JUnit Framework

Introduction

- The JUnit framework is an open source project for unit testing.
- The Eclipse IDE supports the usage of JUnit when performing unit testing.

Unit Testing

- We can develop unit testing code for our android code similarly to unit testing for non-android Java software development.
- The eclipse already includes a wizard we can use to create unit tests for our android application. Using this wizard we can get unit tests that utilize the instrumentation framework.

Unit Testing Sample

```
package com.abelski.samples;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class SimpleApplicationActivity extends Activity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        int num = total(4,7);
        TextView textView = (TextView)findViewById(R.id.totaltext);
        textView.setText(String.valueOf(num));
    }
    public int total(int numA, int numB)
    {
        int total = 0;
        if(numB>=numA)
            for(int i=numA; i<=numB; i++)
                total += i;
        return total;
    }
}
```

this is the activity we test it is part of the android project we test

Unit Testing Sample

```
package com.abelski.samples.test;

import android.app.Activity;
import android.test.ActivityInstrumentationTestCase2;
import android.util.Log;
import android.widget.TextView;
import com.abelski.samples.*;

public class SimpleApplicationActivityTest extends
        ActivityInstrumentationTestCase2<SimpleApplicationActivity>
{
    Activity activityWeTest = null;
    TextView textView = null;
    public SimpleApplicationActivityTest()
    {
        super("com.abelski.samples",SimpleApplicationActivity.class);
        Log.i("tester","within SimpleApplicationActivityTest() constructor");
    }
```

this is the a separated android test project we should create in order to include the testing code

Unit Testing Sample

```
    protected void setUp() throws Exception
    {
        //the setUp method is been called before each every test method
        super.setUp();
        Log.i("tester","within setUp()");
        activityWeTest = this.getActivity();
        textView = (TextView)activityWeTest.findViewById(
            com.abelski.samples.R.id.totaltext);
    }
    public void testTotalMethod()
    {
        assertEquals(9,((SimpleApplicationActivity)activityWeTest).total(2,4));

    }
    public void testTextView()
    {
        assertEquals("22",textView.getText());
    }
}
```

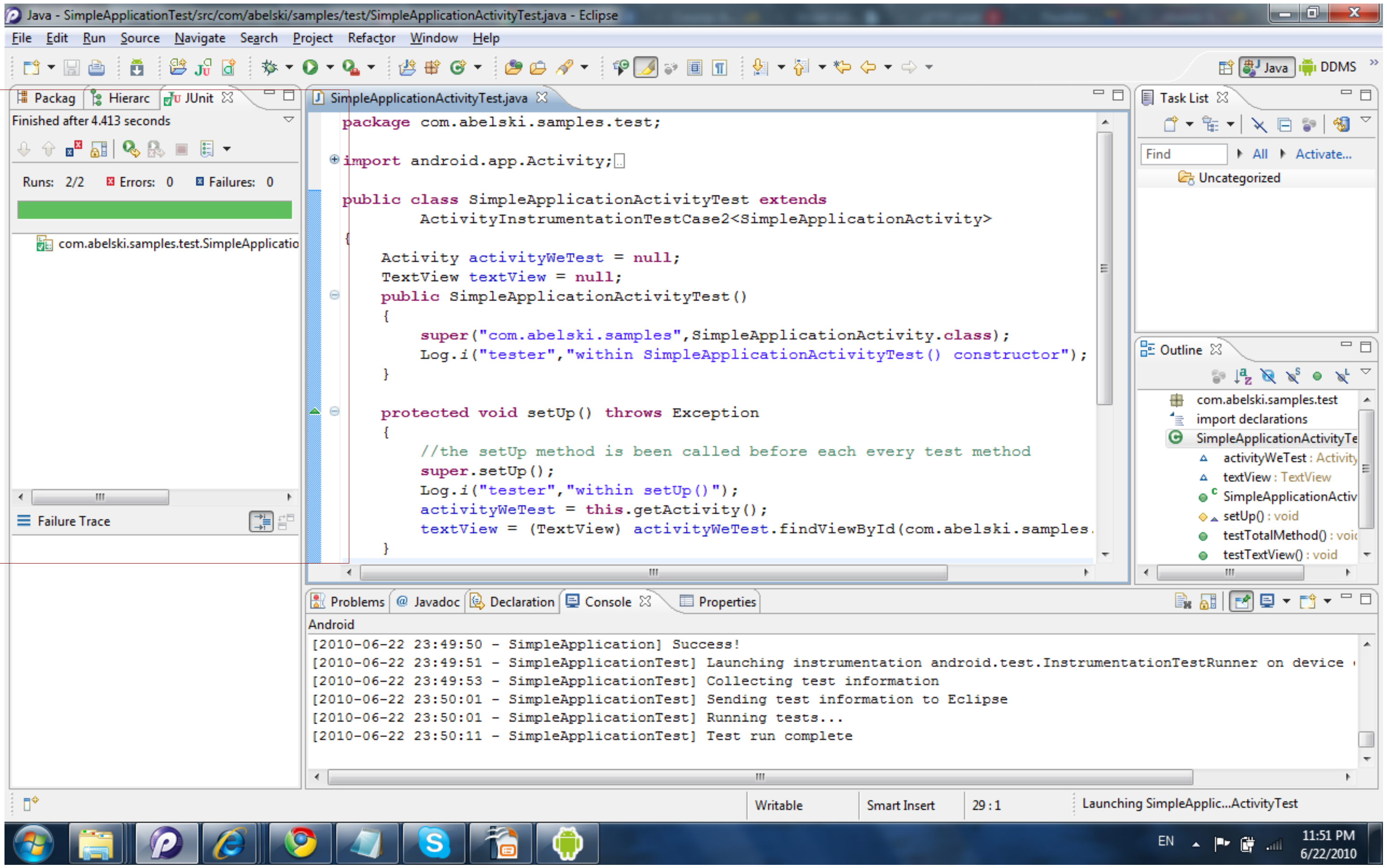
Unit Testing Sample

```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.abelski.samples.test"
      android:versionCode="1"
      android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <uses-library android:name="android.test.runner" />
    </application>

    <uses-sdk android:minSdkVersion="8" />

    <instrumentation android:targetPackage="com.abelski.samples"
        android:name="android.test.InstrumentationTestRunner" />

</manifest>
```

The instrumentation XML element is added automatically. It links this project with the project we test.

Unit Testing Sample

the results of the unit testing


Java - SimpleApplicationTest/src/com/abelski/samples/test/SimpleApplicationActivityTest.java - Eclipse
File Edit Run Source Navigate Search Project Refactor Window Help
Packag Hierarc JUnit
Finished after 4.413 seconds
Runs: 2/2 Errors: 0 Failures: 0
com.abelski.samples.test.SimpleApplicatio
Failure Trace
SimpleApplicationActivityTest.java
Task List
Find All Activate...
Uncategorized
Outline
com.abelski.samples.test
import declarations
SimpleApplicationActivityTe
activityWeTest : Activity
textView : TextView
SimpleApplicationActiv
setUp() : void
testTotalMethod() : void
testTextView() : void
Problems Javadoc Declaration Console Properties
Android
Writable Smart Insert 29 : 1 Launching SimpleApplic...ActivityTest
EN 11:51 PM 6/22/2010

```java
package com.abelski.samples.test;

import android.app.Activity;

public class SimpleApplicationActivityTest extends
        ActivityInstrumentationTestCase2<SimpleApplicationActivity>
{
    Activity activityWeTest = null;
    TextView textView = null;
    public SimpleApplicationActivityTest()
    {
        super("com.abelski.samples",SimpleApplicationActivity.class);
        Log.i("tester","within SimpleApplicationActivityTest() constructor");
    }

    protected void setUp() throws Exception
    {
        //the setUp method is been called before each every test method
        super.setUp();
        Log.i("tester","within setUp()");
        activityWeTest = this.getActivity();
        textView = (TextView) activityWeTest.findViewById(com.abelski.samples.
    }
```

```
[2010-06-22 23:49:50 - SimpleApplication] Success!
[2010-06-22 23:49:51 - SimpleApplicationTest] Launching instrumentation android.test.InstrumentationTestRunner on device
[2010-06-22 23:49:53 - SimpleApplicationTest] Collecting test information
[2010-06-22 23:50:01 - SimpleApplicationTest] Sending test information to Eclipse
[2010-06-22 23:50:01 - SimpleApplicationTest] Running tests...
[2010-06-22 23:50:11 - SimpleApplicationTest] Test run complete
```

The `<instrumentation>` Element

- The instrumentation framework runs boths the main application and the test application in the same process.
- The linkage between the two applications is implemented using the `<instrumentation>` element we should find within the manifest file of the android test project.

The `<instrumentation>` Element

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.abelski.samples.test"
      android:versionCode="1"
      android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <uses-library android:name="android.test.runner" />
    </application>

    <uses-sdk android:minSdkVersion="8" />

    <instrumentation android:targetPackage="com.abelski.samples"
        android:name="android.test.InstrumentationTestRunner" />

</manifest>
```

this is the manifest file of the Android Test Project developed in previous topic

The `InstrumentationTestRunner` Class

- Object instantiated from the `InstrumentationTestRunner` class is responsible for running both the thread that executes the testing activity and the thread that executes the application we test.

The `InstrumentationTestCase` Class

- The `InstrumentationTestCase` class is the base class for various sub classes that have the ability to send a keystroke to touch event to the user interface of the application we test.
- The subclasses include the following:

```
ActivityTestCase
SingleLaunchActivityTestCase
SyncBaseInstrumentation
ActivityUnitTestCase
ActivityInstrumentationTestCase2
```

The `InstrumentationTestCase` Class

- We can define a new class that extends the `InstrumentationTestCase` class.
- One of the methods defined within the `InstrumentationTestCase` class is the `getInstrumentation()` method. Calling this method we get a reference for the `Instrumentation` object.

The `Instrumentation` Class

- The `Instrumentation` class has helper methods that enable us to send key events and strings to the application we test (e.g. `sendStringSync` sends a string to an input box, `sendKeyDownUpSync` sends a specific key event).

Instrumentation Sample

this is the application we want to test

```
package com.abelski.samples;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class SimpleApplicationActivity extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button btPlus = (Button)findViewById(R.id.Button01);
        Button btMinus = (Button)findViewById(R.id.Button02);
        final EditText text1 = (EditText)findViewById(R.id.EditText01);
        final EditText text2 = (EditText)findViewById(R.id.EditText02);
        final EditText text3 = (EditText)findViewById(R.id.EditText03);
```

Instrumentation Sample

```
        btPlus.setOnClickListener(new OnClickListener()
        {
            @Override
            public void onClick(View arg0)
            {
                double result =
                    Double.parseDouble(text1.getText().toString()) +
                    Double.parseDouble(text2.getText().toString());
                text3.setText(""+result);
            }
        });
        btMinus.setOnClickListener(new OnClickListener()
        {
            @Override
            public void onClick(View arg0)
            {
                double result =
                    Double.parseDouble(text1.getText().toString()) -
                    Double.parseDouble(text2.getText().toString());
                text3.setText(""+result);
            }
        });
    }
}
```

Instrumentation Sample

```
<?xml version="1.0" encoding="utf-8"?>

    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">

        <EditText   android:id="@+id/EditText01"
                    android:layout_height="wrap_content"
                    android:layout_width="fill_parent"
                    android:text="4">
        </EditText>

        <EditText   android:id="@+id/EditText02"
                    android:layout_height="wrap_content"
                    android:layout_width="fill_parent"
                    android:text="3">
        </EditText>

        <Button     android:id="@+id/Button02"
                    android:layout_height="wrap_content"
                    android:text="+"
                    android:layout_width="fill_parent">
        </Button>
```

Instrumentation Sample

```
        <Button     android:id="@+id/Button01"
                    android:layout_height="wrap_content"
                    android:text="-"
                    android:layout_width="fill_parent">
        </Button>

        <EditText   android:id="@+id/EditText03"
                    android:layout_height="wrap_content"
                    android:layout_width="fill_parent">
        </EditText>

</LinearLayout>
```

Instrumentation Sample

this is the application that performs the tests

```
package com.abelski.samples.test;

import android.app.Activity;
import android.app.Instrumentation;
import android.test.ActivityInstrumentationTestCase2;
import android.util.Log;
import android.view.KeyEvent;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import com.abelski.samples.*;

public class SimpleApplicationActivityTest extends
    ActivityInstrumentationTestCase2<SimpleApplicationActivity>
{
    Activity activityWeTest = null;
    EditText text1,text2,text3;
    Button btPlus, btMinus;
    Instrumentation instrumentation;
```

Instrumentation Sample

```
public SimpleApplicationActivityTest()
{
    super("com.abelski.samples", SimpleApplicationActivity.class);
}

protected void setUp() throws Exception
{
    // the setUp method is been called before each every test method
    super.setUp();
    Log.i("tester", "within setUp()");
    activityWeTest = this.getActivity();
    instrumentation = this.getInstrumentation();
    text1 = (EditText)
        activityWeTest.findViewById(com.abelski.samples.R.id.EditText01);
    text2 = (EditText)
        activityWeTest.findViewById(com.abelski.samples.R.id.EditText02);
    text3 = (EditText)
        activityWeTest.findViewById(com.abelski.samples.R.id.EditText03);
    btPlus = (Button)
        activityWeTest.findViewById(com.abelski.samples.R.id.Button01);
    btMinus = (Button)
        activityWeTest.findViewById(com.abelski.samples.R.id.Button02);
}
```

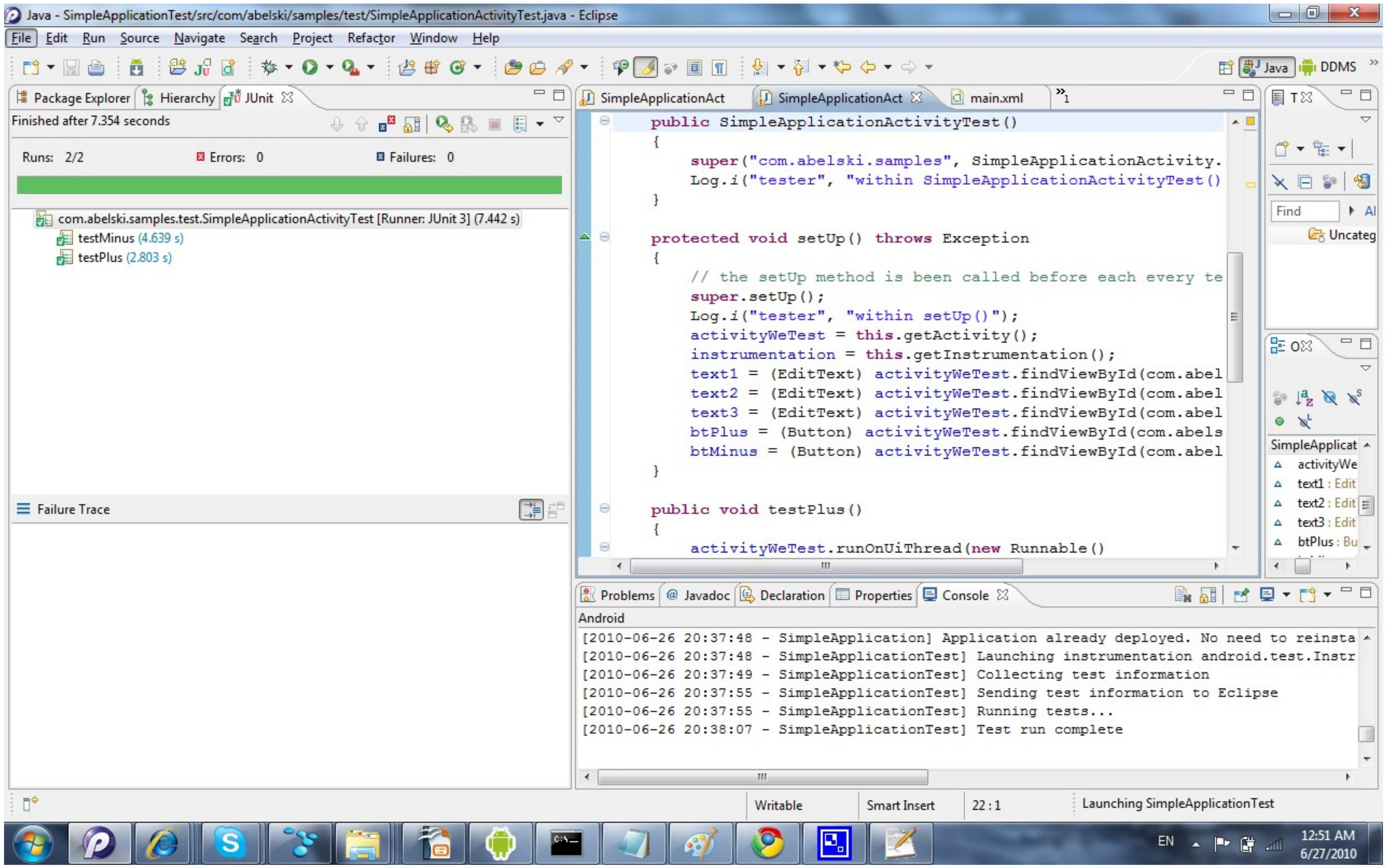
Instrumentation Sample

```
public void testPlus()
{
    activityWeTest.runOnUiThread(new Runnable()
    {
        public void run()
        {
            btPlus.requestFocus();
        }
    });
    instrumentation.waitForIdleSync();
    sendKeys(KeyEvent.KEYCODE_DPAD_CENTER);
    assertEquals(7.0, Double.parseDouble(text3.getText().toString()));
}
```

Instrumentation Sample

```
    public void testMinus()
    {
        activityWeTest.runOnUiThread(new Runnable()
        {
            public void run()
            {
                btMinus.requestFocus();
            }
        });
        instrumentation.waitForIdleSync();
        sendKeys(KeyEvent.KEYCODE_DPAD_CENTER);
        assertEquals(1.0, Double.parseDouble(text3.getText().toString()));
    }

}
```

Instrumentation Sample



Java - SimpleApplicationTest/src/com/abelski/samples/test/SimpleApplicationActivityTest.java - Eclipse

Finished after 7.354 seconds

Runs: 2/2 Errors: 0 Failures: 0

com.abelski.samples.test.SimpleApplicationActivityTest [Runner: JUnit 3] (7.442 s)
- testMinus (4.639 s)
- testPlus (2.803 s)

```java
    public SimpleApplicationActivityTest()
    {
        super("com.abelski.samples", SimpleApplicationActivity.
        Log.i("tester", "within SimpleApplicationActivityTest()
    }

    protected void setUp() throws Exception
    {
        // the setUp method is been called before each every te
        super.setUp();
        Log.i("tester", "within setUp()");
        activityWeTest = this.getActivity();
        instrumentation = this.getInstrumentation();
        text1 = (EditText) activityWeTest.findViewById(com.abel
        text2 = (EditText) activityWeTest.findViewById(com.abel
        text3 = (EditText) activityWeTest.findViewById(com.abel
        btPlus = (Button) activityWeTest.findViewById(com.abels
        btMinus = (Button) activityWeTest.findViewById(com.abel
    }

    public void testPlus()
    {
        activityWeTest.runOnUiThread(new Runnable()
```

Android

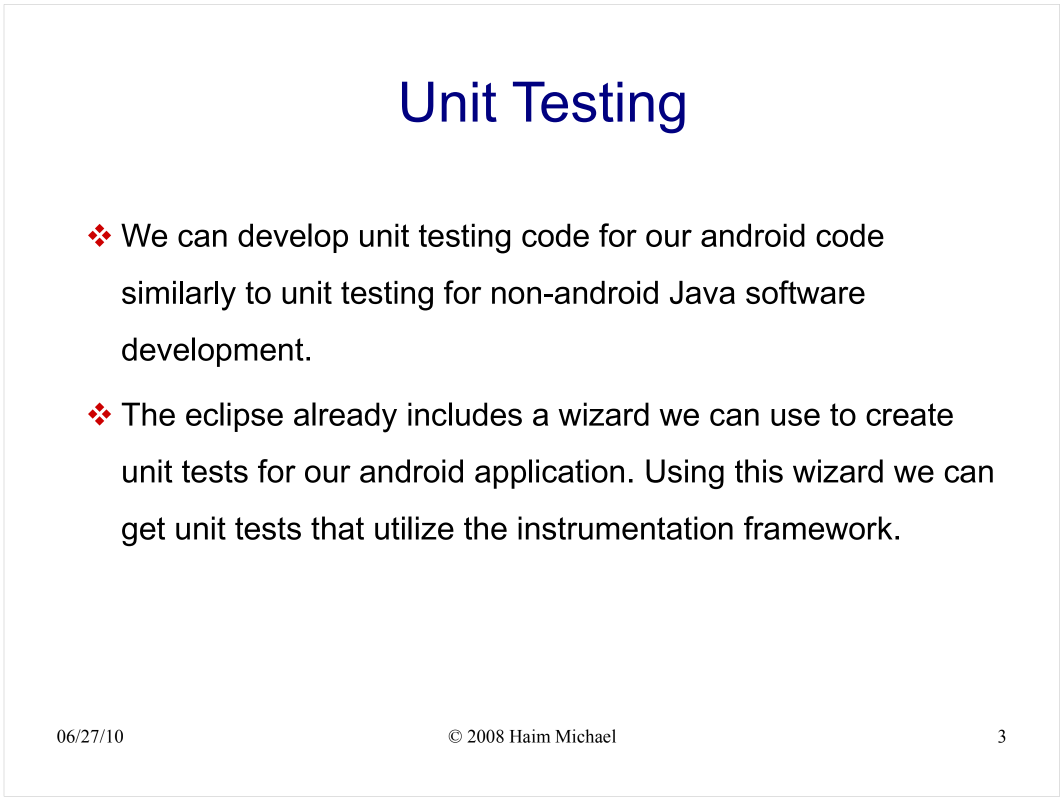
```
[2010-06-26 20:37:48 - SimpleApplication] Application already deployed. No need to reinsta
[2010-06-26 20:37:48 - SimpleApplicationTest] Launching instrumentation android.test.Instr
[2010-06-26 20:37:49 - SimpleApplicationTest] Collecting test information
[2010-06-26 20:37:55 - SimpleApplicationTest] Sending test information to Eclipse
[2010-06-26 20:37:55 - SimpleApplicationTest] Running tests...
[2010-06-26 20:38:07 - SimpleApplicationTest] Test run complete
```

Writable Smart Insert 22 : 1 Launching SimpleApplicationTest

JUnit Framework

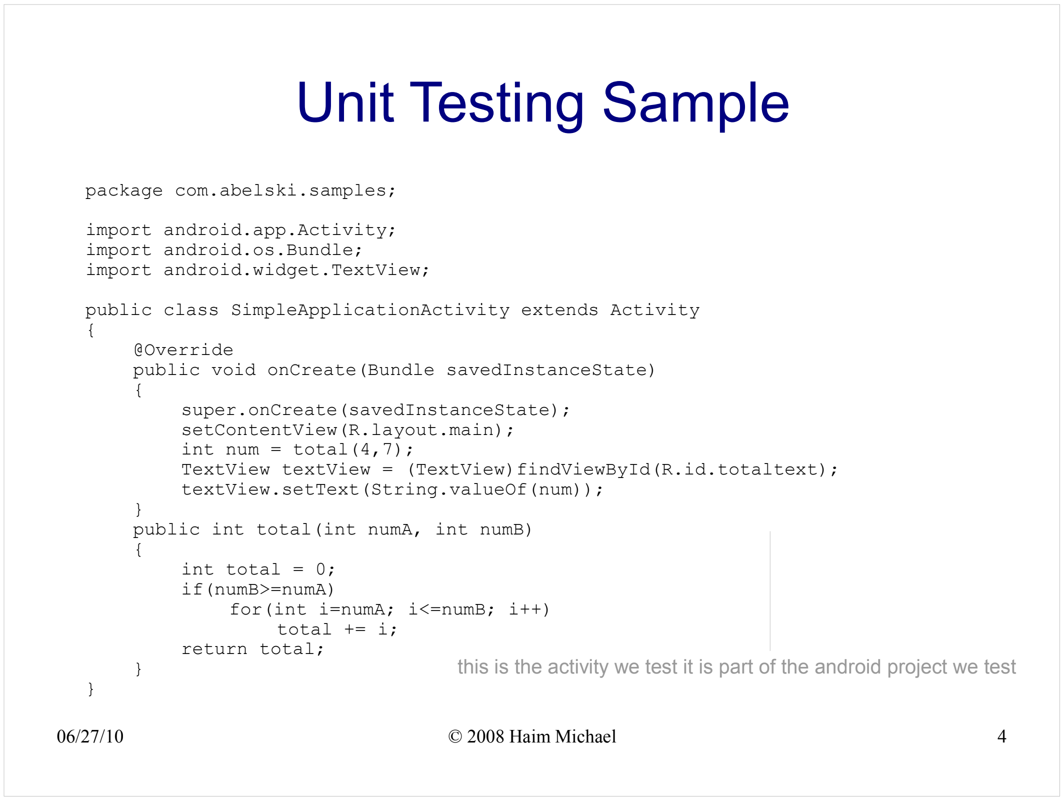
Introduction

- The JUnit framework is an open source project for unit testing.
- The Eclipse IDE supports the usage of JUnit when performing unit testing.



Unit Testing

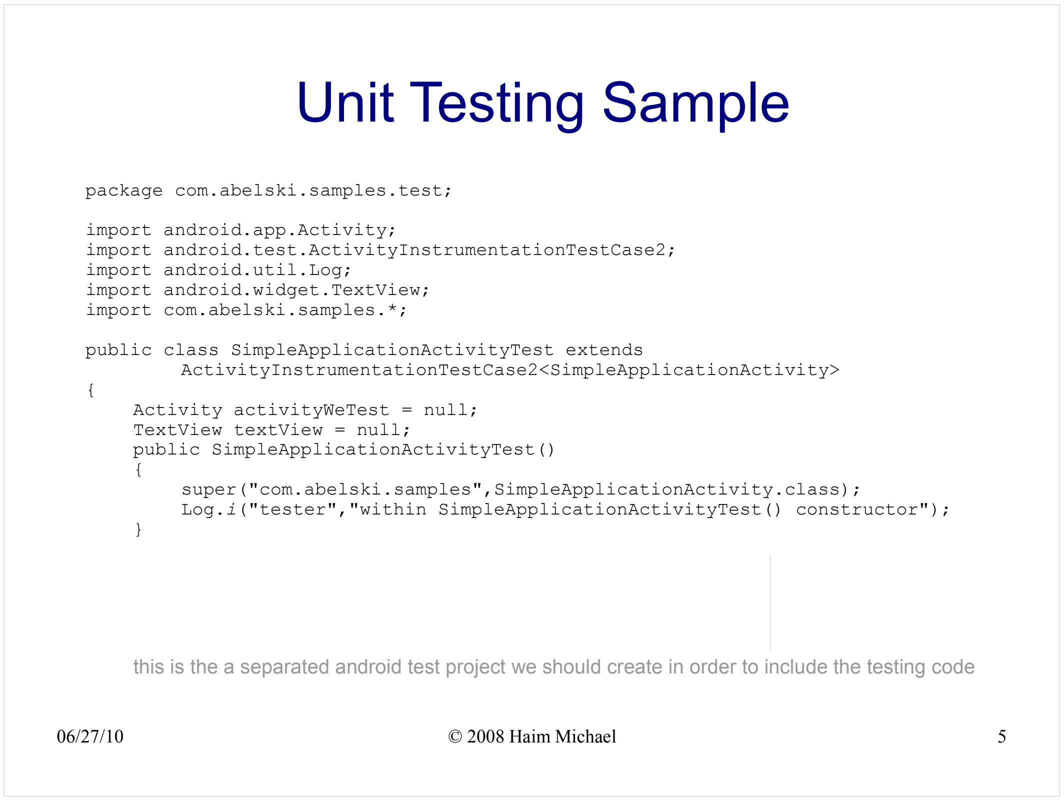
- We can develop unit testing code for our android code similarly to unit testing for non-android Java software development.
- The eclipse already includes a wizard we can use to create unit tests for our android application. Using this wizard we can get unit tests that utilize the instrumentation framework.



Unit Testing Sample

```
package com.abelski.samples;

import android.app.Activity;
import android.os.Bundle;
import android.widget.TextView;

public class SimpleApplicationActivity extends Activity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        int num = total(4,7);
        TextView textView = (TextView)findViewById(R.id.totaltext);
        textView.setText(String.valueOf(num));
    }
    public int total(int numA, int numB)
    {
        int total = 0;
        if(numB>=numA)
            for(int i=numA; i<=numB; i++)
                total += i;
        return total;
    }
}
```

this is the activity we test it is part of the android project we test



Unit Testing Sample

```
package com.abelski.samples.test;

import android.app.Activity;
import android.test.ActivityInstrumentationTestCase2;
import android.util.Log;
import android.widget.TextView;
import com.abelski.samples.*;

public class SimpleApplicationActivityTest extends
        ActivityInstrumentationTestCase2<SimpleApplicationActivity>
{
    Activity activityWeTest = null;
    TextView textView = null;
    public SimpleApplicationActivityTest()
    {
        super("com.abelski.samples",SimpleApplicationActivity.class);
        Log.i("tester","within SimpleApplicationActivityTest() constructor");
    }
```

this is the a separated android test project we should create in order to include the testing code

Unit Testing Sample

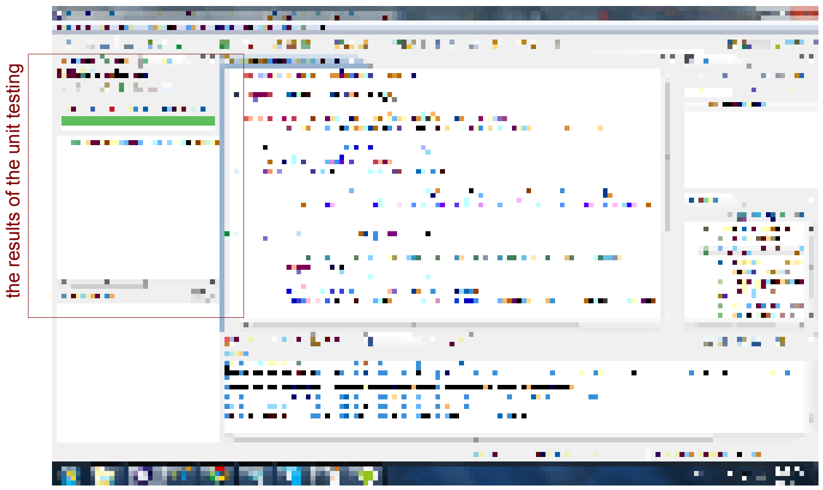
```
        protected void setUp() throws Exception
        {
            //the setUp method is been called before each every test method
            super.setUp();
            Log.i("tester","within setUp()");
            activityWeTest = this.getActivity();
            textView = (TextView)activityWeTest.findViewById(
                com.abelski.samples.R.id.totaltext);
        }
        public void testTotalMethod()
        {
            assertEquals(9,((SimpleApplicationActivity)activityWeTest).total(2,4));

        }
        public void testTextView()
        {
            assertEquals("22",textView.getText());
        }
    }
```

Unit Testing Sample

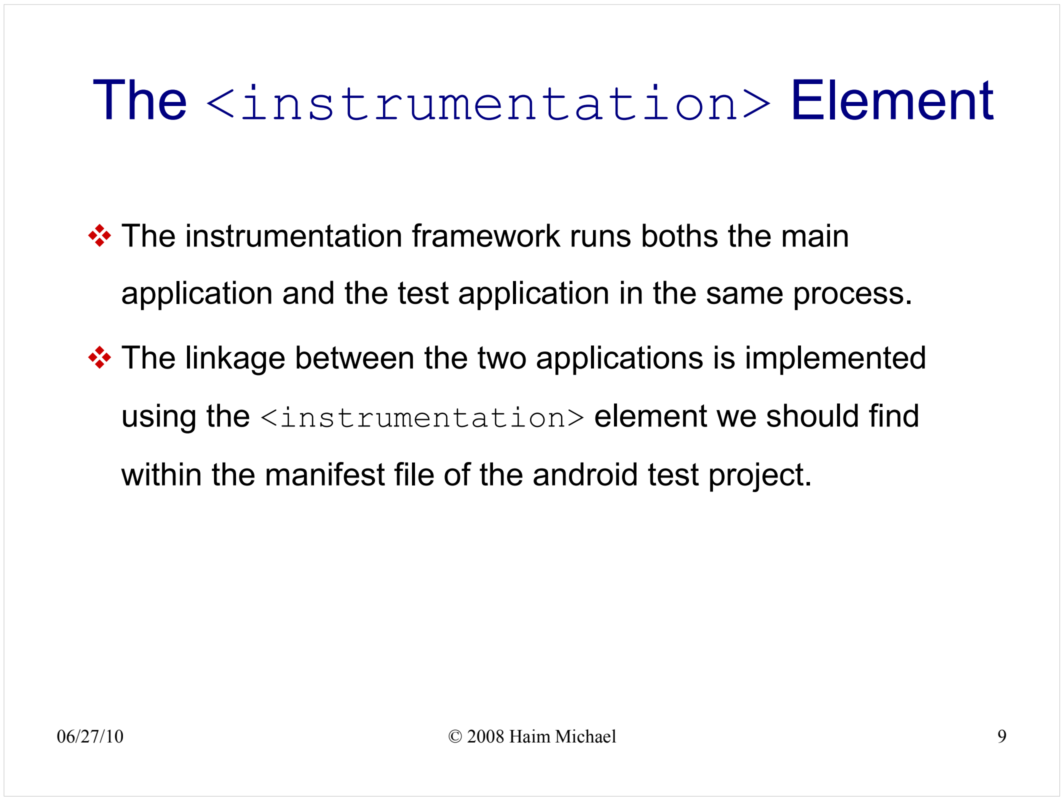
```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.abelski.samples.test"
      android:versionCode="1"
      android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <uses-library android:name="android.test.runner" />
    </application>

    <uses-sdk android:minSdkVersion="8" />

    <instrumentation android:targetPackage="com.abelski.samples"
        android:name="android.test.InstrumentationTestRunner" />

</manifest>
```

The instrumentation XML element is added automatically. It links this project with the project we test.

Unit Testing Sample



06/27/10 © 2008 Haim Michael 8



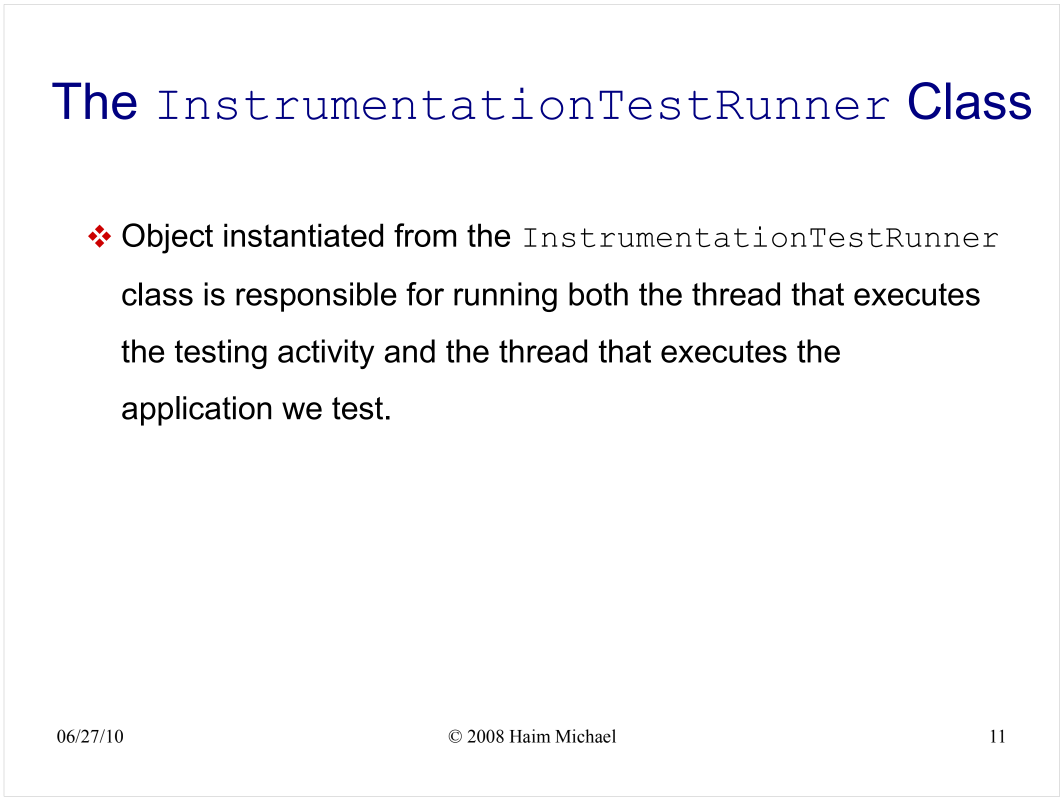
The `<instrumentation>` Element

- The instrumentation framework runs boths the main application and the test application in the same process.
- The linkage between the two applications is implemented using the `<instrumentation>` element we should find within the manifest file of the android test project.

The `<instrumentation>` Element

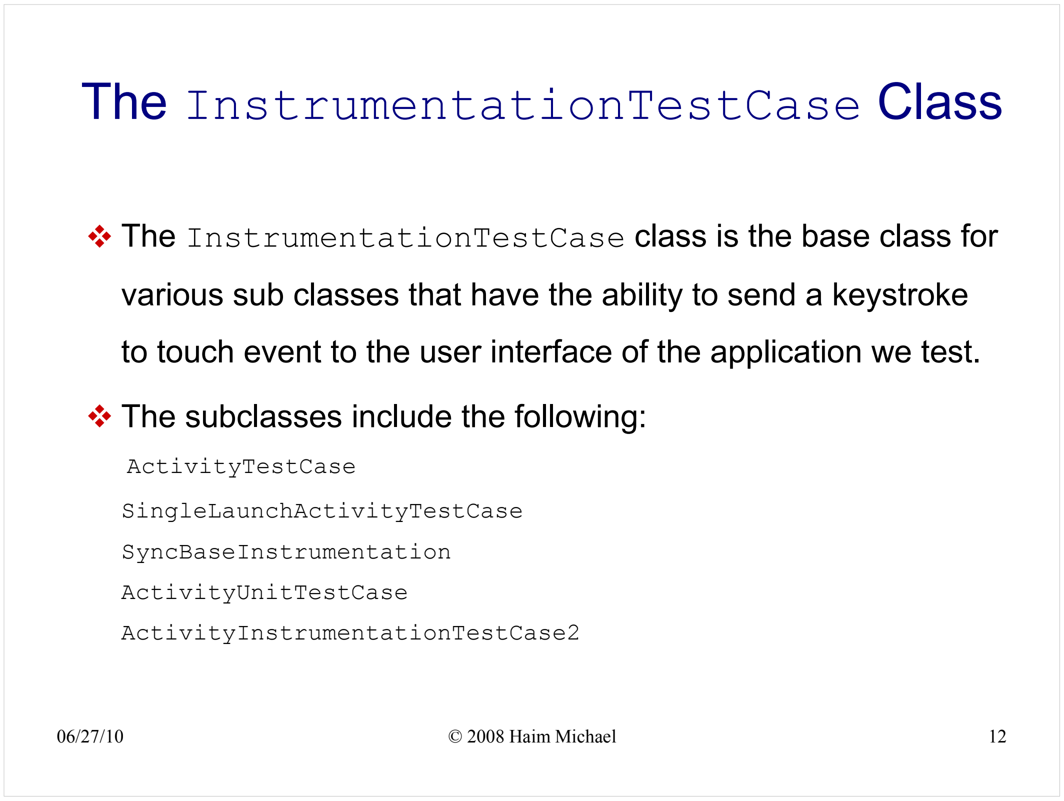
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.abelski.samples.test"
      android:versionCode="1"
      android:versionName="1.0">

    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <uses-library android:name="android.test.runner" />
    </application>

    <uses-sdk android:minSdkVersion="8" />

    <instrumentation android:targetPackage="com.abelski.samples"
        android:name="android.test.InstrumentationTestRunner" />

</manifest>
```

this is the manifest file of the Android Test Project developed in previous topic



The `InstrumentationTestRunner` Class

- Object instantiated from the `InstrumentationTestRunner` class is responsible for running both the thread that executes the testing activity and the thread that executes the application we test.

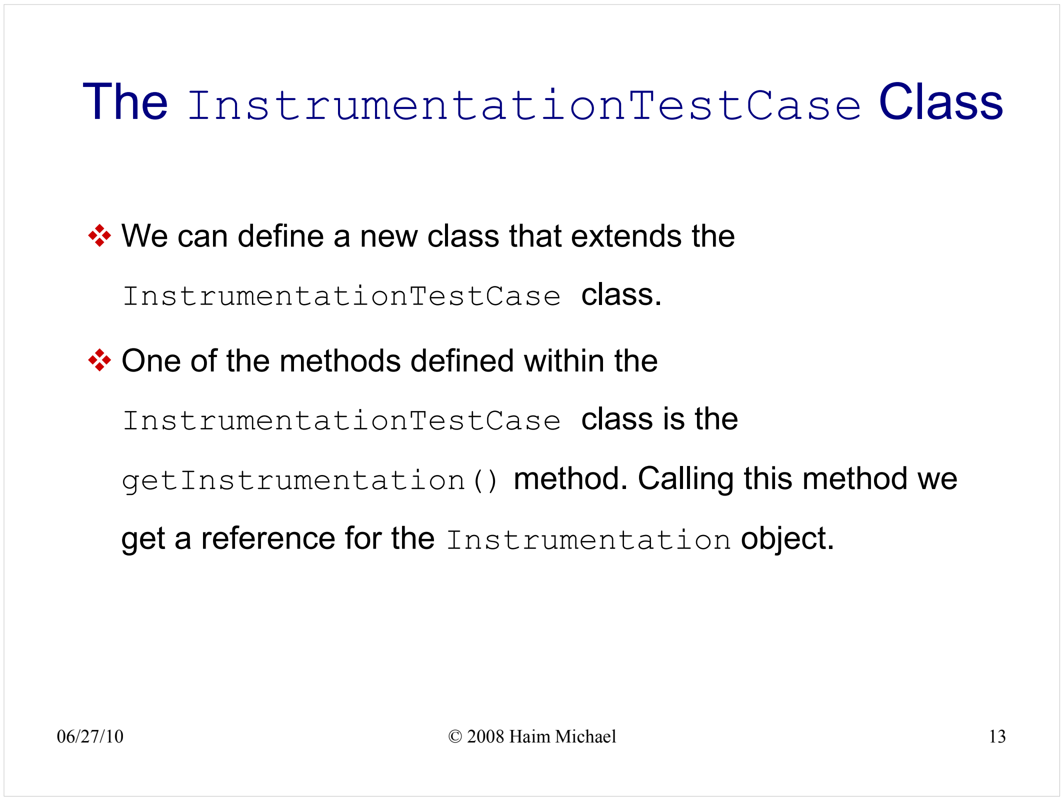


The `InstrumentationTestCase` Class

- The `InstrumentationTestCase` class is the base class for various sub classes that have the ability to send a keystroke to touch event to the user interface of the application we test.
- The subclasses include the following:

```
ActivityTestCase
SingleLaunchActivityTestCase
SyncBaseInstrumentation
ActivityUnitTestCase
ActivityInstrumentationTestCase2
```

06/27/10 © 2008 Haim Michael 12



The `InstrumentationTestCase` Class

- We can define a new class that extends the `InstrumentationTestCase` class.
- One of the methods defined within the `InstrumentationTestCase` class is the `getInstrumentation()` method. Calling this method we get a reference for the `Instrumentation` object.

The `Instrumentation` Class

- The `Instrumentation` class has helper methods that enable us to send key events and strings to the application we test (e.g. `sendStringSync` sends a string to an input box, `sendKeyDownUpSync` sends a specific key event).

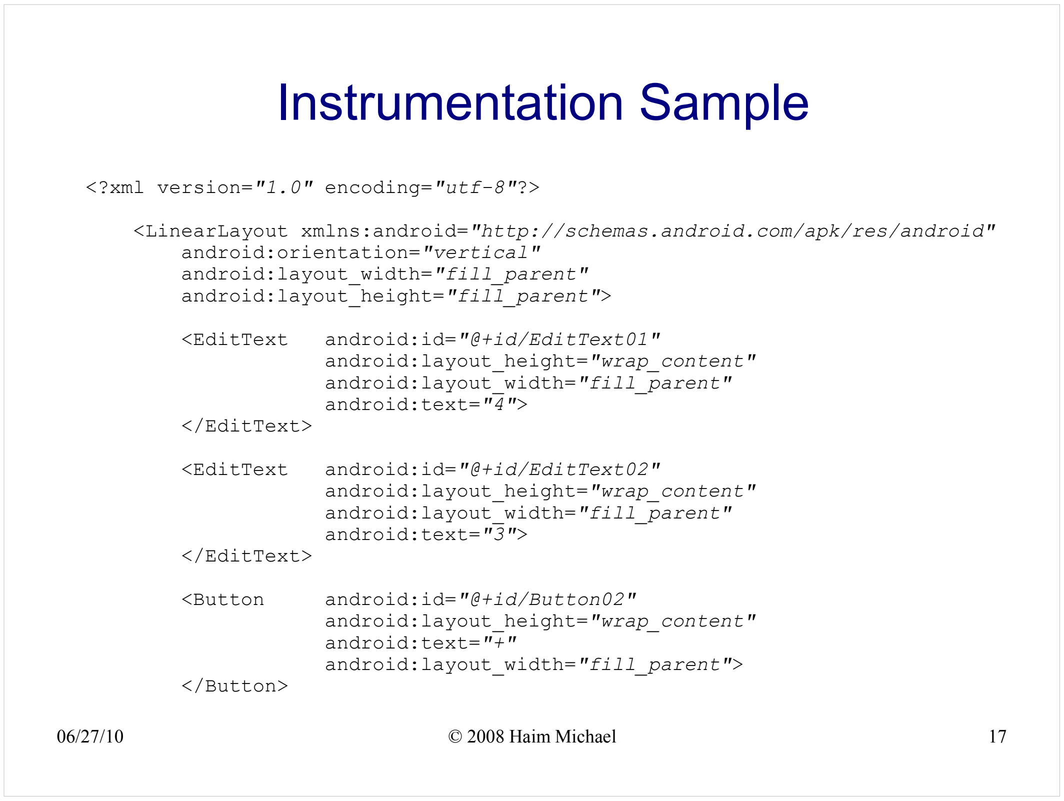
Instrumentation Sample

this is the application we want to test

```
package com.abelski.samples;

import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class SimpleApplicationActivity extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        Button btPlus = (Button)findViewById(R.id.Button01);
        Button btMinus = (Button)findViewById(R.id.Button02);
        final EditText text1 = (EditText)findViewById(R.id.EditText01);
        final EditText text2 = (EditText)findViewById(R.id.EditText02);
        final EditText text3 = (EditText)findViewById(R.id.EditText03);
```

Instrumentation Sample

```
        btPlus.setOnClickListener(new OnClickListener()
        {
            @Override
            public void onClick(View arg0)
            {
                double result =
                    Double.parseDouble(text1.getText().toString()) +
                    Double.parseDouble(text2.getText().toString());
                text3.setText(""+result);
            }
        });
        btMinus.setOnClickListener(new OnClickListener()
        {
            @Override
            public void onClick(View arg0)
            {
                double result =
                    Double.parseDouble(text1.getText().toString()) -
                    Double.parseDouble(text2.getText().toString());
                text3.setText(""+result);
            }
        });
    }
}
```

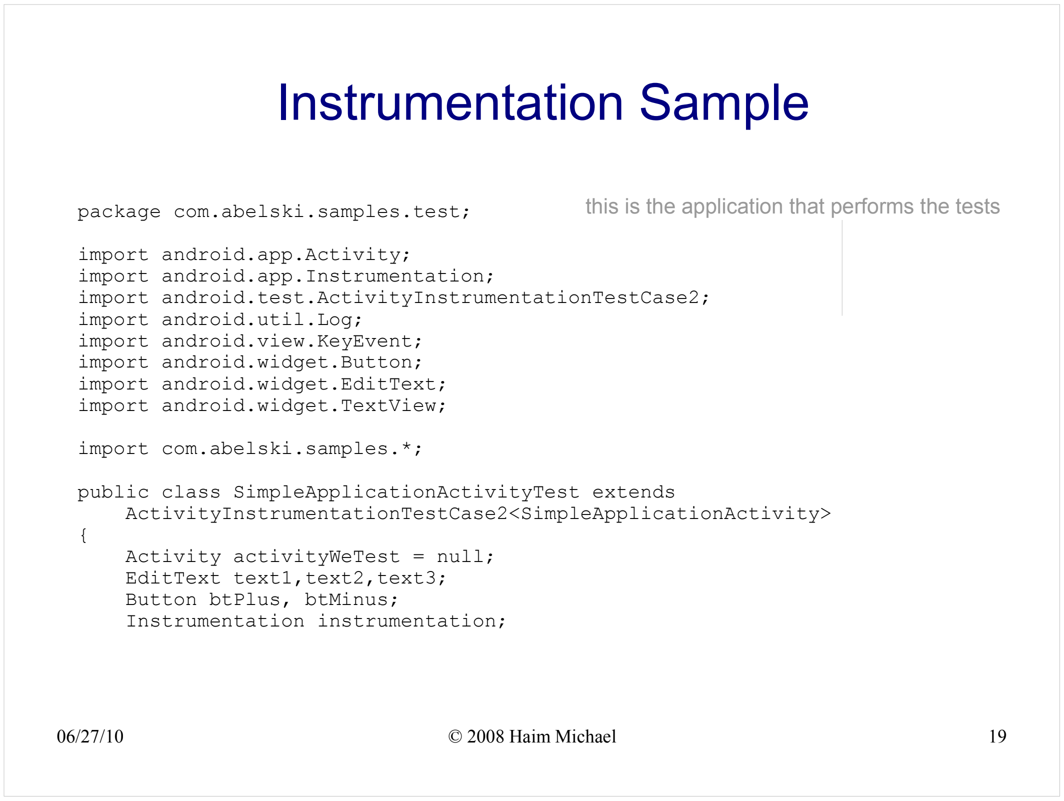


Instrumentation Sample

```
<?xml version="1.0" encoding="utf-8"?>

    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">

        <EditText   android:id="@+id/EditText01"
                    android:layout_height="wrap_content"
                    android:layout_width="fill_parent"
                    android:text="4">
        </EditText>

        <EditText   android:id="@+id/EditText02"
                    android:layout_height="wrap_content"
                    android:layout_width="fill_parent"
                    android:text="3">
        </EditText>

        <Button     android:id="@+id/Button02"
                    android:layout_height="wrap_content"
                    android:text="+"
                    android:layout_width="fill_parent">
        </Button>
```

Instrumentation Sample

```
        <Button     android:id="@+id/Button01"
                    android:layout_height="wrap_content"
                    android:text="-"
                    android:layout_width="fill_parent">
        </Button>

        <EditText   android:id="@+id/EditText03"
                    android:layout_height="wrap_content"
                    android:layout_width="fill_parent">
        </EditText>

</LinearLayout>
```

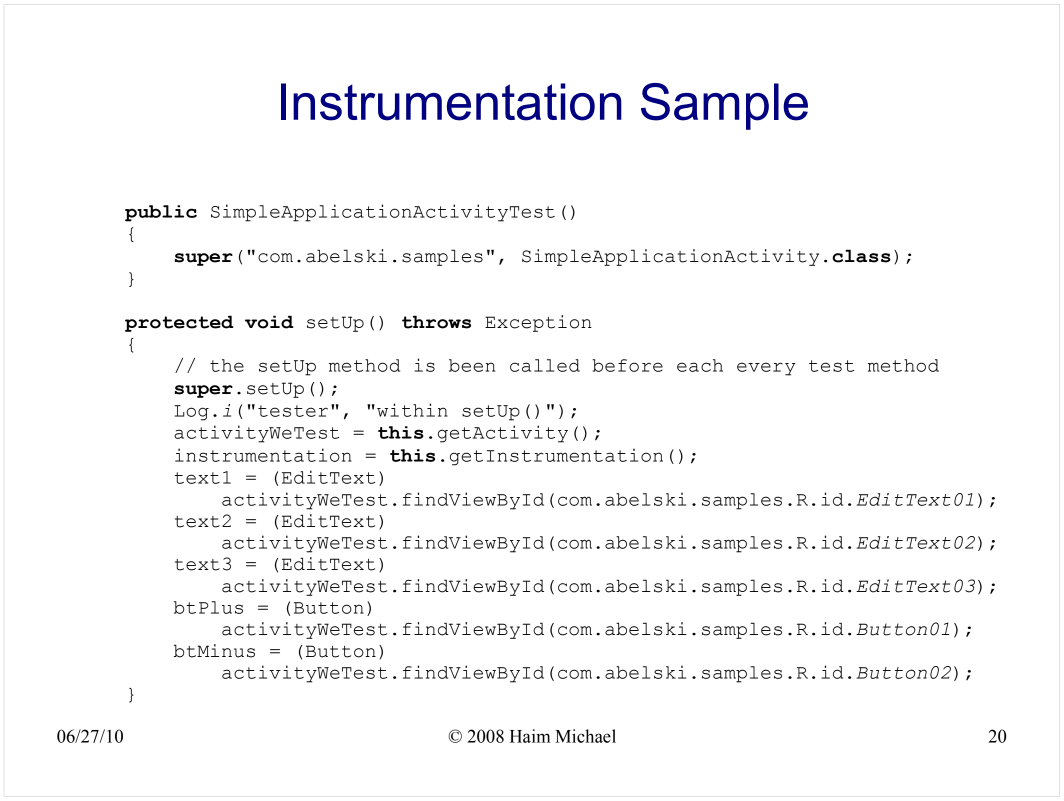


Instrumentation Sample

this is the application that performs the tests

```
package com.abelski.samples.test;

import android.app.Activity;
import android.app.Instrumentation;
import android.test.ActivityInstrumentationTestCase2;
import android.util.Log;
import android.view.KeyEvent;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import com.abelski.samples.*;

public class SimpleApplicationActivityTest extends
    ActivityInstrumentationTestCase2<SimpleApplicationActivity>
{
    Activity activityWeTest = null;
    EditText text1,text2,text3;
    Button btPlus, btMinus;
    Instrumentation instrumentation;
```

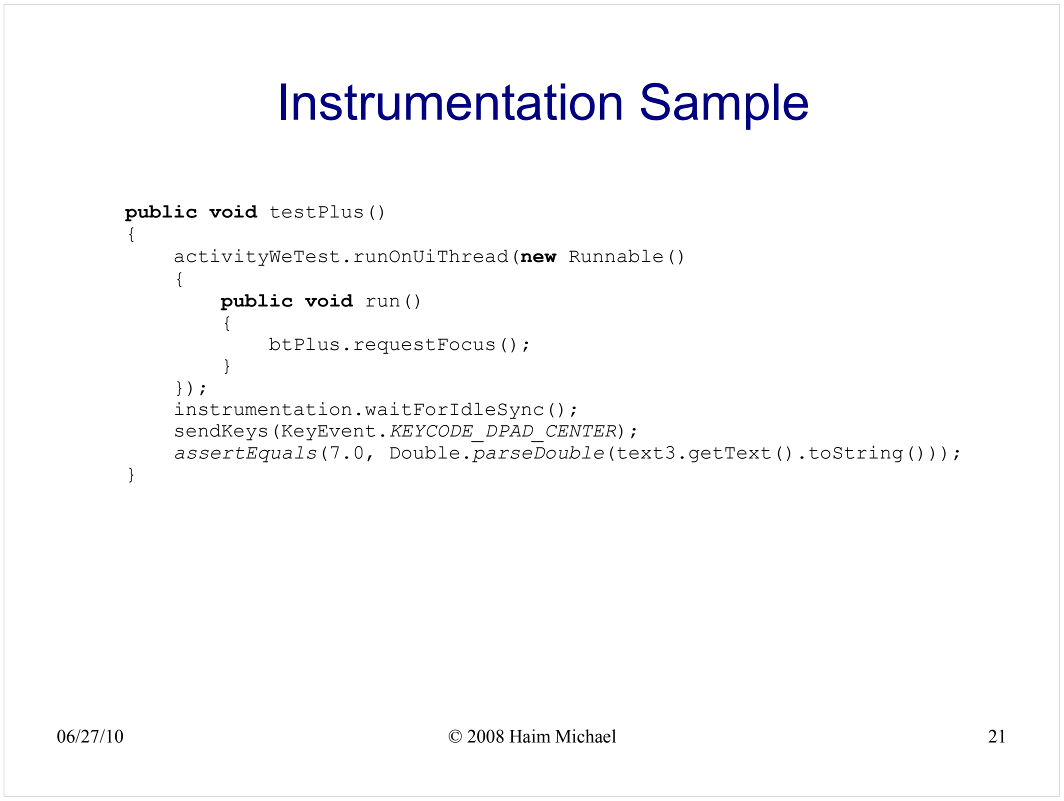
06/27/10 © 2008 Haim Michael 19



Instrumentation Sample

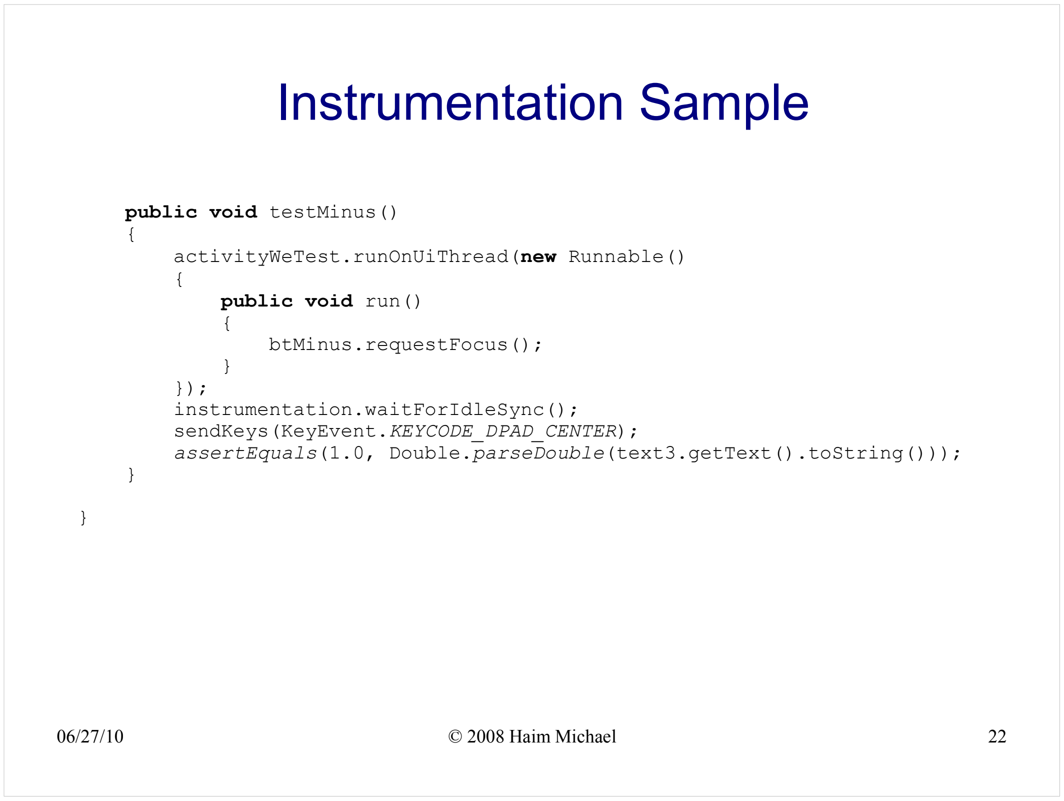
```
public SimpleApplicationActivityTest()
{
    super("com.abelski.samples", SimpleApplicationActivity.class);
}

protected void setUp() throws Exception
{
    // the setUp method is been called before each every test method
    super.setUp();
    Log.i("tester", "within setUp()");
    activityWeTest = this.getActivity();
    instrumentation = this.getInstrumentation();
    text1 = (EditText)
        activityWeTest.findViewById(com.abelski.samples.R.id.EditText01);
    text2 = (EditText)
        activityWeTest.findViewById(com.abelski.samples.R.id.EditText02);
    text3 = (EditText)
        activityWeTest.findViewById(com.abelski.samples.R.id.EditText03);
    btPlus = (Button)
        activityWeTest.findViewById(com.abelski.samples.R.id.Button01);
    btMinus = (Button)
        activityWeTest.findViewById(com.abelski.samples.R.id.Button02);
}
```

06/27/10 © 2008 Haim Michael 20



Instrumentation Sample

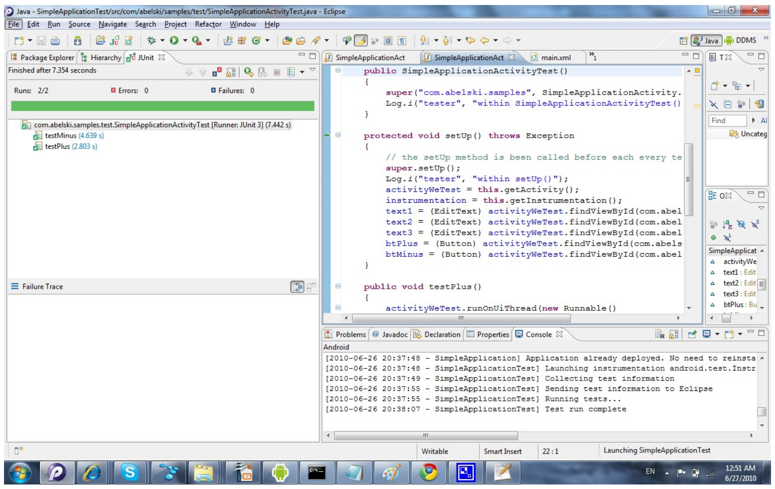
```
public void testPlus()
{
    activityWeTest.runOnUiThread(new Runnable()
    {
        public void run()
        {
            btPlus.requestFocus();
        }
    });
    instrumentation.waitForIdleSync();
    sendKeys(KeyEvent.KEYCODE_DPAD_CENTER);
    assertEquals(7.0, Double.parseDouble(text3.getText().toString()));
}
```



Instrumentation Sample

```
    public void testMinus()
    {
        activityWeTest.runOnUiThread(new Runnable()
        {
            public void run()
            {
                btMinus.requestFocus();
            }
        });
        instrumentation.waitForIdleSync();
        sendKeys(KeyEvent.KEYCODE_DPAD_CENTER);
        assertEquals(1.0, Double.parseDouble(text3.getText().toString()));
    }

}
```

Instrumentation Sample


Java - SimpleApplicationTest/src/com/abelski/samples/test/SimpleApplicationActivityTest.java - Eclipse
File Edit Run Source Navigate Search Project Refactor Window Help
Java DDMS
Package Explorer Hierarchy JUnit
SimpleApplicationAct SimpleApplicationAct main.xml
Problems Javadoc Declaration Properties Console
Failure Trace
Writable Smart Insert 22 : 1 Launching SimpleApplicationTest
EN 12:51 AM 6/27/2010

```
Finished after 7.354 seconds

Runs: 2/2          Errors: 0          Failures: 0

com.abelski.samples.test.SimpleApplicationActivityTest [Runner: JUnit 3] (7.442 s)
    testMinus (4.639 s)
    testPlus (2.803 s)
```

```java
    public SimpleApplicationActivityTest()
    {
        super("com.abelski.samples", SimpleApplicationActivity.
        Log.i("tester", "within SimpleApplicationActivityTest()
    }

    protected void setUp() throws Exception
    {
        // the setUp method is been called before each every te
        super.setUp();
        Log.i("tester", "within setUp()");
        activityWeTest = this.getActivity();
        instrumentation = this.getInstrumentation();
        text1 = (EditText) activityWeTest.findViewById(com.abel
        text2 = (EditText) activityWeTest.findViewById(com.abel
        text3 = (EditText) activityWeTest.findViewById(com.abel
        btPlus = (Button) activityWeTest.findViewById(com.abels
        btMinus = (Button) activityWeTest.findViewById(com.abel
    }

    public void testPlus()
    {
        activityWeTest.runOnUiThread(new Runnable()
```

```
Android
[2010-06-26 20:37:48 - SimpleApplication] Application already deployed. No need to reinsta
[2010-06-26 20:37:48 - SimpleApplicationTest] Launching instrumentation android.test.Instr
[2010-06-26 20:37:49 - SimpleApplicationTest] Collecting test information
[2010-06-26 20:37:55 - SimpleApplicationTest] Sending test information to Eclipse
[2010-06-26 20:37:55 - SimpleApplicationTest] Running tests...
[2010-06-26 20:38:07 - SimpleApplicationTest] Test run complete
```