前端规范文档

一、概述

1、规范目的

为提高团队协作效率， 便于后台人员添加功能及前端后期优化维护， 输出高质量的文档， 特制订此文档。 本规范文档一经确认， 前端开发人员必须按本文档规范进行前台页面开发。 本文档如有不对或者不合适的地方请及时提出，经讨论决定后方可更改。

2、基本准则

符合 web 标准，语义化 html， 结构、表现、行为分离， 兼容性优良。 页面性能方面， 代码要求简洁明了有序， 尽可能的减小服务器负载， 保证最快的解析速度。

二、目录结构

为方便后台与前端引用路径不产生冲突，目录名应对应的文件功能来命名。同一功能下如有多个子目录，建议分别建立文件放入其中。

示例如下：

-plugins ----------第三方插件目录
    -plugin-name ----------插件
    ...
-images- ----------静态图片目录
    -module1----------模块
        -file-name ----------所属文件
            image-name ----------静态图片
        ...
-scripts ----------脚本目录
    -module1----------模块
        js -name ----------脚本文件
            ...
-styles ----------样式目录
    -module1----------模块
        css -name ----------样式文件
          ...
-views ----------视图目录
    -module1----------模块
        view-name ----------视图文件
            …

三、文件规范

1、命名规范

(1) **一律小写，必须是英文单词或者汉语拼音，以英语单词优先，多个单词以连字符（-）连接。**只能出现小写引文字母，数字和连字符。eg：my-project-name

(2) 有复数结构时，要采用复数命名法，eg：scripts, styles, images, mock-datas。

(3) 很多浏览器会将含有这些词的作为广告拦截： ad、ads、adv、banner、sponsor、gg、guangg、guanggao 等 页面中尽量避免采用以上词汇来命名。

(4) 该命令规范适用于所有前端维护的内容和相关目录。(html, css, js, png, gif, jpg, ico)。

2、HTML 规范

（1）语法

a、使用四个空格的 soft tabs — 这是保证代码在各种环境下显示一致的唯一方式(可以使用 tab 键)。

b、在属性上，使用双引号，不要使用单引号。

c、所有编码均遵循 xhtml 标准，所有标签必须闭合，包括 br (
) 、hr(<hr />) 等；

示例如下：

```
<!DOCTYPE html>
<html>
<head>
    <title>Page title</title>
</head>
<body>
    <img src="images/company-logo.png" alt="Company">
    <span>换行</span><br/>
    <h1 class="hello-world">Hello, world!</h1>
</body>
</html>
```

（2）字符编码　统一使用 UTF-8；

示例如下：

```
<head>
  <meta http-equiv=”Content-Type” content=”text/html; charset=utf-8”/>
</head>
```

（3）充分利用无兼容性问题的 html 自身标签，比如 span 、em 、strong、optgroup、label 等等；

（4）ID 选择器和 Class 选择器命名规则

多个单词组成时，以中划线分隔，流入：my-class、my-id。

【注】：前端样式尽量用 class 来定位加层级，方便维护代码与修改，不麻烦后台人员；

（5）尽可能减少 div 嵌套，保证结构简单明了，根据需求功能正确使用常用标签。对于其曾经结构命名要有层级性、主题性，保证块的复用性（详见 css 规范）；

示例如下：

```
<div class=”box”>
    <div class=”box-content”>现在的积分是：
        <div class=” integration”>9527 </div>
    </div>
 </div>
```

完全可以用以下代码替代:

```
<div class=”box”>
   <p>现在的积分是
      <span>9527</span>
   </p>
< /div>
```

（6）能以背景形式呈现的图片, 尽量写入 css 样式，这样可以减少页面加载量；

（7）特殊符号使用：尽可能使用代码替代，比如 <(<)、>(>)、空格(   )、»(») 等等。

（8）img 标签记得添加 alt 图片说明；

示例如下：

```
<img src="图片.gif" alt="给图片提示">
```

3、css 规范

（1）编码统一为 utf-8，放在 CSS 文件的第一行。

示例如下：@charset "UTF-8";

（2）书写代码前，考虑并提高样式重复使用率；尽量按块结构来封装 css;

（3）充分利用 html 自身属性及样式继承原理减少代码量;

示例如下：

<ul class=”list”>
这儿是标题列表2016

定义 ul.list li{position:relative} ul.list li span{position:absolute; right:0}

（4）对于成对出现的属性要注意相互之间关系，

示例如下：

height：30px;line-height:30px;

height：30px;line-height:28px;border:1px solid black;

ul { position:relative} li { position:absolute }

（5）书写规范要求

a、不要在颜色值 rgb() rgba() hsl() hsla()和 rect() 中增加空格，并且不要带有取值前面不必要的 0 (比如，使用 .5 替代 0.5)。

b、所有的十六进制值都应该使用小写字母，例如 #fff。因为小写字母有更多样的外形，在浏览文档时，他们能够更轻松的被区分开来。

c、尽可能使用短的十六进制数值，例如使用 #fff 替代 #ffffff。

d、不要为 0 指明单位，比如使用 margin: 0; 而不是 margin: 0px。

示例如下：

```
/* Bad CSS */
    .selector, .selector-secondary, .selector[type=text] {
        padding: 15px;
        margin: 0px 0px 15px;
        background-color: rgba(0, 0, 0, 0.5);
        box-shadow: 0 1px 2px #CCC, inset 0 1px 0 #FFFFFF
    }
/* Good CSS */
    .selector,
    .selector-secondary,
    .selector[type="text"] {
        padding: 15px;
        margin-bottom: 15px;
        background-color: rgba(0,0,0,.5);
        box-shadow: 0 1px 2px #ccc, inset 0 1px 0 #fff;
}
```

（6）属性简写

坚持限制属性取值简写的使用，属性简写需要你必须显式设置所有取值。常见的属性简写滥用包括:padding、 margin、 font、 background、 border、 border-radius

示例如下：

```
/* Bad example */
    .element {
        margin: 0 0 10px;
        background: red;
        background: url("image.jpg");
        border-radius: 3px 3px 0 0;
}
/* Good example */
    .element {
        margin-bottom: 10px;
        background-color: red;
        background-image: url("image.jpg");
        border-top-left-radius: 3px;
        border-top-right-radius: 3px;
}
```

（7）为了兼容 IE8，在 css 中不要使用 nth-child、last-child 来定位，可以使用 first-child；

4、JavaScript 规范

（1）缩进、分号、单行长度

a、使用四个空格的 soft tabs — 这是保证代码在各种环境下显示一致的唯一方式(可以使用 tab 键)。

b、Statement（声明）之后一律以分号结束，不可以省略。

（2）变量命名

a、标准变量采用小驼峰

b、常量采用大写字母，下划线连接的方式

c、构造函数，大写第一个字母

示例如下：

```
var thisIsMyName;
var MAX_COUNT = 10;
```

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

（3）字符串赋值统一使用单引号

示例如下：

```
var str = 'aaa';
```

（4）总是使用 var 来声明变量，如果不这么做将导致产生全局变量，我们要避免污染全局命名空间。

示例如下：

```
// bad
superPower = new SuperPower();
// good
var superPower = new SuperPower();
```

（5）分号 语句结束一定要加分号。

（6）建议使用单行注释；

5、jQuery

（1）缓存变量 DOM 遍历是昂贵的，所以尽量将会重用的元素缓存。

示例如下：

```
// 糟糕
h = $('#element').height();
$('#element').css('height',h-20);
// 建议
$element = $('#element');
h = $element.height();
$element.css('height',h-20);
```

（2）避免全局变量，写全局变量时必须先声明。

jQuery 与 javascript 一样，一般来说,最好确保你的变量在函数作用域内。

示例如下：

```
//糟糕
$element = $('#element');
h = $element.height();
$element.css('height',h-20);
// 建议
var $element = $('#element');
```

```
var h = $element.height();
$element.css('height',h-20);
```

（3）在变量前加$前缀，便于识别出 jQuery 对象。

示例如下：

```
// 糟糕
var first = $('#first');
var second = $('#second');
var value = $first.val();
// 建议 - 在 jQuery 对象前加$前缀
var $first = $('#first');
var $second = $('#second'),
var value = $first.val();
```

（4）请使用’On’

在新版 jQuery 中，更短的 on(“click”) 用来取代类似 click() 这样的函数。在之前的版本中 on() 就是 bind()。自从 jQuery 1.7 版本后，on()?附加事件处理程序的首选方法。然而，出于一致性考虑，你可以简单的全部使用 on()方法。

示例如下：

```
// 糟糕
$first.click(function(){
    $first.css('border','1px solid red');
    $first.css('color','blue');
});
$first.hover(function(){
    $first.css('border','1px solid red');
});
// 建议
$first.on('click',function(){
    $first.css('border','1px solid red');
    $first.css('color','blue');
});
$first.on('hover',function(){
    $first.css('border','1px solid red');
});
```

```
$(document).on('click', $first, function(event) {
    $(this).hide();
});
```

（5）使用子查询缓存的父元素

正如前面所提到的，DOM 遍历是一项昂贵的操作。典型做法是缓存父元素并在选择子元素时重用这些缓存元素。

示例如下：

```
// 糟糕
var
    $container = $('#container'),
    $containerLi = $('#container li'),
    $containerLiSpan = $('#container li span');
// 建议 (高效)
var
    $container = $('#container '),
    $containerLi = $container.find('li'),
    $containerLiSpan= $containerLi.find('span');
```

（6）避免隐式通用选择符

通用选择符有时是隐式的，不容易发现。

示例如下：

```
// 糟糕
$('.someclass :radio');
// 建议
$('.someclass input:radio');
```