DAY29 cookie

2023年5月26日 16:39

HTTP概述

一.通信协议概述

通信协议是指双方实体完成通信或服务所必须遵循的规则和约定。

可以简单地理解为各计算机之间进行相互会话所使用的共同语言。

两台计算机在进行通信时，必须使用的通信协议。

通信协议：http　https　单向通信，浏览器(客户端)发起的。

二.HTTP协议

超文本传输协议是互联网上应用最为广泛的一种网络协议，是一个客户端和服务器端请求和应答的标准，所有的WWW都必须遵守这个标准(http默认端口：80)

三.HTTPS 协议

HTTPS 协议，是以安全为目标的 HTTP 通道，在HTTP的基础上通过传输加密和身份认证保证了传输过程的安全性 。HTTPS 在HTTP 的基础下加入SSL 层，HTTPS 的安全基础是 SSL，因此加密的详细内容就需要SSL(https默认端口：443)

四.HTTP的请求过程

比如访问京东商城 - https://www.jd.com(理解为访问京东商城源码) - 获取信息

理解为访问京东商城源码 - 存放在服务器(昂贵的电脑) - 电脑唯一标识IP地址

第一步：输入https://www.jd.com(域名) - 域名解析- 将域名和服务器的ip地址绑定到一起

第二步：通过域名获取ip地址，ip地址找到对应的服务器

第三步：通过端口找对应的程序

第四步：访问服务器里面的程序

第五步：解析

第六步：获取信息

五.三次握手，四次挥手

第1次握手：客户端发送数据包给服务端，请求连接

第2次握手：服务端接收客户端数据包成功后，回传一个带有标志的数据包传递确认信息，表示我服务端收到了

第3次握手：客户端再回传一个数据包，表示我知道了，握手结束。

第1次挥手：客户端发送断开连接请求的报文，请求和服务器端断开

第2次挥手：服务端收到断开请求后，发送一个标识给客户端

第3次挥手：服务端确认发送已经完成，通知客户端

第4次挥手：客户端收到服务端的完成通知后，服务端进入关闭状态，客户端也关闭，完成四次挥手。

cookie概述

一.cookie的概述

cookie是由Web服务器保存在用户浏览器上的小文本文件，可以包含相关用户的信息。

注意：服务器保存，vscode下面的live Server插件就是前端的静态服务器

二.cookie的特点

1.禁用Cookie后，无法正常注册登陆。

2.cookie是与浏览器相关的，不同浏览器之间所保存的cookie也是不能互相访问的；

3.cookie可以被删除。因为每个cookie都是硬盘上的一个文件；

4.cookie安全性不够高 - xss攻击

三.cookie的使用及封装

1.cookie以字符串的形式进行存储的。name值相同，存储时就会覆盖。

2.cookie名称和值可以自己定义,存储的是字符串。不能超过4KB

3.如果cookie不存在，输出空白.

4.封装三个方法实现cookie的存储，获取，删除

cookie的封装

一.cookie的存储

1.cookie的存储

```
document.cookie = 'usename=zhangsan';
```

2.cookie的存储+过期时间expires(天) - 利用日期对象

```
let d = new Date();
d.setDate(d.getDate() + 3);
document.cookie = `name=zhaoliu;expires=${d}`
```

3.路径

不同目录里面的cookie不能互相访问

例如：AB具有相同的目录层级，A目录里面存储cookie，B目录无法访问

存储的cookie的时候，需要配置路径path=/ 将cookie存在根目录下面

```
const setCookie = function (name, value, days) {//name：存储的名词  value：名词对应的值   days：存储多少天
    let d = new Date();
    d.setDate(d.getDate() + days);
    document.cookie = `${name}=${value};expires=${d};path=/`;
}

// setCookie('age', 18, 5);
// setCookie('sex', '男', 5);
```

二.封装函数实现获取cookie

```
// console.log(document.cookie);//name=zhaoliu; age=18; sex=男
const getCookie = function (name) {
  let arr = document.cookie.split('; ');//['name=zhaoliu', 'age=18', 'sex=男']
  for (let item of arr) {
    let newArr = item.split('=');//['name','zhangsan']['age',18]['sex','男']
    if (name === newArr[0]) {
      return newArr[1];
    }
  }
}

// console.log(getCookie('name'));//zhaoliu
// console.log(getCookie('age'));//18
// console.log(getCookie('sex'));//男
```

三.封装函数实现删除

1.如果name值相同，覆盖

2.如果过期时间是一个过去时间，肯定消失

```
const removeCooke = function (name) {
  setCookie(name, '', -1);
}
// removeCooke('sex')
```

整合方法

```
const cookie = {
  set(name, value, days) {
    let d = new Date();
    d.setDate(d.getDate() + days);
    document.cookie = `${name}=${value};expires=${d};path=/`;
  },
  get(name) {
    let arr = document.cookie.split("; ");
    for (let item of arr) {
      let newArr = item.split("=");
      if (name === newArr[0]) {
        return newArr[1];
      }
    }
  },
  remove(name) {
    this.set(name, "", -1);
  },
};
```

cookie属性和对比

一.封装函数的整理

将封装的函数进行优化整理,将多个函数打包到对象中，以对象的身份进行调用。

```
cookie.set('username', '尼古拉斯赵四', 10);
```

二.Cookie的属性和Application面板介绍

通过控制面板--application--左侧cookie查看浏览器存储的cookie

name: 自定义的键值

value: 键对应的值

domain: 主机(主域名)

path: 路径

expires: 过期时间

size: cookie的大小， 不能超过4k

httpOnly: 跟后端关系比较密切，如果后端设置httpOnly, js代码无法获取，参数为布尔值。

secure: 安全，跟http / https设置有关。

samesite: 防止CSRF(攻击者盗用了你的身份且以你的名义发送恶意请求)攻击和用户追踪

```
const obj = {
  a: 1,
  b: 3,
  c: 5
}
cookie.set('data', JSON.stringify(obj), 10)
```

三.cookie,localStorage,sessionStorage 的区别

1.cookie服务器端存储，借助本地服务器，localStorage, sessionStorage客户端直接存储

2.cookie仅支持4kb，但是localStorage, sessionStorage支持5mb大小

3.cookie可以设置过期时间，localStorage是永久的，sessionStorage是临时的(会话结束).

4.cookie安全性不及本地存储，了解xss攻击。

XSS(Cross Site Script)攻击全称跨站脚本攻击

通过巧妙的方法注入恶意指令代码到网页，使用户加载并执行攻击者恶意制造的网页程序。

这些恶意网页程序通常是JavaScript。

1.实施XSS攻击需要具备两个条件：

需要向web页面注入恶意代码；

这些恶意代码能够被浏览器成功的执行。

2.总结：输入过滤、输出转义