美食美客

软件规范编码

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [ √ ] 正式发布  [ ] 正在修改 | 文件标识： |  |
| 当前版本： | 1.0 |
| 作 者： | 周伟 |
| 完成日期： | 2018.11-27 |

**目录**

[修订历史记录 2](#_Toc525223149)

[一：JAVA后台规范 4](#_Toc525223150)

[1.规范目的 4](#_Toc525223151)

[1.2 概述 4](#_Toc525223152)

[2.基本准则 4](#_Toc525223153)

[3.文件规范 4](#_Toc525223154)

[3.1 命名规则 4](#_Toc525223155)

[3.2 文件存放位置规范 5](#_Toc525223156)

[3.3 书写规范 5](#_Toc525223157)

[4. 基本原则 5](#_Toc525223158)

[二、 前端规范 6](#_Toc525223159)

[1规范目的 6](#_Toc525223160)

[1.1 概述 6](#_Toc525223161)

[2.基本准则 6](#_Toc525223162)

[3.文件规范 6](#_Toc525223163)

[4语言规范 7](#_Toc525223164)

[4.1 HTML 7](#_Toc525223165)

[4.2 CSS 8](#_Toc525223166)

[4.3 JavaScript 10](#_Toc525223167)

# 一：JAVA后台规范

## 1.规范目的

### 1.2 概述

* 提高团队协作效率
* 便于前端开发以及后期优化维护
* 方便新进的成员快速上手
* 输出高质量的代码

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新此文档。

## 2.基本准则

符合 java 标准, 语义化 java编码, 结构表现行为分离, 兼容性优良。

代码要求简洁明了有序, 尽可能的减小服务器负载, 保证最快的解析速度。

开发时需要遵循如上基本准则，特殊情况可以有所宽限，如一些老项目的页面改造。

## 3.文件规范

### 3.1 命名规则

[使用场景：在新建包、类、声明对象等需要命名的地方时，根据此规则给文件命名并放入指定位置]

统一使用英文来命名，不允许拼音的命名，同时不允许特殊字符出现，甚至符号也不允许（特殊情况下可以，但需要写好注释）

#### 3.1.1包的命名原则

包名采用小写字母开头的驼峰命名法。包的命名以包内类的作用类型为包名，如 controller(接口类)、dao（数据访问）、entity（实体类）、service（服务类）、

serviceImp（服务类实现）、util（工具类）、mapper（数据库实现）等

#### 3.1.2类的命名原则

类的命名采用大写字母开头的的驼峰命名法。 以作用为命名依据，如userController(用户相关接口)、userService、 userServieceImp、userDao、userMapper、StringUtil(字符串处理工具类)

#### 3.1.3. 方法的命名原则

采用小写字母开头的驼峰命名法。以作用为依据进行命名。

例如：getRankByZoneId、checkLoginInfo等

#### 3.1.4 变量命名原则

局部变量一律采用小写字母命名，全局变量采用全大写字母命名，

### 3.2 文件存放位置规范

|  |  |
| --- | --- |
| 文件夹 | 说明 |
| page | 存放 jsp或者html文件 |
| Images | 存放图片文件 |

图片采用”图片文件名-时间戳-用户ID“的方式命名，Excel文件按文件名命名存放即可

### 3.3 书写规范

[使用场景：编码时，根据功能进行模板式的编写以及日志记录，方便后续的修改]

## 4. 基本原则

1.每个接口，对于前端传过来的参数要进行null与空的判断。做出相应的判断

2.每个service实现中要捕获异常，捕获并记录到日志中去。

3.根据service返回的结果，判断各种请求结果（成功，失败等等）

4.注释全部采用/\*\*/的注释

5.方法和异常捕获的大括号的左括号（{）放于方法名后，与方法名同行，左括号独自占一行，如该代码块还有后续（如catch、else等）则在右括号后继续编写后续代码块

6.for、if、switch的大括号采用左括号单独一行 ，右括号单独一行，代码写在中间的书写方式。

12

辩论赛系统后台编写规范9/20/2018

# 前端规范

## 1规范目的

### 1.1 概述

提高团队协作效率

便于前端开发以及后期优化维护

方便新进的成员快速上手

输出高质量的代码

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新此文档。

## 2.基本准则

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. 本文档如有丌对戒者丌合适的地方请及时提出, 经讨论后决定可以更改此文档

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, 尽可能的减小服务器负载, 保证最快的解析速度.

## 3.文件规范

html, css, js, images 文件均归档至<系统开发规范>约定的目录中;

html文件命名: 英文命名, 后缀.htm. 同时将对应界面稿放于同目录中, 若界面稿命名为中文, 请重命名不 html 文件同名, 以方便后端添加功能时查找对应页面;

HTML和CSS文档必须采用UTF-8编码格式。

HTML文档必须使用HTML5的标准文档格式；

css 文件命名: 英文命名, 后缀.css. 共用 base.css, 首页 index.css, 其他页面依实际模块需求命名.;

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

统一使用四个空格作为缩进

## 4语言规范

### 4.1 HTML

标签以及标签属性小写。

可省略的闭合标签不省略，自闭合的标签可不写结束斜线。

嵌套的标签必须被正确的嵌套，嵌套的子元素有一格的缩进。

使用2个空格进行缩进。

有良好的注释规范，针对页面模块有简要的注释，便于维护。

尽量使用语义化的标签，例如header,footer,nav,article,filedset,section等，避免全页使用div布局。

尽量减少标签嵌套。

尽量减少classname，采用语义化标签来代替。

属性值使用双引号，不要使用单引号。

属性的使用顺序 class id name data- src for type href title alt aria role。

布尔型属性 不用赋值，有就是true，没有就是false。

行内元素中不要嵌套块级元素。

段落文字应该用p标签，避免使用div标签。

css、js尽量使用文件引入的形式，不要使用内联。

使用link将css文件引入，并置于head中。使用script将js文件引入，并置于body底部。

### 4.2 CSS

用两个空格来代替制表符（tab） -- 这是唯一能保证在所有环境下获得一致展现的方法。

为选择器分组时，将单独的选择器单独放在一行。

为了代码的易读性，在每个声明块的左花括号前添加一个空格。

声明块的右花括号应当单独成行。

每条声明语句的 : 后应该插入一个空格。

为了获得更准确的错误报告，每条声明都应该独占一行。

所有声明语句都应当以分号结尾。最后一条声明语句后面的分号是可选的，但是，如果省略这个分号，你的代码可能更易出错。

对于以逗号分隔的属性值，每个逗号后面都应该插入一个空格（例如，boxshadow）。

不要在 rgb()、rgba()、hsl()、hsla() 或 rect()值的内部的逗号后面插入空

格。这样利于从多个属性值（既加逗号也加空格）中区分多个颜色值（只加逗号，不加空格）。

对于属性值或颜色参数，省略小于 1 的小数前面的 0 （例如，.5 代替 0.5；

-.5px 代替 -0.5px）。

十六进制值应该全部小写，例如，#fff。在扫描文档时，小写字符易于分辨，因为他们的形式更易于区分。

尽量使用简写形式的十六进制值，例如，用 #fff 代替 #ffffff。

为选择器中的属性添加双引号，例如，input[type="text"]。只有在某些情况下是可选的，但是，为了代码的一致性，建议都加上双引号。

避免为 0 值指定单位，例如，用 margin: 0; 代替 margin: 0px;

HTML和CSS文档必须采用UTF-8编码格式0

CSS编写格式，样式代码保持一行，多个选择器同一个规则必须换行

CSS样式表格分为3个级别：系统级（以lib\_开头）、模块级（以m\_开头）、页面级（以p\_开头）。

修改系统级和模块级的文件必须经过前端主管审批。

模块级的样式必须以“m\_”开头，例如头部模块应该命名为：m\_header。

一个模块一个或多个CSS文件，禁止两个或以上模块使用同一个CSS文件

页面级的样式必须以“p\_”开头、或者外层以p\_开头，例如：p\_goodslist、p\_goodslist .name。

页面级的样式如因特殊原因需要针对这个页面进行对模块级的样式进行重置的时候，须将重置样式写在最顶部。

除系统级文件外，禁止其他页面级和模块级的CSS文件进行系统的样式重置操作，模块级和页面级也禁止直接重置系统级的样式。

### 4.3 JavaScript

二元运算符两侧必须有一个空格，一元运算符与操作对象之间不允许有空格。用作代码块起始的左花括号 { 前必须有一个空格。在对象创建时，属性中的 : 之后必须有空格，: 之前不允许有空格。

每个独立语句结束后必须换行。每行不得超过 120个字符。运算符处换行时，运算符必须在新行的行首。不同行为或逻辑的语句集，使用空行隔开，更易阅读。

不得省略语句结束的分号。在 if / else / for / do / while 语句中，即使只有一行，也不得省略块 {...}。函数定义结束不允许添加分号。 IIFE 必须在函数表达式外添加 (，非 IIFE 不得在函数表达式外添加 (

变量使用Camel命名法常量使用全部字母大写，单词间下划线分隔 的命名方式。函数 使用 Camel命名法。类使用 Pascal命名法。枚举变量使用Pascal命名法，枚举的属性使用全部字母大写，单词间下划线分隔的命名方式。

由多个单词组成的缩写词，在命名中，根据当前命名法和出现的位置，所有字母的大小写与首字母的大小写保持一致。

boolean 类型的变量使用 is 或 has 开头。

单行注释必须独占一行。// 后跟一个空格，缩进与下一行被注释说明的代码一致。多行注释避免使用 /\*...\*/ 这样的多行注释。有多行注释内容时，使用多个单行注释。

类型定义都是以{开始, 以}结束。

使用包装方式扩展类成员时， 必须通过 @lends 进行重新指向。

类的属性或方法等成员信息使用 @public / @protected / @private 中的任意一个，指明可访问性。

在 Equality Expression 中使用类型严格的 ===。仅当判断 null 或 undefined 时，允许使用 == null。

尽可能使用简洁的表达式。

按执行频率排列分支的顺序。

如果函数或全局中的 else 块后没有任何语句，可以删除 else。不要在循环体中包含函数表达式，事先将函数提取到循环体外。对循环内多次使用的不变值，在循环外用变量缓存。

对有序集合进行遍历时，缓存 length。虽然现代浏览器都对数组长度进行了缓存，但对于一些宿主对象和老旧浏览器的数组对象，在每次 length 访问时会动态计算元素个数，此时缓存 length 能有效提高程序性能。

对有序集合进行顺序无关的遍历时，使用逆序遍历。逆序遍历可以节省变量，代码比较优化。

类型检测优先使用 typeof。对象类型检测使用 instanceof。null 或 undefined 的检测使用 == null。

转换成 string 时，使用 + ''。string 转换成 number，要转换的字符串结尾包含非数字并期望忽略时，使用 parseInt，使用 parseInt 时，必须指定进制。 转换成 boolean 时，使用 !!。

number 去除小数点，使用 Math.floor / Math.round / Math.ceil，不使用 parseInt。

使用 + 拼接字符串，如果拼接的全部是 StringLiteral，压缩工具可以对其进行自动合并的优化。所以，静态字符串建议使用 + 拼接。在现代浏览器下，使用 + 拼接字符串，性能较数组的方式要高。

复杂的数据到视图字符串的转换过程，选用一种模板引擎。

对象创建时，如果一个对象的所有 属性 均可以不添加引号，则所有 属性 不得

添加引号。对象创建时，如果任何一个 属性 需要添加引号，则所有 属性 必须添加 '。

属性访问时，尽量使用 .。

for in 遍历对象时, 使用 hasOwnProperty 过滤掉原型中的属性。遍历数组不使用 for in。

使用数组字面量 [] 创建新数组，除非想要创建的是指定长度的数组。

不因为性能的原因自己实现数组排序功能，尽量使用数组的 sort 方法。

一个函数的长度控制在 50 行以内。一个函数的参数控制在 6 个以内。

通过 options 参数传递非数据输入型参数。

在适当的时候将闭包内大对象置为 null。

空函数不使用 new Function() 的形式。

对于性能有高要求的场合，建议存在一个空函数的常量，供多处使用共享。

类的继承方案，实现时需要修正 constructor。通常使用其他 library 的类继承方案都会进行 constructor 修正。如果是自己实现的类继承方案，需要进行 constructor 修正。

声明类时，保证 constructor 的正确性。 属性在构造函数中声明，方法在原型中声明。

自定义事件的 事件名 必须全小写。

避免使用直接 eval 函数。

使用 new Function 执行动态代码。

减少 delete 的使用。处理 delete 可能产生的异常。

避免修改外部传入的对象。JavaScript 因其脚本语言的动态特性，当一个对象未被 seal 或 freeze 时，可以任意添加、删除、修改属性值。

但是随意地对 非自身控制的对象 进行修改，很容易造成代码在不可预知的情况下出现问题。因此，设计良好的组件、函数应该避免对外部传入的对象的修改。

对于单个元素，尽可能使用 document.getElementById 获取，避免使用document.all。

对于多个元素的集合，尽可能使用 context.getElementsByTagName 获取。其中 context 可以为 document 或其他元素。指定 tagName 参数为 \* 可以获得所有子元素。

尽量减少 DOM 操作。操作 DOM 时，尽量减少页面 reflow。

43使用 addEventListener 时第三个参数使用 false。

13