2

L'environnement de développement .NET

2. L'environnement de développement .NET

Page de démarrage

Par défaut, Visual Studio.NET démarre en affichant une page de démarrage sous forme d'une page web. À la gauche de l'écran, des menus vous permettent d'accéder à différentes sections éducatives et informatives. Le menu **Profil**, activé par défaut, vous permet de créer un profil afin de personnaliser le comportement et l'affichage de Visual Studio.NET.

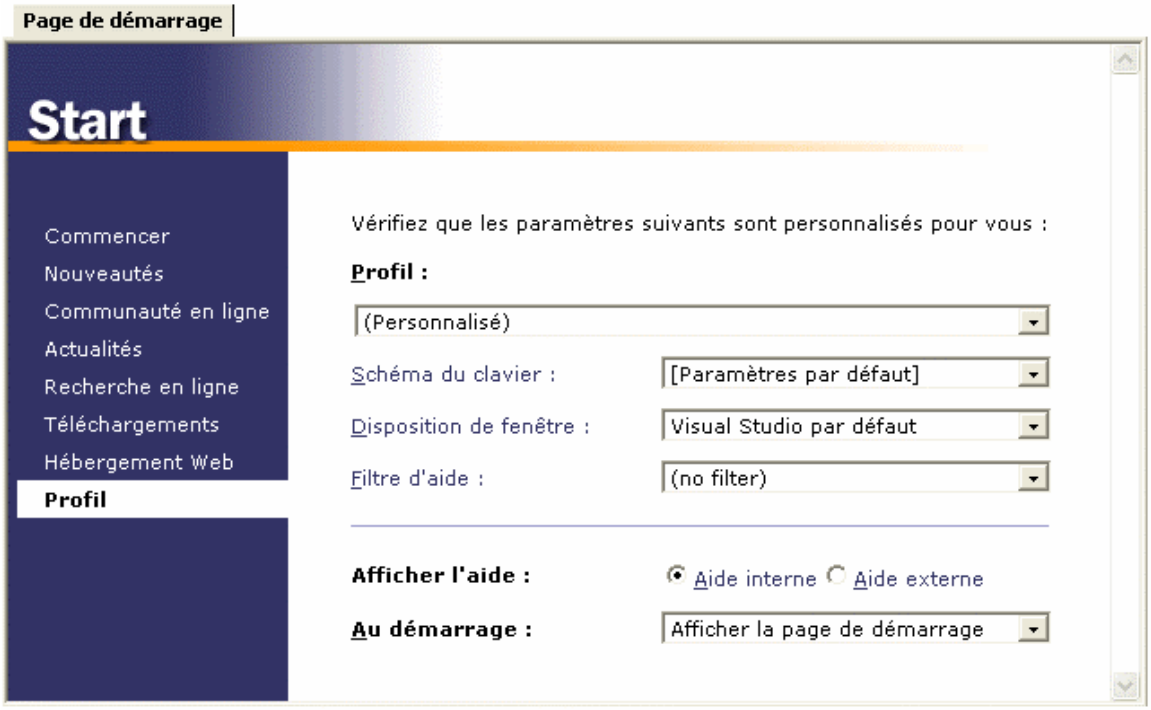


Figure 2.1 – La page de démarrage

Vous pouvez sélectionner un profil prédéfini au sein de la liste déroulante libellée Profil ou créer votre profil personnalisé en modifiant les valeurs des listes déroulantes subséquentes. Vous pouvez ainsi personnaliser la disposition des fenêtres, le schéma du clavier *(touches de raccourdi)*, l'affichage de l'aide, etc. Si vous fermez la page de démarrage, vous pouvez l'afficher à nouveau en activant le menu **Aide / Afficher la page de démarrage** n'important quand au sein de Visual Studio.

Note : L'ensemble des paramètres modifiables via la page de démarrage le sont également par l'intermédiaire de la boîte d'options en activant le menu **Outils / Options**.

Gestion de projets

Lorsque vous démarrez Visual Studio.NET, vous pouvez créer un nouveau projet ou ouvrir un projet existant. Pour démarrer un nouveau projet, sélectionnez le menu **Fichier**, sélectionnez **Nouveau** puis cliquez sur **Projet**. La boîte de dialogue suivante s'affichera alors :

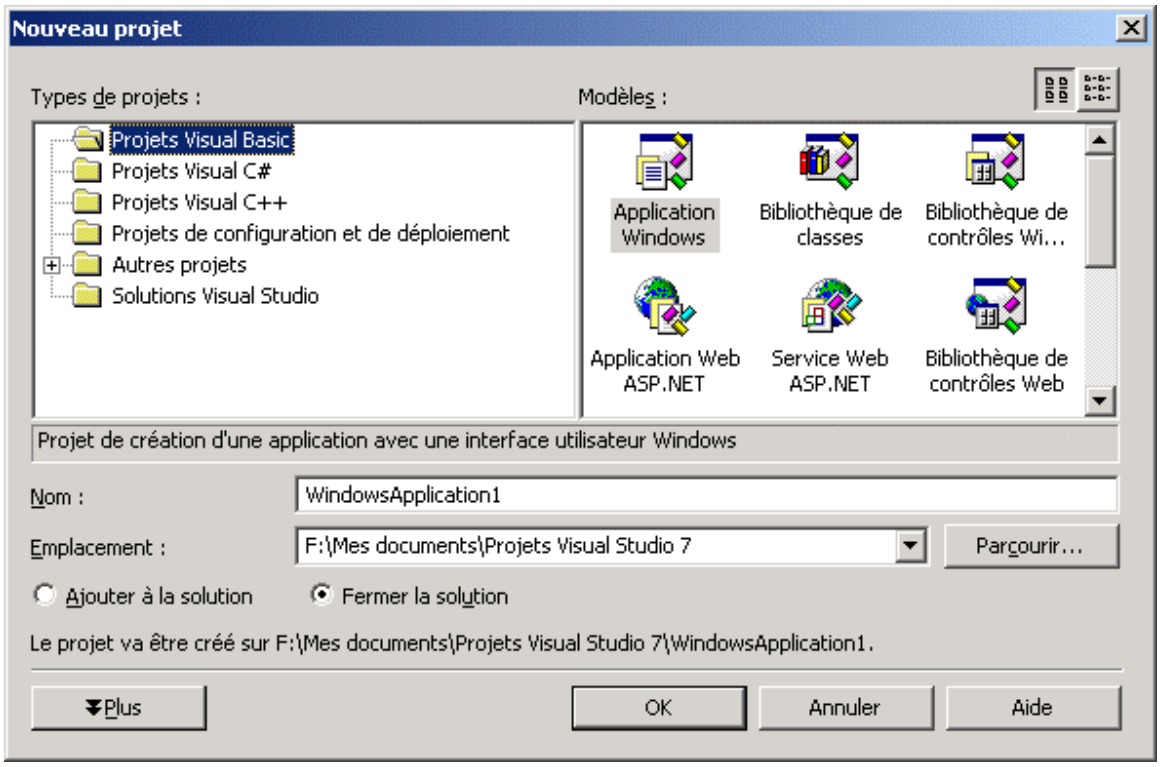


Figure 2.2 – Boîte de dialogue permettant la création d'un nouveau projet.

Vous savez maintenant que Visual Studio.NET prévoit une plateforme standard pour l'ensemble des langages de programmation et qu'il englobe l'ensemble des outils au sein du même environnement de développement. Ainsi, la boîte de dialogue vous offre de choisir le **type de projets** que vous désirez créer dans la liste de gauche.

À chacun des ces langages correspondent différents **modèles** de projets affichés dans la liste de droite. Un modèle de projet vous permet de démarrer un projet de base sans que vous ayez à inscrire les codes minimaux nécessaire à tout projet. Ainsi, en choisissant un modèle de projet, plusieurs instructions et directives seront déjà inscrites au sein de votre code. Il est toutefois possible de démarrer un projet vide en sélectionnant le modèle de projet **Projet vide**.

1) Sélectionnez **Projets Visual Basic** au sein de la liste des types de projets.

2) Sélectionnez **Application Windows** au sein de la liste des modèles de projets.

3) Spécifiez le nom `Premier` à l'intérieur de zone de saisie **Nom**. Quoiqu'il sera ultérieurement possible de modifier le nom inscrit au démarrage du projet, il est préférable que vous nommiez immédiatement correctement votre projet afin d'éviter à avoir à modifier manuellement plusieurs propriétés pour le faire ultérieurement.

4) Sélectionnez l'endroit à lequel vous désirez sauvegarder les fichiers de votre projet sur votre disque dur au sein de la zone de saisie **Emplacement**. Vous pouvez utiliser le bouton **Parcourir** afin de sélectionner le répertoire à l'aide de l'explorateur de fichiers. Par défaut, Visual Studio.NET tentera de créer les nouveaux projets au sein d'un répertoire `Projets Visual Studio` à l'intérieur du répertoire attitré `Mes Documents`.

5) Appuyez sur **Ok** et l'environnement complet de développement de Visual Studio.NET s'affichera.

Pour ouvrir un projet existant, sélectionnez le menu **Fichier**, sélectionnez **Ouvrir** puis cliquez sur **Projet**. La boîte de dialogue suivante s'affichera alors :

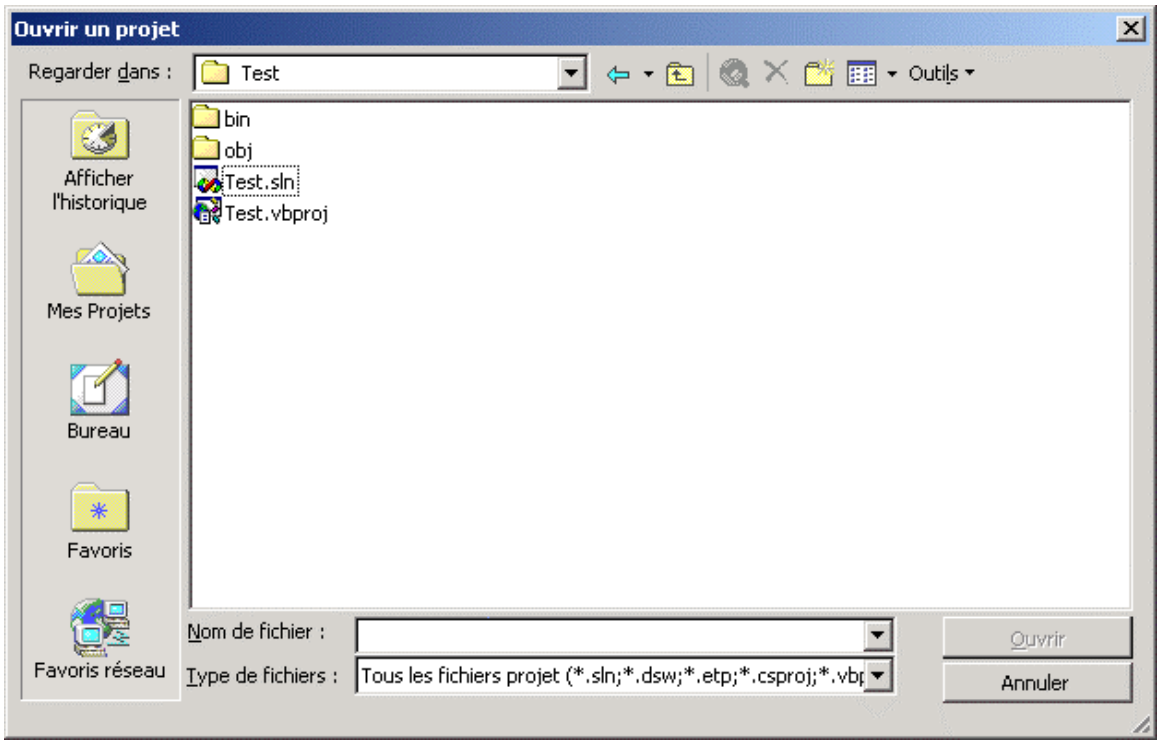


Figure 2.3 – La boîte de dialogue permettant d'ouvrir un projet existant.

Sélectionnez le fichier de solution (*.sln) ou de projet (*.vbproj) puis appuyez sur **Ouvrir** et l'environnement complet de développement de Visual Studio.NET s'affichera.

Notez que Visual Studio.NET prévoit le répertoire `Mes Projets` dans lequel il vous est possible de centraliser les fichiers sources de vos développements.

Gestion des fenêtres

L'environnement de Visual Basic.NET vous offre une multitude de barres d'outils et de fenêtres permettant de gérer différents aspects de votre projet. Les barres d'outils peuvent être affichées et dissimulées à l'aide du menu **Affichage** qui liste l'ensemble des fenêtres et barres d'outils pouvant être affichées. Par défaut, chacune des fenêtres s'ancrent aux côtés de l'écran. Quoique vous puissiez les redimensionner, les fenêtres demeurent à une position fixe. Vous pouvez cependant modifier ce comportement par défaut en appuyant sur le bouton droit de la souris sur la fenêtre désirée. Un menu contextuel s'affichera alors vous permettant de spécifier si la fenêtre est ancrable ou flottante.

- Une **fenêtre ancrable** s'ancrent automatiquement sur les parois de l'écran. Si la fenêtre principale de Visual Basic est redimensionnée, la fenêtre ancrable modifiera sa position afin de conserver ses dimensions.

- Une **fenêtre flottante** s'affiche par-dessus l'ensemble des autres fenêtres normales et est la première à l'écran lorsqu'elle possède le focus. Si la fenêtre principale de Visual Basic est redimensionnée, la fenêtre flottante ne modifiera aucunement ses dimensions ni sa position et pourra même en venir à s'afficher à l'extérieur de la fenêtre principale de Visual Basic.

Voici une brève présentation des principales fenêtres que vous utiliserez sous Visual Basic.

Explorateur de solutions

L'explorateur de solutions s'affiche par défaut à la droite de l'écran et liste l'ensemble des fichiers composants votre projet et votre solution. Les fichiers sont affichés de manière hiérarchique afin de bien discerner quel fichier appartient à quel projet dans le cas où votre solution serait composé de plusieurs projets. De plus, cet affichage vous permet de dissimuler temporairement certains sous-fichiers d'une catégorie afin d'en augmenter la visibilité.

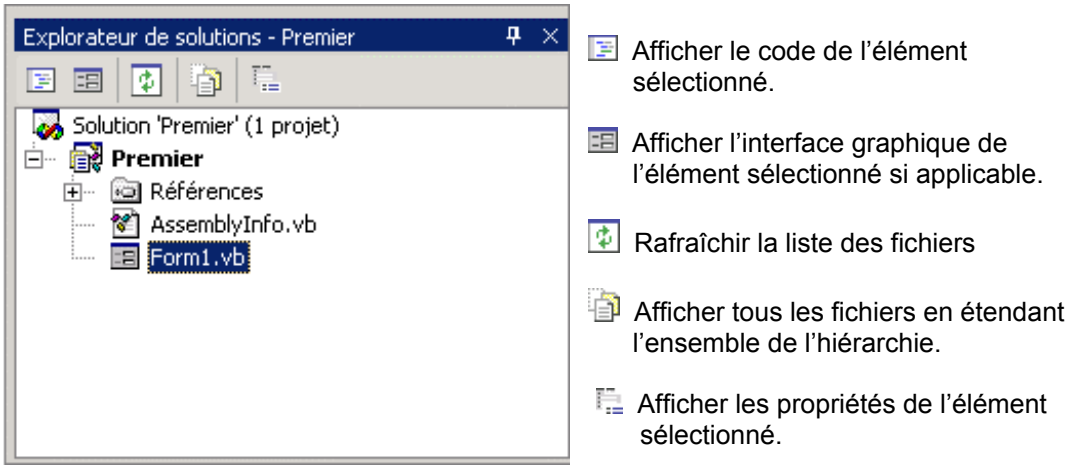


Figure 2.4 – Explorateur de solutions

Lorsque vous double-cliquez sur un fichier de la liste, l'interface graphique ou le code lui correspondant s'affiche dans la partie centrale de l'écran, selon le type de fichier sélectionné. Les fichiers pouvant se retrouver dans l'Explorateur de solutions seront expliqués plus loin. Vous pouvez déplacer un fichier d'un dossier à un autre à l'aide de la technique de glisser-poser *(drag 'n drop)*.

Boîte à outils

La boîte à outils s'affiche par défaut à la gauche de l'écran et liste l'ensemble des contrôles graphiques que vous pouvez déposer sur un formulaire Visual Basic. Un formulaire étant une fenêtre ou une boîte de dialogue qui sera affichée à l'utilisateur, les contrôles sont les éléments qui se retrouveront sur cette fenêtre : boutons, zones de texte, listes déroulantes, etc.

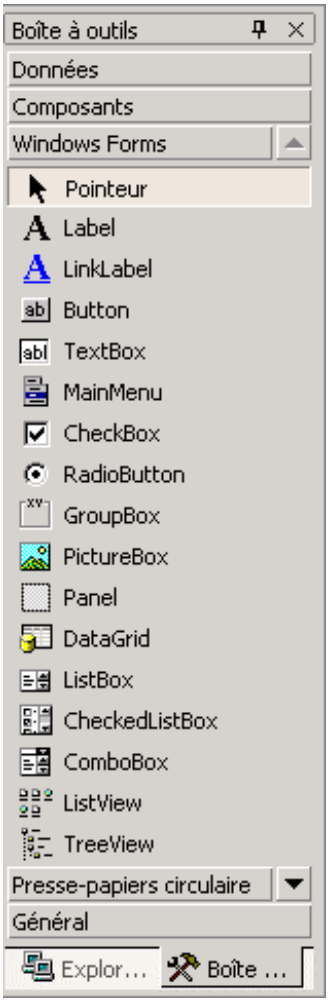


Figure 2.5 – Boîte à outils

Les outils sont regroupés par catégories au sein de la boîte d'outils.

Données
Liste de contrôles graphiques d'accès et de manipulation de bases de données pouvant être déposés sur un formulaire Windows.

Composants
Liste de composants divers pouvant être déposés sur un formulaire Windows.

Windows Forms
Liste de l'ensemble des contrôles graphiques pouvant être déposés spécifiquement sur un formulaire Windows. Il s'agit de la catégorie de contrôles que vous utiliserez le plus dans la conception d'applications Windows.

Presse-papiers circulaire
Liste des éléments copiés ou coupés insérés dans le presse-papier que vous pouvez sélectionner afin de coller à divers endroits. Cette section agit un peu comme le presse-papier multiple de Microsoft® Word 2000.

Général
Par défaut, cette catégorie ne contient qu'un accès au pointeur normal permettant d'annuler toute sélection antérieure de contrôle.

Cette catégorie vous permet d'ajouter des composants personnalisés ou des fragments de code. Pour ce faire, il suffit de sélectionner le fragment de code que l'on désire réutiliser régulièrement puis de glisser la sélection au-dessus de l'onglet `Général` de la Boîte à outils. Un élément supplémentaire s'insérera alors au sein de la Boîte à outils, élément qu'il est possible de renommer ou supprimer à l'aide du menu contextuel en appuyant sur le bouton droit de la souris. Il est ensuite possible de copier ce fragment au sein du code tout simplement en le faisant glisser de la Boîte à outils vers l'endroit désiré dans le code.

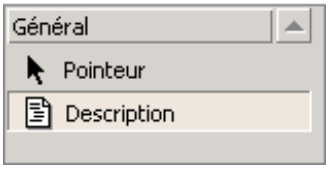
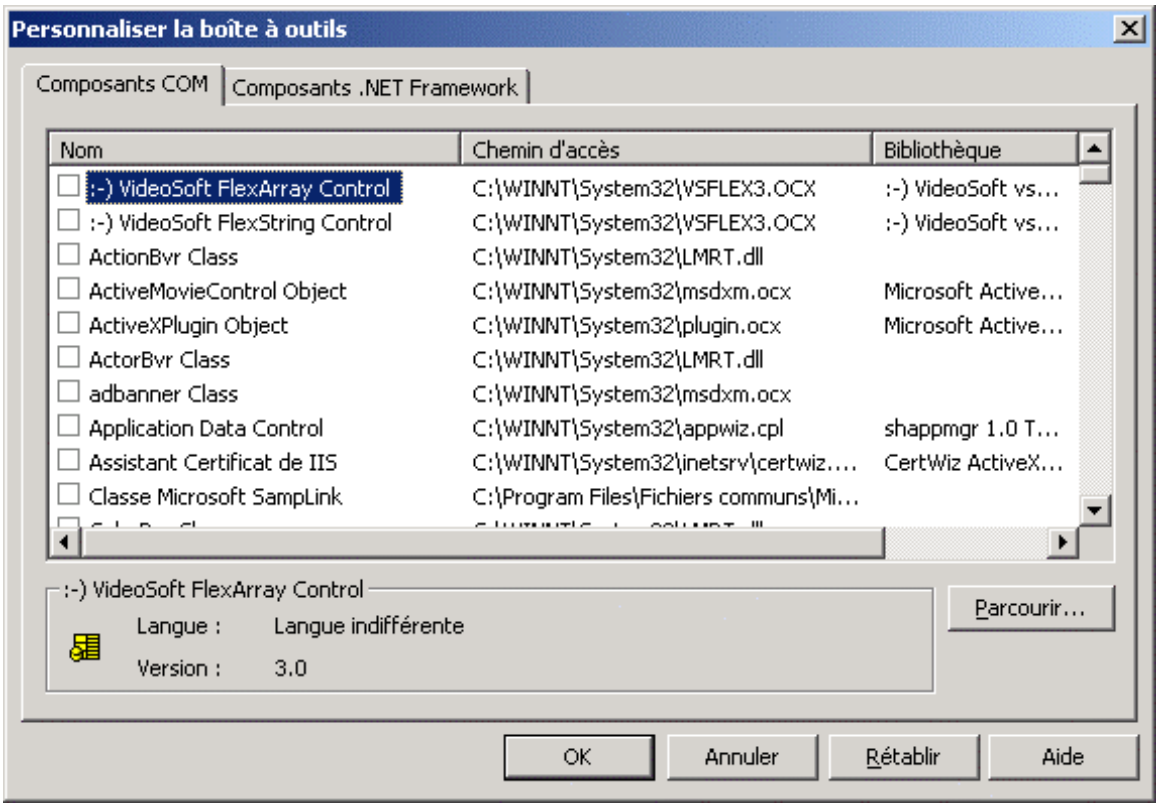


Figure 2.6 – Exemple d'un fragment de code

Vous pouvez renommer tout composant de la boîte à outil en activant le menu contextuel du composant désiré à l'aide du bouton droit de la souris puis en sélectionnant **Renommer**.

La barre à outils affiche certains composants par défaut. Il est cependant possible d'ajouter d'autres contrôles au sein de la boîte à outils en sélectionnant l'élément **Personnaliser la boîte à outils** du menu contextuel de la boîte à outils. La boîte de dialogue suivante s'affiche alors.



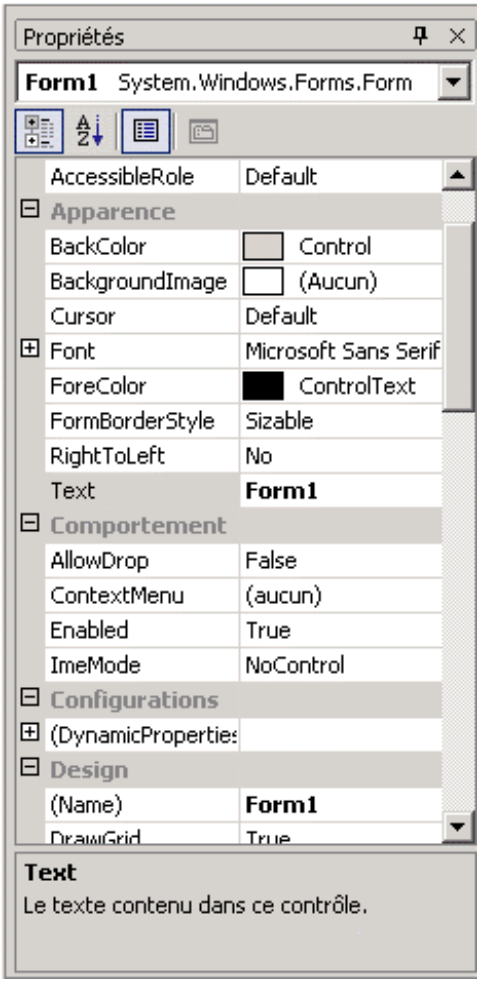
Sont listés sous l'onglet **Composants .NET Framework** l'ensemble des composants de la plate-forme .NET tandis que sont listés sous l'onglet **Composants COM** l'ensemble des composants pouvant provenir de versions antérieures de Windows ou de Visual Basic. Les fichiers des nouveaux composants .NET possèdent l'extension *.DLL ou *.EXE tandis que les fichiers des composants COM possèdent l'extension *.OCX.

Notez que la liste des composants listés au sein de cette boîte de dialogue correspond à ceux dûment enregistrés au sein de la base de registre de votre système d'exploitation. Ainsi, si vous êtes persuadé posséder un composant mais qu'il ne s'affiche pas dans la liste parce qu'il ne serait pas enregistré dans votre base de registres, vous n'avez qu'à appuyer sur le bouton **Parcourir** afin de le localiser. Ensuite, le composant sera d'abord enregistré au sein de votre base de registres avant d'être ajouté dans la boîte à outils de votre projet.

Les composants doivent être cochés dans les listes **Composants .NET Framework** et **Composants COM** afin qu'ils s'affichent au sein de la boîte à outils.

La fenêtre de propriétés

La fenêtre de propriétés s'affiche par défaut à la droite de l'écran et liste l'ensemble des propriétés propres au contrôle sélectionné sur le formulaire Visual Basic. Ainsi, le contrôle que vous sélectionnez au sein de la boîte à outils et que vous déposez sur le formulaire peut posséder différentes caractéristiques. Un bouton peut posséder des dimensions spécifiques, un libellé, une apparence particulière, etc. Les propriétés définissent donc l'aspect et le comportement des contrôles. La fenêtre des propriétés vous permet de modifier ces caractéristiques.



La liste des propriétés différera d'un contrôle à un autre. Ainsi, pour visualiser et modifier les propriétés d'un composant du formulaire, il est nécessaire de le sélectionner en cliquant dessus. La fenêtre des propriétés ajustera ensuite la liste pour n'afficher que celles correspondants au contrôle sélectionné. Le nom du contrôle pour lequel les propriétés sont affichées s'affiche toujours dans le haut de la fenêtre des propriétés. Dans l'illustration ci-contre, les propriétés de l'objet nommé `Form1` sont affichées.

Par défaut les propriétés sont affichées regroupés en catégories selon leur utilité. Par contre, il est possible d'afficher les propriétés en ordre alphabétique selon votre préférence.

Afficher les propriétés par catégorie.

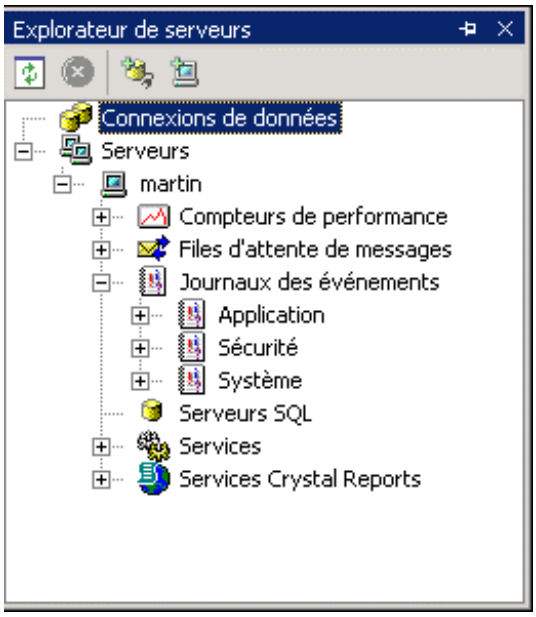
Afficher les propriétés en ordre alphabétique.

Figure 2.8 – Fenêtre des propriétés

Certaines propriétés sont composées de plusieurs propriétés afin d'être complètes. Par exemple, la propriété `Font` définissant l'aspect de la police de caractère utilisée pour afficher un texte sur un contrôle peut être définie par ses dimensions, son aspect gras ou italique, sa famille de police, etc. Ces propriétés définies par d'autre sous-propriétés sont affichées dans la Fenêtre des propriétés précédées d'un symbole ⊞. Il est donc possible d'afficher les sous-propriétés en appuyant sur le ⊞ qui se transformera alors en symbole ⊟ ou de dissimuler les sous-propriétés en appuyant sur le ⊟.

Explorateur de serveurs

L'Explorateur de serveurs s'affiche par défaut à la gauche de l'écran et liste plusieurs fonctionnalités d'un ou de plusieurs serveurs accessibles par Visual Basic. L'Explorateur de serveur permet entres autres d'accéder rapidement aux bases de données d'un serveur SQL Server, aux services d'un serveur Windows, aux journaux d'événement, etc.

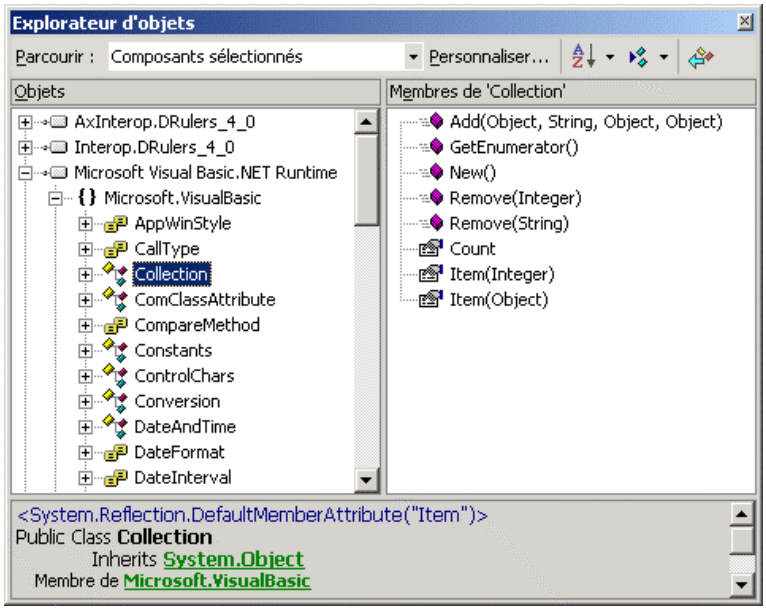


Il est possible de se connecter à un serveur distant afin de visualiser la liste de ses services en appuyant sur le bouton droit de la souris et en sélectionnant le menu Connecter un serveur.

L'utilisation de l'Explorateur de serveurs sera examinée en profondeur plus loin.

Figure 2.9 – Explorateur de serveurs

Explorateur d'objets



L'Explorateur d'objets permet de visualiser le modèle d'objets de la plate-forme .NET et des autres composants personnalisés entres autres créés à l'aide de versions antérieures.

L'Explorateur d'objets liste à la gauche la liste des composants, à la droite les membres du composant sélectionné et en affiche une description dans le bas.

Figure 2.10 – Explorateur d'objets

Chacune des fenêtres précédemment nommées possède un icône en forme de punaise dans le coin supérieur droit. Cette fonctionnalité permet d'optimiser la surface de travail de Visual Studio.NET. En effet, lorsque la punaise s'affiche verticalement la fenêtre demeure fixe et occupe de l'espace à l'écran, diminuant ainsi la surface disponible pour les autres fenêtres importantes telles celles permettant la manipulation du formulaire ou l'écriture du code.

Lorsque la punaise s'affiche horizontalement la fenêtre se dissimulera automatique dès qu'elle perdra le focus au profit d'une autre fenêtre. Vous modifiez le comportement des fenêtres simplement en cliquant sur la punaise qui changera simultanément de position.

Tel qu'illustré ci-contre, l'ensemble des fenêtres dissimulées afficheront un onglet permettant d'y accéder rapidement. Dès que vous glisserez le pointeur de souris au-dessus d'un de ces onglets, la fenêtre correspondante s'affichera à nouveau.

Quoique ce comportement peut s'avérer irritant lors des premières utilisations, on y prend rapidement goût en optimisant l'affichage de la fenêtre sur laquelle on travaille présentement.

Ceux qui ont travaillé avec les versions antérieures de Visual Basic seront certainement heureux d'apprendre que Visual Studio peut afficher les différentes fenêtres en environnement MDI classique tel qu'illustré ci-contre :

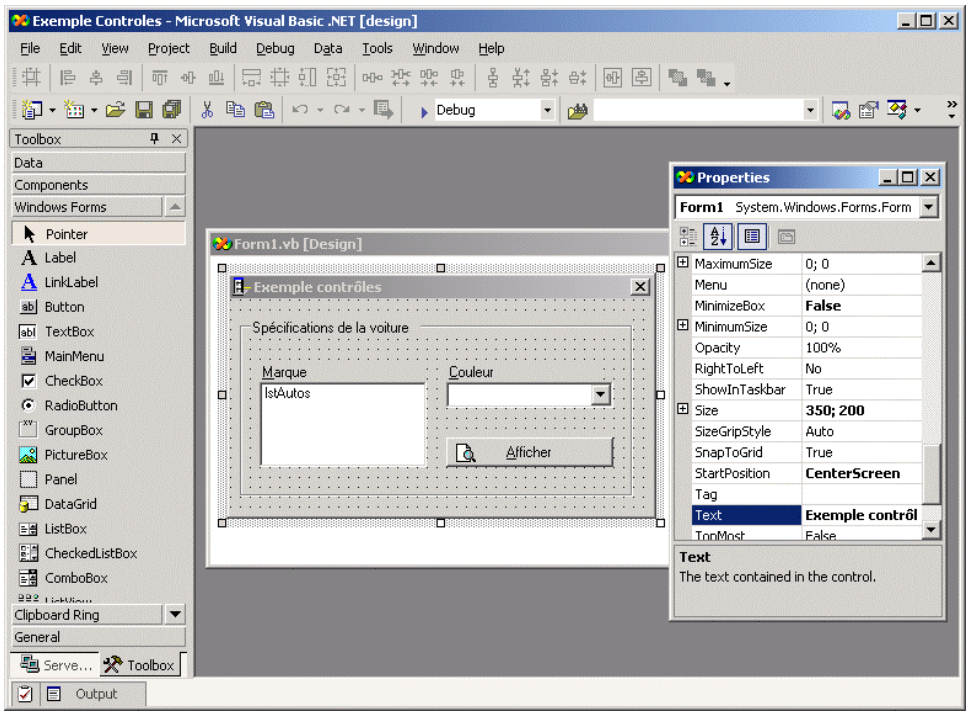


Figure 2.11 – Visual Studio.NET en environnement MDI.

Pour ce faire, il suffit d'activer le menu **Outils / Options** et de cocher l'option **Environnement MDI** située sous l'onglet **Général**.

Gestion des barres d'outils

Vous pouvez personnaliser l'affichage des barres d'outils au sein de votre environnement de développement afin de ne mettre à votre disposition que les fonctionnalités que vous utilisez le plus couramment. Vous pouvez préciser qu'une barre d'outils doit s'afficher ou nom en activant le menu **Affichage / Barres d'outils** puis en cochant seulement les barres d'outils que vous désirez voir s'afficher.

Vous pouvez accéder à des fonctionnalités supplémentaires afin de personnaliser l'affichage des barres d'outils en activant le menu **Affichage / Barres d'outils / Personnaliser**.

La boîte de dialogue affiche à sa gauche la liste des barres d'outils disponibles et, cochées, celles spécifiées comme devant s'afficher au sein de l'environnement de développement.

Notez que le fait qu'une barre d'outils est affichée par défaut ou non est directement relié au type de projet démarré ainsi qu'au profil sélectionné au sein de la Page de démarrage. Voyez le début du présent chapitre à ce sujet.

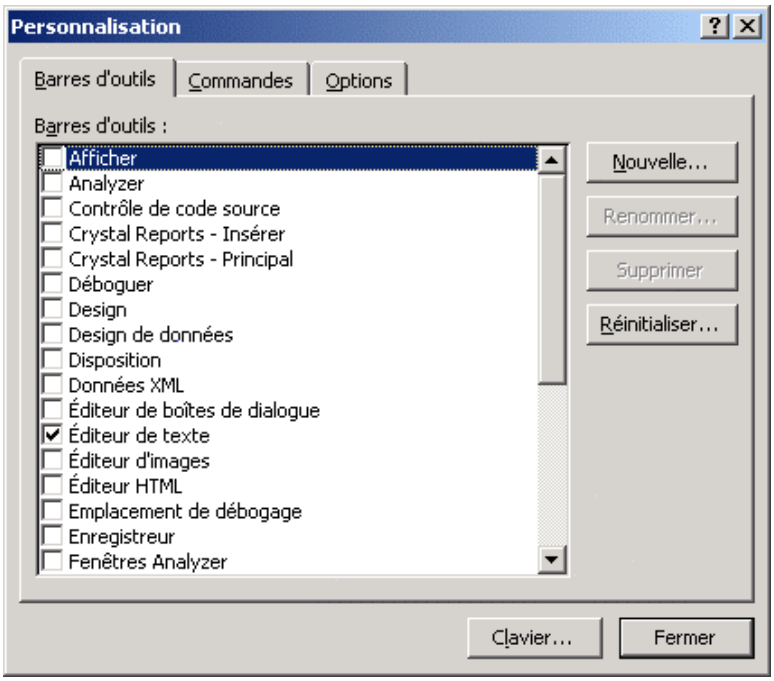


Figure 2.12 – Personnalisation des barres d'outils

Vous pouvez vous créer une nouvelle barre d'outils personnalisée en appuyant sur le bouton **Nouvelle**. Une boîte de dialogue vous invitera alors à préciser les fonctionnalités que vous désirez posséder sur cette nouvelle barre d'outils. Vous pouvez réinitialiser l'affichage des barres d'outils tel qu'il était au premier démarrage de Visual Studio.NET en appuyant sur le bouton **Réinitialiser**.

Visual Studio.NET vous permet pleinement de configurer les raccourcis du clavier permettant d'activer les différentes fonctionnalités du logiciel en appuyant sur le bouton **Clavier**. La fenêtre des options s'affiche alors. Notez que ce bouton est accessible via le présent onglet ainsi qu'à partir de l'ensemble des autres onglets de cette boîte de dialogue.

Lorsqu'une barre d'outils est affichée au sein de l'environnement de Visual Studio.NET, il vous est possible d'en personnaliser les commandes qui y sont disponibles. Pour ce faire, activez l'onglet **Commandes** de la boîte de dialogue et vous verrez s'afficher en deux listes l'ensemble des commandes disponibles regroupées par catégories.

Pour ajouter une commande à une barre d'outils déjà affichée au sein de l'environnement de Visual Studio, sélectionnez une catégorie dans la liste de gauche puis, au sein de la liste de droite, sélectionnez la commande désirée. En tenant enfoncé le bouton de la souris, glissez la commande désirée à l'extérieure de la boîte de dialogue et déposez-la sur la barre d'outils désirée.

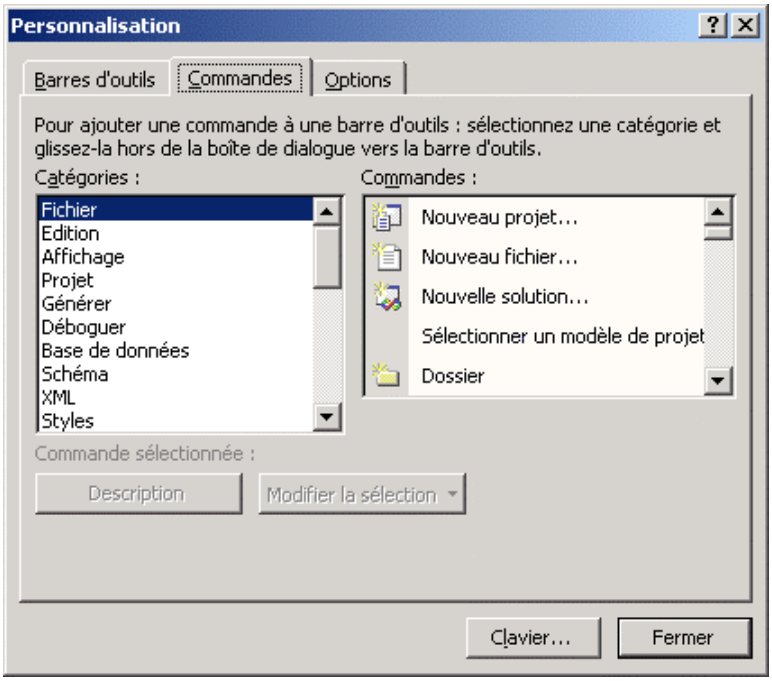


Figure 2.13 – Personnalisation des commandes

Finalement différents autres options dont la pertinence demeure à prouver sont personnalisables à l'aide du dernier onglet libellé **Options** de la même boite de dialogue.

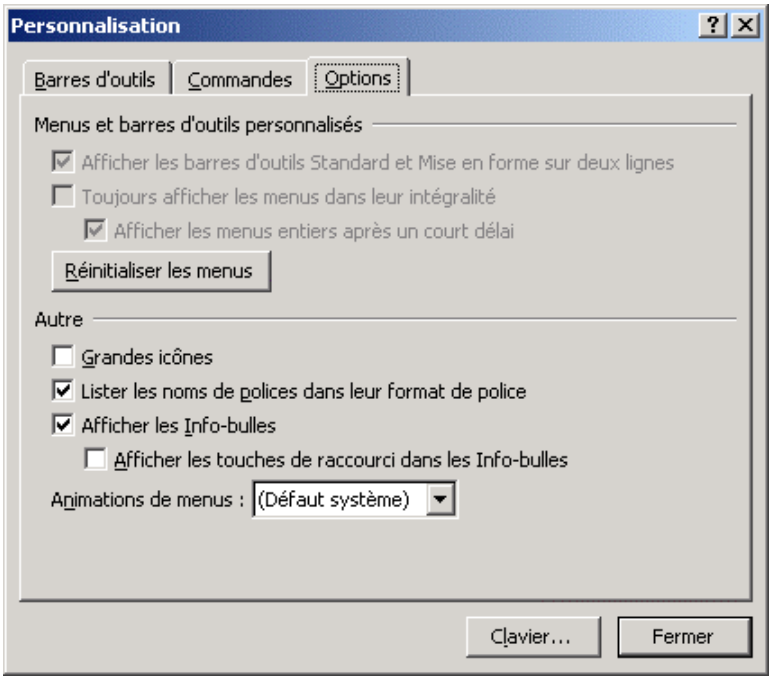


Figure 2.14 – Options personnalisables

Fichiers composants un projet

Un projet Visual Basic.NET est composé de plusieurs fichiers possédant leur utilité propre :

Fichiers de code Visual Basic (*.vb)
Ces fichiers contiennent le code Visual Basic, les informations au sujet de l'interface utilisateur, etc. Ces fichiers peuvent posséder une utilité différente de l'un à l'autre même s'ils possèdent tous la même extension. Les fichiers de classes, de modules, de formulaires posséderont donc tous la même extension.

Fichiers de Projet (*.vbproj)
Un Projet regroupe l'ensemble des fichiers de code afin de créer un seul exécutable (*.EXE, *.DLL, etc.) Les propriétés du projet définiront le comportement de celui-ci.

Fichiers de Solution (*.sln, *.suo)
Une Solution représente une solution complète de développement pouvant être composée de plusieurs projets. Les premiers programmes que vous écrirez seront composés d'un seul projet mais, à un niveau plus avancé, il pourrait être nécessaire d'accéder au code d'une application de gestion en Visual Basic, à un site web affichant les données en C# et le tout au sein de la même fenêtre Visual Studio.

Pour ceux et celles qui auraient travaillé à l'aide des versions antérieures de Visual Basic, comparez la Solution au Groupe de projets des versions antérieures de Visual Basic. Elle permet en effet de regrouper plusieurs projets qui nécessiteraient une interaction serrée entre eux. La principale différence peut se trouver dans le fait qu'une solution peut comporter des projets écrits dans différents langages autres que Visual Basic.

Pour ajouter un nouveau fichier au projet courant, sélectionnez le menu **Fichier**, sélectionnez **Nouveau** puis sélectionnez au sein de la boîte de dialogue telle qu'illustrée ci-dessous le type de fichier à créer.

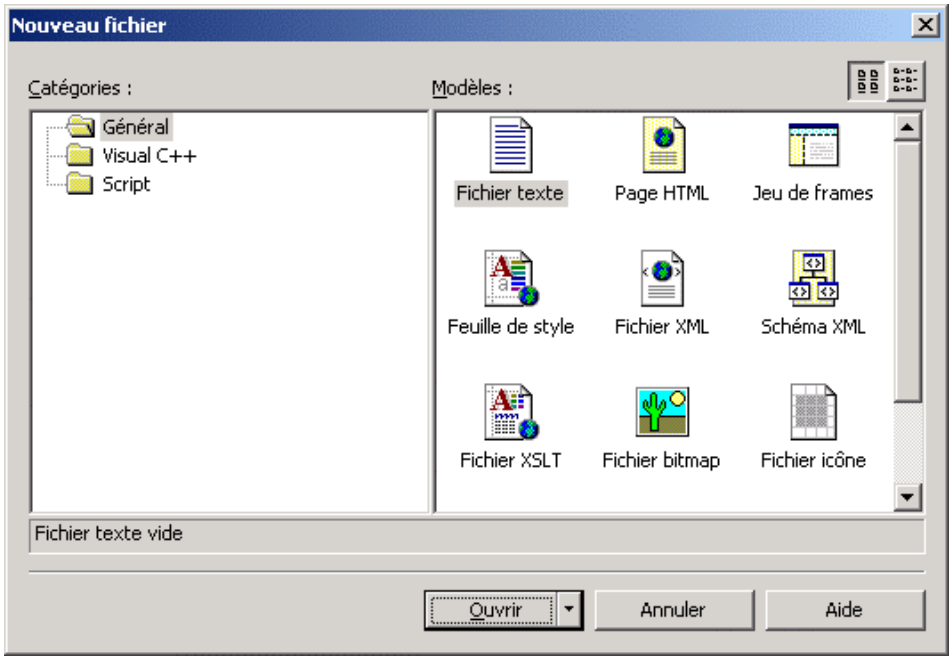


Figure 2.15 – Boîte de dialogue permettant d'ajouter un nouveau fichier au projet

Pour ajouter un nouveau formulaire, module de classe, module ou contrôle personnalisé, sélectionnez le menu **Projet** puis choisissez le menu correspondant.

L'ensemble des tâche précédemment nommées peuvent être réalisées à l'aide des menus contextuels de l'Explorateur de projets en cliquant du bouton droit de la souris sur l'entrée représentant le projet au sein de lequel vous désirez ajouter un fichier.

Dans l'exemple ci-contre, le menu contextuel nous permet rapidement d'ajouter un nouveau formulaire au projet sélectionné.

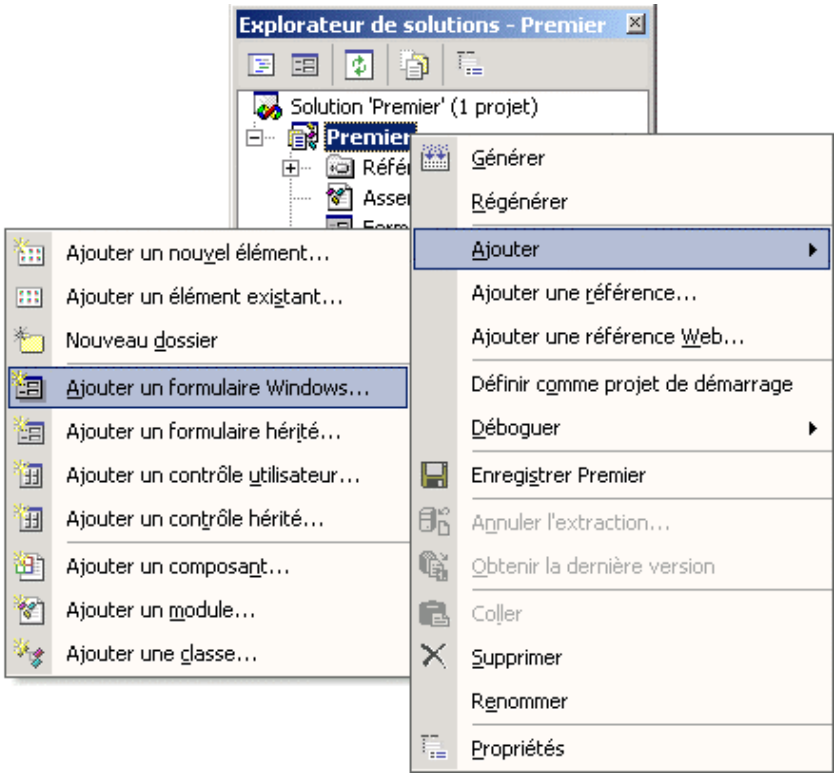


Figure 2.16 – Les menus contextuels vous permettent d'accéder à toutes les fonctionnalités utiles selon le contexte.

Pour ajouter un projet à la solution courante, sélectionnez le menu **Fichier**, sélectionnez **Ajouter un projet** puis choisissez entre **Nouveau projet** ou **Projet existant**.

Pour enlever un fichier du projet ou enlever un projet de la solution, sélectionnez le fichier correspondant puis sélectionnez le menu **Projet** puis le sous-menu **Exclure du projet**. Le tout est également réalisable à l'aide du menu contextuel sur l'Explorateur de solutions.

Gestion du formulaire et de son code

Visual Basic vous permet de visualiser le formulaire désiré en le sélectionnant au sein de l'Explorateur de solutions puis en double-cliquant dessus ou en appuyant sur le bouton de l'Explorateur de solutions permettant d'afficher l'interface utilisateur de l'élément sélectionné. Le formulaire s'affiche alors au centre de l'environnement de Visual Studio.NET.

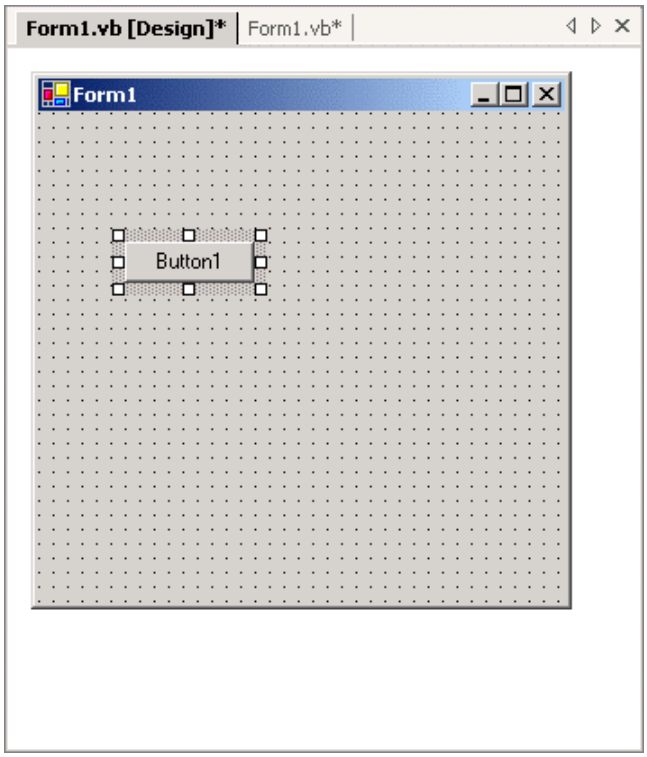


Figure 2.17 – Mode conception du formulaire Windows.

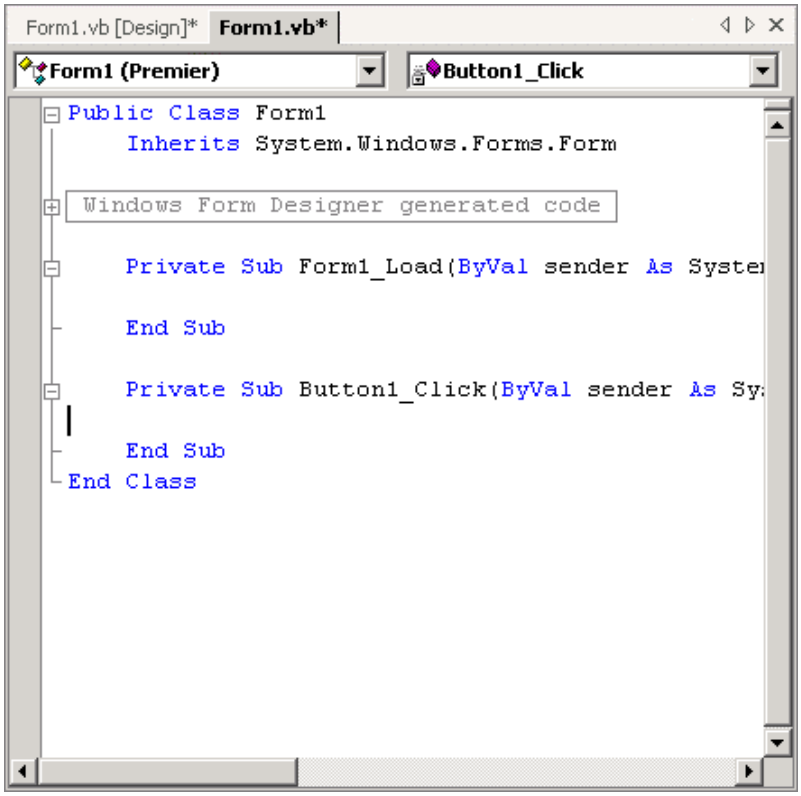
Lorsque le formulaire est affiché en mode conception comme illustré ci-contre, vous pouvez sélectionner des contrôles de la boîte à outils et les glisser sur le formulaire.

L'illustration ci-contre montre un contrôle de type Bouton de commande déposé sur le formulaire. Lorsque l'instance du contrôle est sélectionnée sur le formulaire, la fenêtre des propriétés s'ajuste afin de n'afficher que les propriétés spécifiques à ce composant. Ainsi, vous pouvez modifier les propriétés du contrôle simplement en sélectionnant celui-ci au sein du formulaire.

Les propriétés de chacun des contrôles et leur description respective seront élaborés plus loin.

Lorsqu'un formulaire est affiché dans la section centrale de l'environnement de développement de Visual Basic.NET, les onglets du haut permettent de basculer entre le mode conception graphique *(Design)* du formulaire et son code simplement en cliquant sur l'onglet correspondant.

En mode conception du formulaire, le simple fait de double-cliquer sur le formulaire provoque le passage à l'affichage de son code. Il est également possible de basculer d'un mode à l'autre à l'aide du menu contextuel sur l'une et l'autre des fenêtres. La touche F7 permet également de basculer en mode d'affichage du code tandis que les touches MAJ+F7 permettent de basculer en mode de conception du formulaire.



Le code d'un formulaire est accessible en double-cliquant sur le formulaire ou en sélectionnant l'ongle correspondant dans le haut de l'écran.

La liste déroulante située dans le coin supérieur gauche de la fenêtre liste l'ensemble des composant s du formulaire ainsi que le formulaire lui-même. En sélectionnant le composant pour lequel on désire écrire du code au sein de cette liste, le curseur est automatiquement placé à l'endroit adéquat au sein du code et la liste de droite s'ajuste automatiquement. Cette dernière affiche les événements disponibles pour l'objet sélectionné dans la liste de gauche.

Figure 2.18 – Affichage du code du formulaire Windows.

Les éléments des listes sont affichés en gras lorsque du code y a été spécifié et sont affichés en police de caractères normale lorsque l'élément est inutilisé.

Notez qu'un code est composé de sous-procédures et de fonctions et que Visual Basic.NET les affiche de façon hiérarchique en précédant leurs déclarations d'un symbole ⊞ ou ⊟ .
Toujours dans le but d'obtenir une meilleur visibilité, le programmeur peut cliquer sur le ⊟ et ainsi dissimuler le code de la sous-procédure correspondante afin de n'en voir s'afficher que le prototype ou appuyer sur le ⊞ pour en visualiser l'ensemble du code.

Pour ceux et celles qui auraient travaillé à l'aide des versions antérieures de Visual Basic, notez qu'il est désormais impossible d'exiger l'affichage du module en cours seulement. Avec Visual Studio.NET, l'ensemble du code d'un formulaire s'affiche d'un bloc et ne peut être dissocié des autres sous-procédures et fonctions du même formulaire au grand désagrément des accoutumés de Visual Basic 5.0 et de ses versions antérieures.

Techniques de mise en forme du code

Lors de la mise en marché de Visual Basic 4.0, Microsoft® a séduit les programmeurs en proposant un outil nommé IntelliSense™ permettant à Visual Basic de reconnaître la syntaxe des mots et d'afficher une liste des éléments pouvant être appliqués à l'objet correspondant. Cet outil fut ensuite introduit dans l'ensemble des produits de la suite Visual Studio puisqu'il permettait la réduction des fautes de frappe de la part du programmeur et augmentait sa productivité puisque le nombre d'erreurs qu'ils commettait étaient désormais grandement réduit. Visual Basic.NET vous offre évidemment l'ensemble des fonctionnalités de cet outil.

Liste des propriétés et méthodes (CTRL + J)
Lorsque vous tapez le point (.) séparant un objet d'une de ses propriétés ou de ses méthodes, une liste des propriétés et méthodes pouvant s'appliquer dans le contexte s'affiche. À mesure que vous tapez les premières lettres de la propriété ou de la méthode désirée, la liste s'adapte en appliquant le focus sur l'élément pouvant répondre aux critères selon le contexte. Lorsque le focus se situe sur la propriété ou méthode désirée, vous n'avez qu'à appuyer sur la barre d'espacement afin que Visual Basic complète le mot. Vous pouvez également sélectionner l'élément au sein de la liste déroulante et double-cliquer dessus pour que Visual Basic ne l'inscrive au sein de votre code.

Liste des constantes (CTRL + Maj + J)
Lorsque vous tapez le symbole d'égalité (=) servant à assigner une valeur à une propriété d'un objet, une liste des constantes ou des énumérations représentant les valeurs pouvant s'appliquer dans le contexte s'affiche. À mesure que vous tapez les premières lettres de la constante ou de l'énumération désirée, la liste s'adapte en appliquant le focus sur l'élément pouvant répondre aux critères selon le contexte. Lorsque le focus se situe sur la constante ou énumération désirée, vous n'avez qu'à appuyer sur la barre d'espacement afin que Visual Basic complète le mot. Vous pouvez également sélectionner l'élément au sein de la liste déroulante et double-cliquer dessus pour que Visual Basic ne l'inscrive au sein de votre code.

Information rapide (CTRL + I)
Affiche la syntaxe d'une instruction ou d'une méthode en cours de saisie. Cette fonctionnalité s'active automatiquement lorsque vous tapez les parenthèses ouvrantes (servant à lister les valeurs attendues par les paramètres d'une fonction.

Information sur les paramètres (CTRL + Maj + I)
Présente les informations sur les paramètres d'une instruction ou d'une méthode.

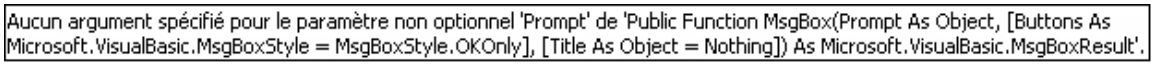
L'outil IntelliSense™ de Visual Basic.NET ajoute à ses fonctionnalités la description des erreurs en mode conception du code. Lorsqu'une erreur est décelée au sein du code, le mot erroné est souligné d'un trait biseauté. De plus, si vous déposez le pointeur de la souris au-dessus du mot erroné, Visual Basic.NET vous affiche une info-bulle décrivant la source de l'erreur.

Pour tester les fonctionnalités de IntelliSense™, tapez le code suivant indiqué en gras, la balance du code étant déjà inscrit par Visual Basic :

```
Public Class Form1
      Inherits System.Windows.Forms.Form

      Private Sub Form1_Load(ByVal sender As System.Object, ▼
            ByVal e As System.EventArgs) Handles MyBase.Load

            MsgBox()

      End Sub
End Class
```

Vous remarquerez que l'instruction `MsgBox(` a provoqué l'affichage d'une info-bulle affichant la liste des paramètres attendus par la fonction `MsgBox` :

De plus, l'instruction est désormais soulignée afin de vous indiquer que celle-ci comporte une erreur. Si vous déposez le pointeur de la souris au-dessus de l'instruction erronée, une info-bulle s'affichera afin de vous indiquer la source de l'erreur. Dans ce cas-ci, l'instruction `MsgBox`, qui a pour fonction de provoquer l'affichage d'une boîte de dialogue, attend impérativement une chaîne de caractères spécifiant le message à inscrire au sein de cette boîte de dialogue. L'info-bulle vous affiche donc la description suivante :



Vous pouvez corriger la situation en modifiant le code précédemment inscrit par celui-ci au sein de lequel nous spécifions le message que la boîte de dialogue devra afficher :

```
Public Class Form1
      Inherits System.Windows.Forms.Form

      Private Sub Form1_Load(ByVal sender As System.Object, ▼
            ByVal e As System.EventArgs) Handles MyBase.Load

            MsgBox("Allô la planète!")

      End Sub
End Class
```

Si vous désirez tester votre premier logiciel écrit sous Visual Basic.NET, sélectionnez le menu **Générer / Générer tout** ou appuyez simplement sur la touche **F5**. Nous reviendrons ultérieurement sur la compilation et l'exécution des logiciels.

La barre d'outils **Éditeur de texte** accessible via le menu **Affichage** / **Barre d'outils** offre une multitude d'outils permettant la mise en forme du code.

Figure 2.19 – Barre d'outils permettant d'éditer le texte.

Voici une description détaillée de l'ensemble des boutons de cette barre d'outils :

Liste des propriétés et des méthodes
Demande explicitement à Visual Basic d'afficher la liste des membres de l'objet sélectionné tel que démontré précédemment.

Informations sur les paramètres
Demande explicitement à Visual Basic d'afficher la liste des paramètres d'une fonction tel que démontré précédemment.

Information rapide
Demande explicitement à Visual Basic d'afficher la syntaxe d'une instruction et d'une fonction en cours de saisie.

Complétion du mot
Demande explicitement à Visual Basic de compléter le mot en cours de saisie tel que démontré précédemment.

Diminuer le retrait
Provoque le retrait du code et en annule ainsi son indentation le cas échéant. Le code est ramené vers la gauche.

Augmenter le retrait
Provoque l'indentation du bloc de code sélectionné. Le code est indenté vers la droite.

Commenter
Provoque la mise en commentaires du bloc de code sélectionné. L'ensemble du bloc de code est commenté.

Enlever le commentaire
Provoque le retrait de commentaires du bloc de code spécifié si celui-ci était en commentaires.

Ajouter un signet
Permet d'ajouter un signet sur la ligne de code présentement sélectionnée. Ainsi, cette ligne sera désormais facile à retrouver à l'aide des deux fonctionnalités suivantes. Un carré cyan s'affiche sur le côté de la ligne de code. Un code peut contenir plusieurs signets.

Atteindre le signet suivant
Provoque le saut au signet suivant. Le code qui y avait été ajouté en signet est affiché.

Atteindre le signet précédent
Provoque le saut au signet précédent. Le code qui y avait été ajouté en signet est affiché.

Supprimer l'ensemble des signets
Provoque la suppression de tous les signets précédemment créés.