前端面试题

1. 请实现一个未知宽高元素的水平垂直居中的多种方法？

1 display:flex和align-items：center；justify-content：center

2 position：absolute；top:50%;left: 50%；transform: translate(-50%, -50%);

3 margin: position:absolute; left:0; right:0; top:0; bottom:0; margin:auto;

1. CSS的盒子模型？

**一个元素占有空间的大小由几个部分构成，其中包括：**

* **元素的内容（content），**
* **元素的内边距（padding），内容与边框之间的距离**
* **元素的边框（border），**
* **元素的外边距（margin），边框与外部元素之间的距离**

**四个部分，这四个部分一起构成了盒子模型。**

**盒模型分为 ： W3C标准盒子 、IE盒子**

**W3C标准盒子：它所说的width一般只包含内容，不包含padding与margin**

**盒子总宽 = margin + border + padding + width**

**IE盒子：元素的 width ，实际上包含了 content + padding + border**

**盒子总宽 = margin + width**

1. Javascript中的定时器有哪些？

setTiemout：设置一个定时器，在定时器到期后执行一次函数或代码段

setInterval：以固定的时间间隔重复调用一个函数或者代码段

setImmediate：在浏览器完全结束当前运行的操作之后立即执行指定的函数requestAnimationFrame：专门为实现高性能的帧动画而设计的API，但是不能指定延迟时间，而是根据浏览器的刷新频率而定（帧）

1. 请描述LocalStorage和SessionStorage以及Cookie的区别？

|  |  |  |  |
| --- | --- | --- | --- |
| **特性** | **Cookie** | **localStorage** | **sessionStorage** |
| 数据的生命周期 | 可设置失效时间，默认是浏览器关闭后失效 | 除非被清除，否则永久保存 | 仅在当前会话下有效，关闭页面或浏览器后被清除 |
| 存放数据大小 | 4K左右 | 一般为5MB | 一般为5MB |
| 与服务器端通信 | 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的问题 | 仅在客户端（即浏览器）中保存，不参与和服务器的通信 | 仅在客户端（即浏览器）中保存，不参与和服务器的通信 |
| 易用性 | 需要程序员自己封装，源生的Cookie接口不友好 | 源生接口可以接受，亦可再次封装来对Object和Array有更好的支持 | 源生接口可以接受，亦可再次封装来对Object和Array有更好的支持 |

1. 请写出你对闭包的理解，并列出相对应的优缺点？

使用闭包主要是为了设计私有的方法和变量。闭包的优点是可以避免全局变量的污染，缺点是闭包会常驻内存，会增大内存使用量，使用不当很容易造成内存泄露。

闭包有三个特性：

1.函数嵌套函数

2.函数内部可以引用外部的参数和变量

3.参数和变量不会被垃圾回收机制回收

1. JavaScript的数据类型有哪几种？

1. Number 数字类型

2. String 字符串类型

3. Boolean 布尔类型

4. Object 对象（Function、Array、Date…）

5. Null

6. Undefined 没有定义类型

1. 对该数组numberArray = [3,9,4,6,1,5]; 实现降序排列?

var numbers = [4, 2, 5, 1, 3];

numbers.sort(function(a, b) {

return a - b;

});

console.log(numbers);

1. 说说你对this的理解？以及ES6中的箭头函数的this指向

一、普通函数中this

（1）总是代表着它的直接调用者，如obj.fn，fn里的最外层this就是指向obj

（2）默认情况下，没有直接调用者，this指向window

（3）严格模式下（设置了'use strict'），this为undefined

（4）当使用call，apply，bind（ES5新增）绑定的，this指向绑定对象

二、ES6箭头函数中this

（1）默认指向定义它时，所处上下文的对象的this指向。即ES6箭头函数里this的指向就是上下文里对象this指向。

（2）即使是call，apply，bind等方法也不能改变箭头函数this的指向

1. HTTP状态码有哪些？分别有哪些含义？

200 OK ：请求正常处理完毕

204 No Content ：请求成功处理，没有实体的主体返回

301 Moved Permanently ：永久重定向，资源已永久分配新URI

302 Found ：临时重定向，资源已临时分配新URI

400 Bad Request ：请求报文语法错误或参数错误

401 Unauthorized ：要通过HTTP认证，或认证失败

403 Forbidden ：请求资源被拒绝

404 Not Found ：无法找到请求资源（服务器无理由拒绝）

500 Internal Server Error ：服务器故障或Web应用故障

503 Service Unavailable ：服务器超负载或停机维护

分类描述：

1\*\*信息，服务器收到请求，需要请求者继续执行操作

2\*\*成功，操作被成功接收并处理

3\*\*重定向，需要进一步的操作以完成请求

4\*\*客户端错误，请求包含语法错误或无法完成请求

5\*\*服务器错误，服务器在处理请求的过程中发生了错误

1. Promise如何使用？有哪些状态和方法？

promise 有三个状态：

1、pending[待定]初始状态

2、fulfilled[实现]操作成功

3、rejected[被否决]操作失败  
Promise 对象的状态改变，只有两种可能： 从 pending 变为 fulfilled。从 pending 变为 rejected。

Promise的参数：

Promise接受一个函数作为参数，该函数的两个参数分别是resolve和reject。这两个函数就是回调函数

resolve函数：在异步操作成功时调用，并将异步操作的结果，作为参数传递出去；

reject函数：在异步操作失败时调用，并将异步操作报出的错误，作为参数传递出去。

1. ES5和ES6的数组操作方法有哪些？

1.ES5中新增的数组扩展方法

//会改变原数组

push()和pop()

shift() 和 unshift()

sort()

reverse()

splice()

//不会改变原数组

concat()

slice()

join()

indexOf()和 lastIndexOf() （ES5新增）

forEach() （ES5新增）

map() （ES5新增）

filter() （ES5新增）

every() （ES5新增）

some() （ES5新增）

reduce()和 reduceRight() （ES5新增）

2. ES6中新增的数组扩展方法

1.Array.from()，将数组的对象和iterable对象，转为真正的数组

2.Array.of() 用来将一组值，转为数组

3.copyWithin(target,start,end),把指定位置的成员复制到指定位置

4.find 找到第一个符合条件的元素，并返回元素

5.findIndex 找到第一个符合条件的元素，并返回位置

6.fill(),填充数组

7.keys() 返回一个可迭代对象，数组中keys()就是下标，对象就是对象名

9.entries() 返回一个可迭代对象 ，类似python中的{}.items()

10.includes() 数组中是否包含某个特定的值

1. Webpack的优点，写出你所知道的Webpack的配置？

Webpack的优点：

1. 代码转换: typeScript 编译成 JavaScript、scss，less 编译成 css.
2. 文件优化：压缩 javaScript、css、html 代码，压缩合并图片。
3. 代码分割：提取多个页面的公共代码、提取首屏不需要执行部分的代码让其异步加载。
4. 模块合并：在采用模块化的项目里会有很多个模块和文件，需要构建功能把模块分类合并成一个文件。
5. 自动刷新：监听本地源代码的变化，自动重新构建、刷新浏览器。
6. 扩展性强，插件机制完善。

Webpack打包过程：

1. 利用babel完成代码转换,并生成单个文件的依赖
2. 从入口开始递归分析，并生成依赖图谱
3. 将各个引用模块打包为一个立即执行函数
4. 将最终的bundle文件写入bundle.js中

Webpack 的四大核心：

entry：js 入口源文件

output：生成文件

loader：进行文件处理

plugins：插件，比 loader 更强大，能使用更多 webpack 的 api

1. Vue的生命周期钩子函数有哪些？

beforeCreate  --------- 组件实例化之前

created  --------- 组件实例化完毕，单页面还未显示

beforeMount  --------- 组件挂载前，页面仍未展示，但虚拟DOM已经配置

mounted  --------- 组件挂载后，此方法执行后，页面显示

beforeUpdate  --------- 组件更新前，页面仍未更新，但虚拟Dom已经配置

updated  --------- 组件更新，此方法执行后，页面显示

beforeDestroy  --------- 组件销毁前

destroyed  --------- 组件销毁

1. 列举一下你知道的Vue通信方式有哪些？

1.props/$emit

概念：父组件通过props的方式向子组件传递数据，而通过$emit 子组件可以向父组件通信。

2.$children/$parent

概念： 通过$parent和$children就可以访问组件的实例。

使用 this.$parent查找当前组件的父组件。  
使用 this.$children查找当前组件的直接子组件，可以遍历全部子组件， 需要注意 $children 并不保证顺序，也不是响应式的。

3.provide/ inject

概念：provide/ inject 是vue2.2.0新增的api, 简单来说就是父组件中通过provide来提供变量, 然后再子组件中通过inject来注入变量。

4.ref/ refs

概念：ref：如果在普通的 DOM 元素上使用，引用指向的就是 DOM 元素；如果用在子组件上，引用就指向组件实例，可以通过实例直接调用组件的方法或访问数据

5.eventBus

概念：eventBus 又称为事件总线，在vue中可以使用它来作为沟通桥梁的概念, 就像是所有组件共用相同的事件中心，可以向该中心注册发送事件或接收事件， 所以组件都可以通知其他组件。

6.Vuex

概念：

Vuex 是一个专为 Vue.js 应用程序开发的状态管理模式。它采用集中式存储管理应用的所有组件的状态，并以相应的规则保证状态以一种可预测的方式发生变化. Vuex 解决了多个视图依赖于同一状态和来自不同视图的行为需要变更同一状态的问题，将开发者的精力聚焦于数据的更新而不是数据在组件之间的传递上。

7.localStorage / sessionStorage

8.attrs/$listeners

概念：是vue2.4.0引入的该方法，为了解决跨级的组件

常见使用场景可以分为三类:

父子组件通信: props; $parent / $children; provide / inject ; ref ; $attrs / $listeners

兄弟组件通信: eventBus ; vuex

跨级通信: eventBus；Vuex；provide / inject 、$attrs / $listeners

1. 从输入URL到页面渲染发生了哪些？
2. 域名解析
3. 发起TCP三次握手建立连接
4. 建立连接后发起http请求
5. 服务器响应请求，浏览器获取html源码
6. 浏览器解析html代码，并请求相关css,js和图片资源
7. 浏览器渲染页面