Session: 1

Building Applications Using C#

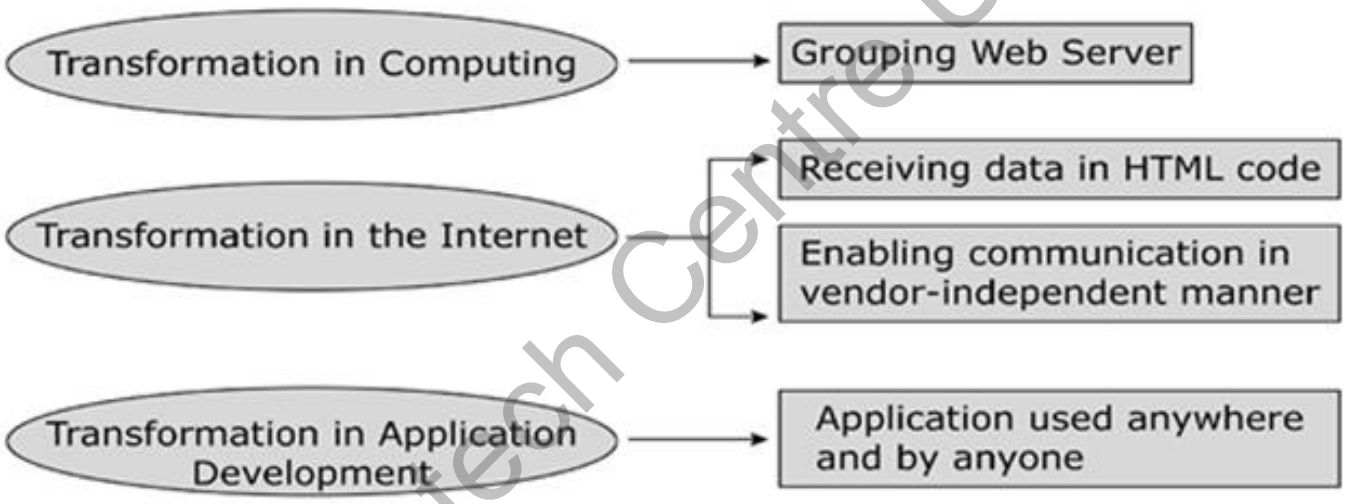
Objectives

- Define and describe the .NET Framework
- Explain the C# language features
- Define and describe the Visual Studio 2012 environment
- Explain the elements of Microsoft Visual Studio 2012 IDE

- The .NET Framework is an infrastructure that is used to:
  - Build, deploy, and run different types of applications and services using .NET technologies.
  - Minimize software development, deployment, and versioning conflicts.

- With improvements in networking technology:
  - Distributed computing has provided the most effective use of processing power of both client and server processors.
  - Applications became platform-independent with the emergence of Internet, which ensured that they could be run on PCs with different hardware and software combination.
  - Communication with each other has become possible for the clients and servers in a vendor-independent manner.

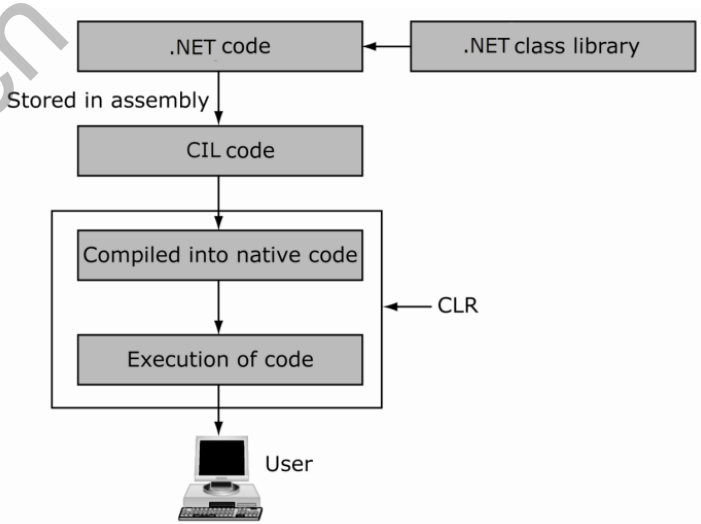
- The following figure shows the different features accompanying the transformation in computing, Internet, and application development:



- All the transformations are supported by the technology platform introduced by Microsoft called as **.NET Framework**.
- Data stored using the .NET Framework is accessible to a user from any place, at any time, through any .NET compatible device.
- The .NET Framework:
  - Is a programming platform that is used for developing Windows, Web-based, and mobile software.
  - Has a number of pre-coded solutions that manage the execution of programs written specifically for the framework.
  - Is based on two basic technologies for communication of data:
    - eXtensible Markup Language (XML)
    - The suite of Internet protocols

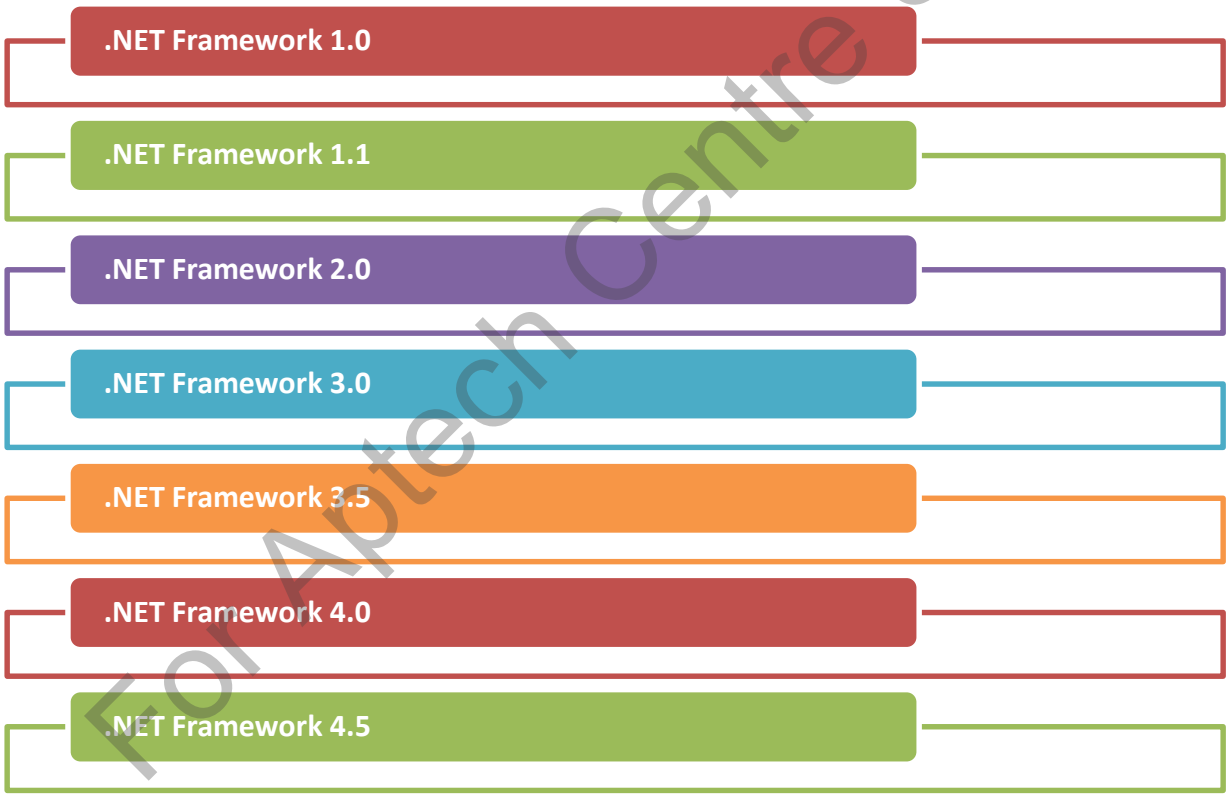
- Following are the key features of XML:
  - It separates actual data from presentation.
  - It unlocks information that can be organized, programmed, and edited.
  - It allows Web sites to collaborate and provide groups of Web services. Thus, they can interact with each other.
  - It provides a way for data to be distributed to a variety of devices.
- Apart from XML, the .NET platform is also built on Internet protocols such as:
  - Hypertext Transfer Protocol (HTTP)
  - Open Data Protocol (OData)
  - Simple Object Access Protocol (SOAP)

- In traditional Windows applications:
  - Codes were directly compiled into the executable native code of the operating system.
- Using the .NET Framework:
  - The code of a program is compiled into CIL (formerly called MSIL) and stored in a file called assembly.
  - This assembly is then compiled by the CLR to the native code at run-time.
- The following figure represents the process of conversion of CIL code to the native code:



- The CLR provides many features such as:
  - Memory management
  - Code execution
  - Error handling
  - Code safety verification
  - Garbage collection
- The applications that run under the CLR are called managed code.

- Microsoft has released different versions of the .NET Framework to include additional capabilities and functionalities with every newer version.
- Following are the versions of the .NET Framework:

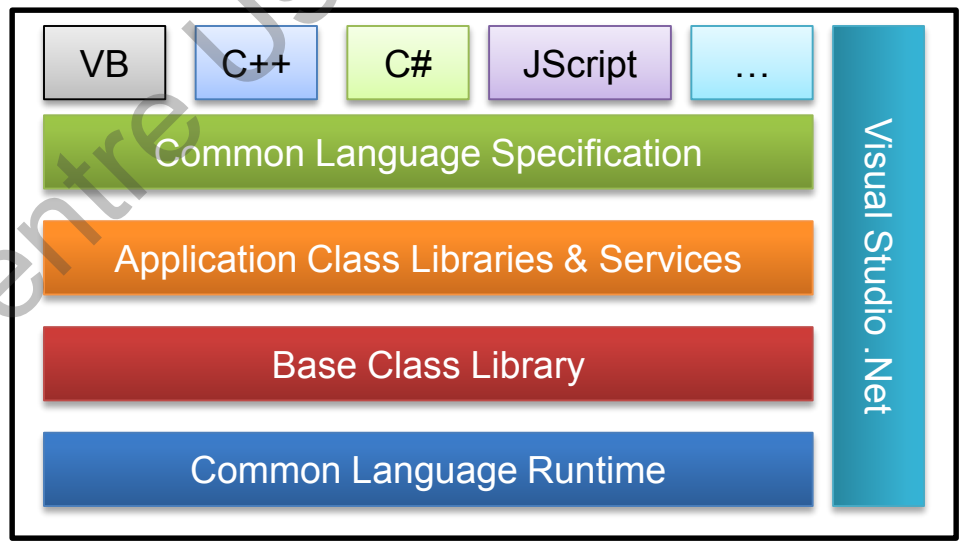


- The following table shows various versions of.NET Framework and Visual Studio:

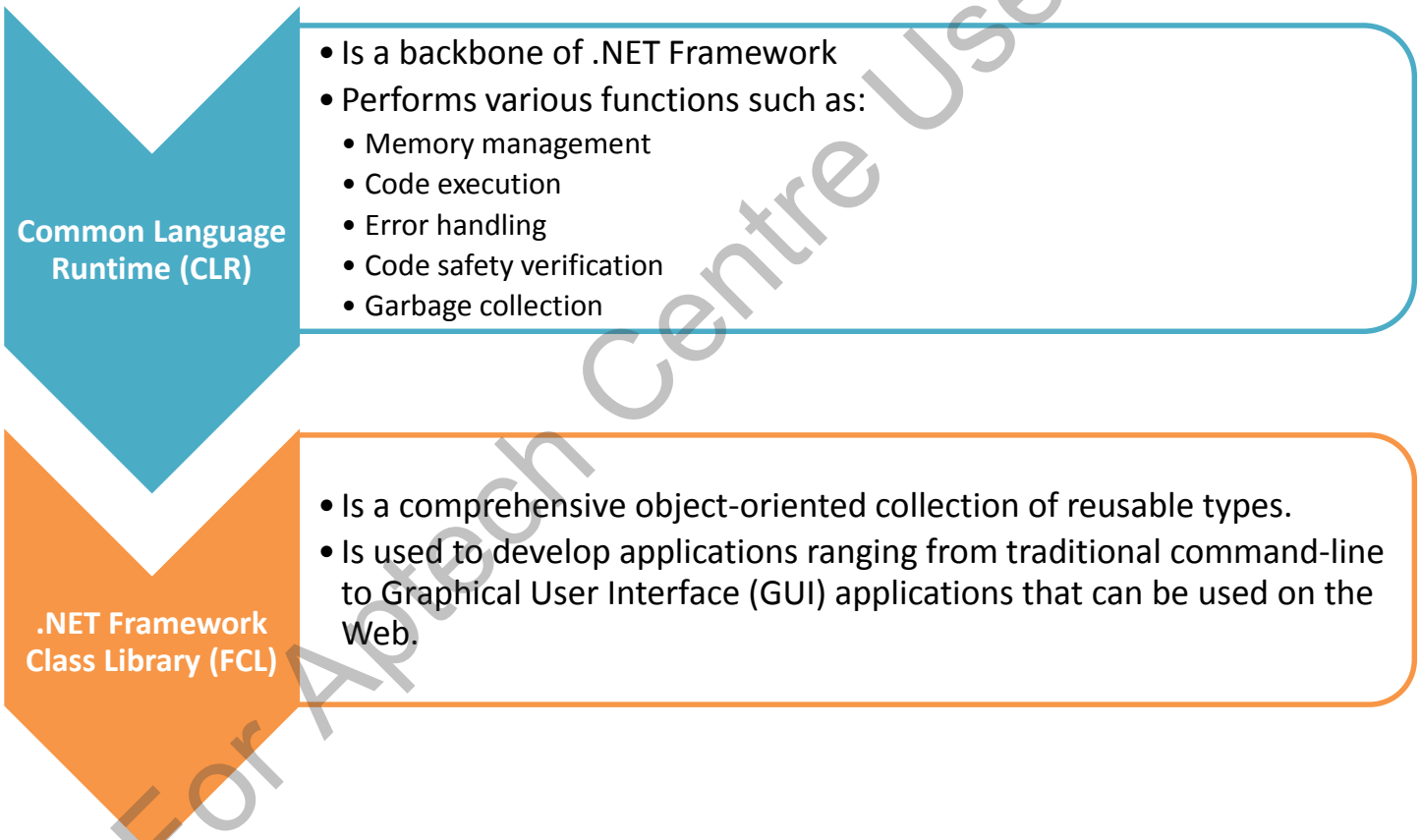
Year	.NET Framework	Distributed with OS	IDE Name
2002	1.0		Visual Studio .NET (2002)
2003	1.1	Windows Server 2003	Visual Studio .NET 2003
2005	2.0		Visual Studio 2005
2006	3.0	Windows Vista, Windows Server 2008	Visual Studio 2005 with .NET Framework 3.0 support
2007	3.5	Windows 7, Windows Server 2008 R2	Visual Studio 2008
2010	4		Visual Studio 2010
2012	4.5	Windows 8, Windows Server 2012	Visual Studio 2012

- The .NET Framework is an essential Windows component for building and running the next generation of software applications and XML Web services.

- The .NET Framework is designed to:
  - Provide consistent object-oriented programming environment.
  - Minimize software deployment and versioning conflicts by providing a code-execution environment.
  - Promote safe execution of code by providing a code-execution environment.
  - Provide a consistent developer experience across varying types of applications such as Windows-based applications and Web-based applications.



- The two core components of the .NET Framework integral to any application or service development are:



- A programmer can develop applications using one of the languages supported by .NET.
- These applications make use of the base class libraries provided by the .NET Framework.

Example

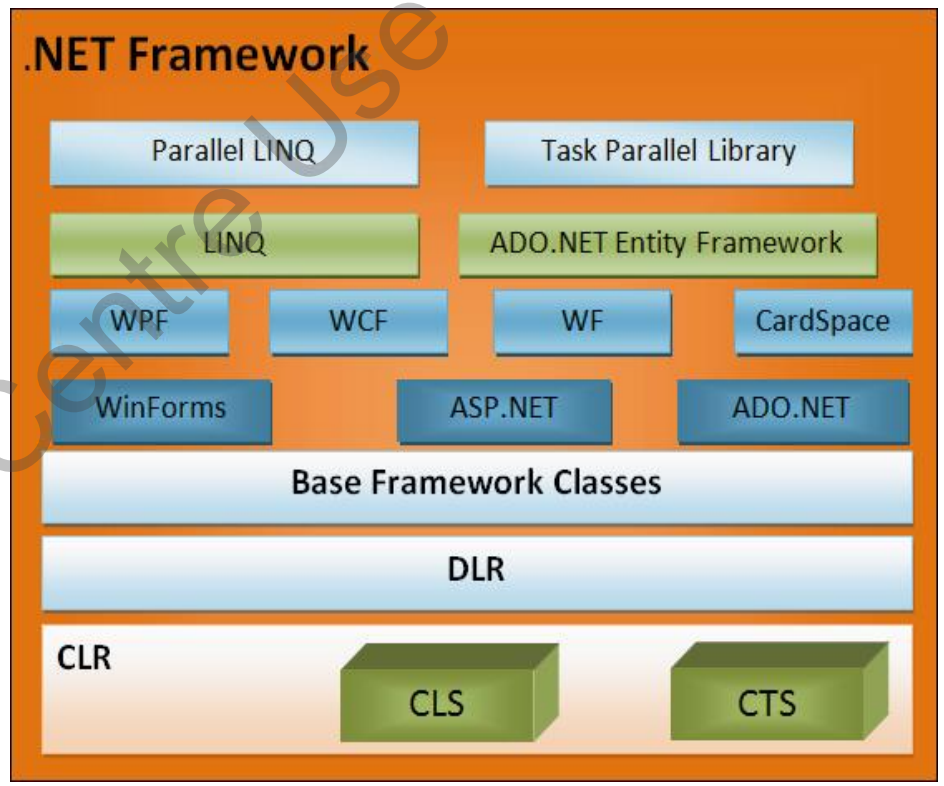
- To display a text message on the screen, the following command can be used:

```
System.Console.WriteLine(".NET Architecture");
```

- The same `WriteLine()` method will be used across all .NET languages.
- This is done by making the Framework Class Library as a common class library for all .NET languages.

Other Components of .NET Framework 1-4

- Following are some other important components:
  - Common Language Specification (CLS)
  - Common Type System (CTS)
  - Base Framework Classes
  - ASP.NET
  - ADO.NET
  - WPF
  - WCF
  - LINQ
  - ADO.NET Entity Framework
  - Parallel LINQ
  - Task Parallel Library



- **Common Language Specification (CLS):**
  - Is a set of rules that any .NET language should follow to create applications that are interoperable with other languages.
- **Common Type System (CTS):**
  - Describes how data types are declared, used, and managed in the runtime and facilitates the use of types across various languages.
- **Base Framework Classes:**
  - These classes provide basic functionality such as input/output, string manipulation, security management, network communication, and so on.
- **ADO.NET:**
  - Provides classes to interact with databases.

Other Components of .NET Framework 3-4

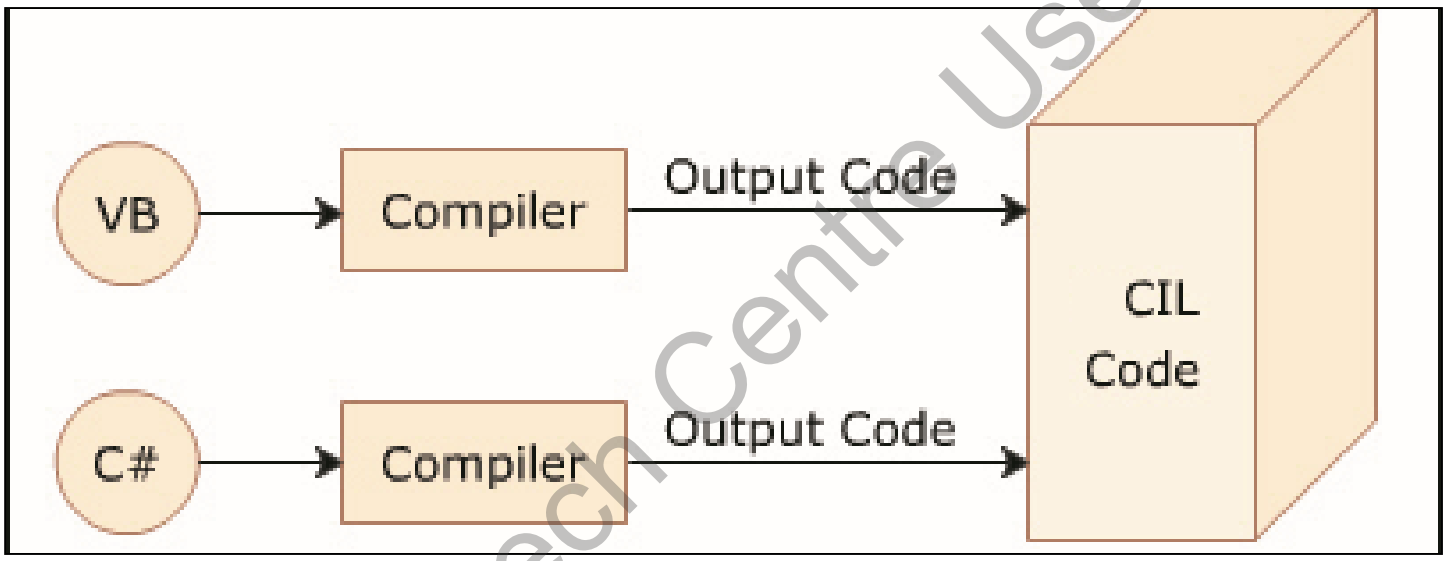
- **ASP.NET:**
  - Provides a set of classes to build Web applications. ASP.NET Web applications can be built using Web Forms, which is a set of classes to design forms for the Web pages similar to the HTML.
  - Supports Web services that can be accessed using a standard set of protocols.
- **WPF:**
  - Is a UI framework based on XML and vector graphics.
  - Uses 3D computer graphics hardware and Direct3D technologies to create desktop applications with rich UI on the Windows platform.
- **WCF:**
  - Is a service-oriented messaging framework.
  - Allows creating service endpoints and allows programs to asynchronously send and receive data from the service endpoint.

- **LINQ:**
  - Is a component that provides data querying capabilities to a .NET application.
- **ADO.NET Entity Framework:**
  - Is a set of technologies built upon ADO.NET that enables creating data-centric applications in object-oriented manner.
- **Parallel LINQ:**
  - Is a set of classes to support parallel programming using LINQ.
- **Task Parallel Library:**
  - Is a library that simplifies parallel and concurrent programming in a .NET application.

Common Intermediate Language (CIL) 1-2

- Every .NET programming language generally has a compiler and a runtime environment of its own.
- The compiler converts the source code into executable code that can be run by the users.
- One of the primary goals of .NET Framework is to combine the runtime environments so that the developers can work with a single set of runtime services.
- When the code written in a .NET compatible language such as C# or VB is compiled, the output code is in the form of MSIL code.
- MSIL is composed of a specific set of instructions that indicate how the code should be executed.
- MSIL is now called as **Common Intermediate Language (CIL)**.

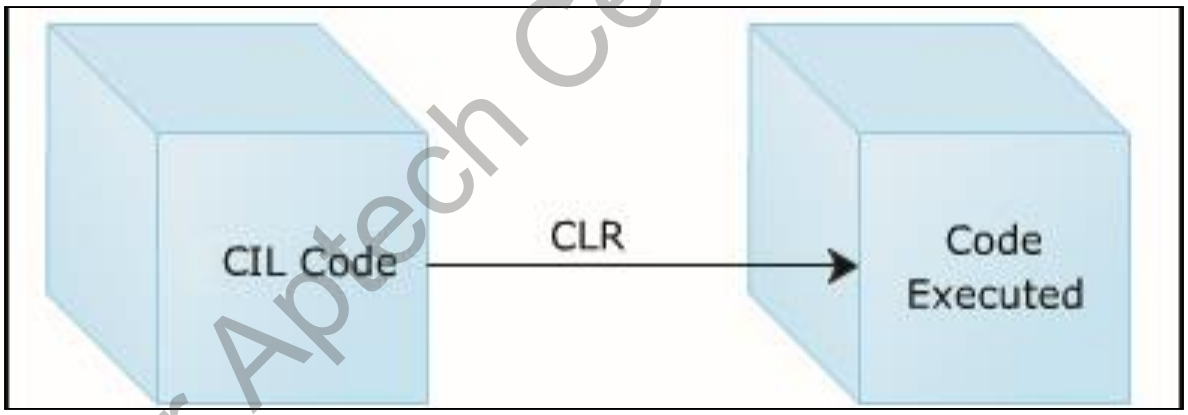
- The following figure depicts the concept of CIL:



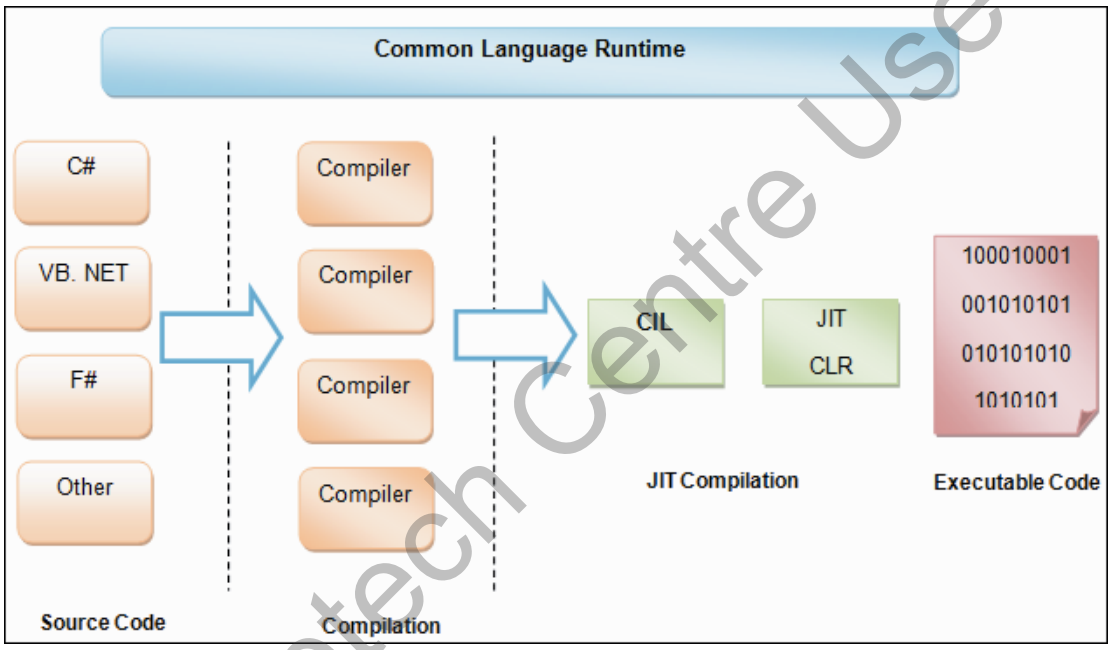
Common Language Runtime (CLR) 1-3

- The CLR:
  - Is the foundation of the .NET Framework.
  - Acts as an execution engine for the .NET Framework.
  - Manages the execution of programs and provides a suitable environment for programs to run.
  - Provides a multi-language execution environment.
- The runtime manages code at execution time and performs operations such as:
  - Memory management
  - Thread management
  - Remoting
- The .NET Framework supports a number of development tools and language compilers in its Software Development Kit (SDK).

- When a code is executed for the first time;
  - The CIL code is converted to a code native to the operating system.
  - This is done at runtime by the Just-In-Time (JIT) compiler present in the CLR.
  - The CLR converts the CIL code to the machine language code.
  - Once this is done, the code can be directly executed by the CPU.
- The following figure depicts the working of the CLR:



- The following figure shows a more detailed look at the working of the CLR:



- Dynamic Language Runtime (DLR):
  - Is a runtime environment built on top of the CLR to enable interoperability of dynamic languages such as Ruby and Python with the .NET Framework.
  - Allows creating and porting dynamic languages to the .NET Framework.
  - Provides dynamic features to the existing statically typed languages. For example, C# relies on the DLR to perform dynamic binding.

- The .NET Framework languages, such as C#, VB, and J# are statically typed languages.
- In dynamic languages, programmers are not required to specify object types in the development phase.

Need for a New Language

- Microsoft introduced C# as a new programming language to address the problems posed by traditional languages.
- C# was developed to:
  - Create a very simple and yet powerful tool for building interoperable, scalable, and robust applications.
  - Create a complete object-oriented architecture.
  - Support powerful component-oriented development.
  - Allow access to many features previously available only in C++ while retaining the ease-of-use of a rapid application development tool such as Visual Basic.
  - Provide familiarity to programmers coming from C or C++ background.
  - Allow to write applications that target both desktop and mobile devices.

Purpose of C# Language

- Microsoft .NET was formerly known as Next Generation Windows Services (NGWS).
- It is a completely new platform for developing the next generation of Windows/Web applications.
- These applications transcend device boundaries and fully harness the power of the Internet.
- However, building the new platform required a language that could take full advantage.
- This is one of the factors that led to the development of C#.
- C# is an object-oriented language derived from C and C++.
- **The goal of C# is to provide a simple, efficient, productive, and object-oriented language that is familiar and yet at the same time revolutionary.**

Language Features

- C# has features common to most object-oriented languages.
- It has language-specific features, such as:
  - Type safety checking
  - Generics
  - Indexers
- These features make the C# as a preferred language to create a wide variety of applications.

Basic Features of C# 1-3

- C# is a programming language designed for building a wide range of applications that run on the .NET Framework.
- Following are some basic key features of C#:
  - Object-oriented Programming
  - Type-safety Checking
  - Garbage Collection
  - Standardization by European Computer Manufacturers Association (ECMA)
  - Generic Types and Methods
  - Iterators
  - Static Classes
  - Partial Classes
  - Anonymous Methods
  - Methods with named Arguments
  - Methods with optional Arguments
  - Nullable Types
  - Accessor Accessibility
  - Auto-implemented Properties
  - Parallel Computing

- **Object-oriented Programming:**
  - Focuses on objects so that code written once can be reused. This helps reduce time and effort on the part of developers.
- **Type-safety Checking:**
  - Checked the overflow of types because uninitialized variables cannot be used in C# as C# is a case-sensitive language.
- **Garbage Collection:**
  - Performs automatic memory management from time to time and spares the programmer the task.
- **Standardization by European Computer Manufacturers Association (ECMA):**
  - Specifies the syntax and constraints used to create standard C# programs.
- **Generic Types and Methods:**
  - Are a type of data structure that contains code that remains the same throughout but the data type of the parameters can change with each use.
- **Iterators**: Enable looping (or iterations) on user-defined data types with the for each loop.
- **Static Classes:** Contain only static members and do not require instantiation.

- **Partial Classes:** Allow the user to split a single class into multiple source code (.cs) files.
- **Anonymous Methods:** Enable the user to specify a small block of code within the delegate declaration.
- **Methods with named Arguments:** Enable the user to associate a method argument with a name rather than its position in the argument list.
- **Methods with optional Arguments:** Allow the user to define a method with an optional argument with a default value.
- **Nullable Types:** Allow a variable to contain a value that is undefined.
- **Accessor Accessibility:** Allows the user to specify the accessibility levels of the `get` and `set` accessors.
- **Auto-implemented Properties:** Allow the user to create a property without explicitly providing the methods to get and set the value of the property.
- **Parallel Computing:** Support for parallel programming using which develop efficient, fine-grained, and scalable parallel code without working directly with threads or the thread pool.

Applications of C#

- C# is an object-oriented language that can be used in a number of applications.
- Following are some of the applications:
  - Web applications
  - Web services
  - Gaming applications
  - Large-scale enterprise applications
  - Mobile applications for pocket PCs, PDAs, and cell phones
  - Simple standalone desktop applications such as Library Management Systems, Student Mark Sheet generation, and so on
  - Complex distributed applications that can spread over a number of cities or countries
  - Cloud applications

Advantages of C#

- C# has become a preferred programming language over C++ because of its simplicity and user-friendliness.
- The advantages of C# are as follows:

Cross Language Support:

- The code written in any other .NET language can be easily used and integrated with C# applications.

Common Internet Protocols:

- .NET offers extensive support for XML, which is the preferred choice for formatting information over the Internet.
- Additionally, support for transfer via SOAP is also integrated.

Simple Deployment:

- Deployment of C# applications is made simple by the concept of assemblies.
- An assembly is a self-describing collection of code and resources.
- It specifies exactly the location and version of any other code it needs.

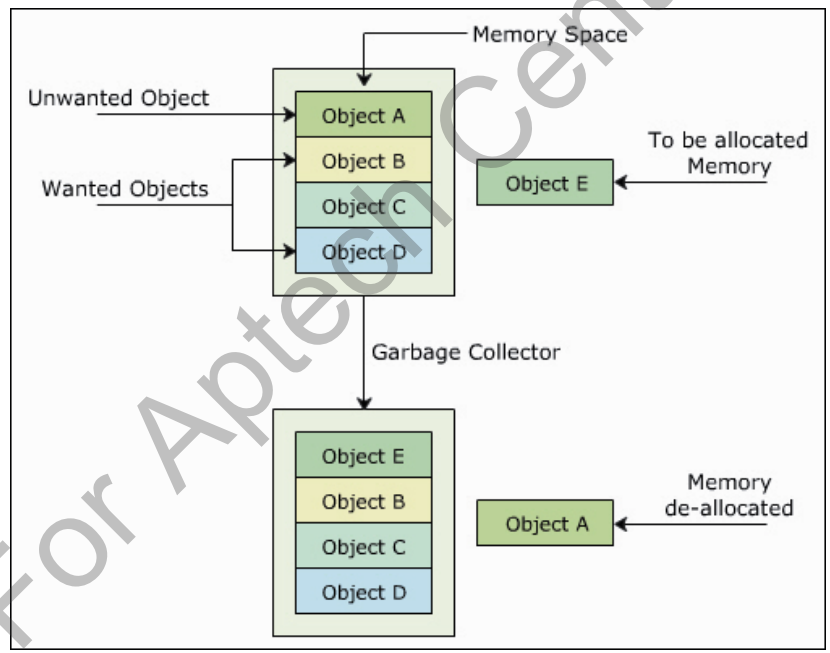
XML Documentation:

- Comments can be placed in XML format and can then be used as needed to document your code.
- It can include example code, parameters, and references to other topics.
- It makes sense for a developer to document his or her code because those comments can actually become documentation independent of the source code.

Memory Management

- In programming languages like C and C++, the allocation and de-allocation of memory is done manually.
- Performing these tasks manually is both, time-consuming and difficult.
- The C# language provides the feature of allocating and releasing memory using automatic memory management.
- This means that there is no need to write code to allocate memory when objects are created or to release memory when objects are not required in the application.
- Automatic memory management increases the code quality and enhances the performance and the productivity.

- Garbage collection:
  - Is the process of automatic reclaiming of memory from objects that are no longer in scope.
  - Helps the process of allocating and de-allocating memory using automatic memory management.
- The following figure illustrates concept of garbage collection:



- Visual Studio 2012 Environment:
  - Provides the environment to create, deploy, and run applications developed using the .NET framework.
  - Comprises the Visual Studio Integrated Development Environment (IDE), which is a comprehensive set of tools, templates, and libraries required to create .NET framework applications.
  - Is a complete set of development tools for building high performance desktop applications, XML Web Services, mobile applications, and ASP Web applications.
  - Is also used to simplify team-based design, development, and deployment of enterprise solutions.
  - Is an IDE used to ease the development process of .NET applications such as Visual C# 2012 and Visual Basic 2012.
  - Uses the same IDE, debugger, Solution Explorer, Properties tab, Toolbox, standard menus, and toolbars for all the .NET compatible languages.

- The following features of IDE make it useful for an easier development process:
  - A single environment is provided for developing the .NET applications.
  - Several programming languages are available to choose from for developing applications.
  - A built-in browser is available in the IDE that is used for browsing the Internet without launching another application.
  - A program can be executed with or without a debugger.
  - The application can be published over the Internet or onto a disk.
  - The application provides Dynamic Help on a number of topics using the MSDN library.
  - The syntax of the code is checked as the user is typing it and the appropriate error notification is provided in case an error is encountered.

- The IDE:
  - Can be customized, based on the user's preferences.
  - Provides a standard code editor to write the .NET applications.
  - Has an integrated compiler to compile and execute the application. The user can either compile a single source file or the complete project.
  - Has a set of visual designers that simplifies application developments.
- Some commonly used visual designers are as follows:
  - **Windows Form Designer:** Allows programmers to design the layout of Windows forms and drag and drop controls to it.
  - **Web Designer:** Allows programmers to design the layout and visual elements of ASP.NET Web pages.
  - **WPF Designer:** Allows programmers to create user interfaces targeting WPF.
  - **Class Designer:** Allows programmers to use UML modeling to design classes, its members, and relationships between classes.
  - **Data Designer:** Allows programmer to edit database schemas graphically.

- Visual Studio 2012 provides multiple advantages in the development process.
- The primary advantages are as follows:
  - Improved developer productivity
  - Development of applications for Microsoft .NET Framework 4.5
  - Development of plug-ins to extend the IDE's capabilities

Visual Studio 2012 Editions

- The IDE of Microsoft Visual Studio is a result of extensive research by the Microsoft team.
- The different editions of Visual Studio 2012 are:

Visual Studio Professional 2012	• This is the entry-level edition that provides support for developing and debugging applications, such as Web, desktop, cloud-based, and mobile applications.
Visual Studio Professional 2012 with MSDN	• This edition provides all the features of the Visual Studio Professional 2012 edition along with an MSDN subscription. • In addition, this edition includes Team Foundation Server and provides access to cloud, Windows Store, and Windows Phone Marketplace.
Visual Studio Test Professional 2012 with MSDN	• This edition targets testers and Quality Assurance (QA) professionals by providing project management tools, testing tools, and virtual environment to perform application testing.
Visual Studio Premium 2012 with MSDN	• This edition provides all the features of the combined Visual Studio Professional 2012 and Visual Studio Test Professional 2012 with MSDN editions. • In addition, this edition supports peer code review, User Interface (UI) validation through automated tests, and code coverage analysis to determine the amount of code being tested.
Visual Studio Ultimate 2012 with MSDN	• This edition has all the features of the other editions. In addition, this edition supports designing architectural layer diagrams, performing Web performance and load testing, and analyzing diagnostic data collected from runtime systems.

Languages in Visual Studio 2012

- Visual Studio 2012 supports multiple programming languages such as:
  - Visual Basic .NET
  - Visual C++
  - Visual C#
  - Visual J#
- The **classes** and **libraries** used in the Visual Studio 2012 IDE are common for all the languages in Visual Studio 2012.
- It makes Visual Studio 2012 more flexible.

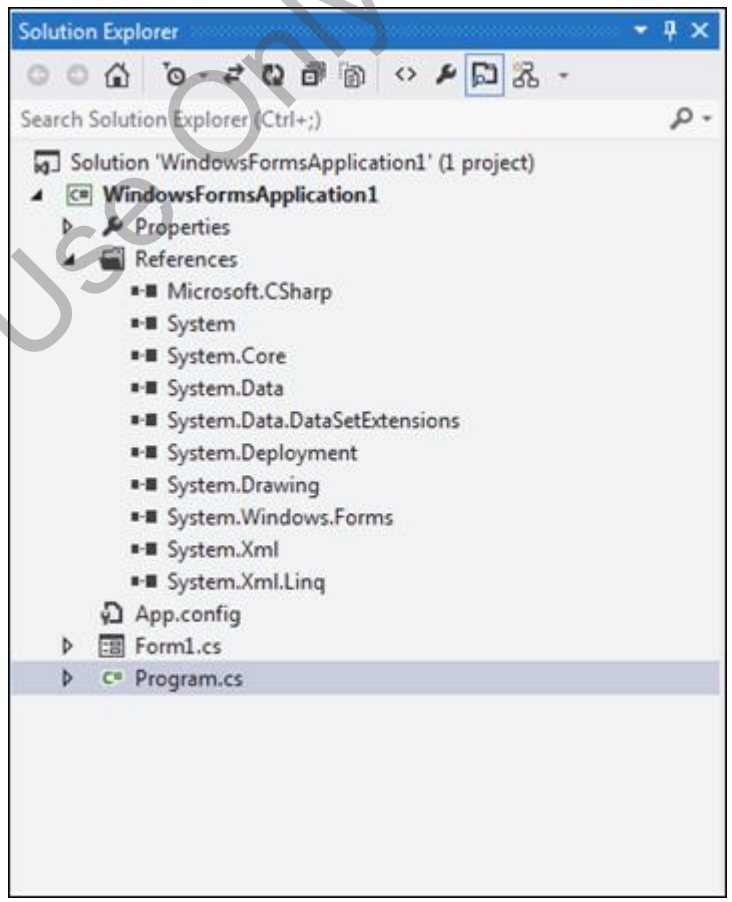
- The features of Visual Studio 2012 are:
  - **Comprehensive Tools Platform:**
    - In Visual Studio 2012, developers of all knowledge levels can make use of developer tools, which offer a development experience tailored for their unique needs.
  - **Reduced Development Complexity:**
    - Visual Studio 2012 enables customers to deliver more easily a broad range of .NET Framework-based solutions including Windows, Office, Web, and mobile applications.
  - **Edit Marks:**
    - Visual Studio 2012 provides a visual indication of the changes that are made and not saved and changes that are made during the current session that have been saved to the disk.
  - **Code Snippets:**
    - Code Snippets are small units of C# source code that the developer can use quickly with the help of certain keystrokes.

- **AutoRecover:**
  - Visual Studio 2012 automatically saves the work on a regular basis and thus, minimizes loss of information due to unexpected closing of unsaved files.
  - In case of an IDE crash, Visual Studio 2012 will also prompt you to recover your work after you restart.
- **IntelliSense:**
  - Visual Studio 2012 has the IntelliSense feature in which syntax tips, lists of methods, variables and classes pop up continually when entering code in the Code Editor, making the process of entering code more efficient.
- **Refactoring:**
  - Refactoring enables developers to automate common tasks when restructuring code.
  - It changes the internal structure of the code, specifically the design of its objects, to make it more comprehensible, maintainable, and efficient without changing its behavior.

Elements of Microsoft Visual Studio 2012 IDE

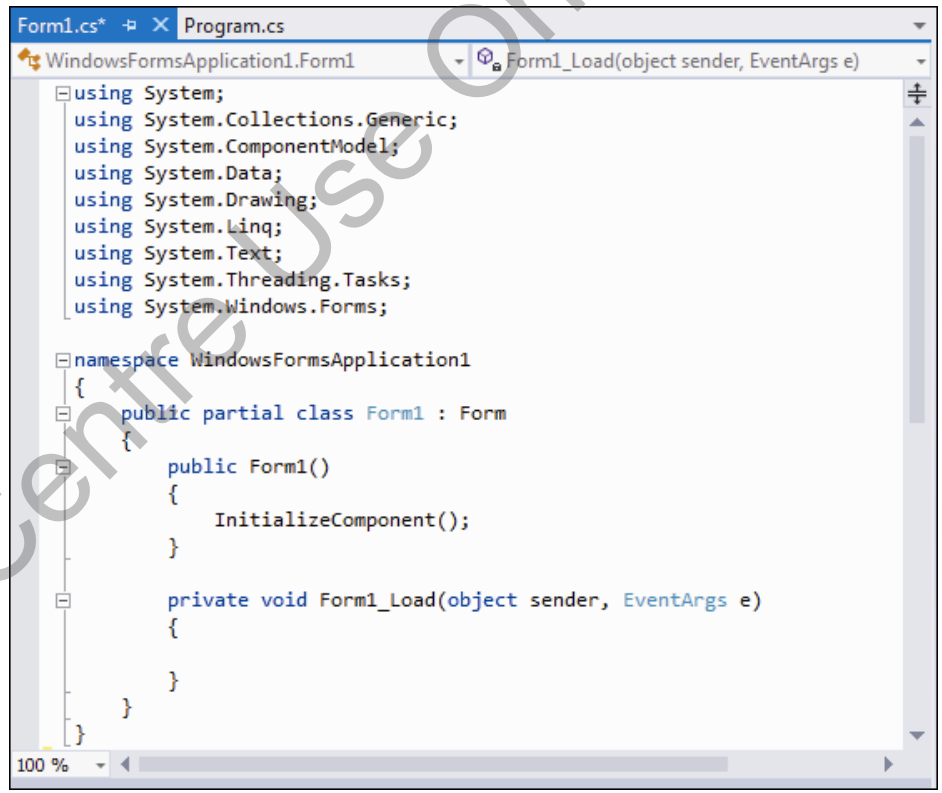
- Visual Studio 2012 contains an extensive set of elements, comprising of editors, toolbox, and different windows to assist developers in creating .NET applications.
- The key elements in Visual Studio 2012 IDE are:
  - **Solution Explorer**
  - **Code Editor**
  - **Properties Window**
  - **Toolbox**
  - **Server Explorer**
  - **Output Window**
  - **Error List**
  - **Dynamic Help**

- **Solution Explorer:**
  - Provides you with an organized view of your projects and their files.
  - Gives you ready access to the commands that pertain to these projects.
  - Helps you to use the Visual Studio editors to work on files outside the context of a solution or project.

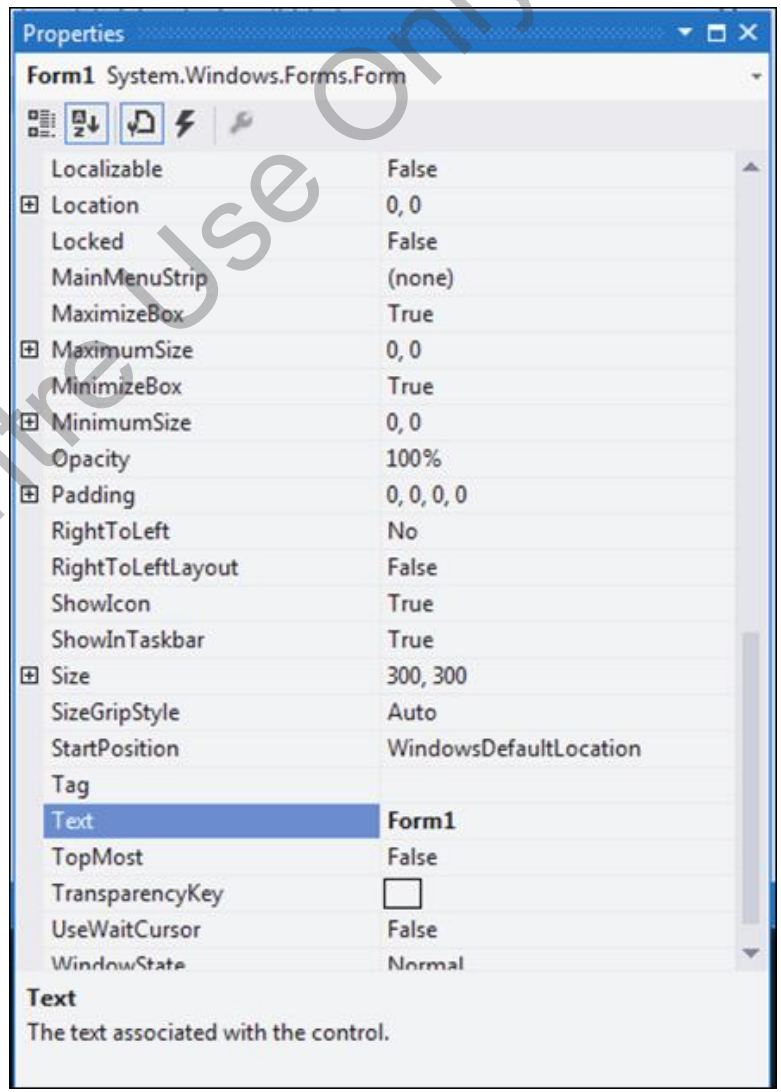


- The reference node consists of the assemblies referenced in the current project. Form1.cs is the name of the source file.
- When a code is written and saved, the .NET solution is saved in a .sln file, the C# project is saved in a .csproj file and the source code is saved in a .cs file.

- **Code Editor:**
  - Used to write, display and edit form, event and method code.
  - Helps you to open as many code windows as you want and easily copy and paste codes from one window to another.

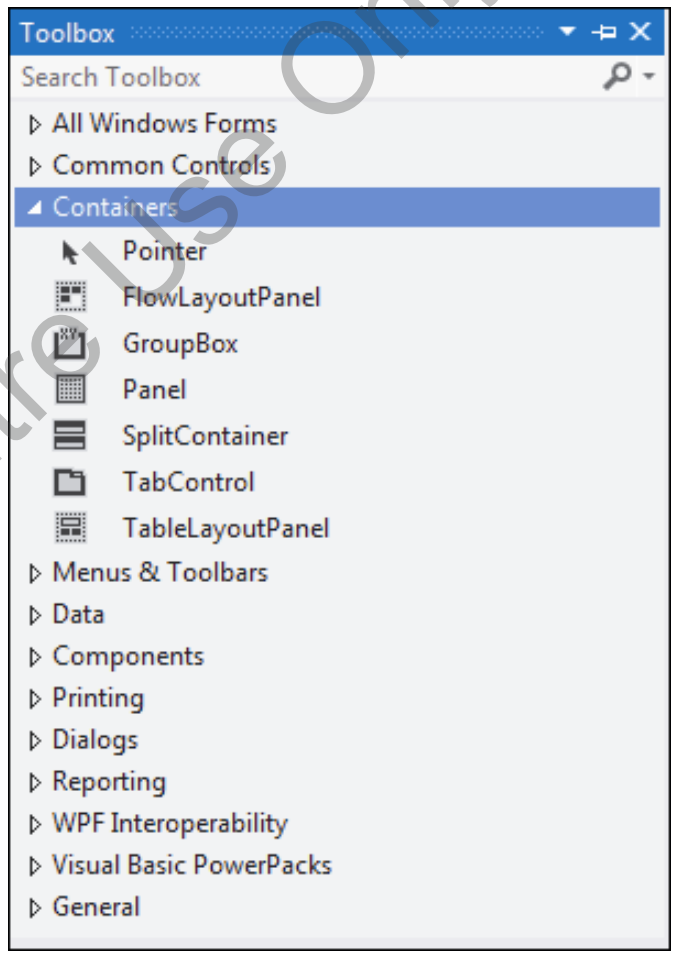


- **Properties Window:**
  - Used to view and change the design-time properties and events of selected objects that are located in editors and designers.
  - Displays different types of editing fields depending on the needs of a particular property.
  - Shows included fields such as edit boxes, drop-down lists, and links to custom editor dialog boxes.



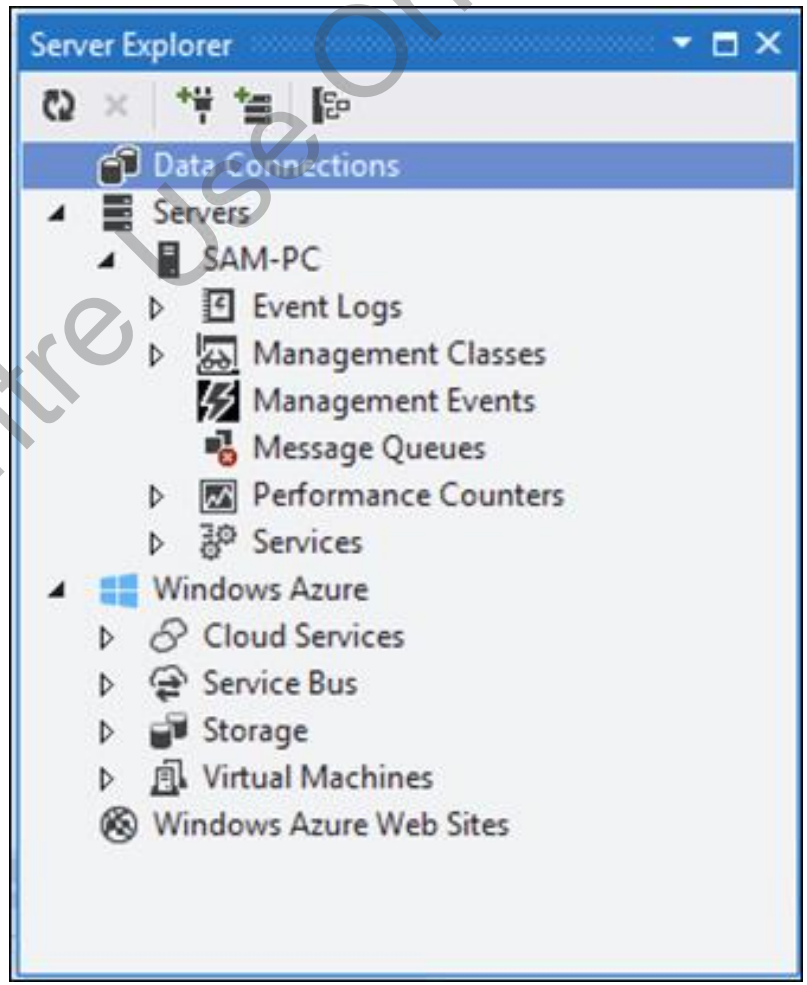
- **Toolbox:**
  - Displays the controls and components that can be added to the Design mode of the form.
  - Displays the contents of the Toolbox window and change according to the type of form the user is creating or editing.

    For example, if the user is adding tools onto a Web form, the Toolbox displays the server controls, HTML controls, data controls, and other components that the Web form may require.

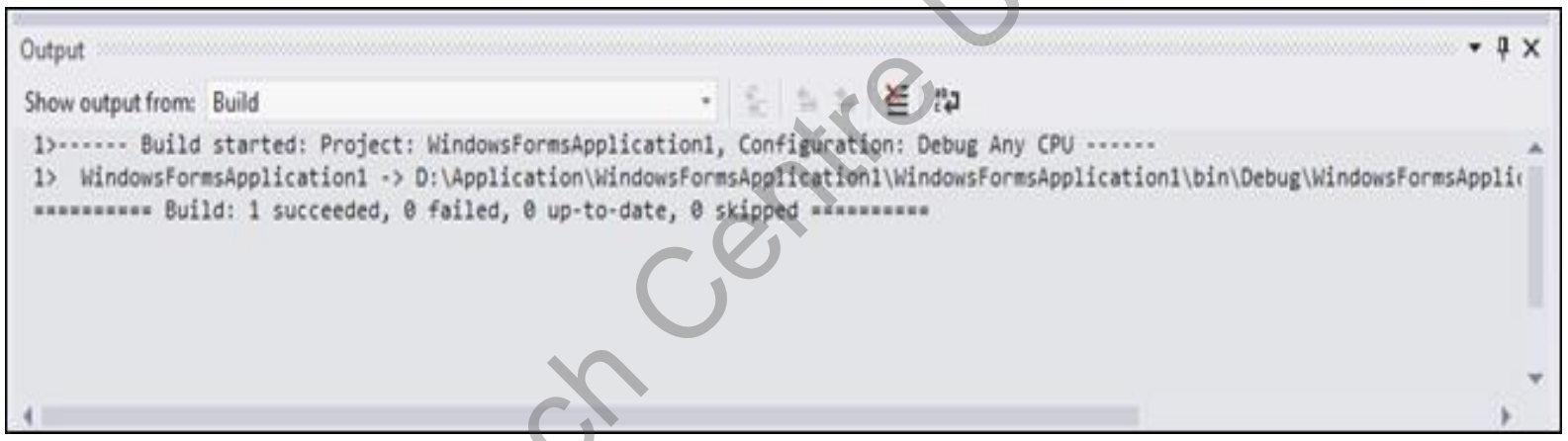


- To use the controls or components from the Toolbox:
  - The user can drag and drop the required control or component onto a form.
  - If the user is creating or editing codes in the code editor, the Toolbox contains only a Clipboard Ring.
  - This Clipboard Ring contains the last 20 items that have been cut or copied so that they can be pasted into the document, if necessary.
  - To paste the text from the Clipboard Ring, click the text and drag it to the place where it is to be inserted.

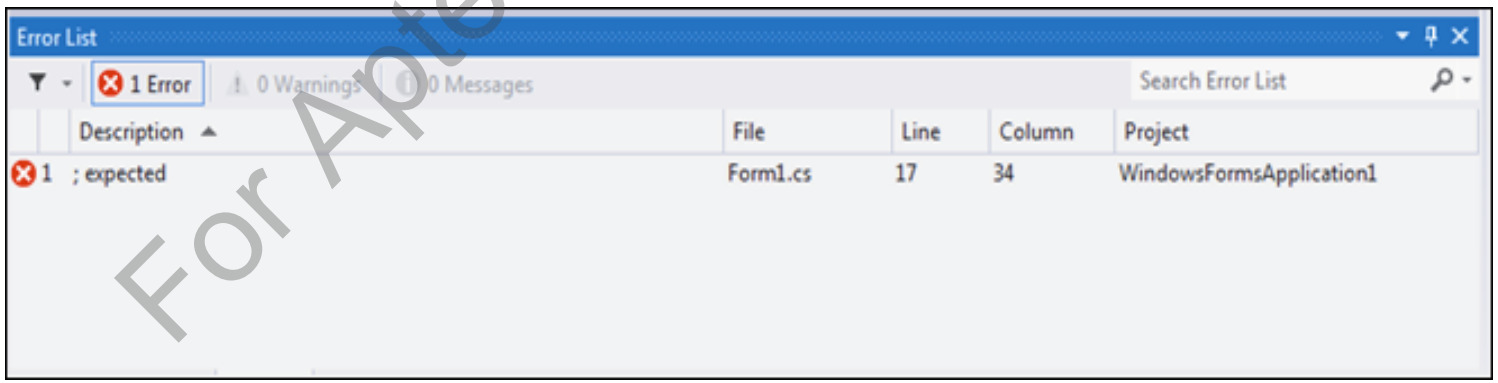
- **Server Explorer:**
  - Is the server management console used for opening data connections, logging on to servers, exploring databases, and system services.
  - Is used to perform the following activities:
    - View and retrieve information from all of the databases that the user is connected to
    - List database tables, views, stored procedures, and functions
    - Expand individual tables to list their columns and triggers
    - Open data connections to SQL and other databases
    - Log on to servers and display their databases and system services such as event logs, message queries, and so on
    - View information about available Web Services



- **Output Window:**
  - The Output window displays the status messages for the various functionalities in the IDE as follows:



- **Error List:**
  - With the help of the Error List, the speed of application development increases. The Error List window does the following:
  - Displays the errors, warnings, and messages produced when the code is edited and compiled.
  - Finds the syntactical errors noted by IntelliSense.
  - Finds the deployment errors, certain static analysis errors, and errors detected while applying Enterprise Template policies.
  - Filters which entries are displayed and which columns of information appear for each entry.
  - When the error occurs, the user can find out the location of the error by double-clicking the error message in the Error List window.



- **Dynamic Help:**
  - Provides a list of topics specific to the area of the IDE you are working in or the task you are working on.

csc Command 1-4

- Console applications that are created in C# run in a console window. This window provides simple text-based output.
- The `csc` (C Sharp Compiler) command can be used to compile a C# program.
- Following are the steps to compile and execute a program:
  1. **Create a New Project**.
  2. **Compile a C# Program**.
  3. **Execute the Program**.

- **Create a New Project:**
  - Following are the steps to create a new project:
    - Start **Visual Studio 2012**.
    - Select **New → Project** from the **File** menu.
    - Expand the **Templates → Visual C#** nodes in the left pane and select Console Application in the right pane of the New Project dialog box.
    - Specify the name and location for the project and click **OK**. Visual Studio 2012 opens the Code Editor with the skeleton code of a class, as shown in the following code:

Snippet

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace SampleProgram
{
  class Program
  {
      static void Main(string[] args)
        {
        }
    }
}
```

    - Add the following code after the opening curly brace of the `Main(string[] args)` method definition:

```
Console.WriteLine("This is a sample C# program");
```

- **Compile a C# Program:**
  - A C# program can be compiled using the following syntax:

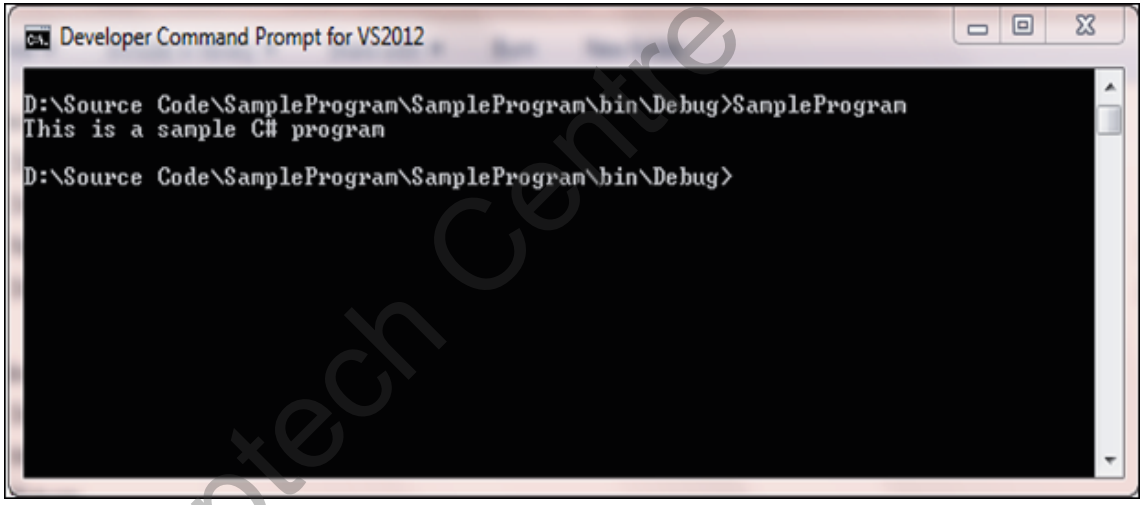
```
csc <file.cs>
```

Example

```
csc SampleProgram.cs
```

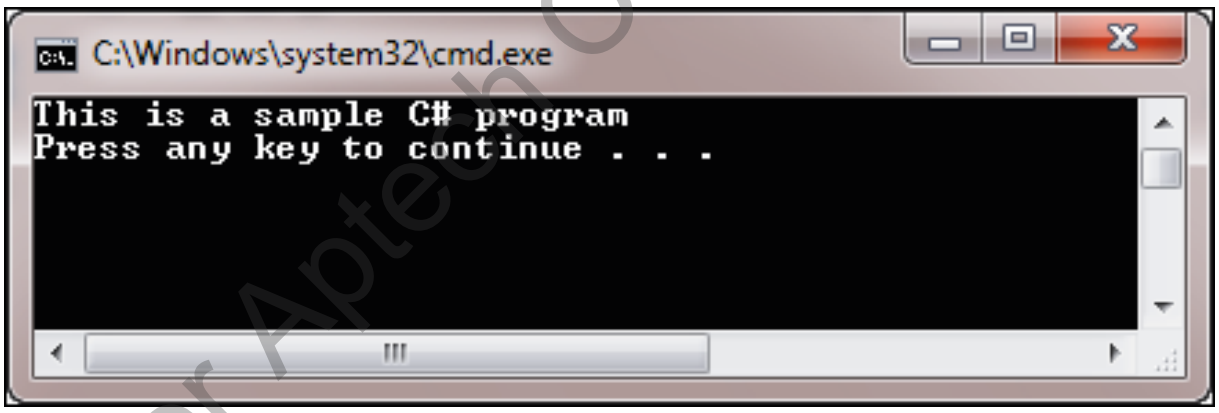
  - In the example:
    - `SampleProgram`: Specifies the name of the program to be compiled.
    - This command generates an executable file `SampleProgram.exe`.

- **Execute the Program:**
  - Open the Developer Command Prompt for VS2012, and browse to the directory that contains the `.exe` file.
  - Type the file name at the command prompt.
  - The following figure shows the developer command prompt for VS2012 window:



  - The .exe file is known as portable EXE as it contains machine-independent instructions.
  - The portable EXE works on any operating system that supports the .NET platform.

- The IDE also provides the necessary support to compile and execute C# programs.
- Following are the steps to compile and execute C# programs:
  - **Compiling the C# Program**
    - Select **Build <application name>** from the **Build** menu. This action will create an executable file `(.exe)`.
  - **Executing the Program:**
    - Select **Start Without Debugging** from the **Debug** menu.



Summary

- The .NET Framework is an infrastructure that enables building, deploying, and running different types of applications and services using .NET technologies.
- The two core components of the .NET Framework which are integral to any application or service development are the CLR and the .NET Framework class library.
- The CLR is a virtual machine component of .NET that is used to convert the CIL code to the machine language code.
- C# is an object-oriented language derived from C and C++.
- The C# language provides the feature of allocating and releasing memory using automatic memory management.
- Visual Studio 2012 provides the environment to create, deploy, and run applications developed using the .NET framework.
- Some of the languages supported by Visual Studio 2012 include Visual Basic .NET, Visual C++, Visual C#, Visual J#, and Visual F#.