Mise en place de l'auto complétion

1. Présentation

L'auto complétion consiste à proposer une liste de valeurs possibles correspondant à la saisie de l'utilisateur afin d'aider l'utilisateur dans ses choix.
Elle est directement intégrée sur les composants textBox et combobox, il suffit d'initialiser certaines propriétés de ces composants.
L'intérêt est double : d'éviter les erreurs et faciliter la saisie
Pour une zone de liste de sélectionner la première valeur correspondant aux caractères déjà saisis.
Sans ce système une zone de liste sélectionne un élément uniquement sur la valeur de son premier caractère.

2. Les principales propriétés à mettre en œuvre

Propriété	Rôle
AutoCompleteMode	La valeur **None** désactive l'autocomplétion pour ce contrôle. La valeur **Append** active la complétion qui écrit automatiquement la fin du texte avec la valeur la plus probable. Il est alors possible avec les flèches haut et bas de naviguer parmi les saisies possibles. La valeur **Suggest** active une liste déroulante n'affichant que les éléments corrects (dont le début correspond au texte actuellement saisi dans la TextBox). La valeur **SuggestAppend** qui combine l'effet des valeurs Append et Suggest.
AutoCompleteSource	La valeur **CustomSource** permet de définir sa propre source de données. La valeur **ListItems** qui s'adresse à un combobox permet de définir la source du combobox comme source de données de l'autocomplétion. Le filtre se réalisant sur chaque caractère saisi, cela évite d'afficher toutes les valeurs de la source.
AutoCompleteCustomSource	Définit l'objet AutoCompleteStringCollection à utiliser pour l'autocomplétion. Cet objet contient un tableau de chaines de aractères qui servira pour l'autocomplétion. Pour alimenter ce tableau il faut utiliser la méthode AddRange([] string) de cet objet

Ces propriétés peuvent être initialisées dans l'IDE ou dans le code de l'application :

Exemple : un champ txtHeure devant afficher des horaires de début de rendez-vous.
On créer une collection de chaine de caractères que l'on passe en paramètre à la méthode AddRange.

```
// paramétrage de l'heure
txtHeure.AutoCompleteMode = AutoCompleteMode.SuggestAppend;
txtHeure.AutoCompleteSource = AutoCompleteSource.CustomSource;
var source = new AutoCompleteStringCollection();
source.AddRange(new string[] {"08:00","08:30","09:00","09:30","10:00","10:30","11:00","11:30",
"12:00","12:30","13:00","13:30", "14:00","14:30","15:00","15:30","16:00","16:30",
"17:00","17:30","18:00","18:30","19:00"});
txtHeure.AutoCompleteCustomSource = source;
```

Exemple : Autocomplétion sur un champ txtVille. La source des données (le nom des villes) est obtenue à partir d'une collection d'objets Ville possédant une propriété Nom

```
txtVille.AutoCompleteMode = AutoCompleteMode.SuggestAppend;
txtVille.AutoCompleteSource = AutoCompleteSource.CustomSource;
var source = new AutoCompleteStringCollection();
foreach (Ville uneVille in LesVilles)
    source.Add(uneVille.Nom);
txtVille.AutoCompleteCustomSource = source;
```

Exemple : Autocomplétion sur une zone de liste qui affiche une liste de praticien (nom du praticien).
Cela permet de filtrer les praticiens sur les lettres saisies ce qui ne se fait pas sur une zone de liste sans système d'autocomplétion (chaque lettre saisie sélectionne le premier élément dont la première lettre correspond).

```
cbxPraticien.AutoCompleteMode = AutoCompleteMode.SuggestAppend;
cbxPraticien.AutoCompleteSource = AutoCompleteSource.ListItems;
```

Exemple : Autocomplétion sur un champ txtMedicament. La source des données (le nom des médicament) est obtenue à partir d'une collection d'objets Medicament (LesMedicaments) possédant une propriété Nom

```
txtMedicament.AutoCompleteMode = AutoCompleteMode.SuggestAppend;
txtMedicament.AutoCompleteSource = AutoCompleteSource.CustomSource;
AutoCompleteStringCollection source = new AutoCompleteStringCollection();
foreach (Medicament unMedicament in Globale.db.LesMedicaments)
      source.Add(unMedicament.Nom);
txtMedicament.AutoCompleteCustomSource = source;
```

Pour récupérer l'id du médicament il suffit d'utiliser la méthode Find sur la collection LesMedicament

```
Medicament leMedicament = Globale.db.LesMedicaments.Find(x => x.Nom == txtMedicament.Text);
```

Webographie https://lgmorand.developpez.com/dotnet/autocompletion/