**Android 中的Hander详解**

1. **Handler是什么**
2. handler是Android给我们提供来更新UI的一套机制，也是一套消息处理的机制，我们可以发送消息，也可以通过塔来处理消息，handler在我们的framework中是非常常见的
3. android 在设计的时候，就封装了一套消息创建 传递 处理机制，如果不遵循这样的机制就没有办法更新UI信息，就会抛出异常信息
4. **Handler用法**
5. 传递Message。用于接受子线程发送的数据, 并用此数据配合主线程更新UI。有以下方法：

sendEmptyMessage(int)

sendMessage(Message)

sendMessageAtTime(Message,long)

sendMessageDelayed(Message,long)

sendMessage类方法, 允许你安排一个带数据的Message对象到队列中，等待更新.

（1）、使用Handler在子线程中向ui线程发送一个消息进行UI的更新；

（2）、创建一个Message; Message msg = new Message(); msg.arg1=88;

（3）、handler.sendMessage(msg);

msg.obj=xxx; 可以传递一个对象；

（4）、当然不一定要用new一个Message，也可以复用系统的message对象 Message msg = handler.obtainMessage();

1. 传递Runnable对象。用于通过Handler绑定的消息队列，安排不同操作的执行顺序。主要有以下方法：

post(Runnable)

postAtTime(Runnable,long)

postDelayed(Runnable long)

post类方法允许你排列一个Runnable对象到主线程队列中

3．传递Callback对象。CallBack用于截获handler发送的消息 如果返回true 就截获成功 不会向下传递了。



上面的实例中，第一个有返回值的handleMessage方法是Callback的回调，如果返回true,则不执行下面的handleMessage方法，从而达到拦截handler发送的消息的目的，如果返回false，则会继续执行handlerMessage方法。

1. **Handler原理**

3.1 Android为什么要设计只能通过Handler机制更新UI呢？

最根本的目的就是解决多线程并发问题

假设在一个Activity当中，有多个线程去更新UI，并且没有枷锁机制，那么会产生什么样子问题呢，那就是更新界面混乱。如果对更新UI的操作进行枷锁处理的话又会产生什么样的问题呢？那就是性能下降。Handler通过消息队列，保证了消息处理的先后有序。

鉴于以上问题的考虑，Android给我们提供了一套更新UI的机制，我们只要使用一套机制就好，所有的更新UI的操作都是在主线程中轮询处理。

3.2 Handler的原理是什么？

（1）Handler封装了消息的发送（主要包括消息发送给谁）

Looper：

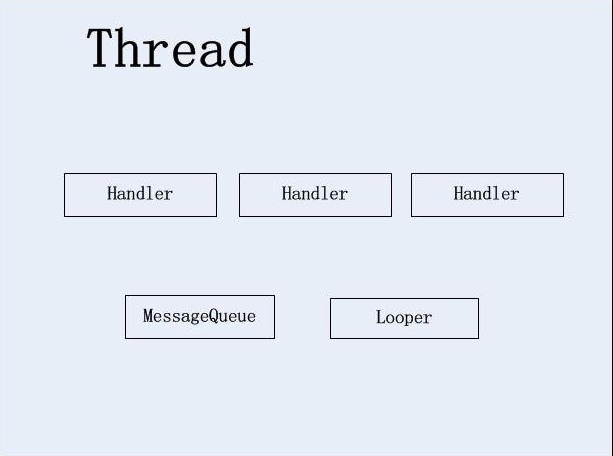
1. 内部包含一个消息队列也就是MessageQueue，所有Handler发送的消息都走向这个队列。
2. Looper.loop()方法，就是一个for死循环，不断的从MessageQueue取消息，如果有消息就处理消息，没有消息就阻塞。

（2）MessageQueue，就是一个消息队列，可以添加消息，处理消息。

（3）Handler 也不难，比较简单，在构造Handler时候内部会跟Looper进行关联，通过Looper.myLooper()获取到Looper，找到了Looper，也就找到了 MessageQueue。在Handler中发生消息，其实是向MessageQueue队列中发生消息。

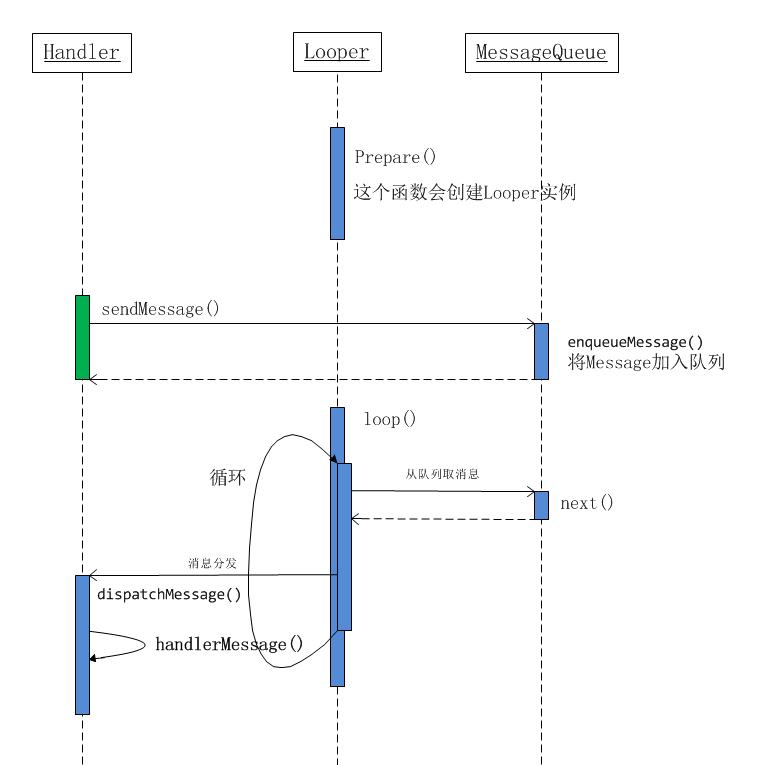
3.3 Handler与Looper、MessageQueue的关系

**总结：handler负责发送消息，Looper负责接收Handler发送的消息，并直接把消息回传给handler自己，MessageQueue就是一个存储消息的容器**

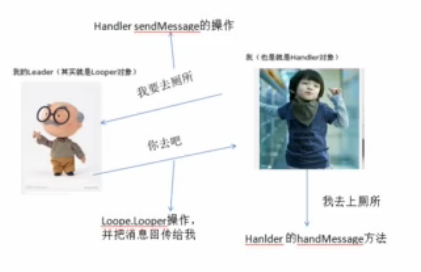


如上图所示：一个线程中只有一个Looper实例，一个MessageQueue实例，可以有多个Handler实例。

下图示意了Handler、MessageQueue、Looper之间是如何协作的。



下图是一个“我要上厕所”的形象图解：



1. **Handler与子线程**
2. 自定义与线程相关的Handler：

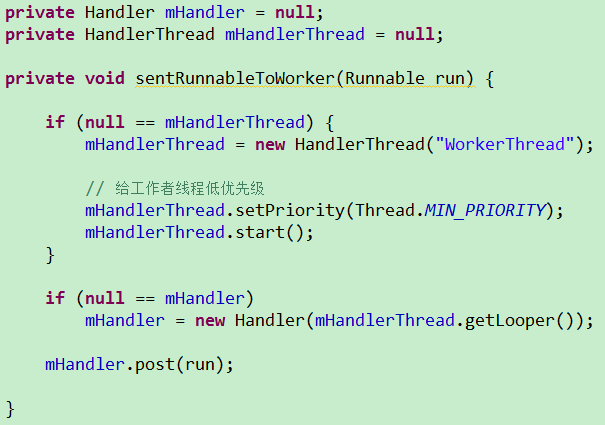


一般UI主线程中不要执行一些耗时的操作，这样就可以通过子线程消息来处理耗时操作。

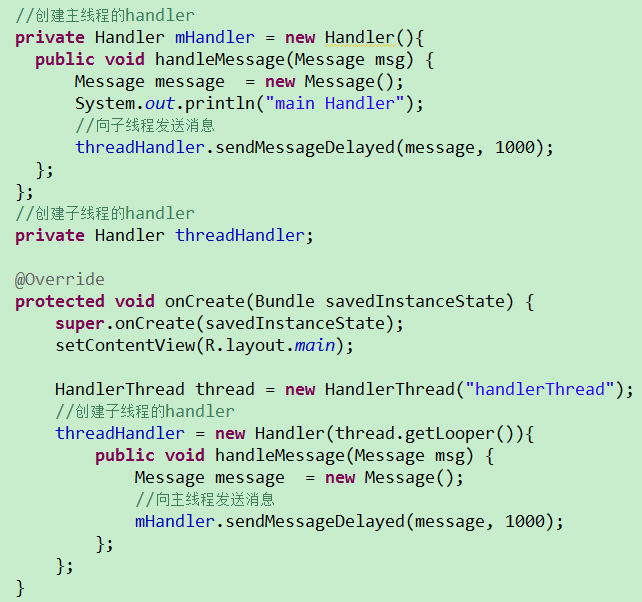
1. HandlerThread是什么

HandlerThread继承于Thread，所以它本质就是个Thread。与普通Thread的差别就在于，它有个Looper成员变量。这个Looper其实就是对消息队列以及队列处理逻辑的封装，简单说就是 消息队列+消息循环。

当我们需要一个工作者线程，而不是把它当作一次性消耗品，用过即废弃的话，就可以使用它。



1. **主线程与子线程之间的信息交互**



1. **Android中更新UI的几种方式：**

Android中更新UI的4种方式：1、runOnUiThread 2、handler的 post 3、handler 的sendMessage 4、View自身的post