CHAPTER 1

UNIT I

.Net Framework

Syllabus Topics

The .NET Framework

.NET Languages ,Common Language Runtime, .NET Class Library

C# Language Basics

Comments Variables and Data Types, Variable Operations, Object-Based Manipulation, Conditional Logic, Loops, Methods, Classes, Value Types and Reference Types, Namespaces and Assemblies, Inheritance, Static Members, Casting Objects, Partial Classes.

ASP.NET

Creating Websites, Anatomy of a Web Form-Page Directive, Doctype, Writing Code-Code Behind Class, Adding Event Handlers, Anatomy of an ASP.NET Application - ASP.NET File Types, ASP.NET Web Folders.

HTML Server Controls

View State, HTML Control Classes, HTML Control Events, HtmlControl Base Class, HtmlContainerControl class, HtmlInputControl Class, Page Class, global.asax File, web.config File.

Syllabus Topic : The .Net Framework

1.1 Introduction to .Net Framework

Q. What is .NET framework ? **(5 Marks)**

- **.NET Framework** is a software framework developed by Microsoft that runs on Microsoft Windows.
- It includes a large class library named Framework Class Library (FCL) and provides language interoperability across several programming languages; that means each language can use code written in other languages.
- Programs written for .NET Framework execute in a software environment named Common Language Runtime (CLR), which is virtual machine that provides services such as security, memory management, exception handling etc.
- Framework Class Library (FCL) and Common Language Runtime (CLR) collectively create .NET Framework.
- Framework Class Library (FCL) provides features like User interface, Data Access, Database Connectivity, Cryptography, Web Application Development, Numeric Algorithms, and Network Communication.
- Programmers produce software by combining application code with .NET Framework and other libraries.
- .NET framework is used by most of the new software build created for the Windows platform.
- Microsoft has produced an integrated development environment mainly for .NET software called Visual Studio.
- .Net Framework provides following features:
  1. An object oriented environment to develop applications.

2. Code execution environment which helps in deployment and versioning.
3. Security to the code which is executing.
4. Language compatibility.

- In .Net user has choice to select the language in which he/she wants to develop the software.
- .NET is platform independent. It also supports language integration.

Advantages of .Net Framework

1. Good Design
2. It supports Object Oriented Programming features like class, object inheritance etc.
3. It has language compatibility.
4. Robust and Secure.
5. Supports window based application and web services.

1.1.1 .Net Framework Architecture

Q. What is .NET framework architecture ? (4 Marks)

- .Net Framework is a combination of CLR, FCL, ADO.Net and XML classes, Web/Window applications and Web services.

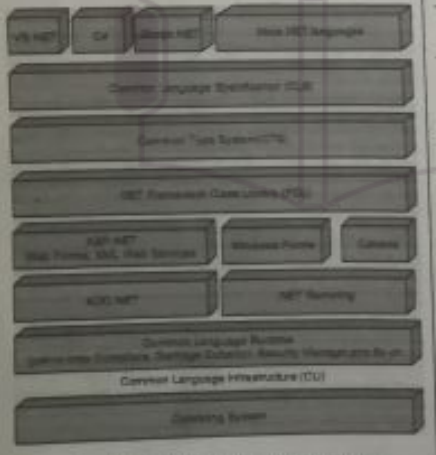


Fig. 1.1.1 : .NET Framework Architecture

- Net Framework is a platform that provides tools and technologies to built Window based as well as Web based applications.
- The .Net architecture is divided into various modules. Each module has its own roles and responsibilities.

Syllabus Topic : .NET Languages

1. .Net Supported Languages

Microsoft itself supports most of the differe languages. .NET is the software development platfo provided by Microsoft. .NET supports more than languages like VB, C#, C++, Jscript, J# etc.

2. Common Language Specification (CLS)

- A **Common Language Specification** (CLS) is document that tells how .NET language programs ca be converted into **Microsoft Intermediate Langua** (MSIL) code.
- CLS is a sub set of **Common Type System (CTS)** a it specifies a set of standards that required to b satisfied by all language compilers targeting CLR. helps in cross language inheritance and cross langua debugging.
- To fully interact with other objects without considerin their language, objects must expose to callers o those features that are common to all the languages.
- CLS specifies the rules and regulations for th languages so as to interact with other .Net languages.
- The CLS was created sufficiently big to conta the language constructs that are commonly required b developers.

3. Common Type System (CTS)

- **Common Type System** (CTS) defines some basic da types that Intermediate Language code ca understand.
- It explains set of data types that can be used in differe .Net languages in common, that means CTS giv guarantee that objects written in different .Ne languages can communicate with each other.
- Each .Net supporting language should match its da types to these standard data types.
- This makes it possible for two .Net supportin languages to communicate by sending/receivin parameters to and from each other.
- For Communicating the programs written in any .NE supporting language, the data types have to be wo suitable to each other on the basic level.
- The common type system supports two gener categories of types :

(i) Value types

- Value types directly hold their data, and instances of value types are either allocated on the stack or allocated inline in a structure.
- Value types can be built-in, user-defined, or enumerations.

(ii) Reference types

- Reference types are used to store a reference to the memory address of value, and reference types are allocated on the heap.
- Reference types may in the form of self-describing types, pointer types, or interface types. The reference type is usually decided by the values of self-describing types.
- Self-describing types are further divided into two part : arrays and class. The class types are user-defined classes, boxed value types, and delegates.

Syllabus Topic : .NET Class Library

4. .Net Framework Class Library (FCL)

- The .Net Framework Class Library (FCL) provides important functionality of .Net Framework. It is also called as Base Class Library. The .Net Framework Class Library (FCL) includes a huge group of reusable classes, interfaces, and value types that speed up the development process and provide access to in-built functionality. It is the foundation on which .NET Framework applications, components, and controls are built. It is common for all types of applications ,that means the way you access the Library Classes and Methods in VB.NET will be the same in C#.NET and it is common for all other languages in .NET.
- The following are different types of applications that can make use of .net class library.
  1. Window based Application.
  2. Console based Application.
  3. Web Application.
  4. XML Web Services.
  5. Windows Services.
- Developers only need to include the FCL in their language code to use predefined functions and properties of FCL to implement frequently used and complicated functions like reading and writing data to file, graphic rendering, data manipulation and XML document manipulation.

Syllabus Topic : Common Language Runtime

5. Common Language Runtime (CLR)

- .Net Framework provides software called **Common Language Runtime** (CLR) to run programs which are written in any .NET language.
- **Common Language Runtime** (CLR) is heart of the .Net Framework.
- CLR resides above the operating system and executes all .Net programs. It handles Garbage Collection, Code Access Security (CAS), Thread Management etc.
- The source code which executes under the **Common Language Runtime** (CLR) is called as Managed Code.
- Programmers need not to worry about managing the memory if the programs are executing under the CLR because CLR provides Memory Management and Thread Management features. When our program needs memory, CLR allocates the memory to that program and de-allocates the memory when program execution is completed.
- The main function of Common Language Runtime (CLR) is to modify the Managed Code into native code and then execute the Program.
- Language Compilers like C#, VB.Net, J# compiler will translate the Program to **Microsoft Intermediate Language** (MSIL)code and then CLR translate that MSIL code into native code.
- The **Microsoft Intermediate Language** (MSIL) code or Managed Code compiled only when it needed. The Common Language Runtime's Just In Time (JIT) compiler converts Intermediate Language (MSIL) to native code on demand at application run time.

Fig. 1.1.2 : Common language Runtime(CLR)

- **Microsoft Intermediate Language (MSIL) Code :** When we compile our .Net code, it is not directly transformed to native or binary code; it is first transformed into intermediate code known as **Microsoft Intermediate Language (MSIL) code** which is then interpreted by the CLR. MSIL is not dependent on hardware and the operating system. Cross language compatibility i.e. program from one language can convert easily into another language is possible because MSIL is the same for all .Net languages. MSIL is further converted into native code by CLR.
- **Just in Time Compilers (JIT) :** It compiles Intermediate Language (IL) code into native executable code (.exe or .dll). Once code is converted to Intermediate Language (IL) then it can be called again by JIT instead of recompiling that code.

1.2 C# Language Basics

- C# is an advanced, object-oriented as well as general purpose programming language developed by Microsoft.
- C# is approved by ECMA (European Computer Manufacturers association) and ISO (International Standards Organization).
- Anders Hejlsberg and his team developed C# in the phases of .Net Framework development. C# is considered as an important language among the .Net Framework languages.

Features of C#

(i) It is a latest programming language providing advanced features.
(ii) It is general-purpose
(iii) It is object oriented.
(iv) It is component based.
(v) It is simple to learn.
(vi) It is a structured language.
(vii) It provides efficiency to programs.
(viii) It can be compiled as well as executed on a various types of computer platforms.
(ix) It is a part of .Net Framework.

Strong Programming Features of C#

C# follows number of constructs from traditional high-level languages such as C and C++ and supports object-oriented paradigm. Maximum of C# concepts strongly resembles to Java. C# provides numerous strong programming features because of which it is widely used.

Important features of C#

(i) Automatic Garbage Collection
(ii) Standard Library
(iii) Assembly Versioning
(iv) Properties and Events
(v) Simple Multithreading
(vi) Integration with Windows
(vii) Delegates and Events Management
(viii) Easy-to-use Generics
(ix) Indexers
(x) Conditional Compilation

Syllabus Topic : Comments

1.2.1 Comments

Q. Explain Comments in C#. (4 Marks)

Comments can be used in document to state the functionality of the program and purpose of specific lines or lines of code. As the compiler ignores the comments, we can include them anywhere in a program without affecting the source code. There are 3 types of comments in C#.

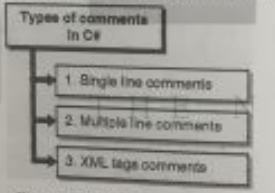


Fig. C1.1 : Types of comments in C#

1. Single line comments

Double slash (//) is used to represent single line comment.

Example

```
// this is single line comment
```

2. Multiple line comments

It is used to write more than single line as comment. Multiline comment start with /* and ends with */

Example

```
/* This is multiline comment in c# which is simply
ignored by compiler.
It contain more than one line */
```


3. XML tags comments

In C#, you can create documentation for source code by inserting XML elements as special comment fields using triple slash before the code block to which the comments refer in the source code.

Example

```
/// <Information>
/// This class performs important function.
/// Information>
```

Syllabus Topic : Variables and Data Types

1.2.2 Variables and Data Types

Q. What is variable? (2 Marks)
Q. Explain data types in C#. (4 Marks)

1.2.2(A) Variables

- A variable is nothing but a name given to a memory location on which our program works.
- Variable is a way to represent memory location using symbol so that it can be easily identified.
- Every variable in C# has a particular data type, that states the amount of memory required to represent that variable, the range of values which can be stored in that memory and the set of operations that can be applied on that variable.
- The Fundamental variable types available in C# can be categorized as :

Sr. No.	Variable Type	Example
1.	Decimal types	Decimal
2.	Boolean types	True or false value, as assigned
3.	Integral types	int, char, byte, short, long
4.	Floating point types	float and double
5.	Nullable types	Nullable data types

Defining Variables

Syntax

```
Data-type variable-list;
```

Here,

data_type: data_type contains any valid data type in C#, like Int, Float, Char,Double, or any user-defined data type.

variable_list: This variable_list includes one or more variable names which are separated using commas.

Rules for defining variables

1. A variable can have alphabets, digits and underscore.
2. A variable name can start with alphabet and underscore only. It can't start with digit.
3. No white space and comma is allowed within variable name.
4. A variable name must not be any reserved word or keyword e.g. char, float ,if, else, switch ,goto, break, while, for etc.

Example of declaring variable

1. int p, q; 2. double d;
3. float f; 4. char ch;

Invalid variable names

1. int 4; 2. int x y;
3. int double;

Initializing Variables

- We can initialize variable i.e. assign a value to that variable using equal to (=) sign.
- Syntax for initialization of variable is :

variable_name=value;

Example

```
//declaration
int a,b;
/* initialization */
a = 10;
b = 20;
```

- Variables can be initialized in their declaration also. The initialization of variable consists of an equal sign followed by a constant expression as follows:

```
Data-type variable_name = value;
```

- Some examples are :

1. int d =3, f =5; 2. double pi =3.14159;
3. char x='x';

1.2.2(B) Data Types

C# is a strongly typed language, it means that we cannot use variable without data types. Data types tell the compiler that which type of data is used for processing. The variables in C# are categorized into the following types :

1) Value types 2) Reference types
3) Pointer types

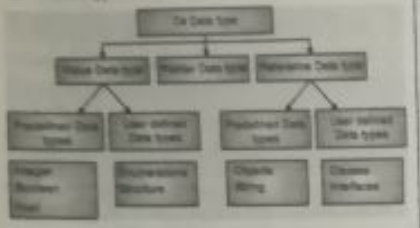


Fig. 1.2.1 : Data Types of C#

Types	Data Types
Value Data Type	int, char, float, Boolean, etc.
Reference Data Type	String, Class, Object and Interface
Pointer Data Type	Pointers

1. Value Data Type

- Value type variables can be assigned a value directly. They are derived from the class **System.ValueType**.
- The value data types are integer-based and floating-point based. The value types directly contain data. C# language supports both signed and unsigned literals.
- There are 2 types of value data type in C# language :
  - (I) **Predefined Data Types :** It includes Integer, Boolean, Float, etc.
  - (II) **User defined Data Types :** It includes Structure, Enumerations, etc.
- The memory size of data types may change according to 32 or 64 bit operating system.
- Following are the value data types. Size is given according to 32 bit Operating System.

Sr. No.	Data Types	Memory Size	Range
1.	Char	1 byte	– 128 to 127
2.	signed char	1 byte	– 128 to 127
3.	unsigned char	1 byte	0 to 127
4.	Short	2 byte	– 32,768 to 32,767
5.	signed short	2 byte	– 32,768 to 32,767
6.	unsigned short	2 byte	0 to 32,767
7.	Int	2 byte	– 32,768 to 32,767
8.	signed int	2 byte	– 32,768 to 32,767
9.	unsigned int	2 byte	0 to 32,767
10.	short int	2 byte	– 32,768 to 32,767
11.	signed short int	2 byte	– 32,768 to 32,767
12.	unsigned short int	2 byte	0 to 32,767
13.	long int	4 byte	
14.	signed long int	4 byte	
15.	unsigned long int	4 byte	
16.	float	4 byte	
17.	double	8 byte	
18.	long double	10 byte	

2. Reference Type

- The reference types do not contain the actual data stored in a variable, but they contain a reference to the variables i.e. address of variable. If the data is changed by one of the variables, the other variable automatically reflects this change in value.
- There are 2 types of reference data type in C# language :
  - (i) **Predefined Types :** such as Objects, String.
  - (ii) **User defined Types :** user defined data types like Classes, Interface which we see later.

i) **Predefined Types**

a) **Object Type**

- The **Object Type** is the ultimate base class for all data types in C# Common Type System (CTS). i.e.in the unified type system of C#, all types predefined and user-defined, reference types and value types are inherited directly or indirectly from Object.
- The object types can be used to assign values of any other types such as value types, reference types, predefined or user-defined types.

- Before assigning values, we must required to do type conversion.
- The process of converting value type to object type is known as **boxing**.
- And the process of converting object type to a value type is known as **unboxing**.
- In the following example, the integer variable i is boxed and assigned to object o.

Example

```
int i = 123;
// The following line boxes i.
object o = i;
```

- The object o can then be unboxed and assigned to integer variable i :

```
i = (int)o; // unboxing
```

b) **String Type**

- The **String Type** permits us to assign any string type value to a variable. The string type is an alternative for the System.String class. It is inherited from object type.
- for a string type ,we can provide value through string literals using two ways quoted and @quoted.

Example

```
String name="C# Example";
```

- Example of @quoted way

```
@" C# Example"
```

ii) **user-defined data types**

The user-defined data types are: class, interface, or delegate. We will see them later.

iii) **Pointer Data Type**

- Pointer type variable store the memory address of variable.
- Pointer in C# is same as pointer in C or C++.

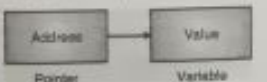


Fig. 1.2.2

- Ampersand sign(&) is used to assign address of variable to pointer which is called as Address of operator and asterisk sign(*) is used to access the value stored at that address which is called as Indirection operator.

Syntax

```
Data-type *identifier;
```

Example

(i) char *cptr;

(ii) int *iptr;

1.2.3 C# Constants

Q. What is constant ? Explain types of constants in c#. (4 Marks)

- The constant refers to variable whose value cannot be modified while throughout the execution of program.
- We cannot be assign value to constant variable at run time, we must assign value to constant at compile time i.e. at time of the declaration of constant variable.
- Constants can declared by using the const keyword.
- The fixed values are also called literals.
- Constants can be of any of the basic data types like an integer constant, a floating constant, a character constant, or a string literal.
- Following are the types of constants in C#

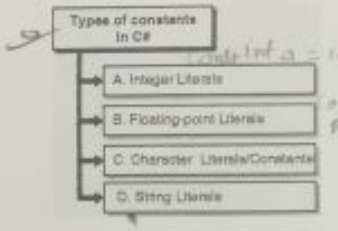


Fig. C1.2 : Types of constants in C#

(A) Integer Literals

- Decimal or octal or hexadecimal constant are used to represent integer literal.
- Prefix is used to state the base for integer literal. 0x or 0X is used as prefix for hexadecimal and we can not add decimal literal.

- An integer literal can have a suffix that is a combination of U or u and L or l. U or u is used to indicate unsigned integer and L or l for long integer.

Examples

(i) 55 (decimal integer constant)
(ii) 0x2a (hexadecimal integer constant)
(iii) 15u (unsigned integer constant)
(iv) 234l (long)
(v) 22ul (unsigned long)

(B) Floating-Point Literals

Integer part , decimal point, a fractional part, and an exponent part are included in floating-point literal . Sometimes floating-point literal consist of expoential part which is denoted by E or e.

Examples

(i) 3.14159
(ii) 314159E-5F

(C) Character Literals/Constants

- Character literals are written in single quotes.
- A character literal can be a plain character such as 'a' or an escape sequence such as '\t','\n' etc.
- There are some characters in C#. When they starts with backslash ,they have specific meaning and these characters are called as escape sequence character.

Examples

List of escape sequence character are as follows :

Sr. No.	Escape sequence	Meaning
1.	\\	Used to represent \ character
2.	\'	Used to represent ' character
3.	\"	Used to represent " character
4.	\?	Used to represent ? character
5.	\a	Used to represent Alert or bell
6.	\b	Used to represent Backspace
7.	\f	Used to represent Form feed
8.	\n	Used to represent Newline
9.	\r	Used to represent Carriage return
10.	\t	Used to represent Horizontal tab
11.	\v	Used to represent Vertical tab
12.	\xhh . . .	Used to represent Hexadecimal number of one or more digits

(D) String Literals

- String literals are written in double quotes " " or can written using @" ".
- A string is array characters. A string contains charac such as plain characters, escape sequences, universal characters.
- You can split a long line into multiple lines using st literals and this is done by separating the long using whitespaces.

Example

1) "hello dear"
2) "hello dear"
3) @"hello dear"

Declaring the constant

Syntax

```
const <data_type> <constant_name> = value;
```

Example

(i) int const a=10;
(ii) int const b=20;

Program 1.2.1

Write a program to demonstrate how to define and use a constant in program.

Solution :

Program that demonstrates how to define and use a constant in program

```
using System;
```

→ the using keyword is used to add the System namespace in our program. A program usually contains more than one using statement.


```
namespace ConstantDeclarationDemo
{
Class constantDemo
{
Static void Main(string[] args)
{
const double pi =3.14159;
double r;

Console.WriteLine("Enter Radius: ");

r =Convert.ToDouble(Console.ReadLine());

double areaofCircle = pi * r * r;
Console.WriteLine("Radius of circle : {0}, Area of circle: {1}", r, areaofCircle);
Console.ReadLine();
}
}
}
```

→ This statement represent namespace declaration. namespace is a collection of classes.

→ declaration of constant

→ This function is used to print message to console

→ Console.ReadLine() function is used to read input from user and Convert.ToDouble() convert that input into double datatype

→ Display parameter and first argument as result

utput

```
Enter Radius:
4
Radius: 4, Area: 50.26544
```

Syllabus Topic : Variable Operations

2.4 Variable Operations

1. Explain types of operators in C#. (4 Marks)

An operator is a symbol that tells the compiler to perform specific mathematical or logical operation or action on operand.

Operand is variable or literal on which operator can be applied to perform specific task.

C# has large set of built-in operators.

Following are types of operators which perform different types of operations on variable in C# language.

- Basically operators are classified in following three types on basis of number of operands they require to perform operation.

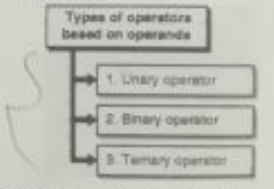


Fig. C1.3 : Types of operators based on operands

1. Unary operator

Operator which require only one operand to operate is called as Unary operator. For e.g. pre increment(++a), post increment(a++), predecrement(--a), postdecrement(a--) etc.

2. Binary operator

Operator which require two operands to operate is called as binary operator. For e.g. +,- etc.

3. Ternary operator

Operator which require three operands to operate is called as Ternary operator. e.g. conditional operator (?:).

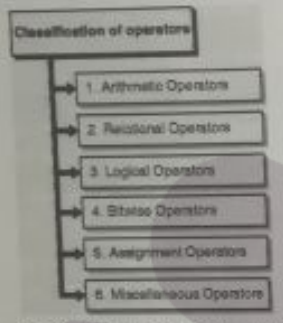


Fig. C1.4 : Classification of operators

1) Arithmetic Operators

Arithmetic operators are binary operators which require two operands to perform operation. Table 1.2.1 contain all the arithmetic operators available in C#. Assume x =10 and y=20 then

Table 1.2.1

Sr. No.	Operator	Description
1.	+	Used to adds two operands
2.	-	Used to Subtracts second operand from the first operand
3.	*	Used to do multiplication of two operands
4.	/	Used to divide numerator by de-numerator and find quotient
5.	%	This operator is called as Modulus operator and it is used to calculate remainder after division of two operands
6.	++	This is Increment operator which is unary operator used to increase value of integer by one
7.	--	This is Decrement operator which is unary operator which decreases value of integer by one

2) Relational Operators

Relational operators are binary operators i.e Rela operators requires two operators to perform task. table display all the relational operators availa C#. Assume x =10 and y = 20, then

Table 1.2.2

Sr. No.	Operator	Description	Exam
1.	==	It is used to Check wheather the values of two operands are equal or not. if values of two operands are equal then condition evaluats to true.	(x == y) true.
2.	!=	It is used to Check wheather values of two operands are equal or not, if values of two operands are not equal then condition evaluats to true.	(x != y) is
3.	>	It is used to Check wheather value of left operand is greater than the value of right operand.if left operand is greater than the value of right operand then evaluated to true.	(x > y)
4.	<	It is used to Check wheather value of left operand is less than the value of right operand. if value of left operand is less than the value of right operand then condition evaluated to true.	(x < y) is
5.	>=	It is used to Check wheather value of left operand is greater than or equal to the value of right operand.if yes then condition evaluated to true.	(x >= y) true.
6.	<=	It is used to Check wheather value of left operand is less than or equal to the value of right operand. if yes then condition evaluated to true.	(x <= y)

3) Logical Operators

Table 1.2.3 display all the logical operators present in C#. In logical operators includes operators such as logical And (&&), logical or (||) which are are binary operators and logical not(!) which is unary operator . Assume variable X contains Boolean value true and variable Y contains Boolean value false, then

Table 1.2.3

Sr. No.	Operator	Description	Example
1.	&&	This operator is known as Logical AND operator. If both the operands contains non zero value then condition evaluated to true.	(X && Y) is false.
2.	||	This operator is known as Logical OR operator. If any one operand from two operands has non zero value then condition evaluated to true.	(X || Y) is true.
3.	!	This operator is known as Logical NOT operator. It is used to reverses the logical state of the operand. If a condition is true then Logical NOT operator will convert it to false.	

4) Bitwise Operators

Bitwise operator operate on single bit at a time and carry out bit by bit operation. The truth tables for &(AND), |(OR), and ^(XOR) are as follows

A	b	a & b	a | b	a ^ b
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

The Bitwise operators which are available in C# are given in the below Table 1.2.4. Assume variable X holds 14 (binary equivalent: 0000 1110) and variable Y holds 11 (binary equivalent: 0000 1011), then

Table 1.2.4

Sr. No.	Operator	Description	Example
1.	&	This Binary AND Operator copy single bit at a time to the result if that bit presents in both operands.	(X& Y) = 10, which is 00001010
2.	|	This Binary OR Operator copy single bit at a time to the result if that bit presents in either operand.	(X| Y) = 15, which is 00001111
3.	^	This Binary XOR Operator copy single bit at a time to the result if that bit is set in one operand but in not in both.	(X^ Y) = 5, which is 00000101
4.	~	This Ones Complement which is unary Operator has the effect of 'flipping' bits.i.e convert 1 to 0 and 0 to 1	(~X) = 241, which is 1111 0001
5.	<<	The Left Shift Operator is Binary operator. The value of left operand is moved to left by the number of bits stated using the right operand.	X << 2 = 56, which is 00111000
6.	>>	This is Right Shift Operator which is binary operator. The value of left operand is moved right by the number of bits specified by the right operand.	X >>2 = 3, which is 00000011

Assume if X = 14; and Y = 11; then in the binary format they are as follows :

X = 0000 1110

Y = 0000 1011

X&Y = 0000 1010

X|Y = 0000 1111

X^Y = 00000101

~X = 1111 0001

5) Assignment Operators

There are following assignment operators supported by C# are listed in Table 1.2.5.

Table 1.2.5

Operator	Description	Example
=	assignment operator which assigns value of right side operand to left side operand.	X= 20 assigns value 20 into X
-=	This is Subtract AND assignment operator which subtracts right operand from the left operand and store the result of subtraction in left operand.	X -= Y is same as X = X -Y
/=	This is Divide AND assignment operator which divides left operand with the right operand and store the result of division in left operand.	X /= Y is same as X = X / Y
+=	This operator is used to add right operand to the left operand and store the result in left operand.	X += Y is same as to X = X + Y
*=	This is Multiply AND assignment operator which perform multiplication of right operand with the left operand and store the result of multiplication in left operand.	X*= Y is same as X = X * Y
%=	This is Modulus AND assignment operator which calculate remainder using two operands and store the result in left operand.	X %= Y is same as to X = X % Y
<<=	This is Left shift AND assignment operator in which value of left operand is moved to left by the number of bits stated using the right operand and result is stored in left operand.	X<<= 3 is same as X = X <<3
>>=	Right shift AND	X >>= 2 is
&=	This is Bitwise AND assignment operator. This Operator copy single bit at a time to the result if that bit presents in both operands and this result is stored in left operand.	X &= 2 same as X= X &
^=	This bitwise exclusive OR assignment operator.	X ^= 2 same as X= X^
|=	This is bitwise inclusive OR assignment operator which copy single bit at a time to the result if it that bit is set in one operand but in not in both and result is stored in left operand.	X |= 2 same as X = X |

6) Miscellaneous Operators

There are some important operators which s sizeof, typeof and **Conditional operator (? :) are available in C#** are listed in Table 1.2.6.

Table 1.2.6

Sr. No.	Operator	Description	Example
1.	sizeof()	This operator is used to return the size of a data type.	sizeof(float) returns 4.
2.	typeof()	This operator is used to return the type of a class.	typeof(StringRe
3.	&	This operator is used to return the address of variable.	&x; returns address of variable.

Sr. No.	Operator	Description	Example
4.	*	This operator is used to represent the Pointer which store address of variable.	*x; creates pointer 'x'
5.	?:	This operator is known as Conditional Expression which is used to check condition.	x=10; y=20; x>y?x:y
6.	is	This operator is used to states whether an object is of a certain type or not.	String name; If(name is String) checks name is an object of the string class.
7.	as	This operator perform type casting without occoumce of an exception if the type casting fails.	Object obj = new StringReader("Tech Max"); StringReader r = obj as StringReader;

1.2.4(A) Precedence of Operators in C#

Q. What is mean by operator precedence ? (2 Marks)

- The precedence of operators specifies that which operator will be evaluated first and which next in an expression.
- The associativity specifies the operators direction to be operate, it may be left to right or right to left.
- The precedence and associativity of C# operators is given in Table 1.2.7.

Table 1.2.7

Sr. No.	Category (By Precedence)	Operator(s)	Associativity
1.	Unary	+ - ! ~ ++ -- (type)* & sizeof	Right to Left
2.	Additive	+ -	Left to Right
3.	Multiplicative	% *	Left to Right
4.	Relational	<><= >=	Left to Right
5.	Shift	<<>>	Left to Right
6.	Equality	== !=	Right to Left
7.	Logical AND	&	Left to Right
8.	Logical OR	|	Left to Right
9.	Logical XOR	^	Left to Right
10.	Conditional OR	||	Left to Right
11.	Conditional AND	&&	Left to Right
12	Null Coalescing	??	Left to Right
13.	Ternary	?:	Right to Left
14	Assignment	= *= /= %= += -= <<= >>= &= ^= |= =>	Right to Left

1.2.5 C# Keywords

Q. What is keyword and list some keyword in C# ? (2 Marks)

- Keywords are predefined sets of reserved words that have special meaning in a program.
- The meaning of keywords cannot be changed; neither can they be directly used as an identifier in a program.

- A list of Reserved Keywords available in C# programming language is given below :

abstract	base	as	bool	break	catch	case
byte	char	checked	class	const	continue	decimal
private	protected	public	return	readonly	ref	sbyte
explicit	extern	false	finally	fixed	float	for
foreach	goto	if	implicit	in	in (generic modifier)	int
ulong	ushort	unchecked	using	unsafe	virtual	void
null	object	operator	out	out (generic modifier)	override	params
default	delegate	do	double	else	enum	event
sealed	short	sizeof	stackalloc	static	string	struct
switch	this	throw	true	try	typeof	uint
abstract	base	as	bool	break	catch	case
volatile	While					

Syllabus Topic : Object-Based Manipulation

1.2.6 Object-Based Manipulation

C# is a pure object oriented programming languages. C# supports all the object oriented programming concepts like :

1. **Object :** Object is an entity having its own properties. We can take an example of flower. Flower is an object having properties like color, fragrance etc.
2. **Class :** Class is collection of data members (variables, arrays, etc) and member methods.
3. **Encapsulation :** Wrapping up of data members and member methods is known as encapsulation.
4. **Polymorphism :** It means to take out more than one forms. That means single entity can behave differently in different situation.

☛ **Example**

+ operator ; for numerical operands, its gives addition while for string type of operands it gives concatenation.

3 + 5 = 8

a + b = ab

5. **Inheritance :** Creating a new class (child) from existing class (parent) is known as inheritance. In this case properties of parent class are accessible in child class.

Note : The important OOP concepts we are going to learn in next sections in detail.

Syllabus Topic : Conditional Logic

1.2.7 Conditional Logic

- Conditional logic is also called as Decision ma structures.
- Decision making structures need that progra states one or more conditions that are checked b program.
- In Decision making statements, statements w executed if the condition is evaluated to true statements after if block are executed when condition is evaluated to false.
- Most control structures in C# are similar to those or C++, but there are some specific differences.
- C# provides following types of decision ma statements.

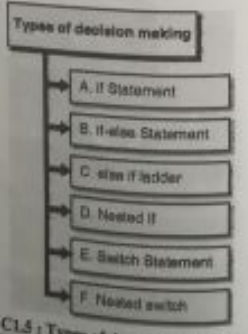


Fig. C1.5 : Types of decision making

1. **if statement** : An if statement consists of a expression which may evaluate to true or false, followed by one or more statements.
2. **if...else statement** : Statement followed by if block are executed when condition is true, if the condition is evaluated to false else part get executed.
3. **else if ladder :** The "else if" ladder is used to test set of conditions in a sequence. It is also considered as multi-way decision making statement.
4. **nested if statements** : One if is written in another if statement.
5. **switch statement** : It is multi-way decision making statement. It provides more alternatives to programmer than if or if-else.
6. **nested switch statements** : One switch statement appear inside another switch statement.

→ 1.2.7(A) if Statement

Q. Explain working of if-else statement with example. **(4 Marks)**

The C# if statement tests the condition. It is executed if condition is true.

☛ **Syntax**

```
if(condition)
{
//code to be executed
}
```

Program 1.2.2

Write a program to check wheather given number is even or not.

Solution :

Program to check even or not number

```
using System;
namespace EvenNo
{
public class IfDemo
  {
    public static void Main(string[] args)
    {
      int num = 10;
      if (num % 2 == 0)
      {
        Console.WriteLine("It is even number");
      }
```

Checks whether number is divisible by 2 or not. If remainder is 0 then num is even

```
    }
}
```

Output

```
It is even number
Press any key to continue . . .
```

→ 1.2.7(B) If-else Statement

In C# if-else statement, it executes the if block if condition is true otherwise executeelse block .

☛ **Syntax**

```
if(condition)
{
//code if condition is true
}
else
{
//code if condition is false
}
```

Program 1.2.3

Write a program to demonstating working of if-else loop.

Solution :

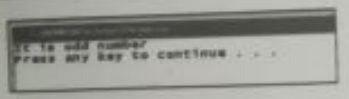
Program that demonstrating working of if-else loop

```
using System;
public class IfExample
  {
    public static void Main(string[] args)
    {
      int num = 11;
      if (num % 2 == 0)
      {
        Console.WriteLine("It is even number");
      }
      else
      {
        Console.WriteLine("It is odd number");
      }
  }
}
```

Checks whether number is divisible by 2 or not. If remainder is 0 then num is even

Checks whether number is divisible by 2 or not. If remainder is not 0 then num is odd

Output



→ 1.2.7(C) else if Ladder

Q. Explain working of else if ladder statement with example. (4 Marks)

- The "else if" ladder is used to test set of conditions in a sequence.
- It is also considered as multi-way decision making statement.
- Number of conditions is given in a sequence with subsequent statements.
- If any of the given condition is satisfied then the related statements are executed and the control exits from the else if ladder. That means further conditions are not going to be checked.
- But if the condition does not satisfy then the compiler goes to next condition to check the condition.

Syntax

```
if(condition1)
{
   //code to be executed if condition1 is true
}
else if(condition2)
{
   //code to be executed if condition2 is true
}
else if(condition3)
{
   //code to be executed if condition3 is true
}
...
else
{
//code to be executed if all the conditions are false
}
```

Program 1.2.4

Write a Program to demonstrate working of else if ladder.

Solution :

Program to demonstrate working of else if ladder

```
using System;
namespace IfElseLadderDemo
{
public class IfExample
  {
    public static void Main(string[] args)
    {
      Console.WriteLine("Enter Marks : ");
      int marks = Convert.ToInt32(Console.ReadLine());

      if (marks <0 || marks >100)
      {
        Console.WriteLine("Invalid marks");
      }
        else if(marks >= 0 && marks < 50)
{
        Console.WriteLine("Fail");
      }
      else if (marks >= 50 && marks < 60)
```

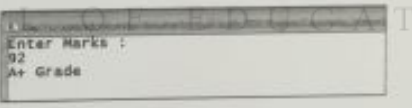
Accepts total marks of student

If num is less than zero or num is greater than 100,num entered by user is wrong

If num greater than or equal to 0 and num less than equal to 50 then grade is fail


```
      {
        Console.WriteLine("D Grade");
      }
      else if (marks >= 60 && marks < 70)
      {
        Console.WriteLine("C Grade");
      }
      else if (marks >= 70 && marks < 80)
      {
        Console.WriteLine("B Grade");
      }
      else if (marks >= 80 && marks < 90)
      {
        Console.WriteLine("A Grade");
      }
      else if (marks >= 90 && marks <= 100)
      {
        Console.WriteLine("A+ Grade");
      }
    Console.ReadLine();
    }
  }
}
```

Output



→ 1.2.7(D) Nested if

- Nested if means we can write one if inside another if statement. While checking number of conditions, we may need to write if statement inside another if statement.

Syntax

```
if(Boolean_expression 1)
{
        /* execute when Boolean expression 1 is true*/
         if(Boolean expression 2)
        {
              /* execute when Boolean expression 1 is true*/
        }
}
```

Program 1.2.5

Write a program for nested if-else.

Solution :

Program for nested if else

```
usingSystem;
namespaceDecisionMaking
{
classProgram
{
staticvoidMain(string[] args)
{
/* local variable definition */
int a =200;
int b =500;
if(a ==200)
{
if(b ==500)
{
Console.WriteLine("Value of a is 200 and b is 500");
}
}
Console.WriteLine("Exact value of a is : {0}", a);
Console.WriteLine("Exact value of b is : {0}", b);
Console.ReadLine();
}
}
}
```

A and b are local variables. Variables defined inside a function are called as local variables

If a equal to 100 is true then check next condition b equal to 500

Output

```
Value of a is 200 and b is 500
Exact value of a is : 200
Exact value of b is : 500
```

→ 1.2.7(E) switch Statement

Q. Explain working of switch statement with example. (4 Marks)

– The switch case is multi way decision making statement which helps to select a single option from multiple given options.

Syntax

```
switch(expression)
{
   case constant_expression 1:
   Statement 1;
   break;

   case constant_expression 2:
   Statement 2;
   break;
   ----------
   case constant_expression n:
   Statement n;
   break;

   default:
   Statement m ;
}
```


Expression

It is usually name of a variable value of which we want to check, or an expression.

Constant expression

– These are the constant values with which the value of expression is compared. The statements in the case are executed whose value matches with the expression.

– If none of the constant expression is matched with the expression, then the statements written in default are executed.

Program 1.2.6

Write a program of switch statement.

Solution :

Program for switch-case statement

```
usingSystem;
namespaceDecisionMaking
{
   classProgram
   {
      staticvoidMain(string[] args)
   {
      /* local variable definition */

      char grade ='B';

      switch(grade)
      {
         case 'A':
         Console.WriteLine("Excellent!");
         break;
```

Passes grade to switch block to match it with different cases

Print "Excellent" if grade matched with A

```
        case 'B':
        case 'C':
        Console.WriteLine("Well done");
        break;
        case 'D':
        Console.WriteLine("Pass");
        break;
        case 'F':
        Console.WriteLine("Better try again");
        break;
        default:
        Console.WriteLine("Invalid grade");
        break;
        }
        Console.WriteLine("Your grade is {0}", grade);
        Console.ReadLine();
    }
}
```

Print " well done" if grade matched with B or C

Print "you passed" if grade matched with D

Print "better try again" if grade matched with F

If grade does not match with any case value then print "Invalid grade"

Output

```
well done
Your grade is B
```

→ 1.2.7(F) Nested switch

Q. Explain working of nested switch statement with example. **(4 Marks)**

- It is possible to have a switch as part of another switch. Even if the case values of the inner and outer switch are same no conflicts will arise.

Syntax

```
switch(expression)
{
    case constant_expression 1:
    Statement 1;
    switch(expression)
    {
        case constant_expression 1:
        Statement 1;
        break;
        case constant_expression 2:
        Statement 2;
        break;
        ----------
        case constant_expression n:
        Statement n;
        break;
        default:
        Statement m ;
    }
    break;
    case constant_expression 2:
    Statement 2;
    break;
    ----------
    case constant_expression n:
    Statement n;
    break;

    default:
    Statement m ;
}
```


Program 1.2.7 : Write a program of nested switch case.

Solution : Program of nested switch case

```
usingSystem;
namespaceSwitchDemo
{
    classNested2switch
    {
        Static void Main(string[] args)
        {
            int a =10;
            int b =20;
            switch(a)
            {
            case 10:
            Console.WriteLine
            ("This is part of outer switch ");
            switch(b)
            {
                Case 20:
                Console.WriteLine("This is part of inner switch ");
                break;
                }
                break;
            }
            Console.WriteLine("Exact value of a is : {0}", a);
            Console.WriteLine("Exact value of b is : {0}", b);
            Console.ReadLine();
        }
    }
}
```

Checks: a is matched with 10,if yes then checks b is matched with 20. If both conditions are true then executes statements in those cases.

Output

```
This is part of outer switch
This is part of inner switch
Exact value of a is : 10
Exact value of b is : 20
```

Syllabus Topic : Loops

1.2.8 Loops

Q. What is use of loop and list various looping statment? (2 Ma

- Loops are used to execute specific task repeatedly in our program.
- Rather than writing the code again and again we can use the concept of loop. There are various situations when we may want to execute specific task multiple times.
- For example student mark sheet. Here we want to accept details from student and want to generate the marks sheet. This task is obviously repeated for number of students. In such situations we use the loops.
- Following are the looping statement in c#

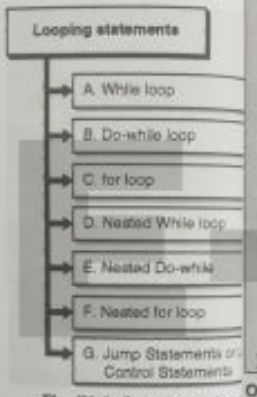


Fig. C1.6 : Looping statem

➜ 1.2.8(A) while Loop

Q. Explain working of while loop with example. **(4 Marks)**

- While loop is used when we want to execute the set of statements repeatedly until the given condition is satisfying.
- While loop is considered as an entry controlled loop. That means the condition is given at the beginning of loop, if the given condition does not satisfy, the loop statements never get executed.
- If the condition is satisfied, then loop statements are repeatedly executed until the condition is satisfying. On condition becomes false, the control exits the loop.

Syntax

```
while(Condition 1)
{
   Statement 1;
}
```

Program 1.2.8

Write a program to check given number is Armstrong or not.

Solution :

Program to checking givne number is Armstrong or not

```
using System;
namespace ArmstrongDemo
{
  class ArmstrongProgram
  {
    static void Main(string[] args)
```



```
    {
      int num , sum = 0, rem = 0, temp;
      Console.WriteLine("Enter Number");
      num = Convert.ToInt32(Console.ReadLine());
      temp = num;
      while (temp != 0)
{
        rem  = temp % 10; //find the reminder
        sum  = sum + (rem * rem * rem);
        temp = temp / 10;
      }
    if (num == sum)
{
  Console.WriteLine("Entered  Number is an Armstrong Number");
  }
      else
      {
        Console.WriteLine("Entered  Number is  not an
 Armstrong Number");
      }
    }
}
```

Until the temp becomes 0 it will repeat the loop statements To get digits from the temp we calculate the remainder by dividing the temp with 10

If given number is same as addition of cubes of digits of the number then it is Armstrong number

Output

```
Enter the Number371
Entered Number is an Armstrong Number_
```

Program 1.2.9

Write a program to accept a number from user and print factorial of it.

e.g. Factorial of 5 is calculated as
= 5*4*3*2*1 = 120

Solution : Program to accept a number from uses and print factorial of it

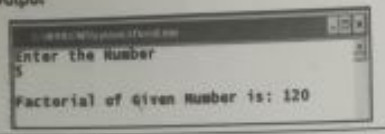
```
usingSystem;
namespace factorial
{
  class Program
  {
    staticvoid Main(string[] args)
    {
      int i, num, fact=1;
```

Declares num and fact ,i and initializes fact to 1

```
        Console.WriteLine("Enter the Number");
        num=int.Parse(Console.ReadLine());
        while( num>0)
        {
            fact = fact *num;
            num--;
        }
        Console.WriteLine("\nFactorial of Given Number
is: "+fact);
        Console.ReadLine();
    }
  }
}
```

Accepts n from user

Output



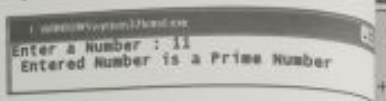
Program 1.2.10

Write a program to find whether the entered number is prime or not. (Prime number is the one which is divisible by 1 and itself only)

Solution : Program to finding entered number is prime or not

```
using System;
namespace example
{
  class prime
  {
    public static void Main()
    {
      Console.Write("Enter a Number : ");
      int num;
      num = Convert.ToInt32(Console.ReadLine());
      int k, i = 1;
      k = 0;
      while (i <= num / 2)
      {
        if (num % i == 0)
        {
          k++;
        }
        i++;
      }
      if (k == 2)
      {
        Console.WriteLine("Entered Number is not a
Prime Number");
      }
      else
      {
        Console.WriteLine(" Entered Number is a Prime
Number");
      }
      Console.ReadLine();
        }
  }
}
```

Output



➜ 1.2.8(B) do-while Loop

Q. Explain working of do-while loop with example. [4 Marks]

- The C# do-while loop is used to iterate a part of program several times.
- If the number of iteration is not fixed then use do-while loop.
- Remember that the conditional expression appears at the end of the loop, so the statement(s) in the loop will execute once before the condition is tested. Hence it is called as exit controlled loop.
- If the condition is true, the flow of control jumps back to do, and the statement(s) in the loop execute again. This process repeats until the given condition becomes false.

Syntax

```
do
{
  Statement 1;
} while(condition 1);
```

Program 1.2.11

Write a program in C# to accept a number from user and print it's cube. Ask user for continuity, if user enters yes, repeat the process.

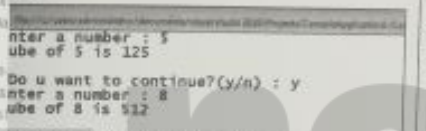
Solution : Program to print cube of given number

```
using System;
namespace RepeatDemo
{
public class DoWhileExample
  {
      public static void Main(string[] args)
  {
      int num,cube;
      char ch;

      do
      {
        Console.Write("Enter a number : ");
        num = int.Parse(Console.ReadLine());
```



```
        cube=num*num*num;
        Console.WriteLine("Cube of "+num + " is
+cube);
        Console.Write("\n Do u want to
ontinue?(y/n) : ");
        ch=char.Parse(Console.ReadLine());
        }while(ch=='y');
    }
  }
```

Output



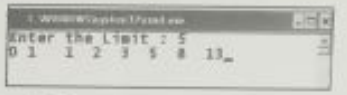
ogram 1.2.12

rite a program to print Fibonacci series.

lution : Program to print Fibonacci series

```
ingSystem;
amespace fibonaci

class Program
{
    staticvoid Main(string[] args)
    {
        int i=0, count, f1 =0, f2 =1, f3 =0;
        Console.Write("Enter the Limit : ");
        Count =int.Parse(Console.ReadLine());
        Console.Write(f1 + " ");
        Console.Write(f2+" ");
        do
        {
            f3 = f1 + f2;
            Console.Write(" "+f3);
            f1 = f2;
            f2 = f3;
            i++;
        } while(i <= count);
        Console.ReadLine();
    }
  }
}
```

Output



Program 1.2.13

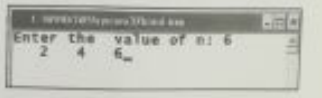
Write a program to display all even numbers from 1 to N.

Solution :

Program to display all even numbers from 1 to N.

```
usingSystem;
namespace EvenNumber
{
  class EvenProgram
  {
      staticvoid Main(string[] args)
      {
          int i=1,n;
          Console.Write("Enter the value of n: ");
          n=int.Parse(Console.ReadLine());
          do
          {
              if(i%2==0)
              Console.Write(" "+i);
              i++;
          }while(i<=n);
      }
  }
}
```

Output



➜ 1.2.8(C) for Loop

Q. Explain working of for loop with example. (4 Marks)

- The C# for loop is used to iterate a part of the program number of times. If the number of iterations is fixed, it is recommended to use for loop instead of while or do-while loop.
- The C# for loop is same as C or C++. We can initialize variable, check condition and increment or decrement value.

Syntax

```
for(initialization; condition; increment/decrement)
{
    Code to be executed
}
```

Program 1.2.14

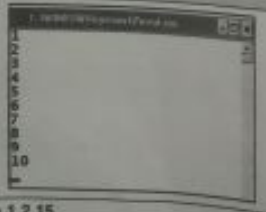
Write a program to print 1 to 10 numbers using for loop.

Solution :

Program to print 1 to 10 numbers using for loop.

```
using System;
namespace Loops
{
    class Program
    {
        static void Main(string[] args)
        {
            for(int a = 1; a <= 10; a++)
            {
                Console.WriteLine(a);
            }
            Console.ReadLine();
        }
    }
}
```

Output



Program 1.2.15

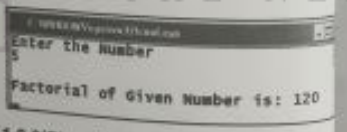
Write a program to accept a number from user and print factorial of it.

Solution :

Program to accept a number from user a factorial of it

```
usingSystem;
namespace factorial
{
    class Program
    {
        staticvoid Main(string[] args)
        {
            int i, number, fact;
            Console.WriteLine("Enter the Numbe
            number =int.Parse(Console.ReadLi
            fact = number;
            for(i = number -1; i >=1; i--)
            {
                fact = fact * i;
            }
            Console.WriteLine("nFactorial of Gi
Number is: "+fact);
            Console.ReadLine();
        }
    }
}
```

Output



→ 1.2.8(D) Nested while Loop

Q. Explain working of nested while loop with

- We can write one while loop inside another
Nested while loop in c# is same as nested
C++.

Syntax

```
while (condition/expression)
{
    while (condition/expression)
    {
        Statement
    }
}
```

Body of inner for loop

Statement ← Body of outer while loop

rogram 1.2.16

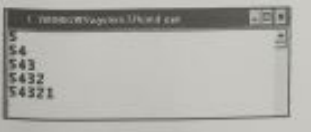
/rite a C# program to print triangle pattern of first five atural numbers using nested while loop.

olution :

rogram to print triangle pattern of first five atural numbers using nested while loop.

```
using System;
namespace loop
{
    class Program
    {
        static void Main(string[] args)
        {
            int i = 5;
            while (i >= 1)
            {
                int j = 5;
                while (j >= i)
                {
                    Console.Write(j);
                    j--;
                }
                i--;
                Console.WriteLine();
            }
            Console.Read();
        }
    }
}
```

Output



→ 1.2.8(E) Nested do-while

Q. Explain working of nested do-while loop with example. (4 Marks)

- We can write one do- while loop inside another while loop. Nested do while loop in c# is same as nested do while in C or C++.

Syntax

```
do
{
    do
    {
        statements;
    }while( condition/expression);
    statements;
}while( condition );
```

Body of inner do-while loop

Body of outer do-while loop

Program 1.2.17

Write a C# program to print triangle pattern of character '*' using nested do while loop.

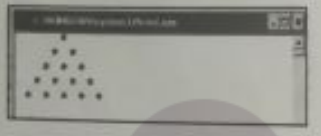
Solution :

Program to print triangle pattern of character '*' using nested do while loop

```
using System;
namespace loop
{
    class Program
    {
        static void Main(string[] args)
        {
            int i = 5;
            do
            {
                int space = i;
                do
                {
                    Console.Write(" ");
                    space--;
                } while (space >= 1);
                int j = 5;
                do
```

```
        {
            Console.Write("* ");
        }
        while(j >= i);
        Console.WriteLine();
    }
    while(i >= 1);
    Console.ReadLine();
  }
}
```

Output



→ 1.2.8(F) Nested for Loop

Q. Explain working of Nested for loop with example. **(4 Marks)**

- We can write one for loop inside another for loop. Nested for loop in c# is same as nested while in C or C++.

Syntax

```
for ( init; condition; increment )
{
    for ( init; condition; increment )
    {
        statement(s);
    }
    statement(s);
}
```

Program 1.2.18

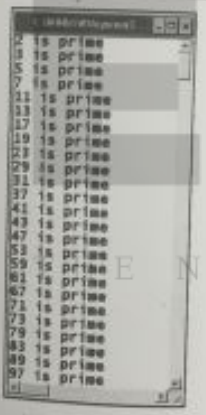
Write a program to a nested for loop to find the prime numbers from 2 to 100.

Solution :

Program to find prime numbers from 2 to 100 using nested for loop

```
using System;
namespace Loops
{
  class Program
  {
    static void Main(string[] args)
    {
      /* local variable definition */
      int p, q;
      for(p = 2; p <100; p++)
      {
        for(q =2; q <=(p / q); p++)
        if((p% q)==0)break;// if fa
found, not prime
        if(q >(p / q))
        Console.WriteLine("{0} is pr
      }
      Console.ReadLine();
    }
  }
}
```

Output



→ 1.2.8(G) Jump Statements or Control Statements

Q. Explain working of loop control statements with example. (4

- Jump statements or Loop control statements execution from its normal sequence i.e Jump st is used to shift the control from one location to the program without checking any condition.
- When execution ends, all automatic objects created in that program are destroyed.
- C# provides the following loop control statement

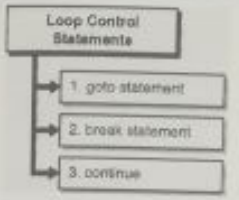


Fig. C1.7 : Loop Control Statements

→ 1) goto statement

- The C# goto statement is also called as jump statement.
- It is used to move control from one part of program to the other part of the program.
- It transfer control without checking any condition.
- The goto needs a label to recognize the place where to transfer the execution control.
- A label is any valid identifier in C# and must be followed by a colon.
- The label is placed immediately before the statement where the control is to be transferred. The goto statement can move the execution control to any block of code in a program.

Syntax

```
        Statement 1;
        goto label 1
        ....
        Statement 4;
label 1 : Statement 5;
        Statement 6;
```

Program 1.2.19

Write a program to demonstrate use of goto statement.

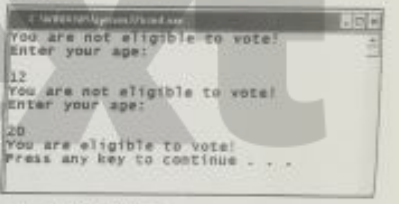
Solution :

Program to demonstrate use of goto statement

```
using System;
namespace JumpStatements
{
public class GotoExample1
  {
    public static void Main(string[] args)
    {
      noteligible:
      Console.WriteLine("You are not eligible to vote!");
      Console.WriteLine("Enter your age:\n");
      int age = Convert.ToInt32(Console.ReadLine());
      if (age < 18)
      {
        goto noteligible;
      }
      else
      {
        Console.WriteLine("You are eligible to vote!");
      }
    }
  }
}
```

if condition is true then control is transfer to label noteligible ,otherwise execute print the message "you are eligible for vote"

Output



→ 2) break statement

- The **break** statement in C# has following two uses:
  1) As soon as the **break** statement is used inside a loop, the loop is instantly terminated and program control transfer to the next statement after the loop.
  2) It can be used to end the execution of a case in the **switch** statement.
- It breaks the current flow of the program at the given condition.
- In the nested loop, the break statement will be used to stop the execution of the innermost loop and begin execution of code after that block.

Syntax

```
break;
```

Program 1.2.20

Write a program to demonstrate use of break statement.

Solution :

Program to demonstrate use of break statement

```
using System;
namespace JumpStatements
{
public class BreakExample
  {
  public static void Main(string[ ] args)
  {
    for (int i = 1; i <= 10; i++)
    {
      if (i == 5)
      {
        break;
      }
      Console.WriteLine(i);
    }
  }
}
```

If i is equal to 5 then Exit from loop

Output

```
1
2
3
4
Press any key to continue . . .
```

- Use C# Break Statement in nested Loop
- Break statement in c# is used to stop execution of loop.

Program 1.2.21

Write a program to demonstrate use of C# Break statement in nested Loop.

Solution :

Program to demonstrate use C# Break statement in nested Loop

```
using System;
namespace JumpStatements
{
public class BreakExample
{
  public static void Main(string[] args)
  {
    for(int i=1;i<=3;i++)
    {
      for(int j=1;j<=3;j++)
      {
        if(i==2&&j==3)
        {
          break;
        }
        Console.WriteLine(i+" "+j);
      }
    }
  }
}
```

If i equal to 2 and j equal to 2 then Exit from inner for loop and execution control transfer to outer for loop, increment the value of i

Output

```
1 1
1 2
1 3
2 1
2 2
3 1
3 2
3 3
Press any key to continue . . .
```

3) C# Continue Statement

- The continue statement set the control at the beginning of loop for particular condition.
- That means for the given condition, the statements below the continue statement are skipped and then execute same block of code from beginning.
- We can use continue statement with any looping statement.

Syntax

```
jump-statement;
continue;
```

Program 1.2.22

Write a program to demonstrate continue statement in C#.

Solution :

Program to demonstrate continue statement in C#.

```
using System;
namespace JumpStatements
{
public class ContinueExample
  {
    public static void Main(string[] args)
    {
    for(int i=1;i<=10;i++)
{
      if(i==5)
{
        continue;
      }
      Console.WriteLine(i);
    }
  }
}
```

When value of i is equal to 5 then console.writeLine() statement is skipped and execution control transfer to beginning of for loop

Output

```
1
2
3
4
6
7
8
9
10
Press any key to continue . . .
```

Syllabus Topic : Methods

1.2.9 Methods

Q. What is function? How to declare, define and call the function? (4 Marks)

- Method is a block of statements used to execute a task repeatedly.
- Every C# program has at least one method named main().
- Defining method in C#

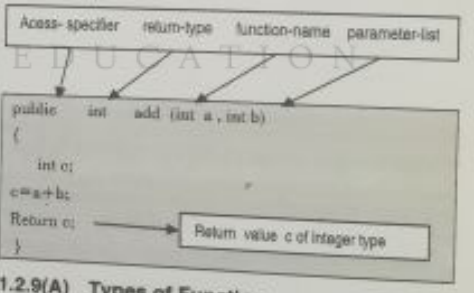
Syntax

```
<access-specifier> <return-type> <FunctionName>(<parameter list>)
{
  // method body
}
```

- Following are the various elements of a method :

1. **Access specifier** : Decides the scope of method.
2. **Return type :** A method may or may not return a value to calling function. Called function can send single value at a time to calling function. The return type is the data type of that value returned by the method. Void is used as return type, if the method is not returning any value.
3. **Method name:** Method name is a unique identifier which is case sensitive. It follows all variable naming rules i.e. start with alphabet, underscore, not start with number, method name should not keyword etc.
4. **Parameter list :** The parameter is used to give input to method and receive data from a method. Method may contain parameters or not. We can pass more than one parameters to function. The parameter list contains the data type and names of the parameters of the method.
5. **Method body :** Method body contains the group of statements needed to execute for performing a particular task.

Example



1.2.9(A) Types of Function

- There are two types of functions in C# code :

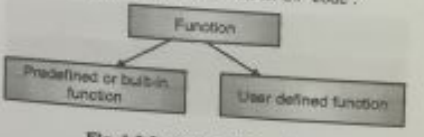


Fig. 1.2.3 : Types of functions

Predefined or built in functions

These are the functions which are already defined by C# such as Console. ReadLine() for reading, Console.WriteLine() for writing etc.

- C# has rich set of built-in functions.

Program 1.2.23 : Write a program to demonstrate built-in function.

Solution :

Program to demonstrate built-in function

```
using System;                                   // Use namespace in your program which support predefined functions
namespace Hello World Demo
{
  Class First Program
  {
    static void main(string[] args)
    {
      Console.WriteLine("Hello World");         // Use built-in function Console.WriteLine() to display the output on screen
    }
  }
}
```

Output

```
Hello World
Press any key to continue . . .
```

2. User-defined functions

These are the functions which are created by the C # programmer to perform specific task. Programmer can use these functions many times.

Syntax

```
return_type function_name(data_type parameter...)
{
  //code to be executed
}
```

Program 1.2.24

Write a program to demonstrate user defined function.

Solution :

Program to demonstrate user defined function

```
using System;
namespace CalculatorDemo
{
class FactorialExample
{
  public int factorial(int n)                   // User defined function factorial () passing number whose factorial we want to calculate
  {
    int result;
    if(n==1)
    {
      return 1;
    }
    else
    {
      result = factorial(n-1)* n;
      return result;
    }
  }
  static void Main(string[] args)
  {
    FactorialExample fact=new FactorialExample();
    int fact=f. factorial(5);                   // Calling factorial() method
    Console.WriteLine("Factorial of number is : {0}", fact);
  }
}
```

Output

```
Factorial of number is : 120
Press any key to continue . . .
```

1.2.9(B) Type of Function Call

Q. Write note on types of function call. **(4 Marks)**

- Function can be called in different ways :

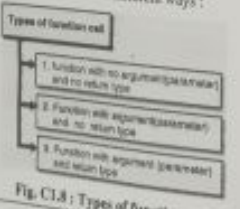


Fig. C1.8 : Types of function call

1) Function with no argument(parameter) and no return type

- In this type of function call,there is no information transfer between calling function and called function.
- When function has no argument, it does not receive any information from calling function.
- Similarly, when it does not return any value ,calling function does not receive any information from called function.

Program 1.2.25

Write a program of calling function with no argument and no return value.

Solution :

Program of calling function with no argument and no return value

```
using System;
namespace FunctionDemo
{
  class FunctionCall                            // User defined function Display() without return type and does not have any parameter.
  {
    public void Display()
    {
      Console.WriteLine("This is non parameterized function");
    }                                           // There is no return statment
    static void Main(string[] args)             // Create object of class
    {
      FunctionCall fun= new FunctionCall ();
      fun.Display();                            // Function call without passing any parameter
    }
  }
}
```

Output

```
This is non parameterized function
Press any key to continue . . .
```

2) Function with argument(parameter) and no return type

- In this type of function call, there is information transfer between calling function and called function.

Program 1.2.26

Write a program of calling function with argument and no return value.

Solution : Program demonstrating function call

```
using System;
namespace FunctionDemo
{
  class FunctionCall
  {
    public void add(int n1, int n2)
    {
      int result;
      result = n1 + n2;
      Console.WriteLine("Result of addition is : " + result);
    }
    static void Main(string[] args)
    {
      FunctionCall p = new FunctionCall();
      p.add(3, 6);
      Console.ReadLine();
    }
  }
}
```

Output

```
Result of addition is : 9
```

3) Function with argument(parameter) and return type

The call function is executed until return statement is encountered. The return value is pass back to function call.

Program 1.2.27

Write a program to demonstrate working of return key word.

Solution :

Program to demonstrate working of return keyword

```
using System;
namespace returnstatementDemo
{
  class Program
  {
```

Return addition of n1 and n2

Returned value is assigned to result

```
Console.WriteLine("result of addition is :"+result);
Console.ReadLine();
```

Output

```
Result of addition is : 8
```

1.2.9(C) Methods of Passing Parameter

Q. Which methods are used to pass parameter to method? **(4 Marks)**

There are three ways to pass parameter to method

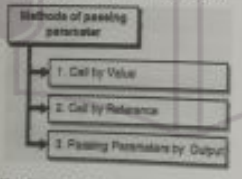


Fig. C1.9 : Methods of passing parameter

1) Call by Value

- In call by value, values of parameters are passed to the function at the time of function call.
- Formal parameters are replaced by actual parameter at the time of function call.
- In this method, we pass values of parameters so the changes in formal parameters do not reflect in actual parameters.
- In previous section, the values are passed by value for the parameters.

Program 1.2.28

Write a program to demonstrate call by value.

Solution : Program to demonstate call by value

```
using System;
namespace FunctionDemo
{
  class CallByValueDemo
  {
    public void Display(int a)
    {
      a= a*a;
      Console.WriteLine("Value inside the display function " +a);
    }
    static void Main(string[] args)
    {
      int a = 50;
      CallByValueDemo c1 = new CallByValueDemo();
      Console.WriteLine("Value before calling the function " +a);
      c1.Display(a);
      Console.WriteLine("Value after calling the function " + a);
    }
  }
}
```

Calling function by passing value

Output

```
Value before calling the function 50
Value inside the display function 2500
Value after calling the function 50
Press any key to continue . . .
```

2) Call by Reference

- C# provides a ref keyword to pass address of parameter to method.
- It passes reference(address) of argument to the function rather than copy of original value.

- The changes in formal parameters also reflect in actual parameters.

Program 1.2.29

Write a program to demonstrate call by reference.

Solution :

Program to demonstrate call by reference

```
usingSystem;
namespace FunctionDemo
{
class SwapDemo
{
void swap(ref int a ,ref int b)
{
    int temp;
    temp = a;
    a = b;
    b = temp;
}
static void Main(string[] args)
{
SwapDemo o =new SwapDemo();
int p=100;
int q=200;
Console.WriteLine("Before swap, value of p: {0}", p);
Console.WriteLine("Before swap, value of q : {0}", q);
o.swap(ref p, ref q);
  Console.WriteLine("After swap, value of p : {0}", p);
  Console.WriteLine("After swap, value of q : {0}", q);
Console.ReadLine();
```

save the value of a

Put value in b in a

Put value of b in temp

Local variable declaration

Value before swapping

calling a function to swap the values

Output

```
Before swap, value of  p: 100
Before swap, value of  q : 200
After swap, value of p : 200
After swap, value of q : 100
```

3) Passing Parameters by Output

- A return statement can be used to return only one value from a called function to calling function.
- But using output parameters, you can return multiple values from a called function to calling function.
- Output parameters are similar to call by reference (use ref keyword),but difference is that they transfer data out of the method rather not accept in method.

Program 1.2.30

Write a program to demonstrate passing parameter by output.

Solution :

Program to demonstrate passing parameter by output

```
using System;
namespace FunctionDemo
{
  class OutParameterDemo
  {
      public void Display(out int v)
      {
        int square = 5;
        v = square;
        v*=v;
      }
    static void Main(string[] args)
    {
      int v = 50;
OutParameterDemo o= new OutParameterDemo ();
 Console.WriteLine("Value before passing out variable " + v);
        o.Showvout v);
        Console.WriteLine("Value after recieving the out variable " + v);
    }
  }
}
```

Declare out parameter

v = v*v

Creating object of class

Passing out parameter

```
Value before passing out variable 10
value after recieving the out variable 25
Press any key to continue . . .
```

1.2.9(D) Scope of variables

Q. Write note on scope of variable. (4 Marks)

- Scope is the area of program in which the variable is accessible.
- There are two areas where variables can be declared in C# programming language

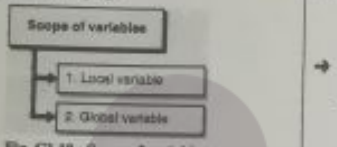


Fig. C1.10 : Scope of variables

→ 1) Local Variables

- Variables that are defined inside a method are called local variables.
- They can be used only inside that function or block of code in which they are defined.
- There is no scope of local variable outside that method.
- Local variables are deleted once the working of function that created them is completed.
- They are recreated every time when that function is called.

Program 1.2.31

Write a program to demonstrate local variable scope.

Solution : Program to demonstrate local variable scope

```
using System;
namespace FunctionDemo
{
  class LocalDemo
  {
    void add(int a,int b)
    {
      int c;
    }
  }
}
```

Local variable declaration

```
    c=a+b;
    Console.WriteLine("addition is:"+c);
  }
  static void Main(String[] args)
  {
    LocalDemo local=new LocalDemo();
    local.add(10,15);
  }
}
```

Local variable can be accessed only inside function

Output

```
addition is:25
Press any key to continue . . .
```

→ 2) Global Variables

- Global variables are defined outside the methods, at the top of the program.
- Global variables maintain their values throughout execution of program.
- Global variable is available throughout the program anywhere. They can be accessed inside any function defined in the program.

Program 1.2.32

Write a program to demonstrate global variable scope.

Solution :

Program to demonstrate global variable scope

```
using System;
namespace FunctionDemo
{
  class LocalDemo
  {
    int c;
  }
  void add(int a,int b)
  {
      c=a+b;
  }
  static void main(String[] args)
  {
    LocalDemo local=new LocalDemo();
```

Global variable declaration


```
    local.add(10,15);
    Console.ReadLine("Addition is:"+ local.c);
  }
}
```

Access value of global variable c

Output

```
Addition is:25
Press any key to continue . . .
```

1.2.9(E) Recursion

Q. What is recursion ? Explain with example. (4 Marks)

- Method can call itself in its definition is called as **recursion**.
- To do same task repetitively, recursion is used.

Program 1.2.33

Write a program to find factorial of number using recursion.

Solution :

Program to find factorial of number using recursion

```
using System;
namespace FunctionDemo
{
  class FactorialDemo
  {
    public int factorial(int num)
    {
      int result;
      if(num == 1)
      {
        return 1;
      }
      else
      {
        result = factorial(num -1)* num;
        return result;
      }
    }
    static void Main(string[] args)
    {
      FactorialDemo f=new FactorialDemo();
      Console.WriteLine("Factorial of 6 is : {0}",
n.factorial(6));
    }
  }
}
```

Local variable declaration

Recursive call to function factorial()

Output

```
Factorial of given no. is720
Press any key to continue . . .
```

1.2.10 C# Object

Q. Write note on object in C#. (4 Marks)

- In C#, Object is a real world thing such as car, person, chair etc.
- That means, object is an entity that has state and behavior. Here, state means data and behavior means functionality.
- Object is a runtime entity that means it is created at runtime.
- Object is an instance of a class.
- All the members of the class can be accessed through object.
- To access the members of class , dot (.) operator is used.
- The dot operator links the name of an object with the name of a member.

Syntax

```
<Class-name> <Object-name> = new <Class-name>
e.g
Student s1 = new Student();
```

creating an object of Student

Syllabus Topic : Classes

1.2.10(A) Classes

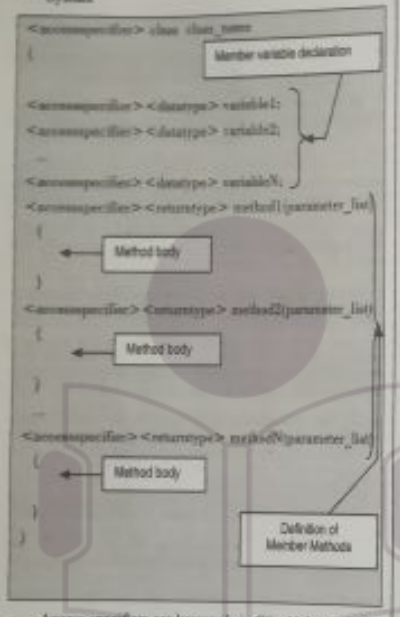
Q. What is mean by class? How to define class? (4 Marks)

- Class is a collections of data members and methods.
- We can create one or more objects of same class to access class members.
- The methods and variables that are included in class are called as members of that class.

Defining Class

Definitions of class starts with the keyword class and followed by the class name and the body of class enclosed by a pair of curly braces. Following is the syntax of a defining class :

Syntax



- Access specifiers are keywords in C# which are used to restrict availability of object, method, class and its members into the program or in application
- Data type state the type of variable, and return type specifies the data type of the data the method returns, if any.

Access Specifier

This determines the accessibility (access permission) of method from another class. There are 5 access specifiers : public ,private, protected, internal, protected internal.

Now we see how to use this access specifiers :

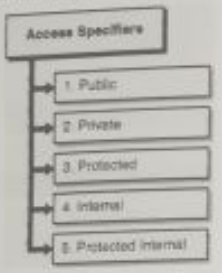


Fig. C1.11 : Access Specifiers

1. Public

- Public is the mostly used access specifier in C# language. Public members can be accessed from anywhere, that means there is no limitation on accessibility of public members.
- We can access public members inside that class in which they define as well as outside of that class.
- Public members can be accessed by any other code in the same assembly or another assembly that references it.
- Where to use public access specifier :
  1. within the class in which they are declared.
  2. within the derived classes of that class available within the same assembly.
  3. Outside the class within the same assembly.
  4. within the derived classes of that class available outside the assembly.
  5. Outside the class, outside the assembly.

2. Private

- Accessibility of private members is limited only inside the classes or struct in which they are declared. The private members cannot be accessed outside the class .
- Where to use Private access specifier :
  1. Only Within the class in which they are declared.

3. Protected

- Accessibility of protected member is limited within the class or struct and the derived class of that class.

- Where to use Protected access specifier :
  1. Within the class in which they are declared.
  2. Within the derived classes of that class available within the same assembly.
  3. Within the derived classes of that class available outside the assembly.

4. Internal

- Members which are defined with internal access modifiers can be accessed within the same program that contain its declarations and also accessed within the same assembly level but not from another assembly.
- Where to use Internal access specifier :
  1. Within the class in which they are declared.
  2. Within the derived classes of that class available within the same assembly.
  3. Outside the class within the same assembly.

5. Protected Internal

- Protected internal members has access permissions same as access specifier both protected and internal. It can access anywhere in the same assembly and in the same class and also in classes derived from that class.
- Where to use Protected Internal access specifier :
  1. Within the class in which they are declared.
  2. Within the derived classes of that class available within the same assembly.
  3. Outside the class within the same assembly.
  4. Within the derived classes of that class available outside the assembly.

Program 1.2.34

Write a program to demonstrate how to define the class and how to access members of class.

Solution :

Program to demonstrate how to define the class and how to access members of class

```
using System;
public class Student
{
    int rollno;
    String name;
    public static void Main(string[] args)
```

data members

```
    {
        Creating an object of Student
        Student s1 = new Student();
        s1.rollno = 101;
        s1.name = "Kunal";
        Console.WriteLine(s1.rollno);
        Console.WriteLine(s1.name);
        Access data members of class using object name and dot(.) operator
    }
}
```

Output

```
101
Kunal
Press any key to continue . . .
```

Program 1.2.35

Write a program to demonstrate the class where we are having main() method inanother class.

Solution :

Program to demonstrate the class where we are having main () method in another class

```
using System;
public class Student
{
    public int id;
    public String name;
}
class TestStudent
{
    To access the member of student class in another class, use access specifier public
    public static void Main(string[] args)
    {
        Student s1 = new Student();
        s1.id = 101;
        s1.name = "Kunal";
        Console.WriteLine(s1.id);
        Console.WriteLine(s1.name);
    }
}
```

Create object of Student class and access members

Output

```
101
Kunal
Press any key to continue . . .
```

Syllabus Topic : Value Types and Reference Types

1.2.11 Value Type and Reference Type

It is explained in previous section 1.2.9(C) Methods of passing parameter.

Syllabus Topic : Namespaces and Assemblies

1.2.12 Namespaces and Assemblies

1.2.12(A) C# Namespaces

Q. Write note on namespaces. (4 Marks)

- Namespaces in C# are used to store multiple classes into single unit so that we can manage and reuse them whenever needed.
- In C# program, we use System.Console where System is the namespace and Console is the class.
- To access the classes from namespace, we need to use namespacename.classname.
- We can use **using** keyword to use specific namespace in our program.

☞ **Defining Namespace**

We can define namespace with the keyword **namespace** followed by the namespace name.

☞ **Syntax**

```
namespace namespace_name
{

}
```

Program 1.2.36

Write a program to demonstrate use of namespace.

Solution :

Program to demonstrate use of namespace

```
using First;
using Second;
namespace First
{
  public class NamespaceDemo1
  {
      public void sayHi()
      {
          Console.WriteLine("Hi  First namespace");
      }
  }
namespace Second
{
  public class NamespaceDemo2
  {
      public void sayWelcome()
      {
          Console.WriteLine("Welcome
Second Namespace");
      }
  }
public class TestNamespace
{
  public static void Main()
  {
      NamespaceDemo1
h1 = new NamespaceDemo1();

  NamespaceDemo2w1 = new NamespaceDemo2();
      h1.sayHi();
      w1.sayWelcome();
  }
}
```

Output

```
Hi  First namespace
Welcome  Second Namespace
Press any key to continue . . .
```

1.2.12(B) Nested Namespaces

- You can define one namespace inside another namespace in such a way that we define one if inside another if.

☞ **Syntax**

```
namespace namespace_name1
{
    Code declaration
```



```
namespace namespace_name2
{
    Code declaration
}
```

- You can access members of nested namespace by using dot (.) operator.

Program 1.2.37

Write a program to show how to access members of nested namespace.

Solution :

Program to show how to access members of nested namespace

```
using System;
using FirstSpace;
using FirstSpace.SecondSpace;
namespace FirstSpace
{
  class NameSpaceExample1
  {
    public void Display()
    {
      Console.WriteLine("Inside first_space");
    }
  }
  namespace SecondSpace
  {
class NameSpaceExample2
    {
      public void Display()
      {
        Console.WriteLine("Inside second_space");
      }
    }
  }
class TestDemo
{
  static void Main(string[] args)
  {
    NameSpaceExample1 fe = new NameSpaceExample1();
      NameSpaceExample2 sc =new
NameSpaceExample2();
      fe.Display();
      sc.Display();
    }
}
```

Output

```
Inside first_space
Inside second_space
Press any key to continue . . .
```

1.2.12(C) Assemblies

Q. Write note on assemblies. (4 Marks)

- When a .Net application is successfully compiled, assembly file is automatically generated.
- An assembly may be an executable file(.exe) or a Dynamic Link Library (DLL). At only first compilation this file is generated, and after that for each subsequent compilation it gets updated.
- The assembly generation is background process of an application.
- An Assembly contains IL (Intermediate Language) code. This IL is same as of the Java byte code.
- In the .Net language, assembly contains metadata. Features of every "type" is provided in the metadata.
- Assembly contains a special file called as Manifest. This manifest file contains the information regarding the current version of the assembly as well as other related information.
- .Net provides two types of assemblies, Single file and Multi file. A single file assembly holds all the necessary information like IL, Metadata, and Manifest in a single package. Most of the assemblies in .Net are of the type single file assemblies.
- Multi file assemblies are made up of various types of .Net binaries or modules and are usually created for larger applications.
- One of the assemblies contains the most important special manifest file and others assemblies may contain IL and Metadata instructions.

Syllabus Topic : Inheritance

1.2.13 Inheritance

Q. What is mean by inheritance? Write note on types of inheritance. (4 Marks)

- Inheritance is one of the characteristic of Object-oriented programming language.
- The mechanism of creating new class from existing class is called as inheritance.
- In inheritance, old class from which we derive new class is called as parent class, base class or super class.
- New class is called as child class, derived class or sub class.
- The derived class has its own attributes (variables) and behavior (functions) and it also access the variables and functions defined in its parent class.
- The thought behind inheritance is to implements the IS-A relationship.
- For example, dog IS-A animal. Here animal is parent class and dog is child class.

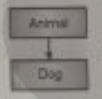


Fig. 1.2.4 : Inheritance Example

Advantages of Inheritance

1. It reduces code repetition.
2. It provides code reuse feature.
3. It reduces size of source code and improves code readability.
4. We can easily divide and manage large source code in parent and child class.
5. It helps to extend the code by overriding the base class functions in its child classes.

Types of Inheritance

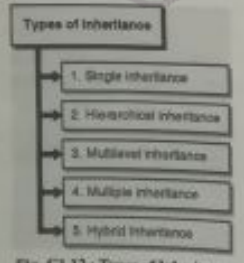


Fig. C1.12 : Types of Inheritance

1) Single inheritance

When one child class is derived from one parent class it is called as single inheritance.

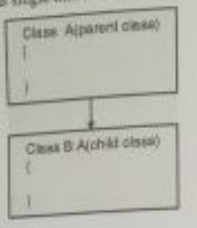


Fig. 1.2.4(a) : Example of single inheritance

Program 1.2.38

Write a program to demonstrate single level inheritance.

Solution :

Program to demonstrate single level inheritance

```
using System;
public class Animal                          // Parent class
{
  public void like()
{
  Console.WriteLine("Like milk...");
}
}
public class Cat: Animal                     // Create child class Cat from parent class Animal
{
        public void type()
{
Console.WriteLine("Domestic Animal");
  }
}
class InheritanceDemo2
{
    public static void Main(string[] args)
    {
      Cat c1 = new Cat();
      c1.like();                             // Call to function of parent class like() using object of child class and also to its own function
      c1.type();
    }
}
```


Output

```
Like milk...
Domestic Animal
Press any key to continue . . .
```

2) Hierarchical inheritance

- This is the type of inheritance in which there are multiple child classes created from one parent class. This type of inheritance is used when there is a requirement accessing one class features in multiple classes.
- In this type of inheritance, there is no single class which get properties of all the classes, hence we have to create objects for all the child classes.

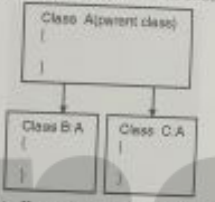


Fig. 1.2.4(b) : Example of Hierarchical inheritance

Program 1.2.39

Write a program to demonstrate Hierarchical inheritance.

Solution :

Program to demonstrate Hierarchical inheritance.

```
using System;
using System.Text;

namespace ConsoleApplication32
{

  class A
  {
    public void msg()
    {
      Console.WriteLine("This is A class Method");
    }
  }
  class B : A
  {
    public void info()
    {
      Console.WriteLine("This is B class Method");
    }
  }
  class C : A
  {
    public void getinfo()
    {
      Console.WriteLine("This is c class Method");
    }
  }

  class D
  {
    public static void Main(String[] args)
    {
      B b1 = new B();
      C c1 = new C();
      b1.msg();      // can be called by both b1 and c1
      b1.info();
      c1.getinfo();
    }
  }
}
```

Output

```
This is A class Method
This is B class Method
This is c class Method
Press any key to continue . . .
```

3) Multilevel inheritance

- In this type of inheritance a class is inherited from another child class.

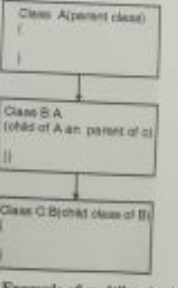


Fig. 1.2.4(c) : Example of multilevel inheritance

Program 1.2.40

Write a program to demonstate multilevel inheritance.

Solution :

Program to demonstrate multilevel inheritance

```
using System;
using System.Text;

namespace ConsoleApplication52
{
  class student
  {
    public int hin, mar, eng, sports, tot, avg;
    public void getmarks()
    {
      Console.Write("Enter marks of three subjects : ");
      hin = Convert.ToInt32(Console.ReadLine());
      mar = Convert.ToInt32(Console.ReadLine());
      eng = Convert.ToInt32(Console.ReadLine());
    }
  }

  class sportsclass : student
  {
    public void getsports()
    {
      Console.Write("Enter marks of sports : ");
      hin = Convert.ToInt32(Console.ReadLine());
    }
  }

  class result : sportsclass
  {
    public void cal()
    {
      tot = hin + mar + eng + sports;
      avg = tot / 3;
      Console.WriteLine("Total marks : " + tot);
      Console.WriteLine("Average : " + avg);
    }
  }

  class D
  {
    public static void Main(String[] args)
    {
      result r = new result();
      r.getmarks();
      r.getsports();
      r.cal();
    }
  }
}
```

Output

```
Enter marks of three subjects : 90
80
70
Enter marks of sports : 85
Total marks : 235
Average : 78
Press any key to continue . . .
```

➜ **4) Multiple inheritance**

- C# does not support multiple inheritances using only class. To overcome this problem, we can use interfaces.

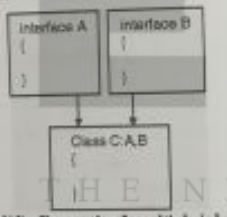


Fig. 1.2.4(d) : Example of multiple inheritance

☞ **Interface**

- Interface is just like a class but in which only declarations of methods is allowed, we cannot define any method.
- All the methods declared in an interface must be declared in the child classes.
- We cannot create an object of interface.

Program 1.2.41

Write a program to demonstrate multiple inheritance using interface.

Solution :

Program to demonstrate multiple inheritance using interface

```
using System;
using System.Text;

namespace ConsoleApplication52
{
  interface student
  {
    void getmarks();
  }
  interface sportsintf
  {
    void getsports();
  }
  class result : student, sportsintf
  {
    public int hin, mar, eng, sports, tot, avg;
    public void getmarks()
    {
      Console.Write("Enter marks of three subjects : ");
      hin = Convert.ToInt32(Console.ReadLine());
      mar = Convert.ToInt32(Console.ReadLine());
      eng = Convert.ToInt32(Console.ReadLine());
    }
    public void getsports()
    {
      Console.Write("Enter marks of sports : ");
      sports = Convert.ToInt32(Console.ReadLine());
    }
    public void cal()
    {
      tot = hin + mar + eng + sports;
      avg = tot / 4;
      Console.WriteLine("Total marks : " + tot);
      Console.WriteLine("Average : " + avg);
    }
  }

  class D
  {
    public static void Main(String[] args)
    {
      result r = new result();
      r.getmarks();
      r.getsports();
      r.cal();
    }
  }
}
```


Output

```
Enter marks of three subjects : 90
80
70
Enter marks of sports : 90
Total marks : 330
Average : 82
Press any key to continue . . .
```

➜ **5) Hybrid Inheritance**

Is the combination of more than one types of inheritances.

Program 1.2.42

Write a program to demonstrate Hybrid inheritance.

Solution :

Program to demonstrate hybrid inheritance

```
using System;
using System.Text;

namespace ConsoleApplication52
{
  interface student
  {
    void getmarks();
  }
  interface sportsintf : student
  {
    void getsports();
  }
  interface testintf
  {
    void gettest();
  }
  class result : sportsintf,testintf
  {
    public int hin, mar, eng, sports, test, tot, avg;
    public void getmarks()
    {
      Console.Write("Enter marks of three subjects : ");
      hin = Convert.ToInt32(Console.ReadLine());
      mar = Convert.ToInt32(Console.ReadLine());
```

```
eng = Convert.ToInt32(Console.ReadLine());

public void getsports()
{
  Console.Write("Enter marks of sports : ");
  sports = Convert.ToInt32(Console.ReadLine());
}
public void gettest()
{
  Console.Write("Enter marks of test : ");
  test = Convert.ToInt32(Console.ReadLine());
}
public void cal()
{
  tot = his + mat + eng + sports + test;
  avg = tot / 5;
  Console.WriteLine("Total marks : " + tot);
  Console.WriteLine("Average : " + avg);
}
}
class D
{
  public static void Main(String[] args)
  {
    result r = new result();
    r.getmarks();
    r.getsports();
    r.gettest();
    r.cal();
  }
}
```

Output

```
Enter marks of three subjects : 90
80
78
Enter marks of sports : 90
Enter marks of test : 89
Total marks : 419
Average : 83
Press any key to continue . . .
```

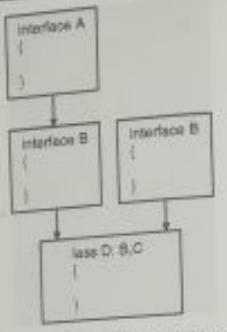


Fig. 1.2.4(e) : Example of hybrid interface

Syllabus Topic : Static Members

1.2.14 Static Members

Q. What is mean by static member? List the static members in class. **(2 Marks)**

- To create functions and variables of a class **static** keyword can be used.
- When we create members of a class as static, then there is no matter how many objects of that class are created. There is only one copy of each static member which is used by all the objects of that class.
- When the first object of class is created, all static data is initialized to zero if so, other initialization for the static data is present.
- We can declare following members of class as static.

Fig. C1.13 : Static members in class

➜ **1) Static Data Members**

- Class can have both static as well as non-static members.
- Static data members of class are also called as class members and non-static members of class are called as instance members.

- Static members are loaded in memory at compile time while instance members of class are loaded in memory at execution time of program.
- 'this' keyword refers methods and variables of current class. We can't use **this** keyword with static members of class to access them.

```
class StaticExample

{
    public static int age;
    public static string name= "Kunal";
}
```

➜ **2) Static Constructor**

- We cannot pass parameter to constructor which is created using static keyword.
- We cannot use any access modifier while defining static constructor.
- To initialize static data members of the class, Static Constructor is used.

➜ **3) Static Properties**

- To get or set the values of static data members of class , Static properties are used.

Program 1.2.43

Write a program to demonstate working of static properties(functions) which contain get and set functions.

Solution :

Program to demonstate working of static properties (functions) which contain get and set functions.

```
using System;
namespace staticExample
{
  class StatDemo
  {
    static string company_name;
    //StaticProperty
    public static string _company_name
    {
      get
      {
        return company_name;
      }
      set
      {
        company_name = value;
      }
    }
    static void Main(string[] args)
    {
      StatDemo.company_name = "ABC Pvt. Ltd.";
      Console.WriteLine(StatDemo.company_name);
    }
  }
}
```

Output

```
ABC Pvt. Ltd.
Press any key to continue . . .
```

➜ **4) Static Methods**

- While calling static method,we use class name and static method name only.
- No need to use object of class. Static methods use only static data members to perform calculation or processing.

Program 1.2.44

Write a program to demonstrate static constructor and static methods.

Solution :

Program to demonstrate static constructor and static methods

```
using System;
  namespace static_example
{
  class StaticDemo
  {
    public class test
    {
      static string tname;
      static int tage;
        static test()
      {
        Console.WriteLine("Use static constructor to
initialize static data members of class");
        tname= "Ishita";
        tage=4;
      }
```

```
public static void display()
{
    Console.WriteLine(name);
    Console.WriteLine(age);
}
}
static void Main(string[] args)
{
    test.display();
}
}
```

Output

```
Use static constructor to initialize static data members
Zahida
4
Press any key to continue . . .
```

Syllabus Topic : Casting Objects

1.2.15 Casting Object

Q. What is mean by casting object ? Explain types of casting object. **(4 Marks)**

- Casting object or Type conversion process is converting value from one data type to another type.
- Data type conversion is based on **type compatibility**.
- It is also called as Type Casting.
- In C#, type casting has two forms :

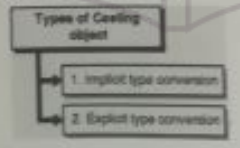


Fig. C1.14 : Types of casting object

➜ 1) Implicit type conversion

- In implicit conversion the compiler will make conversion of data from one data type to another without asking to programmer.
- The process of data conversion is performed by C# in a type-safe way i.e. convert data of one data type into other compatible data type. We can also say that we can convert source data from one type to targeting data type which require more memory size than type of source data.
- For example we can convert easily char value into integer and we can convert integer type value into float value easily.

➜ 2) Explicit type conversion

- These conversions are done explicitly by programmer using the built-in functions.
- Explicit conversions require a cast operator.

Program 1.2.45

Write a program to demonstrate explicit type casting.

Solution :

Program to demonstrate explicit type casting

```
using System;
namespace TypeConversionDemo
{
  class ExplicitDataTypeConversion
  {
    static void Main(string[] args)
    {
      double a = 3.2;
      int j;

      j = (int) a;

      Console.WriteLine(j);
    }
  }
}
```

Output

```
3
Press any key to continue . . .
```

- C# provides the following built-in type conversion methods :

1. **ToBoolean()**- Converts a data type of data member to a boolean value(true or false).
2. **ToByte()**-Convert data type of data member to byte.

3. **ToChar()**-Converts a data type of data member into character type.
4. **ToDateTime()**-Converts data type of data member which are either integer or string type into date-time format.
5. **ToDecimal()**-Converts a floating point or integer type data member into a decimal type.
6. **ToDouble()**-Converts data type of data member into a double type.
7. **ToInt16()**-Converts a data type of data member into a 16-bit integer.
8. **ToInt32()**-Converts a data type of data member into a 32-bit integer.

Program 1.2.46

Write a program of built-in functions used in explicit type casting.

Solution :

Program of built-in functions used in explicit type casting

```
using System;
namespace TypeConversionDemo
{
  class Example
  {
      static void Main(string[] args)
      {
          int a = 75;
          float b = 53.005f;
          double c = 2345.7652;
          bool d = true;

          Console.WriteLine(a.ToString());
          Console.WriteLine(b.ToString());
          Console.WriteLine(c.ToString());
          Console.WriteLine(d.ToString());
      }
  }
}
```

Convert integer value a into string

Convert float type value into string

Convert double type value into string

Convert Boolean data type value into string

Output

```
75
53.005
2345.7652
True
Press any key to continue . . .
```

Syllabus Topic : Partial Classes

1.2.16 Partial Classes

- It is possible to split the definition of a class into two or more source files. Each source file contains a section of the class definition.
- All these parts are combined then into single class, when the application is compiled.
- When working on large projects, spreading a class over separate files allows multiple programmers to work on it at the same time.
- Partial keyword can also use to split struct or interface over two or more files.

Benefit of partial classes

1) More than one developer can write the code for the class at same time.
2) Main advantage is that visual studio uses partial classes to separate automatically generated system code from developer code.

1.3 ASP.NET

Syllabus Topic : Creating Websites

1.3.1 Creating Websites

- ASP.NET is a web application framework designed as well as developed by Microsoft to provide facility to programmers to create attractive web sites, web applications and web services.
- It allows you to use full featured various programming languages such as C#, VB.NET etc. to create web applications easily.
- It provides good combination of HTML, CSS and JavaScript to create dynamic featured application.
- First version of ASP.NET was released in January 2002.
- It is based on the Common Language Runtime (CLR).

ASP.NET is a web development language which provides a programming model and a variety of services which are essential to create robust web applications for personal computer as well as for mobile devices.

- ASP.NET use HTTP protocol for client and server communication over internet.
- The ASP.NET application codes can be written in any of the following languages which support .NET framework :
  1. C# 2. Visual Basic.Net
  3. Jscript 4. J#
- ASP.NET is used to create attractive, data-driven web applications over the internet. It provides a large set of controls such as text boxes, buttons, labels, checkboxes etc. to create attractive user interface.

Syllabus Topic : Anatomy of a Web Form

1.3.2 Anatomy of a Web Form-Page Directive

Q. Write note on page directives. (4 Marks)

- Web forms in ASP.NET expand the event-driven model of interaction between client and server.
- The web browser submits a web form as request to the web server and the web server send a HTML page as response to web browser.
- All user activities at client side are informed to the server for maintaining state i.e. to record all the actions done by user.
- The server processes the input provided by client and provide output to client.
- ASP.NET framework use Page state and Session state to maintain state of page.
- The state of page is the state of the client, that means the content of number of input fields provided by the client in the web form.
- The session contains the collective information received from number of pages that user visited and worked with.

Syllabus Topic : Page Directive

Page Directive

- Fundamentally Page Directives are commands which are used by the compiler when the page is compiled.
- Page directives in ASP.NET are commands which state the optional settings like recording customized controls and page language etc.
- These settings explain way of processing the web forms to the .Net framework.
- It is simple to include directives to an ASP.NET page.

Syntax

```
<%@ directive_name attribute=value [attribute=value] %>
```

Following is the list of directives used in ASP.NET code :

1. The Page Directive

- When we want to state the attributes specific to an ASP.NET page then we need to use @Page Directive.

Syntax

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" Trace="true" %>
```

Example

```
<%@ Page Language="C#" AutoEventWireup="false" CodeFile="Default.aspx.cs" Inherits="_Default" %>
```

The attributes of the Page directive are listed below :

i) **AutoEventWireup** : It is used to enable or disable events of methods . It takes value true or false.

ii) **Buffer** : It is used to enable or disable HTTP response buffering.

iii) **ClientTarget** : The name of the browser for which the server controls render the content.

iv) **ClassName** : It is used to set class name of the page.

v) **Description**: Informative text which is ignored by the parser.

vi) **Language** : It is used to set programming language in which code is written.

vii) **Src**: It is used to set file name of the code outside the class.

viii) **ValidateRequest** : Used to check whether all input data is validated against a hardcoded.

ix) **CodeFile** : It is used to set name of the code behind file.

x) **Transaction** : It indicates whether the transactions are supported or not.

xi) **Debug** : The Boolean value that enables or disables compilation with debug symbols.

xii) **ErrorPage** : URL of the error page to transfer control if an unhandled exception occurs.

xiii) **EnableViewState** : The Boolean value that enables or disables view state across page requests.

xiv) **EnableSessionState** : The Boolean value that enables or disables session state across page requests.

Syllabus Topic : Doctype

1.3.3 Doctype

Q. What is use of Doctype in ASP.NET. ? (2 Marks)

- Use of doctype element ensures that output of page will be compliant to an XHTML standards.
- ASP.NET controls do not render font elements or attributes, such as bgcolor which would not conform to XHTML standards, if the page includes an XHTML DOCTYPE element.
- So ASP.NET permits us to create Web pages that are conforming to XHTML standards.
- XHTML is a World Wide Web Consortium (W3C) standard.
- XHTML defines HTML as an XML document.
- Creating Web pages that are conforming to XHTML standards has several advantages :

  It ensure that the elements in the pages are well formed.
- As many browsers are supporting XHTML, creating pages that conform to XHTML standards support to guarantee that your pages render in same way in all browsers.
- Using XHTML helps to make pages conform more readily to accessibility standards.
- XHTML is extensible, permitting to define new elements.
- We can easily read XHTML page programmatically for situations in which the Web page is processed by a computer instead of being read by users and transformations can be used to manipulate the document.

Syllabus Topic : Writing Code - Code - Behind Class

1.3.4 Writing Code-Code-Behind Class

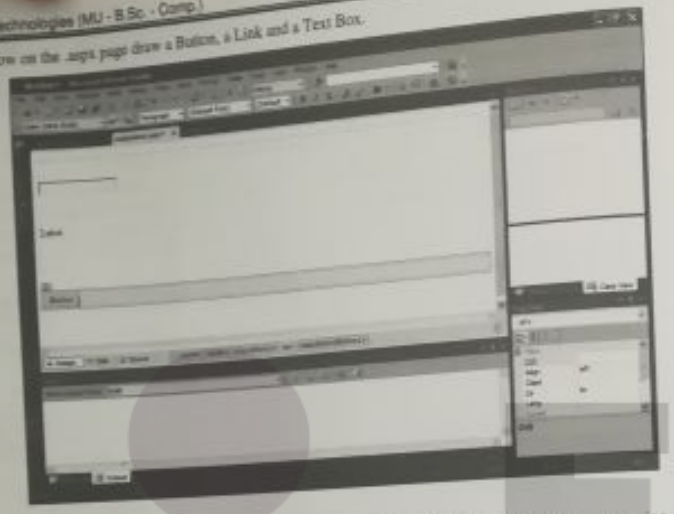
Q. What is mean by code-behind class? How to write code behind class? (4 Marks)

- Code Behind means code for an ASP.NET Web page that is written in a separate class file which can have the extension as .aspx.cs or .aspx.vb depending on the language used.
- We can write the code in a separate .cs or .vb code file for each .aspx page.
- One thing to remember in case of Code Behind is that the code for all the Web pages is compiled into a DLL file i.e. assembly that permits the web pages to be hosted free from any Inline Server Code.

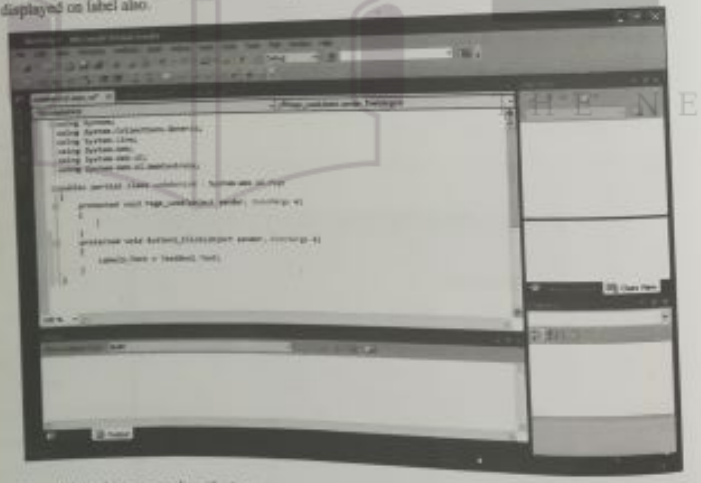
Steps to write code behind the class

Step 1 : Create a new Blank Website in the Visual Studio and then add a Web page to it. Here we are creating a Web site for the Code Behind.

Step 2 : Now on the .aspx page draw a Button, a Link and a Text Box.

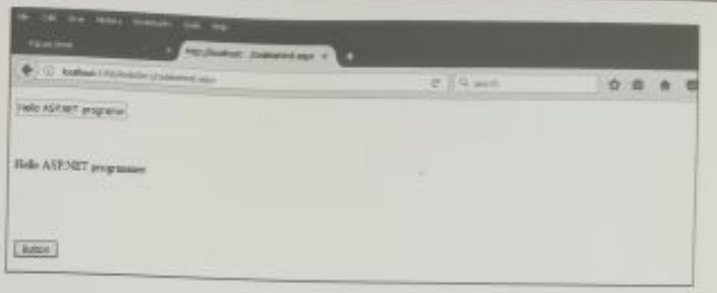


Step 3 : Now double-click on the Button, coding section will opened in a separate window whose extension is .aspx.cs. Write the code in this window. In this example we want that whenever text is written in textbox, it should get displayed on label also.



Step 4 : Now debug this page and verify that our program is running.

Output



Syllabus Topic : Adding Event Handlers

1.3.5 Adding Event Handlers

1.3.5(A) Event

Q. What is mean by event ? (2 Marks)

- Change in the state of an object is called as event.
- Events are generated in response of user interaction with the Graphical User Interface (GUI) components like button, checkbox, radio button etc.
- For example, clicking on a button, moving the mouse, entering a character through keyboard, selecting an item from list, scrolling the page are the activities that are responsible for occurrence of an event.

Types of Events

There are two types of events:

1. **Foreground Events**

- These events need direct interaction of user. They are generated in response to a person interacting with the graphical components such as button, checkbox, radio button etc. in Graphical User Interface.
- Example of foreground events are clicking on a button, moving the mouse, entering a character through keyboard, selecting an item from list, scrolling the page etc.

2. **Background Events**

- These events does not need direct interaction of user are known as background events.
- Examples of background events are Operating system interrupts, hardware or software failure, timer expires, an operation completion.
- Event Handling is the process that controls the event and decides what action should happen when an event occurs.
- In ASP.NET Events occurred at the client machine and these events are handled at the server machine.
- The server has a program or lines of code explaining what to do when the event is occurred. This program is called event handler.
- When event occurrence message is transmitted to the server, it checks what type of event occurs and then execute event handler written for that event.

1.3.5(B) Event Arguments

Q. Give syntax to define event. (2 Marks)

- ASP.NET event handlers take two arguments and do not return any value. The first argument represents the object due to which event occurs and the second is event argument.

Syntax

```
Private void EventName(object sender, EventArgs e);
```

1.3.5(C) Application and Session Events

Q. What are the application and session events? (2 Marks)

The important application events in ASP.NET are :

1. **Application_Start :** This event is occurred when the application is started.
2. **Application_End :** This event is occurred when the application stop its working.

The Session events are :

3. **Session_Start :** This event is occurred when a user first time requests a page from the application.
4. **Session_End :** This event is occurred when the session ends.

1.3.5(D) Page and Control Events

Q. What are the page and control events? (4 Marks)

☞ **Page and control events are :**

(i) **DataBinding :** This event is occurred when control is binds to a data source.

(ii) **Disposed :** This event is occurred when a page or the control is released.

(iii) **Error :** This event is occurred when an unhandled exception is thrown.

(iv) **Init :** This event is occurred when initialization of the page or the control is done.

(v) **Load :** This event is occurred when loading of the page or a control is done.

(vi) **PreRender :** This event is occurred when rendering of the page or the control is done.

(vii) **Unload :** This event is occurred when unloading of page or control from memory is done.

1.3.5(E) Event Handling Using Controls

Q. Write short note on event handling using controls? (4 Marks)

- All ASP.NET web controls i.e. Button, Checkbox etc. are implemented as classes and each control have events which are fired when user do specific action on that control.
- For example, when a user clicks a button the 'Click' event is generated.
- There are predefined attributes and event handlers which are used for handling events.
- Event handler is program which execute in response to an event and take appropriate action on it.
- By default an event handler is created by visual studio by adding a Handles clause on the Sub procedure. This clause assign name to the control and event which is handled by the procedure.

☞ **The ASP.NET tag for a button**

```
<asp:ButtonID="InstitutCancel" runat="server
"Text="Cancel" Onclick="InstitutCancel_Click"/>
```

The event handler for the Click event

```
Protected Sub InstitutCancel_Click(ByVal sender As Object,
ByVal e As System.EventArgs)
HandlesInstCancel.Click
End Sub
```

The list of common events ,there attributes and controls on which they applied are given below.

Event	Attribute	Controls
Click	OnClick	Button, Image button, link button, image map
SelectedIndexChanged	OnSelectedIndexChanged	Drop-down list, list box, radio button list, check box list
TextChanged	OnTextChanged	Text box
Command	OnCommand	Button, image button, link button
CheckedChanged	OnCheckedChanged	Check box, radio button

- Due to some events form need to be send back to the server immediately. Such events are called as the postback events. For example the click event post back the form to server when we click on button.
- Some events do not send form back to the server immediately and these events are called as non-postback events. For example, the change events or selection events such as TextBox.TextChanged or CheckBox.CheckedChanged.
- The nonpostback events can be convert into post back immediately by setting value of their AutoPostBack property to true.

Program 1.3.1

Write a program of event handling by controls.

Solution :

Program of event handling by controls

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default.aspx.cs" Inherits="ChangeEvents" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
  <title>Change Events</title>
</head>
<body>
```

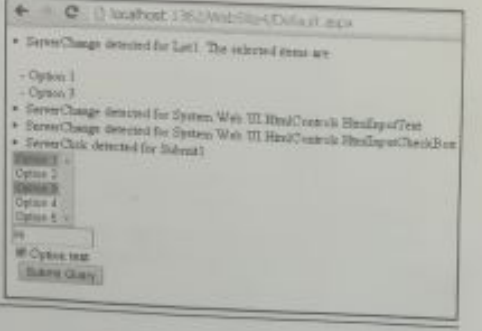


```
  <form id="form1" runat="server" >
  <div>
    <select runat="server"
        id="List1"
        size="5" multiple
        Name="List1"
        onservercange="List1_ServerChange">
    <option>Option 1</option>
    <option>Option 2</option>
  </select>
  <br/>
  <input type="text"
      runat="server"
      ID="Textbox1"
      Size="10"
      Name="Textbox1"
      OnServerChange="Ctrl_ServerChange"><br>
  <input type="checkbox"
      runat="server"
      ID="Checkbox1"
      Name="Checkbox1"
      OnServerChange="Ctrl_ServerChange">Option
text<br>
     
  <input type="submit"
      runat="server"
      ID="Submit1"
      Name="cmdSubmit"
      value="Submit Query"
      onserverclick="Submit1_ServerClick">
    </div>
  </form>
</body>
</html>
```

ASP.NET file Demo.aspx.cs

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
public partial class ChangeEvents : System.Web.UI.Page
{
    protected void Page_Load(object sender,
System.EventArgs e)
    {
        if (!Page.IsPostBack)
        {
            List1.Items.Add("Option 3");
            List1.Items.Add("Option 4");
            List1.Items.Add("Option 5");
        }
    }
    protected void Ctrl_ServerChange(object sender,
System.EventArgs e)
    {
        Response.Write("<li>ServerChange detected for
" + sender + "</li>");
    }
    protected void List1_ServerChange(object sender,
EventArgs e)
    {
        Response.Write("<li>ServerChange detected
for List1." + "The selected items
are:</li><br>"); foreach (ListItem li in
List1.Items)
        {
            if (li.Selected)
                Response.Write("  - " +
li.Value + "<br>");
        }
    }
    protected void Submit1_ServerClick(object sender,
EventArgs e)
    {
        Response.Write("<li>ServerClick detected for
Submit1.</li>");
    }
}
```

Output



5(F) Default Events

What is mean by default event and list some default event in C# ? (4 Marks)

- Every web control has a default event .for example, default event for the Page object is Load event and default event for the button control is the Click event.
- In visual studio, to create default event handler, we have to double click on the control in design view.
- List of default events for common controls :

Control	Default Event
RadioButton	CheckedChanged
RadioButtonList	SelectedIndexChanged
AdRotator	AdCreated
BulletedList	Click
Button	Click
CheckBox	CheckedChanged
Calender	SelectionChanged
CheckBoxList	SelectedIndexChanged
ImageButton	Click
ImageMap	Click
DataGrid	SelectedIndexChanged
DataList	SelectedIndexChanged
DropDownList	SelectedIndexChanged
LinkButton	Click
ListBox	SelectedIndexChanged
Menu	MenuItemClick
HyperLink	Click

Syllabus Topic : Anatomy of an ASP.NET Application-ASP.NET File Types

1.3.6 Anatomy of an ASP.NET Application-ASP.NET File Types

Q. Write note on ASP.NET file types. (4 Marks)

- Web site application can contains different types of files. Some file types are supported and managed by ASP.NET, and other file types are supported and managed by the Internet Information Server(IIS).
- Add New Item menu is used to create ASP.NET file types in Visual Studio. By using mapping, we can relate file type with application.

1.3.6(A) File Types Managed by ASP.NET

File type	Location	Description
.asax	Application root.	application class and optional methods i.e. event handlers , that run at various points in the application life cycle are present in Global.asax file.
.master	Application root or subdirectory.	Layout for other Web pages in the application. More information is defined by a master page.
.ascx	Application root or a subdirectory.	custom functionality that you can add to any ASP.NET Web Forms page is defined by this web user control file.
.config	Application root or a subdirectory.	XML elements that represent settings for ASP.NET features are present in this configuration file.
.ashx	Application root or a subdirectory.	To give response to a Web request in order to generate dynamic content , this file handler is invoked.
.dll	Bin subdirectory.	A Dynamic Link library (DLL) is a library file which includes functions and source code that can be used by many programs at same time.
.asmx	Application root or a subdirectory.	classes and methods that can be invoked by other Web applications are included in this XML web services file.
.browser	App_Browsers subdirectory.	features of an individual browser are identified in this browser definition file.
.aspx	Application root or a subdirectory.	Web controls presentation and business logic are present in this ASP.NET Web Forms page.

File type	Location	Description
.axd	Application root.	A handler file that is used to manage Web site administration requests, such as Trace.axd is managed by this handler file.
.skin	App_Themes subdirectory.	property settings to be applied to Web controls for consistent formatting is present in this skin file.
.cd	Application root or a subdirectory.	This file type represent class diagram file.
.soap	Application root or a subdirectory.	This is a SOAP extension file.
.cs, .vb	App_Code subdirectory, or in the case of a code-behind file for an ASP.NET page, in the same directory as the Web page.	code that can be shared between pages, such as code for custom classes, business logic, HTTP modules, and HTTP handlers is defined in this source code file.
.sln	Visual Studio project directory.	This is a solution file in Visual Studio.
.csproj, .vbproj	Visual Studio project directory.	This is project file in Visual Studio Web-application project.
.disco, .vsdisco	App_WebReferences subdirectory.	Web services are located by using this XML web services discovery file.
.sitemap	Application root or a subdirectory.	logical structure of the Web application is defined by this sitemap file. ASP.NET includes a default sitemap provider that uses sitemap files to display a navigational control in a Web page.
.dsdgm, .dsprototype	Application root or a subdirectory.	A distributed service diagram (DSD) file that can be added to any Visual Studio solution that provides or consumes Web services to reverse-engineer an architectural view.
.lics, .webinfo	Application root or a subdirectory.	To protect intellectual property by checking that a user is authorized to use the control is done by using licensing concept.

File type	Location	Description
.sdm, .sdmDocument	Application root or a subdirectory.	This is a system definition model (SDM) file.
.mdb, .ldb	App_Data subdirectory.	This is an Access database file.
.resources, .resx	App_GlobalResources or App_LocalResources subdirectory.	A resource file that contains resource strings that refer to images, localizable text, or other data are stored in this resource file.
.mdf	App_Data subdirectory.	This is SQL Server Express database file.
.msgx, .svc	Application root or a subdirectory.	This is a Indigo Messaging Framework (MFx) service file.

Syllabus Topic : ASP.NET Web Folders

1.3.7 ASP.NET Web Folders

Q. Write note on ASP.NET web folders. (4 Marks)

Following are the ASP.NET web folders :

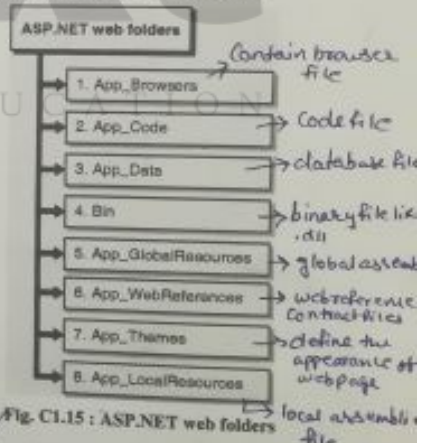


Fig. C1.15 : ASP.NET web folders

→ **1. App_Browsers**

- App_Browsers folder in a website includes web site related definitions of browser files that are used by ASP.NET to recognize browsers and states capabilities of those browsers.

2. App_Code

- The App_Code folder is always available in the our project. This folder contains Source code (for business objects and shared classes like cs, and vb files etc. which we want to compile as element of our application.
- ASP.NET perform compilation of the source code which is present in the App_Code folder on the first request to our application in a dynamically compiled Web site project.
- For any changes in source code, recompilation of contents which are present in this folder is performed.

3. App_Data

- Application data files such as .mdf database files, XML files, and another data store files are included in this folder.
- The App_Data folder is used by ASP.NET to store local database of application like the database for maintaining membership and role information.

4. Bin

- Compiled assemblies i.e. .dll files for controls, components, or other code that we want to use in application are included in this folder bin.
- To automatically refer any classes in our application, code for that class must be present in Bin folder.

5. App_GlobalResources

- Resources such as .resx and .resources files that are compiled into assemblies with global scope are included in this file.
- Resources in the App_GlobalResources folder are strongly typed and can be accessed programmatically.

6. App_WebReferences

- Reference contract files such as .wsdl files, schemas such as .xsd files and discovery document files such as .disco and .dissomap files that we create as Web reference for use in our application are included in this folder.

7. App_Themes

- Group of files such as .skin files, .css files, image files and generic resources that define the appearance of ASP.NET Web pages and controls are included in this folder.

8. App_LocalResources

- Resource files such as .resx and .resource files that are related with a particular page, user control, or master page in our application are included in this folder.

Syllabus Topic : HTML Server Controls

1.4 HTML Server Controls

- The HTML server controls are usually considered as the standard HTML controls which are created to enable server side processing.
- Number of HTML controls are not processed by the server but usually they are sent to browsers for display.
- When attribute runat="server" is added, they get converted into server control and when is attributed is added, they are available for server-side processing.
- For example, consider the HTML input control :

```
<input type="text" size="20" >
```

- Add the runat and id attribute to convert into server control :

```
<input type="text" id="FirstText" size="20"
runat="server">
```

Advantages of using HTML Server Controls

- Even though, All the functionalities of HTML server controls are available in ASP.NET server controls, there are some advantages of HTML server controls.
- Layout can be done using static tables.
- HTML page can be converted so they can run under ASP.NET
- The Table 1.4.1 describes the HTML server controls and corresponding HTML tags :

Table 1.4.1

Name of Control	HTML tag
HtmlHead	<head>element
HtmlInputButton	<input type=button|submit|reset>
HtmlInputCheckbox	<input type=checkbox>
HtmlInputFile	<input type = file>
HtmlInputHidden	<input type = hidden>
HtmlInputImage	<input type = image>
HtmlInputPassword	<input type = password>
HtmlInputRadioButton	<input type = radio>
HtmlInputReset	<input type = reset>
HtmlText	<input type = text|password>

Name of Control	HTML tag
HtmlImage	<img> element
HtmlLink	<link> element
HtmlAnchor	<a> element
HtmlButton	<button> element
HtmlButton	<button> element
HtmlForm	<form> element
HtmlTable	<table> element
HtmlTableCell	<td> and <th>
HtmlTableRow	<tr> element
HtmlTitle	<title> element
HtmlSelect	<select> element
HtmlGenericControl	All HTML controls not listed

Example

The following example uses a basic HTML table for layout. It uses some controls for accepting input from the users such as name, address, city, state etc. It also has a button control, which is clicked to get the user data displayed in the last row of the table.

Webform1.aspx

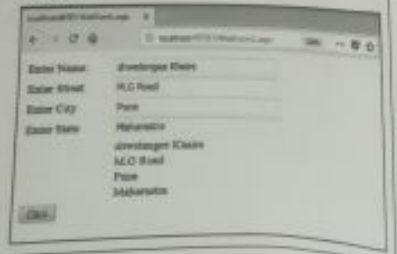
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication11.WebForm1" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <table style="width: 54%;">
      <tr>
        <td >Enter Name:</td>
        <td>
          <asp:TextBox ID="name" runat="server"
style="width:230px">
          </asp:TextBox>
        </td>
      </tr>

      <tr>
        <td >Enter Street</td>
        <td>
          <asp:TextBox ID="street" runat="server"
style="width:230px">
          </asp:TextBox>
        </td>
      </tr>

      <tr>
        <td >Enter City</td>
        <td>
          <asp:TextBox ID="city" runat="server"
style="width:230px">
          </asp:TextBox>
        </td>
      </tr>

      <tr>
        <td >Enter State</td>

        <td>
          <asp:TextBox ID="state" runat="server"
style="width:230px">
          </asp:TextBox>
        </td>

      </tr>

      <tr>
        <td class="style1"></td>
        <td ID="displayrow" runat ="server"
class="style2">
        </td>
      </tr>

    </table>
  </div>
  <asp:Button ID="Button1" runat="server"
onclick="Button1_Click" Text="Click" />
  </form>
</body>
</html>
```

Code behind file

Webform1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication11
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender,
EventArgs e)
        {
            string str1 = "";
            str1 += name.Text + "<br />";
            str1 += street.Text + "<br />";
            str1 += city.Text + "<br />";
            str1 += state.Text + "<br />";
            displayrow.InnerHtml = str1;
        }
    }
}
```

Output



Syllabus Topic : View State

1.4.1 View State

Q. What is mean by view state and how to create view state? (4 Marks)

- We know that web application uses HTTP protocol as it is stateless.
- That means a new object of a page is created each time when we make a request to the server to get the page and after the execution of our page, page object has been lost immediately.
- It only happens because of one server on which all the controls of the Web Page are created and after use of those controls, the server destroys all the objects of that page. So to store the values of the controls, we use state management techniques.

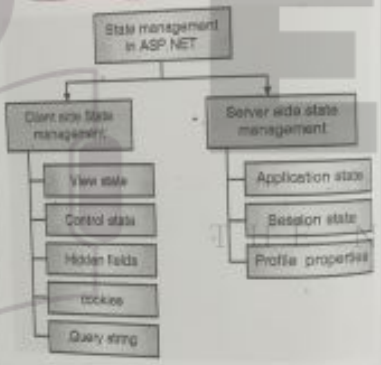


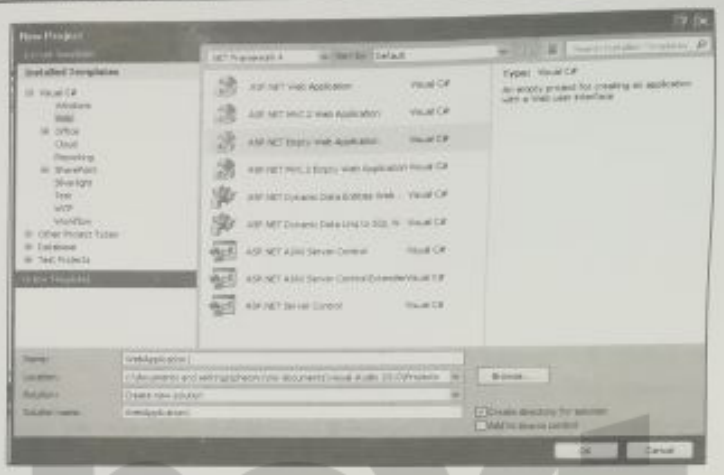
Fig 1.4.1 : State Management

- View State is the method to maintain the Value of the Page and Controls between round trips.
- View state is a Page-Level State Management technique. View State is turned on by default and normally serializes the data in every control on the page regardless of whether it is actually used at the time of a post-back.

Steps to create view state

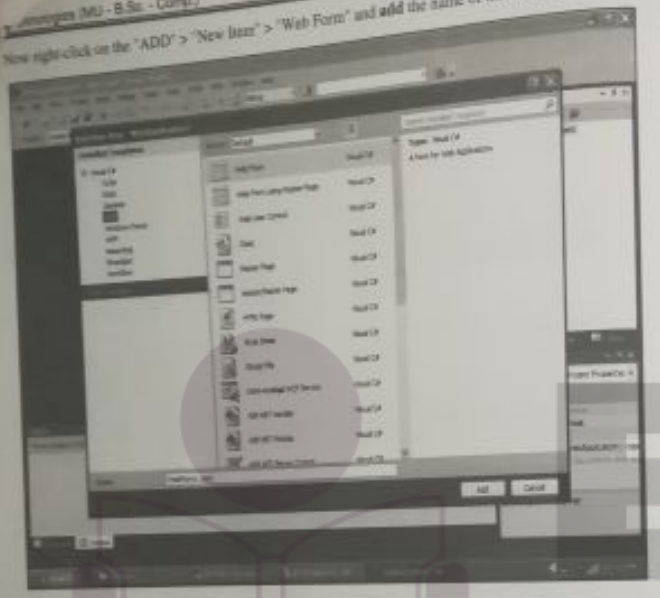
Step 1 : Open Visual Studio 2010.

Step 2 : Then click on "New Project" > "Web" > "ASP.NET Empty Web Application".



Step 3 : Now click on Solution Explorer.

Now right-click on the "ADD" > "New Item" > "Web Form" and add the name of the Web Form.



Step 5 : Write following HTML and ASP.NET code and run web form.

Program 1.4.1 : Write a HTML code.

Solution :

HTML Code

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication1.WebForm1" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
    user Name:-<asp:textbox id="TextBox1"
runat="server"></asp:textbox>
<br />
    Password :-<asp:TextBox id="TextBox2"
runat="server"></asp:textbox>
<br />
<asp:button id="Button1" runat="server"
onclick="Button1_Click" text="Submit" />
<asp:button id="Button3" runat="server"
onclick="Button3_Click" text="Restore" />
</div>
</form>

</body>
</html>
```

Program 1.4.2

Write a program of view state ASP.NET page

Solution : Program of view state ASP.NET page

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

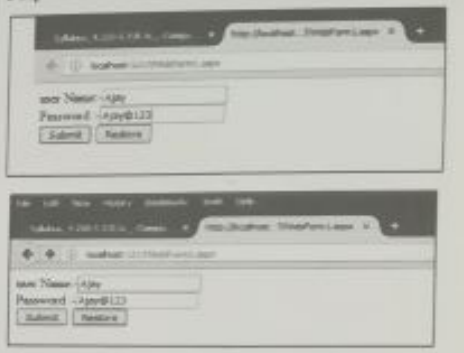


```
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication1
{
  public partial class WebForm1 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    public string a, b;
    protected void Button1_Click(object sender,
EventArgs e)
    {
      //
      ViewState["name"] = TextBox1.Text;
      ViewState["password"] = TextBox2.Text;

    TextBox1.Text = TextBox2.Text = string.Empty;
    }

    protected void Button3_Click(object sender,
EventArgs e)
    {

  if (ViewState["name"] != null)
  {
    TextBox1.Text = ViewState["name"].ToString();
  }
  if (ViewState["password"] != null)
  {
    TextBox2.Text = ViewState["password"].ToString();
  }

    }
  }
}
```

Value of TextBox1 and TextBox2 is assign on the ViewState

After clicking on submit Button TextBox value Will be Cleared

Output



- When we click on the Submit Button, the value of user name and password is submitted in View State and the View State stores the value of user name and password at the time of post-back.
- When we click on the Restore Button, we can get the value of user name and password again.
- The Value of user name and password must be retained at the time of post-back and the values are stored into a base 64 encoded string and this information is then put into the View State Hidden Field.

Features of View State

1. There is no need of session to retain the control values after post-back.
2. The values of all the Pages as well as Control Properties are stored.
3. Generates a custom View State Provider which helps to store View State Information in any data source like SQL Server.

Advantages of View State

1. Easy to use.
2. No server side resources are required to maintains view state.
3. View state enhances security features using Unicode implementation.

Disadvantages of View State

1. **Security Risk :** The Information of View State can be seen in the page output source directly. Using extra coding, you can manually do encryption and decryption of contents of a Hidden Field. If security is

important then consider use of Server-Based state Mechanism so that no sensitive information is sent to the client.

2. **Performance :** If we use a huge amount of data then performance is not good, because View State is stored in the page itself and storing a large value can cause the page to be slow.
3. **Device limitation :** We cannot store large amount of view data in Mobile Devices, because they may not have that much memory capacity to store.
4. View state able to store values for the same page only.

When We Should Use View State

1. When amount of data to be stored is small.
2. Data is not sensitive.

Syllabus Topic : HTML Control Base Class

1.4.2 HTML Control Base Class

Q. Write note on HtmlControl base class. **(4 Marks)**

- System.Web.UI.Html Controls is a namespace which contain HtmlControl class which is parent class for other HTML control classes.
- The HtmlControl object is very important to HTML server controls, because every property and method in this class is derived by each HTML server control class.
- If we understand the methods and properties of the HtmlControl class then we can understood about 80% of the methods and properties of all the objects in the System.Web.UI.HtmlControls namespace.
- Using HtmlControl class, we can easily understand other control classes.
- The HTML server controls have their own methods and properties. But their properties and methods are included in the HtmlControl object.

Properties of HtmlContols class

Property	Description
Visible	This property gives a Boolean value i.e. either true or false. We can get and set value of this property. Value of this property indicates whether a control is rendered to HTML for delivery to the browser of client or not.
Attribute	Group of attributes of HtmlControl object are returned by this property.
Disabled	This property returns boolean value i.e. either true or false. We can get and set value of this property. Value of this property indicates whether a control is disabled or not.
ID	Value of this property is string that you can get and set. This property defines the Identifier for the control.
EnableViewState	This property returns boolean value i.e. either true or false. We can get and set value of this property. Value of this property indicates whether a control should maintain its view state or not.
TagName	Tag name of an element such as input or div are returned by this property.
Style	CSS Style Collection for a control is returned by this property.

Program 1.4.3

Write a program of HtmlControl class.

Solution : Program of HtmlControl Class

```
<%@ page language="cs" runat="server"%>
<script runat=server>
void Page_Load()
{
OurDiv.InnerHtml = "Our align attribute = " +
OurDiv.Attributes["align"] + "<br>";
OurDiv.InnerHtml += "Our font-size style = " +
OurDiv.Style["font-size"];
}
</script>
<html>
<head>
<title>HtmlControl Collections</title>
</head>
<body>
<div id="OurDiv" align="center" style="font-size:14pt;
font-weight:bold" runat="server"/>
</body>
</html>
```


Output

Syllabus Topic : HtmlContainerControl

1.4.3 HtmlContainerControl

Q. Write note on HtmlContainerControl class? **(4 Marks)**

- System.Web.UI.HtmlControls is a namespace which contains HtmlContainerControl class.
- HtmlContainerControl is used as the parent class for all HTML controls that requires a closing tag such as div, select, form etc.
- All properties and methods of HtmlContainerControl class are derived from the HtmlControl class and adds few properties and methods of its own.

Syntax

```
public abstract class HtmlContainerControl : HtmlControl
```

Constuctors of HtmlContainerControl class

1. **HtmlContainerControl() :** This is default constructor of HtmlContainerControl. This constructor Initializes a new instance of the HtmlContainerControl class using default values.
2. **HtmlContainerControl(String) :** This is parametrized constructor of HtmlContainerControl class. It initializes a new instance of the HtmlContainerControl class using the mentioned tag name.

Properties of HtmlContainerControl class

Name	Description
Adapter	This property is derived from HtmlControl class which is used to get the browser-specific adapter for the control.
AppRelativeTemplate SourceDirectory	This property is derived from HtmlControl class which is used to get or set the application-related virtual directory of the Page or UserControl object that contains this control.
Events	This property is read-only and derived from HtmlControl class.This property is used to get a event handler delegates list for the control.
Attributes	This property is derived from HtmlControl which is used to get a group of all attribute name and value pairs mentioned on a server control tag within the ASP.NET page.
ID	This property is derived from HtmlControl class and it is used to get or set the programmatic identifier to the server control.
BindingContainer	This property is derived from HtmlControl class. This API supports the product infrastructure and is not intended to be used directly from our code. This property is used to get the control that contains this control's data binding.
ChildControlsCreated	This property is derived from HtmlControl class. This property is used to get a value that state whether the server controls and child controls have been created or not.
InnerHtml	This property is used to get and set the content found between the opening and closing tags of the specified HTML server control.
ClientID	This property is inherited from HtmlControl class and used to get the control ID for HTML markup that is created by ASP.NET.
ClientIDMode	This property is inherited from HtmlControl class and used to get and set the algorithm which is used to generate the value of the ClientID property.
Visible	This property is derived from HtmlControl class which is used to get and set a value that state whether a server control is rendered as User Interface on the page or not.
Context	This property is derived from HtmlControl class which is used to get the HttpContext object related with the server control for the current Web request.

Name	Description
ViewState	This property is used to get a dictionary of state information that permit us to save and restore the view state of a server control across multiple requests for the same page.
SkinID	This property is derived from HtmlControl class which is used to get and set the skin to apply to the control.
Disabled	This property is derived from HtmlControl class which is used to get and set a value which state whether the HTML server control is disabled or not.
HasChildViewState	This property is derived from HtmlControl class which is used to get a value which state whether controls of current server child controls have any saved view-state settings.
Style	This property is derived from HtmlControl class which is used to get group of all cascading style sheet (CSS) properties which can be applied to a specified HTML server control in the ASP.NET file.
Page	This property is derived from HtmlControl class which is used to get a reference to the naming container of server control which creates a unique namespace for differentiating between server controls with the same ControlID property value.
IsChildControlStateCleared	This property is derived from HtmlControl class which is used to get a value stating whether controls contained within this control have control state or not.
IsTrackingViewState	This property is derived from HtmlControl class which is used to get a value that state whether the server control is saving changes to its view state or not.
IsViewStateEnabled	This property is derived from HtmlControl class which is used to get value indicating whether view state is enabled for this control.

Name	Description
NamingContainer	This property is derived from HtmlControl class which is used to get a reference to the naming container of server control which creates a unique namespace for differentiating between server controls with the same value of Control ID property.
Parent	This property is derived from HtmlControl class which is used to get reference to the parent control of server control in the page control hierarchy.
RenderingCompatibility	This property is derived from HtmlControl class which is used to get a value that states the ASP.NET version with which rendered HTML will be compatible with.
TagName	This property is derived from HtmlControl class which is used to get tag runat=server attribute and value pair.

Program 1.4.4

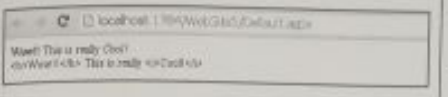
Write a program of HtmlContainerControl class.

Solution :

Program of HtmlContainerControl Class

```
<%@ page language="c#" runat="server"%>
<script runat=server>
 void Page_Load()
 {
  string OurText = "<b>Wow!! </b> This is really
<i>Cool!</i>";
  OurDiv1.InnerHtml = OurText;
  OurDiv2.InnerText = OurText;
 }
</script>
<html>
<head>
<title>HtmlControl InnerStuff</title>
</head>
<body>
<div id="OurDiv1" style="font-size:14px"
runat="server"/>
<div id="OurDiv2" style="font-size:14px"
runat="server"/>
</body>
</html>
```


Output



Syllabus Topic : HtmlInputControl Class

1.4.4 HtmlInput

Q. Write note on HtmlInput class. (4 Marks)

- The HtmlInput class is derived from the Html Control like HtmlContainerControl class and include some properties of its own .

Properties of HtmlInput class

Property	Description
Name	This property is used to get or set the unique name for the HtmlInput control.
Value	This property is used to get and set the value of the contents of the HtmlInput object.
Type	This property states the type of Input element HtmlInput control.

HtmlInput object types

Type	Html Server Control	Tag
Button	HtmlInputButton	<input type='button' runat='server'>
CheckBox	HtmlInputCheckBox	<input type='checkbox' runat='server'>
File	HtmlInputFile	<input type='file' runat='server'>
Hidden	HtmlInputHidden	<input type='hidden' runat='server'>
Image	HtmlInputImage	<input type='image' runat='server'>
Password	HtmlInputText	<input type='password' runat='server'>
Radio	HtmlInputRadioButton	<input type='radio' runat='server'>
Reset	HtmlInputButton	<input type='reset' runat='server'>
Submit	HtmlInputButton	<input type='submit' runat='server'>
Text	HtmlInputText	<input type='text' runat='server'>

Syllabus Topic : Page Class

1.4.5 Page Class

Q. How to generate and run page class? (4 Marks)

- When an ASP.NET page is requested and HTML output is send to browser, ASP.NET creates an object of the class to represents the page.
- That class not only contains code that we have written for the page, but also the code that is automatically generated by ASP.NET.
- An ASP.NET page is executed as a single unit which includes server-side elements in a page such as web controls and event-handling code that we have written.
- We do not need to pre-compile pages into assemblies because we use a Web site project.
- Pages are dynamically complied by ASP.NET and executed when they are requested by a user at first time.
- When there is any change in page or resource , it needs to recompile the page.
- To enhance performance, pre-compilation of a Web project is done which is supported by web site project.
- ASP.NET Web application projects should be explicitly compiled before deployment at client side.
- The class or classes that the compiler creates depends on whether the page uses the single-file model or the code-behind model.

Single - file page

- We can create single-file pages in a Web site project, in which event-handling code ,the markup and server-side elements are all included in a single .aspx file.
- The ASP.NET generates and compiles a new class that is derived from the base Page class or from a custom base class defined with the derived attribute of the @ Page directive when page is compiled.
- For example, if you create a new ASP.NET Web page whose name is CodeSample1 in root directory of our application then a class named ASP.CodeSample1.aspx is generated that derives the Page class.
- The subfolder name is used as part of the generated class for pages in application subfolders.
- The generated class includes declarations for the controls in the .aspx page and the event handlers and other custom code are included in generated class.
- The generated class is compiled into an assembly, and when the page is requested, the assembly is loaded into the application domain, and then the page class is instantiated and executed to render output to the browser.

...t a Web site project, if you make changes to the page ...at would affect the generated class whether by adding controls or modifying the code, the compiled class code is invalidated and a new class is generated.

Code-Behind Pages

- Code-behind pages are by default present in Web application projects and are optional in Web site projects.
- In the code-behind model, the markup of page and server-side elements, including control declarations, are in an .aspx file and our page code is in a separate code file.
- Partial class included in code file i.e partial keyword used in class declaration indicates that it includes only some of the total code that create the full class for the page.
- In the partial class, we add the code that our application needs for the page. This typically consists of event handlers, but can include any methods or properties that we need.
- The inheritance model for code-behind pages is difficult to implement than that for single-file pages.

Syllabus Topic : Global.asax File

1.4.6 Global.asax File

Q. What is global.asax file ? (1 Mark)

- This is an optional file which is known as ASP.NET application file.
- If we does not define this file then ASP.NET page framework assumes that application or session event handlers are not defined.
- This file permits writing event handlers that handles the global events.

Syllabus Topic : Web.config

1.4.7 Web.config

Q. Write note on web.config (4 Marks)

- ASP.NET applications use configuration system that enables defining configuration settings for the web server or a web site.
- These setting can be done at time when ASP.NET application is delivered to client.
- Configuration setting can be added or repeated at any time with less impact on application and web server functions.
- These setting are stored in XML/K-based file.
- It is simple to make changes in configuration of application because format of this file is ASCII text file.
- ASP.NET uses two files for configuration setting of application which are machine.config and web.config.

Review Questions

Q. 1 What is .NET framework architecture ? (4 Marks)
(Refer Section 1.1)

Q. 2 Explain data types in C#. (4 Marks)
(Refer Section 1.2.2)

Q. 3 What is constant ? Explain types of constant in c#. **(Refer Section 1.2.3)** (4 Marks)

Q. 4 What is keyword and list some keyword in C# ? (2 Marks)
(Refer Section 1.2.5)

Q. 5 Write note on types of function call. (4 Marks)
(Refer Section 1.2.9(B))

Q. 6 Write note on namespaces. (4 Marks)
(Refer Section 1.2.12(A))

Q. 7 Write note on assemblies. (4 Marks)
(Refer Section 1.2.12(C))

Q. 8 What is mean by casting object ? Explain types of casting object. (4 Marks)
(Refer Section 1.2.15)

Q. 9 What is mean by code-behind class? How to write code behind class? (4 Marks)
(Refer Section 1.3.4)

Q. 10 What is mean by event ? (2 Marks)
(Refer Section 1.3.5(A))

Q. 11 What is mean by view state and how to create view state? **(Refer Section 1.4.1)** (4 Marks)

Q. 12 Write note on HtmlContainerControl class? (4 Marks)
(Refer Section 1.4.3)

Q. 13 Write note on HtmlInput class. (4 Marks)
(Refer Section 1.4.4)

UNIT II

CHAPTER 2

Web Controls

Syllabus Topics

Web Controls : Web Control Classes, WebControl Base Class, List Controls, Table Controls, Web Control Events and AutoPostBack, Page Life Cycle.

State Management : ViewState, Cross-Page Posting, Query String, Cookies, Session State, Configuring Session State, Application State.

Validation : Validation Controls, Server-Side Validation, Client-Side Validation, HTML5 Validation, Manual Validation, Validation with Regular Expressions.

Rich Controls : Calendar Control, AdRotator Control, MultiView Control

Themes and Master Pages : How Themes Work, Applying a Simple Theme, Handling Theme Conflicts, Simple Master Page and Content Page, Connecting Master pages and Content Pages, Master Page with Multiple Content Regions, Master Pages and Relative Paths.

Website Navigation : Site Maps, URL Mapping and Routing, SiteMapPath Control, TreeView Control, Menu Control.

Syllabus Topic : Web Control

2.1 Web Controls

Q. List all ASP.Net web controls. (2 Marks)

- Small building blocks of the Graphical User Interface are controls, which include buttons, check boxes, list boxes, text boxes, labels, and various other tools.
- Users can enter data, make selections and state priority of selection using these web controls.
- Structural jobs such as validation, data access, security, creating master pages, and data manipulation etc. can be done using these web controls.
- In ASP.NET,there are five types of web controls available which are :
  i) HTML controls
  ii) HTML server controls
  iii) ASP.NET server controls
  iv) User controls and custoASP.NET
  v) User controls and custom controls

Syllabus Topic : Web Control Classes

2.1.1 Web Control Classes

- ASP.NET web control classes are the basic control classes used in ASP.NET.
- These web control classes can be grouped into the following categories :

i) **Validation controls :** Web control classes of this type are used to validate user input and they work by using client-side script which is written in clientside scripting language - JavaScript.

ii) **Data source controls :** Web control classes of this type provides feature of data binding to different data sources.

iii) **Data view controls :** Web control classes of this type are various lists and tables which can bind the data from data sources for displaying the data.

v) **Personalization controls :** Web control classes of this type are used for personalization of a page according to the user priority based on user information.

v) **Login and security controls :** Web control classes of this type are controls which provide user authentication.

vi) **Master pages :** Web control classes of this type provides constant layout and interface throughout the application.

vii) **Navigation controls :** Web control classes of this type helps in navigation. For example, menus, tree view etc.

viii) **Rich controls :** Web control classes of this type implement various types of special features. For example, AdRotator, FileUpload, and Calendar control.

Syntax

```
<asp:controlType ID="ControlID" runat="server"
Property1=value1 [Property2=value2] />
```

- Visual studio provides following features while writing program in ASP.NET:

1) Visual studio provides Dragging and dropping facility for controls in design view.
2) Visual studio provides IntelliSense feature which show and auto-complete the properties.
3) The properties window is used to set the values of property directly.

Properties of the web Control classes

1) ASP.NET web control classes with a visual aspect are derived from the WebControl base class and can access all the properties, events, and methods of this class.
2) The WebControl class itself and some other web control classes that are not visually rendered are inherited from the System.Web.UI.Control class. For example, PlaceHolder control or XML control.
3) ASP.Net web control classes access all properties, events, and methods of the WebControl and System.Web.UI.Control class.
4) The Table 2.1.1 state the derived properties, common to all web control classes.

Table 2.1.1

Property	Description
Back Color	This property is used to set background color of web control.
AccessKey	This property is used to set focus on web control.
BorderWidth	This property is used to set Border width of web control.
Attributes	This property makes the collection of arbitrary attributes which do not correspond to properties on the web control.
BindingContainer	Set the control that contains this control's data binding.
BorderColor	This property is used to set Border color of web control.
ClientID	This property state Control ID for HTML markup.
Causes Validation	This property shows if web control causes validation.
Border Style	This property is used to set Border style of web controls.
Child Control Created	This property state that whether child controls created from web control or not.
Enabled	This property show that control is enabled or disabled i.e active or deactive.
Context	This property define HttpContext object which is associated with the server control.
ControlStyle	This property set style of the Web server control.
Css Class	This property state CSS class for our current program.
Font	This property set font of content in web controls.
DataKeysContainer	If the naming container implements IDataKeysContainer then this property is used to get a reference to the naming container.
ID	This property set Identifier for the control.
EnableTheming	This property state whether theming applies to the web control or not.

Property	Description
EnableViewState	This property state whether the view state of the control is maintained or not.
Events	This property is used to get a list of event handler delegates for the control.
Forecolor	This property is used to set Foreground color of web control.
HasAttributes	This property state whether the control has set of attributes or not.
HasChildViewState	This property state whether the current child controls of web controls have any saved view-state settings or not.
Height	This property is used to set Height of web control in pixels or %.
IsEnabled	This property is used to get a value which state whether the control is enabled or not.
Page	This property is used to create Page containing the control.
Parent	This property is used to set Parent for web control.
SkinID	This property is used get and set skin to apply to the control.
Style	This property is used to get a collection of text attributes that will be rendered as a style attribute on the outer tag of the Web server control.
TagName	This property is used to get the name of the control tag.
UniqueID	This property is used to set Unique identifier to web control
ViewState	This property is used to get a dictionary of state information that saves and restores the view state of a web control across multiple requests for the same page.
Visible	This property states whether a server control is visible.
Width	This property is used to get and set the width of the Web server control.

Methods of the web Control classes

Table 2.1.2 provides the some methods of the web control classes.

Table 2.1.2

Method	Description
AddAttributesToRender	To add HTML attributes and styles in our web page that need to be rendered to the specified HtmlTextWriterTag, AddAttributesToRender() is used.
SaveViewState	This method is used save any state that was modified after the TrackViewState method was invoked.
Focus	This method is used to Sets input focus to a control.
ClearChildState	This method is used to deletes the view-state and control-state details of child controls of web control.
ClearChildViewState	This method is used to delete the view-state information for all the child controls of web controls.
CreateChildControls	This method is used in creating child controls.
DataBind	This method is used to binds a data source to the web control and all its child controls.
DataBind(Boolean)	This method is used to bind a data source to the server control and all its child controls with an option to raise the DataBinding event.
Dispose	This method is used to enable a web control to perform final clean up before it is released from memory.
EnsureChildControls	This method is used to check whether the web control has child controls. If there is no child control exists then creates it.

Method	Description
EnsureID	This method is used to create identifier for controls that do not have an identifier.
Equals(Object)	This method state whether the specified object is equal to the current object or not.
Finalize	Before reclaimed the object by garbage collection, this method allows an object to attempt to free resources and perform other cleanup operations.
FindControl(String)	This method performs searching the current naming container for a server control with the determined id parameter.
GetType	This method is used to get the type of the current instance.
HasControls	This method checks whether the server control contains any child controls or not.
HasEvents	This method states whether events are registered for the control or any child controls.
OnInit	This method is used to raises the Init event.
OnLoad	This method is used to raises the Load event.
OnPreRender	This method is used to raise the PreRender event.
OnUnload	This method is used to raise the Unload event.
OpenFile	This method is used to open the file.
RemovedControl	This method is called after a child control is removed from the controls collection of the control object.
Render	This method is used to render the control to the specified HTML writer.
SaveControlState	This method is used to Saves any server control state changes that appear.
ToString	This method Returns string which represents the current object.
TrackViewState	This method is used to control track of changes to view state of control so that they can be stored in the object's view state property.

Syllabus Topic : WebControl Base Class

2.1.2 WebControl Base Class

- The Web Control class is Serves as the base class which creates the methods, properties and events for all web controls in the System.Web.UI.WebControls namespace.

Syntax

```
Public class WebControl : Control, IAttributeAccessor
```

Following is list of constructors in Web Control base class

Sr. No.	Name	Description
1.	WebControl()	This default constructor is used for initialization of a new instance of the Web Control class that represents a Span HTML tag.
2.	WebControl (HtmlTextWriterTag)	This parameterized constructor is used for initialization of new instance of the Web Control class by using the specific HTML tag.
3.	WebControl(String)	This parameterized constructor is used to initialize a new object of the Web Control class using the specific HTML tag.

Events

Sr. No.	Name	Description
1.	DataBinding()	This event occurs when data source is bind to server control.
2.	Init	This event is occurs when initialization of the server control is done, which is the first step in its lifecycle ASP.NET page

Sr. No.	Name	Description
3.	Load	This event occurs when web control is loaded in page object.
4.	PreRender	This event occurs after the page object has created all the controls that are needed in order to render the page. This page object raises prerender event on the page object and then repeat same for each child control. The prerender event of the page is followed by prerender event of individual Controls.
5.	Unload	After rendering of page, the real cleanup started and the page unload event is raised to clean the memory occupied by that application.
6.	Disposed	This event occurs when a web control is released from memory, which is the last stage of the server control lifecycle when an ASP.NET page is requested.

Syllabus Topic : List Controls

2.1.3 List Controls

Q. Write short note on : List Controls. (4 Marks)

- List controls in ASP.NET are special web controls that contains Drop-down list,List box,Radio button list,Check box list,Bulleted list etc. that may or may not bound to data source or programmatically fills with items.
- These controls allow a user to choose from one or more items from the list.
- These controls are derived from System.Web.UI - WebControls.ListControl class.
- All controls from list controls allow user to select options except Bulletedlist. Bulletedlist show static data.
- RadioButtonList or CheckBoxList provides multiple checkboxes or option buttons.
- DropDownList and ListBox shows predefined values or values which are added at runtime.
- Default event for all this controls is SelectedIndexChanged except for BulletedList.
- This event is fired automatically when autopostback property of control is set to true.

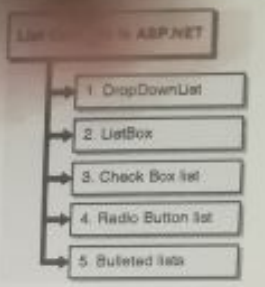


Fig. C2.1 : List Controls in ASP.NET

1. DropDownList

Syntax

```
<asp:DropDownList ID="DropDownList1" runat="server"
AutoPostBack="True" OnSelectedIndexChanged
="DropDownList1_SelectedIndexChanged">
</asp:DropDownList>
```

- This control permit users to select item from predefined list.
- It does not remain hidden until user click on dropdown button like ListBox and it does not support for selecting multiple items at same time.

Following are some properties of drop-down Lists

Property	Description
Items	This property provides group of List Item objects that presents the items in the control. Object of type List Item Collection is returned by this property.
Selection Mode	This property state whether a list box permits single selection or multiple selection.
Rows	This property determines the number of items shown in the box. Scroll bar is added when necessary.
Selected Index	This property determines index of the currently selected item. The index of the first selected item is determined by this property if more than one items are selected. If no item is selected then value of this property is -1.

Property	Description
Selected Value	Returns the value of the currently selected item. If more than one items are selected, then returns the value of the first selected item. If no item is selected, the value of this property is an empty string ("").

2. ListBox

syntax

```
<asp:ListBox ID="ListBox1" runat="server" AutoPostBack="True" OnSelectedIndexChanged="ListBox1_SelectedIndexChanged">
</asp:ListBox>
```

- ListBox web control can be used to show multiple items at a time which permit user to select one or more items from predefined list.

Following are some properties of ListBox

Property	Description
Items	This property provides group of List Item objects that presents the items in the control. Object of type List Item Collection is returned by this property.
Selection Mode	This property state whether a list box permits single selection or multiple selection.
Rows	This property determines the number of items shown in the box. Scroll bar is added when necessary.
Selected Index	This property determines index of the currently selected item. The index of the first selected item is determined by this property if more than one items are selected. If no item is selected then value of this property is -1.
Selected Value	Returns the value of the currently selected item. If more than one items are selected, then returns the value of the first selected item. If no item is selected, the value of this property is an empty string ("").

3. Check Box list

Syntax

```
<asp:CheckBoxList ID="CheckBoxList1" runat="server" AutoPostBack="True" OnSelectedIndexChanged="CheckBoxList1_SelectedIndexChanged">
</asp:CheckBoxList>
```

- It provides multiselection check box group that can be populated at design time as well as runtime.
- It includes Items collection with members corresponding to each item in list.
- To specify checked items, group can be iterated and selected property of each item in list can be tested.

Following are properties of check box lists

Property	Description
Repeat Layout	This property state the table tags or flow to use when formatting the list while it is run server. The default RepeatLayout Table.
Repeat Columns	This property state the column number to use when repeating the controls. Default value of RepeatColumns is 0.
Repeat Direction	This property states the direction in which the controls to be repeated. Available values for Repeat Direction are Horizontal and Vertical. Default value of RepeatDirection is Vertical.

4. Radio Button list

- A radio button list shows a list of mutually exclusive options.

syntax

```
<asp:RadioButtonList ID="RadioButtonList1" runat="server" AutoPostBack="True" OnSelectedIndexChanged="RadioButtonList1_SelectedIndexChanged">
</asp:RadioButtonList>
```

- To allow user to select from small set of mutually exclusive predefined options, this control is used.
- These control permits to define any number of radio buttons with labels.

5. Bulleted lists

- It permit defining list of items either by creating static items at design time or by binding control to data source to manipulate at run time.

Syntax

```
<asp:BulletedList ID="BulletedList1" runat="server">
</asp:BulletedList>
```

- If we know at design time the types of items need to display, we can set group of control items to set of individual items in markup.
- If items to be shown are dynamic ,we can create group of items in code at run time.

Common properties of the bulleted list

Property	Description
BulletStyle	To state the style and looks of the bullet list and number of bullets, this property is used.
RepeatColumns	This property state the column number to use when repeating the controls. Default value of RepeatColumns is 0.
RepeatDirection	This property states the direction in which the controls to be repeated. Available values for RepeatDirection are Horizontal and Vertical. Default value of RepeatDirection is Vertical.

Syllabus Topic : Table Controls

2.1.4 Table Control

Q. Explain any four properties of TableControl class. (4 Marks)

- In the .NET Framework, the Table class is used to build an HTML table.
- Table class is included in System.Web.UI.Controls namespace.
- The Table control is used with the TableCell control and the TableRow control to create a table in .NET.
- We can create a Table control at run-time as well as at design-time using the Visual studio.

Following are the ASP.NET Table and Helper control classes

Control	Code	Description
Table	<asp:Table>	Table class used to create an HTML table with the help of TableRow and Table Cell class.
Table Row	<asp:Table Row>	TableRow class used to create a row in the table which can be useful for getting and setting cells value of rows using TableCell control.
Table Cell	<asp:Table Cell>	Table Cell class used to create cell in table.
Table Row Collection	<asp:Table Row Collection>	Table Row Collection class is used to maintains group of table rows by inserting and removing a row from it.
TableCellCollection	<asp:TableCell Collection>	Table Cell Collection class is used to maintain group of table cells by adding a cell to a row and removing a cell from row.
TableHeaderCollection	<asp:Table HeaderCell>	Table Header Collection class is used to create header cells of table.

Properties of Table class

Property	Description
Runat	This property state that the web control is a server control.For this purpose we have to set value of runat to "server".
Back Image Url	This property state URL to an image which is used as background to table.
Rows	This property state group of rows in the table.
Caption	This property is used to set title to table.
Caption Align	This property is used to set alignment of the caption text.
Horizontal Align	This property is used to set alignment of table as in the page.
Cell Padding	This property is used to state the space between the cell walls and contents in table.
Cell Spacing	This property is used to determines distance between cells of tables.
Grid Lines	This property is used to set gridline format in the table.

Program 2.1.1

Write a program to create HTML table using Table class in ASP.NET.

Solution :

Program to create HTML table using table class in ASP.NET

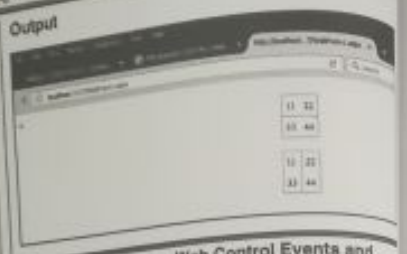
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication1.WebForm1" %>

<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="Form1" runat="server">
<asp:Table ID="Tbl" runat="server" CellPadding="5"
GridLines="Horizontal" HorizontalAlign="Center">

<asp:TableRow>

<asp:TableCell>11</asp:TableCell>
<asp:TableCell>22</asp:TableCell>
</asp:TableRow>
<asp:TableRow>
<asp:TableCell>33</asp:TableCell>
<asp:TableCell>44</asp:TableCell>
</asp:TableRow>
</asp:Table>
<br />
<asp:Table ID="Tbl" runat="server" CellPadding="5"
GridLines="Vertical" HorizontalAlign="Center">
<asp:TableRow>
<asp:TableCell>11</asp:TableCell>
<asp:TableCell>22</asp:TableCell>
</asp:TableRow>
<asp:TableRow>
<asp:TableCell>33</asp:TableCell>
<asp:TableCell>44</asp:TableCell>
</asp:TableRow>
</asp:Table>
</form>

</body>
</html>
```

Create table

Create table row

Create table cells which contains table data

Output



Syllabus Topic : Web Control Events and AutoPostBack

2.1.5 Web Control Events and AutoPostBack

Q. What are the types of web control events? What happen if AutoPostBack is set to "true"? (4 Marks

2.1.5(A) Web Control Events

- Events are occoured due to ASP.NET Web To server controls which work somewhat differently events in HTML pages or events in any W applications which runs at client side.
- The difference is at basic level because of separation of the event itself from where the event handled.
- In client-based applications, events are raised handled at the client side.
- In ASP.NET, events which are related with controls are occour due to client action but these are handled on the Web server by the ASP.NET pag
- For events occoured on the client side(browser ASP.NET Web Forms control event model needed the details of event be captured on the client side as message of event occurrence is transmitted to server using an HTTP post method.

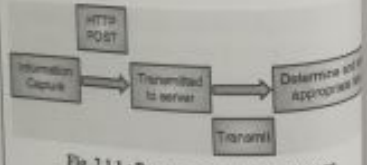


Fig. 2.1.1 : Process of event occourance

- The ASP.NET page must run the post metho determine which event is occurred and then cal correct method on the server to handle the that e
- Task of capturing, transmitting, and running the is handeled by ASP.NET

- ASP.NET web form supports many web controls like Check Box, Button, Radio Button List, Text Box etc.
- Each web control has related events.types of web control events are :

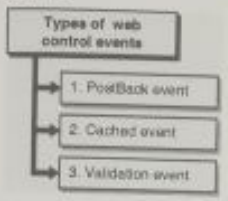


Fig. C2.2 : Types of web control events

➔ **1. PostBack event**

- In PostBack events, we send an ASP.NET page to the server for processing.
- PostBack event is occur when specific credentials of the page are to be cross checked against some sources i.e. verification of username and password using database which cannot done at client machine and thus these information have to be 'posted back' to the server.
- To capture a postback event using web control, the web control must implement the System.Web.UI.I PostBack Event Handler interface.
- Use of this interface permits web control to raise events on the server as response to postback from the client.
- The IPostBackEventHandler interface contains one method Raise Post Back Event() to handle Post Back event.

```
public interface IPostBackEventHandler
{
    void RaisePostBackEvent(string eventArgument);
}
```

➔ **2. Cached event**

- These events store data of page that gets processed when page is submitted to the server by using PostBack event.
- Cached event are placed at view state of page and executed at serverside

☛ **Example**

```
TextChanged event of TextBox
SelectedIndexChanged event of DropDownList
```

➔ **3. Validation event**

- Validation event is handled on the page before the page is posted back to server.
- The sequence of web control events on a Web form is as below.
  1. First validations Event is occurred before the page is send to the server by client for processing.
  2. Postback Event occurs that cause the page to be send to the server.
  3. Then,Page_Init and Page_Load events are handled by server.
  4. Cached events are occourd and handled.
  5. Lastly, the event that caused the postback is processed.

2.1.5(B) AutoPostBack

- AutoPostback and Postback are same.
- These properties are used to submit pages to web server.
- In AutoPostback ,web page is submitted to server, then Server processes the values and server sends response back to same page or redirect that response to different page.
- Every Web controls will have their own AutoPostBack property.
- If we create a web Page which contains one or more Web Controls that are configured to use AutoPostBack property then the ASP.NET adds a special JavaScipt function in HTML Page while running.
- Name of this javascript function is _doPostBack().
- When we Call this function, it sends data to web server for processing i.e postback.
- Two hidden input fields are added by ASP.NET in web form that are used to pass details back to the server.
- These details contains ID of the Control due to which event occurs and any additional details if required.

☛ **Syntax for hidden field**

```
<input type="hidden" name="__EVENTTARGET"
id="__EVENTTARGET" value="" />
```

- It is responsibility of _doPostBack() function to set these values with the appropriate information about the event and submit the form.

- The _doPostBack() function is shown below.

```
<script language="text/javascript">
function __doPostBack(event_Target, event_Argument)
{
    if (Form.onsubmit() || (Form.onsubmit() != false))
    {
        Form.__EVENTTARGET.value = event_Target;
        Form.__EVENTARGUMENT.value =
                                    event_Argument;
        Form.submit();
    }
}
</script>
```

- ASP.NET creates the _doPostBack() function automatically for web control on the web page which uses automatic postbacks.
- For any control if the property AutoPostBack is set to true, then it is possible to connect that control to the _doPostBack() function with the help of onclick and onchange attributes.
- These attributes decides the type of action which should be carried out by the browser response to the Client-Side javascript events- onclick and onchange.
- That means ASP.NET automatically changes a javascript event which raise at client side into a server-side ASP.Net event by using the _doPostBack() function.

Program 2.1.2

Write program in ASP.NET to show ice-cream flavour selected by user using Autopostback

Solution :

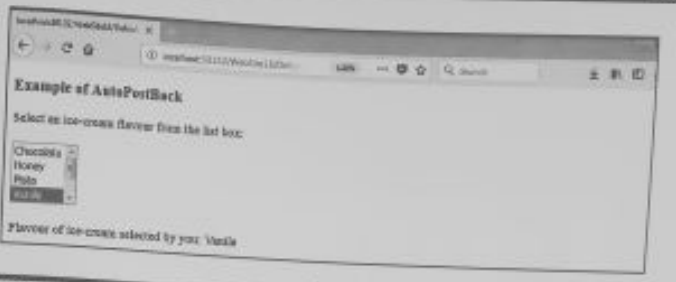
Program to show ice-cream flavour selected by user using Autopostback

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default2.aspx.cs" Inherits="Default2" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <script runat="server">
    void Page_Load(Object sender, EventArgs e)
    {
        if (ListBox1.SelectedItem != null)
        {
            Label1.Text = "Flavour of ice-cream selected by
            you: " + ListBox1.SelectedItem.Value;
        }
        else
            Label1.Text = "";
    }

    </script>
</head>
<body>
    <form id="form1" runat="server">
    <h3> Example of AutoPostBack </h3>

    Select an ice-cream flavour from the list box: <br
    /><br />

    <asp:ListBox id="ListBox1"
    Rows="4"
    AutoPostBack="True"
    SelectionMode="Single"
    runat="server">

    <asp:ListItem>Chocolate</asp:ListItem>
    <asp:ListItem>Honey </asp:ListItem>
    <asp:ListItem>Pista</asp:ListItem>
    <asp:ListItem>Vanila</asp:ListItem>

    </asp:ListBox>

    <br /><br />

    <asp:Label id="Label1"
        runat="server">
    </form>
</body>
</html>
```

Show the flavour selected by user in textbox

Autopostback set to true for submitting web page to server, then Server processes the values and server sends response back to same page

Output



Syllabus Topic : Page Life Cycle

2.1.6 Page Life Cycle

Q. State the page life cycle with diagram. (4 Marks)

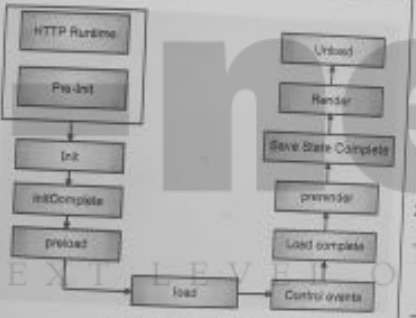


Fig. 2.1.2 : Page life cycle

- When a web page is requested, that request is send to web server and requested web page is loaded into the server memory which is then processed, and send to the browser as response. Then it is unloaded from the memory.
- At each stage, methods and events are available which can be overrided to fulfill the requirement of the application.
- These stages include initialization, instantiating controls, restoring and managing state, running event handler code, and running the page on server.

Steps of ASP.NET page life cycle

1. Pre-Init

- When page lifecycle starts, pre-init event is occoured before next Initialization stage begins.
- This event is used to perform following task :
  1. Check IsPostBack property to state wheather this is a first time page is being processed and determines whether the page is a postback.
  2. To create or recreate dynamic controls.
  3. Set master page or theme dynamically.
  4. To read or assign profile property values.

2. Init

- Init event occours after initialization of all controls is done. Unique id of each control is set and skin setting have been applied.
- Init event of individual control occurs before init event of page.
- This event can be use to read or initialize control properties.
- This event can be handled by overloading the OnInit method or by creating a Page_Init handler.

3. InitComplete

- It occurs when page initialization is done.
- The tracking of view state changes is turned on between Init and InitComplete events.
- View state tracking permit control to store any values that programmatically added to view state collection.
- If tracking of view state is turned off ,any values added to view state are lost during postback
- This event is usually used to make changes to view state that is to be stored during postback event.

4. Preload

- This event is occurred after page load its view state and repetitively load view state for all the controls on the page.
- Before the Page object raise this event, it loads view state for itself and all controls on that page, and then processes any postback data contained by the Request instance.

5. Load

- The Load event is first occours for page and then recursively for all child controls.
- To handle this event OnLoad() method on page object is called and repeat same for all the child controls on that page.
- Load event of page is followed by load event of individual controls.
- This event is useful for setting properties of controls on that page and to connect with database.

6. Control events

- When page is completely loaded and validated ASP.NET fires all the events which are occoured due to last postback.
- ASP.NET events has two types :
  1) **Immediate response event :** This event include clicking on submit button or some other button, image or link due to which post back occurs by calling _doPostBack() javascript function.
  2) **Change event :** This event include changing the selected value in web control i.e. checkbox,radio button or text in textbox. This event fired immediately for web controls if AutoPostBack property is set to true. Otherwise fire next time when page is postback.

7. LoadComlete

- This event is occured when the loading process is completed, event handlers of web controls are run and validation of page is done.
- It is occured at end of event handling stage.
- It is used for task that need all the web controls of page are loaded.

8. PreRender

- This event indicates the last action before output is rendered.
- During prerender stage, page and control instances are available so last-minute action such as storing additional information in view state is done.
- Page object fires PreRender event on page first and then on web controls on that page.
- This event is used to make final changes to contents of page or it's controls before page is rendered.

9. PreRenderComplete

- As the PreRender event is occured repetitively for all controls on the page, this event gurantee the completion of pre-rendering phase.
- This event is occour before DataBind() method for each control is called.

10. SaveStateComplete

- This event is fired when control state and view state information is saved.
- Any change to page or web controls on that page affect rendering of that page but changes are not retrived on next postback.

11. Render

- At the end of life cycle of page it is rendered to HTML.
- At this stage of life cycle, page call this method repetitively for all web controls.
- Each web server control invoke Render() method which create markup of control to be send to browser.

12. Unload

- The UnLoad phase is the last phase of the page life cycle.
- It fires the UnLoad event for all web controls repetitively first and then for the page .
- Final cleanup such as closing file stream,database connection etc. is done and all resources and references are free.
- This event can be handled by using the OnUnLoad() method.
- At this stage, page object is still available but final HTML is already rendered and cannot changed.

Syllabus Topic : State Management

2.2 State Management

Q. Why we need state management concept? List all state management techniques. (4 Marks)

- Web is Stateless that means a new web page object is re-created each time to serve request of client.
- HTTP is a stateless protocol that means it can not maintain the client information entered on page.

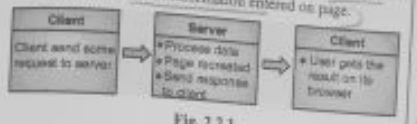


Fig. 2.2.1

- Page is recreated before it comes to clients and it happens for each and every request. So it is a big issue to maintain the state of the page and information for a web application.
- This issue can be solved by using State Management.
- ASP.NET Provides some features like View State, Cookies, Session, Application objects etc. to manage the state of page.

Syllabus Topic : View State

2.2.1 View State

Q. Write short note on view state. (4 Marks)

- View state is one of the most useful client side information management system.
- View state is the mechanism that permits state values of client to be stored during page postbacks.
- Because of the stateless behavior of web pages, values entered by user in web control is not maintained during postbacks.
- When we require values of page variable to be maintained during page postbacks, we can use View state to store those values.
- "EnableViewState" Property is used for both Page Level and Server Control Level to manage the view state.
- Values stored in hidden form fields are send to server to maintain state of client. When we view the page source in browser of a page that uses hidden form field to maintain View state, we see code like this.

```
<input type="hidden" name="viewstate" id="viewstate1"
value="123"/>
```

- This single hidden field contains all the view state values for all the page controls.
- ASP.NET pages provide the View State property as a built-in structure for automatically storing values between multiple requests for the same page.

Program 2.2.1

Write a program using viewstate to maintain and retrieve data entered by user.

Solution : Program using viewstate to maintain and retrieve data entered by user.

Web1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="WebApplication8.WebForm1" %>
```

Create form with text box and two buttons

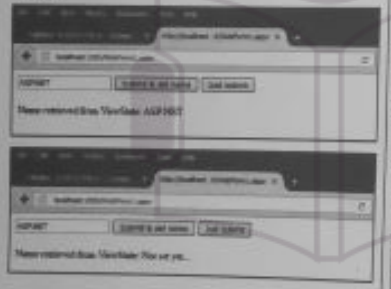
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <asp:TextBox runat="server" id="NameField1" />
    <asp:Button
    runat="server" id="SubmitForm" onclick="SubmitForm_Click" text="Submit & set name" />
    <asp:Button runat="server" id="RefreshPage" text="Just submit" />
    <br /><br />
        Name retrieved from ViewState: <asp:Label
    runat="server" id="NameLabel1" />
    </form>
</body>
</html>
```

Web1.aspx.cs (code behind)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication8
{
Public partial class WebForm1 : System.Web.UI.Page
    {
Protected void Page_Load(object sender, EventArgs e)
```

```
{


if(ViewState["NameOfUser"] != null)
        NameLabel1.Text =
ViewState["NameOfUser"].ToString();
else
        NameLabel1.Text = "Not set yet...";
    }


protectedvoid SubmitForm_Click(object sender, EventArgs
e)
    {
ViewState["NameOfUser"] = NameField1.Text;
        NameLabel1.Text = NameField1.Text;
    }
  }
}
```

Output



Here enter string in the textbox and press the first button "Submit & set name". The name will be saved in the ViewState and set to the label which contains the entered string. Now press the second button. This button just posts back data to the server.

Limitation of view state

1. Viewstate can be used only with single page.
2. Because it stores information as hidden field, it can be seen in source code in browser, hence it is not secure way.

Syllabus Topic : Cross-Page Posting

2.2.2 Cross-Page Posting

Q. What is cross-page posting ? (4 Marks)

- When there is a need to transfer some data from one web page to another web page then usually we use session to maintain state.
- But the use of a session can not be good every time because the page becomes heavy due to session tracking.
- There is other way to maintain state of client and using which we can avoid use sessions, we can use a cross page postback. It simply transfers the data from one page to another.
- ASP.NET by default submit the forms to the same pages.
- Cross page posting submits the form to a different page.
- This is usually needed when we are generating a multipage form to group information from the user on every page, when moving from the source to the target page.

Program 2.2.2

Write a program using cross-page posting to maintain and retrieve data entered by user.

Solution :

Program using cross-page pasting to maintain and retrieve data entered by user

WebForm1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication9.WebForm1" %>


 <html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
<table>
<tr>
<td><b>Enter UserName:</b></td>
```



```
<td><asp:TextBox ID="username"
runat="server"/></td>
</tr>
<tr>
<td><b>Enter city:</b></td>
<td><asp:TextBox ID="city" runat="server"/></td>
</tr>
<tr>
<td></td>
<td><asp:Button ID="btnPostback" Text="Postback"
runat="server"
PostBackUrl="~/WebForm2.aspx" /></td>
</tr>
</table>
</div>
</form>
</body>
</html>
```

Information entered in webForm1.aspx is send to WebForm2.aspx

WebForm2.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm2.aspx.cs"
Inherits="WebApplication9.WebForm2" %>
 <html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
<b><u>WebForm2.aspx Page</u>
</b><br /><br />
<label id="label1" runat="server" />
<br /><br />
<label id="label2" runat="server" />
</div>
</form>
</body>
</html>
```

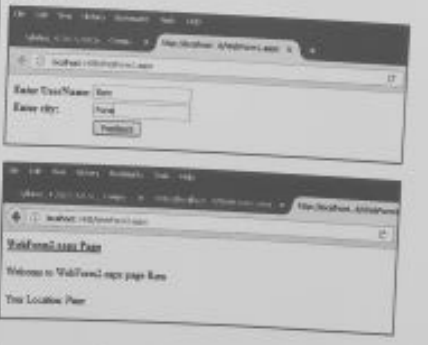
Label are used to show data enter by user at page WebForm1.aspx

Web2.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;


namespace WebApplication9
{
public partial class WebForm2 : System.Web.UI.Page
  {
protected void Page_Load(object sender, EventArgs e)
    {
if (PreviousPage != null &&
PreviousPage.IsCrossPagePostBack)
        {
TextBox name =
(TextBox)PreviousPage.FindControl("username");
            TextBox txtcity =
(TextBox)PreviousPage.FindControl("city");
            label1.InnerText = "Welcome to
WebForm2.aspx page " + name.Text;
            label2.InnerText = "Your Location: " +
txtcity.Text;
        }
else
        {
Response.Redirect("WebForm1.aspx");
        }
    }
  }
}
```

This label shows name and city data enter at web form 1.aspx

Output



Syllabus Topic : Query String

2.2.3 Query String

Q. What is query string ? (Ref Sec. 2.2.3) (2 Marks)

- Query string is the mechanism which used to send data from one web form to another web form using URL.
- Query string is made up of two parts; field and value and each pair is separated using ampersand (&).
- ?(Question Mark) is used at starting of a query string and it's value.
- There is a limitation to length of query string. So query strings cannot be useful to send very large data.
- Query strings are visible to the user, so it should not be used to send sensitive information such as username, Password without encryption.
- Request object of QueryString property is used to retrieve the query string.

Program 2.2.3

Write a program using query string to maintain and retrieve data entered by user.

Solution :

Program using query string to maintain and retrieve data entered by user

WebForm3.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm3.aspx.cs"
Inherits="WebApplication9.WebForm3" %>

<!DOCTYPE html>
<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div><b>QueryString Example</b></div><br />
<table>
<tr>
<td> <b>Enter UserId :</b> </td>
<td> <asp:TextBox ID="userid" runat
="server"> </td>
</tr>
<tr>
<td> <b>Enter UserName :</b> </td>
<td> <asp:TextBox ID="username"
runat="server"></td>
</tr>
<tr>
<td></td>
<td> <asp:Button ID="buttonSend" Text="Send Values"
runat="server"
onclick="btnSend_Click"> </td>
</tr>
</table>
</div>
</form>
</body>
</html>
```

Enter username and password data

WebForm3.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication9
{
public partial class WebForm3 : System.Web.UI.Page
    {
protected void Page_Load(object sender, EventArgs e)
        {

        }

protected void btnSend_Click(object sender, EventArgs e)
        {
Response.Redirect("WebForm4.aspx?UserId=" +
userid.Text +
"&UserName=" + username.Text);
        }
    }
}
```

Send data to next page WebForm4.aspx

WebForm4.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm4.aspx.cs"
Inherits="WebApplication9.WebForm4" %>

<!DOCTYPE html>
```



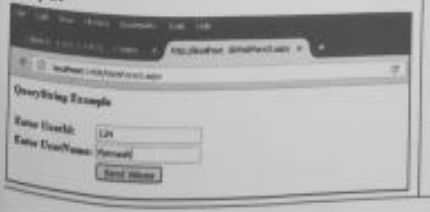
```
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div><b>QueryString parameter Values in
WebForm4.aspx Page</b></div><br />
<div><b>UserId:</b><asp:Label ID="labeluserid"
runat="server"/></div><br />
<div><b>UserName:</b><asp:Label
ID="labelusername" runat="server"/></div>
</form>
</body>
</html>
```

WebForm4.aspx.cs

```
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication9
{
public partial class WebForm4 : System.Web.UI.Page
    {
protected void Page_Load(object sender, EventArgs e)
        {
if (!IsPostBack)
            {
                labeluserid.Text =
Request.QueryString["userid"];
                labelusername.Text =
Request.QueryString["username"];
            }
        }
    }
```

Display userid and username entered by user

Output



Syllabus Topic : Cookies

2.2.4 Cookies

Q. Explain cookies in details. (4 Marks)

- Cookie is a small piece of information stored on the client machine which is used to identify a user.
- It is used to store private information of user such as Username, Password, Address, Contact no. etc on client machines.
- Cookies are used for authentication of a user, store private information of user and shopping cart contents, or anything else that can be accomplished through storing text data.
- Cookies can also be used for transformation of data from one page to another page.
- For example, if a user requests a page from a web site, then the web application sends not just a page,but also cookie which includes the date and time of the page. The browser also gets the cookie, which is stored in a folder on the hard disk of client machine.
- After that, if user requests a page from the site again by entering the URL of web page in the browser, the browser see in the local hard disk for a cookie associated with that URL. If the cookie exists for that URL, the browser sends the cookie to the site with the page request. The application can then state the date and time on which user lastly visit our web site. We can use the information in the cookie to display a message to the user or check an expiration date.
- Cookies are not associated with a Web page, it is associated with web site. So the server and browser will exchange information stored in cookie without limitation of what page the user wants from our site. As the user visits many web sites, each web site may send a cookie to the browser of user. All the cookies separately stored by the browser.
- To use cookie concept, we need to import namespace called Systen.Web.HttpCookie.

Type of Cookies

There are two types of cookie :

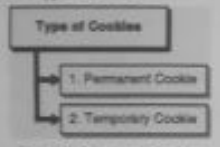


Fig. C2.3 : Type of Cookies

1. Permanent Cookie

- A cookie which does not have expiration time is called as Permanent Cookie.
- Permant cookie are stored as file on hard disk of client machine. These cookies are stored in "C:\Documents and Settings\username\Cookies" folder. Permanent cookie is created by setting "Expires" property.

2. TemporaryCookie

- A cookie which have expiration time is called as Temporary Cookie.
- Temporary Cookie is stored in RAM of client machine.
- It is transferred with request and response. Internally all cookies are temporary.
- It is easy to create a cookie in the ASP.NET with help of Response object of HTTPResponse class or HTTPCookie class.
- Each cookie have unique name so it can be identified later using that name.

Example

```
HttpCookie userInformation
= new HttpCookie("userInformation");
userInformation["username"] = "Ram";
userInformation["userColor"] = "White";
userInformation.Expires.Add(new TimeSpan(0, 1, 0));
Response.Cookies.Add(userInformation);
```

- It is easy to read cookie value form cookies by help of Request object of HTTPRequest class.

Example

```
string user_name = string.Empty;
string user_color = string.Empty;
user_name = Request.Cookies["username"].Value;
user_color = Request.Cookies["userColor"].Value;
```

Properties of cookies are:

1. **Domain** : Domain property is used to define association between cookies to domain.
2. **Secure** : We can set this property to true to permit creation of secure cookie.
3. **Value** : We can use this property to manipulate individual cookie.
4. **Values** : We can use this property to manipulate cookies with key/value pair.
5. **Expires** : This property is used to set expire date for the cookies.

Advantages of Cookie

1. Cookies contain text data only, hence user can easily read data is cookie .
2. We can store secure information of user on the client machine.
3. We can easily maintain cookies.
4. We can instantly acess the cookies.
5. Cookie can work transparently without user being aware that information needs to be stored.

Disadvantages of Cookie

1. If user deletes cookie information, we can not get that information back. There is no backup facility for cookie.
2. Each request of client will have associated cookie information.

Limitation of cookie

1. Cookies is small text file so it can store small amount of data approximately 4096 bytes of data.
2. It is not secure as it is stored on client side and are temporary.
3. In some cases browser does not support cookie.

- We can clear information in cookies in following way.
- We can delete cookie information from client machine from the cookie folder by setting Expires property.

```
userInformation.Expires = DateTime.Now.AddHours(1);
```

- Cookie information will deleted after the one hour of cookie creation.

Program 2.2.4

Write a program to use dropdownlist to show different color options such as pink ,red etc., maintain that selected value of color and set it as background color using cookies.

Solution :

A program to use dropdownlist to show different color options such as pink ,red etc., maintain that selected value of color and set it as background color using cookies

WebForm6.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm6.aspx.cs"
Inherits="WebApplication9.WebForm6" %>

<html>
<head runat="server">
<title></title>
</head>
<body runat="server" id="BodyTag">
<form id="form1" runat="server">

<asp:DropDownList runat="server" id="ColorSelector"
autopostback="true"
onselectedindexchanged="ColorSelector_IndexChanged">
<asp:ListItem value="White">Select
color...</asp:ListItem>
<asp:ListItem value="Pink">Pink</asp:ListItem>
<asp:ListItem value="Green">Green</asp:ListItem>
<asp:ListItem value="Yellow">Yellow</asp:ListItem>
</asp:DropDownList>
</form>
</body>

</html>
```

Create dropdown list

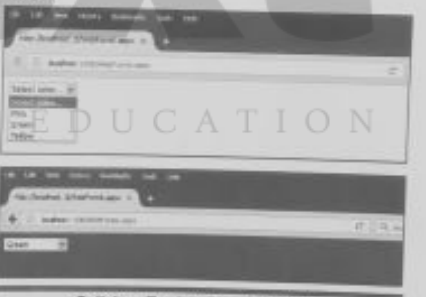
WebForm6.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication9
{
public partial class WebForm6 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void ColorSelector_IndexChanged(object sender,
    EventArgs e)
    {
    BodyTag.Style["background-color"] =
    ColorSelector.SelectedValue;

    HttpCookie cookie = new HttpCookie("BackgroundColor");
        cookie.Value = ColorSelector.SelectedValue;
        cookie.Expires = DateTime.Now.AddHours(1);
    Response.SetCookie(cookie);
    }
  }
}
```

Set selectedcolor as background color

Create cookie using HttpCookie class and set value and expiration time of Cookie. Cookie will expire after one hour

Output



Syllabus Topic : Session State

2.2.5 Session State

Q. Explain session in detail. (4 Marks)

- A session is the timespan in which a user communicate with a Web application.
- ASP.NET session state permit us to store and retrieve values regarding user as the user visits different ASP.NET pages in a Web application.

- HTTP is a stateless protocol that means Web server treats each HTTP request for a page as a different request.
- The server does not maintain knowledge of variable values that were used during previous requests.
- ASP.NET session state identifies requests from the same browser during a limited time as a session, and provides a way to persist variable values for the duration of that session.
- By default, ASP.NET session state is enabled for all ASP.NET applications.
- ASP.NET manage session state by assigning unique Id to the user when the session is started.
- Sessions can be identified easily using unique identifier that can be read by using the SessionID property.
- When session state is enabled for web application, every request for web page in that application is analyzed for a SessionID value sent from the browser.
- If there is no SessionID value exists then ASP.NET starts a new session and the value of SessionID for that session is sent to the browser with the response.
- By default,cookie contains value of SessionID.
- We can also configure the settings of application to store SessionID values in the URL for a "cookieless" session.
- A session is supposed as active till requests are continually made through browser using same value of SessionID.
- If the time between two requests for a specific session cross the particular time-out value in minutes, the session is supposed to be expired.
- Requests made with an SessionID value which is expired can create new session.
- The SessionID is stored in an HTTP cookie and is send by client to the server on each request. The server can then read the SessionID from the cookie and restart the server session state.
- Config.web file which is ASP.NET XML configuration file is used to configure ASP.NET setting. Configuration files have two types: a machine configuration file and an application configuration file.
- Both machine configuration file and an application configuration file has name config.web. These two configuration files are same, except that the machine configuration file applies configuration settings to all applications but the application configuration files applies configuration settings to specific application.

Program 2.2.5

Write a program to accept username and password from user and maintain that data on next page using session state.

Solution :

A program to accept username and password from user and maintain that data on next page using session state.

webForm7.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm7.aspx.cs"
Inherits="WebApplication9.WebForm7" %>

<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
    User Name:-<asp:TextBox ID="txtusername"
runat="server"></asp:TextBox>
<br />
<br />
Password:-<asp:TextBox ID="txtpwd"
runat="server"></asp:TextBox>
<br />
<asp:Button ID="Button1" runat="server"
OnClick="Button1_Click" Text="Submit" />
</div>
</form>
</body>
</html>
```

Take user name and password from user

WebForm7.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication9
{
```



```
public partial class WebForm7 : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
    {

    }

protected void Button1_Click(object sender, EventArgs e)
    {
        //textbox value is stored in Session
Session["UserName"] = txtusername.Text;
Session["Pwd"] = txtpwd.Text;

Response.Redirect("WebForm8.aspx");
    }
}
```

Conduct the session and preserve the username and password value

WebForm8.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm8.aspx.cs"
Inherits="WebApplication9.WebForm8" %>

<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
    User Name:-<asp:TextBox ID="txtUserName1"
runat="server"></asp:TextBox>
<br />
<br />
Password:-<asp:TextBox ID="txtpwd1" runat="server"
ontextchanged="txtpwd_TextChanged"></asp:TextBox>
<br />
</div>
</form>
</body>
</html>
```

WebForm8.aspx.cs

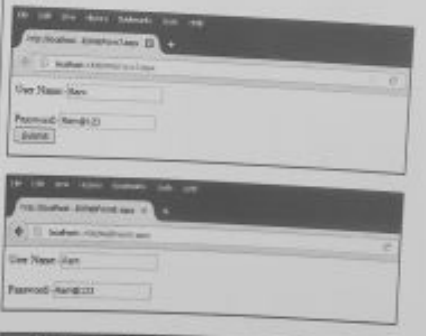
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

```
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication9
{
public partial class WebForm8 : System.Web.UI.Page
    {
protected void Page_Load(object sender, EventArgs e)
    {
        //Session value is assign on the text box
if (Session["UserName"] != null)
    {
        txtUserName1.Text =
Session["UserName"].ToString();
    }
if (Session["Pwd"] != null)
    {
        txtpwd1.Text = Session["Pwd"].ToString();
    }
}
protected void txtpwd_TextChanged(object sender,
EventArgs e)
    {
    }
}
```

Display value username preserve in session

Display value password preserve in session

Output



Syllabus Topic : Configuring Session State

2.2.6 Configuring Session State

Q. Which attributes are used to configure a session state? (4 Marks)

- Following is config.web file which is used to configure the session state settings for our ASP.NET application.

```
<configuration>
<sessionstate
mode="inproc"
cookieless="false"
timeout="25"
sqlconnectionstring="data source=127.0.0.1;user
id=<user_id>;password=<pwd>"
server="127.0.0.1"
port="6534" />
</configuration>
```

Following are attributes in <configuration> tag

i. **Mode** :There are three choices for mode: inproc, sqlserver, and stateserver.

ASP.NET supports two modes : Out of process and in process. There are also two options for out-of-process state management : memory based i.e. state server and SQL Server based i.e. sqlserver.

ii. **Cookieless** : This option has Boolean value true or false.

iii. **Timeout** : This option is used to set the amount of time of session for which session is valid. On each request the timeout value of session is set as the current time and the timeout value.

iv. **Sqlconnectionstring** : The sqlconnectionstring is used to identify the each database individually by assigning name to database.

v. **Server** : In the out-of-process mode stateserver, this attribute is used to assign name to the server that is running the ASP.NET program.

vi. **Port** : This port attribute is used to set port number of application.

Example

```
<Script runat=server>
Public void Session_Add(sender As Object, e As
EventArgs)
{
Session("SessionStateDemo") = text1.Value;
span1.InnerHtml = " data of session is updated ! <P>
our session contains: <font color=blue>" +
Session("SessionStateDemo ").ToString() + "</font>"
}
Public void CheckSession(sender As Object, e As
EventArgs)
{
If (Session("SessionStateDemo ") = Isnull)
{
span1.InnerHtml = "There is no information in session!";
}
Else
{
span1.InnerHtml = "our session contains:
<font color=green>" +
Session("SessionStateDemo ").ToString() + "</font>"
}
}
</Script>
<form runat=server ID="Form12">
<input id=text2
type=text runat=server NAME="text2">
<input type=submit runat=server
OnServerClick="Session_Add()" Value="Add_to_Session
State" ID="Submit2" NAME="Submit2">
<input type=submit runat=server
OnServerClick="CheckSession" Value="View Session
State" ID="Submit3" NAME="Submit3">
</form>
<h1 size=1>
<font size=6><span id=span1 runat=server/></font>
```

- This simple page raise two server-side events ,one is the Add and second for View buttons, and also set the session state to the value entered in the text box.

There are four configuration settings

i. in-process mode ii. out-of-process mode
iii. SQL Server mode iv. State Server Mode

i. In-process Mode

- ASP.NET session state can be used in a same way as of classic ASP session state. This is called In-process mode.
- That means session state is maintained in process and if the process is re-cycled then state of process is lost.
- The performance of session state will be much fast. For example the time it takes to read from and write to the session state dictionary will be less.
- In-process mode is the default mode for ASP.NET web page.
- When In-process mode is set, the only other session config.web settings used are cookieless and timeout.

ii. Out-of-process Mode

- **Out-of process** mode is included with .NET SDK as a Windows NT service.
- The Windows service is used by ASP.NET for out-of process session state management.

We need Out-of-Process Session States because :

1. Default sessions are stored in-process mode. So when the process is failed i.e server go out of order or recycled, the session state is lost. It is happened even if the browser has the Session Key stored with it.
2. In-process mode is not good for web farms. If the application is deployed to the Web Farms with many machines and each request can be served by different machines, then an in-process session state could not track the user.

ii. SQL server mode

- SQLServer mode is used to store details of session state in a SQL Server database.
- Using this mode, we can ensure that session state is maintained even when the Web application is restarted.
- For using SQLServer mode, we need confirmation about properly installation of ASP.NET session state database on SQL Server.
- To configure an ASP.NET application to make use of SQLServer mode,we need to do following changes in Web.config file of our application :
  - Set value of mode attribute of the sessionState to SQLServer.
  - Set the sqlConnectionString attribute to a connection string for our SQL Server database.

```
<configuration>
<system.web>
<sessionState mode="SQLServer"
sqlConnectionString="Integrated
Security=SSPI;data
source=SampleSqlServer;" />
</system.web>
</configuration>
```

iv. State Server Mode

- StateServer mode is used to store information about session state in a process.
- To use StateServer mode, we should sure about ASP.NET state service which is running on the server used for the session store.
- After installing ASP.NET and AP.NET framework, the ASP.NET state service is installed as a service.
- The ASP.Net state service is installed at the below location :

```
systemroot\Microsoft.NET\Framework\versionNumber\aspn
et_state.exe
```

- To use StateServer mode, we need to do following changes in Web.config file of our application :
  - Set the mode attribute of the sessionState element to StateServer.
  - Set the stateConnectionString attribute to tcpip=serverName:42424.
- Below example display changes in web.config file where session state is stored on a computer whose name is DemoStateServer :

```
<configuration>
<system.web>
<sessionState mode="StateServer"
stateConnectionString="tcpip=DemoStateServer:42424"
cookieless="false"
timeout="20" />
</system.web>
</configuration>
```

Syllabus Topic : Application State

2.2.7 Application State

Q. Explain the application state in detail ? (4 Marks)

- Application State is one of the state management mechanism.
- Application state is a data repository that is available to all classes in an ASP.NET application.
- Application State is stored in the memory of the the web server which is user specific.
- Retrieving application state from server is faster than storing and retrieving information in a database.
- Session state is maintained for a single user session, but Application State is applicable for all users and sessions.
- There is no a default expiration period for applications like cookie and session.
- When we close the working process then application object will be expired.
- The data is shared between different users by a HTTPApplicationState class and the data can be stored here in form of a key/value pair.
- This key/value pair data can also be accessed using the application property of the HTTPContext class.
- Application state is a useful for storing small amounts of data that does not change from one user to another user.
- HttpApplicationState class is used to store Application state.

Application State Life Cycle

Step 1 : When web server receives request send by browser , it first verify the extension to state whether or not it is ISAPI. (ISAPI extensions are true applications that run on IIS and have access to all of the functionality provided by IIS.)

Because this request can only be handled by the ISAPI extension. If the extension is different then the request is handled by the server itself.

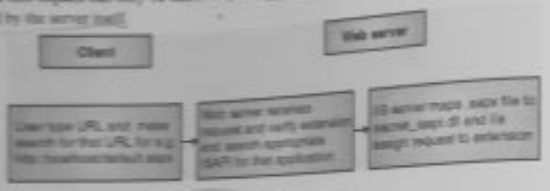


Fig. 2.1.2

Step 2 : After receiving the request, the application domain is created by application manager. In the application domain a instance of the class HostingEnvironment is created that facilates access to information about all application domains.

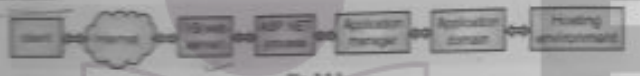


Fig. 2.1.3

Step 3 : After creating the application domain, ASP.NET initializes the basic objects as HTTPContext, HTTPRequest and HTTPResponse. HTTPContext holds objects to the specific application request. HTTPRequest contains all the information regarding the current request like cookies, browser information and so on and the HTTPResponse contains the response that is sent to the client.

Step 4 : Here all the basic objects are being initialized and the application is being started with the creation of the HTTPApplication class.

Step 5 : Then events are started by the HTTPApplication class.

Global.asax file

The Global.asax file is used for handling application events on methods. It resides at root level. Events are one of the following type in the Global application.

1. Events that will be raised on a certain condition.
2. Events that will be raised on every request.

The events of the Global.asax file are

1. **Application_Start()** : This method is used to call initially when first application domain is created.
2. **Session_Start()** : This method is called each time when a session is start.
3. **Application_BeginRequest()** : When an application has started, this method is executed for each user.
4. **Application_AuthenticateRequest()** : This method is used to checks whether the user is valid user or not.
5. **Application_Error()** : This event is raised in response to occurrence of unhandled exception.
6. **Session_End()** : When a user session is ended or all the data associated with that user is deleted then the Session_End() event is called.
7. **Application_End()** : This method is called before the application ends.
8. **Application_Disposed()** : This event is called after the application will be closed.

Program 2.2.6

Example of ASP.NET Application State.

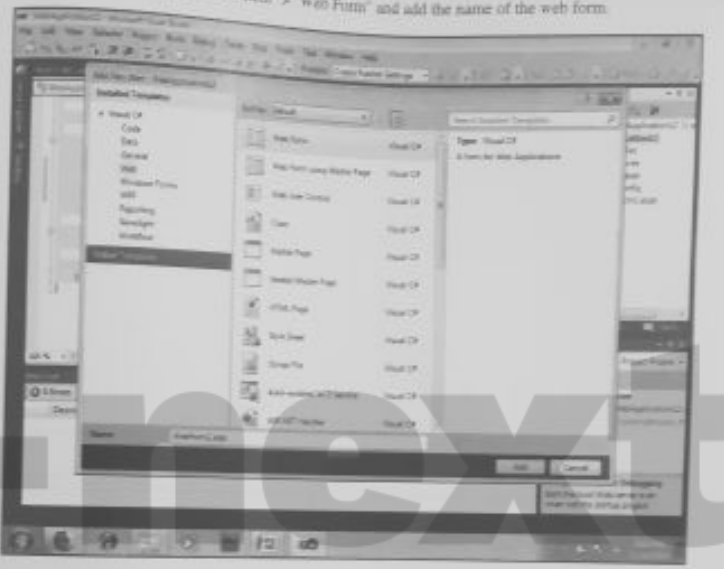
Solution : ASP.NET Application State

Step 1 : Open Visual Studio 2010.

Step 2 : click on "New Project" > "Web" > "ASP.NET Empty Web Application".

Step 3 : Now click on Solution Explorer.

Step 4 : Now right-click on "Add" > "New Item" > "Web Form" and add the name of the web form.



Step 5 : Create the Global.asax file. Again go to Solution Explorer and "Add" > "New Item" > "Global Application Class".code in Global.asax

Step 6 : For configure the session we need to use the web.config file as in the following :

```
<sessionState mode="InProc" timeout="20"
cookieless="true">
</sessionState>
```

Step 7 : For counting the number of online users we need to write following code in the Global.asax.cs file as in the following :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.SessionState;
namespace WebApplication12
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender,
EventArgs e)
        {
            Application["user"] = 0;
        }
        protected void Session_Start(object sender, EventArgs e)
        {
            Application.Lock();
            Application["user"] = (int)Application["user"] + 1;
            Application.UnLock();
        }
```

This event is raised when application starts and it maintain server memory until worker process is restart

when session started application variable is increased by 1

```
protected void Application_BeginRequest(object
sender, EventArgs e)
{
}

protected void Application_AuthenticateRequest(object
sender, EventArgs e)
{
}

protected void Application_Error(object sender,
EventArgs e)
{
}

protected void Session_End(object sender, EventArgs e)
{
    Application.Lock();
    Application["user"] = (int)Application["user"] - 1;
    Application.UnLock();
}

protected void Application_End(object sender,
        EventArgs e)
{
}
```

when session ended, application variable is decrease by 1

Step 8 : Code behind the web form :

```
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication12
{
  public partial class WebForm1 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {
      Response.Write("The number of online users :" +
Application["user"].ToString());
    }
  }
}
```

Output

Syllabus Topic : Validation and Validation Controls

2.3 Validation

Validation stands for checking data for as per the application requirements. ASP.NET provides various types of Validations controls to validate the data.

2.3.1 Validation Controls

Q. What is use of validation controls? List the different validation controls. (4 Marks)

- Validation controls in ASP.NET validate the user input data to assure that unauthenticated data should not get stored in database.
- Validation controls are used to :
  - i) Apply presentation logic.
  - ii) Verify data entered by user.
  - iii) Validate Data format, data type and data range of data entered by user

Validation control has two types

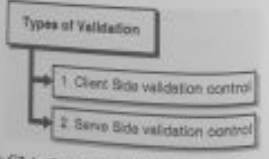


Fig C2.4 : Types of Validation Control

➜ 1. Client Side validation control

- Client side validation is suitable to users as they get the feedback immediately.
- The core advantage is that it stops the process of postback the page to server until the client validation is executed successfully.

- Client side validation is good but in it we have to be dependent on browser and scripting language support.
- For client script .NET uses JavaScript. WebUI Validation.js javascript file is used by .NET for client validation.

➜ 2. Serve Side validation control

According to developer point of view, serve side validation controls are suitable because server control will not fail and not dependent on browser and scripting language.

- We can use ASP.NET validation controls which will assure the client side as well as server validation.
- ASP.NET validation controls work at both the ends. First it will work on client validation and than on server validation.
- ASP.NET provides a set of validation controls which are user friendly and they provide powerful way to check for errors and if needed show error messages to the user.
- ASP.NET provides the following validation controls :
  1. Required Field Validator
  2. Range Validator
  3. Compare Validator
  4. Regular ExpressionValidator
  5. CustomValidator
  6. Validation Summary

Important points for validation controls

1. Control to Validate property is compulsory for all validation controls.
2. One validation control can be used to validate only one input control but multiple validation controls can be assigned to one input control.

Validation Properties

1. Validation of data in web form is done in response to user actions such click on submit button or entering data in textbox.
2. Server validation will only done when value of CauseValidation property is set to true.
3. When the value of the CausesValidation property is set to true ,we can also use the ValidationGroup property to state the name of the validation group for which the Button control causes validation.
4. Page has a Validate() method executes each validation control,if its value set to true.
5. Set the value of CauseValidation property to true for submit button as following :

```
<asp:Button ID="Button2" runat="server" Text="Submit"
CausesValidation=true />
```

Syllabus Topic : Server-Side Validation

2.3.2 Server-Side Validation

Q. Write note on server side validation. (4 Marks)

- This validation is executed after client-side validation is done, if we have client-side validation.
- Sever-side validation is compussary because a user or hacker can send the data through different channels also.
- Server-side validation is done after the user submits the data to server or data posts back to server.
- In the Server Side Validation, one of server side scripting languages such as ASP.Net, PHP etc is used to validate the data submitted by user to server.
- After finishing validation process on the Server Side, the feedback is sent to the client by server using a new dynamically created web page.
- It is better to validate user input on Server Side because we need to protect the data from unauthorized users or hackers.
- Server side validation is more Secure than client-side validation.

Syllabus Topic : Client-Side Validation

2.3.3 Client-Side Validation

Q. Explain client side validation in details. (4 Marks)

- In the Client Side Validation, we can give a better user experience by sending response quickly to browser.
- When we do Client Side Validation, all the user input data is validated in the browser of user.
- Client Side validation does not need a round trip to the server so it decrease the network traffic at server which will help the server to give better performance.
- Client side validation is done on the browser side using client-side scripting languages like JavaScript, VBScript etc.
- For example, if the user enter date in invalid format , our web application can show an error message instantly before the user move to the next field, so the user can correct every field before they submit the form.

- The Client Side Validation depends on the JavaScript Language. So if any user turn JavaScript off in setting of browser then the invalid data can easily bypass and submit to the server.
- So the Client Side Validation can not fully protect our web application from attacks on our server resources and databases.

Syllabus Topic : HTML5 Validation

2.3.4 HTML5 Validation

Q. List all HTML5 validation controls and explain any 2 HTML validation controls in details. **(4 Marks)**

Following are the HTML5 Form Validation controls in ASP.NET :

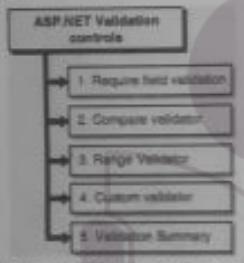


Fig. C2.5 : ASP.NET validation controls

1. Require field validation

- The Require field validation is used to verify user enter data in form filed such as textbox,checkbox , dropdownlist etc.
- Html5 attribute "required" is used to do Required Field validation.
- It is generally tied to a textbox to force input in it.

Syntax

```
<asp:RequiredFieldValidator ID="rfvemailid"
runat="server"ControlToValidate="emailid"
ErrorMessage="Please enter proper Emailid"
InitialValue=" Please enter proper Emailid ">
</asp:RequiredFieldValidator>
```

Program 2.3.1

Write a program to check whether name, email and age fields are filled by user or not. If not then display proper error message using Require field validation.

Solution :

A program to check whether name, email and age fields are filled by user or not. If not then display proper error message using Require field validation.

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm6.aspx.cs"
Inherits="WebApplication7.WebForm6" %>
<html>
<head runat="server">
<title></title>
</head>
<body style = "font-family:Arial;font-size:10pt">
<form id="form1" runat="server">
Name: (Required)<br />
<asp:TextBox ID="txtName"
runat="server"></asp:TextBox>
<asp:RequiredFieldValidator
ID="RequiredFieldValidator1" CssClass
="Validators"
Display="Dynamic" ControlToValidate="txtName"
runat="server"
ErrorMessage="Name is
required."></asp:RequiredFieldValidator>
<br />
Email: (Required Email Address)<br />
<asp:TextBox ID="txtEmail"
runat="server"></asp:TextBox>
<asp:RequiredFieldValidator
ID="RequiredFieldValidator2"CssClass="Validators"
Display="Dynamic" ControlToValidate="txtEmail"
runat="server"
ErrorMessage="Email is
required."></asp:RequiredFieldValidator><br />
Age: (Required and Range 18 - 45)<br />
<asp:TextBox ID="txtAge"
runat="server"></asp:TextBox>
<asp:RequiredFieldValidator
ID="RequiredFieldValidator3" CssClass="Validators"
Display="Dynamic" ControlToValidate="txtAge"
runat="server"
```

Entering value in Name field is compulsory, if not enter then show message "Name is required".

User must enter value in Email field. If user does not enter value, error message shows as "Email is required"


```
ErrorMessage="Age is
required."></asp:RequiredFieldValidator>
<asp:RangeValidator ID="RangeValidator1"
CssClass="Validators" Display="Dynamic"
MinimumValue="18"MaximumValue="45"
Type="Integer" ControlToValidate="txtAge"
runat="server" ErrorMessage="Age Range 18 to
45"></asp:RangeValidator>
<br />
<br />
<asp:Button ID="Button1" runat="server" Text=
"Submit" />
</form>
</body>
</html>
```

User must enter value in age field and it must be in range 18-45 in age field.

Output

2. Compare Validator

- CompareValidator control compares value in one control with value in another control.
- For Example we can use Comapare validator to reconfirm the password provided in one textbox by user with value in another textbox.

Program 2.3.2

Write a program using compare validator to check value entered by user in password and confirm password field is same or not.

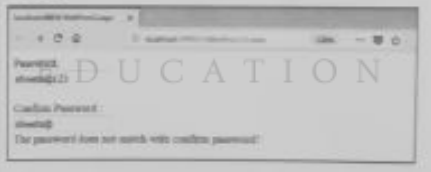
Solution :

A program using compare validator to check value entered by user in password and confirm password field is same or not

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm3.aspx.cs"
Inherits="WebApplication7.WebForm3" %>
<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
Password:<br />
<asp:TextBox runat="server" id="txtpwd" /><br />
<br />
Confirm Password :<br />
<asp:TextBox runat="server" id="txtconfirmpwd" />
<br />
<asp:CompareValidator runat="server" id="cmppwd"
controltovalidate="txtpwd"
controltocompare="txtconfirmpwd"
errormessage="The password does not match with
confirm password"
Type="String" />
<br />
</div>
</form>
</body>
</html>
```

Compare value in password field with confirm password field if values are not matched then display error message.

Output



3. Range Validator

- The RangeValidator control verifies that the input value should in between the range which is specified.

Range validator has three specific properties

Properties	Description
Type	This property state the type of the data. The available values for this property are:String,Integer, Currency, Date, Double.
MinimumValue	This property state lower limit of range.
MaximumValue	This property state upper limit of range.

Syntax

```
<asp:RangeValidator ID="rangevalidationclass"
runat="server" ControlToValidate="classroom"
  ErrorMessage="Enter your class (1 - 10)"
MaximumValue="10"
  MinimumValue="1" Type="Integer">

</asp:RangeValidator>
```

Program 2.3.3

Write a program to check age entered by user is between 18 and 100 using RangeValidator, if not then give appropriate error message.

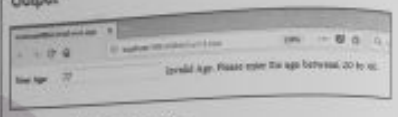
Solution :

A program to check age entered by user is between 18 and 100 using RangeValidator, if not then give appropriate error message.

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication7.WebForm1" %>

<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>

<asp:Label ID="Label2" runat="server"
Font-Bold="True" Font-Size="Small"
      Text="Your Age:"></asp:Label>

    

<asp:TextBox ID="TextBox1" runat="server"
Width="170px"></asp:TextBox>

<asp:RangeValidator ID="RangeValidator1"
runat="server"

      ControlToValidate="TextBox1"

      ErrorMessage="Invalid Age. Please enter the age
between 18 to 100."
      MaximumValue="100" MinimumValue="18"
Type="Integer"></asp:RangeValidator>

 <br />
</div>
</form>
</body>
</html>
```

Create textbox which accept age in range of 18 and 100

Output



4. Custom Validator

- For writing application specific custom validation routines for both the client side and the server side validations, CustomValidator control is used.
- The ClientValidationFunction property is used to perform client side validation.
- The client side validation routine should be written in a scripting language,like JavaScript or VBScript, which can be understood by the browser.
- ServerValidate control event handler is used to call the server side validation routine. The server side validation routine should be written in any .Net language, like C# or VB.Net.

Syntax

```
<asp:CustomValidator ID="CustomValidation1" runat
="server"
ClientValidationFunction="ms_func.ErrorMessage
="CustomValidator">

</asp:CustomValidator>
```

Program 2.3.4

Write a program to check Id enter by user is number or not using custom validator.

Solution :

Program to check id enter by user is number or not using custom validator

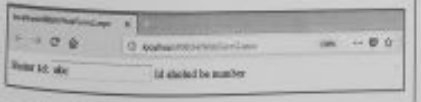
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm2.aspx.cs"
Inherits="WebApplication7.WebForm2" %>
```



```
<html>
<head id="Head1" runat="server">
<script language="javascript" type="text/javascript">
function check(sender, e) {
if (isNaN(e.Value))
      e.IsValid = false;
else

      e.IsValid = false;
  }
</script>

<title></title>
</head>

<body>
<form id="form1" runat="server">
  Enter Id:
<asp:TextBox ID="TextBox1" runat="server"
ontextchanged="TextBox1_TextChanged">
</asp:TextBox>
<asp:CustomValidator ID="CustomValidator1"
ControlToValidate="TextBox1"
    ErrorMessage="Id shoud be number "
ClientValidationFunction="check"
runat="server"></asp:CustomValidator>
<div>
</div>
</form>
</body>
</html>
```

Check input provided by user for id in textbox is number or not using check() javascript function

Output



5. Validation Summary

- Validation Summary control does not perform any validation.
- But Validation Summary control is used to displays a list of all validation errors on the Web page.
- Show Summary property is set as true by default.
- For displaying all validation errors Dispaly Mode property is used whose value can be SingleParagraph,BulletList and List.

Program 2.3.5

Write a program to show list of all the validation error messages using validation summary.

Solution :

A program to show list of all the validation error messages using validation summary.

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm7.aspx.cs"
Inherits="WebApplication7.WebForm7" %>

<html>
<head runat="server">
<title></title>
</head>
<body>

<h3>ValidationSummary Sample</h3>

<form id="Form1" runat="server">

<table cellpadding="10">
<tr>
<td>
<table bgcolor="#eeeeee" cellpadding="10">

<tr>
<td colspan="3">
<b>enter Credit Card Information</b>
</td>
</tr>
<tr>
<td align="right">
Enter  Card Type:
</td>
<td>
<asp:RadioButtonList id="RadioButtonList1"
                RepeatLayout="Flow"
runat="server">

<asp:ListItem>MasterCard</asp:ListItem>
```

```
<asp:ListItem>Visa</asp:ListItem>

</asp:RadioButtonList>
</td>
<td align="middle" rowspan="1">
<asp:RequiredFieldValidator
id="RequiredFieldValidator1"
        ControlToValidate="RadioButtonList1"
        ErrorMessage="Card Type is invalid."
        Display="Static"
        InitialValue=""
        Width="100%"
        Text="*"
runat="server">
</td>
</tr>
<tr>
<td align="right">
        Card Number:
</td>
<td>
<asp:TextBox id="TextBox1"
runat="server" />
</td>
<td>
<asp:RequiredFieldValidator
id="RequiredFieldValidator2"
        ControlToValidate="TextBox1"
        ErrorMessage="Card Number is invalid."
        Display="Static"
        Width="100%"
        Text="*"
runat="server">
</td>
</tr>

<tr>
<td></td>
<td>
<asp:Button id="Button1"
        Text="Verify"
runat="server" />
```

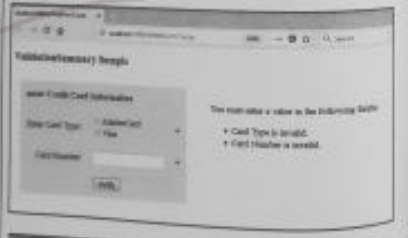
If card type is not selected from given options then error message is shown as "card type is invalid"

User must enter value in card number field.If user does not enter value error message is shown

```
</td>
<td></td>
</tr>
<tr>
</table>
</td>
<td valign="top">
<table cellpadding="20">
<tr>
<td>
<asp:ValidationSummary id="valSum"
        DisplayMode="BulletList"
        EnableClientScript="true"
HeaderText="You must enter a value in the following
fields:"
runat="server">
</td>
</tr>
</table>
</td>
</tr>
</table>

</form>

</body>
</html>
```

ValidationSummary control show list of validation error

Output



Syllabus Topic : Manual Validation

2.3.5 Manual Validation

- Client-side validation is triggered by default when submitting forms using buttons is done.

- Sometimes we may want to do client-side validation on our ASP page manually using custom JavaScript.
- We can do manual validation by calling Javascript validation functions provided by the ASP.NET framework.

Program 2.3.6

Write a program to demonstrate use of manual validation.

Solution :

Program to demonstrate use of manual validation.

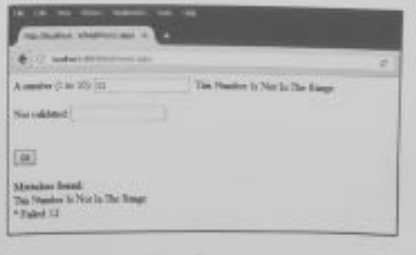
WebForm2.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm2.aspx.cs"
Inherits="WebApplication10.WebForm2" %>

  <html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
    A number (1 to 10):
<asp:TextBox id="txtValidated"
runat="server"></asp:TextBox> 
<asp:RangeValidator id="RangeValidator"
runat="server"
        ErrorMessage="This Number Is Not In
        The Range"
        ControlToValidate="txtValidated"
        MaximumValue="10"
        MinimumValue="1"
        Type="Integer"

EnableClientScript="False"></asp:RangeValidator>
<br/>
<br />
  Not validated:
<asp:TextBox id="txtNotValidated"
runat="server"></asp:TextBox><br/>
<br />
<br />
<asp:Button id="cmdOK" runat="server" Text="OK"
OnClick="cmdOK_Click"
Width="36px"></asp:Button><br />
<br />
<asp:Label id="lblMessage" runat="server"
EnableViewState="False"></asp:Label>
</div>
</form>
</body>
</html>
```

WebForm2.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace WebApplication10
{
public partial class WebForm2 : System.Web.UI.Page
  {
protected void Page_Load(object sender, EventArgs e)
    {
    }
protected void cmdOK_Click(object sender, EventArgs e)
    {
string errorMessage = "<b>Mistakes found:</b>
<br />";
bool pageIsValid = true;
foreach (BaseValidator ctrl in this.Validators)
    {
if (!ctrl.IsValid)
    {
pageIsValid = false;
errorMessage += ctrl.ErrorMessage + "<br />";
        TextBox ctrlInput =
(TextBox)this.FindControl(ctrl.ControlToValidate);
errorMessage += " * Failed: ";
errorMessage += ctrlInput.Text + "<br />";
    }
  }
if (!pageIsValid) lblMessage.Text = errorMessage;
  }
}
```

Output



Syllabus Topic : Validation with Regular Expression

2.3.6 Validation with Regular Expression

Q. Write short note on validation with regular expression. **(4 Marks)**

- The Regular Expression Validator permit us to validate the input text by matching against a pattern of a regular expression.
- The regular expression is used in the Validation Expression property of web control.
- Following are some escape sequence characters used in regular expressions :

Character Escapes	Description
\b	Used to find backspace.
\t	Used to find tab.
\r	Used to find a carriage return.
\v	Used to find vertical tab.
\f	Used to find form feed.
\n	Used to find new line.
\	Used to find Escape character.

- A class of characters could be state that can be matched, called the metacharacters.

Metacharacters	Description
.	Used to find any character except \n.
[abcd]	Used to find any character in the set.
[^abcd]	It is used excludes or remove any character in the set.
[2-7a-mA-M]	Used to find any character stated in the range.
\w	Used to find underscore and any alphanumeric character.
\W	Used to find any non-word character.
\s	Used to find whitespace characters such as space, tab,carriage return , new line etc.
\S	Used to find any non-whitespace character.
\d	Used to find any decimal character.
\D	Used to find any non-decimal character.

- Quantifiers could be used to states how many number of times a character could appear in string.

Quantifier	Description
*	It is used to show Zero or more matches.
+	It is used to show One or more matches.
?	It is used to show Zero or one matches.
{N}	It is used show N number of matches.
{N,}	It is used to show N or more number of matches.
{N,M}	It is used to show number of matches Between N and M.

Syntax

```
<asp:RegularExpressionValidator ID="string"
runat="server" ErrorMessage="string"
ValidationExpression="string" ValidationGroup="string">
</asp:RegularExpressionValidator>
```

Example :

1) Validation of Alphanumeric character with special Characters using RegularExpression

Minimum length is 7 charcters and Maximum length is 10 charcters allowed for input.

Characters allowed in input are a – z, A – Z, 0-9, @, &, #.

```
<asp:RegularExpressionValidator ID="RegularExpression1" runat="server"
ErrorMessage=" length of Password must be in range 7 to 10 charcters"
ControlToValidate=" Pass "
ValidationExpression=" ^ [a-zA-Z0-9@&#]{7,10}$" />
```

2) Validation of Alphanumeric character using RegularExpression

Minimum length is 7 charcters and Maximum length is 10 charcters allowed for input.

Characters allowed in input are a – z, A – Z, 0-9.

```
<asp:RegularExpressionValidator ID="RegExpression2"
runat="server"
ErrorMessage=" length of Password must be in range 7 to 10 charcters "
ControlToValidate=" Pwd1 "
ValidationExpression=" ^ [a-zA-Z0-9]{7,10}$" />
```


3) Validation of Alphabates using RegularExpression

Minimum length is 7 charcters and Maximum length is 10 charcters allowed for input.

Characters allowed in input are a – z, A – Z.

```
<asp:RegularExpressionValidator ID="RegExpression3"
runat="server"
ErrorMessage=" length of Password must be in range 7 to 10 charcters "
ControlToValidate="Pwd3"
ValidationExpression=" ^ [a-zA-Z]{7,10}$" />
```

4) Validation of numeric data using RegularExpression

Minimum length is 7 charcters and Maximum length is 10 charcters allowed for input.

Characters allowed in input are 0-9

```
<asp:RegularExpressionValidator ID="RegExpression4"
runat="server"
ErrorMessage=" length of Password must be in range 7 to 10 charcters "
ControlToValidate="Pwd4"
ValidationExpression=" ^ [0-9]{7,10}$" />
```

Program 2.3.7

Write a program to validate Email id entered by user using RegularExpression Validator.

Solution :

Program to validate Email id entered by user using RegularExpression Validator.

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm5.aspx.cs"
Inherits="WebApplication7.WebForm5" %>
<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>

Email: (Required and Valid Email Address)<br />
<asp:TextBox ID="txtEmail" runat="server"
Width="293px"></asp:TextBox>
<asp:RequiredFieldValidator
ID="RequiredFieldValidator2" CssClass="Validators"
Display="Dynamic" ControlToValidate="txtEmail"
runat="server"
ErrorMessage="Email is
required."></asp:RequiredFieldValidator>
<asp:RegularExpressionValidator
ID="RegularExpressionValidator1" runat="server"
ErrorMessage="Invalid Email Address"
ControlToValidate="txtEmail" CssClass="Validators"
Display="Dynamic"
ValidationExpression="\w+([-+.']\w+)*@\w
+([-.]\w+)*\.\w+([-.]\w+)*">
</asp:RegularExpressionValidator>
<br />
<asp:Button ID="Button1" runat="server" Text
="Submit" />

</div>
</form>
</body>
</html>
```

Check Email id entered by user using ValidationExpression Property of RegularExpression validator

Output

Syllabus Topic : Rich Controls

2.4 Rich Controls

Q. List all rich controls and explain any one rich control in detail. **(4 Marks)**

- Rich controls provide object model that has complex HTML representation and also client side JavaScript.
- They provide object model that is separate from underlying HTML representation.
- Typical rich control is often programmed as single object but renders itself with complex sequence of HTML elements and may even use client-side javascript.
- These controls implement special features.

Some examples of rich controls

i) **AdRotator control** is used to display randomly selected advertisement manner, on web page.

ii) **FileUpload** is used to upload any file.

iii) **Calendar** control is used to display months, days, year.

Syntax

```
<asp:controlTypeID="ControlID"runat="server"Property1
=value1 [Property2=value2] />
```

Syllabus Topic : Calendar Control

2.4.1 Calendar Control

Q. Explain calender control in detail with example. (4 Marks)

- ASP.NET privides a Calendar control that is used to display a calendar on our Web page.
- This control shows a single month calendar that permits the user to select date and move to the next and previous months.
- The user can move between months and select date and even select range of days when multiple selections is turned on.
- Calendar control has number of properties, using which we can customize each part of this control. For example it allow to set foreground and background colors,font size ,date format and so on.
- Calendar control provide events that enable reacting such as when the user changes current month – VisibleMonthChanged event is occurred ,when user select date -*selection changed* event occurs and when calendar is about to render a day then *DayRender* event occurs
- The Calendar control is represented as :

```
<asp:Calendar ID="Calendar1" runat="server"
</asp:Calendar>
```

- By default, this control displays the name of the current month, day headings for the days of the weeks, days of the month and arrow characters for navigation to the previous or next month.

Calendar control provides following features

1. Postback occurs due to user interaction with calendar each time. This allows to give reaction to selection event immediately and re-render its interface,thereby showing new month or newly selected dates.
2. Calendar control does not use Autopostback property.
3. Look and feel of calendar control can be changed according to programmer.
4. Information from database can be showed in calendar.

Properties and Events of the Calendar Control

The calendar control has many properties and events through which we can customize look and feel of calender control.

Properties	Description
Caption	This property is used to get and set the title for the calendar control.
CaptionAlign	This property is used to get and set the alignment for the caption.
WeekendDayStyle	This property is used to get the style for the weekend dates on the Calendar control.
VisibleDate	This property is used to get and set date which is displayed to user.
CellPadding	This property is used to get and set the number of spaces between the data and the cell border.
CellSpacing	This property is used to get and set the the space between cells.
DayNameFormat	This property is used to get and set the format for day of the week.
FirstDayOfWeek	This property is used to get and set the day of week to display in the first column.
TodaysDate	This property is used to get and set the value of today's date.
NextPrevFormat	This property is used to get and set the format of the next and previous month.
OtherMonthDayStyle	This property is used to get the style for the days on the Calendar control which are not selected by user.
TitleFormat	This property is used to get and set the format for the title section.
SelectWeekText	This property is used to get and set the text showed for the week selection element in the selector column.
SelectedDate	This property is used to get the selected date by user.
ShowTitle	This property is used to get and set a value showing whether the title section is displayed or not.

Properties	Description
SelectedDates	This property is used to get a collection of DateTime objects representing the selected dates.
SelectionMode	This property is used to get and set selection mode that state whether the user can choice a single day, a week or an entire month.
SelectMonthText	This property is used to get and set the text for the selected month in the selector column.
TodayDayStyle	This property is used to get the style properties for today's date on the Calendar control.
SelectorStyle	Gets the style properties for the week and month selector column.

- The Calendar control has the following three important events that allow the developers to create the calendar control. They are :

Events	Description
SelectionChanged	This event is raised when day, week or an entire month is selected.
VisibleMonthChanged	This event is raised when user changes a month.
DayRender	This event is raised when each data cell of the calendar control is rendered.

Program 2.4.1

Write a program using calander control to display today's date and birthdate of user.

Solution :

Program using calander control to display today's date and birthdate of user.

WebForm8.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm8.aspx.cs"
Inherits="WebApplication7.WebForm8" %>

<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">

<div>
<h3> Your Birthday:</h3>
<asp:Calendar ID="Calendar1" runat="server"
SelectionMode="DayWeekMonth"
onselectionchanged="Calendar1_SelectionChanged">
</asp:Calendar>
</div>
<p>Todays date is:
<asp:Label ID="labeltday"
runat="server"></asp:Label>
</p>
<p>Your Birday is:
<asp:Label ID="labelbday"
runat="server"></asp:Label>
</p>
</form>
</body>
</html>
```

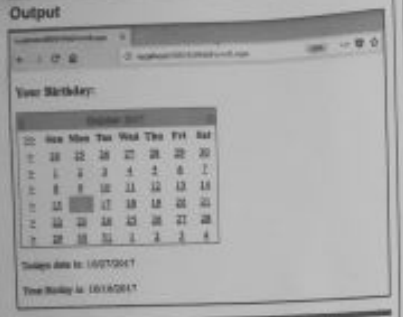
Show the calendar control from which user select his birthdate

webForm8.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace WebApplication7
{
public partial class WebForm8 : System.Web.UI.Page
  {
protected void Page_Load(object sender, EventArgs e)
    {
    }
protected void Calendar1_SelectionChanged
(object sender, EventArgs e)
    {
        labelday.Text =
Calendar1.TodaysDate.ToShortDateString();
        labelbday.Text =
Calendar1.SelectedDate.ToShortDateString();
    }
  }
}
```

When user select his birthdate date, this method is called to show the today's date and birth date of user.

Output



Syllabus Topic : AdRotator Control

2.4.2 AdRotator Control

Q. Explain AdRotator control in detail. (2 Marks)

- AdRotator is a control in ASP.Net which is concerned with advertisements.
- It basically displays a sequence of advertisement images.
- This control uses an XML file to store the information of advertisement.
- The XML file must begin and end with an <Advertisements> tag.
- Inside the <Advertisements> tag there may be several <Ad> tags.
- XML file maintains list of advertisements and their associated attributes.
- This attributes include path of image to display, URL to link when control is clicked and alternate text to display when unable to display image, keyword and frequency of advertisement.

Program 2.4.2

Write a program in ASP.NET using AdRotator control to display advertisements from XML file.

Solution :

Program in ASP.NET using AdRotator control to display advertisements from XML file

XmlFile.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>
 <Ad>
  <ImageUrl>~/Img/1.jpg</ImageUrl>

  <NavigateUrl>http://www.google.com</NavigateUrl>

    <AlternateText>
     AdRotator Control Demo
    </AlternateText>

    <Impressions>20</Impressions>
    <Keyword>Asp.Net</Keyword>

 </Ad>

</Advertisements>
```

Default3.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default3.aspx.cs" Inherits="Default3" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>

    <asp:AdRotator ID="AdRotator1" runat="server"
AdvertisementFile ="~/XMLFile.xml" />
  </div>
</form>

</body>
</html>
```


Output

After clicking on image

Syllabus Topic : MultiView Control

2.4.3 MultiView Control

Q. Write short note on Multiview control. (4 Marks)

- MultiView and View controls permit us to separate the content of a page into various groups and show only one group at a time.
- Every View control is used to group controls of one group and all the View controls are grouped into a MultiView control.
- The MultiView control has task of showing one View control at a time.
- The View which is used to display the group of contents is called the active view.

Syntax of multiview control

```
<asp:MultiView ID= "MultiView1" runat= "server">
</asp:MultiView>
```

Syntax of view control

```
<asp:View ID= "View1" runat= "server">
</asp:View>
```

- We can not use View control only. It would show error if we do not use it with multiview.
- It is always used with a Multiview control as :

```
<asp:MultiView ID= "MultiView1" runat= "server">
<asp:View ID= "View1" runat= "server"></asp:View>
</asp:MultiView>
```

Properties of View and MultiView Controls

1. Control class is parent class for Both View and MultiView control classes and these classes derive all its properties, methods, and events.
2. The vital property of the View control is Visible property whose data type is Boolean i.e. true or false, which sets the visibility of a view.
3. The MultiView control has the following important properties :

Properties	Description
Views	View is Group of View controls in the MultiView control.
ActiveViewIndex	ActiveViewIndex is zero based index which is used to represent the active view. Value of index is -1 if view is not active.

4. The navigation of the MultiView control is performed using CommandName attribute of the button web control.
5. For example, button which has "NextView" as value of CommandName attribute related with the navigation of the multiview, it automatically navigates to the next view when the button is clicked.

The methods of the multiview control are

Methods	Description
SetActiveview	This method is used to set the active view
GetActiveview	This method is used to retrieve the active view

- Each time when view is changed, the page is send back to the server and many events are occoured. Some events are as follows :

Events	Description
ActiveViewChanged	This event is occoured when a view is changed

Events	Description
Activate	This event is fired by the active view
Deactivate	This event is fired by the inactive view

Program 2.4.3

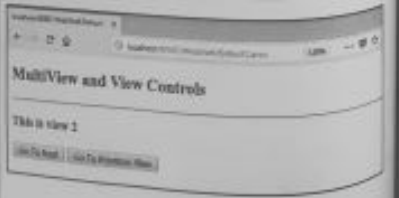
Write a program using multiview and view control to display three different views.

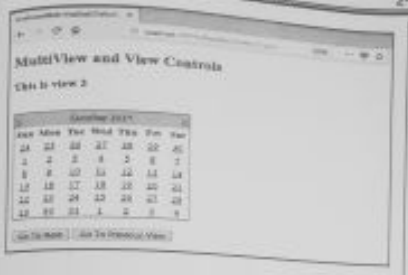
Solution :

Program using multiview and view control to display three different views.

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default2.aspx.cs" Inherits="Default2" %>

<html>
<head runat="server">
<title></title>
</head>
<body>
<form id="form1" runat="server">
<div>
<h2>MultiView and View Controls</h2>

<br />

<asp:MultiView ID="MultiView1" runat="server"
ActiveViewIndex="2"
onactiveviewchanged="MultiView1_ActiveViewChanged">
<asp:View ID="View1" runat="server">
<h3>This is view 1</h3>
<br />
<asp:Button CommandName="NextView" ID="btnnext1"
runat="server"
        Text = "Go To Next"
onclick="btnnext1_Click" />
<asp:Button CommandArgument="View3"
CommandName="SwitchViewByID" ID="btnlast"
runat="server" Text ="Go To Last" />
</asp:View>

<asp:View ID="View2" runat="server">
<h3>This is view 2</h3>
<asp:Button CommandName="NextView" ID="btnnext2"
runat="server" Text = "Go To Next" />
<asp:Button CommandName="PrevView"
ID="btnprevious2" runat="server" Text = "Go To Previous
View" />
</asp:View>

<asp:View ID="View3" runat="server">
<h3>This is view 3</h3>
<br />
<asp:Calendar ID="Calender1" runat="server"
onselectionchanged="Calender1_SelectionChanged">
</asp:Calendar>
<br />
<asp:Button CommandArgument="0"
CommandName="SwitchViewByIndex" ID="btnfirst"
runat="server" Text = "Go To Next" />
<asp:Button CommandName="PrevView"
ID="btnprevious" runat="server"
        Text = "Go To Previous View" />
</asp:View>

</asp:MultiView>
</div>
</form>
</body>
</html>
```

Output





Syllabus Topic : Themes and Master Pages

2.5 Themes and Master Pages

Q. Explain themes and master pages in detail. **(4 Marks)**

Themes

- Themes are used to fix the look and feel of our website.
- It is a group of files. It can include skin files, CSS files and images.
- We can define themes in App_Themes folder. This folder contains one or more subfolders named Theme1, Theme2 etc. that define the actual themes.
- The theme property is applied to the life cycle of page to effectively overriding any customization for individual controls on our web page.

Master Pages

- Master page permits us to create a consistent look and feel for all the web pages in our web applications.
- Design of Master Page is common for all the pages.
- We will discuss Master Page in detail in section 2.5.4

Syllabus Topic : How Themes Work

2.5.1 How Themes Work ?

Q. How theme works? **(4 Marks)**

- There are 3 different options to apply themes to our website :

I. **Setting the theme at the page level :** The Theme attribute is added to the page directive of the page.

```
<%@ Page Language="C#"
AutoEventWireup="true"
CodeFile="Default2.aspx.cs" Inherits="Default2"
Theme="Theme1"%>
```

2. **Setting the theme at the web site level :** For setting the theme for the entire website, we can set the theme in the web.config file of the website. Open the web.config file and add theme attribute in <pages> element :

```
<pages theme="Theme1">
...
...
</pages>
```

3. **Setting the theme programmatically at runtime :** Here the theme is set at runtime through coding. It should be applied earlier in the page's life cycle i.e. PreInit event should be handled for setting the theme. The better option is to apply this to the Base page class of the site as every page in the site inherits from this class.

```
Page.Theme = "Theme1";
```

Uses of Themes

1. Themes can include CSS files, images and skins so we can change colors, fonts, positioning and images simply by applying the desired themes.
2. We can have many themes and we can change theme by setting a single attribute in the web.config file or an individual aspx page. Also we can change themes programmatically.
3. Themes allows us to provide the better usability(user friendliness) to our web site by giving users.

Syllabus Topic : Applying a Simple Theme

2.5.2 Applying a Simple Theme

Q. How to apply simple theme? **(4 Marks)**

Apply simple theme to application using following steps :

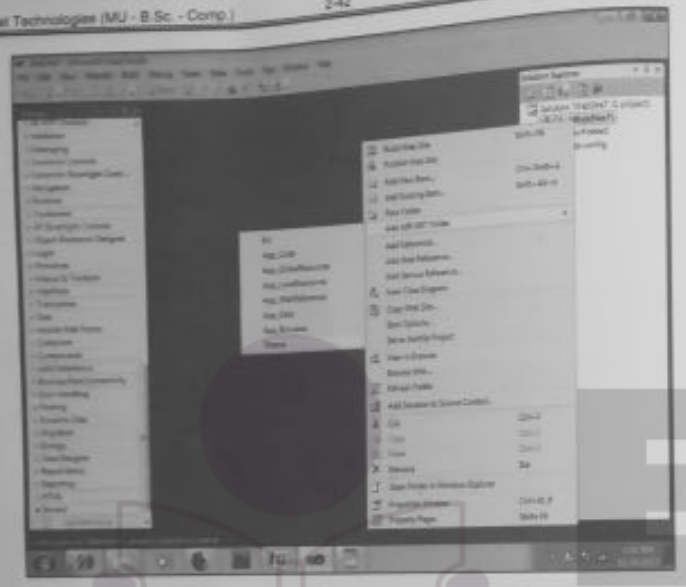
Step 1 : Open the Microsoft Visual Studio 2010.

Step 2 : Select File option,click on New and then select Web Site option. i.e File->New->Web site

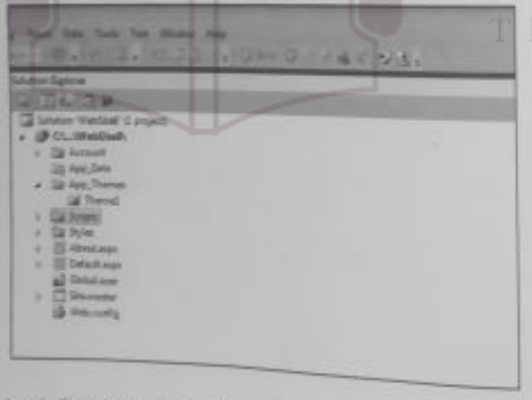
Step 3 : Select ASP.NET Web Site option and then click on Ok.

Step 4 : Then, Right click on the name of application in the Solution Explorer window and select Add ASP.NET Folder and then select submenu Theme from Add ASP.NET folder menu.

i.e. right click on name of application in Solution Explorer ->Add ASP.NET folder->Theme

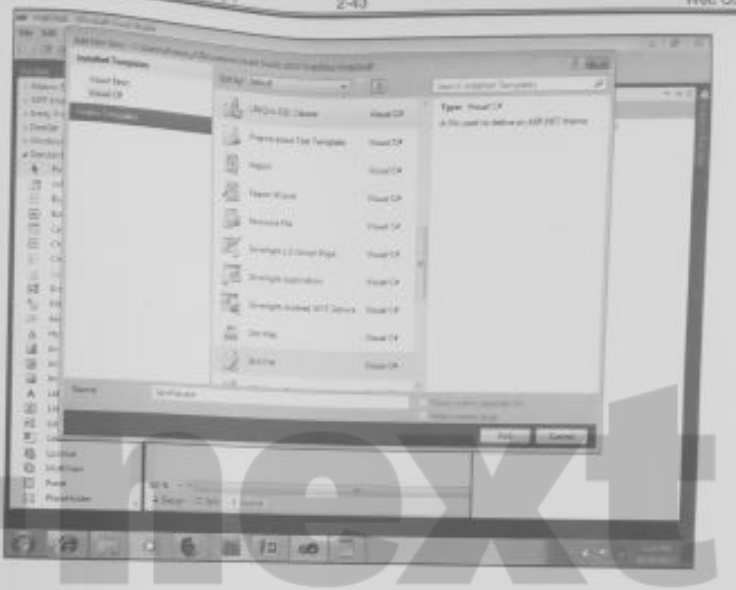


A sub folder named Theme1 is automatically created inside the APP_Themes folder in the Solution Explorer.

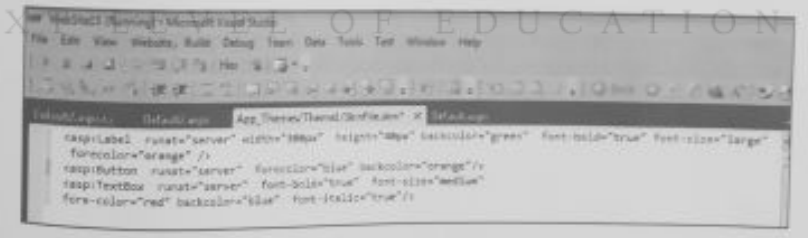


Step 5 : Right-click on the Theme1 folder and select the Add New Item option.i.e Theme1->Add New Item

Step 6 : Select the Skin file option and click on the Add button.



Now write the following code in the skin file.



```
<asp:Label runat="server" width="300px" height="50px" backcolor="green" font-bold="true" font-size="large"
forecolor="orange" />
<asp:Button runat="server" forecolor="blue" backcolor="orange"/>
<asp:TextBox runat="server" font-bold="true" font-size="medium"
fore-color="red" backcolor="blue" font-italic="true"/>
```

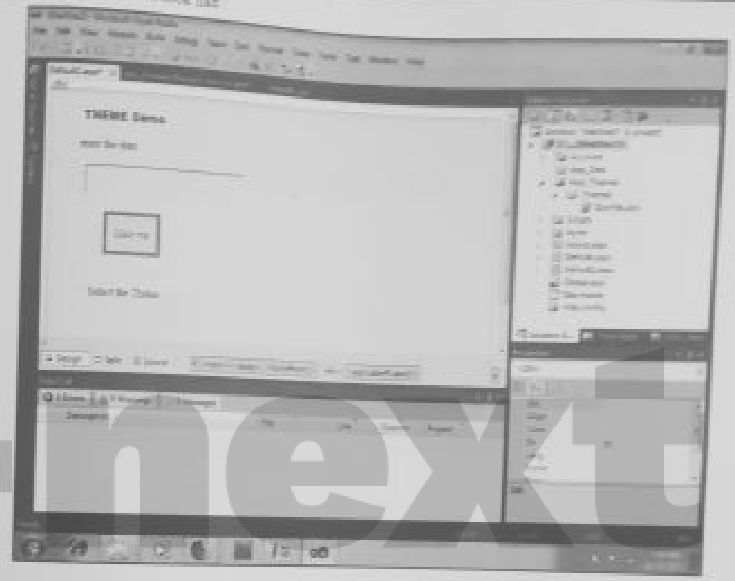
Step 7 : Write the following code for Default2.aspx page.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default2.aspx.cs" Inherits="Default2"
Theme="Theme1" %>

<html>
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
```

Set a Theme to page as Theme1

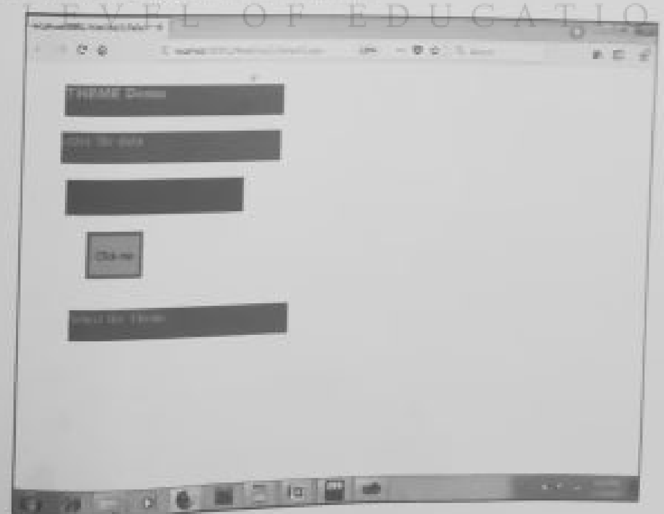
```
<div>     
<br />
           
<asp:Label ID="Label2" runat="server" Font-Bold="True"
        Font-Names="Arial black" Font-Underline="True"
        Text="THEME Demo"></asp:Label>
<br />
<br />
         
<asp:Label ID="Label1" runat="server" Text="enter the data">
</asp:Label>
  <br />
<br />
           
<asp:TextBox ID="TextBox1" runat="server" Height="42px" Width="257px"></asp:TextBox>
<br />
<br />
            
     
<asp:Button ID="Button1" runat="server" Text="Click me"
        BorderColor="#996633" BorderStyle="Solid" BorderWidth="3px" Height="60px" />
<br />
<br />
<br />
              
<asp:Label ID="Label3" runat="server" Text="Select the Theme"></asp:Label>
<br />
<br />

<br />

<br />
<br />
<br />
</div>

</form>
</body>
</html>
```


The design window will look like :



Step 8 : Now write the following code for the Default2.aspx.cs file.

Output

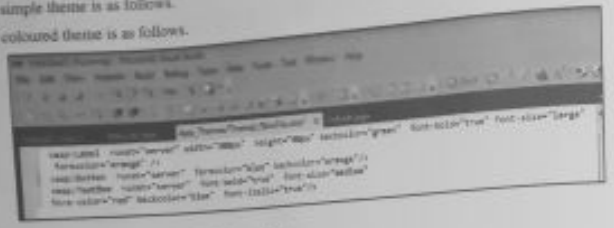


Now we will create an application for applying the theme at run time.

Step 9 : Press Ctrl+F5 keys to run the application.

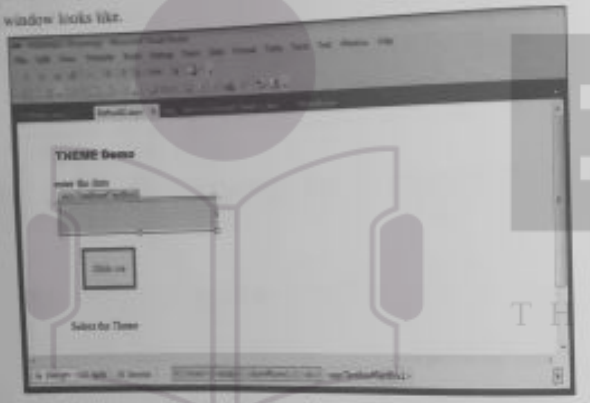
Code for simple theme is as follows.

Code for coloured theme is as follows.



Step 10 : Add the following code to the Default.aspx file.

The design window looks like.



Step 11 : Now write the following code for the default.aspx.cs file.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
public partial class _Default : System.Web.UI.Page
{
    void Page_PreInit(object sender, EventArgs e)
    {
        // Get the theme name from a QueryString variable
        string ThemeName;
        ThemeName = Request.QueryString["Theme"];
        if (ThemeName != null)
```



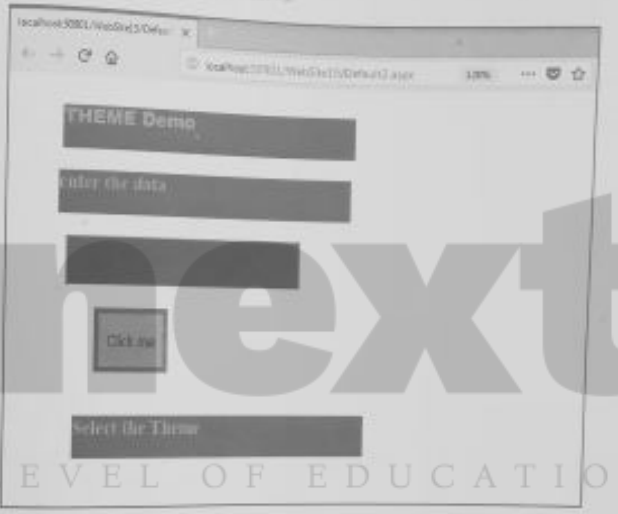
```
    {
        Page.Theme = ThemeName;
    }
}
```

Set theme to page

Step 12 : Press F5 keys to run the application.

Output

After selecting the Coloured theme the output as follows.



Syllabus Topic : Handling Theme Conflicts

2.5.3 Handling Theme Conflicts

Q. How to handle theme conflict? (2 Marks)

- When properties confuse between web controls and theme, properties can give priority to theme.For this purpose just use the StyleSheetTheme attribute at the place of the Theme attribute.

```
<%@ Page Language="C#" AutoEventWireup="true" ...
StyleSheetTheme="FunkyTheme" %>
```

Syllabus Topic : Simple Master Page and Content Page and Connecting Master Pages and Content Page

2.5.4 Simple Master Page and Content Page and Connecting Master pages and Content Page

Q. What is concept of master pages and content pages and how to connect them? (4 Marks)

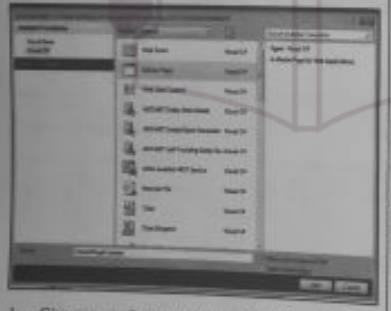
- Master page permits us to create a consistent look and feel for all the web pages in our web applications.
- Design of Master Page is common for all the pages.
- Master page contains of two parts, one is the master page itself and another is one or more content pages.

- A master page is an ASP.NET file with the extension .master.
- ContentPlaceholder control defines a region on the master page at which content pages can plug in the page specific content.
- The @Master directive is used to define a page as master page.
- The ContentPlaceHolder web control is available only for masterpages.
- Content pages are ASP.NET pages which uses master pages.
- Master page provide us a way to create common set of User Interface elements that are needed on number of pages in our website.
- ContentPage are ASP.NET web page that will use master page to have the common UI elements rendering.
- ContentPlaceHolder is a control that should be added on the MasterPage which will reserve the place for the content pages to run their contents.
- ContentControl is control which will be included on content pages to show these pages that the contents inside this control will be run where the ContentPlaceHolder of master page is located.

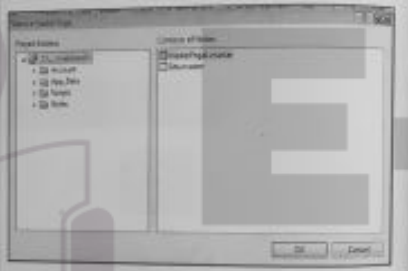
Following are the stpes to create Creating a MasterPage

Step 1 : Go to "Add New Item".

Step 2 : Select the MasterPage.



1. Give name to that master page as MasterPage1.Master.
2. Now add a menu bar on the master page at top of the page. This Menu bar will be common to all the pages because that menu bar is present in Master page.
3. Now we add some content pages like Home.aspx, default.aspx, about.aspx, Contact.aspx etc.
4. When we add these content pages in our web site, we should remember to select the option of "Use master Page" and select the master page.



- When we look at the MasterPage1, we will see that master page has a ContentPlaceHolder control. All the code that is common for the content pages is outside the ContentPlaceHolder control (in our case, a simple menubar).

Program 2.5.1

Write a program to create a master page which contain menu bar that include menus such as Home,About us,Gallary and so on and create some content pages under the master page.

Solution :

Program to create a master page which contain menu bar that include menus such as Home,About us,Gallary and so on and create some content pages under the master page.

MasterPage1.aspx(master page)

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeFile="MasterPage1.master.cs"
Inherits="MasterPage1" %>
```



```
<html>
<head runat="server">
<title></title>
<asp:ContentPlaceHolder id="head" runat="server">
</asp:ContentPlaceHolder>
</head>
<body>
<form id="form1" runat="server">
<div align="center">
<h1>MyFirst WebSite</h1>
<asp:Menu ID="Menu1" runat="server"
BackColor="#B5C7DE" DynamicHorizontalOffset="2"
Font-Names="Verdana" Font-Size="0.8em"
ForeColor="#284E98" Orientation="Horizontal"
StaticSubMenuIndent="10px">
<StaticMenuItemStyle HorizontalPadding="5px"
VerticalPadding="2px" />
<DynamicHoverStyle BackColor="#284E98"
ForeColor="White" />
<DynamicMenuStyle BackColor="#B5C7DE" />
<StaticSelectedStyle BackColor="#507CD1" />
<DynamicSelectedStyle BackColor="#507CD1" />
<DynamicMenuItemStyle HorizontalPadding="5px"
VerticalPadding="2px" />
<Items>
<asp:MenuItem Text="HOME" Value="home"
NavigateUrl="~/Default2.aspx">
</asp:MenuItem>
<asp:MenuItem Text="ABOUT" Value="about"
NavigateUrl="~/about1.aspx">
</asp:MenuItem>
<asp:MenuItem Text="CONTACTNO" Value="contact"
NavigateUrl="~/contact1.aspx">
</asp:MenuItem>
<asp:MenuItem Text="GALLARY" Value="contact"
NavigateUrl="~/gallary.aspx">
</asp:MenuItem>
</Items>
<StaticHoverStyle BackColor="#284E98"
ForeColor="White" />
</asp:Menu>
```

Add menubar with Home,About,Contact No etc menus

```
<asp:contentplaceholder id="ContentPlaceHolder1"
runat="server">
</asp:contentplaceholder>

</div>

</form>
</body>
</html>
```

Program 2.5.2

Write a program to adding the ContentPages to our website.

Solution :

Program to adding the ContentPages to our website

Default2.aspx(content page)

```
<%@ Page Title="" Language="C#"
MasterPageFile="~/MasterPage1.master"
AutoEventWireup="true" CodeFile="Default2.aspx.cs"
Inherits="Default2" %>

<asp:Content ID="Content1"
ContentPlaceHolderID="head" Runat="Server">
</asp:Content>

<asp:Content ID="Content2"
ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">

<h2>This is a the about page.</h2>
</asp:Content>
```

about1.aspx(content page)

```
<%@ Page Title="" Language="C#"
MasterPageFile="~/MasterPage1.master"
AutoEventWireup="true" CodeFile="about1.aspx.cs"
Inherits="about1" %>

<asp:Content ID="Content1"
ContentPlaceHolderID="head" Runat="Server">
</asp:Content>

<asp:Content ID="Content2"
ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">

<h2>This is a the Home page.</h2>
</asp:Content>
```

contact1.aspx(content page)

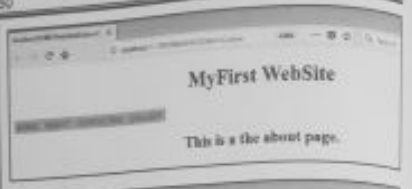
```
<%@ Page Title="" Language="C#"
MasterPageFile="~/MasterPage1.master"
AutoEventWireup="true" CodeFile="contact1.aspx.cs"
Inherits="contact1" %>

<asp:Content ID="Content1"
ContentPlaceHolderID="head" Runat="Server">
</asp:Content>

<asp:Content ID="Content2"
ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">

<h2>This is a the  CONTACT  page.</h2>
</asp:Content>
```

gallary.aspx(content page)

```
<%@ Page Title="" Language="C#"
MasterPageFile="~/MasterPage1.master"
AutoEventWireup="true" CodeFile="contact1.aspx.cs"
Inherits="contact1" %>

<asp:Content ID="Content1"
ContentPlaceHolderID="head" Runat="Server">
</asp:Content>

<asp:Content ID="Content2"
ContentPlaceHolderID="ContentPlaceHolder1"
Runat="Server">

<h2>This is a the  Gallary  page.</h2>
</asp:Content>
```

Once we have all the master pages and content pages ready, we can test our website.

Output



So the common set of controls and behavior has now been added to all the content pages. We can have any number of controls and functionality on the master pages.

Syllabus Topic : Master Page with Multiple Content Regions

2.5.5 Master Page with Multiple Content Regions

Q. Explain with example : Master page with multiple content regions. (4 Marks)

- Masterpage1.master is master page which contain two ContentPlaceHolders by default. It may contains more contentplaceholders .

Program 2.5.3

Write a program of Master Page with multiple contents

Solution :

Program of Master Page with multiple contents

```
<%@ Master Language="C#" AutoEventWireup="true"
CodeFile="MasterPage1.master.cs"
Inherits="MasterPage1" %>

<html>
<head runat="server">
<title></title>
```

First ContentPlaceHolder


```
</asp:ContentPlaceHolder>
</head>
<body>
<form id="form1" runat="server">
<div align="center">
<h1>MyFirst  WebSite</h1>
<asp:ContentPlaceHolder id="body" runat="server">
<asp:Menu ID="Menu1" runat="server"
BackColor="#B5C7DE" DynamicHorizontalOffset="2"

    Font-Names="Verdana" Font-Size="10pt"

    ForeColor="#284E98" Orientation="Horizontal"

    StaticSubMenuIndent="10px">
<StaticMenuItemStyle HorizontalPadding="5px"
VerticalPadding="2px" />
<DynamicHoverStyle BackColor="#284E98"
ForeColor="White" />
<DynamicMenuStyle BackColor="#B5C7DE" />
<StaticSelectedStyle BackColor="#507CD1" />
<DynamicSelectedStyle BackColor="#507CD1" />
<DynamicMenuItemStyle HorizontalPadding="5px"
VerticalPadding="2px" />
<Items>
<asp:MenuItem Text="HOME" Value="home"
NavigateUrl="~/Default2.aspx">
</asp:MenuItem>
<asp:MenuItem Text="ABOUT" Value="about"
NavigateUrl="~/about1.aspx">
</asp:MenuItem>
<asp:MenuItem Text="CONTACTNO" Value="contact"
NavigateUrl="~/contact1.aspx">
</asp:MenuItem>
<asp:MenuItem Text="GALLARY" Value="contact"
NavigateUrl="~/gallary.aspx">
</asp:MenuItem>
</Items>
<StaticHoverStyle BackColor="#284E98"
ForeColor="White" />
</asp:Menu>
</asp:ContentPlaceHolder>
```

Second ContentPlaceHolder

```
<asp:contentplaceholder id="ContentPlaceHolder1"
runat="server">
</asp:contentplaceholder>

</div>
</form>
<asp:ContentPlaceHolder id="footer" runat="server">
</asp:ContentPlaceHolder>
</body>
</html>
```

Third ContentPlaceHolder

Syllabus Topic : Master Pages and Relative Paths

2.5.6 Master Pages and Relative Paths

Q. Explain master pages and relative paths in detail. (4 Marks)

- Relative path is used in master page to specify a link to our client javascript file or a Cascading Style Sheet (CSS) file or images.
- A path on a web page is called as relative path if the location of the resource it points to is relative to the web page's location in the website's folder structure.
- Any path that does not starts with forward slash (/) or a protocol such as http:// is relative because it is resolved by the browser based on the location of the web page that is written in that path.
- For example, our website has an ~/Images/ folder with ,Img1.gif. The master page file MasterPage1.master has an <img> element in the footerContent part with the following HTML code :

```
<div id="footerContent">
<img src="~/Images/Img1.gif"alt="Beautiful Image" />
</div>
```

- The ~/Admin/Default.aspx content page is sent to the HTML for the footerContentarea :

Html

```
<div id="footerContent">
<img src="img2.gif" alt="nic image" />
</div>
```

As the value of <img> element's src attribute is a relative path, the browser try to look for Images folder relative to the web page folder location. In other words, the browser is looking for the image file in Admin/Images/img.gif.

Replacing Relative URLs with Absolute path

- Absolute path is opposite of a relative path which is one that contains a forward slash (/) at start or a protocol such as http://.
- An absolute path state the place of a resource from fixed point.
- The same absolute path is valid in any other web page, regardless of the place of web page in the folder structure of website.
- ASP.NET offers a method for generating a valid relative path at runtime.

Using ~ and ResolveClient path

- In another way of generating absolute URL, ASP.NET permits coder to use the tilde (~) to state the root of the web application.
- For example, earlier we use path ~/Admin/Default.aspx in the text to refer to the Default.aspx page in the Admin folder.
- The tilde (~) state that the Admin folder is a subfolder of the root of web application.

Using ~ in the Declarative Markup

- Several ASP.NET Web controls include path -related properties such as the HyperLink control has a NavigateUrl property and the Image control has an ImageUrl property etc.
- When executed, these controls send their values of path -related properties to ResolveClientUrl. Consequently, if these properties contain a ~ to indicate the root of the web application, the path will be changed to make valid relative path.
- Remember that ASP.NET server controls can only work on tilde (~) in their path-related properties.
- ~ appears in static HTML code, such as

```
<img src="~/Images/Img2.gif" />
```

The ~ send by ASP.NET engine to the browser with the rest of the HTML content.

- The browser suppose that the ~ is part of the relative path.
- For example, if the browser receives the HTML code

```
<img src="~/Images/Img2.gif" />
```

It oppose that there is a subfolder named ~ with a subfolder Images that include the image file Img2.gif

- To fix the image code in MasterPage1.master, replace the existing <img>tag with an ASP.NET Image Web control.
- Set the ID of Image web control as Img2 and value of its ImageUrl property to ~/Images/Img2.gif, and its AlternateText property to "Beautiful Image!"

aspx

```
<div id="Contentatfooter">
<asp:Image ID="Img2" runat="server"
ImageUrl="~/Images/Img2.gif"
AlternateText="Beautiful Image!" />
</div>
```

- After making this change to the master page, now again open the~/Admin/Default.aspx web page again.
- This time the Img2.gif image file appears in the page.
- When the Image Web control is run, it uses the ResolveClientUrl method to resolve its ImageUrl property value.
- In ~/Admin/Default.aspx the ImageUrl is converted into an appropriate relative path, as the following snippet of HTML source shows :

html

```
<div id="footerContent">
<img
id="Img2" src="../Images/Img2.gif" alt="Image
by ASP.NET!" />
</div>
```

Syllabus Topic : Website Navigation

2.6 Website Navigation

- ASP.NET provides various site-navigation features which gives a consistent way for visitors to navigate the site comfortably.
- These features are : Site Maps, URL Mapping and Routing, SiteMapPath.

Syllabus Topic : Site Maps

2.6.1 Site Maps

Q. How to create sitemap? (4 Marks)

- To use ASP.NET site navigation, we must state the design of the site so that the site navigation API and the site navigation controls can interpret the design of site correctly.
- The site navigation system use an XML file that contains the site navigation hierarchy by default.

- We can also made setting in the site navigation system to use another data sources.
- Google, Yahoo and Microsoft create standard sitemap which provide an easy way to owners of web site to send information to search engines about pages on their site that are available for crawling.
- Web crawlers find pages from links within the site and from other sites.
- Sitemap attach this data to permits web crawlers that helps the sitemaps to pick up all URLs in the sitemap and learn about those URLs using the associated metadata(data about data).

The Web.sitemap File

- The easy way to generate a site map file is to create an XML file with name Web.sitemap that arrange the pages in the sitemap.
- The site map is provided automatically by the default site-map provider for ASP.NET.
- The Web.sitemap file must be stored at application root directory.
- The below code state how the site map might look for a simple site. The url attribute contain value that can start with the "~/" which indicates the application root directory.

```
<siteMap>
<siteMapNode title="Home" description="Home"
url="~/home.aspx">
<siteMapNode title="Products" description="products"
url="~/Products.aspx">
<siteMapNode title="Hardware" description="Hardware
choices"
url="~/Hardware.aspx" />
<siteMapNode title="Software" description="Software "
url="~/Software.aspx" />
</siteMapNode>
<siteMapNode title="Hardware" description="Hardware
that we can provide"
url="~/Hardware.aspx">
<siteMapNode title=" Personal Training"
description="Training classes"
url="~/Training/classes.aspx" />
<siteMapNode title="Carrier Consulting" description=
" Carrier Consulting services"
url="~/CarrierConsulting.aspx" />
<siteMapNode title="ProductGallary" description=
" ProductGallary "
url="~/ ProductGallary.aspx" />
<siteMapNode title=" Feedback" description="Feedback"
url="~/Feedback.aspx" />
</siteMapNode>
</siteMapNode>
</siteMap>
```

- We add a siteMapNode element in Web.sitemap file for each page in our Web site.
- Then hierarchy of nested siteMapNode elements is generated.
- In the above example, the pages of Hardware and Software are child elements of the Products element.
- The title attribute provide the text that is generally used as linked text.
- The description attribute is used to represent both documentation and tool tip in the SiteMapPath control.

Valid Site Maps

- A valid site-map file include only one siteMapNode element that is placed immediately under the sitemap element.
- Any number of child siteMapNode elements can be included in the first-level siteMapNode element.
- A valid sitemap file should not contains duplicate URLs.
- This restriction may be only for default site-map provider for ASP.NET.

Configuring Multiple Site Maps

- We can use many site-map files or many site-map providers to state the navigation structure of our whole Web site.
- For example, our root Web.sitemap file contains link for its child site-map file by using reference to it's child site-map file in a siteMapNode element by using the below code :

```
<siteMapNode siteMapFile="FirstSiteMap.sitemap" />
```

Syllabus Topic : URL Mapping and Routing

2.6.2 URL Mapping and Routing

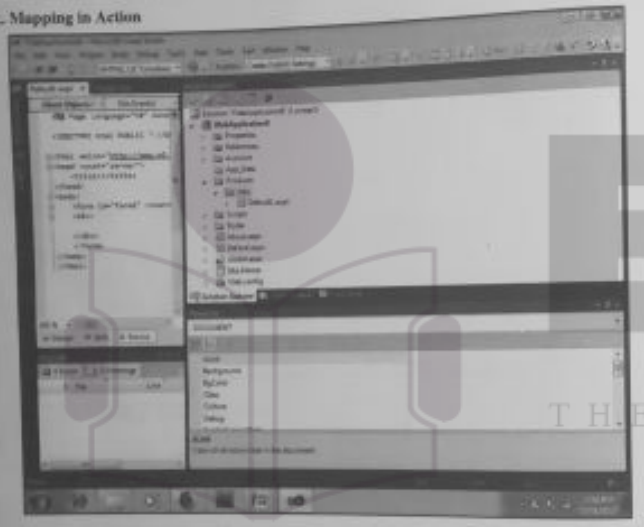
2.6.2(a) URL Mapping

Q. What is use of tilde (~) in URL mapping? (2 Marks)

- Ideally we create our Web application accurately in our first attempt. Pages should be created in the proper folder and stay there.

- For that matter, end users would not care about the URLs of the pages in our Web applications, so we could put pages anywhere without worrying about it.
- In the real world, things are not that much simple.
- We may decide after released an application for which we actually want to identify the folder structure.
- Users may like to see URLs like http://www.PhoenixGlobe.com/Default1.aspx instead of http://www.PhoenixGlobe.com/Products/new/ Default1.aspx.
- The urlMappings element resend URLs to new place. This element is part of an ASP.NET web.config file. It is simply included to Web.config to perform this complex task.

URL Mapping in Action



- By default, the Default1 .aspx page is accessible at ~/Products/new/Default1.aspx where the tilde character (~) is used to indicate root of the our Web site.
- We can change~/Products/new/Default1.aspx to ~/Products/Default1.aspx by making an entry in the web.config file of our web application.
- To do this, we require to include the <urlMapping> section inside of the <system.web> section. Here's an example to do the mapping that we want :

```
<system.web>
  <urlMappings enabled="true">
    <add url="~/Default.aspx" mappedUrl="~/Products/new/Default1.aspx"/>
  </urlMappings>
</system.web>
```

- Inside of the <urlMappings> element we can add many elements using <add> tag.
- Each of these <add> include a url attribute which is used to state the URL that the user will enter, and a mappedUrl attribute, which state the actual URL within the application to deliver to the entered URL.

2.6.2(b) URL Routing

Q. What is URL routing? (2 Marks)

- URL Routing is new feature in ASP.Net 4.0.
- ASP.NET routing permits us to use URL due to which we does not need to map to specific files in a Web site.
- So we can use URLs that describe action of user.
- The ASP.NET MVC framework and ASP.NET Dynamic Data use routing to provide features that are used only in MVC applications and in Dynamic Data applications.
- For example, a request for http://server/application/Products.aspx?id=4 maps to a file that is named Products.aspx that include code and markup for rendering a response to the browser.
- The Web page uses the query string which use id=4 to determine what type of data to show.
- In ASP.NET routing, we can store URL patterns that map to request-handler files.
- ASP.NET routing do not require adding names of those files in the URL.
- In addition, we can add placeholders in a URL pattern so that variable data can be passed to the request handler without using a query string.
- ASP.Net 4.0 URL Routing enables mapping between search engine's optimized URL to physical Web Form pages.

Syllabus Topic : SiteMapPath Control

2.6.3 SiteMapPath Control

Q. Explain siteMapPath control in detail. (4 Marks)

- To access webpage from another webpage SiteMapPath web control is used.
- This control is used mainly for navigation and shows the map of the site related to its web pages.
- This map contains the pages of the particular website and shows the name of those pages.
- We can click on that particular page in the Site Map to go to that page.
- SiteMapPath control is used to show links for connecting to URLs of other pages.
- The SiteMapPath control uses SiteMapProvider property which is useful for accessing data from databases and it stores the information in a data source.
- The SiteMapPath web control find out navigation data from a site map.
- This data contains information about the pages in our website like the URL, title, description, and place in the navigation hierarchy.
- The presentation of the SiteMapPath control is as follows :

Root Node->Child Node

Public Properties of SiteMapPath class

1. **ParentLevelsDisplayed** : Number of levels of parent nodes are shown by this property.
2. **RenderCurrentNodeAsLink** : This property is used to state whether or not the site navigation node that represents the current page is running as a hyperlink.
3. **PathSeperator** : This property states the string that shows the SiteMapPath nodes in the rendered navigation path.

Style properties of the SiteMapPath class

1. **CurrentNodeStyle** : This property is used to specify the style which is used to show the text for the current node.
2. **RootNodeStyle** : This property determine the style which is for the root node style text.
3. **NodeStyle** : This property determines the style which is used to display text for all nodes in the site navigation path.

- Now we can use the SiteMapPath control using following steps :

Step 1 : Open Microsoft Visual Studio 2010.

Step 2 : Select File->New->Web Site.

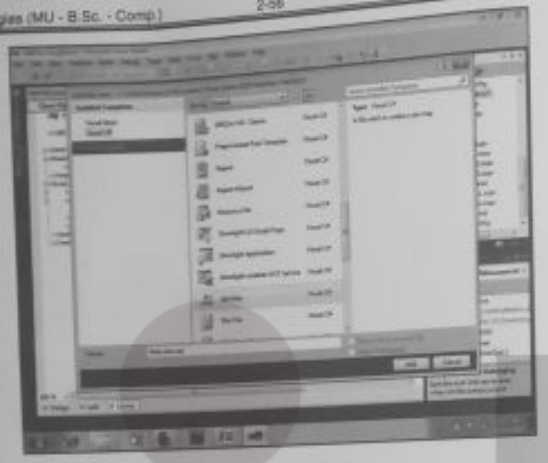
Step 3 : Select ASP.NET Web Site and name it.

Step 4 : Add two web forms to the application named MyWeb1.aspx and AddWeb2.aspx by performing the following steps.

Step 5 : In same way add the AddWeb2.aspx web form to the application. After that we have to add the Site Map file into the project. The Site Map file is the XML file and has the extension .sitemap. The steps are as follows.

Step 6 : Right-click the application in the Solution Explorer window and then click the Add New Item option from the context menu.

Step 7 : Select the Site Map Template from the Templates Pane. Note that, by default, the file has the name web.sitemap.

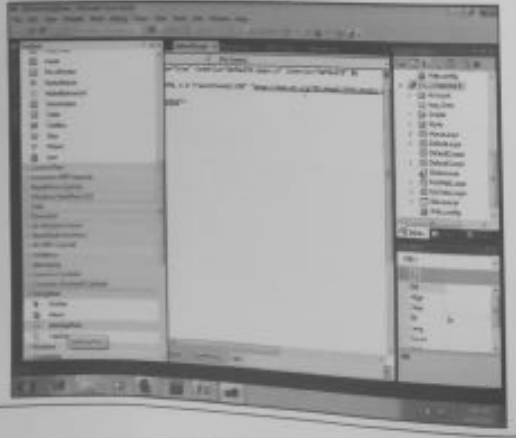


Step 8 : Now click the Add button to add the sitemap file to our application.

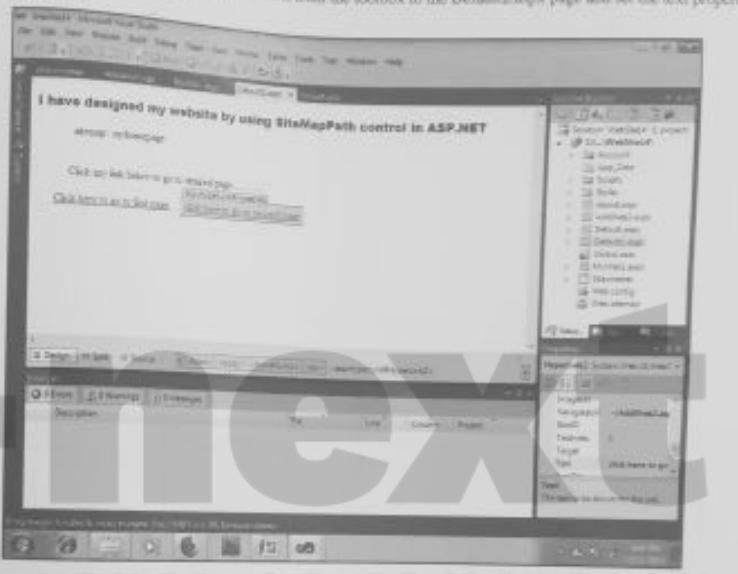
Step 9 : Now we can check that the sitemap file has been included in our project and we can see it in the Solution Explorer. And now we have to set the URL and title attributes in the sitemap file.

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
    <siteMapNode url="Default2.aspx" title="mythomepage"  description="" >
      <siteMapNode url="Mywebl.aspx" title="myfirstpage"  description="" />
      <siteMapNode url="AddWeb2.aspx" title="mysecondpage"  description="" />
    </siteMapNode>
</siteMap>
```

Step 10 : Now Add one SiteMapPath control on the Default2.aspx page from the navigation tab of the toolbox.



Step 11 : Add three labels and two hyperlink controls from the toolbox to the Default2.aspx page and set the text property of the control.



Step 12 : The source code of the Default2.aspx page is as follows :

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default2.aspx.cs" Inherits="Default2" %>

<html>
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <asp:Label ID="Label1" runat="server"
    Text="I have designed my website by using
SiteMapPath control in ASP.NET"
    Font-Bold="True" Font-Names="Arial Black" Font-
Size="Large"></asp:Label>
  <br />
  <br />
          
    
  <asp:Label ID="Label2" runat="server"
Text="sitemap:"></asp:Label>
     
  <asp:SiteMapPath ID="SiteMapPath1"
runat="server">
  </asp:SiteMapPath>
  <br />
  <br />
  <br />

         
  
  <asp:Label ID="Label3" runat="server"
    Text="Click any link below  to go to desired
page.....">
  <br />
  <br />
```

```
             
    <asp:HyperLink ID="HyperLink1" runat="server"
NavigateUrl="~/MyWeb1.aspx">Click here to go to first
page< asp:HyperLink>
            
    <asp:HyperLink ID="HyperLink2" runat="server"
NavigateUrl="~/AddWeb2.aspx"><click here to go to
second page</asp:HyperLink>
    </div>
    </form>
</body>
</html>
```

Step 13 : Now in the design mode of the MyWeb1.aspx page, add the same control but add only one hyperlink control and set its NavigateUrl property to AddWeb2.aspx then click ok.

The source code of the MyWeb1.aspx page is as follows :

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="MyWeb1.aspx.cs" Inherits="MyWeb1" %>

<html>
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
           
        <asp:Label ID="Label1" runat="server" Font-
Bold="True" Font-Names="Arial Black"
            Font-Size="Large"
            Text="Thank you for clicking.This is My First
Webpage.....">
        </asp:Label>
        <br />
        <br />

         
         
     
                 
        <asp:Label ID="Label2" runat="server"
Text="SiteMap:">
        </asp:Label>
               
        <asp:SiteMapPath ID="SiteMapPath1"
runat="server">
        </asp:SiteMapPath>
        <br />
        <br />

         
         
    
        <asp:HyperLink ID="HyperLink1" runat="server"
NavigateUrl="~/AddWeb2.aspx">click here to go to
mywebpage2</asp:HyperLink>
    </div>
    </form>
</body>
</html>
```

Step 14 : Now in the design mode of the AddWeb2.aspx page, add the same controls as in myweb1.aspx and set the NavigateUrl property of the hyperlink as Default2.aspx then click ok.

The source code of AddWeb2.aspx is as follows :

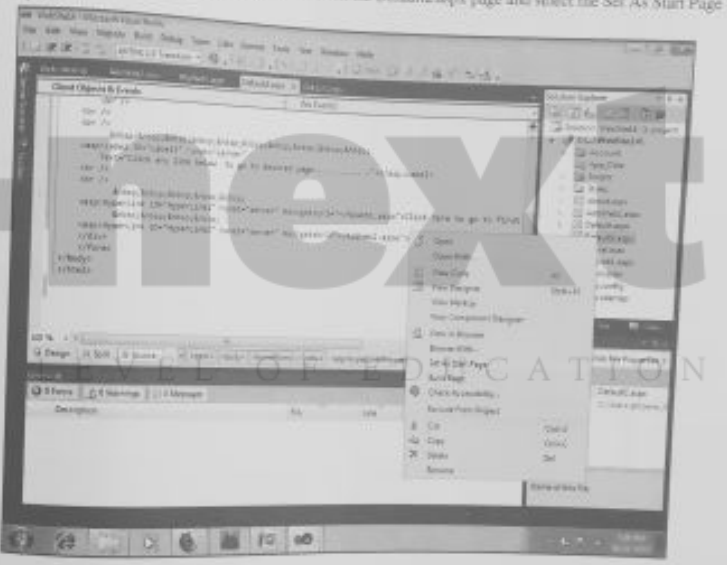
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="AddWeb2.aspx.cs" Inherits="AddWeb2" %>

<html>
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Label ID="Label1" runat="server" Font-
Bold="True" Font-Names="Arial Black"
            Font-Size="Large"
            Text="Thanx for Clicking.This is my Second
webpage by using SiteMapPath control.....">
        </asp:Label>
    </div>

<p>        &
nbsp;     
         
        <asp:Label ID="Label2" runat="server"
Text="SiteMap:"></asp:Label>
            
        <asp:SiteMapPath ID="SiteMapPath1"
runat="server">
```



```
        </asp:SiteMapPath>
    </p>
    <p> </p>

<p>        &
nbsp;     
         
         
     
```

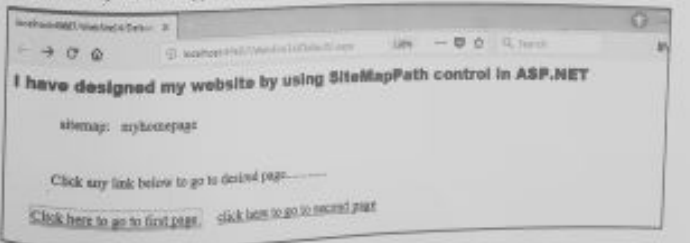
```
           
    <asp:HyperLink ID="HyperLink1" runat="server"
NavigateUrl="~/Default2.aspx">Click here to go to Home
page</asp:HyperLink> 
    </p>
    </form>
</body>
</html>
```

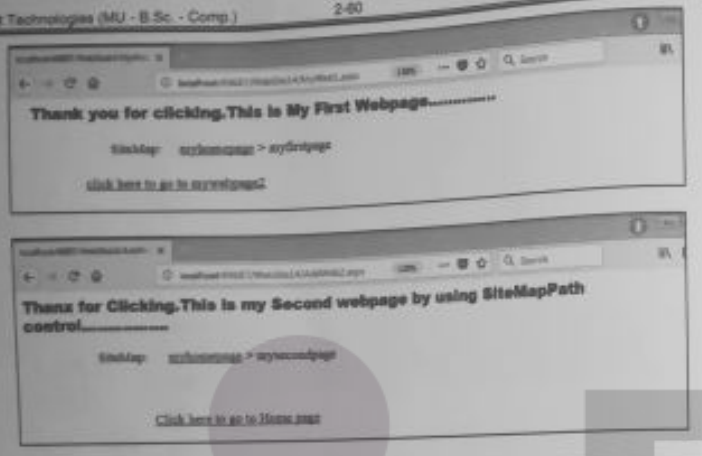
Step 15 : Now in the Solution Explorer window, right click on the Default2.aspx page and select the Set As Start Page option.



Step 16 : Press the F5 key to run the application.

Output





Syllabus Topic : TreeView Control

2.6.4 TreeView Control

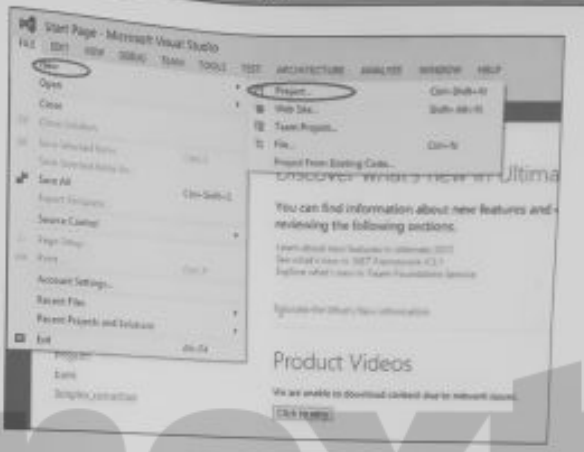
Q. How to create TreeView ? **(4 Marks)**

- The TreeView Web control is used to show the hierarchical data in a tree structure.
- A TreeView is a group of TreeNode objects.
- The contents of the TreeView control can be stated directly in the control :
  1. XML file
  2. Web.sitemap file
  3. Database table
- Following is Constructure of TreeViewControl :

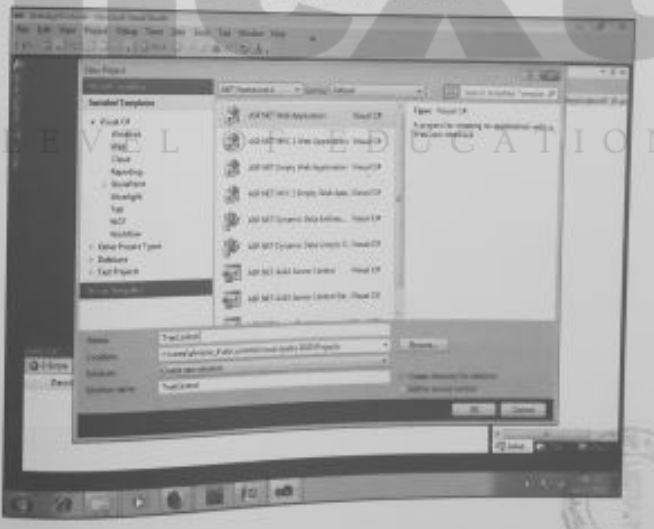
  Treeview() which is used to initialize new object of Treeview class.

☞ **Steps to create a TreeView Control in ASP.NET**

Step 1 : Open Microsoft Visual Studio then select "File" -> "New" -> "Project..."

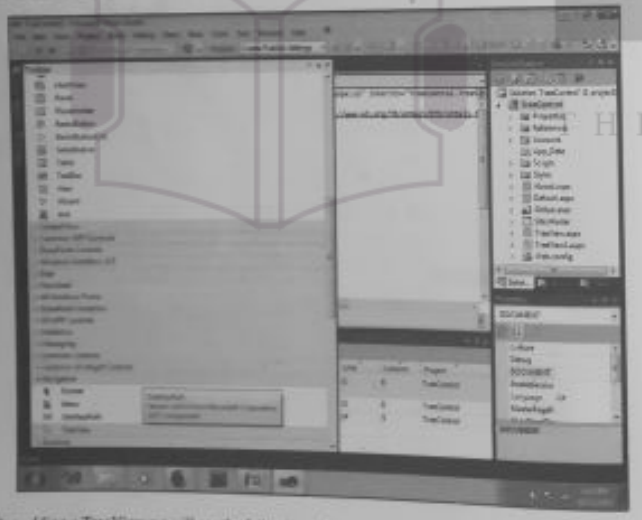


Step 2 : Add an ASP.NET Web Forms Application and give it the name TreeControl.



Step 3 : Open Solution Explorer then add a Webform and give it a name such as "TreeView1.aspx".

Step 4 : Now drag and drop a TreeViewControl from the ToolBox.



And after adding a TreeView we will see the following code :

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="TreeView1.aspx.cs"
Inherits="TreeControl.TreeView1" %>
```



```
<html>
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <d<asp:TreeView ID="TreeView1" runat="server"></asp:TreeView>
  js>
  </div>
  </form>
</body>
</html>
```

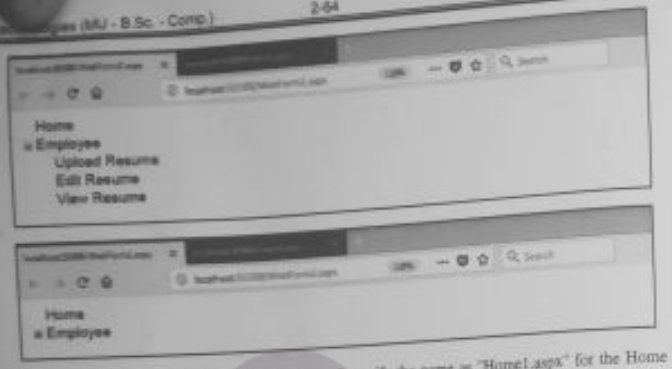
Create Tree view

Step 5 : Write the following code inside the <Nodes></Nodes> element and create Nodes of the TreeView in the "TreeView1.aspx" page as in the following.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="TreeControl.WebForm2" %>
<html>
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    <div style="font-family: Arial">
      <asp:TreeView runat="server" ID="TreeView1">
        <Nodes>
          <asp:TreeNode Text="Home" NavigateUrl="~/Home1.aspx" Target="_blank" />
          <asp:TreeNode Text="Employee" NavigateUrl="~/Emp.aspx" Target="_blank">
            <asp:TreeNode Text="Upload Resume" NavigateUrl="~/Upload_Resume1.aspx" Target="_blank" />
            <asp:TreeNode Text="Edit Resume" NavigateUrl="~/Edit_Resume1.aspx" Target="_blank" />
            <asp:TreeNode Text="View Resume" NavigateUrl="~/ViewResume1.aspx" Target="_blank" />
          </asp:TreeNode>

        </Nodes>
      </asp:TreeView>
    </div>
  </form>
</body>
</html>
```

Create tree node which content hyperlink to different pages like Home1.aspx etc.

Output



Now before going further we add one new webform and specify the name as "Home1.aspx" for the Home link of the TreeView.

Home1.aspx.cs

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Home1.aspx.cs" Inherits="TreeControl.Home1" %>

<html>
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
      <asp:Label ID="HomePage" runat="server" Text="Welcome To Home Page" BackColor="Yellow"
ForeColor="#000000"></asp:Label>
  </div>
  </form>
</body>
</html>
```

Output

Syllabus Topic : Menu Control

2.6.5 Menu Control

Q. Write note on menu control. (4 Marks)

- The Menu control permits the multiple display options adding a static display.
- It also provides dynamic display where part of the menu appear according to position of mouse on screen.
- This control also provides a combination of static and dynamic display modes.
- When we use menu control with root items, static mode is used.
- For menu control with child menu items, we can use dynamic mode.
- We can made setting of ASP.NET Menu control in the designer with static links to our pages or we can bind

menu control automatically to a hierarchical data source such as an XmlDataSource or a SiteMapDataSource control.

- Its properties like BackColor, ForeColor, BorderColor, BorderStyle, BorderWidth, Height etc. are applied using style properties of <table>, < tr>, <td> tags.

Following are some important properties that are very useful

Properties of Menu Control	
DataSourceID	This property used to state the data source to be used.
Text	This property used to state the text to display in the menu.
Tooltip	This property used to state the tooltip of the menu item when we fire mouse over.
Value	This property used to states the nondisplayed value (usually unique id to use in server side events)
NavigateUrl	This property is used to state the target location to shift the user when menu item is clicked. If not set we can use MenuItemClick event to decide what to do.
Target	If NavigationUrl property is set, it indicates where to open the target location (in new window or same window).
Selectable	If false, this item can't be selected.
Image Url	This property is used to state the image that appears next to the menu item.
Image Tool Tip	This property is used to state the tooltip text to display for image next to the item.
Pop Out Image Url	This property is used to state the image that is displayed right to the menu item when it has some subitems.
Target	NavigationUrl property is set to state target location to open the file
Static Menu Style	This property is used to Set the style of the parent box which contains all menu items .
Dynamic Menu Style	This property is used to Set the style of the parent box which contains dynamic menu items.
Static Menu Item Style	This property is used to Set the style of the every static menu items.
Dynamic Menu Item Style	This property is used to Set the style of the every dynamic menu item.
Static Selected Style	This property is used to the style of the selected static items.
Dynamic Selected Style	This property is used to set the style of the selected dynamic items.
Static Hover Style	This property is used to set the mouse overing style of the static items.
Dynamic Hover Style	This property is used to Set the mouse overing style of the dynamic items and subitems.
Target	NavigationUrl property is set to state target location to open the file

Program 2.6.1

Create a program using menu control to show different menus such as Home,About,Gallary etc.

Solution :

Program using menu control to show different menus such as Home,About,Gallary etc.

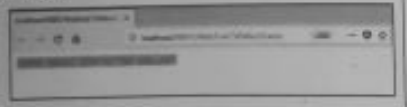
Default2.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default2.aspx.cs" Inherits="Default2" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
```

Create menu control

```
<div>
<asp:Menu ID="Menu1" runat="server"
BackColor="#B5C7DE" DynamicHorizontalOffset="2"

Font-Names="Verdana" Font-Size="0.8em"

ForeColor="#284E98" Orientation="Horizontal"

StaticSubMenuIndent="10px">

<Items>
<asp:MenuItem Text="HOME" Value="home"
NavigateUrl="~/home.aspx">
</asp:MenuItem>
<asp:MenuItem Text="ABOUT" Value="about"
NavigateUrl="~/about1.aspx">
</asp:MenuItem>
<asp:MenuItem Text="CONTACTNO" Value="contact"
NavigateUrl="~/contact1.aspx">
</asp:MenuItem>
<asp:MenuItem Text="GALLARY" Value="contact"
NavigateUrl="~/gallary.aspx">
</asp:MenuItem>
</Items>

</asp:Menu>

</div>
</form>
</body>
</html>
```

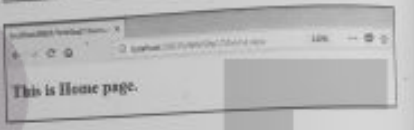
Output



Home.aspx

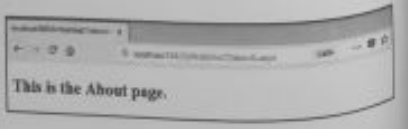
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="home.aspx.cs" Inherits="home" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <h2>This is Home page.</h2>
  </div>
  </form>
</body>
</html>
```



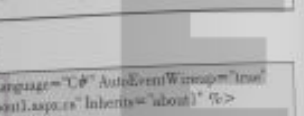
about1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="about1.aspx.cs" Inherits="about1" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <h2>This is the About page.</h2>
  </div>
  </form>
</body>
</html>
```



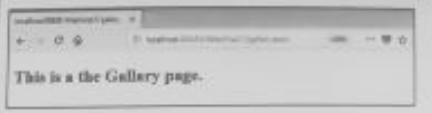
Contact1.aspx

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <h2>This is the contact page.</h2>
  </div>
  </form>
</body>
</html>
```



gallary.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="gallary.aspx.cs" Inherits="gallary" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>
  <h2>This is a the  Gallary page.</h2>
  </div>
  </form>
</body>
</html>
```



Review Questions

Q. 1 Write short note on : List Controls. **(Refer Section 2.1.3)** **(4 Marks)**

Q. 2 State the page life cycle with diagram. **(Refer Section 2.1.6)** **(4 Marks)**

Q. 3 Write short note on view state. **(Refer Section 2.2.1)** **(4 Marks)**

Q. 4 Explain cookies in details. **(Refer Section 2.2.4)** **(4 Marks)**

Q. 5 Explain session in detail. **(Refer Section 2.2.5)** **(4 Marks)**

Q. 6 Explain the application state in detail ? **(Refer Section 2.2.7)** **(4 Marks)**

Q. 7 What is use of validation controls? List the different validation controls. **(Refer Section 2.3.1)** **(4 Marks)**

Q. 8 List all HTML5 validation controls and explain any 2 HTML validation controls in details. **(Refer Section 2.3.4)** **(4 Marks)**

Q. 9 List all rich controls and explain any one rich control in detail. **(Refer Section 2.4)** **(4 Marks)**

Q. 10 Write short note on Multiview control. **(Refer Section 2.4.3)** **(4 Marks)**

Q. 11 Explain themes and master pages in detail. **(Refer Section 2.5)** **(4 Marks)**

Q. 12 What is concept of master pages and content pages and how to connect them? **(Refer Section 2.5.4)** **(4 Marks)**

Q. 13 Explain siteMapPath control in detail. **(Refer Section 2.6.3)** **(4 Marks)**

Q. 14 Write note on menu control. **(Refer Section 2.6.5)** **(4 Marks)**

□□□

UNIT III

CHAPTER 3

ADO .Net

Syllabus Topics

ADO.NET : Data Provider Model, Direct Data Access - Creating a Connection, Select Command, DataReader, Disconnected Data Access.

Data Binding : Introduction, Single-Value Data Binding, Repeated-Value Data Binding, Data Source Controls - SqlDataSource.

Data Controls : GridView, DetailsView, FormView, Working with XML: XML Classes - XMLTextWriter, XMLTextReader.

Caching : When to Use Caching, Output Caching, Data Caching.

LINQ : Understanding LINQ, LINQ Basics.

ASP.NET AJAX : ScriptManager, Partial Refreshes, Progress Notification, Timed Refreshes.

Syllabus Topic : ADO .Net

3.1 Introduction to ADO.NET

3.1.1 Introduction

- Whenever we create a software or website, it is not possible to implement it using only programming languages. Regarding any software or website, there is always important data which we have to store permanently.
- Storing data permanently is not the facility given by any programming language. For this purpose we have to use database. That means software or websites is developed by the combination of programming language and database.
- ADO.Net is a model used by the .Net framework to communicate with database for retrieving and storing data with the help of various built in classes.
- As you know that there are several different types of databases available. Some of them are listed below :
  1. Microsoft SQL Server
  2. Microsoft Access
  3. Oracle
  4. Borland Interbase
  5. IBM DB2, etc

  In this chapter will use SQL Server.

Syllabus Topic : Data Provider Model

3.1.2 Data Provider Model

Q. Explain data provider model in brief. **(4 Marks)**

- The data provider is used for connecting to database, executing queries, and retrieving results.
- Those results are processed and stored in a DataSet in order to be avail to the user as needed.
- The data providers are lightweight components which create a minimal layer between the data source and code and increases performance without sacrificing the functionality.

- Following Data Providers are used in ADO.NET.
  (i) The Microsoft SQL Server
  (ii) OLEDB
- In ADO .Net all the functionalities are implemented with the help of set of core classes. These classes are divided into two groups :
  (i) Classes used to contain and manage data : For example DataSet, DataTable, DataRow, and DataRelation.
  (ii) Classes used to connect to a particular data source : For example Connection, Command, and DataReader.
- The data container classes are in general totally generic. It makes no difference that from where we are extracting the data, it is stored in the similar data container: the specialized DataSet class.
- Dataset is playing similar role just like an array or collection i.e. container or package for the data. But the most important difference is that, the DataSet is specially build for relational data that means it understands concepts such as rows, fields, and table relationships.
- The second group of classes is usually comes in various flavors. All the groups of data interaction classes are known as ADO.NET data providers. Data providers are customized in such a way that all of them use best-performing way for the interaction with their data sources.
- For example, the SQL Server data provider is specially customized to work with SQL Server. Internally, tabular data stream (TDS) protocol of SQL server is used by it for communication purpose. Hence there is guarantee of best performance. For database Oracle the Oracle data provider can be used.
- There is personal prefix of all these data providers for naming the classes. Thus in the SQL Server provider there are classes like SqlConnection and SqlCommand while in the Oracle provider classes named OracleConnection and OracleCommand are present.
- The internal behaviour of these classes is quite different as they have to connect to different databases with the help of different low-level protocols. On the other hand the external display of these classes is very much similar and also provides similar set of basic methods because same common interfaces are implemented by them.
- Because of all these features, the application is protected from the complexity of different standards and SQL Server provider and Oracle provider in the same way. In fact, it is also possible to translate a block of code written for interaction with a SQL Server database into a block of Oracle-specific code just by changing the name of class in the code.
- All the classes regarding database connectivity are categorized under three namespaces

Sr. No.	Namespace	Purpose
1.	System.Data.SqlClient	Provides classes used to connect Microsoft SQL Server database and execute commands like SqlConnection and SqlCommand.
2.	System.Data.SqlTypes	Provides structures for SQL Server–specific data types like SqlMoney and SqlDateTime. These types can be used along with SQL Server data types without need of conversion into standard .NET equivalents like System.Decimal or System.DateTime.
3.	System.Data	Provides fundamental classes with the core ADO.NET functionality. These classes are DataSet and DataRelation which can manipulate structured relational data. These classes are not dependent upon any specific type of database or the connection method.

- Now we are going to see different object of ADO .Net. They are explained below.

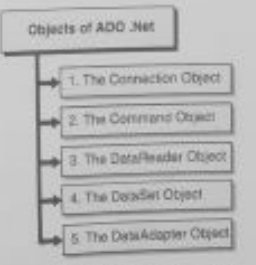


Fig. C3.1 : Objects of ADO .Net

Q. What are the objects of ADO.net? Explain any one. (4 Marks)

1. The Connection Object

- To communicate or interact with a database, you must have a connection with a database.
- The connection helps out to recognize the database server, the database name, user name, password, and other parameters that are required for connecting to the database.
- A connection object is used by command to know on which database to execute the command.

2. The Command Object

- Interacting with a database does not mean only creating as connection; it means you must state the actions that you want to occur. This can be done with the help of command object. The command object can be used to send SQL statements to the database.
- A command object makes use of connection object to tell which database to communicate with. The command object can be used alone, to execute a command directly.

3. The DataReader Object

- In ADO.NET the DataReader Object is a stream-based, read-only, forward-only retrieval of query results from the Data Source, which never update the data.
- A connection oriented data access to the data sources is provided by DataReader. A Connection Object can contain only one DataReader at a time.
- While data is being accessed, the connection in the DataReader remains open and cannot be used for any other purpose. While starting reading through a DataReader it should always be open and positioned price to the first record. The DataReader provides read() method to read the records from it and it always moves forward to a next record if any row exist.

4. The DataSet Object

- The DataSet is used to store the data of database in application. It represents a collection of data retrieved from the Data Source.
- DataSet is tabular representation of data. Tabular representation means it represents data into row and column format.
- We can use Dataset in combination with DataAdapter class. The DataSet object work in disconnected database architecture. It provides a better advantage over DataReader , because the DataReader is working only with the connection oriented database architecture.
- The Dataset contains the copy of the data requested by client. The Dataset can contain more than one tables at a time. We can set up relations between these tables within the DataSet.
- The **DataSet** may contain multiple tables represented by DataTable objects.

5. The DataAdapter Object

- DataAdapter provides the communication between the Dataset and the Datasource.
- DataAdapter can be used in combination with the DataSet Object so that the two objects enable both data retrieval and data manipulation capabilities.
- To retrieve data from a data source the DataAdapter is used and it is also used to manipulates tables which are present in a DataSet.
- The DataAdapter is also used to reflect changes made to the DataSet back to the data source.
- Connection object is used by DataAdapter for connecting to a data source, and Command object is used by DataAdapter to retrieve data and reflect changes to the data source.
- The SelectCommand property of the DataAdapter is used to retrieve data from the data source. The InsertCommand, UpdateCommand, and DeleteCommand properties of the DataAdapter are used to manage updates to the data in the data source according to modifications made to the data in the DataSet.
- The Fill() method of the DataAdapter is used to populate a DataSet with the results of the SelectCommand of the DataAdapter. The arguments of Fill() method are - DataSet to be populated, and a DataTable object, or the name of the DataTable to be filled with the rows returned from the SelectCommand.
- DataReader object is used by Fill() method implicitly for returning the column names and types that are used to create the tables in the DataSet, and the data to manipulate the rows of the tables in the DataSet.
- If Tables and columns are not exist previously then they are created; otherwise existing DataSet schema is used by Fill() method.

- Column types are created as .NET Framework types according to the tables in Data Type Mappings in ADO.NET.
- Primary keys are created if they exist in the data source. If Fill() finds that a primary key exists for a table, it will overwrite data in the DataSet with data from the data source for rows.
- If primary key is not found then the data is appended to the tables in the DataSet. Fill() method uses any mappings that may exist when you manipulate the DataSet.

Syllabus Topic : Direct Data Access

3.1.3 Direct Data Access

Q. Explain Direct Data Access in detail. (10 Marks)

- The Direct Data Access is considered as the most straightforward way to communicate with a database.
- This option gives complete control of building and executing SQL commands to user. All the database operations like insert, update, and delete information can be performed easily.
- While using Direct Data Access, there is no need to store copy of the information in memory. Rather for a brief period of time user work with it and close the connection after finishing the work.
- This option is completely very thus another option of disconnected data access, in which a copy of data is kept in the Dataset object. In disconnected data access, user can work even after closing the connection.
- The Direct Data Access models is basically well suited for ASP.NET web pages in which there is no requirement of keeping a copy of the data in memory for long periods of time.
- We have to keep it in mind that the page is requested and quickly shut down when the response is returned to the user. It indicates that the lifetime of page is only of literally few seconds.
- To access the data with simple data access, we have to follow the given steps :
  1. First Create Connection, Command, and DataReader objects.
  2. Retrieve information from the database with the help of DataReader, and show on a web form.
  3. Disconnect connection by closing it.
  4. Send webpage to user, at this point the database is no longer connected and all the ADO.NET objects are destroyed.
- To add or update information, follow these steps
  1. Create new Connection and Command objects.
  2. Execute the Command by using SQL statement.

Fig. 3.1.1 illustrates how the ADO .NET objects communicate to make direct data access.

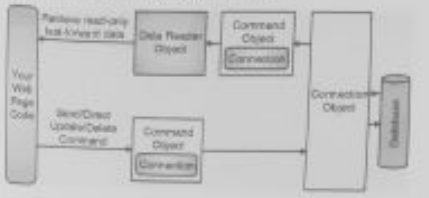


Fig. 3.1.1 : Direct Access with ADO.Net

- For communication with database using ADO.Net, we have to include the important ADO .Net namespaces.
- Here we consider SQL Server provider then we have to import following namespaces :
  - ii) Imports System.Data
  - iii) Imports System.Data.SqlClient

Syllabus Topic : Creating a Connection

3.1.3.1 Creating Connection

Q. Explain how to create a connection in ADO.NET? (6 Marks)

- While interacting with database the first thing is to create a connection. The connection informs the rest of the ADO.NET code which database it is going to be communicated. It handles all of the low level logics related to the particular database protocols.
- Working with connection in ADO.Net is very easy; you just have to understand the connection. When there is single user then you don't have take care of connection but when there are multiple users for a single database you have to take care of it because wrong information makes lot of errors while accessing an application.

Creating a SqlConnection Object

- Like any other C# object SqlConnection is one of the object .You can declare and instantiate the SqlConnection at the same time, as shown below

```
SqlConnection connect = new SqlConnection(
    "Data Source=(local);Initial
Catalog=Northwind;Integrated Security=SSPI");
```

- The above instantiated SqlConnection object uses a constructor which has only one parameter of string data type and this parameter is known as connection string. The following Table 3.1.1 illustrates common parts of a connection string.

Sr. No.	Connection String Parameter Name	Description
1.	Data Source	It is used to Identify the server. It could be local machine, machine domain name, or IP Address.
2.	Initial Catalog	It contains the database name.
3.	Integrated Security	This Parameter is used to set to SSPI for making connection with user's Windows login.
4.	User ID	Name of user configured in SQL Server.
5.	Password	Password matching to SQL Server User ID.

- When you are developing your project using single computer/machine then Integrated Security is secure. You can also provide security based on a SQL Server User ID and the password.
- The following code shows a connection string, using the User ID and Password parameters.

```
SqlConnection connect = new SqlConnection(
        "Data Source=DatabaseServer;Initial
Catalog=Northwind;
                User
ID=YourUserID;Password=YourPassword");
```

- In the above code snippet observe that how the Data Source is set to the DatabaseServer to specify that you can identify a database placed on a different machines, over LAN, or over the Internet. Additionally, User ID and Password replaces the Integrated Security parameter.

Using a SqlConnection

- The aim of creating a SqlConnection object is to allow users to use other ADO.NET code to work with a database. Other ADO.NET objects, such as a SqlCommand and a SqlDataAdapter take a connection object as a parameter.
- The following operations occur during the lifetime of a SqlConnection :

1. Instantiate the SqlConnection.
2. Open the connection.
3. Pass the connection as a parameter to other ADO.NET objects.
4. Perform database operations with the other ADO.NET objects.
5. Close the connection.

- To instantiate the SqlConnection object, the subsequent steps are shown in following code snippet:

Using a SqlConnection

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

class SqlConnectionDemo
{
  static void Main()
  {
    SqlConnection connect = new SqlConnection(
      "Data Source=(local);Initial
Catalog=Database1;Integrated Security=SSPI");

    SqlDataReader datareader = null;

    try
    {

      connect.Open();

SqlCommand command = new SqlCommand("select * from
employee", connect);

      datareader = command.ExecuteReader();

        while (datareader.Read())
        {
          Console.WriteLine(datareader[0]);
        }
    }
    finally
    {
```

Connectivity to database


```
    if datareader != null)
      {
        datareader.Close();
      }

    if (connect != null)
        {
          connect.Close();
        }
    }
}
```

- As shown in the above code snippet, you can open a connection by calling the Open() method of the SqlConnection instance, connect. If you try to perform any operations on a connection that was not opened, the it will produce an exception. So, you have to open the connection before performing any operation on it.
- Before going to use the SqlCommand object, you have to inform the ADO.NET code that which connection it desires. In above code snippet, we have set the second parameter to the SqlCommand object with the SqlConnection object, connect. After passing the SqlConnection object to SqlCommand, the SqlCommand will use that connection for performing the operations.
- The code that uses the connection is a SqlCommand object, which executes a query on the employee table. The result set is returned as a SqlDataReader and the while loop reads the first column from each row of the result set, which is the EmployeeID column.
- When you have done all the operations then you must close the connection. If you do not close the connection then it will affect the performance and scalability of your application. In above code snippet the Close ( ) method is called in a finally block to close the connection.
- Finally block will help you to ensure that a certain part of code will be executed, regardless of whether or not an exception is generated.

Syllabus Topic : Select Command

3.1.3.2 Select Command

- A SqlCommand object permits you to state what type of communication/interaction you want to perform with a database. For example, you can do select, insert, modify, and delete commands on database table.

```
SqlCommand cmd = new SqlCommand("select ename
from employee", connect);
```

- The above line indicates the instantiation of a SqlCommand object. The SqlCommand takes a string parameter which contains the command you desire to execute and a reference to a SqlConnection object.

Querying Data

- To retrieve a data set for viewing to users you make use of SQL select command. The ExecuteReader() method, which returns a SqlDataReader object, which can be used with SqlCommand object to use SQL select command.
- The following example shows how to use the SqlCommand object to acquire a SqlDataReader object

Example

```
SqlCommand command = new SqlCommand("select ename
from employee", connect);
```

instantiate a new command with query and connection

```
SqlDataReader datareader = command.ExecuteReader();
```

Call Execute reader to get query results

Inserting Data

The ExecuteNonQuery () method is used to add data into a database. This method belongs to SqlCommand object. The below code snipper illustrates how to insert data into a database table

Example

```
string instring = @" insert into employee (ename, salary)
values ('Kunal', 30000)";
```

Preparing command string for performing operation

```
SqlCommand command = new SqlCommand(instring,
connect);
```

To send command call ExecuteNonQuery

```
command.ExecuteNonQuery();
```

- In the above code notice that inside the insert command, we explicitly specified the columns ename and salary. The employee table has a primary key field named emp_ID. We are skipping this column from the query because SQL Server will add this field itself.

Trying to add a value to a primary key field, such as emp_ID, will produce an exception.

Updating Data

- To update data in database the ExecuteNonQuery method is used. The below code illustrates how to update data

Example

```
string updated_data = @" update employee set ename =
'Kunal B.' where salary = 30000";
SqlCommand command = new
SqlCommand(updated_data);
command.Connection = connect;
command.ExecuteNonQuery();
```

- Observe that we have used a variable to store the SQL command. Now we have used a different SqlCommand constructor that accepts only the command.

Deleting Data

- The ExecuteNonQuery method is also used to delete data from database. The below example illustrates how to delete a record from a database

Example

```
string delete_data = @" delete from employee where
ename = Kunal B.";
SqlCommand command = new SqlCommand();
command.CommandText = delete_data;
command.Connection = connect();
command.ExecuteNonQuery();
```

Getting Single values

- Sometimes you require just a single value from a database. It could be a count, sum, average, or other aggregated value from a data set. Performing an ExecuteReader and computing the result in your code is not the well-organized way to do this. The best option is to let the database perform the work and return just the single value that you need.
- The below example illustrates how to do this with the ExecuteScalar method

Example

```
SqlCommand command = new SqlCommand("select
count(*) from employee", connect);
int total = (int)command. ExecuteScalar ();
```

- The query given in the SqlCommand constructor gets the count of all records from the employee table. This query will return only one value. The ExecuteScalar method is used to return this single value. Since the return type of ExecuteScalar is type object so we have to convert it. To convert value to int we use a cast operator.

Putting it All Together

- To know each operation in detail we are using the small code snippets. Now we can put it all together to make a single view of all operations. This can be demonstrated as follow

```
using System;
using System.Data;
using System.Data.SqlClient;
class SqlCommandDemo1
{
SqlConnection connect;

  public SqlCommandDemo1()
{
      connect = new SqlConnection(
"Data Source=(local);Initial Catalog=database1;Integrated
Security=SSPI");
  }
   static void Main()
   {
     SqlCommandDemo1 cmddemo = new
SqlCommandDemo1();

     Console.WriteLine("Employee table Before Insertion");

     cmddemo.ReadData();

     cmddemo.Insertdata();
     Console.WriteLine("Employee table After Insertion");

     cmddemo.ReadData();
      cmddemo.UpdateData();
      Console.WriteLine("Employee table After Updation");
      cmddemo.ReadData();
      cmddemo.DeleteData();
```



```
     Console.WriteLine("Categories After Delete");

     cmddemo.ReadData();

     int numrec = cmddemo.GetNumberOfRecords();

        Console.WriteLine("Number of Records: {0}",
numrec);
   }

   public void ReadData()
   {
     SqlDataReader datareader = null;

     try
     {
       connect.Open();

       SqlCommand command = new SqlCommand("select
ename from employee", connect);

       datareader = command.ExecuteReader();
       while (datareader.Read())
       {
         Console.WriteLine(datareader[0]);
       }
     }
     finally
     {

       if (datareader != null)
       {
         datareader.Close();
       }

       if (connect != null)
       {
         connect.Close();
       }
     }
   }
   public void Insertdata()
   {
     try
     {
       conn.Open();

       string instring = @"insert into employee (ename,
salary) values ('Sadashiv patil', 30000)";

       SqlCommand command = new
SqlCommand(instring, connect);

       command.ExecuteNonQuery();
     }
     finally
     {
       if (connect != null)
       {
         connect.Close();
       }
     }
   }
   public void UpdateData()
   {
     try
     {

       connect.Open();

       string updated_data = @" update employee set
ename = Rakesh Jadhav'where salary = 30000";

       SqlCommand command = new
SqlCommand(updated_data);
       command.Connection = connect;
       command.ExecuteNonQuery();
     }
     finally
     {
```

```
    if (connect != null)
    {
      connect.Close();
    }
  }
}

  public void DeleteData()
{
  try
  {
    connect.Open();

    string deleted_data = @"
      delete from employee where ename = 'Rakesh'";

    SqlCommand command = new SqlCommand();

    command.CommandText = deleted_data;

    command.Connection = connect;

      command.ExecuteNonQuery();
  }
  finally
  {
    if (connect != null)
    {
      connect.Close();
    }
  }
}

  public int GetNumberOfRecords()
{
  int count = -1;

  try
  {
```

```
    connect.Open();

    SqlCommand command = new SqlCommand("select
count(*) from employee", connect);

    count = (int) command.ExecuteScalar();
  }
  finally
  {
    if (connect != null)
    {
      connect.Close();
    }
  }
  return count;
}
}
```

Syllabus Topic : Data Reader

3.1.3.3 SqlDataReader

Q. Write note on SQLDAta Reader (4 Marks)

- To read a data in an efficient way a SqlDataReader is useful. SqlDataReader is used for only reading the data you cannot use it for writing a data.
- SqlDataReader can read data in sequential manner i.e. when you read some data and go ahead then you can't come back to again read that data.
- The forward only design of the SqlDataReader is what facilitates it to be quick. It doesn't have overhead related to traversing the data or writing it back to the data source. The SqlDataReader is best option if you want to read collection of data in faster way for only one time.

Creating a SqlDataReader Object

- Creating an object of a SqlDataReader is a quit different than the manner in which you instantiate other objects of ADO.Net. You have to call the ExecuteReader method on a command object. It is shown as follow

```
SqlDataReader datareader = command.ExecuteReader();
```

- The ExecuteReader method of the SqlCommand object, command, returns a SqlDataReader instance

Reading Data

- As discussed earlier the SqlDataReader returns data through an in order stream. That means when you read a single row then you cannot read the previous row. To read that row, you would have to create a new object of the SqlDataReader and read via the data stream.
- One way of reading from the data stream is to use a while loop as given below

```
while (datareader.Read())
{
    string name = (string) datareader["ename"];
    int sal = (int) datareader["salary"];
    Console.Write("{0,-25}", name);
    Console.Write("{0,-20}", sal);
    Console.WriteLine();
}
```

Print out the results

- Observe the call to *Read* on the SqlDataReader, *datareader*, in the while loop condition in the above code. The return value of *Read* is of type Boolean. IT returns true if there are more records to read and returns false when there is no more records available.
- In the above code snippet you extracted two columns from the employee table having name ename, salary. After reading data you can perform any operation on it like printing them to console or modifying them etc.

Finishing Up

- One important thing is that always remember to close your SqlDataReader, in the same way as you close the SqlConnection. Place your data access code in a try block and close operation in the finally block. This can be illustrated as follow

```
try
{
  ...
}
finally
{
    if (datareader!= null)
    {
        datareader.Close();
    }
}
```

- The above code snippet makes sure that the SqlDataReader is not null. After the code knows that a good instance of the SqlDataReader exists, it can close it. Following program combines all the code snippets given above.

Using the SqlDataReader

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

namespace datareed
{
    class ReaderDemo1
    {
        static void Main()
        {
            ReaderDemo1 rd1 = new ReaderDemo1();
            Rd1.SimpledataRead();
        }
        public void SimpledataRead()
        {
            SqlDataReader datareader = null;
                SqlConnection connect = new
SqlConnection(
"Data Source=(local);Initial Catalog=Northwind;Integrated
Security=SSPI");

            SqlCommand command1 = new
SqlCommand(
                "select * from employee", connect);

            try
            {

                connect.Open();

                datareader = command1.ExecuteReader();

                Console.WriteLine("Ename
Salary");

                while (datareader.Read())
                {
```

```
            string name = (string)datareader["ename"];
            int sal = (int) datareader["salary"];
                Console.Write("{0,-25}", name);
                Console.Write("{0,-20}", sal);
                Console.WriteLine();
            }
        }
        finally
        {
            if (datareader != null)
            {
                datareader.Close();
            }
            if (connect != null)
            {
                connect.Close();
            }
        }
    }
}
```

Syllabus Topic : Disconnected Data Access

3.1.3.4 Disconnected Data Access

Q. Explain disconnected data access. (4 Marks)

- In previous section we have seen fully connected mode of operation for interacting with a data source by using the SqlCommand object. This section shows how to do disconnected data access using the DataSet and SqlDataAdapter objects.
- A DataSet is considered as an in-memory data store which can store numerous tables. Th DataSets are used to hold the data.
- They do not interact with data source. The SqlDataAdapter is used to handle the connection with the data source and manage the disconnected behavior.
- The most important thing is that the connection is opened by the SqlDataAdapter when needed and closed as soon as the operation is accomplished.
- For Example, to fill data in DataSet, following steps are performed by SqlDataAdapter
  1. Open connection
  2. Retrieve data into DataSet
  3. Close connection
- To update data source with DataSet changes following steps are performed.
  1. Open connection
  2. Write changes from DataSet to data source
  3. Close connection
- After filling data set the data source connection can be closed before update operation. Even after closing the connection we can read and write data with the DataSet as per requirement. This is the standard mechanism of disconnected data architecture. The application is becomes more scalable as connection is held by application only when required.
- There are number of scenario when this architecture seem to be very useful: for example website making without network connectivity. In another example consider sales persons, who require customer data at the start of the day, they will require to sync up with the main database for getting the latest information available. Throughout the day, they will make necessary changes to existing customer data, include new customers, and enter new orders. This is possible as in the database of their region other people won't be making changes. At the end of the day, the network can be connected by the sales person to update the changes.
- Another scenario is creation of more scalable websites. With a SqlDataReader, we have to go to database whenever we show the page. Hence there is requirement of new connection for each page load. When there is increase in number of users, the scalability gets affected.
- The solution on it is to use DataSet which can be stored in cache once updated. For every request, this data of cache is retrieved which avoid the trip to the database and makes application very efficient.

Creating a DataSet Object

Syntax

```
DataSet CustDS = new DataSet();
```

- The DataSet constructor doesn't require parameters. However there is one overload that accepts a string for the name of the DataSet, which is used if you were to serialize the data to XML. Since that isn't a requirement for this example, I left it out. Right now, the DataSet is empty and you need a SqlDataAdapter to load it.

Creating A SqlDataAdapter

- To read/write data, the SQL commands and connection object are held by the SqlDataAdapter. It is initialized with a SQL select statement and connection object

```
SqlDataAdapter CustDA = new SqlDataAdapter("select CustID, CompName from Customers", conn);
```

- This code will generate a new SqlDataAdapter, CustDA. The SQL select statement is used to specify that what data should be stored in DataSet.
- Here it is considered as the connection object conn is already instantiated. Now SqlDataAdapter is used to open as well as close the connection.
- The connection object, conn, should have already been instantiated, but not opened. It is the SqlDataAdapter's responsibility to open and close the connection during Fill and Update method calls.
- Two option are available to add insert, update, and delete operations : using SqlDataAdapter properties or with a SqlCommandBuilder. Here we are going to use SqlCommandBuilder

```
SqlCommandBuilder cmd_Bldr = new SqlCommandBuilder(CustDA);
```

- The SqlCommandBuilder is instantiated with a single constructor parameter of the SqlDataAdapter.daCustomers, instance.
- To the constructor of SqlCommandBuilder, we have passed CustDA (SqlDataAdapter) instance. The SqlCommandBuilder will read the SQL select statement and infer the insert, update, and delete commands. SqlCommandBuilder is limited to work with single table.

Filling the DataSet

- After generating the DataSet and SqlDataAdapter instances, we have to fill the Dataset. The fill method of SqlDataAdapter is used for this purpose.

```
CustDA.Fill(CustDS, "Customers");
```

- We can see that the fill method accepts two parameters, a DataSet and a table name. The DataSet is filled with data. The table is the one that will be created in the DataSet. It is a customized table name. It is just like nickname to the original table of data source used to refer in the application.

Using the DataSet

- A DataSet can be bind with both ASP.NET and Windows forms DataGrids. Her we ssign the DataSet to a Windows forms DataGrid:

```
dgCustomers.DataSource = CustDS;
dgCustomers.DataMember = "Customers";
```

Updating Changes

- Now after finishing the changes in data, the changes should be written back to the database :

```
CustDA.Update(CustDS, "Customers");
```

- The Update method, here, is invoked on the SqlDataAdapter instance that has filled the CustDSDataSet. The second parameter specifies the name of table from dataset which has modified data.

Putting it All Together

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Windows.Forms;
class DisconnectedDataform : Form
{
    private SqlConnection conn;
    private SqlDataAdapter CustDA;

    private DataSet CustDS;
    private DataGrid dgCustomers;
    private const string tableName = "Customers";
    public DisconnectedDataform()
    {
        InitData();
        dgCustomers = new DataGrid();
        dgCustomers.Location = new Point(5, 5);
dgCustomers.Size = new
Size(this.ClientRectangle.Size.Width - 10,
this.ClientRectangle.Height - 50);
        dgCustomers.DataSource = CustDS;
        dgCustomers.DataMember = tableName;
```

Fill dataset

Set up datagrid

```
        Button btnUpdate = new Button();
        btnUpdate.Text = "Update";
        btnUpdate.Location = new Point(
            this.ClientRectangle.Width/2 -
btnUpdate.Width/2,
            this.ClientRectangle.Height -
(btnUpdate.Height + 10));
        btnUpdate.Click += new
EventHandler(btnUpdateClicked);


        Controls.AddRange(new Control[] { dgCustomers,
btnUpdate });
    }

    public void InitData()
    {
        conn = new SqlConnection(

"Server=(local);DataBase=Northwind;Integrated
Security=SSPI");

        CustDS = new DataSet();

        CustDA = new SqlDataAdapter(
            "select CustomerID, CompanyName from
Customers", conn);

        SqlCommandBuilder cmdBldr = new
SqlCommandBuilder(CustDA);

        CustDA.Fill(CustDS, tableName);
    }

    public void btnUpdateClicked(object sender,
EventArgs e)
    {

        CustDA.Update(CustDS, tableName);
    }
    static void Main()
    {
        Application.Run(new DisconnectedDataForm());
    }
}
```

Make sure controls appear on form

Write changes back to DataBase

Syllabus Topic : Data Binding

3.2 Data Binding

Q. Explain data binding in detail. (4 Marks)

- There are number of controls provided by the Web Forms for supporting Data Binding. At design time or run time, these controls can be connected to ADO .Net components like DataView, DataSet, or a DataViewManager.

Data-Bound Controls

- A rich set of data-bound controls is provided by the ASP .Net. All these controls are easy to use. These controls are categorized in two groups: -single-item data-bound control and multi-item data-bound controls.

Fig. 3.2.1: Data-bound control

- In ASP.NET, these controls are created using a <asp:controlName> tag. Table 3.2.1 describes some common data-bound server-side controls.

Table 3.2.1

Control	ASP.NET Code	Description
DataGrid	<asp:DataGrid>	Database is displayed in a scrollable grid format and supports retrieval, add, update, sort, and paging
DataList	<asp:DataList>	Data is displayed in templates and style format

Control	ASP.NET Code	Description
ListBox	<asp:ListBox>	Display data in a list format
DropDownList	<asp:DropDownList>	Displays data in a drop down combo box format
CheckBox	<asp:CheckBox >	It can be connected to an item of the data source
CheckBoxList	<asp:CheckBoxList>	Display list of check boxes, which can be connected to the list of items in data source
Repeater	<asp:Repeater>	Displays a templated data-bound list
TextBox	<asp:TextBox>	Display data using the property

```
<title>Exampl of single value data binding ASP .Net
with variable</title>
<head>
<body>
  <form id="form1" runat="server">
  <div>
    <asp:Label ID="Label1" runat="server"
Font-Size="Large" ForeColor="Blue">
      EmpID: <%# EmpID %><br />
      empName: <%# EmpName %><br />
      City: <%# City %>
    </asp:Label>
  </div>
  </form>
</body>
</html>
```

Program 3.2.1

Write a program to display single value data binding

Solution :

Program to display single value data binding

```
using System;
using System.Collections;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Xml.Linq;

public partial class DataBindingExample :
System.Web.UI.Page
{
  public int EmpID;
  public string EmpName;
  public string City;
```

Syllabus Topic : Single Value Data Binding

3.2.1 Single Value Data Bound Controls

Q. Explain single value data bound control with example. (4 Marks)

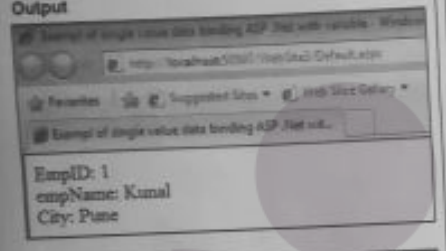
- The single-item data-bound controls as name suggests are used to display the value of a single item of a database table. Direct binding with the data source is not provided by these controls different properties like Text, Caption, or Value are used to show the data field.

Example

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Data_Binding_Example.aspx.cs"
Inherits="DataBindingExample" %>

<!DOCTYPE html>

<html runat="server">
```

```
protected void Page_Load(object sender, EventArgs e)
{
    EmpID = 1;
    EmpName = "Kunal";
    City = "Pune";

    this.DataBind();
}
```

Output



Syllabus Topic : Repeated - Value Data Binding

3.2.2 Multi Value Data Bound Controls

Q. Explain multi value data bound control with example. (4 Marks)

- The multi-item data bound controls are used to display the entire or a partial table. Direct binding to the data source is provided by these controls. The DataSource property of these controls is generally used to bind a database table.

Program 3.2.2

Write a program to display multi item data bound.

Solution :

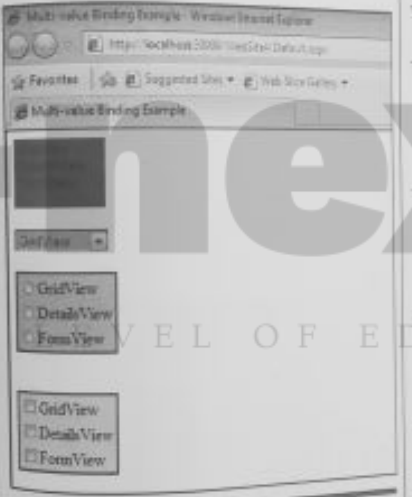
Program to display multi item data bound

```
<%@ Page Language="C#" %>
<%@ Import Namespace="System.Collections.Generic"
%>

<!DOCTYPE html>

<script runat="server">
    protected void Page_Load(object sender,
System.EventArgs e) {
        if (!this.IsPostBack) {
            List<string> DataToolBoxControls = new
List<string>();
            DataToolBoxControls.Add("GridView");
            DataToolBoxControls.Add("DetailsView");
            DataToolBoxControls.Add("FormView");

            List1.DataSource = DataToolBoxControls;
            DropDnList1.DataSource = DataToolBoxControls;
            RadioBList1.DataSource = DataToolBoxControls;
            ChkBoxList1.DataSource = DataToolBoxControls;

            this.DataBind();
        }
    }
</script>

<html>
<head id="Head1" runat="server">
    <title>Multi-value Binding Example</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ListBox
            ID="List1"
            runat="server"
            BackColor="gray"
            ForeColor="Blue"
            >
        </asp:ListBox>
        <br /><br />
        <asp:DropDownList
            ID="DropDnList1"
            runat="server"
            BackColor="SkyBlue"
            ForeColor="Red"
            >
        </asp:DropDownList>
        <br /><br />
        <asp:RadioButtonList
            ID="RadioBList1"
            runat="server"
            BackColor="Pink"
            ForeColor="Black"
            BorderWidth="2"
            BorderColor="Blue"
            >
        </asp:RadioButtonList>
```



```
        <br /><br />
        <asp:CheckBoxList
            ID="ChkBoxList1"
            runat="server"
            BackColor="AntiqueWhite"
            ForeColor="Blue"
            BorderWidth="2"
            BorderColor="Red"
            >
        </asp:CheckBoxList>
    </div>
    </form>
</body>
</html>
```

Output



Syllabus Topic : Data Source Controls - SqlDataSource

3.2.3 Data Source Controls - SqlDataSource

Q. Explain how to issue commands with sqlDatasource control in brief. (4 Marks)

- The SqlDataSource control helps out to make use of Web server control to get data from data source. There are different data bound controls available such as GridView, FormView and DetailsView which are used to show and manipulate data on an ASP.NET Web page.
- To communicate with the multiple databases which can be supported by ADO.NET the SqlDataSource control takes help of various ADO.NET classes. For example Microsoft SQL Server with the help of the System.Data.SqlClient provider, System.Data.OleDb, System.Data.Odbc, and Oracle with the help of System.Data.OracleClient provider.
- SqlDataSource control can be also used to access and change data in an ASP.NET page without taking the help of ADO.NET classes directly. To connect to your database you can provide a connection string and describe the SQL statements or stored procedures that will work on your database. During run-time, this control by-default opens the database connection, executes the SQL statement or stored procedure that you have given, displays the preferred data (if any), and then closes the connection that has been opened.

Connecting the SqlDataSource Control to a Data Source

- While configuring a SqlDataSource control, you have to set some properties. The ProviderName property is set to know the type of database and the ConnectionString property is set to a particular connection string that contains necessary information needed to connect to the database.
- The parameters of a connection string can be different depending on what type of database the data source control is going to use. For example, the SqlDataSource control requires parameters such as a server name, database (catalog) name, and information about how to authenticate the user when connecting to a SQL Server.
- You can store connection string in the configuration settings as a part of your current application with the help of connectionString's configuration element rather than setting connection strings at design time as property settings in the SqlDataSource control.
- This will makes possible to handle connection strings separately of your ASP.NET code, containing encrypting them using Protected Configuration.
- The code snippet given below illustrates a connection to the SQL Server Northwind sample database using a connection string stored in the connectionStrings configuration element named MyNorthwind.

```
<%@ Page Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
  <head runat="server">
```

```
<title>SqlDatasource Example</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:SqlDataSource
      id="SqlDataSource1"
      runat="server"
      DataSourceMode="DataReader"
      ConnectionString="<%$
ConnectionStrings:MyNorthwind%>"
      SelectCommand="SELECT Name FROM
employee">
    </asp:SqlDataSource>

    <asp:ListBox
      id="ListBox1"
      runat="server"
      DataTextField="Name"
      DataSourceID="SqlDataSource1">
    </asp:ListBox>

  </form>
</body>
</html>
```

Issuing Data Commands with the SqlDataSource Control

- The SqlDataSource control can supports four commands i.e. SQL queries. They are listed below
  1. SelectCommand
  2. InsertCommand
  3. UpdateCommand
  4. DeleteCommand
- Every command is a separate property of the data source control and you can define SQL statement to any of these commands for the data source control to execute.
- When you want to select all the columns from the database table you can use the asterisk (*) sign in the Select command, and if you use automatic code generation to perform operation like update or delete then remember that there is no space in columns names.
- You can also generate parameterized commands that contain placeholders for values to be passed at run time. The following example demonstrates a parameterized SQL Select command.

```
Select emp_ID, ename From employee Where salary =
@salary
```

- The DataSource control executes the commands when one of these four methods is called which is equivalent to it.

Returning DataSet or DataReader Objects

- To return data the SqlDataSource control can use one of the two forms :
  1. DataSet object
  2. DataReader object
- You can define which form you want to return by setting the DataSourceMode property of data source control.
- A DataSet object stores all the data in server memory, facilitating you to manipulate the data in a variety of ways after retrieving it.
- A data reader provides you a read-only view that can fetch single record.
- If you want to perform operations on the data such as filtering, sorting, or paging after retrieving it or if you want to maintain a cache then the DataSet is the best choice.
- In contrast, if you want to return the data and are using a control on the page to display that data then DataReader is best one. For example for returning data that you want to show in a ListBox, DropDownList, or GridView control where a list of results is displayed in a read-only format, Datareader is best to use.

Caching with the SqlDataSource Control

- When you retrieve data from the server you can save that data for future use.
- This data is nothing but the cache data. The SqlDataSource can allow this so as to improve the performance of your applications by avoiding costly queries.
- Caching is practical in nearly any situation where the data is not highly volatile and the cached results are small to avoid using the huge system memory.
- By default caching is not enabled, you have to explicitly enable it.
- To enable it set the property EnableCaching to true. You can also provide the cache time by setting the CacheDuration property to the number of seconds you want to cache data.
- The data source control keeps a separate cache entry for each combination of connection, select command, select parameters, and cache settings.

Filtering with the SqlDataSource Control

- To filter the data without re-running the query you have to enable caching for the SqlDataSource control and have to specify a dataset as a form for data returned by a Select query.
- The SqlDataSource control makes the use of FilterExpression property which allows you to give selection criteria that are applied to the data preserved by the data source control.

Sorting with the SqlDataSource Control

- The SqlDataSource control can supports the sort requests coming from the bound control. This can be done when the DataSourceMode is set to DataSet.

Syllabus Topic : Data Controls

3.3 Data Controls

Q. Explain different data controls of ADO.NET.

There are different data controls available in ADO.NET. They are explained briefly in subsequent sections.

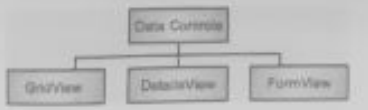


Fig. 3.3.1 : Data control of ADO.NET

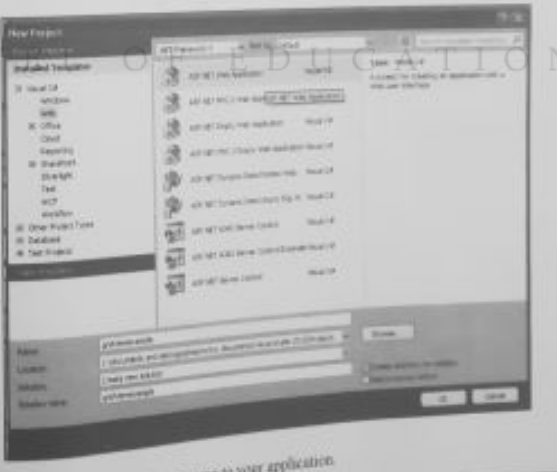
Syllabus Topic : GridView

3.3.1 GridView

Q. Explain GridView data controls of ADO.NET. Write down steps of creating GridView. **(2 Marks)**

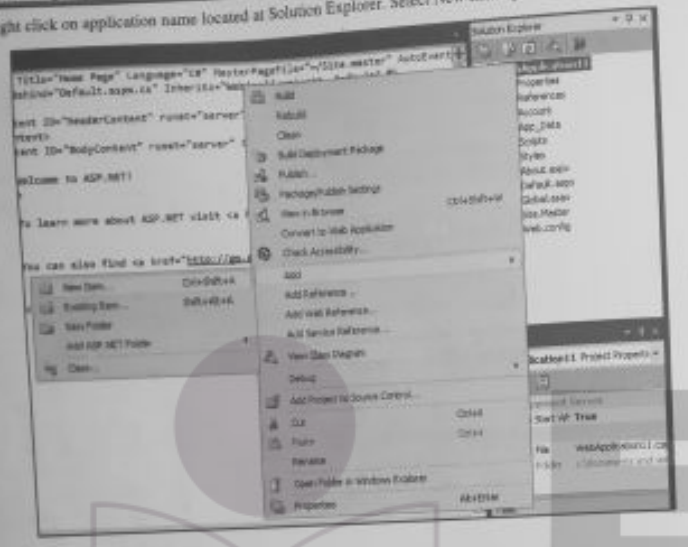
- The GridView is a very flexible grid control which can be used to display a data in tabular format. In grid view every record present inside your data source becomes an individual row and every field becomes an individual column.
- The GridView is considered as very powerful rich data control because it offers various functionalities. This functionalities includes support for automatic paging, sorting of a data, selecting a particular record, and editing a record.
- Following are the steps to create the Gridview control.

Step-1 : Click on File option and click on new project option. It will give the following screen.

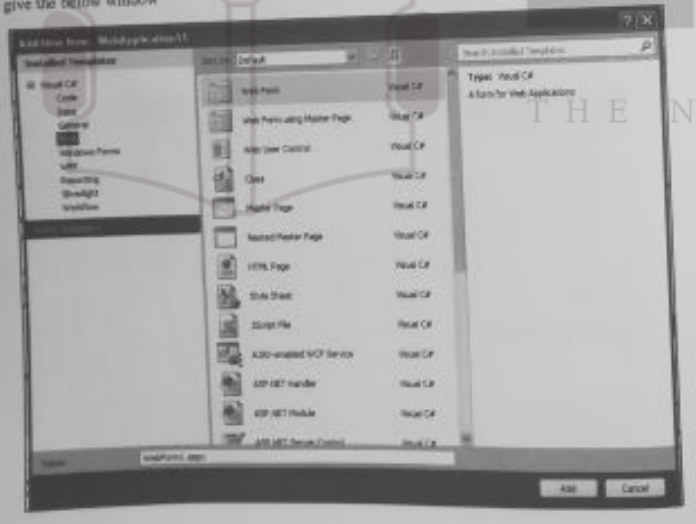


Select ASP.NET web application and give name to your application.

Step 2 : Right click on application name located at Solution Explorer. Select New Item option from Add drop-down list.

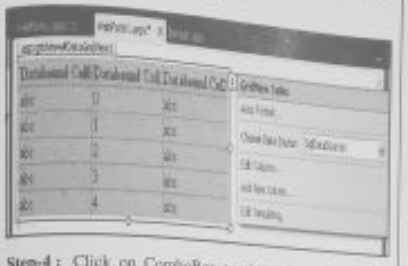


It will give the below window



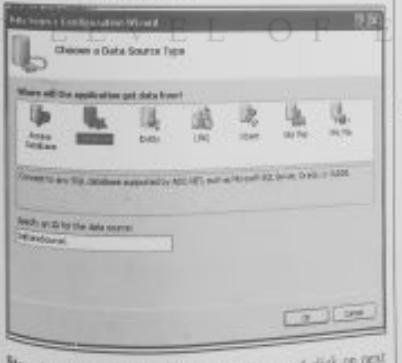
Select Web Form option and click on Add button to add web form to your application.

Step-3 : Take GridView control from the toolbox. Click on right corner arrow symbol. It will open a pop-up window as given below :



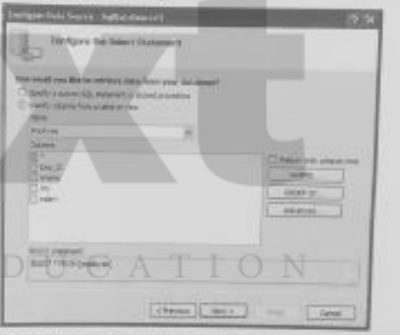
Step-4 : Click on ComboBox to choose the data source. Select new data source option from the list.

Step-5 : Now a new window will appear which will ask you for data source type. Select SQL database and click on OK button.

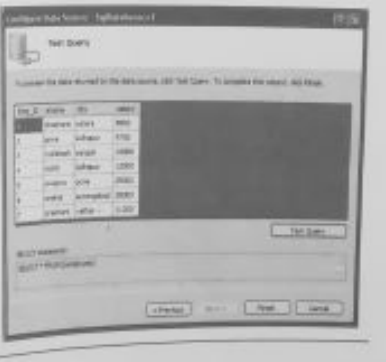


Step-6 : Now select the data connection and click on next button.

Step-7 : Now you can configure the sql statement or stored procedures if any.



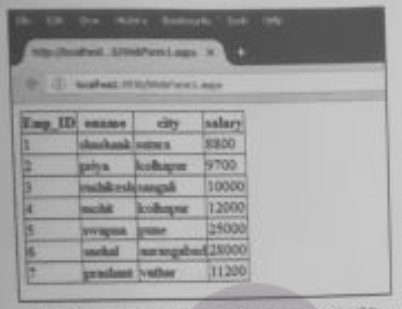
Step-8 : Click on next button and test the query if you want. It looks likes as follow.



Step-9 : Click the Finish button.

Now it's time to run the application.

Output



Emp_ID	ename	city	salary
1	shashank	satara	8800
2	priya	kolhapur	9700
3	sushilesh	sangali	10000
4	mohit	kolhapur	12000
5	swapna	pune	25000
6	snehal	aurangabad	28000
7	prashant	vather	11200

- Another way to attach the database to the GridView is by writing a code. Write the following code on the form load event.

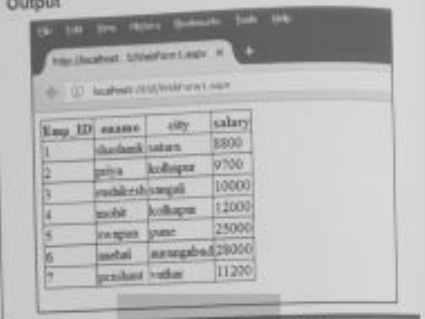
Program 3.3.1

Write a program to attach the database to the GridView.

Solution : Program to attach the database to the GridView

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Data.SqlClient;

namespace DatabindingWithdataGridView
{
    public partial class Form1 : System.Web.UI.Page
    {
        SqlDataAdapter dadapter;
        DataSet dset;
        string connstring =
"server=;database=Database1;user=;password=";
        private void Form1_Load(object sender, EventArgs e)
        {
            dadapter = new SqlDataAdapter("select * from
employee", connstring);
            dset = new System.Data.DataSet();
            dadapter.Fill(dset);
            dataGridView1.DataSource =
dset.Tables[0].DefaultView;
        }
    }
}
```

Output



Emp_ID	ename	city	salary
1	shashank	satara	8800
2	priya	kolhapur	9700
3	sushilesh	sangali	10000
4	mohit	kolhapur	12000
5	swapna	pune	25000
6	snehal	aurangabad	28000
7	prashant	vather	11200

Syllabus Topic : DetailsView

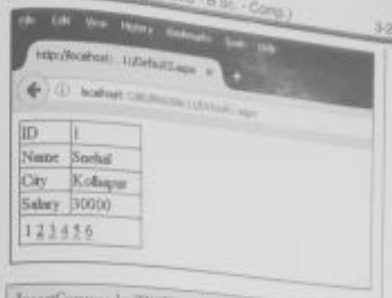
3.3.2 DetailsView

Q. Explain DetailsView data controls of ADO.NET. (4 Marks)

- As you know the GridView control is used to displays all of the records from its data source control at a time, whereas the DetailsView control is used to display single record from a data source at a time in the tabular form.
- The DetailsView control can be used to perform operations on the table data such as updating, inserting, and deleting records from the table.
- Sorting of data cannot be performed with the help of DetailsView control. The GridView control provides an interface for the users to navigate through the records.

☞ **To use the detailsView control perform the following steps :**

1. Take Detailsview control from the toolbox and perform the data binding to it as discussed in Gridview control.
2. When you run the application following output will be produced.
3. If you want to insert a new record to your table then set the AutoGenerateInsertButton to "True" and add the following code in Form tag.

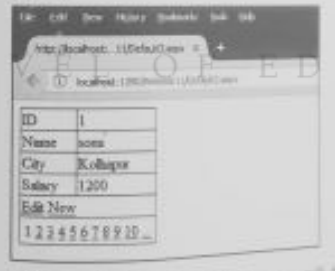


ID	1
Name	Snehal
City	Kolhapur
Salary	30000

1 2 3 4 5 6

```
InsertCommand="INSERT INTO employee(ID, Name, City,
Salary) VALUES (@ID, @name, @city, @salary)"
<InsertParameters>
<asp:Parameter Name="Id" Type="INT32" Size="20" />
                <asp:Parameter Name="name"
Type="string" Size="200" />
                <asp:Parameter Name="City"
Type="string" Size="50" />
                <asp:Parameter Name="Salary"
Type="INT32" Size="20" />
</InsertParameters>
```

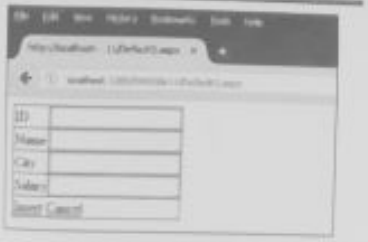
- When you run the application it will give the following window



ID	1
Name	xcxx
City	Kolhapur
Salary	1200
Edit New	

1 2 3 4 5 6 7 8 9 10 ...

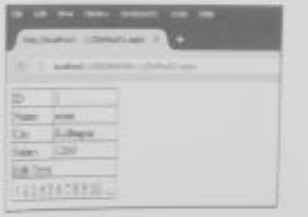
- When you click on new option new page will be loaded. It looks like
- Enter the data and click on Insert option. It will insert the record to your table



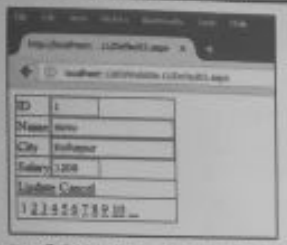
4. Similarly if you want to update a new record to your table then set the AutoGenerateEditButton to "True" and add the following code in Form tag

```
UpdateCommand="UPDATE employee SET ID = @ID,
Name = @name, City = @city, Salary = @salary
WHERE ID = @ID"
    <UpdateParameters>
    <asp:Parameter Name="ID" Type="Int32" />
    <asp:Parameter Name="name" Type="string"
Size="50" />
    <asp:Parameter Name="city" Type="string"
Size="20" />
    <asp:Parameter Name="salary" Type="string"
Size="50" />
    </UpdateParameters>
```

- When you run the application it will give the following window



- When you click on new option new page will be loaded. It looks like
- Edit the data and click on Update option. It will update the record.



- Following Snippet gives the complete code of the DetailsView Control.

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="Default2.aspx.cs" Inherits="Default2" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
  <div>

    <asp:DetailsView ID="DetailsView1" runat="server"
AllowPaging="True"
      AutoGenerateRows="False"
DataSourceID="SqlDataSource1" Height="50px"
      Width="125px" AutoGenerateInsertButton="True"
      AutoGenerateDeleteButton="True"
AutoGenerateEditButton="True">
      <Fields>
        <asp:BoundField DataField="ID"
HeaderText="ID" SortExpression="ID" />
        <asp:BoundField DataField="Name"
HeaderText="Name" SortExpression="Name" />
        <asp:BoundField DataField="City"
HeaderText="City" SortExpression="City" />
        <asp:BoundField DataField="Salary"
HeaderText="Salary"
          SortExpression="Salary" />
      </Fields>
    </asp:DetailsView>
    <asp:SqlDataSource ID="SqlDataSource1"
runat="server"
      ConnectionString="<%$
ConnectionStrings:ConnectionString %>"
      SelectCommand="SELECT * FROM [employee]"

      InsertCommand="INSERT INTO employee(ID,
Name, City, Salary) VALUES (@ID, @name, @city,
@salary)"
      UpdateCommand="UPDATE employee SET ID =
@ID, Name = @name, City = @city, Salary = @salary
WHERE ID = @ID">
      <UpdateParameters>
        <asp:Parameter Name="ID" Type="Int32" />
        <asp:Parameter Name="name" Type="string"
Size="50" />
        <asp:Parameter Name="city" Type="string"
Size="200" />
        <asp:Parameter Name="salary" Type="string"
Size="50" />
      </UpdateParameters>
      <InsertParameters>
        <asp:Parameter Name="Id"
Type="INT32" Size="50" />
        <asp:Parameter Name="name"
Type="string" Size="200" />
        <asp:Parameter Name="City"
Type="string" Size="50" />
        <asp:Parameter Name="salary"
Type="INT32" Size="20" />

      </InsertParameters>
    </asp:SqlDataSource>

  </div>
  </form>
</body>
</html>
```

Syllabus Topic : FormView

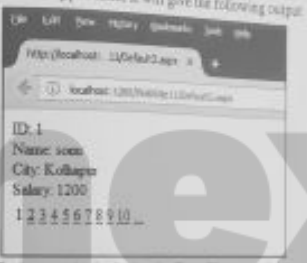
3.3.3 FormView

Q. Explain FormView data controls of ADO.NET. (4 Marks)

- Similar to DetailsView control, the FormView control also provides the facility of displaying single record at a time. The difference between the FormView and DetailsView control is that the DetailsView displays the record in tabular format where as FormView control does not contain any predefined layout for display the record.
- In FormView control you can specify the layout for displaying the records on the browsers.

To add a FormView control to a page

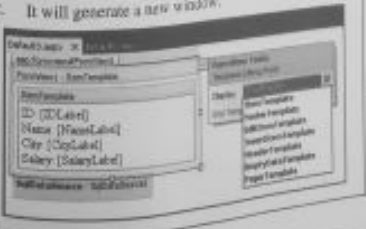
- Select and drag the FormView control from the Toolbox and place where you want.
- To bind the data source perform the data binding to it as discussed in Gridview control.
- Run the application, It will give the following output.



To interactively design the FormView templates

1. In Design view, click on the FormView control. Then on the Common FormView Tasks menu, click Edit Templates.

2. It will generate a new window.



3. From the Display dropdown, select the template you would like to edit. And run the application. Now you have selected the EditItemTemplate.

Syllabus Topic : Working with XML

3.4 Working with XML

- XML stands for eXtensible Markup Language.
- XML is basically designed to store and transport data.
- XML supports both human- and machine-readable data format.
- XML was designed to be self-descriptive.
- XML is a W3C Recommendation.
- XML is a simple text-based format which represents information in structured format like documents, transactions, data, invoices, books etc. This language is derived from comparatively older standard format called SGML (Standard Generalized Markup Language), to make it more suitable for Web use.

Need of XML

- There are several reasons for the need of XML as follows :

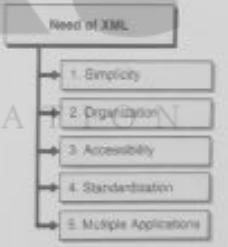


Fig. C3.2 : Need of XML

1. Simplicity

- XML can be easily understood. We can create our own tags and build the application.
- We are free to develop the systems as per our requirements and with our own conventions. This makes the thing very simple for us.

2. Organization

- The design process can be segmented to build the platform. Data can be stored on one page while the formatting rules can be stored on another page.

- It is possible to create the data page to store the content first and later on we can work on design. XML allows us to create the website in stages and stay organized in the entire process.

3. Accessibility

- Data can be divided in XML.
- This makes the access of data easy and fast whenever there is need of making change in the data.

4. Standardization

- XML is an international standard.
- This means XML document can be viewed anywhere in the world.

5. Multiple Applications

- "Write once, use anywhere, any number of times" rule is applied to XML.
- For XML data we can create any number of display pages as we want. XML allows us to create various styles and formats for a single page as per requirement.

XML Key Components

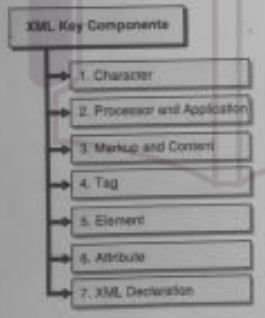


Fig. C3.3 : XML key components

1. Character

- An XML document is a string of characters.
- Almost every legal Unicode character may appear in an XML document.

2. Processor and Application

- The processor analyzes the markup and passes structured information to an application.
- The specification places requirements on what an XML processor must do and not do, but the application is outside its scope.
- The processor (as the specification calls it) is often referred to formally as an XML parser.

3. Markup and Content

- The characters making up an XML document are divided into markup and content, which may be distinguished by the application of simple syntactic rules.
- Generally, strings that constitute markup either begin with the character < and end with a >, or they begin with the character & and end with a ;. Strings of characters that are not markup are content.

4. Tag

- A tag is a markup construct that begins with < and ends with >.
- Tags come in three flavors
  - (i) start-tag, such as <section>
  - (ii) end-tag, such as </section>
  - (iii) empty-element tag, such as <line-break />

5. Element

- An element is a logical document component that either begins with a start-tag and ends with a matching end-tag or consists only of an empty-element tag.
- The characters between the start-tag and end-tag, if any, are the element's content, and may contain markup including other elements which are called child elements.

Example

```
<greeting>Hello, world!</greeting>.
Another is
<line-break />.
```

6. Attribute

- An attribute is a markup construct consisting of a name-value pair that exists within a start-tag or empty-element tag. An example is , where the names of the attributes are "src" and "alt", and their values are " Rose.jpg" and " Rose " respectively.

- Another example is <step number="3">Connect A to B.</step>, where the name of the attribute is "number" and its value is "3".
- An XML attribute can only have a single value and each attribute can appear at most once in each element. In the common situation where a list of multiple values is desired, this must be done by encoding the list into a well-formed XML attribute with some format beyond what XML defines itself. Usually this is either a comma or semi-colon delimited list or, if the individual values are known not to contain spaces, a space-delimited list can be used.
- Welcome! , where the attribute "class" has both the value "inner greeting-box" and also indicates the two CSS class names "inner" and "greeting-box".

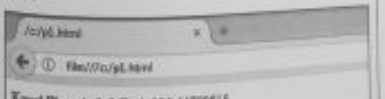
7. XML Declaration

- XML documents may begin with an XML declaration that describes some information about themselves.
- An example is <?xml version="1.0" encoding="UTF-8"?>.

XML Example 1

```
<?xml version="1.0" encoding="UTF-8"?>
<contact-info>
  <name>Kunal</name>
  <company>Phoenix InfoTech</company>
  <phone>020 64700515</phone>
</contact-info>
```

Output



- There are two classes for XML to read and write the data into the document. First one is XMLTextReader and XMLTextWriter respectively.

Syllabus Topic : XML Class - XMLTextWriter

3.4.1 The XMLTextWriter

Q. Explain XMLTextWriter class in detail.

- The XMLTextWriter class is inherited from the XMLWriter class and can be used to create any XML document.

Adding namespace Reference

- The Xml classes are defined in the System.XML namespace, so that our first step is to add this namespace to the project.

```
Imports System.IO
Imports System.Xml
```

Creating an XML Document

- You can create a XML file if it is already not present with the help of constructor of the XmlTextWriter class while writing into the file.
- Consider following example where the new XML file having name xmlf.xml is placed at D:\programs directory.

```
XmlTextWriter writer = new XmlTextWriter
("D:\programs\xmlf.xml", null);
```

- You can also reuse XML document to write data into XML file and can display XML contents on the Console. To do this just pass Console.Out as a parameter of the constructor.

```
XmlTextWriter writer = new XmlTextWriter(Console.Out);
```

Inserting Data to the Document

- To insert data into the document some methods are used. They are listed below.
  1. **WriteStartDocument** : It is used to start the new document.
  2. **WriteString** : It is used to write the string to the document.
  3. **WriteStartElement and WriteEndElement** : It is used in pair to add new elements in to the document.
  4. **WriteComment** : It is used to give comments in the document.
- Consider the following example which can use the XmlTextWriter class and associated methods.

Note : Write code in console application

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Xml;
```

```
namespace ConsoleApplication1
{
  class Program
  {
      public static void Main()
  {
   XmlTextWriter writer = new
XmlTextWriter("Bookdata.xml", null);

   writer.WriteStartElement("Books");

   writer.WriteElementString("id", "01");
   writer.WriteElementString("title", "Introduction to C");
   writer.WriteElementString("price", "450");

   writer.WriteElementString("id", "02");
   writer.WriteElementString("title", "System
Programming");
   writer.WriteElementString("price", "350");

   writer.WriteElementString("id", "03");
   writer.WriteElementString("title", "DOT.NET");
   writer.WriteElementString("price", "500");

   writer.WriteEndElement();

     writer.Close();
     Console.ReadLine();

    }
  }
}
```

Start with the root element

To write the sub element

Output

- The file named as bookdata.xml will be created at yourapplicationname\bin\Debug directory.

Syllabus Topic : XML Class - XMLTextReader

3.4.2 The XMLTextReader

Q. Explain XMLTextReader class in brief.

- To read data from the XML file the XmlTextReader class can be used. Like an XMLTextWriter class the XmlTextReader class is also comes in sequential manner and forward-only option that means once you have parsed the particular node and moved on then you cannot come back.
- The dynamic searching of node in xml file with the help of XMLTextReader is not possible. You have to traverse each and every node of xml file until the desired node is not encountered or till the end of file.
- Therefore, XmlTextReader class is most useful in circumstances where you're dealing with small files or the application requires the reading of the whole file contents.

Reading and Parsing XML Nodes

- To read the content of xml file you have to create an object of an XmlTextReader class and then inside the loop repeatedly call the XmlTextRead.Read () method until that method returns false.
- Consider the following example which will read the content of xml file which was created through XMLTextWriter class.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Xml;
namespace ConsoleApplication1
{
  class Program
  {
      public static void Main()
  {
    try
    {
      String out1="";
      String format = "XmlNodeType::{0,-12}{1,-
10}{2}";
```



```
XmlTextReader xmlreader = new
XmlTextReader("bookdata.xml");

      while (xmlreader.Read())
      {
        out1
=String.Format(format,(xmlreader.NodeType),xmlreader.Na
me,xmlreader.Value);

        Console.WriteLine(out1);
      }
    }
    catch (Exception ex)
    {
      Console.WriteLine(ex.Message);
        }
      Console.ReadLine();
        }
  }
}
```

Read the data from XML file

Output

Syllabus Topic : Caching - When to use Caching

3.5 Caching

3.5.1 When to use Caching

Q. What is mean by caching ? Explain when to use caching. (4 Marks)

- While developing an application some data is needed repeatedly. If the data is stored at different locations such as on server or on any remote machine then accessing the data every time requires lot of wastage of time as well as system resource to calculate the result that you require.
- Caching is a method mainly used for storing frequently used data/information in memory, so that, when the same data/information is needed next time, it could be directly retrieved from the memory rather than of being generated by the application.
- Caching method is particularly important for improving the performance in ASP.NET, as the pages and controls are generated dynamically. When data related transactions are considered then the caching is best choice because these are costly in terms of response time.
- Data that is used most frequently is placed in random access memory by caching as it can be accessed quickly.
- The ASP.NET runtime consist of a key-value mapping of common language runtime objects called cache. This exists with the application and is accessible via the HttpContext and System.Web.UI.Page.
- The data in cache will not be available in the following scenarios
  - (i) Lifetime of data stored in cache expires.
  - (ii) If the application free its memory.
  - (iii) Due to some reason caching does not take place.
- With the help of indexer you can access items from cache. You may also manage the lifetime of objects in the cache and creates associations between the cached objects and their physical sources.

3.5.2 Types of Caching

Q. Explain the different types of caching. (2 Marks)

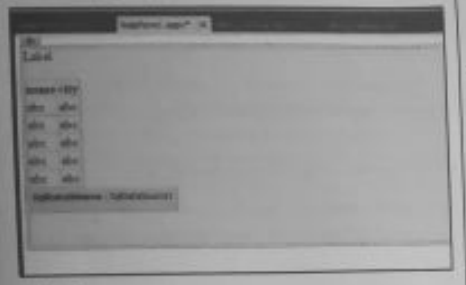
Following are different types of caching

Fig. 3.5.1 : Types of Caching

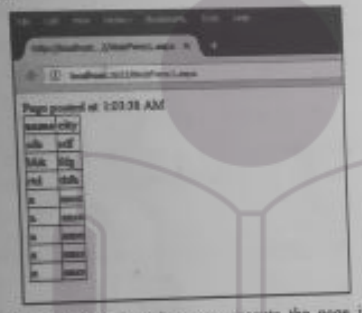
1. Output Caching

- Output cache is used to stores a replica of the final delivered HTML pages or piece of pages sent to the client.

The design of page should look as shown :



Output



- When for the first time you execute the page it will show the time of posted and each time you refresh the page, the page is reloaded and the time shown on the label changes.
- Now we can set the EnableCaching attribute of the data source control to be 'true' and set the Cache duration attribute to '70'. It will implement caching and the cache will expire every 70 seconds.

```
<asp:SqlDataSource ID="SqlDataSource1" runat="server"
        ConnectionString="<%$
ConnectionStrings:ConnectionString %>"
        SelectCommand="SELECT * FROM [data]"
        EnableCaching="true"
CacheDuration="70"></asp:SqlDataSource>
```

Syllabus Topic : LINQ - Understanding LINQ

3.6 LINQ

Q. What is LINQ?

- LINQ Stands for Language Integrated Query. The LINQ is mainly used to provide consistent access to several data sources like databases and XML.

3.6.1 Introduction to LINQ

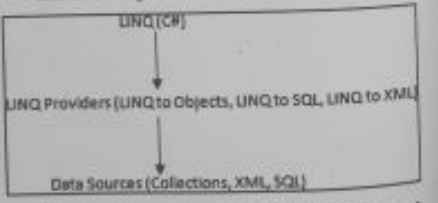
- Now days most of the applications are data-centric, however most of the data repositories are relational databases. Over the years developers have designed applications based on object models.
- The main task of objects is to connect with the data access components, which is also called the Data Access Layer. Consider the following points :

(i) It is not necessary that all the data needed in an application should be stored in the same source. The source could be different such as relational database, some business object, XML file, or a web service.

(ii) Accessing data from particular database or XML file is difficult and costly than accessing in-memory object.

(iii) The data needs to be sorted, ordered, grouped, altered etc. as the accessed data is not used directly.

- LINQ or Language-Integrated Query is a tool that can access all kinds of data sources which permit the combining data from different data sources and perform standard data processing operations.
- ADO.NET is considered as best choice to manipulate the data of a database if you don't want to use the LINQ. But ADO.NET uses tables, relations and other relational database constructs which needs to be mapped to the application objects. This contains more mistakes. Because of this LINQ gets higher priority by users for coding.



- The above diagram shows that the LINQ providers act as a bridge between LINQ and the data sources like SQL and XML, collections etc.
- The three main data providers provided by Microsoft are :
  1. LINQ to objects for querying in-memory collections
  2. LINQ to SQL for querying relational databases
  3. LINQ to XML for querying XML data

- LINQ is included in C# and VB.NET so that we can query any data source in any programming language that you desire. If you don't want to use LINQ and if we wanted to query a database we would use SQL but with LINQ we can do it in the programming language.

Syllabus Topic : LINQ Basics

3.6.2 LINQ Basics

Q. Explain various classes used in LINQ

- LINQ is collection of extensions to the .Net Framework 3.5 and its managed languages that set the query as an object. It describes a common syntax and a programming model to query several types of data with the help of common language.
- For example, querying the employee table in the database, using LINQ query in C#, the code would be

```
var data = from e in dataContext.employee
where e.salary >= 25000
select e;
```

- Where
  - The 'from' keyword can be used to logically loops through the contents of the collection.
  - The expression with the 'where' keyword is estimated for each and every object present inside the collection.
  - The 'select' statement is used to select the estimated object to add to the list being returned.
  - The 'var' keyword is used for variable declaration. You don't know the exact type of the returned object; it indicates that the information will be inferred dynamically.
- LINQ query can be applied to any data-bearing class that is inherited from IEnumerable<T>, here T is any data type, for example, List<Book_data>.

Example

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public class Book_data
{
    public string ID { get; set; }
    public string Title { get; set; }
    public decimal Price { get; set; }

    public static List<Book_data> GetBooks()
    {
        List<Book_data> list = new List<Book_data>();

        list.Add(new Book_data
        {
            ID = "01",
            Title = "System Programming",
            Price = 620 });

        list.Add(new Book_data
        {
            ID = "02",
            Title = "Web Technology",
            Price = 250 });

        list.Add(new Book_data
        {
            ID = "03",
            Title = "Automata Theory",
            Price = 320 });

        list.Add(new Book_data
        {
            ID = "04",
            Title = "Visual Basic",
            Price = 180 });

        list.Add(new Book_data
        {
            ID = "05",
            Title = "Programming in C",
            Price = 210 });

        return list;
    }
}
```

Inserting data to List

- Now take the label control from the toolbox to displays the titles of the books from bookdata. The Page_Load event creates a list of books and returns the titles by using LINQ query.

- When the next client requests for this page, rather than regenerating the page, a cached replica of the page is sent, which will save the time to give response to clients.

2. Data Caching

- Data caching is used to cache data from various data sources. Until the cache is not expired, a request for the data will be accomplished with the help of cache.
- When the cache is expired, the new data is obtained by the data source and the cache is again filled with data.

3. Object Caching

- Object caching is a technique used for caching the objects present on a page.
- For example data-bound controls - gridview, detailsview, etc. The cached data is kept in server memory.

4. Class Caching

- When you run for the first time, the web pages or web services are compiled into a page class in the assembly. Then the assembly is cached in the server.
- Next time when a request is came for the page or service, the cached assembly is referred to. When the source code is modified, the common language runtime recompiles the assembly.

5. Configuration Caching

- The application's configuration information is stored in a configuration file.
- Similarly configuration caching holds the configuration information in the server memory.

Syllabus Topic : Output Caching

3.5.3 Output Caching

Q. Write a short note on output caching. (2 Marks)

- There are some complex processes in page rendering such as database access, rendering complex controls etc. Output cache helps to skip the round trips to server with the help of caching data in memory. It is possible cache even the entire page.
- The output caching is done with the help of OutputCache directive. It enables output caching and provides certain control over its behavior.

Syntax

```
<%@ OutputCache Duration="20" VaryByParam="None" %>
```

- Insert this directive under the page directive. This informs the environment to cache the page for 20 seconds. The following event handler for page load would help in testing that the page was truly cached.

```
protected void Page_Load(object sender, EventArgs e)
{
  Thread.Sleep(20000);
  Response.Write("This page was generated and cache at:"
+ DateTime.Now.ToString());
}
```

- The Thread.Sleep() method is used to stop the process thread for the specific time. In the above example, the thread is stopped for 20 seconds, so when the page is loaded for first time, it takes 20 seconds.
- When next time you refresh the page it does not take any time, because the page is retrieved from the cache.
- The Table 3.5.1 contains the attributes for OutputCache directive, which helps in controlling the behavior of the output cache

Table 3.5.1

Sr. No.	Attribute	Values	Description
1.	DiskCacheable	true/false	This attribute is used to state that output could be written to a disk based cache.
2.	NoStore	true/false	It will be used to state that the "no store" cache control header is sent or not.
3.	CacheProfile	String name	It indicates the name of a cache profile for storing in web.config.
4.	VaryByCustom	Browser Custom string	This attribute is used to inform to vary the output cache by the name of browser and version or by a custom string.

Sr. No.	Attribute	Values	Description
5.	Location	Any	Any : page may be cached anywhere.
		Client	Client : contents of cache will reside at client's machine.
		Downstream	Downstream : contents of cache will reside at downstream and server both.
		Server	Server : contents of cache will reside at server's machine.
		None	None : It will disable the caching.
6.	Duration	Number	It used to state for how much time the page or control is cached.

- Now you can add a text box and a button to the previous example and add this event handler for the button.

```
protected void btnImagic_Click(object sender, EventArgs e)
{
  Response.Write("<br><br>");
  Response.Write("<h3> Welcome, " + txtUname.Text
+ "</h3>");
}
```

- Modify the OutputCache directive :

```
<%@ OutputCache Duration="50"
VaryByParam="txtname" %>
```

- When the program is executed, ASP.NET caches the page on the basis of the name in the text box.

Syllabus Topic : Data Caching

3.5.4 Data Caching

Q. Write a short note on data caching. (2 Marks)

- The key characteristic of data caching is to cache the data through data source controls. As we already know that the data source controls is used to signify the data in a data source.
- The data source controls are inherited from the abstract class named as DataSourceControl. It will have the following properties for implemeneting caching :

Properties for implementing coching

1. **CacheDuration** : This property is used to set the number of seconds for which the data source will cache data.
2. **CacheExpirationPolicy** : This property is used to define the behavior of cache when the data in cache has expired.
3. **CacheKeyDependency** : This property is used to identify a key for the controls that auto-expires the content of its cache when deleted.
4. **EnableCaching** : This property is used to specify whether the data will be cache or not.

Example

To illustrate the data caching perform the following steps :

1. Create a new website in visual studio.
2. Right click on website name present at solution explorer and choose new item from add option. A new window will appear, click on web form option.
3. Add a SqlDataSource control with the database connection.
4. Add a label to the page, which is used to show the response time for the page.

```
<asp:Label ID="labelTime"
runat="server"></asp:Label>
```

5. Add an event handler for the page load event :

```
protected void Page_Load(object sender, EventArgs e)
{
  labelTime.Text = String.Format("Page posted at: {0}",
DateTime.Now.ToLongTimeString());
}
```

```
public partial class _Default : System.Web.UI.Page
{
  protected void Page_Load(object sender, EventArgs e)
  {
    List<Book_data> books = Book_data.GetBooks();
    var booktitles = from t1 in books select t1.Title;

    foreach (var title in booktitles)
      data.Text += String.Format("{0} <br />", title);
  }
}
```

Retrieving data from the list

- When you execute the above code following window will give the results of the query

LABELSYSTEM PROGRAMMING
WEB TECHNOLOGY
AUTOMATA THEORY
VISUAL BASIC
PROGRAMMING IN C

- The above LINQ expression

```
var booktitles = from t1 in books select t1.Title;
```

Is equivalent to the following SQL query

```
SELECT Title from Book_data;
```

LINQ Clauses

In this section you can see the several clauses of LINQ. They are described in below

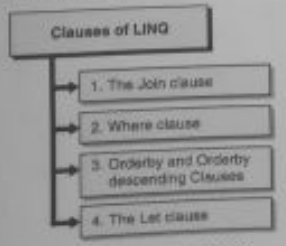


Fig. C3.4 : Clauses of LINQ

1. The Join clause

- In SQL the 'join clause' is used for joining two different data tables and displays a data set containing columns from both the tables.
- LINQ is also able to do this operation. To perform this, add another class named otherbookdata.cs in the previous project.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public class other_data
{
  public int total_copies { get; set; }
  public int total_pages { get; set; }
  public string ID { get; set; }

  public static IEnumerable<other_data> getother_data()
  {
    other_data[] data2 =
    {
      new other_data { ID = "01", total_pages=134,
total_copies = 13000},
      new other_data { ID = "02", total_pages=456,
total_copies = 45000},
      new other_data { ID = "03", total_pages=348,
total_copies = 56000},
      new other_data { ID = "04", total_pages=786,
total_copies = 10000},
      new other_data { ID = "05", total_pages=598,
total_copies = 78000},
      new other_data { ID = "06", total_pages=380,
total_copies = 50000},
      new other_data { ID = "07", total_pages=650,
total_copies = 37000},
    };

    return data2.OfType<other_data>();
  }
}
public class Book_data
{
  public string ID { get; set; }
  public string Title { get; set; }
  public decimal Price { get; set; }
```



```
public static List<Book_data> GetBooks()
{
  List<Book_data> list = new List<Book_data>();
  list.Add(new Book_data
  {
    ID = "01",
    Title = "System Programming",
    Price = 620
  });
  list.Add(new Book_data
  {
    ID = "02",
    Title = "Web Technology",
    Price = 250
  });

  list.Add(new Book_data
  {
    ID = "03",
    Title = "Automata Theory",
    Price = 320
  });

  list.Add(new Book_data
  {
    ID = "04",
    Title = "Visual Basic",
    Price = 180
  });

  list.Add(new Book_data
  {
    ID = "05",
    Title = "Programming in C",
    Price = 210
  });

  return list;
}
}

public partial class _Default : System.Web.UI.Page
{
  protected void Page_Load(object sender, EventArgs e)
  {
    IEnumerable<Book_data> books =
Book_data.GetBooks();
    IEnumerable<other_data> total_copies =
other_data.getother_data();

    var booktitles = from b in books
                     join c in total_copies on b.ID equals c.ID
                     select new { Name = b.Title, total_pages =
c.total_pages };

    foreach (var title in booktitles)
      data1.Text += String.Format("{0} <br />", title);
  }
}
```

Output

```
LABEL{ NAME = SYSTEM PROGRAMMING, TOTAL PAGES = 134 }
{ NAME = WEB TECHNOLOGY, TOTAL PAGES = 456 }
{ NAME = AUTOMATA THEORY, TOTAL PAGES = 348 }
{ NAME = VISUAL BASIC, TOTAL PAGES = 786 }
{ NAME = PROGRAMMING IN C, TOTAL PAGES = 598 }
```

2) Where clause

- The 'where clause' is used to add some conditional filters to the query. Consider the following example, which shows the names and the number of pages which are less than 400.
- For doing this make following changes to Page_Load event handler

```
protected void Page_Load(object sender, EventArgs e)
{
  IEnumerable<Book_data> books =
Book_data.GetBooks();
  IEnumerable<other_data> total_copies =
other_data.getother_data();

  var book1 = from x in books
              join y in total_copies on x.ID equals y.ID
              where y.total_pages <= 400
              select new { Name = x.Title, Pages =
y.total_pages };
  foreach (var t2 in book1)
    data1.Text += String.Format("{0} <br />", t2);
```

- The query returns only those rows, where the number of pages is less than 500.

Output

```
LABEL{ NAME = SYSTEM PROGRAMMING, PAGES = 134 }
{ NAME = AUTOMATA THEORY, PAGES = 348 }
{ NAME = DATABASE SYSTEM, PAGES = 380 }
{ NAME = DATA STRUCTURE, PAGES = 380 }
```

3. Orderby and Orderby descending Clauses

- The Orderby clause is used for sorting the query results.
- To display the name of the book and the number of pages, sorted by the price, write the following code in the Page_Load event handler.

```
protected void Page_Load(object sender, EventArgs e)
  {
    IEnumerable<Book_data> books =
Book_data.GetBooks();
    IEnumerable<other_data> total_copies =
other_data.getother_data();

    var book1 = from b in books
                join s in total_copies on b.ID equals s.ID
                where s.total_pages <= 400
                select new { Name = b.Title, Pages =
s.total_pages };

    var book1titles = from b in books
                join s in total_copies on b.ID equals s.ID
                orderby b.Price
                select new { Name = b.Title, Pages =
s.total_pages, Price = b.Price };

    foreach (var t2 in book1)
      data1.Text += String.Format("{0} <br />", t2);
  }
```

Output

```
LABEL{ NAME = SYSTEM PROGRAMMING, PAGES = 134 }
{ NAME = AUTOMATA THEORY, PAGES = 348 }
{ NAME = DATABASE SYSTEM, PAGES = 380 }
{ NAME = DATA STRUCTURE, PAGES = 380 }
```

4. The Let clause

- The let clause is used to define a variable and assigning it a value calculated from the data values. For example, to calculate the total number of copies, you need to calculate :

```
Finalnumberofcopies = Price of the Book data* otherdata
```

- To achieve this, add the following code snippets in the Page_Load event handler

```
protected void Page_Load(object sender, EventArgs e)
  {
    IEnumerable<Book_data> books =
Book_data.GetBooks();
    IEnumerable<other_data> total_copies =
other_data.getother_data();

        var book1 = from b in books
                    join s in total_copies on b.ID equals s.ID
                    let total = (b.Price * s.total_copies)
                    select new { Name = b.Title, TotalSale =
total };
        foreach (var t3 in book1)
          data1.Text += String.Format("{0} <br />", t3);
  }
```

Output

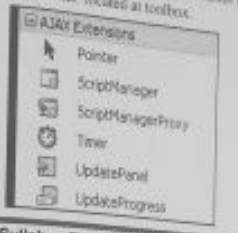
```
LABEL{ NAME = SYSTEM PROGRAMMING, TOTALSALE = 8060000 }
{ NAME = WEB TECHNOLOGY, TOTALSALE = 11250000 }
{ NAME = AUTOMATA THEORY, TOTALSALE = 17920000 }
{ NAME = VISUAL BASIC, TOTALSALE = 1800000 }
{ NAME = PROGRAMMING IN C, TOTALSALE = 16380000 }
{ NAME = DATABASE SYSTEM, TOTALSALE = 25500000 }
{ NAME = DATA STRUCTURE, TOTALSALE = 17500000 }
```

Syllabus Topic : ASP.NET AJAX

3.7 ASP.NET AJAX

- AJAX can be abbreviated as Asynchronous JavaScript and XML. AJAX is a cross platform technology which is used to speed up response time. In ASP.NET the AJAX server controls are used to add script to your page which can be executed and also processed by the browser.

- The AJAX server controls contain methods and event handlers associated with each control similar to other server controls, which are processed on the server side.
- The Visual Studio contains a set of controls named as 'AJAX Extensions' located at toolbox.



Syllabus Topic : ScriptManager

3.7.1 The ScriptManager Control

- The ScriptManager control is considered as the most significant control.
- If other controls which are used on page requires to work properly then the ScriptManager Control have to be present on the page.

Syntax

```
<asp:ScriptManager ID="ScriptManager1"
runat="server">
</asp:ScriptManager>
```

- The ScriptManager control is responsible for taking care of the client-side script.

Syllabus Topic : Partial Refreshes

3.7.2 Partial Refreshes

Q. Write a short note on partial refreshes. (2 Marks)

- The partial refreshes can refreshes only specifies part of the page not the whole page at a time. The partial refreshes can be done with the help of UpdatePanel control.
- The UpdatePanel control is inherited from Control class and it is also called as container control. The UpdatePanel control behaves like a container for the child controls which are placed inside it.
- The UpdatePanel does not contain its own interface. When a control inside it triggers a post back, the UpdatePanel get involved to start the post asynchronously and update only that piece of the page.
- Consider an example, the update panel contains a button control and when it is clicked by user, only the controls which reside within the update panel will have an effect on it, the controls on the outside the panel will not have any effect. This process is known as partial post back or the asynchronous post back.

Example

- Create a new website and add script manager control from AJAX Extension tool. Insert an update panel. Place a button control along with a label control within the update panel control. Place another set of button and label outside the panel.
- The design view looks as follows :

- The source file is as follows

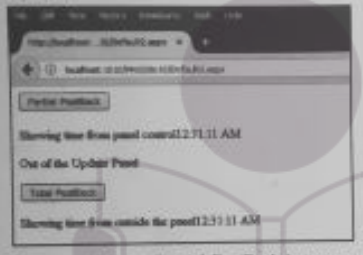
```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:ScriptManager ID="ScriptManager1" runat="server" />
    </div>

        <asp:UpdatePanel ID="UpdatePanel1" runat="server">
            <ContentTemplate>
                <asp:Button ID="btnpartial" runat="server"
onclick="btnpartial_Click" Text="Partial PostBack" />
                <br />
                <br />
                <asp:Label ID="labelpartial"
runat="server"></asp:Label>
            </ContentTemplate>
        </asp:UpdatePanel>

        <p> </p>
        <p>Out of the Update Panel</p>
        <p>
            <asp:Button ID="btntotal" runat="server"
onclick="btntotal_Click" Text="Total PostBack" />
```

```
</p>

<asp:Label ID="labeltotal"
runat="server"></asp:Label>
</form>
```

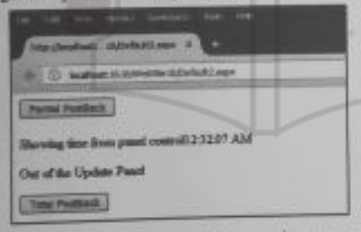
- Write following code on both button controls for the event handler :

```
string t1 = DateTime.Now.ToLongTimeString();
labelpartial.Text = "Showing time from panel control" + t1;
labeltotal.Text = "Showing time from outside the panel" +
t1;
```

- When you run the application and click on Total PostBack button you will get the updated time on both the labels.



- When you click on **Partial PostBack** button you will get the updated time only on partial label.



- A page can hold many update panels, every panel consist other controls such as grid and displaying different part of data.
- When a total post back is occurred, the update panel content is updated by default. You can change this default mode by changing the UpdateMode property of the control.

Syllabus Topic : Progress Notification

3.7.3 Progress Notification

Q. Explain progress notification. **(2 Marks)**

The progress notification can be achieved with the help of UpdateProgress control.

- The UpdateProgress control is useful because it provides a kind of feedback on the browser while one or more update panel controls are being updated.
- For example, while a user logged in or waiting for response from server while doing some database related task, it show "Loading page..." indicating the work is in progress.

☞ **Syntax**

```
<asp:UpdateProgressID="UpdateProgress1"
runat="server" DynamicLayout="true"
AssociatedUpdatePanelID="UpdatePanel1" >

  <ProgressTemplate>
    Loading your page...
  </ProgressTemplate>

</asp:UpdateProgress>
```

- The above code sets a message within the ProgressTemplate tag. You can also specify any image or other control in it.
- The UpdateProgress control is shown for each asynchronous postback unless it is assigned to a single update panel using the AssociatedUpdatePanelID property.

☞ **Properties of the UpdateProgress Control**

- Table 3.7.1 displays the properties of the updateProgress control.

Table 3.7.1

Sr. No.	Properties	Description
1.	AssociatedUpdate PanelID	This property is used to get and set the ID of the update panel which relates it with the UpdateProgress.
2.	Attributes	This property is used to get and set value of cascading style sheet (CSS) attributes of the UpdateProgress control.
3.	DisplayAfter	This property is used to get and set the time in milliseconds after which the progress template is showed. The default value of DisplayAfter property is 500 milliseconds.

Sr. No.	Properties	Description
4.	DynamicLayout	This property is used to state which progress template is rendered dynamically.
5.	ProgressTemplate	This property indicate the template showed at the time of asynchronous post back which takes more time than the DisplayAfter time.

☞ **Methods of the UpdateProgress Control**

- Table 3.7.2 displays the methods of the update progress control.

Table 3.7.2

Methods	Description
GetScriptDescriptors	This property is used to provide a list of components, actions, and client controls that are required for the UpdateProgress control's client functionality.
GetScriptReferences	This property is used to return a list of client script library dependencies for the UpdateProgress control.

Syllabus Topic : Timed Refreshes

3.7.4 Timed Refreshes

Q. Write a short note on timed refreshes.

- The main purpose of timer control is to start the post back automatically.
- This could be done in two way?

(1) Setting the Triggers property of the UpdatePanel control :

```
<Triggers>
  <asp:AsyncPostBackTrigger ControlID="autopanel"
EventName="Click" />
</Triggers>
```

(2) By placing a timer control inside the UpdatePanel we can make it to behave as a child control trigger. One timer can be the trigger for many UpdatePanels.

```
<asp:UpdatePanel ID="UpdatePanel1" runat="server"
UpdateMode="Always">

  <ContentTemplate>
    <asp:Timer ID="Timer1" runat="server"
Interval="1000">
    </asp:Timer>
    <asp:Label ID="Firstlable" runat="server"
Height="203px" style="width:405px" >
    </asp:Label>
  </ContentTemplate>

</asp:UpdatePanel>
```

Review Questions

Q. 1 What are the objects of ADO.net? Explain any one. **(Refer Section 3.2.1)** **(4 Marks)**

Q. 2 Explain Direct Data Access in detail. **(Refer Section 3.1.3)** **(10 Marks)**

Q. 3 Explain data binding in detail. **(Refer Section 3.2)** **(4 Marks)**

Q. 4 Explain different data controls of ADO.NET. **(Refer Section 3.3)** **(10 Marks)**

Q. 5 Explain XMLTextWriter class in detail. **(Refer Section 3.4.1)** **(4 Marks)**

Q. 6 Explain the different types of caching. **(Refer Section 3.5.2)** **(2 Marks)**

Q. 7 Explain various clauses used in LINQ. **(Refer Section 3.6.2)** **(4 Marks)**

Q. 8 Write a short note on timed refreshes. **(Refer Section 3.7.4)** **(4 Marks)**

□□□