Using closures with DBIx::Class transactions

A Lightning Talk

Ewen McNeill
<ewen@naos.co.nz>

Naos Ltd

2007/02/13 — Perl Mongers, Wellington

Trivial Database

balance.sql

```
-- Trivial balance table to demonstrate need for transactions

CREATE TABLE balance (
        client_name     VARCHAR(20),
        balance_cents   INTEGER
);

INSERT INTO balance (client_name, balance_cents)
            VALUES ('alice',     15000        );

INSERT INTO balance (client_name, balance_cents)
            VALUES ('bob',        7500        );
```

- Load with: sqlite3 balance.db <balance.sql

Trivial DBIx::Class environment

DB/Main.pm

```
package DB::Main;

use base qw/DBIx::Class::Schema/;
__PACKAGE__->load_classes();

1;
```

DB/Main/Balance.pm

```
package DB::Main::Balance;

use base qw/DBIx::Class/;

__PACKAGE__->load_components(qw/PK::Auto Core/);
__PACKAGE__->table('balance');
__PACKAGE__->add_columns(qw/ client_name balance_cents /);
__PACKAGE__->set_primary_key('client_name');

1;
```

Naive Money Transfer

```
#! /usr/bin/perl -w
# Transfer money from Alice to Bob (naive version -- no transactions)

use DB::Main;

sub set_balance {
  my ($schema, $params) = @_;
  my $rs = $schema->resultset('Balance')->find($params->{client_name});
     $rs->balance_cents($params->{balance_cents});
     $rs->update;
}

my $schema = DB::Main->connect('dbi:SQLite:balance.db', '', '', { AutoCommit => 1 });

# Desired new values
my $from = { 'client_name' => 'alice', 'balance_cents' => 12500 };
my $to   = { 'client_name' => 'bob',   'balance_cents' => 10000 };

set_balance($schema, $from);
set_balance($schema, $to);
```

Simplistic Transaction

```
#! /usr/bin/perl -w
# Transfer money from Alice to Bob (raw begin/commit transaction)

use DB::Main;

sub set_balance {
  my ($schema, $params) = @_;
  my $rs = $schema->resultset('Balance')->find($params->{client_name});
     $rs->balance_cents($params->{balance_cents});
     $rs->update;
}

my $schema = DB::Main->connect('dbi:SQLite:balance.db', '', '', { AutoCommit => 1 });

# Desired new values
my $from = { 'client_name' => 'alice', 'balance_cents' =>  9500 };
my $to   = { 'client_name' => 'bob',   'balance_cents' => 13000 };

$schema->txn_begin();
set_balance($schema, $from);
set_balance($schema, $to);
$schema->txn_commit();
```

More sophisticated Transaction

```
#! /usr/bin/perl -w
# Transfer money from Alice to Bob (txn_do transaction usage)

use DB::Main;

sub set_balance {
  my ($schema, $params) = @_;
  my $rs = $schema->resultset('Balance')->find($params->{client_name});
     $rs->balance_cents($params->{balance_cents});
     $rs->update;
}

sub do_updates {
  my ($schema, $from, $to) = @_;
  set_balance($schema, $from);
  set_balance($schema, $to);
}

my $schema = DB::Main->connect('dbi:SQLite:balance.db', '', '', { AutoCommit => 1 });

# Desired new values
my $from = { 'client_name' => 'alice', 'balance_cents' =>  7500 };
my $to   = { 'client_name' => 'bob',   'balance_cents' => 15000 };

$schema->txn_do(\&do_updates, $schema, $from, $to);
```

Now with closures

```
#! /usr/bin/perl -w
# Transfer money from Alice to Bob (txn_do closure transaction)

use DB::Main;

sub set_balance {
  my ($schema, $params) = @_;
  my $rs = $schema->resultset('Balance')->find($params->{client_name});
     $rs->balance_cents($params->{balance_cents});
     $rs->update;
}

my $schema = DB::Main->connect('dbi:SQLite:balance.db', '', '', { AutoCommit => 1 });

# Desired new values
my $from = { 'client_name' => 'alice', 'balance_cents' =>  2500 };
my $to   = { 'client_name' => 'bob',   'balance_cents' => 20000 };

$schema->txn_do(sub {
  set_balance($schema, $from);
  set_balance($schema, $to);
});
```