#10 网络访问

目录

1. 访问因特网（HTTP）
2. Android线程模型
3. Web服务
4. 使用Ksoap2访问WebService
5. REST
6. Retrofit2访问WebService
7. 搭建Java版WebService
8. Android WiFi开发

1. 访问因特网（HTTP）

- HTTP

1. 超文本传输协议(HTTP，HyperText Transfer Protocol)是互联网上应用最为广泛的一种网络协议。

2. 最常见的从网络传输数据的方式就是使用HTTP

3. HTTP可以封装几乎所有类型的数据

1. 访问因特网（HTTP）

- Permission

要访问互联网， 首先需要设置好相应权限

文件AndroidManifest添加：

```
<uses-permission
        android:name="android.permission.INTERNET">
</uses-permission>
<uses-permission
        android:name="android.permission.CHANGE_NETWORK_STATE"
>
</uses-permission>
```

android.permission.ACCESS_NETWORK_STATE 主要是用于

访问ConnectivityManager (通常是管理网络访问连接状态)

1. 访问因特网（HTTP）

- 从Web读取数据

1. 创建一个新的URL对象
2. 为这个URL资源打开一个stream
3. 读取数据
4. 关闭InputStream

1. 访问因特网（HTTP）

- 从URL读取数据

1. 读取数据实例代码

```
URL text = new URL("http://sysu.github.io/");
InputStream inputStream = text.openStream();
byte[] bytes = new byte[250];
int readSize = inputStream.read(bytes);
Log.i("HTTP", "readSize = " + readSize);
Log.i("HTTP", "bText = " + new String(bytes));
inputStream.close();
```

2. LogCat观察输出结果

```
readSize = 250
bText = <!DOCTYPE html>
<head>
 <meta charset="utf-8" />
 <title>Homepage &#8212; OPENSYSU</title>
 <meta name="description" content="Open Source in SYSU">
 <link rel="alternate" type="application/atom+xml" href="/feed/index.xml" />
 <link rel="stylesheet" t
```

1. 访问因特网（HTTP）

- 从Web读取数据

1. 上述方法虽简单，但并不严谨
2. 没有很好的错误处理：如手机没有网络、服务器关闭、URL无效、用户操作超时
3. 因此，从一个URL读取数据值之前，往往需要了解更多的信息，例如，需要读取的数据到底有多大

1. 访问因特网（HTTP）

- 使用HttpURLConnection

1. HttpURLConnection可以对URL进行检查，避免错误地传输过多的数据

2. HttpURLConnection获取一些有关URL对象所引用的资源信息

如：HTTP状态、头信息、内容的长度、类型和日期时间等

1. 访问因特网（HTTP）

- 使用HttpURLConnection

1. 创建一个新的URL对象
2. 为这个URL资源打开Connection
3. 读取数据

1. 访问因特网（HTTP）

- 使用HttpURLConnection

1. HttpURLConnection 实例代码

```
URL text = new URL("http://sysu.github.io/");
HttpURLConnection httpURLConnection = (HttpURLConnection)
text.openConnection();
Log.i("HTTP", "respCode = " +
                httpURLConnection.getResponseCode());
Log.i("HTTP", "contentType = " +
                httpURLConnection.getContentType());
Log.i("HTTP", "content = " + httpURLConnection.getContent());
```

2. LogCat观察输出结果

```
I/HTTP: respCode = 200
I/HTTP: contentType = text/html; charset=utf-8
I/HTTP: content = buffer(com.android.okhttp.okio.GzipSource@744fe62).inputStream()
```

1. 访问因特网（HTTP）

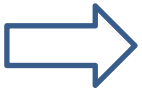
- 解析从网络获取的数据

1. 大部分网络资源的传输存储在一种结构化的形式中， 通常会使用可拓展标记语言(Extensible Markup Language, XML) 或JSON（JavaScript Object Notation）
2. Android提供了一种快速而高效的XML Pull Parse, 是网络应用程序解析器的其中一个选择
3. Google公司发布Gson-一个开放原始码的Java库, 方便了JSON与JAVA之间的转换

1. 访问因特网（HTTP）

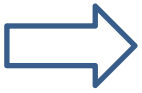
- 解析从网络获取的数据

1. XML Example



```xml
<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>
```

2. JSON Example



```json
{"employees":[
    { "firstName":"John", "lastName":"Doe" },
    { "firstName":"Anna", "lastName":"Smith" },
    { "firstName":"Peter", "lastName":"Jones" }
]}
```

1. 访问因特网（HTTP）

- 解析从网络获取的XML

1. START_TAG:找到一个新的标记时（<tag>）返回
2. TEXT:当找到文本时返回（即<tag>TEXT</tag>）
3. END_TAG:找到标记的结束时（</tag>）返回
4. END_DOCUMENT:当到达XML文件末尾时返回

1. 访问因特网（HTTP）

- 解析从网络获取的XML

1. 创建URL实例

2. 从XmlPullParserFactory中获取一个XmlPullParser实例

```
URL text = new URL( “http://.....");
XmlPullParserFactory parserCreator=xmlPullParserFactory.newInstance();
XmlPullParser parser = parserCreator.newPullParser();
parser.setInput(text.openStream(), null);
status.setText("Parsing...");
```

1. 访问因特网（HTTP）

- 解析从网络获取的XML

若想获取 https://stackoverflow.com/feeds/ 中的link Tag标签中的属性， 可以先创建好相应的XmlPullParser对象

```
XmlPullParserFactory factory = XmlPullParserFactory.newInstance();

factory.setNamespaceAware(true);

XmlPullParser parser = factory.newPullParser();
//获取XmlPullParser 实例

URL text = new URL("https://stackoverflow.com/feeds/");
//获取URL对象

parser.setInput(text.openStream(), null);
```

1. 访问因特网（HTTP）

- 解析从网络获取的XML

可以使用下方所示代码对上述例子解析

```
while(eventType != XmlPullParser.END_DOCUMENT) {
    String tagName = parser.getName();
    switch (eventType) {
        case XmlPullParser.START_TAG:
            if(tagName.compareTo("link") == 0) {
                System.out.println("rel attributeValue  : " + parser.getAttributeValue(null, "rel"));
                System.out.println("href attributeValue :  " + parser.getAttributeValue(null, "href"));
            }
            break;
        case XmlPullParser.TEXT:
            break;

        case XmlPullParser.END_TAG:
            break;
        default:
            break;
    }
    eventType = parser.next();
}
```

解析结果：

```
rel attributeValue  : self
href attributeValue : https://stackoverflow.com/feeds/
rel attributeValue  : alternate
href attributeValue : https://stackoverflow.com/questions
rel attributeValue  : alternate
href attributeValue : https://stackoverflow.com/questions/47711286/gradient-for-l1-l2-in-sgd
```