Projet Pirate

AUTEFAGE Ninon - Responsable Architecture
GUILLOT Laura - Responsable UML
SANCHEZ Emilien - Responsable Spécifications
PUJOL Robin - Responsable Développement noyau
QUANG Lucas - Responsable IHM
GLEDEL Oscar - Chef de projet

Organisation du projet

Réunions de début et de fin de projet

Communications via Discord + Drive

Temps de travail en groupe avant les séances de TP du lundi

Rétroplanning pour suivre les tâches

RETROPLANNING - PROJET PIRATES

Tâche	Catégorie	Responsable	TP1	TP1 -> TP2	TP2	TP2 -> TP3	TP3	TP3 -> TP4	TP4	TP4 -> TP5	TP5	TP5 -> Oral	Oral
			Effectué				A faire						
Mettre en place le Git	Git | Gestion de projet	Emilien											
Mettre au point l'architecture ECB	Architecture	Ninon											
Mettre au point l'idée des cases spéciales	Noyau	Robin											
Developper le contrôleur de jeu	Noyau	Robin											
Développer les contrôleurs de cases spéciales	Noyau	Robin											
Developper les contrôleurs d'événements parties	Noyau	Robin											
Développer le lancer de dés	IHM	Lucas											
Développer la grille de jeu	IHM	Lucas											
Création des entités / Modèles	Architecture	Ninon											
Spécifier les tests	Tests	Oscar											
Spécifier le projet	Git | Gestion de projet	Oscar											
Faire le diagramme de classe	UML - Modèle	Laura											
Faire le diagramme de séquence	UML - Modèle	Laura											
Création des interfaces	Architecture	Ninon											
Créer les tests	Tests	Oscar											
Faire un rapport de Test	Tests	Oscar											
Réunion de début de TP	Reunions | Autre	Oscar											
Tester le Git	Git | Gestion de projet	Emilien											
Développer la frame IHM	IHM	Lucas											
Affichage console programme	Noyau	Robin											
Préparer les oraux	Reunions | Autre	Oscar											
Relire le code	Noyau	Robin											
Faire les dernières modifications du code	Noyau	Robin											
Faire l'ensemble plateau	IHM	Lucas											
Intégrer tout les éléments à la Frame principale	IHM	Lucas											
Faire le JDialog	Architecture	Ninon											
Lier l'IHM au Noyau de jeu	IHM	Lucas											
Finir la spécification	UML - Modèle	Laura											
Faire le plan de tests	Tests	Oscar											
Faire le diagramme de classe	UML - Modèle	Laura											
Faire le diagramme de séquence	UML - Modèle	Laura											
Faire les tests	Tests	Oscar											
Faire le rapport de tests	Tests	Oscar											
Faire le merge sur main	Git | Gestion de projet	Emilien											
Créer l'exécutable	Git | Gestion de projet	Emilien											

Utilisation de GitHub

Historique des commits :

https://github.com/emilien3/Jeu_Pirate/pulse

Présentation du jeu

BOUE

LIANES

NOURRITURE

FALAISE

PASSAGE SECRET

RETOUR DÉBUT

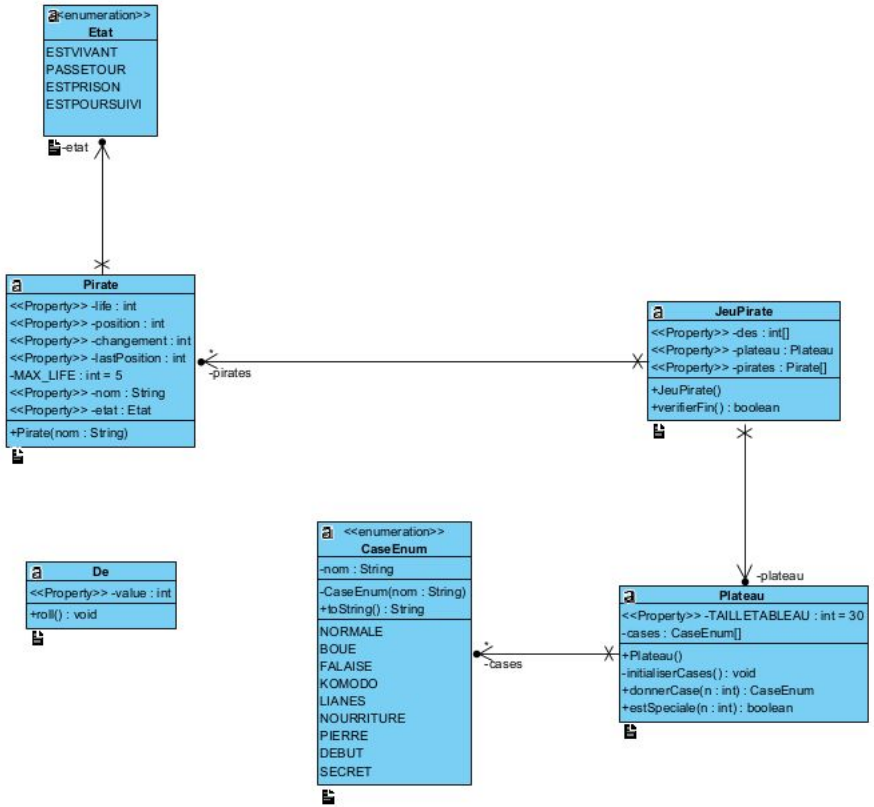
KOMODO

PIERRES

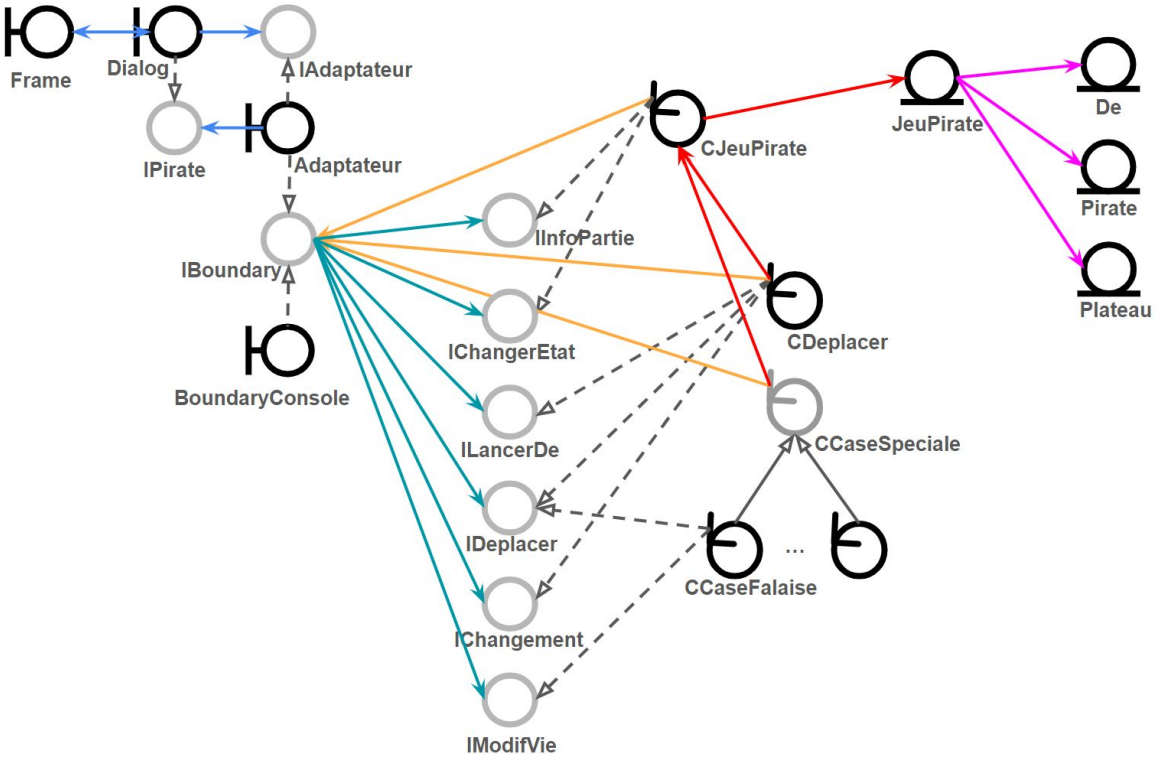
NORMALE

FIN

Entités/modèle



Architecture ECB



Diagrammes de classes

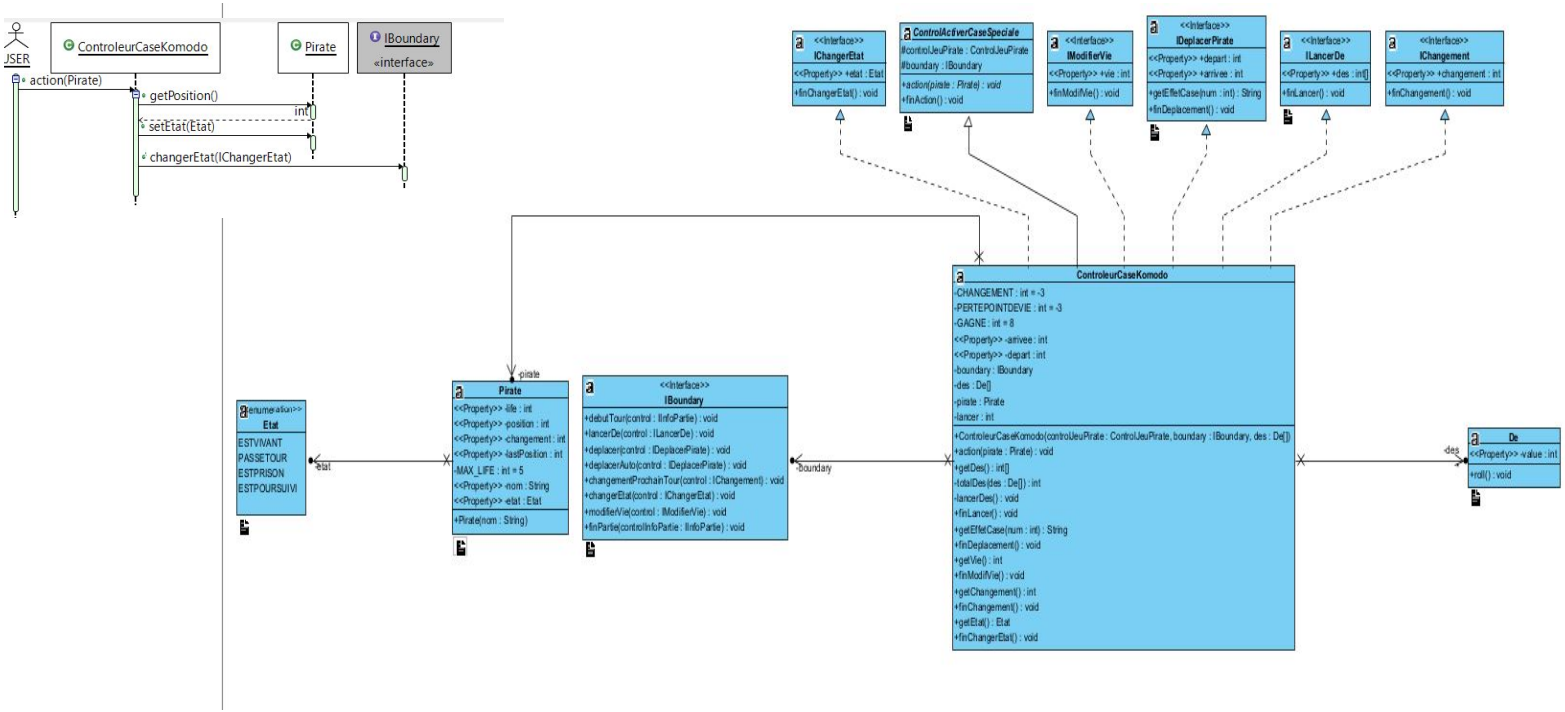


Diagramme de classes

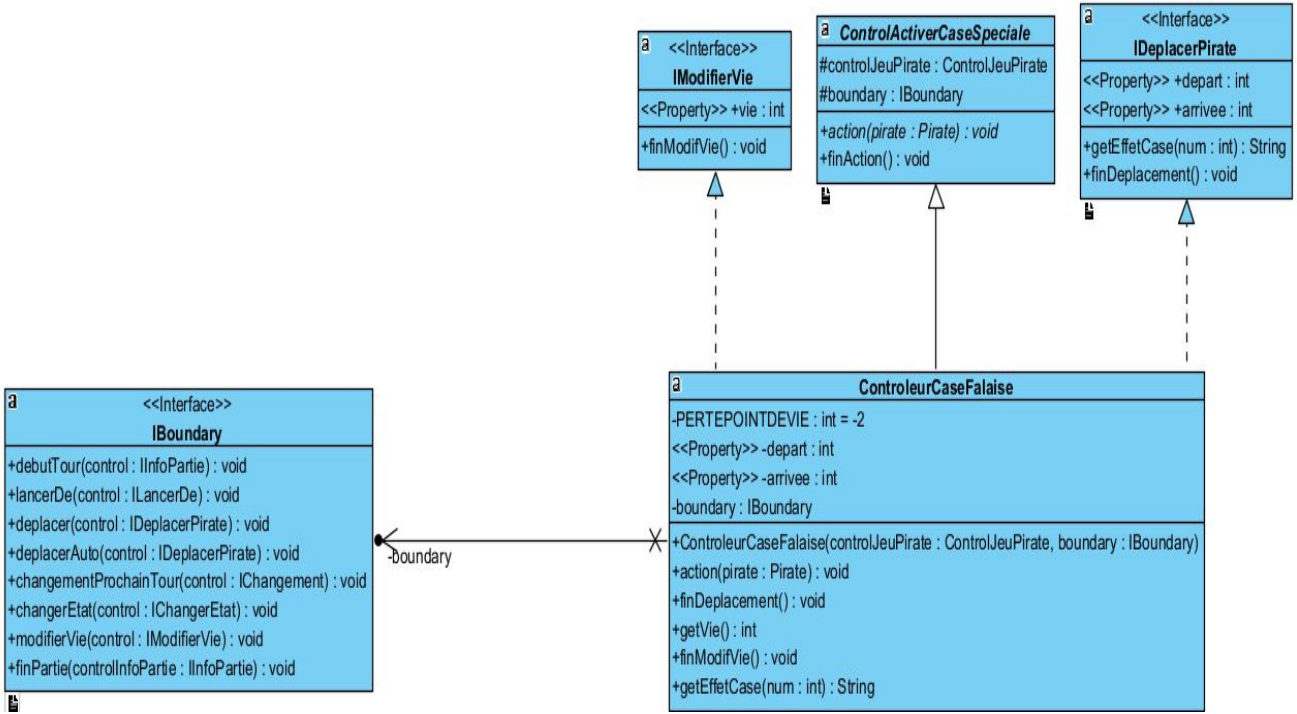


Diagramme de séquences

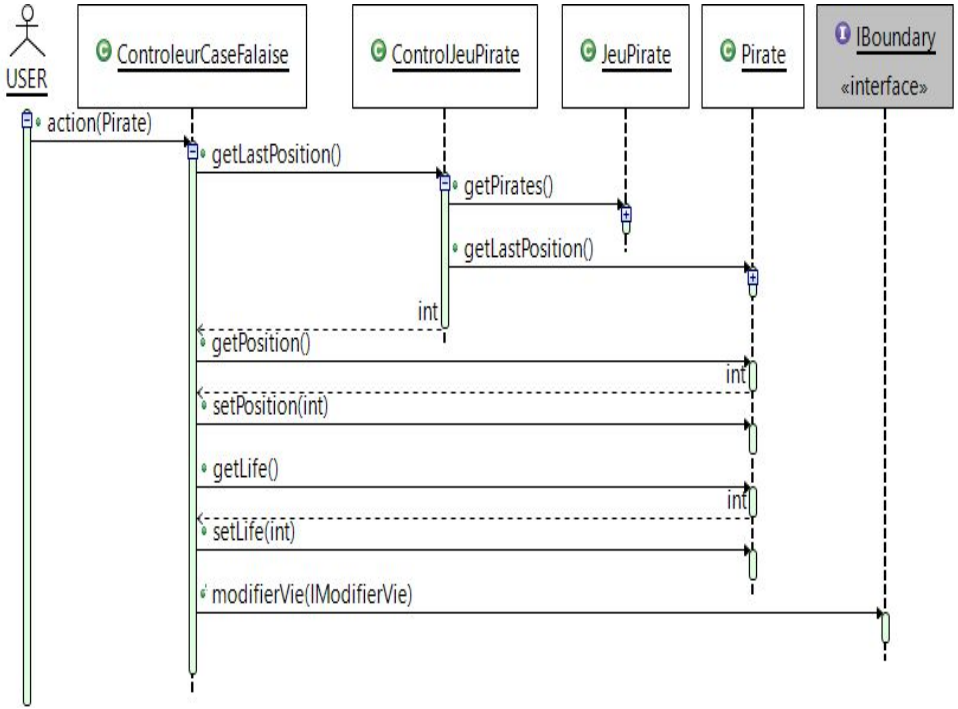
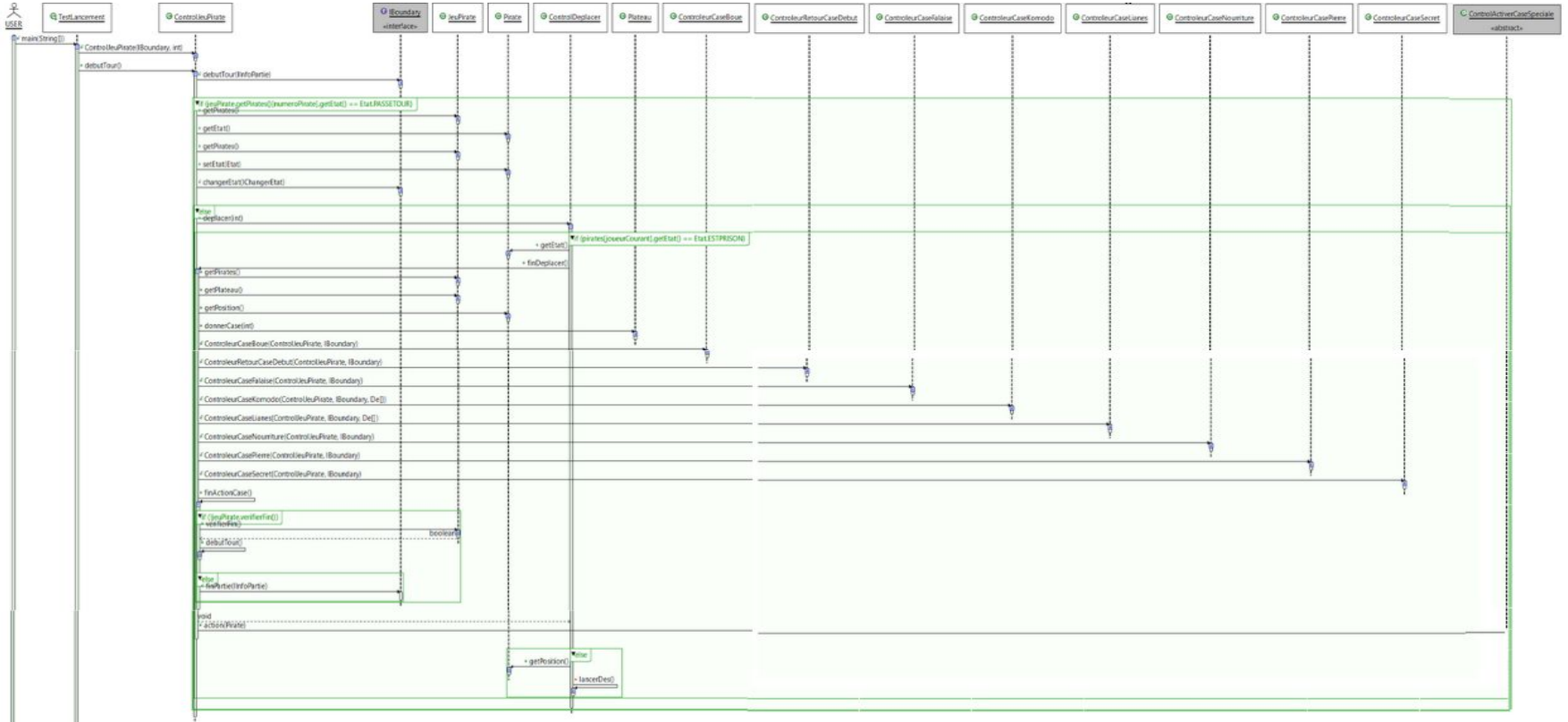


Diagramme de séquences



Apport des lambdas

```
timerArray[0] = new Timer(100, e -> {
    // Changer l'image du dé aléatoirement
    int index = seed.nextInt(6);
    diceArray[0].changeImage(getBuffer().get(index));
});
```

DiceCoursePanel

```
timer = new Timer(1000,  (e) -> {timerEventHandler(e);});
```

PanelCase

```
Collections.sort(cases , (PanelCase case1, PanelCase case2) -> {
int posCase1 = layeredPaneCase.getPosition(case1);
int posCase2 = layeredPaneCase.getPosition(case2);
return Integer.compare(posCase1, posCase2);
});
```

JPlateau

```
private int valeurDeplacement(De[] des){
    //Pour recupérer la somme des valeurs des dés
    Function<De[], Integer> somme = d -> Stream.of(d)
            .mapToInt(d1 -> d1.getValue()).reduce(0, (a, b) -> a+b);
    return somme.apply(des);
}

@Override
public int[] getDes() {
    //Est appelée quand l'affichage a besoin du résultat des dés
    Function<De[], int[]> getValeurs = d -> Stream.of(d)
            .mapToInt(d1 -> d1.getValue()).toArray();
    return getValeurs.apply(de);
}
```

ControlDeplacer

Phase de tests

Problems Javadoc Declaration **Console** SonarLint On-The-Fly Coverage

testControl (May 16, 2024 8:44:07 PM)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
Jeu_Pirate	29.0 %	2,576	6,316	8,892
src/main/java	24.5 %	1,897	5,838	7,735
boundary	4.6 %	279	5,765	6,044
control	95.2 %	1,080	55	1,135
ControlActiverCaseSpeciale.java	100.0 %	7	0	7
ControlDeplacer.java	97.7 %	217	5	222
ControleurCaseBoue.java	100.0 %	25	0	25
ControleurCaseFalaise.java	100.0 %	59	0	59
ControleurCaseKomodo.java	91.7 %	188	17	205
ControleurCaseLianes.java	100.0 %	152	0	152
ControleurCaseNourriture.java	100.0 %	41	0	41
ControleurCasePierre.java	100.0 %	22	0	22
ControleurCaseSecret.java	83.0 %	39	8	47
ControleurRetourCaseDebut.java	81.4 %	35	8	43
ControlJeuPirate.java	94.6 %	295	17	312
model	96.8 %	538	18	556
CaseEnum.java	100.0 %	114	0	114
De.java	100.0 %	18	0	18
Etat.java	100.0 %	72	0	72
JeuPirate.java	91.2 %	83	8	91
Pirate.java	100.0 %	64	0	64
Plateau.java	94.9 %	187	10	197
src/test/java	58.7 %	679	478	1,157

Problems Javadoc Declaration **Console** SonarLint On-The-Fly Coverage

testModel (May 16, 2024 8:36:26 PM)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
Jeu_Pirate	9.2 %	822	8,070	8,892
src/test/java	23.0 %	266	891	1,157
src/main/java	7.2 %	556	7,179	7,735
boundary	0.0 %	0	6,044	6,044
control	0.0 %	0	1,135	1,135
model	100.0 %	556	0	556
De.java	100.0 %	18	0	18
Pirate.java	100.0 %	64	0	64
Etat.java	100.0 %	72	0	72
JeuPirate.java	100.0 %	91	0	91
CaseEnum.java	100.0 %	114	0	114
Plateau.java	100.0 %	197	0	197

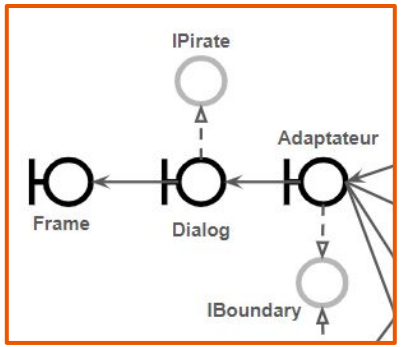
Problems Javadoc Declaration Console SonarLint On-The-Fly Coverage

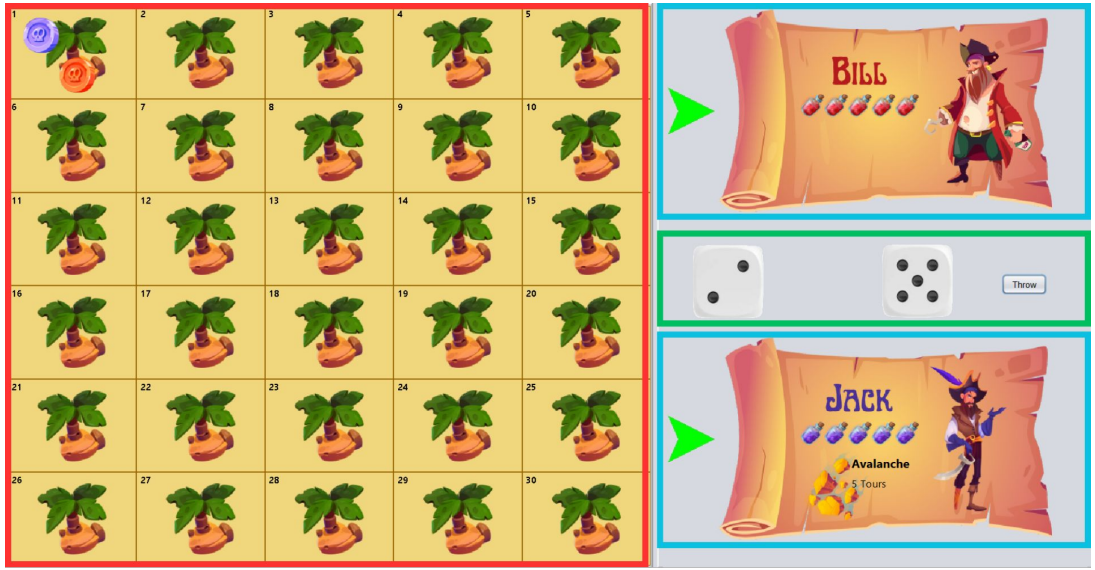
java (May 16, 2024 8:37:40 PM)

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
Jeu_Pirate	32.2 %	2,862	6,030	8,892
src/test/java	89.8 %	1,039	118	1,157
src/main/java	23.6 %	1,823	5,912	7,735
boundary	4.6 %	279	5,765	6,044
model	100.0 %	556	0	556
De.java	100.0 %	18	0	18
Pirate.java	100.0 %	64	0	64
Etat.java	100.0 %	72	0	72
JeuPirate.java	100.0 %	91	0	91
CaseEnum.java	100.0 %	114	0	114
Plateau.java	100.0 %	197	0	197
control	87.0 %	988	147	1,135
ControlActiverCaseSpeciale.java	100.0 %	7	0	7
ControleurCasePierre.java	63.6 %	14	8	22
ControleurCaseBoue.java	68.0 %	17	8	25
ControleurRetourCaseDebut.java	81.4 %	35	8	43
ControleurCaseNourriture.java	100.0 %	41	0	41
ControleurCaseFalaise.java	78.0 %	46	13	59
ControleurCaseSecret.java	100.0 %	47	0	47
ControleurCaseLianes.java	100.0 %	152	0	152
ControleurCaseKomodo.java	87.3 %	179	26	205
ControlDeplacer.java	97.7 %	217	5	222
ControlJeuPirate.java	74.7 %	233	79	312

IHM et liaison noyau/IHM

- Panels indépendants
- Simple d'utilisation
- Propriétés accessibles





Jplateau InfosJoueurs DiceCoursePanel

IHM et liaison noyau/IHM

```java
public interface IPirates {

    //Fonction qui seront appelées depuis l'adaptateur noyau
    public void initDialog();
    public void changerJoueur();
    public void enableLancerDe();
    public void enableDeplacement();
    public void changerPositionPirate(int numCase);
    public void changerChangement();
    public void changerEtat();
    public void changerVie();
    public void finDePartie();
}
```

Figure 1. Interface **IPirates**

```java
@Override
public void initDialog() {
    frame.setDialog(this);
    //Initialisation des pirates
    frame.getInfosJoueurBill().setPV(5);
    frame.getInfosJoueurJack().setPV(5);
    //Initialisation des images des cases
    Map<CaseEnum, String> mapPlateau = new EnumMap<>(CaseEnum.class);
    mapPlateau.put(CaseEnum.NORMALE, "ile.png");
    mapPlateau.put(CaseEnum.DEBUT, "retour.png");
    mapPlateau.put(CaseEnum.FALAISE, "falaise.png");
    mapPlateau.put(CaseEnum.NOURRITURE, "nourriture.png");
    mapPlateau.put(CaseEnum.PIERRE, "avalanche.png");
    mapPlateau.put(CaseEnum.KOMODO, "monstre.png");
    mapPlateau.put(CaseEnum.BOUE, "boue.png");
    mapPlateau.put(CaseEnum.LIANES, "lianes.png");
    mapPlateau.put(CaseEnum.SECRET, "longuevue.png");
    adaptateur.getTypeCase(0);
    int taillePlateau = adaptateur.getNombreCases();
    String[] imageCases = new String[taillePlateau];
    String[] descriptions = new String[taillePlateau];

    for (int i = 0; i < taillePlateau; i++) {
        imageCases[i] = mapPlateau.get(adaptateur.getTypeCase(i));
        descriptions[i] = adaptateur.getTypeCase(i).toString();
    }
    //La dernière case est la case d'arrivee
    imageCases[29] = "bateau.png";
    frame.getjPlateau().setImage(imageCases);
    frame.getjPlateau().setDescriptions(descriptions);

    //On ne peut pas bouger les pions tant que le tour n'a pas commencé
    frame.getjPlateau().getJetonBill().setMovable(false);
    frame.getjPlateau().getJetonJack().setMovable(false);
```

Figure 2. Fonction **initDialog()**

IHM et liaison noyau/IHM

```java
@Override
public void changerVie() {
    int joueurCourant = adaptateur.getPirateCourant();
    int vieDiff = adaptateur.getPointsVie();
    //TODO : mettre vie de numPirate à changement
    PanelInfosJoueur currPlayer = joueurCourant == 0 ? frame.getInfosJoueurBill() : frame.getInfosJoueurJack();
    int vie = Math.min(currPlayer.getPV() + vieDiff, 5);
    currPlayer.setPV(vie);
    adaptateur.finChangerVie();
}
```

Figure 3. Exemple de fonction : **changerVie()**

Démonstration

Partie très rapide

Démonstration

Partie rapide

Démonstration

Partie très lente

Conclusion : retour d'expérience

Compatibilité du code lors de la mise en commun

Gestion du Git : organisation du travail dans les branches

Difficultés de compréhension et de mise en place de l'architecture ECB

Liaison entre IHM et noyau fonctionnel

Acquisition de compétences : GitHub, travail de groupe, architecture ECB, Java…

Développement d'un projet complet de A à Z avec une sortie graphique fonctionnelle

Merci pour votre écoute

Liens utiles & Annexes

GitHub - https://github.com/emilien3/Jeu_Pirate
Moodle ILU4 - https://moodle.univ-tlse3.fr/course/view.php?id=6114

Assets Graphiques par KlyakSun sur Canva
Assets des dés et des cases spéciales générer par DALL-E puis retouchées