需求与解决方法

一、居中（垂直水平居中）

1、标准流

1、盒子(块级元素)居中：

1、子盒子明确宽高

1、绝对定位（+ margin）的方式

.parent {

position: relative;

}

.child {

width: 200px;

height: 200px;

position: absolute;

top: 50%;

left: 50%;

margin-top: -100px;

margin-left: -100px;

}

2、不论子盒子是否明确宽高

1、绝对定位（+ margin）的方式

.parent {

position: relative;

}

.child\_div {

position: absolute;

top: 0;

left: 0;

right: 0;

bottom: 0;

margin: auto;

}

2、绝对定位（+ transform：translate()）的方式

.parent {

position: relative;

}

.child\_div {

position: absolute;

top: 50%;

left: 50%;

transform: translate(-50%, -50%);

}

3、flex布局

.parent {

display: flex;

justify-content: center;

align-items: center;

}

.child\_div { }

2、行内元素、盒子内的单行文本

1、父元素明确高度

1、text-align: center方式

.parent {

height: 200px;

text-align: center;

line-height: 200px;

}

.child\_span { }

2、不论父元素是否明确高度

1、flex布局

.parent {

display: flex;

align-items: center;

justify-content: center;

}

.child\_span { }

3、多行文本垂直居中（上下padding的方式）

.parent {

padding-top: 50px;

padding-bottom: 50px;

}

2、非标准文档流

1、浮动的元素

在浮动的元素外加一层div

定位

flex布局

二、为什么要用闭包?

缓存数据，延长作用域链

三、什么情况下需要跨域?

不同源地址之间的请求

解决方案： jsonp、ajax跨域(CORS)

jsonp原理：

动态插入script标签

在客户端借助script标签请求服务端的一个动态网页（如：php文件），服务端的这个动态网页返回一段带有函数调用的javascript全局函数调用的脚本，将原本需要返回给客户端的数据传递进去。

CORS

在被请求的服务端响应的时候添加一个Access-Control-Allow-Origin的响应头，表示这个资源是否允许指定域请求

1. ajax

作用：

1、不重新加载整个页面的情况下，可以与服务器交换数据并更新部分网页内容

2、可以实现网页异步更新

缺点：

不支持浏览器back按键

暴露了与服务器交互的细节

对搜素引擎的支持比较弱

get 和 post

get 一般用于信息获取，使用url传递参数

post 一般用于表单提交，修改服务器上的资源

异步 ajax 的优缺点都有什么？

优点：

• 相对于同步ajax, 不会造成 UI 卡死，用户体验好。

• 相对于刷新页面，省流量

缺点：

• 后退按钮无效；

• 多个请求同时触发时，由于回调时间不确定，会造成混乱，避免这种混乱 需要复杂的判断机制。

• 搜索引擎不友好

• 数据安全

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。

同源策略

协议、域名、端口相同

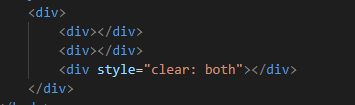
1. 清除浮动的方式

当父容器没有设置高度，里面的子盒子没有设置浮动的情况下会将父容器的高度撑开。一旦父容器中的子盒子设置浮动，脱离标准文档流，**父容器立马没有高度，父容器下面的兄弟盒子会跑到浮动的子盒子下面**。出现这种情况，我们需要清除浮动。

清除浮动不是不浮动，是清除浮动产生的不利影响

1、给浮动元素的父元素设置高度

2、在父元素的最后一个浮动子元素后边添加额外标签，使用clear:both; （不推荐使用，会产生冗余代码）

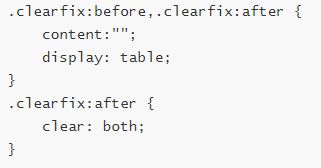


3、后伪元素清除浮动



4、双伪元素清除浮动

给浮动元素的父元素使用双伪元素清除浮动



5、给父元素设置overflow: hidden

注意：如果父元素中有定位的子元素，且该子元素超出了父元素，则不推荐使用这方法

6、给父元素的下一个兄弟元素中设置clear: both

注： 虽然父元素的兄弟元素不会跑到父元素中，但这种方法父元素还是没有高度

六、CSS的盒子模型

　　有两种，IE 盒子模型、标准 W3C 盒子模型；

IE的content部分包含了 border 和 pading;

七、CSS引入的方式有哪些 ？ link和@import的区别是?

内联 内嵌 外链 导入

区别 ：同时加载前者无兼容性问题，后者CSS2.1以下浏览器不支持  
link 支持使用javascript改变样式，后者不可

link 方式的样式的权重 高于@import 的权重.

link 属于 XHTML 标签，而@import 是 CSS 提供的;

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;

八、javascript的typeof返回哪些数据类型

Object number function boolean underfind

九、call和apply的作用和区别

作用：使用参数改变函数内部的this指向

区别：fun.call(box,2,5);

fun.apply(box,[2,6]); //以数组的方式传值

1. 阻止事件继续冒泡

1、window.event.cancelBubble = true

btn.onclick = function() {

console.log(“ko”);

window.event.cancelBubble = true； //IE特有

}

2、stopPropagation()

btn.onclick = function(e) {

console.log(“ko”);

e.stopPropagation()；

}

十一、iframe缺点  
1、iframe会阻塞主页面的Onload事件；

2、iframe和主页面共享连接池，而浏览器对相同域的连接有限制，所以会影响页面的并行加载。

如果需要使用iframe，最好是通过javascript动态给iframe添加src属性值，这样可以可以绕开以上两个缺点。

十二、split() 、 join() 的区别

前者是切割成数组的形式，后者是将数组转换成字符串

十三、.数组方法

pop() push() unshift() shift()

Push()尾部添加 pop()尾部删除

unshift()头部添加 shift()头部删除

十四、创建、添加 删除 替换 插入到某个接点的方法

document.createElement("span")

obj.appendChild()

obj.innersetBefore()

obj.replaceChild()

obj.removeChild()

十五、编写一个数组去重的方法

function oSort(arr) {

var newArr = [];

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

if(newArr.indexOf(arr[i]) < 0){

newArr.push(arr[i]);

}

}

return newArr;

}

18、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。

1. 数组快速排序

选择排序（从小到大）、冒泡排序（从小到大）

快速排序法

快速排序法号称是目前最优秀的算法之一，实现思路是，将一个数组的排序问题看成是两个小数组的排序问题，而每个小的数组又可以继续看成更小的两个数组，一直递归下去，直到数组长度大小最大为2。

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。

20. 什么是语义化的HTML?

直观的认识标签， 对于搜索引擎的抓取有好处

方便团队开发和维护，语义化更更具可读性

21、”==”和“===”的不同  
 前者会自动转换类型，后者不会

1. 数字取整

parseInt(5/2) //2 丢弃小数部分,保留整数部分

Math.floor(5/2) //2 丢弃小数部分,保留整数部分

Math.ceil(5/2) //3 有小数就进一

Math.round(5/2) //3 四舍五入

23、[删除数组中某一项或几项的几种方法](https://www.cnblogs.com/Joans/p/3981122.html)

1、splice方法

splice(index,len,[item])    //该方法会改变原始数组。

有3个参数，它也可以用来替换/删除/添加数组内某一个或者几个值

index: 数组开始下标

len: 替换/删除的长度

item: 替换的值，删除操作的话不需要第三个参数

如：arr = ['a','b','c','d']

删除

arr.splice(1,1)   //['a','c','d']

arr.splice(1,2)  //['a','d']

替换

arr.splice(1,1,'ttt')      //['a','ttt','c','d']

arr.splice(1,2,'ttt')      //['a','ttt','d']

添加

arr.splice(1,0,'ttt')      //['a','ttt','b','c','d']

2：delete

delete删除掉数组中的元素后，会把该下标出的值置为undefined,数组的长度不会变

如：delete arr[1]  //['a', ,'c','d']

中间出现两个逗号，数组长度不变，有一项为undefined

24. 前端性能优化

1、页面级优化

\* 减少HTTP请求

\* 合理设置HTTP缓存

\* css Sprites(css精灵)

\* 资源合并压缩

\* 懒加载（Lazy Load Javascript）

2、代码级优化

避免使用eval 和 Function

避免重定向

25、 从10到100中选10个随机数，并排好序，放入数组中

var arr = [];

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

arr.push(Math.round(Math.random() \* 100));

}

console.log(arr);

//冒泡排序

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

for(var z=0; z<arr.length-j-1; z++) {

if (arr[z] > arr[z+1]) {

var temp = arr[z+1];

arr[z+1] = arr[z];

arr[z] = temp;

}

}

}

console.log(arr);

26、 一个程序至少有一个进程，一个进程至少有一个线程

进程在执行过程中拥有独立的内存单元，多个线程共享内存，从而极大地提高了程序的运行效率

27、 实现浏览器内多个标签页之间的通讯

调用localStroge 、cookies等本地存储方式

28、 H5API

语义化标签（header 、nav 、footer）

音频、视频 （audio、video）

画布（canvas）

地理（Geolocation）

本地存储（localStorage、sessionStorage）

拖拽释放

29、初始化css样式

因为浏览器兼容的问题，不同浏览器对有些标签的默认值是不同的

30、css3新增

选择器

圆角

阴影

文字特效

旋转、缩放、位移、倾斜

MVVM

model 数据模型

Vue UI组件，将数据模型转化为UI展现出来

ViewModel 监听模型数据的改变和控制视图行为、处理用户交互

数据驱动视图，不操作DOM

响应式数据： 数据驱动视图

团队协作工具

teambiton

=============

响应式开发

\* 网页布局方式

1、固定宽度布局：为网页设置一个固定的宽度，通常以px做为长度单位，常见于pc端网页

2、流式布局： 为网页设置一个相对的宽度，通常以百分比做长度单位

3、栅格化布局： 将网页宽度人为的划分均等的长度，然后排版布局时则以这些均等的长度量为单位，通常利用百分比做为长度单位来划分均等的长度

4、响应式布局： 通过检测设备信息，决定网页布局方式，即用户如果采用不同的设备访问一个网页，有可能会看到不一样的内容，一般情况下是检测设备屏幕的宽度来实现

注：实际开发过程中，以上方式往往互相结合使用

\* 响应式开发的原理： 媒体查询

- css语法

@media mediatype and|not|only (media feature) {

CSS-code;

}

也可以针对不同的媒体使用不同stylesheets

<link rel=”stylesheet” media=”mediatype” and|not|only (media feature)”

href=”mystylesheet.css”>

- 媒体类型

all 用于所有设备

print 用于打印机和打印预览

screen 用于电脑屏幕、平板、智能手机

speech 用于屏幕阅读器等发声设备

- 媒体功能

device-height 设备的屏幕可见高度

device-width 设备的屏幕可见宽度

max-device-height 设备的屏幕可见的最大高度

max-device-width 设备的屏幕的最大可见宽度

min-device-height 设备的屏幕可见的最小高度

min-device-width 设备的屏幕的最大可见宽度

（在移动端能正常响应，和期望值一样

device：设备。他指当前设备的宽度，当拖动来改变浏览器大小的时候，当前终端设备的宽度并不会变化，意味着媒体查询不会响应

）

max-height 设备中的页面最大可见区域宽度

max-width 设备中的页面最大可见区域宽度

min-height 设备中的页面最小可见区域宽度

min-width 设备中的页面最小可见区域宽度

（在pc端和移动端都能正常响应，效果一致。

它指当前可视区域的宽度

）

例子：

//设置不同屏幕宽度区间下的body的背景色样式

\* body {

background-color: green;

}

@media screen and (max-width: 768px) {

body {

background-color: red;

}

}

@media screen and (min-width: 768px) and (max-width: 992px) {

body {

background-color: blue;

}

}

@media screen and (min-width: 992px) and (max-width: 1200px) {

body {

background-color: pink;

}

}

@media screen and (min-width: 1200px) {

body {

background-color: purple;

}

}

- 或者如下这样写（推荐，向上兼容，向下覆盖）

body {

background-color: green;

}

@media screen and (min-width: 768px) {

body {

background-color: red;

}

}

@media screen and (min-width: 992px) {

body {

background-color: blue;

}

}

@media screen min-width: 1200px) {

body {

background-color: pink;

}

}

- 再或者如下这样写

body {

background-color: green;

}

@media screen and (max-width: 1200px) {

body {

background-color: pink;

}

}

@media screen and (max-width: 992px) {

body {

background-color: blue;

}

}

@media screen and (max-width: 768px) {

body {

background-color: red;

}

}

\* <link rel=”stylesheet” href=”a.css”>

<link rel=”stylesheet” media=”screen and (min-width: 992) and (max-width: 1200)”

href=”b.css”>

<link rel=”stylesheet” media=”screen and (min-width: 1200)” href=”b.css”>

使用媒体查询设置页面布局

需求： w < 768 子元素宽度为父容器宽度的100%

w 768~992 每一行放放置两个子元素 50%

w 992~1200 每一行放置三个子元素 33.33%

w > 1200 每一行放置四个子元素 25%

.box > div {

width: 100%;

background-color: skyblue;

box-sizing: border-box;

border-right: 2px solid blue;

border-bottom: 2px solid blue;

}

@media screen and (min-width: 768px) {

.box > div {

width: 50%;

}

}

@media screen and (min-width: 992px) {

.box > div {

width: 33.33%;

}

}

@media screen and (min-width: 1200px) {

.box > div {

width: 25%;

}

}

css框架

1、Bootstrap

2、Amaze UI

3、Framework7

Bootstrap

栅格系统

是用来描述设置当前子元素在指定屏幕下，占据当前屏幕默认12等份中的n（1~12）等份

1、先添加容器container/container-fluid

2、在容器中添加row

3、在row中添加子元素，设置栅格样式

4、在子元素添加具体的内容

offset偏移：

col-xs-offset-2 （向后偏移）

偏移的距离是以栅格做为单位，但这种偏移会直接影响后面的元素，有可能造成后面的元素换行显示，应为它是通过margin-left属性来实现

push 往后(右)推 col-xs-push-2

pull 往前(左)推 col-xs-pull-2

通过定位来实现，不会影响其他元素，但可能可其他元素重叠

hiddle

hiddle-sm 针对不同屏幕尺寸隐藏或显示页面内容

<div class=”container”>

<div class=”row”>

<div class=”col-xs-2”>1</div>

<div class=”col-xs-10”>

//这里的外层没有再包含container，那么嵌套列的宽度就是参照当前所在的栅格宽度

如果在外层添加了container，那么参照就是核心样式文件所设置的容器宽度

<div class=”row”>

<div class=”col-xs-6”>3</div>

<div class=”col-xs-6”>3</div>

</div>

</div>

</div>

</div>

less

\* 变量 @变量名：值

@baseColor: #e92322;

a {

color: @baseColor;

}

\* 混入

.addRadius {

border-radius: 10px;

background-color: red;

}

div {

width: 200px;

height: 200px;

.addRadius;

}

\* 函数

// @r就是这个函数的参数，后面加100px表示参数默认值为100px

.box(@r: 100px) {

width: @r;

}

div {

height: 200px;

.box(200px);

}

\* 嵌套

div {

width: 500px;

ul {

list-style: none;

}

> .box {

color: red;

}

&::before {

content: “”;

}

}

sass

1、先安装Ruby 下载网址：https://rubyinstaller.org/downloads/

注意：安装Ruby时一定要勾选 Add Ruby executables to your PATH，用来将Ruby添加到系统变量，后续可以省却很多不必要的麻烦

2、更换gem源

- 删除原gem源

gem sources --remove https://rubygems.org/

- 添加国内源

gem sources -a http://gems.ruby-china.com/

- 打印是否替换成功

gem sources -l

更换成功后打印如下

http://gems.ruby-china.com/

3、安装Sass和Compass

gem install sass

gem install compass

检测是否安装成功

sass -v

compass -v

若要更新sass

gem update sass

===========

webpack

是前端的一个项目构建工具，是基于Node.js开发出来的一个前端工具

借助于webpack，可以完美实现资源的合并、打包、压缩、混淆等诸多功能

安装到项目依赖中（开发依赖中）

npm install webpack --save-dev