

**Authors : Constantin MASSON**

**Jessica FAVIN**

**Anthony CHAFFOT**

Table of contents

[Introduction 3](#_Toc410781140)

[Technical specifications 3](#_Toc410781141)

[Agreement writing and naming 3](#_Toc410781142)

[Writing rules 3](#_Toc410781143)

[Naming rules 3](#_Toc410781144)

[Exception and particular case 3](#_Toc410781145)

[Swing attribute 3](#_Toc410781146)

[Special class for Model View Controller 4](#_Toc410781147)

[Documentation 4](#_Toc410781148)

[Class 4](#_Toc410781149)

[Function 4](#_Toc410781150)

[File 5](#_Toc410781151)

[Architecture 5](#_Toc410781152)

[Used 5](#_Toc410781153)

[Package hierarchy 5](#_Toc410781154)

[models 5](#_Toc410781155)

[views 5](#_Toc410781156)

[controllers 5](#_Toc410781157)

[tools 6](#_Toc410781158)

[img 6](#_Toc410781159)

[sounds 6](#_Toc410781160)

[Documents 6](#_Toc410781161)

[Programming tools 6](#_Toc410781162)

[GIT 6](#_Toc410781163)

[UML 6](#_Toc410781164)

[IDE 6](#_Toc410781165)

[UML Diagrams 6](#_Toc410781166)

# Introduction

This documents describe all inner specifications as attribute naming rules and technical tools used.

# Technical specifications

Programming language: Java

Display mode: Graphic

Library used : Swing

Software architecture: Model View Controller

Documentation: JavaDoc

IDE: NetBeans

Versioning: GIT

Unit test: No

# Agreement writing and naming

## Writing rules

* Write in English only.
* Try not to exceed 80 characters at the same line
* File and folder are only made by lowercase (Except java class / interface etc)
* ATTENTION: always write {} in if condition, even if it has got only one line

For instance:

If(a==0){ return true;} //Correct

If(a==0) return true; //Forbidden

## Naming rules

* Common variables: variableName
* Arguments variables: pVariableName (Always add p before variable)
* Function: functionName
* Class: ClassName
* Setters: setAttributeName
* Getters: getAttributeName
* Constants: CONSTANT\_NAME

## Exception and particular case

### Swing attribute

Add a short flag before attribute to note attribute instance type.

Example:

* JFrame: jf\_nameFrame
* JPanel: jp\_nameAttribute
* JMenu: jm\_menuName
* JLabel: jl\_nameLabel
* JButton: jb\_nameButton
* JTextField: jtf\_nameTextField

### Special class for Model View Controller

When a class has a specific function in MVC architecture, its function must be added in its name. This extra word must be added at the end.

For example, a map could have its model, view and controller, then, there are 3 classes named

|  |  |  |
| --- | --- | --- |
| **Model** | **View** | **Controller** |
| mapModel | mapView | mapController |

# Documentation

## Class

Before each class, JavaDoc must be done!

**/\*\***

**\* <h1>ClassName</h1>**

**\* <p>**

**\* public class ClassName<br/>**

**\* extends ClassExtended<br/>**

**\* implements InterfaceImplemented<br/>**

**\* </p>**

**\***

**\* <p>Description</p>**

**\***

**\* @date**

**\* @author**

**\*/**

Public class ClassName extends ClassExtended implements InterfaceImplemented{

//Code

}

## Function

If function is private, you could add simply comment /\* \*/ otherwise, create javaDoc with /\*\* \*/

**/\*\***

**\* Description function**

**\* @param nameParam description param**

**\* @param nameParam2 description param2**

**\* @return what is returned**

**\*/**

Public void functionName(){

/code

}

## File

Some documentation could be useful for files. (It's not a JavaDoc) Simply add at the top

**/\***

**\* Class: ClassName**

**\* Creation: Date**

**\* Author: Author**

**\*/**

# Architecture

## Used

MVC is used. With push model technique.

That means, when model is modified, view must be updated, which is processed by an observer like notifyObservers(Object newValue); (View implement observer interface from java and will call update (Observable o, Objcet arg);

See API for more information:

http://docs.oracle.com/javase/8/docs/api/ (Observer / Observable)

## Package hierarchy

General package: com.battleship

### models

Every class used for data. Models are totally reusable and no dependent with their view/controller

(See MVC pattern for more information).

### views

Every class used for view. It means every Swing class will be there. A view class knows its controller

### controllers

Manage the views and model. Controllers classes know their model and view.

### tools

Some tools and asset used for the program. These tools could be manager for time or sounds etc

### img

Image are stored in this package

### sounds

Sounds are stored in this package

## Documents

The documents folder, at the project root, store all data about the project. Like specifications, known bugs, updates to do and so on...

# Programming tools

## GIT

Versioning is managed with GIT.

The project is stored on github

* Download the project: https://github.com/FlexCaribou/battleship
* Clone project: git clone https://github.com/FlexCaribou/battleship.git

## UML

Dia: https://wiki.gnome.org/Apps/Dia

## IDE

NetBeans: https://fr.netbeans.org/

Sublime text: http://www.sublimetext.com/

# UML Diagrams

See UML diagrams in documents folder