# 实验二：UI设计

## 实验目的

本次实验的目的是让大家熟悉Android开发中的UI设计，包括了解和熟悉常用控件的使用、界面布局和事件处理等内容。

## 实验要求

1. 熟悉和掌握界面控件设计
2. 了解Android界面布局
3. 掌握控件的事件处理

## 实验内容

### 一、常用控件

#### 1.常用控件介绍

Android中有许多常用控件（简单分类）：

文本框：TextView、EditText

按钮：Button、RadioButton、RadioGroup、CheckBox、ImageButton

列表：List、ExpandableListView、Spinner、AutoCompleteTextView、GridView、ImageView

进度条：ProgressBar、ProgressDialog、SeekBar、RatingBar

选择器：DatePicker、TimePicker

菜单：Menu、ContentMenu

对话框：Dialog、ProgressDialog

常用的控件有文本框、按钮和列表等。

#### 2.控件的实现

在上一个实验中，我们用TextView实现了在界面显示一句“Hello The Android World!”。

**public** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

TextView textView = **new** TextView(**this**);

textView.setText("Hello The Android World!");

setContentView(textView);

}

现在我们来看一看.xml文件和编码的相互关系。

在MainActivity.java中敲入如下代码：

**package** com.dingcheng365.ui;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***main\_layout***);  
 TextView textView = (TextView)findViewById(R.id.***textView01***);  
 Button button = (Button)findViewById(R.id.***button01***);  
 }  
}

注意在上面并没有为textView和button设置文本内容等，另外textView的初始化代码为：

TextView textView = (TextView)findViewById(R.id.*textView01*);

此处初始化用到了id为textView01的TextView。

在项目目录中找到/res/layout/main\_layout.xml，如下：

*<?***xml version="1.0" encoding="utf-8"***?>*<**RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/main\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 tools:context="com.dingcheng365.ui.MainActivity"**>  
  
 <**TextView  
 android:id="@+id/textView01"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/hello"** />  
  
 <**Button  
 android:id="@+id/button01"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/button"  
 android:layout\_marginTop="13dp"  
 android:layout\_below="@+id/textView01"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"** />  
</**RelativeLayout**>

注意上面列出了两个控件：TextView和Button，id分别为textView01和button01，它们的文本内容分别为@string/hello和@string/button。

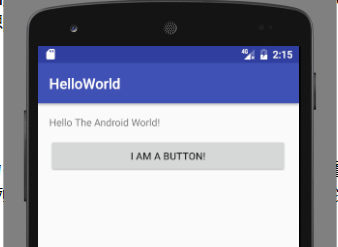
在项目目录中找到/res/values/strings.xml，如下：

<**resources**>  
 <**string name="hello"**>Hello The Android World!</**string**>  
 <**string name="button"**>I am a button!</**string**>  
 <**string name="app\_name"**>HelloWorld</**string**>  
</**resources**>

可以看到两个string：hello和button的内容分别为“Hello The Android World!”和“I am a button!”，即使文本框和按钮在界面上显示的内容，如下图所示：

#### 3.控件的属性

可以直接在类中为控件设置属性，也可以在xml文件中设置属性。每个控件都有一系列的属性，例如id、size、text、color等等，大家可以找找API查看控件的属性。



### 二、界面布局

Android中有很多布局：

* FrameLayout：最简单的一个布局对象。它里面只显示一个显示对象。Android屏幕元素中所有的显示对象都将会固定在屏幕的左上角，不能指定位置。但允许有多个显示对象，但后一个将会直接在前一个之上进行覆盖显示，把前一个部份或全部挡住（除非后一个是透明的）。
* LinearLayout：以单一方向对其中的显示对象进行排列显示，如以垂直排列显示，则布局管理器中将只有一列；如以水平排列显示，则布局管理器中将只有一行。同时，它还可以对个别的显示对象设置显示比例。
* TableLayout：以拥有任意行列的表格对显示对象进行布局，每个显示对象被分配到各自的单元格之中，但单元格的边框线不可见。
* AbsoluteLayout：允许以坐标的方式，指定显示对象的具体位置，左上角的坐标为(0, 0)，向下及向右，坐标值变大。这种布局管理器由于显示对象的位置定死了，所以在不同的设备上，有可能会出现最终的显示效果不一致。
* RelativeLayout：允许通过指定显示对象相对于其它显示对象或父级对象的相对位置来布局。如一个按钮可以放于另一个按钮的右边，或者可以放在布局管理器的中央。

布局中可以放置控件，而每个布局又可以嵌套其他布局，这种思想和之前学习java的布局是一样的。

例如在main.xml中如下布局：

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

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/main"

android:orientation="vertical"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

>

<LinearLayout

android:orientation="horizontal"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:layout\_weight="1">

<TextView

android:text="red"

android:gravity="center\_horizontal"

android:background="#aa0000"

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="1"/>

<TextView

android:text="green"

android:gravity="center\_horizontal"

android:background="#00aa00"

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="1"/>

<TextView

android:text="blue"

android:gravity="center\_horizontal"

android:background="#0000aa"

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="1"/>

<TextView

android:text="yellow"

android:gravity="center\_horizontal"

android:background="#aaaa00"

android:layout\_width="wrap\_content"

android:layout\_height="fill\_parent"

android:layout\_weight="1"/>

</LinearLayout>

<LinearLayout

android:orientation="vertical"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent"

android:layout\_weight="2">

<TextView

android:text="row one"

android:textSize="15pt"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"/>

<TextView

android:text="row two"

android:textSize="15pt"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"/>

<TextView

android:text="row three"

android:textSize="15pt"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"/>

<TextView

android:text="row four"

android:textSize="15pt"

android:layout\_width="fill\_parent"

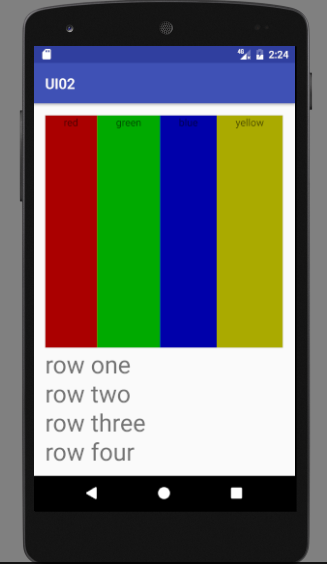
android:layout\_height="wrap\_content"

android:layout\_weight="1"/>

</LinearLayout>

</LinearLayout>

在一个LinearLayout布局下嵌套了两个LinearLayout，出来的效果如下图所示：



其他布局请大家自行尝试。

### 三、事情处理

事件监听和时间处理的概念大家以前都学习过，android中主要有如下事件方法：

* onClick(View v) 一个普通的点击按钮事件
* boolean onKeyMultiple(int keyCode,int repeatCount,KeyEvent event)用于在多个事件连续时发生，用于按键重复
* boolean onKeyDown(int keyCode,KeyEvent event) 用于在按键进行按下时发生
* boolean onKeyUp(int keyCode,KeyEvent event） 用于在按键进行释放时发生
* onTouchEvent(MotionEvent event)触摸屏事件，当在触摸屏上有动作时发生
* boolean onKeyLongPress(int keyCode, KeyEvent event)当长时间按时发生

下面以点击一个Button为例子：

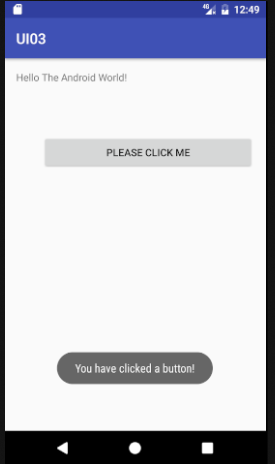
Main\_layout.xml

<**Button  
 android:text="please click me"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/textView"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_marginLeft="36dp"  
 android:layout\_marginStart="36dp"  
 android:layout\_marginTop="70dp"  
 android:onClick="buttton\_onClick"  
 android:id="@+id/button01"** />

MainActivity.java

**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***main\_layout***);  
  
 }  
 **public void** buttton\_onClick(View v) {  
 Toast.*makeText*(v.getContext(), **"You have clicked a button!"**,Toast.***LENGTH\_LONG***).show();  
 }  
}

为button添加了一个监听buttonListener，对应的onClick方法则抛出了一则文本框提示用户，效果图如下所示：



其他事件处理请大家自行尝试。