



河北师范大学软件学院
Software College of Hebei Normal University



Android 基础开发

第二章 第一节 Android用户界面基础



Java与移动智能设备开发



教学目标

- 了解Android四大组件的功能
- 掌握Android中基本视图的使用及事件监听



目录

1 Android四大基本组件

2 用户界面的工作机制

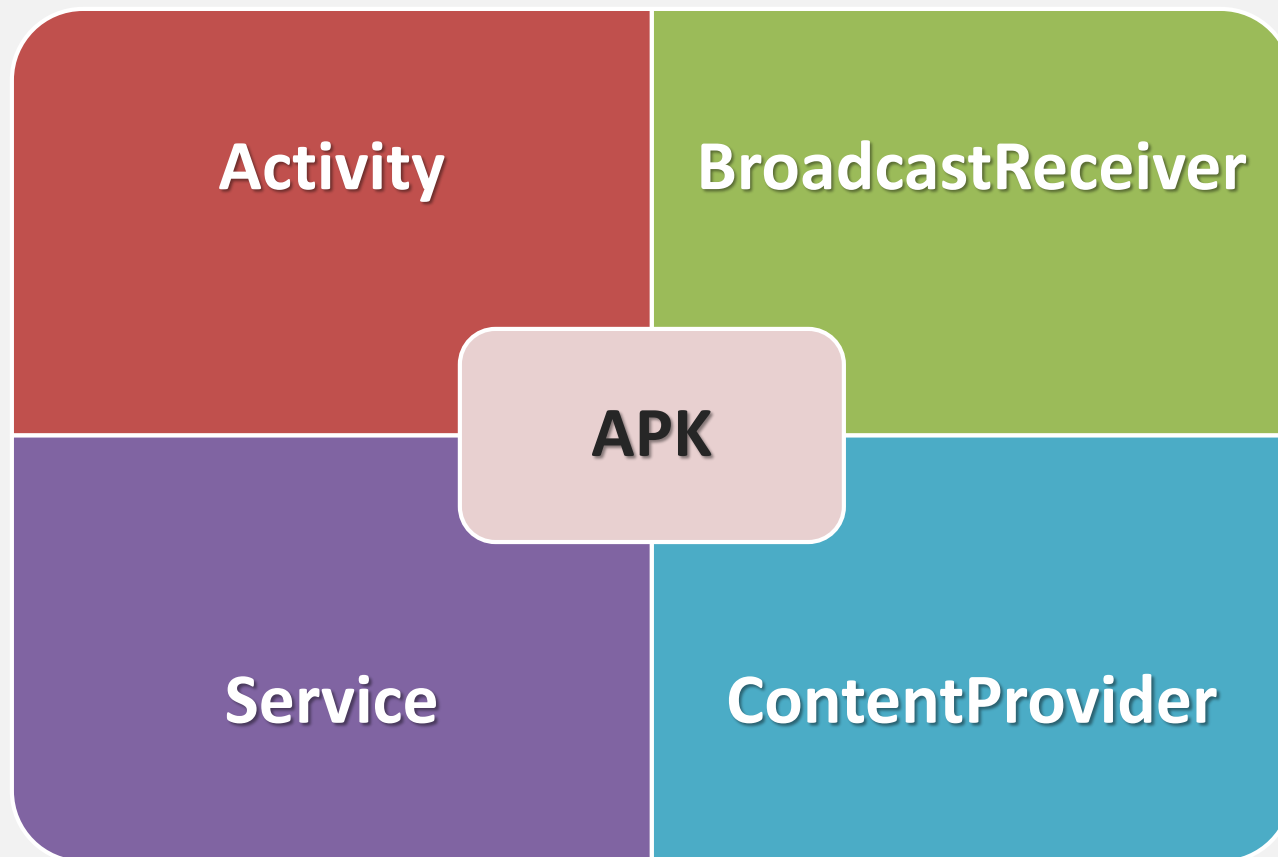
3 基本视图控件的使用

4 事件监听器的使用



Android四大基本组件

- 所有的Android应用都是有以下四大组件组成的：





Android四大基本组件



- 活动 (Activity)
 - Activity是最基本的Android应用程序组件，应用程序中，一个Activity通常就是一个单独的屏幕。每个Activity都被实现为一个独立的类，并且从Activity基类继承而来，Activity类会提供视图控制组件的用户接口，并对事件作出响应，大多数应用程序都是由多个Activity组成的。





Android四大基本组件

- 广播消息接收器（BroadcastReceiver）
 - BroadcastReceiver是Android系统中常用的一种机制，用户让应用对一个外部的事件作出响应。例如：当电话呼入时，数据网络可用时等。





Android四大基本组件

- 服务 (Service)
 - 一个服务是具有一个较长生命周期且没有用户界面的程序。例如：一个正在从播放列表中播放歌曲的媒体播放器。





Android四大基本组件

- 内容提供者 (ContentProvider)
 - 应用程序能够将它们的数据保存到文件、SQLite数据库中，甚至是任何有效的设备中。当需要将当前应用数据与其它应用共享时，ContentProvider类实现了一组标准方法，从而能够让其它的应用保存或读取此ContentProvider处理的各种数据类型。





目录



1 Android四大基本组件

2 用户界面的工作机制

3 基本视图控件的使用

4 事件监听器的使用

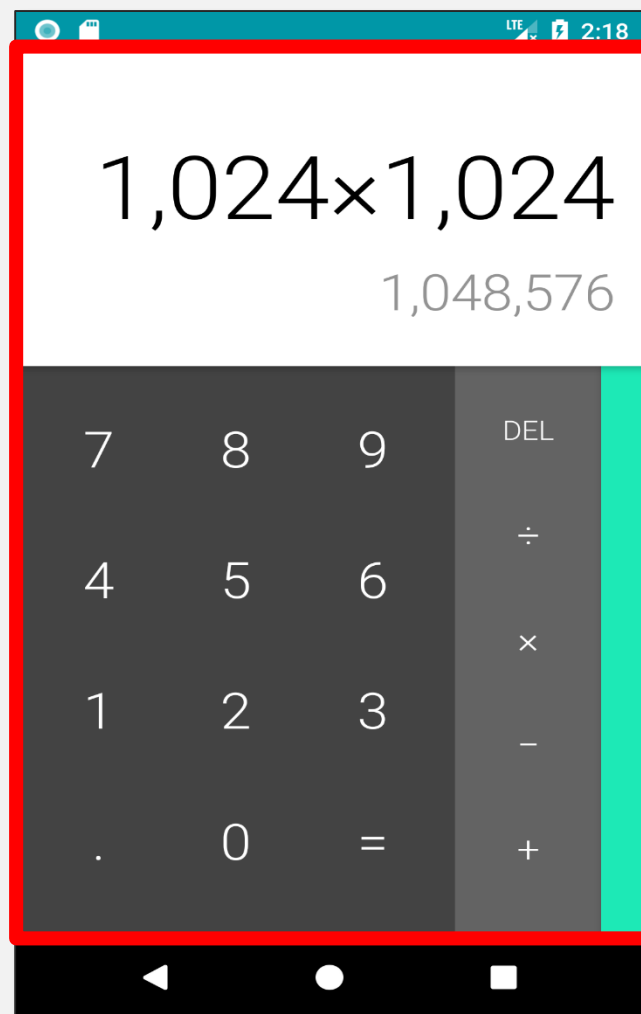


用户界面简介

- 在Android应用中，应用中每个界面对应一个 **Activity**，每个Activity由一个布局来决定如何显示，这就是（ **User Interface** ）。



用户界面简介





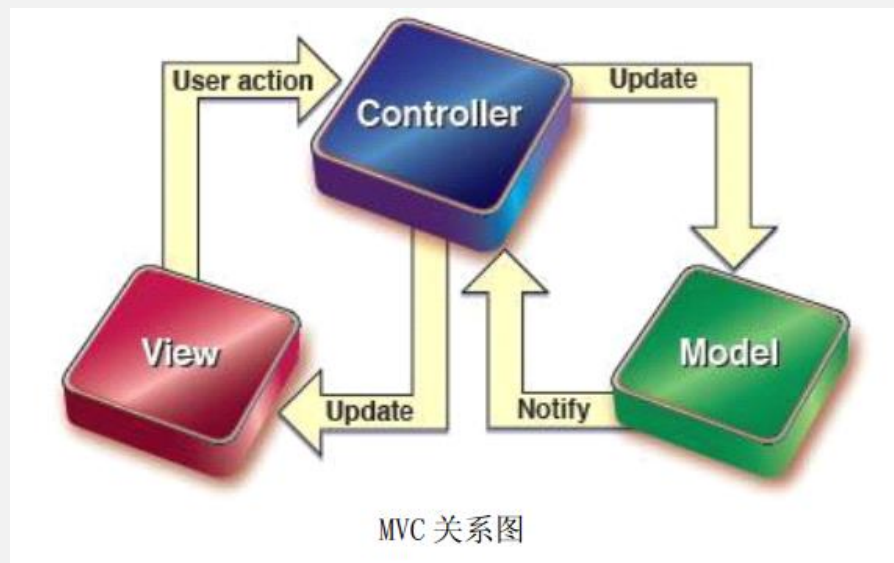
用户界面简介

- UI是人机之间传递、交换信息的接口；它实现了信息的内部形式与用户可接受形式之间的转换。
 - UI负责捕获用户动作，在程序中处理动作。
 - UI负责显示数据给用户。



Android中UI工作机制

- Android用户界面采用**MVC (Model-View-Controller) 框架**来接收用户动作、显示UI界面与及处理数据等工作。
 - 控制器：**处理用户的数据。**
 - 视图：**显示用户界面，与用户交互。**
 - 模型：**数据模型。**





Android中UI工作机制

- Android用户界面MVC模式
 - 模型层
 - 模型层负责对数据的操作、对网络服务等等的操作。
 - 在Android中，数据库/文件操作、ContentProvider、网络访问等等充当模型层。
 - 模型层后续会逐步介绍，在此不再赘述。



Android中UI工作机制

- Android用户界面MVC模式

- 控制器层

- 控制器负责接受用户动作请求（如按键动作或触摸屏动作等），调用指定模型处理用户请求（如读取数据库、发送网络请求等），响应用户结果（如返回视图界面等）。
 - 在Android系统中，控制器的责任由Activity承担，意味着Activity负责接收用户请求、调用模型方法、响应用户界面等操作（Activity不应承担过多业务逻辑（应交给模型层））。



Android中UI工作机制

- Android用户界面MVC模式
 - 视图层
 - 视图层主要负责用户界面（UI）的设计（页面布局XML文件等）。
 - 在Android中使用XML布局文件实现视图层和模型层的分离。

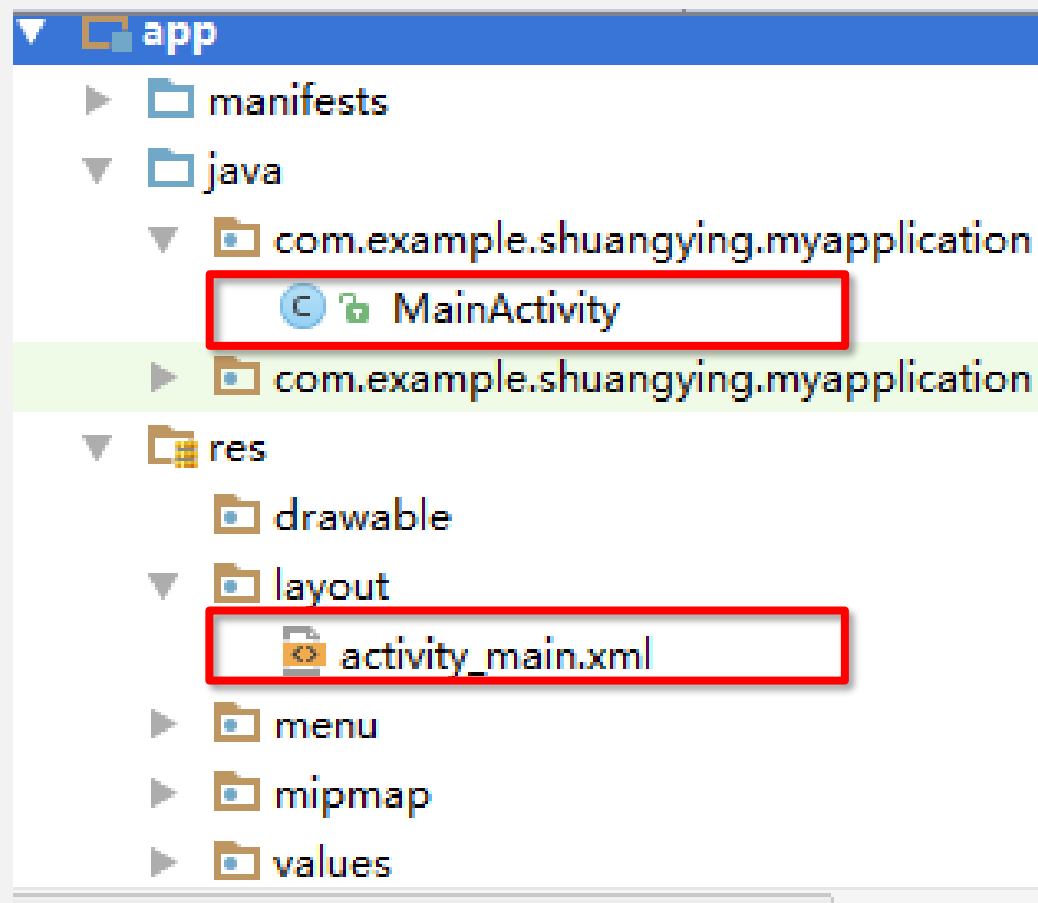


Android中如何实现MVC分离

- Android用户界面MVC模式

控制器层

视图层





Android中MVC如何整合到一起

- Android视图层与控制器层、模型层的整合
 - 在Activity文件，使用`setContentView()`方法，确定当前Activity如何显示。

```
public class MainActivity extends Activity {  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main);  
        //.....  
    }  
}
```

← 用户界面文件



Android界面原理

1. 分析Activity，编写布局
2. 在Activity中建立相应对象，设置属性
3. 在Activity中建立对象，设置相应监听器方法
4. 设计逻辑



目录



1 Android四大基本组件

2 用户界面的工作机制

3 基本视图控件的使用

4 事件监听器的使用

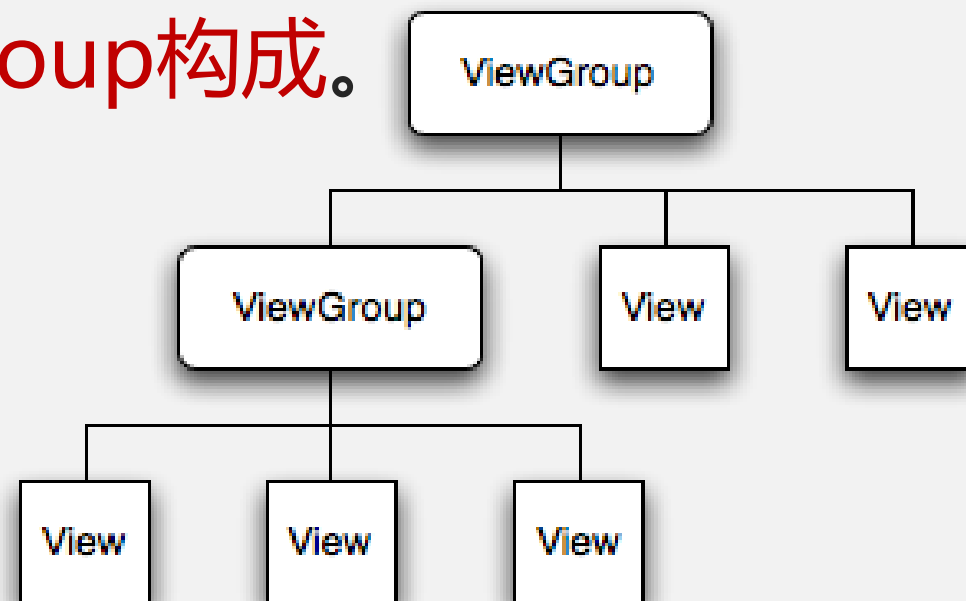


Android中视图层的使用

- Android视图层简介

- 视图层采用**视图树 (View Tree) 模型**：用户界面中的界面元素以树型结构组织在一起，整个视图界面为一个视图树模型。

- 视图树：**由View控件或ViewGroup构成。**



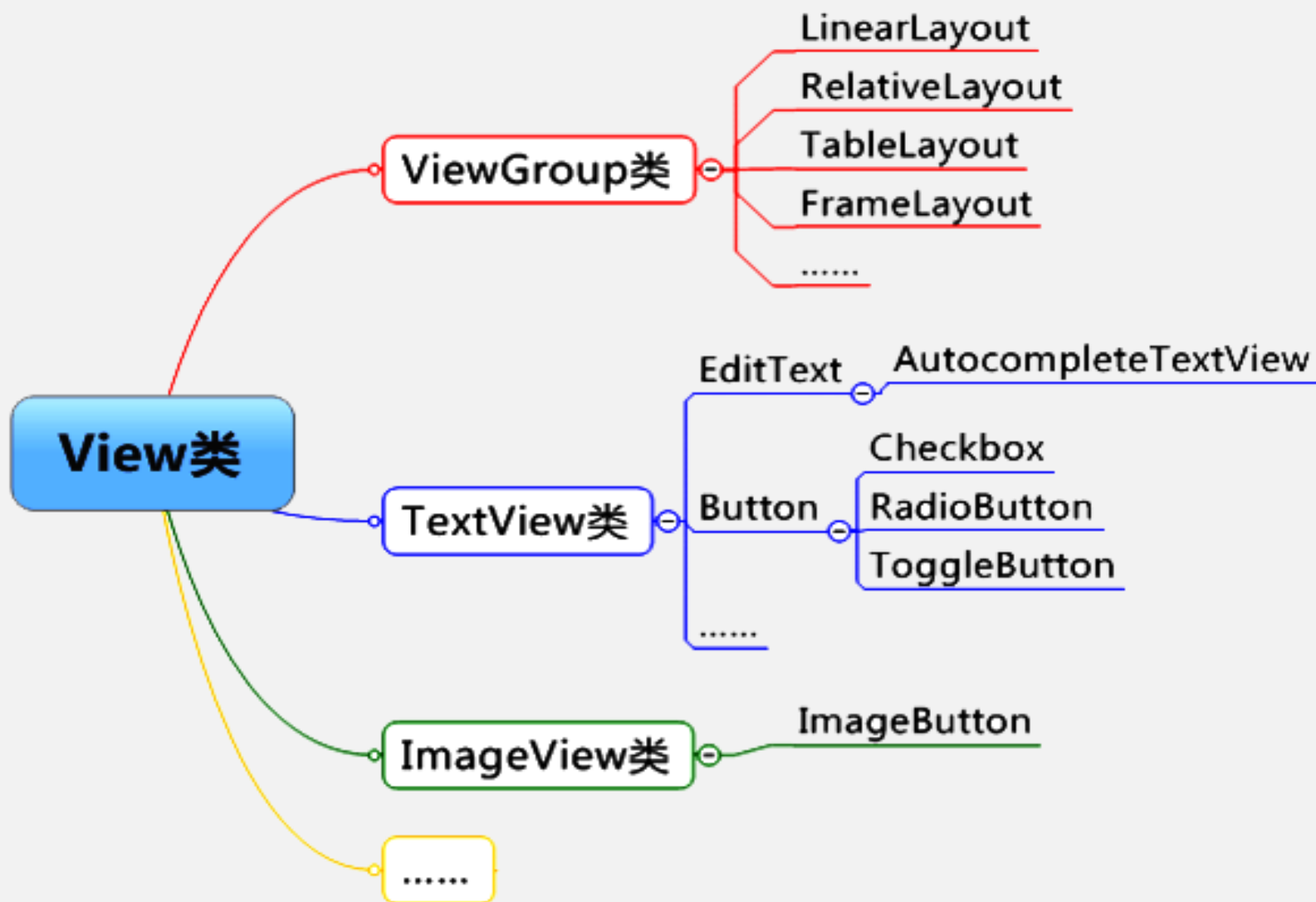


Android中视图层的使用

- Android视图层简介
 - View控件是界面的最基本的可视单元，是Android视图界面的基类。
 - 例如：文本（TextView）、输入框（EditText）.....
 - ViewGroup是由其它View或ViewGroup组成的显示单元，继承自View类。
 - ViewGroup功能：提供了一种布局方法，可以按照该布局定制视图的外观和顺序
 - 例如：LinerLayout、FrameLayout.....



View类及其子类的层次关系





在Android中创建视图界面

- Android中创建用户视图界面基本流程
 - 确定视图界面所采用的布局方式（ 暂用LinearLayout ）
 - 为视图界面添加视图组件



在Android中创建视图界面

- Android中创建布局有3种方法：
 - 使用可视化编辑方式，创建用户视图界面
 - 最简单的布局方式，但不适合创建复杂布局
 - 使用XML代码方式，创建用户视图界面
 - 最常用的布局方式，但只能创建静态界面
 - 使用Java代码方式，动态创建用户视图界面
 - 最灵活的布局方式，但复杂度较大
 - 使用findViewById()方法得到对象



在Android中创建视图界面



- 最基本的视图组件：

- TextView

- EditText

- RadioButton

- Checkbox

- Button

-

用户注册

用户： 请输入

密码： 请输入

性别：☒ 男 ☐ 女

爱好：☐ 音乐 ☐ 读书

☐ 运动 ☐ 游戏

注册 注册



使用基本的视图组件

- XML布局中，视图控件常用的公有布局属性有：

属性名	属性值	说明
android:background	图片资源或颜色值	控件的背景
android:id	@+id/字符串	控件的标识符（在程序中可以通过id获得该控件引用）
android:layout_width	match_parent（铺满父容器） wrap_content（由内容决定） 数据值（固定值）	控件在其父容器中的显示宽度
android:layout_height		控件在其父容器中的显示高度
android:layout_gravity	关键字	控件在其父容器中的相对位置
android:layout_margin	数值	控件在其父容器中与界面四周的页边距
android:padding	数值	控件四周的填充（内边距）

<http://developer.android.com/reference/android/view/View.html#lattrs>



Android常用的View

- TextView：显示一段文本内容
- EditText：显示接收用户输入的输入框

属性名	说明
gravity	TextView内文本对齐方式
text	TextView内文本显示的内容
hint	EditText内默认显示的提示文本
inputType	EditText内文本的格式
ellipsize	如果TextView中文本太长可以设置中间文本用省略号取代，取值center
autoLink	取值email、phone等，给文本中的email或者电话增加链接



Android常用的View

- RadioButton：单选按钮，用户只能在一组单选按钮中选择一个；使用时需要借助**RadioGroup**一起使用。
- CheckBox：多选框。

属性名	描述
orientation	RadioGroup的属性，设置其内部的RadioButton排列方式（水平或者垂直）
checked	RadioButton或者CheckBox的属性，设置此项是否为选中状态



Android常用的View

- Button : 按钮。
- ImageButton : 图片按钮。
- ToggleButton : 切换按钮。

属性名	描述
src	ImageButton的属性，设置图片的资源标识符
checked	ToggleButton的属性，设置是否为选中状态
textOn	ToggleButton的属性，当处于选中状态时显示的文本
textOff	ToggleButton的属性，当处于未被选中状态时显示的文本



Android中常用的View

- Pickers : 时间、日期的选择对话框。
 - 包括 : TimePicker、DatePicker、

属性名	描述
calendarTextColor	设置日历选择器上的文本颜色
calendarViewShown	是否显示日历视图true/false
minDate	设置日期选择器的最小日期格式为mm/dd/yyyy
maxDate	设置日期选择器的最大日期格式为mm/dd/yyyy



目录

1 Android四大基本组件

2 用户界面的工作机制

3 基本视图控件的使用

4 事件监听器的使用

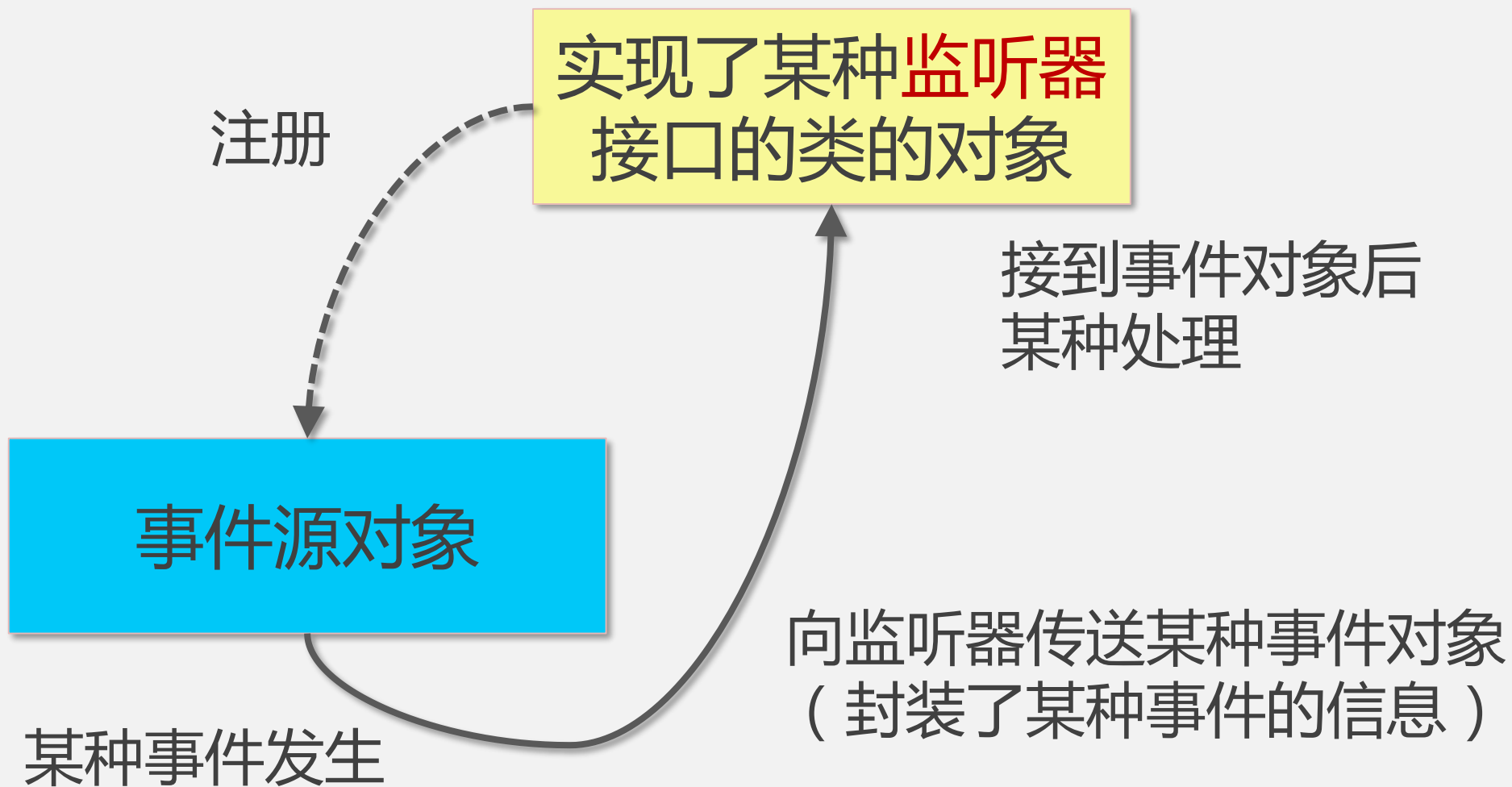


Android中常见的事件监听器

- 使用findViewById获取UI控件对应的对象后，不但可以设置相应属性，而且可以设置相应的事件监听
- 例如：
 - Button点击事件
 - 控件得到焦点，失去焦点事件
 - View的长按事件
 - 屏幕的触摸事件
 - 键盘事件等



事件监听器实现



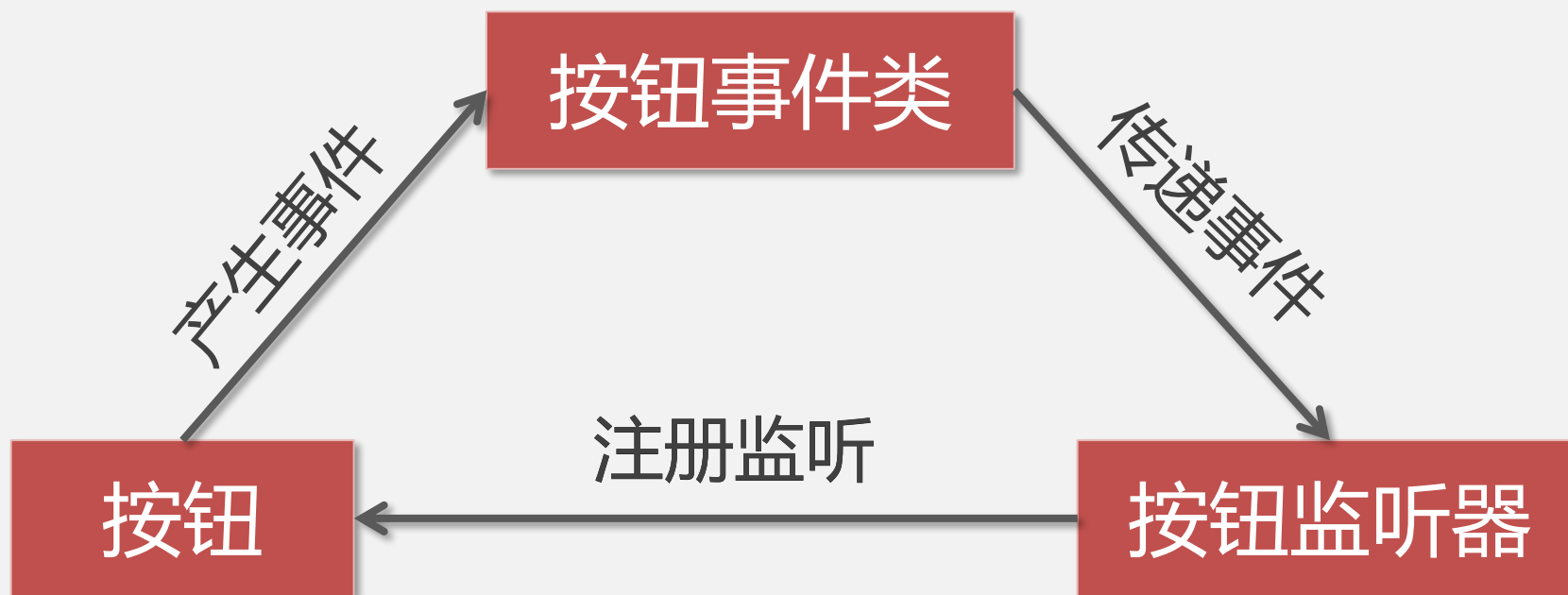


事件监听器实现

- 事件的处理步骤为：
 - 事件源上触发一个事件。比如用户按下鼠标、按下按钮等
 - 系统会自动产生对应的事件对象EventObject，并通知所有授权的事件监听者
 - 事件监听者中有对应的事件处理方法来处理该事件



事件监听器实现





基本视图控件的事件监听器

- 为视图控件绑定事件监听器的步骤
 - 获得视图控件对象
 - 设置事件监听类型
 - 绑定事件监听器

```
控件对象.setOn事件类型Listener(new On事件类型Listener() {  
    public 返回类型 on事件类型(View v, ...) {  
        // TODO Code Here  
    }  
});
```



基本视图控件的事件监听器

- Button控件的事件监听器

```
btn.setOnClickListener(new OnClickListener() {  
  
    @Override  
    public void onClick(View v) {  
        // TODO Auto-generated method stub  
    }  
});
```



视图控件的常用事件类型

- TextView控件
 - Click、LongClick、Touch、CreateContext、FocusChange、Key、
- EditText控件：继承父类 (TextView)
- Button：继承父类 (TextView)
- CheckBox/RadioGroup：继承TextView
 - CheckedChange
-

<http://developer.android.com/guide/topics/ui/ui-events.html#EventListeners>



为视图控件绑定事件监听器

- 实例：扩充用户注册实例，实现用户注册信息的数据校验
 - 当“用户名” / “密码” 输入完毕后校验用户输入信息，保证用户名/密码在6个字符以上，否则提示用户重新输入。



实例





实例





实例



13:29 ... 0.00K/s 0.00K/s 0.00K/s H +

基站导航



账号 填写用户名

密码 填写登陆密码

登录

[忘记密码?](#) [参数配置](#)





目录

1 Android四大基本组件

2 用户界面的工作机制

3 基本视图控件的使用

4 事件监听器的使用



Thank you!

