代码规范

CCC团队纸膜涂鸦

1. **规范目的**

为提高团队协作效率，便于后期优化维护，输出高质量的文档，特制定此文档，请开发人员必须按照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。

1. **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，尽可能的减少服务器负载，保证最快的解析速度。

1. **文件规范**
2. html，css，js，images文档均归档至约定的目录中。
3. html文件命名：英文命名，后缀.html。
4. css文件命名：英文命名，后缀.css。共用base.css，其他页面依实际模块需求命名。
5. js文件命名：英文命名，后缀.js。共用common.js，其他依实际模块需求命名。
6. **Html书写规范**
7. 文档类型声明及编码：统一为HTML5声明类型<!DOCTYPE html>；编码统一为

<meta charset="UTF-8">，书写时利用IDE实现层次分明的缩进。

1. 非特殊情况下样式文件必须外链至<head>…</head>之间；
2. 引入JS库文件，文件名须包含库名称及版本号及是否为压缩版，此项目中固定库为：

jquery-1.8.2.js、Three-57.js，其他库跟据情况而定；引入库的插件，文件名格式为：库名称+插件名称.js。比如jQuery.cookie.js；

4. 所有编码均遵循XHTML标准，标签、属性、属性命名必须有小写字母及下划线数字组成，且所有标签必须闭合，包括br(<br/>)，hr(<hr/>)等；属性值必须用双引号包括；

5. 语义话HTML，页面结构必须使用HTML5新标签，尽可能减少div嵌套；

6. 书写链接地址时，必须避免重定向，例如：href=<http://localhost/>，即须在URL地址后面加上”/”；

7. 在页面中尽量避免使用style属性

**8. 给区块代码及重要功能加上注释，方便管理。**

9. 书写页面过程中，请考虑向后扩展性；

10. class&id参考css书写规范

1. **Css书写规范**

1. 编码统一为utf-8；

2. class与id的使用：id是唯一的并是父级的，class是可以重复的并是子级的，所有id仅使用在大的模块上，class可用在重复使用率高及子级中；

3. 避免class与id命名重复

4. css属性书写顺序，建议遵循：布局定位属性🡺自身属性🡺文本属性🡺其他属性。尽量保证同类属性写在一起。参考：布局定位属性：display&list-style&position&float&clear&visibility&overflow；自身属性：width&height&margin&padding&border&background；文本属性：color&font&text-decoration&text-align&white-space&；其他属性：content；列出的属性指数最常见到的。

5. 考虑并提高样式的重复使用率。

6. 充分利用html自身属性及样式继承原理减少代码量。

7. 尽量使用css sprite技术，减少http请求

**8. 必须为大区块样式添加注释，小区快适量注释；**

1. **JavaScript书写规范**

1. 文件编码统一为utf-8，书写过程，每行代码结束必须有分号；所有功能跟据项目原生开发，避免网上down下来的代码造成代码污染。

2. 变量命名：骆峰式命名。原声JavaScript变量要求是纯英文字母，首字母须小写；jQuery变量要求首字母为’J\_’，局部变量前边加两个(\_)。Three变量要求首字母为’D\_’，局部变量前面加两个下划线(\_),其他与原声JavaScript规范相同；

3. 类命名：首字母大写；

4. 函数命名：首字母小写；

5. 命名语义话，尽可能利用英文单词或其缩写；

6. 尽量避免使用存在消耗资源的方法或者属性

7. 代码结构明了，加适量注释，提高函数重用率

1. **图片规范**

1. 命名全部使用小写英文字母 || 数字 || \_ 的组合，其中不得包含汉字 || 空格 || 特殊字符；尽量用易懂的词汇，便于团队其他成员理解；

2. 图片格式仅限png

3. 所有页面元素类图片均放入images文件夹。

1. **其他规范**

1. 开发工程中严格按分工完成页面，以提高css利用率，避免重复开发；

2. 减少沉冗代码，书写所有人都可以看懂的代码。简洁易懂。

1. **注释规范：** a、html注释：<!-- … -->，注释在块的开始位置

b、css注释： 注释在开始位置

c、JavaScript注释：注释在开始位置

效率考虑建议：

1. canvas绘图时避免使用小数，如果遇到小树坐标，要先将其准换为整数在进行绘图

2. drawImage比putImageData效果好

3. drawImage绘图时，绘制的区域越小，速度越快，而图片的大小对效率影响不大。

4. png格式图片绘制效果相对好些。

5. 用**位运算**代替代码中的所有运算。

6. 少用Math静态类，使用位元算。

7. 优化算法。