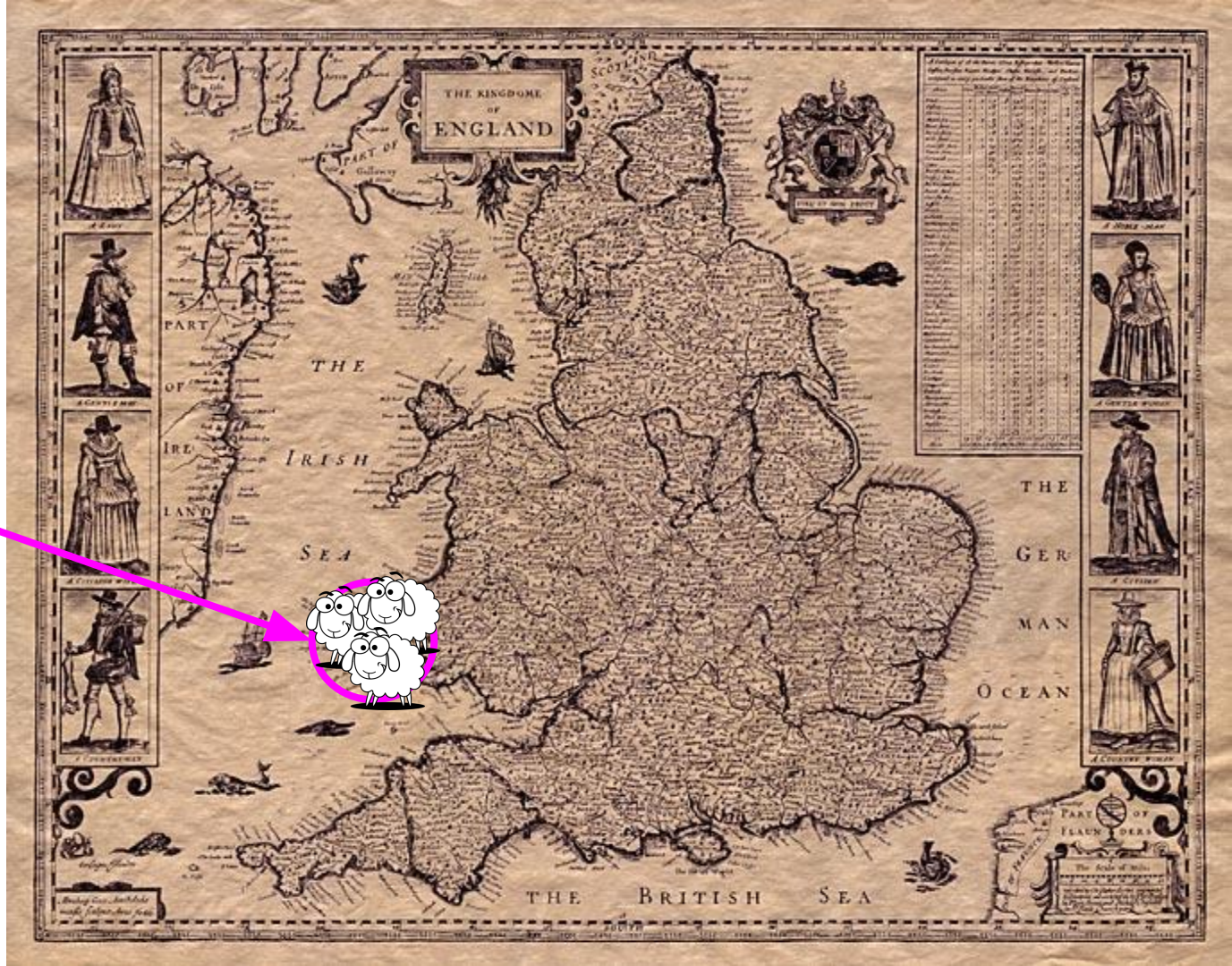

PHP DORSET

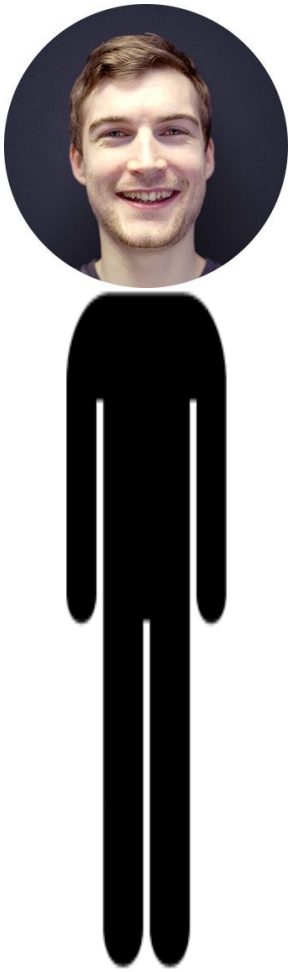
Silex and Twig

Jon Ginn

Silex and Twig

~~Jon Ginn~~


THE KINGDOME OF ENGLAND
PART OF Galloway
PART OF IRELAND
THE IRISH SEA
THE GERMAN OCEAN
THE BRITISH SEA
PART OF FLAUNDERS



Silex and Twig

Alex Ross and Dave Hulbert


BASE

OPEN
DEVICE
LAB

Alex

@rossey

Senior engineer at Base*

Dave

@dave1010

Tech lead at Base*

*we're hiring

wearebase.com

Alex

@rossey

Senior engineer at Base*

Dave

@dave1010

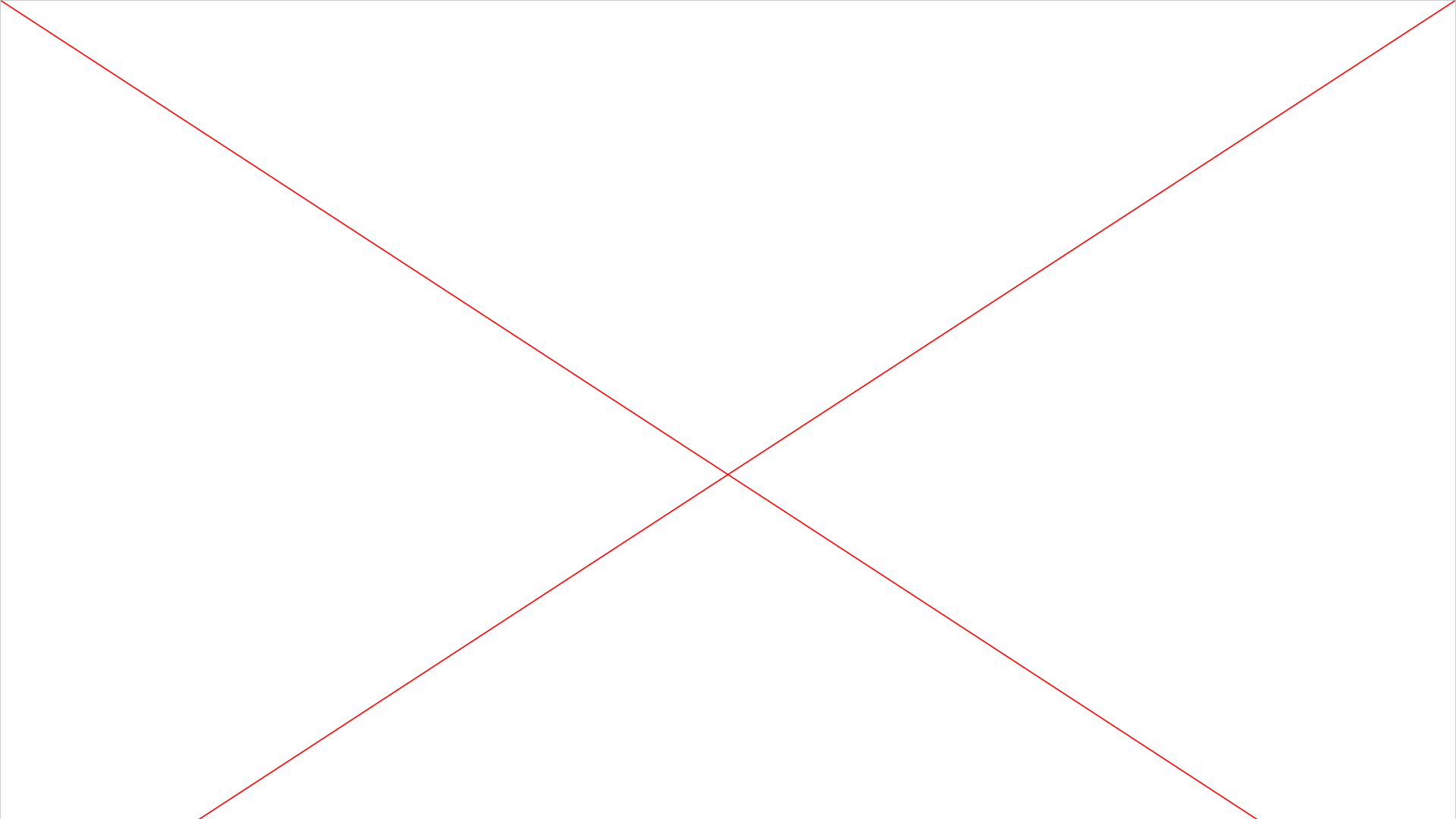
Tech lead at Base*

*we’re hiring

wearebase.com

Silex and Twig

Silex



Micro-framework?

a what?

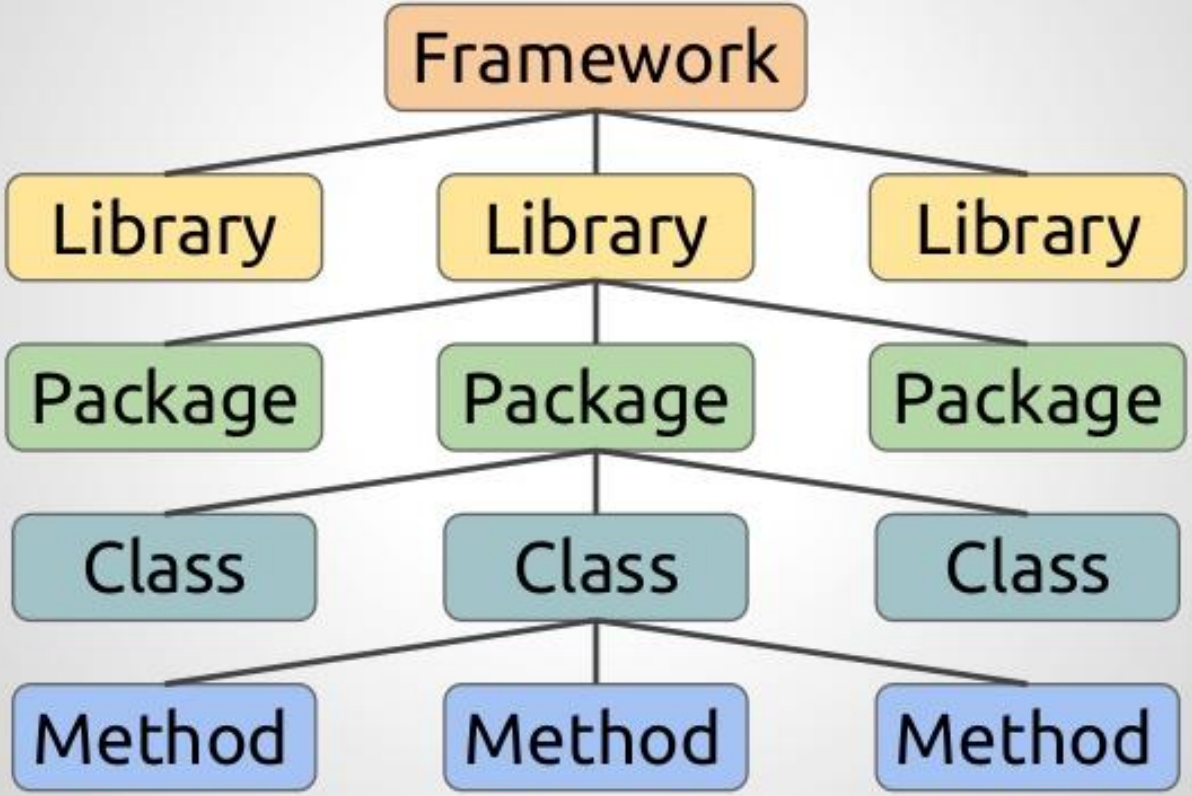
Framework?

...?

“A structure for applications”

(and usually a library of tools too)

APIs


Framework
Library
Library
Library
Package
Package
Package
Class
Class
Class
Method
Method
Method

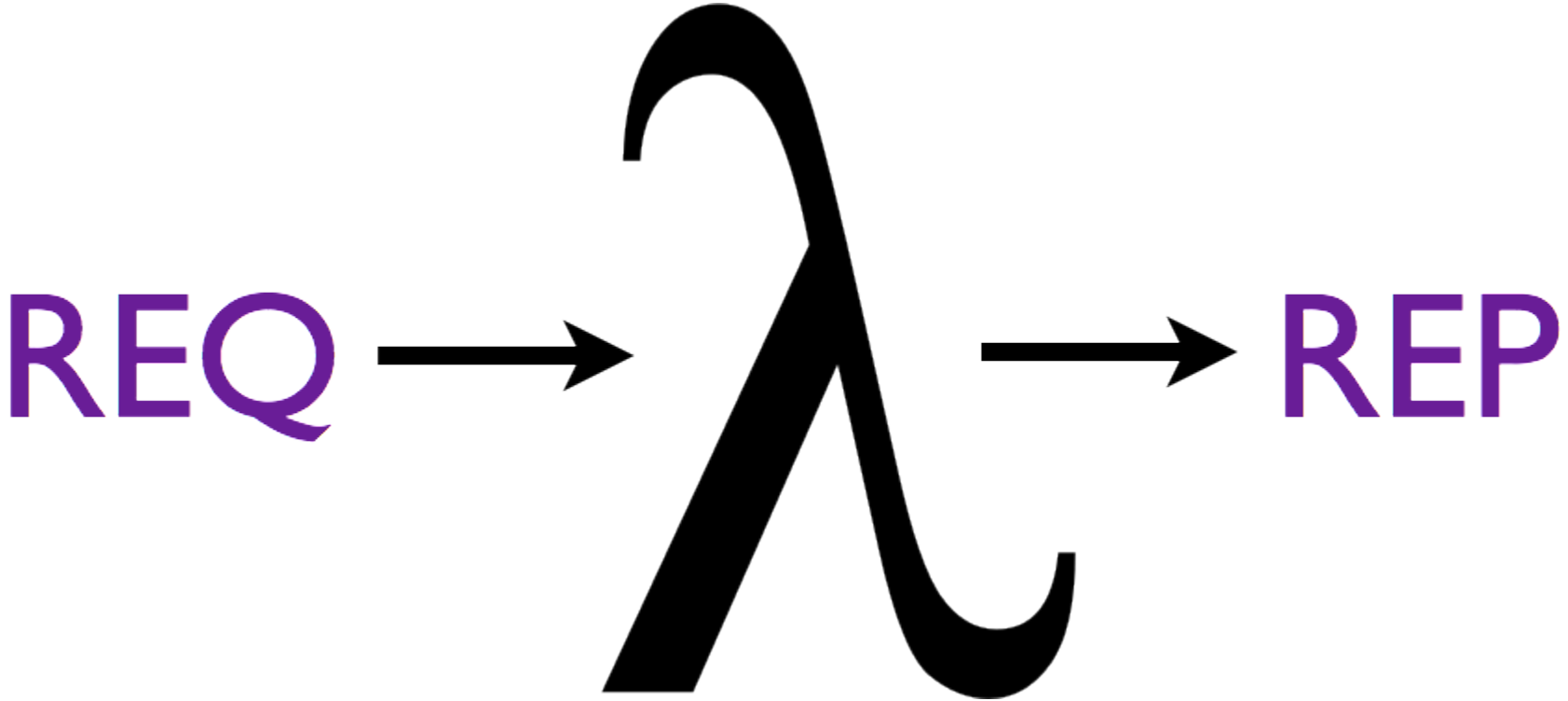
Silex

- a simple web application structure that doesn’t get in your way
- library of tools = Symfony Components

Symfony

Code!

```php
// use the Request
$name = $_GET['name'];

// do some fun stuff
$name = strtoupper($name);

// send back a Response
echo 'Hello ' . htmlspecialchars($name);
```


REQ
REP

REQ

REP

tightly coupling :-(

Tying your code to the input and output:

- Input, global states (`Request`)
  - `$_SERVER`, `$_GET`, `$_POST`, `$_REQUEST`, `$_FILES`, `$_COOKIE`, `$_ENV`
- Output (`Response`)
  - `echo*`, `print`, `printf`, `die`, `exit`

*we could test this with `ob_start`, `ob_get_clean`


quickmeme.com

What if we wanted to make it

- a JSON API?
- send the response as an email or SMS?
- a commandline program?
- an enterprise SOAP Service?
- communicate via Morse code / Enigma machine?


COPY AND PASTE
ALL THE THINGS!
DIYLOL.COM

Dirty code (STUPID)

- S - Singletons and Globals
- T - Tight Coupling
- U - Untestable code
- P - Premature Optimisation
- I - In-descriptive Naming
- D - Duplication
- D - Duplication

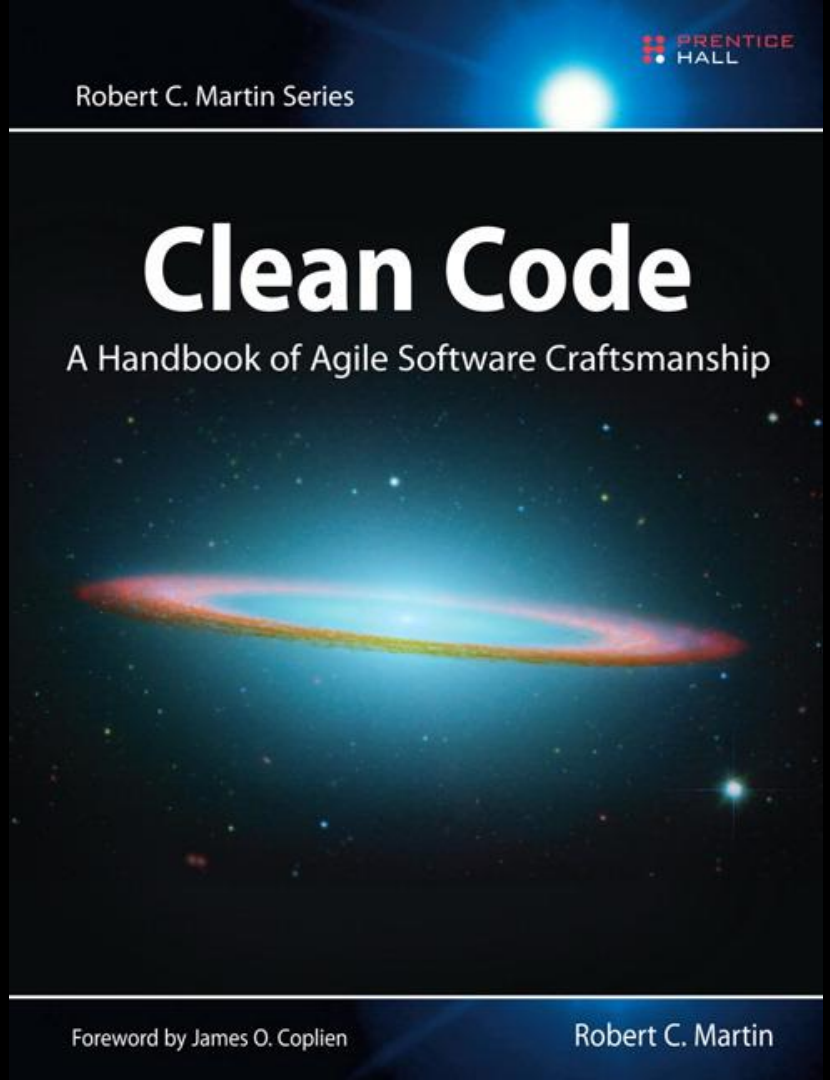
Testability

(PHPUnit)

```php
require_once __DIR__.'/../vendor/autoload.php';

$app = new Silex\Application();

$app->get('/hello/{name}', function($name) use($app) {
    return 'Hello ' . $app->escape($name);
});

$app->run();
```

```ruby
#!/usr/bin/env ruby
require 'sinatra'

get '/hello/:name' do
    "Hello #{params[:name]}!"
end
```

```
var express = require('express');
var app = express();

app.get('/hello.txt', function(req, res) {
    res.send('hello world');
});
```


PRENTICE HALL
Robert C. Martin Series
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Robert C. Martin

Clean code

- Modular
- Reusable
- Easy to extend or change
- Easy to read and understand
- Easy to refactor (maintainable)
- Easy to test

SOLID

Software Development is not a Jenga game

SOLID

- S - SRP
- O - OCP
- L - LSP
- I - ISP
- D - DIP

SOLID

- S - Single Responsibility Principle
- O - Open / Closed Principle
- L - Liskov Substitution Principle
- I - Interface Segregation Principle
- D - Dependency Inversion Principle

3

3-ish

1.2

10

3

Single Responsibility Principle

One reason to change

```php
<?php

$app = new Silex\Application();
$blogPostRepository = new BlogPostRepository;

$app->get('/blog/', function() use($app, $blogPostRepository) {
    $blogPosts = $blogPostRepository->fetchAll();
    return $app['twig']->render('homepage.twig', $blogPosts);
});

$app->run();
```

Open / Closed Principle

Open for extension
Closed for modification

```php
<?php

$app = new Silex\Application();
$blogPostRepository = new BlogPostRepository;

$app->get('/blog/', function() use($app, $blogPostRepository) {
    $blogPosts = $blogPostRepository->fetchAll();
    return $app['twig']->render('homepage.twig', $blogPosts);
});

$app->run();
```

Liskov Substitution Principle

Extending a class shouldn’t break stuff

```php
<?php

$app = new Silex\Application();
$blogPostRepository = new BlogPostRepository;

$app->get('/blog/', function() use($app, $blogPostRepository) {
    $blogPosts = $blogPostRepository->fetchAll();
    return $app['twig']->render('homepage.twig', $blogPosts);
});

$app->run();
```

```php
<?php

interface BlogPostRepositoryInterface
{
    /** @return array of blog posts */
    public function fetchAll();
}

$blogPostRepository = new CachingBlogPostRepository;
$blogPostRepository = new PagedBlogPostRepository;
$blogPostRepository = new RandomizedBlogPostRepository;
$blogPostRepository = new MockBlogPostRepository;
```

Interface Segregation Principle

Don’t depend on stuff you don’t use

```php
<?php

$postController = function(
        Silex\Application $app,
        BlogPostRepositoryInterface $blogPostRepository
    ) {
    $blogPosts = $blogPostRepository->fetchAll();
    return $app['twig']->render('homepage.twig', $blogPosts);
};

$app['posts.controller'] = $app->share(function() use ($app) {
    return $postController($app, new BlogPostRepository);
});

$app->get('/blog/', 'posts.controller');
```

Dependency Inversion Principle

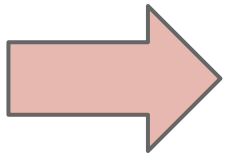
Depend on a concept (abstractions), not an implementation

```php
<?php

$postController = function(
        Silex\Application $app,
        BlogPostRepositoryInterface $blogPostRepository
    ) {
    $blogPosts = $blogPostRepository->fetchAll();
    return $app['twig']->render('homepage.twig', $blogPosts);
};

$app['posts.controller'] = $app->share(function() use ($app) {
    return $postController($app, new BlogPostRepository);
});

$app->get('/blog/', 'posts.controller');
```

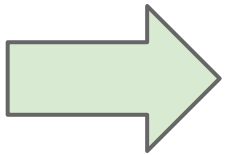
Separation of concerns :-)

GET

`Request`
object



application
business logic



RETURN

`Response`
object

wraps $_SERVER, $_REQUEST, etc nicely

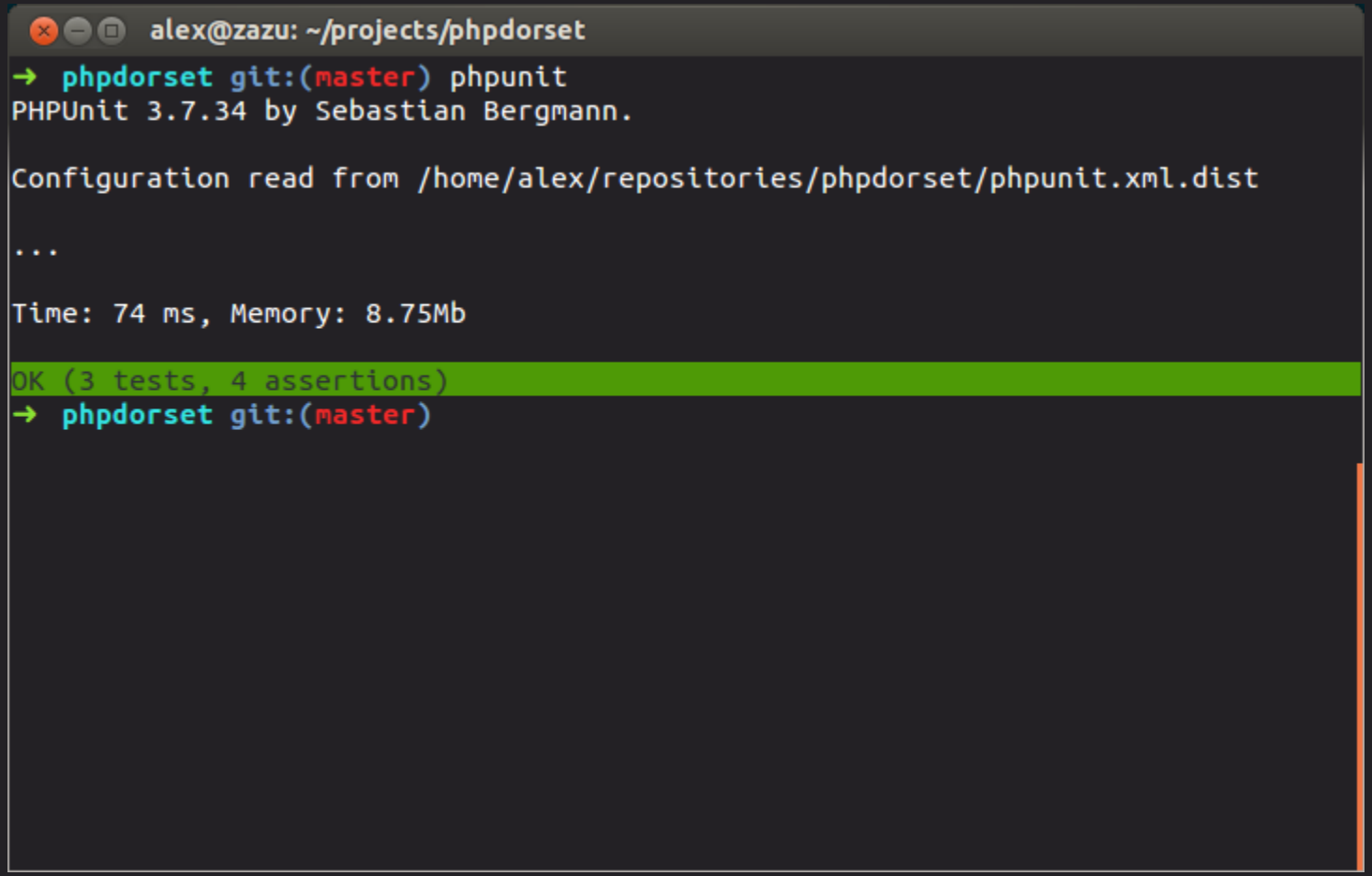
headers and body

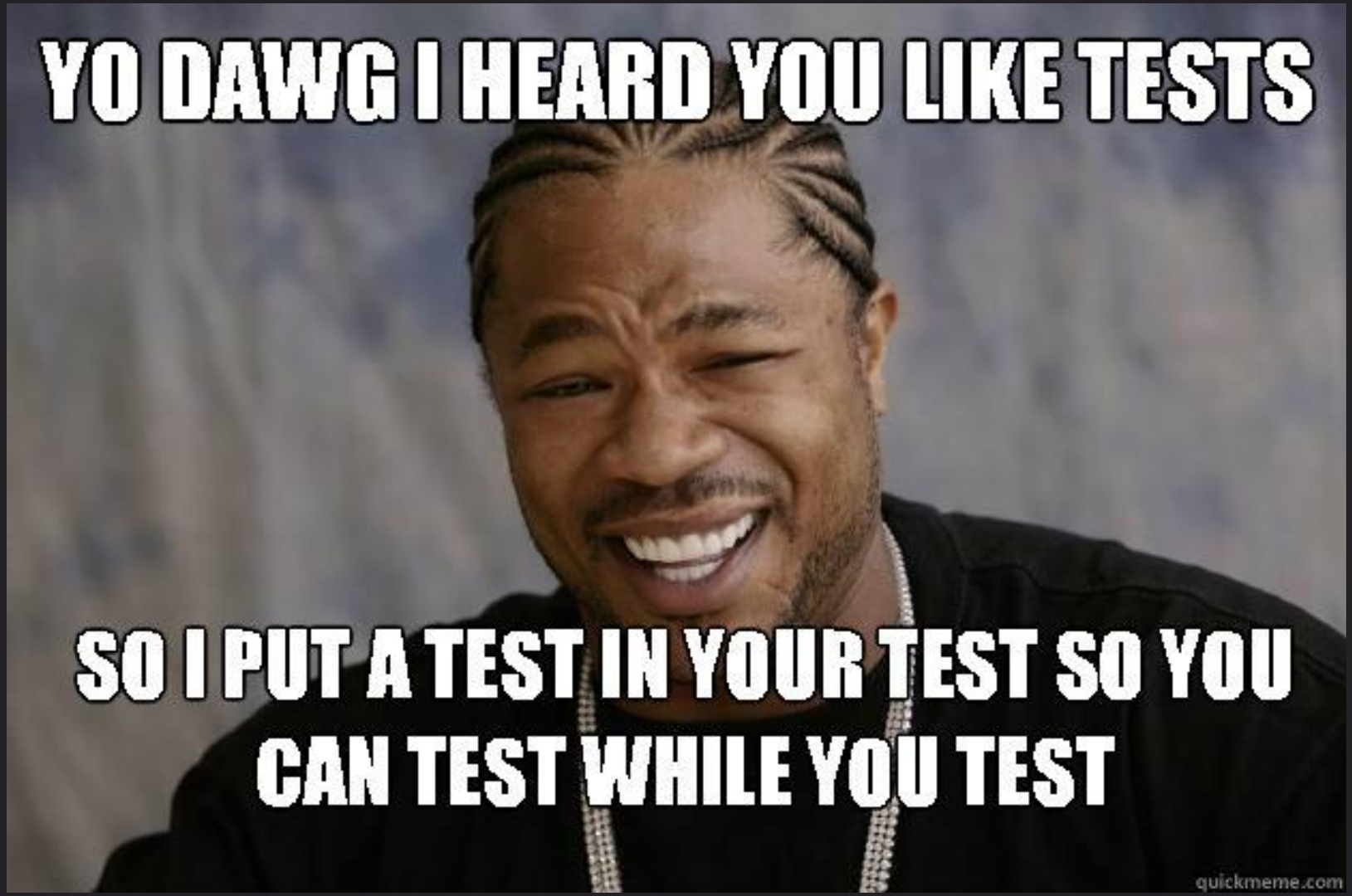
A test

(PHPUnit)

```php
use Silex\WebTestCase;

class ContactFormTest extends WebTestCase
{
    public function testTheContactPage()
    {
        $client = $this->createClient();
        $crawler = $client->request('GET', '/contact');

        $this->assertTrue($client->getResponse()->isOk());
        $this->assertCount(1, $crawler->filter('h1:contains("Contact us")'));
        $this->assertCount(1, $crawler->filter('form'));
        $this->assertCount(1, $crawler->filter('input:contains("Say hi!")'));
    }
}
```

Make the test pass


alex@zazu: ~/projects/phpdorset
➜ phpdorset git:(master) phpunit
PHPUnit 3.7.34 by Sebastian Bergmann.
Configuration read from /home/alex/repositories/phpdorset/phpunit.xml.dist
...
Time: 74 ms, Memory: 8.75Mb
OK (3 tests, 4 assertions)
➜ phpdorset git:(master)


YO DAWG I HEARD YOU LIKE TESTS
SO I PUT A TEST IN YOUR TEST SO YOU CAN TEST WHILE YOU TEST
quickmeme.com

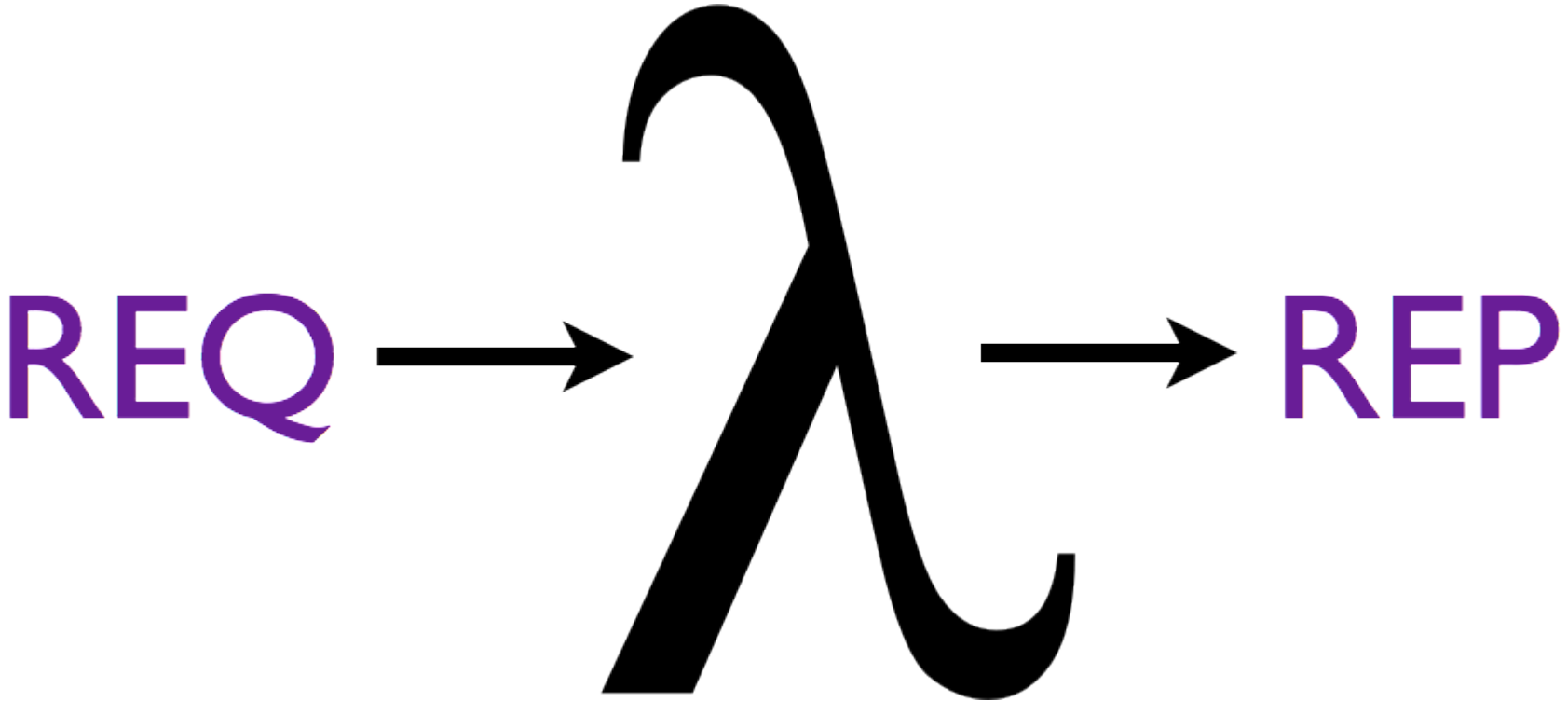
`HttpKernelInterface`

Creates flexible and fast HTTP-based applications

```php
interface HttpKernelInterface
{
    const MASTER_REQUEST = 1;
    const SUB_REQUEST = 2;

    public function handle(
        Request $name,
        $type = self::MASTER_REQUEST,
        $catch = true
    ) {}
}
```

```
/**
 *  Handles a Request to convert it to a Response.
 *
 *  When $catch is true, the implementation must catch all exceptions
 *  and do its best to convert them to a Response instance.
 *
 *  @param Request $request  A Request instance
 *  @param integer $type     The type of the request (MASTER_REQUEST or
SUB_REQUEST)
 *  @param Boolean $catch    Whether to catch exceptions or not
 *  @return Response         A Response instance
 *  @throws \Exception       When an Exception occurs during processing
 */
```

```
/**
 *  Handles a Request to convert it to a Response.
 *
 *  When $catch is true, the implementation must catch all exceptions
 *  and do its best to convert them to a Response instance.
 *
 *  @param Request $request  A Request instance
 *  @param integer $type     The type of the request (MASTER_REQUEST or
SUB_REQUEST)
 *  @param Boolean $catch    Whether to catch exceptions or not
 *  @return Response         A Response instance
 *  @throws \Exception       When an Exception occurs during processing
 */
```



http://stackphp.com

Used by

- Symfony
- Laravel
- Drupal
- phpBB

Symfony

Drupal™

A Silex App *is* Pimple

- Simple container with ~80 lines of code
- ArrayAccess interface
- *IoC container, for dependency injection
- Allows you to loosely couple your classes

*IoC = Inversion of Control

a SENSIOLABS product

SILEX

The PHP micro-framework
based on the Symfony2 Components

DOCUMENTATION DOWNLOAD DEVELOPMENT CONTRIBUTORS

A PHP micro-framework standing on the shoulder of giants

Silex is a PHP microframework for PHP 5.3. It is built on the shoulders of Symfony2 and Pimple and also inspired by sinatra.

A microframework provides the guts for building simple single-file apps. Silex aims to be:

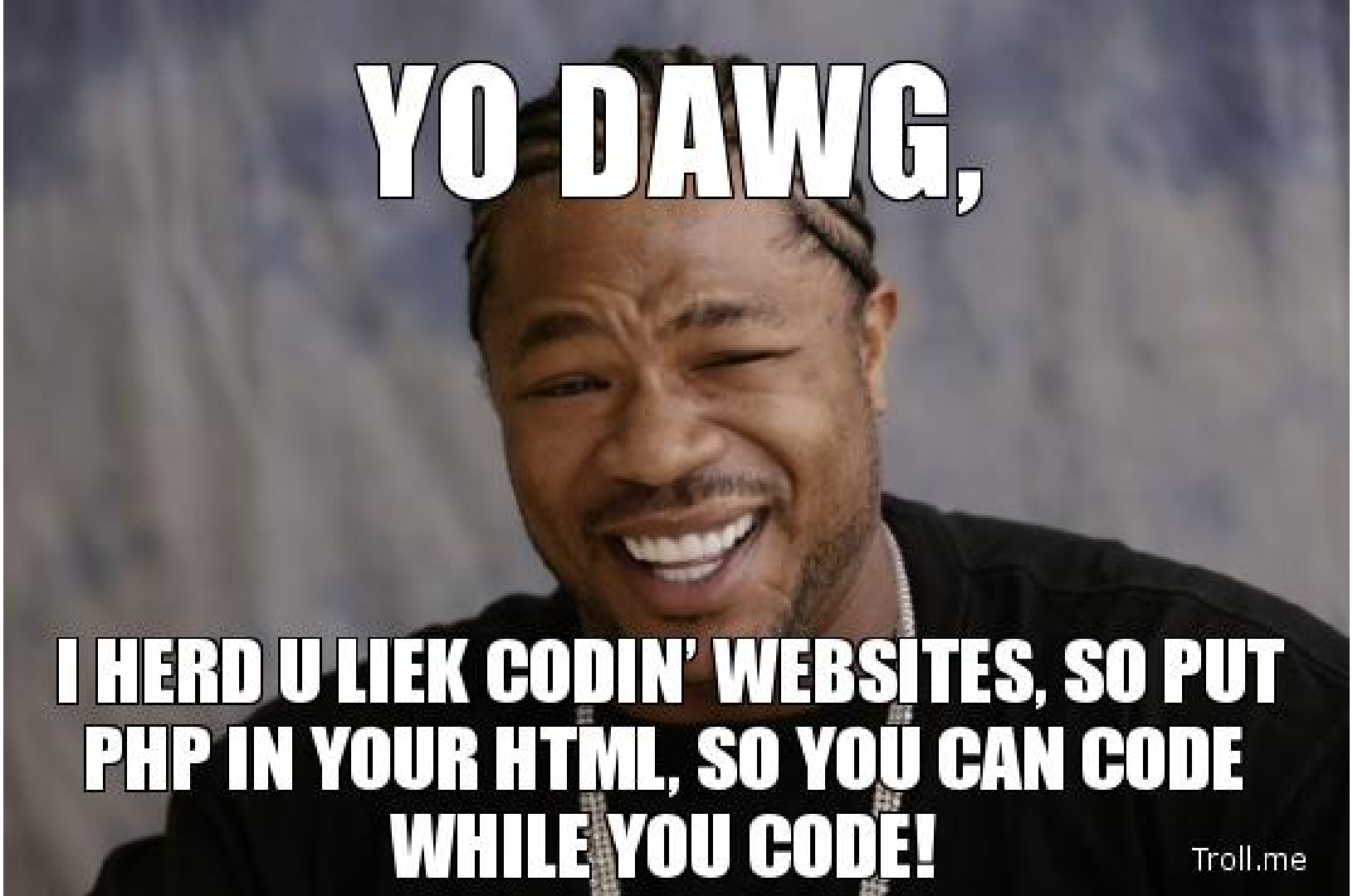
- *Concise*: Silex exposes an intuitive and concise API that is fun to use.
- *Extensible*: Silex has an extension system based around the Pimple micro service-container that makes it even easier to tie in third party libraries.

LEARN MORE →

INSTALL NOW ↓

Twig

Better than mixing PHP & HTML


YO DAWG,
I HERD U LIEK CODIN' WEBSITES, SO PUT PHP IN YOUR HTML, SO YOU CAN CODE WHILE YOU CODE!
Troll.me

Separation

Variables

```
{{ title }}

{{ var|escape }}

{{ foo.bar }}
```

Control blocks

```
{% for user in users %}
    * {{ user.name }}
{% else %}
    No users have been found.
{% endfor %}
```

Filters

```
{{ name|striptags|title|reverse }}
```

Includes

```
{% include 'sidebar.html' %}
```

Inheritance

base.twig

```
<title>{% block title %}My site{% endblock %}</title>
<div>{% block content %}{% endblock %}</div>
```

child.twig

```
{% extends "base.twig" %}
{% block title %}Contact us{% endblock %}
{% block content %}Email{% endblock %}
```

Twig in the wild

Nottingham City Transport

Your frequent City bus network

LOG IN | REGISTER

HOME | PLAN A JOURNEY | TIMETABLES, TICKETS & MAPS | CUSTOMER SERVICES | ABOUT US | MY ACCOUNT

SEARCH FOR STOP, STREET, OR POSTCODE

Enter a location

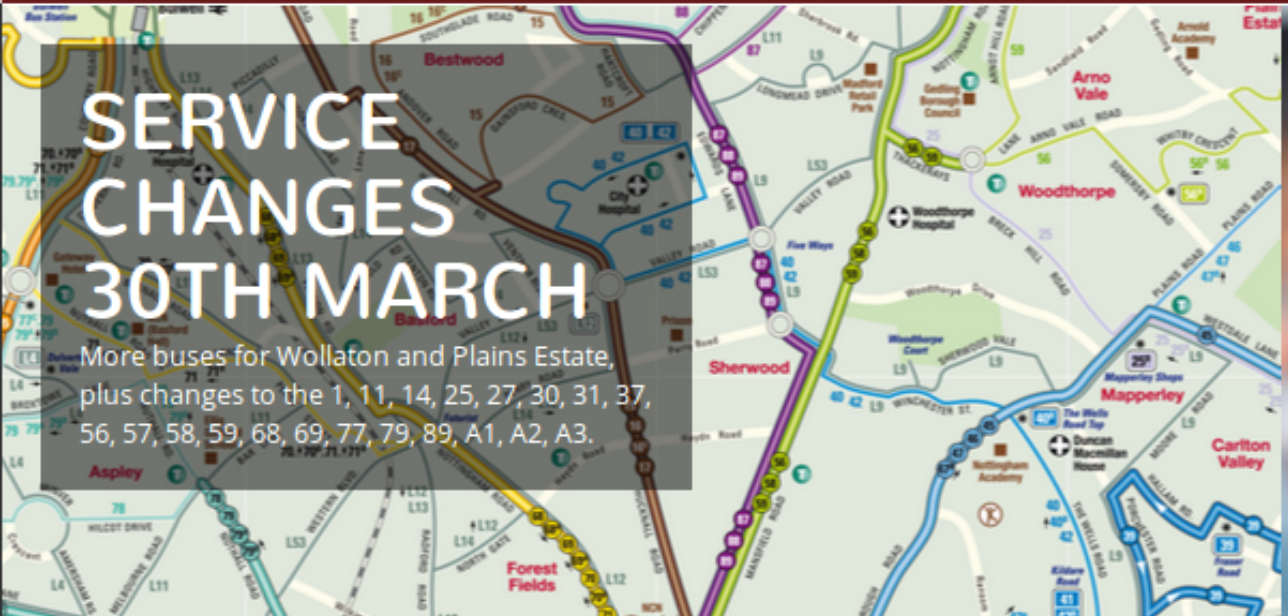
Search

SELECT A SERVICE

Select...

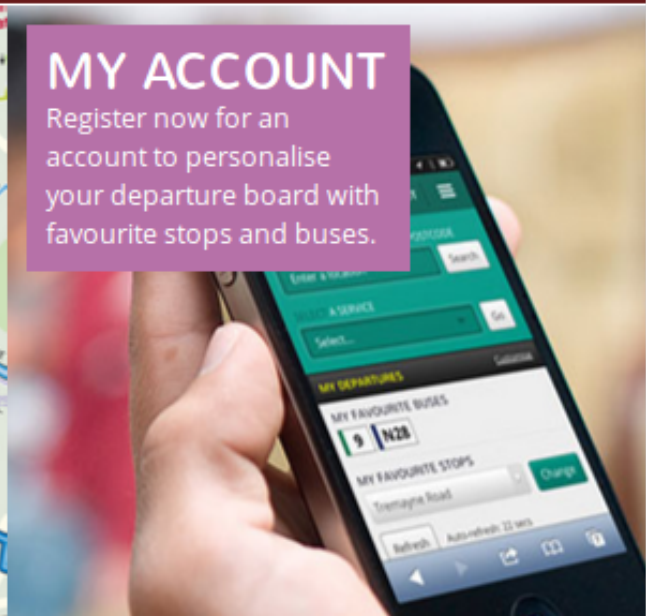
Go

Parliament Street Stop Closure for Duct Works 87 88 89



SERVICE CHANGES 30TH MARCH

More buses for Wollaton and Plains Estate, plus changes to the 1, 11, 14, 25, 27, 30, 31, 37, 56, 57, 58, 59, 68, 69, 77, 79, 89, A1, A2, A3.



MY ACCOUNT

Register now for an account to personalise your departure board with favourite stops and buses.

```
<div class="account-area">
    <div class="col-8 col-md-8 col-lg-8 col-xs-12 col-sm-8">
        <h3>My Favourite Stops</h3>
        <p>Add the stops you regularly use and when you're logged into the site we'll make these easily available on the de

        {% include 'components/messages.twig' %}

        <table class="table table-striped" id="favourite-stops">
            <thead>
            <tr>
                <td>Stop Name</td>
                <td colspan="2">Stop Code</td>
            </tr>
            </thead>
            <tbody>
            {% if favouriteStops|length == 0 %}
                <tr class="no-stops-found">
                    <td colspan="4">
                        No stops found. Why not add one below?
                    </td>
                </tr>
            {% endif %}
            {% for stop in favouriteStops %}
                {% if stop.atco_code != '' %}
                    <tr>
                        <td>
                            <a href="{{ route('journeyplanner') }}/?stop={{ stop.atco_code }}">{{ stop.commonname }}</a>
                        </td>
```

TWIG | The flexible, fast, and secure template engine for PHP

ABOUT DOCUMENTATION BLOG DEVELOPMENT CONTRIBUTORS

Twig for Template Designers ¶

This document describes the syntax and semantics of the template engine and will be most useful as reference to those creating Twig templates.

Synopsis ¶

Table of C

- Twig for
  - Synop
  - IDEs
  - Variab
    - Gl

Silex & Twig

Silex comes with a bridge that provides a Twig service

Register the service and you get twig

```
"require": {
    "twig/twig": ">=1.8,<2.0-dev"
}
```

```
$app->register(new Silex\Provider\TwigServiceProvider(), array
(
    'twig.path' => __DIR__ . '/views',
));

$app->get('/hello/{name}', function ($name) use ($app) {
    return $app['twig']->render('hello.twig', array(
        'name' => $name,
    ));
});
```

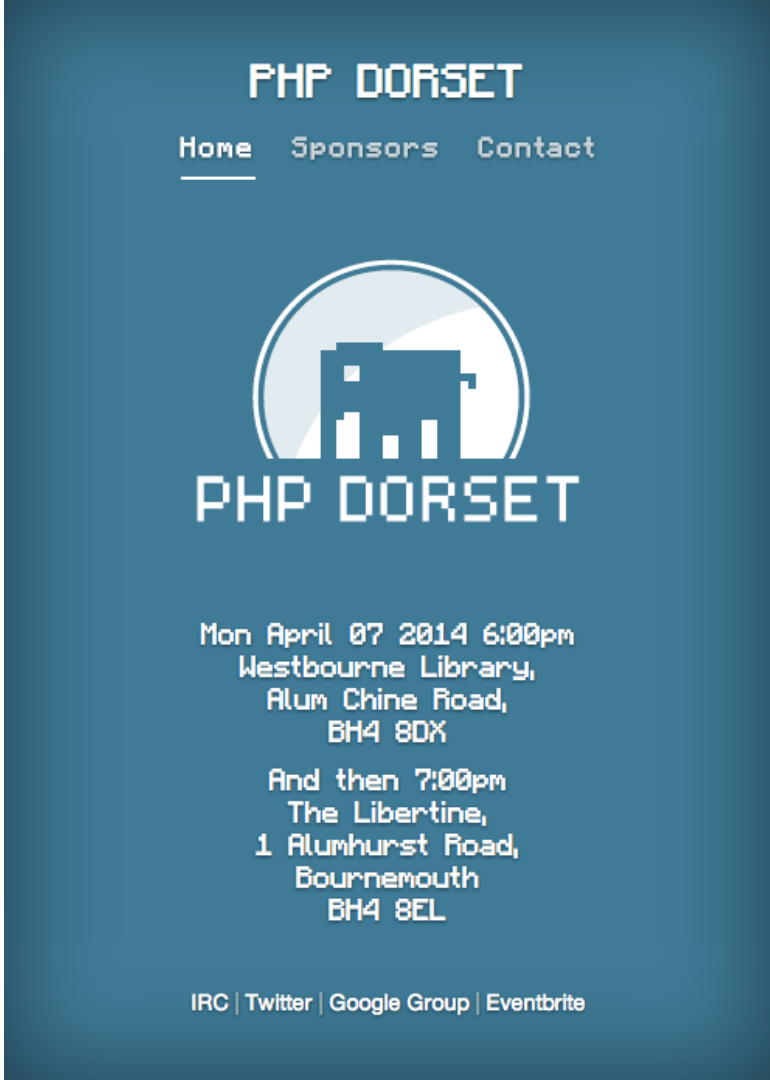
Silex in the wild

WordPress Example

```php
$app->post('/api/v1/user/personal-details', function (Request $request) use ($app) {
    $wordPressUser = get_user_by('email', $request->get('email'));

    if (!$wordPressUser instanceof WP_User) {
        return $app->json(false);
    }

    $userRepository = $app['user_repository'];
    $user = $userRepository->fetchByWordPressId($wordPressUser->ID);
    $translator = new PersonalDetailsTransformer;
    $personalDetails = $translator->translateToPersonalDetails($user);

    return $app->json($personalDetails);
});
```

PHP Dorset site

- silex
- twig



Any Questions & Thank You

Feedback: goo.gl/TnkeCT
Sildes: goo.gl/ilu1rl

Alex (@rossey) & Dave (@dave1010)
wearebase.com/hiring

Tweet Jon Ginn (@jonginn) and tell him what he missed!

PHP DORSET