msdn training

# Module 1: Overview of the Microsoft .NET Platform

**Contents**



*This course is based on the prerelease Beta 1 version of Microsoft® Visual Studio .NET. Content in the final release of the course may be different from the content included in this prerelease version. All labs in the course are to be completed with the Beta 1 version of Visual Studio .NET.*

Microsoft

# Overview

- Introduction to the .NET Platform
- Overview of the .NET Framework
- Benefits of the .NET Framework
- The .NET Framework Components
- Languages in the .NET Framework

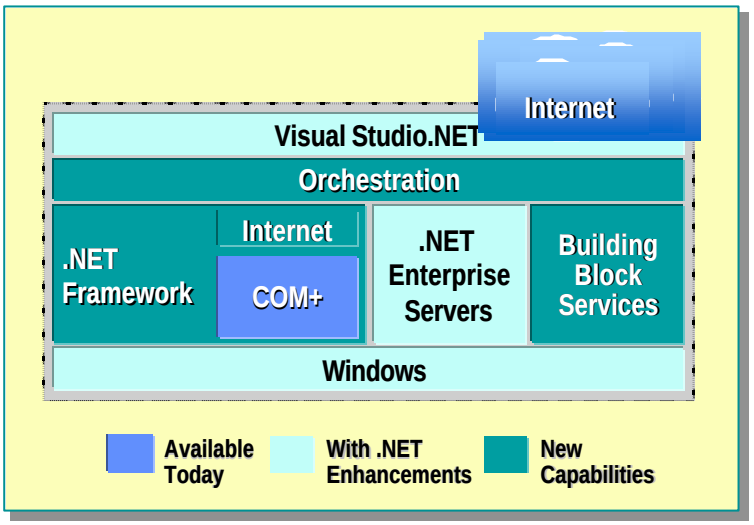
The Microsoft® .NET platform provides all of the tools and technologies that you need to build distributed Web applications. It exposes a language-independent, consistent programming model across all tiers of an application while providing seamless interoperability with, and easy migration from, existing technologies. The .NET platform fully supports the Internet's platform-neutral, standards-based technologies, including HTTP, Extensible Markup Language (XML), and Simple Object Access Protocol (SOAP).

C# is a new language specifically designed for building applications in the .NET environment. As a developer, you will find it useful to understand the rationale and features that provide the foundation for the .NET platform before you start writing C# code.

After completing this module, you will be able to:

- Describe the .NET platform.
- List the main elements of the .NET platform.
- Explain the language support in the .NET Framework.
- Describe the .NET Framework and its components.

# Introduction to the .NET Platform




The .NET platform is made up of several core technologies as shown on the slide. These technologies are described in the following topics.

## The .NET Framework

The .NET Framework is based on a new Common Language Runtime. The Common Language Runtime provides a common set of services for projects built in Microsoft Visual Studio.NET, regardless of the language. These services provide key building blocks for applications of any type, across all application tiers.

Microsoft Visual Basic®, Microsoft Visual C++®, and other Microsoft programming languages have been enhanced to take advantage of these services. Third-party languages that are written for the .NET platform also have access to the same services. The .NET Framework is explained in greater detail later in this module.

## The .NET Building Block Services

The .NET building block services are distributed programmable services that are available both online and offline. A service can be invoked on a stand-alone computer not connected to the Internet, provided by a local server running inside a company, or accessed by means of the Internet. Microsoft .NET building block services can be used from any platform that supports SOAP. Microsoft Windows-based clients are optimized to distribute Web Services to every kind of device. Services include identity, notification and messaging, personalization, schematized storage, calendar, directory, search, and software delivery.

## The .NET Enterprise Servers

The .NET Enterprise Servers provide scalability, reliability, management, integration within and across organizations, and many other features, as described in the following table.

| Server | Description |
|---|---|
| Microsoft SQL Server™ 2000 | Includes rich XML functionality, support for Worldwide Web Consortium (W3C) standards, the ability to manipulate XML data by using Transact SQL (T-SQL), flexible and powerful Web-based analysis, and secure access to your data over the Web by using HTTP. |
| Microsoft BizTalk™ Server 2000 | Provides enterprise application integration (EAI), business-to-business integration, and the advanced BizTalk Orchestration technology to build dynamic business processes that span applications, platforms, and organizations over the Internet. |
| Microsoft Host Integration Server 2000 | Provides the best way to embrace Internet, intranet, and client/server technologies while preserving investments in existing earlier systems. |
| Microsoft Exchange 2000 Enterprise Server | Builds on the powerful Exchange messaging and collaboration technology by introducing several important new features, and further increasing the reliability, scalability, and performance of its core architecture. Other features enhance the integration of Exchange 2000 with Microsoft Windows 2000, Microsoft Office 2000, and the Internet. |
| Microsoft Application Center 2000 | Provides a deployment and management tool for high-availability Web applications. |
| Microsoft Internet Security and Acceleration Server 2000 | Provides secure, fast, and manageable Internet connectivity. Internet Security and Acceleration Server integrate an extensible, multilayer enterprise firewall and a scalable high-performance Web cache. It builds on Windows 2000 security and directory for policy-based security, acceleration, and management of internetworking. |
| Microsoft Commerce Server 2000 | Provides an application framework, sophisticated feedback mechanisms, and analytical capabilities. |

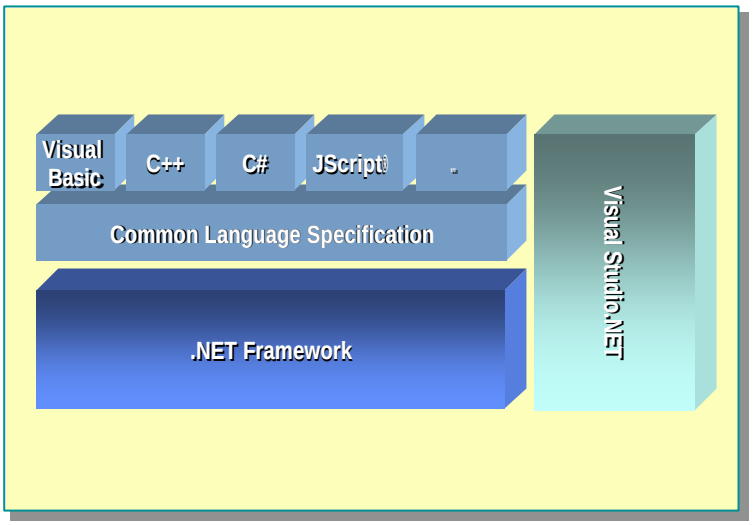
## Visual Studio.NET

Visual Studio.NET provides a high-level development environment for building applications on the .NET Framework. It provides key enabling technologies to simplify the creation, deployment, and ongoing evolution of secure, scalable, highly available Web applications and Web Services.

## Windows

The next generation of Microsoft Windows® will provide the foundation for developers who want to create new .NET applications and services.

# Overview of the .NET Framework




Before COM, applications were completely separate entities with little or no integration. By using COM, you can integrate components within and across applications by exposing common interfaces. However, as a developer, you must still write the code to wrap, manage, and clean up after components and objects.

## Building Components in the .NET Framework

In the .NET Framework, components are built on a common foundation. You no longer need to write the code to allow objects to interact directly with each other. In addition, you no longer need to write component wrappers in the .NET environment, because components do not use wrappers. The .NET Framework can interpret the constructs that developers are accustomed to using in object-oriented languages. The .NET Framework fully supports class, inheritance, methods, properties, events, polymorphism, constructors, and other object-oriented constructs.

## The Common Language Specification

The Common Language Specification (CLS) defines the common standards to which languages and developers must adhere if they want their components and applications to be widely useable by other .NET languages.

## Visual Studio.NET

In the .NET Framework, Visual Studio.NET provides the tools you can use for rapid application development.

# Benefits of the .NET Framework

- Based on Web Standards and Practices
- Designed Using Unified Application Models
- Easy for Developers to Use
- Extensible Classes

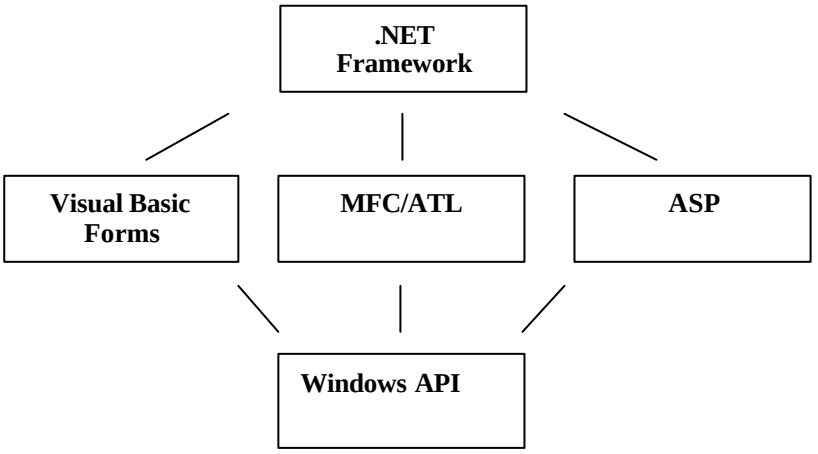
In this topic, you will learn about some of the benefits of the .NET Framework. The NET Framework was designed to meet the following goals.

- **Based on Web standards and practices**

  The .NET Framework fully supports the existing Internet technologies including Hypertext Markup Language (HTML), XML, SOAP, Extensible Stylesheet Language for Transformations (XSLT), Xpath, and other Web standards. The .NET Framework favors loosely connected, stateless Web services.

- **Designed using unified application models**

  A .NET class's functionality is available from any .NET language or programming model.



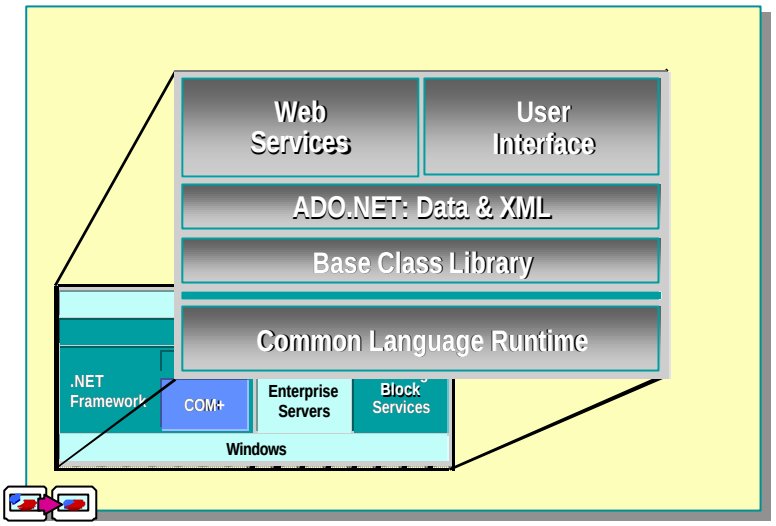
- **Easy for developers to use**

  In the .NET Framework, code is organized into hierarchical namespaces and classes. The Framework provides a common type system, referred to as the unified type system, that is used by any .NET language. In the unified type system, all languages elements are objects. There are no variant types, there is only one string type, and all string data is Unicode. The unified type system is described in more detail in later modules.

- **Extensible classes**

  The hierarchy of the .NET Framework is not hidden from the developer. You can access and extend .NET classes (unless they are sealed) through inheritance. You can also implement cross-language inheritance.

# ◆ The .NET Framework Components

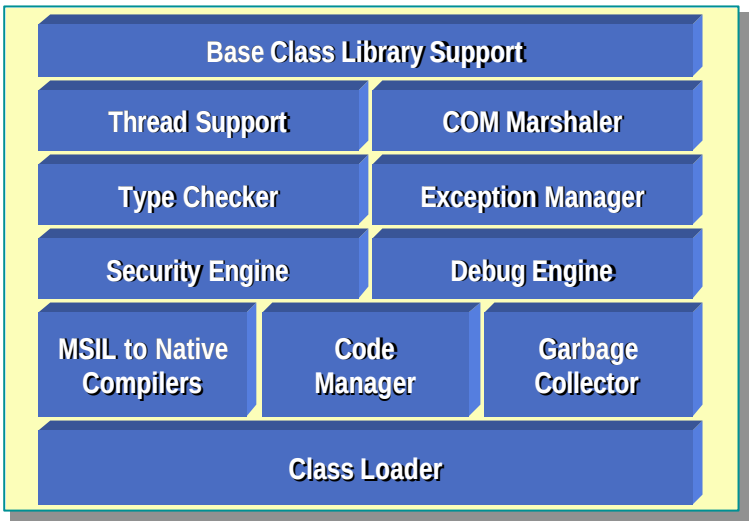



In this section, you will learn about Microsoft's .NET Framework. The .NET Framework is a set of technologies that form an integral part of the Microsoft .NET platform. It provides the basic building blocks for developing Web applications and Web services.

This section includes the following topics:

- Common Language Runtime
- Base Class Library
- ADO.NET: Data and XML
- Web Forms and Services
- User Interface

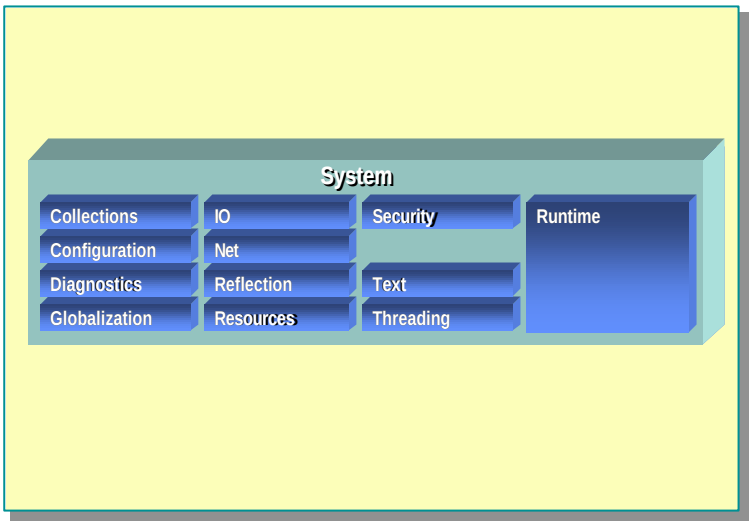
# Common Language Runtime




The Common Language Runtime simplifies application development, provides a robust and secure execution environment, supports multiple languages, and simplifies application deployment and management. The environment is also referred to as a managed environment, one in which common services, such as garbage collection and security, are automatically provided. The Common Language Runtime features are described in the following table.

| Component | Description |
|---|---|
| Class loader | Manages metadata, as well as the loading and layout of classes. |
| Microsoft intermediate language (MSIL) to native compiler | Converts MSIL to native code (Just-in-Time). |
| Code manager | Manages code execution. |
| Garbage collector (GC) | Provides automatic lifetime management of all of your objects. This is a multiprocessor, scalable garbage collector. |
| Security engine | Provides evidence-based security, based on the origin of the code in addition to the user. |
| Debug engine | Allows you to debug your application and trace the execution of code. |
| Type checker | Will not allow unsafe casts or uninitialized variables. MSIL can be verified to guarantee type safety. |
| Exception manager | Provides structured exception handling, which is integrated with Windows Structured Exception Handling (SEH). Error reporting has been improved. |
| Thread support | Provides classes and interfaces that enable multithreaded programming. |
| COM marshaller | Provides marshalling to and from COM. |
| Base Class Library (BCL) support | Integrates code with the runtime that supports the BCL. |

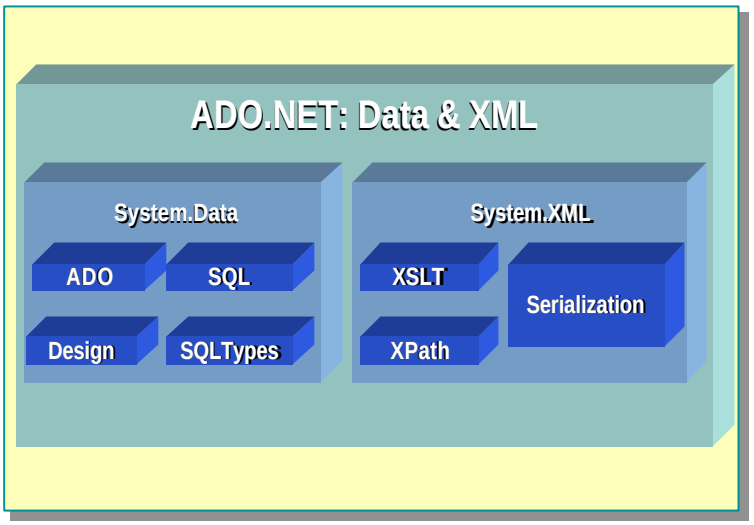
# Base Class Library




The Base Class Library (BCL) exposes features of the runtime and provides other high-level services that every programmer needs through namespaces. For example, the **System.IO** namespace contains input/output (I/O) services.

In the **System.IO** namespace, all of the base data types, such as **int** and **float**, are defined for the platform. Inside the **System.IO** namespace, there are other namespaces that provide various runtime features. The **Collections** namespace provides sorted lists, hash tables, and other ways to group data. The **IO** namespace provides file I/O, streams, and so on. The **Net** namespace provides Transmission Control Protocol/Internet Protocol (TCP/IP) and sockets support. For more information about namespaces, search for "namespaces" in the .NET Framework SDK Help documents.

# ADO.NET: Data and XML




ADO.NET is the next generation of ActiveX® Data Object (ADO) technology. ADO.NET provides improved support for the disconnected programming model. It also provides rich XML support.

## System.Data Namespace

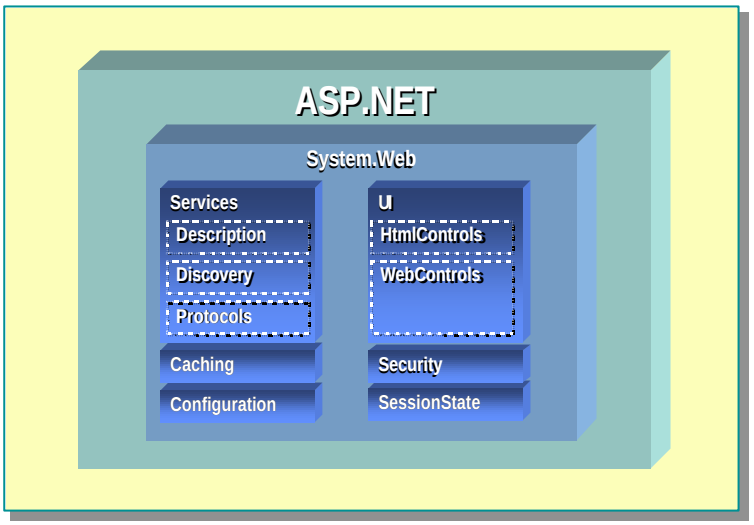
The **System.Data** namespace consists of classes that constitute the ADO.NET object model. At a high level, the ADO.NET object model is divided into two layers: the connected layer and the disconnected layer.

The **System.Data** namespace includes the **DataSet** class, which represents multiple tables and their relations. These **DataSets** are completely self-contained data structures that can be populated from a variety of data sources. One data source could be XML, another could be OLEDB, and a third data source could be the direct adapter for SQL Server.

## System.Xml Namespace

The **System.Xml** namespace provides support for XML. It includes an XML parser and a writer, which are both W3C-compliant. The Extensible Stylesheet Language (XSL) transformation is provided by the **XSLT** namespace. The implementation of XPath allows data graph navigation in XML. The **Serialization** namespace provides the entire core infrastructure for Web Services, including features such as moving back and forth from objects to an XML representation.

# Web Forms and Services




Microsoft ASP.NET is a programming framework built on the Common Language Runtime that can be used on a server to build powerful Web Applications. ASP.NET Web Forms provide an easy and powerful way to build dynamic Web user interfaces (UIs). ASP.NET Web Services provide the building blocks for constructing distributed Web-based applications. Web Services are based on open Internet standards, such as HTTP and XML.

The Common Language Runtime provides built-in support for creating and exposing Web Services by using a programming abstraction that is consistent and familiar to both ASP Web Forms and Visual Basic developers. The resulting model is both scalable and extensible. This model is based on open Internet standards (HTTP, XML, SOAP, SDL) so that it can be accessed and interpreted by any client or Internet-enabled device. Some of the more common ASP.NET classes are described in this topic as follows:

## System.Web

In the **System.Web** namespace, there are lower-level services such as caching, security, configuration, and others that are shared between Web Services and Web user interface (UI).

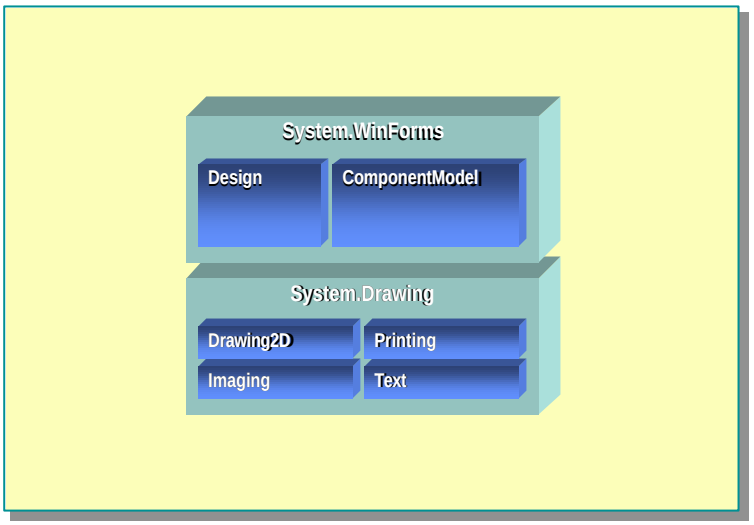
## System.Web.Services

The **System.Web.Services** classes handle Web services such as protocols and discovery.

## System.Web.UI

The **System.Web.UI** namespace provides two classes of controls: HTML controls and Web controls. The HTMLControls give you direct mapping of HTML tags, such as input. There are also WebControls that allow you to structure controls with templates (for example, a grid control).

# User Interface for Windows




## System.WinForms Classes

You can use the **System.WinForms** classes to build the client user interface (UI). This class lets you implement the standards Windows UI in your .NET applications.

## System.Drawing Classes

You can use the **System.Drawing** class to access the new GDI+ features. This class provides support for the next generation of Graphics Device Interface (GDI) two-dimensional graphics. It also provides native support for Graphics Interchange Format (GIF), Tagged Image File Format (TIFF), and other formats.

# Languages in the .NET Framework

- **C# - Designed for .NET**
  New component-oriented language
- **Managed Extensions to C++**
  Enhanced to provide more power and control
- **Visual Basic.NET**
  New version of Visual Basic with substantial language innovations
- **JScript.NET**
  New version of JScript that provides improved performance and productivity
- **Third-party Languages**

The .NET Framework provides support for several programming languages. C# is the programming language specifically designed for the .NET platform, but C++ and Visual Basic have also been upgraded to fully support the .NET Framework.

| Language | Description |
|---|---|
| C# | C# was designed for the .NET platform and is the first modern component–oriented language in the C and C++ family. It can be embedded in ASP.NET pages. Some of the key features of this language include classes, interfaces, delegates, boxing and unboxing, namespaces, properties, indexers, events, operator overloading, versioning, attributes, unsafe code, and XML documentation generation. No header or Interface Definition Language (IDL) files are needed. |
| Managed Extensions to C++ | The managed C++ is a minimal extension to the C++ language. This extension provides access to the .NET Framework that includes garbage collection, single-implementation inheritance, and multiple-interface inheritance. This upgrade also eliminates the need to write " plumbing" code for components. It offers low-level access where useful. |
| Visual Basic.NET | Visual Basic.NET provides substantial language innovations over previous versions of Visual Basic. Visual Basic.NET supports inheritance, constructors, polymorphism, constructor overloading, structured exceptions, stricter type checking, free threading, and many other features. There is only one form of assignment? no **Let** or **Set** methods. There are new Rapid Application Development (RAD) features such as XML Designer, Server Explorer, and Web Forms designer available from Visual Studio.NET to Visual Basic. With this release, Visual Basic Scripting Edition provides full Visual Basic functionality. |
| Microsoft JScript.NET | JScript.NET is rewritten to be fully .NET aware. It includes support for classes, inheritance, types, and compilation. It provides improved performance and productivity features. JScript.NET is also integrated with Visual Studio.NET. You can take advantage of any .NET Framework class in JScript.NET. |
| Third-party languages | Several third-party languages are supporting the .NET platform. These languages include APL, COBOL, Pascal, Eiffel, Haskell, ML, Oberon, Perl, Python, Scheme, and SmallTalk. |

# Review

- Introduction to the .NET Platform
- Overview of the .NET Framework
- Benefits of the .NET Framework
- The .NET Framework Components
- Languages in the .NET Framework

1. What is the .NET platform?

2. What are the core technologies in the .NET platform?

3. List the components of the .NET Framework.

4. What is the purpose of Common Language Runtime?

5. What is the purpose of Common Language Specification?

6. What is a Web Service?

7. What is a managed environment?