**Manipulations de cours pour créer une interface graphique**

Visual Studio bénéficie à la fois d'un générateur de code et d'un éditeur graphique pour faciliter la conception d'Interfaces Graphiques (IG). A chaque instant, vous pouvez aussi bien travailler avec l'éditeur graphique qu'en modifiant directement le code. En noir, les principe de la création d'une IG avec Visual Studio, en bleu, un exemple.

- En premier lieu créez un projet C# / WinForms
  - Éditez le contenu de program.cs
  - Éditez le contenu de Form1.cs et allez voir le contenu de InitializeComponent() Nota : il faut « ouvrir » la zone « code généré par le concepteur Windows Form »
- Utilisez la boîte à outils pour composer votre IG dans la fenêtre Form1
  - Ajoutez un Button (onglet boîte à outils)
  - Ajouter un Label
  - Remarquez que les manipulations sous l'éditeur graphique correspondent à des instances des classes de la boîte à outils graphiques (voir fichier Form1.Designer.cs). Ces classes permettent de créer des objets appelés **Widget** (ex : Button, CheckBox, RadioButton ...).
- Pour chaque objet graphique, vous pouvez éditer ses propriétés. Ceux sont pour l'objet les propriétés telles que nous les avons vues dans le cours sur la programmation objet en C#. Elles correspondent bien souvent à un aspect graphique. Changez la couleur du bouton. Changez le texte du bouton : d'abord en allant sur la propriété puis en allant dans la méthode « InitializeComponent » de Form1.Designer.cs.

A ce stade, votre interface graphique a l'allure que vous souhaitez mais ne "fait rien".

- Chaque **événement** (issu bien souvent d'une interaction avec un objet graphique) est associé à l'appel d'une méthode (dite événementielle) dont le code peut-être modifié pour programmer ainsi le comportement de votre application. Par exemple, la méthode Button1_Click() est définie automatiquement pour contenir le code qui va être exécuté quand l'utilisateur cliquera (événement Click) sur le bouton Button1.
  - Pour trouver toutes les méthodes événementielles définies pour un objet graphique, il vous suffit d'éditer l'onglet avec un symbole "éclair" dans la fenêtre propriété.
  - Pour que la méthode associée à cet événement soit créée, il faut double-cliquer sur l'événement

  Allez dans la fenêtre Form1.cs[Design], sélectionnez le bouton, dans l'onglet « propriétés », « éclair » double-cliquez sur Action/ Click. Cela créé une méthode button1_MouseClick dans Form1.cs. Modifiez cette méthode pour qu'elle affiche « coucou » dans le label. Testez votre programme.

Lorsque l'on créé un projet de type WinForm, VisualStudio génère plusieurs fichiers, et génère le code associé à l'IG. A priori, **vous ne devez ajouter du code que dans le fichier Form1.cs.**

- **Program.cs** contient le main qui déclare, initialise et lance (run) une instance de Form1 la fenêtre de l'IG
- **Form1.cs** contient le constructeur et appelle InitializeComponets(). C'est dans Form1.cs que vous devez déclarer les objets métier s'il y en a (i.e. Instances de classes utiles aux calculs qui s'affichent dans l'IG). C'est aussi dans Form1.cs que VisualStudio génère les fonctions associées aux événements. Il vous faut donc compléter le corps des méthodes événementielle dans Form1.cs.
- **Form1.Designer.cs** contient la méthode InitializeComponents et la déclaration des objets graphiques (i.e. les widgets contenus dans la fenêtre). A chaque fois que vous ajoutez un composant graphique dans l'éditeur graphique, ou que vous changez une de ses propriétés, cela modifie du code dans Form1.Designer.cs.
- **Form1.cs [Design]** est la visualisation graphique de Form1 ; quand vous faite une modification dans cette fenêtre, Visual Studio génère automatiquement du code qui est inséré dans Form1.cs et Form1.Designer.cs.

Notez que les fichiers Form1.cs et Form1.Designer.cs contiennent 2 parties de la classe Form1 qui est « partial ».