

Java语言程序设计

配套教材由清华大学出版社出版发行

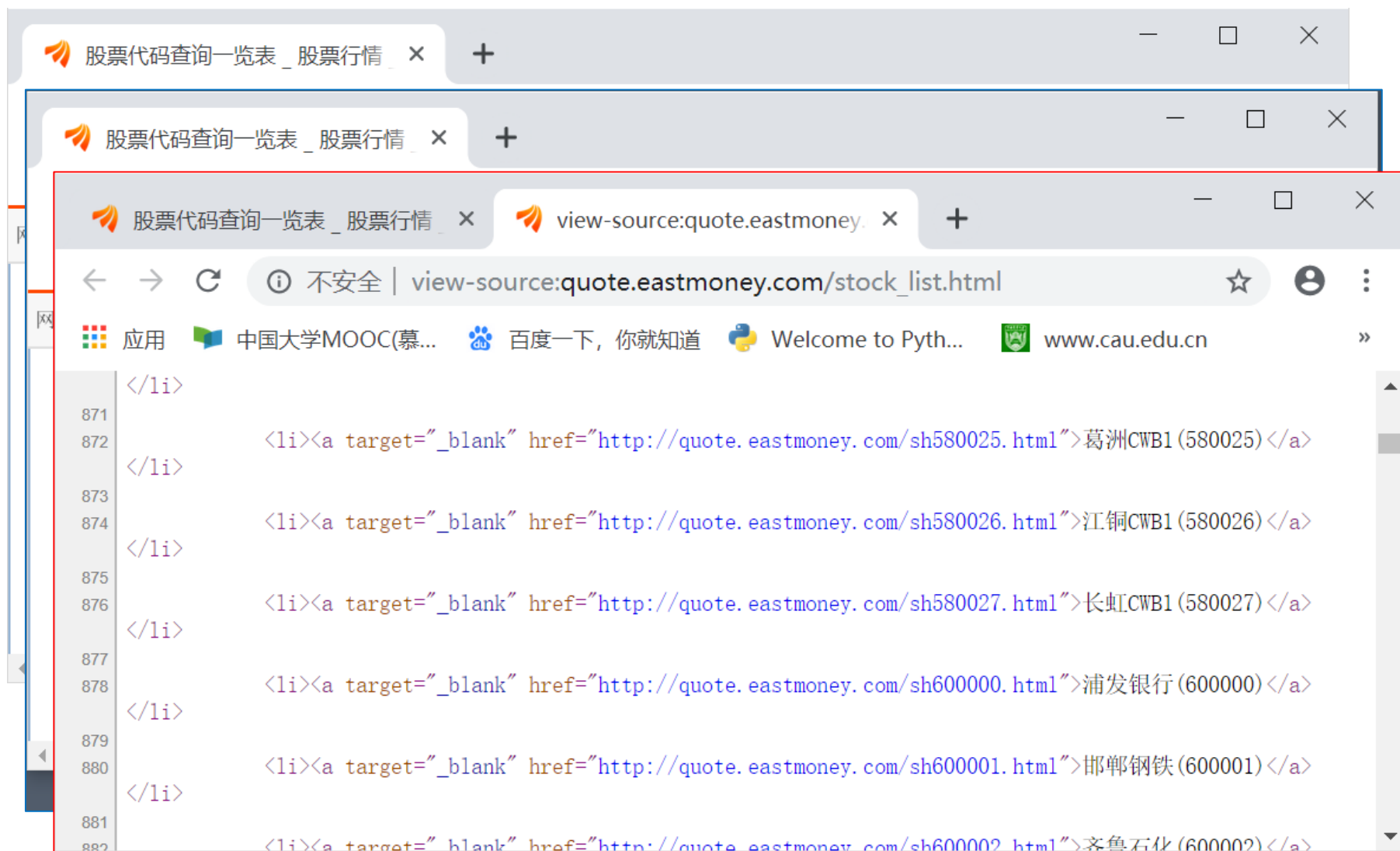
第11章 网页爬虫



中國農業大學

阚道宏

第11章 网页爬虫



第11章 网页爬虫

浦发银行(600000)个股数据查询 x +

← → ↺ <https://gupiao.baidu.com/stock/sh600000.html> ☆ 👤 ⋮

应用 中国大学MOOC(慕... 百度一下, 你就知道 Welcome to Pyth... www.cau.edu.cn »

浦发银行 (600000) 已休市 2019-05-17 15:00:04

11.24 -0.06 -0.53%

今开	成交量
11.32	38.40万手
昨收	换手率
11.30	0.14%

浦发银行(600000)个股数据查询 x view-source:https://gupiao.bai x +

← → ↺ view-source:https://gupiao.baidu.com/stock/sh600000.html ☆ 👤 ⋮

应用 中国大学MOOC(慕... 百度一下, 你就知道 Welcome to Pyth... www.cau.edu.cn »

```
59 <div class="bets-content">
60
61         <div class="line1">
62             <dl><dt>今开</dt><dd class="s-down">11.32</dd></dl>
63             <dl><dt>成交量</dt><dd>38.40万手</dd></dl>
64             <dl><dt>最高</dt><dd class="s-up">11.35</dd></dl>
65             <dl><dt>涨停</dt><dd class="s-up">12.43</dd></dl>
66             <dl><dt>内盘</dt><dd>15.07万手</dd></dl>
67             <dl><dt>成交额</dt><dd>4.33亿</dd></dl>
68             <dl><dt>委比</dt><dd>33.84%</dd></dl>
69             <dl><dt>流通市值</dt><dd>3158.86亿</dd></dl>
70             <dl><dt class="mt-1">市盈率<sup>MRQ</sup></dt><dd>5.01</dd></dl>
71             <dl><dt>每股收益</dt><dd>0.56</dd></dl>
72             <dl><dt>总股本</dt><dd>293.52亿</dd></dl>
73         <div class="clear"></div>
74     </div>
75     <div class="line2">
76         <dl><dt>昨收</dt><dd>11.30</dd></dl>
```

第11章 网页爬虫

- 本章内容
 - [11.1 HTML网页](#)
 - [11.2 HTTP协议](#)
 - [11.3 网页爬虫](#)
 - [11.4 网页信息提取](#)
 - [11.5 正则表达式](#)



11.1 HTML网页

My First Heading

My first paragraph.

```
<html>
  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>
</html>
```

- 一段被**标签**（tag）**标记**（markup）了格式和含义的**文本**（text）
- 这种标记文本的方式被称作**超文本标记语言**（**HTML**, Hyper Text Markup Language）



11.1 HTML网页

- HTML 标签
 - HTML 标签是由尖括号包围的关键词，比如 `<html>`。HTML 标签对大小写不敏感，建议使用小写
 - HTML 标签通常是成对出现的，比如 `` 和 ``
 - 标签对中的第一个标签是**开始标签**，第二个标签是**结束标签**。开始和结束标签也被称为开放标签和闭合标签
- HTML文档 = 网页
 - 保存 HTML 文件时，可以使用 `.htm` 或 `.html` 扩展名，两者没有区别



11.1 HTML网页

My First Heading

My first paragraph.

```
<html>
  <body>
    <h1>My First Heading</h1>
    <p>My first paragraph.</p>
  </body>
</html>
```

- 例子解释
 - <html> 与 </html> 之间的文本描述网页
 - <body> 与 </body> 之间的文本是可见的页面内容
 - <h1> 与 </h1> 之间的文本被显示为标题
 - <p> 与 </p> 之间的文本被显示为段落
- HTML学习网站: <http://www.w3school.com.cn/html/index.asp>



中國農業大學

閻道宏

11.1 HTML网页

- HTML 标题与段落
 - HTML 标题（heading）标签是 `<h1>` ~ `<h6>`
 - HTML 段落（paragraph）标签是 `<p>`

```
<html>
  <body>
    <h1>This is heading 1</h1>
    <h2>This is heading 2</h2>
    <h3>This is heading 3</h3>
    <h4>This is heading 4</h4>
    <h5>This is heading 5</h5>
    <h6>This is heading 6</h6>
    <p>This is text</p>
  </body>
</html>
```

This is heading 1

This is heading 2

This is heading 3

This is heading 4

This is heading 5

This is heading 6

This is text.

11.1 HTML网页

- HTML字体标签

- 标签****: 粗体
- 标签**<i>**: 斜体
- 标签**<u>**: 添加下划线

`<p class="title"> The Dormouse's story </p>`

- HTML 链接（link）标签 **<a>**

`This is a link`

- HTML 图像（image）标签 ****

``



11.1 HTML网页

- HTML 元素
 - HTML 元素以**开始标签**（start tag）起始，以**结束标签**（end tag）终止
 - 元素的**内容**（content）是开始标签与结束标签之间的内容。某些 HTML 元素可能没有内容，这时可以在开始标签中进行关闭（结束）
 - 大多数 HTML 元素可拥有**属性**（attribute）
 - 大多数 HTML 元素可以**嵌套**（即包含其他 HTML 元素）

开始标签	元素内容	结束标签
<p>	This is a paragraph	</p>
	This is a link	



11.1 HTML网页

- HTML 属性

- HTML 标签可以拥有属性。属性提供了有关 HTML 元素的更多的信息
- **属性**和**属性值**总是以键-值（key-value）对的形式出现，比如：name="value"
- 属性和属性值对大小写不敏感，建议使用小写
- 属性值应该始终被包括在引号内，双引号/单引号都可以
- 属性总是在 HTML 元素的开始标签中规定

- 举例

This is a link

<h1 align="center">

<body bgcolor="yellow">



中國農業大學

閻道宏

11.1 HTML网页

- HTML 水平线标签 **<hr />**

```
<html>
  <body>
    <p>hr 标签定义水平线: </p>
    <hr />
    <p>这是段落。 </p>
    <hr />
    <p>这是段落。 </p>
    <hr />
    <p>这是段落。 </p>
  </body>
</html>
```

hr 标签定义水平线:

这是段落。

这是段落。

这是段落。



11.1 HTML网页

- HTML 注释标签 `<!-->`
 - 可以将注释插入 HTML 代码中，这样可以提高其可读性，使代码更易被人理解
 - 浏览器会忽略注释，也不会显示它们
`<!-- This is a comment -->`
- HTML 换行 `
`
 - 无法通过在 HTML 代码中添加额外的空格或换行来改变输出的效果
 - 当显示页面时，浏览器会移除源代码中多余的空格和空行。所有连续的空格或空行都会被算作一个空格



11.1 HTML网页

- HTML 表格
 - 表格由 **<table>** 标签来定义
 - 每个表格均有若干**行**（由 **<tr>** 标签定义），每行被分割为若干**单元格**（由 **<td>** 标签定义）
 - 字母 **td** 指表格数据（table data），即数据单元格的内容。数据单元格可以包含文本、图片、列表、段落、表单、水平线、表格等
 - 表格的表头使用 **<th>** 标签进行定义（将标签**<td>**改为**<th>**）



11.1 HTML网页

```
<html>
  <body>
    <h4>两行三列: </h4>
    <table border="1">
      <tr>
        <td>100</td>
        <td>200</td>
        <td>300</td>
      </tr>
      <tr>
        <td>400</td>
        <td>500</td>
        <td>600</td>
      </tr>
    </table>
  </body>
</html>
```

两行三列:

100	200	300
400	500	600

11.1 HTML网页

- 无序列表

- 无序列表是一个项目的列表，此列项目使用粗体圆点（典型的小黑圆圈）进行标记
- 无序列表标签 ``，列表项标签 ``

- 有序列表

- 有序列表也是一列项目，列表项目使用数字进行标记
- 有序列表标签 ``，列表项标签 ``



11.1 HTML网页

```
<html>
  <body>
    <ul type="disc">
      <li>苹果</li>
      <li>香蕉</li>
      <li>柠檬</li>
      <li>桔子</li>
    </ul>

    <ol>
      <li>苹果</li>
      <li>香蕉</li>
      <li>柠檬</li>
      <li>桔子</li>
    </ol>
  </body>
</html>
```

- 苹果
- 香蕉
- 柠檬
- 桔子

1. 苹果
2. 香蕉
3. 柠檬
4. 桔子

11.1 HTML网页

- 描述列表<dl>
 - <dl> 标签定义一个描述列表
 - <dt> 标签定义一个列表项
 - <dd> 标签定义列表项的描述

```
<html>
  <body>
    <h4>定义一个描述列表: </h4>
    <dl>
      <dt>中国</dt> <dd>China</dd>
      <dt>美国</dt> <dd>USA</dd>
    </dl>
  </body>
</html>
```

定义一个描述列表:

中国	
	China
美国	
	USA

11.1 HTML网页

- HTML **<div>** 元素
 - <div> 标签可以把文档分割为独立的、不同的部分（division）
- HTML **** 元素
 - 标签可以组合文档中的行内元素，这样可以使用样式样式来格式化它们



11.1 HTML网页

- HTML 表单
 - HTML 表单用于接收用户输入
 - 使用<form> 标签定义HTML 表单，接收用户输入

```
<form action="action_page.php" method="GET" target="_blank" accept-charset="UTF-8"
ectype="application/x-www-form-urlencoded" autocomplete="off" novalidate>
  form elements
</form>
```

```
<input type="text" name="firstname">
<input type="radio" name="sex" value="male" checked>Male
<input type="submit" value="Submit">
```

- HTML 表单包含表单元素，例如 <input> 元素



11.1 HTML网页

- HTML 表单
 - **action** 属性定义在提交表单时执行的动作。向服务器提交表单的通常做法是使用提交按钮。通常，表单会被提交到 web 服务器上的网页
 - **method** 属性规定在提交表单时所用的 HTTP 方法（GET 或 POST）
 - 如果要正确地被提交，每个输入字段必须设置一个 **name** 属性



11.1 HTML网页

属性	描述
accept-charset	规定在被提交表单中使用的字符集（默认：页面字符集）
action	规定向何处提交表单的地址（URL）（提交页面）
autocomplete	规定浏览器应该自动完成表单（默认：开启）
enctype	规定被提交数据的编码（默认：url-encoded）
method	规定在提交表单时所用的 HTTP 方法（默认：GET）
name	规定识别表单的名称（对于 DOM 使用： <code>document.forms.name</code> ）
novalidate	规定浏览器不验证表单
target	规定 action 属性中地址的目标（默认：_self）



11.1 HTML网页

- HTML <head> 元素
 - <head> 元素是所有头部元素的容器，其中可以包含以下标签：<title>、<base>、<link>、<style>、<metadata>、<script>等
 - <title> 标签定义文档的标题
 - <base> 标签为页面上的所有链接规定默认路径
 - <link> 标签定义文档与外部资源之间的关系，例如连接外部样式表
 - <style> 标签用于为 HTML 文档定义样式信息
 - <metadata> 标签描述关于数据的信息，例如关键词、作者等
 - <script> 标签用于定义客户端脚本，比如 JavaScript



11.1 HTML网页

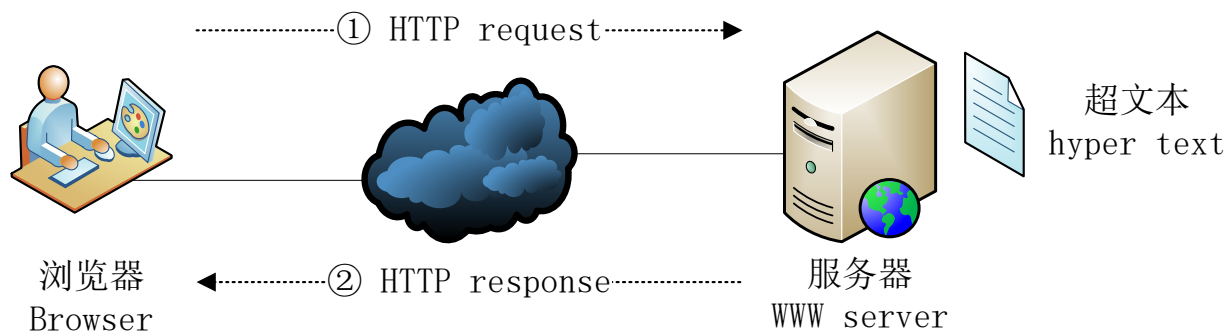
- 文档类型声明**<!DOCTYPE>**
 - <!DOCTYPE> 不是 HTML 标签，其作用是告知浏览器关于本页面的HTML 版本信息
 - <!DOCTYPE> 声明必须是 HTML 文档的第一行，位于 <html> 标签之前
 - 举例：
<!DOCTYPE html>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">



11.2 HTTP协议

- HTTP协议是**超文本传输协议**（Hyper Text Transfer Protocol）的缩写
- HTTP协议用于规范**浏览器**（browser）和WWW（World Wide Web）**服务器**之间的信息**请求**（request）与**响应**（response）服务



11.2 HTTP协议

- 统一资源定位符**URL**
 - URL的全称是 **Uniform Resource Locator**，即统一资源定位符
 - WWW服务器上的网页或图象文件等，统称为网络**资源**（resource）
 - 访问网络资源，需要使用URL来指定资源的地址
 - **http://www.cau.edu.cn:80/index.html**
 - **http://www.cau.edu.cn/images/1182/culture.jpg**
 - 统一资源定位符**URL**可以认为是网络文件的**文件名**



11.2 HTTP协议

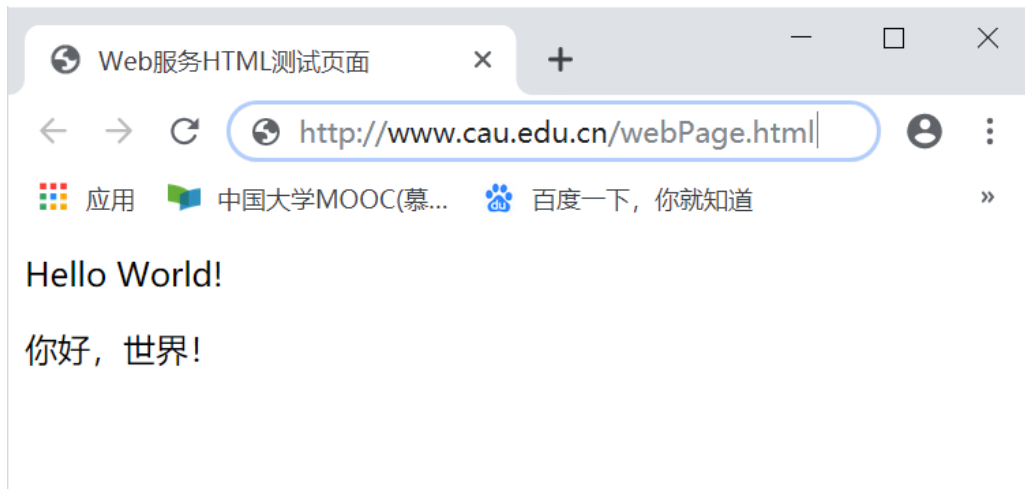
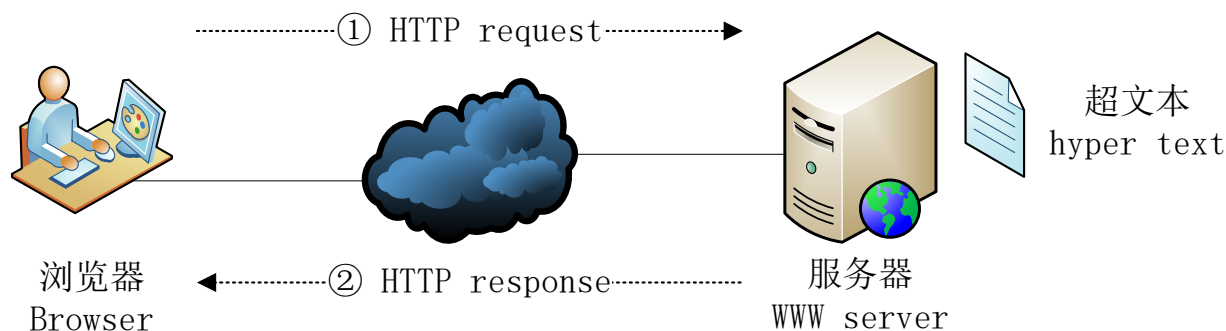
- 统一资源定位符**URL**
 - 网页也可能是由WWW服务器**程序**自动生成的动态网页
 - 访问动态网页程序时，其URL可以附加**参数**（parameter）
 - `http://www.cau.edu.cn/search?id=1234&name=Messi`
 - `http://www.cau.edu.cn/find.jsp?id=1234&name=Messi`
 - 统一资源定位符URL的**语法**
`protocol: //host [: port] path [? parameter]`



11.2 HTTP协议

- 浏览器与Web服务器

<http://www.cau.edu.cn/webPage.html>



```
<html>
<head>
  <title>Web服务HTML测试页面</title>
</head>
<body>
  <p>Hello World!</p>
  <p>你好, 世界!</p>
</body>
</html>
```



中國農業大學

閻道宏

11.2 HTTP协议

- HTTP请求 (**Request**)
 - **地址栏**: <http://www.cau.edu.cn/webPage.html>
 - **浏览器**: 先将URL包装成一个HTTP请求, 然后发送给Web服务器
 - **HTTP请求**的格式

```
GET /webPage.html HTTP/1.1
Host: www.cau.edu.cn
Connection: keep-alive
User-Agent: Mozilla/5.0 .....
Accept: text/html,*/*
Accept-Language: zh-CN
```



11.2 HTTP协议

- HTTP响应（**Response**）
 - **Web服务器**：接收HTTP请求，将所请求的信息内容（例如html文件）包装成一个HTTP响应，然后发送回浏览器
 - **浏览器**：解析信息内容（例如html文本），提取并显示其中的信息
- **HTTP响应的格式**

```
HTTP/1.1 200 OK
Server: Apache/2.4.33
Content-Length: 31201
Content-Type: text/html
<!DOCTYPE HTML>
<html lang="zh-cn">
.....
</html>
```



11.2 HTTP协议

- HTTP响应（**Response**）
 - 常见状态码
 - **200** OK # 成功
 - 400 Bad Request # HTTP请求有语法错误
 - 401 Unauthorized # HTTP请求未经授权
 - 403 Forbidden # HTTP请求被拒绝
 - **404** Not Found # 未找到资源，例如URL有误
 - **500** Internal Server Error # 服务器发生错误
 - 503 Server Unavailable # 服务器当前不可用

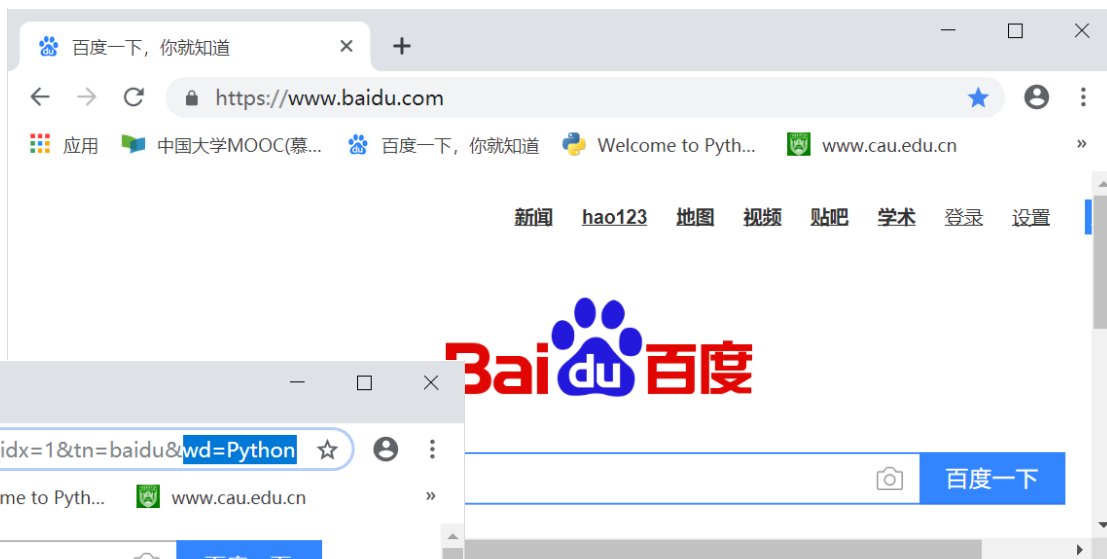


11.2 HTTP协议

- 带参数（**parameter**）的HTTP请求

- 提交**表单**页

- 返回**动态**网页



Baidu 百度

阚道宏

11.2 HTTP协议

- 带参数（**parameter**）的HTTP请求
 - 表单页


```
<html>
<head><title>html form</title></head>
<body>
  <form action="targetURL" method="GET">
    <input type="text" name="id" value="1">
    <br />
    <input type="text" name="password" value="abc">
    <br />
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

GET /targetURL?id=1&password=abc HTTP/1.1

Host: www.cau.edu.cn

Connection: keep-alive

User-Agent: Mozilla/5.0

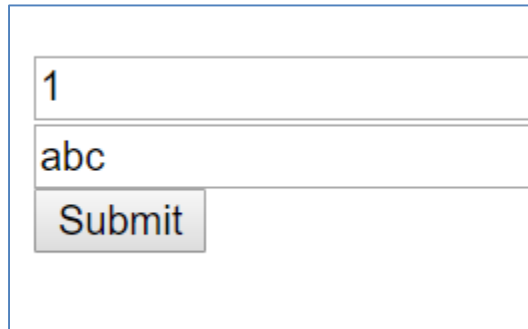
Accept: text/html, */*

Accept-Language: zh-CN

带参数的HTTP请求

11.2 HTTP协议

- 带参数（**parameter**）的HTTP请求
 - 表单页



```
<html>
<head><title>html form</title></head>
<body>
  <form action="targetURL" method="GET">
    <input type="text" name="id" value="1">
    <br />
    <input type="text" name="password" value="abc">
    <br />
  </form>
</body>
</html>
```

POST

GET /targetURL?id=1&password=abc HTTP/1.1

Host: www.cau.edu.cn

Connection: keep-alive

User-Agent: Mozilla/5.0

Accept: text/html, */*

Accept-Language: zh-CN

POST /targetURL HTTP/1.1

Host: www.cau.edu.cn

Connection: keep-alive

User-Agent: Mozilla/5.0

Accept: text/html, */*

Accept-Language: zh-CN

id=1&password=abc

11.3 网页爬虫

- 向Web服务器请求一个网页
 - 爬取一个网页
 - 爬取网页的**程序**被称为**网页爬虫**（web crawler）或**蜘蛛**（spider）
 - **模仿**浏览器**自动生成**HTTP请求，并发送给Web服务器
 - **接收**Web服务器发回的HTTP响应
 - **解析**并**提取**网页中的信息
 - 如何编写网页爬虫程序？



11.3 网页爬虫

- Java标准库：**java.net**包
 - URL、URLConnection、HttpURLConnection

java.net.URL类说明文档			
public final class URL extends Object implements Serializable			
	修饰符	类成员（节选）	功能说明
1		URL (String spec)	创建URL对象
2		URL(String protocol, String host, String file)	给定协议、主机名和文件名创建URL对象
3		URL(String protocol, String host, int port, String file)	给定协议、主机名、端口号和文件名创建URL对象
4		String getProtocol()	获取所使用的服务协议
5		String getHost()	获取主机名
6		int getPort()	获取端口号
7		int getDefaultPort()	获取服务协议的默认端口
8		String getPath()	获取资源的存储路径
9		String getFile()	获取文件名
10		String getQuery()	获取查询参数
11		InputStream openStream ()	将URL包装成输入流
12		URLConnection openConnection ()	建立URL连接
.....			

11.3 网页爬虫

- Java标准库：**java.net**包

java.net. URLConnection类说明文档			
public abstract class URLConnection extends Object			
	修饰符	类成员（节选）	功能说明
1		void setRequestProperty (String key, String value)	设置HTTP请求的头部信息
2		void setDoInput(boolean doinput)	设置是否从URL连接读信息（默认为true）
3		void setDoOutput(boolean dooutput)	设置是否向URL连接写信息（默认为false）
4		void setConnectTimeout(int timeout)	设置连接超时的时间上限
5		void setReadTimeout(int timeout)	设置下载超时的时间上限
6	abstract	void connect ()	与URL建立连接
7		Map<String,List<String>> getHeaderFields ()	读取HTTP响应的头部信息
8		String getContentEncoding()	读取HTTP响应正文的编码
9		String getContentType()	读取HTTP响应正文的类型
10		Object getContent()	读取HTTP响应的正文内容
11		InputStream getInputStream ()	获取URL连接的输入流
12		OutputStream getOutputStream ()	获取URL连接的输出流
13		URL getURL()	获取所连接的URL
14		String getRequestProperty(String key)	读取HTTP请求的头部信息



11.3 网页爬虫

- Java标准库：**java.net**包
 - URL、URLConnection、HttpURLConnection

java.net. HttpURLConnection类说明文档			
public abstract class HttpURLConnection extends URLConnection			
	修饰符	类成员（节选）	功能说明
1		int getResponseCode()	获取HTTP响应的状态码
2		String getResponseMessage()	获取HTTP响应的状态信息
3		String getRequestMethod()	获取HTTP请求所用的方法
4	abstract	void disconnect()	断开URL连接
.....			



例 11-1 使用 GET 方法访问 URL 网页的示例

```

1 import java.net.*; // 导入 java.net 包
2 import java.io.*; // 导入 java.io 包
3 import java.util.*; // 导入 java.util 包
4
5 public class HTTPURLConnectionTest {
6     public static void main(String[] args) {
7         String url = "http://www.baidu.com/";
8         String htmlPage = doGet(url);
9         // 显示 HTML 内容
10        if (htmlPage.length() > 0) {

```

```

26        // 以下代码用于处理所返回的 HTTP 响应
27        if (connection.getResponseCode() == 200) { // 检查响应状态码
28            // 读取并显示 HTTP 响应的头部信息
29            System.out.println("----- HTTP response header -----");
30            Map<String, List<String>> hf = connection.getHeaderFields();
31            for (Map.Entry<String, List<String>> entry : hf.entrySet()) {
32                String key = entry.getKey();
33                List<String> vList = entry.getValue();
34                for (String value : vList)
35                    System.out.println(key + " : " + value);
36            }
37            System.out.println("Content-Encoding : " + connection.getContentEncoding());
38            // 读取并显示 HTTP 响应正文部分的内容
39            System.out.println("----- HTTP response content -----");
40            is = connection.getInputStream(); // 获取 URL 连接的输入流
41            br = new BufferedReader(new InputStreamReader(is, "UTF-8"));
42            StringBuffer sBuf = new StringBuffer(); // 定义存放网页的缓冲区
43            String sLine = null;
44            while ((sLine = br.readLine()) != null) {
45                sBuf.append(sLine); sBuf.append("\r\n");
46            }
47            htmlPage = sBuf.toString(); // 将 StringBuffer 缓冲区传出字符串 String
48        }
49        br.close(); is.close();
50        connection.disconnect(); // 断开 URL 连接
51    }
52    catch (IOException e) { e.printStackTrace(); }
53    return htmlPage; // 返回 URL 所指定的网页内容

```

Problems @ Javadoc Declaration Console

```

<terminated> HTTPURLConnectionTest [Java Application] C:\Java\
----- HTTP response header -----
Keep-Alive : timeout=15
Accept-Ranges : bytes
X-Frame-Options : SAMEORIGIN
null : HTTP/1.1 200 OK
Server : Apache
ETag : "76e3-58c4cbf1857c0"
Connection : Keep-Alive
Vary : Accept-Encoding
Last-Modified : Thu, 27 Jun 2019 11:48:55 GMT
Content-Length : 30435
Date : Thu, 27 Jun 2019 12:03:27 GMT
Content-Type : text/html
Content-Encoding : null
----- HTTP response content -----
<!DOCTYPE HTML>

<html lang="zh-cn">

<head>

```

例 11-2 使用 POST 方法访问 URL 网页的示例代码 (HTTPURLConnectionTest.java)

1	import java.net.*; // 导入 java	41	// 以下代码用于处理所返回的 HTTP 响应
2	import java.io.*; // 导入 java	42	if (connection.getResponseCode() == 200) { // 检查响应状态码
3	import java.util.*; // 导入映	43	// 读取并显示 HTTP 响应的头部信息
4		44	System.out.println("----- HTTP response header -----");
5	public class HTTPURLConnection	45	Map<String,List<String>> hf = connection.getHeaderFields();
6	public static void main(St	46	for (Map.Entry<String,List<String>> entry : hf.entrySet()) {
7	HashMap<String, St	47	String key = entry.getKey();
8	params.put("id", "1	48	List<String> vList = entry.getValue();
9	String htmlPage = d	49	for (String value : vList)
10	if (htmlPage.length(50	System.out.println(key + " : " +value);
11	else System.out.p	51	}
12	}	52	System.out.println("Content-Encoding : " +connection.getContentEncoding());
13	public static String doPos	53	// 读取并显示 HTTP 响应正文部分的内容
14	HttpURLConnection (54	System.out.println("----- HTTP response content -----");
15	OutputStream os = n	55	is = connection.getInputStream(); // 获取 URL 连接的字节型输入流
16	PrintWriter pw = null	56	br = new BufferedReader(new InputStreamReader(is, "UTF-8"));
17	InputStream is = null	57	StringBuffer sBuf = new StringBuffer(); // 定义存放网页的缓冲区
18	BufferedReader br =	58	String sLine = null;
19	String htmlPage = nu	59	while ((sLine = br.readLine()) != null) {
20	try {	60	sBuf.append(sLine); sBuf.append("\r\n");
21	URL url = new U	61	}
22	connection = (H	62	htmlPage = sBuf.toString(); // 将 StringBuffer 缓冲区传出字符串 String
23	// 先设置 HTTP	63	}
24	connection.setR	64	br.close(); is.close();
25	connection.setC	65	connection.disconnect(); // 断开 URL 连接
26	connection.setR	66	}
27	connection.setR	67	catch (IOException e) { e.printStackTrace(); }
28		68	return htmlPage; // 返回 URL 所指定的网页内容
29	connection.setD	69	} }

11.4 网页信息提取

浦发银行(600000)个股数据查询 × view-source:https://gupiao.baidu.com/stock/sh600000

← → ↻ https://gupiao.baidu.com/stock/sh600000

应用 中国大学MOOC(慕... 百度一下, 你就知道 We

浦发银行 (600000) 已休市 2019-05-17 15:00:04

11.24 -0.06 -0.53%

今开	成交量	最高	涨停	内盘
11.32	38.40万手	11.35	12.43	15.07万手
昨收	换手率	最低	跌停	外盘
11.30	0.14%	11.20	10.17	23.33万手

```
45 <div class="stock-info" data-spm="2">
46   <div class="stock-bets">
47     <h1>
48       <a class="bets-name" href="/stock/sh600000.html">
49         浦发银行 (<span>600000</span>)
50       </a>
51       <span class="state f-up">未开盘 2019-05-20 &nbsp;15:00:03
52     </span>
53   </h1>
54   <div class="price s-stop ">
55     <strong class="_close">--</strong>
56     <span>--</span>
57     <span>--</span>
58   </div>
59   <div class="bets-content">
60
61     <div class="line1">
62       <dl><dt>今开</dt><dd class="">11.28</dd></dl>
63       <dl><dt>成交量</dt><dd>34.90万手</dd></dl>
64       <dl><dt>最高</dt><dd class="s-up">11.44</dd></dl>
65       <dl><dt>涨停</dt><dd class="s-up">12.36</dd></dl>
66       <dl><dt>内盘</dt><dd>16.93万手</dd></dl>
67       <dl><dt>成交额</dt><dd>3.96亿</dd></dl>
68       <dl><dt>委比</dt><dd>-16.60%</dd></dl>
69       <dl><dt>流通市值</dt><dd>3186.97亿</dd></dl>
70       <dl><dt class="mt-1">市盈率<sup>MRQ</sup></dt><dd>5.06</dd></dl>
71       <dl><dt>每股收益</dt><dd>0.56</dd></dl>
72       <dl><dt>总股本</dt><dd>293.52亿</dd></dl>
73     </div>
74   </div>
75   <div class="line2">
76     <dl><dt>昨收</dt><dd>11.24</dd></dl>
77     <dl><dt>换手率</dt><dd>0.12%</dd></dl>
78     <dl><dt>最低</dt><dd class="s-up">11.25</dd>
79   </dl>
80   <dl><dt>跌停</dt><dd class="s-down">
81     10.12</dd></dl>
```

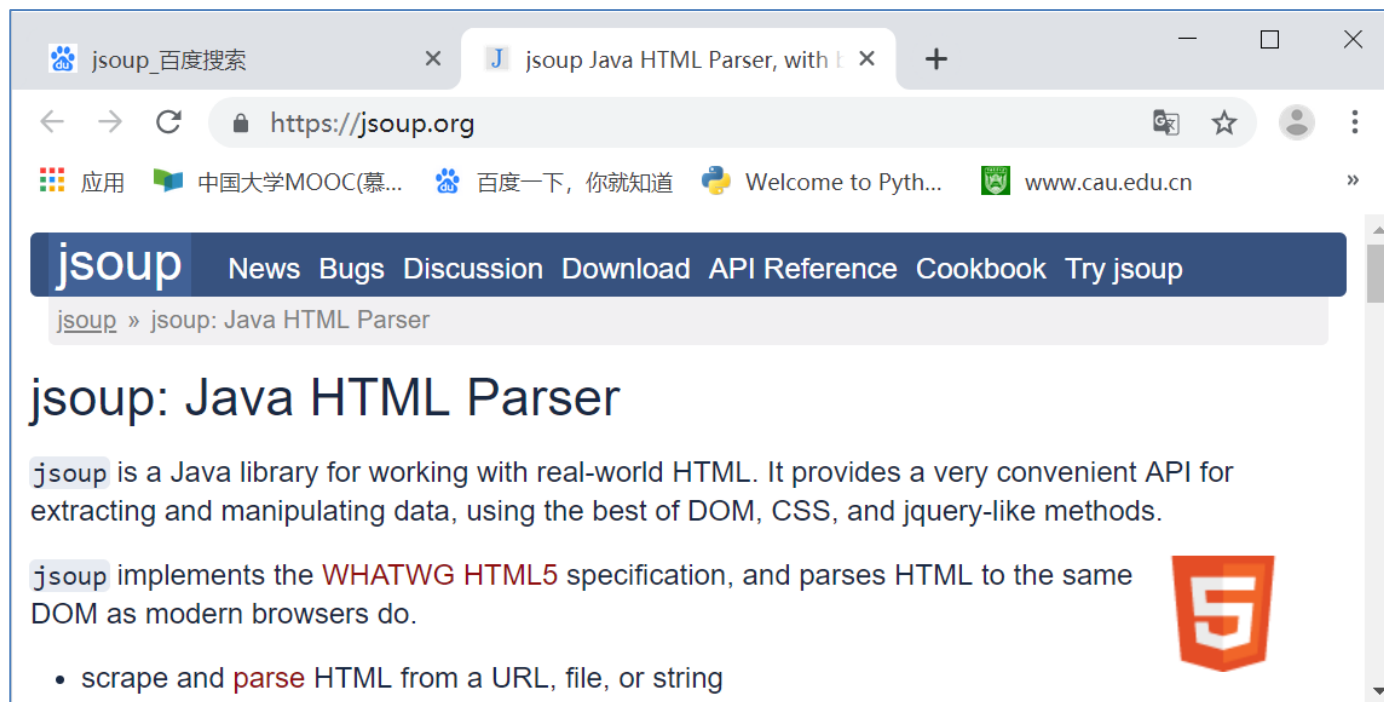
11.4 网页信息提取

- 网页信息提取步骤
 - 人工分析
 - 网页解析
 - 信息提取
 - 信息存储

```
45 <div class="stock-info" data-spm="2">
46   <div class="stock-bets">
47     <h1>
48       <a class="bets-name" href="/stock/sh600000.html">
49         浦发银行 (<span>600000</span>)
50       </a>
51       <span class="state f-up">未开盘 2019-05-20 &nbsp;15:00:03
52     </span>
53   </h1>
54   <div class="price s-stop">
55     <strong class="_close">--</strong>
56     <span>--</span>
57     <span>--</span>
58   </div>
59   <div class="bets-content">
60
61     <div class="line1">
62       <dl><dt>今开</dt><dd class="">11.28</dd></dl>
63       <dl><dt>成交量</dt><dd>34.90万手</dd></dl>
64       <dl><dt>最高</dt><dd class="s-up">11.44</dd></dl>
65       <dl><dt>涨停</dt><dd class="s-up">12.36</dd></dl>
66       <dl><dt>内盘</dt><dd>16.93万手</dd></dl>
67       <dl><dt>成交额</dt><dd>3.96亿</dd></dl>
68       <dl><dt>委比</dt><dd>-16.60%</dd></dl>
69       <dl><dt>流通市值</dt><dd>3186.97亿</dd></dl>
70       <dl><dt class="mt-1">市盈率<sup>MRQ</sup></dt><dd>5.06</dd></dl>
71       <dl><dt>每股收益</dt><dd>0.56</dd></dl>
72       <dl><dt>总股本</dt><dd>293.52亿</dd></dl>
73     </div>
74   </div>
75   <div class="line2">
76     <dl><dt>昨收</dt><dd>11.24</dd></dl>
77     <dl><dt>换手率</dt><dd>0.12%</dd></dl>
78     <dl><dt>最低</dt><dd class="s-up">11.25</dd>
79   </dl>
80     <dl><dt>跌停</dt><dd class="s-down">
81       10.12</dd></dl>
```

11.4 网页信息提取

- 第三方库：**jsoup**包
 - 主页：<https://jsoup.org/>

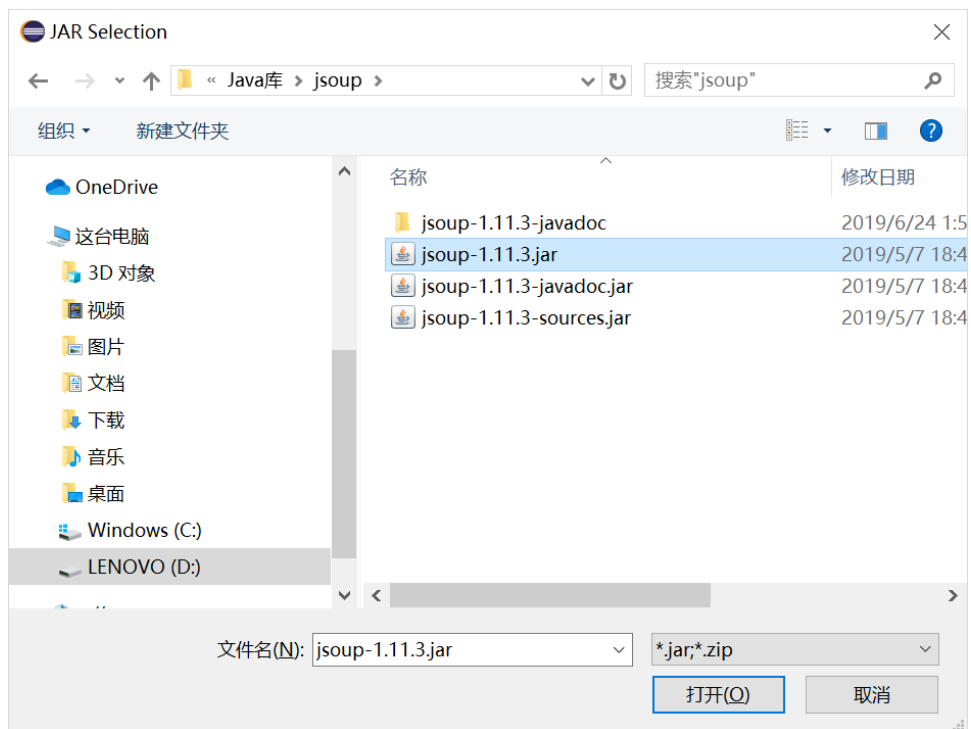


中國農業大學

閻道宏

11.4 网页信息提取

- 第三方库：**jsoup**包
 - 下载jar包：**jsoup-1.11.3.jar**
 - 在Eclipse项目中导入这个外部jar包



11.4 网页信息提取

test.html - 记事本

文件(E) 编辑(E) 格式(O) 查看(V) 帮助(H)

html网页-文档Document

<html>
<head> <title>The Dormouse's story</title>
<body>
<p class="storytitle"> The Dormouse's story </p>
<p class="story">Once upon a time there were three little sisters; and their names were
Elsie, Lacie and
Tillie and they all lived at the bottom of a well.
</p>
<p class="story">...</p>
</body>
</html>

标签<body>-元素Element

标签<a>-元素Element

< 第 12 行, 第 8 列 >



11.4 网页信息提取

- 第三方库: **jsoup**包

org.jsoup.Jsoup类说明文档			
public class Jsoup extends Object			
	修饰符	类成员（节选）	功能说明
1	static	Document parse(String html)	解析字符串中的html
2	static	Document parseBodyFragment(String bodyHtml)	解析字符串中的html正文（body）
3	static	Document parse(File in, String charsetName)	解析文件中的html
4	static	Document parse(URL url, int timeoutMillis)	解析url指定的html
.....			

org.jsoup.nodes.Document类说明文档			
public class Document extends Element			
	修饰符	类成员（节选）	功能说明
1		Charset charset()	返回html的字符编码
2		void charset(Charset charset)	设置html的字符编码
3		Element head()	读取html的头部
4		Element body()	读取html的正文部分
5		String title()	读取html的标题
6		String nodeName()	返回文档的节点名称
.....			



11.4 网页信息提取

org.jsoup.nodes.Element类说明文档

public class Element

extends Node

	修饰符	类成员（节选）	功能说明
1		String tagName()	返回元素的标签名
2		String attr(String attributeKey)	返回指定属性的属性值
3		boolean hasAttr(String attributeKey)	检查元素是否有某个属性
4		Attributes attributes()	返回元素的属性清单
5		String className()	返回元素的class属性
6		boolean hasClass(String className)	检查元素是否有class属性
7		String id()	返回元素的id属性
8		String data()	返回元素的内部数据
9		String text()	返回元素及其子元素的可见文本
10		boolean hasText()	检查元素及其子元素是否有可见文本
11		String ownText()	返回元素自己（不含子元素）的可见文本
12		Element getElementById(String id)	根据id属性查找子元素
13		Elements getAllElements()	查找所有的子元素
14		Elements getElementsByAttribute(String key)	查找所有具有属性key的子元素
15		Elements getElementsByAttributeValue(String key, String value)	根据属性查找子元素
16		Elements getElementsByAttributeValueMatching(String key, String regex)	根据属性（正则表达式）查找子元素
17		Elements getElementsByClass(String className)	根据class属性查找子元素
18		Elements getElementsByTag(String tagName)	根据标签名查找子元素
19		Elements select(String cssQuery)	根据选择器查找所有符合条件的子元素
20		Element selectFirst(String cssQuery)	根据选择器查找符合条件的第一个子元素

.....



11.4 网页信息提取

例 11-3 使用 jsoup 库解析 html 网页的示例代码 (JSoupTest.java)

```

1  import org.jsoup.Jsoup;           18      // 解析网页中的标签 (tag), jsoup 将标签称为元素 (Element)
2  import org.jsoup.nodes.Element;    19      Element ep = doc.selectFirst("p"); // 通过标签名提取标签, 例如<p>标签
3  import java.io.*;                  20      System.out.println("tagName(): " + ep.tagName()); // 提取标签的名称
4                                      21      System.out.println("className(): " + ep.className()); // 提取标签的 class 属性
5  public class JSoupTest {           22      System.out.println("text(): " + ep.text()); // 提取标签的复合文本
6      public static void main(String[] args) { 23      System.out.println("ownText(): " + ep.ownText()); // 提取标签自己的文本
7          try {                        24      System.out.println("-----");
8                                      25      // 通过 id 属性提取标签, 例如: 提取 id="link2"的标签
9                                      26      Element ea = doc.getElementById("link2"); // 提取所有 id="link2"的标签
10                                     27      //Element ea = doc.selectFirst("a#link2"); // 提取第一个 id="link2"的<p>标签
11                                     28      //Element ea = doc.selectFirst("a[id=link2]"); // 提取第一个 id="link2"的<p>标签
12                                     29      System.out.println("id: " + ea.id()); // 提取 id 属性
13                                     30      System.out.println("<html>"); // 提取整个 HTML 文档

```

```

<terminated> JSoupTest [Java Application] C:\Java\jre1.8.0_152\bin\javaw.exe
tagName(): #root
text(): The Dormouse's story The Dormouse's story Once upon a
-----
tagName(): p
className(): storytitle
text(): The Dormouse's story
ownText():
-----
tagName(): a
className(): sister
text(): Lacie
ownText(): Lacie

```



test.html - 记事本

文件(E) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
<html>
<head> <title>The Dormouse's story</title> </head>
<body>
<p class="storytitle"> <b>The Dormouse's story</b> </p>
<p class="story">Once upon a time there were three little sisters; and their names were
<a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
<a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
<a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>; and they lived at the bottom of a well.
</p>
<p class="story">...</p>
</body>
</html>
```

< 第 12 行, 第 8 列

11.4 网页信息提取

- 第三方库: **jsoup**包



中國農業大學

阚道宏

11.4 网页信息提取

例 11-4 使用 jsoup 库从 html 网页中提取股票信息的示例程序 (StockInfo.java)

```
1 import org.jsoup.*; // 导入
2 import org.jsoup.nodes.*; // 导入
3 import org.jsoup.select.*; // 导入
4 import java.net.*; // 导入
5 import java.util.*; // 导入
6
7 public class StockInfo {
8     public static void main(String[] args) {
9         try {
10             URL url = new URL("http://www.pd.com.cn/stock/");
11             Document doc = Jsoup.connect(url).get();
12             //System.out.println(doc.html());
13             // 提取网页中的股票信息
14             ArrayList<String> keys = new ArrayList<>();
15             ArrayList<String> values = new ArrayList<>();
16             // 查找 class 属性
17             Element divInfo = doc.selectFirst("div.stock-bets");
18             // 查找 class 属性为"bets-name"的<a>标签
19             Element eName = divInfo.selectFirst("a.bets-name"); // 角
20             String sName = eName.ownText(); // 提取股票名称文本
21             sName = sName.replace("()", ""); // 去除多余字符
22             key.add("股票名称"); value.add(sName);
23             // 查找 class
24             Element ePrice = divInfo.selectFirst("div.stock-price");
25             String sPrice = ePrice.ownText();
26             key.add("股票价格"); value.add(sPrice);
27             // 查找所有的
28             Elements eDT = divInfo.select("div.stock-dt");
29             Elements eDD = divInfo.select("div.stock-dd");
30             for (int n = 0; n < eDT.size(); n++) {
31                 String k = eDT.get(n).ownText();
32                 String v = eDD.get(n).ownText();
33                 key.add(k); value.add(v);
34             }
35             for (int n = 0; n < eDT.size(); n++) {
36                 System.out.println(k + ": " + v);
37             }
38             catch (Exception e) {
39             }
40         }
41     }
42 }
```

Problems @ Javadoc Declaration Console

<terminated> StockInfo [Java Application] C:\Java\jre

股票名称： 浦发银行
股票价格： **11.68**
今开： **11.67**
成交量： **29.55**万手
最高： **11.68**
涨停： **12.80**
内盘： **12.53**万手
成交额： **3.43**亿
委比： **-28.71%**
流通市值： **3282.52**亿
市盈率： **5.21**
每股收益： **0.56**
总股本： **293.52**亿
昨收： **11.64**
换手率： **0.11%**
最低： **11.54**
跌停： **10.48**
外盘： **17.01**万手
振幅： **1.20%**
量比： **2.31**
总市值： **3428.32**亿
市净率： **0.75**
每股净资产： **15.63**
流通股本： **281.04**亿

11.4 网页信息提取

- 第三方库: **jsoup**包

浦发银行(600000)个股数据查询 × +

← → ↻ <https://gupiao.baidu.com/stock/sh600000.html>

应用 中国大学MOOC(慕... 百度一下, 你就知道 Welcome to Pyth.

 **股市通**
选股更轻松

输入股票名称/代码/首字母

浦发银行 (600000) 已休市 2019-06-28 15:00:13

11.68 +0.04 +0.34%

今开	成交量	最高	涨停	内盘	成交额
11.67	29.55万手	11.68	12.80	12.53万手	3.43亿
昨收	换手率	最低	跌停	外盘	振幅
11.64	0.11%	11.54	10.48	17.01万手	1.20%

Problems @ Javadoc Declaration Console

<terminated> StockInfo [Java Application] C:\Java\jre

股票名称: 浦发银行
股票价格: 11.68
今开: 11.67
成交量: 29.55万手
最高: 11.68
涨停: 12.80
内盘: 12.53万手
成交额: 3.43亿
委比: -28.71%
流通市值: 3282.52亿
市盈率: 5.21
每股收益: 0.56
总股本: 293.52亿
昨收: 11.64
换手率: 0.11%
最低: 11.54
跌停: 10.48
外盘: 17.01万手
振幅: 1.20%
量比: 2.31
总市值: 3428.32亿
市净率: 0.75
每股净资产: 15.63
流通股本: 281.04亿

11.4 网页信息提取

- 第三方库：**jsoup**包
 - 元素（标签）类：`org.jsoup.nodes.Element`

`<p class="storytitle">The Dormouse's story</p>`

`.tagName()`
标签名

`.attrs()`
标签属性

`.text()`
标签文本

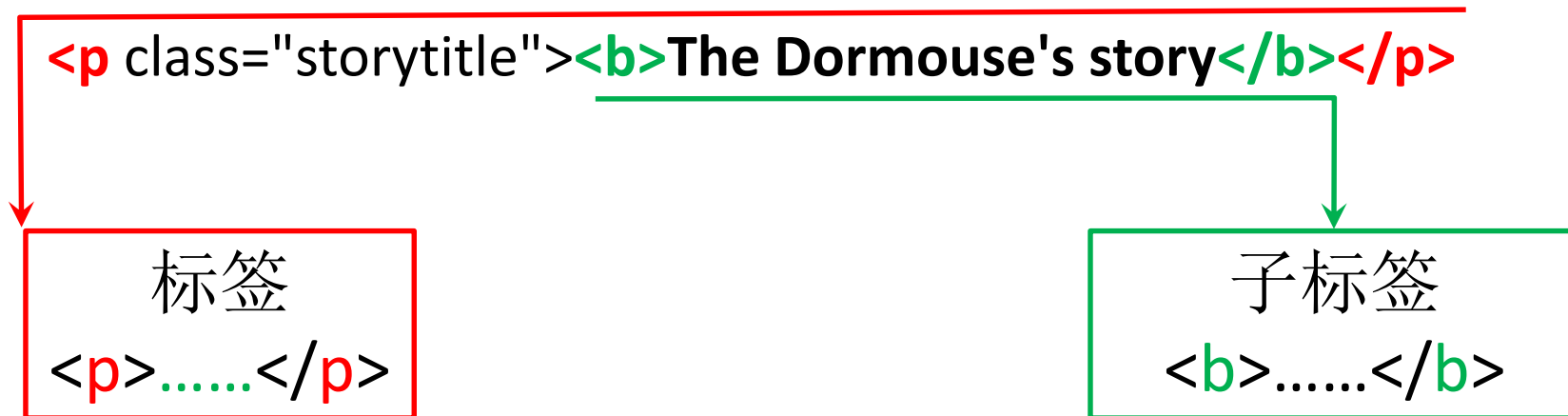


中國農業大學

阚道宏

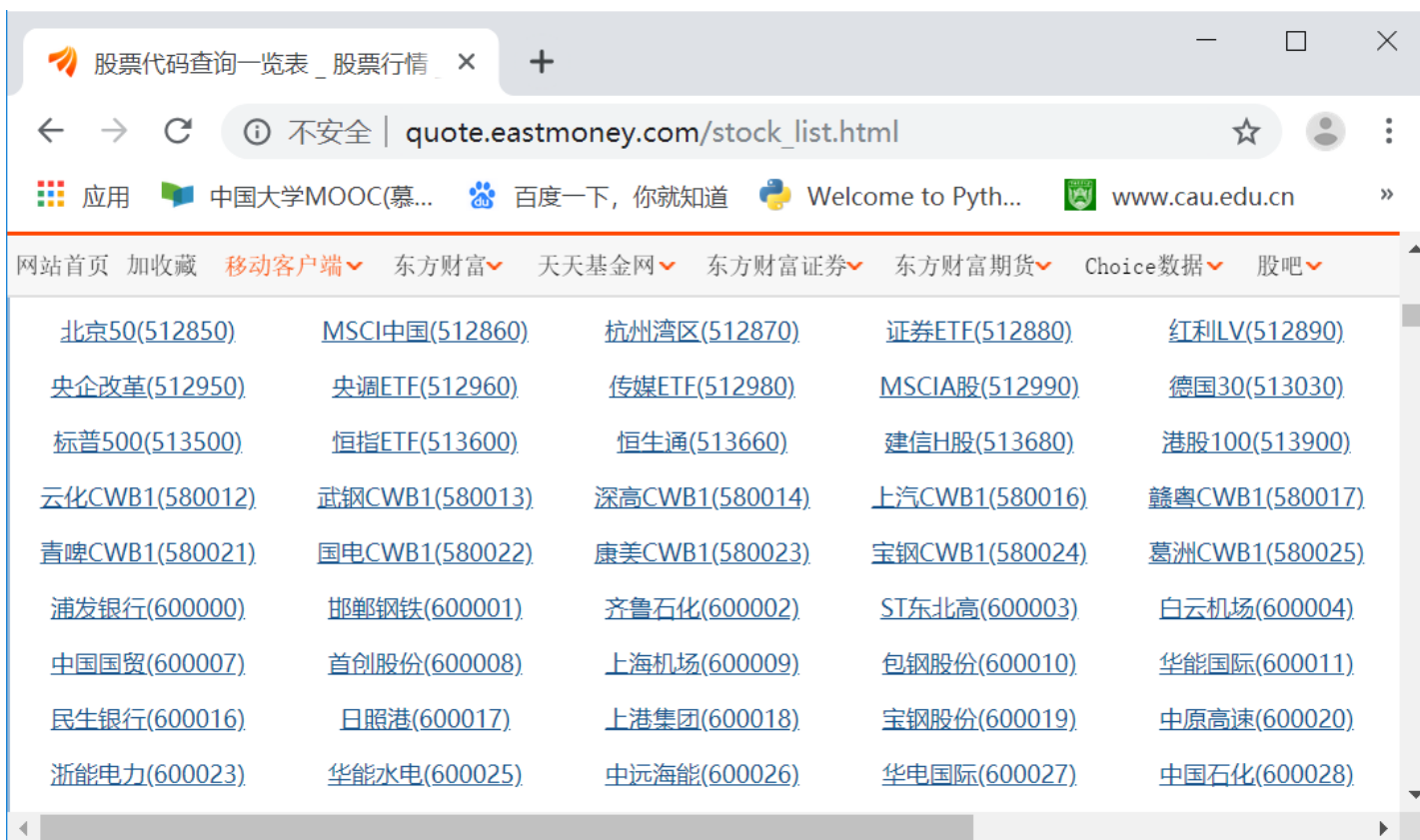
11.4 网页信息提取

- 第三方库：**jsoup**包
 - 元素（标签）类：`org.jsoup.nodes.Element`



11.4 网页信息提取

- 第三方库: **jsoup**包

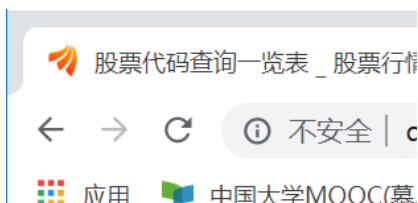


中國農業大學

阚道宏

11

• 第三方库:



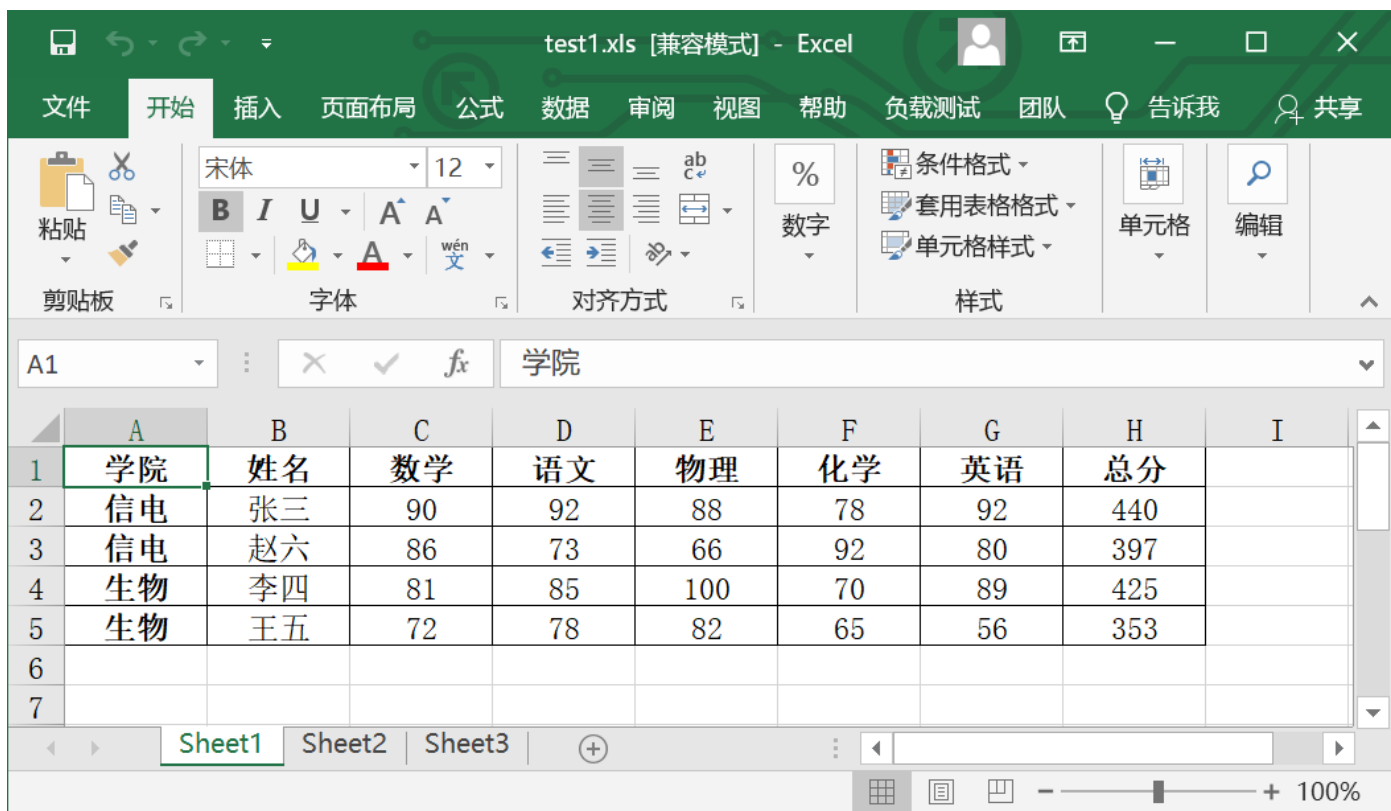
Problems @ Javadoc De
<terminated> StockList [Java

sh600000
sh600001
sh600002
sh600003
sh600004
sh600005
sh600006
sh600007
sh600008
sh600009
sh600010
sh600011

例 11-5 使用 jsoup 库从 html 网页中提取股票代码的示例程序 (StockInfo.java)

```
1  import org.jsoup.*;           // 导入 org.jsoup 包中的类 Jsoup
2  import org.jsoup.nodes.*;     // 导入 org.jsoup.nodes 子包中的类 Document、Element 等
3  import org.jsoup.select.*;    // 导入 org.jsoup.select 包中的类 Jsoup
4  import java.net.*;            // 导入 java.net 包中的类 URL
5  import java.util.regex.*;      // 导入 java.util.regex 包中与正则表达式相关的类
6
7  public class StockList {
8      public static void main(String[] args) {
9          try {
10              URL url = new URL("http://quote.eastmoney.com/stock_list.html");
11              Document doc = Jsoup.parse(url, 60000); // 下载并解析 url 指定的网页
12              //System.out.println("tagName(): " + doc.tagName());
13              // 提取网页中所有的股票代码: 查找 class 属性为"quotebody"的<div>标签
14              Element divInfo = doc.selectFirst("div.quotebody");
15              // 指定 href 属性的正则表达式, 然后查找所有符合条件的标签
16              String regex = "a[href~=http.+s[hz][603]\\d{5}\\d\\.html]";
17              Elements eNo = divInfo.select(regex);
18              // 遍历所查找到的标签: 先提取标签的 href 属性, 再提取其中的股票代码
19              Pattern p = Pattern.compile("s[hz][603]\\d{5}"); // 股票代码的正则表达式
20              for (int n = 0; n < eNo.size(); n++) {
21                  String h = eNo.get(n).attr("href"); // 提取标签中的 href 属性文本
22                  Matcher m = p.matcher(h);           // 使用正则表达式提取其中的股票代码
23                  if (m.find() == true) {              // 存在符合正则表达式的股票代码
24                      int p1 = m.start(), p2 = m.end();
25                      System.out.println( h.substring(p1, p2) );
26                  } }
27              }
28              catch (Exception e) { e.printStackTrace(); }
29      } }
```

11.5 电子表格信息提取



The screenshot shows the Microsoft Excel interface with a spreadsheet titled 'test1.xls [兼容模式] - Excel'. The ribbon is set to '开始' (Home). The spreadsheet contains a table with 9 columns (A-I) and 7 rows (1-7). The data is as follows:

	A	B	C	D	E	F	G	H	I
1	学院	姓名	数学	语文	物理	化学	英语	总分	
2	信电	张三	90	92	88	78	92	440	
3	信电	赵六	86	73	66	92	80	397	
4	生物	李四	81	85	100	70	89	425	
5	生物	王五	72	78	82	65	56	353	
6									
7									

- 第三方库：POI库

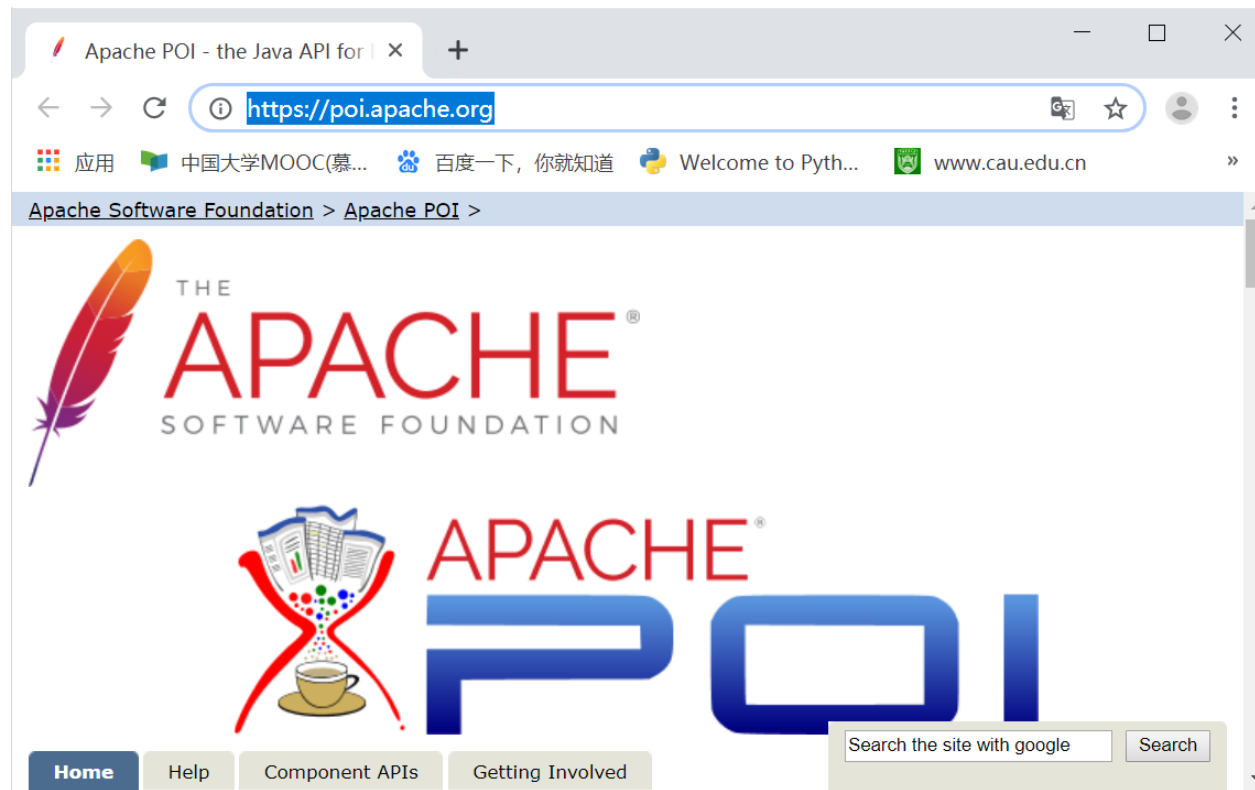


中國農業大學

閻道宏

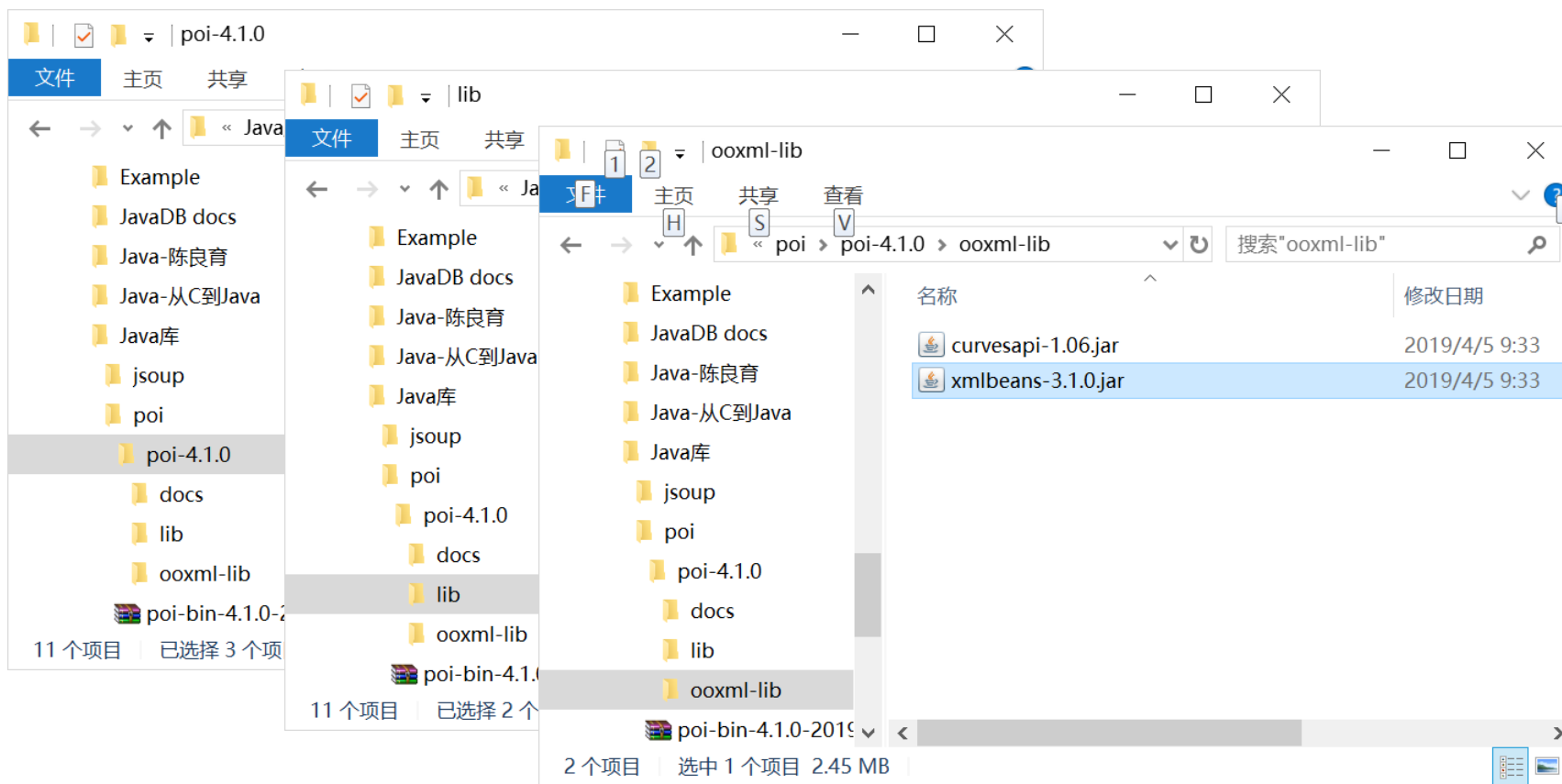
11.5 电子表格信息提取

- 第三方库：**POI**库
 - 主页：<https://poi.apache.org/>



11.5 电子表格信息提取

- 第三方库：**POI**库
 - 下载安装包：poi-bin-4.1.0-20190412.**zip**



11.5 电子表格信息提取

- 第三方库

例 11-6 使用 POI 库读取

```
1 import org.apache.poi.xssf.usermodel.XSSFWorkbook;
2 import org.apache.poi.xssf.usermodel.XSSFSheet;
3 import org.apache.poi.xssf.usermodel.XSSFRow;
4 import java.util.Iterator;
5 import java.io.*;
6
7 public class POI {
8     public static void main(String[] args) {
9         readExcel("example.xlsx");
10        readExcel("example.xls");
11    }
12 }
```

```
13 public static void readXLS(String filename) { // 读取 xls 电子表格文件
14     try {
15         InputStream xlsFile = new FileInputStream(filename); // xls 文件输入流对象
16         HSSFWorkbook wb = new HSSFWorkbook(xlsFile); // 从 xls 文件创建工作簿对象
17         HSSFSheet sheet = wb.getSheetAt(0); // 读取第一张工作表
18         HSSFRow row;
19         HSSFCell cell;
20         Iterator itRow = sheet.rowIterator(); // 取得工作簿的行迭代器
21         while ( itRow.hasNext() ) { // 遍历工作表的每一行
22             row = (HSSFRow) itRow.next(); // 读取下一行
23             Iterator itCell = row.cellIterator(); // 取得行的单元格迭代器
24             while ( itCell.hasNext() ) { // 遍历一行中的所有单元格
25                 cell = (HSSFCell) itCell.next(); // 读取下一个单元格
26                 if (cell.getCellType() == CellType.STRING) // 文本类型的单元格
27                     System.out.print(cell.getStringCellValue() + " ");
28                 else if (cell.getCellType() == CellType.NUMERIC) // 数值类型的单元格
29                     System.out.print(cell.getNumericCellValue() + " ");
30             }
31             System.out.println();
32         }
33         wb.close(); // 关闭工作簿
34     } catch (Exception e) { e.printStackTrace(); }
35 }
36 }
```

11.5 电子表格信息提取

- 第三方库

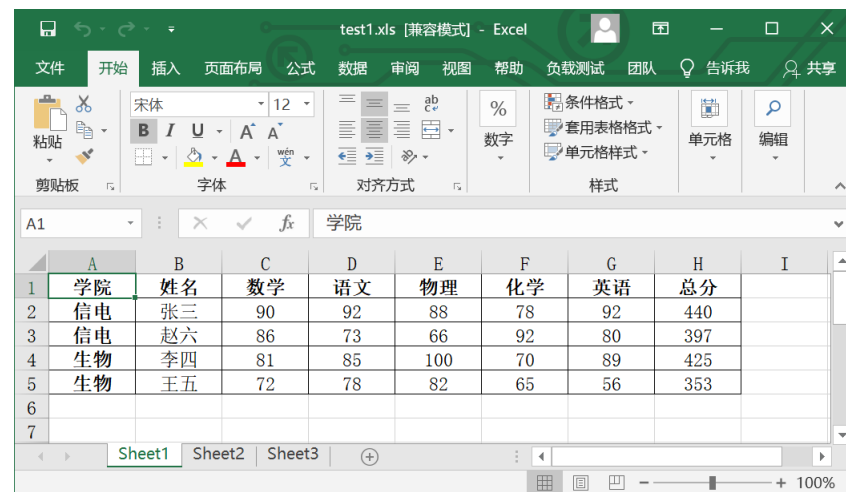
例 11-6 使用 POI 库读取

```
1 import org.apache.poi.xssf.usermodel.XSSFWorkbook;
2 import org.apache.poi.xssf.usermodel.XSSFSheet;
3 import org.apache.poi.xssf.usermodel.XSSFRow;
4 import java.util.Iterator;
5 import java.io.*;
6
7 public class POI {
8     public static void main(String[] args) {
9         readXLS("example.xls");
10        readXLSX("example.xlsx");
11    }
12 }
```

```
13 public static void readXLS(String filename) { // 读取 xls 电子表格文件
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37     public static void readXLSX(String filename) { // 读取xlsx 电子表格文件
38         try {
39             InputStream xlsxFile = new FileInputStream(filename); // xlsx 文件输入流对象
40             XSSFWorkbook wb = new XSSFWorkbook(xlsxFile); // 从xlsx 文件创建工作簿对象
41             XSSFSheet sheet = wb.getSheetAt(0); // 读取第一张工作表
42             XSSFRow row;
43             XSSFCell cell;
44             Iterator itRow = sheet.rowIterator(); // 取得工作簿的行迭代器
45             while ( itRow.hasNext() ) { // 遍历工作表的每一行
46                 row = (XSSFRow) itRow.next(); // 读取下一行
47                 Iterator itCell = row.cellIterator(); // 取得行的单元格迭代器
48                 while ( itCell.hasNext() ) { // 遍历一行中的所有单元格
49                     cell = (XSSFCell) itCell.next(); // 读取下一个单元格
50                     if (cell.getCellType() == CellType.STRING) // 文本类型的单元格
51                         System.out.print(cell.getStringCellValue() + " ");
52                     else if (cell.getCellType() == CellType.NUMERIC) // 数值类型的单元格
53                         System.out.print(cell.getNumericCellValue() + " ");
54                 }
55                 System.out.println();
56             }
57             wb.close(); // 关闭工作簿
58         } catch (Exception e) { e.printStackTrace(); }
59     } }
```

11.5 电子表格信息提取

- 第三方库：**POI**库



The screenshot shows an Excel spreadsheet titled 'test1.xls [兼容模式] - Excel'. The data is organized in a table with columns for '学院' (College), '姓名' (Name), '数学' (Math), '语文' (Chinese), '物理' (Physics), '化学' (Chemistry), '英语' (English), and '总分' (Total Score). The rows contain student information: Zhang San (信电), Zhao Li (信电), Li Si (生物), and Wang Wu (生物).

学院	姓名	数学	语文	物理	化学	英语	总分
信电	张三	90	92	88	78	92	440
信电	赵六	86	73	66	92	80	397
生物	李四	81	85	100	70	89	425
生物	王五	72	78	82	65	56	353

- 工作簿（*.xls, **workbook**）

- 工作表（**sheet**）

- 行（**row**）

- » 单元格（**cell**）

- POI库为它们分别定义了对应的Java类



中國農業大學

閻道宏

11.5 电子表格信息提取

- 第三方库：**POI**库

org.apache.poi.hssf.usermodel.HSSFWorkbook类说明文档			
public final class HSSFWorkbook extends POIDocument implements Workbook			
	修饰符	类成员（节选）	功能说明
1		HSSFWorkbook(java.io.InputStream s)	构造方法
2		int getNumberOfSheets()	返回工作表个数
3		HSSFSheet getSheetAt(int index)	按序号读取某个工作表
4		HSSFSheet getSheet(java.lang.String name)	按名称读取某个工作表
5		Iterator<Sheet> iterator()	获取工作表迭代器
6		void write(java.io.File newFile)	将工作簿保存到文件
.....			

org.apache.poi.hssf.usermodel.HSSFSheet类说明文档			
public final class HSSFSheet extends java.lang.Object implements Sheet			
	修饰符	类成员（节选）	功能说明
1		Iterator<Row> rowIterator()	获取行的迭代器
.....			



11.5 电子表格信息提取

- 第三方库：**POI**库

org.apache.poi.hssf.usermodel.HSSFRow类说明文档

public final class HSSFRow

extends java.lang.Object

implements Row, java.lang.Comparable<HSSFRow>

	修饰符	类成员（节选）	功能说明
1		Iterator<Cell> cellIterator()	获取单元格的迭代器
.....			

org.apache.poi.hssf.usermodel.HSSFCell类说明文档

public class HSSFCell

extends CellBase

	修饰符	类成员（节选）	功能说明
1		CellType getCellType()	获取单元格的数据类型
2		double getNumericCellValue()	读取数值类型的单元格
3		String getStringCellValue()	读取文本类型的单元格
.....			



11.5 电子表格信息提取

- 第三方库：**POI**库
 - 工作簿（*.xls, **workbook**）：HSSFWorkbook
 - 工作表（**sheet**）：HSSFSheet
 - 行（**row**）：HSSFRow
 - » 单元格（**cell**）：HSSFCell
 - 不同Excel版本的文件格式（*.xls、*.xlsx）
 - **HSSF**（Horrible Spreadsheet Format）：*.xls
import org.apache.poi.**hssf**.usermodel.*; // 导入xls相关的类
 - **XSSF**（XML Spreadsheet Format）：*.xlsx
import org.apache.poi.**xssf**.usermodel.*; // 导入xlsx相关的类



第11章 网页爬虫

- 本章学习要点
 - 网页：HTML
 - 访问网站：HTTP，GET、POST
 - 网页爬虫：URLConnection、HttpURLConnection
 - 提取网页信息：jsoup库
 - 提取电子表格信息：POI库
- 课程大作业

