Mage TITANS

MCR 2018

VALENTIN BOYANOV

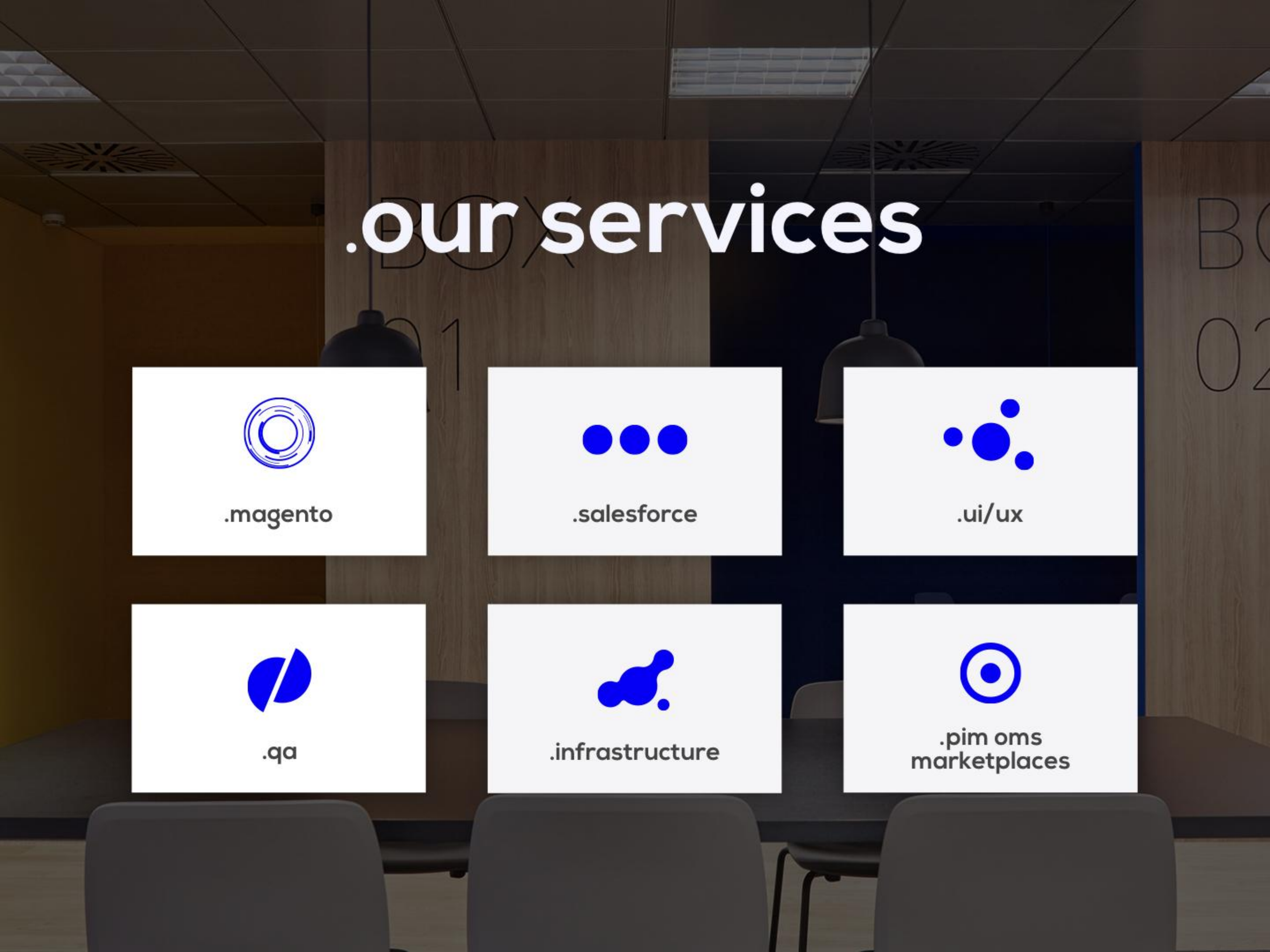
@kolbyfloyd

The True Purpose of Testing

UK.MAGETITANS.COM

#MageTitansMCR @MageTitans

.everyone



.our services

.magento

.salesforce

.ui/ux

.qa

.infrastructure

.pim oms
marketplaces

we work

WITH BIG AND SMALL CLIENTS

Abacus Cooperativa

BERING

casa viva

Castañer

El Celler de Can Roca Girona

Grupo Cofares

Dia %

Druni Perfumerías

DVD Developing Value in Dentistry

Hackett London

nice things Paloma S.

Lékué

MARCA

Oxfam Intermón

PANDORA

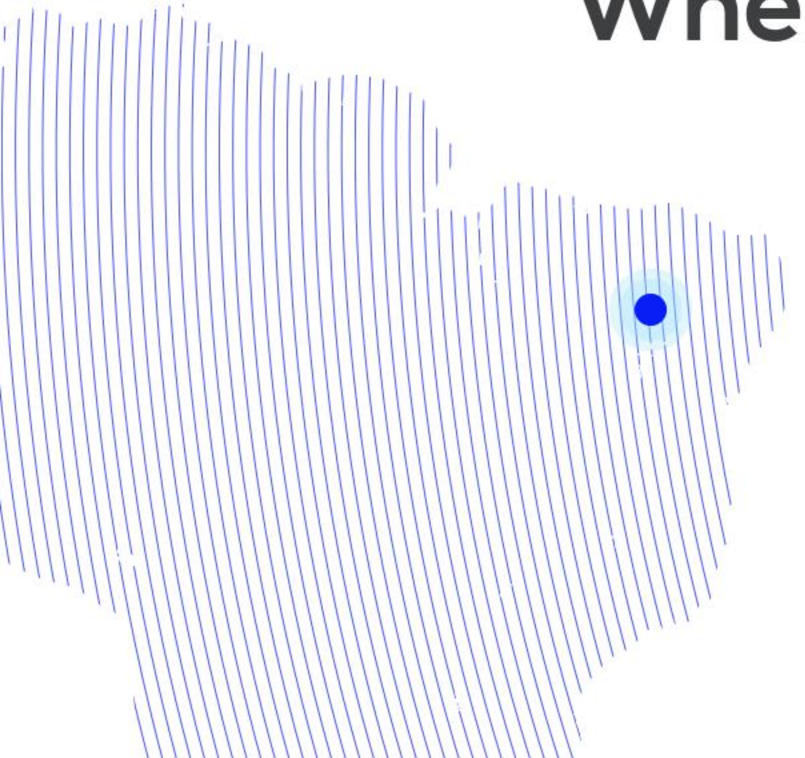
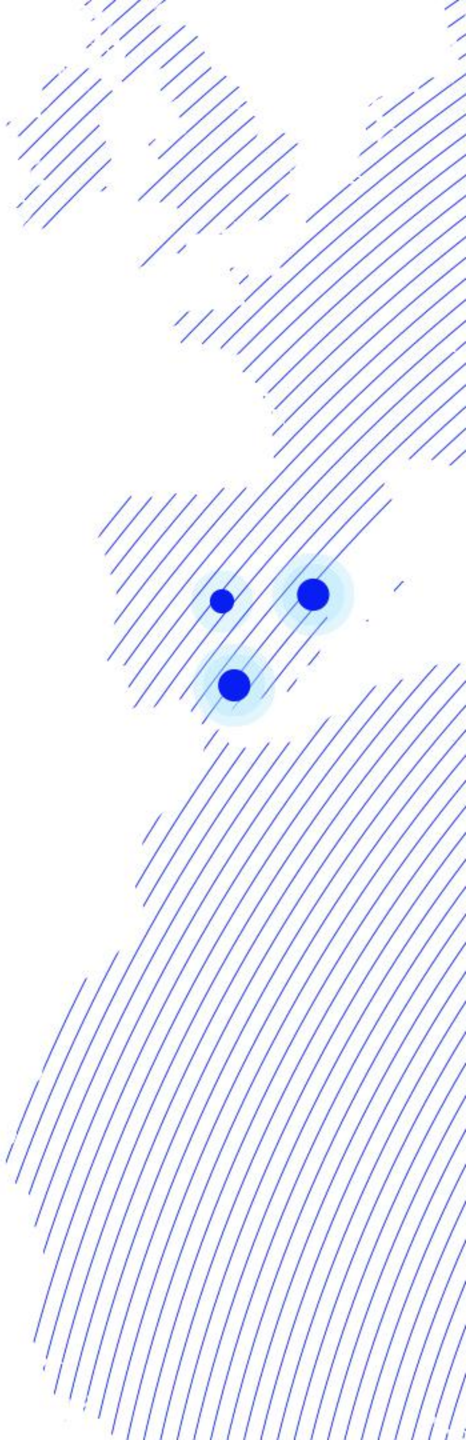
Pepe Jeans London

PG

Satorisan

PRONOVIAS BARCELONA

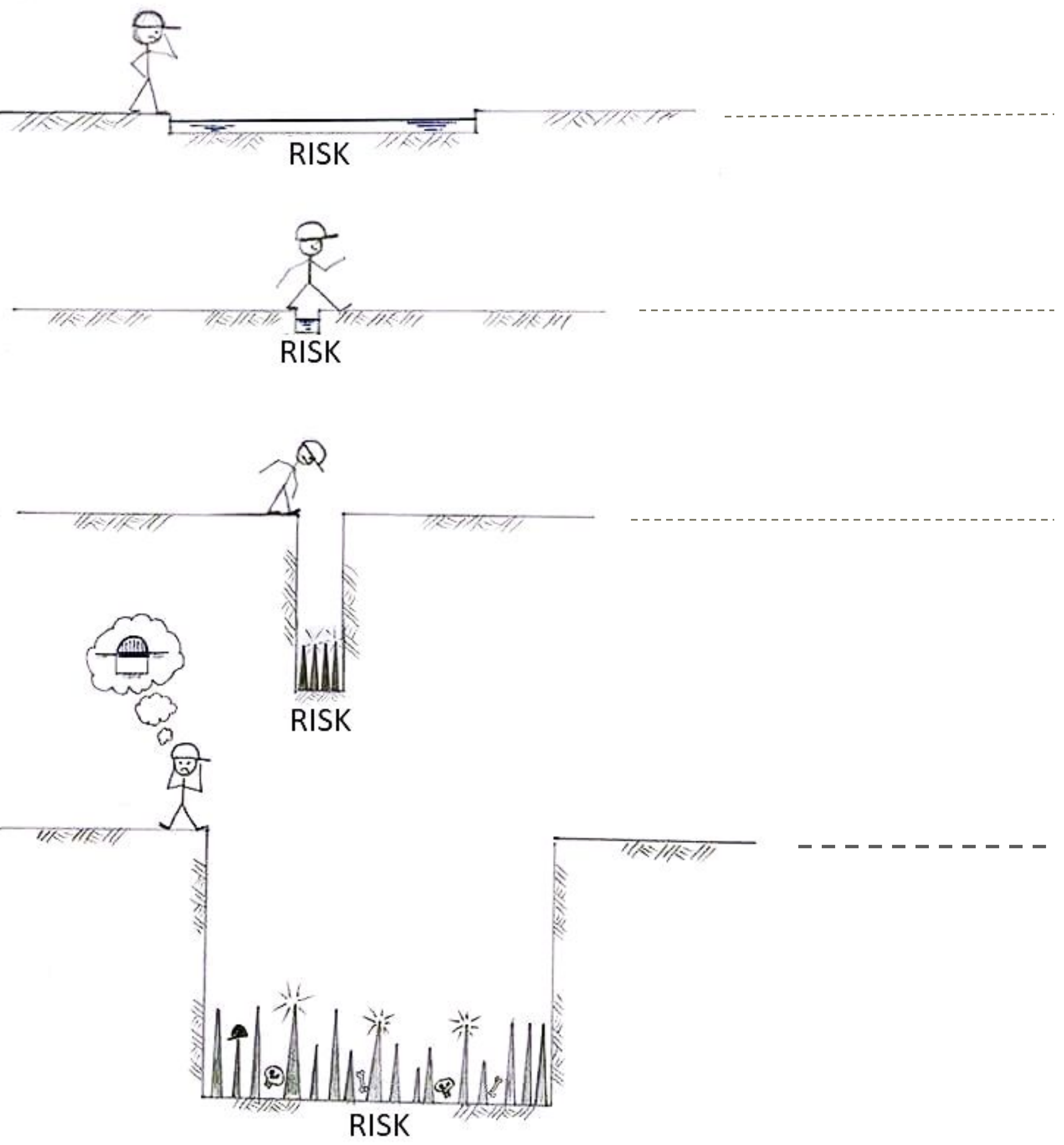
TEXTURA interiors

Where are we?

The true purpose of testing

Why am I interested in this?



Low Probability
Low Impact

Low Probability
Low Impact

Low Probability
High Impact

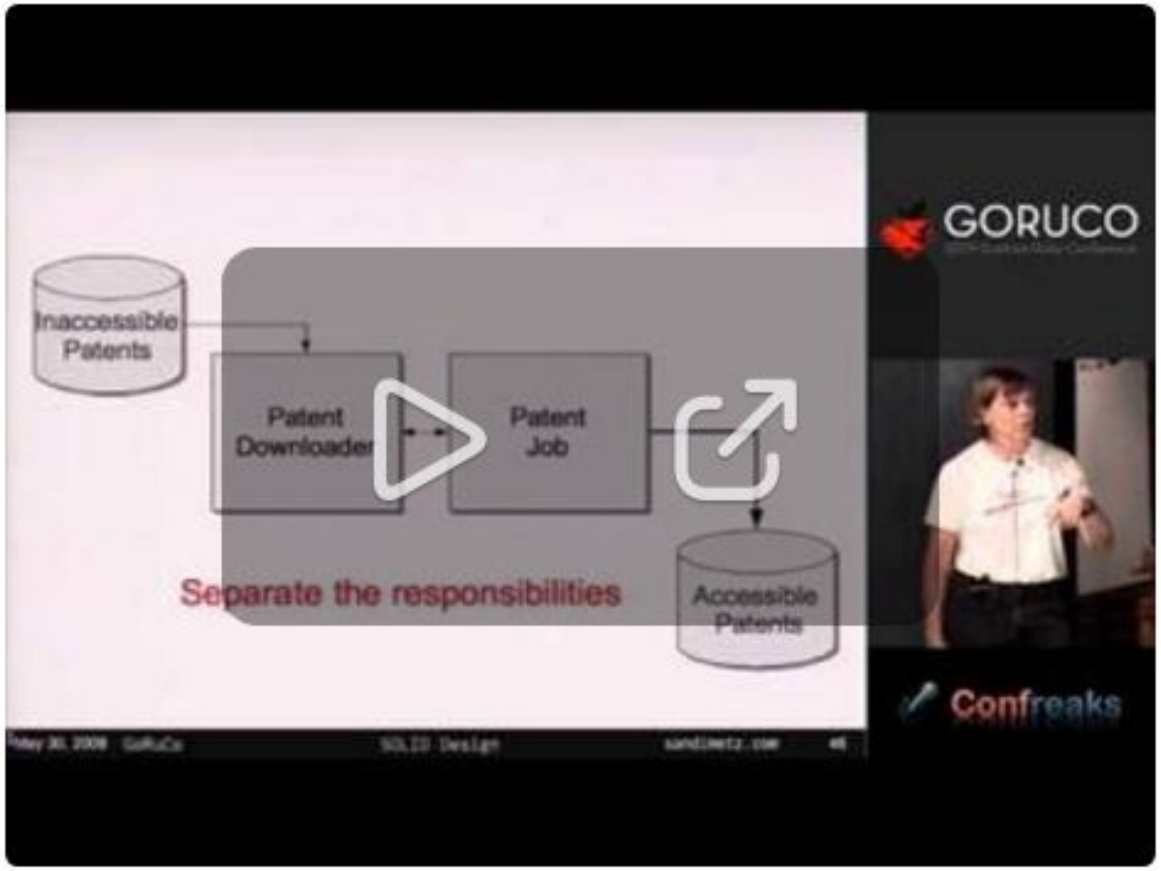
High Probability
High Impact

rafa 9:30 AM

https://www.youtube.com/watch?v=8STtzjyDTTQ

YouTube | Imran Ismail

Sandy Metz - SOLID Design Principle in Ruby



Este es el video del que te hablé (edited)

“The most common arguments for having tests are that they **reduce bugs** and **provide documentation**, and that writing tests first **improves application design**. These benefits, however valid, are proxies for a deeper goal. The true purpose of testing, just like the true purpose of design, is to **reduce costs**.”

— **Sandi Metz** in *Practical Object-Oriented Design, An Agile Primer Using Ruby (POODR)*

Software Costs

Based on **Konstantin Kudryashov's** talk at Laracon EU 2015

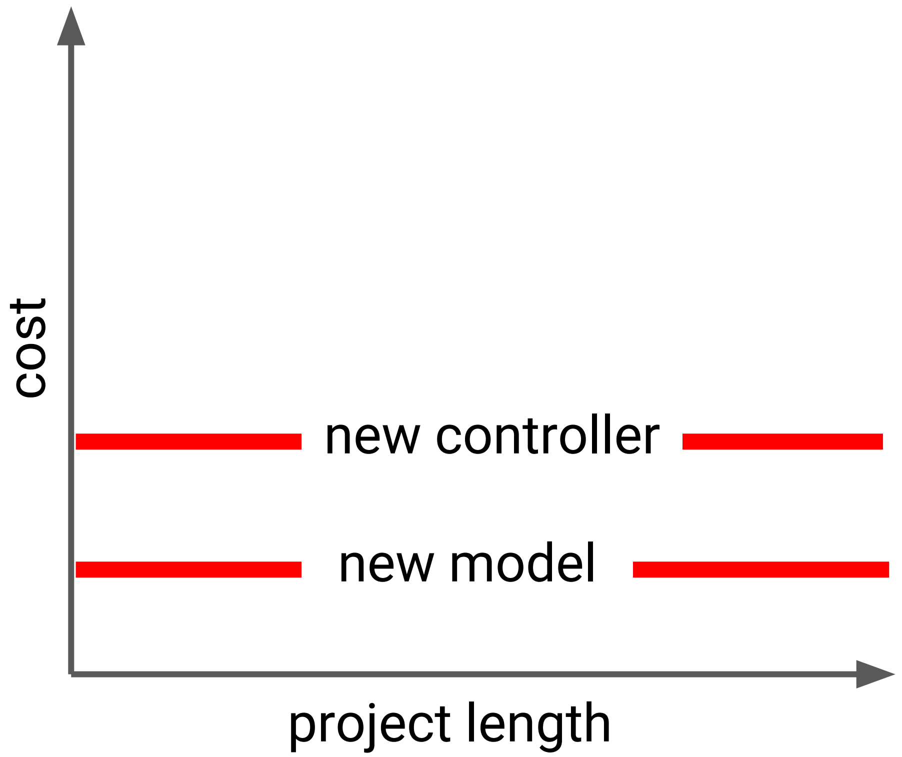
Cost of **Introduction** - Time to write & test code
Cost of **Change** - Time to change code & tests
Cost of **Ownership** - Time to refactor code & tests

Cost of Introduction

The time it takes to introduce new, naturally independent, decoupled, application logic

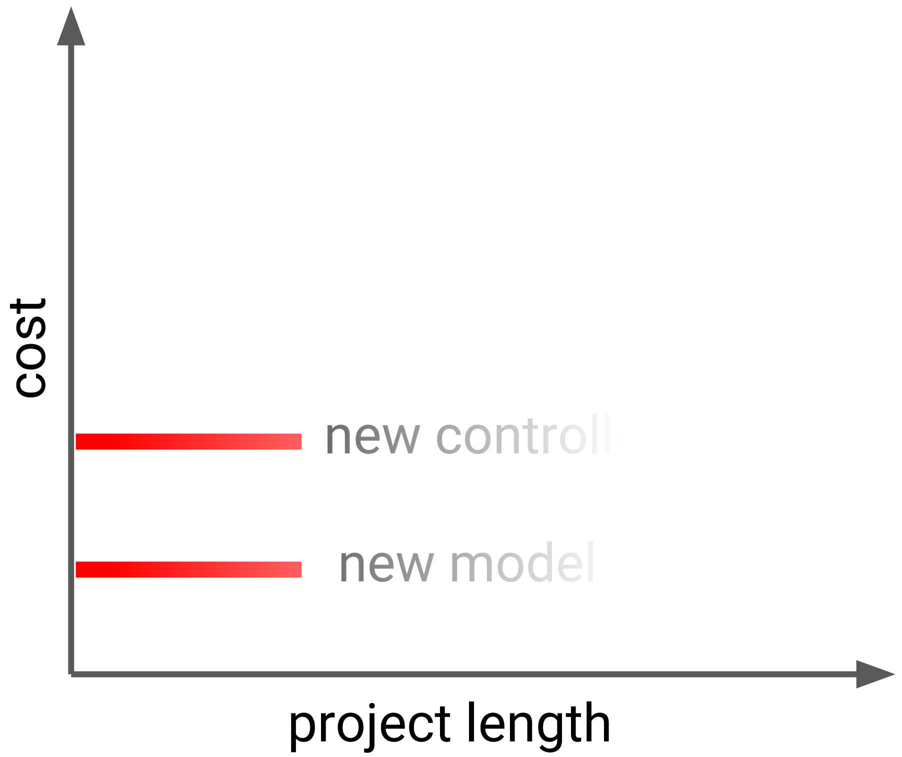
Attributes of Cost of Introduction

- Direct related to business value
- Direct relation to LOC
- Relatively easy to optimise by generalisation



Dynamics of Cost of Introduction

- Visible from the beginning
- Loses relevancy over the project lifetime
- Stable across projects



Optimizing Cost of Introduction

If the life of our project is short, then the cost of introduction is the only cost worth optimizing for.

Change is the only constant

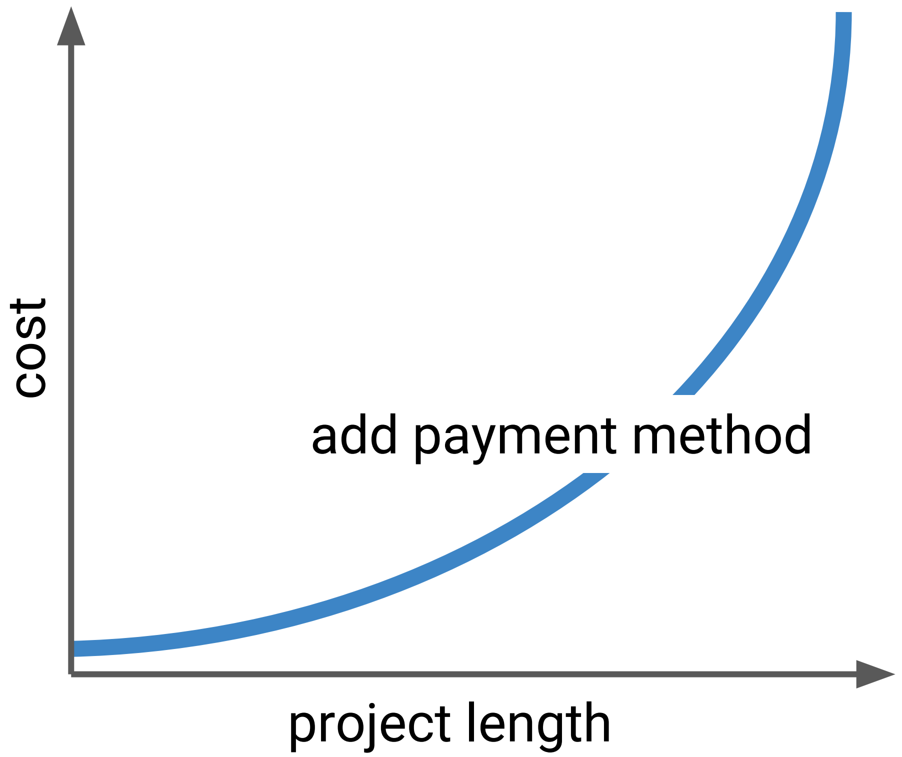
— Heraclitus of Ephesus

Cost of Change

The time it takes to adapt existing application logic to the new business realities

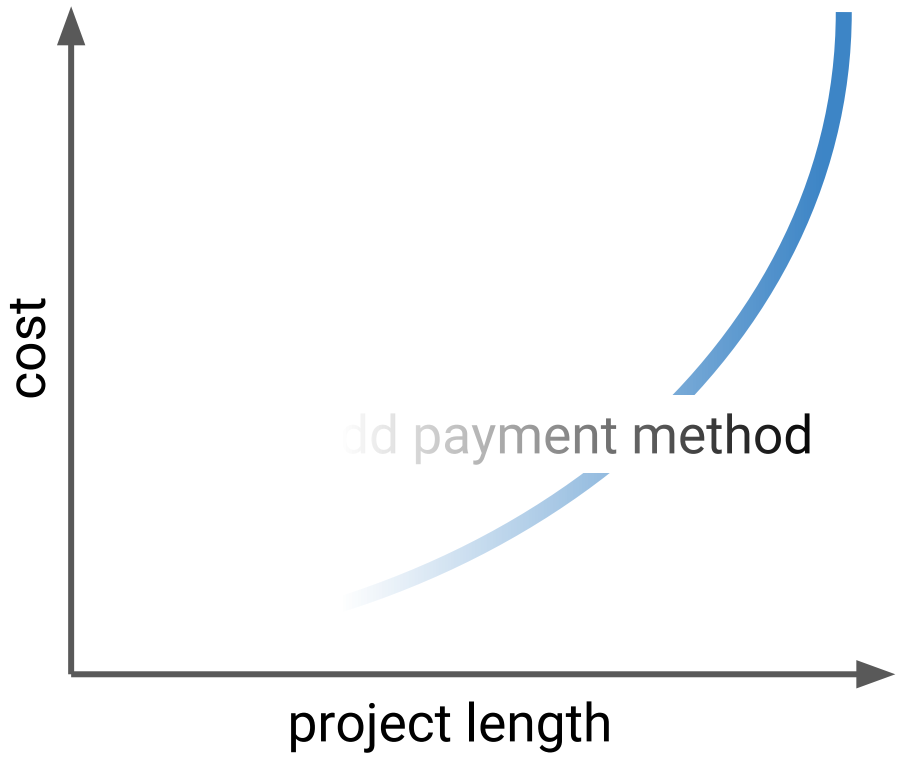
Attributes of Cost of Change

- Direct relation to business value
- No direct relation to LOC
- Affected by generalisation



Dynamics of Cost of Change

- Invisible from the beginning
- Gains relevancy during the project lifetime
- Exponentially increasing over time



Optimizing Cost of Change

If the product life is long enough to encounter exponential growth, then the cost of change is the only cost worth optimizing for.

Upfront Design fails

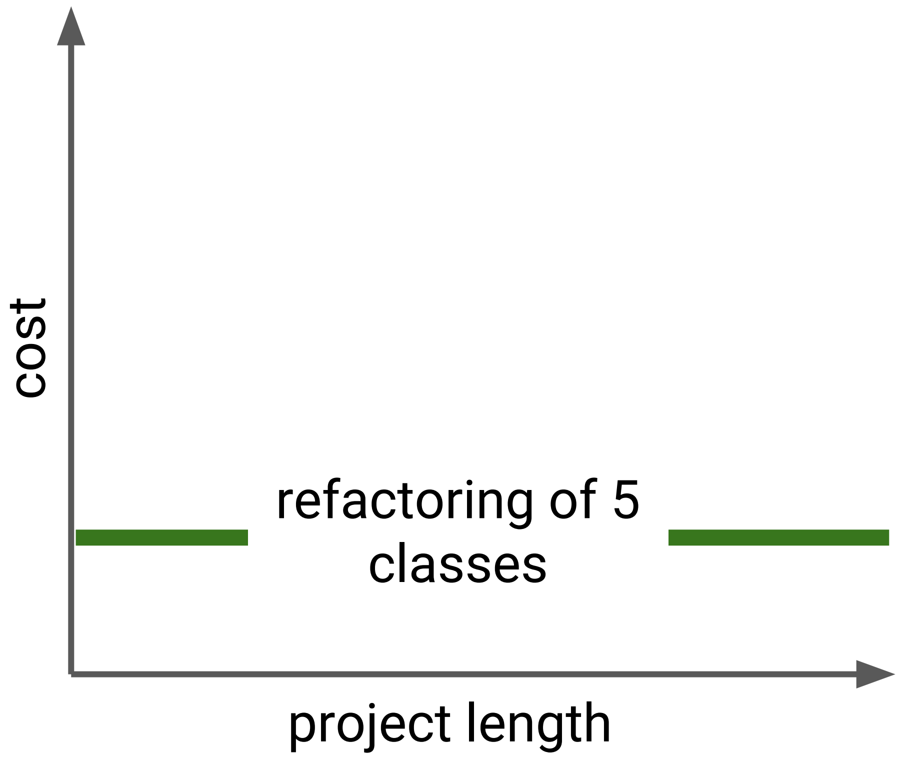
Controlling cost of change by applying enough upfront analysis is an illusion

Cost of Ownership

The time it takes to maintain the own application logic to support its ongoing change

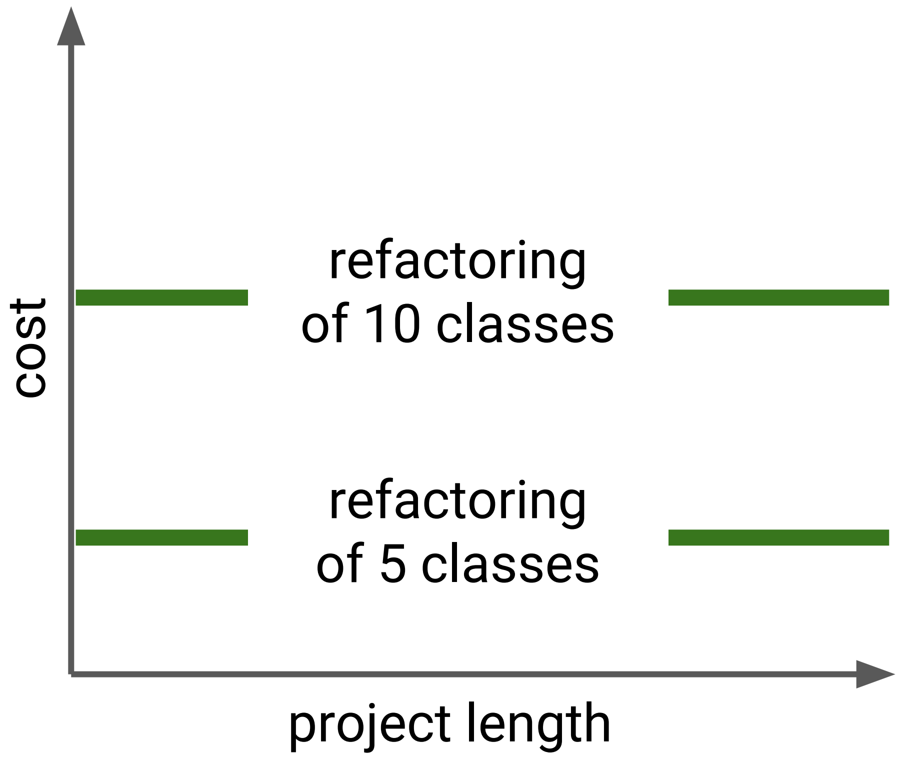
Attributes of Cost of Ownership

- Intermediate between Cost of Introduction and Cost of Change
- No direct relation to business value
- Direct relation to LOC



Dynamics of Cost of Ownership

- Always invisible
- Always relevant
- Stable over time, but adds up



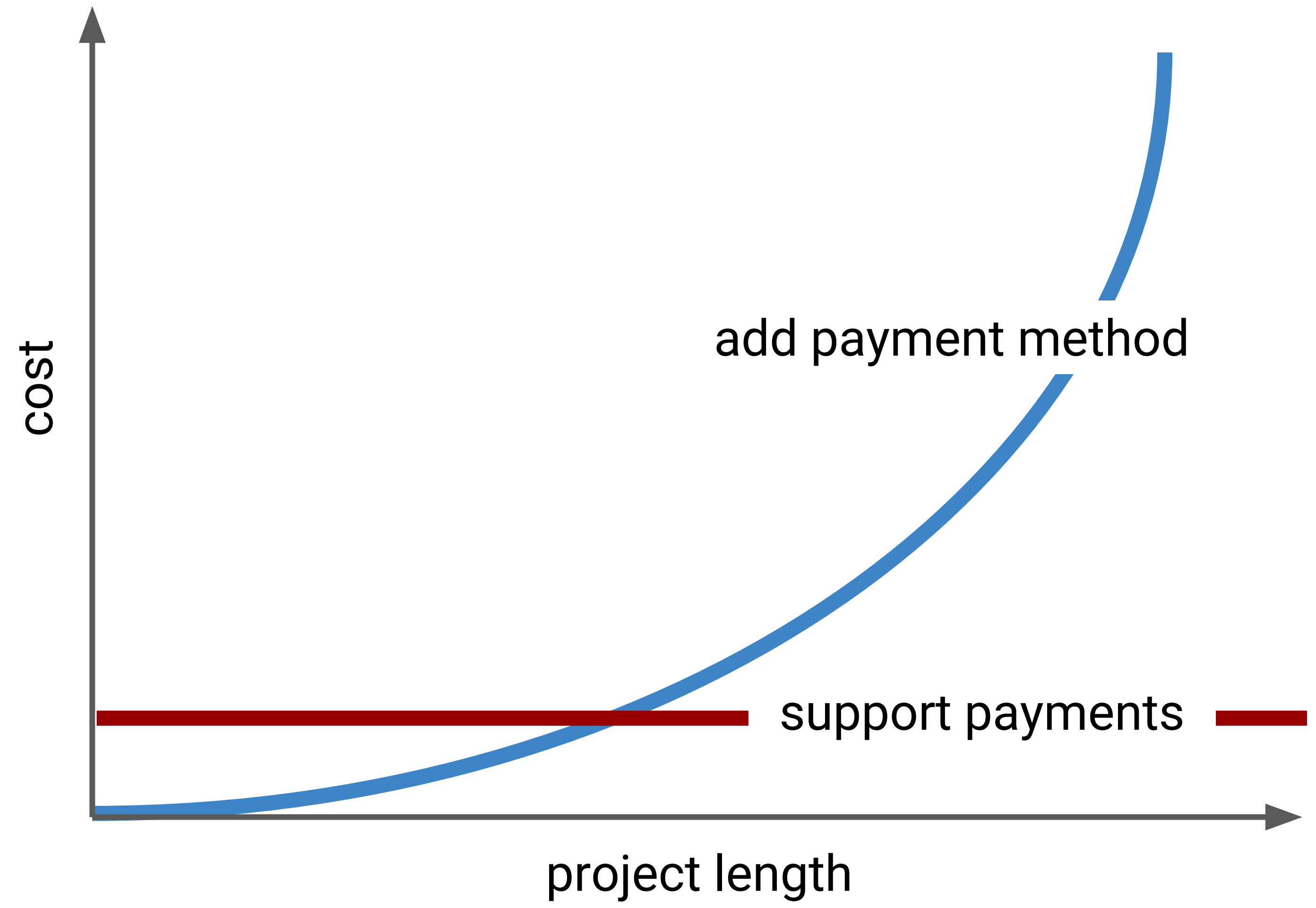
The right to change

Cost of ownership is the cost we pay for the right to change a particular part of the application (module, class, etc.) continuously and sustainably.

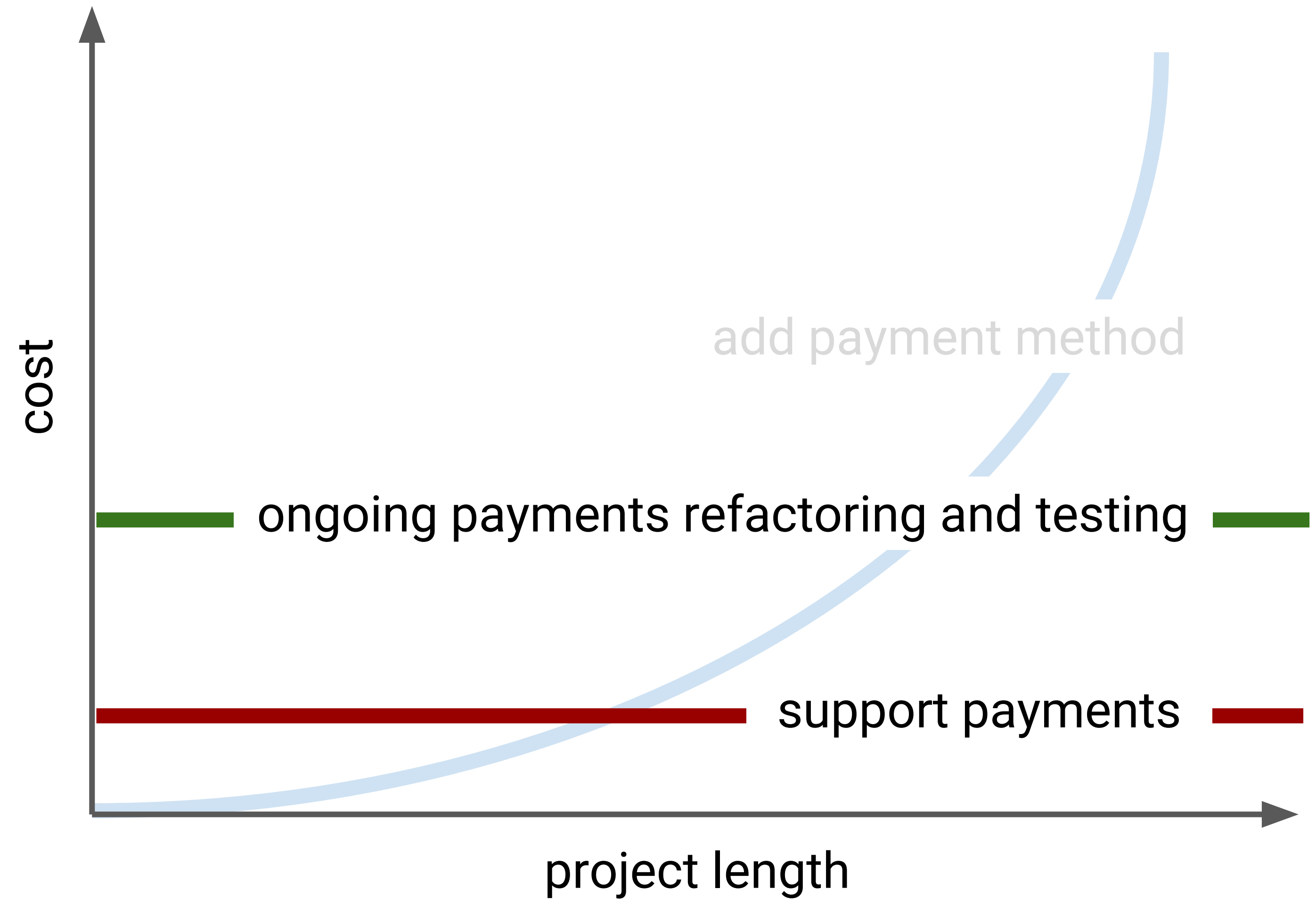
Unit Testing

Unit testing a particular class is a direct statement of owning that class, of owning that method.

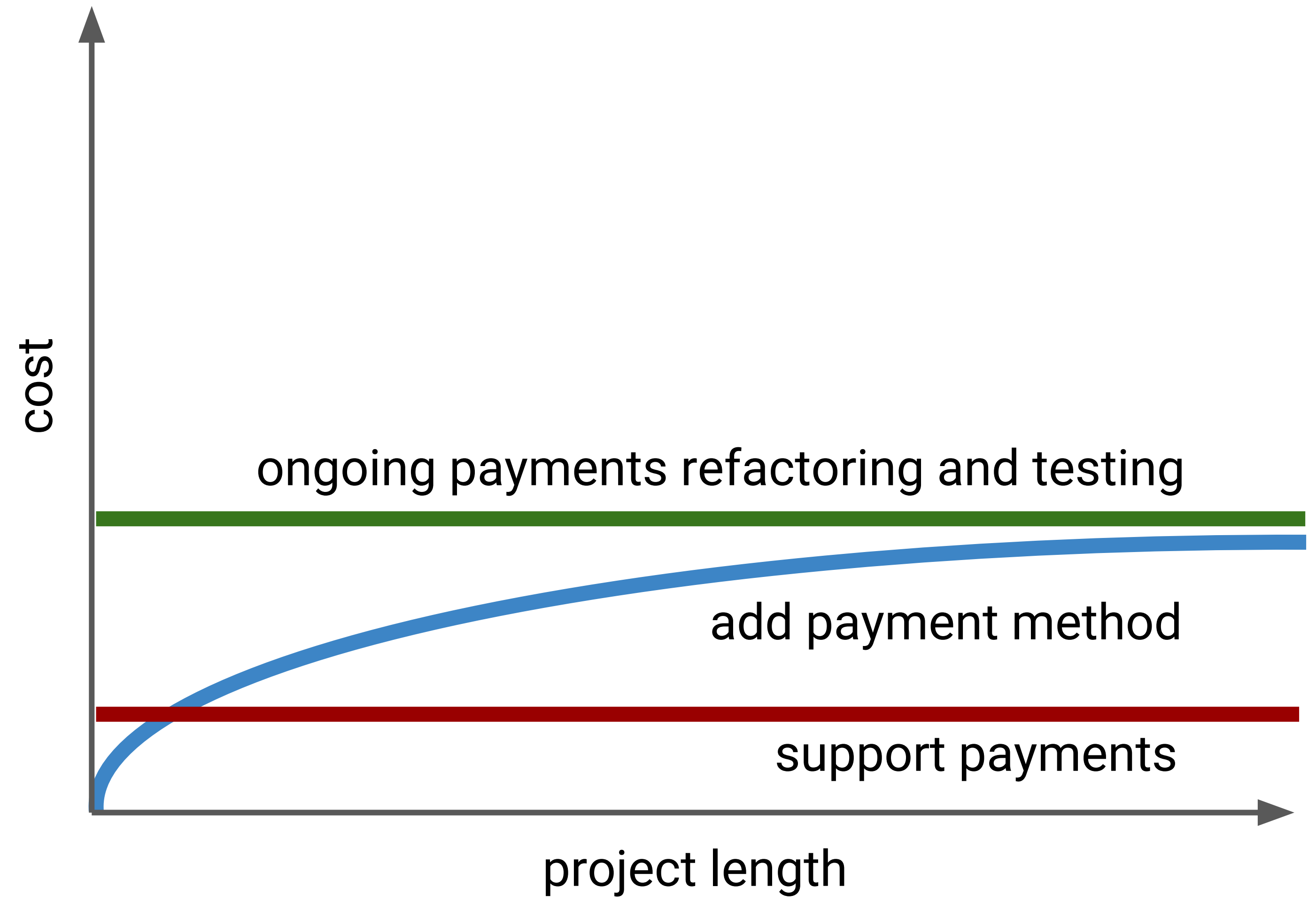
Cost of Ownership effect on **Cost of Change**



Cost of Ownership effect on **Cost of Change**



Cost of Ownership effect on **Cost of Change**



Emergent Design

What changes would we need to make next?
Is it simple enough?
Can we decouple?

TDD is an ownership technique!

It helps to refactor things to support future changes.

Let’s see a Magento example…

Story: Inform customer about free shipping
In order to increase average order value
As a store owner
I want to inform the customer in the header about the free shipping status.

Scenario 1: The cart is below the minimum order amount
Given that the customer cart is “75€”
And minimum order amount is “100€”
When he/she visits the “home page”
Then he/she should see in the header: “You are close to getting the free shipping!”

Scenario 2: The cart is above the minimum order amount
Given that the customer cart is “150€”
And minimum order amount is “100€”
When he/she visits the “home page”
Then he/she should see in the header: “You have free shipping!”.

```
├── CustomerData
│   └── NotificationSection.php
├── etc
│   ├── adminhtml
│   │   └── system.xml
│   ├── config.xml
│   ├── frontend
│   │   ├── di.xml
│   │   └── sections.xml
│   └── module.xml
├── registration.php
└── view
    └── frontend
        ├── layout
        │   └── default.xml
        ├── templates
        │   └── notification.phtml
        └── web
            └── js
                └── free-shipping-notification.js
```

```
├── CustomerData
│   └── NotificationSection.php
├── etc
│   ├── adminhtml
│   │   └── system.xml
│   ├── config.xml
│   ├── frontend
│   │   ├── di.xml
│   │   └── sections.xml
│   └── module.xml
├── registration.php
└── view
    └── frontend
        ├── layout
        │   └── default.xml
        ├── templates
        │   └── notification.phtml
        └── web
            └── js
                └── free-shipping-notification.js
```

../FreeShippingInfo/view/frontend/templates/**notification.phtml**

```
<div data-bind="scope: 'free-shipping-notification'">
   <p data-bind="text: notification().message"></p>
</div>

<script type="text/x-magento-init">
   {
       "*": {
           "Magento_Ui/js/core/app": {
               "components": {
                   "free-shipping-notification": {
                       "component": "VB_FreeShippingInfo/js/notification"
                   }
               }
           }
       }
   }
</script>
```

../FreeShippingInfo/view/frontend/web/js/**notification.js**

```
define([
   'uiComponent',
   'Magento_Customer/js/customer-data'
], function (Component, customerData) {
   'use strict';

   return Component.extend({
     /** @inheritdoc */
     initialize: function () {
       this._super();
       this.notification = customerData.get('free-shipping-notification');
      }
   });
});
```

../FreeShippingInfo/etc/frontend/**di.xml**

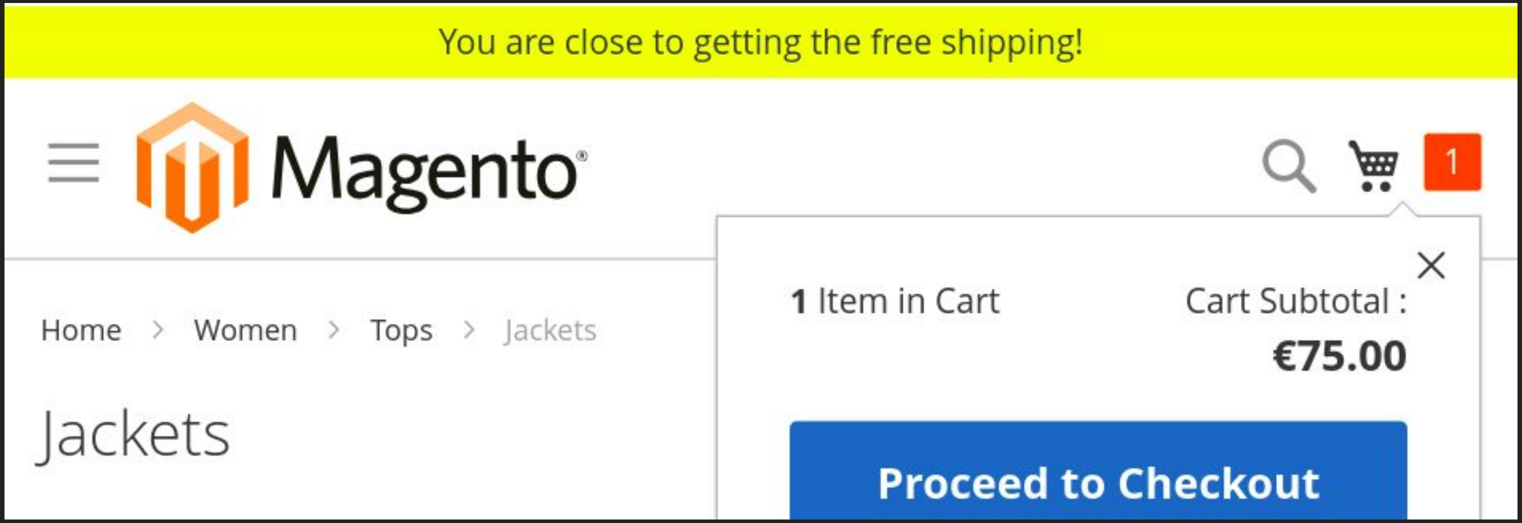
```xml
<?xml version="1.0"?>
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNames..>
   <type name="Magento\Customer\CustomerData\SectionPoolInterface">
       <arguments>
           <argument name="sectionSourceMap" xsi:type="array">
               <item name="free-shipping-notification" xsi:type="string">
                   VB\FreeShippingInfo\CustomerData\NotificationSection
               </item>
           </argument>
       </arguments>
   </type>
</config>
```

../FreeShippingInfo/CustomerData/**NotificationSection.php**

```php
use Magento\Checkout\Model\Session;
use Magento\Customer\CustomerData\SectionSourceInterface;
use Magento\Framework\App\Config\ScopeConfigInterface;
use Magento\Store\Model\ScopeInterface;

class NotificationSection implements SectionSourceInterface
{
   private $config;
   private $checkoutSession;

   public function __construct(
       ScopeConfigInterface $config,
       Session $checkoutSession
   ) {
       $this->config = $config;
       $this->checkoutSession = $checkoutSession;
   }

   public function getSectionData() {...}
}
```

../FreeShippingInfo/CustomerData/**NotificationSection.php**

```php
public function getSectionData()
{
   $message = $this->config->getValue(
       'free_shipping_info/general/message_below',
       ScopeInterface::SCOPE_STORE
   );

   $minimum = (int) $this->config->getValue(
       'free_shipping_info/general/minimum',
       ScopeInterface::SCOPE_STORE
   );

   if ($this->checkoutSession->getQuote()->getBaseGrandTotal() > $minimum) {
           $message = $this->config->getValue(
               'free_shipping_info/general/message_above',
               ScopeInterface::SCOPE_STORE
           );
   }

   return ['message' => $message];
}
```



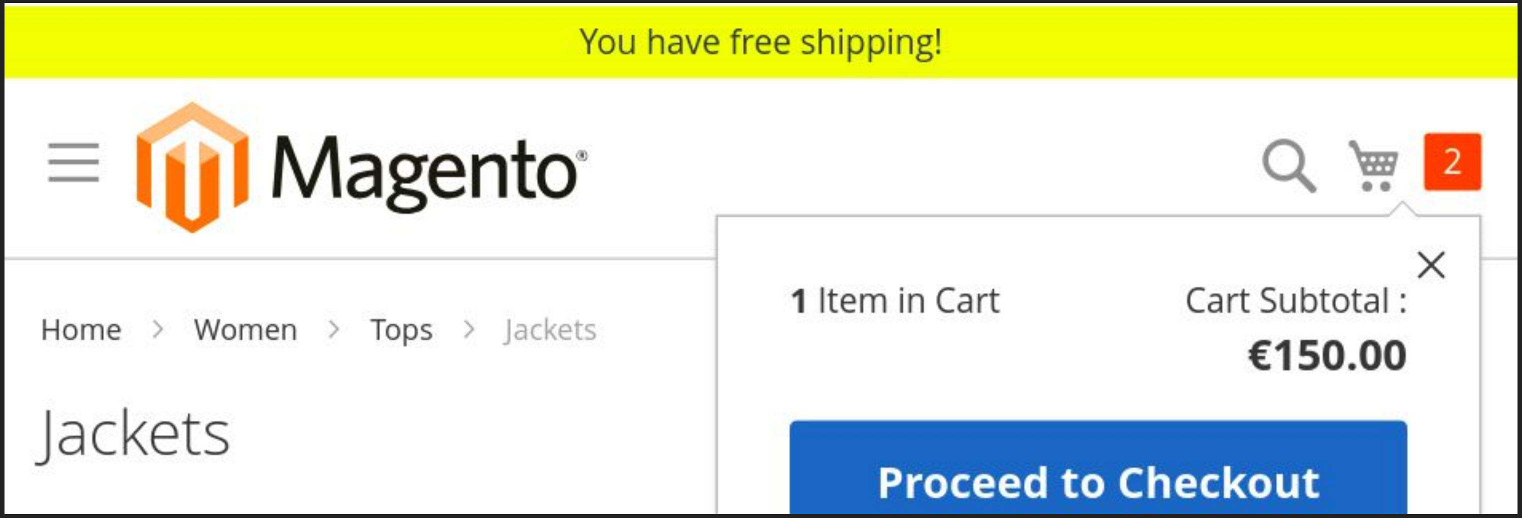
You are close to getting the free shipping!

Home > Women > Tops > Jackets

Jackets

1 Item in Cart

Cart Subtotal : €75.00



You have free shipping!

Home > Women > Tops > Jackets

Jackets

1 Item in Cart

Cart Subtotal : €150.00

Story: Show the remaining amount in the message
In order to be more convincing
As a store owner
I want to include in the message the remaining amount to get free shipping.

Scenario 1: In the message there is the pattern {remainingAmount}
Given that customer's cart is "75€"
And the minimum order amount is "100€"
And the message in config "You are {remainingAmount} away from free shipping!"
When he/she visits the "home page"
Then he/she should see in the header "You are 25€ away from free shipping"

To implement the new requirement we have to:

- Calculate the remaining amount
- Format the remaining amount
- Detect the {remainingAmount} pattern in the message
- Replace the pattern with the formatted remaining amount

../FreeShippingInfo/CustomerData/**NotificationSection.php**

```php
public function getSectionData()
{
   $message = $this->config->getValue(
       'free_shipping_info/general/message_below',
       ScopeInterface::SCOPE_STORE
   );

   $minimum = (int) $this->config->getValue(
       'free_shipping_info/general/minimum',
       ScopeInterface::SCOPE_STORE
   );

   if ($this->checkoutSession->getQuote()->getBaseGrandTotal() > $minimum) {
           $message = $this->config->getValue(
               'free_shipping_info/general/message_above',
               ScopeInterface::SCOPE_STORE
           );
   }

   return ['message' => $message];
}
```

../FreeShippingInfo/CustomerData/Test/Unit/CustomerData/**NotificationSectionTest.php**

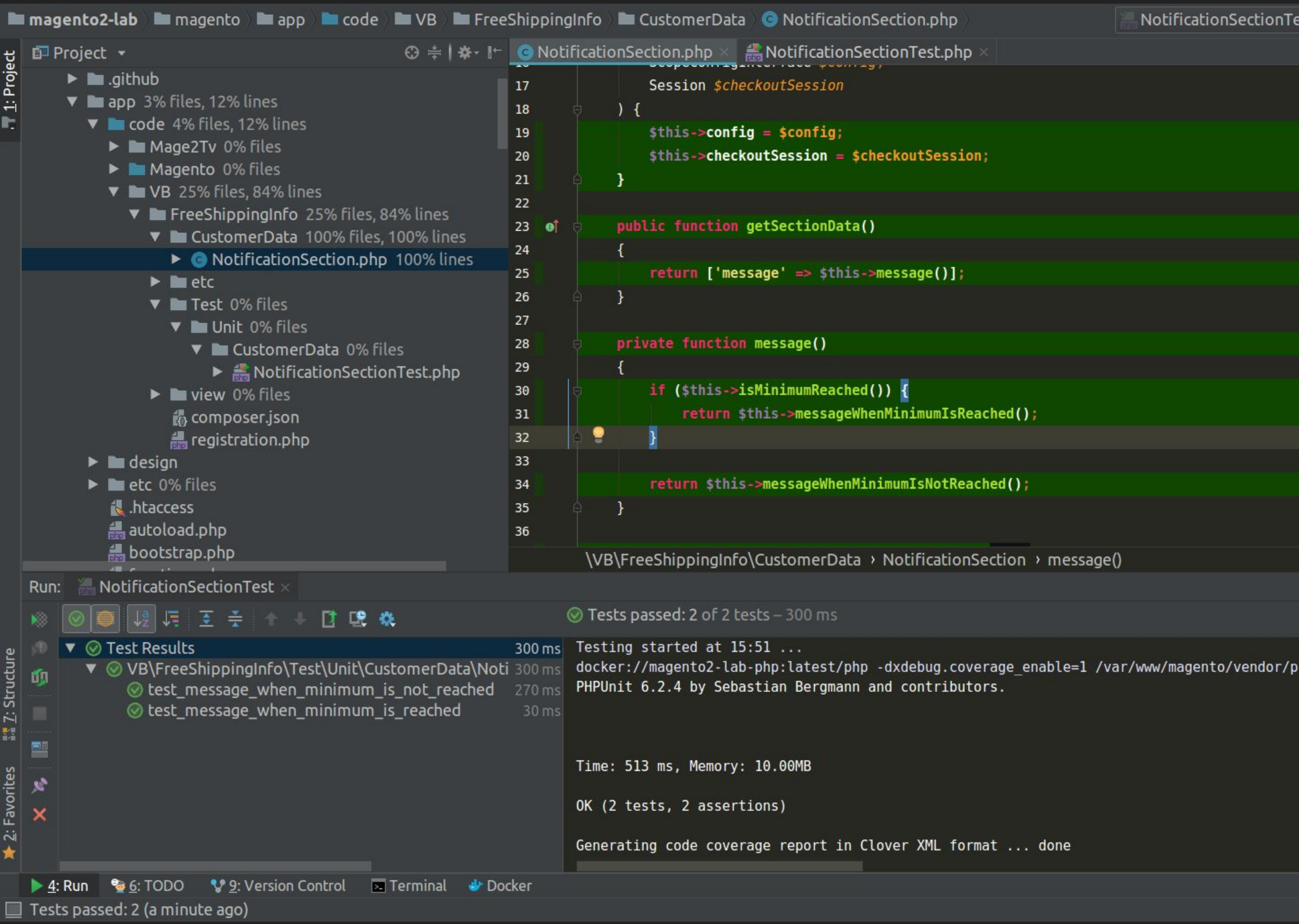
```php
class NotificationSectionTest extends TestCase
{
   private $objectManager;
   private $configStub;
   private $checkoutSessionStub;

   public function setUp()
   {
       $this->objectManager = new ObjectManager($this);
       $this->configStub = $this->createMock(ScopeConfigInterface::class);
       $this->checkoutSessionStub = $this->createMock(Session::class);
   }

   public function test_message_when_minimum_is_not_reached() {...}

   public function test_message_when_minimum_is_reached() {...}
}
```

../FreeShippingInfo/CustomerData/**NotificationSection.php**

```php
public function getSectionData()
{
   $message = $this->config->getValue(
       'free_shipping_info/general/message_below',
       ScopeInterface::SCOPE_STORE
   );

   $minimum = (int) $this->config->getValue(
       'free_shipping_info/general/minimum',
       ScopeInterface::SCOPE_STORE
   );

   if ($this->checkoutSession->getQuote()->getBaseGrandTotal() > $minimum) {
           $message = $this->config->getValue(
               'free_shipping_info/general/message_above',
               ScopeInterface::SCOPE_STORE
           );
   }

   return ['message' => $message];
}
```

../FreeShippingInfo/CustomerData/**NotificationSection.php**

```php
public function getSectionData()
{
    $message = $this->messageWhenMinimumIsReached();

    $minimum = $this->minimumOrderAmount();

    if ($this->checkoutSession->getQuote()->getBaseGrandTotal() > $minimum) {
        $message = $this->messageWhenMinimumIsNotReached();
    }

    return ['message' => $message];
}

private function messageWhenMinimumIsReached() {...}

private function minimumOrderAmount() {...}

private function messageWhenMinimumIsNotReached() {...}
```



magento2-lab › magento › app › code › VB › FreeShippingInfo › CustomerData › NotificationSection.php

NotificationSection.php | NotificationSectionTest.php

Project

```
▶ .github
▼ app  3% files, 12% lines
  ▼ code  4% files, 12% lines
    ▶ Mage2Tv  0% files
    ▶ Magento  0% files
    ▼ VB  25% files, 84% lines
      ▼ FreeShippingInfo  25% files, 84% lines
        ▼ CustomerData  100% files, 100% lines
          ▶ NotificationSection.php  100% lines
        ▶ etc
        ▼ Test  0% files
          ▼ Unit  0% files
            ▼ CustomerData  0% files
              ▶ NotificationSectionTest.php
        ▶ view  0% files
        composer.json
        registration.php
  ▶ design
  ▶ etc  0% files
  .htaccess
  autoload.php
  bootstrap.php
```

```php
        Session $checkoutSession
    ) {
        $this->config = $config;
        $this->checkoutSession = $checkoutSession;
    }

    public function getSectionData()
    {
        return ['message' => $this->message()];
    }

    private function message()
    {
        if ($this->isMinimumReached()) {
            return $this->messageWhenMinimumIsReached();
        }

        return $this->messageWhenMinimumIsNotReached();
    }
```

\VB\FreeShippingInfo\CustomerData › NotificationSection › message()

Run: NotificationSectionTest

Tests passed: 2 of 2 tests – 300 ms

```
▼ Test Results                                          300 ms
  ▼ VB\FreeShippingInfo\Test\Unit\CustomerData\Noti     300 ms
      test_message_when_minimum_is_not_reached          270 ms
      test_message_when_minimum_is_reached               30 ms
```

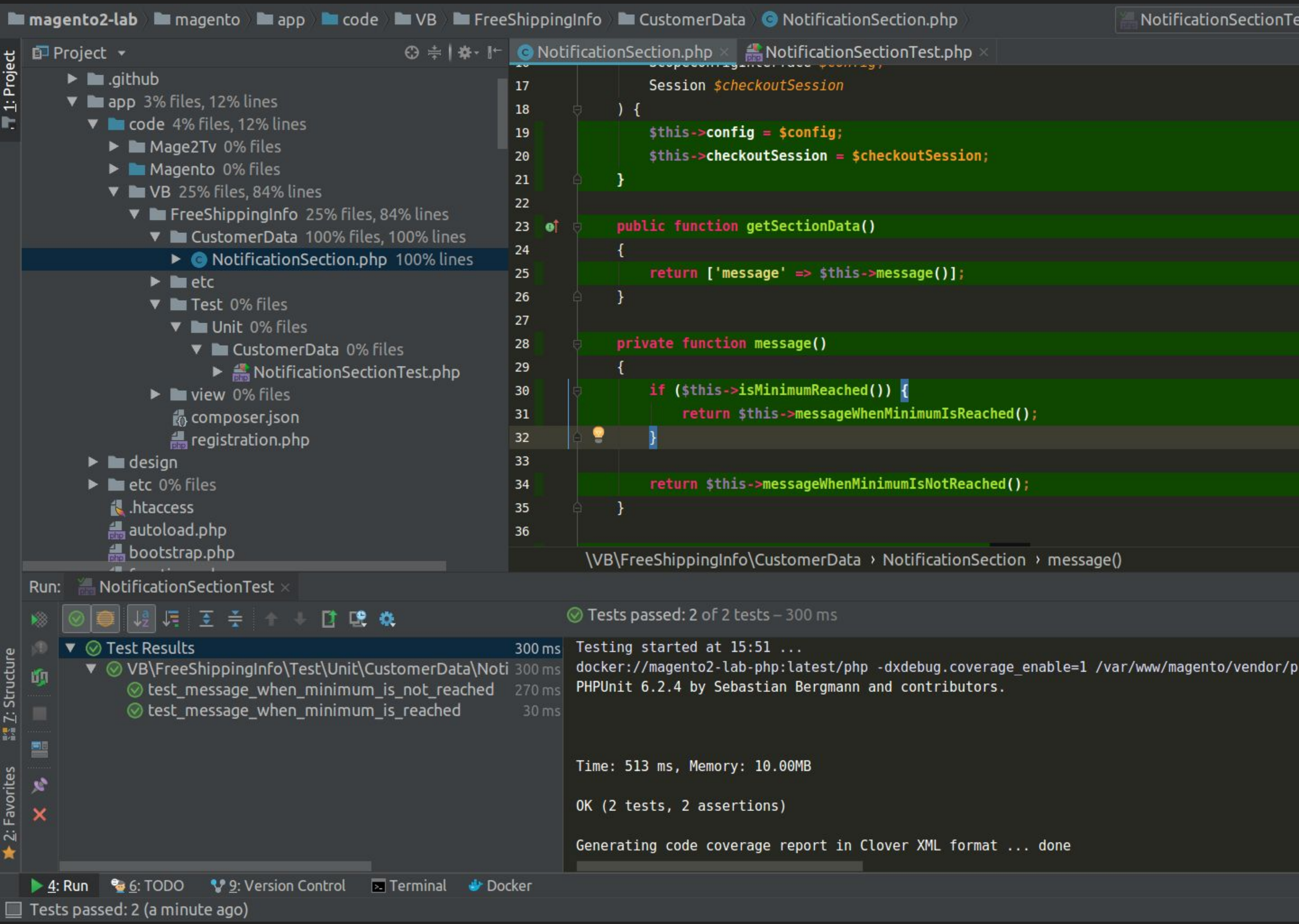
```
Testing started at 15:51 ...
docker://magento2-lab-php:latest/php -dxdebug.coverage_enable=1 /var/www/magento/vendor/p
PHPUnit 6.2.4 by Sebastian Bergmann and contributors.



Time: 513 ms, Memory: 10.00MB

OK (2 tests, 2 assertions)

Generating code coverage report in Clover XML format ... done
```

4: Run | 6: TODO | 9: Version Control | Terminal | Docker

Tests passed: 2 (a minute ago)

../FreeShippingInfo/CustomerData/**NotificationSection.php**

```php
public function getSectionData()
{
   $message = $this->messageWhenMinimumIsReached();

   if ($this->isMinimumReached()) {
       $message = $this->messageWhenMinimumIsNotReached();
   }

   return ['message' => $message];
}

private function messageWhenMinimumIsReached() {...}

private function messageWhenMinimumIsNotReached() {...}

private function isMinimumReached() {...}

private function minimumOrderAmount() {...}
```



magento2-lab › magento › app › code › VB › FreeShippingInfo › CustomerData › NotificationSection.php

NotificationSection.php | NotificationSectionTest.php

```
.github
app 3% files, 12% lines
  code 4% files, 12% lines
    Mage2Tv 0% files
    Magento 0% files
    VB 25% files, 84% lines
      FreeShippingInfo 25% files, 84% lines
        CustomerData 100% files, 100% lines
          NotificationSection.php 100% lines
        etc
        Test 0% files
          Unit 0% files
            CustomerData 0% files
              NotificationSectionTest.php
        view 0% files
        composer.json
        registration.php
  design
  etc 0% files
  .htaccess
  autoload.php
  bootstrap.php
```

```php
        Session $checkoutSession
    ) {
        $this->config = $config;
        $this->checkoutSession = $checkoutSession;
    }

    public function getSectionData()
    {
        return ['message' => $this->message()];
    }

    private function message()
    {
        if ($this->isMinimumReached()) {
            return $this->messageWhenMinimumIsReached();
        }

        return $this->messageWhenMinimumIsNotReached();
    }
```

\VB\FreeShippingInfo\CustomerData › NotificationSection › message()

Run: NotificationSectionTest

Tests passed: 2 of 2 tests – 300 ms

```
Test Results                                          300 ms
  VB\FreeShippingInfo\Test\Unit\CustomerData\Noti      300 ms
    test_message_when_minimum_is_not_reached           270 ms
    test_message_when_minimum_is_reached                30 ms
```

```
Testing started at 15:51 ...
docker://magento2-lab-php:latest/php -dxdebug.coverage_enable=1 /var/www/magento/vendor/p
PHPUnit 6.2.4 by Sebastian Bergmann and contributors.



Time: 513 ms, Memory: 10.00MB

OK (2 tests, 2 assertions)

Generating code coverage report in Clover XML format ... done
```

4: Run | 6: TODO | 9: Version Control | Terminal | Docker

Tests passed: 2 (a minute ago)

../FreeShippingInfo/CustomerData/**NotificationSection.php**

```php
public function getSectionData()
{
   return ['message' => $this->message()];
}

private function message()
{
   if ($this->isMinimumReached()) {
       return $this->messageWhenMinimumIsReached();
   }

   return $this->messageWhenMinimumIsNotReached();
}

private function isMinimumReached() {...}

private function minimumOrderAmount() {...}

private function messageWhenMinimumIsReached() {...}

private function messageWhenMinimumIsNotReached() {...}
```

../FreeShippingInfo/CustomerData/Test/Unit/CustomerData/**NotificationSectionTest.php**

```php
class NotificationSectionTest extends TestCase
{
   private $objectManager;
   private $configStub;
   private $checkoutSessionStub;

   public function setUp()
   {
       $this->objectManager = new ObjectManager($this);
       $this->configStub = $this->createMock(ScopeConfigInterface::class);
       $this->checkoutSessionStub = $this->createMock(Session::class);
   }

   public function test_message_when_minimum_is_not_reached() {...}

   public function test_message_when_minimum_is_reached() {...}

   public function test_parse_remaining_amount_in_message() {...}
}
```

Recap

The one desires to be to express The one
to be to make The one light
non luminous The one light luminous
Eternity is of two Brothers

Thank you!