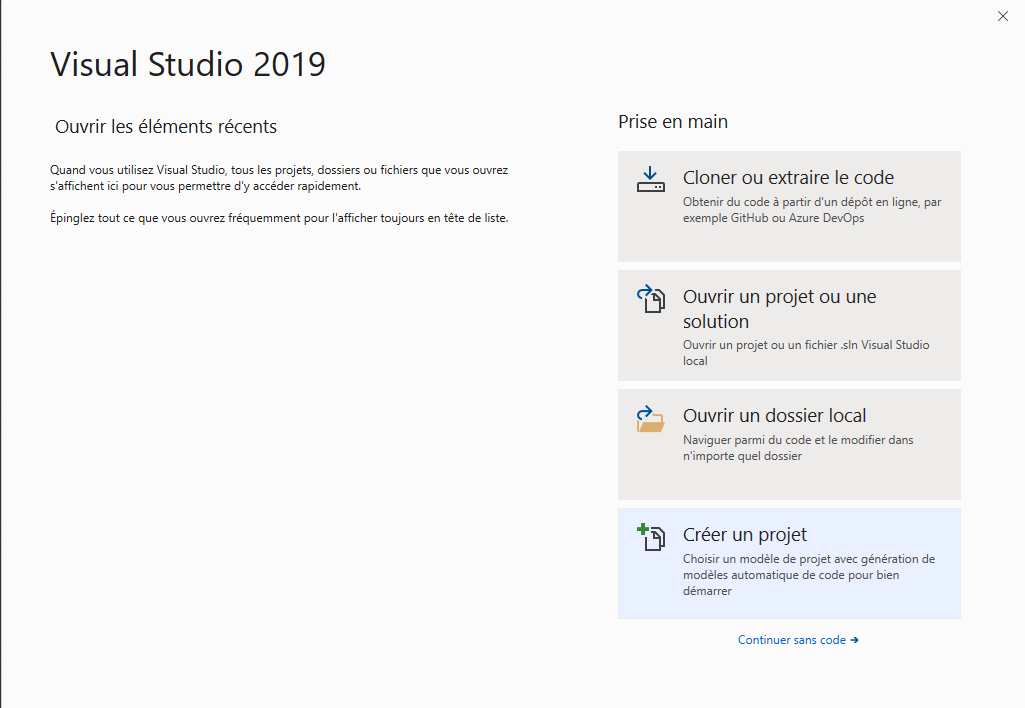
Prise en main de Visual Studio et découverte de C#

## Partie 1 : Prise en main de différents éléments

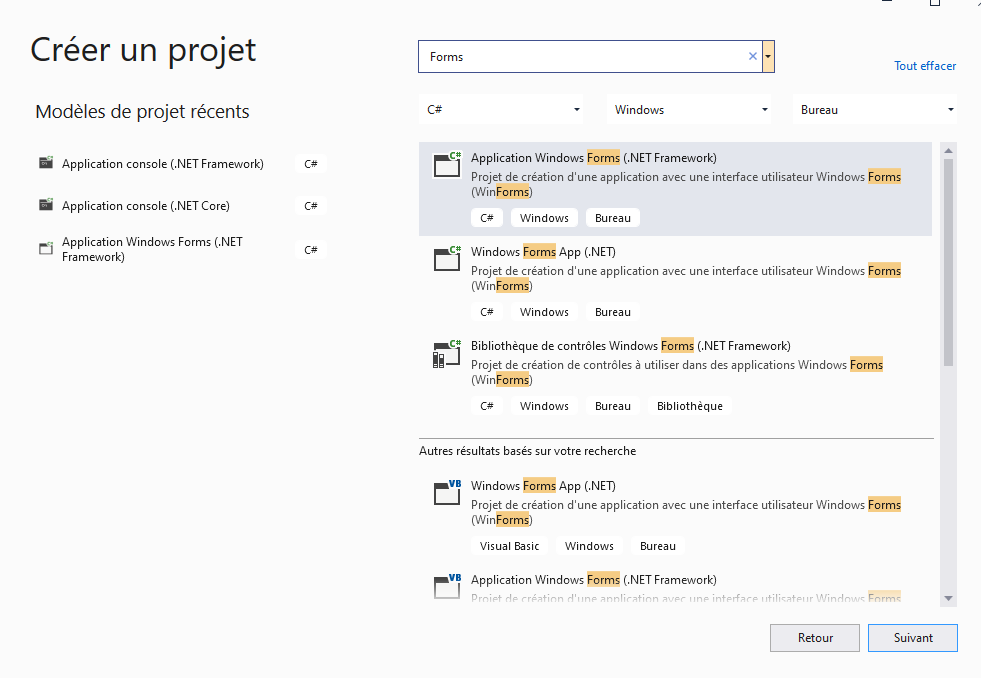
### Phase 1 : Création du projet

Vous devez alors avoir l’écran suivant : 

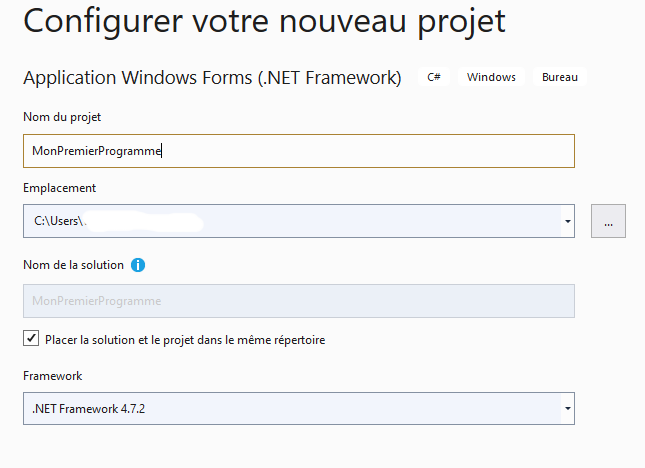
Pour créer votre application, vous devez créer un « projet » :

- 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).

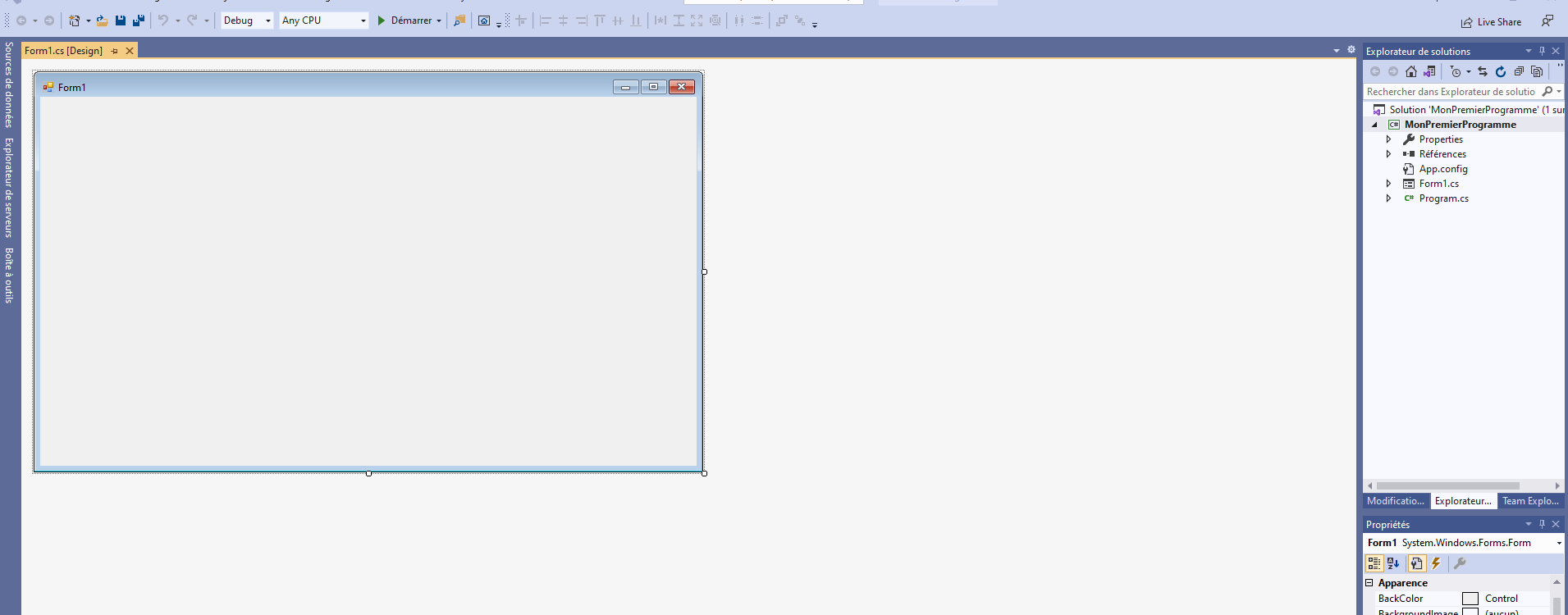


Selon la version de Visual Studio, il est possible que certains types de projets n’apparaissent pas. Vous utiliserez principalement les types qui suivent :

- 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

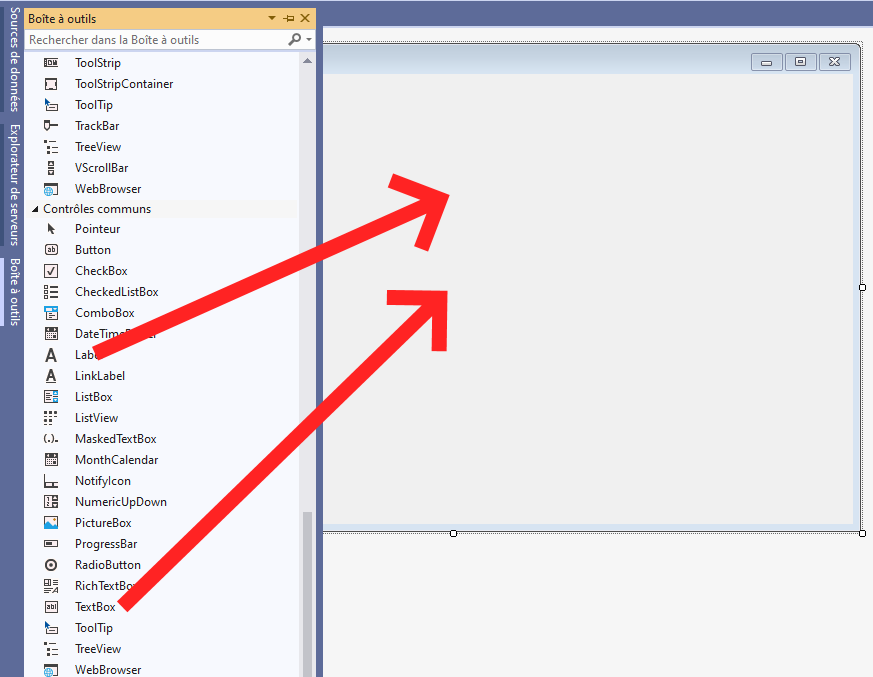
Pour enregistrer le projet pour la première fois, il suffit d’utiliser la démarche traditionnelle sous Windows (menu Fichier – Enregistrer Tout) ou utiliser le raccourcis clavier CTRL + la touche « S ».

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

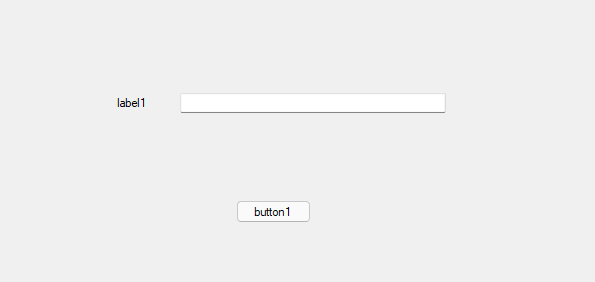
Pour insérer un contrôle, il faut ouvrir la boite à outil. C’est la que tout les contrôles que vous utiliserez seront présents. (Il faut être sur une vue pour que ce menu apparaisse).

### 

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.



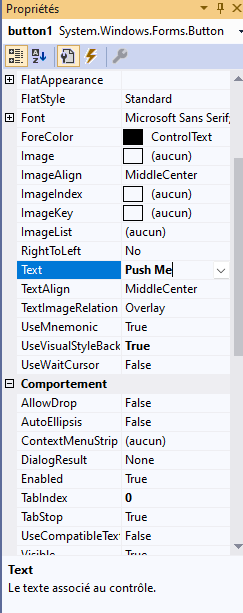
Une fois réalisé, essayez d’avoir le résultat suivant :



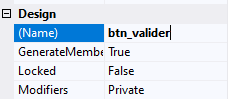
### Phase 4 : Personnalisation des contrôles positionné dans la fenêtre

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

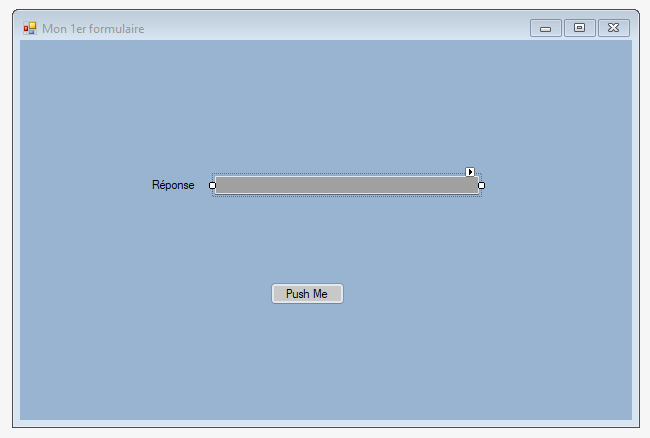


Changer le nom du bouton « button1 » en « btn\_valider »



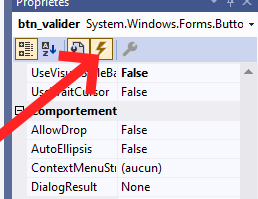
Changer le nom du textbox en « txt\_afficher » et le nom du label en « lbl\_rep »

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.

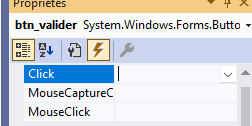


### Phase 5 : Mise en place de code dans l’application

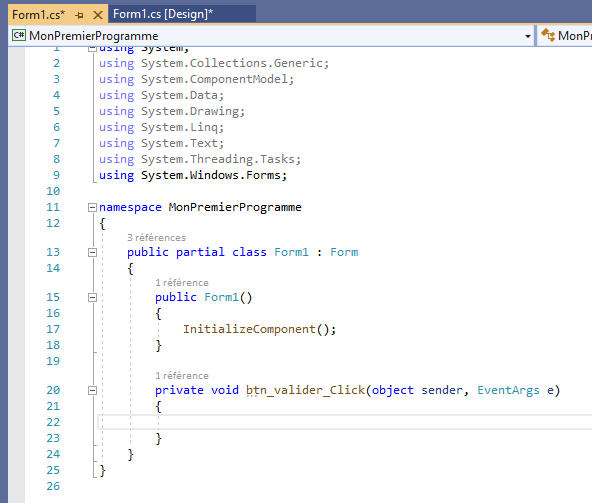
Pour afficher un message lors de l’appuis du bouton, il faut cliquer sur le bouton et cliquer sur le bouton évènement dans les propriétés.



Double cliquez sur le champ à coté de Click

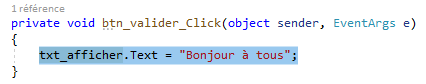


L’écran suivant devrait apparaitre



Insérez le code suivant :

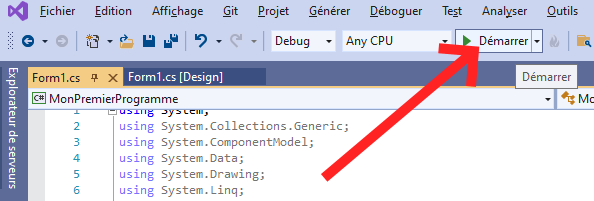
txt\_afficher.Text = "Bonjour à tous"; //Bonjour le commentaire



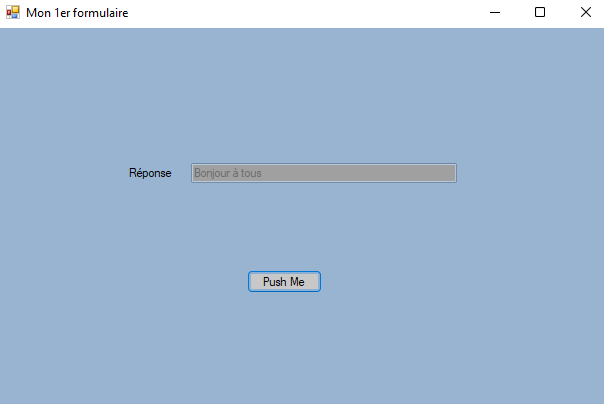
Ici nous utilisons une action (l’appuis sur le bouton) pour que le code s’exécute.

### Phase 6 : Test de l’application

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

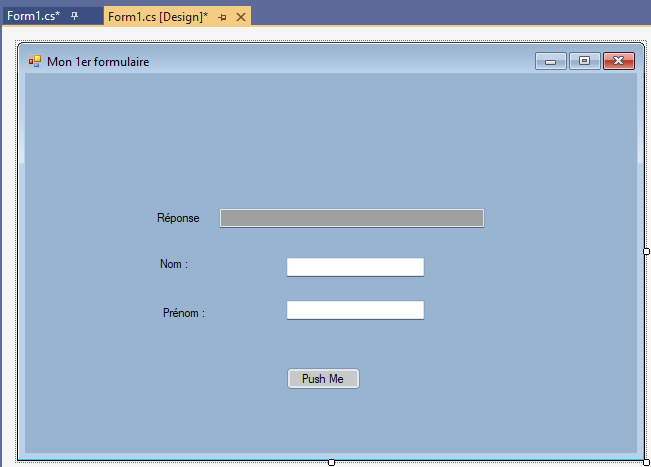


L’écran suivant devrais s’afficher et lors du clic sur le bouton le message devrait s’afficher dans la textbox.



### 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 :

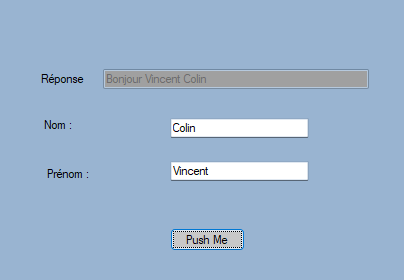
string result = "Bonjour ";

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 != "")

{

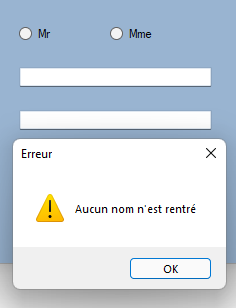
// votre code

}

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

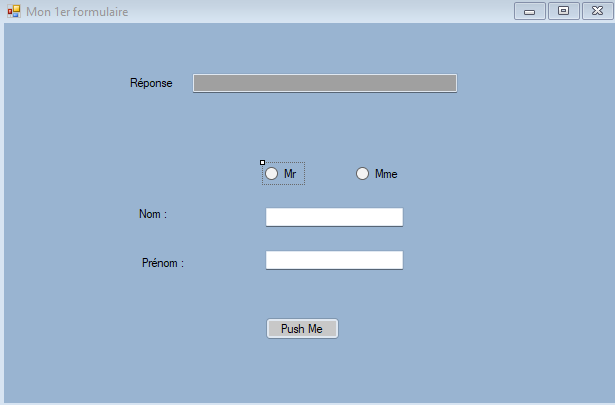
Pour afficher une boite de message à l’utilisateur, on utilise le code suivant : MessageBox.Show("Aucun nom n'est rentré", "Erreur",MessageBoxButtons.OK,MessageBoxIcon.Warning);



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

### Phase 10 : Utilisation de radiobutton

Reproduire l’affichage suivant en n’oubliant pas de renommer les radiobutton



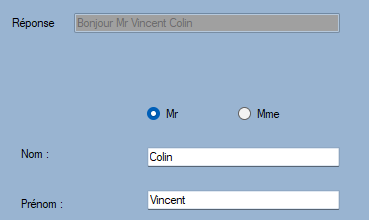
Pour afficher Mr ou Mme utiliser le code suivant :

if (rb\_mr.Checked)

{

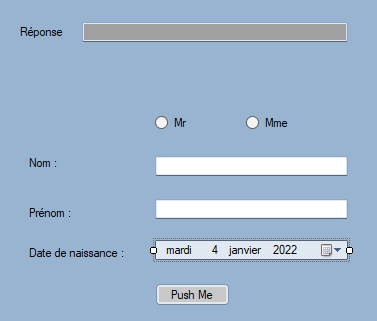
}

Attention, si aucune des deux cases n’est coché alors Mr ou Mme ne doit pas s’afficher



### Phase 11 : Utilisation d’un DateTimePicker

Reproduire l’affichage suivant :



Puis lorsque l’utilisateur saisie sa date de naissance, afficher l’âge de la personne et dire depuis combien d’année il est majeur ou combien d’année il lui reste avant d’être majeur.

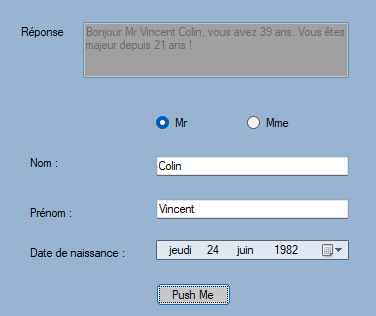
Pour calculer l’âge de la personne, on utilise le code suivant :

TimeSpan calculage;

Int32 age = 0;

calculage = DateTime.Now.Subtract(dtp\_datenaiss.Value);

age = calculage.Days / 365;



### Phase 12 : Utilisation d’une checbox pour afficher ou cacher des composants

Reproduire l’affichage suivant :

### 

Lorsque la case civilité est coché, les radio bouton Mr et Mme doivent s’afficher et lorsque la case civilité est décoché, les radio bouton doivent disparaitre.

Pour cela il faut utiliser l’événement checkchange de la case à cocher.

Pour savoir si l’état de la case à coché on utilise la propriété « Checked » de la case à cocher.

Pour rendre visible un composant, on utilise le code suivant : <Nom du radio buton>.visible = true ;

Attention lorsque l’application est lancée, les radio buton sont visibles alors que la case à coché ne l’est pas. Changer la propriété de chaque radio buton pour qu’ils soient invisible au démarrage de l’application.

Testez l’application et constatez le bon fonctionnement de la case à cocher.

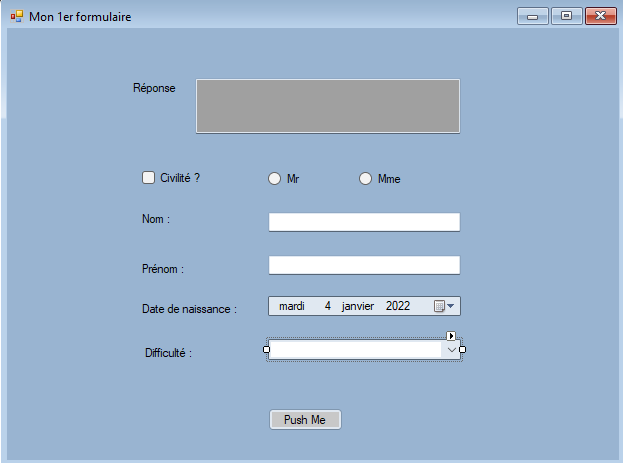
Complétez l’ensemble des informations et testez si le message fonctionne bien

Décochez la case civilité et vérifiez que le message est toujours bon.

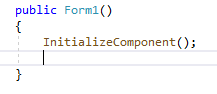
Si le message n’est pas bon tentez de corriger le problème.

### Phase 13 : Utilisation d’un combobox

Reproduire l’affichage suivant :

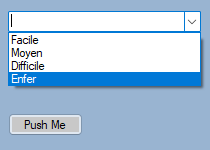


Pour charger un combobox au lancement de l’écran il faut ajouter son code ici :



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>

Gardez bien ce projet, nous nous en resservirons plus tard !!

## Partie 2 : Reverso et boucle

Vous devez réaliser une application permettant d’afficher le texte saisi par l’utilisateur à l’envers

