以下是收集一些面试中经常会遇到的经典面试题以及自己面试过程中有一些未解决的问题，通过对知识的整理以及经验的总结，重新巩固自身的前端基础知识，如有错误或更好的答案，欢迎指正，水平有限，望各位不吝指教。:）。(原文链接：<http://www.cnblogs.com/bigboyLin/p/5272902.html>)

**HTML/CSS部分**

**1、什么是盒子模型？**

在网页中，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），元素的边框（border），元素的外边距（margin）四个部分。这四个部分占有的空间中，有的部分可以显示相应的内容，而有的部分只用来分隔相邻的区域或区域。4个部分一起构成了css中元素的盒模型。

**2、行内元素有哪些？块级元素有哪些？ 空(void)元素有那些？**

行内元素：a、b、span、img、input、strong、select、label、em、button、textarea 块级元素：div、ul、li、dl、dt、dd、p、h1-h6、blockquote 空元素：即系没有内容的HTML元素，例如：br、meta、hr、link、input、img

**3、CSS实现垂直水平居中**

一道经典的问题，实现方法有很多种，以下是其中一种实现： HTML结构：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4 | &lt;div class="wrapper"&gt;          &lt;div class="content"&gt;&lt;/div&gt;      &lt;/div&gt; |

CSS：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | .wrapper{position:relative;}      .content{          background-color:#6699FF;          width:200px;          height:200px;          position: absolute;        //父元素需要相对定位          top: 50%;          left: 50%;          margin-top:-100px ;   //二分之一的height，width          margin-left: -100px;      } |

**4、简述一下src与href的区别**

href 是指向网络资源所在位置，建立和当前元素（锚点）或当前文档（链接）之间的链接，用于超链接。

src是指向外部资源的位置，指向的内容将会嵌入到文档中当前标签所在位置；在请求src资源时会将其指向的资源下载并应用到文档内，例如js脚本，img图片和frame等元素。当浏览器解析到该元素时，会暂停其他资源的下载和处理，直到将该资源加载、编译、执行完毕，图片和框架等元素也如此，类似于将所指向资源嵌入当前标签内。这也是为什么将js脚本放在底部而不是头部。

**5、什么是CSS Hack?**

一般来说是针对不同的浏览器写不同的CSS,就是 CSS Hack。 IE浏览器Hack一般又分为三种，条件Hack、属性级Hack、选择符Hack（详细参考CSS文档：[css文档](http://www.phpstudy.net/css3/)）。例如：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | // 1、条件Hack     &lt;!--[if IE]&gt;        &lt;style&gt;              .test{color:red;}        &lt;/style&gt;     &lt;![endif]--&gt;     // 2、属性Hack      .test{      color:#090\9; /\* For IE8+ \*/      \*color:#f00;  /\* For IE7 and earlier \*/      \_color:#ff0;  /\* For IE6 and earlier \*/      }     // 3、选择符Hack      \* html .test{color:#090;}       /\* For IE6 and earlier \*/      \* + html .test{color:#ff0;}     /\* For IE7 \*/ |

**6、简述同步和异步的区别**

同步是阻塞模式，异步是非阻塞模式。 同步就是指一个进程在执行某个请求的时候，若该请求需要一段时间才能返回信息，那么这个进程将会一直等待下去，直到收到返回信息才继续执行下去； 异步是指进程不需要一直等下去，而是继续执行下面的操作，不管其他进程的状态。当有消息返回时系统会通知进程进行处理，这样可以提高执行的效率。

**7、px和em的区别**

px和em都是长度单位，区别是，px的值是固定的，指定是多少就是多少，计算比较容易。em得值不是固定的，并且em会继承父级元素的字体大小。 浏览器的默认字体高都是16px。所以未经调整的浏览器都符合: 1em=16px。那么12px=0.75em, 10px=0.625em

**8、什么叫优雅降级和渐进增强？**

渐进增强 progressive enhancement： 针对低版本浏览器进行构建页面，保证最基本的功能，然后再针对高级浏览器进行效果、交互等改进和追加功能达到更好的用户体验。

优雅降级 graceful degradation： 一开始就构建完整的功能，然后再针对低版本浏览器进行兼容。

区别： a. 优雅降级是从复杂的现状开始，并试图减少用户体验的供给 b. 渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要 c. 降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带

**9、浏览器的内核分别是什么?**

IE: trident内核 Firefox：gecko内核 Safari：webkit内核 Opera：以前是presto内核，Opera现已改用Google Chrome的Blink内核 Chrome：Blink(基于webkit，Google与Opera Software共同开发)

**JavaScript部分**

**怎样添加、移除、移动、复制、创建和查找节点？**

1）创建新节点 createDocumentFragment() //创建一个DOM片段 createElement() //创建一个具体的元素 createTextNode() //创建一个文本节点

2）添加、移除、替换、插入 appendChild() //添加 removeChild() //移除 replaceChild() //替换 insertBefore() //插入

3）查找 getElementsByTagName() //通过标签名称 getElementsByName() //通过元素的Name属性的值 getElementById() //通过元素Id，唯一性

**实现一个函数clone，可以对JavaScript中的5种主要的数据类型（包括Number、String、Object、Array、Boolean）进行值复制。**

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43 | /\*\*  \* 对象克隆  \* 支持基本数据类型及对象  \* 递归方法  \*/  function clone(obj) {      var o;      switch (typeof obj) {          case "undefined":              break;          case "string":              o = obj + "";              break;          case "number":              o = obj - 0;              break;          case "boolean":              o = obj;              break;          case "object": // object 分为两种情况 对象（Object）或数组（Array）              if (obj === null) {                  o = null;              } else {                  if (Object.prototype.toString.call(obj).slice(8, -1) === "Array") {                      o = [];                      for (var i = 0; i &lt; obj.length; i++) {                          o.push(clone(obj[i]));                      }                  } else {                      o = {};                      for (var k in obj) {                          o[k] = clone(obj[k]);                      }                  }              }              break;          default:              o = obj;              break;      }      return o;  } |

**如何消除一个数组里面重复的元素？**

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | // 方法一：  var arr1 =[1,2,2,2,3,3,3,4,5,6],      arr2 = [];  for(var i = 0,len = arr1.length; i&lt; len; i++){      if(arr2.indexOf(arr1[i]) &lt; 0){          arr2.push(arr1[i]);      }  }  document.write(arr2); // 1,2,3,4,5,6 |

**在Javascript中什么是伪数组？如何将伪数组转化为标准数组？**

伪数组（类数组）：无法直接调用数组方法或期望length属性有什么特殊的行为，但仍可以对真正数组遍历方法来遍历它们。典型的是函数的argument参数，还有像调用getElementsByTagName,document.childNodes之类的,它们都返回NodeList对象都属于伪数组。可以使用Array.prototype.slice.call(fakeArray)将数组转化为真正的Array对象。

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | function log(){        var args = Array.prototype.slice.call(arguments);  //为了使用unshift数组方法，将argument转化为真正的数组        args.unshift('(app)');          console.log.apply(console, args);  }; |

**Javascript中callee和caller的作用？**

caller是返回一个对函数的引用，该函数调用了当前函数；

callee是返回正在被执行的function函数，也就是所指定的function对象的正文。

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

sessionStorage用于本地存储一个会话（session）中的数据，这些数据只有在同一个会话中的页面才能访问并且当会话结束后数据也随之销毁。因此sessionStorage不是一种持久化的本地存储，仅仅是会话级别的存储。而localStorage用于持久化的本地存储，除非主动删除数据，否则数据是永远不会过期的。

web storage和cookie的区别

Web Storage的概念和cookie相似，区别是它是为了更大容量存储设计的。Cookie的大小是受限的，并且每次你请求一个新的页面的时候Cookie都会被发送过去，这样无形中浪费了带宽，另外cookie还需要指定作用域，不可以跨域调用。 除此之外，Web Storage拥有setItem,getItem,removeItem,clear等方法，不像cookie需要前端开发者自己封装setCookie，getCookie。但是Cookie也是不可以或缺的：Cookie的作用是与服务器进行交互，作为HTTP规范的一部分而存在 ，而Web Storage仅仅是为了在本地“存储”数据而生。

**统计字符串中字母个数或统计最多字母数。**

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | var str = "aaaabbbccccddfgh";  var obj  = {};  for(var i=0;i&lt;str.length;i++){      var v = str.charAt(i);      if(obj[v] &amp;&amp; obj[v].value == v){          obj[v].count = ++ obj[v].count;      }else{          obj[v] = {};          obj[v].count = 1;          obj[v].value = v;      }  }  for(key in obj){      document.write(obj[key].value +'='+obj[key].count+' '); // a=4  b=3  c=4  d=2  f=1  g=1  h=1  } |

**jQuery的事件委托方法on、live、delegate之间有什么区别？**

**HTTP**

**一次完整的HTTP事务是怎样的一个过程？**

基本流程： a. 域名解析 b. 发起TCP的3次握手 c. 建立TCP连接后发起http请求 d. 服务器端响应http请求，浏览器得到html代码 e. 浏览器解析html代码，并请求html代码中的资源 f. 浏览器对页面进行渲染呈现给用户

**HTTP的状态码有哪些？**

**HTTPS是如何实现加密？**

**算法相关**

**手写数组快速排序**

关于快排算法的详细说明，可以参考阮一峰老师的文章[快速排序](http://www.ruanyifeng.com/blog/2011/04/quicksort_in_javascript.html) “快速排序”的思想很简单，整个排序过程只需要三步： （1）在数据集之中，选择一个元素作为”基准”（pivot）。 （2）所有小于”基准”的元素，都移到”基准”的左边；所有大于”基准”的元素，都移到”基准”的右边。 （3）对”基准”左边和右边的两个子集，不断重复第一步和第二步，直到所有子集只剩下一个元素为止。 参考代码：

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | var quickSort = function(arr) {  　　if (arr.length &lt;= 1) { return arr; }  　　var pivotIndex = Math.floor(arr.length / 2);  　　var pivot = arr.splice(pivotIndex, 1)[0];  　　var left = [];  　　var right = [];  　　for (var i = 0; i &lt; arr.length; i++){  　　　　if (arr[i] &lt; pivot) {  　　　　　　left.push(arr[i]);  　　　　} else {  　　　　　　right.push(arr[i]);  　　　　}  　　}  　　return quickSort(left).concat([pivot], quickSort(right));  }; |

**JavaScript实现二分法查找**

二分法查找，也称折半查找，是一种在有序数组中查找特定元素的搜索算法。查找过程可以分为以下步骤： （1）首先，从有序数组的中间的元素开始搜索，如果该元素正好是目标元素（即要查找的元素），则搜索过程结束，否则进行下一步。 （2）如果目标元素大于或者小于中间元素，则在数组大于或小于中间元素的那一半区域查找，然后重复第一步的操作。 （3）如果某一步数组为空，则表示找不到目标元素。 参考代码:

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | // 非递归算法          function binary\_search(arr, key) {              var low = 0,                  high = arr.length - 1;              while(low &lt;= high){                  var mid = parseInt((high + low) / 2);                  if(key == arr[mid]){                      return  mid;                  }else if(key &gt; arr[mid]){                      low = mid + 1;                  }else if(key &lt; arr[mid]){                      high = mid -1;                  }else{                      return -1;                  }              }          };          var arr = [1,2,3,4,5,6,7,8,9,10,11,23,44,86];          var result = binary\_search(arr,10);          alert(result); // 9 返回目标元素的索引值 |

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21 | // 递归算法          function binary\_search(arr,low, high, key) {              if (low &gt; high){                  return -1;              }              var mid = parseInt((high + low) / 2);              if(arr[mid] == key){                  return mid;              }else if (arr[mid] &gt; key){                  high = mid - 1;                  return binary\_search(arr, low, high, key);              }else if (arr[mid] &lt; key){                  low = mid + 1;                  return binary\_search(arr, low, high, key);              }          };          var arr = [1,2,3,4,5,6,7,8,9,10,11,23,44,86];          var result = binary\_search(arr, 0, 13, 10);          alert(result); // 9 返回目标元素的索引值 |

**Web安全**

**你所了解到的Web攻击技术**

（1）XSS（Cross-Site Scripting，跨站脚本攻击）：指通过存在安全漏洞的Web网站注册用户的浏览器内运行非法的HTML标签或者JavaScript进行的一种攻击。 （2）SQL注入攻击 （3）CSRF（Cross-Site Request Forgeries，跨站点请求伪造）：指攻击者通过设置好的陷阱，强制对已完成的认证用户进行非预期的个人信息或设定信息等某些状态更新。

**前端性能**

**如何优化图像、图像格式的区别**

**浏览器是如何解析HTML**

**前端架构**

**对MVC、MVVM的理解**

**正则表达式相关**

**写一个function，清除字符串前后的空格。（兼容所有浏览器）**

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4  5  6 | function trim(str) {      if (str &amp;&amp; typeof str === "string") {          return str.replace(/(^\s\*)|(\s\*)$/g,""); //去除前后空白符      }  } |

**使用正则表达式验证邮箱格式**

JavaScript



|  |  |
| --- | --- |
| 1  2  3  4 | var reg = /^(\w)+(\.\w+)\*@(\w)+((\.\w{2,3}){1,3})$/;      var email = "example@qq.com";      console.log(reg.test(email));  // true |

**职业规划**

**对前端工程师这个职位你是怎么样理解的？**

a. 前端是最贴近用户的程序员，前端的能力就是能让产品从 90分进化到 100 分，甚至更好 b. 参与项目，快速高质量完成实现效果图，精确到1px； c. 与团队成员，UI设计，产品经理的沟通； d. 做好的页面结构，页面重构和用户体验； e. 处理hack，兼容、写出优美的代码格式； f. 针对服务器的优化、拥抱最新前端技术。