

# **C#.NET**

**(FAST TRACK NOTES)**

**BY**

# **BALAJI.CH**

**Naresh I Technologies**



24/9/13  
Tuesday

## .NET

(Balaji Sir)

### .NET

.NET stands for Network enable technology -

- refers object oriented & net refers internet.  
so the complt dr .net means through object oriented implement internet application

### \* COM

com stands for component object model.

com is a one of the microsoft technology. using this technology we can implement windows and web applications

### \* Disadvantages of com :-

- i) Incomplete object oriented programming.
- ii) Platform dependent

### \* .NET Framework

- The . net framework is a collection of many small technologies integrated together to developed application which can be executed anywhere.
- .Net framework is a microsoft development platform which is used for development of software applications.

### \* Base Class Libraries

- i) Base class libraries are designed by microsoft.

ii) without this we cannot write any code in .net so base class libraries are known as building blocks of applications of .net. it

- iii) These are installed into the machine, when we instal .net

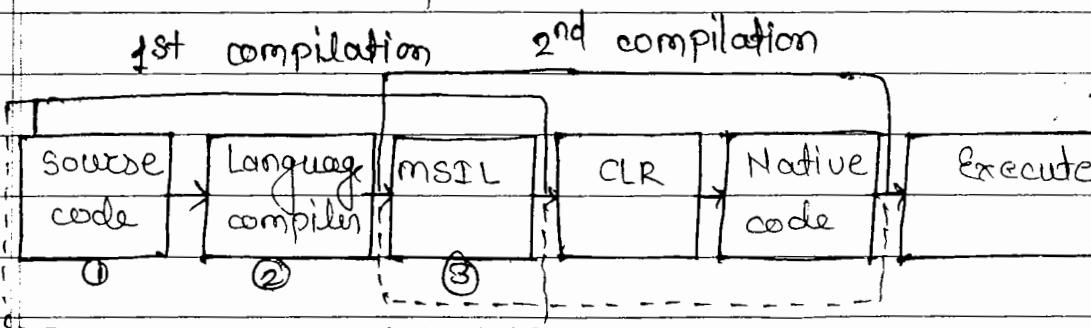
framework into machine.

v) Physical location of base class libraries is  
c:\windows\Assembly

CLR ( common Language Runtime)

CLR is the core component under the .net framework which is responsible converting msIL code into native code & then execution.

CLR execution process



- In .net code is compiled twice
- In 1st compilation source code is compiled by respective language compiler and immediately code is generated known as msIL.
- In 2nd compilation msIL is converted into native code using CLR.
- Always 1st compilation is slow, & 2nd compilation is fast.
- msIL is only CPU independent & will run on only windows operating system using .net framework because .net framework is designed for windows operating system only.
- There is another company now as "Novel" design separate framework is .net monoframework using this framework we can run msIL on different

operating system are Linux, salaries, MACS-OS etc

→ .net is platform dependent using .net framework  
but independent using monolithic framework

→ .net is a framework tool which supports many programming languages.

→ .net support 60+ programming languages. In 60+ programming languages 9 are design by microsoft & remaining are design by non microsoft.

→ List of microsoft design programming languages.

- |             |                        |
|-------------|------------------------|
| 1) VB.net   | 6) Jscript.net         |
| 2) C#.net   | 7) windows powershell. |
| 3) J#.net   | 8) Iron python         |
| 4) F#.net   | 9) Iron Ruby.          |
| 5) VC++.net |                        |
| 8)          |                        |

## Technologies

- 1) ASP.NET (Active server pages .net)
  - 2) ADO.NET (Activex data objects )
  - 3) WCF (windows communication foundation)
  - 4) WPF (windows presentation foundation,
  - 5) WWF (windows workflow foundation)
  - 6) LINQ (Language Integrated query)
  - 7) AJAX (Asynchronous Java script and XML)

- .Net framework is available in 3 diff type.

1) .Net framework :-

This is the general version required to run .NET applications on windows operating system.

2) .Net Mono framework :-

This is required to run .NET applications on other

operating systems like unix, Linux, solaris, etc.

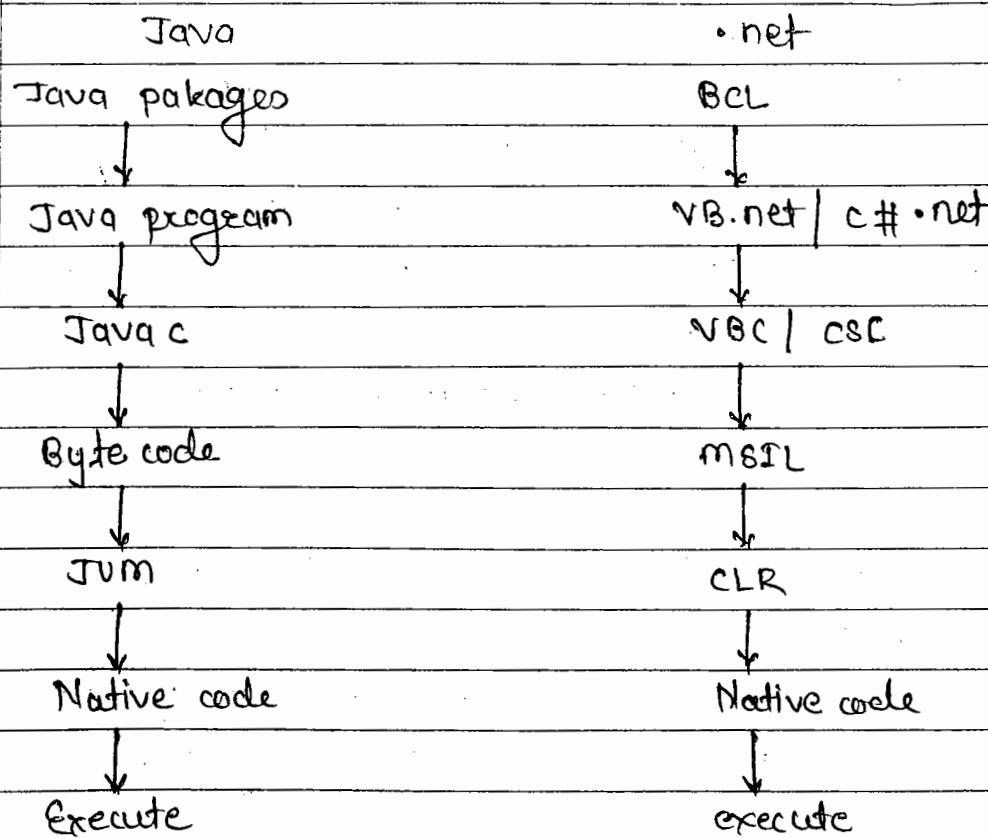
### 3) .Net Compact framework

This is required to run .net applications on other devices like mobile phones, PDA (personal digital assistant) & smart phones.

Every language its own compiler.

Language	Extension	Compiler
VB.net	.VB	VBC
C#.net	.CS	CSC

Execution process difference betn Java & .net application,



5/9/13  
Wednesday

## V Versions of .NET

1998 - NGS → Next generation windows services

2000 The year 2000 → It is a beta version

2000 - VS.NET 2000 → beta version.

FW 1.0 - 2002 - VS.NET 2002



FW 1.1 - 2003 - VS.NET 2003



FW 2.0 - 2005 - VS.NET 2005

FW 3.0 ↓

FW 3.5 - 2008 - VS.NET 2008



FW 4.0 - 2010 - VS.NET 2010



FW 4.5 - 2012 - VS.NET 2012

.NET framework version	Exact version Number	Date of Release	Visual studio
.NET framework 1.0	1.0.3705.0	Feb 13, 2002	Visual studio .NET 2002
.NET framework 1.1	1.1.4322.573	April 24, 2003	Visual studio .NET 2003
.NET framework 2.0	2.0.50727.42	November 17, 2005	No visual studio .NET 2005
.NET framework 3.0	3.0.4506.30	Nov 06, 2006	No visual studio
.NET framework 3.5	3.5.21022.8	Nov 19, 2007	Visual studio 2008
.NET framework 4.0	4.0.30319.0	April 12, 2010	Visual studio 2010
.NET framework 4.5	4.5.50709.17929	15 August 2012	Visual studio 2012

## Assemblies :-

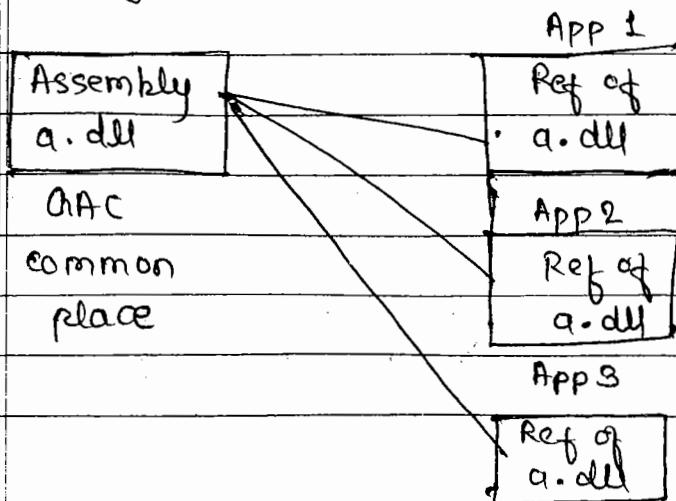
- In .net when an application is compiled into a byte code called MSIL.
- The MSIL code is stored in an assembly. The assembly is contained in one or more portable executable files & end with an .exe or .DLL extension.
- Assembly are two types.
  - 1) Private assemblies
  - 2) public or shared assemblies.

### 1) Private assembly :-

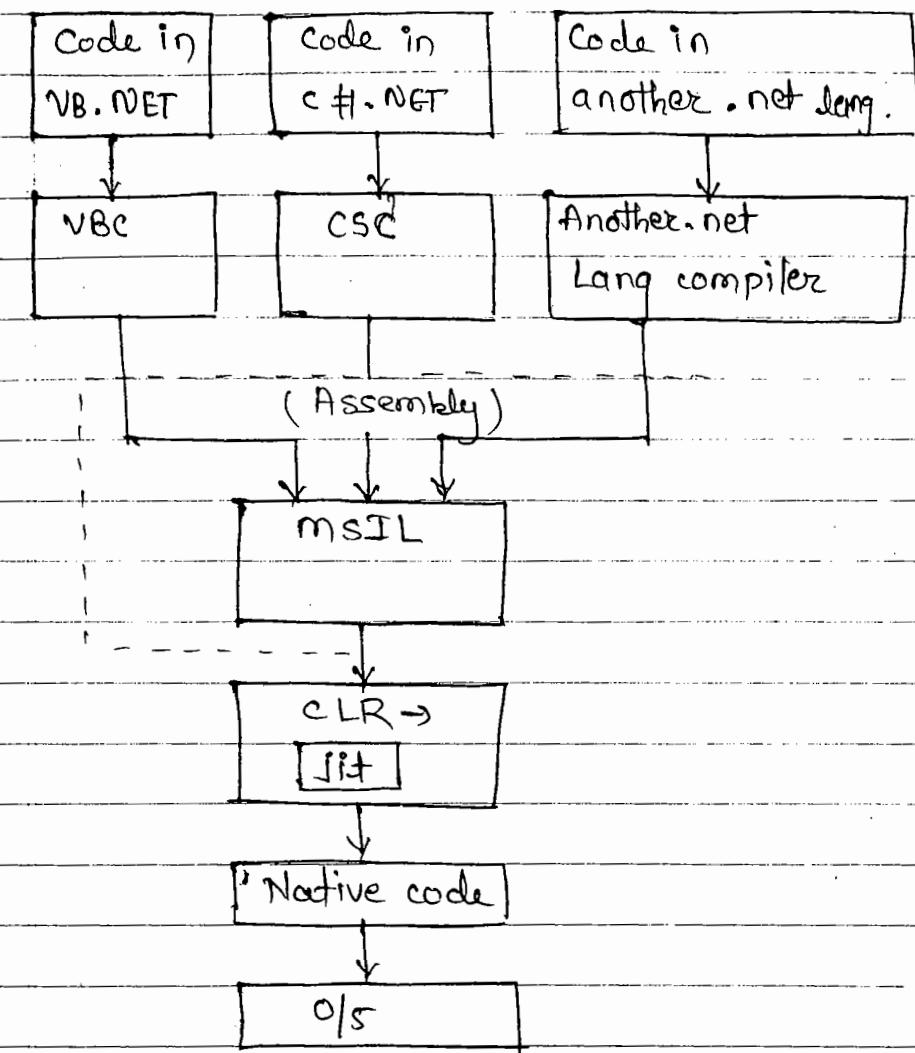
- If an assembly is copied into individual application in which we want to use then it is known as private or local assembly.
- The copy of the assembly is local to that application and other application can not be use this copy.

### 2) Public or shared assemblies

If an assembly is copied into a global place and reference of assembly is used from all other application then assembly is known as public or global or shared assembly.



\* Run time environment of .net application.



\* Components of CLR

CLR contains the following components.

1) Security manager

- This is the initial & most component of CLR
- The security manager component first check privileges of current user that the user is allowed to access the assembly or not.

2) Jit compiler :-

The jit compiler is responsible for compiling

com  
win32 API } Ex of unmanaged code.

→ Translates MSIL code into native code

→ The native code is directly understandable by the system hardware

3) Memory Manager

→ The memory manager component of CLR allocates necessary memory for variables & objects that are to be used by the app!

4) Garbage collector :-

→ This component of CLR deallocates the unnecessary memory of the application after usage automatically

5) Exception Manager :-

→ An exception means the run time error. This component of CLR redirect the processor to execute catch or finally block whenever an exception raised at run time.

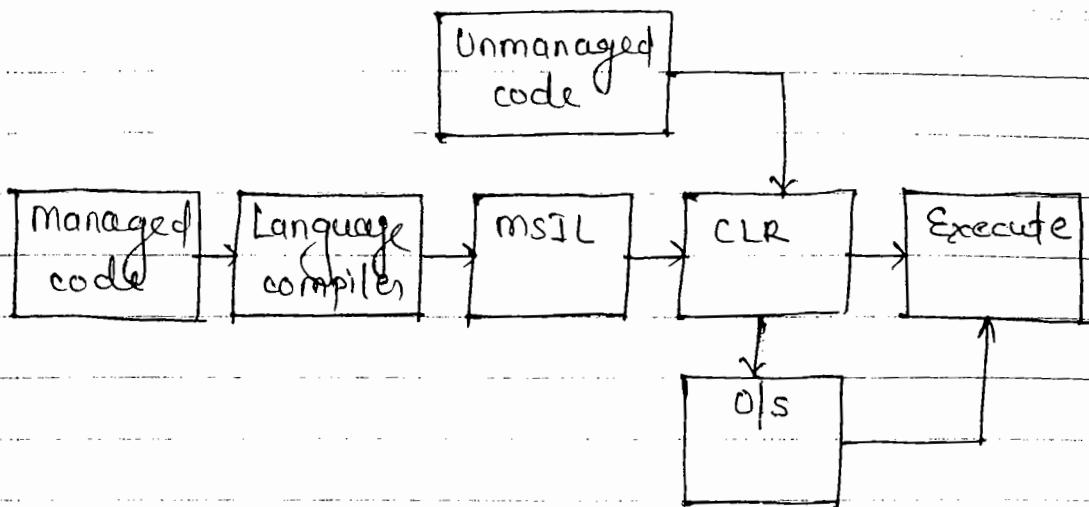
\* CLS (Common Language Specification)

→ CLS is responsible to provide Language & Interoperability.

Language interoperability

→ Providing code execution support that has been written in other programming language is known as language interoperability.

→ Language interoperability is achieved using managed code & unmanaged code.



### Managed Code

- Code for which MSIL is generated after language compiler compilation is directly executed by the CLR is known as managed code.
- CLR will provide all the facilities & features of .NET to the managed code execution. like language interoperability, automatic mem. management, common data type system, exception handling mechanism.

### Unmanaged code.

- Code that has written before development of .NET for which MSIL is not available is not executed by CLR directly rather CLR redirects the code to OS for execution which is known as unmanaged code.
- CLR does not provide any facilities & features of .NET do the unmanaged code execution like language interoperability, automatic mem. management, common data type system, exception handling mechanism etc.
- Examples for un

Example for unmanaged code or

Example for unmanaged code, are COM components, Win 32 API etc.

→ Always managed code execution is faster  
if unmanaged code execution is slow.

### Common type System (CTS)

- .Net supports many programming languages every
- Every programming language has its own data types
- one programming language cannot understand other programming language data type but CLR execute all programming languages datatype this is possible because CLR will contain its own data type which is common to all programming language. at the time of compilation all language specific datatypes are converted into CLR datatype
- This datatype system of CLR which is common to all programming languages of .net is known as common type system.
- This common type system is divided in two categories
  - 1) Value types
  - 2) Reference types.

#### 1) Value types

The data types which stored the data directly into the memory location is known as value types. Ex  $a = 10$

#### 2) Reference types:-

The data types which do not store the data directly into the memory location rather

refers to other memory locations is known as reference types.

$$a = \&P$$

Similarities of .net and java.

- Both of them support to developed windows application, websites, and console application and service oriented applications etc.
- Both uses there own intermediate language java calls if has byte code & .net calls if has msil
- Both are object oriented programming languages.
- Both support multipleting.
- Both support reporting.
- Both are platform independent
- Both support web related languages like java script, xhtml, css etc
- Both support garbage collection which automatically clears the unused memory.
- Both support to develop the application for small devices like POS (personal decision) smart phone, mob phone.
- Both offer better security feature in there own way style.

Difference between .net & java

.NET

1) It is cost effective

JAVA

1) It is open source product which can be freely downloaded on the internet.

2) It supports multiple languages	2) Does not offer multiple languages.
3) It offers IDE as visual studio by micro soft.	3) Does not offer any IDE but other IDE by other windows such as "Eclips"
4) Offers user's easiest & fastest app development, which indirectly reduces the cost of software.	4) Requires much time for application development which indirectly increases the cost of software.
5) Designing the user interface is very much easy with drag & drop technique.	5) UI design required much programmers effort & stretched.
6) AJAX implementation is much easy	6) AJAX implementation is much time taking process required much code to write.

#### • .NET Advantages :-

- It supports multiple languages like visual c++, VB.net, C++-net, etc. so that the program can write the code in interested language.
- It offers flexible data access with ADO.net
- It supports to developed application for small devices like smart phone, mobile phones etc.

- It offers platform independency. because it supports to run the .net applications on other platforms like unix, linux etc with .net mono-framework.
- It offers improved objed oriented programming feature like properties like , sealed classes, inner classes, delegates, annotations, collections, generators, extension method, anonymous meth types, anonymous method, lapped expression , named parameters.
- It offers to write queries. in the programming code itself using the newly added query technology called LINQ (Language integrated query) which is introduced in .net 3.5.
- It still offers a new and attractive feature called WPF (windows presentation foundation)

#### WPF

- which is build of vector based graphics & which enables the programmer create to 2D, 3D, graphics animations, games, adio & video players etc.
- This is newly added feature which is added in .net 3.0
- It offers another better feature called WCF (windows communication foundation) which integrates the several distributed technologies like .net remoting , soap enabled web services etc, which helps the programmer to developed service of new oriented application using .net. this is newly added feature in .net 3.0.

## o Net Architecture :-

FW 4.0	TPL Task library		PLINQ
FW 3.5	LINQ	ADO.NET entity framework.	
FW 3.0	WCF	WPF	WWF card space.
FW 2.0	win forms	ASP.NET	ADO.NET
Common Language runtime Base class libraries.			

## C# .NET

# C#.NET

- C#.NET is one of the Microsoft programming language.
- C#.NET is most powerful programming language because C#.NET will contain all features of C++, Java, VB and some additional features.
- C# with framework any application simplified.
- C#.NET is an object oriented programming language.
- It is a case sensitive language.

## .NET Framework Advantages in C# view

- 1) Good GUI features
- 2) Any database will support
- 3) It prepares web application.
- 4) Very High scalability (more speed, reusability)

Using

Using is keyword refers .NET base class libraries

## Types of Application

i) Windows application.

ii) Web application

iii) Class library :-

Using this template create our own DLL files

iv) Windows forms control library :-

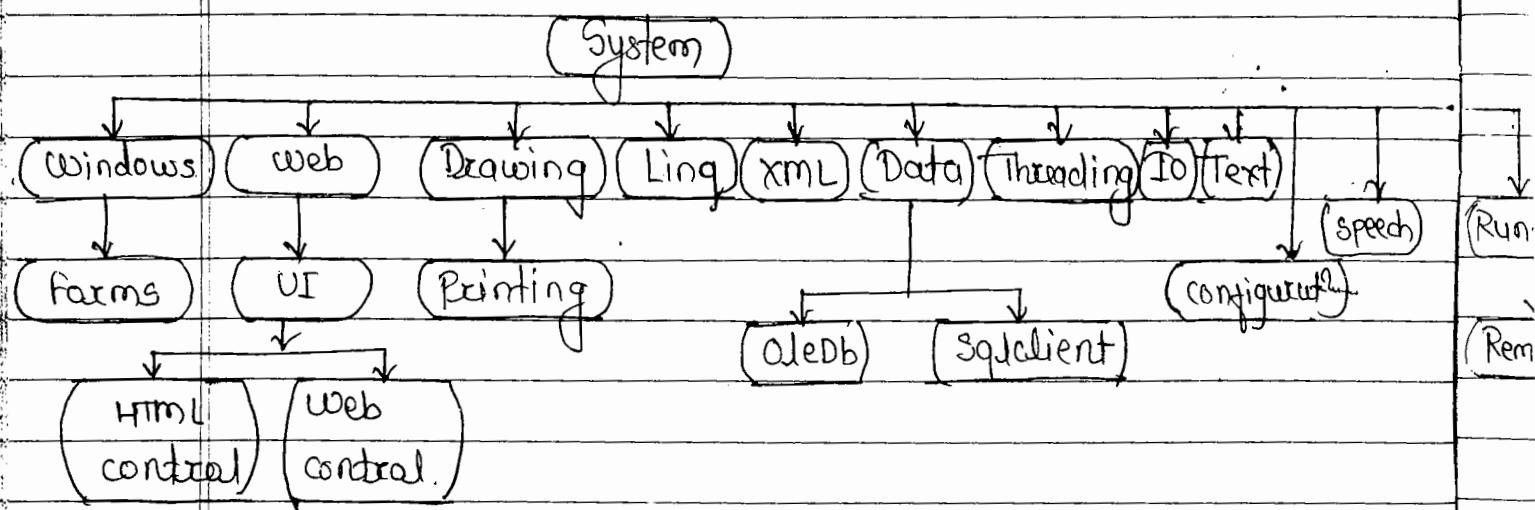
Using this template create our own user controls.

v) Console Application :-

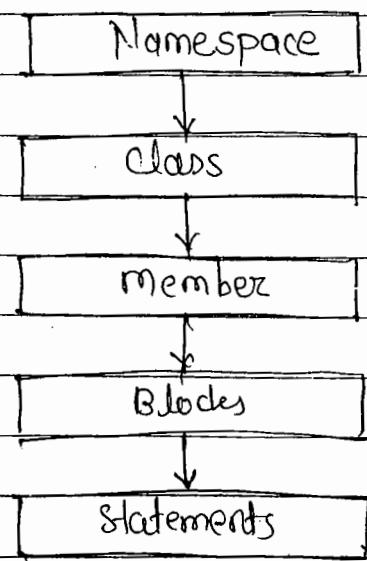
It is a command line app this app run on DOS page.

## • .NET framework class library architecture

- 1) In .net the base class libraries are divided into collection of namespaces
- 2) A namespace is a collection of classes and sub namespaces.
- 3) The inner namespaces contain by another name space is called as "sub namespace"
- 4) The most frequently used namespaces of base class library are listed here.



## Programming Structure



Example

Namespace XYZ

{

Class sample

{

Void Main( )

{

|| statement

{

}

}

}

→ C# .net is a case sensitive language.

b) (Runtime) → Every statement end with semicolon ( ; )

(Remoting)

Open the notepad & write the following code.

Class sample

{

static void Main( )

{

System . Console . Write ("Hello");

namespace class method

}

}

→ Save the program any drive any folder name  
any filename.cs

→ Execution of program.

Steps

Start → go to program → go to MS VS 2012 → go to  
visual studio → command prompt  
Tools

D: (enter)  
D:

D: \ > cd foldername

D: \ foldername > csc filename.cs

D: \ foldername > filename (Enter)

What is visual studio

1) visual studio is nothing but the visual IDE (Integrated development environment) which is needed development of software applications with .net framework.

9

2) The IDE integrate 3 features like

i) editor

ii) Compiler

iii) Interpreter

Console Application Using Visual studio

Steps

Go to start → programs → microsoft vs 2012

→ New project → select language visual c#

→ Select template → console application →

specify name & location → click ok button

Ex

Using System ;

namespace ConsoleApplication1

{

    class program

{

        static void Main ()

{

            Console.WriteLine ("This is First program");

}

        Console.Read ();

}

Execution: Start debugging (F5)

### \* Console Application :-

- 1) These application contains similar user interface to the operating system like ms dos or Unix.
- 2) It is known as CUI (character user interface) application.
- 3) These are similar to C or C++ applications
- 4) These are smaller in size
- 5) It does not contain any graphical features like mouse pointer, colours, fonts, buttons etc.

### Importing Section

- 1) This Section contains importing statements that are used to import the framework class library
- 2) This is most similar to the include statements in C language

Syntax

Using Namespace Name ;

Ex :-

Using System ;

Note : If the required namespace is a member of another namespace we have specify the parent of child namespaces separated with dot (.)

Ex

Using System ;

Using System::Io ;

Using System::Data ;

## 2) Namespace Declaration :-

1) Here the user design namespace is to be declared.

## 2) Syntax

Namespace NamespaceName

{

}

3) Generally the NamespaceName will be same as project name.

## 3) Class Declaration

1) This is to declare the start of the up class of project

2) In every .net app there should a start up class

3) In this app the start up class name is "program"

4) The startup class nothing but a class which contain main() method.

## Syntax

Class className

{

}

## 4) Main() or Main Method

1) In C or C++ language the main method is the starting execution point of the app

- 2) When app is executed the main method will be executed first.
- 3) This method contains the main logic of the application

### Console :-

1) To implement the user interface in the console application microsoft has provided a class called console

### 2) Library :

`System.Console`

3) With the support of properties & methods of console class you get implement the user interface in console application all the properties & methods of console class or static members so that u can access them without creating any object of other class.

### Properties of Console class.

Properties	Description
Title	Specifies the title of the console window.
Background colour	Specifies background colour of the text
Foreground colour	Specifies foreground colour of the text.
Cursor size	Specifies the height of the cursor in the console window. (1 - 100)

## Methods of Console Class.

Method	Description
Clear()	Clear the screen
Beep()	Plays a beep sound using PC speaker at run time.
ResetColor()	Resets the background & foreground colors to its default state
Write("string")	Display the specified message on the console window.
WriteLine("string")	Same as Write() method, but automatically moves the cursor to the next line after printing the message.
Write(variable)	Displays the value of the given variable
WriteLine (variable)	Displays the value of the given variable along with moving the cursor to the next line.
Read	Reads a single character on the keyboard and returns its ASCII value
ReadLine()	Reads a string value from the keyboard and returns the entered value (in string mode only)

1st Example ↴

Open visual studio 2012



New project



Select language



visual c#



Select template console app



specify name & location



click ok button.

```
static void main ( string [ ] args )
```

```
{
```

```
    console . WriteLine ( " Welcome " );
```

```
    console . Write ( " NarshTechnologies " );
```

```
    console . Read ( );
```

```
}
```

```
}
```

Example 2 ↴ Bring the variable data on the screen.

class program

```
{
```

```
static void main ( string [ ] args )
```

```
{
```

```
    string s = " Ganesh " ;
```

```
    console . Write ( s );
```

```
    Console . Read ( );
```

Example to read the variable data at run time.

class program

{

{

    static void Main(string[] args)

}

        Console.WriteLine("Enter Your name:");

        string name = Console.ReadLine();

        Console.WriteLine("Your name is :" + name);

        Console.Read();

Example 4 : to Implement add any two number.

class program

{

{

    static void Main(string[] args)

}

        Console.WriteLine("Enter any two number");

        int a, b, c;

        a = int.Parse(Console.ReadLine());

        b = int.Parse(Console.ReadLine());

        c = a+b;

        Console.WriteLine("The sum is :" + c);

        Console.Read();

}

Ex.

### Example 5

```
static void Main ( string [] args )
```

```
{
```

```
// Output message
```

```
Console.WriteLine ("The .NET framework is :");
```

```
Console.WriteLine ("VB.NET +");
```

```
Console.WriteLine (" C# .NET +");
```

```
Console.WriteLine (" ASP.NET +");
```

```
// Input message
```

```
Console.WriteLine ("Enter your name :");
```

```
string name = Console.ReadLine ();
```

```
// Change the title of console window
```

```
Console.Title = "console Demo";
```

```
// Clear the screen
```

```
Console.Clear ();
```

```
// Change the background colour , foreground colour ,
```

```
Console.BackgroundColor = Console.Color - Yellow
```

```
Console.ForegroundColor = Console.Color - Blue
```

```
Console.WriteLine ("Hello : " + name);
```

```
Console.Read ();
```

```
}
```

Pla

control + alt + L

Place holder syntax

static void Main (string [] args)

{

Console.WriteLine ("Enter Your name :");

string s = Console.ReadLine ();

Console.WriteLine ("Hello {0}", s);

Console.Read ();

Example .

Console.WriteLine ("Enter Your name :");

string s = Console.ReadLine ();

string s1 = "abc"; Console.WriteLine ("Hello {{0}} {{1}}", s, s1);

Console.Read ();

C# .net supports two types of user interfaces

- 1) Character user interface
- 2) Graphical user interface.

1) CUI

- Character  
app
- 1) This user interface supports a black screen with some characters only.
  - 2) It is not attractive to further user. that's why it is not suitable for live project

2) GUI

- Windows  
app
- 1) It supports graphical component like windows mouse pointer buttons, tool box etc
  - 2) It is attractive to further user. that's why it is suitable for the live project

## Windows forms Appl

- 1) Windows forms app will provide complete GUI facilities
- 2) These application are designed similar to the windows operating system.
- 3) In windows forms application we can work with all controls like text boxes buttons radio buttons etc
- 4) It can also be called as windows application.
- 5) It is a collection of windows forms

## Form

- 1) Form is graphical container which can contain the graphical controls like labels text boxes buttons etc.
- 2) It acts as container for the controls.
- 3) It is also called as a "window".
- 4) It has the visual appearance with a title bar, icon, control box [with minimize button, maximize button & close button]

## steps

start → programs → microsoft visual studio 2012 → visual studio 2012 → new project → select language visual c# → select template windows forms app  
specify name of location. click ok button.

## \* Project hierarchy

Tools menubar → view → solution explorer (control + ALT + L)

## \* Add A New Form

Go to solution explorer → right click on app → add new item → select template windows form → click add button

Note 1) In c# .net every form is a class and also a container, collection of controls is called as container.

2) Every form execution starts with main function, the common main function is available separately under program.cs class file. so every form execution starts with program.cs class file.

## Windows Controls :-

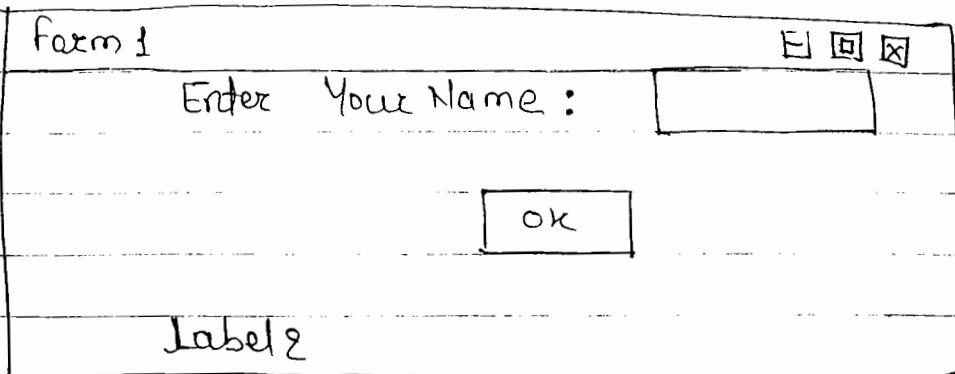
Every windows form controls are designed to provide an application with inputs and outputs. There are many controls are provided in c# the visual representation of an application

### 1) Label :-

It is used to display static information to the form

### 2) Textbox

Textbox is used to get offset values from the user



Label 2

How to change the text.

Right click on Label 1 control go to properties → change the text → Enter your name

### Coding

Double click on ok button.

Private Void button1\_Click (object sender, EventArgs)

{

string s = textBox1.Text;

label2.Text = "Your name is" + s;

↳ concatenation.

}

}

Start debugging F5

C

Control + ALT + X — Tool box short key.

Properties — F4

Tool Box Exp — Ctrl + ALT + L

## integer to string - C.ToString

✓ Example 2 - Design the form

Form 1		
<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>		
Enter A Value :	<input type="text"/>	
Enter B Value :	<input type="text"/>	
Result :	<input type="text"/>	
<input type="button" value="ADD"/>	<input type="button" value="SUB"/>	<input type="button" value="MUL"/>
<input checked="" type="button"/> <small>OK</small> <small>textBox1.Text = " ";</small>		

double click on ADD button

```
{  
    int a, b, c ;  
    a = int.Parse(textBox1.Text); string to int  
    b = int.Parse(textBox2.Text);  
    c = a+b;  
    textBox3.Text = c.ToString();  
}  
}
```

Double click on SUB button.

```
{  
    int a, b, c ;  
    a = int.Parse(textBox1.Text);  
    b = int.Parse(textBox2.Text);  
    c = a-b;  
    textBox3.Text = c.ToString();  
}  
}
```

1/10/13  
Tuesday

Double click on MUL button,

```
{ int a, b, c ;  
    a = int.Parse(textBox1.Text);  
    b = int.Parse(textBox2.Text);  
    c = a * b  
    textBox3.Text = c.ToString();  
    } } Clear textBox.  
    } } textBox1.Text != " ";  
    textBox2.Text = " ";  
    textBox3.Text = " ";
```

## Data types in C#

### • Integer Types.

byte	System.Byte	0 - 255
short	System.Int16	-32768 - 32767
int	System.Int32	-2147483648 - 2147483647
long	System.Int64	-2147483648 - 2147483647
sbyte	System.SByte	-128 - 127
ushort	System.UInt16	0 - 65535
uint	System.UInt32	0 - 4294967295
ulong	System.UInt64	0 - 18446744073709551615

### Decimal or Float Type

float	System.Single	4 bytes
double	System.Double	8 bytes
decimal	System.Decimal	16 bytes

### Boolean Type

bool	System.Boolean	true or false
------	----------------	---------------

## Character Type

char	System.Char	2 bytes
string	System.String	

## Root Type

object	System.Object
--------	---------------

## Operators

### Arithmetic

+ , - , \* , / , %

### Assignment

=, +=, -=, \*=, /=, %=

### Comparison

==, !=, <, <=, >, >=, is, as, like

### Concatenation

+

### Increment and Decrement

++ , --

### Logical

&&, ||, 1

Conditional Statement :-

A block of code that executes basing upon a condition, they are of 2 types :

Branching  
Looping

Conditional Branching

- 1) IF
- 2) Switch

1) IF Syntax

```
If (<condition>)
  <stmts>;
else if (<condition>)
  <stmts>;
-----
else
  <stmts>;
```

2) Switch Syntax

```
switch (<expression>)
```

```
{  
  case <value>:  
    <stmts>;
```

```
  break;  
-----
```

```
  default:  
    <stmts>;  
  break;  
-----
```

```
}
```

## 2) Conditional Loops

c# provides 4 different loops that allows you to execute a block of code repeatedly until a certain condition is met, those are,

- for loop
- while loop
- do - while loop
- foreach loop

A loop required.

- 1) Initialization
- 2) Condition
- 3) Iteration

for loop Syntax

```
for (initializer; condition; iterator)  
    <stmts>;
```

while loop syntax

```
while (condition)
```

```
{
```

```
<stmts>;
```

```
{
```

do - while loop Syntax

```
do { <stmts>;
```

```
{
```

```
    } while (condition);
```

## foreach Syntax

foreach ( Datatype variable in collectionname )

{

<stmts>;

}

## \* Jump Statement

C# provides a number of statements that allows you to jump immediately to another line in a program.

goto: It allows you to jump directly to another specified line in the program, indicated by a label which is an identifier followed by a colon.

Break:

It is used to exit from a case in a switch statement and also used to exit from for, foreach, while, do...while loops which will switch the control to the statement immediately after end of the loop.

Continue

This can be used only in the loop statement which will jump the control to the iteration part without executing the statement present after it.

### ✓ Ex 1) Design A Form

A hand-drawn diagram of a Windows-style application window titled "Form". The window contains two text input fields: one for "User Name" and one for "Password", each preceded by a colon and followed by an empty text box. Below these fields is a large rectangular button labeled "Login".

Private void button1\_Click (object sender,

```
{  
    if (textBox1.Text == "Admin") && textBox2.Text == "Admin")  
    {  
        MessageBox.Show("Login successfully");  
    }  
    else  
    {  
        MessageBox.Show("Login failed");  
    }  
}
```

### Ex 2 Design a form

Generate textbox1, textbox2, textbox3,

Generate Number

From : [ ] To : [ ] Go [ ]

112,

Release break point F11

### Code Button

```
{  
    int n1, n2 ;  
    n1 = int.Parse(textBox1.Text);  
    n2 = int.Parse(textBox2.Text);  
    textBox3.Clear();  
    for (int i = n1; i <= n2; i++)  
    {  
        textBox3.Text = textBox3.Text + i + ", "  
    }  
}
```

### Break point step

Right Click on button event handler

Go to breakpoint → insert breakpoint → run app

### Release break point

After press F11

### Picture Box

Using this control display the images in the form

Go to tool box, select picture box control drag & drop form → go to picture box control property → select property Image → click browse option → select local resource radio button → click import button → select any image → click ok button.

Go

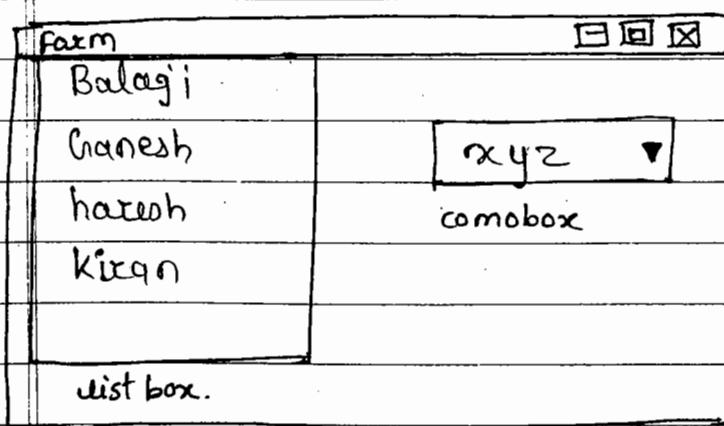
11/10/13  
21

How to picture box property → select property  
size more enter dropdown list → select stretch  
image.

### List Box, ComboBox

- 1) List Box, ComboBox Controls are used to display multiple items
- 2) List box, & combobox are allow list of items they are use to present multiple values and request the user to select one or more values from the list of choices.
- 3) Most of the properties are similar for list box & combobox.

Go to tool box → select list box, combobox control → drag & drop in the form  
Go to list box control properties select property → items → click browse option → enter the value →



## Properties of List Box

Property	Description
Items	: Contains the list of items that can be displayed in the list box.
Selection Mode :	Specifies mode of the item selection. no None : No item can be selected. One : - Single Item can only be selected multisimple : multiple items can be selected directly by clicking on the item multiextend : Multiple items can be selected with control + click
Sorted :	Enables or disable automatic sorting of items

## Runtime properties of List box control

Property	Description
List box name. selected item	Represents the currently selected item in the list box
List box name. selectedIndex	Represents the index of the currently selected item in the list box.
List box name. items.count	Represents the total no of items in the list box
List box name. items[index]	Gets the specified item based on the given index

List box name. Selected items . count : Represents the total no of selected items . count Items currently selected in the list box control.

### \* Methods of List Box

1) List box name. Item. Add (value)

↳ Adds a new item at the end of List box items

2) List box name. Item. Insert (position, value)

Insert a new item at the specified position.

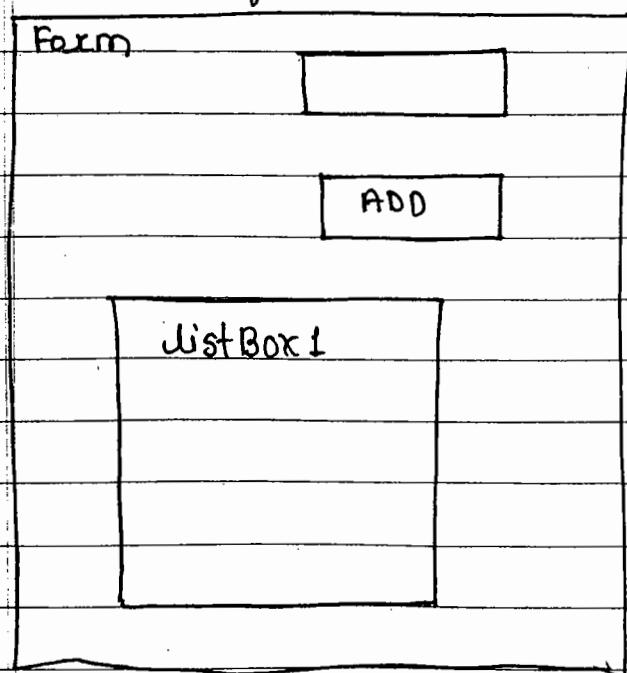
3) List box name. items. RemoveAt (index)

Removes an item based on its index

4) List box name. items. Clear ()

Removes all the items in the list box

✓ Design a form.



```
{  
    listBox1.Items.Add(textBox1.Text);  
    textBox1.Clear();  
    textBox1.Focus();  
}
```

Clear is a method to clear all text from the text Box control

focus is method to set input focus to the control.

Implement the variations. (Duplicate value & space)

```
{  
    if (textBox1.Text == "")  
{
```

messageBox.Show("Enter any text");

textBox1.Focus();

return;

```
}
```

for (int i=0 ; i< listBox1.Items.Count ; i++)

```
{
```

if (textBox1.Text == listBox1.Items[i].ToString());

```
{
```

messageBox.Show("Duplicate are not allowed");

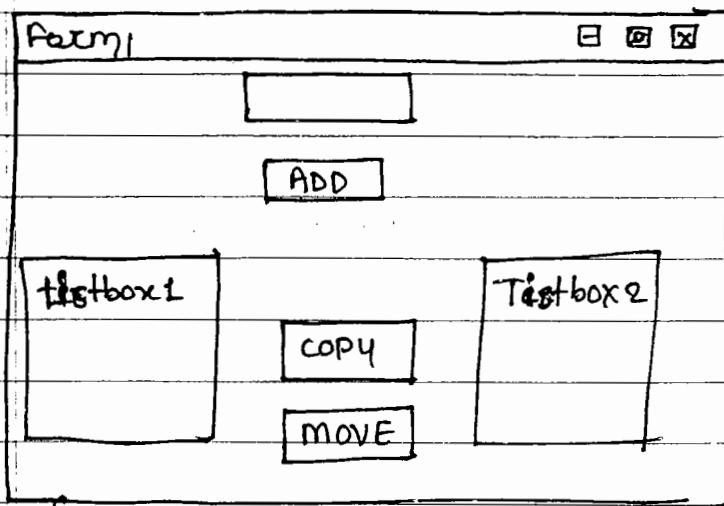
return;

```
}
```

```
}
```

```
listBox1.Items.Add(textBox1.Text);
textBox1.Text = clear();
textBox1.Text.Focus();
} textBox1.Clear();
} textBox1.Focus();
```

Example 2 (copy the value) & (Remove the value)



```
if (listBox1.SelectedIndex != -1)
{
```

```
    listBox2.Items.Add(listBox1.SelectedItems);
```

```
}
```

move button code.

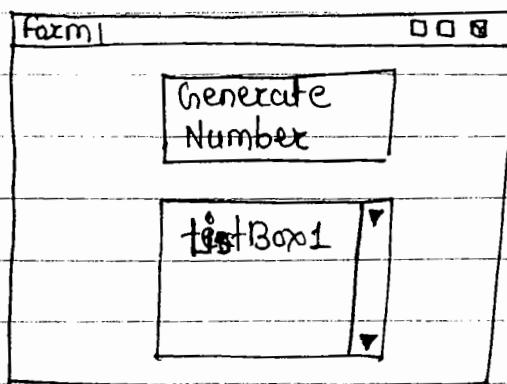
```
if (listBox1.SelectedIndex != -1)
{
```

```
    listBox2.Items.Add(listBox1.SelectedItems);
```

```
    listBox1.Items.Remove(listBox1.SelectedItems);
```

```
}
```

At runtime click on the generate button &  
Generate 1-1000 number in list box 1.



{

```
for (int i=0 ; i<=1000 ; i++)
```

{

```
listBox1.Items.Add(i);
```

}

#### \* Event handler

- 1) An Event handler is a method which is called automatically whenever the user performs a certain event at runtime.
- 2) An event handler should be defined within the form class.

Syntax

```
Private Void ControlName-EventName (Object Sender, EventArgs e)
```

{

}

In above syntax there are two argument

3/10/13  
Thursday

### 3 - Event symbol

#### a) Sender :

The Sender represent control based on each the event is raised.

Eg. In the "button1-Click event handle"  
The sender argument represents button1 control.

#### b) e

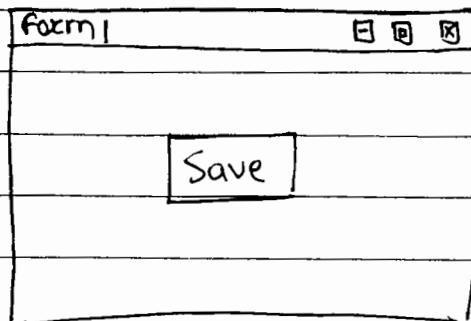
e contains some additional information about the event.

Ex : In the button click event the position of button will be represented where it is click

→ In the previous example we have generated the event handler by double clicking on the controls. Every control is having multiple events. but when u double click the control it will generate the event handler for only one event. The particular event can be called as "default event"

Ex - The default event for the event is "click" event,

If we want to implement the event handlers for other controls you required do follow steps.



Design the button

Go to button control property → go to events  
→ double click on mouse enter event

\* private void button1\_MouseEnter (object ...

{

button1.BackColor = Color.Yellow;

button1.ForeColor = Color.Blue;

}

Go to button control properties → go to events →  
double click on move leave event

\* private void button1\_MouseLeave (object

{

button1.BackColor = Color.Blue;

button1.ForeColor = Color.Yellow;

}

## \* Object Oriented Programming (OOPs)

Types of programming languages.

1) Structure Oriented programming Language:

The programming implementation flow depends on structures that means we need to create the structure for each data type entity.

Those structure member can be created in individual function.

Ex: Pascal, C, C++ etc.

2) Object oriented programming lang

The programming implementation flow depend on "classes". In the main method you need to create instances for the classes and access those method.

These instances are called "objects"

Ex, C++, Java, .NET all languages.

### \* Class

1) In object oriented programming Languages it is must to create a class for representing data. Class contains variables for storing data & functions do specify various operation that can be performed on data.

2) Class will not occupy any memory space & hence it is only logical representation of data

C# class members - private  
int a;

Syntax :-

Class Classname

{  
    // Variables                                  } class member  
    // function  
    // properties ... etc.  
}

\* Access Specifiers or modifiers.

Access specifiers or modifiers are those provide scope of accessibility.

Each and every class member before define scope of accessibility.

There are 5 possible forms of accessibility.

- 1) Public
- 2) Private
- 3) Internal
- 4) Protected
- 5) Protected internal

1) Public :-

If a member or type is declared as public then it is accessible to the correct class and other classes of current assembly (project) & also other assembly.

2) Private :-

Private members are accessible within the class they cannot accessible other classes of current assembly and also other assembly. by default C# members are class private

### 3) Internal :-

If a member or type is declared as using internal keyword then it is accessible to the current class and other classes of current assembly.

They cannot accessible other assembly.

class →	Same assembly			Other assembly	
Access modifier	Same class	Derive class	Non-Derived class	Derive class	Non-Derived class
private	✓	✗	✗	✗	✗
protected	✓	✓	✗	✓	✗
Internal	✓	✓	✓	✗	✗
protected Internal	✓	✓	✓	✓	✗
Public	✓	✓	✓	✓	✓

### \* Function :-

- 1) Any programming language will have method
- 2) A method is a reusable piece of code which can recalled again & again & used to perform required tasks.
- 3) In any programming language a method can be of two types.

1) Procedure  
2) function.

- 4) A procedure is a method which does not ~~return~~ return any value to the calling place.
- 5) A function is a method which always will return single value to the calling place.
- 6) As a function will return some value to the calling place. mentioning returns type is compulsory.  
If function does not return any value then mention return type as void.

Note: C# will support only functions and does not support procedure, whereas VB.net will support both function and procedure.

### Syntax

Accessspecifier Returntype MethodName (arglist)

{

}

Example : <sup>(userdefined)</sup>

Public int add() // Default method

{

}

public void getdata(int a) // parameterised method

{

}

Note : Default accessibility of class will be internal.

Example :

Open visual studio → select language visual c++ and select template → console application → specify the name of location click ok button.

namespace ConsoleApplication1

{

class sample

{

int a, b;

public void m1 (int aval)

{

a = aval;

}

public void m2 (int bval)

{

b = bval;

}

public int add ()

4/10/13  
Friday

{

return a+b;

}

{

class program

{

static void main (string [] args)

{

sample obj = new sample ();

obj. m<sub>1</sub> (10);obj. m<sub>2</sub> (20);

Console. Write (obj. add());

(Console. Read ());

Note ;

✓ Object &amp; Instance

Class will not occupy any memory space Hence to work with the data represented by the class you must create variable for the class which is called as an object.

When an object is created by using the keyword new, then memory will be allocated for the class in heap memory area which is called as object & instance of its starting address will be stored in the object in stack mem. area.

When an object is created without the keyword new then memory will not be allocated in heap, i.e. instance will not be created in the object.

stack contain value null.

When an obj contain null then it is not possible. when obj try to access the members of class using that obj

### Example 2

Open visual studio → select template → windows form application → goto solution explorer → right click on app → add new item → select template class file → name → test.cs → click add button.

#### Class Test

```
{
```

```
public int add (int a , int b)
```

```
{
```

```
return a+b;
```

```
}
```

```
public int sub (int a , int b)
```

```
{
```

```
return a-b ;
```

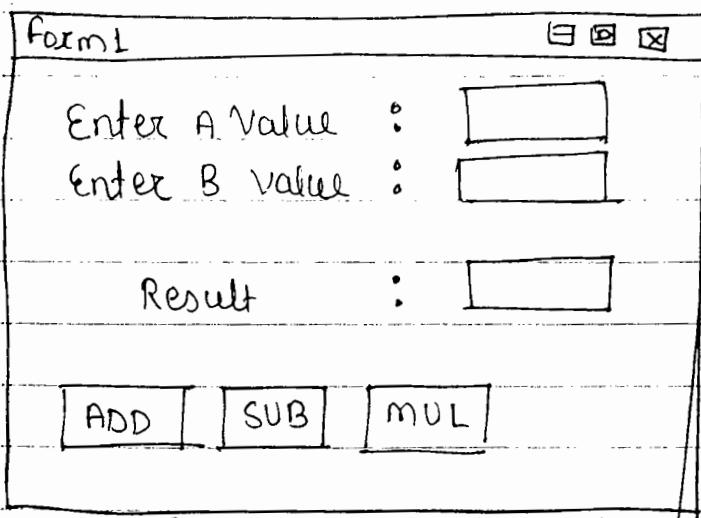
```
}
```

```
public int mul (int a , int b)
```

```
{
```

```
return a * b ;
```

```
}
```



For ADD Private Void

```
Test obj = new Test();
int c = obj.add (int.Parse(textBox1.Text), int.
Parse (textBox2.Text));
textBox3.Text = c.ToString();
```

}

For SUB

{

```
Test obj = new Test();
int c = obj.sub (int.Parse(textBox1.Text), int.Parse
(textBox2.Text));
textBox3.Text = c.ToString();
```

## Class Library

### Creation of a DLL file

- 1) Using class library template create our own dll files. This DLL file can not executable directly because it does not contain execution entering point.
- 2) It generates only dll file ('Dynamic link library')
- 3) It a supertive file.

Open visual studio → select template  
→ class library → specify name & location  
→ click ok button.

namespace ClassLibrary1

{

public class Demo

{

public int add (int a, int b)

{

return a+b;

}

public string getset()

{

return " naresh technologies ";

}

{

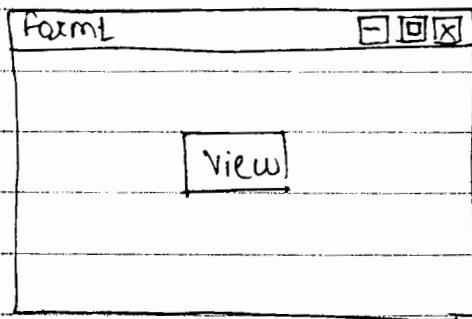
Go to Build → build class library 1

Result:- DLL file created successfully.

c:\ foldername \ classlibrary1 \ bin \ debug \

✓ → Open visual studio select template → windows forms  
app.

Go to solution explorer right click on references →  
add reference → click browse button →  
select dll → click ok button



Using Class Library 1;

private void button ...

{

    Demo obj = new Demo();

    MessageBox.Show(obj.getset());

}

}

5/10/13  
Saturday

## Properties :-

- 1) If property is a member of a class used to write the data in the data field and read the data from the data field of a class.
- 2) A property get never stored the data just used to transfer the data.
- 3) Properties or members that provides a flexible mechanism to read, write the values of private field.

### Accessors within a property

#### a) Set Accessor

- 1) Set Accessor If used to write the data into the data field
- 2) This will contain a default and fixed variable named as "value".
- 3) Whenever we call the property to write the data any data will supplied will come unstored in value variable by default.

#### Syntax

Set

{

DatafieldName = value ;

}

#### b) Get Accessor

is used to read the data from the data field using this accesser we can not write a data

## Syntax

Get

{

    Return Data.fieldName ;

}

## Types of Properties

- 1) C# .net will support 3 types of properties
  - a) Read Only property
  - b) Write only property
  - c) Read write property
- 2) In c#.net 2.0 - using Refactor Automatically generate properties.
- 3) In c#.3.0 microsoft introduced Auto implemented properties.

Go to solution Explorer →

✓ Open visual studio → Select template → class library → specify name & location →

namespace ClassLibrary1

{

    public class sample

{

        int a, b ;

        public int B

{

            get { return b ; }

            set { b = value ; }

}

```

public int A
{
    get { return a; }
    set { a = value; }
}

public int add()
{
    return a + b;
}

```

Right click on variable → go to refactor  
→ go to Encapsulate field → click ok →  
click apply

Go to build → build class library 1 →

Open visual studio → select template  
windows forms appl.

Go to solution explorer → right click on  
references → add reference → click  
browse button add all file.

Form1	X
Enter A Value : <input type="text"/>	
Enter B Value : <input type="text"/>	
Result : <input type="text"/>	
<input type="button" value="ADD"/>	

No

Using class library

Private void button ...

{

Sample obj = new sample();

obj.A = int.Parse(textBox1.Text);

obj.B = int.Parse(textBox2.Text);

textBox3.Text = obj.add().ToString();

} textBox1.Clear();

2  
3

Example :-

Open visual studio → select template console application → specify name & location → click ok button

namespace ConsoleApplication1

{

public class Dept

{

int dno;

string dname, location;

public int Dno

{

get { return dno; }

set { dno = value; }

}

public string dname

{

get { return dname; }

set { dname = value; }

}

```
public string Location  
{  
    get { return location; }  
    set { location = value; }  
}
```

```
class program  
{
```

```
    static void Main(string[] args)  
{
```

```
        Dept obj = new Dept();  
        Console.WriteLine("Enter Department Details");  
        obj.Dno = int.Parse(Console.ReadLine());  
        obj.Dname = Console.ReadLine();  
        obj.Location = Console.ReadLine();  
        Console.WriteLine("DeptNo :" + obj.Dno);  
        Console.WriteLine("Dname :" + obj.Dname);  
        Console.WriteLine("Location :" + obj.Location);  
        Console.Read();  
    }
```

```
}
```

Note: In Above example data fields of Dept are private (Default) so we can not accessible from program class. We transfer the data into data fields with the help of properties and properties code is hidden from program class but we are able to get the result finally. So we providing abstraction to the data field here. This is known as data abstraction.

So properties will provides data abstraction.

Data Abstraction :-

Data Abstraction means hiding the implementation but providing service

### \* Auto Implemented Properties

- 1) This is introduced in c# .net 3.0
- 2) To simplify the syntax of property declaration that contain no extra logic in set accessors and get accessors this concept is introduced.
- 3) According to this no code for set & get accessors needed to write
- 4) No extra variable is needed to be declared.

Syntax :

Accessspecifier Datatype propName ex: Public int A

{

Get ;

Set ;

}

{

Get ;

Set ;

}

### ✓ Example :-

Open visual studio → select template → windows farms app.

Go to solution explorer → right click on app → add new item → select template → classfile → name it has test.cs → click add button.

7/10/11  
m

namespace WindowsFormsApplication2

{

class Test

{

public int A

{

get ;  
set ;

}

public int B

{

get ;  
set ;

}

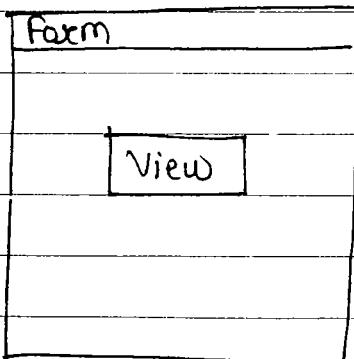
public int add ()

{

return A+B ;

}

}



Design a form

Private

{

Test obj = new Test();

obj.A = 100;

obj.B = 200;

messageBox.show(obj.add().ToString());

}

7/10/13  
Monday

## Inheritance :-

- 1) The process of implementing the "Parent to child" relationship between two or more classes.
- 2) As a part of this Inheritance implementation parent and child classes are to be implemented.
- 3) All the data members and methods of the parent class are accessible by child class this is called as reusability that means the parent class member are reusable in the child class.
- 4) The parent class can also be called as base or super class
- 5) The child class can also be called as derived or sub class

Class A

{

—

}

class B : A → Base | super | parent.

{

- Derived | sub | child

}

Classification of Inheritance supported by C++

- 1) Implementation Inheritance
- 2) Interface Inheritance

- 1) Impli

## 1) Implementation Inheritance :-

- 1) This is commonly used Inheritance
- 2) whenever a class is derived from another class it can be called as "Implementation Inheritance"

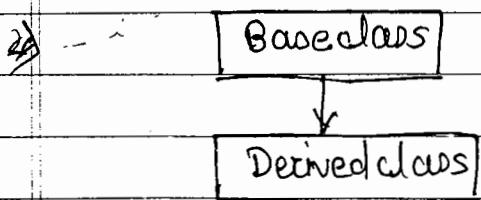
## 2) Interface Inheritance

- 1) This type of Inheritance is taken from "JAVA"
- 2) Whenever a class is derived from an interface it can be called as "interface inheritance"
- 3) The interface is similar to the class but does not contain the method definition. It contains the method declarations only.

## Types of Inheritance

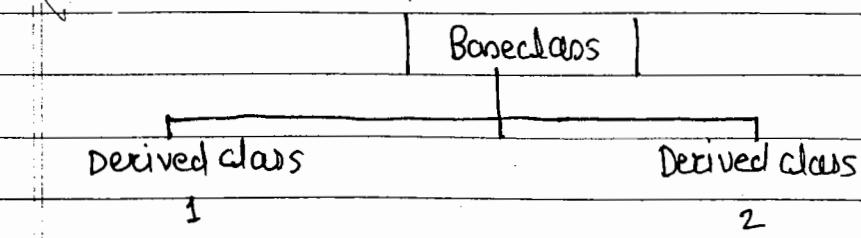
### 1) Single Inheritance

- 1) This is the simplest type of Inheritance
- 2) An Inheritance relationship with only one super class and 1 sub class.



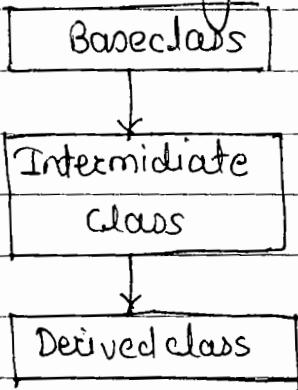
### 2) Hierarchical Inheritance

- An Inheritance relationship with only 1 super class and multiple subclasses .



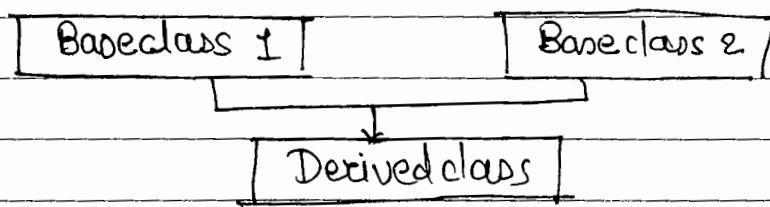
### 3) Multilevel Inheritance :-

- 1) An Inheritance relationship with only 1 super class and multiple sub classes and also extends with another sub class from the 1st subclass.
- 2) In other words 1 class act as super class of subclass simultaneously.



### 4) Multiple Inheritance

- 1) An Inheritance relationship with only multiple super classes and only 1 subclass.
- 2) Multiple Implementation inheritance is not supported by c++ but multiple Interface Inheritance is supported by c#.

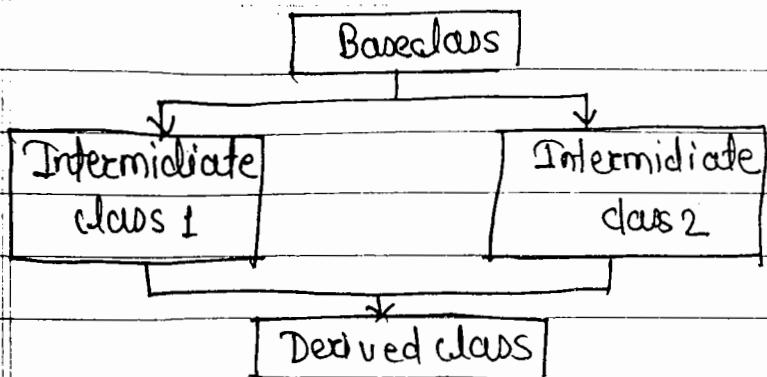


### 5) Hybrid Inheritance

An Inheritance relationship that contains a combination of any other two types of Inheritance.

Ex - Multiple + hierarchical Inheritance

Multiple + multilevel Inheritance



Open visual studio → select template →  
class library →

namespace ClassLibrary1

{

public class Employee

{

int empno;

string ename, job;

public int Empno

{

get { return empno; }

set { empno = value; }

}

public string Ename

{

get { return ename; }

set { ename = value; }

}

public string Job

{

```
get { return job; }
```

```
set { job = value; }
```

```
}
```

```
}
```

```
public class FulltimeEmployee : Employee
```

```
{
```

```
public int salary
```

```
{
```

```
get
```

```
{
```

```
return calcSal();
```

```
}
```

```
}
```

```
public int calcSal()
```

```
{
```

```
return 5000;
```

```
}
```

```
}
```

```
public class ParttimeEmployee : Employee
```

```
{
```

```
public int sal
```

```
{
```

```
get
```

```
{
```

```
return partSal();
```

```
}
```

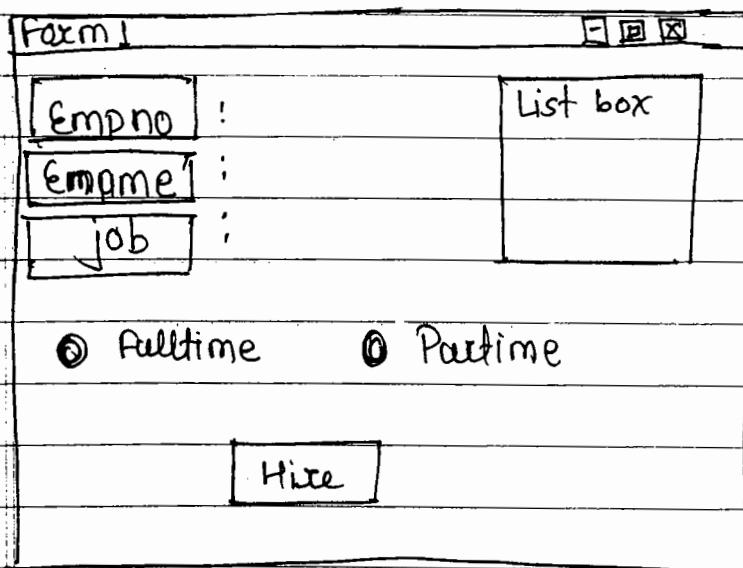
~~10/13~~ 3  
public int dotsal()

{  
    return 3000;  
}

Go to Build → Build class library →  
DLL file created successfully.

Open visual studio → select template →  
windows forms application. → go to  
solution explorer → right click on references  
→ Add reference → add dll file

Design the form.



Using

Using Class Library

Private

{  
if (radioButton1.Checked )

{  
ListBox1.Items.Clear();

FulltimeEmployee obj = new FulltimeEmployee

obj.Empno = int.Parse(textBox1.Text);

obj.Ename = textBox2.Text;

obj.Job = textBox3.Text;

obj.ListBox1.Items.Add(obj.Empno);

ListBox1.Items.Add(obj.Ename);

ListBox1.Items.Add(obj.Job);

ListBox1.Items.Add(obj.Salary);

}

}  
else if (radioButton2.Checked )

{

ListBox1.Items.Clear();

ParttimeEmployee obj = new ParttimeEmployee

obj.Empno = int.Parse(textBox1.Text);

obj.Ename = textBox2.Text;

obj.Job = textBox3.Text;

ListBox1.Items.Add(obj.Empno);

ListBox1.Items.Add(obj.Ename);

ListBox1.Items.Add(obj.Job);

Sal);

Named Parameters :-

1) This feature is introduced in C# .NET 4.0

2) This is used to pass the arguments to the method with those names.

Open visual studio → goto console application

namespace ConsoleApplication1

{

public class users

{

public string fname;

public string lname;

public void setUsername (string fname,  
string lastname)

{

fname = fname;

lname = lastname;

}

public void display ()

{

Console.WriteLine (fname + " " + lname);

}

class Program.

{

static void main (string [] args)

{

users obj = new users();

obj.setUsername (lastname : "Kumar",  
firstname : "Raj");

```
obj. display ();  
Console . Read ();
```

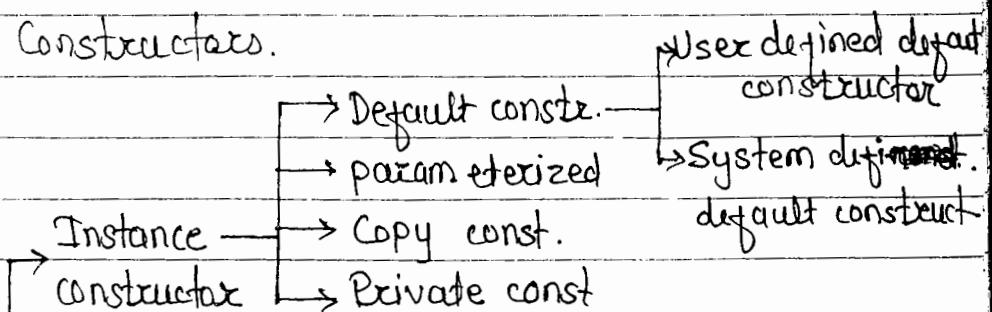
Result - Raj kumar.

## Constructors and Destructors

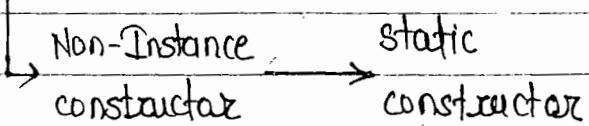
- Properties and Features

  - 1) Constructor is a member<sup>method</sup> of a class which is invoked automatically when an object to the class is created.
  - 2) A constructor name should be same name as class name.
  - 3) A constructor does not have any return type even void also.
  - 4) A constructor is used to keep something ready for object when it is created.

## Types of Constructors.



## Constructors:-



current class = this.

### 1 User Defined Default Constructor.

- 1) This constructor is created by the programmer
- 2) This constructor does not accept any argument or parameters.
- 3) In general this constructor is used to initialize required values into the data fields
- 4) But there is no restriction to write particular code into the constructor.

Go to windows forms application

Go to solution explorer → right click on the application → add new item → select template class file → name it as employee.cs → click add button.

namespace WindowsFormsApplication1

{  
public class Employee

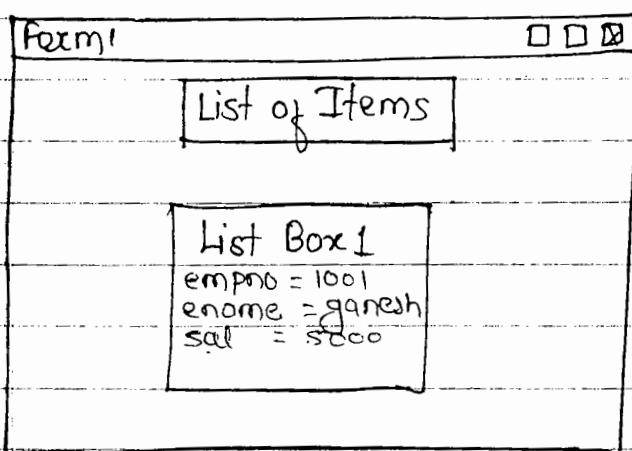
{  
public int empno, age, sal;  
public string ename, job, addr;

Public Employee()

{  
this.empno = 1001;  
this.ename = "Ganesh";  
this.Sal = 5000;  
this.age = 28;  
this.job = "Manager";  
this.addr = "Hyd";

}

Design a form.



Private void button 1

```
{ Employee obj = new Employee ();  
listBox1.Items.Add ("Empno :" + obj.empno);  
("Ename :" + obj.ename);  
("Salary :" + obj.sal);  
("Job :" + obj.job);  
("Address :" + obj.addr);  
("Age :" + obj.age);
```

System Defined Default Constructor

- 1) If there is no constructor defined within a class then system will create its own constructor and will assign default values into the class fields.
- 2) This constructor created by the system is known as system defined Default constructor.
- 3) When an object of the class is created first system will search for a constructor within the class. If there is no constructor available within the class system will create its own constructor & will assign default values into class fields.

- 1) In the above example any no of objects we may create for a class all the objects will stored same data. In their referenced data field.
- 2) To overcome this drawback we use parameterised constructor.

Parameterized Constructor :-

- 1) Parameterized Constructor accepts arguments.
- 2) It stored the values into the data fields.
- 3) Using parameterized constructor we can stored different set of values into different object created by the class.

To do windows forms app → go to solution explorer → right click on app → add new item → select template class file → name it as Test.cs → click add button

namespace WindowsFormsApplication2

{

Class Test

{

int a, b;

public Test (int x, int y)

{

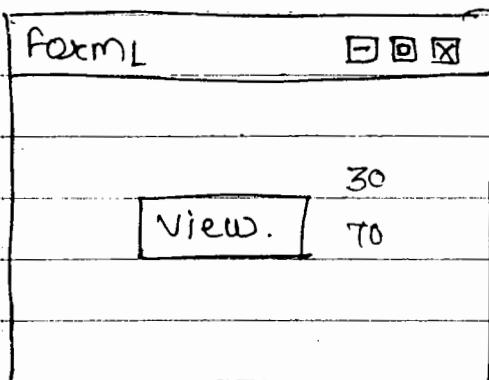
a = x;

b = y;

}

```
public int add ()  
{  
    return a+b;  
}
```

Design a form.



double click on view

Private void button ..

{

```
Test obj1 = new Test (10,20);  
messageBox . show (obj1 . add () . To string ());
```

```
Test obj2 = new Test (30,40);
```

```
messageBox . show (obj2 . add () . To string ());
```

}

Static Const



## Static Constructor :-

- 1) Static constructor is used to initialize static database field
- 2) static datafields will have maintain only one instance for any number of object created to the class to make any no. of a class static we use static keyword.
- 3) static members are not accessible with object rather we need to access with class name only.
- 4) There can be only 1 static constructor in the class.
- 5) The static constructor should be without parameter.
- 6) It can only access the static members of class there should be no access modifier in static constructor defined.
- 7) If a class is static then we can not create object for static class.

## Syntax :-

static classname()

{

}

→ Visual studio to console application. → ↗

namespace ConsoleApplication2

{

```
public static string name;  
public static string addr;  
static college()
```

```
{  
    name = "ganesh";  
    addr = "hyd";  
}
```

## Class Program

```
{  
    static void Main (string [] args)  
{
```

```
        Console.WriteLine (college.name);  
        college.addr);  
        .Read();  
    }
```

```
}  
Ganesh  
Hyd
```

## Sealed classes

- 1) The sealed classes can be declared using Sealed keyword. as follows

```
Sealed class classname
```

```
{  
    // some members  
}
```

- 2) The sealed classes are just like normal classes but u can not inherit the sealed class in other words u can not create a subclass ~~for~~ of the sealed class.

- 3) If u try to inherit the sealed class will receive a compilation error

4) This is just like "final classes" in java.

Sealed class one

{

}

class Two : one // can't inherit this will give a  
compilation error.

{

}

Object Initializer

This concept is introduced in .net framework 3.0  
this is used to initialized all the fields of an  
object without using a constructor.

Syntax :-

classname objectname = new classname() { field1=val1,  
field2=val2,  
..... }

Go to console application.

namespace Console App

{

public class Dept

{

public int dno;

public string dname, loc;

public void display()

{

Console.WriteLine ("Deptno : " + dno);

Console.WriteLine ("Dname : " + dname);

Console.WriteLine ("Location : " + Loc);

}

{

class Program

{

static void Main (string [] args)

{

Dept obj = new Dept () {dno = 10, dname = "Sale",  
Loc = "hyd"};

obj.Display ();

Console.Read ();

{

{

Result :- dno = 10

dname = sale.

Loc = hyd.

{

Destructor :-

- 1) It is also a method of a class which is having some special feature just like constructor.
- 2) But it can be called automatically by the compiler at "Object destruction time"
- 3) Object destruction time means the time of clearing of memory i.e. allocated further object
- 4) The destructor is used to implement any

process ie do be performed at the time of object closing

Rules for destructor.

- 1) Its name must be defined as class name
- 2) Its name must be started with "~" (tilde) character.
- 3) Access modifier can not be specified. for this
- 4) No argument
- 5) No return value.
- 6) Destructor overloading is not possible. that means multiple destructor can not be defined inside of a class

Destructor Implementation.

`~classname ()`

```
{  
    .....;  
    .....;  
    .....;  
}
```

Demo on Constructors & Destructor.

namespace Console Application.

```
{  
    public class Student  
    {  
        public int sno;  
        public string sname;  
    }
```

public void Display()

{

Console.WriteLine ( sno + " " + sname );

{

public Student ()

{

sno = 1001 ;

sname = " deja" ;

{

public Student ( int stno, string stuname )

{

sno = stuno ;

sname = stuname ;

{

~ Student ()

{

Console.WriteLine ( " This is Destructor . . . " );

{

{

Class Program

{

static void Main ( string [ ] args )

{

Result. {  
    1001, dej  
    1002, ganes  
    This is destructor ... (2)

```
Student obj1 = new Student();  
obj1.Display();  
Student obj2 = new Student(1002, "ganes");  
obj2.Display();  
Console.Read();
```

## Abstract Classes

- 1) A project must start either with interfaces or abstract classes. To maintain standards in a project we need abstract classes.
- 2) Abstract is a keyword used at method level & class level.
- 3) To make any function as abstract use abstract keyword.
- 4) An abstract function should be eliminated ( ; )
- 5) Over writing of an abstract function is compulsory.
- 6) An abstract method has no implementation in the class or derived.
- 7) A derived class must implement all abstract methods of abstract class do make any class as abstract use abstract keyword.
- 8) An abstract class can not be instantiated directly.
- 9) An abstract class can contain non abstract function.
- 10) An abstract class can contain all members of a class.
- 11) By default abstract class function are not treated as public and abstract.

## Implementation Syntax

abstract class abstractclassname

{

// data members if any

// non - abstract methods if any

access modifier abstract returntype

methodname ( arguments );

}

class derivedclassname :

{

abstractclassname

{

access modifier override returntype

methodname ( arguments )

// method body

{

}

Rough

public ~~abstract~~ class Sample

{

public abstract void add();

public void m1()

{

====

public class C : Sample

{

override void add()

{

public class B : Sample

{

public override void add()

{

---

Go to class library.

namespace ClassLibrary1

{

public abstract class Employee

}

int empno, sal;

string ename;

public int Empno

{

get { return empno; }

set { empno = value; }

}

public int sal

{

get { return sal; }

set { sal = value; }

}

public string ename

{

get { return ename; }

set { ename = value; }

}

public abstract void add();  
}

public class FulltimeEmployee : Employee  
{

    public override void add()  
    {

        sal = 5000;  
    }

}

public class ParttimeEmployee : Employee  
{

    public override void add()  
    {

        sal = 3000;  
    }

}

5/10/13

## Interfaces

- 1) If a class contains all abstract functions then it is known as an interface.
- 2) Interfaces don't provide implementation they are implemented by classes.
- 3) To create an interface use interface keyword.
- 4) An interface can contain
  - 1) abstract function.
  - 2) Properties
  - 3) Events
  - 4) Indexes.
- 5) An interface can't contain non-abstract function, database, constructor, destructor
- 6) By default interface function are declared as public or abstract.
- 7) An interface can't be instantiated directly.
- 8) Creating a new class from an interface is compulsory or mandatory in order to provide implementation to its abstract function.
- 9) An interface can inherit from multiple interfaces.
- 10) An interface is used to implement multiple inheritance in C# .net.

Class A	Interface A	public class C : B, A
{ void add() { } }	{ void add(); { } }	{ =}
Class B	Interface B	cobj = new C();
{ void add() { } }	{ void add(); { } }	obj.add();

```
class C : A, B  
{  
    void A.add()  
}  
}  
  
void B.add()  
{  
}
```

C# does not support  
multiple inheritance.

```
Cobj = new C();  
obj.A.add();  
obj.B.add();
```

## Implementation Syntax of Interfaces.

```
// interface definition  
interface interfaceName  
{  
    returnType methodName ( arguments );  
}
```

```
// implementation class definition  
class className : interfaceName  
{
```

```
    public returnType methodName ( argument )  
    {  
        // Some code  
    }
```

Example: Go to console application  
namespace ConsoleApplication

```
{ interface A  
{  
    void add();  
}
```

```
class sample : A
{
    public void add()
    {
        Console.WriteLine("This is Interface method");
    }
}

class Program
{
    Sample obj = new Sample();
    obj.add();
    Console.Read();
}
```

Example 2:- Go to Console Application.

```
namespace ConsoleApplication
```

```
{
```

```
interface A
{
    void m1();
}
```

```
interface B : A
{
    void m2();
}
```

```
class Sample : B
{
```

```
    public void m1()
{
```

```
        Console.WriteLine("This is method of interface A");
    }
```

```
    public void m2()
{
```

```
Console.WriteLine ("This is method of interface B");
}
}

class Program
{
    static void main (String [] args)
    {
        Sample obj = new Sample ();
        obj.m2 ();
        Console.Read ();
    }
}
```

Example 3 :- Code

16/10

```
namespace Console Application1
```

```
{  
    Interface A  
}
```

```
    void m1 ();  
}
```

```
    Interface B  
}
```

```
    void m1 ();  
}
```

```
Class Sample : A, B
```

```
{
```

```
    void A.m1 ()  
    {  
        Console.WriteLine ("welcome");  
    }
```

```
{
```

```
    void B.m1 ()  
    {
```

```
        Console.WriteLine ("Hai");  
    }
```

```
}
```

); Class Program

```
{ static void main (string [] args)
```

```
{ Sample obj = new sample ();
```

```
    A obj1 = obj;
```

```
    obj1 . m1 ();
```

```
    B obj2 = obj;
```

```
    obj2 . m1 ();
```

```
    Console . read ();
```

16/10/13

Windows form app? → solution explorer → right click  
on application → add new item → select template  
class file → Text.cs (name as) → add.

Code:-

```
namespace windows . . .
```

```
{
```

```
    interface Addition
```

```
{
```

```
        int add ();
```

```
}
```

```
    interface Multiplication
```

```
{
```

```
        int mul ();
```

```
}
```

```
    public class Test : Addition , Multiplication .
```

```
{
```

```
        int a , b ;
```

```
        public Test (int x , int y)
```

```
{
```

```
            a = x ;
```

```
            b = y ;
```

```
}
```

```
Public int add ()
```

```
{  
    return a+b;  
}
```

```
Public int mul ()
```

```
{  
    return a*b;  
}
```

```
}
```

Form



View

Code :-

```
Private button1
```

```
{
```

```
Test dt1 = new Test (10,20);
```

```
Addition obj1 = dt1;
```

```
Message Box . show (obj1 . add () . ToString ());
```

```
Multiplication obj2 = dt1;
```

```
Message Box . show (obj2 . mul () . ToString ());
```

Difference between abstract classes & interfaces

Abstract Classes	Interfaces
1) A class which contains one or more abstract function is known as an abstract class.	1. A class which contains all abstract function is known as interface.
2) An abstract class can contain non-abstract functions.	2. An Interface can't contain non abstract functions.
3) It <sup>can</sup> contain all member of class	3. It can contain only abstract function, properties, Index, events & can't contain non abstract function, data field, constructor, destructor.
4. Use abstract keyword to create abstract class.	4. Use interface keyword to create an interface.
5. An abstract class can't be used to implement multiple inheritance.	5. An interface can be used to implement multiple inheritance.
6. By Default abstract class members are not public & not abstract.	6. By default interface members are public & abstract.
7. An abstract class can't be instantiated directly.	7. It can't be <del>not</del> instantiated directly.

8 Creating a new class from abstract class is must consume it.

8. Creating a new class from an interface is must to consume it.

## Polymorphism

It is a concept of showing an ability to multiple forms in polymorphism we can overload function & operators.

It is classified into

- 1) Design type binding
- 2) Runtime binding.

1) Design type binding

function over loading

operator over loading

2) function binding.

### Function / Method Overloading

1) Method overloading is nothing but to writing multiple methods with same name & with the different arguments.

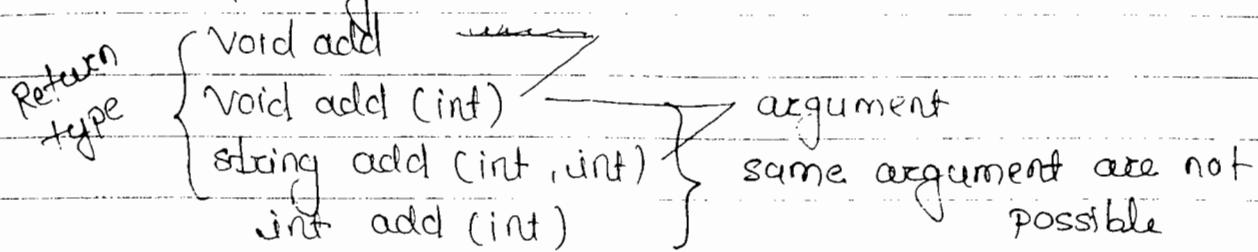
2) To overload methods you should follow the below rules.

3) To overloaded methods you should follow it

4) All of the overloaded method name should be so

5) Any difference between the method arguments should be some maintain the difference may be in member of argument or the data types of argument.

from F to F iii) finally when you call the overloaded method automatically compiler takes decision that "which method is to be called" based on the arguments that you are passing.



example:- Go to windows form → solution exp. right click → add new item → class file → name it as sample sample.cs → add.

code:- using system.windows.forms,  
name space :-

```
{  
    class sample  
    {  
        public void add ()  
        {  
            MessageBox.show ("Default");  
        }  
        public void add (int a)  
        {  
            MessageBox.Show ("One Argument");  
        }  
        public void add (int a, int b)  
        {  
            MessageBox.show ("Two Argument");  
        }  
    }  
}
```

be so  
would  
see a

Code :-

```
private ---  
{  
    Sample obj = new Sample();  
    obj.add(10, 20);  
    obj.add();  
    obj.add(10);  
}  
}
```

Form1



View

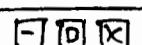
### Virtual function

When we are implementing two same functions in the base class as well as in the derived classes we are getting virtual function concepts as per C#.net we are implementing virtual function. We use `virtual` & `override` keyword.

Example:- Go to windows form application  $\rightarrow$  soln exp  $\rightarrow$  right click  $\rightarrow$  add new item  $\rightarrow$  select class  $\rightarrow$  name it as Dept.cs  
code :- namespace windows form application.  $\rightarrow$  add.

```
{  
    public class Dept1 : System.Windows.Forms.MessageBoxBase  
    {  
        public int sal;  
        public virtual void add()  
    }
```

Form1



View

```
    {  
        sal = 5000;  
    }  
}
```

private ..

```
public class Dept2 : Dept1 {
```

```
    Dept2 obj = new Dept2();
```

```
    public override void add() {  
        obj.add();  
    }
```

```
    base.add();  
}
```

```
    int bonus = 500;  
    ...  
}
```

11/10/18

## ✓ Difference between Overloading and overriding

### Function Overloading

→ Providing new implementation to a function with same name & different signature is known as function overloading.

### Function overriding

→ Providing new implementation to a function with same name & same signature is known as function overriding.

2) Function overloading will be done in same class

2) Function overriding will be done in the base & derived classes

3) This is code refinement technique. 3) This is code reusability tech

No separate keyword are use do implement function overloading.

Use virtual keyword for base class function & override keyword in derive class function to implement fun overriding.

## \* Structures

1) This concept is not new in C# it is taken from C language.

2) In C languages structure u can write only some member variable called as data members.

But in C# structures u can write fields along with some method also. so in C# structures are almost or similar to the classes but having some diff with the classes

Classes	Structure
1) Declared with "class" keyword	1) Declared with "struct" keyword
2) It can contain fields, methods, constructors, destructor, properties etc.	It can contain fields, methods, constructor, destructor, properties etc.
3) It supports Inheritance	3) It does not support Inheritance
4) User defined default constructors can be implemented.	4) User defined default constructors can not be implemented
5) Data members can be initialized in the class definition.	5) Data members can not be initialized in structure definition.

Implementation.  
Defined structure :-

Struct structurename

{

// fields

// Method.

}

Create Instance for structure.

Structurename Instancename = new structurename();

Open visual studio, select template console application.

namespace ConsoleApplication2

{

struct Employee

{

public int Empno;

public string Ename;

public double sal, dae, nesal;

public void Readdata()

{

Console.WriteLine("Enter Employee Details");

Empno = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Employee name:");

Ename = Console.ReadLine();

Console.WriteLine("Enter Employee salary:");

sal = int.Parse(Console.ReadLine());

}

public void calculate()

{

dae = sal \* 10 / 100

nesal = sal - dae;

}

```
public void Display ()  
{
```

```
    Console.WriteLine ("Empno :" + Empno);  
    ("Ename :" + Ename);  
    ("Salary :" + Sal);  
    ("Tax :" + Tax);  
    ("Netsalary :" + NetSal);
```

```
}
```

```
}
```

class Program .

```
{
```

```
    static void main
```

```
{
```

```
    Employee obj = new Employee();  
    obj . Readdata ();  
    obj . calculate ();  
    obj . Display ();  
    Console . Read ();
```

Partial Class Result

Enter Empno value.

Enter Ename .

E

Enter Salary .

Empno =

Ename =

Sal =

## Partial Classes :-

- 1) The "partial" keyword allows you to define a class, struct or interface across multiple source files.
- 2) Each class will be placed in individual files, but in some situations you may require to define a class within multiple files in other words the class defn can be split into multiple files.
- 3) Partial classes will provide flexibility in application development.
- 4) When working on large projects splitting spreading a class over separate files allows multiple programmers to work on it simultaneously.
- 5) In Windows applications the automatic code generator generates some design path code for your own form class in this case also partial classes are required.

Ex: Go to Console Application.

namespace ConsoleApplic 3

```
{  
    public partial class sample  
    {  
        public void m1()  
        {  
            console.WriteLine("First method");  
        }  
    }  
}
```

console.WriteLine("First method");

public partial class sample

```
{  
    public void m1()  
}
```

Console.WriteLine ("Second method");

```
}  
}
```

class program

```
{
```

Sample.static void main (string [] args)

```
{
```

```
    Sample obj = new Sample();
```

```
    obj.m1();
```

```
    obj.m2();
```

```
    Console.Read();
```

Result.

First Method

Second Method.

Hiding the Super class method.

1) A super class & subclass having 2 method with same name & same argument.

2) If u call method from client mode which method would be called ? (either super class method or subclass method)  
ans subclass method only

- 3) this is also called as "hiding method")  
here the super class method is hidden  
by subclass method. But a compil. time warning  
will be displayed in the subclass method defn  
do we a keyword called "new"  
4) The "new" key word hides the super class  
method that contain same name & same argument  
etc. Ofcourse this is done by default  
but to implement a strong programming  
"new" keyword is recommended to be  
used and it when u want to hide the  
super class method with intension.

Ex Hiding Methods with new keywords.  
Go to Console application.

namespace Console Application

{

public class US

{

public void showCountry()

{

Console.WriteLine("This is united state");

}

public class India : US

{

public new void showCountry()

{  
    Console.WriteLine ("This is India");  
}  
}

class Program.

{  
 static void main (string  
 )  
}

India obj = new India();  
obj.showcountry();  
Console.Read();

18/10/13  
Friday

Drop Database databasename

## DataBase

A data base is a large data storage mechanism which can manage more efficient & effective fashion to provide with highly coupled security.

### DBMS (Data Base management System)

- 1) It says how to manage the data.
- 2) Top database servers are
  - i) Oracle
  - ii) SQL Server
  - iii) DB2
  - iv) SYBASE
  - v) MySQL etc
- 3) SQL server is a microsoft own server under windows operating system only.
- 4) SQL server versions are
  - i) SQL Server 2000
  - SQL Server 2005
  - SQL Server 2008
  - SQL Server 2012
- 5) SQL server is having collection of data bases & every base is having collection of object like table, views, stored procedure, triggers etc.

Connect to SQL Server 2012

Go to start → programs → go to microsoft sql server 2012 → go to sql server management studio → login ID sa → password 123  
click connect button

go to new query.

Create Database sample db  
use sampledb1

Create table emp (empno int, ename varchar(30)  
sal int)

Insert into emp values (1001, 'Ganesh', 8000)

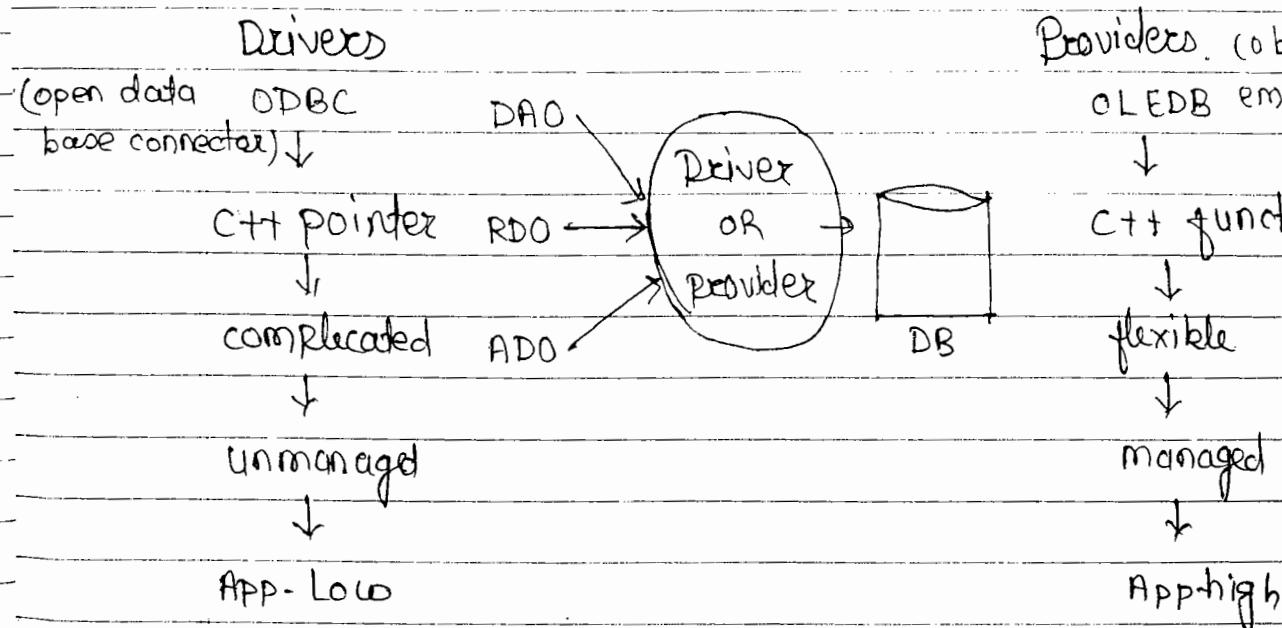
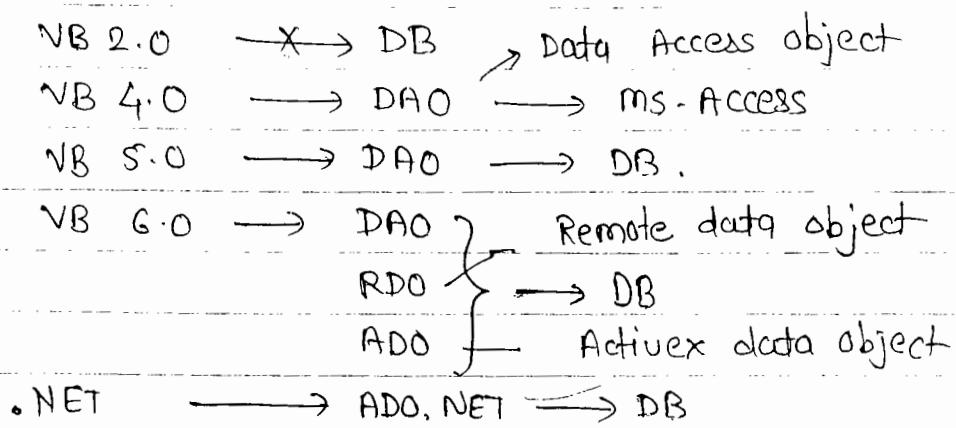
SELECT \* FROM emp

Output -    empno    ename    sal  
              1001      Ganesh     8000

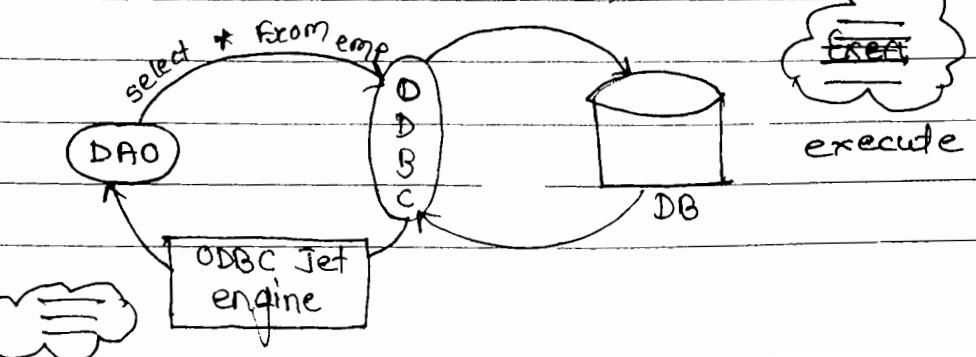
(Open c  
base)

# ADO.NET

- 1) ADO.NET is one of the Microsoft technologies.
- 2) Using ADO.NET predefined classes we can perform the operation with database server.
- 3) The ADO.NET is specially designed for database purpose.

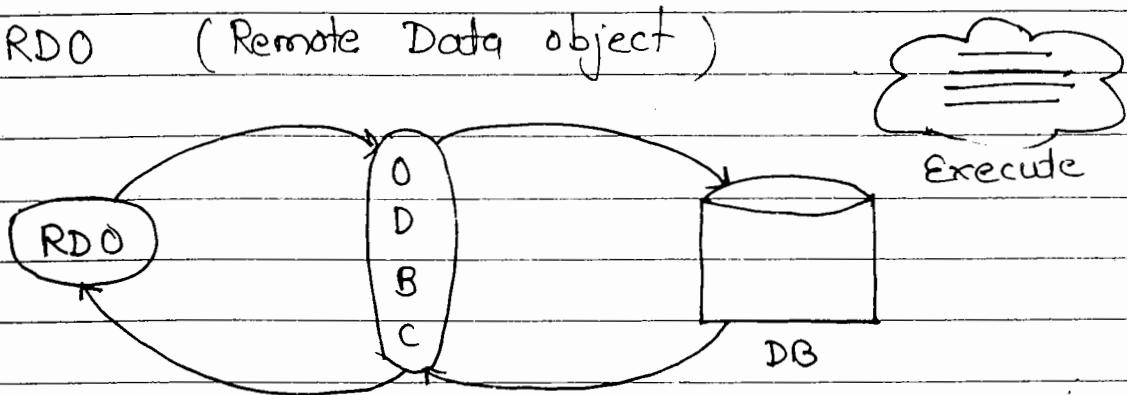


## DAO (Data Access Object)



- 1) In DAO one sql statement execute in the dB as well as ODBC jet engine so application performance is very low
- 2) It does not supports providers.

### RDO ( Remote Data object )



In RDO's the sql statement execute only data base so application performance is high

### Disadv

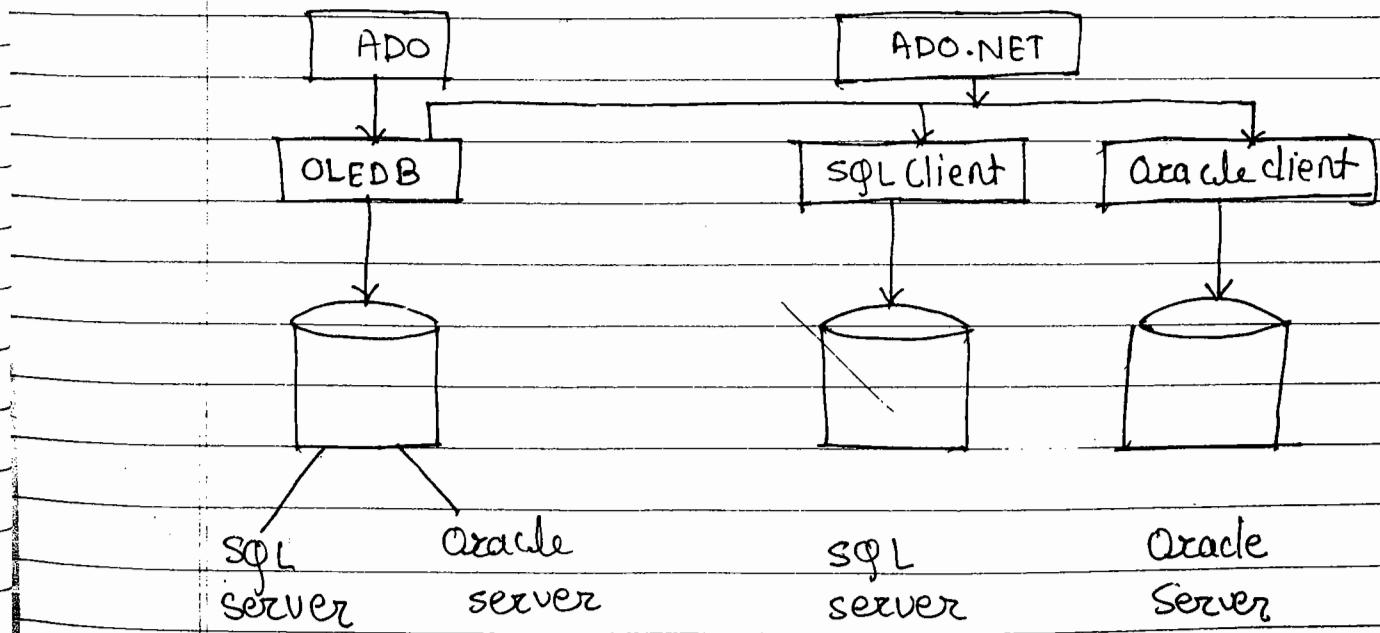
It does not support providers.

### ADO ( Activex

ADO support drivers as well as providers  
so application performance is very high .

### Differences between ADO & ADO.NET

ADO	ADO.NET
1) ADO's implemented in com technology ADO.NET	1) ADO.NET is implemented in .net framework technol
2) It is connection oriented database management system.	2) It is a disconnected oriented database management system
3) In ADO we can stored only one table in front end using "recordset"	3) In ADO.NET we can stored multiple tables in front end using "dataset".
4) It does not support XML integration.	4) It supports XML integration.
5) It does not support multiple transaction.	5) It supports multiple transaction.
6) It can not recommended "Data view".	6) Its recommended "Data view".



## OLEDB Providers :-

For various databases we have respect to provider.

DataBase	Provider	Released By
SQL Server	SQL OLEDB	microsoft corporation.
ORACLE	MSDAORA (microsoft data access oracle)	microsoft corporation
oracledb. oracle	oracledb.oracle	
MS ACCESS / MS-EXEL / FOX PRO dbase	microsoft.jet. OLEDB.4.0	microsoft corporation
MS-ACCESS 2007 or 2000	microsoft.ACE. OLEDB.12.0	microsoft corporation

## Key point

ADO.NET is known as database technology which is use to connect with the databases, that means some object will work for interacting with the databases.

- 2) Basically why we required database connection is : The front end appl' itself cannot stored any data permanently so that we required a storage mechanism. that storage mechanism is nothing but our data bases.

- 3) It can be used for database connection & offers to perform database manipulation like inserting data to the table, deleting unnecessary data, retrieving the required data from table etc.
- 4) It can be used in any type of applications like console application, windows form application, website, web services, WCF services etc.
- 5) It can be used in any .NET language like C#.NET, VB.NET, C++ .NET etc.
- 6) It was developed based on its previous version called "ADO"

\* What type of databases we can connect using ADO.NET.

File Data Base

DBase FOXPRO, MS-ACCESS, MS-EXEL etc

Server Data Base

ORACLE, SQL SERVER, MySQL etc

### System.Data

System.Data.OLEDB	System.Data.SqlClient	System.Data.oracleclient
-------------------	-----------------------	--------------------------

ADO.NET provides us various classes which can be used for communicating with the data sources under the following namespaces.

- 1) System.Data.
- 2) System.Data.OLEDB
- 3) System.Data.SqlClient
- 4) System.Data.oracleclient

## 1) System.Data

If it is a collection of classes which are used for holding and managing the data on client machine.

classes.

- i) Data set
- ii) Data table
- iii) Data view
- iv) Data Row
- v) Data column.
- vi) Data Relation.

## 2) System.Data.OLEDB

If it is collection of classes used for communicating with any data source.

classes

- ij) OLEDB connection
- ii) OLEDB command.
- iii) OLEDB Data Adapter
- iv) OLEDB Data Reader
- v) OLEDB parameter

## 3) System.Data.SqlClient

If it is collection of classes used for communicating only with SQL server DB classes

- ij) SQL connection
- ii) SQL command
- iii) SQL Data Adapter

- iv) SQL data Reader
- v) SQL parameter.

#### 4) System.Data.Oracle client

It is collection of classes used for communicating only with oracle dB

##### classes

- i) Oracle connection
- ii) oracle command
- iii) Oracle Data Adapter
- iv) Oracle Data Reader
- v) Oracle parameter

Every database operation we are going to perform mainly involves in 3 activity those are

- 1) Establish a connection
- 2) Applying a statement
- 3) Expecting the results

##### 1) Establishing a connection

In this process we open a channel for communication with target system to perform over operation.

To open the channel for communication ADO.net provides us a class "connection".

##### Syntax

Connection (String ConnectionString)

## Understanding the connection string.

The connection string provides details about the connection string. That means if u want to connect with the database u have to specify some details about the connection like

- 1) Server
- 2) User ID
- 3) Password
- 4) provider

### 1) Server

Specify the name of the server system which you want to connect. If u want to connect with other server system on the network use specify the name of that system.

### 2) User ID

Specify the username for login with the data base

### 3) Password.

Specify the password for login with the data base

### 4) Provider

Specify the name of provider which you want to use with the connection.

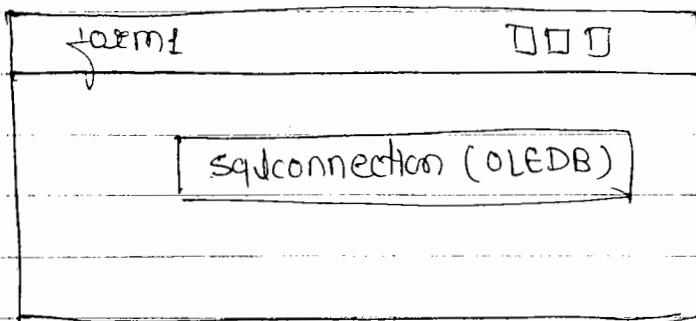
Syntax of connection string.

```
"Provider = ..... ; UserId = ..... ;  
          Pswd = ..... ;  
          Server = ..... "
```

Note :-

Just for separation for individual values we  
are using (1) "semicolon"

- \* SQL Server connection through OLEDB  
Design a button.



using System.Data.OleDb;

```
private void button1
```

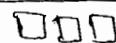
```
{
```

```
    OleDbConnection cn = new OleDbConnection ("provider  
        = sqloledb; user id = sa; password = 123; server = nit");  
    cn.Open();  
    MessageBox.Show ("connection established");
```

```
}
```

## \* SQL Server connection through SQL client

Form2



sqlconnection(sqlclient)

Namespace

Using System.Data.SqlClient;

private void button1

{

sqlconnection cn = new SqlConnection ("uid = sq ;  
password = 123 ; server = nrt - pc ");

cn.Open();

message Box. Show("Connection established");

}

## 2) Applying a statement

In this process we send an instruction to the data source specifying the type of activity which has to be performed in the form of an SQL statement like select, Insert, update, delete.

To apply a statement we can make use of command class

Syntax

Command (String statement, Can obj)

- 1) After connecting data base u can send command to the data base.
- 2) ADO.net support do send the following types of command.
  - 1) Insertion command.
  - 2) Deletion co
  - 3) Updation
  - 4) Select
  - 5) Stared procedure

#### { Insertion command :-

To insert a new row into the table.

SQL statement.

insert into TableName values( val1, val2, ... )

#### 2) Deletion command :-

To delete one or more rows from the table  
SQL statement .

Delete from TableName where condition

#### 3) Updation command

To modify the table data

SQL statement

Update TableName Set col1=val1 ,

col2=val2

----- where condition .

#### 4) Select Command :-

To retrieve the data from data base table into front end application

SQL statement

SELECT \* FROM Table Name

5) Stored procedure :-

To call a stored procedure or fn from the front end app ie already created at backend.

Statement - No SQL stat. is needed.

\* Insert, Update, Delete statement :-

Queries are two type

- 1) Action query.
- 2) Nonaction query.

1) Action queries

Insert, Update, Delete

2) Nonaction queries.

Select

→ ExecuteNonQuery() } command.

→ Execute Reader()

Example :- Design a form

Form 1	Back
<input type="button" value="Insert"/>	

Namespace.

using System.Data.SqlClient;

private void

```
{  
    SqlConnection cn = new SqlConnection("uid = sa ;  
    database = sampledb1 ; password = 123 ; server = nit-pc");  
    cn.Open();  
    SqlCommand cmd = new SqlCommand("insert into  
        emp values(sss, 'balaji', 3000)", cn);  
    int i = cmd.ExecuteNonQuery();  
    MessageBox.Show(i + " record added");  
}
```

Ex-2 Design a form

The diagram shows a window titled "Form1". Inside the window, there are three text input fields with labels: "Empno", "Ename", and "salary", each followed by a rectangular text box. Below these fields is a single rectangular button labeled "Insert".

using System.Data.SqlClient;

private void

```
{  
    SqlConnection cn = new SqlConnection("uid = sa ; database  
    = sampledb1 ; password = 123 ; server = nit-pc");  
    cn.Open();  
}
```

```

SqlCommand cmd = new SqlCommand
{
    "insert into emp values (" + textBox1.Text +
    " " + textBox2.Text + ", " + textBox3.Text + ")",
    int j = cmd.ExecuteNonQuery();
    MessageBox.Show(j + " record added");
}

```

Example

Example Statement

```

update emp set
ename='Rajesh',
sal=1000 where empno=1001

```

Design a form.

Form1	
Empno:	<input type="text"/>
Ename:	<input type="text"/>
Salary:	<input type="text"/>
<input type="button" value="Update"/>	

Code

```

Code:- Using System.Data.SqlClient;
private void

```

```

{
    SqlConnection cn = new SqlConnection
    ("uid = sa; database = sampledb1; password = 123;
    server = \"nit-pc\");
    cn.Open();

```

SqlCommand cmd = new SqlCommand

```

("update emp set ename = " + textBox2.Text +
    sal = " + textBox3.Text + " where empno = " + textBox1.Text +
    +"", cn);

```

```

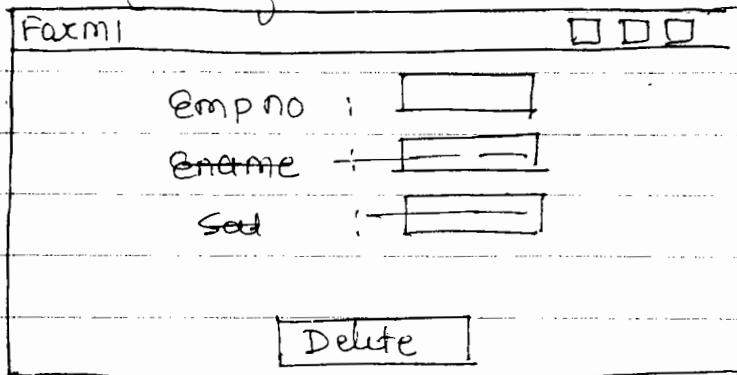
        int i = cmd.ExecuteNonQuery()
        MessageBox.Show("Data updated");
    }
}

```

Example : Statement.

delete from emp where empno = 1001

Design a form



Code:- Using System.Data.SqlClient;

private void button1\_Click

```

{
    SqlConnection cn = new SqlConnection
    ("uid = sa; database = sampledb1; password=123;
    server = unit - pc");
    cn.Open();
    SqlCommand cmd = new SqlCommand
    ("Delete from emp where empno =
    " + textBox1.Text + " ", cn);
}

```

int i = cmd.ExecuteNonQuery()

MessageBox.Show("Record deleted successfully");

}

## Radio1.checked

### \* Execute Non Query :-

This method is used to execute any SQL statement (Insertion statement or Deletion stat. or Updation stat. or stored procedure)

In other word this method moves the execution flow to backend database execute the command there and them compact with come back with some result.

This method return "No of rows affected" which represents the count of the rows which are affected by executing this command.

Suppose after executing a delete statement two rows are deleted so it returns the integer value 2.

form1	<input type="button" value="F"/> <input type="button" value="R"/> <input type="button" value="B"/>
Cid :	
cName :	
Bal :	
Gender	<input type="radio"/> Male <input type="radio"/> Female
<input type="button" value="Insert"/>	

## Customer

Cid	cName	Bal	Gender
1001	abcd	5000	male

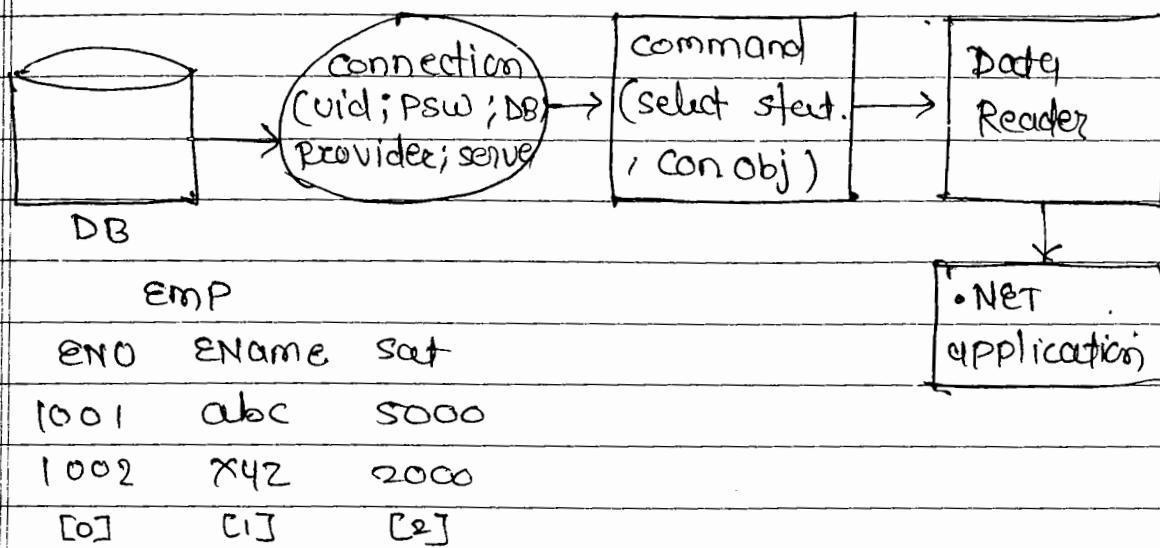
H  
ent

Retrieving data from Database.

Connection oriented arch

Disconnected oriented arch

## 1) Connection Oriented architecture.



- 1) For Retrieving the table data from the database you write a select statement in the front end code.
- 2) Next you need to pass it to back end using command class object.
- 3) Then your select statement will be executed at the backend then the database returns the result data based on the executed query.  
For ex. If your statement is "SELECT \* FROM emp" then the entire employee table data will be returned as it is.
- 4) Then front end you have to receive the data into a temp memory location. (buffer)  
To create a buffer in your code you can use " Data Reader class"

- 5) After that you can present the data on the screen further user.
- 6) This is the flow of data. You can observe this flow in the above diagram.

### Library

- i) Connection :-

Maintaince the connection with database

- ii) Command :-

Sends a SQL statement & executes it at backend.

- iii) Data Reader :-

Acts as a buffers, It holds the data which is received from database.

Design a form

Form1

checkbox

Label

textbox

button

Empno :

Find

Ename :

salary :

Namespace

Using. System. Data. SqlClient;

private void button ..

```
{  
    SqlConnection cn = new SqlConnection("uid = sa;  
    password=123; database = sampledb1; server =  
    nit-pe");  
}
```

```
Cn.Open();  
SqlCommand cmd = new SqlCommand (" select * from  
emp where empno = " + textBox1.Text + " ", Cn );  
SqlDataReader dr = cmd.ExecuteReader ();  
if (dr.Read ())  
{
```

```
    textBox2.Text = dr ["ename"].ToString ();  
    textBox3.Text = dr ["sal"].ToString ();
```

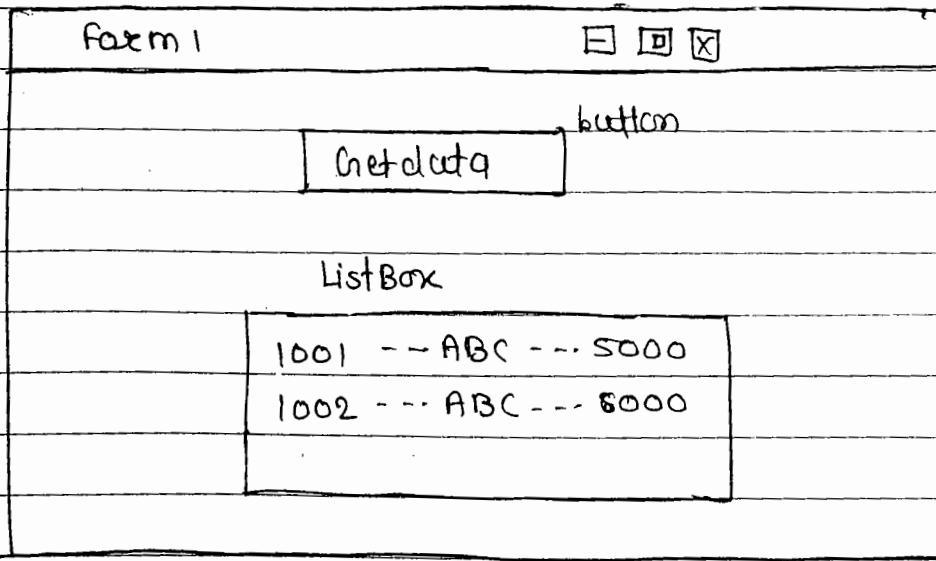
```
}
```

```
else
```

```
    MessageBox.Show ("no such record");
```

Output - no such record.

Example 2 :-



```
Using System.Data.SqlClient;
```

```
private void button
```

```
{
```

```

Sqlconnection cn = new SqlConnection ("uid = sa ;
database = sampledb1 ; password = 123 ;
server = nit - pc");
cn. Open ();
SqlCommand cmd = new SqlCommand ("select * from emp", cn);
SqlDataReader dr = cmd. ExecuteReader ();
while (dr. Read ())
{
    listBox1. Items. add (dr ["empno"] + "----"
+ dr ["ename"] + "----" + dr ["sal"]);
}

```

Points :-

- 1) Sql data reader reads data in the most efficient manner possible.
- 2) SqlDataReader is Read only & forward only, meaning once you read a record and goto next record there is no way go back to the previous record.
- 3) The forward only nature of sqldatareader is what makes it an efficient choice to read the data.
- 4) It is also not possible to change the data using sqldatareader.
- 5) Sqldatareader is connection oriented, meaning it requires an active connection to the data source while reading the data.

23/10/13  
wed

- 6) Instance of SqlDataReader can not be created using the new operator.
- 7) The SQL command objects execute Reader method creates and returns an instance of SqlDataReader. 23
- 8) SqlDataReader is connection oriented & the connection needs to be opened explicitly by calling the open method on the connection object before calling the execute reader method of command object.

### \* Read

This method moves the record pointer to the next record for the first time call of this method the record pointer finds out the the first records. After that for ever call it jumps to next record. If the next record is found then it returns true.

If the next record is not found that means whenever it is reach end of the data then it returns false.

### dr["Colname"]

This Indexer gets the value at the specified column name in the current row which is currently out by the record pointer.

### dr[\*Index\*]

This Indexer gets the value at the specified column index in the current row which is currently pointed out by the record pointer.

23/10/13  
Wednesday

The column index always start from [0]

Create a table

Q. 23  
Create table users (uname varchar (30),  
password varchar (30))

	uname	password
1	abcd	xyz
2	ganesh	ganesh
3	kalyan	kalyan.

Design a form.

form1	日回図
Username :	<input type="text"/>
Password :	<input type="text"/>
<input type="button" value="Login"/>	

Name

Using System.Data.SqlClient ;

code:-

```
private void button1_Click(object sender, EventArgs e)
{
    SqlConnection cn = new SqlConnection("uid=sq;
    database = sampledb; password = 123;
    server = nit@60");
    cn.Open();
    SqlCommand cmd = new SqlCommand("select * from user where uname = '" + TextBox1.Text + "'");
    SqlDataReader dr = cmd.ExecuteReader();
    while (dr.Read())
    {
        Label1.Text = dr["password"].ToString();
    }
}
```

```
SqlConnection cn = new SqlConnection ("uid=sq;
database = sampledb; password = 123;
server = nit@60");
```

cn.Open();

```
SqlCommand cmd = new SqlCommand ("select *
from user where uname = '" + TextBox1.Text + "'")
```

and password = " + textBox2.Text + " ", cn);

```
SqlDataReader dr = cmd.ExecuteReader()  
if (dr.HasRows)  
{
```

```
    Homepage obj = new Homepage();  
    obj.Show();  
    this.Hide();
```

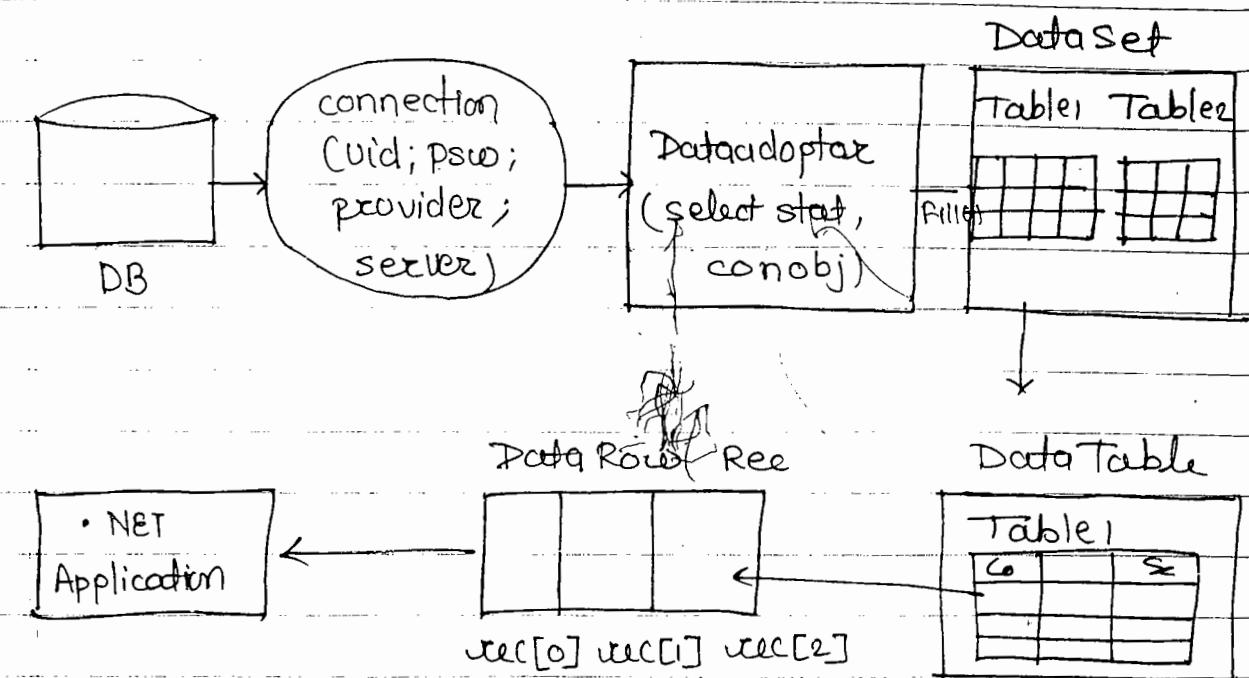
```
}
```

else

```
    MessageBox.Show("invalid uname/password");
```

Go to SQL explorer → right click on  
app → add new item → select  
Windows Form → name it as  
homepage.cs → click add button.

## \* Disconnected ORiented Architecture.



- 1) ADO. Net Data connection is of two types
  - 1) Connected Model : Implemented with Data Reader buffer.
  - 2) Disconnected model : Implemented with DataSet, DataTable, DataRow buffer.

To overcome the limitations of connected model ADO. net introduces Disconnected model

- 2) The main advantage of Disconnected model is even though the connection is not open it works. That means when you execute any stat. the connection will be open then stat will be executed & then finally the connection will be closed.
- 3) But the connected model is not like that if you take the buffer called data Reader which is used in the connected model it

works when the connection open only  
But the disconnected model buffers  
work even though the connection is  
opened or not.

### Library :-

#### 1) Connection :-

maintains the connection with database.

#### 2) Data Adapter

Sends A SQL statements + executes it at  
backend. This just like the command class  
in the connection oriented model. The difference  
bet' command class & data.adapter class is  
The command class is used to execute any  
type of statement (Insert, update, Select,  
Delete statement, stored procedure) but adapter  
execute select statement only.

#### 3) Data Set :-

Holds the data ie received from the  
data base after execution of data adapter.  
it can hold multiple tables data at a time

#### 4) Data Table :-

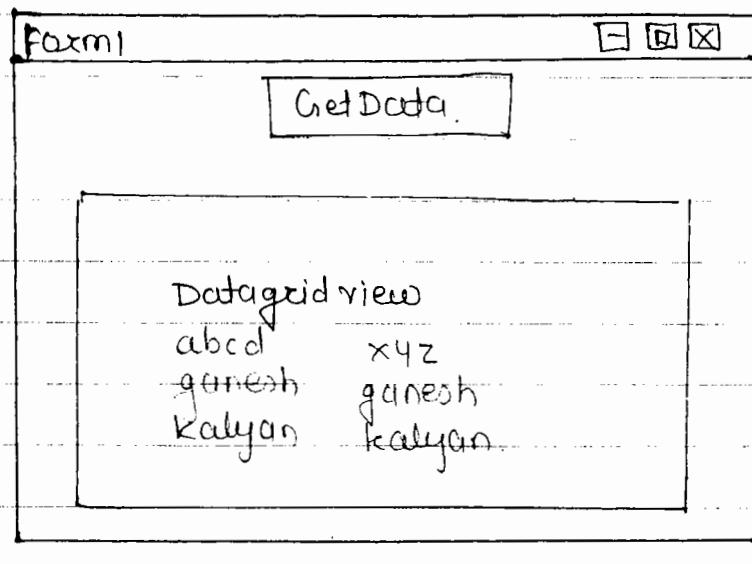
Holds a single table from the data set

#### 5) Data Row

holds a single row from the datatable

Go to tool box → go to data tab → select datagridview control → drag & drop in the form

Design a form



Using System. Data;

Using system. Data. Sqlclient;

private void button1\_Click(object sender, EventArgs e)

```
{  
    SqlConnection cn = new SqlConnection ("uid=su;  
    database = sampledb ; password = 123 ; server =  
    nit 60 ");
```

```
    SqlDataAdapter da = new SqlDataAdapter ("Select  
    * from users" , cn);
```

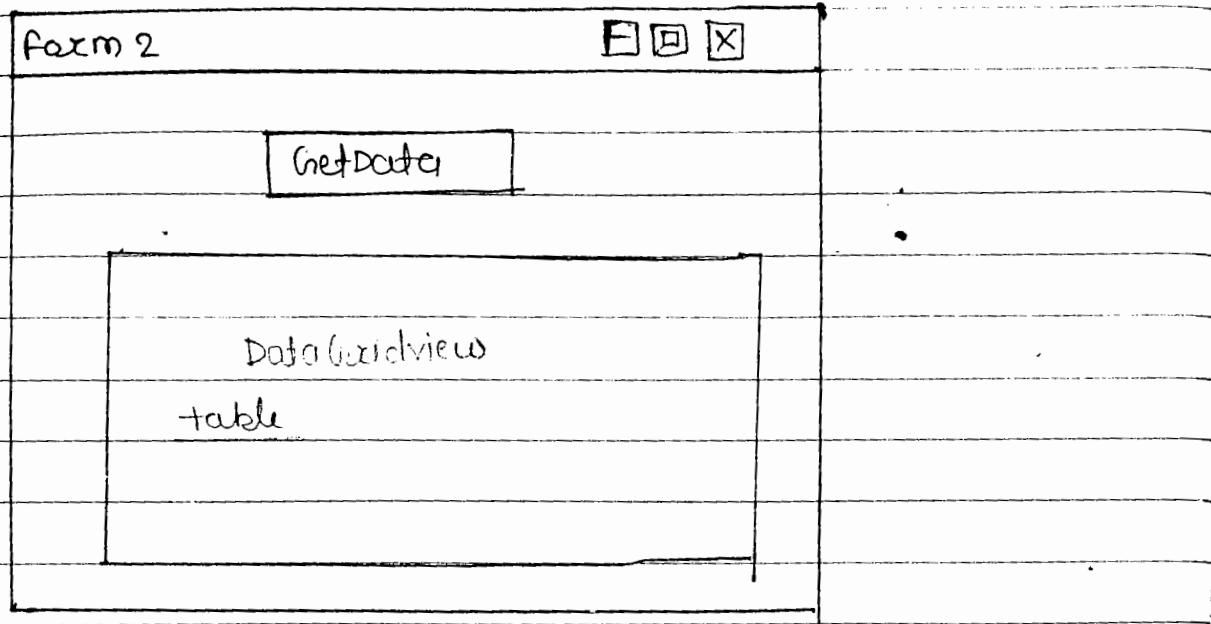
```
    DataSet ds = new DataSet ();
```

```
    da.Fill (ds);
```

```
    dataGridView1.DataSource = ds.Tables [0];
```

```
}
```

\* Multiple table stored in single Data Set



```
Using System.Data;  
Using System.Data.SqlClient;
```

```
private void button
```

```
{
```

```
SqlConnection cn = new SqlConnection ("uid=s4;  
database = sampledb; password = 123; server = nite");
```

```
SqlDataAdapter da1 = new SqlDataAdapter("select *  
from users", cn);
```

```
SqlDataAdapter da2 = new SqlDataAdapter("Select *  
from dept", cn);
```

```
DataSet ds = new DataSet();  
da1.Fill(ds, "Users");  
da2.Fill(ds, "Dept");  
dataGridView1.DataSource = ds.Tables[1];
```

DataSet Navigation :-  
Design a form

Form 1

Empno :	<input type="text"/>		
Ename :	<input type="text"/>		
Salary :	<input type="text"/>		
<input type="button" value="First"/>	<input type="button" value="Next"/>	<input type="button" value="Previous"/>	<input type="button" value="Last"/>

Namespace - Using System.Data.SqlClient

```
classes SqlConnection cn ;
SqlDataAdapter da ;
DataSet ds ;
int i=0 ;
private void form1_Load(object sender,
{
    cn = new SqlConnection ("uid = sq ; database = testdb ;
    password = 123 ; server = nit-pc ");
    da = new SqlDataAdapter ("Select * from emp", cn);
    ds = new DataSet ();
    da.Fill(ds);
}
public void getdata ()
{}
```

```
deretBox1.Text = ds.Tables[0].Rows[i][0].ToString();
deretBox2.Text = da.Tables[0].Rows[j][1].ToString();
deretBox3.Text = da.Tables[0].Rows[i][2].ToString();
```

// First button code

```
private void button1
```

```
{
```

```
i = 0
```

```
getdata();
```

```
}
```

// next button code

```
private void button2_Click
```

```
{
```

```
j = i + 1;
```

```
getdata();
```

```
}
```

// Previous button code

```
private void button3_Click
```

```
{
```

```
j = j - 1;
```

```
getdata();
```

```
}
```

```

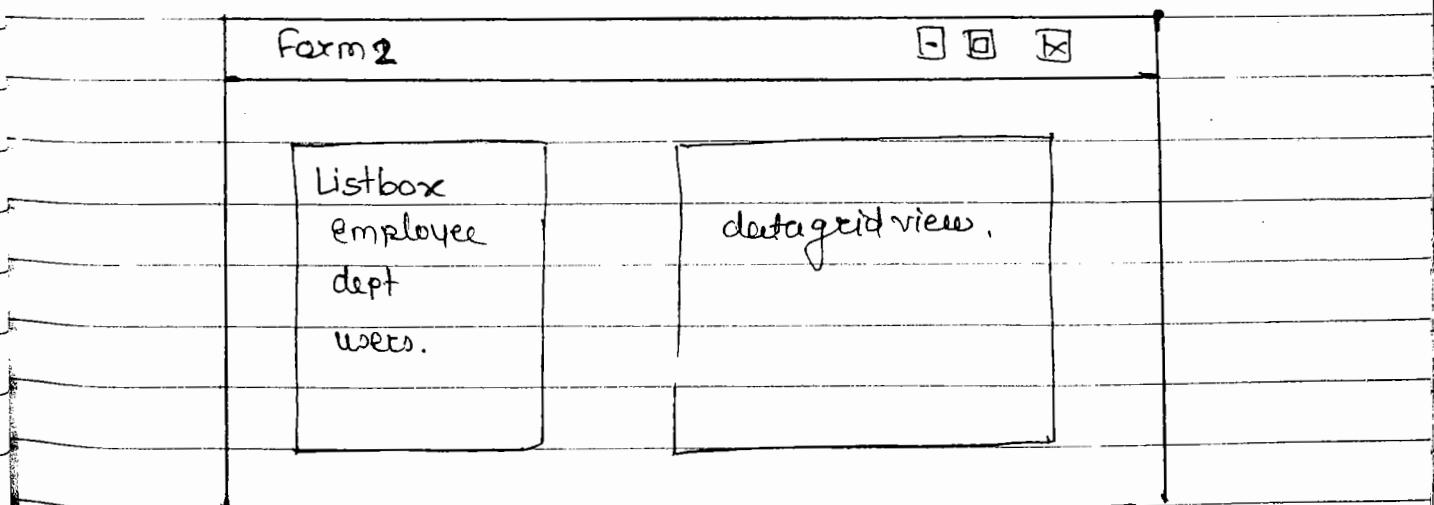
    // last button code
private void button 4 - click
{
    i = ds . Tables [0] . Rows . Count - 1 ;
    getdata () ;
}

```

### Data Table

- 1) Data Table is a class available in System.Data namespace.
- 2) The data table object represent an in memory database table. you can add rows to a data table with a SQL data adapter , with a SQL data reader with an XML file or programmatically.

Design a form



Using System . Data ;

Using System . Data . SqlClient ;

```
Sql connection cn;
```

```
Sql Data Adapter da1, da2, da3;
```

```
Data set ds;
```

```
private void form2_Load (object
```

```
{
```

```
cn = new Sql Connection ("uid = sa; database = deptdb;  
password=123; server = nit-pc");
```

```
da1 = new SqlDataAdapter ("Select * from employee");  
da2 = new SqlDataAdapter ("Select * from dept", cn);  
da3 = new SqlDataAdapter ("Select * from users", cn);
```

employee

```
ds = new DataSet();  
da1.Fill (ds, "employee");  
da2.Fill (ds, "Dept");  
da3.Fill (ds, "users");
```

```
foreach (DataTable dt in ds.Tables)
```

```
{
```

```
listBox1.Items.Add (dt.TableName);
```

```
}
```

```
}
```

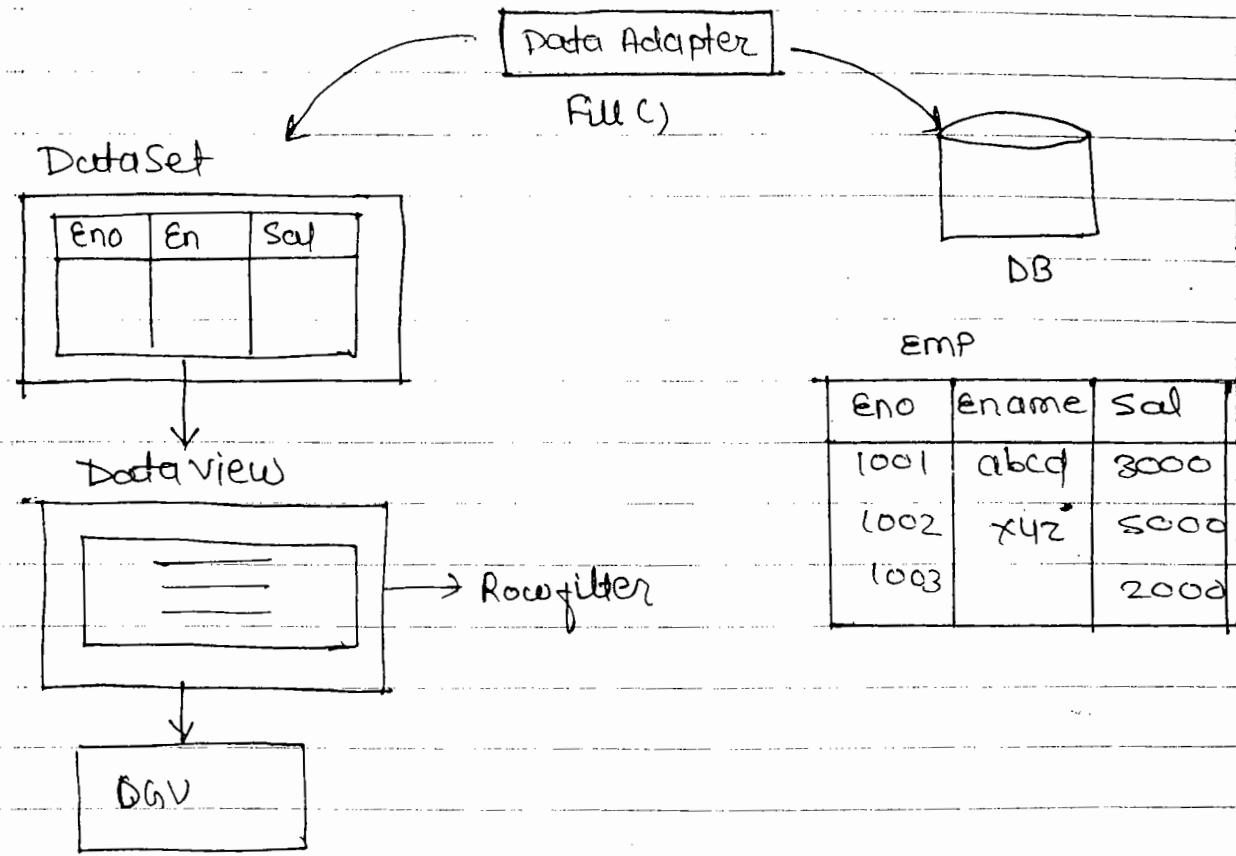
```
dbl click  
on listBox  
private void button1_Click (object
```

```
{
```

```
dataGridview1.DataSource = ds.Tables [listBox1.Selected  
Item.ToString ()];
```

```
}
```

## Data View



- 1) The data view is a class available in `System.Data` namespace.
- 2) The data view object represent an in memory data base view 3) you can use a data view object to create a selectable, so filterable view of a datatable.
- 4) The data view object support 3 imp properties
  - 1) `SORT` unable you to sort the rows represented by the data view

2) Row filter :-

Enables you to filter the rows represented by the data view.

3) Row state filter :-

Enables you to filter the rows represented by the data view according to the row state.

(for ex original rows, current rows, unchanged)

Design a form.

Form 3

日回区

Code cl

on get  
button

DataGrid view

ENO	ENAME	SAL
1001	-	3000
1002	-	5000
1003	-	4000

Salary :

4000

GetData button.

O/P	ENO	ENAME	SAL
	1002	-	5000

(db click on

Using System. Data ;  
Using System. Data. SqlClient ;

farm)

sqlConnection cn ;

sqlDataAdapter da ;

DataSet ds ;

DataTable dv ;

private void Form3\_Load(object sender

{

```
cn = new SqlConnection ("uid = sa; database = testdb;  
password = 123; server = nit-pc");
```

```
da = new SqlDataAdapter ("Select * from emp", cn);
```

```
ds ds = new DataSet();
```

```
da.Fill(ds);
```

```
dataGridView1.DataSource = ds.Tables[0];
```

```
}
```

```
button click private void button1_Click (object sender  
on getdata) {
```

```
button
```

```
dv = new DataView (ds.Tables[0]);
```

```
dv.RowFilter = "sal >" + textBox1.Text;
```

```
dataGridView1.DataSource = dv;
```

```
}
```

10/13  
Friday.

## Relations In Dataset -

Master Table			Child Table				
Dept			Emp				
Dno	Dname	Loc	Eno	Ename	sal	Deptno	
10	Sales	Hyd	1001	abcd	5000	10	
20	HR	Hyd	1002	Xyz	9000	20	
30	Market	Hyd	1003	Balaji	5000	10	

↓                                  ↓  
 primary key                      foreign key

### Creation of Master table :-

Create table dept ( dno int primary key , dname varchar (80) , loc varchar (30) )

insert into dept values (10, 'Sales', 'hyd')

insert into dept values (20, 'HR', 'hyd')

insert into dept values (30, 'Market', 'Hyd')

### Creation of child table

Create table emp ( ~~eno~~ empno int , ename varchar (30) ,  
 sal int , deptno int references dept ( dno ) )

Insert into emp values ( 1001 , 'ganesh' , 4000 , 10 )

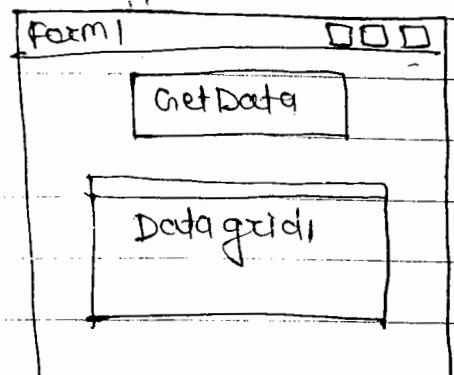
Go to tool box → right click on data tab → choose Items → activate datagrid control checkbox click ok button -

Go to tool box → select data grid control drag and drop in the form.  
namespace

```
using System.Data;  
using System.Data.SqlClient;
```

Code:- private button1 - click(object

```
{
```



```
SqlConnection cn = new SqlConnection ("uid=sa;  
database=sampledb; password=123,server=nit-pc");  
SqlDataAdapter da1 = new SqlDataAdapter ("Select *  
from dept", cn);
```

```
SqlDataAdapter da2 = new SqlDataAdapter ("Select *  
from employee", cn);
```

4d)

```
DataSet ds = new DataSet ();  
da1.Fill (ds, "dept");  
da2.Fill (ds, "Employee");
```

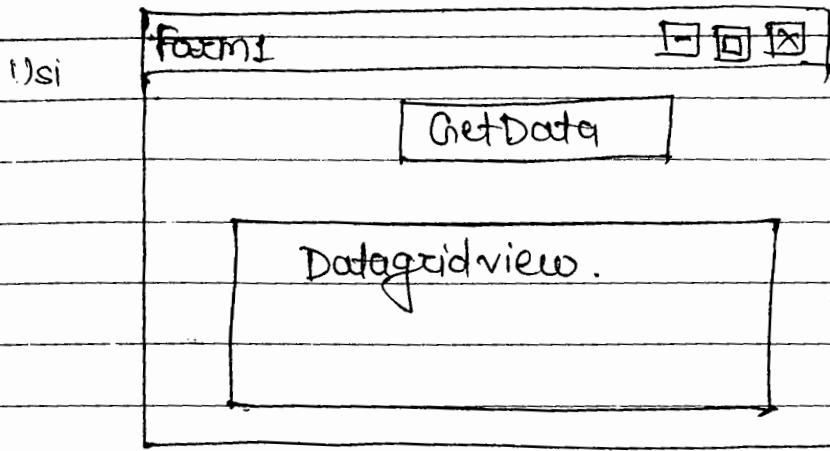
```
DataColumn pk = ds.Tables ["Dept"].Columns ["dno"];  
DataColumn fk = ds.Tables ["Employee"].Columns ["deptno"];
```

```
DataRelation dr = new DataRelation ("ref", pk, fk);  
ds.Relations.Add (dr);  
dataGridView1.DataSource = ds;
```

```
}
```

Dataset visualizes da.

Go to solutionexpl → right click on the app-  
add new item → select template class file →  
name it as Dbaction.cs → click add button.



Using System.Data;  
Using System.Data.SqlClient;  
namespace sampleapplication

{

class Dbaction

{

public static DataSet getdata()

{

SqlConnection cn = new SqlConnection("uid=sq;  
database=testdb;password=123;server=nit-pc");

SqlDataAdapter da = new SqlDataAdapter("Select  
\* from emp", cn);

DataSet ds = new DataSet();

da.Fill(ds);

return ds;

}

Design a form.

dbl click on button1

Code:- private void button1\_Click ( object sender, Event.

{

DataSet myds = Dbaction . getdata ();

dataGridView1 . DataSource = myds . Tables [0] ;

}

Example 3

Form1

1001

Eno :

No of rows displayed

Ename:

GetData.

Using System . Data ;

Using System . Data . SqlClient ;

Code:- private void button1\_Click ( object sender, Event --

{

class Dbaction

{

public static string getrecords ( int eno )

{

26/10/13  
saturday

```
SqlConnection cn = new SqlConnection ("uid = sa; database  
= sampledb; password = 123; server = nit - pc");  
cn.Open();
```

```
SqlCommand cmd = new SqlCommand ("Select * from emp  
where empno = " + eno + ", cn );  
SqlDataAdapter dt = cmd.ExecuteReader();
```

```
if (dt . Read ())
```

```
{
```

```
return dt ["ename"].ToString ();
```

```
}
```

```
else
```

```
return " no such record ";
```

```
}
```

button code

```
private void button2_Click (object
```

```
{
```

```
textBox2.Text = Dbaction . getrecords (int . Parse  
(textBox1.Text));
```

```
}
```

26/10/13  
Saturday

## StoredProcedure

### Syntax

Create procedure < pName >

@par1 datatype,  
@par2 datatype,  
----- )

as

begin  
statements  
end

Example:-

Create procedure total123

(@a int, @b int, @c int, @d int)

as

begin  
print @a + @b + @c + @d

End

To execute :-

exec total123 10, 20, 30, 40

O/P = 100

Example:- Create procedure spInsert1

(@eno int, @name varchar(30),  
@sal int)

as

begin

insert into emp values (@eno, @name, @sal)

End

To execute :-

exec spInsert1 1001, 'Kalyan', 9000

Update :-

Create procedure spupdate

(@eno int, @name varchar(30), @sal int)

as

begin

update into emp values (@eno, @name, @sal)

update emp set (@name = ename, @sal = sal)

ename = @name, sal = @sal where empno = @eno

end

exec spupdate 1001, 'balaji', 3000

Delete :-

Create procedure spdelete1

(@eno int)

as

begin

delete from emp where empno = @eno

end

exec spdelete 1001

select \* from emp

Retrieving data.

Create procedure spgetdata

as

begin

select \* from emp

end

exec spgetdata.

- it)
- 1) When front end encounter cmd.execute nonquery  
front end sends request to the database.
  - 2) At database side 2 actions will be performed
    - 1) Compilation of the query
    - 2) Running the query.
  - 3) At the time of compilation database engine will check for the syntactical error & if there any errors occur that error will be send as it is to the front end.
  - 4) If there are no errors query will be run and the result will be sent to front end
  - 5) How many times the request may be sent these two actions will be performed.
  - 6) But if there are no syntactical error compilation of the query is not required and will increase burden on the database so performance will be reduced.

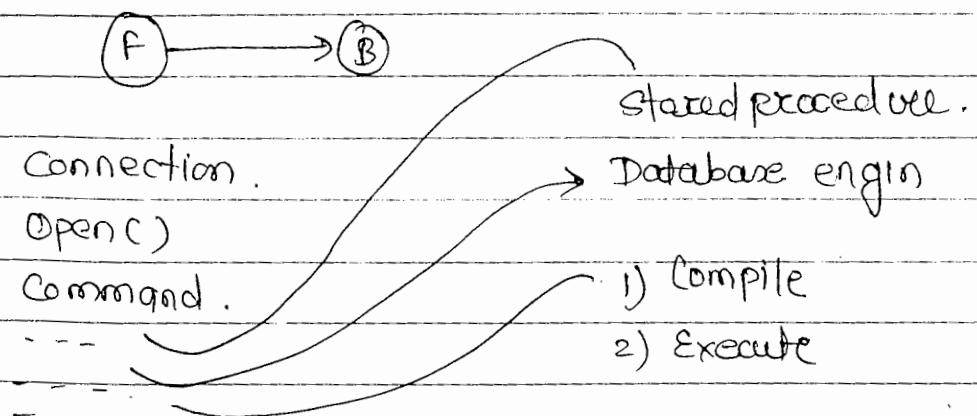
Disadvantage of the working with queries.

- 1) Unnecessary compilation will increase burden on the database
- 2) Application performance is reduced
- 3) User will get delight (date) response
- 4) no code reusability of query.

To overcome these disadvantages stored procedures are used.

A stored procedure is a database object which contains precompiled queries

whenever a stored procedure is called from front end queries present in stored procedure will not be compiled rather will run directly



Design a form

Form 1	
Empno :	<input type="text"/>
Ename :	<input type="text"/>
Salary :	<input type="text"/>
<input type="button" value="Insert"/>	

dbl click on insert -

Using System.Data.SqlClient;

Code:- private void button1\_Click

{

```
SqlConnection cn = new SqlConnection ("uid = sa; database =  
destdb; password = 123; server = nit-pc");  
  
cn.Open();
```

```
SqlCommand cmd = new SqlCommand ("spinsert", cn);
```

```
cmd.CommandType = CommandType.StoredProcedure;
```

```
cmd.Parameters.AddWithValue ("@eno", textBox1.Text);
```

```
cmd.Parameters.AddWithValue ("@name", textBox2.Text);
```

```
cmd.Parameters.AddWithValue ("@sal", textBox3.Text);
```

```
cmd.ExecuteNonQuery();
```

```
MessageBox.Show ("record added");
```

}  
Update button code.

Using System.Data.SqlClient;

code private void button1

{

```
SqlConnection cn = new SqlConnection ("uid = sa; database  
= deotdb; password = 123; server = nit-pc");
```

```
cn.Open();
```

```
SqlCommand cmd = new SqlCommand ("spupdate", cn);
```

```
cmd.CommandType = CommandType.StoredProcedure;
```

```
cmd.Parameters.AddWithValue ("@eno", textBox1.Text);
```

```
cmd.Parameters.AddWithValue ("@name", textBox2.Text);
```

```
cmd.Parameters.AddWithValue ("@sal", textBox3.Text);
```

```
cmd.ExecuteNonQuery();
```

```
MessageBox.Show ("record updated");
```

}

Step of stored procedure.

Step 1 - Import the library :-

Using System.Data.SqlClient

Step 2 - Construct the connection class object

```
SqlConnection cn = new SqlConnection ("-----")
```

Step 3 - Open the connection,

```
cn.open.
```

Step 4 : construct the command class object.

```
SqlCommand cmd = new SqlCommand ("procName", cn);
```

Step 5 :- Assign the command type to command class object.

```
cmd.CommandType = CommandType.StoredProcedure;
```

Step 6 :- Assign the parameters to the command class object

```
cmd.Parameters.AddWithValue ("parName", value);
```

Step 7 :- Execute the procedure

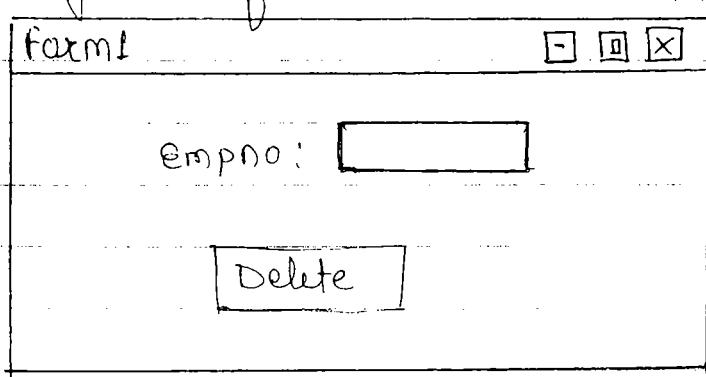
```
cmd.ExecuteNonQuery()
```

Step 8 :- close the connection :-

```
cn.Close()
```

Delete

Design a form.



using System.Data.SqlClient;

```
private void button1_Click()
{
    SqlConnection cn = new SqlConnection("uid = sa; database
                                         = master; password = 123; server = nit-pc");
    cn.Open();
    SqlCommand cmd = new SqlCommand("spdelete", cn);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.AddWithValue("@eno", textBox1.Text);
    cmd.ExecuteNonQuery();
    MessageBox.Show("procedure executed");
}
```

Retrieving the data using stored procedure

Create database spgetdataemp

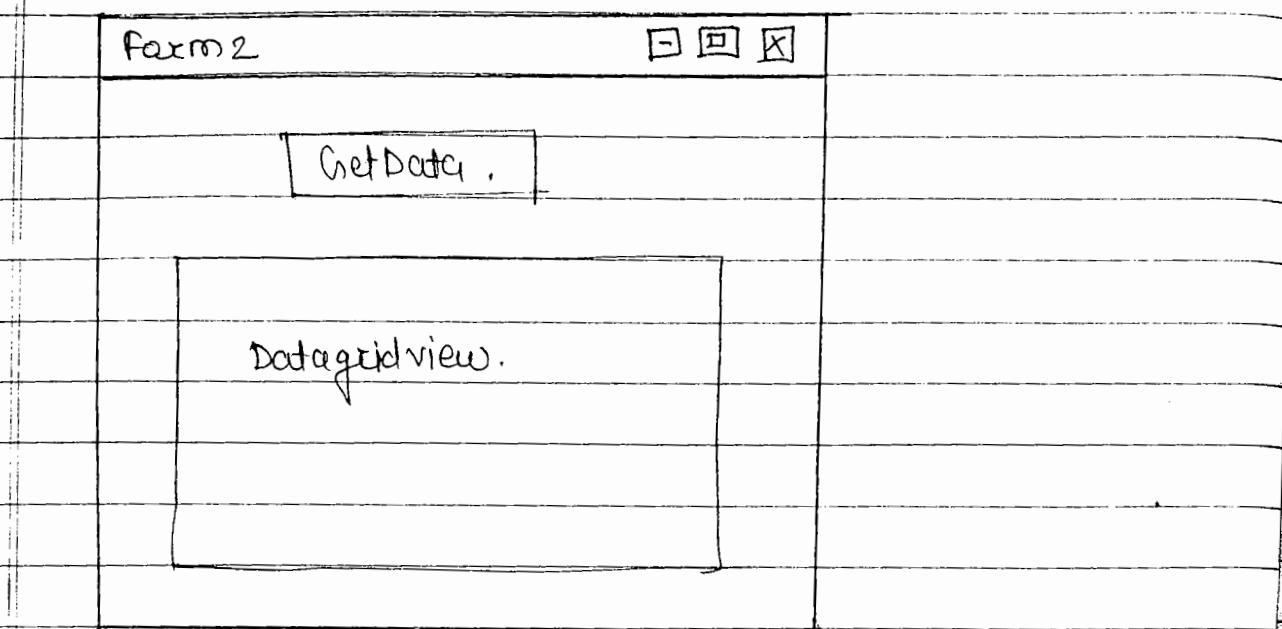
as

begin

Select \* from emp

End.

Go to toolbox → select data gridview control  
→ drag & drop in the form.



```
Using - System. Data ;
Using System. Data. SqlClient ;
!
private void button1 ...
{
    SqlConnection cn = new SqlConnection ("uid = sa; database
        = master, password = 123; server = nit-pc");
    SqlDataAdapter da = new SqlDataAdapter ("sp_getdataemp",
        , cn);
    DataSet ds = new DataSet ();
    da.Fill (ds);
    dataGridView1 .Data Source = ds.Tables [0];
}
```

Stored procedure parameters.

1) IN parameter.

2) Out parameter.

By default every parameter is a IN parameter

Select ename from emp where empno = 1001  
                +  ↓  
                out   IN

Create procedure sp getdata

(@eno int ,

@ename varchar (30) out )

as

begin

    Select @name = ename from emp  
    where empno = @eno

end

To execute

declare @res varchar (30)

exec spgetdata 1002 , @res out

print @res

Output - haresha .

Design a form.

Form 1	
1001	
Empno :	<input type="text"/>
<input type="button" value="Getdata"/>	
Label 2 haresha.	

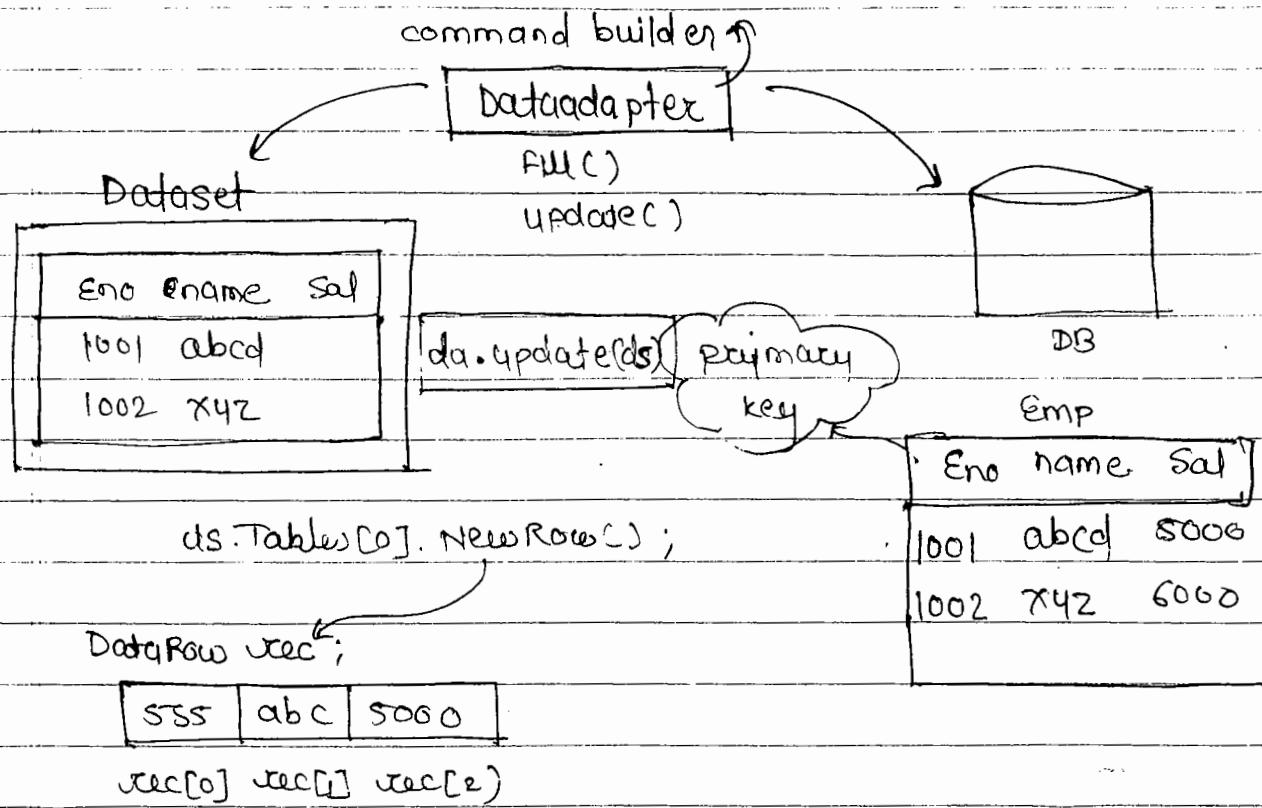
29/10/  
Tues

```
bt click Using System.Data;  
getdata. using System.Data.SqlClient;  
private void button1  
{  
    SqlParameter P1, P2;  
    SqlConnection cn = new SqlConnection("uid=s4; database=  
master; password=123; server=nit-pc");  
    cn.Open();  
    SqlCommand cmd = new SqlCommand("spgetdata", cn);  
    cmd.CommandType = CommandType.StoredProcedure;  
    P1 = new SqlParameter("@eno", SqlDbType.Int);  
    P1.Value = int.Parse(textBox1.Text);  
    cmd.Parameters.Add(P1);  
  
    P2 = new SqlParameter("@name", SqlDbType.VarChar,  
    50);  
    P2.Direction = ParameterDirection.Output;  
    cmd.Parameters.Add(P2);  
  
    cmd.ExecuteNonQuery();  
    label2.Text = cmd.Parameters["@name"].Value.  
    ToString();
```

29/10/13  
Tuesday.

9-11

## \* Command builder :-



you can use a data adapter not only retrieving data from a database. you can also use a data adapter when updating, inserting, & deleting data from a database.

If you call SQL data adapter objects **update** method & passed the method a **datatable** then the SQL data adapter calls its **update** command **insert** command, & **delete** command to make changes from database. you can use the **SQL builder** command object to create **update** command **insert** command & **delete** command.

- g) The SQL command builder class **SqlBuilder** adapts that has a **select** command & generates the other 3 commands automatically.

Design a form

Form1			
Empno :	<input type="text"/>		
Ename :	<input type="text"/>		
Salary :	<input type="text"/>		
<input type="button" value="First"/>	<input type="button" value="Next"/>	<input type="button" value="Clear"/>	
<input type="button" value="Insert"/>	<input type="button" value="Update"/>	<input type="button" value="Delete"/>	

del click on form.

```
using System.Data;
using System.Data.SqlClient;
```

Code} @Initialize

```
SqlConnection cn;
SqlDataAdapter da;
DataSet ds;
DataRow dr;
SqlCommandBuilder bld;
int i;
```

private void form1\_Load

{

```
cn = new SqlConnection("uid=sa; database=master;
password=123; server=nit-pc");
da = new SqlDataAdapter("Select * from emp", cn);
```

```
ds = new DataSet();
da.Fill(ds);
}
public void getdata()
{
```

```
textBox1.Text = ds.Tables[0].Rows[i][0].ToString();
textBox2.Text = ds.Tables[0].Rows[i][1].ToString();
textBox3.Text = ds.Tables[0].Rows[i][2].ToString();
}
```

dbl click on first button.

```
private void button1_Click
{
    j=0
    getdata();
}
```

```
private void button2_Click
{
    i = j + 1;
    getdata();
}
```

```
private void button3_Click
{
    textBox1.Clear();
    textBox2.Clear();
    textBox3.Clear();
}
```

```
private void button4_Click
```

```
{
```

```
    rec = ds.Tables[0].NewRow();
    rec[0] = textBox1.Text;
    rec[1] = textBox2.Text;
    rec[2] = textBox3.Text;
```

```
    ds.Tables[0].Rows.Add(rec);
```

```
    MessageBox.Show("Records added into dataset");
```

```
    bldr = new SqlCommandBuilder(da);
```

```
    da.Update(ds);
```

```
    MessageBox.Show("Record added into database");
```

```
}
```

```
private void button5_Click
```

```
{
```

```
    ds.Tables[0].Rows[i][0] = textBox1.Text;
```

```
    ds.Tables[0].Rows[i][1] = textBox2.Text;
```

```
    ds.Tables[0].Rows[i][2] = textBox3.Text;
```

```
    MessageBox.Show("Record modified in dataset");
```

```
    bldr = new SqlCommandBuilder(da);
```

```
    da.Update(ds);
```

```
    MessageBox.Show("Record modified in database also");
```

```
}
```

```

private void
{
    ds.Tables[0].Rows[i].Delete();
    MessageBox.Show("record deleted in dataset");

    bldr = new SqlCommandBuilder(da);
    da.Update(ds);
    MessageBox.Show("records deleted in database also");
}

```

## Application Configuration files :-

Stating the connection string in App.Config file :-

- 1) In the live project development we may require to offer one or more customizable values in the code. For ex. you can take a database server name at the development time we use the database server in the software company but when the project is given to the customer the project should under the database server which is the customer company.
- 2) At that they should a facility to customize server name after installing the project in the customer company.
- 3) But we don't give the source code to the customer so that the customer can not change the database server name.

- 4) To solve this problem this kind a app" setting could be maintaining another file separately which can be modified on the client system even after installing the project in the customer's work station.
- 5) In .net framework the app" setting can be saved in a "config" file
- 6) A "config" file contains ".config extension" &
- 7) In console of windows app" is called as "App.Config".
- 8) In website it is called as "web.config"
- 9) The config file is written in XML Language

### App Settings :-

- 1) It is used to declare the variables globally whose data can be used in multiple pages
- 2) To create the variable this app settings will contain a subtag known as {add}
- 3) This add subtag contain 2 attribute known as
  - 1) key
  - 2) Value.
- 4) key is used to stored the variable name value is used to stored the data to be stored in that variable.
- 5) To access this variable in our windows forms using "configuration settings".
- 6) configuration setting is class available in System.Configuration namespace
- 7) This class will contain a collection with the name app setting if we can access the value like,

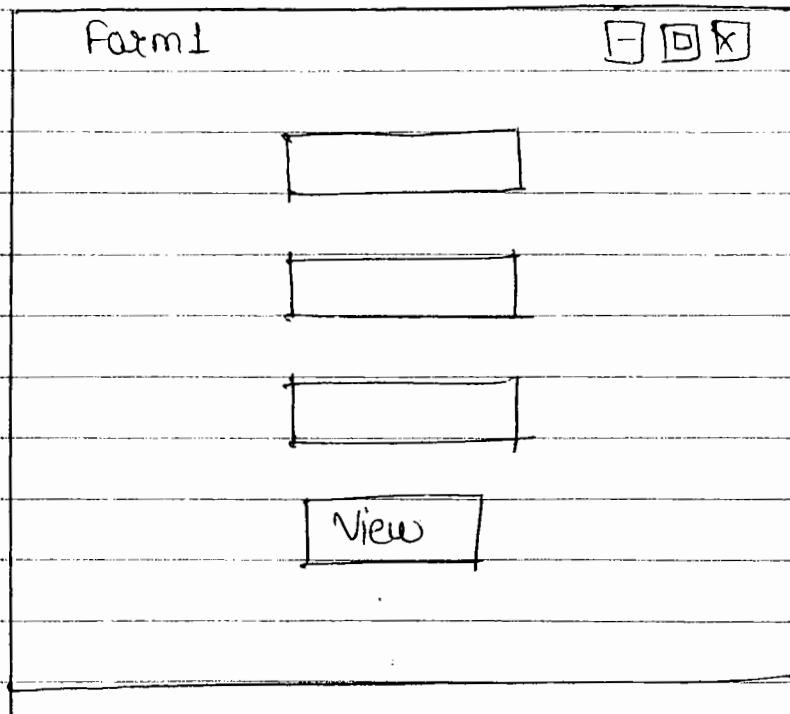
30/10/13  
Wednesday

## Configuration settings. App settings ["a"]

Go to solution explore → right click on app →  
add new item → select template - app configuration  
file click add button.

```
<?xml version = "1.0" encoding = "utf-8"?>
<configuration>
  <appSettings>
    <add key = "a" value = "100"/>
    <add key = "abcd" value = "ganesh"/>
    <add key = "uname" value = "xyz"/>
  </appSettings>
</configuration>
```

Design a form



ujwala

using System.Configuration;

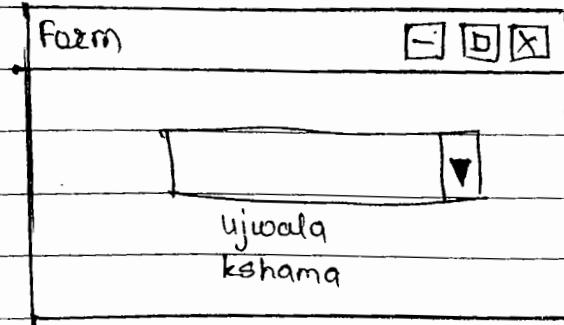
```
private void button1_Click()
{
    textBox1.Text = ConfigurationSetting.AppSettings["a"];
    textBox2.Text = ConfigurationSetting.AppSettings[1];
    textBox3.Text = ConfigurationSetting.AppSettings["uname"];
}
```

Go to App.config file.

```
<configuration>
<appSettings>
<add key="conn" value="ut=sq; database
=master ; password=123 ; server=nit-pc"/>
```

```
</appSettings>
</configuration>
```

Go to toolbar - select combobox control - drag & drop in the form.



```
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
```

```
private void button1_Click(object sender, EventArgs e)
```

```
SqlConnection cn = new SqlConnection ( configuration  
Settings.AppSettings ["conn"]);
```

```
SqlDataAdapter da = new SqlDataAdapter ("Select * from emp",  
cn);
```

```
DataSet ds = new DataSet();
```

da . fill (ds) ;

ComboBox1.DataSource = ds.Tables[0];

ComboBox1.DisplayMember = "ename";

10

Example :-

Go do sal<sup>n</sup> explorer → goto app.config file →

## < configuration >

## <appSettings>

<add key = "conn">

```
value = "uid = sa; database = master; password  
= 123; server = nit-pc" />
```

## < /appSettings >

</configuration>

Go to solution explorer → right click on application add new item → select template class file - name it as Dbaction.cs click add button.

```
using System.Data;  
using System.Data.SqlClient;  
using System.Configuration;
```

namespace

{

class Dbaction

{

```
public static DataSet getdata()
```

{

```
SqlConnection cn = new SqlConnection(ConfigurationManager  
Setting.AppSettings["conn"]);
```

```
SqlDataAdapter da = new SqlDataAdapter  
("select * from emp", cn);
```

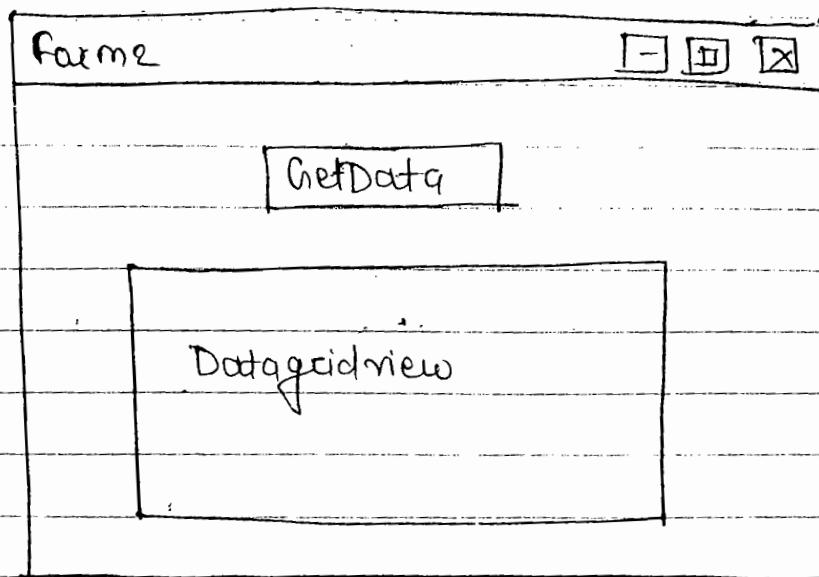
```
DataSet ds = new DataSet();
```

```
da.Fill(ds);
```

```
return ds;
```

}

Design form



dbl click on GetData button.

private void button1\_Click

{

```
DataSet myds = Dbaction.GetData();
dataGridView1.DataSource = myds.Tables[0];
```

}

Example 8:-

Design a login page.

Registration.

Uname

PSW

UName

Age

PSWD

mob no

Email id

LinkedIn

New User Registration

Country

Save

go for homepage

81/10/

## Homepage

Show products  
Add products  
Search products

## Showproduct

DGV

## Add products

PNO   
PName   
price   
qty

## Search products

PNO

31/10/13

## Data Set in XML

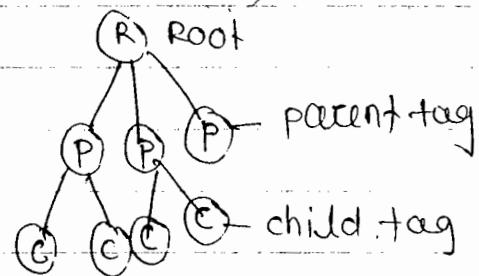
Why XML?

In general when app's designs in various technologies can not communicate with each other because every technology will have its own proprietary format. because all companies can not be computerized with same technology and which is impossible so if we want to communicate among the applications designed in various technologies we use XML.

What is XML?

- 1) XML is an open standard format proposed by W3 organisation.
- 2) XML stands for extensible mark up language.
- 3) XML is a tag based programming language.
- 4) XML is just used to described the data.
- 5) XML tags are case sensitive.
- 6) XML code can be written in any text editor.
- 7) To execute the XML code is required to have the XML parser (compiler).
- 8) By default all the browsers contain XML parser so XML file can be executed in any browser.
- 9) XML file will stored the data in hierarchical manner.
- 10) XML file will contain only one root tag.
- 11) Every root tag will contain one or more parent tags. Every parent tag will contain one or more child tags.
- 12) In general root tag will represent the database name parent tag will represent the table name.

child tag will represents the field names  
within XML data is stored in a file with  
an extension of XML structure is stored  
in a separate file with an extension .xsd  
(XML schema document )



Open the nodepad & write the following code :

```
<rootdata>
<emp>
  <empno>1000</empno>
  <ename>balaji</ename>
  <sal>4000</sal>
</emp>
<product>
  <pno>1</pno>
  <price>20000</price>
  <qty>12</qty>
</product>
</rootdata>
```

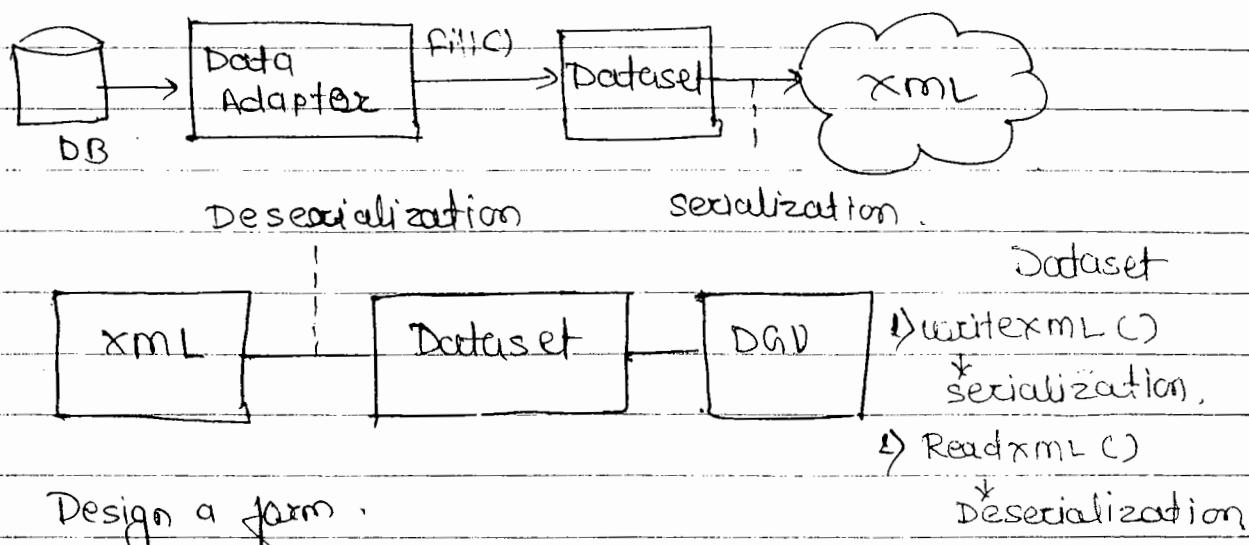
Save - c:\filename.xml

## Serialization

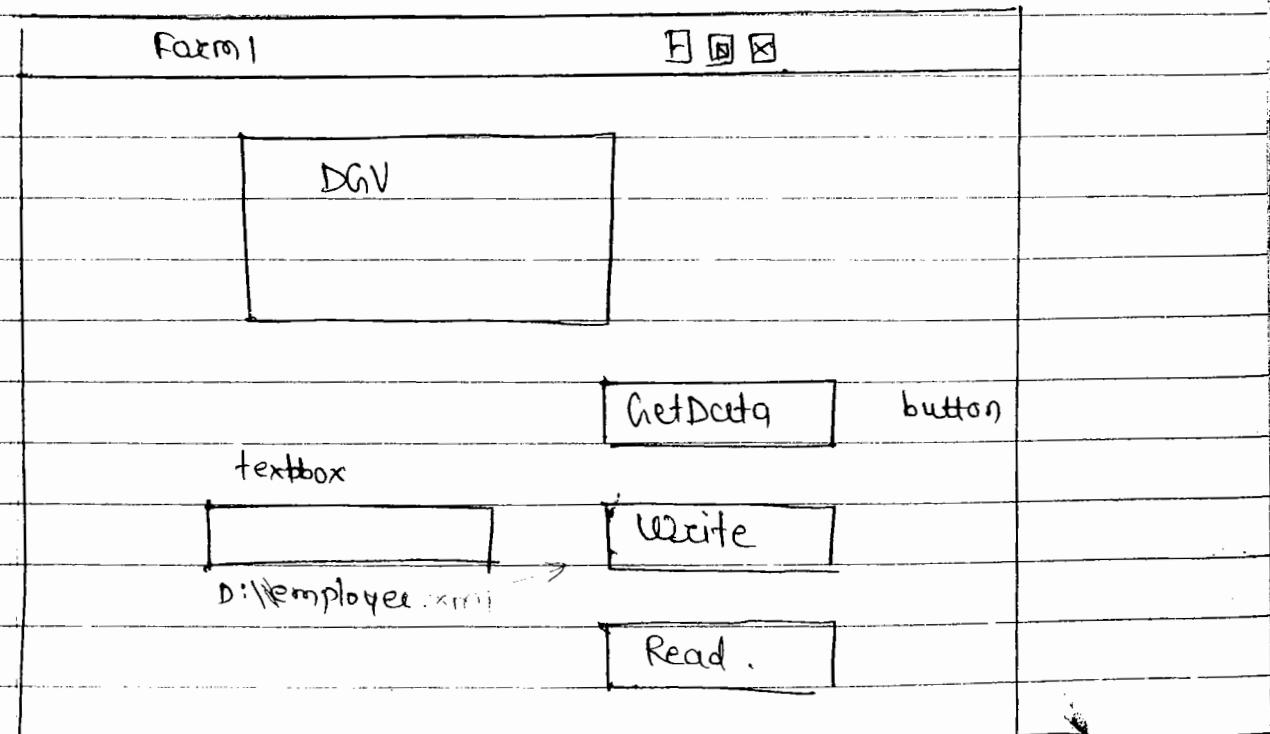
Is a process of converting a ~~data~~ memory objects to physical format

## Deserialization :-

Deserialization is a process of converting the data physical form to memory objects form.



Design a form .



```
using System.Data;
```

```
using System.Data.SqlClient;
```

```
Initialize
```

```
}
```

```
SqlConnection cn;
```

```
SqlDataAdapter da;
```

```
DataSet ds = new DataSet("empdataset");
```

```
// Get Data button code
```

```
private void button1_Click(object
```

```
{
```

```
cn = new SqlConnection("uid = sa; Database = master;
```

```
password = 123; server = nit-pc");
```

```
da = new SqlDataAdapter("Select * from emp", cn);
```

```
da.Fill(ds, "Employee");
```

```
dataGridView1.DataSource = ds.Tables[0];
```

```
}
```

```
// Write button code
```

```
private void button2_Click(object --)
```

```
{
```

```
ds.WriteXml(textBox1.Text);
```

```
messageBox.Show("Serialization
```

```
}
```

```
// Read button code
```

```
private void button3_Click(object --)
```

```
{
```

```
ds.ReadXml(textBox1.Text);
dataGridView1.DataSource = ds.Tables[0];
```

## MultiThreading :-

- 1) MultiThreading is used to execute multiple methods or multiple application are work simultaneously using multithreading we can communicate with network very flexible.
- 2) The thread class sharing the processor speed by default the processor executes only one thread, remaining all threads are deactivated automatically.
- 3) In .net framework we can implement multithreading by importing "System.Threading" name space.

## Thread :-

using this class we are created multiple objects. This class creating & managing the created threads.

### Thread class methods :-

#### 1) Start()

using thread class we are created multiple objects by default every thread is unstarted thread we can start the created thread using "start" method.

#### 2) Sleep()

Sleep() method deactivates the executed thread features.

The sleep method gets expiry time after expiry time thread executes automatically.

Suspend();

The suspend method deactivates the executed thread features. The suspend thread feature can be reusable in the application using resume method.

1/11/13

## \* Implementation of Multithreading \*

- .Net framework offers a namespace called "System.Threading" for implementation of multithreading.

library :- System.Threading.Thread

Step 1 : Import the API  
using System.Threading

Create the thread object

Thread Th = new Thread (method name)

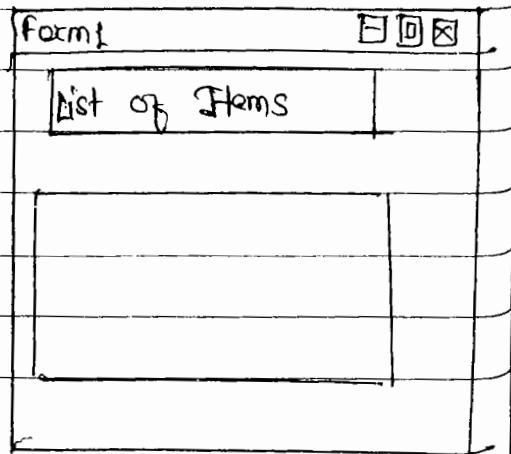
Start the thread  
Th.start

Example:-

using System.Threading;

private void form1\_Load()

{}



```
form1. CheckForIllegalCrossThreadCalls = false;
```

```
}
```

```
void f1()
```

```
{
```

```
for (int j=0; j<=100; j++)
```

```
{
```

```
listBox1.Items.Add("List 2 :" + j);
```

```
}
```

```
}
```

Thread d1, d2

```
ng.
```

```
private void button1_Click()
```

```
{
```

```
d1 = new Thread(f1);
```

```
d2 = new Thread(f2);
```

```
d1.Priority = ThreadPriority.Highest;
```

```
f1.Start();
```

```
d2.Start();
```

```
}
```

Application Logic :-

With

While we are developing some application you write some application logic.

The application logic can be divided as 3 types

1) Presentation logic

2) Business logic

3) Database logic.

## UI - User Interface.

### 1) Presentation logic :-

- This include with reading the input values from the control, updating the UI by changing the properties displaying some output message etc
- For example you take login form.

Login Form

Uname :	<input type="text"/>
password :	<input type="text"/>
<input type="button" value="Login"/>	

In that login form after clicking on of button getting the user name & password from the text Boxes is called as "Presentation Logic".

### 2) Business logic :-

- This includes with validating the input values performing some calculation implementing some formulas.
- In this login example checking the username & password whether those are entered properly or not is called as "Business logic"

### 3) Database logic :-

Based on the day that you are implementing the presentation, business & database logics is

- This include with getting some data from the database or insert or modifying some data in the database.

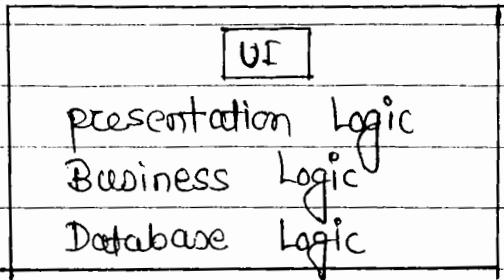
- In this login example getting the users data from the database is called as "database logic"

The Application development Architecture :-

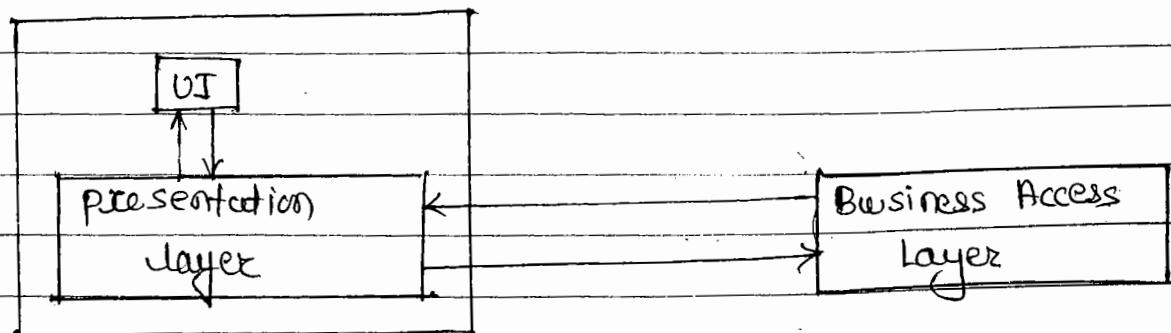
Based on the way that you are implementing the presentation, business & database logic in your app. These are development architecture.

- 1) One tier architectures (or) monolithic architecture
- 2) Two tier architecture.
- 3) Three tier architecture.
- 4) multi-tier or distributed archi or n-tier arch.

- 1) One-tier Architecture / monolithic architecture
  - 1) All type of logic (presentation logic, Business, Database logic) will be implemented directly with in the form
  - 2) That means no separation of presentation business & database logic.
  - 3) There is no reusability of Business code, Database code.
  - 4) This type of application are not in the professional type.

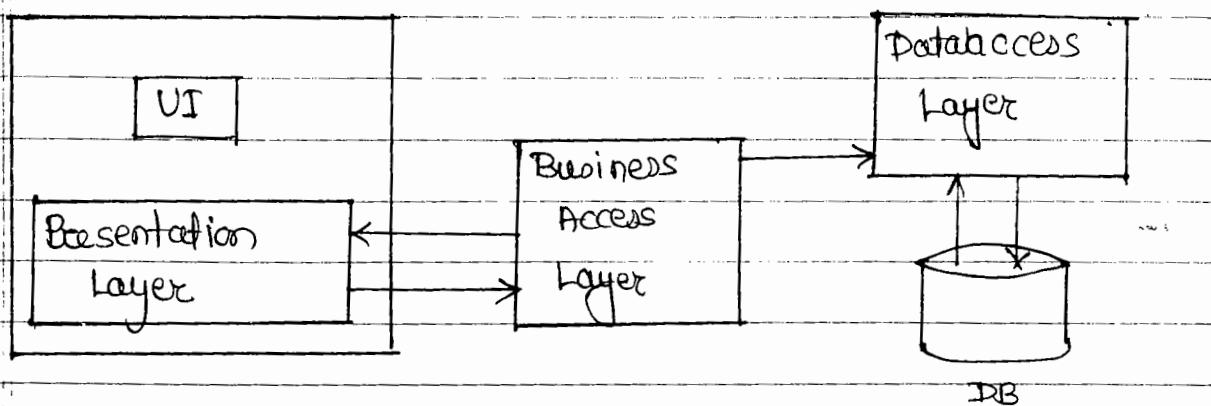


- 2) Two-tier Architecture :-



- The presentation of business logics are maintained separately. The presentation logic is written in presentation layer & business logic is written in business layer or business access layer.
- Here the database logic also can be written in the business layer in only even though this type of architecture is not preferable in the professional projects in the s/w companies.

### 3) Three - tier Architecture :



- The presentation logic, Business logic & database logic are maintain separately this is recommended for the professional project in the s/w companies.

Create table Emp

Create procedure spinsert (@eno int, @name nvarchar(30), @sal int)

as

begin

insert into emp values (@eno, @name, @sal)

end

eno	ename	Sal
-----	-------	-----

1001	Ujwala	5000
------	--------	------

1002	Kshama	5000
------	--------	------

create procedure sp\_delete (@dno int)

as

begin

delete from dept where deptno = @dno

end

Create open vs → goto windows form app.

→ Data Access layer

go to soln explorer → right click on the app →  
add new item → select template " class file" →  
name it as Dalwers.cs → click add button.

using System.Data;

using System.Data.SqlClient;

using System.Configuration;

class Dalwers

{

    keyword                          function

    public static void get\_manipulations(int a, string b, int c)

{

        SqlConnection cn = new SqlConnection ("uid=...")

        cn.open();

        SqlCommand cmd = new SqlCommand ("spinsert", cn);

        cmd.CommandType = CommandType.StoredProcedure;

        cmd.Parameters.Add with value ("@eno", a)

        cmd.Parameters.Add with value ("@name", b)

        (" @sal ", c)

        cmd.ExecuteNonQuery();

}

Business Access Layer :-

goto soln explorer → right click on the app →  
add new item → select template classfile name it  
as Balwers.cs → click ok button.

## class Bal users

{

int eno, sal; right click on eno -

string ename;

public int eno

{

get { return eno; }

set { eno = value; }

{

public int Sal

{

get { return sal; }

set { sal = value; }

{

public string ename.

{

get { return ename; }

set { ename = value; }

{

public void insertdata()

{

Balusers::getmanipulations(eno, name, sal)

{

Presentation layer :-

private void button1\_Click(EventArgs e)

{

Balusers obj = new Balusers();

obj.eno = int.Parse(textBox1.Text);

obj.name = textBox2.Text;

obj.sal = int.Parse(textBox3.Text);

eno :

ename:

Sal :

insert

```
Obj. insertdata();  
messagebox.show (" record is added");  
}
```

5/11/13  
Tuesday

## Collections

- The concept of collections are basically developed from "Arrays"
- Arrays are multiple value container of fixed type
- In order to hold different type of value on the same arrays you required collections.
- Another Advantages of collections is those are dynamic size able . In other words the array is of fixed size & the collection is of dynamic size
- Finally if you do not know how many values are to be stored in the array at the time of its declaration you required to use "collections"
- Net offers some predefined classes for maintenance of collection.

### Collection Classes :-

#### List :-

- It contains 'n' no of values of same type.
- It is a generic class
- This is member of "System.Collection.Generic" namespace.

#### ArrayList :-

- It contain n no of values of different Types
- This is a member of "System.Collection" namespace

## Implementation of ArrayList Class :-

Step 1: Implicit namespace using system.collections.

Step 2: Create Instance

```
ArrayList obj = new ArrayList();
```

Step 3: Add Value.

```
obj.Add (value);
```

Step 4: Get the currently existing no. of values in the collection.

```
obj.Count.
```

Step 5: Get the individual elements in the collection

```
obj [index]
```

Example :-

```
using System.Collections;  
public ArrayList get countries ()  
{
```

```
ArrayList obj = new ArrayList ();  
obj.add ("India");  
obj.add ("china");  
obj.add ("Japan");  
obj.add (1001);  
obj.add (15.555);  
return obj;
```

, further,

List of Items

List Box1  
India  
china  
japan  
1001

Exa:

```
double click on button  
private void button1_Click (----)  
{  
    listBox1.DataSource = getcountries();  
}
```

## Implementation of List Class

Step 1 :- Import namespace

```
using System.Collections.Generic;
```

Step 2 :- Create Instance

```
List < Datatype > obj = new List < Datatype >;
```

Step 3 :- Add Values.

```
obj.Add (values);
```

Step 4 :- Get the currently existing no. of values in the collections

```
obj.Count
```

Step 5 :- Get the individual element in the collection.

```
obj [Index];
```

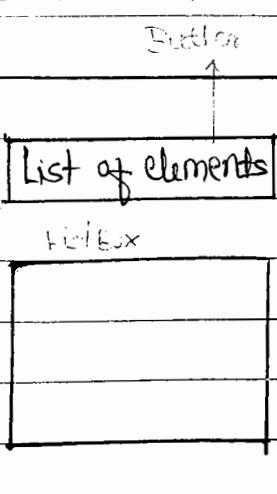
Example - Using System.Collection.Generic :

```
public List < string > mymessages()
```

```
{
```

```
    List < string > obj = new List < string > ();  
    obj.Add ("Good morning");  
    obj.Add ("Good afternoon");  
    obj.Add ("Good evening");  
    return obj;
```

```
}
```



call 1212 for Net

// double click on list of elements .

private void button1\_Click ( . . . )

{

ListBox1.DataSource = my message ();

}

### Collection Initializer :-

As per the previous ex with collections we required to add the values to the collections

using "Add" method . Suppose you want to add so many elements to the collection than if you are using it takes add method . it takes no. of lines of code.

Purpose :-

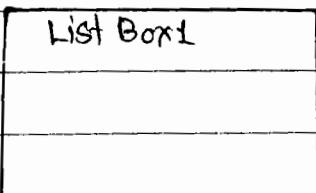
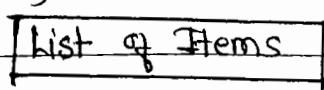
To initialize the element of a collection at the time declaration .

Syntax :-

List < Datatype > obj = new List < Datatype > ()  
{ val1 , value2 , val3 . . . } ;

### ArrayList Collection Class :-

ArrayList obj = new ArrayList { val1 , val2 , val3 }



// double click on List of Items.

= ArrayList obj = new ArrayList c)  
{ 1, 2, 3, 4, "abcd", "exam" };

private void button1\_Click (---)

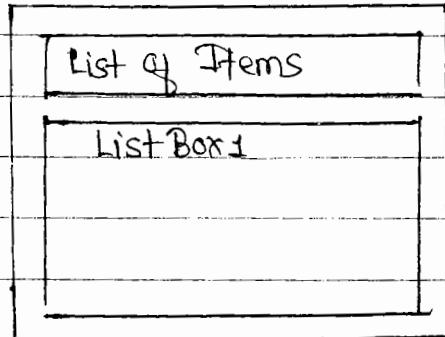
{

foreach (object o in obj)

{

listBox1.Items.Add (o);

}



list < string > obj = new list < string > c) {"exam", "abcd", "ujwala"}

private void button2\_Click (---)

{

foreach (object o in obj)

OR

(string s in obj)

{

listBox1.Items.Add (o);

OR

listBox1.Items.Add (s);

{

}

# Delegates & Events

(using delegates execute methods of events)

- 1) Delegate is a function pointer.
- 2) Using delegate we can store function addresses.
- 3) Delegates are designed by using pointers so delegates are unsafe code.
- 4) The delegate takes argument as well as return type also.
- 5) Based on the function prototype we have to declare the delegates.
- 6) The delegates are classified into 2
  - i) Single Cast Delegate
  - ii) Multicast Delegate.

## i) Single Cast Delegate :

It refers only one function address. After creating delegate instance variable, the instance variable is holding one function address.

## ii) Multicast Delegate

Multicast delegate refers multi single cast delegate, using multicast delegate we can access multiple features from the class.

## Implementation Syntax

- 1) Define the Method :-

access modifier returntype Methodname (arguments)

{

// same code

}

6/11/14  
Wednesday

2) Declare the the delegate (as a data member) in  
delegate returnType delegatename(arguments);

3) Create instance of delegate :-  
delegatename delegate instance name = new delegate name  
(method name);

4) Access the properties of the delegate :-

delegateinstancename.Target

It gets the classname that contain the target Method.

delegate instancename . Method

// gets the signature of the target method.

5) Invoke | Call the target method :-

```
delegate instancename.Invoke();
```

CR delegate instanceName.Invoke(arguments);

~~6/11/13  
wednesday~~

Access specifier Delegate returntype Delegatename(args);

```
public delegate int call methods (int a, int b);
```

See : See

public void m1()

1

public static getdata() public in

1

{

public string getdata() public int sub(int

a, int b)



Example 1 → Example of single cast delegate.

```
public delegate int callmethods(int a, int b);
```

```
call methods obj;
```

```
public int add (int a, int b)
```

```
{
```

```
return a+b;
```

```
}
```

```
public int mul (int a, int b)
```

```
{
```

```
return a * b;
```

```
}
```

```
private void button1_Click (object sender, EventArgs e)
```

```
{
```

```
obj = new callmethods (add);
```

or

```
obj = new callmethods (mul);
```

```
int c = obj (10, 20);
```

```
MessageBox.Show (c.ToString());
```

```
{
```

Example 2 :- public delegate double calculate (int x);

```
public double calculate (int x)
```

```
{
```

```
return 3.14 * x * x;
```

```
}
```

```
private void button1_Click (object sender, EventArgs e)
```

```
{
```

```
CalculateArea obj = new CalculateArea (calculate)
```

```
MessageBox.Show (obj(12).ToString());
```

```
{
```

Output 452.38.

View

dp :- only add fun. is displayed  
mul. is written on the code

OP will display only one  
OP that is multiplication,

Method

Example 3:- Example of multicast delegate.

```
public delegate void call();
```

```
public class sample
```

```
{
```

```
    public void m1()
```

```
{
```

```
        MessageBox.Show("First Method");
```

```
}
```

```
    public void m2()
```

```
{
```

```
        MessageBox.Show("Second Method");
```

```
}
```

```
private void button1_Click(...)
```

```
sample s = new Sample();
```

```
call c1 = new call(s.m1); // single cast
```

```
call c2 = new call(s.m2); // single cast
```

```
call c3 = c1 + c2; // multicast
```

```
c3();
```

```
}
```

#### User Control & Events :-

→ A user control is nothing but user defined control.

→ That means you can create our own control & you can use it in any form where ever req.

→ The user control may contain some user interface with control like button, TextBoxes etc.

→ So, finally you need to design the user interface only once in the user control & you can use it any no. of times in any form.

②

Adv :- Avoids repetition of design & code.

Difference between user control & form.

User Control

Form

- |   |   |
|---|---|
| 1) It is a container for other controls.                                    | 1) It is also a container for other control.                        |
| 2) It can not run individually  | 2) It can run individually  |
| 3) It inherits a predefined class called "System.windows.forms.UserControl" | 3) It inherits a predefined class called "System.windows.form.form" |
| 4) It is meant for reusability  | 4) It is meant for direct execution.                                |

\* Using Windows forms control library template to create our own user controls.

\* It generates DLL file.

\* This DLL file has GUI feature.

\* Create Event handler (own)

Open VS → Select language VC# → and select template → windows forms control library → specify the name & location,

First

Next

Previous

Last

```
public delegate void navicon();  
public event navicon b1_click;  
public event navicon b2_click;  
public event navicon b3_click;  
public event navicon b4_click;
```

```

Public DataTable dt = new DataTable ();
Public DataRow rec;
public int i;

// first Button code
private void button1_Click (----)
{
    class
    {
        form
        {
            rec = dt.Rows [0];
            b1_Click ();
        }
    }

    // Next Button code
    {
        i = i + 1;
        rec = dt.Rows [i];
        b2_Click ();
    }

    // previous button code.
    {
        i = i - 1;
        rec = dt.Rows [i];
        b3_Click ();
    }

    // Last button code
    {
        i = dt.Rows.Count - 1;
        rec = dt.Rows [i];
        b4_Click ();
    }
}

```

(4)

Go to build → Build windows form control library  
DLL file created successfully.

Open visual studio → Select template windows forms app

Go to toolbox → right click on general tab → add  
Tab name it as "my controls" → right click on  
my control tab → choose items → click browse  
button → add the dll file → click "ok" button

Select user control drag & drop in the form.

Form1	<input type="checkbox"/> <input type="radio"/> <input checked="" type="checkbox"/>		
Emp no :	<input type="text"/>		
Ename :	<input type="text"/>		
Sal :	<input type="text"/>		
<input type="button" value="First"/>	<input type="button" value="Next"/>	<input type="button" value="Previous"/>	<input type="button" value="Last"/>

```
using System.Data;
using System.Data.SqlClient;

private void form1_Load(object ...)
{
    SqlConnection cn = new SqlConnection("uid ---");
    SqlDataAdapter da = new SqlDataAdapter("Select * from emp", cn);
    DataSet ds = new DataSet();
    da.Fill(ds);
    usercontrol11.dt = ds.Tables[0];
}
```

ix) public void Display

```
{  
    textBox1.Text = userControl11.Text[0].ToString();  
    textBox2.Text = userControl11.Text[1].ToString();  
    textBox3.Text = userControl11.Text[2].ToString();  
}
```

// go to user control properties → go to events  
double click on b1 click event handler.

Private void Usercontrol - b1 click () -

```
{  
    Display();  
}
```

// go to user control properties click b2 click event  
private void usercontrol - b2 click ()

```
{  
    Display();  
}
```

(5)

## \* Anonymous Types :-

→ In general anonymous types are declared "Var" keyword like all are in lower case

Var a = 100;

Var b = "welcome";

Var c = ss.sss;

→ What we use the datatype in c# like int, double, string, etc. ... are known as "named types"

→ Anonymous types are introduced in 2.0 version of example  
· Net framework in c#.net.

→ Anonymous Types cannot be used as global variable

→ Initializing the data into anonymous type is mandatory

Open visual studio → select template console app.

class program

{

static void main (string [] args)

{

Var a = 100;

Var b = "welcome";

Var c = ss.sss;

Console.WriteLine ("The value of a is :" + a);

Console.Read();

}

example

Example 2 :- static void main (string [] args)

{

Var add = new { hno = "1-1-1111",  
street = "gandhi nagar"  
city = "hyd" };

Console.WriteLine (addr + city);  
 console.ReadLine();

{

### Anonymous Method :-

- Anonymous method is a inline fun.
- Using anonymous method reduce the code.
- Anonymous method used in delegates.

for example:- Public delegate double calculate area  
 (int x);

public double calculate (int x)

{

return 3.14 \* x \* x;

}

private void button1\_Click (---)

{

Calculate obj = new calculate (calculate)

MessageBox.Show (obj(12)).ToString();

}

OR

example:- (Same) by using Anonymous Method

public delegate double calculate area (int x);

private void button1\_Click (object ---)

{

calculate area obj = new calculate area (Delegate(int x))

{

return 3.14 \* x \* x;

});

MessageBox.Show (obj(12)).ToString();

{

7/11/13  
Tuesday

## Lambda Expressions :-

- Lambda expression are used to work on some group of data In general used in aggregate fun.
- A lambda exp. is a fun. without any name calculates & returns single value.
- Lambda exp is represented by " $\Rightarrow$ " (goes to)
- Lambda exp can be divided into 2 categories
  - 1) Expression Lambda
  - 2) Statement Lambda.

### 1) Expression Lambda :-

SI input parameter  $\Rightarrow$  expression ;

### 2) Statement Lambda :-

Input parameter  $\Rightarrow \{$  expression  $\}$  ;

Go to Console App!

```
static void main (String [] args ) {
```

```
    List <int> obj = new List <int> () { 1, 2, 3, 4, 5, 6, 7, 8 }
```

```
    int i = obj . Count (x  $\Rightarrow$  x > 4);
```

```
    Console . WriteLine (" count of greater than 4 : " + i);
```

```
    Console . Read();
```

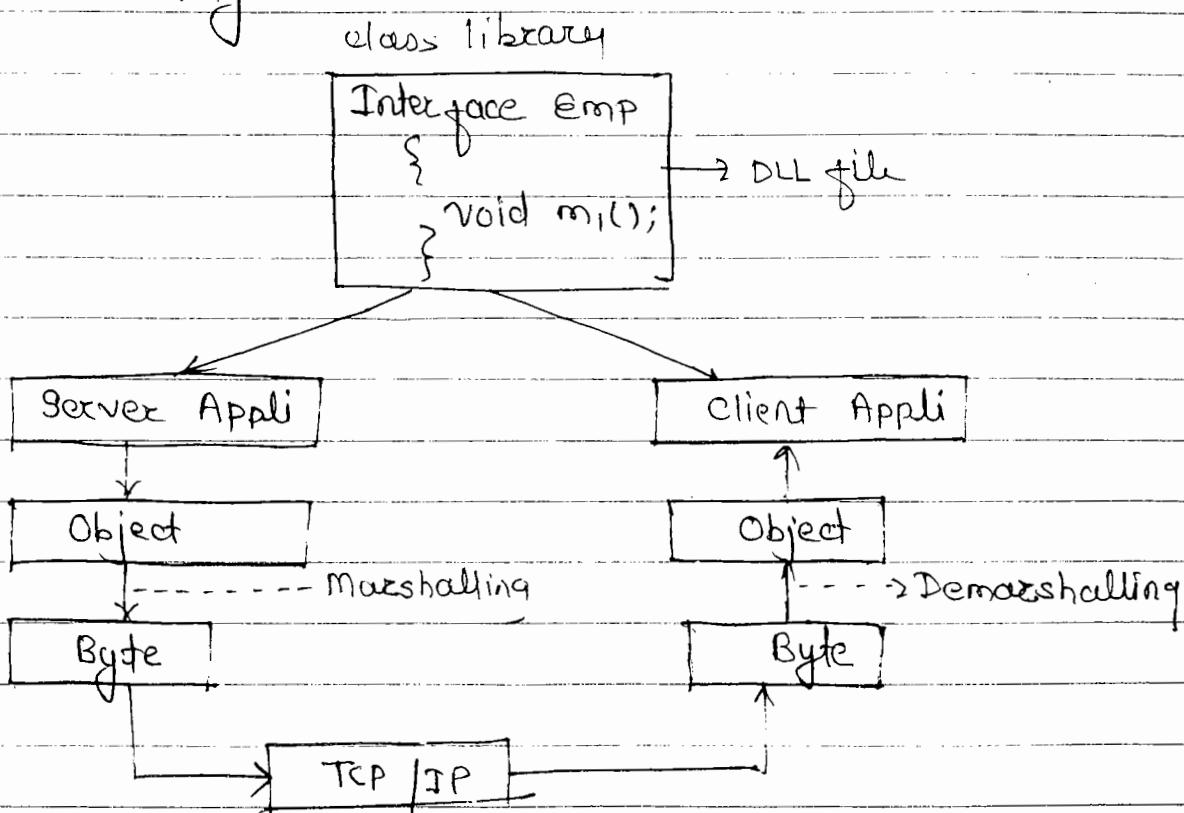
```
}
```

1/1/13  
Thursday

# Remoting

- 1) Remoting is a distributed technology using remoting we can distribute the application features through channels and gRPC parts.
- 2) In Remoting we have to implement
  - 1) Class library.
  - 2) Server application
  - 3) Client application.

## Remoting Architecture



### Class Library :-

- 1) In class library we have to implement interfaces
- 2) the interfaces is having method declarations only
- 3) The class library is giving DLL file
- 4) The DLL file features we are using in the server application as well as in the client application.

## Server application :-

The server application we have to import 4 namespaces.

- 1) System.Runtime.Remoting;
- 2) System.Runtime.Remoting.Channels;
- 3) System.Runtime.Remoting.Channels.TCP;
- 4) user defined namespace;

## Creating server appli class

The server appli class inherits two base classes.

- 1) MarshalByRefObject
- 2) userdefined Interfaces.

Using MarshalByRefObject class we can perform marshalling and unmarshalling.

## Marshalling

By default C# is giving object data the object data does not communicate with channels. so the object format data converting byte format this conversion is marshalling.

The Byte format transfer via protocol.

## Creating channels.

### 1) TCPServerChannel :-

using this class we can create the comm<sup>n</sup> between server app<sup>n</sup> to client app<sup>n</sup>. Every channel refers unique port by default windows application support 9040 port.

2) ChannelServices :-

using this class registering the created channel  
in the server machine.

3) Remoting configuration :-

The Remoting configuration class transfer the  
information service appli to client appli but  
the object format data does not communicate with  
protocols, so the remoting configuration class  
converting the data object format to byte format.  
After that transfer the data through  
Well known object mode

Well known object mode

It is having two constant values.

- 1) Singlecall
- 2) Singleton

1) Singlecall :-

In this constant value every request will creating  
one new object via the new object transfer the  
information.

2) Singleton :-

In this constant value 1st request will creating  
1 new object second request onward using same object  
memory.

Client Application :-

The client appi importing 4 na  
System. Runtime. Remoting  
System. Runtime. Remoting. channels  
System. Runtime. Remoting. channels. TCP  
user defined namespace.

Activator :-

using this class we are calling Remote methods  
client appi to server appi as well as performing  
demarshalling.

Class Library.

Open visual studio → select template class library  
specify name & location → click ok button

namespace ClassLibrary1

{  
public interface Emp

{

string getname();

}

go to Build - build class library

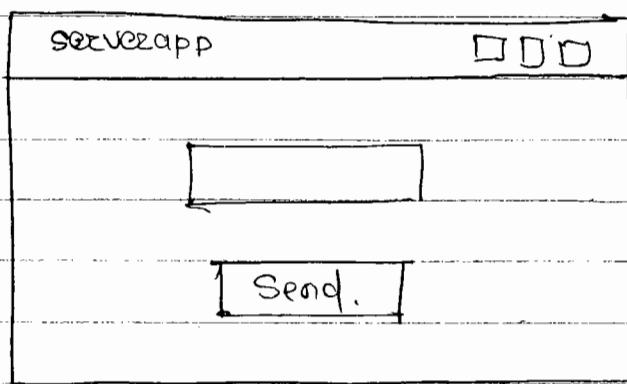
Result - DLL file created successfully.

Server application.

Open visual studio - select template → windows form applic → specify name of doc

go to sol<sup>n</sup> explorer right click on the references add reference, → go to assemblies activate → System.Runtime.Remoting checkbox click ok button.

go to sol<sup>n</sup> explorer - right click on project → add reference click browse button add user dll file → click ok button.



using System.Runtime.Remoting;  
using System.Runtime.Remoting.Channels;  
using System.Runtime.Remoting.Channels.TCP;  
using classlibrary1;

string s;  
public class serverapp : MarshalByRefObject, Emp  
{

public string gettext()  
{  
 return s;  
}

private void form1\_Load()

{

TcpServerChannel cha = new TcpServerChannel(9040);

channelServices.RegisterChannel(cha);

Remoting configuration • RegisterWellKnownServiceType

(type of (scrapp), "ref", WellKnownObjectMode.SingleCall)

}

private void button1\_Click()

{

s = textBox1.Text;

}

### Client Application

Open visual studio → select lang visual c#  
& select complete windows forms application.

go to soln explorer — right click on the references  
add reference go to assembly activate  
System.Runtime.Remoting checkbox.

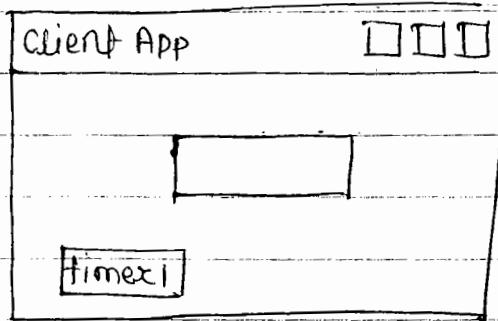
go to soln explorer - right click on the references  
add references - click browse button - select all file  
click ok button

go to toolbox select timer control drag & drop in  
the form.

8/11/15  
Friday

go do timer control property → enable = true 1000ms = 1s

go do timer event - double click on the event.



using System.Runtime.Remoting;

using System.Runtime.Remoting.Channels;

• Tcp

using ClassLibrary1;

private void timer1\_Tick

{

    emp xm = (Emp)Activator.GetObject(typeof(Emp), "tcp://  
    localhost:9040/xm");

    textBox1.Text = xm.GetText();

}

# LINQ

- 1) LINQ stands for Language Integrated Query.
- 2) This concept is introduced in .NET Framework is .NET 3.5.
- 3) This is a "query writing technology".
- 4) This is most useful while working large amount of data in the live projects.

## Introduction :-

- 1) In relational database system data is organized in the form of tables, on which you can write SQL queries to retrieve the required data according to the requirement in the application.
- 2) But you can't write a query on the non-database data, which is in the form of objects in the application. There, you can write the queries using the new concept called "LINQ".
- 3) You can write a queries on arrays, object, database & XML using LINQ.
- 4) Note : Before writing the LINQ queries you should import the "System.Linq" namespace ---

LINQ syntax :-

from ... in ... let ... where ... orderby ...  
select ... groupby ...

Def of clause : Clause means.

" A part of the query"

The above syntax consist of 7 clauses.

- 1) From clause
- 2) In clause
- 3) Let clause
- 4) Where clause
- 5) Order by clause
- 6) Select clause
- 7) group clause

Mandatory clauses :-

- 1) From clause
- 2) In
- 3) Select

Understanding clauses .

1) From clause :- This is used to specify the iteration variable name. This acts as alias name for the datasource.

2) In clause

This is used to specify the main database source for the query .

3) Select clauses - Let clause (optional)

This is used to declare a new identifier with a value that is to be used during the query execution.

4) Where clause :-(optional)

This is the most frequently used optional clause using which can specify the condition in the query

### 5) Orderby clause (optional)

This is used to specify the sorting expression if required.

### 6) Select clause

This is used to specify the object, which is required in the query results.

### 7) Group by (optional)

This is similar to "groupby" clause in SQL

This retrieve grouped data, base on a column.

In general when we deal with object oriented programming we learn opp in C# / java.

But especially when we deal with database again we need to learn database languages, like SQL, TSQL, PLSQL, when dealing with sqlserver / oracle and xpath / xquery etc when dealing with "xml" ie we also need to learn database programming languages to deal with any databases like ...

LINQ will bridge this gap between OOP's and relational databases.

Linq is known as O-R mapping tool for .net  
In java we have hibernate as O-R mapping tool.

Linq will make object oriented programming

where we can work with queries in Linq we need to follow 3 steps.

# LINQ

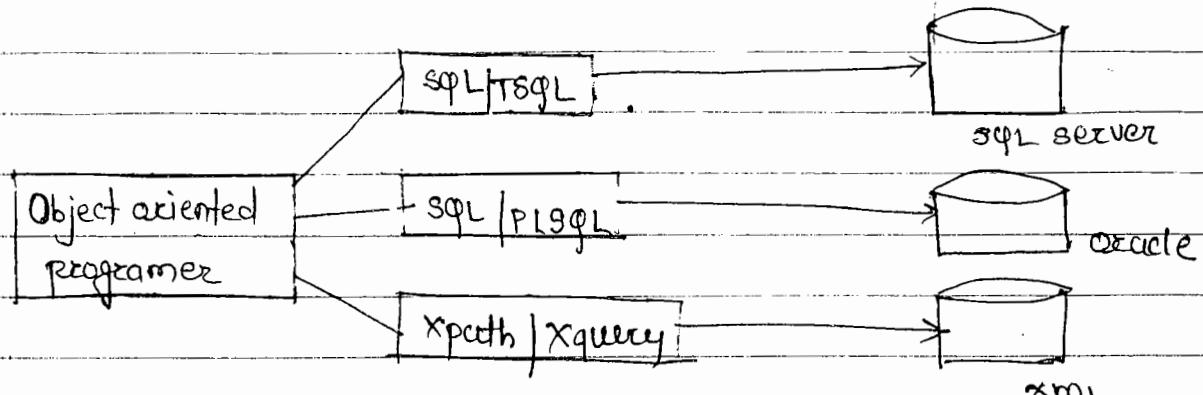
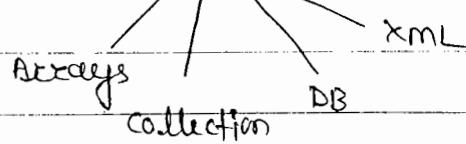
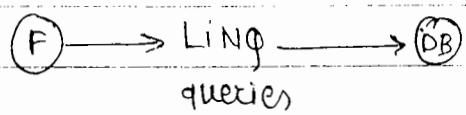
LINQ - 3.5 version

- 1) Preparing datasource
- 2) writing query
- 3) executing query.

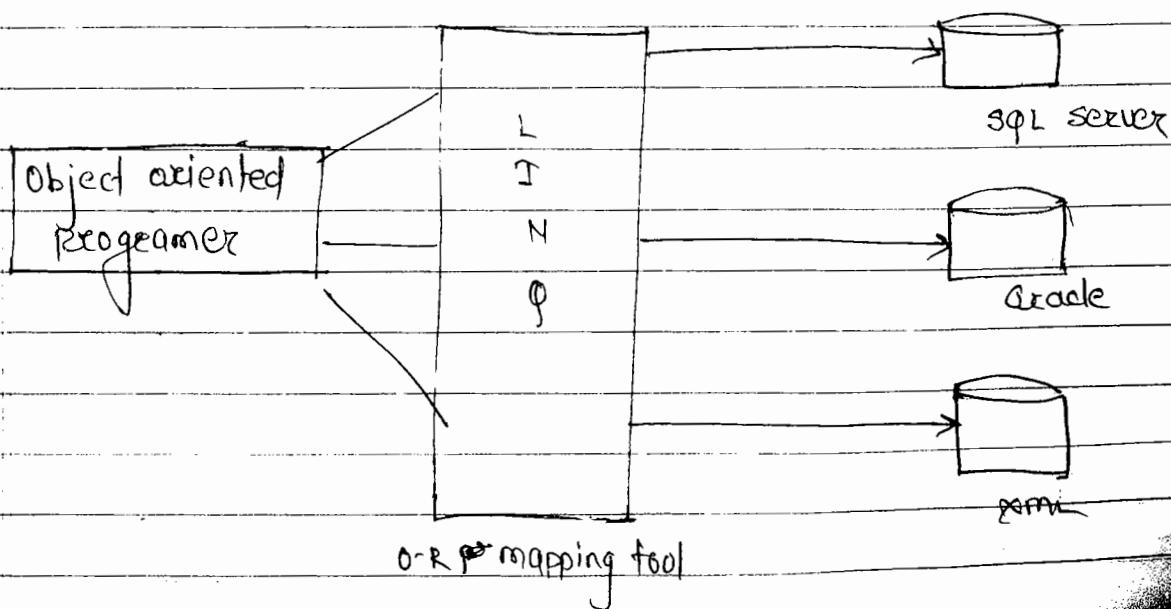
LINQ

Technology - Libraries.

Database.



LINQ is a O-R mapping tool.



## Syntax - LINQ

```
var x = from n in {callc}  
select n;
```

LINQ to object :

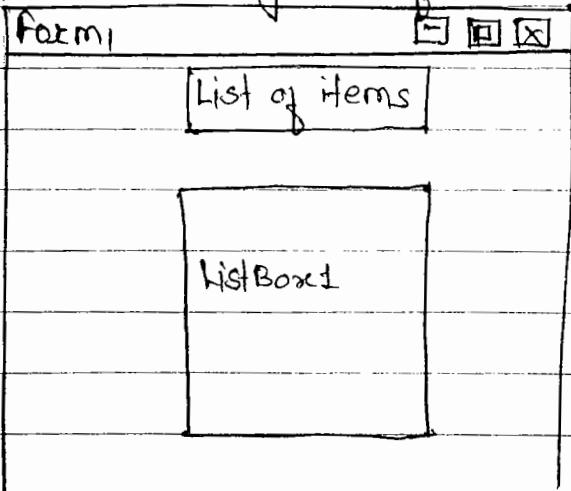
LINQ to classes :

LINQ to SQL :

LINQ to XML :

LINQ to Object :-

Design a form.



```
List < int > obj = new List < int > () { 1, 2, 3, 4, 5, 6, 20, 22, 42, 43 };
```

```
private void button1_Click (object
```

```
{
```

```
    var x = from n in obj  
    where n > 20
```

```
    select n;
```

```
    foreach ( var num in x )
```

```
{
```

```
        listBox1.Items.Add (num);
```

```
}
```

## LINQ to Classes

go to solution explorer → right click on the application  
add new item → select template class file name it as  
student.cs → click add button.

### Class student

{

public int Sno

{

get ;

set ;

{

public string Sname .

{

get ;

set ;

{

public string grade

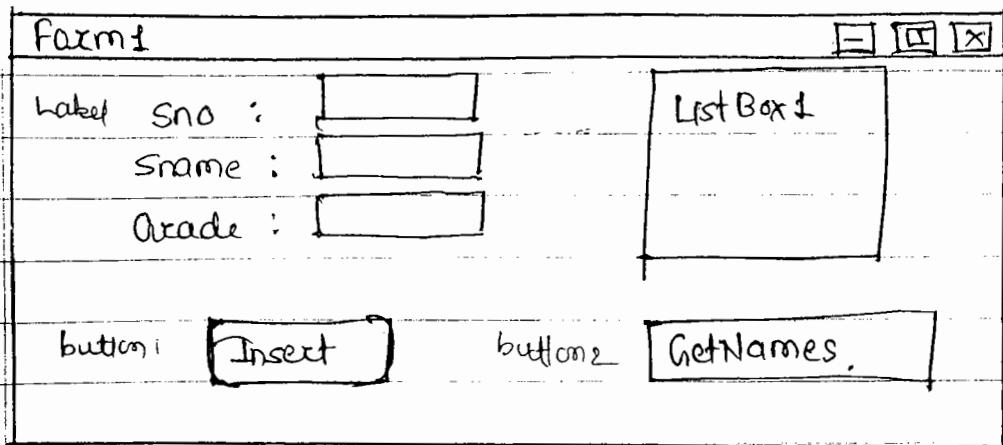
{

get ;

set ;

{

12,45



list < student > stuobj = new list < student >();  
 private void button1\_Click (object sender

{

student obj = new student ();  
 obj.Sno = int.Parse (textBox1.Text);  
 obj.Sname = textBox2.Text;  
 obj.Grade = textBox3.Text;  
 stuobj.Add (obj);

MessageBox.Show ("Student Added");

}

private void button2\_Click

{

var x = from n in stuobj  
 select n;

foreach (student s in x)

{

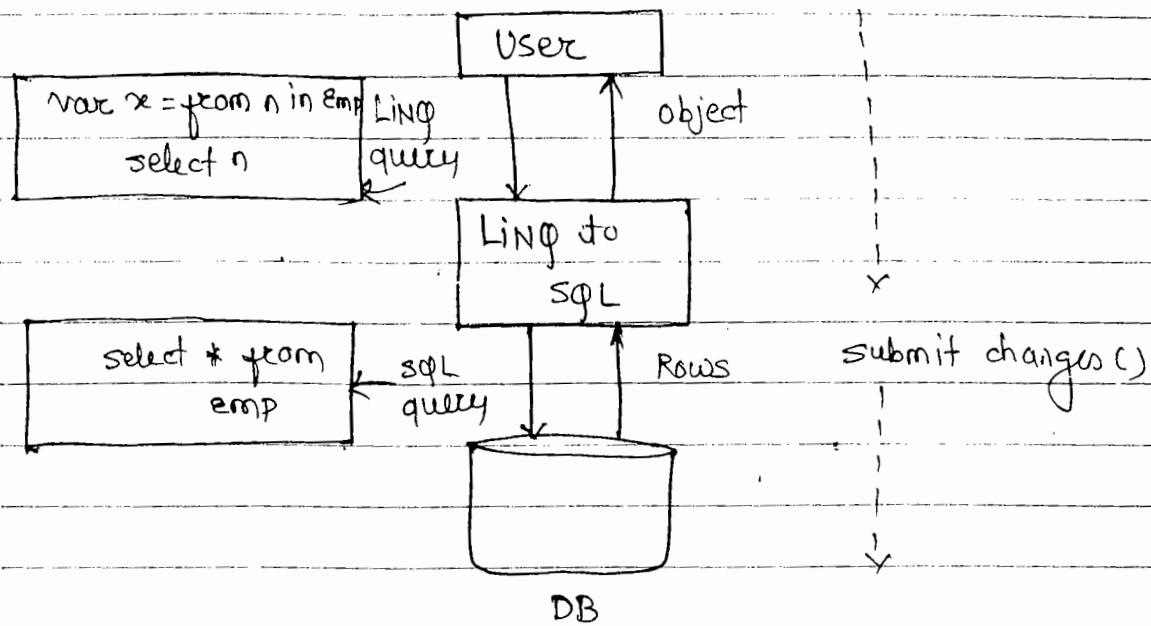
listBox1.Items.Add (s.Sname);

}

}

Linq to SQL

Linq to SQL Architecture.



Go to solution explorer → right click on application  
→ add new item → go to left side panel & select  
data option → in that select template → Linq to SQL  
classes → name it as Demo.dbml (Database markup  
Lang) click add button.

Go to menu bar → go to view → go to server  
explorer → right click on data connection → add  
connection → Enter server name → activate SQL  
serve authentication Radio button → enter login id  
enter password → enter database name click ok  
button.

go to tables → select required table → drag &  
drop in the object relational designer form of  
Linq to SQL classes click save button

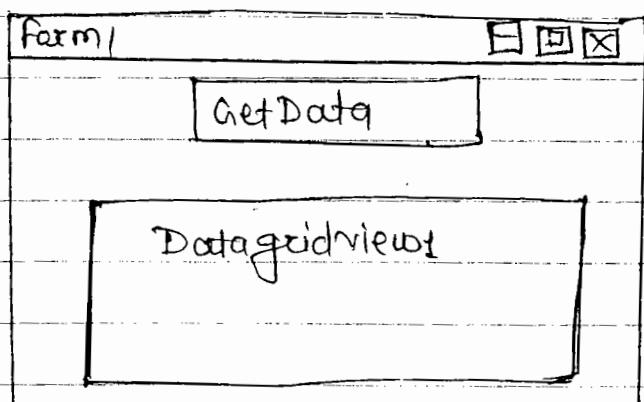
4/11/13  
Mon

## Note

Then automatically table name treats like a class  
& column name treats like property.

go to windows form.

go to tool box → select datagridview control  
drag + drop in the form.



private void button1\_Click (object sender, EventArgs e)

{

DemoDataContext obj = new DemoDataContext();

var x = from n in obj.emps property  
select n;

datagridview1.DataSource = x;

}

11/11/13  
Monday

## Linq to Insert, Update, Delete statement

Form1	
Empno :	<input type="text"/>
Ename :	<input type="text"/>
Salary :	<input type="text"/>
<input type="button" value="Insert"/>	

Go to sal<sup>n</sup> exp → right click on app<sup>n</sup> → add new item → select template Linq to SQL classes name it as sample.dbml — click add button.

e) Go to view → go to server explorer → right click on data connections → add connection enter server name activate SQL server authent radio button - Login id sa enter password + enter database name

go to tables → select required table → drag & drop in the object relational designer form of Linq to SQL classes — click save button

private void button1\_Click

{

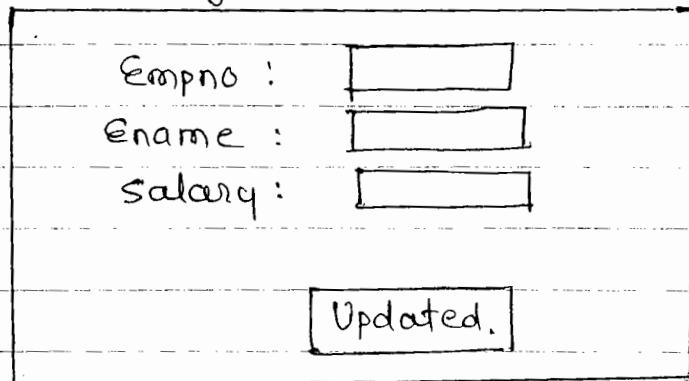
```
SampleDataContext obj = new SampleDataContext  
emp e1 = new emp();  
e1.empno = int.Parse(textBox1.Text);  
e1.ename = textBox2.Text;  
e1.sal = int.Parse(textBox3.Text);
```

```

    property           meth
obj . emps. InsertOnSubmit (e1) ;
obj . SubmitChanges ();
MessageBox . Show (" Record added ");
}

```

I want to update a data from front end to back end using LINQ.



private void button1\_Click

```
{
```

```
SampleDataContext obj = new SampleDataContext();
```

```
var x = from n in obj . emps
```

```
where n . empno == Int . Parse (textBox1 . Text);
```

```
Select n ;
```

```
if (x . Count () == 0)
```

```
{
```

```
MessageBox . Show (" no such record ");
```

```
}
```

else

```
foreach (var rec in x)
```

```
{
```

```

    rec.ename = textBox2.Text;
    rec.Sal = int.Parse(textBox3.Text);
    obj.SubmitChanges();
    MessageBox.Show("Record updated");
}

```

Delete code.

Empno :	<input type="text"/>
ename :	<input type="text"/>
sal :	<input type="text"/>
Delete	

```

private void button1_Click
{
    SampleDataContext obj = new SampleDataContext();
    var x = from n in obj.emps
            where n.empno == int.Parse(textBox1.Text);
    Select n;
    if (x.Count() == 0)
    {
        MessageBox.Show("No such record");
    }
    else
    {
        foreach (var rec in x)
        {
            obj.emps.DeleteOnSubmit(rec);
            obj.SubmitChanges();
            MessageBox.Show("Record Deleted");
        }
    }
}

```

find code

Empno :	<input type="text"/>	<input type="button" value="Find"/>
Ename :	<input type="text"/>	
Sal :	<input type="text"/>	

private void button1\_Click --

{

SampleDataContext obj = new SampleDataContext();

var x = from n in obj.emps

where n.empno == int.Parse(textBox1.Text);

Select n;

if (x.Count() == 0)

{

messageBox.Show("no such record");

}

else

foreach (var rec in x)

{

textBox2.Text = rec.ename;

textBox3.Text = rec.sal.ToString();

}

Linq to stored Procedure.

Create procedure spinsert

(@eno int , @name varchar (30) , @sal int)

as

begin

insert into emp values (@eno , @name , @sal)

end .

go to server explorer → goto

data base → go to stored

procedures → select required

stored procedure drag & drop

in the right pan of linq to sql classes

click save button .

EMPNO :

ENAME :

SAL :

Insert

Insert Button code :

private void button1\_Click

{

```
SampleDataContext obj = new SampleDataContext();
obj.spinsert(int.Parse(textBox1.Text), textBox2.Text
int.Parse(textBox3.Text));
```

MessageBox.Show ("record added");

}

मुख्य स्पार

view - tool box

VB 6.0

ઉન્નતિ

VC++

અની લ

.Net

ANIL

VB .Net

pkg C# .Net

System.out.println("i value : " + i);

System.console.WriteLine("i value : " + i);

[www.w3schools.com](http://www.w3schools.com),

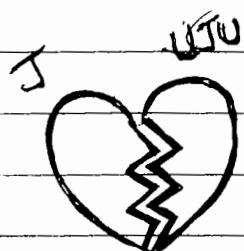
View

public void double click (JLabel obj)

int i = 1; i <= 10; i++)

System.out.println("i value : " + i);

for (int i = 1; i <= 10; i++)



## Example of 3 tier Architecture

1

Create table dept

Create procedure spinsert (@dno int, @name varchar(30),  
@loc varchar(30))

as

begin

insert into dept values (@dno, @name, @loc)

end.

Delete - Create procedure spdelete (@ dno int)

as

begin

delete from dept where deptno = @dno

end.

go to soln exp → right click on app add new item →  
select template application configuration file click add  
button.

<configuration>

<appSettings>

<add key = "conn"

value = "uid = sa; database = mydb; password = 123  
server = nit-pc"/>

</appSettings>

</configuration>

## Data Access Layer

go to soln exp → right click on app → add new item  
select template class file name datusers.cs → click  
add button.

```
using System.Data;  
using System.Data.SqlClient;  
using System.Configuration;
```



class Datas

{

public static void getmainpulations (int a, string b, string c)

{

sqlconnection cn = new sqlconnection (configurationSettings.AppSettings["conn"]);  
cn.open ();

SqlCommand cmd = new SqlCommand ("spinsert", cn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue ("@dno", a);

cmd.Parameters.AddWithValue ("@name", b);

cmd.Parameters.AddWithValue ("@loc", c);

cmd.ExecuteNonQuery ();

}

public static void Deletedata (int a)

{

sqlconnection cn = new sqlconnection ("configurationSettings.AppSettings["conn"]);

cn.open ();

SqlCommand cmd = new SqlCommand ("spdelete", cn);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue ("@dno", a);

cmd.ExecuteNonQuery ();

}

public static DataSet getdeptdata ()

{

sqlconnection cn = new sqlconnection ("configurationSettings.AppSettings["conn"]);

cn.open ();

SqlDataAdapter da = new SqlDataAdapter ("spgetdata", cn);

DataSet ds = new DataSet ();

```
da.Fill(ds);
return ds;
}
```

go to soln explorer right click on app → add new item,  
select complete → add new it class file → Baluvers.cs  
click add button.

```
namespace WindowsForms
{
    class Baluvers
    {
        int dno;           // right click on dno - refactor
        string name, location;
        public int dno
        {
            get { return dno; }
            set { dno = value; }
        }
        public string name
        {
            get { return name; }
            set { name = value; }
        }
        public string location
        {
            get { return location; }
            set { location = value; }
        }
        public void insertdept()
        {
            Baluvers.getmanipulations(dno, name, loc
        }
    }
}
```

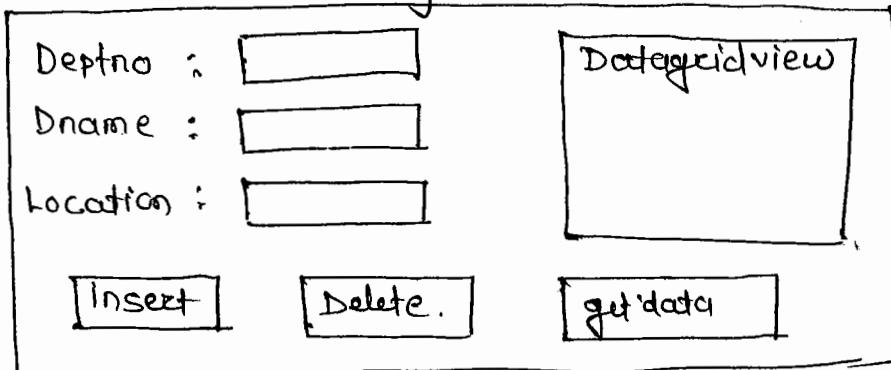
```

public void deletedept()
{
    datusers.DeleteDept(dno);
}

public System.Data.DataSet DeptData()
{
    return datusers.getDeptData();
}
}
}

```

### Presentation Layer



// insert button code.

```

private void button1_Click(object
{
    Balusers obj = new Balusers();
    obj.Dno = int.Parse(textBox1.Text);
    obj.Name = textBox2.Text;
    obj.Location = textBox3.Text;
    obj.InsertDept();
    MessageBox.Show("record added");
}

```

```

private void button2_Click
{

```

```

    Balusers obj = new Balusers();
    obj.Dno = int.Parse(textBox1.Text);
}
```

```
    obj.deleteDept();  
    MessageBox.Show ("Record deleted");  
}
```

// get data button code.

```
private void button3_Click
```

```
{
```

```
    BalUsers obj = new BalUsers();
```

```
    DataSet myds = obj.getDeptData();
```

```
    dataGridView1.DataSource = myds.Tables[0];
```

```
}
```

## \* How To stored Images in the DataBase.

4

Create table employee (empno int, ename varchar(30),  
sal int, photo image, addr varchar(30))

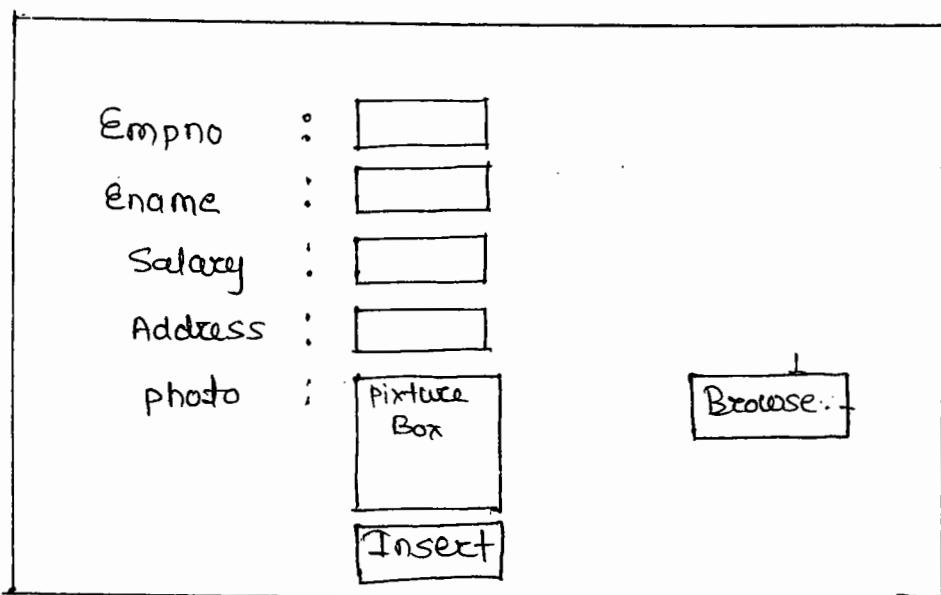
Create procedure spinsertemp (@eno int, @name varchar(30),  
@sal int, @photo image, @addr varchar(30))

as

begin

insert into employee values (@eno, @name, @sal,  
@photo, @addr)

end



~~private void button1\_Click~~  
~~SqlConnection cn = new SqlConnection ("uid = sa ;~~  
~~database = master , password = 123 ; server = nitpc");~~  
~~cn.Open ();~~  
~~SqlCommand cmd = new SqlCommand ("spinsertemp ", cn);~~  
~~cmd.CommandType = CommandType.StoredProcedure~~

go to tool box → go to dialog box → select open dialog control, drag & drop in the form.

```
string imgpath  
// browse button code  
private void button1_Click (object ---)  
{  
    openFileDialog1.ShowDialog();  
    imgpath = openFileDialog1.FileName;  
    pictureBox1.ImageLocation = imgpath;  
}
```

go to picture box control prop → select prop size  
more in the drop down list → select stretched element

~~12/11/13 Tuesday~~

```
using System.Data.SqlClient;  
using System.IO;  
  
string imgpath;  
SqlConnect cn;  
SqlCommand cmd;  
byte [] data;  
  
private void button2_Click (object ---)  
{  
    cn = new SqlConnection ("uid =sa; database =master,  
                           password =123; server =nit-pc");  
    cn.Open ();  
    cmd = new SqlCommand ("SpInsertemp", cn);  
    cmd.Parameters
```

```

cmd.CommandType = CommandType.StoredProcedure;
cmd.Parameters.AddWithValue("@eno", textBox1.Text);
cmd.Parameters.AddWithValue("@name", textBox2.Text);
cmd.Parameters.AddWithValue("@sal", textBox3.Text);
cmd.Parameters.AddWithValue("@addr", textBox4.Text);

// converting image format to byte format
data = File.ReadAllBytes(imgpath)
cmd.Parameters.AddWithValue("@photo", data);
cmd.ExecuteNonQuery();
messageBox.Show("Procedure executed");
}

```

✓ Retrieving Images Backend to front end.

Create procedure spgetdata2 (@eno int)  
as  
begin  
Select \* from employee where empno = @eno  
end

Empno	<input type="text"/>	[Find] - button.
Ename	<input type="text"/>	
Salary	<input type="text"/>	
Address	<input type="text"/>	
photo.	<input type="text"/>	

using System.Data.SqlClient;  
using System.IO;  
using System.Data;

```
private void button1_Click(object sender, EventArgs e)
{
    SqlConnection cn = new SqlConnection("uid = sa; password=123;
cn.open();      server = nit-pc ; database = master");
    SqlCommand cmd = new SqlCommand ("spgetdata2", cn);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.AddWithValue ("@eno", textBox1.Text);
    SqlDataAdapter da = new SqlDataAdapter (cmd);
    dataset ds = new dataset();
    da.Fill (ds);
    if (ds.Tables [0].Rows.Count > 0)
    {
        textBox2.Text = ds.Tables [0].Rows [0] [1].ToString();
        textBox3.Text = ds.Tables [0].Rows [0] [2].ToString();
        textBox4.Text = ds.Tables [0].Rows [0] [3].ToString();
        byte [] data = (byte [])ds.Tables [0].Rows [0] [4].ToString();
        memory stream ms = new memory stream (data);
        pictureBox1.Image = image.fromstream (ms);
    }
}
```

## \* Exception Handling

- 1) An Exception means runtime error.
- 2) The process of handling the runtime exceptions is called as exception handling.

### Types of Errors :-

1) Compile time error

2) Runtime error.

#### 1) Compile time error :-

The error occurs after compiling the program are called as compile type error.

#### 2) Runtime error :-

The errors occurs during the execution of the program are called as runtime errors.

### Overview of Exception handling :-

- 1) The exception may occur at runtime based on the mistake of the user or program or system problem also.
- 2) When exception is raised automatically it leads to "abnormal application termination".
- 3) As a part of this exception handling the program has to display particular error message to the user.
- 4) Purpose of exception handling is to avoid is "abnormal app termination" even though exception occurs.

## Types of Application termination :-

### 1) Normal Application

1) whenever the program execution control executes all the statement in the program & reaches to end of the code the appn will be terminated automatically, it can be called as normal appn termination.

### 2) Abnormal App

whenever an exception occurred at runtime the application will be terminated automatically, it can be called as abnormal App.

If the Appi was terminated abnormally it will be most inconvenient for the user. so the programmer responsible to avoid that kind of abnormal application termination, even though exception is occurred at run time.

### A Simple Demo on Exception.

open v.s → select lang visual c# & select template console app → click ok button.

```
static void main (string [] args)
{
    string [] cities = {"hyd", "Delhi", "Chennai", "banglore"};
    Console.WriteLine (cities [4]);
    Console . Read ();
}
```

Run the application.

In the above code raised the error because because it is trying to access an error element which is an out of range of the array. so it leads to abnormal App termination at runtime.

To avoid this we have to implement exception handling for this code.

Syntax of exception handling.

```
try  
{  
---;  
---;  
}  
}
```

Catch (exception ex)

```
{  
}  
}  
finally  
{  
---;  
---;  
}
```

In the above syntax we can observe to blocks.

- i) try block
- ii) catch block
- iii) finally block

#### i) Try Block :-

- 1) The try block contains the actual code which is to be executed.
- 2) After every try block there should catch block without fail.
- 3) The system tries to execute the code in the try block.
- 4) During the execution if any exception occurs then the execution control automatically goes to catch block

5) At the same time the try block throws the exception to the catch block in the form of an object. That object is called as "Exception object".

#### ii) Catch Block :

- 1) This is also known as error handler
- 2) This is followed the try block
- 3) The catch block will be executed if any exception is occurred during the execution of try block.
- 4) The catch block contain necessary code which displays an error message to the user.
- 5) This receives the exception thrown by the try block in the form of an object.

In the above syntax ex is the exception object.  
The "exception" is the class for the exception object.

### iii) finally block :-

- 1) This block will be executed automatically after executing try block or catch.
- 2) That means even though exception is raised or not raised then the finally block will be executed without executed.
- 3) This is optional block. you can write the exception handling syntax only with try and catch block without finally block.

Example :-

Select Country	<input type="button" value="▼"/>	comobox
Select state	<input type="button" value="▼"/>	

CId	cName
1	-
2	-
3	-

State	SId	sName	CId
	1001	-	-
	1002	-	-
	1003	-	-
	1004	-	-

13/11/13  
Wednesday

## Pemo Exception Handling.

go to Console Application.

```
static void Main (string [] args)
```

```
{
```

by try

```
{
```

```
string [] cities = {"chennai", "Hyd", "Delhi", "Nagpur", "Pune"}
```

```
Console.WriteLine (cities [2]);
```

```
Console.WriteLine (cities [4]);
```

```
}
```

```
catch (Exception ex)
```

```
{
```

```
Console.WriteLine (ex.Message);
```

```
}
```

finally

```
{
```

```
Console.WriteLine ("This is finally block");
```

```
(Console.Read());
```

```
}
```

```
Console.Read();
```

```
}
```

### Example

Enter A Value :	<input type="text"/>
Enter B value :	<input type="text"/>
Result :	<input type="text"/>
<input type="button" value="View"/>	

```

private void button1_Click (object
{
    int a, b, res = 0;
    try
    {
        a = int.Parse(textBox1.Text);
        b = int.Parse(textBox2.Text);
        res = a / b;
    }
    catch (Exception ex)
    {
        MessageBox.Show("ex.Message");
    }
    finally
    {
        textBox3.Text = res.ToString();
    }
}

```

### Types of Error message.

The catch block generates an error message when an exception occurs, that error message can be of two type

- 1) User defined message
- 2) System Defined message.

## 1) User Defined Message :-

your own message can be written  
ex error occurred, operation is not successful etc.

## 2) System Defined Message :-

The system provides the description of error so that you can point that output & directly do access the System defined msg, u can use the exception object as follows  
Syntax - ex.Message.

## Exceptional Classes.

- 1) C# Recognizes the exception as an object
- 2) To declare the exception object we have used a class called System.Exception in the previous examples.
- 3) The Exception class recognises any type of exception in order to catch the particular type of error C# provides other exceptional classes. some of them are given here,

S.No	Exceptional class	Description.
1	System.OverflowException	Occurs when a large value is assigned to a variable, which is not fit in that variable.
2	System.FormatException System.InvalidCastException	Occurs when the casting is failed from one data type of another data.
3	System.DivideByZeroException	Occurs when any number is divided by 0.
4)	System.IndexOutOfRangeException	Occurs when an index is accessed in out of range.
5	System.Finalize	

5	System .InsufficientMemoryException	Occurs when there is no sufficient memory in RAM for the exception of the application.
6.	System .Io .FileNotFoundException	Occurs when a non existing file is accessed.
7	System .Io .DirectoryNotFoundException	Occurs when a non-existing directory is accessed.
8	System .Io .FileLoadException	Occurs when any error occurred during the opening of any file.
	System .Io .IoException	Occurs when any error occurred during the read or writing.
10	System .Threading .ThreadInterruptedException	Occurs when any error occurred during the exception of the thread.
11	System .Threading .ThreadStartException.	Occurs when any error occurred while starting the thread.
12	System .InvalidOperationException	Occurs when any error occurred while opening the database connect.
13	System .Data .OleDb .OleDbException	Occurs when any error occurred while performing query at non query transaction on OleDb database.
14	System .Data .SqlClient .SqlException.	Occurs when any error occurred while performing query at non query transaction on SqlServer databases.
15	System .EntryPointNotFoundException	Occurs when you try to run the app without defining any entry point (main() method).

## 16 System.InvalidTime ZoneException

Occurs when the system has an invalid time zone setting the date of time settings.

Note :- If you want to handle more than one type of exception for the same try block then you need to write multiple catch blocks.

### Demo on Multiple Catch Blocks

go to console application & write the code.

```
static void main (string [] args)
{
    int n1, n2, n3;
    try
    {
        Console.WriteLine ("Enter first number:");
        n1 = Convert.ToInt16 (Console.ReadLine ());
        Console.WriteLine ("Enter Second number:");
        n2 = Convert.ToInt16 (Console.ReadLine ());
        n3 = n1 / n2;
        Console.WriteLine ("The result is :" + n3);
    }
    catch (DivideByZeroException ex)
    {
        Console.WriteLine ("This is Dividebyzeroexception");
    }
    catch (formatOverflowException ex)
    {
        Console.WriteLine ("This is Invalid exception");
    }
}
```

```
9  
catch  
{  
    Console.WriteLine("This is over flow exception");  
}  
Console.Read();  
}
```

## Crystal Reports

- Crystal Report is known as Reporting tool
- This is used for development of database reports in the projects.
- The report file extension is ".RPT" which can be exported to Excel, word, PDF
- Crystal Reports are developed by "Siegele Corporation" and has strong integration with visual studio.
- In VS studio 2010 or 2012 the crystal reports are not available by default we need to install it separately. In the older version (visual studio 2005 & 2008) crystal report are Inbuilt
- Every report when open has 5 section in it
  - 1) Report header
  - 2) Page header
  - 3) Details
  - 4) Report footer
  - 5) Page footer

## Report Header

Content placed under this section gets displayed on top of the report i.e. on top of the first page in report.  
eg. Company Name, Address etc.

## Page Header :-

Content placed under this section gets displayed on top of every page into which the report extends.  
eg. Column Names.

## Details :-

Content into this section generally comes from the DB, this is the actual information that has to be presented to clients.

## Report Footer :-

This is same as Report header but gets displayed on bottom of the report.  
eg signature.

## Page footer

This is the same as Page Header but gets displayed on bottom of every page into which the report extends.  
eg - Page No's.

Open visual studio → select template windows forms app →  
go to solution explorer → right click on application → select  
category → Reporting → and select template → Crystal report  
name it as employee data.RPT click add button.

Select standard click ok button

Now to create new connection → go to microsoft OLEDB ADO →  
select microsoft OLEDB provider for SQL Server → click next  
→ enter server name → Login id sq → password 123 →  
enter database name → click next & click finish.

field exp - C + ALT + T

Go to Database → go to DBO → go to tables → select any table → click next → select fields → click next → next → next → finish.

Go to tool box → go to reporting tab → select crystal report viewer control → drag & drop in the form.

Go to crystal report viewer control properties → select property report source in that drop down list → select crystal report employeedata run the app → click → export report button → select save as type → select any format specify the filename → click save button.

### Example 2

Go to SQL explorer → right click on the application → add new item → select category → reporting → select template → crystal report → click add button →

Activate as a blank report radio button click ok button  
go to field explorer (C + ALT + T) → go right click on database fields → go to database expert → go to create new connection → go to microsoft OLEDB ADO → select microsoft OLEDB provider for SQL server → click next → enter server name → user id = sa, password = 123 → enter database name → click next & finish.

go to DBO select required table click ok button

go to report header section → go to tool box → select text object control drag & drop in report header section → name it as department report.

Report header right side → right click insert special field print date & time.

go to field explorer go to database field → go expand dept → select fields → drag & drop in the details section

go to report footer → right click insert line

go to page footer → right click insert special field → print page no click save button.

go to toolbox → select crystal report viewer control  
drag & drop in the form  
go to crystal report viewer control property → select  
property report source in dropdown list select deptloct  
run the application.

### \* Retrieving Data from multiple table and generate the report.

Create procedure spgetempdept  
as

begin

Select e.empno, e.ename, e.sal, d.deptno, d.dname, d.loc  
from emp e inner join dept d on e.deptno = d.deptno  
end

go to soln exp. → right click on app add new item →  
select category reporting →

Select template crystal report click add button.

go to new connection → go to ms OLEDB → select ms provided  
OLEDB server → click next → enter server name sq login  
id - password → enter database enter Next & finish

go to DBO → go to stored procedure → select required  
procedure - click next → select field → click next →  
next → next & finish.

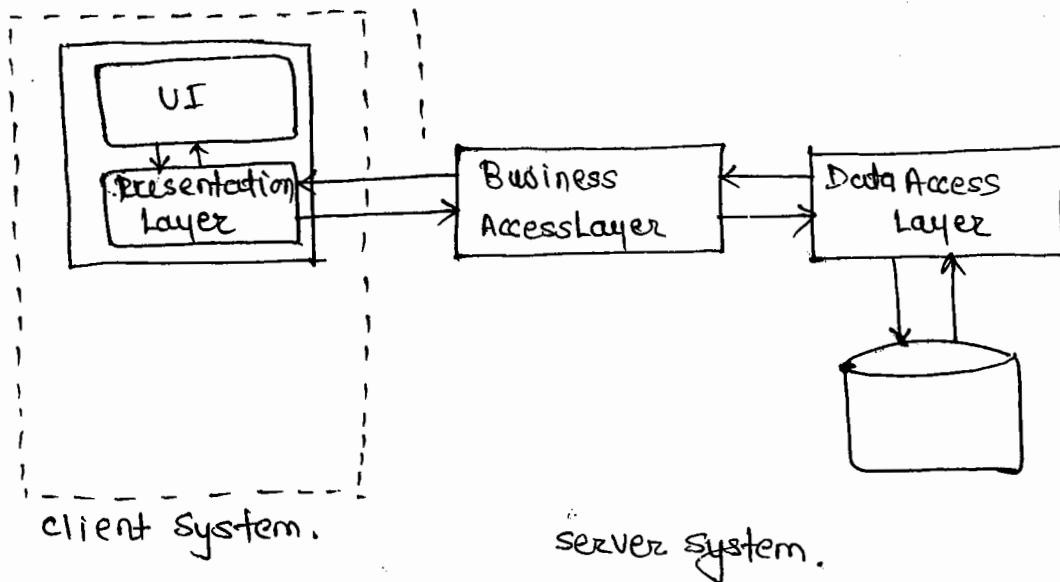
go to tool box go to reporting tag - select crystal viewer  
control drag & drop in form.

go to crystal report viewer control prop → select prop.  
report source → in dropdown list → select crystal

click export report button select save as type specify  
the file name click save button.

15/11/13  
Friday

## N-Tier Architecture



→ In the N tier architecture the UI & presentation layer will be located in the client system & business access layer and Data access Layer will be maintain in server system. Here there is the requirement of technology that allows us business access Layer with presentation layer. That technology is called as distributed technology.

→ The following are the well known & important distributed technology.

- 1) DCOM - Distributed component object model.
- 2) .NET Remoting.
- 3) Web Services
- 4) WCF Services (windows comm' foundation).

→ DCOM (Distributed Component Object Model)

→ It is in usage before .NET

→ It is platform dependent.

## 2) .NET Remoting :-

- It is introduced in .net framework
- It is suitable for the windows app's that run on LAN or Internet.
- It support TCP & HTTP protocols.

## 3) Web Services

- It is available in asp.net.
- It is a platform independent
- It is a language independent.
- It is supported for Asp.net websites only.
- It support SOAP (Simple object access protocol) & HTTP protocol.

## 4) WCF :-

- It is introduce in 3.0
- It is platform independent
- It is Language independent
- It is supported any type of Network (LAN, intranet & Internet etc)
- It is supported in any type of appli (windows app', web app', WPF app' etc)

### Example

```
Create procedure spinsert1 (@eno int, @name varchar(80),  
                           @sal int)
```

as

begin

```
    insert into employee values (@eno, @name, @sal)
```

end

Open Visual studio → select template → windows forms app' →

go to sol' exp → right click on app → add new item →  
select template app configuration file click add button,

```
<?xml version = "1.0" encoding = "utf-8" ?>
<configuration>
<appSettings>
<add key = "conn">
    value = "uid = sa ; database = mydb ; password = 123 ,
    server = nit-pc" />
</appSettings>
<iconfig>
```

## Data Access Layer.

goto sql explorer → right click on application → add new item select template class file → name it as Datusers.cs click add button.

```
using System.Data;
using System.Data.SqlClient;
class Datusers =
{
    public static void getmanipulations (string con, string pro,
                                         SqlParameter[] p)
    {
        try
        {
            SqlConnection cn = new SqlConnection (con);
            cn.open();
            SqlCommand cmd = new SqlCommand (pro, cn);
            cmd.CommandType = CommandType.StoredProcedure;
            foreach (SqlParameter param in p)
            {
                cmd.Parameters.Add (param);
            }
            cmd.ExecuteNonQuery();
        }
    }
}
```

```

        catch (Exception ex)
    {
        throw new ArgumentException(ex.Message);
    }
}

public static DataSet getData(string connec, string proce)
{
    try
    {
        SqlConnection cn = new SqlConnection(connec);
        SqlDataAdapter da = new SqlDataAdapter(proce, cn);
        DataSet ds = new DataSet();
        da.Fill(ds);
        return ds;
    }
    catch (Exception ex)
    {
        throw new ArgumentException(ex.Message);
    }
}

```

### Business Access Layer.

goto SQL explorer → right click on app → add new item →  
 select template class file → Business.cs → click add button

```

using System.Data;
using System.Data.SqlClient;
using System.Configuration;

```

```

{
    class Business
    {
        int empno, sal;   // right click on sal - refator
        string ename
        public int empno
        {
            get { return empno; }
            set { empno = value; }
        }
    }
}

```

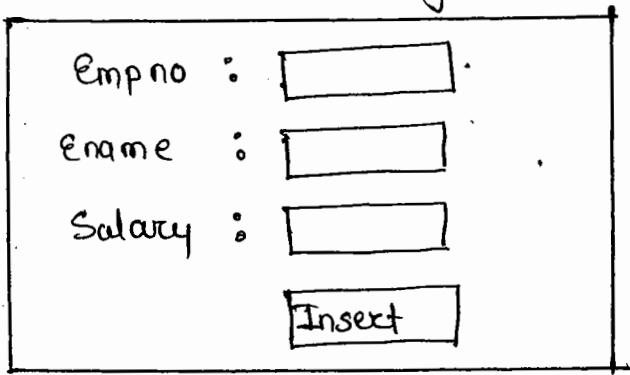
```
public int sal
{
    get { return sal; }
    { sal = value; }
}

public String ename
{
    get { return ename; }
    { ename = value; }
}

public void insertdata()
{
    try
    {
        SqLParameter
        SqLParameter[] P = new SqLParameter[3]
        P[0] = new SqLParameter("@eno", empno);
        P[0].DbType = DbType.Int16;
        P[1] = new SqLParameter("@name", ename);
        P[1].DbType = DbType.String;
        P[2] = new SqLParameter("@sal", sal);
        P[2].DbType = DbType.Int16;

        DataSets.getmanipulations.ConfigurationSettings.
            APPSetting["conn"], "spinser1", P);
    }
    catch (Exception ex)
    {
        throw new ArgumentException(ex.Message);
    }
}
```

## 6 Presentation Layer | UI



Root private void button1.click

{

try

{

Balusers obj = new Balusers();

obj.Empno = int.Parse(textBox1.Text);

obj.Ename = textBox2.Text;

obj.Sal = int.Parse(textBox3.Text);

obj.insertdata();

messageBox.Show("record added");

}

catch (exception ex)

{ messageBox.Show(ex.message);

}

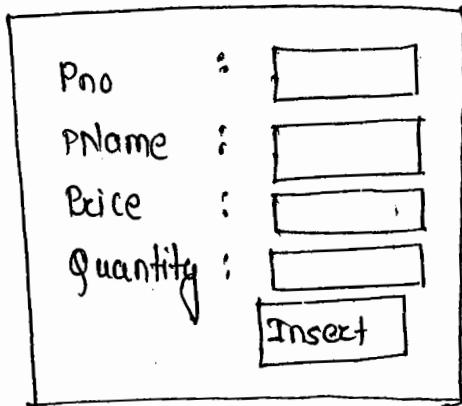
}

16/11/13

## StreamWriter

StreamWriter is a class available System.IO namespace.

It is used to write a files like word, excel, pdf, etc.



using System.IO;

private void button1\_Click(object

```
{  
    StreamWriter sw = new StreamWriter("c://Products.xml");  
    sw.WriteLine("<rootdata>");  
    sw.WriteLine("<products>");  
    sw.WriteLine("<Pno>" + textBox1.Text + "</Pno>");  
    sw.WriteLine("<Pname>" + textBox2.Text + "</Pname>");  
    sw.WriteLine("<Quantity>" + textBox3.Text + "</Quantity>");  
    sw.WriteLine("Price  
    sw.WriteLine("<Quantity>" + textBox4.Text + "</Quantity>");  
    sw.WriteLine("</products>");  
    sw.WriteLine("</rootdata>");  
    sw.Close();  
    MessageBox.Show("File added successfully");  
}
```

## XML TextWriter

XML TextWriter is a class available in System.Xml namespace. It is used to write a file like any XML file. It's like a Stream writer, but the Stream writer can not understand XML hierarchy.

Using XMLTextWriter predefined methods we have to write XML file

1) WriteStartDocument()

It generate like ---

<? XML Version = 1.0 ----->

2) WriteStartElement(Element)

ex WriteStartElement("Emp")

<emp>

3) WriteElementString(Element, value)

ex WriteElementString("sal", 3000)

<sal>3000</sal>

4) WriteEndElement()

5) WriteEndDocument()

### Example L

|   |                      |
|---|----------------------|
| Pno :                                   | <input type="text"/> |
| Pname :                                 | <input type="text"/> |
| price :                                 | <input type="text"/> |
| Quantity :                              | <input type="text"/> |
| <input type="button" value="Insert ."/> |                      |

```
using System.Xml;  
private void button1_Click  
{
```

```

XmlTextWriter xmlwo = new XmlTextWriter("c:\\Sample.xml",
    System.Text.Encoding.UTF8);
xmlwo.Formatting = Formatting.Indented;
xmlwo.Indentation = 2;
xmlwo.IndentChar = ' ';
xmlwo.CloseOutput = true;
xmlwo.WriteStartDocument();
xmlwo.WriteStartElement("products");
xmlwo.WriteElementString("pno", textBox1.Text);
xmlwo.WriteElementString("pname", textBox2.Text);
xmlwo.WriteElementString("peice", textBox3.Text);
xmlwo.WriteElementString("quantity", textBox4.Text);
xmlwo.WriteEndElement();
xmlwo.WriteEndElement();
xmlwo.WriteEndDocument();
xmlwo.Close();
MessageBox.Show("file saved successfully");
}

```

### Example 2

Diagram of a Windows dialog box:

- Filename :
- Empno :
- Ename :
- Salary :
- Address :

Buttons:

- Load → Button
- Insert
- Exist

go to toolbox → select goto dialog stub → select open file dialog control → drag & drop in the form.

```

using System.Xml;
// Load button code
XmlTextWriter xmlwo;
private void button1_Click(object sender
{
    OpenFileDialog1.ShowDialog();
}

```

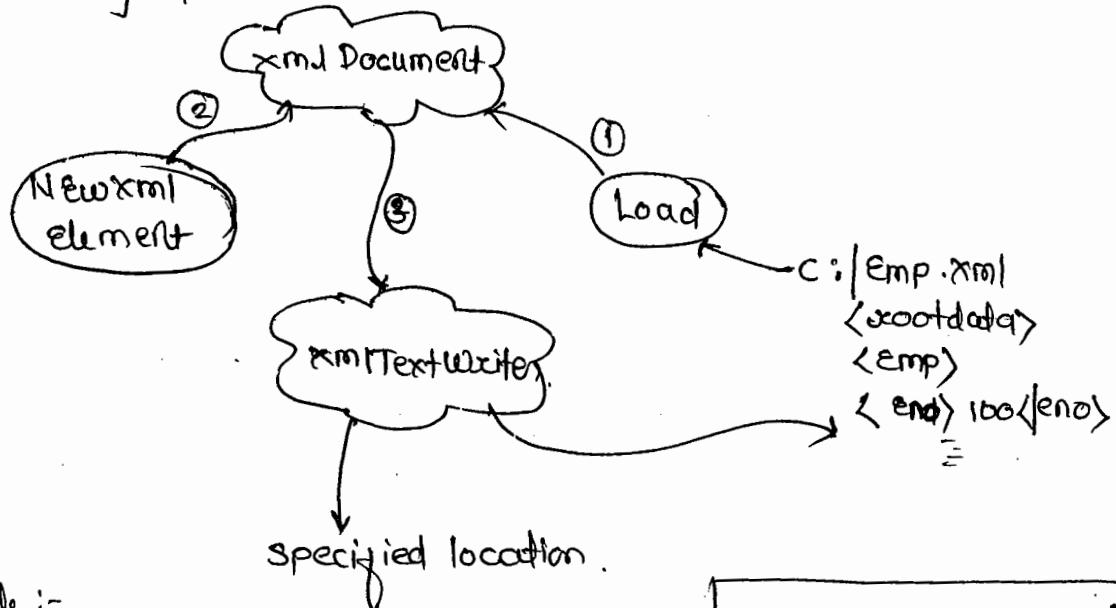
```
textBox1.Text = openFileDialog1.FileName;
xmldw = new XmlTextWriter(textBox1.Text, System.Text.Encoding.UTF8);
xmldw.WriteStartDocument();
xmldw.WriteLineStartElement("rootdata");
}

// insert button code
private void button2_Click -- 
{
    xmldw.WriteLineStartElement("empdata");
    xmldw.WriteLineElementString("empno", textBox2.Text);
    xmldw.WriteLineElementString("ename", textBox3.Text);
    xmldw.WriteLineElementString("sal", textBox3.Text);
    xmldw.WriteLineElementString("Address", textBox4.Text);
    xmldw.WriteLineEndElement();
    MessageBox.Show("file saved successfully");
}

// Exist button code
private void button3_Click -- 
{
    xmldw.WriteLineEndElement();
    xmldw.WriteLineEndDocument();
    xmldw.Close();
    MessageBox.Show("file closed successfully");
}
```

## XML Document

- XML document is a class available in System.Xml namespace. It is used to stored XML data for a temporary purpose.



18/11/13  
Monday

Example :-

```
using System.Xml;
```

```
 XmlDocument xdoc = new XmlDocument();
```

Load button code .

```
private void button1_Click(
```

```
{
```

```
    xdoc.Load(textBox1.Text);
```

```
    MessageBox.Show("file loaded successfully");
```

```
}
```

Insert button code .

```
private void button2_Click(
```

```
{
```

```
    XmlElement product = xdoc.CreateElement("products");
```

```
   XmlElement pno = xdoc.CreateElement("pno");
```

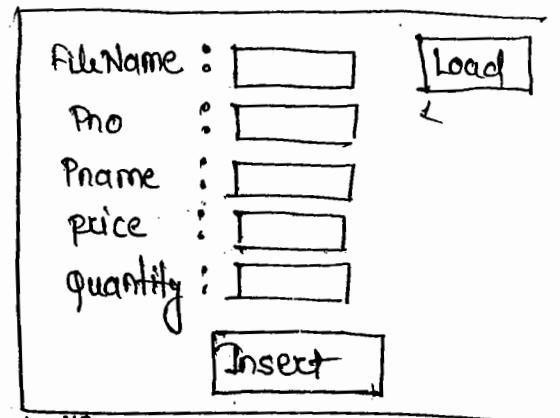
```
    pno.InnerText = textBox2.Text;
```

```
   XmlElement name = xdoc.CreateElement("pname");
```

```
    name.InnerText = textBox3.Text;
```

```
   XmlElement price = xdoc.CreateElement("price");
```

```
    price.InnerText = textBox4.Text;
```



```

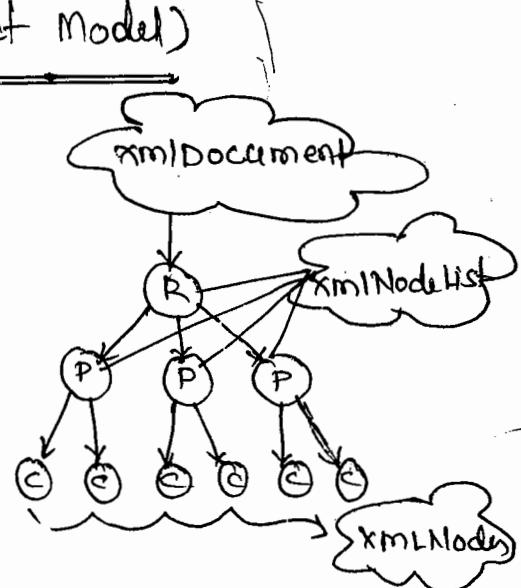
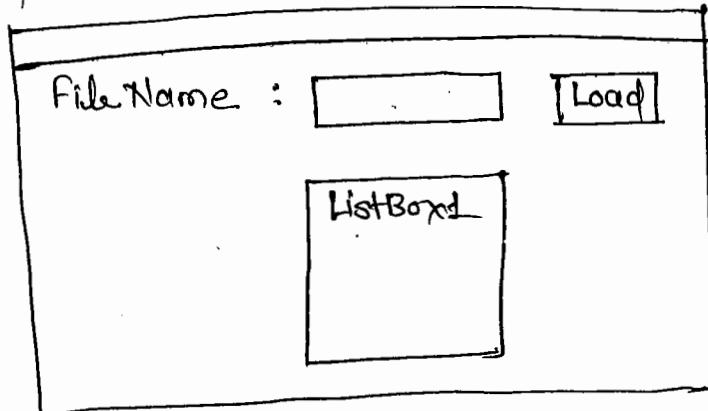
XmlElement qty = xmlDoc.CreateElement ("quantity");
qty.InnerText = textBox1.Text; // -
product.AppendChild (pno);
    (name);
    (price);
    (qty);

xmlElement.AppendChild (product);
XmlTextWriter xmlw = new XmlTextWriter (textBox1.Text, System.
    Text.Encoding.UTF8);
xmlw.WriteContentTo (xmlw);
xmlw.Close ();
MessageBox.Show ("file saved to :" + textBox1.Text);
}

```

## DOM Object ( Document object Model )

- ↳ XML NodeList
- ↳ XML Node



```

using System.Xml;
// Load button code
private void button1_Click (object
{
    XmlDocument xmlDoc = new XmlDocument ();
    xmlDoc.Load (textBox1.Text);
    MessageBox.Show ("file loaded successfully");
}

```

```

XmlNodeList xnl = xmlDoc.GetElementsByTagName("PName");
foreach(XmlNode xn in xnl)
{
    listBox1.Items.Add(xn.InnerText);
}

```

### \* FileInfo Class \*

This class represent a file on the file system. This able to get the information of the file & also To perform sudden operation on that folder.

#### Object Construction :-

Syntax :- FileInfo obj = new FileInfo ("path of the file");  
Ex. FileInfo obj = new FileInfo ("c:\sample.doc");

#### Properties of FileInfo class

##### Exists :-

check the existence of directory if it exist it indicate true otherwise it indicate false.

##### Name :-

get only the name of the file (without path)

##### FullName :-

gets the name along with full path.

##### Extension :-

gets the extension of the file.

##### DirectoryName :-

gets the name of the file in which the file exist.

##### Length :-

gets the size of file (in byte)

##### CreationTime :-

Gets the date and time when the file is created.

##### LastAccessTime :-

Gets the date & time when the file is access last time

##### LastWriteTime :-

Gets the date and time when the file is modified

##### LastTime :-

## ✓ Methods of FileInfo class

Description,



Method

Delete : Deletes the file permanently

CopyTo(Destination fileName) : Copies the file into the destination location with the specified file name.

MoveTo(Destination fileName) : Moves the file into the destination location with the specified file name.

Example:-

Open Visual studio → select lang → vc++ select template  
console application → specify the name & location.

```
using System.IO;
class Program
{
    static void main (string [] args)
    {
        string filepath;
        Console.WriteLine ("Enter file path");
        filepath = Console.ReadLine();
        FileInfo obj = new FileInfo (filepath);
        if (obj.Exists)
        {
            Console.WriteLine ("Name :" + obj.Name);
            Console.WriteLine ("fullName :" + obj.FullName);
            Console.WriteLine ("Extension :" + obj.Extension);
            Console.WriteLine ("Directory :" + obj.Directory);
            Console.WriteLine ("file size :" + obj.Length + " bytes");
            Console.WriteLine ("Created on :" + obj.CreationTime);
            Console.WriteLine ("Last accessed on :" + obj.LastAccessedTime);
            Console.WriteLine ("Last modified on :" + obj.LastWriteTime);
        }
        else
        {
            Console.WriteLine ("file is not found");
        }
    }
}
```

~~10/11/13~~ Tuesday StreamReader :-

Stream reader is available in System.IO namespace. Using Stream Reader we can read any type of files.

1) Import the API

↳ using System.IO

2) Create the stream reader object.

StreamReader obj = StreamReader (filename);

3) Read the Content.

obj. ReadToEnd();

4) Close the Reader.

obj. Close();

✓ Ex - Open vs → select template console application → click ok button

using System.IO

class Program

{

    static void main (string [] args)

{

        string filepath;

        Console.WriteLine ("Enter file path");

        filepath = Console.ReadLine();

        FileInfo obj = new FileInfo (filepath);

        if (obj. Exists)

{

            StreamReader sr = new StreamReader (filepath);

            String content = sr.ReadToEnd();

            Console.WriteLine (content);

            sr.Close();

}

        else

            {Console.WriteLine ("file not found");

            Console.Read();

}

## \* Deployment (Clicks ones deployment)

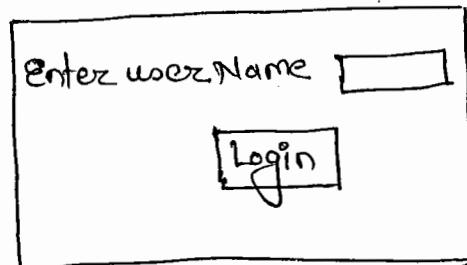
- After development of your application finally you have to generate the setup program & it can be given to the client.
- The client has to install the project simply by double clicking on the "Setup.exe" file.
- On installing the project in the client system, the following changes will be made in that system.
  - i) Then the necessary files such as exe files, dll files, Images, configuration file will be copied in "C:\program file folder" in the client system.
  - ii) The necessary shortcuts to run the app will be created at the user desktop and programs menu

Example :-

Open VS → select template windows forms app → click add button  
goto form1 property - change name Login & change text Login  
goto solution explorer → right click on application → add new item →  
select template - windows form name it as Logout.cs click  
add button.

go to login page.

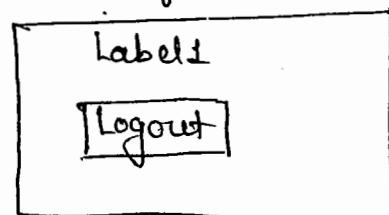
```
private void button1_Click(object
{
    logout obj = new logout();
    obj.Label1.Text = "Welcome to :" + textBox1.Text
    obj.Show();
    this.Hide();
}
```



go to logout form.

goto Label control properties changes the modifier public

```
private void button1_Click(object
{
    login obj = new login();
    obj.Show();
    this.Hide();
}
```



go to menu bar → go to field publish application →  
specify the location → click next → next → next → finish

## Backup & Restore Database

go to SQL server → go to databases → select required database  
right click → go to tasks → go to backup → click ok button.

go to databases → Right click on databases → go to Restore  
database → activate from device radio button click browse  
button → click add button.

go to Drive → select backup file → click on button → select  
Restore  → click ok button.

## Login Example using 3-Tier Architecture.

Create procedure spgetusers

(@name varchar (20),  
@pwd varchar (80))

as

begin

Select \* from users where uname = @name and  
password = @pwd

end .

Execution :-

exec spgetusers 'Ramesh', 'Ramesh'

⇒ Open vs → select template windows forms app → go to  
solution explorer → right click on application → add new item  
→ select template application configuration file → click add button

<?xml version = "1.0" encoding = "utf-8"?>

<configuration>

<appSettings>

<add key = "conn" value = "uid = sa; database = deatdb;  
password = 123"

</add>

<appSettings>

<configuration>

Data access Layer

goto SQL exp → right click on app → add new item → select template class file Dalusers.cs click add button.

```

using System.Data;
using System.SQLClient;
using System.Configuration;
class Dalusers
{
    public static Dataset checkusers(string name, string password)
    {
        SqlConnection cn = new SqlConnection(ConfigurationSettings.AppSettings["conn"]);
        cn.Open();
        SqlCommand cmd = new SqlCommand("spgetusers", cn);
        cmd.CommandType = CommandType.StoredProcedure;
        cmd.Parameters.AddWithValue("@name", name);
        cmd.Parameters.AddWithValue("@pwd", password);
        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataSet ds = new DataSet();
        da.Fill(ds);
        return ds;
    }
}

```

20/11/13  
Wednesday

Business Access Layer

goto SQL exp → right click on app → add new Item → select template class file → Balusers.cs → click add button

```

using System.Data;
class Balusers
{
    string name, password;
    public string password
    {
        get { return password; }
        set { password = value; }
    }
}

```

```

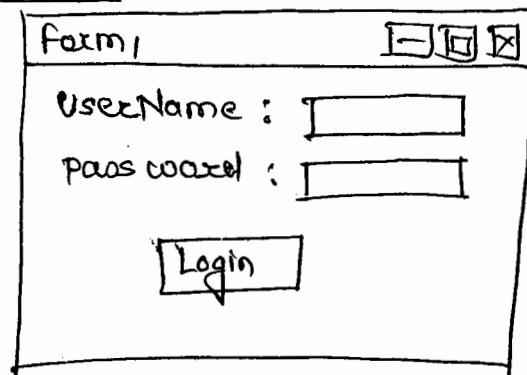
    public string Name
    {
        get { return name; }
        set { name = value; }
    }

    public DataSet loginusers()
    {
        DataSet myds = DatUsers.checkusers(name, password);
        return myds;
    }

```

### User Interface / Presentation Layer

go to sdn exp → right click on application → add new item → select template windows form → name it as Home.cs click → add button.



```

// Login button code
using System.Data;
private void button1_Click (object---)
{
    BalUsers obj = new BalUsers();
    obj.Name = textBox1.Text;
    obj.Password = textBox2.Text;
    DataSet ds = obj.loginusers();
    if (ds.Tables[0].Rows.Count > 0)
    {
        Home obj1 = new Home();
        obj1.show();
        this.Hide();
    }
    else
        MessageBox.Show ("Invalid uname | password");
}

```

## Windows Control

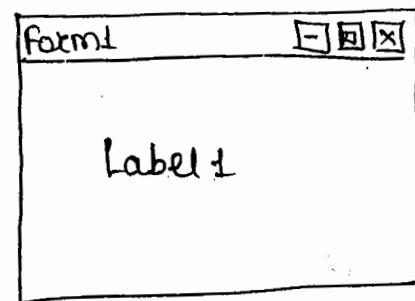
Timer :- Timer is a control which it execute the code at regular interval basis.

→ The main advantage of timer is it runs a separate process.

goto toolbox → select label control drag & drop in the form.

// Form Local code .

```
{
    class      prop      method
    label1.Text = DateTime.Now.ToString()
}
```



In toolbox some controls are displayed in separate tray is called as "component tray". These controls are not visible at runtime.

goto toolbox select timer control drag & drop in the form.

goto timer control properties

- 1) Enabled = True
- 2) Interval = 1000

goto timer events → double click on Tick event

```
private void timer1_Tick (Object sender, EventArgs e)
{
    label1.Text = DateTime.Now.ToString();
}
```

## Extender Controls

- 1) Tool Tip
- 2) Error provider
- 3) Help provider

## 1) ToolTip :-

If is used to display the tool tip as a part of our application.

Ex - goto toolbox → select tooltip control drag & drop in the form.

goto tooltip control properties

1) IsBalloon = True

2) ToolTipIcon = Info or warning or error.

3) Tooltip title = Alert (any text)

goto toolbox → select button control drag & drop in the form.

→ goto button control properties → select property

ToolTip.SetToolTip = enter any text

## 2) Error Provider :-

A better and new method for showing error icons in the form using error provider control.

goto toolbox select error provider control  
drag & drop in the form

private void button1\_Click ( -- )

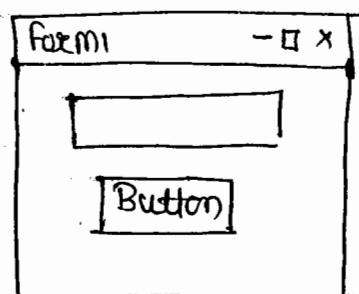
{ if (int.Parse (textBox1.Text) > 10)

{ errorProvider1.SetError (textBox1, "");

}

else

errorProvider1.SetError (textBox1, "Value is must above 10");



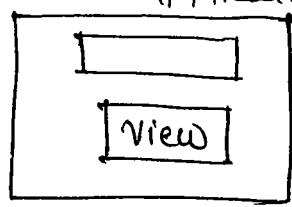
## 3) Help Provider :-

using Help provider control display the help file ~~at runtime~~ at runtime.

goto toolbox → select help provider control → drag & drop in the form.

goto Help provider control properties → select property  
→ Help namespace → click browse option attach  
any file copy the file path

go to form properties → select property HelpString or HelpProvider in that paste the file path → run the application  
Press F1 button. (key)

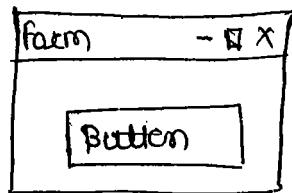


### 3) Help Providers:- NotifyIcon :-

It is used to display an icon in the system tray of taskbar.

go to toolbox → select NotifyIcon control drag & drop in the form.

go to NotifyIcon icon properties.



1) BalloonTip Icon = info / warning / error.

2) BalloonTip Text = any string

3) BalloonTip Title = enter any string

4) Icon — click browse button, select any icon

private void form3\_Load(object sender ...)

{  
    notifyIcon1.ShowBalloonTip(1000);  
}

21/11/13

### Creating a Runtime Control

usually the controls are designed in the form at design time, some time you may need to add the control programmatically at run time.

To generate the Control at Runtime follow the below step

#### 1) Create the Control Object :-

Control Classname obj = new Controlclassname();

#### 2) Assign the Required property (Name, Text etc) :-

obj.property = Value;

#### 3) Add the Control to the Container :-

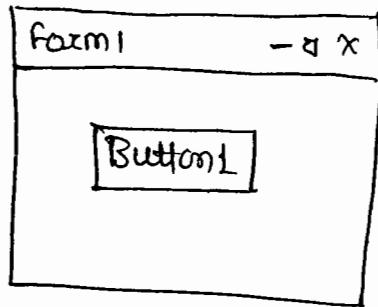
Containername.Controls.Add(obj);

Ex 1

```

private void button1_Click(object sender,
{
    TextBox t1 = new TextBox();
    t1.Location = new Point(230, 120);
    t1.BackColor = Color.Red;
    t1.ForeColor = Color.Yellow;
    this.Controls.Add(t1);
}

```



### Ex 2 - Create event handler

```

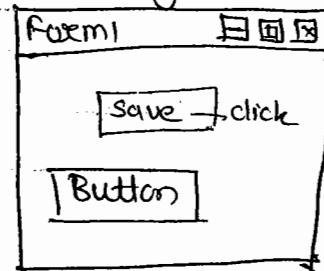
private void b1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Hai");
}

```

```

private void button1_Click(object sender, EventArgs e)
{
    buttonb1 = new Button();
    b1.Location = new Point(220, 120);
    b1.Text = "Save";
    b1.BackColor = Color.Green;
    b1.Click += new EventHandler(b1_Click);
    this.Controls.Add(b1);
}

```



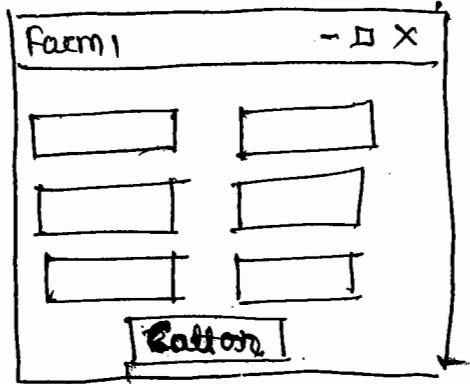
foreach syntax  
foreach (datatype variable in collection)

Ex:

```

private void button1_Click(object sender,
                           EventArgs e)
{
    foreach (Control c in this.Controls)
    {
        if (c is TextBox)
        {
}

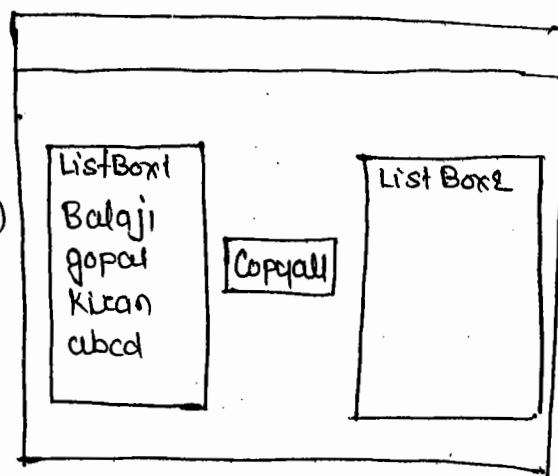
```



```
d. BackColor = Color.Blue;
```

Example 2 :-

```
private void button1_Click(object sender, EventArgs e)
{
    foreach (string s in listBox1.Items)
    {
        listBox2.Items.Add(s);
    }
}
```



## Dialog Boxes

- 1) Open file dialog
  - 2) Font dialog
  - 3) Color dialog
  - 4) Save File Dialog

This Controls are used to perform some tasks like opening a file, saving a file, applying font attributes, color attributes, performing page setting, printsetting etc.

## → OpenFileDialog :-

This control is used to display open file dialog box to the user so that user can select any file to open.

Ex - goto toolbar → goto dialog tab → select Open file Dialog  
control drag & drop in the form

goto toolbar → select picture box control drag & drop in form

go to picture for contract property → select property size more

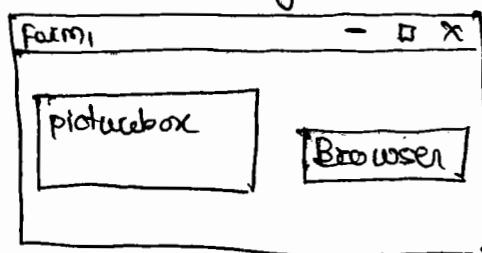
In that drop down list → select stretched image

```
private void button1_Click(object sender
```

```
    openfileDialog1.ShowDialog();
```

Picture Box 1. Image location = openFileDialog1.  
FileName;

File Name:



## Color Dialog :-

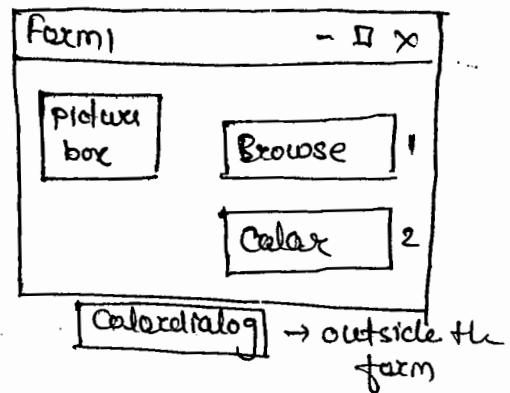
Using this controls back color at runtime

Ex - goto toolbox select colour dialog control drag & drop in the form.

```

private void button2_Click (---)
{
    ColorDialog1.ShowDialog();
    foreach (Control i in this.Controls)
    {
        if (i is Button)
        {
            i.BackColor = ColorDialog1.Color;
        }
    }
}

```



## Font Dialog :-

private void button3\_Click (---)

Using this control applying the font at runtime

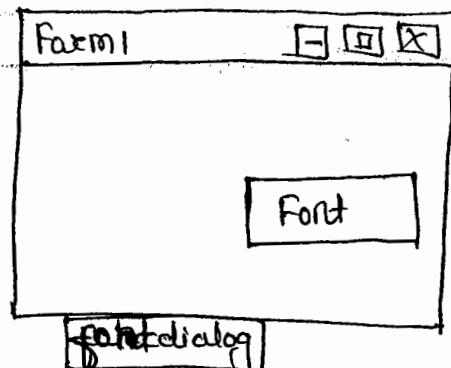
goto toolbox → goto font dialog tab → select font dialog control select drag & drop in form.

private void button3\_Click (---)

```

{
    fontDialog1.ShowDialog();
    foreach (Control i in this.Controls)
    {
        i.Font = fontDialog1.Font;
    }
}

```



## Rich TextBox | SaveFileDialog

### RichTextBox :-

- Using this control we can create our own text editor.
- It offers better feature when compared with the standard Textbox.
- It supports building file interaction with .rtf file (rich text format).

### Runtime Properties of RichTextBox.

- 1) Text :- Gets or sets the text of the entire RichTextBox.
- 2) SelectedText :- Gets or sets the currently selected background color.
- 3) SelectionBackColor :- Represents the background color for the selected text.
- 4) SelectionForeColor :- Represents the foreground color for the selected text.
- 5) SelectionAlignment :- Left or Right or Center or Justify.
- 6) Selection Font :- Represents the font setting for the selected text.

### Methods of RichTextBox

- 1) Clear() :- Clears entire text of control.
- 2) Cut() :- Cuts the selected text.
- 3) Copy() :- Copies the selected text.
- 4) Paste() :- Paste text from clipboard.
- 5) SelectAll() :- Selects the entire text of in the control.
- 6) Loadfile (.rtf file path) :- Loads the text from the specified ".rtf" file.
- 7) Savefile (.rtf file path) :- Saves the text of the control into ".rtf" file.
- 8) Undo :- Undo's the previous action.
- 9) Redo :- Redo's the previous action.

22/11/13

go to toolbox → select "RichTextBox" →  
openFileDialog → save file dialog control  
drag & drop in the form.

Openfile dialog control  
dialog control -

// Save button code

```
private void button2_Click(object sender, EventArgs e)
```

```
    saveFileDialog1.ShowDialog();
```

```
    richTextBox1.SaveFile(saveFileDialog1.FileName);
```

```
    richTextBox1.Clear();
```

```
    MessageBox.Show("file saved successfully");
```

}

// Open button code

```
{ private void button1_Click(object sender, EventArgs e)
```

```
    openFileDialog1.ShowDialog();
```

```
    richTextBox1.LoadFile(openFileDialog1.FileName);
```

}

## Tab Control

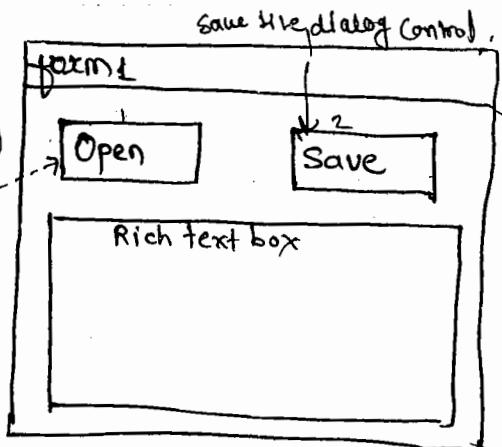
Tab control is a collection of tab pages. Every tab page is a container

Ex → go to toolbox → select tab control → drag & drop in the form.

go to tabcontrol properties select property tabpages click browse option add the tab pages.

go to toolbox → select linkable control drag & drop in the tabpage1.

go to linkable control property → change the start next >>



Move one tab page to another tab page

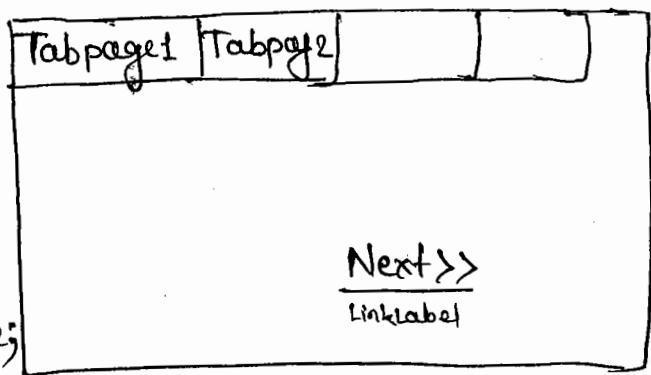
```
private void LinkLabel1_Click(---)
```

```
{
```

```
    TabControl1.SelectedTab = tabPage2;
```

```
}
```

~~Move on.~~



Creating a Runtime control in tab page :-

go to toolbox → select radiobutton control → drag & drop in the tabpage.

```
private void radioButton1_CheckedChanged
```

```
{
```

```
    TextBox dt1 = new TextBox();
    dt1.Location = new Point(125, 100);
    dt1.BackColor = Color.Red;
    dt1.Text = "Welcome";
    dt1.ForeColor = Color.Yellow;
    tabPage1.Controls.Add(dt1);
}
```

## Menus and ToolBox

1) menu strip

2) tool strip

3) status strip

4) context menustrip

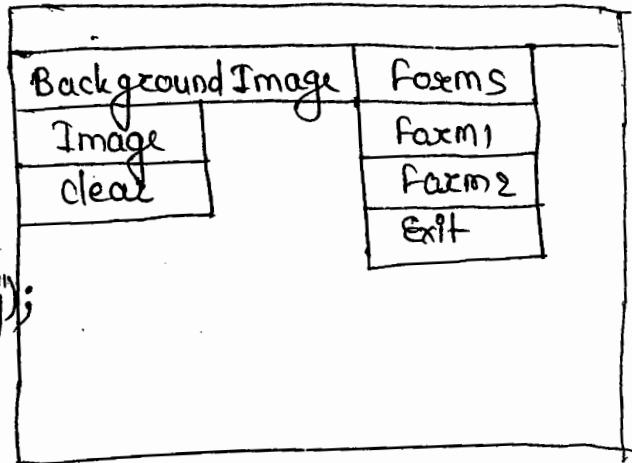
1) Menu Strip :-

go to toolbox → go to menus of toolbox tab → select menustrip control drag & drop in the form.

Right click on menu item → set image activate local resource  
radio button → click import button → select any icon.

↳ Open button.

```
private void image ToolStripMenuItem  
    click (---)  
{  
    this.BackgroundImage = image.  
    FromFile ("C:\\Chrysanthemum.jpg");  
}
```



// Clear menu code

```
private void clear ToolStripMenuItem_Click (---)  
{  
    this.BackgroundImage = null;  
}
```

// Form1 code

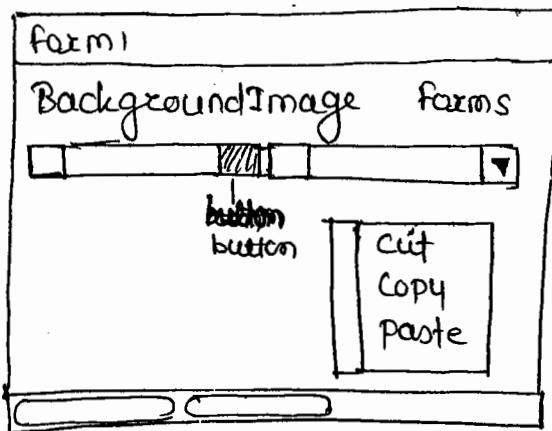
```
private void form1 ToolStripMenuItem_Click (---)  
{  
    Form1 obj = new Form1  
    obj.Show();  
}
```

// Form2 code

```
private void form2 ToolStripMenuItem_Click (---)  
{  
    Form2 obj = new Form2  
    obj.Show();  
}
```

// Exit code

```
private void exit ToolStrip --  
    this.Close();  
}
```



go to tool box → select tool strip  
startes → context menu strip control  
drag & drop in the form.

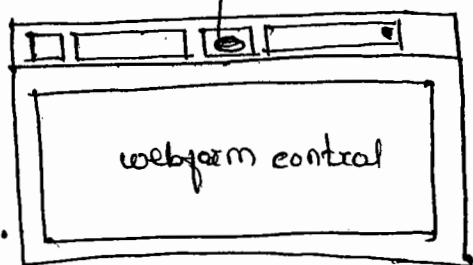
go to form properties → select  
property context menu strip in  
that drop down list apply  
context menu strip

o  
go to tool box → select web browser control tool strip control  
drag & drop in the form.

private void toolstripButton2\_Click(object

{  
}

WebBrowser1.Navigate(toolstripTextBox1.  
Text);  
}  
}



## TRACK BAR

Trackbar offers visualization for the value selection  
properties of track Bar.

- 1) Value :- Gets or sets a current value in the control.
- 2) Minimum :- specifies minimum value in the range.
- 3) Maximum :- specifies maximize value in the range.
- 4) TickFrequency :- specifies difference between each tick
- 5) Orientation :- Horizontal or vertical
- 6) ThickStyle :- None, stop left, bottom weight, both.

Ex - go to tool box → select label track bar control drag & drop

go to label1 properties → font = font

go to Label2 control properties → Text = "Net framework"

go to trackbar control properties minimum = 1 max = 200

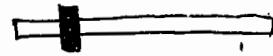
thick frequency = 5.

```

private void trackBar1_Scroll (Object--)
{
    int n = trackBar1.Value ;
    label2.Font = new Font ("Tahoma", n);
}

```

Font :



• .NET framework

## MDI (Multi Document Interface)

The windows app<sup>n</sup> are two type.

- 1) SDI Application (single Document interface)
- 2) MDI App<sup>n</sup> (Multi Document interface)

The SDI and MDI applications contain multiple form, but if SDI app<sup>n</sup> each form will be executed as individual form. whereas in MDI applications one form acts as parent form and the remaining forms acts as child form.

### Features of MDI applications :-

- 1) All the child forms are contained by the parent form so that the parent form is also called as "container form".
- 2) Among several child forms only one form acts as "active child form".
- 3) Generally the parent form contain no UI design it contains a menu.
- 4) Any child form can not be moved outside of its parent form.
- 5) In VB 6.0 only one form can be implemented as parent form in a project, but in C# .net and VB .net you can define multiple parent form within the same project.
- 6) In VB 6.0 the parent form can not contain any type of control but in C# .net and VB .net you can drag any controls.
- 7) The child form icon is not displayed in the windows taskbar.
- 8) If the parent form is moved all the child form will be moved.
- 9) Whenever the parent form is minimize all the child form minim

whenever the parent form is maximize all the child form will be created.

whenever the child form minimize icon will be created at the bottom area of parent form.

whenever the child form maximize the size of the parent form and child form will be concatenated.

whenever you close the parent form all the child form will be closed automatically.

The child form is able to access the reference of its parent form.

The parent form is able to access the references of its child form

### Implementation of MDI Appl' In C#

1) To convert the form as parent form.

To convert set that forms property. "ISMDIContainer = true"

2) Inwork the child form at runtime in the parent form.

```
ChildFormClassName obj = new ChildFormClassName();
```

```
obj.MdiParent = this;
```

```
obj.Show();
```

Farms	ChildForm BackColor
Form2	Blue
Form3	Green
close	
exist	Red

goto form1 properties  
select property ISMDIContainer =  
true.

goto toolbox → select menustrip  
control drag & drop in the  
parent form.

// Form2 code.

```
private void Form2ToolstripMenuItem_Click(---)
```

```
{
```

```
    Form2 obj = new Form2();
```

```
    obj.MdiParent = this;
```

```
    obj.Show();
```

```
}
```

```
private void Form3ToolstripMenuItem_Click(---
```

```
{  
    forms obj = new form3();  
    obj.MdiParent = this;  
    obj.show();  
}
```

### // Close button code

```
private void close ToolStripMenuItem_Click (---)  
{  
    if (this.ActiveMdiChild != null)  
    {  
        this.ActiveMdiChild.Close();  
    }  
}
```

### // Exit code

```
private void exitToolStripMenuItem_Click (object ---)  
{  
    this.Close();  
}
```

### // Blue button Code

```
private void blueToolStripMenuItem_Click (object ---)  
{  
    if (this.ActiveMdiChild != null)  
    {  
        this.ActiveMdiChild.BackColor = Color.Blue;  
    }  
}
```

### // Red button code

```
private void redToolStripMenuItem_Click (object ---)  
{  
    if (this.ActiveMdiChild != null)  
    {  
        this.ActiveMdiChild.BackColor = Color.Red;  
    }  
}
```

```

private void GreenToolStripMenuItem_Click(object ---)
{
    if(this.ActiveMdiChild != null)
    {
        this.ActiveMdiChild.BackColor = Color.Green;
    }
}

```

## Check Box

It is used to take the choice from the user. the checkbox can be checked or unchecked by the user.

### Properties of CheckBox

- 1) Name :- specifies the name of control.
- 2) Checked :- Represents the current status of the checkbox whether it is checked or unchecked.
- 3) Text :- specifies the displayable text of the control.

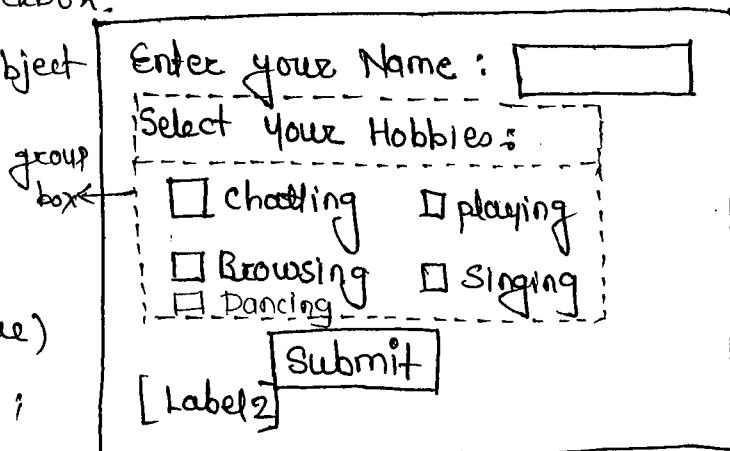
### Event of checkbox

- 1) CheckedChanged :- Executes when the user check or uncheck the checkbox.

```
private void button1_Click(object
```

```
{
    string s = textBox1.Text
    + "Hobbies are";
    if(chkChatting.Checked == true)
        s = s + chkChatting.Text + " ";
}
```

```
if(chkPlaying.Checked == true)
    s = s + chkPlaying.Text + " ";
if(chkBrowsing.Checked == true)
    s = s + chkBrowsing.Text + " ";
if(chkSinging.Checked == true)
    s = s + chkSinging.Text + " ";
```



30 Nov

```
Label2.Text = s;  
}  
} ⇒  
private void button1_Click(object sender  
{  
    string s = textBox1.Text + " Hobbies are";  
    foreach (Control x in this.Controls)  
    {  
        if (x is CheckBox)  
        {  
            CheckBox c = (CheckBox)x;  
            if (c.Checked == true)  
            {  
                s = s + c.Text + " ";  
            }  
        }  
    }  
}
```

Note :- In the above example Is not generic code if no of checkboxes increase or decrease then code should be modified which is not acceptable in real time programming so we write code

### ~~25/11/13 Monday~~ SQL Helper

- Sql helper has a predefined class . This class file contain a set of predefined method.
- using predefine method we can perform insert update delete & select option.

Create procedure spinsert (@eno int, @name varchar(30),  
@sal int)

```
as  
begin  
    insert into emp values (eno, @name, @sal)  
end.
```

Create procedure spgetdata

as  
begin

Select \* from emp

end

open vs → select template windows forms application  
goto solution explorer → right click on application add new  
item select template application configuration click add  
button.

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <appSettings>
    <add key="connec" value="uid=sa; database=
      sampledb8 ; password =123, server=nit-pc" />
  </appSettings>
</configuration>
```

Data access Layer.

goto sql exp → right click on application add existing  
item → add sql helper class file in our project.

Business access layer.

goto sql explorer → right click on application add new item  
select template class file name it as Balusers.cs click add  
button.

```
using System.Data;
using System.Data.SqlClient;
using System.Configuration;
using ComponentNetwork.DAL;

namespace WindowsFormsApplication1
{
  class Balusers
  {
    public int Empno
```

```
    get ;  
    set ;  
}  
public string Ename  
{  
    get ;  
    set ;  
}  
public int sal  
{  
    get ;  
    set ;  
}
```

```
public void insertdata()  
{
```

```
    SqlParameter []P = new SqlParameter [3];  
    P[0] = new SqlParameter ("@eno", Empno);  
    P[0].DbType = DbType.Int16;  
    P[1] = new SqlParameter ("@name", Ename);  
    P[1].DbType = DbType.String;  
    P[2] = new SqlParameter ("@sal", sal);  
    P[2].DbType = DbType.Int16;
```

```
    SqlHelper . ExecuteNonQuery (ConfigurationSettings.AppSettings
```

```
        ["connec"], CommandType.  
        StoredProcedure,"spinsert",P);
```

```
}
```

```
//Returing
```

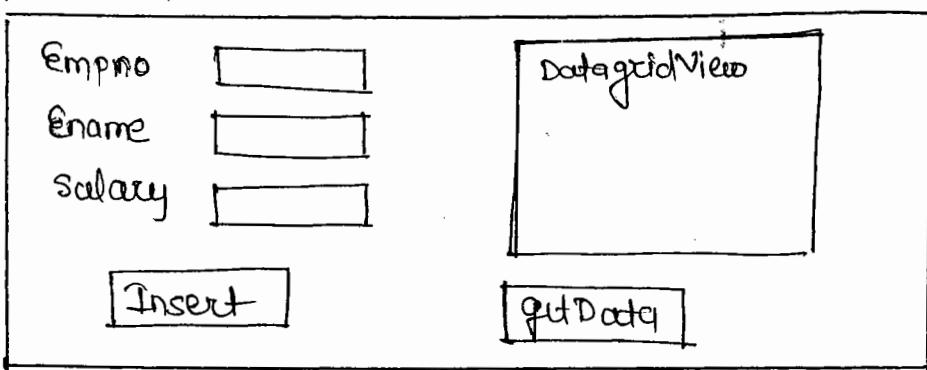
```
public Data Set GetTempdata()  
{
```

```
    DataSet myds = SqlHelper . ExecuteDataSet (ConfigurationSettings  
        . AppSettings ["connec"], CommandType.StoredProcedure,  
        "spgetdata");
```

```
    return myds ;
```

```
}
```

## Presentation Layer.



```
private void button1_Click (object ---)
```

```
{
```

```
    Balusers obj = new Balusers();  
    obj.Empno = int.Parse(textBox1.Text);  
    obj.Ename = textBox2.Text;  
    obj.Sal = int.Parse(textBox3.Text);  
    obj.insertdata();
```

```
    MessageBox.Show ("Record added");
```

```
}
```

```
public void button2_Click (Object ---)
```

```
{
```

```
    Balusers obj = new Balusers();  
    DataSet ds = obj.getempdata();  
    dataGridView1.DataSource = ds.Tables[0];
```

```
}
```

## WCF (Windows Communication foundation)

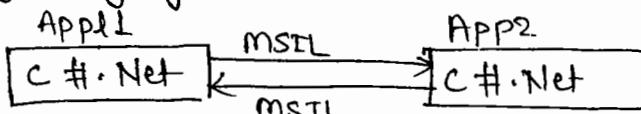
WCF is a unified programming model more secured, reliable service oriented application.

### Types of Application

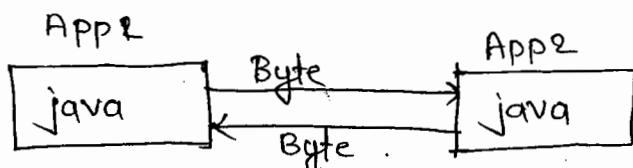
- 1) Homogeneous app
- 2) Heterogeneous app

#### 1) Homogeneous application :-

Applications are developed using same technology or same programming language are known as homogeneous application.



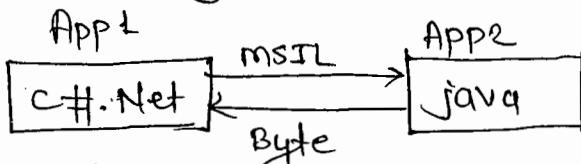
In this both application are developed using C# both applications can communicate by using msIL code because both app's can understand msIL code.



- In this both app's can communicate using byte code because both application can understand byte code.
- The communication in between homogeneous application will take place in the form of object data.

#### 2) Heterogeneous Application :-

Application that are developed using diff technology or different programming language are known as Heterogeneous app



- In the above diagram app1 can understand msIL code & it can also deliver msIL code.
- whereas app2 is developed using java it cannot understand msIL code as well as it can understand byte code as well as it can deliver Byte code. so it is not possible to

make communication in between these two app's because companies followed their own proprietary standard format while developing the technologies or programming language.

→ To overcome this drawback we use service oriented architecture SOA / programming.

### SOA Programming

- SOA programming specially speaks that communicating among heterogeneous app.
- SOA is a flexible setup design principle used during the system development and integration.
- A system based on SOA architecture will provide a loosely coupled & suit of services that can be used within the multiple separate system from different business domain.
- SOA also generally provides a way for consumer services such as webpaged or NonWebpaged app.
- If SOA programming is followed by all the companies then open standard message format can be used to communicate with application developed in any technology.

~~26/11/13  
Tuesday~~

### What is Message

A message is the data ie to be transferred between two heterogeneous application with an open standard format. A message is a set of contain unique of data that consist of several parts including a body, header etc.

### Services

Services are logical encapsulation of self contained business functionality. A service is not a class or not a component.

→ The diff. between a service and a component is

Component is used to communicate between homogeneous application, whereas a service is used to communicate bet<sup>n</sup> heterogeneous contract.

## Programming tool for SOA

- |                |               |  |
|----------------|---------------|--|
| 1) Remoting    | 4) TIBCO      | 6) RMI   |
| 2) webservices | 5) Web Method | 7) CORBA (common object Request Broker architecture) |
| 3) WCF         | 8) msmq       |  |

## SOAP (Simple object access protocol)

- Soap stands for simple object access protocol }
- Soap is completely open standard format in the form of XML
- Soap is a communication protocol and can be set as HTTP + XML
- Soap supports all other communications protocol
- Soap is language independent ie any programming language or technology can understand this soap format.
- Soap is platform independent.
- Soap will transfer the data in the form of envelope called "Soap envelope"
- Soap envelope contain 3 part.
  - i) Soap header      ii) Soap body.
  - iii) Soap Fault
  - iv) Soap Error .

### i) SOAP Header :-

SOAP header contain will contain , application specific information like the type of application , authentication .

### ii) SOAP BODY :-

The SOAP body will contain the original data it that is to be transferred from one application to another .

### iii) SOAP Fault :-

SOAP fault contain any kind of error that may occur while transferring data .

Writing the soap header within the soap envelope is optional if at all soap header is included within the envelope it should be 1st tag .

existing SOAP fault within the envelope also optional. If at all SOAP fault included it should be written as a subtag within the SOAP Body.

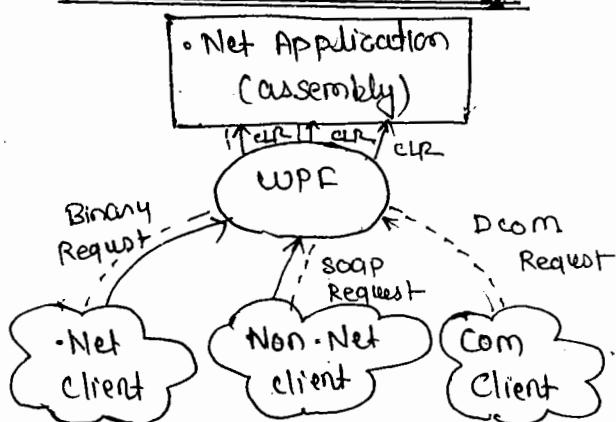
## Structure of SOAP Envelope

```
<SOAP Envelope>
  <SOAP Header>
    </SOAP Header>
  <SOAP Body>
    <emp>
      <empno> 1001 </empno>
      <ename> ABCD </ename>
    </emp>
    <SOAP Fault>
  </SOAP Body>
</SOAP Envelope>
```

## WCF

- WCF :-
- WCF is a unified programming model more secured, reliable services oriented applications.
  - WCF does not support GUI features.
  - WCF is a completely URI based programming.
  - WCF is a one Runtime added to CLR
  - WCF takes care overall infrastructure.

## WCF Architecture :-



Every WCF Service generates end point. The end point is available ABC format

A - Address

B - Binding

C - Contracts.

Address specifies where the service is located.

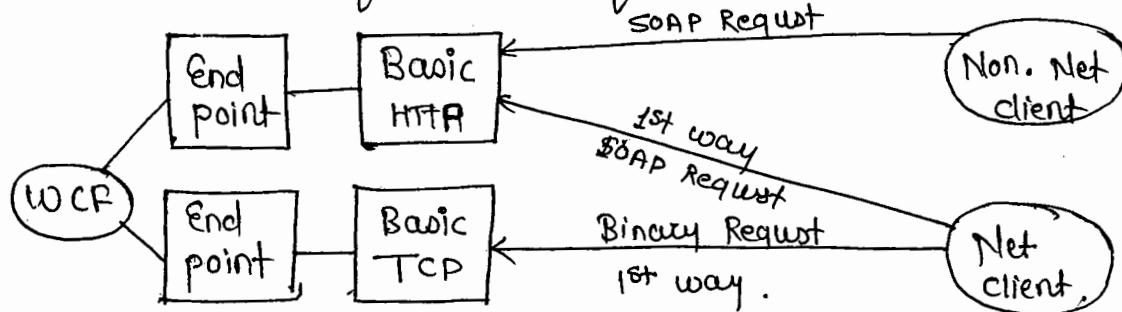
Binding specifies how to interact with WCF services.

Bindings are two type

1) TCP Binding

2) HTTP Binding.

Contracts specifies which feature are available in the service.



### WCF Contracts / Attributes :-

- 1) Service Contract
- 2) Operation Contract.
- 3) Data Contract
- 4) Message contract
- 5) Fault contract.

#### 1) Service Contract :-

Service contracts specifies how many classes are available in the service and which classes are expose to others.

#### 2) Operation Contract :-

Operation contract specifies which method are available in service, and which method expose to other.

Whenever we are open wcf service the automatically generate two files.

- 1) IService1.cs
- 2) Service.cs

Ex :-

Service.cs

```
class service : IService
{
    void m1()
    {
    }
    void m2()
    {
    }
}
```

IService.cs

[Service Contract]

Interface IService

[OperationContract]

void m1();

[OperationContract]

> void m2();

Example :- 27/11/13

Open visual studio → go to new project select language visual c# and select template wcf service application click ok button go to solution explorer Remove IService.cs , Service.cs file.

go to solution explorer → right click on application add new item select template wcf service name it as job.svc - click add button

goto IJobs.cs class file

[ServiceContract]

public interface IJobs

{

[OperationContract]

void DoWork();

[OperationContract]

string getTemp();

}

goto jobs.svc file

public class jobs : IJobs

{

Public void DoWork()

{

}

public string getTemp()

{

```

        return "welcome";
    }
}

```

Run the service → goto address bar → copy the URL

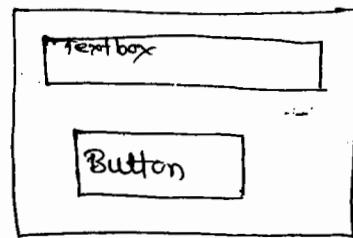
\* open Visual studio → goto file → new website → select template asp.net empty website click ok button.

\* goto sol' explorer → right click on the application go to add service reference → goto address bar paste the URL specify the name → click add reference button.

```
protected void Button1_Click(object
```

```
{
}
```

```
Sample.IjobClient obj = new Sample();
TextBox1.Text = obj.getser();
```



### Example :-

\* goto service → goto IJobs.cs file.  
[ServiceContract]

```
public interface IJobs
```

```
{
```

[OperationContract]

```
int add ( inta , intb );
```

[OperationContract]

```
int sub ( inta , intb );
```

[OperationContract]

```
int mul ( inta , intb );
```

```
}
```

go to class file

```
public class Jobs : IJobs
```

```
{
```

```
public int add( inta , int b )
```

```
{ return a+b ; }
```

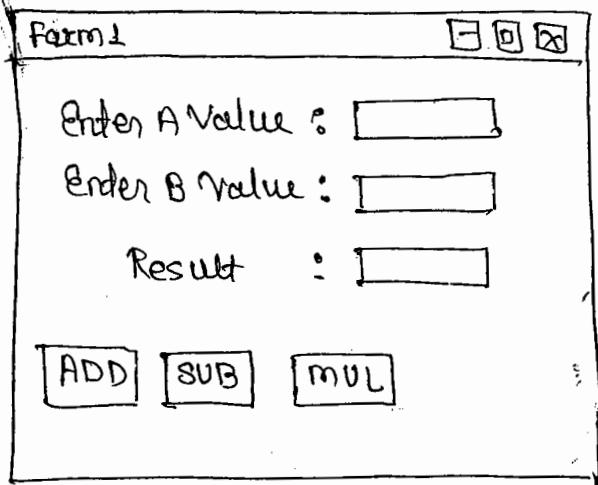
```
}
```

```

public int sub (int a, int b)
{
    return a - b;
}

public int mul (int a, int b)
{
    return a * b;
}

```



Run the service copy the URL

\* Open visual studio → goto windows forms app → go to solution explorer → right click on references → go to add reference service → go to address bar → paste the URL.  
specify the name space name click ok button,

private //Add button code

private void button1\_Click (object sender, EventArgs e)

{

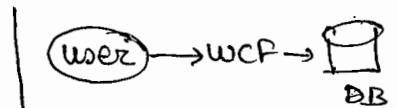
```

xyz.IJobsClient obj = new xyz.IJobsClient ();
int c = obj.add (int.Parse (textBox1.Text), int.Parse
                  (textBox2.Text));

```

textBox3.Text = c.ToString ();

}



\* Using WCF service retrieve the data Back end to front end.

\* goto service → goto IJobs.cs file.

[Service Contract]

using System.Data;  
sqlclient;

public interface IJobs

{

[OperationContract]

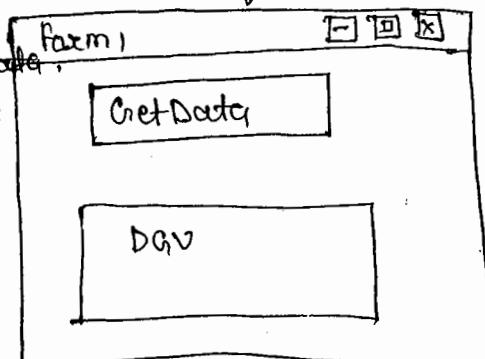
DataSet Getdata();

}

```

goto jobs.svc file System.Data; System.Data.SqlClient;
public class jobs : IJobs

```



```

public DataSet GetData()
{
    SqlConnection cn = new SqlConnection ("uid=sa ;
        password=123 ; Database=mydb ; server=nit-pc");
    SqlDataAdapter da=new SqlDataAdapter ("select * from emp",cn);
    DataSet ds = new DataSet();
    da.Fill(ds);
    return ds;
}

// GetData button code
private void button1_Click (Object ----)
{
    abc.IJobsClient obj = new abc.IJobsClient ();
    DataSet myds = obj.GetData();
    DataGridView1.DataSource = myds.Tables[0];
}

```

pass the empno and  
goto service → gets IJobs.cs file.

[Service Contract]

public interface IJobs  
{}

[Operation Contract]

~~DataSet GetData();~~ String Getname (int empno);

}

gets jobs.svc file

~~public class jobs : IJobs~~

goto service → goto IJobs.cs file

[Service Contract]

public interface IJobs

{

[OperationContract]

String Getname (int empno);

}

public class jobs : IJobs

{

public string Getname (int empno)

{

SqlConnection cn = new SqlConnection ("uid = sq; password

= 123; database = mydb ; server = NH - pc");

cn . Open ();

SqlCommand cmd = new SqlCommand ("Select \*  
from emp where empno = " + empno + " , cn);

SqlDataReader dr = cmd . ExecuteReader ();

if (dr . Read ())

{

string s = dr ["empname"] . ToString ();

return s;

}

else

{

return "no such records";

}

}

private void button1\_Click (object sender , EventArgs e)

{

abcd . IGetNameClient obj = new abcd . IGetNameClient ();

int n = int . Parse (textBox1 . Text);

textBox2 . Text = obj . getname (n);

}

28/11/13 Example :- Insert, update, delete front end to backend using stored procedure.

→ Create procedure spinsert (@eno int, @ename varchar(30),  
@Loc varchar(20))

as

begin

insert into dept values (@dno, @dname, @Loc)

end

→ Create procedure spdelete (@dno int)

as

begin

delete from dept where dno = @dno

end

go to interface Ijobs.cs class file :-

[ServiceContract]

public Interface Ijobs

{

[OperationContract]

void insertdata (int a, string b, string c);

[OperationContract]

void deletedata (int a);

}

go to class file :-

public class jobs : Ijobs

{

public void insertdata (int a, string b, string c)

{

SqlConnection cn = new SqlConnection ("uid = sa; database = mydb; password = 123; server = nit-pc");

cn. Open ();

SqlCommand cmd = new SqlCommand ("spinsert", cn);

cmd. CommandType = CommandType. StoredProcedure;

```

cmd.Parameters.AddWithValue("dno", a);
cmd.Parameters.AddWithValue("dname", b);
cmd.Parameters.AddWithValue("loc", c);
cmd.ExecuteNonQuery();
}
public void deletedata (int a)
{
    SqlConnection cn = new SqlConnection ("uid = sq ;
        database = mydb ; password = 123 ; server = nit-pc");
    cn.open();
    SqlCommand cmd = new SqlCommand ("spdeleted", cn);
    cmd.CommandType = CommandType.StoredProcedure;
    cmd.Parameters.AddWithValue("dno", a);
    cmd.ExecuteNonQuery();
}

```

Run the service → copy the URL → go to windows forms app → abc references.

```

public void button1_Click (object sender, EventArgs e)
{
    abc.Ijobsclient obj = new abc.Ijobsclient ();
    obj.insertdata (int.parse(textBox1.Text), textBox2.Text,
                    textBox3.Text);
    MessageBox.Show ("Record added");
}
public void button2_Click (object sender, EventArgs e)
{
    abc.Ijobsclient obj = new abc.Ijobsclient ();
    obj.deletedata (int.parse(textBox1.Text));
    MessageBox.Show ("Record deleted");
}

```

## Data Contract :-

go to WPF Service → go to solution explorer → right click on the application → add new item → select template class file → name it as employee.cs → click add button.

using System.ServiceModel;

using System.Runtime.Serialization;

{ [DataContract]

public class employee

{

[DataMember]

public int empno

{ get;  
set;

}

[DataMember]

public string ename

{ get;  
set;

}

[DataMember]

public int sal

{ get;  
set;

}

}

go to interface

[ServiceContract]

public interface IJobs

{

```
[OperationContract]
void DoWork();
[OperationContract]
Employee getdata(int empno);
}

goto class file Jobs :-
using System.Data.SqlClient;
{
public class Jobs : IJobs
{
    public void DoWork()
    {
        public Employee getdata(int empno)
        {
            SqlConnection cn = new SqlConnection("uid = sa ;
                database = mydb ; password = 123 ; server = nit-pc");
            cn.Open();
            SqlCommand cmd = new SqlCommand("select * from employee
                where eno = empno", cn);
            SqlDataReader dr = cmd.ExecuteReader();
            Employee obj = new Employee();
            if (dr.Read())
            {
                obj.Ename = dr["ename"].ToString();
                obj.Sal = Convert.ToInt16(dr["Sal"]);
                return obj;
            }
            else
                return obj;
        }
    }
}
```

29/11/13

Run the service copy the URL

\* Open Visual studio → select template → windows form  
appd → g

\* goto soln explorer → right click on references → add service reference → goto address bar → paste the url → specify the name → click add button.

private void button1\_Click (object sender..

{

abcd.Ijobsclient obj = new abcd.Ijobsclient();  
abcd.Employee obj1 = obj.GetData();

(int.Parse(textBox1.Text));

textBox2.Text = obj1.Ename;

textBox3.Text = obj1.sal.ToString();

}

Form1

Empno :	<input type="text"/>
Ename :	<input type="text"/>
Salary :	<input type="text"/>
<input type="button" value="GetData"/>	

\* Through OLEDB Connection with Oracle Client

using System.Data.OLEDB;

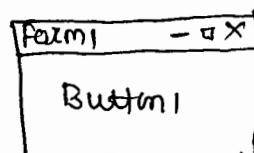
private void button1\_Click (object sender, EventArgs e)

{

OLEDBConnection cn = new OLEDBConnection ("uid=sa; password=Tiger; server=balaji; provider=msdaora");

cn.Open();

MessageBox.Show ("Connection established");



\* ORACLE Connection through ORACLE client  
System.Data.OracleClient;

private void button1

{ oracleConnection cn = new oracleConnection ("uid=sa; password=Tiger; server=balaji");

# Generic Delegate

Through

1) Func < T, T >

2) Action < >

3) Predicate < >

## Example 1

```
Public delegate double Calculatearea  
        (int x);
```

```
public double calculate (int x)
```

```
{
```

```
    return 3.14 * x * x;
```

```
}
```

```
private void button1_Click(object sender --)
```

```
{
```

```
    Calculatearea obj = new Calculatearea (calculate);
```

```
    MessageBox.Show (obj(12).ToString());
```

```
}
```

\* Using Anonymous Method Reduce the code.

Anonymous is a inline function.

```
public delegate double calculatearea (int x);
```

~~private void button1\_Click --~~

```
{
```

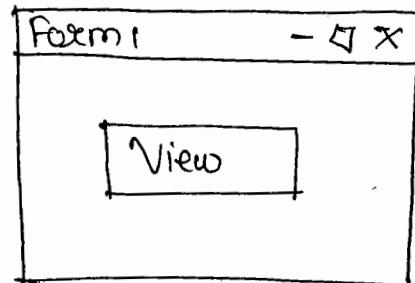
```
    Calculatearea obj = new calculatearea (delegate (int x)
```

```
{
```

```
    return 3.14 * x * x;
```

```
} );
```

```
    MessageBox.Show (obj(12).ToString());
```



## 1) Func

- It is a generic delegate
- It is a usermade delegate
- It takes input values and output values.

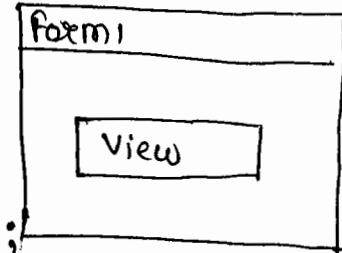
```
private void button1_Click()
```

```
{
```

```
    Func<double, double> obj = x => 3.14 * x * x;
```

```
    MessageBox.Show(obj(12).ToString());
```

```
}
```



## 2) Action :-

- Action is a generic delegate.
- It is a usermade delegate.
- It always takes only input values. return type void

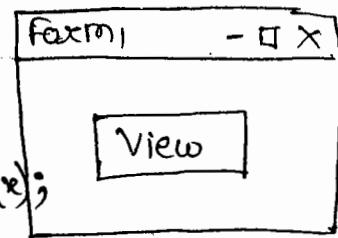
```
private void button1_Click()
```

```
{
```

```
    Action<string> obj = x => MessageBox.Show(x);
```

```
    obj("welcome");
```

```
}
```



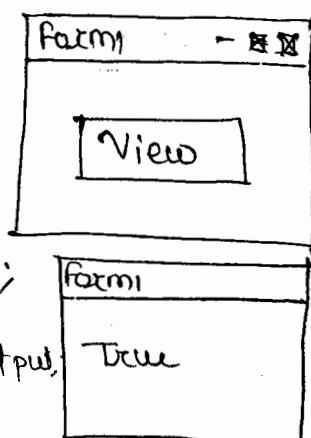
## 3) Predicate :-

- predicate is a predefine delegate.
- It is a generic delegate.
- It takes input values. It always returns boolean value, either true or false.
- It is used for only checking purpose.

```

private void button1_Click(object sender,
{
    predicate<string> obj = x => x.Length > 5;
    MessageBox.Show(obj("welcome").ToString());
}

```



## Generic Classes And Method

- These are known as general datatypes.
- Using generics we will write generic function & generic classes.
- Generics are introduced in .Net framework version 2.0
- Namespaces related to generics are
  - 1) System.Collections.Generic ,
  - 2) System.Collections.ObjectModel .

These namespaces contains generic collection classes that will allow working with generic types

- Generics can be implemented with classes structure interfaces, methods and events & delegates.
- Generics will maximize code reusability, Type safety & performance
- Generics are used to avoid function overloading when data types of arguments are changed.
- To work with generics we use two things.
  - 1) Place holder Represented By <>
  - 2) Type Parameter .

Always Type parameter should be enclose within the placeholder like <Typeparameter>

Eg:- goto Console Application.

```
namespace ConsoleApplication2
{
    public class Sample
    {
        public static void Display <T>(T s)
        {
            Console.WriteLine("The value of s is :" + s);
        }
    }

    class Program
    {
        static void Main (string [] args)
        {
            Sample.Display <int>(10);
            Sample.Display <string>("welcome");
            Console.Read();
        }
    }
}
```

Example:- Goto console application.

```
namespace ConsoleApplication2
{
    public class Sample
    {
        public static void Display <T1,T2>(T1 a , T2 b)
        {
            Console.WriteLine(a + " " + b);
        }
    }

    class Program
    {
        static void Main (string [] args)
        {
        }
    }
}
```

```
Sample.Display<int, int>(10, 10);
Sample.Display<int, string>(10, "abc");
Console.Read();
```

Example :-

→ goto console application  
namespace ConsoleApplication6

```
{ public class Sample <T>
{
    public void display(T a)
    {
        Console.WriteLine("The value of a is :" + a);
    }
}

static void (string [] args)
{
    Sample<int> obj1 = new Sample<int>();
    obj1.Display(10);

    Sample<string> obj2 = new Sample<string>();
    obj2.Display("Welcome");

    Console.Read();
}
```

Example 2 public class Sample <T>

```
{ Public Void Display(T a)
{
    Console.WriteLine("The value of a is :" + a);
}
```

```

class program
{
    static void main (String [] args)
    Sample <int> obj1 = new Sample <int> ();
    obj1.Display (10);
    Sample <String> obj2 = new Sample <String> ();
    obj2.Display ("abc");
    Console. Read ();
}

```

## ADO.NET Entity Models.

### Entity Framework :-

- 1) Enterprise Development.
- 2) works with Conceptual model of database
- 3) works with all data sources.
- 4) ".EDMX" is created by using entity framework.

### LINQ

- 1) Rapid application development
- 2) works with object in database
- 3) mainly work with SQL Server
- 4) .DBML is created by using Linq to SQL.
- 5) Entity framework is more targeted towards interface development where the schema is usually optimized for storage considerations, like performance consistency and partition.

Entity framework is design around exposing an app oriented data model ie loosely coupled and may differ from the existing database schema.

- \* for ex. u can map a single entity (class) to multiple or map multiple entities to the same table.  
Entity framework has ".edmx" (ADO.NET entity model) file when added in the application.
- \* Linq to sql mainly has features to support rapid application development against sql server.
- \* Linq to sql allows you to have a strongly typed view of your existing database schema. you can build Linq queries over table and return result as strong type object.  
Linq to sql has ".dbml" (Linq to sql) file when added in the app. you can use Linq to sql by decorating the existing classes with the attribute

#### \* Linq -To- Sql :-

use this framework if you plan on editing a one-to-one relationship of your data in your presentation layer. meaning you don't plan on combining data from more than one table in any one view or page.

#### → entity framework :-

use this framework if you plan on combining data from more than 1 table in your view or page.

To make this clearer, the above terms are specific to data that will be manipulated in your viewer page, not just display. This is important to understand.

With the entity framework you are able to "merge" table data together to present to the presentation layer in an editable form. and then when that form is submitted. EF will know how to update all the data from the various tables.

There are probably more accurate reason to choose EF over L2S, but this this would probably be the easiest one to understand. L2S does not have ~~not~~ the capability to merge data for view presentation.

## ADO.NET

- ADO.NET entity framework abstracts the relational schema of the data i.e. stored in a database and presents its conceptual schema to the application.
- This abstraction eliminates the object-relational impedance mismatch that is otherwise common in conventional database-oriented programs.
- For example, in a conventional database-oriented system, entries about a customer and their information can be stored in customers table, their orders in orders table and their contact information in another contact table.
- The app that deals with database must know which info is in which table i.e. the relational schema of the data is hardcoded into app.
- The disadvantage of this approach is that if this schema is changed, the application is not shielded from the change. Also the application has to perform SQL joins to traverse the relationships of the data element in order to find related data for ex.  
ex - To find the orders of a ~~certain~~ certain customer to be selected from the customers table, the customers table needs to be joined with the orders table and the joined tables need to be queried for the orders that are linked to the customer.
- This model of traversing relationship between Henry is very different from the model used in object-oriented programming languages where the relationships of an object feature are exposed as properties of the object and accessing property traverses relationship. Also using SQL queries exposes as string only to have it processed by the database, it's the programming language ~~from~~ ~~the~~ programming language from making any guarantee about the operation & from providing

→ The mapping of logical schema into the physical schema that define how the stored on the data is structure & stored on the disk is the job of the database system & client. The database system & client side data access mechanism are shielded from it as the database expose the data in the way specified by its logic schema.

### History :-

version 1.0 entity framework (EFV1) was included with .NET framework 3.5 service pack 1 and visual studio 2008 service pack 1, released on 11 August 2008. This version has been widely criticized, even attracting a 'vote of no confidence' signed by several hundred developers. The second version of entity framework, named entity framework 4.0 (EFV4), was released as part of .NET 4.0 on 12 April 2010 and has addressed many of the criticisms made of version 1.

L. A third version of entity framework, yet unnamed, is planned for release in 1st quarter of 2011. There have been several betas (called community technology preview) of it and the last CTP was released in December 2012.

### Entity data model :

The entity data model (EDM) specifies the conceptual model (CSDL) of the data via the Entity-Relationship data model which deals primarily with entities and the associations they participate in. The EDM schema is expressed in the schema definition of XML. In addition, the mapping (MSL) of the elements of the conceptual schema (CSDL) to the storage schema (SSDL) must also be specified. The mapping specification is also expressed in XML.

Visual Studio also provide entity designer, for visual creation of the EDM and the mapping specification. The UI of the tool is the XML file (\*.EDMX) specifying schema & mapping.

EDMX file contains an metadata artifact (CSDL | ms | ssp1 content.) These 3 files (CSDL, ms, ssp1) can also be created or edited by hand.

## Linq

Ques Console application :-

```
static void main (string [] args)
{
    List<int> obj = new List<int>() { 1, 2, 3, 4, 5,
                                         6, 7, 8, 9, 10};
```

var ~~re~~ ~~for~~ = from n in obj

where n % 2 == 0

select n;

```
Console.WriteLine ("The even numbers are :");
```

```
foreach (var num in x)
```

```
{}
```

```
    Console.WriteLine (num + " ");
```

```
}
```

```
Console.Read();
```

Example 2 :- goto console applications

```
namespace ConsoleApplication1
```

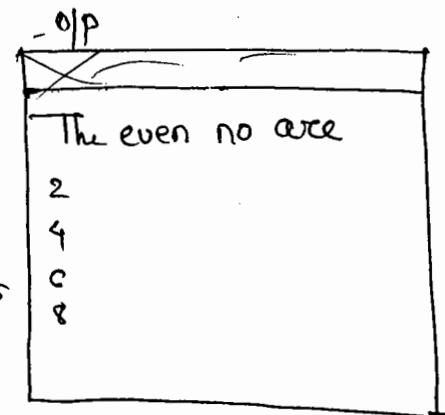
```
{
```

```
    public class Employee
```

```
{
```

```
    public int Empno { get; set; }
```

```
    public string Ename { get; set; }
```



```
public string job {get; set;}  
public int sal {get; set;}  
public string addr {get; set;}  
public int deptno {get; set;}
```

```
}
```

```
public static void class program
```

```
{
```

```
    static void main(string [] args)
```

```
{
```

```
        List<Employee> obj = new List<Employee>()
```

```
{
```

```
        new Employee (empno = 1001, ename = "Balaji", job =  
            "manager", sal = 5000, addr = "hyd" deptno = 10},  
        new Employee (empno = 1002, ename = "Harish", job =
```

```
            "manager", sal = 4000, addr = "hyd" deptno = 20},  
        new Employee (empno = 1003, ename = "Teja", job =
```

```
            "Sales manager", sal = 6000, addr = "hyd" deptno = 10},  
        new Employee (empno = 1004, ename = "Kalyan", job =
```

```
            "Sales", sal = 7000, addr = "hyd" deptno = 10}
```

```
};
```

```
Var rec = from n in obj
```

```
where n.deptno == 10
```

```
Select n;
```

```
Console.WriteLine("Deptno 10 employee are:");
```

```
Console.WriteLine ("Empno\tename\tsalary \tjob\taddress  
\tdeptno\n");
```

```
foreach (var rec in x)
```

```

        {
            Console.WriteLine(rec.Empno + "\t" + rec.Ename + "\t" +
                rec.Sal + "\t" + rec.Job + "\t" + rec.Add + "\t" +
                rec.Deptno + "\n");
        }
    }

    Console.Read();
}

```

### Example -

- \* Open Visual studio → select template windows forms app →
- \* goto SQL explorer → right click on the application → add new item → select category data → select template ADO.NET Entity data model name it as Sample.edmx click → add button.
- \* Select generate from database click next button → goto new connection → enter server name → activate sql server authentication radio button user name =sa , password=123 , enter database name click ok button.
- \* Activate ~~is~~ is include the sensitive data in the connection string radio button click next button goto step → goto DBO → activate table checkbox → click finish button.

\* goto windows form

\* goto tool box → select gridview control → drag & drop in the form.

private void button1\_Click(object sender --

{

    myDBEntities obj = new myDBEntities();

    dataGridView1.DataSource = obj.Emps.ToList();

}

[Emps in innumeric format that's why we need ToList method]



Example :-

Create table Bank (accno int primary key, cname varchar(30),  
bal int)

Refer prev ex steps.

private void button1\_Click(object sender,

{

masterEntities obj = new masterEntities();

Bank b1 = new Bank();

b1.accno = int.Parse(textBox1.Text);

b1.Cname = textBox2.Text;

b1.bal = int.Parse(textBox3.Text);

obj.Banks.Add(b1);

obj.SaveChanges();

MessageBox.Show ("Record added");

Form

Accno :	<input type="text"/>
Cname :	<input type="text"/>
Balance :	<input type="text"/>
<input type="button" value="Insert"/>	

## Boxing & UnBoxing.

Boxing :-

Converting a variable value into object type value implicitly.

Unboxing :-

Converting a object type value into variable explicitly  
(using conversion methods)

Console Application :-

static void Main (string [] args)

{

int x = 10;

object obj;

// boxing.

obj = x;

Console.WriteLine (obj);

// Unboxing

```
x = Convert.ToInt16(obj);
Console.WriteLine(x);
Console.Read();
}
```

## Enumerations :-

An Enumerations is a collection of constants . that means you can create your own set of named constants by using Enumerations.

- Each constant will have a name which an integer value

Syntax for enumeration declaration.

```
Public enum enumname
```

```
{
```

```
    Constant1 = Val1, Constant2 = Val2 ---
```

```
}
```

Syntax for Usage :-

```
enumname.constantName
```

Example - go to console app10

```
namespace ConsoleApplication4
{
    public enum months
    {
        january = 1, january = 1, feb = 2, march = 3, april = 4, may = 5;
    }
    class Program
    {
        static void Main(string[] args)
        {
        }
    }
}
```

```

Console.WriteLine(months.mar);
Console.WriteLine((int)months.mar);
Console.Read();
}
}
}

```

## Arrays

- An array is an user defined ~~data~~ data type used to store more than one values with same name. each value is identified using the index, always 1st index of array starts with '0' and is known as lower bound of array.
- Always last index of array ends with size '-1' of array if is known as upper bound of array.
- In c#.net arrays are reference values types.

### Types of Array.

#### 1) Single Dimensional Array.

An array which contain 1 row or one column is known as single dimensional array.

#### 2) Multidimensional Array:-

An array which contains more than 1 row and more than 1 column is multidimensional array.

#### 3) Jagged Array :-

→ An array which contains another array within it is known as jagged array.

→ A jagged array is known as array of arrays.

### Implementation of Array:-

#### Single Dimensional Array :-

##### Array Declaration

##### Without Initialization :

datatype [] Arrayname = new datatype [size];

With Initialization :-

```
datatype [] arrayname = {val1, val2, val3, ...};
```

Accessing the Element :-

```
arrayname [index]
```

Double Dimensional Array :-

Array declaration ;

without initialization ;

```
datatype [,] arrayname = new datatype [rows size, column size];
```

with initialization :-

```
datatype [,] arrayname = {{val1, val2, ...}, {val1, val2, ...}, ...};
```

Accessing Accessing the elements :-

```
arrayname [row index, column index].
```

Ex : To print array elements on the screen .

\* goto Console applications .

```
class Program
```

```
static void main (string [] args)
```

```
{
```

```
int [] a = new int [] {10, 20, 30, 40, 50, 60,};
```

```
Console.WriteLine ("The elements of array :");
```

```
for (int i=0; i<6; i++)
```

```
{
```

```
Console.WriteLine (a[i] + " ");
```

```
}
```

```
Console.Read();
```

```
}
```

Ex - To print string array element on the screen

\* goto Console application .

```
static void main (string [] args)
```

```
{
```

```
string [] s = new string [6] {"balaji", "teja", "kalyan", "abc",  
"Urvashi", "Dipali"};
```

```

Console.WriteLine ("The array element are :");
for (int i=0; i<6; i++)
{
    Console.WriteLine (s[i] + " ");
}
Console.ReadLine();
}

```

Ex: Using for each loop :- → goto console app1

```

static void main (string [] args)
{
    string [] s = new string [6] {"Balaji", "teja", "Kalyan", "abc",
                                 "Ujwala", "Dipali"};
    Console.WriteLine ("The array element are :");
    foreach (string obj in s)
    {
        Console.WriteLine (obj);
    }
    Console.ReadLine();
}

```

Example - console application.

```

static void main (string [] args)
{
    int [] a = new int [6] {1, 2, 3, 4, 5, 6};
    int [] b = new int [12] {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} , 13, 14, 15};
    Console.WriteLine ("Size of array a is :" + a.Length);
    Console.WriteLine ("Size of array b is :" + b.Length);
    Console.WriteLine ("Rank of a is :" + a.Rank);
    Console.WriteLine ("Rank of b is :" + b.Rank);
    Console.WriteLine ("Element of array a is : ");
    foreach (int i in a)
    {
    }
}

```

```

        Console.WriteLine(i + " ");
    }

    Console.WriteLine("Elements of array b is :");
    foreach (int i in b)
    {
        Console.WriteLine(i + " ");
    }
    Console.Read();
}

```

o/p

Size of array a is → 6
Size of array b is → 12
Rank of array a is 1 (dimension)
Rank of array b is 1 (dimension)
Element of array a is :
12 34 56
Element of array b is :
123456789101112

Example :- goto Console Application.

```

static void (main [] args)
{
    int [][] a = new int [3] [];
    a[0] = new int [5] {1,2,3,4,5};
    a[1] = new int [4] {10,20,30,40};
    a[2] = new int [6] {2,4,5,6,7,8};

    Console.WriteLine("Element of jagged arr :");
    foreach (int [] j in a)
    {
        foreach (int i in j)
        {
            Console.WriteLine(i + " ");
        }
        Console.WriteLine();
    }
    Console.Read();
}

```

4/12/13  
Wednesday

## Windows Services :-

Eventlog = class

- 1) A windows service like a background process or appln.
- 2) It will be installed on the "Services" & will be registered on the windows registry.
- 3) To open the list of currently installed services on your system, click on "Start" → "Control Panel" → Administrative Tools → services.  
Eg :- Windows Time, Windows Audio, plug & play, casting services, anti-virus services.

### Futures of Windows Services

- 1) The windows services is configured to run on windows os's only.
- 2) Most of the windows service will be started automatically at System logon & will be shut-down at system logoff.
- 3) Some other windows services can be configured to be started manually. To start stop the services, open "Services" window as shown above. Then right click on the required device & choose "start" / "stop" option.
- 4) The windows services are best compatible with "Threading" so that you are recommended to write your code in "threading".
- 5) The windows services can't contain any windows forms.
- 6) It is able to show a tray on the system tray (at window taskbar)
- 7) It can't be run directly on your visual studio. It is able to interact with "Eventlog" service on the windows os.
- 8) After the development of your application, if your application code in the windows service, you need a utility software called "installutil.exe", which is used to install your windows services on the system,
- 9) Before uninstalling the Services it requires "ProjectInstaller" class, which provides additional information, i.e. required for uninstalling your services with "installutil.exe".

Create windows services.  
Open Visual studio. Select template windows service →  
specify name and location click ok button →  
\* goto toolbar → select notifyicon control drag & drop  
in the service.  
\* goto notifyicon control properties  
1) BalloonTipIcon = info / warning / error.  
2) BalloonText = This is service.  
3) BalloonTipTitle = NewService  
4) Icon - click brows button set any icon.  
\* go to sol<sup>9</sup> explorer → right click on the references add  
reference → go to assembly → activate → system.Dragging  
checkbox → click ok button.  
\* press F7 then automatically goto code window.  
using System.Threading.

```
void writemessage ()  
{  
    while (true)  
    {  
        string msg = DateTime.Now.ToString();  
        System.Diagnostics.EventLog.WriteEntry (msg, "This is user  
        defined msg")  
        Thread.Sleep (3000);  
    }  
}  
Thread th;  
protected override void OnStart (string l)  
{  
    th = new Thread (writemessage);  
    th.Start ();  
}  
protected override void Onstop ()  
{  
    th.
```

The event handler for ~~constant~~ or ~~nonstop~~ will be created automatically. The `onstartevent` will be executed on the starting the windows service & the `onstop` event will be executed on stopping the windows service.

→ Then we need to add installer for your windows service.

→ ~~do this simply comeback to the design view~~ right click and choose add installer option then you will have project installer class in your project along with two components called.

→ Service process installer

— Service installer

→ go to service<sup>process</sup> 1 installers prop → change the account type local system.

→ go to service installers property enter description, enter display name select start type automatic

→ goto menubar → goto build → build windows service  
Installing the service.

→ goto start → goto programs → goto microsoft visual studio 2012 → goto vs tools → goto vs command prompt.

Installutil specify the path.

See the service

→ goto start → goto control panel → goto system & security → goto administrative tool → goto services.

→ goto our service → right click start button.

5/12/13

Go to Console application.

```
class sample
{
    public void m1()
    {
        Console.WriteLine("Hello");
    }

    public void m2()
    {
        Console.WriteLine("Welcome");
    }
}
```

class program

```
{ static void Main(string[] args)
{
    Sample obj = new Sample();
    obj.m1();
    obj.m2();
    Console.ReadLine();
}}
```

goto build → build solution

\* goto start → program → goto visual studio 2012 → goto  
vs tools → goto vs command path → iildasm (intermediate  
language disassembler.) specify the path. → presenter  
get the msil code window.

## Global Assembly Cache

```
goto Class Library  
namespace ClassLibrary1  
{  
    public class Test  
    {  
        public string getset()  
        {  
            return "welcome";  
        }  
    }  
}
```

go to solution explorer → go to properties → go to signing → activate signing assembly checkbox → choose a strong name key file → goto new → enter keyfile name → deactivate protect my key file with a password checkbox → click ok

goto build → build class library1

open c:\drive → go to assembly folder → drag & drop our dll file into assembly.

Complete

---

