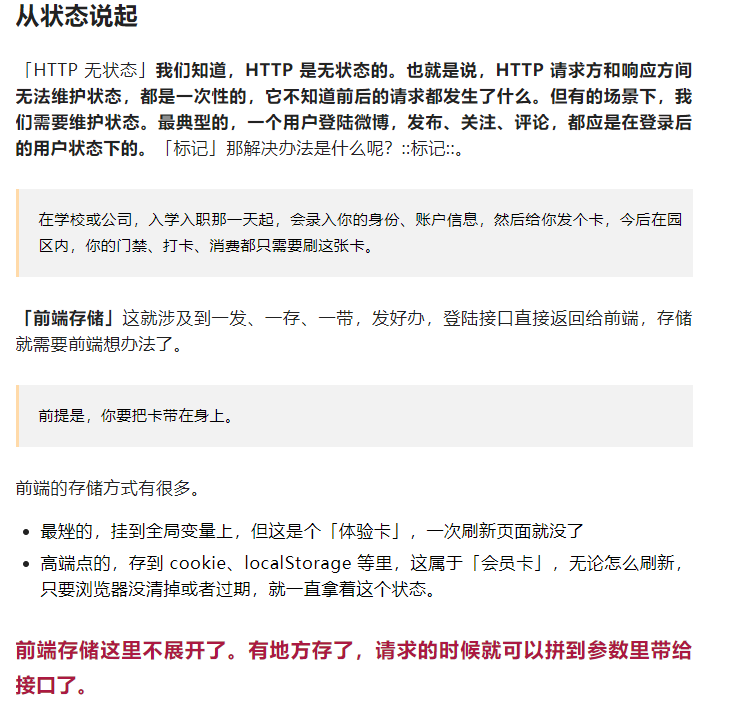
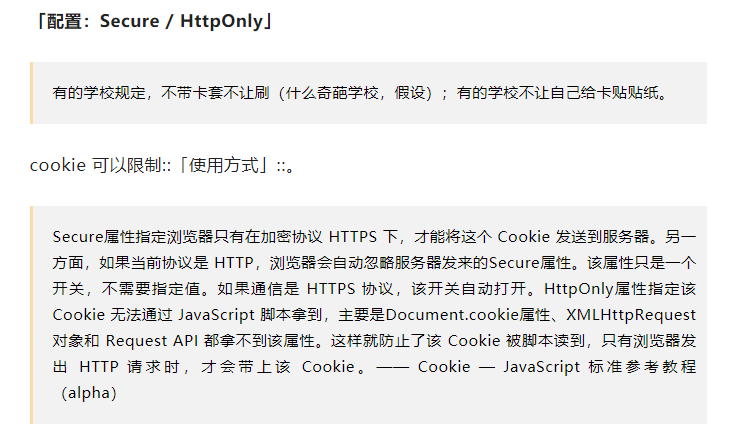
# Token



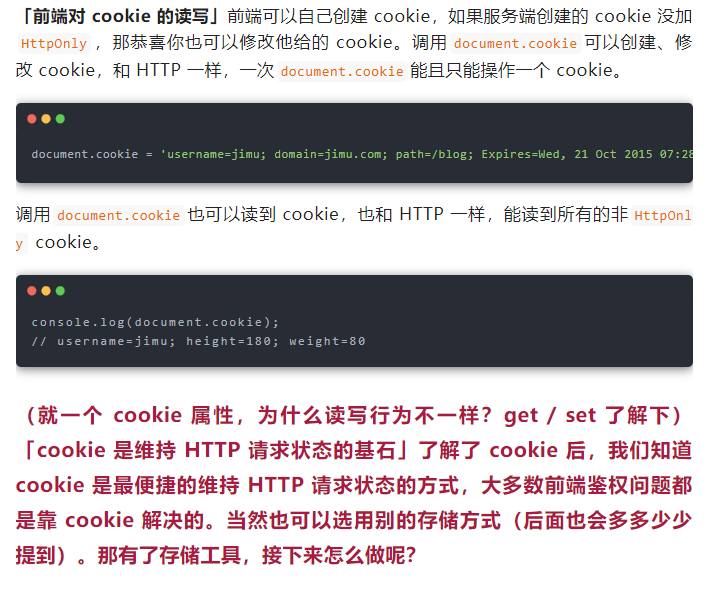


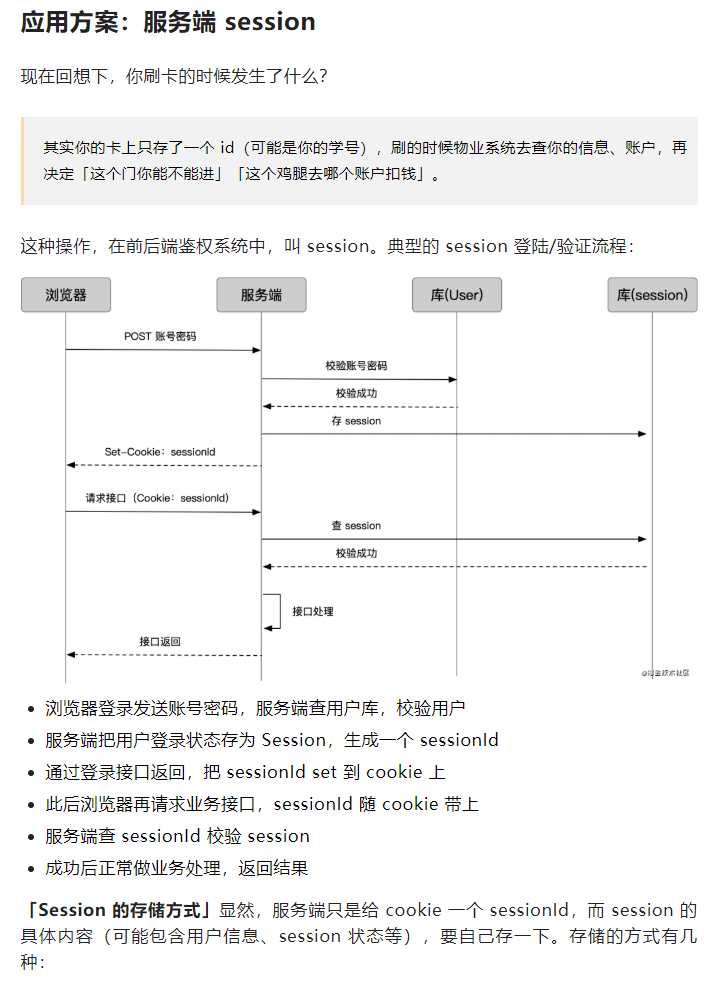






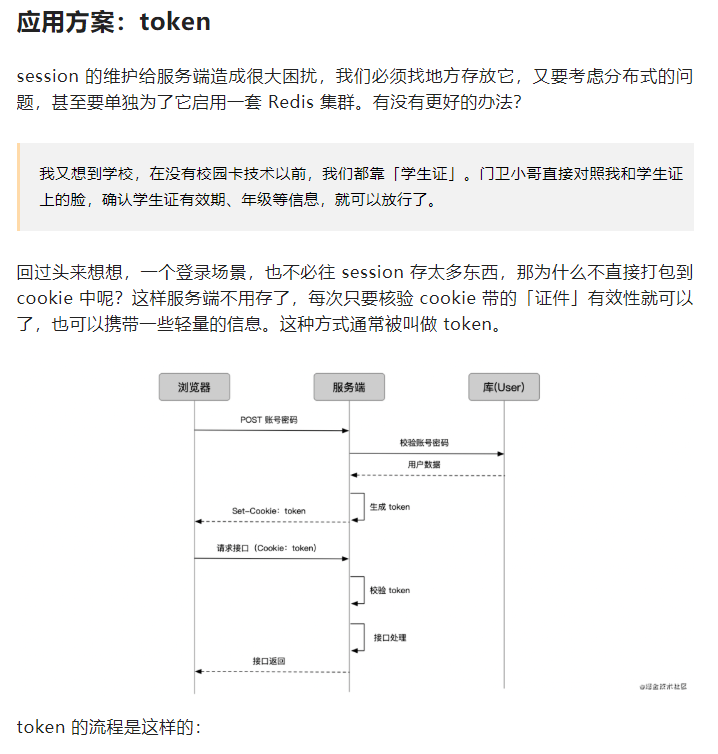






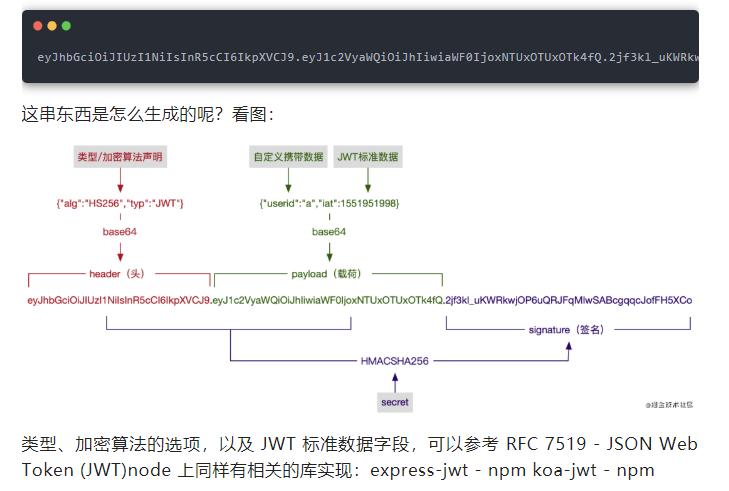


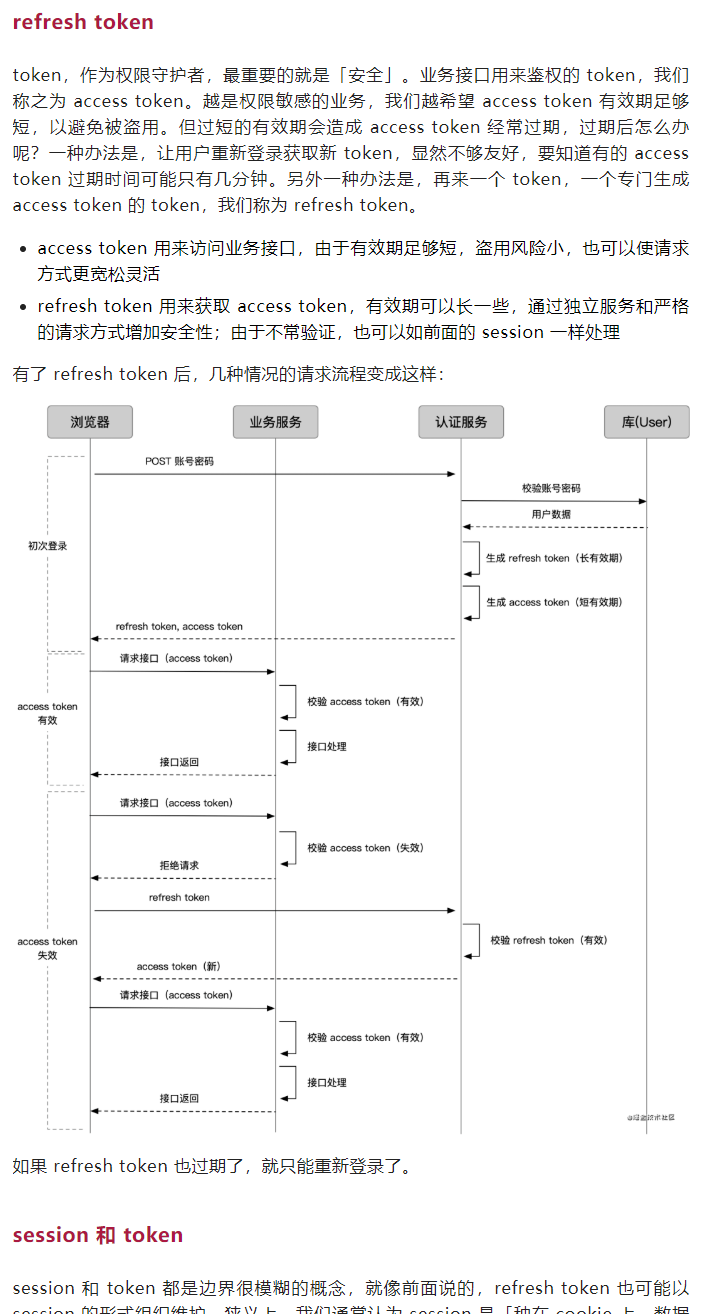




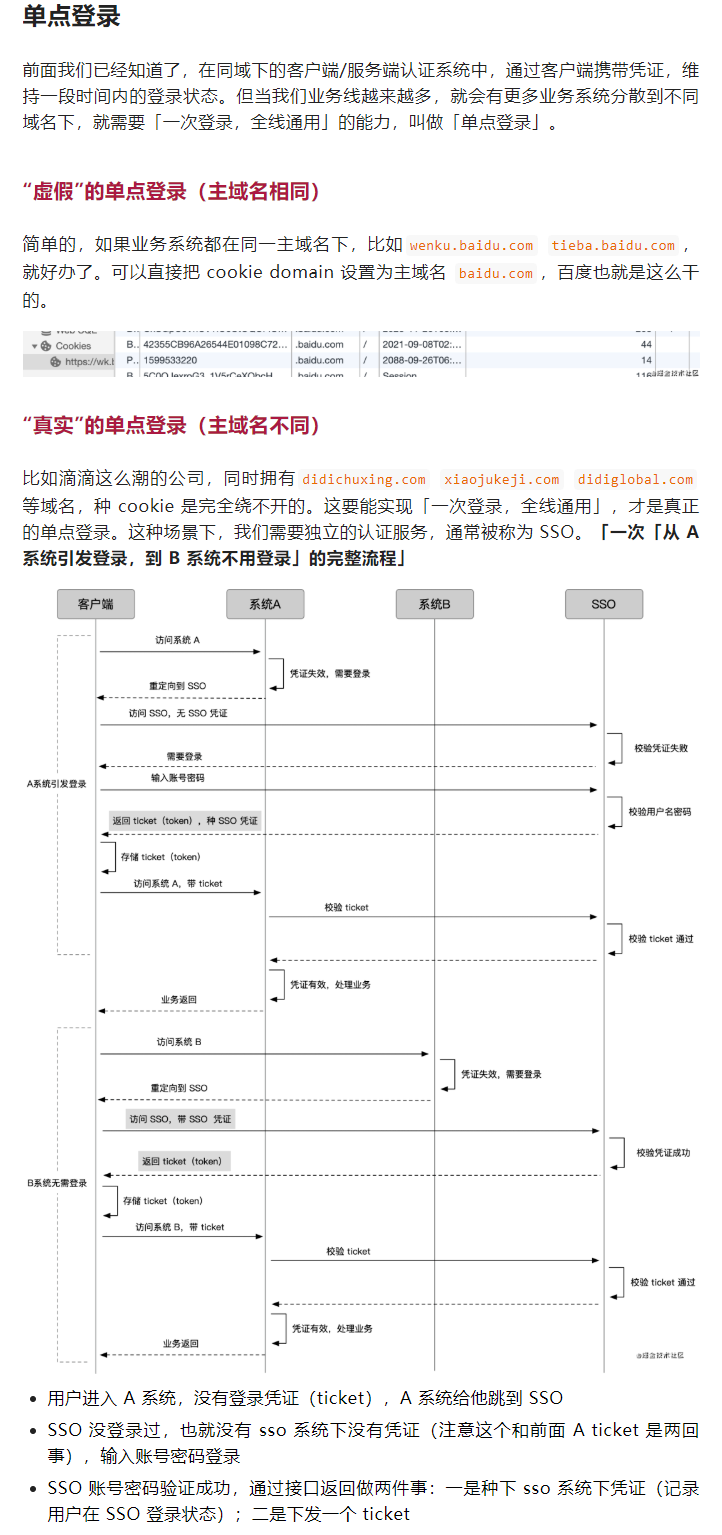


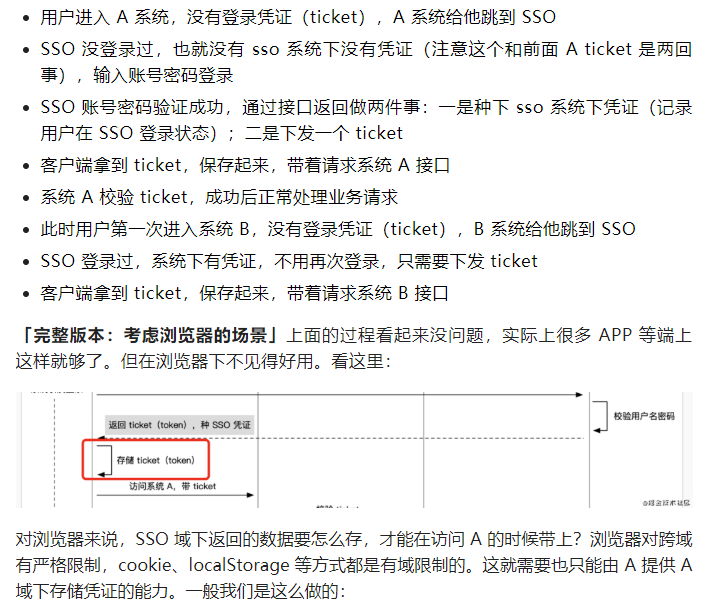


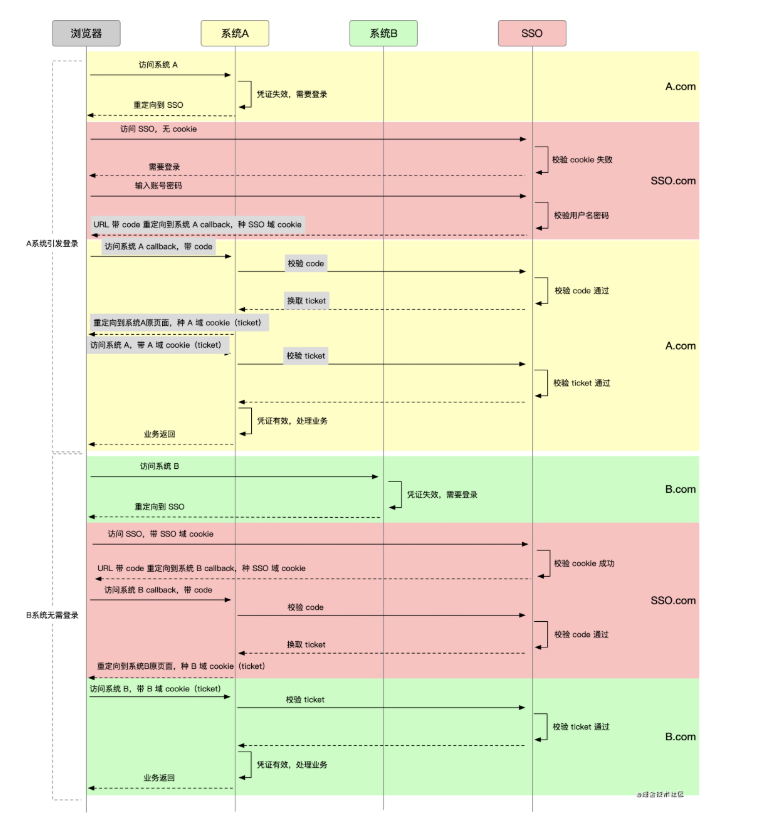


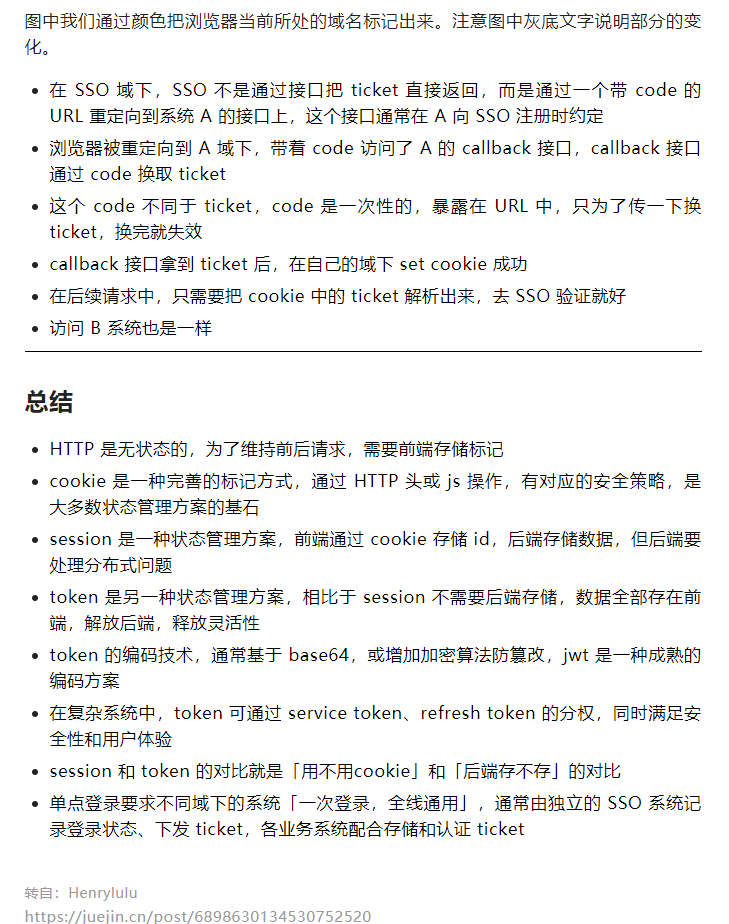












# Cookie、session

## 为什么有Cookie？

cookie是为了辨别用户身份的。我们知道HTTP本身是无状态的协议，服务端不会记得是谁向它发来的请求。但在某些情况下我们需要记住用户在未登录的状态下浏览了什么，比如淘宝。这时候就需要借助我们的Cookie了。 客户端请求服务器后，如果服务器需要记录用户状态，服务器会在响应信息中包含一个Set-Cookie的响应头，客户端会根据这个响应头存储Cookie信息。再次请求服务器时，客户端会在请求信息中包含一个Cookie请求头，而服务器会根据这个请求头进行用户身份、状态等较验。

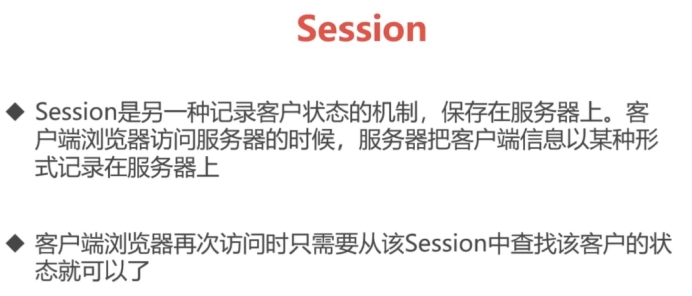
## Cookie的弊端

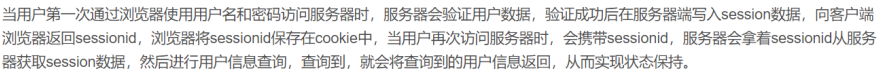
Cookie 虽然很方便，但是使用 Cookie 有一个很大的弊端，Cookie 中的所有数据在客户端就可以被修改，数据非常容易被伪造，那么一些重要的数据就不能存放在 Cookie 中了，而且如果 Cookie 中数据字段太多会影响传输效率。为了解决这些问题，就产生了 Session，Session 中的数据是保留在服务器端的。

## 如果客户端禁用cookie怎么办?

一般这种情况下，会使用一种叫做url重写的技术来进行会话跟踪，即每次http交互，url后面都会被附加上一个诸如sid=\*\*的参数，服务端据此来识别用户。

## session







## cookie和session的对比

1.存放的位置不同：cookie数据存放在客户的浏览器上，session数据放在服务器上(单个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20个cookie)。

2.安全性不同：cookie不是很安全，别人可以分析，复制，甚至是修改存放在本地的COOKIE，考虑到安全应当使用session，因为session存放在服务器上，不会产生敏感信息泄露的风险。如果要使用cookie，那么密码之类的敏感信息要加密。

3.有效期不同：通过设置较大的过期时间，cookie可以在浏览器端保存很久，而服务器端会定期清理超时的sessionid，避免出现过大的压力，另外，session依赖与名为sessionid的cookie，而cookie的sessionid的过期时间默认设为-1，所以只要关闭浏览器结束会话，那么这次的seesion就失效了。

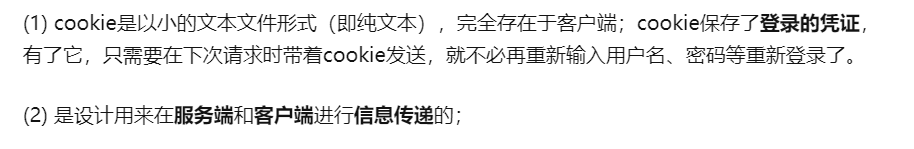
4.对服务器的压力不同：由于session是储存在服务器端的所以当很多用户访问时服务器会有较大压力，而cookie是保存在客户端的，所以对服务器端造成的压力较小。

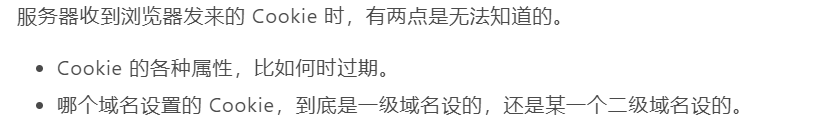
5.跨域支持上不同：Cookie可以实现跨域名访问，而Session则不会支持跨域名访问，Session仅在他所在的域名内有效。

## session弊端：

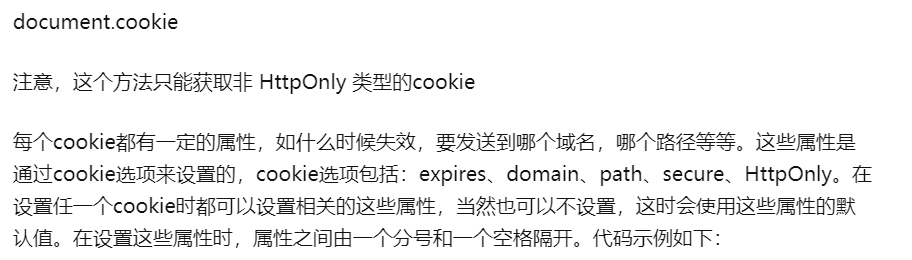
token可以解决session第三个弊端

# cookie





## Document.cookie



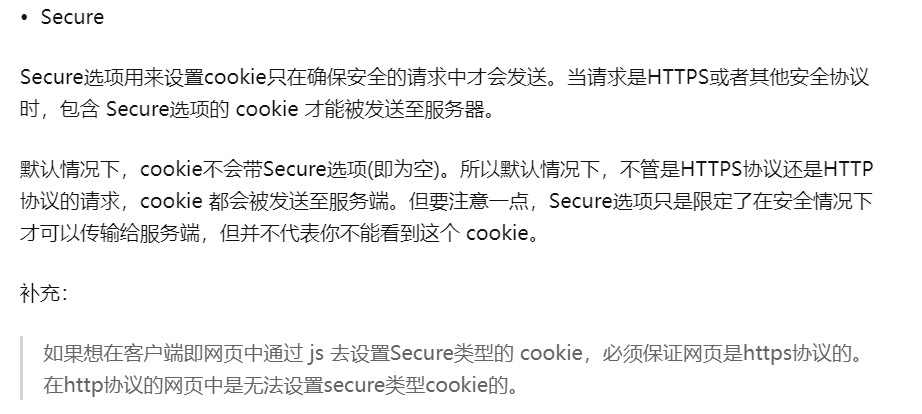
## Expire、Maxage



## Domain、path

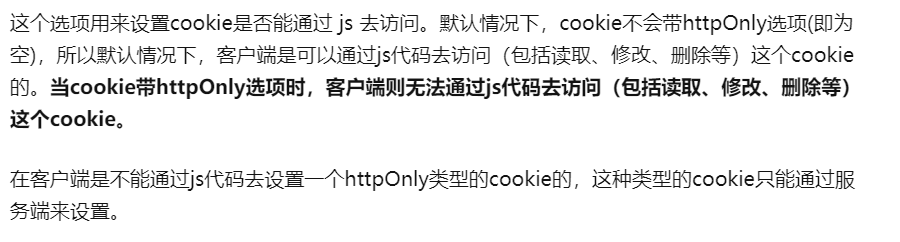


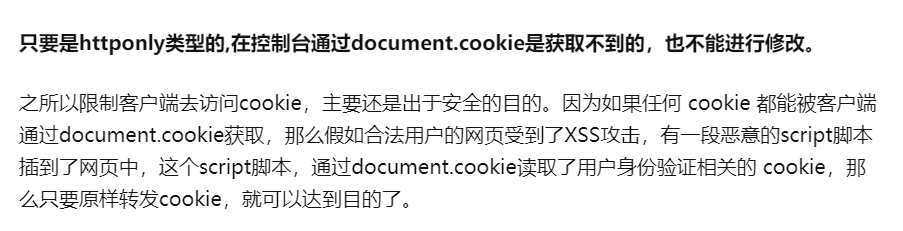
## Secure





## Httponly





## Samesite



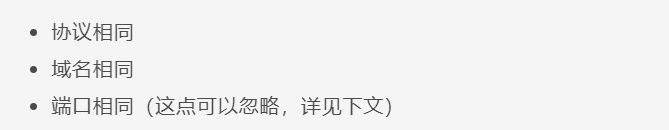


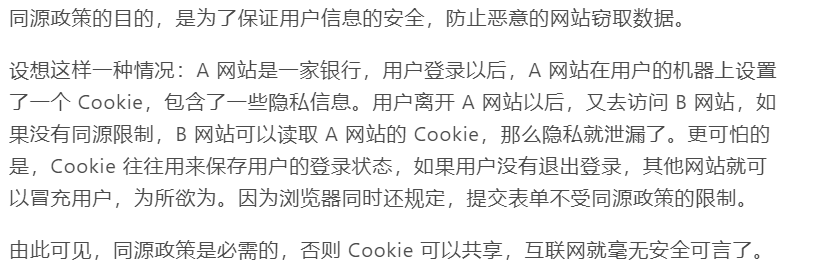




# 同源策略

## 含义与目的





## 限制范围



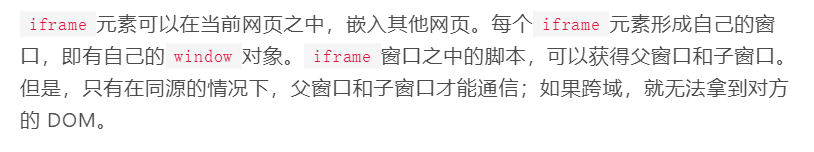


## Cookie





## iframe 和多窗口通信





### 片段识别符





### window.postMessage()

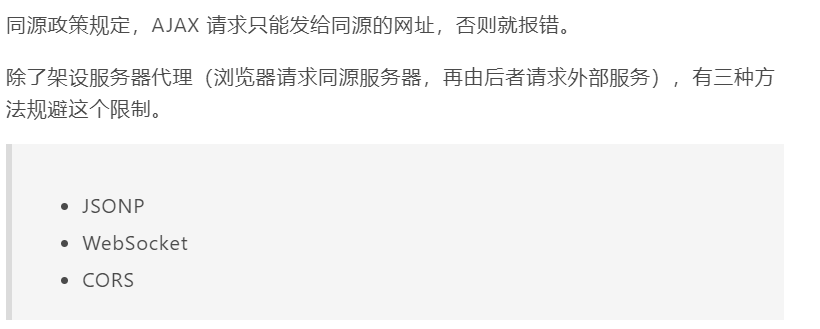




### LocalStorage

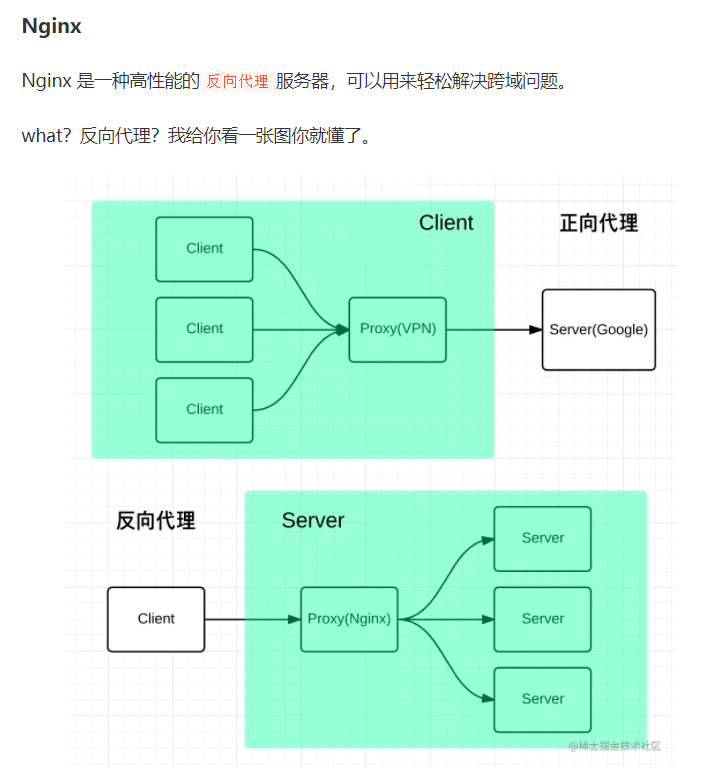


## AJAX

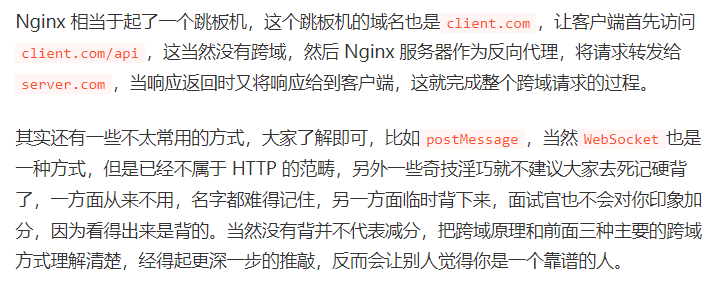






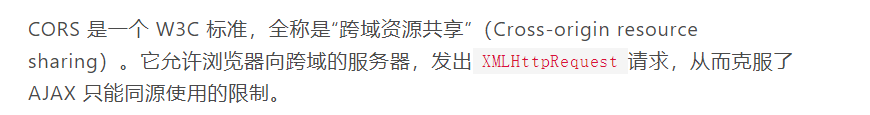


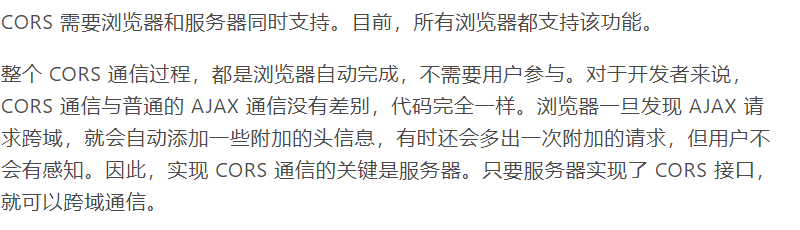






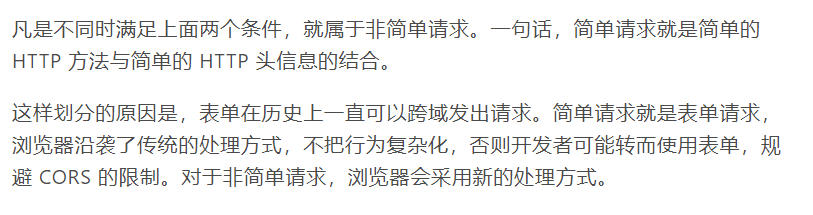
# Cors通信



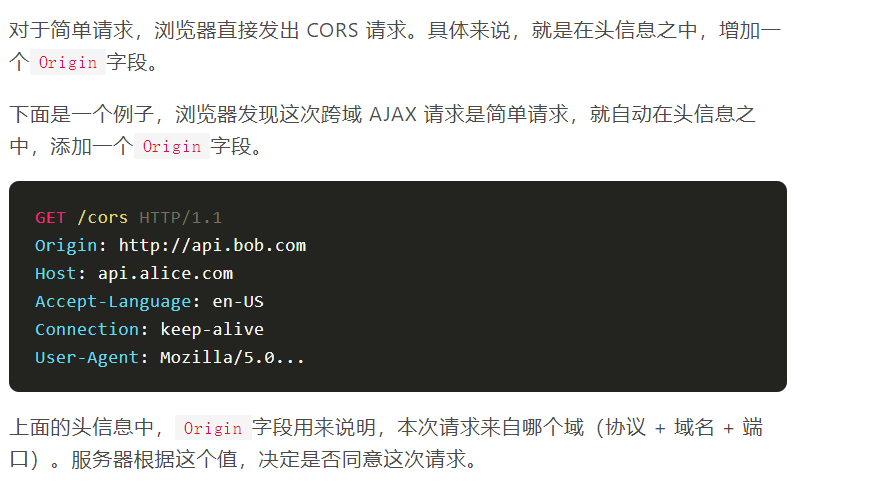


## 两种请求





## 简单请求







## withCredentials



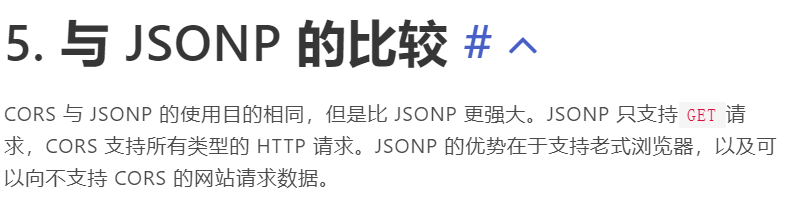
## 复杂请求



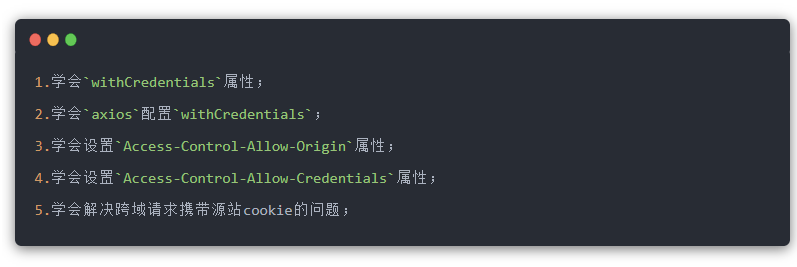








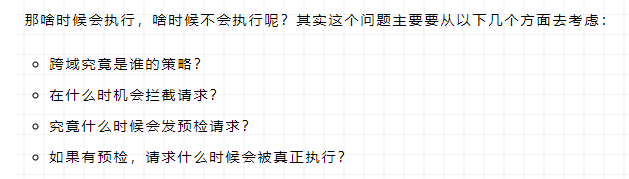
# **跨域携带cookie**

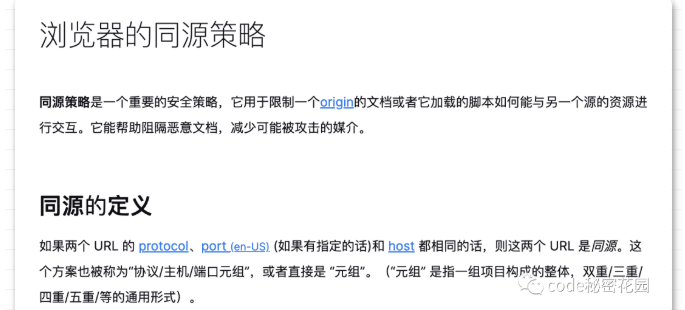


# **跨域四问**

问一个问题，如果一个请求跨域了，这个请求会在服务端被执行吗？

答案是：请求有的时候会被执行，有的时候不会执行















# Storage



# Webworker