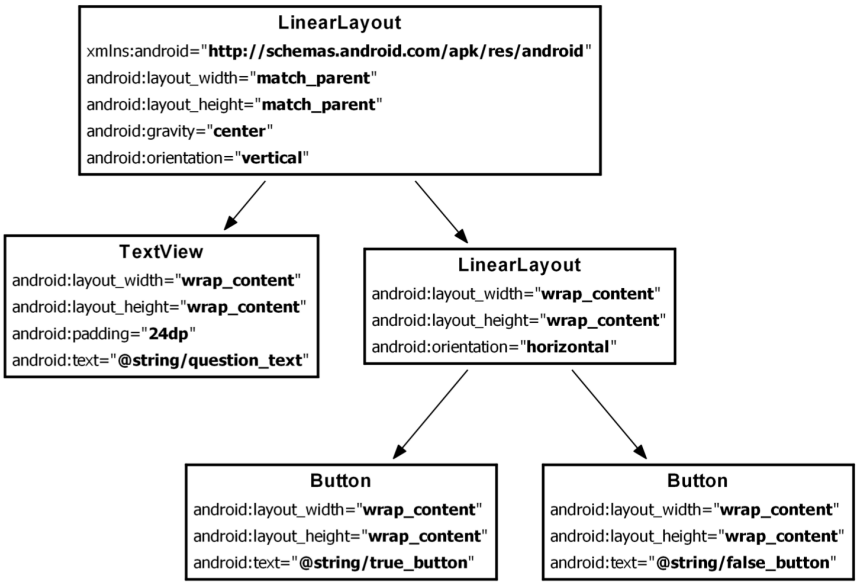
1 第一个用户界面

第一个用户界面如下：

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:gravity="center"
 android:orientation="vertical">

 <TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:padding="24dp"
  android:text="@string/question_text"
 />

 <LinearLayout
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:orientation="horizontal"
 >

  <Button
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="@string/true_button"
  />

  <Button
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="@string/false_button"
  />

 </LinearLayout>

</LinearLayout>
```

1.1 视图层级结构

上述的用户界面的视图层级结构如下：



对视图层级结构的解释：

1. 根元素是一个 LinearLayout 组建，根元素必须指定 Android XML 资源文件的命名空间属性为 http://schemas.android.com/apk/res/android。
2. LinearLayout 有两个子组件：TextView 和 LinearLayout。
3. 作为子组件的 LinearLayout 本身还有两个 Button 子组件。

1.2 组件属性

以下是常见的组件属性：

1. android:layout_width 和 android:layout_height。它们有两个常见的属性值：
   - match_parent，视图与其父视图大小相同。
   - wrap_content，视图将根据其展示的内容自动调整大小。
2. android:padding，用于告诉组件在决定大小时，除内容本身外，还需要增加额外指定量的空间。
3. android:orientation，用于决定组件的子组件是水平放置还是垂直放置，值有 vertical 和 horizon。
4. android:text，用于指定组件要显示的文字内容。它的属性值不是字符串值，而是对字符串资源的引用，字符串资源包含在一个独立的名为 strings 的 XML 文件中。

1.3 第一个用户界面的字符串资源

strings.xml 文件在 app/res/values 目录下，内容如下：

```
<resources>
    <string name="app_name">GeoQuiz</string>
    <string name="question_text">
        Constantionople is the largest city in Turkey.
    </string>
    <string name="true_button">TRUE</string>
    <string name="false_button">FALSE</string>
</resources>
```

1.4 从布局 XML 到视图对象

java 目录是项目全部 java 源代码的存放处，其中 AppCompatActivity 的子类用于把 XML 元素转换为视图对象：

```
package geoquiz.android.bignerdranch.com.geoquiz;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class QuizActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quiz);
    }
}
```

1.5 创建按键动作

1.5.1 资源与资源 ID

Android 项目所有资源存放在 app/res 的子目录下，R.java 文件用于记录项目用到的整个布局文件以及各个字符串的资源 ID，它存放在 app/build/generated/source/r/debug 目录下，只有在编译后才会产生。

需要知道的是，android 不会为布局文件中的组件生成资源 ID，我们需要通过添加 android:id 属性为组件生成资源 ID:

```
<LinearLayout ...>
 <TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:padding="24dp"
```

```
    android:text="@string/question_text"
  />

  <LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
  >

  <Button
    android:id="@+id/true_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/true_button"
  />

  <Button
    android:id="@+id/false_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/false_button"
  />

  </LinearLayout>
</LinearLayout>
```

1.5.2 通过资源 ID 使用组件

使用按钮组件有三个步骤:

1. 添加成员变量。
2. 通过资源 ID 引用生成的视图对象。
3. 为对象设置监听器，以响应用户的操作。

代码如下:

```
import android.widget.Button;

public class QuizActivity extends AppCompatActivity
{
    // 添加成员变量
    private Button mTrueButton;
    private Button mFalseButton;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_quiz);

```

```
15            // 通过资源ID
16            mTrueButton = (Button)findViewById(R.id.true_button);
17            mFalseButton = (Button)findViewById(R.id.false_button);
18        }
19    }
```

监听器用于等待某个特定时间的发生，是实现特定监听器接口的对象实例。Android SDK 为各种事件内置开发了很多监听器接口。

Android 中用于监听单击事件的监听器接口是 View.OnClickListener()，添加监听器操作如下：

```
1     public class QuizActivity extends AppCompatActivity
2     {
3         // 添加成员变量
4         private Button mTrueButton;
5         private Button mFalseButton;
6
7         @Override
8         protected void onCreate(Bundle savedInstanceState)
9         {
10            super.onCreate(savedInstanceState);
11            setContentView(R.layout.activity_quiz);
12
13            // 通过资源ID
14            mTrueButton = (Button)findViewById(R.id.true_button);
15            mFalseButton = (Button)findViewById(R.id.false_button);
16        }
17
18        mTrueButton.setOnClickListener(new View.OnClickListener(){
19            @Override
20            public void OnClick(View v)
21            {
22                Toast.makeText(QuizActivity.this, R.string.incorrect_toast, Toast.
                      LENGTH_SHORT).show();
23            }
24        });
25
26        mFalseButton.setOnClickListener(new View.OnClickListener(){
27            @Override
28            public void OnClick(View v)
29            {
30                Toast.makeText(QuizActivity.this, R.string.correct_toast, Toast.
                      LENGTH_SHORT).show();
31            }
32        })
33    }
```

当然还需要增加 toast 字符串：

```
1     <resources>
2      <string name="correct_toast">Correct!</string>
3      <string name="incorrect_toast">Incorrect!</string>
4     </resources>
```