.NET Framework

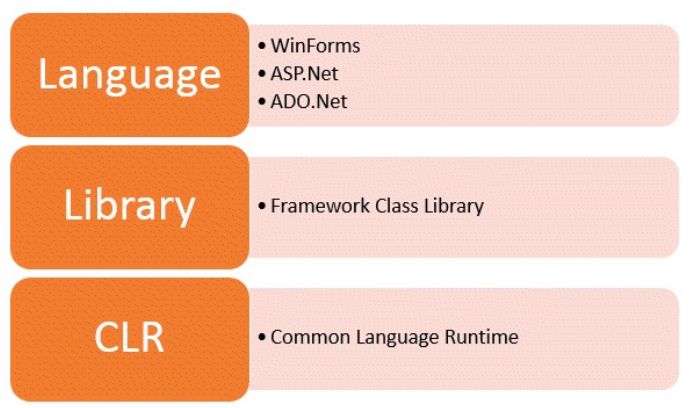
The .Net framework is a software development platform developed by Microsoft. The framework was meant to create applications, which would run on the Windows Platform. The first version of the .Net framework was released in the year 2000. The version was called .Net framework 1.0.

The .Net framework has come a long way since then, and the current version is 4.6.1. The .Net framework can be used to create both - Form based and Web based applications. Web services can also be developed using the .Net framework.

The framework also supports various programming languages such as Visual Basic and C#. So developers can choose and select the language to develop the required application.

**.Net Framework Architecture**

The basic architecture of the .Net framework is as shown below.



The architecture of the .Net framework is based on the following key components;

1. Common Language Runtime - The “Common Language Infrastructure” or CLI is a platform on which the .Net programs are executed. The CLI has the following key features:

**Exception Handling** - Exceptions are errors which occur when the application is executed. Examples of exceptions are: If an application tries to open a file on the local machine, but the file is not present. If the application tries to fetch some records from a database, but the connection to the database is not valid.

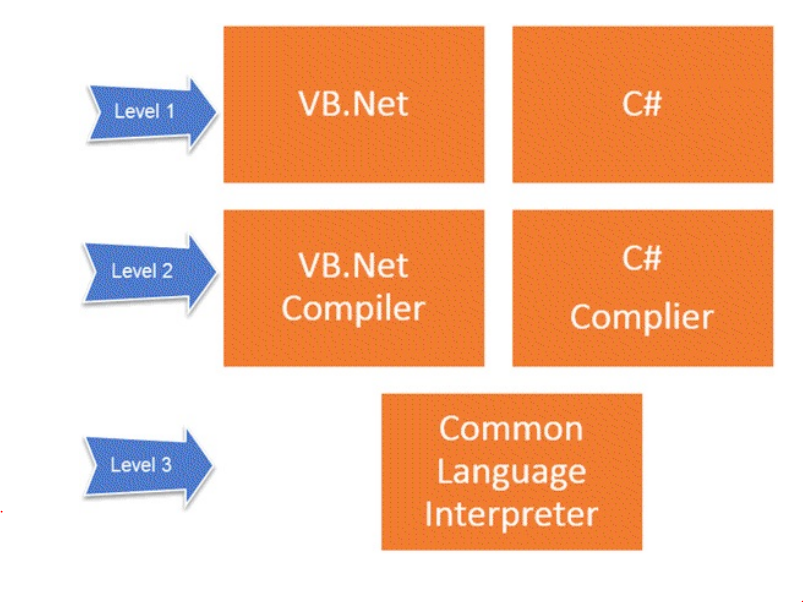
**Garbage Collection** - Garbage collection is the process of removing unwanted resources when they are no longer required. Examples of garbage collection are:

* A File handle which is no longer required. If the application has finished all operations on a file, then the file handle may no longer be required.
* The database connection is no longer required. If the application has finished all operations on a database, then the database connection may no longer be required. Working with Various programming languages – As noted in an earlier section, a developer can develop an application in a variety of .Net programming languages.

1. **Language -** The first level is the programming language itself, the most common ones are VB.Net and C#.

**2.** **Compiler –** There is a compiler which will be separate for each programming language. So underlying the VB.Net language, there will be a separate VB.Net compiler. Similarly for C#, you will have another compiler.

**3.** **Common Language Interpreter –** This is the final layer in .Net which would be used to run a .net program developed in any programming language. So the subsequent compiler will send the program to the CLI layer to run the .Net application.



1. **Class Library -** The .NET Framework includes a set of standard class libraries. A class library is a collection of methods and functions that can be used for the core purpose. For example, there is a class library with methods to handle all file level operations. So there is a method which can be used to read the text from a file. Similarly, there is a method to write text to a file.
2. **Languages -** The types of applications that can be built in the .Net framework are classified broadly into the following categories. WinForms – This is used for developing Forms-based applications, which would run on an end user machine. Notepad is an example of a client-based application. ASP.Net – This is used for developing web based applications, which are made to run on any browser such as Internet Explorer, Chrome or Firefox. The Web application would be processed on a server, which would have Internet Information Services Installed. Internet Information Services or IIS is a Microsoft component which is used to execute an ASP.net application. The result of the execution is then sent to the client machines, and the output is shown in the browser. ADO.Net – This technology is used to develop applications to interact with Databases such as Oracle or Microsoft SQL Server.

Different versions of the .Net framework

Below is the table of .Net framework versions, which have been released with their release dates. Every version has relevant changes to the framework. For example, in framework 3.5 and onwards a key framework called the Entity framework was released. This framework is used to change the approach in which the applications are developed while working with databases.

The biggest advantage of the .Net framework is that it supports Windows platform. Almost everyone works with Windows machines. Microsoft always ensures that .Net frameworks are in compliance with all the supported Windows operating systems. The following design principles of the .Net framework is what makes it very relevant to create .Net based applications.

1. **Interoperability -** The .Net framework provides a lot of backward support. Suppose if you had an application built on an older version of the .Net framework, say 2.0. And if you tried to run the same application on a machine which had the higher version of the .Net framework, say 3.5. The application would still work. This is because with every release, Microsoft ensures that older framework versions gel well with the latest version.

2. **Portability-** Applications built on the .Net framework can be made to work on any Windows platform. And now in recent times, Microsoft is also envisioning to make Microsoft products work on other platforms, such as iOS and Linux.

3. **Security -** The .NET Framework has a good security mechanism. The inbuilt security mechanism helps in both validation and verification of applications. Every application can explicitly define their security mechanism. Each security mechanism is used to grant the user access to the code or to the running program.

4. **Memory management -** The Common Language runtime does all the work or memory management. The .Net framework has all the capability to see those resources, which are not used by a running program. It would then release those resources accordingly. This is done via a program called the “Garbage Collector” which runs as part of the .Net framework. The garbage collector runs at regular intervals and keeps on checking which system resources are not utilized, and frees them accordingly.

5. **Simplified deployment -** The .Net framework also have tools, which can be used to package applications built on the .Net framework. These packages can then be distributed to client machines. The packages would then automatically install the application.

**Data Types**

The C# language comes with a set of Basic data types. These data types are used to build values which are used within an application. Let’s explore the basic data types available in C#.

1. **Integer –** An Integer data types is used to work with numbers. In this case, the numbers are whole numbers like 10, 20 or 30. In C#, the datatype is denoted by the **Int32** keyword.
2. **Double -** A double data type is used to work with decimals. In this case, the numbers are whole numbers like 10.11, 20.22 or 30.33. In C#, the datatype is denoted by the keyword “Double“. Below is an example of this datatype .
3. **Boolean** - A boolean data type is used to work with Boolean values of true and false. In C#, the datatype is denoted by the Boolean keyword. Below is an example of this datatype can be used.
4. **String -** A String data type is used to work with String values. In C#, the datatype is denoted by the keyword ‘String’. Below is an example of this datatype.

**Enumeration**

An enumeration is used in any programming language to define a constant set of values. For example, the days of the week can be defined as an enumeration and used anywhere in the program. In C#, the enumeration is defined with the help of the keyword ‘enum‘.

**Operators and Variables**

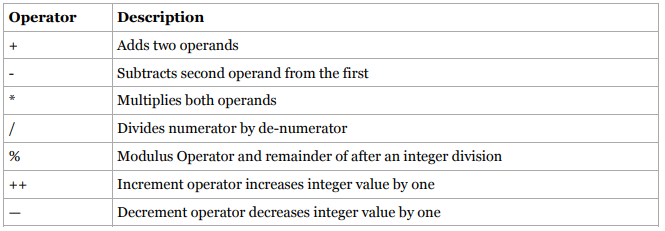
A variable is a name given to a storage area that is used to store values of various data types. Each variable in C# needs to have a specific type, which determines the size and layout of the variable’s memory.

For example, a variable can be of the type String, which means that it will be used to store a string value. Based on the data type, specific operations can be carried out on the variable.

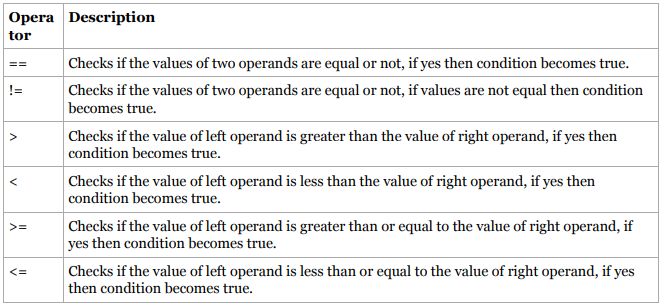
For instance, if we had an Integer variable, then operations such as addition and subtraction can be carried out on the variable. One can declare multiple variables in a program.

**Operators** are used to perform operations on values of various data types. For example, to perform the addition of 2 numbers, the + operator is used. Let’s see the table of operators available for the various data types

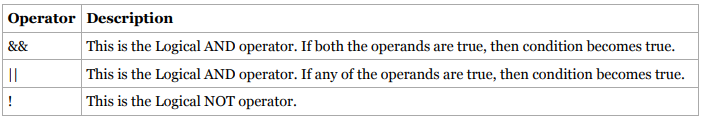
1. **Arithmetic Operators –** These are operators used for performing mathematic operations on numbers. Below is the list of operators available in C#.



1. **Relational Operators –** These are operators used for performing Relational operations on numbers. Below is the list of relational operators available in C#.

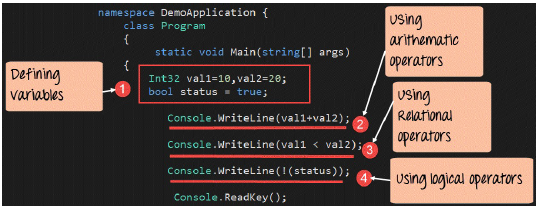


1. **Logical Operators –** These are operators used for performing Logical operations on values. Below is the list of operators available in C#.



Let’s look at a quick example of how the operators can be used in .Net.

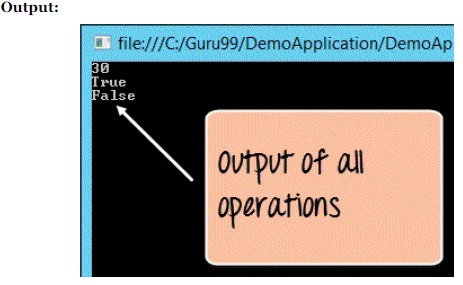
In our example, we will define 2 Integer variables and one Boolean variable. We will then perform the following operations



**Code Explanation:-**

1. Two Integer variables are defined, one being val1 and the other being val2. These will be used to showcase relational and arithmetic operations. A Boolean variable is defined to showcase logical operations.
2. An example of the arithmetic operation is shown wherein the addition operator is carried out on val1 and val2. The result is written to the console.
3. An example of the relational operation is shown wherein the less than operator is carried out on val1 and val2. The result is written to the console.
4. An example of the logical operation is shown, wherein the logical operator (!) is applied to the status variable. The logical NOT operator reverses the current value of any Boolean value. So if a Boolean value is ‘true’, the logical NOT will return the value ‘false’ and vice versa. In our case since the value of the status variable is ‘true’, the result will show ‘false’. The result is written to the console.

If the above code is entered properly and the program is executed successfully, the output will be displayed.



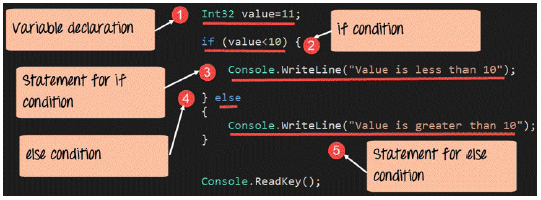
**Flow Control and conditional statements**

Flow control and conditional statements are available in any programming language to alter the flow of a program.

For example, if someone want to execute only a particular set of statements based on some certain logic, then Flow control and conditional statements will be useful.

1. **If statement** – The if statement is used to evaluate a boolean expression before executing a set of statements. If an expression evaluates to true, then it will run one set of statements else it will run another set of statements.

In our example below, a comparison is made for a variable called value. If the value of the variable is less than 10, then it will run one statement, or else it will run on another statement.



**Code Explanation:-**

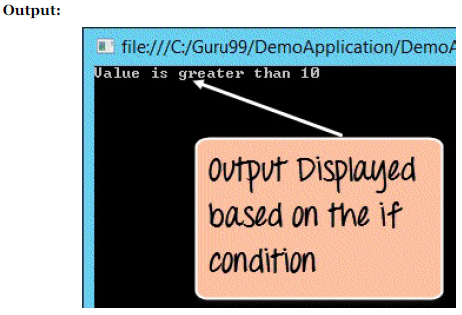
1. We first define a variable called value and set it to the value of 11.

2. We then use the ‘if’ statement to check if the value is less than 10 of the variable. The result will either be true or false.

3. If the if condition evaluates to true, we then send the message “Value is less than 10” to the console.

4. If the if condition evaluates to false, we then send the message “Value is greater than 10” to the console.

If the above code is entered properly and the program is executed successfully, the following output will be displayed.

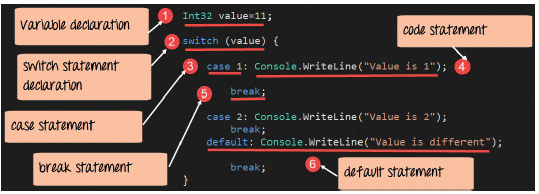


We can clearly see that the ‘if’ statement was evaluated to false. Hence the message “Value is greater than 10” was sent to the console.

2. **Switch statement** – The switch statement is an enhancement to the ‘if’ statement. If you have multiple expressions that need to be evaluated in one shot, then writing multiple ‘if’ statements becomes an issue.

The switch statement is used to evaluate an expression and run different statements based on the result of the expression. If one condition does not evaluate to true, the switch statement will then move to the next condition and so forth.

Let’s see, how this works with the below example. Here, we are again comparing the value of a variable called ‘value.’ We then check if the value is equal to 1, or 2, or something totally different.



**Code Explanation:-**

1. We first define a variable called ‘value’ and set it to the value of 11.

2. We then use the ‘switch’ statement to check the value of the variable ‘value.’

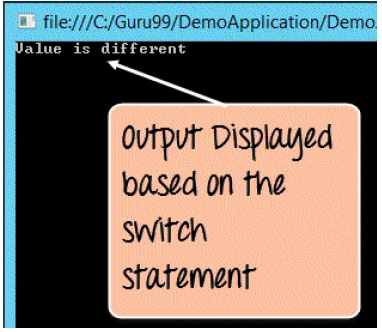
3. Case statements are used to set different conditions. Based on the conditions, a set of statements can be executed. A switch statement can have multiple case conditions. The first case statement checks to see if the value of the variable is equal to 1.

4. If the first case statement is true, then the message “Value is 1” is written to the console.

5. The break statement is used to break from the entire switch statement, once a condition is true.

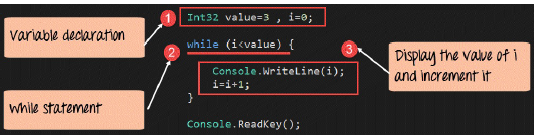
6. The default condition is a special condition. This just means that if no case expression evaluates to true, then run the set of statements for the default condition.

If the above code is entered properly and the program is executed successfully, the following output will be displayed. The output prints the default value “Value is different”, since no condition is satisfied.



3. **While loop** – The while loop is used for iterative purposes. Suppose if you want to repeat a certain set of statements for a particular number of times, then while loop is used.

In our example below, we use the while statement to display the value of a variable ‘i‘. The while statement is used to display the value 3 times.



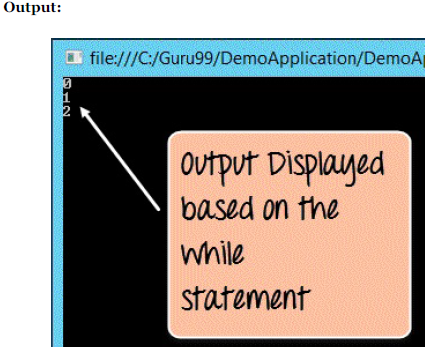
**Code Explanation:-**

1. Two Integer variables are defined, one being value and the other being ‘i‘. The value variable is used as the upper limit to which we should iterate our while statement. And ‘i‘ is the variable which will be processed during the iteration.

2. In the while statement, the value of ‘i‘ is constantly checked against the upper limit.

3. Here we display the value of ‘i‘ to the console. We also increment the value of ‘i‘.

If the above code is entered properly and the program is executed successfully, the following output will be displayed.



Here you can see that the while statement is executed 3 times and incremented at the same time. And each time, it displayed the current value of the variable ‘i‘.

4. **For loop** - The ‘for’ loop is also used for iterative purposes. Suppose if you want to repeat a certain set of statements for a particular number of times, then for loop is used.

In our example below, we use the ‘for’ statement to display the value of a variable ‘i‘. The ‘for’ statement is used to display the value 3 times.

**Code Explanation:-**

1. The ‘for’ keyword is used to start off the ‘for loop’ statement.

2. In the ‘for loop’, we define 3 things. The first is to initialize the value of a variable, which will be used in the ‘for loop’.

3. The second is to compare the value of the ‘i‘ against an upper limit. In our case, the upper limit is the value of 3 (i<3).

4. Finally, we increment the value of ‘i‘ accordingly.

5. Here we display the value of ‘i‘ to the console.

If the above code is entered properly and the program is executed successfully, the following output will be displayed.