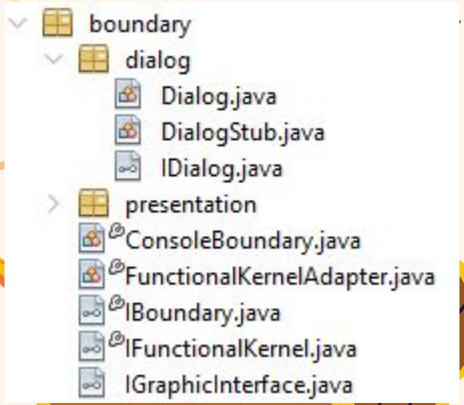
Call of Rum - partie IHM

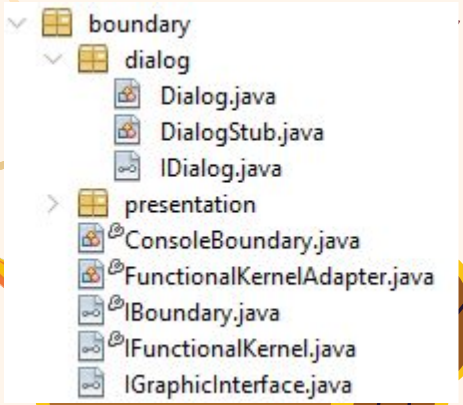
Projet ILU4

Présentation Architecture

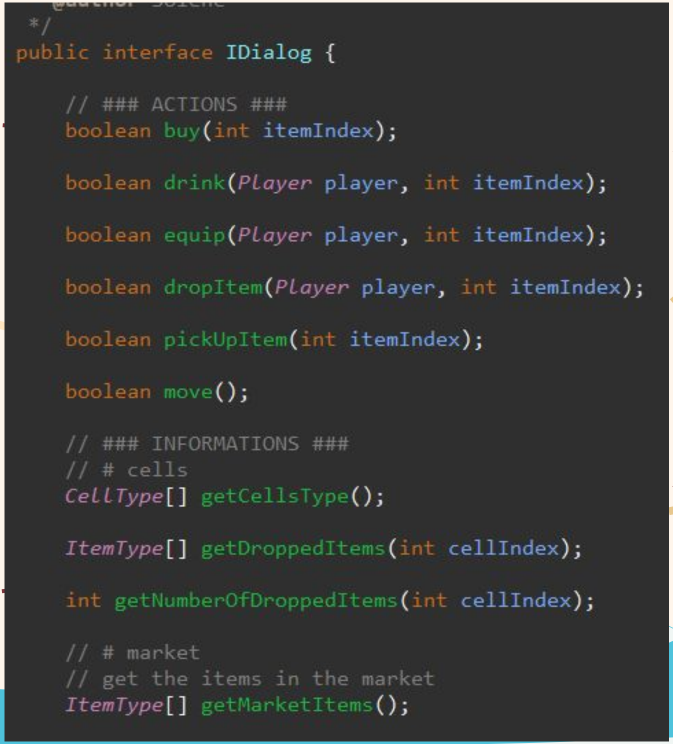


```
boundary
  dialog
    Dialog.java
    DialogStub.java
    IDialog.java
  presentation
  ConsoleBoundary.java
  FunctionalKernelAdapter.java
  IBoundary.java
  IFunctionalKernel.java
  IGraphicInterface.java
```

Présentation Architecture

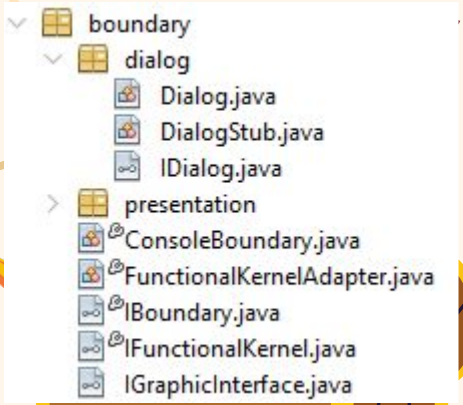


```
boundary
    dialog
        Dialog.java
        DialogStub.java
        IDialog.java
    presentation
    ConsoleBoundary.java
    FunctionalKernelAdapter.java
    IBoundary.java
    IFunctionalKernel.java
    IGraphicInterface.java
```

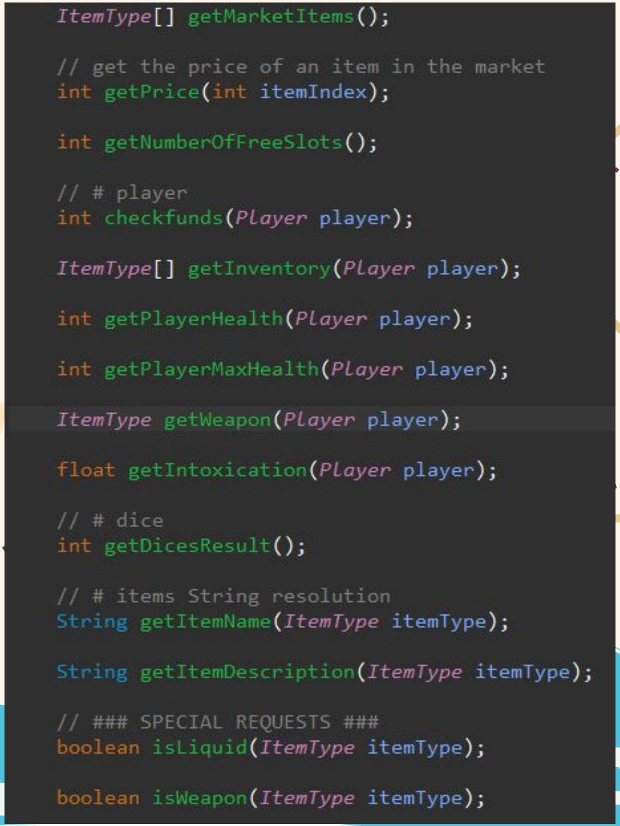


```java
 */
public interface IDialog {

    // ### ACTIONS ###
    boolean buy(int itemIndex);

    boolean drink(Player player, int itemIndex);

    boolean equip(Player player, int itemIndex);

    boolean dropItem(Player player, int itemIndex);

    boolean pickUpItem(int itemIndex);

    boolean move();

    // ### INFORMATIONS ###
    // # cells
    CellType[] getCellsType();

    ItemType[] getDroppedItems(int cellIndex);

    int getNumberOfDroppedItems(int cellIndex);

    // # market
    // get the items in the market
    ItemType[] getMarketItems();
```

Présentation Architecture

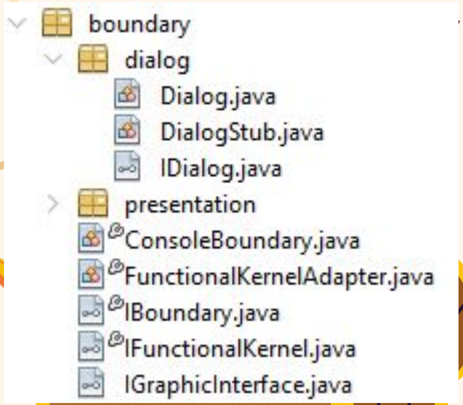


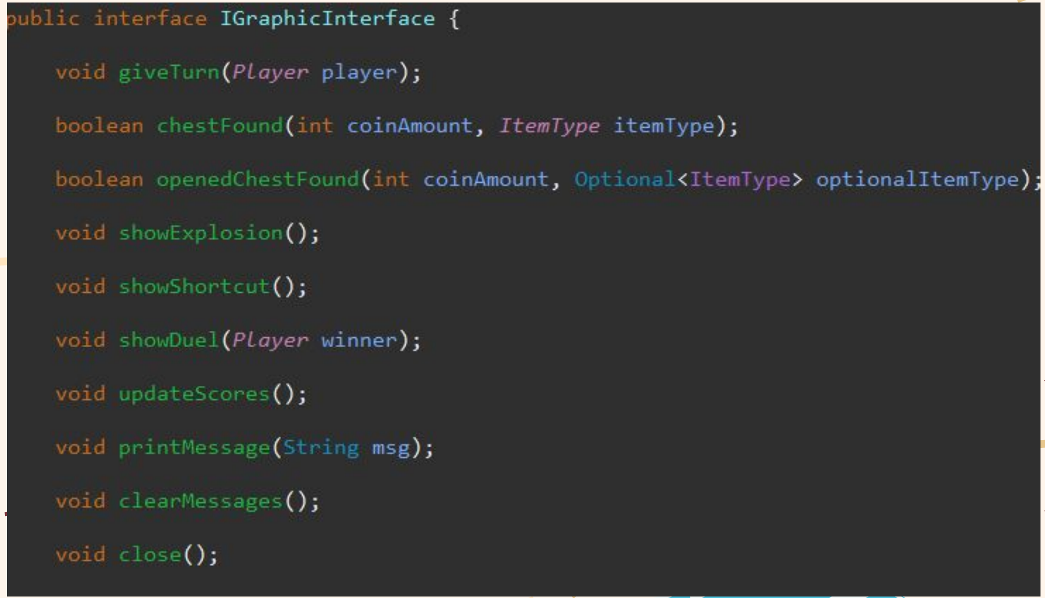
```
boundary
  dialog
    Dialog.java
    DialogStub.java
    IDialog.java
  presentation
  ConsoleBoundary.java
  FunctionalKernelAdapter.java
  IBoundary.java
  IFunctionalKernel.java
  IGraphicInterface.java
```



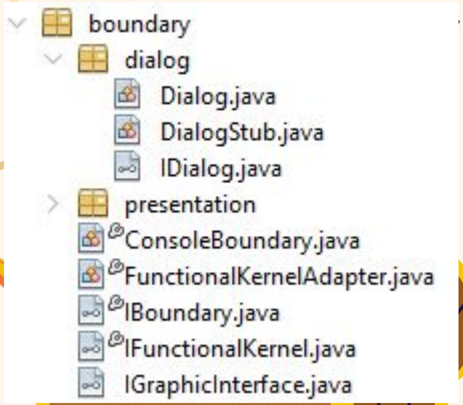
```
ItemType[] getMarketItems();

// get the price of an item in the market
int getPrice(int itemIndex);

int getNumberOfFreeSlots();

// # player
int checkfunds(Player player);

ItemType[] getInventory(Player player);

int getPlayerHealth(Player player);

int getPlayerMaxHealth(Player player);

ItemType getWeapon(Player player);

float getIntoxication(Player player);

// # dice
int getDicesResult();

// # items String resolution
String getItemName(ItemType itemType);

String getItemDescription(ItemType itemType);

// ### SPECIAL REQUESTS ###
boolean isLiquid(ItemType itemType);

boolean isWeapon(ItemType itemType);
```

Présentation Architecture

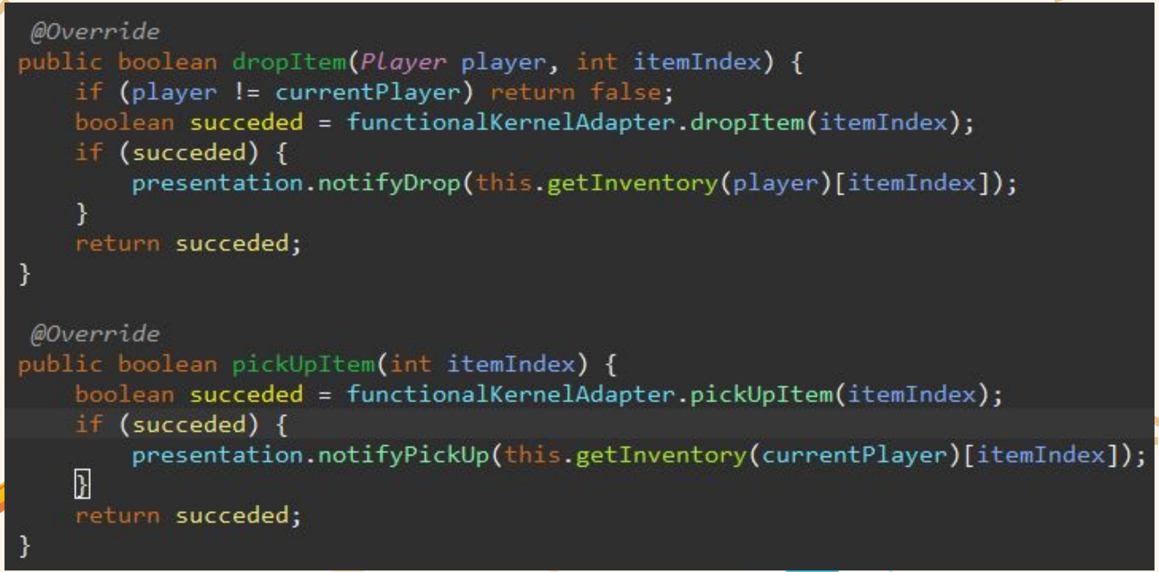

```
boundary
  dialog
    Dialog.java
    DialogStub.java
    IDialog.java
  presentation
  ConsoleBoundary.java
  FunctionalKernelAdapter.java
  IBoundary.java
  IFunctionalKernel.java
  IGraphicInterface.java
```


```java
public interface IGraphicInterface {

    void giveTurn(Player player);

    boolean chestFound(int coinAmount, ItemType itemType);

    boolean openedChestFound(int coinAmount, Optional<ItemType> optionalItemType);

    void showExplosion();

    void showShortcut();

    void showDuel(Player winner);

    void updateScores();

    void printMessage(String msg);

    void clearMessages();

    void close();
```

Présentation Architecture

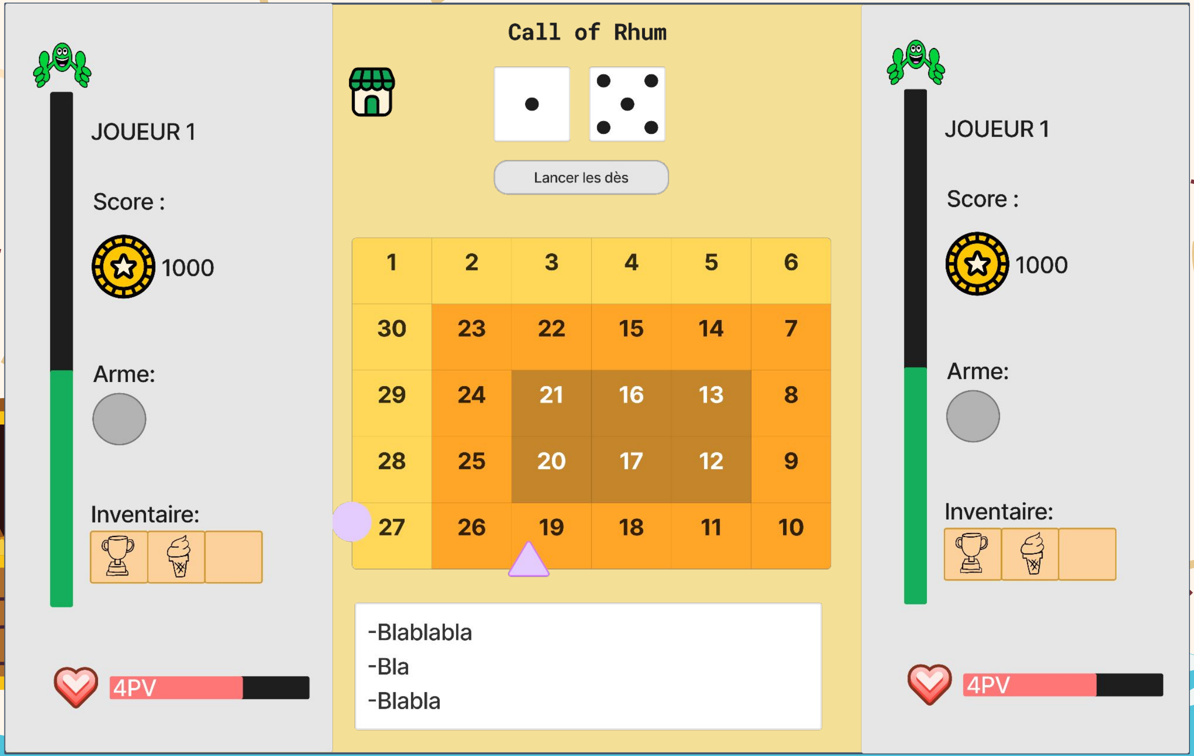


```
boundary
    dialog
        Dialog.java
        DialogStub.java
        IDialog.java
    presentation
    ConsoleBoundary.java
    FunctionalKernelAdapter.java
    IBoundary.java
    IFunctionalKernel.java
    IGraphicInterface.java
```



```java
@Override
public boolean dropItem(Player player, int itemIndex) {
    if (player != currentPlayer) return false;
    boolean succeded = functionalKernelAdapter.dropItem(itemIndex);
    if (succeded) {
        presentation.notifyDrop(this.getInventory(player)[itemIndex]);
    }
    return succeded;
}

@Override
public boolean pickUpItem(int itemIndex) {
    boolean succeded = functionalKernelAdapter.pickUpItem(itemIndex);
    if (succeded) {
        presentation.notifyPickUp(this.getInventory(currentPlayer)[itemIndex]);
    }
    return succeded;
}
```

Présentation du Découpage

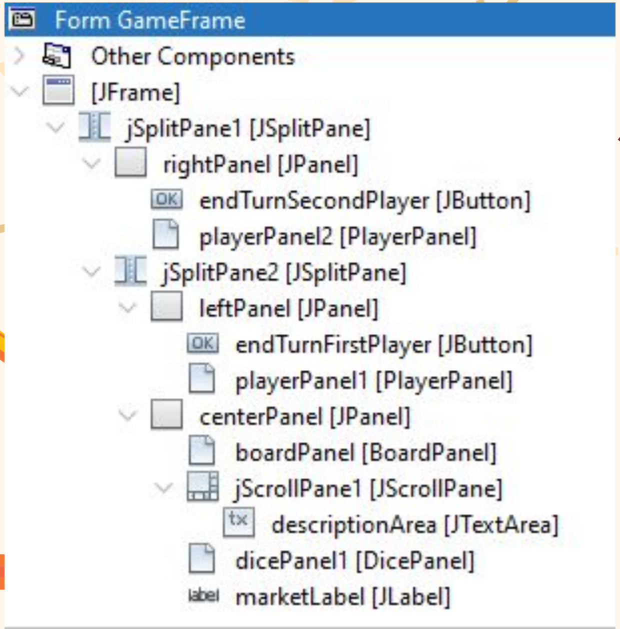


Call of Rhum

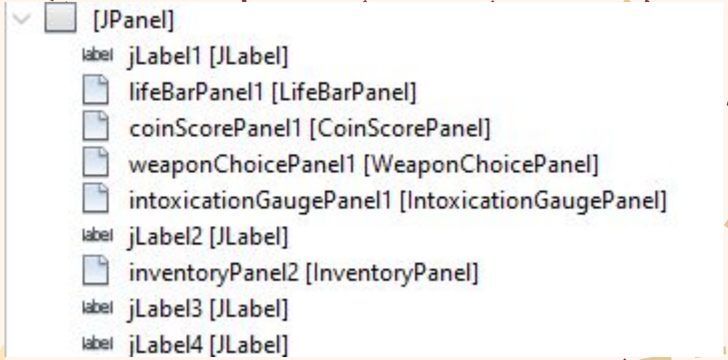
1	2	3	4	5	6
30	23	22	15	14	7
29	24	21	16	13	8
28	25	20	17	12	9
27	26	19	18	11	10

-Blablabla
-Bla
-Blabla

Présentation du Découpage



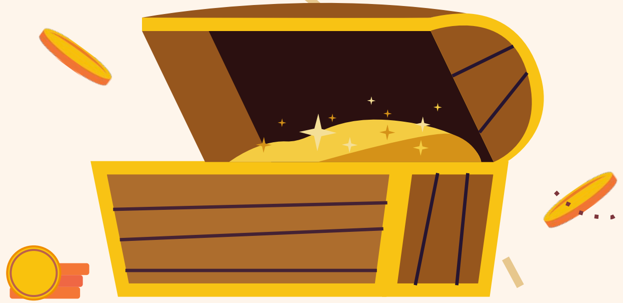
```
Form GameFrame
> Other Components
v [JFrame]
  v jSplitPane1 [JSplitPane]
    v rightPanel [JPanel]
        endTurnSecondPlayer [JButton]
        playerPanel2 [PlayerPanel]
    v jSplitPane2 [JSplitPane]
      v leftPanel [JPanel]
          endTurnFirstPlayer [JButton]
          playerPanel1 [PlayerPanel]
      v centerPanel [JPanel]
          boardPanel [BoardPanel]
        v jScrollPane1 [JScrollPane]
            descriptionArea [JTextArea]
          dicePanel1 [DicePanel]
          marketLabel [JLabel]
```



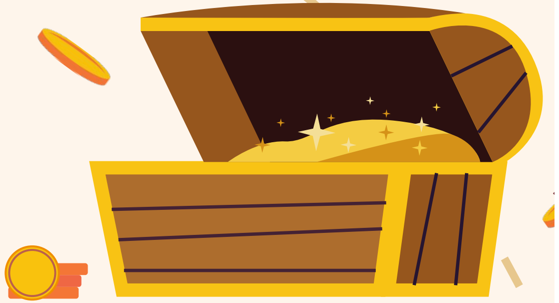
```
v [JPanel]
    jLabel1 [JLabel]
    lifeBarPanel1 [LifeBarPanel]
    coinScorePanel1 [CoinScorePanel]
    weaponChoicePanel1 [WeaponChoicePanel]
    intoxicationGaugePanel1 [IntoxicationGaugePanel]
    jLabel2 [JLabel]
    inventoryPanel2 [InventoryPanel]
    jLabel3 [JLabel]
    jLabel4 [JLabel]
```

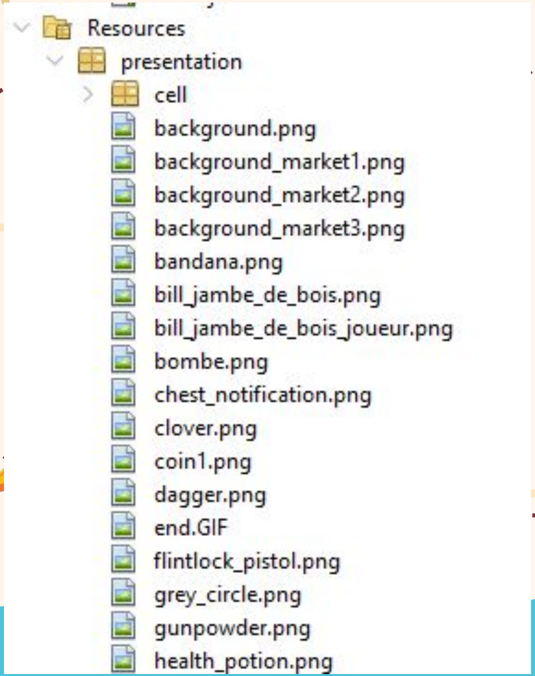
Mateo

- Chargement d'Images
- Jauge d'intoxication
- Dialogue de Coffre



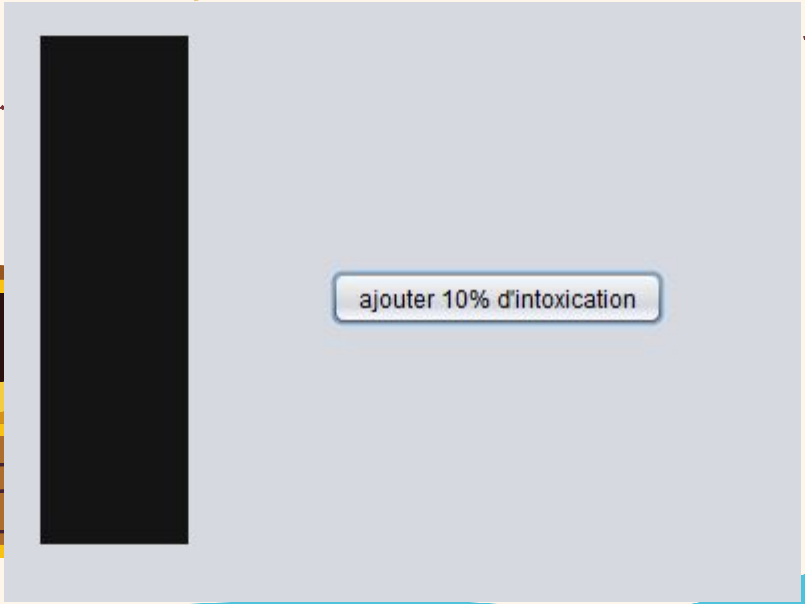
Mateo - Chargement d'image



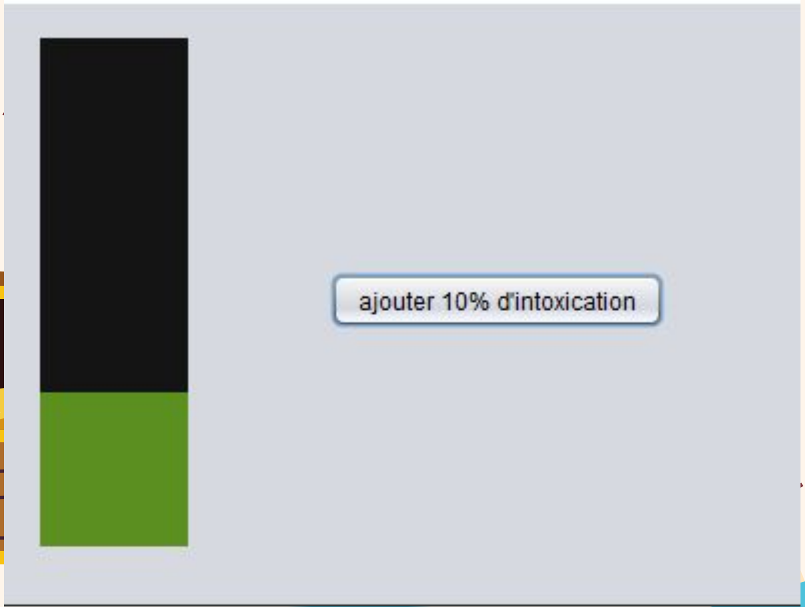


```
Resources
  presentation
    cell
    background.png
    background_market1.png
    background_market2.png
    background_market3.png
    bandana.png
    bill_jambe_de_bois.png
    bill_jambe_de_bois_joueur.png
    bombe.png
    chest_notification.png
    clover.png
    coin1.png
    dagger.png
    end.GIF
    flintlock_pistol.png
    grey_circle.png
    gunpowder.png
    health_potion.png
```

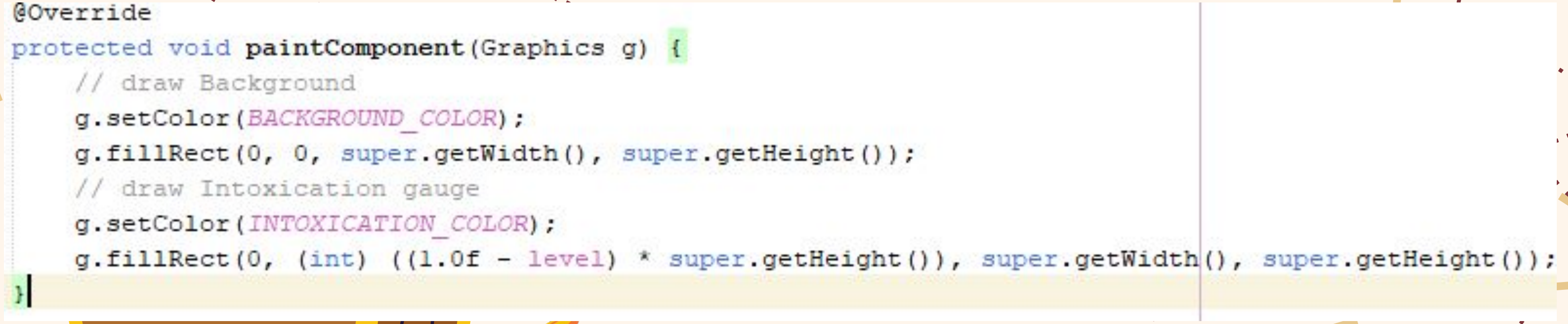
Mateo - Jauge



Mateo - Jauge

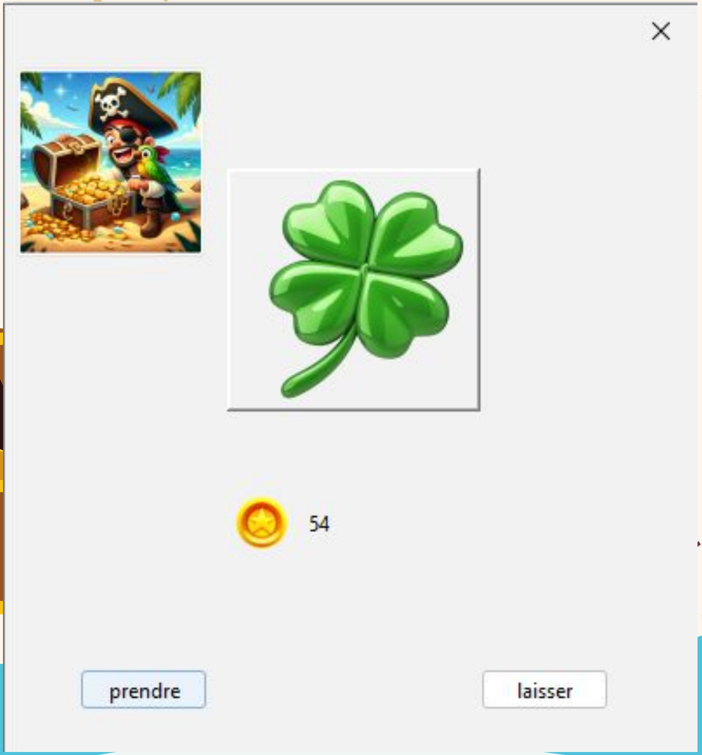


Mateo - Jauge

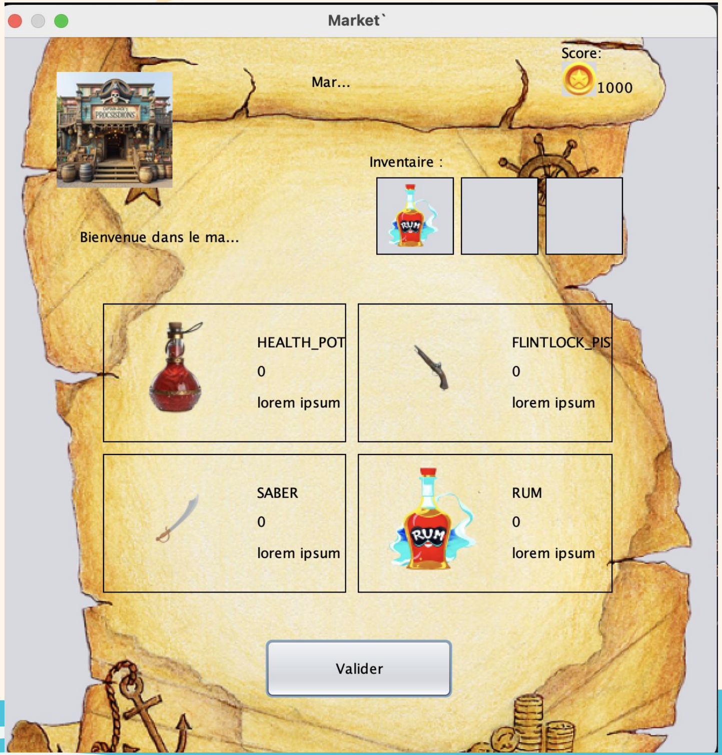


```
@Override
protected void paintComponent(Graphics g) {
    // draw Background
    g.setColor(BACKGROUND_COLOR);
    g.fillRect(0, 0, super.getWidth(), super.getHeight());
    // draw Intoxication gauge
    g.setColor(INTOXICATION_COLOR);
    g.fillRect(0, (int) ((1.0f - level) * super.getHeight()), super.getWidth(), super.getHeight());
}
```

Mateo - Dialogue coffre



Assane



Market`

Mar...

Score:

1000

Inventaire :

Bienvenue dans le ma...

HEALTH_POT
0
lorem ipsum

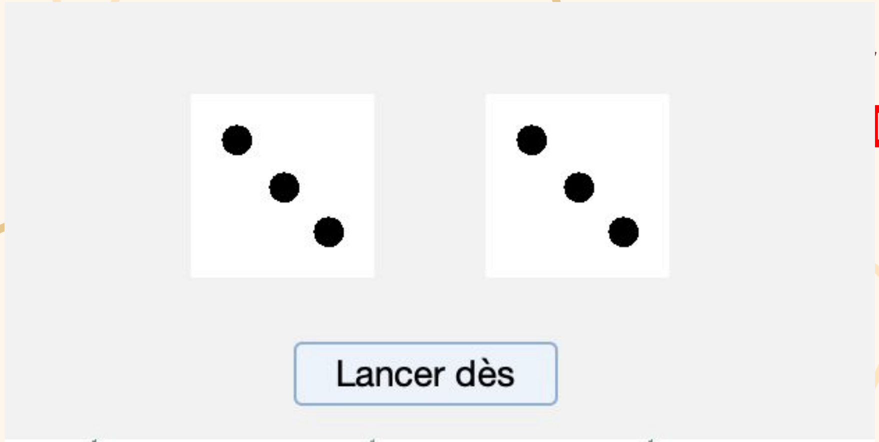
FLINTLOCK_PIS
0
lorem ipsum

SABER
0
lorem ipsum

RUM
0
lorem ipsum

-Market

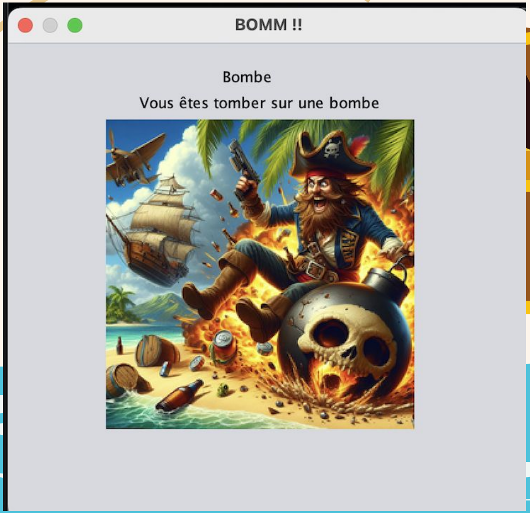
Assane



Dice -> animation des dès

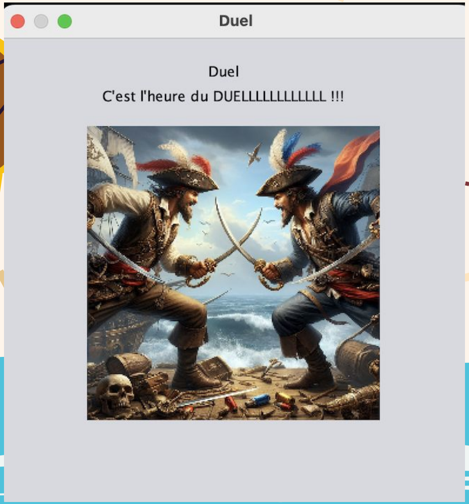
Assane

- -Dialogue raccourci
- -Dialogue bombe
- -Dialogue duel


BOMM !!

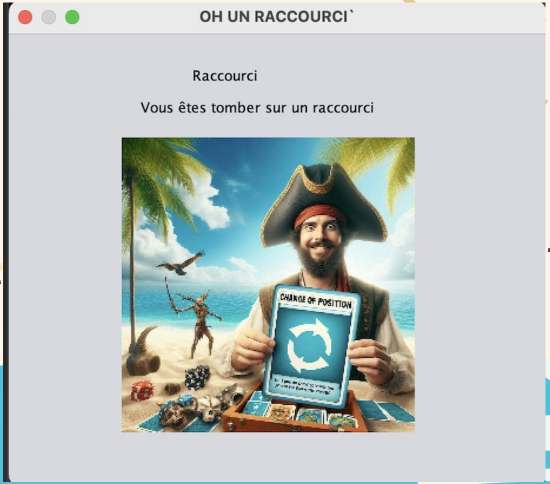
Bombe

Vous êtes tomber sur une bombe


Duel

Duel

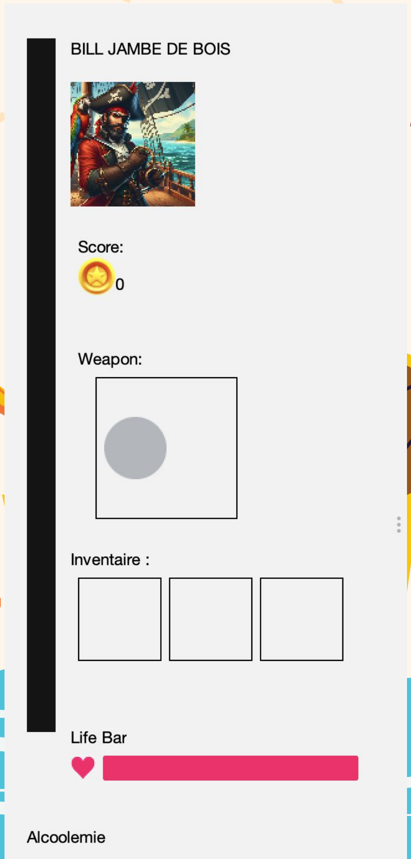
C'est l'heure du DUELLLLLLLLLLLL !!!


OH UN RACCOURCI`

Raccourci

Vous êtes tomber sur un raccourci

Assane


BILL JAMBE DE BOIS

Score:
0

Weapon:

Inventaire :

Life Bar

Alcoolemie

-player panel

Solène

-Image de fond

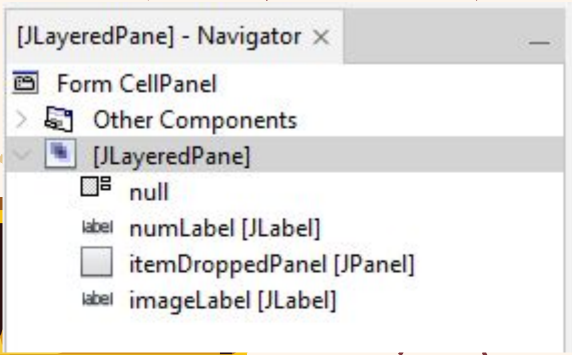
-Cases

-Items sur les cases

-Pions joueurs

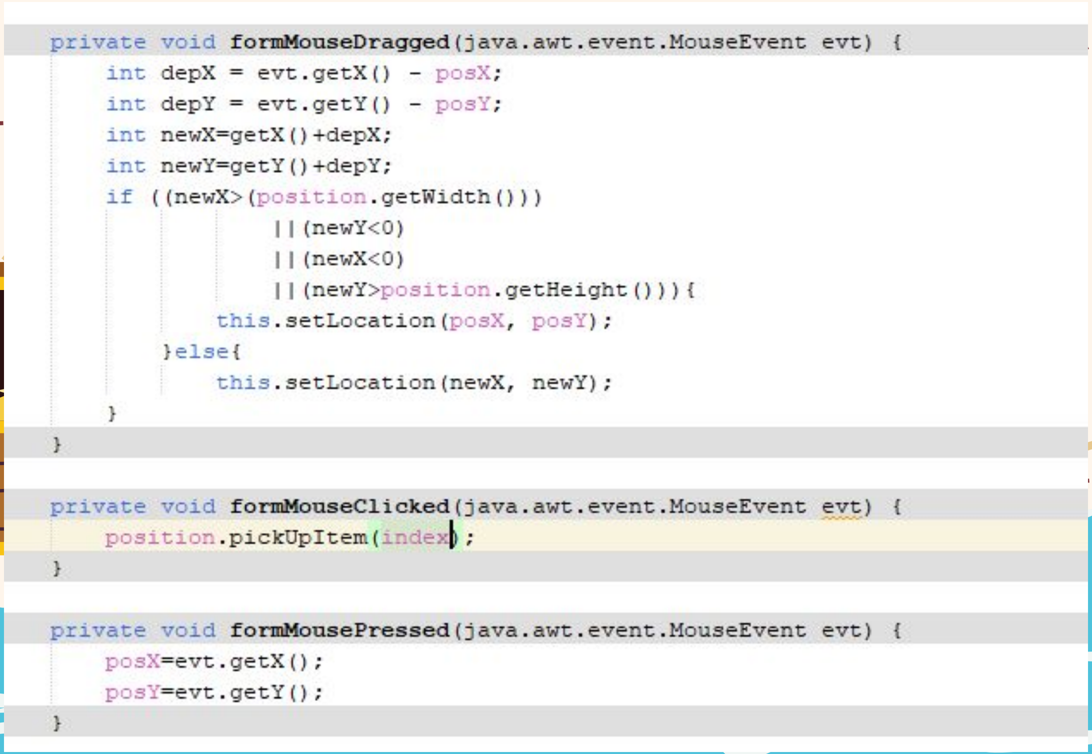
-Plateau

Solène - Cases



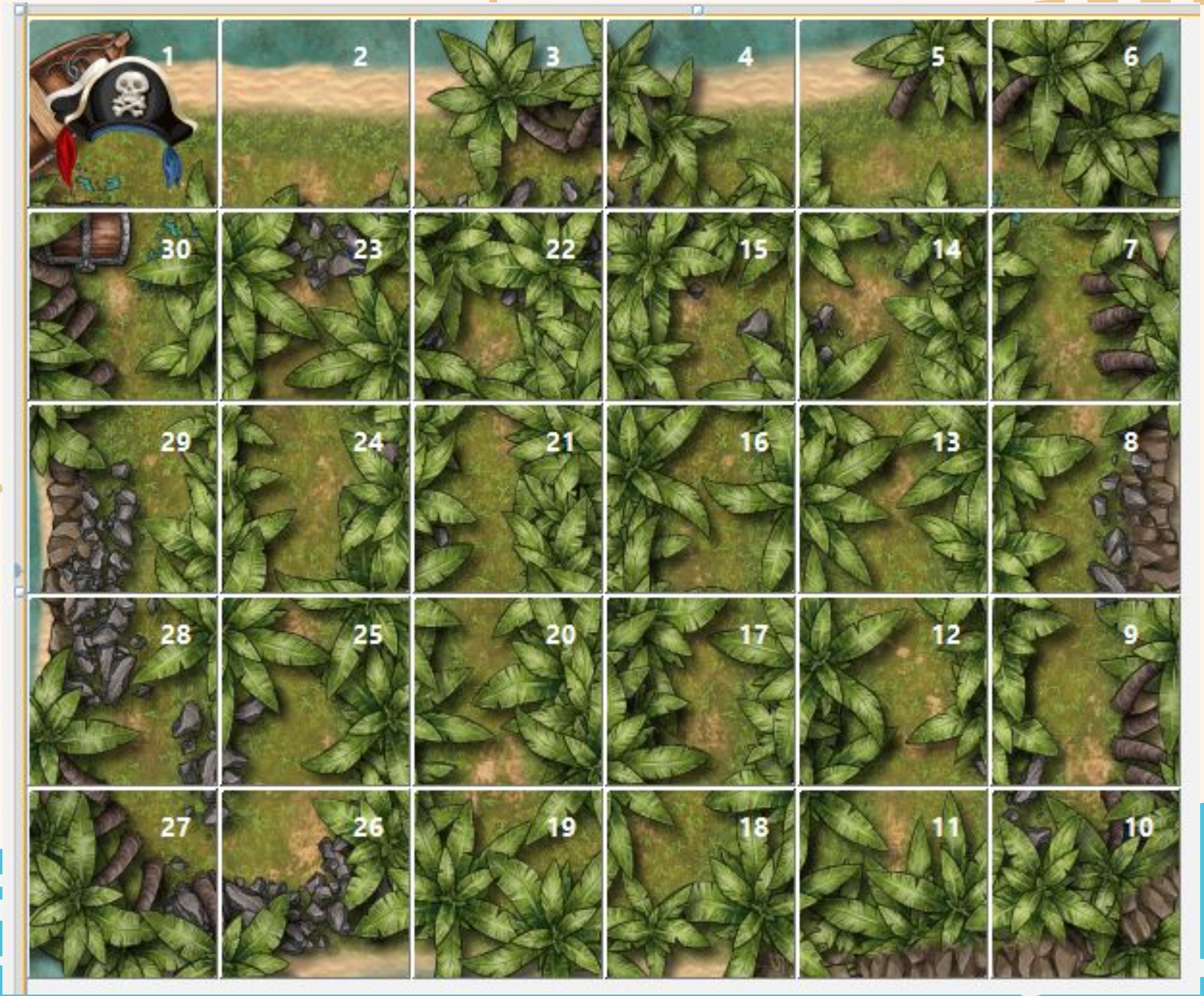
```
[JLayeredPane] - Navigator ×
Form CellPanel
  > Other Components
  v [JLayeredPane]
      null
      numLabel [JLabel]
      itemDroppedPanel [JPanel]
      imageLabel [JLabel]
```

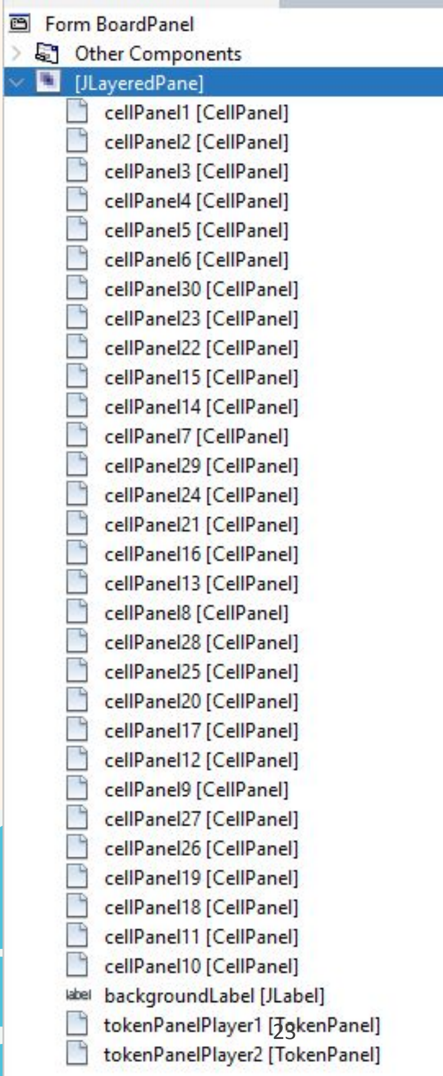
Solène - Items sur les cases (itemDrop)



```
private void formMouseDragged(java.awt.event.MouseEvent evt) {
    int depX = evt.getX() - posX;
    int depY = evt.getY() - posY;
    int newX=getX()+depX;
    int newY=getY()+depY;
    if ((newX>(position.getWidth()))
            ||(newY<0)
            ||(newX<0)
            ||(newY>position.getHeight())){
        this.setLocation(posX, posY);
    }else{
        this.setLocation(newX, newY);
    }
}

private void formMouseClicked(java.awt.event.MouseEvent evt) {
    position.pickUpItem(index);
}

private void formMousePressed(java.awt.event.MouseEvent evt) {
    posX=evt.getX();
    posY=evt.getY();
}
```

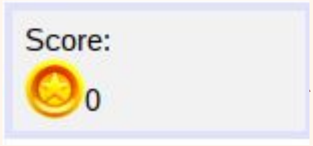
Solène - Pions joueurs




Form BoardPanel
- Other Components
- [JLayeredPane]
  - cellPanel1 [CellPanel]
  - cellPanel2 [CellPanel]
  - cellPanel3 [CellPanel]
  - cellPanel4 [CellPanel]
  - cellPanel5 [CellPanel]
  - cellPanel6 [CellPanel]
  - cellPanel30 [CellPanel]
  - cellPanel23 [CellPanel]
  - cellPanel22 [CellPanel]
  - cellPanel15 [CellPanel]
  - cellPanel14 [CellPanel]
  - cellPanel7 [CellPanel]
  - cellPanel29 [CellPanel]
  - cellPanel24 [CellPanel]
  - cellPanel21 [CellPanel]
  - cellPanel16 [CellPanel]
  - cellPanel13 [CellPanel]
  - cellPanel8 [CellPanel]
  - cellPanel28 [CellPanel]
  - cellPanel25 [CellPanel]
  - cellPanel20 [CellPanel]
  - cellPanel17 [CellPanel]
  - cellPanel12 [CellPanel]
  - cellPanel9 [CellPanel]
  - cellPanel27 [CellPanel]
  - cellPanel26 [CellPanel]
  - cellPanel19 [CellPanel]
  - cellPanel18 [CellPanel]
  - cellPanel11 [CellPanel]
  - cellPanel10 [CellPanel]
  - backgroundLabel [JLabel]
  - tokenPanelPlayer1 [TokenPanel]
  - tokenPanelPlayer2 [TokenPanel]

Wilkens

- **Affichage du nombre de coins.**

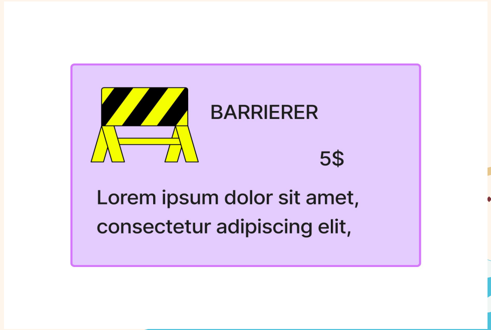


-setScore
-add

- **Dialogue de fin de partie.**

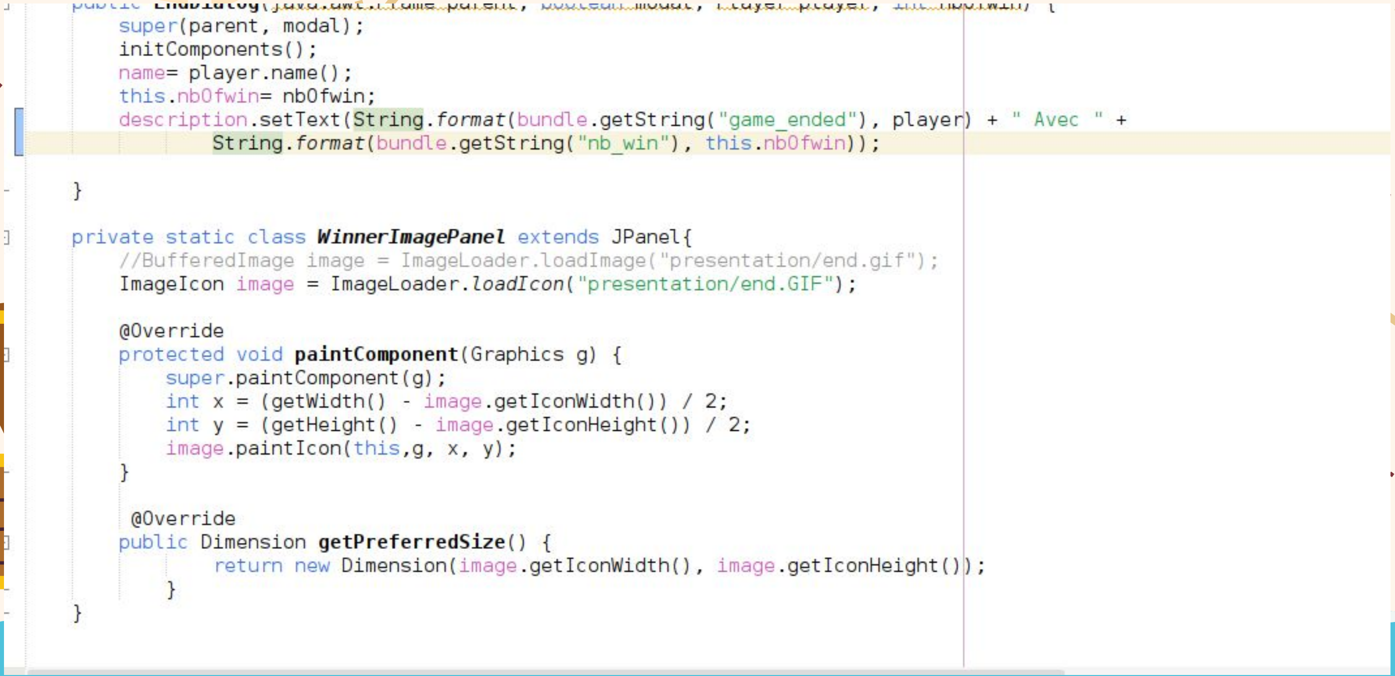
-paintIcon(t,g,h,w)
-Nom du gagnant.

- **Les cartes du jeu.**



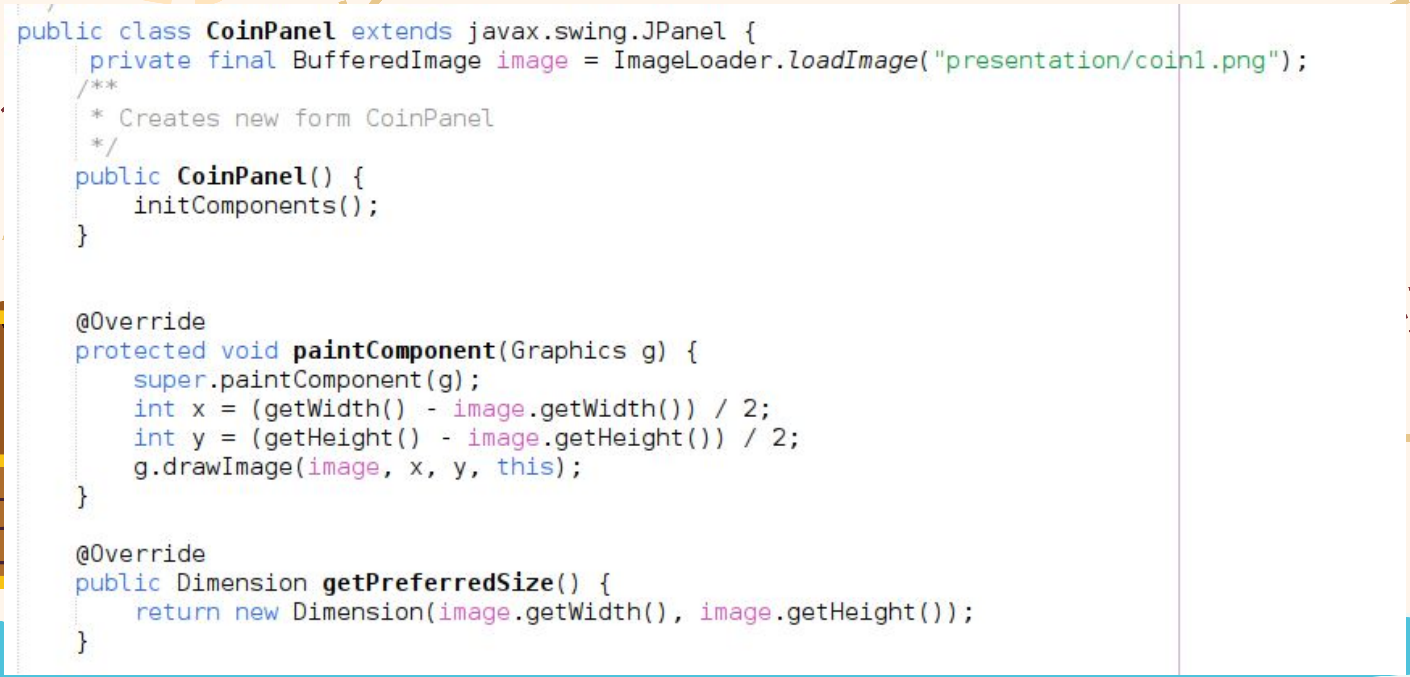
-Rhum
-Sabre
-Pistolet
-Fusil

Dialogue de fin de partie



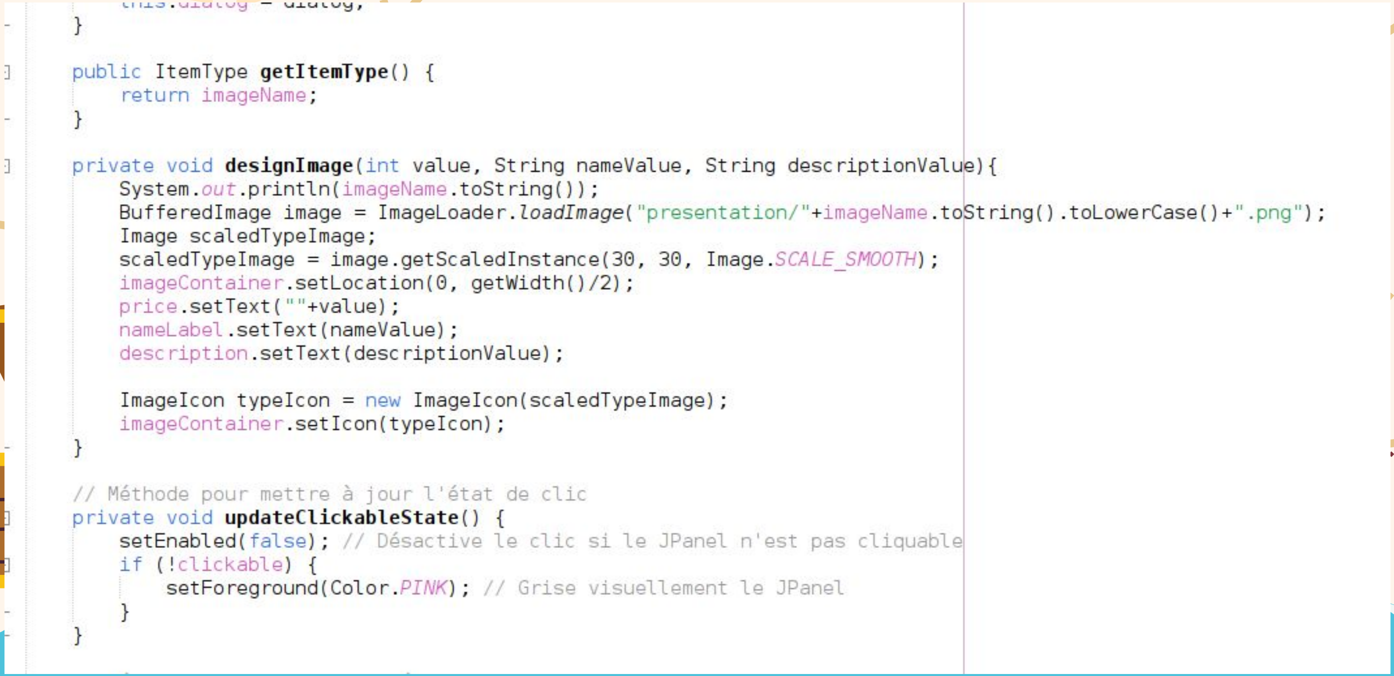
```java
    public EndDialog(java.awt.Frame parent, boolean modal, Player player, int nbOfwin) {
        super(parent, modal);
        initComponents();
        name= player.name();
        this.nbOfwin= nbOfwin;
        description.setText(String.format(bundle.getString("game_ended"), player) + " Avec " +
                String.format(bundle.getString("nb_win"), this.nbOfwin));

    }

    private static class WinnerImagePanel extends JPanel{
        //BufferedImage image = ImageLoader.loadImage("presentation/end.gif");
        ImageIcon image = ImageLoader.loadIcon("presentation/end.GIF");

        @Override
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            int x = (getWidth() - image.getIconWidth()) / 2;
            int y = (getHeight() - image.getIconHeight()) / 2;
            image.paintIcon(this,g, x, y);
        }

         @Override
        public Dimension getPreferredSize() {
                return new Dimension(image.getIconWidth(), image.getIconHeight());
            }
    }
```

Affichage du nombre de coins



```
public class CoinPanel extends javax.swing.JPanel {
    private final BufferedImage image = ImageLoader.loadImage("presentation/coin1.png");
    /**
     * Creates new form CoinPanel
     */
    public CoinPanel() {
        initComponents();
    }


    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        int x = (getWidth() - image.getWidth()) / 2;
        int y = (getHeight() - image.getHeight()) / 2;
        g.drawImage(image, x, y, this);
    }

    @Override
    public Dimension getPreferredSize() {
        return new Dimension(image.getWidth(), image.getHeight());
    }
```

Création des cartes du jeu

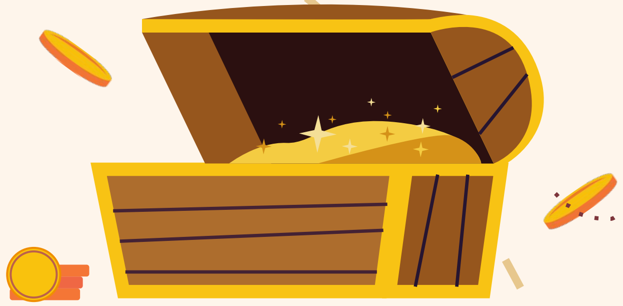


```java
        this.dialog = dialog;
    }

    public ItemType getItemType() {
        return imageName;
    }

    private void designImage(int value, String nameValue, String descriptionValue){
        System.out.println(imageName.toString());
        BufferedImage image = ImageLoader.loadImage("presentation/"+imageName.toString().toLowerCase()+".png");
        Image scaledTypeImage;
        scaledTypeImage = image.getScaledInstance(30, 30, Image.SCALE_SMOOTH);
        imageContainer.setLocation(0, getWidth()/2);
        price.setText(""+value);
        nameLabel.setText(nameValue);
        description.setText(descriptionValue);

        ImageIcon typeIcon = new ImageIcon(scaledTypeImage);
        imageContainer.setIcon(typeIcon);
    }

    // Méthode pour mettre à jour l'état de clic
    private void updateClickableState() {
        setEnabled(false); // Désactive le clic si le JPanel n'est pas cliquable
        if (!clickable) {
            setForeground(Color.PINK); // Grise visuellement le JPanel
        }
    }
```

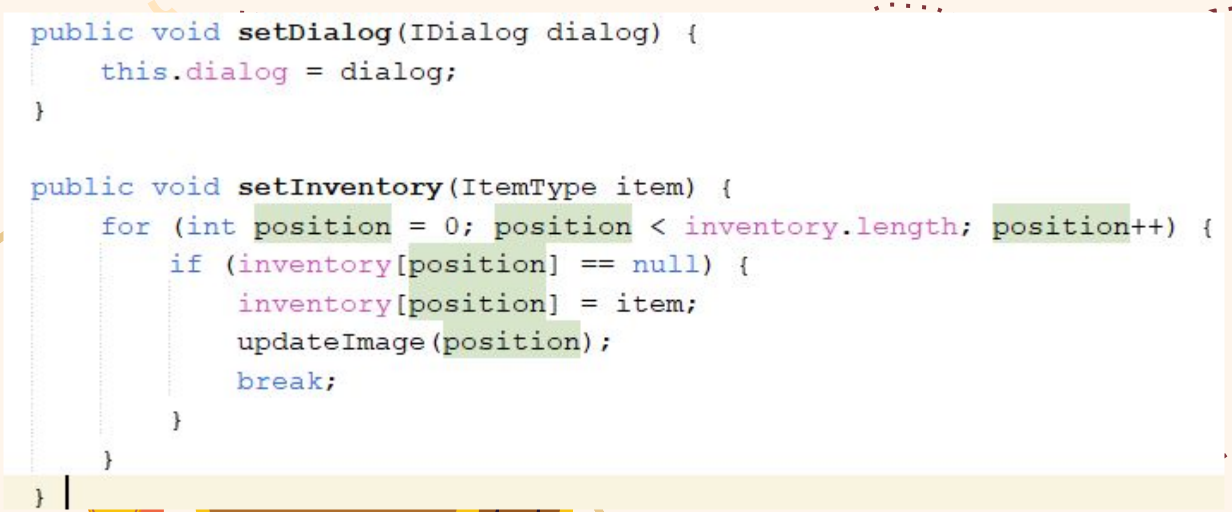
Jérémy

-Inventaire

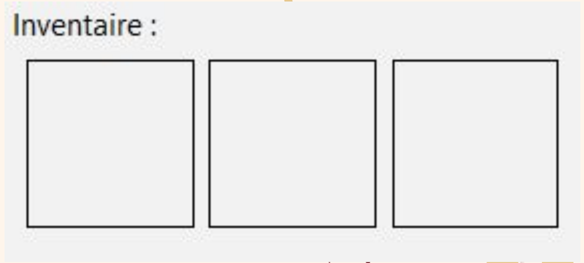
-MessageBox



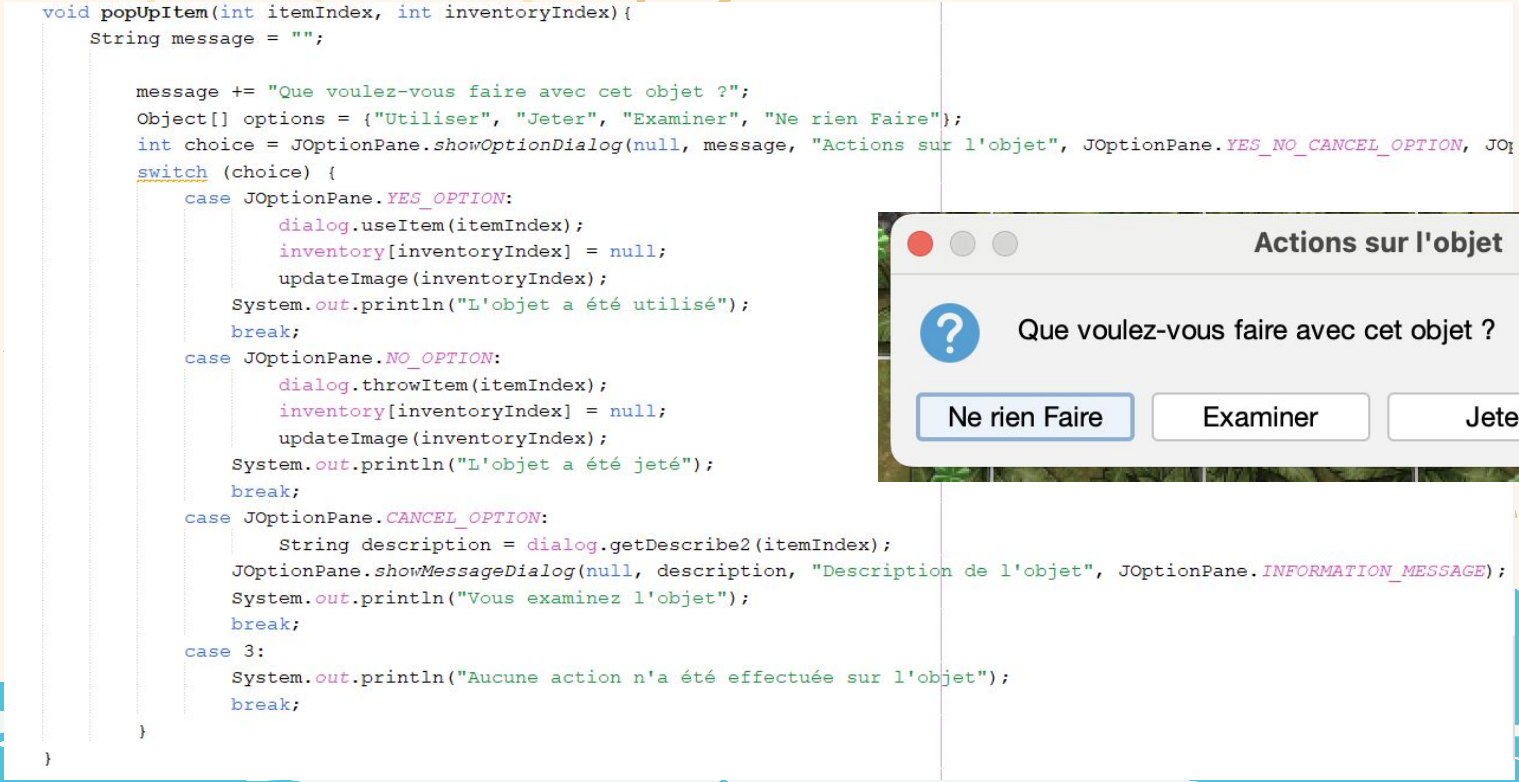
Jérémy - Inventaire



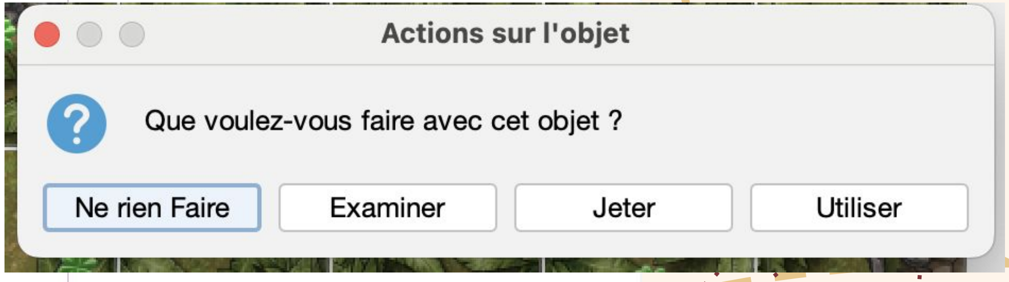
```
public void setDialog(IDialog dialog) {
    this.dialog = dialog;
}

public void setInventory(ItemType item) {
    for (int position = 0; position < inventory.length; position++) {
        if (inventory[position] == null) {
            inventory[position] = item;
            updateImage(position);
            break;
        }
    }
}
```



Jérémy - Inventaire



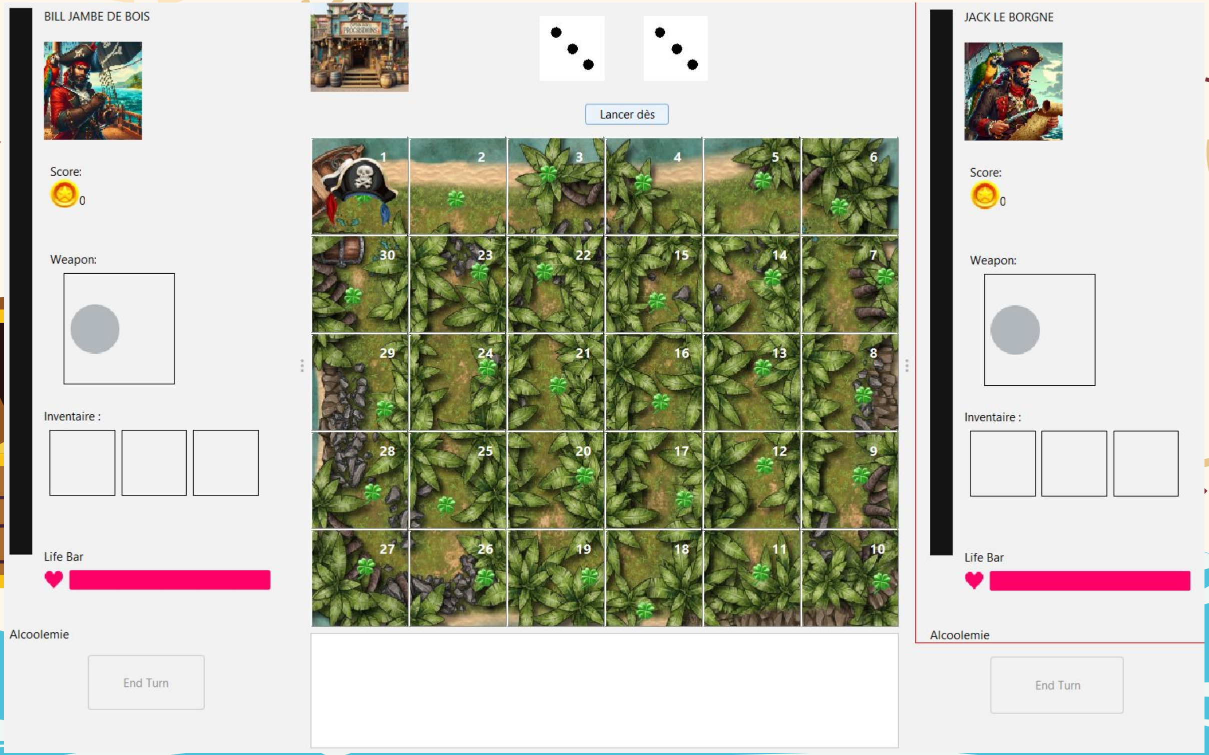
```java
void popUpItem(int itemIndex, int inventoryIndex){
    String message = "";

        message += "Que voulez-vous faire avec cet objet ?";
        Object[] options = {"Utiliser", "Jeter", "Examiner", "Ne rien Faire"};
        int choice = JOptionPane.showOptionDialog(null, message, "Actions sur l'objet", JOptionPane.YES_NO_CANCEL_OPTION, JOp
        switch (choice) {
            case JOptionPane.YES_OPTION:
                    dialog.useItem(itemIndex);
                    inventory[inventoryIndex] = null;
                    updateImage(inventoryIndex);
                System.out.println("L'objet a été utilisé");
                break;
            case JOptionPane.NO_OPTION:
                    dialog.throwItem(itemIndex);
                    inventory[inventoryIndex] = null;
                    updateImage(inventoryIndex);
                System.out.println("L'objet a été jeté");
                break;
            case JOptionPane.CANCEL_OPTION:
                    String description = dialog.getDescribe2(itemIndex);
                JOptionPane.showMessageDialog(null, description, "Description de l'objet", JOptionPane.INFORMATION_MESSAGE);
                System.out.println("Vous examinez l'objet");
                break;
            case 3:
                System.out.println("Aucune action n'a été effectuée sur l'objet");
                break;
        }
}
```



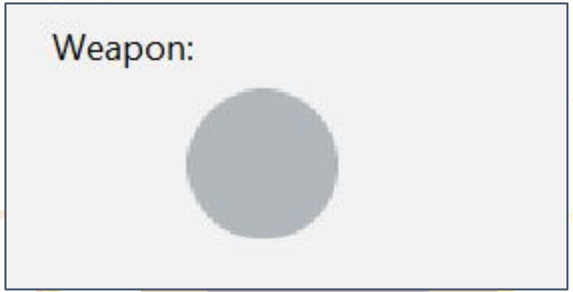
Actions sur l'objet

Que voulez-vous faire avec cet objet ?

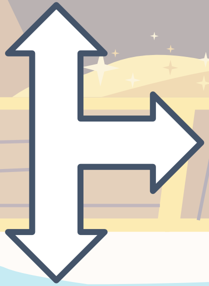
Jérémy - MessageBox

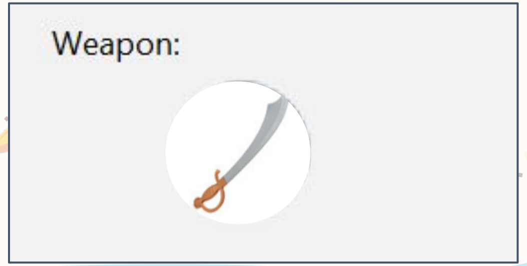


Alina

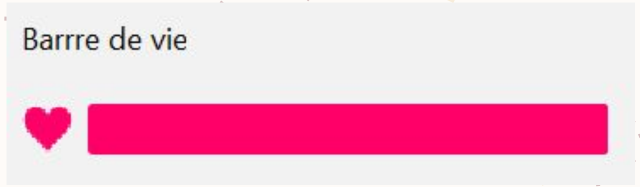

Weapon:

-Arme




Weapon:

-Barre de vie


Barrre de vie

Conclusion