CSS规范

CSS遵循可读性强，便于维护的原则

代码规范

样式表编码!

样式文件必须写上 @charset 规则，并且一定要在样式文件的第一行首个字符位置开始写，编码名用“UTF-8”。

```
@charset "UTF-8";
```

代码风格

代码格式化

样式书写一般有两种：一种是紧凑格式 (Compact)， 一种是展开格式（Expanded），统一使用展开格式书写样式

```
/* 紧凑 */
.nav{ display: block;width: 50px;}


/* 展开 */
.nav {
    display: block;
    width: 50px;
}
```

代码大小写-tsap

样式选择器，属性名，属性值关键字全部使用小写字母书写，并使用“-”连接。

```
/* 不推荐 */
.navList{
        DISPLAY:BLOCK;
}

/* 推荐 */
.nav-list {
        display:block;
}
```

常用方式-bfc

CSS样式常用的为三种方式：内联（行间）、内嵌、外链

```
<!-- 推荐 外链接式 -->
<head>
  <link rel="stylesheet" type="text/css" href="css/style.css" />
</head>

<!-- 内嵌式 适用于小页面 -->
<head>
  <style type="text/css">
    html,body{
                margin: 0;
                padding: 0;
        }
  </style>
</head>

 <!-- 不推荐 行内样式，优先级太高、HTML不纯净 -->
 <p style="color: red;">我是P段落</p>

 <!-- 不推荐 导入式，相比link太慢 -->
<head>
  <style type="text/css">
    @import url("css/style.css");
  </style>
</head>
```

- 推荐使用类选择器，少用ID选择器!
  ID在一个页面中的唯一性导致了如果以ID为选择器来写CSS，就无法重用。

选择器-tsap

- 尽量少用通用选择器 *
- 不使用 ID 选择器
- 不使用无具体语义定义的标签选择器
- 尽量不使用!important

```
/* 推荐 */
.nav {}
.nav li {}
.nav li p{}

/* 不推荐 */
*{}
#nav {}
.nav div{}
```

代码缩进

统一使用四个空格进行代码缩进，使得各编辑器表现一致（各编辑器有相关配置）。

```
.nav {
    width: 100%;
    height: 100%;
}
```

分号-tsap

每个属性声明末尾都要加分号；

```
.nav {
    width: 100%;
    height: 100%;
}
```

代码易读性

- 左括号与类名之间一个空格，冒号与属性值之间一个空格
- 逗号分隔的取值，逗号之后一个空格
- 为单个css选择器或新申明开启新行
- 颜色值 rgb() rgba() hsl() hsla() rect() 中不需有空格，且取值不要带有不必要的 0
- 属性值十六进制数值能用简写的尽量用简写
- 不要为 0 指明单位-tsap

```
/* 不推荐 */
.element {
background-color: rgba(25, 25, 25, 0.6);
    transition: linear 0.8s;
}

/* 推荐 */
.element {
    background-color: rgba(25, 25, 25, .6);
        transition: linear .8s;
}

/* 不推荐 */
.element {
        margin: 0rem;
        border-width: 0px;
}

/* 推荐 */
.element {
        margin: 0;
        border-width: 0;
}
```

属性值引号

css属性值需要用到引号时，统一使用单引号

```
/* 推荐 */
.nav {
        font-family: 'Hiragino Sans GB';
}

/* 不推荐 */
.nav {
        font-family: "Hiragino Sans GB";
}
```

属性书写顺序-tsap

建议遵循以下顺序：

1. 布局定位属性：display / position / float / clear / visibility / overflow
2. 自身属性：width / height / margin / padding / border / background
3. 文本属性：color / font / text-decoration / text-align / vertical-align / white- space / break-word
4. 其他属性（CSS3）：content / cursor / border-radius / box-shadow / text-shadow / background:linear-gradient …

5. CSS3 浏览器私有前缀在前，标准前缀在后

```
.nav {
    display: block;
    position: relative;
    float: left;
    width: 100px;
    height: 100px;
    margin: 0 10px;
    padding: 20px 0;
    font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;
    color: #333;
    background: rgba(0,0,0,.5);
    -webkit-border-radius: 10px;
    -moz-border-radius: 10px;
    -o-border-radius: 10px;
    -ms-border-radius: 10px;
    border-radius: 10px;
}
```

换行-bfc

对于组合的选择器，各选择器单独占一行，利于阅读和修改

```
/* 不推荐 */
.item, .item-secondary, .item[type=text] {
  padding: 15px;
  margin: 0px 0px 15px;
}
/* 如果需要中途修改选择器，可能会有漏改 */
.menu .item, .menu .item-secondary, .item[type=text] {
  padding: 15px;
  margin: 0px 0px 15px;
}

/* 推荐 */
.menu .item,
.menu .item-secondary,
.menu .item[type="text"] {
  padding: 15px;
  margin-bottom: 15px;
}
```

对于只包含一条声明的样式，为了易读性和便于快速编辑，可以将语句放在同一行。花括号{}内两侧都需要一个空格。

```
/* 临近的样式花括号可对齐 */
.icon                   { background-position: 0 0; }
.icon-home              { background-position: 0 -20px; }
.icon-account   { background-position: 0 -40px; }
```

简写形式的属性声明-bfc

需要显示地设置所有值的情况下，尽量使用简写形式的属性声明。如果只需要指定部分属性，不建议使用简写形式。过度使用简写形式会导致代码混乱，并且会对属性带来不必要的覆盖，或许会引起意外的副作用。
[padding，margin，background，border，border-radius]

```
/* 简写形式设置所有属性 */
.element {
        margin: 0 0 10px;
        border: 1px solid #333;
        background: #00FF00 url(bgimage.gif) no-repeat fixed top;
        border-radius: 3px 3px 0 0;
}

/* 设置部分属性 */
.element {
        margin-bottom: 10px;
        border-color: #666;
        border-top-left-radius: 3px;
        border-top-right-radius: 3px;
        background-image: url("image.jpg");
}
```

后代选择器命名-bfc

后代选择器不需要完整表现结构树层级，尽量简短。
通过使用后代选择器的方法，你不需要考虑他的命名是否已被使用，因为他只在当前模块或元件中生效，同样的样式名可以在不同的模块或元件中重复使用，互不干扰；在多人协作或者分模块协作的时候效果尤为明显
注：后代选择器不要单独使用，因为污染的可能性较大；

```
/* 不推荐完整的结构树层级 */
.m-list { margin: 0; }
.m-list .m-list-itm { margin: 1px; }
.m-list .m-list-cnt { margin-left: 100px; }

/* 推荐 简洁写法，这里的.itm和.cnt只在.m-list中有效 */
.m-list { margin: 0; }
.m-list .itm { margin: 1px; }
.m-list .cnt { margin-left: 100px; }
```

媒体查询

尽量将媒体查询的规则靠近与他们相关的规则，不要将他们一起放到一个独立的样式文件中，或者丢在文档的最底部，这样做只会让大家以后更容易忘记他们

```
.element {
    ...
}
.element-avatar{
    ...
}

/* 推荐 媒体查询尽量靠近与其相关的规则 */
@media (min-width: 480px) {
    .element {
        ...
    }
    .element-avatar {
        ...
    }
}
```

注释规范-tsap

单行注释

- 注释以字符 /* 开始，以字符 */ 结束
- 注释不能嵌套

```
/* 这是一行单行注释 */
.element {
    ...
}
```

模块注释

注释内容第一个字符和最后一个字符都是一个空格字符，/* 与 模块信息描述占一行，多个横线分隔符-与*/占一行，行与行之间相隔两行。
推荐：

```
/* Module A
-------------------------------------------------------------- */
.mod_a {}


/* Module B
-------------------------------------------------------------- */
.mod_b {}
```

不推荐：

```
/* Module A ------------------------------------------------- */
.mod_a {}
/* Module B ------------------------------------------------- */
.mod_b {}
```

文件信息注释

在样式文件编码声明 @charset 语句下面注明页面名称、作者、创建日期等信息。

```
@charset "UTF-8";
/**
 * @desc File Info
 * @author Author Name
 * @date 2015-10-10
 */
```