CSS命名规范

命名的三种写法

小驼峰命名规则：第一个单词小写，其他单词首字母大写

写法如：myFirstName

大驼峰命名规则：第一个单词大写，其他单词首字母也大写

写法如：MyFirstName

下划线命名规则：每个单词之间使用 `_` 分割

写法如：my_first_name

命名规范

项目命名

全部采用小写方式， 以下划线分隔。 例： `my_project_name`

目录命名

参照项目命名规则；

有复数结构时，要采用复数命名法。

例： `scripts` , `styles` , `images` , `data_models`

vue的项目中，components下的组件目录名，使用大驼峰命令

例： `LeftBar`

JS文件命名

参照项目命名规则。

例： `account_model.js`

CSS, SCSS文件命名

例： `retina_sprites.css`

HTML文件命名

例： `error_log.html`

HTML规范

语法：

- 缩进使用tab（2个空格）；
- 嵌套的节点应该缩进；
- 在属性上，使用双引号，不要使用单引号；
- 属性名全小写，用中划线（-）做分隔符；
- 要在自动闭合标签结尾处使用斜线，为了适应 react；

```
<!DOCTYPE html>
<html>
    <head>
        <title>Page title</title>
    </head>
    <body>
        <img src="images/company_logo.png" alt="Company" />

        <!-- 属性名全小写，用中划线（-）做分隔符 -->
        <h1 class="hello-world">Hello, world!</h1>
    </body>
</html>
```

标准模式

```
<!DOCTYPE html>
<html>
    ...
</html>
```

规定字符编码

通过声明一个明确的字符编码，让浏览器轻松、快速的确定适合网页内容的渲染方式，通常指定为'UTF-8'。

IE兼容模式

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="X-UA-Compatible" content="IE=Edge">
    </head>
    ...
</html>
```

减少标签数量

在编写HTML代码时，需要尽量避免多余的父节点；

```
<!-- bad -->
<span class="avatar">
    <img src="...">
</span>

<!-- good -->
<img class="avatar" src="...">
```

语义化标签

关于所有命名相关的定义，要求 `语义化` 明确，达到 `见名知意` ，例如：菜单：menu

不要使用拼音，不会的单词可以自行搜索

```
<!-- bad -->
<div>
    <p></p>
</div>

<!-- good -->
<header></header>
<footer></footer>
```

CSS 规范

缩进

使用tab缩进（2个空格）

```
.element {
    border-radius: 10px;
    width: 50px;
    height: 50px;
}
```

分号

每个声明结束都要加分号

```
.element {
    border-radius: 10px;
    width: 50px;
    height: 50px;
}
```

注释

注释统一使用 /**/

```
.element {
    /* border-radius: 10px; */
    width: 50px;
    height: 50px;
}
```

引号

- url的内容要用引号
- 属性选择器中的属性值需要引号

```
.element:after {
    content: "";
    background-image: url("logo.png");
}

li[data-type="single"] {
    ...
}
```

命名

- 类名使用小写字母，以中划线分隔
- id采用驼峰式命名
- scss中的变量、函数、混合、placeholder采用驼峰式命名

```
/* class */
.element-content {
    ...
}

/* id */
#myDialog {
    ...
}

/* 变量 */
$colorBlack: #000;
```

```
/* 混合 */
@mixin centerBlock {
    ...
}
```

JavaScript 规范

缩进

使用tab缩进（2个空格）

```
if (x < y) {
    x += 10;
} else {
    x += 1;
}
```

变量命名

- 标准变量采用驼峰式命名
- 'ID'在变量名中全大写
- 'URL'在变量名中全大写
- 'Android'在变量名中大写第一个字母
- 'iOS'在变量名中小写第一个，大写后两个字母
- 常量全大写，用下划线连接
- 构造函数，大写第一个字母
- jquery对象必须以'$'开头命名

```
var thisIsMyName;

var goodID;

var reportURL;

var AndroidVersion;

var iOSVersion;

var MAX_COUNT = 10;

function Person(name) {
    this.name = name;
}

// not good
var body = $('body');

// good
var $body = $('body');
```

变量声明

一个函数作用域中所有的变量声明尽量提到函数首部，用一个var声明，不允许出现两个连续的var声明。如果可以使用let和const的，要使用let和const。

```
function doSomethingWithItems(items) {
    // use one var
    let value = 10,
        result = value + 10,
        i,
        len;

    for (i = 0, len = items.length; i < len; i++) {
        result += 10;
    }
}
```

分号

统一要加分号。

空格

以下几种情况不用写空格：

- 对象的属性名后
- 函数调用括号前
- 无论是函数声明还是函数表达式，'('前不要空格
- 数组的'['后和']'前
- 对象的'{'后和'}'前
- 运算符'('后和')'前

以下几种情况一定要写空格：

- 三元运算符'?:'前后
- 代码块'{'前
- 下列关键字前：else, while, catch, finally
- 下列关键字后：if, else, for, while, do, switch, case, try,catch, finally, with, return, typeof
- 单行注释'//'后（若单行注释和代码同行，则'//'前也需要），多行注释'*'后
- (对象的属性值前
- for循环，分号后留有一个空格，前置条件如果有多个，逗号后留一个空格
- 无论是函数声明还是函数表达式，'{'前一定要有空格
- 函数的参数之间

```
// not good
var a = {
    b :1
};

// good
var a = {
```

```
    b: 1
};

// not good
++ x;
y ++;
z = x?1:2;

// good
++x;
y++;
z = x ? 1 : 2;

// not good
var a = [ 1, 2 ];

// good
var a = [1, 2];

// not good
var a = ( 1+2 )*3;

// good
var a = (1 + 2) * 3;

// good
var doSomething = function(a, b, c) {
    // do something
};

// good
doSomething(item);

// not good
for(i=0;i<6;i++){
    x++;
}

// good
for (i = 0; i < 6; i++) {
    x++;
}
```

空行

以下几种情况一定要有空行

- 变量声明后（当变量声明在代码块的最后一行时，则无需空行）
- 注释前（当注释在代码块的第一行时，则无需空行）
- 文件最后保留一个空行

换行

换行的地方，行末必须有','或者运算符；

以下几种情况不需要换行：

- 下列关键字后：else, catch, finally
- 代码块'{'前

以下几种情况需要换行：

- 代码块'{'后和'}'前
- 变量赋值后

```
// not good
var a = {
    b: 1
    , c: 2
};

x = y
    ? 1 : 2;

// good
var a = {
    b: 1,
    c: 2
};

x = y ? 1 : 2;

// good
if (condition) {
    ...
} else {
    ...
}

try {
    ...
} catch (e) {
    ...
} finally {
    ...
}

// not good
function test()
{
    ...
}

// good
function test() {
    ...
}

// not good
var a, foo = 7, b,
    c, bar = 8;

// good
```

```
var a,
    foo = 7,
    b, c, bar = 8;
```

注释

单行注释

- 注释单独一行的情况下，注释的//后面要跟一个空格
- 注释如果和代码同一行，代码分号结束后，要跟一个空格，注释的//后也要跟一个空格

```
// 调用函数
foo()

var maxCount= 10; // 这是一个变量
```

多行注释

多行注释使用下面这种形式：

```
/**
 * 代码注释1
 * 代码注释2
 */
```

建议在以下几种情况使用：

- 难于理解的代码段
- 可能存在错误的代码段
- 浏览器特殊的HACK代码
- 业务逻辑强相关的代码

函数注释

复杂的函数，所有类，都必须进行函数注释，函数注释使用业界统一的规范，方便后续使用jsdoc生成文档。

```
/**
 * 获取任务的名称
 * @param id {Number} 传入需要获取名称的人物id
 * @return {String} 返回的姓名
 * @author shi 2015/07/21 可以不写
 * @version 1.1.0 可以不写
 * @example 示例代码，可以不写
*/
function getTaskName(id) {
    let name = 'test';
    return name;
}
```

引号

最外层统一使用单引号

```
// not good
var x = "test";

// good
var y = 'foo',
    z = '<div id="test"></div>';
```

对象，数组

- 对象属性名不需要加引号；
- 对象以缩进的形式书写，不要写在一行；
- 数组、对象最后不要有逗号。

```
// not good
var a = {
    'b': 1
};

var a = {b: 1};

// good
var a = {
    b: 1,
    c: 2
};
```

括号

下列关键字后必须有大括号（即使代码块的内容只有一行）： `if` , `else` , `for` , `while` , `do` , `switch` , `try` , `catch` , `finally` , `with` 。

```
// not good
if (condition)
    doSomething();

// good
if (condition) {
    doSomething();
}
```

undefined

永远不要直接使用undefined进行变量判断；

使用typeof和字符串'undefined'对变量进行判断。

```
// not good
if (person === undefined) {
    ...
}

// good
if (typeof person === 'undefined') {
    ...
}
```

不允许存在多层嵌套的条件判断和循环（最多三层）

条件判断能使用三目运算符和逻辑运算符解决的，就不要使用条件判断，但是**谨记不要写太长的三目运算符。**

```
// bad
if(x === 10) {
  return 'valid';
} else {
  return 'invalid';
}

// good
return x === 10 ? 'valid' : 'invalid'

// bad
if(!x) {
  if(!y) {
    x = 1;
  } else {
    x = y;
  }
}

// good
x = x || y || 1
```

简单解释一下逻辑运算符，逻辑运算符主要有两种，一个是 || 逻辑或，一个是 && 逻辑与。

- 逻辑或 ||：当前一个为真时，返回前一个值，前一个为假时返回后一个值。

```
var x = 1;
console.log(x || 2); // 1

var y = 0;
console.log(y || 2); // 2
```

逻辑与 &&：当前一个为真时，返回后一个值，前一个为假时返回前一个值。

```
var x = 1;
console.log(x && 2); // 2

var y = 0;
console.log(y && 2); // 0
```

其他

- 对上下文this的引用只能使用'_this', 'that', 'self'其中一个来命名；
- 行尾不要有空白字符；
- 不允许有空的代码块。

```
// not good
function Person() {
    // not good
    var me = this;

    // good
    var _this = this;

    // good
    var that = this;

    // good
    var self = this;
}

// not good
if (condition) {

}
```

多人开发同一个页面

关于多人开发同一个页面，加入个人前缀，例如：菜单：w_menu

CSS样式命名方式

常见的命名方式

- 页头：header 如：#header{属性:属性值;} 或.header{属性:属性值;},也许你需要了解class与id 区别及用法
- 分区：section
- 登录条：loginBar
- 标志：logo
- 侧栏：sideBar
- 广告：banner
- 导航：nav
- 子导航：subNav
- 菜单：menu
- 子菜单：subMenu
- 搜索：search
- 滚动：scroll
- 页面主体：main
- 内容：content
- 标签页：tab
- 文章列列表：list
- 提示信息：msg
- 小技巧：tips
- 栏目标题：title
- 加入：joinus
- 指南：guild
- 服务：service
- 热点：hot
- 新闻：news
- 下载：download
- 注册：regsiter
- 状态：status
- 按钮：btn
- 投票：vote
- 合作伙伴：partner
- 友情链接：friendLink
- 页脚：footer
- 版权：copyRight
- 外　套：wrap
- 版 权：copyRight
- 商　标：label
- 标　题：title
- 顶导航：topnav
- 边导航：sidebar
- 左导航：leftsideBar
- 菜单内容1： menu1Content
- 右导航：rightsideBar
- 菜单容量量：　menuContainer
- 标　语：banner
- 子菜单：submenu
- 边导航图标：sidebarIcon
- 注释：note
- 面包屑：breadCrumb(即页面所处位置导航提示)
- 容器：container

- 功能区：shop(如购物车车，收银台)
- 当前：current

页面结构命名

- 容器： container
- 页头： header
- 内容： content / container
- 页面主题： main
- 页尾： footer
- 导航： nav（navigation的缩写）
- 侧栏： sidebar
- 栏目： column
- 页面外围控制整体布局宽度： wrapper
- 左右中： left / right / center
- 导航： nav

导航命名

- 导航： nav
- 子导航： subnav
- 顶导航： topnav
- 边导航： sidebar
- 左导航： leftsidebar
- 右导航： rightsidebar
- 菜单： menu
- 子菜单： submenu
- 标题： title
- 摘要： summary

功能命名

- 标志： logo
- 广告： banner
- 登录： login
- 登录条： loginbar
- 注册： register
- 搜索： search
- 功能区： shop
- 标题： title
- 加入： joinus
- 状态： status
- 按钮： btn

- 滚动： scrol l
- 标签页： tab
- 文章列列表： list
- 提示信息： msg（message的缩写）
- 当前的： current
- 小技巧： tips
- 图标： icon
- 注释： note
- 指南： guild
- 服务： service
- 热点： hot
- 新闻： news
- 下载： download
- 投票： vote
- 合作伙伴： partner
- 友情链接： link
- 版权： copyright

CSS样式表以及文件命名

- 主要的： master.css
- 模块： module.css
- 基本公用： base.css
- 布局、版面： layout.css
- 主题： themes.css
- 专栏： columns.css
- 文字： font.css
- 表单： forms.css
- 补丁： mend.css
- 打印： print.css