# 第8讲 HTTP程序设计

服务器地址改为：202.116.195.71

**教学与实践目的：**学会WEB浏览器基本的程序设计技术。

## 一、概述

HTTP系统包括客户端软件（浏览器）和服务器软件（HTTP服务器）。早期的客户端软件，其主要工作可理解为文件下载和文件显示。

实际上现代的HTTP客户端比文件下载要复杂得多，它包括网页文件的下载、跨平台的本地显示，参数的传递，动态网页的实现，以及交互等功能。

**HTTP系统程序设计包括：**

（1）客户端软件（web浏览器软件如edge浏览器、360浏览器）；

（2）服务器软件（web服务器软件如微软的IIS，Apache Tomcat等）。

**HTTP系统客户端的工作过程是：**

（1）客户端软件和服务器建立连接（TCP的三次握手）；

（2）发送HTTP头格式协议；

（3）接收网页文件；

（4）显示网页。

**HTTP系统服务端的工作过程：**

（1）服务器软件开启80端口；

（2）响应客户的要求、完成TCP连接；

（3）检查客户端的HTTP头格式发送客户请求的网页文件（含动态网页）。

一个完整的HTTP请求-响应如图8.1所示：



图8.1 HTTP请求-响应

本讲主要学习网页下载程序设计技术。网页下载技术是搜索引擎、网络爬虫、网页采集器或网络推送服务等相关应用领域内的基础技术。

## 二、程序设计第一步：基于TCP套接字的网页下载程序设计

利用TCP客户套接字Socket和HTTP服务器进行信息交互，将网页的原始内容下载显示在图形界面中，具体工作如下：

（1）新建一个包chapter08，将第3讲的TCPClient.java复制到此包下，重命名为HTTPClient.java；

（2）创建HTTPClientFX.java程序，界面如图8.2所示，网页地址输入www.baidu.com进行测试，端口为80，在“连接”按钮类似以往的编码方式，放置连接服务器的代码和启动输入流“读线程”。在“网页请求”按钮中发送HTTP请求头标准格式（关于HTTP请求头的更多信息可查阅互联网）。

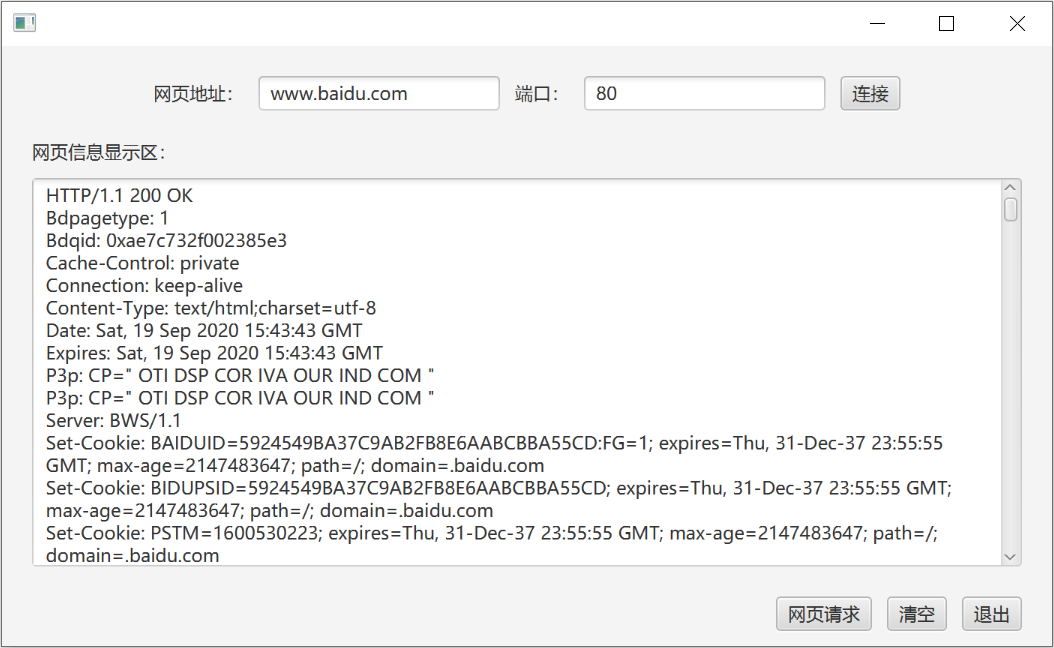


图8.2 手动发送HTTP请求的HTTP连接窗口

我们的程序可以按顺序发送以下HTTP请求头：

**GET / HTTP/1.1** //访问默认网页，注意‘/’前后要留有空格

**HOST:** address //address指服务器的IP或域名

**Accept: \*/\***

**Accept-Language:** zh-cn

**User-Agent:** User-Agent:Mozilla/5.0 (Windows NT 10.0; Win64; x64)

**Connection:** Keep-Alive

将上述的HTTP头格式构成一个字符串（通常用StringBuffer类，其toString()方法可将内容转换为字符串），一致发送到HTTP服务器。

如果网页信息显示区返回的第一条信息是“HTTP/1.1 200 OK”，则说明访问正常。

现在将网页地址改为www.gdufs.edu.cn，其它不变，再次连接并点击网页请求，结果如图8.3所示：

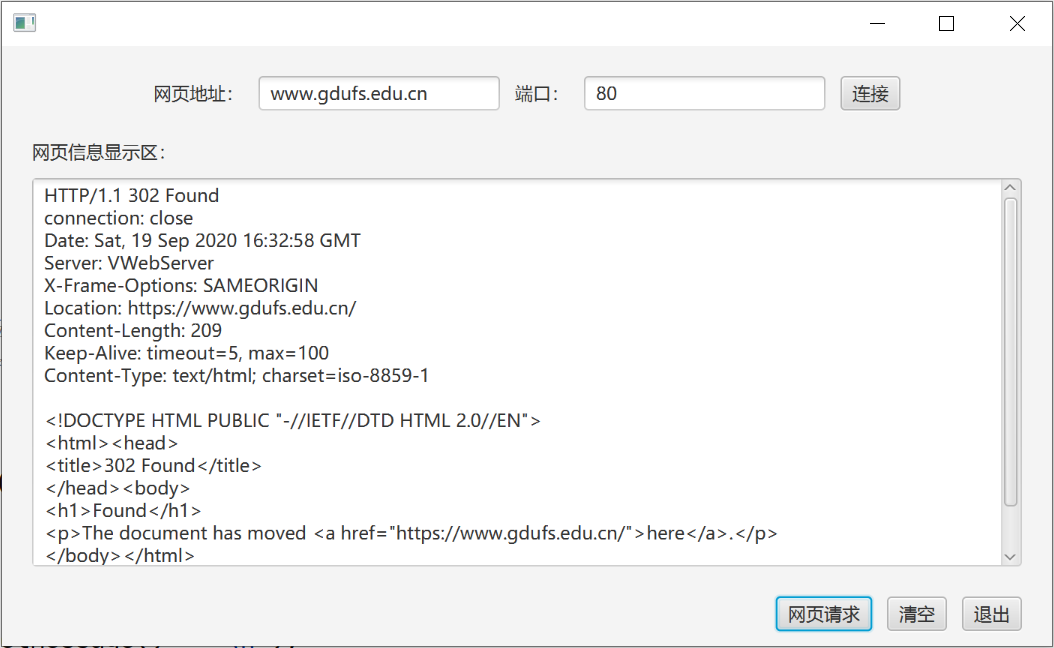


图8.3 HTTP/1.1 302 Found

这种情况HTTP服务器确实有内容返回，却不是我们预期的内容，第一行内容为“HTTP/1.1 302 Found”，这种情况一般是站点关闭了HTTP，只允许启用了SSL/TLS的HTTPS安全连接，这种连接默认是使用443端口。事实上，出于安全考虑，现在绝大部分的web站点都将放弃HTTP而启用HTTPS。

## 三、程序设计第二步：基于SSL Socket的网页下载程序设计

要访问HTTPS站点，就不能用普通的客户端套接字，而是使用SSL套接字。Java安全套接字扩展（Java Secure Socket Extension，JSSE）为基于SSL和TLS协议的Java网络应用程序提供了Java API以及参考实现，相关的类都定义在javax.net.ssl包下面，我们这里只使用其客户端的SSLSocket套接字。SSLSocket相对之前学习的客户端套接字，只是创建方法不同，SSLSocket对象由SSLSocketFactory创建，创建之后用法几乎一致。具体工作如下：

（1）根据HTTPClient.java的内容，稍作修改，新创建HTTPSClient.java程序，与前者相比，主要区别的代码如下：

import javax.net.ssl.SSLSocket;  
import javax.net.ssl.SSLSocketFactory;  
import java.io.\*;  
public class HTTPSClient {  
 *//定义SSL套接字* private SSLSocket socket;  
 *//定义SSL工厂类* private SSLSocketFactory factory;  
 *//定义字符输入流和输出流  
 ……*  
   
 public HTTPSClient(String ip, String port)   
 throws IOException {  
 factory =(SSLSocketFactory)SSLSocketFactory.*getDefault*();  
 *//创建安全套接字实例* socket = (SSLSocket)factory.createSocket(ip,  
 Integer.*parseInt*(port));

*……*

*……*

*……*  
 }

*……*

*……*  
}

（2）根据HTTPClientFX.java程序的内容，稍做修改，新创建一个HTTPSClientFX.java程序（实质上只需要将原来的使用HTTPClient及其对象实例的的地方修改为使用HTTPSClient及其对象实例即可）。

（3）运行该程序，界面如图8.4所示，可以看到已经能够正常获取到广外官网首页的内容。



图8.4 TCP SSL套接字访问下载网页内容

（4）完全没有必要使用两个不同的图形界面来分别访问http和https，创建一个HTTPAllClientFX.java，融合以上两个图形客户端的功能，使得该图形客户端即可访问443的https内容，也可以访问非443端口（一般是80）的http内容。

## 四、程序设计第三步：基于URL类的网页下载程序设计

前面直接发送http请求的程序，对于复杂的地址，例如指向具体页面的地址，无法完成网页下载任务，我们可以使用基于URL类的解决方案。

URL（Uniform Resource Locator）中文名为统一资源定位符，用于表示资源地址，资源如网页或者FTP地址等。

URL的格式为protocol://资源地址，protocol可以是HTTP、HTTPS、FTP 和 File，资源地址中可以带有端口号及查询参数，具体可以自行搜索URL的知识。

在java.net包中定义了URL类，该类用来处理有关URL的内容。并且其封装有一个InputStream返回类型的openStream()方法，我们的程序就可以读取这个字节输入流来获得对应内容。

（1）创建程序URLClientFX.java，界面布局可参见如图8.5所示：



图8.5 使用URL类

**发送按钮**事件响应中的关键代码：

taDisplay.clear();  
String address = tfSend.getText().trim();  
try {  
 URL url = new URL(address);  
 System.*out*.printf("连接%s成功！\n", address);  
  
 *//获得url的字节流输入* InputStream in = url.openStream();  
 *//装饰成字符输入流* br = new BufferedReader(new InputStreamReader(in, "utf-8"));  
  
 *//用于接收信息的单独线程*

…….

请注意URL方式和直接发送HTTP请求报头的方式，返回结果有何不同。

（2）如果用户的输入不符合URL语法，你的程序不是简单地报错，而是在显示区提醒用户输入地址不合规，如图8.6所示。



图8.6 不合规URL示意图

## 五、课堂计分

（1）使用你的URLClientFX.java程序访问<https://oc.our-web.org/ch08>，如图8.7所示。从返回的结果内容中仔细寻找有价值的信息（这个信息值作为一个key，用于下一步骤）；



图8.7

（2）使用第3讲的TCPClientThreadFX程序（或查分的LookupScoreFX程序）访问202.116.195.71 : 8008，根据服务器的提示，发送如下格式的信息：

**学号&姓名&key**

key值就是通过第（1）步骤获取的值。如果一切顺利，就会得到成功提交作业的反馈信息。

## 六、扩展练习：web浏览器

上述程序设计仅仅实现了网页文件的下载，若想要一定的网页显示功能，可以使用JavaFX的WebEngine和WebView组件。

（1）WebEngine类

WebEngine类提供了基本的web页面功能。尽管它并不与用户直接交互，但它也支持用户交互，如导航链接和提交HTML表单。WebEngine类一次处理一个web页面。它支持加载HTML内容和访问DOM对象等基本功能，也支持执行JavaScript指令。

有两个构造方法能够创建WebEngine对象：空构造和带一个URL参数的构造。如果你使用空构造方法来实例化它，那么URL可以通过WebEngine对象的load()方法来传入。WebEngine对象实例的getLocation()方法会返回当前加载页面的URL地址。

（2）WebView类

WebView类是Node类的一个扩展。它封装了WebEngine对象，将HTML内容加入程序的scene中，并且提供各种属性和方法来应用特效和变换。WebView对象的getEngine()方法返回一个与之关联的web engine，例如可以这样使用：

WebView webView = new WebView();  
WebEngine webEngine = webView.getEngine();

webEngine.load("https://www.gdufs.edu.cn");

广外首页的内容就会加载显示在webView组件中，webEngine 如果再次 load另外一个URL，则webView就自动加载另外一个页面的内容。

（3）WebHistory类

WebHistory类可获取已访问的页面列表，它表示与WebEngine对象关联的一个会话历史记录。使用WebEngine.getHistory()方法来获取某特定webEngine对象的WebHistory实例， 即：WebHistory history = webEngine.getHistory();

该历史记录基本上是一种特殊类型的列表。类似普通List，该列表也是下标从0开始，每一个条目表示一个已访问过的页面并且提供对该页面相关信息的访问。该列表可通过getEntries()方法获得。该对象的getCurrentIndex()方法返回值表示当前访问页面在列表中的索引位置。该对象的go(int index)方法表示页面的跳转，例如history.go(-1)就是返回上一个访问的页面，history.go(1)表示前进到下一个访问过的页面，这两个方法使用要注意边界条件判断，否则会抛出异常，例如已经没有可回退的页面记录，再执行回退方法就抛出异常。

（4）创建一个javaFX程序，命名为WebBrowserFX.java，参考界面如图8.8所示，完成界面所示的功能。



图8.8 WebBrowserFX参考界面

**（5）实现附加功能：**

要求“前进”和“后退”按钮能够根据实际情况自动禁用和启用；

在地址框中输入URL地址，回车后载入页面，但如果点击链接后，页面跳转，地址框的地址默认是不会跟随变化的。实现该功能：能自动更新为当前页面的URL地址。

提示：使用webEngine的状态监听器，监听页面的载入状态，只要页面有载入动作，就会触发其中的代码。在这个监听器中，我们就可以将访问的URL地址及时更新到地址栏，并设置前进后退按钮的可用性：

webEngine.getLoadWorker().stateProperty().addListener((observable, oldValue, newValue) -> {

//……… 省略，自行实现

});