.NET

.Net

Cts and CLS

- are responsible for type safety

CLS → common language specificatn.
CLS → is subset of CTS
↳ defines rules that every language need to follow to run in .dnet

CLR → Common Language Runtime
↳ it is virtual machine Component
↳ responsible for execution of code

CTS → Common type System
↳ defines data type used by managed code

Types supported by CTS
① value types ② reference types

---

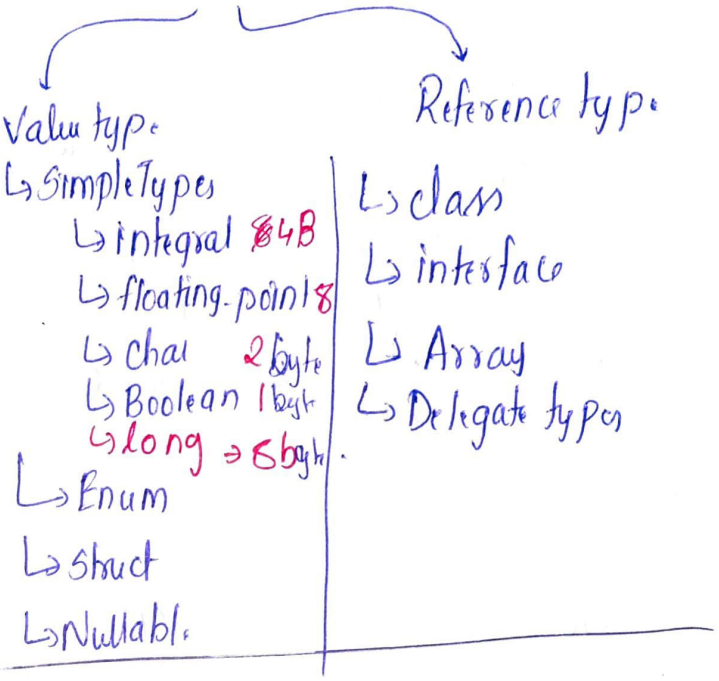
② Configuration files used by .net framework.
Machin. config → used system wide
Web. config → applied to each applicatn

Structure type

① It is value type
② .net uses structure to represent number (int & real), Boolean, unicode, time instanc
③ Struct are instatiated using new operator

Datatypes

① C# is typed language → declaring type of variable is must



---

② byte → 1
Sbyte → 8bit
Short → 16bit
Int → 32bit
long → 64bit
float → 32bit
double → 64 bit
decimal → 128bit

alias	.NET types	
byte	Sys. Byte	struct type
Sbyte	Sys. SByte	
int	Sys. Int32	
short	Sys. Int16	
long	Sys. Int64	
float	Sys. Single ①	
double	Sys. Double ①	
decimal	. Decimal	
DateTime	S . DateTime	

Default values

int → 0
char → '\0'
decimal = 0
bool = false
char =

Const keyword cannot be assigned with expression

Const int k = a * x * y ✗

Const int k = 10 ✓

8bit 16 32 64 32
byte < short < int < long < float
sbyte ushort uint ulong
< double < decimal < 2^128

char → 16 bit

Convert.ToInt32 — passing null → gives 0

int32.parse — passing null throws ArgumentNullException

Default type without decimal is int

- literal with suffix d, D → double
- literal with f or F → float
- literal with m or M → decimal

Precision →

float → 6 ~ 9 digit
double → 15 - 17 digit
decimal → 28 - 29 digits

Looping

- float and double are used to represent floating Binary ($e^x 10^y$)
- Decimal is used for floating Decimal

mantissa → fractional part

NAN → is used when result are undefined

Dictionary in C#

→ generic collection that store key and value.
→ methods → add
Remove

Char Vs Varchar

↳ fixed length
↳ non-unicode characters
↳ Varchar
↳ variable length datatype
↳ non-unicode characters are stored

Compare → is used to compare two string.

CompareTo → compare with instance

String in C#

System.String, String, string as keyword to use

String Comparision

String.Equals, Compare, CompareTo
String is always created on heap

Var & dynamic in C#

Var → to specify type is not compulsory
⊗ type is automatically deduced
C# 3.0

⊗ C# 4.0 → Dynamic type is designed → used to avoid type checking at compile time.
→ run time type deduced

⊗ :: → is used to set class externally

⊗
```
main(){
  Apple t1.
  t1.print
}
class Apple {
    void print(){
        syso("Hello")
    }
```

Ans → error as t is uninitialised

⊗ wrong about ref
* ref cannot be called recursively

⊗ If(Conver.ToBoolean(x=0))

here x=0 so it will get converted to false for x=1 it get converted to true

---

.Net framework

⊗ True regarding clr.
① it collects garbage
② it handles managed code
③ it provides language neutral environment
④ it ensures unathorized memory is not accessed

---

② Valid .Net clr performance counter
① no of method failed to compile in standard jit
② percentage of processor time spent on jit compilation

True regarding managed code
Code written to target services of 'CLR'

⊗ ngen can be used to convert managed code assembly to processor native code

---

.Net framework facts
① it provides object-oriented programming environment, whether object code is executed locally or remotely or semi
② it provides code execution environment that minimises deployment & version conflict
③ it promotes safe execution of code, whether code is executed is by third party

---

⊗ .Net class library is a component that provides extensible classes which may be used by .net compliant programming language

---

⊗ .Net uses Dcom for transition b/w managed and unmanged Code

---

⊗ .dot net provides code access security and Role based security

---

regarding .Net assembly →
① it is smallest ~~execution~~ unit deployable
② each assembly has one entry point main, DLLmain, winmain
③ it can be shared assembly or Private

JIT Compiler

- it compiles instruction in machine code during runtime
- it runs und CLR
- it converts code to cpu native code
- .Net frame work :- conten only class library and CLR

frameworks

Multithreading

- threads created used thread class are termed as child thread of main thread.
- process-based mullitasking :- two program run concurrently, a program that act as small unit of code which can be dispatched
- System.threading
- synchronised → is used to achieved synchronisatn
- When any process is blocked temporarily from running it calls wait() which puts executing thread to sleep & relase resource
- Thread Abort Exception → is thrown by run time to abort the thread
- Error Codes, HERCodes, ~~HR~~ onError are ways to handle run time error

CLR → stands for Common language ~~interface~~ instructions

What is precision of float data-type

→ 7 digits

MSIL → micrsoft Intermediate Language.

- CRL's JIT compiles this MSIL code → OS native Code
- Compiler produces MSIL

Portable executable file

MSIL + metadata

- C# code → compiler → MSIL → JIT → Native Code
- JIT does byte code Compilatn and dynamic compilatn
- most used results are cached

~~Managed Code~~

- MSIL and CIL are same
- JIT compiles only when needed

Assemblies

→ Assembli

Metadata
MSIL a.exe
code a.dll

Managed Code

- the code written in .net is managed when it is executed
  - ↳ CLR looks for memory, Ram, debugging,
- codes not runs under CLR is unmanaged code

state-management

↳ we have three types of state
- ↳ ① application state
- ↳ Session state
- ↳ View state
  - ↳ page specific info

ADO

~~active data object~~

→ ActiveX Data Object

Dot providers

① System.Data.SqlClient
- oracleClient
- OleDB.

Applicatn → ADO.net → Database

ADO.net

- Dataset
  - ↳ DataTable
    - ↳ DataRow
    - ↳ DataColumn
    - ↳ Constraint
- Data provider
  - ↳ Connection
  - ↳ Command
  - ↳ Data Reader
  - ↳ Data Adapter

→ DataSource → Data-adapter → Dataset.

Data provider classes

↳ connection object ↳
- ↳ sqlConnection (e.g)
- ↳ cointains info to open connectn
- ↳ it has single Argument (connection string)
- ↳ this cointains "dbname", credentials

② DataSource → name of server.
Initial Catalog → DB name →
Integrated security → login with windows credential
User id → sql user
user password → sql user password

Command object :-

↳ used to execute commands to dB
↳ these are class
↳ ExecuteReader :- returns DataReader when Query return multiple rows
↳ ExecuteNonQuery :- used to insert, update delete.
↳ ExecuteScalar → used when Query returns single scaler (no of rows)

Data reader

→ it is forward readonly.
↳ class SqlDataReader
- ↳ reads data from dB
- ↳
- ↳ cannot be instantiated.

73 Sql Data-adapter

↳ it is not connection oriented

↳ it accepts two augument
① sql command ② the connectn

↳ it is class that provide disconnected data access.

Dataset → Adapter → Database
↳ SelectCommand
↳ InsertCommand
↳ DeleteCommand
↳ UpdateCommand

fill() → executes Select '*' to put in dataset from sour.

Data-Set

↳ it is disconnected in memory data representatn
↳ local copy of data
↳ Dataset
↳ Data tables
↳ Rows
↳ Columns

☆☆☆

⊗ Data reader object cannot be created

⊗ it always require Connection open

Enums →

⊗ are created on stack rather than heap

⊗ by default enumeratn start from 0

Today Date -

DateTime.Today.ToString()
Now.ToString()

Dot net tutorials

msil code is also called managed code

dotnet framewo.
↓
Technologies
↓ ↓ ↓ ↓
Ado ASP WCF LINQ.

⊗ CLR cointains
↳ Security manager
↳ Exception manager
↳ CTS
↳ CLS
↳ Garbage Collector
↳ JIT
↳ Memory manager
↳ (Type System)

⊗ CTS → deals with all language datatypes.
CLR → deals with all syntax of all language (language specific)

machine code / binary code
Native code all are same

⊗ Desktop Applicatio → .exe Assembly.
Web applicatn → .dll assembly.

⊗ assemblies generated by non-dot net is Native code but dot net generated is partial Compile

ILDASM & ILASM

↳ Intermediate language Disassembler

↳ Intermediate language assembler.

⊗ Assembly → manifest → metadata of assembly
↳ Intermediate language
↳ namespace
↳ program class name
↳ constructor
↳ main method
↳ member

Entity framework

⊗ Befor .Net 3.5 ADO.Net for Crud was used

⊗ it is opensource - ORM framework

⊗ it automatically generate tables as per class or classes as per table

UI
↓
BL
↓
data layer
↓
Entity framework
↓
DB

entity framework architecture

① entity Data model → relation of class
② LINQ to entities
③ entity SQL
④ object service layer
⑤ entity client data provider
⑥ ADO.net data provider

CRUD

saveChange() → all the the changes on contexobject are sent to database

⊗ DbContext keeps track of all things

① Insert
context.students.add(student)
context.saveChanges();

Update →

① var student = context.students.find(id)
student.Ramname = "ram"
context.savechanges()

Delete
context.student.Remove()
—— >> —— save

Controller

⊗ class that handles http requests.

http://localhost:xxxx/Home/Index.
↑ method
↓ controller

⊗ mapping are defined in RegisterRoute() of RouteConfig

name: "Default"
← url: "{controller}/{action}/{id}

• edmx file → ~~graphical view~~
↳ ~~of tables~~ Graphical
↳ class, entity

palle

Views

Views have extension .cshtml

⊗ action name = view name.

passing data to view

① ViewBag ② ViewData ③ TempData
④ Session ⑤ Applicatn.

ViewData :- data is stored in object so while retriving casting is required.

```
ViewData["Head"] = "abcdefg"
```

usage →

```
<h2> @ViewData["Header"]

<body>
    @{
        var emp = ViewData["employ"]
        as FirstMvcDemo.model.Employ.
    }
```

⊗ returns from Dictionary.

Viewbag :-

dont need to typecast its return type
⊗ is dynamic and decides at runtime.

⊗ **View Model used to**

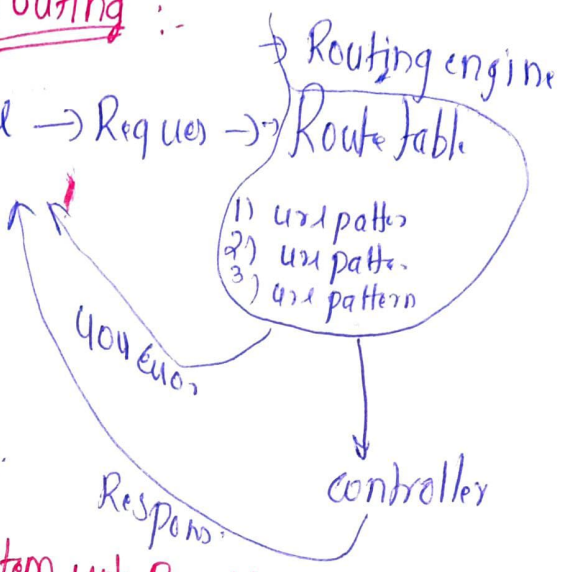
- used to send multiple models to view
- @model is used when
  - ⊗(View(employee)
  - @model.salary
  - @model → tight data typing
  - @Model → retrieve data

Strongly typed view

⊗ when data is passed using viewData, viewBag, data is loosly typed.

```
@model firstDemo.Models.employee
<h1> @Model.address </h1>
```

Routing :-



system.web.Routing

```
routes.MapRoute( controller
                 action
                 id
```

Register Route

Global.asax

Custom route

```
routes.MapRoute(
    name: "Employee"
    url : "Employee/{id}"
    default: new {controller = "Emp
                 action: Index}
```

⊛ Attribute based Routing

[HttpGet]

To enable

in RouteConfig {
    register route
    routes.MapMvcAttribute
        Routes()
    route.map
}

[HttpGet]
[Route("student/{student ID}/courses")]

Attribute with optional param

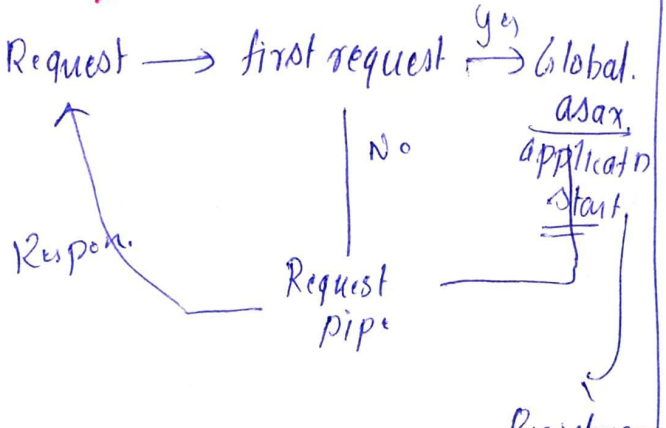
[Route("mvcTest/{studentName?}")]
{studentName="ik R"}

Data Annotation validation

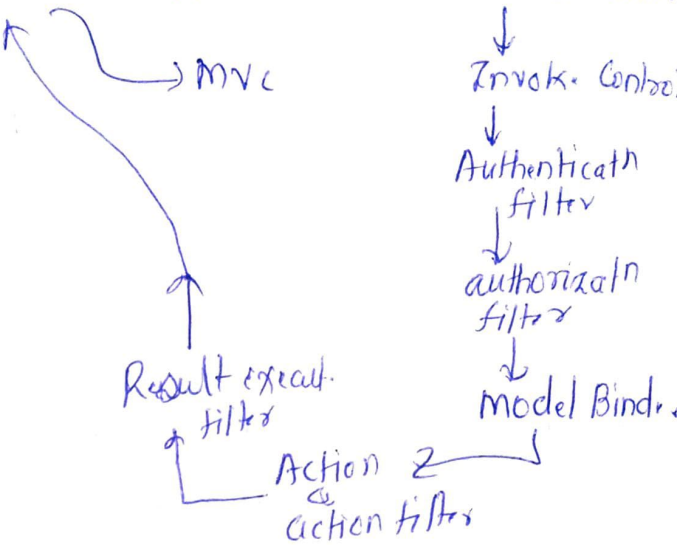
System.ComponentModel.DataAnnotation

① Required ② Range ③ RegularExpression
④ Compare ⑤ stringlength ⑥ Datatype

Application life cycl



Request life cycl :- 76



LINQ :-

⊛ Language integrated Query
> C# 3.0
System.LINQ
* IQueryProvider IQueryabl.
interface to LINQ

⊛ LINQ to objects
→ Query on objects (in memory object)
→ filtering, ordering, Grouping.

Parallel LINQ (PLINQ)

Syntax (query)
from object in datasource ← Initialisatn
where Condition ← Condn
Select objpt ← Selectn

Method
datasource.ConditionMethod().SelectionMethod()

Mixed
(from object in datasource where conditn Sel object).method().

77 List<int> Integer = new List<int>
(){ 1,2,3,4,5,6,7,8
9,10
};

List

Query syntax

Var QuerySyntax = from obj in list
Where obj > 5
Select obj;

list

method

var methodSyntax = list.Where(
x => x > 5).ToList()

mixed

var mixed = (from i in list
where i > 5
select i).sum();

IEnumerable<int>

① system.collection.

② While execution executes select statement
load → filter

③ It is foreach only

IQueryable → applies select + filter rather than only select

To execute Query → it is compulsory to add .toList or else it won't execute.

( ).ToList

Optional parameter

public void (int FN, int SN, params Object[] restNos)

[Optional]

Indexers

to access class like array

Syso("name => " + emp[0])

Syso("lastname" = emp[1])

[<modifiers> <type> this [int index or string name]
{
[get { <statements> }]
[set { <statements> }]
}

class employee {
public object this [int index]
{ get
{ if (index == 0)
return ID;
else
}
set
{
if (index = 0)
Id = convert.ToInt32(value)
}
}

Checked & unchecked → throws exception / ignores

used to enable explicitly
enable overflow checking
suppress overflow checking

int c = checked(a+b);