[×](https://blog.csdn.net/bfboys/article/details/52640587)

# 接口回调本质就这么简单

2016年09月23日 17:26:22

阅读数：1496

**一、为什么会有接口回调？什么是接口回调？**

其实这两个问题是一个问题，知道了接口回调的原理自然就知道了为什么会有这么个东西。我们知道java中接口是不可以直接创建实例的，那么问题来了，假如我把一个接口声明为一个变量，那么我执行这个接口中的方法，接口没有实例它该怎么办呢？啊哈，这里自然又改出现java中的另一个特性---“多态”，这时java虚拟机自然会去找其子类，调用其子类中已经重载的该方法，这里就是接口回调的本质！！我们只需要给该变量指向其子类的地址就可以在调用的时候知道调用子类的方法。那么我们就可以在A类中创建接口的子类实例，在B类中创建一个接口的变量，把A类的地址传给B类的变量，在变量执行接口中的方法的时候就会调用A类中重写的方法，这就是接口回调的执行步骤。我们在网络请求等耗时的操作的时候会使用到该机制，用来把得到的数据传回主线程中。

**二、使用实例**

下面我们就来用一个网络请求的实例来演示如何使用该机制，为了尽量的减少代码，使主要代码能够突出，里面的一些不必要的容错代码就没有写了，大家在正式的项目中希望可以做好容错处理。

我们用一个按钮和一个imageView来演示，当点击按钮的时候去下载数据，在数据下载完成之后使用接口回调把数据传回来显示在imageview中。效果如下：

布局文件很简单就不在贴出来了，我们有3个类，一个是回调的接口，一个是主activity，一个为执行下载并且把数据回调回来的Callee类，下面来看接口类：

**[java]** [view plain](http://blog.csdn.net/qq379454816/article/details/51546638) [copy](http://blog.csdn.net/qq379454816/article/details/51546638)

[print?](http://blog.csdn.net/qq379454816/article/details/51546638)

1. import android.graphics.Bitmap;
3. /\*\*
4. \* Created by JimLv on 2016/5/31.
5. \*/
6. public interface CallBackInterface {
7. void result(Bitmap bm);
8. }

定义了一个用于传递数据的方法，来看看Callee类：

**[java]** [view plain](http://blog.csdn.net/qq379454816/article/details/51546638) [copy](http://blog.csdn.net/qq379454816/article/details/51546638)

[print?](http://blog.csdn.net/qq379454816/article/details/51546638)

1. import java.io.IOException;
2. import java.net.HttpURLConnection;
3. import java.net.URL;
5. /\*\*
6. \* Created by JimLv on 2016/5/31.
7. \*/
8. public class Callee {
9. private final String IMGURL = "http://f.hiphotos.baidu.com/zhidao/pic/item/b21bb051f8198618a323ac464bed2e738ad4e688.jpg";
10. //持有接口变量
11. CallBackInterface mInterface;
13. Callee(CallBackInterface theInterface) {
14. //这里是关键，把外部的接口实例引用到该类，给变量赋值
15. mInterface = theInterface;
16. //创建对象的时候执行下载
17. executeDown();
18. }
20. public void executeDown() {
21. new Thread(new Runnable() {
22. @Override
23. public void run() {
24. try {
25. HttpURLConnection conn = (HttpURLConnection) new URL(IMGURL).openConnection();
26. conn.setConnectTimeout(5 \* 1000);
27. conn.setRequestMethod("GET");
28. mInterface.result(BitmapFactory.decodeStream(conn.getInputStream()));
29. } catch (IOException e) {
30. e.printStackTrace();
31. }
32. }
33. }).start();
34. }
35. }

在这里我们声明了一个接口变量，在类的初始化方法中把接口的子类的地址赋给该变量并且调用下载数据的方法，**在数据下载完成之后调用接口变量的方法把数据传给该方法，执行该方法实际是执行子类的该方法，这就是接口回调真正做的事。**  
最后我们来看MainActivity类：

**[java]** [view plain](http://blog.csdn.net/qq379454816/article/details/51546638) [copy](http://blog.csdn.net/qq379454816/article/details/51546638)

[print?](http://blog.csdn.net/qq379454816/article/details/51546638)

1. public class MainActivity extends AppCompatActivity implements View.OnClickListener {
2. private ImageView imgv;
3. private Button downImgBtn;
5. @Override
6. protected void onCreate(Bundle savedInstanceState) {
7. super.onCreate(savedInstanceState);
8. setContentView(R.layout.activity\_main);
9. imgv = (ImageView) findViewById(R.id.imgv);
10. downImgBtn = (Button) findViewById(R.id.downImgBtn);
11. //点击按钮去下载
12. downImgBtn.setOnClickListener(this);
13. }
15. @Override
16. public void onClick(View v) {
17. //从接口回调得到数据,因为下载是在子线程中，所以这里要变到主线程中设置图片
18. new Callee(new CallBackInterface() {
19. @Override
20. public void result(final Bitmap bm) {
21. runOnUiThread(new Runnable() {
22. @Override
23. public void run() {
24. imgv.setImageBitmap(bm);
25. }
26. });
27. }
28. });
29. }
30. }

在这里我们用匿名内部类创建该接口的子类实例，子类重载的方法中因为是运行在子线程中的，我们无法操作UI，所有调用了runOnUiThread方法，把数据更新到imageview中，整个的demo就这么点是不是很简单呢？

## 不良信息举报

窗体顶端

举报内容：

接口回调本质就这么简单

举报原因：

色情 政治 抄袭 广告 招聘 骂人 其他

原文地址：



原因补充：



最多只允许输入30个字



窗体底端