# js补充：

# arguments:

当一个函数不确定要传入多少参数，用arguments可以记录

arguments.length获取传入参数个数，arguments[0]来获取传参

# 二维数组

var ary = [[], [], [], []]

# 数组过滤方法filter()

arr.filter(function(item){return item >30}); alear(arr)

# onload(function(){…})页面加载后发生

不是必须的；script标签在元素的前面，需要加window.onload；如果script放在了元素后面，就不需要加 window.onload。

$(function(){…})

# DOM获取当前有效样式

getComputedStyle()可获取css中的样式

var a = document.getElementById()

alert(getComputedStyle(‘a’){“width”});

(getComputedStyle(‘a’).width;

# 自定义属性设置setAttribute() getAttribute()

[əˈtrɪbjuːt , ˈætrɪbjuːt]

var a = document.getElementById()

1)直接.设置和访问 alert(a.className = “xxx”)

2) a.setAttribute(“class”)获取class的名字

a.setAttribute(“class”, “xxx”)

删除属性：a.removeAttribute(“class”)

区别：访问用的class不是className 设置可以自定义设置a.setAttribute(“aaa”, “xxx”)

# dom节点类型：元素节点，文本节点，属性节点

这三个节点都有三个属性：nodeName,nodeType,nodeValue

可通过a.childNodes[1].nodeName 获取第二个子节点信息

 <div id="text"><em>aa</em><b>bb</b><span>cc</span></div>

a.attributes 返回属性节点集合(不重复且无序)

获取属性集合长度.length

访问其中一个属性节点a.attributes(‘id’)

# \*去除空白节点

remove.SpaceNode() 不怎么重要尽量不用TagName

上图若将em,b,span三者换行，则会从三个三个节点变成5个(两个换行导致的空白节点)

去除方法： 略

# 给一个块元素添加或替换类名

**el.setAttribute('class','class1') ie7及以下不兼容**

**el.setAttribute('className', 'class2')  和上面相反**

**el.className='class3' 都兼容**

# 代替jq的sibings()

**function** siblings(dom){

**var** result = [];

    // 获取dom的父节点

**var** parent = dom.parentNode || dom.parentElement;

    // 获取dom父节点下的所有子节点，包括dom本身

**var** siblings = parent.children || parent.childNodes;

    // 将dom从结果集中排除

**for**(**var** i = 0, len = siblings.length; i < len; i++){

**if**(siblings[i] != dom){

            result.push(siblings[i]);

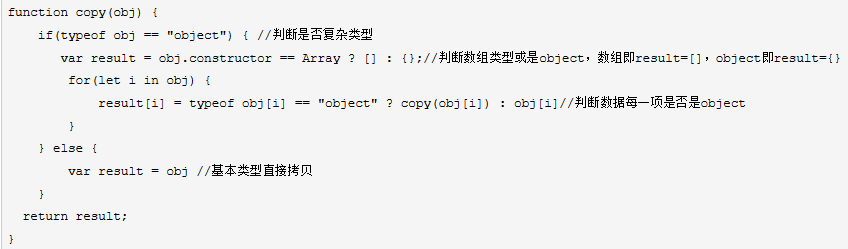
        }

    }

**return** result;

}

# 深拷贝和浅拷贝



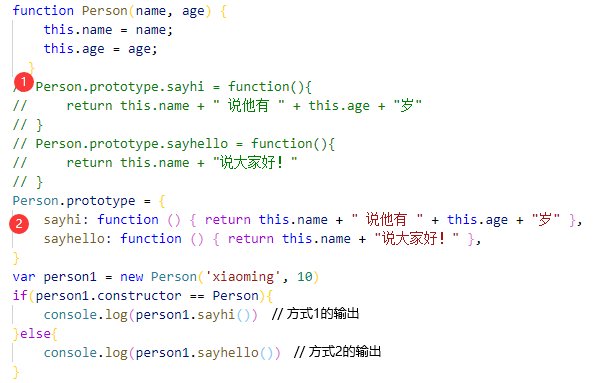
注1：深拷贝就是单独创建一个对象，然后将原对象的数据通过for循环赋值，变成一个全新对象。

注2：obj.constructor属性引用该对象的构造函数；对于obj对象，该指针指向原始的obj()函数。常用来判断未知对象的类型。

# constructor 和prototype

constructor是一个对象的**属性**，这个属性存在在此对象的prototype中, 指向此对象的构造函数

prototype是一个函数属性(只要是函数就有), 此属性同时也是一个对象, 保存着对象实例所**共有**的**属性和方法**



# 防抖：在一定的时间内事件只发生一次

例：

# var的作用域，一般还会比较let和const的区别 未完

使用var在函数或全局内任何地方声明变量相当于在其**内部最顶上声明**它，这种行为称为Hoisting（提升）。

注：只是在最顶部声明，**赋值还是在当前位置。**

# axios和ajax的区别

axios是通过promise实现对ajax技术的一种(部分)封装；就像jQ中对ajax的封装。

优点：从node.js中创建http请求；支持promiseAPI；客户端防止CSRF(跨域请求伪造)；提供了一些并发请求的接口(重要)。

使用：

1. main.js入口文件中修改原型链

import axios from ‘axios’;

Vue.prototype.$ajax = axios;

1. 使用：
2. **get：**

this.$ajax.get(‘/api/cityList’).then(function(res){ … }).catch((res) => { … })

this.$ajax.get(‘/api/cityList?status=0’).then(function(res){ … }).catch((res) => { … })

1. **post：**

this.$ajax.post(‘/text/demo1’).then(function(res){ … }).catch((res) => { … })

带参数的请求（发送带有json格式参数）需要使用qs模块;

导入方法和axios相同：

import qs from ‘qs’;

Vue.prototype.$qs= qs;

使用：

this.$ajax.post(‘/api/cityList’, **this.$qs.stringify({ nm: ‘北京’ })**)

.then(function(res){ … })

.catch((res) => { … })

注：this.$ajax.get()只是部分封装，用起来方便，此外还有delete、request等等；

1. 完整版post带参数请求（config）：

axios({

method: 'post',

url: '/user/12345',

data: {

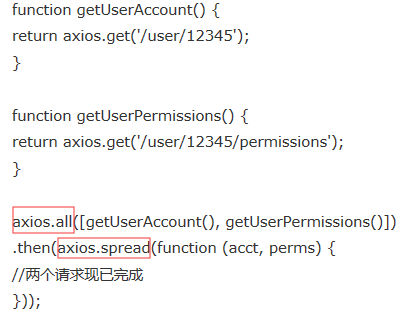
firstName: 'Fred',

lastName: 'Flintstone'

}

});

1. 并发请求：
2. axios.all（iterable） 2) axios.spread（callback）



# 构造函数

1. 构造函数的的方法名首字母要大写，普通函数不需要。

2. 构造函数的调用方法为： new  Show();普通函数：show（）。

3. 构造函数的类名和方法名一样；

4. 构造函数要用this构造属性和方法；

# CSS补充

## 响应式布局和自适应布局的区别

**响应式布局**就是实现不同屏幕分辨率的终端上浏览网页的不同展示方式。通过响应式设计能使网站在手机和平板电脑上有更好的浏览阅读体验。换句话说就是**一个网站能够兼容多个终端**，而不是为了每一个终端做一个特定的版本。

**自适应布局**就是指能忘了使**网页自适应**的显示在**不同大小终端设备**上的新网页设计方式及技术，它**需要开发多套界面来适应不同的终端**。

**区别**

1.自适应布局通过检测视口分辨率，来判断当前访问的设备是：pc端、平板、手机，从而请求服务层，返回不同的页面；响应式布局通过检测视口分辨率，针对不同客户端在客户端做代码处理，来展现不同的布局和内容。

2.自适应布局需要开发多套界面，而响应式布局只需要开发一套界面就可以了。

3.自适应对页面做的屏幕适配是在一定范围：比如pc端一般要大于1024像素，手机端要小于768像素。而响应式布局是一套页面全部适应。

4.自适应布局如果屏幕太小会发生内容过于拥挤。而响应式布局正是为了解决这个问题而衍生出的概念，它可以自动识别屏幕宽度并做出相应调整的网页设计。

总之，响应式布局还是要比自适应布局要好一点，但是**自适应布局更加贴切实际**，因为你只需要考虑几种状态就可以了而不是像响应式布局需要考虑非常多状态。所以的说无论哪种设计都有它们各自的特点，我们要根据项目的需求来选择适合的布局方式。

1. margin为负数时：
   1. margin-{left | top}为负时，元素向左（上）移动
   2. margin-{right |bottom}为负时，元素从右边开始裁剪
2. 实现宽度自适应且宽高比例为16:9

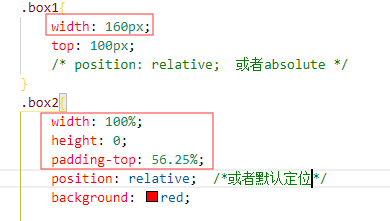
<div class="box1">

<div class="box2"></div>

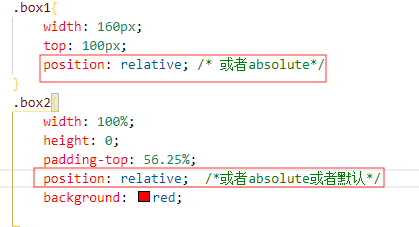
</div>

由box1定宽；box2宽度100%自适应，高度设为0；通过padding-top的百分比进行比例设置。100%时宽高相同

**情况1**：box1默认定位，则box2需要设置positon: relative或者设置默认定位；若设置绝对定位，则会相对浏览器屏幕宽度进行比例显示。



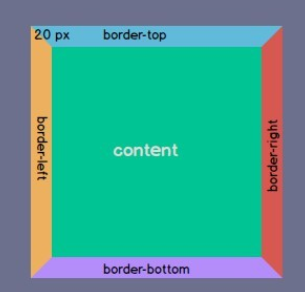
情况2：box1设置了相对定位或者绝对定位时；box2除了fixed之外都可设置。



情况3：若box设置了position: fixed; 则不管box1如何设置，宽高都会根据浏览器屏幕宽高决定。

1. css画出一个三角形和箭头

该方法是利用border四个边框界限划分所导致的。

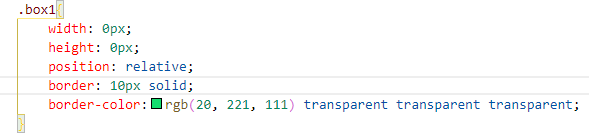


所以，当conten没有宽高时，border的四个

边框会将整个元素划分为四个等腰直角三角形。

然后选择一个边框设置颜色，其他设置成：  
transparent，变成全透明。类似rgba(0,0,0,0)

此时可以得到一个朝下的三角形。



更多位置的三角形通过选择border的边之外，还可以通过

transform: rotate(45deg) 转动角度；

箭头则是通过两个三角形错位覆盖获取。