1. css实现垂直居中？
2. 子元素设置margin：auto
3. 子元素设置 position: absolute和margin负值{width：100px；position：absolute；left：50%；margin-left：-50px}
4. 子元素设置position: absolute和margin:auto；{position: absolute;left: 0; margin: auto;}
5. 子元素position: absolute和transform: translateX(-50%){position: absolute;left: 50%; transform: translateX(-50%);}
6. 父元素设置dispaly:flex;justify-content: center;align-content: center
7. display：flex属性有哪些？

1、flex-direction　　容器内项目的排列方向(默认横向排列)

2、flex-wrap　　容器内项目换行方式

3、flex-flow　　以上两个属性的简写方式

4、justify-content　　项目在主轴上的对齐方式

5、align-items　　项目在交叉轴上如何对齐

6、align-content　　定义了多根轴线的对齐方式。如果项目只有一根轴线，该属性不起作用。

1、order　　项目的排列顺序。数值越小，排列越靠前，默认为0。

2、flex-grow　　项目的放大比例，默认为0，即如果存在剩余空间，也不放大。

3、flex-shrink　　项目的缩小比例，默认为1，即如果空间不足，该项目将缩小。

4、flex-basis　　在分配多余空间之前，项目占据的主轴空间（main size）。浏览器根据这个属性，计算主轴是否有多余空间。它的默认值为auto，即项目的本来大小。

5、flex　　是flex-grow, flex-shrink 和 flex-basis的简写，默认值为0 1 auto。后两个属性可选。

6、align-self　　允许单个项目有与其他项目不一样的对齐方式，可覆盖align-items属性。默认值为auto，表示继承父元素的align-items属性，如果没有父元素，则等同于stretch。

1. es6新语法

[1.let const var比较](https://www.kancloud.cn/lemon-m/js-html-css/1029371)  
[2.反引号（`）标识](https://www.kancloud.cn/lemon-m/js-html-css/1029377)  
[3.函数默认参数](https://www.kancloud.cn/lemon-m/js-html-css/1029378)  
[4.箭头函数](https://www.kancloud.cn/lemon-m/js-html-css/1029379)  
[5.属性简写](https://www.kancloud.cn/lemon-m/js-html-css/1029380) （key和变量名一样）  
[6.方法简写](https://www.kancloud.cn/lemon-m/js-html-css/1029381) (省略function关键字)  
[7.Object.keys()方法，获取对象的所有属性名或方法名](https://www.kancloud.cn/lemon-m/js-html-css/1029382)（只遍历自身的可枚举属性）  
[8.Object.assign ()原对象的属性和方法都合并到了目标对象](https://www.kancloud.cn/lemon-m/js-html-css/1029383)  
[9.for...of 循环](https://www.kancloud.cn/lemon-m/js-html-css/1029384)（只要部署Iterator接口的数据结构，都可以用for...of来遍历）

和for in的区别

for...of 语句遍历可迭代对象（包括数组、Set 和 Map 结构、arguments 对象、DOM NodeList 对象、字符串等）

for...of 可以与 break、continue 和 return 配合使用

for...in 遍历键名，for...of 遍历键值，for...in循环主要是为遍历对象而设计的，不适用于遍历数组

### for...in 会把对象上手动添加的属性和原型上的属性暴露出来

for…in （for…in遍历，只要在遍历对象和对象所在的原型链上的可枚举属性都会被遍历）

[10.import和export](https://www.kancloud.cn/lemon-m/js-html-css/1029385)  
[11.Promise对象](https://www.kancloud.cn/lemon-m/js-html-css/1029387)

ES6 规定，Promise对象 是一个构造函数，用来生成Promise实例。

**new Promise((resolve,reject) =>{ this.simpleGet(apiOnlineConsultationPlatform.getInfoByUid,**

**null,(res) => { console.log("获取当前的的layer",res); resolve(); }**

**) })**

**.then(() =>{ console.log("获取行政层级"); })**

**.then(() =>{ console.log("获取组织"); })**

then方法可以接受两个回调函数作为参数。第一个回调函数是Promise对象的状态变为resolved时调用，第二个回调函数是Promise对象的状态变为rejected时调用。其中，第二个函数是可选的，不一定要提供。这两个函数都接受Promise对象传出的值作为参数。

上面的代码使用then方法，依次指定了两个回调函数。第一个回调函数完成以后，会将返回结果作为参数，传入第二个回调函数。

采用链式的then，可以指定一组按照次序调用的回调函数。这时，前一个回调函数，有可能返回的还是一个Promise对象（即有异步操作），这时后一个回调函数，就会等待该Promise对象的状态发生变化，才会被调用。

[12.解构赋值](https://www.kancloud.cn/lemon-m/js-html-css/1029389)  
[13.set数据结构（可用于快速去重）](https://www.kancloud.cn/lemon-m/js-html-css/1029390)  
[14.Spread Operator 展开运算符(...)](https://www.kancloud.cn/lemon-m/js-html-css/1029393)  
[15.字符串新增方法](https://www.kancloud.cn/lemon-m/js-html-css/1029608)

四、Asyn 和Await

执行async函数返回的是promise对象。内置执行器了。await命令后面，可以是 Promise 对象和原始类型的值（数值、字符串和布尔值，但这时会自动转成立即 resolved 的 Promise 对象）

进一步说，**async函数完全可以看作多个异步操作，包装成的一个 Promise 对象，而await命令就是内部then命令的语法糖。**

[https://note.youdao.com/web/#/file/WEB16fa270d30bc1699bb67d46dd6200582/note/WEB0fca835b43f927341ae12cb70433188e/](https://note.youdao.com/web/" \l "/file/WEB16fa270d30bc1699bb67d46dd6200582/note/WEB0fca835b43f927341ae12cb70433188e/)

五、图片加载

var img = new Image();

img.onload = function () { alert("img is loaded") };

img.onerror = function () { alert("error!") };

1. git常见命令
2. javaScript有三种数据存储方式

分别是：

sessionStorage

localStorage

cookie

相同点：都保存在浏览器端，同源的

不同点：

①传递方式不同

cookie数据始终在同源的http请求中携带（即使不需要），即cookie在浏览器和服务器间来回传递。

sessionStorage和localStorage不会自动把数据发给服务器，仅在本地保存。

②数据大小不同

cookie数据还有路径（path）的概念，可以限制cookie只属于某个路径下。

存储大小限制也不同，cookie数据不能超过4k，同时因为每次http请求都会携带cookie，所以cookie只适合保存很小的数据，如会话标识。

sessionStorage和localStorage 虽然也有存储大小的限制，但比cookie大得多，可以达到5M或更大。

③数据有效期不同

sessionStorage：仅在当前浏览器窗口关闭前有效，自然也就不可能持久保持；

localStorage：始终有效，窗口或浏览器关闭也一直保存，因此用作持久数据；

cookie只在设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭。

④作用域不同

sessionStorage不在不同的浏览器窗口中共享，即使是同一个页面；(如果一个标签页包含多个iframe标签且他们属于同源页面,那么他们之间是可以共sessionStorag)

localStorage 在所有同源窗口中都是共享的；

cookie也是在所有同源窗口中都是共享的。

Web Storage 支持事件通知机制，可以将数据更新的通知发送给监听者。

Web Storage 的 api 接口使用更方便。

八、http缓存

强缓存；

Response**：cache-Control：max-age = 60；**

Espires



协商缓存

**文件最后修改时间字段：**

Response：Last-Modified

Request：If-Modified-Since

**文件内容标识字段：(如果文件修改了，改来改去还是恢复原来的内容,)**

**Response：E-tag**

**Request：If-None-Match**



**最佳优化策略——消灭304**

最佳优化策略：因为协商缓存本身也有http请求的损耗，所以最佳优化策略是要尽可能的将静态文件存储为较长的时间，多利用强缓存而不是协商缓存，即消灭304。

#### **一、什么是闭包**

1、函数里面包含的子函数，子函数访问父函数的局部变量  
2、通过return将子函数暴露在全局作用域，子函数就形成闭包  
3、通过闭包，父函数的局部变量没有被销毁，可通过闭包去调用 ，但同时，这个局部变量也不会被全局变量污染

#### **二、简单的闭包**

function aaa() {

var a = 0;

return function () {

alert(a++);

};

}

var fun = aaa();

fun(); //1

//简单点就是return一个函数

#### **三、闭包的优点和缺点**

优点： 避免全局变量的污染，同时，局部变量没有被销毁，驻留在内存中，还可以被访问  
缺点： 使用不当，会造成内存泄露

主要的生命周期函数分类：

 - 创建期间的生命周期函数：

      + beforeCreate：实例刚在内存中被创建出来，此时，还没有初始化好 data 和 methods 属性

      + created：实例已经在内存中创建OK，此时 data 和 methods 已经创建OK，此时还没有开始 编译模板

      + beforeMount：此时已经完成了模板的编译，但是还没有挂载到页面中

      + mounted：此时，已经将编译好的模板，挂载到了页面指定的容器中显示

 - 运行期间的生命周期函数：

     + beforeUpdate：状态更新之前执行此函数， 此时 data 中的状态值是最新的，但是界面上显示的 数据还是旧的，因为此时还没有开始重新渲染DOM节点

     + updated：实例更新完毕之后调用此函数，此时 data 中的状态值 和 界面上显示的数据，都已经完成了更新，界面已经被重新渲染好了！

 - 销毁期间的生命周期函数：

     + beforeDestroy：实例销毁之前调用。在这一步，实例仍然完全可用。

     + destroyed：Vue 实例销毁后调用。调用后，Vue 实例指示的所有东西都会解绑定，所有的事件监听器会被移除，所有的子实例也会被销毁。