项iang﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽因此需要上网去查找，**Lab9 实验报告**

*13331231*

*孙圣*

一、实验目的

1.练习使用KSOAP2进行网络连接

2.复习使用Handler更新UI

二、实验内容

实现自定义验证码的功能，根据用户的输入（字母，数字或中文），产生相应的验证码图片。附加功能是：进入程序时随机产生验证码图片，并判断用户是否正确输入验证码。

三、实验步骤

1. 先实现出基本UI。

2. Create：

Create负责完成基本项，即根据用户的输入生成所需要的验证码。首先要定义一个public的静态变量handler，用来处理Download子线程传递过来的message。

根据message的类型进行操作，如果类型为UPDATE\_CONTENT，则将数据流解码为bitmap：先将字符串解码为Base64，然后再将字节流转换为bitmap。之后设置UI界面验证码处的图片。

按下创建的按钮后，运行Download子线程，同时显示progressDialog。

3. Download：

Download类实现了Runnable。在其中利用KSOAP2进行网络连接。

要使用KSOAP2，首先要设置依赖，同时还要在Manifest中设置

允许网络访问。

首先，要先指明NAMESPACE（命名空间），METHODNAME（要调用的方法名），SOAPACTION和URL。

之后根据命名空间和方法名创建一个SoapObject。再利用addProperty();方法添加我们所需要请求的参数和数据。

然后生成envelop，通过setOutputSoapObject();方法将刚才设定好的的请求与envelop结合起来。之后通过HttpTransportSE调用call()方法，请求WebService。

通过bodyIn和getProperty();方法来得到请求返回的数据，保存于message的obj中。

最后，由于handler和progressDialog都为静态变量，我们可以直接通过Create.的方式访问它们。因此我们将message发送，从而更新UI，并使progressDialog不再显示。

4. 附加项 AutoCode和MainActivity：

这两者基本和前面实现的一样。

但是对于MainActivity一开始要随机生成一个8位的字符串。首先先定义一个字符串，其中包括了所需要的所有字符（a-zA-Z0-9）。之后通过random类实现随机选择下标。并将选到的字符添加到StringBuffer中。重复8次后，就得到了随机的字符串。并在AutoCode中使用。

之后根据用户的输入进行判断，利用Toast来显示正误信息。

四、实验结果

创建验证码：



附加项（随机生成验证码）：



五、实验总结

这次实验和上次差不多，只不过上一次是通过HttpURLConnection来请求Web服务，这一次是利用KSOAP来实现。总的说来，两者各有优劣。前者比较直观，基本上对于每个设置都能清楚对应的功能。但是缺点在于，得到了相应的xml数据之后还需要通过PullParser来解析，如果返回的条目较多，解析就变得相当复杂。对于KSOAP来说，一开始要定义四个变量，如果不熟悉的话很容易就不知所措。其实是要根据Web服务中对SOAP的请求的说明来对应的填写。不过，KSOAP的请求和获取结果都比较方便，利用HttpTransportSE的call();方法就能完成请求。而且getProperty();方法相当于直接将PullParser封装起来，能够利用它获得每个tag的数据。

这次实验中一直遇到个问题没有解决，就是在网络连接不好甚至没有网络的情况下，会出现WindowLeaked的异常。上网查找资料后发现原因是：Activity结束后progressDialog却没有结束。解决方法大概是在发现网络连接异常时将progressDialog停止，但试了几种方法依然不能够解决。

参考资料：

1. 随机生成字符串

<http://blog.csdn.net/xiaoyukid/article/details/6041697>

2. android.view.WindowLeaked

<http://stackoverflow.com/questions/11590382/android-view-windowleaked>

3. KSOAP

<http://kobjects.org/ksoap2/doc/api/>

联系方式：

Email：[363419254@qq.com](mailto:363419254@qq.com)