MS. Net :

25th July
day-1.

MS. Net	Java
• C# Programming source code ⇒ .cs file	• object oriented Programming source code ⇒ .java file
• Collections	• Collection Framework
• ado.net	• jdbc
• dynamic link library (.dll file) we need to include it for drivers	• .class files / jar files. we import jar files for drivers.
• IL code (intermediate lang)	• Byte code.
• Runtime ⇒ CLR common language runtime	• Runtime ⇒ JVM Java virtual machine.
• Memory management through GC (garbage collector)	• Memory management through GC (garbage collector)
• Entity framework	• hibernate (ORM)
• asp.net mvc	• JSP / servlet spring mvc
• web api	• spring boot api (rest)
• Microservices	• Microservices.

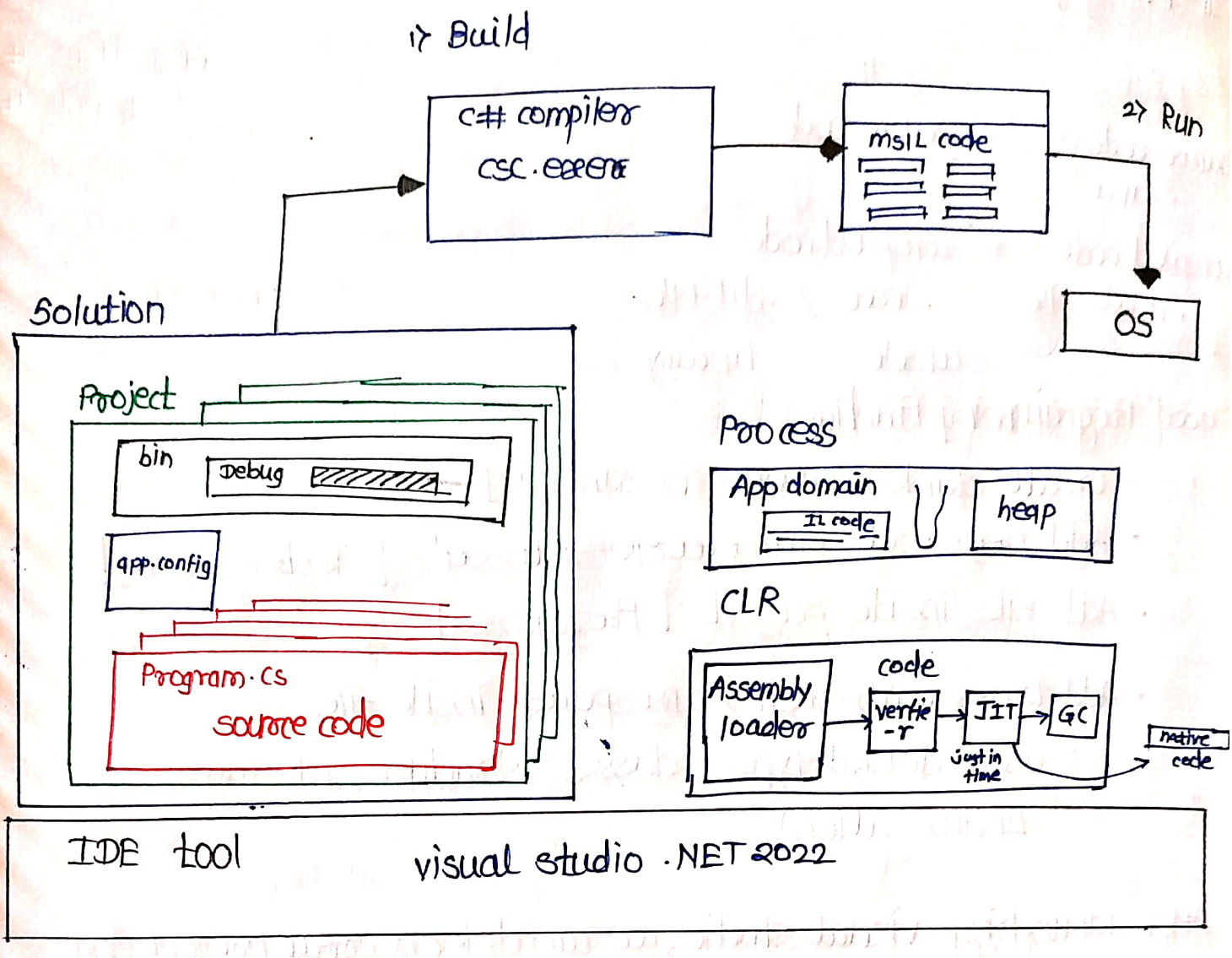
C# Programming

C# was introduced by Anders Hejlsberg
Core developer of typescript (angular)

C# has power of ⇒ C, C++,
simplicity of ⇒ java
elegance of ⇒ vb (visual basic)

learning curve is less.

C# Program	java program
namespace HelloWorldApp	Just like the package in Java, C# programming have namespace which is to be imported by using keyword using
internal class Program	
// These are the access specifiers in C# programming public, internal (like packag/ default scope)	Public, private, protected are the access specifiers in java
{	
static void Main (string[]args) ↑upc ↑smc	there is main f^n is java, in C# f^n name start with upper case letter
{	
In namespace System →console.WriteLine ("Hello");	class String is in java string is datatype in C#
// to print a string	⇒ just like sysout
console.ReadLine ();	
// to hold the scree to read	# Java does not support operator overloading C# does.
System is inbuid namespace having console class in it.	
• folder structure solution project prop App.config reff source code (.cs file)	• folder structure Project src package class



- We can add as many functions we want in the same class. These functions are helper functions can be called in main function.
- It also support method overloading like in java.
  Add function is overloaded in Program.cs.
- Math App ⇒ Program.cs ← main
  Person.cs ← person class
  getters setter
  constructor overloading
  to string overloaded.

Must see HelloWorldApp for all constructor, property & getter setter trial.

day-2
26th July.

Java	C#
Source code:	source code
.java	.cs
compiled code	compiled code
.class file	.exe / .dll file.
	console Library.

Good Programming Practice:

- Create blank solution [.sln file]
- Add new .net Framework into solⁿ [Tester.csproj]
- Add files inside project [Program.cs]
- Add types with their namespaces inside file
  (.user-defined types, classes, structure, interfaces, Enumeration)

After launching visual studio, we can click on creat project and search blank solution & creat it.
After that, to add project in it ⇒ right click ⇒ Add ⇒ new project

In the reference folder, we can see many .dll files, these are ntg bt the dynamic linked libraries (.jar files in java)
person.dll, Banking.dll are the libraries that have referrances in our program.cs.

Person class is made in Person.dll, Account in Banking.dll & its objects are created in program.cs.

We can also add reff of another library to a library
person.dll is added to project management.dll

day-3
27th July

Shopping Application: * Catalog [.dll] : Product, Product Controller, Product Manager } classes.
Library

* shopping console - Console App (.cs)
Main method.

Product.cs : Product class is defined with all its attributes & getter setter blocks.

ProductManager.cs : All CRUD operations are written it.
Initialize, Insert, Update, GetAll, GetByID, Delete

ProductController.cs : It is a controller which handles all requests & generates response. It is Action method handle

SalaryApp :- Program.cs

object of Sales Manager is created.

SalesManager derived from SalesEmployee
SalesEmployee —┬— Employee
Employee —┬— User
User —┬— Person

∴ Person is a base class for all the derived classes.

This is nothing but the inheritance in C# (IS-A relation)

In Utility, Address class is defined & it is used in Customer
HAS-A relation.

Customer class is selaed by using keyword sealed.

~~selat~~ sealed classes are used to restrict the user from inheriting the class. A class can be sealed by using sealed keyword. The keyword tells the compiler that the class is sealed & therefore cannot be extended. No class can be derived from a sealed class.

day-4
28th July.

Virtual & override keyword.

In C#, for overriding the base class method in derived class, you have to declare a base class method as virtual & derived class method as override.

Implementation demo App.

Shape is an Abstract class, (cannot be instantiated) having methods with ~~virtual~~ virtual keyword, so these will be override in derived (Line) class.

Shape is a base class ⟹ Line is derived class
Inheritance ⇒ Is-A relation.

Point is class ⟹ Line has-A reff of point in it.
Inheritance ⇒ HAS-A relation.

IOrderDetails & ICustomerDetails are interfaces having some methods.

All the methods in interfaces are by default public.

If a class impliments two interfaces having same method, the method implented using qualified name & it behaves as private.

Keywords in C#

i) Const : Const is used to create a constant at compile time.
Const field value cannot be changed after declaration.

ii) Readonly : Readonly is the constant datatype defined at runtime
readonly Field value can be changed after declaration.

iii) Out : out is for outgoing parameter

Boxing & unboxing → Just like Java.

iv) Params : specify a method parameter that takes a variable no. of arguments. (single dimentional array)

v) ref : ref keyword passes the ~~value of~~ referance of a variable to a function.

for each syntax in C#

```
foreach ( string name in trainers)
    {
        Console.WriteLine (name);
    }
```

single data

list, array etc.

e.g.

```
foreach ( Student s in students)
    {
        Console.WriteLines( s.Id+" "+ s.Name.+" "+s.location);
    }
```

day-5
29th July.

Serialization & Deserialization

Two namespaces are required ⇒ System.IO
System.Runtime.serialization.
Formatters.Binary

SerializationDemoApp ⇒ Persistanceday5.

1. Save method :- Serialize the data into the specified file.

filestream object ⇒ taking parameters ⇒
path ⇒ text file path on machine

FileMode.OpenOrCreate ⇒ if file is present then open or else create.

FileAccess.ReadWrite ⇒ Read & write access to file.

Binary formatter object ⇒ new object

↳ obj.serialize (filestream obj , classobj)
↑products

filestream.close.

2. Load method. ⇒ Loads the data from file.
deserilization of data from file specified.

serialization ⇒ object ⇒ to JSON / text in file

deserialization ⇒ JSON/text in file ⇒ to object in code.

JSON App.

To serialize the data in JSON file, we need to install namespaces JSON

project ⇒ right click ⇒ Manage Nuget Packages ⇒ search system.JSON ⇒ install.

In References, system.Text.JSON in added ⇒ Packages are installed

DotNetTypeApp:

Delegate : A delegate is reference type that is basically encapsulates the reference of methods.

- Type safe
- object oriented
- secure

Two types → single delegate
→ multicast delegate ⇒ chaining

steps to create & use delegate

i) Declare a delegate type

e.g. Public delegate void outputAction (string data)

ii) create method having same signature

print, display, log methods in controller

iii) create obj of delegate & pass method reff

OutputAction action1 = new OutputAction (print);

iv) Invoke delegate obj by passing data

action1. Invoke ("India");

day-6
30th July

day-6 Application (MVC) Application.

Project creation flow :-

create new project ⇒ · ASP.NET Web Application (.NET Framework)
→ Give Name → create
→ Empty → untick configure HTTPS
tick mvc

create
[Project gets created]

Project execution flow :-

step-1 ⇒ RouteConfig.CS

Here, a default route is mapped giving url.

url : " {controller} // {action} //{id} "

default : new { controller = " Home", action="Index", id=UrlParameter.Optional.

From RouteConfig.cs ⇒ the controll shifts to controller Home & its index Action is called.

controller ⇒ Add ⇒ controller → MVC 5 controller - Empty = Add
[Give name Home]

step-2 ⇒ HomeController.CS

Homecontroller is derived class of controller having a method Index which returns a view

click on
Index ⇒ Add view ⇒ MVC 5 view ⇒ Add ⇒ scaffolding process

This will generate Index.cshtml file, we can write an html code we want for our web Application here to display on browser.

day - 7.
1st August

EcommerceDay7 ⇒ onlineshoppingWeb ⇒

Folderstructure

- Connected services
- Properties
- References ← All dll references, dependencies etc.
- App_Data
- App_Start
  - RouteConfig.cs ← starting point of execution
- Content
  - Bootstrap.min.css
  - site.css
- Controllers
  - HomeController.cs
  - ProductController.cs
  - AccountController.cs

  All the controllers required for project will reside here.
- DAL (Data Access Layer)
  - DataManager.cs

  All the data manager will reside here, All CRUD operations are from here.
- images
  - .png
  - .png
  - .png

  All the images required for application will be stored here only.
- Models
  - Product.cs

  All the classes defined in project stored here
- scripts
  - jquery.min.js

  script reff is stored here
- views

  All the views required for each controller gets added here only.

In shared folder, _Layout.cshtml is stored having reff of all the necessary adaptions. (like pom.xml in java)