

# Getting Started with Zend\_Auth

By Rob Allen, [www.akrabat.com](http://www.akrabat.com)

Document Revision 1.0.2

Copyright © 2007

翻译 Jason Qi, [zft.backupdiy.com](http://zft.backupdiy.com)

本教程打算对使用 Zend Framework 中的 Zend\_Auth 做一个最基本的介绍。它基于前一个教程“Getting Started with the Zend Framework Tutorial”（我把它翻译为“和 Zend Framework 一起成长”，下同），见这里 <http://akrabat.com/zend-framework-tutorial>。

注：本教程已经在 Zend Framework 的 0.9 和 0.9.1 版测试通过。它很有希望和以后的版本工作，但可以确定的是它不和 0.9 以前的版本工作。

## 认证

对于本教程的目的，认证就是某人登陆到 web 应用程序里的处理过程。我们打算修改在“Getting Started with the Zend Framework”里创建的 CD 列表程序，要求用户登录后才能访问程序的任何部分。

松散地，我们就是要做这些事情：

- 为用户创建一个数据库表（并加上一个用户记录！）。
- 创建一个登陆表单。
- 创建一个控制器，它包含登录和登出的 actions。
- 修改 footer 来允许登出。
- 确保用户再允许访问程序之前被登录。

## users 表

我们需要的第一件事情是一个 users 表。它不需要太复杂，schema 看起来像这样：

字段名	类型	Null?	注
id	Integer	No	Primary key, Autoincrement
username	Varchar(50)	No	Unique key
Password	Varchar(50)	No	
real_name	Varchar(100)	No	

当使用 MySQL，建表的 SQL 语句是：

```
CREATE TABLE users (  
    id int(11) NOT NULL auto_increment,  
    username varchar(50) NOT NULL,  
    password varchar(50) NOT NULL,  
    real_name varchar(100) NOT NULL,  
    PRIMARY KEY (id),  
    UNIQUE KEY username (username)  
)
```

我们也需要一个能登录的用户：

```
INSERT INTO users (id, username, password, real_name)  
VALUES (1, 'rob', 'rob', 'Rob Allen');
```

在 MySQL 客户端如 phpMyAdmin 或标准的 MySQL 命令行客户端运行这些语句（显然，你需要选择一个较好的用户名和密码！）。

## Bootstrap 修改

为了跟踪用户已经登录的事件，我们打算使用 `session`。Zend Framework 提供了 `Zend_Session_Namespace`，对于 `session`，这是一个挺好的面向对象的接口。

`index.php` 修改是：

**zf-tutorial/index.php:**

```
...
Zend_Loader::loadClass('Zend_Db_Table');
Zend_Loader::loadClass('Zend_Debug');
Zend_Loader::loadClass('Zend_Auth');
```

```
// load configuration
```

```
...
```

和

```
..
// setup database
$dbAdapter = Zend_Db::factory($config->db->adapter,
    $config->db->config->asArray());
Zend_Db_Table::setDefaultAdapter($dbAdapter);
Zend_Registry::set('dbAdapter', $dbAdapter);

// setup controller
$frontController = Zend_Controller_Front::getInstance();
...
```

这里所有必需要做的事情是确保我们加载了 `Zend_Auth` 类和把数据库适配器注册到注册表里。我们把它存储到注册表里，在后面认证的时候会需要的。

## Auth 控制器

我们需要一个控制器来放置登录和登出 `actions`，把它叫做 `AuthController`。

We'll start it with the basics from the `IndexController`:

**zf-tutorial/application/controllers/AuthController.php:**

```
class AuthController extends Zend_Controller_Action
{
    function init()
    {
        $this->initView();
        $this->view->baseUrl = $this->_request->getBaseUrl();
    }

    function indexAction()
    {
        $this->_redirect('/');
    }
}
```

我们设置 `init()`，这样视图就被初始化并且把 `baseUrl` 分配给它。我们也创建一个 `indexAction()` 函数，因为 `Zend_Controller_Action` 要求这样做。我们不需要 `indexAction()`，我们打算使用 `loginAction()` 和 `logoutAction()`，所以，如果有人确实要导航到 `auth/index`，我们就把它重定向到缺省。

## 登录

我们需要一个表单来做登录程序，所以登录的 `action` 和其它在 `IndexController` 下的表单的 `actions` 工作非常类似。表单的模板将放在 `views/scripts/auth/login.phtml` 里并且代码将在 `AuthController::loginAction()` 里。表单非常简单，只需要两个东东：用户名和密码：

### zf-tutorial/application/views/scripts/auth/login.phtml:

```
<?php echo $this->render('header.phtml'); ?>
<h1><?php echo $this->escape($this->title); ?></h1>

<?php if(!empty($this->message)) :?>
<div id="message">
<?php echo $this->escape($this->message);?>
</div>
<?php endif; ?>

<form action="<?php echo $this->baseUrl ?>/auth/login" method="post">
<div>
    <label for="username">Username</label>
    <input type="text" name="username" value="" />
</div>
<div>
    <label for="password">Password</label>
    <input type="password" name="password" value="" />
</div>
<div id="formbutton">
<input type="submit" name="login" value="Login" />
</div>
</form>

<?php echo $this->render('footer.phtml'); ?>
```

通常，模板在顶部和地部分别调用 `header.phtml` 和 `footer.phtml`。注意只有在 `$this->message` 不是空的时候现实信息，这个通常告诉用户登录失败。模板中剩下的部分就是登录表单自己的东西了。

好了，我们一个表单，我们需要创建将要运行的控制器 `action`，它被添加到 `AuthController.php`：

### zf-tutorial/application/controllers/AuthController.php:

```
class AuthController extends Zend_Controller_Action
{
    ...
    function loginAction()
    {
        $this->view->message = '';
        $this->view->title = "Log in";
        $this->render();
    }
}
```

开始，所有我们需要做的是设置标题和信息并且调用表单。如果你导航到 <http://zf-tutorial/auth/login> 就应当看到登录表单。

那么表单提交后怎么办？同（处理）添加和编辑表单一样，当请求方式是 `post` 的时候，我们在 `IndexController` 里来处理它。修改我们刚创建的 `loginAction()`：

### zf-tutorial/application/controllers/AuthController.php:

```
class AuthController extends Zend_Controller_Action
{
    ...
    function loginAction()
    {
        $this->view->message = '';
        if ($this->_request->isPost()) {
            // collect the data from the user
            Zend_Loader::loadClass('Zend_Filter_StripTags');
            $f = new Zend_Filter_StripTags();
            $username = $f->filter($this->_request->getPost('username'));
            $password = $f->filter($this->_request->getPost('password'));

            // setup Zend_Auth adapter for a database table
            Zend_Loader::loadClass('Zend_Auth_Adapter_DbTable');
            $dbAdapter = Zend_Registry::get('dbAdapter');
            $authAdapter = new Zend_Auth_Adapter_DbTable($dbAdapter);
            $authAdapter->setTableName('users');
            $authAdapter->setIdentityColumn('username');
            $authAdapter->setCredentialColumn('password');

            // Set the input credential values to authenticate against
            $authAdapter->setIdentity($username);
            $authAdapter->setCredential($password);

            // do the authentication
            $auth = Zend_Auth::getInstance();
            $result = $auth->authenticate($authAdapter);
            if ($result->isValid()) {
                // success : store database row to auth's storage system
                // (not the password though!)
                $data = $authAdapter->getResultRowObject(null, 'password');
                $auth->getStorage()->write($data);
                $this->_redirect('/');
            } else {
                // failure: clear database row from session
                $this->view->message = 'Login failed.';
            }
        }
        $this->view->title = "Log in";
        $this->render();
    }
}
```

相当多的东西，让我们来过一遍它：

```
// collect the data from the user
Zend_Loader::loadClass('Zend_Filter_StripTags');
$f = new Zend_Filter_StripTags();
$username = $f->filter($this->_request->getPost('username'));
$password = $f->filter($this->_request->getPost('password'));
```

同往常一样，我们设置一个 **filter** 并从 **POST** 数据里析取出用户名和密码。注意我们用请求的 **getPost()** 函数，因为它将用 **isset()** 帮我们来检查，并且如果在 **POST** 数组里不存在那样的字段，它将返回一个空的串。

```

// setup Zend_Auth adapter for a database table
Zend_Loader::loadClass('Zend_Auth_Adapter_DbTable');
$dbAdapter = Zend_Registry::get('dbAdapter');
$authAdapter = new Zend_Auth_Adapter_DbTable($dbAdapter);
$authAdapter->setTableName('users');
$authAdapter->setIdentityColumn('username');
$authAdapter->setCredentialColumn('password');

```

Zend\_Auth 利用适配器系统来允许你使用任何系统来做实际的认证。我们想使用数据库表，所以我们使用 Zend\_Auth\_Adapter\_DbTable。为了设置一个适配器，你告诉它需要用的字段并传递一个有效的连接给数据库。

```

// Set the input credential values to authenticate against
$authAdapter->setIdentity($username);
$authAdapter->setCredential($password);

```

我们需要确切地把用户在表单里所填写的用户名和密码告诉给适配器。

```

// do the authentication
$auth = Zend_Auth::getInstance();
$result = $auth->authenticate($authAdapter);

```

为了认证，我们调用 Zend\_Auth 中的 authenticate () 函数。这就确保认证的结果被自动地存储到 session 中。

```

if ($result->isValid()) {
    // success : store database row to auth's storage system
    // (not the password though!)
    $data = $authAdapter->getResultRowObject(null, 'password');
    $auth->getStorage()->write($data);
    $this->_redirect('/');
}

```

在认证成功的情况下，我们存储整个数据库记录（除了密码）到 Zend\_Auth singleton。这确保我们能收集到用户的名字去显示到页脚。

```

} else {
    // failure: clear database row from session
    $this->view->message = 'Login failed.';
}

```

一旦认证失败，我们设置一些信息让用户知道发生了什么。

登录的认证处理现在大功告成。

## 登出

登出比登录容易多了，因为我们只需要告诉 Zend\_Auth singleton 去清除它的数据。这个由在 AuthController 中的一个新的 logoutAction() action 来完成，我们可以导航到 <http://zftutorial/auth/logout>，这样用户就登出了。

### zf-tutorial/application/controllers/AuthController.php:

```

class AuthController extends Zend_Controller_Action
{
    ...
    function logoutAction()
    {
        Zend_Auth::getInstance()->clearIdentity();
        $this->_redirect('/');
    }
}

```

logoutAction() 函数太微不足道了，我觉得没有什么好说的！

我们现在需要给用户提供一个链接，这样他们就可以登出程序。这个很容易在页脚上做。我们也将告诉用户他们的名字，这样他们能够确信他们已经正确地登录了。他们的名字存储在 `users` 表的 `real_name` 字段，这样在 `Zend_Auth` 实例里是可用的。首先要做的是把它送给视图，我们是在 `IndexController()` 里的 `init()` 中来处理的。

#### zf-tutorial/application/controllers/IndexController.php:

```
class IndexController extends Zend_Controller_Action
{
    function init()
    {
        $this->initView();
        Zend_Loader::loadClass('Album');
        $this->view->baseUrl = $this->_request->getBaseUrl();
        $this->view->user = Zend_Auth::getInstance()->getIdentity();
    }
    ...
}
```

`Zend_Auth` 是一个 `singleton` 确实方便，否则我们现在就要把它存到注册表里！

我们现在需要给 `footer.phtml` 添加一些 HTML:

#### zf-tutorial/application/views/footer.phtml:

```
<?php if($this->user) : ?>
<p id="logged-in">Logged in as <?php
    echo $this->escape($this->user->real_name);??.
<a href="<?php echo $this->baseUrl ?>/auth/logout">Logout</a></p>
<?php endif; ?>
</div>
</body>
</html>
```

HTML 应当看起来相当熟悉，因为这里没有什么新鲜的。我们使用 `escape()` 来确保用户的真名安全正确地显示并且我们使用 `baseUrl` 来设置 `anchor` 的 `href` 到正确的地方。

这就是 `logout` 要求的一切。

## 保护 Actions

所有剩下的事情就是确保如果你没有登录就没有其他 `actions` 可以访问。为了完成这个，我们在 `IndexController` 中的 `preDispatch()` 里添加一些代码。

#### zf-tutorial/application/controllers/IndexController.php:

```
class IndexController extends Zend_Controller_Action
{
    ...
    function preDispatch()
    {
        $auth = Zend_Auth::getInstance();
        if (!$auth->hasIdentity()) {
            $this->_redirect('auth/login');
        }
    }
    ...
}
```

在控制器中，`preDispatch()` 在每个 `action` 之前调用。我们把 `Zend_Auth` 实例和它的 `hasIdentity()` 函数收集到一起告诉我们是否有用户登录，如果没有我们就重定向到 `auth/login` `action`。

都在这里了！

## 结论

作为结论，我们简要地着眼于把 `Zend_Auth` 集成到 MVC 应用程序中。毫无疑问，用 `Zend_Auth` 你可以做很多事情并且还有许多办法改善这些代码，特别是如果你要使用多个要保护的控制器的时候。注意我们还没有涉及到授权的细节，那是由 `Zend_Acl` 来完成的。`Zend_Acl` 用来和 `Zend_Auth` 协作来提供对 `action` 和数据的不同级别访问控制，但这将是另外一个教程的目标。

我希望你觉得本教程有趣和有用。如果你发现任何错误，请给我发 email 到 [rob@akrobat.com](mailto:rob@akrobat.com)。

## 译者注

由于看好 `Zend Framework` 的发展前景，所以想系统地学习它的每个部分。如果能通过一个好的教程来学习，那将事半功倍。这个教程是我目前看到的最好的一个，限于我的水平，如果您觉得某些地方的翻译不是很恰当，请参考原文 <http://www.akrobat.com>。如果您有任何建议，请发邮件到 [jason@backupdiy.com](mailto:jason@backupdiy.com)