**Session 9: Var Dynamic Keyword And Properties**

**Var:** introduced in C# 3.0. Its statically type. Var keyword is used to define and declare. Initialisation is done at the time of declaration then compiler can decide the type of variable at compile time. If the variable not initialised it throw error. Cannot user for **properties.** Only used as local variable. getType method is used to check the type of datatype. Intelisence support. Its **value type.**

EG: public var Method(var a) Error.

**Dynamic:** introduced in C# 4. Its dynamic in nature. Dynamic keyword is used to define variable. Run time variable type decide. Type of variable need to define at initialisation level. If variable not initialised in wont throw error. Used for **properties** to return value. Intelisence not support. getType method is used to check the type of datatype. Its **Refrence Type.**

EG: public dynamic Method(dynamic a)

**Properties:** its nothing but natural extension of data fields. it’s a member of class provide flexible mechanism to expose private field. It also used for **Encapsulation.** Its special method called accessor. Its having two values **Get:** Return the property value. and **Set:** assign new value. Properties can be **read & write:** it implement bot get and set. **writeOnly:** it implement set accessor. **Read-Only:-** implement get accessor. Value keyword is used to take the value and print.

**Static Properties: -**  used to access static fields of class safe manner. Static keyword is used. Static property used using classname.cannot initialise instance field withing static property.

S9\_\_VarAndDynamicAndProperties.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace Basic\_\_AllSession

{

class S9\_\_VarAndDynamicAndProperties

{

public void VarAndDynamic()

{

Console.WriteLine("\*\*\*\*\*\*\*\*\*You are in Var \*\*\*\*\*\* \n");

var data = "Program One";

//data = 100; //Error value cannot change

foreach (char program in data)

Console.Write(program);

Console.WriteLine("\nDatatype of Var is: "+ data.GetType()+"\n");

Console.WriteLine("\*\*\*\*\*\*\*You Are in Dynamic \*\*\*\*\*\*\*\*\*\n");

dynamic name = "This is The Program of Imran";

//name = 100;

Console.Write("Value is : "+name + "\n");

Console.Write("Message is :");

foreach(dynamic message in name)

Console.Write(message);

Console.WriteLine("\nDatatype of Dynamic is: " + name.GetType() + "\n");

}

//Properties

public void PropertiesMethod()

{

Properties properties = new Properties();

Console.Write("Enter Student Id: ");

properties.studentId = int.Parse(Console.ReadLine());

Console.Write("Enter Student Name: ");

properties.studentName = Console.ReadLine();

Console.Write("Enter Student Phone: ");

properties.studentPhone = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter Student City: ");

properties.student\_City = Console.ReadLine();

Console.Write("Enter Student Add: ");

properties.student\_Add = Convert.ToInt32(Console.ReadLine());

//Console.Write("Enter StudentMark : "); //Read Only Properties;

//properties.studentMarks = Convert.ToInt32(Console.ReadLine());

Console.WriteLine($"Student ID:{properties.studentId} \nStudentAdd: {properties.student\_Add} \nStudent Name: {properties.studentName} \nStudent Phone: {properties.studentPhone} \nStudent City: {properties.student\_City}");

}

//Static Property

public void StaticProperty()

{

Console.Write("Enter School Name : ");

Properties.school\_Name = Console.ReadLine();

Console.Write("Schol Name is: "+Properties.school\_Name);

}

}

class Properties

{

public int studentId { get; set; }

public string studentName { get; set; }

public string student\_City { get; set; }

public dynamic studentPhone { get; set; }

//public var studentGender { get; set; }

public int studentMarks {get;} // Read Only

private int studentAdd;

public int student\_Add

{

set

{

if (value <= 0)

Console.WriteLine("Not allowed");

else

this.studentAdd = value;

}

get { return this.studentAdd; }

}

//Static Property

static string schoolName;

static string schoolAdd;

public static string school\_Name

{

get

{return schoolName;}

set

{

if(string.IsNullOrEmpty(value))

Console.WriteLine("Empty String Not Allowed");

else

schoolName = value;

}

}

public static string school\_Add { get; set; }

}

}

Program.cs

using Basic\_\_AllSession;

using Basic\_\_AllSession.nestedNamespace; // Assembly Refrence OR Namespace OR PAckage

using System;

namespace Basic\_\_AllSession

{

class Program : S4\_\_AccessModifiereAndKeyword

{

static void Main(string[] args)

{

//Session 9 Var and Dynamic

S9\_\_VarAndDynamicAndProperties varDynamic = new S9\_\_VarAndDynamicAndProperties();

varDynamic.VarAndDynamic();

varDynamic.PropertiesMethod();

varDynamic.StaticProperty();

}

}

}