**My sprint templet**

# User Case Sprint

# My Activities:

## 下载SDK from 地址：[http://developer.android.com/sdk/index.html](http://developer.android.com/sdk/index.html" \t "_blank)

## Unzip on Ocean-pc

## Exe run sdk manager.ere

## Exe run 虚拟机 avd manager.exe

## Download and install android studio on Pc-pc

# Diagram

# All excerpt

[**MyEclipse 10下搭建Android开发环境[Win8]**](http://www.cnblogs.com/techpalm/archive/2013/02/08/2910924.html)

今天试了下在win8系统下搭建Android的开发环境，所用的开发平台用的是myeclipse 10，相较于eclipse，myeclipse拥有更多的功能，更易于提高开发者的开发效率。

Youtube-Introduction to Mobile testing – Mobile Testing Tutorial Video 1-15

Different types of Mobile testing

1. UI testing
2. Functional Testing
3. Performance Testing
4. Network Performance Testing
5. Installation Testing
6. Interruption testing

Manual testing

Emulator: with (like real) device (like real) ui real pc (like real) mobile, mimic mobile envirement

Setup android driver and emulator

Simulator: only check use interface without real device, not mimic real

Automation test

Tools:

1. Android driver(just testing android website)- you cannot test mobile app

* Is Emulator, is AVD android visual device

1. Perfecto mobile (paid tool)
2. Monkeytalkfree (free)
3. Robotium(open source)- android app
4. Appium
5. Ranorex (paid) – Test mobile app
6. keynote – [www.deviceanywhere.com](http://www.deviceanywhere.com)

Host lots real mobile device on their data center

Less availability of mobile testing tools, identifying appropriate tools, no generic testing framework

Mobile device diversity:

60+ Android, 7-10 iOS mobile devices with various:

* Screen size; Resulution; Form Factor; Hardware configuration etc;

Mobile Device Connectivity:

Windows/MacConnectivity options:

* Wi-Fi, USB, MobileCloud

Frequent Release:

* major/minor OS releases, app upgrde, bug fixes

Varying network condition:

Different network conditions;

-Wi-Fi, 2G/3G/4G, GSM/CDMA, Bandwidth,Connection Speed etc,

Differenc Mobile app types:

Mobile applications landscape:

* Mobile web apps
* Native apps
* Hybrid apps(includes features of both, web & native apps)

Test execution challenge:

Test execution across multiple:

* Mobile devices(including different OS and Manufactures)
* Browsers (chrome, safari, ie, firefox etc.)

Mobile automations & Performance testing :

High cost, Mobile apps object identification, lack of right and complete solution

Target device selection:

* optimal mix of emulator and real device of different models to maximize test coverage

Network environment;

* consider primary tresting on WI-fi or network simulator and then real network conditions;

identifying types of testing:

Combinations fo testing approaches:

* Functional; Performance; Security; Compliance; Usability: Beta testing etc.

Mobile Cloud Testing:

--Cloud testing environments have capability to support complex apps and provide real-time testing results

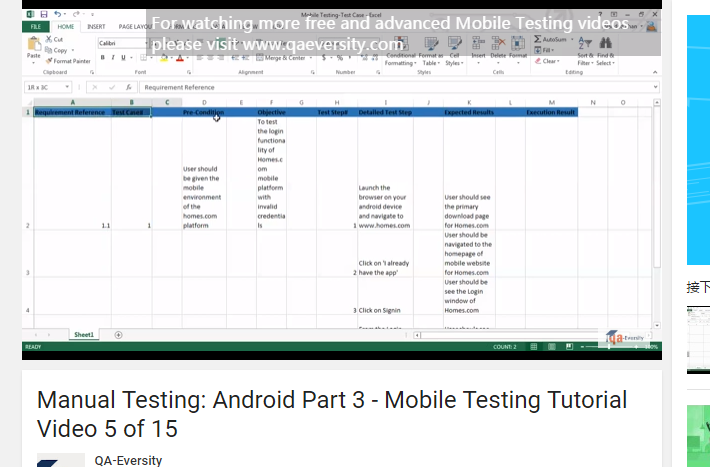
Mobile Test automation:

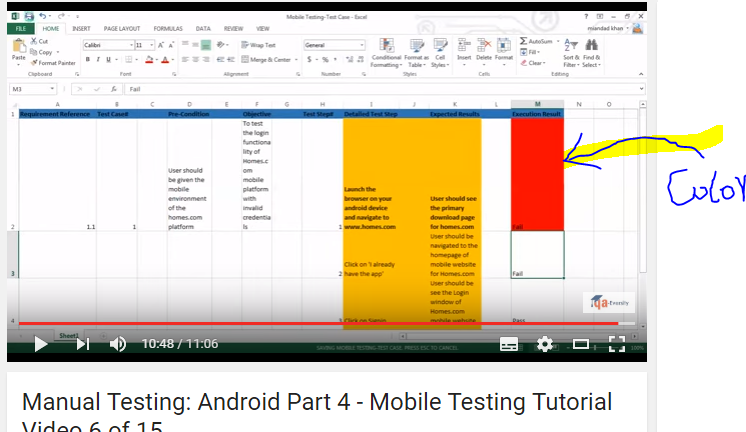
-Selecting appropriate mobile automation solution

-Regression automation

Test case

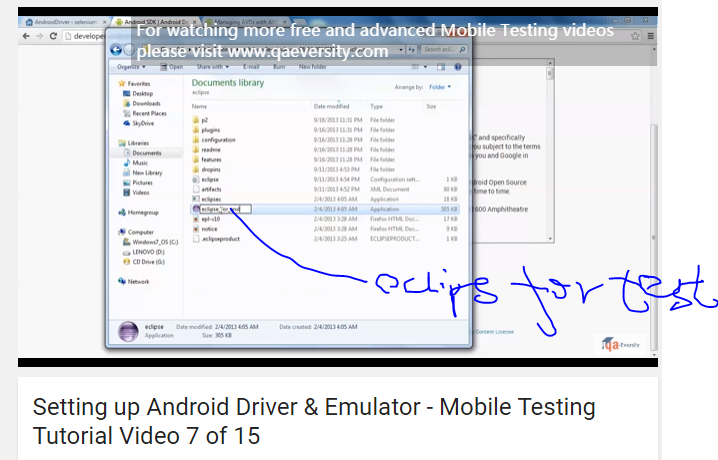
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H | I | J | K | L | M |
| Requirement reference | Test case # |  | Pre-Conditions |  | Objective |  | TestStep# | Detailed test step |  | Expected results |  | Execution result |

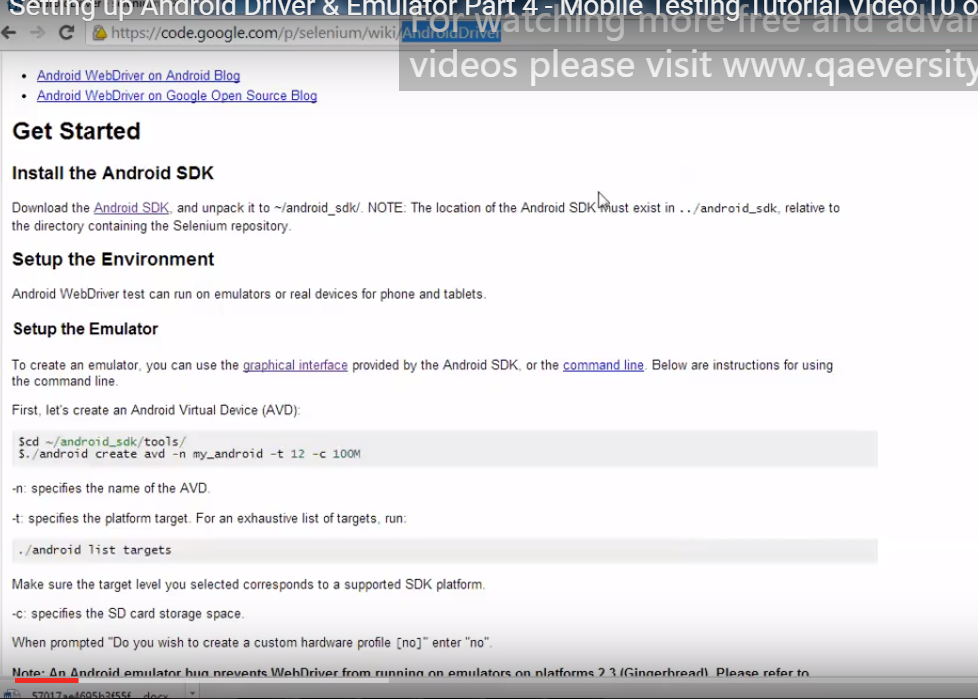




SDK include a eclipse, which could be used for testing with Android driver to testing

As following





**在搭建Android开发环境之前，首先确保你有：**

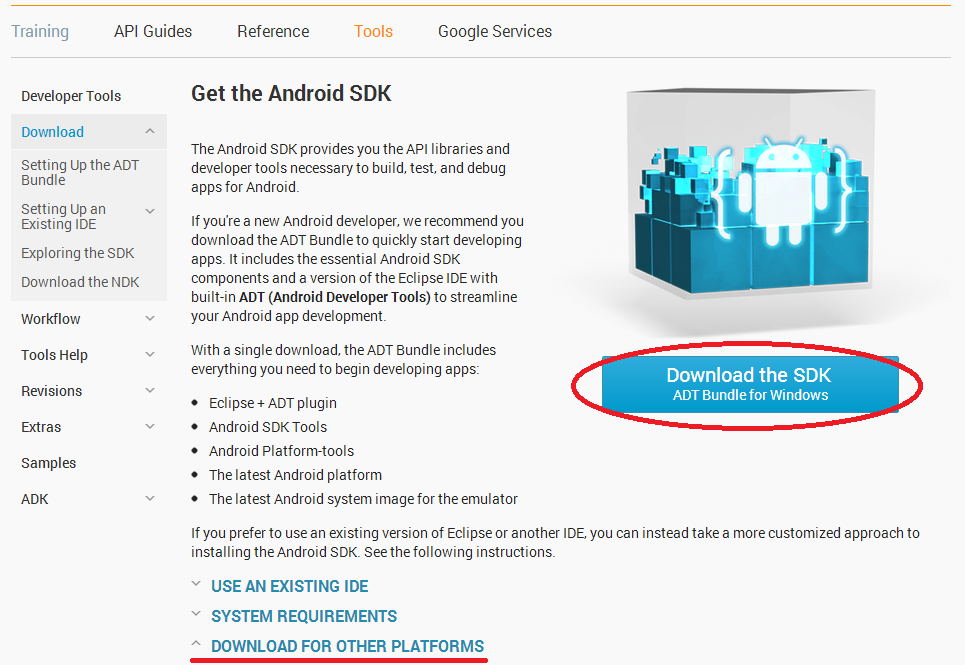
1.Java开发环境

2.安装有myeclipse 10的破解版

（下载地址在：<http://download.csdn.net/detail/ll136078/4018765>）

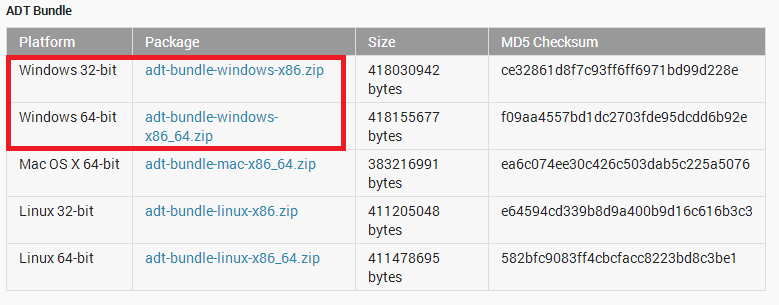
**第一步 下载最新sdk以及工具包**

下载地址：<http://developer.android.com/sdk/index.html>

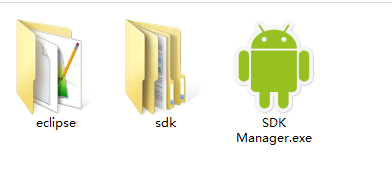


如果你是Windows平台，可以直接选择"Download the SDK",如果是Linux或者是MacOS可以选择"Download For other Plantforms"

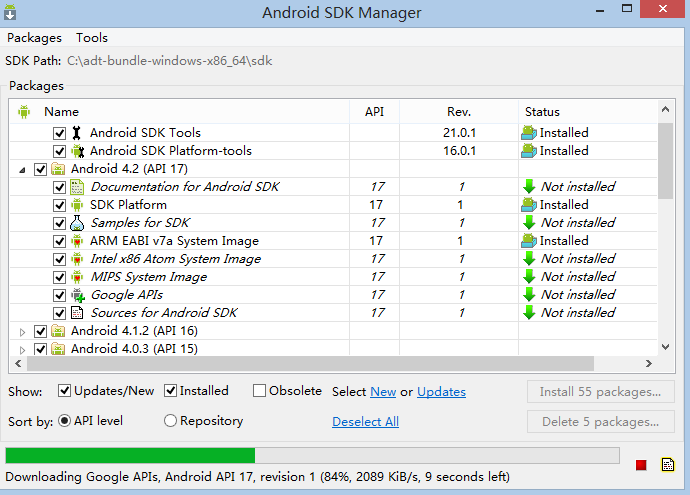
我是Windows平台，所以选择直接下载Windows 64位的解压包



下载完成后，执行SDK Manager.exe



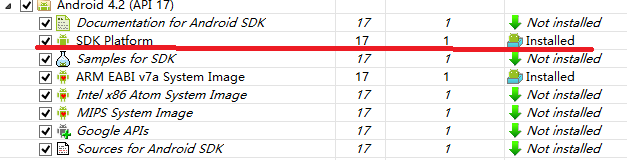
运行后，如下图：



这里是选择你要安装的更新和软件包，我这里选择Android4.2~Andriod 2.3（毕竟Android系统还是向下兼容，而且大多数手机仍是Android2.3，建议选上2.3）

下载的时间根据你的网速和安装包的数量，一般全部选上要3G左右。。过程非常漫长

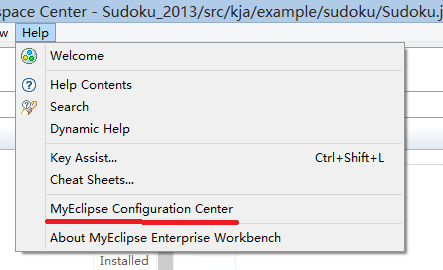
下载和安装是同时进行的，所以安装完这些API后，会在相应包的状态里显示"installed"



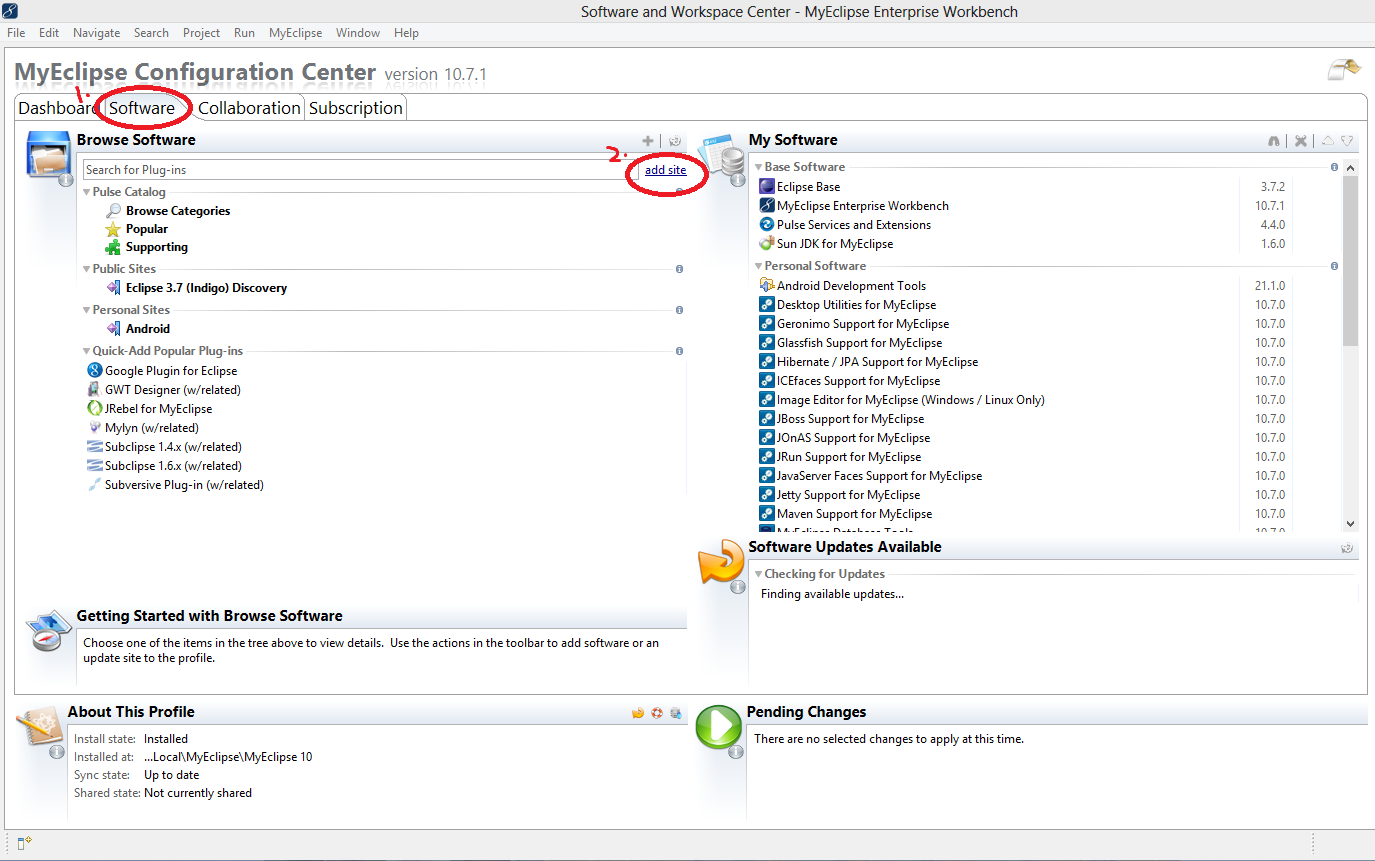
**第二步 在MyEclipse 10下安装Android Development Tools (ADT) 插件**

要想在MyEclipse 10下开发Android，就必须下载相关Android插件，Google官方提供了给eclipse的开发插件，同样可以用在MyEclipse 10平台上

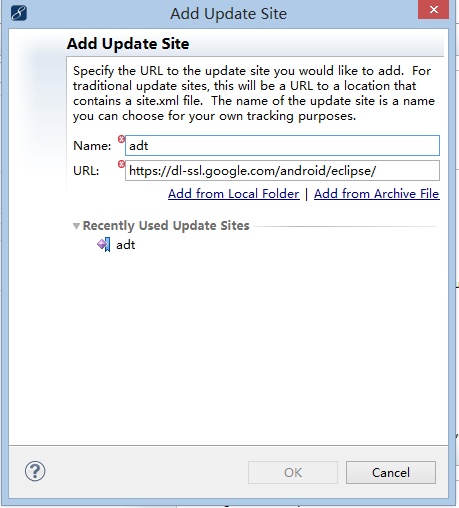
选择Help->MyEclipse Configuration Center



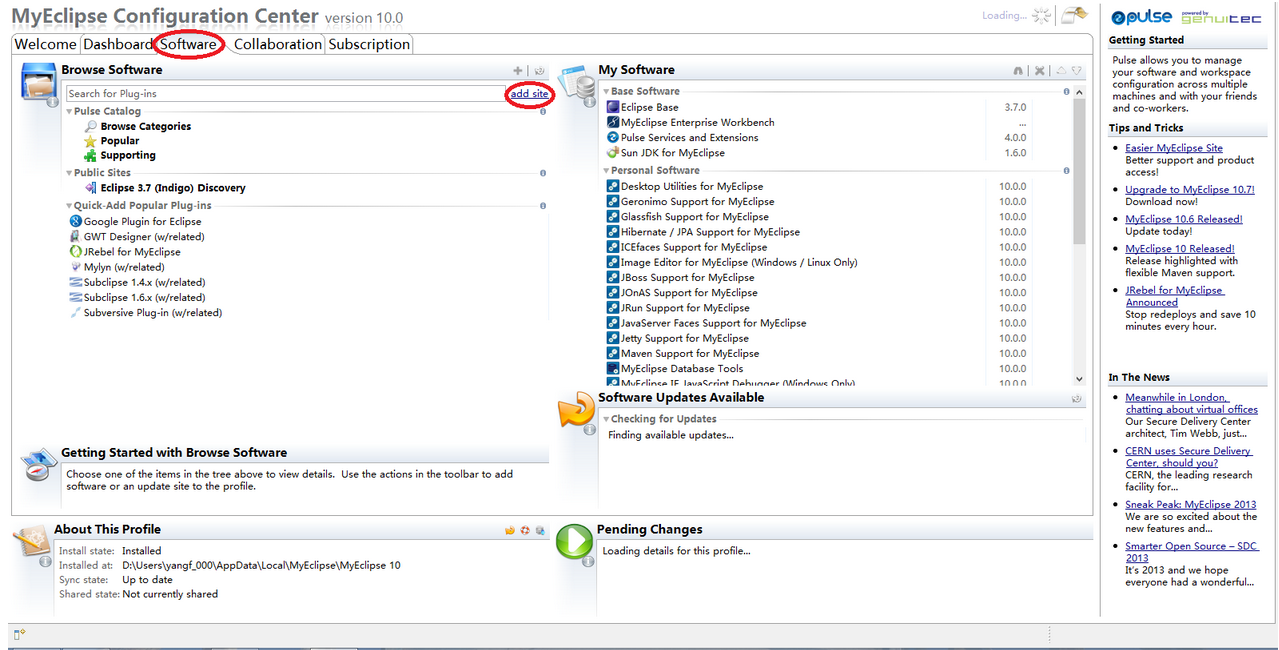
进入configuration center 后选择software选项卡，再点击add site，添加myeclipse的Android开发插件



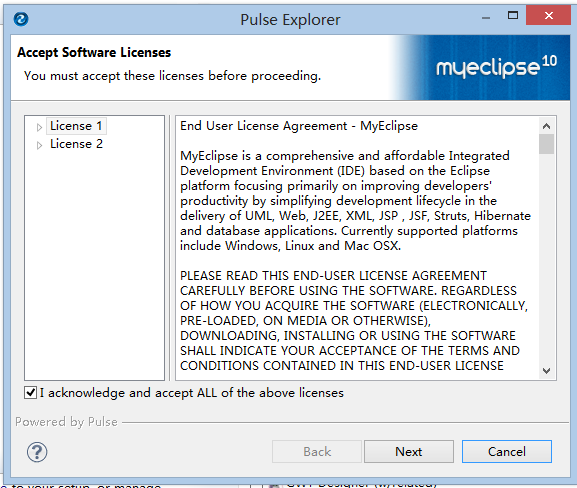
弹出的add site窗口，name可以随便填，URL填 https://dl-ssl.google.com/android/eclipse/



关于ADT的安装步骤，可以参考<http://developer.android.com/tools/sdk/eclipse-adt.html>

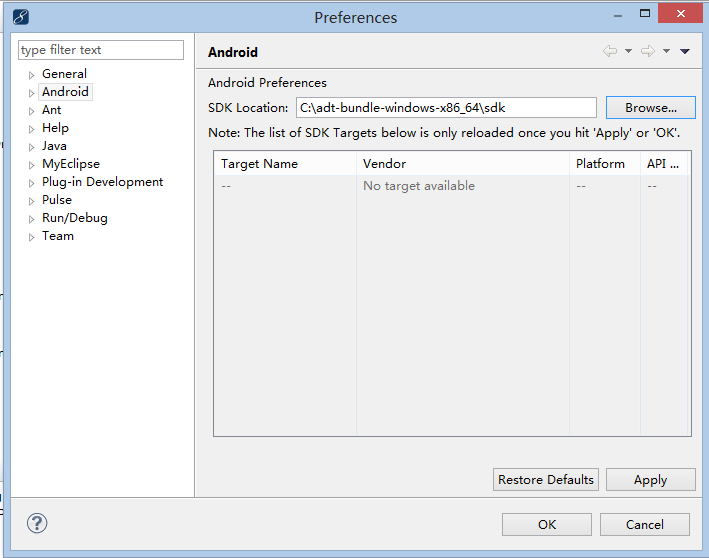


经过下载后，MyEclipse 10会对环境进行部署，速度会比较慢，全部部署完以后会要求重启MyEclipse 10



**第三步 选择SDK路径**

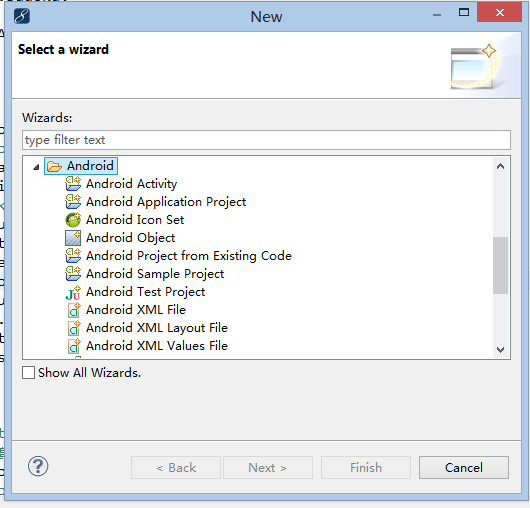
重启之后，会让你选择SDK路径，选择你之前安装的sdk路径即可



全部设置完成后，应该在MyEclipse 10的工具栏上显示Android SDK manager和虚拟机的manager

http://images.cnitblog.com/blog/496148/201303/03132718-0996f4f57fdb4705863dd66330e6ed84.png

现在你就可以新建Android项目进行开发啦~



说明Android环境正式部署完成！

一年之后，我会非常庆幸自己今天选择了开始。 当我觉得为时已晚的时候，恰恰是最早的时候。 It does not matter how slow I go so long as I do not stop.

**Android 开发环境搭建（MyEclipse+Android sdk+ADT环境）**

2014-04-10 18:18:35

标签：[Android](http://blog.51cto.com/tag-Android.html" \t "_blank) [Windows](http://blog.51cto.com/tag-Windows.html) [software](http://blog.51cto.com/tag-software.html) [管理工具](http://blog.51cto.com/tag-%E7%AE%A1%E7%90%86%E5%B7%A5%E5%85%B7.html) [对话框](http://blog.51cto.com/tag-%E5%AF%B9%E8%AF%9D%E6%A1%86.html)

所需开发环境;

[Eclipse](http://www.eclipse.org/downloads/) 3.3 (Europa), 3.4 (Ganymede)

(注：如果使用Eclipse，其他的版本有的会导致一些缺少包，可能会导致最后安装配置ADT时有误。这个我在网上看到的，但是我没有使用这些，所以真实性待考证)

安装jdk（1.5或者1.6版本）

设置jdk的path与classpath(略)

安装eclipse（我安装的是myeclipse 8.5M1版，因为之前做别的东西时就已经安装的）

安装Android sdk，可以在[这里](http://www.android123.com.cn/android_kit.html" \t "_blank)下载最新版本，（如果是第一次安装sdk，**可以直接使用sdk setup工具** ，[使用方法](http://www.android123.com.cn/zhongwensdk/366.html" \t "_blank)）

设置环境变量：

**Windows环境下** , 右击我的电脑, 并选择属性. 在标签页高级, 点击环境变量, 当对话框出现, 在系统变量栏目里双击路径 ( Path ) .并添加tools/ 文件夹的完整路径.（利用sdk setup管理工具貌似也可以设置）

**安装配置ADT：**

这个方面有很多教材与网友的方法，但是有些在我的这里无法使用，不过我依旧给整理下

**方法一：** .启动myeclipse->help->software updates->find and install->search for new features to install->new remote site下选择填写了 <https://dl-ssl.google.com/android/eclipse/> 地址

然后点击finish

如果有误，可以将https,改为http。[http://dl-ssl.google.com/android/eclipse/](https://dl-ssl.google.com/android/eclipse/" \t "_blank)

找不到software updates这个选项的话可以在windows->preference->general->capabilities勾选classic updates,然后apply，确定

（目前我使用这个方法没有成功，因为这个网址至少在墙内无法登陆，原因我没去查）

**方法2：** 启动启动myeclipse->help->software updates->find and install->search for new features to install->new local site下选择已经下好的ADT解压文件，然后应用

（我使用这种方法也会出错，而且两次的错误还不同囧rz）

**方法3：把下载的adt文件解压后，拷贝feature与plugin文件夹到myeclipse安装目录下的dropins文件下，重启myeclipse**

（我最后是用这个方法成功的）

安装成功后

重启MyEclipse之后，设置MyEclipse的参数，使之指向开发包的根目录（$SDK\_ROOT）：   
  a) 选择Window>Preferences…打开参数设置面板   
  b) 在左侧面板中选择Android。   
  c) 点击右侧面板的SDK Location后面的地址栏中输入SDK包的地址，或者点击Browse按钮在系统中寻找。

如果一切成功，可以利用下面方法来测试下

File->New->Project->Android->Android Project

如果可以在project目录下发现Android则之前的操作成功

**测试入门—Android应用测试指南(1)**

发布时间: 2016-3-10 09:23    作者: 李江    来源: 51Testing软件测试网原创

字体:  [小](javascript:doZoom('12');)  [中](javascript:doZoom('14');)  [大](javascript:doZoom('16');)  | [上一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=up&itemid=229485&catid=96) [下一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=down&itemid=229485&catid=96) | [打印](javascript:doPrint();)  | [我要写文](http://bbs.51testing.com/thread-77515-1-1.html)  | 推荐标签： [Android测试](http://www.51testing.com/?action-tag-tagname-Android%B2%E2%CA%D4) [软件测试技术](http://www.51testing.com/?action-tag-tagname-%C8%ED%BC%FE%B2%E2%CA%D4%BC%BC%CA%F5) [测试丛书](http://www.51testing.com/?action-tag-tagname-%B2%E2%CA%D4%B4%D4%CA%E9)

　　（51Testing软件**[测试](javascript:;" \t "_self)**网获人民邮电出版社和作者授权连载本书部分章节。任何个人或单位未获得明确的书面许可，不得对本文内容复制、转载或进行镜像，否则将追究法律责任。）

**第1章  测 试 入 门**

　　本章介绍了不同类型的测试方法，它们在**[软件开发](javascript:;" \t "_self)**项目工程中的基本用法以及在[**Android**](javascript:;)项目中特殊用法。

　　关于"Android"和"开放**[手机](javascript:;" \t "_self)**联盟"，很多书中都有谈及，我们就不累述。本书涉及更高级的主题，我们希望在您阅读本书之前，最好有Android程序开发经验。不过，我们会先回顾一下测试的基本概念、技术、框架以及Android平台上的测试工具。

**1.1  简史**

　　Android平台是在2007年末引进的。那时候，基于Android平台测试的技术支持很少，而且我们中一些人习惯于边开发边测试，将测试作为开发流程中紧密耦合的一部分，因此，是时候开发一些框架和工具来支持这种测试方式了。

　　那时候，Android平台用Junit提供了一些不够成熟的功能支持**[单元测试](javascript:;" \t "_self)**，但是，支持力度不够并且帮助文档很少。

　　在我编写自己的库和工具的过程中，发现了Phil Smith的Positron库。他的库是开源的，非常适用于**[Android测试](javascript:;" \t "_self)**。于是，我在他的杰作基础上进行补充，新增功能、弥补不足。他们的库中不包含某些[**自动化测试**](javascript:;)的东西，所以我新起了一个项目与之互补，项目命名为Electron。Positron和Electron两个项目，当然不是像真正的正反粒子那样互斥，相反，它们像正反粒子那样蕴含着大能量，能产生大量的光波。

　　后来，2008年初，Electron项目参加第一届Android开发挑战赛。虽然在一些类目中，Electron的分数不错，但是在框架类项目比赛中毫无立足之地。那个时候，Eclipse上已经可以执行单元测试了。但是，并不是在真机上执行测试，而是在本地开发机上的JVM虚拟机上。

[**Google**](javascript:;)也提供了执行应用程序的模拟器代码，通过Instrumention类实现了这一功能。当你打开模拟器运行程序时，Instrument类会在你应用程序之前初始化，可以通过Instrument来模拟各种系统交互，执行程序。我们通过AndriodManifest.xml文件来设置模拟器。

　　在Android发展演变早期，我开始在博客里写一些**[文章](javascript:;" \t "_self)**来弥补这块测试的空白。本书就是将这些[**工作**](javascript:;)的演变和完成过程，用一种有序、容易理解的方式写下来，让你接触那些Android测试中的问题。

**1.2  软件[Bug](javascript:;" \t "_self)**

　　无论你多努力，代码设计花费多少时间，编程时候有多小心，你的程序中都会有Bug,这是不可避免的。

　　Bug和软件开发是息息相关的。硬件工程中，用Bug这个单词来描述瑕疵、问题、错误已经有几十年了，甚至比计算机出现得还早。尽管如此，关于Bug这个单词的故事是由哈佛大学的Mark II计算机操作员创造的，1878年，在托马斯爱迪生给普斯卡斯蒂瓦达的信中可以看到这个单词的早期应用。

　　"我所有的发明都如此。第一步是直觉，随之是头脑风暴，然后困难都浮现出来。这些困难一点点被解决，然后Bug出现了，这些Bug就是所谓的小错误和困难。Bug出现后，需要投入几个月的精力去密切观察、**[学习](javascript:;" \t "_self)**，最终达到商业上的成功，否则，必然失败。"

　　Bug是如何严重影响你的项目的呢？

　　众所周知，Bug会从很多方面对你的项目工程造成影响，越早发现修复越好。无论你是为了优化用户体验， 开发一个简单的Android程序；还是为设备操作员新建一个Android客户版本，Bug都在延迟你的交付时间，燃烧你的金钱。

　　在所有的软件研发模式中，"测试驱动开发"是软件开发流程中最敏捷的方式。它驱使你在开发过程中更早地发现、面对Bug，你也很可能预先解决更多的问题。

　　此外，比起那些在最后才进行测试的团队，利用这种测试驱动开发模式的研发团队，生产效率更高。如果你在[**移动**](javascript:;)行业参与软件开发，有理由相信赶时间的情况下，"测试驱动开发"这种方案不合适。因为，通常这种方式解决的问题都很可能是已经规避了的，这点很有趣。

　　2002年美国国家研究所的一项研究调查表明，每年软件Bug损失595亿美元，如果**[软件测试](javascript:;" \t "_self)**执行得更好的话，超过三分之一的损失是可以避免的。

　　然而，请别误解以上所说的。软件开发没有特效药，是什么让你高效，让你的项目易于管理？是你有条理地利用这些方法和技术，掌控你的项目。

**测试的种类—Android应用测试指南(2)**

发布时间: 2016-3-11 11:44    作者: 李江    来源: 51Testing软件测试网原创

字体:  [小](javascript:doZoom('12');)  [中](javascript:doZoom('14');)  [大](javascript:doZoom('16');)  | [上一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=up&itemid=229485&catid=96) [下一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=down&itemid=229485&catid=96) | [打印](javascript:doPrint();)  | [我要写文](http://bbs.51testing.com/thread-77515-1-1.html)  | 推荐标签： [Android测试](http://www.51testing.com/?action-tag-tagname-Android%B2%E2%CA%D4) [软件测试技术](http://www.51testing.com/?action-tag-tagname-%C8%ED%BC%FE%B2%E2%CA%D4%BC%BC%CA%F5) [测试丛书](http://www.51testing.com/?action-tag-tagname-%B2%E2%CA%D4%B4%D4%CA%E9)

　　（51Testing软件**[测试](javascript:;" \t "_self)**网获人民邮电出版社和作者授权连载本书部分章节。任何个人或单位未获得明确的书面许可，不得对本文内容复制、转载或进行镜像，否则将追究法律责任。）

**1.4  测试的种类**

　　在开发过程中，任何时间段都可以参与测试，这取决于采用何种测试方案。但是，我们推荐测试[**工作**](javascript:;)在项目开发早期就介入，甚至可以在完整需求出来之后、刚开始开发的时候就开始做准备。

　　基于被测对象的不同，有好几种不同的测试方法。但是无论采用哪种测试方法，[**测试用例**](javascript:;)都包含执行条件和执行结果，执行结果返回True或者False来表示用例是否正确。

**1.4.1  单元测试**

[**单元测试**](javascript:;)，指的是程序员在开发阶段写的测试用例。这种测试用例需要将被测对象独立隔离起来，也就是mock掉外部关联对象。单元测试用例应用是可以重复执行的。这也是为什么我们常把单元测试和mock对象关联在一起。因为你要通过mock对象来模拟外部交互从而达到隔离被测对象的目的。当然，这样的用例可以重复执行任何次数。例如，假设你要从**[数据库](javascript:;" \t "_self)**中删除某些数据，但是下一次执行这个用例时这些数据还需要用，因此不想这些数据真正被删除，这时候mock数据库的返回，假装数据已经删除成功了。

[**Junit**](javascript:;)是约定俗成的标准单元测试框架。它是一个简单、开源的自动化单元测试框架，由ErichGamma和KentBeck两位作者创建。

[**Android**](javascript:;)要用Junit 3。这个版本没有注释，而是通过内部自查来感知测试用例的。一个典型的Junit测试用例写法如框1.1中所示的代码，其中测试方法用高亮度显示：

**框1.1  Junit测试代码样例**

|  |
| --- |
| /\*\*  \* Android Application Testing Guide  \*/  package com.example.aatg.test;  import JUnit.framework.TestCase;  /\*\*  \* @author diego  \*/  public class MyUnitTests extends TestCase {  private int mFixture;  /\*\*  \* @param name test name  \*/  public MyUnitTests(String name) {  super(name);  }  /\* (non-Javadoc)  \* @see JUnit.framework.TestCase#setUp()  \*/  protected void setUp() throws Exception {  super.setUp();  mFixture = 1234;  }  /\* (non-Javadoc)  \* @see JUnit.framework.TestCase#tearDown()  \*/  protected void tearDown() throws Exception {  super.tearDown();  }  /\*\*  \* Preconditions  \*/  public void testPreconditions() {  }  /\*\*  \* Test method  \*/  public void testSomething() {  fail("Not implemented yet");  }  } |

　　如果你购买了Packt书，可以访问http://www.PacktPub.com，在个人账户中下载样本源代码。如果你是从其他地方购买的书，可以访问http://www.PacktPub.com/support来注册用户，然后我们将源码文件直接发E-mail给您。

**我们将在下一节详细阐述测试用例的每个细节。**

**1．测试套件**

　　测试套件是个为人熟知的名词，它表示执行用例的标准流程模式。每个测试用例都用同一套标准流程。因此，它也是测试用例设计的基础。

　　通常情况下，按照Android的约定，它由一系列成员变量构成。通常以m开头，如： mActivity。但是，它也有一些扩展数据，作为数据库和文件系统操作的特殊入口。

**2．setUp方法**

　　这个方法是用来初始化测试套件用的。

　　通过重载这个方法，你可以新建对象，初始化元素。在每个测试用例执行之前，这个SetUp方法都会执行一次。

**3．tearDown方法**

　　tearDown方法是在测试套件中最后执行的函数。

　　在测试用例执行过程中，会初始化一些对象，这些对象可以在tearDown函数中进行销毁。因为tearDown函数是每个测试用例最后必须执行的，是销毁对象的最佳阶段。

　　比如：你可以在tearDown中释放掉数据库连接以及网络连接。

　　Junit设计的流程是这样的：首先，将整个库的用例都编译完。然后，在第二阶段再执行测试用例。因此，在测试执行过程中，执行器对所有用例都有强依赖。也就是说，对于那些用例很多、耗时很长的用例来说，在所有用例完成之前，是不会对变量、对象进行回收的。这点在**[Android测试](javascript:;" \t "_self)**中特别重要，因为在某些设备上测试失败的原因不是因为固有的逻辑问题，而是因为用例执行太多导致资源不足了。

　　因此，在Android应用中，你若测试使用了额外的、有限的资源，比如Services服务或者contentProvides，那么，你应该注意要及时释放掉。在tearDown方法中，严格遵守将对象设置成null的规则，以便及时回收，避免一直占用资源，一直到所有用例跑完才释放。

**Android软件测试—Android应用测试指南(3)**

**第2章  Android软件测试**

　　既然我们已经了解了测试的基本概念及其重要性，那么现在是时候付诸于实践了。

　　在本章，我们将要讲述：

　　"  在**[Android](javascript:;" \t "_self)**平台上测试；

　　"  单元测试和Junit用法；

　　"  创建一个**[Android测试](javascript:;" \t "_self)**工程；

　　"  测试执行。

　　我们会创建一个简单的Android主程序和与之对应的测试工程项目。主工程基本上是一个空项目，因为我们将重点看测试部分。以我个人经验，建议没有任何Android测试经验的同学好好**[学习](javascript:;" \t "_self)**下本章。如果你有过Android工程的经验，并且用过相关的**[测试技术](javascript:;" \t "_self)**，那么你完全可以以浏览的方式读读本章，复习一下概念即可。

　　实践证明，测试最好在独立的、没有干扰的环境中执行，当然，这不是强制措施。Android的ADP插件支持这个功能，但也做不到完全隔离。前段时间，我曾经发表过一篇**[文章](javascript:;" \t "_self)**（http://dtmilano.blogspot.com/2008/ll/andriod-testing-on-andriod-platf.html），文章描述了人工获得两个相关工程的方法-主工程和测试工程。文章中并没有数据证明隔离测试的优势，但是，我们可以发现：

　　"  测试代码可以很容易地从生产环境构件中剥离出来，因为它不在主项目中，因此不会被包含在APK结果中。

　　"  通过在开发工具中设置模拟器来执行**[测试用例](javascript:;" \t "_self)**，变得更加简单。

　　"  当测试工程和开发工程分开之后，对于大项目来说，部署编译包花费的时间更少了。

　　"  在同一个项目中，对代码利用率的要求更高了。

**2.1  Junit**

　　前面一章我们已经对Junit进行了简单介绍，因此这里就不再累述。值得一提的是，Android测试项目的默认测试框架是Junit。Eclipse，AndroidADT插件以及Ant都支持Junit框架，所以你不用担心自己没有使用IDE开发而不能使用Junit。

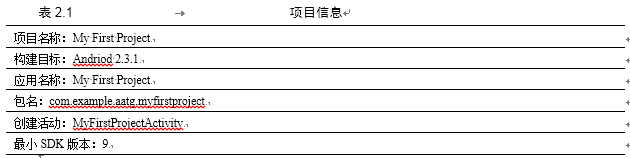
　　你可以自由选择测试框架。

　　后面大多数的实例都是基于Eclipse的，因为Eclipse也是最常用的工具。那么，我们不废话了，打开Eclipse一起开始学习。

**2.2  创建一个Android主项目**

　　我们先创建一个新的Android项目。在Eclipse里面单击菜单栏File-new-Project -Andriod-Andriod Project。

　　然后，给组件命名如下，我们取个特殊点儿的名称，信息如表2.1所示。

[](http://www.51testing.com/batch.download.php?aid=58379)

　　最小SDK版本：9

　　当你输入完这些值时，会出现下面的对话框，如图2.1所示。

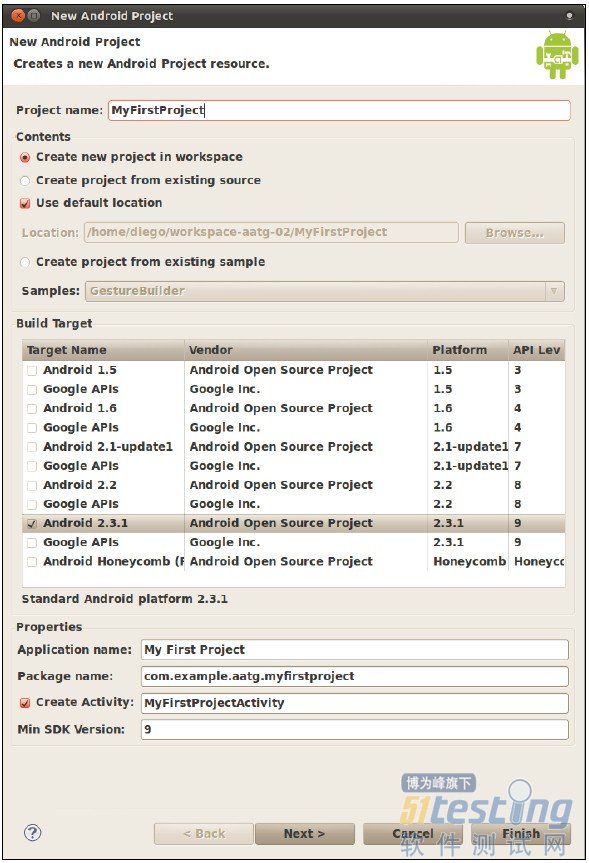
[](http://www.51testing.com/batch.download.php?aid=58377)

图2.1  新建Andriod工程对话框

**测试驱动开发—Android应用测试指南(4)**

发布时间: 2016-3-15 09:22    作者: 李江    来源: 51Testing软件测试网采编

字体:  [小](javascript:doZoom('12');)  [中](javascript:doZoom('14');)  [大](javascript:doZoom('16');)  | [上一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=up&itemid=229485&catid=96) [下一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=down&itemid=229485&catid=96) | [打印](javascript:doPrint();)  | [我要写文](http://bbs.51testing.com/thread-77515-1-1.html)  | 推荐标签： [Android测试](http://www.51testing.com/?action-tag-tagname-Android%B2%E2%CA%D4) [软件测试技术](http://www.51testing.com/?action-tag-tagname-%C8%ED%BC%FE%B2%E2%CA%D4%BC%BC%CA%F5) [测试丛书](http://www.51testing.com/?action-tag-tagname-%B2%E2%CA%D4%B4%D4%CA%E9)

　　（51Testing软件**[测试](javascript:;" \t "_self)**网获人民邮电出版社和作者授权连载本书部分章节。任何个人或单位未获得明确的书面许可，不得对本文内容复制、转载或进行镜像，否则将追究法律责任。）

**第4章  测试驱动开发**

　　这章节介绍测试驱动开发的原则。从一般的修订版开始，然后逐步介绍跟[**Android**](javascript:;)平台相关的一些概念和技术。

　　本章会有很多代码，所以，准备好边看边写代码，用这种方式来领略文中例子的含义，是最快的[**学习**](javascript:;)方式。

　　本章的主要内容有。

　　"  介绍和解释了TDD测试驱动开发的含义。

　　"  分析了TDD的优势。

　　"  介绍了一个真实的工程的生命周期。

　　"  通过写**[测试用例](javascript:;" \t "_self)**来理解需求。

　　"  项目用TDD的方式来进化。

　　"  完成一个工程，这个工程要完全符合最初的需求。

**4.1  TDD测试驱动开发入门**

　　简单地说，测试驱动开发简称TDD，是一种测试策略。这种策略要求随着开发过程来跟进测试用例。这些测试用例在代码开发之前就准备好了，开发人员开发的代码必须能够通过这些测试用例。

　　添加一个用例，然后工程代码必须满足能够编译、执行这个用例，并且结果要跟用例中的结果集合一致。

　　这种测试策略跟其他策略相比，不同点在于其他的测试方法都是在代码写完之后再写测试用例，但是TDD是在代码写完之前写测试用例。

　　在代码开发之前写完测试用例有以下几个优势：第一，测试用例的编写方式很多，如果留到最后开发完再来写，测试人员也很可能不写测试用例了；第二，开发人员对自己开发代码的质量也更加负责了。

　　开发设计一步步定下来之后，如果开发的代码不能通过测试用例，那就需要重构改进了。

　　图4.1帮助我们理解测试驱动开发的过程：

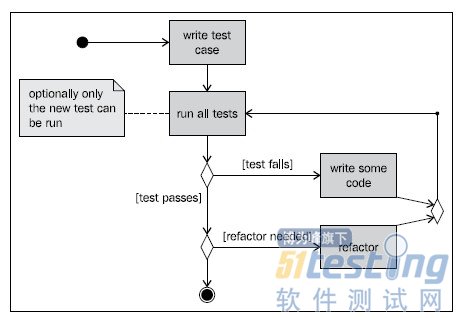
[](http://www.51testing.com/batch.download.php?aid=58414)

图4.1  UML活动图

**Android测试环境—Android应用测试指南(5)**

**第5章  Android测试环境**

　　为了验证温度转换器的基本功能和行为交互，我们在第4章节已经新建了测试应用，并添加了不少**[测试用例](javascript:;" \t "_self)**。下面，我们在不同的条件下执行这些[**自动化测试**](javascript:;)用例，并且手工执行一些测试用例，站在客户的角度，体验真实使用的感觉。

　　本章节中，我们会涵盖以下内容。

　　"  新建AVD Android虚拟器，通过设置不同的配置，为应用程序营造不同的环境条件。

　　"  重点理解，每个AVD中配置对应具体的含义。

　　"  如何启动执行AVD Android虚拟器。

　　"  如何将AVD跟外面的窗口剥离，创建一个Headless模拟器。

　　"  锁住屏幕，执行所有的自动化测试用例。

　　"  模拟真实的网络条件。

　　"  用monkey来给应用程序发送操作事件。

**5.1  新建[Android](javascript:;" \t "_self)虚拟设备**

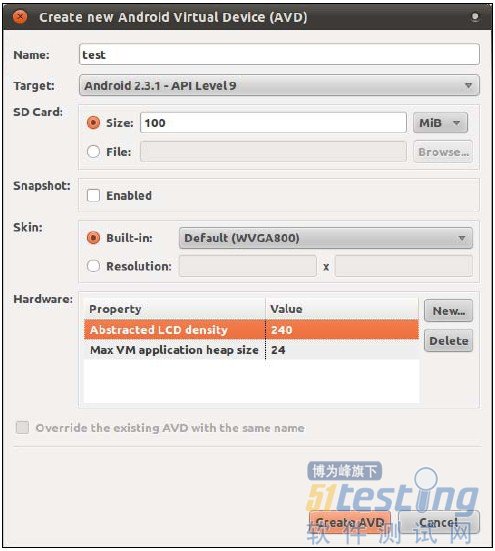
　　发现跟设备相关的问题的最佳时机，是用虚拟机设置、模拟尽可能多的[**操作系统**](javascript:;)和设备来执行测试用例的时候。

　　然而，最终的验收测试应该在日益更新的各式**[手机](javascript:;" \t "_self)**设备上执行，当然，这是不可能的，机型、操作系统这么多，不可能在每种机器上都测试一遍所有的用例。当然，还可以在[**移动**](javascript:;)云测试平台上挑选很多设备进行测试，不过，这种方式一般都会超过研发预算。幸运的是，Android平台可以通过在模拟器和AVD中不同的设置达到模拟各种各样设备的效果。

　　这章的所有例子都是在一个Ubuntu10.04（Lucid Lynx）64bit上执行的，用Android SDK和AVD管理器10，Android SDK是版本2.3的，内带API 9。

　　新建一个AVD，我们可以通过命令行andriod，或者在eclipse界面中单击Windows-Andriod SDK and AVD，弹出来一个浮层界面，单击New按钮就可以新建一个AVD了，操作界面如图5.1所示。

　　图5.1  操作界面

[](http://www.51testing.com/batch.download.php?aid=58446)

LCP density = 240 pixels-per-inch

**用命令行来启动虚拟设备—Android应用测试指南(6)**

**5.2  用命令行来启动虚拟设备**

　　如果可以通过命令行来实现启动不同类型的虚拟设备那该多好啊，这样，我们就可以通过写脚本来自动化回归。

　　释放窗口中的虚拟设备，为我们自动化脚本打开了另一扇大门。

　　下面，让我们来看看门后面的世界。

**5.2.1  Headless模拟器**

　　无头模拟器，就是不显示交互窗口的模拟器。这种模拟器在[**自动化测试**](javascript:;)的时候很有用，因为我们在自动化测试的时候，没有人会盯着屏幕看，以及人机交互，因为自动化执行得都很快，想看也看不了。

　　当然，值得一提的是，如果你看不到屏幕，即使用例失败了，你可能还不知道原因，因此，我们在选择的时候，要根据具体场景需要，选择是否需要这种模拟器。

　　我们注意到一件事情，就是在运行AVD虚拟机的时候，通信端口在执行期间就已经分配好了，端口从5554开始，将最新实用的端口加2，就是当前分配的端口号。这个端口号可以用来标记模拟器，给模拟器命名，设置模拟器序列号，比如，模拟器用端口5554，那它的名字就是"模拟器-5554"。在我们用虚拟机模拟器的时候很有用，这样我们就不用关注分配的端口号是啥了，直接看名字就知道了。当然，在你同时启动多个模拟器，跑自动化程序的时候，就会造成一些困扰，较难跟踪了。

　　这种情况下，我们建议设置指定的端口号来通信，使得虚拟器在我们的掌控之中。

　　通常，如果我们在同一个时间点启用多个虚拟器来跑多个用例，除了不想看屏幕之外，还希望有输出。我们有下面这些选择项。

　　1．启动刚刚创建的虚拟机命令行如下：

　　$ emulator -avd test -no-window -no-audio -no-boot-anim -port 5580 &

　　2．端口必须是5554和5584之间的整数：

　　$ adb devices

　　List of devices attached

　　Emulator-5580 device

　　这样列出所有的设备列表。

　　3．下一步就是安装应用和测试工程：

|  |
| --- |
| $ adb -s emulator-5580 install\  TemperatureConverter/bin/TemperatureConverter.apk  347 KB/s (16632 bytes in 0.046s)  pkg: /data/local/tmp/TemperatureConverter.apk  Success  $ adb -s emulator-5580 install\  TemperatureConverterTest/bin/TemperatureConverterTest.apk  222 KB/s (16632 bytes in 0.072s)  pkg: /data/local/tmp/TemperatureConverterTest.apk  Success |

　　4．然后，我们可以用具体的序列号来标记正在执行的**[测试用例](javascript:;" \t "_self)**：

|  |
| --- |
| $adb-semulator-5580shellaminstrument-w\  com.example.aatg.tc.test/android.test.InstrumentationTestRunner  com.example.aatg.tc.test.EditNumberTests:......  com.example.aatg.tc.test.  TemperatureConverterActivityTests:..........  com.example.aatg.tc.test.TemperatureConverterTests:....  TestresultsforInstrumentationTestRunner=....................  Time:25.295  OK(20tests) |

**5.2.2  禁用锁屏功能**

　　我们可以看到，这种情况下，测试用例执行起来，并没有中断让用户去输入什么东西。不过，有时候，如果我们用标准的方法，在Eclipse中启动标准的模拟器来执行测试用例，即使用例没有失败，也会报错。那是因为模拟器在第一个屏幕出现的时候就锁住了，我们需要将UI相关的屏幕解锁来执行用例。

　　解锁屏幕你可以用下面的命令：

　　$ adb -s emulator-5580 emu event send EV\_KEY:KEY\_MENU:1 EV\_KEY:KEY\_MENU:0

　　还有一种方式，就是在程序里面将锁屏禁用；但是呢，有点不好的就是需要在应用程序中插入测试的代码。因此，程序发布之前，需要将这段测试代码从应用程序中删除掉。

　　在此之前，需要在AndriodManifest.xml中加入下面这段准入代码，然后再在你的程序中加入禁止锁屏的代码。

　　Manifest配置加入下面元素，如框5.1所示。

**框5.1  manifest配置**

　　<manifest>

　　...

　　<uses-permission [**android**](javascript:;):name="android.permission.DISABLE\_KEYGUARD"/>

　　...

　　</manifest>

**测试方案—Android应用测试指南(7)**

**第7章  测 试 方 案**

　　本章将列出测试过程中经常遇到的问题，通过比较接近实际的例子，将前面章节中讲到的原则、技术应用到里面去。我们将会用处方的形式展现给大家看，方便大家根据自己项目的需要选择性**[学习](javascript:;" \t "_self)**。

**本章包含下面几个主题：**

　　"  Android单元测试；

　　"  测试行为和应用程序；

　　"  测试**[数据库](javascript:;" \t "_self)**和ContentProviders的测试；

　　"  测试本地和远程服务；

　　"  测试UI交互；

　　"  测试异常；

　　"  测试转换；

　　"  测试内存泄露。

　　通过这章学习，你会对测试自己的工程需要干什么，用到什么有一定的了解。

**7.1  Android单元测试**

　　我们知道，对工程中的一些组件的[**测试用例**](javascript:;)，真的需要在一个独立的，跟底层系统隔离开的环境中才好执行。在执行这些用例的时候，我们要选择一个高权限高层次的类，要可以除去被测模块对其他模块的依赖，但是又不会影响一些基础架构。

　　对于这个类，我们有个候选类，那就是AndriodTestCase。样例详情可以在Andriod CTS测试集合中下载，如框7.1所示。

　　框7.1

|  |
| --- |
| /\*  \* Copyright (C) 2009 The [**Android**](javascript:;) Open Source Project  \*  \* Licensed under the Apache License, Version 2.0 (the "License");  \* you may not use this file except in compliance with the License.  \* You may obtain a copy of the License at  \*  \* http://www.apache.org/licenses/LICENSE-2.0  \*  \* Unless required by applicable law or agreed to in writing,  \* software distributed under the License is distributed on an  \* "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,  \* either express or implied.  \* See the License for the specific language governing permissions  \* and limitations under the License.  \*/  package com.android.cts.appaccessdata;  import **[java](javascript:;" \t "_self)**.io.FileInputStream;  import java.io.FileNotFoundException;  import java.io.IOException;  import android.test.AndroidTestCase;  /\*\*  \* Test that another app's private data cannot be accessed.  \*  \* Assumes that {@link APP\_WITH\_DATA\_PKG} has already created  the private data.  \*/  public class AccessPrivateDataTest extends AndroidTestCase {  /\*\*  \* The Android package name of the application that owns  the private data  \*/  private static final String APP\_WITH\_DATA\_PKG =  "com.android.cts.appwithdata"; |

**移动App测试的22条军规**

发布时间: 2015-7-20 09:56    作者: 黄勇    来源: 51Testing软件测试网原创

|  |  |
| --- | --- |
| [http://www.51testing.com/attachments/2016/02/15047838_2016022416351017UVj.jpg](http://www.51testing.com/batch.download.php?aid=57962) | **本书概述：**全面讲解了移动App测试必知必会的技术、工具和**[测试](javascript:;" \t "_self)**用例移动App测试中的测试方法和案例实践，以及提高效率的技术和技巧，通过22条来自于测试实战中的最佳实践，展示了移动App测试中心思想，图文并茂，帮助读者更快速掌握移动App测试全流程。 |
| **作 者：**黄勇 |
| **出版社：**人民邮电出版社 |
| **出版日期：**2015 年7月 |
| **内容简介** | |
| 本书全面讲解了移动App测试的技术、技巧、工具、案例和**[测试用例](javascript:;" \t "_self)**，全书共分23章，主要内容为：移动App的特性，关注多任务和意外情况处理，避免手势冲突，关注用户体验，设计通知和消息展示，支持**[操作系统](javascript:;" \t "_self)**特性，及时显示和同步消息，支持多种文件格式，支持多语言和地区设置，重点测试高内存占用的功能、降低流量和电量消耗，确保成功集成和调用第三方App，尽量不使用非标准控件，iOS 8升级所引入的新特性，**[Android](javascript:;" \t "_self)** 5.0升级所引入的新特性，自动化和探索性测试，**[自动化测试](javascript:;" \t "_self)**中模拟器的使用，用户界面自动化测试的常见工具，性能和安全性测试，使用Log定位问题，充分使用持续集成、持续部署，以及微信App测试综合案例分析等核心技术。本书适合软件的测试初学者、测试从业人员及程序员阅读，也可以作为大专院校相关专业师生的**[学习](javascript:;" \t "_self)**用书，以及培训学校的教材。 | |
| **作者简介** | |
| 黄勇，现任ThoughtWorks中国区QA Lead。先后任职于博彦科技，普华永道GTS和ThoughtWorks；作为资深质量分析师，先后**[工作](javascript:;" \t "_self)**在[**Oracle**](javascript:;)，**[微软](javascript:;" \t "_self)**，普华永道，澳洲电信等多家公司的大型项目上。 | |
| **本书详细介绍** | |
| [自序 目录](http://www.51testing.com/html/07/n-3578207-2.html) [书评 独家连载](http://www.51testing.com/html/07/n-3578207.html) [军规1  确定设备和平台再动手 1.1  移动App的特性](http://www.51testing.com/html/37/n-3578237.html) [1.2  移动App的生命周期载](http://www.51testing.com/html/37/n-3578237-2.html) [1.3  设备的硬件参数](http://www.51testing.com/html/74/n-3578274.html) [军规3  关注多任务和意外情况处理 3.1  第一个场景](http://www.51testing.com/html/19/n-3578319.html) [3.2  第二个场景](http://www.51testing.com/html/19/n-3578319-2.html) [3.3  需注意的场景](http://www.51testing.com/html/41/n-3578341.html) [军规5  关注用户体验](http://www.51testing.com/html/41/n-3578341-2.html) [5.1  横竖屏幕测试](http://www.51testing.com/html/84/n-3578384.html) [5.2  WebView的测试](http://www.51testing.com/html/84/n-3578384-2.html) [5.3  规范与习惯 军规8  及时显示和同步消息](http://www.51testing.com/html/23/n-3578423.html) [军规16  尽量不使用非标准控件](http://www.51testing.com/html/69/n-3578469.html) [军规18  尽量减少依赖试 18.1  对于既有Web版本又有App版本的App要减少依赖](http://www.51testing.com/html/06/n-3578506.html) [18.2  没有Web版本的App也需要考虑App的依赖](http://www.51testing.com/html/06/n-3578506-2.html) | |

**确定设备和平台再动手-移动App测试的22条军规(1)**

发布时间: 2015-7-21 11:12    作者: 黄勇    来源: 51Testing软件测试网原创

**军规1  确定设备和平台再动手**

　　在测试设计之初，测试人员首先会考虑的是什么呢？没错，就是测试的环境，也就是确定App究竟需要运行在什么样的设备和平台上。

　　显然，在[**移动**](javascript:;)设备和平台碎片化的现实中，测试人员穷尽所有设备和[**操作系统**](javascript:;)的版本来实现全覆盖的测试是不可能的。那如何在有限的时间和精力投入下，从投入产出比的角度出发，达到尽可能多的测试覆盖呢？这里主要考虑以下几个方面。

**1.1  移动App的特性**

　　（1）如果App是针对心率监测、指纹识别、近场通信（NFC）、红外线操控这些需要特殊传感器设计的，那对测试设备和平台的选择就相对少一些，只需要考虑那些拥有这些传感器的设备。例如对于支持指纹识别的App，测试人员需要考虑的设备也就是iPhone 5s、iPhone 6、iPhone 6Plus、iPad Air2、iPad mini3、LG G3、三星Galaxy S5、三星Galaxy Note4、HTC One Max和华为Mate7这些设备（不考虑市场占有率比较低的vivo和OPPO的**[手机](javascript:;" \t "_self)**）；而如果App支持心率监测，测试人员就只能选择三星Galaxy S5和Galaxy Note4了。

　　这里推荐大家使用一个网站（http://www.phonearena.com）来做设备的查找**[工作](javascript:;" \t "_self)**。通过这个网站不仅可以查询到各种手机和平板设备的详细参数信息，还可以对它们进行横向对比，方便测试人员找到适合用来做测试的设备（如图1.1所示）。

　　（2）如果App是针对某种平台所独有的功能设计的，或者是某种平台独占的，测试人员就只需要考虑相应平台下的设备。比如App是类似**[Android](javascript:;" \t "_self)**设备上层出不穷的"xx清理大师"，那在确定测试设备和平台时就不需要考虑iOS平台了；又比如之前Instagram选择只支持iOS平台，那作为Instagram的测试人员只需要关注于iOS设备就足够了。

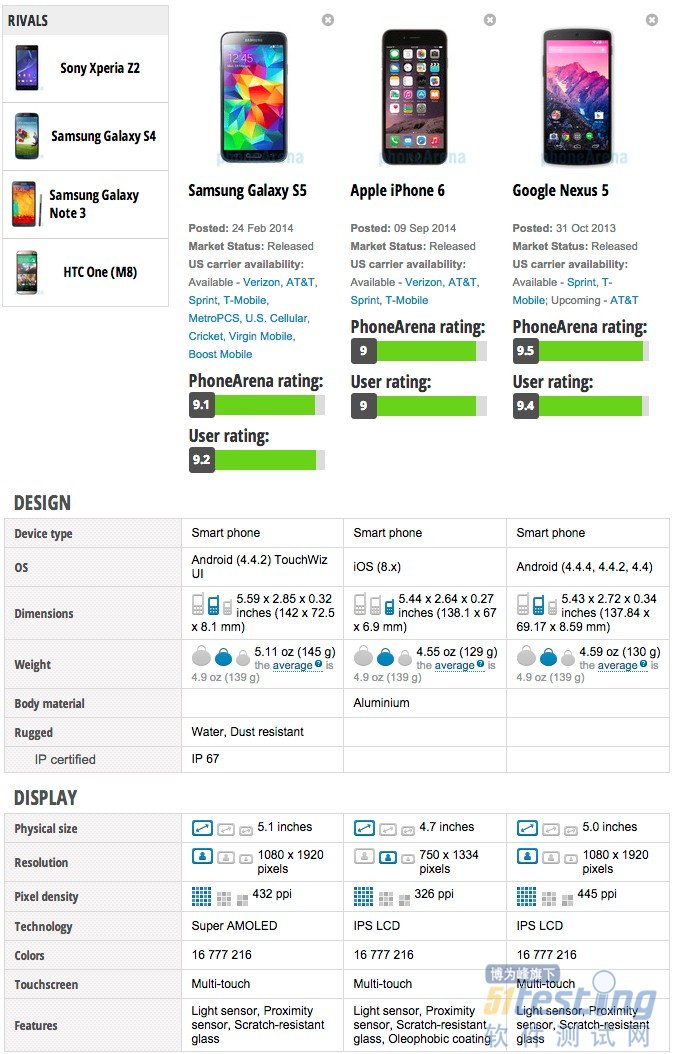
[](http://www.51testing.com/batch.download.php?aid=54118)

图1.1  http://www.phonearena.com设备对比

　　或者，如果App选择不支持某种平台，相应的，测试人员也就不需要测试运行这些平台的设备了。比如WindowsPhone、黑莓（BlackBerry）以及塞班（Symbian）平台在市场上的占有率已经很低了（根据2014年第四季度的调查，详见图1.2），如果在开发时选择不支持这些平台，那在测试时测试人员就完全可以忽略相关的设备。

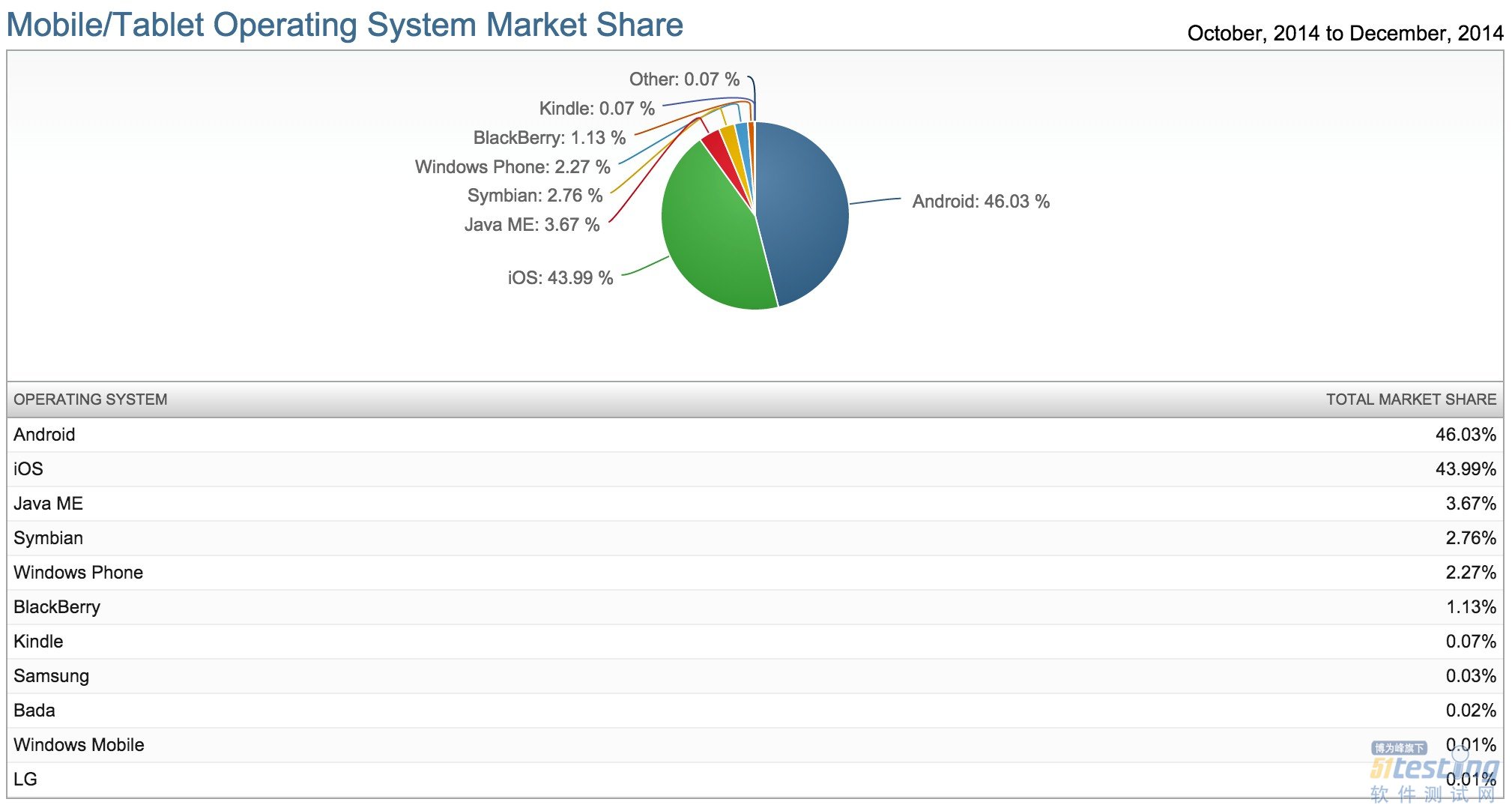
[](http://www.51testing.com/batch.download.php?aid=54119)

图1.2  2014年第四季度各操作系统市场占有率调查表（数据来源：www.netmarketshare.com）

　　（3）如果App是面向大众的通用型App，测试人员就需要结合移动App的生命周期和测试设备的硬件参数来确定测试设备和平台了

**设备的硬件参数-移动App测试的22条军规(2)**

发布时间: 2015-7-22 10:15    作者: 黄勇    来源: 51Testing软件测试网原创

字体:  [小](javascript:doZoom('12');)  [中](javascript:doZoom('14');)  [大](javascript:doZoom('16');)  | [上一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=up&itemid=229485&catid=96) [下一篇](http://www.51testing.com/batch.common.php?action=viewnews&op=down&itemid=229485&catid=96) | [打印](javascript:doPrint();)  | [我要写文](http://bbs.51testing.com/thread-77515-1-1.html)  | 推荐标签： [app测试](http://www.51testing.com/?action-tag-tagname-app%B2%E2%CA%D4) [软件测试技术](http://www.51testing.com/?action-tag-tagname-%C8%ED%BC%FE%B2%E2%CA%D4%BC%BC%CA%F5) [手机测试](http://www.51testing.com/?action-tag-tagname-%CA%D6%BB%FA%B2%E2%CA%D4) [测试方法](http://www.51testing.com/?action-tag-tagname-%B2%E2%CA%D4%B7%BD%B7%A8)

　　（51Testing软件**[测试](javascript:;" \t "_self)**网获人民邮电出版社和作者授权连载本书部分章节。任何个人或单位未获得明确的书面许可，不得对本文内容复制、转载或进行镜像，否则将追究法律责任。）

**1.3  设备的硬件参数**

　　（1）屏幕尺寸。现在**[手机](javascript:;" \t "_self)**越出越大，连坚持自己风格的[**苹果**](javascript:;)公司也开始跟风发布大屏手机了。屏幕大小除了会影响显示效果外，还会影响到用户的使用习惯。一般用户手持6英寸屏幕的设备时，会采取双手操作的方式，所以App如果同时支持横纵屏显示会带来更好的用户体验（如图1.7所示）。

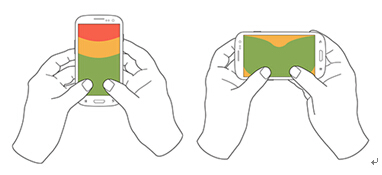
[](http://www.51testing.com/batch.download.php?aid=54151)

图1.7  双手持握设备的方式

　　而对于4～5英寸这种可以单手持握的设备，如果App无论横纵向显示，按钮都最好不要放在屏幕四个角，以免用户很难点击（如图1.8所示）。

[](http://www.51testing.com/batch.download.php?aid=54152)

图1.8  单手操作范围

　　49%的单手操作用户采用的是以上两种姿势（左手用户相反）。绿色代表容易点击区域，黄色为拇指伸展可点击区域，红色区域为超出单手可点击范围。

　　（2）分辨率。分辨率的大小会决定显示内容的多少，这对显示图片和视频时会有一定的影响（如图1.9所示）。

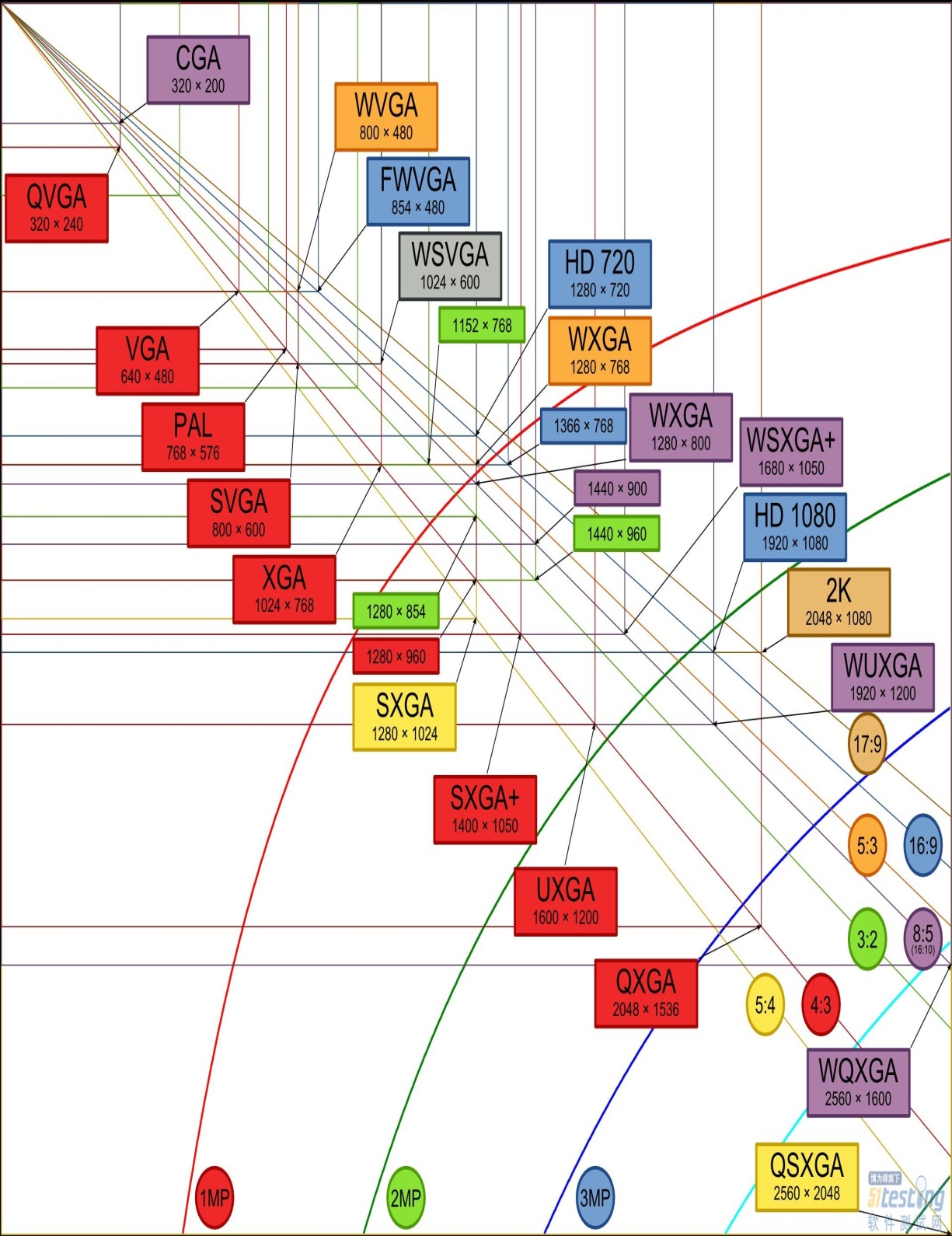
[](http://www.51testing.com/batch.download.php?aid=54153)

图1.9  不同分辨率下显示内容的大小以及显示比例

**关注多任务和意外情况处理-移动App测试的22条军规(3)**

发布时间: 2015-7-23 13:00    作者: 黄勇    来源: 51Testing软件测试网原创

**军规3  关注多任务和意外情况处理**

　　想必我们都有过这样的体验：在购物的App中填写信息，比如说收货地址的时候，忘记了具体地址，然后切换出该App到"印象笔记"之类的记录App中查找到地址，复制下来，再切换回购物App的时候发现，刚才填写的好多信息都没有了，还得手动输入一遍，这样就会觉得App的功能和体验很差。

　　这种情况其实就是没有处理好多任务时App的表现。

　　不同于功能机的时代，在使用**[智能手机](javascript:;" \t "_self)**的时候，经常会同时运行多个程序（如图3.1所示），这就要求测试人员在设计和测试App的时候考虑到App被别的程序或者用户切换到后台时，需要进行什么操作。

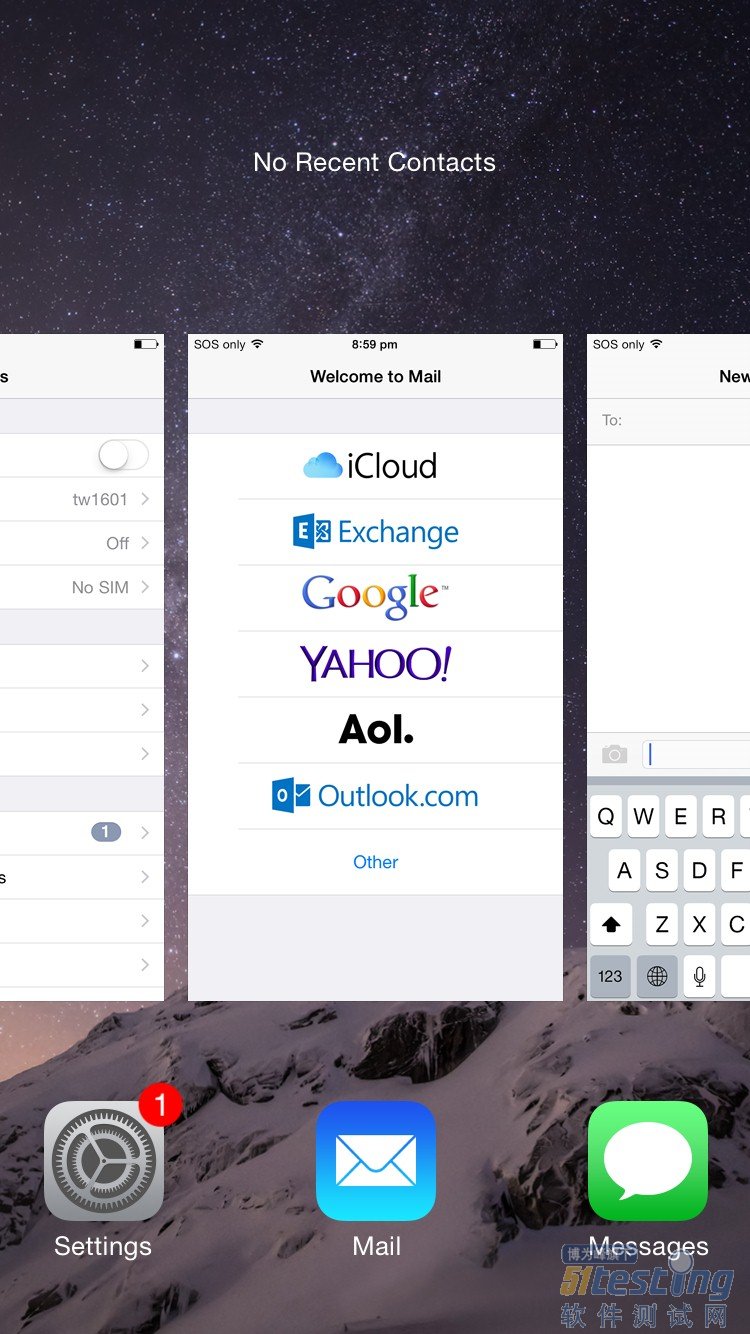
[](http://www.51testing.com/batch.download.php?aid=54175)

图3.1  iOS的多任务处理

**3.1  第一个场景**

　　一个典型的场景就是，App在使用过程中用户接听一个来电，App应该如何处理（如图3.2所示）。

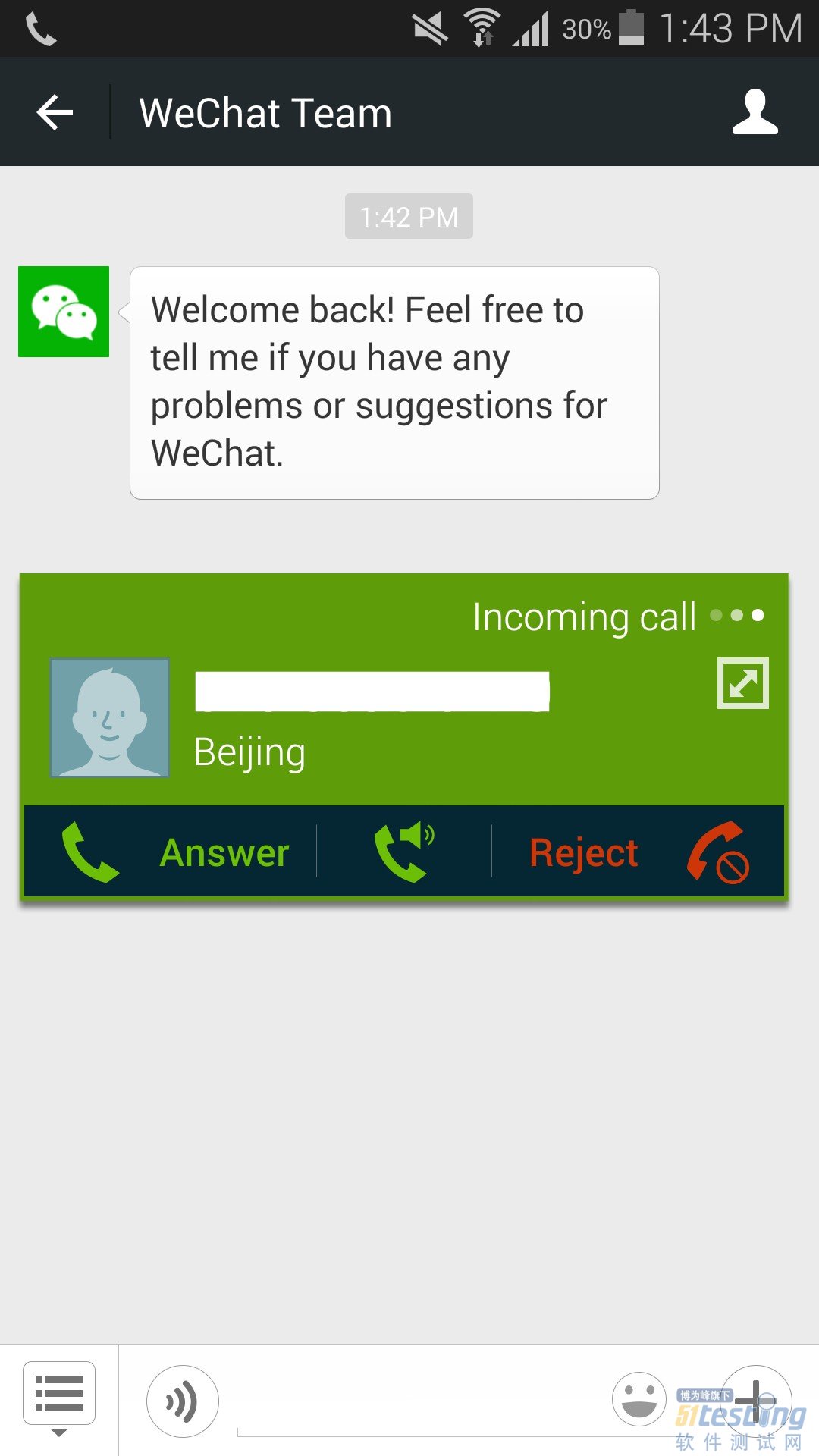
[](http://www.51testing.com/batch.download.php?aid=54176)

图3.2  使用App时接收到来电

　　App是否需要在后台运行？是否需要在状态栏和通知栏显示信息？当用户挂机后，App是否需要恢复之前的状态，还是需要重新刷新？

　　不同的App需要有不同的处理，比如说用户在接听**[电话](javascript:;" \t "_self)**前正在使用微信编辑消息，当挂断电话后，用户自然希望能继续编辑，并且刚才填写的消息内容都还在；而如果用户刚才打开的是一个计时器，用户自然希望得到App一直运行的时间；而对于音乐或视频播放类App，在接听电话前已经暂停播放，在挂断之后，用户也希望保证音乐或视频还是处在暂停状态，或者反之。

**需注意的场景-移动App测试的22条军规(4)**

**3.3  需注意的场景**

　　有种场景需要单独注意：对于在具备同样功能的App，尤其是具有视频和音频播放功能的App之间进行切换时，需要注意它们之间的播放控制是否会对另外的App产生影响。例如我们正使用QQ音乐播放着歌曲，这时切换到酷狗音乐，酷狗音乐里的歌曲是否会自动播放呢？要是暂停了酷狗音乐的音乐播放，再回到QQ音乐里，这时QQ音乐是否会继续播放音乐呢？通常的做法是，App的操作只对本App有效，所以QQ音乐不应该影响到酷狗音乐，也不应该被酷狗音乐所影响。

　　同时，App被切换回当前应用时是否刷新，也会因App后台数据是否有可能改变而有所不同。比如在使用具有**[记录](javascript:;" \t "_self)**通话和网络流量功能的App时，用户拨打/接听**[电话](javascript:;" \t "_self)**，或者在使用别的App之后切换回来，因为用户关心的通话和流量的使用量很可能发生变化，所以App必须要刷新显示。微信这种通信类的App也是如此，用户切换出App的时候，无法避免App中数据的更新。运动和健康监测类的App更是如此，无论App是否被切换到后台，改变都需要随时被记录，如图3.5所示。

[](http://www.51testing.com/batch.download.php?aid=54188)

图3.5  iOS 8上的Health会随时记录运动和健康监测数据

　　对于App切换或停止，还需要考虑的是，切换App究竟是需要用户打开多任务处理界面，选择App才能恢复App的运行（这时直接点击桌面App图标意味着重新运行App），还是允许用户通过点击桌面图标来恢复App运行状态。一般我们都会选择后者，但是也有少部分软件出于设计的考虑而采用的是前者。

**横竖屏幕测试-移动App测试的22条军规(5)**

**5.1  横竖屏幕测试**

　　在[**移动**](javascript:;)设备上做用户体验测试，最容易想到的就是对App做横竖屏幕的测试，来观察App的显示效果。

　　首先需要被测试的App支持横竖屏。如果App不支持行不行呢？其实也是可以的，但是随着大屏幕**[手机](javascript:;" \t "_self)**的流行，连保守的iPhone都发布了5.5英寸屏幕的iPhone 6Plus，可见大屏幕手机是很有吸引力的。用户在操作大屏幕手机的时候，通常选用的都是横屏来使用App的。所以大家就尽量确保App支持横屏操作吧（如图5.3所示）。

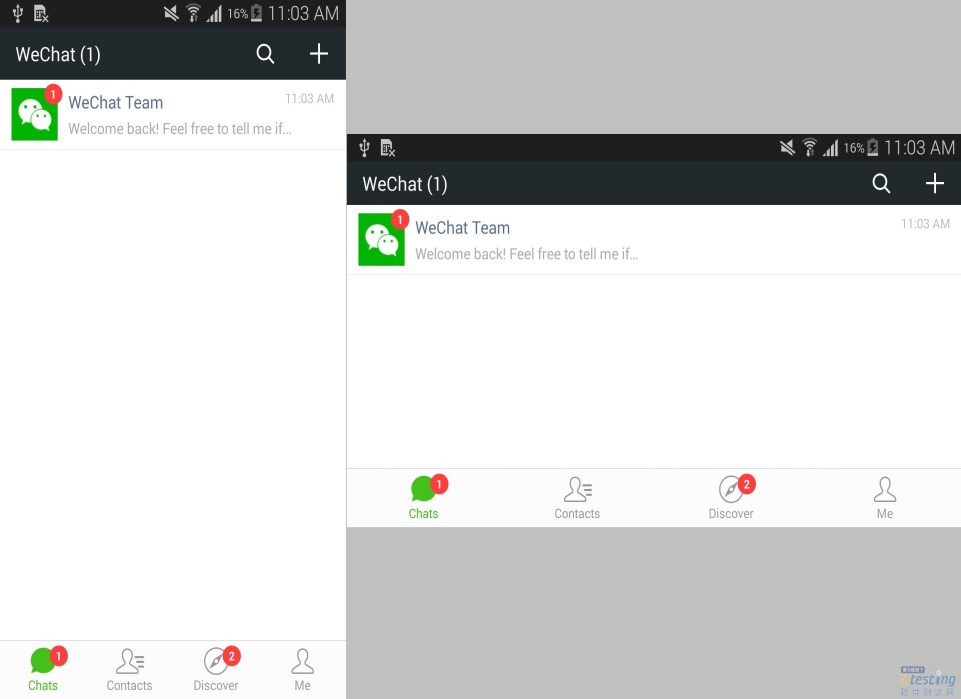
[](http://www.51testing.com/batch.download.php?aid=54233)

图5.3  大量的App都已经支持了横竖屏的显示

　　其次，要解决横竖屏切换的问题。别看这是个很简单的功能，貌似只要在代码中设置支持横竖屏显示就可以避免横竖屏切换出现问题。但实际上，在某些情况下App代码有可能破坏了屏幕旋转的功能，比如说在App中的某些页面限制了屏幕显示的方向（如图5.4所示）。

[](http://www.51testing.com/batch.download.php?aid=54234)

图5.4  在图片上我们可以看见App展示内容的方向和设备横屏的方向不一致

　　除此之外，还需要注意在App中嵌入了WebView的页面的显示。在支持横竖屏切换的App中的页面嵌入了WebView，当WebView读取完成时，有可能横竖屏切换功能就被破坏了（如图5.5所示，游戏中这种弹出页面就破坏了横竖屏的切换）。

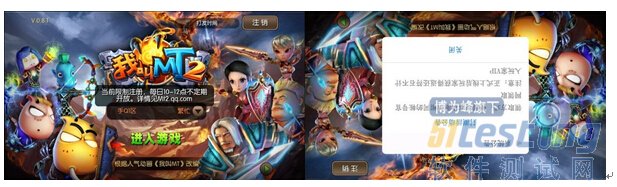
[](http://www.51testing.com/batch.download.php?aid=54235)

图5.5  当把移动设备向右横屏（设备底部在左手边），然后打开App，App本身显示正常。但是当有WebView弹出的时候，页面会反向；而当设备向左横屏的时候，却没有这个问题

　　值得一提的是，如果App支持显示图表，测试人员更需要关注图表在横竖屏之间的切换，因为横竖屏的显示宽度不一样，图表在不同屏幕状态下，显示的内容和样式很可能也是不一样的（如图5.6所示）。

　　在测试中，测试人员最好对每个页面都进行横竖屏显示的测试。当然，要更加关注于嵌入WebView和其他弹出式的控件的页面，以及图表这类可能因屏幕宽度和高度不同而改变显示内容和效果的页面。

[](http://www.51testing.com/batch.download.php?aid=54236)

图5.6  App中同一个图表在横竖屏下显示的内容和样式都有可能是不一样的

　　对于可以设置横竖屏显示的App，一旦设置了横屏或者竖屏，从启动App开始到关闭App为止，用户所有操作的页面都应该以设置的屏幕显示方向显示，不能出现有些页面横屏，有些页面竖屏这种混杂显示的问题。

**及时显示和同步消息-移动App测试的22条军规(6)**

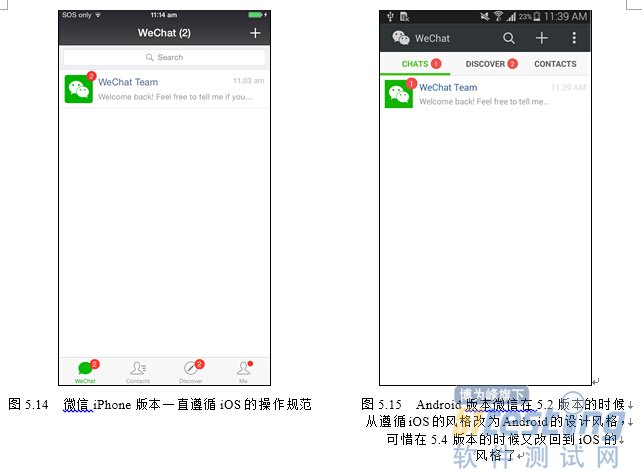
**5.3  规范与习惯**

　　对于支持多个[**操作系统**](javascript:;)平台的移动App，也需要在不同的操作系统上，遵循当前操作系统的设计规范和使用习惯，而不要一味地为了自己各个App的一致性而破坏操作系统的设计规范和使用习惯。

　　iOS的设计规范要求把菜单放置在设备底端，在**[记录](javascript:;" \t "_self)**上从右向左滑动会呼出"删除"和"更多"菜单等（如图5.14所示）。

[**Android**](javascript:;)的设计规范则要求把多于3个的菜单放置在右上角3个点的按钮中，而长按记录则可以呼出更多的操作选项等（如图5.15所示）。

　　不同的操作系统有不同的特性，因此也有自己独特的设计和使用习惯，测试人员在开发和测试移动App的时候，都需要尽可能遵循这些规范，减少用户的**[学习](javascript:;" \t "_self)**成本，提高使用App的便利性。

[](http://www.51testing.com/batch.download.php?aid=54263)

**军规8  及时显示和同步消息**

　　试想一下，如果用户使用的是我们的即时通信App，某天登录突然收到了朋友在几个月之前发来的一条信息，那用户会有什么样的反应呢？如果是笔者的话会毫不犹豫地删除这个App，而且还会劝说身边的朋友不去使用这个App。

　　如果我们的移动App是金融类的App，银行发布的信息或者股市、汇市的信息差一秒钟显示给用户，可能都会让用户蒙受巨大的损失（如图8.1所示），比如说在银行App后台升级的当天才提示用户服务不可用。

[](http://www.51testing.com/batch.download.php?aid=54264)

　　所以对于消息显示，测试人员在测试的过程中不能只是简单地验证消息在各种情况下是否能正常的显示，还需要考虑到App中各种缓存对于消息显示的影响。

**一般在以下的环境中，我们才会在App中使用缓存机制。**

　　（1）App需要提供网络服务。

　　（2）App中有至少一部分数据不需要实时更新，哪怕是很短时间（3～5分钟）的缓存也是有好处的。

　　（3）App可以针对某种数据设定固定的过期时间，而不会导致其功能和用户体验出现问题。

**App使用缓存机制有哪些好处呢？**

　　（1）对用户来说可以减少流量的消耗。

　　（2）减少由于网速过慢而导致用户长时间等待的状况，App的响应速度变快了。

　　（3）App因从网络加载数据而出错的情况会大大减少，提高了App的稳定性。

　　（4）App在一定程度上可以算是支持了离线浏览。

　　（5）对App开发和维护团队来说，则可以减轻App请求对服务器进行大量访问的压力。

　　我们在设计App的过程中一般都会采取"内存"→"文件"→"网络"的结构来设计App的缓存机制。因此，测试人员在测试App的时候也可根据这个特点来设计测试场景。

　　（1）对于App采用内存来缓存数据的方式，一般来说，测试人员不用特定去测试，因为iOS和Android操作系统会处理App使用内存来缓存数据的请求。但是，对于浏览器这样的App，测试人员可以通过同时打开多个页面，访问同样的网址，来测试是否App对数据在内存中进行了缓存（如图8.2所示）。

[](http://www.51testing.com/batch.download.php?aid=54265)

　　（2）对于App采用文件来缓存数据的方式，由于iOS并不支持用户手动清除App的文件缓存，所以这一部分在iOS App测试中可以不涉及；但是对于Android操作系统，可以通过在App的应用程序信息中清除App缓存来测试App是否对于数据文件进行了正确的缓存（如图8.3所示）。

[](http://www.51testing.com/batch.download.php?aid=54266)

**尽量不使用非标准控件-移动App测试的22条军规(7)**

**军规16  尽量不使用非标准控件**

　　在特定的[**操作系统**](javascript:;)版本上，如果想要App支持某种特性，可是操作系统并没有提供这种特性，开发人员很可能会自己设计控件，或者使用第三方库来实现这个功能。

　　但是这么做却会带来一些问题。接下来以App可以切换主题（Theme）这个特性为例（如图16.1所示）具体说明一下。

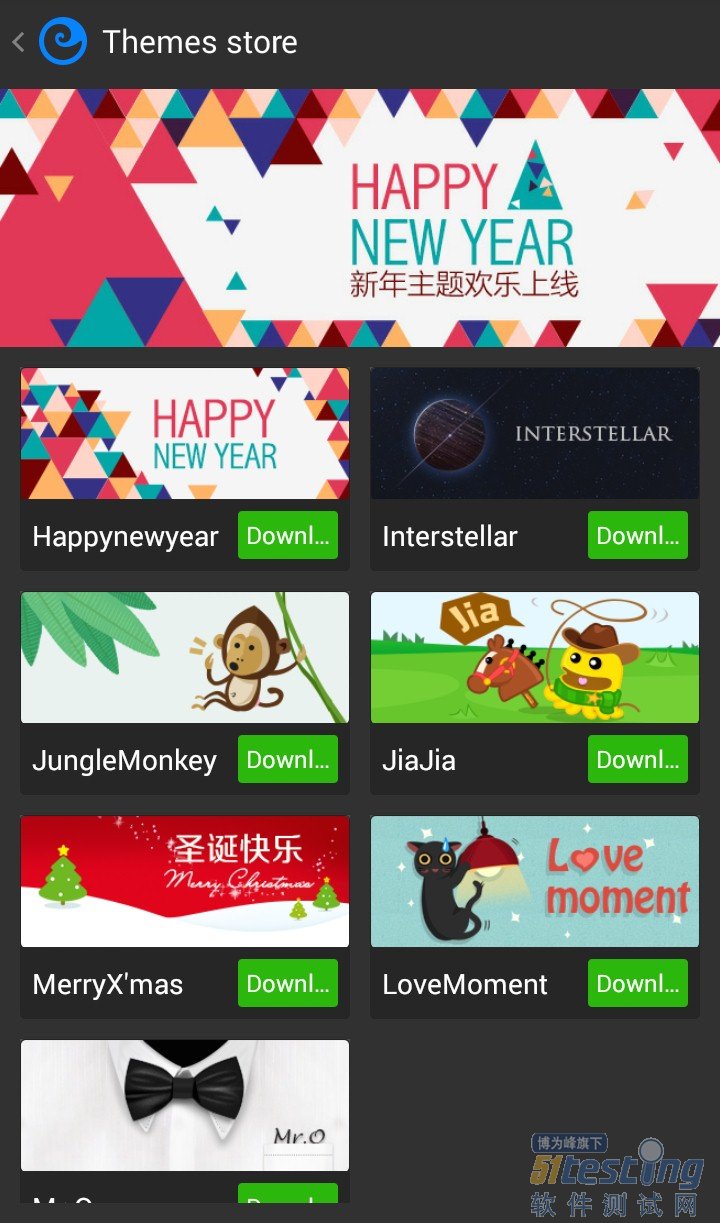
[](http://www.51testing.com/batch.download.php?aid=54308)

图16.1  不少App都支持在App中切换主题的功能

　　在iOS 4中App如果想提供用户切换App主题颜色的功能，就需要App自己实现相应的代码，因为iOS 4.x操作系统并不提供相应的功能。而当iOS操作系统升级到iOS 5的时候，开发者可以使用iOS 5新增的UIApperance协议定义一些可以自定义外观的控件。到了iOS 7时，开发者可以只使用着色（tintColor）属性来设置控件的颜色，以实现App主题颜色的切换（如图16.2所示）。

[](http://www.51testing.com/batch.download.php?aid=54309)

图16.2  对于如何实现在App中切换主题颜色，不同iOS版本实现的方式也不相同

　　通过这个例子可以看到，如果App在iOS 4就实现了主题颜色切换的功能，也就是App自己实现了对应的代码，当iOS操作系统升级到iOS 5之后，实现的方式改变了，如果需要把主题颜色切换功能的实现代码切换到iOS 5上使用UIApperance的方式，就需要对于所有涉及到这些代码的功能进行回归测试。

　　如果不使用iOS 5原生代码支持的实现方式，还是保持之前代码的实现方式，当iOS操作系统升级到iOS 7时，就会发现新版iOS原生代码提供的实现方式大大方便了App的开发、维护和测试**[工作](javascript:;" \t "_self)**。如果还按照以前的方式来实现代码，随着iOS操作系统的升级，这些代码会越来越臃肿和复杂。这是因为后来添加的很多功能代码都是基于这些代码添加的，而之前的这些代码都成为了App代码的基础设施，随着时间的推移，想要替换这些基础设施的开发和测试的代价会越来越大，而不替换这些基础设施，开发和测试的代价会更大。

　　同样还有一个例子，是关于iOS操作系统上的小工具Widget的（如图16.3所示）。

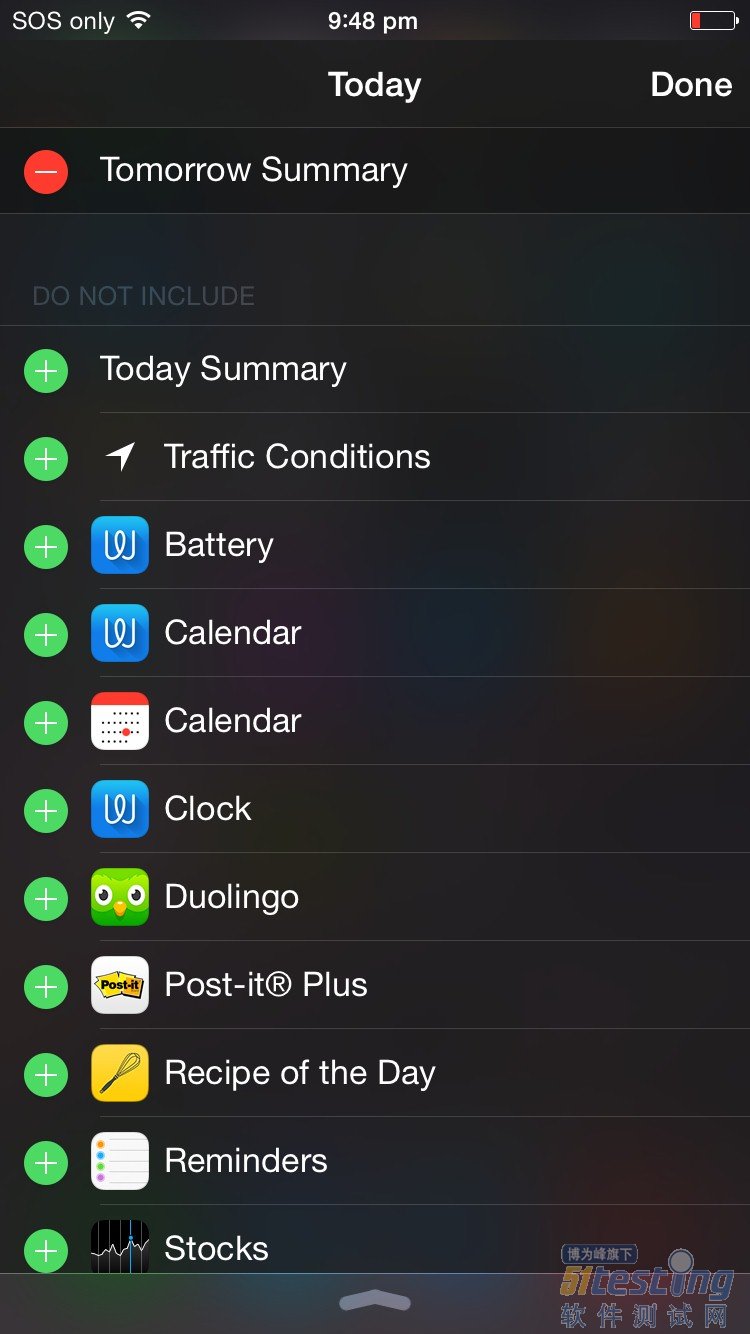
[](http://www.51testing.com/batch.download.php?aid=54310)

图16.3  从iOS 8开始支持的小工具Widget

**尽量减少依赖-移动App测试的22条军规(8)**

**军规18  尽量减少依赖**

　　大家都知道，如果App的依赖越多，出现问题和错误的可能性也就越大，要解决问题付出的成本也越多。所以摆在眼前的问题是，如何尽量减少App的依赖。

　　从笔者的经验来说，减少依赖主要是从App的系统架构入手，尽量减少依赖。

**18.1  对于既有Web版本又有App版本的App要减少依赖**

　　很多App开发时都已经先有了Web的版本，而App只是把Web的内容展示在**[移动](javascript:;" \t "_self)**设备上，这就会让App的很多功能依赖于Web事先实现的方式（如图18.1所示）。

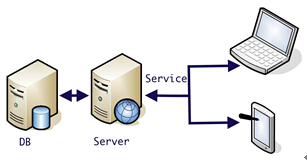
[](http://www.51testing.com/batch.download.php?aid=54330)

　图18.1  常见的先开发Web，后开发App的系统架构

　　这样的系统架构，目的在于让App使用的service能够更多地重用已有的Web版本，实际上，App只是在移动设备上做展示，和Web没有本质的区别。

　　但是由于App所运行的移动设备的特殊性，比如其屏幕大小、设备性能、网络环境等和Web所使用的桌面环境差别很大，所以如果单纯在App上使用Web的service，很有可能造成App性能和样式出现问题。

　　那是不是必须为Web和App都开发独立的service呢？（如图18.2所示）。

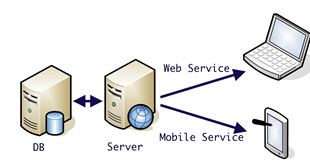
[](http://www.51testing.com/batch.download.php?aid=54341)

图18.2  为Web和App单独开发独立的service

　　其实也没必要这么做，因为重新把Web所有的service再为App实现一遍的开发和测试成本很高，后期的维护成本也很高。

　　所以，笔者的经验是，首先分析，如果已有的Web的service使用在App上，App中会有哪些场景使用到这些service，哪些service会有性能和样式的问题；其次，选择在App上直接使用可能会出现问题的那些Web service来实现独立的service（如图18.3所示）。

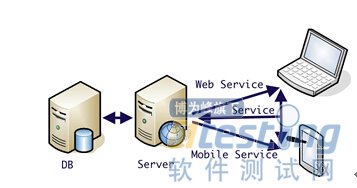
[](http://www.51testing.com/batch.download.php?aid=54342)

图18.3  只针对App与Web不同之处单独开发独立的service

　　也许读者会问，App的系统架构确实对测试有影响，不过系统架构的设计并不是测试人员负责的，那和具体测试没什么关系吧？其实不尽然，在做service分析的时候，对于App使用的场景和需要用到的具体service，没有人比测试人员更熟悉了，这也正是测试人员可以为App贡献的关键时刻。

**Mobile Testing: Complete Guide to Test your Mobile Apps**

Some or all of the following testing types may be performed depending on your mobile testing requirements **[](http://www.guru99.com/images/Mobile_Testing.png)**

* Functional testing
* Performance testing
* Security testing
* Usability testing
* Compatibility testing
* Recoverability Testing

**Functional testing:**

The functional testing of Mobiles normally consists in the areas of testing user interactions as well as testing the transactions. The various factors which are relevant in functional testing are

1. Type of application based upon the business functionality usages (banking, gaming, social or business)
2. Target audience type (consumer, enterprise, education)
3. Distribution channel which is used to spread the application (e.g. Apple App Store, Google play, direct distribution)

The most fundamental test scenarios in the functional testing can be considered as :

1. To validate whether all the required mandatory fields are working as required.
2. To validate that the mandatory fields are displayed in the screen in a distinctive way than the non-mandatory fields.
3. To validate whether the application works as per as requirement whenever the application starts/stops.
4. To validate whether the application goes into minimized mode whenever there is an incoming phone call. In order to validate the same we need to use a second phone, to call the device.
5. To validate whether the phone is able to store, process and receive SMS whenever the app is running. In order to validate the same we need to use a second phone to send sms to the device which is being tested and where the application under test is currently running.
6. To validate that the device is able to perform required multitasking requirements whenever it is necessary to do so.
7. To validate that the application allows necessary social network options such as sharing, posting and navigation etc.
8. To validate that the application supports any payment gateway transaction such as Visa, Mastercard, Paypal etc as required by the application.
9. To validate that the page scrolling scenarios are being enabled in the application as necessary.
10. To validate that the navigation between relevant modules in the application are as per the requirement.
11. To validate that the truncation errors are absolutely to an affordable limit.

/\*

# truncation error

* [inShare](javascript:void(0);)
* [Email](http://whatis.techtarget.com/definition/truncation-error)
* [Comment](http://whatis.techtarget.com/definition/truncation-error#comments)
* [RSS](http://whatis.techtarget.com/rss)
* [Print](http://whatis.techtarget.com/definition/truncation-error?vgnextfmt=print)
* [A](http://whatis.techtarget.com/definition/truncation-error)
* [AA](http://whatis.techtarget.com/definition/truncation-error)
* AA

Truncation error is the difference between a [truncated](http://whatis.techtarget.com/definition/truncate) value and the actual value. A truncated quantity is represented by a numeral with a fixed number of allowed digits, with any excess digits "chopped off" (hence the expression "truncated").

As an example of truncation error, consider the [speed of light](http://whatis.techtarget.com/definition/speed-of-light) in a vacuum. The official value is 299,792,458 meters per second. In [scientific (power-of-10) notation](http://whatis.techtarget.com/definition/scientific-notation-power-of-10-notation), that quantity is expressed as 2.99792458 x 108. Truncating it to two decimal places yields 2.99 x 108. The truncation error is the difference between the actual value and the truncated value, or 0.00792458 x 108. Expressed properly in scientific notation, it is 7.92458 x 105.

In computing applications, truncation error is the discrepancy that arises from executing a finite number of steps to approximate an infinite process. For example, the infinite series 1/2 + 1/4 + 1/8 + 1/16 + 1/32 ... adds up to exactly 1. However, if we truncate the series to only the first four terms, we get 1/2 + 1/4 + 1/8 + 1/16 = 15/16, producing a truncation error of 1 - 15/16, or 1/16.

\*/

1. To validate that the user receives an appropriate error message like “Network error. Please try after some time” whenever there is any network error.
2. To validate that the installed application enables other applications to perform satisfactorily, and it does not eat into the memory of the other applications.
3. To validate that the application resumes at the last operation in case of a hard reboot or system crash.
4. To validate whether the installation of the application can be done smoothly provided the user has the necessary resources and it does not lead to any significant errors.
5. To validate that the application performs auto start facility according to the requirements.
6. To validate whether the application performs according to the requirement in all versions of Mobile that is 2g, 3g and 4g.
7. To perform regression testing to uncover new software bugs in existing areas of a system after changes have been made to them. Also rerun previously performed tests to determine that the program behavior has not changed due to the changes.
8. To validate whether the application provides an available user guide for those who are not familiar to the app

**Performance testing:**

This type of testing’s fundamental objective is to ensure that the application performs acceptably under certain performance requirements such as access by a huge number of users or the removal of a key infrastructure part like a database server.

The general test scenarios for performance testing in a Mobile application are:

1. To determine whether the application performs as per the requirement under different load conditions.
2. To determine whether the current network coverage is able to support the application at peak, average and minimum user levels.
3. To determine whether the existing client-server configuration setup provides the required optimum performance level.
4. To identify the various application and infrastructure bottlenecks which prevent the application to perform at the required acceptability levels.
5. To validate whether the response time of the application is as per as the requirements.
6. To evaluate product and/or hardware to determine if it can handle projected load volumes.
7. To evaluate whether the battery life can support the application to perform under projected load volumes.
8. To validate application performance when network is changed to WIFI from 2G/3G or vice versa.
9. To validate each of the required the CPU cycle is optimization

/\* A cycle generally refers to a clock cycle (a single tick of the internal clock.) Cycles per second are also called Hertz, or Hz, thus a 2 GHz CPU (two gigahertz) goes through 2,000,000,000 cycles every second.

Some instructions on a CPU take multiple cycles to execute, and optimization means in most cases multiple instructions are executed in a single cycle... Thus the number of instructions executed in a given cycle is variable, but the number of CPU cycles any given instruction takes is usually fixed.

\*/

1. To validate that the battery consumption, memory leaks, resources like GPS, Camera performance is well within required guidelines.
2. To validate the application longevity whenever the user load is rigorous.
3. To validate the network performance while moving around with the device.
4. To validate the application performance when only intermittent phases of connectivity is required.

**Security testing:**

[](http://www.guru99.com/images/Mobile_Testing1.png)

The fundamental objective of security testing is to ensure that the application’s data and networking security requirements are met as per guidelines.

The following are the most crucial areas for checking the security of Mobile applications.

1. To validate that the application is able to withstand any brute force attack which is an automated process of trial and error used to guess a person’s username, password or credit-card number.
2. To validate whether an application is not permitting an attacker to access sensitive content or functionality without proper authentication.
3. To validate that the application has a strong password protection system and it does not permit an attacker to obtain, change or recover another user’s password.
4. To validate that the application does not suffer from insufficient session expiration.
5. To identify the dynamic dependencies and take measures to prevent any attacker for accessing these vulnerabilities.
6. To prevent from [SQL](http://www.guru99.com/sql.html) injection related attacks.
7. To identify and recover from any unmanaged code scenarios.
8. To ensure whether the certificates are validated, does the application implement Certificate Pinning or not.
9. To protect the application and the network from the denial of service attacks.
10. To analyze the data storage and data validation requirements.
11. To enable the session management for preventing unauthorized users to access unsolicited information.
12. To check if any cryptography code is broken and ensure that it is repaired.
13. To validate whether the business logic implementation is secured and not vulnerable to any attack from outside.
14. To analyze file system interactions, determine any vulnerability and correct these problems.
15. To validate the protocol handlers for example trying to reconfigure the default landing page for the application using a malicious iframe.
16. To protect against malicious client side injections.
17. To protect against malicious runtime injections.
18. To investigate file caching and prevent any malicious possibilities from the same.
19. To prevent from insecure data storage in the keyboard cache of the applications.
20. To investigate cookies and preventing any malicious deeds from the cookies.
21. To provide regular audits for data protection analysis.
22. Investigate custom created files and preventing any malicious deeds from the custom created files.
23. To prevent from buffer overflows and memory corruption cases.
24. To analyze different data streams and preventing any vulnerabilities from these.

**Usability testing:**

[](http://www.guru99.com/images/Mobile_Testing2.png)

The usability testing process of the Mobile application is performed to have a quick and easy step application with less functionality than a slow and difficult application with many features. The main objective is to ensure that we end up having an easy-to-use, intuitive and similar to industry-accepted interfaces which are widely used.

1. To ensure that the buttons should have the required size and be suitable to big fingers.
2. To ensure that the buttons are placed in the same section of the screen to avoid confusion to the end users.
3. To ensure that the icons are natural and consistent with the application.
4. To ensure that the buttons, which have the same function should also have the same color.
5. To ensure that the validation for the tapping zoom-in and zoom-out facilities should be enabled.
6. To ensure that the keyboard input can be minimized in an appropriate manner.
7. To ensure that the application provides a method for going back or undoing an action, on touching the wrong item, within an acceptable duration.
8. To ensure that the contextual menus are not overloaded because it has to be used quickly.
9. To ensure that the text is kept simple and clear to be visible to the users.
10. To ensure that the short sentences and paragraphs are readable to the end users.
11. To ensure that the font size is big enough to be readable and not too big or too small.
12. To validate the application prompts the user whenever the user starts downloading a large amount of data which may be not conducive for the application performance.
13. To validate that the closing of the application is performed from different states and verify if it re-opens in the same state.
14. To ensure that all strings are converted into appropriate languages whenever a language translation facility is available.
15. To ensure that the application items are always synchronized according to the user actions.
16. To ensure that the end user is provided with a user manual which helps the end user to understand and operate the application who may be not familiar with the application’s proceedings

Usability testing is normally performed by manual users since only human beings can understand the sensibility and comfort ability of the other users.

**Compatibility testing:**

Compatibility testing on mobile devices is performed to ensure that since mobile devices have different size, resolution, screen, version and hardware so the application should be tested across all the devices to ensure that the application works as desired.

The following are the most prominent areas for compatibility testing.

1. To validate that the user Interface of the application is as per the screen size of the device, no text/control is partially invisible or inaccessible.
2. To ensure that the text is readable for all users for the application.
3. To ensure that the call/alarm functionality is enabled whenever the application is running. The application is minimized or suspended on the event of a call and then whenever the call stops the application is resumed.

**Recoverability Testing**

1. Crash recovery and transaction interruptions
2. Validation of the effective application recovery situation post unexpected interruption/crash scenarios.
3. Verification of how the application handles a transaction during a power failure (i.e. Battery dies or a sudden manual shutdown of the device)
4. The validation of the process where the connection is suspended, the system needs to re-establish for recovering the data directly affected by the suspended connection.

**Other Important Checks:**

1. Installation testing (whether the application can be installed in a reasonable amount of time and with required criterion)
2. Uninstallation testing (whether the application can be uninstalled in a reasonable amount of time and with required criterion)
3. Network test cases (validation of whether the network is performing under required load or not, whether the network is able to support all the necessary applications during the testing procedures)
4. Check Unmapped keys
5. Check application splash screen

/\*

A **splash screen** is a graphical control element consisting of window containing an image, a logo and the current version of the software. A **splash screen** usually appears while a game or program is launching.

\*/

1. Continued keypad entry during interrupts and other times like network issues
2. Methods which deal with exiting the application
3. Charger effect while an application is running in the background
4. Low battery and high performance demand
5. Removal of battery while an application is being performed
6. Consumption of battery by application
7. Check Application side effects

-------------------------

SQL Injection (SQLi)

SQL injection (SQLi) refers to an injection attack wherein an attacker can execute malicious SQL statements (also commonly referred to as a malicious *payload*) that control a web application’s database server (also commonly referred to as a *Relational Database Management System – RDBMS*). Since an SQL injection vulnerability could possibly affect any website or web application that makes use of an SQL-based database, the vulnerability is one of the oldest, most prevalent and most dangerous of web application vulnerabilities.

By leveraging an SQL injection vulnerability, given the right circumstances, an attacker can use it to bypass a web application’s authentication and authorization mechanisms and retrieve the contents of an entire database. SQL injection can also be used to add, modify and delete records in a database, affecting data integrity.

To such an extent, SQL injection can provide an attacker with unauthorized access to sensitive data including, customer data, personally identifiable information (PII), trade secrets, intellectual property and other sensitive information.

How SQL Injection works

In order to run malicious SQL queries against a database server, an attacker must first find an input within the web application that is included inside of an SQL query.

In order for an SQL injection attack to take place, the vulnerable website needs to directly include user input within an SQL statement. An attacker can then insert a payload that will be included as part of the SQL query and run against the database server.

The following server-side pseudo-code is used to authenticate users to the web application.

# Define POST variables

**uname = request.POST['username']**

**passwd = request.POST['password']**

# SQL query vulnerable to SQLi

sql = “**SELECT** id **FROM** users **WHERE** username=’” + **uname** + “’ **AND** **password**=’” + **passwd** + “’”

# **Execute** the **SQL** statement

**database**.**execute**(**sql**)

The above script is a simple example of authenticating a user with a username and a password against a database with a table named users, and a username and password column.

The above script is vulnerable to SQL injection because an attacker could submit malicious input in such a way that would alter the SQL statement being executed by the database server.

A simple example of an SQL injection payload could be something as simple as setting the password field topassword’ OR 1=1.

This would result in the following SQL query being run against the database server.

**SELECT** id **FROM** users **WHERE** username=’username’ **AND** **password**=’**password**’ **OR** 1=1’

An attacker can also comment out the rest of the SQL statement to control the execution of the SQL query further.

-- MySQL, MSSQL, Oracle, PostgreSQL, SQLite

' OR '1'='1' **--**

' OR '1'='1' **/\***

-- MySQL

' OR '1'='1' **#**

-- Access (using null characters)

' OR '1'='1' **%00**

' OR '1'='1' **%16**

Once the query executes, the result is returned to the application to be processed, resulting in an authentication bypass. In the event of authentication bypass being possible, the application will most likely log the attacker in with the first account from the query result — the first account in a database is usually of an administrative user.

What’s the worst an attacker can do with SQL?

SQL is a programming language designed for managing data stored in an RDBMS, therefore SQL can be used to access, modify and delete data. Furthermore, in specific cases, an RDBMS could also run commands on the operating system from an SQL statement.

Keeping the above in mind, when considering the following, it’s easier to understand how lucrative a successful SQL injection attack can be for an attacker.

* An attacker can use SQL injection to bypass authentication or even impersonate specific users.
* One of SQL’s primary functions is to select data based on a query and output the result of that query. An SQL injection vulnerability could allow the complete disclosure of data residing on a database server.
* Since web applications use SQL to alter data within a database, an attacker could use SQL injection to alter data stored in a database. Altering data affects data integrity and could cause repudiation issues, for instance, issues such as voiding transactions, altering balances and other records.
* SQL is used to delete records from a database. An attacker could use an SQL injection vulnerability to delete data from a database. Even if an appropriate backup strategy is employed, deletion of data could affect an application’s availability until the database is restored.
* Some database servers are configured (intentional or otherwise) to allow arbitrary execution of operating system commands on the database server. Given the right conditions, an attacker could use SQL injection as the initial vector in an attack of an internal network that sits behind a firewall.

The anatomy of an SQL Injection attack

An SQL injection needs just two conditions to exist – **a relational database that uses SQL, and a user controllable input which is directly used in an SQL query.**

In the example below, it shall be assumed that the attacker’s goal is to exfiltrate data from a database by exploiting an SQL injection vulnerability present in a web application.

Supplying an SQL statement with improper input, for example providing a string when the SQL query is expecting an integer, or purposely inserting a syntax error in an SQL statement cause the database server to throw an error.

Errors are very useful to developers during development, but if enabled on a live site, they can reveal a lot of information to an attacker. SQL errors tend to be descriptive to the point where it is possible for an attacker to obtain information about the structure of the database, and in some cases, even to enumerate an entire database just through extracting information from error messages – this technique is referred to as *error-based SQL injection*. To such an extent, database errors should be disabled on a live site, or logged to a file with restricted access instead.

Another common technique for exfiltrating data is to leverage the UNION SQL operator, allowing an attacker to combine the results of two or more SELECT statements into a single result. This forces the application to return data within the HTTP response – this technique is referred to as *union-based SQL injection*.

The following is an example of such a technique. This can be seen on **testphp.vulnweb.com**, an intentionally vulnerable website hosted by Acunetix.

The following HTTP request is a normal request that a legitimate user would send.