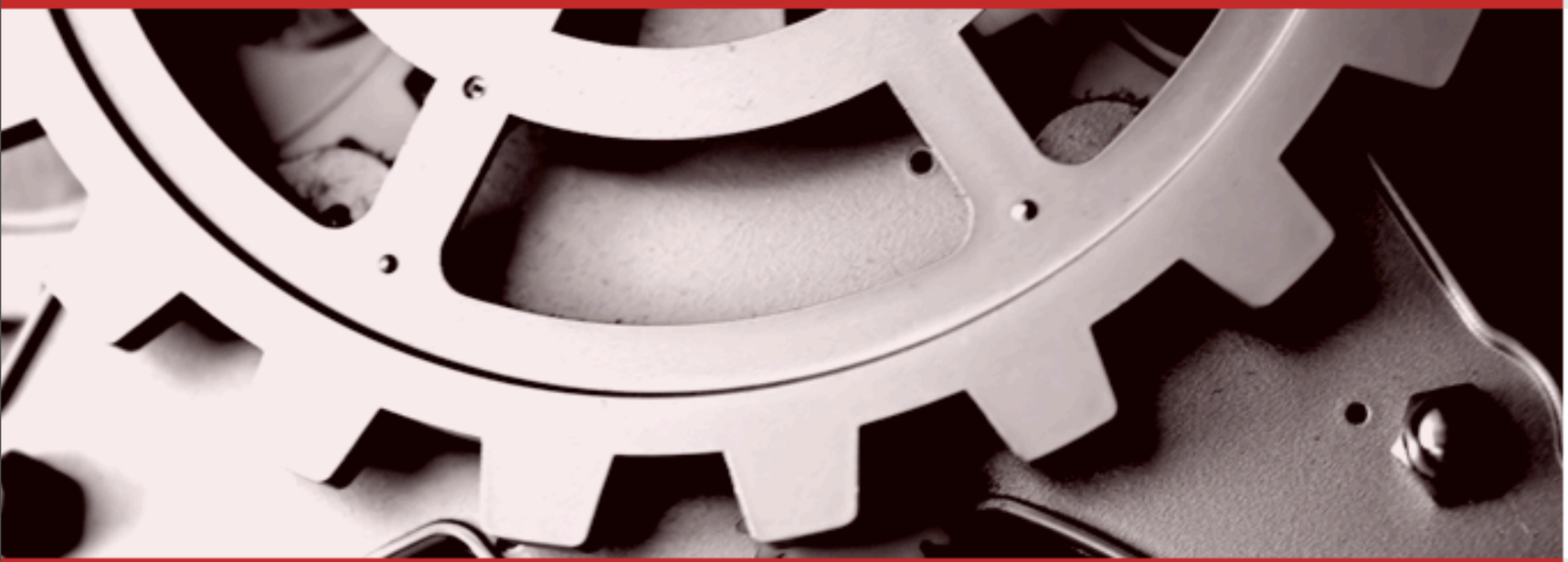


ibuildings [i]
THE PHP PROFESSIONALS

The Power of Refactoring

Stefan Koopmanschap

4developers, Krakow, March 7 2009

What will we do?

ibuildings
THE PHP PROFESSIONALS

- What is refactoring?
- Why refactoring?
- Requirements
- HOWTO

ibuildings
THE PHP PROFESSIONALS

Stefan Koopmanschap

Who is he?

About me

ibuildings
THE PHP PROFESSIONALS

- Professional Services Consultant at Ibuildings
- 10 years of PHP experience
- auto didact
- married, 2 kids, 6 cats, a bunch of fish
- Apple lover
- music addict
- symfony advocate

ibuildings
THE PHP PROFESSIONALS

Refactoring

What is it?

ibuildings
THE PHP PROFESSIONALS

Refactoring is a disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behavior. Its heart is a series of small behavior preserving transformations. Each transformation (called a 'refactoring') does little, but a sequence of transformations can produce a significant restructuring. Since each refactoring is small, it's less likely to go wrong. The system is also kept fully working after each small refactoring, reducing the chances that a system can get seriously broken during the restructuring.

(Martin Fowler - Refactoring.com)

Code refactoring is the process of changing a computer program's code to make it amenable to change, improve its readability, or simplify its structure, while preserving its existing functionality.

(Wikipedia)

What

ibuildings
THE PHP PROFESSIONALS

Code refactoring is the process of changing a computer program's internal structure without modifying its external behavior or existing functionality.

Why

ibuildings
THE PHP PROFESSIONALS

This is usually done to improve code readability, simplify code structure, change code to adhere to a given programming paradigm, improve maintainability, or improve extensibility.

Improve readability

ibuildings
THE PHP PROFESSIONALS

Improve maintainability

ibuildings
THE PHP PROFESSIONALS

Extend/change/implement

ibuildings
THE PHP PROFESSIONALS

Is it the holy grail?

ibuildings
THE PHP PROFESSIONALS

Rewrite

ibuildings
THE PHP PROFESSIONALS

A rewrite in computer programming is the act or result of re-implementing a large portion of existing functionality without re-use of its source code. When the rewrite is not using existing code at all, it is common to speak of a rewrite from scratch. When instead only parts are re-engineered, which have otherwise become complicated to handle or extend, then it is more precise to speak of code refactoring.

(Wikipedia)

Refactor vs Rewrite

ibuildings
THE PHP PROFESSIONALS

Refactor vs Rewrite

ibuildings
THE PHP PROFESSIONALS

- Refactor whenever it is possible

Refactor vs Rewrite

ibuildings
THE PHP PROFESSIONALS

- Refactor whenever it is possible
- Big change? Try to cut it into smaller units

Refactor vs Rewrite

ibuildings
THE PHP PROFESSIONALS

- Refactor whenever it is possible
- Big change? Try to cut it into smaller units
- Still have to break the API? Go ahead, and rewrite!

Refactor vs Rewrite

ibuildings
THE PHP PROFESSIONALS

- Refactor whenever it is possible
- Big change? Try to cut it into smaller units
- Still have to break the API? Go ahead, and rewrite!
- Don’t take the decision in 5 minutes

Refactor vs Rewrite

ibuildings
THE PHP PROFESSIONALS

Refactoring takes less effort

- Nice for you
- Better for the quality of your work

Breaking the API is more work

- Ensure all calling code is altered as well
- Update your unit tests
- Update your documentation

ibuildings
THE PHP PROFESSIONALS

Requirements

for successful refactoring

Requirements

ibuildings
THE PHP PROFESSIONALS

- Codebase knowledge
- Structured API
- Unit testing

Codebase knowledge

ibuildings
THE PHP PROFESSIONALS

- Know what to change quickly
- Be aware of your changes
- Be aware of impact of your changes on other components

Structured API

ibuildings
THE PHP PROFESSIONALS

- Refactoring DOES NOT REPLACE design!
- You need a good API that allows internal changes
- Additions to existing API's should not change existing API
- Keep small "units"

Unit testing

ibuildings
THE PHP PROFESSIONALS

```
require File.dirname(__FILE__) + "/../../lib/internationaliser"

class StringInternationalisationTest < Test::Unit::TestCase
  def test_can_translate_sausage_to_german
    INTERNATIONAL['sausage'] = 'würst'
    assert_equal('würst', _('sausage'))
  end
end
```

Unit testing

ibuildings
THE PHP PROFESSIONALS

- Unit testing is your first point of QA
- They safeguard you from unexpected side effects
- Test - Change - Test

ibuildings
THE PHP PROFESSIONALS

An example

user authentication

A test-driven approach

ibuildings
THE PHP PROFESSIONALS

UserTest.php

```php
class UserTest extends PHPUnit_Framework_TestCase
{
    public function testCorrectUserCredentials()
    {
      $this->assertTrue(User::authenticate('test', 'test'));
    }

    public function testIncorrectUsername()
    {
      $this->assertFalse(User::authenticate('wrong', 'test'));
    }

    public function testIncorrectPassword()
    {
      $this->assertFalse(User::authenticate('test', 'wrong'));
    }
}
```

Line: 1 Column: 1 PHP Tab Size: 2 —

My first implementation

ibuildings
THE PHP PROFESSIONALS

user.class.php

```php
class User {

  /**
   * authenticate a user
   *
   * @param string $username
   * @param string $password
   * @return boolean
   * @todo actually implement this
   */
  public static function authenticate($username, $password)
  {
    if ('test' == $username && 'test' == $password)
    {
      return true;
    }
    else
    {
      return false;
    }
  }
}
```

Line: 21 Column: 20 PHP Tab Size: 2 authenticate

My first implementation

ibuildings
THE PHP PROFESSIONALS

user.class.php

```
class User {

  /**
   * authenticate a user
   *
   *
   *
   *
   *
   *
  pu
  {
    return false;
  }
 }
}
```

Session

```
stefan-koopmanschaps-imac:code stefan$ phpunit UserTest.php
PHPUnit 3.3.4 by Sebastian Bergmann.

...

Time: 0 seconds

OK (3 tests, 3 assertions)
stefan-koopmanschaps-imac:code stefan$
```

Line: 21 Column: 20 PHP Tab Size: 2 authenticate

A more dynamic version

ibuildings
THE PHP PROFESSIONALS

user2.class.php

```php
class User {

  /**
   * authenticate a user
   *
   * @param string $username
   * @param string $password
   * @return boolean
   * @todo actually implement this
   */
  public static function authenticate($username, $password)
  {
    $sql = "SELECT `id` FROM `user`
            WHERE `username`='" . mysql_real_escape_string($username) . "'
            AND `password`='" . md5($password) . "'";
    $res = mysql_query($sql);
    if (mysql_num_rows($res) == 1)
    {
      return true;
    }
    else
    {
      return false;
    }
  }
}
```

Line: 28 Column: 2 PHP Tab Size: 2 authenticate

A more dynamic version

ibuildings
THE PHP PROFESSIONALS

user2.class.php

```php
class User {

  /**
   * authenticate a user
   *
   * @param string $username
   * @p
   * @r
   * @t
   */
  publi
  {
    $sq

    $re
    if
    {
      r
    }
    else
    {
      return false;
    }
  }
}
```

Session

```
stefan-koopmanschaps-imac:code stefan$ phpunit UserTest.php
PHPUnit 3.3.4 by Sebastian Bergmann.

...

Time: 0 seconds

OK (3 tests, 3 assertions)
stefan-koopmanschaps-imac:code stefan$
```

Line: 28 Column: 2 PHP Tab Size: 2 authenticate

ibuildings
THE PHP PROFESSIONALS

Tips and tricks

Some tips and tricks

ibuildings
THE PHP PROFESSIONALS

- Write unit tests before you start refactoring
- Code documentation (phpDoc) is your friend
- Don’t trust your IDE’s refactoring options

ibuildings
THE PHP PROFESSIONALS

Will you refactor now?

Looking back

ibuildings
THE PHP PROFESSIONALS

- What is refactoring?
- Why do we want to refactor?
- Is it the holy grail?
- Refactoring vs Rewriting
- Requirements
- How to do it
- Tips and tricks

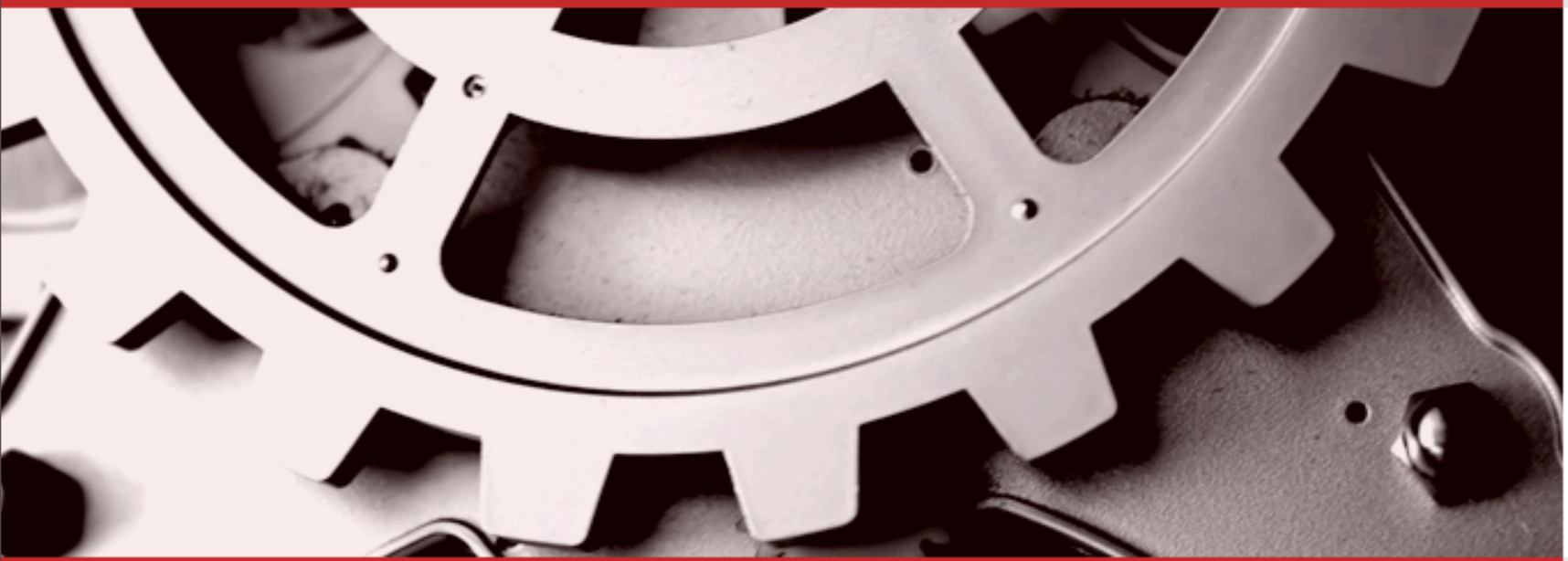
ibuildings [i]
THE PHP PROFESSIONALS

Creative Commons Rocks!

Readability:
http://www.flickr.com/photos/vipulmathur/471634239/ (by-nc-sa/2.0)
Maintainability:
http://www.flickr.com/photos/comicbase/2117680144/ (by-nc-sa/2.0)
Extend:
http://www.flickr.com/photos/auntiep/221951988/ (by-nc-sa/2.0)
Holy Grail:
http://www.flickr.com/photos/joebeone/153921142/ (by/2.0)
Würst test ever:
http://www.flickr.com/photos/pingles/1341229619/ (by-nc-nd/2.0)

ibuildings
THE PHP PROFESSIONALS

Questions ?



ibuildings [i]
THE PHP PROFESSIONALS

Thank you!

Contact details:

Stefan Koopmanschap
http://www.leftontheweb.com/
http://www.ibuildings.com/
http://www.twitter.com/skoop
http://www.slideshare.net/skoop