**SOEN 6441: RISKGAME BUILD 2 DOCUMENTATION:**

**CODING STANDARDS**

**GROUP MEMBERS:**  
**40087621 An Nguyen**  
**40104630 Naga Satish**

**Basic coding rules:**

This refers to the set of rules and guidelines to be followed when coding as well as other factors like comments, declarations etc. Not only is a good coding convention best practice it also ensures the readability and manageability of the code. As a team the following coding conventions were agreed upon and used:

**Naming convention and Capitalization:**

* Use descriptive name for all variables, methods name, constants and any other if any. For example:

Une image contenant capture d’écran

Description générée automatiquement

* Class names always start with an upper case and first letter of every next word capitalized (Upper Camel Case). For example: RiskBoardModel, RiskGameModel, etc…

Une image contenant capture d’écran

Description générée automatiquement

* Variables name always start with a lower case and first letter of every next word capitalized (Lower Camel Case). Except for constant variables (final keyword) that are predefined in RiskGameConstants class, the name of those constants will be all upper cases and use underscore to separate each word. For example:

Une image contenant capture d’écran

Description générée automatiquement

* Methods name also start with a lower case and first letter of every next word capitalized. Parameters of the method are also following the lower camel case.

Une image contenant capture d’écran

Description générée automatiquement

* No use of hyphens or underscore to separate multi-word variables.

**Code Layout Conventions:**

* Code are indented with respect to its nesting level and indent a space equals to one tab key.
* The body of the while/for loop, switch statement is always indented. The same approach is applied for if statement or any other nested structure. For example:

**Une image contenant capture d’écran

Description générée automatiquement**

* Open braces (“{“) will begin at the same line of the method name
* Close braces (“}”) will start at a new line.

For example:

Une image contenant capture d’écran

Description générée automatiquement

**Javadoc commenting conventions:**

Javadoc comments will precede every class with descriptive comment about the purpose of a particular class. Include @author notation to indicate who is in charge of that class. One line for each @author notation if there are more than 1 team member working on it.

Javadoc comments will also precede every method with descriptive comment about the purpose of a particular method. The Javadoc comments should include the following:

* A description of the method
* The name and the description of each parameter (Will not present if the method does not have any parameter). One line for each parameter if there are more than 1 parameters.
* The name and the description of the return value (Will not present if the method does not return anything).
* Exception throwing if there is any. If the method has already included the catch exception, it is not necessary to include them in the Javadoc commenting.

An example:

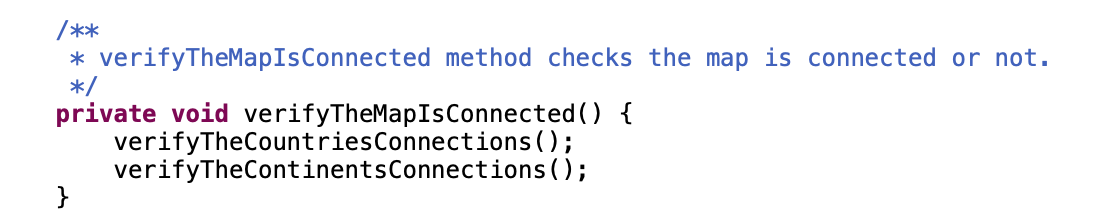
Une image contenant capture d’écran

Description générée automatiquement

Une image contenant capture d’écran

Description générée automatiquement**Spacing:**

Always put a line between two method within a class. For example:

Une image contenant capture d’écran

Description générée automatiquementAlways put a space before every “{“. For example:

**Exception Handling:**

* Une image contenant capture d’écran

  Description générée automatiquementAny exception such as IOException, NullPointerException, NumberFormatException, etc… will be handle by using try/catch block in case an exception happens. A method is written and when an exception happens, that method will be called and decide if the game needs to be stop or not. For example:

Une image contenant capture d’écran

Description générée automatiquement