

## Saé 2.01 – Développement d'une application

### Chifoumi – Dossier d'Analyse et conception

---

#### Table des matières

1. Compléments de spécifications externes.....	2
2. Diagramme des Cas d'Utilisation .....	2
3. Scénarios .....	2
4. Diagramme de classe (UML) .....	2
Version v0.....	6
5. Implémentation et tests.....	6
Version v1.....	7
6. Classe Chifoumi : Diagramme états-transitions .....	7
(b) Dictionnaires des états, événements et Actions.....	8
7. Éléments d'interface .....	8
8. Implémentation et tests.....	9
Version v2.....	10
9. Implémentation et tests.....	10
Version 3.....	12
10. Implémentation et tests.....	12

## 1. Compléments de spécifications externes.

On précise **uniquement** les points qui vous ont semblé flous ou bien incomplets. Rien de plus à signaler dans cette étude.

## 2. Diagramme des Cas d'Utilisation

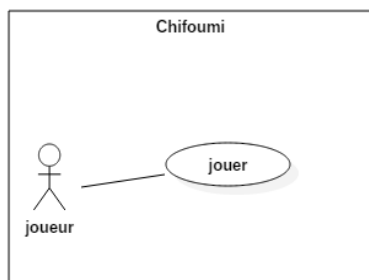


Figure 1 : Diagramme des Cas d'Utilisation du jeu Chifoumi

## 3. Scénarios

### (a) Exemple Scénario

Cas d'utilisation	JOUER	
Résumé	Le joueur joue une partie.	
Acteur primaire	Joueur	
Système	Chifoumi	
Intervenants		
Niveau	Objectif utilisateur	
Préconditions	Le jeu est démarré et se trouve à l'état initial.	
Postconditions		
Date de création		
Date de mise à jour		
Créateur		
Opérations	Joueur	Système
1	Démarre une nouvelle partie.	
2		Rend les figures actives et les affiche actives.
3	Choisit une figure.	
4		Affiche la figure du joueur dans la zone d'affichage du dernier coup joueur.
5		Choisit une figure.
6		Affiche sa figure dans la zone d'affichage de son dernier coup.
7		Détermine le gagnant et met à jour les scores.
8		Affiche les scores. Retour à l'étape 3.
Extension		
3.A	Le joueur demande à jouer une nouvelle partie.	
3.A.1	Choisit une nouvelle partie	
3.A.2		Réinitialise les scores.
3.A.3		Réinitialise les zones d'affichage des derniers coups.
3.A.4		Retour à l'étape 3.

Tableau 1 :  
Scénario  
nominal

### (b) Remarques :

- *Le scénario est très simple.*
- *L'objectif est de mettre en évidence les actions de l'utilisateur, celles du système, sachant que ces actions sont candidates à devenir des méthodes du système*

## 4. Diagramme de classe (UML)

(a) Le diagramme de classes UML du jeu se focalise sur les classes **métier**, cad celles décrivant le jeu

indépendamment des éléments d’interface que comportera le programme.

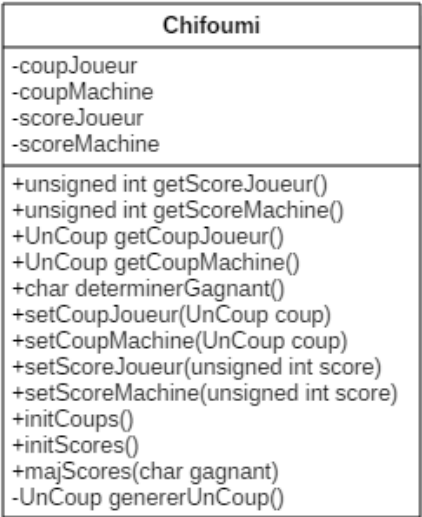


Figure 2 : Diagramme de Classes UML du jeu Chifoumi

(b) Dictionnaire des éléments de la **Classe Chifoumi**

Nom attribut	Signification	Type	Exemple
scoreJoueur	Nbre total de points acquis par le joueur durant la partie courante	unsigned int	1
scoreMachine	Nbre total de points acquis par la machine durant la partie courante	unsigned int	1
coupJoueur	Mémoire la dernière figure choisie par le joueur. Type énuméré enum unCoup {pierre, ciseau, papier, rien};	UnCoup	papier
coupMachine	Mémoire la dernière figure choisie par la machine.	UnCoup	Ciseau

Tableau 2 : Dictionnaire des éléments - Classe Chifoumi

**(c) Dictionnaire des méthodes : intégrées dans l'interface de la classe : cf Figure 3**

```
using namespace std;
class Chifoumi
{
    ///  
---- PARTIE MODÈLE -----  

    ///  
Une définition de type énuméré  

public:  
    enum UnCoup {pierre, papier, ciseau, rien};  

    ///  
Méthodes publiques du Modèle  

public:  
    Chifoumi();  
    virtual ~Chifoumi();  

    // Getters  

    UnCoup getCoupJoueur();  
        /* retourne le dernier coup joué par le joueur */  

    UnCoup getCoupMachine();  
        /* retourne le dernier coup joué par le joueur */  

    unsigned int getScoreJoueur();  
        /* retourne le score du joueur */  

    unsigned int getScoreMachine();  
        /* retourne le score de la machine */  

    char determinerGagnant();  
        /* détermine le gagnant 'J' pour joueur, 'M' pour machine, 'N' pour match nul  
        en fonction du dernier coup joué par chacun d'eux */  

    ///  
Méthodes utilitaires du Modèle  

private :  
    UnCoup genererUnCoup();  

    /* retourne une valeur aléatoire = pierre, papier ou ciseau.  
    Utilisée pour faire jouer la machine */  

    // Setters  

public:  
    void setCoupJoueur(UnCoup p_coup);  
        /* initialise l'attribut coupJoueur avec la valeur  
        du paramètre p_coup */  

    void setCoupMachine(UnCoup p_coup);  
        /* initialise l'attribut coupMachine avec la valeur  
        du paramètre p_coup */  

    void setScoreJoueur(unsigned int p_score);  
        /* initialise l'attribut scoreJoueur avec la valeur  
        du paramètre p_score */  

    void setScoreMachine(unsigned int p_score);  
        /* initialise l'attribut coupMachine avec la valeur  
        du paramètre p_score */  

    // Autres modificateurs  

    void majScores(char p_gagnant);  
        /* met à jour le score du joueur ou de la machine ou aucun  
        en fonction des règles de gestion du jeu */  

    void initScores();  
        /* initialise à 0 les attributs scoreJoueur et scoreMachine  
        NON indispensable */  

    void initCoups();  
        /* initialise à rien les attributs coupJoueur et coupMachine  
        NON indispensable */  

    ///  
Attributs du Modèle  

private:  
    unsigned int scoreJoueur;    // score actuel du joueur  
    unsigned int scoreMachine;  // score actuel de la Machine  
    UnCoup coupJoueur;          // dernier coup joué par le joueur  
    UnCoup coupMachine;         // dernier coup joué par la machine  
};
```

Figure 3 : Schéma de classes = Une seule classe Chifoumi

**(d) Remarques concernant le schéma de classes**

1. On ne s'intéresse qu'aux attributs et méthodes métier. Notamment, on ne met pas, pour l'instant, ce qui relève de l'affichage car ce sont d'autres objets du programme (widgets) qui se chargeront de l'affichage. Par contre, on n'oublie pas les méthodes `getXXX()`, qui permettront aux objets métier de communiquer leur valeur aux objets graphiques pour que ceux-ci s'affichent.
2. On n'a mis ni le constructeur ni le destructeur, pour alléger le schéma.
3. D'autres attributs et méthodes viendront compléter cette vision ANALYTIQUE du jeu. Il s'agira des attributs et méthodes dits DE CONCEPTION nécessaires au développement de l'application.

## Version v0

### 5. Implémentation et tests

#### 5.1 Implémentation

Liste des fichiers de cette version :

- chifoumi.h : interface la classe Chifoumi
- chifoumi.cpp : Corps de la classe Chifoumi
- main.cpp : Moteur + tests de la classe
- acChifoumiAlvesJouve\_TP4.pdf : Document d'analyse

Respectivement spécification et corps de la classe Chifoumi décrite au paragraphe 4.

#### 5.2 Test

Test avec le programme fourni main.cpp

Valeurs fournies / attendues... comme montré dans la ressource R2.03 (partie tests)

Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
Pierre	Score joueur + 1 et score machine	Score joueur + 1 et score machine	ok
Pierre	Score joueur et score machine	Score joueur et score machine	ok
Pierre	Score joueur et score machine + 1	Score joueur et score machine + 1	ok
Papier	Score joueur + 1 et score machine	Score joueur + 1 et score machine	ok
Papier	Score joueur et score machine	Score joueur et score machine	ok
Papier	Score joueur et score machine + 1	Score joueur et score machine + 1	ok
Ciseau	Score joueur + 1 et score machine	Score joueur + 1 et score machine	ok
Ciseau	Score joueur et score machine	Score joueur et score machine	ok
Ciseau	Score joueur et score machine + 1	Score joueur et score machine + 1	ok

## Version v1

### 6. Classe Chifoumi : Diagramme états-transitions

#### (a) Diagramme états-transitions -actions du jeu

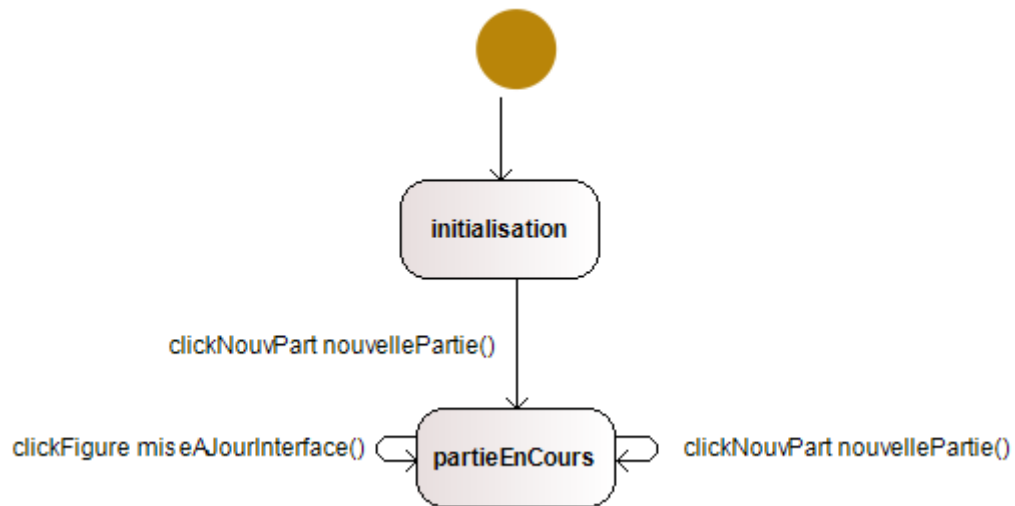


Figure 4 : Diagramme états-transitions

## (b) Dictionnaires des états, événements et Actions

### Dictionnaire des états du jeu

<i>nomEtat</i>	<i>Signification</i>
initialisation	État initial de la création du jeu
partieEnCours	La partie est en cours, le joueur sélectionne la figures pour jouer

Tableau 3 : États du jeu

### Dictionnaire des événements faisant changer le jeu d'état

<i>nomEvénement</i>	<i>Signification</i>
clickFigure	Le joueur clique sur une figure du Chifoumi
clickNouvPart	Le joueur clique sur le bouton de Nouvelle Partie (bNewGame)

Tableau 4 : Evénements faisant changer le jeu d'état

### Description des actions réalisées lors de la traversée des transitions

<b>miseAJourInterface()</b>	Mise à jour d'affichage par la figure jouée par le joueur et celle de l'ordinateur.
<b>nouvellePartie()</b>	Initialisation de la partie : met à jours les scores du joueur et de la machine à 0. Il réinitialise les figures jouées par une case vide.
<b>majScores ()</b>	Mise à jour des scores

Tableau 5 : Actions à réaliser lors des changements d'état

## (c) Préparation au codage :

**Table T\_EtatsEvenementsJeu** correspondant à la version matricielle du diagramme états-transitions du jeu :

- en ligne : les *événements* faisant changer le jeu d'état
- en colonne : les *états* du jeu

Événement → <i>nomEtatJeu</i>	clickFigure	clickNouvPart
initialisation	---	partieEnCours/nouvellePartie()
partieEnCours	partieEnCours/miseAJourInterface() + majScores()	partieEnCours/nouvellePartie()

Tableau 6 : Matrice d'états-transitions du jeu chifoumi

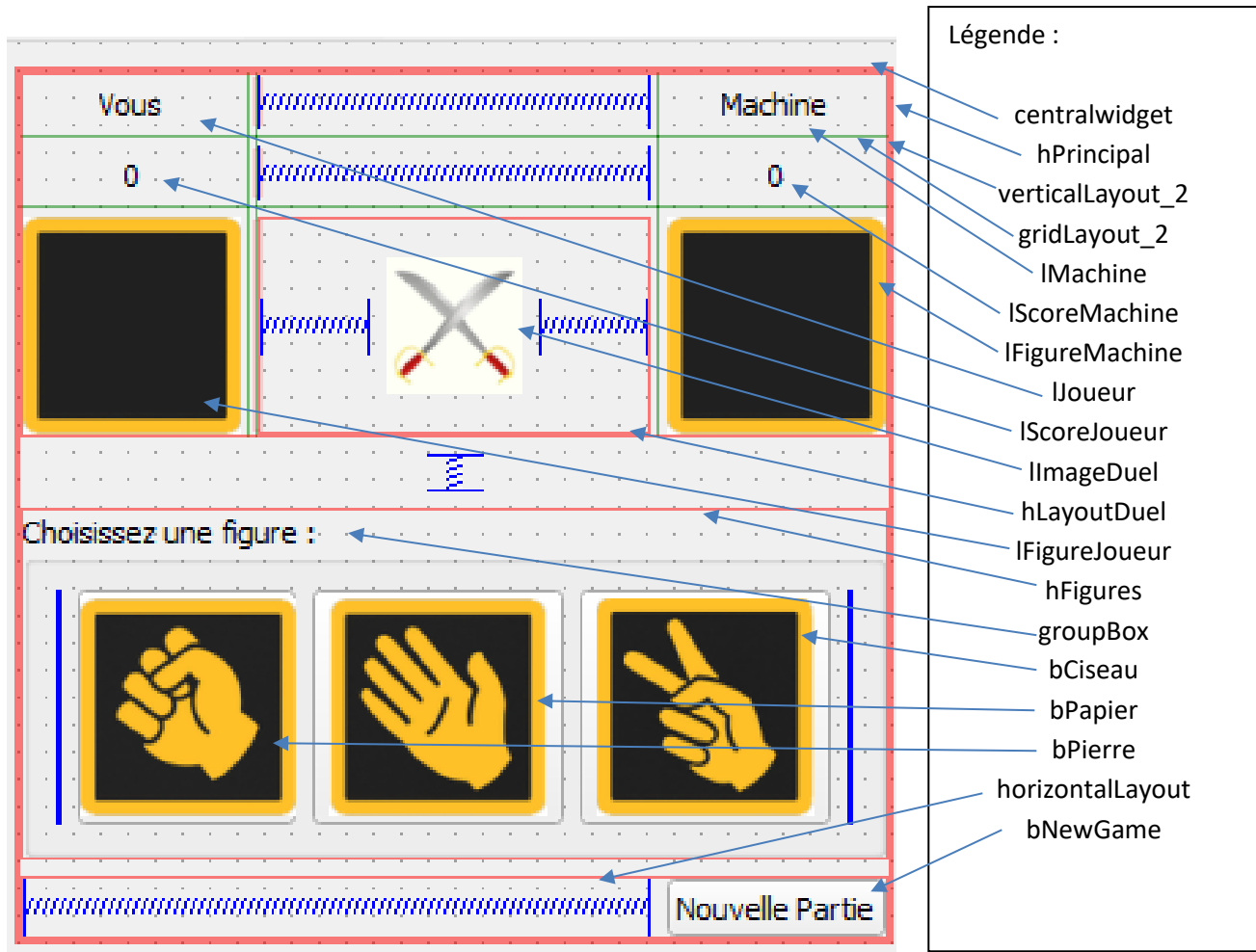
*L'intérêt de cette vue matricielle est qu'elle permet une préparation naturelle et aisée de l'étape suivante de programmation.*

## 7. Éléments d'interface

*A faire ici : description sommaire des éléments de l'interface, par exemple, avec une copie*



d'écran sur laquelle sont nommés les variables/objets graphiques et où les layouts sont positionnés et nommés.



## 8. Implémentation et tests

### 8.1 Implémentation

A faire :

Lister les fichiers impliqués dans cette version (répertoire, nom de fichier, rôle de chaque fichier)

Commenter brièvement les choix importants d'implémentation réalisés, comme par exemple, les signals/slots

- .gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaire à l'exécution du projet
- ChifoumiInterface.pro : fichier permettant l'ouverture et exécution du projet QT
- chifoumiVue.h : Module de la classe de la vue du chifoumiVue
- chifoumiVue.cpp : Corps de la classe chifoumiVue
- chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi
- main.cpp : Affichage de la fenêtre principale
- images : Répertoire contenant les images des figures du Chifoumi
- chifoumi.h : Module de la classe chifoumi
- chifoumi.cpp : Corps de la classe chifoumi

### 8.2 Test

A faire :

Décrire les tests prévus / réalisés pour montrer :

<i>Valeur rentrée</i>	<i>Valeur attendue</i>	<i>Valeur affichée</i>	<i>Commentaire</i>
<i>ClickFigure Pierre</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>ClickFigure Pierre</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>ClickFigure Pierre</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickNouvPart</i>	<i>Score joueur = 0 et score machine = 0</i>	<i>Score joueur = 0 et score machine = 0</i>	<i>ok</i>

## Version v2

### 9. Implémentation et tests

#### 9.1 Implémentation

Liste des fichiers de cette version :

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaire à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi

presentation.h : Module de la classe présentation

presentation.cpp : Sources de la classe présentation

chifoumi.cpp : Corps de la classe chifoumi

chifoumi.h : Module de la classe chifoumi

#### 9.2 Présentation des .h

La classe presentation représente le modèle de la classe Chifoumi. (Fonctionnement général du jeu)

La classe chifoumi représente la présentation de la classe Chifoumi. (Fonctionnement complet du jeu)

La classe chifoumiVue représente la vue de la classe Chifoumi. (Aspect graphique)

### 9.3 Tests

Test avec le programme fourni main.cpp

*Valeurs fournies / attendues... comme montré dans la ressource R2.03 (partie tests)*

<i>Valeur rentrée</i>	<i>Valeur attendue</i>	<i>Valeur affichée</i>	<i>Commentaire</i>
<i>ClickFigure Pierre</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>ClickFigure Pierre</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>ClickFigure Pierre</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickNouvPart</i>	<i>Score joueur = 0 et score machine = 0</i>	<i>Score joueur = 0 et score machine = 0</i>	<i>ok</i>

## Version 3

### 10. Implémentation et tests

#### 10.1 Implémentation

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaires à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et l'exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi

presentation.h : Module de la classe présentation

presentation.cpp : Sources de la classe présentation

chifoumi.cpp : Corps de la classe chifoumi

chifoumi.h : Module de la classe chifoumi

Les seuls fichiers .h modifiés pendant la mise en œuvre de la v3 sont « chifoumiVue.h » et « presentation.h ».

Le slot de la présentation appelle la méthode de la vue qui affiche la message Box.

#### 10.2 Tests

<i>Valeur rentrée</i>	<i>Valeur attendue</i>	<i>Valeur affichée</i>	<i>Commentaire</i>
<i>ClickFigure Pierre</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>ClickFigure Pierre</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>ClickFigure Pierre</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>clickFigure Papier</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur + 1 et score machine</i>	<i>Score joueur + 1 et score machine</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur et score machine</i>	<i>Score joueur et score machine</i>	<i>ok</i>
<i>clickFigure Ciseau</i>	<i>Score joueur et score machine + 1</i>	<i>Score joueur et score machine + 1</i>	<i>ok</i>
<i>clickNouvPart</i>	<i>Score joueur = 0 et score machine = 0</i>	<i>Score joueur = 0 et score machine = 0</i>	<i>ok</i>
<i>triggered(actionQuitter)</i>	<i>Fin du programme</i>	<i>Fin du programme</i>	<i>ok</i>
<i>triggered(actionA_propos_de)</i>	<i>Afficher Message Box</i>	<i>Affiche Message Box</i>	<i>ok</i>
<i>F1</i>	<i>Afficher Message Box</i>	<i>Affiche Message Box</i>	<i>ok</i>
<i>Alt + F3</i>	<i>Fin du programme</i>	<i>Fin du programme</i>	<i>ok</i>