WORKING WITH A DATABASE IN UNIT TESTS WHEN YOU HAVE A LARGE TEST SUITE, HOW TO WRAP PROVE AND MAKE STUFF AVAILABLE,

(thanks Rick)

Julien Fiegehenn (simbabque)
simbabque@cpan.org

ASSUMPTIONS

- existing code base
- application with MySQL DB
- Mojolicious
- lots of tests (Test::Mojo)

TESTS USE A REMOTE DATABASE

- slow
- inflexible
- error prone
- doesn't work well for more than one dev
- try working from a German high speed train

Basic app

```
#!/usr/bin/env perl
use Mojolicious::Lite;

use feature 'state';
use lib 'lib';

get '/' => sub {
    my $c = shift;
    $c->render( text => $c->db->resultset('Foo')->last->message );
};
```

Database model

```
use MyApp::Schema;

helper db => sub {
    my $c = shift;

    state $schema = MyApp::Schema->connect(
        'dbi:SQLite:myapp.db',
        'user',
        'password',
    ) or die;

    return $schema;
};
```

Adding a message

```
get '/add/:message' => sub {
    my $c = shift;
    $c->db->resultset('Foo')->create(
        { message => $c->param('message') }
    );
    $c->redirect_to('/');
};
```

- Always show the latest message
- Add a message

LIVE DEMO APP BEFORE

TESTS

```
$ prove -ls t
t/001.t .. ok
All tests successful.
Files=2, Tests=15,  1 wallclock secs
Result: PASS
```

RUN THEM AGAIN

```
$ prove -ls t
t/001.t .. 1/?
#   Failed test 'content is similar'
#   at t/001.t line 8.
#                   'unittest'
#     doesn't match '(?^:Hello)'
# Looks like you failed 1 test of 8.
t/001.t .. Dubious, test returned 1 (wstat 256, 0x100)
Failed 1/8 subtests

Test Summary Report
-------------------
t/001.t (Wstat: 256 Tests: 8 Failed: 1)
  Failed test:  3
  Non-zero exit status: 1
Files=1, Tests=8,  0 wallclock secs
Result: FAIL
```

OOPS! WE'VE CHANGED PRODUCTION DATA!

1. MAKE DB CONFIGURABLE

```
helper db => sub {
    my $c = shift;

    state $schema = do {
        my $config = plugin 'Config';
        MyApp::Schema->connect(
            $config->{dsn},             # this is now coming
            $config->{user},            # from a config file
            $config->{password},        #
        ) or die;
    };

    return $schema;
};
```

MYAPP.CONF

```
{
    dsn      => 'dbi:SQLite:myapp.db',
    user     => 'user',
    password => 'password',
}
```

2. CREATE A TEMPORARY CONFIG

```
#!/usr/bin/env perl
use strict;
use warnings;
use App::Prove;

# ...

# run prove
my $app = App::Prove->new;
$app->process_args(@ARGV);
exit $app->run ? 0 : 1;
```

2.1 CREATE A TEMPORARY TEST DATABASE

```
use FindBin;

use lib "$FindBin::Bin/../lib";

# create temorary DB
use Test::DBIx::Class {
    schema_class => 'MyApp::Schema',
    connect_info => [ 'dbi:SQLite:dbname=unittest.db', '', '' ],
};


# insert fixtures
ResultSet('Foo')->create( { message => "Hello World" } );
```

2.2 BUILD THE CONFIG HASH

```
# grab connection info from Schema
my $connect_info = Schema->storage->connect_info->[0];
my $config       = {
    dsn      => $connect_info->{dsn},
    user     => $connect_info->{user},
    password => $connect_info->{password},
};
```

2.3 WRITE IT TO A TEMPORARY FILE

```
use File::Temp 'tempfile';
use Data::Dump 'pp';

# ...

# write config to a temporary file
my ( $fh, $config_file ) = tempfile(
    'tmpconfigXXXX',
    DIR    => '.',
    UNLINK => 1,
);
print $fh pp $config;
close $fh or die $!;
```

2.4 USE THE CONFIG FILE

```
# force mojo to use temporary config
$ENV{MOJO_CONFIG} = $config_file;
```

LIVE DEMO APP AFTER

WITH MOJO

- every `.t` will start a new app
- all apps will use this DB
- use `$t->app->db` to get the schema object

CATALYST

- `$ENV{ MYAPP_CONFIG_LOCAL_SUFFIX }`
- C::Plugin::ConfigLoader placeholders (`__ENV()__`)

DANCER2

- `$ENV{DANCER_ENVIRONMENT}`
- `$ENV{DANCER_CONFDIR}` and `$ENV{DANCER_ENVDIR}`

EXTENDING

- use arbitary fixture modules
- use MySQL or Postgres database server spawning with Test::DBIx::Class
- or roll your own with Test::mysqld or Test::PostgreSQL

THANK YOU

Slides will be on github soon.

https://github.com/simbabque/talk-testing-with-db-mojo