IUT Paris Descartes 2017-2018

Interfaces Homme—Machine

Mémoire de Projet

Maud GELLEE Groupe 101

Arsène LAPOSTOLET Groupe 112

Table des matières

I/ Présentation de l'application

Introduction :

Le Sudoku est un jeu de réflexion inventé en 1979 par l'américain Howard Garns .

Problématique :

Programmer une application graphique en MS Visual Basic .NET permettant de jouer au Sudoku.

Rôle fonctionnel :

Le but de ce projet est de programmer une application qui va gérer un plateau de 81 cases en contrôlant à chaque action du joueur que les règles du sudoku sont respectées. De même, il faut gérer différents joueurs et enregistrer des informations (nom, meilleur temps, nombre parties jouées, temps total joué) à propos de ces derniers qui pourront être consultée dans un formulaire dédié.

Fonctionnalités additionnelles :

- Grilles additionnelles : possibilité de charger une grille choisie par le joueur au début de la partie à partir d'un fichier texte.

- Interface de création : possibilité de créer des fichiers de grilles lisibles par l'application à partir d'une interface dédiée;

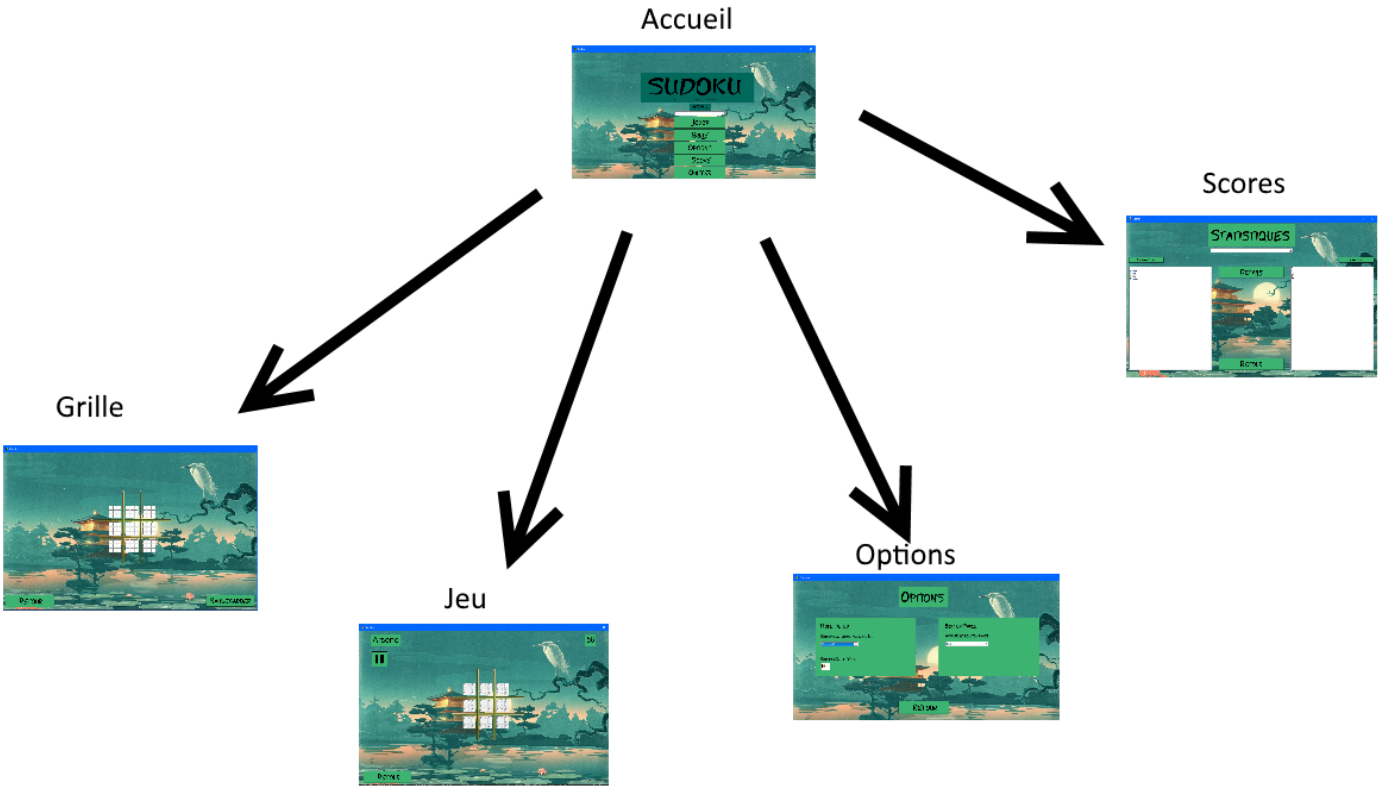
- Interface d'options : possibilité de changer les paramètres du jeu depuis une interface dédiée. Choix du mode de jeu (Temps limité ou sans fin), choix du temps pour le temps limité, choix de l'activation ou non du bouton pause.

- Bouton pause : bouton qui permet de mettre le timer en pause mais désactive les texteboxes.

- Sauvegardes Joueurs : sauvegarde automatique des informations des joueurs dans un fichier binaire : « joueurs.sav ».

- Sauvegarde options : sauvegardes automatiques des options dans un fichier texte : « options.txt ».

III/ Ordonnancement



III/ Vues de l'application

1) Le formulaire d'accueil

Bouton pour lancer l'interface de création de grille

ComboBox de Saisie des noms de joueurs

Image titre

Bouton pour lancer une partie

Bouton quitter l'application

Bouton pour lancer l'interface de consultation des score

Bouton pour lancer l'interface d'options du jeu

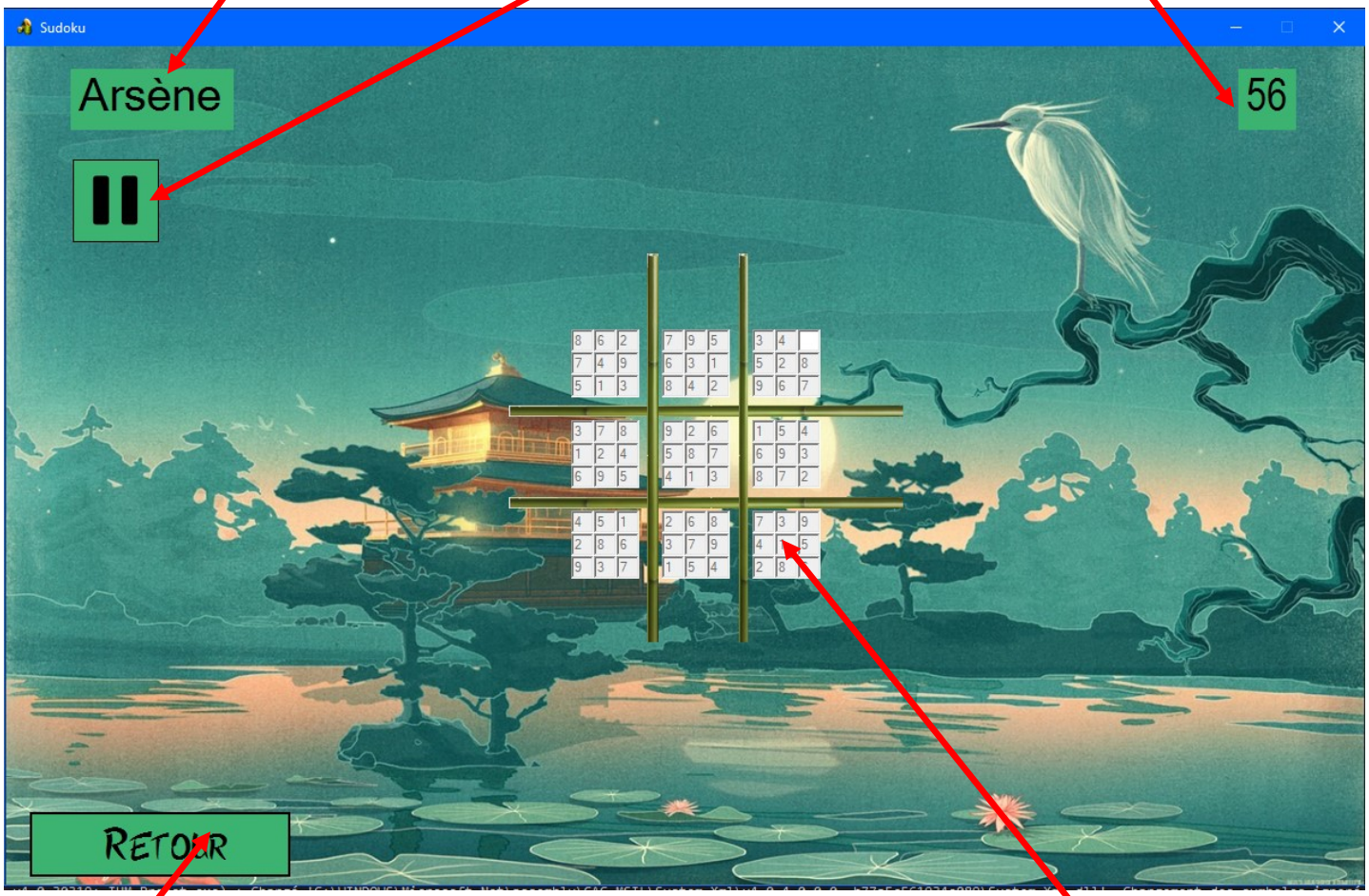
III/ Vues de l'application

2) Le formulaire de jeu

Label du nom du joueur

Bouton Pause

Label du Timer



Bouton de retour à l'accueil

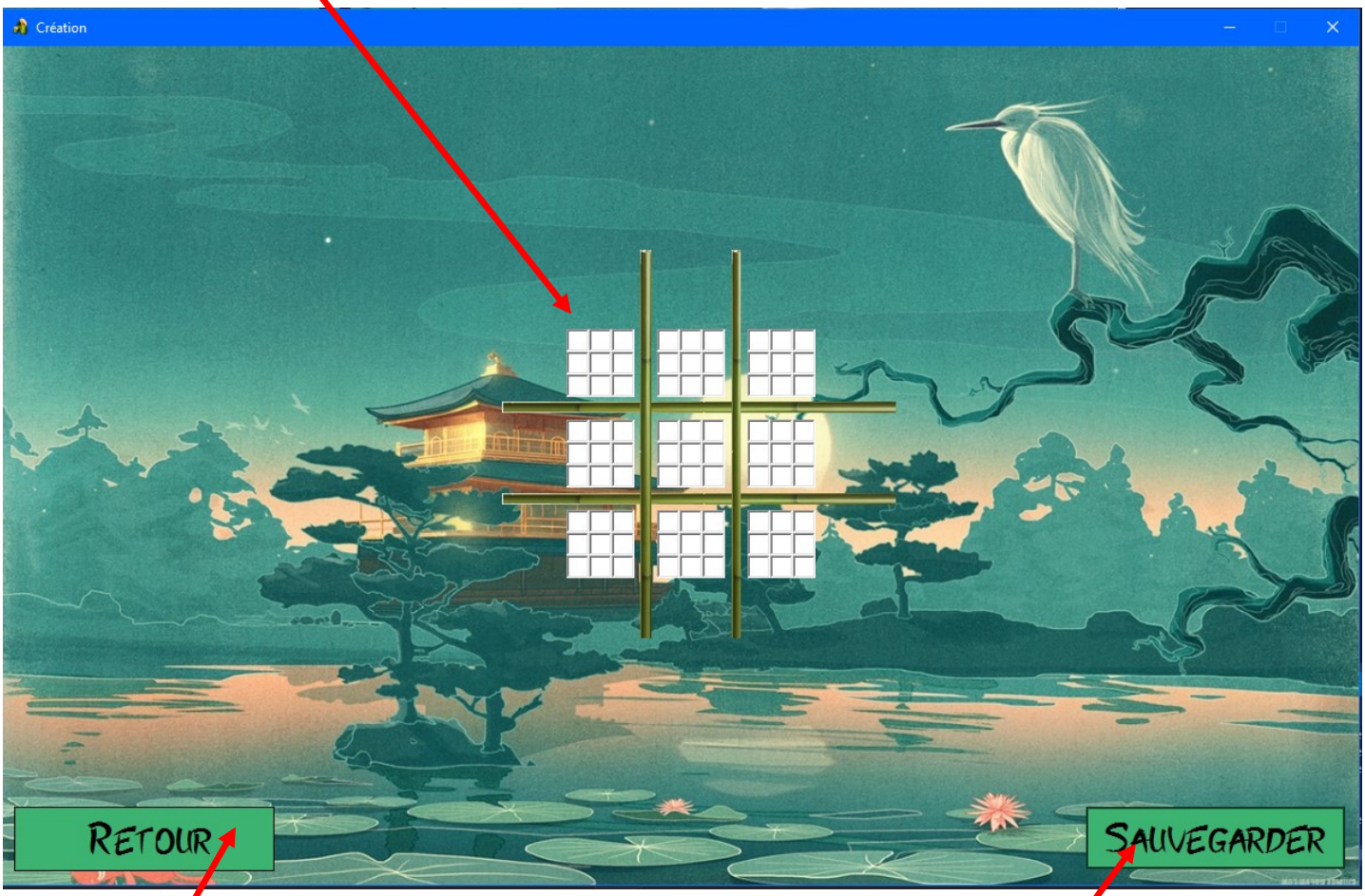
Plateau de jeu

(81 TextBoxes)

III/ Vues de l'application

3) Le formulaire de création de grille

Plateau de création



Bouton de retour à l'accueil

Bouton pour sauvegarder la grille sous forme de fichier texte lisible par l'application

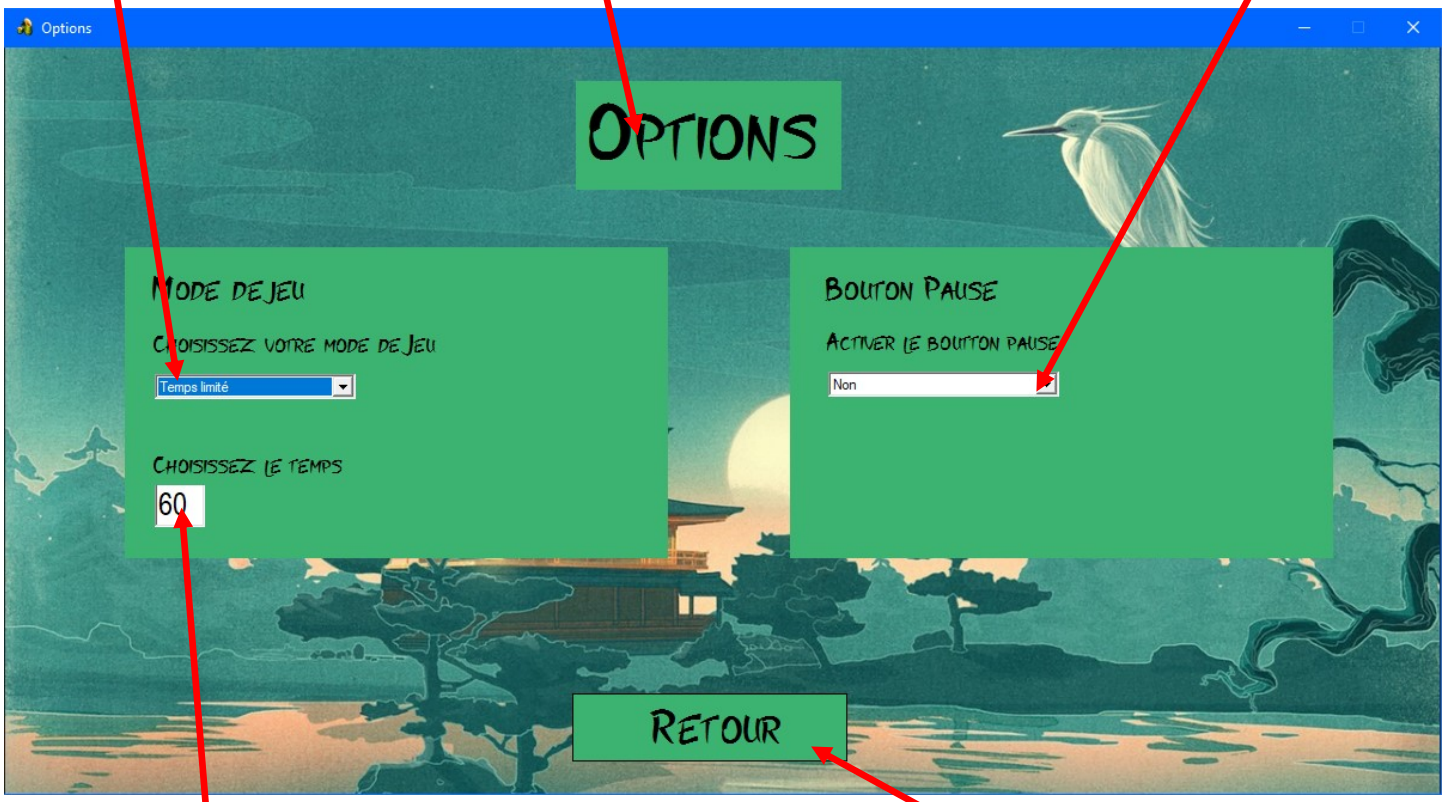
III/ Vues de l'application

4) Le formulaire des options

Combobox pour choisir Son mode de jeu

Label Titre

ComboBox pour choisir si on active le bouton Pause

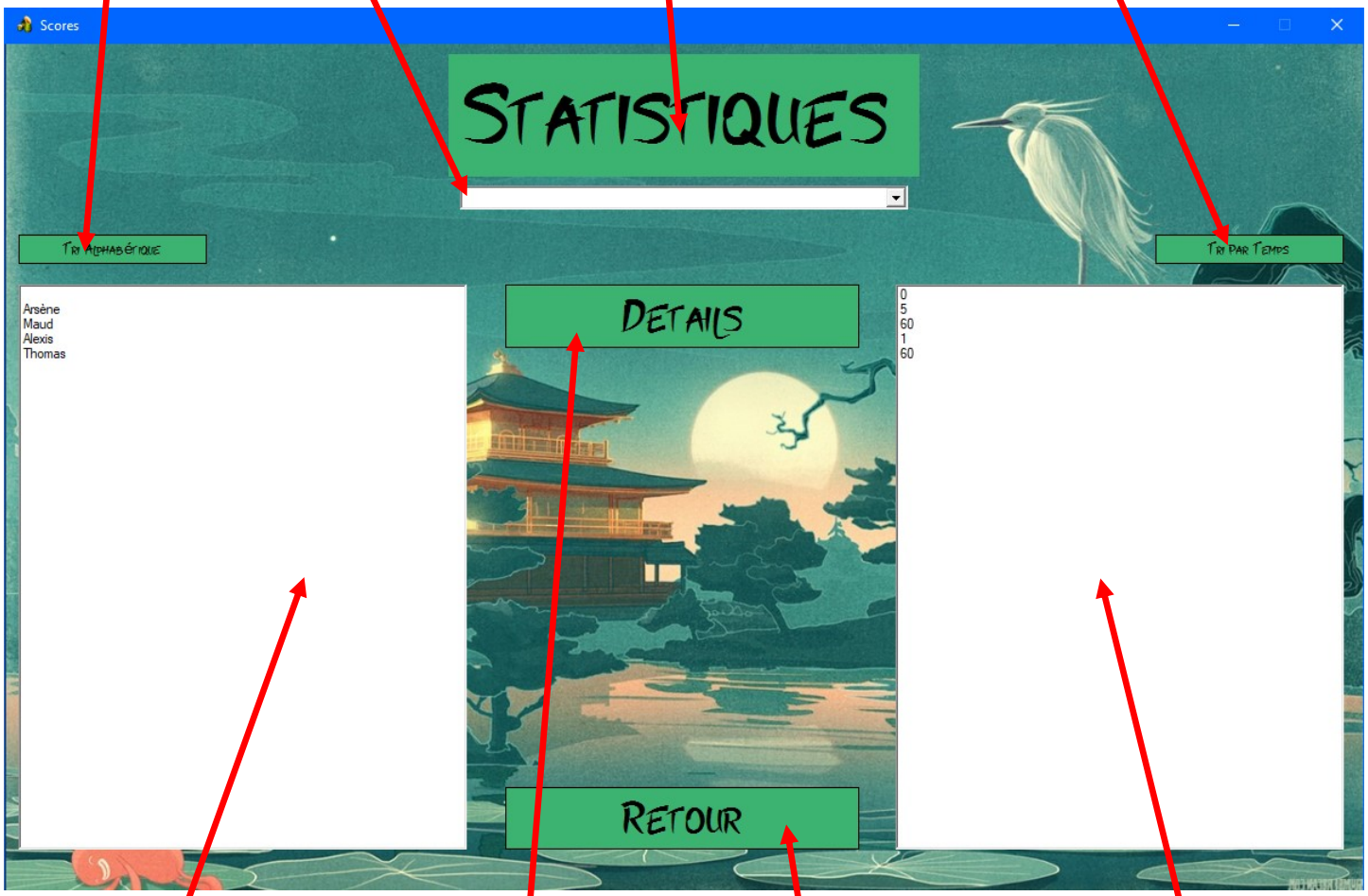


TextBox pour choisir le temps imparti

Bouton de retour à l'accueil

III/ Vues de l'application

5) Le formulaire de consultation des scores



Bouton pour trier les joueurs par ordre alphébatique

Combobox pour séléc-tionner un joueur

Label Titre

Bouton pour trier les joueurs par ordre de meilleur temps

ListBox qui liste les noms des joueurs

Bouton pour voir toutes les infos sur le joueur séléctionné

Bouton de retour à l'accueil

ListBox qui liste les meilleurs temps des joueurs

IV/ Stockage des données

Les données stockées sont les joueurs et les options. Elle sont rangées dans leurs modules respectifs ModuleJoueurs et ModuleOptions et manipulées par des Getters et des Setters. Les joueurs sont nombreux et leur nombre précis est inconnu, nous avons donc créé une structure Joueur et un tableau dynamique les regroupant. Les options quand à elles n'existent qu'en un exemplaire et sont entrées en dur dans le module. Pour ce qui est de la sauvegarde des fichiers, nous avons procédé comme suivant :

- Les joueurs sont stockés dans un fichier binaire « joueurs.sav ». On y écrit directement la Structure.

- Les options sont stockées dans un fichier texte « Options.txt ». On y écrit simplement ligne après ligne le contenu des options à la fermeture de l'application, après avoir préalablement supprimé l'ancien fichier. De même à l'ouverture, on lit le contenu du fichier ligne par ligne et on le stock dans les options.

De plus, l'interface de création de grille permet de créer des grilles. Elles sont sauvegardées dans un fichier texte dont le nom et l'emplacement est choisi par l'utilisateur au moyen d'un Prompt. Chaque numéro entré dans chaque case de la grille est écrit sur une ligne. Si le joueur n'a rien écrit, on écrit 0. De même lorsque le joueur choisit une grille personnalisé au moyen du Prompt de début de partie, le fichier texte est lu ligne par ligne et le chiffre est mis dans la TextBox correspondante, qui est ensuite verrouillée, si on lit, 0 on laisse la case vite et on ne la verrouille pas.

V/ Bilan du projet

Pendant ce projet, nous avons été initiés à la conception et à la programmation en paradigme événementiel. Cette approche de programmation étant nouvelle pour nous, cela nous a demandé un court temps d'adaptation. Cependant, nous nous sommes vite approprié cette nouvelle manière d'organiser le code. Nous en avons appréciée l'approche très structurée et parfois plus concrète, ainsi que la possibilité d'obtenir facilement des sorties graphiques.

De même, nous avons découvert MS Visual Basic .NET, un nouveau langage. Son caractère semi-compilé et la richesse de la bibliothèque standard ainsi que le Framework .NET le rende plus rapide à utiliser. Cependant, certaines spécificités nous ont posé quelques problèmes. On peut citer notamment le fait qu'en VB.NET, la caractère référence ou valeur d'un objet est moins explicit qu'en C ou Java, ce qui a été, dans certaines situations complexes une source de confusion pour nous.

D'un point de vue plus général, ce projet a été une très bonne expérience qui nous a permis de découvrir VB.NET et la programmation événementielle dans de bonne conditions. Le sujet, à la fois ludique et intéressant nous a permis de nous faire ressentir l'esprit du développeur. En effet, le fait de créer un tel logiciel fourni un sentiment d'accomplissement très agréable.

Enfin, l'aspect collaboratif de ce projet fait partie de ses points forts. En effet, la motivation mutuelle est un moteur puissant qui permet de surmonter tout type de difficulté. De plus, la résonnance de deux esprits focalisés sur le même objectif, communicant sur un problème est une des formes les plus efficaces de réflexion. Chacun comblant les faiblesses de l'autre, nous avons pu nous tirer mutuellement vers le haut. Ayant déjà l'expérience du travail en commun, nous avons renforcé notre niveau de collaboration et cela nous a permis d'avancer encore plus , chacun connaissant les spécificités de l'autre.

Code des Formulaires

1) Formulaire d'accueil

```
Public Class Accueil

    'Initialisation du formulaire
    Private Sub Accueil_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        ModuleInterface.Ambiance()
        ModuleInterface.ForbidResize(Me)
        For i As Integer = 0 To ModuleJoueur.getNbJoueurs - 1
            ComboBoxNomJoueur.Items.Add(Trim(ModuleJoueur.getNomJoueur(i)))
        Next
    End Sub
    'Lance une partie grace au formulaire jeu
    Private Sub BtnJouer_Click(sender As Object, e As EventArgs) Handles BtnJouer.Click
        If ComboBoxNomJoueur.Text = "" Then
            MsgBox("Veuillez renseigner un nom", MsgBoxStyle.OkOnly)
        Else
            If ModuleJoueur.isNouveau(ComboBoxNomJoueur.Text) Then
                ajoutJoueur(ComboBoxNomJoueur.Text)
                Me.Hide()
                Jeu.Show()
            Else
                Me.Hide()
                Jeu.Show()
            End If
        End If
    End Sub
    'Quitte l'application
    Private Sub BtnQuitter_Click(sender As Object, e As EventArgs) Handles BtnQuitter.Click
        Application.Exit()
    End Sub
    'Ouvre la fenêtre des statistiques
    Private Sub BtnScore_Click(sender As Object, e As EventArgs) Handles BtnScore.Click
        Statistique.Show()
        Me.Hide()
    End Sub
```

Code des Formulaires

```
    'Ouvre la fenêtre de création
    Private Sub BtnCreation_Click(sender As Object, e As EventArgs) Handles BtnCreation.Click
        Me.Hide()
        Creation.Show()
    End Sub

    'Effet de swapcolor au survole
    Private Sub Btn_In(sender As Button, e As EventArgs) Handles BtnCreation.MouseEnter, BtnJouer.MouseEnter, BtnQuit-
ter.MouseEnter, BtnScore.MouseEnter
        ModuleInterface.swapColorIn(sender)
    End Sub

    'Effet de swapcolor au survole
    Private Sub Btn_Out(sender As Button, e As EventArgs) Handles BtnCreation.MouseLeave, BtnJouer.MouseLeave, BtnQuit-
ter.MouseLeave, BtnScore.MouseLeave
        ModuleInterface.swapColorOut(sender)
    End Sub

    Private Sub BtnOptions_Click(sender As Object, e As EventArgs) Handles BtnOptions.Click
        Me.Hide()
        Options.Show()
    End Sub

    'Lancer du jeu si on tape enter dans la combobox de nom de joueur
    Private Sub ComboBoxNomJoueur_SelectedIndexChanged(sender As Object, e As KeyPressEventArgs) Handles Combo-
BoxNomJoueur.KeyPress
        If e.KeyChar = Microsoft.VisualBasic.ChrW(Keys.Return) Then
            BtnJouer.PerformClick()
            e.Handled = True
        End If
    End Sub
End Class
```

Code des Formulaires

2) Formulaire de jeu

```
Imports System.IO

Public Class Jeu

    'Constantes de jeu
    Dim temps As Integer = ModuleOptions.getTemps()
    Const NB_CASES_GRID As Integer = 81
    Const NB_CASES_LC_GRID As Integer = 9
    Dim jouable As Boolean = True
    'Tableaux de stockage
    Dim boxes(NB_CASES_GRID - 1) As TextBox
    Dim carres(NB_CASES_LC_GRID - 1) As Carre
    Dim lignes(NB_CASES_LC_GRID - 1) As Ligne
    Dim colonnes(NB_CASES_LC_GRID - 1) As Colonne
    'Structures de stockage
    Public Structure Carre
        Dim boxes() As TextBox
    End Structure
    Public Structure Ligne
        Dim boxes() As TextBox
    End Structure
    Public Structure Colonne
        Dim boxes() As TextBox
    End Structure
    'Initialisation des structures de stockage du jeu
    Public Sub initBoxes()
        Dim x As Integer = 500
        Dim y As Integer = 250
        Dim k As Integer = 0
        'Loop de création des textBoxes
        For i As Integer = 1 To NB_CASES_LC_GRID
            For j As Integer = 1 To NB_CASES_LC_GRID
```

```
            Dim t As New TextBox
            t.Size = New Size(20, 20)
            t.Location = New Point(x, y)
            If Not (j Mod 3 = 0) Then
                x += 20
            Else
                x += 40
            End If
            boxes(k) = t
            AddHandler t.TextChanged, AddressOf TxtBx_Grid_TxtChanged
            AddHandler t.KeyPress, AddressOf TxtBx_GridKeypress
            Me.Controls.Add(t)

            k += 1
        Next
        If Not (i Mod 3 = 0) Then
            y += 20
        Else
            y += 40
        End If

        x = 500
    Next
    'Initialisation des carrés
    For i As Integer = 0 To NB_CASES_LC_GRID - 1
        Dim c As Carre
        ReDim c.boxes(NB_CASES_LC_GRID - 1)
        carres(i) = c
    Next
```

Code des Formulaires

```
'Initialisation des lignes
        For i As Integer = 0 To NB_CASES_LC_GRID - 1
            Dim l As Ligne
            ReDim l.boxes(NB_CASES_LC_GRID - 1)
            lignes(i) = l
        Next
        'Initialisation des des colonnes
        For i As Integer = 0 To NB_CASES_LC_GRID - 1
            Dim col As Colonne
            ReDim col.boxes(NB_CASES_LC_GRID - 1)
            colonnes(i) = col
        Next
        'Remplissage des carres
        carres(0).boxes(0) = boxes(0)
        carres(0).boxes(1) = boxes(1)
        carres(0).boxes(2) = boxes(2)
        carres(0).boxes(3) = boxes(9)
        carres(0).boxes(4) = boxes(10)
        carres(0).boxes(5) = boxes(11)
        carres(0).boxes(6) = boxes(18)
        carres(0).boxes(7) = boxes(19)
        carres(0).boxes(8) = boxes(20)
        carres(1).boxes(0) = boxes(3)
        carres(1).boxes(1) = boxes(4)
        carres(1).boxes(2) = boxes(5)
        carres(1).boxes(3) = boxes(12)
        carres(1).boxes(4) = boxes(13)
        carres(1).boxes(5) = boxes(14)
        carres(1).boxes(6) = boxes(21)
        carres(1).boxes(7) = boxes(22)
        carres(1).boxes(8) = boxes(23)
```

```
carres(2).boxes(0) = boxes(6)
carres(2).boxes(1) = boxes(7)
carres(2).boxes(2) = boxes(8)
carres(2).boxes(3) = boxes(15)
carres(2).boxes(4) = boxes(16)
carres(2).boxes(5) = boxes(17)
carres(2).boxes(6) = boxes(24)
carres(2).boxes(7) = boxes(25)
carres(2).boxes(8) = boxes(26)
carres(3).boxes(0) = boxes(27)
carres(3).boxes(1) = boxes(28)
carres(3).boxes(2) = boxes(29)
carres(3).boxes(3) = boxes(36)
carres(3).boxes(4) = boxes(37)
carres(3).boxes(5) = boxes(38)
carres(3).boxes(6) = boxes(45)
carres(3).boxes(7) = boxes(46)
carres(3).boxes(8) = boxes(47)
carres(4).boxes(0) = boxes(30)
carres(4).boxes(1) = boxes(31)
carres(4).boxes(2) = boxes(32)
carres(4).boxes(3) = boxes(39)
carres(4).boxes(4) = boxes(40)
carres(4).boxes(5) = boxes(41)
carres(4).boxes(6) = boxes(48)
carres(4).boxes(7) = boxes(49)
carres(4).boxes(8) = boxes(50)
carres(5).boxes(0) = boxes(33)
carres(5).boxes(1) = boxes(34)
carres(5).boxes(2) = boxes(35)
carres(5).boxes(3) = boxes(42)
```

```
carres(5).boxes(4) = boxes(43)
carres(5).boxes(5) = boxes(44)
carres(5).boxes(6) = boxes(51)
carres(5).boxes(7) = boxes(52)
carres(5).boxes(8) = boxes(53)
carres(6).boxes(0) = boxes(54)
carres(6).boxes(1) = boxes(55)
carres(6).boxes(2) = boxes(56)
carres(6).boxes(3) = boxes(63)
carres(6).boxes(4) = boxes(64)
carres(6).boxes(5) = boxes(65)
carres(6).boxes(6) = boxes(72)
carres(6).boxes(7) = boxes(73)
cares(6).boxes(8) = boxes(74)
carres(7).boxes(0) = boxes(57)
carres(7).boxes(1) = boxes(58)
carres(7).boxes(2) = boxes(59)
carres(7).boxes(3) = boxes(66)
carres(7).boxes(4) = boxes(67)
carres(7).boxes(5) = boxes(68)
carres(7).boxes(6) = boxes(75)
carres(7).boxes(7) = boxes(76)
carres(7).boxes(8) = boxes(77)
carres(8).boxes(0) = boxes(60)
carres(8).boxes(1) = boxes(61)
carres(8).boxes(2) = boxes(62)
carres(8).boxes(3) = boxes(69)
carres(8).boxes(4) = boxes(70)
carres(8).boxes(5) = boxes(71)
carres(8).boxes(6) = boxes(78)
carres(8).boxes(7) = boxes(79)
carres(8).boxes(8) = boxes(80)
```

Code des Formulaires

```
'Remplissage des lignes
        For i As Integer = 0 To NB_CASES_LC_GRID - 1
            For j As Integer = 0 To NB_CASES_LC_GRID - 1
                lignes(i).boxes(j) = boxes(j + (9 * i))
            Next
        Next

        'Remplissage des colonnes
        For i As Integer = 0 To NB_CASES_LC_GRID - 1
            For j As Integer = 0 To NB_CASES_LC_GRID - 1
                colonnes(i).boxes(j) = boxes(i + 9 * j)
            Next
        Next
    End Sub
    'check de la validité des modifs
    Private Function Checkboxes(box As TextBox) As Boolean

        Dim currentCarre As Integer
        Dim currentLigne As Integer
        Dim currentCol As Integer
        'Identifie le carré dans lequel le sender se trouve
        For i As Integer = 0 To NB_CASES_LC_GRID - 1
            For j As Integer = 0 To NB_CASES_LC_GRID - 1
                If (carres(i).boxes(j).Equals(box)) Then
                    currentCarre = i
                End If
            Next
        Next
```

Code des Formulaires

```
'Identifie la ligne dans laquel on se trouve
        For i As Integer = 0 To NB_CASES_LC_GRID - 1
            For j As Integer = 0 To NB_CASES_LC_GRID - 1
                If (lignes(i).boxes(j).Equals(box)) Then
                    currentLigne = i
                End If
            Next
        Next

        'Identifie la colonne dans la laquelle on se trouve
        For i As Integer = 0 To NB_CASES_LC_GRID - 1
            For j As Integer = 0 To NB_CASES_LC_GRID - 1
                If (colonnes(i).boxes(j).Equals(box)) Then
                    currentCol = i
                End If
            Next
        Next
        'On vérérifie si une autre textBox que le sender a la même valeur dans le carré
        For Each t As TextBox In carres(currentCarre).boxes
            If (t.Text = box.Text) And Not t.Equals(box) Then
                Return False
            End If
        Next
        'On vérérifie si une autre textBox que le sender a la même valeur dans la ligne
        For Each t As TextBox In lignes(currentLigne).boxes
            If (t.Text = box.Text) And Not t.Equals(box) Then
                Return False
            End If
        Next
```

Code des Formulaires

```
        'On vérifie si une autre textBox que le sender a la même valeur dans la colonne
        For Each t As TextBox In colonnes(currentCol).boxes
            If (t.Text = box.Text) And Not t.Equals(box) Then
                Return False
            End If
        Next

        'On vérifie si la configuration actuelle est gagnante
        Dim victoire As Boolean = True
        For Each t As TextBox In boxes
            If t.Text = "" OrElse t.BackColor = Color.Red Then
                victoire = False
            End If
        Next
        If victoire Then
            EndVictoire()
        End If

        'Si on est pas sortie jusque là c'est que la valeur est bonne
        Return True

    End Function

'Handler KeyPress textboxes
    Private Sub TxtBx_GridKeypress(sender As TextBox, e As KeyPressEventArgs)
        If ((Not IsNumeric(e.KeyChar)) OrElse (Not sender.Text.Length = 0)) AndAlso Not e.KeyChar =
ControlChars.Back Then
            e.Handled = True
        End If
    End Sub
```

Code des Formulaires

```
Private Sub TxtBx_Grid_TxtChanged(sender As TextBox, e As EventArgs)
        'modification
        Dim isValide As Boolean = Checkboxes(sender)
        If (isValide) Then
            sender.BackColor = SystemColors.Control
        Else
            sender.BackColor = Color.Red
        End If
    End Sub

    'Charge une grille depuis un fichier
    Sub readGridContent()
        Dim i As Integer
        Dim fileName As String
        If FileDialog.ShowDialog() = DialogResult.OK Then
            fileName = FileDialog.FileName
        Else
            fileName = "grid.txt"
        End If
        Dim f As StreamReader = New StreamReader(fileName)
        For Each t As TextBox In boxes
            i = f.ReadLine
            If Not i = 0 Then
                t.Text = i
                t.Enabled = False
            End If
        Next
    End Sub
```

```
    'Fin à l'écoulement du temps
    Private Sub Endgame()
        Dim currentJoueur As Integer
        For i As Integer = 0 To ModuleJoueur.getNbJoueurs() - 1
            If Accueil.ComboBoxNomJoueur.Text = Trim(ModuleJoueur.getNomJoueur(i)) Then
                currentJoueur = i
            End If
        Next
        ModuleJoueur.majStatJoueur(currentJoueur, 60, 60)
        Me.Close()
        Accueil.Show()
    End Sub

    'Fin en cas de victoire
    Private Sub EndVictoire()
        Dim currentJoueur As Integer
        For i As Integer = 0 To ModuleJoueur.getNbJoueurs() - 1
            If Accueil.ComboBoxNomJoueur.Text = Trim(ModuleJoueur.getNomJoueur(i)) Then
                currentJoueur = i
            End If
        Next
        If ModuleOptions.getMode = "Temps limité" Then
            ModuleJoueur.majStatJoueur(currentJoueur, ModuleOptions.getTemps() - LabelTimer.Text, Mo-
duleOptions.getTemps() - LabelTimer.Text)
        Else
            ModuleJoueur.majStatJoueur(currentJoueur, 10000, LabelTimer.Text)
        End If

        MsgBox("Victoire !", MsgBoxStyle.OkOnly)
        Me.Close()
        Accueil.Show()
    End Sub
```

Code des Formulaires

```
    'Fin quand on click sur retour
    Private Sub EndRetour()
        Dim currentJoueur As Integer

        For i As Integer = 0 To ModuleJoueur.getNbJoueurs() - 1
            If Accueil.ComboBoxNomJoueur.Text = Trim(ModuleJoueur.getNomJoueur(i)) Then
                currentJoueur = i
            End If
        Next
        If ModuleOptions.getMode = "Temps limité" Then
            ModuleJoueur.majStatJoueur(currentJoueur, ModuleOptions.getTemps() - LabelTimer.Text, Mo-
duleOptions.getTemps() - LabelTimer.Text)
        Else
            ModuleJoueur.majStatJoueur(currentJoueur, 60, LabelTimer.Text)
        End If
    End Sub

    'Bouton retour menu principal
    Private Sub BtnRetour_Click(sender As Object, e As EventArgs) Handles BtnRetour.Click
        EndRetour()
        Me.Close()
        Accueil.Show()
    End Sub
    'Tick du Timer
    Private Sub Timer_Tick(sender As Object, e As EventArgs) Handles Timer.Tick
        If Not ModuleOptions.getMode = "Sans fin" Then
            temps -= 1
        Else
            temps += 1
        End If
```

```
        If temps = 0 Then
            Timer.Stop()
            LabelTimer.Text = 0
            MsgBox("Perdu !", MsgBoxStyle.OkOnly)
            Endgame()
            Me.Hide()
            Accueil.Show()
        Else
            LabelTimer.Text = temps
        End If
    End Sub
    'Load du jeu qui appelle toutes les sub pour chaque étape de la préparation
    Private Sub Jeu_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        ModuleInterface.ForbidResize(Me)
        NomJoueur.Text = Accueil.ComboBoxNomJoueur.Text

        temps = ModuleOptions.getTemps()
        LabelTimer.Text = temps

        initBoxes()
        readGridContent()
        Timer.Start()
        If ModuleOptions.getMode = "Sans fin" Then
            LabelTimer.Visible = False
            LabelTimer.Text = 0
        End If
        BtnPause.Visible = (ModuleOptions.getBtnPauseActive() = "Oui")
    End Sub
```

```
    Private Sub BtnPause_Click(sender As Object, e As EventArgs) Handles BtnPause.Click
        If Timer.Enabled Then
            Timer.Stop()
            ModuleInterface.swapBtnCostumePlay(sender)
        Else
            Timer.Start()
            ModuleInterface.swapBtnCostumePause(sender)
        End If
        If Timer.Enabled Then
            For Each t As TextBox In boxes
                t.Enabled = True
            Next
        Else
            For Each t As TextBox In boxes
                t.Enabled = False
            Next
        End If
    End Sub
End Class
```

Code des Formulaires

3) Formulaire de Création de grille

```
Imports System.IO
Public Class Creation
    Const NB_CASES_GRID As Integer = 81
    Const NB_CASES_LC_GRID As Integer = 9
    Dim boxes(NB_CASES_GRID - 1) As TextBox

    'Génération des textbox
    Private Sub Creation_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        ModuleInterface.ForbidResize(Me)
        Dim x As Integer = 500
        Dim y As Integer = 250
        Dim k As Integer = 0
        For i As Integer = 1 To NB_CASES_LC_GRID
            For j As Integer = 1 To NB_CASES_LC_GRID
                Dim t As New TextBox
                t.Size = New Size(20, 20)
                t.Location = New Point(x, y)
                If Not (j Mod 3 = 0) Then
                    x += 20
                Else
                    x += 40
                End If
                boxes(k) = t
                AddHandler t.KeyPress, AddressOf TxtBx_GridKeypress
                Me.Controls.Add(t)

                k += 1
            Next
```

```
            If Not (i Mod 3 = 0) Then
                y += 20
            Else
                y += 40
            End If

            x = 500
        Next
    End Sub
 'Empêche d'écrire autre chose qu'un chiffre
    Private Sub TxtBx_GridKeypress(sender As TextBox, e As KeyPressEventArgs)
        If ((Not IsNumeric(e.KeyChar)) Or (Not sender.Text.Length = 0)) And Not e.KeyChar = Control-
Chars.Back Then
            e.Handled = True
        End If
    End Sub

    'Sauvegarde la grille dans un fichier texte, lisible et jouable dans l'application
    Private Sub BtnSave_Click(sender As Object, e As EventArgs) Handles BtnSave.Click
        Dim s As String = ""
        For Each t As TextBox In boxes
            If t.Text <> "" Then
                s += t.Text
                s += vbCrLf
            Else
                s += "0"
                s += vbCrLf
            End If
        Next
        Dim output As StreamWriter
```

```
        If SaveFileDialog.ShowDialog() = DialogResult.OK Then
            output = My.Computer.FileSystem.OpenTextFileWriter(SaveFileDialog.FileName, True)
            output.WriteLine(s)
            output.Close()
        End If
    End Sub
    'Retour à l'accueil
    Private Sub BtnRetour_Click(sender As Object, e As EventArgs) Handles BtnRetour.Click
        Me.Close()
        Accueil.Show()
    End Sub
End Class
```

Code des Formulaires

4) Formulaire d'options

```
Imports System.IO
Public Class Options
    Private Sub Options_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        ModuleInterface.ForbidResize(Me)
        ComboBoxMode.Items.Add("Temps limité")
        ComboBoxMode.Items.Add("Sans fin")
        ComboBoxBtnPause.Items.Add("Non")
        ComboBoxBtnPause.Items.Add("Oui")
        ComboBoxBtnPause.SelectedItem = ModuleOptions.getBtnPauseActive()
        ComboBoxMode.SelectedItem = ModuleOptions.getMode()
        TextBoxTemps.Text = ModuleOptions.getTemps()
    End Sub

    Private Sub BtnRetour_Click(sender As Object, e As EventArgs) Handles BtnRetour.Click

        ModuleOptions.setTemps(TextBoxTemps.Text)
        Me.Hide()
        Accueil.Show()
    End Sub

    Private Sub TextBoxTemps_Keypress(sender As Object, e As KeyPressEventArgs) Handles
TextBoxTemps.KeyPress
        If ((Not IsNumeric(e.KeyChar)) AndAlso Not e.KeyChar = ControlChars.Back) Then
            e.Handled = True
        End If
    End Sub
```

```
    Private Sub ComboBoxMode_SelectedIndexChanged(sender As Object, e As EventArgs) Handles Combo-
BoxMode.SelectedIndexChanged
        If sender.Text <> "Sans fin" Then
            TextBoxTemps.Visible = True
            LabelTime.Visible = True
        Else
            TextBoxTemps.Visible = False
            LabelTime.Visible = False
        End If
        ModuleOptions.setMode(sender.Text)
    End Sub

    Private Sub ComboBoxBtnPause_SelectedIndexChanged(sender As Object, e As EventArgs) Handles Combo-
BoxBtnPause.SelectedIndexChanged
        ModuleOptions.setBtnPauseActive(sender.Text)
    End Sub
End Class
```

Code des Formulaires

5) Formulaire de consultation des scores

```
Public Class Statistique


    'Load du formulaire
    'Chargement des données des joueurs depuis le tableau des joueurs
    Private Sub Statistique_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        ModuleInterface.ForbidResize(Me)
        For i As Integer = 0 To ModuleJoueur.getNbJoueurs() - 1
            ComboBoxNomJoueur.Items.Add(Trim(ModuleJoueur.getNomJoueur(i)))
            ListBoxJoueurs.Items.Add(Trim(ModuleJoueur.getNomJoueur(i)))
            ListBoxTemps.Items.Add(Trim(ModuleJoueur.getMeilleurTpsJoueur(i)))
        Next
    End Sub

    'Click du bouton détail
    'Msg qui donne toutes les infos d'un joueur
    Private Sub BtnInfo_Click(sender As Object, e As EventArgs) Handles BtnInfo.Click
        If ComboBoxNomJoueur.Text <> "" Then
            Dim currentJoueur As Integer

            For i As Integer = 0 To ModuleJoueur.getNbJoueurs() - 1

                If ComboBoxNomJoueur.Text = Trim(ModuleJoueur.getNomJoueur(i)) Then

                    currentJoueur = i
                End If
            Next
```

Code des Formulaires

```
            MsgBox("Nom : " & ModuleJoueur.getNomJoueur(currentJoueur) & vbCrLf &
                   "Meilleur Temps : " & ModuleJoueur.getMeilleurTpsJoueur(currentJoueur) & vbCrLf &
                   "Nombre de parties jouées : " & ModuleJoueur.getNbPartiesJoueur(currentJoueur) &
vbCrLf &
                   "Temps de Jeu : " & ModuleJoueur.getTpsJoueJoueur(currentJoueur) \ 60 & ":" & Modu-
leJoueur.getTpsJoueJoueur(currentJoueur) Mod 60 & vbCrLf, MsgBoxStyle.OkOnly)
        Else
            MsgBox("Veuillez séléctionner un joueur", MsgBoxStyle.OkOnly)
        End If
    End Sub

    'Synchronisation entre la combobox des nom et la séléction dans la listeBox
    Private Sub ListBoxJoueurs_SelectedIndexChanged(sender As ListBox, e As EventArgs) Handles
ListBoxJoueurs.SelectedIndexChanged
        ComboBoxNomJoueur.Text = sender.SelectedItem.ToString
    End Sub
    'Bouton retour Click
    'Ferme se formulaire et ouvre l'accueil
    Private Sub BtnRetour_Click(sender As Object, e As EventArgs) Handles BtnRetour.Click
        Me.Close()
        Accueil.Show()
    End Sub

    'Synchronisation entre la combobox des nom et la séléction dans la listeBox
    Private Sub ComboBoxNomJoueur_SelectedIndexChanged(sender As ComboBox, e As EventArgs) Handles
ComboBoxNomJoueur.SelectedIndexChanged
        ListBoxJoueurs.SelectedItem = sender.Text

    End Sub
```

Code des Formulaires

```
'Tri de manière alphabétique les nom des joueurs dans la listBox des joueurs
'Synchronisation avec la listebox des temps
Private Sub BtnAlpha_Click(sender As Object, e As EventArgs) Handles BtnAlpha.Click
    Dim fait As Boolean = True

    Do
        fait = True
        For j As Integer = 0 To ListBoxJoueurs.Items.Count - 2
            If ListBoxJoueurs.Items(j + 1) < ListBoxJoueurs.Items(j) Then
                fait = False
                Dim temp1 As String = ListBoxJoueurs.Items(j + 1)
                ListBoxJoueurs.Items(j + 1) = ListBoxJoueurs.Items(j)
                ListBoxJoueurs.Items(j) = temp1

                Dim temp2 As String = ListBoxTemps.Items(j + 1)
                ListBoxTemps.Items(j + 1) = ListBoxTemps.Items(j)
                ListBoxTemps.Items(j) = temp2
            End If

        Next
    Loop Until fait
End Sub
```

```
    'Tri de manière croissante les temps dans la listBox des temps
    'Synchronisation avec la listebox des joueurs
    Private Sub BtnTps_Click(sender As Object, e As EventArgs) Handles BtnTps.Click

        Dim fait As Boolean = True

        Do
            fait = True
            For j As Integer = 0 To ListBoxJoueurs.Items.Count - 2
                If ListBoxTemps.Items(j + 1) < ListBoxTemps.Items(j) Then
                    fait = False
                    Dim temp1 As String = ListBoxJoueurs.Items(j + 1)
                    ListBoxJoueurs.Items(j + 1) = ListBoxJoueurs.Items(j)
                    ListBoxJoueurs.Items(j) = temp1

                    Dim temp2 As String = ListBoxTemps.Items(j + 1)
                    ListBoxTemps.Items(j + 1) = ListBoxTemps.Items(j)
                    ListBoxTemps.Items(j) = temp2
                End If

            Next
        Loop Until fait
    End Sub
End Class
```

Code des Modules

1) Module Joueurs

```
Imports System.IO
Module ModuleJoueur

    'Tableau qui contient les joueurs
    Dim joueurs(0) As Joueur

    'Structure joueur
    Public Structure Joueur
        <VBFixedString(20)> Dim Nom As String
        Dim MeilleurTemps As Integer
        Dim nbPartiesJouees As Integer
        Dim tpsJoue As Integer
    End Structure

    'Mise à joueur des statistiques d'un joueur en fin de partie
    Public Sub majStatJoueur(noJoueur As Integer, temps As Integer, tpsJoue As Integer)
        joueurs(noJoueur).nbPartiesJouees += 1
        If joueurs(noJoueur).MeilleurTemps > temps Then
            joueurs(noJoueur).MeilleurTemps = temps
        End If
        joueurs(noJoueur).tpsJoue += tpsJoue
    End Sub

    'Getter du nombre de joueurs
    Function getNbJoueurs()
        Return joueurs.Length
    End Function
```

Code des Modules

```
'Getter d'un joueur
Public Function getJoueur(i As Integer)
    Return joueurs(i)
End Function

'Permet de savoir si un joueur est déjà enregistré par son nom
Public Function isNouveau(Nom As String)
    For Each j As Joueur In joueurs
        If Trim(Nom) = Trim(j.Nom) Then
            Return False
        End If
    Next
    Return True
End Function

'Getter pour le nom d'un joueur
Public Function getNomJoueur(i As Integer) As String
    Return joueurs(i).Nom
End Function

'Getter pour le meilleur temps d'un joueur
Public Function getMeilleurTpsJoueur(i As Integer) As Integer
    Return joueurs(i).MeilleurTemps
End Function
'Getter pour le temps de jeu d'un joueur
Public Function getTpsJoueJoueur(i As Integer) As Integer
    Return joueurs(i).tpsJoue
End Function
```

Code des Modules

```
'Getter pour le nombre de parties jouées par un joueur
Public Function getNbPartiesJoueur(i As Integer) As Integer
    Return joueurs(i).nbPartiesJouees
End Function

'ajout d'un joueur
Public Sub ajoutJoueur(Nom As String)

    If isNouveau(Nom) Then
        Accueil.ComboBoxNomJoueur.Items.Add(Nom)
        Dim j As Joueur
        j.Nom = Nom
        j.MeilleurTemps = 61
        j.nbPartiesJouees = 0
        j.tpsJoue = 0
        ReDim Preserve joueurs(getNbJoueurs)
        joueurs(getNbJoueurs() - 1) = j
    End If

End Sub

'Programme principal
Sub Main()
    'Lecture des joueurs dans la sauvegarde
    ReDim joueurs(0)
    Dim num As Integer = FreeFile()
    FileOpen(num, "joueurs.sav", OpenMode.Random, , , Len(New Joueur))
    Dim i As Integer = 0
```

Code des Modules

```
        While Not EOF(num)
            ReDim Preserve joueurs(i)
            FileGet(num, joueurs(i), i + 1)
            i += 1
        End While
'Lecture des options dans le fichier-
        Try
            Dim input As StreamReader = New StreamReader("Options.txt")
            ModuleOptions.setTemps(input.ReadLine)
            ModuleOptions.setMode(input.ReadLine)
            ModuleOptions.setBtnPauseActive(input.ReadLine)
            input.Close()
        Catch

        End Try


        'Lancement de l'application
        Application.Run(Accueil)

        'Ecriture des options dans le fichier
        File.Delete("Options.txt")
        Dim output As StreamWriter
        output = My.Computer.FileSystem.OpenTextFileWriter("Options.txt", True)
        output.WriteLine(ModuleOptions.getTemps())
        output.WriteLine(ModuleOptions.getMode())
        output.WriteLine(ModuleOptions.getBtnPauseActive())
        output.Close()
```

Code des Modules

```
        'Ecriture des joueurs dans la sauvegarde
        For j As Integer = 0 To getNbJoueurs() - 1
            FilePut(num, joueurs(j), j + 1)
        Next
        FileClose()
    End Sub
End Module
```

Code des Modules

2) Module Options

```
Imports System.IO
Module ModuleOptions
    Dim temps As Integer = 60
    Dim mode As String = "Temps Limité"
    Dim BtnPauseActivé As String = "Non"
    'Getter pour le temps
    Public Function getTemps()
        Return temps
    End Function
    'Getter pour le mode
    Public Function getMode()
        Return mode
    End Function
    'Getter pour l'activation du bouton pause
    Public Function getBtnPauseActive()
        Return BtnPauseActivé
    End Function
    'setter pour le temps
    Public Sub setTemps(t As Integer)
        temps = t
    End Sub
    'Setter pour le mode
    Public Sub setMode(m As String)
        mode = m
    End Sub
    'Setter pour l'activation du bouton pause
    Public Sub setBtnPauseActive(p As String)
        BtnPauseActivé = p
    End Sub
End Module
```

Code des Modules

3) Module Interface

```
Module ModuleInterface
    Dim imgPlay As Image = Image.FromFile("play.png")
    Dim imgPause As Image = Image.FromFile("pause.png")
    'Sub qui lance la musique
    Sub Ambiance()
        My.Computer.Audio.Play("ambiance.wav", AudioPlayMode.BackgroundLoop)
    End Sub
    'Empêche le redimensionnement du formulaire en paramètre
    Sub ForbidResize(f As Form)
        f.FormBorderStyle = FormBorderStyle.Fixed3D
        f.MaximizeBox = False
    End Sub
    'Change la couleur du bouton en param en Aquamarine
    Public Sub swapColorIn(b As Button)
        b.BackColor = Color.Aquamarine
    End Sub
    'Change la couleur du bouton en param en medium sea green
    Public Sub swapColorOut(b As Button)
        b.BackColor = Color.MediumSeaGreen
    End Sub
      'Change l'image du bouton en param par play
    Public Sub swapBtnCostumePlay(b As Button)
        b.BackgroundImage = imgPlay
    End Sub
      'Change l'image du bouton en param par la pause
    Public Sub swapBtnCostumePause(b As Button)
        b.BackgroundImage = imgPause
    End Sub
End Module
```