UNIVERSITÉ
CÔTE D'AZUR

Université Côte d'Azur

Takenoko
https ://github.com/Mahe-Thomas/takenoko

Auteur :
Jeremy Bonsaudo
Matthias Percelay
Brandon Fontany–Legall
Thomas Mahe

4 novembre 2018

Table des matières

Chapitre 1

Présentation du projet

1.1 Sujet

Le projet consiste à réaliser en Java une version numérique du jeu Takenoko créé par Antoine Bauza. Version textuel n'étant pas destinée à être jouée par des êtres humains mais par des robots jouants de façon autonome. Dans le Takenoko nous allons endosser le rôle d'un jardinier japonais affairé à répondre au mieux aux requêtes du vénérable empereur. Pour ce faire, nous aurons, durant notre tour de jeu, deux choses à faire : Regarder quel temps il fait et effectuer deux actions parmi les possibilités suivantes : agrandir le jardin, irriguer, bouger le jardinier, bouger le panda.

1.2 Avancement du projet

Le projet en l'état est pratiquement complet, il manque en effet le dès qui pourra être implémenter ultérieurement car nous avons à l'heure actuelle privilégié l'injection de dépendance au dès.

1.3 Problématique soulevée

Plusieurs problématiques ont été soulevées durant le projet qui sont les suivantes :

1.3.1 Le bon partitionnement du code

L'organisation du code a été pour nous un assez grand problème notamment à cause de l'apparition de "God classe" [1]
comme la classe de Jeu qui concentré un trop grand nombre de fonctions. Nous avons donc fait plusieurs ré-aménagements (Voir paragraphe Conception3)

1.3.2 L'injection de dépendance

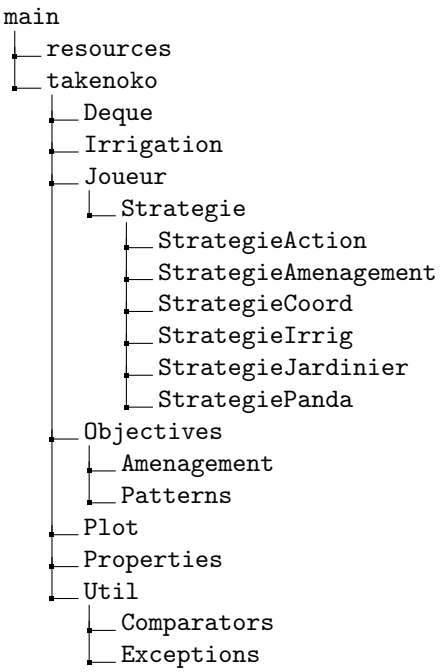
Du fait de la nouveauté de l'outils, nous avons rencontrer plusieurs problèmes lors de l'injection de dépendance. (Voir paragraphe Spring4)

1. La "God classe" est une classe comportant un très grand nombre de fonctions qui pourraient être dispatcher dans d'autre classes

Chapitre 2

Organisation du code

2.1 Packages



Après de multiples ré-aménagements, nous sommes arrivé à cette structuration des packages car ...

2.2 Interfaces

Les principales Interfaces sont celle des Stratégies, chaque stratégie que ce soit StratégieAction, StratégieAmenagement, etc... ont chacune leur Interfaces car plusieurs implémentations avec des idées différentes sont présentes. Toutes ces interfaces permettant l'implémentation d'une stratégie complexe composée d'une stratégie de chaque catégorie.

2.3 héritage

AA

Chapitre 3

Conception

Intro [1]

3.1 Patron de conception

Bla(cf. fig. 1.1)

Bla
Bla

Bla
Bla [2]

Bla(cf. ref. [**?**]).

1. Note bas de page "intro"
2. Note bas de page "bla"

Chapitre 4

Spring

Intro [1]

4.1 Notre Spring

Bla(cf. fig. 1.1)

Bla
Bla

Bla
Bla [2]

Bla(cf. ref. [**?**]).

1. Note bas de page "intro"
2. Note bas de page "bla"

Chapitre 5

Bilan

Intro / Rappel Contexte
Nous avons donc pu en tirer la problématique suivante :

Problématique du sujet

Bla
Bla

Bla

Bla
Bla

Bla
Bla