



# **Code Smell**

**ตัวชี้วัดปัญหา  
แบบลึกสุดใจของระบบ**



# **Code Smell**

**ช่วยบอกว่า  
เมื่อใดควรจะแก้ไข code**



# **Code Smell**

## **ช่วยบอกว่า เมื่อใดควรจะ Refactoring**

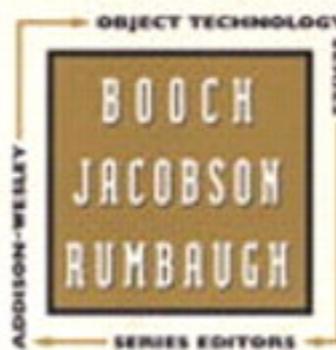
# REFACTORING

IMPROVING THE DESIGN  
OF EXISTING CODE

MARTIN FOWLER

With Contributions by Kent Beck, John Brant,  
William Opdyke, and Don Roberts

Foreword by Erich Gamma  
Object Technology International Inc.



SPRINT3R

Siam Chamnankit Co., Ltd., Odd-e (Thailand) Co., Ltd. and Alliance

- Add Parameter
- Change Bidirectional Association to Unidirectional
- Change Reference to Value
- Change Unidirectional Association to Bidirectional
- Change Value to Reference
- Collapse Hierarchy
- Consolidate Conditional Expression
- Consolidate Duplicate Conditional Fragments
- Decompose Conditional
- Duplicate Observed Data
- Dynamic Method Definition
- Eagerly Initialized Attribute
- Encapsulate Collection
- Encapsulate Downcast
- Encapsulate Field
- Extract Class
- Extract Interface
- Pull Up Constructor
- Pull Up Field
- Pull Up Method
- Push Down Field
- Push Down Method
- Recompose Conditionals
- Remove Assignment Parameters
- Remove Control Flag
- Remove Middle Man
- Remove Named Parameters
- Remove Parameter
- Remove Setting Method
- Remove Unused Parameters
- Rename Method
- Replace Abstract Super Module
- Replace Array with Object
- Replace Conditional with Polymorphism
- Inline Class
- Inline Method
- Inline Module
- Inline Temp
- Introduce Assertion
- Introduce Class Annotation
- Introduce Expression Builder
- Introduce Foreign Method
- Introduce Gateway
- Introduce Local Extension
- Introduce Named Parameter
- Introduce Null Object
- Introduce Parameter Object
- Isolate Dynamic Receptor
- Lazily Initialized Attribute
- Move Eval from Runtime to Parse Time
- Move Field
- Move Method
- Parameterize Method
- Replace Exception with Test
- Replace Hash with Object
- Replace Inheritance with Delegation
- Replace Loop with Collection Closure Method
- Replace Magic Number with Symbolic Constant
- Replace Method with Method Object
- Replace Nested Conditional with Guard Clauses
- Replace Parameter with Explicit Methods
- Replace Parameter with Method
- Replace Record with Data Class
- Replace Subclass with Fields
- Replace Temp with Chain
- Replace Temp with Query
- Replace Type Code with Class
- Replace Type Code with Module Extension



<http://refactoring.com/catalog/>

# Code Smell

มือ�ู่ 21 รูปแบบ

## Feature envy

Duplicated code      Long method

Large class      Inappropriate naming  
                    Long parameter list

Lazy class      Divergent change

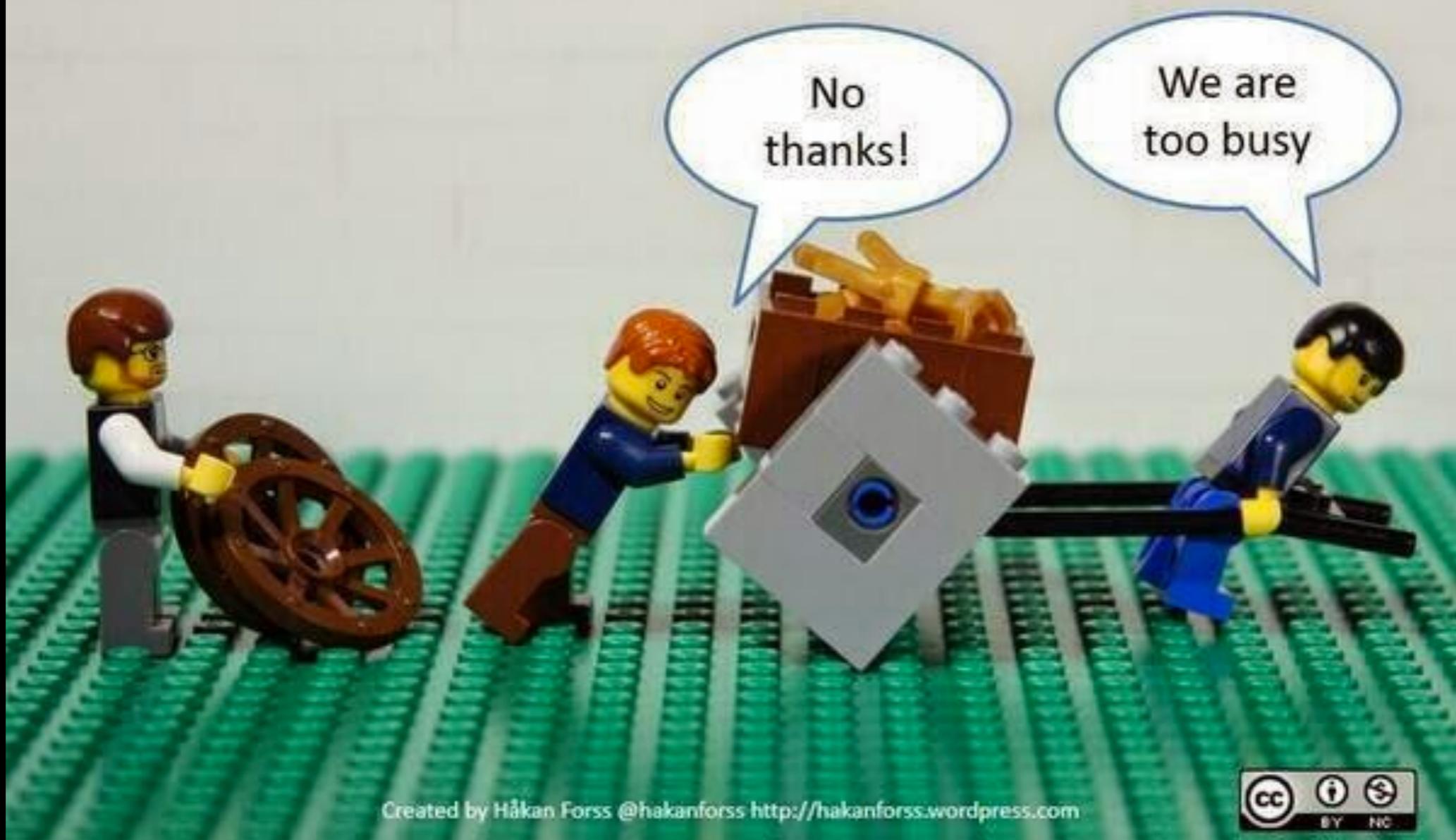
## Code Smell

Shotgun Surgery      Data class  
Dead code      Comment      Data clumps  
Primitive obsession      Parallel inheritance

Switch statement      Temporary field

etc ...

# Are you too busy to improve?



# ชื่อนั้นสำคัญไฉน ?

a b c d e f g

# ชื่อนั้นสำคัญไฉน ?

a b c d e f g

## A Boy Can Do Everything For Girl

# ทำไมชอบใช้ช้อยอ ?

err

usr

str

btn

txt

eff

frm

pnt

# ทำไมชอบใช้ชื่อที่ไม่สือ ?

run                  count  
flag  
input                output  
process              perform  
handle

# ໄມ້ມີ coding standard ?

i\_love\_you

Number\_FORMAT

idontcareanything

# ซื้อกับหน้าที่ไม่ตรงกัน ?

process()

writeFile()

isValid()

# ทำไมต้องเขียน comment ?

# ทำไมต้องเขียน comment ?

เพราะว่า code มันไม่สามารถ  
อธิบายตัวมันเองได้

# ตัวอย่างเช่น

int d;

# เพิ่ม comment ไปสี่

```
int d; //Day of Week
```

# แก้ไขชื่อคิ่งใหม่ ?

```
int dayOfWeek;
```

# ตัวอย่างเช่น

```
public void printOwning(String name, double amount){  
    printBanner();  
    //Print detail  
    System.out.println("Name : " + name);  
    System.out.println("Amount : " + amount);  
}
```

# Extract method គីឡូម៊ ?

```
public void printOwning(String name, double amount){  
    printBanner();  
    printDetail(name, amount);  
}  
  
private void printDetail(String name, double amount) {  
    System.out.println("Name : " + name);  
    System.out.println("Amount : " + amount);  
}
```

REAL PROGRAMMERS  
DON'T COMMENT THEIR CODE.

IF IT WAS HARD TO WRITE,  
IT SHOULD BE HARD TO  
UNDERSTAND.

# เขียน method ยังไง ไปทำไม?

```
public String generateHTMLHelper() {  
    StringBuilder result = new StringBuilder();  
    result.append("<");  
    result.append(name);  
    result.append(">");  
    if(!value.isEmpty()) {  
        result.append(value);  
    }  
    result.append("<");  
    result.append(name);  
    result.append("/>");  
    return result.toString();  
}
```

# เขียน comment สี

```
public String generateHTMLHelper() {  
    StringBuilder result = new StringBuilder();  
    //Generate open tag  
    result.append("<");  
    result.append(name);  
    result.append(">");  
    //Generate value  
    if(!value.isEmpty()) {  
        result.append(value);  
    }  
    //Generate end tag  
    result.append("<");  
    result.append(name);  
    result.append("/>");  
    return result.toString();  
}
```

# Extract method គីឡូម៊ែ ?

```
public String generateHTMLTagHelper() {  
    StringBuilder result = new StringBuilder();  
    generateOpenTag(result);  
    generateValue(result);  
    generateEndTag(result);  
    return result.toString();  
}
```

# ทำไมชอบเขียน if กันจัง ?

```
function register()
{
    if (empty($_POST)) {
        $msg = '';
        if ($_POST['user_name']) {
            if ($_POST['user_password_new']) {
                if ($_POST['user_password_new'] === $_POST['user_password_repeat']) {
                    if (strlen($_POST['user_password_new']) > 5) {
                        if (strlen($_POST['user_name']) < 65 && strlen($_POST['user_name']) > 1) {
                            if (preg_match('/^([a-zA-Z\d]{2,64})$/i', $_POST['user_name'])) {
                                $user = read_user($_POST['user_name']);
                                if (!isset($user['user_name'])) {
                                    if ($_POST['user_email']) {
                                        if (strlen($_POST['user_email']) < 65) {
                                            if (filter_var($_POST['user_email'], FILTER_VALIDATE_EMAIL)) {
                                                create_user();
                                                $_SESSION['msg'] = 'You are now registered so please login';
                                                header('Location: ' . $_SERVER['PHP_SELF']);
                                                exit();
                                            } else $msg = 'You must provide a valid email address';
                                        } else $msg = 'Email must be less than 64 characters';
                                    } else $msg = 'Email cannot be empty';
                                } else $msg = 'Username already exists';
                            } else $msg = 'Username must be only a-z, A-Z, 0-9';
                        } else $msg = 'Username must be between 2 and 64 characters';
                    } else $msg = 'Password must be at least 6 characters';
                } else $msg = 'Passwords do not match';
            } else $msg = 'Empty Password';
        } else $msg = 'Empty Username';
        $_SESSION['msg'] = $msg;
    }
    return register_form();
}
```



ก้าวกระโดด



## SINGLE RESPONSIBILITY PRINCIPLE

Every object should have a single responsibility, and all its services should be narrowly aligned with that responsibility.



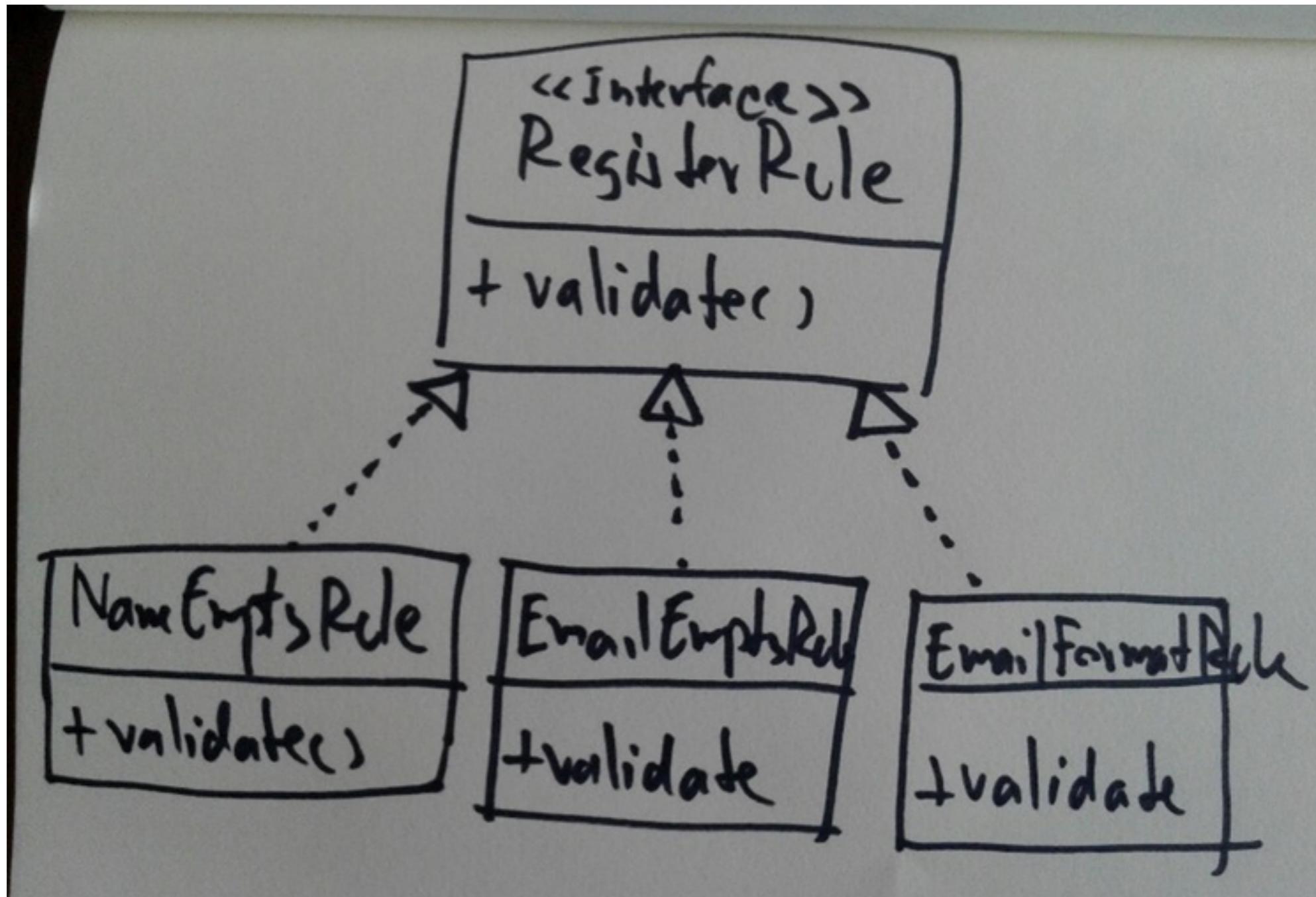
# OPEN CLOSED PRINCIPLE

Open Chest Surgery Is Not Needed When Putting On A Coat

# ตัวอย่าง

```
1 public boolean register(User user) {  
2     if (user.getName() == null || user.getName().trim().equals("")) {  
3         throw new IllegalArgumentException("Name should not empty");  
4     }  
5     if (user.getEmail() == null || user.getEmail().trim().equals("")) {  
6         throw new IllegalArgumentException("Email should not empty");  
7     }  
8  
9     if ( !user.getName().matches("[a-zA-Z]+")){  
10        throw new IllegalArgumentException("Name is wrong format");  
11    }  
12    String EMAIL_PATTERN = "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@[A-Za-z0-9-]+(\\.  
13    Pattern validEmailPattern = Pattern.compile(EMAIL_PATTERN);  
14    if( !validEmailPattern.matcher(user.getEmail()).matches() ) {  
15        throw new IllegalArgumentException("Email is wrong format");  
16    }  
17  
18    List<String> notAllowDomains = Arrays.asList("domain01.cc","domain02.cc", "domai  
19    if(notAllowDomains.contains(user.getEmail().split("@")[1])) {  
20        throw new IllegalArgumentException("Domain Email is not allow");  
21    }  
22  
23    if( user.getAge() < 20 ) {  
24        throw new IllegalArgumentException("Age should more than 20 years");  
25    }  
26  
27    return true;  
28 }
```

# แก้ไขในโลกของ OOP



# แก้ไขในโลกของ OOP

```
1 public class EmailFormatRule implements RegisterRule {  
2  
3     public void validate(User user) {  
4         String EMAIL_PATTERN = "^[_A-Za-z0-9-\\\\+]+(\\.[_A-Za-z0-9-]+  
5         Pattern validEmailPattern = Pattern.compile(EMAIL_PATTERN);  
6         if( !validEmailPattern.matcher(user.getEmail()).matches() )  
7             throw new IllegalArgumentException("Email is wrong f  
8     }  
9 }  
10  
11 }
```

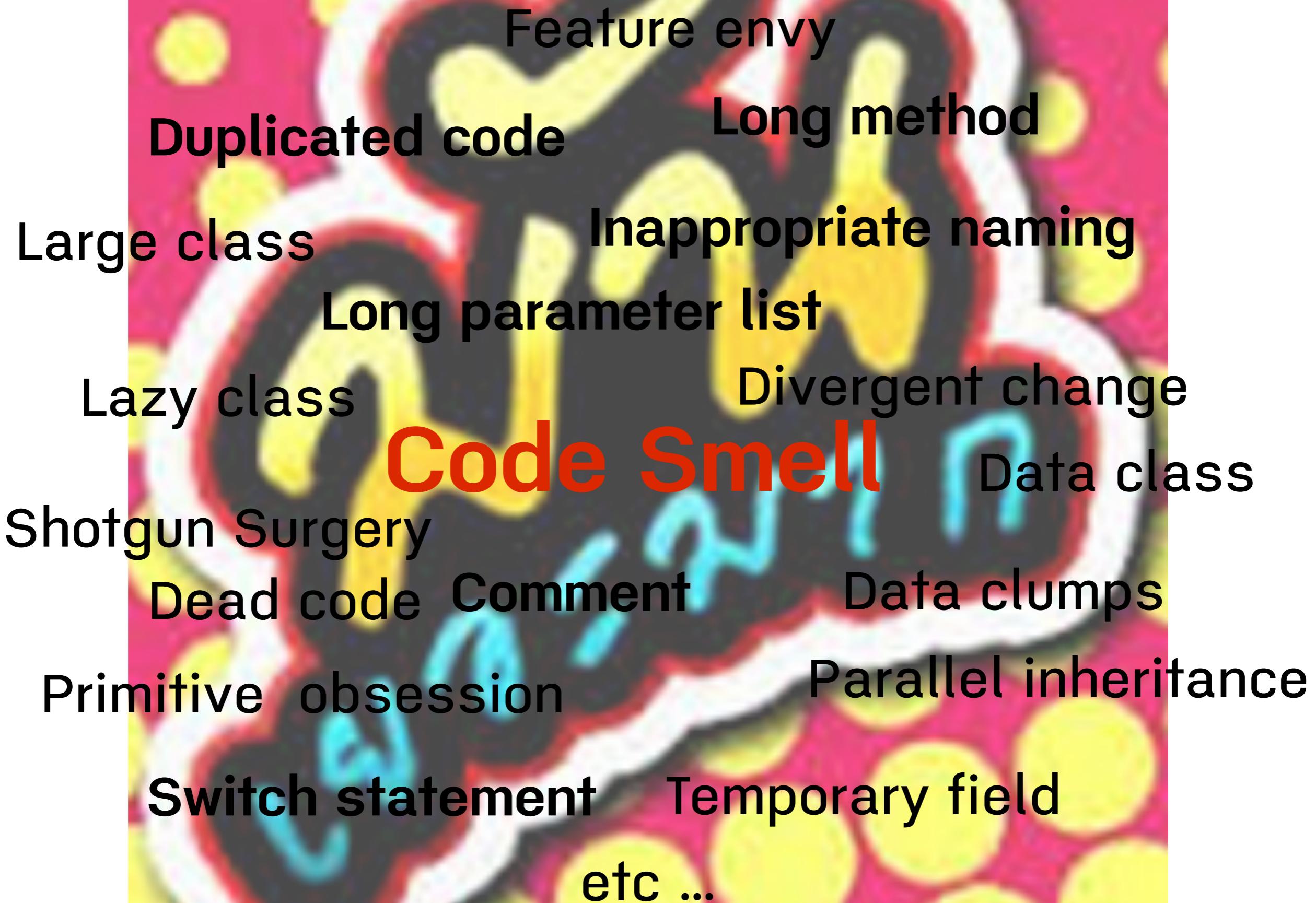
# แก้ไขในโลกของ OOP

```
1 public boolean register(User user) {  
2  
3     List<RegisterRule> registerRules = new ArrayList<RegisterRule>();  
4     registerRules.add(new NameEmptyRule());  
5     registerRules.add(new NameFormatRule());  
6     registerRules.add(new EmailEmptyRule());  
7     registerRules.add(new EmailFormatRule());  
8     registerRules.add(new EmailDomainRule());  
9     registerRules.add(new AgeAllowRule());  
10    for (RegisterRule registerRule : registerRules) {  
11        registerRule.validate(user);  
12    }  
13  
14    return true;  
15 }
```

# Duplicate code

DRY (Don't Repeat Yourself)

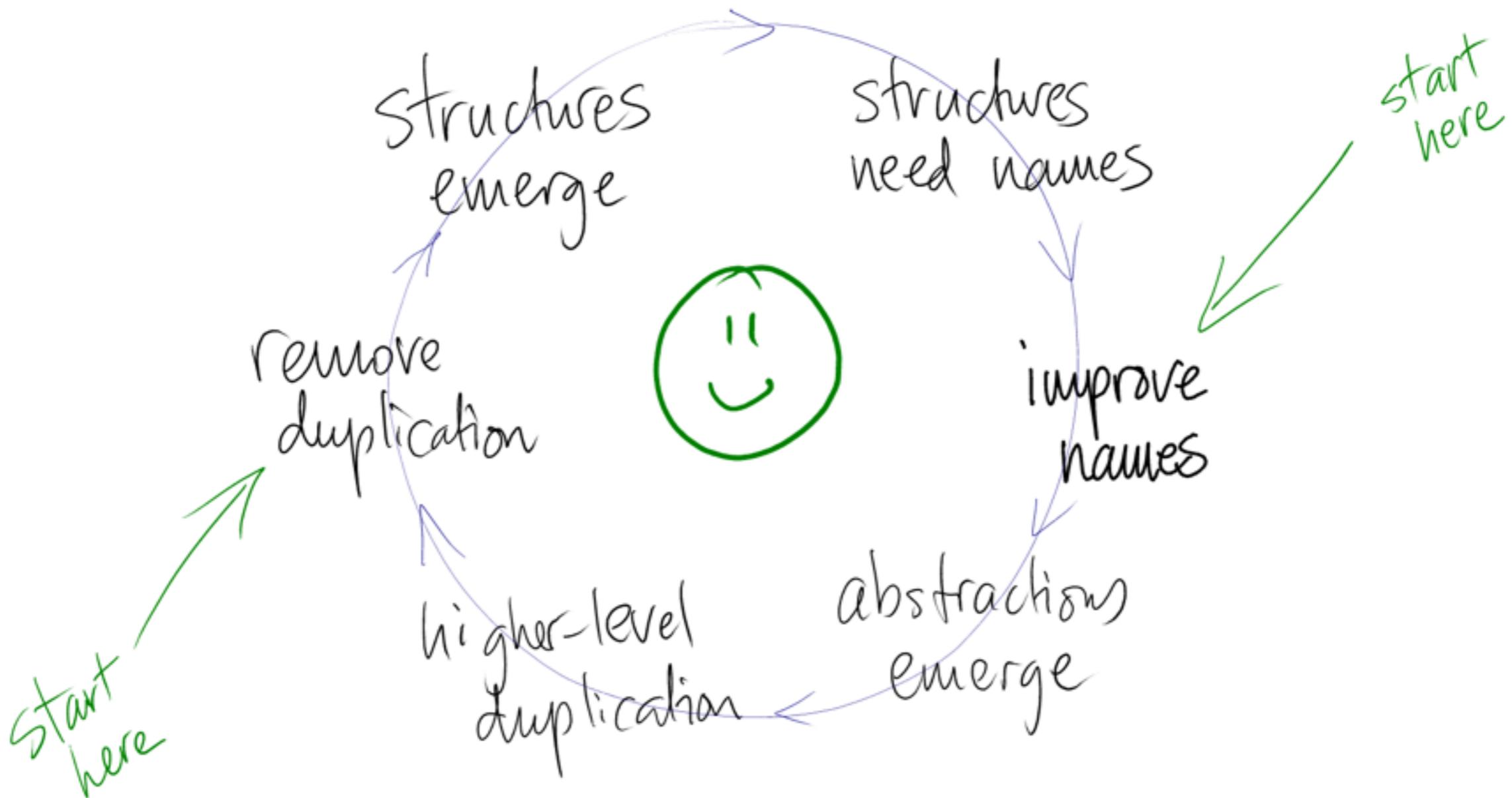
Don't copy and paste



# 4 Rules of Simple Design

1. All test pass
2. Express intent
3. No duplication
4. Keep it small

# Simple Design Dynamo



© jbrains 2013

SPRINT3R

การ coding มี 2 ขั้นตอน

Writing it  
Maintaining it

# Are you too busy to improve?



# Test Smell

# Code Smell

---

เมื่อ Unit test ซับซ้อน

# ทำไม unit test ชักช้อน ?

---

Object มันผูกติดกันมาก

# ทำไม unit test ซับซ้อน ?

---

Object มันมีหน้าที่การทำงาน yeo..

# Code Smell

---

เมื่อ Unit test ทำงานช้า

# ซื้าแล้วได้อะไร ?

---

ไม่สามารถ run บอยๆ ได้

# ซื้าแล้วได้อะไร ?

---

รอนาน

# ซ้ำแล้วได้อะไร ?

---

## Feedback loop ซ้ำมากๆ

# ทำงานช้าเพราะว่าอะໄร ?

---

ต้องทำงานกับ component ที่ช้า

# ทำงานซ้ำเพราจะว่าอะไร ?

---

มี component ที่เกี่ยวข้อง曳อะ

# ทำงานช้าเพราะว่าอะไร ?

---

## Framework มัน load เยอะ

# ความเชื่อที่น่ากลัว

Unit test มันช้าเพราะว่ามีเยอะเกินไป

# ความเชื่อที่น่ากลัว

Unit test มันช้า เพราะว่ามี酵母เกินไป  
ดังนั้นลดจำนวน test ลง

# ความเชื่อที่น่ากลัว

Unit test มันช้า เพราะว่ามี酵母เกินไป  
ดังนั้นให้มันทดสอบแบบนานๆ

# ความเชื่อที่น่ากลัว

Unit test มันช้า เพราะว่ามี酵母เกินไป  
และไม่ใช้ Test double ด้วย

ให้ใส่ใจในความเร็ว  
แต่ละ test ที่เขียน



# Code Smell

---

เมื่อผู้ก่อตั้ง framework มาไป

# การใช้งาน Framework ควรขึ้นอยู่กับ Application

ไม่ใช่ Application  
ขึ้นอยู่กับ Framework

# Feedback loop ที่ดี

Unit test ช่วยแยก code เป็นส่วนๆ

Unit test ช่วยลดการผูกมัดของ code

Unit test ช่วยลดการผูกมัด framework

# FASTER TEST



SPRINT3R

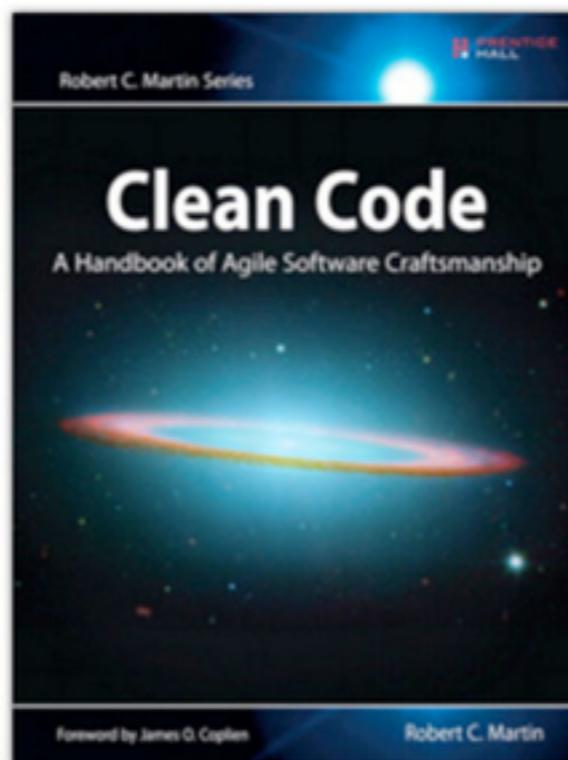
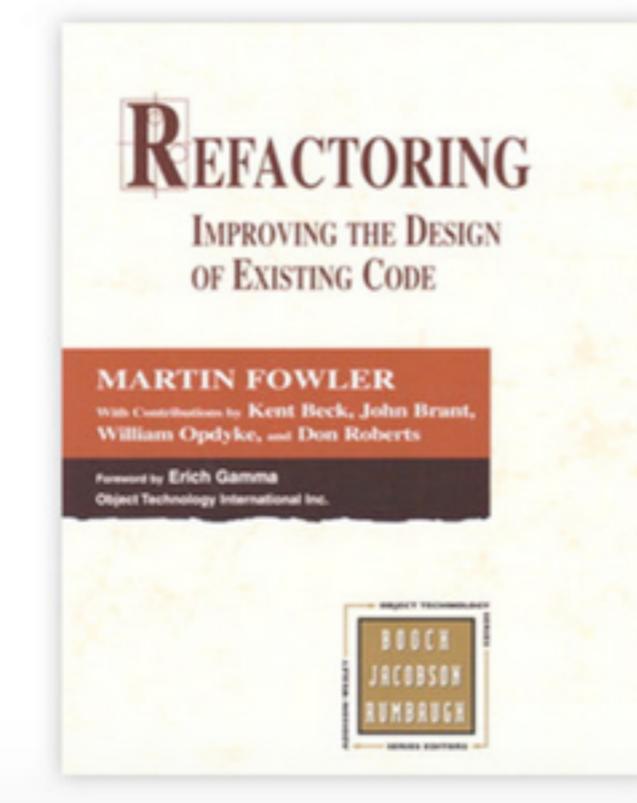
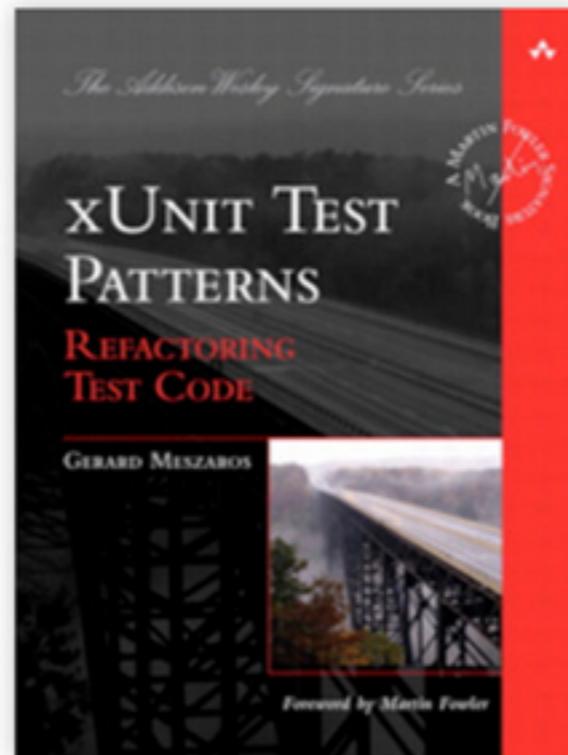
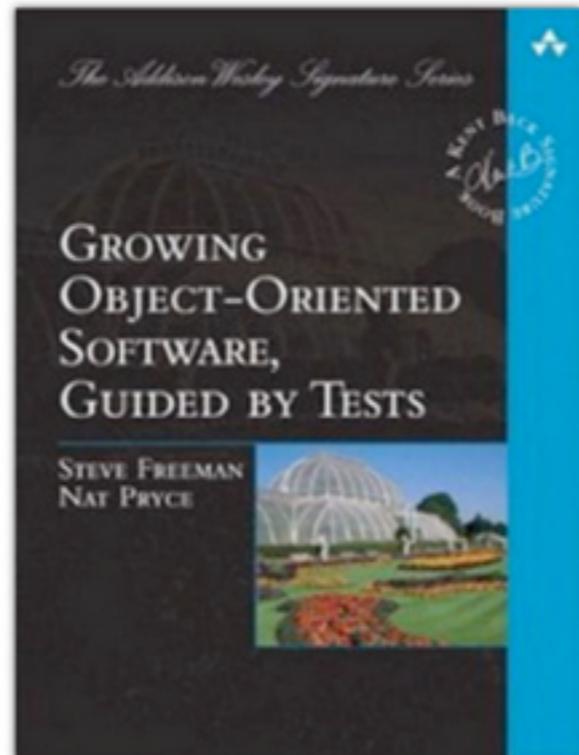
Siam Chamnankit Co., Ltd., Odd-e (Thailand) Co., Ltd. and Alliance

# แท่งอนอื่น

## คุณต้องมีวินัยในการเขียน code



# หนังสือ 4 เล่มที่นักพัฒนาต้องอ่านก่อนตาย



**Code doesn't lie.  
If you're not listening,  
you won't hear the truths it tells**

by Kent Beck

# Are you too busy to improve?

