**FineCMS 使用手册**

目录

[一、 主要功能模块 2](#_Toc414910091)

[二、 安装及升级 3](#_Toc414910092)

[(一) 环境需求 3](#_Toc414910093)

[(二) 系统安装 3](#_Toc414910094)

[(三) 系统升级 4](#_Toc414910095)

[三、 使用说明 5](#_Toc414910096)

[(四) 系统设置 5](#_Toc414910097)

[(五) 系统安全 5](#_Toc414910098)

[(六) 栏目管理 6](#_Toc414910099)

[(七) 附件管理 6](#_Toc414910100)

[(八) 内容管理 7](#_Toc414910101)

[(九) 推荐位管理 7](#_Toc414910102)

[(十) 文字块管理 8](#_Toc414910103)

[(十一) 会员投稿流程 9](#_Toc414910104)

[四、 功能教程 10](#_Toc414910105)

[(十二) 模型使用说明 10](#_Toc414910106)

[(十三) 字段类别说明 12](#_Toc414910107)

[(十四) [静态/伪静态]自定义URL地址详细教程 16](#_Toc414910108)

[(十五) 生成静态教程 19](#_Toc414910109)

[(十六) 表单使用教程 22](#_Toc414910110)

[(十七) Ucenter使用说明 25](#_Toc414910111)

[(十八) 网站多语言配置方法 27](#_Toc414910112)

[(十九) 后台管理目录(admin)更改方法介绍 28](#_Toc414910113)

[(二十) 部分表的数据结构 29](#_Toc414910114)

[(二十一) 自定义Tag标签的URL规则 30](#_Toc414910115)

[(二十二) 站内搜索参数说明及表单设计 31](#_Toc414910116)

[(二十三) 多站点及分站配置 33](#_Toc414910117)

[(二十四) 会员扩展使用教程 34](#_Toc414910118)

[(二十五) FineCMS 模板设计指南 34](#_Toc414910119)

[(二十六) FineCMS万能标签list功能介绍 36](#_Toc414910120)

[(二十七) 标签语法说明 43](#_Toc414910121)

[(二十八) 系统变量介绍 45](#_Toc414910122)

[(二十九) 系统函数介绍 46](#_Toc414910123)

[(三十) 关于栏目{$cats}调用说明 47](#_Toc414910124)

[五、 进阶教程 49](#_Toc414910125)

[(三十一) FineCMS框架基础知识介绍 49](#_Toc414910126)

[(三十二) FineCMS的Model应用 50](#_Toc414910127)

[(三十三) FineCMS的View应用 63](#_Toc414910128)

[(三十四) FineCMS的Controller应用 66](#_Toc414910129)

[(三十五) FineCMS插件开发指南 68](#_Toc414910130)

[(三十六) 自定义字段类别 71](#_Toc414910131)

[(三十七) 自定义编辑器 74](#_Toc414910132)

[(三十八) 个性化URL结构 75](#_Toc414910133)

[(三十九) 表单模型和会员扩展模型中的回调函数用法 77](#_Toc414910134)

[(四十) 数据读写分离配置 79](#_Toc414910135)

**FineCMS（简称FC）是一款基于PHP+MySql开发的内容管理系统，其中免费版是完全开源的版本，采用MVC设计模式实现业务逻辑与表现层的适当分离，使网页设计师能够轻松 设计出理想的模板， 插件化方式开发功能易用便于扩展，支持自定义内容模型和会员模型，并且可以自定义字段，可面向中小型站点提供重量级网站建设方案。对于那些想要进行开发的技术不足，又南中羞涩的站长们是一个难得的CMS平台。目前FC只有在线手册，使用起来有些不方便我对FC手册进行了整理，希望对各位有一定帮助。**

# 主要功能

**自定义模型和字段**超强的自定义模型和字段功能则把系统灵活度发挥到了极致，不用编程就实现各种信息发布和检索。   
**多站点及分站**  
支持多个站点管理及分站功能，多站只需绑定域名到根目录，使用相当方便。  
**多国语言支持**  
系统自带语言有简体中文、繁体中文、英文，其他语言扩展相当方便。  
**负载能力强**从缓存技术、数据库设计、代码优化等多个角度入手进行优化，支持百万级数据量。  
**模板制作方便**采用MVC设计模式实现了程序与模板完全分离，灵活的模板标签能完全显示全站信息。  **支持文章内链**  
有助于提高搜索引擎对网站的爬行索引效率，支持Tag自动内链到文章。  
**表单功能**  
用于拓展内容模型和会员模型，如报名、评论、询价、咨询等。  
**推荐位功能**推荐位功能可以让编辑随时把信息推送至指定位置，操作简单实用。  
**文字块功能**把一些小段内容放在文字块中，支持HTML代码和图片上传。  
**自定义URL规则**可以完全自定义URL地址规则，包括栏目、内容及自定义页。  
**功能插件化**按照官方提供的插件为蓝本，用户可开发出属于自己的插件。  
**SEO处理**性化设置每个栏目的标题标签、描述标签、关键词标签，自动生成百度谷歌网站地图。

# 安装及升级

## 环境需求

FineCMS 居于PHP和MySQL技术开发，可同时使用于Windows、Linux、Unix平台，环境需求如下：

**1、Windows 平台**  
IIS/Apache + PHP5.2.x + MySQL4/5

**2、Linux/Unix 平台**  
Apache + PHP5.2.x + MySQL3/4/5 (PHP必须在非安全模式下运行)

建议使用平台：Linux + Apache2.2 + PHP5.2.x + MySQL5.0

**3、PHP必须环境或启用的系统函数**  
allow\_url\_fopen  
GD扩展库  
MySQL扩展库  
JSON支持

**4、基本目录结构**  
..../cache        程序缓存目录[必须有可写入权限]  
..../config        系统配置文件目录[必须有可写入权限]  
..../controllers 系统控制器目录  
..../core          系统核心程序目录  
..../extensions 系统扩展程序目录  
..../models      系统模型文件存放目录[必须有可写入权限]  
..../uploadfiles 文件上传目录[必须可写入]  
..../plugins      插件目录[必须可写入]  
..../views         系统模板目录

## 系统安装

1、解压FineCMS软件压缩包，将压缩包内FineCMS目录内的所有文件及目录上传至网站根目录。

2、如果第一次安装系统，系统会自动进入程序安装向导，默认地址为http://网站/index.php?c=install。

3、根据安装向导提示设置必要的数据库连接参数以及必要的文件属性完成系统安装。

4、安装完成之后，系统默认后台管理路径是http://网站/index.php?s=admin，为了安全也可以修改路径（后面有介绍）。

## 系统升级

  
登录管理后台如果发现“系统信息”中出现“有新版本”提示，我们可以

**方法一、重新下载程序（重新安装数据会重置）**  
下载地址：<http://down.finecms.net/finecms.zip>。

**方法二、安装在线升级插件（若无写入权限该插件无法完成升级）**  
如果您是linux服务器，需要将全站设置0777权限，否则无法解压覆盖。

**方法三、下载更新程序（推荐）**  
地址：<http://down.finecms.net/>  
注意：升级包必须按版本顺序下载，解压后按版本顺序覆盖根目录，如果有sql文件，请导入到数据库中（注意修改表前缀哦）。

# 使用说明

## 系统设置



一看就很简单，每一项都有说明的

## 系统安全

一、存在多管理员账号建议开启“后台操作日志”（系统设置-系统相关）。

二、更改管理目录路径（默认admin）。

1、在config/config.ini.php配置文件中找到“ADMIN\_NAMESPACE”，设置成想要的目录名称（如test）。  
2、再将该目录（admin）重命名为前面设置的目录（如“test”，命名规则只能为小写英文字母）。  
3、访问地址就变成了http://网站/后台目录/（如http://网站/test/）。

robots.txt文件中增加屏蔽地址技巧：  
Disallow: /index.php?s=**t**\*   
Disallow: /**t\***  
表示以**t**开头的控制器或者目录禁止蜘蛛爬行，这样可以不暴露管理目录

**注意事项：config.ini.php不能用记事本等软件编辑，且保存编码为utf-8无BOM格式，否则会出大问题。**

三、开启“非法字符过滤”以及开启"攻击日志"（系统设置-系统相关）。

1、非法字符可以到config/attackcode.ini.php中去配置，系统集成了一些非法字符，用户可根据自己网站的需求来配置非法字符。

2、受到非法字符攻击时可以开启邮件通知（系统设置-系统相关-开启邮件通知，一定要设置”网站管理邮箱以及邮件服务器配置“哦）。

3、Ip禁止功能，若有非法字符攻击时系统会记录攻击者Ip、会员的id、受攻击页面、非法字符等信息，可以让其Ip禁止方法（系统设置-禁止访问）。

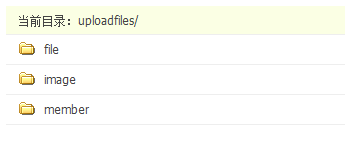
## 栏目管理



如图所示，栏目管理相当方便，内部栏目有直接发布信息的链接。

栏目类型分为：内部栏目（用于发布信息，必须绑定模型）、单网页（单独的一个页面）、外部链接（链接到外部的地址）。

## 附件管理

  
file： 管理员用户文件上传的目录

image：管理员用户图片上传的目录

member：会员附件目录

## 内容管理

  
操作方便，用户可以执行排序、删除、标记、移动、修改、发布等操作

## 推荐位管理

1、推荐位列表操作  


2、推荐位中的信息管理  


3、添加推荐位内容（可以直接在内容表中提取内容）  


## 文字块管理



把一些小段内容放在文字块中，支持HTML代码和图片上传

Tag标签管理



汉字标签自动转换为拼音，支持导入标签。

将标签加入到文章内容中（也就是文章的内链方式）。

1、更新标签缓存。

2、在”系统设置“-”Tag相关“开启”文档内容中关联“（选择”是“）即可。

3、如果文章使用的是生成静态，那么必须重新生成一次，伪静态会立即生效。

## 会员投稿流程

**1、会员/游客投稿流程**  
  
发布稿件 -> 数据存入审核临时表(若会员组权限无需审核则直接发布) -> 管理员审核通过 -> 发布内容表中（同时删除临时表数据）

**2、会员修改文档流程**  
  
修改稿件 -> 修改数据存入审核临时表 -> 待管理员审核通过 -> 更新内容表数据(同时删除临时数据)  
修改稿件 -> 修改数据存入审核临时表 -> 被管理员审核拒绝 -> 修改稿件被退回

# 功能教程

## 模型使用说明

模型是FineCMS的核心，下面来介绍如何创建内容模型（会员模型、表单模型与内容模型一致）

**一、添加一个模型**后台进入“内容模型”，执行“添加”  
   
模型名称：是你为这个模型取一个名字，比如”文章“，”产品“，”报表“  
数据表名：不能重复，小写英文即可，比如”news“，”pro“  
栏目模板、列表模板、内容模板可以自定义，如果不填系统会自动给你取一个名字，必须是".html"文件   
关于”关联表单“在表单章节有介绍

**二、字段管理，添加字段**  
1、进入“模型列表”，执行“字段管理”，见下图  
  
2、可以看到默认的系统字段，你可以对默认的系统字段进行修改，见下图  
   
3、执行菜单上的“添加”，来为当前模型添加字段，见下图  
  
字段别名：当然这个别名指的是字段的标题，比如：文章标题，电话号码等  
字段名称：只能由英文字母、数字和下划线组成，并且仅能字母开头，不以下划线结尾  
字段类别：根据实际情况来选择（[实在不懂的可以参考”字段类别说明“这一节](http://www.dayrui.net/help/3-2.html)）  
字段类型：根据你的情况来设定字段类别，就行mysql建表字段一样  
字段长度：由上面的“字段类型”来决定长度，不能超过了该字段类型的最大长度，不了解可以翻翻Mysql手册  
字段索引：提供了普通、唯一两种索引（不熟悉索引的朋友不要随便选择此项）  
字段提示：提示这个字段是用来干什么的  
前台显示：该字段在会员投稿的时候显示，选择了“隐藏”，会员投稿就不会显示了  
必填字段：这里可以对字段进行有效性验证，比如简单的正则等

## 字段类别说明

**一、字段类别说明**

1、单行文本（input）  
①、参数说明  
长度：文本框的width值，以px为单位  
②、调用方法  
{$字段名称}

2、多行文本（textarea）  
①、参数说明  
宽度：文本框的width值，以px为单位  
高度：文本框height值，以px为单位  
②、调用方法  
{$字段名称}

3、下拉选择框（select）  
①、参数说明  
选项列表：选项名称1|选项值1 (回车换行)  
②、调用方法  
{$字段名称}

4、编辑器（editor）  
①、参数说明  
宽度：编辑器的width值，以百分比为单位  
高度：编辑器的height值，以px为单位  
类型：两种模式而已  
②、调用方法  
{$字段名称}

5、单选按钮（radio）  
①、参数说明  
选项列表：选项名称1|选项值1 (回车换行)  
②、调用方法  
{$字段名称}

6、复选框（checkbox）  
①、参数说明  
选项列表：选项名称1|选项值1 (回车换行)  
②、调用方法

{php $value=string2array($字段名称);} (v1.7.3及以上版本可直接调用$字段名称)

返回数组格式的$value，$value是选中的值，可以用loop循环操作它

7、单图上传（image）  
①、参数说明  
jpg、gif、png格式图片，不得超过2MB  
②、调用方法

{thumb($字段名称)}

8、文件上传（file）  
①、参数说明  
限制格式：多个格式以,号分开，如：zip,rar,tar   
文件大小：单个文件的大小限制，以MB为单位  
②、调用方法

{getfile($字段名称)}

9、多文件上传（files）  
①、参数说明  
限制格式：多个格式以,号分开，如：zip,rar,tar,jpg,png   
文件大小：单个文件的大小限制，以MB为单位  
②、调用方法

{php $value=string2array($字段名称);}(v1.7.3及以上版本可直接调用$字段名称)

文件地址的数组：$value['file']，  
文件名称的数组：$value['alt']  
循环示例如下：

{loop $value['file'] $k=>$file}  
文件地址：{getfile($file)}  
文件名称：{$value['alt'][$k]}  
{/loop}

10、日期时间（date）  
①、参数说明  
显示格式：与PHP的date函数格式一致  
②、调用方法  
{$字段名称}

11、联动菜单（linkage）  
①、参数说明  
绑定菜单：选择已经建好的联动菜单  
联动级别：就是这个菜单有多少级  
②、调用方法

{php $data=linkagelist(联动菜单keyid, 菜单id);}  
{loop $data $t}  
名称：{$t['name']}  
id号：{$t['id']}  
子菜单集合：{$t['arrchilds']} (若无子菜单，该值为空).  
如果该菜单有子菜单就循环调用linkagelist(联动菜单keyid, 子菜单id),依次循环  
{/loop}

更多联动菜单功能介绍请参考系统默认模板中的房产模型及联动菜单相关函数介绍

12、地图字段（map）  
①、参数说明  
百度KEY：到百度去申请  
默认城市：可填写中文，如“北京”  
②、调用方法

{baiduMap(模型id, HTMLdiv的id, $字段, 宽, 高)}

13、组合字段（merge）  
这个最简单，不用多介绍，例子很清楚  
{字段名称}[介绍]，例如：{shi}室，{ting}厅，{wei}卫

14、多字段组合（fields）  
“多字段组合”要区别于“组合字段”  
可以把“多字段组合”看成一个独立字段，而“组合字段”是由多个独立字段组成  
按照公交线路举例：  
建立一个“公交线路”字段，类型为“多字段组合”字段，由“线路”，“车时”，“价钱”等字段组合，用户可以添加多组“公交线路”  
“多字段组合”可以自由删除，排序选项，实现多栏展示数据  
调用方式为：

{php $value=string2array($“多字段组合”名称);}(v1.7.3及以上版本可直接调用$字段名称)  
{loop $value $t} //循环多字段  
值：{$t[字段]}  
{/loop}

**二、自定义字段表单调用**

{list action=field modelid=$modelid name=字段名称 value=字段默认值} ... {/list}

modelid：栏模型id（必选）  
name：字段名称（必选）  
value：字段值（可选）  
返回数据{$t['form']}：用于表单显示（就是一个表单输入框）  
返回数据{$t['data']}：返回复选框、单选按钮中的选项名称和值（数组类型，若要输出请用loop配合）  
例：  
①、调用模型2中的自定义字段zhiduan（默认值200）的输入表单

{list action=field modelid=2 name=zhiduan value=200} {$t['form']} {/list}

②、调用模型2中的自定义字段zhiduan（必须是复选框、单选按钮类型）的值

{list action=field modelid=2 name=zhiduan}  
{loop $t['data'] $n=>$v}  
名称：{$n}，值：{$v}；  
{/loop}  
{/list}

因为$t['data']是数组类型，说以要用loop循环输出，不用解释太多大家都懂

## [静态/伪静态]自定义URL地址详细教程

**一、如何开启自定义URL**  
进入后台，“内容管理”-“栏目管理”，选择修改或者添加栏目，“URL配置”，开启“自定义URL规则”，再配置适合自己的规则。

**二、规则标签说明**  
{dir} ：   表示栏目目录  
{pdir}：  表示包含父级层次的栏目目录（不能与{dir}同时出现，两者二选一）  
{page}：表示分页id  
{id}：     表示内容或栏目的id  
{y}：      表示年（内容规则中有效）  
{m}：     表示月（内容规则中有效）  
{d}：      表示日（内容规则中有效）

**三、举例说明**  
在添加/修改栏目时候，会发现一个选项“URL设置”。  
  
如上图，URL格式4项必填，填写规则有说明，然后保存之后更新缓存和URL地址。  
记住一定要更新缓存并且更新URL地址，保障规则url生效。  
这时你会发现：  
“新闻（news）”栏目的地址已经变成了：http://localhost/finecms/news/china/  
其“内容页面”的地址已经变成了：http://localhost/finecms/news/china/31.html

**问题来了**，你点这些地址，会不会发现会出现“404”错误或者是指向首页，这是为什么呢？解决方案有2种。

**方案1、生成静态页面**  
   
这是静态生成的配置，“静态文件夹”是指静态文件生成的目录，当然也可以不填（留空）表示生成在根目录  
（如果填写了“静态文件夹”，地址就会变成：http://localhost/finecms/静态文件夹/news/china/）  
然后提交保存，再次更新缓存和URL地址，接下来即可生成页面了（不用说了，在“生成静态”中生成~）  
**注意：如果内容URL没有变化的话请“更新内容地址”**

**方案2、设置伪静态**  
①、指向地址说明  
栏目列表指向地址：index.php?c=content&a=list&catdir=栏目目录 或者 index.php?c=content&a=list&catid=栏目ID  
栏目分页指向地址：index.php?c=content&a=list&catdir=栏目目录&page=分页 或者 index.php?c=content&a=list&catid=栏目ID&page=分页  
内容页面指向地址：index.php?c=content&a=show&id=文章id  
内容分页指向地址：index.php?c=content&a=show&id=文章id&page=分页

②、举例说明  
栏目列表：http://www.demo.com/list-news.html （news是栏目目录）  
栏目分页：http://www.demo.com/list-news-1.html （news是栏目目录，1是分页id）  
内容页面：http://www.demo.com/show-1.html （1是文章的id）  
内容分页：http://www.demo.com/show-1-1.html （前者1是文章的id，后者1是分页id）

**伪静态方案一：修改服务器伪静态配置文件**  
以apache为例，修改.htaccess文件：

RewriteEngine On  
RewriteCond %{REQUEST\_FILENAME} !-f  
RewriteCond %{REQUEST\_FILENAME} !-d  
RewriteRule ^list-([a-z0-9A-Z]+).html$ index.php?c=content&a=list&catdir=$1 [L]  
RewriteRule ^list-([a-z0-9A-Z]+)-([0-9]+).html$ index.php?c=content&a=list&catdir=$1&page=$2 [L]  
RewriteRule ^show-([0-9]+).html$ index.php?c=content&a=show&id=$1 [L]  
RewriteRule ^show-([0-9]+)-([0-9]+).html$ index.php?c=content&a=show&id=$1&page=$2 [L]

以iis为例，修改httpd.ini文件：

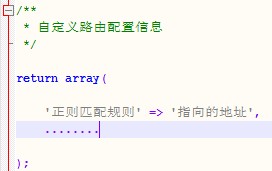
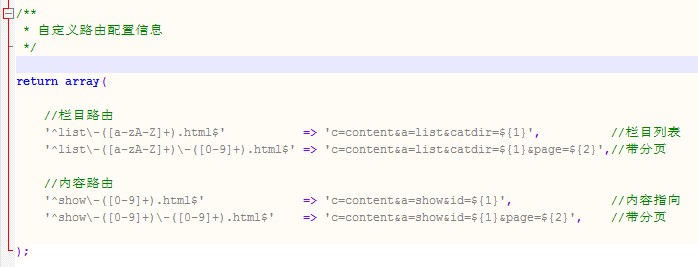
[ISAPI\_Rewrite]  
CacheClockRate 3600  
RepeatLimit 32  
RewriteRule ^(.\*)/list-([a-z0-9A-Z]+)\.html$ $1/index.php?c=content&a=list&catdir=$2 [L]  
RewriteRule ^(.\*)/list-([a-z0-9A-Z]+)-([0-9]+)\.html$ $1/index.php?c=content&a=list&catdir=$2&page=$3 [L]  
RewriteRule ^(.\*)/show-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2 [L]  
RewriteRule ^(.\*)/show-([0-9]+)-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2&page=$3 [L]

以Nginx为例：

rewrite ^([^\.]\*)/list-([a-z0-9A-Z]+)\.html$ $1/index.php?c=content&a=list&catdir=$2 last;  
rewrite ^([^\.]\*)/list-([a-z0-9A-Z]+)-([0-9]+)\.html$ $1/index.php?c=content&a=list&catdir=$2&page=$3 last;  
rewrite ^([^\.]\*)/show-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2 last;  
rewrite ^([^\.]\*)/show-([0-9]+)-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2&page=$3 last;

**伪静态方案二：修改网站配置文件router.ini.php（Apache推荐）**  
当然这种方案的前提需要服务器开启伪静态并指向index.php，再次说一下方法，[goooooo](http://bbs.finecms.net/forum.php?mod=viewthread&tid=175)。  
以apache为例，修改.htaccess文件：

RewriteEngine On  
RewriteCond %{REQUEST\_FILENAME} !-f  
RewriteCond %{REQUEST\_FILENAME} !-d  
RewriteRule !\.(js|ico|gif|jpg|png|css|swf)$ index.php [L]

然后根据上面你自定义的路由格式来设计伪静态指向（**一定要懂正则表达式**）。  
规则匹配文件config/router.ini.php （一定注意不能用记事本之类的工具修改）  
规则格式图：  
   
按照上面的例子，我们可以设计成如下匹配规则：  


熟悉正则表达式的朋友应该一眼能看懂，再次强调使用该方法一定要懂正则表达式哦~~

**注意：伪静态方案二的指向地址不需要再加index.php了**

## 生成静态教程

**方案1、生成静态页面**  
   
这是静态生成的配置，“静态文件夹”是指静态文件生成的目录，当然也可以不填（留空）表示生成在根目录  
（如果填写了“静态文件夹”，地址就会变成：http://localhost/finecms/静态文件夹/news/china/）  
然后提交保存，再次更新缓存和URL地址，接下来即可生成页面了（不用说了，在“生成静态”中生成~）  
**注意：如果内容URL没有变化的话请“更新内容地址”**

**方案2、设置伪静态**  
①、指向地址说明  
栏目列表指向地址：index.php?c=content&a=list&catdir=栏目目录 或者 index.php?c=content&a=list&catid=栏目ID  
栏目分页指向地址：index.php?c=content&a=list&catdir=栏目目录&page=分页 或者 index.php?c=content&a=list&catid=栏目ID&page=分页  
内容页面指向地址：index.php?c=content&a=show&id=文章id  
内容分页指向地址：index.php?c=content&a=show&id=文章id&page=分页

②、举例说明  
栏目列表：http://www.demo.com/list-news.html （news是栏目目录）  
栏目分页：http://www.demo.com/list-news-1.html （news是栏目目录，1是分页id）  
内容页面：http://www.demo.com/show-1.html （1是文章的id）  
内容分页：http://www.demo.com/show-1-1.html （前者1是文章的id，后者1是分页id）

**伪静态方案一：修改服务器伪静态配置文件**  
以apache为例，修改.htaccess文件：

RewriteEngine On  
RewriteCond %{REQUEST\_FILENAME} !-f  
RewriteCond %{REQUEST\_FILENAME} !-d  
RewriteRule ^list-([a-z0-9A-Z]+).html$ index.php?c=content&a=list&catdir=$1 [L]  
RewriteRule ^list-([a-z0-9A-Z]+)-([0-9]+).html$ index.php?c=content&a=list&catdir=$1&page=$2 [L]  
RewriteRule ^show-([0-9]+).html$ index.php?c=content&a=show&id=$1 [L]  
RewriteRule ^show-([0-9]+)-([0-9]+).html$ index.php?c=content&a=show&id=$1&page=$2 [L]

以iis为例，修改httpd.ini文件：

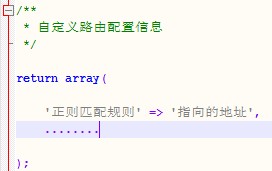
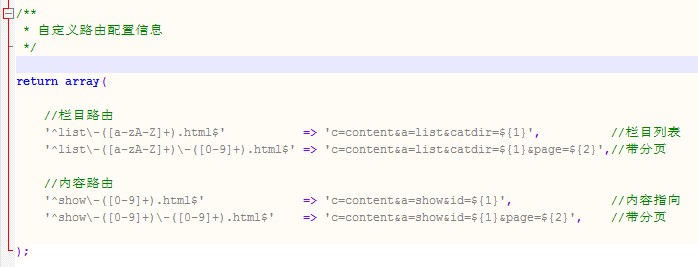
[ISAPI\_Rewrite]  
CacheClockRate 3600  
RepeatLimit 32  
RewriteRule ^(.\*)/list-([a-z0-9A-Z]+)\.html$ $1/index.php?c=content&a=list&catdir=$2 [L]  
RewriteRule ^(.\*)/list-([a-z0-9A-Z]+)-([0-9]+)\.html$ $1/index.php?c=content&a=list&catdir=$2&page=$3 [L]  
RewriteRule ^(.\*)/show-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2 [L]  
RewriteRule ^(.\*)/show-([0-9]+)-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2&page=$3 [L]

以Nginx为例：

rewrite ^([^\.]\*)/list-([a-z0-9A-Z]+)\.html$ $1/index.php?c=content&a=list&catdir=$2 last;  
rewrite ^([^\.]\*)/list-([a-z0-9A-Z]+)-([0-9]+)\.html$ $1/index.php?c=content&a=list&catdir=$2&page=$3 last;  
rewrite ^([^\.]\*)/show-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2 last;  
rewrite ^([^\.]\*)/show-([0-9]+)-([0-9]+)\.html$ $1/index.php?c=content&a=show&id=$2&page=$3 last;

**伪静态方案二：修改网站配置文件router.ini.php（Apache推荐）**  
当然这种方案的前提需要服务器开启伪静态并指向index.php，再次说一下方法，[goooooo](http://bbs.finecms.net/forum.php?mod=viewthread&tid=175)。  
以apache为例，修改.htaccess文件：

RewriteEngine On  
RewriteCond %{REQUEST\_FILENAME} !-f  
RewriteCond %{REQUEST\_FILENAME} !-d  
RewriteRule !\.(js|ico|gif|jpg|png|css|swf)$ index.php [L]

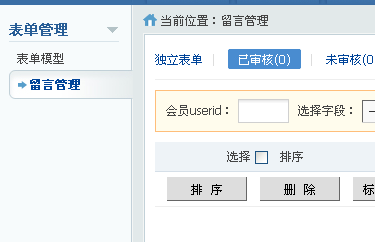
然后根据上面你自定义的路由格式来设计伪静态指向（**一定要懂正则表达式**）。  
规则匹配文件config/router.ini.php （一定注意不能用记事本之类的工具修改）  
规则格式图：  
   
按照上面的例子，我们可以设计成如下匹配规则：  


熟悉正则表达式的朋友应该一眼能看懂，再次强调使用该方法一定要懂正则表达式哦~~

**注意：伪静态方案二的指向地址不需要再加index.php了**

## 表单使用教程

表单，顾名思义就是与前端用户提交数据相关的，如报名、留言、评论等  
FineCMS系统支持两种类型的表单，即是独立表单和关联表单

**一、独立表单介绍**  
  
**1、创建表单**  
后台进入“表单管理”，选择“表单模型”，再“添加”模型  
  
表单模板：提交表单的模板（默认模板中包含“post\_form.html”）  
列表模板：表单数据列表的模板（默认模板中包含“list\_form.html”）  
内容模板：表单数据详细内容的模板（默认模板中包含“list\_form.html”）  
创建模型之后，可以对其增加字段，跟前面的方式一致，不再重复介绍   
  
**2、查看后台表单。**  
创建完成之后，请F5刷新浏览器，再进入“表单管理”，会发现左侧菜单出现了刚才添加的表单“留言管理”   
   
  
**3、配置表单**  
进入“留言管理”，首先需要“配置表单”，各个参数都有说明的，不难看懂  
  
**4、管理表单和发布内容**  
这里不再介绍它了，跟内容管理和发布内容一致

**5、表单预览**  
这里一定要配置好模板，否则无法预览的

**二、关联表单介绍**  
  
**1、创建表单**  
我们来创建一个“文章留言”的表单，跟上面介绍一致，不再描述  
  
**2、配置表单**  
基本跟前面介绍一致  
v1.7.3及以上版本支持在会员中心显示表单内容列表以及“与我相关”的表单内容  
“与我相关”的表单内容是什么意思？举个例子：  
先把该表单“询价”（假设该表单叫“询价”，当然你也可以创建）关联到“商品”模型中；  
**我**发布了一件商品“测试商品示例”，其他人在“测试商品示例”提交的表单内容（询价）将会显示在这里；  
方便**我**查看其他人对**我**的这件商品“测试商品示例”的询价情况。  
  
**3、关联表单到模型**  
可以关联内容模型也可关联到会员模型，我们用“关联内容模型”举例  
进入“内容管理”，再进入“内容模型”，选择需要关联的模型(我们这里关联“文章”)  
   
选择“关联表单”中的“文章留言”，这样就关联好了  
v1.7.3及以上版本支持关联多个表单，操作方式类似   
  
**4、管理表单**  
可以通过上面介绍的方式进行管理，也可通过关联的“文章”模型中管理  
①、通过关联的“文章”模型中管理  
你会发现“文章”模型管理中会多出一个“留言”，这就是管理关联内容的入口  
点击“留言”进入管理页面，跟内容管理类似，下面多了一个“批量关联”命令，用于把表单内容重新关联到其他文章中。  
这里管理只能对当前文章的关联内容管理。  
  
②、通过表单管理（跟上面介绍的一样了）。  
这里是对所有表单内容管理（上面是对单个文章对应的内容管理），顶部链接可以返回到关联的“文章”页面

**5、表单配置（“表单调用”）**  
关联类型的表单，一定要传入参数"cid"(及是文章id)，否则无法提交哦  
格式如：http://localhost/index.php?c=form&a=post&modelid=表单模型id&cid=文章内容id  
其他都和上面介绍的一致了  
  
**三、表单URL处理**  
列表页面地址：index.php?c=form&a=list&modelid=表单模型id&page=分页id&cid=被关联的内容/会员id（cid参数可选）  
内容页面地址：index.php?c=form&a=show&modelid=表单模型id&id=表单内容id  
  
内容页面URL的构架可以用url()函数实现，也可以伪静态或者自定义函数实现  
默认内容页面URL地址是：{url('form/show', array('modelid'=>表单模型id, 'id'=>表单内容id))}   
  
如果采用伪静态模式，就把规则指向上面的地址哦，举个例子：  
表单内容URL地址为：http://xxx/form\_10\_1000.html (10表示模型id，1000表示内容id，内容地址必须传递2个参数)。  
伪静态设置规则为（apache为例）：

RewriteRule ^form\_([0-9]+)\_([0-9]+).html$ index.php?c=form&a=show&modelid=$1&id=$2 [L]

就这么简单，跟淘宝客插件的URL规则设定一致

## Ucenter使用说明

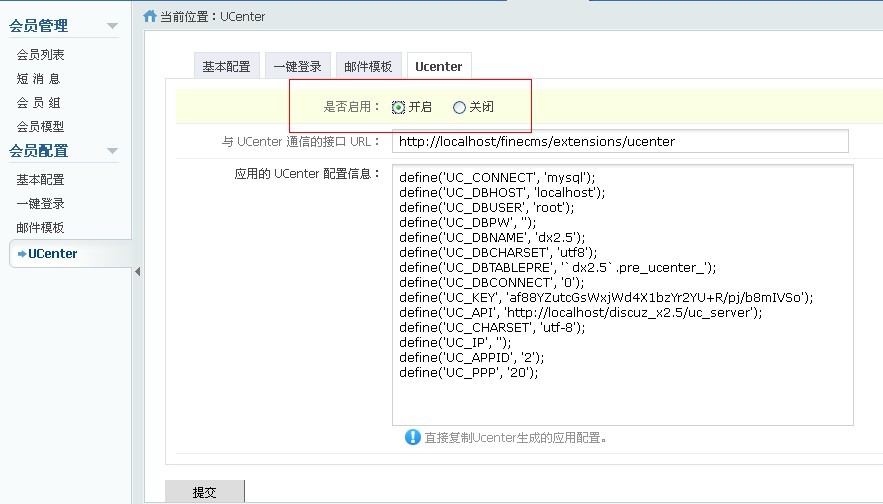
1、进入“会员管理”，复制“与 UCenter 通信的接口 URL”的地址，如下图  


2、进入Ucenter管理中心，添加一个新应用，如下图   


3、应用名称随便写，应用的主URL地址必须填写（1）处复制的地址  


4、选中“开启同步登陆”，然后提交  


5、提交之后，会出现一个配置信息，复制这段代码  


6、把这段代码复制到FineCMS会员管理的Ucenter配置中，注意要选择“开启”Ucenter，再保存  


7、再到Ucenter管理中心查看通信结果  


## 网站多语言配置方法

1. **配置方法**（v1.7.3版本及以上）  
   在config/config.ini.php配置文件中找到“SYS\_LANGUAGE”，默认为“zh-cn”  
     
   **二、语言文件**  
   1、语言目录  
   extensions/language/语言名称/，如“extensions/language/zh-cn/”  
     
   2、文件介绍  
   ①、网站语言文件：lang.php  
   ②、JS的语言文件：lang.js  
   ③、编辑器的语言文件：kindeditor.js  
   ④、日期时间字段的语言文件：calendar.js  
   ⑥、后台模板编辑器的语言文件：edit\_area.js  
     
   **三、配置语言**  
   方法很简单，将上面介绍的文件全部翻译成想要的语言即可  
     
   注意事项：所有文件不能用记事本等软件编辑，且只能保存编码为utf-8无BOM格式，否则会出大问题

## 后台管理目录(admin)更改方法介绍

后台管理目录默认：admin

配置方法（v1.7.3版本及以上）  
1、在config/config.ini.php配置文件中找到“ADMIN\_NAMESPACE”，设置成想要的目录名称（如test）  
2、再将该目录（admin）重命名为前面设置的目录（如“test”，命名规则只能为小写英文字母）  
3、访问地址就变成了http://网站/后台目录/（如http://网站/test/）  
robots.txt文件中增加屏蔽地址技巧：

Disallow: /index.php?s=t\*  
Disallow: /t\*

表示以t开头的控制器或者目录禁止蜘蛛爬行，这样可以不暴露管理目录

如果更改了默认管理目录，原来的访问方式将不会生效（如下图），提高了网站的安全性

补充说明：若是升级用户，在config.ini.php中无法找到“ADMIN\_NAMESPACE”参数，  
请在后台“核心管理”-“基本配置”中点一下提交按钮，系统会重新索引配置文件，“ADMIN\_NAMESPACE”参数就出现了

注意事项：config.ini.php不能用记事本等软件编辑，且保存编码为utf-8无BOM格式，否则会出大问题

## 部分表的数据结构

**1、内容表content\_站点id，如content\_1，表示站点1的内容表**

|  |  |
| --- | --- |
| id | 内容id，自增 |
| catid | 栏目id |
| modelid | 模型id |
| title | 文档标题 |
| thumb | 缩略图 |
| keywords | 关键词，小写分号","分隔 |
| description | 文档简介，描述 |
| url | 内容地址 |
| listorder | 排序值 |
| status | 状态，1：通过，2：拒绝，3：待审，0：回收站，查询数据时只需匹配该字段不等于0即可 |
| hits | 浏览量 |
| userid | 发布人id |
| username | 发布人用户名 |
| inputtime | 文档录入时间 |
| updatetime | 文档更新时间 |

**2、栏目表category**

|  |  |
| --- | --- |
| catid | 栏目id |
| typeid | 栏目类型，1内部栏目，2单页面，3外部链接 |
| modelid | 模型id |
| parentid | 父级栏目，顶级栏目为0 |
| child | 是否存在子栏目，1存在子栏目，0不存在子栏目 |
| arrchildid | 所有子栏目id，应用于catid in ($cat['arrchildid']) |
| catname | 栏目名称 |
| image | 栏目图片 |
| content | 单网页内容 |
| catdir | 栏目目录 |
| url | 栏目地址 |
| items | 该栏目的数据量 |
| ismenu | 是否在菜单中显示（针对顶级栏目） |
| pagesize | 显示数据量 |
| setting | 数组，栏目的相关配置信息，如URL格式，权限等 |

**3、会员表member**

|  |  |
| --- | --- |
| id | 会员id |
| username | 会员账号 |
| email | 邮箱地址 |
| nickname | 会员昵称 |
| avatar | 头像地址 |
| groupid | 会员组id |
| modelid | 会员模型id |
| credits | 会员积分数量 |
| regdate | 注册时间 |
| regip | 注册ip |
| status | 状态，1通过，0未审核 |
| lastloginip | 上次登陆ip |
| lastlogintime | 上次登陆时间戳 |
| loginip | 本次登陆ip |
| logintime | 本次登陆时间戳 |

**4、推荐位数据表position\_data**

|  |  |
| --- | --- |
| id | 推荐位数据id |
| posid | 推荐位置id |
| thumb | 缩略图 |
| title | 文档标题 |
| description | 文档描述，简介 |
| url | 文档地址 |

## 自定义Tag标签的URL规则

“核心管理” - “系统设置” - “TAG相关”  
**1、TAG列表地址规则**  
参数：Tag拼音{tag}，分页{page}  
举例：http://www.xxx.com/tag-**php**-**20**.html（php表示Tag的拼音，20表示第20页）  
规则：tag-{tag}-{page}.html  
指向：c=tag&a=list&kw=**拼音**&page=**分页**  
示例：'^tag-([a-zA-Z0-9]+)-([0-9]+).html$' => 'c=tag&a=list&kw=${1}&page=${2}',  
  
**2、TAG标签地址规则**  
参数：Tag拼音{tag}  
举例：http://www.xxx.com/tag-**php**.html（php表示Tag的拼音）  
规则：tag-{tag}.html  
指向：c=tag&a=list&kw=**拼音**  
示例：'^tag-([a-zA-Z0-9]+).html$' => 'c=tag&a=list&kw=${1}',

## 站内搜索参数说明及表单设计

站内搜索页面地址：index.php?c=content&a=search  
  
**一、搜索方式介绍**  
1、普通搜索：配置一些时间变量就行，“核心管理”-“基本配置”中设置“搜索”  
2、Sphinx：高效搜索方式，要求懂得配置  
  
**二、搜索表单介绍**  
搜索表单最基本的结果如下：

<form action="{url("content/search")}" name="formsearch" method="get">  
<input name="c" type="hidden" value="content" />  
<input name="a" type="hidden" value="search" />  
<input name="kw" type="text" />  
<input name="submit" type="submit" value="搜索" />  
</form>

a，c这两个隐藏域必须存在，kw是搜索关键字（也可以写出username，表示会员名称）  
当然你也可以加入一些自定义字段搜索（要加自定义字段搜索，必须加上栏目catid或者模型modelid）  
如：商品作者：<input name="zuozhe" type="text" />（栏目id或者模型id必须有哦<input name="catid" type="hidden" value="栏目的catid值" />）

**三、相关示例**  
1、按照会员来搜索

<form action="{url("content/search")}" name="formsearch" method="get">  
<input name="c" type="hidden" value="content" />  
<input name="a" type="hidden" value="search" />  
<input name="username" type="text" /> //这里就以username作为搜索条件  
<input name="submit" type="submit" value="搜索" />  
</form>

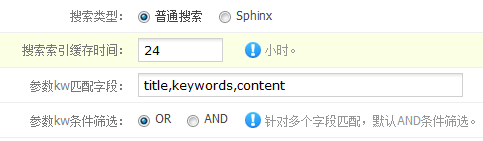
2、在栏目catid=100中按自定义字段aaa来搜索

<form action="{url("content/search")}" name="formsearch" method="get">  
<input name="c" type="hidden" value="content" />  
<input name="a" type="hidden" value="search" />  
<input name="catid" type="hidden" value="100" /> //栏目=100  
<input name="aaa" type="text" /> //aaa是自定义字段  
<input name="submit" type="submit" value="搜索" />  
</form>

**四、URL规则配置**

1、伪静态配置，搜索规则必须服务器支持伪静态  
指向地址：index.php?c=content&a=search&id=搜索id&page=分页id  
  
2、规则配置  
  
必填参数：{id}搜索id，{page}分页id  
举例说明：/search-**99**-page-**10**.html（99是搜索id，10是分页id）  
填写规则：search-{id}-page-{page}.html  
伪  静 态：RewriteRule ^search-([0-9]+)-page-([0-9]+).html$ index.php?c=content&a=search&id=$1&page=$2 [L]

**五、kw匹配字段设置**

  
通过设置kw参数可匹配多个字段，支持附表字段（必须存在catid或者modelid参数）  
举例说明  
1、kw匹配title,keywords,description字段，OR关系  
解析为：title like '%$kw%' **OR** keywords like '%$kw%' **OR** description like '%$kw%'  
模糊匹配一般设置OR关系  
2、kw匹配title,content,cname附表字段，OR关系  
解析为：主表.catid=1 AND (主表.title like '%$kw%' **OR** 附表.content like '%$kw%' **OR** 附表.cnamelike '%$kw%')  
匹配附表字段必须有caitd或者modelid  
 **六、支持显示附表字段**  
参数：more=1（必须存在catid或者modelid参数）  
举例：?c=content&a=search&catid=2&more=1  
  
**七、自定义搜索排序**  
参数：order=字段\_desc[,字段\_asc] 默认DESC排序（若排序附表字段必须存在catid或者modelid参数）  
举例：  
1、按照hits排序：?c=content&a=search&order=hits\_desc  
2、按照栏目3附表字段aaa排序：?c=content&a=search&catid=3&order=aaa（默认为DESC排序）

## 多站点及分站配置

**一、视频教程链接**

1、多站点：<http://bbs.finecms.net/forum.php?mod=viewthread&tid=994&extra=page%3D1>   
2、分站：<http://bbs.finecms.net/forum.php?mod=viewthread&tid=940&extra=page%3D1>

**二、多站点功能介绍**

多站点采用域名匹配方式，不同的域名绑定不同的站点，站点间内容相互独立，可增强负载能力  
  
1、独立站点  
栏目、内容模型及表单模型相互独立，每个站点需要重新创建栏目及模型   
  
2、继承站点  
栏目、内容模型及表单模型继承“被继承的站点”（如地区分站，各个站的栏目都一样），由于该类型站的栏目都一致，因此栏目不能生成静态

**三、模板数据调用**

1、循环多站点的名称

{loop $sites $site\_id=>$config}  
站点id：$site\_id  
站点地址：$config['URL']  
站点名称：$config['SITE\_NAME']  
.... 打印下$config就明白了  
（说明：当前站点id是变量{$siteid}，当前站点高亮显示可以用到它）  
{/loop}

2、调用多站点内容数据

{list site=2 num=10} //调用站点2的所有内容  
....   
{/list}

## 会员扩展使用教程

**一、会员扩展的作用**

1、对会员空间评价、留言

2、会员发布动态信息，如企业的动态、新闻、产品图片等等

3、会员模型的从属关系，如“经销商会员”向某一个“品牌商会员”申请经销、“房产经纪人会员”向某一个“房产中介公司会员”申请加入等等

**二、会员扩展的使用方法**

这一节不再做过多的介绍，可以参考“表单的使用方法”，作用1、作用2跟表单使用差不多，作用3需要懂一下脑筋

## FineCMS 模板设计指南

**配置**系统模板：后台-"核心管理"-“基本配置”-选择需要修改的“模板风格”下拉框  
模板**默认**目录： /views/模板名称/  
  
**一、模板文件介绍**  
1、前端  
网站首页：index.html  
栏目首页：不固定，根据自己设定的模板而定  
栏目列表：不固定，根据自己设定的模板而定  
文档内容：不固定，根据自己设定的模板而定  
单页页面：不固定，默认page.html，根据自己设定的模板而定  
标签页面：keyword.html  
标签列表：tag.html  
搜索页面：search.html  
会员信息：user.html（会员登录信息）  
消息提示：msg.html  
游客投稿：post.html  
  
2、会员  
会员注册：member/register.html  
会员登录：member/login.html  
找回密码：member/repass.html  
激活会员：member/active.html  
绑定会员：member/bang.html  
消息提示：member/msg.html  
会员首页：member/index.html  
会员空间：member/space.html  
收藏夹列：member/favorite.html  
资料修改：member/edit.html  
密码修改：member/password.html  
头像修改：member/avatar.html  
一键登录：member/oauth.html  
内容管理：member/list.html  
发布内容：member/add.html  
附件管理：member/attachment.html  
表单管理：member/form\_list.html  
查看表单：member/form\_show.html  
短消息列：member/pms\_list.html  
发短消息：member/pms\_send.html  
读短消息：member/pms\_read.html  
待审信息：member/verify.html  
  
**二、模板标签**  
标签语法：<http://www.finecms.net/tag/>  
变量函数：<http://www.finecms.net/var/>  
1、所有标签语法及函数遵循PHP语法结构，不做太多解释了  
2、模板标签灵活，完全支持PHP语句  
3、若有JS标签，请区分“{}”系统标识符（“{ js内容 }”括号内加空格来区分就行）  
  
**三、设计技巧**  
1、如何在一个模板中引用另外一个模板或者引用一个公共模板？  
{template 模板名称}: 调用指定“模板名称”，不能加扩展名，支持目录  
  
2、如何根据不同的会员模型来调用不同的会员模板呢？

{if $memberinfo['modelid']==**123**}  
{template 模型为123的模板}  
{/if}

3、如何让文章内容只能让登陆会员查看呢？

{if $memberinfo}  
查看内容等等  
{else}  
你没有登陆不允许查看  
{/if}

4、如何让文章内容只能让某一个会员组查看呢？

{if $memberinfo['groupid'] == **22**}  
可以让会员组groupid是22的会员查看  
{else}  
无权限查看  
{/if}

5、如何让会员发布信息页不同的模型或者栏目使用不同的模板？

{if $catid==**22**}  
{template member/栏目是22的模板名称}  
{/if}

模型的话就改成$modelid==xxx就ok  
  
**四、注意问题**  
1、若修改过模板文件，请务必把默认目录（把"default"改成其他的）更改一下，否则下次升级会被覆盖  
2、不能用记事本等软件编辑，且只能保存编码为utf-8无BOM格式，否则会出大问题

## FineCMS万能标签list功能介绍

|  |  |
| --- | --- |
| action | 支持"推荐位(position)"、"搜索关键字(keywords)"、"相关文章(relation)"模式，普通数据查询不需要此参数 |
| table | 循环表名称，默认为内容表，可以用其他表，插件表格式为：插件名称.表名称(link.linkname) |
| catid | 当前栏目ID号 |
| num | 显示数量，还可以写成"num=1,5"相当于limit 1,5 |
| return | 返回数组名称，用于循环，默认值为't'，调用方式:{$t['title']} |
| more | 值为1时，自动调用内容表的附表内容（也就是自定义的字段） |
| order | 排序，例如: order=id\_desc,userid\_desc，默认desc排序，可以省掉\_desc |
| join | 联合查询表名称（用该表的主键关联），例如：join=member |
| on | 其他表的字段（非join表的自段），必须与join成对出现，否则join失效，例如：join=member on=userid |
| page | 用于分页查询，如果出现了num参数，分页查询将会失效 |
| urlrule | 分页URL格式（分页标识符\_page\_或者[page]），例如:urlrule=$catid-[page].html |
| pagesize | 分页查询每页显示的记录数量（如果出现catid，系统默认会使用该栏目的分页数量） |
| cache | 数据缓存，单位秒，在这缓存时间内调用缓存文件而不再去访问数据库 |
| thumb | 值等于1的时候，显示有缩略图的文档；等于0的时候，显示无缩略图的文档（v1.7） |
| form | 调用表单，例如:form=liuyan ，就是调用liuyan的表单数据（v1.7.5） |
| site | 站点id，用于多站点调用，例如: site=2 table=content ，就是调用站点2的内容表（v1.7.5） |
| fields | 为sql语句指定字段，例如: fields=title,url ，与sql语法一致（不能使用`），支持自定义字段名称（v1.7.5） |
| extend | 对特殊sql语句进行延展，具体教程[传送口](http://bbs.finecms.net/forum.php?mod=viewthread&tid=871) |

格式： {list 参数1=值1 参数2=值2 ... 参数N=值N} ... 循环体 ... {/list}

注意：参数和值不能出现引号

**一、参数介绍**  
  
1、action  
模式选择参数，支持"推荐位(position)"、"搜索关键字(keywords)"、"相关文章(relation)"、"自定义字段信息(field)"模式  
例如：推荐位查询

{list action=position id=1} ... {/list}

返回推荐位id=1的数据并循环  
  
2、return  
数据返回变量，默认为t，循环内直接使用{$t[字段]}

{list ... return=item} {$item['title']} {/list}

返回变量就成了$item，而不再是$t，多层list标签可以用它来解决变量冲突  
  
3、more  
当其值为1时，表示联合附表查询，必须与catid或者modelid参数联合使用

{list catid=$catid more=1 cache=36000} ... {$t[附表字段]} ... {/list}

可快速查询出该栏目的自定义字段数据，可配合fields参数减少缓存大小  
  
4、join和on  
联合查询表名称，必须与on成对出现

{list table=content join=member on=userid} ... {/list}

表content与member关联查询，on是表content中的关联字段，就是以member表的主键与content表的userid字段先关联查询  
  
5、page、pagesize(显示数量)和urlrule(分页规则)  
分页查询，如果存在caitd，则使用该栏目的分页规则和显示数量，返回分页结果变量{$pagelist}

{list catid=$catid page=$page} ... {/list}

栏目分页时，会调用该栏目的分页规则配置，无需后面两个参数  
  
6、num  
Limit控制，最大显示数量

{list catid=$catid num=5} ... {/list}

最大显示5条记录(可以写成num=1,5，表示从1开始到5行结束)  
  
7、order

{list catid=$catid order=updatetime,id} ... {/list}

示例按updatetime降序，id降序，如果升序就附加:字段\_ASC（默认是降序）  
  
8、字段作为参数（同一字段只能出现一次）

{list catid=1 username=admin num=10} ...{/list}

其中的username是内容表中的字段，解析为：catid=1 and username='admin'

{list NOTcatid=1 num=10} ...{/list}

解析为：catid not in (栏目集合)  
  
9、cache  
缓存参数，推荐list查询都加上它

{list catid=1 cache=3600} ...{/list}

缓存3600秒（1小时），在3600秒内调用缓存数据而不访问数据库，减轻服务器压力  
  
10、thumb  
是否显示缩略图

{list thumb=1 cache=3600} ...{/list}

当thumb=1时，表示显示有缩略图的文档，相当于SQL中的thumb<>''；当thumb=0时，相当于SQL中的thumb=''  
  
11、form  
表单数据查询快捷参数

{list form=liuyan cache=3600} ...{/list}

调用表单，例如表单为liuyan ，示例就是调用liuyan的表单数据，当然相对应本站点  
  
12、site  
用于查询指定站点数据

{list site=2 table=content num=10 cache=3600} ...{/list}

站点id，用于多站点调用，示例就是调用站点2的内容表数据  
  
12、fields  
为sql语句指定字段，与sql语法一致（各个字段不能使用`字段名`，程序会自动为你加上），支持自定义字段名称

{list fields=title,url,content catid=1 more=1 num=10 cache=3600} ...{/list}

示例中解析为SQL是：select `主表`.`title`,`主表`.`url`,`附表`.`content` from .... 

**二、相关示例**  
  
1、栏目查询

{list catid=$catid order=updatetime num=10} ... {/list}

查询当前栏目的10条数据，并按updatetime desc排序  
  
2、栏目分页查询

{list catid=$catid order=updatetime page=$page} ... {/list}

查询当前栏目，按照当前栏目默认分页方式分页  
  
3、自定义分页查询

{list catid=$catid order=updatetime page=$page pagesize=10 urlrule=$catid-[page].html} ... {/list}

分页查询，每页显示10条，url地址格式为：栏目id-分页id.html  
  
4、联合附表的自定义字段查询

{list catid=$catid more=1 num=10} ... {/list}

5、Join关联查询

{list catid=$catid join=member on=userid} ... {/list}

解析为：from 内容表 left join member on 内容表.userid=member.主键  
  
6、自定义表查询

{list table=member groupid=1} ... {/list}

查询member表中groupid=1的用户  
  
7、插件表查询

{list table=review.revirew\_total contentid=10} ... {/list}

查询review插件中的表review\_total中contentid=10的数据

**三、Action参数**  
  
1、推荐位(position)

{list action=position id=1 catid=$catid} ... {/list}

id：推荐位的id  
catid：可选，根据推荐位的类别而决定  
输出字段请参考position\_data表结构，不用解释太多大家都懂  
  
2、搜索关键字(keywords)

{list action=keywords order=addtime} ... {/list}

order：排序字段参考search表  
输出字段请参考search表结构，不用解释太多大家都懂  
  
3、相关文章(relation)  
①、调用手动添加的相关文章

{list action=relation id=$id num=10} ... {/list}

id：文章的id  
num：显示数量，默认10条  
②、通过关键字自动调用相关文章

{list action=relation tag=$keywords id=$id num=5}...{/list}

id：当前文章id  
tag：当前文章的关键字  
catid：栏目id(可选)  
modelid：模型id(可选)  
  
4、自定义字段显示(field)（适合自定义字段表单显示）

{list action=field modelid=$modelid name=字段名称 value=字段默认值} ... {/list}

modelid：栏模型id（必选）  
name：字段名称（必选）  
value：字段值（可选）  
返回数据{$t['form']}：用于表单显示（就是一个表单输入框）  
返回数据{$t['data']}：返回复选框、单选按钮中的选项名称和值（数组类型，若要输出请用loop配合）  
例：  
①、调用模型2中的自定义字段zhiduan（默认值200）的输入表单

{list action=field modelid=2 name=zhiduan value=200} {$t['form']} {/list}

②、调用模型2中的自定义字段zhiduan（必须是复选框、单选按钮类型）的值

{list action=field modelid=2 name=zhiduan}  
{loop $t['data'] $n=>$v}  
名称：{$n}，值：{$v}；  
{/loop}  
{/list}

因为$t['data']是数组类型，说以要用loop循环输出，不用解释太多大家都懂

**四、特殊字段辅助参数**  
  
1、NOTcatid=值  
注：NOT只能与栏目catid搭配  
表示catid not in (栏目集合)，不用解释太多大家都懂  
  
2、OR字段名称=值

{list catid=$catid ORuserid=1 ... }

表示where ... catid=$catid or userid=1  
3、IN字段名称=值

{list catid=$catid INstatus=0,1,2 ... }

表示where ... catid=$catid and status in (0,1,2)  
  
4、BW字段名称=值

{list catid=$catid BWstatus=1,2 ... }

表示where ... catid=$catid and status between 1 and 2  
  
5、LIKE字段名称=值

{list LIKEtitle=%finecms% ...}

表示where ... title LIKE '%finecms%'，不用解释太多大家都懂

**五、返回数据**  
  
1、循环变量return=?，默认为$t  
2、分页变量{$pagelist}  
3、SQL查询字符串{$sql}（用于测试，打印当前查询的sql语句）  
4、统计变量{$total}，返回数据总数量  
5、循环计数{$key}，用于list循环计数变量，从0开始  
6、循环统计变量{$count}，统计本次循环记录数量  
7、支持多层lsit，防止变量冲突，必须设置不同的return值（v1.7.1及以上版本）  
如果存在return值，那么所有变量必须加上后缀"\_return值"，例如：

{list ... return=s}  
循环计数变量：{$key\_s}  
{/list}

分页：{$pagelist\_s}  
总数：{$total\_s}

## 标签语法说明

**一、模板变量**1、普通变量：{$finecms}  
2、数组变量：{$finecms['name']}  
3、系统常量：{SITE\_PATH}

**二、判断语句**{if $模板变量1 运算符 $模板变量}  
模板内容1  
{else}  
模板内容2  
{/if}  
或  
{if $模板变量1 运算符 $模板变量2}  
模板内容1  
{else if $模板变量1 运算符 $模板变量3}  
模板内容2  
{else}  
模板内容3  
{/if}  
注：这里的运算符支持：> 、< 、>= 、<=、==、！= 、<>

**三、循环语句**  
{loop 模板数组 $索引变量=>$值变量}  
执行语句  
{/loop}  
或  
{loop 模板数组 $值变量}  
执行语句  
{/loop}

**四、Template模板**  
{template 模板文件}或者{include 模板文件}，此模板文件一般为共享性模板文件  
例如：{template header} ，模板文件这里省略了.html的后缀名

**五、执行函数**  
1、{php PHP语句}，{php echo "1";}  
2、{函数名(参数)}，{strlen($str)}相当于<?php echo strlen($str)?>

**六、页面查询**  
①、{sql:模型名称 执行方法}  
格式：{sql:user where("pid=1")->order("uid asc")->select();}  
返回：$return数组

②、{plugin:插件名称::表名称 执行方法}  
格式：{plugin:link::link findAll()}  
返回：$return数组

## 系统变量介绍

**一、通用变量**

1、{SITE\_PATH}：不带域名的网站地址  
2、{SITE\_URL}：带域名的网站地址  
3、{CMS\_NAME}：软件名称  
4、{CMS\_VERSION}：软件版本  
5、{CMS\_UPDATE}：更新时间  
6、{SITE\_THEME}：前台风格路径  
7、{ADMIN\_THEME}：后台风格路径  
8、{EXT\_PATH}：扩展目录路径  
9、{LANG\_PATH}：语言目录路径  
10、{$数组键名}：网站配置信息（config.ini.php），如{$SITE\_NAME}  
11、数组{$cats}：所有栏目数组（{$cats[栏目ID]}）  
12、{$meta\_title}：网页title标题  
13、{$meta\_keywords}：网页关键词  
14、{$meta\_description}：网页描述信息  
15、{$indexc}：是否是首页index  
16、{$param[参数名称]}：用于接收URL参数  
17、{$memberinfo}：当前登陆会员的信息，数组格式  
18、{$membermodel}：会员模型数组  
19、{$membergroup}：会员组数组  
20、{$sites}：多站点数组  
21、{$siteid}：当前站点的id

**二、栏目列表变量**

1、{$page}：当前栏目的页数  
2、{$arrchilds}：内部栏目的子栏目集（用于栏目列表查询list）  
3、{$pageurl}：栏目分页地址格式（用于分页）  
4、{$catid}：栏目id  
5、{$catname}：栏目名称  
... 自动分析category表中字段

**三、搜索列表变量**

1、{$searchdata}：搜索结果集  
2、{$searchpage}：搜索分页  
3、{$kw}：关键字   
4、{$searchnums}：搜索结果总数  
5、{$modelid}： 搜索模型id  
6、{$catid}： 搜索栏目id

**四、内容页变量**

1、{$cat}：当前栏目信息  
2、{$page}：分页id  
3、{$prev\_page}：上一篇数据  
4、{$next\_page}：下一篇数据  
5、{$pageurl}：分页格式  
6、{$contentpage}：内容分页（多维数组[分页id=>对应URL地址]）  
... 自动分析content表与对应附表中字段

## 系统函数介绍

1、image($thumb)：返回完整的图片地址

 2、thumb($thumb,宽,高)：如果存在缩略图，优先调用缩略图

 3、strcut($string, 长度, 截取后替代字符)：字符截取

 4、clearhtml($text)：清除HTML格式

 5、catpos(栏目ID, 间隔符)：栏目面包屑导航

 6、plugin(插件名称)：运行插件或者判断内置控制器型插件是否存在

 7、getCatNav($catid)：当前栏目的子栏目或同级栏目菜单

 8、getParentData($catid)：父级栏目的信息

 9、baiduMap($catid/$modelid[说明：v1.7.2及以下使用$catid，后面版本使用$modelid], '地图div的id', 地图值, 宽, 高)：百度地图显示

10、string2array/array2string：字符串转化为数组/数组转换为字符串

11、linkagelist(顶级菜单keyid, 显示菜单列表的id)：显示联动菜单子集或者同集，数组格式

12、linkagedata(顶级菜单keyid, 菜单id)：取得菜单id的数据，数组格式

13、linkageids(顶级菜单keyid, 菜单id)：取得菜单id的父级菜单

14、linkagepos(顶级菜单keyid, 菜单id, URL规则, 分隔符)：菜单面包屑导航，URL规则中以{linkageid}匹配菜单id

15、linkageform(顶级菜单keyid, 默认值, 表单名称, 菜单级数)：调用联动菜单表单选择框

16、getfile($file)：返回文件完整地址

17、tag\_url(关键词)：返回Tag标签地址

18、get\_member\_avatar(会员id, 大小120/90/45/30)：调用会员头像

19、get\_member\_info(会员id, $more=0)：会员详细详细，若more=1显示自定义字段内容

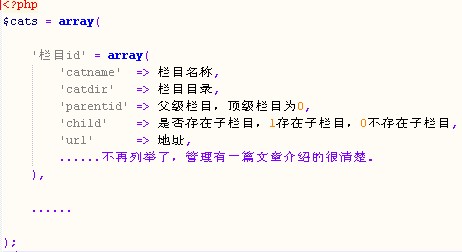
20、get\_tag\_data(关键字字段)：返回关键字数据，数组格式（v1.7.3以上）

21、count\_member\_size(会员id)：统计会员附件目录大小（v1.7.5以上）

22、form\_show\_url(表单模型id, 表单数据)：返回表单自定义url地址（v1.7.5以上）

23、html\_to\_data(数据数组, 模板文件)：用于静态页面，将“数据数组”解析到指定“模板文件”中，Js动态调用它（v1.7.5以上）

## 关于栏目{$cats}调用说明

FineCMS栏目信息保存在全局数组$cats中，当然你要知道栏目数组的结构了  
  
通过这张图，大家应该对$cats数组结构了如指掌了吧

1、调用顶级栏目

{loop $cats $cat}  
{if $cat['paentid'] == 0 && $cat['ismenu']} //这里的$cat[‘ismenu’]是指作为菜单（在栏目管理处可以看到）。  
业务逻辑处理....  
{/if}  
{/loop}

2、调用顶级栏目（存在下级栏目）

{loop $cats $cat}  
{if $cat['paentid'] == 0 && $cat['ismenu']} //先调出顶级栏目  
业务逻辑处理....  
{if $cat['child']} //判断该栏目是否有子栏目，N级栏目以此类推。  
{loop $cats $a} //循环栏目组，$a防止与$cat冲突。  
{if $a['parentid']==$cat['catid']} //子栏目的父栏目id=它父栏目id  
子栏目业务逻辑处理....  
{/if}  
{/loop}  
{/if}  
{/if}  
{/loop}

3、循环栏目文章

{loop $cats $cat}  
{if $cat['paentid'] == 0 && $cat['typeid']==1} //先调出顶级栏目,且为内部栏目。  
<h1>当前顶级栏目：{$cat['catname']}</h1>  
{if $cat['child']} //判断该栏目是否有子栏目。  
{loop $cats $a} //循环子栏目数组，赋值$a。  
{if $a['parentid']==$cat['catid']} //子栏目的父栏目id=它父栏目id  
{list catid=$a[catid] num=10}  
<li>{$t['title']}</li> //循环子栏目中的文章i信息  
{/list}  
{/if}  
{/loop}  
{/if}  
{/if}  
{/loop}

4、献给小白，在列表中取栏目信息

{list catid=$catid .....}  
栏目：{$cats[$t['catid']]['catname']}  
{/list}

# 进阶教程

## FineCMS框架基础知识介绍

**一、目录结构**  
admin：         后台面板引入文件  
cache：         缓存目录，必须具有读写权限  
config：         配置目录，部分文件需要开启读写权限  
controllers：  控制器目录，支持子目录（namespace）  
core：            框架核心  
extensions：  扩展目录，用于放置第三方控件  
models：        数据模型目录  
plugins：        插件目录  
uploadfiles：上传文件目录  
views：          视图模板目录  
index.php：   项目入口文件

**二、命名规则**  
1、数据表命名规则：      数据库中数据表的命名将全部采用小写字母  
2、配置文件的命名规则：配置文件名字字母全部小写,：后缀为.ini.php如：config.ini.php  
3、控制器命名规则：      首字母大写+Controller.php如IndexController.php  
4、数据模型命名规则：  首字母大写+Model.php 如UserModel.php  
5、视图文件命名规则：  有效的html文件即可如index.html

**三、控制器名字空间Namespace**  
相当于控制器目录，如admin就是一个名字空间

**四、控制器Controller文件的开发规则**

<?php  
class IndexController extends Common {  
public function \_\_construct() {  
parent::\_\_construct();  
}  
public function indexAction() {  
内容  
}  
}

Controller文件的命名采用驼峰命名法则如：建立index的controller文件，其名字为IndexController.php  
解释：index首字母要大写，后再加上Controller,注意这个Controller也是首字母大写的  
Controller文件名字和内容中类的名字相同(字母大小写也一致的相同)  
如上所示，IndexController.php的内容类的名字按照此规应为：IndexController  
注意：继承Common，也可以直接继承Controller，推荐继承Common，因为系统会自动加载Common，它将作为每个namespace的公共控制器类

访问地址：index.php?s=Namespace名称&c=控制器名称&a=Action名称

五、Model文件的开发规则  
Model文件的开发规则类似于Controller文件的开发规则只是把Controller换成Model的区别，最好参考Controller文件的开发规则来理解  
Model文件命名采用驼峰命名法则  
Model文件内容类的名字和文件名要一致  
例：要建一个新Model文件，假设为demo表的model文件文件名字为：DemoModel.php内容为：

<?php  
class DemoModel extends Model {  
public function get\_primary\_key() {  
return $this->primary\_key = 'id'; //表demo的主键  
}  
  
public function get\_fields() {  
return $this->get\_table\_fields();  
}  
  
}

解释：以上格式是固定格式，如果需要扩展，可在里面添加function

## FineCMS的Model应用

模型(Model)是 MVC里的M，模型用于操作数据以及数据相关的业务规则。

**一、创建模型**  
在项目目录models中新建Model文件(具体如何操作，前面介绍时，已经详细讲述过了)。  
例如：要建UserModel.php文件(注：Model文件中所有的类必须继承Model基类)。

<?php  
class UserModel extends Model {  
public function get\_primary\_key() {  
return $this->primary\_key = 'id'; //设置主键  
}  
public function get\_fields() {  
return $this->get\_table\_fields();  
}  
}

Model文件创建好后，怎么在Controller文件中调用呢？  
首先对model进行实例化  
$model = $this->model('user');  
如果查询user数据表的所有数据，操作如下：  
$data = $model->findAll();  
上面的例子，用到函数：findAll()，这就是在Model类中封装的函数。

**二、Model 类所提供的函数及说明。**  
1、from($tabs, $fields) 实现SQL语句中：“select `数据表字段` from 数据表 ”部分的处理。  
参数说明：  
$tabs为所要查询的数据表。如果所要查询的数据表为多个(大于1个)，参数要为数组。如：array(''数据表1, '数据表2', ...) ;  
注：当参数为空时,数据表则默认为当前model所对应的数据表。  
$fields 为所要查询的数据表字段。默认为“\*”，即当此参数为null时，则视为查询数据表所有的字段。  
如果查询字段为多个，参数要为数组类型。如：array('字段1', '字段2');  
使用举例：  
例一、  
$model->from('user');  
SQL语句为：select \* from user

例二、  
$model->from('user', array('userid', 'username'));  
显示SQL语句为：SELECT `userid`, `username` FROM `user`

例三、  
$model->from(array('content', 'user'), array('title', 'username'));  
显示SQL语句为：SELECT `title`, `username` FROM `content`, `user`

例四、  
$model->from(array('p'=>'content', 'u'=>'user'), array('p.title', 'u.username'));  
显示SQL语句为：  
SELECT `p`.`title`, `u`.`username` FROM `content` AS `p`, `user` AS `u`  
注：array('p'=>'content', 'u'=>'user')，这里数组索引为非数字时，则代表SQL语句中的ALIAS(更名)情况。

例五、  
$model->from(array('p'=>'content', 'u'=>'user'), array('name'=>'p.title', 'u.username'));  
显示SQL语句为：SELECT `p`.`title` AS `name`, `u`.`username` FROM `content` AS `p`, `user` AS `u`

例六、  
对常用SQL函数的支持。

(1)、函数count()  
$model->from('user', array('total\_num'=>'count(`userid`)'));  
显示SQL语句为：SELECT count(`userid`) AS `total\_num` FROM `user`

(2)、函数avg()  
$model->from('user', array('avg\_num'=>'avg(`userid`)'));  
显示SQL语句为：SELECT avg(`userid`) AS `avg\_num` FROM `user`

(3)、函数sum()  
$model->from('user', array('num'=>'sum(`userid`)'));  
显示SQL语句为：SELECT sum(`userid`) AS `num` FROM `user`

(4)、函数max()  
$model->from('user', array('num'=>'max(`userid`)'));  
显示SQL语句为：SELECT max(`userid`) AS `num` FROM `user`

(5)、函数min()  
$model->from('user', array('num'=>'min(`userid`)'));  
显示SQL语句为：SELECT min(`userid`) AS `num` FROM `user`

(6)、函数distinct()  
$model->from('user', 'distinct `userid`');  
显示SQL语句为：SELECT distinct `userid` FROM `user`

2、where($where, $value) 用于处理SQL语句中where部分的语句。  
参数说明：  
$where:SQL语句的条件语句，参数非空。  
$value:SQL语句中?的值,参数默认为null。  
注：当SQL语句中有查询的字段值为字符或字符串时，不要忘记将字符或字符串进行转义。  
使用举例：

例一、  
$model->where('id=23');  
echo $model;  
显示SQL语句为：WHERE `id`=23

例二、  
$model->where('name=?', 'finecms');  
或  
$model->where('name=\'finecms\'');  
echo $model;  
显示SQL语句为：WHERE `name`='finecms'

例三、  
$data = $model->quote\_into('finecms');  
$model->where('name=' . $data);  
echo $model;  
显示SQL语句为：WHERE `name`='finecms'

例四、  
$model->where('id>6')->where('name=?', 'finecms');  
或  
$model->where(array('id>6', 'name=?'), 'finecms');  
或  
$model->where(array('id>6', 'name=\'finecms\''));  
echo $model;  
显示SQL语句为：WHERE `id`>6 AND `name`='finecms'

例五、  
$model->where('p.id>6')->where('u.name=?', 'finecms')->where('p.user\_id=u.user\_id');  
echo $model;  
显示SQL语句为：WHERE `p`.`id`>6 AND `u`.`name`='finecms' AND `p`.`user\_id`=u.user\_id

例六、  
$model->where('id<=100')->where('name like ?', 'finecms%');  
echo $model;  
显示SQL语句：WHERE `id`<=100 AND `name` LIKE 'finecms%'

例七、  
$model->where(array('name=?', 'group=?'), 'finecms');  
echo $model;  
显示SQL语句：WHERE `name`='finecms' AND `group`='finecms'

例八、  
$model->where('id between 10 and 20');  
echo $model;  
显示SQL语句为：WHERE `id` BETWEEN 10 AND 20

例九、  
$model->where('id in(10, 12, 15)');  
echo $model;  
显示SQL语句为：WHERE `id` IN(10, 12, 15)

例十、  
$model->where('match(name, email) against(?)', 'finecms');  
echo $model;  
显示SQL语句为：WHERE MATCH (name, email) AGAINST ('finecms')  
注：本例为全文搜索实例，mysql的MYISAM引擎才支持哦，INNODB引擎不支持全文搜索。

3、orwhere($where, $value) 用于处理SQL语句中orwhere部分的语句。  
参数说明：  
$where:SQL语句的条件语句，参数非空。  
$value:SQL语句中?的值,参数默认为null。  
注：当SQL语句中有查询的字段值为字符或字符串时，不要忘记将字符或字符串进行转义。  
使用举例：

例一、  
$model->orwhere('id>90')->orwhere('id<100');  
echo $model;  
显示SQL语句为：OR `id`>90 AND `id`<100  
注：本函数在使用上与where()使用函数完全相同，在本函数的使用上要参考下where()的使用方法。前面有详细的说明。

例二、  
$model->where('id>30')->where('id<40')->orwhere('id>90')->orwhere('id<100');  
echo $model;  
显示SQL语句为：WHERE `id`>30 AND `id`<40 OR `id`>90 AND `id`<100  
注：本函数要与where()一块用，即：在出现orwhre()地方一定要出现where()。  
不过有where()出现的地方，不一定要有orwhere()。  
Where()和orwhere()没有前后顺序限制。  
最后无论这两个函数哪一个最先出现，最后系统还是先处理where(),后出理orwhere()。

4、order($order\_desc) 完成SQL语句中order部分语句的处理。  
参数说明：  
$order\_desc是oder语句中排序一个单元，参数非空，支持数组。  
使用举例：

例一、  
$model->order('id asc');  
echo $model;  
显示SQL语句为：ORDER BY `id` ASC

例二、  
$model->order('p.id asc')->order('u.name desc');  
或  
$model->order(array('p.id asc', 'u.name desc'));  
echo $model;  
显示SQL语句为：ORDER BY `p`.`id` ASC , `u`.`name` DESC

5、join($table\_name, $where) 用于处理SQL中left join(左关联)语句的处理。  
参数说明：  
$table\_name为关联的数据表名。参数同前面所讲from()的$tabs参数略同，可以进行alias(更名)操作，不过本参数不支持多个数据表。  
$where为关联数据表的条件。  
使用举例：

例一、  
$model->from(array('p'=>'post'),'p.post\_title')->join(array('u'=>'user'), 'p.user\_id=`u`.`user\_id`');  
echo $model;  
显示SQL语句为：SELECT `p`.`post\_title` FROM `post` AS `p` LEFT JOIN `user` AS `u` ON `p`.`user\_id`=`u`.`user\_id`

6、group($group\_name)  
用于完成SQL中group by 语句处理。  
参数说明：  
$group\_name是分组名称。  
使用举例：

例一、  
$model->group('name');  
echo $model;  
显示SQL语句为：GROUP BY `name`

例二、  
$model->group('name')->group('class');  
echo $model;  
显示SQL语句为：GROUP BY `name`, `class`

例三、  
$model->group('p.name')->group('u.class');  
echo $model;  
显示SQL语句为：GROUP BY `p`.`name`, `u`.`class`

7、having($where, $value) 用于完成SQL语句中having部分的处理。  
注：本函数一般与group()同用。having的SQL语句用于“限制”group语句的，一般出现在group之后。  
参数说明：  
$where为条件语句，与where()中参数相同。可以参考where()来理解。  
$value为SQL语句中?的值,参数默认为null。  
使用举例：

例一、  
$model->group('name')->having('coun(`name`)>2');  
echo $model;  
显示SQL语句为：GROUP BY `name` HAVING coun(`name`)>2

例二、  
$model->group('name')->having('coun(`name`)>2')->having('coun(`name`)<6');  
echo $model;  
显示SQL语句为：GROUP BY `name` HAVING coun(`name`)>2 AND coun(`name`)<6

例三、  
$model->group('team')->having('team\_name=?', 'finecms');  
echo $model;  
显示SQL语句为：GROUP BY `team` HAVING team\_name='finecms'

8、orhaving($where, $value) 用于完成SQL语句中orhaving部分的处理。  
注：本函数一般与having()同用。  
参数说明：  
$where为条件语句，与where()中参数相同。可以参考having()来理解。  
$value为SQL语句中?的值,参数默认为null。  
使用举例：

例一、  
$model->group('name')->having('coun(`name`)>2')->orhaving('coun(`name`)<6');  
echo $model;  
显示SQL语句为：GROUP BY `name` HAVING coun(`name`)>2 or coun(`name`)<6

9、limit($num1, $num2) 用于完成SQL语句的limit部分的处理。  
参数说明：  
$num1, 即SQL中limit 后的第一个数字，用于定义查询显示的起始值。  
$num2, 即SQL中limit 后的第二个数字，用于定义查询显示多少行数据。默认为null即:本参数可以为空。  
使用举例：

例一、  
$model->limit(10);  
echo $model;  
显示SQL语句为：LIMIT 10

例二、  
$model->limit(10, 30);  
echo $model;  
显示SQL语句为：LIMIT 10, 30

10、page\_limit($page, $list\_num) 用于完成SQL语句的limit部分的处理，常用于有分页情况下的数据查询。  
参数说明：  
$page：当前的页数。  
$list\_num：每页所显示数据的行数。  
使用举例：

例一、  
$model->page\_limit(3, 10);  
echo $model;  
显示SQL语句为：LIMIT 20, 10

11、select($item) 用于完成from(),where()等函数所生成的SQL片段的组装，组装成一个完整理的SQL查询语句，并将其执行，且返回查询信息。  
参数说明：  
$item为返回数据的选项，当为true时，则select返回的数据是多维组数组，即返回的为SQL查询语句的全部数据。当为false时，为select查询后第一条数据。默认为true。  
使用举例：

例一、  
$data = $model->from('centent')->where('id>27')->order('id desc')->limit(10, 20)->select();  
print\_r($data);  
显示多行数据(二维数组)。

例二、  
$data = $model->from('centent')->where('id>27')->order('id desc')->limit(10, 20)->select(false);  
print\_r($data);  
显示单行数据(一维数组)。

12、quote\_into($value) 本函数用来完成字符串的转义的。  
防止SQL注入，提高程序安全。辅助于where(), orwhere(), having()这三个函数。  
参数说明：  
$value：字符串内容。本参数支持数组。  
使用举例：

例一、  
$m = $model->quote\_into(‘wang's book’);  
echo $where;  
显示SQL语句为：'wang\'s book'。(函数运行原理极为简，相当于PHP的addslashes()函数操作)。

例二、  
$name = $model->quote\_into('finecms');  
$model->where('name='.$name);  
echo $model;  
显示SQL语句为：WHERE `name`='finecms'

14、主键查询

(1)、find($id, $fields = null) 用于查找并返回主键为$id的数据。  
(注：此主键并非数据表的PRIMARY\_KEY，这个主键是相对的，如果没有自定义，则默认为数据表的PRIMARY\_KEY。)  
参数说明：  
$id为主键的ID，由于这个主键可以自定义，所以此参数也可是字符串。当然也可是数组。参数为数组时，相当于SQL语句中 WHERE IN(参数内容)。  
$fields：所要查询的数据表字段.默认为全部数据表字段  
使用举例：

例一、  
$data = $model->find(23);

例二、  
$data = $model->find(array(21,23,27));  
注：本例参数为数组型，返回的数据为多维数组。

例三、  
$model->primary\_key = 'title';  
$data = $model->find('\'my god\'');  
注：本例为主键自定义实例。自定义操作：在本函数调用之前$model->primary\_key = '自定义的主键'；如果参数为字符串，别忘了对参数进行转义(使用quote\_into()函数或参数中用\'这种方式)。

例四、  
$data = $model->find(21, array('name', 'age', 'addr'));

例五、  
$data = $model->find(array(21,23,27), array('name', 'age'));

(2)、findAll($fields = null, $order\_asc = true，$offset = null, $count = null)  
以数据表主键为顺序查询并返回数据表内所有数据。本函数获取数据表全部数据时只适用于数据表数据量不大时。如果数据表的数据是千万级1TB以上。数据全返回，程序运行消耗时间不说，单单服务器内存的占用也是吃不消的，所以在数据表数据量不大时用于调试程序所用(对开发者建议)，要么$offset,$count参数赋上值则相当于SQL中的limit $offset, $count。本函数与find()一样，也支持order及主键的自定义，操作函数与find()相同。对此如不明白，可以参考find()的用法。  
参数说明：  
$fields：所要查询的数据表字段.默认为全部数据表字段  
$order\_asc：数据排序,若为true时为ASC,为false时为DESC, 默认为ASC  
$offset: limit 启起ID  
$count: limit 显示的行数  
使用举例：

例一、  
$data = $model->findALL();

例二、  
$model->primary\_key = 'title';  
$data = $model->findALL();

例三、  
$data = $model->findALL(array('name', 'tel'), false);

例四、  
$data = $model->findALL('name', false);

例五、  
$data = $model->findALL(array('name', 'tel'), false, 5, 10);

例六、  
$data = $model->findALL(array('name', 'tel'), false, 10);

15、根据一个查询条件，获取数据

(1)、getOne($where, $value = null, $fields = null) 查询并获取满足$where条件的第一行数据。  
参数说明：  
$where:查询条件，参数类似于where()函数。如果参数里有字符串，不要忘记对其进行转义。参数非空，支持数组。  
$value:默认为null, 是查询条件中?的替换数据。具体使用，请参考前面所讲的where()的使用方法  
$fields：所要查询的数据表字段.默认为全部数据表字段。注：当本参数为true时，本方法刚返回当前数据表的主键值  
使用举例：

例一、  
$data = $model->getOne('title=\'my god\'');  
或  
$data = $model->getOne('title=?', 'my god');

例二、  
$where = $model->quote\_into( 'my god');  
$data = $model->getOne('title='.$where);

例三、  
$data = $model->getOne(array('id'>13, 'name=?'), 'finecms');

例四、  
$data = $model->getOne('name=?', 'finecms', array('info', 'group'));

例五、  
$data = $model->getOne('name=?', 'finecms', 'info');

(2)、getAll($where, $value=null, $fields = null, $order = null, $offset = null, $count = null)  
查询并获取满足$where条件的所有数据。  
参数说明：  
$where是查询条件，参数类似于where()函数的。如果参数里有字符串，不要忘记对其进行转义。参数非空，支持数组  
$value是查询条件中?的替换数据，默认为null。  
$fields：所要查询的数据表字段.默认为全部数据表字段  
$order是数据显示的排序条件，默认为null。  
$offset: limit 启起ID  
$count: limit 显示的行数  
使用举例：

例一、  
$data = $model->getAll('title=?', 'my god');  
或  
$data = $model->getAll('title=\'my god\'');

例二、  
$where = $model->quote\_into('my god');  
$data = $model->getAll('title='.$where);

例三、  
$data = $model->getAll(array('post\_id>23', 'title=?'), 'finecms', null, 'post desc');

例四、  
$data = $model->getAll(array('post\_id>23', 'title=?'), 'finecms', array('desc', 'author'));

例五、  
$data = $model->getAll(array('post\_id>23', 'title=?'), 'finecms', null, 'post desc', 5, 10);

例六、  
$data = $model->getAll(array('post\_id>23', 'title=?'), 'finecms', null, 'post desc', 10);

(3)、count($table\_name, $field\_name = null, $where = null, $value = null)  
获取某数据表满足一定条件的数据的总行数.注：分页程序常用  
参数说明：  
$table\_name:所要查询的数据表名  
$field\_name:所要查询字段名称,当参数为null时，系统默认为当前数据表的主键  
$where :查询条件  
$value :数值（数据表某字段所对应的数据，通常都为字符串或字符）

使用举例：

例一、  
$total\_num = $model->count('centent', 'id', 'id>1000');

例二、  
$total\_num = $model->count('centent', 'id', 'username=?', 'finecms');

16、对数据表的UID操作(update、insert、delete)

(1)、update($data, $where, $params = null)  
对model所对应的数据表进行更改操作。本函数均对所要update的数据进行了addslashes()转义，有效防止SQL注入。  
参数说明：  
$data为所要更改的内容，为数组型。参数数组中数组索引为数据表字段，数组的值为所要写入的内容。如果想更改一行数据，字段1为a, 字段2为b，那参数应为：array('字段1'=>'a', '字段2'=>'b');当参数数组有索引不是数据表中的字段时，程序执行时则对其进行过滤掉，不加利用。  
$where为满足的条件。参数格式与whrer()，getOnet()等函数的参数相同。  
$params为替换$where里?的替换数据,默认为null  
使用举例：

例一、  
$update\_array = array('title'=>'DEMO', 'content'=>'IT IS A DEMO');  
$model->update($update\_array, 'id=5');

例二、  
$update\_array = array('title'=>'DEMO', 'content'=>'IT IS A DEMO');  
$model->update($update\_array, 'name=?', 'dayrui');

例三、  
$update\_array = array('hits'=>'hits+1');  
$model->update($update\_array, 'name=?', 'dayrui');  
hits字段值加1。

(2)、insert($data)  
向model所对应的数据表插入一条数据信息。本函数均对所要insert的数据进行了addslashes()转义，有效防止SQL注入。  
参数说明：  
$data为所要写入的数据，数组型。参数数组中数组索引为数据表字段，数组的值为所要写入的内容。  
如果想写入一行数据，字段1为a, 字段2为b，那参数应为：array('字段1'=>'a', '字段2'=>'b');  
返回自增ID值。  
使用举例：

例一、  
$insert\_array = array('title'=>'DEMO', 'content'=>'IT IS A DEMO');  
$insert\_id = $model->insert($insert\_array);

(3)、delete($where, $value = null)  
从model所对应的数据表中删除满足参数条件的数据。  
参数说明：  
$where为删除数据的条件，参数格式，参考where()的参数。同样如果参数中有字符，不应忘记对其转义。  
$value为删除数据的条件中?的替换数据，默认为null。  
使用举例：

例一、  
$model->delete('id=5');

例二、  
$where = $model->quote\_into('dayrui');  
$model->delete('name='.$where);

例三、  
$model->delete('name=?', 'dayrui');

17、execute($sql, $all\_rows = true)  
执行一个SQL查询语句，并返回查询的数据。如果你认为前面那些对数据库进行操作的函数不够灵活，满足不了需要，这个函数至少弥补了这点不足。使用SQL语句操作数据表，灵活机动，随心所欲。  
参数说明：  
$sql为所要执行的SQL语句。注：参数只支持select的查询语句，其它的操作，函数会报错。  
$all\_rolws：返回的数据是否为全部数据，当为true时返回全部数据，当为false时，返回第一行数据，默认为true。  
注:本函数对于常用的SQL语言进行判断,如果以select开头的SQL语句,则返回查询的数据,对于update,insert,delete等则执行,仅返回执行结果,但对于show tables、desc tablename之类的查询语句,不能智能判断出，所以需要第三个参数。当第三个参数为true时,本函数将按查询语句执行。  
使用举例：

例一、  
$sql = "select u.\* from user as u, content as p where u.userid=p.userid and p.id<15 order by p.id desc limit 0, 10";  
$data = $model->execute($sql);  
返回多维数组。

例二、  
$sql = "select u.\* from user as u, content as p where u.userid=p.userid and p.id<15 order by p.id desc limit 0, 10";  
$data = $model->execute($sql, false);  
返回一条数据。

例三、  
$sql = "update content set title='demo' where id=5";  
$model->execute($sql);

18、get\_table\_prefix() 获取数据表前缀。  
使用举例：  
$table\_prefix = $this->get\_table\_prefix();

19、query($sql) 执行一条sql语句。  
使用举例：  
$this->query('update user set status=1 where userid=1');

## FineCMS的View应用

视图(view)是MVC中的V。作用是完成视图文件的处理并将页面内容显示出来。

**一、FineCMS视图机制的运行原理**  
首先需要在controller文件中实现化Views(视图基类)。  
通过视图基类中的assign()对视图变量完成赋值，然后由display()完成视图文件的加载，编译和显示。  
当display()运行时，首先分析视图文件的编译文件，当编译文件需要重新生成时则加载视图文件，将视图文件中的“模板标签”转换为PHP代码。  
将处理后的内容写入编译文件。如编译文件无需重新生成时，则直接加载编译文件并显示其内容。

**二、视图基类的模板标签说明**  
FineCMS的视图标签详细情况请参考上面的标签教程

1、模板变量。  
{$模板变量}  
{$模板变量['索引']}  
例：  
{$sitename}  
{$lines['name']}

2、判断语句的模板标签。  
{if $模板变量1 运算符 $模板变量}  
模板内容1  
{else}  
模板内容2  
{/if}  
或  
{if $模板变量1 运算符 $模板变量2}  
模板内容1  
{else if $模板变量1 运算符 $模板变量3}  
模板内容2  
{else}  
模板内容3  
{/if}  
注：这里的运算符支持：> 、< 、>= 、<=、==、！= 、<>。  
例一、  
{if $x>$y}  
x大于y  
{else}  
x不大于y  
{/if}

例二、  
{if $x>$y}  
x大于y  
{else if $x==$y}  
x等于y  
{else}  
X小于y  
{/if}

例三、  
{if $x >300}  
x大于300  
{else if $x>200}  
X大于200  
{else}  
x小于等于200  
{/if}

例四、  
{if $x == true}  
x为true  
{else}  
x为false  
{/if}

例五、  
{if $x == false}  
x为false  
{else}  
x为true  
{/if}

例六、  
{if $data['num'] == $total\_num}或{if $data.num == $total\_num}  
数量为 {$total\_num}  
{else}  
数量不为{$total\_num}  
{/if}

例七、  
{if $data['num'] == 200}或{if $data.num == 200}  
数量为200  
{else}  
数量不为200  
{/if}

例八、  
{if $data['num'] > 200}或{if $data.num > 200}  
数量大于200  
{else if $data['num'] > 100}  
数量大于100  
{else}  
数量不大于100  
{/if}

例九、  
{if $data['state']== true}  
state开启  
{else}  
state未开启  
{/if}

例十、  
{if $data['menu']== 'index'}  
首页  
{else}  
其它  
{/if}

3、循环语句的模板标签。  
{loop 模板数组 $索引变量=>$值变量}  
执行语句  
{/loop}  
或  
{loop 模板数组 $值变量}  
执行语句  
{/loop}

例一、  
{loop $data $key=>$value}  
{$key}:{$value}  
{/loop}

例二、  
{loop $data $value}  
n+1次  
{/loop}

4、template/include模板标签。  
{template模板文件} 或者 {include模板文件}  
注：此模板文件一般为共享性模板文件。如：每个页面的版信息。如果将版权信息这部分单独自作一个独立模板，让每个页面通过include的方式进行共享。

例一、  
{template common/copyright } 或者 {include common/copyright }  
注：common为系统目录views中的子目录，copyright为文件名。模板文件这里省略了.html的后缀名。

例二、  
{template copyright } 或者 {include copyright }  
注：这里的模板名称只有copyright，没有上面例一中的common目录。  
当模板文件名不含模板所在的目录时，系统默认为这个页面。

## FineCMS的Controller应用

Controller是MVC中的C，即控制程序，是M和V的纽带

**一、文件命名**  
控制器名称：控制器名称（首字母大写）+ Controller.php，如TestController.php  
控制器方法：方法名称（小写）+ Action，如indexAction()

**二、程序格式**

<?php  
class TestController extends Common { //继承公共控制器类  
  public function \_\_construct() {  
   parent::\_\_construct();  
  }  
  
  public function indexAction() { //控制器方法index  
   $id = $this->get('id');//获取GET['id']值      
   $category = $this->model("category"); //实例化栏目表category对象，可以使用Model类中的方法调用数据   
   $title = 'hello finecms~~';//普通变量  
   $this->view->assign('title', $title);//将变量赋值给模板  
   $this->view->display('test');//加载模板，test表示test.html  
  }  
}

**三、访问地址**  
格式：网站地址/index.php?s=空间&c=控制器名称(小写)&a=控制器方法(小写)  
例如：http://localhost/index.php?c=test&a=index

**四、常用方法介绍**  
1、get($name)  
获取全局变量$\_GET的$name的值，并进行trim()及htmlsepcialchars()处理  
参数说明：  
$name: 与$\_GET的参数相同  
举例说明：  
$name = $this->get('name')   
2、post($name)  
获取全局变量$\_POST的$name的值，并进行trim()及htmlsepcialchars()处理，支持多维数组  
参数说明：  
$name: 与$\_POST的参数相同  
举例说明：  
$name = $this->post('name')

3、isPostForm($var='submit', $emp=0)  
验证是否来自表单POST提交。  
参数说明：  
$var: 表单提交按钮的名称，默认为submit  
$emp: 提交按钮的value值，默认为0  
举例说明：  
if ($this->isPostForm) {  
echo '是post提交';  
}

4、get\_server\_name()  
获取页面网址中的域名  
使用举例：echo $this->get\_server\_name()。//http://localhost/

5、get\_base\_url()  
获取当前项目的根目录的URL。  
使用举例：$this->get\_base\_url(); // /cms/

6、get\_theme\_url()  
当前项目的风格目录。  
使用举例：$this->get\_theme\_url(); // /views/default/

7、redirect($url)  
网页的url跳转。  
参数说明：$url地址。  
使用举例：$this->redirect('http://www.xx.com');

8、create\_url($route, $params = null)  
组装URL路由功能的网址，可直接使用url()函数  
参数说明：  
$route为namespace、controll及页面action的组合。  
$params为网址中的参数部分。此参数为数组。索引为参数名，值为参数值。此参数可为空。  
使用举例：  
$this->create\_url('post/view', array('id'=>35)); //URL:/index.php?c=post&a=view&id=35  
$this->create\_url('admin/post/view', array('id'=>35)); //URL:/index.php?s=admn&c=post&a=view&id=35  
推荐使用全局函数url($route, $params = null),第三参数为兼容旧版本无实际意义，开发中只需前2个参数即可

## FineCMS插件开发指南

**一、插件类型介绍**  
FineCMS插件目前分为“内置控制器”、“页面嵌入”。  
1、内置控制器类型插件。  
有独立的控制器、模型、视图目录，后面将重点介绍该类型插件开发。  
2、页面嵌入类型插件。  
该类型插件只包含两个核心文件：run.php和run.html。

**二、“内置控制器”插件开发介绍**

1、插件文件及目录结构  
目录controllers：     插件控制器目录  
目录models：          插件模型目录  
目录views：             插件视图目录  
文件config.php：     插件配置信息文件  
文件install.php：      安装插件执行的SQL语句  
文件uninstall.php：删除插件执行的SQL语句  
文件md5.php：        MD5校验文件，官方校验文件  
文件mark.txt：         官方发布插件的唯一标示，若是自己开发的插件就不需要该文件

2、插件访问路径  
网站地址/index.php?s=插件名称&c=插件控制器&a=对应的Action。

3、插件配置文件config.php规则

<?php  
return array(  
'name' => '文章踩顶',  
'author' => 'start',  
'version' => '1.0',  
'typeid' => 1,  
'description' => "必须有Jquery支持，将下面代码放在内容页面中",  
'fields' => array(  
array('field'=>'dingname', 'name'=>'“顶”名称设置', 'tips'=>'默认“顶一下”。', 'formtype'=>'input', 'setting'=>"array('size'=>'100')"),  
array('field'=>'cainame', 'name'=>'“踩”名称设置', 'tips'=>'默认“踩一下”。', 'formtype'=>'input', 'setting'=>"array('size'=>'100')"),  
)  
);

该配置文件返回数组格式信息，参数如下介绍。  
name：          插件名称。  
author：        插件作者。  
version：       插件版本。  
typeid：         1，（固定格式值）。  
description： 插件描述信息（注意html转义）。  
fields：          插件配置参数（若没有配置信息设置，该值可省略）。  
array('field'=>'字段名称', 'name'=>'字段显示名称', 'tips'=>'提醒文字', 'formtype'=>'字段类型（可参考扩展目录的fields.php文件）', 'setting'=>"该字段类型的配置信息"),

4、插件安装/卸载文件规则  
(1)、安装文件install.php

<?php  
if (!defined('IN\_FINECMS')) exit('');  
return array(  
"DROP TABLE IF EXISTS `{prefix}digg`;",  
"CREATE TABLE IF NOT EXISTS `{prefix}digg` (  
`id` mediumint(8) NOT NULL AUTO\_INCREMENT,  
`contentid` mediumint(8) NOT NULL,  
`title` varchar(100) NOT NULL,  
`cai` mediumint(8) NOT NULL,  
`ding` mediumint(8) NOT NULL,  
`addtime` int(10) NOT NULL,  
PRIMARY KEY (`id`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8;"  
);

支持多条SQL语句，数据表前缀请用“{prefix}”表示。

(2)插件卸载uninstall.php

<?php  
if (!defined('IN\_FINECMS')) exit('');  
return array(  
"DROP TABLE IF EXISTS `{prefix}digg`;"  
);

5、插件控制器controllers规则  
(1)、插件公共控制器类Common.php。

<?php  
/\*\*  
\* 插件控制器公共类  
\*/  
class Plugin extends Common { //继承通用公共类  
  
protected $plugin; //插件模型  
protected $data; //插件数据  
protected $viewpath; //视图目录  
protected $digg; //用到的插件模型实例  
  
public function \_\_construct() {  
parent::\_\_construct();  
$this->plugin = $this->model('plugin');  
$this->data = $this->plugin->where('dir=?', $this->namespace)->select(false);  
if (empty($this->data)) $this->adminMsg('插件尚未安装', url('admin/plugin'));  
if ($this->data['disable']) $this->adminMsg('插件尚未开启', url('admin/plugin'));  
$this->viewpath = SITE\_PATH . $this->site['PLUGIN\_DIR'] . '/' . $this->data['dir'] . '/views/';  
$this->assign(array(  
'viewpath' => $this->viewpath,  
'pluginid' => $this->data['pluginid'],   
));  
//以上代码为固定格式，用户在开发插件时，可以直接copy。  
$this->digg = $this->model('digg'); //实例化用到的模型  
}  
  
}

插件控制器公共类用于实例化该插件用到的模型，或者该插件用到的函数等。  
(2)、控制器Controller介绍

<?php  
class AdminController extends Plugin { //AdminController继承公共插件类  
  
public function \_\_construct() {  
parent::\_\_construct();  
//Admin控制器进行登录验证  
if (!$this->session->is\_set('user\_id') || !$this->session->get('user\_id')) $this->adminMsg('请登录以后再操作', url('admin/login')); //如果需要管理员登陆验证，就加上这一句。  
}  
  
public function indexAction() {  
echo "插件indexAction，欢迎使用FineCMS插件。";  
}  
}

加载视图文件和赋值变量与主控制器不同以外，其他规则与Controller一致。  
变量赋值：$this->assign(变量名称, 变量值); $this->assign(支持数组);  
加载视图：$this->display(视图文件名称（不带扩展名）);例如：$htis->display('test');//插件目录/views/test.php

(3)、插件模型models介绍  
与系统模型规则一致。

(4)、插件视图views介绍  
只支持原生态PHP语句，不支持模板标签。

## 自定义字段类别

自定义字段的相关函数一般放在/extensions/custom.php文件中，因为是自定义的，此文件不影响升级，大家都懂

**一、函数介绍**  
1、字段类别函数：\_formtype()，格式array('字段英文名字'=>'字段提示名称')  
2、字段配置函数：form\_字段英文名字($setting=null)  
3、表单输出函数：content\_字段英文名字($name, $content=null, $field=null)

**二、举例说明**  
1、建立一个文本框字段（名称为test）  
①、字段类别函数

function \_formtype() {  
  return array(  
   'test' => '测试', //前者test表示字段的英文名称，后者表示字段的描述名称  
  );  
}

②、字段配置函数

function form\_test($setting=null) {  
$size = isset($setting['size']) ? $setting['size'] : 150; //如果不填写长度，默认为150  
$default = isset($setting['default']) ? $setting['default'] : ''; //默认值，都能看懂  
return '  
<table width="98%" cellspacing="1" cellpadding="2">  
<tbody>  
<tr>  
<td width="100">文本框长度 ：</td>  
<td><input type="text" class="input-text" size="10" value="' . $size . '" name="setting[size]"><font color="gray">px</font></td>  
</tr>  
<tr>  
<td>默认值 ：</td>  
<td><input type="text" class="input-text" size="30" value="' . $default . '" name="setting[default]"></td>  
</tr>  
</tbody>  
</table>';  
}

③、表单输出函数

function content\_test($name, $content=null, $field=null) {  
$setting = isset($field['setting']) ? string2array($field['setting']) : $field; //配置  
$required = isset($field['not\_null']) && $field['not\_null'] ? ' required' : ''; //必填字段  
//$errortips = isset($field['errortips']) && $field['errortips'] ? ' placeholder="' . $field['errortips'] . '"' : ''; //错误提示  
$pattern = isset($field['pattern']) && $field['pattern'] ? ' pattern="' . $field['pattern'] . '"' : ''; //正则判断  
$content = is\_null($content[0]) ? get\_content\_value($setting['default']) : $content[0]; //字段内容，固定值  
return '<input type="text" value="' . $content . '" class="input-text" name="data[' . $name . ']" width=' . $setting['size']. '>'; //表单Name值格式为data[字段名称]  
}

2、建立一个读取会员组表中的数据（名字为membergroup）  
①、字段类别函数

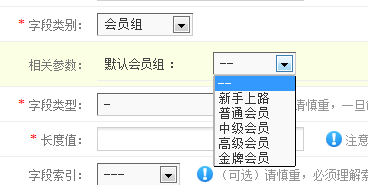
function \_formtype() {  
return array(  
'test' => '测试', //前者test表示字段的英文名称，后者表示字段的描述名称  
'membergroup' => '会员组', //前者membergroup表示字段的英文名称，后者表示字段的描述名称  
);  
}

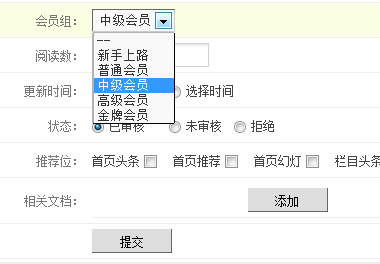
②、字段配置函数

function form\_membergroup($setting=null) {  
$default = isset($setting['default']) ? $setting['default'] : ''; //默认值，都能看懂  
//调用会员组表中的数据  
$model = Controller::model('member\_group');//实例化会员组模型  
$data = $model->getAll(1);//获取会员组表中的全部数据，用于设置默认值  
$select = '<select name="setting[default]"><option> -- </option>';  
foreach ($data as $t) {  
$selected = $default == $t['id'] ? 'selected' : '';  
$select .= '<option value="' . $t['id'] . '" ' . $selected . '> ' . $t['name'] . ' </option>';  
}  
$select .= '</select>';  
return '  
<table width="98%" cellspacing="1" cellpadding="2">  
<tbody>  
<tr>  
<td width="100">默认会员组 ：</td>  
<td>' . $select . '</td>  
</tr>  
</tbody>  
</table>';  
}

③、表单输出函数

function content\_membergroup($name, $content=null, $field=null) {  
$setting = isset($field['setting']) ? string2array($field['setting']) : $field; //配置  
$required = isset($field['not\_null']) && $field['not\_null'] ? ' required' : ''; //必填字段  
//$errortips = isset($field['errortips']) && $field['errortips'] ? ' placeholder="' . $field['errortips'] . '"' : ''; //错误提示  
$pattern = isset($field['pattern']) && $field['pattern'] ? ' pattern="' . $field['pattern'] . '"' : ''; //正则判断  
$content = is\_null($content[0]) ? get\_content\_value($setting['default']) : $content[0]; //字段内容，固定值  
//调用会员组表中的数据  
$model = Controller::model('member\_group');//实例化会员组模型  
$data = $model->getAll(1);//获取会员组表中的全部数据，用于设置默认值  
$select = '<select name="data[' . $name . ']"><option> -- </option>';//表单Name值格式为data[字段名称]  
foreach ($data as $t) {  
$selected = $content == $t['id'] ? 'selected' : '';  
$select .= '<option value="' . $t['id'] . '" ' . $selected . '> ' . $t['name'] . ' </option>';  
}  
$select .= '</select>';  
return $select;  
}

④、效果预览  
添加字段时：  


发布内容时：   


就像示例2，自定义字段类别支持表数据显示，当然也能自动将数据插入表单，完全支持自定义，只要你想要FineCMS就能帮你实现

## 自定义编辑器

此功能其实就与上一节的自定义字段类别一样

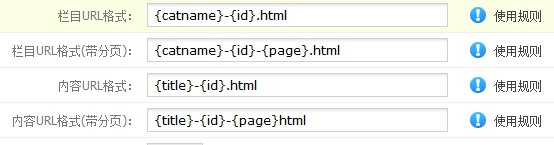
函数名称格式：编辑器名称\_editor($name, $content=null, $field=null)

举例，定义一“test”编辑器：  
function test\_editor($name, $content=null, $field=null) {  
............程序结构可以参考Kindeditor与ueditor  
}  
  
自定义编辑器的相关函数一般放在/extensions/custom.php文件中，因为是自定义的，此文件不影响升级，大家都懂

## 个性化URL结构

**一、支持主表字段（如果需要附表字段可以使用方法“三”）**

要使用个性化URL结构，必须先阅读上面“数据结构”章节

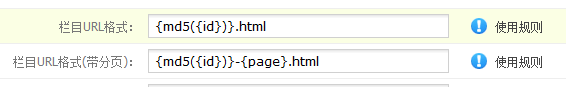


{catname}：示栏目名称，看过“数据结构”的都懂，不解释  
{title}:是文章的id

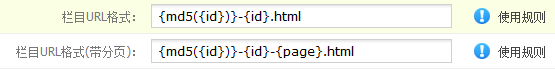
1、静态生成模式  
静态生成模式，不用加{id}标签，但要确保url地址的唯一（可以加其他参数来区分，如时间）  
2、伪静态模式  
伪静态模式，栏目必须有{id}或者{dir}/{pdir}，内容必须加{id}标签，前面已经说的很清楚了  
  


如果你的服务器不支持多编码，可能会出现中文乱码，慎用

**二、运用普通函数（以md5函数为例子）**

1、静态生成模式  


{md5({id})}.html : {id}表示调用当前的id，{md5(xxx)}表示进行md5运行，因为是静态生成模式，所有可以不用再次加id了

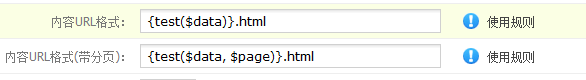
2、伪静态模式  


{md5({id})}-{id}.html : 前面一节跟上面一样，后面{id}表示调用当前的id，因为是伪静态嘛，必须与服务器的id匹配运算，所有必须加一个id  
{word2pinyin({title})}-{id}.html : 拼音标题+id.html（需要用的可以参考，不支持特殊汉字）

那么伪静态代码如下：

RewriteRule ^([a-zA-Z0-9]{32})-([0-9]+).html$ index.php?c=content&a=list&catid=$2 [L] (md5是32位的所有前面写了{32})

**  
  
三、运用自定义函数**

例子：文章标题的拼音.html（我们以静态为例子吧，伪静态时记得跟上面例子一样必须加一个id）  
  
从上面不难看出，我们是调用了自定义函数test()，那么我们就定义一个test的函数（自定义函数应该放在专用文件里/trunk/extensions/custom.php，不会影响系统升级）

function test($data, $page=1) {  
    if ($page > 1) { //当带有分页时  
        //word2pinyin函数是系统自带的拼音转换函数，特殊字不支持哦~  
        return word2pinyin($data['title']) . '-' . $page; //上面规则中写了.html为后缀，在这里就不用写了  
    } else { //当不带分页时  
        return word2pinyin($data['title']); //上面规则中写了.html为后缀，在这里就不用写了  
    }  
}  
//其实这种URL格式方法二完全可以实现的



补充： 上面函数中的$data数组是指当前文章表（不包括自定义字段，若要使用自定义字段必须查询）、栏目表、表单（包括自定义字段）的数据

## 表单模型和会员扩展模型中的回调函数用法

**一、回调函数的作用**

默认情况下，我们在提交之后会有如下提示

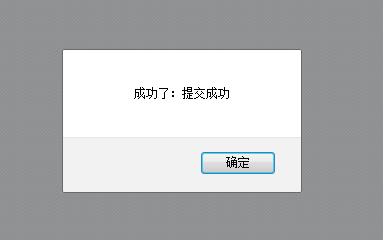


回调函数就是用来自定义这种提示的功能

**二、使用回调函数**

1、在后台设置中设置“回调函数”  
  
在上面的配置中，我们设置了“mycallback”作为回调函数，下面的“回调函数格式”不用管它，它只是一个格式举例  
  
2、在自定义函数文件中创建一个叫“mycallback”的函数(要跟你填写的名字一样)

function mycallback($msg, $url='', $state=0) { //参数在上面有介绍  
    if ($state = 1) { //表示成功  
        echo "<script>alert('成功了：" . $msg . "')</script>";  
    } else { //表示失败  
        echo "<script>alert('失败了：" . $msg . "')</script>";   
    }  
}  
//运用$url参数，你可以设置跳转，这里不举例了，



示例只有了js提示，当然你也可以使用更多JS特效来运用回调函数

## 数据读写分离配置

在v1.7.7版本 程序里面内置了读写分离的功能，方便了一些负载比较大的站点使用来降低一定的负载。

**配置方法**

配置文件：/config/database.slave.ini.php

    /\*\*  
     \* 数据库读写分离配置  
     \* 支持多个从库，格式如：1=>array(), 2=>array() ...   
     \* 从库的数据库名、前缀、编码必须与主库保持一致  
     \*/  
    return array(  
        '1' => array(  
            'host' => '127.0.0.1',  
            'username' => 'root',  
            'password' => '',  
            'port' => '3307'  
        ),  
    );

[MySQL服务器的主从配置步骤](http://bbs.finecms.net/forum.php?mod=viewthread&tid=1213&extra=page%3D1)

更多内容参考官方发布的手册http://www.dayrui.net/help/