

BUT Informatique
Année 2021-2022, Semestre 2

Etudiants TDI TP2

- **DUJARDIN Esteban**
- **LANUSSE Damien**
- **RODRIGUEZ SINCLAIR Juan David**



SAE 2.01 Conception d'une application

Dossier d'analyse et de conception

Lien GitHub:

https://github.com/Darima177/S201-D-veloppement_d_une_application

v0	
Liste des fichiers présents :	2
Résultats de l'exécution du fichier main.cpp pour les tests	2
v1	
Diagramme d'état	3
Dictionnaire des objets associés au diagramme d'état	3
Disposition et Description des éléments d'interface	4
Liste des fichiers présents	5
Tableau des tests	6-7
v2	
Liste des fichiers présents	8-11
Tableau des tests	12-13
v3	
Liste des fichiers présents	14
Description des modifications fichiers .h et .cpp	14
Ajouts dans le tableau des tests	15
v4	
Diagramme d'état	16
Dictionnaire des objets associés au diagramme d'état	16
Version matricielle du diagramme d'état transition	17
Description des éléments d'interface	17
Liste des fichiers présents	17-18
Description des modifications des fichiers .h et .cpp	18
Tableau des tests	18-19
v5	
Diagramme d'état	19
Dictionnaire des objets associés au diagramme d'état	19-20
Version matricielle du diagramme d'état transition	21
Description des éléments d'interface	22
Liste des fichiers présents	22
Description des modifications des fichiers .h et .cpp	23
Tableau des tests	23-24
v6	
Diagramme d'état	24
Dictionnaire des objets associés au diagramme d'état	24-25
Version matricielle du diagramme d'état transition	26
Description des éléments d'interface	27
Disposition et Description des éléments d'interface	28
Liste des fichiers présents	28-29
Description des modifications des fichiers .h et .cpp	29
Nouveau tableau des tests	30

v7

Diagramme d'état	32
Dictionnaire des objets associés au diagramme d'état	32-33
Version matricielle du diagramme d'état transition	34
Description des éléments d'interface	35
Disposition et Description des éléments d'interface	35
Liste des fichiers présents	36-37
Description des modifications des fichiers .h et .cpp	37-38
Nouveau tableau des tests	39

v8

Diagramme d'état	39
Dictionnaire des objets associés au diagramme d'état	40-41
Version matricielle du diagramme d'état transition	42
Liste des fichiers présents	43
Description des modifications des fichiers .h et .cpp	44
Nouveau tableau des tests	45

v9

Diagramme d'état	46
Dictionnaire des objets associés au diagramme d'état	46-47
Version matricielle du diagramme d'état transition	48
Liste des fichiers présents	49
Description des modifications des fichiers .h et .cpp	50
Nouveau tableau des tests	50

v0

Liste des fichiers présents :

- [main.cpp](#) : appelle du constructeur [chifoumi.h](#) et test effectue les tests des différentes variables et fonctions présentes.
Vérification du bon fonctionnement du constructeur, des valeurs associées à score, coup et set (lors de l'initialisation et des modifications des valeurs), des tours de jeux.
- [chifoumi.h](#) : initialisation des différentes fonctions, variables et procédures nécessaires au bon fonctionnement du jeu chifoumi
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et variables globales.

Résultats de l'exécution du fichier main.cpp :

```
C:\WINDOWS\system32\cmd.exe - main.exe
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

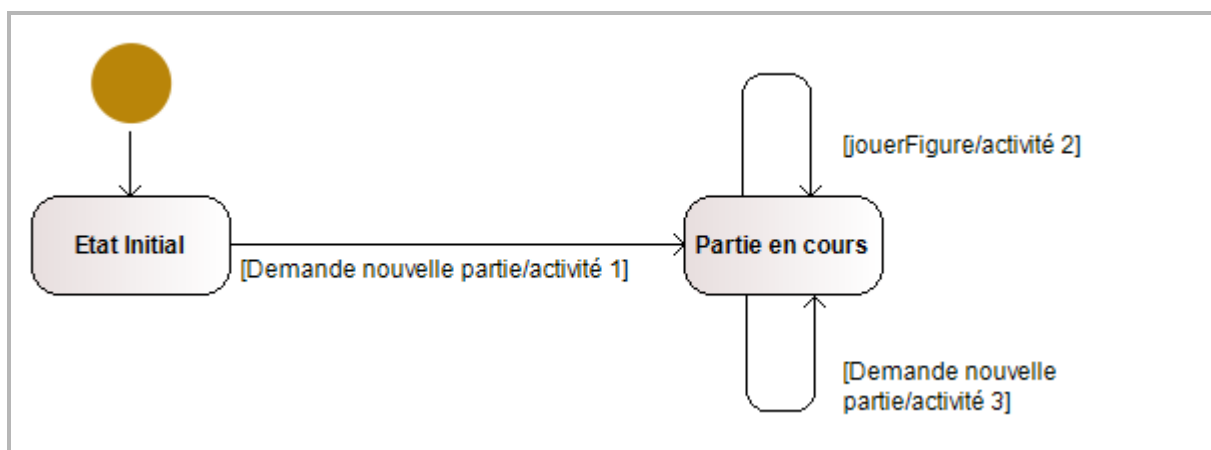
teste initScores()
score Joueur : 0      score Machine : 0

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 : ciseau  coup Machine : ciseau
score Joueur : 0      score Machine : 0
```

v1

Diagramme d'état :



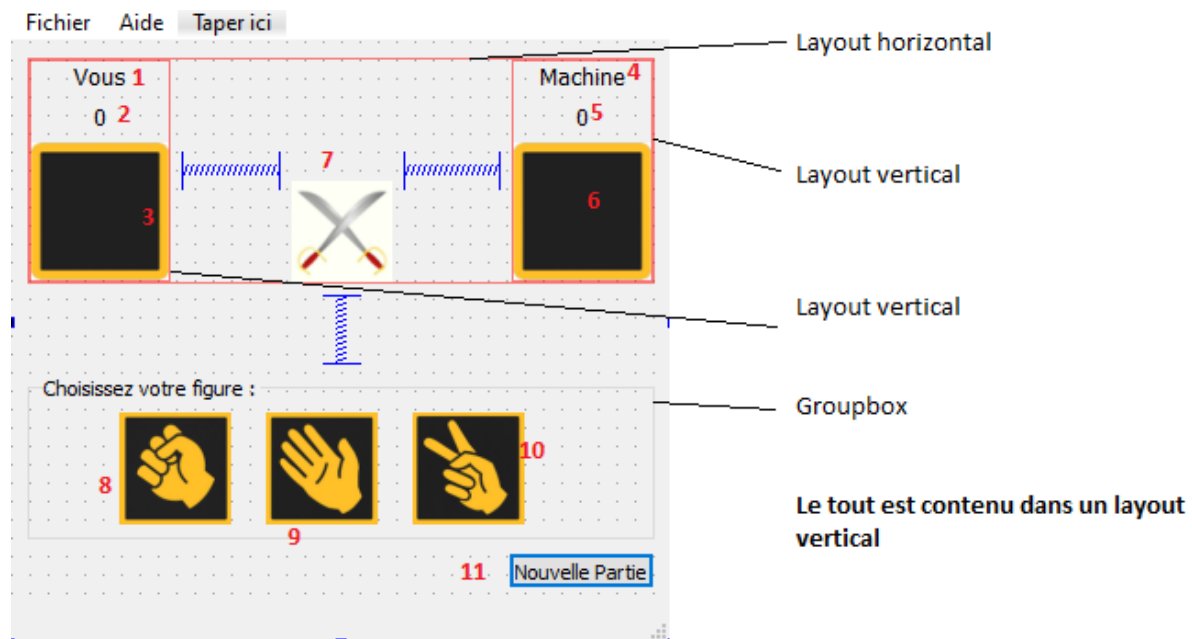
Dictionnaire des états, événements et actions associées au diagramme d'activité :

Nom	Type	Description
Etat initial	Etat	La fenêtre est affichée avec les boutons des figures indisponibles. Les scores sont à 0 et les joueurs ont "rien" comme figure
Partie en cours	Etat	Les boutons sont disponibles, les scores actuels sont affichés et les dernières figures du joueur et de la machine aussi
Jouer figure	Evenement	La figure du joueur est enregistrée comme étant celle choisie ; une figure est assignée à la machine de manière aléatoire ; comparaison des figures ; modification des scores ; mise à jour de l'affichage
Demande nouvelle partie	Evenement	Si l'état est initial : modification de l'état, les scores du joueur et de la machine sont remis à 0 et les figures sont modifiées vers "rien", mise à jour de l'affichage

Diagramme d'état-transition version matricielle

	pushButtonCiseau	pushButtonPierre	pushButtonpapier	pushButtonNouvellePartie
État actuel \ Évènement	Jouer figure			Demande nouvelle partie
État initial	-----			Activite 1
Partie en cours	Activite 2			Activite 3

Disposition et Description des éléments d'interface



Dû à un oubli avant la capture d'écran, ceci n'est pas présent sur l'image mais les boutons dans le groupbox sont placés dans un layout horizontal avec des stretchers horizontaux à gauche et à droite.

N° Element	Elément	Interface
1	labelJoueur	Label pour l'affichage du "nom" du joueur
2	labelScoreJoueur	Label pour l'affichage du score du joueur
3	labelFigureJoueur	Label pour l'affichage de la figure jouée par le joueur
4	labelMachine	Label pour l'affichage du "nom" de la machine
5	labelScoreMachine	Label pour l'affichage du score de la machine
6	labelFigureMachine	Label pour l'affichage de la figure jouée par la machine
7	labelVS	Label pour l'affichage de l'image versus
8	pushButtonPierre	Bouton poussoir pour jouer la figure pierre
9	pushButtonPapier	Bouton poussoir pour jouer la figure papier
10	pushButtonCiseau	Bouton poussoir pour jouer la figure ciseau

11	pushButtonNouvelle Partie	Bouton poussoir pour commencer une nouvelle partie
----	------------------------------	---

Liste des fichiers présents :

- [main.cpp](#) : exécute l'application
- [chifoumivue.h](#) : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- [chifoumivue.cpp](#) : déclaration des différents sous-programmes et slots.
- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les image affichées

Tableau des tests

Elément testé : distribution des points selon les figures

Date : 29 / 04 / 2022

Version : v1

<u>Classe</u>	<u>Description</u>	<u>Forme choisie par le joueur</u>	<u>Forme générée pour la machine</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>
Égalité	Le joueur et la machine jouent la même figure	Ciseaux	Ciseaux	pas de modification des points	pas de modification des points
Égalité	Le joueur et la machine jouent la même figure	Pierre	Pierre	pas de modification des points	pas de modification des points

Égalité	Le joueur et la machine jouent la même figure	Feuilles	Feuilles	pas de modification des points	pas de modification des points
Victoire joueur	Le joueur choisit une figure qui bat la machine	Ciseaux	Feuilles	score du joueur +1	score du joueur +1
Victoire joueur	Le joueur choisit une figure qui bat la machine	Pierre	Ciseaux	score du joueur +1	score du joueur +1
Victoire joueur	Le joueur choisit une figure qui bat la machine	Feuilles	Pierre	score du joueur +1	score du joueur +1
Victoire machine	le joueur choisit une figure qui perd contre la machine	Ciseaux	Pierre	score de la machine +1	score de la machine +1
Victoire machine	le joueur choisit une figure qui perd contre la machine	Pierre	Feuilles	score de la machine +1	score de la machine +1
Victoire machine	le joueur choisit une figure qui perd contre la machine	Feuilles	Ciseaux	score de la machine +1	score de la machine +1

Liste des fichiers présents :

- [main.cpp](#) : exécute l'application
- [chifoumivue.h](#) : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- [chifoumivue.cpp](#) : déclaration des différents sous-programmes et slots.
- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- [chifoumi.h](#) : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- [presentation.cpp](#) : déclaration des différents sous-programmes et slots.
- [presentation.h](#) : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les image affichées

Description des fichiers .h

Nom de l'élément	Elément	Déclaration dans le code	Fonctionnement
etatPartie	Etat de la partie	chifoumi.h	Lorsqu'il est en état initial, bouton figure non cliquable (grisé) et devient cliquable lorsque etatPartie devient enCours
scoreJoueur	Score actuel du joueur	chifoumi.h	Se modifie selon le résultat du tour précédent et est affiché grâce à labelScoreJouer qui récupère l'information.
scoreMachine	Score actuel de la machine	chifoumi.h	Se modifie selon le résultat du tour

			précédent et est affiché grâce à labelScoreMachine qui récupère l'information.
coupJoueur	Dernier coup du joueur	chifoumi.h	Enregistre le choix de la figure du joueur qui est ensuite affiché dans labelFigureJoueur
coupMachine	Dernier coup de la machine	chifoumi.h	Enregistre la figure de la machine qui est ensuite affiché dans labelFigureMachine
pushButtonPierre	Bouton poussoir pour jouer la figure pierre	ui_chfoumivue.h	Appelle la fonction jouerPierre
pushButtonPapier	Bouton poussoir pour jouer la figure papier	ui_chfoumivue.h	Appelle la fonction jouerPapier
pushButtonCiseau	Bouton poussoir pour jouer la figure ciseau	ui_chfoumivue.h	Appelle la fonction jouerCiseau
pushButtonNouvellePartie	Bouton poussoir pour commencer une nouvelle partie	ui_chfoumivue.h	Appelle la fonction nouvellePartie()
getEtat()	Retourne l'état de la partie actuelle	chifoumi.h	Renvoie la valeur de la variable etatPartie de type unEtat.
getCoupJoueur()	Retourne le dernier coup du joueur	chifoumi.h	Renvoie la valeur de la variable coupJoueur de type figure
getCoupMachine()	Retourne le dernier coup de la machine	chifoumi.h	Renvoie la valeur de la variable coupJoueur de type figure
getScoreJoueur()	Retourne le score du joueur	chifoumi.h	Récupérer le score du joueur pour l'actualiser
getScoreMachine()	Retourne le score de la machine	chifoumi.h	Récupérer le score de la machine pour l'actualiser
determinerGagnant()	Fonction qui permet de déterminer le gagnant de la partie	chifoumi.h	Détermine le gagnant 'J' pour joueur et 'M' pour machine

genererUnCoup()	Procédure qui permet de générer un coup pour la machine	chifoumi.h	Retourne une figure aléatoire possible entre pierre, papier et ciseau pour la machine
setEtat()	Procédure qui permet de changer l'état de la partie	chifoumi.h	Initialise l'état de la partie
setCoupJoueur()	Procédure qui permet de changer le coup du joueur	chifoumi.h	Initialise le coup du joueur
setCoupMachine()	Procédure qui permet de changer le coup de la machine	chifoumi.h	Initialise le coup de la machine
setScoreJoueur()	Procédure qui permet de changer le score du joueur	chifoumi.h	Initialise le score du joueur
setScoreMacine()	Procédure qui permet de changer le score de la machine	chifoumi.h	Initialise le score de la machine
majScore()	Procédure qui permet de mettre à jour les scores du joueur et de la machine	chifoumi.h	Mise à jour des scores du joueur et de la machine
initScore()	Procédure qui permet d'initialiser les scores au début de la partie	chifoumi.h	Initialise à 0 les attributs scoreJoueur et scoreMachine
initCoup()	Procédure qui permet d'initialiser les figures au début de la partie	chifoumi.h	Initialise à rien les attributs coupJoueur et coupMachine
nvllConnexion()	Procédure qui permet une nouvelle connexion	chifoumivue.h	Créer la connexion de la vue avec la présentation
supprConnexion()	Procédure qui supprime la connexion	chifoumivue.h	Supprime la connexion de la vue avec la présentation
majInterface()	Procédure qui met à jour l'interface	chifoumivue.h	Contient les ordres de la vue destiné à la présentation

getModele()	Retourne le modele	presentation.h	Récupérer le modèle
setModele()	Associe la présentation à une variable de type chifoumi	presentation.h	Associe le modèle chifoumi à la présentation
getVue()	Procédure qui permet de retourner la vue	presentation.h	Récupérer la vue
setVue()	Procédure qui permet d'associer la présentation à une variable de type chifoumivue	chifoumivue.h	Associe la vue à la présentation
nouvellePartie()	Procédure qui permet de lancer une nouvelle partie	chifoumivue.h	Réinitialise les compteur du joueur et de la machine a 0, réinitialise les figure du joueur et de la machine a 'rien'
jouerPapier()	Permet à la machine de jouer la figure Papier	chifoumivue.h	CoupJouer est actualisée à papier et ensuite affiché dans labelFigureJouer
jouerCiseau()	Permet à la machine de jouer la figure Ciseau	chifoumivue.h	CoupJouer est actualisée à Ciseau et ensuite affiché dans labelFigureJouer
jouerPierre()	Permet à la machine de jouer la figure Pierre	chifoumivue.h	CoupJouer est actualisée à pierre et ensuite affiché dans labelFigureJouer

Tableau des tests

Elément testé : distribution des points selon les figures

Date : 13/05/2022

Version : v2

<u>Classe</u>	<u>Description</u>	<u>Forme choisie par le joueur</u>	<u>Forme générée pour la machine</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>
Égalité	Le joueur et la machine jouent la même figure	Ciseaux	Ciseaux	pas de modification des points	pas de modification des points
Égalité	Le joueur et la machine jouent la même figure	Pierre	Pierre	pas de modification des points	pas de modification des points
Égalité	Le joueur et la machine jouent la même figure	Feuilles	Feuilles	pas de modification des points	pas de modification des points
Victoire joueur	Le joueur choisit une figure qui bat la machine	Ciseaux	Feuilles	score du joueur +1	score du joueur +1
Victoire joueur	Le joueur choisit une figure qui bat la machine	Pierre	Ciseaux	score du joueur +1	score du joueur +1
Victoire joueur	Le joueur choisit une figure qui bat la machine	Feuilles	Pierre	score du joueur +1	score du joueur +1
Victoire machine	le joueur choisit une figure qui perd contre la machine	Ciseaux	Pierre	score de la machine +1	score de la machine +1

Victoire machine	le joueur choisit une figure qui perd contre la machine	Pierre	Feuilles	score de la machine +1	score de la machine +1
Victoire machine	le joueur choisit une figure qui perd contre la machine	Feuilles	Ciseaux	score de la machine +1	score de la machine +1

v3

Liste des fichiers présents :

- `main.cpp` : exécute l'application
- `chifoumivue.h` : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- `chifoumivue.cpp` : déclaration des différents sous-programmes et slots.

- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- [chifoumi.h](#) : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- [presentation.cpp](#) : déclaration des différents sous-programmes et slots.
- [presentation.h](#) : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les image affichées

Description des modifications des fichiers .h et .cpp

Nom de l'élément	Élément	Déclaration dans le code	Fonctionnement
aProposDe()	Procédure du menu aide de l'application	presentation.h	La procédure se lance au moment où le bouton aide est sélectionné pour afficher la section 'à propos de' disponible qui indique les noms et prénoms ainsi que le groupe de TD et de TP des membres du groupe.

Modification du fichier chifoumivue.ui : Action quitter ajouté au Menu-Fichier

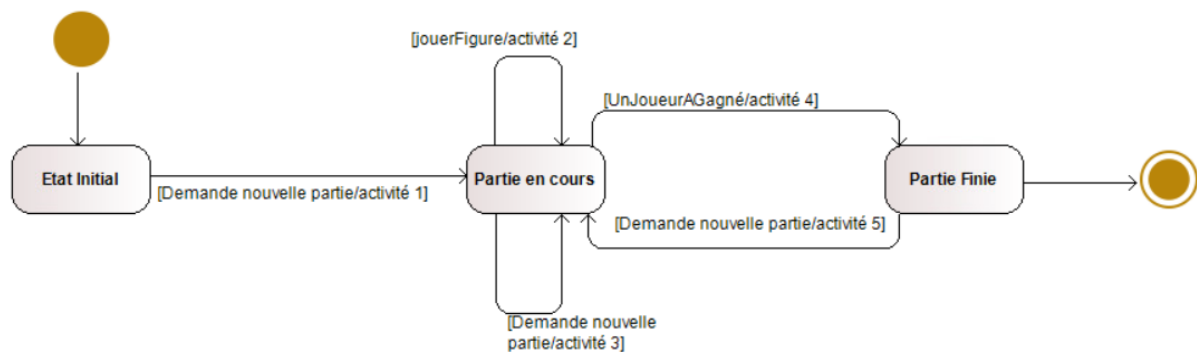
Ajouts dans le tableau des tests

<u>Classe</u>	<u>Description</u>	<u>Forme choisie par le joueur</u>	<u>Forme générée pour la machine</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>

Quitter	Le joueur sélectionne l'option 'quitter' dans le menu-Fichier de l'application	aucune	aucune	Fermer l'application	Fermer l'application
aProposDe	Le joueur sélectionne l'option 'a propos de' dans le menu-Aide de l'application	aucune	aucune	Affichage d'un onglet avec un texte qui affiche noms,prénoms, groupe de TD et TP des membres du groupe	Affichage d'un onglet avec un texte qui affiche noms,prénoms, groupe de TD et TP des membres du groupe

v4

Diagramme d'état :



Dictionnaire des états, événements et actions associées au diagramme d'activité :

Nom	Type	Description
-----	------	-------------

Etat initial	Etat	La fenêtre est affichée avec les boutons des figures indisponibles. Les scores sont à 0 et les joueurs ont “rien” comme figure
Partie en cours	Etat	Les boutons sont disponibles, les scores actuels sont affichés et les dernières figures du joueur et de la machine aussi
Partie Finie	Etat	Les boutons figures sont rendus indisponibles mais le reste reste le même
Jouer figure	Evenement	La figure du joueur est enregistrée comme étant celle choisie ; une figure est assignée à la machine de manière aléatoire ; comparaison des figures ; modification des scores ; mise à jour de l’affichage
Demande nouvelle partie	Evenement	Si l’état est initial : modification de l’état, les scores du joueur et de la machine sont remis à 0 et les figures sont modifiées vers “rien”, mise à jour de l’affichage
UnJoueurAGagné	Evenement	Événement lancé lorsque le score d’un des joueurs arrive à 5. Affiche la messageBox de victoire et désactive les boutons figures

Version matricielle du diagramme d’état-transition

	pushButtonCi seau	pushButtonP ierre	pushButtonp apier	pushButtonNouvell ePartie	
État actuel \ Évènem ent	Jouer figure			Demande nouvelle partie	Un Joueur A Gagné
État initial	-----			Activite 1	-----
Partie en cours	Activite 2			Activite 3	Activite 4
Partie Finie	-----			Activite 5	-----

Description des éléments d’interface

N° Element	Elément	Interface
------------	---------	-----------

1	labelScoreVainqueur	Label pour l'affichage du score à avoir pour gagner
2	labelVictoire	Label pour l'affichage du texte "Gagnant à"

Liste des fichiers présents :

- [main.cpp](#) : exécute l'application
- [chifoumivue.h](#) : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- [chifoumivue.cpp](#) : déclaration des différents sous-programmes et slots.
- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- [chifoumi.h](#) : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- [presentation.cpp](#) : déclaration des différents sous-programmes et slots.
- [presentation.h](#) : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les image affichées

Description des modifications des fichiers .h et .cpp

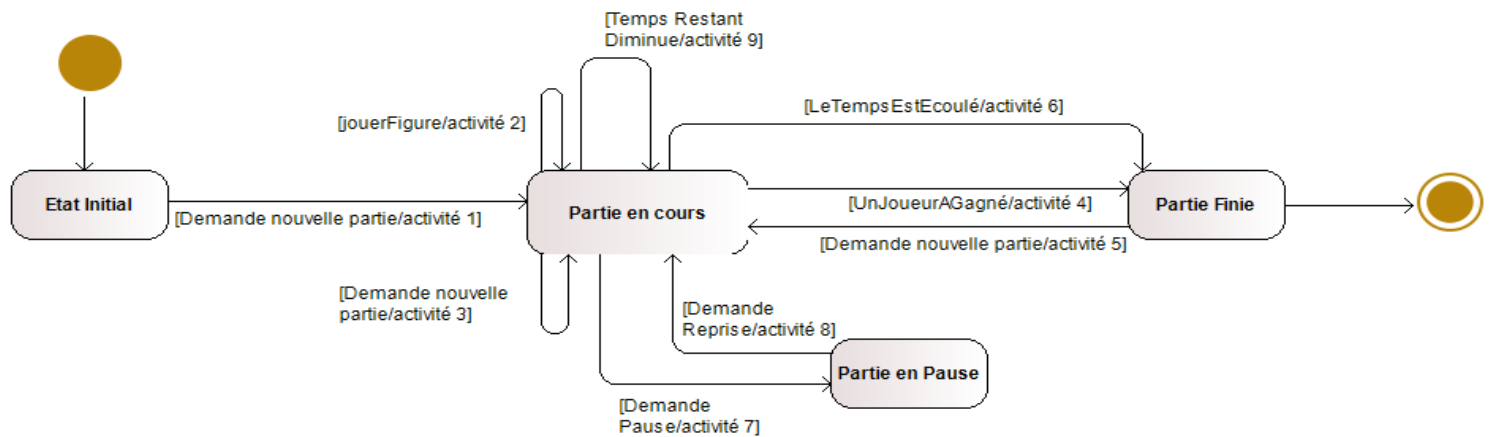
Nom de l'élément	Élément	Déclaration dans le code	Fonctionnement
unJoueurAGagne	Procédure qui permet	presentation.h	La procédure se lance au moment où le joueur ou la machine gagne la partie donc

	d'annoncer la victoire du joueur ou de la machine (défaite du joueur) et arrête la partie		atteint 5 points. De ce fait, l'état de la partie est changé à partieFinie et affiche un message de victoire ou de défaite au joueur.
--	---	--	---

Ajouts dans le tableau des tests

<u>Classe</u>	<u>Description</u>	<u>Forme choisie par le joueur</u>	<u>Forme générée pour la machine</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>
UnJoueur AGagne	Le joueur à gagné la partie	aucune	aucune	Affichage du message de victoire au joueur et arrêt de la partie en changeant l'état de la partie à 'partieFinie'	Affichage du message de victoire au joueur et arrêt de la partie en changeant l'état de la partie à 'partieFinie'
UnJoueur AGagne	Le joueur à perdu la partie	aucune	aucune	Affichage du message de défaite au joueur et arrêt de la partie en changeant l'état de la partie à 'partieFinie'	Affichage du message de défaite au joueur et arrêt de la partie en changeant l'état de la partie à 'partieFinie'

Diagramme d'état :



Dictionnaire des états, événements et actions associées au diagramme d'activité :

Nom	Type	Description
Etat initial	Etat	La fenêtre est affichée avec les boutons des figures indisponibles. Les scores sont à 0 et les joueurs ont "rien" comme figure
Partie en cours	Etat	Les boutons sont disponibles, les scores actuels sont affichés et les dernières figures du joueur et de la machine aussi
Partie Finie	Etat	Les boutons figures sont rendus indisponibles mais le reste reste le même
Partie en Pause	Etat	Tous les boutons sauf celui de reprise du timer sont désactivés
Jouer figure	Evenement	La figure du joueur est enregistrée comme étant celle choisie ; une figure est assignée à la machine de manière aléatoire ; comparaison des figures ; modification des scores ; mise à jour de l'affichage
Demande nouvelle partie	Evenement	Si l'état est initial : modification de l'état, les scores du joueur et de la machine sont remis à 0 et les figures sont modifiées vers "rien", mise à jour de l'affichage
UnJoueurAGagné	Evenement	Événement lancé lorsque le score d'un des

		joueurs arrive à 5. Affiche la messageBox de victoire et passe le programme à l'état final
LeTempsEstEcoulé	Evenement	Lorsque le timer est fini. Affiche la messageBox de fin de timer et passe le programme à l'état final
Demande Pause	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Demande reprise	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Temps Restant Diminue	Evenement	Une seconde est passée, le temps restant est modifié.

Version matricielle du diagramme d'état-transition

	pushButtonCiseau	pushButtonPierre	pushButtonPapier	pushButtonNouvellePartie			pushButtonPause	pushButtonPause	
État actuel \ Évènement	Jouer figure			Demande nouvelle partie	Un Joueur A Gagné	Le Temps est écoulé	Demande Pause	Demande Reprise	Temps Restant Diminue
État initial	-----			Activite 1	----- ---	-----	-----	-----	-----
Partie en cours	Activite 2			Activite 3	Activite 4	Activite 6	Activite 7	-----	Activite 9
Partie Finie	-----			Activite 5	----- ---	-----	-----	-----	-----
Partie en pause	-----			-----	----- ---	-----	-----	Activite 8	-----

Description des éléments d'interface

N° Element	Elément	Interface
1	pushButtonPause	Bouton permettant de mettre en pause la partie courante.

Liste des fichiers présents :

- [main.cpp](#) : exécute l'application
- [chifoumivue.h](#) : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- [chifoumivue.cpp](#) : déclaration des différents sous-programmes et slots.
- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- [chifoumi.h](#) : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- [presentation.cpp](#) : déclaration des différents sous-programmes et slots.
- [presentation.h](#) : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les image affichées

Description des modification des fichiers .h et .cpp

Nom de l'élément	Elément	Déclaration dans le code	Fonctionnement
1	_leTimer	presentation.h	Appel de la fonction _leTimer toute les secondes
2	demandePause()	presantation.h	Changement de l'état de la partie en pause et arrêt du timer (UnEtat = enPause)
3	demandeReprise()	presantation.h	Changement de l'état de la partie en cours et reprise du timer (UnEtat = enCours)
4	tempsRestantDiminue()	presantation.h	Changement du modèle en continue jusqu'à ce que le timer soit à 0

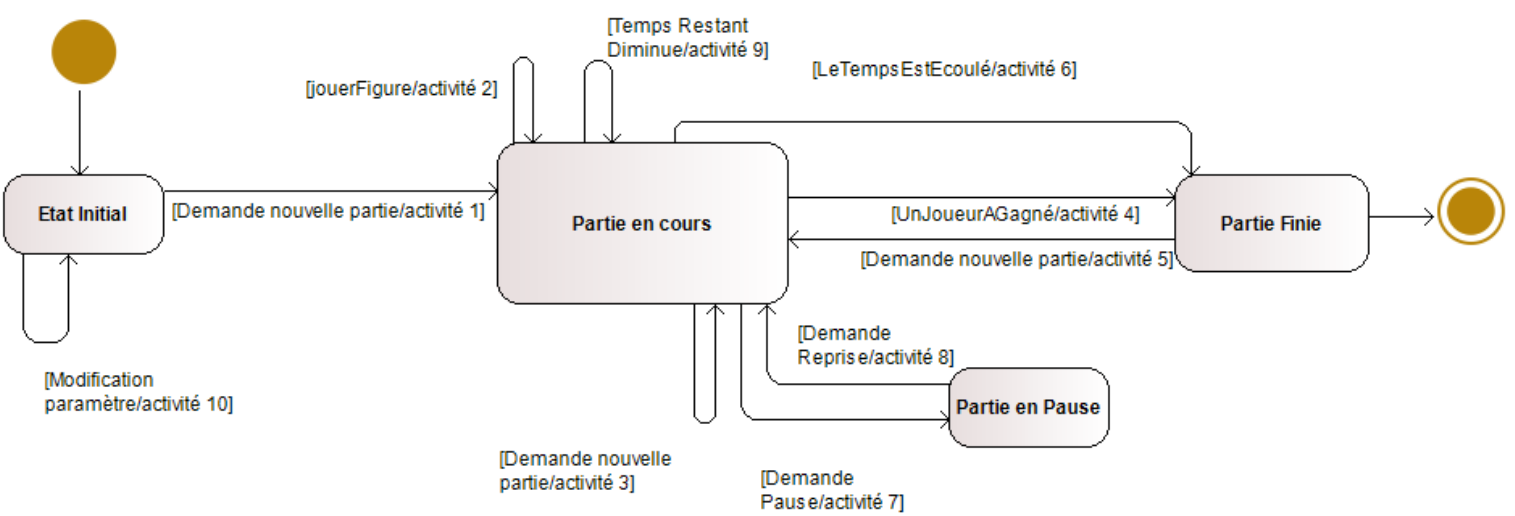
Ajouts dans le tableau des tests

Classe	Description	Forme choisie par le joueur	Forme générée pour la machine	Résultats attendus	Résultats obtenus
demande Pause	Le joueur appuie sur le bouton pause	aucune	aucune	Met en pause la partie : met sur pause le timer, et met l'état de la partie setEtat à enPause	Met en pause la partie : met sur pause le timer, et met l'état de la partie setEtat à enPause
demande Reprise	Le joueur appuie sur le bouton pause avec un libellé différents qui devient reprise	aucune	aucune	Met en pause la partie : reprend le temps du timer après la pause, et met l'état de la partie setEtat à enCours	Met en pause la partie : reprend le temps du timer après la pause, et met l'état de la partie setEtat à enCours

tempsRestantDiminue	Le timer se décrémente de 1 s'il n'est pas égal mais supérieur à 0	aucune	aucune	Met à jour la vue avec le timer qui se décrémente	Met à jour la vue avec le timer qui se décrémente
---------------------	--	--------	--------	---	---

v6

Diagramme d'état :



Dictionnaire des états, événements et actions associées au diagramme d'activité :

Nom	Type	Description
Etat initial	Etat	La fenêtre est affichée avec les boutons des figures indisponibles. Les scores sont à 0 et les joueurs ont “rien” comme figure
Partie en cours	Etat	Les boutons sont disponibles, les scores actuels sont affichés et les dernières figures du joueur et de la machine aussi
Partie Finie	Etat	Les boutons figures sont rendus indisponibles mais le reste reste le même
Partie en Pause	Etat	Tous les boutons sauf celui de reprise du timer

		sont désactivés
Jouer figure	Evenement	La figure du joueur est enregistrée comme étant celle choisie ; une figure est assignée à la machine de manière aléatoire ; comparaison des figures ; modification des scores ; mise à jour de l'affichage
Demande nouvelle partie	Evenement	Si l'état est initial : modification de l'état, les scores du joueur et de la machine sont remis à 0 et les figures sont modifiées vers "rien", mise à jour de l'affichage
UnJoueurAGagné	Evenement	Événement lancé lorsque le score d'un des joueurs arrive à 5. Affiche la messageBox de victoire et passe le programme à l'état final
LeTempsEstEcoulé	Evenement	Lorsque le timer est fini. Affiche la messageBox de fin de timer et passe le programme à l'état final
Demande Pause	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Demande reprise	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Temps Restant Diminue	Evenement	Une seconde est passée, le temps restant est modifié.
Modification Paramètre	Evenement	Ouverture d'une fenêtre de dialogue dont la fermeture entraîne une modification des paramètres

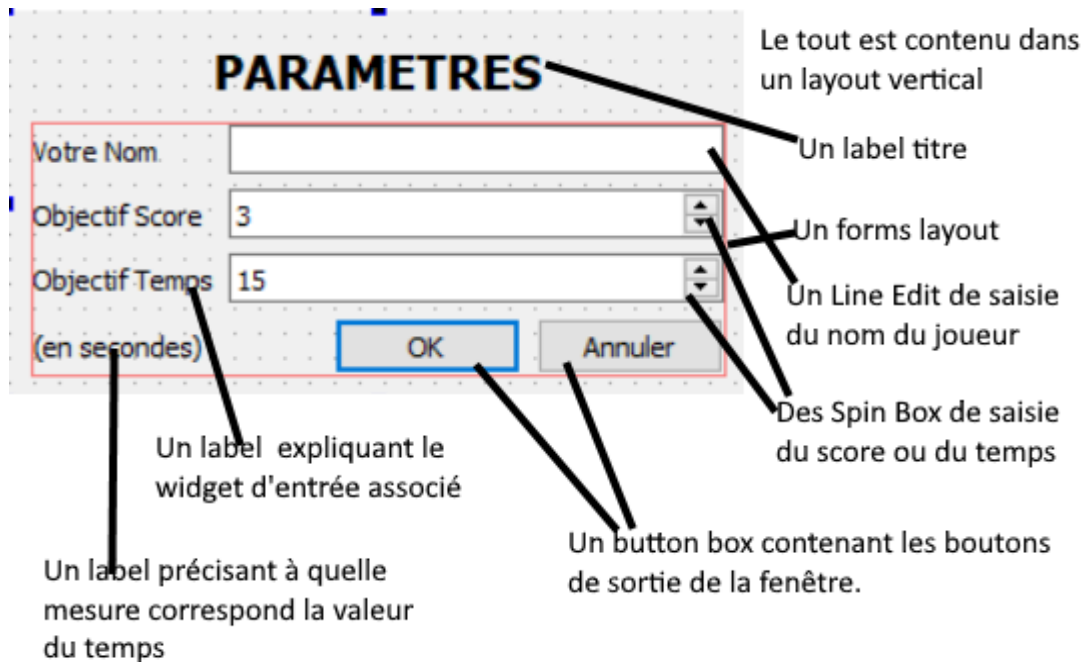
Version matricielle du diagramme d'état-transition

	pushButton Ciseau	pushButton Pierre	pushButton papier	pushButton NouvellePa rtie			pushButtonPaus e	pushButtonPause		Bouton de menu "Paramétrer"
État actuel \ Évènement	Jouer figure			Demande nouvelle partie	Un Joueur A Gagné	Le Temps est écoulé	Demande Pause	Demande Reprise	Temps Restant Diminue	Modification Paramètre
État initial	-----			Activite 1	----- ---	----- --	-----	-----	----- --	Activité 10
Partie en cours	Activite 2			Activite 3	Activite 4	Activite 6	Activite 7	-----	Activite 9	-----
Partie Finie	-----			Activite 5	----- ---	----- --	-----	-----	----- ---	-----
Partie en pause	-----			-----	----- ---	----- --	-----	Activite 8	----- ---	-----

Description des éléments d'interface

N° Element	Elément	Interface
1	labelNom	Label contenant la chaîne de caractère "Votre Nom"
2	labelScore	Label contenant la chaîne de caractère "Objectif Score"
3	labelTps	Label contenant la chaîne de caractère "Objectif Temps"
4	label_2	Label contenant la chaîne de caractère "(en seconde)"
5	buttonBox	Contient 2 boutons : un bouton 'Ok' et un bouton 'Cancel'
6	lineEditNom	Zone de texte qui contiendra le nom du joueur une fois saisi
7	spinBoxScore	Menu déroulant contenant des entiers compris entre 3 et 99 (prédéfini à 3) correspondant au score que doit atteindre le joueur ou la machine afin de remporter la partie
8	spinBoxTps	Menu déroulant contenant des entiers compris entre 15 et 99 (prédéfini à 3) correspondant au temps que ne doit pas dépasser la partie

Disposition et Description des éléments d'interface :



Liste des fichiers présents :

- [main.cpp](#) : exécute l'application
- [chifoumivue.h](#) : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- [chifoumivue.cpp](#) : déclaration des différents sous-programmes et slots.
- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- [chifoumi.h](#) : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- [presentation.cpp](#) : déclaration des différents sous-programmes et slots.
- [presentation.h](#) : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- [parametre.h](#) : initialisation de la classe Parametre avec ses variables, sous-programmes et slots QT.
- [parametre.cpp](#) : déclaration des différents sous-programmes et slots.

- [Parametre.ui](#) : fichier permettant la disposition des éléments d'interfaces de la fenêtre de dialogue ou le joueur saisit des paramètres pour modifier la partie (son nom, le nombre de points maximum pour gagner et un temps maximum pour le timer précédemment créé.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les image affichées

Description des modifications des fichiers .h et .cpp

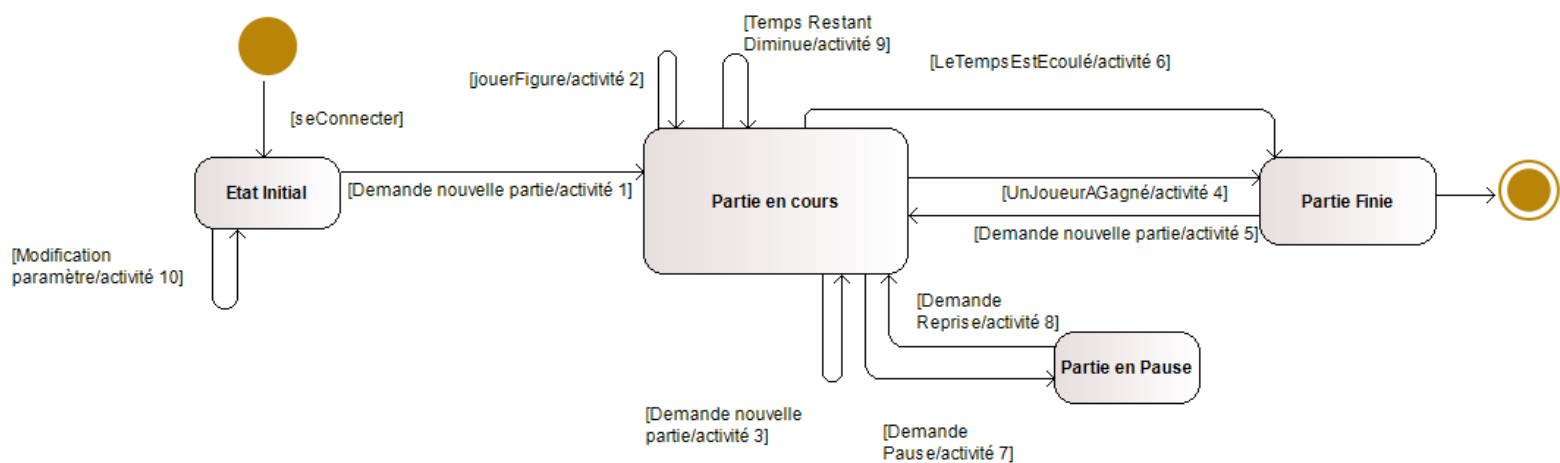
Nom de l'élément	Élément	Déclaration dans le code	Fonctionnement
modificationParametre()	Procédure qui permet de créer la boîte de dialogue et la redimensionne	Presentation.h	Création d'une boîte de dialogue avec une taille prédéfini et l'affiche à l'écran pour permettre au joueur de modifier certains paramètres de la partie
modifierParametres()	Procédure qui permet de modifier les paramètres du modèle pour la partie	Présentation.h	Modification du modèle selon les différents paramètres saisis ou choisis dans la boîte de dialogue

Nouveau tableau de test

<u>Classe</u>	<u>Description</u>	<u>Valeur modifiée</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>
nomJoueur	Nom saisi par le joueur	label Joueur	Affichage du nom saisi au dessus de la figure choisie du joueur	Affichage du nom saisi au dessus de la figure choisie du joueur
strScore	Valeur nécessaire choisie par le joueur pour la fin de partie	labelScore Vainqueur	Fin de la partie lorsque le joueur ou la machine atteigne ce score	Fin de la partie lorsque le joueur ou la machine atteigne ce score
strTps	Temp choisie par le joueur pour définir une fin de partie si le score n'a pas été atteint	labelNbTpsRestant	Fin de partie lorsque le compteur modifier atteint zéro	Fin de partie lorsque le compteur modifier atteint zéro
modifierParametres()	Slot regroupant les valeurs introduite du temps, du score et du nom du joueur	Modification effectuée avant le début de la partie	Les temps avant la fin de partie ,le score à atteindre et le nom du joueur ont été modifié	Les temps avant la fin de partie ,le score à atteindre et le nom du joueur ont été modifié
modifierParametres()	Slot regroupant les valeurs introduite du temps, du score et du nom du joueur	Modification effectuée pendant que la partie est en cours	Les temps avant la fin de partie ,le score à atteindre et le nom du joueur n'ont pas été modifié	Les temps avant la fin de partie ,le score à atteindre et le nom du joueur n'ont pas été modifié

v7

Diagramme d'état :



Dictionnaire des états, événements et actions associées au diagramme d'activité :

Nom	Type	Description
Etat initial	Etat	La fenêtre est affichée avec les boutons des figures indisponibles. Les scores sont à 0 et les joueurs ont "rien" comme figure
Partie en cours	Etat	Les boutons sont disponibles, les scores actuels sont affichés et les dernières figures du joueur et de la machine aussi
Partie Finie	Etat	Les boutons figures sont rendus indisponibles mais le reste reste le même

Partie en Pause	Etat	Tous les boutons sauf celui de reprise du timer sont désactivés
Jouer figure	Evenement	La figure du joueur est enregistrée comme étant celle choisie ; une figure est assignée à la machine de manière aléatoire ; comparaison des figures ; modification des scores ; mise à jour de l’affichage
Demande nouvelle partie	Evenement	Si l’état est initial : modification de l’état, les scores du joueur et de la machine sont remis à 0 et les figures sont modifiées vers “rien”, mise à jour de l’affichage
UnJoueurAGagné	Evenement	Événement lancé lorsque le score d’un des joueurs arrive à 5. Affiche la messageBox de victoire et passe le programme à l’état final
LeTempsEstEcoulé	Evenement	Lorsque le timer est fini. Affiche la messageBox de fin de timer et passe le programme à l’état final
Demande Pause	Evenement	L’état est passé à “en Pause” et l’affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Demande reprise	Evenement	L’état est passé à “en Pause” et l’affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Temps Restant Diminue	Evenement	Une seconde est passée, le temps restant est modifié.
Modification Paramètre	Evenement	Ouverture d’une fenêtre de dialogue dont la fermeture entraîne une modification des paramètres
Se Connecter	Action	Action nécessaire à l’ouverture de l’application pour accéder au jeu

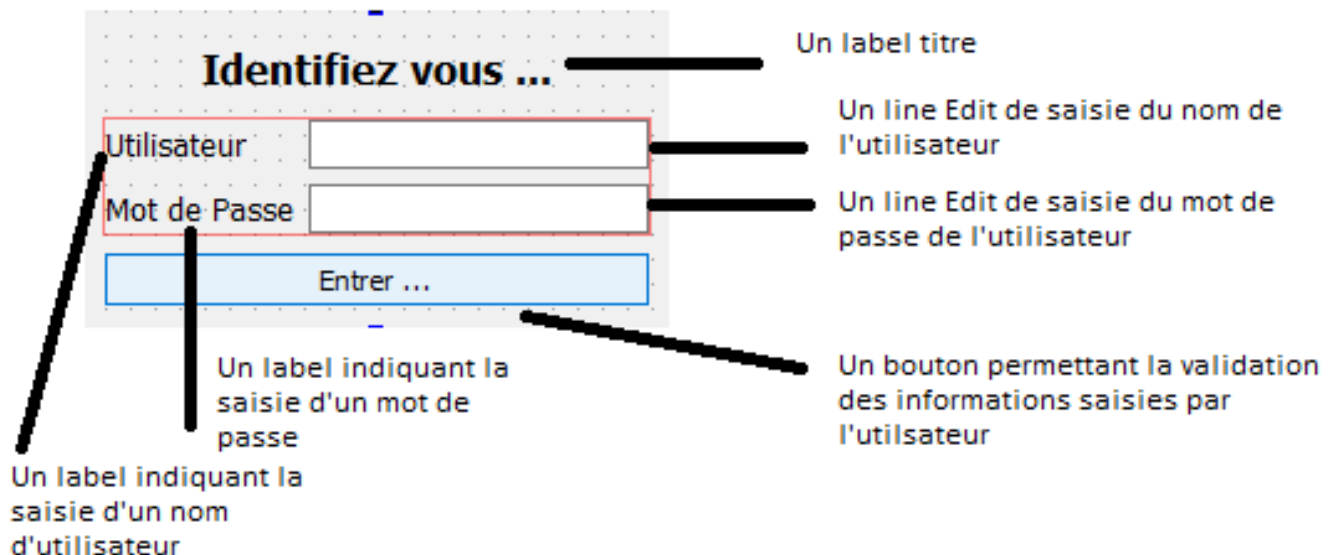
Version matricielle du diagramme d'état-transition

	pushBut tonCise au	pushButt onPierre	pushButt onpapier	pushButto nNouvelle Partie			pushButtonPause	pushButtonPause		Bouton de menu "Paramétrer"
État actuel \ Évènement	Jouer figure			Demande nouvelle partie	Un Joueur A Gagné	Le Temps est écoulé	Demande Pause	Demande Reprise	Temps Restant Diminue	Modification Paramètre
État initial	-----			Activite 1	----- --	----- --	-----	-----	-----	Activité 10
Partie en cours	Activite 2			Activite 3	Activite 4	Activite 6	Activite 7	-----	Activite 9	-----
Partie Finie	-----			Activite 5	----- --	----- -	-----	-----	-----	-----
Partie en pause	-----			-----	----- -	----- -	-----	Activite 8	-----	-----

Description des éléments d'interface

N° Element	Elément	Interface
1	labelUtilisateur	Label contenant la chaîne de caractère "Utilisateur"
2	labelMDP	Label contenant la chaîne de caractère "Mot de Passe"
3	lineEditUtilisateur	Zone de texte contenant le nom de l'utilisateur saisie au clavier
4	lineEditMDP	Zone de texte contenant le mot de passe de l'utilisateur saisie au clavier
5	labelTitre	Label pour l'affichage du titre de la fenêtre
6	pushButtonEntrer	Bouton poussoir pour valider les informations entrées

Disposition et Description des éléments d'interface :



Liste des fichiers présents :

- `main.cpp` : exécute l'application
- `chifoumivue.h` : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- `chifoumivue.cpp` : déclaration des différents sous-programmes et slots.
- `chifoumivue.ui` : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- `chifoumi.cpp` : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- `chifoumi.h` : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- `presentation.cpp` : déclaration des différents sous-programmes et slots.
- `presentation.h` : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- `parametre.h` : initialisation de la classe Parametre avec ses variables, sous-programmes et slots QT.
- `parametre.cpp` : déclaration des différents sous-programmes et slots.
- `parametre.ui` : fichier permettant la disposition des éléments d'interfaces de la fenêtre de dialogue où le joueur saisit des paramètres pour modifier la partie (son nom, le nombre de points maximum pour gagner et un temps maximum pour le timer précédemment créé).
- `identification.h` : initialisation de la classe Identification avec ses variables, sous-programmes et slots QT.
- `identification.cpp` : déclaration des différents sous-programmes et slots.
- `identification.ui` : fichier permettant la disposition des éléments d'interfaces de la fenêtre de dialogue où le joueur saisit des paramètres pour modifier la partie (son nom d'utilisateur et son mot de passe pour s'enregistrer).
- `database.h` : initialisation de la classe Identification avec ses variables, sous-programmes et slots QT.
- `database.cpp` : déclaration des différents sous-programmes et slots.

- [images.qrc](#) : fichier ressource nécessaire à l’affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les image affichées

Description des modifications des fichiers .h et .cpp

Nom de l’élément	Élément	Déclaration dans le code	Fonctionnement
fermerFenetre()	Procédure qui permet de fermer la fenêtre	Identification.h	Déconnecte la fenêtre de la présentation et ferme la fenêtre
tentativeConnexion()	Signal permettant d’établir la connexion avec la présentation et la fenêtre	Identification.h	Sert de lien entre la présentation et la fenêtre
userTentativeConnexion()	Procédure qui permet de faire une tentative de connexion dans la fenêtre d’identification par l’utilisateur	Identification.h	Initialise 2 booléens à ‘false’, vérifie que l’utilisateur a entré les 2 champs demandé et si c’est le cas les 2 booléens passent à ‘true’ et pour finir, si les 2 booléens sont à vrai, alors, on émet un signal (tentativeConnexion) que la présentation recevra
openDatabase()	Fonction qui permet d’ouvrir la base de données et renvoie un booléen	Database.h	On ajoute une connexion ODBC vers une base de données, on identifie l’hôte de la base, on définit le nom de la base, on rentre le pseudonyme et le mot de passe d’un utilisateur, on demande à ouvrir l’accès à la

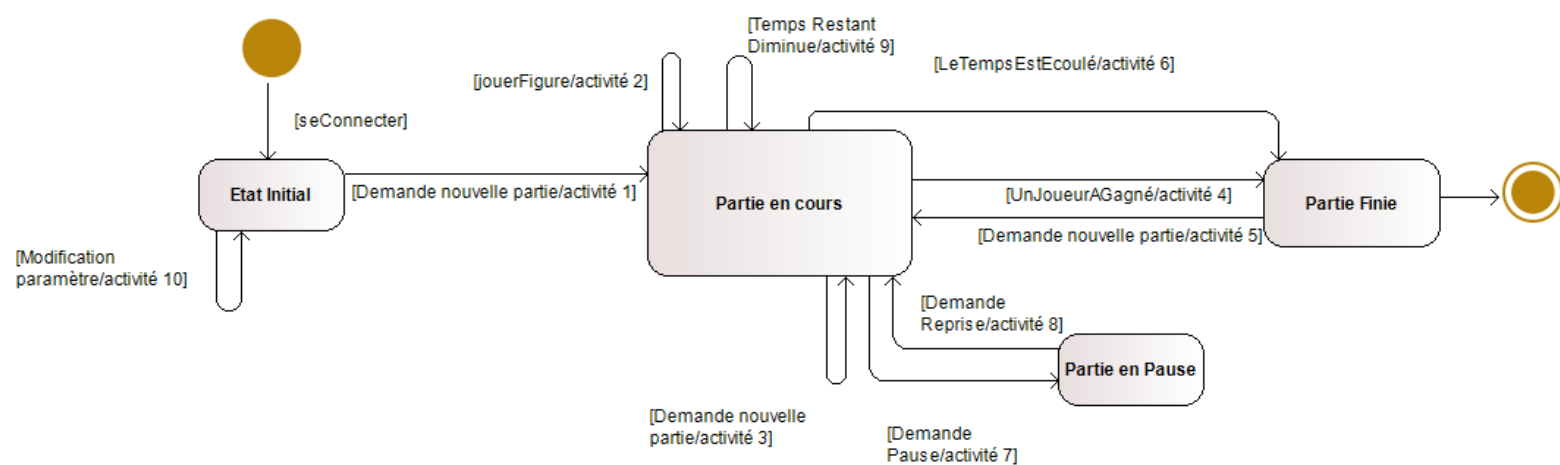
			base, on retourne si l'accès a pu être ouverte.
closeDatabase()	Procédure qui permet de fermer la base de données	Database.h	On demande à fermer l'accès à la base de données
restoreDatabase()	Fonction qui permet d'insérer une table dans la base de données si elle existe et si la table n'existe pas	Database.h	On initialise les variables, On vérifie que la BD existe, On vérifie qu'elle contienne la table Utilisateur, Si la table n'existe pas on la crée, S'il y a un problème dans la création, on affiche un message dans l'interface de commande, on retourne si la table a été modifiée
tentativeConnexion()	Fonction qui permet à l'utilisateur de faire une tentative de connexion à l'aide d'un identifiant et un mot de passe qu'il peut entrer sur une page de dialogue	Database.h	On initialise deux booléens: connexionReussie et commandeExecutee, on récupère le nom de l'utilisateur et on interroge avec une commande SQL pour savoir si le nom est présent dans la base. Si la connexion est réussie et que la commande SQL est exécutée alors l'utilisateur peut se connecter à l'application. On retourne si la connexion est réussie.

Nouveau tableau de test

<u>Description</u>	<u>Action réalisé</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>
On saisit un identifiant et un mot de passe qui seront ensuite vérifiés s'ils se trouvent dans la base de données.	Saisie d'un identifiant et d'un mot de passe se trouvant dans la base de donnée	Identifiant et mot de passe correct. Ouverture de la fenêtre principal du jeu	Identifiant et mot de passe correct. Ouverture de la fenêtre principal du jeu
On saisit un identifiant et un mot de passe qui seront ensuite vérifiés s'ils se trouvent dans la base de données.	Saisie d'un identifiant et d'un mot de passe ne se trouvant pas dans la base de donnée	Identifiant et mot de passe incorrect. Aucun changement, aucune réaction.	Identifiant et mot de passe incorrect. Aucun changement, aucune réaction.

v8

Diagramme d'état :



Dictionnaire des états, événements et actions associées au diagramme d'activité :

Nom	Type	Description
Etat initial	Etat	La fenêtre est affichée avec les boutons des figures indisponibles. Les scores sont à 0 et les joueurs ont "rien" comme figure
Partie en cours	Etat	Les boutons sont disponibles, les scores actuels sont affichés et les dernières figures du joueur et de la machine aussi
Partie Finie	Etat	Les boutons figures sont rendus indisponibles mais le reste reste le même
Partie en Pause	Etat	Tous les boutons sauf celui de reprise du timer sont désactivés
Jouer figure	Evenement	La figure du joueur est enregistrée comme étant celle choisie ; une figure est assignée à la machine de manière aléatoire ; comparaison des figures ; modification des scores ; mise à jour de l'affichage
Demande nouvelle partie	Evenement	Si l'état est initial : modification de l'état, les scores du joueur et de la machine sont remis à 0 et les figures sont modifiées vers "rien", mise à jour de l'affichage
UnJoueurAGagné	Evenement	Evenement lancé lorsque le score d'un des joueurs arrive à 5. Affiche la messageBox de victoire et passe le programme à l'état final
LeTempsEstEcoulé	Evenement	Lorsque le timer est fini. Affiche la messageBox de fin de timer et passe le programme à l'état final
Demande Pause	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Demande reprise	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Temps Restant	Evenement	Une seconde est passée, le temps restant est

Diminue		modifié.
Modification Paramètre	Evenement	Ouverture d'une fenetre de dialogue dont la fermeture entraîne une modification des paramètres
Se Connecter	Action	Action nécessaire à l'ouverture de l'application pour accéder au jeu
Ajouter Partie BD	Evenement	Ajout de la partie finie à la base de données

Version matricielle du diagramme d'état-transition

	pushBut tonCise au	pushButt onPierre	pushButt onpapier	pushButto nNouvelle Partie			pushButton Pause	pushButto nPause		Bouton de menu "Paramétrer "	
État actuel \ Évènement	Jouer figure			Demande nouvelle partie	Un Joueur A Gagné	Le Temps est écoulé	Demande Pause	Demande Reprise	Temps Restant Diminue	Modification Paramètre	Ajouter Partie BD
État initial	----- --			Activite 1	----- --	----- --	----- -	----- --	----- -	Activité 10	-----
Partie en cours	Activite 2			Activite 3	Activite 4	Activite 6	Activite 7	----- --	Activite 9	----- --	-----
Partie Finie	-----			Activite 5	----- --	----- -	-----	----- --	----- --	----- --	Activite 11
Partie en pause	-----			-----	----- -	----- -	-----	Activite 8	----- --	----- -	-----

Liste des fichiers présents :

- [main.cpp](#) : exécute l'application
- [chifoumivue.h](#) : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- [chifoumivue.cpp](#) : déclaration des différents sous-programmes et slots.
- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- [chifoumi.h](#) : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- [presentation.cpp](#) : déclaration des différents sous-programmes et slots.
- [presentation.h](#) : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- [parametre.h](#) : initialisation de la classe Parametre avec ses variables, sous-programmes et slots QT.
- [parametre.cpp](#) : déclaration des différents sous-programmes et slots.
- [parametre.ui](#) : fichier permettant la disposition des éléments d'interfaces de la fenêtre de dialogue ou le joueur saisit des paramètres pour modifier la partie (son nom, le nombre de points maximum pour gagner et un temps maximum pour le timer précédemment créé).
- [identification.h](#) : initialisation de la classe Identification avec ses variables, sous-programmes et slots QT.
- [identification.cpp](#) : déclaration des différents sous-programmes et slots.
- [identification.ui](#) : fichier permettant la disposition des éléments d'interfaces de la fenêtre de dialogue ou le joueur saisit des paramètres pour modifier la partie (son nom d'utilisateur et son mot de passe pour s'enregistrer).
- [database.h](#) : initialisation de la classe Identification avec ses variables, sous-programmes et slots QT.
- [database.cpp](#) : déclaration des différents sous-programmes et slots.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.

- **images** : dossier contenant les image affichées

Description des modifications des fichiers .h et .cpp

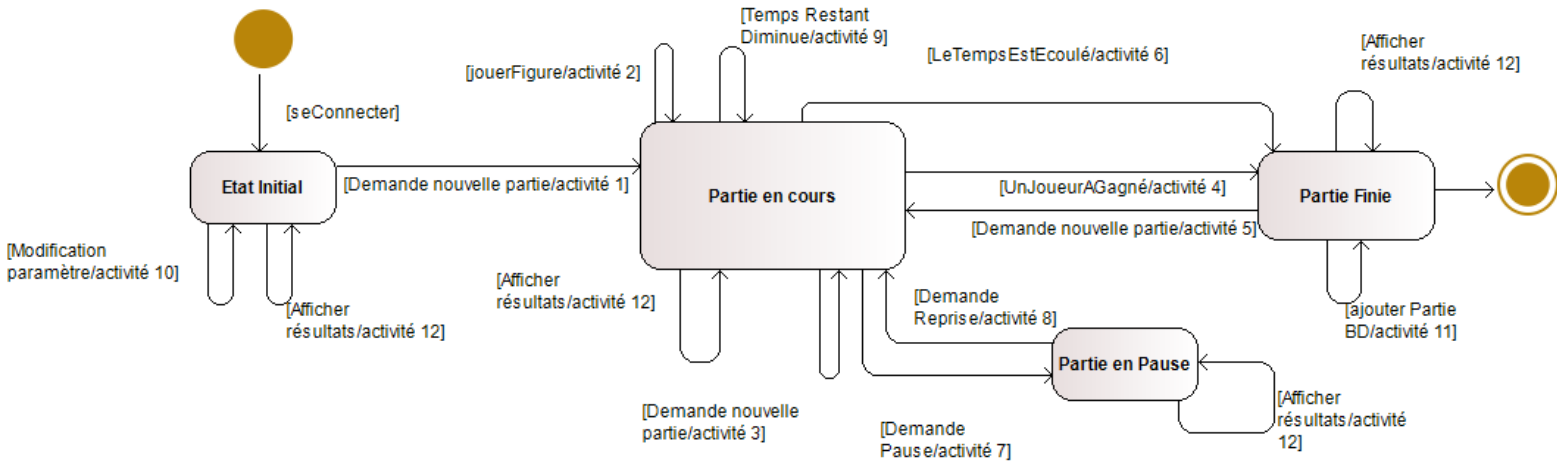
Nom de l'élément	Elément	Déclaration dans le code	Fonctionnement
ajouterPartie()	Procédure qui permet d'ajouter une partie terminée à la base de données avec différents paramètres	Database.h	On récupère la liste contenant toutes les parties, on recherche à l'aide d'une commande SQL le nombre de parties inscrites dans la base, si la commande réussit, on prépare la commande, on place les paramètres dans la commande, on demande d'exécuter la commande. Si la commande ne s'exécute pas correctement, on le signale dans l'interface de commande.
ajouterPartieBD()	Procédure pour préparer les paramètres et lancer la fonction ajouterPartie() de la classe Database	Presentation.h	On initialise les variables, On place les valeurs dans les variables, On lance la procédure ajouterPartie() de la classe Database avec en paramètre les variables.

Nouveau tableau de test

<u>Description</u>	<u>Action réalisé</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>
On saisit un identifiant et un mot de passe qui seront ensuite vérifiés s'ils se trouvent dans la base de données. Lorsque la partie est terminée les score sont enregistrés dans la base de donnée	Saisie d'un identifiant et d'un mot de passe se trouvant dans la base de données. L'utilisateur joue jusqu'à ce que cette dernière se termine (par manque de temps ou atteinte du score fixé).	Enregistrement des résultats de la partie dans la base de donne, les résultats sont liés avec le numéro de l'identifiant du joueur.	Enregistrement des résultats de la partie dans la base de donne, les résultats sont liés avec le numéro de l'identifiant du joueur.
On saisit un identifiant et un mot de passe qui seront ensuite vérifiés s'ils se trouvent dans la base de données. Lorsque la partie est terminée les score sont enregistrés dans la base de donnée	Saisie d'un identifiant et d'un mot de passe se trouvant dans la base de données. L'utilisateur commence une partie mais ne la termine pas	Les données de la partie sont perdues et ne sont pas enregistrés dans la base de donne	Les données de la partie sont perdues et ne sont pas enregistrés dans la base de donne

v9

Diagramme d'état :



Dictionnaire des états, événements et actions associées au diagramme d'activité :

Nom	Type	Description
Etat initial	Etat	La fenêtre est affichée avec les boutons des figures indisponibles. Les scores sont à 0 et les joueurs ont "rien" comme figure
Partie en cours	Etat	Les boutons sont disponibles, les scores actuels sont affichés et les dernières figures du joueur et de la machine aussi
Partie Finie	Etat	Les boutons figures sont rendus indisponibles mais le reste reste le même
Partie en Pause	Etat	Tous les boutons sauf celui de reprise du timer sont désactivés
Jouer figure	Evenement	La figure du joueur est enregistrée comme étant celle choisie ; une figure est assignée à la machine de manière aléatoire ; comparaison des figures ; modification des scores ; mise à jour de l'affichage
Demande nouvelle	Evenement	Si l'état est initial : modification de l'état, les

partie		scores du joueur et de la machine sont remis à 0 et les figures sont modifiées vers "rien", mise à jour de l'affichage
UnJoueurAGagné	Evenement	Événement lancé lorsque le score d'un des joueurs arrive à 5. Affiche la messageBox de victoire et passe le programme à l'état final
LeTempsEstEcoulé	Evenement	Lorsque le timer est fini. Affiche la messageBox de fin de timer et passe le programme à l'état final
Demande Pause	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Demande reprise	Evenement	L'état est passé à "en Pause" et l'affichage est mis à jour. On modifie le texte du bouton de Pause ainsi que sa connection
Temps Restant Diminue	Evenement	Une seconde est passée, le temps restant est modifié.
Modification Paramètre	Evenement	Ouverture d'une fenêtre de dialogue dont la fermeture entraîne une modification des paramètres
Se Connecter	Action	Action nécessaire à l'ouverture de l'application pour accéder au jeu
Ajouter Partie BD	Evenement	Ajout de la partie finie à la base de données
Afficher Résultats	Evenement	Affichage des 10 meilleurs résultats contre la machine

Version matricielle du diagramme d'état-transition

	push Butto nCise au	pushB uttonPi erre	pushButton papier	pushButtonNou vellePartie			pushButton Pause	pushButton Pause		Bouton de menu "Paramétrer"	
État actuel \ Évènement	Jouer figure			Demande nouvelle partie	Un Joueur A Gagné	Le Temps est écoulé	Demande Pause	Demande Reprise	Temps Restant Diminue	Modification Paramètre	Ajouter Partie BD
État initial	-----			Activite 1	-----	----- --	----- -	-----	-----	Activité 10	----- -
Partie en cours	Activite 2			Activite 3	Activite 4	Activite 6	Activite 7	-----	Activite 9	-----	----- -
Partie Finie	-----			Activite 5	-----	----- --	-----	-----	-----	-----	Activite 11
Partie en pause	-----			-----	-----	----- --	-----	Activite 8	-----	-----	----- -

Liste des fichiers présents :

- [main.cpp](#) : exécute l'application
- [chifoumivue.h](#) : initialisation de la classe ChifoumiVue avec ses variables, sous-programmes et slots QT
- [chifoumivue.cpp](#) : déclaration des différents sous-programmes et slots.
- [chifoumivue.ui](#) : fichier permettant la disposition des éléments d'interfaces de l'application de manière graphique.
- [chifoumi.cpp](#) : déclaration des différents sous-programmes et slots afin que le jeu fonctionne.
- [chifoumi.h](#) : initialisation des différents sous-programmes et slots afin que le jeu fonctionne.
- [presentation.cpp](#) : déclaration des différents sous-programmes et slots.
- [presentation.h](#) : initialisation de la classe Presentation avec ses variables, sous-programmes et slots QT.
- [parametre.h](#) : initialisation de la classe Parametre avec ses variables, sous-programmes et slots QT.
- [parametre.cpp](#) : déclaration des différents sous-programmes et slots.
- [parametre.ui](#) : fichier permettant la disposition des éléments d'interfaces de la fenêtre de dialogue où le joueur saisit des paramètres pour modifier la partie (son nom, le nombre de points maximum pour gagner et un temps maximum pour le timer précédemment créé).
- [identification.h](#) : initialisation de la classe Identification avec ses variables, sous-programmes et slots QT.
- [identification.cpp](#) : déclaration des différents sous-programmes et slots.
- [identification.ui](#) : fichier permettant la disposition des éléments d'interfaces de la fenêtre de dialogue où le joueur saisit des paramètres pour modifier la partie (son nom d'utilisateur et son mot de passe pour s'enregistrer).
- [database.h](#) : initialisation de la classe Identification avec ses variables, sous-programmes et slots QT.
- [database.cpp](#) : déclaration des différents sous-programmes et slots.
- [images.qrc](#) : fichier ressource nécessaire à l'affichage correct des images
- [S201.pro](#) : fichier permettant de modifier le projet grâce à QT Creator.
- [images](#) : dossier contenant les images affichées

Description des modifications des fichiers .h et .cpp

Nom de l'élément	Élément	Déclaration dans le code	Fonctionnement
afficherResultat()	Procédure qui permet au joueur d'afficher les résultats des dix meilleures parties	Presentation.h	Créer un model et une vue, charge le modèle et affiche la vue.
chargerResultats()	Procédure qui permet d'insérer les tuples dans la base de données	Database.h	Exécute une requête SQL qui permet de récupérer les 10 meilleures parties, récupère le model et crée les différentes colonnes du modèle.

Nouveau tableau de test

<u>Description</u>	<u>Action réalisé</u>	<u>Résultats attendus</u>	<u>Résultats obtenus</u>
Après la connexion dans la base de donnée l'utilisateur peut regarder quelles sont les 10 meilleur partie réalisées avec cet identifiant (elle sont classés par ratio, nombre de victoire joueur / nombre de victoire machine).	Ouvertures des score qui se trouve dans de la base de données.	Affichage des parties avec les plus grand ratio, ordonné du plus grand au plus petit.	Affichage des parties avec les plus grand ratio, ordonné du plus grand au plus petit.