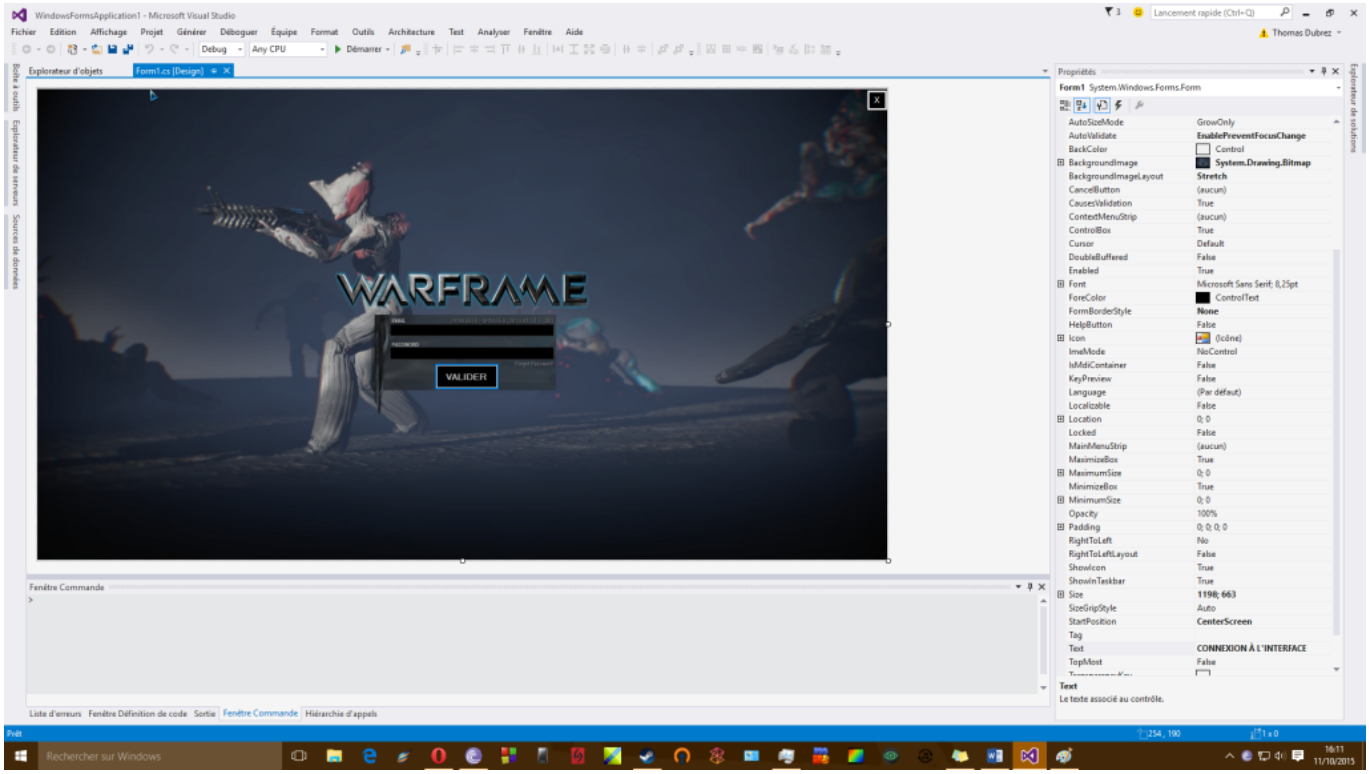
Identification en C# et Windows Form

http://www.freedom-substitute.fr/programmation/identification-c-windows-form/

Apprenez à créer un système d'identification en C# en plus d'en apprendre plus sur les fonctions "KeyPress" et les fonctions avancées des "MessageBox" !

Pour ce cours, nous utiliserons un programme qui se lance en plein écran à la façon du célèbre jeu "Warframe". Cela permet d'avoir quelque chose de plus familier selon vos usages, alors n'hésitez pas à essayer de recréer le système d'identification de votre programme ou jeu favori !

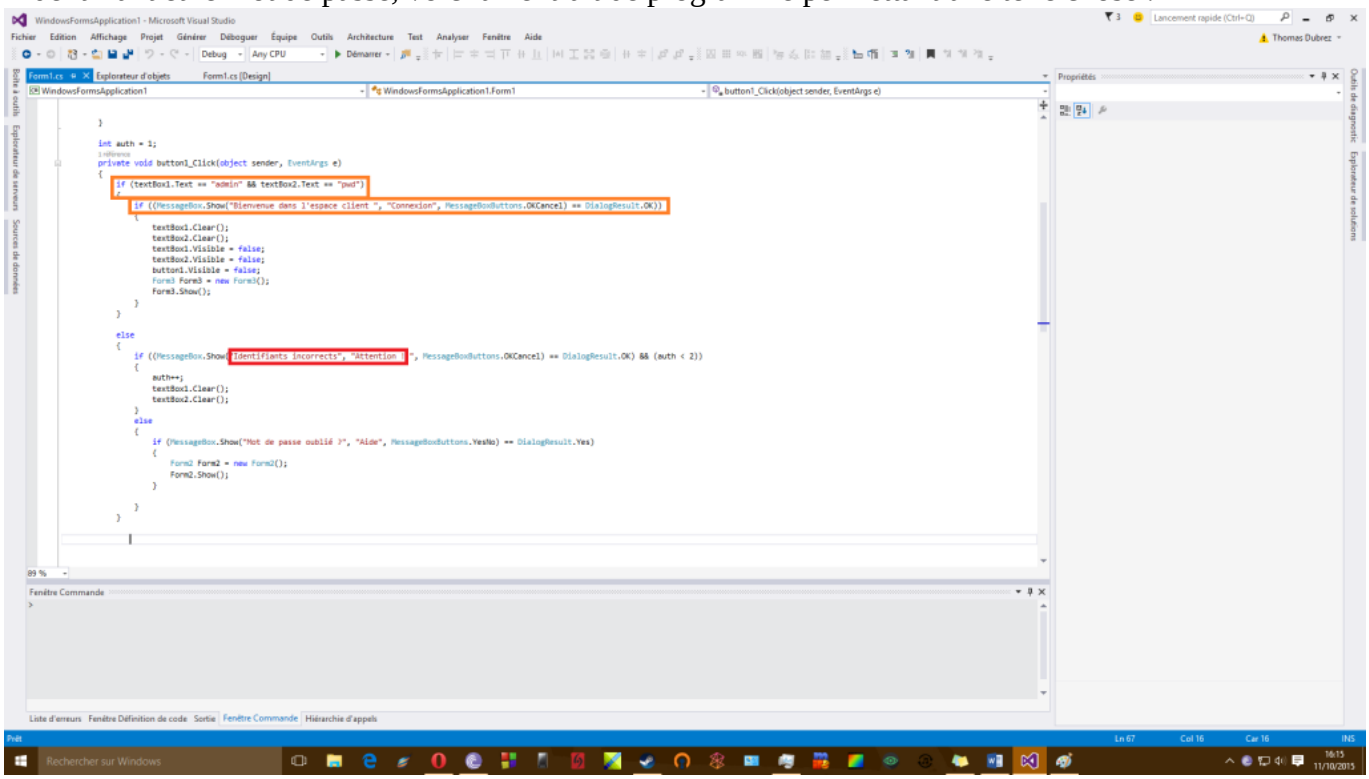
Comment marche le login en C# ?



Dans cette fenêtre, nous voyons l'interface de login proposée à l'utilisateur.

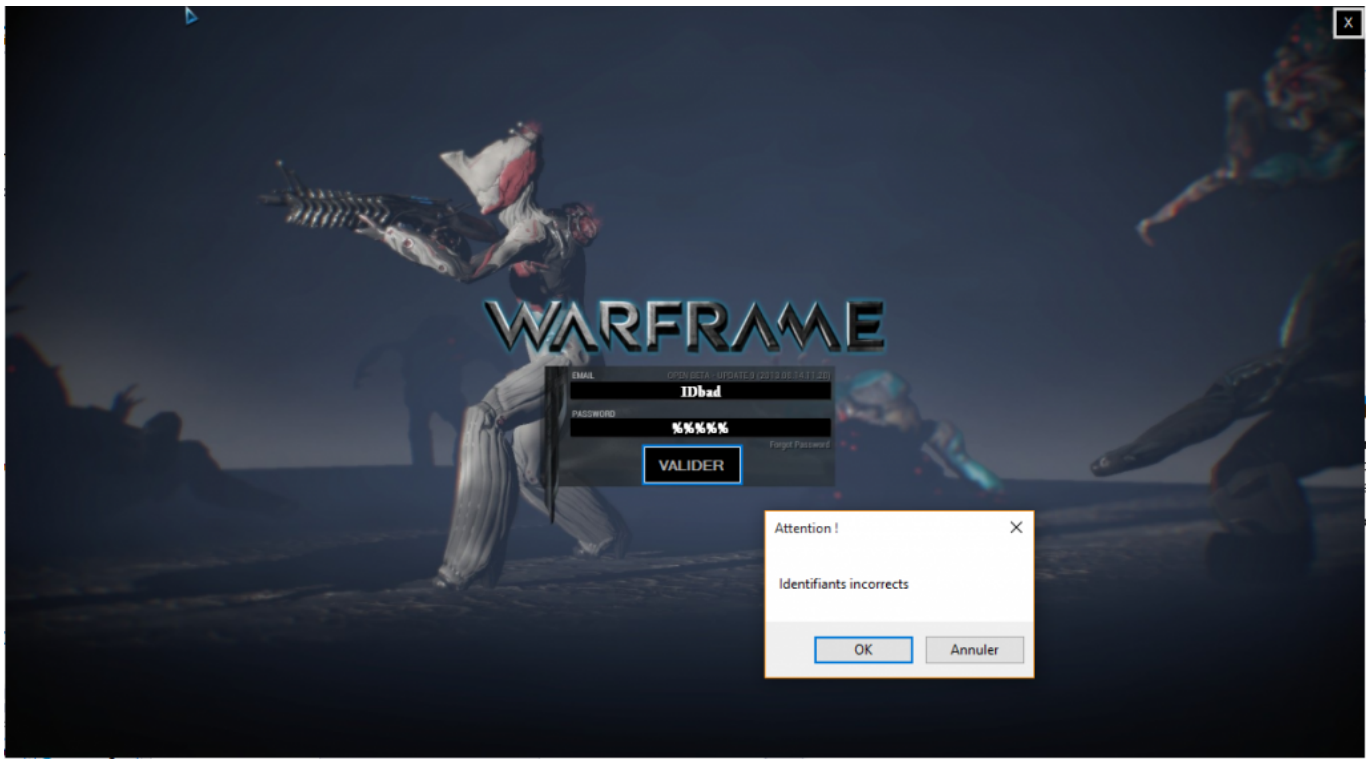
Pour que l'utilisateur puisse se connecter, il doit entrer ses identifiants dans deux champs, « Identifiant » & « Mot de passe ».

Ces champs sont utilisés pour reconnaître une personne à l'aide d'un programme qui comparera l'identifiant et le mot de passe, voici un extrait de programme permettant une telle chose :

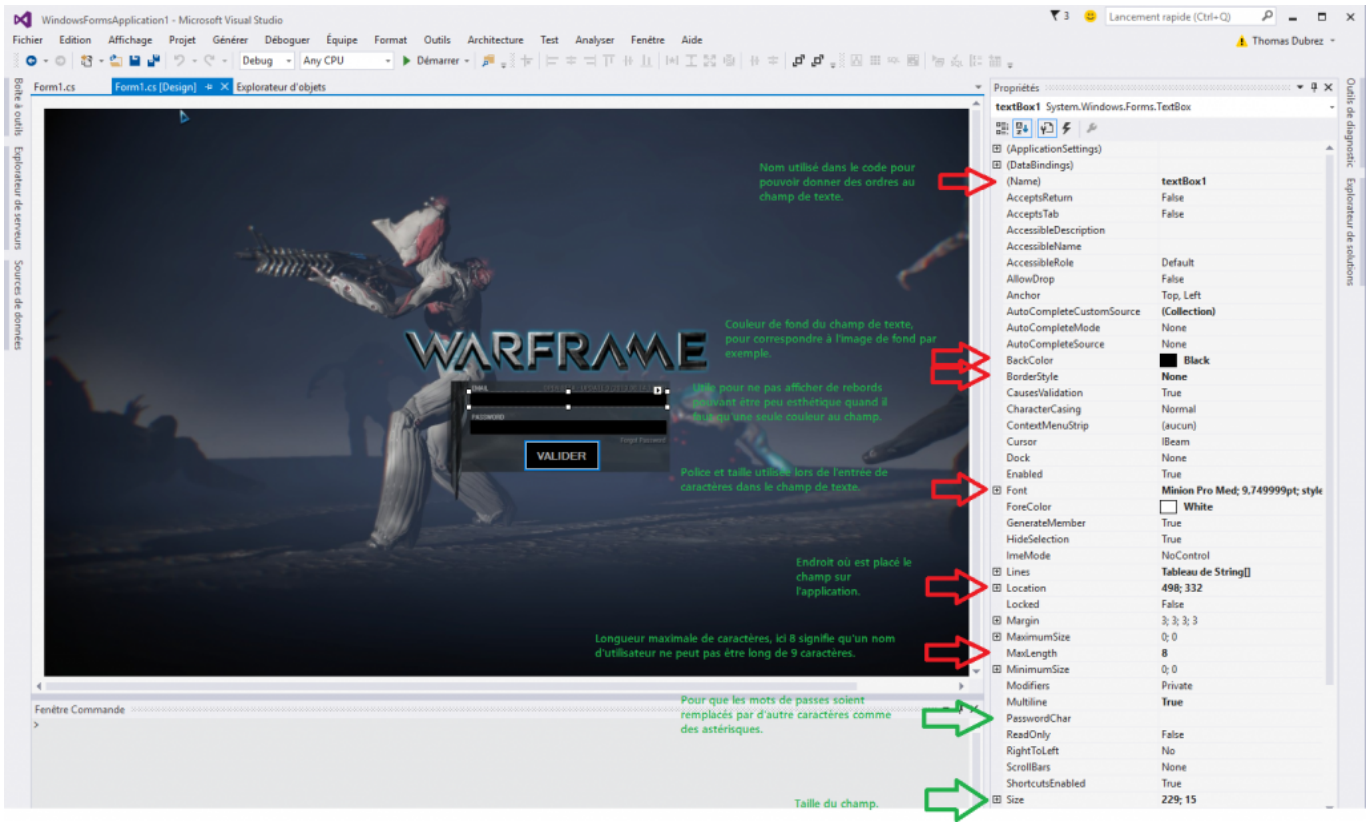


L'encadré orange montre le code permettant une connexion, cela permet de vérifier que l'identifiant et le mot de passe entrés correspondent à ce qui est dans la base de donnée auparavant (ici on n'en utilise pas, c'est à titre d'exemple).

Sinon, l’encadré rouge indiquera à l’utilisateur que ce qu’il a entré ne correspond pas à un identifiant & mot de passe reconnu : il y a échec d’authentification.



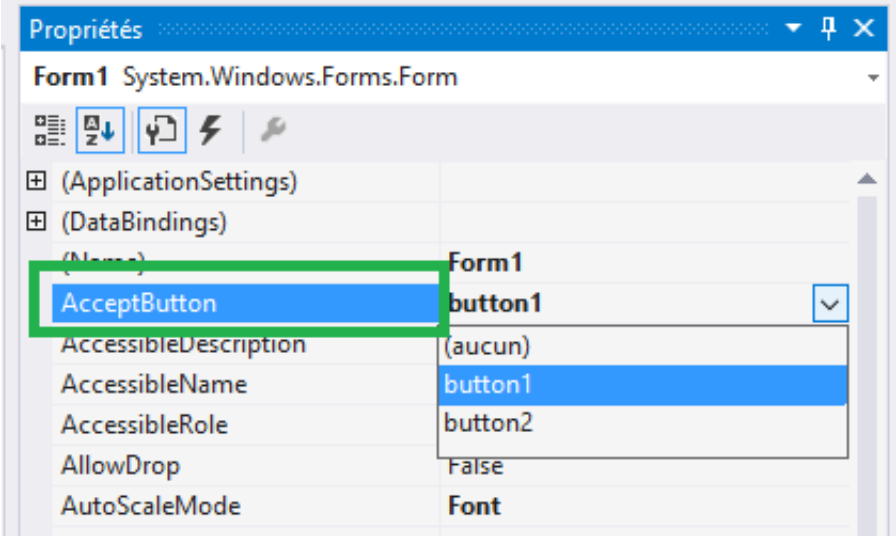
Proprieté des zones de saisies



Cliquez pour agrandir

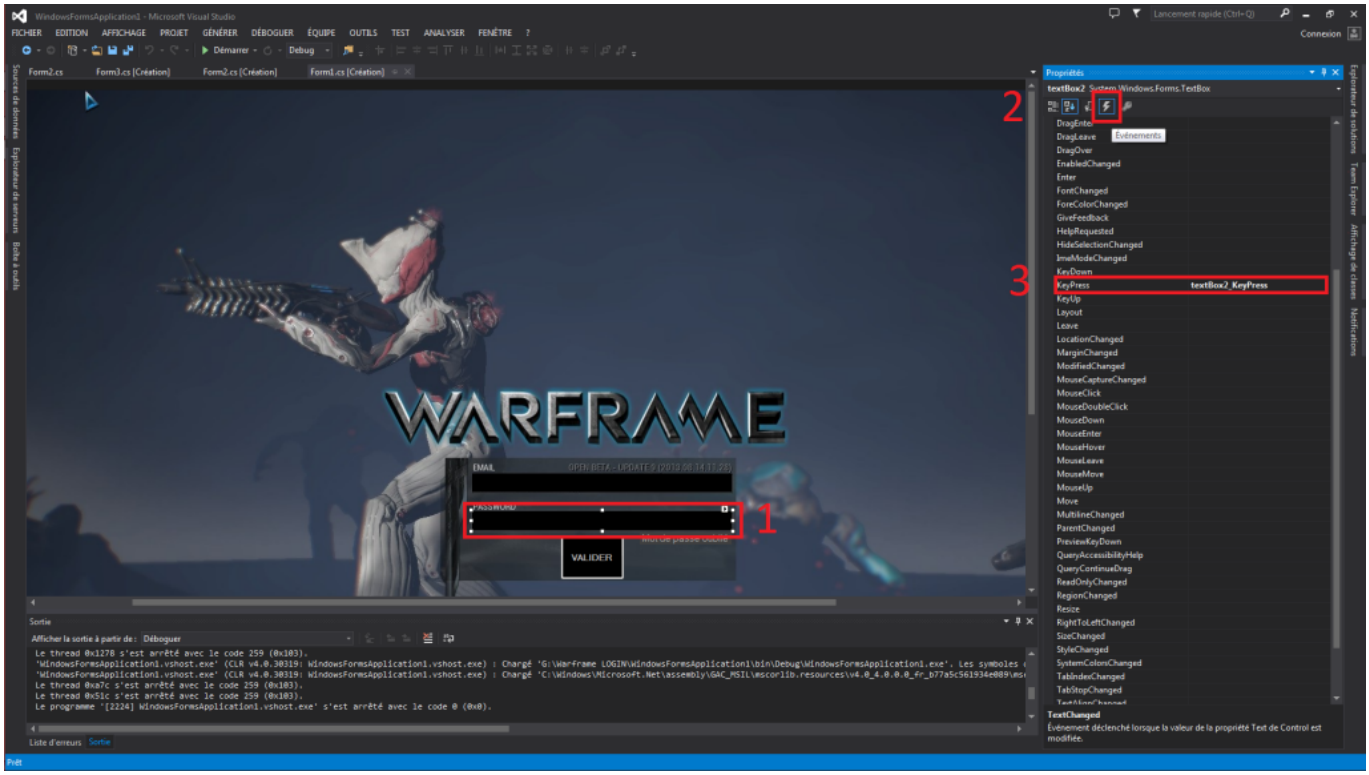
Valider à l’aide de la touche entrée

Première solution



Il existe déjà cette option dans Visual Studio, il suffit de sélectionner l'option « AcceptButton » en ayant sélectionné Form1 (la fenêtre principale), une fois dedans, plusieurs choix sont possible, il suffit de choisir sur quel bouton notre entrée doit se répercuter, ici on veut utiliser le bouton « VALIDER » qui correspond au « button1 ».

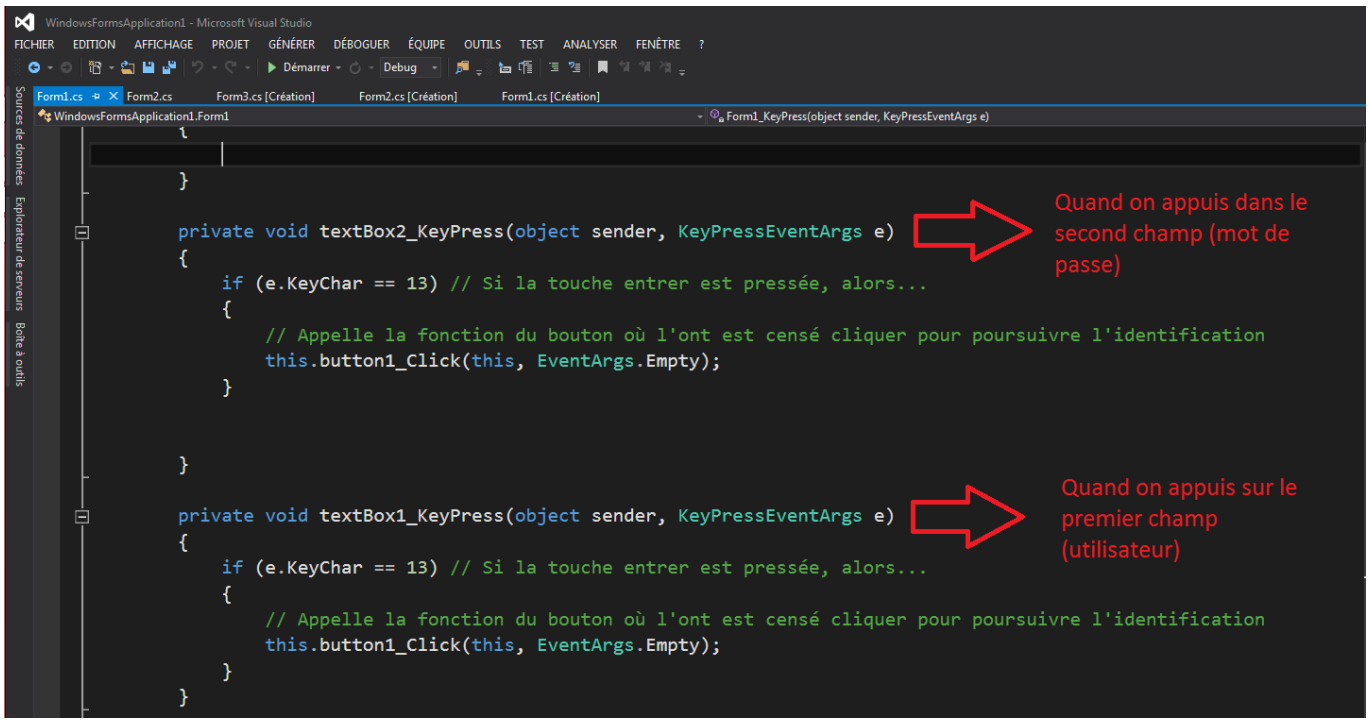
Deuxième solution



1. Tout d'abord, il faut sélectionner le champ où sera placé le code permettant la détection d'un appui sur une touche (1).
2. Ensuite, dans la fenêtre des propriétés à droite il faut appuyer sur l'éclair qui permet de gérer les

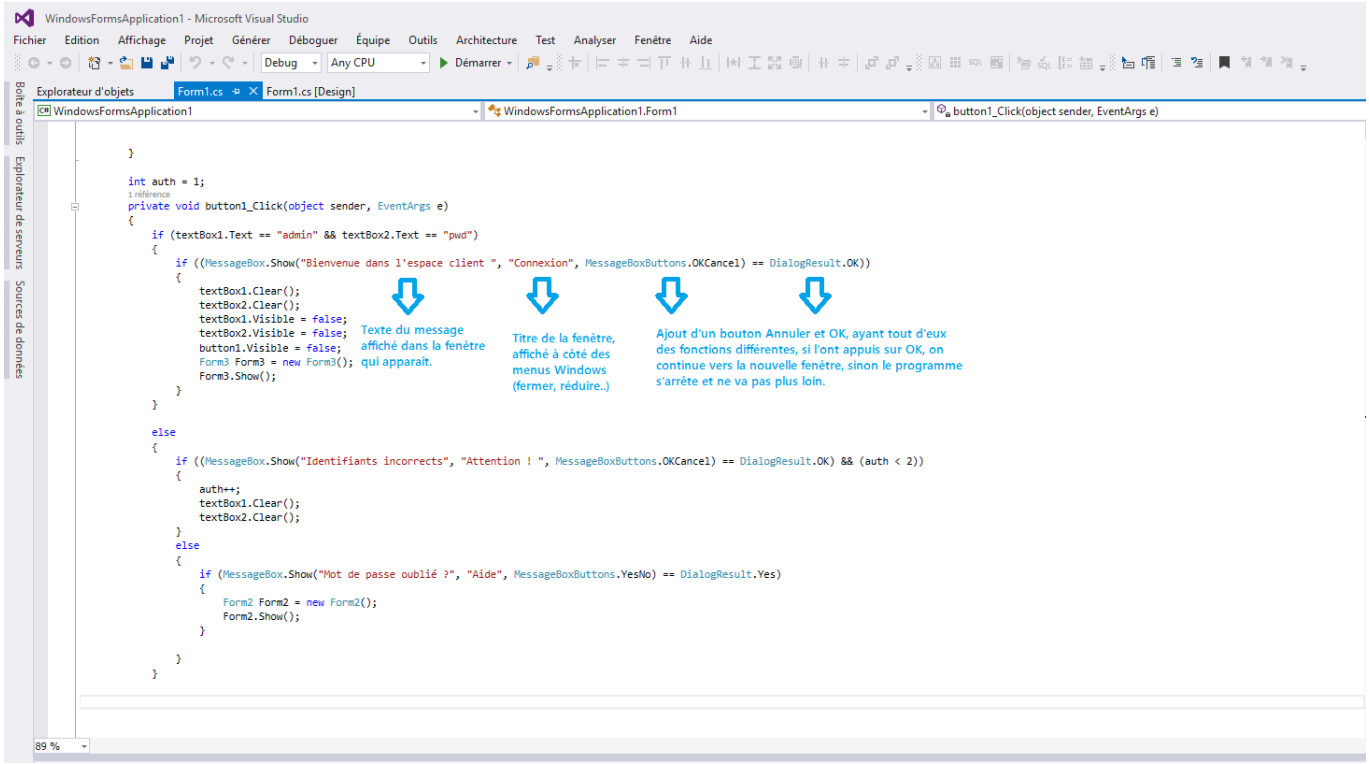
évènements (2).

3. Enfin, on appuis sur l'onglet « KeyPress » et on pourra enfin commencer à taper le code (3).



Voici un exemple de méthode permettant d'identifier un appui sur la touche entrée et de renvoyer par la suite sur la fonction permettant l'identification.

La MessageBox en détails



Pour ajouter une petite icône sur la « MessageBox » qui s'affiche, il suffis de rajouter :

`MessageBoxIcon.[Parametre]`

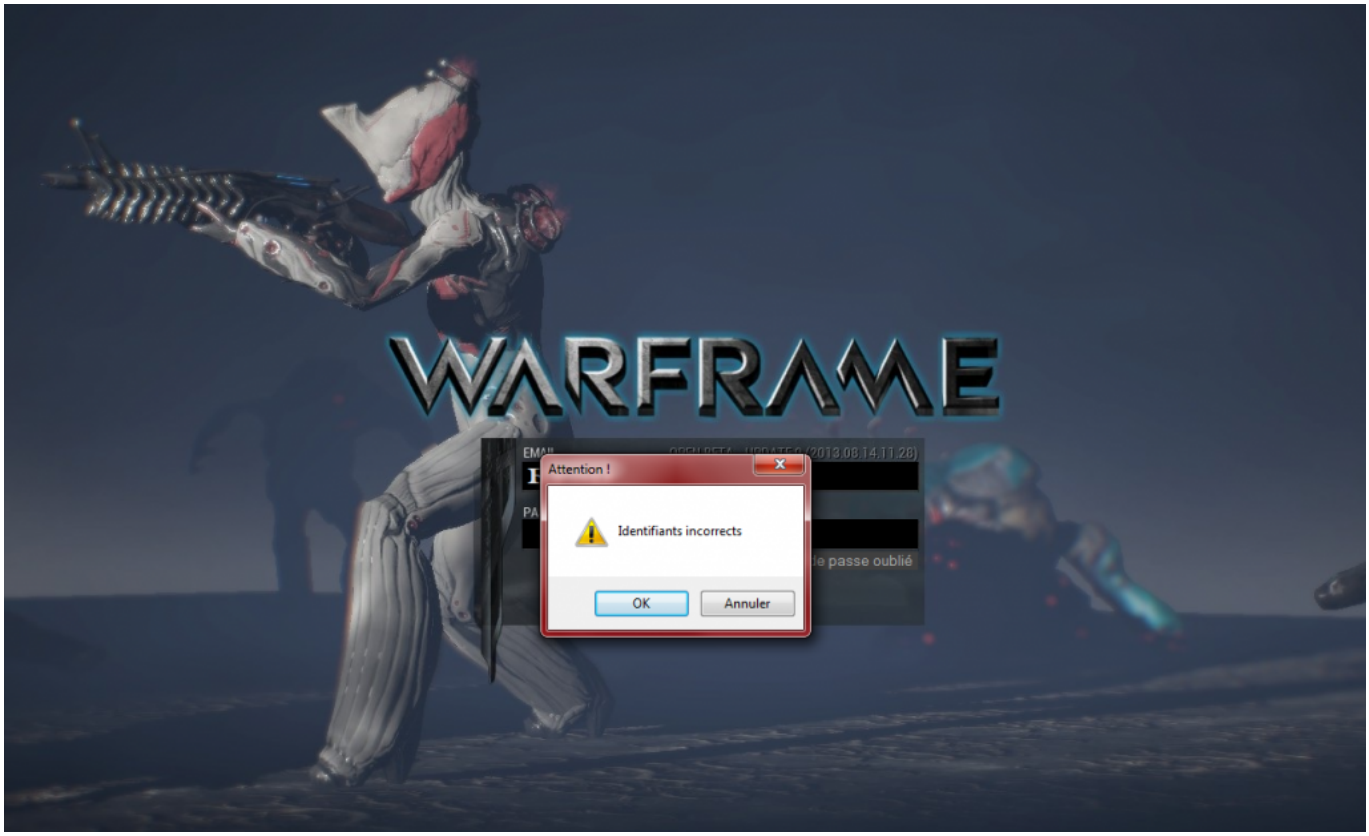
```
else // sinon
{
    // On préviens que les identifiants sont mauvais
    if ((MessageBox.Show("Identifiants incorrects", "Attention ! ",  MessageBoxButtons.OKCancel, MessageBoxIcon.Warning) == DialogResult.OK) && (auth < 2))
    {

        auth++; // Incrémente la variable qui calcule le nombre de tentatives
        textBox1.Clear(); // Efface le champs identifiant pour recommencer sans avoir à effacer
        textBox2.Clear(); // Efface le champs mot de passe pour recommencer sans avoir à effacer
    }
    else
    {   // Si les 2 esssais sont utilisée une fenêtre demande si l'ont veut réinitialiser le mot de passe
        if (MessageBox.Show("Mot de passe oublié ?", "Aide", MessageBoxButtons.YesNo,MessageBoxIcon.Question) == DialogResult.Yes)
        {
            // Appelle la deuxième fenêtre où l'ont peut réinitialiser le mot de passe
            Form2 Form2 = new Form2();
            Form2.Show();
        }
```

Il existe plusieurs types d'icône :

- Pour un point d'interrogation c'est « MessageBoxIcon.Question »
- Pour un point d'exclamation c'est « MessageBoxIcone.Warning »
- …

Exemple d'icône « Question » :

Exemple d’icône « Warning » :



Refuser la validation si des champs sont vides

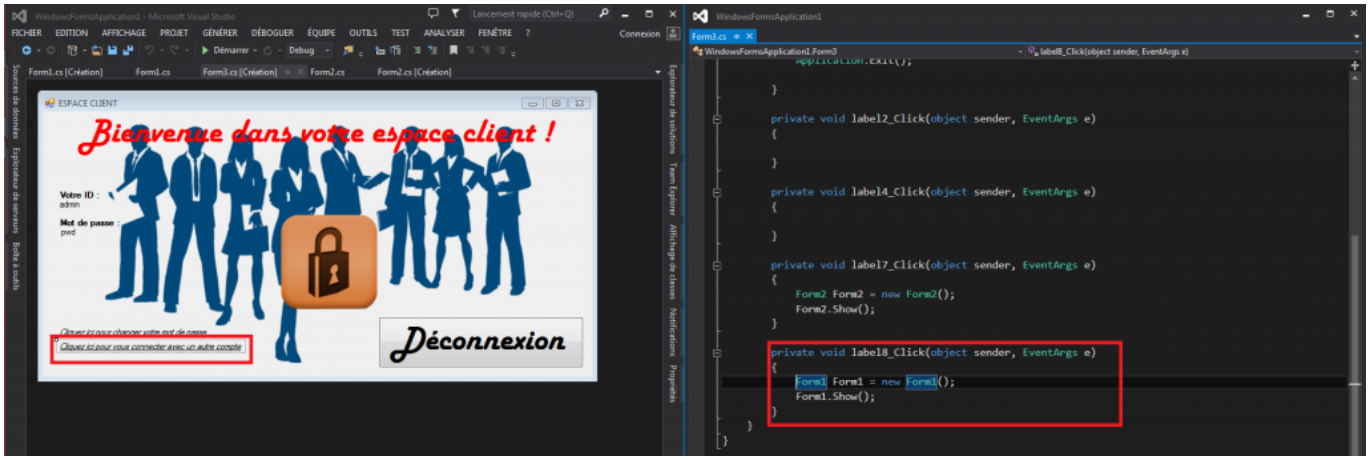
Il faut donc ajouter un code simple :

La première condition permet de vérifier si les champs sont différents de vide « = ! ‘’ ‘’ », et donc s’ils sont vides, ont envois un message comme quoi c’est le cas.

```
sApplication1.Form2                                    button1_Click(object sender, EventArgs e)

    private void button1_Click(object sender, EventArgs e)
    {
        if (textBox1.Text != "" && textBox2.Text !="" && textBox3.Text != "")
        {
            MessageBox.Show("Un email contenant un lien de réinitialisation du mot de passe va vous été envoyé.");
            Application.Exit();
        }
        else
        {
            MessageBox.Show("Les champs sont vides !");
        }
    }
```

Si ces étapes simples ne sont pas effectuées, alors il sera possible de dire à l’utilisateur qu’un email lui ait été envoyé SANS préciser d’adresse mail.

Éviter de fermer le programme à la fermeture d’une Form



Il suffit simplement d'entrer ces deux lignes et de remplacer par le nom de la Form, cela évite de relancer le programme plusieurs fois pour tester.

Télécharger l'exemple

Télécharger

FREEDOM SUBSTITUTE