Testing PHP with Perl

Chris Shiflett
shiflett@php.net

Geoffrey Young
geoff@modperlcookbook.org
PHP and Perl?

- Testing a basic PHP application
- Using the Apache-Test framework
Add a User

Username: geoff

Password: ******

Add
The Code
<?php

include ' ../../../functions.inc';

...

if (add_user($_POST['username'], $_POST['password'])) {
    echo '<p>User Added</p>);
    echo '<p><a href="/admin/">Admin Home</a></p>');?></p>
The Testing Paradigm

• Adopted from the time-tested Perl mythology (*sic*)

• `plan()` the number of tests

• `call ok()` for each test you plan
  – or `is()`, or `like()`, or `unlike()`, etc...

• Framework keeps track of the results and writes out the report

• Test Anything Protocol (TAP)
The Test
<?php
require 'test-more.php';
require "${_SERVER['DOCUMENT_ROOT']}/functions.inc";

plan(2);

{
    # no user or password
    $rc = add_user('', '');
    ok (!$rc, 'no user/pass fails');
}

{
    # some generic user/password
    $rc = add_user('user', 'password');
    ok ($rc, 'user/pass successfully added');

    # cleanup
    delete_user('user');
}?>
test-more.php

- Automagically generated
- Interface into Apache-Test
- Provides simple, intuitive functions
  - ok()
  - is()
  - like()
- Takes care of bookkeeping
  - plan()
ok ()

- Used for boolean comparisons
  ok($foo == $bar, '$foo equals $bar');

- Gives some diagnostic output on failure
  not ok 1 - no user/pass fails
  # Failed test (add_user.php at line 10)
Goodness

• No tests in application code
• Simple interface
• Repeatable
  – tests are self-contained
• No Perl involved
  – Chris particularly likes this aspect
Testing Ideology

• A good testing environment should provide
  – tools to make writing tests simple
  – a self-contained and pristine environment
  – test automation

• Basically do everything for you except write your tests
$ make test
$ make test
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST  -clean
APACHE_TEST_GROUP= APACHE_TEST_HTTPD= APACHE_TEST_PORT= APACHE_TEST_USER= APACHE_TEST_APXS=
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST  -bugreport -verbose=1
using Apache/1.3.32

waiting 60 seconds for server to start: ..
waiting 60 seconds for server to start: ok (waited 1 secs)
server localhost.localdomain:8529 started
t/functions/add_user....1..2
ok 1 - no user/pass fails
ok 2 - user/pass successfully added
ok
All tests successful.
Files=1, Tests=2, 1 wallclock secs ( 0.51 cusr + 0.10 csys = 0.61 CPU)
[warning] server localhost.localdomain:8529 shutdown
$

$ make test
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST -clean
APACHE_TEST_GROUP= APACHE_TEST_HTTPD= APACHE_TEST_PORT= APACHE_TEST_USER= APACHE_TEST_APXS = \n/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST -bugreport -verbose=1

using Apache/1.3.32

waiting 60 seconds for server to start: ..
waiting 60 seconds for server to start: ok (waited 1 secs)
server localhost.localdomain:8529 started
t/functions/add_user....1..2
ok 1 - no user/pass fails
ok 2 - user/pass successfully added
ok
All tests successful.
Files=1, Tests=2, 1 wallclock secs ( 0.51 cusr + 0.10 csys = 0.61 CPU)
[warning] server localhost.localdomain:8529 shutdown
$
$ make test
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST -clean
APACHE_TEST_GROUP= APACHE_TEST_HTTPD= APACHE_TEST_PORT= APACHE_TEST_USER= APACHE_TEST_APXS = \
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST -bugreport -verbose=1
using Apache/1.3.32

waiting 60 seconds for server to start: ..
waiting 60 seconds for server to start: ok (waited 1 secs)
server localhost.localdomain:8529 started
t/functions/add_user...1..2
ok 1 - no user/pass fails
ok 2 - user/pass successfully added
ok
All tests successful.
Files=1, Tests=2, 1 wallclock secs ( 0.51 cusr + 0.10 csys =  0.61 CPU)
[warning] server localhost.localdomain:8529 shutdown
$
$ make test
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST -clean
APACHE_TEST_GROUP= APACHE_TEST_HTTPD= APACHE_TEST_PORT= APACHE_TEST_USER= APACHE_TEST_APXS = \
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
t/TEST -bugreport -verbose=1
using Apache/1.3.32

waiting 60 seconds for server to start: ...
waiting 60 seconds for server to start: ok (waited 1 secs)
server localhost.localdomain:8529 started
t/functions/add_user....1..2
ok 1 - no user/pass fails
ok 2 - user/pass successfully added
ok
All tests successful.
Files=1, Tests=2,  1 wallclock secs ( 0.51 cusr + 0.10 csys =  0.61 CPU)
[warning] server localhost.localdomain:8529 shutdown
$
$ make test

/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \
 t/TEST -clean
APACHE_TEST_GROUP= APACHE_TEST_HTTPD= APACHE_TEST_PORT= APACHE_TEST_USER= APACHE_TEST_APXS = \\n/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \n t/TEST -bugreport -verbose=1
using Apache/1.3.32

waiting 60 seconds for server to start: ..
waiting 60 seconds for server to start: ok (waited 1 secs)
server localhost.localdomain:8529 started
t/functions/add_user....1..2
ok 1 - no user/pass fails
ok 2 - user/pass successfully added
ok
All tests successful.
Files=1, Tests=2,  1 wallclock secs ( 0.51 cusr + 0.10 csys =  0.61 CPU)
[warning] server localhost.localdomain:8529 shutdown

$
$ make test
/perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \ 
t/TEST -clean
APACHE_TEST_GROUP= APACHE_TEST_HTTPD= APACHE_TEST_PORT= APACHE_TEST_USER= APACHE_TEST_APXS=" /perl/perl-5.8.5/bin/perl -Ilib/arch -Ilib/lib \ 
t/TEST -bugreport -verbose=1
using Apache/1.3.32

waiting 60 seconds for server to start: ..
waiting 60 seconds for server to start: ok (waited 1 secs)
server localhost.localdomain:8529 started

t/functions/add_user....1..2
ok 1 - no user/pass fails
ok 2 - user/pass successfully added
ok
All tests successful.
Files=1, Tests=2, 1 wallclock secs ( 0.51 cusr + 0.10 csys =  0.61 CPU)
[warning] server localhost.localdomain:8529 shutdown
Behold the Power of Perl

• How did we do it?
• Apache-Test
Apache-Test

• Framework for testing Apache-based application components
• Part of the httpd-test ASF project
• Provides tools to make testing Apache simple
• Written in Perl
Apache Foo

• Apache needs a basic configuration to service requests
  – ServerRoot
  – DocumentRoot
  – ErrorLog
  – Listen

• Apache-Test "intuits" these and creates its own httpd.conf

• Uses an httpd binary you specify
  – patience, young grasshopper
Cross Pollination

• Apache-Test provides a default php.ini
  -php.ini-recommended

• Also provides test-more.php
  <?php require 'test-more.php'; ?>
  -modified include_path

• Fertile soil so your PHP can grow
Altering the Defaults

• `httpd.conf` and `php.ini` are autogenerated
  – don't touch them

• Supplement default `httpd.conf` and `php.ini` with custom configurations

• Create `t/conf/extra.conf.in`
extra.conf.in

- Same directives as `httpd.conf`
- Pulled into `httpd.conf` via `Include`
- Allow for some fancy variable substitutions
extra.conf.in

AddType application/x-httpd-php .php
DirectoryIndex index.php index.html

<IfModule @PHP_MODULE@>
    php_flag register_globals On
</IfModule>
AddType application/x-httpd-php .php
DirectoryIndex index.php index.html

<IfModule mod_php5.c>
    php_flag register_globals On
</IfModule>
So Far...

• We have
  – PHP test script (*add_user.php*)
  – test library (*test-more.php*)
  – *httpd.conf*
  – *php.ini*
  – local overrides

• We still need
  – a client to call the PHP script
  – a running server
The Gory Details
The Gory Details

- Create PHP scripts as
  
  \texttt{t/response/TestFunc/add\_user.php}

- Apache-Test \textbf{will automagically create a client script that calls} \texttt{add\_user.php}
  
  \texttt{t/func/add\_user.t}

- \textbf{make test will}
  
  - \texttt{run add\_user.t}
  
  - \texttt{which will request add\_user.php}
  
  - \texttt{which will send data to Apache-Test}
Makefile.PL

• We still need to create the Makefile
  – so make test works

• We also need to choose an Apache installation

• Taken care of in one single step
  perl Makefile.PL -httpd /path/to/httpd
Cool!

• The glory will sink in tomorrow
  – we hope
• PHP development will never be the same
• See what happens when a Perl guy and a PHP guy start drinking?
Code

• All the code from this presentation can be found here


• Be sure to read the README and INSTALL docs
Brought To You By...

http://shiflettt.org/
http://modperlcookbook.org/~geoff/