# 面试知识点总结

1. **问题：**

1. XXX（某个比较重要的点）是怎么实现的？

2. 你在项目中遇到的最大的困难是什么，怎么解决的？

3. 项目某个部分考虑的不够全面，如果XXXX，你怎么优化？

4. XXX（一个新功能）需要实现，你有什么思路？

1. **自己应该问的问题：**

1. 贵公司一向以XXX著称，能不能说明一下公司这方面的特点？

2. 贵公司XXX业务发展很好，这是公司发展的重点么？

3. 对技术和业务怎么看？

4. 贵公司一般的团队是多大，几个人负责一个产品或者业务？

5. 贵公司的开发中是否会使用到一些最新技术？

6. 对新人有没有什么培训，会不会安排导师？

7. 对Full Stack怎么看？

8. 你觉得我有哪些需要提高的地方？

## 四．Android部分

**1. activity的启动模式：**

**Standard**

**singleTop：置顶**

**SingleTask:清空置顶**

**singleInstance：废资源的activity：浏览器**

**2. 用于实现后台线程与UI线程的交互。**

* 1. **Handler**
  2. **Activity.runOnUIThread(Runnable)**
  3. **View.Post(Runnable)**
  4. **View.PostDelayed(Runnabe,long)**
  5. **AsyncTask**

**3. Broadcast Receiver生命周期**

**生命周期只有十秒左右，如果在onReceive()内做超过十秒内的事情，就会报ANR(Application NoResponse)程序无响应的错误信息。它的生命周期为从回调onReceive()方法开始到该方法返回结果后结束。**

# **6.ANR**

**1. ANR错误一般有三种类型**

1. **KeyDispatchTimeout(5 seconds) --主要是类型按键或触摸事件在特定时间内无响应**
2. **BroadcastTimeout(10 seconds) --BroadcastReceiver在特定时间内无法处理完成**
3. **ServiceTimeout(20 secends) --小概率事件 Service在特定的时间内无法处理完成**
4. **哪些操作会导致ANR 在主线程执行以下操作：** 
   1. **高耗时的操作，如图像变换**
   2. **磁盘读写，数据库读写操作**
   3. **大量的创建新对象**
5. **如何避免**
   1. **UI线程尽量只做跟UI相关的工作**
   2. **耗时的操作(比如数据库操作，I/O，连接网络或者别的有可能阻塞UI线程的操作)把它放在单独的线程处理**
   3. **尽量用Handler来处理UIThread和别的Thread之间的交互**
6. **解决的逻辑：**
   1. **使用AsyncTask： 在doInBackground()方法中执行耗时操作，在onPostExecuted()更新UI**
   2. **使用Handler实现异步任务**
      1. **在子线程中处理耗时操作 ；**
      2. **处理完成之后，通过handler.sendMessage()传递处理结果**
      3. **在handler的handleMessage()方法中更新UI**
      4. **或者使用handler.post()方法将消息放到Looper中**

# **7.FC(Force Close)**

## **什么时候会出现**

1. **Error**
2. **OOM，内存溢出**
3. **StackOverFlowError**
4. **Runtime,比如说空指针异常**

## **解决的办法**

1. **注意内存的使用和管理**
2. **使用Thread.UncaughtExceptionHandler接口\**

### 9. [Service与IntentService的使用比较](http://1028826685.iteye.com/blog/1485800)

## Service运行在主线程，操作耗时操作会阻塞，intentService运行在 一个单独的worked Thread里处理，不会阻塞主线程。当多次启动时IntentService会处理完成一个之后再处理第二个

## 10. android的触摸机制：

## 1）View的消息分发机制：

a**）View.dispatchEvent->View.setOnTouchListener->View.onTouchEvent**

**注解：在dispatchTouchEvent中会进行OnTouchListener的判断，如果 OnTouchListener不为null且返回true，则表示事件被消费，onTouchEvent不 会被执行；否则执行onTouchEvent。**

**b) 设置了onLongClickListener，且onLongClickListener.onClick返回true， 则点击事件OnClick事件无法触发；没有设置onLongClickListener或者 onLongClickListener.onClick返回 false，则点击事件OnClick事件依然可以 触发；**

**2）ViewGroup消息分发机制：**

**MyLinearLayout的dispatchTouchEvent -> MyLinearLayout的onInterceptTouchEvent -> MyButton的dispatchTouchEvent ->Mybutton的onTouchEvent**

## 总结：

**以上的几种情况就是我们经常遇到的了，总结起来有几个重要的点 :  
 1、如果Parent ViewGroup的onInterceptTouchEvent返回false, 并且触摸的目标view对于触摸事件的处理结果返回的是true,那么后续事件会先经过parent 的onInterceptTouchEvent， 然后再交给目标view进行处理；  
 2、如果Parent ViewGroup的onInterceptTouchEvent返回true,即对事件进行拦截，那么事件将不会再经过onInterceptTouchEvent，而是直接进入到onTouchEvent进行处理；如果onTouchEvent返回true，则表示该事件被处理了；如果返回FALSE，则代表事件没有被处理，那么事件会被上交给它的parent来处理，如果没有parent来处理，那么最终会交给Activity来处理；  
 3、如果用户在触摸的某个事件才拦截，那么目标view会收到一个CANCEL事件，然后后续的事件不会再交给目标view，而被转交给Parent的onTouchEvent方法进行处理。比如情景6当中，在TouchLayout的DOWN时不对事件进行拦截，这时事件会被TouchTv正常处理。但是在MOVE时事件被拦截了，此时TouchTv收到了一个CANCEL事件，MOVE以及后续的事件就交给了TouchLayout进行处理。这个情景就是我们做下拉刷新时要用的场景了。**

**例：**需要写一个类似slidingmenu的左侧隐藏menu，主Activity上有个Button、ListView或者任何可以响应点击的View，你在当前View上死命的滑动，菜单栏也出不来；因为MOVE事件被子View处理了~ 你需要这么做：在ViewGroup的dispatchTouchEvent中判断用户是不是想显示菜单，如果是，则在onInterceptTouchEvent(ev)拦截子View的事件；自己进行处理，这样自己的onTouchEvent就可以顺利展现出菜单栏了~~

## C++ 的 delete 和 C 的 free 有啥区别？

## http和HTTPS的区别。

## hashmap的实现原理

## 设计模式

## Android内存泄漏问题，内存泄漏分析工具

## Android的消息机制，Handler原理

## view的绘制流程

## Java GC机制

## 多线程同步

## Android里的序列化

## Android里的设计模式

## string stringbuild stringbuffer区别，栈，堆

## 写出单例模式

## final finalized finally区别

## 内存泄露原因，如何解决

## static 方法，变量

## wait notify

## sync 和 lock 的区别

## 实际开发中软引用或者弱引用的使用场景：

****利用软引用和弱引用解决OOM问题****：用一个HashMap来保存图片的路径和相应图片对象关联的软引用之间的映射关系，在内存不足时，JVM会自动回收这些缓存图片对象所占用的空间，从而有效地避免了OOM的问题 通过软可及对象重获方法实现Java对象的高速缓存:比如我们创建了一Employee的类，如果每次需要查询一个雇员的信息。哪怕是几秒中之前刚刚查询过的，都要重新构建一个实例，这是需要消耗很多时间的。我们可以通过软引用和 HashMap 的结合，先是保存引用方面：以软引用的方式对一个Employee对象的实例进行引用并保存该引用到HashMap 上，key 为此雇员的 id，value为这个对象的软引用，另一方面是取出引用，缓存中是否有该Employee实例的软引用，如果有，从软引用中取得。如果没有软引用，或者从软引用中得到的实例是null，重新构建一个实例，并保存对这个新建实例的软引用

35. A-->B onpauseA->oncreateB->onstartB->onresumeB->onstopA

按back键or finish：onpauseB->onrestart-A>onstartA->onresumeA->onstopB->ondestroyB

1. **Content Provider的理解以及用法**
2. HashMap的底层实现
3. 如何避免overdraw
4. Fragment的生命周期，栈管理，和commit()类似的方法还有哪个，commit类似的那个之前真没注意，应该是commitAllowingStateLoss()，区别是是否能在activity保存之后调用
5. 三次握手，四次断开
6. 进程和线程区别
7. 排序算法