前端开发规范

最佳原则

坚持制定好的代码规范。

无论团队人数多少，代码应该同出一门。

项目命名

全部采用小写方式， 以中划线分隔。例如：my-project-name

文件命名

文件命名参照项目命名规则，多个单词组成时，采用中划线连接方式，比如说: hotel-list.html

有复数结构时，要采用复数命名法，比如说： scripts, styles, images, data-models

文件名中只可由小写英文字母a~z、排序数字0~9或间隔符-组成，禁止包含特殊符号，比如空格、$等

为更好的表达语义，文件名使用英文名词命名，或英文简写。

不允许命名带有广告等英文的单词，例如ad,adv,adver,advertising，防止该模块被浏览器当成垃圾广告过滤掉。任何文件的命名均如此。

HTML规范

缩进使用tab（4个空格）

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；

不要在自动闭合标签结尾处使用斜线，例如<img>而非<img/>

浏览器兼容html5，必须遵循html5规范，优先使用html5属性，例如：<header></header>属性，

HTML5 doctype

HTML5 doctype统一大写 <!DOCTYPE html>。

lang属性

统一用"en", <html lang="en"></html>

type 属性

省略样式表与脚本上的 type 属性。

1. <link rel="stylesheet" href="main.css">
2. <script src="main.js"></script>

脚本引用

如果只需兼容 IE10+，脚本引用写在head标签，并带上 async 属性，它可防止阻塞 DOM 的解析。

兼容老旧浏览器(IE9-)时,，脚本引用写在 body 结束标签之前，并带上 async 属性。

属性顺序

属性应该按照特定的顺序出现以保证易读性；

class

id

name

data-\*

src, for, type, href, value , max-length, max, min, pattern

placeholder, title, alt

aria-\*, role

required, readonly, disabled

class是为高可复用组件设计的，所以应处在第一位；

例：

<a class="..." id="..." data-modal="toggle" href="#">Example link</a>

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；很多时候，需要通过迭代和重构来使HTML变得更少。

<!-- Not well -->

<span class="avatar">

<img src="...">

</span>

<!-- Better -->

<img class="avatar" src="...">

CSS，SASS规范

z-index设置最高为10，弹窗为10，黑色背景遮罩层为9，其他的按照1,2,3....顺序设置。

空格

以下几种情况不需要空格：

* 属性名后
* 多个规则的分隔符','前
* !important '!'后
* 属性值中'('后和')'前
* 行末不要有多余的空格

以下几种情况需要空格：

* 属性值前
* 选择器'>', '+', '~'前后
* '{'前
* !important '!'前
* @else 前后
* 属性值中的','后
* 注释'/\*'后和'\*/'前

/\* not good \*/

.element {

color :red! important;

background-color: rgba(0,0,0,.5);

}

/\* good \*/

.element {

color: red !important;

background-color: rgba(0, 0, 0, .5);

}

/\* not good \*/

.element ,

.dialog{

...

}

/\* good \*/

.element,

.dialog {

}

/\* not good \*/

.element>.dialog{

...

}

/\* good \*/

.element > .dialog{

...

}

/\* not good \*/

.element{

...

}

/\* good \*/

.element {

...

}

/\* not good \*/

@if{

...

}@else{

...

}

/\* good \*/

@if {

...

} @else {

...

}

引号

最外层统一使用双引号；

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使用小写字母，以中划线分隔

### SCSS相关

声明顺序：

* @extend
* 不包含 @content 的 @include
* 包含 @content 的 @include
* 自身属性
* 嵌套规则

@import 引入的文件不需要开头的'\_'和结尾的'.scss'；

嵌套最多不能超过5层；

@extend 中使用placeholder (%) 选择器；

去掉不必要的父级引用符号'&'。

/\* not good \*/

@import "\_dialog.scss";

/\* good \*/

@import "dialog";

/\* not good \*/

.fatal {

@extend .error;

}

/\* good \*/

.fatal {

@extend %error;

}

/\* not good \*/

.element {

& > .dialog {

...

}

}

/\* good \*/

.element {

> .dialog {

...

}

}

### 杂项

不允许有空的规则；

元素选择器用小写字母；

去掉小数点前面的0；

去掉数字中不必要的小数点和末尾的0；

属性值'0'后面不要加单位；

同个属性不同前缀的写法需要在垂直方向保持对齐，具体参照右边的写法；

无前缀的标准属性应该写在有前缀的属性后面；

不要在一个文件里出现两个相同的规则；

用 border: 0; 代替 border: none;；

选择器不要超过4层（在scss中如果超过4层应该考虑用嵌套的方式来写）；

发布的代码中不要有 @import；

尽量少用'\*'选择器。

/\* not good \*/

.element {

}

/\* not good \*/

LI {

...

}

/\* good \*/

li {

...

}

/\* not good \*/

.element {

color: rgba(0, 0, 0, 0.5);

}

/\* good \*/

.element {

color: rgba(0, 0, 0, .5);

}

/\* not good \*/

.element {

width: 50.0px;

}

/\* good \*/

.element {

width: 50px;

}

/\* not good \*/

.element {

width: 0px;

}

/\* good \*/

.element {

width: 0;

}

/\* not good \*/

.element {

border-radius: 3px;

-webkit-border-radius: 3px;

-moz-border-radius: 3px;

background: linear-gradient(to bottom, #fff 0, #eee 100%);

background: -webkit-linear-gradient(top, #fff 0, #eee 100%);

background: -moz-linear-gradient(top, #fff 0, #eee 100%);

}

/\* good \*/

.element {

-webkit-border-radius: 3px;

-moz-border-radius: 3px;

border-radius: 3px;

background: -webkit-linear-gradient(top, #fff 0, #eee 100%);

background: -moz-linear-gradient(top, #fff 0, #eee 100%);

background: linear-gradient(to bottom, #fff 0, #eee 100%);

}

Javascript规范

文档注释

代码要养成注释的习惯，简明扼要,一行注释用//，多行用/\*\*\*\*/

建议在以下情况下使用：

* 所有常量
* 所有函数
* 所有类

/\*\*

\* @func

\* @desc 一个带参数的函数

\* @param {string} a - 参数a

\* @param {number} b=1 - 参数b默认值为1

\* @param {string} c=1 - 参数c有两种支持的取值</br>1—表示x</br>2—表示xx

\* @param {object} d - 参数d为一个对象

\* @param {string} d.e - 参数d的e属性

\* @param {string} d.f - 参数d的f属性

\* @param {object[]} g - 参数g为一个对象数组

\* @param {string} g.h - 参数g数组中一项的h属性

\* @param {string} g.i - 参数g数组中一项的i属性

\* @param {string} [j] - 参数j是一个可选参数

\*/

function foo(a, b, c, d, g, j) {

...

}

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>';

变量命名

标准变量采用驼峰式命名

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 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。

function doSomethingWithItems(items) {

// use one var

var value = 10,

result = value + 10,

i,

len;

for (i = 0, len = items.length; i < len; i++) {

result += 10;

}

}

数组、对象

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};

var a = {

b: 1,

c: 2,

};

// good

var a = {

b: 1,

c: 2

};

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') {

...

}

杂项

对上下文this的引用只能使用'\_this'来命名；

不允许有空的代码块。