会话控制

HTTP协议是一个无状态的协议，它无法区分多次请求是否发送自同一客户端。而我们在实际的使用中，却有大量的这种需求，我们需要通过会话的控制来解决该问题。

Cookie

1.是什么？
本质就是一个【字符串】，里面包含着浏览器和服务器沟通的信息（交互时产生的信息）。
存储的形式以：【key-value】的形式存储。
浏览器会自动携带该网站的cookie，只要是该网站下的cookie，全部携带。
2.分类：
--会话cookie（关闭浏览器后，会话cookie会自动消失，会话cookie存储在浏览器运行的那块【内存】上）。
--持久化cookie：（看过期时间，一旦到了过期时间，自动销毁，存储在用户的硬盘上,备注：如果没有到过期时间，同时用户清理了浏览器的缓存，持久化cookie也会消失）。

3.工作原理：
--当浏览器第一次请求服务器的时候，服务器可能返回一个或多个cookie给浏览器
--浏览器判断cookie种类
--会话cookie：存储在浏览器运行的那块内存上
--持久化cookie：存储在用户的硬盘上
--以后请求该网站的时候，自动携带上该网站的所有cookie（无法进行干预）
--服务器拿到之前自己“种”下cookie，分析里面的内容，校验cookie的合法性，根据cookie里保存的内容，进行具体的业务逻辑。

4.应用：
解决http无状态的问题（例子：7天免登录，一般来说不会单独使用cookie，一般配合后台的session存储使用）

5.不同的语言、不同的后端架构cookie的具体语法是不一样的，但是cookie原理和工作过程是不变的。
备注：cookie不一定只由服务器生成，前端同样可以生成cookie，但是前端生成的cookie几乎没有意义。

6.对比浏览器的本地存储：
1.localStorage：
(1).保存的数据，只要用户不清除，一直存在
(2).作为一个中转人，实现跨页签通信。
(3).保存数据的大小：5MB - 10MB
2.sessionStorage：
(1).保存的数据，关闭浏览器就消失
(3).保存数据的大小：5MB - 10MB
3.cookie：
(1).分类：会话cookie----关浏览器消失； 持久化cookie----到过期时间消失
(2).保存数据的大小:4K --- 8K
(3).主要用于解决http无状态(一般配合后端的session会话存储使用)
(4).浏览器请求服务器时，会自动携带该网站的所有cookie

Session

1.是什么？
标准来说，session这个单词指的是会话。
(1).前端通过浏览器去查看cookie的时候，会发现有些cookie的过期时间是：session，意味着该cookie是会话cookie。
(2).后端人员常常把【session会话存储】简称为：session存储，或者更简单的称为：session

2.特点：
1.存在于服务端
2.存储的是浏览器和服务器之间沟通产生的一些信息

3.默认session的存储在服务器的内存中，每当一个新客户端发来请求，服务器都会新开辟出一块空间，供session会话存储使用。

4.工作流程：
--第一次浏览器请求服务器的时候，服务器会开辟出一块内存空间，供session会话存储使用。
--返回响应的时候，会自动返回一个cookie（有时候会返回多个，为了安全），cookie里包含着，上一步产生会话存储“容器”的编号 （id）
--以后请求的时候，会自动携带这个cookie，给服务器。
--服务器从该cookie中拿到对应的session的id，去服务器中匹配。
--服务器会根据匹配信息，决定下一步逻辑

5.备注： 1.一般来说cookie一定会配合session使用。
2.服务端一般会做session的持久化，将session保存在数据库中，防止由于服务器重启，造成session的丢失。
3.session什么时候销毁？
(1).服务器没有做session的持久化的同时，服务器重启了。
(2).给客户端种下的那个用于保存session编号的cookie销毁了，随之服务器保存的session销毁(不管是否做了session的 持久化)。
(3).用户主动在网页上点击了“注销” “退出登录”等等按钮。

cookie和session的区别

1) 存在的位置：
cookie 存在于客户端，临时文件夹中；
session 存在于服务器的内存中，一个session域对象为一个用户浏览器服务
2) 安全性：
cookie是以明文的方式存放在客户端的，安全性低，可以通过一个加密算法进行加密后存放；
session存放于服务器的内存中，所以安全性好
3) 网络传输量：
cookie会传递消息给服务器；
session本身存放于服务器，但是通过cookie传递id，会有少量的传送流量
4) 生命周期(以20分钟为例)：
cookie的生命周期是累计的，从创建时，就开始计时，20分钟后，cookie生命周期结束；
session的生命周期是间隔的，从创建时，开始计时，如在20分钟，没有访问session，那么session生命周期被销毁；但是，如果在20分 钟内（如在第19分钟时）访问过session，那么，将重新计算session的生命周期；关机会造成session生命周期的结束，但是对cookie没 有影响。
5) 访问范围：
session为一个用户浏览器独享；
cookie为多个用户浏览器共享
6) 大小：
cookie 保存的数据不能超过4K，很多浏览器都限制一个站点最多保存50个cookie；

`session` 保存数据理论上没有任何限制（内存有多大就能有多大）

数据加密的方法

MD5、sha1

使用：

1）下载对应的包 npm i md5

2）引入 const md5 = require('md5')

3）使用 md5(value)