.NET Introduction

.NET

- NET is an open source development platform developed by Microsoft to make end-to-end applications.

Applications are divided into

* Console application

* Window application

* Web application

* Mobile application

.NET Framework

- The .NET Framework is a software framework developed by Microsoft that runs primarily on Microsoft windows.
- .NET includes a large class library called Framework class library.
- FCL provides user interface, database connection, cryptography, web development and many more.
- .NET Framework applications execute in CLR

History of .NET

- 1997 : Microsoft wants to develop a new program language to develop desktop, web, mobile apps.

- 1998 : Microsoft has developed “SMC” language.

  Simple Managed code. It contains three features:

  1. Light weight : Easy to implement
  2. Hardware Independent : To able to run on different hardware.
  3. Shortcut code : For perform common activities, some predefined libraries are available.

     Ex: Arrays.Sort(arrayname);

- 1999: SMS → renamed as → C#

  ASP(ASP)

  CE

  C# + ASP + CE are together , it is called as NGWS(Next Generation Windows Services).

2000 : NGWS → renamed as → .NET

2001: Development..

2002: .NET 1.0 was released

2003: .NET 1.1

2005 : .NET 2.0

2006 : .NET 3.0

2007: .NET 3.5

2007: .NET3.5

2010: .NET4.0

2012: .NET4.5

2013: .NET 4.5.1

2014: .NET 4.5.2

2015: .NET 4.6

2016: .NET 4.6.2

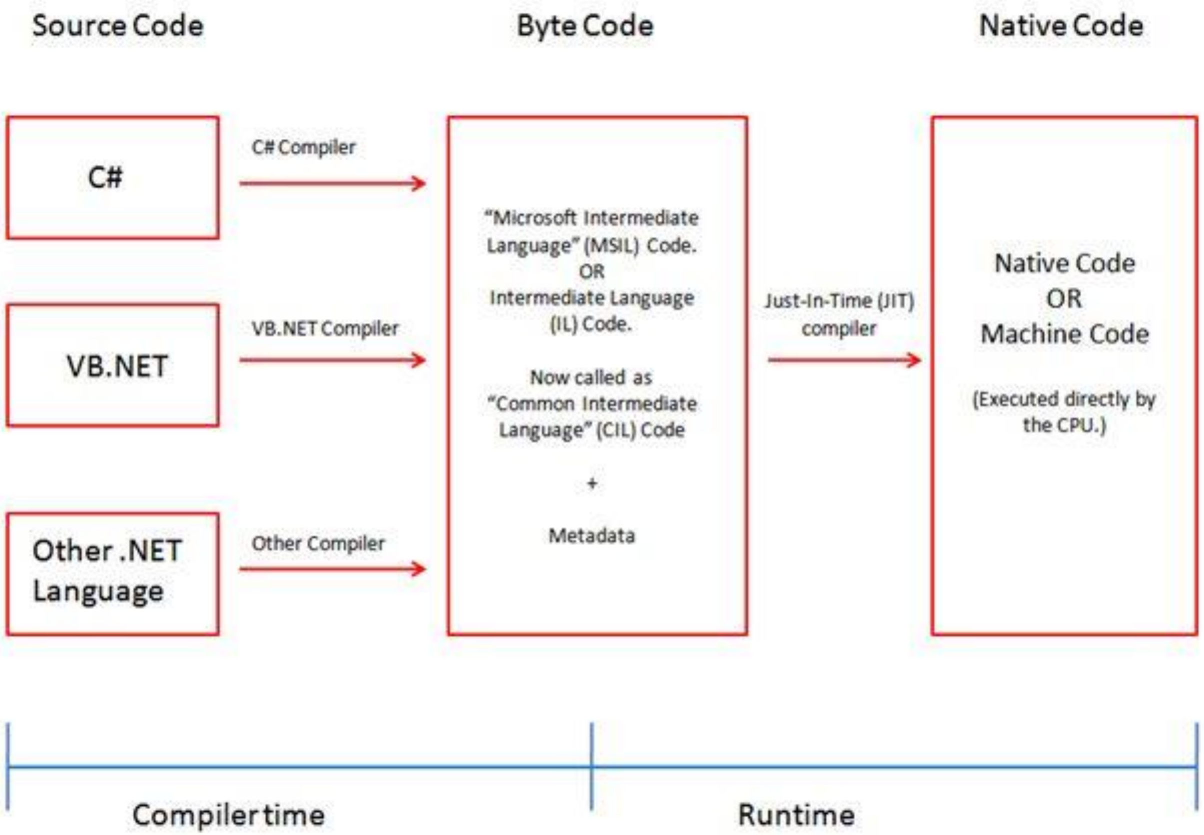
2017: .NET 4.7

2019: .NET 4.8

Version of .NET Framework

Version	Release Date	
1.0	2002	→We cannot develop web application, only develop base application
1.1	2003	->Inbuilt web applications
2.0	20005	-> add Generic collection
3.0	2006	→Add advance technologies(WPF(windows presentation foundation),WCF,Wf)
3.5	2007→	Linq
4.0	2010 ->	Integrated silver light application
4.6	2012→	Metro apps using window 8 operation system
4.7		->Performance and reliability improvement
4.8		

Compilation process


Source Code
Byte Code
Native Code
C#
C# Compiler
VB.NET
VB.NET Compiler
Other .NET Language
Other Compiler
"Microsoft Intermediate Language" (MSIL) Code.
OR
Intermediate Language (IL) Code.
Now called as "Common Intermediate Language" (CIL) Code
+
Metadata
Just-In-Time (JIT) compiler
Native Code
OR
Machine Code
(Executed directly by the CPU.)
Compilertime
Runtime

Compilation process is done in two phases:

- First phase is done by language compilers.
- Second phase is done by JIT(Just-In-Time).

Components of .NET Framework

1. CLR(Common Language Runtime)
2. BCL(Base class Library)

1. CLR :

- CLR is runtime execution engine of .NET. It acts as interface between .NET application and operation system.
- CLR provides many features like manage memory, handle exception.
- Without CLR, we cannot execute application.

CLR provides services like CLS, CTS

CLS(Common language specification):

The code which is there in one .NET framework language can be used in another .NET framework language.

CTS(Common Type System):

* In CTS, it deals with datatypes. Here we have several languages, each & every language have own datatype, one language datatype cannot understand to other language.

- CLR understands all language because CLR having own datatype.

2. BCL(Base class Library):

- It is also known as class library. Class library is collection of reusable types that are closely integrated with CLR.
- Class library provides classes and types that are helpful in performing day to day operation.
- Ex: Dealing with string , primitives types, Database connection, IO operations.

BCL can be divided into two:

* User defined class library : unit part of deployment is called Assembly

* Predefined class library : collection of predefined class & methods.

Memory Management

- Memory management is a specific programming language is responsible for allocating and deallocating memory used by applications without involving developers.
- .NET manages memory automatically.

  * Create objects onto managed memory blocks(heaps).
- Allocates objects onto one of two heaps:

  1. Simple Object Heap(SOH)

  2. Large Object Heap(LOH)
- You allocate onto a heap whenever you the “new” keyword in code.

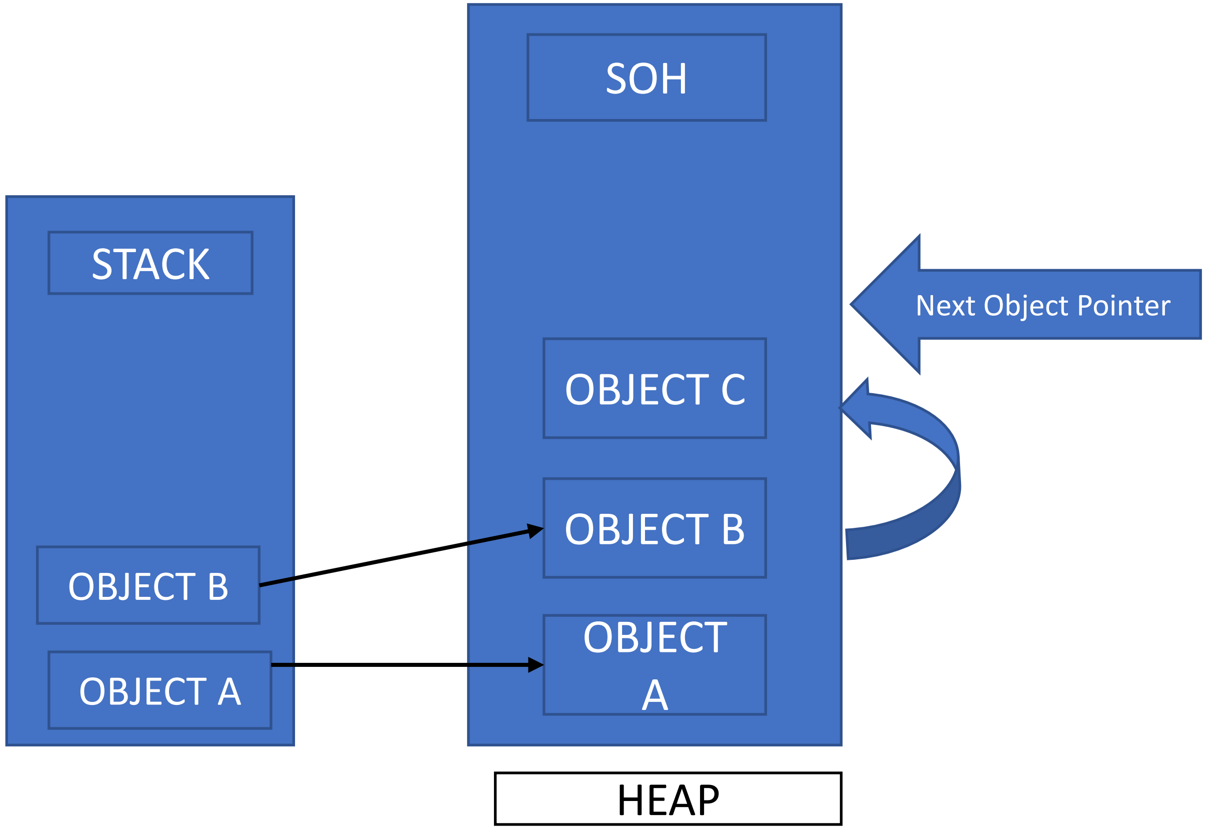
Simple Object Heap:

- Allocation of objects < 85KB, that objects are allocated in SOH.
- Contiguous Heap

* objects allocated consecutively

* Next object pointer maintained.

- SOH

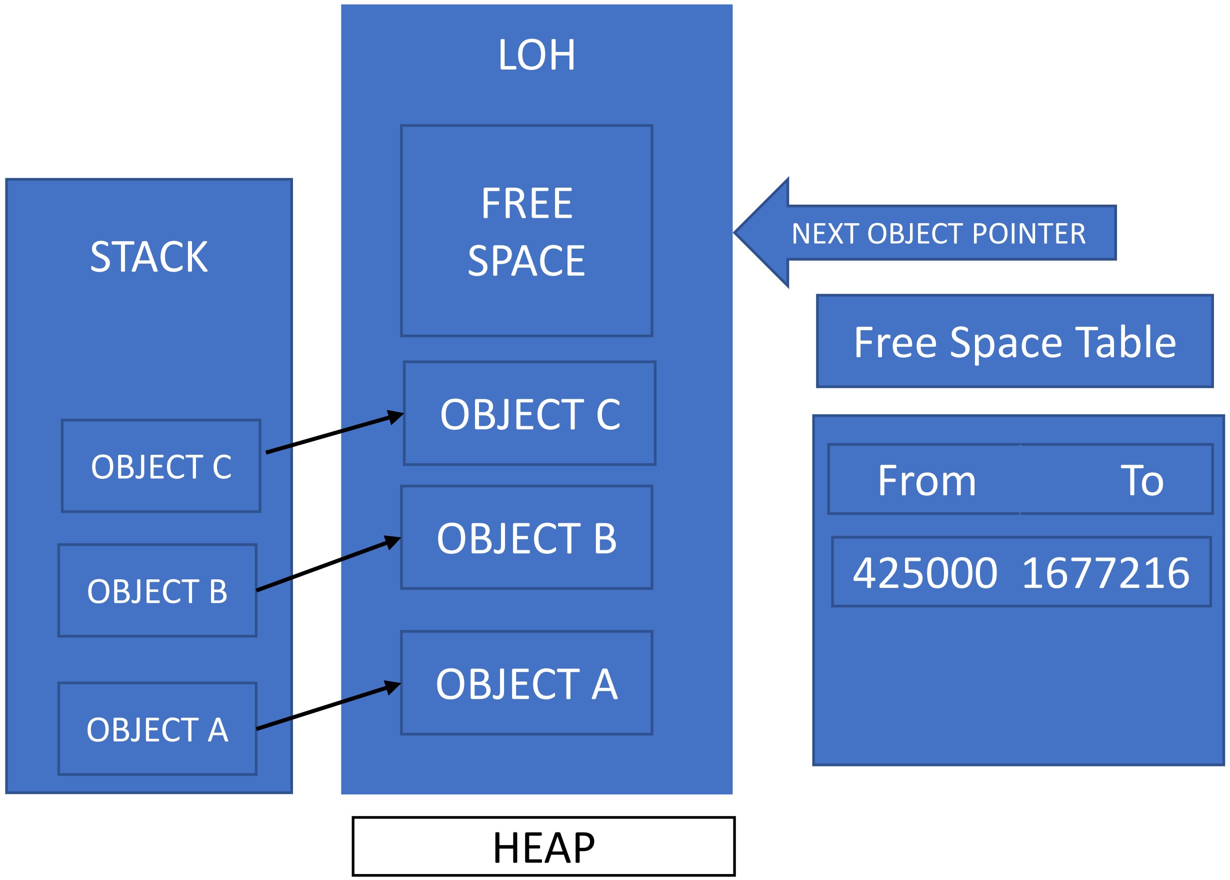


Large Object Heap:

- Allocation of objects >=85KB, that objects are allocated in LOH.
- Non contiguous Heap

  * Objects allocated using Free Space Table
- Garbage collected when LOH Threshold is reached.
- LOH uses a Free Space Table to find where to allocate.

- LOH



From	To
425000	1677216

Garbage Collection:

- Garbage Collection is one of the services that provided by CLR.
- Manages the allocation and release of memory for an application.
- Purpose of Garbage collector is AMM(Automatic memory management).
- To clear memory in two ways:

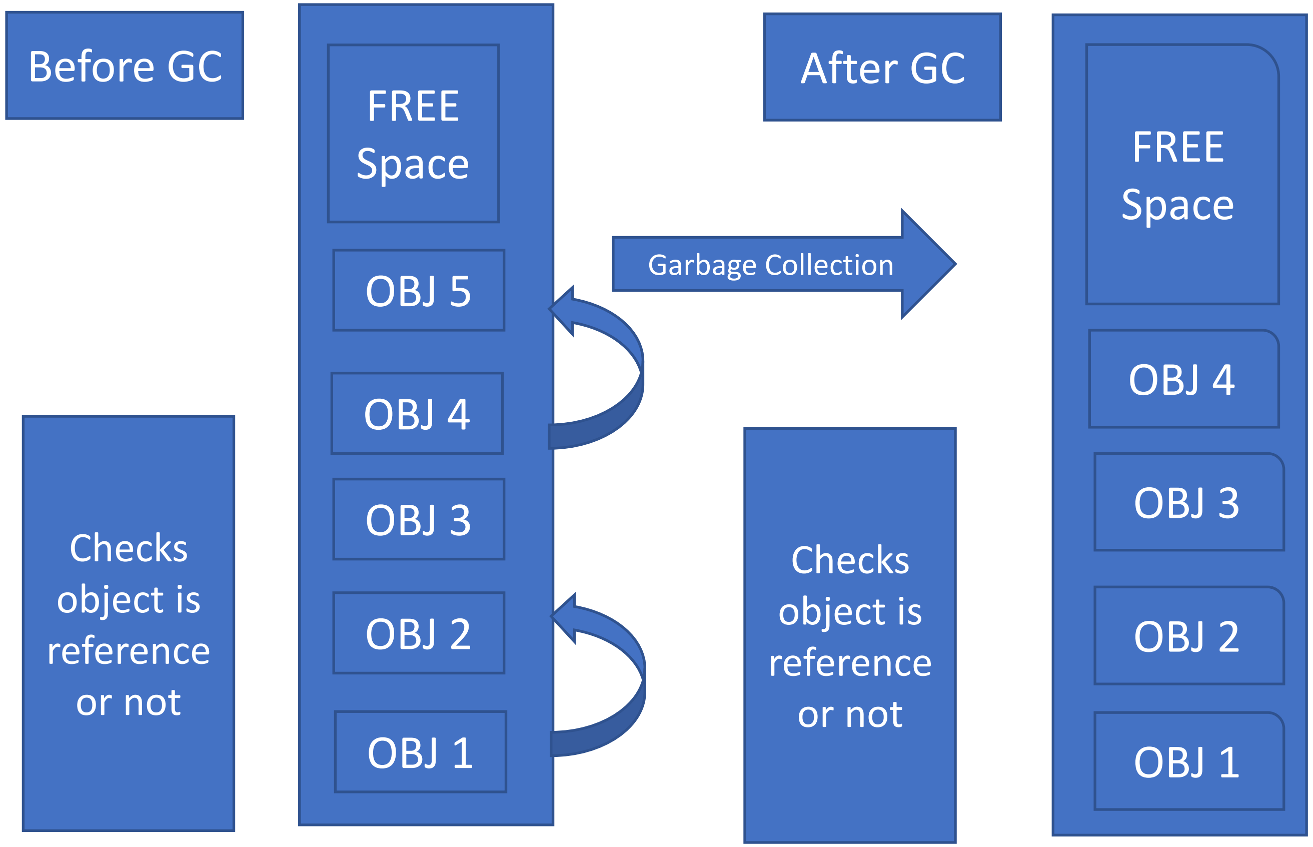
  1. Manually → If you go for manually, it will leads to the errors.
  2. Automatic → We go for AMM, Which will be provided by Garbage collector.

Garbage Collection

Every object holds memory in the managed heap. When the heap is full, the garbage collector(GC) is called. The GC goes through every object to examine

- First, Gc checks for each object. That object refers to reference or null.
- It only remove dead objects.

Garbage collection:



Before GC

FREE Space

OBJ 5

OBJ 4

OBJ 3

OBJ 2

OBJ 1

Checks object is reference or not

Garbage Collection

After GC

FREE Space

OBJ 4

OBJ 3

OBJ 2

OBJ 1

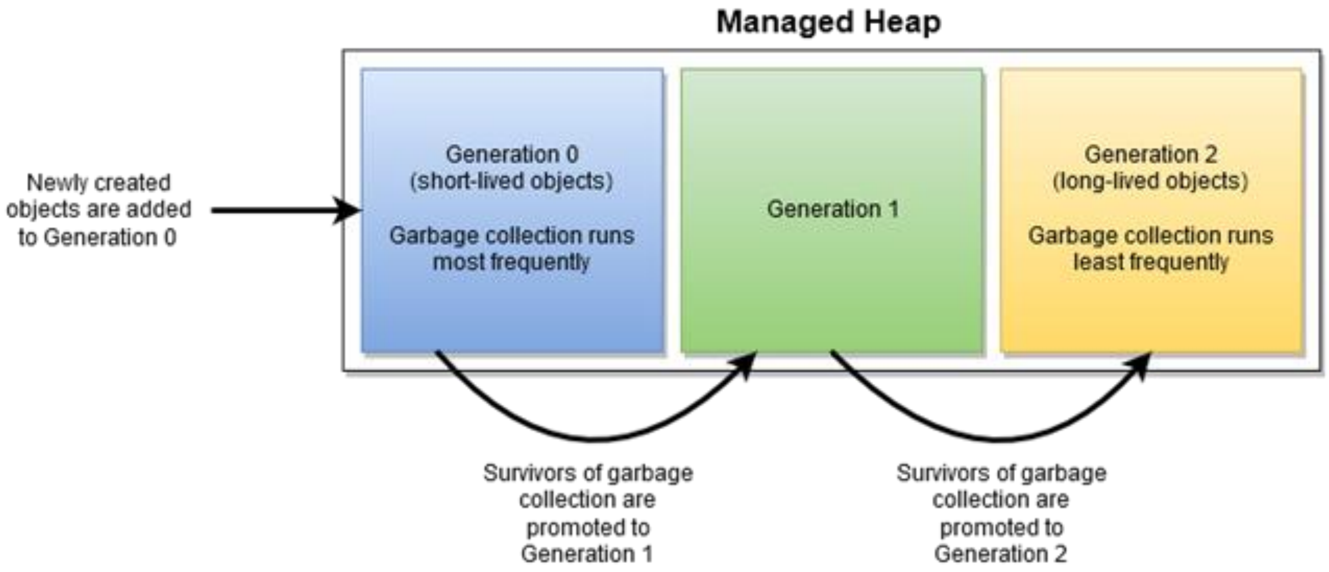
Checks object is reference or not

Heap memory is divided into 3 generations:

1. Generation 0
2. Generation 1
3. Generation 2

Garbage collector has 3 phases:

1. Marking phase : It create list of live object.
2. Relocating phase : In this phase, it updates the reference objects.
3. Compacting phase : It clears unuse objects from heap memory.


Managed Heap
Newly created objects are added to Generation 0
Generation 0 (short-lived objects)
Garbage collection runs most frequently
Generation 1
Generation 2 (long-lived objects)
Garbage collection runs least frequently
Survivors of garbage collection are promoted to Generation 1
Survivors of garbage collection are promoted to Generation 2

Static class

In C#, one is allowed to create, by using static keyword. A static class con only contain static data members, static methods and static constructors. It is not allowed to create objects of the static class. Static classes are sealed, means you cannot inherit a static class from another class.

Syntax:

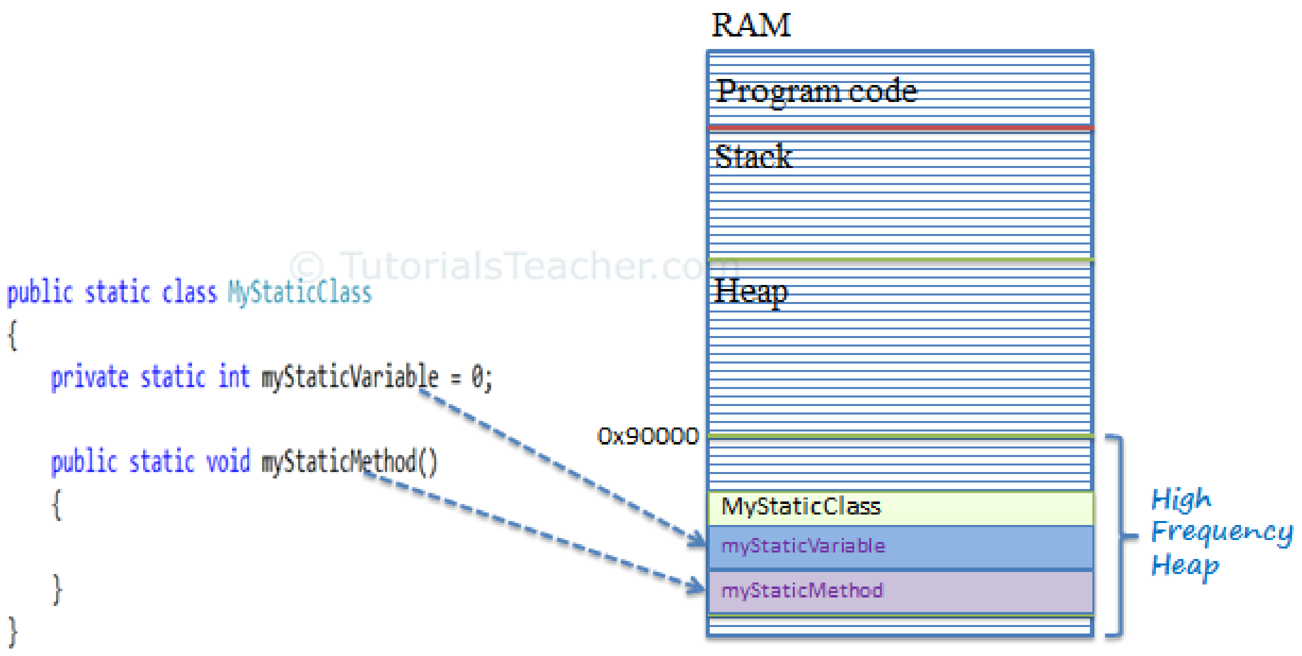
```
Static class Class Name

{

      //static data members

     // static methods

}
```



```
public static class MyStaticClass
{
    private static int myStaticVariable = 0;

    public static void myStaticMethod()
    {

    }
}
```

GAC :

- GAC is short version of “Global Assembly Cache”. It is a common place in the OS where assemblies that are going to be shared between different applications can be stored. The .NET assemblies are there, for one

GAC = Global Assembly Cache

Let’s break it down:

- Global – applies to the entire machine
- Assembly –what .NET calls its code-libraries(DLLs)
- Cache – a place to store things for faster/common access

So, the GAC must be a place to store code libraries so they’re accessible

To all applications running on the machine.

Assembly

- Assembly is .NET term for a deployment on unit.
- Files with extensions .exe/.DLL are known as assembly.
- Assemblies includes metadata.
- Assembly can be classified into two

  * private Assembly

  * Shared Assembly

Public Assembly:

- Follows 3 steps:
- Building strong name by using command is sn.exe –k keyname.snk
- Singing Assembly
- Copying into GAC by using GACUTIL.exe -i dllfilename.dll

Private Assembly:

- It is used by multiple applications.
- Drawback of private assembly is, when DLL file use in every application, each & every time copy of DLL file will be placed.
- Location of private assemblies in own folder.

Shared Assembly:

- It is also called as public assembly.
- In this, we can use DLL is reusability in several application.
- But , DLL file is placed in GAC(Global Assembly Cache).
- Shared Assembly must have a version number & unique strong name.

String pooling in .NET

- The string intern pool is a table that contains a single reference to each unique literal string declared or created programmatically in your application. The CLR uses the intern pool to minimize string storage requirements. As a result, an instance of a literal string with

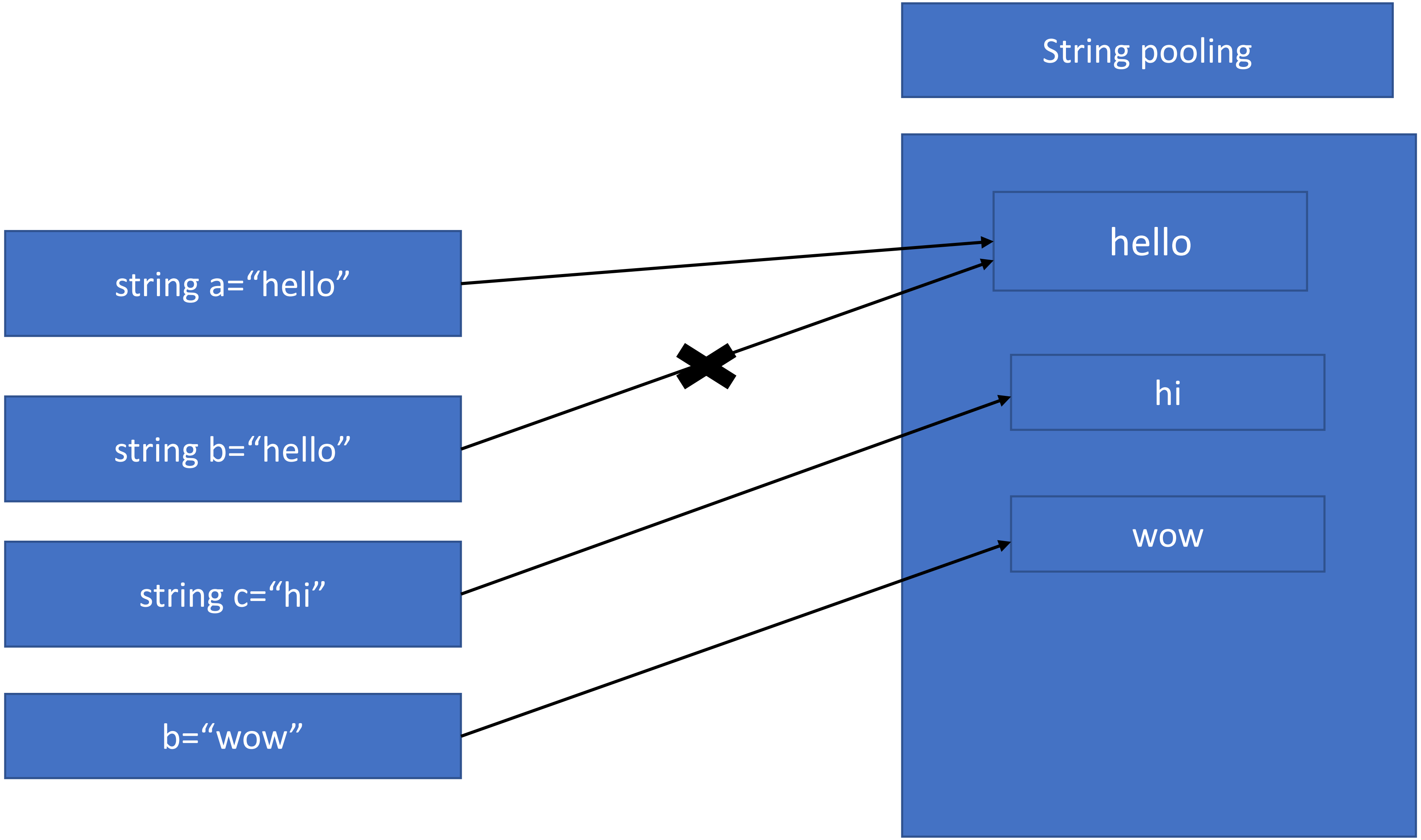
a particular value only exist once in the system.

Eg: string a=“hello”;

string b=“hello”

In the above case when the variable ‘b’ is assigned value, the CLR first checks it in the string pool for an object with same value, if such a variable exists in the pool, the reference to the same object is assigned to variable ‘b’.

String pooling in C#



String pooling

string a="hello"

string b="hello"

string c="hi"

b="wow"

hello

hi

wow

Visual studio

- Visual studio is an Advance integrated Development Environment developed by Microsoft corporation in the year 2000.
- It is actively used to develop computer programs, websites, Desktop applications, Games and much more.
- It supports many languages like C, C++,C#, F#, python etc…

Features of Visual Studio:

- Code Editor
- Debugger
- Designer
- Other Tools

Visual studio is Most power IDE, that reason behind

- Light weight
- Fast
- Open source
- Cross-platform
- Extensible

Visual Studio Code

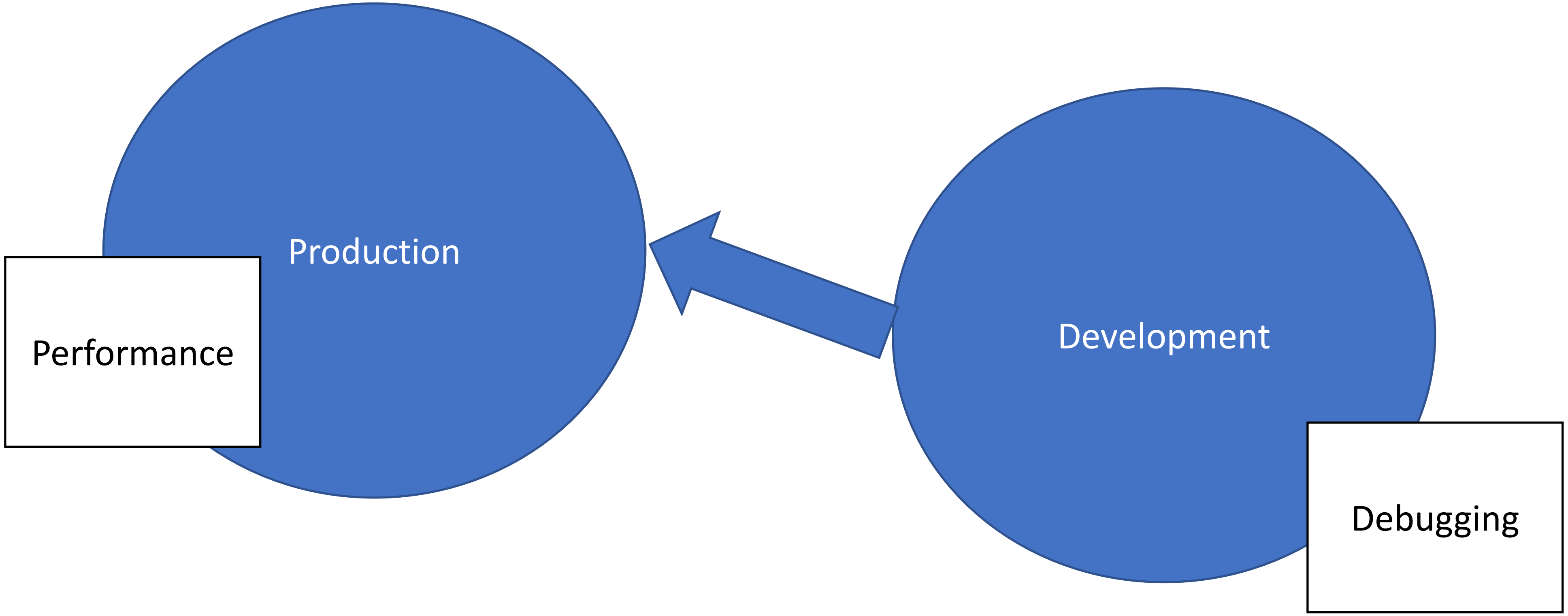
- Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for windows, macOS, Linux.
- It has a rich ecosystem of extensions for other languages(such as c, C++, C#, python etc..)
- It is combines the simplicity of a source code editor with powerful developer tooling, like debugging.
- And also it provides lot of features like syntax highlighting, bracket-matching, auto-indentation, box-selection and , more.
- By using visual studio code , easily handle with shortcut keys.

- The main features that make VS code unique is the sidebar that hosts

the core features you will be interacting with to code . Everything else you need is likely an extension you just need to install.

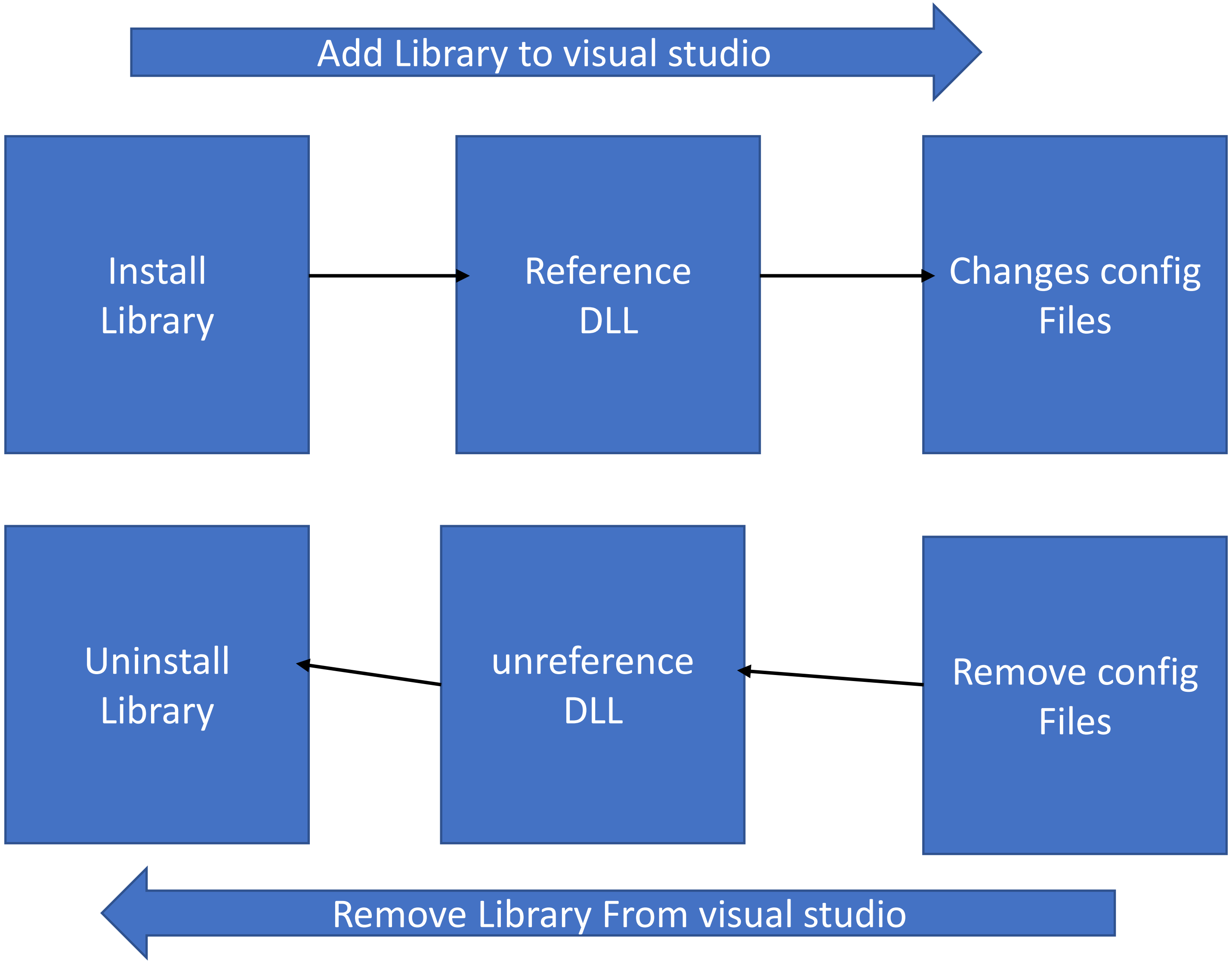
Difference between Debug and Release

Debug & Release



- In Debug mode, code is not optimized.
  In Release mode, code is optimized.
- In Debug mode, shows full stack trace.
- In release mode, directly show errors.
- In Release mode, Debug symbols are removed.

NuGet



Add Library to visual studio

Install Library

Reference DLL

Changes config Files

Uninstall Library

unreference DLL

Remove config Files

Remove Library From visual studio

NuGet:

- NuGet is a visual studio extension which helps us to search, locate the library, download them, reference them in our visual studio project

  And make appropriate changes.
- NuGet packages contain reusable code that other developers make available to you for use in your projects.
- Packages are installed into visual studio project using the NuGet package Manager or package Manager Console.
- Once installed, refer to the package in code with using <namespace> where <namespace> is specific to the package you are using.

- NuGet is the official Package manager for .NET
- Package is compiled library and descriptive metadata.

The Package Manager Console provides a PowerShell interface within Visual Studio on Windows to interact with NuGet through the specific commands listed below:

- get-help NuGet : show all command details
- Find-Package : Searches a package source using a package ID or keyword
- Get-Package : Retrieves the list of packages installed in the local repository.
- Install-Package : Install a package into the project.
- Uninstall-Package : Uninstall a package. If other packages depend on this package, the command will fail unless the –Force option is specified.

- Update-Package : Updates a package to a newer version.
- Sync-Package : Get the version of installed package from specified project and syncs the version to the rest of projects in the solution.
- Get-Project : Displays information about the default or specified project.

GitHub

- GitHub is a code hosting platform for collaboration.

GitHub essentials are:

* Repositories

* Branches

* Commits

* Pull Request

* Git

Repository:

- Storage space for your project.
- It can contains folders and any type of files(HTML, CSS, JavaScript , Documents etc..)
- It also include README file about project.

Branch:

- Branches allow you to work on other features.
- By default a repository has master branch.

Commits:

- At GitHub, changes are called commits.
- Each commit has a description explaining why a change was made.

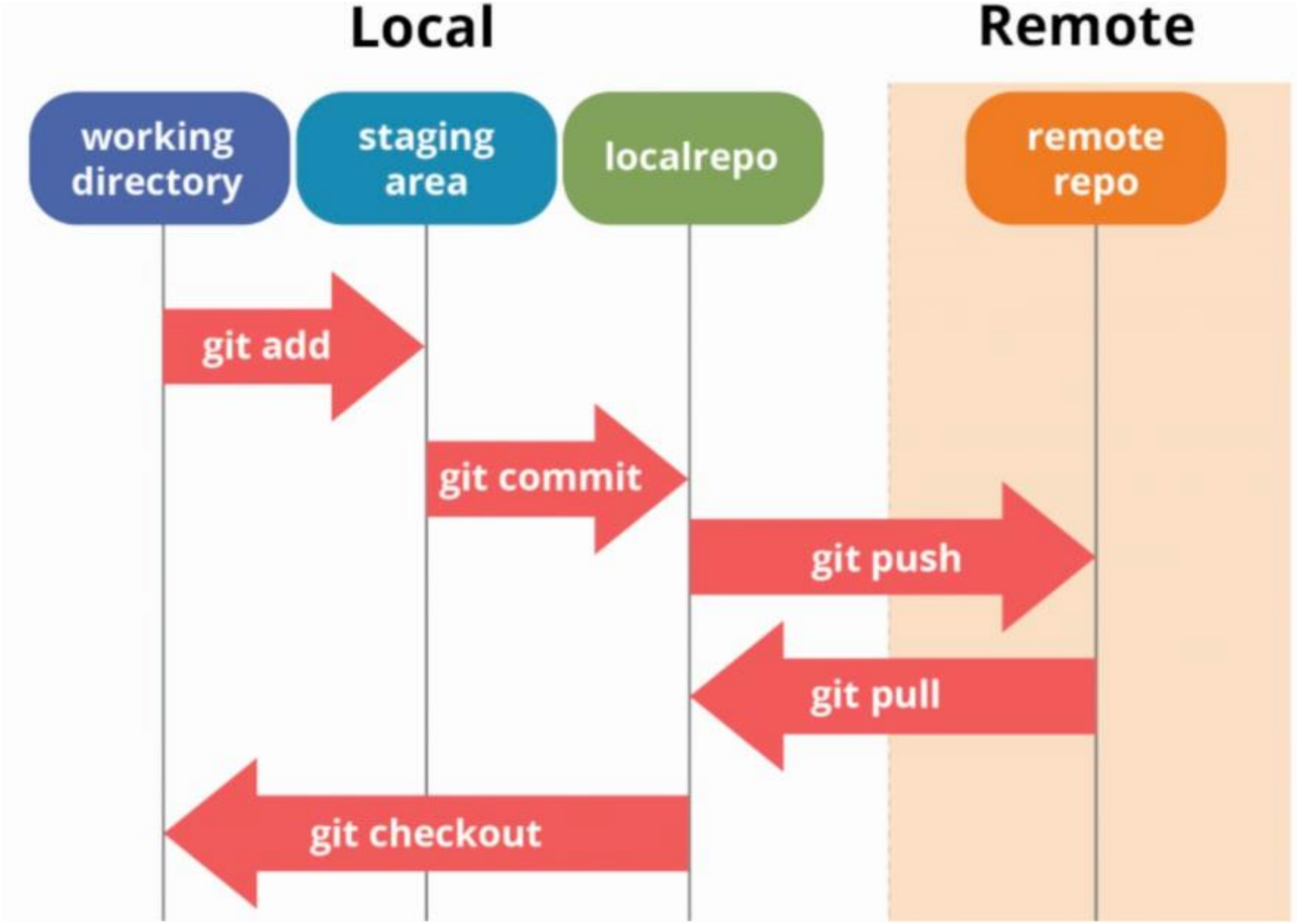
Pull Request :

- Pull requests are the heart of GitHub collaboration.
- With a pull request you are proposing that your changes should be merged with master.
- Pull request show content differences, changes, additions and subtractions.

Git:

- Git is a revision control system, a tool to manage your code history.
- Installed & maintained in your local system.
- Git push or pull data from central server.

GitHub:


Local
Remote
working directory
staging area
localrepo
remote repo
git add
git commit
git push
git pull
git checkout

Git Repository structure consists of 4 parts:

- Working directory : This is your local directory where you make the project(write code) and make changes to it.
- Staging Area : This is an area where you first need to put your project before committing. This is used for code review by other team members.
- Local Repository : This is your local repository where you commit changes to the project before pushing them to central repository on GitHub. This what is provided by distributed version control system. This corresponds to the .git folder in our directory.
- Central Repository : This is the main project on the central server, a copy of which is with every team member as local repository.

Git commands

- git config : This command sets the author name and email address respectively to be used with your commits.

Usages: git config –global user.name “name”

git config –global user. Email “email address”

- git init : This command is used to start a new repository.

Usages : git init [repository name]

- git clone : This command is used to obtain a repository from an existing URL.

Usages : git clone [URL]

- git add : This command adds a file to the staging area.

Usage : git add [file]

git add *.txt

git add .

- git add *: This command adds one or more to the staging area.
- git commit : This command records the file permanently in the version history.

Usages : git commit –m [“message”]

git commit –a –m [“message”]

git commit –A(for all file and folders move to staging area)

- git status : This command lists all the files that have to be committed.

Usages: git status

- git rm : This command deletes the file from your working directory and stages the deletion.

Usages : git rm [file]

- git branch : This command lists all the local branches in the current repository.

Usages : git branch [branch name]

git branch –d [branch name] →To delete branch

git branch → shows all branches in current repository.

- git checkout : This command is used to switch from one branch to another.

Usages: git checkout [branch name]

git checkout –b [branch name]→to make and switch to branch.

- git remote : This command is used to connect your local repository to the remote server.

Usage: git remote add [variable name] [remote server link]

- git push : This command sends the committed changes of master branch to your remote repository.

Usage : git push [variable name] master

git push [variable name] [branch]→specific branch

git push –all [variable name]→pushes all branches to your remote repository.

git push [variable name] : [branch name]→deletes a branch on your remote repository.

- git pull: This command fetches and merges changes on the remote server to your working directory.

Usages : git pull [repository Link]

git pull [variable name] [branch name]

Bitbucket

- Bitbucket is a Git-based source code repository hosting service.
- In Bitbucket, to create unlimited private repositories.
- Bitbucket is mostly used for code and code review.
- Bitbucket Cloud is a Git based code hosting and collaboration tool, built for teams.
- We provide one place for your team to collaborate on code from concept to cloud, build quality code through automated testing .

TFS

- Previously known as Team Foundation Server, Azure DevOps Server is a set of collaborative software development tools.

C# :

- C# is a programming language of .NET Framework.
- C# is pronounced as “C-Sharp”. It is an object-oriented programming language provided by Microsoft that runs on .Net Framework.

- By using C#, to develop different types of application
- Windows applications
- Web applications
- Distributed applications

Java Vs C#

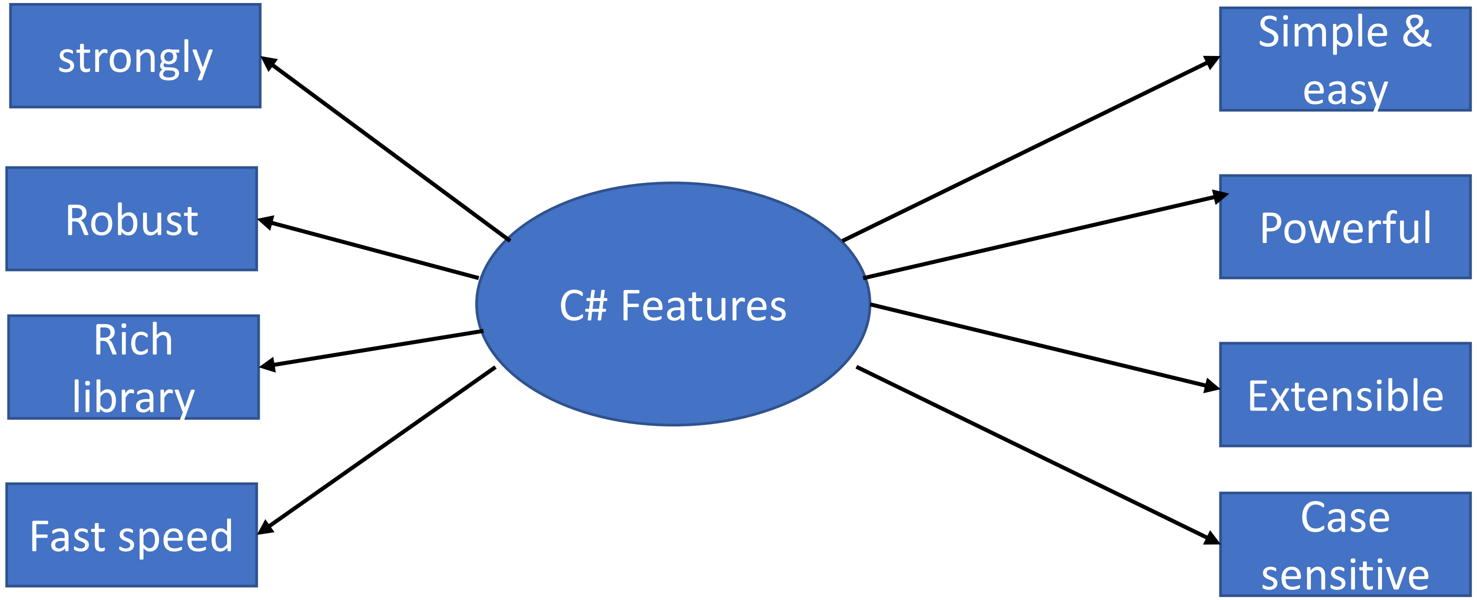
Java

- Java programming language is designed to be run on a java platform , by help of JVM.
- Java doesn't support goto statement.
- Java doesn't support structure and unions.
- In java, built-in data types that are passed by value are called primitive types.

C#

- C# programming language is designed to be run on on the CLR.
- C# supports goto statement.
- C# supports structure and unions.
- In C#, built-in data types that are passed by value are called simple types.

C# features:



C# Features

strongly

Robust

Rich library

Fast speed

Simple & easy

Powerful

Extensible

Case sensitive

- Simple and Easy: C# is a simple and easy to learn programming language. C# programs contain English like commands/instructions, hence easy to learn, code, understand, modify, debug and test.
- Powerful: C# is a powerful language. In our today life we find various types of data for number, text, list etc. C# provides rich set of data types, which help us to store variety of data and perform operations on them.
- Extensible and more flexible: C# programs are extensible to any extent from a small and simple program to a large and complex application and C# programs or applications more flexible to modify. C# programs are extended by creating methods, structures, classes, interfaces, etc..
- Case sensitive: C# treats lower case letters and upper case letters separately. If we create three identifiers num, NUM and Num then they are considered as three different identifiers.

- Strongly or strictly typed language: In C# programs while creating variables, constants or parameters, programmers must and should explicitly define their data type.
- Robust: means reliability

Application should never get crash after deploying on client machine.

That is reliability of application indicates robustness of the language.

- Rich Library: C# provides a lot of inbuilt functions that makes the development fast.
- Fast Speed: The compilation and execution time of C# language is fast.

- C# is an object-oriented programming language. In Object-Oriented Programming methodology, a program consist of various objects that interact with each other by means of actions. The actions that an object may take are called methods.
- Ex: using System

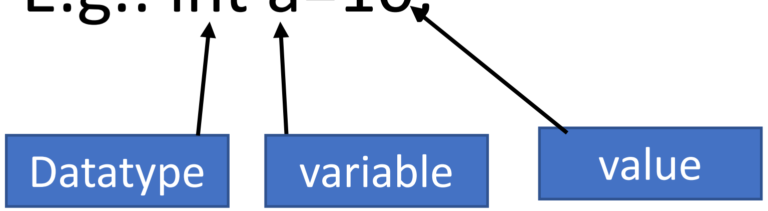
```
namespace  sample
{
        class Demo
      {
              static void Main(String[] args)
               {
                  Console. WriteLine(“hello”);
               }
      }
}
```

- The **using** keyword is used for including the namespace in the program. A program can include multiple using statements.
- **Class** is a keyword which is used to define class.
- **Program** is the a class name. A class is a blueprint or template from which objects are created. It contains data members and data methods.
- **static** is a keyword which means object is not required to access static members.
- **Void** is the return type of the method. It doesn’t return any value.
- **Main** is the method name. It is the entry point for any C# program. Whenever we run the C# program, Main() method is invoked first before any other method. It represents start up of the program.

- **String[] args** is used for command line arguments in C#. While running the C# program, we can pass values. These values are known as arguments which we can use in the program.
- **Console. WriteLine** is printing statement. Console is a class defined in System namespace. WriteLine() is a static method of Console class which is used to write the text on the console.

Variable:

- A variable is a name of memory location. It is used to store data. Its value can be changed and it can be reused many times.
- E.g.: int a=10;



Datatype

variable

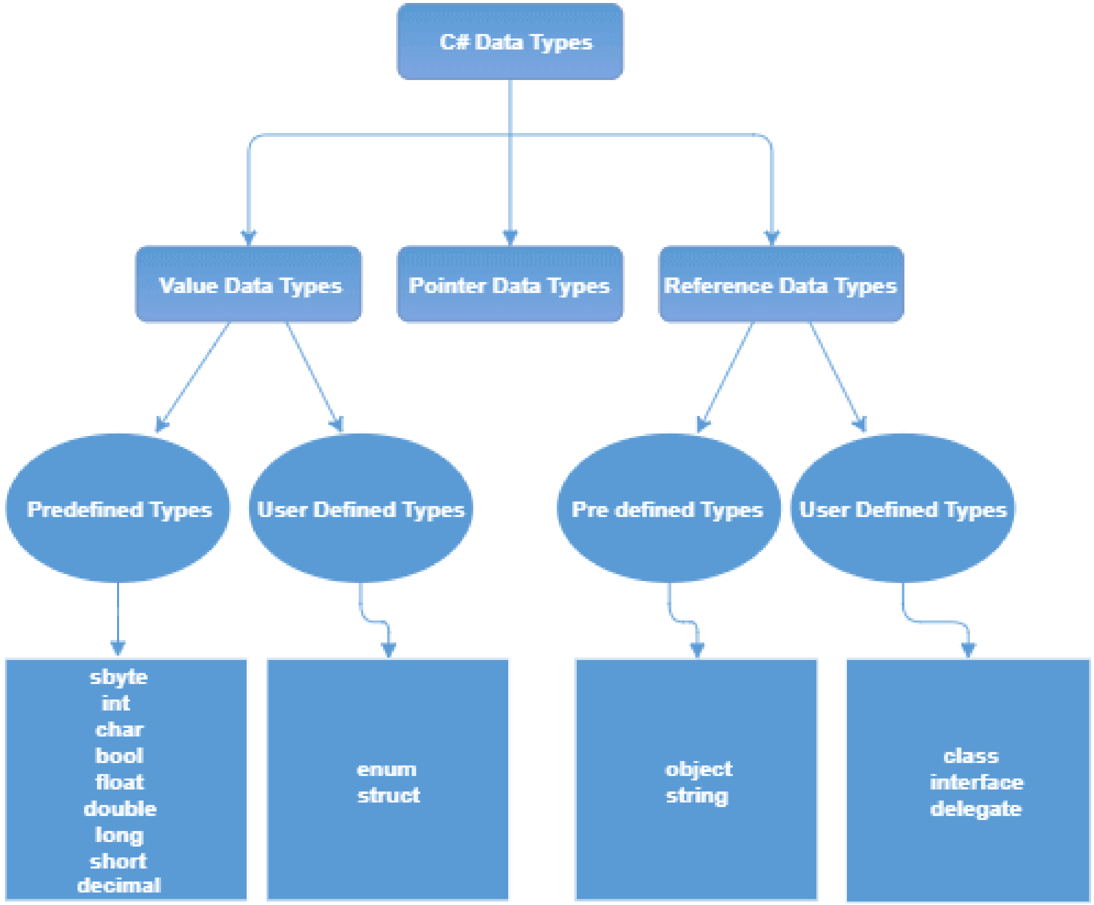
value

Rules to Declare C# variable

- You can define a variable name with a combination of alphabets, numbers and underscore.
- A variable name must always start with either alphabet or underscore but not with numbers.
- While defining the variable, no white space is allowed within the variable name.
- Don’t use any reserved keywords such as int, float , char etc.. For a variable name.
- In c#, once the variable is declared with a particular data type, it cannot be re-declared with a new type.

Datatype:

A data type specifies what type of data store in variable such as integer, float, character etc..



There are 2 types of datatypes in C#

- Value Data type : short , int, char, float, double etc
- Reference Data type : String, Class, Object and Interface

Operators

- An operator is simply a symbol that is used to perform operations. There can be many types of operations like arithmetic, logical, bitwise etc..

Types of operations:

- Arithmetic operators
- Relational Operators
- Logical Operators
- Bitwise Operators
- Assignment Operators
- Unary Operators
- Ternary Operators
- Misc. Operators

Arithmetic Operators

Operator	Description	Example
+	Adds two operands	A+B
-	Subtraction	A-b
*	Multiplication	A*B
/	Division	A/B
%	Modulo Division	A%B
++	Increment	A++
--	Decrement	A--

Relational Operators

Operator	Description	Example
==	Equal	A==B
!=	Not equal	A!=B
>	Greater than	A>B
<	Less than	A<B
>=	Greather than or equal	A>=B
<=	Less than or equal	A<=B

Logical Operators

• Operator	Description	Example
&&	Logical AND	A&&B
||	Logical OR	A||B
!	Logical Not	!(A&&B)

Bitwise Operators

Operator	Description	Example
&	Bitwise AND	A&B
|	Bitwise OR	A|B
^	Bitwise XOR	A^B
~	complementary	
<<	Left shift	A<<2
>>	Right Shift	A>>2

Assignment Operators

Operator	Description	Example
=	Equal	C=A+B
+=	Add and assign	C+=a
-=		

Miscellaneous Operators

- Sizeof() Returns the size of a data type
- Typeof() Returns the type of a class
- & Returns address of an variable
- * Pointer to a variable
- ?: Conditional Expressiong
- Is Determines whether an object is of a certain type.
- As cast without raising an exception if the cast fails.

Conditional Statement:

- C# provides following types of decision making statements.
- If
- If else statement
- Nested if statement
- Switch statement
- Nested switch statement

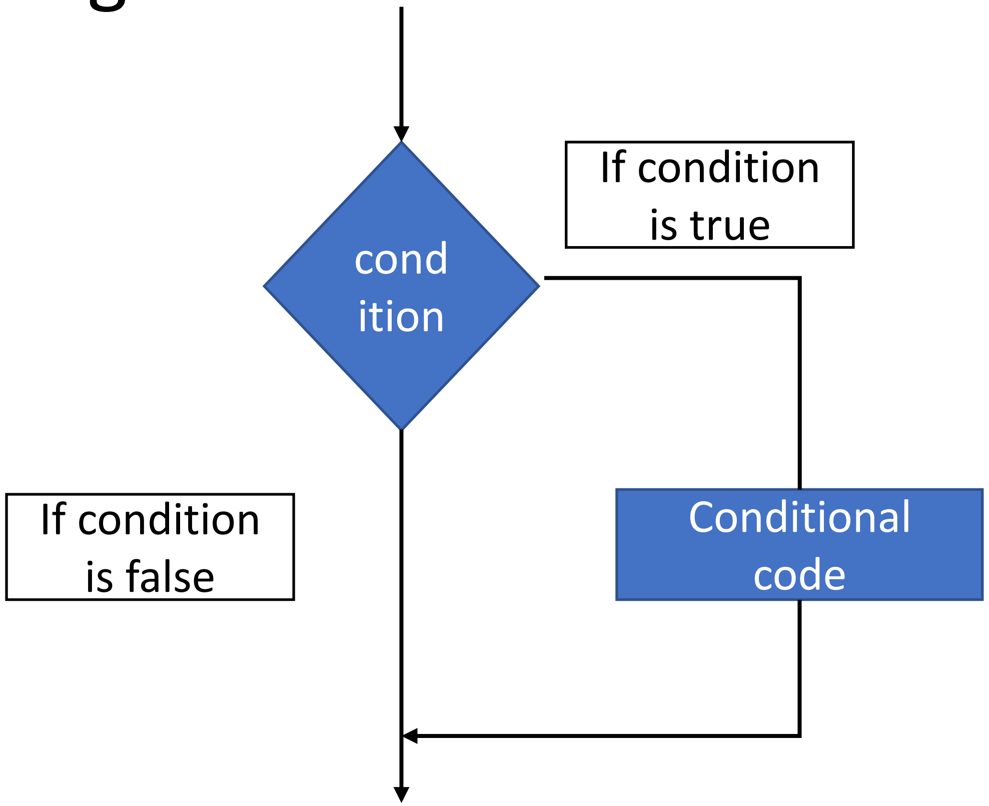
If statement:

An if statement consists of a Boolean expression followed by one or more statements.

Syntax:

```
If(bool_expression)
{
  // some code
}
```

- Flow Diagram

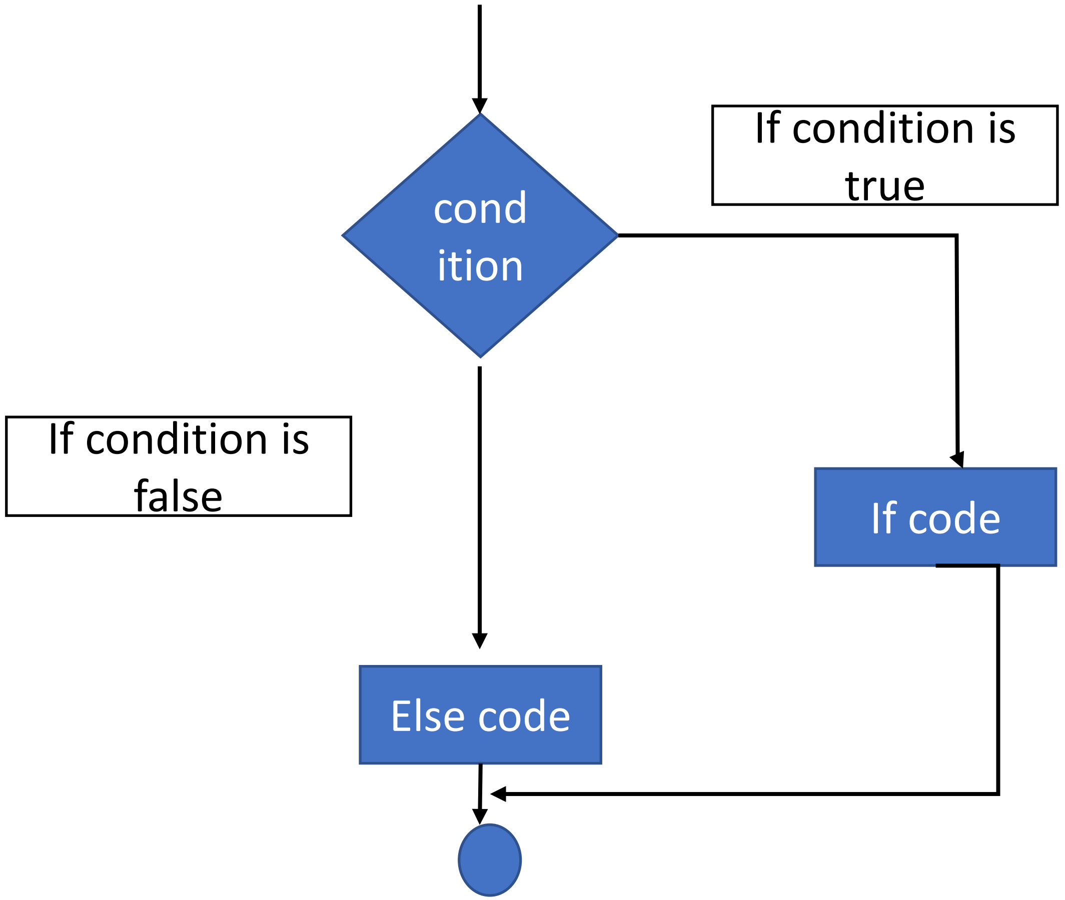


If else statement

An If statement can be followed by an optional else statement, which executes when the Boolean expression is flase.

Syntax:

```
If(Boolean_expression)
{
  //code
}
Else{
  //code
}
```



Nested if statements

- It is always in c# to nest if-else statement, which means you can one if or else if statement inside another if or else if statement.
- Syntax:

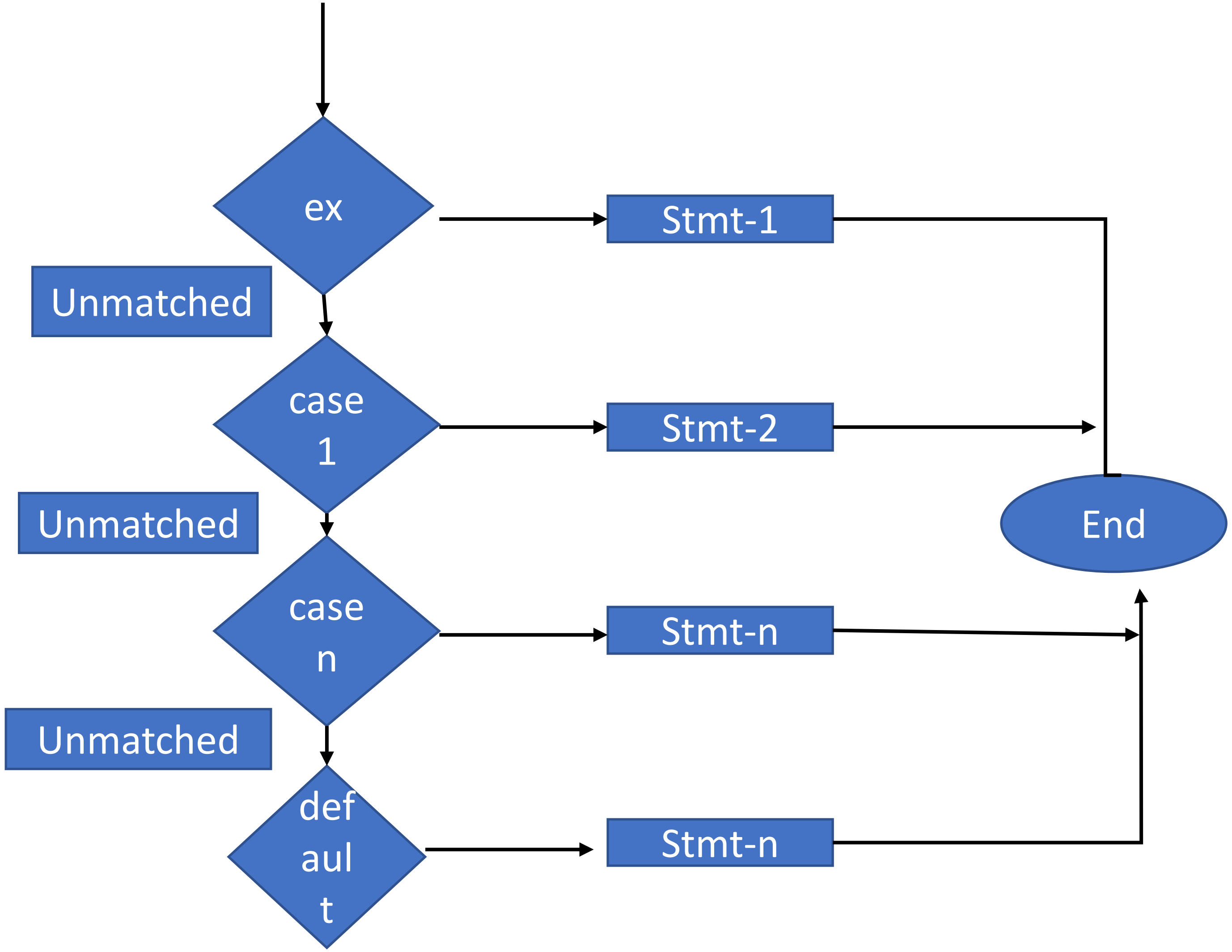
```
If(codition){
        if(condition){
        }
        else{
        }
}
Else{
}
```

Switch statement

- In C#, Switch is a selection statement, and it will execute a single case statement from the list of multiple case statements based on the pattern match with the defined expression.
- Syntax:
- Switch(variable/expressiong)

```
 {
       case 1://statements
                      break;
         :
         default://statements
                      break;
}
```

Flow Diagram:



Iteration Loops in C#

- For loop
- While loop
- Do-while loop
- Foreach loop

For loop

- In C#, for loop is useful to execute a statement or a group of statements repeatedly until the defined condition returns true.
- Syntax:

```
for(Initialization; condition; inc/dec){
        //statements
}
```

While loop

- In C#, while loop is used to execute a block of statements until the specified expression return as true.
- Syntax:

```
while(condition){
   //statements
 }
```

Do-while:

- In C#, the Do-while loop is used to execute a block of statements until

The specified expression return as ture.

Only the difference is while loop will exectute the statements only when the defined condition returns true, but the do-while will execute the statements

At least once .

Foreach

- In C#, the foreach loop is useful to loop through each item in an array or collection object to execute the block of statements repeatedly.
- Syntax:

```
foreach(Type var_name in collection_object){
        //statements to execute
}
```

Jumping Statements in C#

- In C#, Break statement is useful to break or terminate the execution of loops.
- In C#, Continue statement is used to pass control to the next iteration of loops.
- In C#, the Goto statement is used to transfer program control to the defined labelled statement and it is useful to get out of the loop or exit from deeply nested loops based on our requirements.
- In C#, Return statement is useful to terminate the execution of method in which it appear and return the control back to the calling method.

Methods

- In C#, Method is a separate code block, and that contains a series of statements to perform particular operations. Methods must be declared either in class or struct by specifying the required parameters.
- Generally, methods are useful to improve code reusability by reducing code duplication.
- Syntax:

```
 <Access_specifier> <Return Type> Method_name()
{
    //statements
}
```

Types of Methods:

In C#, we have different ways to pass parameters to methods; those are:

- Pass by value
- Pass by reference
- Out parameter
- Params keyword

Pass by reference

- Pass by reference: In c#, passing a value type parameter to a method by reference means passing a reference of the variable to the method.
- So the changes made to the parameter inside the called method will affect the original data stored in the argument variable.
- Syntax:

```
int x=10;//variable need to initialized
Multiplication(ref x)
```

Pass by out

- In C#, out keyword is used to pass arguments to the method as a reference type. The out keyword same as the ref keyword , but the only difference is out doesn’t require a variable to be initialized before we pass it as an argument to the method.
- Still, the variable must be initialized in called method before it returns a value to the calling method.

syntax:

```
int x; //No need to initialize variable
Multiplication(out x);
```

Pass by in

- In C#, “in parameter” has been introduced which allows passing read-only reference of a variable.
- Before , C# we used “ref” and “out” keywords for passing reference of a variable. “out” is meant for output only whereas “ref” is meant for input and output both.
- Syntax;
- Int x=10;

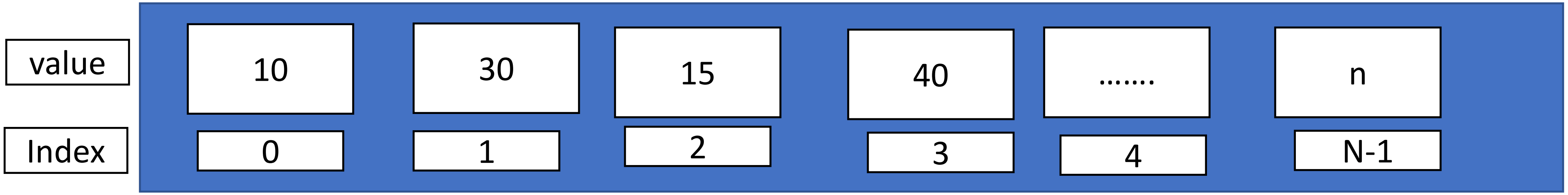
  Multiplication(x)

Params

- In C#, params keyword is useful to specify a method parameter that takes a variable number of arguments. The params keyword is useful when we are not sure about the number of arguments to send as a parameter.
- In C#, during method declaration, only one params keyword is allowed, and no additional parameters are permitted after the params keyword in a method declaration.

Arrays

- In c#, Arrays are useful for storing multiple elements of the same data type at contiguous memory location and arrays. It will store a fixed number of elements sequentially based on the predefined number of items.
- An array can start storing the values from index 0. If we have an array with n elements, it will start storing the elements from index 0 to n-1.



value	10	30	15	40	…….	n
Index	0	1	2	3	4	N-1

Declaration of array

//Declaring and Initializing an array with size of 4

- Int[] array=new int[4]

//Defining and assigning an elements at the same time

- Int[] array2=new int[5]{1,2,3,4,5};

//Initialize with 5 elements

- Int[] array3=new int[]{1,2,3,5};
- Int[] array4={1,2,3,4,5};
- Int[] array5;
- array5=new int[]{1,2,3};

Array Types

- Sigle-Dimensional Arrays
- Multi-Dimensional Arrays
- Jagged Arrays

Multi-Dimensional Array

- In c#, Multidimensional Arrays can be declared by specifying the data type of elements followed by the square brackets [] with comma(,) separator.
- Syntax:
- int[,] arr=new int[4,2]
- //three Dimensional

  int[, ,] arr1=new int[4,2,3]

Jagged Array

- In c#, Jagged Array is an array whose elements are arrays with different dimensions and sizes. Sometimes a jagged array is called an array of arrays and can store arrays instead of a particular data type value.
- Syntax:
- Int[][] a=new int[2][];
- Int[][,] a=new int[3][,];//jagged Array with two dimensional array.

List

- In c#, List is a generic type of collection, so it will allow storing only strongly typed object, i.e,. Elements of the same data type.
- The size of the list will vary dynamically based on our application requirements, like adding or removing elements from the list.
- List<T> l=new List<T>();
- Ilist<T> l=new List<T>();

Dictionary

- In c#, Dictionary is a generic type of collection, and it is used to store a collection of key/value pairs organized based on the key. The dictionary in c# will allow to store only strongly-typed objects, i.e, the key/value pairs of the specified data type.
- In C#, while storing the elements in the dictionary object, you need to make sure that the keys are unique because the dictionary object will allow us to store duplicate values, but the keys must be unique.
- The size of the dictionary object will vary dynamically so that you can add or remove elements from dictionary based on our requirements.
- Syntax:
- Dictionary<Tkey, Tvalue> d=new Dictionary<Tkey, Tvalue>();

Structure

- In c#, structures are same as classes, but the only difference is classes are the reference types and structures are the value types.
- As a value type, the structures directly contain their value, so their object or instance is stored on the stack, and structures are faster than classes.
- Syntax:
- Public struct users{

```
 //Methods etc..
}
```

- In c#, structures can be instantiated with or without new keyword.

Enum(Enumerator)

- In c#, enum is a keyword that is useful to declare an enumeration. In c#, the enumeration is a type that consists of a set of named constants as a list.
- By using an enumeration, we can group constants that are logically related to each other.
- Syntax:

```
 enum enum_name{
    //enumeration list
  }
```

Properties

- Property is a member that provides a flexible mechanism to read, write or compute the value of private field.
- In c#, properties can contain one or two code blocks called accessors, and those are called a get accessor and set accessor.
- Generally, in object-oriented programming languages like c# you need to define fields as private and then use properties to access their values in a public way with get and set accessors.
- Syntax:

```
<access_modifier><return_type><property_name>
{
        get{ return property value}
        set{set a new value}
}
```

C# Access Modifiers

- In c#, Access Modifiers are the keywords used to define an accessibility level for all types and type members.
- By specifying an access level for all types and type member, we can control whether they can be accessed in other classes or the current assembly or other assemblies based on our requirements.
- Different types of Access Modifiers:
- Public : It is used to specifies that access is not restricted.
- Private: It is used to specifies that access is limited to the containing type.

- Protected : It is used to specifies that access is limited to the current class or types derived from the containing class.
- Internal : It is used to specifiers that access is limited to the current assembly.
- Protected internal : It specifies that access is limited to the current assembly or types derived from the containing class.
- Private protected: It is used to specifies that access is limited to the containing class or types derived from the containing class within the current assembly.

OOPs in c#

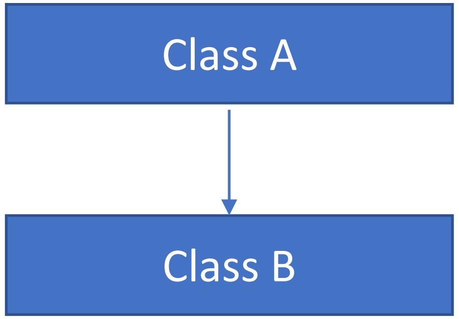
- Encapsulation : It is a process of binding the data members and member functions into a single unit. In c# , the class is the real-time example for encapsulation because it will combine various types of data members and member functions into a single unit.
- Abstraction : It is a principle of object-oriented programming language and it is used to hide the implementation details and display only essential features of the object.
- Inheritance : It is one of the primary concepts of object- oriented programming and it is used to inherit the properties from one class to another class.

Types of Inheritance

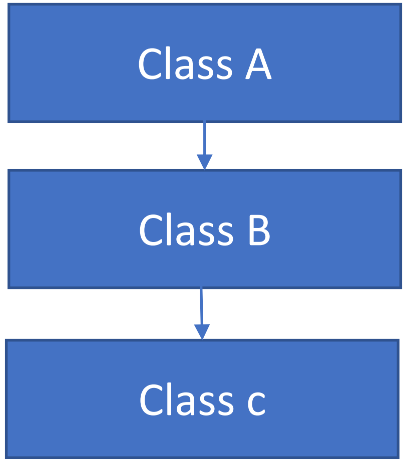
- Single
- Multi-level
- Hierarchical
- Hybrid
- Multiple
- Multipath

Single Inheritance

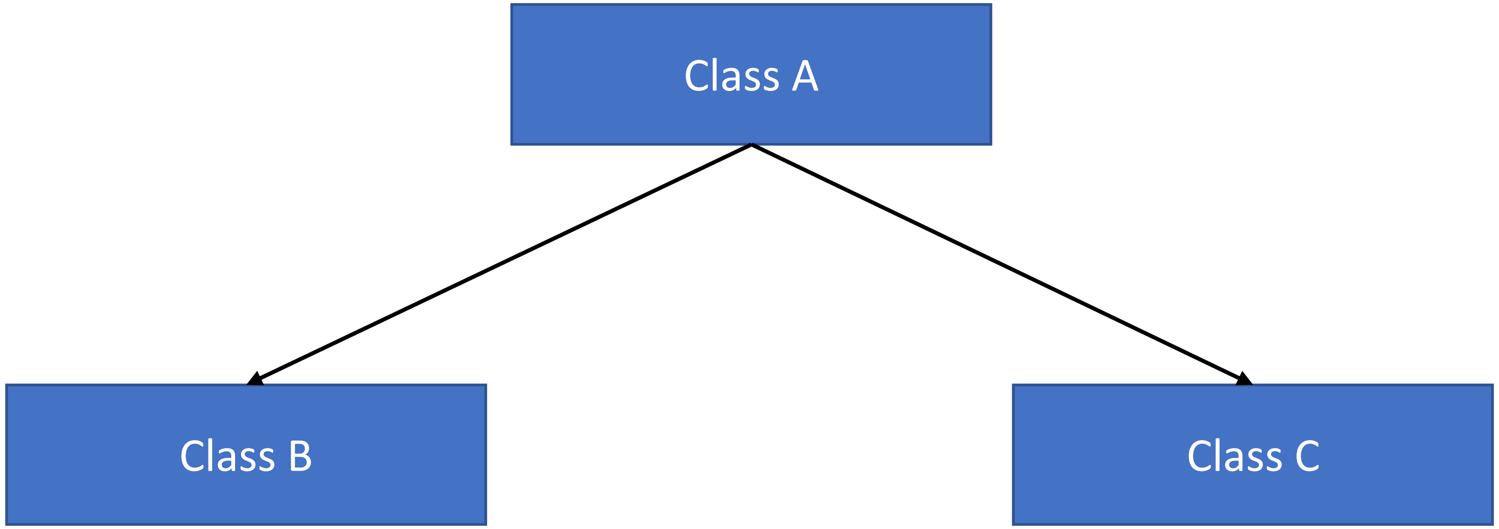
- In this inheritance, a derived class is created from a single base class.



- Multi-level : A derived class is created from another derived class.



- Multiple : If class has more than 1 immediate parent class to it we call it as multiple inheritance.



Class A

Class B

Class C

- Hierarchical : More than one derived classes are created from a single base class.
- Hybrid : It is combination of more than one inheritance. It may be a combination of multilevel and Multiple or Hierarchical.
- Advantages of Inheritance:
- Reduce code redundancy.
- Provides code reusability.
- Reduces source code size and improves code readability.

- Method overloading : It means defining multiple methods with the same name but different parameters. Using method overloading, we can perform different tasks with the same method name by passing different parameters.
- Method overriding: It means override a base class method in derived class by creating a method with the same name and signatures to perform a different task. The Method overriding in c# can be achieved using override and virtual keywords and inheritance principle.

Polymorphism

- Polymorphism means to ability to create more than one form.
- Generally, poly means multiple & morphism means forms.
- Two types of polymorphism.
- Compile Time polymorphism : It can be achieved by using method overloading and it is also called early binding.
- Run Time polymorphism : It can be achieved by using method overriding and it is also called as late binding.

Class and object

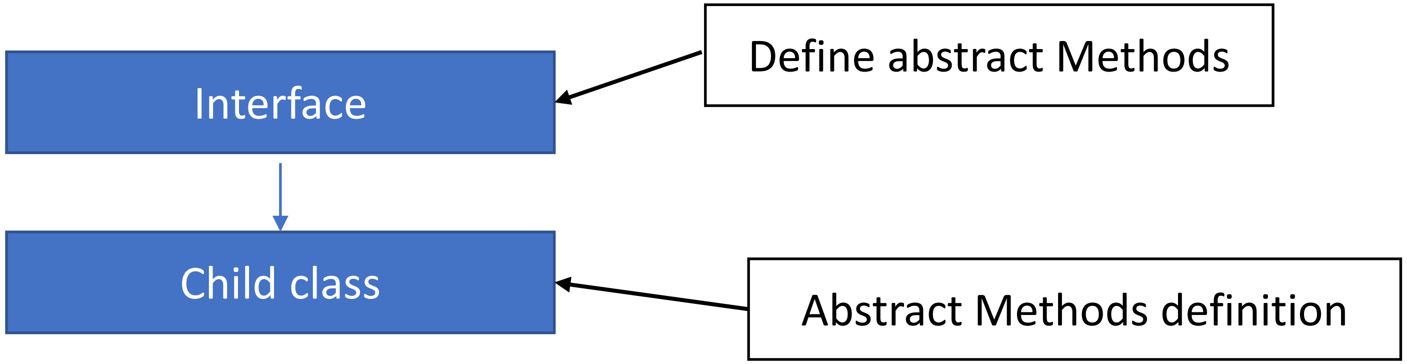
- Class is user defined type. It is nothing but a collection of various data members.
- Object : Object is instance of class.

- Abstract Class : It contains Non-Abstract methods(with definition) and Abstract Methods(without definition).
- If abstract method is exist in any class, then that class is called as abstract class.

- Abstract Method : A method without any method definition is known as an abstract method.

Interface

- Interface is also user-defined type.
- It contains only Abstract methods (without definition).
- Every abstract method of an interface should be implemented by the child class of the interface without fail.



Interface

Child class

Define abstract Methods

Abstract Methods definition

-
  ```
  [<modifier>] interface <Name>
  {
      //Abstract Member declaration here….
  }
  ```

Constructor

- Constructor is special method that will invoke automatically whenever an instance of class or struct is created.
- Constructor will have the same name as the class or struct and it useful to initialize and set default values for the data members of the new object.
- Syntax:

```
•  public class User{
          //constructor
          public User(){
          }
}
```

Types of constructors

- Default constructor
- Parameterized constructor
- Copy constructor
- Static constructor
- Private constructor…

Namespaces

- Namespaces is container. It contains class and methods.
- It is used to organize the multiple classes in our applications.
- The global namespace is the root namespace: system.

Delegates

- It is a type safe function pointer.
- A delegate holds the reference of a method and then calls the method for execution.
- Define a delegate

[<modifier>] delegate void/type <Name>([<parameterList>])

Ex: public delegate string sayDelegate(string name);

Two types of Delegates:

1. Single Cast Delegate
2. Multicast Delegate.

- Single cast Delegate : It is points to a single method then it is called as single cast Delegate. It is used to hold the reference of a single method.
- Multicast Delegate : It is points to multiple methods then it is called as multicast delegate. It is used to hold the reference of multiple methods..

C# String Format

- In C#, the string Format method is used to insert the variable’s value or an object or expression into another string. Using the string Format method, we can replace the format items in the specified string with the string representation of specified objects.
- Syntax:
- Public string Format(string, object)
- Public string Format(string, object, object)
- Public string Format(IFormatProvider, string, object)

- Ex: string s=“age is {0}\n” + “Name is {1}\n”;
- Console. WritLine(string.Format(s,22,”veerababu”));
- Output: 22

  veerababu

.NET Core

- .NET Core is the open-source, free development platform maintained by Microsoft. It is a cross-platform framework that runs on windows, macOS & Linux operating system.
- .NET Core Framework can be used to build different types of applications such as mobile, desktop, web, game etc.

.NET Core Characteristics

- Open-source Framework: .NET Core is an open-source framework maintained by Microsoft.
- Cross-platform: .NET Core runs on windows, macOS and Linux operating system. There are different runtime for each operating system that executes the code and generates the same output.
- Wide-range of applications: Various types of applications can be developed and run on .NET Core platform such as mobile, web, game etc..
- Supports Multiple languages: You can use C#, F#, and visual Basic programming languages to develop .NET Core applications.

.NET Core

- .NET Core is an open source.
- It is compatible with various operating system like windows, Linux and Mac OS. It is cross-platform.
- .NET Core is packaged and installed independently of the underlying operating system as it is cross-platform.
- .NET Core is shipped as a collection of NuGet packages.

.NET Framework

- Certain components of the .NET framework are open source.
- It is compatible with only windows.
- .NET Framework is installed as a single package for windows operating system.
- All the libraries of .NET Framework are packaged and shipped together.