华清远见 FARSIGHT 嵌入式培训专家

嵌入式学院 华清远见旗下品牌

ANDROID网络编程

版权声明

目录

- 获取网络状态
- 使用java.net中的相关类访问网络
- 使用apache的相关类访问网络

Android网络编程

- Android中提供了Apache HttpClient库用于网络访问
- 同时，我们也可以使用Java中的网络库来访问网络，但Android也会将其转换成Apache HttpClient库来使用
- 对于Android 2.2以上版本，Android提供了android.net.AndroidHttpClient类用于网络访问，它支持SSL连接，并针对Gzip压缩做了优化。
- 另外，要让你的应用可以访问网络，必须赋予android.permission.INTERNET的权限
- 另外还需要注意，如果你访问的本地（localhost），需要注意应该使用10.0.2.2这个IP来访问，否则这里的localhost会被Android当成是它本身
- 对于网络状态，可以通过Android提供的ConnectivityManager来判断

获取手机联网状态

ConnectivityManager

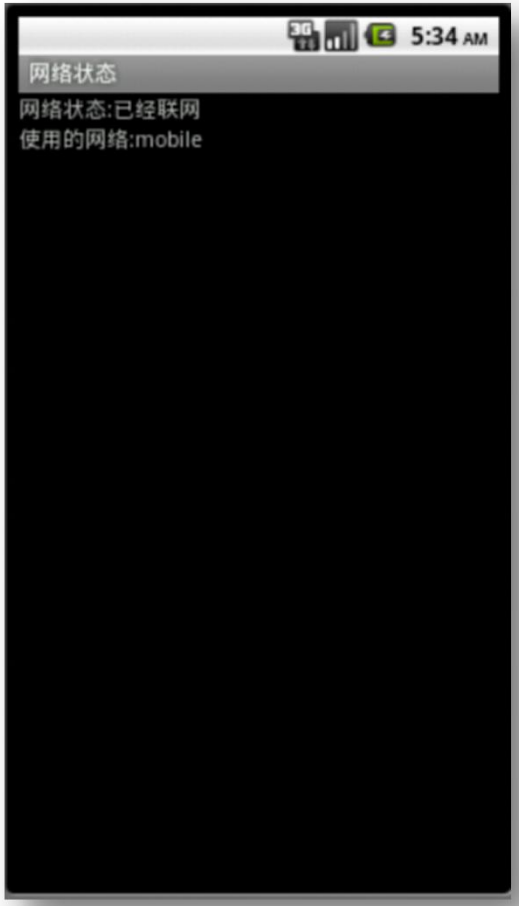
- ConnectivityManager的主要作用：
  - 监控网络连接(Wi-Fi、GPRS、 UMTS等)
  - 当网络连接状态发生改变时，发送广播消息
  - 在连接中断的时候试图转移到其他网络连接
  - 给应用提供一个查询网络状态是否可用的API——可以提供粗粒或者细粒的消息

NetworkInfo

- 通过NetworkInfo，可以得到当前所使用的网络的类型，例如：wifi或者mobile
- 另外还可以通过这个类，得到当前的网络状态，例如：

Detailed state	Coarse-grained state
IDLE	DISCONNECTED
SCANNING	CONNECTING
CONNECTING	CONNECTING
AUTHENTICATING	CONNECTING
CONNECTED	
DISCONNECTING	DISCONNECTING
DISCONNECTED	DISCONNECTED
UNAVAILABLE	DISCONNECTED
FAILED	DISCONNECTED

Android网络状态



Android网络状态

```
 ConnectivityManager connec =
(ConnectivityManager)getSystemService(Context.CONNECTIVITY_
SERVICE);
        NetworkInfo info = connec.getActiveNetworkInfo();
        if(info==null){
        tv1.setText(tv1.getText()+"没有联网");
        }else{
        tv1.setText(tv1.getText()+"已经联网");
        tv2.setText(tv2.getText()+info.getTypeName());
        }
```

注意需要加上*android.permission.ACCESS_NETWORK_STATE*权限

Android网络编程

- HTTP协议中，在客户端用于请求数据的方法常用的有2种——get和post：
  - get：一般用于请求静态页面，也可以将参数附加在URL后面请求动态页面，它传递的参数大小有限制
  - post：一般用于请求动态页面，它会把参数放在http请求的正文内传递。

使用WebView访问Web页面

WebView

- Webview是用于显示web页面的视图。
- 在活动中的任何一个浏览器或简易地显示一些在线内容。
- 它使用Webkit绘图引擎去显示页面且包含
  - 上一页
  - 下一页
  - 放大缩小
  - 文字搜索…等等方法

WebView

- loadUrl(String url)：装载url对应的网页
- setWebViewClient(WebViewClient client)：设置WebView客户端，通过WebViewClient中的回调方法实现客户端的行为，例如，对WebView中的链接点击行为做出响应等（通过shouldOverrideUrlLoading(WebView view,String url)来实现）

编写自己的浏览器


9:23 AM
Browser
http://www.farsight.com.cn
Go
网页装载
Page Loading


3:53
Browser
http://www.farsight.com.cn
Go
华清远见
FARSIGHT
企业学院
首页
关于我们
企业培训
师资团队
原创教材
全国免费
咨询电话
400-706-1880
800-810-3930
节假日值班手机：15010390988
近期免费活动
华清远见嵌入式Linux入门实战免费体验日，
7月09日，深圳，热报中！
3G移动开发入门实战免费体验日，
7月09日，北京，热报中！
2011中国嵌入式系统暨物联网教育发展高峰论坛，
7月23日，热报中！
更多>>

使用apache相关类访问网络

Android的网络编程

- 在Android中，提供了apache的网络访问类库用于处理网络访问，常用的类（接口）：
  - DefaultHttpClient
  - HttpGet/HttpPost
  - HttpResponse
  - HttpRequest接口，HttpGet/HttpPost为其实现类
  - HttpEntity：可以发送到HTTP消息或者从HTTP消息中获取，用于设置request的头讯息和请求参数，或者用于从response中返回数据
    - 当用于设置request的请求参数的时候，一般使用UrlEncodedFormEntity将其按照一定的字符编码格式进行封装

Android的网络编程

- HttpParams：接口，封装了HTTP头信息，可以通过它设置和读取Http请求头信息，例如：设置字符编码、User-Agent等
  - HttpConnectionParams：常用于设置连接超时时间（setConnectionTimeout(HttpParams params, int timeout)）、设置Socket连接超时时间（setSoTimeout(HttpParams params, int timeout)）、设置Socket缓冲大小（setSocketBufferSize(HttpParams params, int size)）等
  - HttpClientParams：设置和读取客户端的参数，主要有：是否支持认证（setAuthenticating (HttpParams params, boolean value)）、是否支持客户端重定向（setRedirecting (HttpParams params, boolean value)）以及Cookie策略（setCookiePolicy (HttpParams params, String cookiePolicy)）等
  - HttpProtocolParams：用于设置和Http协议相关的参数，例如设置User-Agent（setUserAgent(HttpParams params, String useragent)）、Http协议版本（setVersion(HttpParams params, ProtocolVersion version)）、内容体的字符编码（setContentCharset(HttpParams params, String charset)）以及HTTP头信息的字符编码（setHttpElementCharset(HttpParams params, String charset)）等

Android的网络编程

- HttpEntity常用方法：
  - InputStream getContent()
  - Header getContentEncoding()
  - long getContentLength()
  - Header getContentType()

Android网络编程

- DefaultHttpClient
  - 使用其构造器来获得一个实例
    - DefaultHttpClient(ClientConnectionManager conman, HttpParams params)
    - DefaultHttpClient(HttpParams params)
    - DefaultHttpClient()
  - 大量重载的execute()方法，常用的有：
    - execute(HttpUriRequest request)，将请求参数封装在request中
    - final HttpResponse execute(HttpHost target, HttpRequest request)，指定连接的主机，以及通过request参数指定请求方式

Android网络编程

Android网络编程

```
//获得HttpClient对象
HttpClient client = new DefaultHttpClient();
//使用Get连接方式发送请求
HttpGet request = new HttpGet(urlText.getText().toString());
HttpResponse response = client.execute(request);
// 获得响应，并使用流来读取
BufferedReader rd = new BufferedReader(new
InputStreamReader(
response.getEntity().getContent()));
String line = "";
while ((line = rd.readLine()) != null) {
textView.append(line);
}
```

Android网络编程

- 通过GET方式传递参数，实现步骤如下：
  - 将参数附加在url字符串之后，形成类似http://host/servlet.jsp?name1=value1&name2=value2的格式
  - 使用HttpGet来创建一个请求指定url的request对象
  - 通过DefaultHttpClient来得到一个HttpClient对象
  - 在HttpClient对象上调用execute方法，将request作为其参数传递进去，得到相应对象HttpResponse
  - 在HttpResponse对象上调用getStatusLine()方法，得到StatusLine对象，这里封装了关于响应状态的信息，再通过StatusLine的getStatus()方法，获得相应代码，只有返回的是200状态码时，才是获得正常的响应。可以通过和HttpStatus上的常量（如SC_OK）来比较

Android网络编程

- 通过HttpResponse上的getEntity()方法获得从服务器得到的响应数据，它被封装在HttpEntity对象中，并且对其作出相关的处理：
  - InputStream getContent()
  - Header getContentEncoding()
  - long getContentLength()
  - Header getContentType()
  - void writeTo(OutputStream outstream)：直接写入一个OutputStream

Android网络编程

Android网络编程

```
try {
    // 添加请求参数到请求对象
    for (int i = 0; i < params.size(); i++) {
        //拼接请求字符串，即name1=value1&name2=value2格式
        nvp = params.get(i);
        queryString+=nvp.getName()+"="+nvp.getValue()+"&";
    }
    //和原来的url拼接，成http://host/login.jsp?name1=value1&name2=value2的格式
    if(queryString!=""){
        url += "?"+queryString.substring(0, queryString.length()-1);
    }
    Log.e("",url);
    /* 建立HTTPGet对象 */
    HttpGet httpRequest = new HttpGet(url);

    /* 发送请求并等待响应 */
    HttpResponse httpResponse = httpClient.execute(httpRequest);
    /* 若状态码为200 ok */
    if (httpResponse.getStatusLine().getStatusCode() == 200) {
        /* 读响应数据 */
        strResult = EntityUtils.toString(httpResponse.getEntity());
    } else {
        strResult = "Error Response: "
        + httpResponse.getStatusLine().toString();
    }
}
```

Android网络编程

- 通过POST方式传递参数，实现步骤如下：
  - 使用HttpPost和url来创建一个request对象
  - 将需要传递的表单参数封装到一个List<NameValuePair>列表对象中
  - 使用UrlEncodedFormEntity()，将参数列表对象封装在这个对象中，可以指定编码格式
  - 在request对象上调用setEntity()，将前面封装了参数的Entity作为参数附加在request上
  - 通过DefaultHttpClient来得到一个HttpClient对象
  - 在HttpClient对象上调用execute方法，将request作为其参数传递进去，得到相应对象HttpResponse

Android网络编程

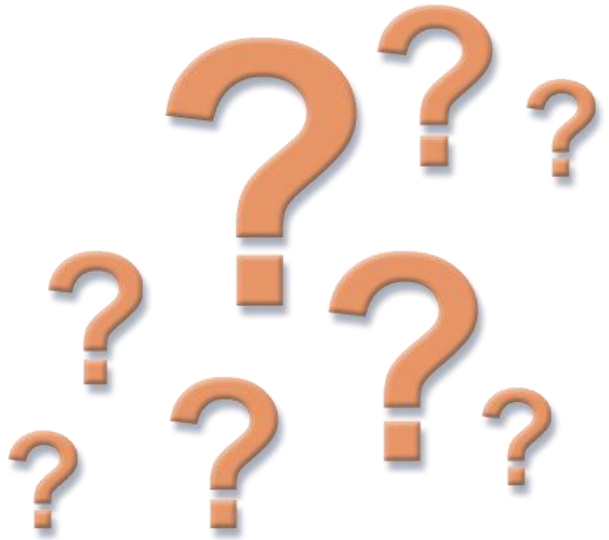
- 在HttpResponse对象上调用getStatusLine()方法，得到StatusLine对象，这里封装了关于响应状态的信息，再通过StatusLine的getStatus()方法，获得相应代码，只有返回的是200状态码时，才是获得正常的响应。可以通过和HttpStatus上的常量（如SC_OK）来比较
- 通过HttpResponse上的getEntity()方法获得从服务器得到的响应数据，它被封装在HttpEntity对象中，并且对其作出相关的处理

Android网络编程

Android网络编程

```
List<NameValuePair> params= new ArrayList<NameValuePair>();
params.add(new BasicNameValuePair("userName",
et1.getText().toString()));
params.add(new
BasicNameValuePair("password",et2.getText().toString()));
HttpPost httpRequest = new HttpPost(url);
String strResult = "doPostError";
try {
    /* 添加请求参数到请求对象 */
    httpRequest.setEntity(new UrlEncodedFormEntity(params,
    HTTP.UTF_8));
    /* 发送请求并等待响应 */
    HttpResponse httpResponse = httpClient.execute(httpRequest);
    /* 若状态码为200 ok */
    if (httpResponse.getStatusLine().getStatusCode() == 200) {
        /* 读响应数据 */
        strResult =
        EntityUtils.toString(httpResponse.getEntity());
    }
}
```

Q&A



谢谢！