# 71 道经典 Android 面试题和答案,重要知识点都包含了

**1.　　下列哪些语句关于内存回收的说明是正确的? (b )**

A、 程序员必须创建一个线程来释放内存

B、内存回收程序负责释放无用内存

C、内存回收程序允许程序员直接释放内存

D、内存回收程序可以在指定的时间释放内存对象

**2. 下面异常是属于 Runtime Exception 的是（abcd）(多选)**

A、ArithmeticException

B、IllegalArgumentException

C、NullPointerException

D、BufferUnderflowException

**3. Math.round(11.5)等于多少(). Math.round(-11.5)等于多少(c).**

A、11 ,-11　B、11 ,-12　C、12 ,-11　D、12 ,-12

**4.　　下列程序段的输出结果是：（b ）**

```
void complicatedexpression_r(){
int x=20, y=30;
boolean b;
b=x>50&&y>60||x>50&&y<-60||x<-50&&y>60||x<-50&&y<-60;
System.out.println(b);
}
```

**A、true B、false C、1 D、011.activity**

**5. 对一些资源以及状态的操作保存，最好是保存在生命周期的哪个函数中进行(d)**

A、onPause() B、onCreate() C、 onResume() D、onStart()

**6. Intent 传递数据时,下列的数据类型哪些可以被传递（abcd）(多选)**

A、Serializable B、charsequence C、Parcelable D、Bundle

**7. android 中下列属于 Intent 的作用的是(c)**

A、实现应用程序间的数据共享

B、是一段长的生命周期，没有用户界面的程序，可以保持应用在后台运行，而不会因为切换页面而消失

C、可以实现界面间的切换，可以包含动作和动作数据，连接四大组件的纽带

D、处理一个应用程序整体性的工作

**8. 下列属于 SAX 解析 xml 文件的优点的是(b)**

A、将整个文档树在内存中，便于操作，支持删除，修改，重新排列等多种功能

B、不用事先调入整个文档，占用资源少

C、整个文档调入内存，浪费时间和空间

D、不是长久驻留在内存，数据不是持久的，事件过后，若没有保存数据，数据就会消失

**9. 下面的对自定 style 的方式正确的是（a）**

A、 <resources>

<style name="myStyle">

<itemname="android:layout_width">fill_parent</item>

</style>

</resources>

B、 <style name="myStyle">

<itemname="android:layout_width">fill_parent</item>

</style>

C、 <resources>

<itemname="android:layout_width">fill_parent</item>

</resources>

D、 <resources>

<stylename="android:layout_width">fill_parent</style>

</resources>

**10. 在 android 中使用 Menu 时可能需要重写的方法有（ac）。(多选)**

A、onCreateOptionsMenu()

B、onCreateMenu()

C、onOptionsItemSelected()

D、onItemSelected()

**11. 在 SQL Server Management Studio 中运行下列 T-SQL 语句，其输出值（c）。** SELECT @@IDENTITY

A、 可能为 0.1

B、 可能为 3

C、 不可能为-100

D、 肯定为 0

**12.　在 SQL Server 2005 中运行如下 T-SQL 语句，假定 SALES 表中有多行数据，执行查询之　后的结果是（d）。**

```
BEGIN TRANSACTION A
    Update SALES Set qty=30 WHERE qty<30
     BEGIN TRANSACTION B
        Update SALES Set qty=40 WHEREqty<40
        Update SALES Set qty=50 WHEREqty<50
        Update SALES Set qty=60 WHEREqty<60
     COMMIT    TRANSACTION B
     COMMIT TRANSACTION A
```

A、SALES 表中 qty 列最小值大于等于 30

B、SALES 表中 qty 列最小值大于等于 40

C、SALES 表中 qty 列的数据全部为 50

D、SALES 表中 qty 列最小值大于等于 60

**13.　在 android 中使用 SQLiteOpenHelper 这个辅助类时，可以生成一个数据库，并可以对数据库版本进行管理的方法可以是(ab)**

A、getWriteableDatabase()

B、getReadableDatabase()

C、getDatabase()

D、getAbleDatabase()

**14.　android 关于 service 生命周期的 onCreate()和 onStart()说法正确的是(ad)(多选题)**

A、当第一次启动的时候先后调用 onCreate()和 onStart()方法

B、当第一次启动的时候只会调用 onCreate()方法

C、如果 service 已经启动，将先后调用 onCreate()和 onStart()方法

D、如果 service 已经启动，只会执行 onStart()方法，不在执行 onCreate()方法

**15. 下面是属于 GLSurFaceView 特性的是(abc)(多选)**

A、管理一个 surface，这个 surface 就是一块特殊的内存，能直接排版到 android 的视图

view 上。

B、管理一个 EGL display，它能让 opengl 把内容渲染到上述的 surface 上。

C、让渲染器在独立的线程里运作，和 UI 线程分离。

D、可以直接从内存或者 DMA 等硬件接口取得图像数据

**16. 下面在 AndroidManifest.xml 文件中注册 BroadcastReceiver 方式正确的(a)**

A、<receiver android:name="NewBroad">

<intent-filter>

<action

android:name="android.provider.action.NewBroad"/>

<action>

</intent-filter>

</receiver>

B、<receiver android:name="NewBroad">

<intent-filter>

android:name="android.provider.action.NewBroad"/>

</intent-filter>

</receiver>

C、<receiver android:name="NewBroad">

<action

android:name="android.provider.action.NewBroad"/>

<action>

</receiver>

D、<intent-filter>

<receiver android:name="NewBroad">

<action>

android:name="android.provider.action.NewBroad"/>

<action>

</receiver>

</intent-filter>

**17. 关于 ContenValues 类说法正确的是(a)**

A、他和 Hashtable 比较类似，也是负责存储一些名值对，但是他存储的名值对当中的

名是 String 类型，而值都是基本类型

B、他和 Hashtable 比较类似，也是负责存储一些名值对，但是他存储的名值对当中的

名是任意类型，而值都是基本类型

C、他和 Hashtable 比较类似，也是负责存储一些名值对，但是他存储的名值对当中的

名，可以为空，而值都是 String 类型

D、他和 Hashtable 比较类似，也是负责存储一些名值对，但是他存储的名值对当中

的名是 String 类型，而值也是 String 类型

**18.　我们都知道 HanIder 是线程与 Activity 通信的桥梁,如果线程处理不当，你的机器就会变得越慢，那么线程销毁的方法是(a)**

A、onDestroy()

B、onClear()

C、onFinish()

D、onStop()

**19.　下面退出 Activity 错误的方法是（c）**

A、finish()

B、抛异常强制退出

C、System.exit()

D、onStop()

**20.　下面属于 android 的动画分类的有(ab)(多项)**

A、Tween　B、Frame C、Draw D、Animation

**21.　下面关于 Android dvm 的进程和 Linux 的进程,应用程序的进程说法正确的是(d)**

A、DVM 指 dalivk 的虚拟机.每一个 Android 应用程序都在它自己的进程中运行,不一定拥有一个独立的 Dalvik 虚拟机实例.而每一个 DVM 都是在 Linux 中的一个进程,所以说可以认为是同一个概念.

B、DVM 指 dalivk 的虚拟机.每一个 Android 应用程序都在它自己的进程中运行,不一定拥有一个独立的 Dalvik 虚拟机实例.而每一个 DVM 不一定都是在 Linux 中的一个进程,所以说不是一个概念.

C、DVM 指 dalivk 的虚拟机.每一个 Android 应用程序都在它自己的进程中运行,都拥有一个独立的 Dalvik 虚拟机实例.而每一个 DVM 不一定都是在 Linux 中的一个进程,所以说不是一个概念.

D、DVM 指 dalivk 的虚拟机.每一个 Android 应用程序都在它自己的进程中运行,都拥有一个独立的 Dalvik 虚拟机实例.而每一个 DVM 都是在 Linux 中的一个进程,所以说可以认为是同一个概念.

**22. Android 项目工程下面的 assets 目录的作用是什么 b**

A、放置应用到的图片资源。

B、主要放置多媒体等数据文件

C、放置字符串，颜色，数组等常量数据

D、放置一些与 UI 相应的布局文件，都是 xml 文件

**23. 关于 res/raw 目录说法正确的是(a)**

A、这里的文件是原封不动的存储到设备上不会转换为二进制的格式

B、这里的文件是原封不动的存储到设备上会转换为二进制的格式

C、这里的文件最终以二进制的格式存储到指定的包中

D、这里的文件最终不会以二进制的格式存储到指定的包中

**24. 下列对 android NDK 的理解正确的是(abcd )**

A、 NDK 是一系列工具的集合

B、 NDK 提供了一份稳定、功能有限的 API 头文件声明。

C、 使“Java+C”的开发方式终于转正，成为官方支持的开发方式

D、 NDK 将是 Android 平台支持 C 开发的开端

## 二．填空题

**25.** **android 中常用的四个布局是 framlayout，linenarlayout，relativelayout 和 tablelayout。**

**26.** **android 的四大组件是 activiey，service，broadcast 和 contentprovide。**

**27.** **java.io 包中的 objectinputstream 和 objectoutputstream 类主要用于对对象(Object)的读写。**

**28.** **android 中 service 的实现方法是：startservice 和 bindservice。**

**29.** **activity 一般会重载 7 个方法用来维护其生命周期，除了 onCreate(),onStart(),onDestory() 外还有 onrestart,onresume,onpause,onstop。**

**30.** **android 的数据存储的方式 sharedpreference,文件,SQlite,contentprovider,网络。**

**31.** **当启动一个 Activity 并且新的 Activity 执行完后需要返回到启动它的 Activity 来执行的回调函数是** startActivityForResult

startActivityForResult(Intent,requestCode)//启动一个 activity 包含参数请求码和具体的 intent 数据，其中请求码可以用来识别子活动。

**32.　请使用命令行的方式创建一个名字为 myAvd,sdk 版本为 2.2,sd 卡是在 d 盘的根目录下,名字为 scard.img，并指定屏幕大小 HVGA.**

**android create avd –n myAvd –t 8 –c D:/scard.img –s HVGA**

**33.程序运行的结果是：_____good and gbc_____。**

```
public classExample{
 String str=new String("good");
 char[]ch={'a','b','c'};
 public static void main(String args[]){
        Example ex=new Example();
        ex.change(ex.str,ex.ch);
        System.out.print(ex.str+" and ");
        Sytem.out.print(ex.ch);
 }
 public void change(String str,char ch[]){
        str="test ok";
        ch[0]='g';
 }
}
```



**34.　在 android 中，请简述 jni 的调用过程。(8 分)**

1)安装和下载 Cygwin，下载 Android NDK

2)在 ndk 项目中 JNI 接口的设计

3)使用 C/C++实现本地方法

4)JNI 生成动态链接库.so 文件

5)将动态链接库复制到 java 工程，在 java 工程中调用，运行 java 工

程即可

**35. 简述 Android 应用程序结构是哪些?**（7 分）Android 应用程序结构是：

Linux Kernel(Linux 内核)、Libraries(系统运行库或者是 c/c++核心库)、Application
Framework(开发框架包)、Applications (核心应用程序)

**36. 请继承 SQLiteOpenHelper 实现：(10 分)**

1）.创建一个版本为 1 的“diaryOpenHelper.db”的数据库，

2）.同时创建一个 “diary” 表（包含一个_id 主键并自增长，topic 字符型 100 长度， content 字符型 1000 长度）

3）.在数据库版本变化时请删除 diary 表，并重新创建出 diary 表。

```
publicclass DBHelper  extends SQLiteOpenHelper{

    public final static String DATABASENAME
="diaryOpenHelper.db";
    public final static int DATABASEVERSION =1;

    //创建数据库
    public DBHelper(Context context,Stringname,CursorFactory
factory,int version)
    {
        super(context, name, factory,version);
    }
    //创建表等机构性文件
    public void onCreate(SQLiteDatabase db)
```

```
    {
        String sql ="create tablediary"+
                    "("+
                    "_idinteger primary key autoincrement,"+
                    "topicvarchar(100),"+
                    "contentvarchar(1000)"+
                    ")";
        db.execSQL(sql);
    }
    //若数据库版本有更新，则调用此方法
    public void onUpgrade(SQLiteDatabasedb,int oldVersion,int
newVersion)
    {

        String sql = "drop table ifexists diary";
        db.execSQL(sql);
        this.onCreate(db);
    }
}
```

**37.　页面上现有 ProgressBar 控件 progressBar，请用书写线程以 10 秒的的时间完成其进度显示工作。（10 分）**答案

```
publicclass ProgressBarStu extends Activity {

    private ProgressBar progressBar = null;
    protected void onCreate(BundlesavedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.progressbar);
        //从这到下是关键
        progressBar = (ProgressBar)findViewById(R.id.progressBar);

        Thread thread = new Thread(newRunnable() {
```

```
                @Override
                public void run() {
                    int progressBarMax =progressBar.getMax();
                    try {
                        while(progressBarMax!=progressBar.getProgre
ss())
                        {

                            intstepProgress = progressBarMax/10;
                            intcurrentprogress =
progressBar.getProgress();
                            progressBar.setProgress(currentprogress+
stepProgress);
                            Thread.sleep(1000);
                        }

                    } catch(InterruptedException e) {
                        // TODO Auto-generatedcatch block
                        e.printStackTrace();
                    }

                }
            });

            thread.start();

            //关键结束
        }

}
```

**38.　　请描述下 Activity 的生命周期。**　　必调用的三个方法：

onCreate() --> onStart() --> onResume()，用 AAA 表示

（1）父 Activity 启动子 Activity，子 Actvity 退出，父 Activity 调用顺序

如下

AAA --> onFreeze() --> onPause() --> onStop() --> onRestart()--> onStart(),onResume() …

（2）用户点击 Home，Actvity 调用顺序如下

AAA --> onFreeze() --> onPause() --> onStop() -- Maybe -->onDestroy() – Maybe

（3）调用 finish()， Activity 调用顺序如下

AAA --> onPause() --> onStop() --> onDestroy()

（4）在 Activity 上显示 dialog，Activity 调用顺序如下

AAA

（5）在父 Activity 上显示透明的或非全屏的 activity，Activity 调用顺序如下

AAA --> onFreeze() --> onPause()

（6）设备进入睡眠状态，Activity 调用顺序如下

AAA --> onFreeze() --> onPause()

**39. 如果后台的 Activity 由于某原因被系统回收了，如何在被系统回收之前保存当前状态？** onSaveInstanceState()

当你的程序中某一个 Activity A 在运行时，主动或被动地运行另一个新的 Activity B，这个时候 A 会执行 onSaveInstanceState()。B 完成以后又会来找 A，这个时候就有两种情况：一是 A 被回收，二是 A 没有被回收，被回收的 A 就要重新调用 onCreate()方法，不同于直接启动的是这回 onCreate()里是带上了参数 savedInstanceState；而没被收回的就直接执行 onResume()，跳过 onCreate()了。

**40. 如何将一个 Activity 设置成窗口的样式。**

在 AndroidManifest.xml 中定义 Activity 的地方一句话 android:theme="@android:style/Theme.Dialog"或 android:theme="@android:style/Theme.Translucent"就变成半透明的

**41. 如何退出 Activity？如何安全退出已调用多个 Activity 的 Application？**

对于单一 Activity 的应用来说，退出很简单，直接 finish()即可。

当然，也可以用 killProcess()和 System.exit()这样的方法。

但是，对于多 Activity 的应用来说，在打开多个 Activity 后，如果想在最后打开的 Activity 直接退出，上边的方法都是没有用的，因为上边的方法都是结束一个 Activity 而已。

当然，网上也有人说可以。

就好像有人问，在应用里如何捕获 Home 键，有人就会说用 keyCode 比较 KEYCODE_HOME 即可，而事实上如果不修改 framework，根本不可能做到这一点一样。

所以，最好还是自己亲自试一下。

那么，有没有办法直接退出整个应用呢？

在 2.1 之前，可以使用 ActivityManager 的 restartPackage 方法。

它可以直接结束整个应用。在使用时需要权限 android.permission.RESTART_PACKAGES。

注意不要被它的名字迷惑。

可是，在 2.2，这个方法失效了。

在 2.2 添加了一个新的方法，killBackgroundProcesses()，需要权限 android.permission.KILL_BACKGROUND_PROCESSES。

可惜的是，它和 2.2 的 restartPackage 一样，根本起不到应有的效果。

另外还有一个方法，就是系统自带的应用程序管理里，强制结束程序的方法，forceStopPackage()。

它需要权限 android.permission.FORCE_STOP_PACKAGES。

并且需要添加 android:sharedUserId="android.uid.system"属性

同样可惜的是，该方法是非公开的，他只能运行在系统进程，第三方程序无法调用。

因为需要在 Android.mk 中添加 LOCAL_CERTIFICATE := platform。

而 Android.mk 是用于在 Android 源码下编译程序用的。

从以上可以看出，在 2.2，没有办法直接结束一个应用，而只能用自己的办法间接办到。

现提供几个方法，供参考：

1、抛异常强制退出：

该方法通过抛异常，使程序 ForceClose。

验证可以，但是，需要解决的问题是，如何使程序结束掉，而不弹出 Force Close 的窗口。

2、记录打开的 Activity：

每打开一个 Activity，就记录下来。在需要退出时，关闭每一个 Activity 即可。

3、发送特定广播：

在需要结束应用时，发送一个特定的广播，每个 Activity 收到广播后，关闭即可。

4、递归退出

在打开新的 Activity 时使用 startActivityForResult，然后自己加标志，在 onActivityResult 中处理，递归关闭。

除了第一个，都是想办法把每一个 Activity 都结束掉，间接达到目的。但是这样做同样不完美。

你会发现，如果自己的应用程序对每一个 Activity 都设置了 nosensor，在两个 Activity 结束的间隙，sensor 可能有效了。

但至少，我们的目的达到了，而且没有影响用户使用。

为了编程方便，最好定义一个 Activity 基类，处理这些共通问题。

**42. 请介绍下 Android 中常用的五种布局。**

FrameLayout（框架布局），LinearLayout（线性布局），AbsoluteLayout（绝对布局），RelativeLayout（相对布局），TableLayout（表格布局）

**43. 请介绍下 Android 的数据存储方式。**

一.SharedPreferences 方式

二.文件存储方式

三.SQLite 数据库方式

四.内容提供器（Content provider）方式

五. 网络存储方式

**44. 请介绍下 ContentProvider 是如何实现数据共享的。**

一个程序可以通过实现一个 Content provider 的抽象接口将自己的数据完全暴露出去，而且 Content providers 是以类似数据库中表的方式将数据暴露。Content providers 存储和检索数据，通过它可以让所有的应用程序访问到，这也是应用程序之间唯一共享数据的方法。

要想使应用程序的数据公开化，可通过 2 种方法：创建一个属于你自己的 Content provider 或者将你的数据添加到一个已经存在的 Content provider 中，前提是有相同数据类型并且有写入 Content provider 的权限。
如何通过一套标准及统一的接口获取其他应用程序暴露的数据？
Android 提供了 ContentResolver，外界的程序可以通过 ContentResolver 接口访问 ContentProvider 提供的数据。



**45. 如何启用 Service，如何停用 Service。**Android 中的 service 类似于 windows 中的 service，service 一般没有用户操作界面，它运行于系统中不容易被用户发觉，

可以使用它开发如监控之类的程序。

一。步骤

第一步：继承 Service 类

```
public class SMSService extends Service { }
```

第二步：在 AndroidManifest.xml 文件中的<application>节点里对服务进行配置:

```
<service android:name=".DemoService" />
```

二。Context.startService()和 Context.bindService

服务不能自己运行，需要通过调用 Context.startService()或 Context.bindService()方法启动服务。这两个方法都可

以启动 Service，但是它们的使用场合有所不同。

1.使用 startService()方法启用服务，调用者与服务之间没有关连，即使调用者退出了，服务仍然运行。

使用 bindService()方法启用服务，调用者与服务绑定在了一起，调用者一旦退出，服务也就终止。

2.采用 Context.startService()方法启动服务，在服务未被创建时，系统会先调用服务的 onCreate()方法，

接着调用 onStart()方法。如果调用 startService()方法前服务已经被创建，多次调用 startService()方法并

不会导致多次创建服务，但会导致多次调用 onStart()方法。

采用 startService()方法启动的服务，只能调用 Context.stopService()方法结束服务，服务结束时会调用

onDestroy()方法。

3.采用 Context.bindService()方法启动服务，在服务未被创建时，系统会先调用服务的 onCreate()方法，

接着调用 onBind()方法。这个时候调用者和服务绑定在一起，调用者退出了，系统就会先调用服务的 onUnbind()方法，

。接着调用 onDestroy()方法。如果调用 bindService()方法前服务已经

被绑定，多次调用 bindService()方法并不会

导致多次创建服务及绑定(也就是说 onCreate()和 onBind()方法并不会被多次调用)。如果调用者希望与正在绑定的服务

解除绑定，可以调用 unbindService()方法，调用该方法也会导致系统调用服务的 onUnbind()-->onDestroy()方法。

三。Service 的生命周期

1.Service 常用生命周期回调方法如下：

onCreate() 该方法在服务被创建时调用，该方法只会被调用一次，无论调用多少次 startService()或 bindService()方法，

服务也只被创建一次。 onDestroy()该方法在服务被终止时调用。

2. Context.startService()启动 Service 有关的生命周期方法

onStart() 只有采用 Context.startService()方法启动服务时才会回调该方法。该方法在服务开始运行时被调用。

多次调用 startService()方法尽管不会多次创建服务，但 onStart()方法会被多次调用。

3. Context.bindService()启动 Service 有关的生命周期方法

onBind()只有采用 Context.bindService()方法启动服务时才会回调该方法。该方法在调用者与服务绑定时被调用，

当调用者与服务已经绑定，多次调用 Context.bindService()方法并不会导致该方法被多次调用。

onUnbind()只有采用 Context.bindService()方法启动服务时才会回调该方法。该方法在调用者与服务解除绑定时被调用。

备注：

1. 采用 startService()启动服务

```
Intent intent =new Intent(DemoActivity.this, DemoService.class);
startService(intent);
```

2.Context.bindService()启动

```
Intent intent =new Intent(DemoActivity.this, DemoService.class);
bindService(intent, conn, Context.BIND_AUTO_CREATE);
//unbindService(conn);//解除绑定
```

**46. 注册广播有几种方式，这些方式有何优缺点？请谈谈 Android 引入广播机制的用意。** Android 广播机制（两种注册方法）

在 android 下，要想接受广播信息，那么这个广播接收器就得我们自己来实现了，我们可以继承 BroadcastReceiver，就可以有一个广播接受器了。有个接受器还不够，我们还得重写 BroadcastReceiver 里面的 onReceiver 方法，当来广播的时候我们要干什么，这就要我们自己来实现，不过我们可以搞一个信息防火墙。具体的代码：

```
public class SmsBroadCastReceiverextends BroadcastReceiver
{

  @Override
  public void onReceive(Context context, Intent intent)
  {
     Bundle bundle = intent.getExtras();
     Object[] object = (Object[])bundle.get("pdus");
     SmsMessage sms[]=new SmsMessage[object.length];
```

```
        for(int i=0;i<object.length;i++)
        {
            sms[0] =SmsMessage.createFromPdu((byte[])object);
            Toast.makeText(context, "来自
"+sms.getDisplayOriginatingAddress()+"的消息是：
"+sms.getDisplayMessageBody(),Toast.LENGTH_SHORT).show();
        }
        //终止广播，在这里我们可以稍微处理，根据用户输入的号码可以
实现短信防火墙。
        abortBroadcast();
    }

}
```

当实现了广播接收器，还要设置广播接收器接收广播信息的类型，这里是信息：android.provider.Telephony.SMS_RECEIVED

我们就可以把广播接收器注册到系统里面，可以让系统知道我们有个广播接收器。这里有两种，一种是代码动态注册：

```
//生成广播处理
smsBroadCastReceiver = newSmsBroadCastReceiver();
//实例化过滤器并设置要过滤的广播

IntentFilter intentFilter =
newIntentFilter("android.provider.Telephony.SMS_RECEIVED");

//注册广播
BroadCastReceiverActivity.this.registerReceiver(smsBroadCastRecei
```

ver,intentFilter);

一种是在 AndroidManifest.xml 中配置广播

```
<?xml version="1.0"encoding="utf-8"?>
<manifestxmlns:android="http://schemas.android.com/apk/res/androi
d"
    package="spl.broadCastReceiver"
    android:versionCode="1"
    android:versionName="1.0">
  <application
android:icon="@drawable/icon"android:label="@string/app_name">
      <activity android:name=".BroadCastReceiverActivity"
                android:label="@string/app_name">
          <intent-filter>
              <actionandroid:name="android.intent.action.MAIN" />
              <categoryandroid:name="android.intent.category.LAUNCH
ER" />
          </intent-filter>
      </activity>

      <!--广播注册-->
      <receiver android:name=".SmsBroadCastReceiver">
          <intent-filterandroid:priority="20">
              <actionandroid:name="android.provider.Telephony.SMS_R
ECEIVED"/>
          </intent-filter>
      </receiver>

  </application>

  <uses-sdk android:minSdkVersion="7" />

  <!-- 权限申请 -->
```

<uses-permissionandroid:name="android.permission.RECEIVE_SMS"></uses-permission>

</manifest>

两种注册类型的区别是：

1)第一种不是常驻型广播，也就是说广播跟随程序的生命周期。

2)第二种是常驻型，也就是说当应用程序关闭后，如果有信息广播来，程序也会被系统调用自动运行。

**47. 请解释下在单线程模型中 Message、Handler、MessageQueue、Looper 之间的关系。**

Handler 简介：

一个 Handler 允许你发送和处理 Message 和 Runable 对象，这些对象和一个线程的 MessageQueue 相关联。每一个线程实例和一个单独的线程以及该线程的 MessageQueue 相关联。当你创建一个新的 Handler 时，它就和创建它的线程绑定在一起了。这里，线程我们也可以理解为线程的 MessageQueue。从这一点上来看，Handler 把 Message 和 Runable 对象传递给 MessageQueue，而且在这些对象离开 MessageQueue 时，Handler 负责执行他们。

Handler 有两个主要的用途：（1）确定在将来的某个时间点执行一个或者一些 Message 和 Runnable 对象。（2）在其他线程（不是 Handler 绑定线程）中排入一些要执行的动作。

Scheduling Message，即（1），可以通过以下方法完成：

post(Runnable):Runnable 在 handler 绑定的线程上执行，也就是说不创建新线程。

postAtTime(Runnable,long):
postDelayed(Runnable,long):
sendEmptyMessage(int):
sendMessage(Message):
sendMessageAtTime(Message,long):
sendMessageDelayed(Message,long):

post 这个动作让你把 Runnable 对象排入 MessageQueue,MessageQueue 受到这些消息的时候执行他们，当然以一定的排序。sendMessage 这个动作允许你把 Message 对象排成队列，这些 Message 对象包含一些信息，Handler 的 hanlerMessage(Message)会处理这些 Message.当然，handlerMessage(Message)必须由 Handler 的子类来重写。这是编程人员需要作的事。

当 posting 或者 sending 到一个 Hanler 时，你可以有三种行为：当 MessageQueue 准备好就处理，定义一个延迟时间，定义一个精确的时间去处理。后两者允许你实现 timeout,tick,和基于时间的行为。

当你的应用创建一个新的进程时，主线程（也就是 UI 线程）自带一个 MessageQueue，这个 MessageQueue 管理顶层的应用对象（像 activities,broadcast receivers 等）和主线程创建的窗体。你可以创建自

己的线程，并通过一个 Handler 和主线程进行通信。这和之前一样，通过 post 和 sendmessage 来完成，差别在于在哪一个线程中执行这么方法。在恰当的时候，给定的 Runnable 和 Message 将在 Handler 的 MessageQueue 中被 Scheduled。

Message 简介：

Message 类就是定义了一个信息，这个信息中包含一个描述符和任意的数据对象，这个信息被用来传递给 Handler.Message 对象提供额外的两个 int 域和一个 Object 域，这可以让你在大多数情况下不用作分配的动作。

尽管 Message 的构造函数是 public 的，但是获取 Message 实例的最好方法是调用 Message.obtain(),或者 Handler.obtainMessage()方法，这些方法会从回收对象池中获取一个。



MessageQueue 简介：

这是一个包含 message 列表的底层类。Looper 负责分发这些 message。Messages 并不是直接加到一个 MessageQueue 中，而是通过 MessageQueue.IdleHandler 关联到 Looper。

你可以通过 Looper.myQueue()从当前线程中获取 MessageQueue。

Looper 简介：

Looper 类被用来执行一个线程中的 message 循环。默认情况，没有一个消息循环关联到线程。在线程中调用 prepare()创建一个 Looper，然后用 loop()来处理 messages，直到循环终止。

大多数和 message loop 的交互是通过 Handler。

下面是一个典型的带有 Looper 的线程实现。

```
class LooperThread extends Thread {
    public Handler mHandler;

    public void run() {
        Looper.prepare();

        mHandler = new Handler() {
            public voidhandleMessage(Message msg) {
                // process incomingmessages here
            }
        };

        Looper.loop();
    }
}
```

**48.　AIDL 的全称是什么？如何工作？能处理哪些类型的数据？**

AIDL 的英文全称是 Android Interface Define Language

当 A 进程要去调用 B 进程中的 service 时，并实现通信，我们通常都是通过 AIDL 来操作的

A 工程：

首先我们在 net.blogjava.mobile.aidlservice 包中创建一个

RemoteService.aidl 文件，在里面我们自定义一个接口，含有方法 get。ADT 插件会在 gen 目录下自动生成一个 RemoteService.java 文件，该类中含有一个名为 RemoteService.stub 的内部类，该内部类中含有 aidl 文件接口的 get 方法。

说明一：aidl 文件的位置不固定，可以任意

然后定义自己的 MyService 类，在 MyService 类中自定义一个内部类去继承 RemoteService.stub 这个内部类，实现 get 方法。在 onBind 方法中返回这个内部类的对象，系统会自动将这个对象封装成 IBinder 对象，传递给他的调用者。

其次需要在 AndroidManifest.xml 文件中配置 MyService 类，代码如下：

```
<!-- 注册服务 -->
<service android:name=".MyService">
  <intent-filter>
   <!--  指定调用 AIDL 服务的 ID  -->
      <actionandroid:name="net.blogjava.mobile.aidlservice.RemoteService" />
  </intent-filter>
</service>
```

为什么要指定调用 AIDL 服务的 ID,就是要告诉外界 MyService 这个类能够被别的进程访问，只要别的进程知道这个 ID，正是有了这个 ID,B 工程才能找到 A 工程实现通信。

说明：AIDL 并不需要权限

B 工程：

首先我们要将 A 工程中生成的 RemoteService.java 文件拷贝到 B

工程中，在 bindService 方法中绑定 aidl 服务

绑定 AIDL 服务就是将 RemoteService 的 ID 作为 intent 的 action 参数。

说明：如果我们单独将 RemoteService.aidl 文件放在一个包里，那个在我们将 gen 目录下的该包拷贝到 B 工程中。如果我们将 RemoteService.aidl 文件和我们的其他类存放在一起，那么我们在 B 工程中就要建立相应的包，以保证 RmoteService.java 文件的报名正确，我们不能修改 RemoteService.java 文件

bindService(newInten("net.blogjava.mobile.aidlservice.RemoteService"),serviceConnection, Context.BIND_AUTO_CREATE);

ServiceConnection 的 onServiceConnected(ComponentName name, IBinderservice)方法中的 service 参数就是 A 工程中 MyService 类中继承了 RemoteService.stub 类的内部类的对象。

**49. 请解释下 Android 程序运行时权限与文件系统权限的区别。**

运行时权限 Dalvik( android 授权)

文件系统 linux 内核授权

**50. 系统上安装了多种浏览器，能否指定某浏览器访问指定页面？请说明原由。**

通过直接发送 Uri 把参数带过去，或者通过 manifest 里的 intentfilter 里的 data 属性

**51. 你如何评价 Android 系统？优缺点。**答：Android 平台手机 5 大

优势：

一、开放性

在优势方面，Android 平台首先就是其开发性，开发的平台允许任何移动终端厂商加入到 Android 联盟中来。显著的开放性可以使其拥有更多的开发者，随着用户和应用的日益丰富，一个崭新的平台也将很快走向成熟。开放性对于 Android 的发展而言，有利于积累人气，这里的人气包括消费者和厂商，而对于消费者来讲，随大的受益正是丰富的软件资源。开放的平台也会带来更大竞争，如此一来，消费者将可以用更低的价位购得心仪的手机。

二、挣脱运营商的束缚

在过去很长的一段时间，特别是在欧美地区，手机应用往往受到运营商制约，使用什么功能接入什么网络，几乎都受到运营商的控制。从去年 iPhone 上市，用户可以更加方便地连接网络，运营商的制约减少。随着 EDGE、HSDPA 这些 2G 至 3G 移动网络的逐步过渡和提升，手机随意接入网络已不是运营商口中的笑谈，当你可以通过手机 IM 软件方便地进行即时聊天时，再回想不久前天价的彩信和图铃下载业务，是不是像噩梦一样？互联网巨头 Google 推动的 Android 终端天生就有网络特色，将让用户离互联网更近。

三、丰富的硬件选择

这一点还是与 Android 平台的开放性相关，由于 Android 的开放性，众多的厂商会推出千奇百怪，功能特色各具的多种产品。功能上的差异和特色，却不会影响到数据同步、甚至软件的兼容，好比你从诺基亚

Symbian 风格手机 一下改用苹果 iPhone ，同时还可将 Symbian 中优秀的软件带到 iPhone 上使用、联系人等资料更是可以方便地转移，是不是非常方便呢？

四、不受任何限制的开发商

Android 平台提供给第三方开发商一个十分宽泛、自由的环境，不会受到各种条条框框的阻扰，可想而知，会有多少新颖别致的软件会诞生。但也有其两面性，血腥、暴力、情色方面的程序和游戏如可控制正是留给 Android 难题之一。

五、无缝结合的 Google 应用

如今叱诧互联网的 Google 已经走过 10 年度历史，从搜索巨人到全面的互联网渗透，Google 服务如地图、邮件、搜索等已经成为连接用户和互联网的重要纽带，而 Android 平台手机将无缝结合这些优秀的 Google 服务。

再说 Android 的 5 大不足：

一、安全和隐私

由于手机与互联网的紧密联系，个人隐私很难得到保守。除了上网过程中经意或不经意留下的个人足迹，Google 这个巨人也时时站在你的身后，洞穿一切，因此，互联网的深入将会带来新一轮的隐私危机。

二、首先开卖 Android 手机的不是最大运营商

众所周知，T-Mobile 在 23 日，于美国纽约发布了 Android 首款手机 G1。但是在北美市场，最大的两家运营商乃 AT&T 和 Verizon，而目前所知取得 Android 手机销售权的仅有 T-Mobile 和 Sprint，其中 T-Mobile 的

3G 网络相对于其他三家也要逊色不少，因此，用户可以买账购买 G1，能否体验到最佳的 3G 网络服务则要另当别论了！

三、运营商仍然能够影响到 Android 手机

在国内市场，不少用户对购得移动定制机不满，感觉所购的手机被人涂画了广告一般。这样的情况在国外市场同样出现。Android 手机的另一发售运营商 Sprint 就将在其机型中内置其手机商店程序。

四、同类机型用户减少

在不少手机论坛都会有针对某一型号的子论坛，对一款手机的使用心得交流，并分享软件资源。而对于 Android 平台手机，由于厂商丰富，产品类型多样，这样使用同一款机型的用户越来越少，缺少统一机型的程序强化。举个稍显不当的例子，现在山寨机泛滥，品种各异，就很少有专门针对某个型号山寨机的讨论和群组，除了哪些功能异常抢眼、颇受追捧的机型以外。

五、过分依赖开发商缺少标准配置

在使用 PC 端的 Windows Xp 系统的时候，都会内置微软 Windows Media Player 这样一个浏览器程序，用户可以选择更多样的播放器，如 Realplay 或暴风影音等。但入手开始使用默认的程序同样可以应付多样的需要。在 Android 平台中，由于其开放性，软件更多依赖第三方厂商，比如 Android 系统的 SDK 中就没有内置音乐播放器，全部依赖第三方开发，缺少了产品的统一性。

**52. 什么是 ANR 如何避免它?**

答：ANR：Application NotResponding，五秒

在 Android 中，活动管理器和窗口管理器这两个系统服务负责监视应用程序的响应。当出现下列情况时，Android 就会显示 ANR 对话框了：

对输入事件(如按键、触摸屏事件)的响应超过 5 秒

意向接受器(intentReceiver)超过 10 秒钟仍未执行完毕

Android 应用程序完全运行在一个独立的线程中(例如 main)。这就意味着，任何在主线程中运行的，需要消耗大量时间的操作都会引发 ANR。因为此时，你的应用程序已经没有机会去响应输入事件和意向广播(Intentbroadcast)。

因此，任何运行在主线程中的方法，都要尽可能的只做少量的工作。特别是活动生命周期中的重要方法如 onCreate()和 onResume()等更应如此。潜在的比较耗时的操作，如访问网络和数据库;或者是开销很大的计算，比如改变位图的大小，需要在一个单独的子线程中完成(或者是使用异步请求，如数据库操作)。但这并不意味着你的主线程需要进入阻塞状态已等待子线程结束 -- 也不需要调用 Therad.wait()或者 Thread.sleep()方法。取而代之的是，主线程为子线程提供一个句柄(Handler)，让子线程在即将结束的时候调用它(xing:可以参看 Snake 的例子，这种方法与以前我们所接触的有所不同)。使用这种方法涉及你的应用程序，能够保证你的程序对输入保持良好的响应，从而避免因为

输入事件超过 5 秒钟不被处理而产生的 ANR。这种实践需要应用到所有显示用户界面的线程，因为他们都面临着同样的超时问题。

**53. 什么情况会导致 Force Close ?如何避免?能否捕获导致其的异常?**

答：一般像空指针啊，可以看起 logcat，然后对应到程序中 来解决错误

**54. Android 本身的 api 并未声明会抛出异常，则其在运行时有无可能抛出 runtime 异常，你遇到过吗?诺有的话会导致什么问题?如何解决?**

会。比如 nullpointerException。我遇到过。比如空指针异常是最常见的异常，只要对 null 调用方法就会出现 nullpointerException。会导致程序无法正常运行出现 forceclose。可以打开控制台查看 logcat 信息，然后找到抛出异常信息的代码段并进行修改。



**55. 简要解释一下 activity、 intent 、intent filter、service、Broadcase、BroadcaseReceiver**

答：一个 activity 呈现了一个用户可以操作的可视化用户界面

一个 service 不包含可见的用户界面，而是在后台无限地运行

可以连接到一个正在运行的服务中，连接后，可以通过服务中暴露出来的借口与其进行通信

一个 broadcast receiver 是一个接收广播消息并作出回应的 component，broadcastreceiver 没有界面

intent:content provider 在接收到 ContentResolver 的请求时被激活。

activity, service 和 broadcast receiver 是被称为 intents 的异步消息激活的。

一个 intent 是一个 Intent 对象，它保存了消息的内容。对于 activity 和 service 来说，它指定了请求的操作名称和待操作数据的 URI

Intent 对象可以显式的指定一个目标 component。如果这样的话，android 会找到这个 component(基于 manifest 文件中的声明)并激活它。但如果一个目标不是显式指定的，android 必须找到响应 intent 的最佳 component。

它是通过将 Intent 对象和目标的 intent filter 相比较来完成这一工作的。一个 component 的 intent filter 告诉 android 该 component 能处理的 intent。intent filter 也是在 manifest 文件中声明的。

**56. IntentService 有何优点?**

答：IntentService 的好处

* Acitivity 的进程，当处理 Intent 的时候，会产生一个对应的 Service

* Android 的进程处理器现在会尽可能的不 kill 掉你

* 非常容易使用

**57. 横竖屏切换时候 activity 的生命周期?**

1、不设置 Activity 的 android:configChanges 时，切屏会重新调用各个生命周期，切横屏时会执行一次，切竖屏时会执行两次

2、设置 Activity 的 android:configChanges="orientation"时，切屏还是会重新调用各个生命周期，切横、竖屏时只会执行一次

3、设置 Activity 的 android:configChanges="orientation|keyboardHidden"时，切屏不会重新调用各个生命周期，只会执行 onConfigurationChanged 方法

如何将 SQLite 数据库(dictionary.db 文件)与 apk 文件一起发布?

解答：可以将 dictionary.db 文件复制到 Eclipse Android 工程中的 res/raw 目录中。所有在 res/raw 目录中的文件不会被压缩，这样可以直接提取该目录中的文件。可以将 dictionary.db 文件复制到 res/raw 目录中

**58. 如何打开 res/raw 目录中的数据库文件?**

解答：在 Android 中不能直接打开 res/raw 目录中的数据库文件，而需要在程序第一次启动时将该文件复制到手机内存或 SD 卡的某个目录中，然后再打开该数据库文件。复制的基本方法是使用 getResources().openRawResource 方法获得 res aw 目录中资源的 InputStream 对象，然后将该 InputStream 对象中的数据写入其他的目录中相应文件中。在 Android SDK 中可以使用 SQLiteDatabase.openOrCreateDatabase 方法来打开任意目录中的 SQLite 数据库文件。

**59. Android 引入广播机制的用意?**

答：a:从 MVC 的角度考虑(应用程序内)

其实回答这个问题的时候还可以这样问，android 为什么要有那 4 大组件，现在的移动开发模型基本上也是照搬的 web 那一套 MVC 架构，只不过是改了点嫁妆而已。android 的四大组件本质上就是为了实现移动或者说嵌入式设备上的 MVC 架构，它们之间有时候是一种相互依存的关系，有时候又是一种补充关系，引入广播机制可以方便几大组件的信息和数据交互。

b：程序间互通消息(例如在自己的应用程序内监听系统来电)

c：效率上(参考 UDP 的广播协议在局域网的方便性)

d：设计模式上(反转控制的一种应用，类似监听者模式)

**60. Android dvm 的进程和 Linux 的进程，应用程序的进程是否为同一个概念**

DVM 指 dalivk 的虚拟机。每一个 Android 应用程序都在它自己的进程中运行，都拥有一个独立的 Dalvik 虚拟机实例。而每一个 DVM 都是在 Linux 中的一个进程，所以说可以认为是同一个概念。

**61. sim 卡的 EF 文件有何作用**

sim 卡的文件系统有自己规范，主要是为了和手机通讯，sim 本身可以有自己的操作系统，EF 就是作存储并和手机通讯用的

**62. 嵌入式操作系统内存管理有哪几种，各有何特性**

页式，段式，段页，用到了 MMU,虚拟空间等技术

**63. 什么是嵌入式实时操作系统，Android 操作系统属于实时操作系统吗？**

嵌入式实时操作系统是指当外界事件或数据产生时，能够接受并以足够快的速度予以处理，其处理的结果又能在规定的时间之内来控制生产过程或对处理系统作出快速响应，并控制所有实时任务协调一致运行的嵌入式操作系统。主要用于工业控制、军事设备、航空航天等领域对系统的响应时间有苛刻的要求，这就需要使用实时系统。又可分为软实时和硬实时两种，而 android 是基于 linux 内核的，因此属于软实时。

**64. 一条最长的短信息约占多少 byte?**

中文 70(包括标点)，英文 160，160 个字节。

**65. android 中的动画有哪几类，它们的特点和区别是什么?**

两种，一种是 Tween 动画、还有一种是 Frame 动画。Tween 动画，这种实现方式可以使视图组件移动、放大、缩小以及产生透明度的变化;另一种 Frame 动画，传统的动画方法，通过顺序的播放排列好的图片来实现，类似电影。

**66. handler 机制的原理**

andriod 提供了 Handler 和 Looper 来满足线程间的通信。Handler 先进先出原则。Looper 类用来管理特定线程内对象之间的消息交换(MessageExchange)。

1)Looper: 一个线程可以产生一个 Looper 对象，由它来管理此线程里的 MessageQueue(消息队列)。

2)Handler: 你可以构造 Handler 对象来与 Looper 沟通，以便 push 新消息到 MessageQueue 里;或者接收 Looper 从 Message Queue 取出)所送来的消息。

3) Message Queue(消息队列):用来存放线程放入的消息。

4)线程：UIthread 通常就是 main thread，而 Android 启动程序时

会替它建立一个 MessageQueue。

**67. 说说 mvc 模式的原理，它在 android 中的运用**

MVC(Model_view_contraller)”模型_视图_控制器”。 MVC 应用程序总是由这三个部分组成。Event(事件)导致 Controller 改变 Model 或 View，或者同时改变两者。只要 Controller 改变了 Models 的数据或者属性，所有依赖的 View 都会自动更新。类似的，只要 Contro

**68. DDMS 和 TraceView 的区别?**

DDMS 是一个程序执行查看器，在里面可以看见线程和堆栈等信息，TraceView 是程序性能分析器。

**69. java 中如何引用本地语言**

可以用 JNI（java nativeinterface java 本地接口）接口。

**70. 谈谈 Android 的 IPC（进程间通信）机制**

IPC 是内部进程通信的简称，是共享"命名管道"的资源。Android 中的 IPC 机制是为了让 Activity 和 Service 之间可以随时的进行交互，故在 Android 中该机制，只适用于 Activity 和 Service 之间的通信，类似于远程方法调用，类似于 C/S 模式的访问。通过定义 AIDL 接口文件来定义 IPC 接口。Servier 端实现 IPC 接口，Client 端调用 IPC 接口本地代理。

**71. NDK 是什么?**

NDK 是一些列工具的集合，NDK 提供了一系列的工具，帮助开发者迅速的开发 C/C++的动态库，并能自动将 so 和 java 应用打成 apk 包。

NDK 集成了交叉编译器，并提供了相应的 mk 文件和隔离 cpu、平台等的差异，开发人员只需简单的修改 mk 文件就可以创建出 so