Conference Driven Development
Conference
Driven
Development
Conference Driven Thinking
I blame Leo Lapworth
London.pm
Tech Meet
14th August
Talking about PSGI
Mentioned Matt Wright
Contemplating something along the lines of Matt's PSGI Archive... :-)

1 RETWEET

7:29 PM · 14 Aug 12 · Embed this Tweet

Reply to @davorg
By Dave Cross (davorg) (edit) from London.pm

Lightning talk

Target audience: Any
Language: English
Tags: mattwright nms perl psgi

I don't really know. It's just an idea that came to me a few days ago.

Consider it conference driven development.
Some history
Most popular Perl web site on the web
Most well-known Perl code on the internet
FormMail
Most widely installed Perl program on the internet
Most popular spamming tool on the internet
Terrible example of Perl programming
Popular
!=
Good
People “learn” Perl from copying Matt's code
People learn “Perl” from copying Matt's code.
Not Matt's Scripts
The nms Project
Web programs written by experts
New versions of Matt's scripts
Fewer Bugs
Fewer Security Holes
Higher quality code
Targeting cheap hosting plans
1. No CPAN Modules
2. Perl
5.004_04
(First version to include CGI.pm)
Interesting challenge
“Interesting challenge”
Results not completely satisfactory
No Template Toolkit
No Email:::*
Not best practice
(Even for 2001)
NMS looks embarrassing
Perl web programming has changed a lot
People still find Matt's code
People still “learn” “Perl” from Matt's code
Some people are directed to nms project.
Not best practice
Mojolicious: an unexpected result

By Ovid on October 14, 2012 1:27 PM

While at the Italian Perl Workshop I was talking with a gentleman who does a lot of contract work (and gave me permission to anonymously share this story). Most of his contract work deals with the Web and he's fortunate enough to have worked with quite a few companies who are a bit more sophisticated than the old CGI.pm days. In fact, some of them use Mojolicious, an excellent Web framework that many developers are enjoying. Mojolicious is fast, flexible, robust, and has no CPAN dependencies.

This developer hates working for clients who use Mojolicious. I confess that I was surprised when I found out why. It's an exercise in "unintended consequences".
Why People Don't Like Mojolicious
By Joel Berger on October 14, 2012 2:27 PM under Mojolicious

Ovid posted an interesting article about why one Perler (not himself) dislikes Mojolicious.

I have read several articles, and had several conversations about why people don't like Mojolicious. I have been meaning to write an article about people's dislike for Mojolicious for a while now, so I'm going to take this opportunity to do so while responding to that article.

Mojolicious is Anti-CPAN

Some people don't like it claiming that it is "anti-CPAN" and in fact this comes in two flavors. First, they believe that because a tool is available from CPAN that it should be used rather...
Matt targeted cheap hosting plans.
nms targeted cheap hosting plans
Cheap hosting plans are not CPAN friendly.
Unintended consequences
nms discourages CPAN use
Best Practice vs Easy to install
Cheap hosting plans still exist
But do their users still use Perl/CGI?
PHP
NMS still exists
So I started thinking...
Why not...
...rewrite Matt's scripts again?
Using Modern Perl Best Practices
CPAN
PSGI
Dancer
Catalyst
Mojolicious
Web:::Simple
Template Email::*
DBIx::*Class
Don't expect many users
Simple examples
Solutions to common problems
Entry-level web programming in Perl
Some of this stuff is ridiculously easy
text_clock
rand_text
About five lines of code
Others were overcomplicated by Matt's implementation.
Both far easier if you use a database
Progress report
Github
Four (simple) programs written
Others to follow soon
Hopefully by YAPC::Europe
Conference Driven Development
Show us some code
Remember I said it was really easy
rand_text
my $random_file = 'random.txt';
my $delimiter = "%%\n";

get '/' => sub {
    open my $file, '<', get_file($random_file)
        or error '!
    my @phrases;
    {
        local $/ = $delimiter;
        chomp(@phrases = <$file>);
    }
    my $phrase = @phrases[rand @phrases];
    return $phrase;
};
dance;
rand_image
my $img_dir = 'public/img';

get '/' => sub {
    opendir my $dir, $img_dir or die $!;
    my @imgs = grep { -f "$img_dir/$_" } readdir $dir;
    my $img = @imgs[rand @imgs];

    return redirect "//img/$img";
};

get '/img/:img' => sub {
    return send_file 'img/' . params->{img};
};

dance;
text_clock
use Time::Piece;

get '/' => sub {
    return date();
};

sub date {
    if (! $Display_Format) {
        $Display_Format = build_format();
    }

    return localtime->strftime($Display_Format);
}
sub build_format {
    my @date_fmt;

    push @date_fmt, '%A' if $Display_Week_Day;
    push @date_fmt, '%B' if $Display_Month;
    push @date_fmt, '%d' if $Display_Month_Day;
    push @date_fmt, '%Y' if $Display_Year;
    push @date_fmt, '%H:%M:%S' if $Display_Time;
    push @date_fmt, '%Z' if $Display_Time_Zone;

    return join ' ', @date_fmt;
}
github.com/davorg/matts-psgi-archive
Feel free to submit code
Full replacements are good
Patches to existing programs are good.
Documentation is good
Any help is good
github.com/davorg/matts-psgi-archive
Dave Cross
dave@dave.org.uk
@davorg