**Android移动应用开发基础教程（微课版）**

**习题答案**

**第1章**

1.Android平台具有哪些特点？

答：Android平台特点主要包括：开放性、不再受营运商限制、丰富的硬件选择、开发不受限制以及与Google应用无缝结合。

2.Android平台体系架构可分为哪些层？

答：Android平台体系架构可分为5层：Linux内核层、硬件抽象层、系统运行库层、Java API框架层和系统应用层。

3.在Android Studio中，可选择哪些设备来运行Android应用程序？

答：可选择在Android Studio中创建的模拟器或连接的物理设备来运行Android程序。

4.Android中，注册活动、内容提供器等组件的文件名是什么？

答：AndroidManifest.xml

5.可用哪些对象来输出日志信息？

答：可用System.out、System.err和Log对象在Android应用程序中输出日志信息。

**第2章**

1.请简述为一个活动绑定自定义视图的基本步骤。

答：为一个活动绑定自定义视图的基本步骤包括：

（1）创建项目。

（2）为项目添加布局资源文件。

（3）在活动的onCreate()方法中使用setContentView()方法来为活动绑定一个视图。

2.请问在一个活动中启动另一个活动的基本句法格式是什么？

答：在一个活动中启动另一个活动的基本句法格式如下：

startActivity(new Intent(context,class));

其中，context为当前活动上下文，class为另一个活动类。

3.Intent有哪些类型，这些类型之间有何区别？

答：Intent可分为显式Intent和隐式Intent。显式Intent指明了要启动的组件，隐式Intent则相反，它不指明要启动的组件，而是指明要执行的操作，让系统去选择可完成该操作的组件。

4.请简述向启动的活动中传递数据的基本过程。

答：向启动的活动中传递数据的基本过程包括下列几个步骤：

（1）创建用于启动活动的Intent对象

（2）调用Intent对象的putExtra()方法向Intent对象封装数据

（3）在启动对象后，首先执行getIntent()方法获取活动接收到的Intent对象，然后调用Intent对象的getStringExtra()等方法获得前一个活动传递过来的数据。

5.请问活动在其生命周期内可能有哪些状态？

答：活动在其生命周期内可能有4种状态：运行状态、暂停状态、停止状态和销毁状态。

6.请问活动在其生命周期内可能会调用哪些生命周期回调方法？

答：活动在其生命周期内可能会调用onCreate()、onStart()、onResume()、onPause()、onStop()、onRestart()和onDestroy()等方法。

7.请问活动有哪几种启动模式？

答：活动的启动模式有4种：standard、singleTop、singleTask和singleInstance。

**第3章**

1.简述视图（View）和视图组（ViewGroup）的联系和区别。

答：View类是所有用于设计界面组成元素的基类，Button、CheckBox、ExitView、ImageView、ProgressBar、TextView以及其他的UI组件，都是View类的子类或子类的派生类。

视图组是一种特殊的视图，它不具有可见性，而是一种容器。在视图组中可包含视图组和视图。ViewGroup类是View类的一个子类，它又是各种布局类的基类。常用的布局类有LinearLayout（线性布局）、RelativeLayout（相对布局）和FrameLayout（帧布局）类等。

2.列举5种基本的Android控件及其基本作用。

答：TextView控件用于显示文本，EditText控件用于输入文本，CheckBox控件用于创建复选框。RadioButton控件用于创建单选按钮，RadioGroup是RadioButton的容器，用于单选按钮分组，每组中的单选按钮只能选中一项。

3.简述如何使用Toast。

答：Toast是在应用运行期间，通过类似于对话框的方式向用户显示消息提示。Toast只占用很少的屏幕，并会在一段时间后自动消失。

在使用时，首先创建Toast，然后调用show()方法显示Toast。

4.简述如何使用Notification。

Notification用于创建通知区域通知，展开抽屉式通知栏可查看通知的详细信息。一个通知通常由图标、标题和内容等组成。

可按下列步骤使用Notification。

第一步：创建NotificationCompat.Builder对象。

第二步：调用NotificationCompat.Builder对象方法设置通知相关内容。

第三步：创建在抽屉式通知栏中单击通知时启动活动的Intent。

第四步：创建Notification对象。

第五步：创建NotificationManager对象显示通知。

5.简述如何使用RecyclerView。

答：RecyclerView用于创建滚动列表，并可实现垂直或水平滚动，甚至可实现多行或多列列表。要使用RecyclerView控件，首先需要在app\buil.gradle文件的dependencies闭包中添加支持库。

RecyclerView在使用时包含下列基本步骤：

第一步：准备列表显示的资源（字符串、图片）等。

第二步：在布局或代码中添加RecyclerView控件。

第三步：设置RecyclerView控件布局管理器类型。

第四步：定义RecyclerView适配器，完成用资源填充RecyclerView列表。

6.修改本章3.8实例，使用对话框显示登录验证信息。

只需将MainActivity.java中的：

if(isLoged){

//执行登录成功操作

tvShow.setText("用户名和密码正确，登录成功！");

}else {

//执行登录失败操作

tvShow.setText("用户名或密码错误，请重新登录！");

}

修改为：

AlertDialog.Builder dialog=new AlertDialog.Builder(MainActivity.this);

dialog.setTitle("系统登录");

String msg;

if(isLoged){

msg="用户名和密码正确，登录成功！";

}else {

msg="用户名或密码错误，请重新登录！";

}

dialog.setMessage(msg);

dialog.show();

**第4章**

1.Android中广播可分哪些类型，分别有什么特点？

答：Android中的广播可分为两种类型：标准广播和有序广播。

标准广播在发出后，所有接收器均可接收到广播消息。各个接收器之间没有先后顺序之分。标准广播发出后，不可能被中断。有序广播：有序广播在发出后，同一时间只有优先级较高的一个接收器接收到广播消息。只有在优先级较高的接收器处理完广播消息后，广播才能继续向优先级较低的接收器继续传递。在当前接收器中，可中断广播，使后继接收器无法收到广播消息。

2.简述一个广播接收器类的基本结构。

答：一个广播接收器类的基本结构如下：

public class 子类名称 extends BroadcastReceiver {

public BootReceiver() {

}

@Override

public void onReceive(Context context, Intent intent) {

//广播消息出来逻辑

}

}

3.简述如何注册广播接收器。

答：广播接收器可以采用静态和动态方式进行注册。

静态注册是指在应用程序的清单文件AndroidManifest.xml中添加广播接收器的注册信息。动态注册是指在程序中用registerReceiver()方法进行注册。

4.简述如何使用有序广播。

答：有序广播接收器可在AndroidManifest.xml中使用<intent-filter>标签的android:priority属性来设置广播接收器优先级。在动态注册时可调用IntentFilter的setPriority()方法来设置接收器优先级。

发送有序广播需要调用sendOrderedBroadcast()方法，在处理有序广播消息时，可使用abortBroadcast()方法来终止广播。

**第5章**

1.简述如何访问一个内部存储文件。

答：

向一个内部存储文件写入数据的基本步骤如下：

（1）调用Context类的openFileOutput()方法打开指定文件。

（2）将openFileOutput()方法返回的FileOutputStream对象封装为OutputStreamWriter对象。

（3）使用OutputStreamWriter对象将字符串存入文件。

从一个内部存储文件读出数据的基本步骤如下：

（1）调用Context类的openFileInput()方法打开指定文件。

（2）将openFileIntput()方法返回的FileInputStream对象封装为InputStreamReader对象。

（3）使用InputStreamReader对象从文件读取char数组。

2.简述如何访问一个外部存储文件。

答：访问外部存储文件与访问内部存储文件方法类似，不同之处在于：（1）需要为应用申请READ\_EXTERNAL\_STORAGE（读）或WRITE\_EXTERNAL\_STORAGE（写）权限，并在设备中允许应用访问存储器。（2）调用Environment.getExternalStorageDirectory()方法获得外部存储路径，然后调用File对象构造方法打开文件。

打开文件后，后继文件操作与访问内部存储文件相同。

3.简述如何使用SharedPreferences共享存储。

答：SharedPreferences共享存储采用键值对方式读写SharedPreferences文件。保存SharedPreferences数据基本步骤如下：

（1）获得SharedPreferences对象。

（2）获得SharedPreferences对象的Editor对象。

（3）调用Editor对象的方法向文件添加数据。

（4）提交数据，完成数据存储操作。

读取SharedPreferences数据只需调用SharedPreferences对象相应方法即可。

4.简述访问SQLite数据库的基本步骤。

答：访问SQLite数据库的基本步骤如下：

（1）创建SQLiteOpenHelper的子类，并实现onCreate()和OnUpgrade()方法。

（2）调用getWritableDatabase()和getReadableDatabase()方法打开数据库，获得SQLiteDatabase对象。

（3）调用SQLiteDatabase对象的insert()、update()、delete()和query()等方法完成数据的添加、更新、删除和查询等操作。

也可调用SQLiteDatabase对象的execSQL(）和rawQuery()执行SQL命令完成各种数据库操作。

**第6章**

1.简述使用SoundPool播放音效的基本步骤。

答：基本步骤如下：

第1步：使用SoundPool.Builder或SoundPool构造方法创建SoundPool对象。

第2步：调用load方法加载音频资源。

第3步：调用play、pause、stop等方法。

2.简述使用MediaPlayer播放音效的基本步骤。

答：基本步骤如下：

第1步：创建MediaPlay对象。

第2步：调用setDataSource方法设置音频文件路径。

第3步：调用prepare方法加载音频。

第4步：调用start、pause、stop、reset等方法。

3.简述使用MediaRecorder记录声音的基本步骤。

答：基本步骤如下：

第1步：在应用程序的清单文件AndroidManifest.xml添加RECORD\_AUDIO权限申请使用麦克风。

第2步：检查并动态申请用户授权。

第3步：创建MediaRecorder对象。

第4步：调用相关方法设置音频来源、输出格式、输出文件以及音频编码方式等信息。

第5步：调用prepare方法准备MediaRecorder。

第6步：调用start方法开始采集音频。

第7步：调用stop方法结束采集音频。

4.简述使用摄像头拍照的基本步骤。

答：基本步骤如下：

第1步：创建用于保存所拍照片的文件。

第2步：创建调用系统摄像头拍照程序的Intent对象。

第3步：执行startActivityForResult方法启动系统摄像头拍照程序。

第4步：在onActivityResult方法中处理系统摄像头拍照程序返回信息。

**第7章**

1.简述使用HttpURLConnection完成HTTP请求的基本步骤。

答：

第1步：创建HttpURLConnection实例对象。

第2步：调用setRequestMethod设置HTTP请求方法。

第3步：设置请求相关参数。

第4步：从InputStream对象中获取HTTP请求返回结果。

2.简述使用OkHttp完成HTTP请求的基本步骤。

答：

第1步：需要修改项目的app/build.gradle文件，添加OkHttp库编译信息。

第2步：创建OkHttpClient对象。

第3步：创建Request对象。

第4步：调用Request对象的execute()方法执行请求。

第5步：从Response对象中获得字符串形式的返回结果。

3.简述使用DOM解析XML的基本步骤。

答：

第1步：创建DocumentBuilderFactory对象。

第2步:创建DocumentBuilder对象。

第3步：将XML文档封装到InputSource对象中。

第4步：使用DocumentBuilder对象解析InputSource获得表示XML文档的Document对象。

第5步：调用Document对象的相关方法获取XML文档各个节点及其文本。

**第8章**

1.请说明如何创建线程。

答：有3中方法可以创建线程。

方法1：使用匿名类。例如：

new Thread(new Runnable() {

@Override

public void run() {

//在此编写线程功能代码

}

}).start();

方法2：先创建一个类来实现Runnable接口。例如：

class MyThread implements Runnable{

@Override

public void run() {

//在此编写线程功能代码

}

}

然后，按照下面的方式来启动线程。

new Thread(new MyThread()).start();

方法3：先定义一个类继承内置的Thread类来实现线程功能。例如：

class MyThread extends Thread{

@Override

public void run() {

//在此编写线程功能代码

}

}

然后，按照下面的方式来启动线程。

new MyThread().start();

2.请问能够在子线程中改变界面中TextView控件的文本吗？

答：不能直接在子线程中改变界面中TextView控件的文本。可以执行runOnUiThread()方法返回UI线程，或者通过Handler和Message对象来完成。

3.请问AsyncTask的哪些方法在主线程中执行，哪些方法在子线程中执行？

答：onPreExecute()、onPostExecute ()和onProgressUpdate ()方法在主线程中执行。doInBackground()方法在子线程中执行。

4.请问如何定义一个服务类。

答：定义一个服务类时，需要继承Service，并重写相应方法。基本结构如下：

public class MyService extends Service {

public MyService() {

}

@Override

public IBinder onBind(Intent intent) {

// TODO: Return the communication channel to the service.

throw new UnsupportedOperationException("Not yet implemented");

}

public void onCreate() {

super.onCreate();

}

@Override

public int onStartCommand(Intent intent, int flags, int startId) {

return super.onStartCommand(intent, flags, startId);

}

@Override

public void onDestroy() {

super.onDestroy();

}

}