GESTIN 24/05/18

Gaëtan Semaine 21

Suivi de Projet – Projet Arcanor

Résumé :

Finition des signatures de toutes les classes, création de la javaDoc par rapport à l'analyse du projet, diagrammes et groupe. Ansi que la création du build.xml avec Mathias.

Annexe :

Arcanor.java

```
package arcanor;

/**
 * Cette classe est la classe qui lance le jeu
 */
public class Arcanor{

   /**
    * Vrai si jeu fini
    */
   private boolean end;

   /**
    * Routine principale de lancement
    * @param arg arguments possibles
    */
   public static void main(String[] arg) {

   }
}
```

GameBoard.java

```
package arcanor;
import java.util.ArrayList;

/**
 * Cette classe représente un plateau de jeu avec son déroulement
 */
public class GameBoard{

   /**
    * le numero du tour courrant
    */
   private int turn;

   /**
    * le tableau de paramètres de la partie
    */
   private ParamMenu gameParams;

   /**
    * la liste de joueurs dans la partie
    */
   private ArrayList<Player> playerList;

   /**
    * le tableau de jeu (plateau)
    */
   private Piece[][] board;

   /**
    * Le constructeur
    * @param params le tableau de paramètres
    * @param list la liste de joueurs
    */
   public GameBoard(ParamMenu params, ArrayList<Player> list){}

   /**
    * Fonction qui permet la sauvegarde
    */
   public void save(){}

   /**
    * Getter de turn
    * @return tour courrant
    */
   public int getTurn(){}

   /**
    * Setter de turn
    * @param turn le nouveau tour
    */
   public void setTurn(int turn){}

   /**
    * Fonction qui est un tour du jeu
```

```
     */
    public void play(){}

    /**
     * Fonction d'affichage
     */
    public void display(){}

}
```

GameMenu.java

```
package arcanor;

/**
 * Cette classe est le menu permettant de rediriger vers les pages
de lancement de parties
 */
public class GameMenu extends Menu{

      /**
       * Fonction menant vers StartMenu
       */
      public void startPlay(){}

      /**
       * Fonction menant vers LoadMenu
       */
      public void loadMenu(){}

      /**
       * Fonction d'affichage
       */
      public void display(){}

      /**
       * Fonction de retour arrière du menu
       */
      public void back(){}

}
```

LoadMenu.java

```
package arcanor;

/**
 * Cette classe est le menu permettant de charger une partie
 */
public class LoadMenu extends Menu{

    /**
     * Variable determinant le mode de jeu
     */
```

```
    private boolean mode;

    /**
     * Le Contructeur
     */
    public LoadMenu(){}

    /**
     * Getter de mode
     * @return le mode actuel
     */
    public boolean getMode(){}

    /**
     * Setter de mode
     * @param mode mode a changer
     */
    public void setMode(boolean mode){}

    /**
     * Fonction d'affichage
     */
    public void display(){}

    /**
        * Fonction de retour arrière du menu
        */
    public void back(){}

}
```

MainMenu.java

```
package arcanor;

/**
 * Cette classe est le menu principal du jeu
 */
public class MainMenu extends Menu{

      /**
       * Fonction menant vers GameMenu
       */
      public void lauchGame(){}

      /**
       * Fonction menant vers ScoreMenu
       */
      public void scoreMenu(){}

      /**
     * Fonction d'affichage
     */
    public void display(){}
```

```
    /**
     * Fonction de retour arrière du menu
     */
    public void back(){}

}
```

Menu.java

```
package arcanor;

/**
 * Cette classe abstraite sert de modèle aux autres menus
 */
public abstract class Menu{

    /**
     * titre du menu
     */
    private String title;

    /**
     * Le constructeur
     * @param title titre a afficher
     */
    public Menu(String title){
        this.title=title;
    }

    /**
     * Fonction d'affichage
     */
    public void display(){}

    /**
     * Fonction de retour arrière du menu
     */
    public void back(){}

}
```

ParamMenu.java

```
package arcanor;

/**
 * Cette classe est le menu permettant de modifier les paramètres de
la partie
 */
public class ParamMenu extends Menu{

    /**
     * tableau de paramètres codés
     */
```

```
    private String[] tabParams;

    /**
     * Le contructeur
     */
    public ParamMenu(){}

    /**
     * Getter de tabParams
     * @return le tableau de paramètres
     */
    public String[] getTabParams(){}

    /**
     * Setter de tabParams
     * @param tabParams le tableau de paramètres
     */
    public void setTabParams(String[] tabParams){}

    // public void setDifficulte(int niveau){}
    //
    /**
    * Fonction d'affichage
    */
    public void display(){}

    /**
    * Fonction de retour arrière du menu
    */
    public void back(){}

}
```

Piece.java

```
package arcanor;

/**
 * Cette classe représente une pièce du jeu
 */
public class Piece{

    /**
     * la couleur de la pièce
     */
    private int color;

    /**
     * la valeur de la pièce
     */
    private int value;

    /**
     * le contenu de la pièce
     */
```

```
    private Piece contain;

    /**
     * le Contructeur
     * @param color la couleur de la pièce
     * @param value la valeur de la pièce
     * @param contain le contenu de la pièce
     */
    public Piece(int color, int value, Piece contain){}

    /**
     * Getter de color
     * @return la couleur de la pièce
     */
    public int getColor(){}

    /**
     * Setter de color
     * @param color la couleur a attribuer
     */
    public void setColor(int color){}

    /**
     * Getter de value
     * @return la valeur de la pièce
     */
    public int getValue(){}

    /**
     * Setter de value
     * @param value la valeur a attribuer
     */
    public void setValue(int value){}

    /**
     * Getter de contain
     * @return le contenu de la pièce
     */
    public Piece getContain(){}

    /**
     * Setter de contain
     * @param contain le contenu a attribuer
     */
    public void setContain(Piece contain){}

}
```

Player.java

```
package arcanor;

/**
 * Cette classe représente un joueur
 */
```

```java
public class Player{

   /**
    * variable d'humanité, true si humain
    */
   private boolean human;

   /**
    * nom du joueur
    */
   private String name;

   /**
    * Le constructeur
    * @param type humain ou non
    * @param name nom du joueur
    */
   public Player(boolean type, String name){}

   /**
    * Fonction demandant les entrées clavier pour jouer
    * @return le tableau d'actions a effectuer
    */
   public int[][] playTurn(){}

}
```

ScoreMenu.java

```java
package arcanor;

/**
 * Cette classe est le menu qui permet d'afficher les scores
 */
public class ScoreMenu extends Menu{

      /**
       * le tableau contenant le nom des joueurs
       */
      private Player[] playerList;

      /**
       * le tableau contenant le score de chaque joueurs
       */
      private int[] tabScores[];

      /**
       * Le constructeur
       */
   public ScoreMenu(){}

      /**
       * Fonction permettant de charger depuis les fichiers
       */
      public void loadScores(){}
```

```
	/**
	 * Getter de playerList
	 * @return le tableau des joueurs
	 */
	public Player[] getPlayerList(){}

	/**
	 * Getter de tabScores
	 * @return le tableau des scores
	 */
	public int[] getTabScores(){}

	/**
	 * Setter de tabScores
	 * @param tabScores le tableau des scores
	 */
	public void setTabScores(int[] tabScores){}

	/**
	 * Fonction d'affichage
	 */
	public void display(){}

	/**
	 * Fonction de retour arrière du menu
	 */
	public void back(){}

}
```

StartMenu.java

```
package arcanor;

/**
 * Cette classe est le menu de lancement d'une partie (création)
 */
public class StartMenu extends Menu{

	/**
	 * le nombre de joueurs humain
	 */
	private int playerNb;

	/**
	 * le tableau de paramètres
	 */
	private ParamMenu startParams;

	/**
	 * Le constructeur
	 * @param nbJoueurs le nombre de joueurs
	 */
	public StartMenu(int nbJoueurs){}
```

```
    /**
     * Getter de playerNb
     * @return le nombre de joueurs humain
     */
    public int getPlayerNb(){}

    /**
     * Setter de playerNb
     * @param playerNb le nombre a attribuer
     */
    public void setPlayerNb(int playerNb){}

    /**
     * Setter de nomJoueurs
     * @param nomJoueurs [description]
     */
    public void setNomJoueurs(String[] nomJoueurs){}

    /**
     * Getter de startParams
     * @return le tableau de paramètres
     */
  public ParamMenu getStartParams(){}

}
```

build.xml

```
<project name="Arcanor" default="jar" basedir="./src">
    <description>
        Build permettant la compilation de tous les fichiers .java
    </description>

  <!-- set global properties for this build -->
  <property name="src" location="src"/>
  <property name="jar" location="jar"/>
  <property name="class" location="class"/>
  <property name="javadoc" location="javaDoc"/>
  <property name="main-class"  value="LancementPartie"/>

  <target name="clean"
        description="clean up" >
    <!-- Delete the ${build} and ${dist} directory trees -->
    <delete dir="${jar}"/>
    <delete dir="${class}"/>
    <delete dir="${javadoc}"/>
    <echo message="Files cleared..."/>
  </target>

  <target name="init" depends="clean"
     description="create all directories" >
    <!-- Create the build directory structure used by compile -->
    <mkdir dir="${jar}"/>
    <mkdir dir="${class}"/>
```

```
    <mkdir dir="${javadoc}"/>
    <echo message="Folder created..."/>
  </target>

  <target name="compile" depends="init"
        description="compile all files" >
    <!-- Compile the java code from ${src} into ${build} -->
    <javac srcdir="${src}" destdir="${class}"/>
    <javac srcdir="${src}/arcanor" destdir="${class}"/>
    <javac srcdir="${src}/tests" destdir="${class}"/>
    <echo message="Classes compiled..."/>
  </target>

  <target name="javadoc" depends="compile"
        description="generate the JavaDoc" >
    <!-- Create the javaDoc of all classes -->
    <javadoc charset="UTF8" access="private" destdir="${javadoc}"
sourcepath="${src}"/>
    <echo message="Javadoc created..."/>
  </target>

  <target name="jar" depends="compile">
    <jar destfile="${jar}.jar" basedir="${class}">
       <manifest>
             <attribute name="Main-Class" value="${main-class}"/>
       </manifest>
    </jar>
    <echo message="Jar created..."/>
  </target>
</project>
```