Java Refactoring Cheat Sheet

The disciplined application of a series of small, behavior-preserving transformations

turley.labs()

Code Smells

Surface-level indications of potential underlying problems

Mysterious Name

A name that does not obviously describe the structure

```
String type = "default";
int weight = 50;
```

Long Method

Any method longer than 10 lines

Duplicate Code

Literally the exact same characters, similar constructs with minor internal variation, or intra-line expressions duplicated across lines

Switch Statement

The keyword switch or if/else if/else blocks

Complex Conditional

Multiple boolean expressions or nested blocks

Feature Envy

Reading and modifying another class's data while not using its own

```
// GildedRose class is envious of Item class
private void increaseQualityWhenBelow50(Item item) {
    if (item.quality < 50) { item.quality++; }
}
```

Data Class

A class with only fields, getters & setters, or a Java bean

```
public class Item {
    public String name;
    public int sellIn;
    public int quality;
}
```

Refactors

Structural transformations which preserve behavior

Refactor	IntelliJ Shortcut
Extract Variable	Ctrl + Alt + V
Extract Method	Ctrl + Alt + M
Inline Variable / Method	Ctrl + Alt + N
Extract Parameter	Ctrl + Alt + P
Move Instance Method	F6
Rename Method / Variable / Class	Shift + F6

Replace Type Code with Subclass

1. If no existing class, wrap type code or string value in a class
2. Create a getter method for the type code
3. Create a subclass for the type
4. Override the getter and return a hard coded type code
5. Repeat steps 2-4 until all type codes have a subclass
6. Remove type code field from the base class

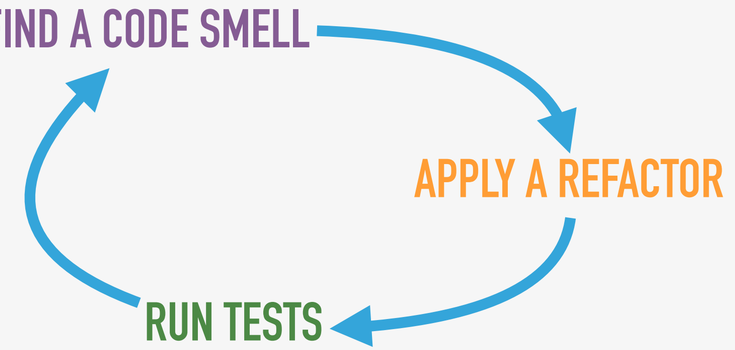
Replace Conditional with Polymorphism

1. Ensure necessary inheritance structure, possibly by using Replace Type Code with Subclass
2. Pick a subclass, override the method containing the conditional.
3. Copy and paste entire method body from base class
4. In the subclass, replace conditionals related to that type with true, replace all other type code conditionals with false
5. Simply remaining boolean expressions and unwrap if statements
6. In the base class, replace conditionals related to that type with false
7. Simply remaining boolean expressions and unwrap if statements
8. Repeat steps 2-7 until all conditionals are replaced with overridden methods

Replace Nested Conditional with Guard Clause

1. Extract check and create new if at the top of the method
2. Invert condition
3. Return early or throw exception
4. Reduce nesting of original block

Workflow



Mapping of Code Smells to Refactors

Code Smell	Refactor (Transformation)
Mysterious Name	Rename Method / Variable / Class
Long Method	Extract Method
Duplicate Code	Extract Method Extract Variable
Feature Envy	Move Instance Method
Data Class	Move Instance Method
Switch Statement	Replace Type Code with Subclass Replace Conditional with Polymorphism
Complex Conditional	Replace Nested Conditional with Guard Clause

References:
Fowler, Martin. Refactoring: Improving the Design of Existing Code
Kerievsky, Joshua. Refactoring to Patterns

© Steve Turley - www.turleylabs.com