**Code Style Rules**

**Duty Assigner**

**For Java (1.6.0\_26 or above)**

Last modification: May.11.2021

Author: Marco Backman

**Preface**

**Content**

1. Variable/Method rules

1.1. final (fixed) variable declaration.

1.2. Global (unfixed, non-final) variable.

1.3. Local variables(general).

1.5. Access specifier(modifiers) declaration.

1.6. Parameters.

1.7. Indentation, braces (Curly Braces).

1.8. Type declaration.

1.9. ‘this’ keyword.

1.10. return type.

2. Class declaration rules

2.1.

2.2.

3. Commenting rules

3.1.

3.2.

4. Encoding rules

4.1. General Language

4.2. General Encoding

4.3. Others.

5. Generics

6. Package Rule

7. importation Rule

8. Compiler/Environment version check.

9. Code page/Class file structure.

**1. Variable/Method Rules**

**1.0 general spacing**

(1) Always put spaces between identifiers, operator, separator, etc.

(2) No trailing spaces at every end of the syntax.

(3) No meaningless tab or enter gap.

**1.1. final(fixed) variable declaration.**

(1) All final variable constants are defined as following:

- Example: Always **uppercase** letter, space (and number) separated by **underscore.**

final int MARGIN\_SIZE\_1 = 10;

- Bad example 1:

final int marginSize1 = 10;

- Bad example 2:

final int margin\_size1 = 10;

(2) Final variable listing by types:

* Example: Differentiate by **Enter spacing** between different type of final variables. **No gap** between same type declaration sentence.

final int MARGIN\_SIZE\_X = 10;

final int MARGIN\_SIZE\_Y = 10;

final char LETTER\_1 = ‘a’;

final char LETTER\_2 = ‘b’;

final char LETTER\_3 = ‘c’;

- Bad example 1:

final int MARGIN\_SIZE\_X = 10;

final int MARGIN\_SIZE\_Y = 10;

final char LETTER\_1 = ‘a’;

final char LETTER\_2 = ‘b’;

final char LETTER\_3 = ‘c’;

- Bad example 2:

final int MARGIN\_SIZE\_X = 10;

final char LETTER\_1 = ‘c’;

final char LETTER\_2 = ‘c’;

final char LETTER\_3 = ‘c’;

final int MARGIN\_SIZE\_Y = 10;

**1.2. Global (unfixed, non-final) variable.**

(1) same rule applies as local variable.

**1.3. Local variables(general).**

**1.4. Generics.**

**1.5. Access Specifier (Modifiers) Declaration**.

(1) For access modifier declaration for **variables**, you **may** **label private as needed**, however, anything that may cause **identical naming from other packages must be** labeled as private to notify the developer for further efficient development.

* Example: Both classes use similarly named variable. Therefore, outer class’(upper) declaration access modifier as private.

(at MainFrame.java -> StanderFrame.java)

private JLabel rankLabel;

private JLabel nameLabel; <-

private JLabel exceptionLabel;

(at MainFrame.java -> StanderFrame.java ->

StanderScrollPane.java -> StanderListView.java ->

StanderSetPanel.java)

JLabel regNumLabel,

enlistNumLabel,

rankLabel,

nameLabel; <-

(2) Any identifier without a modifier has similar access level of private level (except allowing package access). Therefore, caution advised when creating any identifiers (variables, class, methods, etc.) that need access from other packages or that refrain to access to itself (e.g., public, protected).

(3) All **methods** except constructors must be labeled related to its required access level (public, private, protected, but no empty).

**1.6. Parameters**

(1) Method’s parameters that have similar attributes to other methods’ parameters must be placed in similar order.

- Example 1: Parameter’s orders are in order in contrast with others

public int getDay(int month, int year) {}

public int getMaxDates(int month, int year) {}

public int getDate(int date, int month, int year) {}

- Bad Example: Parameter’s orders are mixed

public int getDay(int month, int year) {}

public int getMaxDates(int year, int month) {}

public int getDate(int date, int year, int month) {}

**1.7. Indentation, (Curly Braces)**

**1.8. Type declaration.**

**1.9. ‘this’ keyword.**

**1.10. return type.**

**→** Never Return null.

**→** Always check the return type and value.

**2. Class rules**

2.1

**3. Commenting Rules**

3.1.

3.2.

**4. Encoding Rules**

4.1. General Language

4.2. General Encoding

4.3. Others.