Android教程：（三）创建简单的界面，启动其它的Activity

本教程指导你如何使用Android Studio创建一个简单的界面，如何在Activity中启动其它的Activity。

**要点** 本教程不仅适用于Windows，还适用于Linux和OSX。本系列教程的代码都保存在[Github](https://github.com/Leclex/AndroidExamples.git)上面。

在此处，你将了解如何：

* 创建一个新项目
* 创建简单的界面
* 创建一个Activity
* 启动其它的Activity

**步骤 1：在 Android Studio 中创建新项目**

我们来创建名为 HelloAndroid 的新应用。操作方法如下：

1. 启动Android Studio。
2. Start a new Android Studio project(启动一个新的Android Studio工程)。
3. Config your new project(配置你的新工程)。

Application name: StartOtherActivity

Company Domain: .leclex.com

Project location: D:\projects\github\AndroidExamples\StartOtherActivity

1. Select the form factors your app will run on。

勾选Phone and Tablet（默认）

Mininum SDK: API 15: Android 4.0.3(IceCreamSandwich)(默认)

1. Add an activity to Mobile。

选择**Blank Activity**(默认)

1. Customize the Activity。

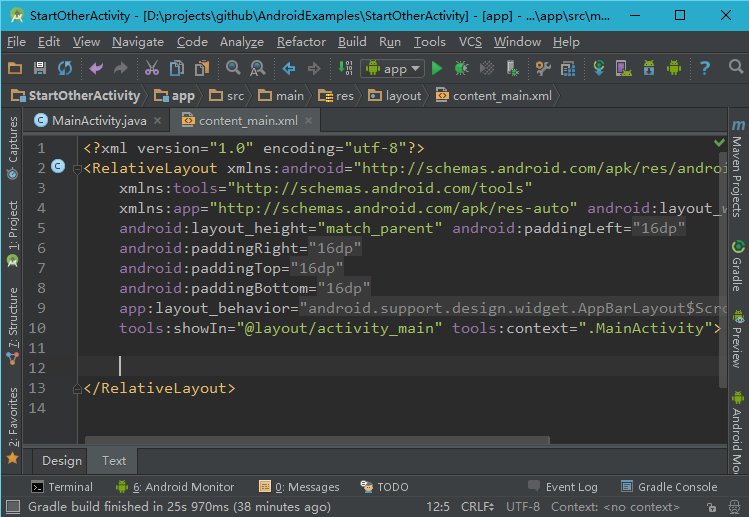
Activity Name: MainActivity

Layout Name: activity\_main

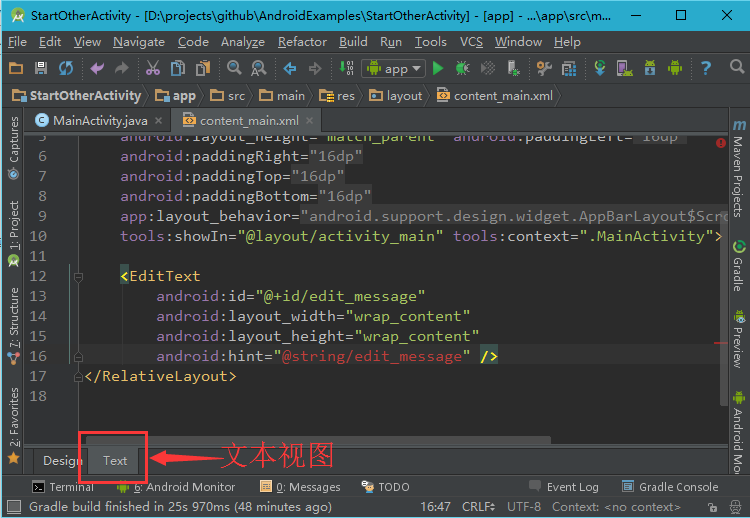
Title: MainActivity

Menu Resource Name: menu\_main

1. 单击**Finish**，Android Studio创建工程并打开Content\_main.xml的**Design**（设计）视图。
2. 创建简单界面。
   1. 单击Content\_main.xml左下角的**Text**切换到文本视图。删除TextView标签。



* 1. 添加一个（EditText）文本输入框。



各属性说明:

#### android:id

这是定义View的唯一标识符。可以在程序代码中通过该标识符对对象进行引用，例如对这个对象进行读和修改的操作(在下一课里将会用到)。

当想从XML里引用资源对象的时候必须使用@符号。紧随@之后的是资源的类型(这里是id)，然后是资源的名字(这里使用的是edit\_message)。

+号只是当你第一次定义一个资源ID的时候需要。这里是告诉SDK此资源ID需要被创建出来。在应用程序被编译之后，SDK就可以直接使用ID值，edit\_message是在项目gen/R.java文件中创建一个新的标识符，这个标识符就和EditText关联起来了。一旦资源ID被创建了，其他资源如果引用这个ID就不再需要+号了。这里是唯一一个需要+号的属性。

#### android:layout\_width 和android:layout\_height

对于宽和高不建议指定具体的大小，使用wrap\_content指定之后，这个视图将只占据内容大小的空间。如果你使用了match\_parent，这时EditText将会布满整个屏幕，因为它将适应父布局的大小。

#### android:hint

当文本框为空的时候,会默认显示这个字符串。对于字符串@string/edit\_message的值所引用的资源应该是定义在单独的文件里，而不是直接使用字符串。因为使用的值是存在的资源，所以不需要使用+号。然而，由于你还没有定义字符串的值，所以在添加@string/edit\_message时候会出现编译错误。下边你可以定义字符串资源值来去除这个错误。

**Note**: 该字符串资源与id使用了相同的名称（edit\_message）。然而，对于资源的引用是区分类型的（比如id和字符串），因此，使用相同的名称不会引起冲突。

* 1. 增加资源字符串。

默认情况下，你的Android项目包含一个字符串资源文件，res/values/strings.xml。打开这个文件，为"edit\_message"增加一个供使用的字符串定义，设置值为"输入一条消息"。

* 打开res/values/string.xml。
* 添加一个string名为"edit\_message" ,值为 "输入一条消息"。
* 再添加一个string名为 "button\_send"，值为"发送"。下面的内容将使用这个string来创建一个按钮。

下边就是修改好的res/values/strings.xml：

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

<resources>

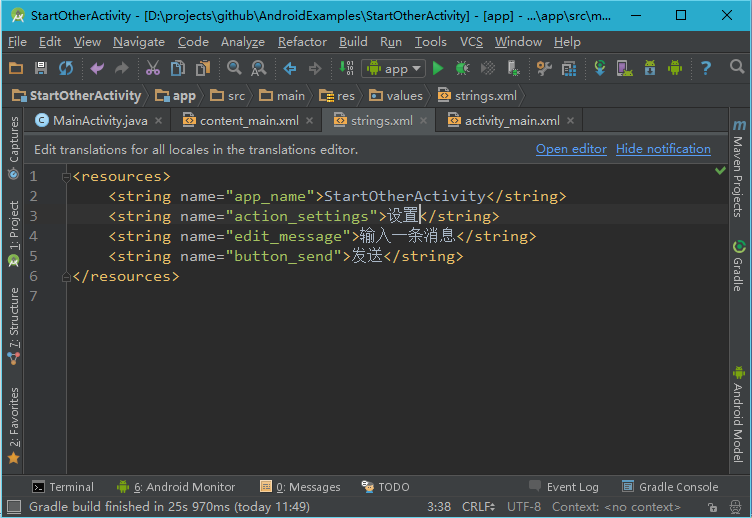
<string name="app\_name">StartOtherActivity</string>

<string name="action\_settings">设置</string>

<string name="edit\_message">输入一条消息</string>

<string name="button\_send">发送</string>

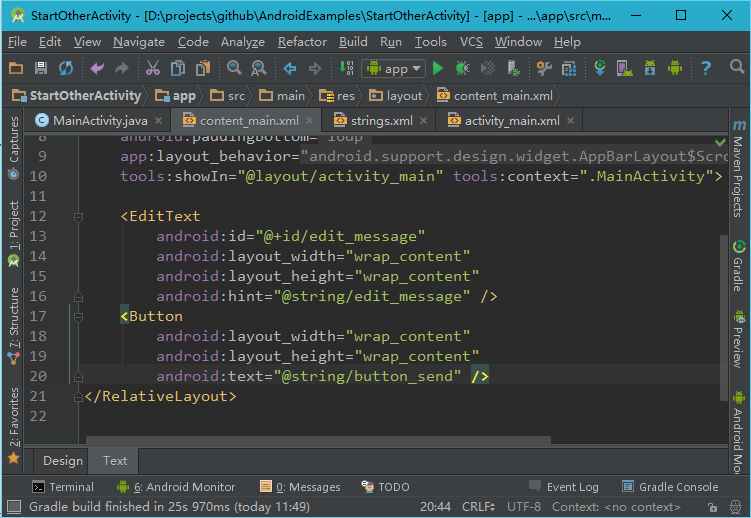
</resources>



当你在用户界面定义一个文本的时候，你应该把每一个文本字符串列入资源文件。这样做的好处是：对于所有字符串值，字符串资源能够单独的修改，在资源文件里你可以很容易的找到并且做出相应的修改。通过选择定义每个字符串，还允许您对不同语言本地化应用程序。

* 1. 添加一个Button（按钮）
* 切换到content\_main.xml的文本视图，输入:

<Button android:layout\_width="wrap\_content" android:layout\_height="wrap\_content" android:text="@string/button\_send" />



**Note** 宽和高被设置为"wrap\_content"，这时按钮占据的大小就是按钮里文本的大小。这个按钮不需要指定android:id的属性，因为Activity代码中不会引用该Button。

当前EditText和Button部件只是适应了他们各自内容的大小，如下图所示：

5.png

这样设置对按钮来说很合适，但是对于文本框来说就不太好了，因为用户可能输入更长的文本内容。因此如果能够占满整个屏幕宽度会更好。RelativeLayout使用权重属性来达到这个目的，你可以使用android:layout\_weight属性来设置。

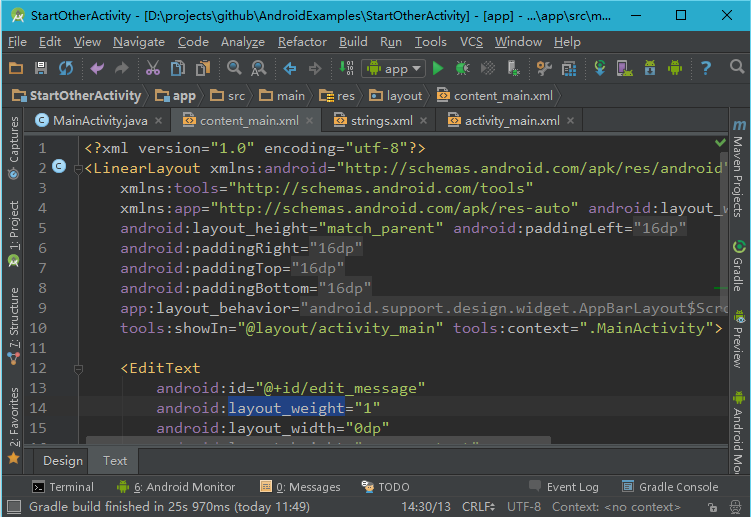
权重的值指的是每个部件所占剩余空间的大小，该值与同级部件所占空间大小有关。就类似于饮料的成分配方：“两份伏特加酒，一份咖啡利口酒”，即该酒中伏特加酒占三分之二。例如，我们设置一个View的权重是2，另一个View的权重是1，那么总数就是3，这时第一个View占据2/3的空间，第二个占据1/3的空间。如果你再加入第三个View，权重设为1，那么第一个View(权重为2的)会占据1/2的空间，剩余的另外两个View各占1/4。(请注意，使用权重的前提一般是给View的宽或者高的大小设置为0dp，然后系统根据上面的权重规则来计算View应该占据的空间。但是很多情况下，如果给View设置了match\_parent的属性，那么上面计算权重时则不是通常的正比，而是反比，也就是权重值大的反而占据空间小)。

对于所有的View默认的权重是0，如果只设置了一个View的权重大于0，则该View将占据除去别的View本身占据的空间的所有剩余空间。因此这里设置EditText的权重为1，使其能够占据除了按钮之外的所有空间。

* 1. 让输入框占满整个窗口

为让 EditText充满剩余空间，在content\_main.xml文件里做如下操作：

* 修改RelativeLayout为LinearLayout
* 设置EditText的layout\_weight属性值为1。
* 设置EditText的layout\_width值为0dp。



为了提升布局的效率，在设置权重的时候，应该把EditText的宽度设为0dp。如果设置"wrap\_content"作为宽度，系统需要自己去计算这个部件所占有的宽度，而此时的因为设置了权重，所以系统自动会占据剩余空间，EditText的宽度最终成了不起作用的属性。

设置权重后的效果图

7.png

切换到**Design**（设计视图）可以看到效果。

现在我们已经拥有了显示一个activity（一个界面）的app（应用），该activity包含了一个EditText（文本输入框）和一个Button(按钮)。接下来，我们将添加一些新的代码到MainActivity中，当用户点击发送按钮时启动一个新的activity。

* 1. 响应发送按钮
* 在content\_main.xml中为 Button 标签添加android:onClick属性

res/layout/activity\_my.xml

<Button

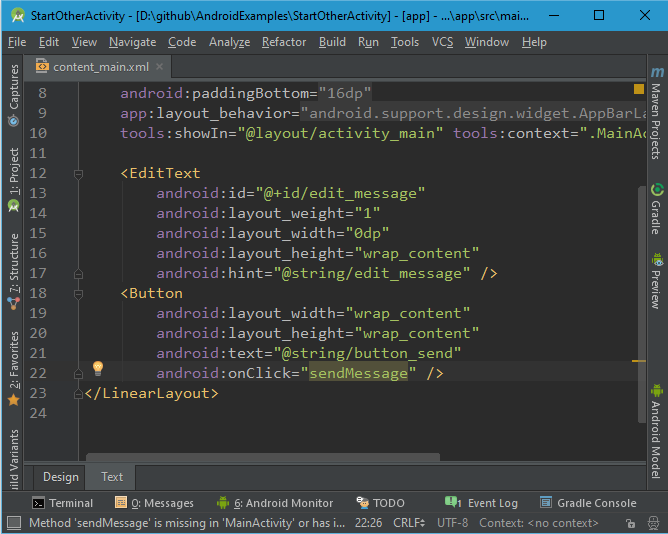
android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="@string/button\_send"

android:onClick="sendMessage" />

android:onClick属性的值"sendMessage"即为用户点击屏幕按钮时触发方法的名字。



* 打开java/com.leclex.startotheractiviry目录下的MyActivity.java 文件，在MyActivity.java 中添加sendMessage() 函数：

java/com.mycompany.myfirstapp/MainActivity.java

*// 当用户单击发送按钮时调用*

**public** **void** **sendMessage**(View view) {

}

为使系统能够将该方法（你刚在MainActivity.java中添加的sendMessage方法）与在android:onClick属性中提供的方法名字匹配，它们的名字必须一致，特别需要注意的是，这个方法必须满足以下条件：

* 是public函数
* 无返回值
* 参数唯一(为View类型,代表被点击的视图）

接下来，你可以在这个方法中编写读取文本内容，并将该内容传到另一个Activity的代码。

* 创建一个Intent

Intent是在不同组件中(比如两个Activity)提供运行时绑定的对象。Intent代表一个应用"想去做什么事"，你可以用它做各种各样的任务，不过大部分的时候他们被用来启动另一个Activity。

在MainActivity.java的sendMessage()方法中创建一个Intent并启动名为DisplayMessageActivity的Activity：

java/com.leclex.startotheractivity/MainActivity.java

Intent intent = **new** Intent(**this**, DisplayMessageActivity.**class**);

**Note**：如果使用的是类似Android Studio的IDE，这里对DisplayMessageActivity的引用会报错，因为这个类还不存在；暂时先忽略这个错误，我们很快就要去创建这个类了。

在这个Intent构造函数中有两个参数：

* 第一个参数是Context(之所以用this是因为当前Activity是Context的子类)
* 接受系统发送Intent的应用组件的Class（在这个案例中，指将要被启动的activity）。

Android Studio会提示导入Intent类。

* 在文件开始处导入Intent类(Android Studio会自动导入):

java/com.leclex.startotheractivity/MainActivity.java

**import** android.content.Intent;

**Tip:**在Android Studio中，按Alt + Enter 可以导入缺失的类(在Mac中使用option + return)。

* 在sendMessage()方法里用[findViewById()](http://developer.android.com/reference/android/app/Activity.html" \l "findViewById(int))方法得到[EditText](http://developer.android.com/reference/android/widget/EditText.html)元素。

java/com.leclex.startotheractivity /MainActivity.java

**public** **void** **sendMessage**(View view) {

Intent intent = **new** Intent(**this**, DisplayMessageActivity.**class**);

EditText editText = (EditText) findViewById(R.id.edit\_message);

}

* 在文件开始处导入EditText类（Android Studio会自动导入）。

在Android Studio中，按Alt + Enter 可以导入缺失的类(在Mac中使用option + return)。

* 把EditText的文本内容关联到一个本地 message 变量，并使用putExtra()方法把值传给intent.

java/com.leclex.startotheractivity/MainActivity.java

**public** **void** **sendMessage**(View view) {

Intent intent = **new** Intent(**this**, DisplayMessageActivity.**class**);

EditText editText = (EditText) findViewById(R.id.edit\_message);

String message = editText.getText().toString();

intent.putExtra(EXTRA\_MESSAGE, message);

}

Intent可以携带称作 extras 的键-值对数据类型。 putExtra()方法把键名作为第一个参数，把值作为第二个参数。

* 在MainActivity类（class），定义EXTRA\_MESSAGE :

java/com.leclex.startotheractivity/MainActivity.java

**public** **class** **MainActivity** **extends** **ActionBarActivity** {

**public** **final** **static** String EXTRA\_MESSAGE = "com.leclex.startotheractivity.MESSAGE";

...

}

为让新启动的activity能查询extra数据。定义key为一个public型的常量，通常使用应用程序包名作为前缀来定义键是很好的做法，这样在应用程序与其他应用程序进行交互时仍可以确保键是唯一的。

* 在sendMessage()函数里，调用startActivity()完成新activity的启动，现在完整的代码应该是下面这个样子：

java/com.leclex.startotheractivity/MainActivity.java

**public** **void** **sendMessage**(View view) {

Intent intent = **new** Intent(**this**, DisplayMessageActivity.**class**);

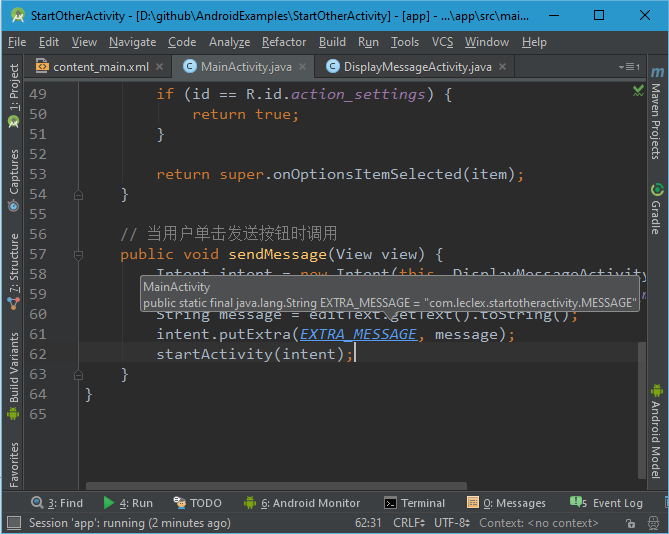
EditText editText = (EditText) findViewById(R.id.edit\_message);

String message = editText.getText().toString();

intent.putExtra(EXTRA\_MESSAGE, message);

startActivity(intent);

}



运行这个方法，系统收到我们的请求后会实例化在Intent中指定的Activity，现在需要创建一个DisplayMessageActivity类使程序能够执行起来。

* 创建DisplayMessageActivity。

Activity所有子类都必须实现onCreate()方法。创建activity的实例时系统会调用该方法，此时必须用 setContentView()来定义Activity布局，以对Activity进行初始化。

* **使用Android Studio创建新的Activity**

使用Android Studio创建的activity会实现一个默认的onCreate()方法.

* 在Android Studio的java 目录, 选择包名 **com.leclex.startotheractivity**,右键选择 **New > Activity > Blank Activity**.
* 在**Customize the Activity**窗口，配置activity：

Activity Name: DisplayMessageActivity

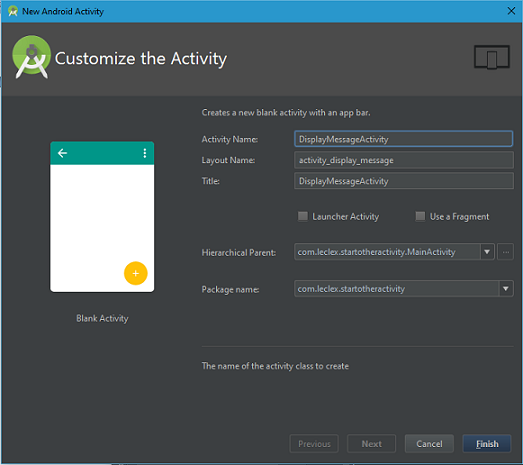
Layout Name: activity\_display\_message

Title: 我的消息

Hierarchical Parent: com.leclex.startotheractivity.MainActivity

Package name: com.leclex.startotheractivity

点击 **Finish**。



现在已经可以点击发送按钮启动这个activity了，但显示的是模板提供的默认空白内容，稍后修改显示自定义的文本内容。

* 接收Intent。

不管用户导航到哪，每个[Activity](http://developer.android.com/reference/android/app/Activity.html)都是通过[Intent](http://developer.android.com/reference/android/content/Intent.html)被调用的。我们可以通过调用[getIntent()](http://developer.android.com/reference/android/app/Activity.html" \l "getIntent())来获取启动activity的[Intent](http://developer.android.com/reference/android/content/Intent.html)及其包含的数据。

1 编辑java/com.leclex.startotheractivity目录下的DisplayMessageActivity.java文件。

2 删除onCreate()方法中下面一行:

setContentView(R.layout.activity\_display\_message);

3 得到intent 并赋值给本地变量。

Intent intent = getIntent();

4 为Intent导入包，Android Studio已自动导入。

在Android Studio中，按Alt + Enter 可以导入缺失的类(在Mac中使用option + return)。

5 调用 getStringExtra()提取从 MyActivity 传递过来的消息.

String message = intent.getStringExtra(MyActivity.EXTRA\_MESSAGE);

* 显示文本。

1 在onCreate() 方法中, 创建一个 TextView 对象.

TextView textView = new TextView(this);

2 设置文本字体大小和内容.

textView.setTextSize(40);

textView.setText(message);

3 通过调用activity的setContentView()把TextView作为activity布局的根视图.

setContentView(textView);

4 为TextView 导入包.

在Android Studio中，按Alt + Enter 可以导入缺失的类(在Mac中使用option + return).

DisplayMessageActivity的完整onCreate()方法应该如下：

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

Intent intent = getIntent();

String message = intent.getStringExtra(MyActivity.EXTRA\_MESSAGE);

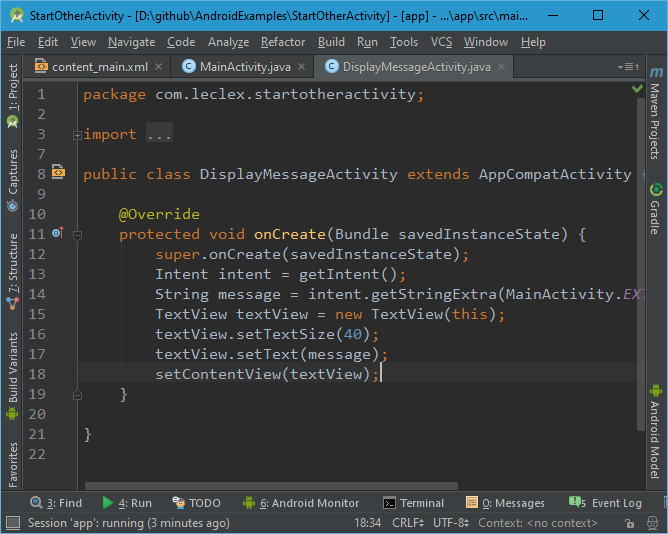
TextView textView = new TextView(this);

textView.setTextSize(40);

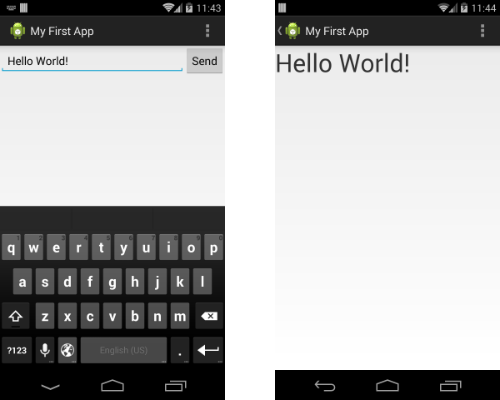
textView.setText(message);

setContentView(textView);

}



现在你可以运行app，在文本框中输入信息，点击发送按钮，好，现在就可以在第二Activity上看到发送过来的信息了。如图：



**摘要**

祝贺你，到此为止，已经创建好我们的启动其它的Activity应用了！