JORGE Thomas
GOMES Noah
POTIES Guilhem

TD III / TP 5

Date de remise : 7 juin 2022

BUT Informatique - Semestre 2 (2021/2022)

S2.01 - Développement d'une application

Chifoumi : Dossier d’analyse et conception

I. Compléments

1. Diagramme des cas d'utilisation

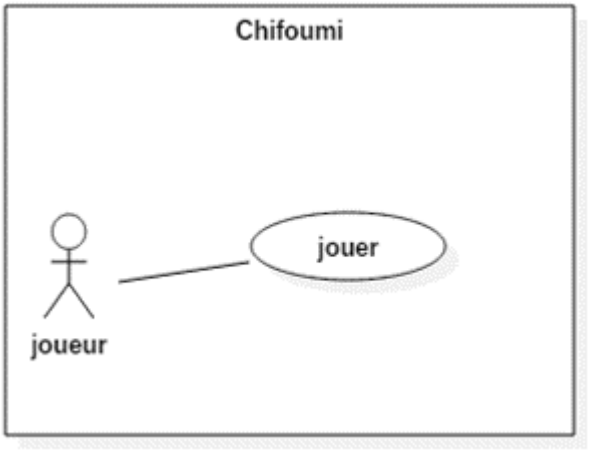


Figure 1 : Diagramme des Cas d'Utilisation du jeu Chifoumi

2. 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

3. 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.

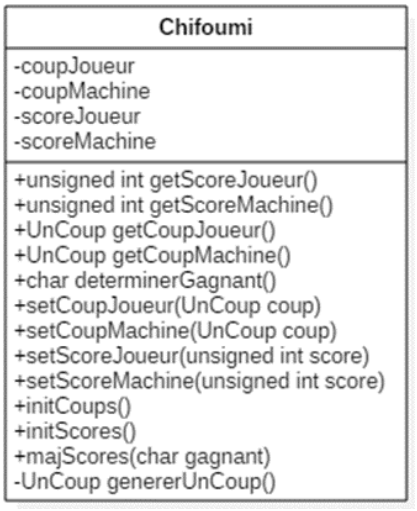


Figure 2 : Diagramme de Classes UML du jeu Chifoumi

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`	Mémorise la dernière figure choisie par le joueur. Type énuméré enum unCoup {pierre, ciseau, papier, rien};	`UnCoup`	`Papier`
`coupMachine`	Mémorise la dernière figure choisie par la machine.	`UnCoup`	`Ciseau`

Tableau 2 : Dictionnaire des éléments - Classe Chifoum

c. Dictionnaire des méthodes : intégrées dans l'interface de la classe

```
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 4 : Schéma de classes = Une seule classe Chifoumi

d. Remarques concernant le schéma de classes

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.

On n'a mis ni le constructeur ni le destructeur, pour alléger le schéma.

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.

II. v0 : Sources C++ de la classe Chifoumi dans un projet Non Qt

1. Liste et rôles des fichiers sources

chifoumi.h : Entête du module Chifoumi qui permet de programmer le jeu.
chifoumi.cpp : Corps du module Chifoumi qui permet de programmer le jeu.
main.cpp : Fichier permettant d'effectuer les tests du module Chifoumi.

2. Résultats des tests réalisés

Méthodes testées	Fichier	Valeur attendue	Valeur obtenue	Commentaire
joueur pierre machine pierre	main.cpp	determinerGagnant = N	N	ok
joueur pierre machine feuille	main.cpp	determinerGagnant = M	M	ok
joueur pierre machine ciseaux	main.cpp	determinerGagnant = J	J	ok
joueur feuille machine pierre	main.cpp	determinerGagnant = J	J	ok
joueur feuille machine feuille	main.cpp	determinerGagnant = N	N	ok
joueur feuille machine ciseaux	main.cpp	determinerGagnant = M	M	ok
joueur ciseaux machine pierre	main.cpp	determinerGagnant = M	M	ok

Méthodes testées	Fichier	Valeur attendue	Valeur obtenue	Commentaire
joueur ciseaux machine feuille	main.cpp	determinerGagnant = J	J	ok
joueur ciseaux machine ciseaux	main.cpp	determinerGagnant = N	N	ok

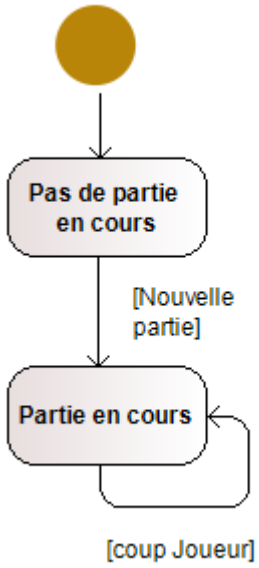
Lors de l'exécution du programme,

```
appel du constructeur : construction d'un chifoumi : scores a 0, et coupsJoueurs a RIEN'

teste les methodes get() associees aux attributs 'score'
score Joueur : 0       score Machine : 0

teste les methodes get() associees aux attributs 'coup'
coup Joueur : rien   coup Machine : rien

teste les methodes set() associees aux attributs 'score'
score Joueur : 1       score Machine : 2

teste initScores()
score Joueur : 0       score Machine : 0

teste les methodes set() et get() associees aux attributs 'coup'/'choix'
coup Joueur : pierre   coup Machine : ciseau

quelques tours de jeu pour tester l'identification du gagnant et la maj des scores
coup Joueur : pierre   coup Machine : ciseau
score Joueur : 1       score Machine : 0
```

III. v1 : Version simple correspondant au sujet

1. Diagramme d'état-transitions du jeu dans sa version initiale

demande nouvelle partie + coup joueur



2. Dictionnaires états, événements, actions associés

a. Dictionnaire des états du jeu

Nom de l'état	**Signification**
pasDePartie	Il n'y a pas de partie lancé
partieEnCours	Le joueur fait une partie contre l'ordinateur

b. Dictionnaire des événements faisant changer le jeu d'état

Nom de l'événement	**Signification**
nouvellePartie	Le joueur lance une partie
coupJoueur	Le joueur joue un coup

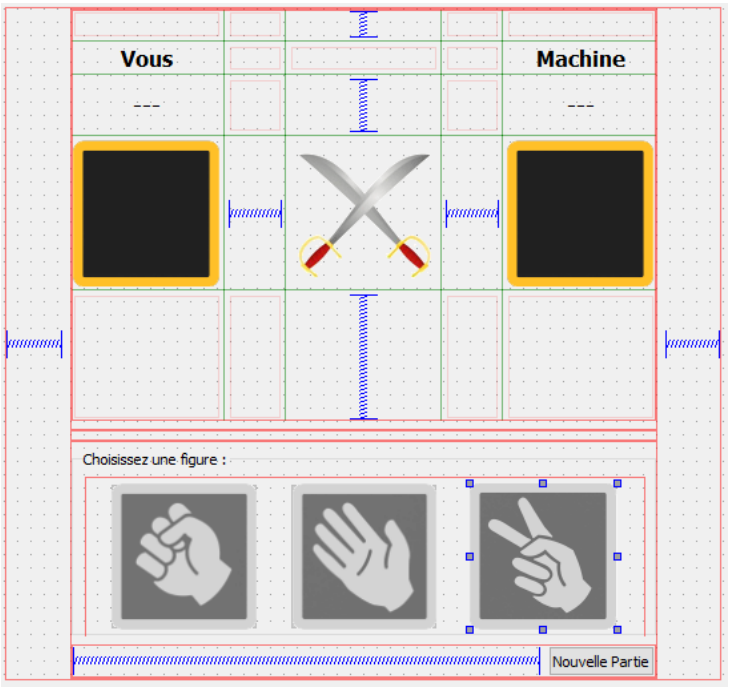
c. Description des actions réalisées lors de la traversée des transitions

Nom de l'action	**Signification**
lancerPartie	Une nouvelle partie est lancée
jouerCoup	Un coup a été joué
lancerJeu	Le jeu est exécuté

3. Version matricielle du diagramme états-transitions + identification des éléments d'interface

Evénement → ↓ *nomEtatJeu*	**coupJoueurJoue**	**nvllePartieDemandee**
pasDePartie	Faux	Vrai
partieEnCours	Vrai	Faux

4. Description des éléments d'interface



Dans l'interface nous avons tout d'abord la fenêtre principale dans laquelle sont disposés les différents éléments. On peut la fermer, la réduire et la redimensionner. Ensuite nous avons en bas un tableau contenant les différentes figures ou coup possible inactif au départ mais cliquable une fois le bouton "nouvelle partie" cliqué. En haut il y a 2 labels permettant d'afficher les scores du joueur et du système. Pour finir les 2 cadres en dessous des scores sont vides au départ et accueillent ensuite l'image sélectionnée par le joueur en bas et celle sélectionnée automatiquement par le système. Ils permettent de visualiser les coups choisies

nomJoueur : QLabel contenant "Vous".
nomMachine : QLabel contenant "Machine".
scoreJoueur : QLabel contenant le score du joueur.
scoreMachine : QLabel contenant le score de la machine.
coupJoueur : QLabel contenant l'image du coup du joueur.
coupMachine : QLabel contenant l'image du coup de la machine.
affichageEpee : QLabel contenant l'image des deux sabres croisés.
pierre : QLabel contenant l'image de la pierre.
feuille : QLabel contenant l'image de la feuille.
ciseau : QLabel contenant l'image du ciseau.
boutonPartie : QPushButton permettant de lancer une nouvelle partie.

5. Liste des fichiers sources de cette version (et rôle de chacun)

chifoumiJeu.h : Entête du module ChifoumiJeu (anciennement Chifoumi) qui permet de programmer le jeu.

chifoumiJeu.cpp : Corps du module ChifoumiJeu (anciennement Chifoumi) qui permet de programmer le jeu.

chifoumi.h : Entête du module Chifoumi (module graphique)

chifoumi.cpp : Corps du module Chifoumi (module graphique)

chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi

chifoumi.pro : Fichiers projet sur Qt

ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées

6. Résultats des tests réalisés

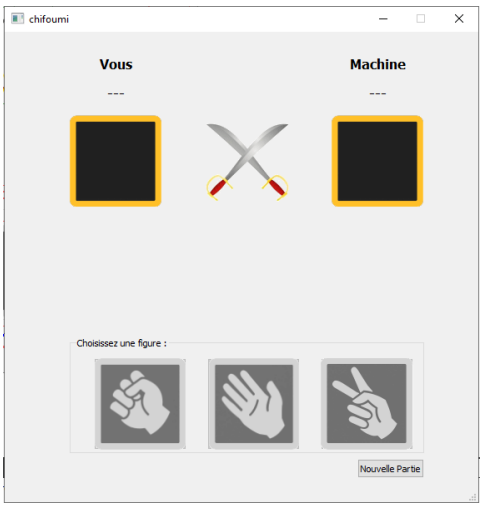


Image 1 : *Page venant après l'exécution*

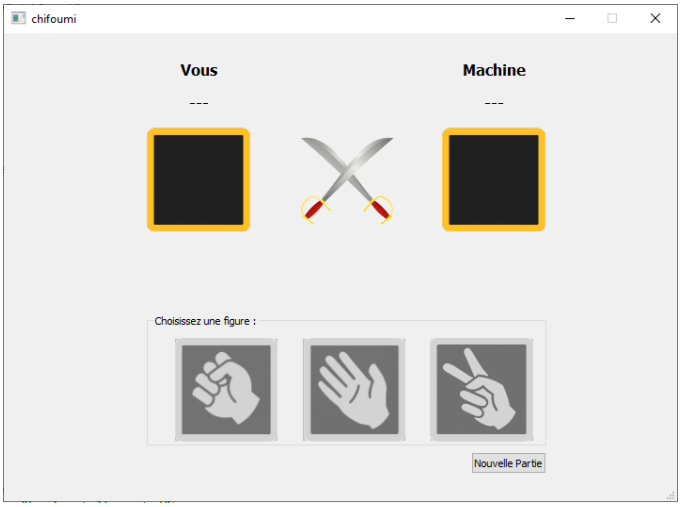


Image 2 *: Page après agrandissement*

IV. v3 : Ajout Barre Menus Fichier et Aide

1. Liste des fichiers sources de cette version (et rôle de chacun)

chifoumiJeu.h : Entête du module ChifoumiJeu (anciennement Chifoumi) qui permet de programmer le jeu.

chifoumiJeu.cpp : Corps du module ChifoumiJeu (anciennement Chifoumi) qui permet de programmer le jeu.

chifoumi.h : Entête du module Chifoumi (module graphique).

chifoumi.cpp : Corps du module Chifoumi (module graphique).

chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi.

chifoumi.pro : Fichiers projet sur Qt.

ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées.

2. Présentation des seuls fichiers .h modifiés par la mise en œuvre de la v3

chifoumi.h : Ajout d'un slot "infosApp" permettant d'afficher les propriétés de l'application.

3. Si pertinent, explications sur des points importants à savoir concernant les .cpp

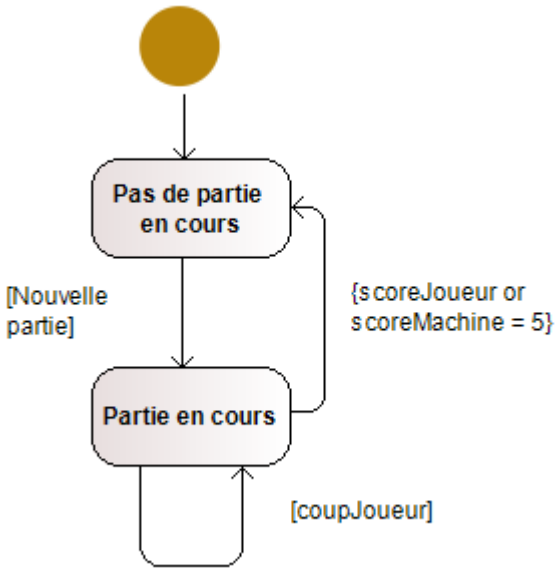
chifoumi.cpp : Contient le corps du slot "infosApp".

4. Résultats des tests réalisés

Méthodes testées	Fichier	Résultat attendu	Résultat obtenu	Commentaire
infosApp	chifoumi.h chifoumi.cpp	Affichage des propriétés de l'application.	Propriétés de l'application affichées.	ok
actionQuitter	chifoumi.ui	Fermeture de la fenêtre de jeu.	Fenêtre de jeu fermée.	ok

V. v4 : Jeu en 5 points

1. Diagramme d'état-transitions de cette version du jeu



2. Dictionnaires états, événements, actions associés

a. Dictionnaire des états du jeu

Nom de l'état	**Signification**
pasDePartie	Il n'y a pas de partie lancé
partieEnCours	Le joueur fait une partie contre l'ordinateur

b. Dictionnaire des événements faisant changer le jeu d'état

Nom de l'événement	**Signification**
nouvellePartie	Le joueur lance une partie
coupJoueur	Le joueur joue un coup
scoreJoueur or scoreMachine = 5	Le joueur ou la machine a atteint un score de 5

c. Description des actions réalisées lors de la traversée des transitions

Nom de l'action	**Signification**
lancerPartie	Une nouvelle partie est lancée
jouerCoup	Un coup a été joué
lancerJeu	Le jeu est exécuté
partieFinie	La partie est terminée

3. Version matricielle du diagramme états-transitions + identification des éléments d'interface supplémentaires éventuellement nécessaires

Evénement → ↓ *nomEtatJeu*	**coupJoueurJoue**	**nvllePartieDemandee**	**scoreJoueur or scoreMachine = 5**
pasDePartie	Faux	Vrai	Faux
partieEnCours	Vrai	Faux	Vrai

4. Décrire les nouveaux éléments d'interface

gagnant : QLabel contenant la chaîne de caractères "GAGNANT".

pointsPartie : QLabel contenant le nombre de points nécessaire pour gagner la partie.

5. Liste des fichiers sources de cette version

chifoumiJeu.h : Entête du module ChifoumiJeu qui permet de programmer le jeu.

chifoumiJeu.cpp : Corps du module ChifoumiJeu qui permet de programmer le jeu.

chifoumi.h : Entête du module Chifoumi (module graphique).

chifoumi.cpp : Corps du module Chifoumi (module graphique).

chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi **.**

chifoumi.pro : Fichiers projet sur Qt.

ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées.

6. Présentation des seuls fichiers .h modifiés par la mise en œuvre de la v4

chifoumi.h : Ajout d'une procédure "finPartieGagnant" permettant de déterminer le gagnant lorsque le score est atteint par le joueur ou la machine.

7. Si pertinent, explications sur des points importants à savoir concernant les .cpp

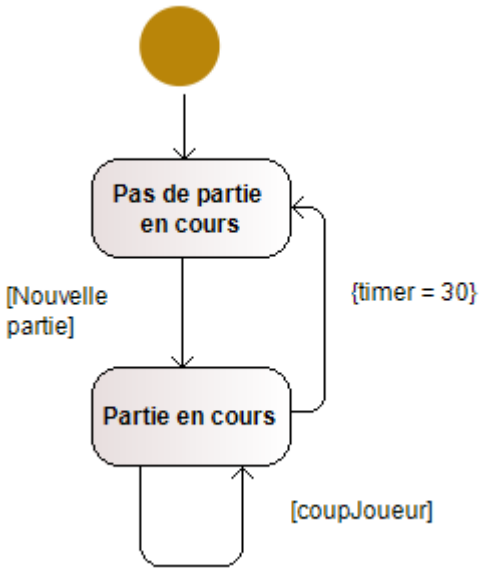
chifoumi.cpp : Contient le corps du slot "finPartieGagnant".

8. Résultats des tests réalisés

Méthodes testées	Fichier	Résultat attendu	Résultat obtenu	Commentaire
finPartieGagnant	chifoumi.h chifoumi.cpp	Termine la partie lorsque le joueur atteint 5	Partie terminée lorsque le joueur atteint 5	ok
finPartieGagnant	chifoumi.h chifoumi.cpp	Termine la partie lorsque la machine atteint 5	Partie terminée lorsque la machine atteint 5	ok

VI. v5 : Jeu en temps limité

1. Diagramme d'état-transitions de cette version du jeu



2. Dictionnaires états, événements, actions associés (mettre en évidence les changements)

a. Dictionnaire des états du jeu

Nom de l'état	**Signification**
pasDePartie	Il n'y a pas de partie lancé
partieEnCours	Le joueur fait une partie contre l'ordinateur

b. Dictionnaire des événements faisant changer le jeu d'état

Nom de l'événement	**Signification**
nouvellePartie	Le joueur lance une partie
coupJoueur	Le joueur joue un coup
timer = 30	Le timer de 30 secondes s'est écoulé

c. Description des actions réalisées lors de la traversée des transitions

Nom de l'action	**Signification**
lancerPartie	Une nouvelle partie est lancée
jouerCoup	Un coup a été joué
lancerJeu	Le jeu est exécuté
partieFinie	La partie est terminée

3. Version matricielle du diagramme états-transitions + identification des éléments d'interface supplémentaires éventuellement nécessaires

Evénement → ↓ *nomEtatJeu*	**coupJoueurJoue**	**nvllePartieDemandee**	**timer = 30**
pasDePartie	Faux	Vrai	Faux
partieEnCours	Vrai	Faux	Vrai

4. Décrire les nouveaux éléments d'interface

timer : QTimer contenant le compte à rebours avant la fin de la partie.

5. Liste des fichiers sources de cette version (et rôle de chacun)

chifoumiJeu.h : Entête du module ChifoumiJeu qui permet de programmer le jeu.

chifoumiJeu.cpp : Corps du module ChifoumiJeu qui permet de programmer le jeu.

chifoumi.h : Entête du module Chifoumi (module graphique).

chifoumi.cpp : Corps du module Chifoumi (module graphique).

chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi **.**

chifoumi.pro : Fichiers projet sur Qt.

ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées.

6. Présentation des seuls fichiers .h modifiés par la mise en œuvre de la v4

chifoumi.h : Ajout d'une procédure "temp" permettant d'arrêter la partie lorsque le compte à rebours arrive à 0.

7. Si pertinent, explications sur des points importants à savoir concernant les .cpp

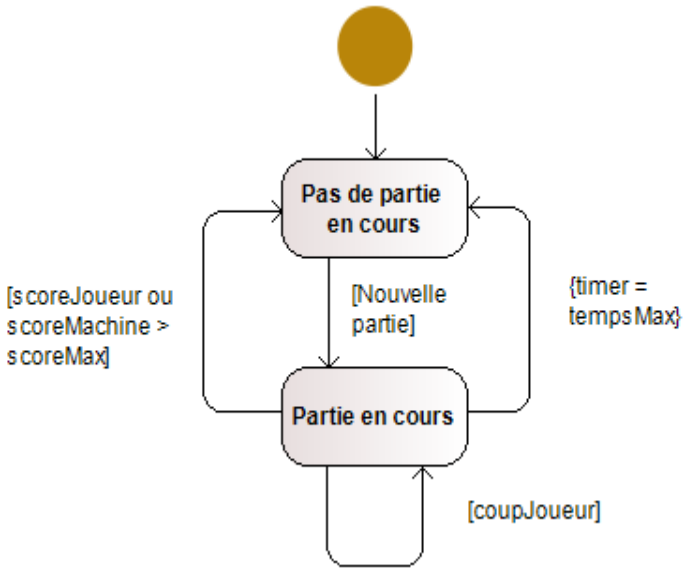
chifoumi.cpp : Contient le corps du slot "temp".

8. Résultats des tests réalisés

Méthodes testées	Fichier	Résultat attendu	Résultat obtenu	Commentaire
temp	chifoumi.h chifoumi.cpp	Termine la partie lorsque le compte à rebours arrive à 0.	Partie terminée après que le compte à rebours soit arrivé à 0.	ok

VII. v6 : Ajout Fichier >> Paramétrage

1. Diagramme d'état-transitions de cette version du jeu



2. Dictionnaires états, événements, actions associés

a. Dictionnaire des états du jeu

Nom de l'état	**Signification**
pasDePartie	Il n'y a pas de partie lancé
partieEnCours	Le joueur fait une partie contre l'ordinateur

b. Dictionnaire des événements faisant changer le jeu d'état

Nom de l'événement	**Signification**
nouvellePartie	Le joueur lance une partie
coupJoueur	Le joueur joue un coup
scoreJoueur or scoreMachine = 5	Le joueur ou la machine a atteint le score max
timer = 30	Le timer dont la durée correspond au temps max s'est écoulé

c. Description des actions réalisées lors de la traversée des transitions

Nom de l'action	**Signification**
lancerPartie	Une nouvelle partie est lancée
jouerCoup	Un coup a été joué
lancerJeu	Le jeu est exécuté
partieFinie	La partie est terminée

3. Version matricielle du diagramme états-transitions + identification des éléments d'interface supplémentaires éventuellement nécessaires

Evénement → *↓ nomEtatJeu*	**coupJoueurJoue**	**nvllePartieDemandee**	**scoreJoueur or scoreMachine = 5**	**timer = 30**
pasDePartie	Faux	Vrai	Faux	Faux
partieEnCours	Vrai	Faux	Vrai	Vrai

4. Décrire les nouveaux éléments d'interface

nomJoueur : QLabel contenant la chaîne de caractères "Nom".
scoreJoueur : QLabel contenant la chaîne de caractères "Durée de la partie (en secondes)".
pointsMax : QLabel contenant la la chaîne de caractères "Points maximums".
nom : QLineEdit contenant le nom entré par le joueur.
temps : QLineEdit contenant la durée de la partie entré par le joueur.
points : QLineEdit contenant le nombre de points max entré par le joueur.

5. Liste des fichiers sources de cette version

chifoumidialog.h : Entête du module ChifoumiDialog qui permet de gérer une fenêtre de dialogue

chifoumidialog.cpp : Corps du module ChifoumiDialog.

chifoumiJeu.h : Entête du module ChifoumiJeu qui permet de programmer le jeu.

chifoumiJeu.cpp : Corps du module ChifoumiJeu qui permet de programmer le jeu.

chifoumi.h : Entête du module Chifoumi (module graphique).

chifoumi.cpp : Corps du module Chifoumi (module graphique).

chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi **.**

chifoumiDialog.ui : Fichier contenant les élément graphiques de ChifoumiDialog.

chifoumi.pro : Fichiers projet sur Qt.

ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées.

6. Présentation des seuls fichiers .h modifiés par la mise en œuvre de la v6

chifoumi.h : Implémentation de la classe dialogchifoumi, création d'un objet dialogchifoumi et déclaration d'une procédure "infosApp"

7. Si pertinent, explications sur des points importants à savoir concernant les .cpp

chifoumi.cpp : Contient le corps de la procédure "infosApp".

8. Résultats des tests réalisés

Méthodes testées	Fichier	Résultat attendu	Résultat obtenu	Commentaire
infosApp	chifoumi.h chifoumi.cpp	Modifie le nom du joueur, le temps de la partie et le score max en fonction des paramètres entrés	Les paramètres sont mis à jour en début de partie	ok

VIII. v7 : Enregistrement joueur en BD

1. Décrire les nouveaux éléments d'interface

mdp : QLineEdit permettant de saisir le mot de passe.
motdepasseLabel : QLabel contenant “Mot de passe”
username : QLineEdit permettant de saisir le nom d’utilisateur.
utilisateurLabel : QLabel contenant “Utilisateur”.
boutonEntrer : QPushButton permettant de valider les informations saisies et affichant “Entrer”.
titre : QLabel contenant “Identifiez-vous”.

2. Liste des fichiers sources de cette version

chifoumiJeu.h : Entête du module ChifoumiJeu qui permet de programmer le jeu.
chifoumiJeu.cpp : Corps du module ChifoumiJeu qui permet de programmer le jeu.
chifoumi.h : Entête du module Chifoumi (module graphique).
chifoumi.cpp : Corps du module Chifoumi (module graphique).
chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi.
chifoumiDialog.ui : Fichier contenant les élément graphiques de ChifoumiDialog.
chifoumi.pro : Fichiers projet sur Qt.
ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées.
chifoumiConnexion.h : Entête du module ChifoumiConnexion qui permet de gérer la connexion à la base de données.
chifoumiConnexion.cpp : Corps du module ChifoumiConnexion.
chifoumiConnexion.ui : Fichier contenant les éléments graphique la fenêtre de connexion à la base de données.
database.h : Entête du module database permettant d’intéragir avec des bases de données.
database.cpp : Corps du module database.

3. Présentation des seuls fichiers .h modifiés par la mise en œuvre de la v4

chifoumiConnexion.h : Ajout d’une méthode “connexion” permettant de se connecter à une base de données.

4. **Si pertinent, explications sur des points importants à savoir concernant les .cpp**

   chifoumiConnexion.cpp : Contient le corps de la méthode “connexion”.

5. **Résultats des tests réalisés**

Méthodes testées	**Fichier**	**Résultat attendu**	**Résultat obtenu**	**Commentaire**
connexion	chifoumiConnexion.h chifoumiConnexion.cpp	Erreur de saisie donc connexion refusée.	Connexion refusée car identifiant ou mdp pas bon.	ok
connexion	chifoumiConnexion.h chifoumiConnexion.cpp	Saisie valide donc connexion effectuée.	Connexion effectuée car saisie des identifiants valide.	ok

IX. v8 : Enregistrement résultat partie en BD

1. Liste des fichiers sources de cette version

chifoumiJeu.h : Entête du module ChifoumiJeu qui permet de programmer le jeu.

chifoumiJeu.cpp : Corps du module ChifoumiJeu qui permet de programmer le jeu.

chifoumi.h : Entête du module Chifoumi (module graphique).

chifoumi.cpp : Corps du module Chifoumi (module graphique).

chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi.

chifoumiDialog.ui : Fichier contenant les élément graphiques de ChifoumiDialog.

chifoumi.pro : Fichiers projet sur Qt.

ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées.

chifoumiConnexion.h : Entête du module ChifoumiConnexion qui permet de gérer la connexion à la base de données.

chifoumiConnexion.cpp : Corps du module ChifoumiConnexion.

chifoumiConnexion.ui : Fichier contenant les éléments graphiques de la fenêtre de connexion à la base de données.

database.h : Entête du module database permettant d'intéragir avec des bases de données.

database.cpp : Corps du module database.

2. Présentation des seuls fichiers .h modifiés par la mise en œuvre de la v4

chifoumiConnexion.h : Ajout d'une méthode "enregistrementPartie" permettant d'enregistrer le nom du joueur, le score final ainsi que le timer d'une partie.

3. Si pertinent, explications sur des points importants à savoir concernant les .cpp

chifoumiConnexion.cpp : Contient le corps de la méthode "enregistrementPartie".

chifoumi.cpp : Utilisation de la méthode "enregistrementPartie" pour enregistrer les informations des parties après la fin de celles-ci.

4. Résultats des tests réalisés

Méthodes testées	Fichier	Résultat attendu	Résultat obtenu	Commentaire
enregistrementPartie	chifoumiConnexion.h chifoumiConnexion.cpp chifoumi.cpp	Enregistrement de la partie dans la base de données après la victoire du joueur.	Partie enregistrée dans la base de données après la victoire du joueur.	ok
enregistrementPartie	chifoumiConnexion.h chifoumiConnexion.cpp chifoumi.cpp	Enregistrement de la partie dans la base de données après une égalité.	Partie enregistrée dans la base de données après une égalité.	ok
enregistrementPartie	chifoumiConnexion.h chifoumiConnexion.cpp chifoumi.cpp	Enregistrement de la partie dans la base de données après la victoire de la machine..	Partie enregistrée dans la base de données après la victoire de la machine..	ok

v9 : Ajout Fichier >> Résultats

1. Décrire les nouveaux éléments d'interface

tableResultat : QTableView permettant d'afficher les parties enregistrées dans la base de données.

boutonQuitter : QPushButton permettant de quitter la fenêtre.

titre : QLabel contenant "Résultats".

2. Liste des fichiers sources de cette version

chifoumiJeu.h : Entête du module ChifoumiJeu qui permet de programmer le jeu.

chifoumiJeu.cpp : Corps du module ChifoumiJeu.

chifoumi.h : Entête du module Chifoumi (module graphique).

chifoumi.cpp : Corps du module Chifoumi (module graphique).

chifoumiDialog.ui : Fichier contenant les élément graphiques de ChifoumiDialog.

chifoumi.ui : Fichier contenant les éléments graphiques du Chifoumi.

chifoumi.pro : Fichiers projet sur Qt.

ressourcesChifoumi.qrc : Fichier contenant les références des ressources utilisées.

chifoumiConnexion.h : Entête du module ChifoumiConnexion qui permet de gérer la connexion à la base de données.

chifoumiConnexion.cpp : Corps du module ChifoumiConnexion.

chifoumiConnexion.ui : Fichier contenant les éléments graphiques de la fenêtre de connexion à la base de données.

database.h : Entête du module database permettant d'intéragir avec des bases de données.

database.cpp : Corps du module database.

afficherResultat.h : Entête du module afficherResultat qui permet d'afficher les parties enregistrées dans la base de données.

afficherResultat.cpp : Corps du module afficherResultat.

afficherResultat.ui : Fichier contenant les éléments graphiques de afficherResultat.

3. Présentation des seuls fichiers .h modifiés par la mise en œuvre de la v4

chifoumi.h : Ajout d'une méthode "resultats" permettant d'ouvrir la fenêtre de résultats.

4. **Si pertinent, explications sur des points importants à savoir concernant les .cpp**

   chifoumi.cpp : Contient le corps de la méthode “resultats”.

5. **Résultats des tests réalisés**

Méthodes testées	**Fichier**	**Résultat attendu**	**Résultat obtenu**	**Commentaire**
resultats	chifoumi.h chifoumi.cpp afficherResultats.h afficherResultats.cpp	Lorsque le joueur clique sur ficher/Résultats, s’ouvre la fenêtre contenant les résultats.	Après le clique sur joueur, la fenêtre s’ouvre.	ok