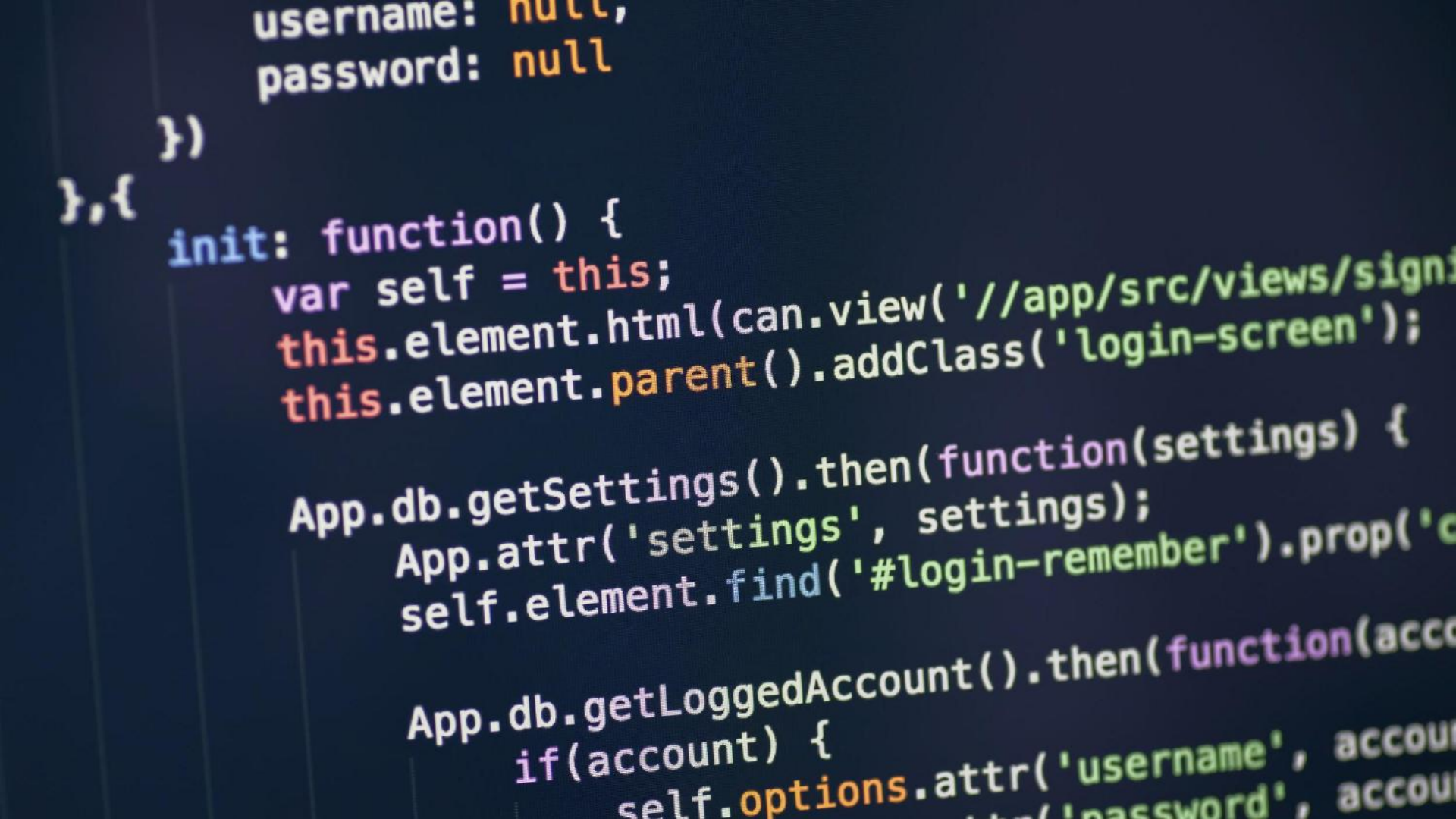
Janos Pasztor

Clean Code in Small Companies

www.pasztor.at


Stock photo, not actual developer

Stock photo, not actual developer


username: null,
password: null
})
},{
init: function() {
var self = this;
this.element.html(can.view('//app/src/views/signi
this.element.parent().addClass('login-screen');
App.db.getSettings().then(function(settings) {
App.attr('settings', settings);
self.element.find('#login-remember').prop('c
App.db.getLoggedAccount().then(function(acco
if(account) {
self.options.attr('username', accou


Stock photo, not actual developer

Robert C. Martin

“Uncle Bob”


ENTRECLOUD

1. Reading code is hard

1. Reading code is hard
2. We all have to read code

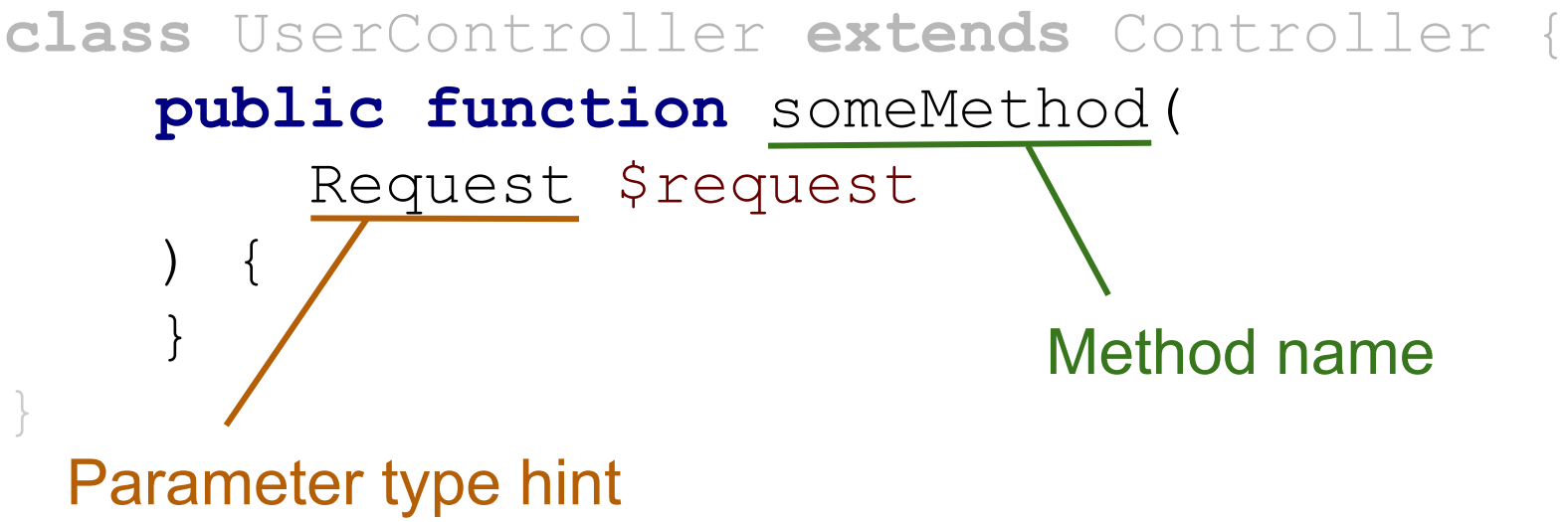
1. Reading code is hard
2. We all have to read code
3. Surprises are bad

Code in these Slides

```
$iAmAVariable
```


class UserController extends Controller {
}
Class name
Parent class name

```
class UserController extends Controller {
    private $memberVariable;
}
```


class UserController extends Controller {
public function someMethod(
Request $request
) {
}
}
Parameter type hint
Method name

https://pasztor.at

A few words on testing...

More on this later

```php
class UserController {
    public function __construct(
        UserBusinessLogic $userBusinessLogic
    ) {

    }
}
```

```
function testRegistration() {




}
```

```
function testRegistration() {
    $userBusinessLogic =
        new UserBusinessLogicFake();




}
```

```
function testRegistration() {
    $userBusinessLogic =
        new UserBusinessLogicFake();
    $userController =
        new UserController(
            $userBusinessLogic
        );

}
```

```
function testRegistration() {
    $userBusinessLogic =
        new UserBusinessLogicFake();
    $userController =
        new UserController(
            $userBusinessLogic
        );
    //Test the user controller
}
```

Dependency Injection

Don’t look for things!

```php
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            $this->container
                 ->get('security.authorization_checker')
                 ->isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

```php
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            $this->container
                 ->get('security.authorization_checker')
                 ->isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

```
$this->container
     ->get('security.authorization_checker')
     ->isGranted('ROLE_SUPER_ADMIN')
```

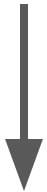
```
$this->container
      ->get('security.authorization_checker')
      ->isGranted('ROLE_SUPER_ADMIN')
```

```
$this->container
     ->get('security.authorization_checker')
     ->isGranted('ROLE_SUPER_ADMIN')
```

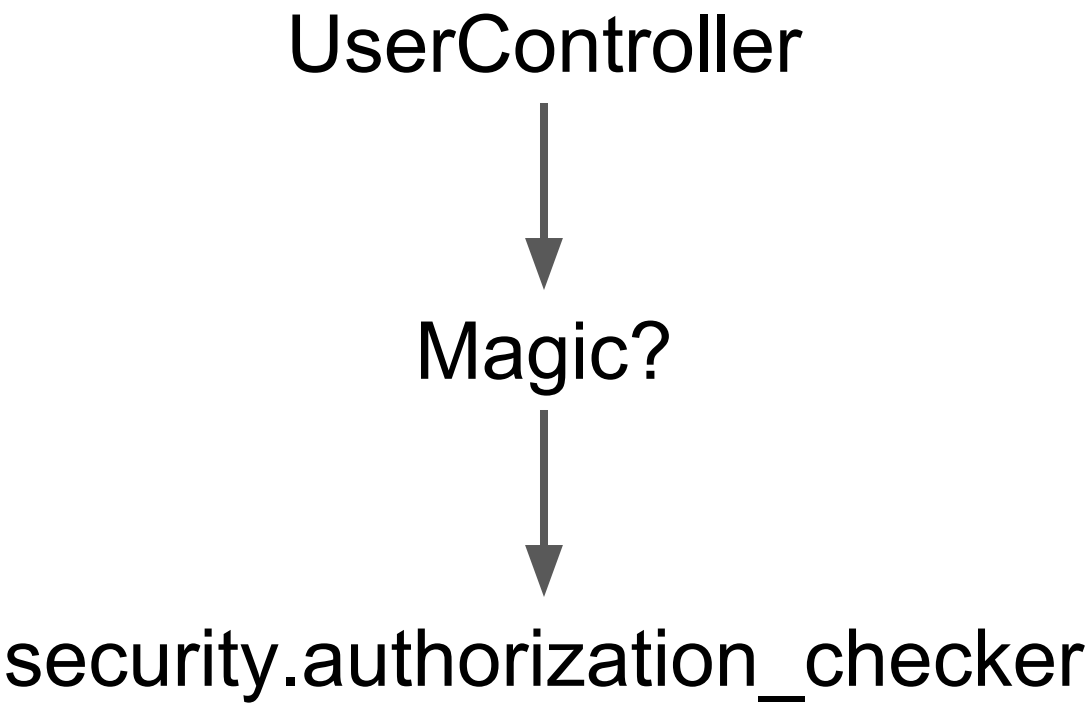
```
$this->container
     ->get('security.authorization_checker')
     ->isGranted('ROLE_SUPER_ADMIN')
```

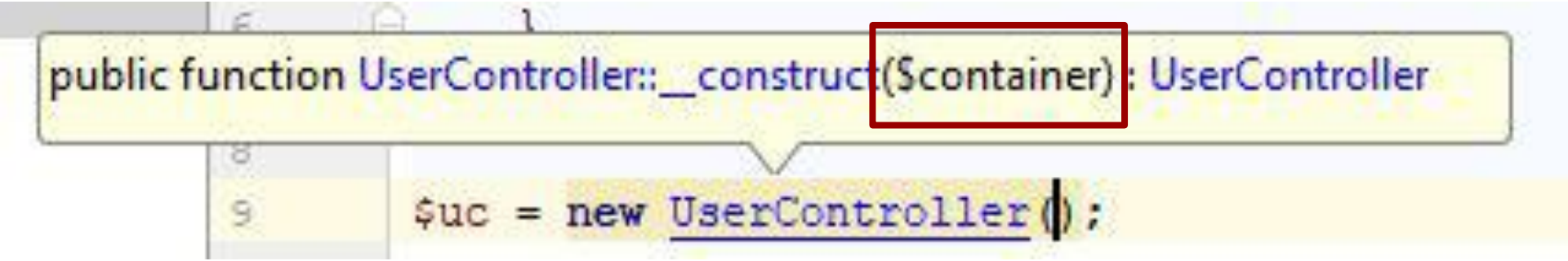
UserController

UserController



Magic?


UserController
Magic?
security.authorization_checker


public function UserController::__construct($container) : UserController
9
$uc = new UserController();

```
class UserController extends Controller {









}
```

```
class UserController extends Controller {

}
```

```
class UserController {

    public function __construct(

    ) {

    }

}
```

```
class UserController {


    public function __construct(
        SecurityAuthorizationChecker  $securityAuthorizationChecker
    ) {

    }









}
```

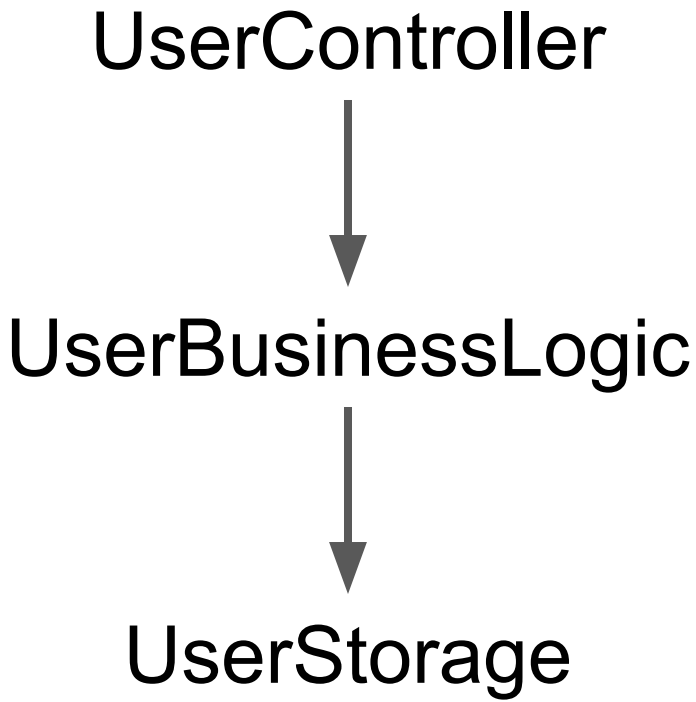
```php
class UserController {
    private $securityAuthorizationChecker;

    public function __construct(
        SecurityAuthorizationChecker $securityAuthorizationChecker
    ) {
        $this->securityAuthorizationChecker = $securityAuthorizationChecker ;
    }









}
```

```php
class UserController {
    private $securityAuthorizationChecker;

    public function __construct(
        SecurityAuthorizationChecker $securityAuthorizationChecker
    ) {
        $this->securityAuthorizationChecker = $securityAuthorizationChecker;
    }

    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if ($this->securityAuthorizationChecker->isGranted('ROLE_SUPER_ADMIN')) {
            return $this->redirectToRoute('admin_dashboard');
        }
    }
}
```

```php
class UserController {
    private $securityAuthorizationChecker;

    public function __construct(
        SecurityAuthorizationChecker $securityAuthorizationChecker
    ) {
        $this->securityAuthorizationChecker = $securityAuthorizationChecker;
    }

    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if ($this->securityAuthorizationChecker->isGranted('ROLE_SUPER_ADMIN')) {
            return $this->redirectToRoute('admin_dashboard');
        }
    }
}
```

Dependency Injectors

Moving the Magic out of your Program


UserController
UserBusinessLogic
UserStorage

```
class UserController {
    public function __construct(
        UserBusinessLogic $userBusinessLogic
    ) {
    }
}
```

```
class UserController {
    public function __construct(
        UserBusinessLogic $userBusinessLogic
    ) {
    }
}
class UserBusinessLogic {
    public function __construct(
        UserStorage $userStorage
    ) {
    }
}
```

```
class UserController {
    public function __construct(
        UserBusinessLogic $userBusinessLogic
    ) {
    }
}
class UserBusinessLogic {
    public function __construct(
        UserStorage $userStorage
    ) {
    }
}
class UserStorage {
}
```

```
$uc = new UserController(
    new UserBusinessLogic(
        new UserStorage()
    )
);
```

```php
$injector = new Injector();
```

```php
$injector = new Injector();

$uc = $injector->make(UserController::class);
```

```
class MySQLConnection {
    public function __construct(
        string $server,
        string $username,
        string $password,
        string $db
    ) {
    }
}
```

```
$injector->define(MySQLConnection::class, [
    'server'   => 'localhost',
    'user'     => 'root',
    'password' => 'changeme',
    'db'       => 'app'
]);
```

- **PHP:** Auryn, Laravel Service Container, Symfony Service Container
- **Java:** Gource, Dagger, Dagger2, Opsbears Web Components DIC
- **Python:** dependency_injector
- **Javascript:** InversifyJS

```php
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            $this->container
                 ->get('security.authorization_checker')
                 ->isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

```php
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            $this->container
                 ->get('security.authorization_checker')
                 ->isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

```php
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            $this->injector
                 ->make(SecurityAuthorizationChecker::class)
                 ->isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

```
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            $this->injector
                ->make(SecurityAuthorizationChecker::class)
                ->isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

Static Function Calls

Might Be Bad For Your Code Quality

```php
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            $this->injector
                 ->make(SecurityAuthorizationChecker::class)
                 ->isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

```php
class UserController extends Controller {
    /**
     * @Route("/user/account", name="user_account")
     */
    public function accountAction(Request $request) {
        if (
            SecurityAuthorizationChecker
                ::isGranted('ROLE_SUPER_ADMIN')
        ) {
            return $this->redirectToRoute('admin_dashboard');
        }
        // Other stuff here
    }
}
```

```php
$uc = new UserController();
```

Immutable Objects

Avoiding Surprises

```php
class User {
    private $id;

    public function setId($id) {
        $this->id = $id;
    }

    public function getId() {
        return $this->id;
    }
}
```

```
new User()
```

```php
class User {
    private $id;

    public function setId($id) {
        $this->id = $id;
    }

    public function getId() {
        return $this->id;
    }
}
```

```php
class User {
    private $id;

    public function setId($id) {
        $this->id = $id;
    }

    public function getId() {
        return $this->id;
    }
}
```

```php
class User {
    private $id;

    public function __construct($id) {
        $this->id = $id;
    }

    public function getId() {
        return $this->id;
    }
}
```

```php
class UserStorage {
    private $users = [];

    public function store(User $user) {
        $this->users[$user->getId()] = $user;
    }

}
```

```php
class UserStorage {
    private $users = [];

    public function store(User $user) {
        $this->users[$user->getId()] = $user;
    }

    public function retrieve($id) {
        if (isset($this->users[$id])) {
            return $this->users[$id];
        } else {
            throw new UserNotFoundException($id);
        }
    }
}
```

Less Code in One Class

Your All-In-One Weightloss Program

```
class UserController {
    public function register() {}

    public function search() {}

    public function get() {}

    public function update() {}

    public function delete() {}
}
```

```
Route::get(
    '/users',
    function (

    ) {
        return 'User list';
    }
);
```

```
Route::get(
    '/users',
    function (
        UserBusinessLogic $userBusinessLogic
    ) {
        return 'User list';
    }
);
```

```
class UserRegisterController {
    public function __construct(
        UserBusinessLogic $userBusinessLogic
    ) {
    }

    public function register() {
    }
}
```

Static Typing

*Saves you from a **** ton of issues*

```
function search(
    $needle,
    $haystack
) {

}
```

```php
function search(
    string $needle,
    array $haystack
) {

}
```

JavaScript	Typescript
PHP	phpstan
Python	mypy
Java	*builtin*

Strict Typing

Because your String is not an Integer

PHP

```php
<?php

declare(strict_types=1);
```

PHP

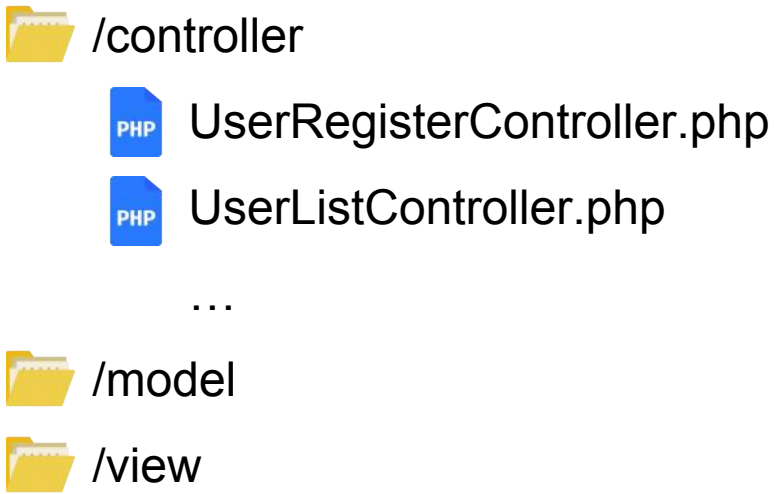
```php
<?php

declare(strict_types=1);
```

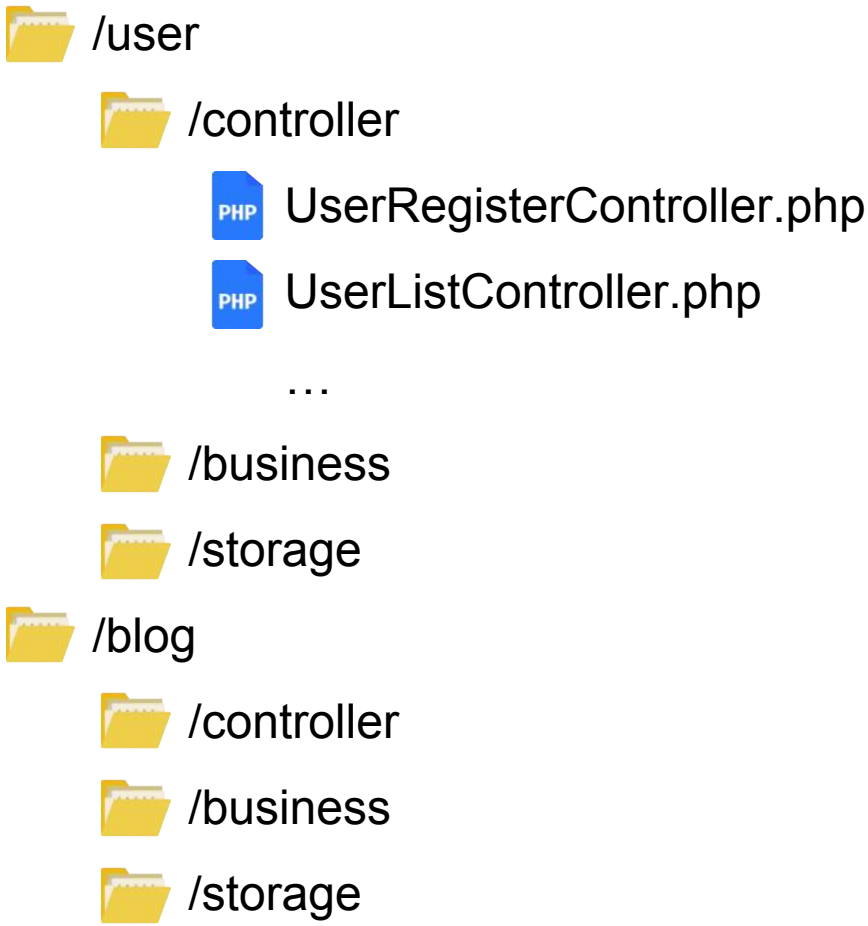
JavaScript

Typescript

Structuring your Code

Because your Code is not a Clown Car

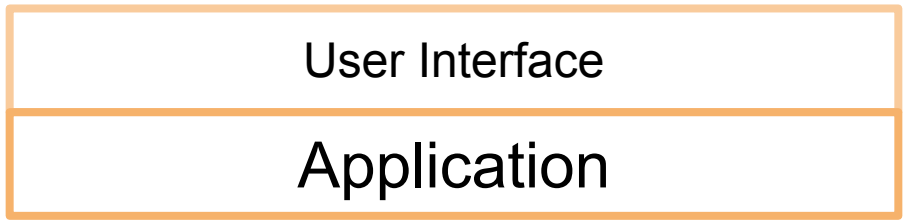

/controller
UserRegisterController.php
UserListController.php
…
/model
/view

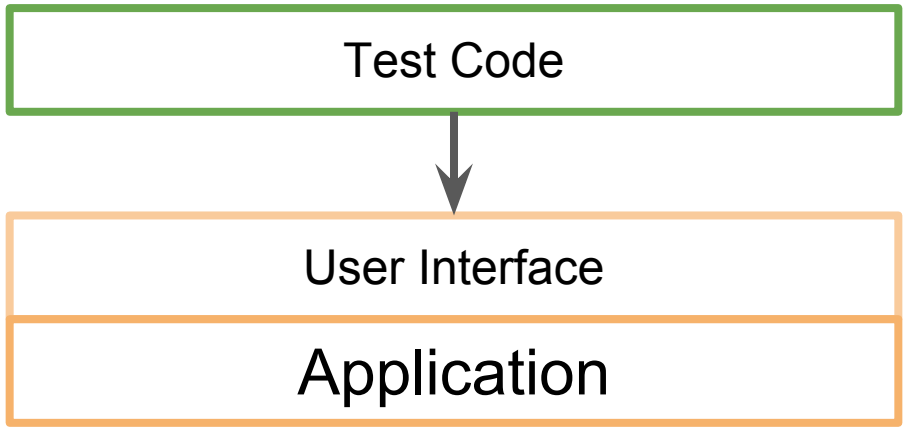

/user
/controller
PHP
UserRegisterController.php
PHP
UserListController.php
…
/business
/storage
/blog
/controller
/business
/storage

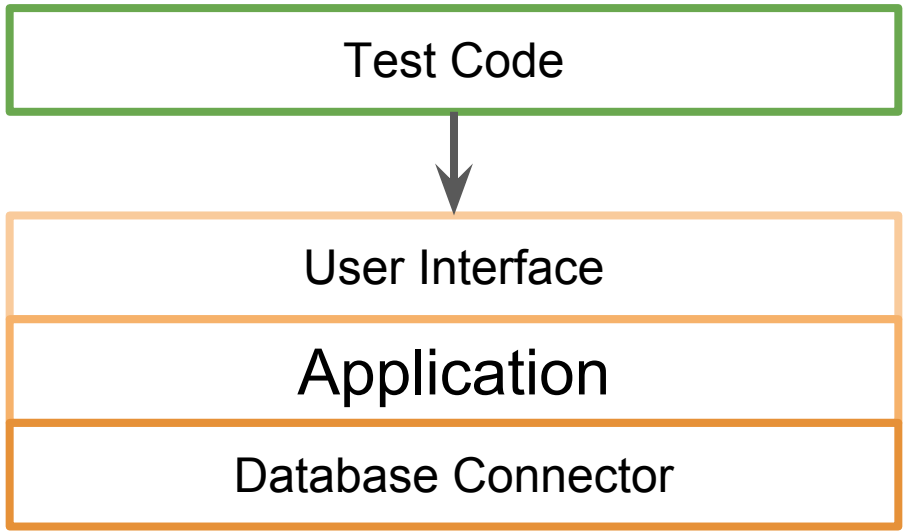
Testing

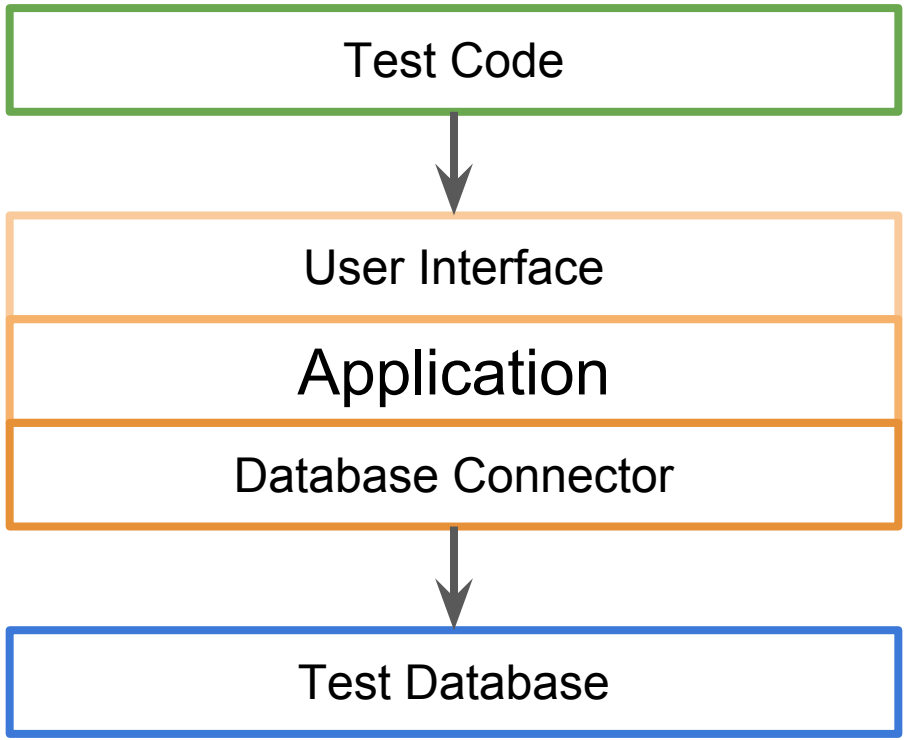
You test your code, right?

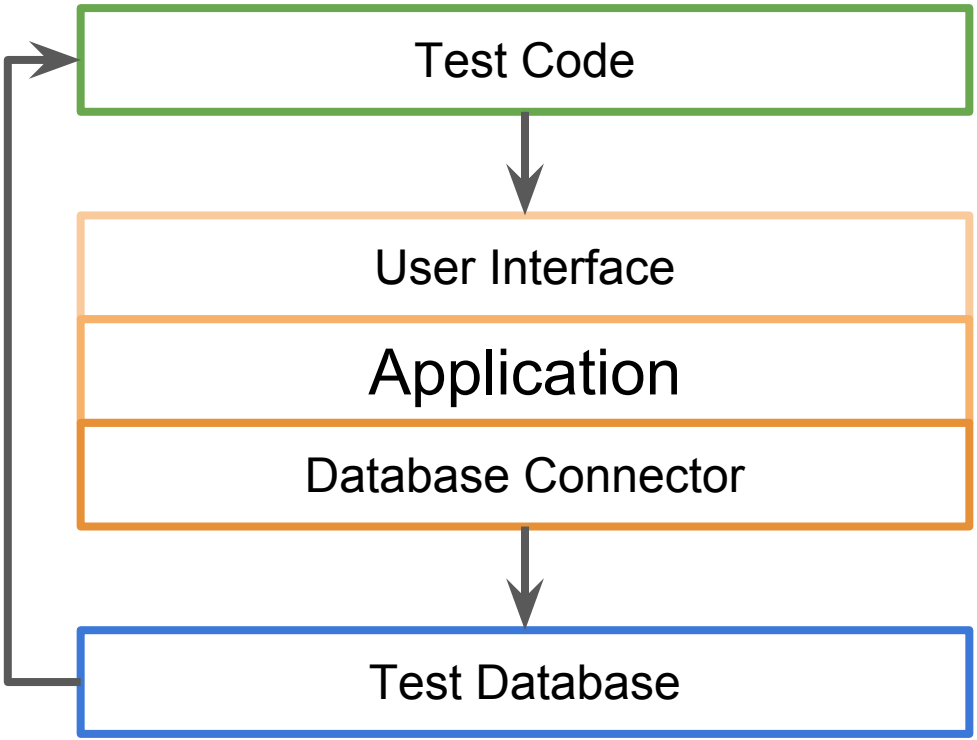
Application


User Interface
Application

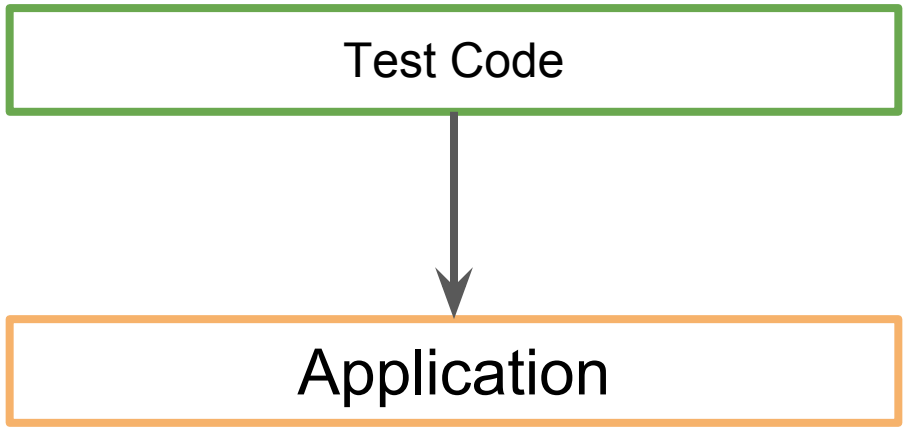

Test Code
User Interface
Application

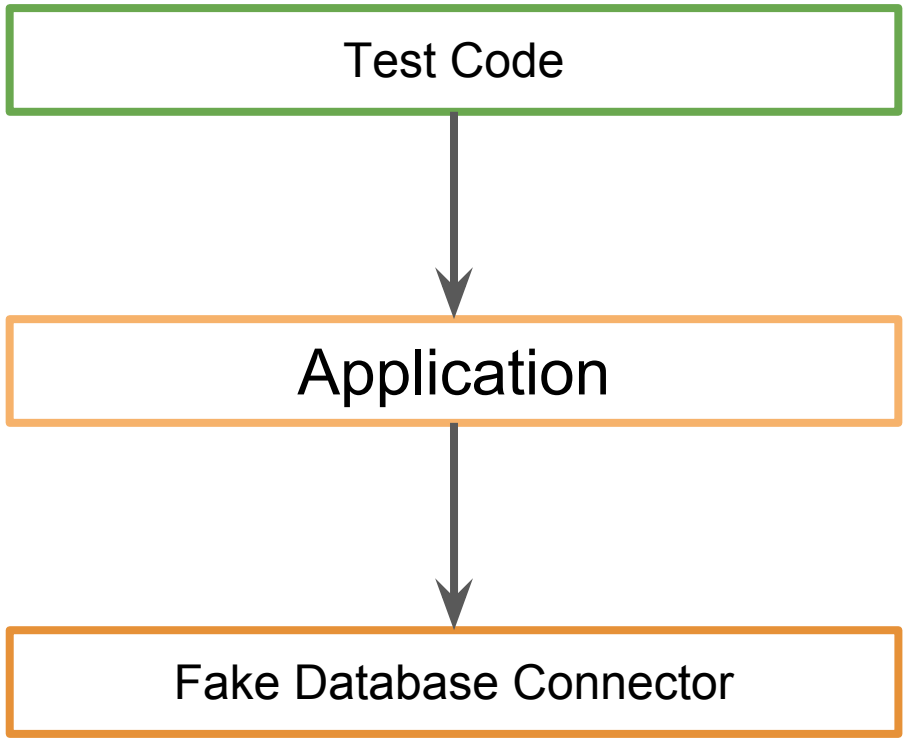

Test Code
User Interface
Application
Database Connector

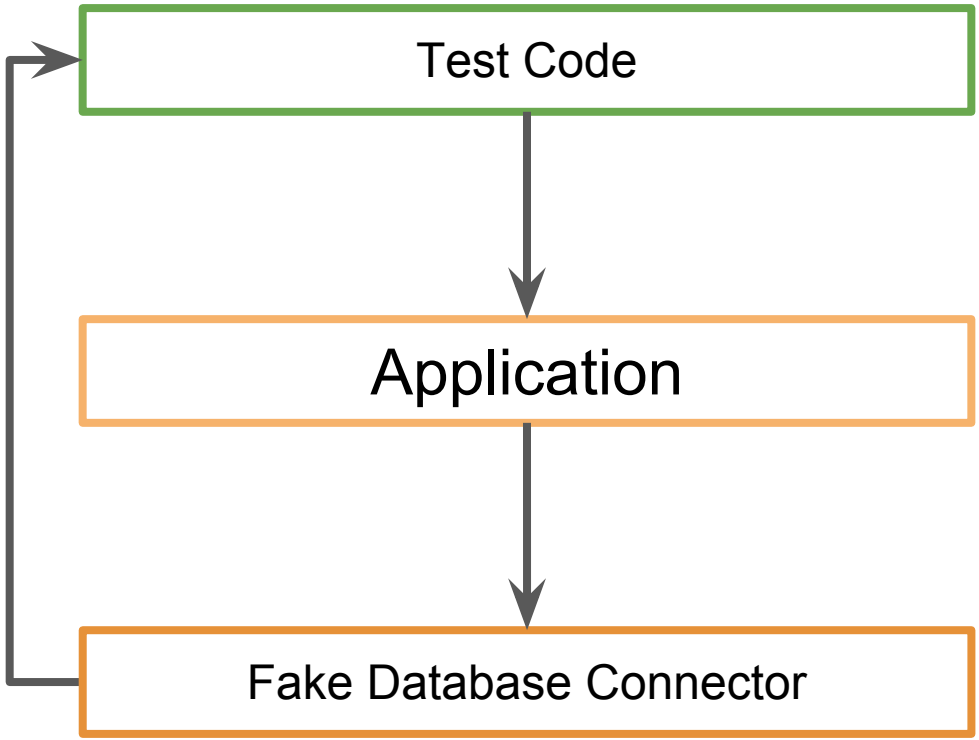

Test Code
User Interface
Application
Database Connector
Test Database

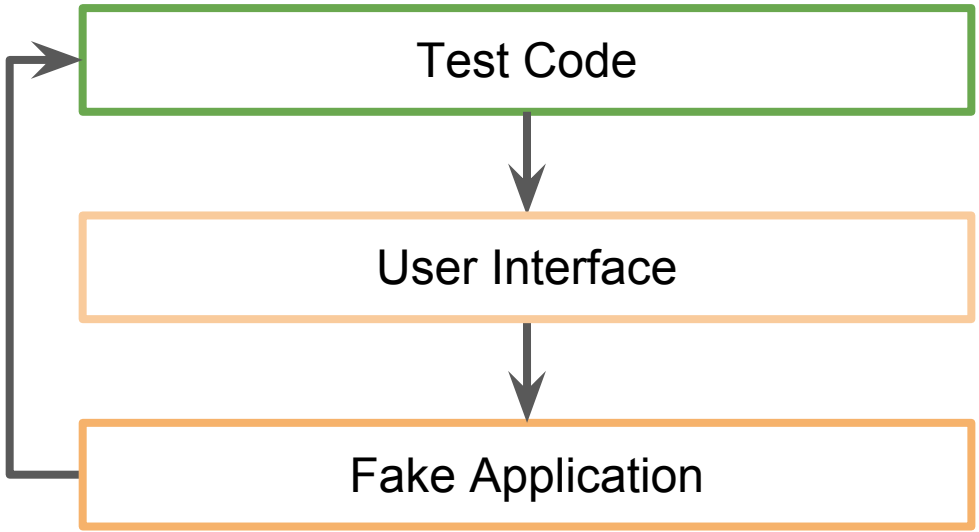

Test Code
User Interface
Application
Database Connector
Test Database

Application


Test Code
Application


Test Code
Application
Fake Database Connector


Test Code
Application
Fake Database Connector


Test Code
User Interface
Fake Application

```
function testGetShouldReturnUser(){
    //region Setup
    $userStorage = new UserStorageFake();
    $userStorage->backingStorage['test-user'] =
        UserFactory::create(
            "test-user",
            "Test User",
            "test@example.com",
            "*"
        );
    $business = new UserGetBusinessLogicImpl($userStorage);
    //endregion

    //region Execute...

    //region Asset...
}
```

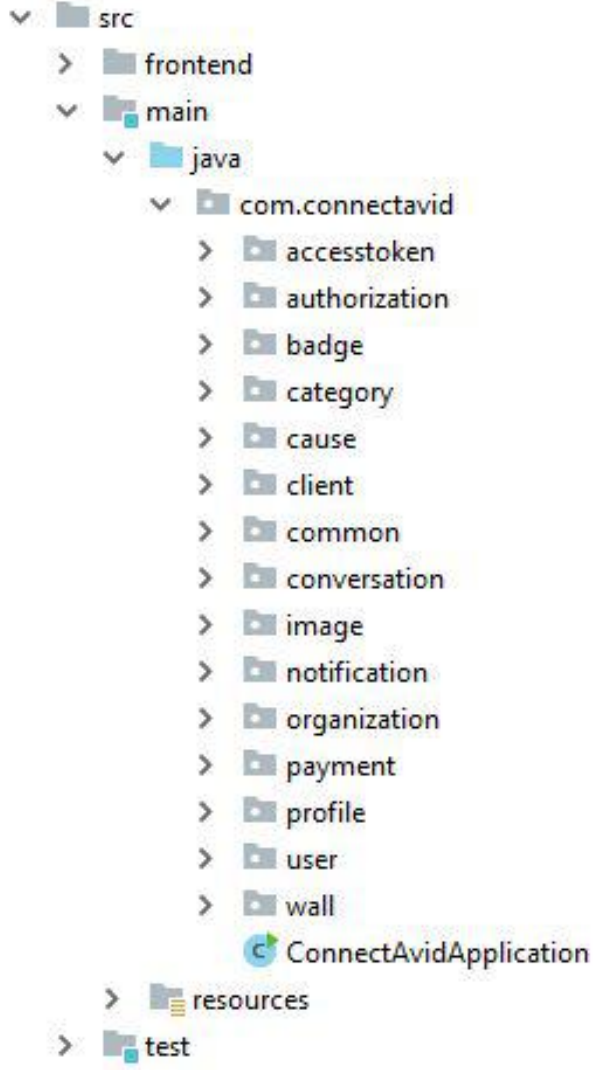
```
function testGetShouldReturnUser(){
    //region Setup...

    //region Execute
    $user = $business->getById("test-user");
    //endregion

    //region Assert
    assertEquals("test-user", $user->getId());
    //endregion
}
```

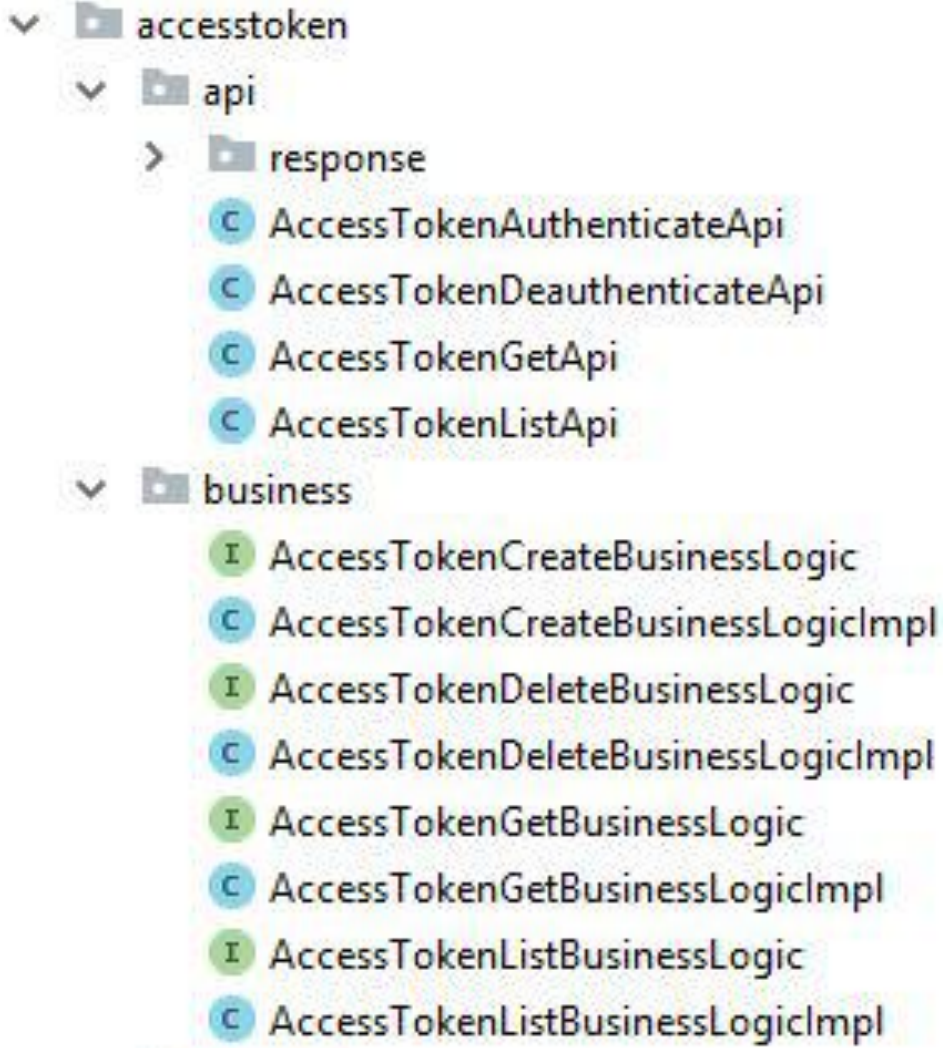
Putting it together

Building an actual system

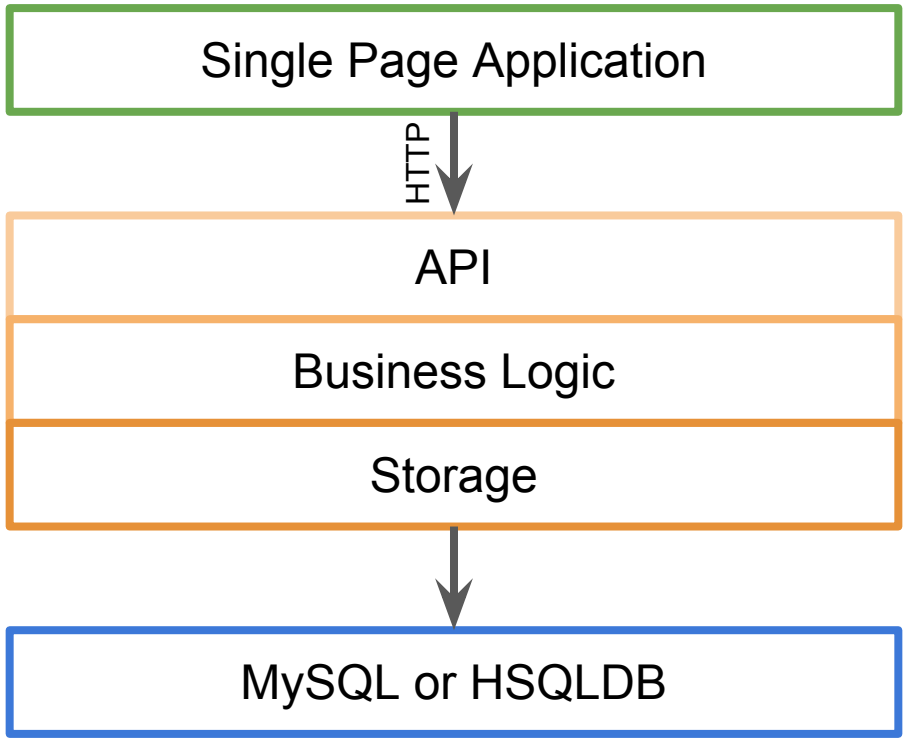
CONNECTAVID

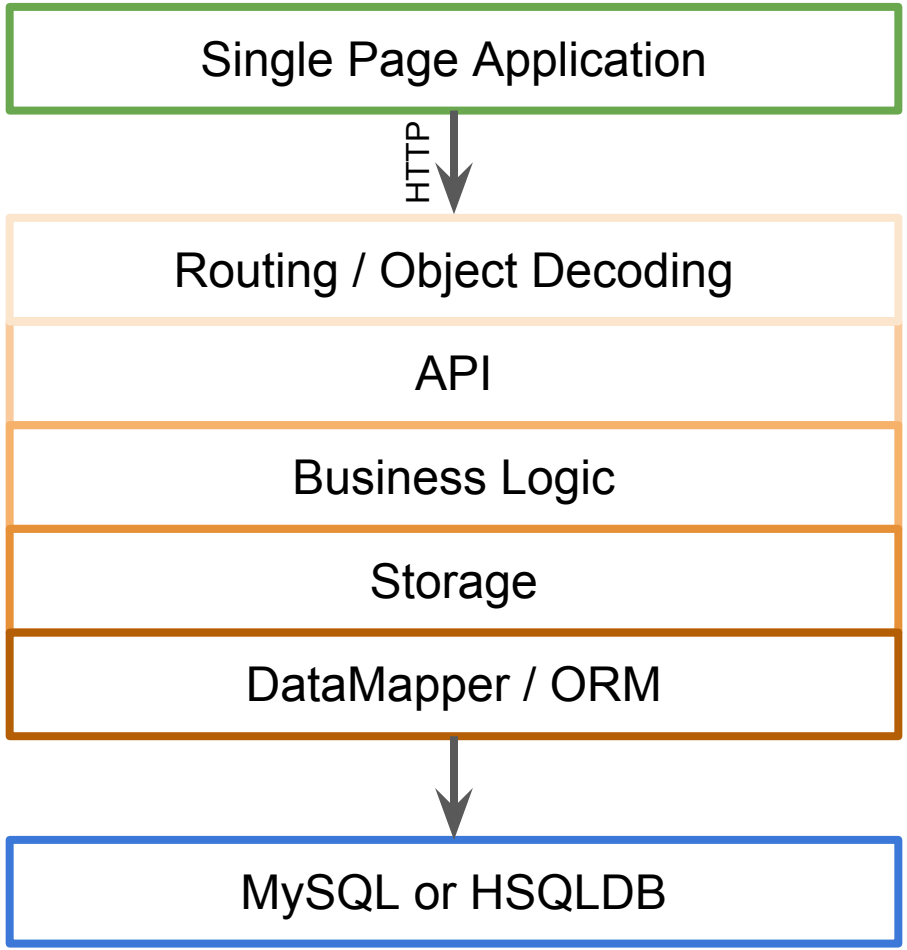
Many thanks to:
Cristina Laskar


src
frontend
main
java
com.connectavid
accesstoken
authorization
badge
category
cause
client
common
conversation
image
notification
organization
payment
profile
user
wall
ConnectAvidApplication
resources
test


accesstoken
api
response
AccessTokenAuthenticateApi
AccessTokenDeauthenticateApi
AccessTokenGetApi
AccessTokenListApi
business
AccessTokenCreateBusinessLogic
AccessTokenCreateBusinessLogicImpl
AccessTokenDeleteBusinessLogic
AccessTokenDeleteBusinessLogicImpl
AccessTokenGetBusinessLogic
AccessTokenGetBusinessLogicImpl
AccessTokenListBusinessLogic
AccessTokenListBusinessLogicImpl


entity
AccessToken
exception
AccessTokenAlreadyExistsException
AccessTokenNotFoundException
migration
hsqldb
mysql
storage
AccessTokenStorageCreate
AccessTokenStorageDelete
AccessTokenStorageGetById
AccessTokenStorageListByUserId
DataMapperAccessTokenStorage
AccessTokenModule

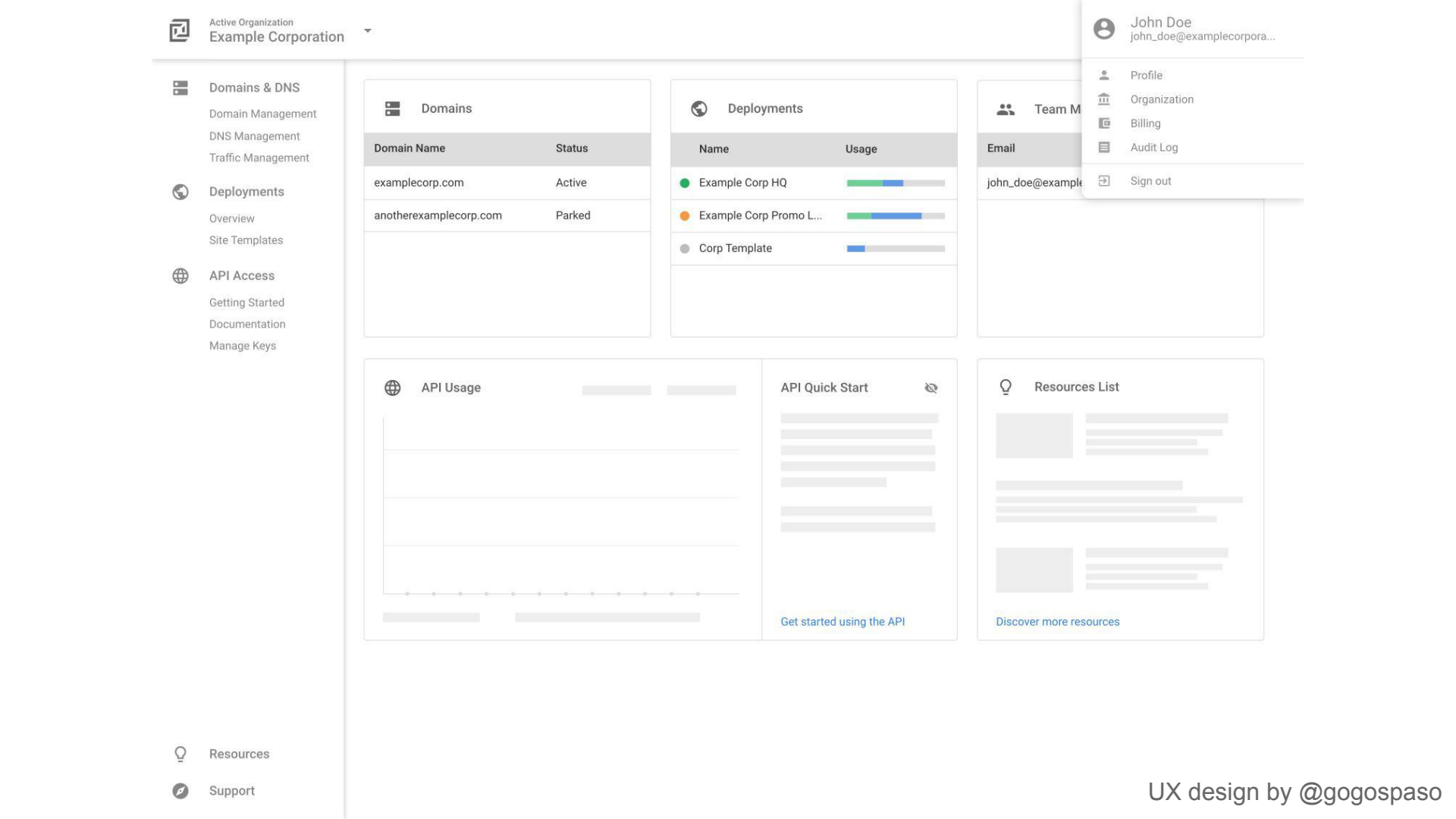

Single Page Application
HTTP
API
Business Logic
Storage
MySQL or HSQLDB


Single Page Application
HTTP
Routing / Object Decoding
API
Business Logic
Storage
DataMapper / ORM
MySQL or HSQLDB


SWAGGER

Non-technical Ways

Customer Communication is Important



UX design by @gogospaso

More information

Because one talk is not enough

Questions?

www.pasztor.at