Robotium中文API(01)-solo.assertCurrentActivity

**AssertCurrentActivity**

public void assertCurrentActivity([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) message,

[Class](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/Class.html?is-external=true) expectedClass)

检查当前程序显示的Activity是否是预期的Activity。

**参数:**

message – 如果断言失败，显示此消息

expectedClass – 预期的Activity类对象

例：assertCurrentActivity(“不是MyActivity”, MyActivity.class);

public void assertCurrentActivity([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) message,

[Class](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/Class.html?is-external=true) expectedClass,

                                  boolean isNewInstance)

检查当前程序显示的Activity是否是预期的Activity。并验证预期的Activity是否是一个新的Activity实例。

参数:

message -如果断言失败，显示此消息

expectedClass -预期的Activity类

isNewInstance – true 如果预期的Activity是一个新的Activity实例。

public void assertCurrentActivity([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) message,

[String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) name)

检查当前程序显示的Activity是否是预期的Activity。

参数:

message – 如果断言失败，显示此消息

name – 预期的Activity类的名称

例：assertCurrentActivity(“不是MyActivity”, “MyActivity”);

public void assertCurrentActivity([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) message,

[String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) name,

                                  boolean isNewInstance)

检查当前程序显示的Activity是否是预期的Activity。并验证预期的Activity是否是一个新的Activity实例。

参数:

message -如果断言失败，显示此消息

expectedClass -预期的Activity类的名称

isNewInstance – true 如果预期的Activity是一个新的Activity实例

# Robotium中文API(02)-solo.assertMemoryNotLow

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### ****assertMemoryNotLow****

public void **assertMemoryNotLow**()

*断言系统可用内存是否过低。*

Robotium中文API(03)-solo.clearEditText

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

**clearEditText**

public void **clearEditText**(android.widget.EditText editText)

清空输入框的值。

**参数:**

editText – 要清空的输入框

——————————————————————————————————————————————

public void **clearEditText**(int index)

清空输入框的值。

**参数:**

index – 要清空的输入框位置. 如果是0说明只要一个输入框可用。

Robotium中文API(04)-solo.clickInList

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

**clickInList**

public [ArrayList](http://java.sun.com/j2se/1.5.0/docs/api/java/util/ArrayList.html?is-external=true)<android.widget.TextView> **clickInList**(int line)

点击一个给定的列表行并返回此行显示的TextView集合，默认操作第一个ListView。

**参数:**

line – 被点击的行

**返回:**

返回包含TextView对象的数值

# Robotium4.0 ChangeLog

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

## Robotium4.0最大更新是支持了对WebView的操作，另外增加和弃用部分方法。

**新特性:**

* waitForCondition(Condition condition, final int timeout)    //**等待条件是否满足**
* **支持WebView操作**
* waitForWebElement(By by) //Example: waitForWebElement(By.id(“id”));
* waitForWebElement(By by, int timeout, boolean scroll)
* waitForWebElement(By by, int match, int timeout, boolean scroll)
* clickOnWebElement(WebElement webElement)   //**点击某个WebElement**
* clickOnWebElement(By by)
* clickOnWebElement(By by, int match)
* clickOnWebElement(By by, int match, boolean scroll)
* enterTextInWebElement(By by, String text)   //**某个WebElement中输入指定文本**
* typeTextInWebElement(By by, String text)
* typeTextInWebElement(By by, String text, int match)
* typeTextInWebElement(WebElement webElement, String text)
* clearTextInWebElement(By by)
* getWebElement(By by, int index)
* getCurrentViews(Class classToFilterBy)
* getCurrentViews(Class classToFilterBy, View parent)
* takeScreenshot(String name, int quality)   //**截取图片并且以一定比例压缩**

**RobotiumUtils:**

* removeInvisibleViews(Iterable viewList)
* filterViews(Class classToFilterBy, Iterable<?> viewList)
* filterViewsToSet(Class classSet, Iterable viewList)
* sortViewsByLocationOnScreen(List views)
* sortViewsByLocationOnScreen(List views, boolean yAxisFirst)
* getNumberOfMatches(String regex, View view, Set uniqueTextViews)

**弃用的方法:**

* getAllOpenedActivities()
* finishInactiveActivities()
* getCurrentXViews //replaced by getCurrentViews(Class classToFilterBy). Example: getCurrentViews(ListView.class)
* Robotium中文API(05)-solo. clickLongInList
* *作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)
* **clickLongInList**
* public [ArrayList](http://java.sun.com/j2se/1.5.0/docs/api/java/util/ArrayList.html?is-external=true)<android.widget.TextView> **clickLongInList**(int line)
* 长按一个给定的列表行并返回此行显示的TextView集合，默认操作第一个找到的ListView。
* **参数:**
* line – 被点击的行
* **Returns:**
* 返回此行一个包含TextView对象的数组
* public [ArrayList](http://java.sun.com/j2se/1.5.0/docs/api/java/util/ArrayList.html?is-external=true)<android.widget.TextView> **clickLongInList**(int line,
* int index)
* 长按一个指定的ListView中给定的列表行并返回此行显示的TextView集合。
* **参数:**
* line – 被点击的行
* index – 列表索引. 1 表示当前有两个列表
* **Returns:**
* 返回位于指定列表中指定行的TextView对象数组
* public [ArrayList](http://java.sun.com/j2se/1.5.0/docs/api/java/util/ArrayList.html?is-external=true)<android.widget.TextView> **clickLongInList**(int line,
* int index,
* int time)
* 长按一个指定的ListView中给定的列表行并返回此行显示的TextView集合。
* **参数:**
* line – 被点击的行
* index – 列表索引. 1 表示当前有两个列表
* time – 长按的时间
* **Returns:**
* 返回位于指定列表中指定行的TextView对象数组

# Robotium中文API(06)-solo.clickLongOnScreen

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickLongOnScreen

public void **clickLongOnScreen**(float x,

                              float y)

长按屏幕上给定的坐标

**参数:**

x – x的坐标

y – y的坐标

public void **clickLongOnScreen**(float x,

                              float y,

                              int time)

长按屏幕上给定的坐标一段时间

**参数:**

x – x的坐标

y – y的坐标

time – 长按时间

# Robotium中文API(07)-solo.clickLongOnText

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickLongOnText

public void **clickLongOnText**([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) text)

*长按一个给定的视图（控件）.当需要的时候自动滚动. 然后*[*clickOnText(String)*](file:///C:\Users\Administrator\Desktop\robotium\robotium-solo-4.0-javadoc\com\jayway\android\robotium\solo\Solo.html#clickOnText(java.lang.String))*可以在长按以后用来点击上下文显示的菜单项。*

***参数:***

*text – 被点击的文本. 这个参数可以作为一个正则表达式*

public void **clickLongOnText**([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) text, int match)

*长按一个给定的视图（控件）.当需要的时候自动滚动. 然后*[*clickOnText(String)*](file:///C:\Users\Administrator\Desktop\robotium\robotium-solo-4.0-javadoc\com\jayway\android\robotium\solo\Solo.html#clickOnText(java.lang.String))*可以在长按以后用来点击上下文显示的菜单项。*

***参数:***

*text – 被点击的文本. 这个参数可以作为一个正则表达式*

*match – 如果多个对象（控件）匹配这个文本, 这将确定哪一个被点击*

public void **clickLongOnText**([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) text, int match, boolean scroll)

*长按一个给定的视图（控件）. 然后*[*clickOnText(String)*](file:///C:\Users\Administrator\Desktop\robotium\robotium-solo-4.0-javadoc\com\jayway\android\robotium\solo\Solo.html#clickOnText(java.lang.String))*可以在长按以后用来点击上下文显示的菜单项。*

***参数:***

*text – 被点击的文本. 这个参数可以作为一个正则表达式*

*match – 如果多个对象（控件）匹配这个文本, 这将确定哪一个被点击*

*scroll – 如果为真则表示在需要时滚动，否则只在当前屏幕内查找*

public void **clickLongOnText**([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) text, int match, int time)

*长按一个给定的视图（控件）. 然后*[*clickOnText(String)*](file:///C:\Users\Administrator\Desktop\robotium\robotium-solo-4.0-javadoc\com\jayway\android\robotium\solo\Solo.html#clickOnText(java.lang.String))*可以在长按以后用来点击上下文显示的菜单项。*

***参数:***

*text – 被点击的文本. 这个参数可以作为一个正则表达式*

*match – 如果多个对象（控件）匹配这个文本, 这将确定哪一个被点击*

*time – 长按的时间*

# Robotium中文API(08)-solo.clickLongOnTextAndPress

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickLongOnTextAndPress

public void **clickLongOnTextAndPress**([String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) text, int index)

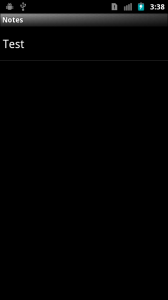
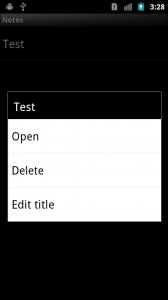
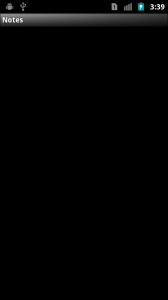
*长按一个给定的视图（控件），然后从显示的上下文菜单中选择一个选项。当需要的时候自动滚动。*

***参数:***

*text – 被点击的文本. 这个参数可以作为一个正则表达式*

*index – 被点击的菜单项索引. 如果为0表示仅仅一个可用*

**示例：**

[](http://www.robotium.cn/wp-content/uploads/2013/04/device-2013-04-05-233802.png)    [](http://www.robotium.cn/wp-content/uploads/2013/04/device-2013-04-05-232850.png)    [](http://www.robotium.cn/wp-content/uploads/2013/04/device-2013-04-05-233921.png)

## 图 一                  图二                           图三

上面是Android官方的记事本示例截图，当前存在一个名称为“Test”的记录，

solo.clickLongOnTextAndPress(“Test”, 1) 表示长按图一中名称为“Test”的记录并弹出图二中含有菜单项的弹出框，index参数1表示在弹出上下文菜单（图二的弹出框）后点击索引为1的选项也就是Delete，图三表示删除记录成功。

注：Open、Delete、Edit title对应的索引为0、1、2。

# Robotium中文API(09)-solo.clickLongOnView

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickLongOnView

public void **clickLongOnView**(android.view.View view)

*长按一个给定的视图（控件）.*

***参数:***

*view – 被长按的视图*

public void **clickLongOnView**(android.view.View view, int time)

*长按一个给定的视图（控件）.*

***参数:***

*view – 被长按的视图*

*time – 长按的时间*

# Robotium中文API(10)-solo.clickOnActionBarHomeButton

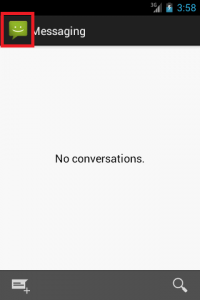
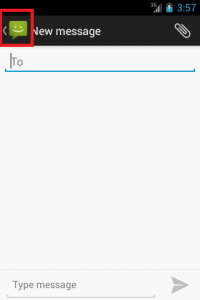
*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

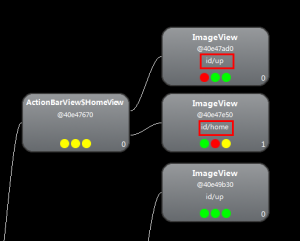
### clickOnActionBarHomeButton

public void **clickOnActionBarHomeButton**()

*点击ActionBar的home或者up按钮.*

**示例：**

**图一**[](http://www.robotium.cn/wp-content/uploads/2013/04/device-2013-04-06-235831.png)    **图二**[](http://www.robotium.cn/wp-content/uploads/2013/04/device-2013-04-06-235810.png)

**图三**[](http://www.robotium.cn/wp-content/uploads/2013/04/20130407000000.png)

图一为Android4.2系统短信界面，状态栏下面就是ActionBar（关于ActionBar介绍：<http://developer.android.com/guide/topics/ui/actionbar.html>）

图二是进入新建短信界面，此时可以看出红框中的区别，出现可以回到上一级标志，也可以参考官方文档的邮件图标：[](http://www.robotium.cn/wp-content/uploads/2013/04/actionbar-logo.png)

图三是图一、图二中红框标出的区域的布局，其中home和up就是应用图标和上一级标志。

clickOnActionBarHomeButton()的作用就是点击ActionBar的此区域。

# Robotium中文API(11)-solo.clickOnActionBarItem

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickOnActionBarItem

public void **clickOnActionBarItem**(int resourceId)

*点击一个给定id的ActionBar选项。*

***参数:***

*resourceId – ActionBar选项的选项对应的id(R.id)*

# Robotium中文API(12)-solo.clickOnButton

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickOnButton

public void **clickOnButton**(int index)

*通过一个给定的索引点击一个按钮。*

***参数:***

*index – 点击的按钮索引. 如果是0则表示只有一个可用*

*public void* ***clickOnButton****(*[*String*](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true)*name)*

*通过一个给定的字符串点击一个按钮. 当需要的时候讲自动滚动.*

***参数:***

*name – 呈现给用户的按钮名称. 参数可以作为一个正则表达式。*

# Robotium中文API(13)-solo.clickOnCheckBox

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickOnCheckBox

public void **clickOnCheckBox**(int index)

*通过给定的索引点击一个复选框.*

***参数:***

*index – 被点击的复选框索引. 如果是0则表示只有一个可用。*

# Robotium中文API(14)-solo.clickOnEditText

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickOnEditText

public void **clickOnEditText**(int index)

*通过给定的索引点击一个文本框.*

***参数:***

*index – 被点击的文本框索引. 如果是0则表示只有一个可用。*

# Robotium中文API(15)-solo.clickOnImage

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

### clickOnImage

public void **clickOnImage**(int index)

*通过指定的索引点击一个ImageView.*

***参数:***

*index – 被点击的ImageView索引. 0表示仅仅一个可用*

# Robotium中文API(16)-solo.clickOnImageButton

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

#### clickOnImageButton

public void clickOnImageButton(int index)

点击一个匹配的ImageButton索引

***参数:***

*index – 被点击的ImageButton索引.0表示仅仅一个可用*

Robotium中文API(17)-solo.getWebUrl

*作者*[**HAO.SHEN**](http://www.robotium.cn/archives/author/hao-shen)

public [String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) getWebUrl()

返回当前页面的URL.

***返回:***

*当前页面的URLL*

注意：此方法默认返回index为0的WebView的URL。

# Robotium中文API(25)-solo.typeTextInWebElement

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

### typeTextInWebElement

该方法根据传入参数不同有不同使用方法。

public void **typeTextInWebElement**(By by, String text)

**参数:**

by – By对象，如: By.id("id") 和 By.name("name")

text – 待输入的文本内容。

Match – 假定有多个对象，根据此值决定匹配哪一个。

例子：

public void testSearch(){ solo.sleep(4000); webview = (WebView) solo.getView(R.id.webView1); //word是百度网页搜索框的name值，补充手机访问的是wap.baidu.com 查询id或name可以通过firefox很方便的得到 By search = By.name("word"); solo.typeTextInWebElement(search, "robotium中文网");

* Robotium中文API(26)-solo.enterText
* *作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)
* **enterText**
* public void enterText(int index, String text) 在一个给定位置的EditText中输入文本。 参数： index – EditText的位置。如果只有一个可用的则为0。 text – 输入到EditText中的文本字符串。
* public void enterText(android.widget.EditText editText, String text)     在一个给定的EditText输入文本。 参数： editText – 待输入的editText。 text - 输入到EditText中的文本字符串。

Robotium中文API(27)-solo.typeText

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

public void typeText(int index, [String](http://java.sun.com/j2se/1.5.0/docs/api/java/lang/String.html?is-external=true) text)

        在一个给定位置的EditText输入文本。

参数：

        index – EditText的位置。如果只有一个可用的则为0。

        text – 输入到EditText中的文本字符串。

public void typeText(android.widget.EditText editText, String text)

        在一个给定的EditText输入文本。

参数：

        editText – 待输入的editText。

        text - 输入到EditText中的文本字符串。

# Robotium中文API(28)-solo.waitForActivity

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

### waitForActivity

public boolean waitForActivity(java.lang.String name)

等待一个匹配指定名称的Activity，默认超时时间30秒。  
参数：  
name – 指定的Activity名字，如“MyActivity“。  
返回：  
true 假如指定的Activity在超时前出现；反之则，false。

public boolean waitForActivity(java.lang.String name,int timeout)

参数：  
timeout – 等待的时间（毫秒计算）。

public boolean waitForActivity(java.lang.Class<? extends android.app.Activity> activityClass)

参数：  
activityClass – 指定的Activity，如“MyActivity.class”。

该方法使用Waiter类中的waitForActivity方法来实现，通过getCurrentActivity方法得到当前Activity，来和指定的Activity进行对比。

# Robotium中文API(29)-solo.waitForDialogToClose

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

### waitForDialogToClose

public boolean **waitForDialogToClose**(long timeout)

等待一个对话框关闭

**参数:**

timeout – 等待的超时时间。

**返回:**

如果对话框在超时时间之前关闭就返回true,反之则false。

# Robotium中文API(30)-solo.waitForDialogToOpen

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

### waitForDialogToOpen

public boolean **waitForDialogToOpen**(long timeout)

等待一个对话框打开

**参数:**

timeout – 等待的超时时间。

**返回:**

如果对话框在超时时间之前打开就返回true,反之则false。

# Robotium中文API(31)-solo.waitForFragmentById

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

### waitForFragmentById

public boolean **waitForFragmentById**(int id)

 等待一个匹配id的fragment,默认超时时间20秒。

**参数:**

id – 给定的匹配的id。

**返回:**

如果fragment在超时时间之前出现就返回true,反之则false。

Robotium中文API(32)-solo.waitForFragmentByTag

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

public boolean **waitForFragmentByTag**(java.lang.String tag)

等待一个匹配标签的fragment,默认超时时间20秒。

**参数:**

tag –标签的名称。

**返回:**

如果fragment在超时时间出现打开就返回true,反之则false。

Robotium中文API(33)-solo.waitForLogMessage

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

public boolean **waitForLogMessage**(java.lang.String logMessage)

等待指定的日志消息出现。需要读日志权限（android.permission.READ\_LOGS），在应用程序的AndroidManifest.xml文件中配置。

**参数:**

logMessage –等待的日志信息内容。

**返回:**

如果日志在超时时间之前找到就返回true,反之则false。

# Robotium中文API(34)-solo.waitForText

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

### waitForText

public boolean **waitForText**(java.lang.String text)

等待指定的文本出现。默认的超时时间是20秒。

**参数:**

text –等待的出现的文本内容，。

**返回:**

如果文本在超时时间之前找到就返回true,反之则false。

Robotium中文API(35)-solo.waitForView

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

public <T extends android.view.View> boolean**waitForView**(java.lang.Class<T> viewClass)

等待一个指定的匹配类型的View出现，默认超时时间20秒。

**参数:**

text – 等待出现View对象

**返回:**

如果View在超时时间之前找到就返回true,反之则false。

Robotium中文API(36)-solo.waitForWebElement

*作者*[**喜力**](http://www.robotium.cn/archives/author/lizhenghuan)

public boolean **waitForWebElement**(By by)

等待一个指定的匹配的By对象的WebElement出现，默认超时时间20秒。

**参数:**

by – by对象，例如： By.id(“id”) 和 By.name(“name”)

**返回:**

如果WebElement在超时时间之前出现就返回true,反之则false。