Refactoring
and other small animals

Marco Tabini
@mtabini

CODEWORKS 2011

What we'll cover

- **What** is refactoring
- **Why** refactoring is important
- **When** to refactor
- **How** to refactor safely

Show of Hands

Do you use refactoring?

“Refactoring is the process of changing a software system in such a way that it does not alter the external behaviour of the code yet improves its internal structure.”

Martin Fowler

“Refactoring is the process of **changing** a software system in such a way that it does not alter the external behaviour of the code yet improves its internal structure.”

"Refactoring is the process of **changing** a software system in such a way that it does **not alter the external behaviour** of the code yet improves its internal structure."

“Refactoring is the process of **changing** a software system in such a way that it does **not alter the external behaviour** of the code yet **improves** its internal structure.”

Change

Consistency

Improvement

Good reasons to refactor

Good reasons to refactor

Code smells
Functionality smells
Too much code!
Futureproofing

Good reasons to refactor

Code smells

Functionality smells

Too much code!

Futureproofing

Good reasons to refactor

Code smells
Functionality smells
Too much code!
Futureproofing

Good reasons to refactor

Code smells
Functionality smells
Too much code!
Futureproofing

Good reasons to refactor

Code smells

Functionality smells

Extensibility

Futureproofing

Good reasons to refactor

Code smells

Functionality smells

Extensibility

Futureproofing

Good reasons to refactor

Code smells

Functionality smells

Extensibility

Maintainability

How do you refactor?

Well…

Before you refactor

Test!

So, about those refactoring techniques…

So, about those refactoring techniques…

Abstract
Break (apart)
Rename

So, about those refactoring techniques…

Abstract

Break (apart)

Rename

So, about those refactoring techniques…

Abstract
Break (apart)
Rename

So, about those refactoring techniques…

Abstract
Break (apart)
Rename

Some Examples...

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```
<?php

class Codeworks {
    function doSomething() {
        ...
    }
}
```

Some Examples...

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```php
<?php

class Codeworks {
    function loadDataFromServer() {
        ...
    }
}
```

Some Examples…

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```php
<?php

class AClass {
    public $x;
}
```

Some Examples...

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```
<?php

class AClass {
    protected $_x;

    public function __get($name) {
        if ($name == 'x') {
            ...
        }
    }

    public function __set($name, $value) {
        if ($name == 'x') {
            ...
        }
    }
}
```

Some Examples…

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```
<?php

    class Cal {
        function isOnTime() {
            return 'Always, or die trying';
        }
    }

    class Marco {
        function isOnTime() {
            return 'Sometimes';
        }
    }

    class Arbi {
        function isOnTime() {
            throw new Exception("Where's Arbi?");
        }
    }
```

Some Examples...

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```
<?php

    interface BlueParabola {
        function isOnTime();
    }

    class Cal implements BlueParabola {
        function isOnTime() {
            return 'Always, or die trying';
        }
    }

    class Marco implements BlueParabola {
        function isOnTime() {
            return 'Sometimes';
        }
    }

    class Arbi implements BlueParabola {
        function isOnTime() {
            throw new Exception("Where's Arbi?");
        }
    }
```

Some Examples...

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```php
<?php

    class Computer {
        const WINDOWS = 1;
        const MAC = 2;
        const LINUX = 3;

        public $_type;

        function getCost() {
            switch ($this->type) {
                case Computer::WINDOWS:
                    return 'medium';

                case Computer::MAC:
                    return 'high';

                case Computer::LINUX:
                    return 'low';
            }
        }
    }
```

Some Examples...

- Renaming
- Encapsulation
- Type generalization
- State/Strategy type replacement

```php
<?php
    interface ComputerType {
        function getCost(); function getType();
    }

    class Windows implements ComputerType {
        function getCost() { return 'medium'; }

        function getType() { return Computer::WINDOWS; }
    }

    class Computer { ...
        protected $_type;

        function __set($name, $value) {
            if ($name == 'type') {
                switch ($value) {
                    case Computer::WINDOWS:
                        $this->_type = new Windows;
                }
            }
        }

        function __get($name) {
            if ($name == 'type') {
                return $this->_type->getType();
            }
        }

        function getCost() {
            return $this->_type->getCost();
        }
    }
```

Change

Consistency

Improvement

“Refuctoring”

“Beware the
Ides of March”

– Bill Shakespeare

“Beware the IDEs”

– Me

CW

CODEWORKS 2011

@mtabini