在 Android 上实现 Junit 单元测试的四部曲

其实在 Android 上实现 JUnit 单元测试也不是很困难，主要是在配置文件和测试环境上将花费很长时间，下面从四步简单讲一下在 Android 上实现 Junit 单元测试。

第一步：新建一个 TestCase，记得要继承 androidTestCase，才能有 getContext() 来获取当前的上下文变量，这在 Android 测试中很重要的，因为很多的 Android api 都需要 context。

Java 代码

```
package szy.android;

import android.test.AndroidTestCase;

/**
*coolszy
*2010-12-31
*/
public class TestMath extends AndroidTestCase
{
private int i1;
private int i2;
static final String LOG_TAG = "MathTest";

@Override
protected void setUp() throws Exception
{
i1 = 2;
i2 = 3;
}

public void testAdd()
{
assertTrue("testAdd failed", ((i1 + i2) == 5));
}

public void testDec()
{
```

```
assertTrue("testDec failed", ((i2 - i1) == 1));
}

@Override
protected void tearDown() throws Exception
{
super.tearDown();
}

@Override
public void testAndroidTestCaseSetupProperly()
{
super.testAndroidTestCaseSetupProperly();
// Log.d( LOG_TAG, "testAndroidTestCaseSetupProperly");
}
}
```

第二步：新建一个 TestSuit，这个就继承 Junit 的 TestSuite 就可以了，注意这里是用的 addTestSuite 方法，一开始使用 addTest 方法就是不能成功。

Java 代码

```
package szy.android;

import junit.framework.TestSuite;

/**
*coolszy 2010-12-31
*/
public class ExampleSuite extends TestSuite
{
public ExampleSuite()
{
addTestSuite(TestMath.class);
}
}
```

第三步：新建一个 Activity，用来启动单元测试，并显示测试结果。系统的 AndroidTestRunner 竟然什么连个 UI 界面也没有实现，这里只是最简单的实现了一个

```
package szy.android;

import junit.framework.AssertionFailedError;
import junit.framework.Test;
import junit.framework.TestListener;
import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;
```

```
import android.os.Handler;
import android.os.Message;
import android.test.AndroidTestRunner;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity
{
private TextView resultView;
private TextView barView;
private TextView messageView;
private Thread testRunnerThread;
private static final int SHOW_RESULT = 0;
private static final int ERROR_FIND = 1;
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
resultView = (TextView) findViewById(R.id.ResultView);
barView = (TextView) findViewById(R.id.BarView);
messageView = (TextView) findViewById(R.id.MessageView);
Button lunch = (Button) findViewById(R.id.LunchButton);
lunch.setOnClickListener(new View.OnClickListener()
{
@Override
public void onClick(View v)
{
startTest();
}
});
}

private void showMessage(String message)
{
hander.sendMessage(hander.obtainMessage(ERROR_FIND, message));
}

private void showResult(String text)
{
hander.sendMessage(hander.obtainMessage(SHOW_RESULT, text));
}
```

```
private synchronized void startTest()
{
if (testRunnerThread != null && testRunnerThread.isAlive())
{
testRunnerThread = null;
}
if (testRunnerThread == null)
{
testRunnerThread = new Thread(new TestRunner(this));
testRunnerThread.start();
} else
{
Toast.makeText(this, "Test is still running", Toast.LENGTH_SHORT).show();
}
}

public Handler hander = new Handler()
{
public void handleMessage(Message msg)
{
switch (msg.what)
{
case SHOW_RESULT:
resultView.setText(msg.obj.toString());
break;
case ERROR_FIND:
messageView.append(msg.obj.toString());
barView.setBackgroundColor(Color.RED);
break;
default:
break;
}
}
};

class TestRunner implements Runnable, TestListener
{
private Activity parentActivity;
private int testCount;
private int errorCount;
private int failureCount;

public TestRunner(Activity parentActivity)
```

```
{
this.parentActivity = parentActivity;
}

@Override
public void run()
{
testCount = 0;
errorCount = 0;
failureCount = 0;
ExampleSuite suite = new ExampleSuite();
AndroidTestRunner testRunner = new AndroidTestRunner();
testRunner.setTest(suite);
testRunner.addTestListener(this);
testRunner.setContext(parentActivity);
testRunner.runTest();
}

@Override
public void addError(Test test, Throwable t)
{
errorCount++;
showMessage(t.getMessage() + "\n");
}

@Override
public void endTest(Test test)
{
showResult(getResult());
}

@Override
public void startTest(Test test)
{
testCount++;
}

private String getResult()
{
int successCount = testCount - failureCount - errorCount;
return "Test:" + testCount + " Success:" + successCount + " Failed:" + failureCount + "
Error:" + errorCount;
}
```

```
@Override
public void addFailure(Test test, AssertionFailedError t)
{
failureCount++;
showMessage(t.getMessage() + "\n");
}
}
}
```

第四步：修改 AndroidManifest.xml, 加入，不然会提示找不到 AndroidTestRunner，这里需要注意是这句话是放在 applications 下面的，我一开始也不知道，放错了地方，浪费了不少时间

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="szy.android"
android:versionCode="1"
android:versionName="1.0">
<application android:icon="@drawable/icon" android:label="@string/app_name">
<activity android:name=".MainActivity"
android:label="@string/app_name">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
<uses-library android:name="android.test.runner" />
</application>
<uses-sdk android:minSdkVersion="9" />

</manifest>
```

最后把 main.xml 也附上

```
<?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"
>
<TextView
android:id="@+id/ResultView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
/>
<TextView
android:id="@+id/BarView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
```

```
/>
<TextView
android:id="@+id/MessageView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
/>
<Button
android:id="@+id/LunchButton"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="测试"
/>
</LinearLayout>
```