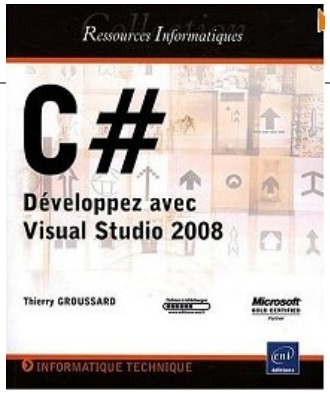
# Présentation de la plateforme .NET

| BTS CIEL | Compétence visée | C08 | CODER |
|---|---|---|---|




*D'après:*
*C# - Développez avec Visual Studio 2008 de Thierry GROUSSARD*

*https://learn.microsoft.com/fr-fr/dotnet/framework/get-started/overview*

*http://fr.wikipedia.org/wiki/Microsoft_.NET*

*http://fr.wikipedia.org/wiki/Framework_.NET*

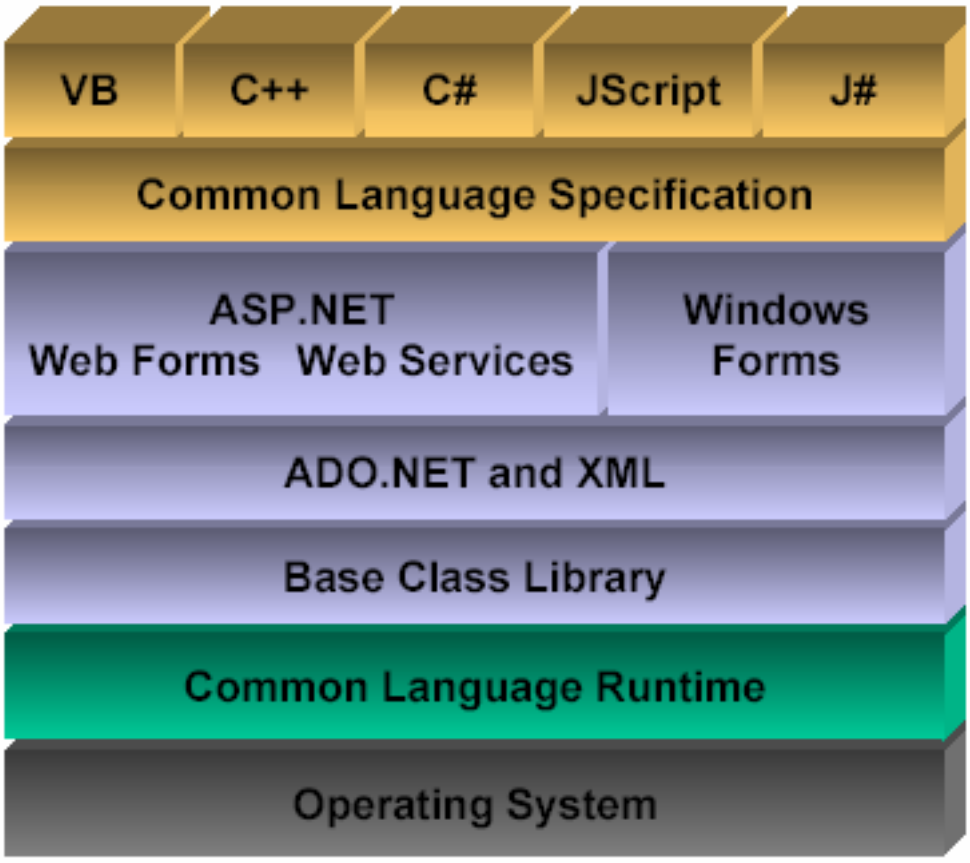
# I) Introduction

La plate-forme .NET fournit un ensemble de technologies et d'outils facilitant le développement d'applications et propose une solution pour pratiquement tous les types d'applications :

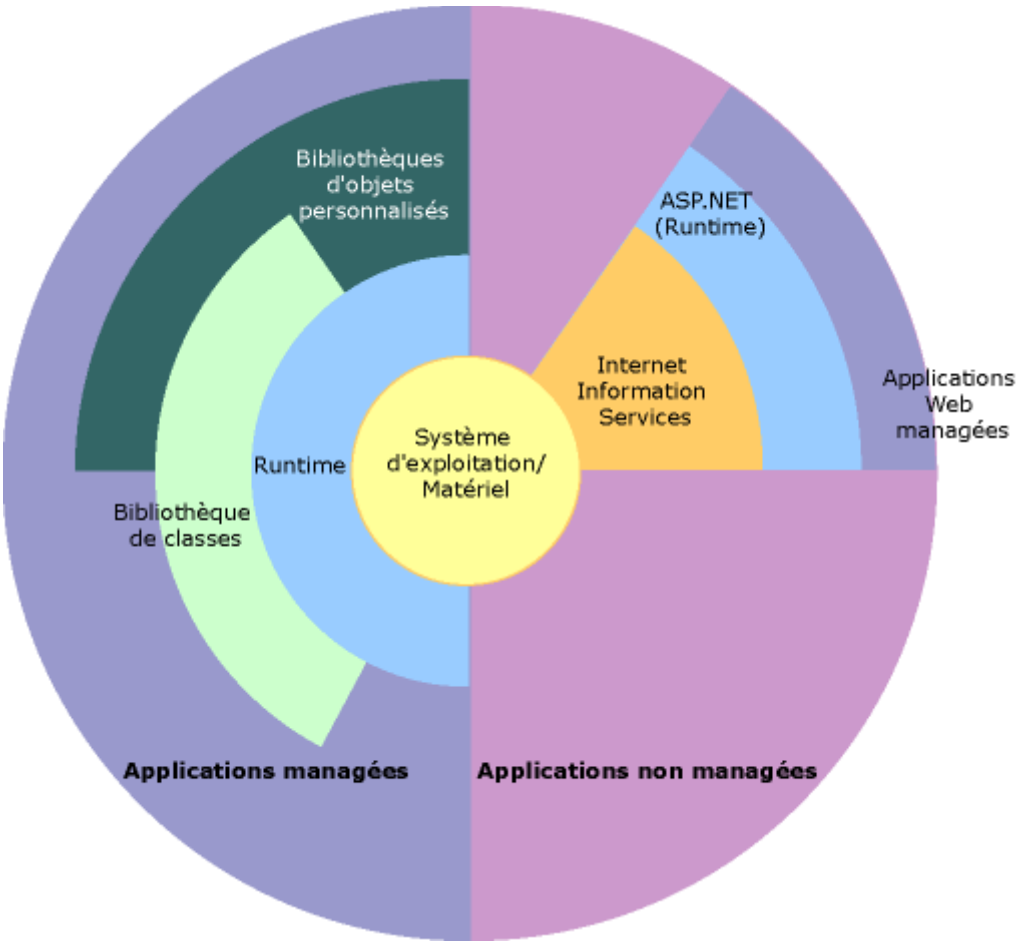
- applications Windows classiques ;
- application Web ;
- services Windows ;
- services Web.

Tous ces types d'applications sont réalisables grâce à un élément essentiel : le Framework .NET. Ce Framework prend en charge, par l'intermédiaire de nombreuses couches logicielles superposées, l'intégralité de la vie d'une application, du développement jusqu'à l'exécution. Le framework doit être hébergé par un système d'exploitation avec lequel il va interagir. Le premier système permettant de l'accueillir est bien sûr Windows mais d'autres versions sont disponibles permettant l'adaptation de la plate-forme .NET à des systèmes tels Linux ou Unix.

*En programmation informatique, un framework est un kit de composants logiciels structurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel (architecture).*



Le framework contient deux éléments principaux : le Common Language Runtime et la bibliothèque de classes du .NET Framework.
Le Common Language Runtime est la base du .NET Framework. Le runtime peut être considéré comme un moteur d'exécution qui gère l'exécution du code et assure également la gestion de la mémoire. Le code pris en charge par le Common language Runtime est appelé **code managé**.
La bibliothèque de classes est un ensemble de classes pouvant être utilisé pour le développement de tout type d'application.




# 1) Bibliothèques de classes .NET

Il y a trois couches au-dessus du Common Language Runtime (CLR), chacune comportant des classes apportant des fonctionnalités de plus en plus évoluées.

## Base Class Library

Dans la première couche se trouve la Base Class Library (BCL), rassemblant des classes permettant les manipulations de chaînes de texte, la gestion des entrées/sorties, des communications réseaux, des processus légers et le design des interfaces graphiques.

## Classes d'accès aux données

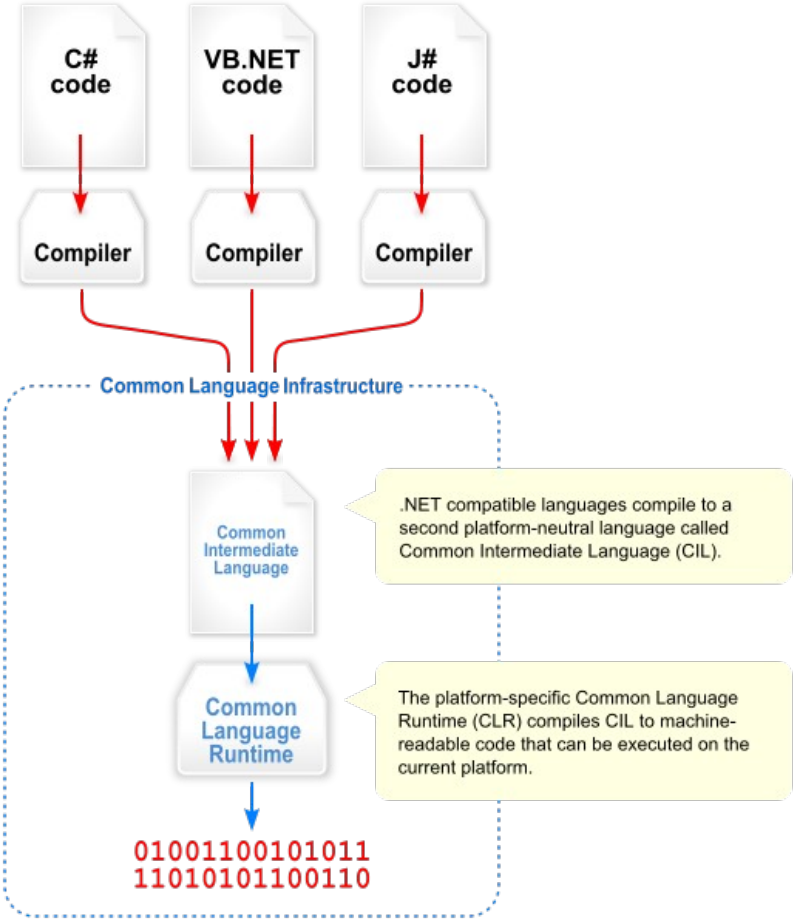
La deuxième couche est composée de deux bibliothèques de classes d'accès aux données. Tout d'abord, ADO.NET, s'élevant sur les bases de l'ancien ADO (ActiveX Data Objects) utilisé par les développeurs ASP, et permettant l'accès sous format XML aux interfaces de bases de données.

## XML Web Services, Web Forms, Windows Forms

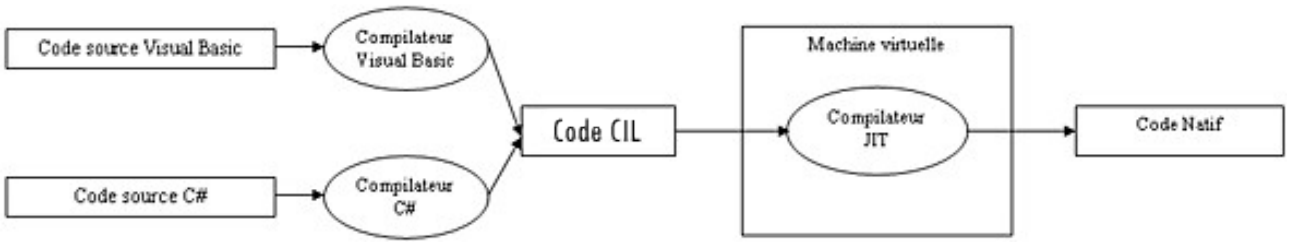
La dernière couche, la plus élevée, est utilisée pour la création de services web, de pages Web, et d'applications Windows. Les deux premiers forment ASP.NET, là aussi une référence à l'ancien langage de programmation Web.

## 2) Principe de fonctionnement du Common Language Runtime

Les langages de programmation du framework sont compilés dans un langage intermédiaire appelé Common Intermediate Language, ou CIL (anciennement connu sous le nom de Microsoft Intermediate Language, ou MSIL). Ce langage n'est pas interprété, mais subit une compilation à la volée et une compilation au niveau de la Common Language Runtime (CLR). La CLR est l'implémentation de la CLI.
Le framework .Net repose sur la Common Langage (ou CLI). Son but est de fournir un langage indépendant de la plate-forme, aussi bien pour le développement que pour l'exécution. Elle inclut des fonctions pour gérer les erreurs, le ramasse-miettes, la sécurité et l'interopérabilité avec les objets COM.
L'implémentation de la CLI par Microsoft est appelée Common Language Runtime (ou CLR).

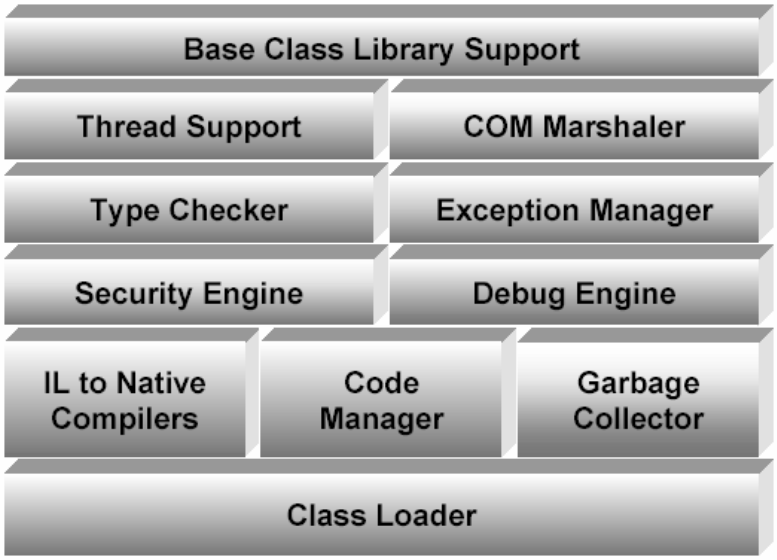



Lors de l'exécution de l'application, ce code est pris en charge par la machine virtuelle qui en assure la traduction en instructions spécifiques pour le microprocesseur de la machine. Cette traduction n'est pas effectuée, en bloc dès le début de l'application, mais uniquement au fur et à mesure des besoins. En effet, pourquoi perdre du temps à traduire du code CIL, s'il n'est jamais utilisé par la suite. C'est pour cette raison que le compilateur utilisé pour cette traduction s'appelle compilateur Just In Time (JIT).
Les avantages de cette solution sont évidents car pour exécuter une même application sur plusieurs plates-formes matérielles et ou logicielles, il suffit d'obtenir la machine virtuelle capable d'effectuer la traduction. Cette machine virtuelle est disponible pour tous les systèmes Microsoft. Le projet Mono propose une version de la machine virtuelle pour les plates-formes suivantes : Linux, Mac OS X, Sun Solaris, …



## 3) Les services du Common Language Runtime

La machine virtuelle ne se contente pas d'effectuer la traduction du code. Le code CIL est aussi appelé **code managé**, ce qui sous-entend qu'un certain nombre d'opérations supplémentaires seront réalisées sur le code au moment de l'exécution. La figure ci-après reprend l'ensemble des fonctionnalités disponibles dans le Common Language Runtime.




**Class Loader**

Il gère le chargement en mémoire des instances de classes.

**IL To Native Compilers**

Il convertit le code intermédiaire (MSIL) en code natif.

**Code Manager**

Il gère l'exécution du code.

**Garbage Collector**

Il assure la gestion de la mémoire en surveillant les instances de classes qui ne sont plus accessibles.

**Security Engine**

Il permet la vérification de l'identité de la personne demandant l'exécution du code et accepte ou non cette exécution, en fonction des autorisations accordées.

**Debug Engine**

Il permet le débogage de l'application, en prenant en charge par exemple l'exécution en pas à pas du code.

**Type Checker**

Il surveille l'utilisation de variables non initialisées et les conversions entre des variables de type différent.

**Exception Manager**

Il fournit la gestion structurée des exceptions en liaison avec Windows Structured Exception Handling (SEH). Cette technique permet une gestion individuelle de chaque exception plutôt qu'une gestion globale.

**Thread Support**

Il propose un ensemble de classes permettant la réalisation d'applications multithread.

**COM Marshaler**

Il permet de traduire des appels vers des composants COM, assurant par exemple la conversion des types de données.

**Base Class Library Support**

Il fournit l'accès aux services disponibles sur le système d'exploitation hôte.

## 4) La Base Class Library

Le Framework .NET met à la disposition des développeurs un ensemble d'outils lui permettant d'obtenir une solution rapide à une majorité de problèmes rencontrés lors de la réalisation d'une application.
Ces outils sont disponibles sous forme de classes. À l'inverse des bibliothèques de code des langages de la génération précédente, qui n'étaient qu'une liste interminable de procédures ou fonctions, la bibliothèque de classes est organisée sous forme d'une structure hiérarchisée. L'élément essentiel de cette hiérarchisation est l'espace de nom (Namespace). Il permet le regroupement logique de classes ayant des points communs. Par exemple, on retrouve dans le namespace System.Data toutes les classes utilisables pour accéder à une base de données.
Cette bibliothèque de classes est bien sûr indépendante d'un quelconque langage de programmation. Elle permet donc

le mélange de différents langages au cours du développement d'une application. Elle est également parfaitement intégrée à Visual Studio, ce qui nous procure un confort d'utilisation appréciable avec des outils comme **Intelissence**. Comme cette librairie est orientée objet, elle est facilement extensible par le biais de relations d'héritage.
La bibliothèque contient une quantité impressionnante d'espaces de nom et de classe, tant et si bien que, au cours de vos développements avec Visual C#, il y a de fortes chances pour que vous n'utilisiez jamais certains d'entre eux.
Les espaces de noms les plus utilisés sont les suivants :

#### System

C'est l'espace de nom racine pour les types de données dans le Framework .NET. Il contient notamment la définition de la classe Object, qui est l'ancêtre de tous les types de données du Framework .NET.

#### System.Windows

Il contient l'ensemble des éléments permettant la création d'interfaces utilisateurs Windows.

#### System.Web

Il contient toutes les ressources nécessaires pour la création d'applications Web, avec par exemple, les classes de la technologie ASP.NET ou les classes utilisables pour la création de services Web XML.

#### System.data

Il contient un ensemble de classes spécialisées dans l'accès aux bases de données, avec le support de ADO.NET.

#### System.Xml

Le langage Xml est devenu omniprésent et cet espace de nom contient les classes assurant la manipulation de documents Xml.

La bibliothèque de classes .NET Framework est une collection de types réutilisables qui s'intègrent parfaitement au Common Language Runtime. La bibliothèque de classes est orientée objet et fournit des types à partir desquels votre propre code managé peut dériver des fonctionnalités. Les types .NET Framework n'en sont que plus faciles à utiliser et le temps d'apprentissage des nouvelles fonctionnalités du .NET Framework s'en trouve réduit. De plus, les composants tiers peuvent s'intégrer parfaitement aux classes du .NET Framework.

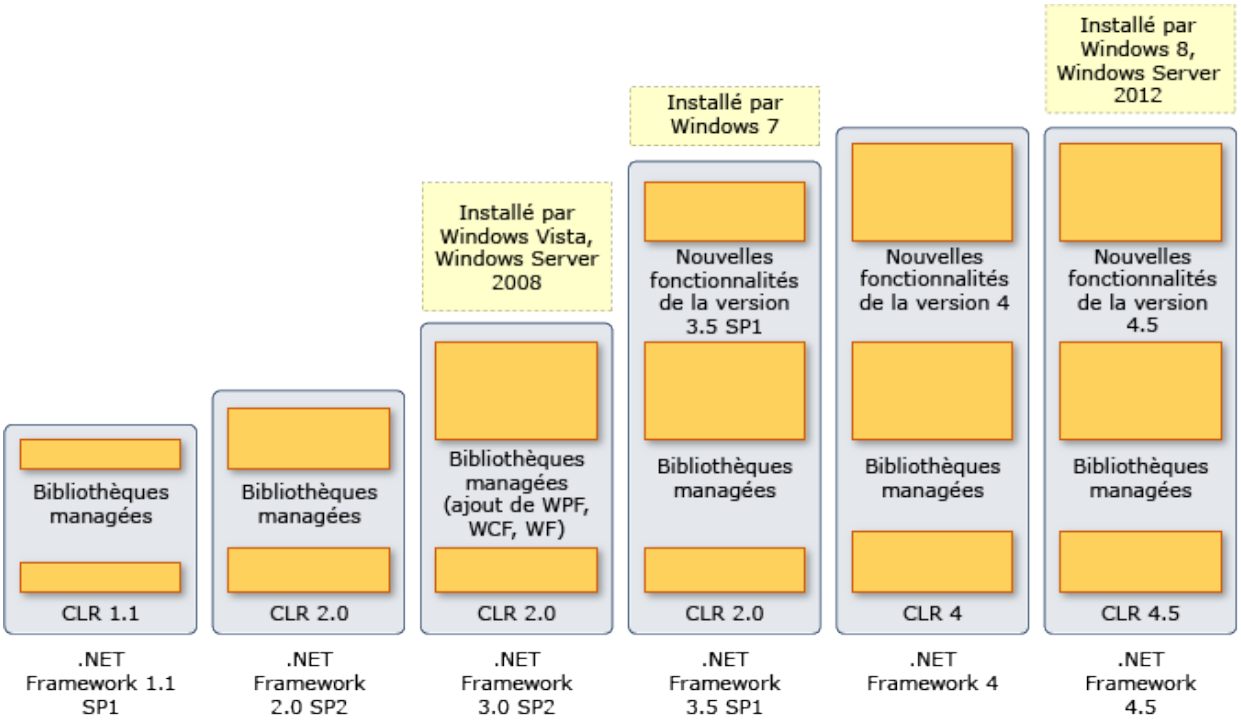
Par exemple, les classes de collection du .NET Framework implémentent un jeu d'interfaces que vous pouvez utiliser pour développer vos propres classes de collection. Vos classes de collection s'intégreront parfaitement aux classes dans le .NET Framework.

Comme pour toute bibliothèque de classes orientée objet, les types .NET Framework vous permettent d'accomplir un éventail de tâches courantes de programmation y compris des tâches comme la gestion de chaînes, la collection de données, la connectivité de bases de données, et l'accès aux fichiers. En plus de ces tâches courantes, la bibliothèque de classes comprend des types qui prennent en charge une variété de scénarios de développement spécialisé. Par exemple, vous pouvez utiliser le .NET Framework pour développer les types d'applications et de services suivants :

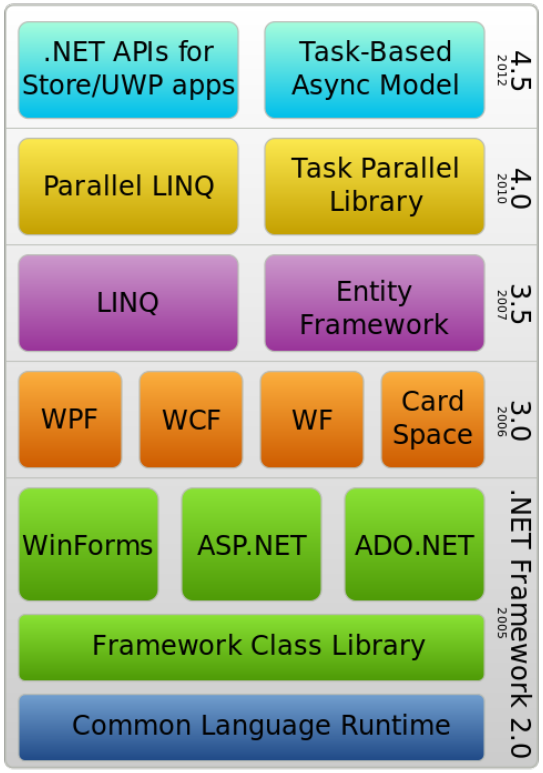
- Les applications consoles. Consultez Génération d'applications de console dans le .NET Framework.
- Les applications GUI Windows (Windows Forms). Consultez Windows Forms.
- Les applications Windows Presentation Foundation (WPF). Consultez Windows Presentation Foundation.
- Les applications ASP.NET. Consultez Développement d'applications Web avec ASP.NET.
- Les services Windows. Consultez Introduction to Windows Service Applications.
- Les applications orientées service qui utilisent Windows Communication Foundation (WCF). Consultez Développement d'applications orientées service avec WCF.
- Les applications prenant en charge les flux de travail qui utilisent Windows Workflow Foundation (WF). Consultez Building Workflows in the .NET Framework.

Par exemple, les classes Windows Forms sont un ensemble complet de types réutilisables qui simplifient grandement le développement GUI Windows. Si vous écrivez une application Web Form ASP.NET, vous pouvez utiliser les classes Web Forms.

## 5) Les versions et évolutions de la plate-forme .NET




Framework 4.6, 4.7 installé par Windows 10 et Windows Serveur 2016/2019, la 4.8 avec Windows 11 et Windows Serveurs 2022...




Pour connaître quelle(s) version(s) du Framework est(sont) installée(s) sur un poste Windows, il suffit de consulter le répertoire `C:\WINDOWS\Microsoft.NET\Framework` (chaque sous-répertoire non vide correspondant à une version présente)