《慧择网前端开发规范》

版本：V1.0.1

慧择前端团队 2015-04-15

深圳市慧择保险经纪有限公司

目录

1. 概述

为提高团队协作效率，便于后期优化维护扩展，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，经讨论决定后可以更改此文档。

2. 阅读对象

所有慧择前端技术开发人员。

3. 基本准则

代码符合 W3C 标准，结构、表现、行为分离，代码语义化，逻辑清晰，层次分明，易读，易扩展，兼容所有主流浏览器，尽可能的减小服务器负载，保证最快的解析速度。

4. 文件规范

文件夹命名规范

1. 语义化，简单明了；
2. 多个单词使用下划线“-”连接；

文件命名规范

1. 语义化，简单明了；
2. 多个单词使用下划线“_”连接；
3. 体现层级或插件相关使用“.”连接，连接版本使用中划线“-”；
   如：jquery.slide-1.0.js

5. 编码规范

所有文件一律采用 UTF-8 编码方式。

6. 基础框架规范

一、 Javascript

PC 端：jQuery （版本：V1.8.0）+ Requrejs（版本：V2.1.11）；

移动端：Zepto （版本：V1.1.6）+ Requrejs（版本：V2.1.11）；

二、 CSS

PC 端：normalize.css（版本：V1.0.0）；

移动端：normalize.css（版本：V3.0.0）；

7. HTML 规范

书写规范

符合 W3C 标准，结构简洁，清晰，合逻辑，层次分明，逻辑模块之间添加适当的注释。

DocType 声明

为每个 HTML 页面的第一行添加标准模式（standard mode）的声明，这样能够确保在每个浏览器中拥有一致的展现。

语言属性

根据 HTML5 规范： 强烈建议为 html 根元素指定 lang 属性，从而为文档设置正确的语言。这将有助于语音合成工具确定其所应该采用的发音，有助于翻译工具确定其翻译时所应遵守的规则等等。

字符编码

通过明确声明字符编码，能够确保浏览器快速并容易的判断页面内容的渲染方式。这样做的好处是，可以避免在 HTML 中使用字符实体标记（character entity），从而全部与文档编码一致（一般采用 UTF-8 编码）。

IE 兼容模式

IE 支持通过特定的 <meta> 标签来确定绘制当前页面所应该采用的 IE 版本。除非有强烈的特殊需求，否则最好是设置为 edge mode，从而通知 IE 采用其所支持的最新的模式。

引入 CSS 和 JavaScript 文件

1. 根据 HTML5 规范，在引入 CSS 和 JavaScript 文件时一般不需要指定 type 属性，因为 text/css 和 text/javascript 分别是它们的默认值。
2. CSS 在头部<head></head>中引用，JS 原则上放在页面的后面，即 body 结束之前</body>。

属性顺序

class
id, name
data-*
src, for, type, href
title, alt
aria-*, role
class 用于标识高度可复用组件，因此应该排在首位。id 用于标识具体组件，应当谨慎使用（例如，页面内的书签），因此排在第二位。一个页面只允许出现一个名字相同的 ID。

SEO 规范

1. Title、Decription、Keyword 不能为空。
2. H 标签的使用，值得注意的是，不论任何页面，h1 标签只能出现一次，它是当前页面的主标题，权重最高，对蜘蛛的吸引力是最强的。再往下就是 h2、h3、h4、h5、h6 这些副标题了，所强调的重点也是递减的，当然，它们的出现频率没有明确限制。
3. 关键词在 Meta Description 中的使用，Description：为搜索引擎提供参考，网页的描述信息;搜索引擎采纳后，作为搜索结果中的页面摘要显示，主流搜索引擎对其的建议是不超过 400 字节
4. 图片的关键词优化：HTML 标签中，对于图片 img 标签有帮助的还有 alt 属性，这个属性可以告诉浏览器，当图片无法显示的时候，用 alt 属性中的值来替代。同样这个属性搜索引擎也看得到。
5. Strong 标签的使用，Strong 标签对关键词的强调作用仅次于 H 标签，个人实际布局中会比较在文章内容里出现，用于加粗段落标题或是重点关键词。如果不是重点内容仅仅 UI 突出，请使用 CSS 控制，不使用 Strong。
6. 文字颜色不能和背景一样，不能设置文字大小为 0，少用 visibility 去隐藏关键字，否则当作作弊。
7. URL 命名使用关键，有多个时，使用“-”分隔，如：搜索引擎不能实别“digitalcamerabattery”因为这个单词在搜索引擎切词系统中是不存在的，但可以识别“digital camera battery”或者“digital-camera-battery” 。
8. HTML 优化要富于逻辑，重点明确，层次分明，也是符合 SEO 精神的。

HTML 模板

PC 端：

```
<!DOCTYPE html>
<html lang="cn">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
         <meta http-equiv="Cache-Control" content="no-transform" />
         <meta http-equiv="Cache-Control" content="no-siteapp" />
        <title>PC 端 HTML 模板</title>
        <meta content="" name="Keywords">
        <meta name="description" content="" />
        <link rel="shortcut icon" href="favicon.ico" type="image/x-icon"/>
    </head>
    <body>

    </body>
</html>
```

移动端：

```
<!DOCTYPE html>
<html lang="cn">
    <head>
        <meta charset="UTF-8">
        <title>移动端 HTML 模板</title>
```

```
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <meta content="" name="Keywords">
        <meta name="description" content="" />
        <link rel="shortcut icon" href="favicon.ico" type="image/x-icon"/>
    </head>
    <body>

    </body>
</html>
```

8. CSS 规范

CSS 预处理技术

使用 Less 作为 CSS 预处理技术。

命名规范

1. 命名尽量简洁易懂语义化；
2. 多个单词的使用中划线“-”连接；
3. 公共模块使用.ui-开头，如：.ui-a；
4. 少用 ID，多用 Class；

CSS Hack 规范

我们不建议使用 CSS Hack，尽量少用，能不能就不用，尽量采用其它解决方案。

CSS 性能规范

1. 不建议使用 CSS 表达式，严重影响性能；
2. 避免使用 Filter，严重影响性能；
3. 尽量使用英文，比如：使用 Microsoft YaHei 代替 微软雅黑
4. CSS 层级不宜太深，尽量直接使用 Class；

```
/*Bad*/
.header div ul li p a {
    color: #333;
}
/*Good*/
.class-a {
    color: #333;
```

```
}
```

5. 使用缩写

```
/*Bad*/
.test {
    border-width: 1px;
    border-style: solid;
    border-color: #ddd;
    background-color: #ddd;
    background-image: url(images/xx.jpg);
    background-position: 0 0;
    background-repeat: no-repeat;
    color: #000000;
}
/*Good*/
.test {
    border: 1px solid #ddd;
    background: #ddd url(images/xx.jpg) no-repeat 0 0;
    color: #000;
}
```

6. 避免通配选择器

```
/*避免*/
* {
    font-style: 12px;
}
.class * {
    font-size: 12px;
}
/*少用*/
#header a {
    font-size: 12px;
}
[hidden="true"] {
    font-size: 12px;
}
```

7. 尽可能利用 CSS 的继承机制；
8. 移除无匹配的样式；
9. 禁止使用@import，使用 CSS 预处理技术代替；
10. 背景图使用 CSSSprites，尽量使用 iconfont；

注释规范

1. 每个文件顶简洁说明一下文件的用途相关负责人；
2. 每个模块应该有个简洁的说明；

9. Javascript 规范

变量声明

先使用 var 声明再使用

```
//Bad
hzins = '慧择';
//Good
var hzins = '慧择';
```

命名规范

1. 采用驼峰命名法即——第一个单字以小写字母开始，之后每个单词的首字母大写；如果是构造函数，首字母大写，之后每个单词首字母大写；

```
//变量名，普通函数
var myClass;
function myFunction() {
    //code
}
//构造函数
function MyFuction() {
    //code
}
```

2. 语义化

```
//Bad
var a = 18;
var b = '慧择';
//Good
var age = 18;
var com = '慧择';
```

3. 私有对象前添加下划线“_”

```
//私有函数
function _setName() {
    //code
}
```

4. 禁止使用中文变量名

```
//Bad
```

```
var 慧择 = 'HuiZe';
```

5. 区分不同对象

```
var $id = $('#id'); //jQuery 对象
var img = document.getElementById('img'); //Dom 对象
```

性能规范

1. 缓存常用对象，避免对象过深，如：

```
//缓存常用对象
var body = document.body;
var de = document.documentElement;
//过深
HZ.A.B.C.D.E.domSomething();
```

2. 推荐使用直接量，如：

```
//Bad
var obj = new Object;
var arr = new Array();
var str = new String();

//Good
var obj = {};
var arr = [];
var str = '';
```

3. 禁止使用 with，使用缓存变量替代，如：

```
//Bad
with (document.forms[0]) {
    name.value = "lee king";
    address.value = "Peking";
    zipcode.value = "10000";
}
//Good
var forms = document.forms[0];
forms.name.value = "lee king";
forms.address.value = "Peking";
forms.zipcode.value = "10000";
```

4. setTimeout，setInterval，第一个参数禁止使用字符串形式

```
var time = 1000;
function doSomething() {
    //code
```

```
}
//Bad
setTimeout('doSomething()', time);

//Good
setTimeout(doSomething, time);
//Or
setTimeout(function () {
    //code
}, time);
```

5. 慎用 Eavl、Function ；
6. if 条件：把最有可能的放在最前，反之亦然，如果条件超过三个，可以考虑用 swith case 代替；
7. 减少对 Dom 的操作，推荐使用 createDocumentFragment

```
//Bad
var body = document.body,
    p;
    for (var i = 0, len = 10; i < len; i++) {
        p = document.createElement('p');
        body.appendChild(p);
    }
//Good
var body = document.body,
    frg = document.createDocumentFragment();
    for (var i = 0, len = 10; i < len; i++) {
        p = document.createElement('p');
        frg.appendChild(p);
    }
    body.appendChild(frg);
```

8. 优先使用内置方法；
9. 改变 css 优先使用 class 而不是样式 style；

风格规范

1. 缩进应该是 4 个空格，不要在代码中使用 Tab 字符。在设置开发环境时，将编辑器里的 TAB 快捷键重新设置为 4 个空格；
2. 使用===代替==，用！==代替！=；
3. 多个变量声明

```
//Bad
var hzins = '';
var com = '';
var name = '';

//Good
```

```
var hzins = '',
    com = '',
    name = '';
```

4. 语句必须都有分号结尾，除了 for、function、if、switch、try、while ；
5. 长语句时，连接符号放后面

```
var str = '<ul>' +
    '<li>长语句加号放后面</li>' +
    '<li>长语句加号放后面</li>' +
    '</ul>';
```

6. 所有的循环体和判断体都需要用 "{}" 括起来，函数名与大括号加一个空格

```
//Bad
if (true)
    //code

//Good
if (true) {
    //code
}

/*
括号位置
 */
//Bad
function doSomething()
{
    //Code
}
//Good
function doSomething() {
    //Code
}
/*
函数名跟括号加一个空格
*/
//Bad
function doSomething(){
    //Code
}
//Good
function doSomethin() {
    //Code
}
```

7. 引用对象成员用 obj.property 代替 obj[“property”]，除非属性名是变量或属性名有特殊符号；

8. 去掉多余的逗号

```
//Bad
var obj = {
    name : 'hzins',
    year : 2006,
};
//Good
var obj = {
    name : 'hzins',
    year : 2006
};
```

9. 链式操作

```
//Bad
$('body').delegate('li', 'click', function () {
    //code
}).delegate('div', 'click', function () {
    //code
}).delegate('ul', 'click', function () {
    //code
});
//Good
$('body')
    .delegate('li', 'click', function () {
        //code
    })
    .delegate('div', 'click', function () {
        //code
    })
    .delegate('ul', 'click', function () {
        //code
    });
```

代码检测

所有代码使用工具 jsLint 检测。

注释规范

1. 单选注释方式

```
//单选注释
```

2. 多行注释方式

```
/*
```

```
多行注释
*/
```

3. 每写一个功能函数需要做一些简洁的注释说明

```
/*
例子：
名称：trim
功能：去掉首尾空白字符
说明：接受一个字符串参数
Author：某某
时间：2015-04-14
 */
function trim() {
    //code
}
```

4. 不要滥用注释；

安全规范

1. 避免全局变量污染，使用闭包或命名空间；
2. 慎重使用 Eval 函数，某些情况可以使用 `JSON.stringify` 代替；
3. 避免 XSS 攻击，对有 HTML 输入的需要编码；
4. 敏感信息不存储在 Cookie 里；

10. 前后端合作规范

格式规范

后端返回格式为“JSON 格式”。

接口规范

1. 如果是字符串，就直接返回字符串；
2. 如果是对象，就返回一个 Object 对象：

```
//应该是
typeof obj === "object"; //true
//不是
typeof obj === 'string'; //true
```

3. 如果返回数字，应该是数字类型，而不是字符串类型：

```
typeof obj === 'number'; //true
```

4. 如果是布尔值，应该是 false 或 true，不是 'false' 或 'true'。

11. 相关资料查阅地址

HTML，CSS

1. HTML,CSS 相关

附录：

基础库

公共组件

功能组件