**前端面试题**

**JavaScript:**

* **js内存泄露情况**

1.全局变量不用 var 声明的变量，相当于挂载到 window 对象上

它们被定义为不可回收（除非定义为空或重新分配）。尤其当全局变量用于临时存储和处理大量信息时，需要多加小心。如果必须使用全局变量存储大量数据时，确保用完以后把它设置为 null 或者重新定义。与全局变量相关的增加内存消耗的一个主因是缓存。缓存数据是为了重用，缓存必须有一个大小上限才有用。高内存消耗导致缓存突破上限，因为缓存内容无法被回收。

1. 计时器或回调函数。setInterval

一旦它们不再需要（或者关联的对象变成不可达），明确地移除它们非常重要。老的 IE 6 是无法处理循环引用的。如今，即使没有明确移除它们，大部分浏览器是可以回收处理函数的。

1. 脱离 DOM 的引用

保存 DOM 节点内部数据结构很有用。假如你想快速更新表格的几行内容，把每一行 DOM 存成字典（JSON 键值对）或者数组很有意义。此时，同样的 DOM 元素存在两个引用：一个在 DOM 树中，另一个在字典中。将来你决定删除这些行时，需要把两个引用都清除。



1. 闭包

* **offsetWidth/offsetHeight,clientWidth/clientHeight 与 scrollWidth/scrollHeight 的区别**
* offsetWidth/offsetHeight 返回值包含**content + padding + border**，效果与 e.getBoundingClientRect()相同
* clientWidth/clientHeight 返回值只包含**content + padding**，如果有滚动条，也**不包含滚动条**
* scrollWidth/scrollHeight 返回值包含**content + padding + 溢出内容的尺寸**

**html:**

## 页面渲染html的过程？(理解整个过程即可)

1.浏览器解析html源码，然后创建一个 DOM树。并行请求 css/image/js在DOM树中，每一个HTML标签都有一个对应的节点，并且每一个文本也都会有一个对应的文本节点。DOM树的根节点就是 documentElement，对应的是html标签。

2.浏览器解析CSS代码，计算出最终的样式数据。构建CSSOM树。对CSS代码中非法的语法它会直接忽略掉。解析CSS的时候会按照如下顺序来定义优先级：浏览器默认设置 < 用户设置 < 外链样式 < 内联样式 < html中的style。

3.DOM Tree + CSSOM --> 渲染树（rendering tree）。渲染树和DOM树有点像，但是是有区别的。

DOM树完全和html标签一一对应，但是渲染树会忽略掉不需要渲染的元素，比如head、display:none的元素等。而且一大段文本中的每一个行在渲染树中都是独立的一个节点。渲染树中的每一个节点都存储有对应的css属性。

4.一旦渲染树创建好了，浏览器就可以根据渲染树直接把页面绘制到屏幕上。

以上四个步骤并不是一次性顺序完成的。如果DOM或者CSSOM被修改，以上过程会被重复执行。实际上，CSS和JavaScript往往会多次修改DOM或者CSSOM。

* **简单说一下CORS**

跨域资源共享([CORS](https://developer.mozilla.org/zh-CN/docs/Glossary/CORS)) 是一种机制，它使用额外的 [HTTP](https://developer.mozilla.org/zh-CN/docs/Glossary/HTTP) 头来告诉浏览器  让运行在一个 origin (domain) 上的Web应用被准许访问来自不同源服务器上的指定的资源。当一个资源从与该资源本身所在的服务器不同的域、协议或端口请求一个资源时，资源会发起一个跨域 HTTP 请求。

* **get、post的区别（基础题越细越好）**

1.get传参方式是通过地址栏URL传递，是可以直接看到get传递的参数，post传参方式参数URL不可见，get把请求的数据在URL后通过？连接，通过&进行参数分割。psot将参数存放在HTTP的包体内

2.get传递数据是通过URL进行传递，对传递的数据长度是受到URL大小的限制，URL最大长度是2048个字符。post没有长度限制

3.get后退不会有影响，post后退会重新进行提交

4.get请求可以被缓存，post不可以被缓存

5.get请求只URL编码，post支持多种编码方式

6.get请求的记录会留在历史记录中，post请求不会留在历史记录

7.get只支持ASCII字符，post没有字符类型限制

* **对前端性能优化有什么了解？一般都通过那几个方面去优化的？**

请求方面

* 1. 减少 HTTP 请求：合并文件、CSS 精灵、inline Image
  2. 减少 DNS 查询：DNS 查询完成之前浏览器不能从这个主机下载任何任何文件。方法：DNS 缓存、将资源分布到恰当数量的主机名，平衡并行下载和 DNS 查询
  3. 避免重定向：多余的中间访问
  4. 使 Ajax 可缓存
  5. 非必须组件延迟加载
  6. 未来所需组件预加载
  7. 减少 DOM 元素数量
  8. 将资源放到不同的域下：浏览器同时从一个域下载资源的数目有限，增加域可以提高并行下载量
  9. 减少 iframe 数量

Server 方面

* 1. 使用 CDN
  2. 添加 Expires 或者 Cache-Control 响应头
  3. 对组件使用 Gzip 压缩
  4. 配置 ETag
  5. Flush Buffer Early
  6. Ajax 使用 GET 进行请求
  7. 避免空 src 的 img 标签

css 方面

* 1. 将样式表放到页面顶部
  2. 不使用 CSS 表达式
  3. 使用不使用@import
  4. 不使用 IE 的 Filter

Javascript 方面

* 1. 将脚本放到页面底部
  2. 将 javascript 和 css 从外部引入
  3. 压缩 javascript 和 css
  4. 删除不需要的脚本
  5. 减少 DOM 访问
  6. 合理设计事件监听器

图片方面

* 1. 优化图片：根据实际颜色需要选择色深、压缩
  2. 优化 css 精灵
  3. 不要在 HTML 中拉伸图片
  4. 保证 favicon.ico 小并且可缓存

### HTML全局属性(global attribute)有哪些

accesskey:设置快捷键，提供快速访问元素如[aaa](https://github.com/qiu-deqing/FE-interview)在 windows 下的 firefox 中按alt + shift + a可激活元素

class:为元素设置类标识，多个类名用空格分开，CSS 和 javascript 可通过 class 属性获取元素

contenteditable: 指定元素内容是否可编辑

contextmenu: 自定义鼠标右键弹出菜单内容

data-\*: 为元素增加自定义属性

dir: 设置元素文本方向

draggable: 设置元素是否可拖拽

dropzone: 设置元素拖放类型： copy, move, link

hidden: 表示一个元素是否与文档。样式上会导致元素不显示，但是不能用这个属性实现样式效果

id: 元素 id，文档内唯一

lang: 元素内容的的语言

spellcheck: 是否启动拼写和语法检查

style: 行内 css 样式

tabindex: 设置元素可以获得焦点，通过 tab 可以导航

title: 元素相关的建议信息

translate: 元素和子孙节点内容是否需要本地化

**CSS:**

* **transform、translate、transition 分别是什么属性？CSS 中常用的实现动画方式**

transform 是指变换、变形，是 css3 的一个属性，和 width，height 属性一样；  
translate是transform的属性值，是指元素进行 2D(3D)维度上位移或范围变换;  
transition 是指过渡效果，往往理解成简单的动画，需要有触发条件。

### 工作中用到了哪些 SEO优化；

1. 合理的 title、description、keywords：搜索对着三项的权重逐个减小，title 值强调重点即可，重要关键词出现不要超过 2 次，而且要靠前，不同页面 title 要有所不同；description 把页面内容高度概括，长度合适，不可过分堆砌关键词，不同页面 description 有所不同；keywords 列举出重要关键词即可
2. 语义化的 HTML 代码，符合 W3C 规范：语义化代码让搜索引擎容易理解网页
3. 重要内容 HTML 代码放在最前：搜索引擎抓取 HTML 顺序是从上到下，有的搜索引擎对抓取长度有限制，保证重要内容一定会被抓取
4. 重要内容不要用 js 输出：爬虫不会执行 js 获取内容
5. 少用 iframe：搜索引擎不会抓取 iframe 中的内容
6. 非装饰性图片必须加 alt
7. 提高网站速度：网站速度是搜索引擎排序的一个重要指标

### link与@import的区别

1.link是 HTML 方式， @import是 CSS 方式

2.link最大限度支持并行下载，@import过多嵌套导致串行下载，出现[FOUC](http://www.bluerobot.com/web/css/fouc.asp/)

3.link可以通过rel="alternate stylesheet"指定候选样式

4.浏览器对link支持早于@import，可以使用@import对老浏览器隐藏样式

5.@import必须在样式规则之前，可以在 css 文件中引用其他文件

6.总体来说：[link 优于@import](http://www.stevesouders.com/blog/2009/04/09/dont-use-import/)

### PNG,GIF,JPG 的区别及适用场景

GIF:

1. 8 位像素，256 色
2. 无损压缩
3. 支持简单动画
4. 支持 boolean 透明
5. 适合简单动画

JPEG：

1. 颜色限于 256
2. 有损压缩
3. 可控制压缩质量
4. 不支持透明
5. 适合照片

PNG：

1. 有 PNG8 和 truecolor PNG
2. PNG8 类似 GIF 颜色上限为 256，文件小，支持 alpha 透明度，无动画
3. 适合图标、背景、按钮

### 什么是 FOUC?如何避免

Flash Of Unstyled Content：用户定义样式表加载之前浏览器使用默认样式显示文档，用户样式加载渲染之后再从新显示文档，造成页面闪烁。解决方法：把样式表放到文档的head

### 外边距折叠(collapsing margins)

毗邻的两个或多个margin会合并成一个 margin，叫做外边距折叠。规则如下：

1. 两个或多个毗邻的普通流中的块元素垂直方向上的 margin 会折叠
2. 浮动元素/inline-block 元素/绝对定位元素的 margin 不会和垂直方向上的其他元素的 margin 折叠
3. 创建了块级格式化上下文的元素，不会和它的子元素发生 margin 折叠
4. 元素自身的 margin-bottom 和 margin-top 相邻时也会折叠

**http:**

### HTTP request 报文结构是怎样的？

1. 首行是Request-Line包括：请求方法，请求 URI，协议版本，CRLF
2. 首行之后是若干行请求头，包括general-header，request-header或者entity-header，每个一行以 CRLF 结束
3. 请求头和消息实体之间有一个CRLF 分隔
4. 根据实际请求需要可能包含一个消息实体 一个请求报文例子如下：

GET /Protocols/rfc2616/rfc2616-sec5.html HTTP/1.1

Host: www.w3.org

Connection: keep-alive

Cache-Control: max-age=0

Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,\*/\*;q=0.8

User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/35.0.1916.153 Safari/537.36

Referer: https://www.google.com.hk/

Accept-Encoding: gzip,deflate,sdch

Accept-Language: zh-CN,zh;q=0.8,en;q=0.6

Cookie: authorstyle=yes

If-None-Match: "2cc8-3e3073913b100"

If-Modified-Since: Wed, 01 Sep 2004 13:24:52 GMT

name=qiu&age=25

### HTTP response 报文结构是怎样的

1. 首行是状态行包括：HTTP 版本，状态码，状态描述，后面跟一个 CRLF
2. 首行之后是若干行响应头，包括：通用头部，响应头部，实体头部
3. 响应头部和响应实体之间用一个 CRLF 空行分隔
4. 最后是一个可能的消息实体 响应报文例子如下：

HTTP/1.1 200 OK

Date: Tue, 08 Jul 2014 05:28:43 GMT

Server: Apache/2

Last-Modified: Wed, 01 Sep 2004 13:24:52 GMT

ETag: "40d7-3e3073913b100"

Accept-Ranges: bytes

Content-Length: 16599

Cache-Control: max-age=21600

Expires: Tue, 08 Jul 2014 11:28:43 GMT

P3P: policyref="http://www.w3.org/2001/05/P3P/p3p.xml"

Content-Type: text/html; charset=iso-8859-1

{"name": "qiu", "age": 25}

1. **babel原理**

Babel 是 JavaScript 编译器，更确切地说是源码到源码的编译器，通常也叫做“转换编译器（transpiler）”

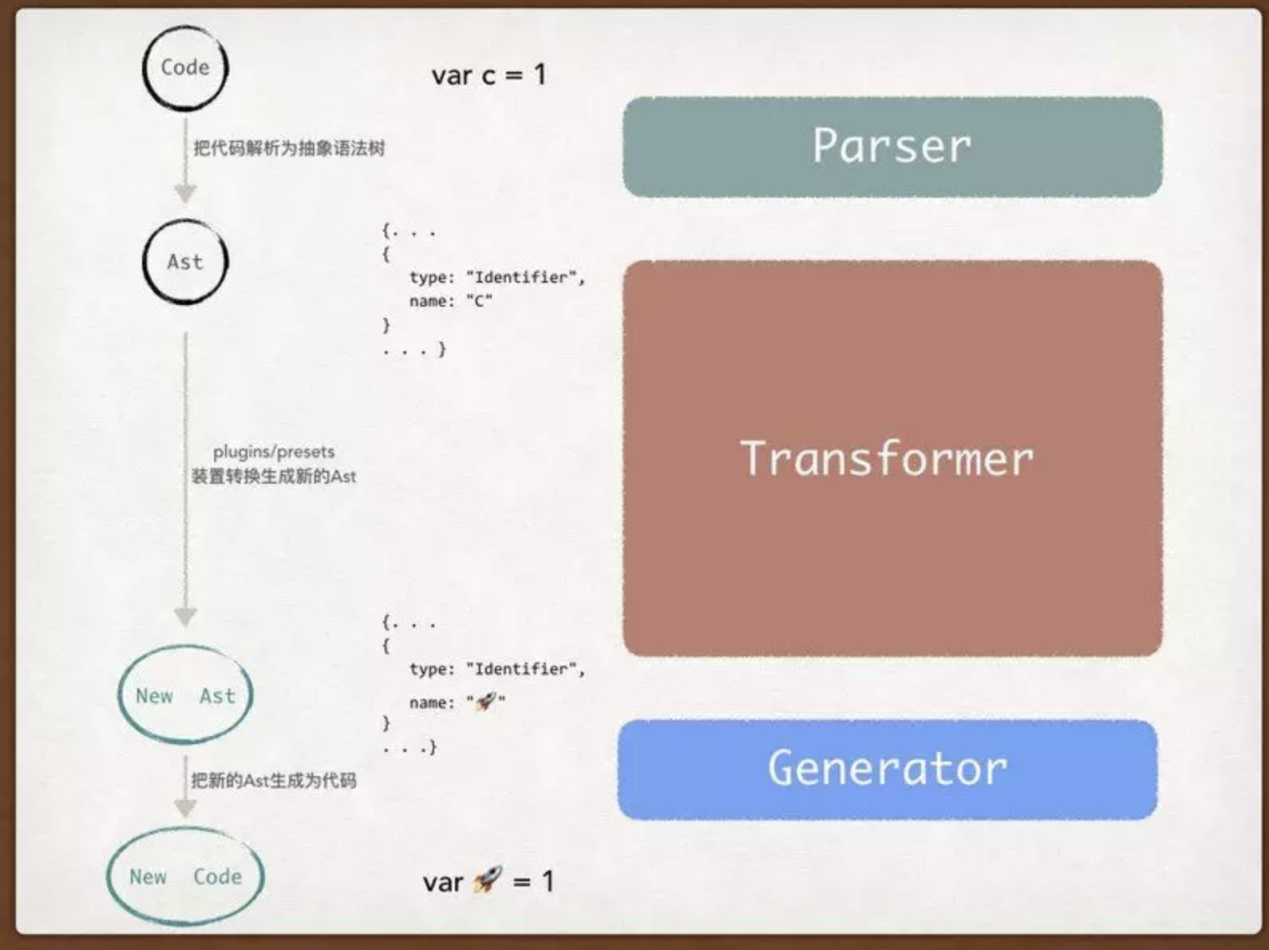
babel 的三个主要处理步骤分别是： 解析（parse），转换（transform），生成（generate），

解析过程分为2个阶段，词发分析和语法分析，词发分析相单于把字符串代码转化成令牌流（语法片段数组）。语法分析把令牌流转化成AST（抽象语法树）。

转换过程针对解析出的AST（抽象语法树）进行遍历，在此过程中对节点进行添加、更新及移除等操作。 这是 Babel 或是其他编译器中最复杂的过程 同时也是插件将要介入工作的部分。

生成过程把最终转化后的AST转化成字符串形式的代码，同时创建源码映射（source maps

）

****

1. **什么是AST？**

抽象语法树（abstract syntax tree 或者缩写为 AST），或者语法树（syntax tree），是源代码的抽象语法结构的树状表现形式，这里特指编程语言的源代码。树上的每个节点都表示源代码中的一种结构，之所以说语法是「抽象」的，是因为这里的语法并不会表示出真实语法中出现的每个细节。

使用场景：

* JS 反编译，语法解析
* Babel 编译 ES6语法
* 代码高亮
* 关键字匹配
* 作用域判断
* 代码压缩

在线解析：<https://astexplorer.net/>

1. **webpack相关**

一、什么是loader

loader是文件加载器，能够加载资源文件，并对这些文件进行一些处理，诸如编译、压缩等，最终一起打包到指定的文件中处理一个文件可以使用多个loader，loader的执行顺序和配置中的顺序是相反的，即最后一个loader最先执行，第一个loader最后执行,第一个执行的loader接收源文件内容作为参数，其它loader接收前一个执行的loader的返回值作为参数，最后执行的loader会返回此模块的JavaScript源码

二、什么是plugin

在webpack运行的生命周期中会广播出许多事件，plugin可以监听这些事件，在合适的时机通过webpack提供的API改变输出结果。

三、loader和plugin的区别

对于loader，它是一个转换器，将A文件进行编译形成B文件，这里操作的是文件，比如将A.scss转换为A.css，单纯的文件转换过程

plugin是一个扩展器，它丰富了webpack本身，针对是loader结束后，webpack打包的整个过程，它并不直接操作文件，而是基于事件机制工作，会监听webpack打包过程中的某些节点，执行广泛的任务

# 弹性盒子中 flex: 0 1 auto 表示什么意思

三个参数分别对应的是 flex-grow, flex-shrink 和 flex-basis，默认值为0 1 auto。 1.flex-grow属性定义项目的放大比例，默认为0，即如果存在剩余空间，也不放大。 2.flex-shrink属性定义了项目的缩小比例，默认为1，即如果空间不足，该项目将缩小。

1. flex-basis属性定义了在分配多余空间之前，项目占据的主轴空间（main size）。

# vue 渲染大量数据时应该怎么优化？

1.添加加载动画，优化用户体验

2.利用服务器渲染SSR，在服务端渲染组件

3.避免浏览器处理大量的dom，比如懒加载，异步渲染组件，使用分页

4.对于固定的非响应式的数据，使用Object.freeze冻结

**接口如何防刷?**

1．网关控制流量洪峰，对在一个时间段内出现流量异常，可以拒绝请求

2．源ip请求个数限制。对请求来源的ip请求个数做限制

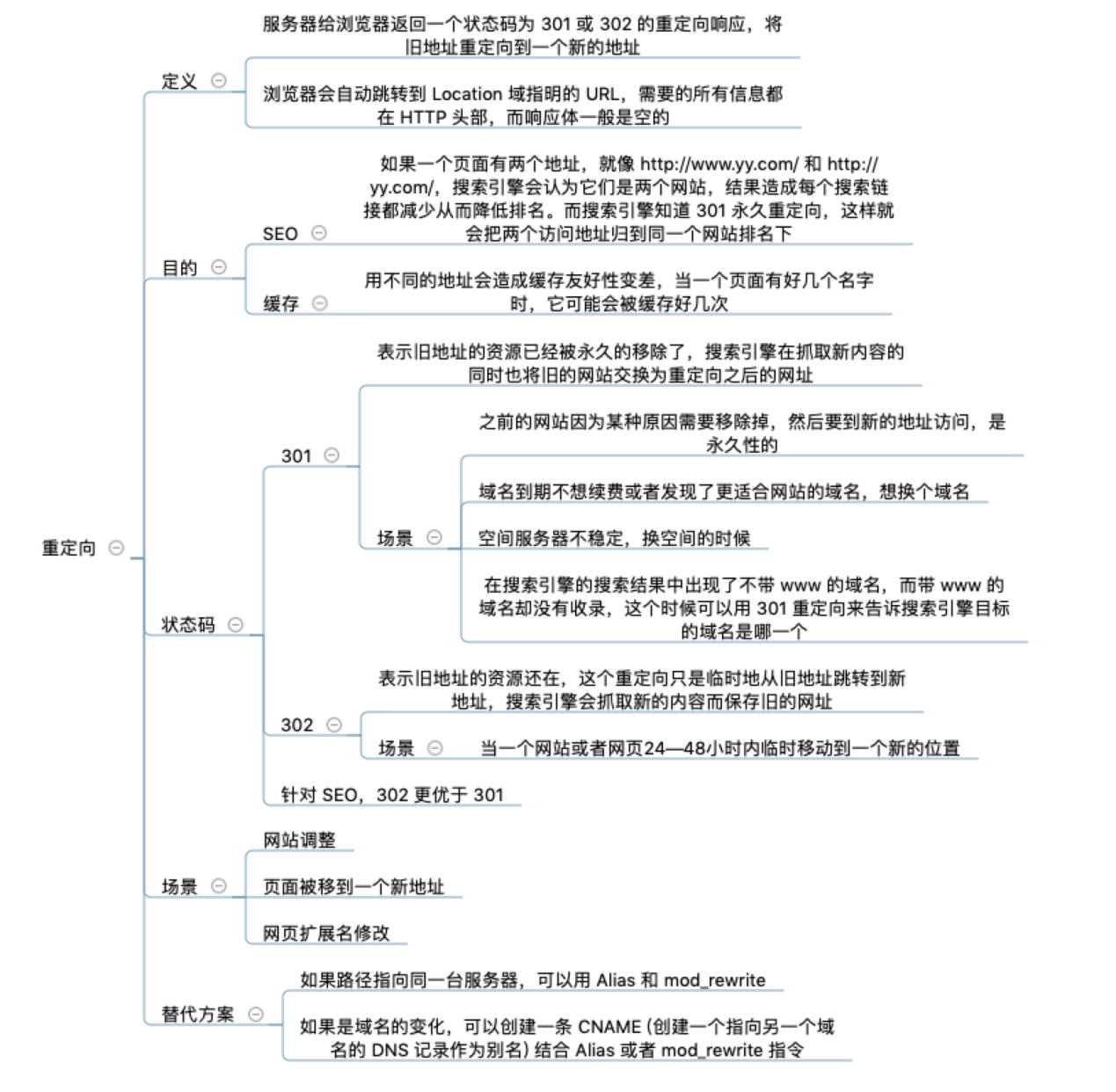
3．http请求头信息校验；（例如host，User-Agent，Referer）

4．对用户唯一身份uid进行限制和校验。例如基本的长度，组合方式，甚至有效性进行判断。或者uid具有一定的时效性

5．前后端协议采用二进制方式进行交互或者协议采用签名机制

6．人机验证，验证码，短信验证码，滑动图片形式，12306形式

**Http 状态码 301 和 302 的应用场景分别是什么**



h**ttp2.0的多路复用是什么**

HTTP2采用二进制格式传输，取代了HTTP1.x的文本格式，二进制格式解析更高效。 多路复用代替了HTTP1.x的序列和阻塞机制，所有的相同域名请求都通过同一个TCP连接并发完成。在HTTP1.x中，并发多个请求需要多个TCP连接，浏览器为了控制资源会有6-8个TCP连接都限制。 HTTP2中 同域名下所有通信都在单个连接上完成，消除了因多个 TCP 连接而带来的延时和内存消耗。 单个连接上可以并行交错的请求和响应，之间互不干扰

**介绍下 webpack 热更新原理，是如何做到在不刷新浏览器的前提下更新页面**

1.当修改了一个或多个文件；

2.文件系统接收更改并通知webpack；

3.webpack重新编译构建一个或多个模块，并通知HMR服务器进行更新；

4.HMR Server 使用webSocket通知HMR runtime 需要更新，HMR运行时通过HTTP请求更新jsonp；

5.HMR运行时替换更新中的模块，如果确定这些模块无法更新，则触发整个页面刷新。

**介绍下 BFC 及其应用**

BFC 就是块级格式上下文，是页面盒模型布局中的一种 CSS 渲染模式，相当于一个独立的容器，里面的元素和外部的元素相互不影响。创建 BFC 的方式有：

1．html 根元素

2．Float 浮动

3．对定位

4．overflow 不为 visiable

5．display 为表格布局或者弹性布局

BFC 主要的作用是： 清除浮动 防止同一 BFC 容器中的相邻元素间的外边距重叠问题

**浏览器和Node 事件循环的区别**

关于微任务和宏任务在浏览器的执行顺序是这样的：

\* 执行一只task（宏任务）

\* 执行完micro-task队列 （微任务）

如此循环往复下去

常见的 task（宏任务） 比如：setTimeout、setInterval、script（整体代码）、 I/O 操作、UI 渲染等。

常见的 micro-task 比如: new Promise().then(回调)、MutationObserver(html5新特性) 等。