

Mille Bornes

Projet Informatique Deuxième année

Pons Thomas • Puaud Audrey • Relland Régis • Séité Damien



École Nationale de la Statistique et de l'Analyse de l'Information

Mardi 06 Décembre 2016



Introduction

Sous titre



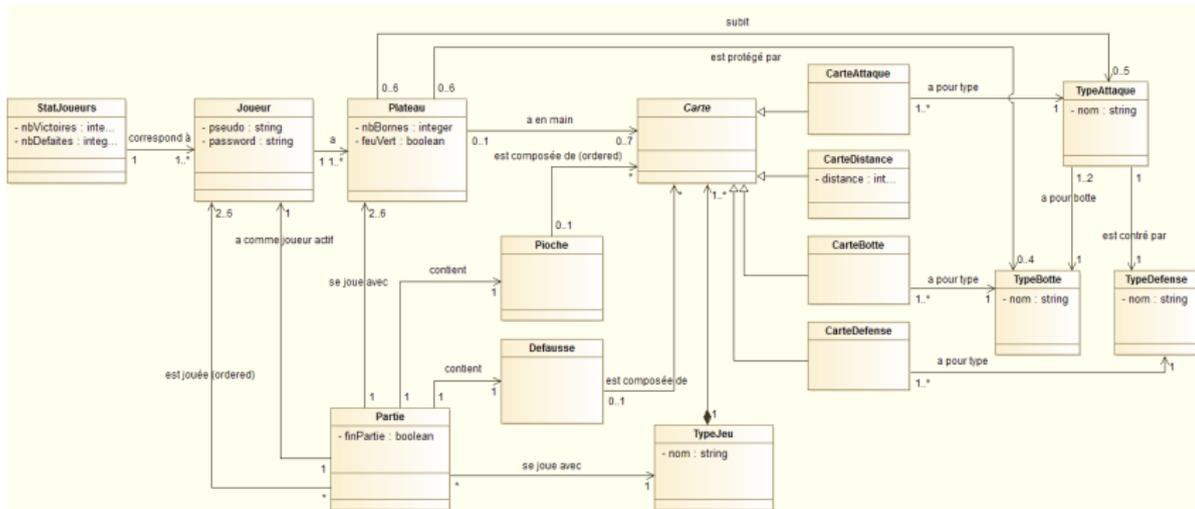


- 1 Modélisation
- 2 Implémentation
- 3 Graphisme
- 4 Gestion des exceptions
- 5 Stratégie de tests
- 6 Organisation du groupe



Modélisation

Diag. classes





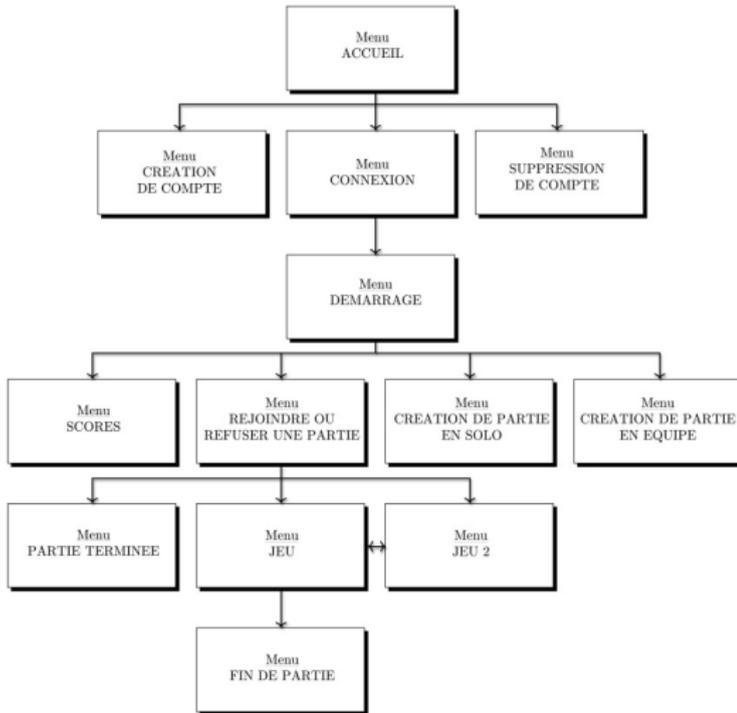
Sommaire

- 1 Modélisation
- 2 Implémentation**
- 3 Graphisme
- 4 Gestion des exceptions
- 5 Stratégie de tests
- 6 Organisation du groupe



Implémentation

Comment avons nous créé le jeu ?





Implémentation

Le coup fourré



```
// COUP FOURRÉ !!!
if (plateauDAO.possedeEnMainBotteCorrespondante(plateau, carteAttaque)) {
    long bornes = plateau.getNbBornes();
    plateau.setNbBornes(bornes + 300);
    plateau.setPeutRouler(true);
    plateauDAO.update(plateau);

    TypeBotte typeBotte1 = plateauDAO.findBotteCorrespondante(carteAttaque);
    long idcb = 0;
    if (typeBotte1.getIdTypeBotte() == 1) {
        idcb = 16;
    }
    if (typeBotte1.getIdTypeBotte() == 2) {
        idcb = 17;
    }
    if (typeBotte1.getIdTypeBotte() == 3) {
        idcb = 18;
    }
    if (typeBotte1.getIdTypeBotte() == 4) {
        idcb = 19;
    }
}

// le joueur ciblé pose la botte jouée
this.poseCarte(idPartie, idJoueurChoisi, idcb);
```



Implémentation

Le coup fourré

```

..
// le joueur ciblé pose la botte jouée
this.poseCarte(idPartie, idJoueurChoisi, idcb);

plateauDAO.createProtection(plateau, typeBottle);
plateauDAO.supprimeSubitSelonBotte(plateau, typeBottle);

// trouver le joueur juste avant le joueur ciblé
// afin qu'au chgt de joueur de fin de tour,
// le joueur ciblé devienne le joueur actif
long idJoueurPrePrecedent = this.findIdPrePrecedent(idPartie, idJoueurChoisi);
this.changeJoueurActif(idJoueurPrePrecedent, idPartie);

// si le joueur ciblé subissait l'attaque correspondante, il defausse la carte
List<Long> ListeCartesAdefausser = plateauDAO.findSubitAttaqueCorrespondantABotte(plateau.getIdPartie(),
    typeBottle.getIdTypeBotte());
if (ListeCartesAdefausser != null) {
    for (Long id : ListeCartesAdefausser) {
        this.defausseCarte(idPartie, idJoueurChoisi, id);
    }
}
}

```



Sommaire

- 1 Modélisation
- 2 Implémentation
- 3 Graphisme**
- 4 Gestion des exceptions
- 5 Stratégie de tests
- 6 Organisation du groupe



Graphisme

Sous titre





Sommaire

- 1 Modélisation
- 2 Implémentation
- 3 Graphisme
- 4 Gestion des exceptions**
- 5 Stratégie de tests
- 6 Organisation du groupe



Gestion des exceptions

Exception en Java

```
SaisieIncorrecteException.java
1 package projet.vue.exception;
2
3 /**
4  * Classe gérant les exceptions pour les saisies incorrectes.
5  * @author Mickael
6  *
7  */
8 public class SaisieIncorrecteException extends Exception{
9
10     /** Serial UID. */
11     private static final long serialVersionUID = 4021494048051860945L;
12
13     /**
14      * @param message le message expliquant pourquoi la saisie est incorrecte.
15      */
16     public SaisieIncorrecteException(String message){
17         super(message);
18     }
19
20
21
22 }
23
```



Gestion des exceptions

Erreur de saisie

```
[a] Créer une nouvelle partie  
[b] Rejoindre ou refuser une partie  
[c] Afficher les scores  
[d] Créer une nouvelle partie en équipe  
[r] Retour à l'écran d'accueil
```

kjhkh

L'option que vous avez choisie : kjhkh n'est pas admissible.

Tapez :

```
[a] Créer une nouvelle partie  
[b] Rejoindre ou refuser une partie  
[c] Afficher les scores  
[d] Créer une nouvelle partie en équipe  
[r] Retour à l'écran d'accueil
```



Gestion des exceptions

Exemple : InputMismatchException

```
try {
    optionChoisie2 = scanner.nextInt();
} catch (InputMismatchException e) {
    System.out.println("La valeur saisie n'est pas un entier");
    return new MenuDemarrage();
}
```



Sommaire

- 1 Modélisation
- 2 Implémentation
- 3 Graphisme
- 4 Gestion des exceptions
- 5 Stratégie de tests**
- 6 Organisation du groupe

Stratégie de tests

Sous titre

Java - ProjetTemplate/test/projet/controller/TestPartieController.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help



Quick Access

Package Explorer



Finished after 12,758 seconds

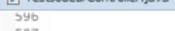
Runs: 26/26 Errors: 0 Failures: 0



test.projet.controller.TestPartieController

- testFindJoueurActif (2,173 s)
- testJouerCarteDistance (1,361 s)
- testUpdateEtatJoueurDansPartie (0,000 s)
- testFindEquipements (0,062 s)
- testFindJoueurs (0,047 s)
- testJouerCarteBottle (0,891 s)
- testPocher (0,595 s)
- testCreerPartie (0,031 s)
- testFindTypeJeuSelonPartie (0,015 s)
- testFindPartie (0,094 s)
- testFindTypeJeu (0,000 s)
- testFindEtatJoueur (0,063 s)
- testFindPlateau (0,766 s)
- testverifieSSubmitAttaqueLimitationVit (0,031 s)
- testFindEtat (0,031 s)
- testFindNb200km (1,829 s)
- testChangeJoueurActif (0,203 s)
- testFindBornesOrdonneesSelonBorne (0,000 s)
- testTestLancement (0,047 s)
- testFindScoresOrdonneesSelonBornes (0,000 s)
- testJouerCarteDefense (0,641 s)
- testFindTousTypeJeu (0,000 s)
- testCompteTaillePioche (0,797 s)
- testCompteNbJoueurs (0,063 s)
- testDeletePartieLong (0,031 s)
- testDeletePartieString (0,062 s)

TestJoueurController.java



596
597
598
599

```
@Test
```

```
public void testJouerCarteDefense() throws SQLException{
```

```
PartieController partieController = new PartieController();
```

```
JoueurController joueurController = new JoueurController();
```

```
CarteAttaqueDAO carteAttaqueDAO = new CarteAttaqueDAO();
```

```
CarteDefenseDAO carteDefenseDAO = new CarteDefenseDAO();
```

```
604
```

```
605
```

```
TypeJeu typeJeu = new TypeJeu(1);
```

```
Joueur joueur = joueurController.creerJoueur("AudreyTest", "audrey");
```

```
Joueur joueur2 = joueurController.creerJoueur("DamienTest", "damien");
```

```
List<Joueur> listeJoueurs = new ArrayList<Joueur>(2);
```

```
listeJoueurs.add(joueur);
```

```
listeJoueurs.add(joueur2);
```

```
Partie partie = partieController.creerPartie(listeJoueurs,typeJeu);
```

```
partieController.initialiser(partie.getIdPartie(), listeJoueurs, 1);
```

```
CarteAttaque carteAttaque = carteAttaqueDAO.find(6); // feu rouge
```

```
614 CarteDefense carteDefense = carteDefenseDAO.find(11); // feu vert
```

```
615
```

```
partieController.jouerCarteAttaque(partie.getIdPartie(), carteAttaque, joueur2.getId());
```

```
partieController.jouerCarteDefense(partie.getIdPartie(), joueur2.getId(), carteDefense);
```

```
618 assertTrue(partieController.verifieSSubmitAttaqueLourde(partie.getIdPartie(), joueur2.getId()));
```

```
619
```

```
620
```

```
/* Suppression des lignes créées dans la BDD au cours du test */
```

```
Session.getInstance().setJoueur(joueur);
```

```
partieController.deletePartie("AudreyTest", "audrey");
```

```
623 joueurController.deleteJoueur("AudreyTest", "audrey");
```

```
624 Session.getInstance().setJoueur(joueur2);
```

```
partieController.deletePartie("DamienTest", "damien");
```

```
626 joueurController.deleteJoueur("DamienTest", "damien");
```

```
627 }
```

```
628
```

```
K7R
```

TestPartieController.java



596
597
598
599

```
@Test
```

```
public void testJouerCarteDefense() throws SQLException{
```

```
PartieController partieController = new PartieController();
```

```
JoueurController joueurController = new JoueurController();
```

```
CarteAttaqueDAO carteAttaqueDAO = new CarteAttaqueDAO();
```

```
CarteDefenseDAO carteDefenseDAO = new CarteDefenseDAO();
```

```
604
```

```
605
```

```
TypeJeu typeJeu = new TypeJeu(1);
```

```
Joueur joueur = joueurController.creerJoueur("AudreyTest", "audrey");
```

```
Joueur joueur2 = joueurController.creerJoueur("DamienTest", "damien");
```

```
List<Joueur> listeJoueurs = new ArrayList<Joueur>(2);
```

```
listeJoueurs.add(joueur);
```

```
listeJoueurs.add(joueur2);
```

```
Partie partie = partieController.creerPartie(listeJoueurs,typeJeu);
```

```
partieController.initialiser(partie.getIdPartie(), listeJoueurs, 1);
```

```
CarteAttaque carteAttaque = carteAttaqueDAO.find(6); // feu rouge
```

```
614 CarteDefense carteDefense = carteDefenseDAO.find(11); // feu vert
```

```
615
```

```
partieController.jouerCarteAttaque(partie.getIdPartie(), carteAttaque, joueur2.getId());
```

```
partieController.jouerCarteDefense(partie.getIdPartie(), joueur2.getId(), carteDefense);
```

```
618 assertTrue(partieController.verifieSSubmitAttaqueLourde(partie.getIdPartie(), joueur2.getId()));
```

```
619
```

```
620
```

```
/* Suppression des lignes créées dans la BDD au cours du test */
```

```
Session.getInstance().setJoueur(joueur);
```

```
partieController.deletePartie("AudreyTest", "audrey");
```

```
623 joueurController.deleteJoueur("AudreyTest", "audrey");
```

```
624 Session.getInstance().setJoueur(joueur2);
```

```
partieController.deletePartie("DamienTest", "damien");
```

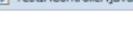
```
626 joueurController.deleteJoueur("DamienTest", "damien");
```

```
627 }
```

```
628
```

```
K7R
```

TestIACController.java



596
597
598
599

```
@Test
```

```
public void testJouerCarteDefense() throws SQLException{
```

```
PartieController partieController = new PartieController();
```

```
JoueurController joueurController = new JoueurController();
```

```
CarteAttaqueDAO carteAttaqueDAO = new CarteAttaqueDAO();
```

```
CarteDefenseDAO carteDefenseDAO = new CarteDefenseDAO();
```

```
604
```

```
605
```

```
TypeJeu typeJeu = new TypeJeu(1);
```

```
Joueur joueur = joueurController.creerJoueur("AudreyTest", "audrey");
```

```
Joueur joueur2 = joueurController.creerJoueur("DamienTest", "damien");
```

```
List<Joueur> listeJoueurs = new ArrayList<Joueur>(2);
```

```
listeJoueurs.add(joueur);
```

```
listeJoueurs.add(joueur2);
```

```
Partie partie = partieController.creerPartie(listeJoueurs,typeJeu);
```

```
partieController.initialiser(partie.getIdPartie(), listeJoueurs, 1);
```

```
CarteAttaque carteAttaque = carteAttaqueDAO.find(6); // feu rouge
```

```
614 CarteDefense carteDefense = carteDefenseDAO.find(11); // feu vert
```

```
615
```

```
partieController.jouerCarteAttaque(partie.getIdPartie(), carteAttaque, joueur2.getId());
```

```
partieController.jouerCarteDefense(partie.getIdPartie(), joueur2.getId(), carteDefense);
```

```
618 assertTrue(partieController.verifieSSubmitAttaqueLourde(partie.getIdPartie(), joueur2.getId()));
```

```
619
```

```
620
```

```
/* Suppression des lignes créées dans la BDD au cours du test */
```

```
Session.getInstance().setJoueur(joueur);
```

```
partieController.deletePartie("AudreyTest", "audrey");
```

```
623 joueurController.deleteJoueur("AudreyTest", "audrey");
```

```
624 Session.getInstance().setJoueur(joueur2);
```

```
partieController.deletePartie("DamienTest", "damien");
```

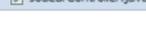
```
626 joueurController.deleteJoueur("DamienTest", "damien");
```

```
627 }
```

```
628
```

```
K7R
```

JoueurController.java



596
597
598
599

```
@Test
```

```
public void testJouerCarteDefense() throws SQLException{
```

```
PartieController partieController = new PartieController();
```

```
JoueurController joueurController = new JoueurController();
```

```
CarteAttaqueDAO carteAttaqueDAO = new CarteAttaqueDAO();
```

```
CarteDefenseDAO carteDefenseDAO = new CarteDefenseDAO();
```

```
604
```

```
605
```

```
TypeJeu typeJeu = new TypeJeu(1);
```

```
Joueur joueur = joueurController.creerJoueur("AudreyTest", "audrey");
```

```
Joueur joueur2 = joueurController.creerJoueur("DamienTest", "damien");
```

```
List<Joueur> listeJoueurs = new ArrayList<Joueur>(2);
```

```
listeJoueurs.add(joueur);
```

```
listeJoueurs.add(joueur2);
```

```
Partie partie = partieController.creerPartie(listeJoueurs,typeJeu);
```

```
partieController.initialiser(partie.getIdPartie(), listeJoueurs, 1);
```

```
CarteAttaque carteAttaque = carteAttaqueDAO.find(6); // feu rouge
```

```
614 CarteDefense carteDefense = carteDefenseDAO.find(11); // feu vert
```

```
615
```

```
partieController.jouerCarteAttaque(partie.getIdPartie(), carteAttaque, joueur2.getId());
```

```
partieController.jouerCarteDefense(partie.getIdPartie(), joueur2.getId(), carteDefense);
```

```
618 assertTrue(partieController.verifieSSubmitAttaqueLourde(partie.getIdPartie(), joueur2.getId()));
```

```
619
```

```
620
```

```
/* Suppression des lignes créées dans la BDD au cours du test */
```

```
Session.getInstance().setJoueur(joueur);
```

```
partieController.deletePartie("AudreyTest", "audrey");
```

```
623 joueurController.deleteJoueur("AudreyTest", "audrey");
```

```
624 Session.getInstance().setJoueur(joueur2);
```

```
partieController.deletePartie("DamienTest", "damien");
```

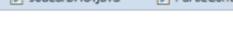
```
626 joueurController.deleteJoueur("DamienTest", "damien");
```

```
627 }
```

```
628
```

```
K7R
```

JoueurDAO.java



596
597
598
599

```
@Test
```

```
public void testJouerCarteDefense() throws SQLException{
```

```
PartieController partieController = new PartieController();
```

```
JoueurController joueurController = new JoueurController();
```

```
CarteAttaqueDAO carteAttaqueDAO = new CarteAttaqueDAO();
```

```
CarteDefenseDAO carteDefenseDAO = new CarteDefenseDAO();
```

```
604
```

```
605
```

```
TypeJeu typeJeu = new TypeJeu(1);
```

```
Joueur joueur = joueurController.creerJoueur("AudreyTest", "audrey");
```

```
Joueur joueur2 = joueurController.creerJoueur("DamienTest", "damien");
```

```
List<Joueur> listeJoueurs = new ArrayList<Joueur>(2);
```

```
listeJoueurs.add(joueur);
```

```
listeJoueurs.add(joueur2);
```

```
Partie partie = partieController.creerPartie(listeJoueurs,typeJeu);
```

```
partieController.initialiser(partie.getIdPartie(), listeJoueurs, 1);
```

```
CarteAttaque carteAttaque = carteAttaqueDAO.find(6); // feu rouge
```

```
614 CarteDefense carteDefense = carteDefenseDAO.find(11); // feu vert
```

```
615
```

```
partieController.jouerCarteAttaque(partie.getIdPartie(), carteAttaque, joueur2.getId());
```

```
partieController.jouerCarteDefense(partie.getIdPartie(), joueur2.getId(), carteDefense);
```

```
618 assertTrue(partieController.verifieSSubmitAttaqueLourde(partie.getIdPartie(), joueur2.getId()));
```

```
619
```

```
620
```

```
/* Suppression des lignes créées dans la BDD au cours du test */
```

```
Session.getInstance().setJoueur(joueur);
```

```
partieController.deletePartie("AudreyTest", "audrey");
```

```
623 joueurController.deleteJoueur("AudreyTest", "audrey");
```

```
624 Session.getInstance().setJoueur(joueur2);
```

```
partieController.deletePartie("DamienTest", "damien");
```

```
626 joueurController.deleteJoueur("DamienTest", "damien");
```

```
627 }
```

```
628
```

```
K7R
```

PartieController.java



596
597
598
599

```
@Test
```

```
public void testJouerCarteDefense() throws SQLException{
```

```
PartieController partieController = new PartieController();
```

```
JoueurController joueurController = new JoueurController();
```

```
CarteAttaqueDAO carteAttaqueDAO = new CarteAttaqueDAO();
```

```
CarteDefenseDAO carteDefenseDAO = new CarteDefenseDAO();
```

```
604
```

```
605
```

```
TypeJeu typeJeu = new TypeJeu(1);
```

```
Joueur joueur = joueurController.creerJoueur("AudreyTest", "audrey");
```

```
Joueur joueur2 = joueurController.creerJoueur("DamienTest", "damien");
```

```
List<Joueur> listeJoueurs = new ArrayList<Joueur>(2);
```

```
listeJoueurs.add(joueur);
```

```
listeJoueurs.add(joueur2);
```

```
Partie partie = partieController.creerPartie(listeJoueurs,typeJeu);
```

```
partieController.initialiser(partie.getIdPartie(), listeJoueurs, 1);
```

```
CarteAttaque carteAttaque = carteAttaqueDAO.find(6); // feu rouge
```

```
614 CarteDefense carteDefense = carteDefenseDAO.find(11); // feu vert
```

```
615
```

```
partieController.jouerCarteAttaque(partie.getIdPartie(), carteAttaque, joueur2.getId());
```

```
partieController.jouerCarteDefense(partie.getIdPartie(), joueur2.getId(), carteDefense);
```

```
618 assertTrue(partieController.verifieSSubmitAttaqueLourde(partie.getIdPartie(), joueur2.getId()));
```

```
619
```

```
620
```

```
/* Suppression des lignes créées dans la BDD au cours du test */
```

```
Session.getInstance().setJoueur(joueur);
```

```
partieController.deletePartie("AudreyTest", "audrey");
```

```
623 joueurController.deleteJoueur("AudreyTest", "audrey");
```

```
624 Session.getInstance().setJoueur(joueur2);
```

```
partieController.deletePartie("DamienTest", "damien");
```

```
626 joueurController.deleteJoueur("DamienTest", "damien");
```

```
627 }
```

```
628
```

```
K7R
```

Problems Javadoc Declaration Search Console Call Hierarchy Tasks Debug

<terminated> TestPartieController (1) [JUnit] C:\Program Files\Java\jre1.8.0_65\bin\javaw.exe (5 déc. 2016 11:56:19)

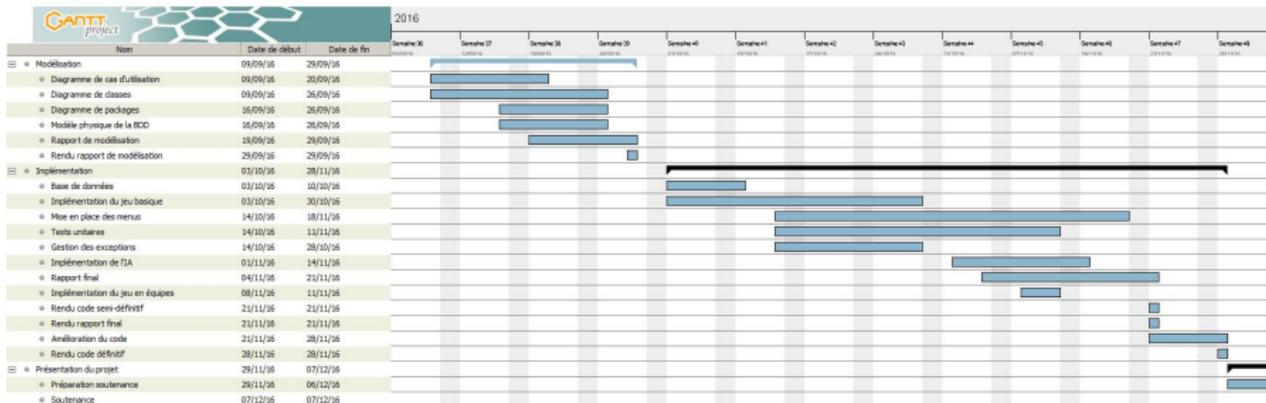


- 1 Modélisation
- 2 Implémentation
- 3 Graphisme
- 4 Gestion des exceptions
- 5 Stratégie de tests
- 6 Organisation du groupe**



Organisation du groupe

Sous titre



Mille Bornes

Projet Informatique Deuxième année

Pons Thomas • Puaud Audrey • Relland Régis • Séité Damien



École Nationale de la Statistique et de l'Analyse de l'Information

Mardi 06 Décembre 2016