Nestjs 中的 Cookie

主讲教师：（大地）

合作网站：www.itying.com （IT 营）

我的专栏：https://www.itying.com/category-79-b0.html

一、Cookie 简介

● HTTP 是无状态协议。简单地说，当你浏览了一个页面，然后转到同一个网站的另一个页面，服务器无法认识到这是同一个浏览器在访问同一个网站。每一次的访问，都是没有任何关系的。如果我们要实现多个页面之间共享数据的话我们就可以使用 Cookie 或者 Session 实现

● cookie 是存储于访问者的计算机中的变量。可以让我们用同一个浏览器访问同一个域名的时候共享数据。

二、Cookie 特点

- cookie 保存在浏览器本地
- 正常设置的 cookie 是不加密的，用户可以自由看到;
- 用户可以删除 cookie，或者禁用它
- cookie 可以被篡改
- cookie 可以用于攻击
- cookie 存储量很小。未来实际上要被 localStorage 替代，但是后者 IE9 兼容。

三、Nestjs 中使用 Cookie

NestJs 中使用 Cookie 的话我们可以用 cookie-parser 来实现。

https://docs.nestjs.com/techniques/cookies#cookies

1.安装

```
cnpm instlal cookie-parser --save
npm i -D @types/cookie-parser
```

2. 在 main.ts 中引入 cookie-parser

```
import * as cookieParser from 'cookie-parser'
```

3. 在 main.ts 配置中间件

```
app.use(cookieParser());
```

4.设置 cookie

```
res.cookie("name",'zhangsan',{maxAge: 900000, httpOnly: true});

//HttpOnly 默认 false 不允许 客户端脚本访问
```

5、**获取 Cookies**

```
@Get('getCookies')
getCookies(@Request() req){
    return req.cookies.name;
}
```

四、Nestjs 中 Cookie 中的一些参数

Property	Type	Description
domain	String	Domain name for the cookie. Defaults to the domain name of the app.
expires	Date	Expiry date of the cookie in GMT. If not specified or set to 0, creates a session cookie.
httpOnly	Boolean	Flags the cookie to be accessible only by the web server.
maxAge	String	Convenient option for setting the expiry time relative to the current time in milliseconds.
path	String	Path for the cookie. Defaults to "/".
secure	Boolean	Marks the cookie to be used with HTTPS only.
signed	Boolean	Indicates if the cookie should be signed.

属性说明：

domain：域名

expires：过期时间（秒），在设置的某个时间点后该 Cookie 就会失效，如 expires=Wednesday, 09-Nov-99 23:12:40 GMT

maxAge：最大失效时间（毫秒），设置在多少后失效

secure：当 secure 值为 true 时，cookie 在 HTTP 中是无效，在 HTTPS 中才有效

path：表示 cookie 影响到的路，如 path=/。如果路径不能匹配时，浏览器则不发送这个 Cookie

httpOnly：是微软对 COOKIE 做的扩展。如果在 COOKIE 中设置了“httpOnly”属性，则通过程序（JS 脚本、applet 等）将无法读取到 COOKIE 信息，防止 XSS 攻击产生

signed：表示是否签名 cookie，设为 true 会对这个 cookie 签名，这样就需要用 res.signedCookies 而不是 res.cookies 访问它。被篡改的签名 cookie 会被服务器拒绝，并且 cookie 值会重置为它的原始值

设置 cookie

```
res.cookie('rememberme', '1', { maxAge: 900000, httpOnly: true })

res.cookie('name', 'tobi', { domain: '.example.com', path: '/admin', secure: true });

res.cookie('rememberme', '1', { expires: new Date(Date.now() + 900000), httpOnly:
true });
```

获取 cookie

```
req.cookies.name
```

删除 cookie

```
res.cookie('rememberme', '', { expires: new Date(0)});

res.cookie('username','zhangsan',{domain:'.ccc.com',maxAge:0,httpOnly:true});
```

四、Nestjs 中 Cookie 加密

1.配置中间件的时候需要传参

```
app.use(cookieParser('123456'));
```

2.设置 cookie 的时候配置 signed 属性

```
res.cookie('userinfo','hahaha',{domain:'.ccc.com',maxAge:900000,httpOnly:true,signed:true});
```

3. signedCookies 调用设置的 cookie

```
console.log(req.signedCookies);
```