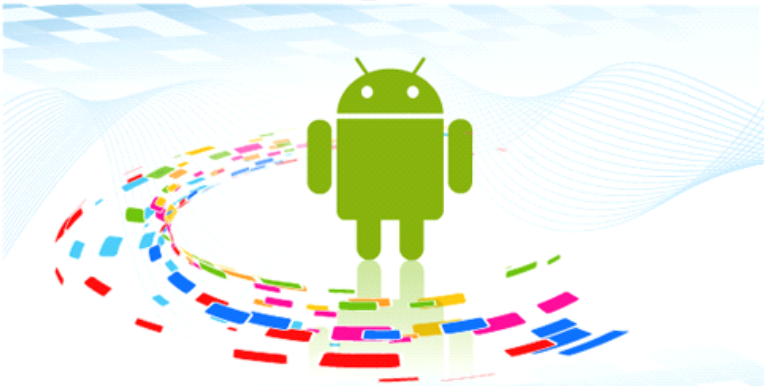
eoe特刊

第十二期：Android network processing



Android 网络处理

eoeandroid.com
做最棒的Android开发社区

MOBILE eoe 易联致远 | 优亿市场 Android应用发布与分享平台

目录

【Android 网路处理详解】

1.1 Android 网路通信之 HTTP 经典讲解

HTTP 协议是网络编程的主要组成部分。不论是电脑网络程序开发还是手机网络程序开发，必须掌握的通信协议。

1.1.1 什么是 HTTP 协议

HTTP 协议是一种应用层协议，HTTP 是 HyperText Transfer Protocol(超文本传输协议)的英文缩写。HTTP 可以通过传输层的 TCP 协议在客户端和服务器之间传输数据。

HTTP 协议主要用于 Web 浏览器和 Web 服务器之间的数据交换。我们在使用 IE 或 Firefox 浏览网页或下载 Web 资源时，通过在地址栏中输入 http://host:port/path，开头的 4 个字母 http 就相当于通知浏览器使用 HTTP 协议来和 host 所确定的服务器进行通讯。目前主要有两个版本 http1.0 和 http1.1。本文主要讲解 http 的工作原理。

1.1.2 HTTP 的工作方式

HTTP 协议采用了请求/响应的工作方式。

1、 HTTP1.0 工作方式：

基于 HTTP1.0 协议的客户端在每次向服务器发出请求后，服务器就会向客户端返回响应消息（包括请求是否正确以及所请求的数据），在确认客户端已经收到响应消息后，服务端就会关闭网络连接（其实是关闭 TCP 连接）。在这个数据传输过程中，并不保存任何历史信息和状态信息，因此，HTTP 协议也被认为是无状态的协议。

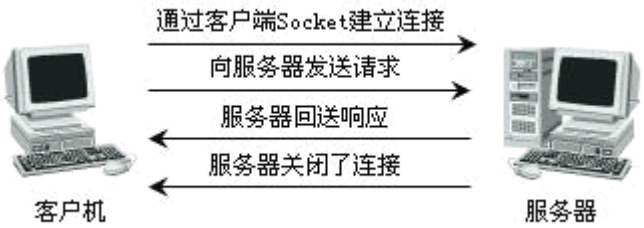


图 1：描绘了 HTTP1.0 协议的通讯过程。

在 HTTP1.0 协议中，当 Web 浏览器发出请求时，就意味着一个请求/响应会话已经开始。在请求、响应结束后，服务器就会立刻关闭这个连接。这种会话方式虽然简便，但它会带来另外一个问题。如果客户端浏览器访问的某个 HTML 或其他类型的 Web 页中包含有其他的 Web 资源，如 JavaScript 文件、图像文件、CSS 文件等；当浏览器每遇到这样一个 Web 资源，就会建立一个 HTTP 会话。如果这样的资源很多的话，就会加重服务器的负担，同时也会影响客户端浏览器加载 HTML 等 Web 资源的效率。

在对上述的缺陷进行改进和完善后，HTTP1.1 协议进入了我们的视线

2、 HTTP1.1 工作方式：

HTTP1.1 和 HTTP1.0 相比较而言，最大的区别就是增加了持久连接支持。当客户端使用 HTTP1.1 协议连接到服务器后，服务器就将关闭客户端连接的主动权交还给客户端；也就是说，在客户端向服务器发送一个

请求并接收以一个响应后，只要不调用 Socket 类的 close 方法关闭网络连接，就可以继续向服务器发送 HTTP 请求。当 HTML 中含有其他的 Web 资源时，浏览器就可以使用同一个网络连接向下载这些资源，这样就可以大大减轻服务器的压力。图 2 演示了这一过程。

图 2 演示了这一过程。

HTTP1.1 除了支持持久连接外，还将 HTTP1.0 的请求方法从原来的三个(GET、POST 和 HEAD)扩展到了八个（OPTIONS、GET、HEAD、POST、PUT、DELETE、TRACE 和 CONNECT）。而且还增加了很多请求和响应字段，如上述的持久连接的字段 Connection。这个字段有两个值，Close 和 Keep-Alive。如果使用 Connection:Close，则关闭 HTTP1.1 的持久连接的功能，要打开 HTTP1.1 的持久连接的功能，必须使用 Connection:Keep-Alive，或者不加 Connection 字段（因为 HTTP1.1 在默认情况下就是持久连接的）。除了这些，还提供了身份认证、状态管理和缓存（Cache）等相关的请求头和响应头。

（作者：MrJing ）

1.1.3 Android 中的 http 通信具体代码

要说 Android 网络通讯平台支持还是比较丰富的，除了兼容 J2ME 中的 java.net api 外还提供了一些 Android 平台独有的类 android.net 这个 Package，似乎更强大的是 org.apache.http 类，这个是 apache 实验室开源的包，对于 Http 请求处理很方便

1、java.net.HttpURLConnection:

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import android.app.Activity;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class Activity01 extends Activity
{
        private final String DEBUG_TAG = "Activity02";
        private TextView mTextView;
        private Button mButton;
        /** Called when the activity is first created. */
        @Override
        public void onCreate(Bundle savedInstanceState)
```

```
{
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        mTextView = (TextView)this.findViewById(R.id.TextView01);
        mButton = (Button)this.findViewById(R.id.Button01);
        mButton.setOnClickListener(new Button.OnClickListener()
   {
    @Override
    public void onClick(View arg0)
    {
     // TODO Auto-generated method stub
       refresh();
    }
   });
        //开启线程
        new Thread(mRunnable).start();
}
//刷新网页显示
private void refresh()
{
        String httpUrl = " http://wap.sohu.com/ ";
        String resultData = "";
        URL url = null;
        try
        {
                // 构造一个 URL 对象
                url = new URL(httpUrl);
        }
        catch (MalformedURLException e)
        {
                Log.e(DEBUG_TAG, "MalformedURLException");
        }
        if (url != null)
        {
                try
                {
                        // 使用 HttpURLConnection 打开连接
                        HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
                        // 得到读取的内容(流)
                        InputStreamReader in = new InputStreamReader(urlConn.getInputStream());
                        // 为输出创建 BufferedReader
                        BufferedReader buffer = new BufferedReader(in);
                        String inputLine = null;
                        // 使用循环来读取获得的数据
                        while (((inputLine = buffer.readLine()) != null))
                        {
                                // 我们在每一行后面加上一个"\n"来换行
                                resultData += inputLine + "\n";
                        }
                        // 关闭 InputStreamReader
                        in.close();
                        // 关闭 http 连接
                        urlConn.disconnect();
                        // 设置显示取得的内容
```

```
                        if (resultData != null)
                        {
                                mTextView.setText(resultData);
                        }
                        else
                        {
                                mTextView.setText("读取的内容为 NULL");
                        }
                }
                catch (IOException e)
                {
                        Log.e(DEBUG_TAG, "IOException");
                }
        }
        else
        {
                Log.e(DEBUG_TAG, "Url NULL");
        }
}
 private Runnable mRunnable = new Runnable()
 {
  public void run()
  {
   while (true)
   {
    try
    {
     Thread.sleep(5 * 1000);
     //发送消息
     mHandler.sendMessage(mHandler.obtainMessage());

    } catch (InterruptedException e)
    {
     // TODO Auto-generated catch block
        Log.e(DEBUG_TAG, e.toString());
    }
   }
  }
 };
 Handler mHandler = new Handler()
 {
  public void handleMessage(Message msg)
  {
   super.handleMessage(msg);
   //刷新
   refresh();
  }
 };
}
```

2、 java.net.URL

1） 直接获取数据

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

//直接获取数据
public class Activity02 extends Activity
{
	private final String DEBUG_TAG = "Activity02";
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState)
	{
		super.onCreate(savedInstanceState);
		setContentView(R.layout.http);
		TextView mTextView = (TextView)this.findViewById(R.id.TextView_HTTP);
		//http 地址
		String httpUrl = " http://wap.sohu.com";
		//获得的数据
		String resultData = "";
		URL url = null;
		try
		{
			//构造一个 URL 对象
			url = new URL(httpUrl);
		}
		catch (MalformedURLException e)
		{
			Log.e(DEBUG_TAG, "MalformedURLException");
		}
		if (url != null)
		{
			try
			{
				//使用 HttpURLConnection 打开连接
				HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
				//得到读取的内容(流)
				InputStreamReader in = new InputStreamReader(urlConn.getInputStream());
				// 为输出创建 BufferedReader
				BufferedReader buffer = new BufferedReader(in);
				String inputLine = null;
```

```
                        //使用循环来读取获得的数据
                        while (((inputLine = buffer.readLine()) != null))
                        {
                                //我们在每一行后面加上一个"\n"来换行
                                resultData += inputLine + "\n";
                        }
                        //关闭 InputStreamReader
                        in.close();
                        //关闭 http 连接
                        urlConn.disconnect();
                        //设置显示取得的内容
                        if ( resultData != null )
                        {
                                mTextView.setText(resultData);
                        }
                        else
                        {
                                mTextView.setText("读取的内容为 NULL");
                        }
                }
                catch (IOException e)
                {
                        Log.e(DEBUG_TAG, "IOException");
                }
        }
        else
        {
                Log.e(DEBUG_TAG, "Url NULL");
        }
        //设置按键事件监听
        Button button_Back = (Button) findViewById(R.id.Button_Back);
        /* 监听 button 的事件信息 */
        button_Back.setOnClickListener(new Button.OnClickListener()
        {
                public void onClick(View v)
                {
                        /* 新建一个 Intent 对象 */
                        Intent intent = new Intent();
                        /* 指定 intent 要启动的类 */
                        intent.setClass(Activity02.this, Activity01.class);
                        /* 启动一个新的 Activity */
                        startActivity(intent);
                        /* 关闭当前的 Activity */
                        Activity02.this.finish();
                }
        });
    }
}
```

2） 以 Get 方式

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
//以 Get 方式上传参数
public class Activity03 extends Activity
{
	private final String DEBUG_TAG = "Activity03";
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState)
	{
		super.onCreate(savedInstanceState);
		setContentView(R.layout.http);
		TextView mTextView = (TextView)this.findViewById(R.id.TextView_HTTP);
		//http 地址"?par=abcdefg"是我们上传的参数
		String httpUrl = "http://192.168.1.110:8080/httpget.jsp?par=abcdefg";
		//获得的数据
		String resultData = "";
		URL url = null;
		try
		{
			//构造一个 URL 对象
			url = new URL(httpUrl);
		}
		catch (MalformedURLException e)
		{
			Log.e(DEBUG_TAG, "MalformedURLException");
		}
		if (url != null)
		{
			try
			{
				// 使用 HttpURLConnection 打开连接
				HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
				//得到读取的内容(流)
				InputStreamReader in = new InputStreamReader(urlConn.getInputStream());
				// 为输出创建 BufferedReader
				BufferedReader buffer = new BufferedReader(in);
				String inputLine = null;
				//使用循环来读取获得的数据
				while (((inputLine = buffer.readLine()) != null))
				{
```

```
                        //我们在每一行后面加上一个"\n"来换行
                        resultData += inputLine + "\n";
                    }
                    //关闭 InputStreamReader
                    in.close();
                    //关闭 http 连接
                    urlConn.disconnect();
                    //设置显示取得的内容
                    if ( resultData != null )
                    {
                        mTextView.setText(resultData);
                    }
                    else
                    {
                        mTextView.setText("读取的内容为 NULL");
                    }
                }
                catch (IOException e)
                {
                    Log.e(DEBUG_TAG, "IOException");
                }
            }
            else
            {
                Log.e(DEBUG_TAG, "Url NULL");
            }
            Button button_Back = (Button) findViewById(R.id.Button_Back);
            /* 监听 button 的事件信息 */
            button_Back.setOnClickListener(new Button.OnClickListener()
            {
                public void onClick(View v)
                {
                    /* 新建一个 Intent 对象 */
                    Intent intent = new Intent();
                    /* 指定 intent 要启动的类 */
                    intent.setClass(Activity03.this, Activity01.class);
                    /* 启动一个新的 Activity */
                    startActivity(intent);
                    /* 关闭当前的 Activity */
                    Activity03.this.finish();
                }
            });
        }
}
```

3） 以 post 方式

```
import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

以 post 方式上传参数
public class Activity04  extends Activity
{
	private final String DEBUG_TAG = "Activity04";
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState)
	{
		super.onCreate(savedInstanceState);
		setContentView(R.layout.http);

		TextView mTextView = (TextView)this.findViewById(R.id.TextView_HTTP);
		//http 地址"?par=abcdefg"是我们上传的参数
		String httpUrl = "http://192.168.1.110:8080/httpget.jsp";
		//获得的数据
		String resultData = "";
		URL url = null;
		try
		{
			//构造一个 URL 对象
			url = new URL(httpUrl);
		}
		catch (MalformedURLException e)
		{
			Log.e(DEBUG_TAG, "MalformedURLException");
		}
		if (url != null)
		{
			try
			{
				// 使用 HttpURLConnection 打开连接
				HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
				//因为这个是 post 请求,设立需要设置为 true
				urlConn.setDoOutput(true);
				urlConn.setDoInput(true);
			// 设置以 POST 方式
```

```
                urlConn.setRequestMethod("POST");
            // Post 请求不能使用缓存
                urlConn.setUseCaches(false);
                urlConn.setInstanceFollowRedirects(true);
            // 配置本次连接的 Content-type，配置为 application/x-www-form-urlencoded 的
                urlConn.setRequestProperty("Content-Type","application/x-www-form-
urlencoded");
            // 连接，从 postUrl.openConnection()至此的配置必须要在 connect 之前完成，
            // 要注意的是 connection.getOutputStream 会隐含的进行 connect。
                urlConn.connect();
                //DataOutputStream 流
            DataOutputStream out = new DataOutputStream(urlConn.getOutputStream());
            //要上传的参数
            String content = "par=" + URLEncoder.encode("ABCDEFG", "gb2312");
            //将要上传的内容写入流中
            out.writeBytes(content);
            //刷新、关闭
            out.flush();
            out.close();
            //获取数据
            BufferedReader reader = new BufferedReader(new
InputStreamReader(urlConn.getInputStream()));
                String inputLine = null;
                //使用循环来读取获得的数据
                while (((inputLine = reader.readLine()) != null))
                {
                    //我们在每一行后面加上一个"\n"来换行
                    resultData += inputLine + "\n";
                }
                reader.close();
                //关闭 http 连接
                urlConn.disconnect();
                //设置显示取得的内容
                if ( resultData != null )
                {
                    mTextView.setText(resultData);
                }
                else
                {
                    mTextView.setText("读取的内容为 NULL");
                }
            }
            catch (IOException e)
            {
                Log.e(DEBUG_TAG, "IOException");
            }
        }
        else
        {
            Log.e(DEBUG_TAG, "Url NULL");
        }

        Button button_Back = (Button) findViewById(R.id.Button_Back);
        /* 监听 button 的事件信息 */
```

```
		button_Back.setOnClickListener(new Button.OnClickListener()
		{
			public void onClick(View v)
			{
				/* 新建一个 Intent 对象 */
				Intent intent = new Intent();
				/* 指定 intent 要启动的类 */
				intent.setClass(Activity04.this, Activity01.class);
				/* 启动一个新的 Activity */
				startActivity(intent);
				/* 关闭当前的 Activity */
				Activity04.this.finish();
			}
		});
	}
}
```

3、 org.apache.http

```
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class Activity01 extends Activity
{
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState)
	{
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);

		Button button_Get = (Button) findViewById(R.id.Button_Get);
		/* 监听 button 的事件信息 */
		button_Get.setOnClickListener(new Button.OnClickListener() {
			public void onClick(View v)
			{
				/* 新建一个 Intent 对象 */
				Intent intent = new Intent();
				/* 指定 intent 要启动的类 */
				intent.setClass(Activity01.this, Activity02.class);
				/* 启动一个新的 Activity */
				startActivity(intent);
				/* 关闭当前的 Activity */
				Activity01.this.finish();
			}
		});

		Button button_Post = (Button) findViewById(R.id.Button_Post);
		/* 监听 button 的事件信息 */
		button_Post.setOnClickListener(new Button.OnClickListener() {
```

```
                public void onClick(View v)
                {
                        /* 新建一个 Intent 对象 */
                        Intent intent = new Intent();
                        /* 指定 intent 要启动的类 */
                        intent.setClass(Activity01.this, Activity03.class);
                        /* 启动一个新的 Activity */
                        startActivity(intent);
                        /* 关闭当前的 Activity */
                        Activity01.this.finish();
                }
        });
    }
}
```

```
import java.io.IOException;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class Activity02 extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.http);
        TextView mTextView = (TextView) this.findViewById(R.id.TextView_HTTP);
        // http 地址
        String httpUrl = "http://192.168.1.110:8080/httpget.jsp?par=HttpClient_android_Get";
        //HttpGet 连接对象
        HttpGet httpRequest = new HttpGet(httpUrl);
        try
        {
            //取得 HttpClient 对象
            HttpClient httpclient = new DefaultHttpClient();
            //请求 HttpClient，取得 HttpResponse
            HttpResponse httpResponse = httpclient.execute(httpRequest);
            //请求成功
            if (httpResponse.getStatusLine().getStatusCode() == HttpStatus.SC_OK)
            {
```

```
                        //取得返回的字符串
                        String strResult = EntityUtils.toString(httpResponse.getEntity());
                        mTextView.setText(strResult);
                }
                else
                {
                        mTextView.setText("请求错误!");
                }
        }
        catch (ClientProtocolException e)
        {
                mTextView.setText(e.getMessage().toString());
        }
        catch (IOException e)
        {
                mTextView.setText(e.getMessage().toString());
        }
        catch (Exception e)
        {
                mTextView.setText(e.getMessage().toString());
        }

        //设置按键事件监听
        Button button_Back = (Button) findViewById(R.id.Button_Back);
        /* 监听 button 的事件信息 */
        button_Back.setOnClickListener(new Button.OnClickListener()
        {
                public void onClick(View v)
                {
                        /* 新建一个 Intent 对象 */
                        Intent intent = new Intent();
                        /* 指定 intent 要启动的类 */
                        intent.setClass(Activity02.this, Activity01.class);
                        /* 启动一个新的 Activity */
                        startActivity(intent);
                        /* 关闭当前的 Activity */
                        Activity02.this.finish();
                }
        });
    }
}
```

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.HttpStatus;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
```

```
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.util.EntityUtils;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class Activity03 extends Activity
{
	/** Called when the activity is first created. */
	@Override
	public void onCreate(Bundle savedInstanceState)
	{
		super.onCreate(savedInstanceState);
		setContentView(R.layout.http);
		TextView mTextView = (TextView) this.findViewById(R.id.TextView_HTTP);
		// http 地址
		String httpUrl = "http://192.168.1.110:8080/httpget.jsp";
		//HttpPost 连接对象
		HttpPost httpRequest = new HttpPost(httpUrl);
		//使用 NameValuePair 来保存要传递的 Post 参数
		List<NameValuePair> params = new ArrayList<NameValuePair>();
		//添加要传递的参数
		params.add(new BasicNameValuePair("par", "HttpClient_android_Post"));
		try
		{
			//设置字符集
			HttpEntity httpentity = new UrlEncodedFormEntity(params, "gb2312");
			//请求 httpRequest
			httpRequest.setEntity(httpentity);
			//取得默认的 HttpClient
			HttpClient httpclient = new DefaultHttpClient();
			//取得 HttpResponse
			HttpResponse httpResponse = httpclient.execute(httpRequest);
			//HttpStatus.SC_OK 表示连接成功
			if (httpResponse.getStatusLine().getStatusCode() == HttpStatus.SC_OK)
			{
				//取得返回的字符串
				String strResult = EntityUtils.toString(httpResponse.getEntity());
				mTextView.setText(strResult);
			}
			else
			{
				mTextView.setText("请求错误!");
			}
		}
		catch (ClientProtocolException e)
		{
			mTextView.setText(e.getMessage().toString());
		}
```

```
                catch (IOException e)
                {
                        mTextView.setText(e.getMessage().toString());
                }
                catch (Exception e)
                {
                        mTextView.setText(e.getMessage().toString());
                }
                //设置按键事件监听
                Button button_Back = (Button) findViewById(R.id.Button_Back);
                /* 监听 button 的事件信息 */
                button_Back.setOnClickListener(new Button.OnClickListener()
                {
                        public void onClick(View v)
                        {
                                /* 新建一个 Intent 对象 */
                                Intent intent = new Intent();
                                /* 指定 intent 要启动的类 */
                                intent.setClass(Activity03.this, Activity01.class);
                                /* 启动一个新的 Activity */
                                startActivity(intent);
                                /* 关闭当前的 Activity */
                                Activity03.this.finish();
                        }
                });
        }
}
```


1.2 Android 网络多线程断点下载

我们编写的是 Andorid 的 HTTP 多线程断点下载应用程序。因为之间我们学习的学习积累，直接使用单线程下载 HTTP 文件对我们来说是一件非常简单的事。那么，多线程断点下载的难点在哪里？

1.多线程下载

2.支持断点

多线程下载：

如何才能从文件的指定位置处开始下载文件？（比如从 50MB 开始）这一点我们可以通过 HTTP 请求信息头来设置，还记得 HTTP 请求信息头的“Range”属性吗？

断点：

首要问题（多线程下载）已经被我们解决了，支持断点下载想必大家也已经想到了。就是将下载的进度保存到文件中，但在 Android 中却不能这么做。通过老黎的试验，在 Android 平台中，我们需要向文件中写出下载的文件数据，还需要向另一个文件中写出下载进度，这样会出错。这样会导致有一个文件的内容没有被写出。所以我们就不能以文件的方式来保存下载进度，但可以通过数据库的方式保存下载进度。

这两大问题我们已经有了解决思路，那么就开始动手编写吧！

1.2.1 创建 Android 工程

Project name:MulThreadDownloader
BuildTarget:Android2.1
Application name：多线程断点下载
Package name:com.changcheng.download
Create Activity:MulThreadDownloader
Min SDK Version:7

1.2.2 AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.changcheng.download"
android:versionCode="1"
android:versionName="1.0">
<application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".MulThreadDownloader"  android:label="@string/app_name">
                <intent-filter>
                        <action android:name="android.intent.action.MAIN" />
                        <category android:name="android.intent.category.LAUNCHER" />
                </intent-filter>
        </activity>
</application>
<uses-sdk android:minSdkVersion="7" />
<!-- 在 SDCard 中创建与删除文件权限 -->
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"/>
<!-- 往 SDCard 写入数据权限 -->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<!-- 访问 internet 权限 -->
<uses-permission android:name="android.permission.INTERNET"/>
</manifest>
```

1.2.3 strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
<string name="hello">Hello World, DownloadActivity!</string>
<string name="app_name">多线程断点下载</string>
<string name="path">下载路径</string>
<string name="downloadbutton">下载</string>
<string name="sdcarderror">SDCard 不存在或者写保护</string>
</resources>
```

1.2.4 main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
                        android:orientation="vertical"
                        android:layout_width="fill_parent"
                        android:layout_height="fill_parent"
>
```

```
<!-- 下载路径 -->
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="@string/path"
/>

<EditText
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="http://www.winrar.com.cn/download/wrar380sc.exe"
android:id="@+id/path"
/>

<!-- 下载按钮 -->
<Button
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="@string/downloadbutton"
android:id="@+id/button"
/>

<!-- 进度条 -->
<ProgressBar
android:layout_width="fill_parent"
android:layout_height="20dip"
style="?android:attr/progressBarStyleHorizontal"
android:id="@+id/downloadbar"/>
<!-- 进度% -->
<TextView
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:gravity="center"
android:id="@+id/resultView"
/>
</LinearLayout>
```


1.2.5 MulThreadDownloader

```
package com.changcheng.download;

import java.io.File;
import com.changcheng.net.download.DownloadProgressListener;
import com.changcheng.net.download.FileDownloader;
import com.changcheng.download.R;
import android.app.Activity;
import android.os.Bundle;
import android.os.Environment;
import android.os.Handler;
import android.os.Message;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
```

```
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

public class MulThreadDownloader extends Activity {

private EditText pathText;
private ProgressBar progressBar;
private TextView resultView;
private Handler handler = new Handler(){


public void handleMessage(Message msg) {

        if(!Thread.currentThread().isInterrupted())
        {
                switch (msg.what)
                {
                    case 1:  // 获取当前文件下载的进度
                                int size = msg.getData().getInt("size");
                                progressBar.setProgress(size);
                                int result = (int)(((float)size/(float)progressBar.getMax()) * 100);
                                resultView.setText(result+ "%");
                                if(progressBar.getMax() == size)
                                {
                                    Toast.makeText(MulThreadDownloader.this, "文件下载完成", 1).show();
                                }
                                break;

case -1:    String error = msg.getData().getString("error");
             Toast.makeText(MulThreadDownloader.this, error, 1).show();
             break;
    }
}

super.handleMessage(msg);

}
};


public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        pathText = (EditText)this.findViewById(R.id.path);
        progressBar = (ProgressBar)this.findViewById(R.id.downloadbar);
        resultView = (TextView)this.findViewById(R.id.resultView);
        Button button = (Button)this.findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {

        public void onClick(View v)
        {
                String path = pathText.getText().toString();
                if(Environment.getExternalStorageState().equals(Environment.MEDIA_MOUNTED))
                {
```

```
//下载文件需要很长的时间，主线程是不能够长时间被阻塞，如果主线程被长时间阻塞， 那么 Android 被回收应用
                        download(path, Environment.getExternalStorageDirectory());
                }
                else{
                        Toast.makeText(MulThreadDownloader.this, R.string.sdcarderror, 1).show();
                    }

        }
        });

}

/**
* 下载文件
* @param path 下载路径
* @param saveDir 文件保存目录
*/

//对于 Android 的 UI 控件，只能由主线程负责显示界面的更新，其他线程不能直接更新 UI 控件的显示

public void download(final String path, final File saveDir){

new Thread(new Runnable() {

public void run()
{
        FileDownloader downer = new FileDownloader(MulThreadDownloader.this, path, saveDir, 3);

        progressBar.setMax(downer.getFileSize());//设置进度条的最大刻度
        try {
                downer.download(new DownloadProgressListener()
                {
                        public void onDownloadSize(int size)
                        {
                                Message msg = new Message();
                                msg.what = 1;
                                msg.getData().putInt("size", size);
                                handler.sendMessage(msg);//发送消息
        }});
        } catch (Exception e) {
                Message msg = new Message();
                msg.what = -1;
                msg.getData().putString("error", "下载失败");
                handler.sendMessage(msg);
        }
    }
}).start();
}
}
```

1.2.6 FileDownload

```
package com.changcheng.net.download;

import java.io.File;
import java.io.RandomAccessFile;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.LinkedHashMap;
import java.util.Map;
import java.util.UUID;
import java.util.concurrent.ConcurrentHashMap;
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import com.changcheng.download.service.FileService;
import android.content.Context;
import android.util.Log;

/**
* 文件下载器
* @author lihuoming@sohu.com
*
*/

public class FileDownloader {

private Context context;
private FileService fileService;
private static final String TAG = "FileDownloader";

/* 已下载文件大小 */
private int downloadSize = 0;

/* 原始文件大小 */
private int fileSize = 0;

/* 线程数 */
private DownloadThread[] threads;

/* 下载路径 */
private URL url;

/* 本地保存文件 */
private File saveFile;

/* 下载记录文件 */
private File logFile;

/* 缓存各线程最后下载的位置*/
private Map<Integer, Integer> data = new ConcurrentHashMap<Integer, Integer>();

/* 每条线程下载的大小 */
private int block;
```

```
private String downloadUrl;//下载路径

/**
* 获取线程数
*/

public int getThreadSize() {
      return threads.length;
}

/**
* 获取文件大小
* @return
*/

public int getFileSize() {
     return fileSize;
}

/**
* 累计已下载大小
* @param size
*/

protected synchronized void append(int size) {
      downloadSize += size;
}

/**
* 更新指定线程最后下载的位置
* @param threadId 线程 id
* @param pos 最后下载的位置
*/
protected void update(int threadId, int pos) {
      this.data.put(threadId, pos);
}

/**
* 保存记录文件
*/

protected synchronized void saveLogFile() {
      this.fileService.update(this.downloadUrl, this.data);
}

/**
* 构建文件下载器
* @param downloadUrl 下载路径
* @param fileSaveDir 文件保存目录
* @param threadNum 下载线程数
*/

public FileDownloader(Context context, String downloadUrl, File fileSaveDir, int threadNum) {
```

```
try
{
        this.context = context;
        this.downloadUrl = downloadUrl;
        fileService = new FileService(context);
        this.url = new URL(downloadUrl);
        if(!fileSaveDir.exists()) fileSaveDir.mkdirs();
        this.threads = new DownloadThread[threadNum];
        HttpURLConnection conn = (HttpURLConnection) url.openConnection();
        conn.setConnectTimeout(6*1000);
        conn.setRequestMethod("GET");
        conn.setRequestProperty("Accept", "image/gif, image/jpeg, image/pjpeg, image/pjpeg, application/x-
        shockwave-flash, application/xaml+xml, application/vnd.ms-xpsdocument, application/x-ms-xbap,
        application/x-ms-application, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword,
        */*");
        conn.setRequestProperty("Accept-Language", "zh-CN");
        conn.setRequestProperty("Referer", downloadUrl);
        conn.setRequestProperty("Charset", "UTF-8");
        conn.setRequestProperty("User-Agent", "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.2;
        Trident/4.0; .NET CLR 1.1.4322; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30; .NET CLR
        3.0.4506.2152; .NET CLR 3.5.30729)");
        conn.setRequestProperty("Connection", "Keep-Alive");
        conn.connect();
        printResponseHeader(conn);
        if (conn.getResponseCode()==200)
         {
             this.fileSize = conn.getContentLength();//根据响应获取文件大小
             if (this.fileSize <= 0) throw new RuntimeException("无法获知文件大小 ");
             String filename = getFileName(conn);
             this.saveFile = new File(fileSaveDir, filename);/* 保存文件 */
             Map<Integer, Integer> logdata = fileService.getData(downloadUrl);
             if(logdata.size()>0)
             {
                 data.putAll(logdata);
            }
            this.block = this.fileSize / this.threads.length + 1;
            if(this.data.size()==this.threads.length)
            {
                 for (int i = 0; i < this.threads.length; i++)
                 {
                     this.downloadSize += this.data.get(i+1)-(this.block * i);
                 }
            print("已经下载的长度"+ this.downloadSize);
            }
            }else{
            throw new RuntimeException("服务器响应错误 ");
            }
} catch (Exception e) {
            print(e.toString());
            throw new RuntimeException("连接不到下载路径 ");
}
}

/**
* 获取文件名
```

```
*/

private String getFileName(HttpURLConnection conn) {

String filename = this.url.toString().substring(this.url.toString().lastIndexOf('/') + 1);
if(filename==null || "".equals(filename.trim()))
{//如果获取不到文件名称
        for (int i = 0;; i++)
        {
            String mine = conn.getHeaderField(i);
            if (mine == null) break;
            if("content-disposition".equals(conn.getHeaderFieldKey(i).toLowerCase()))
            {
                    Matcher m = Pattern.compile(".*filename=(.*)").matcher(mine.toLowerCase());
                    if(m.find()) return m.group(1);
             }
       }
       filename = UUID.randomUUID()+ ".tmp";//默认取一个文件名
}
return filename;
}

/**
* 开始下载文件
* @param listener 监听下载数量的变化,如果不需要了解实时下载的数量,可以设置为 null
* @return 已下载文件大小
* @throws Exception
*/

public int download(DownloadProgressListener listener) throws Exception{

try {
if(this.data.size() != this.threads.length)
{
    this.data.clear();
    for (int i = 0; i < this.threads.length; i++)
    {
            this.data.put(i+1, this.block * i);
    }
}
for (int i = 0; i < this.threads.length; i++)
{
    int downLength = this.data.get(i+1) - (this.block * i);
    if(downLength < this.block && this.data.get(i+1)<this.fileSize)
    { //该线程未完成下载时,继续下载
        RandomAccessFile randOut = new RandomAccessFile(this.saveFile, "rw");
        if(this.fileSize>0) randOut.setLength(this.fileSize);
        randOut.seek(this.data.get(i+1));
        this.threads[i] = new DownloadThread(this, this.url, randOut, this.block, this.data.get(i+1), i+1);
        this.threads[i].setPriority(7);
        this.threads[i].start();
    }else{
            this.threads[i] = null;
    }
}
```

```
this.fileService.save(this.downloadUrl, this.data);
boolean notFinish = true;//下载未完成
while (notFinish) {// 循环判断是否下载完毕
Thread.sleep(900);
notFinish = false;//假定下载完成
for (int i = 0; i < this.threads.length; i++)
{
    if (this.threads[i] != null && !this.threads[i].isFinish())
     {
        notFinish = true;//下载没有完成
        if(this.threads[i].getDownLength() == -1)
        {//如果下载失败,再重新下载
            RandomAccessFile randOut = new RandomAccessFile(this.saveFile, "rw");
            randOut.seek(this.data.get(i+1));
            this.threads[i] = new DownloadThread(this, this.url, randOut, this.block, this.data.get(i+1), i+1);
            this.threads[i].setPriority(7);
            this.threads[i].start();
        }
    }
}
if(listener!=null) listener.onDownloadSize(this.downloadSize);
}
fileService.delete(this.downloadUrl);
} catch (Exception e) {
    print(e.toString());
    throw new Exception("下载失败");
}
return this.downloadSize;
}

/**
* 获取 Http 响应头字段
* @param http
* @return
*/

public static Map<String, String> getHttpResponseHeader(HttpURLConnection http) {

Map<String, String> header = new LinkedHashMap<String, String>();
for (int i = 0;; i++)
{
    String mine = http.getHeaderField(i);
    if (mine == null) break;
    header.put(http.getHeaderFieldKey(i), mine);
}
return header;
}

/**
* 打印 Http 头字段
* @param http
*/

public static void printResponseHeader(HttpURLConnection http){
```

```
Map<String, String> header = getHttpResponseHeader(http);
for(Map.Entry<String, String> entry : header.entrySet())
{
    String key = entry.getKey()!=null ? entry.getKey()+ ":" : "";
    print(key+ entry.getValue());
}
}

private static void print(String msg)
{
Log.i(TAG, msg);
}
}
```

1.2.7 DownloadProgressListener

```
package com.changcheng.net.download;

public interface DownloadProgressListener {
        public void onDownloadSize(int size);
}
```

1.2. 8 FileService


```
package com.changcheng.download.service;

import java.util.HashMap;
import java.util.Map;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

/**
* 业务 bean
*
*/

public class FileService {

private DBOpenHelper openHelper;
public FileService(Context context) {
        openHelper = new DBOpenHelper(context);
}

/**
* 获取线程最后下载位置
* @param path
* @return
*/

public Map<Integer, Integer> getData(String path){

SQLiteDatabase db = openHelper.getReadableDatabase();
Cursor cursor = db.rawQuery("select threadid, position from filedown where downpath=?", new String[]{path});
Map<Integer, Integer> data = new HashMap<Integer, Integer>();
```

```
while(cursor.moveToNext())
{
        data.put(cursor.getInt(0), cursor.getInt(1));
}

cursor.close();
db.close();
return data;
}

/**
* 保存下载线程初始位置
* @param path
* @param map
*/

public void save(String path, Map<Integer, Integer> map){//int threadid, int position

SQLiteDatabase db = openHelper.getWritableDatabase();
db.beginTransaction();
try{
    for(Map.Entry<Integer, Integer> entry : map.entrySet())
    {
        db.execSQL("insert into filedown(downpath, threadid, position) values(?,?,?)",
        new Object[]{path, entry.getKey(), entry.getValue()});
    }
    db.setTransactionSuccessful();
    }finally{
        db.endTransaction();
    }
db.close();
}

/**
* 实时更新线程的最后下载位置
* @param path
* @param map
*/

public void update(String path, Map<Integer, Integer> map){

SQLiteDatabase db = openHelper.getWritableDatabase();
db.beginTransaction();
try{
        for(Map.Entry<Integer, Integer> entry : map.entrySet())
        {
            db.execSQL("update filedown set position=? where downpath=? and threadid=?",
            new Object[]{entry.getValue(), path, entry.getKey()});
        }
        db.setTransactionSuccessful();
    }finally
    {
        db.endTransaction();
     }
db.close();
```

```
}

/**
* 当文件下载完成后，清掉该文件对应的下载记录
* @param path
*/

public void delete(String path){

SQLiteDatabase db = openHelper.getWritableDatabase();
db.execSQL("delete from filedown where downpath=?", new Object[]{path});
db.close();
}
}
```

1.1.9 DownloadThread

```
package com.changcheng.net.download;

import java.io.InputStream;
import java.io.RandomAccessFile;
import java.net.HttpURLConnection;
import java.net.URL;
import android.util.Log;

public class DownloadThread extends Thread {

private static final String TAG = "DownloadThread";
private RandomAccessFile saveFile;
private URL downUrl;
private int block;

/* 下载开始位置 */
private int threadId = -1;
private int startPos;
private int downLength;
private boolean finish = false;
private FileDownloader downloader;

public DownloadThread(FileDownloader downloader, URL downUrl, RandomAccessFile saveFile, int block, int
startPos, int threadId) {

this.downUrl = downUrl;
this.saveFile = saveFile;
this.block = block;
this.startPos = startPos;
this.downloader = downloader;
this.threadId = threadId;
this.downLength = startPos - (block * (threadId - 1));
}


public void run() {
    if(downLength < block)
    {//未下载完成
```

```
        try
        {
            HttpURLConnection http = (HttpURLConnection) downUrl.openConnection();
            http.setRequestMethod("GET");
            http.setRequestProperty("Accept", "image/gif, image/jpeg, image/pjpeg, image/pjpeg, application/x-
            shockwave-flash, application/xaml+xml, application/vnd.ms-xpsdocument, application/x-ms-xbap,
            application/x-ms-application, application/vnd.ms-excel, application/vnd.ms-powerpoint, application/msword,
            */*");
            http.setRequestProperty("Accept-Language", "zh-CN");
            http.setRequestProperty("Referer", downUrl.toString());
            http.setRequestProperty("Charset", "UTF-8");
            http.setRequestProperty("Range", "bytes=" + this.startPos + "-");
            http.setRequestProperty("User-Agent", "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.2;
            Trident/4.0; .NET CLR 1.1.4322; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30; .NET CLR
            3.0.4506.2152; .NET CLR 3.5.30729)");
            http.setRequestProperty("Connection", "Keep-Alive");
            InputStream inStream = http.getInputStream();
            int max = block>1024 ? 1024 : (block>10 ? 10 : 1);
            byte[] buffer = new byte[max];
            int offset = 0;
            print("线程 " + this.threadId + "从位置"+ this.startPos+ "开始下载 ");
            while (downLength < block && (offset = inStream.read(buffer, 0, max)) != -1)
             {
                saveFile.write(buffer, 0, offset);
                downLength += offset;
                downloader.update(this.threadId, block * (threadId - 1) + downLength);
                downloader.saveLogFile();
                downloader.append(offset);
                int spare = block-downLength;//求剩下的字节数
                if(spare < max) max = (int) spare;
            }
            saveFile.close();
            inStream.close();
            print("线程 " + this.threadId + "完成下载 ");
            this.finish = true;
            this.interrupt();
            } catch (Exception e) {
                this.downLength = -1;
                print("线程"+ this.threadId+ ":"+ e);
                }
            }
        }

    private static void print(String msg){
    Log.i(TAG, msg);
}

/**
* 下载是否完成
* @return
*/

public boolean isFinish() {
            return finish;
}
```

```
/**
* 已经下载的内容大小
* @return 如果返回值为-1,代表下载失败
*/

public long getDownLength() {
        return downLength;
}
}
```

1.1.10 DBOpenHelper

```
package com.changcheng.download.service;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DBOpenHelper extends SQLiteOpenHelper {

private static final String DBNAME = "download.db";
private static final int VERSION = 2;

public DBOpenHelper(Context context) {
        super(context, DBNAME, null, VERSION);
}

public void onCreate(SQLiteDatabase db) {

        db.execSQL("CREATE TABLE IF NOT EXISTS filedown (id integer primary key autoincrement, downpath
        varchar(100), threadid INTEGER, position INTEGER)");

}
public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS filedown");
    onCreate(db);
}
}
```

1.3 Android 网路处理之蓝牙通信

Package name: **android.bluetooth.***，主要相关类介绍如下：

- BluetoothAdapter: 本地蓝牙设备的适配类，所有的蓝牙操作都要通过该类完成；
- BluetoothDevice: 蓝牙设备类，代表了蓝牙通讯过程中的远端设备；
- BluetoothSocket: 蓝牙通讯套接字，代表了与远端设备的连接点，使用 socket 本地程序可以通过 inputstream 和 outputstream 与远端程序进行通讯；
- BluetoothServerSocket: 服务器通讯套接字，与 TCP ServerSocket 类似；
- BluetoothClass: 用于描述远端设备的类型，特点等信息，通过 getBluetoothClass()方法获取代表远端设备属性的 BluetoothClass 对象。

1.3.1 使用蓝牙必须获取的权限：

一定要在 AndroidManifest.xml 配置文件中配置上一下两个权限否则蓝牙设备是不可用的

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
```

1.3.2 建立蓝牙连接：

通过 BluetoothAdapter.getDefaultAdapter()方法获取 BluetoothAdapter 对象。
判断当前蓝牙是否启动，如果没有启动提示用户手动启动：

```
if (!mBluetoothAdapter.isEnabled()) {
    Intent enableBtIntent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(enableBtIntent, REQUEST_ENABLE_BT);
}
```

在 Activity 的 onActivityResult()方法中，对用户的设定结果进行处理。

1.3.3 搜寻远端蓝牙设备

1、首先获取已配对的远端设备：

Set<BluetoothDevice> pairedDevices = mBluetoothAdapter.getBondedDevices();

2、然后通过 BluetoothAdapter.startDiscovery()方法启动蓝牙设备的搜寻。

这是个异步方法，调用的时候立刻就会返回。为了获得搜寻的结果，必须在用户自己的 Activity 中注册一个 BroadcastReceiver，代码如下：

```
// Create a BroadcastReceiver for ACTION_FOUND
private final BroadcastReceiver mReceiver = new BroadcastReceiver() {
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        // When discovery finds a device
        if (BluetoothDevice.ACTION_FOUND.equals(action)) {
            // Get the BluetoothDevice object from the Intent
            BluetoothDevice device = intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
            // Add the name and address to an array adapter to show in a ListView
            mArrayAdapter.add(device.getName() + "\n" + device.getAddress());
        }
    }
};
// Register the BroadcastReceiver
IntentFilter filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
registerReceiver(mReceiver, filter); // Don't forget to unregister during onDestroy
```

3、设置本地设备可以被发现

只有将本地设备设置为可被发现，远端的蓝牙设备才能够找到并和本地设备建立连接。通过下面的代码发送 Intent 对象，让用户手动启动可发现设置。

```
Intent discoverableIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
discoverableIntent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, 300);
startActivity(discoverableIntent);
```

在本地设备可发现的过程中，可以通过注册 BroadcastReceiver 监听可发现状态的改变。

1.3.4 实现编码片段：

```
服务端
UUID uuid = uuid.fromString("27648B4D-D854-5674-FA6OE4F535E44AF7");
BluetoothAdapter adapter = BluetoothAdapter.getDefaultAdapter();
BluetoothServerSocket serverSocket = adapter.listenUsingRfcommWithServiceRecord("MyBluetoothApp", uuid);
BluetoothSocket socket = serverSocket.accept(); // blocks until a connection is accepted
serverSocket.close(); // close the listening socket

客户端:
UUID uuid = uuid.fromString("27648B4D-D854-5674-FA6OE4F535E44AF7"); // UUID of server socket
BluetoothAdapter adapter = BluetoothAdapter.getDefaultAdapter();
BluetoothDevice device = adapter.getRemoteDevice("OO:11:22:33:44:55"); // BT MAC address of server

// 发现连接后会获取服务端socket 套接字
BluetoothSocket socket = device.createRfcommSocketToServiceRecord(uuid);
// 取消主动发现设备
adapter.cancelDiscovery();
// 与服务端建立连接
adapter.connect();

双方连接上后，就开始读写了
InputStream in = socket.getInputStream();
OutputStream out = socket.getOutputStream();
out.write(…);
in.read(…);
…
in.close();
out.close();
socket.close();
```

通信方式跟 socket 通信原理是一样的，只不过协议不一样，但是协议对于用户都是封装的，所以不必担心。

【Android 网路处理实例教程】

2.1 Android Socket 网络通信

2.1.1 服务器程序：

```
package com;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

/**
 * com Server
 *
 * @author Aina.huang E-mail: 674023920@qq.com
 * @version 创建时间：2010 Jul 14, 2010 10:45:35 AM 类说明
 */
public class Main {

    private static final int PORT = 9999;// 端口监听
    private List<Socket> mList = new ArrayList<Socket>();// 存放客户端 socket
    private ServerSocket server = null;
    private ExecutorService mExecutorService = null;// 线程池

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        new Main();
    }

    public Main() {
        try {
            server = new ServerSocket(PORT);
            mExecutorService = Executors.newCachedThreadPool();// 创建一个线程池
            System.out.println("Server Start...");
            Socket client = null;
            while (true) {
```

```
            client = server.accept();
            mList.add(client);
            mExecutorService.execute(new Service(client));// 开启一个客户端线程.
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

public class Service implements Runnable {

    private Socket socket;
    private BufferedReader in = null;
    private String msg = "";

    public Service(Socket socket) {
        this.socket = socket;
        try {
            in = new BufferedReader(new InputStreamReader(socket
                    .getInputStream()));
            msg = "user:" + this.socket.getInetAddress() + " come total:"
                    + mList.size();
            this.sendmsg();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void run() {
        // TODO Auto-generated method stub
        try {
            while (true) {
                if ((msg = in.readLine()) != null) {
                    if (msg.equals("exit")) {
                        System.out.println("sssssssssss");
                        mList.remove(socket);
                        in.close();
                        msg = "user:" + socket.getInetAddress()
                                + " exit total:" + mList.size();
                        socket.close();
                        this.sendmsg();
                        break;
                    } else {
                        msg = socket.getInetAddress() + " : " + msg;
                        this.sendmsg();
                    }
                }

            }
        } catch (Exception ex) {
            System.out.println("server 读取数据异常");
            ex.printStackTrace();
        }
    }
```

```
        /**
         * 发送消息给所有客户端
         */
        public void sendmsg() {
            System.out.println(msg);
            int num = mList.size();
            for (int i = 0; i < num; i++) {
                Socket mSocket = mList.get(i);
                PrintWriter pout = null;
                try {
                    pout = new PrintWriter(new BufferedWriter(
                            new OutputStreamWriter(mSocket.getOutputStream())),
                            true);
                    pout.println(msg);
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }
}


package com;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;

/**
 * com Server
 *
 * @author Aina.huang E-mail: 674023920@qq.com
 * @version 创建时间：2010 Jul 14, 2010 10:45:35 AM 类说明
 */
public class Main {

    private static final int PORT = 9999;// 端口监听
    private List<Socket> mList = new ArrayList<Socket>();// 存放客户端 socket
    private ServerSocket server = null;
    private ExecutorService mExecutorService = null;// 线程池

    /**
     * @param args
     */
    public static void main(String[] args) {
```

```
    // TODO Auto-generated method stub
    new Main();
}

public Main() {
    try {
        server = new ServerSocket(PORT);
        mExecutorService = Executors.newCachedThreadPool();// 创建一个线程池
        System.out.println("Server Start...");
        Socket client = null;
        while (true) {
            client = server.accept();
            mList.add(client);
            mExecutorService.execute(new Service(client));// 开启一个客户端线程.
        }
    } catch (Exception ex) {
        ex.printStackTrace();
    }
}

public class Service implements Runnable {

    private Socket socket;
    private BufferedReader in = null;
    private String msg = "";

    public Service(Socket socket) {
        this.socket = socket;
        try {
            in = new BufferedReader(new InputStreamReader(socket
                    .getInputStream()));
            msg = "user:" + this.socket.getInetAddress() + " come total:"
                    + mList.size();
            this.sendmsg();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void run() {
        // TODO Auto-generated method stub
        try {
            while (true) {
                if ((msg = in.readLine()) != null) {
                    if (msg.equals("exit")) {
                        System.out.println("sssssssssss");
                        mList.remove(socket);
                        in.close();
                        msg = "user:" + socket.getInetAddress()
                                + " exit total:" + mList.size();
                        socket.close();
                        this.sendmsg();
                        break;
                    } else {
                        msg = socket.getInetAddress() + " : " + msg;
```

```
                    this.sendmsg();
                }
            }

        }
    } catch (Exception ex) {
        System.out.println("server 读取数据异常");
        ex.printStackTrace();
    }
}

/**
 * 发送消息给所有客户端
 */
public void sendmsg() {
    System.out.println(msg);
    int num = mList.size();
    for (int i = 0; i < num; i++) {
        Socket mSocket = mList.get(i);
        PrintWriter pout = null;
        try {
            pout = new PrintWriter(new BufferedWriter(
                    new OutputStreamWriter(mSocket.getOutputStream())),
                    true);
            pout.println(msg);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
}
}
```

2.1.2 客户端程序：

```
package com.Aina.Android;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.Socket;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
```

```
import android.widget.TextView;

public class Test extends Activity implements Runnable {
    /** Called when the activity is first created. */
    private TextView tv_msg = null;
    private EditText ed_msg = null;
    private Button btn_send = null;
    private Button btn_login = null;
    private static final String HOST = "192.168.0.132";
    private static final int PORT = 9999;
    private Socket socket = null;
    private BufferedReader in = null;
    private PrintWriter out = null;
    private String content = "";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        tv_msg = (TextView) this.findViewById(R.id.TextView);
        ed_msg = (EditText) this.findViewById(R.id.EditText01);
        btn_login = (Button) this.findViewById(R.id.Button01);
        btn_send = (Button) this.findViewById(R.id.Button02);
        try {
            socket = new Socket(HOST, PORT);
            in = new BufferedReader(new InputStreamReader(socket
                    .getInputStream()));
            out = new PrintWriter(new BufferedWriter(
                    new OutputStreamWriter(socket.getOutputStream())),
                    true);
        } catch (Exception ex) {
            ex.printStackTrace();
            ShowDialog("登陆异常:" + ex.getMessage());
        }
        btn_send.setOnClickListener(new Button.OnClickListener() {

            public void onClick(View v) {
                // TODO Auto-generated method stub
                String msg = ed_msg.getText().toString();
                if (socket.isConnected()) {
                    if (!socket.isOutputShutdown()) {
                        out.println(msg);
                    }
                }
            }

        });
        new Thread(this).start();
    }

    public void ShowDialog(String msg) {
        new AlertDialog.Builder(this).setTitle("提示").setMessage(msg)
                .setPositiveButton("OK", new DialogInterface.OnClickListener() {

                    public void onClick(DialogInterface dialog, int which) {
```

```
                // TODO Auto-generated method stub

            }

        }).show();
    }

    public void run() {
        try {
            while (true) {
                if(socket.isConnected()){
                    if(!socket.isInputShutdown()){
                        if ((content = in.readLine()) != null) {
                            Log.i("TAG", "++ "+content);
                            content += "\n";
                            mHandler.sendMessage(mHandler.obtainMessage());
                        }else{

                        }
                    }
                }

            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    public Handler mHandler = new Handler() {
        public void handleMessage(Message msg) {
            super.handleMessage(msg);
            Log.i("TAG", "-- "+msg);
            tv_msg.setText(tv_msg.getText().toString() + content);
        }
    };
}

package com.Aina.Android;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.PrintWriter;
import java.net.Socket;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
```

```
import android.widget.TextView;

public class Test extends Activity implements Runnable {
    /** Called when the activity is first created. */
    private TextView tv_msg = null;
    private EditText ed_msg = null;
    private Button btn_send = null;
    private Button btn_login = null;
    private static final String HOST = "192.168.0.132";
    private static final int PORT = 9999;
    private Socket socket = null;
    private BufferedReader in = null;
    private PrintWriter out = null;
    private String content = "";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        tv_msg = (TextView) this.findViewById(R.id.TextView);
        ed_msg = (EditText) this.findViewById(R.id.EditText01);
        btn_login = (Button) this.findViewById(R.id.Button01);
        btn_send = (Button) this.findViewById(R.id.Button02);
        try {
            socket = new Socket(HOST, PORT);
            in = new BufferedReader(new InputStreamReader(socket
                    .getInputStream()));
            out = new PrintWriter(new BufferedWriter(
                    new OutputStreamWriter(socket.getOutputStream())),
                    true);
        } catch (Exception ex) {
            ex.printStackTrace();
            ShowDialog("登陆异常:" + ex.getMessage());
        }
        btn_send.setOnClickListener(new Button.OnClickListener() {

            public void onClick(View v) {
                // TODO Auto-generated method stub
                String msg = ed_msg.getText().toString();
                if (socket.isConnected()) {
                    if (!socket.isOutputShutdown()) {
                        out.println(msg);
                    }
                }
            }

        });
        new Thread(this).start();
    }

    public void ShowDialog(String msg) {
        new AlertDialog.Builder(this).setTitle("提示").setMessage(msg)
                .setPositiveButton("OK", new DialogInterface.OnClickListener() {

                    public void onClick(DialogInterface dialog, int which) {
```

```
                // TODO Auto-generated method stub

            }

        }).show();
    }

    public void run() {
        try {
            while (true) {
                if(socket.isConnected()){
                    if(!socket.isInputShutdown()){
                        if ((content = in.readLine()) != null) {
                            Log.i("TAG", "++ "+content);
                            content += "\n";
                            mHandler.sendMessage(mHandler.obtainMessage());
                        }else{

                        }
                    }
                }

            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }

    public Handler mHandler = new Handler() {
        public void handleMessage(Message msg) {
            super.handleMessage(msg);
            Log.i("TAG", "-- "+msg);
            tv_msg.setText(tv_msg.getText().toString() + content);
        }
    };
}


<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView android:id="@+id/TextView" android:singleLine="false"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />
    <EditText android:hint="content" android:id="@+id/EditText01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
    </EditText>
    <Button android:text="login" android:id="@+id/Button01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
    </Button>
    <Button android:text="send" android:id="@+id/Button02"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
```

```
    </Button>
</LinearLayout>

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TextView android:id="@+id/TextView" android:singleLine="false"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content" />
    <EditText android:hint="content" android:id="@+id/EditText01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
    </EditText>
    <Button android:text="login" android:id="@+id/Button01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
    </Button>
    <Button android:text="send" android:id="@+id/Button02"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
    </Button>
</LinearLayout>
```

2.2 使用 Google Weather API 制作的天气预报应用

大家在使用 android 手机的时候，肯定都是用过天气预报的应用，market 上面已经有了不少很成熟的产品．当然，作为开发者而言，一定会对这种应用的开发很感兴趣，我们能不能自己来写一款类似的应用呢？答案当然是可以的，而且非常的简单。下面就是我们最终完成的截图：

首先，要开发一款天气预报应用，一定要有一个 web 服务端来提供数据，这个数据源我们自己肯定是没办法弄的，所以就需要一个第三方机构为我们提供天气数据。

这种机构其实有很多，不过大多数都是收费的，当然这些收费的数据源提供的数据会更加丰富详细．如果不想花钱去购买这些收费的数据服务，我们还有另一种替代方案－就是使用免费的天气数据，这篇文章了为大家介绍一个 Google 提供的天气 API，通过浏览器访问下面的链接：

http://www.google.com/ig/api?hl=zh-cn&weather=Beijing

如果你的浏览器可以直接显示 XML 文档，那么就会得到类似下面这样的数据：

```
<weather module_id="0" tab_id="0" mobile_row="0" mobile_zipped="1" row="0" section="0">
. . .
<forecast_conditions>
  <day_of_week data="周四" />
  <low data="4" />
  <high data="19" />
  <icon data="/ig/images/weather/mostly_sunny.gif" />
```

```
  <condition data="以晴为主" />
</forecast_conditions>
 <forecast_conditions>
  <day_of_week data="周五" />
  <low data="3" />
  <high data="19" />
  <icon data="/ig/images/weather/sunny.gif" />
  <condition data="晴" />
</forecast_conditions>
. . .
</weather>
```

当然，这里只给大家列出一个片断，完整的数据大家可以自己用浏览器来查看．上面这段数据给我们提供了气温的数字和文字描述，还给我们提供了一幅表示当天天气状况的图片。对于我们这个简单的天气应用，这些数据已经足够了。

有了数据之后，我们就开始开发吧，怎么建项目就不用我说了吧，呵呵。虽然这个应用很简单，但我们还需要把结构稍微整理一下，我们需要用一个实体类来表示天气数据:

```
public class Weather {

   private String day;
   private String lowTemp;
   private String highTemp;
   private String imageUrl;
   private String condition;
}
```


我们通过 XML 文档提供的数据格式来定义我们实体类，这里面包含了，当天是周几，最低气温，最高气温，天气图片的地址，和天气状况的文字描述．为了节省篇幅　getter 和 setter 方法就省略了，现在我们已经把我们需要的数据封装好了。

接下来我们需要解析 XML 数据，将服务器返回给我们的 XML 格式的数据，转换成程序比较好操作的对象，我们可以使用 SAX 来解析 XML 文档，关于 SAX 的更多细节，不是本篇文章要讨论的内容，不过为了让大家好理解，还是简单的叙述一下。

SAX 其实是解析 XML 文档的一种方法，一般处理 XML 数据有两种方法，一种是将数据先解析为一种树形结构，然后我们再来在这个结构上访问数据，这种方法是我们通常会直接想到的，而 SAX 则采用了另外一种方法，这种方法简单来说就是，当解析器遍历 XML 文档的时候，会给提供我们一些回调函数，比如遇到起始标签，遇到结束标签，或是遇到标签中的文字等等，这是一种基于事件的解析方式，所以我们需要一个类来处理这些事件，并且将需要的数据保存下来，就产生了下面这段代码：

```
public class XmlHandler extends DefaultHandler {

   private List<Weather> weatherList;
   private boolean inForcast;
   private Weather currentWeather;

   public List<Weather> getWeatherList() {
      return weatherList;
   }
```

```
    public void setWeatherList(List<Weather> weatherList) {
        this.weatherList = weatherList;
    }

    public XmlHandler() {

        weatherList = new ArrayList<Weather>();
        inForcast = false;
    }

    @Override
    public void startElement(String uri, String localName, String qName,
            Attributes attributes) throws SAXException {

        String tagName = localName.length() != 0 ? localName : qName;
        tagName = tagName.toLowerCase();
        if(tagName.equals("forecast_conditions"))
        {
            inForcast = true;
            currentWeather = new Weather();
        }
        if(inForcast)
        {
             if(tagName.equals("day_of_week")) {
                currentWeather.setDay(attributes.getValue("data"));
            }else if(tagName.equals("low")) {
                currentWeather.setLowTemp(attributes.getValue("data"));
            }else if(tagName.equals("high")) {
                currentWeather.setHighTemp(attributes.getValue("data"));
            }else if(tagName.equals("icon")) {
                currentWeather.setImageUrl(attributes.getValue("data"));
            }else if(tagName.equals("condition")) {
                currentWeather.setCondition(attributes.getValue("data"));
            }
        }

    }

    @Override
    public void endElement(String uri, String localName, String qName)
            throws SAXException {
        String tagName = localName.length() != 0 ? localName : qName;
        tagName = tagName.toLowerCase();
        if(tagName.equals("forecast_conditions"))
        {
            inForcast = false;
            weatherList.add(currentWeather);
        }
    }
}
```

startElement 方法，代表遇到起始标签，我们在这里得到了标签名，如果遇到 forecast_conditions 标签，我们就会标记一下，并且创建一个天气实体对象，下面的 if 语句中，判断了是否在 forecast_conditions 标签内，如果在的话，就把它里面相应的属性提取出来。endElement 方法，代表遇到结束标签，我们的代码里，

如果遇到 forecast_conditions 标签，那么就证明当前这条天气数据已经解析完成，所以我们将该实体对象保存到 List 列表中，以便以后使用。

现在终于处理完数据了，其实这个程序本身并不是很复杂，大半的代码都用在了解析数据上面。下面开始进入我们的主程序，首先来看看我们的布局文件：

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
>
        <LinearLayout  android:layout_width="fill_parent"
                        android:layout_height="wrap_content">
                <EditText
                        android:id="@+id/txCity"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:layout_weight="9"
                />
                <Button
                        android:id="@+id/btnSearch"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:layout_weight="2"
                        android:text="查询"
                />
        </LinearLayout>
        <TableLayout
                android:id="@+id/table"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:stretchColumns="1,2,3,4"
           >
</TableLayout>
</LinearLayout>
```

下面就到了最后一个部分，也是程序中主要的部分，我们的 Activity 代码，首先，我们需要定义一个查询天气的方法：

```
   private void searchWeather(String city) {

      SAXParserFactory spf = SAXParserFactory.newInstance();
      try {
                        //此处为 SAX 解析部分
         SAXParser sp = spf.newSAXParser();
         XMLReader reader = sp.getXMLReader();
         XmlHandler  handler = new XmlHandler();
         reader.setContentHandler(handler);
         URL url = new URL("http://www.google.com/ig/api?hl=zh-cn&weather=" + URLEncoder.encode(city));
         InputStream is = url.openStream();
         InputStreamReader isr = new InputStreamReader(is,"GBK");
         InputSource source = new InputSource(isr);
         reader.parse(source);
```

```
        List<Weather> weatherList = handler.getWeatherList();
        TableLayout table = (TableLayout)findViewById(R.id.table);
        table.removeAllViews();
        for(Weather weather : weatherList)
          {
                        TableRow row = new TableRow(this);
                         row.setLayoutParams(new LayoutParams(LayoutParams.FILL_PARENT,
                         LayoutParams.WRAP_CONTENT));
                      row.setGravity(Gravity.CENTER_VERTICAL);
                       ImageView img = new ImageView(this);
                       img.setImageDrawable(loadImage(weather.getImageUrl()));
                       img.setMinimumHeight(80);
                       row.addView(img);
                       TextView day = new TextView(this);
                       day.setText(weather.getDay());
                       day.setGravity(Gravity.CENTER_HORIZONTAL);
                       row.addView(day);
                       TextView temp = new TextView(this);
                       temp.setText(weather.getLowTemp() + "℃ - " + weather.getHighTemp() + "℃");
                       temp.setGravity(Gravity.CENTER_HORIZONTAL);
                       row.addView(temp);
                       TextView condition = new TextView(this);
                       condition.setText(weather.getCondition());
                       condition.setGravity(Gravity.CENTER_HORIZONTAL);
                       row.addView(condition);
                       table.addView(row);
        }
    } catch (Exception e) {

        new AlertDialog.Builder(this)
        .setTitle("解析错误")
        .setMessage("获取天气数据失败，请稍候再试。")
        .setNegativeButton("确定", null)
        .show();
    }
}
```

大家看看代码应该就差不多都明白了，这个方法开始的部分，我们使用 SAX 相关的 API 来处理 XML 数据，有几处地方需要说明一下：

```
InputStreamReader isr = new InputStreamReader(is,"GBK");
```

由于 API 返回给我们的中文是国标编码的，而 SAX 默认会以 UTF-8 来处理得到的数据，所以我们要在输入流中指定一下编码格式。接下来调用 reader.parse(source);方法来解析我们的输入源，这里的一连串 SAX 方法调用，表达的目的应该很清楚了，相信以大家的水平，即使以前没有用过，也能很容易看明白，当然，如果是在看不懂，可以先参考一下我的那篇帖子，补充一下基础知识。

接下来的代码应该就比较简单了，都是一些控件的操作，当我们解析完数据，得到对象集合之后，我们就能够遍历这个集合，然后将每条记录，用一个 TableRow 来包含上。在异常处理中，我们弹出一个对话框，来告诉用户本次请求中出现了问题。注意到我们在处理图片的时候用到了一个 loadImage 方法，这个是我们自己定义的工具方法，用来通过图片的 url 来加载相应图片：

```
private Drawable loadImage(String url) {
    try {
```

```
            return Drawable.createFromStream((InputStream) new URL("http://www.google.com/" + url).getContent(),
                                                    "test");
        } catch (MalformedURLException e) {
            Log.e("exception",e.getMessage());
        } catch (IOException e) {
            Log.e("exception",e.getMessage());
        }
        return null;
    }
```

这个方法做的事情就是将 google 返回给我们的 url，转换为 android 中的 Drawable 对象，仔细观察的人在刚才看 xml 数据的时候可能注意到了，google 给我们提供的图片地址是相对于它的站点根目录的相对路径，所以我们在请求图片的时候，还需要加上 google 站点的前缀。这样，我们获取天气数据的逻辑就彻底完成了。

当然，基本功能虽然实现了，但作为一个应用，我们是不是应该把它的体验做的更好呢，如果我们在应用主线程中调用这个方法，就会造成同步网络通信，这个可是客户端应用程序最忌讳的东西，在通信过程中用户的界面会被完全阻塞住，非常影响体验。所以我们一定需要一个异步的通信机制来完成请求数据的过程。异步操作的话，就需要另外一个线程来获取数据并且更新视图。而在 android 中，处于安全方面的原因，非 GUI 线程是不能操作用户的界面控件的。也就是说我们没有办法在另外一个单独的线程中直接给我们的 Table 增加子控件。

但这并不代表我们就没有办法了，android 为我们提供了一种叫做 Handler 的机制来异步更新我们的界面元素，与线程不同的是，它需要一个 Message 来激活它，当然，这个听起来好像挺复杂的，不过使用起来真的很简单，就来看看下面的代码吧：


```
    private TextView txCity;
    private Button btnSearch;
    private Handler weatherHandler;
    private Dialog progressDialog;
    private Timer timer;


    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        timer = new Timer();
        txCity = (TextView)findViewById(R.id.txCity);
        btnSearch = (Button)findViewById(R.id.btnSearch);
        progressDialog = new AlertDialog.Builder(this)
        .setTitle("读取数据中")
        .setMessage("正在加载数据，请稍等")
        .create();

        weatherHandler = new Handler() {

            public void handleMessage(Message msg)
             {
               final String cityName = txCity.getText().toString();
               searchWeather(cityName);
               progressDialog.hide();
            }
        };

        btnSearch.setOnClickListener(new OnClickListener() {
```

```
        public void onClick(View v)
          {
                    progressDialog.show();
                    timer.schedule(new TimerTask()
                    {
                              public void run()
                              {
                                        Message msg = new Message();
                                        msg.setTarget(weatherHandler);
                                        msg.sendToTarget();
                    }
                  },100);
          }
        });

}
```

我们定义了几个私有成员，这其中有 TextView，Button，Handler，Dialog 和 Timer。在 onCreate 方法开始时，我们做了一些初始化操作，下面对需要讲解的地方简单说明一下：

```
progressDialog = new AlertDialog.Builder(this)
.setTitle("读取数据中")
.setMessage("正在加载数据，请稍等")
.create();
```

这段代码定义了一个对话框，用来提示用户程序正在请求天气数据，这里使用里 Builder 方式来构建对话框，到这里只是将这个对话框构造出来,但并没有显示给用户：

```
weatherHandler = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        final String cityName = txCity.getText().toString();
        searchWeather(cityName);
        progressDialog.hide();
    }
};
```

这里定义了前面提到的 Handler,注意到我们继承了这个类,并且实现了 handleMessage 方法,这是一个回调方法,需要有一个 Message 来激发它，当 Message 到来的时候，就会执行这个方法中的代码，这里面的代码是指我们主线程之外执行的，所以不必担心会阻塞用户界面，方法的实现也很简单，我们获取了文本框中输入的城市，然后调用了前面定义好的 searchWeather 方法来获取天气数据，当获取到数据之后，就会调用 hide 方法来隐藏提示退化框。

我们定义好了消息的接收者和具体处理方式，接下来就需要调用它了，大家可能已经想到了，那就是在我们前面定义的 Button 中来给 Handler 发送消息，如下代码：

```
btnSearch.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v)
         {
            progressDialog.show();
            timer.schedule(new TimerTask()
            {
```

```
                public void run()
                {
                        Message msg = new Message();
                        msg.setTarget(weatherHandler);
                        msg.sendToTarget();
                }
            },100);
        }
    });
```

发送消息也很简单吧，只需要一个 Message 对象，设置好发送目标之后就可以了。到此为止，我们的天气预报小程序就完成了，由于文章篇幅原因，在这里只能给大家一个这样的原型应用，还是以说明问题为主，所以这个程序并不是非常完善。当然，大家可以尽情发挥自己的想法和创意，一步步的完善这个程序，那么这这篇文章抛砖引玉的作用就算起到了。

下面是 Activity 的完整代码：

```
public class Main extends Activity {
      private TextView txCity;
   private Button btnSearch;
   private Handler weatherHandler;
   private Dialog progressDialog;
   private Timer timer;

     @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);
      timer = new Timer();
      txCity = (TextView)findViewById(R.id.txCity);
      btnSearch = (Button)findViewById(R.id.btnSearch);
      progressDialog = new AlertDialog.Builder(this)
      .setTitle("读取数据中")
      .setMessage("正在加载数据，请稍等")
      .create();

      weatherHandler = new Handler() {
         public void handleMessage(Message msg)
          {
                  final String cityName = txCity.getText().toString();
                   searchWeather(cityName);
                  progressDialog.hide();
         }
      };

      btnSearch.setOnClickListener(new OnClickListener() {

         public void onClick(View v)
          {
             progressDialog.show();
             timer.schedule(new TimerTask()
              {
               public void run()
               {
                  Message msg = new Message();
```

```
                msg.setTarget(weatherHandler);
                msg.sendToTarget();
            }
        },100);
    }
});
}

private void searchWeather(String city) {

    SAXParserFactory spf = SAXParserFactory.newInstance();
    try {
        SAXParser sp = spf.newSAXParser();
        XMLReader reader = sp.getXMLReader();
        XmlHandler  handler = new XmlHandler();
        reader.setContentHandler(handler);
        URL url = new URL("http://www.google.com/ig/api?hl=zh-cn&weather=" + URLEncoder.encode(city));
        InputStream is = url.openStream();
        InputStreamReader isr = new InputStreamReader(is,"GBK");
        InputSource source = new InputSource(isr);
        reader.parse(source);
        List<Weather> weatherList = handler.getWeatherList();
        TableLayout table = (TableLayout)findViewById(R.id.table);
        table.removeAllViews();
        for(Weather weather : weatherList)
         {
            TableRow row = new TableRow(this);
            row.setLayoutParams(new LayoutParams(LayoutParams.FILL_PARENT,
LayoutParams.WRAP_CONTENT));
            row.setGravity(Gravity.CENTER_VERTICAL);
            ImageView img = new ImageView(this);
            img.setImageDrawable(loadImage(weather.getImageUrl()));
            img.setMinimumHeight(80);
            row.addView(img);
            TextView day = new TextView(this);
            day.setText(weather.getDay());
            day.setGravity(Gravity.CENTER_HORIZONTAL);
            row.addView(day);
            TextView temp = new TextView(this);
            temp.setText(weather.getLowTemp() + "℃ - " + weather.getHighTemp() + "℃");
            temp.setGravity(Gravity.CENTER_HORIZONTAL);
            row.addView(temp);
            TextView condition = new TextView(this);
            condition.setText(weather.getCondition());
            condition.setGravity(Gravity.CENTER_HORIZONTAL);
            row.addView(condition);
            table.addView(row);
        }
    } catch (Exception e) {
        new AlertDialog.Builder(this)
        .setTitle("解析错误")
        .setMessage("获取天气数据失败，请稍候再试。")
        .setNegativeButton("确定", null)
        .show();
    }
```

```
    }


    //加载天气图片
    private Drawable loadImage(String url) {

        try {
            return Drawable.createFromStream((InputStream) new URL("http://www.google.com/" + url).getContent(),
"test");
        } catch (MalformedURLException e) {

            Log.e("exception",e.getMessage());
        } catch (IOException e) {
            Log.e("exception",e.getMessage());
        }
        return null;
    }
}
```

（作者：springfieldx ）

【其　他】

3.1　BUG 提交

如果你发现文档中翻译不妥的地方，请到如下地址反馈，我们会定期更新、发布更新后的版本
http://www.eoeandroid.com/thread-34359-1-1.html

3.2　关于 eoeandroid

eoeandroid 团队是一个有远大的梦想，有充沛的激情，有高效执行力的团队。北京易联致远无线技术有限公司于 2009 年 8 月成立。eoeandroid 的核心成员有在摩托罗拉、卓望科技、T3g 和手机 design house 的工作经历和相关丰富的行业经验。eoeandroid 致力于让移动互联网的软件开发变得容易，发布变得更加方便，传播变得更加迅捷,让用户以最快的速度获取最适合自己的移动互联网应用。

3.3　2010 Google AdSense 合作伙伴日研讨会火热召集中

Google 将再次携手 **eoe** 在**成都、重庆、杭州、深圳**继续开展 2010 Google AdSense 合作伙伴日全国巡回活动。欢迎各位同学，及现阶段正在积极拓展全球市场的移动应用团队来参加我们的活动。

我们相信，这场顶尖的公司举办顶尖的沙龙活动一定会让你受益匪浅，它是不可错过的！

每位到场嘉宾将获赠精美大礼品，期待您的积极参与！

小小的透露一下，上次我们社区里参加的同学获得的是漫步者的音箱还有 eoe 的最新书籍《Google Android 创赢路线与产品开发实战》哦！

同时我们诚挚的欢迎所有公司的高层管理人员参加这次大会，交流和分享最新的移动领域新机会！希望下一次，我们能离得更近！

参会城市及时间 ：
成都（11 月 23 日） 重庆（11 月 25 日） 杭州（11 月 30 日） 深圳（12 月 2 日）

详细信息及报名地址请点击→google 合作伙伴日研讨会

3.4 【eoe 小编】直击 2010 联想移动开发者大会

11 月 12 日，联想 2010 年移动互联开发者大会在北京拉开帷幕，所谓在京，其实遥远的已经快出京，真的是很遥远啊！我们 eoe 作为本次活动的协办方，被邀请参加会议室必然的。大清早赶到三元桥，坐上通往会场的大巴，居然花了将近一小时才到。

会场在一号地艺术区，不知道有没有同学去过。再次感叹一下，真的好远。不过联想的排场搞得很大，包下了整个艺术区。（照片拍的有点曝光过度了……）

在人山人海中，从签到处领了参会证件，组委会还给我们发了凡客诚品的马甲，貌似质量不太好，还标价 399，神马鬼价格啊！估计是 399 日元吧。衣服就不晒了，已经送人啦！

大会开始了，迈克尔.杰克逊的劲爆音乐响起，主持人入场，带着联想的吉祥物“乐精灵”上台，并展示了一下“乐疯”。。。

随后杨元庆登场开始演讲。他演讲的主题是：营造最佳互联网体验。元庆演讲很随和，低调而儒雅。主要是宣扬了移动互联网的好处，并且乐 phone 能带来比较好的体验，当然我们 eoe 也很好！

点击观看全文→【eoe 直击】联想集团 CEO 杨元庆：营造最佳互联网体验

联想 CTO 贺志强第三个上场，气势很有魄力，感觉震撼人心！他演讲的题目是：构建良好应用开发平台。

点击观看全文→【eoe 直击】联想副总裁贺志强：构建良好应用开发平台

上午场发布了“十大应用首发”，说是首发，但是很搞笑，有些早就在我们 eoe 的优亿市场（www.eoemarket.com)上推过了，大家有兴趣可以去下载来玩。之后还发布了乐基金，类似于天使投资的项目，做的事创业孵化的事情，这对广大的创业者来说无疑是个福音。大家好好把握机会呀！

上午场到这里就结束了，中午新裤子乐队现场演出，他们的台风让人超级疯狂，激情澎湃！

下午就几个分论坛，分别是 LBS、广告、游戏等，几个场轮流停了几个比较感兴趣的议题，总体来说本次大会还是很成功的。

很悲剧的是，抽了很多乐 phone，没一台是我的啊，于是我就带着失落的心回家啦！各位同学，一起晒晒你们参会的心情和照片咯！

（作者：Ｌｅｅｘ）

关注 eoeandroid 的新浪微薄：http://t.sina.com.cn/eoeandroid00 带给你更多好玩、有趣的 android 资讯!!

北京易联致远无限技术有限公司

电话：（010）82176766

E-mail：company@eoemobile.com

网址：www.eoemobile.com

中国最大的 Android 开发者社区：www.eoeandroid.com

中国本土的 Android 软件下载平台：www.eoemarker.com