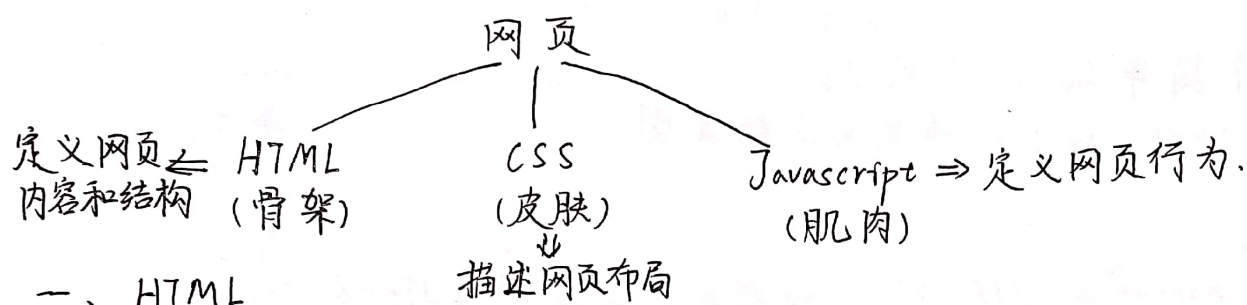


2.2 网页基础

§ 2.2.1 网页组成



一、HTML

1. 全称 Hyper Text Markup Language, 即超文本标记语言.

2. 网页的框架:

网页中包含许多元素(如文字、按钮、图片等), 不同类型的元素通过不同的标签来表示, 如图片为 `img`、视频为 `video`、段落用 `p`, 这些元素标签之间的布局又常通过 布局标签 `div` 嵌套组合而成, 各标签通过不同的排列和 ~~其~~ 嵌套才形成网页的框架.

二、CSS (Cascading Style Sheets)

1. 中文名为: 层叠样式表.

2. 作用: 当 HTML 中引用了多个样式文件, 并且样式发生冲突时, 浏览器能够依据 层叠顺序 处理.

! 它的作用类似于 PS 中的图层 优先显示.

3. CSS 是目前唯一的网页页面排版样式标准

4. 实际应用中, 一般会统一定义整个网页的样式规则, 写入一个 CSS 文件中(后缀为 `css`). 在 HTML 中, 只需要用 `link` 标签来引入即可.

三、JavaScript

1. 简称 JS, 是一种脚本语言.

2. 作用: 使网页实现一种实时、动态、交互的功能.

§ 2.2.2 网页结构

一个网页的一般结构是：html 标签内嵌套 head 和 body 标签，
head 内定义网页配置和引用，body 内定义网页正文。

举例：一个简单的 HTML 实例

<!DOCTYPE html> ⇒ 定义文档类型

<html>

<head>

head 标签表示网页头 { <meta charset = "UTF-8" > ⇒ 指定网页编码为 UTF-8

<title> This is a Demo </title> ⇒ 定义网页标题

</head>

body 标签表示网页体 { <body>

<div id = "container">

<div class = "wrapper">

<h2 class = "title"> Hello World </h2> ⇒ h2 标签表示一个二级标题

<p class = "text"> Hello, this is a paragraph. </p> ⇒ p 标签表示一个段落

</div>

</div>

</body>

</html>

} 每一个标签都要有相应的
的闭合。

§ 2.2.3 节点树及节点之间的关系

一、在 HTML 中，所有标签定义的内容都是节点，它们构成了一个 HTML DOM 树。

二、DOM (Document Object Model, 文档对象模型)

1. 它定义了访问 HTML 和 XML 文档的标准：

DOM 是中立于语言和平台的接口，它允许程序和脚本动态地访问和更新文档的内容、结构和样式。

2. W3C DOM 标准分为 3 个部分：

{ 核心 DOM：针对任何结构化文档的标准模型

XML DOM：针对 XML 文档的标准模型

HTML DOM：针对 HTML 文档

三、节点和节点树

1. 节点：根据 HTML DOM 的标准，HTML 文档中所有内容都是节点

{ 整个文档是一个文档节点

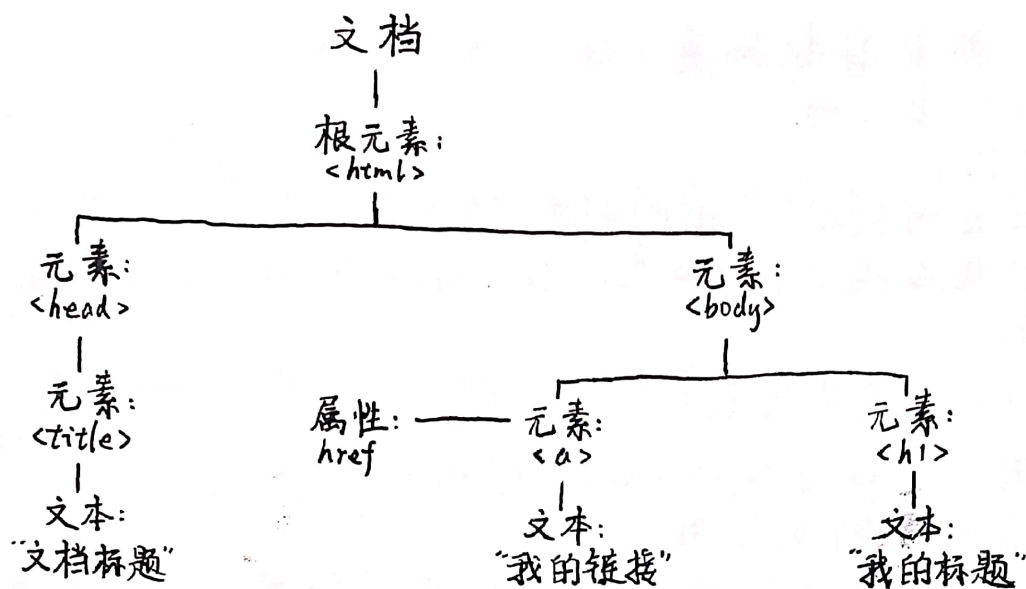
每个 HTML 元素是一个元素节点

HTML 元素内的文本是文本节点

每个 HTML 属性是一个属性节点

注释是注释节点

2. 节点树



- (1) 节点树中的所有节点均可通过 JavaScript 访问, 所有节点元素均可以被修改、创建或删除。
- (2) 节点之间存在层级关系, 如: 父 (parent)、子 (child) 和兄弟 (sibling) 等, 父节点拥有子节点, 同级子节点称为兄弟节点。
- (3) 顶端节点称为根节点 (root)

§ 2.2.4 选择器

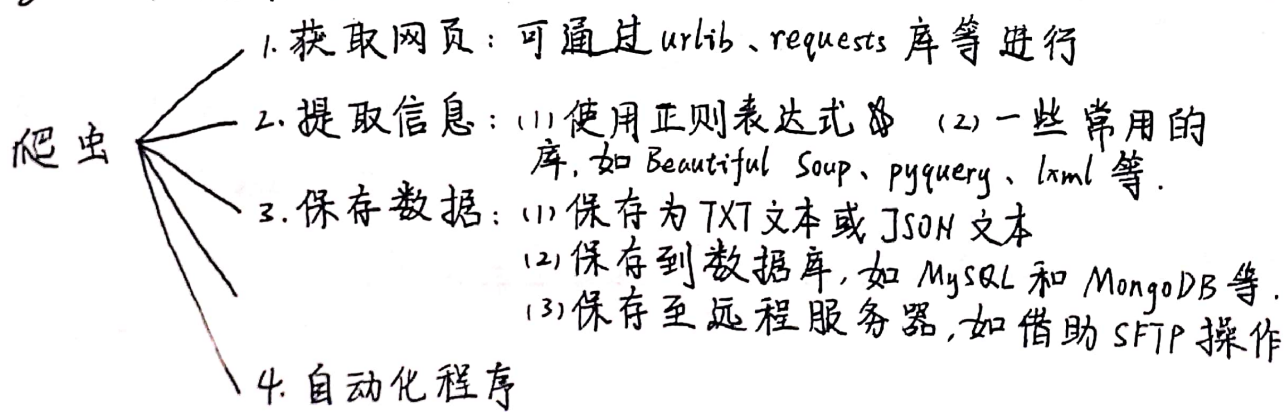
1. 在 CSS 中, 我们用 CSS 选择器来定位节点
2. CSS 选择器的语法规则:
* 详见书 P91

常见的语法规则:

- (1) 以 “#” 开头加上 id, 表示选择该 id 所代表的节点
- (2) 以 “.” 开头加上 class, 表示选择该属性 class 所代表的节点
- (3) 选择器间加空格表示嵌套, 不加空格则表示并列关系。

2.3 爬虫的基本原理

§ 2.3.1 爬虫概述



§ 2.3.2 爬虫能抓取怎样的数据.

爬虫可以抓取的数据有:

1. 网页 HTML 源代码
2. JSON 字符串
3. 各种二进制数据 (如图片、视频、音频等)
4. 各种扩展名的文件, 如 CSS、JavaScript 和配置文件等.

总之, 只要是对应各自的 url 并基于 HTTP 或 HTTPS 协议的, 都可以爬取.

§ 2.3.3 JavaScript 渲染页面

我们有时用 urllib 或 requests 抓取网页时, 会发现得到的源码与浏览器中看到的不同

1. 原因: 现在的网页大都通过 JavaScript 进行渲染, 而我们爬取到的只是 HTML 文件, 而不会加载其中的 JS 文件

2. 解决方法: ① 分析其后台 Ajax 接口

② 用 Selenium、Splash 这样的库进行模拟 JavaScript 渲染.

2.4 会话和 Cookies.

§ 2.4.1 静态网页和动态网页

一、静态网页: 内容由 HTML 代码编写, 图片、文字等内容都通过写好的代码指定

└─ 优点: 加载速度快, 编写简单

└─ 缺点: 可维护性差, 不能根据 URL 灵活地显示内容.

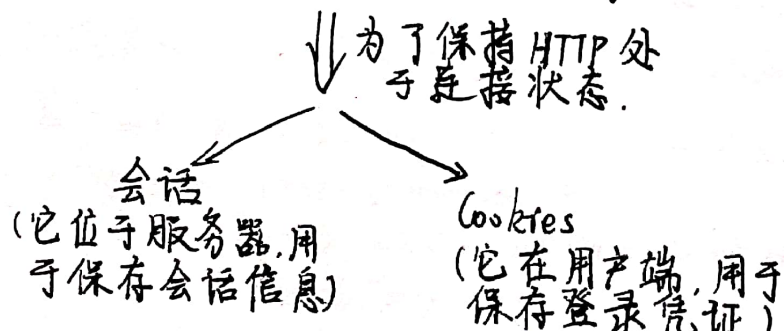
二、动态网页: 可以动态解析 URL 参数的变化, 关联数据库并呈现不同的页面内容.

! 它可能由 JSP、PHP、Python 等其他语言编写.

§ 2.4.2 无状态 HTTP

! 什么是无状态 HTTP?

HTTP 的无状态是指 HTTP 协议对事务处理是没有记忆能力的. 即服务器的作用过程是完全独立的.



一、会话

在 Web 中, 会话对象用来存储特定用户会话所需的属性及配置信息,

作用: 当用户在应用程序的 Web 页之间跳转时, 存储在会话对象中的变量将不会丢失。

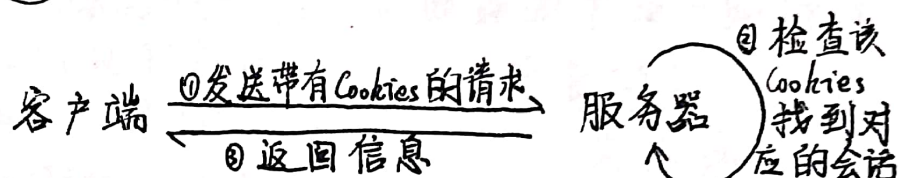
二、Cookies

1. 什么是 Cookies: 网站为了辨别用户身份、进行会话跟踪而存储在用户本地终端上的数据。

简单来说: Cookies 就是一种用户的身份凭证。

2. 会话维持:

第一次请求: 

第二次请求: 

总结: 会话的维持需要 Cookies 和会话, 一个在客户端, 一个在服务器, 二者相互协作, 相互配合。

3. Cookies 的属性结构。

开发者工具 → Application → Storage → Cookies

Cookies 的属性结构有如下几种: (详见 P₃₈)

- (1) Name: Cookie 的名称, 一旦创建, 不可修改
- (2) Value: Cookie 的值
- (3) Domain: 设置可以访问 Cookie 的域名
- (4) Max Age: 该 Cookie 失效的时间, 注意: 若其为负则表示一旦关闭浏览器, Cookie 就会失效, 且不会保存。
- (5) Path: 设置可以访问 Cookies 的路径。
- (6) Size: 该 Cookie 的大小。
- (7) HTTP 字段: Cookie 的 httponly ~~字段~~ 属性。
- (8) Secure: 该 Cookie 是否仅被使用安全协议运输, 默认为 False。

4. 会话 Cookie 和持久 Cookie

{ 会话 Cookie, 一旦关闭浏览器就失效

{ 持久 Cookie, 关闭浏览器后则储存在硬盘中持续一段时间才失效。

二者无本质区别, 只是由于 Max Age 不同而已。

2.5 代理的基本原理

2.5.1 基本原理

一、代理：即代理服务器

二、代理的作用：“中转站”的作用

具体来说：代理就是代理网络用户去获得网络信息

用户 $\xrightarrow[\text{响应}]{\text{请求}}$ 服务器

用户 $\xrightarrow[\text{响应}]{\text{请求}}$ 代理服务器 $\xrightarrow[\text{响应}]{\text{请求}}$ 服务器

由此即可实现IP伪装的效果

2.5.2 代理的具体作用

1. 突破自身IP访问限制，访问一些平时不能访问的站点。
2. 访问一些单位或团体的内部资源。
3. 提高访问速度
4. 隐藏真实IP

2.5.3 爬虫代理

有些服务器会有反爬虫设置，我们可以利用代理来隐藏真实IP，让服务器误以为是代理服务器在请求自己，从而突破封锁。

2.5.4 代理分类

一、根据代理的协议划分

FTP代理服务器：主要用于访问FTP服务器，端口一般为21、2121等

HTTP代理服务器：主要用于访问网页，端口一般为80、8080、3128等。

SSL/TLS代理：主要用于访问加密网站，端口一般为443

RTSP代理：主要用于访问Real流媒体服务器，端口554

Telnet代理：主要用于telnet远程控制，端口：23。

POP3/SMTP代理：主要用于POP3/SMTP收发邮件，端口：110/25。

SOCKS代理：只单纯传递数据包，不关心具体协议和用法，端口：108

二、根据代理的匿名程度划分

高度匿名代理：

普通匿名代理：代理服务器通常会加入的协议头有HTTP_VIA和

透明代理

间谍代理

HTTP_X_FORWARDED_FOR