

[**Common Language Runtime(CLR)**](https://www.geeksforgeeks.org/common-language-runtime-clr-in-c/)**:** **Common Language Infrastructure" or CLI is a platform on which the .Net programs are executed.** It is the run-time environment in the .NET Framework that runs the codes and helps in making the development process easier by providing the various services such as **remoting, thread management, type-safety, memory management, robustness, Exception Handling etc.**

**Framework Class Library(FCL):**  Also called the **Assemblies.** It is just like the header files in C/C++ and **packages in the java**. **It is the collection of reusable libraries and methods, etc that can be integrated with CLR**.

A .NET application is platform dependent because of the .NET framework which is only able to run on the Windows-based operating system. The .Net application is platform independent also because of **Mono framework**. Using Mono framework the .Net application can run on any Operating System including windows. **Mono framework is a third party software** developed by [**Novell Company**](https://www.microfocus.com/novell/) which is now a part of [**Micro Focus Company**](https://www.microfocus.com/novell/). It is a paid framework.

**Types of Applications :** Mainly the applications which are built in .Net framework is divided into the following three categories :

* **WinForms :** Form – Based applications are considered under this category. In simple terms, we can say **client based applications which read and writes** the file system comes under this category.
* **ASP .NET :** Web-Based applications come under this category. ASP.Net is a framework for web and it provides the awesome **integration of HTML, CSS and** J**avaScript** which makes it useful to develop the web applications, websites and web services. **Web services were added in .Net Framework 2.0 and considered as a part of ASP.NET web applications.**
* **ADO .NET :** It includes **the application which are developed to communicate with the database like MS SQL Server**, Oracle etc. comes. It mainly consists of classes that can be used to connect, retrieve, insert and delete data.

**WPF (Windows Presentation Foundation):** called **Avalon”**

graphical subsystem, UI (User Interface)

**WCF (Windows Communication Foundation) :** called **Indigo**.

used to transmit the data as **asynchronous** from one service endpoint to another service point.

**WF (Windows Workflow Foundation)**

workflows within .Net applications.

**LINQ (Language Integrated Query) :**

it is a query language used to make the query for data sources

**Entity Framework :**

It enables the .Net developer to work with database **using .Net objects**. Before entity framework, .Net developers have performed a lot of things related database.

**Parallel LINQ (Language Integrated Query) :** It comes in .Net Framework version 4.0 and also termed as PLINQ. It provides a concurrent query execution engine for **LINQ**. It executes the **LINQ** in parallel

In order to memorize whole page life cycle process of Asp.net, keep store this word "**SILVER**" in your memory, which is defined as

**PAGE REQUEST:**

The page request occurs before the page life cycle begins.

**S = Start**

**I  = Initialization**

**L = Load**

**V = Validate**

**E = Event Handlers**

**R = Render**

**PAGE REQUEST:**

The page request occurs before the page life cycle begins.

**START:**

In the start stage, page properties such as Request and Response are set. At this stage, the page also determines whether the request is a postback or a new request and sets the IsPostBack property.

**INITIALIZATION:**

During page initialization, controls on the page are available and each control's UniqueID property is set. A master page and themes are also applied to the page if applicable. If the current request is a postback, the postback data has not yet been loaded and control property values have not been restored to the values from view state

**LOAD:**

During load, if the current request is a postback, control properties are loaded with information recovered from view state and control state.

**VALIDATE & EVENT HANDLING:**

If the request is a postback, control event handlers are called. After that, the Validate method of all validator controls is called, which sets the IsValid property of individual validator controls and of the page. There is an exception to this sequence: the handler for the event that caused validation is called after validation.

**RENDER:**

Before rendering, view state is saved for the page and all controls. During the rendering stage, the page calls the Render method for each control, providing a text writer that writes its output to the OutputStream object of the page's Response property.

**UNLOAD:**

ASP.NET goes through a series of stages in the life cycle of each page.

* **Page request**. The user requests a page. ASP.NET decides whether to compile it or serve it from a cache.
* **Page Start.** The Request and Response objects are created.
* **Page Initialization.** All page controls are initialized, and any themes are applied.
* **Page Load.** ASP.NET uses the view state and control state properties to set the control properties. Default values are set in the controls.
* **Postback event handling.** This event is triggered if the same page is loaded again.
* **Rendering.** ASP.NET saves the view state for the page and writes the output of rendering to the output stream. It happens just before the complete web page is sent to the user.
* **Unload.** The rendered page gets sent to the client. ASP.NET unloads page properties and performs cleanup. All unwanted objects are removed from memory.

**. List the events in page life cycle.**

1) Page\_PreInit: Raised after the start stage is complete and before the initialization stage begins.   
2) Page\_Init  
3) Page\_InitComplete  
4) Page\_PreLoad  
5) Page\_Load  
6) Page\_LoadComplete  
7) Page\_PreRender  
8) Render

**View state**

View state is where data is used to preserve page values and control values of Web Forms during postback event handling. Data can be stored as hidden fields on the client web page.

If you want to NOT maintain the ViewState, include the **directive <%@ Page Enable View State="false"%>** at the top of an .**aspx page**

**Protected Configuration f**eature used to secure connection string information.

**What are the different Validators in ASP.NET**

There are two types of validation in ASP.NET:

* Client-Side Validation
* Server-Side Validation

The following are the **Validation Controls** in ASP.NET:

* RequiredFieldValidator Control
* CompareValidator Control
* RangeValidator Control
* RegularExpressionValidator Control
* CustomFieldValidator Control
* ValidationSummary

**Data source controls** **represent data** in a data source, like a database or an XML file. These controls derive from the abstract class DataSourceControl

**Caching:**

ASP.NET has 3 kinds of caching :

1. Output Caching(Page Level Caching)
2. Fragment Caching(Partial-Page Output Caching)
3. Data Caching.( Data caching means caching data from a data source.)

 Fragment caching is useful when you need to cache only a subset of a page.

<%@ OutputCache Duration="60" VaryByParam="CategoryID" %>

If your page generates different **content based on** the query string, then you need to put CategoryID". If you want to cache a **new version of the page based** on any differences in the query string parameters, use VaryByParam = "\*",

## **DataReader or DataSet**

**DataSet:** **disconnected,** It is a collection of data tables that contain the data. It is used to fetch data without interacting with a Data Source that's why, it also known as **disconnected** data access method.

**DataReader** object to be **connected,** This class is used to read data from SQL Server database.

The **DataAdapter** works as a bridge between a DataSet and a Data source to retrieve data. DataAdapter is a class that represents a set of SQL commands and a database connection.

**Difference between ExecuteScalar and ExecuteNonQuery?**

ExecuteScalar returns output value where as ExecuteNonQuery does not return any value but the number of rows affected by the query. ExecuteScalar used for fetching a single value and ExecuteNonQuery used to execute Insert and Update statements.

**.aspx:** (web page)A Web form that may include a code-behind file

**.ascx:** A Web user control.

**.asmx: (method)**An ASP.NET Web service; it may include a corresponding code-behind file as well that includes its code.

**.ashx:** A paAge for implementing a generic handler.

**.asax:** This file allows you to write code **to handle global** ASP.NET application-level events.

**State Management**

There are two ways to maintain a state in .NET, Client-Based state management and Server-Based state management.

The following techniques can be used to implement the Client-Based state management:

* View State
* Hidden Fields
* Cookies
* **Query Strings**
* **Control State**

The following techniques can be used to implement Server-Based state management:

* Application State
* Session State
* **Profile Properties**

**The trip of a Web page** from the client to the server and then back to the client is **known as a round trip.**

The major built-in objects in ASP.NET are as follows:

* Application
* Request
* Response
* Server
* Session
* **Context**
* **Trace**

**Code-behind feature in ASP.NET:**

The code-behind feature of ASP.NET enables you to divide an ASP.NET page into two files - one consisting of the presentation data, and the second, which is also called the code-behind file, consisting of all the business logic. The presentation data contains the interface elements, such as HTML controls and Web server controls, and the code-behind contains the event-handling process to handle the events that are fired by these controls. **The file that contains the presentation data has the .aspx extension.**

ASP .NET MVC supports different **view engines**, which are responsible for rendering the View’s HTML. **The default view engine is called Razor and was designed for ASP .NET MVC**

**Assemblies:**

The following are the two types of assemblies:

* **Private Assembly** - Refers to the assembly that is used by a **single application.** Private assemblies are kept in a local folder in which the client application has been installed.
* **Public or Shared Assembly** - Refers to the assembly that is allowed to be shared by multiple applications. A shared assembly must reside in Global Assembly Cache (GAC) with a strong name assigned to it. **versioning must in GAC**

**Manifest:**

Assembly metadata is stored in **Manifest.**

**Can one DLL file contain the compiled code of more than one .NET language?**

No, a DLL file can contain the compiled code of only one programming language.

**EXE**

1. It is an **executable file**, which can be run independently.
2. It cannot be reused in an application.
3. It has a main function.

**DLL**

1. It is **Dynamic Link Library** that is used as a part of EXE or other DLLs. **It cannot be run independently.**
2. It runs in the application process memory, so it is called as in-process component.
3. It can be reused in an application.
4. It does not have a main function.

**What is the difference between Skins and Css files?**

Css is applied to HTML controls whereas skins are applied to server controls.

**What is a User Control?**

User controls are reusable controls, similar to web pages. They cannot be accessed directly.

**Where do you store your connection string information?**

The connection string can be stored in configuration files (web.config).