is installing SAMSUNG USB Driver for Mobile Phones V1.5.14.0.
MSS InstallWizard
< Previous
Install
Cancel


SAMSUNG USB Driver for Mobile Phones V1.5.14.0 - MSS InstallWizard
MSS InstallWizard for SAMSUNG USB Driver for Mobile Phones V1.5.14.0 completed
The MSS InstallWizard has installed 'SAMSUNG USB Driver for Mobile Phones V1.5.14.0'.
To exit the wizard, press the [Finish] button.
MSS InstallWizard
< Previous
Finish
Cancel


Driver Software Installation
Installing device driver software
SGH-T989
Searching Windows Update...
Obtaining device driver software from Windows Update might take a while.
Skip obtaining driver software from Windows Update
Close


Driver Software Installation
Device driver software was not successfully installed
Please consult with your device manufacturer for assistance getting this device installed.
SAMSUNG Mobile USB Composite Device
Failed
SGH-T989
Ready to use
SAMSUNG Mobile USB Modem
Ready to use
SAMSUNG Android ADB Interface
Ready to use
What can I do if my device did not install properly?
Close



ADB: UI Interactions

input text <string>
input keyevent <key code number or name>
input tap <x> <y>
input swipe <x1> <y1> <x2> <y2>

ADB: Sending buttons to Android

http://developer.android.com/reference/android/view/KeyEvent.html

adb shell input keyevent 3 HOME

ADB: Sending touches to Android

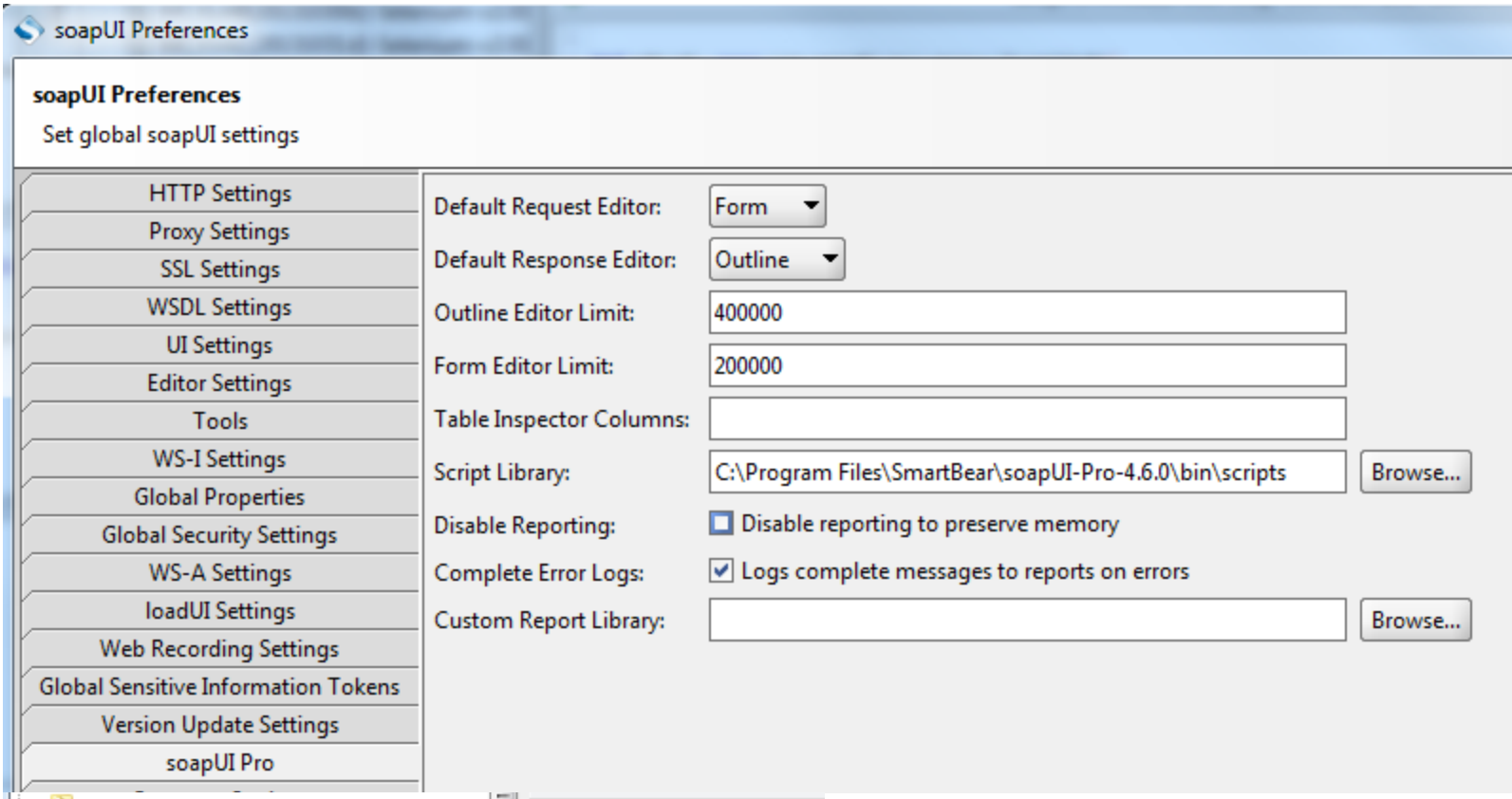
<EclipseADT>/sdk/platform-tools/
adb shell input tap 245 440
adb shell input tap 245 540
adb shell input tap 245 640
adb shell input tap 245 750

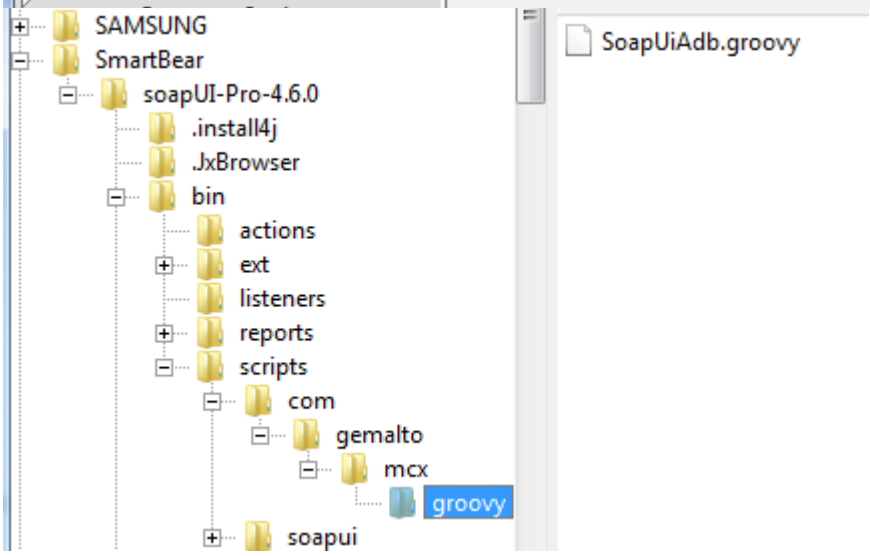
ADB: Swiping Screen

adb shell swipe 240 770 240 240

Groovy Script Library and Groovy Classes

Script Library Location

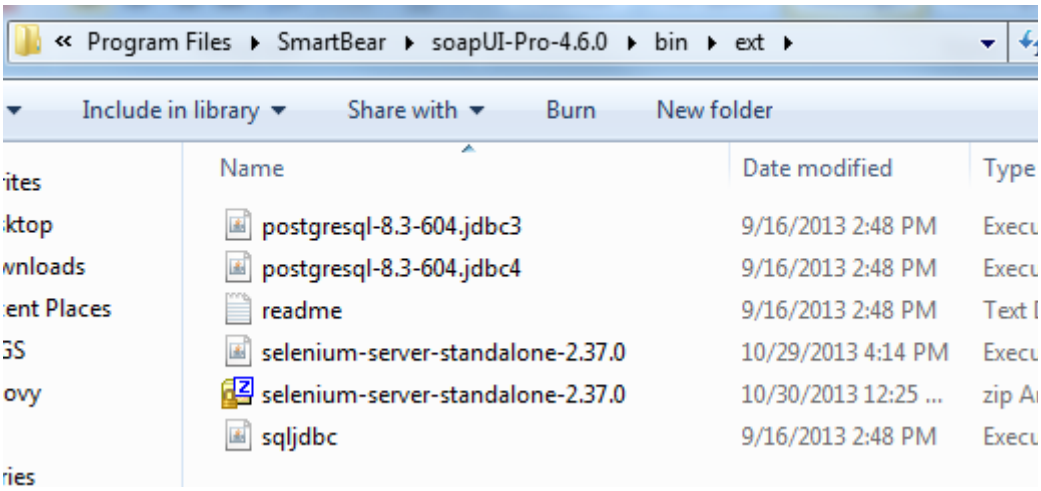




SoapUI Extensions/JAR file locations

When extending SoapUI with custom features or including other sources, place files under the <soapUI>/bin/ext folder

Restart SoapUI if it doesn't immediately recognize the files



Creating and Calling a Groovy Class in the Library

By creating the following groovy class, calls can be made from ANY Groovy script in the soap project:

Class to Send a HOME key press

```
package com.gemalto.mcx.groovy

class SoapUiAdb {
	def adbexe
	SoapUiAdb(){
		adbexe = "V:\\DEVL\\EclipseADT\\sdk\\platform-tools\\adb.exe"
	}

	def AdbShellExec(){
		//A string can be executed in the standard java way:
		def command = """${adbexe} shell input keyevent 3"""// Create the String
		//log.info "Executing: ${command}"
		def proc = command.execute()  // Call *execute* on the string
		proc.waitFor()                               // Wait for the command to finish

		// Obtain status and output
		//log.info "return code: ${ proc.exitValue()}"
		//log.info "stderr: ${proc.err.text}"
		//log.info "stdout: ${proc.in.text}" // *out* from the external program is *in* for
groovy
	}
}
```

Calling the class from SoapUI Groovy Script

```
def adbcall = new com.gemalto.mcx.groovy.SoapUiAdb()
adbcall.AdbShellExec()
```

Passing the testRunner and log objects to external Groovy Scripts

Call:

```
new com.gemalto.mcx.groovy.SoapUiSelenium(log,testRunner).MFS_Login()
```

the above call is a compound statement

1) Instantiate the class
    a. new com.gemalto.mcx.groovy.SoapUiSelenium(log,testRunner)
    b. pass log Object = for log.info
    c. pass testRunner = for access to properties, and all things soapui
2) Call the method
    a. .MFS_Login()

Class and Method:

```
package com.gemalto.mcx.groovy

class SoapUiSelenium {


	/*
	 *  Calls to this CLASS MUST pass the *log* and *testRunner* objects
	 *   log: used for sending log info to SoapUI
	 *   testRunner: provides hooks for Test Case/Suite/Project/Environment Properties
	 *     along with anything that is accessible by using the 'Groovy Script' Test Step
	 *     directly within SoapUI
	 */
	def log;
	def testRunner;
	WebDriver testBrowser = new FirefoxDriver();

		def SoapUiSelenium(log, testRunner){
		this.log = log;
		this.testRunner = testRunner;
		// Create a new instance of the Firefox driver
		//WebDriver driver = new FirefoxDriver()
		testBrowser.manage().window().setPosition(new Point(20,20))
		testBrowser.manage().window().setSize(new Dimension(750,500))

	}


	def MFS_Login(){
		// Open Admin App and Login
		testBrowser.get("https://127.0.0.1:18202/AdminApp/rest/login")
		WebElement element = testBrowser.findElement(By.name("identifier"))
		element.sendKeys("soapui_ma")
		element = testBrowser.findElement(By.name("credential"))
		element.sendKeys("SoapUI_01")
		element = testBrowser.findElement(By.id("loginBtn"))
		element.click()

	}
```

Appendices

ADB: keyevents

Constants		
int	ACTION_DOWN	`getAction()` value: the key has been pressed down.
int	ACTION_MULTIPLE	`getAction()` value: multiple duplicate key events have occurred in a row, or a complex string is being delivered.
int	ACTION_UP	`getAction()` value: the key has been released.
int	FLAG_CANCELED	When associated with up key events, this indicates that the key press has been canceled.
int	FLAG_CANCELED_LONG_PRESS	Set when a key event has `FLAG_CANCELED` set because a long press action was executed while it was down.
int	FLAG_EDITOR_ACTION	This mask is used for compatibility, to identify enter keys that are coming from an IME whose enter key has been auto-labelled "next" or "done".
int	FLAG_FALLBACK	Set when a key event has been synthesized to implement default behavior for an event that the application did not handle.
int	FLAG_FROM_SYSTEM	This mask is set if an event was known to come from a trusted part of the system.
int	FLAG_KEEP_TOUCH_MODE	This mask is set if we don't want the key event to cause us to leave touch mode.
int	FLAG_LONG_PRESS	This flag is set for the first key repeat that occurs after the long press timeout.
int	FLAG_SOFT_KEYBOARD	This mask is set if the key event was generated by a software keyboard.
int	FLAG_TRACKING	Set for `ACTION_UP` when this event's key code is still being tracked from its initial down.
int	FLAG_VIRTUAL_HARD_KEY	This key event was generated by a virtual (on-screen) hard key area.
int	FLAG_WOKE_HERE	This mask is set if the device woke because of this key event.

int	KEYCODE_0	Key code constant: '0' key.
int	KEYCODE_1	Key code constant: '1' key.
int	KEYCODE_2	Key code constant: '2' key.
int	KEYCODE_3	Key code constant: '3' key.
int	KEYCODE_3D_MODE	Key code constant: 3D Mode key.
int	KEYCODE_4	Key code constant: '4' key.
int	KEYCODE_5	Key code constant: '5' key.
int	KEYCODE_6	Key code constant: '6' key.
int	KEYCODE_7	Key code constant: '7' key.
int	KEYCODE_8	Key code constant: '8' key.
int	KEYCODE_9	Key code constant: '9' key.
int	KEYCODE_A	Key code constant: 'A' key.
int	KEYCODE_ALT_LEFT	Key code constant: Left Alt modifier key.
int	KEYCODE_ALT_RIGHT	Key code constant: Right Alt modifier key.
int	KEYCODE_APOSTROPHE	Key code constant: ''' (apostrophe) key.
int	KEYCODE_APP_SWITCH	Key code constant: App switch key.
int	KEYCODE_ASSIST	Key code constant: Assist key.

t		
int	KEYCODE_AT	Key code constant: '@' key.
int	KEYCODE_AVR_INPUT	Key code constant: A/V Receiver input key.
int	KEYCODE_AVR_POWER	Key code constant: A/V Receiver power key.
int	KEYCODE_B	Key code constant: 'B' key.
int	KEYCODE_BACK	Key code constant: Back key.
int	KEYCODE_BACKSLASH	Key code constant: '\' key.
int	KEYCODE_BOOKMARK	Key code constant: Bookmark key.
int	KEYCODE_BREAK	Key code constant: Break / Pause key.
int	KEYCODE_BRIGHTNESS_DOWN	Key code constant: Brightness Down key.
int	KEYCODE_BRIGHTNESS_UP	Key code constant: Brightness Up key.
int	KEYCODE_BUTTON_1	Key code constant: Generic Game Pad Button #1.
int	KEYCODE_BUTTON_10	Key code constant: Generic Game Pad Button #10.
int	KEYCODE_BUTTON_11	Key code constant: Generic Game Pad Button #11.
int	KEYCODE_BUTTON_12	Key code constant: Generic Game Pad Button #12.
int	KEYCODE_BUTTON_13	Key code constant: Generic Game Pad Button #13.
int	KEYCODE_BUTTON_14	Key code constant: Generic Game Pad Button #14.

int	KEYCODE_BUTTON_15	Key code constant: Generic Game Pad Button #15.
int	KEYCODE_BUTTON_16	Key code constant: Generic Game Pad Button #16.
int	KEYCODE_BUTTON_2	Key code constant: Generic Game Pad Button #2.
int	KEYCODE_BUTTON_3	Key code constant: Generic Game Pad Button #3.
int	KEYCODE_BUTTON_4	Key code constant: Generic Game Pad Button #4.
int	KEYCODE_BUTTON_5	Key code constant: Generic Game Pad Button #5.
int	KEYCODE_BUTTON_6	Key code constant: Generic Game Pad Button #6.
int	KEYCODE_BUTTON_7	Key code constant: Generic Game Pad Button #7.
int	KEYCODE_BUTTON_8	Key code constant: Generic Game Pad Button #8.
int	KEYCODE_BUTTON_9	Key code constant: Generic Game Pad Button #9.
int	KEYCODE_BUTTON_A	Key code constant: A Button key.
int	KEYCODE_BUTTON_B	Key code constant: B Button key.
int	KEYCODE_BUTTON_C	Key code constant: C Button key.
int	KEYCODE_BUTTON_L1	Key code constant: L1 Button key.
int	KEYCODE_BUTTON_L2	Key code constant: L2 Button key.
int	KEYCODE_BUTTON_MODE	Key code constant: Mode Button key.
in	KEYCODE_BUTTON_R1	Key code constant: R1 Button key.

t		
int	KEYCODE_BUTTON_R2	Key code constant: R2 Button key.
int	KEYCODE_BUTTON_SELECT	Key code constant: Select Button key.
int	KEYCODE_BUTTON_START	Key code constant: Start Button key.
int	KEYCODE_BUTTON_THUMBL	Key code constant: Left Thumb Button key.
int	KEYCODE_BUTTON_THUMBR	Key code constant: Right Thumb Button key.
int	KEYCODE_BUTTON_X	Key code constant: X Button key.
int	KEYCODE_BUTTON_Y	Key code constant: Y Button key.
int	KEYCODE_BUTTON_Z	Key code constant: Z Button key.
int	KEYCODE_C	Key code constant: 'C' key.
int	KEYCODE_CALCULATOR	Key code constant: Calculator special function key.
int	KEYCODE_CALENDAR	Key code constant: Calendar special function key.
int	KEYCODE_CALL	Key code constant: Call key.
int	KEYCODE_CAMERA	Key code constant: Camera key.
int	KEYCODE_CAPS_LOCK	Key code constant: Caps Lock key.
int	KEYCODE_CAPTIONS	Key code constant: Toggle captions key.
int	KEYCODE_CHANNEL_DOWN	Key code constant: Channel down key.

int	KEYCODE_CHANNEL_UP	Key code constant: Channel up key.
int	KEYCODE_CLEAR	Key code constant: Clear key.
int	KEYCODE_COMMA	Key code constant: ',' key.
int	KEYCODE_CONTACTS	Key code constant: Contacts special function key.
int	KEYCODE_CTRL_LEFT	Key code constant: Left Control modifier key.
int	KEYCODE_CTRL_RIGHT	Key code constant: Right Control modifier key.
int	KEYCODE_D	Key code constant: 'D' key.
int	KEYCODE_DEL	Key code constant: Backspace key.
int	KEYCODE_DPAD_CENTER	Key code constant: Directional Pad Center key.
int	KEYCODE_DPAD_DOWN	Key code constant: Directional Pad Down key.
int	KEYCODE_DPAD_LEFT	Key code constant: Directional Pad Left key.
int	KEYCODE_DPAD_RIGHT	Key code constant: Directional Pad Right key.
int	KEYCODE_DPAD_UP	Key code constant: Directional Pad Up key.
int	KEYCODE_DVR	Key code constant: DVR key.
int	KEYCODE_E	Key code constant: 'E' key.
int	KEYCODE_EISU	Key code constant: Japanese alphanumeric key.
in	KEYCODE_ENDCALL	Key code constant: End Call key.

t		
int	KEYCODE_ENTER	Key code constant: Enter key.
int	KEYCODE_ENVELOPE	Key code constant: Envelope special function key.
int	KEYCODE_EQUALS	Key code constant: '=' key.
int	KEYCODE_ESCAPE	Key code constant: Escape key.
int	KEYCODE_EXPLORER	Key code constant: Explorer special function key.
int	KEYCODE_F	Key code constant: 'F' key.
int	KEYCODE_F1	Key code constant: F1 key.
int	KEYCODE_F10	Key code constant: F10 key.
int	KEYCODE_F11	Key code constant: F11 key.
int	KEYCODE_F12	Key code constant: F12 key.
int	KEYCODE_F2	Key code constant: F2 key.
int	KEYCODE_F3	Key code constant: F3 key.
int	KEYCODE_F4	Key code constant: F4 key.
int	KEYCODE_F5	Key code constant: F5 key.
int	KEYCODE_F6	Key code constant: F6 key.
int	KEYCODE_F7	Key code constant: F7 key.

int	KEYCODE_F8	Key code constant: F8 key.
int	KEYCODE_F9	Key code constant: F9 key.
int	KEYCODE_FOCUS	Key code constant: Camera Focus key.
int	KEYCODE_FORWARD	Key code constant: Forward key.
int	KEYCODE_FORWARD_DEL	Key code constant: Forward Delete key.
int	KEYCODE_FUNCTION	Key code constant: Function modifier key.
int	KEYCODE_G	Key code constant: 'G' key.
int	KEYCODE_GRAVE	Key code constant: '`' (backtick) key.
int	KEYCODE_GUIDE	Key code constant: Guide key.
int	KEYCODE_H	Key code constant: 'H' key.
int	KEYCODE_HEADSETHOOK	Key code constant: Headset Hook key.
int	KEYCODE_HENKAN	Key code constant: Japanese conversion key.
int	KEYCODE_HOME	Key code constant: Home key.
int	KEYCODE_I	Key code constant: 'I' key.
int	KEYCODE_INFO	Key code constant: Info key.
int	KEYCODE_INSERT	Key code constant: Insert key.
in	KEYCODE_J	Key code constant: 'J' key.

t		
int	KEYCODE_K	Key code constant: 'K' key.
int	KEYCODE_KANA	Key code constant: Japanese kana key.
int	KEYCODE_KATAKANA_HIRAGANA	Key code constant: Japanese katakana / hiragana key.
int	KEYCODE_L	Key code constant: 'L' key.
int	KEYCODE_LANGUAGE_SWITCH	Key code constant: Language Switch key.
int	KEYCODE_LEFT_BRACKET	Key code constant: '[' key.
int	KEYCODE_M	Key code constant: 'M' key.
int	KEYCODE_MANNER_MODE	Key code constant: Manner Mode key.
int	KEYCODE_MEDIA_AUDIO_TRACK	Key code constant: Audio Track key Switches the audio tracks.
int	KEYCODE_MEDIA_CLOSE	Key code constant: Close media key.
int	KEYCODE_MEDIA_EJECT	Key code constant: Eject media key.
int	KEYCODE_MEDIA_FAST_FORWARD	Key code constant: Fast Forward media key.
int	KEYCODE_MEDIA_NEXT	Key code constant: Play Next media key.
int	KEYCODE_MEDIA_PAUSE	Key code constant: Pause media key.
int	KEYCODE_MEDIA_PLAY	Key code constant: Play media key.
int	KEYCODE_MEDIA_PLAY_PAUSE	Key code constant: Play/Pause media key.

int	KEYCODE_MEDIA_PREVIOUS	Key code constant: Play Previous media key.
int	KEYCODE_MEDIA_RECORD	Key code constant: Record media key.
int	KEYCODE_MEDIA_REWIND	Key code constant: Rewind media key.
int	KEYCODE_MEDIA_STOP	Key code constant: Stop media key.
int	KEYCODE_MENU	Key code constant: Menu key.
int	KEYCODE_META_LEFT	Key code constant: Left Meta modifier key.
int	KEYCODE_META_RIGHT	Key code constant: Right Meta modifier key.
int	KEYCODE_MINUS	Key code constant: '-'.
int	KEYCODE_MOVE_END	Key code constant: End Movement key.
int	KEYCODE_MOVE_HOME	Key code constant: Home Movement key.
int	KEYCODE_MUHENKAN	Key code constant: Japanese non-conversion key.
int	KEYCODE_MUSIC	Key code constant: Music special function key.
int	KEYCODE_MUTE	Key code constant: Mute key.
int	KEYCODE_N	Key code constant: 'N' key.
int	KEYCODE_NOTIFICATION	Key code constant: Notification key.
int	KEYCODE_NUM	Key code constant: Number modifier key.
int	KEYCODE_NUMPAD_0	Key code constant: Numeric keypad '0' key.

t		
int	KEYCODE_NUMPAD_1	Key code constant: Numeric keypad '1' key.
int	KEYCODE_NUMPAD_2	Key code constant: Numeric keypad '2' key.
int	KEYCODE_NUMPAD_3	Key code constant: Numeric keypad '3' key.
int	KEYCODE_NUMPAD_4	Key code constant: Numeric keypad '4' key.
int	KEYCODE_NUMPAD_5	Key code constant: Numeric keypad '5' key.
int	KEYCODE_NUMPAD_6	Key code constant: Numeric keypad '6' key.
int	KEYCODE_NUMPAD_7	Key code constant: Numeric keypad '7' key.
int	KEYCODE_NUMPAD_8	Key code constant: Numeric keypad '8' key.
int	KEYCODE_NUMPAD_9	Key code constant: Numeric keypad '9' key.
int	KEYCODE_NUMPAD_ADD	Key code constant: Numeric keypad '+' key (for addition).
int	KEYCODE_NUMPAD_COMMA	Key code constant: Numeric keypad ',' key (for decimals or digit grouping).
int	KEYCODE_NUMPAD_DIVIDE	Key code constant: Numeric keypad '/' key (for division).
int	KEYCODE_NUMPAD_DOT	Key code constant: Numeric keypad '.' key (for decimals or digit grouping).
int	KEYCODE_NUMPAD_ENTER	Key code constant: Numeric keypad Enter key.
int	KEYCODE_NUMPAD_EQUALS	Key code constant: Numeric keypad '=' key.
int	KEYCODE_NUMPAD_LEFT_PAREN	Key code constant: Numeric keypad '(' key.

int	KEYCODE_NUMPAD_MULTIPLY	Key code constant: Numeric keypad '*' key (for multiplication).
int	KEYCODE_NUMPAD_RIGHT_PAREN	Key code constant: Numeric keypad ')' key.
int	KEYCODE_NUMPAD_SUBTRACT	Key code constant: Numeric keypad '-' key (for subtraction).
int	KEYCODE_NUM_LOCK	Key code constant: Num Lock key.
int	KEYCODE_O	Key code constant: 'O' key.
int	KEYCODE_P	Key code constant: 'P' key.
int	KEYCODE_PAGE_DOWN	Key code constant: Page Down key.
int	KEYCODE_PAGE_UP	Key code constant: Page Up key.
int	KEYCODE_PERIOD	Key code constant: '.' key.
int	KEYCODE_PICTSYMBOLS	Key code constant: Picture Symbols modifier key.
int	KEYCODE_PLUS	Key code constant: '+' key.
int	KEYCODE_POUND	Key code constant: '#' key.
int	KEYCODE_POWER	Key code constant: Power key.
int	KEYCODE_PROG_BLUE	Key code constant: Blue "programmable" key.
int	KEYCODE_PROG_GREEN	Key code constant: Green "programmable" key.
int	KEYCODE_PROG_RED	Key code constant: Red "programmable" key.
int	KEYCODE_PROG_YELLOW	Key code constant: Yellow "programmable" key.

t		
int	KEYCODE_Q	Key code constant: 'Q' key.
int	KEYCODE_R	Key code constant: 'R' key.
int	KEYCODE_RIGHT_BRACKET	Key code constant: ']' key.
int	KEYCODE_RO	Key code constant: Japanese Ro key.
int	KEYCODE_S	Key code constant: 'S' key.
int	KEYCODE_SCROLL_LOCK	Key code constant: Scroll Lock key.
int	KEYCODE_SEARCH	Key code constant: Search key.
int	KEYCODE_SEMICOLON	Key code constant: ';' key.
int	KEYCODE_SETTINGS	Key code constant: Settings key.
int	KEYCODE_SHIFT_LEFT	Key code constant: Left Shift modifier key.
int	KEYCODE_SHIFT_RIGHT	Key code constant: Right Shift modifier key.
int	KEYCODE_SLASH	Key code constant: '/' key.
int	KEYCODE_SOFT_LEFT	Key code constant: Soft Left key.
int	KEYCODE_SOFT_RIGHT	Key code constant: Soft Right key.
int	KEYCODE_SPACE	Key code constant: Space key.
int	KEYCODE_STAR	Key code constant: '*' key.

int	KEYCODE_STB_INPUT	Key code constant: Set-top-box input key.
int	KEYCODE_STB_POWER	Key code constant: Set-top-box power key.
int	KEYCODE_SWITCH_CHARSET	Key code constant: Switch Charset modifier key.
int	KEYCODE_SYM	Key code constant: Symbol modifier key.
int	KEYCODE_SYSRQ	Key code constant: System Request / Print Screen key.
int	KEYCODE_T	Key code constant: 'T' key.
int	KEYCODE_TAB	Key code constant: Tab key.
int	KEYCODE_TV	Key code constant: TV key.
int	KEYCODE_TV_INPUT	Key code constant: TV input key.
int	KEYCODE_TV_POWER	Key code constant: TV power key.
int	KEYCODE_U	Key code constant: 'U' key.
int	KEYCODE_UNKNOWN	Key code constant: Unknown key code.
int	KEYCODE_V	Key code constant: 'V' key.
int	KEYCODE_VOLUME_DOWN	Key code constant: Volume Down key.
int	KEYCODE_VOLUME_MUTE	Key code constant: Volume Mute key.
int	KEYCODE_VOLUME_UP	Key code constant: Volume Up key.
in	KEYCODE_W	Key code constant: 'W' key.

t		
int	KEYCODE_WINDOW	Key code constant: Window key.
int	KEYCODE_X	Key code constant: 'X' key.
int	KEYCODE_Y	Key code constant: 'Y' key.
int	KEYCODE_YEN	Key code constant: Japanese Yen key.
int	KEYCODE_Z	Key code constant: 'Z' key.
int	KEYCODE_ZENKAKU_HANKAKU	Key code constant: Japanese full-width / half-width key.
int	KEYCODE_ZOOM_IN	Key code constant: Zoom in key.
int	KEYCODE_ZOOM_OUT	Key code constant: Zoom out key.
int	MAX_KEYCODE	*This constant was deprecated in API level 3. There are now more than MAX_KEYCODE keycodes. Use* `getMaxKeyCode()` *instead.*
int	META_ALT_LEFT_ON	This mask is used to check whether the left ALT meta key is pressed.
int	META_ALT_MASK	This mask is a combination of META_ALT_ON, META_ALT_LEFT_ON and META_ALT_RIGHT_ON.
int	META_ALT_ON	This mask is used to check whether one of the ALT meta keys is pressed.
int	META_ALT_RIGHT_ON	This mask is used to check whether the right the ALT meta key is pressed.
int	META_CAPS_LOCK_ON	This mask is used to check whether the CAPS LOCK meta key is on.
int	META_CTRL_LEFT_ON	This mask is used to check whether the left CTRL meta key is pressed.
int	META_CTRL_MASK	This mask is a combination of META_CTRL_ON, META_CTRL_LEFT_ON and META_CTRL

		_RIGHT_ON.
int	META_CTRL_ON	This mask is used to check whether one of the CTRL meta keys is pressed.
int	META_CTRL_RIGHT_ON	This mask is used to check whether the right CTRL meta key is pressed.
int	META_FUNCTION_ON	This mask is used to check whether the FUNCTION meta key is pressed.
int	META_META_LEFT_ON	This mask is used to check whether the left META meta key is pressed.
int	META_META_MASK	This mask is a combination of META_META_ON, META_META_LEFT_ON and META_META_RIGHT_ON.
int	META_META_ON	This mask is used to check whether one of the META meta keys is pressed.
int	META_META_RIGHT_ON	This mask is used to check whether the right META meta key is pressed.
int	META_NUM_LOCK_ON	This mask is used to check whether the NUM LOCK meta key is on.
int	META_SCROLL_LOCK_ON	This mask is used to check whether the SCROLL LOCK meta key is on.
int	META_SHIFT_LEFT_ON	This mask is used to check whether the left SHIFT meta key is pressed.
int	META_SHIFT_MASK	This mask is a combination of META_SHIFT_ON, META_SHIFT_LEFT_ON and META_SHIFT_RIGHT_ON.
int	META_SHIFT_ON	This mask is used to check whether one of the SHIFT meta keys is pressed.
int	META_SHIFT_RIGHT_ON	This mask is used to check whether the right SHIFT meta key is pressed.
int	META_SYM_ON	This mask is used to check whether the SYM meta key is pressed.

ADB: