面试前准备

简历引导、面试时优势方向引导

项目

* 这是一个怎样的项目
* 用到了什么技术，为什么用这项技术（以及每项技术很细的点以及扩展）
* 过程中遇到了什么问题，怎么解决的。

面试后

不断的调整和查缺

个人简历命名方式：求职职位+姓名；

# 面试题

## http和https

### 区别

http: 是一种超文本传输协议，https: 是以安全为目标的HTTP通道，由http和ssl协议构建的可进行**加密传输**和**身份认证**的网络协议，比http协议的安全性更高。

### https协议的工作原理

访问->返回证书（公钥）->加密等级->建立会话密钥->公钥加密->私钥解密->保持通信

### https协议的优点

可进行加密传输、身份认证的网络协议，要比http协议安全，可防止数据在传输过程中**不被窃取、改变，确保数据**的完整性。

### https协议的缺点

https握手阶段比较费时，会使页面加载时间延长50%

https缓存不如http高效，

## tcp三次握手

首先C发起请求连接S确认，第一次握手：S只可以确认 自己可以接受C发送的报文段第二次握手：C可以确认 S收到了自己发送的报文段，并且可以确认 自己可以接受S发送的报文段第三次握手：S可以确认 C收到了自己发送的报文段

## TCP和UDP的区别

TCP是**面向连接**提供可靠的服务。也就是说，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达;UDP是无连接尽最大努力交付，即不保证可靠交付。

TCP只能是1对1的，UDP支持1对1,1对多。

## get和post的区别

GET - 从指定的资源请求数据。

POST - 向指定的资源提交要被处理的数据。

get请求在url中传递的参数是有长度限制的，而post没有。

GET：不同的浏览器和服务器不同，一般限制在2~8K之间，更加常见的是1k以内。

GET和POST本质上就是TCP链接，并无差别。

GET产生一个TCP数据包；POST产生两个TCP数据包。

对于GET方式的请求，浏览器会把http header和data一并发送出去，服务器响应200（返回数据）；

而对于POST，浏览器先发送header，服务器响应100 continue，浏览器再发送data，服务器响应200 ok（返回数据）。

get参数通过url传递，post放在request body中。

get比post更不安全，因为参数直接暴露在url中，所以不能用来传递敏感信息。

get请求只能进行url编码，而post支持多种编码方式

get请求会浏览器主动cache，而post支持多种编码方式。

get请求参数会被完整保留在浏览历史记录里，而post中的参数不会被保留。

post做的一般是修改和删除的工作，所以必须与数据库交互，所以不能使用缓存。因此get请求适合于请求缓存。

## 状态码

200    OK    请求成功。一般用于GET与POST请求

301    Moved Permanently    永久移动。请求的资源已被永久的移动到新URI，返回信息会包括新的URI，浏览器会自动定向到新URI。

302    Found    临时移动。与301类似。但资源只是临时被移动。

304：未修改。所请求的资源未修改，将不会返回资源，客户端将从缓存中获取已有资源，客户端通常会缓存访问过的资源，通过提供一个头信息指出客户端希望只返回在指定日期之后修改的资源所请求的资源未修改，服务器返回此状态码时，不会返回任何资源。

(1)400状态码： 客户端请求的语法错误，服务器无法理解

产生原因：

前端提交数据的字段名称和字段类型与后台的实体没有保持一致

前端提交到后台的数据应该是json字符串类型，但是前端没有将对象JSON.stringify转化成字符串。

解决方法：

对照字段的名称，保持一致性

将obj对象通过JSON.stringify实现序列化

401状态码：当前请求需要用户验证

403状态码：服务器已经得到请求，但是拒绝执行

404 网页或文件未找到

500服务器内部错误，无法完成请求

## cookie和session的区别，localstorage和sessionstorage的区别

Cookie和session都可用来存储用户信息，cookie存放于客户端，session存放于服务器端，因为cookie存放于客户端有可能被窃取，所以cookie一般用来存放不敏感的信息，比如用户设置的网站主题，敏感的信息用session存储，比如用户的登陆信息，session可以存放于文件，数据库，内存中都可以，cookie可以服务器端响应的时候设置，也可以客户端通过JS设置cookie会在请求时在http首部发送给客户端，cookie一般在客户端有大小限制，一般为4K，

下面从几个方向区分一下cookie，localstorage，sessionstorage的区别

1、生命周期：

Cookie：可设置失效时间，否则默认为关闭浏览器后失效

Localstorage:除非被手动清除，否则永久保存

Sessionstorage：仅在当前网页会话下有效，关闭页面或浏览器后就会被清除

2、存放数据：

Cookie：4k左右

Localstorage和sessionstorage：可以保存5M的信息

3、http请求：

Cookie：每次都会携带在http头中，如果使用cookie保存过多数据会带来性能问题

其他两个：仅在客户端即浏览器中保存，不参与和服务器的通信

4、易用性：

Cookie：需要程序员自己封装，原生的cookie接口不友好

其他两个：即可采用原生接口，亦可再次封装

5、应用场景：

从安全性来说，因为每次http请求都回携带cookie信息，这样子浪费了带宽，所以cookie应该尽可能的少用，此外cookie还需要指定作用域，不可以跨域调用，限制很多，但是用户识别用户登陆来说，cookie还是比storage好用，其他情况下可以用storage，localstorage可以用来在页面传递参数，sessionstorage可以用来保存一些临时的数据，防止用户刷新页面后丢失了一些参数，

### cookie的作用：

保存用户登录状态。例如将用户id存储于一个cookie内，这样当用户下次访问该页面时就不需要重新登录了。 cookie还可以设置过期时间，当超过时间期限后，cookie就会自动消失。

跟踪用户行为。比如在浏览器中浏览天气预报网站，当设置一次后位置再次访问将不在需要设置。

## cookie session区别

1.  cookie数据存放在客户的浏览器上，session数据放在服务器上。

2.cookie不是很安全，别人可以分析存放在本地的COOKIE并进行COOKIE欺骗  
考虑到安全应当使用session。

3. session会在一定时间内保存在服务器上。当访问增多，会比较占用你服务器的性能  
考虑到减轻服务器性能方面，应当使用COOKIE。

4.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。

## 缓存分为几种

缓存分为两种：强缓存和协商缓存，根据响应的header内容来决定。

强缓存相关字段有expires，cache-control。如果cache-control与expires同时存在的话，cache-control的优先级高于expires。

协商缓存相关字段有Last-Modified/If-Modified-Since，Etag/If-None-Match

|  |  |  |  |
| --- | --- | --- | --- |
| 强缓存 | 从缓存取 | 200（from cache） | 否，直接从缓存取 |
| 协商缓存 | 从缓存取 | 304（not modified） | 是，通过服务器来告知缓存是否可用 |

### 强缓存、协商缓存什么时候用哪个

当服务器上的资源更新了浏览器就请求新的资源，没有更新就使用本地的缓存，以最大程度的减少因网络请求而产生的资源浪费。

## 前端优化

降低请求量：合并资源，减少HTTP 请求数，minify / gzip 压缩，webP，lazyLoad。

加快请求速度：预解析DNS，减少域名数，并行加载，CDN 分发。

缓存：HTTP 协议缓存请求，离线缓存 manifest，离线数据缓存localStorage。

渲染：JS/CSS优化，加载顺序，服务端渲染，pipeline。

## 浏览器在生成页面的时候，会生成那两颗树？

构造两棵树，DOM树和CSSOM规则树

当浏览器接收到服务器相应来的HTML文档后，会遍历文档节点，生成DOM树，

CSSOM规则树由浏览器解析CSS文件生成，

## 在地址栏里输入一个URL,到这个页面呈现出来，中间会发生什么？

输入url后，首先需要找到这个url域名的服务器ip,为了寻找这个ip，浏览器首先会寻找缓存，查看缓存中是否有记录，缓存的查找记录为：浏览器缓存-》系统缓存-》路由器缓存，缓存中没有则查找系统的hosts文件中是否有记录，如果没有则查询DNS服务器，得到服务器的ip地址后，浏览器根据这个ip以及相应的端口号，构造一个http请求，这个请求报文会包括这次请求的信息，主要是请求方法，请求说明和请求附带的数据，并将这个http请求封装在一个tcp包中，这个tcp包会依次经过传输层，网络层，数据链路层，物理层到达服务器，服务器解析这个请求来作出响应，返回相应的html给浏览器，因为html是一个树形结构，浏览器根据这个html来构建DOM树，在dom树的构建过程中如果遇到JS脚本和外部JS连接，则会停止构建DOM树来执行和下载相应的代码，这会造成阻塞，这就是为什么推荐JS代码应该放在html代码的后面，之后根据外部央视，内部央视，内联样式构建一个CSS对象模型树CSSOM树，构建完成后和DOM树合并为渲染树，这里主要做的是排除非视觉节点，比如script，meta标签和排除display为none的节点，之后进行布局，布局主要是确定各个元素的位置和尺寸，之后是渲染页面，因为html文件中会含有图片，视频，音频等资源，在解析DOM的过程中，遇到这些都会进行并行下载，浏览器对每个域的并行下载数量有一定的限制，一般是4-6个，当然在这些所有的请求中我们还需要关注的就是缓存，缓存一般通过Cache-Control、Last-Modify、Expires等首部字段控制。 Cache-Control和Expires的区别在于Cache-Control使用相对时间，Expires使用的是基于服务器 端的绝对时间，因为存在时差问题，一般采用Cache-Control，在请求这些有设置了缓存的数据时，会先 查看是否过期，如果没有过期则直接使用本地缓存，过期则请求并在服务器校验文件是否修改，如果上一次 响应设置了ETag值会在这次请求的时候作为If-None-Match的值交给服务器校验，如果一致，继续校验 Last-Modified，没有设置ETag则直接验证Last-Modified，再决定是否返回304

## 输入URL到页面加载显示完成发生了什么?

DNS解析

TCP连接

发送HTTP请求

服务器处理请求并返回HTTP报文

浏览器解析渲染页面

连接结束