



西南交通大学
Southwest Jiaotong University

移动商务应用开发

第11章 网络编程



学习内容



10.1 HTTP简介

10.2 使用OkHttp发送请求

10.3 使用WebView显示网页

主要内容

HTTP简介



- **HTTP** (Hypertext Transfer Protocol, 超文本传输协议) 是一种**通信协议**, 它定义了**客户端 (通常指浏览器) 与服务器之间通信的格式**。
 - 客户端连上服务器后, 若想获得服务器中的某个资源, 需遵守一定的通讯格式, HTTP用于定义客户端与服务器通讯的格式。
- HTTP是一种**请求/响应式**的协议。
 - 客户端向服务器发送的请求, 称作**HTTP请求**。
 - 服务器端接收到请求后会做出响应, 称为**HTTP响应**。



HTTP简介



➤ HTTP的工作过程如下：

- 客户端与服务器**建立连接**；
- 建立连接后，客户端向服务器发送一个**HTTP请求**；
- 服务器接收到请求后，向客户端发送**响应**信息；
- 客户端与服务器**断开连接**。



发送HTTP请求

访问百度网站

HTTP响应

返回百度首页



HTTP简介



- 一个 HTTP 请求报文由**请求行**、**请求头**、**空行**和**请求体** 4 个部分组成。





HTTP请求报文



➤ 一个例子

```
POST /user HTTP/1.1           // 请求行
Host: www.user.com
Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
User-agent: Mozilla/5.0.       // 以上是请求头

username=Jack&&password=123    // 请求体(可选)
```

HTTP简介



➤ 常用的请求方法有GET和POST。

➤ GET

- 传递的参数直接表示在地址栏中，传递参数长度受限制。GET不适合用来传递私密数据，也不适合拿来传递大量数据。
- 典型场景：访问百度首页
 - 百度搜索：https://www.baidu.com/s?ie=utf-8&f=8&rsv_bp=1&tn=44004473_8_oem_dg&wd=java
 - 百度首页：<https://www.baidu.com>

HTTP简介



➤ 常用的请求方法有**GET**和**POST**。

➤ **POST**

- 把传递的数据封装在HTTP请求数据中，以**键/值对**的形式出现，不会显示在地址栏中，对数据量没有限制，可以传输大量数据。
- 典型场景：用户登录后界面
 - 需要用户提交私密数据

HTTP简介



- **URL** (Uniform Resource Locator, 统一资源定位符) 是**互联网**上用于**标识和定位资源**的地址。
- URL 是 URI 的子类。
- URL 由**协议名、主机、端口、路径、查询参数和片段标识符**等组成。
 - <http://www.example.com:80/index.html?name=John#section1>
- 请求报文中由**HOST和请求URL**共同构成了一个完整的URL。

HTTP简介



- 一个 HTTP 响应报文由**状态行**、**响应头**、**空行**和**响应体** 4 个部分组成。



HTTP简介



- 状态码由三位数字组成，第一位数字表示响应的类型，常用的状态码有五大类。
- 1xx：表示服务器已接收了客户端请求，客户端可继续发送请求；
 - 2xx：表示服务器**已成功接收**到请求并进行处理；
 - 3xx：表示服务器要求客户端**重定向**；
 - 4xx：表示**客户端**的请求有非法内容；
 - 5xx：表示**服务器**未能正常处理客户端的请求而出现意外错误；

HTTP简介



➤ 常见的状态码及其描述文本

200	OK	表示客户端请求成功
400	Bad Request	表示客户端请求有语法错误，不能被服务器所理解
401	Unauthorized:	表示请求未经授权，该状态代码必须与 WWW-Authenticate 报头域一起使用
403	Forbidden	表示服务器收到请求，但是拒绝提供服务，通常会在响应正文中给出不提供服务的原因
404	Not Found	请求的资源不存在，例如，输入了错误的URL
500	Internal Server Error	表示服务器发生不可预期的错误，导致无法完成客户端的请求
503	Service Unavailable	表示服务器当前不能够处理客户端的请求，在一段时间之后，服务器可能会恢复正常



学习内容



10.1 HTTP简介

10.2 使用OkHttp发送请求

10.3 使用WebView显示网页

主要内容



使用OkHttp发送请求



- 通过OkHttp发送HTTP请求，访问网络资源。
- 使用OkHttp之前需要在项目中**添加OkHttp库的依赖**。
 - `implementation("com.squareup.okhttp3:okhttp:4.4.0")`
- 访问网络资源之前需要在清单文件中**添加Internet访问权限**。
 - `<uses-permission android:name="android.permission.INTERNET"/>`



使用OkHttp发送请求



➤ 使用OkHttp发送**GET**请求的步骤如下：

1. 创建一个OkHttpClient对象

➤ `OkHttpClient httpClient = new OkHttpClient();`

2. 创建一个Request对象

➤ `Request request = new Request.Builder().url(url).build();`

3. 创建一个Call对象

➤ `Call call = client.newCall(request);`

4. 调用Call对象的execute()方法发送请求并获取服务器的响应。

➤ `Response response = call.execute();`

➤ `byte[] data = response.body().bytes();`



使用OkHttp发送请求



➤ 代码展示

```
OkHttpClient client = new OkHttpClient();
Request request = new Request.Builder().url(url).build();
Call call = client.newCall(request);
new Thread(new Runnable() {
    @Override
    public void run() {
        try {
            Response response = call.execute();
            byte[] data = response.body().bytes();
            html = new String(data, "UTF-8");
            handler.sendMessage(2);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}).start();
```




使用OkHttp发送请求



➤ 结果展示

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />
    <meta http-equiv="Cache-control"
content="no-cache" />
    <meta name="viewport"
content="width=device-width,minimum-scale
=1.0,maximum-scale=1.0,user-scalable=no" /
>
    <title>百度一下</title>
    <style type="text/css">
      * {
        margin: 0;
        padding: 0;
      }
      html, body {
        width: 100%;
        font-family: Arial, Helvetica, sans-serif;
      }
      body {
        font-size: 16px;
        overflow-x: hidden;
      }
      a {
        text-decoration: none;
      }
      .page {
        width: 100%;
        height: 200px;
        margin-top: 62px;
      }
```



使用OkHttp发送请求



➤ 使用OkHttp发送**POST**请求的步骤如下：

1. 创建一个OkHttpClient对象
2. 创建一个RequestBody对象存放请求数据
 - `RequestBody requestBody = new FormBody.Builder()
 .add("username", "Jack")
 .add("password", "123")
 .build();`
3. 创建一个Request对象
 - `Request request = new Request.Builder().url(url).post(requestBody).build();`
4. 创建一个Call对象
5. 调用Call对象的execute()方法发送请求并获取服务器的响应。



使用OkHttp发送请求



- `execute()`方法用于发送请求并获取服务器的响应，耗时，所以需要手动创建并启动一个**子线程**，在子线程上运行`execute()`。
- Call类提供了另外一个方法`enqueue(callback)`，**无需手动创建子线程**，可以直接写在主线程内。
- `enqueue(callback)`有两个回调方法
 - `onFailure()`
 - `onResponse()`
- `enqueue(callback)`自身创建了**子线程**，**回调方法都在子线程中运行**
- `enqueue(callback)`中如果需要更新UI，仍需要**Handler机制**



使用OkHttp发送请求



➤ 代码展示

```
OkHttpClient client = new OkHttpClient();
Request request = new Request.Builder().url(url).build();
Call call = client.newCall(request);
call.enqueue(new Callback() {
    @Override
    public void onFailure(@NonNull Call call, @NonNull IOException e) {
    }

    @Override
    public void onResponse(@NonNull Call call, @NonNull Response response) throws IOException {
        byte[] data = response.body().bytes();
        html = new String(data, "UTF-8");
        handler.sendMessage(2);
    }
});
```



学习内容



10.1 HTTP简介

10.2 使用OkHttp发送请求

10.3 使用WebView显示网页

主要内容



使用WebView显示网页



- WebView是一个**显示网页**的控件，内部采用渲染引擎来展示网页内容，提供网页前进、后退、放大、缩小及搜索功能。

方法	作用
<code>loadUrl(String url)</code>	加载Url
<code>loadDataWithBaseUrl(String baseUrl, String data, String mimeType, String encoding, String historyUrl)</code>	加载data，如html
<code>setWebViewClient(WebViewClient client)</code>	为WebView指定一个WebViewClient对象，辅助WebView处理各种通知、请求等事件。 指定后，网页跳转时目标网页仍会在当前WebView中显示，而不是打开系统浏览器
<code>getSettings()</code>	返回一个WebSettings对象，用来控制WebView的属性设置



使用WebView显示网页



➤ 代码展示

```
// 指定一个WebViewClient对象
webView.setWebViewClient(new WebViewClient());
// 加载Url
// webView.loadUrl("https://m.baidu.com");
// 或加载html
webView.loadDataWithBaseURL("", html, "text/html", "UTF-8", "");
```



使用WebView显示网页



➤ 结果展示





练习



1. 利用OkHttp获取网络图片
2. 利用OkHttp获取网页html
3. 利用WebView显示网页

本章作业

- 利用OkHttp加载图片、HTML和网页
 - 视频记录

作业提交方式

- 视频，文件名：学号+姓名+第11章作业
- 邮件给助教，主题：学号+姓名+第11章作业

