面试

CSS

布局

sticky footers (middle)

实现一个布局，如图（可以给他看图片或者在纸上画）。

要求：

1. 当内容很少的时候，footer固定在底部。
2. 如果页面的内容超出视口底部，则页脚将正常位于内容下方。

sticky footers

```
<div class="wrapper">
    <header class="page-header">This is the header</header>
    <article class="page-body">
        <p>Main page content here, add more if you want to see the footer
push down.</p>
    </article>
    <footer class="page-footer">Sticky footer</footer>
</div>
```

Grid解法：

```
.wrapper {
    min-height: 100%;
    display: grid;
    grid-template-rows: auto 1fr auto;
}
```

考点：

1. min-height设置为 100% 这意味着它与它所在的容器一样高。
2. grid-template-rows的原理，单位 fr。
3. 延伸grid布局的其他内容。

Flex解法：

```
.wrapper {
    min-height: 100%;
    display: flex;
    flex-direction: column;
```

```
}

.page-header,
.page-footer {
    flex-shrink: 0;
}

.page-content {
    flex-grow: 1;
}
```

考点：

1. `min-height`设置为 `100%` 这意味着它与它所在的容器一样高。
2. `flex-grow`、`flex-shrink`、`flex-basic`的默认值和原理。
3. 延伸flex布局的其他内容。

position解法：

```
.wrapper {
    min-height: 100%;
}

.page-footer {
    position: sticky;
    top: calc(100%);
}
```

考点：

1. `min-height`设置为 `100%` 这意味着它与它所在的容器一样高。
2. `position: sticky;`的原理，为什么要设置top为`calc(100%)`。

给分：

1. Grid解法，给出方案给4分，答出原理给6分。（加分项）
2. Flex解法，给出方案给3分，答出原理给5分。
3. position解法，给出方案给3分，答出原理给5分。
4. 答出其他方案，若合理给3分；若超出预期给5分；若错误给0分。

选择器优先级 (easy)

```
#test1 #parent {
    color: green;
}
#test h1 {
    color: purple;
}
```

```
#parent2 h1 {
    color: green;
}

#test2 h1 {
    color: purple;
}

* #foo {
  color: green;
}
*[id="foo"] {
  color: purple;
}

#test4 div.outer p {
  color:orange;
}
#tes4t div:not(.outer) p {
  color: blueviolet;
}
```

```
<html>
<body id="parent">
    <section id="test1">
        <header id="parent">
            <h1>Here is a title!</h1>
        </header>
    </section>
    <section id="test2">
        <header id="parent2">
            <h1>Here is a title</h1>
        </header>
    </section>
    <section id="test3">
        <p id="foo">I am a sample text.</p>
    </section>
    <section id="test4">
        <div class="outer">
            <p>This is in the outer div.</p>
            <div class="inner">
                <p>This text is in the inner div.</p>
            </div>
        </div>
    </section>
</body>
</html>
```

考察：

1. 为目标元素直接添加样式，永远比继承样式的优先级高，无视优先级的遗传规则。所以第一个例子字体的颜色是紫色。
2. 优先级无视DOM树中的距离。所以第二个例子字体的颜色是紫色。
3. 优先级是基于选择器的形式进行计算的。在第三个例子中，尽管选择器*[id="foo"] 选择了一个ID，但是它还是作为一个属性选择器来计算自身的优先级。所以字体的颜色为绿色。
4. :not 否定伪类在优先级计算中不会被看作是伪类. 事实上, 在计算选择器数量时还是会把其中的选择器当做普通选择器进行计数。所以第一行是橘色，第二行是紫色。（加分项）

给分：

1. 前三项都答对5分，答对两项3分，答对一项1分。第四项答对加2分。

js

Number、String、Boolean

convert a string to number (middle)

实现一个函数，要求对于数字内容的输入（例如：123, '123'），输出它的数值类型（即123）；对于其他输入，抛出异常。

示例：

```
// right
toNumber('123') // output 123
toNumber(123) // output 123
toNumber('  123') // output 123

// wrong
toNumber('') // throw error
toNumber(NaN) // throw error
toNumber('1a') // throw error
toNumber('  ') // throw error
```

考点： `parseFloat`、`Number()`、`isNaN()`、`Number.isNaN()`（加分项）

解法：

```
const toNumber = (numberStr) => {
    let num1 = parseFloat(numberStr);
    let num2 = Number(numberStr);
    if (!Number.isNaN(num1) && !Number.isNaN(num2)) {
        return num1;
    } else {
        throw new Error('value is not number');
    }
}
```

给分：

1. 答出 `parseFloat`、`Number`的原理给3分。
2. 答出 `Number.isNaN()` 与 `isNaN()`的区别给2分。
3. 实现函数给5分。

变量的声明

变量的声明有多少种 (easy)

`var`、`function`、`let`、`const`、`import`、`class` （后面两种加分项）

如何遍历对象的key和value 或 遍历一个对象，打印出value。 (hard)

解法1：

```
for (const key in O) {
    console.log(O[key]);
}
```

考点：const，let的区别。可以接下来询问在for循环外面打印key的值会得到什么，为什么？考察块作用域；或者询问用这种方式可不可以遍历一个类数组，考察for in的原理。

解法2：

```
for (const key of Object.keys(O)) {
    console.log(O[key]);
}

// or

let keys = Object.keys(O);
keys.forEach(key => console.log(O[key]));
```

考点：第一种方式考察for...of、Iterator的概念，接下来可以让面试者简单实现一个`makeIterator`函数。第二种考察Object.keys的原理（可枚举，自身成员），延伸问`Object.entries()`、`Object.values()`以及for...of

makeIterator的要求：输入一个数组，返回一个遍历器对象，每一次调用遍历器对象的next方法，都会返回当前成员的信息。返回的信息（数据结构）是一个包含value和done两个属性的对象。其中，value属性是当前成员的值，done属性是一个布尔值，表示遍历是否结束。

Promise

1 (easy)

有三个函数A、B、C，每个函数都是一个请求任务，要求当三个请求任务都结束后，打印出success。如何实现这个三个函数以及主函数（main）。

考察：Promise的基本使用，Promise的三种状态以及Promise.all方法的原理。

打分：

1. Promise的基本使用，給2分。
2. Promise的基本使用及三种状态，給3分。
3. Promise.all的原理给5分。

2 (hard)

下面的代码结果是什么？为什么？

```
console.log('script start');
setTimeout(function() {
  console.log('setTimeout');
}, 0);
Promise.resolve().then(function() {
  console.log('promise1');
}).then(function() {
  console.log('promise2');
});
console.log('script end');
```

考点：什么是事件循环？调用堆栈和任务队列之间有什么区别？macrotask与microtask。

Array

Array.map与forEach的区别 (easy)

如何copy一个数组，你能想到几种方式。 (easy)

1. arr.filter(true)
2. arr.slice();
3. ....

打分：

1. 答出至少两个以上给3分。

Array from与of的区别 (easy)

如何连接多个数组 (easy)

1. arr1.concat(arr2).concat(arr3)
2. [...arr1, ...arr2, ...arr3];

框架 (根据情况，二选一)

Vue

什么是vue的计算属性？计算属性可不可以被赋值？如何做？(middle)

为什么vue实例的data必须是一个函数以及为什么可以直接在vue实例上访问到data上的属性值？(middle)

考察javascript原型链以及代理模式

对 keep-alive 的了解？（加分）(hard)

React

1 (middle)

```
<Demo style={{height: '100px'}}></Demo>
```

按照如上代码写，可不可以？会有什么影响嘛？

你了解React fiber嘛？（加分）(hard)

http

跨域问题 (middle)

1. 什么是跨域？
2. 为什么不能跨域？
3. 跨域的解决方案

http、https、http2的区别（加分）(hard)

构建工具

webpack你了解多少？

总览

- easy: 7 (每道题2-3分钟左右)
- middle: 6 （每道题4-5分钟）
- hard: 4 （每道题8-10分钟）