Leask Huang / 黄思夏 / www.leaskh.com

Yii 框架快速入门

Version 3.9

注意！本教程中的实例只适合于PHP5.3或以上，PHP5.2可能会有细节上的差异。

1 Yii是什么？

什么是Yii，Yii的官方解释可以概括为以下几点：

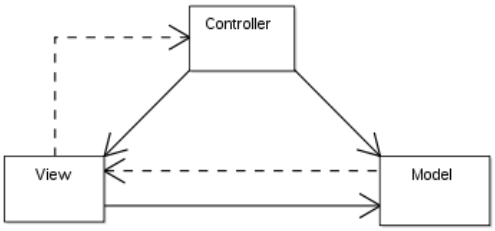
1. Yii是一个高性能、组件化、面向大型Web应用的PHP开发框架。
2. Yii的设计完全面向对象，基于完整的MVC编程思想；
3. Yii的发音类似于英文Yee[ji:]，或者汉语普通话的"易"；
4. Yii是开源软件，基于BSD许可发布。

更多信息可参考Yii官方网站： http://www.yiiframework.com/

2 MVC是什么？

上文中提到MVC，到底什么是MVC呢：

```
MVC -> M-V-C -> Model-View-Controller -> 模型-视图-控制器
```



MVC是一种重要的编程思想，最初由挪威计算机科学家Trygve Reenskaug于1979提出，无论是B/S还是C/S模式的开发都能从中受益。

MVC思想带来以下几点比较突出的优势：

1. 数据结构得以独立，使得部署应用的时候便于处理不同平台的文件系统，数据库系统等的差异；
2. 视图呈现得以独立，便于把同质的应用部署在不同的平台上，例如一个Web应用可以通过桌面浏览器、掌上设备或其他客户端软件访问，而不需要重写大部分的程序；
3. 大型复杂的算法和逻辑得以独立，有利于最大限度的代码重用，使代码重用不单是项目内的重用，甚至能实现跨项目的重用；

然而正因为有上文的第3点优势，才使得通过Yii、CodeIgniter、Kohana、Zend等开源框架能大大加快PHP应用的开发，因本质上我们是通过重用框架的"类"和"方法"以提高效率的。

MVC思想还大大有利于团队协作和敏捷开发。

想了解更多，可参阅维基百科： http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller 。

3 了解Yii

在开始折腾Yii之前，我们首先需要理清一些Yii的基本概念：

3.1 入口脚本 / Entry Script

Yii框架需要项目根目录下有一个index.php来响应所有用户的所有请求。

这个index.php主要有两个作用：

1. 负责加载Yii框架，Yii框架被加载后，根据入口脚本（index.php）中指定的配置文件来创建应用程序实例；
2. 负责传递用户的请求，包括"路由请求"和"GET/POST请求"到Yii应用实例的控制器中，控制器根据请求调用相应的业务逻辑作出响应；

入口脚本可以自己编写，也可以通过Yii命令行（CLI）工具，也就是Yii Shell在创建项目的时候自动建立。

以下是一个典型的入口脚本：

```php
<?php

// change the following paths if necessary
$yii=dirname(__FILE__).'/../framework/yii.php';
$config=dirname(__FILE__).'/../protected/config/main.php';

// remove the following lines when in production mode
defined('YII_DEBUG') or define('YII_DEBUG',true);
defined('YII_TRACE_LEVEL') or define('YII_TRACE_LEVEL',3);

require_once($yii);
Yii::createWebApplication($config)->run();
```

下面我简单解释一下这个入后脚本：

- 第4行：$yii字符串用于指定Yii框架所在的位置，这个字符串必须定位到Yii框架的“yii.php”文件；
- 第5行：$config字符串用于指定应用程序的配置文件，yii会根据配置文件的参数来初始化应用程序；
- 第8、9行：配置Yii应用实例的调试功能，这两行代码在实际部署应用的时候应该注释之或者直接把其删除，否则程序出错的输出可能招致安全隐患；
- 第11行，载入Yii框架；
- 第12行，创建Yii应用实例并初始化。

3.2 应用程序 / Application

“应用程序”是Yii区别于其他PHP MVC框架的特性之一，其主要任务是负责根据不同的配置文件和用户请求来实例化控制器。

严格意义上来讲，这是把M-V-C中的C（Controller）再细分为两层，因此“应用程序”也叫做“front-fontroller”（前端控制器）。

在Yii框架中的任何地方都可以通过：

```
Yii::app()
```

访问“应用程序”。

当一个应用程序的功能比较多，逻辑比较复杂，在必要时还可以把“应用程序”分割为若干的应用程序组件（Application Component）。这是比较进阶的技巧，大家可以稍后再详细了解。

3.3 控制器 / Controller

控制器是整个应用的核心，负责管理着整个项目绝大部分的功能逻辑。

控制器接收来自入口脚本的用户请求，分析这些请求并调用不同的方法进行处理，最后交由“视图”把处理结果返回给用户。

我们的开发任务绝大部分将在控制器中编程完成。

事实上，在控制器中您可以用原始的PHP语言来实现各种功能逻辑，但适当调用Yii框架内置的常用对象更能够大大提高开发效率。

3.4 模型 / Model

模型就是MVC中的M（Model），结合到Web开发中，主要指的是数据的抽象层。

Yii的“模型”主要具象化为以下两个部分：

- CActiveRecord（AR）：用于抽象化数据库访问；
- CFromModel：用于处理用户表单。

3.5 视图 / View

“视图”是比较好理解的，在控制器处理逻辑结束后，一般情况下会交由视图把处理结果呈现给用户。

这里的呈现是多种形式的，包括生成网页返回给浏览器，或提供Feed或各种API的形式输出。

视图主要包含的是HTML、CSS、Javascript、XML、图片或其他多媒体内容等。

有些情况下我们可能需要把PHP当作模板语言使用，主要是能够便于通过循环输出生成复杂的视图。

所以，视图文件中事实上是可以包含PHP代码的，但是Yii框架建议只把负责控制输出的PHP代码写在视图文件中，而其他实现功能逻辑的PHP代码尽量写在控制器中。

至于如何渲染视图以及视图中如何访问控制器中的变量等问题下文将详细讲解。

3.6 组件 / Component

组件是Yii框架内部的逻辑单位，组件有自己的属性、事件和行为，Yii框架由若干个组件封装而成。

这里提出组件仅仅是让大家明确一下Yii的一些概念，添加和修改组件是扩展Yii的高级技巧，暂时可以不理会。

3.7 模块 / Module

模块就像一个小型的Application（应用程序）。

模块封装在独立的一个文件夹中，包含自己独立的MVC分层结构，在执行逻辑上也和Application十分类似。

“模块”和“应用程序”的主要区别是，“应用程序”可以单独部署，“模块”不可以单独部署（必须依附在一个“应用程序中”）。

在大型应用的开发中，通常根据功能需求把应用分割为若干个模块，由不同的开发人员对其进行开发和维护。

通过细分模块也为代码重用带来便利。

下面来看看一个典型的模块的“骨架结构”：

```
forum/
    |-ForumModule.php                -> 模块的类文件
    |-components/                    -> 组件文件夹
    |      |-views/
    |-controllers/                   -> 控制器文件夹（C）
    |      |-DefaultController.php
    |-extensions/                    -> 扩展文件夹
    |-models/                        -> 模型文件夹（M）
    |-views/                         -> 视图文件夹（V）
          |-layouts/
          |-default/
          |-index.php
```

至于如何创建模块，将在下文详细讲解。

3.8 路径别名 / Path Alias

路径别名用于快速定位资源，Yii中别名的使用是很广泛的。

例如你可以这样指向一个Class：

```
1 RootAlias.path.to.target
```

这里的RootAlias（根别名）一共有6个被预定义：

```
system          -> 指向 Yii 框架目录;
zii             -> 指向zii库目录；
application     -> 指向应用程序（base directory）；
webroot         -> 指向入口脚本所在的目录；
ext             -> 指向扩展目录（该目录包含所有第三方扩展）;
root            -> 指向module所在的目录（该别名仅限在module内可用）。
```

任何时候都可以通过“YiiBase::getPathOfAlias()”把别名转换为完整的路径，如：

```
1 YiiBase::getPathOfAlias('system.web.CController')
```

将返回：

```
1 'yii/framework/web/CController'
```

反之“YiiBase::setPathOfAlias()”可以设置指定路径为一个新的别名。

了解别名后，我们就可以用别名来导入一个预定义的类了。

```
1 Yii::import('system.web.CController');
```

您甚至可以使用通配符，如：

```
1 Yii::import('system.web.*');
```

的方式实现批量导入。

为什么用“Yii::import()”而不直接用PHP的“include”和“require”语句呢？

事实上由于Yii智能的导入机制，使用“Yii::import()”导入的类直到其真正需要调用之前，是没有被“包含”的。

当这个被“Yii::import()”导入的类真正需要被用到的时候，Yii才会载入它。这使得你可以一次性导入很多的类，而不需要担心额外的资源消耗。这一个做法显然比PHP的“include”和“require”语句效率高得多。同样的，多次导入同一个命名空间要比“include_once”和“require_once”快很多。

类似地，不单是“Yii::import()”，如“Yii::createComponent()”等很多Yii内置的方法，都支持通过别名来指定资源，而不需要指定完整的路径。

【TIPS】当调用一个通过Yii框架定义的类时，我们不必导入或者包含它。所有的Yii核心类都是被预定义的。

3.9 名称空间 / Name Spaces

Yii的官方教材（Guide）中，把“名称空间”与上文的“路径别名”放在同一节叙述。对于基础较为薄弱的同学比较不好理解，因此，这里单独谈谈名称空间。

大型软件项目常会出现函数名或类名重复的问题，为了解决这个问题，很多编程语言都引入了“名称空间”的概念。

通过名称空间来规范一个函数名或类名的作用范围，是各编程语言的发展趋势，PHP从5.3.0版本开始也引入了此特性。

具体到Yii框架中，“名称空间”的名称和“路径别名”的名称是相同的，这个名称既可以用来指定一个“类”文件，也可以用在指定该“类”所包含的“名称空间”。

但是注意不要把路径别名和命名空间在意义上混淆了：

- “名称空间”调用了一些类名的逻辑分组以便他们可以同其他类名区分开，即使他们的名称是一样的；

- “路径别名”则是用来引用类文件或者目录的。

所以路径别名和名称空间并不冲突，即使他们写法上类似。

【注意！】因为PHP 5.3.0以前的版本并不支持名称空间，所以并不能创建两个有着同样名称但是不同定义的类的实例。为此，所有Yii框架的系统类都以字母“C”（代表“Class”）为前缀，以避免与用户自定义类冲突。我们推荐为Yii框架保留“C”字母前缀的唯一使用权，用户自定义类则可以使用其他字母作为前缀，以保持整体代码的一致性和稳定性。

3.10 约定俗成 / Conventions

Yii主张开发过程中遵守约定俗成的惯例，这样的好处是可以通过Yii开发复杂的应用程序而不需要重新制定和维护复杂的配置文件。

这些约定俗成的惯例并不是必要的，事实上Yii绝大多数单元部件的特性，都可以通过手动配置来修改之。

但是遵守惯例最少能带来以下几点优势：

1. 便于团队协作，让其他团队成员更容易看懂和维护您的代码；
2. 在升级Yii框架版本的过程中，更便于实现平滑过渡；
3. 更容易实现代码块的迁移和重用。

那么Yii中有哪些常用的惯例呢？

3.10.1 URL惯例

默认情况下，Yii能识别如下的URL格式：

```
http://hostname/index.php?r=ControllerID/ActionID
```

Yii解释GET方法请求中的变量r的值，然后路由到相应的控制器和动作中。

如果“ActionID”没有指定，那么默认动作就会被激活，按照惯例，Yii中默认的动作是“CController::defaultAction”;

类似地如果“ControllerID”没有指定，那么默认控制器就会被激活，按照惯例，Yii中默认的控制器是“CWebApplication::defaultController”。

当然，可以修改“CUrlManager”中的配置，以实现更复杂的和个性化的URL路由特性。

3.10.2 代码惯例

Yii推荐在命名变量、函数和类的时候使用驼峰式（Camel Case）风格，且必须遵守以下规则：

1. 变量和函数名称的首字母小写；
2. 类名称首字母大写,
3. 后续的字符中每个单词的首字母大写；
4. 私有类成员变量添加下划线前缀；
5. 为避免Yii框架的类和用户自定义类发生冲突，所有Yii框架类都以“C”（大写字母C）作为前缀，用户自定义类应避免使用“C”作为前缀。
6. 控制器类名的特别规则是，他们必须附上“Controller”后缀，如“ExampleController”，同时相应地该类的ID将自动识别为“example”（类ID的首字母将自动识别为小写字母），此规则使代码更安全，也使URL更简洁（可用“/index.php?r=example/index”替代较繁琐的“/index.php?r=ExampleController/index”）。

如：

```
1 $iAmAVar              // 普通变量
2 iAmAFunction()        // 普通函数
3 IAmAClass             // 普通类
4 CIAmAClassFromYii     // Yii框架类
5 $_iAmAPrivateVar      // 私有类成员变量
6 ExampleController     // 控制器类（其ID自动识别为"example"）
```

3.10.3 配置惯例

配置应为数组关键值对，每个键代表对象名称属性的配置，每个值代表相应属性的初始值，如：

```
1 array(
2     'nameA'=>'valueA',
3     'nameB'=>'valueA'
4 )
```

3.10.4 文件命名惯例

文件命名的管理会根据文件的类型不同而有所差异。

- 公有类的类文件需要根据其包含的公有类（Public Class，即可被其他任何类使用的类）来命名，如：“CController”类应该保存在“CController.php”文件内；
- 私有类（Private Class，仅能被唯一公有类使用的类）应该保存在其依附的公有类的类文件内；
- 视图文件需要根据其包含的视图名称来命名，如：“index”视图应该保存在“index.php”文件内；
- 配置（Configuration）文件的命名相对比较自由，您可以根据自己的喜好来对其命名。

3.10.5 文件系统惯例

- WebRoot/protected：此文件夹包含所有程序文件和资源文件，因此，出于安全理由，应该限制用户通过Web服务器直接访问这个文件夹以及其所有子文件夹；
- WebRoot/protected/runtime：这个文件夹包含所有运行时创建的临时文件，这个文件夹必须给予Web服务器进程以可写入的权限，这个文件夹的路径可通过“CApplication::runtimePath”配置；
- WebRoot/protected/extensions：此文件夹包含所有的第三方扩展，可通过“CApplication::extensionPath”配置；
- WebRoot/protected/modules：此文件夹包含所有的模块，每个模块占用一个独立的子文件夹；
- WebRoot/protected/controllers: 此文件夹包含所有的控制器文件，可通过“CWebApplication::controllerPath”配置；
- WebRoot/protected/views：此文件夹包含所有的视图文件，包括视图文件、布局文件以及系统视图文件，可通过“CWebApplication::viewPath”配置；
- WebRoot/protected/views/ControllerID：此文件夹包含某一ID为ControllerID的控制器的视图文件，可通过“CController::viewPath”配置；
- WebRoot/protected/views/layouts：此文件夹包含所有的布局文件，可通过“CWebApplication::layoutPath”配置；

- WebRoot/protected/views/system：此文件夹包含所有的系统视图文件，这些视图文件用作格式化显示系统的意外信息以及错误信息，可通过“CWebApplication::systemViewPath”配置；
- WebRoot/assets：此文件夹用于管理发布在Web上的文件，这些文件将直接发布在Web服务器上，供Web访客访问，这个文件夹必须给予Web服务器进程以可写入的权限，可通过“CAssetManager::basePath”配置；
- WebRoot/themes：此文件夹包含可应用在应用程序上的主题文件，每一个主题独占一个与该主题同名的子文件夹，可通过“CThemeManager::basePath”配置。

3.10.6 数据库惯例

大多数Web应用需要使用数据库，为了获得最好的兼容性，同时也增加代码的易读性，我们推荐大家在使用数据库的时候遵守以下惯例：

- 所有的数据表名，字段名（列名）均以小写字母命名；
- 数据表名和字段名的各个单词应该用下划线来分隔，如“product_table”;
- 在命名表明的时候，您可以使用单数或者复数形式，如“user”或“users”均可，但是注意要保持统一，不要混用，为了简单起见，我们推荐统一使用单数形式的表名；
- 数据表在命名的时候应该考虑加上如“tbl_”这样的常规前缀，当应用程序需要访问几个数据库且数据库中有相同表名的时候就可以通过不同的前缀来区分，让代码更容易被理解。

【注意！】以上使用数据库的惯例在Yii的开发中并不是必须的，Yii推荐使用这些惯例仅为了最大限度提高数据库的兼容性和代码的易读性。

3.11 开发流程 / Development Workflow

基于Yii框架开发应用程序，一般可以参考以下流程进行：

1. 创建Yii应用程序骨架，这一步大家可以自己动手，也可以通过Yii命令行工具yiic在瞬间完成；
2. 配置应用程序，这一步一般可以通过修改配置文件来完成，当然，对于比较复杂的应用程序，可能还需要编写某些组件的代码（如用户组件）；
3. 为每一种数据创建数据模型以便对其管理，你也可以通过yiic自动根据数据库配置为某些数据表创建active record类，然后通过修改定制这些类以完成此步骤；
4. 为每一种用户请求创建相应的控制器来响应之，如何编写控制器取决于具体的功能需求，一般情况下如果某个模块的数据需要被用户访问，那么应该有相应的控制器，yiic能为您生成常规的控制器骨架代码（这些骨架代码能够完成基本的CRUD功能）；// CRUD指对数据库的四种基本操作“Creat-Retrieve-Update-Delete”
5. 完善控制器的动作以及制作视图文件，这是整个开发工作中真正需要话时间和精力的地方；
6. 为控制器类建立必要的过滤器，例如那些动作需要哪些权限级别的用户才能执行；
7. 如果该应用有主题功能的需求还需要创建主题；
8. 如果应用有国际化需求还需要创建翻译文本；
9. 如果必要，考虑使用缓存技术对数据或视图文件进行缓存处理，提高应用的执行效率；
10. 完善细节，准备部署；

4 安装Yii

4.1 获得Yii

获得Yii最便捷的方法就是到Yii的官方网站去下载： http://www.yiiframework.com/download/

此文成文的时候Yii框架的最新版本是1.1.2，发布日期是2010年5月2日，此版本可以通过以下地址直接下载：

- .tar.gz 方式压缩（适合Unix、Linux等系统）：
  http://www.yiiframework.com/files/yii-1.1.2.tar.gz
- .zip 方式压缩（适合Windows系统）：
  http://www.yiiframework.com/files/yii-1.1.2.zip
- 本地高速下载：
  http://192.168.1.247/shared/yii-1.1.2.r2086p.zip

4.2 安装Yii

严格来说Yii并不需要安装，把Yii随便解压在电脑的任何地方，并不需要是Web服务器的发布文件夹内，都可以顺利创建Yii应用。

但是为了确保服务器满足Yii所要求的运行环境，我们需要借助Yii内置的环境检测功能对服务器进行简单检测。

我们计划建立“SaaS测试项目”到“用户文件夹”内的“SaaS目录”下，步骤如下：

1. 部署好PHP开发环境，通常包含：Apache + PHP + MySQL；
2. 创建“SaaS”文件夹到你的“用户文件夹”中；
3. 创建“sources”文件夹到“SaaS”文件夹中；
4. 解压下载得到的Yii框架到任意文件夹，复制解压得到的“framework”到“SaaS”目录中（Linux 下 “ln -s” 亦可）；
5. 复制解压得到的“requirements”文件夹到“sources”文件夹中（Linux 下 “ln -s” 亦可）；
6. 把Apache的DocumentRoot配置为刚刚创建的“sources”文件夹，如：

```
DocumentRoot "/path/to/SaaS/sources"
```

一切准备就绪后，打开浏览器，访问：

http://localhost/requirements/index.php

如果顺利，你应该看到一个如下的页面：

Yii Requirement Checker

Description

This script checks if your server configuration meets the requirements for running Yii Web applications. It checks if the server is running the right version of PHP, if appropriate PHP extensions have been loaded, and if php.ini file settings are correct.

Conclusion

Your server configuration satisfies the minimum requirements by Yii. Please pay attention to the warnings listed below if your application will use the corresponding features.

Details

Name	Result	Required By	Memo
PHP version	Passed	Yii Framework	PHP 5.1.0 or higher is required.
$_SERVER variable	Passed	Yii Framework	
Reflection extension	Passed	Yii Framework	
PCRE extension	Passed	Yii Framework	
SPL extension	Passed	Yii Framework	
DOM extension	Passed	CWsdlGenerator	
PDO extension	Passed	All DB-related classes	
PDO SQLite extension	Passed	All DB-related classes	This is required if you are using SQLite database.
PDO MySQL extension	Passed	All DB-related classes	This is required if you are using MySQL database.
PDO PostgreSQL extension	Warning	All DB-related classes	This is required if you are using PostgreSQL database.
Memcache extension	Warning	CMemCache	
APC extension	Warning	CApcCache	
Mcrypt extension	Passed	CSecurityManager	This is required by encrypt and decrypt methods.
SOAP extension	Passed	CWebService, CWebServiceAction	
GD extension	Passed	CCaptchaAction	

passed failed warning

Apache/2.2.15 (Debian) Yii Framework/1.1.2 2010-05-12 10:09

简单说明：

- 其中凡Required By标记为Yii Frameworks的都是必须要满足的，否则框架将无法正常运行；
- 数据库扩展中PDO extension是使用Yii DAO和AR（Active Record）所必需的，Yii DAO和AR为Yii框架提供数据库访问功能，所以请务必满足这一条件；
- 数据库扩展中PDO SQLite extension、PDO MySQL extension、PDO PostgreSQL extension这三项并不需要同时满足，根据具体的需求进行配置即可；
- Mencache extension和APC extension都是缓存插件，在需要通过缓存来加速项目的运行速度时才需要配置；
- DOM extension、Mcrypt extension、SOAP extension和GD extension都是Yii框架实现某些关键功能的时候所需要的，严格来说不算是必须，但是最好都具备他们，否则框架可能会出现某些重大的功能缺失。

5 创建第一个Yii项目

上文我们已经配置好"DocumentRoot "/path/to/SaaS/sources""以及已经解压Yii框架（"framework"文件夹）到"/path/to/SaaS/framework"文件夹下。

下文的实验都将在这个环境下进行，如果您需要测试文中的命令或代码，请修改其中的路径为您服务器上正确的配置值。

5.1 认识脚手架

Yii框架集成一个强大而实用的命令行工具yiic（Yii Console），yiic主要是便于快速创建应用程序或者某一功能模块的骨架（骨架包括必要的文件夹结构和基本的示例代码），提高项目开发初期的工作效率，因此，yiic在官方教材中也被形象地称为"Scaffolding"（脚手架）。

其实yiic并非仅此而已，yiic基于PHP实现，所以在任何安装了PHP CLI（PHP命令行执行模式）的系统都能使用。

yiic和Yii框架（PHP Web框架）共享着一个叫做YiiBase的类，YiiBase包含着Yii框架中大部分的基础逻辑，因此你其实可以基于yiic做开发，甚至可以在yiic中通过MVC思想构建复杂的命令行应用程序，步骤上和开发基于Yii的Web应用十分类似。

所以，在项目开发的中后期，巧妙扩展yiic将对调试和部署Yii应用带来极大的便利。

5.2 使用脚手架创建一个新的Yii应用骨架

5.2.1 在Windows系统中创建

进入Command或者Power Shell，运行如下命令：

```
X:\path\to\SaaS\framework\yiic webapp X:\path\to\SaaS\sources
```

如果您的命令行“PATH”环境变量中没有添加“php.exe”所在的路径，你可能需要运行如下命令：

```
C:\Progra~1\PHP\php.exe X:\path\to\SaaS\framework\yiic.php webapp X:\path\to\SaaS\sources
```

为了方便工作，建议把php.exe添加到“PHTH变量中”，下文的命令均以第一种风格输入，也就是假定你已经配置好该变量，方法可参考：本地测试环境的搭建。

5.2.2 在Unix或Linux的Bash Shell环境中创建

```
$ /path/to/SaaS/framework/yiic webapp /path/to/SaaS/sources
```

以上三个命令的效果是等价的，程序会询问你是否需要创建一个新的Web应用到“sources”文件夹，回答“yes”，程序将自动生成该文件夹，并创建一个新的应用程序骨架在其中。

5.2.3 移动“protected”文件夹并更改“入口脚本”

出于安全性考虑，我们移动“protected”到“DocumentRoot之外”，直接放在“SaaS”文件夹下（也就是把“protected”移动到上一级文件夹中）。

最终的项目文件夹结构如下：

```
SaaS/
    |-framework/
    |-protected/
    |-sources/      -> DocumentRoot
```

【TIPS】这样做在程序开发的角度看是不必要的，但是这样杜绝了网站访客直接访问“protected”文件夹，有利于加强了项目的安全系数。

由于移动了“protected”文件夹，我们还需要在入口脚本里做一点小修改，把：

```
1 $config=dirname(__FILE__).'/protected/config/main.php';
```

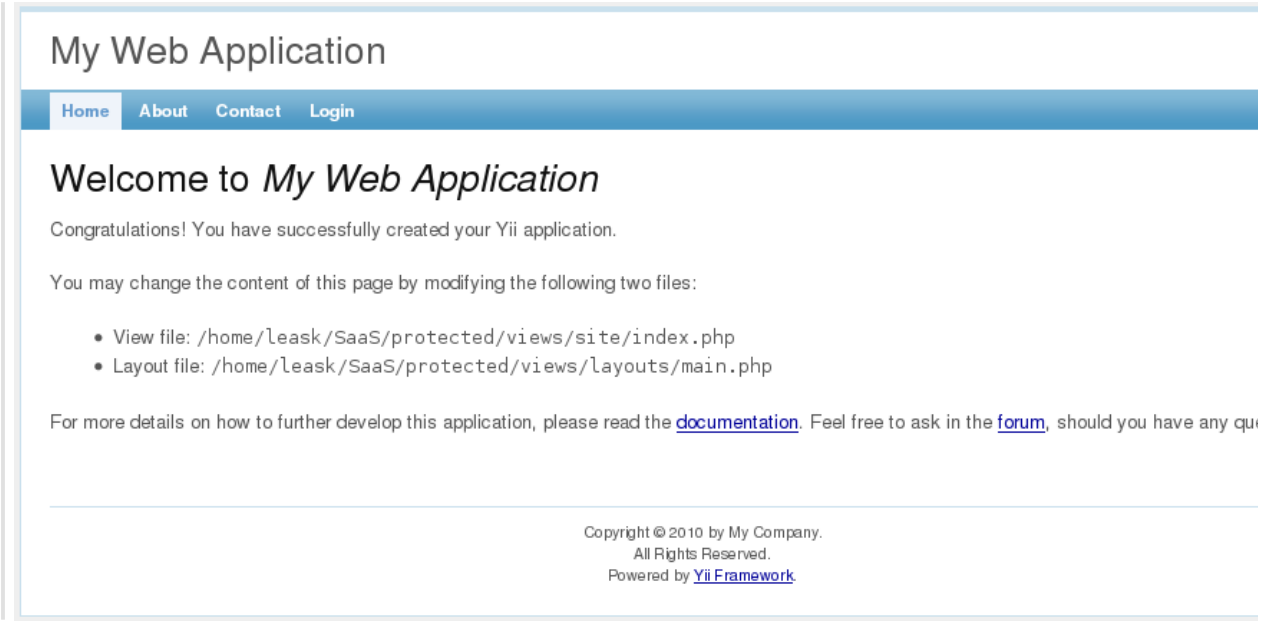
改为：

```
1 $config=dirname(__FILE__).'/../protected/config/main.php';
```

可以通过浏览器访问如下地址进行测试：

http://localhost/index.php

如果您可以看到如下的页面，说明您的Yii应用已经创建成功了：

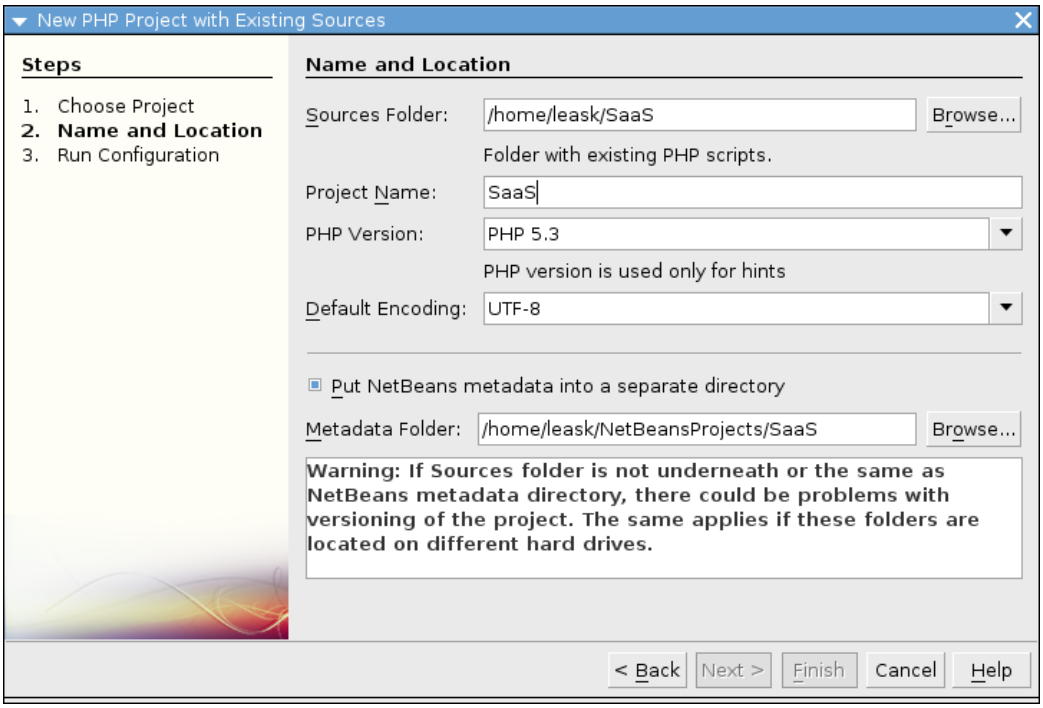


5.3 Hello World

在本小节中，我们将创建一个新的控制器和一个新的视图文件，并用新的控制器和视图文件显示“Hello World”。

5.3.1 准备工作一：

首先打开NetBeans，根据下图的配置创建新项目：



NetBeans的简介，配置、安装和创建新项目的教程可根据需要参考以下的文章：

- NetBeans 基本说明（待完成）
- Linux 下 NetBeans 配置与使用简介（待完成）
- Windows 下 NetBeans 配置与使用简介（已完成）

5.3.2 准备工作二：

对项目启用版本管理。

启用版本管理来跟踪代码的更改不是必须，但推荐养成版本管理的习惯，在日后代码出现问题，才能够从容处理。

启用版本管理功能需要先初始化代码库，如何初始化代码库请参考以下的文章：

- 版本控制的概念、分布式版本控制与 git 简介
- Linux 下 git 配置与使用指南
- Windows 下 git 配置与使用指南
- git 进阶功能

5.3.3 第一步：创建控制器

在“protected/controllers”目录下创建“HelloworldController.php”，并编辑如下代码：

```
<?php

class HelloworldController extends Controller
{
    public $strToDisplayByPull = 'Hello';

    public function actionIndex()
    {
        $this->renderPartial('index',array(
            'strToDisplayByPush'=>'World'
        ));
    }
}
```

简单分析：

- 该例子中创建了一个名为“HelloworldController”的，继承自“Controller”的控制器类，该类的ID已被自动识别为“helloworld”（此规则可参见上文有关控制器概念的章节），该控制器类包含一个默认方法“actionIndex()”，用户访问该控制器的时候如果没有指定执行什么动作，那么这个默认动作将被调用；
- 为“HelloworldController”类定义一个公有属性“strToDisplayByPull”，并设置值为字符串'Hello'，接下来的教程中，将演示如何在视图中访问这个属性，进而将'Hello'字符串传递到视图中；
- “renderPartial()”方法同“render()”方法都是用来输出结果的，用法也完全一样，区别是前者不会加载视图的Layout（布局）文件，在实际使用中，用“render()”输出的场合比较多，“renderPartial()”则在Hello World、调试、输出为API等场合十分实用。
- 我们在“renderPartial()”方法中指定了两个参数，第一个是字符串“index”，这将告诉系统，使用“helloworld”控制器的“index”视图来渲染页面；
- 第二个参数是一个数组，数据里面的值将被传递到视图中，在视图里可以通过访问“$strToDisplay”变量得到所传递的'World'字符串。

【注意！】控制器文件的PHP代码以“”结尾，这是因为事实上“?>”在PHP编码规则中并不是必须的，在HTML中嵌入PHP程序时加入“?>”仅为了让解释程序区分PHP代码与HTML代码，所以在纯PHP程序中，并不需要以“?>”结尾，而且由于“?>”后面的内容将不经解释而被直接输出，因此容易造成错误输出额外的空格，以及其他字符。正因为这样，在如Zend这样要求代码严谨的框架中，是禁止添加“?>”结尾的。Yii中并没有强制要求，但是为了输出精确，我们推荐类文件中的代码不以“?>”结尾。

5.3.4 第二步：创建视图

在“protected/views”目录下创建“helloworld”文件夹，这个文件夹将用来存储所有ID为“helloworld”的控制器要用到的所有视图文件；

在“protected/views/helloworld”目录下创建“index.php”文件，并编辑如下代码：

```
<?php
    echo $this->strToDisplayByPull.' '.$strToDisplayByPush;
?>
```

简单分析：

- $this->strToDisplayByPull返回字符串'Hello'，演示通过Pull方法直接访问调用该视图的控制器的公有属性的值；
- $strToDisplayByPush的值等于字符串'World'，演示通过Push方法得到“renderPartial()”传递过来的字符串值；

Pull和Push方法在Yii开发中都很常用，他们都能实现在控制器中往视图传递变量，至于那种情况下用哪一种，需要具体情况具体分析。

5.4 阶段小结

在浏览器访问：

http://localhost/index.php?r=helloworld

如无意外您应该看到“Hello World”了。

这一章是Yii入门教程中最重要的一章，在这一章中我们从创建应用程序骨架，到实现第一个“Hello World”。不管你是顺顺利利还是经历无数艰难曲折，那都是可喜可贺的。

这一章不同于上一章，上一章我基本上按照官方Guide教程的模式，叙述Yii基本概念中的精华部分，这一章我自己构思了一个“Hello World”例子，因为这正是官方教材所缺失的，官方Guide仅以创建应用骨架作为“Hello World”教程，而且官方教材只包含Push方法传递变量，对Pull方法仅一笔带过，这样对初学者难免造成困惑。

相信至此大家对基于Yii进行开发已经少有心得，在下面的教程中，我将针对常见的开发任务，使用表单、访问数据库配置URL路由等方面继续深入Yii。

6 从Yii中使用表单

Yii内置的表单功能是很强大的，从生成表单到收集用户输入都有专门的类来完成。这里举一个例子，模拟一个用户反馈表单：

6.1 第一步：建立表单模型

在“protected/models”目录下创建“FeedbackForm.php”，并编辑如下代码：

```
<?php

class FeedbackForm extends CFormModel
{
    public $name;
    public $email;
    public $subject;
    public $body;

    public function rules()
    {
        return array(
            // name, email, subject and body are required
            array('name, email, subject, body', 'required'),
            // email has to be a valid email address
            array('email', 'email'),
            // verifyCode needs to be entered correctly
            );
    }

}
```

这样就新建一个表单模型了。

【注意！】代码中的“rules()”方法用来检查数据的有效性，Yii在接收用户输入的数据之前，会先检查数据的有效性，如果此方法缺失，将导致无法确保数据是有效的，Yii将拒绝使用这些数据。因此请务必在建立模型的时候加入这个方法，如果您通过“脚手架”创建模型，类似的验证代码将自动被创建。

【TIPS】

- Yii在验证数据有效性这方面提供了很多实用的参数，如“校验码”等，具体可以参考这个地址： http://www.yiiframework.com/doc/api/CModel#rules-detail ；
- Yii还提供一个“Validators Cheatsheet”帮助大家快速创建校验规则，这个文件在本文附录处有提供下载地址。

6.2 第二步：建立反馈表单视图

在“protected/views/helloworld”目录下创建“feedback.php”，并编辑如下代码：

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<div>

<?php $form=$this->beginWidget('CActiveForm'); ?>
```

```
 6     <div>
 7         <?php echo $form->labelEx($model,'姓名'); ?>
 8         <?php echo $form->textField($model,'name'); ?>
 9     </div>
10
11     <div>
12         <?php echo $form->labelEx($model,'E-mail'); ?>
13         <?php echo $form->textField($model,'email'); ?>
14     </div>
15
16     <div>
17         <?php echo $form->labelEx($model,'主题'); ?>
18         <?php echo $form->textField($model,'subject',array('size'=>60,'maxlength'=>128)); ?>
19     </div>
20
21     <div>
22         <?php echo $form->labelEx($model,'内容'); ?>
23         <?php echo $form->textArea($model,'body',array('rows'=>6, 'cols'=>50)); ?>
24     </div>
25
26     <div>
27         <?php echo CHtml::submitButton('提交'); ?>
28     </div>
29 <?php $this->endWidget(); ?>
30
31 </div>
```

简单分析：

- 第一行不要漏掉哦，要不然中文会出现乱码；
- “$form=$this->beginWidget('CActiveForm');”到“$this->endWidget();”之间是表单的内容，在输出这个视图的时候，他们将输出为“<form *>”和“</form>”；
- “echo $form->labelEx($model,'姓名');”此代码用于输出一个Label标签；
- “echo $form->textField($model,'name');”此代码用于生成一个文本框，
- “echo $form->textField($model,'subject',array('size'=>60,'maxlength'=>128));”此代码用于生成指定尺寸的文本域；
- “echo CHtml::submitButton('Submit');”此代码用于生成一个提交按钮。

表单视图的源代码比较简单，有HTML基础的话一下子应该就能理解了。

值得一提的是，“$form->***”方法还能生成如密码文本框、单选框、复选框等，和生成基本的文本框在用法上是一致的，因此就不啰嗦了，具体可参考： http://www.yiiframework.com/doc/api/CActiveForm

上面的地址还提到在“CActiveForm”中配置AJAX结合表单模型中的规则对用户输入的数据进行有效性检查的方法，这暂不属于入门教程探讨的范畴，但是在实际工作中经常会用到，建议大家在学习完本教程后务必要仔细研究一下。

6.3 第三步：建立反馈表单控制器

接下来回到我们在上一章中创建的“helloworld”控制器，根据以下Diff添加一个“actionFeedback”方法：

```
 1 <?php
 2
 3 class HelloworldController extends Controller
 4 {
 5     public $strToDisplayByPull = 'Hello';
 6
 7     public function actionIndex()
 8     {
 9         $this->renderPartial('index',array(
10             'strToDisplayByPush'=>'World'
11         ));
12     }
13
14 +   public function actionFeedback()
15 +   {
16 +       $model=new FeedbackForm;
17 +       $this->renderPartial('feedback',array('model'=>$model));
18 +   }
19
20 }
```

下面我们集中看看新添加的几行代码：

```
1 public function actionFeedback()
2 {
3     $model=new FeedbackForm;
4     $this->renderPartial('feedback',array('model'=>$model));
5 }
```

简单分析：

- 第三行：创建一个“FeedbackForm”类的实例，名为“model”；
- 第四行：将“model”传递到视图中，并输出“feedback”视图。

在浏览器中通过以下地址测试：

http://localhost/index.php?r=helloworld/feedback

如果看到下图，说明表单已经被成功创建了：

6.4 第四步：建立处理反馈表单的视图

在“protected/views/helloworld”目录下创建“feedbackshow.php”，并编辑如下代码：

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<div>

    <div>
        <?php echo '姓名: '; ?>
        <?php echo $model->name; ?>
    </div>

    <div>
        <?php echo 'E-mail: ' ?>
        <?php echo $model->email; ?>
    </div>

    <div>
        <?php echo '主题: ' ?>
        <?php echo $model->subject; ?>
    </div>

    <div>
        <?php echo '内容: ' ?>
        <?php echo $model->body; ?>
    </div>

</div>
```

这次用到的代码比较简单，就不多解释了。

6.5 第五步：建立反馈表单控制器

接下来我们重新回到“helloworld”控制器，根据以下Diff修改代码：

```
<?php

class HelloworldController extends Controller
{
    public $strToDisplayByPull = 'Hello';

    public function actionIndex()
    {
        $this->renderPartial('index',array(
            'strToDisplayByPush'=>'World'
        ));
    }

    public function actionFeedback()
    {
        $model=new FeedbackForm;
-       $this->renderPartial('feedback',array('model'=>$model));

+       if (isset($_POST['FeedbackForm'])) {
+           $model->attributes=$_POST['FeedbackForm'];
+           $this->renderPartial('feedbackshow',array('model'=>$model));
+       } else {
+           $this->renderPartial('feedback',array('model'=>$model));
+       }
+   }

}
```

聚焦到修改后的“actionFeedback()”方法：

```
public function actionFeedback()
{
    $model=new FeedbackForm;
    if (isset($_POST['FeedbackForm'])) {
        $model->attributes=$_POST['FeedbackForm'];
        $this->renderPartial('feedbackshow',array('model'=>$model));
    } else {
        $this->renderPartial('feedback',array('model'=>$model));
    }
```

```
10 }
```

这次的修改中，只需要好好理解以下这一句就可以了：

```
1 $model->attributes=$_POST['FeedbackForm'];
```

这种赋值方式是Yii 1.1.1中新加入的，也是Yii目前推荐的接收表单输入的方式，在使用该方式的时候请确保“FeedbackForm”这个Model中已经具备用于校验数据有效性的Rule，否则此赋值语句将无效（这一点前文已有解释）。

再次通过浏览器访问：

http://localhost/index.php?r=helloworld/feedback

随便输入一些测试数据，提交表单后服务器返回您填写的内容，则实验已顺利通过。

【TIPS】Yii中还有更强大的表单功能，可参阅： http://www.yiiframework.com/doc/guide/form.overview 。

7 从Yii中访问数据库

上文我们已经学会了如何构建用户表单，以及如何收集、校验和利用用户填写的数据，现在我们来看看如何把这些数据存入数据库，并以此为例介绍Yii中访问数据库的一些常识。

Yii提供两组访问数据库的API：

- Yii DAO：Yii Data Access Objects，基于PHP Data Objects（PDO）扩展建立，提供通过统一的接口访问各种常见数据库的能力，Yii DAO能实现所有类型的数据库操作；
- AR：Active Record，AR提供“更”面向对象的方法处理数据且无需书写SQL语句，是一种流行的对象关系映射（ORM / Object-Relational Mapping）技术，一个AR类代表一个数据模型（可以是数据表或表单），其字段作为AR的属性，一个AR实例代表在表中的一行，常见的CRUD操作对应AR中的成员函数，但AR只能实现对数据库进行常见的CRUD操作，不能实现所有类型的数据库操作。

【TIPS】由于AR经过高度的封装，其实际运行性能不如Yii DAO，且AR的支持受限于数据库管理系统，并不是所有的数据库都支持AR（如MySQL需要4.1或以上版本）。

下文将尝试通过Yii DAO访问MySQL数据库，因为Yii DAO更加“通用”、强大和高效，如愿进一步了解AR可参阅： http://www.yiiframework.com/doc/guide/database.ar 。

7.1 准备工作一：准备数据库和数据表

我们首先创建“/path/to/SaaS/protected/data/dbinit.sql”数据库脚本（SQL脚本并不一定要存放在该文件夹，放这里仅是Yii框架的习惯而已，事实上可根据喜好放置），用于创建表格结构：

```
1 CREATE TABLE tbl_feedback (
2     id INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,
3     fb_name VARCHAR(128) NOT NULL,
4     fb_email VARCHAR(128) NOT NULL,
5     fb_subject TEXT NOT NULL,
6     fb_body TEXT NOT NULL
7 );
```

我们推荐把建立数据表结构的命令存放在脚本中，这将便于日后在其他机器部署该应用的时候避免因为数据表结构的差异而影响应用程序的运行。

进入CLI，如下执行：

```
$ mysql -u root -p
Enter password: *******

mysql> create database saas character set utf8;
Query OK, 1 row affected (0.01 sec)

mysql> use saas;
Database changed

mysql> source /path/to/SaaS/protected/data/dbinit.sql
Query OK, 0 rows affected (0.29 sec)

mysql> grant all privileges on saas.* to saas_user@localhost identified by 'saas_password';
Query OK, 0 rows affected
```

完成后我们就创建了一个名为“saas”的新数据库，一个名为“tbl_feedback”的新数据表，以及在数据表中添加了若干字段，下面我们来预览一下数据库的结构：

```
mysql> show tables;
+------------------+
| Tables_in_saas   |
+------------------+
| tbl_feedback     |
+------------------+

mysql> describe tbl_feedback;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| id         | int(11)      | NO   | PRI | NULL    | auto_increment |
| fb_name    | varchar(128) | NO   |     | NULL    |                |
| fb_email   | varchar(128) | NO   |     | NULL    |                |
```

```
| fb_subject | text         | NO   |     | NULL    |                |
| fb_body    | text         | NO   |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
5 rows in set (0.02 sec)
```

【TIPS】当然，您也可以选择在phpMyAdmin中进行以上操作，但在CLI中，您将对过程的细节有更好的把握，而且在实际项目的部署中，往往并不部署phpMyAdmin，因此应该学会在CLI中进行常规的数据库操作。

7.2 准备工作二：配置Yii的数据库功能

打开Yii应用的主配置文件“protected/config/main.php”，根据以下Diff修改配置：

```
 1 +    // 注释以下语句因为我们需要使用MySQL数据库代替默认的SQLite数据库
 2 +    /*
 3      'db'=>array(
 4          'connectionString' => 'sqlite:protected/data/testdrive.db',
 5      ),
 6 +     */
 7      // uncomment the following to use a MySQL database
 8 -    /*
 9 -    'db'=>array(
10 -        'connectionString' => 'mysql:host=localhost;dbname=testdrive',
11 -        'emulatePrepare' => true,
12 -        'username' => 'root',
13 -        'password' => '',
14 -        'charset' => 'utf8',
15 -    ),
16 -    */
17 +    'db'=>array(
18 +        'connectionString' => 'mysql:host=localhost;dbname=saas',
19 +        'tablePrefix' => 'tbl_',
20 +        'emulatePrepare' => true,
21 +        'username' => 'saas_user',
22 +        'password' => 'saas_password',
23 +        'charset' => 'utf8',
24 +    ),
```

我们仅做了两步修改：

1. 注释默认的SQLite配置代码；
2. 添加MySQL数据库的配置代码。

聚焦到新添加的代码：

```
1 'db'=>array(
2     'connectionString' => 'mysql:host=localhost;dbname=saas',
3     'tablePrefix' => 'tbl_',
4     'emulatePrepare' => true,
5     'username' => 'saas_user',
6     'password' => 'saas_password',
7     'charset' => 'utf8',
8 ),
```

以上代码我们根据Yii的规则通过数组的形式配置了一个MySQL数据库连接，代码很容易看懂，仅有以下细节需要解释一下：

- 第3行的“tablePrefix”参数是“使用表前缀以简化SQL代码”的时才是必要的，在这个例子中暂时不需要使用，为了便于扩展我先配置了它，可参阅：http://www.yiiframework.com/doc/guide/database.dao ；
- 第7行的“charset”需要根据您的项目语言来配置，如果需要在项目中支持中文，建议如上配置为“utf8”。

7.3 保存数据条目

接下来修改“helloworld”控制器“actionFeedback”方法的代码为：

```
 1 public function actionFeedback()
 2 {
 3     $model=new FeedbackForm;
 4     if (isset($_POST['FeedbackForm'])) {
 5         $model->attributes=$_POST['FeedbackForm'];
 6         //$this->renderPartial('feedbackshow',array('model'=>$model));
 7         $dbconnect=Yii::app()->db;
 8         $strSQL="insert into tbl_feedback
 9                 (fb_name,fb_email,fb_subject,fb_body) VALUES
10                 ('$model->name','$model->email',
11                 '$model->subject','$model->body')";
12         $dbcommand=$dbconnect->createCommand($strSQL);
13         $dbcommand->execute();
14     } else {
15         $this->renderPartial('feedback',array('model'=>$model));
16     }
17 }
```

简单分析：

- 第7行：引用Yii主配置文件中的数据库连接（这个连接的实例会被自动创建）；
- 第8-11行：新建一个字符串“$strSQL”以保存需要执行的SQL语句；
- 第12行：用“createCommand()”方法根据“$strSQL”（SQL语句）创建一个SQL命令对象；
- 第13行：用“execute()”方法执行SQL命令；

这是执行结果：

1. 浏览器打开：
   http://localhost/index.php?r=helloworld/feedback
   填入一些信息并提交表单；
2. 进入MySQL的CLI查看信息是否已经被保存入数据库：

```
mysql> select * from tbl_feedback;
+----+---------+--------------+------------+---------------------+
| id | fb_name | fb_email     | fb_subject | fb_body             |
+----+---------+--------------+------------+---------------------+
|  3 | Leask   | i@leaskh.com | Hello Yii  | I do not like wall! |
+----+---------+--------------+------------+---------------------+
1 row in set (0.00 sec)
```

7.4 读取并显示数据库条目

接下来我们测试直接在网页上输出这个数据表的所有条目，以学习在Yii中如何查询数据条目。

首先创建一个新的视图“protected/views/dbfetch.php”，用来显示数据条目。代码如下：

```
1 <table border="1" cellpadding="7">
2     <tr>
3         <td>姓名</td>
4         <td>E-mail</td>
5         <td>主题</td>
6         <td>内容</td>
7     </tr>
8 <?php foreach ($results as $result){ ?>
9     <tr>
10        <td><?php echo $result['fb_name'] ; ?></td>
11        <td><?php echo $result['fb_email'] ; ?></td>
12        <td><?php echo $result['fb_subject'] ; ?></td>
13        <td><?php echo $result['fb_body'] ; ?></td>
14    </tr>
15 <?php } ?>
16 </table>
```

如下修改“actionFeedback”方法的代码为：

```
1 public function actionFeedback()
2 {
3     $model=new FeedbackForm;
4     if (isset($_POST['FeedbackForm'])) {
5         $model->attributes=$_POST['FeedbackForm'];
6         //$this->renderPartial('feedbackshow',array('model'=>$model));
7         $dbconnect=Yii::app()->db;
8         $strSQL="insert into tbl_feedback
9                  (fb_name,fb_email,fb_subject,fb_body) VALUES
10                 ('$model->name','$model->email',
11                 '$model->subject','$model->body')";
12        $dbcommand=$dbconnect->createCommand($strSQL);
13        $dbcommand->execute();
14
15        $strSQL="select * from tbl_feedback";
16        $dbcommand=$dbconnect->createCommand($strSQL);
17        $results=$dbcommand->queryAll();
18        $this->renderPartial('dbfetch',array('results'=>$results));
19    } else {
20        $this->renderPartial('feedback',array('model'=>$model));
21    }
22 }
```

其中第15至18行为新增代码，实现查询数据库并通过视图“dbfetch”输出结果；

重复上文的测试方法，得到如下输出表示实验成功：

姓名	E-mail	主题	内容
Leask	i@leaskh.com	Hello Yii	I do not like wall!
Sixia	leaskh@gmail.com	Hi Nanjing	I am working now.
HSX	e@leaskh.com	Test again!	This is another test.

“queryAll()”是Yii查询数据库的其中一种方式，常用的查询（访问）数据库的方式还包括：

- execute() // 执行非查询性质的SQL语句（如“insert”、“delete”等），返回被操作的行数；
- query() // 执行查询性的SQL语句，返回一个“DataReader”对象；
- queryAll() // 执行查询性的SQL语句，以二维数组的形式返回所有行；
- queryRow() // 执行查询性的SQL语句，以一维数组的形式返回第一行；
- queryColumn() // 执行查询性的SQL语句，以一维数组的形式返回第一列；
- queryScalar() // 执行查询性的SQL语句，返回第一行的第一个字段。

【TIPS】更详细的介绍可以参阅： http://www.yiiframework.com/doc/guide/database.dao

7.5 删除数据条目

删除数据条目这里就不罗嗦了，直接用“execute()”方法执行如下SQL语句即可：

```
1 delete from [数据表名称] where [删除条件]
```

7.6 其他数据库功能简介

我们已经学习了通过Yii访问数据库的一些常识，事实上Yii还内置很多有用的类和方法以大大方便我们操作数据库，如：

1. Yii DAO部分（本文没有介绍的）： http://www.yiiframework.com/doc/guide/database.dao
    - 以“事务”方式操作数据库；
    - 参数绑定；
    - 字段绑定；
    - 表前缀。
2. Active Record： http://www.yiiframework.com/doc/guide/database.ar ；
3. 在Active Record中使用跨表关联： http://www.yiiframework.com/doc/guide/database.arr 。

8 简单的URL路由配置

我们回顾一下之前测试代码的时候所使用的URL，如：

- http://localhost/index.php?r=helloworld
- http://localhost/index.php?r=helloworld/feedback

您也许已经留意到，其实两个地址访问的都是同一个入口脚本“index.php”，只是后面跟的“r”参数的值不一样。

事实上无论Yii承载多大的项目，用户直接访问的都仅是“index.php”这个文件，Yii通过根据不同的访问参数，决定实例化那个类来响应用户的访问请求。

然而面对纷繁复杂的用户请求，如何正确响应呢？Yii通过“URL路由”解决了这个问题。

可是Yii默认的URL无论对用户还是对搜索引擎都不够友好，因此，我们需要稍加配置来完善它。

在这个例子中我们尝试把以下URL：

http://localhost/index.php?r=helloworld/feedback

转化为更友好的形式：

http://localhost/helloworld/feedback

8.1 第一步：配置服务器的URL Rewrite（URL重写）功能

在Apache的站点配置文件中添加如下配置代码（其他Web服务器可能会有所差异）：

```
Options +FollowSymLinks
IndexIgnore */*
RewriteEngine on

# if a directory or a file exists, use it directly
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d

# otherwise forward it to index.php
RewriteRule . index.php
```

以告知Web服务器，在该站点中，如果用户需要访问的文件是存在的，那么直接访问该文件，否则就重定向到“index.php”文件。

【TIPS】

- 如果以上配置不起作用，请确定“RewriteEngine”模块是否已被正确配置；
- 虽然您也可以在Yii应用程序的根文件夹里通过“.htaccess”文件实现以上功能，但是这种方法会降低服务器的性能，因此并不推荐。

8.2 第二步：配置Yii处理URL的规则

编辑“protected/config/main.php”，加入如下配置：

```
1 'urlManager'=>array(
2     'urlFormat'=>'path',
3     'showScriptName' => false,
4     'rules'=>array(
5         'post/<id:\d+>/<title:.*?>'=>'post/view',
6         'posts/<tag:.*?>'=>'post/index',
7         '<controller:\w+>/<action:\w+>'=>'<controller>/<action>',
8     ),
9 ),
```

这将告知Yii以虚拟路径的方式处理URL，并在创建新URL的时候在路径中隐藏“index.php”。

其中的"rule"参数只是一个例子，大家可以根据具体需要进行个性化。

8.3 小结

至此您已经可以通过新的URL：

http://localhost/helloworld/feedback

访问前文所创建的程序了。

Yii还提供更复杂的URL路由配置，来满足个性化和SEO的需求，例如"伪静态化"等。

想了解更多Yii URL路由的配置以及负责URL生成的类，可参阅： http://www.yiiframework.com/doc/guide/topics.url 。

9 进阶指引

通过以下的学习，大家对基于Yii进行开发已经有一个大致的了解了。事实上，通过对以上技巧的灵活运用，构建一些逻辑相对简单的应用来说，已经足够了。然而Yii的目标是面向大型网站的，提高开发大型网站的效率才是学习Yii的目的。因此在掌握Yii的基础以后，如何进一步提高呢？这里列举一下接下来要学习的相关知识：

- 进一步了解Yii命令行（yiic），特别是"model"和"crud"命令：
  http://www.yiiframework.com/doc/guide/topics.console ；
- 了解Theming（主题）功能：
  http://www.yiiframework.com/doc/guide/topics.theming ；
- 了解Yii的"认证和授权"机制：
  http://www.yiiframework.com/doc/guide/topics.auth ；
- 了解Yii的官方扩展"Zii"：
  http://code.google.com/p/zii/ ；
- 了解如何通过"component"、"behavior"、"widget"、"filter"、"helper"、"module"形式等扩展Yii，这些形式严格上来说是不必要的，但是合理利用他们能够更有效组织大型项目，更便于功能的增删调整和局部调试：
  http://www.yiiframework.com/doc/guide/extension.overview ；
- 了解如何使用Cache（缓存）来为应用加速：
  http://www.yiiframework.com/doc/guide/caching.overview ；
- 学会调试Yii，以及通过配置"PHPUnit"和"xdebug"实现更高级的调试功能：
  http://www.yiiframework.com/doc/guide/test.overview 。
- Yii官方教材《Guide》中还包括：国际化（I18N）、安全技术、性能优化等实用章节，可以根据您的项目有哪些具体要求再仔细研究，这些内容都在《Special Topics》一章，这里就不一一给出链接了。

10 附录 / 列举一些较理想的学习资源

10.1 必修教程

- The Yii Blog Tutorial / 语言：英文+局部汉化 / 评级：★★★★★
  在线阅读： http://www.yiiframework.com/doc/blog/
  PDF下载： http://www.yiiframework.com/files/yii-blog-1.1.1.pdf
  本地下载： http://192.168.1.247/shared/yii-blog-1.1.1.pdf （推荐）
  局部汉化： http://www.yiiframework.com/doc/blog/zh_cn/start.overview （汉化不完整，API较旧）
  中文SVN： http://yii.googlecode.com/svn/branches/1.0/docs/guide/zh_cn/ （汉化有进展但仍不完整，API较旧）
  备注：此教材英文版已经打印装订成册

- The Definitive Guide to Yii / 语言：英文+局部汉化 / 评级：★★★★★
  在线阅读： http://www.yiiframework.com/doc/guide/
  PDF下载： http://www.yiiframework.com/files/yii-guide-1.1.1.pdf
  本地下载： http://192.168.1.247/shared/yii-guide-1.1.1.pdf （推荐）
  局部汉化： http://www.yiiframework.com/doc/guide/zh_cn/index （汉化不完整，API较旧）
  非官方汉化： http://dreamneverfall.cn/yiidoc/quickstart.what-is-yii.htm （汉化比官方版本完整，但部分版本有误，详情请见：Yii 文档勘误、陷阱提示与心得记录）

10.2 选修教程

- Larry Ullman's Learning Yii Series / 语言：英文 / 评级：★★★☆☆
  在线阅读： http://blog.dmcinsights.com/2009/06/18/introduction-to-the-yii-framework/

- The Yii Cookbook / 语言：英文 / 评级：★★★☆☆
  在线阅读： http://www.yiiframework.com/doc/cookbook/

10.3 常备查阅

- Yii Framework Class Reference / 语言：英文 / 评级：★★★★★

在线阅读： http://www.yiiframework.com/doc/api/

- The Yii Cheat Sheet / 语言：英文 / 评级：★★☆☆☆
  PDF下载： http://www.yiiframework.com/files/yii-1.0-cheatsheet.pdf
  本地下载： http://192.168.1.247/shared/yii-1.0-cheatsheet.pdf （推荐）
  备注：此文件当前版本为1.0.8，滞后于框架当前版本

- Yii validator cheatsheet / 语言：英文 / 评级：★★☆☆☆
  PDF下载： http://yii.googlecode.com/files/yii-1.1.0-validator-cheatsheet.pdf
  本地下载： http://192.168.1.247/shared/yii-1.1.0-validator-cheatsheet.pdf （推荐）
  备注：此文件当前版本为1.1.0，滞后于框架当前版本

10.4 多媒体教程

- How to create a blog system using Yii in less than 30 minutes / 语言：英文 / 评级：★★★☆☆
  在线观看： http://www.yiiframework.com/screencast/blog/ （缓冲较慢）
  本地下载： http://192.168.1.247/shared/yii_blog_in30ms.mov （推荐）

- Yii Radio Podcasts / 语言：英文 / 评级：★★☆☆☆
  在线收听： http://yiiradiio.mehesz.net/
  RSS订阅： http://feeds.feedburner.com/YiiRadiio-APodcastAboutTheYiiphpFramework （推荐）
  iTunes订阅： http://itunes.apple.com/WebObjects/MZStore.woa/wa/viewPodcast?id=325740146

11 总结

终于完成此教程了，也衷心感谢能够看到最后一节的朋友们！

此文从比较基础的概念入手，通过简单的例子介绍了Yii开发中的一些常识，由于本人才疏学浅，文中难免有错漏粗糙之处，还望大家批评指出，我将及时修正。

本文由于成文时间仓促以及本人对Yii的认识也实为有限，并不能囊括Yii开发的方方面面，因此本文只愿能起到抛砖引玉之作用，成为大家学习Yii的一份比较理想的中文入门教程。

最后希望Yii能成为大家日后进行Web开发的又一神兵利器。

yii_express_mvc_logic.png (5.1 kB) 黄 思夏, 2010-04-06 11:58
yii_yiic_creat_testapp.png (37 kB) 黄 思夏, 2010-04-21 20:00
yii_testapp_in_netbeans.png (54.1 kB) 黄 思夏, 2010-04-21 20:08
yii_helloworld_feedback_form_creat.png (5.4 kB) 黄 思夏, 2010-04-21 21:14
yii_mysql_db_fetch.png (9.9 kB) 黄 思夏, 2010-04-21 21:15
yii_requirements_checker.png (88.3 kB) 黄 思夏, 2010-05-12 10:14