# 全栈开发面试题

## 一、****选择题****

* 1、以下选项中，不是页面布局模型的是：a
  1. **盒子模型**
  2. 层模型
  3. 流动模型
  4. 浮动模型

* 2、在CSS样式中，如果想让a标签继承父级标签的颜色，属性color的值为：a
  1. **inherit**
  2. initial
  3. implement
  4. extend

* 3、以下值会被当作假（false）值的有：c
  1. !!’0’
  2. False == ‘false’
  3. **!!0**
  4. 字符串”false”

* 4、假设：var Arr = [“g”,”i”,”n”,”s”,”a”,”b”]；下列哪些方法可以返回数组：b
  1. join
  2. **slice**
  3. unshift
  4. push
* 5、以下哪些规范是JavaScript模块化编程规范：abd
  1. **AMD**
  2. **CMD**
  3. PMD
  4. **CommonJS**

* 6、以下关于JavaScript字符和字符串的断言，正确的有：cd
  1. JavaScript没有字符串类型
  2. JavaScript中的所有字符都是16位的Unicode编码
  3. **反斜线符号（\）是JavaScript中唯一的转义字符**
  4. **字符串是不可改变的，一旦被创建，就永远无法改变**
* 7、以下关于JavaScript运算符的断言，正确的有：acd
  1. **使用delete运算符并不会删除原型链中的任何对象**
  2. 尝试检索一个undefined值将会导致TypeError异常，可以通过“||”运算符来避免错误
  3. **使用delete运算符并不会删除通过var定义的对象**
  4. **在JavaScript中，关于属性索取“.”、“[]”及函数调用“()”这三个运算符相较于其它运算符优先级最高**
* 8、以下关于Web应用性能优化的断言，正确的有：bc
  1. 下载单个100KB的文件将比下载4个25KB的文件慢，因为并发请求比逐个请求快
  2. **当页面存在大量元素时，不要将事件绑定在单独的元素中，而应该使用事件委托或事件代理技术，如使用事件冒泡（Bubble）原理将事件都绑定在最外层元素中**
  3. **使用JavaScript实现的动画效果，应该使用绝对定位，让元素脱离DOM数，以减少重排**
  4. 一般浏览器使用并行方式下载CSS文件，因此动态加载CSS文件并没有太大的网络性能优化的意义
* **9、关于flex-direction属性值错误的是（a）。**
  1. col
  2. row
  3. row-reverse
  4. column-reverse
* **10、 如果让一个元素在pc端显示而在手机端隐藏，下列选项正确的是（b）。**
  1. visible-xs-8  hidden-md
  2. visible-md-8 hidden-xs
  3. visible-md-8 hidden-sm
  4. visible-sm-8 hidden-md

## ****二、简答****

**1、H5的新特性有哪些？**

用于绘画的 canvas 元素 Websocket, <video> <audio> <header> <footer> 正则表达式

HTML5添加了许多新新的语法特征，其中包括**<video>、<audio>和<canvas>元素**，同时集成了[SVG](http://www.bing.com/knows/SVG_%E5%8F%AF%E7%BC%A9%E6%94%BE%E7%9F%A2%E9%87%8F%E5%9B%BE%E5%BD%A2)内容。

**2、scss是什么？在vue.cli中的安装使用步骤是？有哪几大特性？**

答：css的预编译。

使用步骤：

第一步：用npm 下三个loader（sass-loader、css-loader、node-sass）

第二步：在build目录找到webpack.base.config.js，在那个extends属性中加一个拓展.scss

第三步：还是在同一个文件，配置一个module属性

第四步：然后在组件的style标签加上lang属性 ，例如：lang=”scss”

有哪几大特性:

1、可以用变量，例如（$变量名称=值）；

2、可以用混合器，例如（）

3、可以嵌套

**3、页面导入样式时，使用link和@import有什么区别？**  
（1）link属于XHTML标签，除了加载CSS外，还能用于定义RSS, 定义rel连接属性等作用；而@import是CSS提供的，只能用于加载CSS;  
（2）页面被加载的时，link会同时被加载，而@import引用的CSS会等到页面被加载完再加载;  
（3）import是CSS2.1 提出的，只在IE5以上才能被识别，而link是XHTML标签，无兼容问题;

**4、请描述一下 cookies，sessionStorage 和 localStorage 的区别？**

cookie是网站为了标示用户身份而储存在用户本地终端（Client Side）上的数据（通常经过加密）。  
cookie数据始终在同源的http请求中携带（即使不需要），记会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。

存储大小：  
cookie数据大小不能超过4k。  
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。

有期时间：  
localStorage 存储持久数据，浏览器关闭后数据不丢失除非主动删除数据；  
sessionStorage 数据在当前浏览器窗口关闭后自动删除。  
cookie 设置的cookie过期时间之前一直有效，即使窗口或浏览器关闭

**5、箭头有哪些新特点？**

* 不需要function关键字来创建函数
* 省略return关键字
* 继承当前上下文的 this 关键字
* 没有arguments参数

## **三、操作题**

**1、把以下代码使用两种方法，来依次输出0到9？**

var funcs = []

for (var i = 0; i < 10; i++) {

funcs.push(function() { console.log(i) })

}

funcs.forEach(function(func) {

func()

})

  答：分别使用es5的闭包和es6的let

// ES5告诉我们可以利用闭包解决这个问题

var funcs = []

for (var i = 0; i < 10; i++) {

func.push((function(value) {

return function() {

console.log(value)

}

}(i)))

}

// es6

for (let i = 0; i < 10; i++) {

func.push(function() {

console.log(i)

})

}

**2、快速排序**

var arr = [3, 1, 4, 6, 5, 7, 2];

function quickSort(arr) {

if(arr.length == 0) {

return []; // 返回空数组

}

var cIndex = Math.floor(arr.length / 2);

var c = arr.splice(cIndex, 1);

var l = [];

var r = [];

for (var i = 0; i < arr.length; i++) {

if(arr[i] < c) {

l.push(arr[i]);

} else {

r.push(arr[i]);

}

}

return quickSort(l).concat(c, quickSort(r));

}

console.log(quickSort(arr));

**3、Vue的双向数据绑定原理是什么？**

答：vue.js 是采用数据劫持结合发布者-订阅者模式的方式，通过Object.defineProperty()来劫持各个属性的setter，getter，在数据变动时发布消息给订阅者，触发相应的监听回调。

具体步骤：

第一步：需要observe的数据对象进行递归遍历，包括子属性对象的属性，都加上 setter和getter

这样的话，给这个对象的某个值赋值，就会触发setter，那么就能监听到了数据变化

第二步：compile解析模板指令，将模板中的变量替换成数据，然后初始化渲染页面视图，并将每个指令对应的节点绑定更新函数，添加监听数据的订阅者，一旦数据有变动，收到通知，更新视图

第三步：Watcher订阅者是Observer和Compile之间通信的桥梁，主要做的事情是:

1、在自身实例化时往属性订阅器(dep)里面添加自己

2、自身必须有一个update()方法

3、待属性变动dep.notice()通知时，能调用自身的update()方法，并触发Compile中绑定的回调，则功成身退。

第四步：MVVM作为数据绑定的入口，整合Observer、Compile和Watcher三者，通过Observer来监听自己的model数据变化，通过Compile来解析编译模板指令，最终利用Watcher搭起Observer和Compile之间的通信桥梁，达到数据变化 -> 视图更新；视图交互变化(input) -> 数据model变更的双向绑定效果。

**4、使用过element.ui吗？说下它其中两个组件的使用方法？(答案不固定)**

使用过用过一个布局的，它是由24份，它的写法是:span后面带的数字它占24份里面的宽度。:offset是它的间距，后面也是跟数字，也是从24份里面取的。

input按钮，标签是el-input，后面type跟上一个属性就是显示不同按钮的类型，有默认的default

（默认的）、success（成功的）、warning（警告）、danger（危险）、info（）、primary（）

**5、说下你了解的axios相关配置属性？**

`url`是用于请求的服务器URL

`method`是创建请求时使用的方法,默认是get

`baseURL`将自动加在`url`前面，除非`url`是一个绝对URL。它可以通过设置一个`baseURL`便于为axios实例的方法传递相对URL

`transformRequest`允许在向服务器发送前，修改请求数据，只能用在'PUT','POST'和'PATCH'这几个请求方法

`headers`是即将被发送的自定义请求头

headers:{'X-Requested-With':'XMLHttpRequest'},

`params`是即将与请求一起发送的URL参数，必须是一个无格式对象(plainobject)或URLSearchParams对象

params:{

ID:12345

},

`auth`表示应该使用HTTP基础验证，并提供凭据

这将设置一个`Authorization`头，覆写掉现有的任意使用`headers`设置的自定义`Authorization`头

auth:{

username:'janedoe',

password:'s00pers3cret'

},

'proxy'定义代理服务器的主机名称和端口

`auth`表示HTTP基础验证应当用于连接代理，并提供凭据

这将会设置一个`Proxy-Authorization`头，覆写掉已有的通过使用`header`设置的自定义`Proxy-Authorization`头。

proxy:{

host:'127.0.0.1',

port:9000,

auth::{

username:'mikeymike',

password:'rapunz3l'

}

},