面试题V2

CSS

- 两栏布局：口述float、position absolute、flex的实现方法。
  - 左边固定宽度，右边自适应宽度

- flex使用

```
父div  width: 900px  display: flex
子div1 width: 300px  flex: 1
子div2 width: 100px  flex: 2
计算div1和div2的最终宽度
```

- 盒模型
- 响应式布局怎么做的?
- rem?

JS

- 变量数据类型判断：String、Array、BigInt、Symbol、NaN、Object（空数组）
- this考察

PHP

```php
// 下面代码console.log输出的结果是什么?
const length = 10;
function fn() {
 return this.length + 1;
}

const obj = {
 length: 5,
 test1: function() {
   return fn();
 }
};

obj.test2 = fn;

console.log(obj.test1())
console.log(fn() === obj.test2())
```

- 【编码】一般浏览器会限制并发请求数，微信小程序之前也限制过最多请求不超过10个。

  实现限制函数scheduler，接收最大并发请求数max，返回函数schedule。

  使用schedule执行请求函数时，实现同时并发数最大为max。

TypeScript

```typescript
function scheduler(max) {

  return function schedule() {

  };
}

const schedule = scheduler(5);
// request是一个function，返回promise
schedule(request)
schedule(request)
schedule(request)
schedule(request)
schedule(request)
schedule(request)
```

- 连续对同一个搜索接口发出多个请求，怎样保证最后发出的请求会最后生效?
- 深拷贝对象

- 正则表达式：写一下判断合法https url的正则表达式
- 防抖和节流 （leading or trailing）

JavaScript

```
// 编辑
// @param fn 函数
// @param ms 时间
function debounce(fn, ms) {

}
```

项目经验

- 开发流程和上线流程
- 跨域问题
- 事件委托
- cookie、sessionStorage、localStorage 之间的区别、以及在你项目中的应用?
  - 鉴权
- websocket是什么？和http区别？IM中使用websocket的原因?
  - websocket有三次握手么?
  - 怎样知道连接断开了？怎样知道对方读了消息？怎样鉴权?
- SSR

框架

React

- 一个组件有哪些情况会触发重新渲染?
- 生命周期?
- useEffect执行逻辑、document.addEventListener使用注意、和componentDidUpdate区别。
- 使用function component时，提供防抖函数debounce和请求函数fetchData，实现对请求函数防抖

JavaScript

```
import { debounce } from 'lodash';

function CustomComponent() {

  function fetchData() {
    // 不用实现fetchData
  }

  return (
    <Button onClick={fetchData} />
  );
}
```

- 实现倒计时组件

CoffeeScript

```

```

Vue

- 一个组件有哪些情况会触发重新渲染?
- 设计并实现一个弹窗组件

JavaScript

```

```

编码

- 写一个函数deepCamelCase，将一个对象的key深度转化成驼峰。
  - 可以使用camelCaseKey函数：camelCaseKey函数作用是将字符串转化成驼峰形式

JavaScript

```
// api函数，可以在deepCamelCase中使用。输入 camel_key  => 输出 camelKey
function camelCaseKey(str) {
  // 不用实现

}

// 输入示例 { camel_key: { camel_key: [ { camel_Key: 123 }]}}
// 输出示例 { camelKey:  { camelKey:  [ { camelKey:  123 }]}}
function deepCamelCase(obj) {

}

deepCamelCase({ camel_key: { camel_key: [ { camel_Key: 123 }]}})
```

· 基础：蛇形打印二叉树

https://marvel.bytedance.net/#/question/details/859

评分标准：代码良好3分

输入一棵二叉树，比如：

JavaScript

```

```

Plain Text

```
              0
          1       2
        3   4   5   6
      7   8         9  10
将它蛇形输出，结果如下：  0, 1, 2, 6, 5, 4, 3, 7, 8, 9, 10
```

· 基础：字符串操作

Rust

```
对输入的字符串，去除其中的字符'b'以及连续出现的'a'和'c'，
不允许使用类似string.replace以及正则。要求时间、空间复杂度尽量优化
'aacbd' -> 'ad'
'aabcd' -> 'ad'
'aaabbccc' -> '' // 'aaaccc' => 'aacc' => 'ac' => ''

```

- 深拷贝实现
- bind实现
- Promise实现
- 函数科里化

TypeScript

```
// 给定函数fn和数字n，表示科里化函数的整体参数数量
function curryAdd(fn, n)
```

智力题

组件设计

- 弹窗组件实现思路
- 如何实现一个轮播组件?
  - 随鼠标拖动而滚动
    - 随鼠标拖动移动到下一页
  - 过渡动画
  - 事件监听
    - 如何区分用户的意图是点击还是滑动?
  - 无限循环
  - 图片懒加载

其他

- 怎样学习新知识的？有看过哪些书籍?
- 代码出现bug怎样解决?
  - 如果遇到第三方库有问题怎么办?