Prise en main de Visual Studio et découverte de C#

**Disclaimer : Il est important de comprendre le code présent dans le TP que vous allez copier-coller.**

**Il est aussi important de comprendre comment les actions sont imbriquées pour arriver au résultat final. Sans cette compréhension, vous ne pourrais pas progresser.**

## Partie 1 : Prise en main de différents éléments

### Phase 1 : Création du projet

Attention, vous devez lancer Visual Studio 2022 et non Visual Studio Code. Visual Studio doit déjà être installé sur les postes.

Une image contenant texte, Police, capture d’écran, logo

Description générée automatiquement

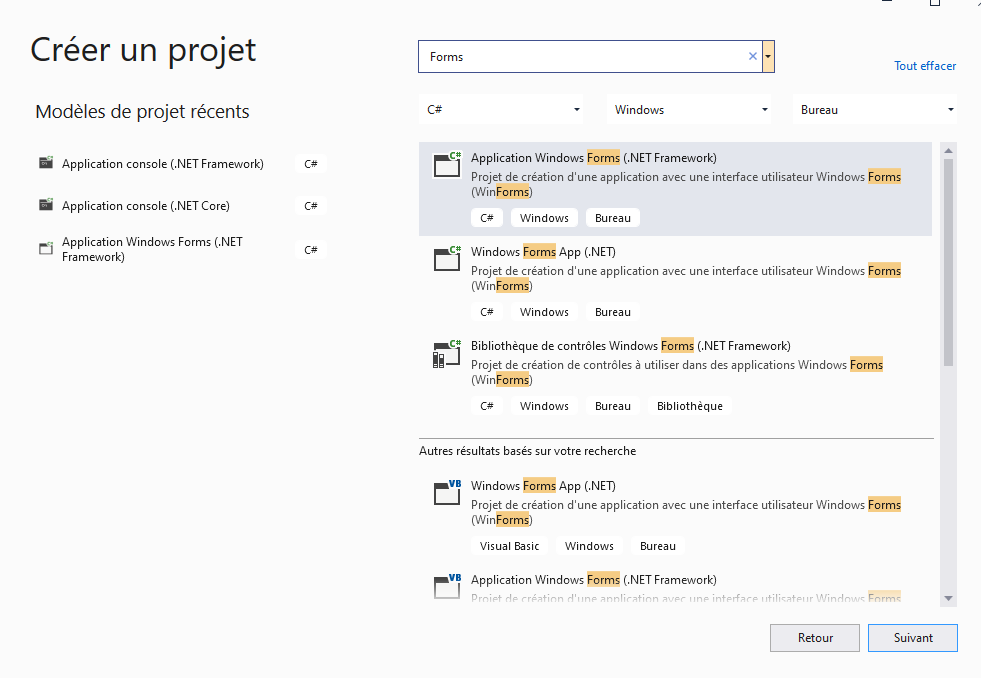
Vous devez alors avoir l’écran suivant : Une image contenant texte, capture d’écran, Police

Description générée automatiquement

Pour créer votre application, vous devez créer un « projet » :

- En choisissant le lien dédié sur la page précédente,

L’IDE vous demande alors de choisir le type de projet que vous souhaitez mettre en place. Nous travaillerons avec le type « Application Windows Forms (.NET Framework)».



L’interface vous demande le nom de votre projet, par défaut WindowsFormsApp1. Renommez le nom du projet afin de ne pas laisser le nom par défaut ainsi que **l'emplacement du projet** (Dans votre dossier personnel ou sur votre clé USB par exemple).

Une image contenant texte, capture d’écran, Police, nombre

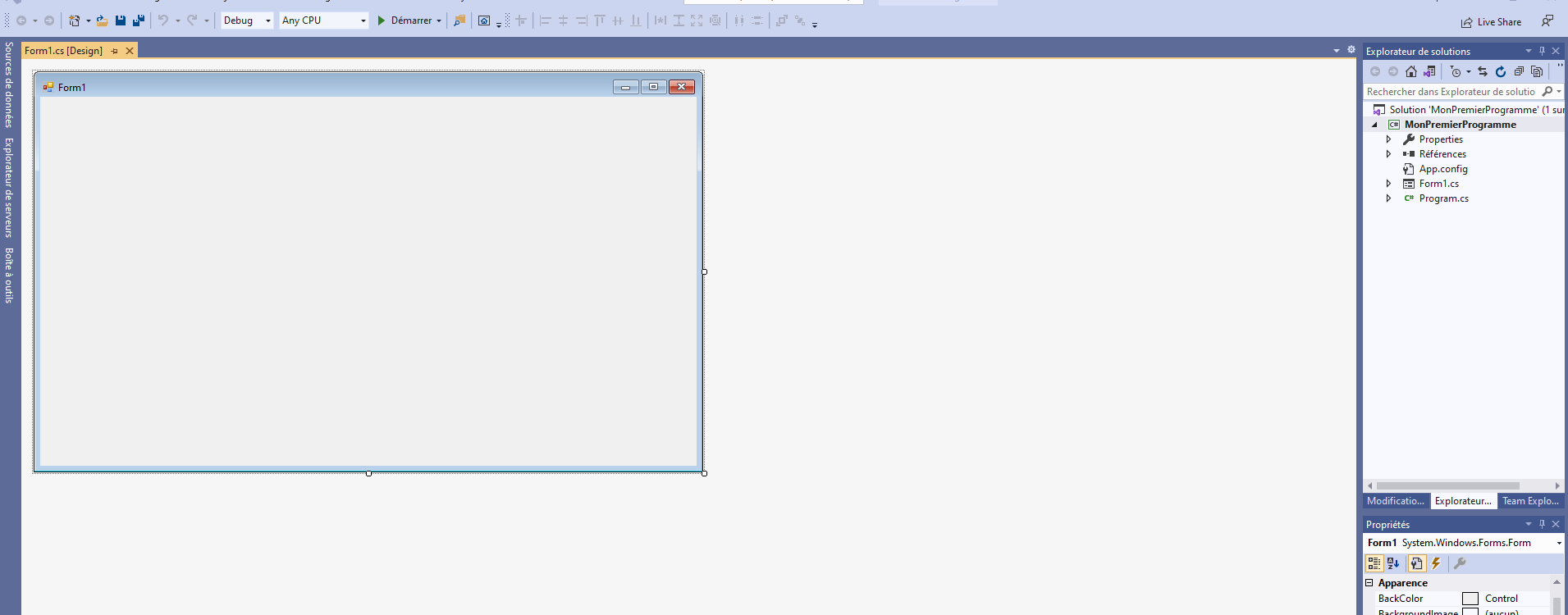
Description générée automatiquement

Selon la version de Visual Studio, il est possible que certains types de projets n’apparaissent pas. Vous utiliserez principalement les types qui suivent :

- **Application Windows Form** : des projets basés sur des interfaces Windows simples,

- **Bibliothèques de classes** : des projets pour créer vos propres composants.

L’interface de l’IDE s’ouvre enfin, elle se présente ainsi (vue partielle) :



### Phase 2 : Enregistrement du projet

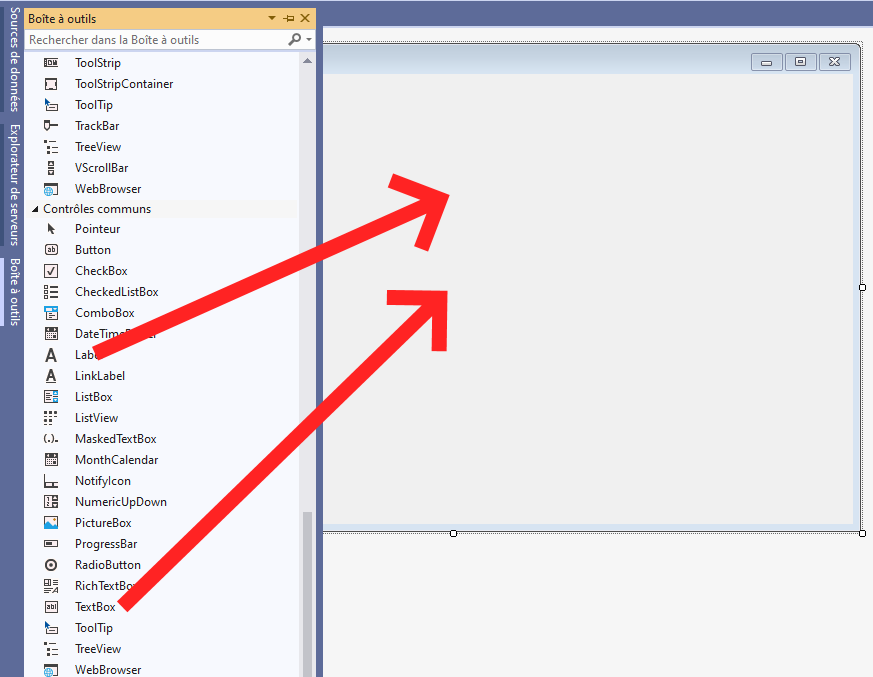
Pour enregistrer le projet pour la première fois, il suffit d’utiliser la démarche traditionnelle sous Windows (menu Fichier – Enregistrer Tout) ou utiliser le raccourcis clavier CTRL + la touche « S ».

### Phase 3 : Insertion de contrôles commun dans une fenêtre

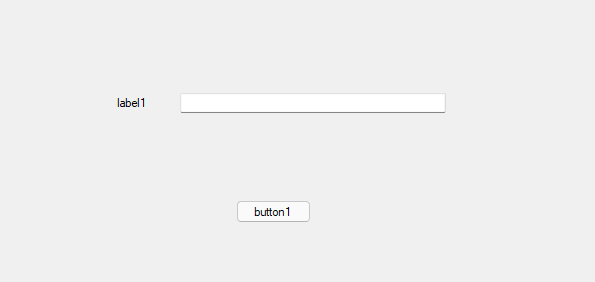
Pour insérer un contrôle, il faut ouvrir la boite à outil. C’est là que tous les contrôles que vous utiliserez seront présents. (Il faut être sur une vue pour que ce menu apparaisse).

### 

Maintenant vous devez disposer sur l’écran un label, un bouton et une textbox. Pour cela il suffit de faire un glisser déposer depuis la barre d’outils vers votre écran. **Attention, le volet « Boite à outils » n’apparait que pour le mode design.**



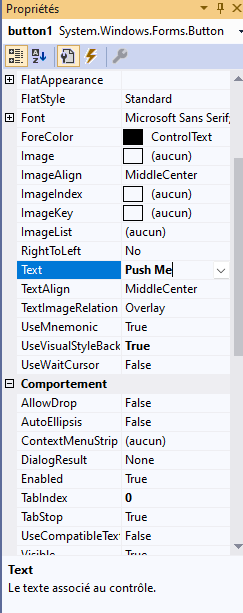
Une fois réalisé, essayez d’avoir le résultat suivant :



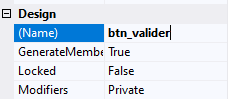
### Phase 4 : Personnalisation des contrôles positionné dans la fenêtre

Pour modifier le texte du bouton il faut aller dans les paramètres du bouton. Pour cela, il faut sélectionner le bouton en cliquant dessus. Les paramètre du bouton devrait apparaitre sur le côté droit (S’il n’est pas apparu, utiliser le raccourcis ALT+ENTRER).

Utiliser le champ « Text » pour changer le texte du bouton

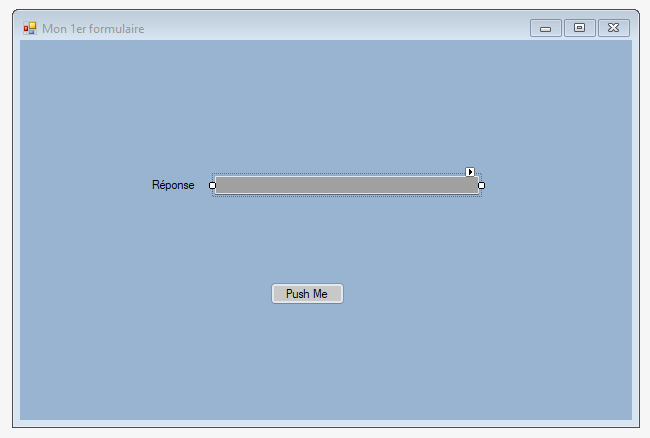


Changer le nom du bouton « button1 » en « btn\_valider ».



Changer le nom du textbox en « txt\_afficher » et le nom du label en « lbl\_rep »

Dans les propriétés du textbox, cherchez la propriété enable et passez-la à « False ». Puis tentez de reproduire l’affiche suivant grâce aux différents paramètres dans les propriétés.



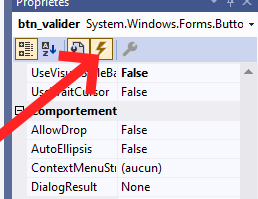
**Il est important de nommer correctement les objets que vous créez. En effet, je vous conseille de nommer les objets de cette façon : « Type d’objet » \_ « Fonction ».**

**Cette bonne pratique vous fera gagner un temps précieux surtout lorsqu’il y a beaucoup de type d’objet sur le même formulaire.**

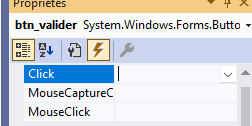
**IL est essentiel de nommer de la même façon l’ensemble des objets que vous créez**

### Phase 5 : Mise en place de code dans l’application

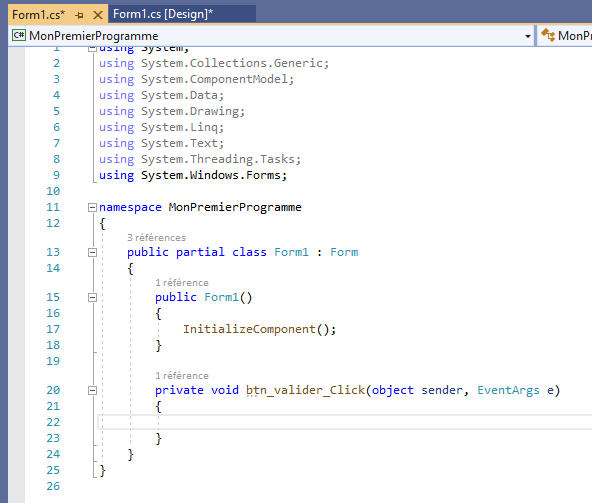
Pour afficher un message lors de l’appuis du bouton, il faut cliquer sur le bouton et cliquer sur le bouton évènement dans les propriétés.



Double cliquez sur le champ à coté de Click

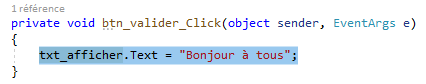


L’écran suivant devrait apparaitre



Insérez le code suivant :

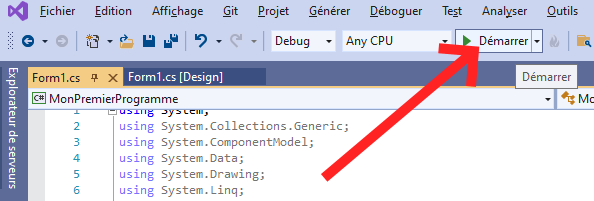
txt\_afficher.Text = "Bonjour à tous"; //Bonjour le commentaire



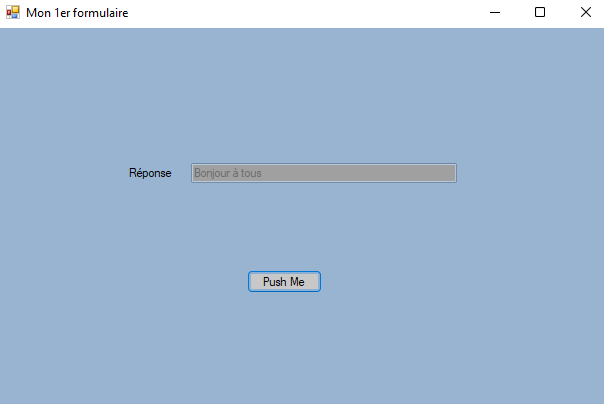
Ici nous utilisons une action (l’appuis sur le bouton) pour que le code s’exécute.

### Phase 6 : Test de l’application

Pour lancer l’application il faut cliquer sur le bouton démarrer.



L’écran suivant devrais s’afficher et lors du clic sur le bouton le message devrait s’afficher dans la textbox.

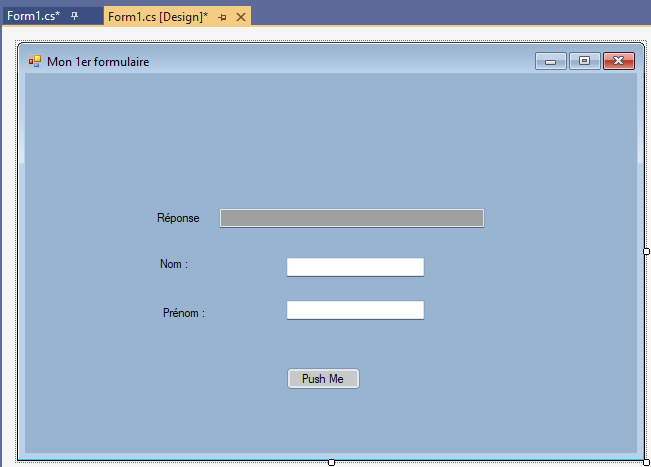


**Si vous avez fermé votre onglet design il suffit de double cliquer sur le form correspondant.**

**Si vous avez fermé votre onglet contenant le code, il suffit de sélectionner le form et d’appuyer sur F7**

### Phase 7 : Récupérer des données dans des textbox et afficher un message

Reproduire l’affichage suivant en n’oubliant pas de renommer les textbox que vous venez d’ajouter « txt\_ <nom à afficher> »



Modifier votre code par celui-ci :

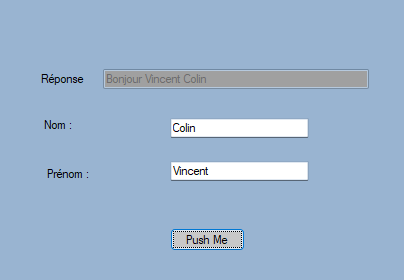
string result = "Bonjour ";

result += txt\_nom.Text ;

txt\_afficher.Text = result;

Vérifier que lorsque vous remplissez votre nom et que vous cliquez sur le bouton le message « Bonjour <Votre nom> » apparait.

Modifier votre code pour que le message « Bonjour <Votre prénom>  <Votre nom> »



### Phase 8 : Contrôle des données dans les textbox

Pour forcer l’utilisateur à remplir les champs on utilise le code suivant :

if (txt\_nom.Text != "")

{

// votre code

}

Adaptez le code pour tester si tous les champs sont remplis.

### Phase 9 : Afficher un message dans une boite de dialogue à l’utilisateur lorsque le nom ou le prénom n’est pas remplis

Pour afficher une boite de message à l’utilisateur, on utilise le code suivant : MessageBox.Show("Aucun nom n'est rentré", "Erreur",MessageBoxButtons.OK,MessageBoxIcon.Warning);

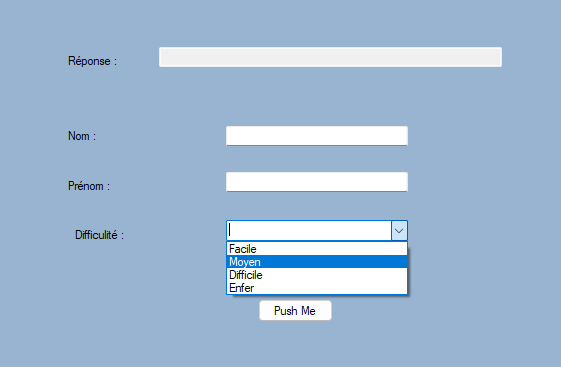
Une image contenant texte, capture d’écran, logiciel, Système d’exploitation

Description générée automatiquement

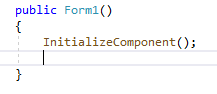
Affichez une boite de dialogue lorsque le nom ou le prénom n’est pas rentré

### Phase 10 : Utilisation d’un combobox

Reproduire l’affichage suivant :

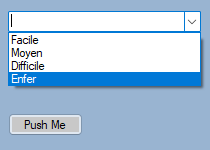


Pour charger un combobox au lancement de l’écran il faut ajouter son code ici :



Pour ajouter des éléments on utilise le code suivant : <Nom du combobox>.Items.Add("item");

Ajouter les items suivant dans le combo box : Facile, Moyen, Difficile, Enfer



Lorsque la difficulté « Enfer » est sélectionné, le fond d’écran doit devenir rouge et le texte de chaque label doit être en gras.

Pour cela on va utiliser l’événement SelectIndexChanged du combobox

Pour connaitre la valeur sélectionné on utilise le code suivant : <Nom du combobox>.SelectedItem

Pour changer la couleur de fond de la Form on utilise le code suivant : this.BackColor = Color.Red;

Pour mettre en gras le text d’un label on utilise le code suivant :

<Nom du label>.Font = new Font(<Nom du label>.Font, FontStyle.Bold);

Pour retrouver la couleur d’origine lorsque une autre valeur que « Enfer » est sélectionnée on utilise le code suivant : Color.FromArgb(153, 180, 209);

Remettre les labels en normal lorsqu’une autre valeur que « Enfer » est sélectionnée.

Ajouter au texte lorsque l’on appuis sur le bouton : \r\n La difficultés sélectionnée est : <Valeur sélectionnée>

### Phase 11 : Résolution d’une erreur typique

Si vous double cliquer sur le champ nom

Une image contenant capture d’écran, texte, ligne, Rectangle

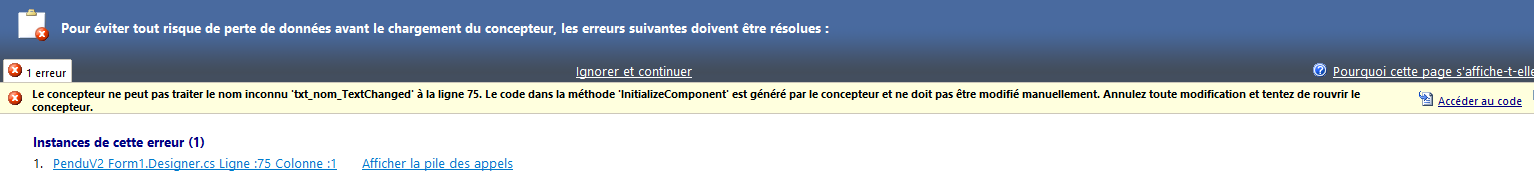
Description générée automatiquement

Vous devriez arriver sur l’écran où sont stockées les différentes actions.

Une image contenant texte, ligne, Police, Tracé

Description générée automatiquement

Vous venez de créer une action non désirée. Si vous effacez le code généré, vous obtiendrez l’erreur suivante en retournant sur la page de génération de l’affichage.



Pour résoudre cette erreur, il faut cliquer sur le lien Accéder au code

Une image contenant texte, Police, capture d’écran, Rectangle

Description générée automatiquement

Puis il faut effacer seulement la ligne qui possède l’erreur

Une image contenant texte, Police, ligne, capture d’écran

Description générée automatiquement

**ATTENTION SI VOUS EFFACER PLUSIEURS LIGNES QUI NE SONT PAS EN ERREUR, VOTRE PROJET RISQUE DE NE PLUS FONCTIONNER !!**

En retournant dans le Design le formulaire devrais réapparaitre.

### Phase 12 : Création d’un nouveau form et liaison

Nous allons créer un nouveau Form, pour cela il faut cliquez droit sur le nom de votre projet et choisir : Ajouter > Nouvel élément

Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement

Il faut ensuite choisir Formulaire WindowForm

Une image contenant texte, capture d’écran, logiciel, nombre

Description générée automatiquement

**N’oubliez pas de changer le nom de votre formulaire !! Puis cliquez sur Ajouter**

Normalement le nouveau formulaire devrait apparaitre dans votre explorateur de solution.

**Une image contenant texte, capture d’écran, logiciel, Icône d’ordinateur

Description générée automatiquement**

En double cliquant sur votre nouveau formulaire vous pourrez accéder au design

Reproduire l’affichage suivant :

Une image contenant texte, capture d’écran, affichage, logiciel

Description générée automatiquement

**N’oubliez pas de bien nommer vos boutons, les 3 textbox, ainsi que la picturebox et le panel.**

**Le panel nous servira dans un autre TP.**

**La picturebox permettra de faire défiler des images.**

**La grande textbox permettra d’afficher le mot à trouver.**

**Les deux textbox en haut à droite permettrons d’afficher plus tard le nom, prénom et la difficulté choisi par le joueur.**

Les textbox ne doivent pas pouvoir être cliquable.

### Phase 13 : Modification de l’ancien form

Modifiez le Form1 pour qu’il est cet aspect

Une image contenant texte, capture d’écran, logiciel, conception

Description générée automatiquement

Des erreurs dans votre code devrais apparaitre, corrigez-les !!

Codez le contrôle de contenu des champs Nom, Prénom et Difficulté lors de l’appuis sur le bouton valider.

Pour contrôler qu’un combobox à une valeur sélectionnée, on utilise le code suivant <combobox>. SelectedIndex > -1

### Phase 14 : Appel d’un form à un autre

Une fois tous les contrôles réalisés, vous pouvez faire appel à votre 2éme Form en utilisant le code suivant :

Jeu J = new Jeu();

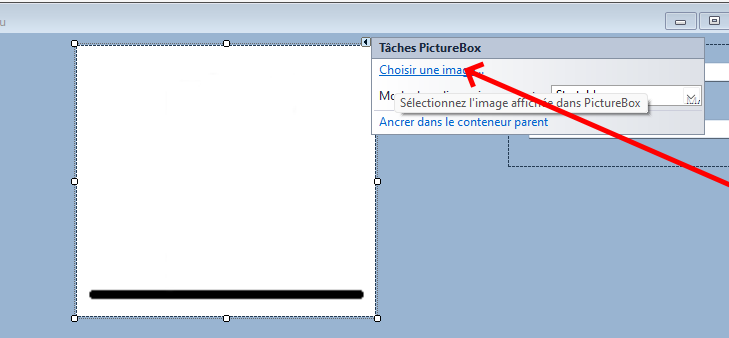
J.Show();

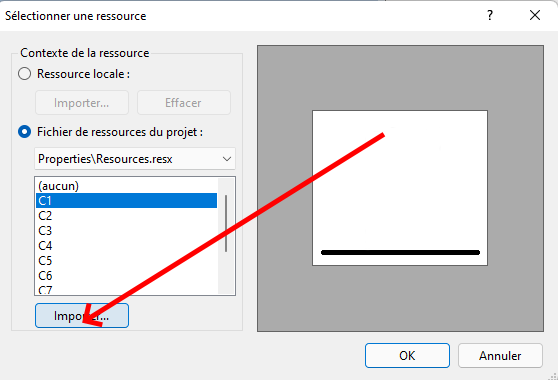
this.Hide();

Ici le code suivant fait référence

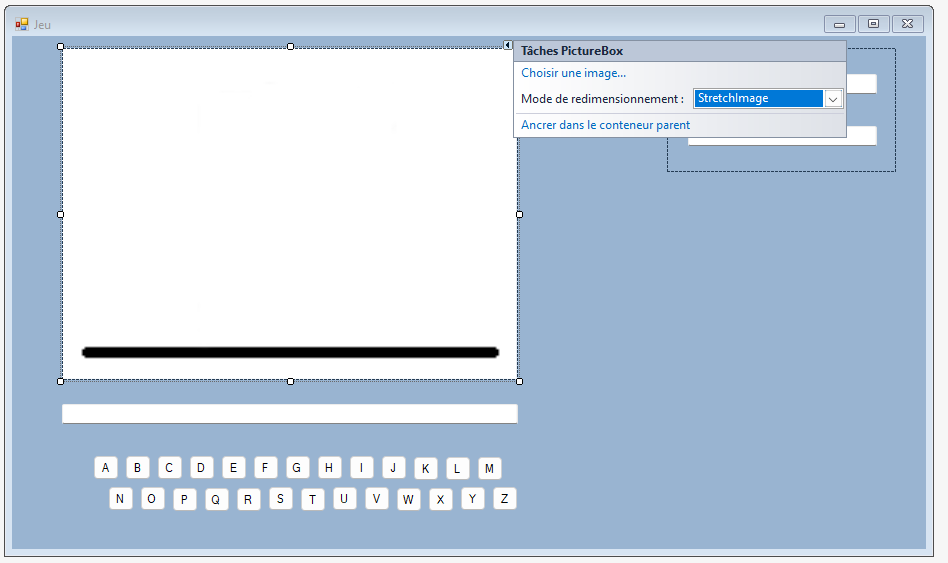
### Phase 15 : Affichage de l’image 1 dans le pictureBox

Pour ajouter les images il vous suffit de les importer



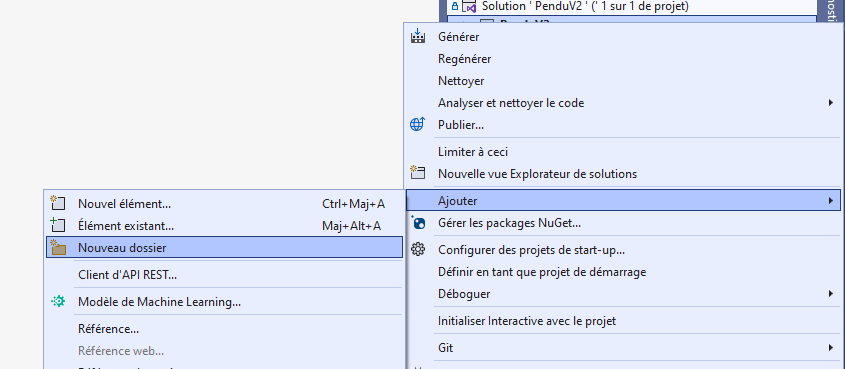


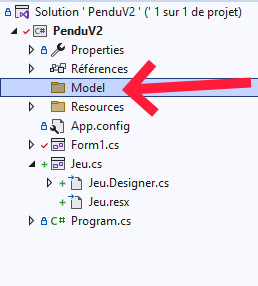
Et sélectionnez l’ensemble des images du projet. Puis une fois les images importées, reproduisez l’écran suivant.



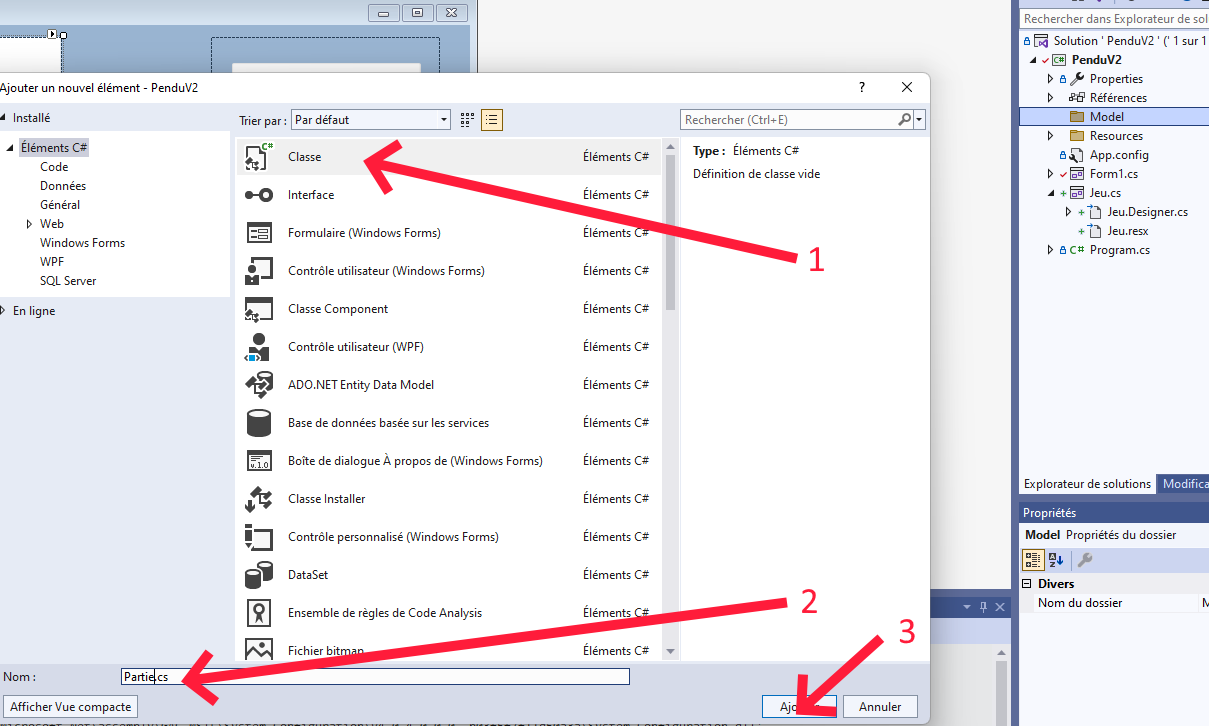
Maintenant nous allons réaliser le code permettant de changer l’image en fonction d’un nombre. Pour cela nous allons créer un nouveau fichier contenant le code que nous allons utiliser dans le formulaire jeu.

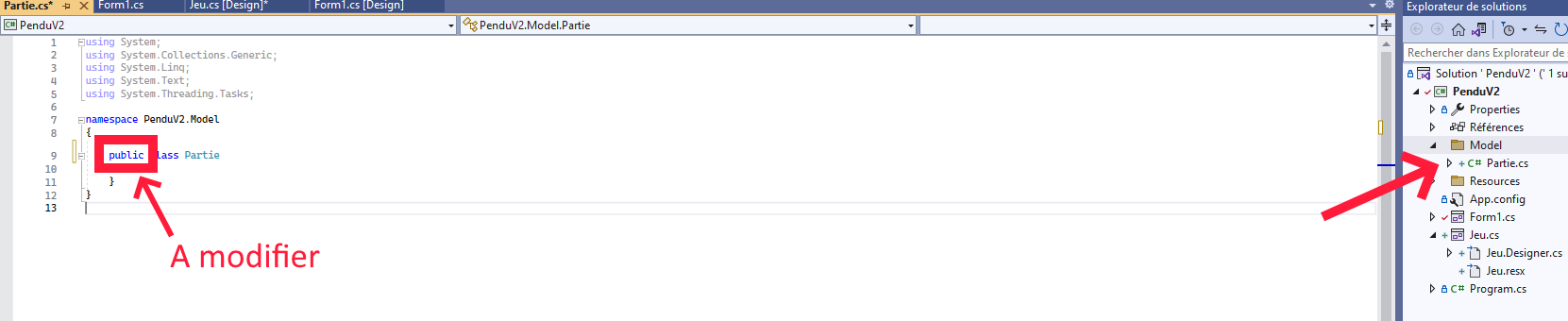
Nous allons ajouter un dossier qui va contenir l’ensemble de nos objets que nous allons créer





Maintenant nous allons créer la **classe** « Partie »





Nous allons déclarer un attribut nommé « nbEssais » de type Int en public



Maintenant nous allons réaliser une méthode permettant de changer l’image d’une pictureBox en fonction d’un nombre. La méthode prendra en paramètre une pictureBox

public void changerIMG(PictureBox pb\_pendu)

{

switch(nbEssais)

{

case 0:

pb\_pendu.Image = Properties.Resources.C1;

break;

// Code à completer pour prendre en compte toutes les images

default:

pb\_pendu.Image = Properties.Resources.C1;

break;

}

}

### Phase 16 : Coder le pendu

Nous allons déclarer 2 attributs de type String en public, une pour stocker le mot à trouver, une pour le mot à afficher. Pour cela on utilise le code suivant :

### public String motatrouver;

### 

**Créer l’autre attribut**

Nous allons créer une méthode qui permettra de vérifier si la lettre que l’on a choisie est présent dans le mot à trouver.

Pour cela créez une nouvelle méthode publique qui prend en paramètre une lettre. Appelez cette fonction « verification ».

Dans cette fonction, on vérifie que la lettre que l’on a tapée est présente dans le mot à trouver.

Si la lettre est présente, on met à jour le mot à afficher dans la textbox avec la nouvelle lettre trouvée

Si la lettre n’est pas présente dans le mot à trouver, alors on met à jour le compteur d’erreurs

Le code suivant est à compléter pour que le fonctionnement corresponde à la demande

public void verifier(string lettretape, TextBox txt\_afficher)

{

char[] cArray = motatrouver.ToCharArray();

char[] motaff = motaafficher.ToCharArray();

for (//Paramètres à completer)

{

if (cArray[i] == Convert.ToChar(lettretape))

{

motaafficher += lettretape;

}

else

motaafficher += motaff[i];

}

txt\_afficher.Text = motaafficher;

### }

**Identifiez a quoi sert le type** char[]

### Phase 17 : Génération du mot à afficher

Pour cela nous allons créer une méthode permettant de générer le mot à afficher en fonction du mot à trouver.

Exemple : le mot à trouver est « MAISON » donc le mot à afficher sera « \_\_\_\_\_\_ »

Pour cela il faut créer une méthode permettant de retourner une chaine de caractère ne contenant que le caractère « \_ » du même nombre que le nombre de lettre que le paramètre envoyé (une chaine de caractère).

public void genererMotAfficher()

{

string motaff = "";

//Code à completer

motaafficher = motaff;

}

### Phase 18 : Génération du mot à trouver

Il faut réaliser une fonction permettant de sélectionner un mot de manière aléatoire à partir d’une liste et de renvoyer. Pour ajouter une liste obligatoire dans les paramètres, on utilise le code suivant :

List<String> listeATrouver

Pour générer un nombre aléatoire, on utilise le code suivant :

Random aleatoire = new Random();

int nbAleatoire = aleatoire.Next(listeATrouver.Count);

Pour sélectionner un mot dans la liste on utilise le code suivant :

string motchoisi = listeATrouver[nbAleatoire].ToUpper();

### Phase 19 : Liaison de notre Classe Partie avec notre formulaire Jeu

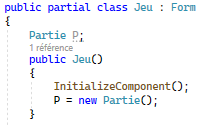
Allez dans le code du formulaire Jeu. Nous allons instancier la classe Partie, ce qui nous permettra de créer un objet que nous pourrons utiliser dans le code du formulaire. On utilise le code suivant :

Partie P ;

P = new Partie();

**Expliquez la raison de la déclaration séparée de l’instanciation.**

**L’importation de la classe/modèle**



On met le code juste après la ligne « InitializeComponent(); », comme cela le code s’exécutera lorsque la fenêtre sera généré

Maintenant nous allons générer un mot à trouver et un mot à afficher. Pour cela nous allons créer une méthode init

private void init()

{

List<string> listeMotaTrouver = new List<string> { "Francophile", "Chlorophylle", "Conspirateur", "Qualification", "Attraction", "Cornemuse", "Tourisme", "Diapason", "Brouhaha" };

P.choisirMotATrouver(listeMotaTrouver);

P.genererMotAfficher();

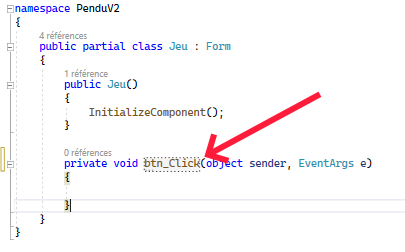
txt\_afficher.Text = P.motaafficher;

}

Il faut appeler init après P = new Partie(); dans le constructeur Jeu()

### Phase 20 : Gestion des boutons

Allez dans le code du formulaire Jeu et créez une méthode btn\_Click



Ajouter le code suivant :

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

{

P.verifier(((Button)sender).Text.ToString(), txt\_afficher);

P.changerIMG(pb\_pendu);

}

Lier cet évènement sur le clic de tous les boutons. Vous pouvez désormais tester si votre code marche.

Vous devriez trouver une erreur de fonctionnement lorsque vous appuyez sur un bouton. En effet le mot à afficher grandit. Trouvez l’erreur et corrigez là.

### Phase 21 : Rendre inutilisable les boutons qui ont déjà été appuyé

Pour cela vous devez ajouter une ligne de code à votre événement btn\_Click

((Button)sender) vous permettra d’identifier le bouton qui a été appuyé. Il faut maintenant utiliser la propriété enable pour rendre le bouton non utilisable une deuxième fois dans la partie.

### Phase 22 : Gérer les conditions de victoire et de défaite

Pour cela nous allons vérifier à la fin du code dans l’événement btn\_Click si le nombre maximum d’essai sans trouver le mot est atteint. Si le joueur gagne, un message apparait et propose de rejouer une partie. Si le joueur perd un message apparait montrant la solution et propose de rejouer une partie. Pour cela on utilise le code suivant dans la méthode btn\_click :

P.victoire(this,txt\_afficher, listeMotaTrouver,pb\_pendu);

Ensuite vous allez créer 2 méthodes dans votre classe Partie. Une méthode victoire et une méthode remiseAZero

Pour déclarer la méthode victoire, on utilise le code suivant :

public void victoire(Form formulaireJeuActif, TextBox txt\_motAafficher, List<string> listeMotaTrouver,PictureBox pbpendu)

Dans la méthode victoire, il faut :

* Déclarer une variable msg de type DialogResult
* Vérifier que le nombre d’essai n’est pas supérieur ou égale à 9
  + Si oui alors on affiche le message suivant :
  + msg = MessageBox.Show("Vous avez perdu !! \r\nVous deviez trouver le mot: " + motatrouver + "\r\nVoulez vous faire une autre partie ??", "You loose",MessageBoxButtons.YesNo,MessageBoxIcon.Exclamation,MessageBoxDefaultButton.Button1);

Si oui alors on appelle la méthode (msg == DialogResult. Yes)

* + - remiseAZero(formulaireJeuActif, txt\_motAafficher, listeMotaTrouver, pbpendu) ;
    - Si non on revient à l’écran précédent
    - Form1 Accueil = new Form1();
    - Accueil.Show();
    - formulaireJeuActif.Hide();
* Faire la même chose lorsque le joueur gagne

Voici le code de la méthode remiseAZero

public void remiseAZero(Form formulaireJeuActif,TextBox txt\_motAafficher, List<String> listeMotaTrouver, PictureBox pb\_pendu)

{

//Mettre à vide les attribut motaafficher et motatrouver

//Mettre le nombre d’essais à 1

//Choisir un nouveau mot à trouver

//Générer un nouveau mot à afficher

txt\_motAafficher.Text = motaafficher;

foreach (Control c in formulaireJeuActif.Controls)

{

if (c.GetType() == typeof(Button))

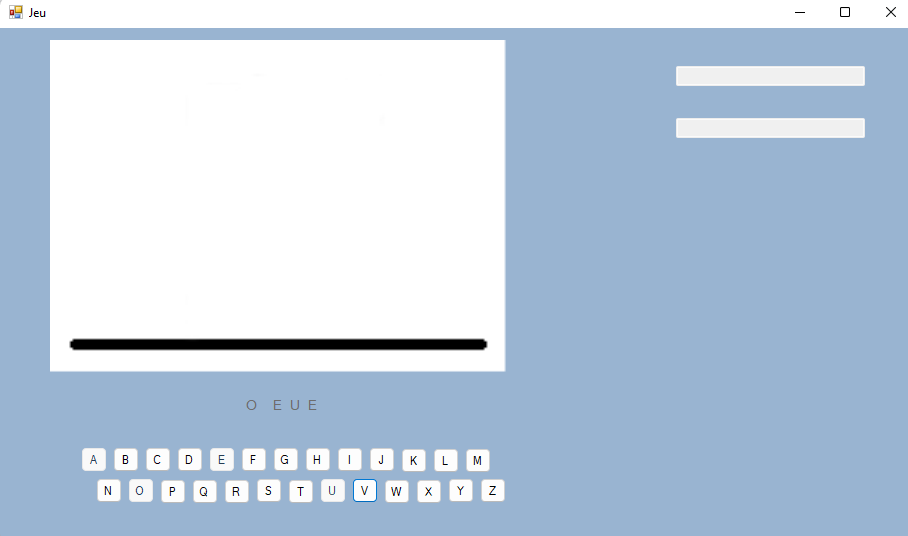
c.Enabled = true;

}

//mettre l’image à jour

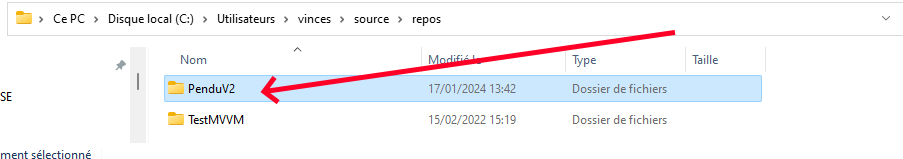
}

Une fois fini, tester si tout fonctionne bien, puis adapter votre application pour que le résultat ressemble à cela :

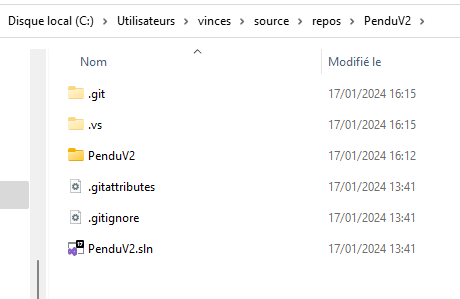


**N’oubliez pas de garder le projet une fois terminé nous le réutiliseront dans le TP suivant**

Si vous souhaitez sauvegarder votre projet dans votre espace personnel ou sur une clé USB il faut copier l’ensemble du projet.



Dans ce dossier vous devez avoir l’arborescence suivante (peut être sans GIT)



SI VOUS NE COPIER PAS L’ENSEMBLE DU PROJET, VOUS NE POURREZ PAS CONTINUER A TRAVAILLER AVEC CE PROJET.