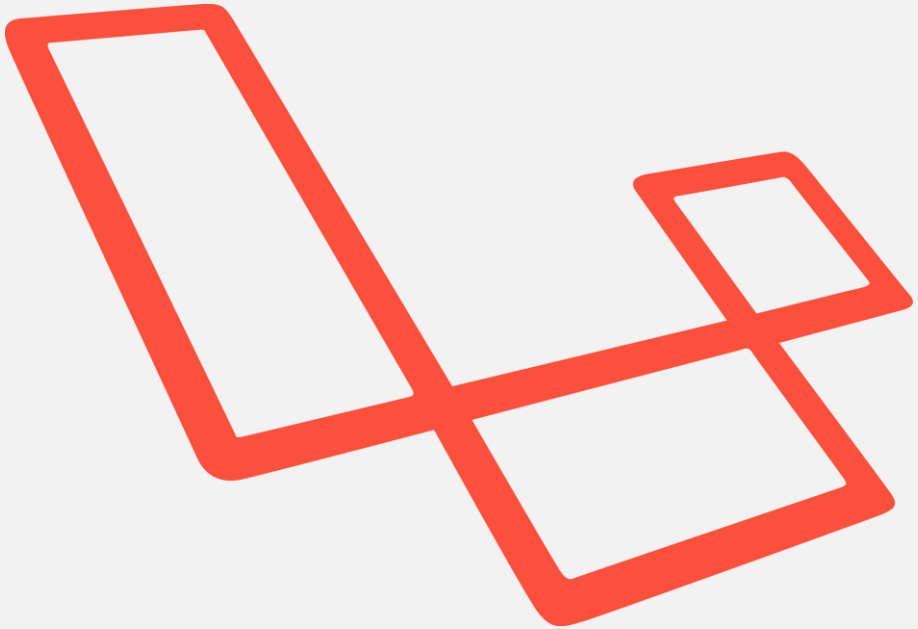
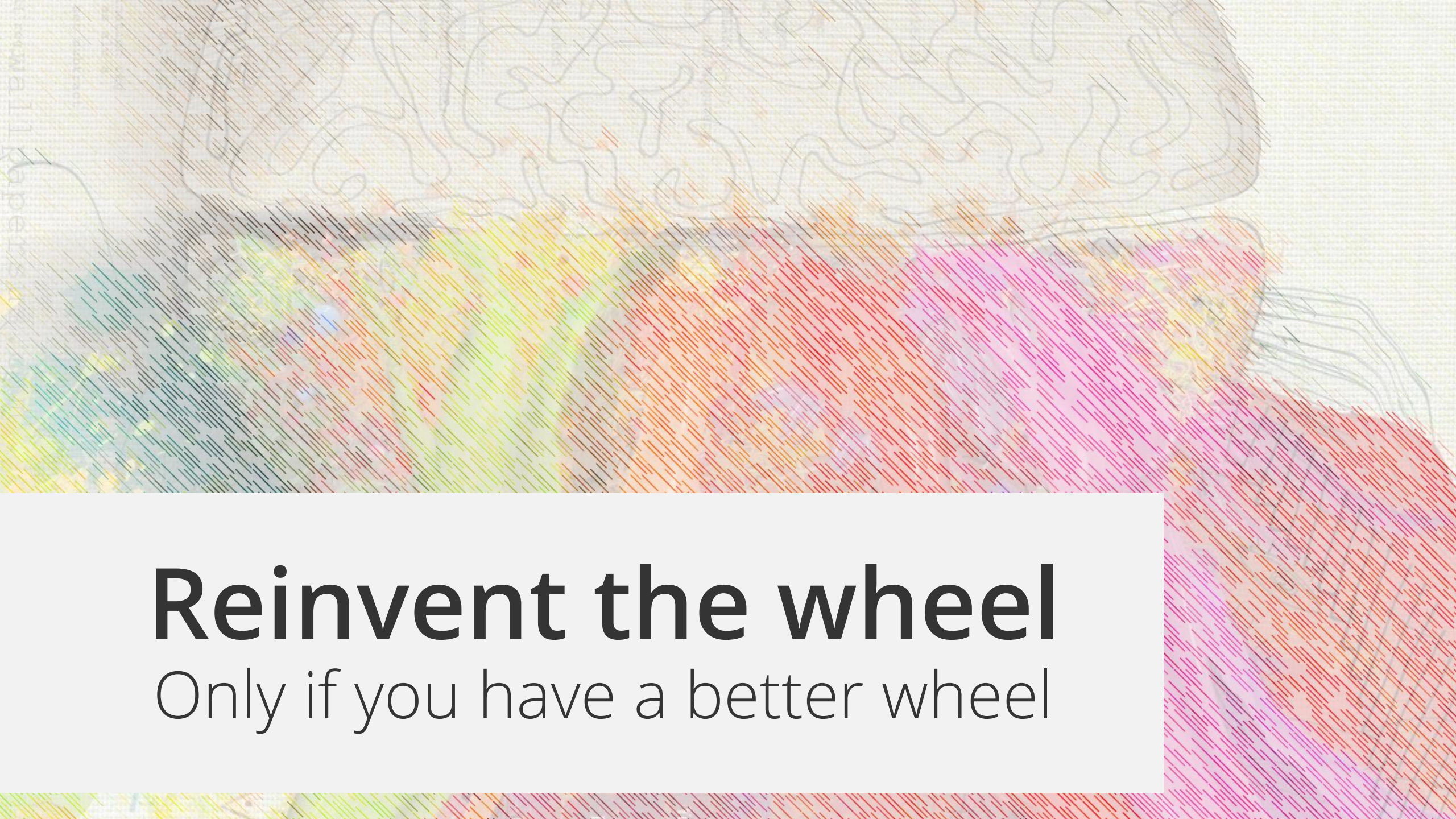


Laravel

You have arrived.

Francisco Neves

@github: francisconeves
@website: www.francisconeves.com
@contact: contact@francisconeves.com


Reinvent the wheel
Only if you have a better wheel

About Laravel

Let me tell you how awesome it is

Laravel is

A recent PHP framework

Laravel is

Totally open-source

Laravel is

Very lightweigth

Laravel is

Easy but useful

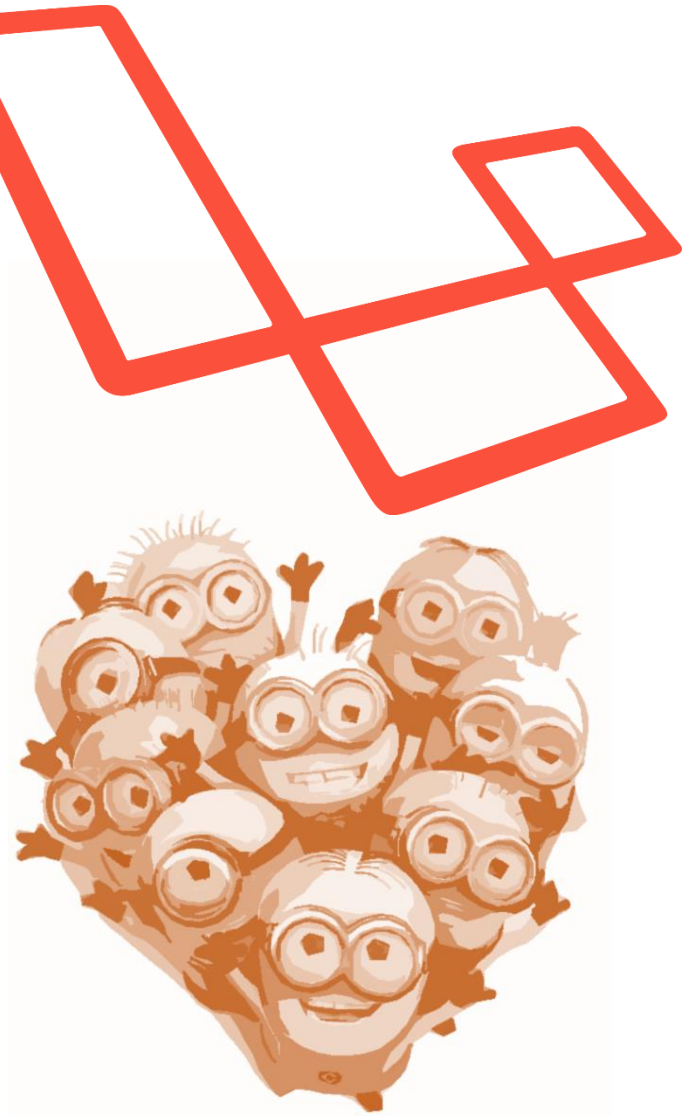
Laravel is

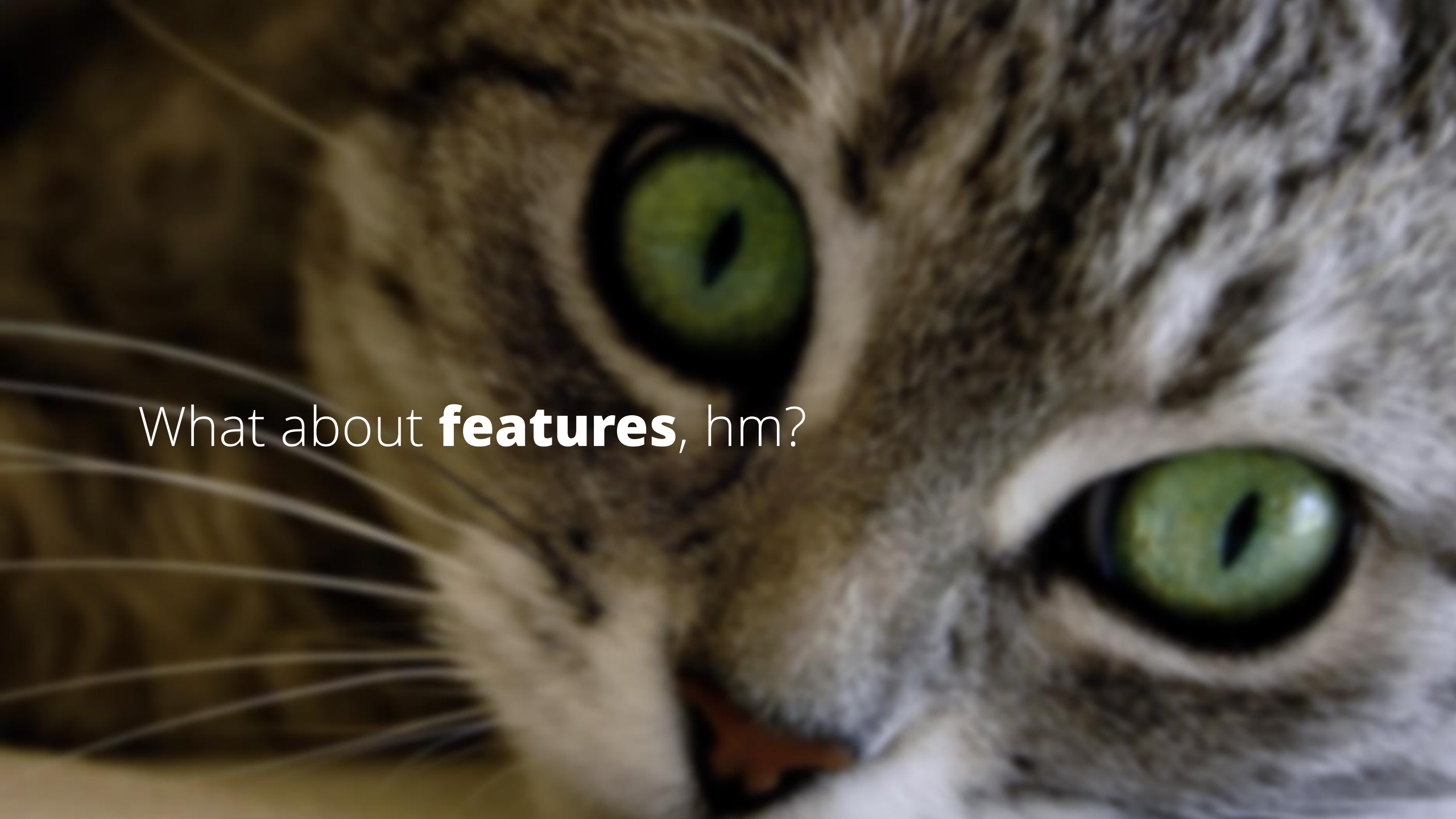
Inspiration

Laravel is

Laravel and just Laravel

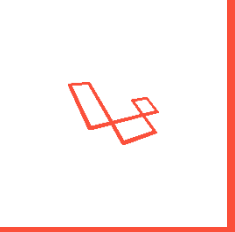
You will **looooove** it!




What about **features**, hm?

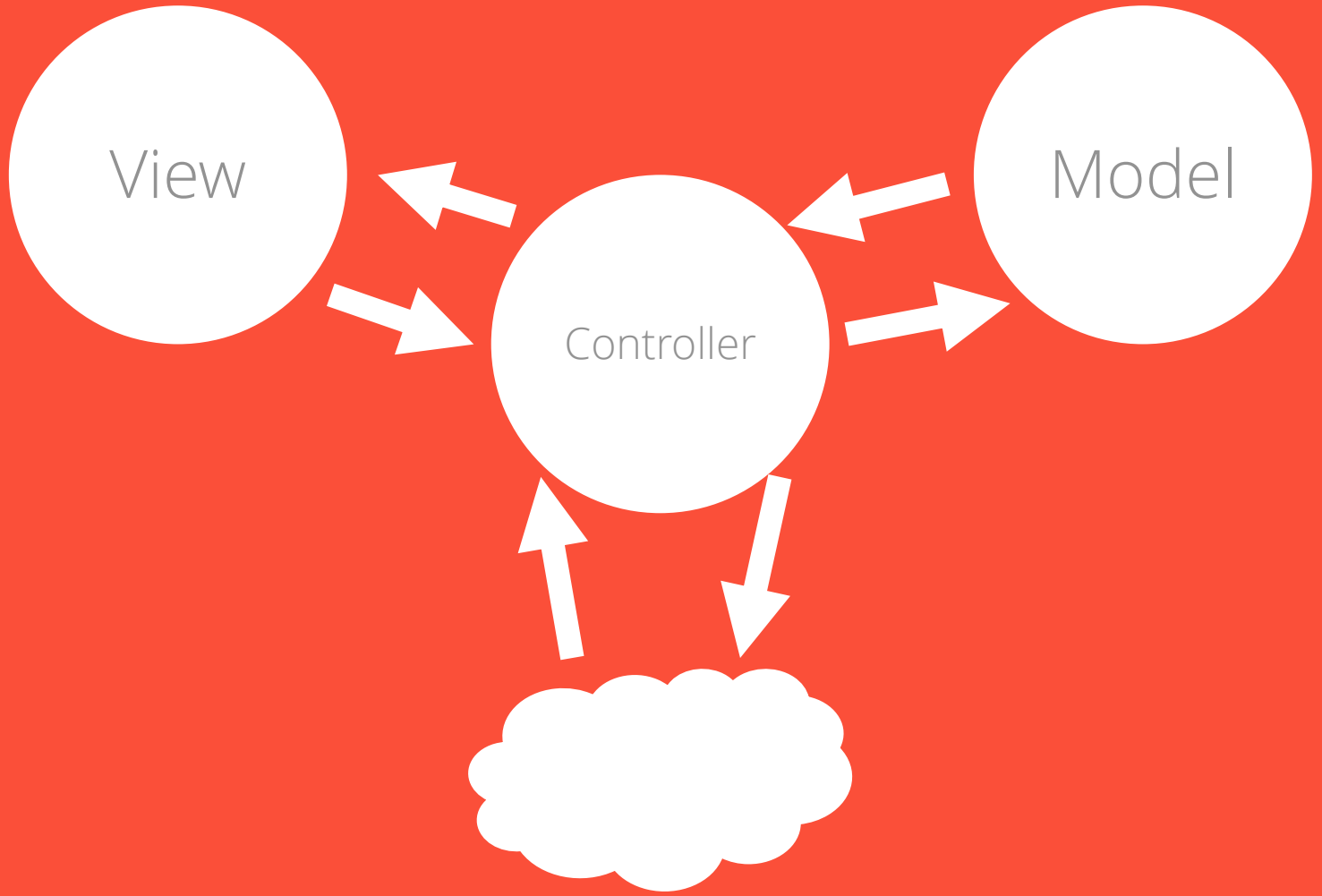
MVC Pattern

Great and modular organization



MVC Pattern

Great and modular organization



Easy Routing

Gives awesome, user-friendly and beautiful links

Easy Routing

Gives awesome, user-friendly and beautiful links

http://website.com/public/**tasklist**/**show**/12

Controller Action ID

Authentication

Yes, you don’t need to worry about that

Blade Templating

Very simple and useful templating

Migrations

Let us save database's changes

Eloquent ORM

Object-Relational-Mapping without stress

Artisan CLI Tool

Helpful developer commands

Artisan CLI Tool

Helpful developer commands

```
$ php artisan controller:make TasklistController
```

The awesome thing? We can extend it!

Bundles

Easy to add new features

Bundles

Easy to add new features

```
$ php artisan bundle:install laraveless
```

http://bundles.laravel.com

Helpers

Improve workflow with Laravel helpers

Helpers

Improve workflow with Laravel helpers

```php
<?php
echo link_to('foo/bar', $title, $attributes = array(), $secure = null);
```

WOW, THIS IS AWESOME!

And how can I work with that awesomeness?

Installing Laravel

Let’s get this rolling

Laravel needs

PHP 5.3.7 or later
MCrypt PHP extension

Get Composer

```
$ curl -sS https://getcomposer.org/installer | php
$ mv composer.phar /usr/local/bin/composer
```

Composer is a PHP dependency manager

Get Laravel

$ composer create-project laravel/laravel [path]

Start Development Server

$ php artisan serve –-port=[port]

Go to

http://localhost[:port]/

You have arrived.

Routing

We have a lot of possibilities

app/routes.php

Closures Routing

Route to function (closure)

Closures Routing

Route to function (closure)

```php
<?php
// http://website.com/tasklist
Route::get('tasklist', function(){
    // Get all Tasklists
    $tasklists = Tasklist::all();

    // Create view for Tasklists
    return View::make('tasklist.index')->with('tasklists', $tasklists)
});
```

Controller Actions

Route to controller's action

Controller Action

Route to controller's action

```php
<?php
// GET Request
Route::get('tasklist', 'TasklistController@index');
Route::get('tasklist/show/{id}', 'TasklistController@show');
// POST Request
Route::post('tasklist', 'TasklistController@create');


```

Routing

We have a lot of possibilities

RESTful Controller

Rest a little, controller has the force

RESTful Controller

Rest a little, controller has the force

```php
<?php
// app/routes.php
Route::controller('tasklist', 'TasklistController');

// app/controllers/TasklistController.php
class TasklistController extends BaseController
{
    // http://website.com/tasklist/list/1
    function getList($id)
    {
        ...
    }
}
```

Routing

We have a lot of possibilities

Resource Controller

Controller has what you need

```
$ php artisan controller:make TasklistController
```

Resource Controller

Controller has what you need

Verb	Path	Action	Route Name
GET	/resource	index	resource.index
GET	/resource/create	create	resource.create
POST	/resource	store	resource.store
GET	/resource/{resource}	show	resource.show
GET	/resource/{resource}/edit	edit	resource.edit
PUT/PATCH	/resource/{resource}	update	resource.update
DELETE	/resource/{resource}	destroy	resource.destroy

Resource Controller

Controller has what you need

```php
<?php
// Resource Controller
Route::resource('tasklist', 'TasklistController');


```

Route Groups

Do the same thing to many routes

Route Groups

Do the same thing to many routes

```php
<?php
// Route Groups
Route::group(array('prefix' => 'admin'), function(){
    Route::get('tasklist', function(){
        ...
    });
});

```

http://website.com/admin/tasklist

Routing

We have a lot of possibilities

Route Filters

"Let them get in, if..."

Route Filters

"Let them get in, if..."

```php
<?php
// Route Filters
Route::get('tasklist', array('before' => 'auth', function(){
        // Do it only if user is authenticated
}));
```

Filters file: app/filters.php

Authentication

Validate credentials and users

Authentication

Validate credentials and users

```php
<?php
if (Auth::attempt(array('email' => $email, 'password' => $password)))
{
    // User is logged
}
```

Blade Templating

Very simple and useful templating

app/views/.../filename.blade.php

Blade Templating

Very simple and useful templating

```
<?php
@foreach($tasks as $task)
    @if($task->done)
        <p class="done">{{$task->name}}</p>
    @else
        <p>{{$task->name}}</p>
    @endif
@endforeach
```

Migrations

Let us save database's changes

```
$ php artisan migrate:make create_users_table
```

Migrations

Let us save database's changes

```php
<?php
class CreateUsersTable extends Migration {

    public function up()
    {
        Schema::create( 'users', function( $table ){
            ...
        });
    }

    public function down()
    {
        Schema::drop( 'users' );
    }

}
```

Migrations

Let us save database's changes

Schema Builder

Change database with minimal code

Schema Builder

Change database with minimal code

```php
<?php
class CreateUsersTable extends Migration
{
    public function up()
    {
        Schema::create( 'users', function( $table ){
            $table->increments('id');
            $table->string('name');
            $table->string('password');
            $table->string('email')->unique();
            $table->timestamps();
        });
    }
    ...
}
```

Migrations

Let us save database's changes

Migrate

$ php artisan migrate

Rollback last migration

$ php artisan migrate:rollback

Fresh install

$ php artisan migrate:refresh

Query Builder

Construct queries to database

Query Builder

Construct queries to database

```php
<?php
// Get all
$users = DB::table('users')->get();

// Get one
$user = DB::table('users')->where('name', 'Francisco Neves')
                          ->first();

// Get records (name => id)
$users = DB::table('users')->lists('name', 'id');

```

Eloquent ORM

Object-Relational-Mapping without stress

Eloquent ORM

Object-Relational-Mapping without stress

```php
<?php
class TasklistController extends \BaseController {

    public function index()
    {
        $tasklists = Tasklist::all();

        return View::make('tasklist.index')->with('tasklists', $tasklists);
    }
}
```

Query Scopes

Create scopes for what you use the most

Query Scopes

Create scopes for what you use the most

```php
<?php
class Task extends Eloquent
{
    public function scopeTodo($query)
    {
        return $query->where('todo', '=', false);
    }
}

$tasks = Task::todo()->get();

```

Eloquent Relationships

From reality to models

Eloquent Relationships – One to One

From reality to models

```php
<?php
class User extends Eloquent
{
    public function tasklist()
    {
        return $this->hasOne('Tasklist');
    }
}

// Get the user's tasklist
$tasklist = User::find(1)->tasklist->name;
```

Eloquent Relationships – One to Many

From reality to models

```php
<?php
class User extends Eloquent
{
    public function tasklists()
    {
        return $this->hasMany('Tasklist');
    }
}

// Get the user's tasklist
$tasklists = User::find(1)->tasklists;
foreach( $tasklists as $tasklist )
{
    ...
}
```

Eloquent Relationships – Inverse Relation

From reality to models

```php
<?php
class Tasklist extends Eloquent
{
    public function user()
    {
        return $this->belongsTo('User');
    }
}

$user = Tasklist::find(1)->user;
```

Eloquent Relationships – Many to Many

From reality to models

```php
<?php
class Tasklist extends Eloquent
{
    public function users()
    {
        return $this->belongsToMany('User');
    }
}

$users = Tasklist::find(1)->users;
foreach(...){...}

```

Caching, Events, Pagination, Mailer

And more, more & more...

http://four.laravel.com/docs
http://laravel.io

Marry me, Laravel?

Thank you, guys!

Francisco Neves

@github: francisconeves
@website: www.francisconeves.com
@contact: contact@francisconeves.com