Effective Usage of DBIx::Class

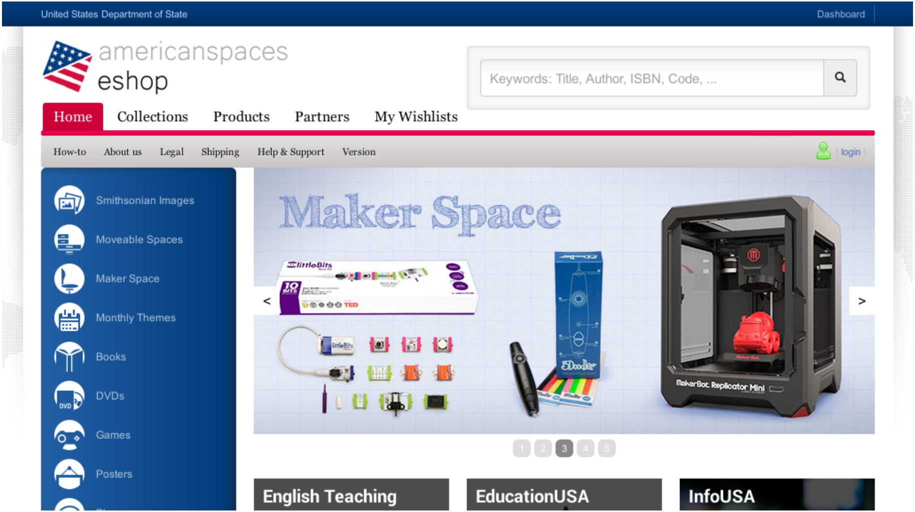
Stefan Hornburg (Racke)

Alpine Workshop 2.9.2016

Me

- Nick name: racke
- Self employed programmer since 1998
- Web applications
- Ecommerce
- Database
- Customers in Germany, Switzerland, USA, ...

eShop



United States Department of State

Dashboard

americanspaces eshop

Keywords: Title, Author, ISBN, Code, ...

Home | Collections | Products | Partners | My Wishlists

How-to | About us | Legal | Shipping | Help & Support | Version | login

- Smithsonian Images
- Moveable Spaces
- Maker Space
- Monthly Themes
- Books
- DVDs
- Games
- Posters

English Teaching

EducationUSA

InfoUSA

Interchange

Interchange

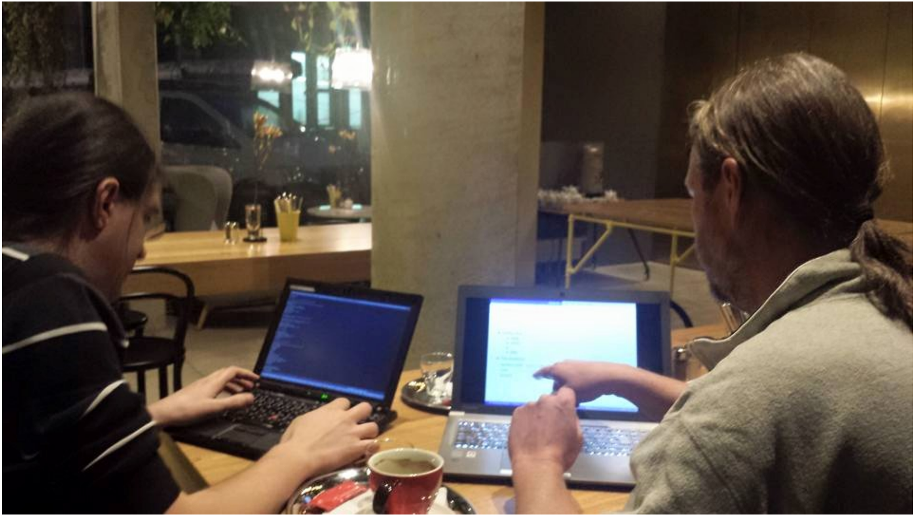
- Embedded SQL
- ~~Modern Perl~~

Dancer / DBIx::Class

- Dancer
- DBIx::Class
- Interchange6

Perl Dancer Conference

DBIx::Class Training



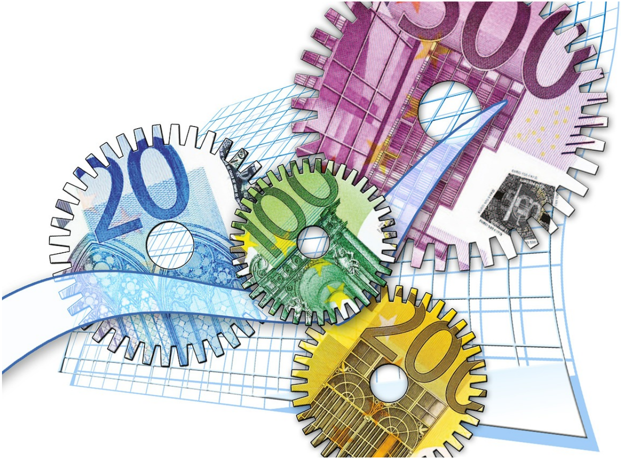
Conference website

- ACT 2014
  - Legacy code
  - No server control
- www.perl.dance 2015
  - Dancer / DBIx::Class
  - Most ACT features + new features
  - `https://github.com/interchange/Perl-Dancer-Conference`

DBIx::Class Features

- OO instead of SQL
- Abstracts different SQL implementations
- Business Logic
- Performance
- “ResultSet” features
- Ecosystem
    - IRC / Mailing list
    - Components
    - Helpers

Business Logic



Business Logic Benefits

- Multiple Consumers
    - Web application(s)
    - Cron jobs, scripts
    - Test environments
- Hide Implementation
    - SQL Dialects
    - Application View
- Changes / DRY

Performance

- Experience
- Test results
- Use cases

Literal SQL

```
$schema->resultset('PlaceVisited')->search(
{
  users_id => {
    -in => \[
       'SELECT users_id FROM users WHERE users_id > ?',
       2
    ]
  }
});
```

SQL Headache

Set Based DBIx::Class

Arthur Axel "fREW" Schmidt

http://www.perladvent.org/2012/2012-12-21.html

Set Based DBIx::Class

- ORM
  - Objects
  - Abstracts SQL
  - ...
- ResultSet
  - Chaining
  - Relationship Traversal
  - Correlated Subqueries
  - ...

Using ResultSet

- Simple SQL query
- Turn into DBIx::Class search
- Use ResultSet features

Simple SQL query

List of talks on last day of Perl Dancer Conference:

```
SELECT talks_id, author_id, conferences_id, duration,
title, tags, abstract, url, comments, accepted, confirmed,
lightning, scheduled, start_time, room, survey_id
FROM talks
WHERE accepted is TRUE
AND conferences_id = 1
AND room != ’’
AND start_time >= ’2015-10-22 00:00:00’
AND start_time <= ’2015-10-23 00:00:00’
```

Simple SQL query

With DBIx::Class:

```
my $talks = $schema->resultset('Talk')->search(
    {
        -bool          => 'accepted',
        conferences_id => 1,
        room           => { '!=' => '' },
        start_time     => {
            '>=' => '2015-10-22 00:00:00'
            '<=' => '2015-10-23 00:00:00'
            },
    },
);
```

Truth of ResultSet

~~ResultSet~~

```
isa(Query Plan);
```

Truth of ResultSet

This doesn’t execute SQL:

```
my $talks = $schema->resultset(’Talk’)->search(...);
```

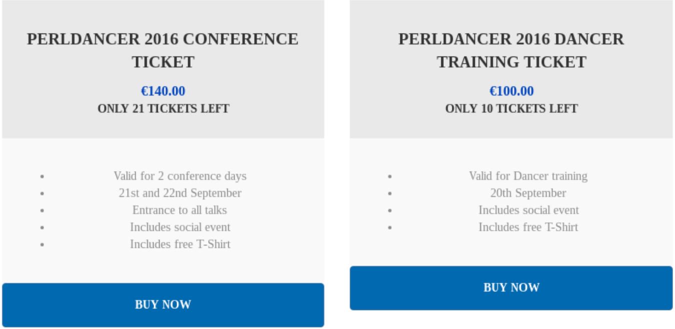
This does:

```
my $first_talk = $schema->resultset(’Talk’)
                 ->search(...)->first;
```

Tickets

Eintrittskarte
Aufbewahren und auf
Verlangen vorzeigen
050590
050590
050590

Tickets



PERLDANCER 2016 CONFERENCE TICKET

€140.00

ONLY 21 TICKETS LEFT

- Valid for 2 conference days
- 21st and 22nd September
- Entrance to all talks
- Includes social event
- Includes free T-Shirt

PERLDANCER 2016 DANCER TRAINING TICKET

€100.00

ONLY 10 TICKETS LEFT

- Valid for Dancer training
- 20th September
- Includes social event
- Includes free T-Shirt

Tickets with inventory

- Conference
- Tickets
- Conference Tickets
- Inventory

Tickets with inventory

```
SELECT ticket.sku, ticket.name,
  ticket.short_description, ticket.description,
  ticket.price, inventory.sku, inventory.quantity
  FROM conference_tickets me
  JOIN products ticket
    ON ticket.sku = me.sku
  LEFT JOIN inventories inventory
    ON inventory.sku = ticket.sku
WHERE me.conferences_id = ’2’ AND ticket.active = ’1’
```

Chaining

Tickets

```
$tickets = $schema->resultset( 'Conference' )
   ->find( {name => 'Perl Dancer Conference 2016'}, )
   ->tickets
   ->active
   ->prefetch('inventory');
```

Tickets - Relationship Traversal

```
package PerlDance::Schema::Result::Conference;

...

has_many
  conference_tickets =>
     'PerlDance::Schema::Result::ConferenceTicket',
  "conferences_id";

many_to_many tickets => "conference_tickets", "ticket";
```

Tickets - Predefined Search

```
package Interchange6::Schema::ResultSet::Product;

...

sub active {
    return $_[0]->search({ $_[0]->me(’active’) => 1 });
}
```

DBIx::Class::Helper::ResultSet::Me

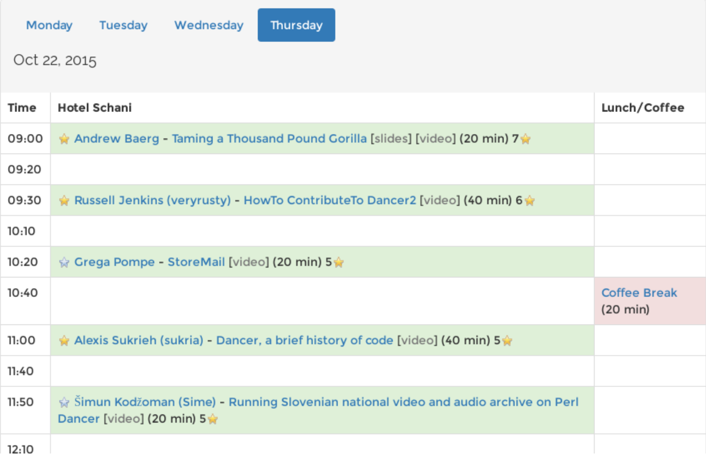
Tickets - Helpers

DBIx::Class::Helper::ResultSet::Shortcut

```
$tickets_rs->search( undef, { prefetch => ’inventory’ } );

$tickets_rs->prefetch(’inventory’);
```

Talks with number of attendees



Monday Tuesday Wednesday Thursday

Oct 22, 2015

Time	Hotel Schani	Lunch/Coffee
09:00	★ Andrew Baerg - Taming a Thousand Pound Gorilla [slides] [video] (20 min) 7★	
09:20		
09:30	★ Russell Jenkins (veryrusty) - HowTo ContributeTo Dancer2 [video] (40 min) 6★	
10:10		
10:20	☆ Grega Pompe - StoreMail [video] (20 min) 5★	
10:40		Coffee Break (20 min)
11:00	★ Alexis Sukrieh (sukria) - Dancer, a brief history of code [video] (40 min) 5★	
11:40		
11:50	☆ Šimun Kodžoman (Sime) - Running Slovenian national video and audio archive on Perl Dancer [video] (20 min) 5★	
12:10		

The naive way

```
my @data = map +{
   %{ $_->as_hash },
   attendee_count => $_->attendee_talks->count,
}, $talks_rs->all
```

Problem: one extra query per talk

Other solution: `GROUP BY` *(not recommended)*

Correlated Subqueries

```
package PerlDance::Schema::ResultSet::Talk;

sub with_attendee_count {
  my $self = shift;

  $self->search(
    undef,
      {
        ’+columns’ => {
           attendee_count =>
             $self->correlate(’attendee_talks’)
               ->count_rs
               ->as_query
        }
      }
  );
}
```

Correlated Subqueries

```
(
  SELECT COUNT( * )
  FROM "attendee_talks" "attendee_talks_alias"
  WHERE "attendee_talks_alias"."talks_id" = "me"."talks_id"
),
```

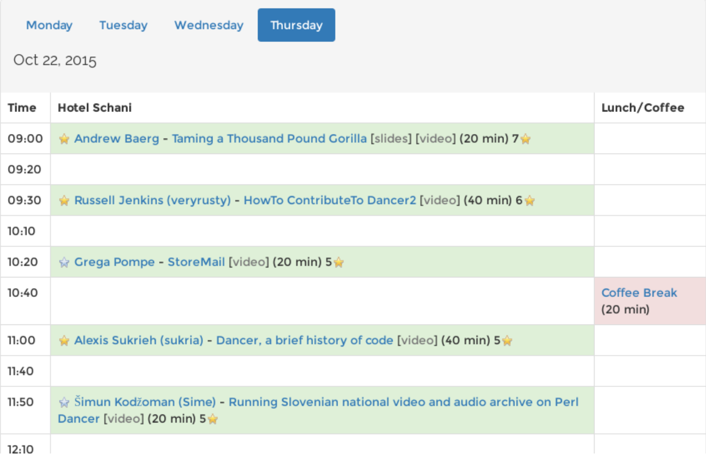
Correlated Subqueries

```
my $talks = rset(’Talk’)->search(
  {
    accepted       => 1,
    conferences_id => setting(’conferences_id’),
    room           => { ’!=’ => ’’ },
    start_time     => {
      ’!=’ => undef,
      ’>=’ => $schema->format_datetime($dt_date),
      ’<=’ =>
         $schema->format_datetime(
           $dt_date->clone->add( days => 1 ) )
    },
  },
  {
    order_by => ’start_time’,
    prefetch => ’author’,
  }
)->with_attendee_count;
```

Correlated Subqueries

`DBIx::Class::Helper::ResultSet::CorrelateRelationship`

Talks with number of attendees



Monday | Tuesday | Wednesday | **Thursday**

Oct 22, 2015

Time	Hotel Schani	Lunch/Coffee
09:00	★ Andrew Baerg - Taming a Thousand Pound Gorilla [slides] [video] (20 min) 7★	
09:20		
09:30	★ Russell Jenkins (veryrusty) - HowTo ContributeTo Dancer2 [video] (40 min) 6★	
10:10		
10:20	☆ Grega Pompe - StoreMail [video] (20 min) 5★	
10:40		Coffee Break (20 min)
11:00	★ Alexis Sukrieh (sukria) - Dancer, a brief history of code [video] (40 min) 5★	
11:40		
11:50	☆ Šimun Kodžoman (Sime) - Running Slovenian national video and audio archive on Perl Dancer [video] (20 min) 5★	
12:10		

Correlated Subqueries

```
sub with_attendee_status {
  my ( $self, $users_id ) = @_;

  $self->search(
    undef,
    {
      ’+columns’ => {
        attendee_status =>
          $self->correlate(’attendee_talks’)->search(
          {
            users_id => $users_id
          }
        )->count_rs->as_query
      }
    }
  );
}
```

Correlated Subqueries

```
$talks = $talks->with_attendee_status( $user->id );

(
  SELECT COUNT( * )
    FROM attendee_talks attendee_talks_alias
    WHERE attendee_talks_alias.talks_id = me.talks_id
    AND users_id = ’4’
),
```

Correlated Subqueries

```
$talks = $talks->with_attendee_status( $user->id );
```

Correlated Subqueries

```
my $talks = rset('Talk')->search(
  {
    accepted       => 1,
    conferences_id => setting('conferences_id'),
    room           => { '!=' => '' },
    start_time     => {
      '!=' => undef,
      '>=' => $schema->format_datetime($dt_date),
      '<=' =>
         $schema->format_datetime(
           $dt_date->clone->add( days => 1 ) )
    },
  },
  {
    order_by => 'start_time',
    prefetch => 'author',
  }
)->with_attendee_count->with_attendee_status( $user->id );
```

Correlated Subqueries

```
SELECT me."talks_id", me."author_id", me."conferences_id",
    SELECT COUNT( * )
      FROM "attendee_talks" "attendee_talks_alias"
    WHERE "attendee_talks_alias"."talks_id" = me."talks_id"
   ), (
    SELECT COUNT( * )
      FROM "attendee_talks" "attendee_talks_alias"
    WHERE "attendee_talks_alias"."talks_id" = me."talks_id"
   ), "author"."users_id", "author"."username", "author"."n
  FROM "talks" me LEFT JOIN "users" "author"
    ON "author"."users_id" = me."author_id"
WHERE "accepted" = ’1’ AND "conferences_id" = ’1’ AND "room
AND "start_time" IS NOT NULL AND "start_time" <= ’2015-10-2
AND "start_time" >= ’2015-10-22 00:00:00’
ORDER BY "start_time"
```

Predefined Searches

```
my $talks = rset(’Talk’)->search(
  {
    accepted       => 1,
    conferences_id => setting(’conferences_id’),
    room           => { ’!=’ => ’’ },
    start_time     => {
      ’!=’ => undef,
      ’>=’ => $schema->format_datetime($dt_date),
      ’<=’ =>
         $schema->format_datetime(
           $dt_date->clone->add( days => 1 ) )
    },
  },
  {
    order_by => ’start_time’,
    prefetch => ’author’,
  }
)->with_attendee_count->with_attendee_status( $user->id );
```

Predefined Searches

```
package PerlDance::Schema::ResultSet::Talk;

sub scheduled_for {
  my ( $self, $dt_date ) = @_;
  my $schema = $self->result_source->schema;

  $self->search(
    {
      start_time     => {
        ’!=’ => undef,
        ’>=’ => $schema->format_datetime( $dt_date ),
        ’<=’ =>
                $schema->format_datetime(
                  $dt_date->clone->add( days => 1 ) ),
      },
    }
  );
}
```

Predefined Searches

```
my $talks = rset(’Talk’)->search(
  {
    accepted       => 1,
    conferences_id => setting(’conferences_id’),
    room           => { ’!=’ => ’’ },
  },
  {
    order_by => ’start_time’,
    prefetch => ’author’,
  }
)
->scheduled_for( $dt_date )
->with_attendee_count
->with_attendee_status( $user->id );
```

Conclusion

- Resources
- Slides
- Perl Dancer Conference 2016
- Questions

Resources

- Extensive Documentation
  - `DBIx::Class::Manual::*`
  - `DBIx::Class::Manual::ResultClass`
  - `DBIx::Class::ResultSet`
  - `DBIx::Class::Relationship::*`
- Helpers by fREW
  - https://metacpan.org/pod/DBIx::Class::Helpers
- Perl Advent Calendar 2012: Set-based DBIx::Class by fREW

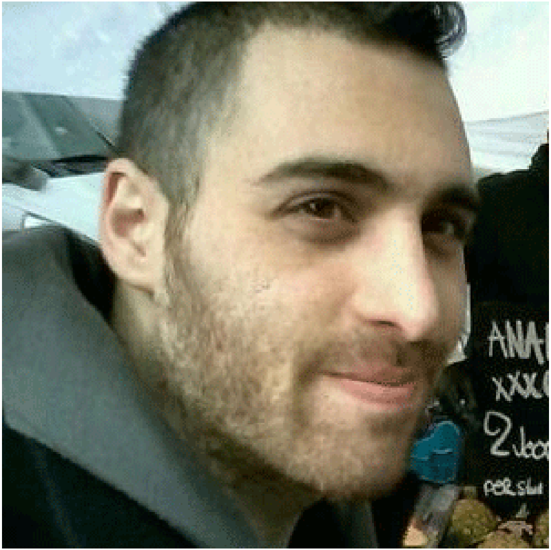
Slides

Slides: `https://www.linuxia.de/talks/alpine2016/dbic-pr-en-beamer.pdf`

Perl Dancer Conference 2016

PERL DANCER 2016
VIENNA, AUSTRIA
SEPT 20 - SEPT 22

Dancer Training



New People

NLPW
2016
THE Perl FOUNDATION
craigslist
cPanel
Booking.com
SPEAKER

Questions

Questions ?