ThinkPHP Framework 1.0 QuickStart

ThinkPHP 1.0 快速入门

编写：ThinkPHP 文档组

最后更新：2008-05-11

目录

1 版权信息

对 ThinkPHP 有任何疑问或者建议，请进入官方论坛［ http://bbs.thinkphp.cn ］发布相关讨论。

并在此感谢 ThinkPHP 团队的所有成员和所有关注和支持 ThinkPHP 的朋友。

有关 ThinkPHP 项目及本文档的最新资料，请及时访问 ThinkPHP 项目主站 http://thinkphp.cn 。

2 什么是 ThinkPHP

ThinkPHP 是一个免费开源的，快速、简单的面向对象的轻量级 PHP 开发框架，遵循 Apache2 开源协议发布，是为了简化企业级应用开发和敏捷 WEB 应用开发而诞生的。借鉴了国外很多优秀的框架和模式，使用面向对象的开发结构和 MVC 模式，融合了 Struts 的 Action 思想和 JSP 的 TagLib（标签库）、RoR 的 ORM 映射和 ActiveRecord 模式，封装了 CURD 和一些常用操作，单一入口模式等，在模版引擎、缓存机制、认证机制和扩展性方面均有独特的表现。

使用 ThinkPHP，你可以更方便和快捷的开发和部署应用，当然不仅仅是企业级应用，任何 PHP 应用开发都可以从 ThinkPHP 的简单、兼容和快速的特性中受益。简洁、快速和实用是 ThinkPHP 发展秉承的宗旨，为此 ThinkPHP 会不断吸收和融入更好的技术以保证其新鲜和活力，提供 WEB 应用开发的最佳实践！

ThinkPHP 遵循 Apache2 开源许可协议发布，意味着你可以免费使用 ThinkPHP，甚至允许把你的 ThinkPHP 应用采用商业闭源发布。

3 系统特性

- 简单易用的 MVC 模式
- 独创的核心编译和项目编译机制
- 内置 XML 模板引擎，支持标签库
- 富模型支持
- CURD 和操作高度自动化支持
- 丰富的查询语言支持

- 目录结构自动创建
- 分布式数据库支持
- 多数据库连接和切换支持
- ActiveRecords 模式和丰富的 ROR 特性
- 灵活简单的项目配置
- 模型自动验证和处理
- 静态页面生成和多元化缓存机制
- 丰富的数据库及 PDO 支持
- SEO 和 URL 路由支持
- AJAX 支持
- 易扩展的系统基类库
- 自动编码转换
- 组件和插件支持
- 基于角色的权限控制体系
- 详尽的开发指南和全中文注释

4 环境要求

ThinkPHP可以支持WIN/Unix服务器环境，正式版需要PHP5.0以上版本支持，支持Mysql、PgSQL、Sqlite 以及 PDO 等多种数据库，ThinkPHP 框架本身没有什么特别模块要求，具体的应用系统运行环境要求视开发所涉及的模块。

5 获取 ThinkPHP

获取 ThinkPHP 的方式很多，官方网站（ http://thinkphp.cn ）是最好的下载和文档获取来源。

官方首页提供了 ThinkPHP 的核心包和完整包下载，

- 核心包仅仅包含 ThinkPHP 框架本身
- 完整包除了核心框架外，还包括示例、手册和开发指南

你还还可以通过 SVN 获取最新的更新版本。

SVN 地址：http://thinkphp.googlecode.com/svn/trunk

更多的 ThinkPHP 相关资源：

Google 项目地址：http://code.google.com/p/thinkphp/

SF 项目地址：http://sourceforge.net/projects/thinkphp

6 目录结构

ThinkPHP 的目录结构非常清晰和容易部署。大致的目录结构如下，以项目为基础进行部署。

┌─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

│ └ 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）
┃
├─App  App 项目目录
┃  ├ index.php 项目入口文件
┃  ├ 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   Model 类文件目录
┃    … 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
```

┃　┗ Help 助手类目录

┃

┃...更多项目目录（和 App 目录类似，每个项目采用独立目录，便于部署）

┃

┣Public 网站公共目录（多项目公用）

┃ ┣ Js JS 类库目录（建议）

┃ ┣ Images 公共图像目录（建议）

┃ ┗ Uploads 公共上传目录（建议）

ThinkPHP 框架除了模板目录和网站入口文件必须放到 WEB 目录下之外，其它所有框架的文件和目录可以单独存放，不受限制，您需要做的仅仅是在首页文件中指定 ThinkPHP 框架的包含目录，我们建议您如果可能的话把 ThinkPHP 框架的目录包放到其它网站不能访问的目录下面，以保障应用的安全性。项目独立目录，方便部署和团队开发。每个项目有自身的配置文件、语言文件、插件文件和日志文件。

如果在类 Linux 环境下面部署，需要对以下目录设置可写权限（这些目录仅仅针对项目目录，系统目录无需设置任何可写权限，因为每个项目的模版缓存和数据缓存，以及日志文件都是独立的）。

项目目录下面的 Cache（模版缓存目录）、Temp（数据缓存目录）、Conf（项目配置目录，写入权限用于自动生成配置缓存和插件缓存文件）、 Logs（日志文件目录）、如果设置了 Uploads 上传目录和 Data 数据目录的话也必须设置为可写。另外，如果设置了 Public 目录下面的 Uploads 目录作为公共上传目录，也需要设置可写权限。通常的设置都是设置目录属性为 777。

一定要注意在 Linux 环境下面的文件大小写问题，否则会导致文件加载错误。

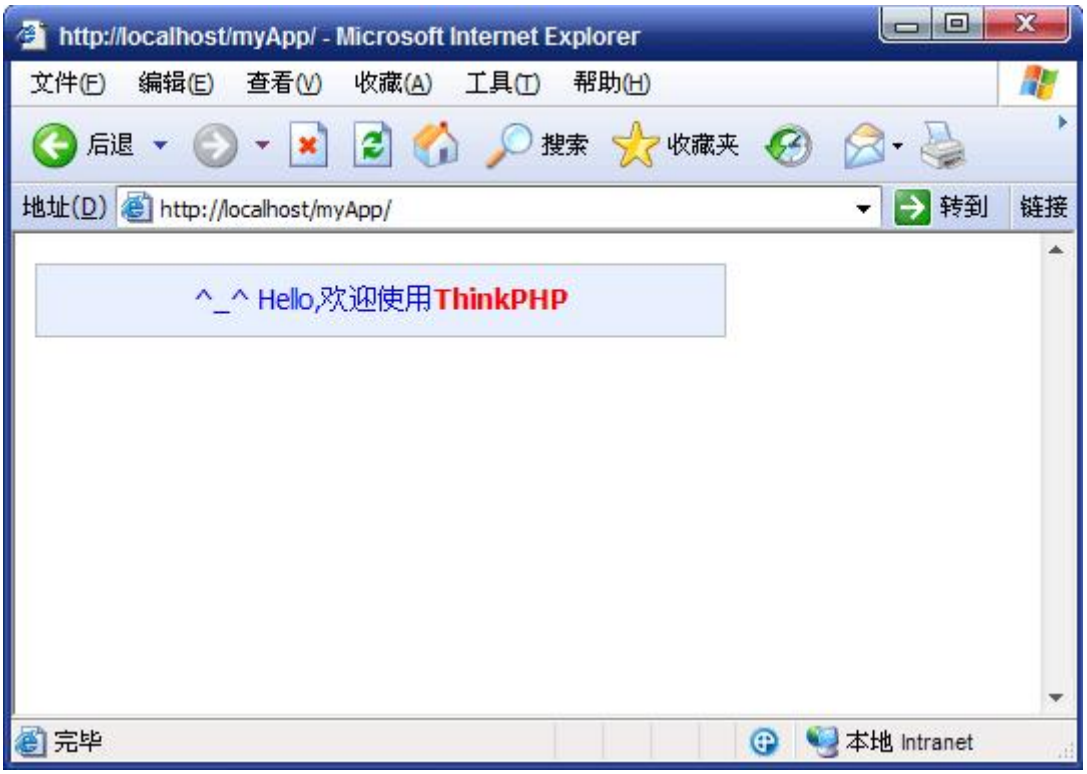
7 创建项目

利用 ThinkPHP 创建项目非常简单，ThinkPHP 具有项目目录自动创建功能，你只需要定义好项目的入口文件，第一次执行入口文件的时候，系统会自动创建项目的相关目录结构，如果是 linux 环境下面需要给项目入口文件里面指定的路径设置可写权限。

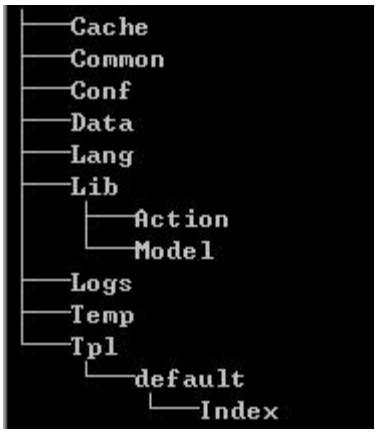
在进行后面的操作之前，我们假设把之前获取的 ThinkPHP 核心目录放到 WEB 根目录下面，然后，我们在 WEB 的根目录下面创建一个 myApp 目录，并且在下面定义了一个项目入口文件 index.php，内容如下：

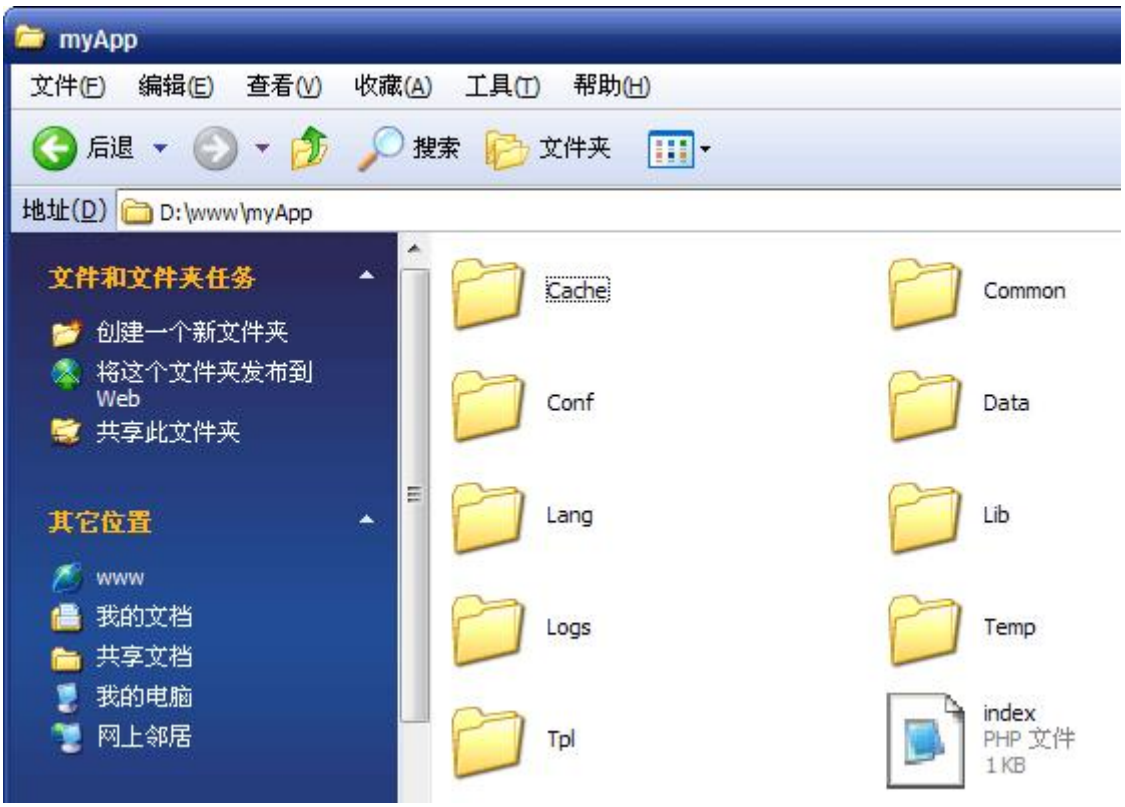
```php
<?php
// 定义 ThinkPHP 框架路径
define('THINK_PATH', '../ThinkPHP/');
//定义项目名称和路径
define('APP_NAME', 'myApp');
define('APP_PATH', '.');
// 加载框架入口文件
require(THINK_PATH."/ThinkPHP.php");
//实例化一个网站应用实例
$App = new App();
//应用程序初始化
$App->run();
?>
```

然后，我们在浏览器里面输入http://localhost/myApp/，你会在浏览器中看到下面的画面



系统已经在第一次执行的时候自动完成项目的目录结构的创建，同时，还自动创建了一个 Hello,world 应用，下面是自动生成的目录结构：





需要引起注意的是，在项目的 Temp 目录下面，还生成了两个编译缓存文件：~runtime.php 和 ~app.php。这两个文件是 ThinkPHP 的为了减少运行时的文件加载开销而生成的编译缓存文件。其中，~runtime.php 是核心编译缓存，把框架运行所需要的核心类库缓存到一个文件里面，~app.php 是项目编译缓存文件，其中包含了项目的公共文件、项目配置等，一旦项目配置和公共文件发生修改，就必须删除该文件重新生成。

注意 ThinkPHP 框架的所有文件都是采用 UTF-8 编码保存，但是这不影响你的项目中使用其他编码开发和浏览。请注意确保文件保存的时候去掉 UTF-8 的 BOM 头信息，防止因产生隐藏的输出而导致程序运行不正常。

8 URL 访问

ThinkPHP 框架的应用采用单一入口文件来执行，所有的模块和操作都通过 URL 的参数来访问和执行。

ThinkPHP 支持的 URL 模式包括普通模式、PATHINFO 模式和 REWRITE 模式，并且都提供路由支持。

默认为 PATHINFO 模式，提供最好的用户体验和搜索引擎友好支持。

例如普通模式下面的 URL 为：

http://localhost/appName/index.php?m=moduleName&a=actionName&id=1

如果使用 PATHINFO 模式的话，URL 成为：

http://localhost/appName/index.php/moduleName/actionName/id/1/

PATHINFO 模式对以往的编程方式没有影响，因为 GET 和 POST 方式传值依然有效，因为系统会对 PATHINFO 方式自动处理。

如果使用 REWRITE 模式，通过配置 URL 可以成为：

http://localhost/appName/moduleName/actionName/id/1/

例如上面生成的 myApp 项目我们可以通过下面的 URL 访问

http://localhost/myApp/

其实是定位到 myApp 项目的 Index 模块的 index 操作，因为系统在没有指定模块和操作的时候，会执行默认的模块和操作，这个在 ThinkPHP 的惯例配置里面是 Index 模块和 index 操作。因此下面的 URL 和上面的结果是相同的：

http://localhost/myApp/index.php/Index/index/

通过项目配置参数，我们可以改变这个默认配置。

9 控制器

ThinkPHP 的控制器就是模块类，位于项目的 Lib\Action 目录下面。我们可以发现在 Action 目录下面存在了一个 IndexAction.class.php 文件，这个就是自动创建的默认模块类。在 ThinkPHP 框架的规范里面，类名和文件名是一致的，并且类文件都是以.class.php 为后缀。而 IndexAction 类就表示了 Index 模块。index 操作其实就是 IndexAction 类的一个方法，所以我们在浏览器里面输入 URL

http://localhost/myApp/index.php/Index/index/

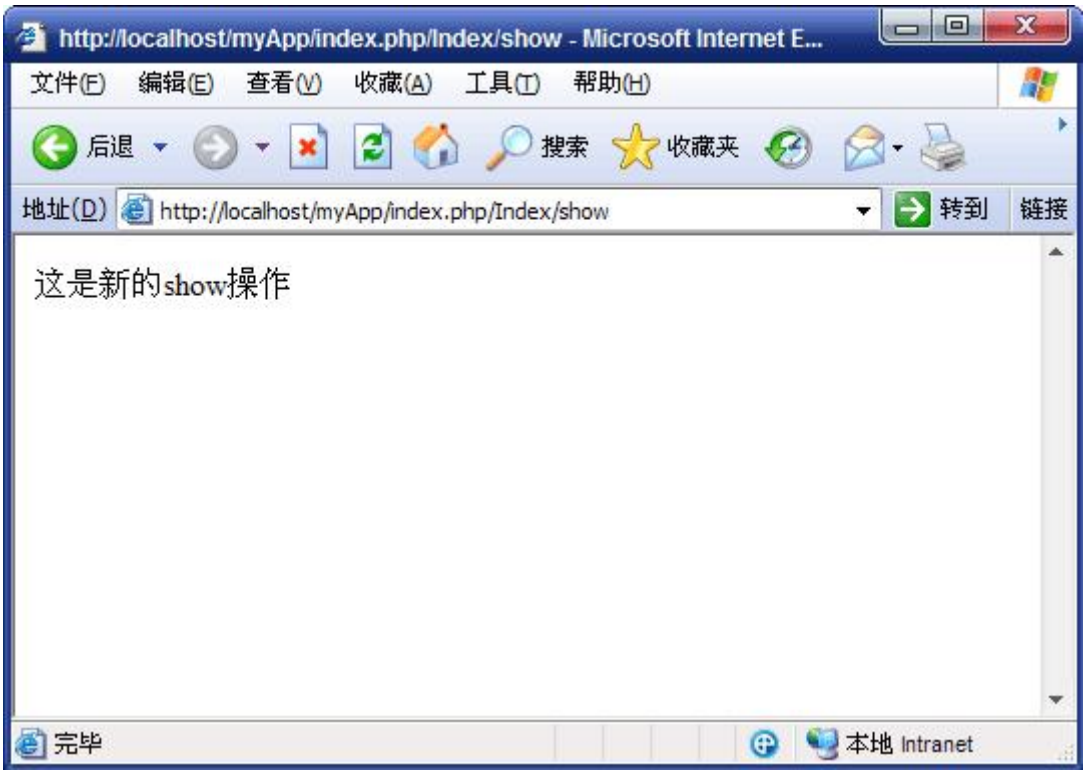
的时候其实就是执行了 IndexAction 类的 index（公共）方法，因此我们看到了输出的结果。我们来新增一个操作方法 show。

```
class IndexAction extends Action{
    public function show(){
        echo '这是新的 show 操作';
    }
}
```

然后在浏览器里面输入

http://localhost/myApp/index.php/Index/show/

我们就可以看到刚才定义的输出了。



如果你看到的页面是乱码，请选择页面的查看编码为 unicode(utf-8)。

如果你看到的仍然是之前的 index 操作方法的输出信息，那么可能你的运行环境不支持 PATHINFO，所以无法识别 Index 模块的 show 操作，而仍然去执行默认的 index 操作。如果遇到这个问题，我们先继续往后面看，了解了如果配置后我们依然可以看到上面的页面 ^_^

10 配置文件

对于简单的应用 ThinkPHP 可以无需定义任何配置文件，因为 ThinkPHP 框架有自身的的惯例配置。默认生成的项目并没有包含项目配置文件，如果需要我们可以自己定义。ThinkPHP 的项目配置文件采用 PHP 数组的方式配置，我们在 Conf 目录下面创建一个 config.php 文件，内容如下：

```
<?php
return array(
'DEFAULT_ACTION' => 'show',
);
```

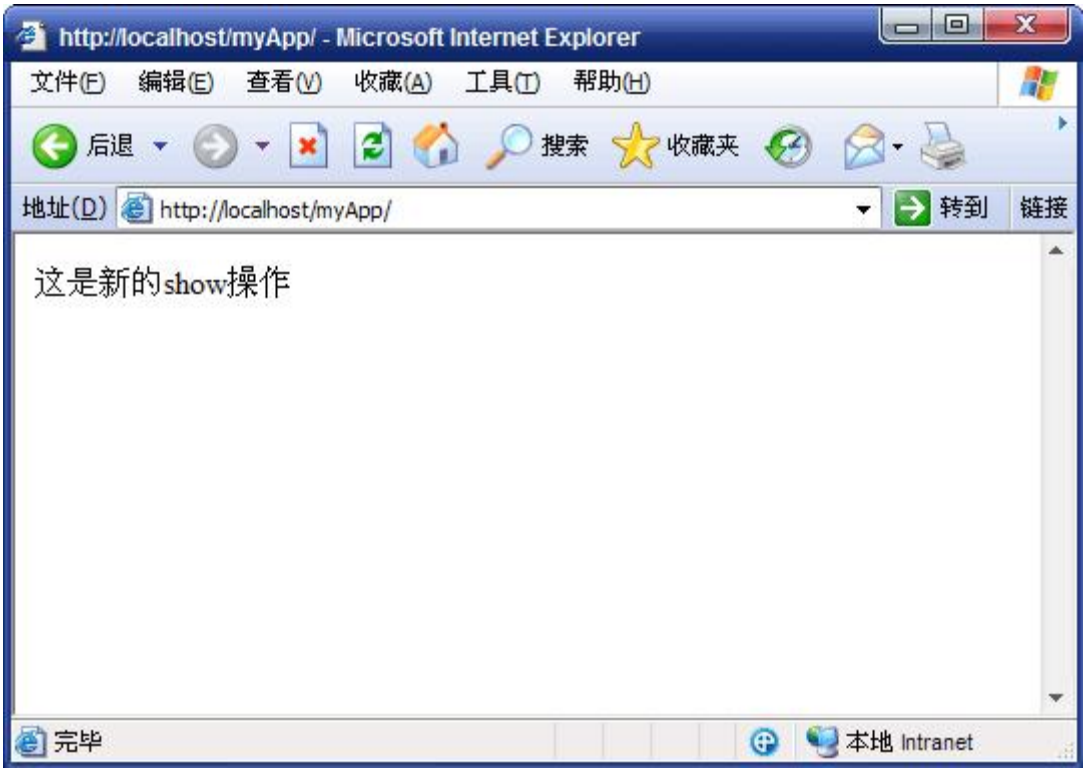
```
?>
```

上面的配置参数更改了默认操作为 show。配置文件的参数可以参考惯例配置（位于 ThinkPHP 系统目录下面的 Common\convention.php）文件，也可以增加自己项目需要的参数用来帮助开发。

需要特别注意的是，一旦我们增加或者修改了配置文件，必须删除我们前面提到的项目编译缓存文件~app.php，否则修改的配置将不会生效。

删除 Temp 目录下面的~app.php 文件后，我们在浏览器里面输入：

http://localhost/myApp/



可以看到现在输出的是 show 方法里面定义的输出了，而不是之前的 index 方法的输出了，说明我们的默认操作已经变成 show 方法了。如果这个时候要访问 index 操作，就必须使用

http://localhost/myApp/index.php/Index/index/

的 URL 来访问了。

下面我们来处理之前提过的由于环境不支持 PATHINFO 而可能出现的问题，如果你没有遇到此类问题，可以跳过，也许你也非常希望了解 ThinkPHP 的传统 URL 模式怎么定义。其实很简单，我们只要把 URL 模式切换回普通模式就可以了，或者关闭 DISPATCH 都可以。我们修改配置文件为：

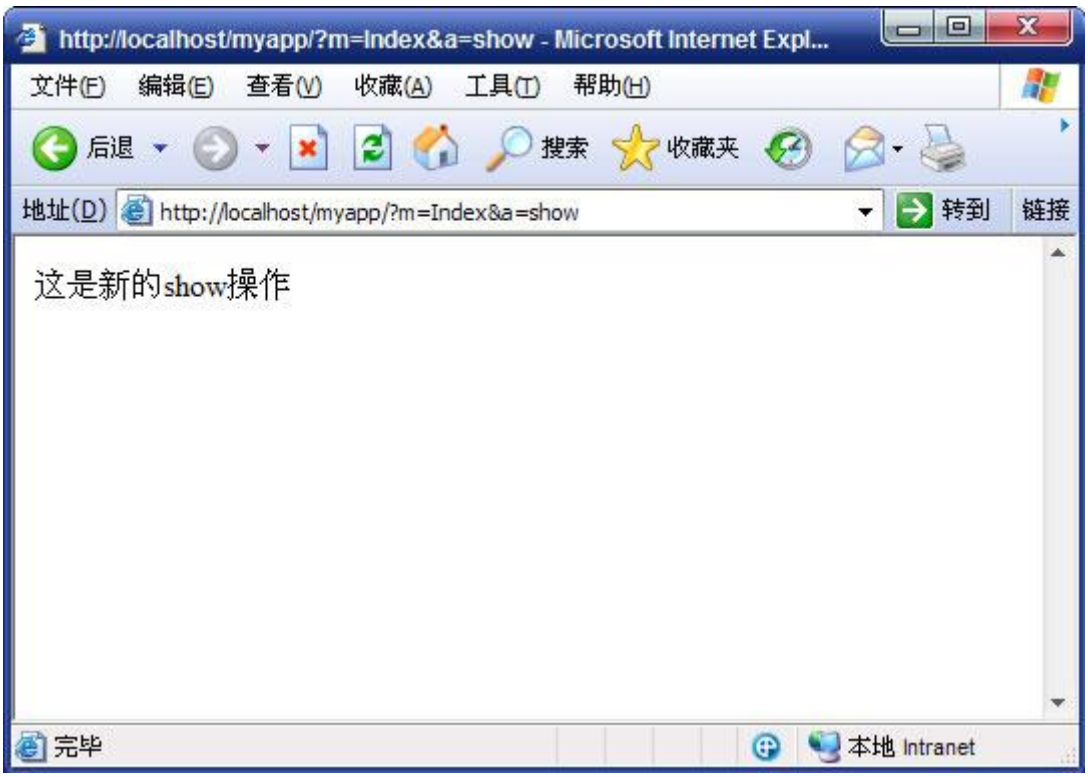
```
<?php
return array(
'URL_MODEL' => 0,
 // 或者使用下面的方式关闭 DISPATCH
'DISPATCH_ON' => false,
);
```

记得修改配置文件，一定要删除 Temp 目录下面的~app.php 文件。

接下来，我们使用下面的方式来访问 Index 模块的 show 操作

http://localhost/myApp/?m=Index&a=show

现在我们看到了 show 操作的输出了



回到了传统的 URL 参数模式 m 变量表示模块名 a 表示操作名，这也是框架的惯例配置定义的，你完全可以改变参数的定义。

11 读取数据库

目前为止，我们只是简单的输出静态数据，如果要读取数据库的动态数据，需要定义模型和数据库连接信息。首先，我们需要创建数据表（以 MySQL 为例，如果是其他数据库请修改后再执行）

```
CREATE TABLE `think_blog` (
  `id` int(11) unsigned NOT NULL auto_increment,
  `title` varchar(255) NOT NULL default '',
  `content` longtext NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 ;
Insert think_blog (id,title,content) values
   (1, '第一条标题', '测试内容'),
   (2, 'Hello,World! ', '欢迎使用 ThinkPHP！');
```

在数据库创建了 think_blog 表之后，我们在项目配置文件里面添加数据库连接信息：

```
<?php
return array(
 // 定义数据库连接信息
'DB_TYPE'=> 'mysql',
'DB_HOST'=> 'localhost',
'DB_NAME'=>'thinkphp',
'DB_USER'=>'root',
'DB_PWD'=>'',
'DB_PORT'=>'3306',
```

```
'DB_PREFIX'=>'think_',
);
?>
```

确保配置文件里面的设置和你本地的数据库连接信息一致。

接下来，我们在 Lib\Model 目录下面创建一个 BlogModel.class.php 文件，内容如下：

```
class BlogModel extends Model{
}
```

只需要建立一个空的 BlogModel 类就可以了，我们就可以完成常用的数据存取操作了。

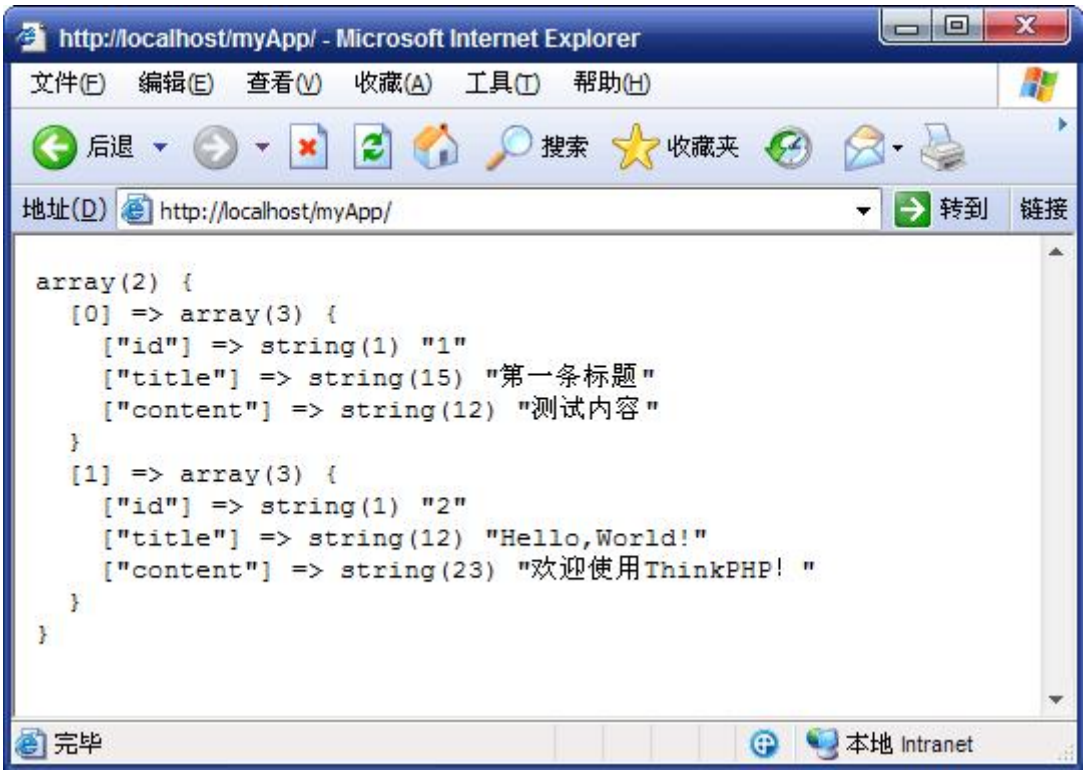
定义了模型类，我们还需要修改 Action 类的操作方法，来获取数据并显示出来。

我们把原来自动生成的 index 操作方法修改成下面的代码：

```
class IndexAction extends Action{
    public function index(){
      $Blog = new BlogModel();
      $list = $Blog->findAll();
      dump($list);
     }
}
```

然后我们就可以在浏览器里面输入下面的 URL 访问 index 操作

http://localhost/myApp/



另外我们看到了页面下面有一些额外的输出信息 这个开启调试模式之后 系统默认的调试信息输出。

这个功能在 ThinkPHP 里面称为页面 Trace 信息，可以显示当前页面的执行信息，以及发生的错误和执行的 SQL 语句，该功能可以通过关闭调试模式来关闭显示。

12 使用模板

到目前为止，我们只是使用了控制器和模型，还没有接触视图，下面来给上面的应用添加视图模板。

首先我们修改下 Action 的 index 操作方法，添加模板赋值和渲染模板操作。

```
class IndexAction extends Action{
    public function index(){
      $Blog = new BlogModel();
      $list = $Blog->findAll();
      $this->assign('title', 'ThinkPHP 示例');
```

```
        $this->assign('list',$list);
        $this->display();
    }
}
```

现在我们在浏览器里面输入 http://localhost/myApp/ 后出现了下面的页面

出现上面的错误，可是觉得莫名其妙，因为没有说明任何导致错误的原因。

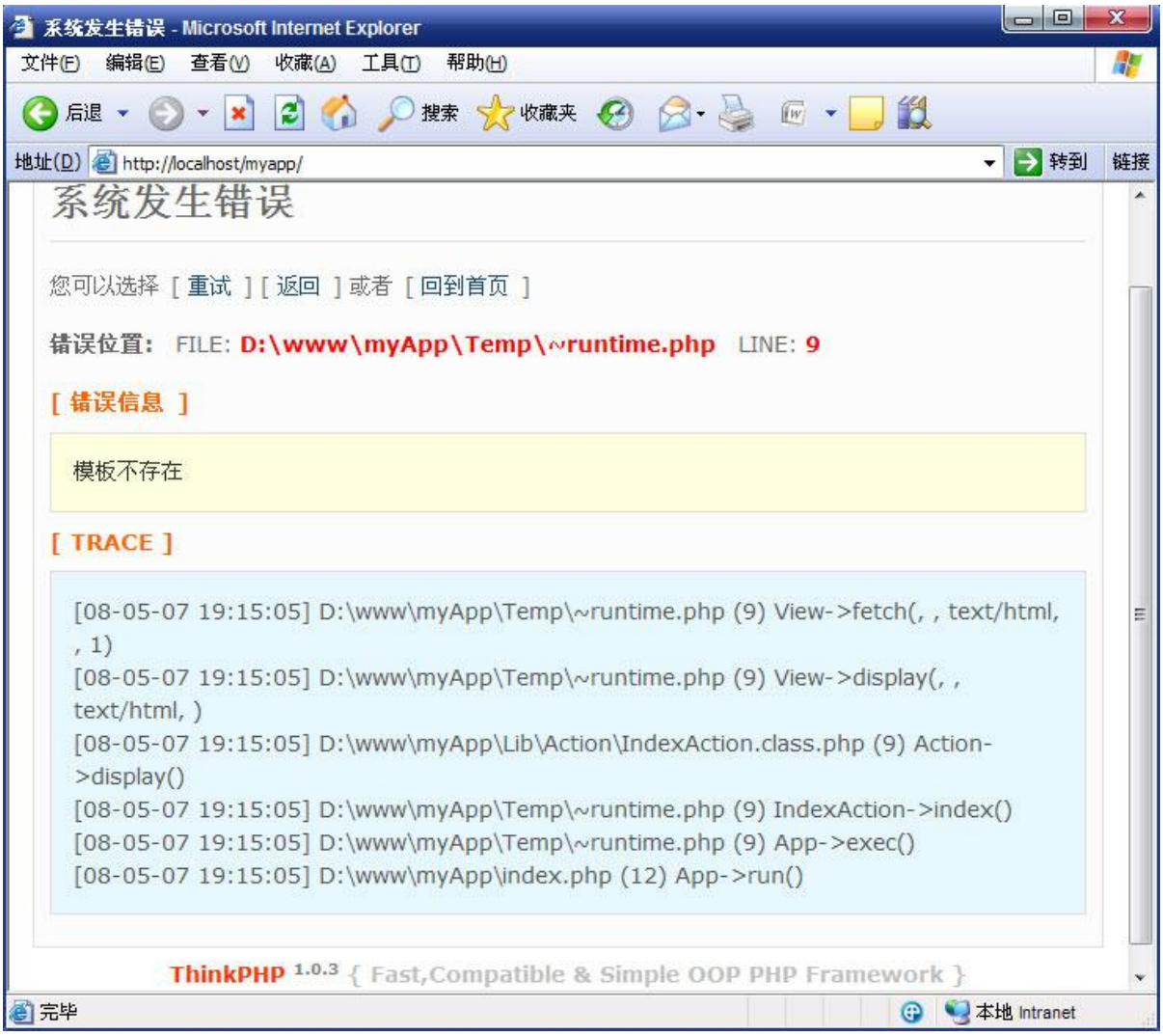
其实，这个提示信息是 ThinkPHP 在部署模式下面的默认提示信息，没有提示具体的错误原因是为了避免把一些不必要的信息暴露给用户，从而导致一些安全隐患，在开发过程中，我们只需要开启调试模式就可以看到具体的错误信息了。

下面，我们修改下项目配置文件，加上调试模式的配置：

```
<?php
return array(
 // 定义数据库连接信息
'DB_TYPE'=> 'mysql',
```

```
'DB_HOST'=> 'localhost',
'DB_NAME'=>'thinkphp',
'DB_USER'=>'root',
'DB_PWD'=>'',
'DB_PORT'=>'3306',
'DB_PREFIX'=>'think_',
// 开启调试模式
'DEBUG_MODE'=>true,
);
```

删除 Temp 目录下面的~app.php 文件后，我们再次刷新下浏览器，看到了错误信息



看了提示的错误信息，原来我们还没有给操作定义模板文件，我们在项目的 Tpl\default\Index\下面

创建一个 index.html 模板文件，内容如下：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>{$title}</title>
</head>
<body>
<volist name="list" id="vo">
[ {$vo.title} ] {$vo.content}<br/>
</volist>
</body>
</html>
```

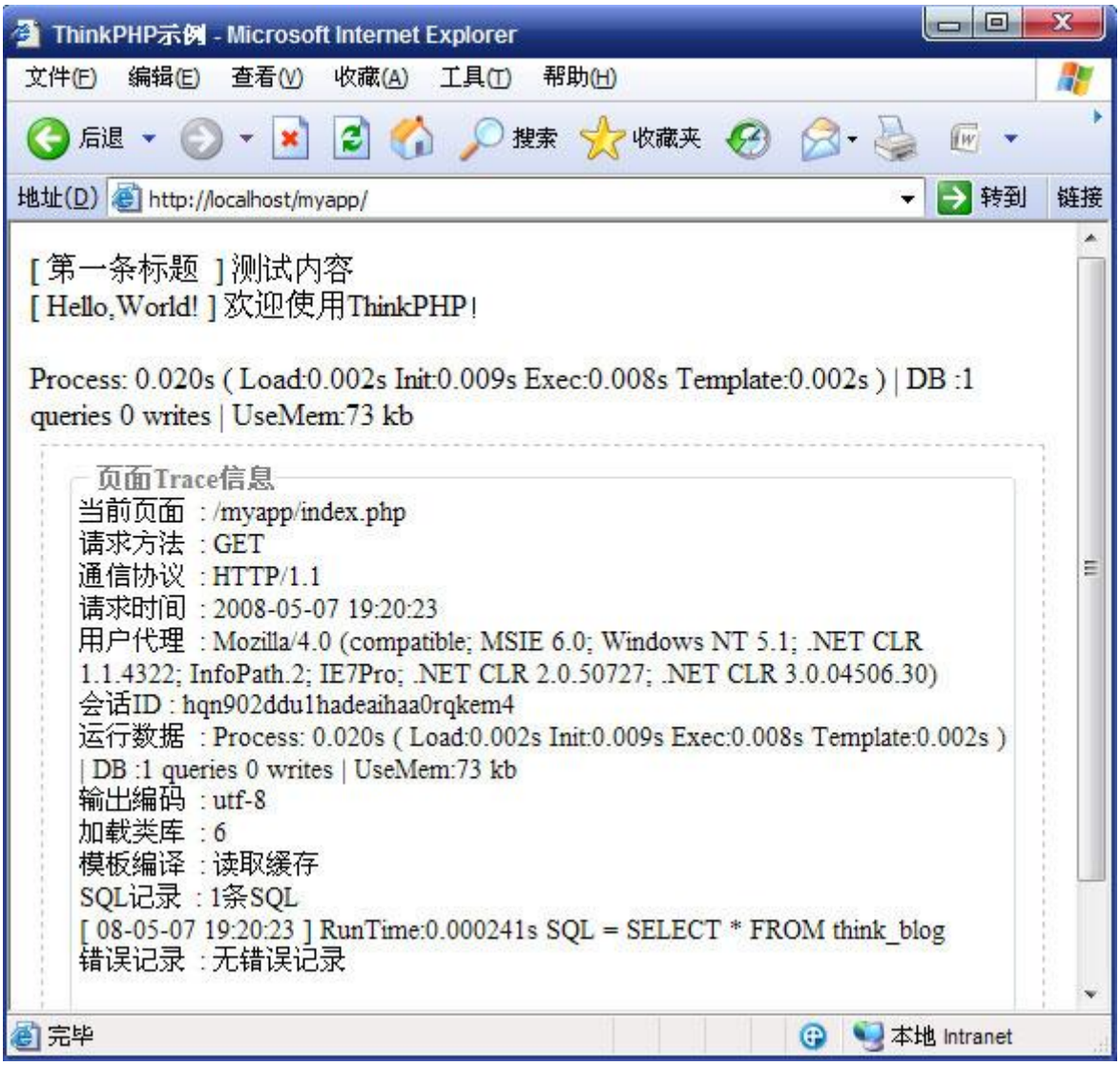
IndexAction 类的 index 方法里面使用

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

输出页面的时候，会自动读取 Tpl\default\Index\index.html 模板文件来输出。

现在我们再次在浏览器里面输入 http://localhost/myApp/

就可以看到动态数据的输出了，注意看页面的 Title 文字的变化。



我们看到了页面输出了数据库的两条记录，并且下面还显示了很多的运行时间信息，这个是开启调试模式之后系统默认的显示信息，包括了页面执行时间、数据库操作次数、内存使用情况 当前加载的类库数量，以及页面执行的 SQL 语句和错误记录等等。这些信息是能够给开发调试带来很大的帮助，而且页面 Trace 信息是可以定制显示的，你现在看到的是系统默认的显示项目。

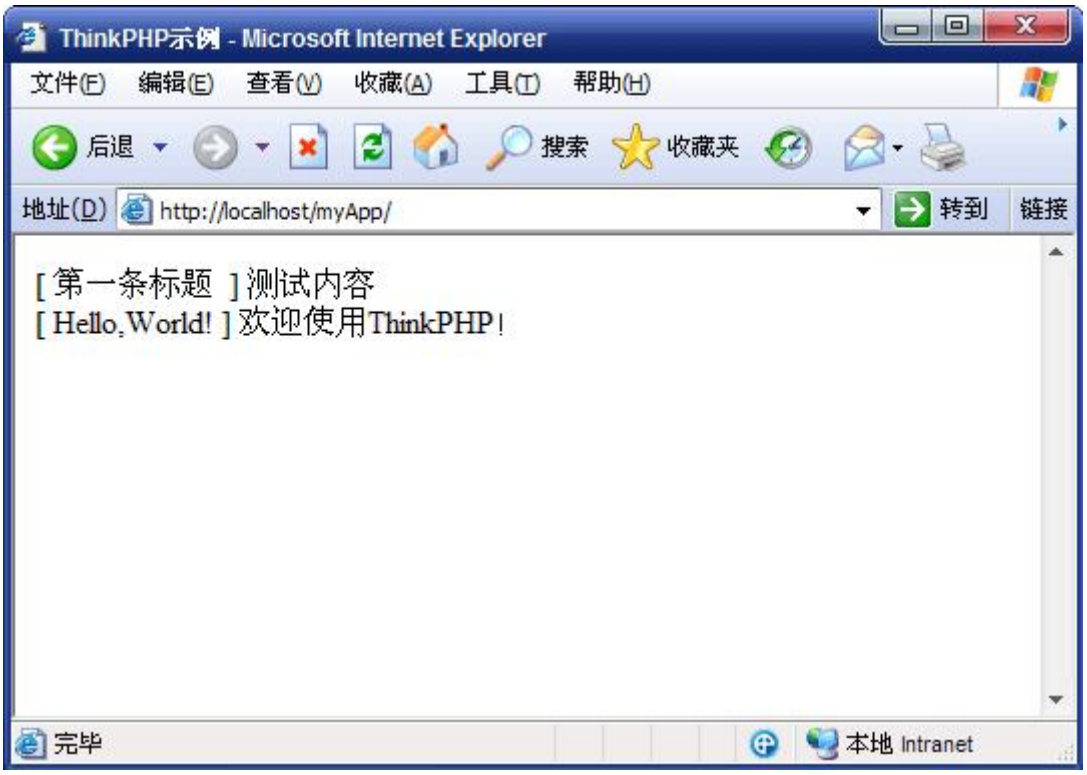
OK，假设我们项目开发完成了，需要正式部署到服务器了，建议关闭调试模式，配置文件中原来的

```
// 开启调试模式
'DEBUG_MODE'=>true,
```

修改为

```
// 关闭调试模式
'DEBUG_MODE'=>false,
```

现在，我们重新刷新下浏览器，就会看到下面的页面了，显然干净了很多。



到此为止，我们已经完成了一个简单的数据库读取的例子，也带您领略了使用 ThinkPHP 开发的大致过程。当然，ThinkPHP 的殿堂还有更多更好的功能在等着你，再次希望您的 ThinkPHP 学习之旅顺利、工作顺利！