前端开发命名规范文档整理（初稿）

(PS:网络上面流传着各种前端规范,在这里突然萌生了想把比较好的规范整理出来，so 有了类似下面的东东。)

作者：杭州-小白

博客：http://www.xiaobai8.com .

Javascript 命名规范

一．变量命名：

变量名包括全局变量，局部变量，类变量，函数参数等等，他们都属于这一类。

变量命名都以类型前缀+有意义的单词组成，用驼峰式命名法增加变量和函式的可读性。

例如：sUserName，nCount。

前缀规范：

1.每一个局部变量都需要一个类型前缀，按照类型分为：

s：表示字符串。例如：sName，sHtml;

n：表示数字。例如：nPage，nTotal;

b：表示逻辑。例如：bChecked，bHasLogin;

a：表示数组。例如：aList，aGroup;

r：表示正则表达式。例如：rDomain，rEmail;

f：表示函数。例如：fGetHtml，fInit;

o：表示以上未涉及到的其他对象，例如：oButton，oDate;

g：表示全局变量，例如：gUserName，gLoginTime;

2.当然根据团队的不同，打造自己合适的规范：

$：表示 Jquery 对象。例如：$Content，$Module;

一种比较广泛的 Jquery 对象变量命名规范。

j：表示 Jquery 对象。例如：jContent， jModule;

另一种 Jquery 对象变量命名方式。

fn：表示函数。例如：fnGetName，fnSetAge;

和上面函数的前缀略有不同，改用 fn 来代替，个人认为 fn 能够更好的区分普通变量和函数变量。

dom：表示 Dom 对象，例如：domForm，domInput;

项目中很多地方会用到原生的 Dom 方法及属性，可以根据团队需要适当修改。

3.可以根据项目及团队需要，设计出针对项目需要的前缀规范，从而达到团队开发协作便利的目的。

1：作用域不大临时变量可以简写，

比如：str，num，bol，obj，fun，arr。

2：循环变量可以简写，比如：i，j，k 等。

3：某些作为不允许修改值的变量认为是常量，全部字母都大写。例如：COPYRIGHT，PI。常量可以存在于函数中，也可以存在于全局。

备注：为什么需要这样强制定义变量前缀？正式因为 javascript 是弱语言造成的。在定义大量变量的时候，我们需要很明确的知道当前变量是什么属性，如果只通过普通单词，是很难区分的。

```
//普通代码
var checked =false;
var check =function() {
    return true;
}
/**
some code
**/
if(check) {//已经无法很确切知道这里是要用 checked 还是 check()从而导致逻辑错误
    //do some thing
}
//规范后代码
var bChecked =false;
var fnCheck =function() {
    return true;
}
/**
some code
**/
if(bChecked) {
    // do some thing
}
if(fnCheck()) {
    // do other thing
}
```

上面的例子很容易看出，强制添加前缀的好处了吧。。。Ok。我们继续下面的整理。

二．函数命名：

统一使用动词或者动词+名词形式，

例如：fnGetVersion()，fnSubmitForm()，fnInit();

涉及返回逻辑值的函数可以使用 is，has，contains 等

表示逻辑的词语代替动词，例如：fnIsObject()，fnHasClass()，fnContainsElment().

如果有内部函数:

使用_fn+动词+名词形式，内部函数必需在函数最后定义。例如：

```
function fnGetNumber(nTotal) {
    if(nTotal < 100) {
        nTotal = 100;
    }
    return _fnAdd(nTotal);

    function _fnAdd(nNumber) {
        nNumber++;
        return nNumber;
    }
}
alert(fGetNumber(10));//alert 101
```

对象方法与事件响应函数：

对象方法命名使用 fn+对象类名+动词+名词形式；例如 fnAddressGetEmail()，主观觉得加上对象类名略有些鸡肋，个人认为一个对象公开的属性与方法应该做到简洁易读。多增加一个对象类名单词，看着统一了，但有点为了规范而规范的味道，这里根据自身喜好仁者见仁智者见智吧。(PS:根据实际情况而定。)

某事件响应函数命名方式为：fn+触发事件对象名+事件名或者模块名

例如：fnDivClick()，fnAddressSubmitButtonClick()

补充一些函数方法常用的动词：

get 获取/set 设置，　add 增加/remove 删除

create 创建/destory 移除　start 启动/stop 停止

open 打开/close 关闭，　read 读取/write 写入

load 载入/save 保存，create 创建/destroy 销毁

begin 开始/end 结束，backup 备份/restore 恢复

import 导入/export 导出，　split 分割/merge 合并

inject 注入/extract 提取，　attach 附着/detach 脱离

bind 绑定/separate 分离，　view 查看/browse 浏览

edit 编辑/modify 修改，　select 选取/mark 标记

copy 复制/paste 粘贴，　undo 撤销/redo 重做

insert 插入/delete 移除，　add 加入/append 添加

clean 清理/clear 清除，　index 索引/sort 排序

find 查找/search 搜索，　increase 增加/decrease 减少

play 播放/pause 暂停，　launch 启动/run 运行

compile 编译/execute 执行，debug 调试/trace 跟踪

observe 观察/listen 监听，　build 构建/publish 发布

input 输入/output 输出，encode 编码/decode 解码

encrypt 加密/decrypt 解密，compress 压缩/decompress 解压缩

pack 打包/unpack 解包，　parse 解析/emit 生成

connect 连接/disconnect 断开，send 发送/receive 接收

download 下载/upload 上传，refresh 刷新/synchronize 同步

update 更新/revert 复原，　lock 锁定/unlock 解锁

check out 签出/check in 签入，submit 提交/commit 交付

push 推/pull 拉，　expand 展开/collapse 折叠

begin 起始/end 结束，start 开始/finish 完成

enter 进入/exit 退出，　abort 放弃/quit 离开

obsolete 废弃/depreciate 废旧，　collect 收集/aggregate 聚集

上面讨论了基本的 JS 书写命名规范，按我个人看法，只要能够按照上面的规范写出来的代码，一般不会太糟糕，最不济没人会说你代码乱的难以阅读。代码规范对于大团队的维护建设是毋庸置疑的，当然对于个人的代码素养也是很有帮助的，希望你能够通过上文能够加强你的代码规范，写出易读易维护的代码。(PS:引用海玉博客，作者原话，但是这个也是我想表达的意思。没有好的规范写不出好的代码。)

CSS 规范

本规范概述

制定本规范的目的在于使我们的 CSS 代码更加易于维护和重用，从而提升效率执行本规范时建议的流程 建议使用 D(esign)C(oding)D(ebug)V(alidate)R(oundup)，即 DCDVR 的流程。首先需要规划样式并分为共有样式和页面个性化样式，然后才开始编码，编码的同时进行 Debug，Validate 和代码片断的总结，而不是在所有模板都完成后才进行这三个步骤。

一．关于样式库构建规范：

可以看原文：http://aliceui.org/docs/rule.html

二．selector 命名规范：

1.命名除 .fn- / .ui- / .sl- 外，可自定义命名。

请慎用 selected current disabled first last success error

2.一般情况下，如果命名比较通用，比如 current，请限定在相应的上下文环境中。比如其父节点 ID 为#parent 等比较通用的命名，建议写成 #parent .current{}，而非 .current{}，即使是为了重用，也应该注意。只有在非常明确不会影响到其他组件工作，并且其他人不会写这种命名的情况下，才让它变成全局通用的。

3.作为 JS 接口的 class 或者 ID，必须是以 J- 前缀开头的。除 JS 接

口命名外，其他命名一律使用小写字母

三．注释：

1.组件注释:

```
/**
 * @name: 组件名
 * @overview: 组件介绍
 * @require: 依赖的样式
 * @author: 小鱼(sofish@163.com)
 */
```

2.块状或行内元素，都请使用 /* comment */ 注释，注释文字前后端保持各有一个空格

3.为了您的体验着想，一目了解的代码，就不要注释了，比如“display:none; /* 让元素看不见 */” 工作还有很多啊，同学，请为了自己的体力着想。

四．CSS 文件

1.文件编码必须使用 utf-8（无 BOM）

2.文件一律通过 link 链入 (NOT @import)

3.当只是单个页面使用时，才写在 <head> 的 <style> 中

五．Hack 规则

1.一般情况下，不要使用 IE 条件注释： <!－[if IE]><![endif]－>

2.通用 Hack:

```
all-IE{property:value\9;}
:root .IE-9{property:value\0/;}
.gte-IE-8{property:value\0;}
.lte-IE-7{*property:value;}
.IE-7{+property:value;}
.IE-6{_property:value;}
.not-IE{property//:value;}
@-moz-document url-prefix() { .firefox{property:value;} }
@media all and (-webkit-min-device-pixel-ratio:0) { .webkit{property:value;} }
@media all and (-webkit-min-device-pixel-ratio:10000),not all and (-webkit-min-device-pixel-ratio:0) { .opera{property:value;} }
@media screen and (max-device-width: 480px) { .iphone-or-mobile-s-webkit{property:value;} }
```

3.当然，强烈建议你使用更优雅的 hack 方式。那就是避免 hack。或者在书写上，做点小 trick。比如：

```
.selector .child{property:value;} /* for ie-6 */
.selector > .child{property:value;} /* except ie-6 */
```

关于 Hack: 在 firefox 写完，IE 有问题？还是其他浏览器也出现了。你知道 IE Hack 能解决。我想，你也可能知道，用其他方法也能绕过。建议少用 Hack。

(PS:关于 hack 这个东西，作者也说的挺对的，其实能少用就少用，能够避免就避免哦)

六．书写规范：

1. 以如下写法为例：

第一种方式：（强烈推荐）

```
/* 区域模块-1 */
.tech, .ued{
    background:#f60 url(alipay.com/orange.png) no-repeat 0 0;
}
/* 区域模块-2 */
.tech{
    width:950px;
    margin:0 auto;
}
.tech .wd{
    width:620px;
    float:left;
}
```

第二种方式：（如果让其他人看，请先格式化）

```
/* 区域模块-3 */
.ued{width:100%;padding:30px 50px;}
.ued .visual{display:inline-block;font:700 normal 12px/1.5 arial;}
```

2. 非常重要，需要你注意的点：

区域模块间用空行分隔

A.多个选择器写一起时，逗号（,）后紧跟一个空格

B.避免行内样式，即使是 JS，也应该尽量使用 class/ID 来决定显示，而非行内样式

C.避免使用低效选择器

(所以,别滥用 class;当然,不滥用 ID 我相信你是知道的)

详情请见：

http://code.google.com/speed/page-speed/docs/rendering.html

D.尽量使用缩写，比如 font:700 normal 12px/1.5 arial; 一般不要写成:

```
font-style:normal;
font-size:12px;
line-height:1.5;
font-family:arail;
```

E.通常需要使用缩写规则的:

如:

padding-top:1px;

padding-right:2px;

padding-bottom:3px;

padding-left:4px;

简写：padding:1px 2px 3px 4px; 记忆顺序上右下左

margin: 同上

如使用像’red’这样的颜色名，采用小写；16 进制写法使用大写；

写 3 位，还是 6 位，自便：

color: red; color;#FFF; color:#ABCABC

background: color imageUrl repeat attachment position;

如:

background:#ddd url(alipay-wd.png) no-repeat scroll 10px 20px;

border: size style color;

如: border:1px solid #ddd;

```
font: weight variant style size/lineHeight family;
```

如 font:700 small-caps italic 12px/1.5 "Courier New";

font-weight 统一用 500 代替 normal, 用 700 代替 bold;

```
list-style: type position image;
```

如：list-style: square inside url(alipay-wd.png);不过，通常我们要使用的是 list-style:none;

CSS3 书写规范：浏览器私有写法在前，标准写法在后

```
-moz-box-shadow: 1px 2px 3px #ddd;
-webkit-box-shadow: 1px 2px 3px #ddd;
box-shadow: 1px 2px 3px #ddd;
```

(PS:上面代码原著可能因为编辑器的问题。这里纠正下.)

1.书写顺序

不强制书写顺序。但我们应该养成良好的习惯，让看代码的人更易理解。易读对于团队协作来说是非常重要的：框架为先，细节次之比如写一个浮动容器的样式，我们应该先让这个容器的框架被渲染出来，让大家看到基本的网站框架。然后再再去渲染容器里面的内容。最终呈现给用户. 通常像 color font padding 之类的，写在后面。

```
.selector{float:left;width:300px;height:200px;font-size:14px;color:#f36;}
```

2.有因才有果

比如想使用”图片替换文字”技术，通常要使用的 text-indent。如果我们使用标签的是 span：这个文字将被图片替换我们应该是先将 span 变成”块级元素”（使用 display:block，虽然他永远不是块级元素），再将文字 indent。而不是先 indent 再变成块级的：

```
.thepic{display:block;text-indent:-9999em;}
```

又如我们，如果想让一个 span 使用 margin，那么我们应该这样写:span{display:block;margin-bottom:10px;} 而非 span{margin-bottom:10px;display:block;}。

因为没有 display 之前，行内元素是没有 margin 的。

(PS:作者的这个有因才有果，一直是我所推崇的，东西都有因果。)

七. IE 私有特性

1. expression

记住一句话：无论什么时候，都不要使用它。用 Javascript 吧。更优雅，更灵活。

2. filter

应该尽量避免使用 AlphaImageLoader

可以适当在投影/发光/渐变/去色方面上使用

3. IE bug

常见 BUG，详见：http://sofish.de/1400

(PS:IE 私有属性很多，作者这里列举的是最常见的。)

八．如何规划你的 CSS 文件结构（PS:大餐来咯!）

1.一定要有全局设置

全局设置可以避免重复书写同样的东西。比如 3 人在一个项目中，假设项目中涉及到的链接有 10 种颜色，如果有全局重设，我们就可以统一设置颜色。如果没有，我们可能每个都会为自己所负责部分的链接定义相应的颜色。这样一来就导致多处定义，且定义不统一。以后维护需要到各个地方都改。这是很麻烦的事。另外就是 css 文件也会因此变大。所以，在写之前，大家应该先分析视觉稿，统一全局设置。

```
/* global reset */
body{padding:0;margin:0;font.....}
a{color:#07f;}
a:hover{color:#555;}
```

（PS：reset 样式，相信大家都知道）

需要注意的是，一般情况下，不要直接给标签写样式。而应该使用 class。像下面这种写法，并不是很合适:(PS：真心觉得不合适)

```
h1{font-size:30px}
h2{font-size:20px}
h3{font-size:10px;}
```

如果有另外一个 h2 也要使用 10px 的，而其他的都仍使用 20px 的，那可就不好办了。所以，推荐用这种方法：

```
/* global classes */
.text-size30{font-size:30px;}
.text-size20{font-size:20px;}
.text-size10{font-size:10px;}
```

(PS:这种做法，也是很多大牛推荐的写法，高度重用)

<h2 class="text-size20">… <h2 class="text-size10">

2.一定要模块化

有两点需要注意的，一是，注意代码重用的模块化；一是，注意 HTML 结构的模块化，而不是分块。

我们是这样重用的：

```
<div id="module-1" class="module">
    <h3>TITLE</h3>
    <p class="module-item">
        some text
    </p>
</div>
<div id="module-2" class="module">
    < h3>TITLE</h3>
    <p class="module-item">
        some text
    </p>
</div>
```

```
/* module, reuse style in module scrope*/
.module{}
.module-status{}
.module-item{}
.module-status .module-1-item{}

/* customize */
#module-1 .module-item{}
#module-2 .module-item{}
```

（PS:亲们仔细看这个代码，就知道了代码模块化了。如果要区别不同，就用 ID 的。要慢慢理解这个）

HTML 的模块化：我们应该这样写

（他们的视觉是一体的，代码也应该是一体的）：

```
<div id="module-1" class="module">
<h3>TITLE</h3>
<p class="module-item">
    some text
</p>
</div> <!-- #module .module -->
```

而不要这么写：

```
<h3>TITLE</h3>
<!-- 第一块 -->
<div id="module-1" class="module">
    <p class="module-item">
        some text
    </p>
</div> <!-- 第二块 -->
```

九．值得注意的事

1.Background Color:

一般我们都会写:

如：background:url(path/to/image) no-repeat 0 0;

当元素背景是深色的时候，比如#000，我们通常会选择一种比较浅的颜色来做为文本的颜色，比如#fff，为了避免网速缓慢导致 CSS 已经加载，而图片仍未加载完成或图片服务器挂掉时文本不可见，请尽量使用加上 CSS 定义的背景颜色，

如：background:#e8edef url(path/to/image) no-repeat 0 0;

2. has Layout

别用轻易使用 hack，IE 下很多兼容性问题都是 has Layout 引起的。试着给元素加上:

```
display: inline-block
height: (除 auto 外任何值)
```

width: (除 auto 外任何值)

float: (left 或 right)

position: absolute

writing-mode: tb-rl

zoom: (除 normal 外任意值)

3. 代码测试:

一般情况下，我们需要测试通过的是电脑端的 A-Grade 浏览器：

	Win XP	Win 7	Mac 10.6.†	iOS 3.†	iOS 4.†	Android 2.2.†
Safari 5.†			A-grade			
Chrome † (latest stable)	A-grade					
Firefox 4.†		A-grade (upon GA release)	A-grade (upon GA release)			
Firefox 3.6.†	A-grade	A-grade	A-grade			
IE 9.0		A-grade (upon GA release)				
IE 8.0	A-grade	A-grade				
IE 7.0	A-grade					
IE 6.0	A-grade					
Safari for iOS				A-grade	A-grade	
WebKit for Android OS						A-gradea

更多关于这个列表的资料，请看：

http://developer.yahoo.com/yui/articles/gbs/

总结

也许大家会有疑问，看到这个就是 alice（阿里）的规范呀。我为什么要引用他们的规范呢。我个人觉得他们的规范从某种意义上来说，影响着很多前端，在国内他们走到了前沿，而且他们的规范代码真心觉得挺容易就看懂了，不会觉得很乱，整洁,高效,可扩展性，可维护性都还可以的。。。。。规范规范其实说白了，就是为了提高团队开发效率，和减少后期维护成本，真心觉得他们的规范很不错的，可以借鉴，同样很多大牛的博客也介绍了规范。。。

……搜集比较好的规范，吸收之……

……期待下次添加更好的规范到这个里面来,持续整理中……

最后更新时间：2012-09-07

作者：杭州-小白

博客：http://www.xiaobai8.com