**安卓线程通信机制**

安卓是一个多线程的操作系统，与其他操作系统类似，不同线程之间的通信需要消息来传递，安卓多线程之间通信的消息机制需要handler, looper, message这几个对象共同完成。

安卓默认会有一个主线程，在任何安卓编译器中，新生成的工程中，一定有一个类似的叫做MainActivity的类，这个就是任何一个安卓程序的入口，与C工程中的main()函数相类似，安卓程序从这里开始执行。

而安卓的主线程ActivityThread就被封装在这个MainActivity类中，称ActivityThread所在的线程为UI线程或主线程，它默认包含一个消息队列， Message Queue，安卓规定，所有修改UI的操作都必须在主线程中完成。如果要实现在其他的线程中修改UI的操作，或者回传一些消息到主线程，那么就需要使用安卓的异步消息机制来实现。

简单来说，安卓的消息message都需要通过looper来发布，而looper发布消息则是从消息队列Message queue中提取的，而handler则是将message放入message queue，在新的子线程中，他们之间的关系可以简单表述为：

首先在主线程中new一个子线程出来并且启动这个子线程

获取当前线程的looper对象

New一个handler出来用于发布和处理消息message

调用looper对象的loop方法，启动循环处理消息机制，类似于uC/OS-II中每个子线程都有一个while(1)，保证每个子线程都是一个死循环

所以handler, looper, message queue这几个概念的关系可以更加直接的描述为：

1. 安卓中有一个共用的消息队列message queue，消息的发布和处理，都需要消息处理者handler来完成
2. Handler负责将需要传递的信息封装成为message，通过handler.obtainMessage()方法实现
3. 然后将message传递给looper，这是通过handler.sendMessage()来实现的
4. 接下来由looper将message放入message queue中
5. 当looper对象看到message queue中有message时，就会将其广播出去
6. 当handler对象收到该消息后，就会调用相应的handler.handlerMessage()方法对接收到的消息进行处理

所以理论上来说，handler和thread没有关系，如果说新起了一个子线程，却并不需要发布或者处理消息的话，那么就不需要新定义一个handler来处理。

Looper对象的获取可以分为两种方法：

Looper.getMainLooper(); //用于获取主线程的looper，方便与主线程之间进行通信

Looper.myLooper(); //用于获取当前线程绑定的looper对象

举个子线程与主线程之间通信的例子：

import android.app.Activity;

import android.os.Bundle;

import android.os.Handler;

import android.os.HandlerThread;

import android.os.Looper;

import android.os.Message;

import android.view.Window;

/\*目的：**开启一个子线程且实现线程之间的通讯(主线程发送消息给子线程的handler处理)**

handler并不是创建了一个线程，而是在原有的线程上加了一个方法去调用Run方法

所以需要解决这个问题的方法就是利用HandlerThread开启一个线程，再用handler实现通讯

通讯的原因：msg.sendToTarget();--执行对应的--handleMessage方法

\*/

public class MainActivity extends Activity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

requestWindowFeature(Window.FEATURE\_NO\_TITLE);

setContentView(R.layout.activity\_main);

System.out.println("Activity-->"+Thread.currentThread().getName());//主线程

//实现了使用Looper来处理消息队列的功能

HandlerThread handlerThread=new HandlerThread("handler\_Thread");

handlerThread.start();//开启线程

MyHandler myHandler=new MyHandler(handlerThread.getLooper());//必须在上一条代码的下面，这样就是开启了一个新的线程，实际上时获取了子线称的looper对象

Message msg = myHandler.obtainMessage();//获取发送信息的对象，已经指明了时来自于子线程的

msg.arg1=23;//需要发送的数据

msg.obj="abc";//发送对象

Bundle data=new Bundle();//通过捆绑发送更复杂的数据

data.putString("键", "value");//通过键值对是形式捆绑数据

msg.setData(data);//将数据发送出去

//将信息发送到目标对象，目标对象指的是生成msg的对象的对象

msg.sendToTarget();//也就会去执行相对应的handleMessage方法，msg是由myHandler对象封装的，那么sendToTarget()就会将这个msg发送到myhandler对象中，由myHandler中的handleMessage()方法进行处理

System.out.println("主线程-->"+Thread.currentThread().getName());

}

//创建子线程的handler处理主线程发送来的消息

class MyHandler extends Handler{

@Override

public void handleMessage(Message msg) {//处理信息，相当于一个线程

//在子线程运行的,线程名就为handler\_Thread(上面创建的)

System.out.println("MyhandlerMessage-->"+Thread.currentThread().getName());

int arg=msg.arg1;//通过msg对象获取传过来的数据

String str=(String) msg.obj;//获取数据

Bundle data = msg.getData();//获取捆绑的对象

String string = data.getString("键");//通过键获取值

System.out.println(arg+ str + string);

}

public MyHandler(Looper looper) {

super(looper);

}

}

}

所以安卓中用子线称来发布或接受，处理消息是很简单且容易的，具体做法归纳如下：

1. 在主线程中实例化一个myhandler对象，在这个对象中重写handleMessage方法，比如可以在这里更新UI
2. 在主线程中起一个新线程mythread，在这个线程中，需要发布一些消息来改变UI
3. 在子线称中，通过myhandler.obtainMessage()方法指明，这条msg是由名叫myhandler的处理器封装的，那么也就指定了接受这条消息的处理器就是myhandler，一旦发布消息，则会转入myhander这个对象中的handlerMessage()方法执行其中的代码，更新UI