Architecture

Architecture d’un programme
,différence entre les langages objets
principe de programmation objet,

Env.dev: SDK , Visual Studio…

La différence entre les langages

- Entre C# et VB.net:
  C# est un langage sensible a la casse
  différence de syntaxe: { } vs End

  vb.net est aussi un langage objet
  les 2 utilisent les FCL

  qq. différence :IsNumeric est un méthode appartenant a VB.net qu'on ne trouve pas en c# repris de VB6(Microsoft.VisualBasic)

  Dans les collections , redimensionner possible en vb.net et pas en c# (redim preserve …)

La différence entre les langages

- Entre C# et Java:

  c’est le java de Microsoft

  JIT Compiler = JVM

  tous héritent d object

  Pas de surcharge d’opérateur en Java

  en java on peut importer juste une classe et pas obligatoirement tout le package

La différence entre les langages

- Entre C# et C++:

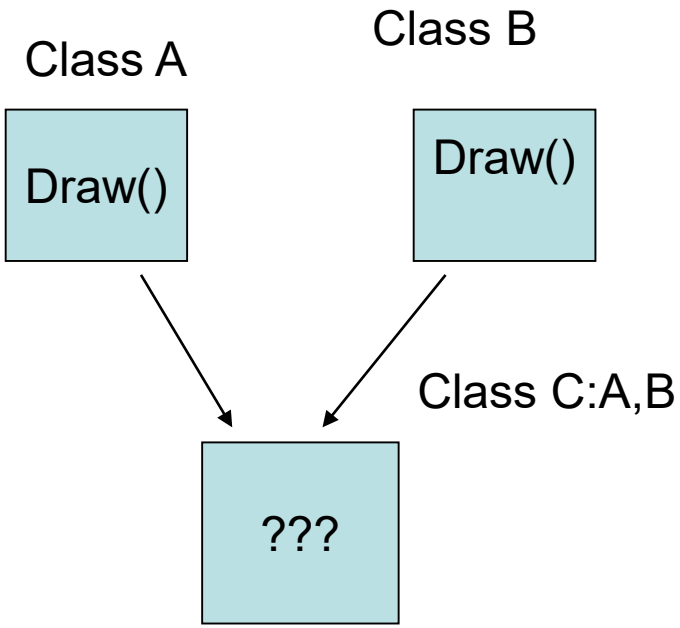
  plus de pointeur(on peut toujours les utiliser)

  plus de multiple héritage(un équivalent est l'interface)

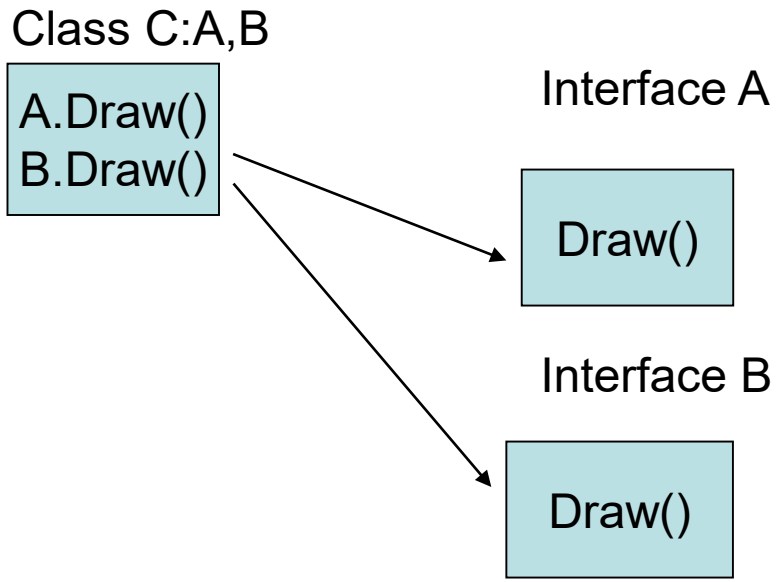
  nouveauté :les properties get et set

Entre C# ou vb.net et C++

- C++



- C#



L’environnement du SDK

- C’est un environnement de développement avec des outils ,de la doc et des compilateurs

- Il faut « Loader« l’environnement du sdk avec ses compilateurs:

  cd > Program Files \ Microsoft Visual Studio.Net \ Common7 \ Tools \

  cd>vsvars32.bat (a exécuter a partir du dos)

- On peut écrire un 1e programme grâce a un éditeur quelconque comme edit ou notepade

L’environnement du SDK:un 1er programme

```
class HelloWorld  {
   static void  Main()  {
   Console.WriteLine(“Hello World!”);}}
```

- Ce programme sera sauvegardé en tant que .cs

- Ensuite il faudra le compiler grâce a :
  csc toto.cs est la commande qui permet de lancer le compilateur c#(CS Compiler) a la suite de quoi est crée un .exe

L’environnement du SDK

- Pour l’exécuter tout simplement 2 clicks sur le .exe qui fait appel au JIT Compiler
- C’est le programme le plus basic , tout doit être dans une classe a l'exception du namespace
- Les méthodes peuvent être dans une classe ou plusieurs,une structure ou une interface

L’environnement du SDK

- il n’y a qu’un seul Main qui sera noté avec un grand M etc…

- Le nom de la classe et le nom du fichier de classe ne doivent pas obligatoirement être le même(différent en java)

Les executables du framework

- Visual C# .NET Compiler (csc.exe)
  - Compilateur de sources C#
- Visual J# .NET Compiler (vjc.exe)
  - Compilateur de sources J#
- Visual Basic .NET Compiler (vbc.exe)
  - Compilateur de sources VB.NET
- JScript .NET Compiler (jsc.exe)
  - Compilateur de sources JavaScript
- CLR Native Image Generator (ngen.exe)
  - Compilateur permettant de créer une image native à partir d'un assembly managé

…

SDK

- C'est l'environnement minimum pour développer une application C#
  - Compilateurs
  - .NET Framework class library
  - Téléchargeable à l'adresse : http://msdn.microsoft.com/netframework

les outils du sdk:

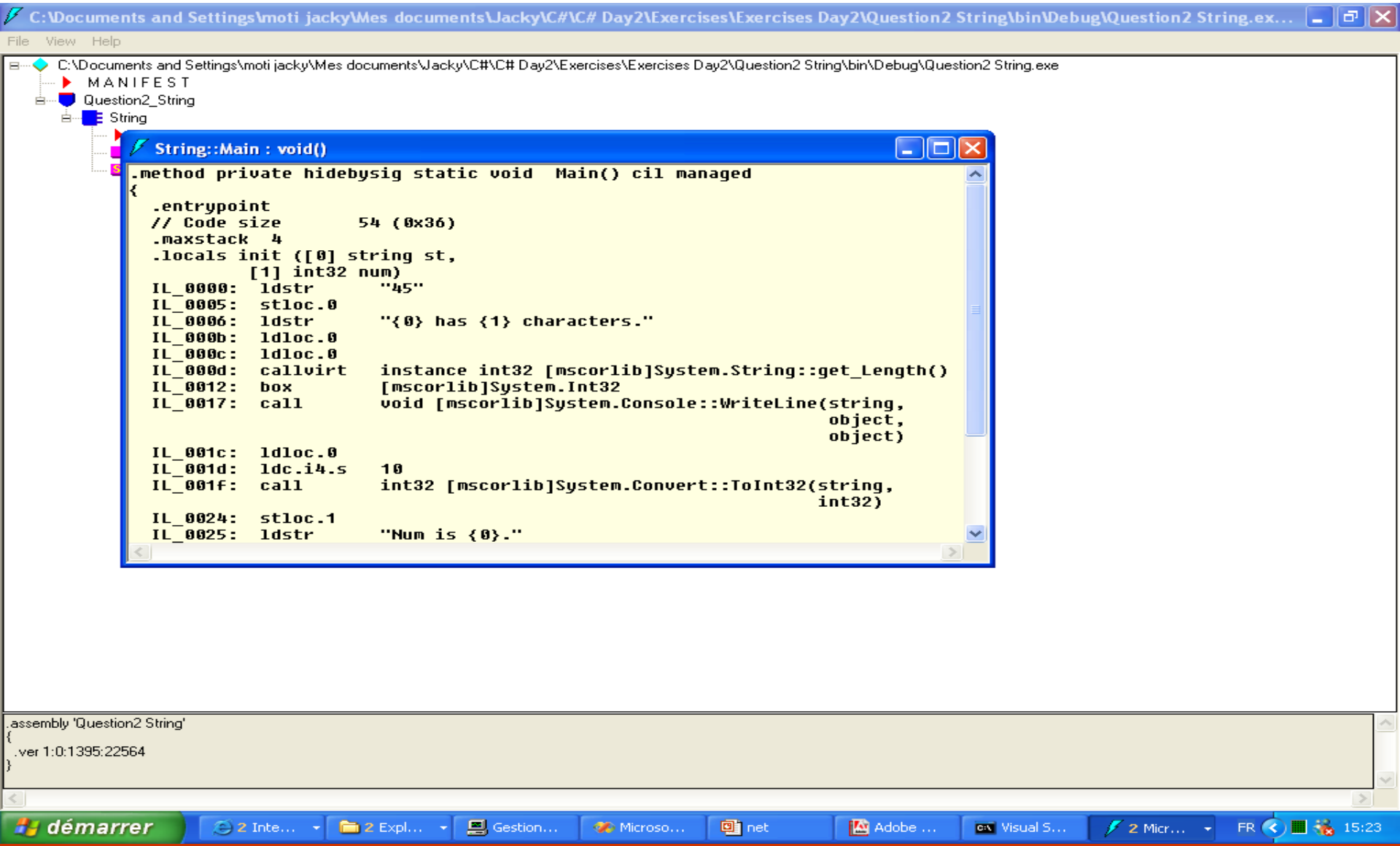
- MSIL Disassembler (Ildasm.exe)
  - Désassembleur MSIL
- Outil XML Schema Definition Tool (Xsd.exe)
  - Génère des schémas XML
  - Génère les classes du CommonLanguage Runtime et les classes à partir d'un fichier de schéma XSD
- Outil Web Services Discovery Tool (Disco.exe)
  - Découvre les URL des services Web XML situés sur un serveur Web
- Outil Web Services Description Language Tool (Wsdl.exe)
  - Génère le code pour les services Web XML et les clients des services Web XML

Les outils

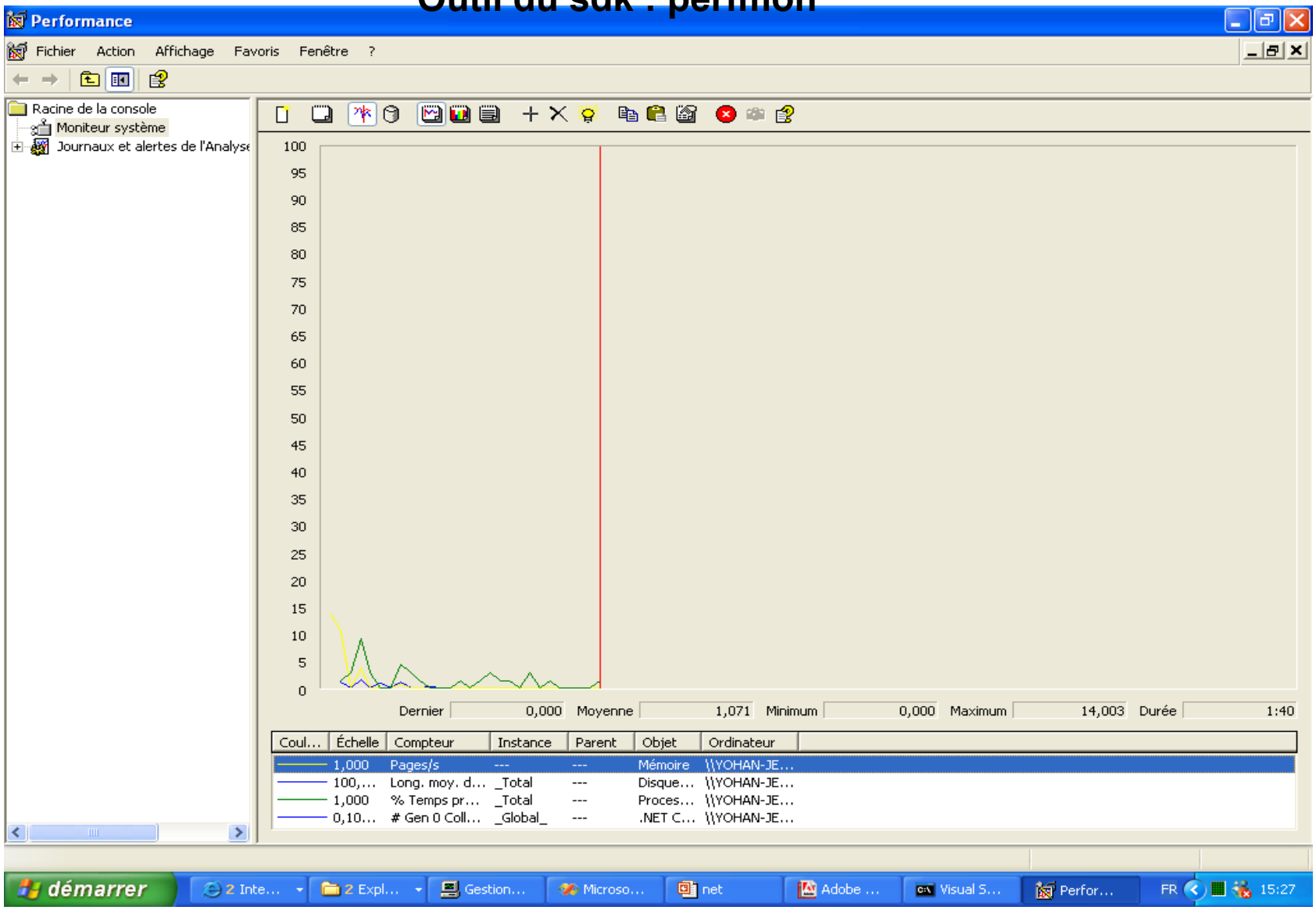
- *Integrated Development Environment*
- Les IDE du marché
  - Visual Studio .NET (Microsoft)
  - WebMatrix (Microsoft)
  - SharpDevelop
  - Plug-in C# pour Eclipse
  - **X-develop** (Omnicore)

- WebMatrix
  - Une version allégée et gratuite de Visual Studio.
  - Développement d'applications ASP.NET
  - WebMatrix propose également un mini serveur Web pour les personnes ne disposant pas d'IIS.
- MSDE (Microsoft SQL Server Desktop Edition)
  - Permet d'avoir accès à un SGBD performant et gratuit
  - Pas d'interface graphique, mais son noyau est bien celui de SQL Server
  - Redistribuable gratuitement avec toute application.

Outil du sdk : ildasm



Outil du sdk : perfmon



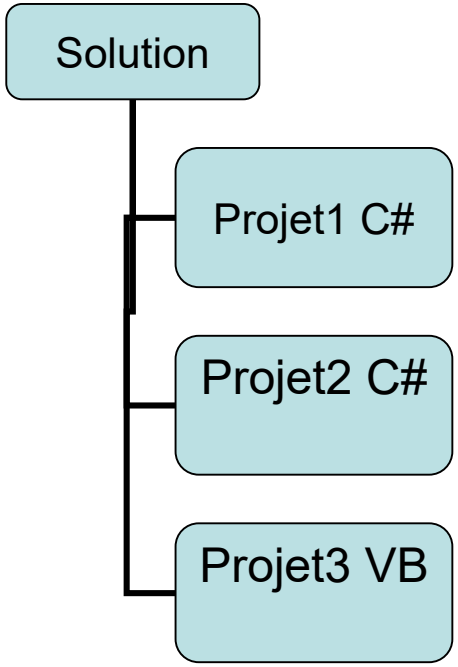
L’environnement de Visual Studio

- Architecture des projets:

.un projet appartient forcement a une solution

.une solution peut avoir plusieurs projets

.lorsqu’on exécute un projet la compilation se fait sur tout les projets de la solution



L’environnement de Visual Studio

- Un solution est un fichier .sln
- Un projet c# .csproj ou vb .vbproj
- Le fichier .cs est un fichier de class

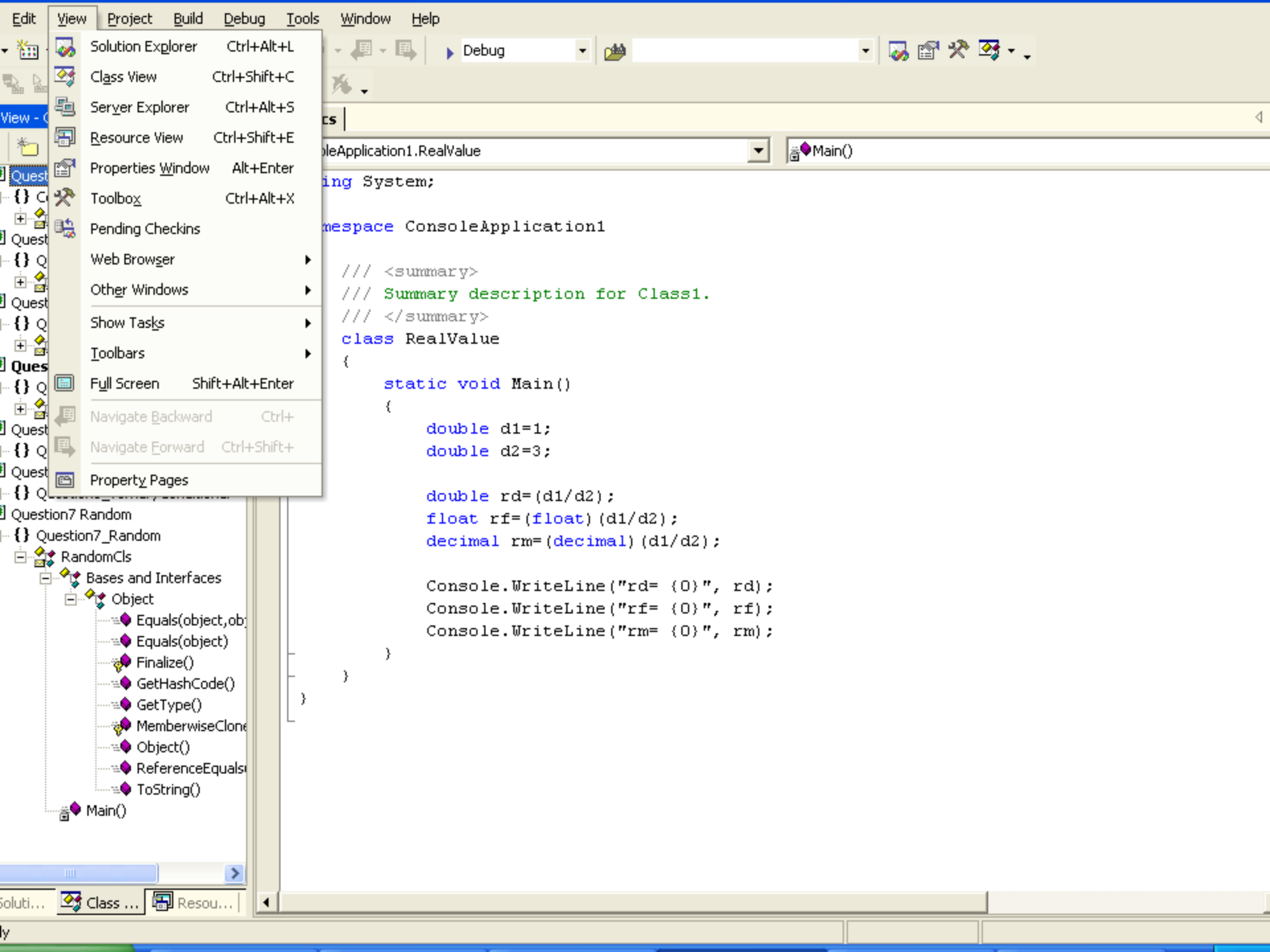
  ceux sont les fichiers les plus important

  les autres fichiers appartenants a a bin/debug et obj sont le résultat de la compilation ou du déboguage
- Pour découvrir le langage dans un 1er temps nous créerons des Console Application

L’environnement de Visual Studio

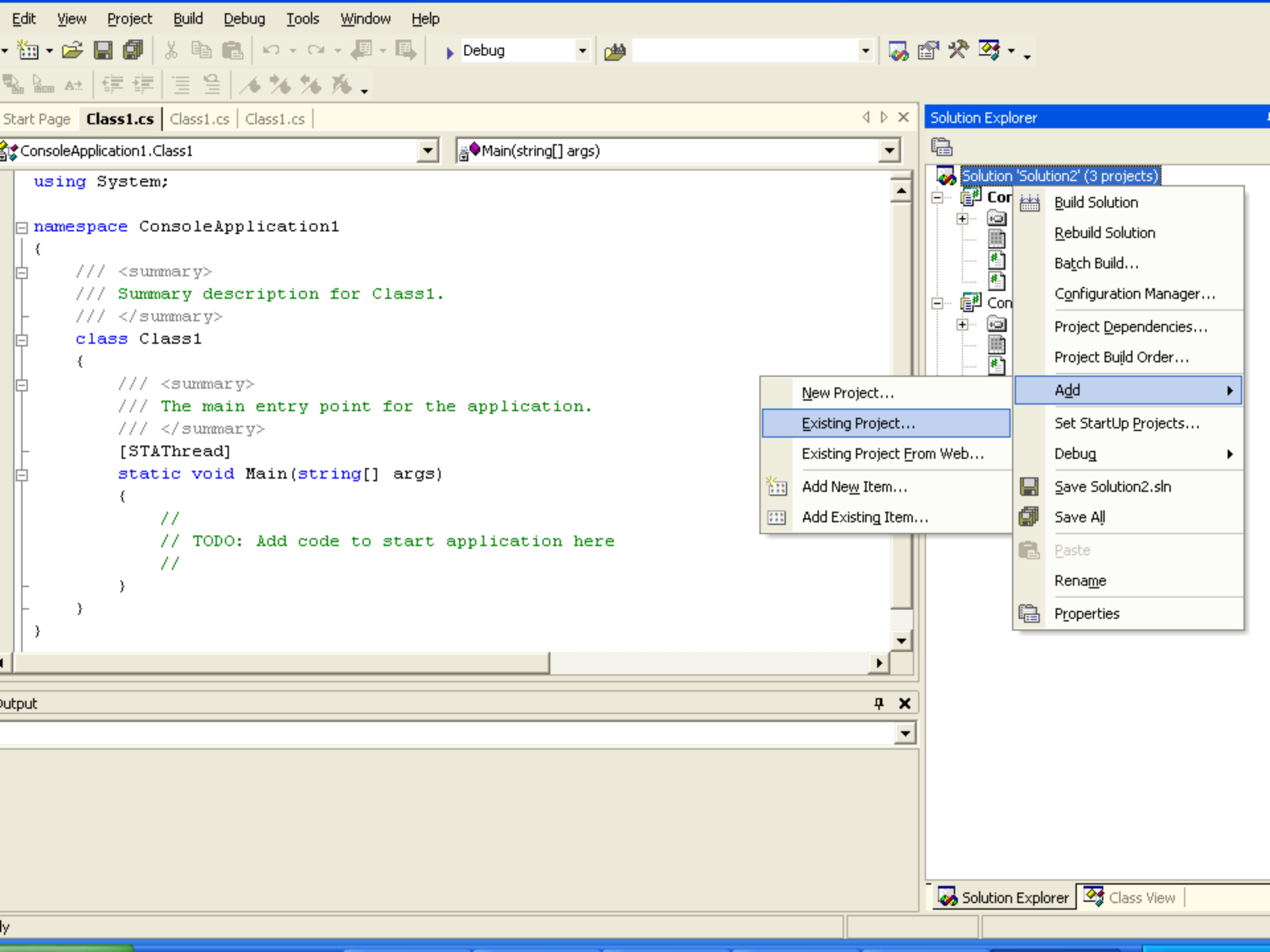
- Il existe plusieurs options dans le menu qui facilite la programmation comme différente vues des solution

- On peut parcourir les propriétés d’un projet et comprendre les options et les propriétés d’un projet


Edit
View
Project
Build
Debug
Tools
Window
Help
Solution Explorer Ctrl+Alt+L
Class View Ctrl+Shift+C
Server Explorer Ctrl+Alt+S
Resource View Ctrl+Shift+E
Properties Window Alt+Enter
Toolbox Ctrl+Alt+X
Pending Checkins
Web Browser
Other Windows
Show Tasks
Toolbars
Full Screen Shift+Alt+Enter
Navigate Backward Ctrl+
Navigate Forward Ctrl+Shift+
Property Pages
Debug
leApplication1.RealValue
Main()
ing System;
mespace ConsoleApplication1
/// <summary>
/// Summary description for Class1.
/// </summary>
class RealValue
{
static void Main()
{
double d1=1;
double d2=3;
double rd=(d1/d2);
float rf=(float)(d1/d2);
decimal rm=(decimal)(d1/d2);
Console.WriteLine("rd= {0}", rd);
Console.WriteLine("rf= {0}", rf);
Console.WriteLine("rm= {0}", rm);
}
}
}
Question7 Random
Question7_Random
RandomCls
Bases and Interfaces
Object
Equals(object,ob
Equals(object)
Finalize()
GetHashCode()
GetType()
MemberwiseClone
Object()
ReferenceEquals
ToString()
Main()
Class ...
Resou...

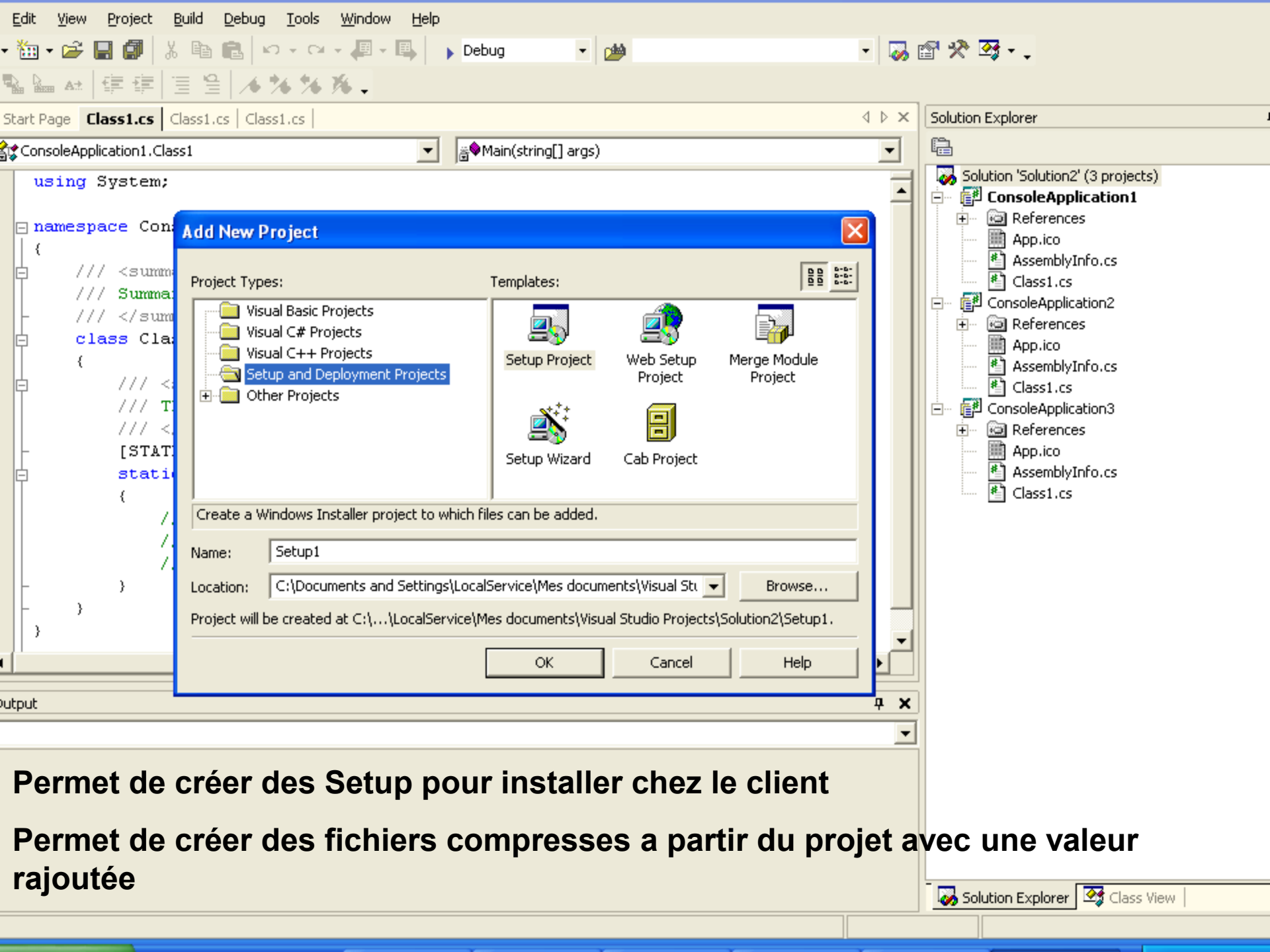
Manipuler les projets

- A travers une solution on peut:
  rajouter un nouveau projet ,
  importer un projet existant ,
  une classe existante etc…


Edit View Project Build Debug Tools Window Help
Debug
Start Page
Class1.cs
Class1.cs
Class1.cs
ConsoleApplication1.Class1
Main(string[] args)
using System;
namespace ConsoleApplication1
{
/// <summary>
/// Summary description for Class1.
/// </summary>
class Class1
{
/// <summary>
/// The main entry point for the application.
/// </summary>
[STAThread]
static void Main(string[] args)
{
//
// TODO: Add code to start application here
//
}
}
}
Solution Explorer
Solution 'Solution2' (3 projects)
Build Solution
Rebuild Solution
Batch Build...
Configuration Manager...
Project Dependencies...
Project Build Order...
Add
Set StartUp Projects...
Debug
Save Solution2.sln
Save All
Paste
Rename
Properties
New Project...
Existing Project...
Existing Project From Web...
Add New Item...
Add Existing Item...
Output
Solution Explorer
Class View

Les types d'applications

- Il existe plusieurs types d'applications ou projets:
  - console
  - windows
  - application Web
  - application de service Web
  - librairie de classe
  - projet de déploiement
  - et bien d'autre ….



Permet de créer des Setup pour installer chez le client

Permet de créer des fichiers compresses a partir du projet avec une valeur rajoutée

La page de démarrage:Start Page

- C’est une page intéressante qui nous permet de:
  être a jour avec Microsoft et les nouveautés,
  avoir des infos sur les services Web
  Une recherche en ligne