TP – C#

Prise en main : interface graphique, animation

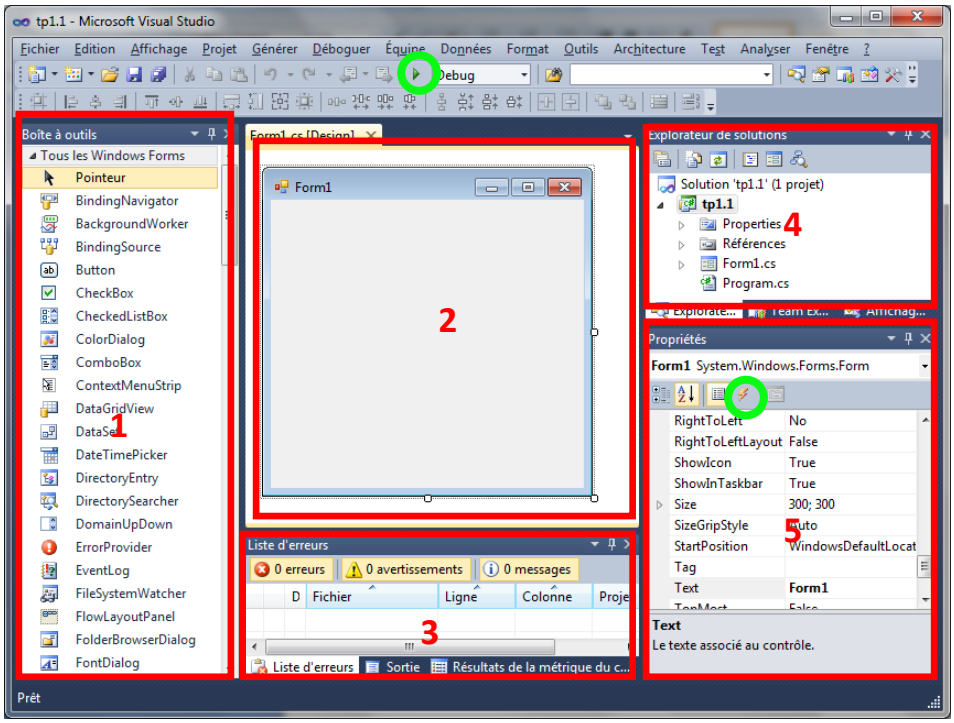
1. Hello World !

Description : Vous allez construire une application graphique dotée d'un unique bouton qui affiche le message « Hello World ! » lorsque l'on clique dessus.

Contenu : Prise en main de Visual Studio, création d'une application Windows Form, création d'évènements.

Instructions :

- Commencez par lancer Visual Studio et créez un projet Windows Form. La fenêtre que vous obtenez ressemble à cela :



1) L'ensemble des composants disponibles que vous pouvez ajouter à votre projet. Il peut s'agir de composants visuels (un bouton) ou non (un timer).

2) La zone de conception, ou vous construisez l'interface graphique et éditez le code source.

3) La zone de débogage : tous les messages de compilateur sont indiqués ici.

4) L'explorateur de solution : tous les fichiers (codes, configurations, ressources, ...) qui composent votre solution.

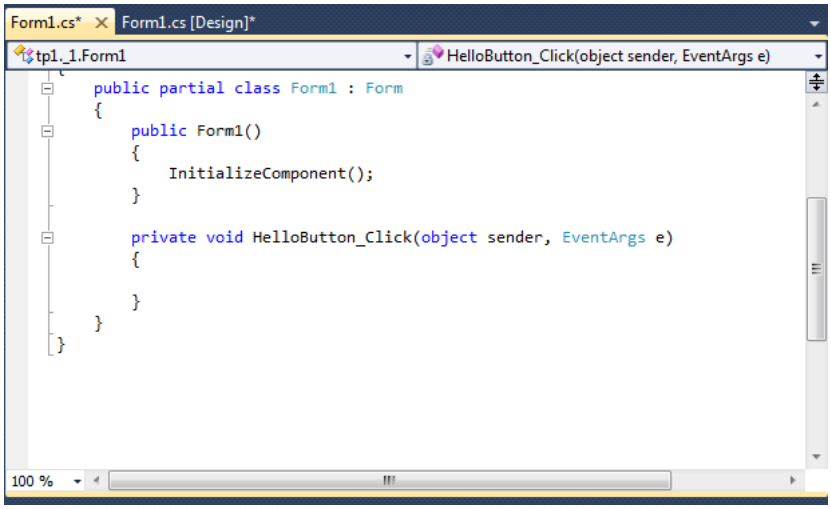
5) Liste des propriétés (nom, texte, taille, couleur, ...) du composant sélectionné.

La flèche verte, au milieu en haut, permet de construire et d'exécuter le projet.

Le petit éclair dans la zone 5 permet d'accéder à tous les évènements (MouseClick, MouseMove, KeyPressed, Paint, ...) gérés par le composant sélectionné et de définir de nouveaux comportements.

En plus des raccourcis habituels des applications Windows (Copier-Coller,...) Visual Studio propose de nombreux raccourcis dont les deux plus importants sont certainement : **Ctrl+Espace** pour la complétion automatique et l'aide en ligne et **Ctrl + .** pour l'ajout automatique des directives using .

- Modifiez les propriétés de l'objet form1 (la fenêtre principale), donnez lui un nouveau nom, une nouvelle couleur de fond, et un nouveau curseur par défaut.
- Ajoutez maintenant un bouton à votre fenêtre. Modifiez son nom et appelez le HelloButton, modifiez également le texte affiché dans le bouton.
- Ajoutez un évènement Click à votre bouton (faites un double cliques sur le bouton ou cherchez l'évènement dans la liste des évènements gérés par l'objet) :



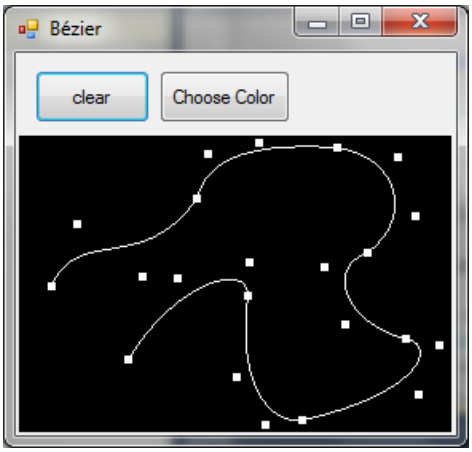
- Ajoutez du code dans la fonction <nom_du_composant>_Click pour afficher le message « Hello World ! » (utilisez la méthode statique Show de la classe MessageBox).
- Vérifiez que tout fonctionne bien !

2. Petit logiciel de dessin de courbes de Bézier

Description : Nous allons construire une petite application permettant de tracer des courbes de Bézier en ajoutant les points de contrôle avec la souris. L'application comprendra en outre deux boutons : un pour tout effacer, l'autre pour choisir la couleur de dessin.

Principe de fonctionnement : Le logiciel va se reposer sur une liste de points, un Panel pour dessiner et les évènements Paint et Click du Panel. La méthode Click aura pour but de rajouter un point dans la liste de points aux coordonnées actuelles de la souris et d'invalider le contenu du Panel (méthode Invalidate). Cette invalidation va provoquer un appel à la méthode Paint du Panel qui va alors dessiner la liste des points entrés par l'utilisateur et les courbes de Bézier correspondantes. Le bouton « effacer » videra la liste des points et invalidera le Panel. Le bouton « Choix d'une couleur » fera apparaitre un dialogue de sélection de couleur et invalidera le Panel.

Contenu : Windows Form, Graphics, Point, Collection List, BufferedGraphics (double buffering)



Instructions :

- Créez un nouveau projet Windows Form.
- Ajoutez : deux boutons (« Clear » et « Choose Color ») et un composant Panel qui servira comme zone de dessin.
- Dessin des points :
  - Commencez par ajouter un champ contenant une liste de Points à la fenêtre principale. Vous utiliserez le conteneur générique List de .net. List représente une liste de taille variable (redimensionnement automatique) du type spécifié. Par exemple pour déclarer une variable locale de type liste de Points, on peut écrire : `List<Point> listeDePoints = new List<Point>()`. Apprenez à vous servir de cet objet en regardant la MSDN library.
  - Ajoutez un évènement Click sur le Panel. Modifiez le code de l'évènement pour ajouter un nouveau point de coordonnées égales à celles de la souris à la liste de points. Appelez ensuite la méthode Invalidate de Panel (pour demander au logiciel de rafraichir l'affichage).
  - Ajoutez un évènement Paint sur le Panel. Cet évènement est appelé à chaque fois qu'il faut redessiner le composant. L'argument PaintEventArgs de la méthode Paint contient un objet Graphics qui permet de faire des dessins (dessiner des lignes, des rectangles, des ellipses, du texte, des courbes, ...). Utilisez la MSDN et trouver comment utiliser cet objet pour dessiner les points stockés dans la liste de points.
  - Vérifiez que tout fonctionne.
- Bouton « Effacer »
  - Modifier l'évènement Click du bouton Clear de manière à vider la liste de points et invalider le Panel.
- Bouton « Choose Color » :
  - Ajouter un composant ColorDialog à votre projet.
  - Modifier l'évènement Click du bouton « Choose Color » de manière à afficher le ColorDialog. Trouver ensuite comment récupérer la couleur que l'utilisateur a choisie et faites en sorte de l'utiliser dans la méthode Paint du Panel.
- Courbe de Béziers :
  - L'objet Graphics de la méthode Paint sait tracer des courbes de Béziers. Pour cela il suffit de lui fournir un tableau de Points contenant les points de contrôle. Ce tableau doit obligatoirement contenir un multiple de 3 plus 1 points (4, 7, 11, 13, ... points).
  - Modifiez la méthode Paint de manière à tracer la courbe de Béziers si le nombre de points donnés par l'utilisateur correspond à la contrainte précédente. Vérifiez que cela fonctionne.
  - Modifiez à nouveau la méthode Paint de manière à afficher la plus longue courbe de Béziers possible. Par exemple si l'utilisateur a donné 6 points, la plus longue courbe est celle donnée par les 4 premiers points. Vérifiez que cela fonctionne.
- Double buffering : Lors de vos tests vous avez peut-être remarqué un scintillement occasionnel lors de l'ajout de points. Cela provient du fait que la zone de dessin est directement lu par le matériel pour réaliser le rendu à l'écran et que ce rendu n'est pas synchronisé avec vos actions. Il est donc possible que l'écran affiche à un

moment une version incomplète du dessin que vous être en train de réaliser. Ce problème peut être résolu grâce à la technique du double buffering. Concrètement nous allons utiliser deux zones graphiques, l'une contient ce qui est affiché et l'autre nous permet de dessiner en toute tranquillité. Lorsque notre dessin est terminé il suffit d'inverser les 2 buffers : la zone de dessin est affichée et l'ancienne zone affichée peut être utilisée pour dessiner. Ceci est réalisé grâce à la classe BufferedGraphics.

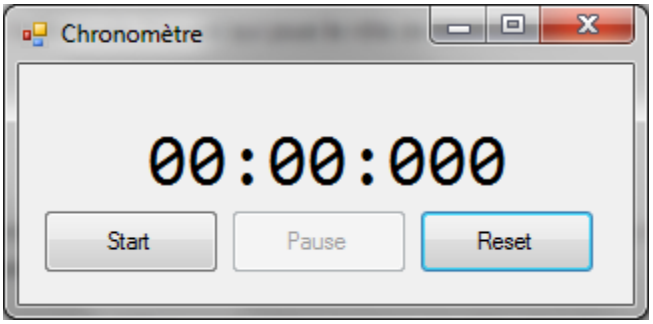
- Pour utiliser cette classe, il va nous falloir réserver de l'espace au début de la méthode Paint, dessiner dedans, échanger les deux buffers et libérer l'espace réservé.
- L'objet BufferedGraphics est relativement complexe et n'expose pas de constructeur public. Pour le créer il faut passer par la méthode Allocate de l'objet de type BufferedGraphicsContext représentant le contexte d'exécution actuel (qui est stocké dans la propriété statique Current de la classe BufferedGraphicsManager). La méthode Allocate demande 2 arguments, l'objet Graphics actuel et la zone de dessin (que vous pouvez récupérer dans les arguments de la méthode Paint). (En Génie Logiciel, vous verrez que cette technique de construction d'objet correspond au Design Pattern « Builder »).
- Une fois l'objet BufferedGraphics créé, vous pour récupérer sa propriété Graphics pour effectuez vos dessins.
- Une fois vos dessins terminé, appelez la méthode Render de l'objet BufferedGraphics qui réalisera l'échange de buffer.
- Pour finir, libérez l'objet BufferedGraphics en appelant sa méthode Dispose (L'objet BufferedGraphics utilise des ressources non managées par le CLR, qui ne seront donc pas récupérable par le ramasse-miettes : il faut donc les libérés manuellement avec la méthode Dispose qui joue le rôle de destructeur).
- Vérifiez que tout fonctionne.

3. Chronomètre

Description : Nous allons écrire un chronomètre avec une précision de l'ordre de la milliseconde. Ce chronomètre possèdera trois boutons : Start, Pause et Reset.

Principe de fonctionnement : L'objet Timer permet de définir une fonction qui sera appelée périodiquement à un intervalle de temps exprimé en millisecondes. Ce composant permet de rafraichir un affichage régulièrement néanmoins, en pratique il s'agit d'une politique de "Best Effort" et rien ne garantit que l'intervalle de temps spécifié sera respecté rigoureusement. Le Timer ne peut donc pas être utilisé pour des mesures précises du temps (notamment en cas de surcharge du système, le timer est susceptible de retarder voir de rater des échéances). Le Timer n'est donc pas suffisant pour faire du temps réel. Pour atteindre cet objectif, il faut combiner le Timer avec un objet de type Timewatch qui lui permet de mesurer le temps avec une précision de l'ordre de la milliseconde (voir plus sur certains systèmes qui permettent de compter directement les cycles processeurs).

Contenu : Timer, Timewatch, machine à état, String.Format.



Instructions :

- Commencez par créer un classe Temps. Cette classe possède 3 champs : milliseconde, seconde, minute, un constructeur qui reçoit en paramètre un nombre de millisecondes (long) et initialise les champs pour qu'ils correspondent à cette durée totale et une redéfinition de ToString qui renvoie une représentation en string formaté de cette manière "mm:ss:msmsms"
- Créez ensuite l'interface graphique qui comprend: un label, trois boutons et un Timer. Le bouton "pause" est grisé à l'origine (Enabled=false)
- Définissez ensuite les évènements suivants:
    - Click sur Start : déclenche le Timewatch, déclenche le Timer, grise le bouton Start, active le bouton Pause.
    - Click sur Pause : stoppe le Timewatch, stoppe le Timer, grise le bouton Pause, active le bouton Start.
    - Click sur Reset : stoppe le Timewatch et le remet à zéro, stoppe le Timer, grise le bouton Pause, active le bouton Start
    - Tick du Timer : Crée un objet Temps correspondant à la durée mesurée par le Timewatch, affiche l'objet Temps dans le Label.
- Vérifiez que tout fonctionne.