第 2 章　PEAR、PECL 的应用

第一篇和第二篇介绍了 PHP 的基本使用以及与数据库结合的实例。通过对前面的学习，读者应该能够使用 PHP 设计并制作一些常见的功能了。很多时候，并不需要自己逐行完成应用中的所有代码，很多常见的功能可能早已被人开发出来了。如果能够有效的利用这些资源，就能在实际开发中达到事半功倍的效果。本章将介绍的 PEAR 与 PECL 就是这样的资源库。

2.1　PEAR 与 PECL 的介绍

PEAR 是“PHP Extension and Application Repository”的缩写，即 PHP 扩展与应用类库。PEAR 的主要作用就是将 PHP 项目开发过程中的一些常用功能编写成类库的形式供用户方便的调用。PEAR 的组件包括数据结构、数学、支付、校验、Web Service 等许多方面。用户可以通过下载这些类库并适当的作一些定制以实现自己需要的功能。PEAR 的出现大大提高了 PHP 程序的开发效率和开发质量。

PECL 是“PHP Extension Community Library”的缩写，即 PHP 扩展库。PECL 可以看作 PEAR 的一个组成部分，提供了与 PEAR 类似的功能。但是与 PEAR 不同的是 PEAR 的所有扩展都是用 PHP 代码编写的，用户在下载到 PEAR 扩展以后可以直接使用将扩展的代码包含到自己的 PHP 文件中使用。而 PECL 是使用 C 语言开发的，通常用于补充一些用 PHP 难以完成的底层功能，往往需要重新编译或者在配置文件中设置后才能在用户自己的代码中使用。

2.2　PEAR 的安装

由于 PEAR 在 PHP 中的广泛应用，在 PHP 的安装包里已经集成了 PEAR 的组件。PEAR 的安装非常简单，只需要在命令行下运行 http://pear.php.net/go-pear 文件即可，具体运行方法及安装过程如下所示。

（1）在命令行下输入如下命令读取 http://pear.php.net/go-pear 文件。

```
C:\php>php -r "readfile('http://pear.php.net/go-pear');" > go-pear
```

（2）在命令行下输入如下命令运行 go-pear 安装程序，会出现 go-pear 的欢迎信息。

```
C:\php>php go-pear
Welcome to go-pear!

Go-pear will install the 'pear' command and all the files needed by
it.  This command is your tool for PEAR installation and maintenance.

Use 'php go-pear local' to install a local copy of PEAR.

Go-pear also lets you download and install the PEAR packages bundled
with PHP: MDB2.
```

```
If you wish to abort, press Control-C now, or press Enter to continue:
```

（3）单击回车键继续。这时需要用户输入 HTTP 代理地址。如果不输入，则表示不使用代理连接网络。

```
HTTP proxy (http://user:password@proxy.myhost.com:port), or Enter for none::
```

（4）单击回车键继续。这时命令行提示一个选择菜单。用户可以通过这个菜单对 PEAR 的安装路径等进行配置。

```
recognized as an internal or external command,
operable program or batch file.

*WARNING*
We found php.exe under C:\php\php.exe, it uses an unknown SA
PI. PEAR commandline
tool has not been tested with it, if you have a CLI (or CGI) php.exe available,
we strongly recommand to use it.



Below is a suggested file layout for your new PEAR installation.    To
change individual locations, type the number in front of the
directory.    Type 'all' to change all of them or simply press Enter to
accept these locations.

 1. Installation prefix                : C:\php
 2. Binaries directory                  : $prefix
 3. PHP code directory ($php_dir) : $prefix\pear
 4. Documentation base directory    : $php_dir\docs
 5. Data base directory                 : $php_dir\data
 6. Tests base directory                : $php_dir\tests
 7. Temporary files directory         : $prefix\temp
 8. php.exe path                          : C:\php\php.exe

1-8, 'all' or Enter to continue:
```

（5）配置完成后，单击回车键继续。这时程序将开始下载和安装 PEAR 包。

```
Loading zlib: ok
Downloading package: PEAR-stable......ok
Downloading package: Archive_Tar-stable....ok
Downloading package: Console_Getopt-stable....ok
Downloading package: XML_RPC-stable....ok
Bootstrapping: PEAR.....................(remote) ok
Bootstrapping: Archive_Tar.............(remote) ok
Bootstrapping: Console_Getopt.........(remote) ok
Downloading package: MDB2.............ok
Extracting installer.....................ok
pear/PEAR can optionally use package "pear/PEAR_Frontend_Web" (version >= 0.5.0)

pear/PEAR can optionally use package "pear/PEAR_Frontend_Gtk" (version >= 0.4.0)

pear/PEAR can optionally use package "pear/PEAR_Frontend_Gtk2" (version >= 0.1.0
)
```

```
install ok: channel://pear.php.net/PEAR-1.4.11
install ok: channel://pear.php.net/Archive_Tar-1.3.1
install ok: channel://pear.php.net/Console_Getopt-1.2
install ok: channel://pear.php.net/XML_RPC-1.5.0
install ok: channel://pear.php.net/MDB2-2.2.1

**************************************************************************
WARNING!  The include_path defined in the currently used php.ini does not
contain the PEAR PHP directory you just specified:
<C:\php\pear>
If the specified directory is also not in the include_path used by
your scripts, you will have problems getting any PEAR packages working.
```

（6）安装完成后，程序提示是否修改 PHP 的配置文件。如果选择“Y”将更新 php.ini 文件中的 include_path 参数。当前的 PEAR 安装路径将被放置到 include_path 中以供 PHP 程序调用。

```
Would you like to alter php.ini <C:\php\php.ini>? [Y/n] : n

Please look over your php.ini file to make sure
C:\php\pear is in your include_path.
Current include path           : .;C:\php\pear\
Configured directory           : C:\php\pear
Currently used php.ini (guess) : C:\php\php.ini
Press Enter to continue:
```

（7）修改了 PHP 配置文件以后，单击回车键将结束 PEAR 的安装程序。

```
The 'pear' command is now at your service at C:\php\pear.bat



** The 'pear' command is not currently in your PATH, so you need to
** use 'C:\php\pear.bat' until you have added
** 'C:\php' to your PATH environment variable.

Run it without parameters to see the available actions, try 'pear list'
to see what packages are installed, or 'pear help' for help.

For more information about PEAR, see:

   http://pear.php.net/faq.php
   http://cvs.php.net/co.php/pearweb/doc/pear_package_manager.txt?p=1
   http://pear.php.net/manual/

Thanks for using go-pear!




* WINDOWS ENVIRONMENT VARIABLES *
For convenience, a REG file is available under C:\php\PEAR_E
NV.reg .
This file creates ENV variables for the current user.

Double-click this file to add it to the current user registry.
```

安装完成后，安装程序会在 PHP 的安装目录下创建一个 PEAR_ENV.reg 文件用来存储当前环境变量。然后通过双击这个文件将其中的内容导入注册表即可。需要注意的是为了防止注册表的破坏，在导入前最好备份一下注册表，以防止不必要的麻烦。安装 PEAR 的同时也安装了 PEAR Package Manager。

PEAR Package Manager 是一个用来下载、安装和管理 PEAR 组件的命令行工具。在 PHP 的安装目录下执行 PEAR.bat 文件，如果可以在命令行下看到类似下面的内容，则说明 PEAR 已经安装成功了。

```
C:\php>pear
Commands:
build                 Build an Extension From C Source
bundle                Unpacks a Pecl Package
channel-add           Add a Channel
channel-alias         Specify an alias to a channel name
channel-delete        Remove a Channel From the List
channel-discover      Initialize a Channel from its server
channel-info          Retrieve Information on a Channel
channel-update        Update an Existing Channel
clear-cache           Clear Web Services Cache
config-create         Create a Default configuration file
config-get            Show One Setting
config-help           Show Information About Setting
config-set            Change Setting
config-show           Show All Settings
convert               Convert a package.xml 1.0 to package.xml 2.0 format
cvsdiff               Run a "cvs diff" for all files in a package
cvstag                Set CVS Release Tag
download              Download Package
download-all          Downloads each available package from the default channel

info                  Display information about a package
install               Install Package
list                  List Installed Packages In The Default Channel
list-all              List All Packages
list-channels         List Available Channels
list-files            List Files In Installed Package
list-upgrades         List Available Upgrades
login                 Connects and authenticates to remote server
logout                Logs out from the remote server
makerpm               Builds an RPM spec file from a PEAR package
package               Build Package
package-dependencies  Show package dependencies
package-validate      Validate Package Consistency
pickle                Build PECL Package
remote-info           Information About Remote Packages
remote-list           List Remote Packages
run-scripts           Run Post-Install Scripts bundled with a package
run-tests             Run Regression Tests
search                Search remote package database
shell-test            Shell Script Test
sign                  Sign a package distribution file
uninstall             Un-install Package
```

```
update-channels          Update the Channel List
upgrade                  Upgrade Package
upgrade-all              Upgrade All Packages
Usage: pear [options] command [command-options] <parameters>
Type "pear help options" to list all options.
Type "pear help shortcuts" to list all command shortcuts.
Type "pear help <command>" to get the help for the specified command.
```

这里列出的是 PEAR Package Manager 的一些常见命令，对于这些命令的一些常见使用方法，本书将在下一节具体介绍。

2.3　PEAR 的安装与使用

前面介绍了 PEAR 的安装方法并简要介绍了如何使用 PEAR Package Manager。下载、安装、升级、卸载一个 PEAR 包均可以通过 PEAR Package Manager 来完成，也可以通过访问 PEAR 的官方网站获取。本节将具体介绍如何应用 PEAR 包。

2.3.1　查看已安装 PEAR 包

查看当前服务器已经安装了哪些 PEAR 包可以通过下面的命令来完成。

```
pear list
```

输出结果类似下面所示。

```
C:\php>pear list
INSTALLED PACKAGES, CHANNEL PEAR.PHP.NET:
=========================================
PACKAGE                    VERSION   STATE
Archive_Tar                1.3.1     stable
Auth_PrefManager           1.1.4     stable
Auth_RADIUS                1.0.4     stable
Auth_SASL                  1.0.1     stable
Benchmark                  1.2.4     stable
Cache                      1.5.4     stable
Cache_Lite                 1.5.2     stable
Config                     1.10.4    stable
Console_Getargs            1.3.0     stable
Console_Getopt             1.2       stable
Console_Table              1.0.2     stable
Contact_Vcard_Build        1.1.1     stable
Contact_Vcard_Parse        1.31.0    stable
```

这里，PACKAGE 指的是 PEAR 包的名称，VERSION 指的是 PEAR 包的版本号，STATE 指的是 PEAR 包的状态。如果 PEAR 包的状态是 stable，则表示这个 PEAR 包已经被严格检查并测试通过。

2.3.2　查看 PEAR 包的详细信息

在浏览了服务器上安装的 PEAR 包列表以后，如果需要进一步查看某一个 PEAR 包的详细信息可以

通过下面的命令来完成。

```
pear info <pear-package-name>
```

这里，pear-package-name 指的是 PEAR 包的名称。

例如，想要查看 Cache 的详细信息可以通过下面的命令来实现。

```
C:\php>pear info Cache
```

输出结果如下所示。

```
ABOUT CACHE-1.5.4
=================
Provides            Classes:
Package             Cache
Summary             Framework for caching of arbitrary data.
Description         With the PEAR Cache you can cache the result of
                    certain function
                    calls, as well as the output of a whole script
                    run or share data
                    between applications.
Maintainers         Christian Stocker <chregu@php.net> (lead)
                    Ulf Wendel <ulf.wendel@phpdoc.de> (developer)
Version             1.5.4
Release Date        2004-04-01
Release License PHP License
Release State       stable
Release Notes       - Removed Cache_DB as it apparently never really
                    worked.
                    - Added trifile container. See
                    http://atomized.org/PEAR/Cache_trifile.html for
                    more information. (By Ian Eure)
Release Deps        Package HTTP_Request ==
Last Modified       2004-10-01
```

这里给出了 PEAR 包的描述、维护人员、版本号、发布日期、协议类型、最后修改时间等信息。

2.3.3　安装一个 PEAR 包

安装一个 PEAR 包是通过下面的命令来实现的。

```
pear install [option] <pear-package-name>
```

这里，pear-package-name 指的是 PEAR 包的名称，option 是一个可选的选项，表示安装时的一些安装选项。下面的例子是安装 DB_QueryTool 包的过程。

```
C:\php>pear install DB_QueryTool
downloading DB_QueryTool-1.0.3.tgz ...
Starting to download DB_QueryTool-1.0.3.tgz (40,429 bytes)
.....done: 40,429 bytes
install ok: channel://pear.php.net/DB_QueryTool-1.0.3
```

可以看到 DB_QueryTool 包被成功下载并安装了。安装后会在 PHP 安装目录下的 PEAR 文件夹下创建一个新的以 PEAR 包名称命名的文件夹。

除了使用这种方式安装以外，PEAR 包还可以通过访问 PEAR 官方网站获取，在浏览器的地址栏输入 http://pear.php.net 即可。下载选择好的 PEAR 包以后，将其解压缩到一个指定的文件夹中就可以使用了。

有一些 PEAR 包中调用了其他 PEAR 包的代码，那么被调用的 PEAR 包则称之为它的依赖。在安装这类 PEAR 包时，如果没有安装它的依赖，在使用时就会出现问题。例如，Auth 包就是 Auth_HTTP 包的依赖，如果没有安装 Auth，Auth_HTTP 是无法被正常使用的。解决这个问题主要有两种方法。一种就是先将其依赖安装，然后再安装它本身。即先安装 Auth，然后再安装 Auth_HTTP。或者通过 PEAR Package Manager 的方式安装来自动获取依赖。第二种方法比较简单易行，安装过程如下所示。

```
C:\php>pear install auth_http
pear/Auth can optionally use PHP extension "vpopmail"
downloading Auth_HTTP-2.1.6.tgz ...
Starting to download Auth_HTTP-2.1.6.tgz (9,327 bytes)
.....done: 9,327 bytes
downloading Auth-1.3.2.tgz ...
Starting to download Auth-1.3.2.tgz (41,766 bytes)
...done: 41,766 bytes
install ok: channel://pear.php.net/Auth-1.3.2
install ok: channel://pear.php.net/Auth_HTTP-2.1.6
```

可以看到，在安装 Auth_HTTP 的同时，Auth 也被安装了。

2.3.4　PEAR 包的升级

PEAR 包也是不断的被完善的，因此，获取最新的 PEAR 包不仅可以获得更完善的功能，还会消除因为低版本 PEAR 包中的 bug 带来的隐患。

升级 PEAR 包的方式主要有两种。一种是通过 PEAR Package Manage 对 PEAR 包进行升级，还有一种是通过访问 PEAR 的官方网站获取 PEAR 包的最新版本，然后下载并覆盖旧版本文件。第二种方法与前面介绍的通过访问网站安装一个 PEAR 包的方法相同，这里不再介绍。

使用 PEAR Package Manager 升级 PEAR 包的命令如下所示。

```
pear upgrade <pear-package-name>
```

这里，pear-package-name 指的是 PEAR 包的名称。

下面是升级 Log 包的过程。

```
C:\php>pear upgrade log
downloading Log-1.9.8.tgz ...
Starting to download Log-1.9.8.tgz (38,841 bytes)
.........done: 38,841 bytes
upgrade ok: channel://pear.php.net/Log-1.9.8
```

可以看到升级 PEAR 包与安装 PEAR 包的过程几乎完全相同，不同的是安装前会校验当前版本是否为最新版本。例如，在将 Log 包升级到最新后再次试图升级 Log 包时会出现如下错误。

```
Skipping package "pear/Log", already installed as version 1.9.8
No valid packages found
upgrade failed
```

如果需要对服务器上安装的所有 PEAR 包进行升级，可以通过下面的命令来完成。

```
pear upgrade-all
```

执行后，所有已经安装的 PEAR 包都会升级到最新的版本。

2.3.5　PEAR 包的使用

PEAR 包的使用非常简单，只需要在 PHP 脚本文件中包含 PEAR 包文件即可，如下所示。

```
require_once('fpdf.php');
```

包含后，就可以直接在脚本中调用 PEAR 包中的类了。某些时候，还可以根据需要直接对 PEAR 包中的代码进行修改。

2.4　常用的 PEAR 类库实例

前面介绍了 PEAR 包的安装与基本使用方法。目前，可用的 PEAR 类库已经有很多种，而且还在不断的增加中。本节将简要介绍几个常用的 PEAR 类库，对于其他的类库，读者可以通过类库中的说明文件或者网站上的介绍进行使用。

2.4.1　使用 DB 类库进行数据库查询

在本书第二篇介绍了直接用 PHP 操作数据库的方法，除此之外，还可以通过 PEAR 包中的 DB 类库进行数据库的操作。DB 类库提供了数据库连接和操作的简便方法。

1．连接数据库

连接数据库是通过 DB 类中的 connect 函数来实现的，其语法格式如下所示。

```
DB::connect(string conn_str);
```

这里，conn_str 是一个连接字符串。以 MySQL 数据库为例，连接字符串的形式如下所示。

```
mysql://username:password@host_name/database_name
```

这里 username 是 MySQL 的用户名，password 是该用户名相应的密码，host_name 是服务器地址，database_name 是要连接的数据库名。以下代码是一个连接 MySQL 数据库的例子。

```
<?php
require_once "DB.php";                                    //包含类库文件
$conn = DB::connect("mysql://root@localhost/mydb");       //连接数据库
if (!DB::isError($conn)) {                                //判断是否连接成功
    print "数据库连接成功";
}
?>
```

如果数据库连接成功，则上述程序会输出“数据库连接成功”。需要注意的是这里用到了一个函数 DB::isError，该函数也是 DB 类中的一个方法，用来判断是否有错误产生。

2．执行 SQL 语句查询数据库

查询数据库是通过 DB 类中的 query 函数来实现的，其语法格式如下所示。

```
query($sql_statement);
```

这里，$sql_statement 是一个用于执行查询的 SQL 语句。该函数是上述连接字符串返回对象的一个方法，执行完成后返回一个结果集。可以通过调用结果集对象的方法 fetchInto 将结果集放置到数组中。以下代码是一个执行 SQL 语句查询数据库的例子。

```
<?php
require_once "DB.php";
$conn = DB::connect("mysql://root@localhost/mydb");       //调用 connect 连接数据库
if (DB::isError($conn))                                   //如果连接出错则报错
{
    print "数据库连接失败";
```

```
}
$rs = $conn->query("select serial_no, name, age from mytable where salary = 5000.00");        //执行 SQL 语句
if (DB::isError($rs))                                                                          //判断是否执行成功
{
    print "数据查询失败";
}
while ($rs->fetchInto($rows))                                                                  //循环输出查询结果
{
    print "员工号：$rows[0]<BR>";
    print "姓名：$rows[1]<BR>";
    print "年龄：$rows[2]<BR>";
    print "<HR>";
}
?>
```

运行结果如图 2-1 所示。

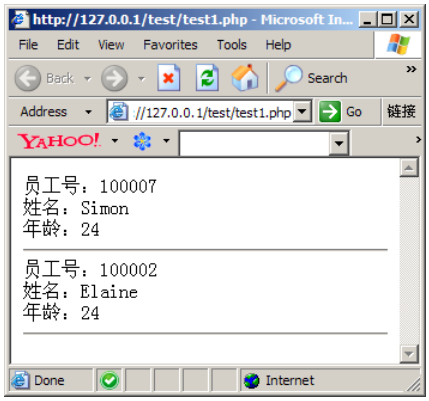

图 2-1　query 的运行结果

除了 query 函数以外，DB 类还提供了一个查询数据库的方法 limitQuery。limitQuery 方法与 query 类似，只是提供了可以返回结果集中某几条记录的方法，其语法格式如下所示。

```
limitQuery($sql_statement, $start, $limit);
```

这里，$sql_statement 是一个用于执行查询的 SQL 语句，$start 是返回记录的起始数，$limit 是一共返回几条记录。该函数返回由$sql_statement 产生的结果集中自$start 开始的$limit 条数据。需要注意的是这里的$start 是从零开始计数的。以下代码改写了上面的例子，使其只返回第一条记录。

```
<?php
require_once "DB.php";
$conn = DB::connect("mysql://root@localhost/mydb");                    //调用 connect 连接数据库
if (DB::isError($conn))                                                //如果连接出错则报错
{
    print "数据库连接失败";
}
//执行 SQL 语句，从第 0 条开始返回 1 条记录
$rs = $conn->limitQuery("select serial_no, name, age from mytable where salary = 5000.00",0,1);
if (DB::isError($rs))                                                  //如果查询出错则报错
{
    print "数据查询失败";
}
while ($rs->fetchInto($rows))                                          //循环输出查询结果
```

```
{
    print "员工号：$rows[0]<BR>";
    print "姓名：$rows[1]<BR>";
    print "年龄：$rows[2]<BR>";
    print "<HR>";
}
?>
```

运行结果如图 2-2 所示。

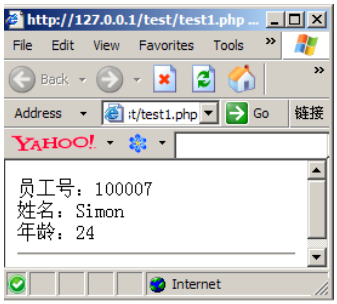

图 2-2　limitQuery 的运行结果

从运行结果可以看到，只有第一条记录显示出来了。

3．执行 SQL 语句更新数据库

更新数据库是通过 DB 类中的 prepare 函数和 execute 函数来实现的，其语法格式如下所示。

```
prepare($sql_statement);
```

这里，$sql_statement 是一个用于执行查询的 SQL 语句。该函数是上述连接字符串返回对象的一个方法，执行完成后返回一个可执行得 SQL 操作对象。可以通过调用该对象的方法 execute 完成对数据库的更新。以下代码是一个执行 SQL 语句更新数据库的例子。

```
<?php
require_once "DB.php";
$conn = DB::connect("mysql://root@localhost/mydb");                    //连接数据库
if (DB::isError($conn))
{
    print "数据库连接失败";
}
//使用 prepare 函数准备 SQL 语句
$rs = $conn->prepare("update mytable set name = 'Susan' where serial_no = 100003");
if (DB::isError($rs))
{
    print "数据更新失败";
}
else
{
    $conn->execute($rs);                                               //执行 SQL 语句更新数据库
    print "数据更新成功";
}
?>
```

与 Java 语言中的 prepare 类似，这里也可以构造动态 SQL 语句。以下代码改写了上面的例子并实现了同样的功能。

```
<?php
```

```
require_once "DB.php";
//准备参数数组
$parm = array(100003, "Susan");
//连接数据库
$conn = DB::connect("mysql://root@localhost/mydb");
if (DB::isError($conn))
{
    print "数据库连接失败";
}
//准备 SQL 语句
$rs = $conn->prepare("update mytable set name = ? where serial_no = ?");
if (DB::isError($rs))
{
    print "数据更新失败";
}
else
{
    //执行 SQL 语句
    $conn->execute($rs, $parm);
    print "数据更新成功";

}
?>
```

注意这里 execute 的第二个参数，是一个有两个元素的数组。在实际执行的时候，会将这两个元素分别放在 prepare 函数中 SQL 语句中的两个问号处代替。

由于篇幅有限，DB 类库就介绍到这里，对于 DB 类库的其他功能就不一一介绍了。读者可以通过查看 DB 类库的使用手册学习 DB 类库的更多功能。

2.4.2　使用 Auth_HTTP 类库进行身份校验

Auth 类库是一个专门用来实现访问用户身份校验的类库。该类库可以通过密码文件、数据库、LDAP 服务器、SOAP 等多种方式获取身份校验信息进行身份校验。Auth_HTTP 类库通过调用 Auth 的功能函数，以对话框的形式实现对用户信息的强制校验。这种形式的身份校验在实际应用中非常常用，特别是一些基于 Web 的管理系统中。以下代码是一个使用 Auth_HTTP 类库进行身份校验的例子。该例子采用 MySQL 作为用户身份校验的信息来源。

```
<?php
require_once("Auth/HTTP.php");
//设置数据库的连接选项
$auth_options=array(
    'dsn'=>"mysql://root@localhost/mydb",          //数据库连接字符串
    'table'=>"myadmin",                            //表名
    'usernamecol'=>"username",                     //用于存储用户名的列
    'passwordcol'=>"password",                     //用于存储密码的列
    'cryptType'=>"none",                           //密码加密方式
);
//创建 Auth_HTTP 对象，指明采用 DB 作为信息来源
$auth = new Auth_HTTP("DB", $auth_options);
```

```
//设置对话框上的说明信息
$auth->setRealm('Login');
//身份校验失败或者用户取消时的错误信息
$auth->setCancelText('身份校验失败！');
//开始进行用户身份校验
$auth->start();
//如果身份校验成功，显示信息
if($auth->getAuth())
{
      echo "身份校验成功，欢迎". $auth->username;
};
?>
```

myadmin 表的结构及其中的内容如下所示。

```
mysql> select * from myadmin;
+----------+----------+
| username | password |
+----------+----------+
| Simon    | 12345678 |
+----------+----------+
1 row in set (0.00 sec)
```

运行结果如图 2-3 所示。

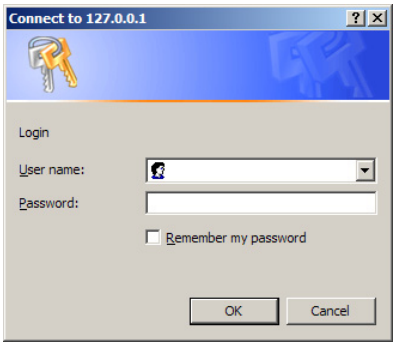

图 2-3　Auth_HTTP 的运行画面

可以看到，通过 setRealm 设置的文字信息在提示对话框上显示了出来。在对话框上正确的输入表中的数据以后，就可以看到“身份校验成功”的提示了。这里，在 User name 处输入 Simon，在 Password 处输入 12345678，可以看到如图 2-4 的结果。

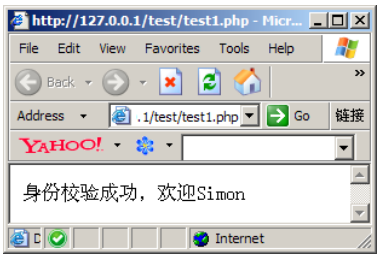

图 2-4　Auth_HTTP 的校验成功画面

从图 2-4 中可以看出，用户登陆时输入的用户名在页面上显示出来了，也就是$auth->username 发生了作用。

由于篇幅有限，Auth 以及 Auth_HTTP 类库就介绍到这里，对于这两种类库的其他功能就不一一介

绍了。在实际应用中，使用 Auth 以及 Auth_HTTP 类库可以有效的提供方便、安全的用户身份校验操作。读者可以通过查看该类库的使用手册学习到更多功能。

2.4.3　使用 HTML_Template_IT 类库进行模版替换

在进行一些网站设计时，往往需要对网站提供多模版界面，这样才能给访问者一种人性化的耳目一新的感觉。PEAR 中的 HTML_Template_IT 类库提供了方便的功能对网站进行模版替换。该类库要求提供模版文件，在模版中使用大括号“{}”标明要被替换的域。在程序中，使用 setVariable 函数对这些域进行替换。以下代码实现了一个简单的模版替换，程序根据地址栏参数的不同替换不同的模版。

模版一 T1.htm 的代码如下所示。

```
<html>
  <head>
    <title>{title}</title>
  </head>
  <body>
    <font color=red size=6><center>{title}</center></font>
    <hr>
    <pre>{body}</pre>
  </body>
</html>
```

模版二 T2.htm 的代码如下所示。

```
<html>
  <head>
    <title>{title}</title>
  </head>
  <body>
    <font color=green size=7>{title}</font>
    <pre>{body}</pre>
  </body>
</html>
```

PHP 实现代码如下所示。

```
<?php
require_once "HTML/Template/IT.php";
//创建新的 HTML_Template_IT 对象，其中参数为模版文件所在路径
$template = new HTML_Template_IT('templates/');
//读取模版文件，通过读取地址栏上的参数获得模版信息
$template->loadTemplateFile($_GET['template'].".htm");
//设置模版中的参数
$template->setVariable('title', 'HTML_Template_IT');
$template->setVariable('body', 'Hello World');
//显示页面
$template->show();
?>
```

上述代码对模版中的{title}和{body}进行了替换。通过 test1.php?template=T1 的形式访问可以得到如图 2-5 所示的结果，通过 test1.php?template=T2 的形式访问可以得到如图 2-6 所示的结果。

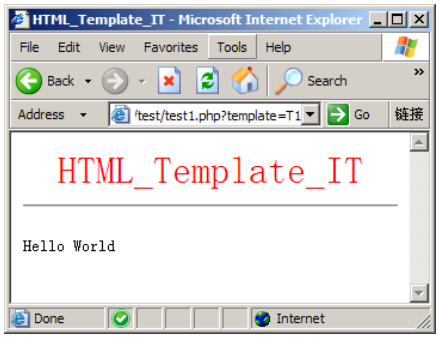

图 2-5　HTML_Template_IT 的模版一

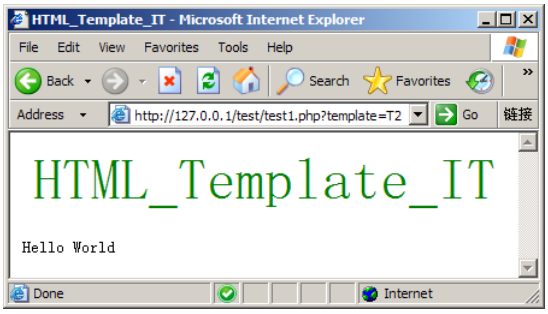

图 2-6　HTML_Template_IT 的模版二

2.5　PECL 的安装与使用

前面已经介绍了 PECL 与 PEAR 的区别。与 PEAR 不同，PECL 提供了使用其他编程语言编写的扩展库，以实现 PHP 难于完成的功能。

PECL 的安装与使用并不像 PEAR 那么容易。理论上说，PECL 扩展库的下载也是可以通过类似 pear 命令的 pecl 命令来完成。该命令的用法与 pear 命令的用法完全相同。需要注意的是 pecl 命令下载的文件为 PECL 包的源代码，pecl 命令将自动调用编译器对其进行编译。但是，由于 pecl 命令往往不能成功编译源代码，在实际应用中很少使用 pecl 命令来实现 pecl 扩展库的安装。

通常，下载 PECL 扩展库可以从其官方网站 http://pecl.php.net/下载，下载后往往是没有编译好的二进制文件。所以，在使用 PECL 之前需要准备相应的编译环境进行编译。编译前，也可以根据自身需要定制其代码。由于开发人员考虑到 Windows 平台编译的麻烦，在 http://pecl4win.php.net/网站提供了 PECL 的编译后的文件，用户可以直接下载使用。

将 PECL 源代码编译成 dll 文件后，将编译好的 dll 文件拷贝到 PHP 的扩展目录下，并修改 php.ini 文件，在其中加入类似下面的代码即可。

```
extension=<DLL filename>.dll
```

这里的 DLL filename 是编译好的 dll 文件。

修改好以后，重新启动 Apache 即可开始新下载的 PECL 了。为了确认扩展库的安装成功，可以通过如下测试页面查看扩展库的信息。

```
<?php
    phpinfo();
?>
```

下一节将以一个常见的 PECL 扩展库为例，简要介绍如何安装和使用一个 PECL 扩展。

2.6　PECL 扩展应用实例——Zip

Zip 扩展是 PECL 扩展库中最常用的 PHP 扩展之一，该扩展可以实现读取 ZIP 压缩文件的功能。

2.6.1　Zip 扩展的安装

Zip 扩展可以从 http://pecl.php.net/package/zip 网站下载到，下载后的文件为 Zip 扩展的源代码。对于 Windows 用户，可以从 http://pecl4win.php.net/直接下载相应版本的 DLL 文件。下载后，将编译过的 DLL 文件放在 PHP 安装目录下的 ext 文件夹中。修改 PHP 的配置文件 php.ini 并增加如下的配置信息。

```
extension=php_zip.dll
```

修改后，重新启动 Apache 服务器即可完成安装。在服务器上运行如下代码，可以检测 Zip 扩展库是否安装成功。

```
<?php
    phpinfo();
?>
```

运行结果如图 2-7 所示，可以看到 Zip 扩展已经安装成功了。

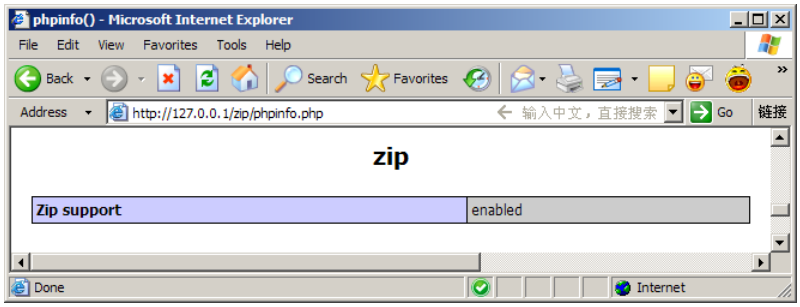

图 2-7　Zip 扩展库

2.6.2　一个 Zip 扩展应用实例

Zip 扩展提供了几个用于读取 Zip 文件的函数，在 PHP 中可以直接使用这些函数读取 Zip 压缩包中的每一个文件。以下代码实现了一个循环读取 Zip 压缩包中每一个文件的功能。

```
<?php
$zip = zip_open("test.zip");                                          //打开 zip 压缩包
while ($zip_entry = zip_read($zip))                                   //循环读取 zip 压缩包中的每一个文件
{
    zip_entry_open($zip, $zip_entry);                                 //打开 zip 压缩包中的文件
    $zip_entry_name = zip_entry_name($zip_entry);                     //获取文件名
    echo "文件名称：$zip_entry_name";
    $zip_entry_filesize = zip_entry_filesize($zip_entry);             //获取文件大小
    echo "文件大小：$zip_entry_name";
    $zip_entry_read = zip_entry_read($zip_entry, $zip_entry_filesize); //获取文件内容
    echo "文件内容：$zip_entry_read";
    zip_entry_close($zip_entry);                                      //关闭文件
}
zip_close($zip);                                                      //关闭 zip 压缩包
```

```
?>
```

代码运行后，压缩包 test.zip 中的每一个文件内容都将被显示在网页上。如果将每个文件内容保存到硬盘中，可以实现 Zip 压缩包的解压缩操作。

2.7　小结

本章主要介绍了 PEAR 扩展库的使用方法以及一些常用的 PEAR 扩展库。PEAR 扩展库在实际应用中非常常见，PEAR 扩展库充分体现了 PHP 代码重用的原则。PEAR 的存在大大加快了 PHP 程序员的开发效率。读者如果能够熟练的掌握 PEAR 的使用方法，可以大大提高实际开发中的开发效率和代码稳定性。对于本章没有介绍的 PEAR 类库，可以通过访问 http://pear.php.net 获取更多关于 PEAR 扩展库的信息。

对于 PECL 扩展库，往往为了实现一些直接用 PHP 难以实现的功能。在实际应用中，PECL 的使用较少。对于不同的 PECL 扩展库，读者可以通过参考 PECL 扩展附带的文档获取 PECL 扩展中的函数说明和语法格式。