

Secteur Tertiaire Informatique Filière étude - développement

Développer des composants d'interface

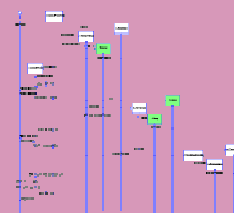
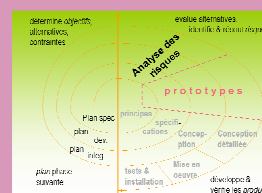
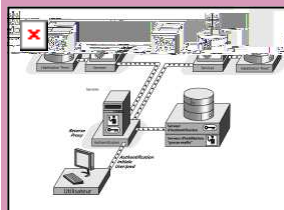
Présentation de la plateforme .NET

Accueil

Apprentissage

Période en
entreprise

Evaluation



Code barre

SOMMAIRE

I.	L'ARCHITECTURE .NET	3
I.1	Présentation.....	3
I.2	Schéma d'architecture	4
I.3	SDK (Software Development Kit).....	4
I.4	NFmk = « .Net Framework »	5
I.5	Fonctionnement d'une application .Net.....	7
I.6	Déploiement d'application.....	8
II.	ENVIRONNEMENT DE DEVELOPPEMENT	9
I.7	Environnements de développement	9
I.8	Installation.....	10
I.9	Exécution en ligne de commande	11
I.10	Visual Studio : création d'un nouveau projet.....	12
I.11	Visual Studio : visualisation du nouveau projet.....	12
I.12	Visual Studio : paramétrage.....	14
I.13	Visual Studio : compilation / exécution.....	16
I.14	Visual Studio : débogage	17
I.15	Aide en ligne du développeur	19
III.	EXERCICES – INSTALLATION ET ESSAI DE L'ENVIRONNEMENT DE DEVELOPPEMENT	21
I.16	Travaux pratiques	21
I.17	Questions.....	22
IV.	ANNEXE : TYPES DE FICHIERS	24
I.18	Projet local	24
I.19	Projet Web	25

I. L'ARCHITECTURE .NET

I.1 PRESENTATION

« **.Net** » (Dot Net) désigne le nouveau modèle d'architecture logicielle de **Microsoft**, le nouvel environnement de développement et d'exécution associé, ainsi qu'un ensemble de spécifications (C#, CIL, ...). Le socle d'exécution « **CLR** » et le nouveau langage de Microsoft « **C#** » ont été normalisés par l'Ecma (European Computer Manufacturer's Systems).

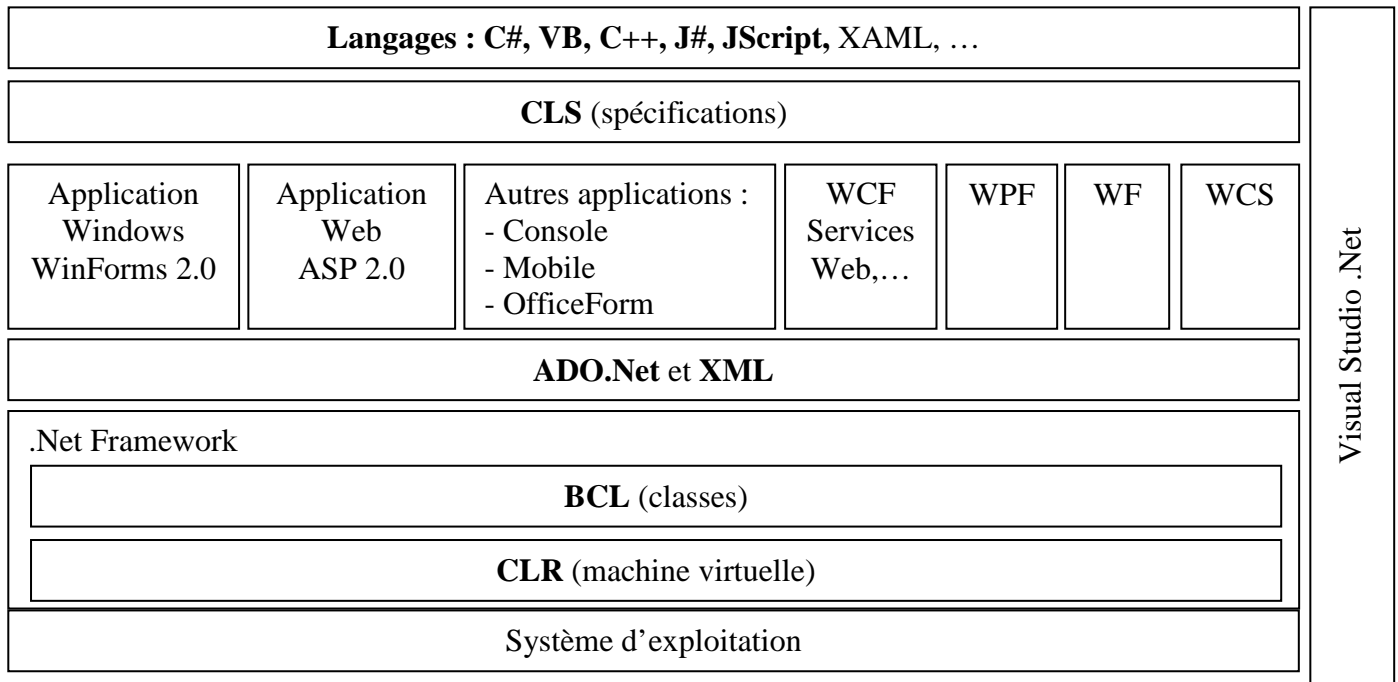
🔑 Historique

Date	Framework	EDI	Langage
2002	Fmk 1.0	Visual Studio 2002	C# 1.0
2003	Fmk 1.1	Visual Studio 2003	
2005	Fmk 2.0	Visual Studio 2005	C# 2.0
2006	Fmk 3.0 (ex WinFX) = Fmk 2.0 ... + WPF (Windows Presentation Foundation, ex Avalon) = moteur d'affichage basé sur XAML (pour Vista, mais fonctionne sous XP et 2003) + WCF (Windows Communication Foundation, ex Indigo) = serveur d'application regroupant services Web , .Net Remoting, MSMQ et BizTalk + WF (Windows Workflow Foundation) = moteur de workflows + WCS (Windows Card Space, ex Infocard) = technique de carte d'identité virtuelle		
Fin 2007	Fmk 3.5 = Fmk 3.0 ... + Implémentation pour Ajax ASP.Net + Langage de requête Linq + etc.		
Fin 2007		Visual Studio 2008 « Orcas » : Nouveaux outils de développement ... Web en Ajax et Silverlight (plugin WPF, ex WPF/E = WPF Everywhere) Plate-forme Office System 2007 - Vista (mise en œuvre simplifiée de WPF/XAML, WCF, WF)	C# 3.0
2012	Fmk 4.0 « Hawaï » : code modulaire permettant ... - un téléchargement à travers Internet uniquement des modules dont l'application à besoin - une installation de l'essentiel du framework dans Windows Server Core		

🔑 Plates-formes cibles

- Plate-forme propriétaire : Windows, Pocket PC (ex Win CE).
- Projet de portage du Framework (voir définition ci-dessous) :
 - projet **Mono 2.2** compatible DotNet 2.0 : Linux, Unix, Free BSD, Solaris et Mac OS
 - projet **Rotor** : Free BSD
 - projet **DotGnu** : Portable.DotNet.
- Annonce de portage sous Stinger (téléphones mobiles), Xbox (console de jeux) et TV Platform.

I.2 SCHEMA D'ARCHITECTURE



I.3 SDK (SOFTWARE DEVELOPMENT KIT)

Le **SDK** est un ensemble de compilateurs et d'outils de développement.

☞ Environnement de développement : **Visual Studio .Net** (voir ce chapitre).

☞ **Une vingtaine de langages** ont été complétés pour être compatibles avec la technologie « .Net ». Ils se partagent un système unique de types de données et une même bibliothèque de classes (**BCL**, voir ci-dessous).

- Langages fournis par Microsoft : Visual Basic (incompatible avec VB6), Visual C++, Visual C#, Visual J# (Java), JScript (devient compilé).

C# (C Sharp) est un nouveau langage réalisant la synthèse de C, C++ et Java : ramasse-miettes, typage fort, code managé ou non (avec pointeurs).

- Autres langages disponibles : C, Ada, Cobol, Fortran (Salford), F#, RPG, Perl, Eiffel, Python, Pascal, Delphi, Smalltalk, Mercury, Mondrian, Oberon, Standard ML, Dialog APL, Haskell, ...

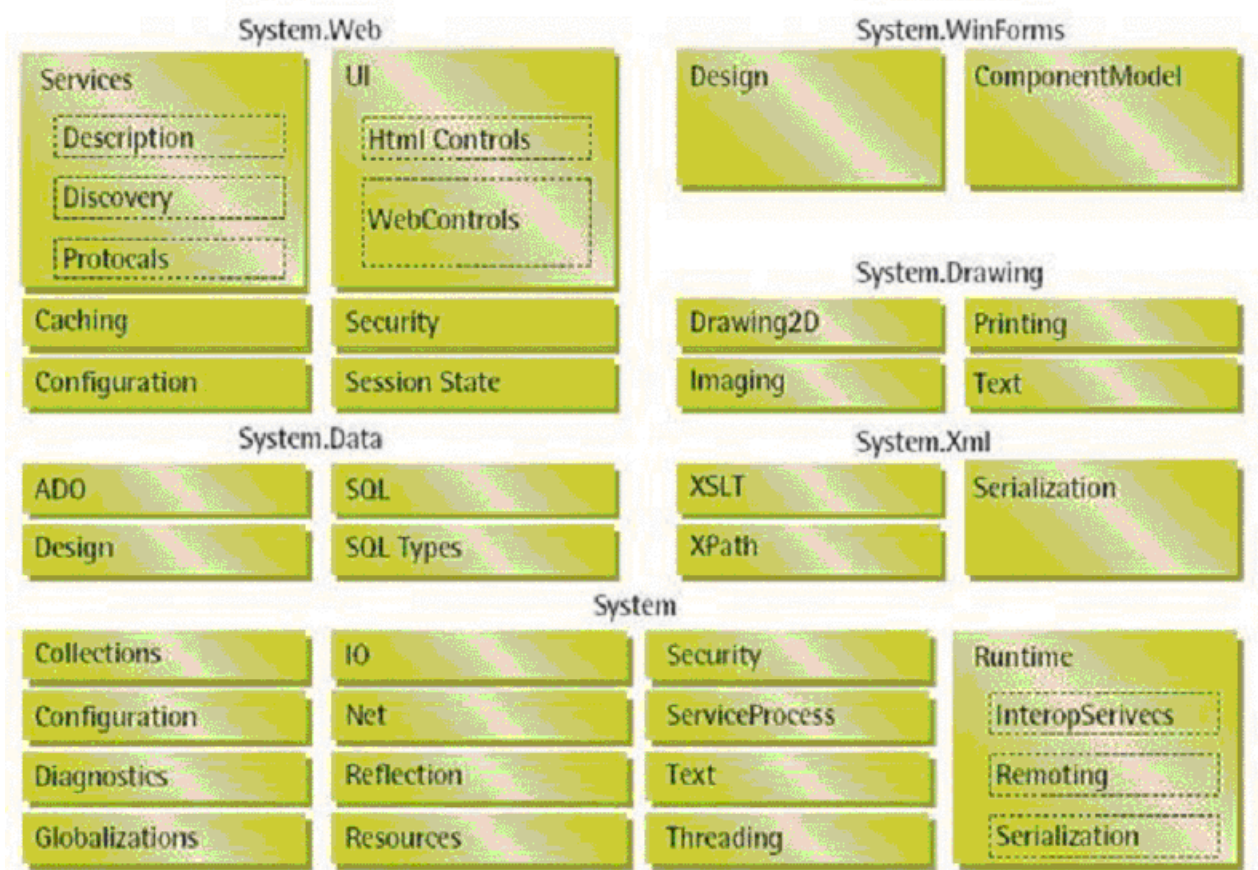
Pour qu'un langage soit portable sur « .Net », il faut qu'il réponde aux spécifications du **CLR** (voir ci-dessous) et qu'il existe un compilateur pour .Net. Le **CLS** (Common Language Specification) définit l'interface que doit respecter le langage pour attaquer les bibliothèques .Net.

☞ **Interopérabilité des langages** : par exemple, une classe écrite en C# peut être instanciée en VB.Net.

☞ **Interopérabilité** avec COM (ActiveX et anciennes DLL par COM Interop) et Win32 (API Windows).

I.4 NFMK = « .NET FRAMEWORK »

☞ **BCL** (Base Class Library) = **FCL** (Framework Class Library) = mscorlib.dll : **bibliothèque standard de classes** (ainsi que de structures et d'énumérations) de base commune (8 500 classes, chacune composée de dizaines de propriétés et méthodes) fournissant des services sous forme de **Namespaces** (espace de noms) = ensembles de classes = thèmes : System, Data, XML, Drawing, Web,...



BCL repose sur Win32, mais propose une vision objet des API. BCL évité le problème des nombreuses API complexes (typées C) et redondantes.

☞ **CLR** (Common Language Runtime) : **Noyau d'exécution** (= DLL **run-time** équivalent à la JVM = machine virtuelle Java) : chargement des assemblages dans l'espace mémoire du processus, compilation JIT du code CIL.

Base Class Library Support		
Support des threads	COM Marshaler : conversion des paramètres lors de l'appel d'un composant COM	
Vérification de types	Gestion des exceptions	
Moteur de sécurité du code (*)	Moteur de debug	
Compilateur IL → Natif	Gestion du code	Garbage Collector (ramasse-miettes) Gestion d'allocation mémoire
Class Loader (chargement des classes)		

(*) **Sécurité du code** : un composants managé se voit attribuer un niveau de confiance en fonction de son origine (Internet, réseau local ou poste local) définit les droits d'accès de l'application aux ressources (fichiers, base de registre, ...).

Le CLR est normalisé par l'**Ecma** et l'**ISO**.

Le CLR peut être hébergé ...

- par le système d'exploitation (Windows, Pocket PC, Linux, ...),
- par ASP.Net,
- par le SGBD (SQL Server 5, DB2-UDB 8.2 « Stinger »),
- par un composant non managé qui le charge dans son processus, ...

Internet Explorer est un exemple d'application non managée qui héberge le CLR (sous la forme d'une extension MIME). ceci permet d'incorporer des contrôles WinForms dans des documents HTML

Plusieurs versions du CLR peuvent cohabiter sur la même machine (mscorlib.dll / mscorsvr.dll).

☞ **Framework = BCL + CLR**

Version de Fmk	Prix	Description
NFmk SDK	gratuit	Le Kit de développement Microsoft .NET Framework SDK 2.0 comprend des outils, de la documentation et des exemples utiles aux développeurs pour écrire, générer, tester et déployer des applications .NET Framework sur des plates-formes x86
NFmk Redistribuable	gratuit	Pour le déploiement client Windows ou serveur Web. Contient les classes de bases (BCL) et l'environnement d'exécution (CLR)
.Net Compact Fmk	gratuit	Pour périphérique mobile (PDA,...)
Micro Fmk.Net	gratuit	Pour application embarquée « contrainte » (c'est-à-dire disposant de peu de ressources) telles que les télécommandes, les appareils électroménagers,...

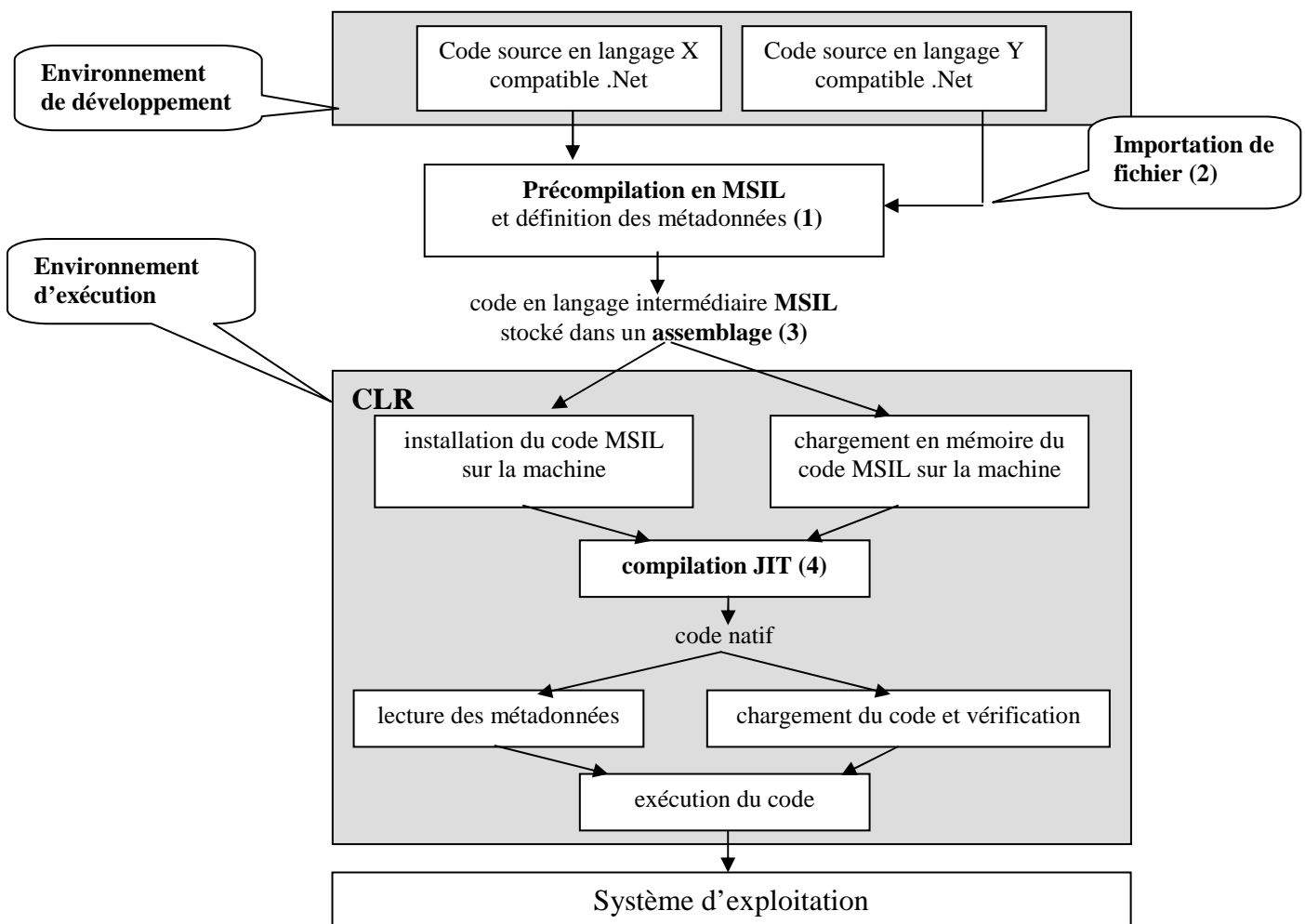
Le NFmk est intégré aux versions récentes de Windows.

I.5 FONCTIONNEMENT D'UNE APPLICATION .NET

Pour du **code non managé** (possible uniquement en C++ et C#), il y a exécution directe du code natif issu de la compilation.

Pour du **code managé**, la traduction du programme s'effectue en deux temps :

Etape	Outil	Format d'origine	Format produit
1- Précompilation	Précompilateur = compilateur spécifique au langage	Code source (C#, VB,...)	Code MSIL (équivalent du bytecode java)
2- Exécution	CLR = machine virtuelle = compilateur JIT + moteur d'exécution	Code MSIL	Code exécutable (natif / plate-forme)



(1) Le **compilateur du langage concerné** traduit le code source en instructions **MSIL** (Microsoft Intermediate Language).

Le MSIL est l'implémentation Microsoft du **CIL** (Common Intermediate language) qui est l'équivalent du bytecode Java. C'est un langage semi-compilé (proche de l'assembleur) qui est commun à tous les langages.

Le compilateur produit également des **métadonnées** : elles ont pour rôle de décrire les types de données en provenance du code source (stockées dans des fichiers au format binaire).

(2) Grâce à l'enregistrement des types dans des métadonnées, il est possible de récupérer un type de données dans un langage autre que celui dans lequel il a été défini.

(3) L'**assemblage** (« **assembly** ») est l'unité de déploiement (EXE ou DLL). Attention, un exécutable « .Net » a besoin du CLR pour s'exécuter.

(4) **Compilation JIT** (Just In Time) des instructions MSIL en code natif (dépendant de la plate-forme d'exécution) permettant d'exécuter le même assemblage sur plusieurs types de machines.

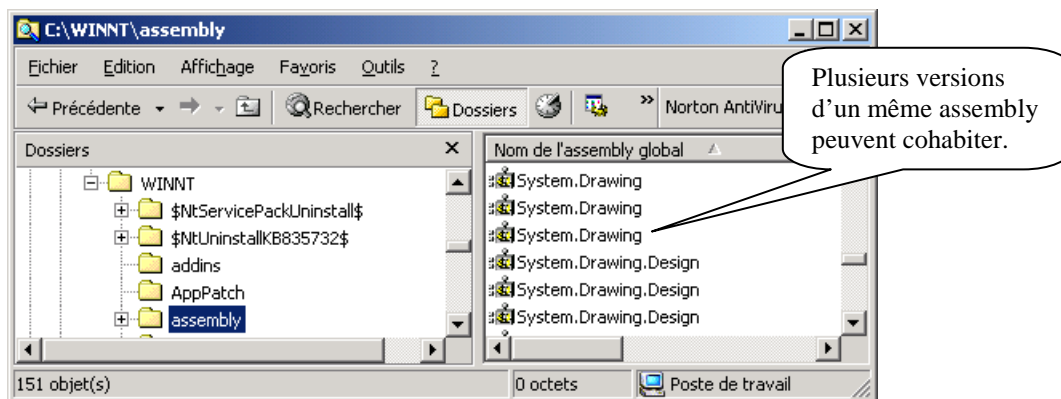
I.6 DEPLOIEMENT D'APPLICATION

☞ L'installation est simplifiée

Technique	Description
Simple copie de fichiers « XCopy »	Copie des fichiers assemblies (EXE ou DLL) : il n'y a plus de problème de DLL (différentes versions peuvent cohabiter), ni de base de registre. De même, la désinstallation peut être effectuée par une simple suppression des fichiers de l'application.
Windows Installer « MSI »	Programme d'installation à distribuer et installer sur le client. Visual Studio proposent les assistants adéquates
Lien hypertexte	Installation via un lien hypertexte vers un emplacement publication (serveur Web, serveur FTP, partage de fichiers, supports CD/DVD). - NTD (No Touch Deployment) de .Net 1 - Click Once de .Net 2

☞ **Stockage des DLL**

- Dans le répertoire de l'application,
- ou bien dans le **GAC** (Global Assembly Cache) de la machine pour les **classes du Framework** et les **DLL partagées** entre plusieurs applications (répertoire WinNT\Assembly). On peut y stocker deux versions d'une même DLL (donc deux fichiers de même nom).



Liste des assemblies : GACUTIL / L | MORE

La CLR recherche d'abord les assemblies dans le GAC.

- **Pas d'inscription dans la base de registre** (le registry n'est plus utile pour l'exécution). Chaque application a son propre **fichier XML de configuration** (DLL utilisées, chaîne de connexion,...)

II. ENVIRONNEMENT DE DEVELOPPEMENT

I.7 ENVIRONNEMENTS DE DEVELOPPEMENT

Environnement	Origine	Date	Fmk	Prix	Langages	Remarques
Compilateur en ligne Exemple : csc.exe	Microsoft	2002		gratuit (fourni avec le Framework)	Un compilateur par langage (exemple : C#)	Compilateur en ligne de commande
Visual Studio 2005 : - Versions Express	Microsoft	2005	2.0	gratuites	- C#, C++, VB.Net, J# - Web Developer (ASP) - SQL Server allégé	Ces versions font entre 35 et 70 Mo
Visual Studio 2005 - Version Standard - Version Professional	Microsoft	2005	2.0	320 € 855 €	C#, C++, VB.Net, J#, ASP.Net	- version française : début 2006 - SP 1 : juin 2006 avec Fmk 2.1
Visual Studio .Net 2005 - Version Team System VS Team Architect VS Team Developer VS Team Test Team System Suite	Microsoft	2006	2.0	- 5 899 € - 5 899 € - 5 899 € - 11 711 €		Offre des outils de gestion de versions, de tests, d'analyse de performance et de gestion de projet
SharpDevelop 2	open source	2006	2.0	gratuit	C#	http://www.icsharpcode.net/opensource/sd/Default.aspx
C# Builder 2006	Borland	2006	1.1	gratuit pour la version personnelle	C#	
Delphi 2006	Borland	2006	1.1	payant	Pascal, C#	
Dreamweaver CS3	Macromedia	2005	1.1	payant	ASP.Net	
Plug-in C# pour Eclipse	Open source				C#	

☞ **Visual Studio** est un IDE (Integrated Development Environment), ou EDI (Environnement de Développement Intégré), commun à tous les langages .Net. Il assure toutes les fonctions liées au développement d'application : édition/compilation de code, debug, maquettage de l'interface utilisateur (fenêtre Windows, page Web, états d'imprimante, ...), lien avec le SGBD, ...

☞ **Visual Studio Team System** est la version haut de gamme de l'IDE

	VS Team Architect	VS Team Developer	VS Team Test
Fonctions	- Concepteur d'application - Concepteur d'infrastructure - Concepteur de déploiement	- Analyse dynamique de code - Analyse statique de code - Profiler de code	- Tests de charge - Tests manuels - Tests web - Gestion de cas de tests
Visual Studio Professional	X	X	X
Client Team Foundation	X	X	X
Visio et modélisation UML	X	X	
Concepteur de classes	X	X	
Test unitaire et couverture de code		X	X
Tests de charge, manuels et Case Management			X

Team System Suite comprend les trois produits **Team Architect** + **Team Developer** + **Team Test**

Team Foundation Server est la partie serveur permettant de gérer les projets, les tâches (besoins, bugs, ...), le versioning, de générer des rapports d'avancement, d'automatiser les builds. Chaque client Visual Studio doit avoir une **CAL** (Client Access Licence) pour accéder à cette partie serveur.

I.8 INSTALLATION

☞ Configuration

- Processeur : 1 GHz (minimum 600 MHz)
- Mémoire : 256 Mo et plus pour les projets volumineux
- Espace disque (avec MSDN) : 1 Go pour le disque système + 3,8 Go pour le disque d'installation
- Lecteur : DVD (minimum CD)
- Vidéo : 1024 x 768 (minimum 800x600)
- Système : Windows 2000 SP4, Windows XP SP2, Windows 2003 SP1

☞ Installation de Visual Studio .Net 2005 Professionnel

1- Installation de **Visual Studio** : ne sélectionner que le langage utilisé

Le programme installe préalablement le .Net Framework 2.0.

Répertoire du Framework : C:\Windows\Microsoft.NET

Répertoire des utilitaires du Framework : C:\Windows\Microsoft.NET\Framework*n° version Fmk*

⚠ En cas de problème d'installation, copier préalablement les fichiers d'installation en local.

2- Installation de **MSDN** (Microsoft Developer network) : ne sélectionner que le langage utilisé

I.9 EXECUTION EN LIGNE DE COMMANDE

- ☞ Créer un simple fichier source avec le Bloc-notes.
- ☞ Compiler à partir de la ligne de commande : **csc.exe** (voir aussi « al.exe »)

Utiliser le fichier de commande fourni avec Visual Studio « Visual Studio Tools – Invite de commande de Visual Studio 2005 ». Celui-ci complète la variable d'environnement PATH pour accéder au compilateur.

The screenshot shows a Windows Command Prompt window titled "Invite de commandes de Visual Studio 2005 - bonjour". The window contains the following text:

```
Setting environment for using Microsoft Visual Studio 2005 x86 tools.
C:\Program Files\Microsoft Visual Studio 2005\Tools\VC\bin>
D:\>cd essai
D:\Essai>csc bonjour.cs
Microsoft (R) Visual C# 2005 Compiler version 8.00.50727.42
for Microsoft (R) Windows (R) 2005 Framework version 2.0.50727
Copyright (C) Microsoft Corporation 2001-2005. Tous droits réservés.
D:\Essai>dir
Le volume dans le lecteur D s'appelle Donnee
Le numéro de série du volume est 6836-8A59

Répertoire de D:\Essai
11/09/2006  15:18    <REP>          .
11/09/2006  15:18    <REP>          ..
16/09/2004  11:47             340 Bonjour.cs
11/09/2006  15:18             3 072 Bonjour.exe
                2 fichier(s)             3 412 octets
                2 Rép(s)    6 925 684 736 octets libres

D:\Essai>bonjour
Bonjour !
```

Three callout boxes provide additional context:

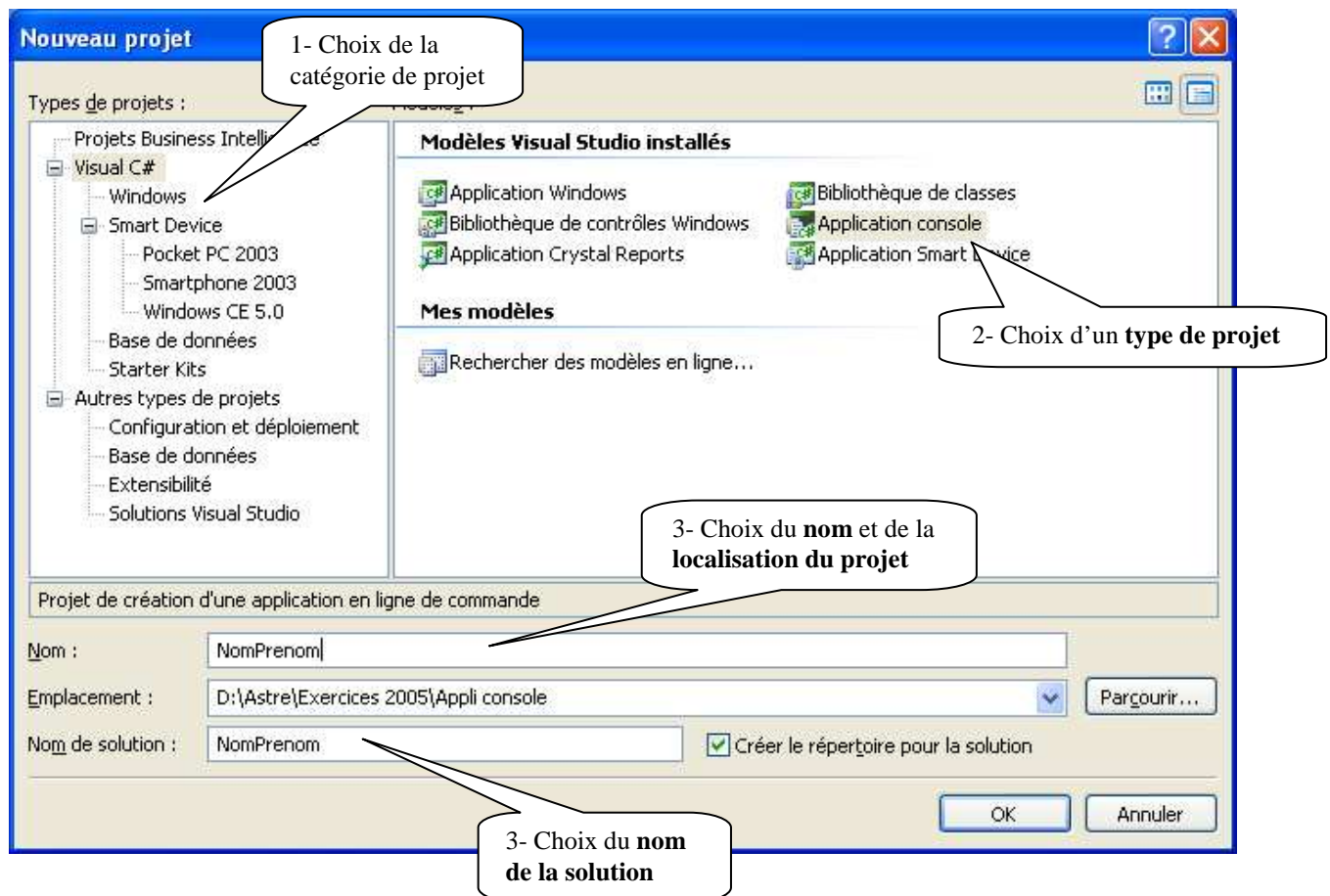
- Top callout: "Compiler avec la commande « csc Bonjour.cs »."
- Middle callout: "Un exécutable est généré."
- Bottom callout: "Le programme peut alors être exécuté."

Exemple de compilation de « File.cs » pour produire « File.exe » : `csc File.cs`

Exemple de compilation de « File.cs » pour créer « My.exe » : `csc /out:My.exe File.cs`

I.10 VISUAL STUDIO : CREATION D'UN NOUVEAU PROJET

- Le menu « Fichier – Nouveau ... » offre le choix de créer un nouveau projet, site web ou fichier.
- Nouveau projet** : les choix possibles dépendent de ce qui a été installé

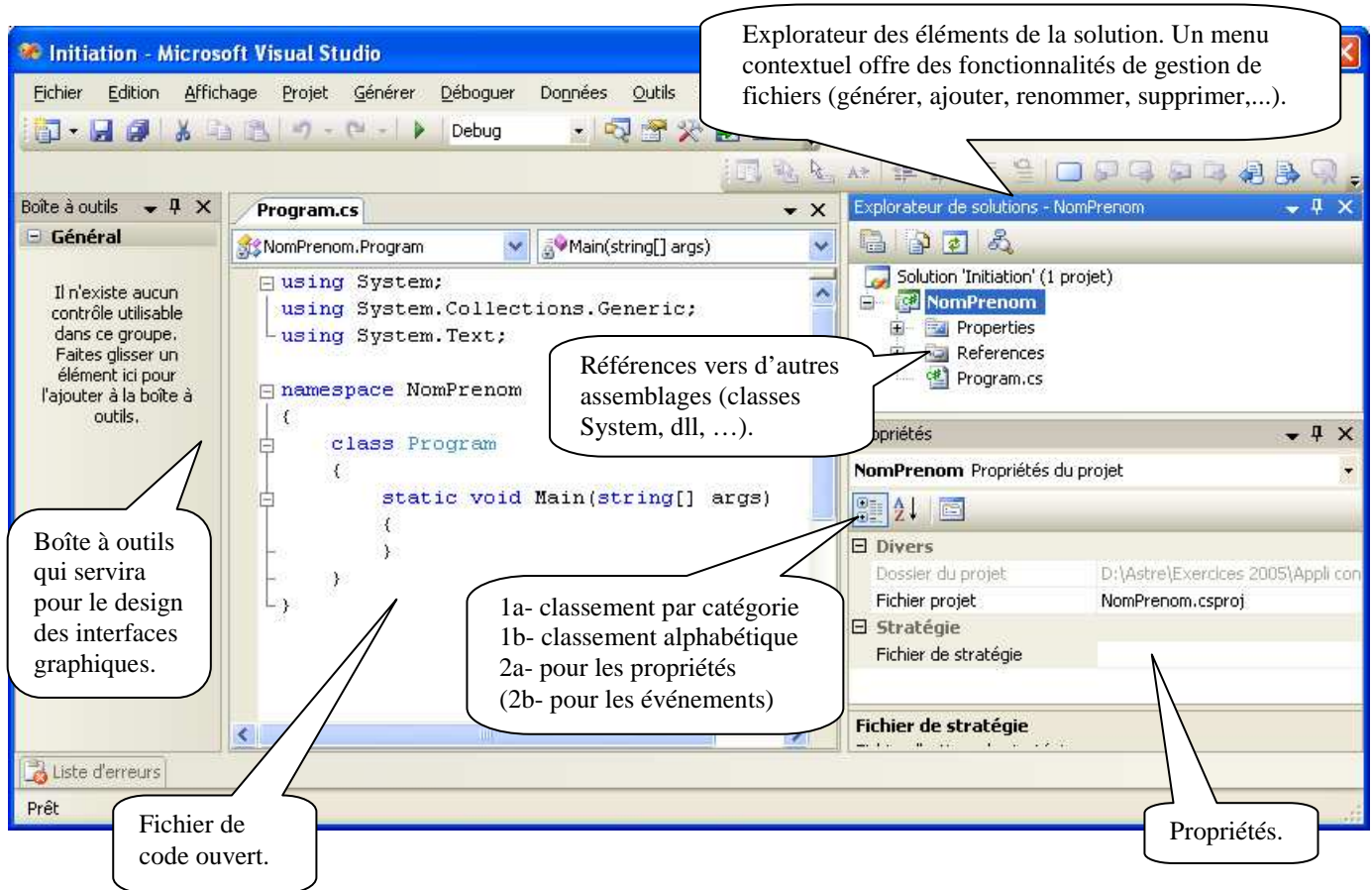


I.11 VISUAL STUDIO : VISUALISATION DU NOUVEAU PROJET

- Sur le disque.

Dossiers	Description
<i>solution</i> <i>projet</i> bin obj properties	Les répertoires bin et obj contiendront les versions exécutables.
Fichiers	Description
<i>solution.sln</i>	Fichier de solution = Organise les projets, les éléments de projet et les éléments de solution au sein de la solution en fournissant à l'environnement les références relatives à leur emplacement sur le disque.
<i>projet.csproj</i>	Fichier de projet C#.
<i>projet.csproj.user</i>	Fichier d' options de projet C#.
AssemblyInfo.cs	Fichier de description de l'assemblage (auteur, version, ...).
Program.cs	Fichier de code qui contient une déclaration de classe avec la méthode Main().

☞ Sous Visual Studio.



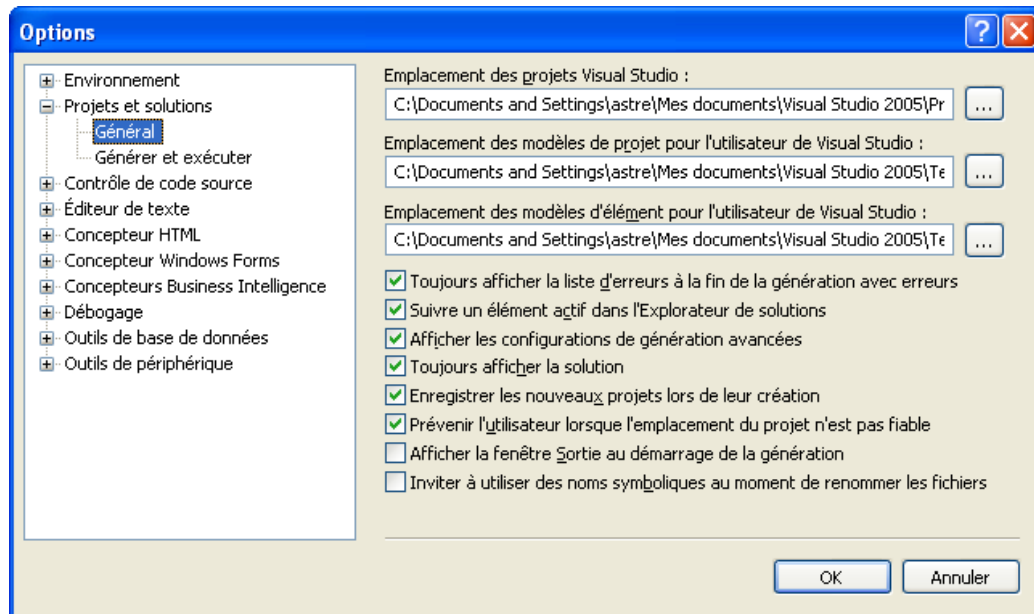
Un **projet** peut contenir **plusieurs répertoires** qui correspondent en général à des **namespaces** distincts

Une **solution** peut contenir **plusieurs projets** : par exemple, un projet d'application Windows, un projet de bibliothèque de classes et un diagramme UML réalisé sous Visio. Si aucune solution n'est ouverte, la création d'un nouveau projet crée automatiquement une nouvelle solution. Dans l'écran ci-dessus, la solution « Initiation » contient un projet qui a été renommé en « NomPrenom ».

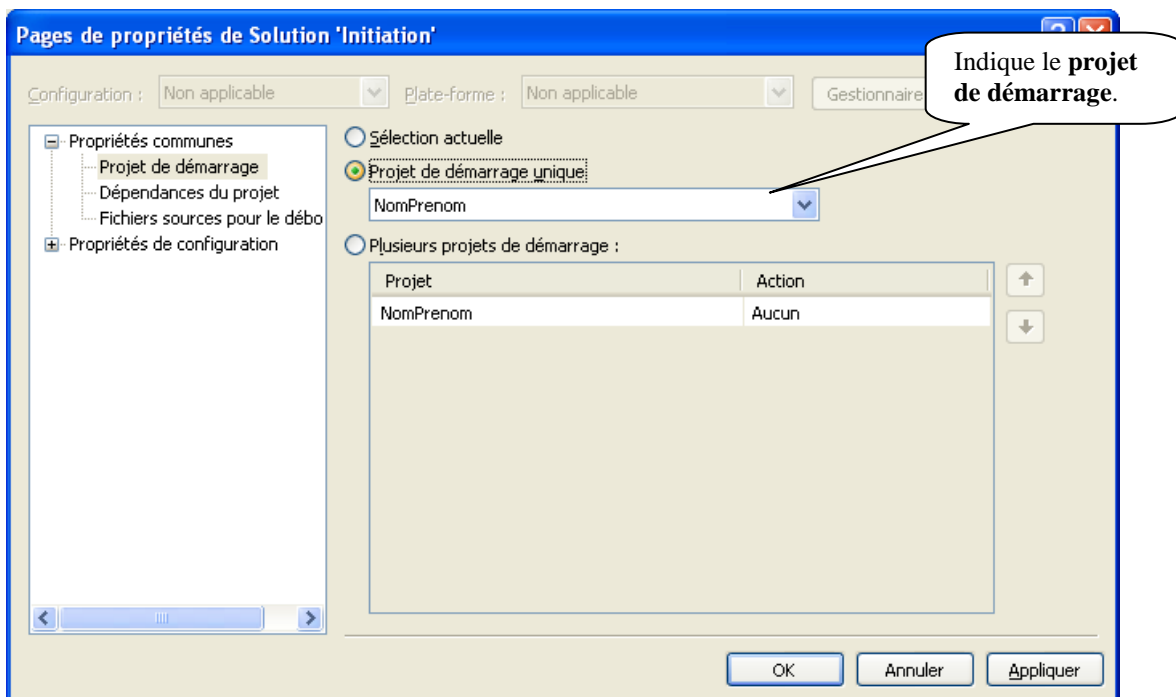
☛ Par défaut, le nom d'une solution mono-projet ne s'affiche pas. Cela peut se modifier avec le menu « Outils – Options – Projets et solutions – Général » - case « **Toujours afficher la solution** ».

I.12 VISUAL STUDIO : PARAMETRAGE

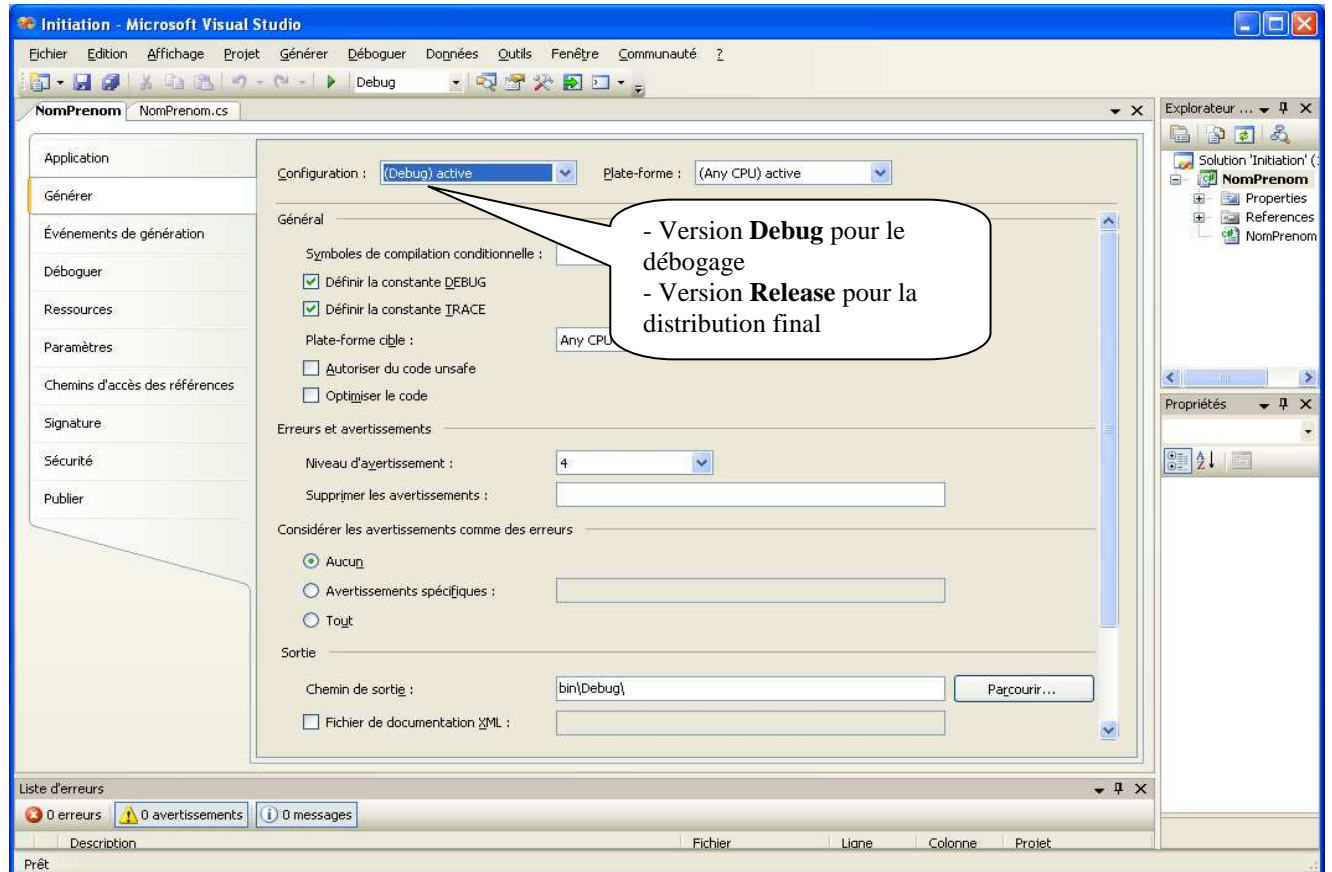
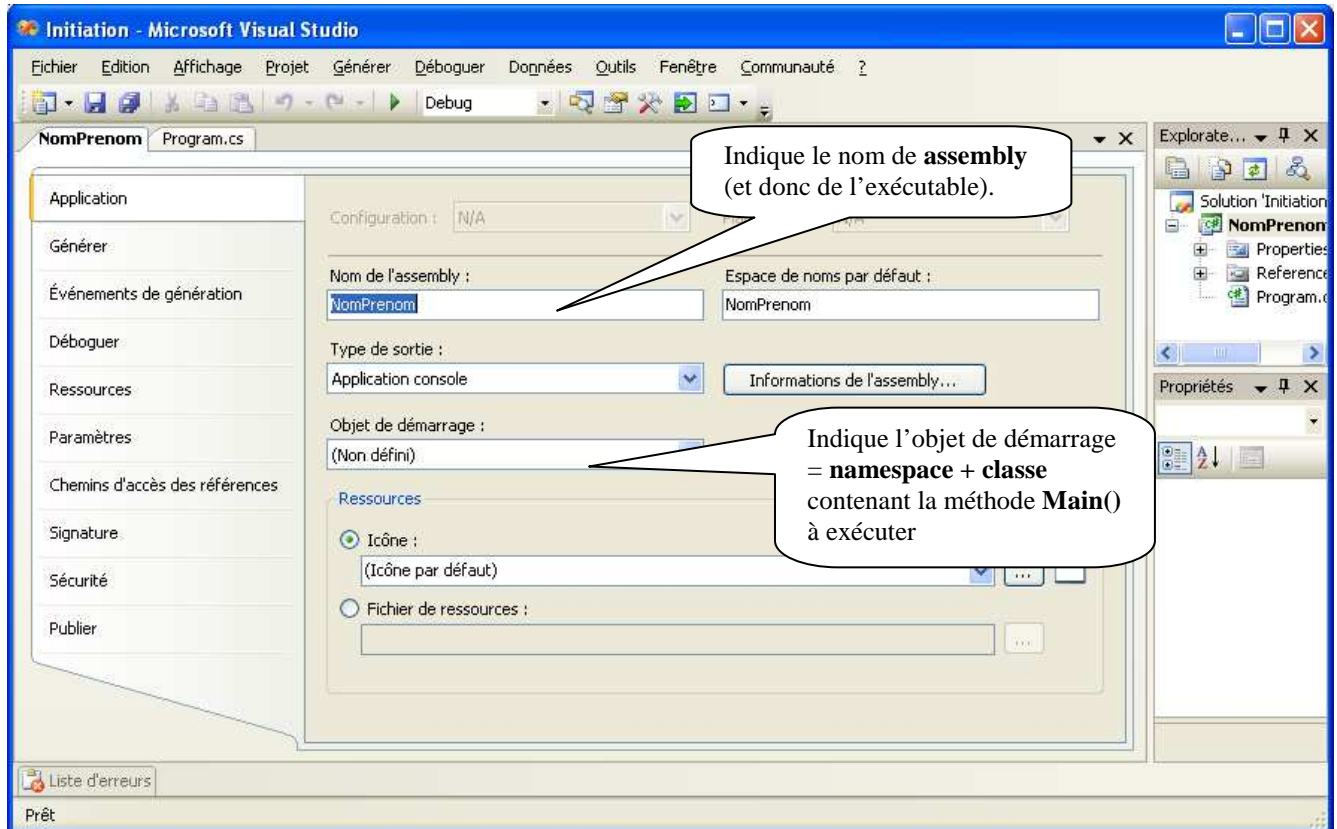
☞ **Paramétrage de l'environnement de développement** : menu « Outils – Options »



☞ **Paramétrage de la solution** : « Explorateur de solutions » cliquer sur la solution - menu contextuel « Propriétés »

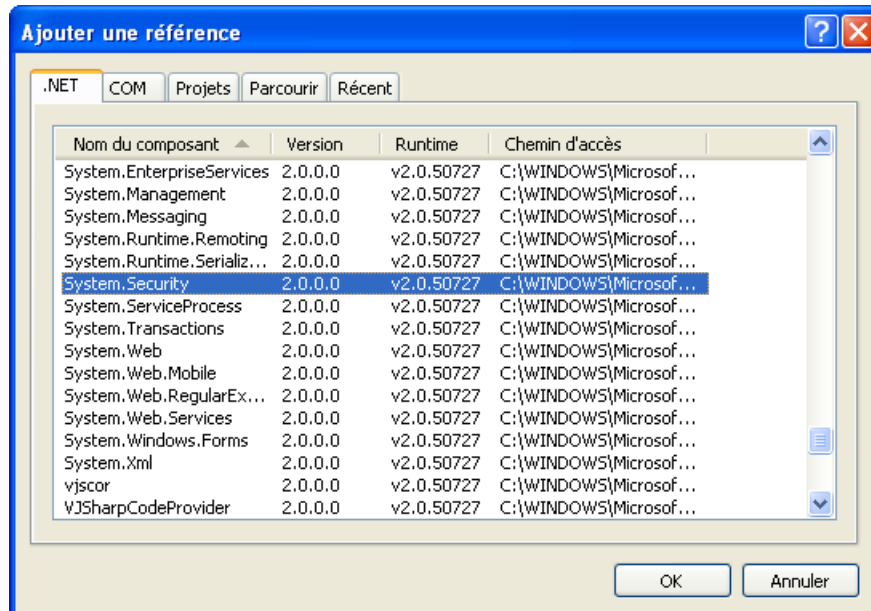


☛ **Paramétrage du projet** : « Explorateur de solutions » cliquer sur le projet - menu contextuel
« Propriétés »



☞ Utilisation d'une bibliothèque dans un projet

- Indiquer au compilateur la localisation de la bibliothèque.
 - Soit avec le compilateur en ligne de commande : utiliser l'option « /r ».
 - Soit dans Visual Studio : « Explorateur de solutions » cliquer sur le dossier « Références » du projet - menu contextuel « Ajouter une référence »



- Référencer la bibliothèque dans le programme source avec « using ».

I.13 VISUAL STUDIO : COMPILATION / EXECUTION

☞ Compilation

Le **compilateur** effectue une **analyse syntaxique du code**, qui délivre les messages d'erreurs éventuels, avant de produire le code exécutable.

```
Sortie
Générer
----- Début de la génération : Projet : Circonference, Configuration : Debug .NET -----

Préparation des ressources...
Mise à jour des références...
Compilation principale en cours...
d:\astre\circonference\circonfil.cs(25,4): error CS0103: Le nom 'Rayon' n'existe pas dans la classe ni dans l'espace de noms 'Circonf.Circ
d:\astre\circonference\circonfil.cs(29,58): error CS0103: Le nom 'Pi' n'existe pas dans la classe ni dans l'espace de noms 'Circonf.Circonf
d:\astre\circonference\circonfil.cs(30,4): error CS0117: 'System.Console' ne contient pas de définition pour 'Readline'

Génération terminée -- 3 erreurs, 0 avertissements
Génération d'assemblys satellites en cours...

----- Terminé -----

Génération : 0 a réussi, 1 a échoué, 0 a été ignoré
```

☞ Le **lancement de l'application** déclenche automatiquement la précompilation (si celle-ci n'a pas été faite).

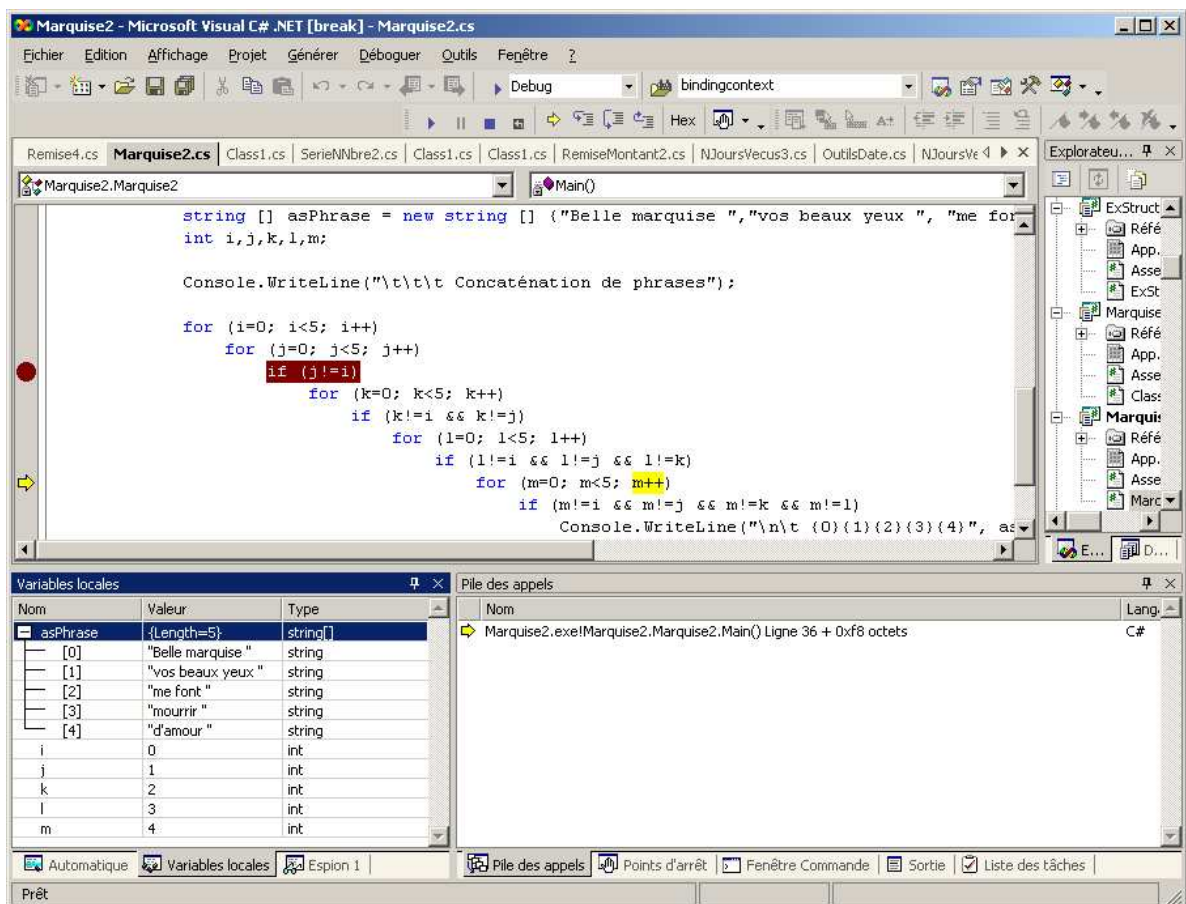
- **Icône Démarrer ►**

- **Menus « Déboguer – Démarrer le débogage » et « Déboguer – Exécuter sans débogage »**

☞ Il est aussi possible de lancer la **précompilation sans exécution** : menus « Générer – Générer/Régénérer solution/projet ».

I.14 VISUAL STUDIO : DEBOGAGE

Le débogueur permet une analyse symbolique du code. Le développeur interprète le comportement du programme en examinant le code, en donnant des valeurs aux variables et en parcourant les différents chemins du programme. Cette méthode est pratiquée plus ou moins implicitement par chaque programmeur (« manuellement » ou à l'aide d'un débogueur)









☞ Configuration en mode « **Debug** » : dans la barre d'outils ou dans les propriétés.

☞ **Passage en débogage**

- Automatique quand une erreur d'exécution survient
- Sur un point d'arrêt posé préalablement (avec la touche « F9 » ou simplement en cliquant dans la marge gauche du code)

☞ Actions de débogage

Action	Menu	Icône	Raccourci	Description
Start	Continuer		F5	Continue l'exécution du programme s'il est en pause
Stop	Arrêter le débogage		Maj+F5	Interrompt l'exécution du programme.
Restart	Redémarrer		Ctrl+Maj+F5	Relance l'exécution au début du programme.
Step Into	Pas à pas détaillé		F11	Exécution pas à pas du detail des fonctions appelées.
Step Over	Pas à pas principal		F10	Exécution pas à pas sans rentrer dans le détail des fonctions.
Step Out	Pas à pas sortant		Maj+F11	Reprise de l'exécution normale jusqu'à ce que le programme sorte de la fonction courante.

☞ Visualisation du contexte : menu « Déboguer – fenêtres »

- Fenêtre **Automatique** : noms/valeurs/types des variables dans l'instruction en cours ainsi que dans l'instruction précédente.
- Fenêtre **variables locales** : noms/valeurs/type des variables dans la portée en cours.
- Fenêtre **espions** : noms/valeurs/types des expressions à évaluer (menu « Débogage – Espion express », touches Ctrl+Alt+Q »).
- Fenêtre **pile des appels** : noms des fonctions dans la pile des appels (avec types et valeurs des paramètres)
- Fenêtre des **points d'arrêt**.
- Fenêtre des **sortie** : affichage de message de l'IDE, de commandes appelées dans la fenêtre Commande, d'outils externes (fichiers .bat et .com), d'instruction applicative de diagnostic (classes Debug ou Trace)
- Fenêtre **commande** : permet d'exécuter directement des instructions telles que modifier la valeur d'une variable.

☞ Modifications tout en déboguant

- Lors d'un simple survol d'une variable, un « tooltip » affiche son contenu. Le débogueur permet de modifier ce contenu.
- De même, Visual Studio autorise la modification des lignes de code tout en déboguant. Il offre même la possibilité de modifier l'ordre d'exécution du code en déplaçant la flèche jaune à gauche vers une autre ligne de code.

☞ Arrêt impératif du programme : pause « || » avec « Ctrl+Alt+Break ».

I.15 AIDE EN LIGNE DU DEVELOPPEUR

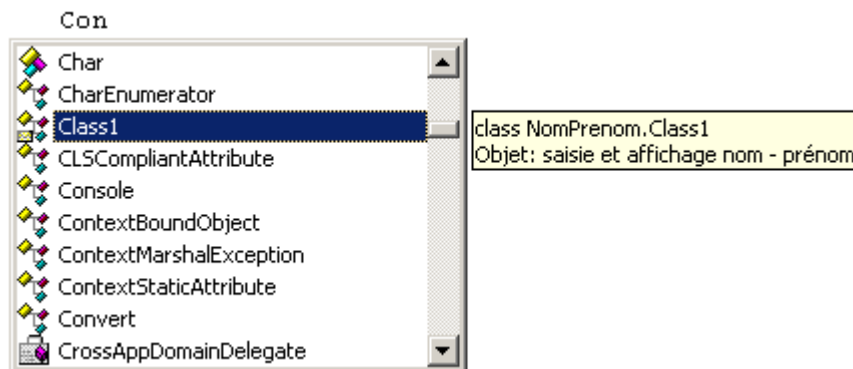
☞ **IntelliSense** (AutoCompletion) propose un ensemble d'options qui facilitent l'accès aux guides de référence du langage lors de la saisie du code.

- **Infos express** : bulle d'aide sur l'élément pointé par la souris

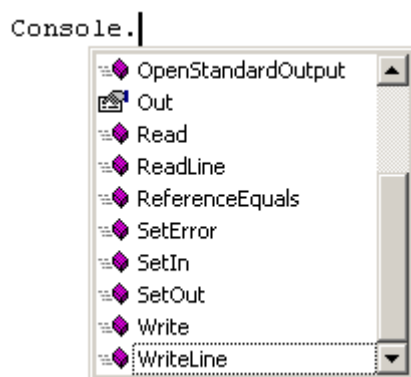
```
Console.WriteLine("\n\n\tNom : ");  
strNom=Console.  
Console.WriteLine  
strPrenom=Console.ReadLine();
```

void Console.WriteLine(string value) (+ 17 surcharges)
Écrit la valeur de chaîne spécifiée dans le flux de sortie standard.

- Saisie semi-automatique lors de frappe du code : taper quelques lettre, puis « Ctrl+Espace ».



- Saisie semi-automatique des membres d'une classe : taper le nom de la classe, puis le point.



- Saisie semi-automatique des paramètres d'une méthode : taper le nom de la méthode, puis une parenthèse ouvrante.

```
Console.WriteLine(  
1 sur 18 void Console.WriteLine(string value)  
value: Valeur à
```

Les flèches haut/bas
permettent de naviguer
entre les différentes
surcharges de la méthode.

III. EXERCICES – INSTALLATION ET ESSAI DE L'ENVIRONNEMENT DE DEVELOPPEMENT

I.16 TRAVAUX PRATIQUES

☐ Installation de l'environnement de développement à partir du serveur de fichiers (ou du DVD) :

☞ Environnement De Développement (EDI) : **Visual Studio**

☞ Aide en ligne : **MSDN**

☞ Outil de modélisation : **Visio**

☐ Consulter la liste des assemblies installés.

☐ Quelle est la version du Framework utilisée ?

☐ Avec le **Bloc-notes**, taper le code source suivant :

```
using System;

class Hello
{
    public static void Main()
    {
        string sReponse="";
        Console.WriteLine("Bonjour !");
        sReponse=Console.ReadLine();
    }
}
```

Sauvegarder ce code sous le nom de « hello.cs », compiler, exécuter en mode ligne de commande (interface texte de type MS/DOS).

☐ Réaliser le même TP avec **Visual Studio** (application en mode Console).

☞ **Compiler**, puis **exécuter** le programme.

☞ Mettre un **point d'arrêt**, puis exécuter **pas à pas** le programme en visualisant le contenu de la variable.

☐ Repérer les principaux éléments d'un projet sous Visual Studio, et rechercher leur utilité dans l'aide.

Elément	Localisation	Utilité ?
Différentes commandes du menu déroulant	en haut	
Différentes icônes de la barre d'outils	en haut	
Editeur de code	au centre	
Fenêtre Explorateur de solutions	à droite	
Fenêtre Affichage des classes	à droite	
Fenêtre de propriétés	à droite	
Fenêtre boîte à outils	à gauche	
Fenêtre liste des tâches	en bas	
Fenêtre liste d'erreurs	en bas	
Fenêtre sortie	en bas	
Fenêtre définition de code	en bas	
Fenêtre Explorateur de serveurs	à gauche	
Explorateur d'objets	au centre	
Options de paramétrage de l'environnement	boîte de dialogue	
Page de propriété de la solution	boîte de dialogue	
Page de propriété du projet	boîte de dialogue	

☐ Effectuer les recherches suivantes dans **MSDN** ...

☞ Configurer MSDN : « Outils – Options – Aide »

- Ne pas inclure les correspondances partielles dans les résultats de recherche.
- N'afficher que les rubriques en français.
- N'effectuer les recherches que sur le contenu local (et non en ligne).

☞ Recherche hiérarchique avec le sommaire : Développement .NET → Kit de développement .NET Framework SDK → Référence de bibliothèque de classes → System → Console → Méthodes → WriteLine

☞ Recherche par mot-clé avec l'index : méthode « WriteLine » pour la Console, puis synchroniser la page trouvée avec la table des matières.

☞ Recherche plein texte avec une expression et/ou plusieurs mots-clés : « "écrire" "flux de sortie" console ».

☞ Mémoriser une page d'aide dans les favoris.

☐ Essai des fonctionnalités de **Visio**

I.17 QUESTIONS

☐ Quel est le principal concurrent du langage C# ?

☐ Quelle est la principale technologie concurrente de .Net ?

☐ Quelles sont les principales architectures clientes cibles pour les applications .Net ?

☐ Quelle est la différence entre une solution et un projet ?

☐ Définir les termes suivants ...

Terme	Définition
.Net	
ActiveX	
ADO	
API	
ASP.Net	
Assembly	
Breakpoint	
C#	
CIL	
CLR	
CLS	
Code managé	
Code natif	
Code non managé	
COM	
Compact Framework	
CSC	
Debug	
DLL	
EDI	
FCL	
Framework	
GAC	
Intellisense	
JIT	
Machine virtuelle	
MSDN	
MSI	
MSIL	
Name Space	
Runtime	
SDK	
Source	
Visual Studio	
Web Services	
WebForms	
WinForms	

IV. ANNEXE : TYPES DE FICHIERS

I.18 PROJET LOCAL

Élément de projet	Extension	Objet de l'élément de projet
Windows Form	.cs	Formulaire destiné à la création d'applications Windows.
Classe	.cs	Fichier de code qui contient une déclaration de classe.
Classe de composant	.cs	Classe pour la création de composants à l'aide du concepteur visuel.
Contrôle utilisateur	.cs	Classe destinée à la création d'un contrôle Windows Form à l'aide du concepteur visuel.
Assistant Formulaire de données	.cs (.aspx pour les projets Web)	Formulaire de données destiné à la création d'applications Windows.
DataSet	.xsd	Fichier destiné à la création d'un schéma XML à l'aide de classes DataSet .
Fichier XML	.xml	Fichier XML vide.
Schéma XML	.xsd	Fichier destiné à la création d'un schéma destiné aux documents XML.
Fichier de code	.cs	Fichier de code vide.
Contrôle Custom	.cs	Classe destinée à la création d'un contrôle Windows Form dessiné par l'utilisateur.
Page HTML	.htm	Page HTML pouvant inclure du code côté client.
Formulaire hérité	.cs	Formulaire reposant sur un Windows Form existant.
Contrôle Web personnalisé	.cs	Classe destinée à la création d'un contrôle serveur ASP.NET..
Contrôle utilisateur hérité	.cs	Nouveau contrôle reposant sur un contrôle Windows Form existant.
Service Windows	.cs	Classe destinée à la création de services Windows.
Classe COM	.cs	Classe pouvant être exposée au modèle COM..
Composant transactionnel	.cs	Classe destinée à être utilisée avec des composants transactionnels.
Fichier texte	.txt	Un fichier texte vide.
Jeu de frames	.htm	Fichier HTML qui contient plusieurs pages HTML.
Fichier XSLT	.xslt	Fichier utilisé pour transformer des documents XML.
Feuille de style	.css	Feuille de style en cascade utilisée pour les définitions de style HTML élaborées.
Classe Installer	.cs	Classe à appeler au moment de l'installation..
Crystal Report	.rpt	Fichier Crystal Report qui publie des données dans un formulaire Windows.
Fichier bitmap	.bmp	Un fichier image bitmap vide qui peut servir à créer des images simples.
Fichier curseur	.cur	Fichier qui permet de créer des curseurs personnalisés.
Fichier icône	.ico	Un fichier image qui permet de créer une icône personnalisée.
Fichier de ressources de l'assembly	.resx	Fichier de ressources d'application au format XML.
Fichier d'informations de l'assembly	.cs	Fichier contenant des informations d'assembly générales.
Fichier de configuration d'application	.config	Fichier utilisé pour définir des paramètres de configuration d'une application.
Fichier JScript	.js	Fichier de script contenant du code JScript.
Fichier VBScript	.vbs	Fichier de script contenant du code VBScript.
Windows Script Host	.wsf	Fichier contenant un script exécuté comme un programme Windows.

I.19 PROJET WEB

Élément de projet	Extension	Objet de l'élément de projet
Web Form	.aspx et .cs	Formulaire destiné à la création d'applications Web.
Service Web	.asmx et .cs	Composant offrant la possibilité d'échanger des messages capables d'interopérer dans un environnement faiblement couplé à l'aide de protocoles standard, tels que HTTP, XML, XSD, SOAP et WSDL.
Formulaire web mobile	.aspx	Formulaire destiné à la création d'applications Web mobiles.
Classe	.vb ou .cs	Fichier de code qui contient une déclaration de classe.
Module (Visual Basic uniquement)	.vb	Fichier permettant de stocker des groupes de fonctions.
Classe de composant	.vb ou .cs	Classe pour la création de composants à l'aide du concepteur visuel.
Assistant Formulaire de données	.aspx (.cs pour les projets locaux)	Formulaire de données destiné à la création d'applications Web. Pour plus d'informations, consultez Assistant Formulaire de données.
DataSet	.xsd	Fichier destiné à la création d'un schéma XML à l'aide de classes DataSet .
Contrôles utilisateur Web	.ascx	Contrôle serveur ASP.NET créé à l'aide du concepteur visuel.
Contrôle Utilisateur Web Mobile	.ascx	Contrôle serveur ASP.NET créé à l'aide du concepteur visuel et utilisé dans une application Web mobile.
Page HTML	.htm	Page HTML pouvant inclure du code côté client.
Jeu de frames	.htm	Fichier HTML qui contient plusieurs pages HTML.
Feuille de style	.css	Feuille de style en cascade utilisée pour les définitions de style HTML élaborées.
Fichier XML	.xml	Fichier XML vide.
Schéma XML	.xsd	Fichier destiné à la création d'un schéma destiné aux documents XML.
Fichier XSLT	.xslt	Fichier utilisé pour transformer des documents XML. Pour plus d'informations, consultez Guide du développeur XSLT.
Contrôle Web personnalisé	.cs	Classe destinée à la création d'un contrôle serveur ASP.NET. ASP.NET
Fichier de code	.cs	Fichier de code vide.
Fichier de découverte statique	.disco	Fichier facultatif qui se comporte comme un mécanisme de découverte pour le service Web XML. Le fichier .disco n'est pas créé automatiquement pour un service Web XML.
Classe d'application globale	.asax	Parfois appelé fichier asax, ce fichier vous permet d'écrire du code pour gérer les événements d'application ASP.NET globaux comme Session_OnStart et Application_OnStart . Le fichier porte le nom global.asax, lequel ne peut être modifié.
Fichier de configuration Web	.config	Fichier utilisé par ASP.NET pour configurer les paramètres Web d'un projet Web. Le fichier porte le nom Web.config, lequel ne peut être modifié.
Fichier texte	.txt	Un fichier texte vide.
Classe Installer	.cs	Classe à appeler au moment de l'installation..
Crystal Report	.rpt	Fichier Crystal Report qui publie des données dans un Web Form.
Fichier bitmap	.bmp	Un fichier image bitmap vide qui peut servir à créer des images simples.
Fichier curseur	.cur	Fichier qui permet de créer des curseurs personnalisés.
Fichier icône	.ico	Un fichier image qui permet de créer une icône personnalisée.
Fichier de ressources de l'assembly	.resx	Fichier utilisé pour modifier et définir des ressources d'application.
Fichier d'informations de l'assembly	.cs	Fichier contenant des informations d'assembly générales.
Fichier JScript	.js	Fichier de script contenant du code JScript.
Fichier VBScript	.vbs	Fichier de script contenant du code VBScript..
Windows Script Host	.wsf	Un fichier de code vide utilisé pour les scripts Windows..

Etablissement référent

Beaumont (63)

Equipe de conception

Michel Astre

Mis en forme par Elisabeth Cattaneo

Remerciements :

Reproduction interdite

Article L 122-4 du code de la propriété intellectuelle.
« toute représentation ou reproduction intégrale ou partielle
faite sans le consentement de l'auteur ou de ses ayants droits
ou ayants cause est illicite. Il en est de même pour la
traduction, l'adaptation ou la reproduction par un art ou un
procédé quelconques. »

Date de mise à jour jj/mm/aa
afpa © Date de dépôt légal mois année



**afpa / Direction de l'Ingénierie 13 place du Générale de Gaulle / 93108 Montreuil
Cedex**

**association nationale pour la formation professionnelle des
adultes**

**Ministère des Affaires sociales du Travail et de la
Solidarité**