第一章

1、Android的优势：

（1）系统的开放性和免费性

（2）移动互联网的发展

（3）相关厂商的大力支持

挑战：

（1）用户体验不一致，安全性有待提高

（2）技术的进一步完善

（3）开放收集联盟模式的挑战

（3）其他技术的竞争

2、常见的手机操作系统：Symbian、Android、windows Mobile、IOS、OMS

3. Android 平台的技术架构

Android平台采用了软件栈(Software Stack)，又名软件叠层的架构，由低到高分为四部分:

(1) Linux 内核层:该层是基础，包含各种驱动，并提供操作系统的基本功能。

(2)中间层:该层包括程序库(Libraries)和Android运行时环境。

(3)应用程序框架:该层是编写核心应用所使用的API框架，开发者可以使用这些框架来开发自己的应用，但必须遵守该框架的开发原则。

(4)应用层:该层是各种应用软件，包括通话、短信、日历、地图、浏览器等核心应用程序，这些应用程序都是使用Java编写的。

3、一个android应用程序通常由4个组件构成：

（1）活动（Activity），是最基本的android应用程序组件

（2）意图（Intent），利用消息实现应用程序见的交互机制

（3）服务（Service），是android应用程序中具有较长的生命周期但是没有用户界面的程序。

（4）内容提供器（Content Provider），提供了一种多应用间数据共享的方式。

其中活动（Activity）是必要部分。

第二章：

课后习题

1、 Activity生命周期中的(OnCreate())方法用于Activity初次创建时被调用。

2、 (暂停)状态下的Activity失去了焦点，但是仍然对用户可见。

3、 Activity的可见生命周期是(从调用onStart()方法到onStop()方法的整个过程)

4、 Android程序不能直接访问的资源存放在(assets目录)下

5、 简述Activity的生命周期中的各种方法。

每个Activity类在定义是都必须继承android.app.Activity

（1）onCreate():Activity初次创建时被调用，在该方法中一般进行一些静态设置，如创建View视图，进行数据绑定。如果Activity是首次创建，本方法后将会调用onStart（），如果Activity是停止后重新显示，则调用onRestart（）。

（2）onStart（）：当Activity对用户即将可见的时候调用。

（3）onRestart():当Activity从停止状态重新启动时调用。

（4）onResume():当Activity将要与用户交互时调用此方法。

（5）onPause（）：当系统要启动一个其他的Activity是（其他的Activity显示之前），这个方法将被调用，用于提交持久数据的改变，停止动画等。

（6）onStop（）：当另外一个Activity恢复并遮盖住当前的Activity，导致其对用户不再可见时，这个方法被调用

（7）onDestroy（）：在Activity被销毁前所调用的最后一个方法。

1、Activity具有生命周期，共有四种状态

（1）激活或者运行状态。此时Activity运行在屏幕的前台

（2）暂停状态：此时Activity失去了焦点，但是仍然对用户可见

（3）停止状态：此时Activity被其他Activity覆盖

（4）终止状态：此时Activity将会被系统清理出内存

2、Activity的生命周期还可以根据不同标准分类：

（1）完整生命周期：从Activity最初调用onCreate()方法到最终调用onDestroy（）方法的过程。

（2）可见生命周期：从Activity调用onStart（）方法开始，到调用对应的onStop()方法

（3）前台生命周期：从Activity调用onResume()方法开始，到调用对应的onPause()方法。

3、LogCat视图：

Tag:日志标记

Msg:日志信息

Tr:异常信息

4、资源使用：

Res目录存放android程序能通过R资源类直接访问的资源。

Assets目录存放android程序不能直接访问的资源（原生文件），必须通过assermanager类以二进制流的形式读取。

5、对于字符串资源，在其他资源文件中使用“@string/资源名称”的形式访问，在代码中可通过“R.string.资源名称”访问

6、对于图片资源，在其他资源文件中使用“@drawable/资源名称”的形式访问，在代码中可通过“R.s drawable.资源名称”访问

第三章：

课后习题：

1、 下面不属于Android用户界面元素的是(资源引用R)

2、 (事件)不是通过new运算符创建的，而是由用户操作触发的

3、 Spinner是(下拉列表)组件。

4、 简述创建选项菜单和上下文菜单的步骤

创建选项菜单（Option Menu）步骤：

（1） 覆盖Activity的onCreateOptionsMenu()方法，当第一次打开菜单是，该方法被自动调用。

（2） 调用Menu的add()方法添加菜单项（MenuItem），可以调用MenuItem的setIcon()方法为菜单设置图标。

（3） 当菜单项被选择时，覆盖Activity的OptionsItemSelected（）方法来响应事件。

上下文菜单（Context Menu）步骤：

（1） 覆盖Activity的onCreateContextMenu（）方法，调用Menu的add（）方法添加菜单项（MenuItem）.

（2） 覆盖onContextItemSelected（）方法，响应菜单的单击事件。

（3） 在Activity的onCreate（）方法中，调用registerForContextMenu()方法，为视图注册上下文菜单。

本章重点：

1、用户界面元素分类

视图组件（View）

视图容器（ViewGroup）

布局管理（Layout）： LinearLayout（线性布局），RelativeLayout（相对布局），TableLayout（表格布局），AbsoluteLayout（绝对布局）

2、android系统中引用Java的事件处理机制，包括事件，事件源和事件监听器三个事件模型。

（1）事件（Event）：是一个描述事件源状态改变的对象，事件不是通过new运算符创建的，而是有用户操作触发的。

（2）事件源（Event Source）：即产生事件的对象，事件源通常是UI组件，

（3）事件监听器（Event Listener）:当事件产生是，事件监听器用于对该事件进行响应和处理。监听器需要事件监听接口中定义的事件处理方法。

2.1事件处理步骤：

（1）创建事件监听器

（2）在事件处理方法中编写事件处理代码

（3）在相应的组件上注册监听器

3、常用Widget组件：文本框（TextView），编辑框（EditText），复选框（CheckBox），单选按钮组（RadioGroup），下拉列表（Spinner），图片视图（ImageView），滚动视图（ScrollView）,网格视图（GridView），列表视图（ListView）

4、android中提供了两种创建布局的方式：XML布局文件和代码直接实现。

5、android系统中提供了4种对话框：AlertDialog，ProgressDialog，DatePickerDialog，TimePickerDialog。

6、创建提示对话框的步骤如下：

（1）获得AlertDialog的静态内部类Builder对象，由该类创建对话框。

（2）通过Builder对象设置对话框的标题、按钮以及按钮将要响应的事件。

（3）调用Builder的Create（）方法创建对话框

（4）调用AlertDialog的show()方法显示对话框。

7、创建Toast的步骤：

（1）调用Toast的静态方法makeText（）创建一个指定文本和时长的提示信息。

（2）调用Toast的show()方法显示提示信息。

提示信息（Toast）是android中用来显示提示信息的一种机制。与对话框不同，Toast是没有焦点的，而且Toast显示时间有限，超过一定的时间就会自动消失。

第四章

课后习题：

1、下列Intent的Action属性中，用来标识应用程序入口的是(ACTION\_MAIN)

2、Android系统提供了终端用户用于开发应用程序交互功能的组件，这些组件包括(广播接收器、意图、适配器、内容提供器)

3、下列关于启动Intent的说法，正确的是

Context.startActivity用于启动Activity

Context.startService()用于启动Service

Context.sendBroadcast()用于发送广播

4、Intent由动作、数据、分类、类型、组件、扩展信息等内容组成

5、简述Intent的过滤机制

Intent过滤器是用来通知系统它们能够处理哪种类型隐式的Intent对象，Activity、Service、Broadcast Receiver能够有一个或多个Intent过滤器。每个过滤器都描述了组件的一种能力，说明了组件将会接受的Intent对象集。它滤如有效的期望类型的Intent对象，滤出不想要的Intent对象。隐式的Intent对象仅能发送给能够通过组件的一个过滤器来传递它的一个组件。

一个Intent过滤器是IntentFilter类的实例。Intent过滤器通常都不是用Java代码来建立的，而是在应用程序的清单文件（AndroidManifest.xml）中用<intent-filter>元素来声明。

过滤器有类似于Intent对象的动作、数据、和分类的字段，过滤器会用这三个域来检测一个隐式的Intent对象。对于要传递给拥有过滤器的组件的Intent对象，必须传递所有的这三个要检测的字段。如果其中之一失败了，Android系统也不会把它发送给对应的组件---至少在基于那个过滤器的基础上不会发送。

6、简述Android广播机制的作用

Android广播机制指的是，可以通过实现广播接收器来监听和响应这些广播的Intent。通常广播Intent用于向监听器通知系统给事件或应用程序事件，从而扩展应用程序间的事件驱动的编程模型。广播Intent可以使应用程序更加开放，通过使用Intent来广播一个事件，可以在不用修改原始应用程序的情况下，对事件作出响应。Android中大量使用广播Intent来广播系统事件，如电池电量、网络连接和来电。

本章重点：

1、Intent由动作（Action），数据（Data），分类（Category）类型（Type）组件（Component）和扩展信息（Extra）

2、Android系统提供了终端用户用于开发应用程序交互功能的组件，广播接收器（Broadcast Receivers），意图（Intent）、适配器（Adapters）以及内容提供器（Content Providers）.

3、Action属性用于描述Intent要完成的动作，对要执行的动作进行一个简要描述。

4、Intent的Data属性是执行动作的URI和MIME类型。

5、Category属性指明一个执行Action的分类。

6、Component属性用于指明Intent的目标组件的类名称。

7、Extra属性用于添加一些附加信息。

8、多Activity的Android应用程序可通过startActivity（）方法制定相应的intent对象来启动另外一个Activity。

9、Intent的Extra属性用于添加一些附加信息，利用该属性可以进行消息的传递

10、将传递的信息存放到Extra属性中的两种方式：一种是直接将信息添加到Extra属性中，另一种是将数据封装到Bundle包中。

11、Intent可以通过显式方式或隐式方式找到目标组件，显式方式直接通过设置组件名来实现，而隐私方式则是通过Intent Filter过滤实现。

12、在AndroidMainfest.xml配置文件中，Intent Filter以<intent-filter>元素来指定，一个组件中可以有多个<intent-filter>元素，每个<intent-filter>元素描述不同的功能。

13、广播接收器（Broadcast Receiver）是接收广播消息，并对消息作出响应的组件。

14、Activity的sendBroadcast(),sendOrderBroadcast(),sendStrikyBoradcast()方法都可以广播Intent到广播接收器。

15.实现广播和接收Intent机制步骤:

(1)注册相应的广播接收器(Broadcast Receiver),广播接收器是接收广播消息并对消息作出反应的组件。

(2)发送广播，该过程将消息内容和用于过滤的信息封装起来，并广播给广播接收器。

(3)满足条件的广播接收器执行接收方onReceiver()。

(4)销毁广播接收器。

第五章

课后习题：

1. 下列不是Service特点的是：比Activity的优先级低

2. 关于启动、停止Service的说法，错误的是：Context.bindService()启动的Secvice可以调用Context.stopService()结束

3. Service生命周期方法有onCreate()、onStart()和onDestroy()

4. 简述Service的生命周期：

Servlet被服务器实例化后，容器运行其init方法，请求到达时运行其service方法，service方法自动派遣运行与请求对应的doXXX方法（doGet，doPost）等，当服务器决定将实例销毁的时候调用其destroy方法。

本章重点：

1、Service分为：本地服务和远程服务

2、Service的特点：

① 没有用户界面，不与用户交互

② 长时间运行，不占程序控制权

③ 比Activity的优先级高，不会轻易被Android系统终止，即使Service被系统终止，在系统资源恢复后，Service仍将自动运行

4 用于进程间通信，解决两个不同进程间的调用和通信问题

3、Service的实现

① 创建一个Service类并配置

② 启动或绑定Service

③ 停止Service

4、Service的创建

onCreate()方法：初始化Service，标志Service生命周期开始

onStart()方法：启动一个Service，代表Service进入了运行的状态

onDestroy()方法：释放Service所占用的资源，标志Service声明周期的结束

5、Service的两种启动方式：

启动方式：使用Context.startService()方法启动Service，调用者与Service之间没有关联，即使调用者退出，Service服务依然运行

绑定方式：通过Context.bindService()启动Service，调用者与Service之间绑定在一起，调用者一旦退出，Service服务也就终止。

第六章

课后习题：

适合结构化数据存储的是：SQLite

1. 可以存储为XML文件的存储方式是：Preference

2. 关于Android数据存储的说法，不正确的是：Android的文件存储无法使用标准Java(JavaSE)中的IO机制

3. 关于SQLite的说法，不正确的是：

SQLite不支持事务

SQLite只能用于Android系统

SQLite允许网络访问

4. Android的4种数据存储机制是：Preference、文件方式、SQLite、网络。

本章重点：

1.使用SharedPreference存储数据的步骤：

1、 使用getSharePreference()生成SharePreference对象。

2、 使用SharePreferences.Editor的putXXX()方法保存数据。

3、 使用SharedPreferences.Editor的commit()方法将上一步保存的数据写到XML文件中。

4、 使用SharedPreferences的getXXX()方法获取相应的数据。

2.SQLite的特点：轻量级、独立、便于管理和维护、可移植性、语言无关、事务性

3.数据库的操作：

(1) 创建会打开数据库 openDatabase()

(2) 删除数据库 android.content.Context.deleteDatabase()方法

(3) 关闭数据库 close()

4.记录操作：

(1) 插入记录：insert()和execSQL()

(2) 更新记录：update()和execSQL()

(3) 查询记录：query()

(4) 删除记录：delete()和execSQL()

5.在URL后拓展一个记录的ID：withAppendedId()和withAppendedPath()

6.使用Preference方式来存取数据时，需要用到SharedPrefences提供了获得数据的方法

SharedPreference.Editor编辑器提供保存数据的方法

7.File存储数据 文件操作

和在传统的Java中实现I/O的程序类似，通过Context.openFileInput()方法可以获取标准的文件输入流(FileInputStream)，以读取设备上的文件;通过Context.openFileOuput()方法可以获取标准的文件输出流(FileOutputStream),将数据写到文件中。