通用规范

**写在最前：以下规范必须严格按照规范来编码**

下面是通用规范的正文:

* 所有代码规范和开发规范等说明在特殊情况下可以特殊处理,视具体情况而定
* 文件名和文件夹名用中划线分词
* 变量设置初始值的时候,要根据它的使用用途来初始化为不同的值,比如用于存放计数的变量,它的初始值应该是0而不是'',也不能是false
* 约定变量可以保持原有的约定变量名,不需要另外详细命名(前提是当该作用域没有相同作用的约定变量时,比如只有1个for循环)

约定变量就是大部分程序员都惯用的一种变量名,比如for循环里的i j k变量,捕捉异常里的e变量等等;然而PHP的foreach里的$k(就是 $key)

* 全部缩进都要尽量使用TAB符而不是空格(一是字节大小问题,二是大部分编辑器都可以设置4个空格字长显示tab符,一般代码编辑/阅读场景下不存在显示不统一的问题)
* 等号,双目运算符和三目运算符的两边都要加空格

a=1; //错  
a = 1; //对  
      
a = 1+2; //错  
a = 1 + 2; //对  
      
a = b?1:2; //错  
a = b ? 1 : 2; //对

运算符加空格隔开运算参数可以使运算表达式阅读起来舒服些,也比较工整,这不仅仅是我个人感受,你看人家开源代码也这样

* 一行只写一句代码(但PHP模板文件尽量做到,太短时可以挤在一行,模板里的逻辑块都要有花括号包住)

php模板里允许这样将整个if写在一行:

<img<?php if($a == 1){ echo ' class="xxx"'; }?> src="..." />

顺便推荐这样写,开头阅读可能不习惯,但要理解这个意思并上手也不困难:

<img<?php $a == 1 && print(' class="xxx"'); ?> src="..." />

* if,else,for,while,function里面的代码块要基于外层代码缩进一个tab,并且都要用花括号包住里面的代码(尽管只有1行)
* switch里面的case和default关键字也要缩进

例子:

switch(value){  
    case 1:  
        XXXXX  
  
    case 2:  
        YYYYY  
  
    default:  
        ZZZZZ  
}

* 每一块不同作用的工作代码之间要空开一行(这个不好界定,自行把握)
* 空函数/方法可以全部写在一行
* 用于保存int,string,bool,array以外的变量类型初始化赋默认值时,要用null作为默认值
* 函数与方法里面的第一行不要多余地拍多一个空行
* 变量,函数名,对象方法名都用小写驼峰命名
* 字符串都用单引号
* 当变量在if,for,while等控制结构后面要进行使用,而控制结构内又要对变量做计算的时候,该变量必须在控制结构外面先进行声明并赋予默认值

if(a == b){  
    c = 1;  
}  
使用(c);    //错误! 这里a不一定等于b,所以c变量不一定会存在

if(a == b){  
    c = 1;  
}else{  
    c = 2;  
}  
使用(c);    //错误!虽然有个else,但是项目开发过程中根据需求的变化,else可能会被撤销,到时候$c就会得不到赋值,一般程序员也不会完全检查这些,就会导致bug发生,所以也不行,必须在if前先声明变量c

数组 = get数据();  
for(var i in 数组){  
    a += 数组[i];  
}  
alert(a);    //错误!当数组为空数组的时候进不去for循环,所以a变量不会产生

while,switch这些控制结构也是类似,请注意

# PHP规范

* 变量,全局函数,以及类的公共方法都用小写驼峰分词

//错的  
$user\_name = '小红';  
$UserName = '小红';  
$Add\_Time = 111;  
      
//对的  
$userName = '小红';  
$addTime = 111;  
      
      
//错的  
function GetName(){}  
public function GetXXX(){}  
      
//对的  
function getName(){}  
public function getUserInfo(){}

* 类的私有属性,方法都在命名前加下划线

//错的  
private $name;  
private $parseName(){}  
protected $parseUrl(){}  
      
//对的  
private $\_name;  
private $\_parseName(){}  
protected $\_parseUrl(){}

* 在模板里计算产生的变量和函数也要加下划线开头命名,以防误代码量大时阅读起来以为是注册进来的变量
* 常量全大写下划线分词

define('ADMIN\_EMAIL', 'xxx@umelook.com')  
class User{  
    const MAX\_AGE = 200;   //类常量也是全大写下划线分词  
}

* 数组下标小写下划线分词

这里注意的是,由客户端发起GET/POST请求提交的参数命名用小写陀峰命名,所以$\_POST和$\_GET两个全局数组的下标是陀峰的

$aData = [  
    'user\_name' => 'xxx',  
    'add\_time' => 123,  
];  
      
$\_POST = [  
    'userName' => 'xxx',  
    'addTime' => 123,  
];

* 类名用大写开头的陀峰

class User{}    //一个词  
class CookieCollection{}    //二个词

* 类的文件名前缀和类名一致

User类就是User.php

Login类就是Login.php

* url上的get参数前面的片段全部要小写中划线分词,而GET参数的命名则是小写陀峰命名

http://xxx.com/path/userInfo.html?user\_id=1&pageSize=10 错误

http://xxx.com/path/user-info.html?userId=1&pageSize=10 正确

使用原则

* 表示 是/否 逻辑的变量用true/false

//错误的:  
$isBoy = 1;  
if($user['sex'] == User::SEX\_GIRL){  
    $isBoy = 0;  
}  
  
//正确的:  
$isBoy = true;  
if($aUser['sex'] == User::SEX\_GIRL){  
    $isBoy = false;  
}

* 接收前端表示是/否的0/1时要转换成true/false

$isSave = Yii::$app->request->post('isSave') ? true : false;  
//或者  
$isSave = (bool)Yii::$app->request->post('isSave');

* 非特殊情况下不允许使用全局变量$GLOBALS
* 用于保存数组内容的变量初始化时如果要设置默认值则用 [ ]
* 禁止将变量放在双引号的字符串里面,像这样"姓名:$sName"是不允许的
* 如果字符串里面带有单引号则可以变通一下,使用'What\'s your name'或者"What's your name"都行

备注

基本上使用phpdoc备注格式书写备注,如果在Yii2框架中,则增加以下备注规范:

1.说明一个常量是事件名称

当一个类有const EVENT\_BEFORE\_LOGIN = 'before\_login'的常量时,如果这个常量用在这样的代码中:

$class->on(MyClass::EVENT\_BEFORE\_LOGIN, function(){...});

2.说明这个常量就是一个事件标识,则这个常量需要基于phpdoc规范来扩展描述它的作用,配合定义的位置这样写备注:

/\*\*  
 \* @event 这里是事件的介绍,比如 登陆前的事件  
 \*/  
const EVENT\_BEFORE\_LOGIN = 'before\_login';

3.说明明一个函数/方法的作者时,@author可以不输入邮箱,直接写姓名标记即可

/\*\*  
 \* 退出登陆  
 \* @author 小明  
 \*/  
public function logout(){  
      
}

4.说明一个方法有没有编写单元测试时,增加@test来标注,值是单元测试用例的类名(带命名空间)和方法

/\*\*  
 \* 退出登陆  
 \* @author 小明  
 \* @test \tests\UserTestSuite::testLogout  
 \*/  
public function logout(){  
      
}

# JS规范

### 命名

1.变量与函数都用小写驼峰命名

2.jQuery选择器返回的fn对象变量要在前面加$

wrap = $('#wrap'); //错误的  
  
$wrap = $('#wrap'); //正确的

3.用于ajax传参的Object的属性名称用小写陀峰命名

$.ajax({  
    url : 'xx.php',  
    type : 'post',  
    data : {  
        isSave : 0,  
        lastId : 9  
    }  
});

4.插件和对象要用大写开头的陀峰

function Editor(){  
    this.method1 = function(){}  
    this.method2 = function(){}  
}  
Editor.prototype.xxx = function(){};  
  
$.fn.AjaxUpload = function(){}  
  
window.Editor = {  
    attr1 : 3,  
    method1 : function(){},  
    method2 : function(){}  
}

5.对象的方法要用小写陀峰

6.闭包内的变量要用下划线开头,但闭包里的函数里的变量就不用

7.由于闭包内的函数只能在闭包里面使用,相当于私有方法,所以要用下划线开头再小写陀峰

(function(){  
    //这个函数相当于私有方法,所以下划线开头  
    function \_showUserIcon(){  
        var user = getUser(); //这个user变量在闭包的函数里,不需要下划线开头  
        if(\_config.xxx){}  
    }  
          
          
    var \_config = {}; //这个变量在闭包里面算是类算是私有属性,加下划线开头  
})();

### 使用原则

1.用于保存数组内容的变量初始化时要用 [ ]

2.用于保存其它类型的变量初始化时要用 null,未知变量类型的也用 null

3.表示是否的变量要用逻辑值true/false

//错误的:  
var isBoy = 1;  
if(sex == 2){  
    isBoy = 0;  
}  
  
//正确的  
var isBoy = true;  
if(sex == 2){  
    isBoy = false;  
}

4.ajax向后端传递 是/否 的意思时要用 0/1

5.闭包插件在内部调用自己时不能写自己的名字,要先用self变量存起自身然后使用self.xxx来调用属性或方法

(function(container){  
    container.Plugin1 = {  
        attr1 : 1,  
        method1 : function(){}  
    };  
          
    var self = container.Plugin1;  
})(window);

函数对象也是如此

6.在jQuery的回调里,比如click,hover或each传入的function回调参数中,this代指所选择的原生DOM对象,可以用$this = $(this)来存起当前对象

$this表示当前回调里的DOM对象,如果同一区域内对有3次或以上使用相同选择器或将原生dom对象转成fn对象的时候,一定要先用变量存起来,因为多次jQuery转换对象消耗大

7.闭包插件在某个DOM(比如DIV)区域内构造UI后,代码里查找自己的DOM的时候非必须的话禁止直接$('.J-xxx')这样对整个页面文档进行全局查找,要基于该闭包所定义的容器DOM开始查找,就是先用变量或属性存起插件的外框,然后通过这个$外框变量.find 来查找以尽量避免找到其它模块的相同选择器DOM

8.除了全局变量声明，所有代码都要在window.onload事件触发后才能运行,并且这些代码都要放到js代码的底部运行,只一个模板里存在多个script标签时就放在最后一个script标签中,如果是独立的JS文件就放在文件内容最后

9.参数表逗号后面要有空格

//错误的:  
function xxx(param1,param2,param3){}  
xxx(1,2,3);  
  
//正确的:  
function xxx(param1, param2, param3){}  
xxx(1, 2, 3)

10.给Object初始内容时最后一个元素后面不需要加,号

//错误的:  
var config = {  
    option1 : 1,  
    option2 : 2,    //这里作为最后一个元素不要追加逗号结尾!!!  
};  
  
//正确的:  
var config = {  
    option1 : 1,  
    option2 : 2  
};

11.基于jQuery开发的插件文件用 jquery.插件名称.js 来命名,比如 jqeury.data\_builder.js

12.用jQuery获取DOM对象的时候要用 #id 或者.J-xxx命名的class选择器

13.插件生成的HTML标签不能带ID

# HTML+CSS规范

1.所有标签名称都用小写

2.属性名称小写

3.属性值统一使用双引号包住,不使用单引号

4. id、class、itemid和name属性值命名要陀峰

5.script标签里的第一层代码建议不要缩进

6.html的块级元素之间嵌套要换行,联级元素或者块级套联级的话,看代码长短情况,最终是要实现阅读方便排版干净

7.id和J-xxx开头的class用于提供给js获取DOM,CSS样式不能使用这两种选择器

8.jQuery要批量删除或移动之类的批量操作用J-XXX的class做选择器

9.假如有class="xx"的时候,这个class只能用来做CSS样式选择器的,然而class="J-xxx"则这个不能做样式选择器,是用于提供 $('.J-xxx') 这样的jQuery选择器用的,详情请见开发引导的程序演示代码

10.非不得已的情况不能在html内写style属性嵌样式不能用id、itemid、data-xxx做样式选择器

11.尽量把样式写在css文件中,除非这个页面的样式实在没有重用性并且很少加载才能在模板文件中写style标签

12.每个选择器及其样式都要写在同一行

13.data-xxx的属性在xxx部分有多个词语时必须用下划线分词,比如data-user\_name