# REFACTORING : BASIC CONCEPT

Fabian (@FabianFabz)
Bukalapak.com

Jakarta.rb July Meetup
11 July 2013

# Refactoring

- What is Refactoring?
- Why we should refactor?
- When we need to do refactoring?
- How to do refactor?

# What

- Refactoring keywords :

  changes, easier to understand, cheaper, behaviour stay the same.

# Why

- Improves Software Design

Poorly designed code, usually takes more code to do the same thing. With refactor, you can improve its design

- Easier to understand

Think of other. Don't think only for yourself.

- Find bugs easier

if you understand the code, it'll help you find bugs.

- At the end : you can program faster

# When

- When you modify function

  When you need to think to understand the code, try refactor

- When you need to fix a bug

  You might find bugs because you can't understand the code, so try refactor.

- When you do a code review

  My code may look clear to me but not to my team, so try to refactor.

# Some Bad Codes

- Duplicated Codes
- Long Methods
- Long Parameters
- Feature Envy : change one method, the other envy
- Message Chain : go to A from D via B and then C.

# Some ‘How’

- Extract Methods
- Inline Methods
- Inline Temp
- Replace Temporary Variable with Query
- Replace Temporary Variable with Chain
- Introduce Explaining Variable
- Remove Assignment to params
- Dynamic Method Definition

# Extract Methods

- Turn a part of method into its own method
- When : you have a code fragment that can be grouped.
- Example : Let’s calculate customer purchases.

# Inline Methods

- Put the method’s body into the body of its callers and remove the method
- When : method's body is as clear as its name.
- Example : give point to customer if they buy more than 5 items.

# Inline Temp

- Replace references with expression
- When : you have a temp that is assigned with expression
- Example : check if it is an expensive order or not.

# Replace Temp with Query

- Extract the expression into a method. Replace all references to the temp with the expression. The new method can then be used in other methods
- When : you use temp to hold the result of an expression
- Example : got discount if you buy more.

# Replace Temp with Chain

- Change the methods to support chaining, thus removing the need for a temp.
- When : you use temp to hold the result of an expression
- Example : find orders that are made between some dates and prices more than 500000

# Explaining Variable

- Create temporary variable that is easy to read.
- When : complicated expression
- Example : if customer has enough balance and s/he is an active customer, then do something.

# Remove Assignment from Params

- Try not to change the params. Use temp variable instead.
- When : the code assigns the param.
- Example : give discount to order that price is more than 500000

# Quotes

Any fool can write code that a computer can understand. Good programmers write code that humans can understand.

Martin Fowler

# Thank You For Your Attention

# Dynamic Method Definition

- Define methods dynamically
- When : you think it'll be consice if it is defined dynamically
- Example : change order state