

Modern Web Development in Perl

What we'll cover

- A Brief History of Perl
- Perl's Strengths and Weaknesses
- Building a small web app and API with Dancer2

Who am I to take over the next hour
of your life?

Jason A. Crome

- Pilot
- Hockey player
- Likes long walks on the beach

Does any of this make me eminently qualified to talk to you about Perl today?

NO

- Software developer for nearly 30 years, currently @ Best Practical developing RT
- Using Perl since 2004, full time since 2015
- Have built, managed, maintained apps for banking, healthcare, ERP, ticketing, and more.
- Former Perl and Raku Foundation Grants Committee Secretary
- Organizer of Charlotte Perl Mongers
- Dancer2 Core Developer
- Former startup founder that specializes in esoteric and “dead” technologies (Perl, Powerbuilder, BSDs)

What is Perl?

Camel (Dromedary)¹



¹ Photo by [Hongbin](#) on [Unsplash](#)

The Onion²



² Photo by K8 on Unsplash





Jason A. Crome (CromeDome)
Carolina Codes, August 24, 2024

User-Friendly³



³ Taken from <https://www.bleepingcomputer.com/news/security/angry-it-admin-wipes-employer-s-databases-gets-7-years-in-prison/>

"Most of you are familiar with the virtues of a programmer. There are three, of course: laziness, impatience, and hubris."

– Larry Wall



A brief history of Perl

- Dynamic language created in 1987 by Larry Wall
- Linguist turned programmer
- Wrote a little utility named patch
- Wrote Perl to address shortcomings in sed, awk, and bash
- Number of releases leading up to Perl 5 in 1994
- Then things go crazy - dot-com boom, etc. etc.
- **There is no internet as we know it without Perl 5**

Why you should use Perl

- Expressive
- Terse, yet still readable
- Fast
- First-class OO programming
- Excellent testing culture
- Robust and mature toolchain
- CPAN
- TIMTOWTDI

Things Perl is good at

- Data munging
- Text processing
- Devops
- Web app development
- System utilities
- Glue
- Online games
- Database applications
- GUI applications
- Anything requiring stability
- Whipupitude
- ...just to name a few!

Things Perl is not good at

- Device drivers
- 3D game engines
- Threaded applications
- Any time you don't have time to handle the learning curve

Common Misconceptions

- Perl is dead
- Perl is dying
- It's just for sysadmins
- It's just for crusty old greybeards
- It's not for real development
- It's not modern and hasn't been updated in years

Actively developed

- 24th release of Perl 5
- One major production release every year
- Will there be a Perl 7? 8? Maybe!



Charlie Stross

@cstross@wandering.shop

Wow. Despite the received wisdom that "Perl is dying", going by the package release frequency it seems to have peaked more than a decade later than I'd have expected. (Also: not dead yet.)

fosstodon.org/@metacpan/112972...

Aug 17, 2024, 05:36 · 0 · 20 · ★ 3



...

Stability - Perl's most important feature

- Nobody talks about stability because stability is boring
- ...unless you an enterprise that can't tolerate apps breaking when OS/packages update
- Much code written in the 1990s can still run on current versions without modification
- ...even though the Perl of 2024 doesn't quite resemble the Perl of 1995

But how?

Feature guards!

```
use v5.40;  
use feature 'signatures';  
use experimental 'class';
```

Companies that use Perl because stability is boring

- Fastly
- Fastmail
- IEEE
- Craigslist
- Venmo
- Roblox
- Zappos
- Estee Lauder
- Shutterstock
- ZipRecruiter
- BBC
- Booking.com
- Bloomberg
- New York Times
- DuckDuckGo
- Wells Fargo
- Request Tracker/BPS
- BofA
- Vimeo
- cPanel
- Reuters
- Grant Street Group
- NYTimes
- JP Morgan Chase
- Ticketmaster
- Google
- Amazon

Let's build a web app!

Dancer2

- Successor to Dancer
- Lightweight MVC framework created at a time Perl was missing one
- Uses a DSL to describe and construct web applications and APIs
- You don't need to be a Perl expert to create great applications in Dancer2!

Dancer2 (continued)

- Runs anywhere Perl does
- Many deployment options
- ...including Docker/containerization out of the box
- AMAZING friendly, welcoming community

Create a new app

```
dancer2 gen -a Gorge --docker  
cd Gorge  
plackup bin/app.psgi
```

App Basics

```
package Gorge;  
use Dancer2;  
  
get '/' => sub {  
    template 'index' => { 'title' => 'Gorge' };  
};
```

Adding a database

```
plugins:  
  Database:  
    driver: "SQLite"  
    database: "db/grail.db"  
    dbi_params:  
      RaiseError: 1  
      AutoCommit: 1
```

DB in Controller

```
package Gorge;
use Dancer2;
use Dancer2::Plugin::Database;

get '/' => sub {
    my $sth = database->prepare( qq{ SELECT * FROM questions WHERE ... });
    $sth->execute;
    my $question = $sth->fetchrow_hashref;
    my @answers = database->quick_select(
        'answers', { question_id => $question->{ question_id } },
    );
    template 'index', { gotcha => $question, answers => \@answers };
};


```

Templates

```
template: "template_toolkit"
engines:
  template:
    template_toolkit:
      start_tag: '[%'
      end_tag: '%]'
```

Templates

```
<form action="/quest" method="POST">
  <label for="answer">[% gotcha.question %]</label>
  <select name="answer" id="answer">
    [% FOREACH a IN answers %]
      <option value="[% a.answer_id %]">[% a.answer %]</option>
    [% END %]
  </select>
  <br>

  <button type="submit">Try to cross...</button>
</form>
```

Layout

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="[% settings.charset %]">
    <title>[% settings.appname %]</title>
    <script src="/javascripts/counter.js"></script>
  </head>
  <body>
    [% content %]
  </body>
</html>
```

Form processing, sessions

```
post "/quest" => sub {
    my $name = body_parameters->get( 'name' );
    session name => $name;

    my $a = database->quick_select( 'answers', { answer_id => body_parameters->get( 'answer' ) } );
    session answer => $a->{ answer };
    if( $a->{ is_correct } ) {
        database->quick_insert( 'crossers', { name => $name } );

        if( $a->{ casts_keeper } ) {
            template 'you-have-to-know-these-things';
        } else {
            forward '/off-you-go', {}, { method => 'GET' };
        }
    } else { ... }
};
```

Demo!

Creating APIs, JSON helpers

```
get "/my-quest" => sub {
    my @response;
    foreach my $param ( qw( name quest answer ) ) {
        push @response, { $param => session->read( $param ) };
    }
    send_as JSON => \@response;
};
```

Closing thoughts

“Perl is worse than Python because people wanted it worse.”

– *Larry Wall*

Thank you!

GitHub: cromedome

CPAN: CROMEDOME

<https://cromedome.blog>

hello@jason.cromedome.dev

