



ThinkPHP Framework 1.5 FAQs

ThinkPHP 1.5 常见问题集合



编写：ThinkPHP 文档组

最后更新：2008-12-24

目录

1	概述.....	3
2	基础知识.....	3
3	控制器	8
4	模型相关.....	14
5	配置相关.....	19
6	模板引擎.....	20
7	错误和调试.....	23
8	其他.....	24

1 概述

本文档描述了新手在使用 ThinkPHP 的学习过程中比较关注的一些疑问，或者容易出现的一些问题，并尽可能给出官方的合理答案（该文档会不断补充和完善）。

2 基础知识



ThinkPHP 是什么



ThinkPHP 是一个基于 MVC 模式的面向对象的 PHP 开发框架，基于 Apache2 开源协议发布，是一个基于 PHP 轻量级的 PHP 开发框架，提供 WEB 应用开发的快速解决方案。ThinkPHP 不是博客系统，也不是 CMS 系统，但是可以开发出任何类似博客或者 CMS 系统的应用出来。



ThinkPHP 是开源的吗



ThinkPHP 基于 Apache2 开源协议(<http://www.apache.org/licenses/LICENSE-2.0>)发布，并且永久免费下载和使用的轻量级 PHP 开发框架。



ThinkPHP 有 SVN 地址吗



ThinkPHP 有在 Google 项目申请注册，SVN 地址：

完整版本<http://thinkphp.googlecode.com/svn/trunk>

核心版本<http://thinkphp.googlecode.com/svn/trunk/ThinkPHP>




ThinkPHP 支持的 PHP 版本是多少




ThinkPHP1.5 版本需要 PHP5（建议 5.2.0 以上版本）版本的支持，但是同时我们还发布了一个针对 PHP4 的 0.9.9 版本。


ThinkPHP 有额外的环境要求么

 ThinkPHP 对服务器和操作系统环境没有太多要求 ,经过我们的测试在 Apache、IIS ,甚至在 Nginx 下面都可以运行。并且核心框架没有依赖任何 PHP 的其他模块 ,只有在应用的时候才需要根据自己的需求来考虑是否需要额外的环境要求。


ThinkPHP 的目的和宗旨是什么

 ThinkPHP 的目的是为了简化企业级的应用开发和敏捷 WEB 开发 ,宗旨是**简洁、快速和实用**。


ThinkPHP 和其他框架比较有什么特色

 ThinkPHP 的主要特色是官方团队花费三年所打造的为用户考虑的众多细节 ,包括架构、功能和使用方面的一系列特点。不但融会了众多不同语言和领域的优秀框架的思想下 ,也给出了自己独有的创新和用户实践。其类库导入、项目编译、视图模型、ORM 实现、动态查询、分布式和多数据库支持、静态缓存、配置文件、缓存机制、模型自动验证和自动完成、空模块和空操作、权限认证、URL 模式等功能较国内外的框架有明显的不同和创新 ,内置了独立开发的基于 XML 和标签库的 PHP 模板引擎作为更是同类框架的首创。另外的优势就是本地化的文档优势 ,我们有包括 PDF、CHM 和 SWF 格式超过 50 份的完整文档 ,以及及时更新的在线手册。概括而言 ,ThinkPHP 特色包括 :**功能全面、独具创新、文档齐全、快速开发**。

ThinkPHP 里面 MVC 分别是对应哪些

 在 ThinkPHP 里面 ,M 是指模型类 Model V 是指模板文件 C 主要是指 Action 控制器。之所以说是主要 ,是因为有一些额外的控制操作是在核心控制器 App 和 Dispatch 处理方面的 ,严格来说 ,他们也属于 C 的范畴。

什么是 CURD

 CRUD 是一个数据库技术中的缩写词，一般的项目开发的各种参数的基本功能都是 CURD。

它代表创建（ Create ） 读取（ Read ） 更新（ Update ） 和删除（ Delete ） 操作。CRUD 定义了用于处理数据的基本原子操作。 .

什么是单一入口

 传统的模式下面 ,当 WEB 服务器收到一个 http 请求时 ,会解析该请求以确定要访问哪一个文件。

例如 `http://www.xxx.com/news.php` 的解析结果就是要求 web 服务器解析 `news.php` 文件 ,并返回结果给浏览器。而单一入口则是无论什么功能操作都请求同一个 `index.php` 文件 ,然后根据 url 参数进行了第二次解析 ,以确定要访问的文件和操作 ,而 `index.php` 通常被称为入口文件。注意 ,单一入口并不代表网站是唯一入口的。

ThinkPHP 的目录结构包括哪些

 ThinkPHP 的系统目录结构如下 :

- └─ThinkPHP 框架系统目录
 - | └ ThinkPHP.php 系统公共文件
 - | └ Common 公共文件目录
 - | └ Tpl 框架系统模版目录
 - | └ Lang 系统语言包目录
 - | └ PlugIns 公共插件目录
 - | └ Vendor 第三方类库目录
 - | └ Lib 系统基类库目录

- | └ Think 系统运行库（必须）
- | └ Com 扩展类库包（非必须）
- | └ ORG 扩展类库包（非必须）

项目目录结构如下：

- |—App App 项目目录
- | └ index.php 项目入口文件（位置可选）
- | └ Cache 模版缓存目录
- | └ Common 公共文件目录（非必须）
- | └ Conf 项目配置目录
- | └ Data 项目数据目录
- | └ Html 静态文件目录（非必须）
- | └ PlugIns 插件目录（非必须）
- | └ Tpl 模版文件目录
- | └ Lang 语言包目录（非必须）
- | └ Logs 日志文件目录
- | └ Temp 数据缓存目录
- | └ Lib 应用类库目录
 - | └ Action 控制器（模块）类目录
 - | └ Model 模型类目录
 - | ... 下面的应用目录可根据需要选择和定义
 - | └ Exception 异常类库目录
 - | └ Common 公共应用类目录



什么是系统基类库



基类库位于框架系统目录下面的 Lib 目录，这些类库除了框架运行所需要的核心类库，还包括网站和项目开发过程中经常会用到的常用工具类。目前主要包含 Think 核心类库、ORG 扩展类库、Com 扩展类库，其中 Think 核心基类库的作用是完成框架的通用性开发而必须的基础类和常用工具类等，包含有：

Think.Core 核心类库包

Think.Db 数据库类库包

Think.Util 系统工具类库包

Think.Template 内置模板引擎类库包

Think.Exception 异常处理类库包

ORG 和 Com 类库包主要用于基类库的扩展，ORG 类库包主要是第三方的公共类库，而 Com 类库包通常用于公司或者大项目，或者多年的开发经验而积累形成的类库包，主要是内部或者局部范围使用。默认情况下，ORG 类库包是已经有创建的，并且也包含了一些常用的类库，而 Com 类库包是需要自己来创建的，因此你在系统的 Lib 目录下面是看不到 Com 目录的。




ThinkPHP 是低耦合的框架么




可以这么认为，ThinkPHP 的核心（类库）是高耦合的，这些效率和机制的考虑，因为 ThinkPHP 的惯例配置贯穿始终，因此，ThinkPHP 的核心是不可拆分的（这里指的核心是指 Think 基类库。事实上，拆分出来使用的意义不大）。而扩展类库是低耦合的或者可替换的，因此扩展是非常容易的。

ThinkPHP 可以使用第三方的类库或者类库包么


 ThinkPHP 完全可以支持第三方的类库引入，放入 Vendor 目录后即可直接使用。导入方式参考下面的方式：Vendor('Zend.Filter.Dir'); 利用这个机制，我们完全可以把国外其他框架的低耦合类库直接在 ThinkPHP 中调用。

ThinkPHP 的类库一定要使用.class.php 后缀么

 ThinkPHP 的默认规范是类库名和文件名一致（包括大小写），并且后缀使用.class.php，这是为了更加方便系统内置的类库导入（import）机制。如果你的后缀使用.php 的话，一样可以通过参数控制导入，并不会影响正常使用。

3 URL 和控制器

可以让编译缓存保留空白和注释或者关闭编译缓存吗

 默认的情况下，为了缩小文件大小，系统对核心编译缓存和项目编译缓存文件去掉了空白和注释，但是可以通过如下的配置保留，以进行更加方便的调试定位。在入口文件里面添加下面的常量定义即可：

```
define('STRIP_RUNTIME_SPACE',false);
```

还可以关闭核心编译缓存（同样在入口文件里面定义）


```
defined('CACHE_RUNTIME',false);
```

如果开启了项目的调试模式的话，也会关闭项目的编译缓存。


只要在项目配置文件里面添加

```
'DEBUG_MODE'=>true,
```


ThinkPHP 是如何分析和识别 URL 的

 ThinkPHP 里面会根据当前的 URL 来分析要执行的模块和操作。这个分析工作由 URL 调度器来实现，官方内置了 Dispatcher 来完成该调度。在 Dispatcher 调度器中，会根据 `http://domainName/appName/moduleName/actionName/params` 来获取当前需要执行的项目（appName）、模块（moduleName）和操作（actionName），在某些情况下，appName 可以不需要（通常是网站的首页，因为项目名称可以在入口文件中指定，这种情况下，appName 就会被入口文件替代），另外针对不同的 URL 模式设置系统会进行不同的智能识别。

 **ThinkPHP 里面必须给每个操作定义方法么**

 对于没有任何业务逻辑的操作我们可以直接定义模板文件即可，这种情况下无需定义操作方法，系统会直接渲染模板文件输出。


 **ThinkPHP 的 Action 里面的 display 方法是如何定位模板文件的**

 ThinkPHP 的模板输出通常只需要写一个空白的 display 方法

```
$this->display();
```

该方法没有定义任何要输出的模板文件，但是系统会根据默认的规则去寻找模板目录下面的以模块目录的操作模板文件。例如，假如 display 方法位于 UserAction 类的 add 操作方法，那么系统会自动寻找模板目录下面的 User/add.html 模板文件，这就是为什么空的 display 方法也能输出模板的原因。当然，display 方法一样可以支持参数输出，而且有很多的规则。

 **ThinkPHP 的控制器名称是否一定要和数据表一致**

 ThinkPHP 的控制器（Action）类和数据表完全没有关系，怎么命名取决于你的项目如何进行模块化的设计。



ThinkPHP 的控制器名称是否一定要包含 Action 名称



ThinkPHP 的控制器命名规范默认是具有 Action 命名后缀的，你完全可以通过更改配置参数改变或者去掉后缀，例如设置：

```
'CONTR_CLASS_SUFFIX'=>"
```

就可以去掉 Action 后缀了，甚至我们可以给控制器类增加前缀定义，例如：

```
'CONTR_CLASS_PREFIX'=>'Think'
```

在做更改之前，请确保类名之间没有其他可能导致的冲突。



如何让访问<http://thinkphp.cn/index.php>的时候访问 Blog 模块而不是 Index 模块



可以在项目配置文件里面配置：

```
'DEFAULT_MODULE'=>'Blog'
```

就可以改变默认模块访问。



入口文件里面的 THINK_PATH 应该如何定义



入口文件里面的 THINK_PATH 主要用于定位 ThinkPHP 系统目录的位置，可以使用相对路径或者决定路径都可以，例如：

```
define('THINK_PATH', '../ThinkPHP');
```

或者

```
define('THINK_PATH', '/Home/ThinkPHP');
```

如果该位置已经加入 PHP 的搜索路径，可以无需定义。也就是说，框架的系统目录可以随意放置，哪怕是不在 WEB 访问路径下面。

或者干脆不用定义 THINK_PATH，而是直接包含框架的系统入口文件，例如把原来的：

```
define('THINK_PATH', '../ThinkPHP');
```

```
require(THINK_PATH."/ThinkPHP.php");
```

改成：

```
require ("../ThinkPHP/ThinkPHP.php");
```



入口文件里面的项目路径应该如何定义



入口文件里面的项目路径指的是项目目录（也就是项目 Lib、Conf 所在的目录）所在的路径，和项目的入口文件的位置没有关系，原则上，项目的入口文件可以随意摆放，只要是在 WEB 访问目录下面即可。只是随着项目入口文件的位置不同，项目路径的定义也会变化。项目路径的定义和 THINK_PATH 定义一样，采用相对路径和绝对路径都可以。



ThinkPHP 是否支持 PATHINFO



ThinkPHP 可以完美支持 PATHINFO，并且可以配置 PATHINFO 方式的 URL 分隔符，例如可以支持下面的 URL

<http://domainName/User/read/id/1>

<http://domainName/User-read-id-1>



ThinkPHP 必须要求服务器支持 PATHINFO 吗



ThinkPHP 除了 PATHINFO 模式外，还可以支持普通 URL 模式和兼容 URL 模式，这两种模式都可以用于不支持 PATHINFO 的服务器环境。例如，原来的 URL 可能变化为：


PATHINFO 模式(URL_MODEL=1)：<http://domainName/User/read/id/1>

普通 URL 模式 (URL_MODEL=0)：<http://domainName/?m=User&a=read&id=1>

兼容 URL 模式 (URL_MODEL=3)：<http://domainName/?s=/User/read/id/1>

例如对于 Nginx 环境和个别 FASTCGI 模式(国外的空间居多)下面有可能不支持 PATHINFO 模式，官方建议采用兼容 URL 模式进行处理，这样的方便之处是可以和 PATHINFO 模式实现配置切换，而不需更改任何模板文件。


 **ThinkPHP 能够实现下面的 URL 么 <http://domainName/User/3>**

 因为 User 后面的 id 是一个可变的参数，所以无法当成一个普通的操作名称来解析，ThinkPHP 里面有多种方案可以实现类似的 URL，包括：

- 1、使用 URL 路由功能把 User 路由到 User 模块的 read 操作
- 2、使用空模块和空操作功能

第二种方案还可以实现 <http://domainName/3> 这样的 URL


 **ThinkPHP 是否支持泛路由**

 ThinkPHP 可以很好的支持泛路由功能，可以通过配置正则来支持同名的路由实现不同的功能，例如：

<http://domainName/Blog/3>

<http://domainName/Blog/2008/12/>

 **ThinkPHP 对 SEO 优化是否有一定的支持**

 ThinkPHP 可以针对 URL 进行定制 在一定程度上可以满足 SEO 对 URL 设计的要求。例如，可以实现类似下面的 URL 地址：

<http://domainName/blog/3>

<http://domainName/blog-3.html>

http://domainName/blog_2008_12

 **如何隐藏网站 URL 地址里面的 index.php**

 去掉 URL 里面的 index.php 是为了 SEO 的需要，需要服务器开启 URL_REWRITE 模块。

下面的配置过程可以参考下：

- 1、httpd.conf 配置文件中加载了 mod_rewrite.so 模块
- 2、AllowOverride None 将 None 改为 All
- 3、确保 URL_MODEL 设置为 2
- 4、把.htaccess 文件放到入口文件的同级目录下，其中添加下面内容：

```
<IfModule mod_rewrite.c>

RewriteEngine on

RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ index.php/$1 [QSA,PT,L]

</IfModule>
```

ThinkPHP 的验证码为什么无法显示

 验证码无法正常显示，通常包括几个原因：

1. 检查你的 GD 库模块是否开启；
2. 检查你的程序在验证码之前是否有任何输出；
3. 如果使用了 UTF8 编码，请确保删除 UTF8 的 BOM 信息头；


ThinkPHP 可以同时执行多个操作么

 ThinkPHP 的操作链功能可以按照顺序执行多个定义好的操作，其访问格式是：

<http://domainName/appName/User/action1:action2:action3/>

上面的访问地址可以同时执行 User 模块的 action1、acton2 和 action3 操作方法。

ThinkPHP 会自动加载需要的类库么

 ThinkPHP 内置了类库自动加载机制，当前项目的 Action 类和 Model 类是会自动加载的，并且这个机制是可配置的，可以增加额外的自动搜索路径，只要配置 AUTO_LOAD_PATH 参数，例如系统默

认的配置是：

```
'AUTO_LOAD_PATH'          => 'Think.Util.'
```

因此 Think.Util 类库包下面的类库也是会自动加载的，如果需要额外的搜索路径，在配置参数里面增加就可以了，多个之间用逗号分割，注意搜索的顺序是按照定义的顺序。



可以让 ThinkPHP 在执行某个应用的时候自动初始化加载一些类库吗



可以在项目配置文件里面添加下面的配置参数：

```
'AUTO_LOAD_CLASS'        => '@.Action.CommonAction '
```

这样，系统在初始化的时候会自动导入 CommonAction 类库，利用这个机制你可以自动加载任何的基类库或者项目类库。

4 模型相关



ThinkPHP 可以支持哪些数据库



ThinkPHP 可以支持的数据库包括 MySQL、PgSql、Sqlite 和 Oracle，更多的数据库支持可以使用 PDO 方式连接。



ThinkPHP 的模型类的名称是否一定要和数据表一致




ThinkPHP 的模型（Model）类和数据表可以不一致，你只要设置模型类的 tableName 或者 trueTableName 属性即可。默认情况下保持和数据表一致的定义是为了让系统可以自动识别对应的数据表。系统可以自动识别的模型名称定义是包括下面两种方式(假设数据表的前缀定义是 think_)：

- 1、模型名和数据表名一致（不包括数据表的前缀和后缀），例如：UserModel 可以字段对应


数据表 think_user

- 2、 模型名采用驼峰法命名 (**1.5 正式版开始支持**), 例如 : UserTypeModel 可以自动对应数据表 think_user_type

ThinkPHP 是否支持跨数据库和跨服务器操作


 ThinkPHP 的模型类可以定义单独的数据库名称 , 查询的时候会自动加上当前所属的数据库。还可以给模型定义单独的数据库连接 , 可以使得某个模型可以支持不同的服务器上面的不同数据库类型。要注意的是跨服务器的查询要避免使用视图和 JOIN 查询。

为什么修改了数据库的字段后程序无法识别新的字段

 ThinkPHP 会对数据表的字段信息进行缓存 , 如果你修改了数据表的字段 , 会发生对应的字段信息无法保存的情况 , 这个时候需要删除 Data 目录下面的字段缓存文件。1.5 版本在调试模式下面会关闭数据表字段缓存。你也可以自己在项目配置里面添加 :

```
'DB_FIELDS_CACHE'=>false,      // 不缓存数据表的字段信息
```

可以自己在模型类里面定义数据表的字段信息么

 答案是肯定的 , 如果不希望 ThinkPHP 自动去获取数据表的字段信息并缓存 , 可以直接在模型类里面定义好相关的字段信息 , 这样的好处是可以节省文件读取的 IO 开销 , 建议在部署的时候可以使用 , 缺点是每次更改数据库的字段都必须手动更新。手动定义的格式是在模型类添加下面的格式定义 :

```
protected $fields = array(
    'id', 'username', 'email', 'age',      // 字段信息
    '_pk'=>'id',                          // 主键名称
    '_autoInc'=>true                      // 主键是否属于自动增长类型
)
```



ThinkPHP 的表单字段名称是否一定要和数据表的字段保持一致



默认情况下,表单提交的字段名要和数据表的字段保持一致,否则无法写入到数据表里面。但是,系统提供了表单字段映射功能,可以给数据表的字段定义表单映射,来避免用户直接知道数据表的字段设计。例如,下面的例子给用户模型定义了字段映射:

```
Class UserModel extends Model{
    protected $_map = array(
        'name' => 'username',
        'mail'  => 'email',
    );
}
```

这样,在表单里面就可以使用 name 和 mail 作为表单字段提交数据了,而实际的字段名是 username 和 email,数据表中相同的字段可以不用定义映射,仅仅需要定义不同的就可以了。



ThinkPHP 是否支持分布式数据库



ThinkPHP 内置支持分布式数据库的定义和查询,包括读写分离。可以参考如下的项目配置:

在项目配置文件里面定义

```
Return array(
    'DB_DEPLOY_TYPE' => 1, // 启用分布式数据库支持
    'DB_RW_SEPARATE' => true, // 设置读写操作分离
    'DB_TYPE' => 'mysql', // 分布式数据库的类型必须相同
    'DB_HOST' => '192.168.0.1,192.168.0.2', // 分布式数据库的地址
    'DB_NAME' => 'thinkphp', // 如果相同可以不用定义多个
    'DB_USER' => 'user1,user2',
    'DB_PWD' => 'pwd1,pwd2',
    'DB_PORT' => '3306',
```



```
'DB_PREFIX'=>'think_',  
  
..... 其它项目配置参数  
  
);
```

但是数据库的同步不是由框架自动完成，应该交给数据库本身来实现。



ThinkPHP 是否支持同时多个数据库连接



在系统的配置数据库连接之外，ThinkPHP 可以良好的支持多个数据库的连接和切换。这个连接是动态的，在程序里面实现。例如：

```
$User = D("User");  
  
// 创建多个数据库连接的 DSN  
  
$myConnect1 = 'mysql://username:passwd@192.168.1.1/DbName1';  
$myConnect2 = 'pgsql://username:passwd@192.168.1.2/DbName2';  
  
// 增加数据库连接 第二个参数表示连接的序号  
  
// 注意内置的数据库连接序号是 0,所以额外的数据库连接序号应该从 1 开始  
  
// 可以同时增加多个数据库连接  
  
$User->addConnect($myConnect1,1);  
$User->addConnect($myConnect2,2);  
  
// 切换当前要操作的数据库到连接 2  
  
$User->switchConnect(2);  
  
// 关闭连接序号为 2 的数据库连接  
  
$User->closeConnect(2);
```



查询的使用一定要使用 HashMap 对象么



早期的 ThinkPHP 版本采用 HashMap 进行查询的情况比较多，但是 1.0 版本之后官方推荐的查询方式是采用 PHP 数组的方式，效率更加高效。也就是说，之前的

```
$map = new HashMap();  
$map->put('name', 'ThinkPHP');
```

可以改成

```
$map['name'] = 'ThinkPHP';
```

的方式来查询了，前者是对象方式，后者是数组方式，结果一样，效率更高。



模型的自动验证和自动完成有什么区别



模型的自动验证主要是对表单提交的数据进行验证是否符号要求，自动完成是对表单没有的数据进行添加，或者对提交的数据进行过滤，两者的配合可以保证写入数据库的数据是符合要求的数据信息。这两个功能需要数据的创建是使用 Create 方法创建的，如果没有使用 Create 方法创建数据对象的话，可以使用 1.5 正式版添加的字段自动过滤功能。



在自动验证的定义里面 Callback 和 function 方式的区别是什么



callback 是模型的方法调用 function 是函数调用



视图模型是什么含义




ThinkPHP 在 ORM 模型里面模拟实现了数据库的视图模型，该功能可以用于多表查询。和数据库的视图概念区别是视图模型修改比较方便，无需修改数据库本身。而且不需要数据库的视图支持，这是 ThinkPHP 框架的亮点之一。下面定义了一个 Blog 视图模型，我们通过创建 BlogView 模型来快速读取一个包含了 User 名称和类别名称的 Blog 记录（集），其查询方式和普通模型一样。

```
class BlogViewModel extends Model  
{  
    protected $viewModel = true;  
    protected $viewFields = array(  
        'Category'=>array('title'=>'categoryName'),
```


```
'User'=>array('name'=>'userName'),  
'Blog'=>array('id','name','title'),  
);  
}
```

5 配置相关


ThinkPHP 的配置文件采用什么格式

 ThinkPHP 的配置文件采用效率最高的 PHP 返回数组方式，不需要额外的解析过程。只要会使用 PHP 的数组，就会定义 ThinkPHP 的配置文件。


ThinkPHP 的配置文件的优先级别依次是什么

 ThinkPHP 中配置文件的优先顺序从低到高依次是：（在没有生效的前提下）
惯例配置 → 项目配置 → 调试配置 → 模块配置 → 操作（动态）配置


如何查看 ThinkPHP 的系统惯例配置都有哪些参数

 ThinkPHP 的系统惯例配置文件在系统目录的 Common\convention.php，里面列出了系统的所有配置参数以及默认配置，并且有详细的注释，具体也可以参考官方的配置指南和参考文档。

ThinkPHP 的模块配置的名称格式是什么

 ThinkPHP 的模块配置文件位于项目的 Conf 目录下面，命名规范是：
模块名称_config.php （注意模块名称区别大小写）


ThinkPHP 有什么命名规范

 ThinkPHP 的开发过程中尽量遵循下面的文件命名规范：

- 1、 所有类文件必须使用.class.php 作为文件后缀；
- 2、 类名和文件名保持一致（包括大小写一致）；
- 3、 尽量使用驼峰法命名类名

6 模板引擎


如何关闭 ThinkPHP 的模板缓存

 ThinkPHP 的模板缓存是无法关闭的，因为内置的模板引擎是一个编译型的模板引擎，必须经过编译后生成一个可执行的缓存文件才能被执行。但是可以设置缓存的有效期，例如设置

```
'TMPL_CACHE_TIME'    =>3,        // 模板缓存有效期 -1 永久 单位为秒
```

这样，每隔 3 秒系统会自动重新编译模板文件。默认的配置是-1 表示永久缓存，除非模板文件有改动，模板文件一旦有改动会自动重新编译，如果是包含进来的外部文件有修改，系统是不会自动重新编译的。

ThinkPHP 的模板如何使用 PHP 本身作为模板引擎

 ThinkPHP 内置的模板引擎也支持直接在模板文件里面使用 PHP 代码，如果你不想使用任何模板引擎标签的话，可以配置模板引擎类型为 PHP 就可以完全使用 php 本身作为框架的模板引擎，在项目配置里面添加：


```
'TMPL_ENGINE_TYPE'    =>'php'
```

ThinkPHP 的模板可以使用第三方的模板引擎吗

 ThinkPHP 框架允许你使用第三方的模版引擎。目前官方已经提供了 Smarty 模版引擎的插件，已经有人给 ThinkPHP 开发了 TemplateLite、EaseTempalte 和 DzTemplate 模版引擎插件。而且对于自己

熟悉的模版引擎来说，非常容易扩展类似的插件。然后在项目配置文件里面配置使用何种模板引擎就可以了。

如何输出其他模块的操作模板

 系统提供的 display 方法支持调用不同位置的模板文件，包括其他模块的操作，例如下面的方法可以调用 Member 模块的 read 操作模板：

```
$this->display('Member:read');
```


模板文件开头使用<tagLib name="html" />是什么意思

 这表示当前模板文件要加载 html 标签库，这样在模板文件里面就能使用类似

<html:list > <html:link > 之类的标签了，内置的模板引擎是基于标签库和 XML 解析的，所以必须要引入相应的标签库才能进行标签解析，因为系统默认会加载 cx 标签库，所以

<volist > <eq > 这样的标签是不需要自己加载标签库的。Cx 标签库之外的都需要在模板文件的开头用<tagLib 标签首先引入标签库。

某些编辑器无法识别 XML 标签，模板标签的定界符可以修改吗

 内置的模板引擎默认采用的是 XML 标签作为标签的定界符，但是可以修改的，下面是系统默认的配置，包括普通模板引擎和标签库的标签的起始和结束标记：

```
'TMPL_L_DELIM'=>'{' ,    // 模板引擎普通标签开始标记
```


```
'TMPL_R_DELIM'=>'}' ,    // 模板引擎普通标签结束标记
```

```
'TAGLIB_BEGIN'=>'<' ,    // 标签库标签开始标记
```


```
'TAGLIB_END'=>'>' ,      // 标签库标签结束标记
```

需要注意的两种类型的标记不要设置为相同的，以免引起混淆而无法正确解析。

 我不想直接输出模板文件的内容，而是想获取模板输出的内容应该怎么处理

 Action 类的 display 方法是用于渲染模板文件并输出，可以使用 fetch 方法渲染模板文件但不是直接输出，而是返回内容。

 模板文件里面经常使用到的__URL__和__APP__有什么作用

 如果使用了内置模板引擎的话，可以在模板文件里面使用一些已经定义好的特殊字符串，系统在解析模板的时候会自动替换成相关的系统常量，这些字符串的解析过程是在模板编译的时候进行的。

这些可替换的字符串包括：

../public //项目公共目录

__PUBLIC__ //网站公共目录

__ROOT__ //网站根目录

__TMPL__ //当前模板目录

__APP__ //当前项目地址

__URL__ //当前模块地址

__ACTION__ //当前操作地址

__SELF__ //当前页面地址

 如何在模板文件里面直接输出系统变量和常量

 系统变量，必须以\$Think.打头，如

{ \$Think.server.script_name } //取得\$_SERVER 变量

{ \$Think.session.session_id } // 获取\$_SESSION 变量

{ \$Think.get.pageNumber } //获取\$_GET 变量

```
{Think.cookie.name } //获取$_COOKIE 变量
```

输出系统常量

```
{Think.const.__FILE__ }
```

```
{Think.const.MODULE_NAME }
```

7 错误和调试



ThinkPHP 调试起来方便吗



ThinkPHP 支持调试模式，在调试模式下面系统默认开启了日志记录、关闭了页面防刷新机制、关闭了模板缓存，记录了执行过程中的 SQL 语句和运行时间，并且开启了页面运行时间显示和 Trace 功能。

在程序的数据调试输出方面，我们提供了三个非常有用的方法：

dump(\$var) //在浏览器输出方便查看的变量信息

halt(\$msg) //输出信息，并中止执行

system_out(\$msg) //输出调试信息到日志文件



ThinkPHP 的页面 Trace 是什么意思



页面 Trace 功能是为了用于调试当前页面状态信息、错误记录和 SQL 记录，是一个非常有帮助的调试手段。而且开发人员可以定制需要显示的信息。



ThinkPHP 有记录 SQL 日志的功能吗



系统可以开启 SQL 日志记录功能，设置

```
'SQL_DEBUG_LOG' => true
```

开启后会在日志目录 (Logs) 下面按照日期生成类似下面的日志文件：

08_05_13_systemSql.log // 2008 年 5 月 13 日的 SQL 日志文件

里面详细记录了 SQL 的执行时间和语句，格式如下：

[08-05-07 20:38:33]

RunTime:0.000528s SQL = select `id` from `think_cache` where `cachekey`='aaaa' limit 0,1



怎么查看我上次执行的查询语句是什么



如果你开启了 SQL 日志记录功能，可以在 sql 日志文件里面查看最近执行过的 sql 语句和时间。

如果没有开启，也可以使用调试方法来查看，下面的代码可以查看上次执行的 SQL 语句：

```
$User = D('User');  
  
// 执行查询  
  
$User->where('status>1')->order('create_time desc')->limit(10)->findAll();  
  
// 查看上次执行的 SQL 语句  
  
echo $User->getLastSql();
```

8 其他



经常看到的 D 方法和 C 方法是什么意思



ThinkPHP 为一些常用的操作定义了快捷方法，这些方法具有单字母的方法名，具有比较容易记忆的特点，D 方法和 C 方法是其中用的比较多的。

D 方法用于快速创建模型对象的实例，并且单例化，例如：

```
$User = D("User");
```

等效为

```
$User = new UserModel();
```


C 方法用于快速获取和修改配置参数，例如：

设置名称为 USER_AUTH_ON 的配置参数


```
C('USER_AUTH_ON',true);
```

获取 USER_AUTH_ON 的变量值


```
C('USER_AUTH_ON');
```

除了 D 和 C 方法外，系统还提供了 A、S 和 L 方法，具体可以查阅手册。

ThinkPHP 是否可以支持 JQuery 和其他的 JS 框架

 ThinkPHP 框架本身不会干涉客户端的任何东西，包括 JS，唯一的一个区别是因为框架采用了单一入口和默认的 PATHINFO 模式，改变了传统的 URL 路径，也就是说所有所有的调用路径应该都是基于入口文件的 URL 位置来。


如何采用子目录的方式来缓存数据

 系统默认的文件缓存是在同一个目录下面的，也就是 Temp 目录，但是可以配置启用子目录缓存：


```
'DATA_CACHE_SUBDIR'=>True
```

这样，系统会自动生成文件的哈希子目录来存放数据缓存，防止出现同一个目录下面缓存数据过多的情况。

使用 ThinkPHP 的过程中为什么总是容易发生乱码

 ThinkPHP 默认使用的是 UTF8 编码，确保你的编码设置正确并且和你的配置保持一致。

使用 ThinkPHP 开发一定要使用 UTF8 编码么

 ThinkPHP 使用 UTF8 编码只是参考目前的网站开发标准而采取的一种默认配置和建议，你完全可以通过配置更改你需要使用的编码格式，包括数据库编码、模板文件编码和输出编码，而且一旦配

置系统还可以实现编码的自动转换。默认的配置下，以上三者的编码都是 UTF8 编码，对于相同的编码格式系统不会进行额外的编码转换过程。



如何用 S 方法删除缓存



利用 `S('name',NULL)`；可以删除标识为 `name` 的缓存。



如何设置永久缓存某个数据



设置缓存有效期为 -1 就可以永久缓存某个数据，例如 `S('name',$data,-1)`；



ThinkPHP 是怎么进行安全过滤



系统提供了多个层面的安全过滤机制，最大程度的保证了数据的安全。

- 1、 首先，数据库底层驱动类已经对查询进行了安全过滤；
- 2、 在模型里面可以定义自动验证来对提交的数据进行校验；
- 3、 利用自动完成机制对非法篡改的数据进行重新写入；
- 4、 利用字段自动过滤功能对写入数据库的字段进行过滤；
- 5、 利用 Action 的 `getParam` 方法对提交的数据进行更严格的用户自定义过滤。
- 6、 利用基于 RBAC 的权限验证机制防范没有授权的操作

ThinkPHP 提供了各种手段，但不代表系统会自动帮你完成各种安全过滤，完全要根据项目的要求进行合理的配置。



更新 ThinkPHP 后为什么出现 `file_exists_case` 方法不存在的错误



更新新版后请记得删除 Temp 目录下面的项目编译缓存文件。