

# Warming up to modular testing

Raleigh.pm 2009-03-19  
Brad Oaks of Plus Three, LP

# Harness to run the show

```
./bin/simple_test --files=t/pod.t
```

```
#!/usr/bin/env perl
```

```
use strict;
```

```
use warnings;
```

```
use Test::Harness;
```

```
my @files = glob("t/pod.t");
```

```
my $harness = TAP::Harness->new(
```

```
    {verbosity => 1, merge => 0}
```

```
);
```

```
$harness->runtests(@files);
```

# TAP::Harness can

- run multiple test jobs at once (`aggregate_tests`)
- try to emit results with color
- be verbose or varying levels of quiet

see "`perldoc TAP::Harness`"

`Test::Harness` is for backwards compatibility.  
For new work, you should start with `TAP::Harness`.

# package Arcos::Test::Class

```
use base 'Test::Class';
use Arcos::Test;
use Arcos::TestData;
use Test::Builder qw(ok is_eq);
use Test::LongString;

sub tc_startup : Test(startup) {
    my $self = shift;
    $self->{td} = Arcos::TestData->new; }
sub tc_shutdown : Test(shutdown) {
    my $self = shift;
    $self->{td}->cleanup(ignore_deleted => 1); }
```

# Arcos::Test::Class

We provide a number of methods at this level:

- `contains_url()`
- `lacks_url()`
- `json_success()`
- `json_failure()`
- `json_contains()`
- `json_lacks()`
- `input_value_is()`

```
local $Test::Builder::Level = $Test::Builder::Level + 1;
```

# Test::Builder

```
t/lib/Arcos/Test/Class.pm  
use Test::Builder qw(ok is_eq);  
# make Test::Builder give the right line number for failures  
local $Test::Builder::Level = $Test::Builder::Level + 1;
```

Also provides plan

- `$Test->plan('no_plan');`
- `$Test->plan( skip_all => $reason );`
- `$Test->plan( tests => $num_tests );`

# An example of inherited tests

Arcos::Form::Test is the parent of

- Arcos::Form::ContactOfficial::Test
- Arcos::Form::Register::Test

They each have

use base 'Arcos::Form::Test';

```
sub tc_startup : Test(startup) {  
    $self->{td} = Arcos::TestData->new();  
    return $self->SUPER::tc_startup;  
}
```

# package Arcos::Form::Test

```
sub tc_startup : Test(startup) {  
  sub tc_shutdown : Test(shutdown) {  
    sub x_create_form : Test(startup => 3) {  
  
      sub additional_elements { "" }  
      sub form_db_class { "" }  
      sub mech_shows_custom_fields  
      sub person_has_custom_fields  
      sub hit_form  
      sub custom_field_default  
      sub custom_fields_for_mech
```



# t/register\_form.t

```
use strict;
```

```
use warnings;
```

```
use Arcos::Test;
```

```
use Arcos::Test::Script;
```

```
use Arcos::Form::Register::Test;
```

```
Arcos::Test->needs_running_krang();
```

```
Arcos::Test->needs_running_apache();
```

```
Arcos::Form::Register::Test->new->runtests;
```

# t/contact\_official\_form.t

```
use Arcos::GeoCoder;
my $coder = Arcos::GeoCoder->new;
if (!$coder->available) {
    Test::More::plan(skip_all =>
        'geocoding databases not available.');
```

```
};
unless (Arcos::Test->stop_queue()) {
    Test::More::plan(skip_all =>
        'Could not stop Queue daemon in time.');
```

```
};
Arcos::Form::ContactOfficial::Test->new->runtests;
Arcos::Test->restart_queue;
```

# separating tests within a module

```
t/pac_contribution_form-authorize.net.t
```

```
Arcos::PACContributionForm::Test::AuthorizeNet
```

```
Arcos::PACContributionForm::Test
```

```
  b_submit_empty_form : Test(no_plan)
```

```
  b_require_security_code : Test(5)
```

```
  c_address_validation : Test(21)
```

```
  e_amount_other_validation : Test(31)
```

```
  f_two_names_in_first_name : Test(3)
```

```
  g_card_security_code : Test(6)
```

# return to skip

return

'The default templates do not currently have security code; skipping for now.';

return 'You have no test payment account.'  
unless \$self->test\_account;

return 'The deleted form test is not implemented for this form type yet.';

# plan to skip

```
use Arcos::Conf qw(ContributionTestMode);
Arcos::Test->needs_run_daily();
if (!ContributionTestMode) {
    plan(skip_all => 'contributions not in test mode');
} elsif (!Arcos::Test->can_reach_url(
    'https://www.vancodev.com:443/cgi-bin/wstest.vps')) {
    plan(skip_all => "Can't reach https://www.vancodev.com:443/cgi-
bin/wstest.vps");
} else {
    Arcos::ContributionForm::Test::Vanco->runtests();
}
```

# calling plan from Arcos::Test

```
if ($ENV{'RUN_DAILY_TESTS'}) {  
  if ($ENV{'OVERRIDE_DAILY_TESTS'} or $class->_run_daily()) {  
    $class->_update_run_daily_timestamp();  
    Test::More::plan(@args);  
  }  
} else {  
  Test::More::plan(  
    skip_all => 'RUN_DAILY_TESTS environment variable not set;  
    skipping daily tests.');
```

# needs\_run\_daily

```
if ($ENV{'RUN_DAILY_TESTS'}) {  
  if ($ENV{'OVERRIDE_DAILY_TESTS'} or $class->_run_daily()) {  
    $class->_update_run_daily_timestamp();  
    Test::More::plan(@args);  
  }  
} else {  
  Test::More::plan(  
    skip_all => 'RUN_DAILY_TESTS environment variable not set;  
skipping daily tests.');
```

# a lot of reuse

```
use base 'Arcos::Report::SiteEvent::Test';
```

```
Arcos::Report::Volunteer::Test
```

```
Arcos::Report::Contribution::Test
```

```
Arcos::Report::ContactOfficial::Test
```

```
Arcos::Report::Petition::Test
```

```
Arcos::Report::Tellafriend::Test
```

```
Arcos::Report::Subscription::Test
```

```
Arcos::Report::RadioCall::Test
```

```
Arcos::Report::LTE::Test
```

```
Arcos::Report::ContactUs::Test
```



# TEST\_METHOD

```
TEST_METHOD=a_good_file bin/arcos_test --files=t/admin-  
listupload.t
```

```
TEST_METHOD=e_failed_contributions bin/arcos_test --  
files=t/report-contribution.t
```

```
TEST_METHOD=k_listing_checkbox bin/arcos_test --  
files=t/admin-mailingmessage.t
```

```
TEST_METHOD='(a|b)_.*' bin/arcos_test --files=t/warehouse-  
loader.t
```

# pausing to look around

```
warn 'pausing to look around;  
    hit return to continue:';  
my $PAUSE = <STDIN>;
```

This will keep the test from cleaning up before you've had a chance to look around.

Add a few warns of URL values and you can bring up the reports in a browser to get a better idea of what's going on.

# Additional Resources

<http://www.modernperlbooks.com/mt/2009/03/organizing-test-suites-with-testclass.html>

Organizing Test Suites with Test::Class