Web前端开发规范手册

**一、规范目的**

1.1 概述 ..................................................................................................................................... 1

**二、文件规范**

2.1 文件命名规则.........................................................................................................................1  
2.2 文件存放位置.........................................................................................................................2

2.3 html 书写规范.......................................................................................................................2  
2.4 css 书写规范..........................................................................................................................7  
2.5 JavaScript 书写规范............................................................................................................12  
2.6 图片规范.................................................................................................................................19  
2.7 注释规范.................................................................................................................................20   
2.8 css 浏览器兼容......................................................................................................................21

**一、规范目的**

1.1 概述

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, 经讨论决定后可以更改此文档.

**二、文件规范**

**2.1文件命名规则**

文件夹和文件名一律全部用小写英文单词，禁止出现简拼、拼音、数字、无意义的命名，英文单词尽量使用一个进行描述，简洁易懂；多个单词用驼峰命名法。

**2.2 文件存放位置**

cn 存放中文 HTML 文件

en 存放英文 HTML 文件

flash 存放 Flash 文件

images 存放图片文件

imagestudio 存放 PSD 源文件

flashstudio 存放 flash 源文件

inc 存放include 文件

library 存放 DW 库文件

media 存放多媒体文件

project 存放工程项目资料

temp 存放客户原始资料

js 存放 JavaScript 脚本

css 存放 CSS 文件

**2.3 html 书写规范**

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

|  |
| --- |
|  <!DOCTYPE html> |

* 文档类型声明统一为HTML5声明类型，编码统一为UTF-8。

|  |
| --- |
| <meta charset="UTF-8"> |

<HEAD>中添加信息。

|  |
| --- |
| <meta name="author" content="smile@kang.cool">//作者  <meta name="description" content="hello">//网页描述  <meta name="keywords" content="a,b,c">//关键字,“，”分隔  <meta http-equiv="expires" content="Wed, 26 Feb 1997 08：21：57 GMT">//设定网页的到期时间。一旦网页过期，必须到服务器上重新调阅  <meta http-equiv="Pragma" content="no-cache">//禁止浏览器从本地机的缓存中调阅页面内容  <meta http-equiv="Window-target" content="\_top">//用来防止别人在框架里调用你的页面  <meta http-equiv="Refresh" content="5;URL=http://kahn1990.com/">//跳转页面，5指时间停留5秒 网页搜索机器人向导。用来告诉搜索机器人哪些页面需要索引，哪些页面不需要索引  <meta name="robots" content="none">//content的参数有all,none,index,noindex,follow,nofollow，默认是all  <link rel="Shortcut Icon" href="favicon.ico">//收藏图标  <meta http-equiv="Cache-Control" content="no-cache, must-revalidate">//网页不会被缓存 |

IE支持通过特定<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edgemode ，从而通知IE采用其所支持的最新的模式。

|  |
| --- |
| <meta http-equiv="X-UA-Compatible" content="IE=Edge"> |

* 非特殊情况下CSS样式文件外链至HEAD之间，JAVASCRIPT文件外链至页面底部。

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <link rel="stylesheet" href="css/main.css">  </head>  <body>  <!-- 逻辑代码 -->  <!-- 逻辑代码底部 -->  <script src="lib/jquery/jquery-2.1.1.min.js"></script>  </body>  </html> |

引入JAVASCRIPT库文件，文件名须包含库名称及版本号及是否为压缩版。

|  |
| --- |
| jQuery-1.8.3.min.js |

引入JAVASCRIPT插件, 文件名格式为库名称+.+插件名称。

|  |
| --- |
| jQuery.cookie.js |

* HTML属性应当按照以下给出的顺序依次排列，来确保代码的易读性。

|  |
| --- |
| Class  id 、 name  data-\*  src、for、 type、 href  title、alt  aria-\*、 role |

避免使用中文拼音尽量简易并要求语义化。

|  |
| --- |
| CLASS --> nHeadTitle --> CLASS遵循小驼峰命名法（little camel-case）  ID --> n\_head\_title --> ID遵循名称+\_  NAME --> N\_Head\_Title --> NAME属性命名遵循首个字母大写+\_  <div class="nHeadTitle" id="n\_head\_title" name="N\_Head\_Title"></div> |

当JAVASCRIPT获取单个元素时，通常使用document.getElementById来获取dom元素，document.getElementById兼容所有浏览器，但IE浏览器会混淆元素的ID和NAME属性，所以要区分ID和NAME命名。

|  |
| --- |
| <input type="text" name="test">  <div id="test"></div>  <button onclick="alert(document.getElementById('test').tagName)"></button>  <!-- ie6下为INPUT --> |

* 特殊符号应使用转意符。

|  |
| --- |
| < --> <  > --> >  空格 --> |

* 含有描述性表单元素（INPUT，TEXTAREA）添加LABEL。

|  |
| --- |
| <p>  <label for="test">测试</label>  <input type="text" id="test" />  </p> |

● 多用无兼容性问题的HTML内置标签,  
比如span、em、strong、optgroup、label等,需要自定义html标签属性时，首先考虑是否存在已有的合适标签可替换，如果没有,可使用须以“data-”为前缀来添加自定义属性，避免使用其他命名方式。

● 语义化HTML。

● 尽可能减少<DIV>嵌套。

● 书写链接地址时避免重定向。

|  |
| --- |
| href="http://www.kahn1990.com/" //即在URL地址后面加“/” |

* HTML中对于属性的定义，确保全部使用双引号，绝不要使用单引号

**2.4 css 书写规范**

● 为了欺骗W3C的验证工具,可将代码分为两个文件，一个是针对所有浏览器,一个只针对IE。即将所有符合W3C的代码写到一个文件中,而一些IE中必须而又不能通过W3C验证的代码（如:  
cursor:hand;）放到另一个文件中，再用下面的方法导入。

|  |
| --- |
| <!-- 放置所有浏览器样式-->  <link rel="stylesheet" type="text/css" href="">  <!-- 只放置IE必须，而不能通过w3c的-->  <!--[if IE]  <link rel="stylesheet" href="">  <![endif]--> |

* CSS样式新建或修改尽量遵循以下原则。

|  |
| --- |
| 根据新建样式的适用范围分为三级：全站级、产品级、页面级。  尽量通过继承和层叠重用已有样式。  不要轻易改动全站级CSS。改动后，要经过全面测试。 |

* CSS属性显示顺序。

|  |
| --- |
| 显示属性  元素位置  元素属性  元素内容属性  CSS书写顺序。  .header {  /\* 显示属性 \*/  display || visibility  list-style  position top || right || bottom || left  z-index  clear  float  /\* 自身属性 \*/  width max-width || min-width  height max-height || min-height  overflow || clip  margin  padding  outline  border  background  /\* 文本属性 \*/  color  font  text-overflow  text-align  text-indent  line-height  white-space  vertical-align  cursor  content  };  兼容多个浏览器时，将标准属性写在底部。  -moz-border-radius: 15px; /\* Firefox \*/  -webkit-border-radius: 15px; /\* Safari和Chrome \*/  border-radius: 15px; /\* Opera 10.5+, 以及使用了IE-CSS3的IE浏览器 \*//标准属性 |

* 使用选择器时，命名比较短的词汇或者缩写的不允许直接定义样式。

|  |
| --- |
| .hd,.bd,.td{};//如这些命名 |

可用上级节点进行限定。

|  |
| --- |
| .recommend-mod .hd |

多选择器规则之间换行，即当样式针对多个选择器时每个选择器占一行。

|  |
| --- |
| button.btn,  input.btn,  input[type="button"] {…}; |

优化CSS选择器。

|  |
| --- |
| #header a { color: #444; };/\*CSS选择器是从右边到左边进行匹配\*/ |

浏览器将检查整个文档中的所有链接和每个链接的父元素，并遍历文档树去查找ID为header的祖先元素，如果找不到header将追溯到文档的根节点，解决方法如下。

|  |
| --- |
| 避免使用通配规则和相邻兄弟选择符、子选择符,、后代选择符、属性选择符等选择器  不要限定id选择符，如div#header（提权的除外）  不要限定类选择器，如ul.recommend（提权的除外）  不要使用 ul li a 这样长的选择符  避免使用标签子选择符，如#header > li > a |

* 使用z-index属性尽量z-index的值不要超过150（通用组的除外），页面中的元素内容的z-index不能超过10（提示框等模块除外但维持在150以下），不允许直接使用（999~9999）之间大值。
* 尽量避免使用CSS Hack。

|  |
| --- |
| property:value; /\* 所有浏览器 \*/  +property:value; /\* IE7 \*/  \_property:value; /\* IE6 \*/  \*property:value; /\* IE6/7 \*/  property:value\9; /\* IE6/7/8/9，即所有IE浏览器 \*/ |
| \* html selector { … }; /\* IE6 \*/  \*:first-child+html selector { … }; /\* IE7 \*/  html>body selector { … }; /\* 非IE6 \*/  @-moz-document url-prefix() { … }; /\* firefox \*/  @media all and (-webkit-min-device-pixel-ratio:0) { … }; /\* saf3+/chrome1+ \*/  @media all and (-webkit-min-device-pixel-ratio:10000),not all and (-webkit-min-device-pixel-ratio:0) { … }; /\* opera \*/  @media screen and (max-device-width: 480px) { … }; /\* iPhone/mobile webkit \*/ |

* 避免使用低效的选择器。

|  |
| --- |
| body > \* {…};  ul > li > a {…};  #footer > h3 {…};  ul#top\_blue\_nav {…};  #searbar span.submit a { … }; /\* 反面示例 \*/ |

* 六个不要三个避免一个使用。

|  |
| --- |
| 不要在标签上直接写样式  不要在CSS中使用expression  不要在CSS中使用@import  不要在CSS中使用!important  不要在CSS中使用“\*”选择符  不要将CSS样式写为单行  避免使用filter  避免使用行内（inline）样式  避免使用“\*”设置{margin: 0; padding: 0;}  使用after或overflow的方式清浮动 |

* 减少使用影响性能的属性。

|  |
| --- |
| position:absolute;  float:left;//如这些定位或浮动属性 |

减少在CSS中使用滤镜表达式和图片repeat,  
尤其在body当中,渲染性能极差, 如果需要用repeat的话,图片的宽或高不能少于8px。

**2.5 javaScript书写规范**

● 命名规范。

|  |
| --- |
| 常量名  全部大写并单词间用下划线分隔  如：CSS\_BTN\_CLOSE、TXT\_LOADING  对象的属性或方法名  小驼峰式（little camel-case）  如：init、bindEvent、updatePosition  示例：Dialog.prototype = {  init: function () {},  bindEvent: function () {},  updatePosition: function () {}  …  };  类名（构造器）  -->小驼峰式但首字母大写  -->如：Current、DefaultConfig  函数名  -->小驼峰式  -->如：current()、defaultConfig()  变量名  -->小驼峰式  -->如：current、defaultConfig  私有变量名  -->小驼峰式但需要用\_开头  -->如：\_current、\_defaultConfig  变量名的前缀  -->续 |

* 代码格式。

|  |
| --- |
| "()"前后需要跟空格  "="前后需要跟空格  ","后面需要跟空格  JSON对象需格式化对象参数  if、while、for、do语句的执行体用"{}"括起来 |

"{}"格式如下。

|  |
| --- |
| if (a==1) {  //代码  }; |

避免额外的逗号。

|  |
| --- |
| var arr = [1,2,3,]; |

for-in循环体中必须用hasOwnProperty方法检查成员是否为自身成员，避免来自原型链上的污染。

长语句可考虑断行。

|  |
| --- |
| TEMPL\_SONGLIST.replace('{TABLE}', da['results'])  .replace('{PREV\_NUM}', prev)  .replace('{NEXT\_NUM}', next)  .replace('{CURRENT\_NUM}', current)  .replace('{TOTAL\_NUM}', da.page\_total); |

为了避免和JSLint的检验机制冲突，“.”或“+”这类操作符放在行尾。

|  |
| --- |
| TEMPL\_SONGLIST.replace('{TABLE}', da['results']).  replace('{PREV\_NUM}', prev).  replace('{NEXT\_NUM}', next).  replace('{CURRENT\_NUM}', current).  replace('{TOTAL\_NUM}', da.page\_total); |

如果模块代码中，使用其它全局变量想跳过JSLint的检查，可以在该文件中加入/\*global\*/声明。

|  |
| --- |
| /\*global alert: true, console: true, top: true, setTimeout: true \*/ |

使用严格的条件判断符。用===代替==，用!==代替!=，避免掉入==造成的陷阱  
在条件判断时，这样的一些值表示false。

|  |
| --- |
| null  undefined与null相等  字符串''  数字0  NaN |

在==时，则会有一些让人难以理解的陷阱。

|  |
| --- |
| (function () {  var undefined;  undefined == null; // true  1 == true; //true  2 == true; // false  0 == false; // true  0 == ''; // true  NaN == NaN;// false  [] == false; // true  [] == ![]; // true  })(); |

对于不同类型的 == 判断，有这样一些规则，顺序自上而下：

|  |
| --- |
| undefined与null相等  一个是number一个是string时，会尝试将string转换为number  尝试将boolean转换为number  0或1  尝试将Object转换成number或string |

而这些取决于另外一个对比量，即值的类型，所以对于0、空字符串的判断，建议使用===  
。===会先判断两边的值类型，类型不匹配时为false。

* 下面类型的对象不建议用new构造。

|  |
| --- |
| new Number  new String  new Boolean  new Object //用{}代替  new Array //用[]代替 |

引用对象成员用obj.prop代替obj["prop"]，除非属性名是变量。

* 从number到string的转换。

|  |
| --- |
| /\*\* 推荐写法\*/  var a = 1;  typeof(a); //"number"  console.log(a); //1  var aa=a+'';  typeof(aa); //"string"  console.log(aa); //'1'  /\*\* 不推荐写法\*/  new String(a)或a.toString() |

从string到number的转换，使用parseInt，必须显式指定第二个参数的进制。

|  |
| --- |
| /\*\* 推荐写法\*/  var a = '1';  var aa = parseInt(a,10);  typeof(a); //"string"  console.log(a); //'1'  typeof(aa); //"number"  console.log(aa); //1 |

从float到integer的转换。

|  |
| --- |
| /\*\* 推荐写法\*/  Math.floor/Math.round/Math.ceil  /\*\* 不推荐写法\*/  parseInt |

字符串拼接应使用数组保存字符串片段，使用时调用join方法。避免使用+或+=的方式拼接较长的字符串，每个字符串都会使用一个小的内存片段，过多的内存片段会影响性能。

|  |
| --- |
| /\*\*推荐的拼接方式array的push、join\*/  var str=[],  list=['测试A','测试B'];  for (var i=0 , len=list.length; i < len; i++) {  str.push( '<div>'+ list[i] + '</div>');  };  console.log(str.join('')); //<div>测试A</div><div>测试B</div>  /\*\* 不推荐的拼接方式+=\*/  var str = '',  list=['测试A','测试B'];  for (var i = 0, len = list.length; i< len; i++) {  str+='<div>' + list[i] + '</div>';  };  console.log(str); //<div>测试A</div><div>测试B</div> |

* 尽量避免使用存在兼容性及消耗资源的方法或属性。

|  |
| --- |
| 不要使用with，void，evil，eval\_r，innerText |

* 注重HTML分离, 减小reflow, 注重性能。

**2.6图片规范**

● 命名应用小写英文、数字、\_组合，便于团队其他成员理解。

|  |
| --- |
| header\_btn.gif  header\_btn2.gif |

* 页面元素类图片均放入img文件夹,  
  测试用图片放于img/testimg文件夹，psd源图放入img/psdimg文件夹。
* 图片格式仅限于gif、png、jpg等。

用png图片做图片时,  
要求图片格式为png-8格式,若png-8实在影响图片质量或其中有半透明效果,  
请为ie-6单独定义背景，并尽量避免使用半透明的png图片。

● 背景图片请尽可能使用sprite技术, 减小http请求。

**2.7注释规范**

● JAVASCRIPT、CSS文件注释需要标明作者、文件版本、创建/修改时间、重大版本修改记录、函数描述、文件版本、创建或者修改时间、功能、作者等信息。

|  |
| --- |
| /\* \* 注释块 \*/ |

中间可添加如下信息

|  |
| --- |
| @file 文件名  @addon 把一个函数标记为另一个函数的扩张，另一个函数的定义不在源文件中  @argument 用大括号中的自变量类型描述一个自变量  @author 函数/类作者的姓名  @base 如果类是继承得来，定义提供的类名称  @class 用来给一个类提供描述，不能用于构造器的文档中  @constructor 描述一个类的构造器  @deprecated 表示函数/类已被忽略  @exception 描述函数/类产生的一个错误  @exec @extends 表示派生出当前类的另一个类  @fileoverview 表示文档块将用于描述当前文件，这个标签应该放在其它任何标签之前  @final 指出函数/类  @ignore 让jsdoc忽视随后的代码  @link 类似于@link标签，用于连接许多其它页面  @member 定义随后的函数为提供的类名称的一个成员  @param 用大括号中的参数类型描述一个参数  @private 表示函数/类为私有，不应包含在生成的文档中  @requires 表示需要另一个函数/类  @return 描述一个函数的返回值  @see 连接到另一个函数/类  @throws 描述函数/类可能产生的错误  @type 指定函数/成员的返回类型  @version 函数/类的版本号 |

**2.8 css浏览器兼容**  **CSS hack**

一、标识区别:

区别 IE6,IE7,IE8,FF。  
1. IE 都能识别\* ; 标准浏览器(如 FF)不能识别\*;

2. IE6能识别\*,但不能识别 !important; IE6在样式前面加\_3. IE7能识别\*,也能识别!important;4.IE8能识别\9例如:background:red \9;  
5. firefox 不能识别\*,但能识别!important;

1.IE6和 firefox 的区别:  
background:orange;\*background:blue;意思就是火狐浏览器的背景颜色是橙色,而 IE 浏览器的背景色是蓝色.

2. IE6和 IE7的区别:  
background:green !important;background:blue;意思指的是:IE7的背景颜色是绿色,IE6的背景颜色是蓝色

3. IE7和 FF 的区别:  
background:orange; \*background:green;意思指的是:火狐浏览器的背景颜色是橙色,而 IE7的背景颜色是绿色

4. FF,IE7,IE6的区别:  
background:orange;  
\*background:green !important;  
\*background:blue;  
意思是火狐浏览器的的背景橙色,IE7浏览器的背景颜色是绿色,而 IE6浏览器的颜色是蓝色.

二、实践建议

(1). 开发平台的选择

在 Firefox 上编写 CSS, 同时兼容其他浏览器的. 这样做肯定会比在 IE 做好再到别的浏览器兼容来得容易, 因为 IE 对老标准支持还是很不错的, 而 IE 的一些特有功能人家却不支持. 所以推荐以Firefox 结合 Firebug 扩展作为平台。

(2). CSS Hack 的顺序

使用 Firefox 作为平台, 只要代码写得够标准, 其实要 Hack 的地方不会很多的, IE 以外的浏览器几乎都不会有问题, 所以可以暂时忽略,

顺序如下:Firefox -> IE6 -> IE7 -> 其他(3). Hack 的方法

说到方法有两种, 一种是在不同文件中处理, 另一种则是在同一个文件中处理. 其实作用是相同的,只是出发点不一样而已.

同一文件中处理.  
如: id="bgcolor"的控件要在 IE6中显示蓝色, IE7中显示绿色, Firefox 等其他浏览器中显示红色。

IE6不认 !important,也不认 \*+html.所以 IE6只能是 blue.  
IE7认 !important,也认 \*+html,优先度: (\*+html + !important) > !important > +html. IE7可以是 red,blue 和 green,但 green 的优先度最高.  
Firefox 和其他浏览器都认 !important. !important 优先, Firefox 可以是 red 和 blue,但 red 优先度高.上述的优先符号均是 CSS3标准允许的,其他浏览器也还有其他的 Hack 方法,但我迄今还没遇到过Firefox 正常, IE 以外的其他浏览器不正常的情况,所以无可分享.只要代码规范,相信这种情况的发生应该是很罕见 (JavaScript 除外).

2. 不同文件中处理.  
为什么同一文件中可以处理还要写在多个文件里面针对不同的浏览器?这是为了欺骗 W3C的验证工具,其实只需要两个文件,一个是针对所有浏览器的,一个只为 IE 服务.将所有符合 W3C 的代码写到一个里面去,而一些 IE 中必须的,又不能通过 W3C 验证的代码 (如: cursor:hand;)放到另一个文件中,再用下面的方法导入.