2016

4.0

XiunoBBS 开发手册

比基于框架开发更快速

www.xiuno.com

目　　录

xn.empty()
xn.ceil()
xn.round()
xn.floor()
xn.strtolower()
xn.strtoupper()
xn.json_encode()
xn.json_decode()
xn.min()
xn.max()
xn.str_replace()
xn.strpos()
xn.strrpos()
xn.substr()
xn.explode()
xn.implode()
xn.array_merge()
xn.array_diff()
xn.array_keys()
xn.array_values()
xn.in_array()
xn.rand()
xn.template()
xn.is_mobile()
xn.is_email()
xn.is_string()
xn.is_function()

xn.is_array()
xn.is_number()
xn.is_regexp()
xn.is_object()
xn.is_element()
xn.lang()
xn.url()
xn.image_resize()
xn.upload_file()
$.location()
$.pdata()
$.cookie()
$.xget()
$.xpost()
$.require()
$.require_css()
$.each_sync()
$.fn.removeDeep()
$.fn.emptyDeep()
$.fn.checked()
$.fn.button()
$.fn.location()
$.fn.alert()
$.fn.serializeObject()
$.fn.reset()
$.fn.base_href()
$.fn.base64_encode_file()

Xiuno BBS 入门

Xiuno BBS 是什么?

Xiuno BBS 4.0 是 2016 年诞生的，国产、小巧、精悍的 Web 产品，后端基于 PHP + MySQL，前端基于 Bootstrap 4.0 + JQuery 3.1，是一套通用的轻论坛系统。

主程序架构采用函数风格的 MVC，插件机制采用 AOP 机制，大大的简化了程序的复杂度，在同等复杂度的功能实现上比同类产品的代码简洁很多，核心只有 15 个表，非常利于二次开发。

【 社区】

http://bbs.xiuno.com/

【 git】

http://git.oschina.net/xiuno/xiunobbs/

Xiuno BBS 是什么?

Xiuno BBS 是什么?

Xiuno BBS 4.0 是 2016 年诞生的，国产、小巧、精悍的 Web 产品，后端基于 PHP + MySQL，前端基于 Bootstrap 4.0 + JQuery 3.1，是一套通用的轻论坛系统。

主程序架构采用函数风格的 MVC，插件机制采用 AOP 机制，大大的简化了程序的复杂度，在同等复杂度的功能实现上比同类产品的代码简洁很多，核心只有 15 个表，非常利于二次开发。

如何获取?

如何获取?

下载地址：http://bbs.xiuno.com/down/xiunobbs_4.0.beta_005.tar.gz

GIT：

git clone https://git.oschina.net/xiuno/xiunobbs.git

包含 Bootstrap 4 构建工具，在 view/bootstrap 目录，正式部署可以删除。

如何安装?

##如何安装 Xiuno BBS 4.0 ?

1.确认您的主机支持 PHP，并且已经开通并且配置好了 MySQL。

2.设置如下目录和文件为可写(Linux: 目录权限为 0777，Windows 设置用户 everyone 可读写)

./upload

./tmp

./log

./conf

3.上传所有文件到你的网站根目录

4.访问 http://www.domain.com/install/, 根据提示安装。

5.删除 install 目录

URL-Rewrite 网址美化

URL-Rewrite 网址美化

只需要一条规则：

将 *.htm* 转发到 index.php?*.htm* 即可。

具体需要以下 2 步开启 URL-Rewrite

```
1. 编辑 conf/conf.php 'url_rewrite_on'=>1,
2. 清空 tmp 目录
```

转发规则

Nginx：

打开 nginx 配置文件 /usr/local/nginx/conf/nginx.conf 找到对应的虚拟主机配置处，追加加粗行:

```
location / {
          rewrite "^(.*)/(.+?).htm$" $1/index.php?$2.
htm last;
          if (! -e $request_filename) {
                  rewrite  ^(.*)$  /index.php?s=$1  l
ast;
        }
        index    index.html index.htm index.php;
        root     /data/wwwroot/xiuno.com;
}
```

然后重新启动 nginx: service nginx restart

Apache:

vim /etc/httpd/conf/httpd.conf

```
<Directory d:/xiuno.com>
    Options FollowSymLinks ExecCGI Indexes
    AllowOverride all
    Order deny,allow
    Allow from all
    Satisfy all
</Directory>
NameVirtualHost *:80
```

Apache .htaccess

如果Appache 支持 .htaccess，那么可以编辑 .htaccess 文件放置于根目录下:

```
<IfModule mod_rewrite.c>
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^admin/(.*)\.htm(.*)$ /admin/index.php?$1.htm$2 [L]
RewriteRule ^(.*)\.htm(.*)$ /index.php?$1.htm$2 [L]
</IfModule>
```

Apache httpd.conf

如果将规则直接放入 httpd.conf 则需要在前面加 /，看来 Apache 也反人类：

```
<IfModule mod_rewrite.c>
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^/admin/(.*)\.htm(.*)$ /admin/index.php?$1.htm$2 [L]
RewriteRule ^/(.*)\.htm(.*)$ /index.php?$1.htm$2 [L]
</IfModule>
```

更多细节参考：http://bbs.xiuno.com/thread-2.htm

性能优化

性能优化

Xiuno BBS 支持通过 Cache 加速，默认不开启。

推荐的技术栈：Linux / Nginx / PHP7/OPCache/Yac MySQL 5.6+

开启方法：

1. 编译安装、配置好环境
2. 编辑 conf/conf.php，修改

```
'cache' =>
  array (
    'enable' => true,
    'type' => 'yac',
```

编译配置方法参考：http://bbs.xiuno.com/thread-12701.htm

前端技术栈

Bootstrap 4

JQuery 3

Tether.js

Fontawesome

xiuno.js

Bootstrap 4

Bootstrap 4.0

Bootstrap 是 Twitter 前端团队开发的一套公共的开源 UI 库，用来解决各种设备屏幕大小不一致的问题。采用了响应式布局，自动适应平板、手机、PC等各类设备。

Bootstrap 3 是全球最流行的前端 UI 库，在全球有大量的支持者和良好的生态。也许它的写法不是最佳，但是有这广泛的群众基础，是前端 UI 交流的普通话。

Bootstrap 4 是最近才出来的新版本，完全放弃了 IE8，未来可能会完全取代 Bootstrap 3，成为新的王者。

Xiuno BBS 4 前瞻性的采用了 Bootstrap 4，从 alpha2 一直跟进到 alpha4，在 我们欣喜的发现 alpha4 开始支持 JQuery 3.1，也就意味着全面抛弃 IE8，拥抱标准浏览器。

效果：

```
<form class="form-inline">
  <div class="form-group">
    <label class="sr-only" for="exampleInputAmount">
Amount (in dollars)</label>
    <div class="input-group">
      <div class="input-group-addon">$</div>
      <input type="text" class="form-control" id="ex
ampleInputAmount" placeholder="Amount">
      <div class="input-group-addon">.00</div>
    </div>
  </div>
  <button type="submit" class="btn btn-primary">Tran
sfer cash</button>
</form>
```

Bootstrap 4 官方：

http://v4-alpha.getbootstrap.com/

中文资料：

http://wiki.jikexueyuan.com/project/bootstrap4/

JQuery 3

JQuery 3.1

JQuery 是前端最流行 JS 库，它的口号：write less, do more，确实做到了。DOM 操作的方便性简洁几乎到了极致。虽然在复杂的前端应用里 DOM 操作显得有点力不从心，开始逐渐需要 MVVM 等模式的支持，但是对于大多数应用来说 JQuery 是一把锋利好用的瑞士军刀。

JQuery 3.1 全面抛弃 IE8，彻底拥抱标准浏览器，代码精简的同时性能也得到了提升。Xiuno BBS 4.0 从开始就不再考虑旧版 IE，所以毅然采用了 JQuery 3.1。

效果：

```
$( "button.continue" ).html( "Next Step..." )
```

JQuery 官方：

http://jquery.com/

中文相关资料：

http://jquery.cuishifeng.cn/

http://www.w3school.com.cn/jquery/jquery_reference.asp

Tether.js

Tether.js

Tether.js 是 Bootstrap 4 里面用来绝对定位的一个小巧的 JS 库。

它封装了各种复杂情况下的定位功能。

效果：

在 div scroll 滚动的过程中，greenBox 对象始终在 yellowBox 的右侧。

这通过普通的 position absolute relative 定位是很难实现的，或者会导致 DOM 嵌套变得复杂。

官方网站：

http://tether.io/

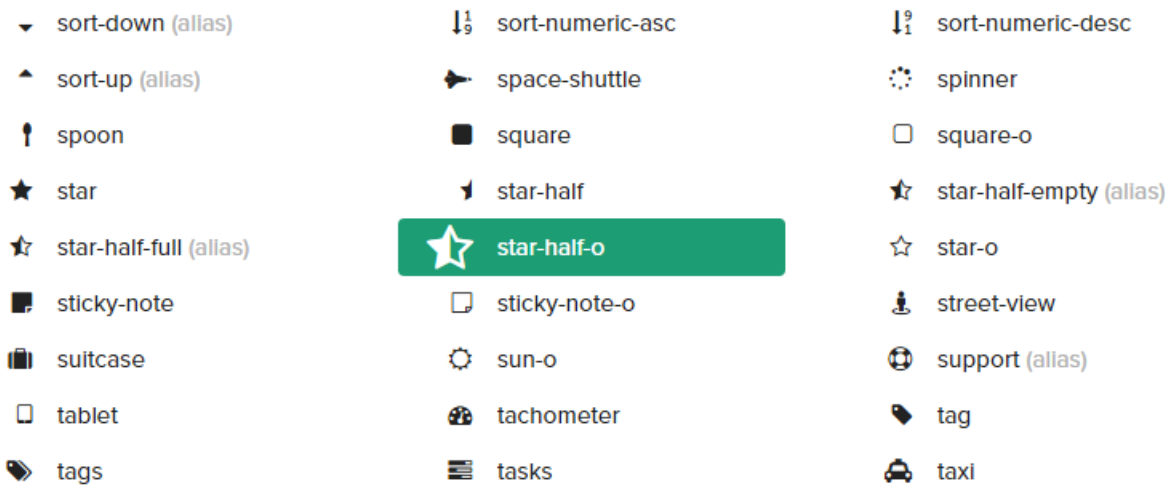
Fontawesome

Fontawesome

Fontawesome 是一套开源的字体图标，含有大量的常用图标，而且一直在更新。可惜 Bootstrap 4 不再默认集成它，Xiuno BBS 4.0 对它进行了集成。并且让它支持简写：

```
<i class="icon-qq"></i>
```

部分图标预览



官方网站

http://fontawesome.io/icons/

xiuno.js

xn.min()

xn.max()

xn.str_replace()

xn.strpos()

xn.strrpos()

xn.substr()

xn.explode()

xn.implode()

xn.array_merge()

xn.array_diff()

xn.array_keys()

xn.array_values()

xn.in_array()

xn.rand()

xn.template()

xn.is_mobile()

xn.is_email()

xn.is_string()

xn.is_function()

xn.is_array()

xn.is_number()

xn.is_regexp()

xn.is_object()

xn.is_element()

xn.lang()

xn.url()

xn.image_resize()

xn.upload_file()

$.location()

$.pdata()

$.cookie()

$.xget()

$.xpost()

$.require()

$.require_css()

$.each_sync()

$.fn.removeDeep()

$.fn.emptyDeep()

$.fn.checked()

$.fn.button()

$.fn.location()

$.fn.alert()

$.fn.serializeObject()

$.fn.reset()

$.fn.base_href()

$.fn.base64_encode_file()

$.alert()

$.confirm()

$.ajax_modal()

xiuno.js 是什么?

Xiuno.js 是什么?

Xiuno.js 是作者在开发 Xiuno BBS 当中的衍生物，因为 JS 与 PHP 有大量函数风格命名和参数不一致，导致一些记忆错乱，所以就用 JS 实现了一些常见的 PHP 函数的功能。

所以，它不是一个框架！ 它是一个库。

效果：

```
var s = xn.substr("abcdefg", 0, -3);

console.log(s);

// 结果：  abcd
```

Object.keys()

Object.keys()

```
Object.keys(obj)
```

【 功能】

返回一个数组，包含对象的的所有的 Keys。

【 参数】

```
obj：对象
```

【 用例】

```
var obj = {"username": "Jack", "email": "jack@gmail.
com"};

var arr = Object.keys(obj);

console.log(arr);

// 结果：["username", "email"]

?>
```

Object.length()

Object.length()

```
Object.length(obj)
```

【 功能】

返回对象包含的元素的个数。。

【 参数】

```
obj：对象
```

【 用例】

```
var obj = {"username": "Jack", "email": "jack@gmail.
com"};

var n = Object.length(obj);

console.log(n);

// 结果：2

?>
```

Object.count()

Object.count()

```
Object.count(obj)
```

【 功能】

返回对象包含的元素的个数，不包含继承而来的 Key。

【 参数】

```
obj：对象
```

【 用例】

```
var obj = {"username": "Jack", "email": "jack@gmail.
com"};

var n = Object.count(obj);

console.log(n);

// 结果：2

?>
```

xn.htmlspecialchars()

xn.htmlspecialchars()

```
xn.htmlspecialchars(s)
```

【 功能】

对字符串进行 HTML 转义。

PHP htmlspecialchars() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var s = "<b>Hello</b>";

var s2= xn.htmlspecialchars(s);

console.log(s2)

// 结果: &lt;b&gt;Hello&lt;/b&gt;

?>
```

xn.urlencode()

xn.urlencode()

```
xn.urlencode(s)
```

【 功能】

对字符串进行安全的 url encode 转义，JS 内置的字符编码为 UNICODE，会被编码为 UTF-8。

编码后的字符串仅仅包含 字母、数字、下划线，可以被安全的通过 URL 传递。

XiunoPHP xn_urlencode() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var s = "中国";

var s2= xn.urlencode(s);

console.log(s2)

// 结果: _E4_B8_AD_E5_9B_BD

?>
```

xn.urldecode()

xn.urldecode()

```
xn.urldecode(s)
```

【 功能】

对 xn.urlencode() 编码过的字符串进行解码。

XiunoPHP xn_urldecode() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var s = "_E4_B8_AD_E5_9B_BD";

var s2= xn.urldecode(s);

console.log(s2)

// 结果: 中国

?>
```

xn.nl2br()

xn.nl2br(s)

```
xn.nl2br(s)
```

【 功能】

将换行 \r\n 替换为

PHP nl2br() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var s = "Hello \r\n World";

var s2= xn.nl2br(s);

console.log(s2)

// 结果: Hello <br> World

?>
```

xn.time()

xn.time()

```
xn.time()
```

【 功能】

获取当前的 UNIX 时间戳

PHP time() 的 JS 版本。

【 用例】

```
var t = xn.time();

console.log(t)

// 结果: 1474781352

?>
```

xn.intval()

xn.intval()

```
xn.intval(s)
```

【 功能】

将字符串转为 INT 类型，不会出现 NaN 类型，格式不良好返回 0。

PHP intval() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var n = xn.intval("123个");

console.log(n)

// 结果: 123

?>
```

xn.floatval()

xn.floatval()

```
xn.floatval(s)
```

【 功能】

将字符串转为 float 类型，不会出现 NaN 类型，格式不良好返回 0。

PHP floatval() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var n = xn.intval("123.45 个");

console.log(n)

// 结果: 123.45

?>
```

xn.isset()

xn.isset(obj)

```
xn.isset(obj)
```

【 功能】

判断一个变量是否被定义过

PHP isset() 的 JS 版本。

【 参数】

```
obj：对象
```

【 用例】

```
var r = xn.isset(var1);

console.log(r)

// 结果：false

?>
```

xn.empty()

xn.empty(obj)

```
xn.empty(obj)
```

【 功能】

判断一个变量是否为空，以下值都会被当做空，而返回 true

0, '', '0', [], {}, null, undefined, unknown

PHP empty() 的 JS 版本。

【 参数】

```
obj：对象
```

【 用例】

```
var r = xn.empty(window.xxx);

console.log(r)

// 结果：false

?>
```

xn.ceil()

xn.ceil()

```
xn.ceil(n)
```

【 功能】

对一个数取整，向上圆整。

PHP ceil() 的 JS 版本。

【 参数】

```
n：数值对象
```

【 用例】

```
var r = xn.ceil('0.2');

console.log(r)

// 结果：1

?>
```

xn.round()

xn.round()

```
xn.round(n)
```

【 功能】

对一个数取整，四舍五入。

PHP round() 的 JS 版本。

【 参数】

```
n：数值对象
```

【 用例】

```
var r = xn.round('0.6');

console.log(r)

// 结果：1

?>
```

xn.floor()

xn.floor()

```
xn.floor(n)
```

【 功能】

对一个数取整，四舍五入。

PHP floor() 的 JS 版本。

【 参数】

```
n：数值对象
```

【 用例】

```
var r = xn.floor('0.2');

console.log(r)

// 结果：0

?>
```

xn.strtolower()

xn.strtolower()

```
xn.strtolower(s)
```

【 功能】

对字符串转为小写

PHP strtolower() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var s = xn.strtolower('Abcd');

console.log(s)

// 结果: abcd

?>
```

xn.strtoupper()

xn.strtoupper()

```
xn.strtoupper(s)
```

【 功能】

对字符串转为小写

PHP strtoupper() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var s = xn.strtoupper('Abcd');

console.log(s)

// 结果: ABCD

?>
```

xn.json_encode()

xn.json_encode()

```
xn.json_encode(obj)
```

【 功能】

对 obj 进行 json 编码，返回编码后的字符串。

PHP json_encode() 的 JS 版本。

【 参数】

```
obj：对象
```

【 用例】

```
var obj = {username: "Jack", email: "jack@gmail.com"
};
var s = xn.json_encode(obj);

console.log(s)

// 结果："{\"username\":\"Jack\",\"email\":\"jack@gma
il.com\"};"

?>
```

xn.json_decode()

xn.json_decode()

```
xn.json_decode(s)
```

【 功能】

对 s 进行 json 解码，返回解码后的对象。

PHP json_encode() 的 JS 版本。

【 参数】

```
s: 字符串 对象
```

【 用例】

```
var s ="{\"username\":\"Jack\",\"email\":\"jack@gmai
l.com\"};"
var obj = xn.json_decode(s);

console.log(obj)

// 结果: {username: "Jack", email: "jack@gmail.com"}

?>
```

xn.min()

xn.min()

```
xn.min(...)
```

【 功能】

返回参数中最小的值。

PHP min() 的 JS 版本。

【 参数】

```
...: 不定个数的变参
```

【 用例】

```
var n = xn.min(3, 1, 2, 4, 5);

console.log(n)

// 结果: 1

?>
```

xn.max()

xn.max()

```
xn.max(...)
```

【 功能】

返回参数中最大的值。

PHP max() 的 JS 版本。

【 参数】

```
...: 不定个数的变参
```

【 用例】

```
var n = xn.max(3, 1, 2, 4, 5);

console.log(n)

// 结果: 5

?>
```

xn.str_replace()

xn.str_replace()

```
xn.str_replace(s, d, str)
```

【 功能】

将 str 中的 s 字符串替换为 d 字符串。

PHP str_replace() 的 JS 版本。

【 用例】

```
var s = xn.str_replace('a', 'b', 'abc');

console.log(s)

// 结果: "bbc"

?>
```

xn.strpos()

xn.strpos()

```
xn.strpos(str, s)
```

【 功能】

返回字符串 str 中 s 字符第一次出现的位置。

PHP strrpos() 的 JS 版本。

【 参数】

```
str：在该字符串 查找
s：待查找的字符串
```

【 用例】

```
var n = xn.strpos("123.jpg.php", "jpg");

console.log(n)

// 结果：3

?>
```

xn.strrpos()

xn.strrpos()

```
xn.strrpos(str, s)
```

【 功能】

返回字符串 str 中 s 字符最后出现的位置。

PHP strrpos() 的 JS 版本。

【 参数】

```
str：在该字符串 查找
s：待查找的字符串
```

【 用例】

```
var n = xn.strrpos("123.jpg.jpg.php", "jpg");

console.log(n)

// 结果：8

?>
```

xn.substr()

xn.substr()

```
xn.substr(str, start, len)
```

【 功能】

对字符串 str 进行截取，从 start 指定位置开始，截取 len 长度，支持负数。

PHP substr() 的 JS 版本。

【 参数】

```
str：对该字符串 进行截取
start：开始位置
len：截取的长度，负数表示从后往前的偏移量
```

【 用例】

```
var s = xn.substr("123456", 0, -1);

console.log(s)

// 结果：123456

?>
```

xn.explode()

xn.explode()

```
xn.explode(sep, s)
```

【 功能】

用字符串 sep 对字符串 s 进行进行分割，返回一个数组。

PHP explode() 的 JS 版本。

【 参数】

```
sep：分隔符
s：对该字符串进行分割
```

【 用例】

```
var arr = xn.explode(".", "abc.jpg");

console.log(arr)

// 结果：["abc", "jpg"]

?>
```

xn.implode()

xn.implode()

```
xn.implode(glur, arr)
```

【 功能】

用数组 arr 通过 glur 进行合并，返回合并后的对字符串。

PHP implode() 的 JS 版本。

【 参数】

```
glur：分隔符
arr：对该数组进行合并
```

【 用例】

```
var s = xn.implode(".", ["abc", "jpg"]);

console.log(s)

// 结果："abc.jpg"

?>
```

xn.array_merge()

xn.array_merge()

```
xn.array_merge(arr1, arr2)
```

【 功能】

将数组 arr1 和 arr2 进行合并，返回合并后的数组。

PHP array_merge() 的 JS 版本。

【 参数】

```
arr1: 数组 1
arr2: 数组 2
```

【 用例】

```
var arr = xn.array_merge(["a", "b"], ["c", "d"]);

console.log(arr)

// 结果: ["a", "b", "c", "d"]

?>
```

xn.array_diff()

xn.array_diff()

```
xn.array_diff(arr1, arr2)
```

【 功能】

返回存在于 $arr1，但不存在于 $arr2 的元素集合的数组。

PHP array_diff() 的 JS 版本。

【 参数】

```
arr1: 数组 1
arr2: 数组 2
```

【 用例】

```
var arr = xn.array_diff(["a", "b"], ["b", "c", "d"])
;

console.log(arr)

// 结果: ["a"]

?>
```

xn.array_keys()

xn.array_keys()

```
xn.array_keys(obj)
```

【 功能】

返回一个数组，包含对象中所有的 key 值

PHP array_keys() 的 JS 版本。

【 参数】

```
obj：对象
```

【 用例】

```
var obj = {"username": "Jack", "email": "jack@gmail.
com"};
var arr = xn.array_keys(obj);

console.log(arr)

// 结果：["username", "email"]

?>
```

xn.array_values()

xn.array_values()

```
xn.array_values(obj)
```

【 功能】

返回一个数组，包含对象中所有的 value 的值

PHP array_values() 的 JS 版本。

【 参数】

```
obj：对象
```

【 用例】

```
var obj = {"username": "Jack", "email": "jack@gmail.
com"};
var arr = xn.array_values(obj);

console.log(arr)

// 结果：["Jack", "jack@gmail.com"]

?>
```

xn.in_array()

xn.in_array()

```
xn.in_array(v, arr)
```

【 功能】

判断 v 是否在 arr 当中。

PHP in_array() 的 JS 版本。

【 参数】

```
v: 查找的对象
arr: 从这个数组进行查找
```

【 用例】

```
var r = xn.in_array(3, [3, 2, 1, 4, 5]);

console.log(r)

// 结果: true

?>
```

xn.rand()

xn.rand()

```
xn.rand(n)
```

【 功能】

生成一个位数为 n 的随机字符串。

XiunoPHP xn_rand() 的 JS 版本。

【 参数】

```
n: 随机字符串 位数
```

【 用例】

```
var s = xn.rand(6);

console.log(s)

// 结果: "kd3i9w"

?>
```

xn.template()

xn.template()

```
xn.template(s, json)
```

【 功能】

对字符串进行替换，按照 json 指定的 key value。

【 参数】

```
s：对此字符串 进行替换，一般是 HTML 模板
json：对象
```

【 用例】

```
var s = xn.template("Hi, {username}", {"username": "Jack"});

console.log(s)

// 结果："Hi, Jack"

?>
```

xn.is_mobile()

xn.is_mobile()

```
xn.is_mobile(s)
```

【 功能】

判断字符串是否为手机号码格式

【 参数】

```
s: 包含手机号码的字符串
```

【 用例】

```
var r = xn.is_mobile("18812345678");

console.log(r)

// 结果: true

?>
```

xn.is_email()

xn.is_email()

```
xn.is_email(s)
```

【 功能】

判断字符串是否为 Email 格式

【 参数】

```
s: 包含 Email 的字符串
```

【 用例】

```
var r = xn.is_email("abc@gmail.com");

console.log(r)

// 结果: true

?>
```

xn.is_string()

xn.is_string()

```
xn.is_string(obj)
```

【 功能】

判断是否为字符串对象

【 参数】

```
obj：对象
```

【 用例】

```
var r = xn.is_string("abc");

console.log(r)

// 结果：true

?>
```

xn.is_function()

xn.is_function()

```
xn.is_function(obj)
```

【 功能】

判断是否为函数对象

【 参数】

```
obj：对象
```

【 用例】

```
var a = function() {alert(123); };
var r = xn.is_function(a);

console.log(r)

// 结果：true

?>
```

xn.is_array()

xn.is_array()

```
xn.is_array(obj)
```

【 功能】

判断是否为函数对象

【 参数】

```
obj：对象
```

【 用例】

```
var a = [1,2,3];
var r = xn.is_function(a);

console.log(r)

// 结果：true

?>
```

xn.is_number()

xn.is_number()

```
xn.is_number(obj)
```

【 功能】

判断是否为数值对象

【 参数】

```
obj：对象
```

【 用例】

```
var a = 123;
var r = xn.is_regexp(a);

console.log(r)

// 结果：true

?>
```

xn.is_regexp()

xn.is_regexp()

```
xn.is_regexp(obj)
```

【 功能】

判断是否为正则表达式对象

【 参数】

```
obj：对象
```

【 用例】

```
var a = /^\w+/;
var r = xn.is_regexp(a);

console.log(r)

// 结果：true

?>
```

xn.is_object()

xn.is_object()

```
xn.is_object(obj)
```

【 功能】

判断是否为对象

【 参数】

```
obj：对象
```

【 用例】

```
var a = {};
var r = xn.is_object(a);

console.log(r)

// 结果：true

?>
```

xn.is_element()

xn.is_element()

```
xn.is_element(obj)
```

【 功能】

判断是否为 HTML 元素。

nodeType == 1

【 参数】

```
obj：对象
```

【 用例】

```
var a = document.body;
var r = xn.is_element(a);

console.log(r)

// 结果：true

?>
```

xn.lang()

xn.lang()

```
xn.lang(key, arr)
```

【 功能】

语言包功能函数

【 参数】

```
key：查找语言包的 key
arr: 替换的变量
```

【 用例】

```
var lang = {
    "login_successfully": "{username}, 登陆成功！"
}
var s = xn.lang("login_successfully", {username: "Jack"});

console.log(s)

// 结果：Jack, 登陆成功！

?>
```

xn.url()

xn.url()

```
xn.url(u, url_rewrite)
```

【 功能】

生成 URL， 格式与 XiunoPHP 保持一致。

【 参数】

```
u: URL
url_rewrite: 格式
```

【 用例】

```
var s = xn.url("user-login");

console.log(s)

// 结果: "?user-login.htm"

?>
```

xn.image_resize()

xn.image_resize()

```
xn.image_resize(file_base64_data, callback, options)
```

【 功能】

对图片进行缩放

【 参数】

```
file_base64_data: 图片的 base64 编码数据。
callback: 处理完以后的回调函数:
    function(code, message) {
        // code = 0
        // message = {width: width, height: height,
 data: s}
    }
options: 选项:
    {
        width: 1200,
        height: 2400,
        action: "thumb", // clip
        filetype: "jpg", // 如果不指定，则与原来的 bas
e64 中指定的格式保持一致
        qulity: 0.7,    // 图片质量。
    }
```

【 用例】

```
var imgdata = "data:image/gif xxxx";

var callback = function(code, message) {

    if(code != 0) return alert(message)

    console.log(message);

    // 结果:  {width: width, height: height, data: s}
}

var options = {width: 1200};
xn.image_resize(file_base64_data, callback, options)
;
```

xn.upload_file()

xn.upload_file()

```
xn.upload_file(file, upload_url, postdata, complete_
callback, progress_callback, thumb_callback)
```

【 功能】

通过 POST 方式上传 base64 编码过的文件（可以设置对图片进行缩略和裁切）

【 参数】

```
file: 文件对象
upload_url: 服务器 URL
postdata: POST 数据，格式：username=Jack&email=jack@g
mail.com
complete_callback:  完成后的回调函数:
    function(code, message) {
        // code 为服务端返回的 json 数据
        // message 为服务端返回的 json 数据
    }
progress_callback: 进度回调函数:
    function(percent) {
         // percent 为数值：0 - 100
    }
thumb_callback: 可选：缩略图回调函数:
    function(base64_data) {
        // 此处可以用来显示缩略图，一般不需要。
    }
```

【 用例】

```
    var file = e.target.files[0]; // 文件控件 onchange 后触发的 event;
    var upload_url = 'xxx.php'; // 服务端地址
    var postdata = {width: 2048, height: 4096, action: 'thumb', filetype: 'jpg'};
    var progress = function(percent) { console.log('progress:'+ percent); }}; // 如果是图片，会根据此项设定进行缩略和剪切 thumb|clip
    xn.upload_file(file, upload_url, postdata, function(code, json) {
        // 成功
        if(code == 0) {
            console.log(json.url);
            console.log(json.width);
            console.log(json.height);
        } else {
            alert(json);
        }
    }, progress);
```

$.location()

$.location()

```
$.location(href)
```

【 功能】

等价于 window.location = href;

【 参数】

```
href：跳转的 URL
```

【 用例】

```
$.location("user-login.htm");
```

$.pdata()

$.pdata()

```
$.pdata(key, value)
```

【 功能】

存储数据到浏览器端的 sessionStorage 对象当中，可以保存比 Cookie 大很多的数据，并且不会被发送到服务端。

【 参数】

```
key：键名
value：键值
```

【 用例】

```
$.pdata('key1', 'value1');
console.log($.pdata('key1'));

// 结果："value1"
```

$.cookie()

$.cookie()

```
$.cookie(name, value, time, path)
```

【 功能】

设置或者读取客户端 Cookie，会随浏览器发送到服务器。

【 参数】

```
name：cookie 名
value：cookie 值
time：过期时间，单位为秒。
path：路径
```

【 用例】

```
$.cookie('key1', 'value1');

console.log($.cookie('key1'));

// 结果："value1"
```

$.xget()

$.xget()

```
$.xget(url, callback, retry)
```

【 功能】

AJAX 请求服务端。

与 $.get() 不同在于，当服务器返回非 json 数据的时候，$.xget() 能返回错误回调。

【 参数】

```
url: AJAX 请求的 URL
callback: 回调函数
retry: 重试次数
```

【 用例】

```
$.xget('user-login.htm', function(code, message) {
    if(code == 0) {
        alert('成功');
    } else {
        alert('错误: '+message);
    }
})
```

$.xpost()

$.xpost()

```
$.xpost(url, postdata, callback, progress_callback)
```

【 功能】

AJAX POST 请求服务端。

与 $.post() 不同在于，当服务器返回非 json 数据的时候，$.xpost() 能返回错误回调，并且能指定进度回调函数。

【 参数】

```
url: AJAX 请求的 URL
callback: 回调函数
progress_callback: 进度回调函数，参数为一个数值：0-100
```

【 用例】

```
$.xpost('user-login.htm', "username=Jack", function(code, message) {
    if(code == 0) {
        alert('成功');
    } else {
        alert('错误: '+message);
    }
})
```

$.require()

$.require()

```
$.require(... callback)
```

【 功能】

异步加载 js, 加载成功以后 callback

【 参数】

```
...:  js 文件的URL
callback: 最后一个参数为回调函数
```

【 用例】

```
$.require('1.js', '2.js', function() {
    alert('after all loaded');
});

$.require(['1.js', '2.js', function() {
    alert('after all loaded');
}]);
```

$.require_css()

$.require_css()

```
$.require_css(filename)
```

【 功能】

异步加载 css

【 参数】

```
filename:  css 文件的URL
```

【 用例】

```
$.require('1.css');
```

$.each_sync()

$.each_sync()

```
$.each_sync(array, func, callback)
```

【 功能】

串行化异步动作。

对 async 进行了封装，避免并发导致的乱序。

【 参数】

```
array:  数组
func: 函数
callback: 回调函数
```

【 用例】

```
$.each_sync(items, function(i, callback) {
        var item = items[i];
        $.post(url, function() {
            // ...
            callback();
        });
    });
```

$.fn.removeDeep()

$.fn.removeDeep()

```
$.fn.removeDeep()
```

【 功能】

清理节点，和节点上的事件。

remove() 并不清除子节点事件! ！ 用来替代 remove()，避免内存泄露

【 用例】

```
$('div.card').removeDeep();
```

$.fn.emptyDeep()

$.fn.emptyDeep()

```
$.fn.emptyDeep()
```

【 功能】

清理节点，和节点上的事件，与 removeDeep() 不同的是它保留当前节点。

【 用例】

```
$('body').emptyDeep();
```

$.fn.checked()

$.fn.checked()

```
$.fn.checked()
```

【 功能】

获取选中的控件的值。

主要针对 checkbox radio select 控件。

【 返回值】

可能是数组，也可能是字符串，取决于控件类型。

【 用例】

```
var v = $('select').checked();
// 返回字符串： 123

var arr = $('input[type="checkbox"]').checked();
// 返回数组: [1, 2, 3]
```

$.fn.button()

$.fn.button()

```
$.fn.button(v)
```

【 功能】

获取选中的控件的值。

主要针对 checkbox radio select 控件。

【 参数】

v: button 上的文字和状态

【 用例】

```
// 正在加载，会调用 loading-text 属性
$('button').button('loading');

// 禁用状态
$('button').button('disabled');

// 启用
$('button').button('enable');

// 重设状态
$('button').button('reset');

// 使用非状态值得文字内容
```

```
$('button').button('非状态值的文字内容');
```

$.fn.location()

$.fn.location()

```
$.fn.location(href)
```

【 功能】

页面跳转，支持连续操作。

【 参数】

href: 跳转的 URL

【 用例】

```
jsubmit.button(message).delay(1000).button('reset').
delay(1000).location('http://xxxx');
```

$.fn.alert()

$.fn.alert()

```
$.fn.alert(message)
```

【 功能】

在控件的上方显示提示信息。

【 参数】

message: 信息内容

【 用例】

```
$('input.subject').alert("请输入标题");
```

$.fn.serializeObject()

$.fn.serializeObject()

```
$.fn.serializeObject()
```

【 功能】

对 Form 表单的控件序列化，生成对象。

$.fn.serialize() 生成的是字符串。

【 用例】

```
var postdata = $('form').serializeObject();
consolelog(postdata);

// 结果: {"username": "Jack", "email": "jack@gmail.com"}
```

$.fn.reset()

$.fn.reset()

```
$.fn.reset()
```

【 功能】

对 Form 表单状态重设。

会对内 所以后控件进行重设，并且清楚 alert() 等残余信息。

【 用例】

```
$('form').reset();
```

$.fn.base_href()

$.fn.base_href()

```
$.fn.base_href(base)
```

【 功能】

用 JS 实现 相对路径的功能。

一般用来处理公共模板的路径不正确问题。

【 参数】

`base：相对路径值`

【 用例】

```
$('#threadlist').base_href('../');
```

$.fn.base64_encode_file()

$.fn.base64_encode_file()

【 功能】

将文件的内容 base64 编码放入隐藏的同名控件。方便文件上传

【 参数】

```
标签属性传参：
<input type="file" multiple="multiple" class="form-control" name="file1" value="" data-assoc="img1" placeholder="选择文件" />

data-assoc: 表示图片缩略图的 ID（如果非图片则不显示）
```

【 原理】

在文件选择后，生成一个隐藏的控件 hidden，名字与文件控件相同。

内容为文件的 base64 编码。

这样会随着表单一起 POST 发送到服务端。

【 用例】

```
var jform = $("#form");
var jsubmit = $("#submit");

jform.base64_encode_file(); // 对文件进行 base64 编码，处理文件上传，很方便
```

```
jform.on('submit', function(){
    jform.reset();
    var postdata = jform.serialize();
    jsubmit.button('loading');
    $.xpost(jform.attr('action'), postdata, function
(code, message) {
        if(code == 0) {
            $.alert(message);
            jsubmit.text(message).delay(3000).locati
on();
            return;
        } else {
            alert(message);
            jsubmit.button('reset');
        }
    });
    return false;
});
```

服务端获取：

```
<?php

// ...

$data = param_base64('file1');

file_put_contents('1.jpg', $data);
```

```
?>
```

$.alert()

$.alert()

```
$.alert(s, timeout, options)
```

【 功能】

弹出提示对话框

【 参数】

```
s: 对话框中的字符
timeout: 超时后关闭( 秒)
options: 其他参数
    .size: sm|md|lg    对话框的大小
```

【 用例】

```
$.alert('成功！');
$.alert('成功！', 3, {size: "md"});
```

$.confirm()

$.confirm()

```
$.confirm(s, ok_callback, options)
```

【 功能】

弹出确认对话框

【 参数】

```
s: 对话框中的标题
ok_callback: 点击确认后的回调函数
options: 其他参数
    options.size: sm|md|lg  对话框的大小
    options.body: string  对话框中的文本内容
```

【 用例】

```
$.confirm('确定删除吗? ', function() {
    alert("确定! ");
});
```

$.ajax_modal()

$.ajax_modal()

```
$.ajax_modal(url, title, size, callback, arg)
```

【 功能】

AJAX 请求服务端，获取的内容弹出对话框。

【 参数】

```
url：请求的服务端 URL
title：对话框标题
size：对话框大小
callback：回调函数
arg：其他参数
```

【 特别说明】

在下面的用例中，将展示这个过程，并没有直接调用 $.ajax_modal() 函数，而是绑定到了标签的属性当中。

【 原理】：

根据标签中的属性定义的参数，data-modal-title="" data-modal-url="" 等来调用 $.ajax_modal()

鼠标点击标签定义的元素后，产生了一个 AJAX 请求，服务端会返回标准的 HTML 文档，其中元素包含 class="ajax_modal_body" 的 innerHTML 会被“放入”对话框中。

并且里面的 JS 会被执行，而对话框相关的参数 args.jmodal,

args.callback, args.arg 都会传递过来。这样可以方便的控制对话框的关闭。并且还可以再回调原页面的函数（属性 data-modal-callback="" 定义）

【 用例】

```
    ------------------------------------------------
--------------
    index.htm
    ------------------------------------------------
--------------

     范例 1:
    <button id="button1" data-modal-url="user-login.
htm" data-modal-title="用户登录" data-modal-arg="xxx"
data-modal-callback="login_success_callback" data-mo
dal-size="md"></button>

    范例 2:
    <a id="button1" href="user-login.htm" data-modal
-title="用户登录" data-modal-arg="xxx" data-modal-cal
lback="login_success_callback" data-modal-size="md">
link</a>

    范例 3:
    <a href="user-login.htm" data-modal-title="用户登
录" data-modal-size="md">link</a>

    <script>
    // 如果需要指定回调( 可选)
```

```
    function login_success_callback(code, message) {
        alert(message);
    }
    </script>
```

```
    ------------------------------------------------
---------------
    route/user.php
    ------------------------------------------------
---------------

    if($action == 'login') {
        if($method == 'GET') {
            include './view/user_login.htm';
        } else {
            $email = param('email');
            $password = param('password');
            // ...
            message(0, '登陆成功');
        }
    }
```

```
    ------------------------------------------------
---------------
    view/user_login.htm
    ------------------------------------------------
---------------

    <?php include './view/header.inc.htm';?>
```

```
    <div class="card">
        <div class="card-header">登陆</div>
        <div class="card-body ajax_modal_body">
            <form action="user-login.htm" method="po
st" id="login_form">
                <div class="form-group input-group">
                    <div class="input-group-prepend"
>
                        <span class="input-group-tex
t"><i class="icon-user"></i></span>
                    </div>
                    <input type="text" class="form-c
ontrol" placeholder="Email" name="email">
                    <div class="invalid-feedback"></
div>
                </div>
                <div class="form-group input-group">
                    <div class="input-group-prepend"
>
                        <span class="input-group-tex
t"><i class="icon-lock"></i></span>
                    </div>
                    <input type="password" class="fo
rm-control" placeholder="密码" name="password">
                    <div class="invalid-feedback"></
div>
                </div>
                <div class="form-group">
                    <button type="submit" class="btn
btn-primary btn-block" data-loading-text="正在提
```

```
交...">登陆</button>
                    </div>
                </form>
            </div>
        </div>
        <?php include './view/footer.inc.htm';?>
        <script>

        // 模态对话框的脚本将会在父窗口，被闭包起来执行。

        // 接受传参
        var args = args || {jmodal: null, callback: null
, arg: null};
        var jmodal = args.jmodal;  // 对应当前模态对话框
        var callback = args.callback;  // 对应 data-callb
ack=""
        var arg = args.arg; // 对应 data-arg=""

        var jform = $('#login_form');
        var jsubmit = jform.find('input[type="submit"]')
;
        var jemail = jform.find('input[name="email"]');
        var jpassword = jform.find('input[name="passwor
d"]');
        jform.on('submit', function() {
            jform.reset();
            jsubmit.button('loading');
            var postdata = jform.serializeObject();
            $.xpost(jform.attr('action'), postdata, func
tion(code, message) {
```

```
            if(code == 0) {
                jsubmit.button(message);

                // 关闭当前对话框
                if(jmodal) jmodal.modal('dispose');
                // 回调父窗口
                if(callback) callback(message);

            } else if(code == 'email') {
                jemail.alert(message).focus();
                jsubmit.button('reset');
            } else if(code == 'password') {
                jpassword.alert(message).focus();
                jsubmit.button('reset');
            } else {
                alert(message);
                jsubmit.button('reset');
            }
        });
        return false;
    });
    </script>
    ```
```
```

程序结构

目录结构

Xiuno BBS 4.0 目录结构

```
admin/                              -- 后台管理目录
    route                           -- 后台路由
    view                            -- 后台模板
    index.php                       -- 后台入口
    index.inc.php                   -- 入口包含的代码段
    admin.func.php                  -- 后台依赖的函数
    menu.conf.php                   -- 后台配置文件
conf/                               -- 全站配置文件
    conf.php                        -- 配置文件
    conf.default.php                -- 默认配置文件
    attach.conf.php                 -- 附件配置文件
    smtp.conf.php                   -- 发送邮件的配置文件
install/                            -- 安装目录
lang/                               -- 语言包
log/                                -- 日志目录，按照天存放日志
model/                              -- 数据处理的函数文件目录
plugin/                             -- 插件目录，一个插件一个目录
route/                              -- 路由目录，业务逻辑处理
tmp/                                -- 临时文件存放目录，插件和代码合并后的文件存放于此
upload/                             -- 上传目录
view/                               -- 前端模板目录
robots.txt                          -- 屏蔽蜘蛛的配置文件
.htaccess                           -- Apache URL-Rewrite
```

```
文件
index.inc.php                    --  前端入口 包含代码段
model.inc.php                  -- Model 包含目 录
index.php                        -- 前台 程序入口
```

表结构

Xiuno BBS 4.0 表结构

```
### 用户表 ###
DROP TABLE IF EXISTS `bbs_user`;
CREATE TABLE `bbs_user` (
  uid int(11) unsigned NOT NULL AUTO_INCREMENT COMME
NT '用户编号',
  gid smallint(6) unsigned NOT NULL DEFAULT '0' COMM
ENT '用户组编号',    # 如果要屏蔽，调整用户组即可
  email char(40) NOT NULL DEFAULT '' COMMENT '邮箱',
  username char(32) NOT NULL DEFAULT '' COMMENT '用户
名',    # 不可以重复
  realname char(16) NOT NULL DEFAULT '' COMMENT '用户
名',    # 真实姓名，天朝预留
  idnumber char(19) NOT NULL DEFAULT '' COMMENT '用户
名',    # 真实身份证号码，天朝预留
  `password` char(32) NOT NULL DEFAULT '' COMMENT
'密码',
  `password_sms` char(16) NOT NULL DEFAULT '' COMMEN
T '密码',    # 预留，手机发送的 sms 验证码
  salt char(16) NOT NULL DEFAULT '' COMMENT '密码混
杂',
  mobile char(11) NOT NULL DEFAULT '' COMMENT '手机
号',        # 预留，供二次开发扩展
  qq char(15) NOT NULL DEFAULT '' COMMENT 'QQ',
# 预留，供二次开发扩展，可以弹出QQ直接聊天
  threads int(11) NOT NULL DEFAULT '0' COMMENT '发帖
数',        #
```

```
  posts int(11) NOT NULL DEFAULT '0' COMMENT '回帖数'
,       #
  credits int(11) NOT NULL DEFAULT '0' COMMENT '积分'
,       # 预留，供二次开发扩展
  golds int(11) NOT NULL DEFAULT '0' COMMENT '金币',
# 预留，虚拟币
  rmbs int(11) NOT NULL DEFAULT '0' COMMENT '人民币',
# 预留，人民币
  create_ip int(11) unsigned NOT NULL DEFAULT '0' CO
MMENT '创建时IP',
  create_date int(11) unsigned NOT NULL DEFAULT '0'
 COMMENT '创建时间',
  login_ip int(11) unsigned NOT NULL DEFAULT '0' COM
MENT '登录时IP',
  login_date int(11) unsigned NOT NULL DEFAULT '0' C
OMMENT '登录时间',
  logins int(11) unsigned NOT NULL DEFAULT '0' COMME
NT '登录次数',
  avatar int(11) unsigned NOT NULL DEFAULT '0' COMME
NT '用户最后更新图像时间',
  PRIMARY KEY (uid),
  UNIQUE KEY username (username),
  UNIQUE KEY email (email),                    # 升
级的时候可能为空
  KEY gid (gid)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8;
INSERT INTO `bbs_user` SET uid=1, gid=1, email='admi
n@admin.com', username='admin',`password`='d98bb50e8
08918dd45a8d92feafc4fa3',salt='123456';
```

```
# 用户组
DROP TABLE IF EXISTS `bbs_group`;
CREATE TABLE `bbs_group` (
  gid smallint(6) unsigned NOT NULL,            #
  name char(20) NOT NULL default '',            # 用
户组名称
  creditsfrom int(11) NOT NULL default '0',     # 积
分从
  creditsto int(11) NOT NULL default '0',       # 积
分到
  allowread int(11) NOT NULL default '0',       # 允
许访问
  allowthread int(11) NOT NULL default '0',     # 允
许发主题
  allowpost int(11) NOT NULL default '0',       # 允
许回帖
  allowattach int(11) NOT NULL default '0',     # 允
许上传文件
  allowdown int(11) NOT NULL default '0',       # 允
许下载文件
  allowtop int(11) NOT NULL default '0',        # 允
许置顶
  allowupdate int(11) NOT NULL default '0',     # 允
许编辑
  allowdelete int(11) NOT NULL default '0',     # 允
许删除
  allowmove int(11) NOT NULL default '0',       # 允
许移动
  allowbanuser int(11) NOT NULL default '0',
# 允许禁止用户
```

```
  allowdeleteuser int(11) NOT NULL default '0',
# 允许删除用户
  allowviewip int(11) unsigned NOT NULL default '0',
# 允许查看用户敏感信息
  PRIMARY KEY (gid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_ge
neral_ci;
INSERT INTO `bbs_group` SET gid='0', name="游客组", c
reditsfrom='0', creditsto='0', allowread='1', allowt
hread='0', allowpost='1', allowattach='0', allowdown
='1', allowtop='0', allowupdate='0', allowdelete='0'
, allowmove='0', allowbanuser='0', allowdeleteuser=
'0', allowviewip='0';

INSERT INTO `bbs_group` SET gid='1', name="管理员组",
creditsfrom='0', creditsto='0', allowread='1', allow
thread='1', allowpost='1', allowattach='1', allowdow
n='1', allowtop='1', allowupdate='1', allowdelete=
'1', allowmove='1', allowbanuser='1', allowdeleteuse
r='1', allowviewip='1';
INSERT INTO `bbs_group` SET gid='2', name="超级版主
组", creditsfrom='0', creditsto='0', allowread='1',
 allowthread='1', allowpost='1', allowattach='1', al
lowdown='1', allowtop='1', allowupdate='1', allowdel
ete='1', allowmove='1', allowbanuser='1', allowdelet
euser='1', allowviewip='1';
INSERT INTO `bbs_group` SET gid='4', name="版主组", c
reditsfrom='0', creditsto='0', allowread='1', allowt
hread='1', allowpost='1', allowattach='1', allowdown
='1', allowtop='1', allowupdate='1', allowdelete='1'
```

```
, allowmove='1', allowbanuser='1', allowdeleteuser=
'0', allowviewip='1';
INSERT INTO `bbs_group` SET gid='5', name="实习版主
组", creditsfrom='0', creditsto='0', allowread='1',
 allowthread='1', allowpost='1', allowattach='1', al
lowdown='1', allowtop='1', allowupdate='1', allowdel
ete='0', allowmove='1', allowbanuser='0', allowdelet
euser='0', allowviewip='0';

INSERT INTO `bbs_group` SET gid='6', name="待验证用户
组", creditsfrom='0', creditsto='0', allowread='1',
 allowthread='0', allowpost='1', allowattach='0', al
lowdown='1', allowtop='0', allowupdate='0', allowdel
ete='0', allowmove='0', allowbanuser='0', allowdelet
euser='0', allowviewip='0';
INSERT INTO `bbs_group` SET gid='7', name="禁止用户
组", creditsfrom='0', creditsto='0', allowread='0',
 allowthread='0', allowpost='0', allowattach='0', al
lowdown='0', allowtop='0', allowupdate='0', allowdel
ete='0', allowmove='0', allowbanuser='0', allowdelet
euser='0', allowviewip='0';

INSERT INTO `bbs_group` SET gid='101', name="一级用户
组", creditsfrom='0', creditsto='50', allowread='1',
allowthread='1', allowpost='1', allowattach='1', all
owdown='1', allowtop='0', allowupdate='0', allowdele
te='0', allowmove='0', allowbanuser='0', allowdelete
user='0', allowviewip='0';
INSERT INTO `bbs_group` SET gid='102', name="二级用户
组", creditsfrom='50', creditsto='200', allowread=
```

```
'1', allowthread='1', allowpost='1', allowattach='1'
, allowdown='1', allowtop='0', allowupdate='0', allo
wdelete='0', allowmove='0', allowbanuser='0', allowd
eleteuser='0', allowviewip='0';
INSERT INTO `bbs_group` SET gid='103', name="三级用户
组", creditsfrom='200', creditsto='1000', allowread=
'1', allowthread='1', allowpost='1', allowattach='1'
, allowdown='1', allowtop='0', allowupdate='0', allo
wdelete='0', allowmove='0', allowbanuser='0', allowd
eleteuser='0', allowviewip='0';
INSERT INTO `bbs_group` SET gid='104', name="四级用户
组", creditsfrom='1000', creditsto='10000', allowrea
d='1', allowthread='1', allowpost='1', allowattach=
'1', allowdown='1', allowtop='0', allowupdate='0', a
llowdelete='0', allowmove='0', allowbanuser='0', all
owdeleteuser='0', allowviewip='0';
INSERT INTO `bbs_group` SET gid='105', name="五级用户
组", creditsfrom='10000', creditsto='10000000', allo
wread='1', allowthread='1', allowpost='1', allowatta
ch='1', allowdown='1', allowtop='0', allowupdate='0'
, allowdelete='0', allowmove='0', allowbanuser='0',
 allowdeleteuser='0', allowviewip='0';

# 板块表，一级，runtime 中存放 forumlist 格式化以后的数
据。
DROP TABLE IF EXISTS bbs_forum;
CREATE TABLE bbs_forum (
  fid int(11) unsigned NOT NULL auto_increment,
# fid
 # fup int(11) unsigned NOT NULL auto_increment,
```

```
# 上一级版块，二级版块作为插件
  name char(16) NOT NULL default '',            # 版块名称
  rank tinyint(3) unsigned NOT NULL default '0',  # 显示，倒序，数字越大越靠前
  threads mediumint(8) unsigned NOT NULL default '0',   # 主题数
  todayposts mediumint(8) unsigned NOT NULL default '0',# 今日发帖，计划任务每日凌晨 0 点清空为 0，
  todaythreads mediumint(8) unsigned NOT NULL default '0',# 今日发主题，计划任务每日凌晨 0 点清空为 0
  brief text NOT NULL,                    # 版块简介 允许HTML
  announcement text NOT NULL,                # 版块公告 允许HTML
  accesson int(11) unsigned NOT NULL default '0',  # 是否开启权限控制
  orderby tinyint(11) NOT NULL default '0',     # 默认列表排序，0: 顶贴时间 last_date， 1: 发帖时间 tid
  create_date int(11) unsigned NOT NULL default '0',  # 板块创建时间
  icon int(11) unsigned NOT NULL default '0',  # 板块是否有 icon，存放最后更新时间
  moduids char(120) NOT NULL default '',        # 每个版块有多个版主，最多10个： 10*12 = 120，删除用户的时候，如果是版主，则调整后再删除。逗号分隔
  seo_title char(64) NOT NULL default '',       # SEO 标题，如果设置会代替版块名称
  seo_keywords char(64) NOT NULL default '',  # SEO keyword
```

```
  PRIMARY KEY (fid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;
INSERT INTO bbs_forum SET fid='1', name='默认版块', brief='默认版块介绍';
#  cache_date int(11) NOT NULL default '0',      # 最后 threadlist 缓存的时间，6种排序前10页结果缓存。如果是前10页，先读缓存，并依据此字段过期。更新条件：发贴

# 版块访问规则，forum.accesson 开启时生效，记录行数： fid * gid
DROP TABLE IF EXISTS bbs_forum_access;
CREATE TABLE bbs_forum_access (              # 字段中文名
  fid int(11) unsigned NOT NULL default '0',
# fid
  gid int(11) unsigned NOT NULL default '0',
# fid
  allowread tinyint(1) unsigned NOT NULL default '0',   # 允许查看
  allowthread tinyint(1) unsigned NOT NULL default '0',    # 允许发主题
  allowpost tinyint(1) unsigned NOT NULL default '0',   # 允许回复
  allowattach tinyint(1) unsigned NOT NULL default '0',    # 允许上传附件
  allowdown tinyint(1) unsigned NOT NULL default '0',   # 允许下载附件
  PRIMARY KEY (fid, gid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_ge
```

```
neral_ci;

# 论坛主题
DROP TABLE IF EXISTS bbs_thread;
CREATE TABLE bbs_thread (
  fid smallint(6) NOT NULL default '0',          # 版
块 id
  tid int(11) unsigned NOT NULL auto_increment,
# 主题id
  top tinyint(1) NOT NULL default '0',           # 置
顶级别: 0: 普通主题, 1-3 置顶的顺序
  uid int(11) unsigned NOT NULL default '0',
# 用户id
  userip int(11) unsigned NOT NULL default '0',
# 发帖时用户ip ip2long(), 主要用来清理
  subject char(128) NOT NULL default '',         # 主
题
  create_date int(11) unsigned NOT NULL default '0',
# 发帖时间
  last_date int(11) unsigned NOT NULL default '0',
# 最后回复时间
  views int(11) unsigned NOT NULL default '0',
# 查看次数, 剥离出去, 单独的服务, 避免 cache 失效
  posts int(11) unsigned NOT NULL default '0',
# 回帖数
  images tinyint(6) NOT NULL default '0',        # 附
件中包含的图片数
  files tinyint(6) NOT NULL default '0',         # 附
件中包含的文件数
  mods tinyint(6) NOT NULL default '0',          # 预
```

```
留：版主操作次数，如果 > 0，则查询 modlog，显示斑竹的评分
  closed tinyint(1) unsigned NOT NULL default '0',
# 预留：是否关闭，关闭以后不能再回帖、编辑。
  firstpid int(11) unsigned NOT NULL default '0',
# 首贴 pid
  lastuid int(11) unsigned NOT NULL default '0',
# 最近参与的 uid
  lastpid int(11) unsigned NOT NULL default '0',
# 最后回复的 pid
  PRIMARY KEY (tid),                    # 主键
  KEY (lastpid),                        # 最后回复排序
  KEY (fid, tid),                       # 发帖时间排序，正序。数据量大时可以考虑建立小表，对小表进行分区优化，只有数据量达到千万级以上时才需要。
  KEY (fid, lastpid)                    # 顶贴时间排序，倒序
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

# 置顶主题
DROP TABLE IF EXISTS bbs_thread_top;
CREATE TABLE bbs_thread_top (
  fid smallint(6) NOT NULL default '0',          # 查找板块置顶
  tid int(11) unsigned NOT NULL default '0',
# tid
  top int(11) unsigned NOT NULL default '0',
# top: 0 是普通最新贴，> 0 置顶贴。
  PRIMARY KEY (tid),                    #
```

```
  KEY (top, tid),                    # 最新贴: top=0 order by tid desc / 全局置顶:  top=3
  KEY (fid, top)                     # 版块置顶的贴 fid=1 and top=1
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

# 论坛帖子数据
DROP TABLE IF EXISTS bbs_post;
CREATE TABLE bbs_post (
  tid int(11) unsigned NOT NULL default '0',
# 主题id
  pid int(11) unsigned NOT NULL auto_increment,
# 帖子id
  uid int(11) unsigned NOT NULL default '0',
# 用户id
  isfirst int(11) unsigned NOT NULL default '0',
# 是否为首帖，与 thread.firstpid 呼应
  create_date int(11) unsigned NOT NULL default '0',
# 发贴时间
  userip int(11) unsigned NOT NULL default '0',
# 发帖时用户ip ip2long()
  images smallint(6) NOT NULL default '0',      # 附件中包含的图片数
  files smallint(6) NOT NULL default '0',       # 附件中包含的文件数
  doctype tinyint(3) NOT NULL default '0',      # 类型，0: html, 1: txt; 2: markdown; 3: ubb
  quotepid int(11) NOT NULL default '0',        # 引用哪个 pid，可能不存在
```

```
  message longtext NOT NULL,                    # 内容，
用户提示的原始数据
  message_fmt longtext NOT NULL,                # 内容，
存放的过滤后的html内容，可以定期清理，减肥。
  PRIMARY KEY (pid),
  KEY (tid, pid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_ge
neral_ci;

#论坛附件表  只能按照从上往下的方式查找和删除！ 此表如果
大，可以考虑通过 aid 分区。
DROP TABLE IF EXISTS bbs_attach;
CREATE TABLE bbs_attach (
  aid int(11) unsigned NOT NULL auto_increment ,
# 附件id
  tid int(11) NOT NULL default '0',             # 主题id
  pid int(11) NOT NULL default '0',             # 帖子id
  uid int(11) NOT NULL default '0',             # 用户id
  filesize int(8) unsigned NOT NULL default '0',
# 文件尺寸，单位字节
  width mediumint(8) unsigned NOT NULL default '0',
# width > 0 则为图片
  height mediumint(8) unsigned NOT NULL default '0',
# height
  filename char(120) NOT NULL default '',       # 文
件名称，会过滤，并且截断，保存后的文件名，不包含URL前缀 u
pload_url
  orgfilename char(120) NOT NULL default '',
# 上传的原文件名
  filetype char(7) NOT NULL default '',         # 文
```

```
件类型: image/txt/zip，小图标显示 <i class="icon filetype image"></i>
  create_date int(11) unsigned NOT NULL default '0',
# 文件上传时间 UNIX 时间戳
  comment char(100) NOT NULL default '',          # 文件注释 方便于搜索
  downloads int(11) NOT NULL default '0',         # 下载次数，预留
  credits int(11) NOT NULL default '0',           # 需要的积分，预留
  golds int(11) NOT NULL default '0',             # 需要的金币，预留
  rmbs int(11) NOT NULL default '0',              # 需要的人民币，预留
  isimage tinyint(11) NOT NULL default '0',       # 是否为图片
  PRIMARY KEY (aid),                                  # aid
  KEY pid (pid),                                  # 每个帖子下多个附件
  KEY uid (uid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

# 我的主题，每个主题不管回复多少次，只记录一次。大表，需要分区。
DROP TABLE IF EXISTS bbs_mythread;
CREATE TABLE bbs_mythread (
  uid int(11) unsigned NOT NULL default '0',
# uid
  tid int(11) unsigned NOT NULL default '0',
```

```
# 用来清理，删除板块的时候需要
  PRIMARY KEY (uid, tid)                    # 每一个帖子只能插入一次 unique
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

# session 表
# 缓存到 runtime 表。 online_0 全局 online_fid 版块。提高遍历效率。
DROP TABLE IF EXISTS bbs_session;
CREATE TABLE bbs_session (
  sid char(32) NOT NULL default '0',            # 随机生成 id 不能重复 uniqueid() 13 位
  uid int(11) unsigned NOT NULL default '0',
# 用户id 未登录为 0，可以重复
  fid tinyint(3) unsigned NOT NULL default '0',
# 所在的版块
  url char(32) NOT NULL default '',          # 当前访问 url
  ip int(11) unsigned NOT NULL default '0',     # 用户ip
  useragent char(128) NOT NULL default '',      # 用户浏览器信息
  data char(255) NOT NULL default '',           # session 数据，超大数据存入大表。
  bigdata tinyint(1) NOT NULL default '0',      # 是否有大数据。
  last_date int(11) unsigned NOT NULL default '0',
# 上次活动时间
  PRIMARY KEY (sid),
```

```
  KEY ip (ip),
  KEY fid (fid),
  KEY uid (uid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_ge
neral_ci;

DROP TABLE IF EXISTS bbs_session_data;
CREATE TABLE bbs_session_data (
  sid char(32) NOT NULL default '0',             #
  last_date int(11) unsigned NOT NULL default '0',
# 上次活动时间
  data text NOT NULL,                      # 存超大数据
  PRIMARY KEY (sid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_ge
neral_ci;

# 版主操作日志
DROP TABLE IF EXISTS bbs_modlog;
CREATE TABLE bbs_modlog (
  logid int(11) unsigned NOT NULL auto_increment,
# logid
  uid int(11) unsigned NOT NULL default '0',
# 版主 uid
  tid int(11) unsigned NOT NULL default '0',
# 主题id
  pid int(11) unsigned NOT NULL default '0',
# 帖子id
  subject char(32) NOT NULL default '',          # 主
题
  comment char(64) NOT NULL default '',          # 版
```

```
主评价
  rmbs int(11) NOT NULL default '0',            # 加减人民币, 预留
  create_date int(11) unsigned NOT NULL default '0', # 时间
  action char(16) NOT NULL default '',          # top|delete|untop
  PRIMARY KEY (logid),
  KEY (uid, logid),
  KEY (tid)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

# 持久的 key value 数据存储, ttserver, mysql
DROP TABLE IF EXISTS bbs_kv;
CREATE TABLE bbs_kv (
  k char(32) NOT NULL default '',
  v mediumtext NOT NULL,
  expiry int(11) unsigned NOT NULL default '0', # 过期时间
  PRIMARY KEY(k)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_general_ci;

# 缓存表，用来保存临时数据。
DROP TABLE IF EXISTS bbs_cache;
CREATE TABLE bbs_cache (
  k char(32) NOT NULL default '',
  v mediumtext NOT NULL,
  expiry int(11) unsigned NOT NULL default '0',
```

```
# 过期时间
  PRIMARY KEY(k)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_ge
neral_ci;

# 临时队列，用来保存临时数据。
DROP TABLE IF EXISTS bbs_queue;
CREATE TABLE bbs_queue (
  queueid int(11) unsigned NOT NULL default '0',
# 队列 id
  v int(11) NOT NULL default '0',            # 队列中
存放的数据，只能为 int
  expiry int(11) unsigned NOT NULL default '0',
# 过期时间，默认 0，不过期
  UNIQUE KEY(queueid, v),
  KEY(expiry)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_ge
neral_ci;

# 系统表，id
# MAXID 表，几个主要的大表，每天的最大ID，用来削减索引 c
reate_date
# day = 0 表示月；  month = 0 AND day = 0 表示年
# 计划任务，1点执行。 不需要太精准，用来作为过滤条件。
# 可以有效的过滤冷热数据
DROP TABLE IF EXISTS `bbs_table_day`;
CREATE TABLE `bbs_table_day` (
  `year` smallint(11) unsigned NOT NULL DEFAULT '0'
 COMMENT '年',  #
  `month` tinyint(11) unsigned NOT NULL DEFAULT '0'
```

```
 COMMENT '月',  #
  `day` tinyint(11) unsigned NOT NULL DEFAULT '0' CO
MMENT '日',          #
  `create_date` int(11) unsigned NOT NULL DEFAULT
'0' COMMENT '时间戳',    #
  `table` char(16) NOT NULL default '' COMMENT '表名'
,           #
  `maxid` int(11) unsigned NOT NULL DEFAULT '0' COMM
ENT '最大ID',    #
  `count` int(11) unsigned NOT NULL DEFAULT '0' COMM
ENT '总数',          #
  PRIMARY KEY (`year`, `month`, `day`, `table`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

MVC 分层架构

MVC 分层架构

什么是MVC?

MVC 全名是 Model View Controller，是模型(model) - 视图(view) - 控制器(controller)的缩写，一种软件设计典范，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。

其实直白的理解就是，将代码分成三块：

一块：处理业务逻辑，这块叫控制器，英文：Controller，缩写：C

一块：处理数据（对数据进行增删改查，英文叫 CURD），英文：Model，缩写：M

一块：显示模板，在WEB 里就是输出 htm 字符数据，英文叫：View，缩写：V

在 WEB 后端编程里通常采用瀑布流，从头到尾一口气执行完，输出，完工。

但是前端（浏览器端）往往将 Model 和 View 做双向绑定，在 Model 中的数据发生变化的时候，要对 View 进行重绘（刷新）。现在管这种也叫 MVVM。传统的客户端和 WEB 前端往往采用这种模型。

而 Controller 与 URL 路由离的最近，所以也有把 Controller 改叫 Route （路由）。

在 Xiuno BBS 4.0 当中，采用的单入口设计，全部从 index.php 进。

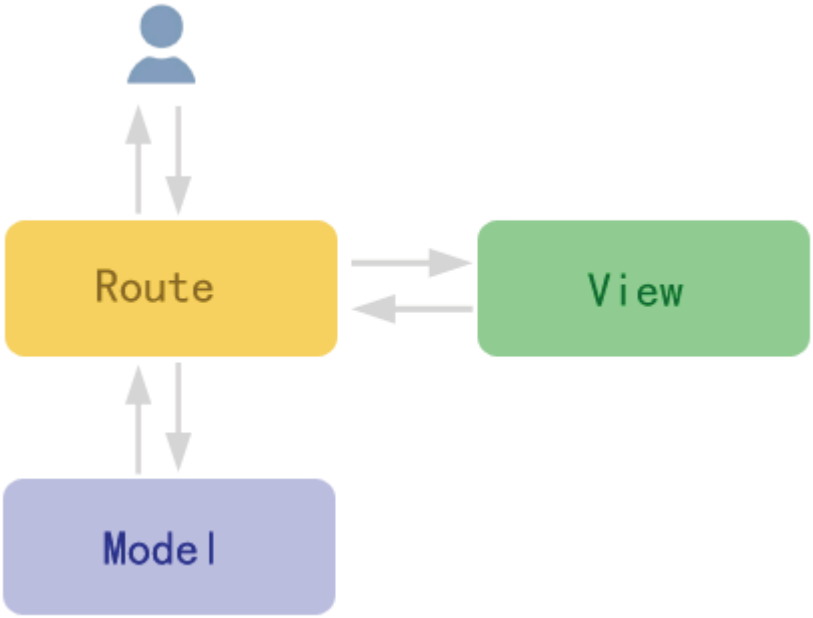
所有的 xxx-xxx.htm 都通过 Web Server 转发到了 index.php?route-action.htm。

由 route 目录下对应的 php 文件进行处理（Controller 层）。

model 则为数据处理目录（Model 层）。

view 为 js css font 等负责显示的文件 目录（View 层）。

图例：



AOP 插件机制

AOP 插件机制

什么是 AOP?

AOP 为 Aspect Oriented Programming 的缩写，意为：面向切面编程，通过预编译方式和运行期动态代理实现程序功能的统一维护的一种技术。AOP 概念早起被应用在 Java 当中，一般被应用到日 志、监控等可以切入的辅助功能里。说的直白一点就是往代码里插入代码，合并后运行。

Xiuno BBS 将这个概念应用到了 WEB 领域，作为对 MVC 机制的补充，取得了不错的效果，其插件机制就是采用的类似 AOP 的概念，开发起来非常的简便。

这样不用在定义大量的 API，只需要在代码文件中加一些注释标示钩子的位置，插件即可插入进来。方便又高效。

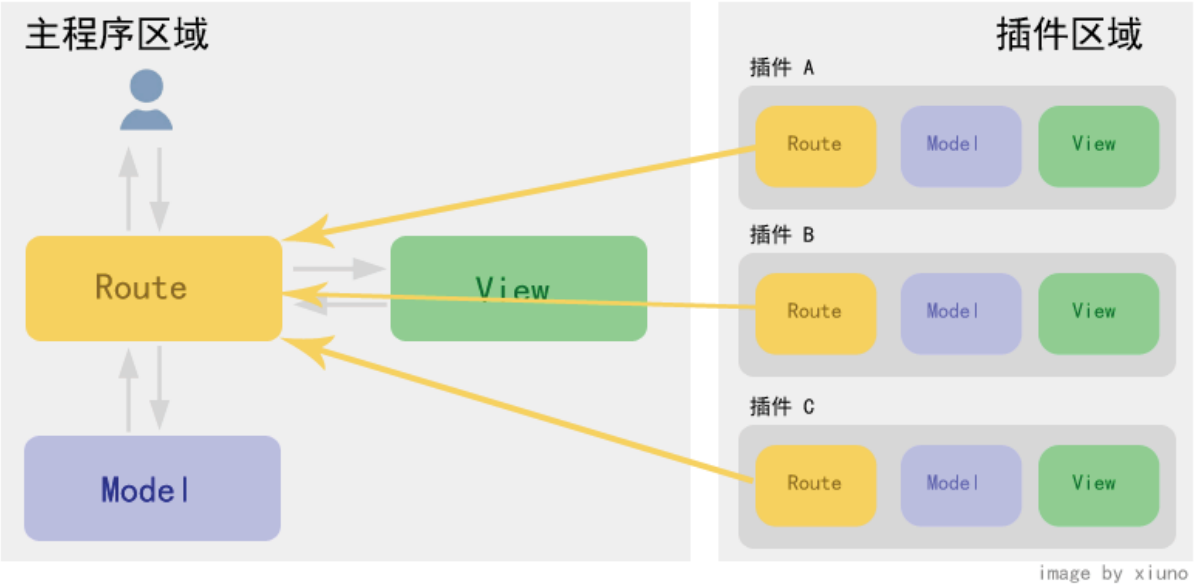


image by xiuno

插件开发

Hello, Xiuno Plugin!

Hello, Xiuno Plugin!

我们来制作一个简单的插件。

首先，我们需要了解下 Xiuno BBS 4.0 的文件结构：

```
conf/ 配置文件目录
lang/ 语言包
log/ 日志目录
tmp/ 临时目录
model/ 数据调用（重用度高）
route/ 业务逻辑（重用度低）
plugin/ 插件目录
upload/ 上传文件
view/ 模板、静态资源（js, css, htm, font）
xiunophp/ 公共的函数库
admin/ 后台管理
index.php 入口程序
```

我们重点关注：plugin, model, view, route 这几个目录。

Xiuno BBS 的插件是基于 AOP 机制，所谓的面向切面编程，也就是往代码里插入代码，合并后再执行（最后合并后的代码存放于 tmp 目录下），一个插件一个目录，我们来示范一下最简单的 Hello, Plugin!

1.打开 index.php，修改 DEBUG 为 2 （这样可以及时看到效果，上线后还原为 0）

```
!defined('DEBUG') AND define('DEBUG', 2);
```

2.新建目录，文件：

```
plugin/
    my_hello/
        conf.json
        hook/
            body_start.htm
```

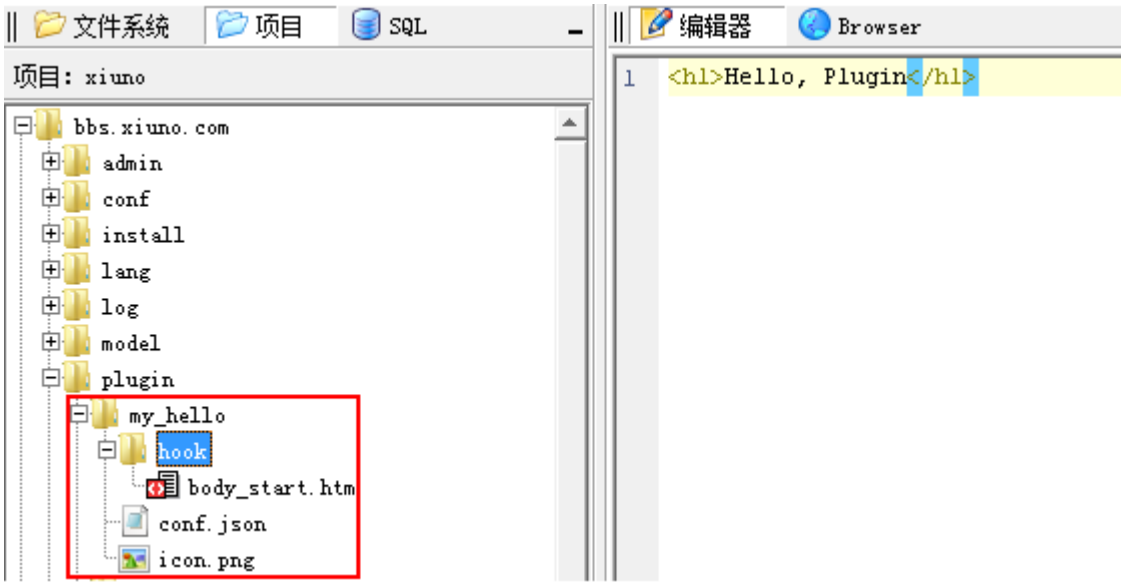
3. body_start.htm 文件内容：

```
<h1>Hello, Plugin</h1>
```

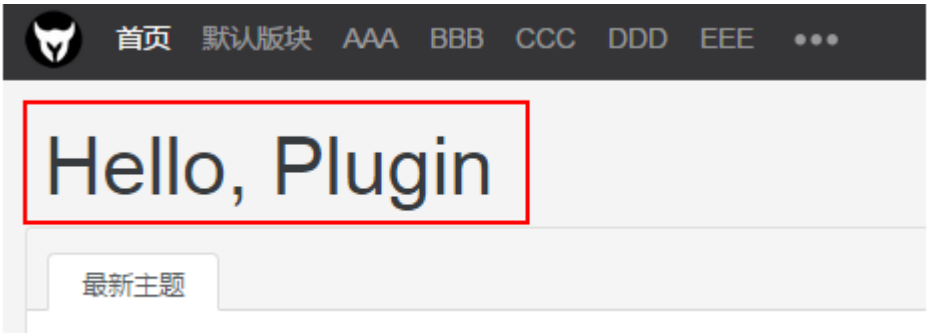
4. conf.json 文件内容：

```
{
    "name":"我的第一个 Xiuno BBS 插件",
    "brief":"我的插件介绍。",
    "version":"1.0",
    "bbs_version":"4.0",
    "installed":1,
    "enable":1,
    "hooks_rank":[],
    "overwrites_rank":[],
    "dependencies":[]
}
```

5.为插件制作一个图标，宽 54 像素，高 54像素,我们这里拷贝一个 plugin/xn_ad/icon.png



6.访问前台，看看效果吧!



【 完】

补充:

Xiuno BBS 预埋了很多 hook，你可以通过打开源代码查找你想插入的地方，比如 view/htm/header.inc.htm 中:

```
<html lang="<?php echo $conf['lang'];?>">
<head>

        <?php echo defined('BASE_HREF') ? '<base href="'.BASE_HREF.'" />' : '' ;?>

        <!--{hook header_mete_before.htm}-->

        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <meta name="author" content="XiunoBBS 4.0" />
        <meta name="keywords" content="<?php echo $header['keywords'];?>" />
        <meta name="description" content="<?php echo $header['description'];?>" />
        <meta name="renderer" content="webkit">
        <meta http-equiv="X-UA-Compatible" content="IE=Edge,chrome=1" >

        <title><?php echo $header['title'];?></title>

        <link rel="shortcut icon" href="<?php echo $conf['view_url'];?>img/favicon.ico"
        <link rel="icon" sizes="32x32" href="<?php echo $conf['view_url'];?>img/favicon
        <link rel="Bookmark" href="<?php echo $conf['view_url'];?>img/favicon.ico" />
        <?php if(DEBUG) { ?>
        <link rel="stylesheet" href="<?php echo $conf['view_url'];?>css/bootstrap.css<?
        <?php } else { ?>
        <link rel="stylesheet" href="<?php echo $conf['view_url'];?>css/bootstrap.min.c
        <?php } ?>

        <!--{hook header_css_after.htm}-->

```

如果你要插入到钩子所在位置，只需要在你所在的插件目录的 hook 目录下，建立同名文件即可。比较常见的几个文件：

```
view/htm/header.inc.htm 头部模板文件
view/htm/footer.inc.htm    页脚模板文件
view/htm/index.htm 首页模板文件
view/htm/forum.htm 列表页模板文件
view/htm/thread.htm 详情页模板文件
view/htm/post.htm 发帖模板页面
route/index.php   首页
route/forum.php 列表页
```

```
route/thread.php 详情页
route/post.php 发帖页
```

hook 机制

Hook 机制

Xiuno BBS 的插件机制分为两种，一种是 Hook，一种是 Overwrite。

所谓 Hook，就是往代码里插入代码，多个插件的代码合并后插入到 hook 指定的位置，最后生成的代码存放于 tmp 目录，被 include

在”Hello, Xiuno Plugin“章节中的实例就是基于 Hook 的。

文件 view/htm/header.inc.htm 中的代码，包含一个 hook header_body_start.htm，我们来将代码插入到此处：

```
...
<body>

<! --{hook header_body_start.htm}-->

<div id="wrapper">
...
```

制作插件 A：

```
plugin/
    my_plugin_a/
        conf.json
        hook/
            header_body_start.htm
```

假定 header_body_start.htm 的内容为：

```
Hello, Pugin A
```

有插件 B：

```
plugin/
    my_plugin_B/
        conf.json
        hook/
            header_body_start.htm
```

假定 header_body_start.htm 的内容为：

```
Hello, Pugin B
```

那么最后生成的文件位置在 tmp/view_htm_header_body_start.htm，内容为：

```
...
<body>

Hello, Pugin A
Hello, Pugin B

<div id="wrapper">
...
```

因为程序在 include 时候做了转换：

```
include _include('./view/htm/header.inc.htm');

// 基本等价于：

include ''./tmp/view_htm_header_body_start.htm;
```

overwrite 机制

Overwrite 机制

我们已经知道了 Hook 机制就是插入合并，那么 Overwrite 就很好理解了。

Overwrite 就是覆盖的意思，Xiuno BBS 的 overwrite 机制就是用来"覆盖"原来的文件。

比如你的插件目录如下：

```
plugin/
    my_plugin/
        conf.json
        overwrite/
            view/
                htm/
                    header.inc.htm
```

那么这个插件的 header.inc.htm 就会“覆盖”view/htm/header.inc.htm，并不是真正的覆盖，而是它优先加载，最后代码合并以后存放到了

```
tmp/view_htm_header.inc.htm
```

以下文件可以被 overwrite：

```
index.inc.php
view/htm/*.htm
route/*.php
```

```
model/*.php
admin/view/htm/*.htm
admin/route/*.php
admin/index.inc.php
admin/menu.conf.php
lang/*.php
```

风格模板

风格模板

Xiuno BBS 4.0 前端基于 Bootstrap 4.0 + jQuery 3.1， 所以通过标准化流程就可以构建自己的风格。

如何玩转 CSS3、SASS、Xiuno 4.0 模板风格?

https://bbs.xiuno.com/thread-20040.htm

如果觉得 SASS 麻烦，可以直接撸 CSS：

http://bbs.xiuno.com/thread-20050.htm

发布你的插件

风格模板

将你的插件目录 my_plugin 打包成 my_plugin.zip，通过以下网址发布：

http://plugin.xiuno.com/

在插件审核通过后，其他人就可以通过后台在线安装了。注意加开发者 QQ 群：2759536，管理员审核的时候可能会有一些问题进行交流，一般是代码格式、性能、安全、易用性方面的改进意见。

插件示例

插件示例

请参看 Xiuno BBS plugin 目录，一个插件一个目录。

一个单页的例子

一个单页的例子

新建目录和文件，假定插件名为 my_plugin：

```
plugin/
    my_plugin/
        conf.json（ 配置文件）
        icon.png（ 图标宽 高：54*54）
        hook/
           index_route_case_end.php （ 插入点，该插入点在 index.php）
        hello.php（ 你的业务逻辑文件）
```

conf.json 内容：

```
{
    "name":"我的第一个 Xiuno BBS 插件",
    "brief":"我的插件介绍。",
    "version":"1.0",
    "bbs_version":"4.0",
    "installed":0,
    "enable":0,
    "hooks_rank":[],
    "overwrites_rank":[],
    "dependencies":[]
}
```

index_route_case_end.php 内容：

```
case 'hello': include APP_PATH.'plugin/my_plugin/hel
lo.php'; break;
```

hello.php 内容：

```
<?php
message(0, 'Hello, Plugin');
?>
```

网址访问：http://mydomain.com/?hello.htm

常见问题

post 表中的 message message_fmt 字段的区别?

如何调用百度编辑器?

Xiuno BBS 4.0 中的几种缓存 API

post 表中的 message message_fmt 字段的区别?

post 表中的 message message_fmt 的字段区别

post 表是 bbs 中的保存帖子的核心表，Xiuno BBS 支持帖子多种数据格式，方便扩展和其他程序的转化。 doctype 用来标示该帖子的内容(message)是何种文档格式，

保留以下格式：0: html, 1: txt; 2: markdown; 3: ubb。message 保存的是原始的格式，message_fmt 保存的是格式化和安全过滤过以后的 HTML 格式数据，可以直接用于显示（用来提高效率）。

论坛帖子数据

DROP TABLE IF EXISTS bbs_post;

CREATE TABLE bbs_post (

tid int(11) unsigned NOT NULL default '0', # 主题id

pid int(11) unsigned NOT NULL auto_increment, # 帖子id

uid int(11) unsigned NOT NULL default '0', # 用户id

isfirst int(11) unsigned NOT NULL default '0', # 是否为首帖，与 thread.firstpid 呼应

create_date int(11) unsigned NOT NULL default '0', # 发贴时间

userip int(11) unsigned NOT NULL default '0', # 发帖时用户ip ip2long()

images smallint(6) NOT NULL default '0', # 附件中包含的图片数

files smallint(6) NOT NULL default '0', # 附件中包含的文件数

doctype tinyint(3) NOT NULL default '0', # 类型，0: html, 1: txt; 2: markdown; 3: ubb

quotepid int(11) NOT NULL default '0', # 引用哪个 pid，可能不存在

message longtext NOT NULL, # 内容，用户提示的原始数据

message_fmt longtext NOT NULL, # 内容，存放的过滤后的html内容，可以定期清理，减肥。

PRIMARY KEY (pid),

KEY (tid, pid)

```
) ENGINE=MyISAM DEFAULT CHARSET=utf8
COLLATE=utf8_general_ci;
```

如何调用百度编辑器?

如何调用百度编辑器?

指定上传的 URL： window.UMEDITOR_CONFIG.upload_url

```
<?php include _include(ADMIN_PATH.'view/htm/footer.inc.htm');?>

<link href="../plugin/xn_umeditor/umeditor/themes/default/css/umeditor.css<?php echo $static_version;?>" type="text/css" rel="stylesheet"/>
<link href="../plugin/xn_umeditor/umeditor/umeditor-bbs.css<?php echo $static_version;?>" type="text/css" rel="stylesheet"/>
<script type="text/javascript" src="../plugin/xn_umeditor/umeditor/umeditor.config.js<?php echo $static_version;?>"></script>

<script>window.UMEDITOR_CONFIG.upload_url = xn.url('product-upload_image');</script>

<script type="text/javascript" src="../plugin/xn_umeditor/umeditor/umeditor.js<?php echo $static_version;?>"></script>
<script type="text/javascript" src="../plugin/xn_umeditor/umeditor/umeditor-insertcode.js<?php echo $static_version;?>"></script>
<script type="text/javascript" src="../plugin/xn_umeditor/umeditor/umeditor-bbs.js<?php echo $static_version;?>"></script>
```

```
<script type="text/javascript" src="../plugin/xn_umeditor/umeditor/lang/zh-cn/zh-cn.js<?php echo $static_version;?>"></script>
```

服务端处理上传，范例：

```
<?php

// ...

if ($action == 'upload_image') {

    $width    = param('width', 0);
    $height   = param('height', 0);
    $is_image = param('is_image', 0);
    $name     = param('name');
    $data     = param_base64('data');

    $conf['upload_url'] = substr(http_url_path(), 0, -6) . $conf['upload_url'];

    $product_image_path = $conf['upload_path'] . 'product_image/' . date('Ym') . '/';
    $product_image_url  = $conf['upload_url'] . 'product_image/' . date('Ym') . '/';
    xn_mkdir($product_image_path, 0777, TRUE);

    empty($group['allowattach']) AND $gid != 1 AND message(-1, '您无权上传');
```

```
    empty($data) AND message(-1, lang('data_is_empt
y'));
    $filesize = strlen($data);
    //$size > 20480000 AND message(-1, lang('filesiz
e_too_large', array('maxsize'=>'20M', 'size'=>$siz
e')));

    // 111.php.shtmll
    $ext       = file_ext($name, 7);
    $filetypes = include APP_PATH . 'conf/attach.con
f.php';
    !in_array($ext, $filetypes['all']) AND $ext =
'_' . $ext;
    $filetype = attach_type($name, $filetypes);

    $tmpanme = $uid . '_' . xn_rand(15) . '.' . $ext
;
    $tmpfile = $product_image_path . $tmpanme;
    $tmpurl  = $product_image_url . $tmpanme;

    file_put_contents($tmpfile, $data) OR message(-1
, lang('write_to_file_failed'));

    $attach = array(
        'url'         => $tmpurl,
        'path'        => $tmpfile,
        'orgfilename' => $name,
        'filetype'    => $filetype,
        'filesize'    => $filesize,
```

```
            'width'         => $width,
            'height'        => $height,
            'isimage'       => $is_image,
        );

        message(0, $attach);

}

?>
```

Xiuno BBS 4.0 中的几种缓存 API

##Xiuno BBS 4.0 中的几种缓存 API

1. 持久存储，永不过期

```
kv_set('key1', 'value1');
kv_get('key1');
kv_delete('key1');
```

2. 缓存，可以设置过期时间

```
cache_set('key1', 'value1', 60);
cache_get('key1');
cache_delete('key1');
```

3. 持久存储，CACHE 加速

```
kv_cache_set('key1', 'value1');
kv_cache_get('key1');
kv_cache_delete('key1');
```

4. 合并到 setting 进行存储，持久存储，并且通过 cache 加速（如果开启 cache）

```
setting_set('key1');
setting_get('key1', 'value1');
```

```
setting_delete('key1');
```

5. 合并到 runtime 中进行存储，持久存储，并且通过 cache 加速（如果开启 cache）

```
runtime_set('key1');
runtime_get('key1', 'value1');
runtime_delete('key1');
```

插件互相卸载机制

插件互相卸载机制

有时候某一类功能，只希望有一个插件，安装多个类似插件会导致功能重复，甚至 BUG。

Xiuno BBS 引入了互相卸载的机制，通过插件名规范来约定。

```
xn_mobile
jack_mobile
tom_mobile
xxx_mobile
```

插件名通过下划线分割，第一个单词是插件作者名缩写，第二个是功能名称，第三个如果有是用来做额外的标志。

功能名称是唯一标志，相同功能名称的插件的插件只会有一个被安装，其他相同功能名的插件都会被卸载。

同理，风格插件也是只能安装一个：

```
xxx_theme_red
yyy_theme_blue
zzz_theme_white
```

其他

JSON API

JSON API

最新的 Xiuno BBS 4.0 git 版本已经加入了全站的 JSON 数据返回支持，方便 APP 或者其他接口调用（后台安装 JSON 插件）

只需要传入参数 ajax=1 或者直接通过 ajax 请求。

开放的接口如下，分为 GET/POST 两大类：

GET：

最新主题：/index-{page}.htm

最新精华：/index-{page}-1.htm

版块最新主题：/forum-{fid}-{page}.htm

版块精华主题：/forum-{fid}-{page}-1.htm

用户最新主题：/user-{uid}-{page}.htm

用户精华主题：/user-{uid}-{page}-1.htm

我的最新主题：/my-thread-{page}.htm

我的精华主题：/my-thread-{page}-1.htm

主题+回帖列表：/thread-{tid}-{page}.htm

搜索：/search-{keyword}.htm

POST：

用户登录：/user-login.htm

email, password（md5 过以后的值）

用户注册：/user-create.htm

email, username, password

统一返回 JSON 格式：

{code: 0, message: "登录成功"}

{code: -1, message: "登录失败"}

{code: 'username', message: "用户名错误"}

{code: 0, message: {"key": "value"}}

发表新主题：/thread-create.htm

fid, subject, doctype, message

doctype 值参考： install/install.sql

发表回复：/post-create.htm

tid, doctype, message

编辑帖子：/post-update-{pid}.htm

subject, message

删除帖子：/post-delete-{pid}.htm

版主管理：参看 route/mod.php

其他请参看代码 route/*.php

注意：

code: 0 表示成功，-1 表示失败，其他值表示错误代码。

实例代码：

JS 调用：

```
<script>$.xget("forum-1.htm", function(code, message
) {
        console.log(message);
});
</script>
```

PHP 调用：

```
<?php
$s = file_get_contents("http://bbs.xiuno.com/forum-
1.htm?ajax=1");
$arr = json_decode($s);
print_r($arr);
?>
```

最新主题，精华主题调用什么的极其简单了，APP 开发读接口数据也更容易了。