**Université Espoir**

**Syllabus**

**C# (Projet Indépendant) – INF111**

Horaire : Lundi 9am ou sur Rendez-vous (Octobre 2018)

**Enseignant :** Seige Poteau

**Email :** spoteau@gmail.com

**Objectif**

L’Objectif de ce cours est d’aider l’étudiant à maitriser le langage très utilisé appelé C# à travers des projets pratiques : Applications Desktop et Web.

**Prérequis**

**Java II** – Le cours « Base de Données » est recommandé

**Matériels et Livres**

Les manuels sont sous forme PDF et seront mis à la disposition des étudiants dès le premier jour de classes.

**Contenu du Cours**

* Variables et Constants – Opérateurs Arithmétiques, Booléens/Logiques et autres
* Opérateurs de Comparaison – Incrémentation/Décrémentation ++x ; x++ ;
* Décision (ou Logique) – Boucles – Tableaux – Classes
* Fichiers – Concept de Base de Données
* Projets Pratiques utilisant Visual Studio 2017 et Microsoft SQL Server

**Règlements du cours**

L’étudiant doit être toujours présent, sauf en cas de problème étudié par l’administration et pour lequel la motivation est agréée par l’administration de l’UEspoir. 20 minutes après le début du cours est considérées comme une absence. Un étudiant qui quitte la salle de cours avant la fin du cours est considérée comme un absentéiste, sauf en cas de problème pour lequel le départ est agréé par le professeur. La présence d’un étudiant vaut 5% de sa note finale.

Les devoirs doivent être remis le jour fixé par le professeur, tous les devoirs non-soumis et non-achevés comptent pour un 0. Les devoirs retardés ne seront pas acceptés.

Tout étudiant se doit de respecter ses camarades de classes, son professeur, de ne pas apporter ni de la nourriture, ni de liquide dans laboratoires, et de ne pas pénétrer les laboratoires avec un ou des enfants.

Aux heures de cours un étudiant se doit de fermer son téléphone ou de le mettre sur vibreur. Un étudiant qui veut répondre à ou passer un appel se doit de laisser la salle de cours pour répondre à ou passer l’appel.

**La note finale est une combinaison de :**

|  |  |
| --- | --- |
| 1. **Projets** | **50%** |
| 1. **Examen de Mi-Chemin (Intra)** | **20%** |
| 1. **Examen Final** | **30%** |

**DEVOIRS**

**Semaines 1 à 5 (Manuel : « Programmation C# »)**

**DEVOIR 1**

Jeu de TiTaTo

1. Le jeu est un board de 9 casiers.
2. Le jeu est entre 2 joueurs représentés sur le board par un X ou un O
3. Gagner consiste à essayer d’avoir 3 X ou 3 O alignés.
4. Un début de ce projet est fourni (**Voir Exibit A**)

**DEVOIR 2**

Data Entry : Capturer les données sur une liste d’Etudiants.

1. Forme Desktop pour capturer les données
2. Classe représentant la structure des données sur un étudiant
3. Liste pour stocker les données des étudiants
4. Fichier pour sauvegarder / reprendre les données au commencement et la fin du cycle de vie du logiciel

**DEVOIR 3**

Prendre les informations sur les films les plus populaires dans le monde à partir du site : <http://api.themoviedb.org/3/movie/popular>

1. Forme Desktop pour montrer les informations et le poster (image) des films un par un
2. Boutons : Premier – Dernier – Précédent – Suivant pour consulter les films de la liste
3. Création d’une classe Film pour mettre les informations sur chaque film
4. Format Json à convertir afin de mettre les infos sur les films dans une liste (voir Devoir 2)
5. Mettre des images appropriées sur les boutons.
6. Info sur les films sur l’écran :
   1. Titre Original
   2. Poster
   3. Overview
   4. Date de sortie du film (Release Date)
7. Un début de ce projet est fourni (**Voir Exibit B**)

**DEVOIR 4**

Améliorer le Jeu de TiTaTo en faisant de l’ordinateur l’un des joueurs.

1. Le jeu est un board de 9 casiers.
2. Le jeu est entre 2 joueurs représentés sur le board par un X ou un O
3. Gagner consiste à essayer d’avoir 3 X ou 3 O alignés.

**DEVOIR 5**

**Web Application**

Améliorer le logiciel de CCPS (Calvary Christian Professional School)

**EXIBIT A**

public class NomEtScore

{

private String \_nom;

private int \_score;

public String Nom

{

get

{

return \_nom;

}

set

{

\_nom = value;

}

}

public int Score

{

get

{

return \_score;

}

set

{

\_score = value;

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Diagnostics;

using System.IO;

namespace WindowsFormsApp1

{

public partial class frmTiTaTo : Form

{

String[] positions = new String[9] {

"", "", "" ,

"", "", "" ,

"", "", ""

};

int CompteurMove = 0;

bool bPremierJoueur = true;

List <NomEtScore> ListeScores = new List<NomEtScore>();

String TextFileName = "\\titato.txt";

public frmTiTaTo()

{

InitializeComponent();

}

private String calculateWinner()

{

// Tabeau représentant les positions gagnantes

int[][] lines = new int[][]{

new int[]{0, 1, 2},

new int[]{3, 4, 5},

new int[]{6, 7, 8},

new int[]{0, 3, 6},

new int[]{1, 4, 7},

new int[]{2, 5, 8},

new int[]{0, 4, 8},

new int[]{2, 4, 6},

};

for (int i = 0; i < lines.Length; i++)

{

int a = lines[i][0], b = lines[i][1], c = lines[i][2];

if (positions[a] != "" &&

positions[a] == positions[b]

&& positions[a] == positions[c])

{

return positions[a];

}

}

return "";

}

/// <summary>

/// Le programme passe par ici en premier

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void frmTiTaTo\_Load(object sender, EventArgs e)

{

txtNom1.Focus();

btn0.Enabled = false; btn1.Enabled = false;

btn2.Enabled = false; btn3.Enabled = false;

btn4.Enabled = false; btn5.Enabled = false;

btn6.Enabled = false; btn7.Enabled = false;

btn8.Enabled = false;

// Read file

if (File.Exists(Application.StartupPath + TextFileName))

{

StreamReader sr = new StreamReader(Application.StartupPath + TextFileName);

while (!sr.EndOfStream)

{

String[] ligne = sr.ReadLine().Split('=');

NomEtScore nes = new NomEtScore();

nes.Nom = ligne[0];

nes.Score = int.Parse(ligne[1]);

ListeScores.Add(nes);

}

sr.Close();

UpdateEcran();

}

}

/// <summary>

/// Le joueur veut commencer à jouer

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void btnCommencez\_Click(object sender, EventArgs e)

{

// Au cas où c'est un nouveau jeu

bPremierJoueur = true;

//

if (txtNom1.Text.Trim() == String.Empty

|| txtNom2.Text.Trim() == String.Empty)

{

lblMessage.Text = "Entrez les noms des joueurs!";

lblMessage.ForeColor = Color.Red;

txtNom1.Focus();

return;

}

lblMessage.Text = String.Empty;

txtNom1.Text = ArrangezNom(txtNom1.Text);

txtNom2.Text = ArrangezNom(txtNom2.Text);

txtNom1.Enabled = false;

txtNom2.Enabled = false;

// Rendre les boutons prêts

btn0.Enabled = true; btn0.Text = "";

btn1.Enabled = true; btn1.Text = "";

btn2.Enabled = true; btn2.Text = "";

btn3.Enabled = true; btn3.Text = "";

btn4.Enabled = true; btn4.Text = "";

btn5.Enabled = true; btn5.Text = "";

btn6.Enabled = true; btn6.Text = "";

btn7.Enabled = true; btn7.Text = "";

btn8.Enabled = true; btn8.Text = "";

// Ré-intinialiser compteur

CompteurMove = 0;

// Ré-intinialiser positions

for (int i=0; i <=8; i++)

{

positions[i] = "";

}

//

lblMessage.Text = String.Empty;

}

/// <summary>

/// Méthode de Support pour ne pas répéter les mêmes codes par click

/// </summary>

/// <param name="btn"></param>

/// <param name="index"></param>

private void GererClique(Button btn, int index)

{

if (CompteurMove > 8)

return; // Ne rien faire

if (bPremierJoueur)

{

bPremierJoueur = false;

positions[index] = "X";

btn.Text = "X";

btn.ForeColor = Color.Red;

if (calculateWinner() == "X")

{

lblMessage.Text = "Joueur 1 a gagné !!!!";

CompteurMove = 9;

EnregistrezScoresDansListe(txtNom1.Text);

}

}

else

{

bPremierJoueur = true;

positions[index] = "O";

btn.Text = "O";

btn.ForeColor = Color.Blue;

if (calculateWinner() == "O")

{

lblMessage.Text = "Joueur 2 a gagné !!!!";

CompteurMove = 9;

EnregistrezScoresDansListe(txtNom2.Text);

}

}

CompteurMove++;

}

/// <summary>

/// Nettoyage des Noms

/// </summary>

/// <param name="value"></param>

/// <returns></returns>

String ArrangezNom(String value)

{

String valRetourne = value.Trim();

valRetourne = value.Substring(0, 1).ToUpper() +

value.Substring(1).ToLower();

int index = valRetourne.IndexOf(" ");

if (index > 0)

{

String moso1 = valRetourne.Substring(0, index) + " ";

String moso2 = valRetourne.Substring(index + 1).Trim();

valRetourne = moso1 + moso2.Substring(0, 1).ToUpper() +

moso2.Substring(1).ToLower();

}

return valRetourne;

}

/// <summary>

/// Enregistrer les scores dans une liste de noms

/// </summary>

/// <param name="nom"></param>

private void EnregistrezScoresDansListe(String nom)

{

foreach (NomEtScore nes in ListeScores)

{

if (nes.Nom == nom)

{

nes.Score++;

return;

}

}

// Le nom n'est pas trouvé

NomEtScore nouvo = new NomEtScore();

nouvo.Nom = nom;

nouvo.Score = 1;

ListeScores.Add(nouvo);

//

UpdateEcran();

}

/// <summary>

/// Enregistrer les scores dans un fichier

/// </summary>

private void EnregistrezScoresDansFichierV1()

{

String path = Application.StartupPath.ToString();

path += TextFileName;

StreamWriter sw = null;

try

{

sw = new StreamWriter(path);

foreach (NomEtScore nes in ListeScores)

{

sw.WriteLine(nes.Nom + "=" + nes.Score);

}

sw.Close();

}

catch(Exception ex)

{

Debug.WriteLine(ex.Message);

}

finally

{

if (sw != null)

sw.Close();

sw = null;

}

}

/// <summary>

/// Rafraichir les infos sur l'écran

/// </summary>

void UpdateEcran()

{

lvScoreRecord.Items.Clear();

foreach (NomEtScore nes in ListeScores)

{

ListViewItem lvItem = new ListViewItem();

lvItem.Text = nes.Nom;

lvItem.SubItems.Add(nes.Score.ToString());

lvScoreRecord.Items.Add(lvItem);

}

}

/// <summary>

/// Enregistrer les scores dans un fichier - Améliorer plus tard

/// par les étudiants en utilisant la Sérialisation

/// </summary>

/// <param name="nom"></param>

private void EnregistrezScoresDansFichierV2(String nom)

{

String path = "titato.txt";

System.IO.StreamWriter sw = new System.IO.StreamWriter(path);

}

private void btn0\_Click(object sender, EventArgs e)

{

GererClique(btn0, 0);

}

private void btn1\_Click(object sender, EventArgs e)

{

GererClique(btn1, 1);

}

private void btn2\_Click(object sender, EventArgs e)

{

GererClique(btn2, 2);

}

private void btn3\_Click(object sender, EventArgs e)

{

GererClique(btn3, 3);

}

private void btn4\_Click(object sender, EventArgs e)

{

GererClique(btn4, 4);

}

private void btn5\_Click(object sender, EventArgs e)

{

GererClique(btn5, 5);

}

private void btn6\_Click(object sender, EventArgs e)

{

GererClique(btn6, 6);

}

private void btn7\_Click(object sender, EventArgs e)

{

GererClique(btn7, 7);

}

private void btn8\_Click(object sender, EventArgs e)

{

GererClique(btn8, 8);

}

private void frmTiTaTo\_FormClosing(object sender, FormClosingEventArgs e)

{

if (ListeScores.Count > 0)

{

EnregistrezScoresDansFichierV1();

}

}

}

}

**EXIBIT B**

class Movie

{

public int vote\_count;

public int id;

public bool video;

public double vote\_average;

public String titleVenom;

public double popularity;

public String poster\_path;

public String original\_language;

public String original\_title;

public String genre\_ids;

public String backdrop\_path;

public bool adult;

public String overview;

public String release\_date;

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Net.Http;

using System.Text;

using System.Configuration;

using System.Xml;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Runtime.Serialization.Json;

using System.Xml.Linq;

using System.Xml.XPath;

namespace InfoSurFilms

{

public partial class frmMain : Form

{

/// <summary>

/// Liste des films

/// </summary>

List<Movie> listeDeFilms = new List<Movie>();

/// <summary>

/// Position de la poster courant du film sur l'écran

/// </summary>

int Position = 0;

/// <summary>

/// adresse du site où obtenir les infos sur les films

/// </summary>

String siteURL = "http://api.themoviedb.org/3/movie/popular";

public frmMain()

{

InitializeComponent();

}

/// <summary>

/// Lire le résultat du site et mettre les informations

/// obtenues dans une liste de films

/// </summary>

async void ObtenirJsonString()

{

// Get the object used to communicate with the server.

String Url\_la = siteURL + "?api\_key=" + ConfigurationSettings.AppSettings.Get("jsonCode").ToString();

// Create a New HttpClient object.

HttpClient client = new HttpClient();

// Contenu obtenu du site

string repons = "";

// Call asynchronous network methods in a try/catch block to handle exceptions

try

{

HttpResponseMessage response = await client.GetAsync(Url\_la);

response.EnsureSuccessStatusCode();

repons = await response.Content.ReadAsStringAsync();

// Above three lines can be replaced with new helper method below

// string repons = await client.GetStringAsync(uri);

Console.WriteLine(repons);

}

catch (HttpRequestException e)

{

Console.WriteLine("\nException Caught!");

Console.WriteLine("Message :{0} ", e.Message);

}

// Need to call dispose on the HttpClient object

// when done using it, so the app doesn't leak resources

client.Dispose();

XmlDictionaryReader reader = JsonReaderWriterFactory.CreateJsonReader(

Encoding.UTF8.GetBytes(repons), new System.Xml.XmlDictionaryReaderQuotas());

XElement root = XElement.Load(reader);

// The fields we'd like to extract

XElement form = root.XPathSelectElement("//page");

XElement status = root.XPathSelectElement("//total\_results");

XElement type = root.XPathSelectElement("//total\_pages");

// Field set

IEnumerable<XElement> fields = root.XPathSelectElements("//results/\*");

foreach(XElement xel in fields)

{

Movie movie = new Movie();

movie.original\_title = xel.XPathSelectElement("original\_title").Value.ToString();

movie.poster\_path = xel.XPathSelectElement("poster\_path").Value.ToString();

movie.overview = xel.XPathSelectElement("overview").Value.ToString();

movie.release\_date = xel.XPathSelectElement("release\_date").Value.ToString();

listeDeFilms.Add(movie);

}

}

void MontreUnPosterSurEcran()

{

if (listeDeFilms.Count > 0) // Liste est remplie

{

Movie m = listeDeFilms[Position];

picbMain.ImageLocation = "https://image.tmdb.org/t/p/w500" + m.poster\_path;

picbMain.Load();

}

}

private void frmMain\_Load(object sender, EventArgs e)

{

ObtenirJsonString(); // + remplir la liste des films

MontreUnPosterSurEcran(); // Remplir Ecran avec info sur un film

}

private void btnPremier\_Click(object sender, EventArgs e)

{

Position = 0;

MontreUnPosterSurEcran();

}

private void btnDernier\_Click(object sender, EventArgs e)

{

Position = listeDeFilms.Count - 1;

MontreUnPosterSurEcran();

}

}

}