面试题

CSS：

- CSS实现一组自适应九宫格有几种方法？
- flex 布局与 grid 布局的特点和用法？两者比较？应用场景？
- CSS实现动画有哪些属性？transition/transform/animation有什么区别？如果用JS实现动画有哪些方法呢？两者有什么差异？怎么衡量一个动画流不流畅（fps）？如何实现一个动画将一个元素从左边滑到右边？
- 怎么用CSS实现一个长宽比固定（比如1：3）的响应式div（伪元素嵌套 / padding / vw或vh）（宽度等分，高度固定）
- 为什么给一个动画元素设置CSS will-change: transform; perspective: 1000;可以提高性能？
- 移动端尺寸适配方案有哪些？优缺点？（vw/vh/rem）

JavaScript：

- 判断一个JS变量的类型有哪些方法？
- 什么是闭包，原理？有什么应用场景？
- 浏览器数据存储数据方案有哪些？区别？
- 事件冒泡的应用场景？如何识别是哪个节点触发的？
- JS的symbol类型了解吗？有什么用途？
- 如何解决script标签阻塞浏览器解析的问题？defer和async有什么区别？
- 如何实现一个懒加载图片组件？
- 浏览器原生怎么发起请求？ajax和fetch的区别？
- CommonJS,CMD,AMD,ES6模块之间的差异以及优缺点？（输出拷贝/输出引用、静态分析/动态加载）
- 为什么export和import只能在模块最外层作用域，为什么不能在条件语句内？
- 如何让一个对象不能被修改（比如修改、增添属性）？
- package.json里面的 "browser", "module", "main" 三个入口字段区别？使用场景？如何确定我们开发的时候依赖引入的是哪一个？或者说引入优先级
- equestIdleCallback和requestAnimationFrame分别有什么用途？

- 如果公司图片部门发明了一种新的图片格式可以在相同显示效果下体积很小传输速度很快，但是这个格式浏览器原生不支持直接显示，可以用什么方法可以让前端用起来？

Typescript：

- Typescript怎么获取一个函数的返回值类型？实现：

type Foo = () => {a: string}

type A = MyReturnType // {a: string}

// 实现MyReturnType

// 答案

type MyReturnType<T extends (...params: any[]) => any> =

T extends (...params: any[]) => infer P ? P : never;

- TS 怎么把一个每个字段都是必选的对象参数变成可选的？

Vue/React（框架）：

- v-show和v-if指令都能隐藏元素有什么不同？分别适合什么场景？
- Vue中组件的data为什么是一个返回对象函数，而不是一个对象
- react-router/Vue-router实现原理？或者说单页应用前端路由的实现原理？
- React几个hooks的作用是什么？hooks比起class的优缺点？踩过什么坑？hooks内部的实现了解？（为什么需要按顺序？不能if else，FiberNode，链表）
- SSR的优缺点？除了SSR还了解其他SEO方法吗？怎么衡量SEO效果？
- SSR是针对整个页面的吗？能不能只针对页面上的一个组件SSR？
- redux 做状态管理和直接用发布订阅模式有什么不同？
- react-redux 的原理，是怎么把 redux 跟 react 连接起来的？
- vue $nextTick的原理？
- React 16的底层架构有什么变化，Fiber是什么？解决了什么问题？
- 了解 React 一些新特性吗？suspense、server component usetransition？
- Vue3.0有了解吗？相比2.0有什么变化？Vite的原理了解？

Node.js：

- 列举几个常用的Node内置模块？Node的Stream模块是什么？有什么应用场景？
- 为什么Node服务需要用pm2部署？实现一个PM2大概要做哪些事情？

- Node性能问题，比如线上应用CPU负载过高、内存泄漏，经常崩溃，要怎么去排查？（比如说怎么查看CPU和内存使用情况？）
- 追踪异步调用的场景如何打日志？考察async_hooks AsyncLocalStorage

webpack：

- webpack Plugin 和 Loader 的区别？写过loader或者plugin吗？它们的工作原理是怎么样的？Compiler和Compilation知道吗？
- Webpack 中 chunkHash 与 contentHash 区别？
- Webpack treeShaking 的原理？
- webpack import 动态加载如何实现？
- 除了Webpack还知道哪些前端构建工具？优缺点分析？比webpack好在哪？

网络：

- HTTP缓存机制？分别什么时候用？
- TCP和UDP的区别？分别适用场景？
- HTTP和HTTPS协议的区别，HTTPS的加密过程？
- HTTP2相比HTTP1有什么新增特性？HTTP3/quic有了解吗？
- TCP是如何保证可靠传输的？有哪些策略机制？

移动端：

- 小程序底层的设计原理是怎么样？为什么可以禁止直接操作DOM和访问window？
- Flutter和RN有什么不同，从原理上分析？优缺点？
- App与H5间通信是怎么样做的？bridge的底层实现原理是什么样？
- H5&小程序&RN&Flutter四种跨平台技术的性能比较？原因？
- 小程序性能优化有哪些方法？
- 线上有用户反馈使用我们的App端内h5白屏要怎么去排查处理？
- Taro跨平台的原理是什么？Taro3的实现有了解吗？编译型和运行时型两种方案有什么优缺点？
- 小程序中客户端原生组件是怎么渲染到页面上的？小程序的同层渲染了解吗？原理是什么？可能引起什么问题？如何解决？

计算机基础：

- 进程和线程的区别？什么是死锁，如何避免？进程通信的几种方式
- 操作系统页表的作用？置换算法？
- 什么是数据库事务？有什么用？可以举举例子
- 知道数据库的索引吗？为什么不对表中的每一列都创建索引呢？知道索引用什么数据结构实现的吗？
- 一组100个顺序打乱的数，取前面最大的三个？
- JavaScript 中数组sort排序用的哪种算法？假如让你来设计？数据量小时？数据量大时？稳定和非稳定排序的意思？有哪些？

其他：

- CDN 是什么？怎么运行的？为什么能加速？
- 怎么衡量一个页面的性能？指标有哪些？怎么采集/计算这些指标？优化方法？
- 前端页面错误异常收集都有哪些手段？
- Echarts背后实现技术有了解吗？比较下canvas和svg？
- 账号登录功能前后端如何实现，背后原理和流程是什么？

编码：

- 节流和防抖
- 实现URLSearch参数解析器

```
// 构造函数支持传入 URL 参数串
searchParams = new URLSearchParams("foo=1&bar=2")
// 构造函数也支持传入一个包含参数键值对的对象
searchParams = new URLSearchParams({foo: "1", bar: "2"})
// 实例支持 get、set、has 三个方法 console.log(searchParams.get("foo")) // "1"
console.log(searchParams.set("foo", "10”)) // "foo=10&bar=2"
console.log(searchParams.has("bar")) // true
```

- 异步调度器

```
class Scheduler { async add(promiseFunc: () => Promise): Promise { }
}
const scheduler = new Scheduler()
const timeout = (time) => {
```

```
return new Promise(r => setTimeout(r, time))
}
const addTask = (time, order) => {
scheduler.add(() => timeout(time))
.then(() => console.log(order))
}
addTask(1000, 1)
addTask(500, 2)
addTask(300, 3)
addTask(400, 4)
// log: 2 3 1 4
```

- 实现find函数

```
const data = [
{userId: 8, title: 'title1'},
{userId: 11, title: 'other'},
{userId: 15, title: null},
{userId: 19, title: 'title2'}
];
// 查找 data 中，符合条件的数据，并进行排序
const result = find(data).where({
'title': /\d$/
}).orderBy('userId', 'desc');
console.log(result);  // [{ userId: 19, title: 'title2'}, { userId: 8, title: 'title1' }];
```

算法：

- 版本号排序 ['0.1.1', '2.3.3', '0.302.1', '4.2', '4.3.5', '4.3.4.5'] -> ['4.3.5','4.3.4.5','2.3.3','0.302.1','0.1.1']
- 假设你正在爬楼梯。需要 n 阶你才能到达楼顶。

  每次你可以爬 1 或 2 个台阶。你有多少种不同的方法可以爬到楼顶?

  追问：不用递归怎么写？（尾递归优化）
- 找零钱（贪心法）：https://leetcode-cn.com/problems/coin-change/
- 全排列（回溯）：https://leetcode-cn.com/problems/permutations/

- 跳跃游戏（贪心法）：https://leetcode-cn.com/problems/jump-game/
- 无重复字符的最长子串：https://leetcode-cn.com/problems/longest-substring-without-repeating-characters/