

Ui Testing

The Android SDK provides the following tools to support automated, functional UI testing on your application:

- 1. uiautomatorviewer** - A GUI tool to scan and analyze the UI components of an Android application.
- 2. uiautomator** - A Java library containing APIs to create customized functional UI tests, and an execution engine to automate and run the tests.

Minimum Requirements to set Environment

3. JDK 1.6
4. Android SDK Tools, Revision 21 or higher
5. Android SDK Platform, API 16 or higher
6. Ant

Analyzing Your Application's UI

To analyze the UI components of the application

1. Connect your Android device to your development machine.
2. Open a terminal window and navigate to `<android-sdk>/tools/` and Run the tool with this command:
`./uiautomatorviewer`

To Start analyze click on Device Screen shot button in ui automator viewer tool

Reference URL:

http://developer.android.com/tools/testing/testing_ui.html#uianalysis

UI Automator Tool

UI Automator Viewer

Untitled 1.ods Installation steps Untitled Document launch

UI Automator Viewer

Device Screenshot

3G 5:25

ME 6 contacts

Set up my profile

T

Test Contact1087

Test Contact1552

Test Contact1629

Test Contact2600

Node Detail

index	2
text	Test Contact1087
class	android.widget.TextView
package	com.android.contacts
content-desc	
checkable	false
checked	false
clickable	false
enabled	true
focusable	false
focused	false
scrollable	false
long-clickable	false

Configuring development environment

- Create a new Java project in Eclipse, and give your project a name that is relevant to the tests you're about to create (for example, "MyAppNameTests"). In the project, you will create the test cases that are specific to the application that you want to test.
- From the **Project Explorer**, right-click on the new project that you created, then select **Properties > Java Build Path**, and do the following:
 - Click **Add External JARs...** and navigate to the SDK directory. Under the platforms directory, select the latest SDK version and add both the uiautomator.jar and android.jar files.

Development Project Structure

Java - UIAutomationTest/src/com/calculator/test/AddContactTest.java - Eclipse SDK

File Edit Refactor Source Navigate Search Project Run Window Help

Quick Access

Package Explorer

- AllIntentsActivity
- AndroidCalculator
- AndroidCalculatorTest
- DialtactsActivity
- NotePad
- NotePadTest
- SampleAndroid
- UIAutomationTest
 - src
 - com.calculator.test
 - AddContactTest.java
 - DeleteContactTest.java
 - com.calculator.utility
 - UtilityConstants.java
 - JRE System Library [JavaSE-1.6]
 - Referenced Libraries
 - libs
 - android.jar
 - uiautomator.jar
 - build.xml
 - local.properties
 - project.properties

AddContactTest.java

```
package com.calculator.test;

import android.util.Log;

public class AddContactTest extends UiAutomatorTestCase {
    int uniqueNumber = 0;
    String contactName = null;

    public void testAddContact() throws UiObjectNotFoundException {
        uniqueNumber = UtilityConstants.getUniqueVariable();
        launchContactApp();
        //get contact screen
        UiScrollable contactScreen = getNonScrollableUiObject();
        // validate whether contact screen is opened
        validateContactScreen();
        showAddContactScreen(contactScreen);
        insertDetailToAddCntScreen(contactScreen);
        //show current window screen ie- contact summary screen
        UiScrollable contactDtlScreen = getNonScrollableUiObject();
        verifyCntSummary(contactDtlScreen);
        //User is in contact screen. click BACK KEY to return to All Apps Launcher
        getUiDevice().pressBack();
    }

    /**
     * Description: launches contact app
     * @throws UiObjectNotFoundException
     */
    private void launchContactApp() throws UiObjectNotFoundException{
        // Go to Home screen by pressing on the HOME key.
        getUiDevice().pressHome();
        //Next, check for All Apps button's content-description property has
        UiObject allAppsButton = new UiObject(new UiSelector()
            .description("Apps"));
        // Simulate a click to bring up the All Apps screen.
    }
}
```

com.calculator.test

- uniqueNu
- contactNi
- testAddC
- launchCor
- getNonSc
- showAdd
- insertDet
- verifyCnt
- validateC
- verifyAdd

Problems Javadoc Declaration Search Console LogCat File Explorer Devices

Name	Size	Date	Time	Permissions	Info
------	------	------	------	-------------	------

Developing

- For further reference please have a look on following link

http://developer.android.com/tools/testing/testing_ui.html#creating

Ui Elements

Classes

Class	Description
<code>com.android.uiautomator.core.UiCollection</code>	Used to enumerate a container's user interface (UI) elements for the purpose of counting, or targeting a sub elements by a child's text or description.
<code>com.android.uiautomator.core.UiDevice</code>	Provides access to state information about the device. You can also use this class to simulate user actions on the device, such as pressing the d-pad hardware button or pressing the Home and Menu buttons.
<code>com.android.uiautomator.core.UiObject</code>	Represents a user interface (UI) element.
<code>com.android.uiautomator.core.UiScrollable</code>	Provides support for searching for items in a scrollable UI container.
<code>com.android.uiautomator.core.UiSelector</code>	Represents a query for one or more target UI elements on a device screen.

Interfaces

Interface	Description
<code>com.android.uiautomator.core.UiWatcher</code>	Represents a conditional watcher on the target device.
<code>com.android.uiautomator.testrunner.IAutomationSupport</code>	Provides auxiliary support for running test cases.
<code>com.android.uiautomator.testrunner.UiAutomatorTestCase</code>	Defines an environment for running multiple tests. All <code>uiautomator</code> test cases should extend this class.

Exceptions

Exception	Description
<code>com.android.uiautomator.core.UiObjectNotFoundException</code>	Indicates when a <code>UiSelector</code> could not be matched to any UI element displayed.

Deploying And Running uiautomator Tests

1. First check list of android devices that you have. Use following syeps

Open your terminal. Navigate to tools

Example: Run following command

```
cd /home/sunil/Development/android-sdk-linux/tools/
```

Run following command in your terminal to get list of devices and its id.

```
./android list targets
```

2. Create the required build configuration files i.e. (build.xml) .

Run the following command: in terminal

```
<android-sdk>/tools/android create uitest-project -n <name> -t 1 -p <path>
```

<name> is the name of the project that contains your uiautomator test source files, and the <path> is the path to the corresponding project directory.

Example: `./android create uitest-project -n UIAutomationTest -t 6 -p /home/sunil/workspace/UIAutomationTest`

O/p: build.xml will be generated if not generated before, otherwise build.xml file will be updated

3. From the command line, set the ANDROID_HOME variable:

In Windows: `set ANDROID_HOME=<path_to_your_sdk>`

In UNIX: `export ANDROID_HOME=<path_to_your_sdk>`

4. Go to the project directory where your build.xml file is located and build your test JAR.ant build

Example:

```
cd /home/sunil/workspace/UIAutomationTest
```

And run following command in terminal to generate jar file

```
ant build
```

5. Deploy your generated test JAR file to the test device by using following command

```
./adb push command:adb push <path_to_output_jar> /data/local/tmp/
```

Example:

```
cd /home/sunil/Development/android-sdk-linux/platform-tools/
```

push the jar file

```
./adb push /home/sunil/workspace/UIAutomationTest/bin/UIAutomationTest.jar /data/local/tmp/
```

Running uiautomator Tests

Example:

```
./adb shell uiautomator runtest UIAutomationTest.jar -c com.calculator.test.AddContactTest
```

4. Go to the project directory where your build.xml file is located and build your test JAR.ant build

Example:

```
cd /home/sunil/workspace/UIAutomationTest
```

And run following command in terminal to generate jar file

```
ant build
```

5. Deploy your generated test JAR file to the test device by using following command

```
./adb push command:adb push <path_to_output_jar> /data/local/tmp/
```

Example:

```
cd /home/sunil/Development/android-sdk-linux/platform-tools/
```

push the jar file

```
./adb push /home/sunil/workspace/UIAutomationTest/bin/UIAutomationTest.jar /data/local/tmp/
```

Running uiautomator Tests

Example:

```
./adb shell uiautomator runtest UIAutomationTest.jar -c com.calculator.test.AddContactTest
```

Assertion Failure Message

```
sunil@sunil-desktop: ~/Development/android-sdk-linux/platform-tools
INSTRUMENTATION_STATUS: class=com.calculator.test.LaunchSettings
INSTRUMENTATION_STATUS: stream=
com.calculator.test.LaunchSettings:
INSTRUMENTATION_STATUS: numtests=1
INSTRUMENTATION_STATUS: test=testDemo
INSTRUMENTATION_STATUS_CODE: 1
INSTRUMENTATION_STATUS: current=1
INSTRUMENTATION_STATUS: id=UiAutomatorTestRunner
INSTRUMENTATION_STATUS: class=com.calculator.test.LaunchSettings
INSTRUMENTATION_STATUS: stream=
Failure in testDemo:
junit.framework.AssertionFailedError: Unable to detect Settings
    at com.calculator.test.LaunchSettings.validateContactScreen(LaunchSettin
gs.java:151)
    at com.calculator.test.LaunchSettings.testDemo(LaunchSettings.java:21)
    at java.lang.reflect.Method.invokeNative(Native Method)
    at com.android.uiautomator.testrunner.UiAutomatorTestRunner.start(UiAuto
matorTestRunner.java:144)
    at com.android.uiautomator.testrunner.UiAutomatorTestRunner.run(UiAutoma
torTestRunner.java:87)
    at com.android.commands.uiautomator.RunTestCommand.run(RunTestCommand.ja
va:90)
    at com.android.commands.uiautomator.Launcher.main(Launcher.java:83)
    at com.android.internal.os.RuntimeInit.nativeFinishInit(Native Method)
    at com.android.internal.os.RuntimeInit.main(RuntimeInit.java:235)
    at dalvik.system.NativeStart.main(Native Method)
INSTRUMENTATION_STATUS: numtests=1
INSTRUMENTATION_STATUS: stack=junit.framework.AssertionFailedError: Unable to de
tect Settings
    at com.calculator.test.LaunchSettings.validateContactScreen(LaunchSettin
gs.java:151)
    at com.calculator.test.LaunchSettings.testDemo(LaunchSettings.java:21)
    at java.lang.reflect.Method.invokeNative(Native Method)
    at com.android.uiautomator.testrunner.UiAutomatorTestRunner.start(UiAuto
matorTestRunner.java:144)
    at com.android.uiautomator.testrunner.UiAutomatorTestRunner.run(UiAutoma
torTestRunner.java:87)
    at com.android.commands.uiautomator.RunTestCommand.run(RunTestCommand.ja
va:90)
    at com.android.commands.uiautomator.Launcher.main(Launcher.java:83)
    at com.android.internal.os.RuntimeInit.nativeFinishInit(Native Method)
    at com.android.internal.os.RuntimeInit.main(RuntimeInit.java:235)
    at dalvik.system.NativeStart.main(Native Method)
INSTRUMENTATION_STATUS: test=testDemo
INSTRUMENTATION_STATUS_CODE: -2
INSTRUMENTATION_STATUS: stream=
Test results for WatcherResultPrinter=.F
Time: 10.39

FAILURES!!!
Tests run: 1, Failures: 1, Errors: 0
```

Thank You