2019年前端面试题总结以及一些面试心得（附答案）

html中文网 html中文网 昨天

前沿

小编在这几年的前端开发过程中，经历了由js,jquery到vue,由操作dom到数据驱动页面，由只写pc网站，到写移动端网页，再到移动端app,再到微信公众号，小程序，可谓是风云变幻，不得不感叹H5的能力越来越强，正所谓学到老，活到老大概说的就是程序员吧。在悲催的经历了互联网公司一家又一家倒闭，带来的是我也经历了无数次的面试，也算是系统的过了一遍知识体系。

在这过程中，发现了好多自己的不足之处，于是总结下来，以方便后续学习。可能还有一些不全面对的地方，希望朋友看见可以在评论区指正，补充，谢谢。

注释：随着前端技术的发展，本篇简文会持续更新，请关注更新。

> 随着前端的发展，涌现出越来越多的框架,知识体系，所以小伙伴们学习的路还很长。就目前状况来看，前端面试分为以下几大类。
>
> 1、css3
> 2、javaScript
> 3、jquery
> 4、HTML5新特性和语义化
> 5、Vue,React,Angular
> 6、ES6
> 7、Webpack
> 8、微信公众号，小程序，小游戏
> 9、Dcloud开发，一套代码多终端
> 10、大数据可视化
> 11、node.js
> 12、当然你要是会java或者其他语言，那就还有其他语言。

关键点一：CSS3

1，描述一下你认为的盒子模型?

> 答案：盒子模型分为ie盒子模型和w3c盒子模型

- ie的盒子模型包括（margin(外边距)，padding(内边距)，边框(border)，内容(content) (ie的width=content+padding+borde)）；

- w3c的盒子模型包括（margin(外边距)，padding(内边距)，边框(border)，内容(content) w3c的width=content)

2，css3的选择器有哪几种?

答案：*通用选择器，ID选择器 ，.类选择器class，标签选择器，标签组合选择器，伪类选择器：，+相邻元素选择器，>子元素选择器，~同辈选择器，

x[title]属性选择器[type="button"]，

x[href^="http"]匹配以href值为http打头的地址，

x[href$=".jpg"]匹配所有的图片链接，

input[type=checkbox]:checked{};选择checkbox为当前选中的那个标签。

> 伪类选择器：
>
> p:empty 选择没有子元素的每个 <p> 元素（包括文本节点）。
> :first-child 第一个
> :last-child 最后一个
> :nth-child(11) 1--11个
> x:first/x:after 在x选择器之前或者之后插入内容
>
> 伪元素选择器
>
> ::before ,::after 通过 css 模拟出来html标签的效果

3，请写出css3样式的优先级,并排序

答案：!important(权重最大)1000>内嵌样式（style=""）>内部样式>外联样式>@import url("url");

4，css3的新特性

答案：
1、@font-face加载字体样式；

2、文字渲染，text-decoration,text-fill-color:文字内部填充颜色，text-stroke-color:文字边框填充颜色，text-stroke-width:文字边界宽度。

3、css3的多列布局 Column-count：表示布局几列。Column-rule：表示列与列之间的间隔条的样式 Column-gap：表示列与列之间的间隔。

4、边框圆角的布局。border-radius:50px;

5、css3的渐变效果， （gradient）

6、css3的阴影效果图 （shadow反射和reflect反射效果）

7、css3的多背景图片

8、css3的动画效果 animation

5，行内元素有哪些？块级元素与那些？

答案：

- 块级元素：div/p/form/ul/li/ol/hr/fieldset/table/dd/dt/dl
- 行内元素：span/strong/a/b/em/img/input/lable/small/sub

6，为什么使用css3和div布局？

答案：
1，代码精简（没有本身自带的一些属性，容易设置样式）
2，解决了table表格的嵌套问题
3，速度问题（页面代码减少，增加了编写代码的速度）
4 ,对排名的影响，基于xhtml标准的div+css布局会更快的通过w3c标准

7，隐藏一个button的方法（2种），写出一个button的按钮（2种）

答案：

隐藏一个button的方法：

- display:none;
- visibility：hidden；

写出一个button的按钮：

- <input type = button>
- <button>这是一个按钮 </button>

8，请简述web开发中的兼容问题

答案：

1，浏览器默认的内外边距不一样，所以用通用选择器*设置margin和padding来设置。

2，块标签设置浮动后，有设置margin的情况下，在ie6下的margin比别的浏览器大。

3，img标签会有默认的间距，需要用浮动去设置

4，火狐浏览器中的点击事件和滚动事件需要加（event）来兼容

5，div里的内容，ie默认为居中，firefox默认为左对齐，需要用margin:0px auto来调节

6，css3的动画效果不兼容ie8以下。需要使用js去写动画。

7，ie6的双倍边距，比如设置了margin:10px,ie6中默认为20px，需要使用display:inline,block.

9，css3实现水平垂直居中------(特别注意，里边的固定还是不固定)

a,----定位方式（父元素宽高固定，子元素宽高固定）

```
<div>
       <div></div>
   </div>
 <style scoped>
  .Father{
    position: relative;
  }
  .children{
    width: 50px;
    height: 50px;
    position: absolute;
    top: 0px;
    bottom: 0px;
    left: 0px;
    right: 0px;
    margin: auto;
  }
</style>
```

b, ----flex布局方式（父元素宽高不固定，子元素宽高不固定）

```
   <div>
       <div></div>
   </div>
 <style scoped>
 .Father {
   border: 1px solid red;
   height: 100px;
   display: flex;
   justify-content: center;
   align-items: center;
 }
```

```
  .children {
    width: 50px;
    height: 50px;
    border: 1px solid blue;
  }
</style>
```

c, ----transform方式（父元素宽高不固定，子元素宽高不固定）

```
    <div>
        <div></div>
    </div>
  <style scoped>
  .Father {
    border: 1px solid red;
    height: 100px;
    position: relative;
  }
  .children {
    width: 50px;
    height: 50px;
    border: 1px solid blue;
    position: absolute;
    top: 50%;
    left: 50%;
    transform: translate(-50%, -50%);  /* 使用css3的transform来实现 */
  }
</style>
```

10，css3实现左侧固宽，右侧随着屏幕，右侧随着屏幕变化而变化

A, ----float+calc(css3新属性计算属性)方式

```
   <div>
      <div>我是左侧</div>
      <div>我是右侧</div>
    </div>
<style  rel="stylesheet/scss">
  .Father {
    border: 1px solid red;
    height: 100px;
  }
  .LeftChildren {
    width: 50px;
    border: 1px solid blue;
    float: left;
  }
  .RightChildren {
    width: calc(100% - 50px);
    border: 1px solid green;
    float: right;
```

```
  }
}
</style>
```

B, ----flex方式

```
   <div>
      <div>我是左侧</div>
      <div>我是右侧</div>
    </div>
<style  rel="stylesheet/scss">
  .Father {
    border: 1px solid red;
    height: 100px;
    display: flex;
  }
  .LeftChildren {
    width: 50px;
    border: 1px solid blue;
  }
  .RightChildren {
    border: 1px solid green;
    flex: 1;
  }
</style>
```

11，请写出你使用过的浏览器？并写出浏览器的内核?

答案:

> IE(内核：trident); internet explorer
> 火狐浏览器mozilla firefox (内核：gecko)
> 谷歌浏览器chrome(内核：webkit)
> opera浏览器（内核：presto）

12，em,rem,px的区别，以及实现原理?

答案:

- px像素（Pixel）。相对长度单位。像素px是相对于显示器屏幕分辨率而言的。
- em是相对长度单位。相对于当前对象内文本的字体尺寸

> 举个例子:

> 比如说当前容器`font-size:16px;`则`1em`就等于16px;

- rem 是CSS3新增的一个相对单位（相对于根元素的）,使用方法就是

> 浏览器的默认字体是16px，那么`1rem=16px`以此类推计算12px=0.75rem，10px=0.625rem，2rem=32px;
>
> 这样使用很复杂。

现在有一个根据浏览器尺寸计算大小的rem的计算方法，使用移动端，详情请查看前端开发公共css，js笔记

13，css3清除浮动的方式?

答案:

```
<div class="father" style="overflow: hidden;">
       <div class="type_lei_left">备注</div>
       <div class="type_lei_left_1" style="word-break: break-all;word-wrap: break-word;"></div>
        <div style="clear: both;display: none;"></div>
</div>
```

父级div加overflow: hidden;，在父级div的最后清除所有。

关键点二：HTML5

一，描述一下HTML5的新特性以及新增了哪些语义化标签

答案:
1：新的文档类型（<!DOCTYPE html>,即使浏览器不懂这句话，也会去渲染。
2：figure元素，语义化的标签，
3: 重新定义的<small>用来定义小的排版
4：去掉link和script标签里边的type属性
5：让div成为可编辑属性只需要加一个
contenteditable
属性
6：html5的新属性中加入<input type=email>浏览器会自动验证这个email属性
7：placeholder属性，加上不需要在进行验证了。（input里边的自带文字，点进去之后会自动消失）；
8：local stroge 可以在客户端存储大的数据，除非主动删除

-

新增的语义化标签

< title></title>：简短、描述性、唯一（提升搜索引擎排名）。

<header></header>：页眉通常包括网站标志、主导航、全站链接以及搜索框

<nav></nav>：标记导航，仅对文档中重要的链接群使用。

<main></main>：页面主要内容，一个页面只能使用一次。如果是web应用，则包围其主要功能。

<article></article>：包含像报纸一样的内容= =||是这么理解的，表示文档、页面、应用或一个独立的容器。

<section></section>：具有相似主题的一组内容，比如网站的主页可以分成介绍、新闻条目、联系信息等条

<aside></aside>：指定附注栏，包括引述、侧栏、指向文章的一组链接、广告、友情链接、相关产品列表

<footer></footer>：页脚，只有当父级是body时，才是整个页面的页脚。

<small></small>：指定细则，输入免责声明、注解、署名、版权。

：表示内容重要性

<mark></mark>：突出显示文本（yellow），提醒读者

<cite></cite>：指明引用或者参考，如图书的标题，歌曲、电影、等的名称，演唱会

<video>定义视频。(内嵌元素)

示例：

<video width="320" height="240" >

<source src="movie.ogg" type="video/ogg" />

Your browser does not support the video tag.

</video>

<audio>定义声音内容。(内嵌元素)

<mark> 标签定义带有记号的文本

二，HTML5本地存储有哪些方式？描述一下他们的异同点？

答案：

- 相同点：

都是保存在浏览器端的且同源的。

- 不同点：

（1）cookie数据始终在同源的http请求头中携带，cookie在浏览器和服务器之间来回传递，cookie可以限制保存在某个路径下，sessionstorage和localstorage不会自动把数据发送给服务器，仅在本地保存。cookie是浏览器和服务器之间传递数据的媒介。

（2）存储大小不同 cookie数据存储为4k，sessionstorage和localstorage可以达到5m甚至更大。

（3）数据有效期不同，sessionstorage仅在当前浏览器关闭前有效，localstorage始终有效，cookie仅在设置的过期时间前有效。

（4）作用域不同：sessionstorage不在不同的浏览器窗口中共享，即使是同一个界面，localstorage和cookie在同源窗口中都是共享的

****

*COOKIE*的增删该查

```
设置cookie
  function setCookie(name,value,days){     //封装一个设置cookie的函数
        var oDate=new Date();
        oDate.setDate(oDate.getDate()+days);    //days为保存时间长度
        document.cookie=name+'='+value+';expires='+oDate;
    }
    setCookie('password','12345','14');      //保存密码,14天后过期
获取cookie
 function getCookie(name){
        var arr=document.cookie.split(';');
        for(var i=0;i<arr.length;i++){
            var arr2=arr[i].split('=');
            if(arr2[0]==name){
                return arr2[1];  //找到所需要的信息返回出来
            }
        }
        return '';         //找不到就返回空字符串
    }
getCookie(password);
删除COOKIE
    function removeCookie(name){
        setCookie(name,1,-1);     //-1表示昨天过期,系统自动删除
        //将日期设置为过去
    }
修改cookie
document.cookie="password=88";
```

****

*sessionStorage*的增删该查

```
设置sessionStorage或者localStoragewindow.sessionStorage.setItem("strogeIndexcodes", "1234567");
```

```
设置sessionStorage或者localStorage window.sessionStorage.setItem("strogeIndexcodes",  123456);
获取sessionStoragelet contrastLists = window.sessionStorage.getItem("strogeIndexcodes");
删除sessionStoragelet contrastLists = window.sessionStorage.removeItem("strogeIndexcodes");
localStorage.clear();//清空localStorage中所有信息
```

三，对HTNL5语义化的理解?

- 什么是HTML语义化?

根据内容的结构化（内容语义化），选择合适的标签（代码语义化）便于开发者阅读和写出更优雅的代码的同时让浏览器的爬虫和机器很好地解析。

- 为什么要语义化?

答案:

（1）html语义化让页面内容结构更加清晰，便于浏览器搜索引擎解析，更容易阅读;
（2）利于搜索引擎的爬虫，利于SEO;
（3）使阅读源码的人对网站更容易将网站分块，便于阅读维护理解;
（4）便于团队开发和维护，语义化更具可读性，是下一步把网页的重要动向，遵循W3C标准的团队都遵循这个标准，可以减少差异化。

四，什么叫优雅降级和渐进增强?

答案:

（1）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。

（2）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。

五，CSS3文本效果

答案:

```
1)、text-shadow  文本阴影
text-shadow: 5px 5px 5px #FF0000;
2)、box-shadow  盒子阴影
box-shadow: 10px 10px 5px #888888;
3)、text-overflow  文本溢出
white-space: nowrap;
width: 200px;
overflow: hidden;
text-overflow: ellipsis;
4)word-wrap  换行
word-wrap:break-word;
5)word-break  单词拆分换行
  word-break: break-all;
```

关键点三：javaScript和Jquery

一，javascript的数据类型

```
基础的数据类型：
    1、字符串（string）
    2、 数字（number）
    3、布尔  boolean
    4、数组(array)
    5、null
    6、undefined）
复合的数据类型：
    1、object对象
    2、function  构造函数
    3、array数组
```

二，什么是window对象？什么是document对象?

答案:

- window对象是指浏览器打开的窗口。
- document对象是Document对象（HTML 文档对象）的一个只读引用，window对象的一个属性。

三，请简述call和apply的区别

答案

- call(), 调用一个对象的一个方法以另一个对象替换当前对象。
- apply(),应用某一个对象的一个方法，用另一个对象替换当前对象（继承性）

四，document.load和document.ready的区别。

答案：

- document.load在所有的资源全部加载完成后，执行下一个函数（会有卡顿）。
- document.ready只要加载完dom树就执行函数（加载比较快）。

五，JS如何阻止事件冒泡和阻止默认事件。

答案：

- event.stoppropagation()阻止事件冒泡；

- event.preventdefault()阻止默认事件。

> 在这插播一个事件捕获OR事件冒泡

- 事件捕获指的是从document到触发事件的那个节点，即自上而下的去触发事件。
- 相反的，事件冒泡是自下而上的去触发事件
- 绑定事件方法的第三个参数，就是控制事件触发顺序是否为事件捕获。

  true,事件捕获；

  false,事件冒泡。

  默认false,即事件冒泡。

看个例子：

```
document.getElementById("parent").addEventListener("click",function(e){
        alert("parent事件被触发，"+e.target.id);
 },true)
document.getElementById("child").addEventListener("click",function(e){
        alert("child事件被触发，"+e.target.id)
 },true)
```

这样的话就是自上而下的触发事件。

六，JS 对象和字符串互转

答案:

```
JSON.stringify()用于从一个对象解析出字符串
例如：var a = {a:1,b:2},JSON.stringify(a),输出 '{“a”：1，“b”：2}'
JSON.parse()从一个json字符串解析出json对象.
var str = '{"name":"huangxiaojian","age":"23"}'
json.parse(str)输出
      str = {
            name:"huangxiaojian",
            age:"23"
      }
```

七，JS splice()方法 或者删除数组中某个值的方法

答案:

```
数组中的splice()方法，删除数组中的某一个或几个值，
```

splice(index,length,item),index为数组开始的下标，length替换或删除的长度，item替换的值。
item若为0则直接删除。

八，eval()函数是做什么的？或者说如何将字符串解析成js对象并运行？？

答案:

- 它的功能是把对应的字符串解析成JS对象并运行；
  ！！应该避免使用eval，不安全，非常耗性能（2次，一次解析成js语句，一次执行）。
  由JSON字符串转换为JSON对象的时候可以用eval，var obj =eval('('+ str +')');

> 对数组做操作的一写方法

九，请用简单的语言描述数组方法pop(),push(),unshift(),shift()

答案:

- pop(),删除数组的最后一个元素，把数组长度减1，并返回它删除的元素的值，如果数组为空，则POP不改变数组，并返回undefined值
- push(),向数组的末尾添加一个或多个元素，并返回新的长度
- unshift(),向数组的开头添加一个或多个元素，并返回新的长度
- shift()方法，将数组的第一个元素从其中删除，并返回新的长度

十，数组的截取和合并

答案:

- array.slice(start, [end]);-->截取从start之后不包括end的元素；
- array.concat([item1[, item2[, . . . [,itemN]]]]); -->合并数组；

  两者都是返回数组的拷贝数组，注意是一个新的数组。

substring(start,stop)方法：
用于提取字符串中介于两个制定下标之间的字符. 包括start出的字符，但是不包括stop处的字符。
substr()
用于返回一个从指定位置开始的指定长度的子字符串。substr(start,length).

十一，数组元素的排序

答案：
array.reverse(); //方法用于颠倒数组中元素的顺序。

- array.sort(); //方法用于对数组的元素进行排序。请注意，数组在原数组上进行排序，不生成副本。

  默认为升序。如果调用该方法时没有使用参数，将按字母顺序对数组中的元素进行排序

  以下例子为对数值的大小进行排序。

```
function sortNumber(a,b){
     return a - b
}
let NumberLists = [10,5,30,25,1000,1,6];
如果直接使用sort进行排序则输出的值为“：[1, 10, 1000, 25, 30, 5, 6]
document.write(NumberLists .sort(sortNumber))  //1,5,6,10,25,30,1000
```

十二，数组元素的字符串化

答案：

arrayObj.join(separator); //返回字符串，这个字符串将数组的每一个元素值连接在一起，中间用 separator 隔开。

```
let NumberLists = [10,5,30,25,1000,1,6];
NumberLists.join(",");   //输出结果    "10,5,30,25,1000,1,6"
```

十三，将字符串分割成数组

答案：

split()方法，用于把一个字符串分割成数组。

```
split("s","d"),
s代表字符串或正则表达式，从该参数指定的地方分割。
d代表返回的数组的最大长度，返回的长度不会多余这个值得长度
看例子：
var str="How are you doing today?"str.split(" ")
["How", "are", "you", "doing", "today?"]
```

十四，编写一个方法，去掉一个数组的重复元素(有多种后期不断更新)

第一种方法：/使用indexof（）如果检索到值则返回1，如果检测不到则返回-1，的方法，和push（）向数组的末尾添加一个或多个元素的方法。

```
var arr = [1,1,1,2,2,3,45,45,65,76]
     function quchong(){
          //new一个新的数组来存放去重后的数组
          var newarr = [];
        //循环遍历需要去重的数组，如果数组中中已有值，就继续循环，如果没有值就添加到新的数组中
       for(var i = 0;i<arr.length;i++){
```

```
        for(var i = 0;i<arr.length;i++){
                if(newarr.indexOf(arr[i])==-1){
                    newarr.push(arr[i]);
                }
        }
        document.write(newarr)
    }
    quchong(arr);
```

第二种方法：双重循环，每次比较与他相邻的数，如果有相等的就删除相等的那个，如果不相等就继续循环。不做操作

```
var arr = [1,23,43,54,1,32,43,54,54,32]
       //循环遍历这个数组中每一个数然后与他相邻的数做比较，
      for (var i = 0; i < arr.length; i++) {
        for (var a = i + 1; a < arr.length; a++) {
          if (arr[i] === arr[a]) {
            arr.splice(i, 1);
            i--;
          }
        }
      }
```

十五，JS创建函数的方法

答案：

（1）声明函数：function name(){}

（2）匿名函数：function(){} 首先创建一个变量，变量是一个没有名字的函数

（3） 具名函数：首先创建一个变量，变量时一个有名的函数，var a = function bb(){}

（4）构造函数 new function

（5）立即函数（自动执行函数） 有四种写法：

```
1、 (function(){})()
2、 (function(){} ())
3、   !function(){}()
4、 void function()
```

十六，http请求状态码？也就是常见的HTTP协议状态？

答案：

- 200 请求成功

- 4开头的是前端（客户端）的错误
  400 错误的请求（bad request）
  404 not found 未找到 页面未响应，
  403服务器了解客户的请求，但是拒绝处理

- 5开头的一般是服务器的事
  500服务器内部错误 不能发完成客户的请求
  502 服务器作为网关或者代理收到了无效的响应
  503 service unavailable 服务不可用 服务器由于维护或 者负载过重未能应答。

十七，编写一个方法 ，写出系统时间，年/月/日/时/分/秒/

答案

```
var time = new Date();
var xitongshijian = time.tolocalestring();  // 得到本地的系统时间
```

十八，谈谈你对This对象的理解?

1、this总是指向函数的直接调用者（而非间接调用者）；
2、如果有new关键字，this指向new出来的那个对象；
3、在事件中，this指向触发这个事件的对象，特殊的是，IE中的attachEvent中的this总是指向全局对象Window；

看看下边几个例子，或许可以更好的理解this对象

this的指向

- this表示当前对象，this的指向是根据调用的上下文来决定的，默认指向window对象

全局环境

> 全局环境就是在<script></script>里面，这里的this始终指向的是window对象，
>
> <script>console.log(this)</script> 指向window对象

局部环境

> 在全局作用域下直接调用函数，this指向window
>
> <script>
>
> function func(){
>
> console.log(this) ;//this指向的还是window对象
>
> }
>
> func();

</script>

对象函数调用，哪个对象调用就指向哪个对象

```
<input type="button"id="btnOK" value="OK">
<script>
varbtnOK=document.getElementById("btnOK");
btnOK.onclick=function(){
console.log(this);//this指向的是btnOK对象
}
</script>
```

使用 new 实例化对象，在构造函数中的this指向实例化对象。

```
<script>
var Show=function(){
    this.myName="Mr.Cao";   //这里的this指向的是obj对象
}
var obj=new Show();
</script>
```

使用call或apply改变this的指向

```
<script>
var Go=function(){
     this.address="深圳";
}
var Show=function(){
     console.log(this.address);//输出 深圳
}
var go=new Go();
Show.call(go);//改变Show方法的this指向go对象
</script>
```

十九，请用js写一个判断某个字符在该字符串中出现了几次?

答案:

```
//经测试好使。
<script>
var a = "jhdoiwejesdds";
var n = (a.split('j')).length-1;
console.log(n);
</script>
```

about at Ajax , HTTP协议

一，.Ajax 是什么？Ajax的工作原理是什么？Ajax的优缺点是什么？

答案：

- Ajax其核心有 JavaScript、XMLHTTPRequest、DOM对象组成，通过XmlHttpRequest对象来向服务器发异步请求，从服务器获得数据。是一种创建交互式网页应用的网页开发技术。

（1）使用XMLHttpRequest对象与Web服务器进行异步数据通信。
（2）使用Javascript操作Document Object Model进行动态显示及交互。
（3）使用JavaScript绑定和处理所有数据。

- Ajax的工作原理相当于在用户和服务器之间加了一个中间层(AJAX引擎),使用户操作与服务器响应异步化。并不是所有的用户请求都提交给服务器,像一些数据验证和数据处理等都交给 Ajax引擎自己来做, 只有确定需要从服务器读取新数据时再由Ajax引擎代为向服务器提交请求。

> 优点
>
> （1）可以实现异步通信效果，实现页面局部刷新，带来更好的用户体验；按需获取数据，节约带宽资源。
>
> （2）前端和后端负载平衡
>
> （3）界面与应用分离。

> 缺点
>
> <1>.AJAX干掉了Back和History功能，即对浏览器机制的破坏。
>
> <2>.AJAX的安全问题。
>
> <3>.对搜索引擎支持较弱。
>
> <4>.破坏程序的异常处理机制。
>
> <5>.违背URL和资源定位的初衷。
>
> <6>.AJAX不能很好支持移动设备。
>
> <7>.客户端过肥，太多客户端代码造成开发上的成本。

二.简述ajax 的过程。

答案：

（1）创建XMLHttpRequest对象,也就是创建一个异步调用对象
（2）创建一个新的HTTP请求,并指定该HTTP请求的方法、URL及验证信息

（3）设置响应HTTP请求状态变化的函数
（4）发送HTTP请求
（5）获取异步调用返回的数据
（6）使用JavaScript和DOM实现局部刷新

三，请简述ajax中get()和post()的区别

（1）get请求，参数会显示在url中，将参数添加在请求的url路径后面，进行传参，（受url的限制）发送的数据量小，会被客户端的浏览器缓存起来，会带来严重的安全问题，一般在获取数据时使用get。
（2）post请求，需要添加HTTP请头，来发送到服务器，传送得数据量大，一般在上传数据时使用post。

区别一：get重点在从服务器上获取资源,post重点在想服务器发送数据;

区别二：get传输数据是通过URL请求,以filed(字段)=value的形式,置于URL后,并用"?"连接,

多个请求数据之间用
"&"连接,如http://127.0.0.1/Test/login.action?name=admin&password=admin，这个过程用户是可见的

区别三：get传输量小,因为受URL长度限制,但效率较低
post可以传输大量数据,所以上传文件时只能用post方式

区别四：get是不安全的,因为URL是可见的,可能会泄露私密信息,如密码等
post较get安全

四，什么是HTTP协议?

答案：

超文本传输协议(HTTP)是一种通信协议，它允许将超文本标记语言(HTML)文档从Web服务器传送到客户端的浏览器。

五，请说一下从输入 URL到加载出页面过程中都发生了什么？（问这个问题首先会考察你对HTTP协议的理解，而来知道你知识的深度以及扩展面）

答案：

- step1,输入URL，
- step2，浏览器查找域名的IP地址
- step3，查找完成之后浏览器向web服务器发送一个HTTP请求，

- step4，服务器的永久重定向相应
- step5，浏览器跟踪重定向地址
- step6，服务器处理请求
- step7，服务器返回一个http响应。
- step8，浏览器显示HTML
- step9,浏览器发送异步请求

六，浏览器是如何渲染页面的?

答案：

- step1,解析HTML文件，创建DOM树（自上而下的创建），
- step2，解析CSS，解析的优先级（浏览器默认--<link style>--内联样式<style></style>--行内标签）
- step3，将CSS与DOM合并，构建渲染树，（Render Tree）
- step4,布局与绘制，重绘(repaint)与重排（reflow）

> 接下来聊一聊常见的跨域问题。

一，说一下什么是跨域?为什么会出现跨域问题?

答案：

- 跨域，指的是浏览器不能执行其他网站的脚本.
- 出现跨域问题是因为受浏览器的同源策略（下边会有介绍）的影响，是浏览器对[JavaScript])施加的安全限制。出现跨域主要是为了用户的上网安全。

二，什么是同源策略?

答案：
浏览器只允许访问同一协议，域名，端口相同的网页。

三，解决跨域的几种办法?

答案：

- CORE跨域资源共享cross-origin-resource-sharing

服务器端对于 CORS的支持，主要就是通过设置Access-Control-Allow-Origin来进行的。如果浏览器检测到相应的设置，就可以允许Ajax进行跨域的访问--也就是常说的服务器加请求头。

不过后台用了spring boot 加一个注解就好了。框架里边自带的@CrossOrigin

- JSONP解决跨域

原理：借助

不受同源策略的影响进行数据请求。动态插入script标签，通过script标签引入一个js文件，这个js文件载入成功后会执行我们在url参数中指定的函数，并且会把我们需要的json数据作为参数传入。注意，只支持GET请求不过兼容性比较好，无法判断请求是否失败，安全性不高。

```
//js原生写法的JSONP
<script>
    function createJs(sUrl){
        var oScript = document.createElement('script');
        oScript.type = 'text/javascript';
        oScript.src = sUrl;
document.getElementsByTagName('head')[0].appendChild(oScript);
    }
    createJs('jsonp.js');
</script>

//Jquery版本的jsonp
 $.ajax({
   type:"get",
   url:'http://api.map.baidu.com/telematics/v3/weather?
               output=json&ak=0A5bc3c4fb543c8f9bc54b77bc155724',
   data:{
        location:$("#city").val()||"上海"
   },
   dataType:"jsonp",
   success: function (data) {
        //渲染模版
         var html = template('template',{list:data.results[0].weather_data})
         $('tbody').html(html);
   }
 });
```

现在衍生出来的vue-cli脚手架项目里的Npm模块http-proxy-middleware也可以解决跨域，详细的教程请看Vue面试题详解

> 再者谈谈经常遇到的
>
> 问题，多数情况下只是理解他的解释，但并未实际理解，也就是做题做不出来，自己写也写不好的那种，还是需要好好理解理解。

一，什么是闭包？谈谈你对闭包的了解？

答案：

- 闭包是指有权访问另一个函数作用域中变量的函数，创建闭包的最常见的方式就是在一个函数内创建另一个函数，通过另一个函数访问这个函数的局部变量,利用闭包可以突破作用链域，将函数内部的变量和方法传递到外部。

- 函数内部可以直接读取全局变量，但在函数外部无法读取到内部的变量，（定义在一个函数内部的函数），闭包是反垃圾回收机制的，这就是所谓的作用链域，闭包的特性是私有性，这种特性是闭包得变量存在内存中，内存消耗很大，造成网页性能问题，（在ie中会造成内存泄漏问题）在退出函数之前将全部变量清除。私有化变量

回答重点的四句话：

1，作用链域（子对象一级一级向上寻找父对象的变量，父对象的所有变量对子对象是可见的，反之不成立）

2，垃圾回收机制（违背）

3，私有化变量

4， 闭包会在父函数外部改变父函数内部变量的值

二，什么是垃圾回收机制？

答案：

- javascript具有自动垃圾回收机制，执行环境会负责管理代码执行过程中使用的内存。原理就是找出那些不再继续使用的变量，然后释放其占有内存。

找出程序中不在使用的变量，然后释放掉其占用的内存，垃圾回收机制会按照固定的时间周期来执行。不在使用的变量是局部变量，（生命周期结束的变量），全局变量的生命周期只有在浏览器页面关闭后才会结束。

- 垃圾回收机制的好处和坏处

- 好处：大幅简化程序的内存管理代码，减轻程序猿负担，并且减少因为长时间运转而带来的内存泄露问题。

- 坏处：自动回收意味着程序猿无法掌控内存。ECMAScript中没有暴露垃圾回收的借口，我们无法强迫其进行垃圾回收，更加无法干预内存管理。

三，Javascript作用链域?

答案:

当需要从局部函数查找某一属性或方法时，如果当前作用域没有找到，就会上溯到上层作用域查找，直至全局函数，这种组织形式就是作用域链。
全局函数无法查看局部函数的内部细节，但局部函数可以查看其上层的函数细节，直至全局细节。

> JS面向对象，创建对象，事件委托，原型链,前端性能优化等的有关的问题

一，Javascript面向对象?

答案:

一切事物皆对象
对象具有封装和继承特性
JavaScript 语言是通过一种叫做 原型（prototype）的方式来实现面向对象编程的
基于原型的面向对象方式中，对象（object）则是依靠 构造器（constructor）利用 原型（prototype）构造出来的。然而原型方式中的构造器 (constructor) 和原型 (prototype) 本身也是其他对象通过原型方式构造出来的对象。

二，Javascript原型链?

答案：回答重点
在调用某个对象的属性或方法时，js引擎会优先在该对象自身上查找该属性或方法，如果没有便去该对象的构造函数中去查找，如此逐级递归的查找，直到原型链的顶端，（**向上一级一级查找**）

三，Javascript创建对象的几种方式?

答案:

```
1、 对象字面量的方式
person = {
        firstname: "Mark",
        lastname: "Yun",
        age: 25,
        eyecolor: "black"
};
```

```
,,

2、 用function来模拟无参的构造函数
function Person() {}
var person = new Person(); //定义一个function，如果使用new"实例化",该function可以看作是一个Class
person.name = "Mark";
person.age = "25";
person.work = function() {
        alert(person.name + " hello...");
}
person.work();

3、 用function来模拟有参构造函数来实现（ 用this关键字定义构造的上下文属性）
function Pet(name, age, hobby) {
        this.name = name; //this作用域：当前对象
        this.age = age;
        this.hobby = hobby;
        this.eat = function() {
                alert("我叫" + this.name + ",我喜欢" + this.hobby + ",是个程序员");
        }
}
var maidou = new Pet("麦兜", 25, "coding"); //实例化、创建对象
maidou.eat(); //调用eat方法

4、 用工厂方式来创建（ 内置对象）
var wcDog = new Object();
wcDog.name = "旺财";
wcDog.age = 3;
wcDog.work = function() {
        alert("我是" + wcDog.name + ",汪汪汪.....");
}
wcDog.work();

5、 用原型方式来创建
function Dog() {}
Dog.prototype.name = "旺财";
Dog.prototype.eat = function() {
        alert(this.name + "是个吃货");
}
var wangcai = new Dog();
wangcai.eat();

6、 用混合方式来创建
function Car(name, price) {
        this.name = name;
        this.price = price;
}
Car.prototype.sell = function() {
        alert("我是" + this.name + ", 我现在卖" + this.price + "万元");
}
var camry = new Car("凯美瑞", 27);
camry.sell();
```

四，浅谈前端性能优化的问题?

答案:

- step1，使用CDN加速，
- step2，静态资源的压缩合并，
- step3，减少HTTP请求，
- step4，尽可能的减少DOM操作，
- step5，由于浏览器渲染机制的影响，JS,CSS文件放在页面最底部
- step6，使用懒加载，
- step7，使用浏览器缓存,本地存储，减少请求数量
- step8，使用SSR(server side rendering)后端渲染，数据直接输出到HTML
- step9，项目中采用按需加载

渲染类的一些方法:

- step1，压缩图片，
- step2，尽量使用CSS动画，开启GPU加速
- step3，Iconfont代替图片
- step4, 精简代码，压缩代码

五，DOM操作——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()
    removeChild()
    replaceChild()
    insertBefore() //在已有的子节点前插入一个新的子节点
（3）查找
    getElementsByTagName()    //通过标签名称
    getElementsByName()    //通过元素的Name属性的值(IE容错能力较强，会得到一个数组，其中包括id等于name值的)
    getElementById()    //通过元素Id，唯一性
```

结束语

以上为总结的常见的面试问题。

原作者：愿醒静卧忘尘谷

阅读原文