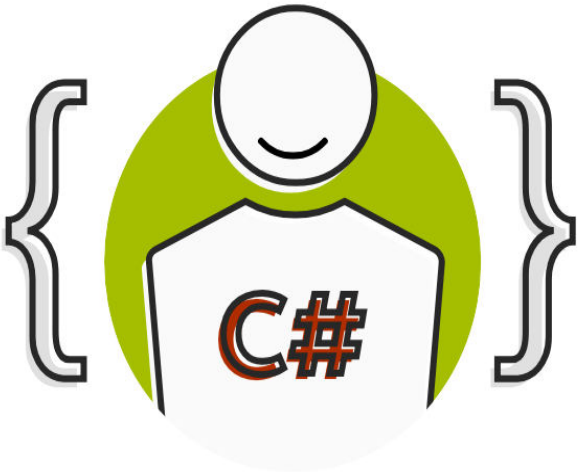
Gaurav.K.Sardhara

(Affiliated to Saurashtra University)

BCA – SEM-4 - C#.NET



Website: https://gauravprajapatidot.wordpress.com
Email : prajapatigks69@gmail.com

Gaurav.K.Sardhara

(Affiliated to Saurashtra University)

C#.Net

Chapter No.	Chapter Name	Topics
1	**.Net Framework & Visual Studio IDE , Language Basics**	• Introduction Microsoft .NET Framework • Features of the .NET Platform: • Components of the .NET Architecture: CLR,CTS,CLS,BCL/BCL , Name Space,Assembly, Meta Data, Managed code and Unmanaged code, • Reflection, • Just In Time Compilation, • Garbage Collection • Data Types (Value Type & Reference Type) • Boxing and UnBoxing • Operators (Arithmetic,Relational, Bitwise, etc.) • Arrays (One Dimensional,Rectangular, Jagged) • Decisions (If types and switchcase) • Loops (for, while, do..while,foreach
2	**Class &Inheritance, Property, Indexer, Pointers, Delegates, Event, Collections**	• Concept of Class, Object,Encapsulation, Inheritance,Polymorphism • Methods with “ref” and “out” parameters • Static and Non-Static Members • Constructors • Overloading Constructor, • Method Overloading • Operator Overloading • Inheritance • Sealed Class & Abstract Class • Overriding Methods • Interface and Interface inheritance • Creating and using Property • Creating and using Indexer • Creating and using Pointers (unsafe concept)

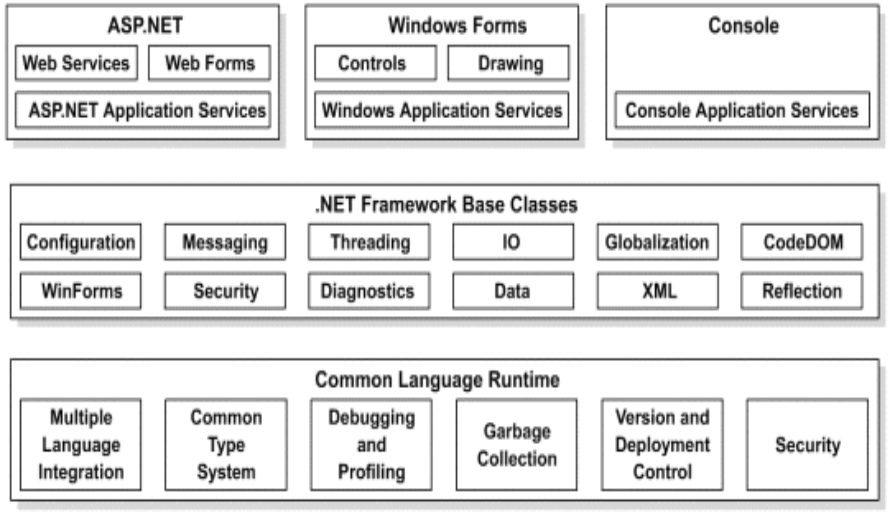
		• Creating and using Delegates (Single / Multicasting) • Creating and using Events with Event Delegate • Collections (ArrayList, • HashTable, Stack, Queue, SortedList) and their differences.
3	**Windows Programming**	• Creating Windows Application • MessageBox class with all types of Show() method • Basic Introduction to Form and Form Properties • Concept of Adding different Events with Event Parameters • Different Windows Controls • Label • TextBox • RadioButton • CheckBox • ComboBox • ListBox • PictureBox • ScrollBarTreeView • Menu (MenuStrip, ContextMenuStrip) • ToopStrip • Timer • Panel and GroupBox • Dialog Boxes (ColorDialog, FontDialog, ShowFileDialog and OpenFileDialog) • MDI Concept with MDI Notepad • Concept of Inheriting Form

4	**Database Programming with ADO.NET**	• Concept of Connected and Disconnected Architecture • Data Providers in ADO.NET • Connection Object • Connected Architecture • Command • Data Reader • Disconnected Architecture • Data Adapter • DataSetDataSet • DataTable • DataRow • DataColumn • DataRelation • DataView • Gridview Programming
5	**User Controls(Components), Crystal Reports, Setup Project**	• Creating User Controls with : • Property • Event • Method • Creating Crystal Reports • Report Sections • Formula, Special Fiend and Summary in Report • Creating Setup Project • File System Editor • User Interface Editor • Launch Conditions Editor

Chapter-1 :- Framework & Visual Studio IDE , Language Basics

.NET Framework :

- The .NET Framework is a service or platform for building, deploying, and running applications. The .NET Framework consists of 2 main parts: common language runtime and class libraries



What are the features of .NET Platform / .NET Framework?

- There are many features provided by .NET Framework which has made .NET popular and reliable in software development and web development industry. Following are features of .NET Platform / .NET Framework.

1) Multilanguage Development

C#.NET supports multiple languages. This is definitely one of the biggest advantages of .NET Framework because programmers having ability in their own languages, can use their skills in their languages. Another advantage of Multilanguage is that all are developed under same basic environment.

2) Multi-Device Development

Apart from that .NET supports multiple developments. You can create Mobile Application, PDA Application, etc.

3) Platform and Processor independence

Generally when you compile a code written in some language, it is converted directly to Native Code i.e. EXE or DLL. In C#.NET execution of programs is done in two process. First program is converted from language code to IL Code and then from IL Code to Native Code, which makes .NET application to become Platform and Processor independence.

4) Automatic memory management

Memory managed is always one of biggest headache of Developers. C#.NET handles memory managed by itself. Under Garbage Collection method, it automatically collects the objects which are no longer needed and removes it from memory

5) Easy Deployment

In many languages, Deployment is one of the tedious task. Using C#.NET application becomes easy to deploy. You can create Deployment project easily which helps to deploy application on target machines.

6) Distributed Architecture

C#.NET applications have capability to be executed on Distributed Architecture. You can create applications which can be executed on Distributed Architecture.

7) Interoperability with Unmanaged code

Because interaction between new and older applications is commonly required, the .NET Framework provides means to access functionality that is implemented in programs that execute outside the .NET environment. So, Interoperability with Unmanaged Code is provided.

8) Security

The design is meant to address some of the vulnerabilities, such as buffer overflows, that been exploited by malicious software. Additionally, .NET provides a common security model for all applications.

9) Performance and Scalability

As far as Performance and Scalability is concerned, .NET based applications give better performance in terms of memory, device management, etc. You can create Robust Application with full scalability provided by application.

10) XML Support

Today XML is used widely for the transportation of data between Client and Server via HTTP. Because XML works in Text which can be understood by all OS and Hardwares. .NET supports writing, manipulating and transforming of XML documents.

Components Of .NET Architecture:

CLR (Common Language Runtime):

- It is the execution engine for .NET Framework applications.
- It is the heart or backbone of the .NET.
- It's is the runtime engine provided by the .NET framework.
- It provides an infrastructure for run programs and allows them to communicate with other parts of the .NET framework.
- It provides a number of services, including the following:
- Code loading and execution
- Application memory isolation
- Verification of type safety
- Conversion of IL to native code
- Access to metadata
- Managing memory for managed objects
- Enforcement of code access security
- Exception handling, including cross-language exceptions

CTS (Common Type System):

- CTS allow programs written in different programming languages to easily share information.
- A class written in C# should be equivalent to a class written in vb.
- Languages must agree on the meanings of these concepts before they can integrate with one another.
- CTS forms a subset of CLS. This implies that all the rules that apply to CTS apply to CLS also.

- **It defines rules** that a programming language must follow to ensure that objects written in different programming languages can interact with each other.
- CTS provide cross language integration.
- The common type system supports two general categories of types:
  1) Value types
  2) Reference types Value types

CLS (Common Language Specification):

- CLS includes basic language features needed by almost all the applications.
- It serves as a guide for library writers and compiler writers.
- The Common Language Specification is a subset of the common type system.
- The Common Language Specification is also important to application developers who are writing code that will be used by other developers.

Assembly:

- An assembly is the primary building block of a .NET Framework **application.**
- An Assembly is a logical DLL.
- It consists of DLLs or executables.
- It is a collection of functionality that is built, versioned, and deployed as a single implementation unit (as one or more files).
- **All managed types and resources are marked either as accessible only within their implementation unit or as accessible by code outside that unit.**
- **There are mainly two types of assemblies:**
  1) **Private Assembly**
  2) **Shared Assembly**

- A private assembly is used only by a single application, and is stored in that application's install directory (or a subdirectory therein).
- A shared assembly is one that can be referenced by more than one application.
- In order to share an assembly, the assembly must be explicitly built for this purpose by giving it a cryptographically strong name (referred to as a strong name).
- By contrast, a private assembly name need only be unique within the application that uses it.

Meta Data:

- Metadata stored within the Assembly.
- NET records information about compiled classes as Metadata.
- Metadata means data about data.
- A .NET language compiler will generate the metadata and store this in the assembly.
- On the .NET Platform programs are compiled into .NET PE (Portable Executable) files.
- The header section of every .NET PE file contains a special new section for Metadata.
- Metadata is nothing but a description of every namespace, class, method, property etc. contained within the PE file.
- The CLR uses this metadata to

  Locate classes
  Load classes
  Generate native code
  Providesecurity

FCL (Framework Class Library):

- In C, <conio.h>, <stdio.h> etc. are header files. We add those header files in our program to use inbuilt functions.
- Same here, the .NET Framework are collection of classes or namespace that can be used to develop applications.
- The class library consists of data classes, XML classes, Web Forms classes and Windows Forms classes, Smart device classes, Input Output classes.
- Other name of FCL is BCL - Base class library.

Namespace:

- As above, the .NET Framework class library is collection of namespaces.
- Namespace is a logical naming scheme for types that have related functionality.
- Namespace means nothing but a logical container or partition.
- It's like Drives of our computer.
- Like my computer contain C:, D:, E: and F: . My F: contains songs and videos my C: contains installed file so on.
- For example, my friend wants songs. So I will directly go to my computer's F: because songs are placed there.

- **Common Namespaces**

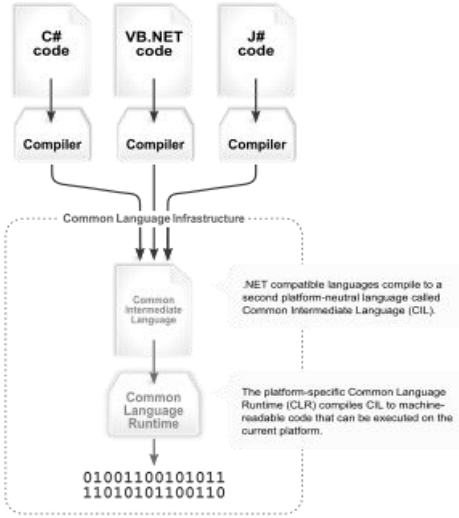
Namespace	What It Contains	Example Classes and Subnamespaces
System.Collections	Creation and management of various types of collections	Arraylist, Hashtable, SortedList
System.Data	Classes and types related to basic database management (see Chapter 11 for details)	DataSet, DataTable, DataColumn,
System.Diagnostics	Classes to debug an application and to trace the execution of code	Debug, Trace
System.IO	[illegible] writing to and from files and other data streams	File, FileStream, Path, StreamReader, StreamWriter
System.Math	Members to calculate common mathematical quantities, such as trigonometric and logarithmic functions	Sqrt (square root), Cos (cosine), Log (logarithm), Min (minimum)
System.Reflection	Capability to inspect metadata	Assembly, Module
System.Security	Types that enable security capabilities (see Chapter 24 for details)	Cryptography, Permissions, Policy

Garbage Collection (GC):

- Garbage Collection is a mechanism that allows the computer to detect when an object is no longer needed.
- It then automatically free up the memory used by that object.
- One of the advantages of CLR is automatic memory management that uses the garbage collection mechanism.
- The CLR's garbage collector (GC)manages the allocation and release of memory for an application.
- We do not have to write code to perform memory management tasks when you develop managed applications.
- There are some steps that are carried out while executing C#.NET program. They are :

Program Execution of C#.NET

- Select Compiler
- Compile Code to MSIL
- Convert MSIL to Native Code
- Execute Code



Step-1 :

First of all C#.NET finds that which program is being compiled. According to the program, C#.NET selects compiler. For C#.NET it selects C#C (C#Compiler), For C# it selects CSC (CSharpCompiler), and similar to different languages which are supported.

Step-2 :

After selecting compiler, it will convert the language code to Intermediate Language. So that, can be clubbed together with other ILs which are produced by other languages. This Intermediate Language (Code) is known as MSIL (MicroSoft Intermediate Language), CIL (Common Intermediate Language) or IL (Intermediate Language).

Step-3 :

After converting program to MSIL. It is converted to Native Code (Binary Code) so that it can be executed on machine.

Step-4 :

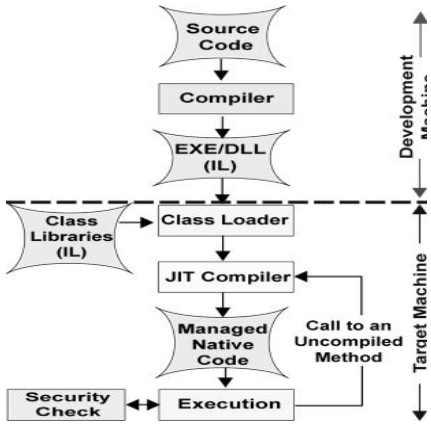
Now as MSIL is converted into Native Code (Binary Code), it can be executed.

Microsoft Intermediate Language (MSIL):

- MSIL is the CPU-independent instruction set into which .NET Framework programs are compiled.
- It contains instructions for loading, storing, initializing, and calling methods on objects.
- Combined with metadata and the common type system, MSIL allows for true cross-language integration.
- MSIL also known as CIL - Common Intermediate Language or IL-Intermediate Language

Just In Time Compiler (JIT):

- It stands for "just-in-time".
- It's a smart compiler.
- JIT does not compile whole program each time and every time. It compiles only that portion of the program which functions are called that time.And suppose Native code is already present then that data will not again compiled. If changes are made then possible that it will again generates MSIL to Native.
- Firstly any program compile by its own compiler then it will convert into MSIL



then with the help of JIT; MSIL compile into Native code but CLR does not convert whole MSIL code to Native code on Load time of that application; instead of it compiles the MSIL instructions as they are called.

- There are 3 types of JIT
  - **Pre-JIT** - It compiles complete program into native code in a single compilation cycle. This work is done at the time of deployment of the program.
  - **Econo-JIT** - It compiles only those methods that are called at runtime.
  - **Normal-JIT** - It's like Econo-JIT. The methods are compiled the 1st time they are stored in cache. When the same methods are called again the- compiled code from cache is used for execution.

Managed & Unmanaged Code:

Managed Code

- Managed Code is what C#.NET,J# and vb.net Compilers create.
- Code that targets the CLR (Common Language Runtime), the foundation of .NET Framework, is known as Managed Code.
- It compiles IL (Intermediate Code), not to machine code that could run directly on your computer. The IL is kept in a file called an Assembly which is known as **AssemblyInfo.C#** file., along with metadata that describes the classes, methods, and attributes of the code which you have created. You can copy it to another server / pc to deploy the assembly there – and often that coping is the only step required in the deployment.
- Managed code runs in the CLR. The runtime offers a wide variety of services to your running code.
- The managed code is always executed by a managed runtime execution environment rather than the operating system directly.
- Applications written in Java, C#, vb.NET, etc. target a runtime environment which manages the execution and the code written using these types of languages is known as Managed Code.

- Managed Code is always compiled to IL, so it provides platform independence.
- Managed Code provides information to allow the CLR to locate methods encoded in assembly modules, store and retrieve security information, handle exception, and walk the program stack.
- Managed code can access both managed data and unmanaged data.

Unmanaged Code

- Code that does not target the CLR (Common Language Runtime) is known as Unmanaged Code.
- Unmanaged code is what you use to make before C#.NET 2003 was released.
- Code that is directly executed by the OS is known as Unmanaged Code.
- Typically applications written in C# 6, C++, C, COM components, ActiveX components are an example of Unmanaged Code.
- Unmanaged Code typically targets the processor architecture and is always dependent on the computer architecture.
- Unmanaged code is always compiled to target a specific architecture and will only run on the intended platform, this means if you want to execute the same application on different machines, you need to recompile your program again.
- Unmanaged code is always compiled to the native code which is architecture specific.

Reflection:

- Reflection is a feature that enables you to obtain information about a type.
- Using this information you can construct and use objects at runtime.
- Reflection is the ability of managed code to read its own metadata for the purpose of encapsulates the type information.
- In other words, Reflection is a powerful mechanism because it allows learning and using the capabilities of types that are known only at runtime.
- Reflection is similar to C++ RTTI (Run Time Type Information).
- The "System.Reflection" namespace contains classes and interfaces that provide a value of methods, fields and types.
- C# code for reflection uses the type of operator and GetTypes() method.
- There are four key Reflection techniques :
    1. Obtaining information about methods.
    2. Invoking methods.
    3. Constructing objects.
    4. Loading types from assemblies.

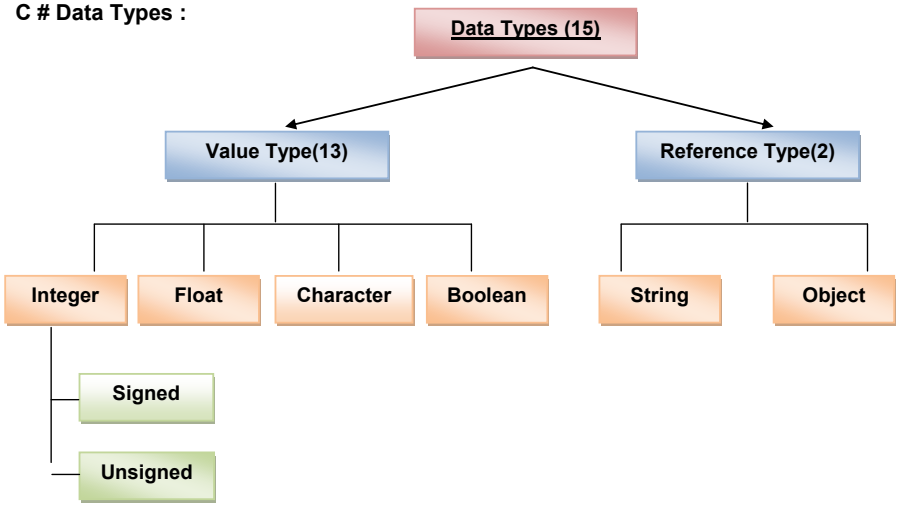
- **Example :**

```
using System.Reflection ;
class MyClass
{
        public int Add(int a, int b)
        {
                return (a+b);
        }
}
class Demo
{
        public static void Main( )
        {
            MyClass obj = new MyClass( );
            Type t = obj.GetType( );
            MethodInfo m = t.getMethod("Add");
            Object[ ] arr = new Object[ ]{3,5};
            Console.WriteLine("Type  Name="+t.FullName+"\nSum="+ m.Invoke(obj,arr));

            Console.ReadKey( ) ;
        }
}
```

- **Why we need Reflection (Use of Reflection) :**
  - To create an instance of a class whose name is not known until the runtime.
  - To invoke a method of an object even method is not known until the runtime.
  - To create a new array whose size and component type is not known until the runtime.
  - To define and execute dynamic method.
  - So, Reflection is a powerful mechanism because it allows to learn and use the capability of types that are known only at runtime

- **C # Data Types :**



- **Signed Integers (Value Type):**

Name	Full Name	Size	Range
Sbyte	System.Sbyte	8-Bits	-2^{7} to $2^{7}-1$
Short	System.Int16	16-Bits	-2^{15} to $2^{15}-1$
Int	System.Int32	32-Bits	-2^{31} to $2^{31}-1$
Long	System.Int64	64-Bits	-2^{63} to $2^{63}-1$

- **Unsigned Integers (Value Type):**

Name	Full Name	Size	Range
Byte	System.Byte	8-Bits	0 to $2^{8}-1$
Ushort	System.UInt16	16-Bits	0 to $2^{16}-1$
Uint	System.UInt32	32-Bits	0 to $2^{32}-1$
Ulong	System.UInt64	64-Bits	0 to $2^{64}-1$

- **Floating Point (Value Type):**

Name	Full Name	Size	Range
Float	System.Single	32-Bits	Single precision No.
Double	System.Double	64-Bits	Double precision No.
Decimal	System.Decimal	128-Bits	High precision No.

- **Character (Value Type):**

Name	Full Name	Size	Range
Char	System.Character	8-Bits	Any one character

- **Boolean (Value Type):**

Name	Full Name	Size	Range
Bool	System.Boolean	1-Bit	True(1) / False(0)

- **String (Reference Type):**

Name	Full Name	Size	Range
String	System.String	-	-

- **Object (Reference Type):**

Name	Full Name	Size	Range
Object	System.Object	-	-

- **Boxing and UnBoxing :**

  - **Boxing :** when an object refers to a value type, a process is known as "boxing".
    - In simple words, casting of value type to object type is called "boxing".
    - No explicit type casting is required, it occurs automatically.
    - It causes the value of value type tobe stored in object type.
    - Thus, value type is boxed inside the object.

- **UnBoxing :** the reverse process of "boxing" is called "unboxing".
  - In simple words, casting of object type to value type is called "unboxing".
  - This action is performed using explicit cast from object type to corresponding value type.
  - It is necessary to unbox object type to only that value type from which it was boxed.
  - Attempting to unbox an object into different type will result in runtime error

- Example of Boxing/UnBoxing is shown below.

```
class Demo
{
    Static void Main()
    {
        int X ;
        object Obj ;
```

- **Operators in C# :**

  - **Arithmetic Operators :**

Operator	Meaning
+	Addition
-	Subtraction / Unary minus
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement

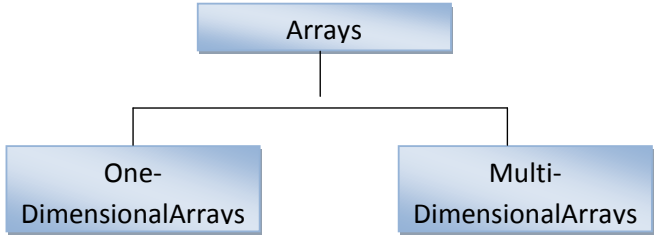
Relational and Logical Operators :

Operator	Meaning
==	Equal to
!=	Not Equal to
>	Greater than
<	Less than
>=	Greater than or Equal to
<=	Less than or Equal to

  - **Bitwise Operators :**

Operator	Meaning
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR (XOR)
>>	Shift right
<<	Shift left
~	One's complement (NOT)

- **Array in C# :**



  - An array is a collection of variables of the same type that are referred to by a common name.
  - In c#, array can have one or more dimensions.
  - Arrays are used for variety of purposes because they offer a convenient means of grouping together related variables.
  - The advantage of an array is that it organizes data in such a way that it can be easily manipulated.

- **One-dimensional Array :**
  - A one-dimensional array is a list of related variables.
  - Because arrays in c# are implemented as objects, two steps are needed to obtain an array for use it in program.
  - First, declare a variable that can refer to an array.
  - Second, create an instance of an array by use of "new".
  - Syntax or general form to declare one-dimensional array is as follows :

```
type[] array-name = new type [size] ;
```

  - Example : conceptually, the sample one-dimensional array looks like this :

A	B	C	D	E	F	G	H	I	J

Index : [0] [1] [2] [3] [4] [5] [6] [7] [8] [9]

- **Multi-dimensional Array :**If array has more then one row and column, its called Multi Dimension Array. There are two kinds of Multi Dimension Array In C# :
  - Rectangular Array
  - Jagged Array
  - A multi-dimensional array is an array that has two or more dimensions, and an individual element is accessed through the combination of two or more indices.
  - In two-dimensional array the location of any specific element is specified by two indices (x,y).
  - Two-dimensional array can be same as table, one index indicates the row, the other index indicates the column.
  - Syntax or general form to declare two-dimensional array is as follows :

```
type[ , ] array-name = new type [row-size,column-size] ;
```

  - To declare a two-dimensional array table of size 3, 5 :

```
int[ , ] table = new int [3,5] ;
```


	0	1	2	3	4
0	[0,0[0,0]	[0,1]	[0,2]	[0,3]	[0,4]
1	[1,0[1,0]	[1,1]	[1,2]	[1,3]	[1,4]
2	[2,0[2,0]	[2,1]	[2,2]	[2,3]	[2,4]

```
Array-name[row-index,column-index] = value ;
```

- **Jagged Array :**
  - C# allows creating special type of two-dimensional array called Jagged Array.
  - Jagged array is an "array of array" in which the length of each array can differ.
  - Thus, a jagged array can be used to create a table in which the lengths of the rows are not same.
  - Jagged array are declared by using sets of square brackets to indicate each dimension.
  - To declre two-dimensional jagged array :

```
type[ ][ ] array-name = new type [row-size][ ] ;
```

  - The size indicates number of rows in the jagged array.
  - Length of each row is vary :

```
array-name[0] = new type [column-size] ;
```

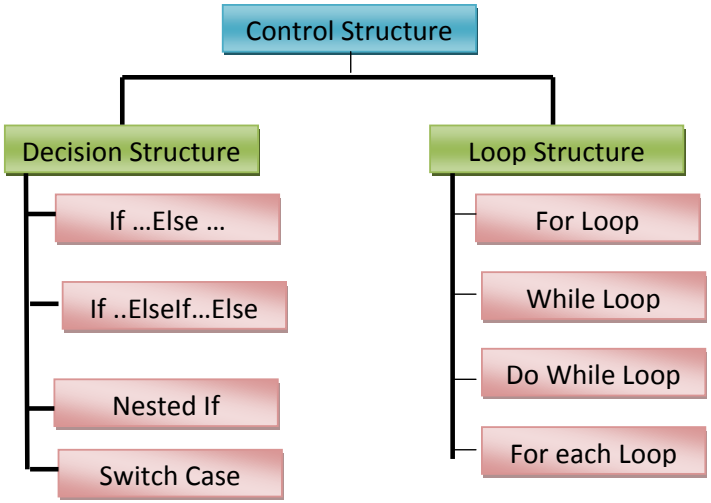
  - Example :

```
Int [ ][ ] table = new int [3][ ] ;
table[0] = new int [3] ;
table[1] = new int [2] ;
table[2] = new int [4] ;
```

  - This example will make a table of varing row size as shown below :

[0][0]	[0][1]	[0][2]	
[1][0]	[1][1]		
[2][0]	[2][1]	[2][2]	[2][3]

- **Control Structure of C# :**



- **The if statement :**
  - If statement checks the condition, if the condition is true, the target statement of if will be executed, otherwise if exists, and the target statements of else will be executed.
  - At no time both of them will be executed.
  - Either the statements of "if" or "else" will be executed.
  - "else" clause is optional.
  - The general form or syntax of "if" is as follows :


```
if (condition)
{
  statements;
}
else
{
  statements;
}
```

- **The if-else-if Ladder :**

  - This type of conditional structure is used to check multiple conditions.
  - The conditional expressions are evaluated from the top to downward.
  - As soon as a true condition is found, the statement associated with it is executed and the rest of statements are bypassed.
  - If none of the conditions is true, then the final "else" clause will be executed.
  - The final "else" is like a default condition.
  - If there is no final "else" and all other conditional tests fails, then no action will take place.
  - The general form or syntax is as follows :

```
if (condition1){ statement;  ......}
else if (condition2)
{
  statement;
  .......
}
else if (condition3)
{
  statement;
.......
}
.......
.......
else
{
  statement;
}
```

- **Nested ifs :**
  - A nested if is an if statement that is the target of another if or else.
  - In general nested if is a “if within if”.
  - The if which contains another if is called “Outer if”.
  - The if which is inside another if is called “Inner if”.
  - The general form or syntax is as follows :

```
if (condition)
{
  if (condition)
  {
    statement;
    ……
  }
  else
  {
    statement;
    …….
  }
}
else
{
  statement;
}
```



```
switch (expression)
{
  case constant1 :
                    Statements;
                    break ;
  case constant2 :
                    Statements;
                    break ;
  case constant3 :
                    Statements;
                    break ;
  …..
  …..
  default :
                   Statements;
                   break ;
}
```

  - The switch expression must be of an integer,char,byte,or string type.
  - The “default case” is executed if no “case” constant matches the expression.
  - The “default case” is optional.
  - The statements associated with any case are executed until the break is encountered.

- **The for loop :**
  - For loop is used to repeatedly execute a sequence of code.
  - The general form or syntax is as follows :

```
for (initialization ; condition ; increment)
{
    Statements;
}
```

- The “initialization” portion of the for loop sets a loop control variable to an initial value.
- The “condition” is a Boolean expression that tests the loop control variable.
- If test is “true”, the for loop continues to iterate.
- If it is “false”, the loop terminates.
- The “increment” expression determines how the loop control variable is changed each time the loop iterates.

- **The while loop :**

- Like other loop “while loop” is also used to repeatedly execute statements.
- The general form or syntax is as follows :

```
while (condition)
{
    Statements;
}
```


- Statement can be a single statement or a block of statements.
- Condition defines the condition that controls the loop and may be valid Boolean expression.
- The statement is executed while the condition is “true”.
- When the condition becomes “false”, program control passes to the line immediately following the loop.

- **The do-while loop :**
  - Unlike the for and while loops, in which the condition is tested at the top of the loop, the do-while loop checks its condition at the bottom of the loop.
  - This means that a do-while loop will always execute at least once.
  - The do-while loop executes as long as the conditional expression is true.
  - The general form or syntax is as follows :

```
do
{
    Statements;
}
while (condition) ;
```

- **The for each loop :**
  - for each loop is used to cycle through the elements of a collection.
  - A collection is a group of objects.
  - C# defines several types of collections, of which one is an array.
  - The general form or syntax of for each loop is as follows :

```
Foreach (type loopvariable in collection)
{
    Statements;
}
```

- The "type" and "loopvariable" specifies the datatype and the name of loop variable.
- The variable receives the value of next element in the collection each time the foreach loop iterates.
- When loop begins, the first element of the collection is obtained and assigned to loopvariable.
- Each subsequent iteration obtains the next element from the collection.
- The loop ends when there is no more elements to obtain.
- Thus, "foreach" loop cycles through the collection one element at a time, from start to finish.

➢ **Exception Handling in C# :**

- Exception means runtime errors.
- Exception handling is a technique to handle runtime errors without terminating execution of program.
- In c#, Exception Handling is managed via four keywords : try, catch, throw and finally.
- Program statements that are supposed to generate errors are contained within a try block.
- If an exception occurs within the try block, it is thrown.
- Exception is catched and handled by using catch block.
- System generated exceptions are automatically thrown by the c# runtime system.
- To manually throw an exception, the keyword "throw" is used.
- The general form of "try-catch" is shown below.


```
try
{
    //block of code to monitor for errors
}
catch (ExceptionType1 object)
{
    //handler for ExceptionType1
}
```

- Catch can not be used without try, and more than one catch clause can be associated with a try.
- The type of exception determines which catch is executed.
- When exception is caught, the exception variable "object" will receive its value.
- If no exception is thrown, then a try block ends normally, and all of the catch clauses are bypassed.
- To catch all exception with the use of single catch following syntax is used

```
try
{
    //block of code to monitor for errors
}
catch
{
    //handler for Exceptions
}
```

- **Using "finally" :**
  - Sometimes when a try/catch block is left and exception might teminates the current block.
  - To specify a block of code to execute when an exception can not catched by perticular "catch" clause and exception must be handled, the "finally" is used.
  - The general form of "try/catch" which includes "finally" is shown below

```
try
{
      //block of code to monitor for errors
}
catch
{
      //handler for Exceptions
}
...
...
finally
{
      // finally code
}
```

  - The nested try blocks means one try within another is also possible.

- **"throw" an exception manually :**
  - It is also possible to throw an exception manually by using the "throw" statement.
  - The general form is shown below :

```
try
{
      throw exception-object;
}
catch(exception-type)
{
      //handler for Exceptions }
```


- Commonly used exceptions are :
  - **DivideByZeroException**
    IndexOutOfRangeException
  - **OverflowException**
  - **InvalidCastException ,etc…**

Chapter :-2 Class &Inheritance, Property, Indexer,Pointers, Delegates,Event,Collections

- **Class :**
  - Classes are the basic ingredients of Object-Oriented languages.
  - It is used to define custom data and methods.
  - Classes can be declared by using the "class" keyword followed by the name of class and the brackets surrounding the body of the class.
  - Supports member variables and methods.

- **Object :**
  - An object is a variable or an instance of the type Class.
  - It can access all the members or properties of class by using dot (.) Operator.

- **Structure :**
  - It is the user defined datatype provided by c#.
  - Like classes, structures can also contain both data and method definitions.
  - Also like a class, a structure can contain constructors, constants, fields, methods, properties, indexers, operators and nested types.

- **Namespace :**
  - Namespace is used to organize classes and other types into a single hierarchical structure.
  - Namespace is the logical collection of related classes.
  - The proper use of namespaces will make classes easy to use and prevent collisions with classes written by other authors.
  - The namespace can contain classes and other namespaces.
  - In C# "System" is the root namespace.
  - All the data related classes are within System.Data namespace.
  - All Input/Output related classes are within System.IO namespace, etc.

- **"using" Keword :**
  - To use the classes within the particular namespace, the namespace must be inherited before the use.

  - The namespace is inherited in C# program by using "using" keyword.
  - EX : using System.IO;

- **Encapsulation** means that a group of related properties, methods, and other members are treated as a single unit or object..
- **Polymorphism** means that you can have multiple classes that can be used interchangeably, even though each class implements the same properties or methods in different ways.

- **Properties & Fields:**
  - Fields and properties represent information that an object contains. Fields are like variables because they can be read or set directly.
  - Both C# and Visual Basic allow you either to create a private field for storing the property value or use so-called auto-implemented properties that create this field automatically behind the scenes and provide the basic logic for the property procedures.

- **Methods & Events:**
  - A method is an action that an object can perform. A class can have several implementations, or overloads, of the same method that differ in the number of parameters or parameter types.
  - Events enable a class or object to notify other classes or objects when something of interest occurs. The class that sends (or raises) the event is called the publisher and the classes that receive (or handle) the event are called subscribers. For more information about events, how they are raised and handled.

  - To declare an event in a class, use the event (C# Reference) keyword.
  - To raise an event, invoke the event delegate.
  - To subscribe to an event, use the += operator; to unsubscribe from an event, use the -= operator.

- **Methods with "ref" and "out" parameters:**

Ref

The ref keyword is used to pass an argument as a reference. This means that when value of that parameter is changed in the method, it gets reflected in the calling method. An argument that is passed using a ref keyword must be initialized in the calling method before it is passed to the called method.

Out

The out keyword is also used to pass an argument like ref keyword, but the argument can be passed without assigning any value to it. An argument that is passed using an out keyword must be initialized in the called method before it returns back to calling method.

Example:

```
public class Example
 {
        public static void Main() //calling method
        {
                int val1 = 0; //must be initialized
                  int val2; //optional

                  Example1(ref val1);
                  Console.WriteLine(val1); // val1=1

                  Example2(out val2);
                  Console.WriteLine(val2); // val2=2
        }
        static void Example1(ref int value) //called method
         {
                  value = 1;
         }
         static void Example2(out int value) //called method
         {
                  value = 2; //must be initialized
         }
  }
```


- **Static and Non-Static Members:**

Static Members	Non-Static Members
1) Static members are one per class	1) Non-Static members are one per instance
2) Static members are accessed by their class name which encapsulates them	2) Non-Static members are accessed by object reference
3) Static members can not use non-static methods without instantiating an object	3) Non-Static members can use static members directly.

- **Constructor:**
  - Constructors are class methods that are executed automatically when an object of a given type is created. Constructors usually initialize the data members of the new object. A constructor can run only once when a class is created. Furthermore, the code in the constructor always runs before an other code in a class. However, you can create multiple constructor overloads in the same way as for any other method.

```
public class SampleClass
{
        public SampleClass()
        {
                // Add code here
        }
}
```

- **Overloading Constructor:**

Constructor overloading in c# is one type of static polymorphism. Using constructor overloading, any number of constructors can be defined for same class. But ensure each constructor must have different number and type of parameters defined.

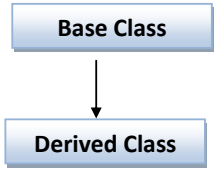
```
using System;
namespace ConsoleApplication3
{
	class Sample
	{
		public string param1, param2;

		public Sample()	// Default Constructor
		{
			param1 = "Hi";
			param2 = "I am Default Constructor";
		}
		public Sample(string x, string y)	// Declaring Parameterized
	constructor with Parameters
		{
			param1 = x;
			param2 = y;
		}
	}
	class Program
	{
		static void Main(string[] args)
		{
			Sample obj = new Sample();	// Default Constructor will Called
			Sample obj1=new Sample("Welcome","C#-GauravKSardhara");	//
		Parameterized Constructor will Called
			Console.WriteLine(obj.param1 + ", "+obj.param2);
			Console.WriteLine(obj1.param1 +" to " + obj1.param2);
			Console.ReadLine();
		}
	}
```

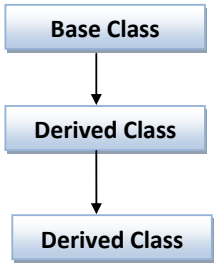

- **Inheritance in C# :**
  - Inheritance is the process by which one object can acquire the properties of another object.
  - Inheritance is one of the principle of object-oriented programming supports the concept of hierarchical(top-down) classification.
  - Using inheritance the object can inherits its general attributes from its parent.
  - It is the concept of reusing something that is already exists rather than to create the same again and again.
  - The class that is inherited is called “base class” or “parent class” and the class that does the inheriting is called “derieved class” or “child class”.

- **Types of Inheritance :**

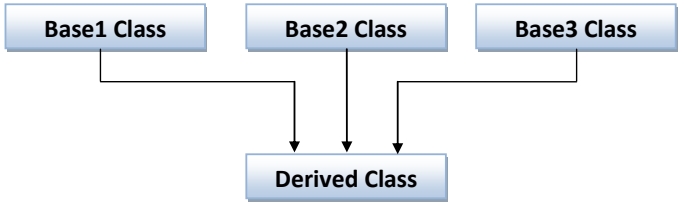
(1) Single Inheritance :



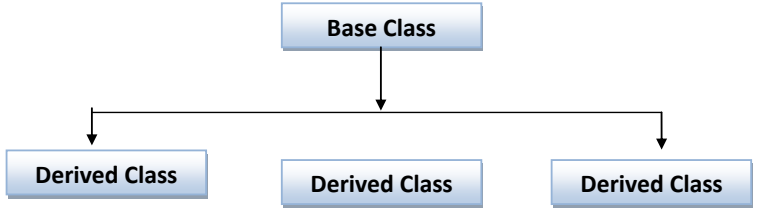
(2) Multi-Level Inheritance :



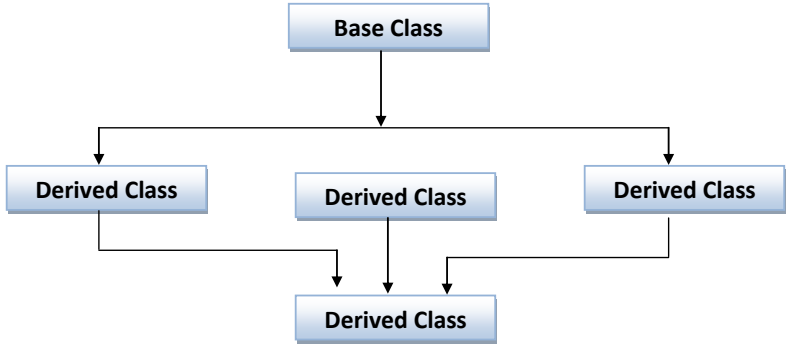
(3) Multiple Inheritance :



(4) Hierarchical Inheritance :



(5) Hybrid Inheritance :



- Multiple inheritances are not supported directly by c#.
  - The general form of inheritance is shown below.
  - In following example Class A is a “base” class and class B is “derived” class.
  - The object of Class B has accessibility of members(variables and methods) of Both Class A and Class B.

```
class A
{
    Variables and Methods of Class A;
}
class B : A
{
    Variables and Methods of Class B;
}
```

- **Sealed class :**
  - It is a powerful concept used when you want to prevent the concept of “Inheritance”.
  - If you want to create a class which will never inherited by any derived class then the class must be declared as “Sealed” class.
  - To prevent a class from being inherited, precede it declaration with “sealed” keyword.
  - Sealed class is totally opposite of “Abstract” class.
  - So it is illegal to declare a class as both “sealed” and “abstract”.
  - Sealed can also be used on virtual methods to prevent further overrides.
  - The example of sealed class is :

```
Sealed class A
{
    Variables and Methods of Class A;
}
```

- Now, inheriting above class A will causes the error.

➢ **Abstract class :**

- The abstract keyword enables you to create classes and class members that are incomplete and must be implemented in a derived class
- Classes can be declared as abstract by putting the keyword abstract before the class definition. For example:

```
public abstract class A
{
        // Class members here.
}
```

- An abstract class cannot be instantiated. The purpose of an abstract class is to provide a common definition of a base class that multiple derived classes can share.
- For example, a class library may define an abstract class that is used as a parameter to many of its functions, and require programmers using that library to provide their own implementation of the class by creating a derived class.
- Abstract classes may also define abstract methods. This is accomplished by adding the keyword abstract before the return type of the method.
- For example: abstract methods have no implementation, so the method definition is followed by a semicolon instead of a normal method block.
- Derived classes of the abstract class must implement all abstract methods. When an abstract class inherits a virtual method from a base class, the abstract class can override the virtual method with an abstract method. For example:


```
public class D
{
   public virtual void DoWork(int i)
   {
      // Original implementation.
   }
}

public abstract class E : D
{
   public abstract override void DoWork(int i);
}

public class F : E
{
   public override void DoWork(int i)
   {
      // New implementation.
   }
}
```

➢ **Overriding Methods:**

- ➢ Creating a method in derived class with same signature as a method in base class is called as method overriding.
- ➢ Same signature means methods must have same name, same number of arguments and same type of arguments.
- ➢ Method overriding is possible only in derived classes, but not within the same class.
- ➢ When derived class needs a method with same signature as in base class, but wants to execute different code than provided by base class then method overriding will be used.
- ➢ To allow the derived class to override a method of the base class, C# provides two options, **virtual** methods and **abstract** methods.

- **Examples for Method Overriding in C#**

```
using System;
namespace methodoverriding
{
    class BaseClass
    {
        public virtual  string YourCity()
        {
            return "New York";
        }
    }

    class DerivedClass : BaseClass
    {
        public override string YourCity()
        {
            return "Rajkot";
        }
    }
    class Program
    {

        static void Main(string[] args)
        {
            DerivedClass obj = new DerivedClass();
            string city = obj.YourCity();
            Console.WriteLine(city);
            Console.Read();
        }
    }
}
```

- Output

Rajkot

- **Overloading Methods :**
  - Overloading is the concept of polymorphism.
  - In c#, two or more methods within the same class can share the same name, as long as their parameter declarations are different.
  - C# supports several versions of the method that have different signatures(argument list and return type).
  - They must differ in their type and number of parameters.
  - When the method is called by its name, the method is called whose return type and parameter list is matched.
  - The example of method overloading is shown below.

```
class MyClass
{
    int add(int x, int y)
    {
        return (x+y);
    }
    float add(float x, float y)
    {
        return (x+y);
    }
}
class Demo
{
    static void Main( )
    {
        MyClass obj=new MyClass( );

        int result = obj.add(10,20); // calling first method

        float answer = obj.add(10.5,20.5); // calling second method

    }

}
```

- **Overloading Operators :**
  - Operator overloading is closely related to method overloading.
  - Operator overloading is a concept of polymorphism, used to expand the usage of operator.
  - To overload an operator the "operator" keyword is used to define an "operator method" which defines the action of the operator.
  - When operator is overloaded, none of its original meaning is lost.
  - There are two forms of operator methods :
    1. For "unary" operators.
    2. For "binary" operators.

- **Overloading "unary" operator :**
  - Operators which take single Operand are known as "unary" operators.
  - Ex. Unary + (++ Increment), Unary Minus (- - Decrement), etc.
  - The general form for unary operator overloading is shown below :

```
public static ret-type operator op(type operand)
{
    // operations
}
```

  - The "ret-type" in above syntax specifies the type of value returned by operation.
  - "ret-type" can be any type, but it is often of the same type as the class for which the operator is being overloaded.
  - For unary operator, the operand is passed in "operand".
  - Operand must be of the same type of class.
  - Note : operator methods must be both "public" and "static".

- **Overloading "binary" operator :**
  - Operators which take two Operands are known as "binary" operators.
  - Ex. Binary + (Addition), Binary Minus (Subtraction), etc.
  - The general form for binary operator overloading is shown below :

```
public static ret-type operator op(type1 operand1,type2 operand2)
{
    // operations
}
```

  - The "ret-type" in above syntax specifies the type of value returned by operation.
  - "ret-type" can be any type, but it is often of the same type as the class for which the operator is being overloaded.
  - For binary operator, the operands are passed in "operand1" and "operand2".
  - Atleast one of the Operands must be of the same type of its class.
  - Note : operator methods must be both "public" and "static".

Properties, Events and Methods in C# :

  - It is a one type of class member.
  - Property combines the field (variable) with the methods that access it.
  - A property consists of a name along with get and set accessors.
  - The accessors are used to get and set the value of variable.
  - Accessors are similar to the method except that it does not declare a returen type or parameters.
  - The key benefit of property is that its name can be used in expressions and assignments like a normal variable, and get and set are automatically invoked.
  - General form of property is :

```
type name
{
    get
    {
        // get accessor code
    }
    set

    {

        // set accessor code
```

- **Indexers :**
  - An indexer allows an object to be indexed like an array.
  - The main use of indexers is to support the creation of specialized arrays supports one or more constraints.
  - Indexers can have one or more dimensions.

- **Creating One dimensional Indexers :**
  - The general form of One-dimensional indexer is :

```
element-type this[int index]
{
    get
    {
        // return the value specified by index
    }
    set
    {
        // set the value specified by index
    }
}
```


- The "element-type" is the datatype of element of the indexer.
- So, each element accessed by the indexer will be of the type element-type of an array.
- The parameter "index" receives the index of the element being accessed.
- Because of index of an array is generally of type integer, the parameter "index" is of type "int".
- Inside the body of indexer two accessors "get" and "set" are defined.
- Accessors are similar to the method except that it does not declare a returen type or parameters.
- The Accessors are automatically called when the indexer is used, and both the accessors receive index as a parameter.
- If the indexer is on the left side of an assignment statement, then "set" accessor is called and the element specified by the index must be set by the implicit parameter called "value".
- Otherwise, the "get" accessor is called and the value associated with index must be returned.

- **Creating Multi dimensional Indexers :**
  - It is also possible to create indexer for multi-dimensional array.
  - The general form of One-dimensional indexer is :

```
element-type this[int index1,int index2]
{
    get
    {
        // return the value specified by [index1,index2]
    }
    set
    {
        // set the value specified by [index1,index2]
    }
```

- **Overloading Indexers :**
  - An indexer can also be overloaded.
  - The version of an indexer is invoked which matches the number of parameter and type.

```
element-type this[int index]
{
    get {
        // get accessor code  }
    set {
        // set accessor code  }
}
element-type this[double index]
{
    get {
        // get accessor code  }
    set {
        // set accessor code  }
```


- **Creating and Using Pointers (Unsafe Concept)**
  - A **pointer** is a variable whose value is the address of another variable i.e., the direct address of the memory location. Like any variable or constant, you must declare a pointer before you can use it to store any variable address.
  - Unsafe keyword denotes an unsafe context, which is required for any operation involving pointers. You can use unsafe modifier in the declaration of a type or a member . The entire textual extent of the type or member is therefore considered an unsafe context.
  - The general form of a pointer variable declaration is:

```
type *var-name;
```

- Following are valid pointer declarations:

```
int    *ip;    /* pointer to an integer */
double *dp;    /* pointer to a double */
float  *fp;    /* pointer to a float */
char   *ch     /* pointer to a character */
```

- The following example illustrates use of pointers in C#, using the unsafe modifier:

```
using System;
namespace UnsafeCodeApplication
{
   class Program
   {
            static unsafe void Main(string[] args)
            {
                    int var = 20;
                    int* p = &var;
                    Console.WriteLine("Data is: {0} ",  var);
                    Console.WriteLine("Address is: {0}",  (int)p);
                    Console.ReadKey();
                }
    }
}
```

Output:

Data is: 20
Address is: 99215364

- **Creating and using Delegates (Single / Multicasting):-**
  - In C#, delegates represent methods that are callable without knowledge of the target object. Delegates enable scenarios that some other languages have addressed with function pointers. However, unlike function pointers, delegates are object-oriented and type-safe.
  - There are three steps in defining and using delegates: declaration, instantiation, and invocation.

- **Delegate Declaration:**

```
public class DeligateClass
{
    public DeligateClass()
    {
    }
}
    // Declare a delegate
   public delegate void MessageHandler(string message);
    // The use of the delegate.


    public void Process(MessageHandler handler)
    {
        if (handler != null)
            handler("Begin Message");
        if (handler != null)
            handler("End Message");
    }
}
```


Function Class:

This class contains the functions will be called by delegates

```
public class FunctionClass
{
    public FunctionClass()
    {
    }
//This method will show alert message to user
    public static void AlertMessage(string s)
    {
        System.Windows.Forms.MessageBox.Show(s);
    }
//This method will write output to console
    public static void ConsoleMessage(string s)
    {
        Console.WriteLine(s);
    }
}
```

Instantiation and Invocation:

Client Application to use the defined Delegate

Simple Delegate

Create a windows form and add one button to it and paste the following code on its click event.

```
private void button1_Click(object sender, System.EventArgs e)
{
    //Instantiating a delegate
    DeligateClass dc = new DeligateClass();
    DeligateClass.MessageHandler ll = null;
    ll += new DeligateClass.MessageHandler(FunctionClass.AlertMessage);

    //Calling a delegate
    dc.Process(ll);
}
```

When you will click on button1, it will show 'Begin Message' and 'End Message' message box.

Multicast Delegate

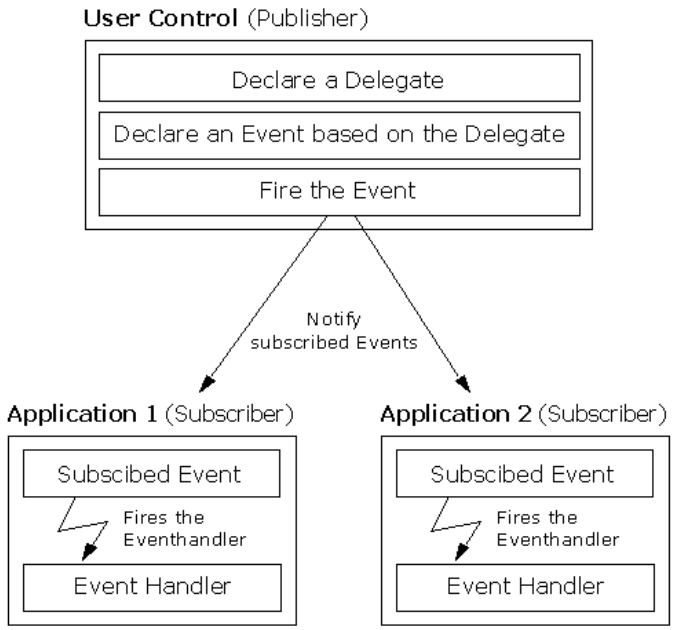
Add one more button to form with following code.

```
private void button2_Click(object sender, System.EventArgs e)
{
   DeligateClass dc = new DeligateClass();
   DeligateClass.MessageHandler ll = null;
   ll += new DeligateClass.MessageHandler(FunctionClass.AlertMessage);
   ll += new
   DeligateClass.MessageHandler(FunctionClass.ConsoleMessage);
   dc.Process(ll);
}
```

➢ Creating and using Events with Event Delegate:-

- ❖ A delegate in C# is similar to a function pointer in C or C++. Using a delegate allows the programmer to encapsulate a reference to a method inside a delegate object. The delegate object can then be passed to code which can call the referenced method, without having to know at compile time which method will be invoked.
- ❖ The Event model in C# finds its roots in the event programming model that is popular in asynchronous programming. The basic foundation behind this programming model is the idea of "publisher and subscribers."
- ❖ In this model, you have ***publishers*** who will do some logic and publish an "event." Publishers will then send out their event only to ***subscribers*** who have subscribed to receive the specific event.

- ❖ In C#, any object can *publish* a set of events to which other applications can *subscribe*. When the publishing class raises an event, all the subscribed applications are notified. The following figure shows this mechanism.



Conventions

- The following important conventions are used with events:
  - Event Handlers in the .NET Framework return void and take two parameters.

- The first paramter is the source of the event; that is the publishing object.
- The second parameter is an object derived from EventArgs.
- Events are properties of the class publishing the event.
- The keyword event controls how the event property is accessed by the subscribing classes.

➢ **Collections (ArrayList, HashTable, Stack, Queue, SortedList) and their differences.**

- **ArrayList** - automatically growing array. Adds more overhead. Can enum., probably slower than a normal array but still pretty fast. These are used a lot in .NET
- **Hashtable** - plain old hashtable. O(1) to O(n) worst case. Can enumerate the value and keys properties, and do key/val pairs.
- **Stack.** We push and pop elements onto the top of a Stack. With this functionality, we implement certain kinds of parsers. An array would work, but the Stack has a clearer interface.
- **Queue.** This removes elements that were added first (FIFO, first-in-first-out). So we keep items in order and service the ones that were added first, like a line at a government agency.
- **SortedList** - a sorted generic list. Slowed on insertion since it has to figure out where to put things. Can enum., probably the same on retrieval since it doesn't have to resort, but deletion will be slower than a plain old list.

Chapter-3 :- Window Programming

Windows Forms :

- In C# .Net it's these Forms with which we work. They are the base on which we build/ develop all our user interface and they come with a rich set of classes.
- Forms allow us to work visually with controls and other items from the toolbox.
- In C# .NET forms are based on the System.Windows.Forms namespace and the form class is System.Windows.Forms.Form.
- The form class is based on the Control class which allows it to share many properties and methods with other controls.
- Forms can be standard windows, multiple document interface (MDI) windows, dialog boxes, or display surfaces for graphical routines. The easiest way to define the user interface for a form is to place controls on its surface.
- Forms are objects that expose properties which define their appearance, methods which define their behavior, and events which define their interaction with the user.
- By setting the properties of the form and writing code to respond to its events, you customize the object to meet the requirements of your application.

MessageBox() :

- It will display message or string in another window.
- The following argument of messagebox can be given in the Msgbox().

- **Properties Of Msgbox :**

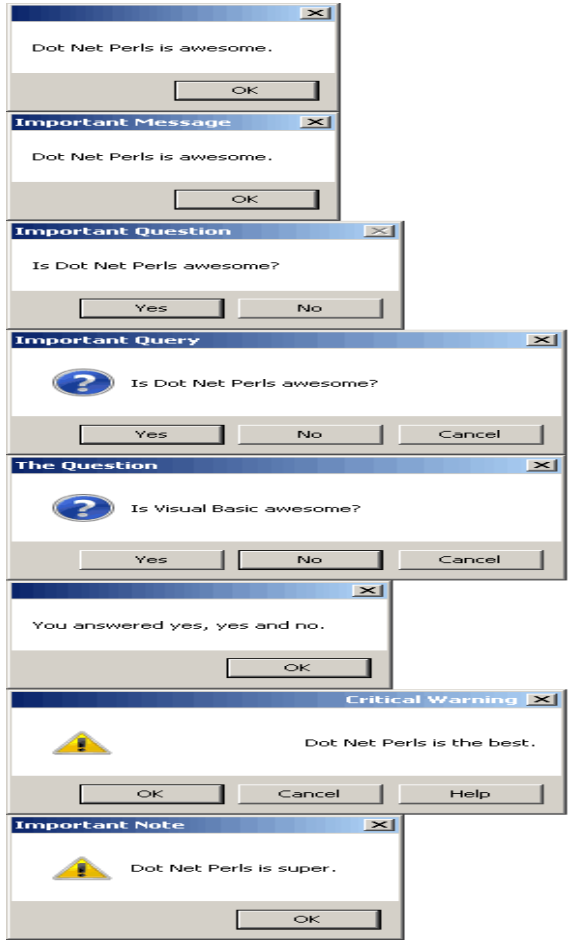
Name	Description
Text	It will display text given in this argument on the messagebox.
Caption	The text in this argument is display in the titlebar of the Msgbox.
Button	The argument values will be buttons to be displayed in the Msgbox.
Icon	The icon name even in the argument is displayed on the msgbox.
DefaultButton	The value set will specify each default button is to be display in the msgbox.

- Button property can have following values:
  1. **AbortRetryIgnore**
  2. **Ok**
  3. **OkCancel**
  4. **RetryCancel**
  5. **YesNo**
  6. **YesNoCancel**

- Icon can have following values:
  1. **warning**
  2. **None**
  3. **Exclamation**
  4. **Stop**
  5. **Error**
  6. **Hand**
  7. **Question**
  8. **Astrik**
  9. **Information**

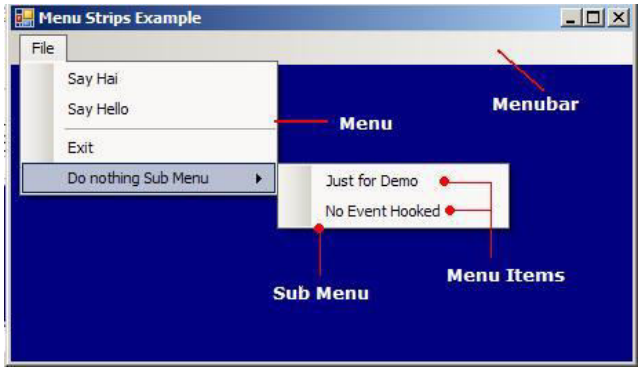
- Default Button can have following values:
  - **DefaultDesktopOnly**
  - **RightAlign**
  3. **RTLReading**
  4. **ServiceNotification**

➢ Dialog boxes interrupt users. They force users to respond before further action is taken. This is necessary in some situations.



➢ **Menu (MenuStrip, ContextMenuStrip) :**

- ❖ Strip is a relatively narrow piece of something. Dot net has three important strip controls namely ***MenuStrip***, ***StatusStrip*** and ***ToolStrip***. In this article we will start with the ***menustrip*** control.
- ❖ All the strips controls accommodate some other UI elements in it. A menu strips allows you to add Menu and Menu allows you to add menu items. Similarly the toolstrip control allows you add one or more tool bar buttons in it. OK, let us go to the MenuStrip. Look at the below screen shot:

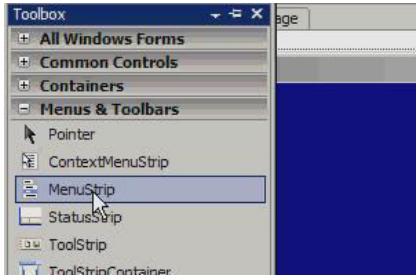


- ❖ A Menubar, dot net calls it Menu strip can accommodate multiple ***Menus***. In the above picture only File menu is added to it. ***Menu items*** are usually added to the menu. Say for example in the above picture, the ***Exit*** and ***Just for Demo*** are menu items. Menu items are usually linked to the command handler and when you click the menu item the corresponding handler gets called.
- ❖ If a menu item is linked with one more menu, then the resulting menu is called as Sub-Menu. The above picture shows one such sub-menu with two menu items in it.

- ❖ In this example, we will create the above shown form with menu and menu items. Then we will provide some simple handler for it.

2. Adding menu Strips control

- ❖ The menu strip is grouped in the toolbox under the Menus & Toolbars group. This is shown in the below picture. To use menustrip in the form drag the menustrip from the toolbox and drop it to the form.



- ❖ In the below video, a menu strip is added to the form. To add a ***separator*** in the menu just type the ***hyphen*** (-) for the menu name.
- ❖ The placement ***ampersand*** (&) in the menu item name displays underline for letter next to it.
- ❖ This under line is displayed at runtime when alt key is pressed and typing the letter with underline (alt key is still kept down) will call the menu item handler.
- ❖ The usage of ***ampersand*** in the menu item with alt key is shown below:

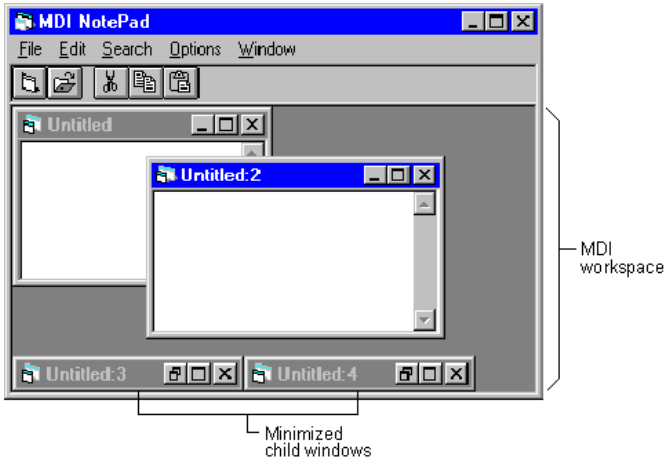
- So in the above picture in place of clicking the Exit menu item, you can call the handler by using keys ***Alt+F,X***

➢ **MDI Concept with MDI Notepad**

- The multiple-document interface (MDI) allows you to create an application that maintains multiple forms within a single container form. Applications such as Microsoft Excel and Microsoft Word for Windows have multiple-document interfaces.
- An MDI application allows the user to display multiple documents at the same time, with each document displayed in its own window. Documents or *child windows* are contained in a *parent window*, which provides a workspace for all the child windows in the application. For example, Microsoft Excel allows you to create and display multiple-document windows of different types. Each individual window is confined to the area of the Excel parent window. When you minimize Excel, all of the document windows are minimized as well; only the parent window's icon appears in the task bar.
- A child form is an ordinary form that has its MDIChild property set to True. Your application can include many MDI child forms of similar or different types.
- At run time, child forms are displayed within the *workspace* of the MDI parent form (the area inside the form's borders and below the title and menu bars).

When a child form is minimized, its icon appears within the workspace of the MDI form instead of on the taskbar, as shown in Figure 6.4.

Figure 6.4 Child forms displayed within the workspace of the MDI form



Note Your application can also include standard, non-MDI forms that are not contained in the MDI form. A typical use of a standard form in an MDI application is to display a modal dialog box.

- An MDI form is similar to an ordinary form with one restriction. You can't place a control directly on a MDI form unless that control has an Align property (such as a picture box control) or has no visible interface (such as a timer control).

Creating an MDI Application

Use the following procedure to create an MDI form and its child forms.

To create an MDI application

1. Create an MDI form.

From the **Project** menu, choose **Add MDI Form**.

Note An application can have only one MDI form. If a project already has an MDI form, the Add MDI Form command on the Project menu is unavailable.

2. Create the application's child forms.

   To create an MDI child form, create a new form (or open an existing one) and set its MDIChild property to True.

Concept of Inheriting Form:-

- Creating new Windows Forms by inheriting from base forms is a handy way to duplicate your best efforts without going through the process of entirely recreating a form every time you require it.
- For more information about inheriting forms at design time using the Inheritance Picker dialog box and how to visually distinguish between security levels of inherited controls, see How to: Inherit Forms Using the Inheritance Picker Dialog Box.
- Note In order to inherit from a form, the file or namespace containing that form must have been built into an executable file or DLL. To build the project, choose Build from the Build menu. Also, a reference to the namespace must be added to the class inheriting the form. The dialog boxes and menu commands you see might differ from those described in Help depending on your active settings or edition. To change your settings, choose Import and Export Settings on the Tools menu. For more information, see Customizing Development Settings.

To inherit a form programmatically

1. In your class, add a reference to the namespace containing the form you wish to inherit from.
2. In the class definition, add a reference to the form to inherit from. The reference should include the namespace that contains the form, followed by a period, then the name of the base form itself.

Controls used in Form :

- Controls are an extremely important part of any interactive application.
- They give information to the user (Label, ToolTip, TreeView, PictureBox, etc.) and organize the information so that it's easier to understand (GroupBox, Panel, TabControl).
- They enable the user :
  - To enter data (TextBox, RichTextBox, ComboBox, MonthCalendar)
  - To select options (RadioButton, CheckBox, ListBox)
  - To control the application (Button, MenuStrip, ContextMenuStrip)
  - To interact with objects outside of the application (OpenFileDialog, SaveFileDialog, PrintDocument, PrintPreviewDialog)
  - Some controls also provide support for other controls (ImageList, ToolTip, ContextMenuStrip, and ErrorProvider).

Before going for specific control, here is the list of some common Properties, Methods and Events.

Common Properties :

Name	Description
ContextMenuStrip	Gets or sets the ContextMenuStrip associated with this control.
Dock	Gets or sets which control borders are docked to its parent control and determines how a control is resized with its parent.
Enabled	Gets or sets a value indicating whether the control can respond to user interaction.
Font	Gets or sets the font of the text displayed by the control.
Name	Gets or sets the name of the control.
TabIndex	Gets or sets the tab order of the control within its container.
TabStop	Gets or sets a value indicating whether the user can give the focus to this control using the TAB key.
Visible	Gets or sets a value indicating whether the control and all its child

	controls are displayed.
Size	Gets or sets the height and width of the control.

- **Common Methods :**

Name	Description
Focus( )	Sets input focus to the control.

- **Common Events :**

Name	Description
FontChanged	Occurs when the Font property value changes.
GotFocus	Occurs when the control receives focus.
LostFocus	Occurs when the control loses focus.
MouseDown	Occurs when the mouse pointer is over the control and a mouse button is pressed.
MouseEnter	Occurs when the mouse pointer enters the control.
MouseMove	Occurs when the mouse pointer is moved over the control.
MouseUp	Occurs when the mouse pointer is over the control and a mouse button is released.
MouseWheel	Occurs when the mouse wheel moves while the control has focus.
Resize	Occurs when the control is resized.

- **Basic GUI Controls :**
- **Label :**

- You use labels for just what they sound like—to label other parts of your application. Labels usually are used to display text that cannot be edited by the user. Your code can change the text displayed by a label.
- The caption for a label is stored in the **Text** property. Because you can change that caption in code, labels can act a little like non-editable text boxes, displaying text and messages to the user. The **TextAlign** (formerly **Alignment**) property allows you to set the alignment of the text within the label.

- **Properties of Label :**

Name	Description
AutoSize	Gets or sets a value indicating whether the control is automatically resized to display its entire contents.
Left	Gets or sets the distance, in pixels, between the left edge of the control and the left edge of its container's client area.
Text	Gets or sets the text associated with this control.
TextAlign	Gets or sets the alignment of text in the label.

- **Methods of Label :**

ResetText( )	Resets the Text property to its default value.

- **Events of Label :**

Name	Description
AutoSizeChanged	Occurs when the value of the AutoSize property changes.
VisibleChanged	Occurs when the Visible property value changes.

- **TextBox :**
  - Windows users should be familiar with textboxes. This control looks like a box and accepts input from the user.
  - The TextBox is based on the TextBoxBase class which is based on the Control class.
  - TextBoxes are used to accept input from the user or used to display text.
  - By default we can enter up to 2048 characters in a TextBox but if the Multiline property is set to True we can enter up to 32KB of text.
  - The TextBox control allows the user to enter text in an application. This control has additional functionality that is not found in the standard Windows text box control, including multiline editing and password character masking.

  - **Properties of TextBox :**

Name	Description
MaxLength	Gets or sets the maximum number of characters the user can type or paste into the text box control.
Multiline	Gets or sets a value indicating whether this is a multiline TextBox control.
PasswordChar	Gets or sets the character used to mask characters of a password in a single-line TextBox control.
ReadOnly	Gets or sets a value indicating whether text in the text box is read-only.
ScrollBars	Gets or sets which scroll bars should appear in a multiline TextBox control.
Text	Gets or sets the current text in the TextBox.
TextAlign	Gets or sets how text is aligned in a TextBox control.
WordWrap	Indicates whether a multiline text box control automatically wraps words to the beginning of the next line when necessary.
SelectedText	Gets or sets a value indicating the currently selected text in the

	control.
SelectionLength	Gets or sets the number of characters selected in the text box.

  - **Methods of TextBox :**

Name	Description
Clear( )	Clears all text from the text box control.
Copy( )	Copies the current selection in the text box to the Clipboard.
Cut( )	Moves the current selection in the text box to the Clipboard.
Paste( )	Replaces the current selection in the text box with the contents of the Clipboard.
SelectAll( )	Selects all text in the text box.
DeselectAll( )	Specifies that the value of the SelectionLength property is zero so that no characters are selected in the control.
Undo( )	Undoes the last edit operation in the text box.

  - **Event of TextBox :**

Name	Description
KeyDown	Occurs when a key is pressed while the control has focus.
KeyPress	Occurs when a key is pressed while the control has focus.
KeyUp	Occurs when a key is released while the control has focus.
MultilineChanged	Occurs when the value of the Multiline property has changed.
TextChanged	Occurs when the Text property value changes.

- **RichTextBox :**
  - The RichTextBox control is one of the most powerful controls provided in C#.Net.
  - In a nutshell, it's a text box that's able to display text stored in Rich Text Format (RTF), a standard format recognized by virtually all word processors, including Microsoft WordPad (not surprisingly, since WordPad internally uses the RichTextBox control).
  - This control supports multiple fonts and colors, left and right margins, bulleted lists, and more.
  - The RichTextBox control is data-aware and therefore exposes the usual Dataxxxx properties that let you bind the control to a data source. In other words, you can write entire TXT or RTF documents in a single field of a database.
  - **Changing character attributes** :
  - The RichTextBox control exposes many properties that affect the attributes of the characters in the selected text: These are FontName, FontSize, Color, Bold, Italic, Underline, etc.
  - Their names are self-explanatory, so won't describe what each one does. You might find it interesting to note that all of the properties work as they would within a regular word processor.
  - It supports the properties, events, and methods of TextBox control.

- **RadioButton :**
  - The RadioButton control can display text, an Image, or both.
  - When the user selects one option button (also known as a radio button) within a group, the others clear automatically. All RadioButton controls in a given container, such as a Form, constitute a group.
  - To create multiple groups on one form, place each group in its own container, such as a GroupBox or Panel control.
  - RadioButton and CheckBox controls have a similar function: they offer choices a user can select or clear.
  - The difference is that multiple CheckBox controls can be selected at the same time, but option buttons are mutually exclusive. Use the Checked property to get or set the state of a RadioButton.

  - The option button's appearance can be altered to appear as a toggle-style button or as a standard option button by setting the Appearance property.
  - **Properties of RadioButton :**

Name	Description
Checked	Gets or sets a value indicating whether the control is checked.
Enabled	Gets or sets a value indicating whether the control can respond to user interaction.

  - **Methods of RadioButton :**
  - RadioButton supports all the general methods, which are already discussed in the beginning.
  - **Events of RadioButton :**

Name	Description
CheckedChanged	Occurs when the value of the Checked property changes.
Click	Occurs when the control is clicked.

- **CheckBox :**
  - Checkboxes are also familiar controls—you click a checkbox to select it, and click it again to deselect it. When you select a checkbox, a check appears in it, indicating that the box is indeed selected. You use a checkbox to give the user an option, such as true/false or yes/no. The checkbox control can display an image or text or both.
  - RadioButton and CheckBox controls have a similar function: they offer choices a user can select or clear. The difference is that multiple CheckBox controls can be selected at the same time, but option buttons are mutually exclusive.
  - **Properties of CheckBox :**

Name	Description
Checked	Gets or sets a value indicating whether the control is checked.

Enabled	Gets or sets a value indicating whether the control can respond to user interaction.

- **Methods of CheckBox :**

Name	Description
AddText( )	Adds a specified text string to the CheckBox.
Enabled( )	Gets or sets a value indicating whether the control can respond to user interaction.

- **Events of CheckBox :**

Name	Description
Checked	Occurs when the Checkbox is checked.
Unchecked	Occurs when the Checkbox is unchecked.

➢ **ListBox :**

- As you know, list boxes display a list of items from which the user can select one or more. If there are too many items to display at once, a scroll bar automatically appears to let the user scroll through the list.
- The items in list boxes are stored in the **Items** collection; the **Items.Count** property holds the number of items in the list. (The value of the **Items.Count** property is always one more than the largest possible **SelectedIndex** value because **SelectedIndex** is zero-based.) To add or delete items in a **ListBox** control, you can use the **Items.Add**, **Items.Insert**, **Items.Clear,** or **Items.Remove** methods. You also can add a number of objects to a list box at once with the **AddRange** method. Or you can add and remove items to the list by using the **Items** property at design time.

- You also can support multiple selections in list boxes. The **SelectionMode** property determines how many list items can be selected at a time; you can set this property to **None**, **One**, **MultiSelect**, or **MultiExtended**:

  - **MultiExtended—** Multiple items can be selected, and the user can use the Shift, Ctrl, and arrow keys to make selections.
  - **MultiSimple—** Multiple items can be selected.
  - **None—** No items may be selected.
  - **One—** Only one item can be selected.

- When you support multiple selections, you use the **Items** property to access the items in the list box, the **SelectedItems** property to access the selected items, and **SelectedIndices** property to access the selected indices.

- **Properties of ListBox :**

Name	Description
DataSource	Gets or sets the data source for this ListBox.
DropDownStyle	Gets or sets a value specifying the style of the combo box. The DropDownStyle property specifies whether the list is always displayed or whether the list is displayed in a drop-down. The DropDownStyle property also specifies whether the text portion can be edited.
SelectionMode	As listboxes can be used for Multiple selection. This specified how many list items can be selected at a time. You can set this property to **None**, **One**, **MultiSelect**, or **MultiExtended**.
SelectedIndex	Gets or sets the index specifying the currently selected item. If more then one items are selected it gives index no. of first item.
SelectedItem	Gets or sets currently selected item in the ListBox. If more then one items are selected it gives first selected item.
SelectedText	Gets or sets the text that is selected in the editable portion of a ListBox.

SelectedIndices	Similar to SelectedIndex property. SelectedIndex property gives index no. of first selected item where as SelectedIndices gives you index nos of all the selected items in the ListBox.
SelectedItems	It gives list of all selected item. By using loop of SelectedItems you can find the selected items in ListBox.
SelectionLength	Gets or sets the number of characters selected in the editable portion of the combo box.
SelectionStart	Gets or sets the starting index of text selected in the combo box.
Sorted	Gets or sets a value indicating whether the items in the combo box are sorted.
Items.Count	Displays the no. of items available in ListBox.

- **Methods of ListBox :**

Name	Description
FindString(String)	Returns the index of the first item in the ListBox that starts with the specified string.
Items.Add	Allows you to add item to the ListBox. The added item is kept last if sorted property is not set to true.
Items.Insert	Inserts (Adds) the item at specified position in ListBox.
Items.Remove	Allows you to remove particular item by specifying text of item.
Items.RemoveAt	Allows you to remove particular item by specifying index of item.
Items.Clear	Clears the items which are available in ListBox.
Items.IndexOf	Gives the index no. of specified item in ListBox

- **Events of ListBox :**

Name	Description
Click	Occurs when the control is clicked.
SelectedIndexChanged	Occurs when the SelectedIndex property has changed.
SelectedValueChanged	Occurs when the SelectedValue property changes.
TextChanged	Occurs when the Text property value changes.

- **CheckedListBox :**
  - CheckedListBox is the ordinary ListBox with checkbox for each items in the list.
  - It supports all properties, methods, and events of ListBox.
  - It provides the facility to check the item from the list.

- **ComboBox :**
  - The Windows forms combo box control is used to display data in a drop-down combo box. The combo box is made up of two parts: The top part is a text box that allows the user to type in all or part of a list item. The other part is a list box that displays a list of items from which the user can select one or more. You can allow the user to select an item from the list, or enter their own data.
  - ComboBox is called as List Control which is used to display list of items.

  - **Properties of ComboBox :**

Name	Description
DataSource	Gets or sets the data source for this ComboBox.
DropDownStyle	Gets or sets a value specifying the style of the combo box. The DropDownStyle property specifies whether the list is always displayed or whether the list is displayed in a drop-down. The DropDownStyle property also specifies whether the text portion can be edited.

SelectedIndex	Gets or sets the index specifying the currently selected item.
SelectedItem	Gets or sets currently selected item in the ComboBox.
SelectedText	Gets or sets the text that is selected in the editable portion of a ComboBox.
SelectionLength	Gets or sets the number of characters selected in the editable portion of the combo box.
SelectionStart	Gets or sets the starting index of text selected in the combo box.
Sorted	Gets or sets a value indicating whether the items in the combo box are sorted.
Items.Count	Displays the no. of items available in ComboBox.

- **Methods of ComboBox :**

Name	Description
FindString(String)	Returns the index of the first item in the ComboBox that starts with the specified string.
Items.Add	Allows you to add item to the Combobox. The added item is kept last if sorted property is not set to true.
Items.Insert	Inserts (Adds) the item at specified position in ComboBox.
Items.Remove	Allows you to remove particular item by specifying text of item.
Items.RemoveAt	Allows you to remove particular item by specifying index of item.
Items.Clear	Clears the items which are available in ComboBox.
Items.IndexOf	Gives the index no. of specified item in ComboBox

- **Events of ComboBox :**

Name	Description
Click	Occurs when the control is clicked.
SelectedIndexChanged	Occurs when the SelectedIndex property has changed.
SelectedValueChanged	Occurs when the SelectedValue property changes.
TextChanged	Occurs when the Text property value changes.

➢ **ListView :**

- If tree views are all about displaying node hierarchies, like the folder hierarchy on a disk, then list views are all about displaying lists of items. You can see a list view in the right pane in the Windows Explorer (the part that displays what files are in a folder).
- You can add list views to a Windows form just like any other control—just drag it from the toolbox to the form at design time. To add items to a list box, open the **Items** property in the Properties window.
- There are four views you can use in a list view. You select one by assigning a value (**View.LargeIcon**, **View.SmallIcon**, **View.List**, or **View.Details**) to the list view's **View** property:
- **Large icon mode**—displays large icons (large icons are 32×32 pixels) next to the item text.
- **Small icon mode**—is the same as the large icon mode except that it displays items using small icons (small icons are 16×16 pixels).
- **List mode**—displays small icons, and always in one column.
- **Report mode** (also called the details mode)—displays items in multiple columns, displaying column headers and fields.

- **Properties of ListView :**

Name	Description
CheckBoxes	Gets/sets if every item should show a checkbox.
CheckedIndices	Gets the indices of currently checked items.
CheckedItems	Gets the currently checked items.
Columns	Gets a collection of columns.

FullRowSelect	Gets/sets whether selecting an item will select the entire row.
GridLines	Gets/sets whether grid lines are drawn between items and their subitems.
Items	Gets the list items.
LargeImageList	Gets/sets the ImageList for large icon view.
MultiSelect	Gets/sets whether multiple items can be selected.
SelectedIndices	Gets the indices of the selected items.
SelectedItems	Gets the selected items.
SmallImageList	Gets/sets the small icon image list.
Sorting	Gets/sets the sort order for the items.
View	Gets/sets the current view mode.

- **Methods of ListView :**

Name	Description
ArrangeIcons	Arranges the displayed items in Large Icon or Small Icon view.
Clear	Removes all items from the list view.
GetItemAt	Gets the item corresponding to the given X,Y coordinate.

Events of ListView :

Name	Descrption
ColumnClick	Occurs when a column is clicked.
ItemCheck	Occurs when an item is checked.
SelectedIndexChanged	Occurs when the selected index changes.

- **TreeView :**
  - A **TreeView** control displays a hierarchical list of **Node** objects, each of which consists of a label and an optional bitmap. A **TreeView** is typically used to display the headings in a document, the entries in an index, the files and directories on a disk, or any other kind of information that might usefully be displayed as a hierarchy.
  - You use a tree view to display a hierarchy of nodes. Each node is not only displayed visually, but also can have child nodes. An example of this is the Windows Explorer, which uses a tree view in its left pane to display the hierarchy

of folders on disk. You can expand and collapse parent nodes by clicking them; when expanded, their children are also visible.

- There are 3 types of nodes in tree view as follows :
- **Root Node** : The node which is at root level is called as Root Node
- **Parent Node**: The node which is a child of Root Node and has got its children is called Parent Node.
- **Child Node** : The node which comes under Parent Node and is the leaf node, is Called as Child Node. It is also known as Leaf Node.
- After creating a **TreeView** control, you can add, remove, arrange, and otherwise manipulate **Node** objects by setting properties and invoking methods. You can programmatically expand and collapse **Node** objects to display or hide all child nodes. Three events, the Collapse, Expand, and NodeClick event, also provide programming functionality.

- **Properties of TreeView :**

Name	Description
HasChildren	Gets a value indicating whether the control contains one or more child controls.
CheckBoxes	Gets/sets whether checkboxes should be displayed next to tree nodes.
ImageList	Gets or sets the ImageList that contains the Image objects that are used by the tree nodes.
Nodes	Gets the collection of tree nodes that are assigned to the tree view control.
PathSeparator	Gets or sets the delimiter string that the tree node path uses.
SelectedNode	Gets or sets the tree node that is currently selected in the tree view control.
ShowLines	Gets or sets a value indicating whether lines are drawn between tree nodes in the tree view control.
ShowPlusMinus	Gets or sets a value indicating whether plus-sign (+) and minus-sign (-) buttons are displayed next to tree nodes that contain child tree nodes.
ShowRootLines	Gets or sets a value indicating whether lines are drawn between the tree nodes that are at the root of the tree view.
TopNode	Gets or sets the first fully-visible tree node in the tree view control.
VisibleCount	Gets the number of tree nodes that can be fully visible in the tree view control.

- **Methods of TreeView**

Name	Description
BeginUpdate	Disables redrawing of the tree view.
CollapseAll	Collapses all nodes.
EndUpdate	Enables redrawing of the tree view.
ExpandAll	Expands all the nodes.
GetNodeAt	Gets the node that is at the given location.
GetNodeCount	Gets the number of nodes.

- **Methods of Nodes Collection :**

Name	Description
Nodes.Add	Used to add a node in tree view. The new node is added at last.
Nodes.Insert	Used to add a node at specific position.
Nodes.Remove	Used to remove node from tree view
Nodes.RemoveAt	Used to remove specific node by specifying index no.
Nodes.Clear	Used to clear all the nodes of specific node.
Nodes.Contains	Used to check whether specified node exists or not.
Nodes.IndexOf	Used to know index no. of specific node. If specific node does not exist it returns -1.

- **Events of TreeView**

Name	Description
AfterCheck	Occurs when a node checkbox is checked.
AfterCollapse	Occurs when a tree node is collapsed.
AfterExpand	Occurs when a tree node is expanded.
AfterLabelEdit	Occurs when a tree node label text is edited.
AfterSelect	Occurs when a tree node is selected.
BeforeCheck	Occurs before a node checkbox is checked.
BeforeCollapse	Occurs before a node is collapsed.
BeforeExpand	Occurs before a node is expanded.
BeforeLabelEdit	Occurs before a node label text is edited.

BeforeSelect	Occurs before a node is selected.
ItemDrag	Occurs when an item is dragged into the tree view.

- The **Nodes** collection for a node holds the node's child **TreeNode** objects. You can add, remove, or clone a **TreeNode**, and when you do, all child tree nodes are added, removed, or cloned at the same time. Each **TreeNode** can contain a collection of other **TreeNode** objects, which means you can use expressions like this: **MyNode.Nodes(3).Nodes(5)** to refer to child nodes (in this case, actually grandchild nodes). You also can use the **FullPath** property to specify nodes in terms of their absolute, not relative, locations. And you can use the **Nodes** collection's **Add** or **Remove** methods to add or remove nodes in code.

➢ **PictureBox :**

- Picture boxes are used to display graphics from a bitmap, icon, JPEG, GIF or other image file type.
- To display an image in a picture box, you can set the **Image** property to the image you want to display, either at design time or at run time. You can clip and position an image with the **SizeMode** property, which you set to values from the **PictureBoxSizeMode** enumeration:
- **Normal—** Standard picture box behavior (the upper-left corner of the image is placed at upper left in the picture box).
- **StretchImage—** Allows you to stretch the image in code.
- **AutoSize—** Fits the picture box to the image.
- **CenterImage—** Centers the image in the picture box.
- You also can change the size of the image at run time with the **ClientSize** property, stretching an image as you want. By default, a **PictureBox** control is displayed without any borders, but you can add a standard or three-dimensional border using the **BorderStyle** property. And you can even handle events such as **Click** and **MouseDown** to convert an image into an image map.

- **Properties of PictureBox :**

Name	Description
BorderStyle	Gets/sets the border style for the picture box.
Image	Gets/sets the image that is in a picture box.

- **Methods of PictureBox :**
  (There are no specific methods for PictureBox apart from common methods)

- **Events of PictureBox :**

Name	Description
Resize	Occurs when the picture box is resized.
SizeModeChanged	Occurs when **SizeMode** changes.

➢ **Panel :**

- Panels are those controls which contain other controls, for example, a set of radio buttons, checkboxes, etc. Panels are similar to Groupboxes but the difference, Panels cannot display captions where as GroupBoxes can and Panels can have scrollbars where as GroupBoxes can't.
- If the Panel's Enabled property is set to False then the controls which the Panel contains are also disabled. Panels are based on the ScrollableControl class.
- Notable property of the Panel control in the appearance section is the BorderStyle property. The default value of the BorderStyle property is set to None. You can select from the predefined list to change a Panels BorderStyle.
- Notable property in the layout section is the AutoScroll property. Default value is set to False. Set it to True if you want a scrollbar with the Panel.

- **Difference between Panel and GroupBox :**

Panel	GroupBox
Panel can not be used to display Caption/Text.	GroupBox can be used to display Caption/Text.
Panels can have scroll bars.	GroupBox can not have scrollbar
Generally Panel is used if you don't want to show different groups under Border. Although they have BorderStyle property but bydefault BorderStyle is set to None.	Generally GroupBox is used to display group along with Border.

- **Properties of Panel :**

Name	Description
AutoScroll	Gets or sets a value indicating whether the container enables the user to scroll to any controls placed outside of its visible boundaries.
BorderStyle	Indicates the border style for the control.
Enabled	Gets or sets a value indicating whether the control can respond to user interaction.

- **Methods of Panel :**
  (Apart from general methods Panel control does not have any special methods)

- **Events of Panel :**
  (Apart from Scroll event all other events are common to Panel control)

Name	Description
Scroll	Occurs when the user or code scrolls through the Panel.

➢ **ScrollBar :**

- Windows Forms **ScrollBar** controls are used to provide easy navigation through a long list of items or a large amount of information by scrolling either horizontally or vertically within an application or control. Scroll bars are a common element of the Windows interface, so the **ScrollBar** control is often used with controls that do not derive from the ScrollableControl class. Similarly, many developers choose to incorporate the **ScrollBar** control when authoring their own user controls.
- The **HScrollBar** (horizontal) and **VScrollBar** (vertical) controls operate independently from other controls and have their own set of events, properties, and methods. **ScrollBar** controls are not the same as the built-in scroll bars that are attached to text boxes, list boxes, combo boxes, or MDI forms (the **TextBox** control has a ScrollBars property to show or hide scroll bars that are attached to the control).

- The **ScrollBar** controls use the Scroll event to monitor the movement of the scroll box (sometimes referred to as the thumb) along the scroll bar. Using the **Scroll** event provides access to the scroll bar value as it is being dragged.
- **Value Property :**
- The Value property (which, by default, is 0) is an **integer** value corresponding to the position of the scroll box in the scroll bar. When the scroll box position is at the minimum value, it moves to the left-most position (for horizontal scroll bars) or the top position (for vertical scroll bars). When the scroll box is at the maximum value, the scroll box moves to the right-most or bottom position. Similarly, a value halfway between the bottom and top of the range places the scroll box in the middle of the scroll bar.

- **LargeChange and SmallChange Properties :**

- When the user presses the PAGE UP or PAGE DOWN key or clicks in the scroll-bar track on either side of the scroll box, the **Value** property changes according to the value set in the LargeChange property. When the user presses one of the arrow keys or clicks one of the scroll-bar buttons, the **Value** property changes according to the value set in the SmallChange property.

- **Properties of HScrollBar / VScrollBar :**

Name	Description
Maximum	Gets or sets the upper limit of values of the scrollable range.
Minimum	Gets or sets the lower limit of values of the scrollable range.
LargeChange	Gets or sets a value to be added to or subtracted from the Value property when the scroll box is moved a large distance.
SmallChange	Gets or sets the value to be added to or subtracted from the Value property when the scroll box is moved a small distance.
Value	Gets or sets a numeric value that represents the current position of the scroll box on the scroll bar control.

- **Methods of HScrollBar / VScrollBar :**
  (There are no specific methods used for HscrollBar or VScrollBar instead of general methods)

- **Events of HScrollBar / VScrollBar :**

Name	Description
Scroll	Occurs when the scroll box has been moved by either a mouse or keyboard action.
ValueChanged	Occurs when the Value property is changed.

➢ **Tooltip :**

- A ToolStrip is a container for ToolStripItem elements. Each individual element on the ToolStrip is a ToolStripItem that manages the layout and event model for the type it contains. For example, elements visible on the toolbar, such as buttons, text boxes, labels, or combo boxes, or visible on the menu bar.
- ToolStrip is a container for ToolStripItem elements. Each individual element on the ToolStrip is a ToolStripItem that manages the layout and event model for the type it contains. . The ToolStrip controls provide a common interface for Menus and Strips in Windows Forms.
- Following is a list of ToolStripItem controls you can place within a ToolStrip.

Control Implementation	Description
ToolStripLabel	Used to display normal text, hyperlinks, and images.
ToolStripButton	Provides a typical pushbutton that you can configure to support both text and images.
ToolStripComboBox	A ComboBox with methods/properties to configure various styles.
ToolStripSeparator	A separator that you can use to visually separate groups of ToolStripItem elements.
ToolStripDropDownButton	This control provides a button which, when clicked, displays a ToolStripDropDown control. You'll implement

	a sample of this to get a better idea of how to use it.
ToolStripTextBox	A normal textbox which can be used to enter text.
ToolStripMenuItem	A special menu control built specifically for use with the MenuStrip and ContextMenuStrip controls.
ToolStripProgressBar	A specialized progress bar implementation for use within a ProgressBar control.
ToolStripSplitButton	A combination of normal button and a DropDownButton.
ToolStripControlHost	A control that acts as a host to your customized implementations or any other Windows Form controls.

- All these controls are usual controls which you use in developing Windows Forms. The only difference is they appear under ToolStrip. ToopStrip is just a container of all these controls. All the related properties, methods and events are used as we use in programming.
- Toolstrip has got one important property called "Items" which is collection of all the items (toolbar items) which are placed under.

➢ **DialogBox:**

- There are a number of built-in dialog boxes in Visual Basic, which is great, because developing your own file open, file save, and other dialog boxes not only takes a lot of work, but gives your program a different look from what Windows users are already used to. They are listed as follows

  - **Color Dialog Box (ColorDialog)**
  - **Font Dialog Box (FontDialog)**
  - **Open File Dialog Box (OpenFileDialog)**
  - **Save File Dialog Box (SaveFileDialog)**
  - **Print File Dialog Box (PrintDialog)**

- There are also other built-in dialog boxes like Print Preview Dialog Box, Page Setup Dialog Box, etc. which are used as per requirement.
- You use the **ShowDialog** method to display the dialog at run time and can check its return value (such as **DialogResult.OK** or **DialogResult.Cancel**) to see which

button the user has clicked. Here are the possible return values from this method, from the **DialogResult** enumeration:

Name	Description
Abort	The dialog box return value is **Abort** (usually from a button labeled Abort).
Cancel	The dialog box return value is **Cancel** (usually from a button labeled Cancel).
Ignore	The dialog box return value is **Ignore** (usually from a button labeled Ignore).
No	The dialog box return value is **No** (usually from a button labeled No).
None	Nothing is returned from the dialog box. This means that the modal dialog Continues running.
OK	The dialog box return value is **OK** (usually from a button labeled OK).
Retry	The dialog box return value is **Retry** (usually from a button labeled Retry).
Yes	The dialog box return value is **Yes** (usually from a button labeled Yes).

- Now all the dialog boxes are explained in detail.

➢ **Color Dialog Box (ColorDialog)**

- Color dialogs let the user select a color in an easy way. The principal property you use of these dialogs is the **Color** property, which returns a **Color** object, ready for use.
- If you set the **AllowFullOpen** property to **False**, on the other hand, the Define Custom Colors button is disabled and the user can select colors only from the predefined colors in the palette. Note also that if you set the **SolidColorOnly** property to **True**, the user can select only solid (not dithered) colors.

- **Properties of ColorDialog :**

Property	Description
AllowFullOpen	Gets/sets whether the user can use the dialog box to define custom colors.

AnyColor	Gets/sets whether the dialog box displays all available colors in the set of basic colors.
Color	Gets/sets the color selected by the user.
CustomColors	Gets/sets the set of custom colors shown in the dialog box.
FullOpen	Gets/sets whether the controls used to create custom colors are visible when the dialog box is opened
ShowHelp	Gets/sets whether a Help button appears in the color dialog box.
SolidColorOnly	Gets/sets whether the dialog box will restrict users to selecting solid colors only.

- **Methods of ColorDialog :**

Method	Means
Reset	Resets all dialog options to their default values.
ShowDialog	Shows the dialog.

- **Event of ColorDialog :**

Event	Means
HelpRequest	Occurs when the user clicks the Help button.

➢ **Open File Dialog Box(OpenFileDialog) and Save File Dialog Box (SaveFileDialog)**

- As you'd expect from its name, the Open File dialog and Save File dialog lets the user select a file to open or save. In fact, it's the same Open File dialog and Save File dialog used by Windows itself.
- There is no difference between OpenFileDialog and SaveFileDialog except OpenFileDialog is used to open a file and SaveFileDialog is used to save a file.
- Open File dialogs are supported with the **OpenFileDialog** class. And Save File dialogs are supported with the **SaveFileDialog** class. You can let users select multiple files with the **Multiselect** property. You can use the **ShowReadOnly** property to determine if a read-only checkbox appears in the dialog box. The **ReadOnlyChecked** property indicates whether the read-only checkbox is selected.
- The **Filter** property sets the current file name filter string, which determines the choices that appear in the "Files of type" box in the dialog box. The name and path the user selected is stored in the **FileName** property of the **OpenFileDialog**

or **SaveFileDialog** object—and there's a neat shortcut here: you can use the **OpenFile** method to open the selected file directly.

- **Properties of OpenFileDialog / SaveFileDialog :**

Property	Means
CheckFileExists	Gets/sets if the dialog box displays a warning if the user specifies a nonexistent file.
CheckPathExists	Gets/sets whether the dialog box displays a warning if the user gives a path that does not exist.
DefaultExt	Gets/sets the default file extension.
FileName	Gets/sets the file name selected in the file dialog box.
FileNames	Gets the file names of all selected files.
Filter	Gets/sets the current file name filter string, which sets the choices that appear in the "Save as file type" or "Files of type" box.
FilterIndex	Gets/sets the index of the filter selected in the file dialog box.
InitialDirectory	Gets/sets the initial directory used in the file dialog box.
Multiselect	Gets/sets whether the dialog box allows multiple file selections.
ShowHelp	Gets/sets whether the Help button should be displayed.
ShowReadOnly	Gets/sets whether the dialog displays a read-only check box.
Title	Gets/sets the file dialog box title.

- **Methods of OpenFileDialog / SaveFileDialog :**

Method	Means
OpenFile	Opens the file selected by the user, with read-only permission. The file is specified by the FileName property.
Reset	Resets all options to their default values.
ShowDialog	Shows the dialog box.

- **Events of OpenFileDialog / SaveFileDialog :**

Event	Means
FileOk	Occurs when the user clicks the Open or Save button.
HelpRequest	Occurs when the user clicks the Help button.

Font Dialog Box (FontDialog)

- Font dialogs let the user select a font size, face, color, and so on. To display the font dialog box, call the **ShowDialog** method. This dialog shows list boxes for **Font**, **Style**, and **Size**, checkboxes for effects like **Strikeout** and **Underline**, and a sample of how the font will appear. You can recover these settings using properties of the same names of the **Font** object returned by the **Font** property.
- The **FontDialog** class displays a dialog box that lets the user select a font. It returns a **Font** object in the **Font** property, and a **Color** object in the **Color** property. **Font** dialogs are supported by the **FontDialog** class.

- **Properties of FontDialog :**

Property	Description
Color	Gets/sets the selected font color.
FixedPitchOnly	Gets/sets whether the dialog box allows only the selection of fixed-pitch fonts.
Font	Gets/sets the selected font.
FontMustExist	Gets/sets whether the dialog box specifies an error condition if the user attempts to select a font or style that does not exist.
MaxSize	Gets/sets the maximum point size a user can select.
MinSize	Gets/sets the minimum point size a user can select.
ShowApply	Gets/sets whether the dialog box contains an Apply button.
ShowColor	Gets/sets whether the dialog box displays the color choice.
ShowEffects	Gets/sets whether the dialog box contains controls that allow the user to specify strikethrough, underline, and text color options.
ShowHelp	Gets/sets whether the dialog box displays a Help button.

- **Methods of FontDialog :**

Method	Description
Reset	Resets all dialog options to default values.
ShowDialog	Shows the dialog.

- **Events of FontDialog :**

Event	Description
Apply	Occurs when the user clicks the Apply button.
HelpRequest	Occurs when the user clicks the Help button.

Print Dialog Box (PrintDialog)

- Print dialogs let the user print documents, and these dialogs are supported with the **PrintDialog** class. Before displaying a Print dialog, you set the **Document** property of a **PrintDialog** object to a **PrintDocument** object, and the **PrinterSettings** property to a **PrinterSettings** object of the kind set by Page Setup dialogs.
- When the dialog is closed, you can print the document by assigning the **PrintDialog** object's **PrinterSettings** property (which returns a **PrinterSettings** object as configured by the user, indicating the number of copies to print, the printer to use, and so on) to the **PrinterSettings** property of the **PrintDocument** object and use the **PrintDocument** object's **Print** method to actually print the document.

- **Properties of PrintDialog :**

Property	Means
AllowPrintToFile	Gets/sets whether the Print to file checkbox is enabled.
AllowSelection	Gets/sets whether the Selection radio button is enabled.
AllowSomePages	Gets/sets whether the From... To... Page radio button is enabled.
Document	Gets/sets the **PrintDocument** used to obtain **PrinterSettings**.
PrinterSettings	Gets/sets the **PrinterSettings** dialog box to modify.
PrintToFile	Gets/sets whether the Print to file checkbox is checked.
ShowHelp	Gets/sets whether the Help button is displayed.
ShowNetwork	Gets/sets whether the Network button is displayed.
Method	Means
Reset	Resets all dialog options.
ShowDialog	Shows the dialog.

- **Methods of PrintDialog :**

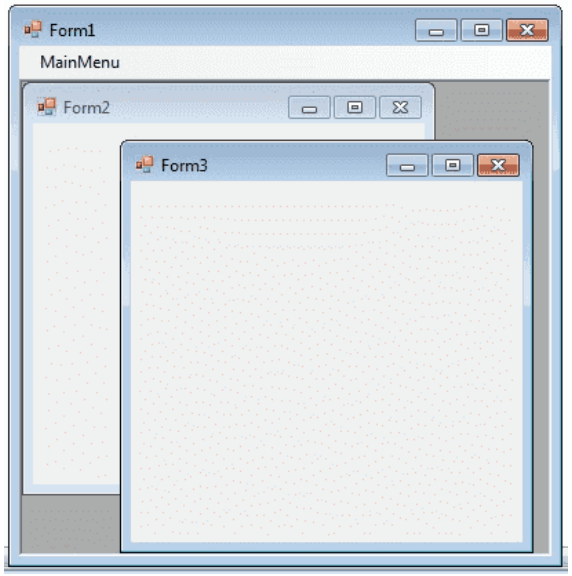
Method	Means
Reset	Resets all dialog options.
ShowDialog	Shows the dialog.

- **Event of PrintDialog :**

Event	Means
HelpRequest	Occurs when the user clicks the Help button.

➢ **MDI Concept:**

- A Multiple Document Interface (MDI) programs can display multiple child windows inside them.
- This is in contrast to single document interface (SDI) applications, which can manipulate only one document at a time.
- Visual Studio Environment is an example of Multiple Document Interface (MDI) and notepad is an example of an SDI application.
- MDI applications often have a Window menu item with submenus for switching between windows or documents.



Any windows can become an MDI parent, if you set the IsMdiContainer property to True.

```
IsMdiContainer = true;
```

The following C# program shows a MDI form with two child forms. Create a new C# project, then you will get a default form Form1 . Then add two mnore forms in the project (Form2 , Form 3) . Create a Menu on your form and call these two forms on menu click event. Click here to see how to create a Menu on your form How to Menu Control C#.

NOTE: If you want the MDI parent to auto-size the child form you can code like this.

```
form.MdiParent = this;
form.Dock=DockStyle.Fill;
form.Show();
```

- **MDI Notepad Concept:**

  - The MDI NotePad sample application is a simple text editor similar to the NotePad application included with Microsoft Windows.
  - The MDI NotePad application, however, uses a multiple-document interface (MDI). At run time, when the user requests a new document (implemented with the New command on the application's File menu), the application creates a new instance of the child form.
  - This allows the user to create as many child forms, or documents, as necessary.

To create a document-centered application in Visual Basic, you need at least two forms — an MDI form and a child form.

At design time, you create an MDI form to contain the application and a single child form to serve as a template for the application's document.

To create your own MDI NotePad application

1. From the **File** menu, choose **New Project**.
2. From the **Project** menu, choose **Add MDI Form** to create the container form.

   The project should now contain an MDI form (MDIForm1) and a standard form (Form1).
3. Create a text box (Text1) on Form1.
4. Set properties for the two forms and the text box as follows.

Object	Property	Setting
MDIForm1	Caption	MDI NotePad
Form1	Caption MDIChild	Untitled True

Text1	MultiLine Text Left Top	True (Empty) 0 0

5.
6. Using the **Menu Editor** (from the **Tools** menu), create a File menu for MDIForm1.

Caption	Name	Indented
&File	mnuFile	No
&New	mnuFileNew	Yes

7.
8. Add the following code to the mnuFileNew_Click procedure:
9. Private Sub mnuFileNew_Click ()
10. ' Create a new instance of Form1, called NewDoc.
11. Dim NewDoc As New Form1
12. ' Display the new form.
13. NewDoc.Show
14. End Sub

This procedure creates and then displays a new instance (or copy) of Form1, called NewDoc. Each time the user chooses New from the File menu, an exact duplicate (instance) of Form1 is created, including all the controls and code that it contains.

15. Add the following code to the Form_Resize procedure for Form1:

```
Private Sub Form_Resize ()
  ' Expand text box to fill the current child form.
  Text1.Height = ScaleHeight
   Text1.Width = ScaleWidth
End Sub
```

Press F5 to run the application.

- **Timer :**
  - Timers are very useful controls, because they let you create periodic events. Strictly speaking, timers are no longer controls but components, and they do not appear in a window at run time. At design time, they appear in the component tray under the form you've added them.
  - Windows timers are designed for a single-threaded (as opposed to multithreaded) environment; you set how often you want the timer to generate **Tick** events by setting the **Interval** property (in milliseconds, one thousandths of a second). Each time a **Tick** event happens, you can execute code in a handler for this event, just as you would for any other event.
  - **Properties of Timer :**

Name	Description
Enabled	Gets/sets whether the timer is running.
Interval	Gets/sets the time (in milliseconds) between timer ticks.

  - **Methods of Timer :**

Name	Description
Start	Starts the timer whenever called.
Stop	Stops the timer whenever called.

  - **Event of Timer**

Name	Description
Tick	Occurs when the timer interval has elapsed (and the timer is

	enabled).

- **Handling Mouse Events :**
  - Mice can do a lot more than nibble cheese. In a Windows UI, they can let you know when one of their buttons has been clicked or released, or when the mouse pointer leaves or enters some part of the application.
  - This information is provided in the form of **MouseDown**, **MouseUp**, **MouseMove**, **MouseEnter**, **MouseLeave**, and **MouseHover** events.
  - When wiring up mouse events, you can also change the mouse cursor. You typically marry the ability to change the cursor to the **MouseEnter** and **MouseLeave** events. These are used to provide feedback to the user that something is happening, not happening, or that certain areas are offlimits or welcome to the explorative nature of your cursor.
  - Mouse events occur in the following order:
    1. MouseEnter **2. MouseMove**
    3. MouseHover/MouseDown/MouseWheel **4. MouseUp**
    5. MouseLeave

Property	Purpose
Button	Tells you which mouse button was pressed
Clicks	Tells you how many times the mouse button was pressed and released
Delta	○ Retrieves a signed count Keyboard events are fired when a key is pressed in a control that has the focus. ○ The key event handler receives an argument of type **KeyPressEventArgs** containing data related to this keypress event. Table lists the **KeyPressEventArgs** properties that provide information related to the event received. ○ ***KeyPressEvenArgs* Properties :** of the number of *detents* the mouse wheel has rotated. A

	detent is one notch of the mouse wheel.
X	Retrieves the X coordinate of a mouse click
Y	Retrieves the Y coordinate of a mouse click

- **Handling Key Events :**
  - **KeyPress** events also bubble up from the OS and are made available to you in **KeyPress**, **KeyDown**, and **KeyUp** events. Mouse event handlers receive an argument of type **EventArgs** containing data related to their events; however, key−press event handlers receive an argument of type **KeyEventArgs** (a derivative of **EventArgs**) containing data specific to the keypress or key release events.

Property	Purpose
Handled	Retrieves or returns a value to the property indicating whether the **KeyPress** event was handled
KeyChar	Retrieves the **KeyChar** (character pressed) value corresponding to the key pressed Key events occur in the following order: **1.KeyDown 2.KeyPress 3. KeyUp**

Link Label:

- A Link Label control is a label control that can display a hyperlink. A Link Label control is inherited from the Label Class so it has all the functionality provided by the windows form label control.
- Link Label control does not participate in user input or capture mouse or keyboard events.

Chapter-4 :- Database Programming With ADO.NET

- **ADO .NET :**

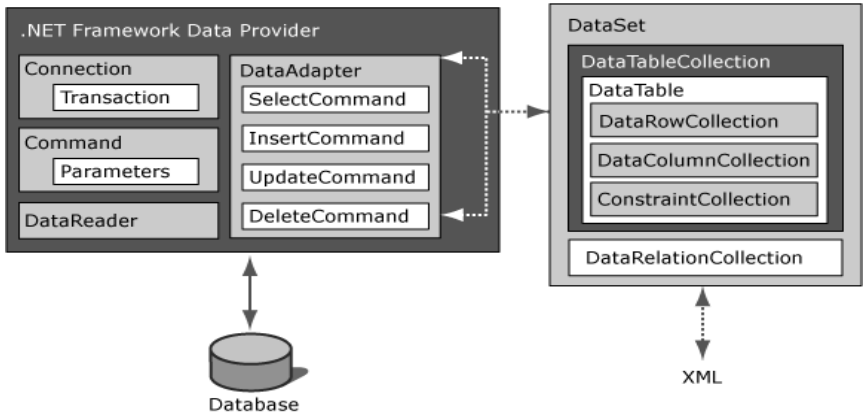
ADO.NET provides excellent support for reading data from a DBMS, including a set of classes for fast, efficient access to data in SQL Server databases and another set of classes to support OLE DB data sources. Of course you can also use other database under ADO.NET.

ADO.NET is a large set of .NET classes that enable us to retrieve and manipulate data, and update data sources, in very many different ways.

ADO.NET has several advantages. They are as follows :

- You can use ADO.NET to connect with any type of database.
- ADO.NET supports Connected as well as Disconnected Architecture. Disconnected Architecture is a new invention since ADO.NET is introduced.
- XML is supported by ADO.NET which is widely used for data transformation widely today. This makes your data transformation much more faster.
- You can create Fast, Scalable and Secured applications using ADO.NET
- Cross Language support is provided by ADO.NET as Multilanguage is a prime feature of .NET framework.

- **Architecture of ADO .NET :**



- There are two types of Architecture under ADO.NET.

  - Connected Architecture
  - Disconnected Architecture

1) Connected Architecture :

- Connected Architecture simply means, you are connected with the database throughout your operations.
- Although connected architecture is faster then disconnected architecture, It has one main problem of creating more traffic at database end.
- Because you are constantly connected to database. And so do all other users.

- In Connected Architecture, you are blocking a memory portion of database which makes performance slower at some extent. Client's memory is not utilized in this.
- DataReader is Connected Architecture since it keeps the connection open untill all rows are fetched one by one.
- Following set of classes are used while using Connected Architecture method :
  - **Connection :** This allows you to connect with database.
  - **Command :** This allows you to give commands like Insert, Update, Delete, Select, etc.
  - **DataReader :** This allows you to read data directly from the database..

2) DisConnected Architecture

- Disconnected Architecture is the new concept / feature introduced in ADO.NET. The "Disconnected" name itself is the answer.
- Disconnected architecture is a method of retrieving a record set from the database and storing it giving you the ability to do many CRUD (Create, Read, Update, Delete) operations on the data in memory, then it can be re-synchronized with the database when reconnecting. A method of using disconnected artitecture is using a DataSet.
- DataSet is DisConnected Architecture since all the records are brought at once and there is no need to keep the connection alive.
- Following set of classes are used while using Disconnected Architecture method :

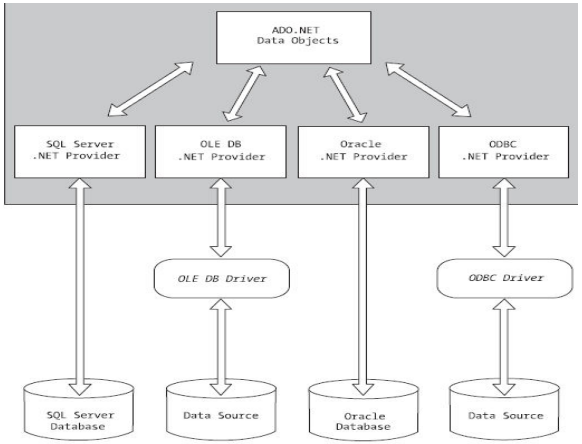
  - **Connection :** This allows you to connect with database.
  - **DataAdapter :** This plays an important role in Disconnected Architecture. DataAdapter is a mediator which provides (fills) data into DataSet / DataTable and again updates the data back to database.
  - **DataSet :** This is used if you want to load some group of tables into local memory. You can also use DataSet to load single table data. DataSet is collection of tables.
  - **DataTable :** This is used if you want to load only one table into local memory.
  - **DataRow :** This is used incase you want to add one new row into DataSet or DataTable. You can also use DataRow to remove rows from DataSet or DataTable.
  - **DataColumn :** This is used incase you want to modify any column or you want to add a new column to DataSet or DataTable.

➢ **Difference between Connected and Disconnected Architecture**

Connected Architecture	Disconnected Architecture
In this, you are connected with database throughout your operation. Data is stored in database while doing operation.	In this, you get the copy of data and then do operations in local memory. Data is stored in local memory while doing operation.
This creates Heavy Traffic at database server side when no. of user increases.	As data is loaded into local memory, does not create Heavy Traffic at database server side even no. of user increases.
This utilizes memory of Database server as the data remains in database itself.	This utilizes memory of client as the data is loaded locally into client's machine.
Generally, transactional commands like Insert, Update, Delete, etc. are performed using Connected Architecture, as they do not require more time.	Generally, if you want to load data just for display purpose. Disconnected Architecture is best choice as you just want a copy of data into local memory which makes faster data access.
Command and DataReader objects are used for connected architecture.	DataAdapter, DataSet, DataTable, DataRow and DataColumn objects are used in this architecture.

DataReader is used to fetch the data in Connected Architecture.	DataSet or DataTable is used to fetch the data in Disconnected Architecture.

- **Data Providers in ADO .NET :**
  - Data Providers are different set of classes which provide functionality to access to different data sources like Access, Oracle, SQL Server, DB2, etc… databases.
  - There are four types of Data Providers are provided by .Net.



- **SQL Server Provider :** This is special data provider which gives access to only SQL Server database. It supports connection to SQL Server 7.0 or later. The benefit of using this provider is it is faster as it directly deals with SQL Server.
- **Oracle Provider :** This is special data provider which gives access to only Oracle Server or client. It supports connection to Oracle 8i or later. The benefit of using this provider is it is faster as it directly deals with Oracle database.
- **OLEDB Provider :** This provider gives access to any database that has OLEDB driver. The benefit of using OLEDB provider is you can connect with any database using this.
- **ODBC Provider :** This provider gives access to any database that has ODBC (Open DataBase Connectivity) driver.

- **Namespaces used in ADO.NET**

Use of Namespaces depends on which provider you choose. Some of the classes are obtained from some common namespaces. But in order to use specific provider you need to import specific namespace. Following table shows list of some namespaces used in ADO.NET

Namespace	Description
System.Data	Contains fundamental classes with the core ADO.NET functionality. This includes DataSet and DataRelation, which allow you to manipulate structured relational data. These classes are totally independent of any specific type of database or the way you use to connect to it.
System.Data.SqlClient	Contains the classes you use to connect to a Microsoft SQL Server database (version 7.0 or later). These classes, such as SqlCommand and SqlConnection, provide all the same roperties and methods as their counterparts in the System.Data.OleDb namespace. The only difference is that they are optimized for SQL Server and provide better performance by eliminating the extra OLE DB layer (and by connecting directly to the optimized TDS interface).
System.Data.OracleClient	Contains the classes you use to connect to an Oracle database, such as OracleCommand and OracleConnection.
System.Data.OLEDB	Contains the classes you use to connect to an OLE DB data source, including OleDbCommand and OleDbConnection.
System.Data.ODBC	Contains the classes you use to connect to a data source through an ODBC driver. These classes include OdbcCommand and OdbcConnection.

Depending on the namespace which you have imported, you can use provider specific classes in order to use ADO.NET. Consider following table for provider specific classes.

	SQL Server .NET Provider	OLE DB .NET Provider	Oracle .NET Provider	ODBC .NET Provider
Connection	SqlConnection	OleDbConnection	OracleConnection	OdbcConnection
Command	SqlCommand	OleDbCommand	OracleCommand	OdbcCommand
DataReader	SqlDataReader	OleDbDataReader	OracleDataReader	OdbcDataReader

- **Connection :**
  - The first thing we need to do with a database application is establish a connection to the database. ADO.NET handles this by using connection classes. In this article, Michael Youssef shows you how to start using connection classes, with examples.
  - When developing database applications using .NET, the very first thing that we need is a connection to the database. ADO.NET provides us with connection classes like the SqlConnection class and OleDbConnection class. The SqlConnection class is part of the SQL Server .NET Data Provider. This data provider has been designed for performance optimization with SQL Server 7.0 and later.
  - The OleDbConnection is part of the OLEDB .NET Data Provider, which is used to access a data source that has an OLEDB Provider. In this article (and actually most of my articles on ADO.NET) I use the SQL Server .NET Data Provider. So let's talk a little about the SqlConnection class before we write code.

  - **Properties of Connection Object**

Property	Description
CommandTimeout	Sets or returns the number of seconds to wait while attempting to execute a command
ConnectionString	Sets or returns the details used to create a connection to a data source
ConnectionTimeout	Sets or returns the number of seconds to wait for a connection to open
DefaultDatabase	Sets or returns the default database name
Mode	Sets or returns the provider access permission
Provider	Sets or returns the provider name
State	Returns a value describing if the connection is open or closed
Version	Returns the ADO version number

  - **Methods of Connection Object**

Method	Description
BeginTrans	Begins a new transaction
Close	Closes a connection
CommitTrans	Saves any changes and ends the current transaction
Open	Opens a connection
RollbackTrans	Cancels any changes in the current transaction and ends the transaction

  - **Events of Connection Object**

Event	Description
BeginTransComplete	Triggered after the BeginTrans operation
CommitTransComplete	Triggered after the CommitTrans operation
ConnectComplete	Triggered after a connection starts
Disconnect	Triggered after a connection ends
ExecuteComplete	Triggered after a command has finished executing
RollbackTransComplete	Triggered after the RollbackTrans operation
WillConnect	Triggered before a connection starts
WillExecute	Triggered before a command is executed

- **Command :**

- The SqlCommand class is at the heart of the Data Provider's namespace. It is used to execute operations on a database and retrieve data.
- This is specially used under Connected Architecture. Command allows you to specify different types of SQL Commands like Insert, Update, Delete, Select, etc. It has several methods which help you to manipulate or fetch the data from database.
- While using Command object, it is necessary to open the connection using Open() method before attempting any operation. It will not open the connection automatically like Disconnected Architecture. The same way after operation is done, you need to close the connection using Close() method.
- **Properties of Command Object**

Property	Description
CommandText	Contains the text of a SQL query
CommandTimeout	Contains the length of the timeout of a query, in seconds
CommandType	Specifies the type of command to be executed
Connection	Specifies the connection to the database
Parameters	Specifies a collection of parameters for the SQL query
Transaction	Specifies a transaction object, which enables developers to run queries in a transaction

- **Methods of Command Object**

Method	Description
Cancel()	Cancels the running query
CreateParameter()	Returns a new SQL parameter
ExecuteNonQuery()	Executes the CommandText property against the database and does not return a result set. This method is used incase you have given Transactional Commands like INSERT, UPDATE, DELETE, etc. This method returns integer value as answer. If value is > 0, this means command executed successfully. 0 means it did not have any effect.
ExecuteReader()	Executes the CommandText property and returns data in a DataReader object . This method is used if you have specified SELECT. This allows command object to connect with DataReader which helps you to read the data.
ExecuteScalar()	Executes the CommandText property and returns a single value. This method is also used when you have specified SELECT command. But it returns only one value. It returns value of First Row's First Column. For example your command was "Select * from student", even though entire table, it will give you value of first student's first column.

- **Event of Command Object**

Event	Description
StatementCompleted	This event occurs when the Transactional-SQL command is finished at database end.

- **DataReader :**
  - The DataReader object defines a lightweight yet powerful object that is used to read information from a database.
  - This is used under Connected Architecture. DataReader is used with Command object in order to read (fetch) the data. But remember that DataReader can be used to only read the data in sequential manner (forward only manner). You can not go back or go on particular record directly.
  - **Properties of DataReader Object**

Property	Description
FieldCount	Contains the number of fields retrieved from the query
IsClosed	Contains True if the DataReader object is closed
Item	Contains A collection of values that are accessible both by field name and by ordinal number
RecordsAffected	Returns the number of records affected by an executed query
HasRows	Gets a value that indicates whether the DataReader contains one or more rows.

Methods of DataReader Object

Method	Description
Close()	Closes the DataReader object
IsDBNull()	Returns True if the field contains Null
Read()	Reads the next result in the result set into memory . The Read method of the DataReader object is used to obtain a row from the results of the query. Each column of the returned row may be accessed by passing the name or ordinal reference of the column to the DataReader,

- **DataAdapter :**
  - DataAdapter is used under Disconnected Architecture. It plays an important role in disconnected architecture. It allows you to fetch the data from database to local memory of client and also to save the updated data back to database.
  - In Disconnected Architecture we use DataSet or DataTable to fill the data which resides in local memory. DataAdapter object does two important tasks as follows
  - Fill data from Database to DataSet or DataTable
  - Update the data back to Database
  - DataAdapter fills the specified data into DataSet or DataTable which is in our (client's) local memory. After filling the data connection is automatically terminated (closed) with DataAdapter. But now the data has been loaded into DataSet or DataTable. All types of operations like INSERT, UPDATE, DELETE, SELECT, etc. are performed locally in DataSet or DataTable.
  - **Properties of DataAdapter Object**

Property	Description
SelectCommand	This property works in background which generates your Select command automatically. OleDbCommandBuilder class works behind this to generate Select command automatically.
DeleteCommand	This property also works in background which generates your

	Delete command. If you have deleted some rows from DataSet or DataTable, OleDbCommandBuilder automatically generates Delete Command and stores in this.
InsertCommand	This property also works in background which generates Insert Command. If you have insert any new row in DataSet or DataTable, OleDbCommandBuilder automatically generates Insert command and stores in this.
UpdateCommand	This property also works in background which generates Update Command. If you have done some changes in DataSet or DataTable data, OleDbCommandBuilder automatically generates Update command and stores in this.

- **Methods of DataAdapter Object**

Method	**Description**
Fill	Most important method of DataAdapter. This method helps you to fill the data into DataSet or DataTable. Remember that in DataAdapter we can give only SELECT command which is filled into DataSet or DataTable. The Fill method is probably the DataAdapter method you will use most frequently. Simply stated, the Fill method adds data from your data source to a dataset. The Fill method accepts a variety of parameters including the DataSet object to fill, a string representing the alias for the newly created DataSet object, an integer representing the lower bound of records to retrieve, and an integer representing the upper bound of records to retrieve from our data source.
Update	Most important method of DataAdapter. This method is used to save

	the changes back from DataSet / DataTable to actual database table. Before using Update() method you need to use SqlCommandBuilder class so that it can generate related INSERT, UPDATE, DELETE, ALTER, etc. command automatically The Update method calls the respective insert, update, or delete command for each inserted, updated, or deleted row in the DataSet. There are three different ways to call the Update method—you can pass: An array of DataRow objects A DataSet object A DataSet object and a string representing a table name

- **Events of DataAdapter Object**

Event	**Description**
FillError	This event is raised when there is a problem in filling data into DataSet or DataTable.
RowUpdating	This event is raised while changes are being made in Row. When you call Update() method, DataAdapter makes changes to actual database table. At that time, while being updating each row this event is raised.
RowUpdated	This event is raised after changes are done in Row. When you call Update() method, DataAdapter makes changes to actual database table. After changes have been done, this event is raised.

- **DataSet :**
  - The DataSet object represents an in-memory group of database tables. You can have more then one tables under DataSet. It is collection of DataTables.
  - The DataSet is the main in disconnected, data-driven application; it is an in-memory representation of a complete set of data, including tables, relationships, and constraints. The DataSet does not maintain a connection to a data source, enabling true disconnected data management.
  - The data in a DataSet can be accessed, manipulated, updated, or deleted, and then reconciled with the original data source. Since the DataSet is disconnected from the data source, there is less contention for valuable resources, such as database connections, and less record locking.
  - **Properties of DataSet Object**

Property	Description
DataSetName	Gets or sets the name of the current DataSet.
HasErrors	Gets a value indicating whether there are errors in any of the DataTable objects within this DataSet.
IsInitialized	Gets a value that indicates whether the DataSet is initialized.
Relations	Get the collection of relations that link tables and allow navigation from parent tables to child tables.
Tables	Gets the collection of tables contained in the DataSet.

- **Methods of DataSet Object**

Method	Description
AcceptChanges()	Commits all the changes made to this dataset since the last time AcceptChanges was called.

Method	Description
Clear()	Clears all the DataTables of DataSet
ReadXml()	Reads XML schema and data into DataSet using specified Stream or File
ReadXmlSchema()	Reads an XML schema into the DataSet using the specified stream. or File
RejectChanges	Rolls back all changes that have been made to the dataset since it was loaded, or the last time AcceptChanges was called.
Select	Allows you to select group of data from dataset. It has different methods to select the data. This is overloaded method which allows you to filter the data in variety of ways.
WriteXml()	Writes the current contents of the DataSet as XML using the specified Stream or File.
WriteXmlSchema()	Writes the current data structure of the DataSet as an XML schema to the specified stream or file.

Events of DataSet Object

Method	Description
Initialized	Occurs after the DataSet is initialized.

- **DataTable :**
  - The DataTable object represents an in-memory database table. You can add rows to a DataTable with a DataAdapter.
  - A DataTable is defined in the “System.Data” namespace and represents a single table of memory-resident data. It contains a collection of columns represented by

the DataColumnCollection, which defines the schema and rows of the table. It also contains a collection of rows represented by the DataRowCollection, which contains the data in the table. Along with the current state, a DataRow retains its original state and tracks changes that occur to the data.

- The DataTable class is a central class in the ADO.NET architecture; it can be used independently, and in DataSet objects. A DataTable consists of a Columns collection, a Rows collection, and a Constraints collection. The Columns collection combined with the Constraints collection defines the DataTable schema, while the Rows collection contains the data.

- **Properties of DataTable Object**

Property	Description
Columns	The Columns collection is an instance of the DataColumnCollection class, and is a container object for zero or more DataColumn objects. The DataColumn objects define the DataTable column, including the column name, the data type, and any primary key or incremental numbering information.
Constraints	The Constraints collection is an instance of the ConstraintCollection class, and is a container for zero or more ForeignKeyConstraint objects and/or UniqueConstraint objects. The ForeignKeyConstraint object defines the action to be taken on a column in a primary key/foreign key relationship when a row is updated or deleted. The UniqueConstraint is used to force all values in a column to be unique.
DataSet	Gets the DataSet to which this table belongs.
HasErrors	Gets a value indicating whether there are errors in any of the rows in any of the tables of the DataSet to which the table belongs.

IsInitialized	Gets a value that indicates whether the DataTable is initialized.
PrimaryKey	Gets or sets an array of columns that function as primary keys for the data table.
Rows	The Rows collection is an instance of the DataRowCollection class, and is a container for zero or more DataRow objects. The DataRow object contains the data in the DataTable, as defined by the DataTable.Columns collection. Each DataRow has one item per DataColumn in the Columns collection.
TableName	Gets or sets the name of the DataTable.

- **Methods of DataTable Object**

Method	Description
AcceptChanges()	Commits all the changes made to this table since the last time AcceptChanges was called.
Clear()	Clears the DataTable of all data.
NewRow()	Creates a new DataRow with the same schema as the table.
ReadXml()	Reads XML schema and data into the DataTable using the specified Stream or File
ReadXmlSchema()	Reads an XML schema into the DataTable using the specified stream. or File
RejectChanges	Rolls back all changes that have been made to the table since it was loaded, or the last time AcceptChanges was called.
Select	Allows you to select group of data from table. It has different methods to select the data. This is overloaded method which

Method	Description
	allows you to filter the data in variety of ways.
WriteXml()	Writes the current contents of the DataTable as XML using the specified Stream or File.
WriteXmlSchema()	Writes the current data structure of the DataTable as an XML schema to the specified stream or file.

- **Events of DataTable Object**

Event	Description
Initialized	Occurs after the DataTable is initialized.
RowChanged	Occurs after a DataRow has been changed successfully.
RowChanging	Occurs when a DataRow is changing.
RowDeleted	Occurs after a row in the table has been deleted.
RowDeleting	Occurs before a row in the table is about to be deleted.
TableCleared	Occurs after the Table is cleared.
TableClearing	Occurs when then Table is being cleared.
TableNewRow	Occurs when you insert a new Row in table.

- **DataRow :**
  - DataRow can contain any single row of any table. Whenever you store any DataTables single row into object of DataRow, it automatically creates its column as per source table.
  - For example, your DataTable object has 6 columns, when you copy one row of DataTable into DataRow object, DataRow object automatically creates 6 columns for itself as per DataTable object.
  - To populate a DataTable we add new DataRow objects to the DataTable.Rows collection. Each DataRow can reference each DataColumn in the DataTable schema. To create a new row in the DataTable, we invoke the DataTable.NewRow method, which returns a DataRow using the DataTable's current schema.
  - **DataRow** objects represent rows in a **DataTable** object. You use **DataRow** objects to get access to, insert, delete, and update the records in a table.
  - To create a new **DataRow** object, you usually use the **NewRow** method of a **DataTable** object, and after configuring the row with data, you can use the **Add** method to add the new **DataRow** to the table. In addition, you also can call the **AcceptChanges** method of the **DataTable** object to make that table treat the new row as it would its original data.
  - You can delete a **DataRow** from the **Rows** collection in a data table by calling the **Remove** method, or by calling the **Delete** method of the **DataRow** object itself. Note that the **Remove** removes the row from the collection, and the **Delete** method simply marks the **DataRow** for deletion. (The actual deletion occurs when you use the **AcceptChanges** method.)

  - **Properties of DataRow Object**

Property	Description
HasErrors	Indicates if there are errors in the row.
Item	Gets/sets data in a specified column.
RowError	Gets/sets a row's error description.

RowState	Gets the current state of a row.
Table	Gets the table that contains this row.

- **Methods of DataRow Object**

Method	Description
AcceptChanges	Accepts (commits) the changes made to the row.
RejectChanges	Rolls back the changes made to the table since it was created or since the AcceptChanges method was called.
BeginEdit	Begins an edit operation.
EndEdit	Ends the current edit operation.
CancelEdit	Cancels the current edit operation.
ClearErrors	Clears the errors in the row.
Delete	Deletes the row.
IsNull	Indicates if a column contains a null value.

- **DataColumn :**
  - **DataColumn** objects represent the columns, that is, the fields, in a data table. In ADO.NET terms, the columns in a table specify its XML schema. When you create a table and add columns to it, you specify the name of the column and the type of data it stores.
  - DataRow is collection of DataColumns. One DataRow can have multiple columns within it because always a row contains multiple columns.

- **Properties of DataColumn Object**

Property	Description
AllowDBNull	Gets/sets if null values are allowed.
AutoIncrement	Gets/sets if the column automatically increments the column's value when new rows are added to the table.
AutoIncrementSeed	Gets/sets the starting value for an autoincrement column.
AutoIncrementStep	Gets/sets the increment for an autoincrement column.
Caption	Gets/sets the caption for the column.
ColumnName	Gets/sets the name of the column.
DataType	Gets/sets the type of data in the column.
DefaultValue	Gets/sets the default value for the column (used in new rows).
MaxLength	Gets/sets the maximum length of a text column.
ReadOnly	Gets/sets if the column is read-only.
Table	Gets the table the column belongs to.
Unique	Gets/sets if the values in this column must be unique.

- **Methods of DataColumn Object**

Method	Description

ToString	Gets the Expression value for this column, if there is one.
GetType	Gets the type of object.

- **DataView :**
  - When you bind to a DataTable, you actually use another object called the DataView. The DataView sits between the ASP.NET web page binding and your DataTable. Usually it does little aside from providing the information from the associated DataTable. However, you can customize the DataView so it applies its own sort order. That way, you can customize the data that appears in the web page, without needing to actually modify your data.
  - You can create a new DataView object by hand and bind the DataView directly to a data control.
  - Data views are much like read-only mini-datasets; you typically load only a subset of a dataset into a data view.
  - Data views represent a customized view of a single table that can be filtered, searched, or sorted. In other words, a data view, supported by the DataView class, is a data "**snapshot**" that takes up few resources.
  - To create a filtered and sorted view of data, set the **RowFilter** and **Sort** properties. Then use the **Item** property to return a single **DataRowView**.
  - **Properties of DataView Object**

Property	Description
AllowDelete	Sets or gets a value that indicates whether deletes are allowed.
AllowEdit	Gets or sets a value that indicates whether edits are allowed.
AllowNew	Gets or sets a value that indicates whether the new rows can be added by using the AddNew method.
Count	Gets the number of records in the DataView after RowFilter and RowStateFilter have been applied.

IsOpen	Gets a value that indicates whether the data source is currently open and projecting views of data on the DataTable.
Item	Gets a row of data from a specified table.
RowFilter	Gets or sets the expression used to filter which rows are viewed in the DataView.
Sort	Gets or sets the sort column or columns, and sort order for the DataView.
Table	Gets or sets the source DataTable.

  - **Methods of DataView Object :**

Method	Description
AddNew	Adds a new row to the DataView.
Close	Closes the DataView.
Delete	Deletes a row at the specified index.
Open	Opens a DataView.

  - **Events of DataView Object**

Events	Description
ListChanged	Occurs when the list managed by the DataView changes.

- **GridView :**
  - The GridView is an extremely flexible grid control that displays a multicolumn table. Each

record in your data source becomes a separate row in the grid. Each field in the record

becomes a separate column in the grid.

- GridView control is a grid control which displays data in a tabular format.
- That means you can bind table with GridView which can be displayed under GridView.
- Each record of table becomes row of GridView and each column of table becomes column of GridView control.
- GridView has inbuilt features that allows to Edit, Delete or Insert data directly into GridView.
- The GridView control looks like follows :

Column 0	Column 1	Column 2
Abc	Abc	abc
Abc	Abc	abc
Abc	Abc	abc
Abc	Abc	abc

- Following is the list of properties supported by GridView control :

Property	Description
AllowPaging	Used to set paging if table data is too long. By default “false”.
AllowSorting	Used to sort the data in GridView. By default “false”.
PageSize	Used to set the size of page if AllowPaging is “true”. By default size is 10 rows per page.
AutoGeneragteDeleteButton	Displays Delete link along with each rows of GridView if set to “true”.

AutoGenerateEditButton	Displays Edit link along with each rows of GridView if set to “true”.
AutoGenerateSelectButton	Displays Select link along with each rows of GridView if set to “true”.
Caption	Used to set heading for GridView.
DataSource	Runtime property used to display data from table.

- GridView supports following Methods :

Method	Description
DataBind( ) :	Used to bind data with specified data source.
DeleteRow( ) :	Deletes specified row from GridView.
UpdateRow( ) :	Updates specified row from GridView.
Sort( ) :	Sorts GridView data depending on the two parameters passed : 1. Sort Expression 2. Sort Direction

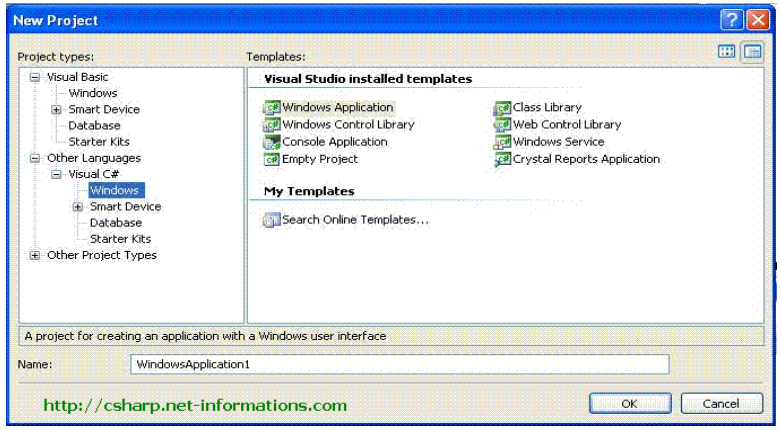
- GridView has define following Events :

Event	Description
PageIndexChanged	Occurs if AllowPaging property is set to “true” and page index is chaged.
SelectedIndexChaged	Raised after index of selected link button is chaged in GridView control.

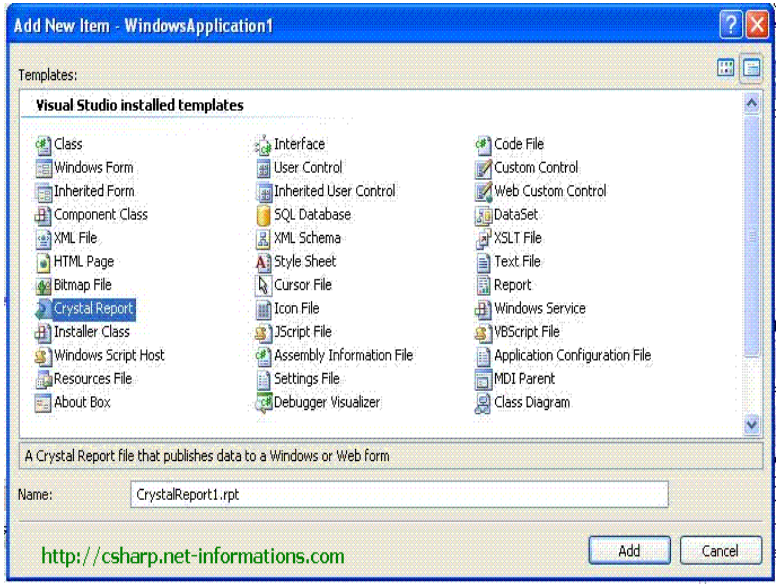
Chapter-5:- User Controls(Components),Crystal Report, Set up Projects

- **Creating Crystal Reports:-**

  - **Crystal Report** is a Reporting Application that can generate reports from various Data Sources like Databases , XML files etc.
  - The Visual Studio.NET Integrated Development Environment comes with Crystal Reports tools.
  - The Crystal Reports makes it easy to create simple reports, and also has comprehensive tools that you need to produce complex or specialized reports in csharp and other programming languages.
  - Crystal Reports is compatible with most popular development environments like **C# , VB.NET** etc.
  - You can use the Crystal Reports Designer in Visual Studio .NET to create a new report or modify an existing report.
  - Open Visual Studio .NET and select a new CSharp Windows project.



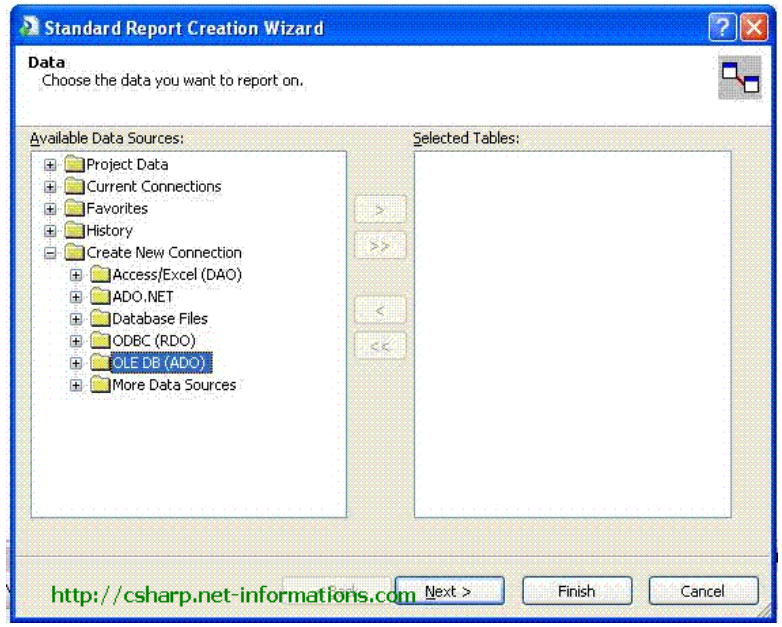
- Now you will get the default Form1.cs.
- From the main menu in Visual Studio C# project select PROJECT-->Add New Item . Then Add New Item dialogue will appear and select Crystal Reports from the dialogue box.



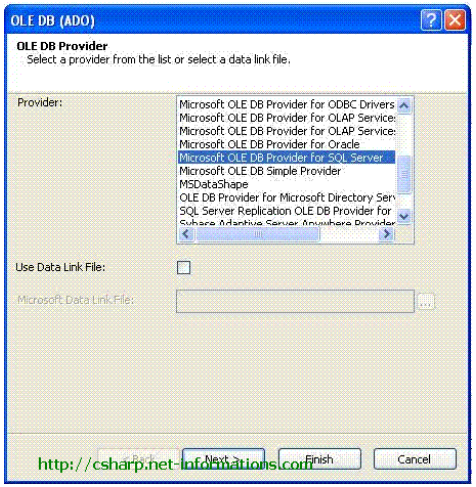
- Select Report type from Crystal Reports gallery.

- Accept the default settings and click OK.
- Next step is to select the appropriate connection to your database (here crstaldb). Here we are going to select **OLEDB Connection** for SQL Server to connect Crystal Reports in C#.

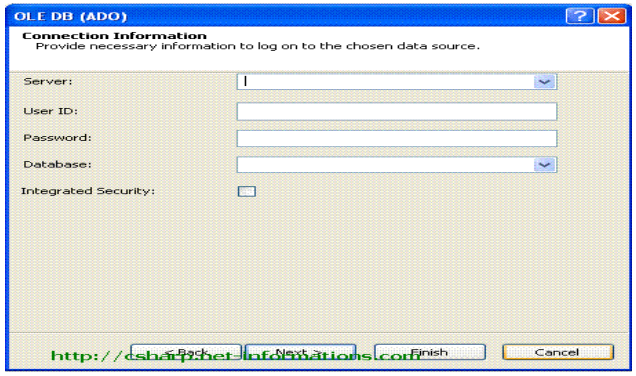
- Select OLE DB (ADO) from Create New Connection .



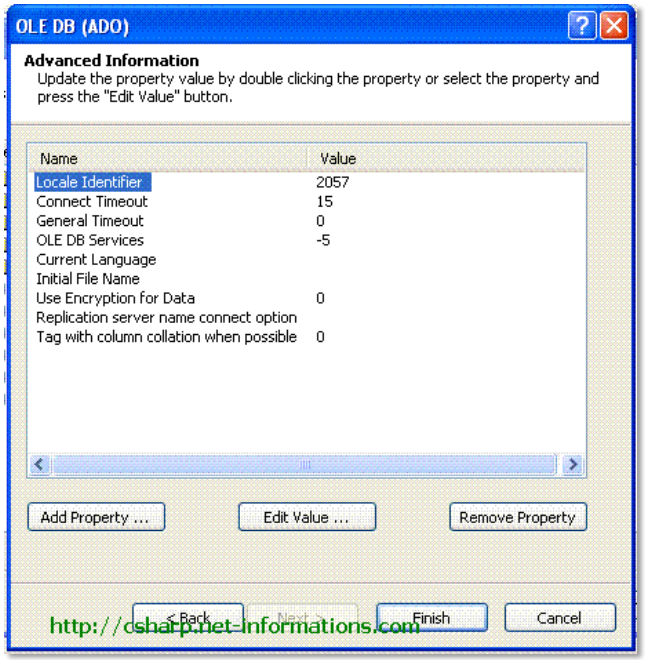
- Select Microsoft OLE DB Provider for SQL Server.



- The next screen is the SQL Server authentication screen for connecting to the database - crystalDB. Select your Sql Server name , enter userid , password and select your Database Name .

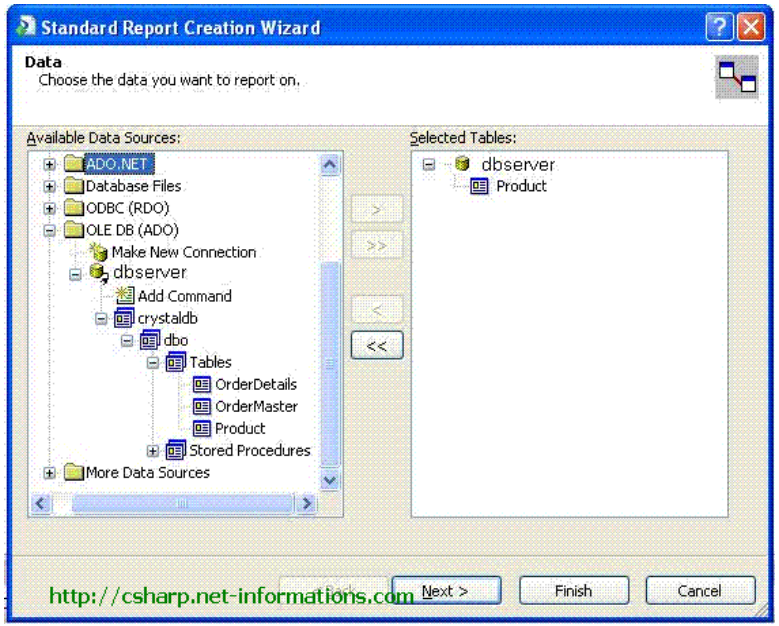


- Click next , Then the screen shows OLE DB Property values , leave it as it is , and then click finish button.

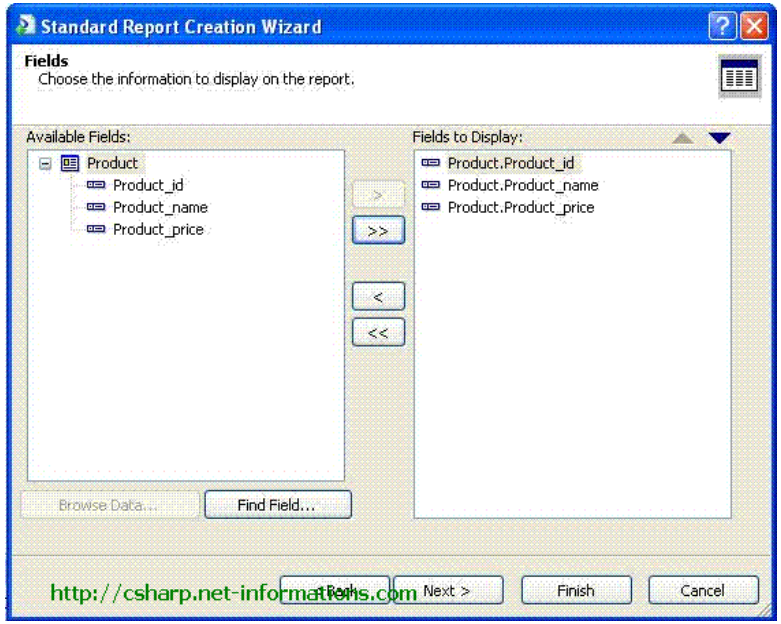


- After you click the finish button , the next window you will get your Server name under OLEDB Connection, from there selected database name (Crystaldb) and click the tables , then you can see all your tables from your database.

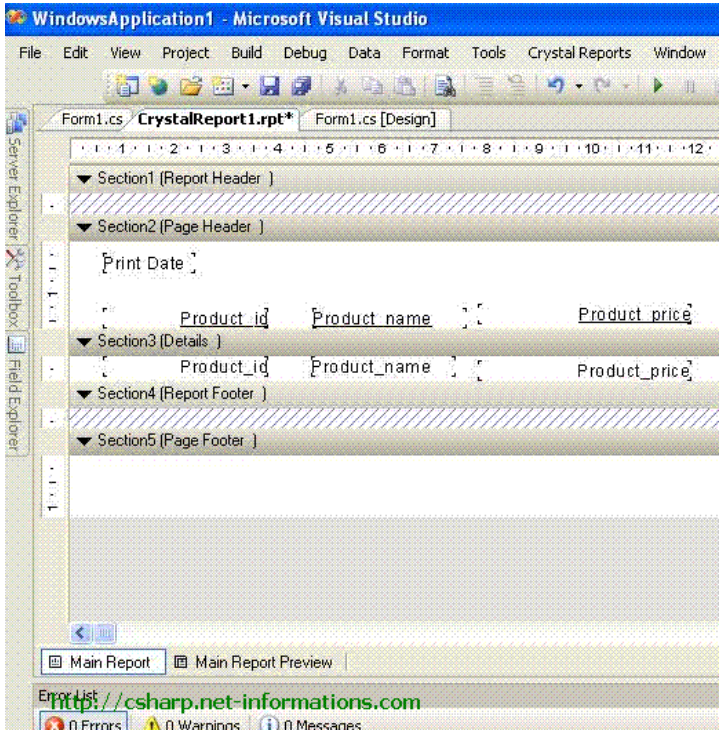
- From the tables list double click the Product table then you can see the Product table will come in the right side list.



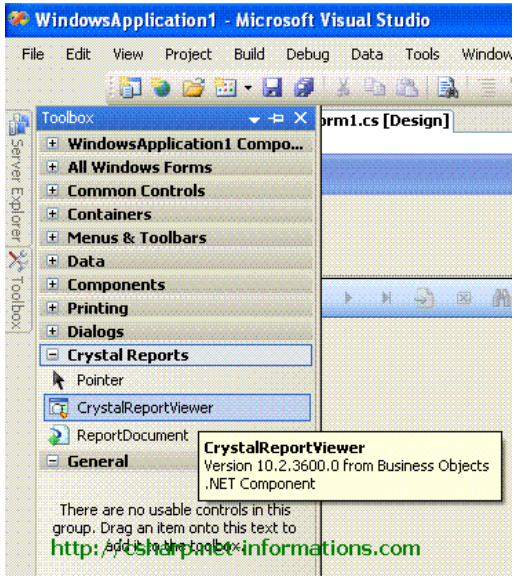
- Click Next Button
- Select all fields from Product table to the right side list .



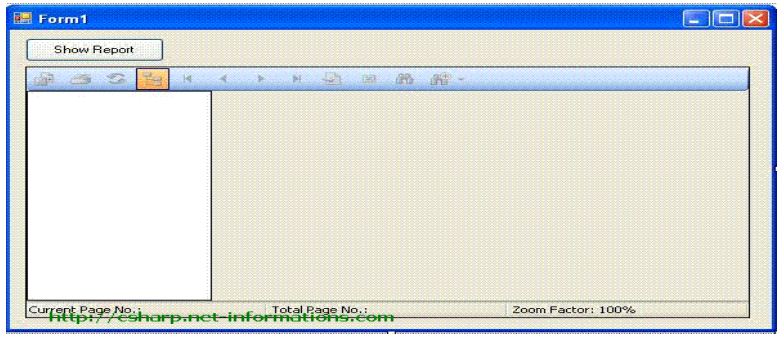
- Click Finish Button.
- Then you can see the Crystal Reports designer window in your C# project.
- In the Crystal Reports designer window you can see the selected fields from Product table.
- You can arrange the field Objects and design of the screen according your requirements. After that your screen is look like the following picture.



- Now the designing part is over and the next step is to call the Crystal Reports in your C# application and view it through Crystal Reports Viewer control in C#.
- Select the default form (Form1.cs) you created in C# and drag a button and a CrystalReportViewer control to your form .



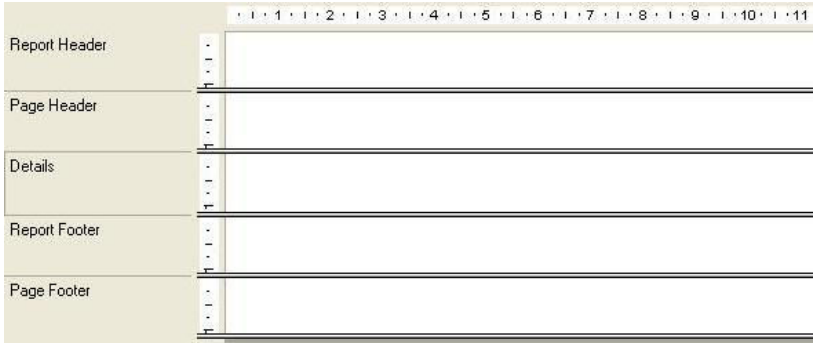
- After you drag the CrystalReportViewer to your form , it will look like the following picture.



- **Report Sections:-**

  - Sections are the design areas which you use to build your report.
  - Crystal Reports by default provides five main sections:

    - Report Header – fields placed in this section are printed once, at the beginning of the report
    - Page Header - fields placed in this section are printed at the beginning of each new page
    - Details – fields in this section are printed with each new record
    - Report Footer - fields placed in this section are printed once, at the end of the report
    - Page Footer - fields placed in this section are printed at the bottom of each new page



More sections will appear if you add groups or simply insert new one.

Working with sections

To insert a new section you need to:

- **Click Section Expert button in the top menu. Section Expert contains a list of all sections in the report**

- Select section and click insert. New section will appear after the section you have selected. If there are more than one section of a kind, there appear with lettered a, b, c and so on

To delete a section:

- Open Section expert
- Select section you want to delete
- Click Delete. Note that you can only delete section if there are lettered. In mean that you cannot delete sections originally provided by Crystal Reports

To move a section (change sections' order):

- Open Section Expert
- Select section you want to move
- Use up and down arrows to change order of the sections. Although the alphabetic order of sections is the same, their data and display order is changed

To merge sections you need to:

- Open Section Expert
- Move sections so the sections you want to merge fallow each other
- Select the top section
- Click merge. Section will be merged with section that is next on the list

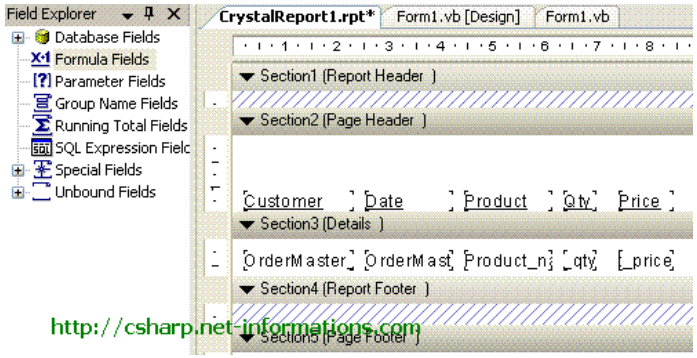
To split sections:

- Click on the on the boundary of the section you want to split. Horizontal line that splits section will appear
- Drag-and-drop it to the place where you want to split section

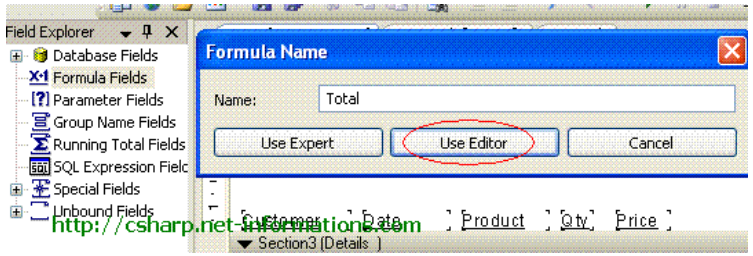
➢ Formula, Special Field and Summary in Report:-

Formula Field:

❖ The following C# - Crystal Reports section describes how to add a formula field in the Crystal Reports .

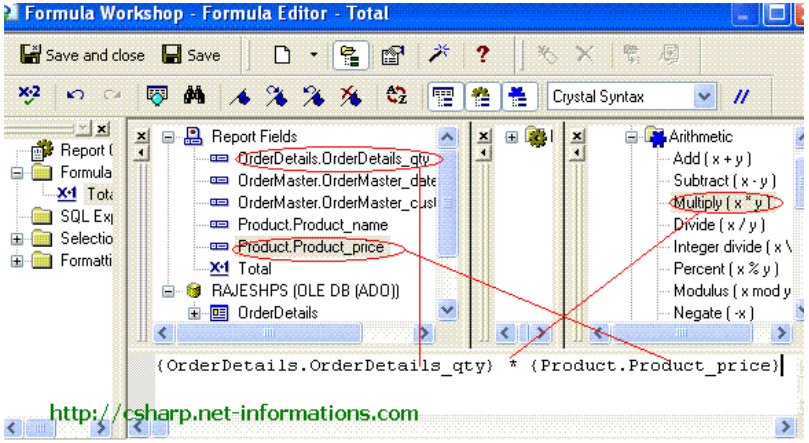


❖ Next step is to create a Formula Field for showing the result of **Qty X Price** .
❖ Right Click the Formula Field in the Field Explorer and click New. Then you will get an Input Message Box , type Total in textbox and click Use Editor.

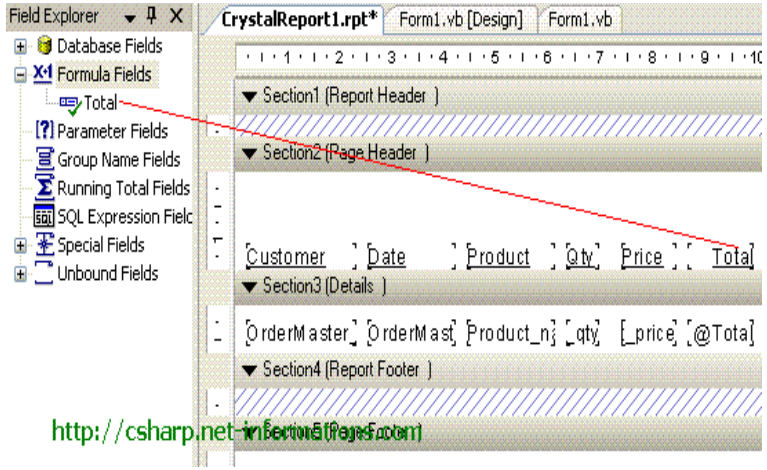


❖ Now you can see the Formula Editor screen . Here you can enter which formula you want . Here we want the result of Qty X Price . For that we select

OrderDetails.Qty , the multipy operator (*) and Product.Price . Double click each field for selection.

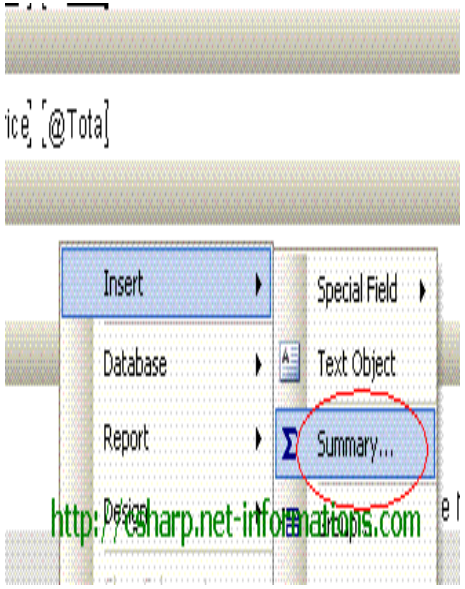


- Now you can see Total Under the Formula Field . Drag the field in to the Crystal Reports where you want to display Total.

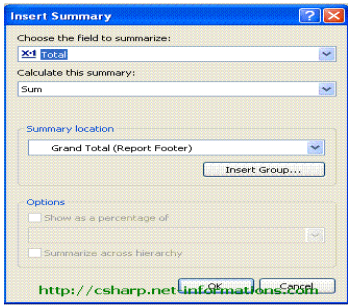


Summary Field:

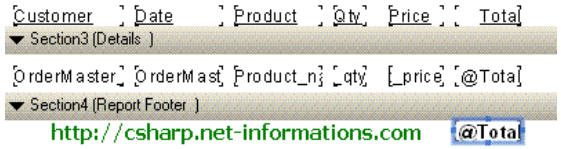
- The following C# - Crystal Reports section describes how to add a summary field in the Crystal Reports.
- In the Crystal Reports designer view window, right click on the Report Footer , just below the Total field and select Insert -> Summary .



❖ Then you will get a screen , select the Total from the combo box and select Sum from next Combo Box , and summary location Grand Total (Report Footer) . Click Ok button



❖ Now you can see @Total is just below the Total field in the report Footer.



➢ Creating Setup Projects

❖ Setup projects are used to create Windows Installer (.msi) files, which are used to distribute your application for installation on another computer or Web server. There are two types of setup projects. Standard setup projects create installers that install Windows applications on a target computer; Web setup projects create installers that install Web applications on a Web server.

Note:

The dialog boxes and menu commands you see might differ from those described in Help depending on your active settings or edition. To change your settings, choose Import and Export Settings on the Tools menu. For more information, see Working with Settings.

To create a new setup project

1. On the File menu, point to Add, then click New Project.
2. In the resulting Add New Project dialog box, in the Project Types pane, open the Other Project Types node, open Setup and Deployment Projects, and select Visual Studio Installer.
3. In the Templates pane, choose Setup Project for a standard setup, or Web Setup Project for a Web application.

To add an existing setup project to a solution

1. On the File menu, point to Add, then click Existing Project.

2. In the resulting Add Existing Project dialog box, browse to the location of the setup project and click Open.

File System Editor:

- The **File System** editor presents a graphic representation of the files, folders, and shortcuts your product installation creates or modifies on the target machine. Use this editor to:
  - Add and delete files, folders, and shortcuts in the installer project.
  - Control where to install folders and files on the target machine.
  - Establish and modify the file structure of installer components.
- You can open this editor by either:
  - Double-clicking **File System** in the **Target Machine** node in the **Project Explorer** window.

  –or–

  - Selecting the **File System** option on the **Project** menu.
- As you work in the **File System** editor, you can add, modify, or delete folders, files, and shortcuts with:
  - The context menu.

  Note : You can display the context menu by right-clicking in the **File System** editor. Context menu activated options differ depending on what is selected when you right-click.

  - Options on the **Actions** menu.

  Note **Actions** menu active options differ depending on your current file or folder selection.

  - The DELETE key (for item removal).

Options

Left pane of File System editor

The left pane of the **File System** editor shows a folder view of the target machine. You can add and delete folders your product installs by setting them up here. You can also establish contents of target machine folders. Microsoft® Visual Studio® Installer provides these initial folders for you to work with:

- **Application Folder**. Sets the root directory for your application on the target machine.
- **User's Desktop**. Place items (such as files or shortcuts) you want to install to the user's desktop in this folder.
- **User's Start Menu**. Place items (such as files or shortcuts) you want to install to the user's Start menu in this folder.

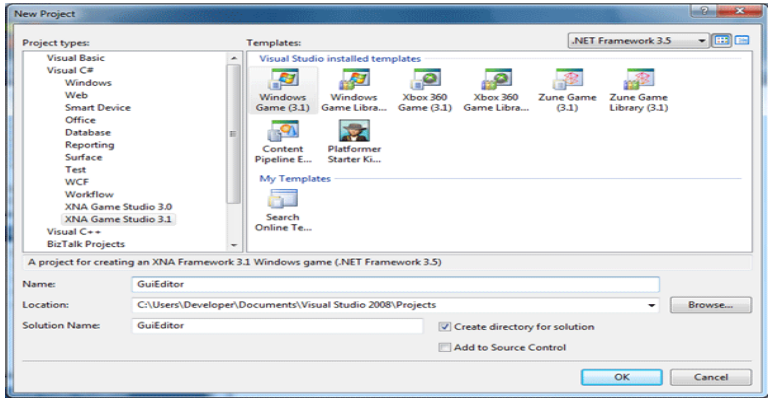
  Note If the installer project contains a Microsoft® Visual Basic® project with dependencies that can't be mapped to merge modules, a fourth folder labeled **Windows System Folder** is displayed in the left pane of the **File System** editor.

User Interface Editor:

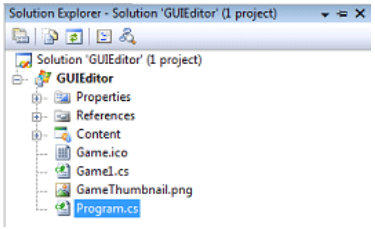
- "GUI Editor" or "GUI Builder" is a Software Development Tool. It has a kind of WYSIWYG structure and helps the user to build a structure without coding or less coding. Without GUI's we had to write code for everything.

STEP BY STEP BUILDING A "GUI EDITOR"

First of all, we are creating a new XNA 3.1 Game Project...



Here it is..

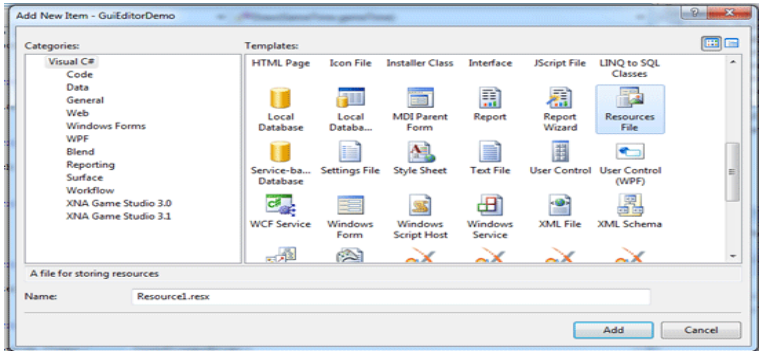


1. Add a resource file that includes Skin images for controls.
2. A Structure that will help us export the control List as an XML File.
3. A "Screens" Folder that will help us to read this controls.
4. For Communication of Windows & XNA creating a class where we will declare "public static" variables
5. A "Properties Panel" where we can change the properties of selected controls.
6. A "Toolbox Panel" where we can add controls.
7. Our Custom Controls

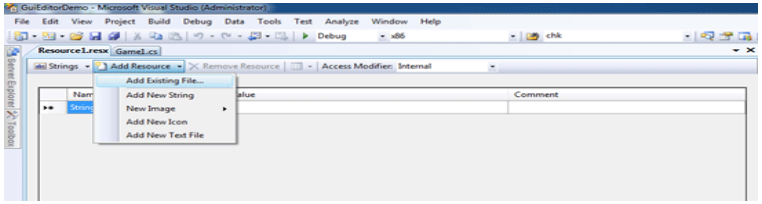
1. Add a resource file that includes Skin images for controls

We are adding 3-4 skin images for our Button control...

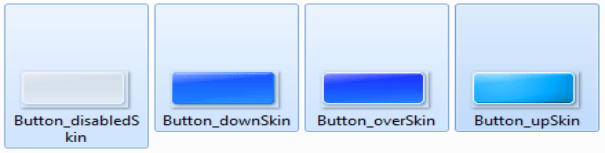
First of all we're creating a Resource File that will help us storing the skins:



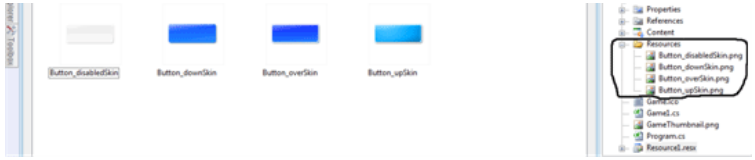
After that open the resource file and "Add Resource->Add Existing File..."



Then add 4 sample skins(you can create your own skins named 'disabled','down','over' and 'up'):

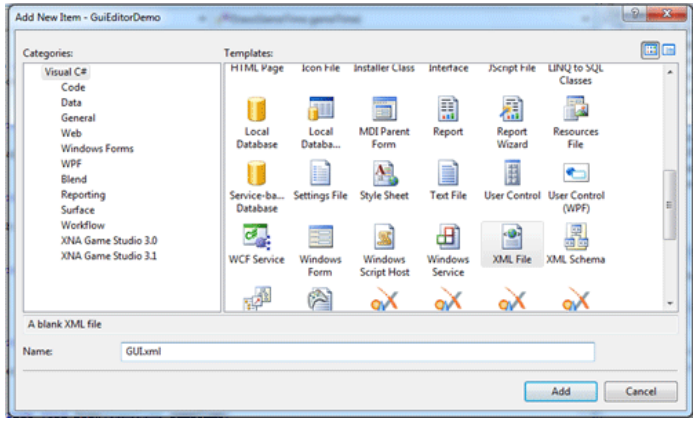


Here they are added at Resources...



2. A Structure that will help us export the control List as an XML File

First creating a new XML File:



Then make it similar to the codes below:

```
<?xml version="1.0" encoding="utf-8" ?>
<GUI>
 <Control Name="">
   <Type></Type>
   <AllowDrop></AllowDrop>
   <Enabled></Enabled>
   <ForeColor></ForeColor>
   <LocationX></LocationX>
   <LocationY></LocationY>
   <SizeW></SizeW>
```

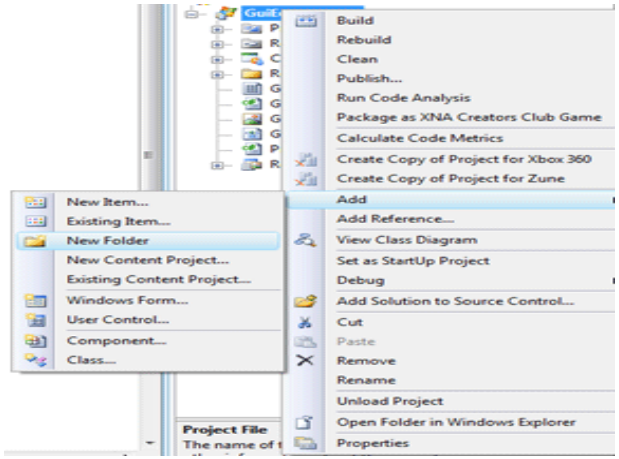


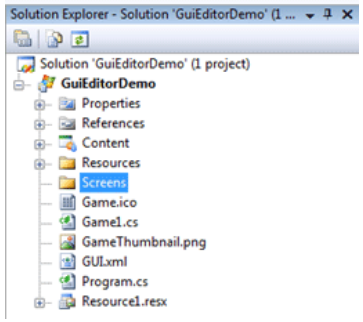
```
   <SizeH></SizeH>
   <Text></Text>
 </Control>
</GUI>
```

We are taking advantage of XML for the structure we will be building on GUI Editor & as you can see we added elements as they are already properties of the controls.Actually it would be a big mistake not to use XML in this kind of applications.

3. A "Screens" Folder that will help us to read this controls.

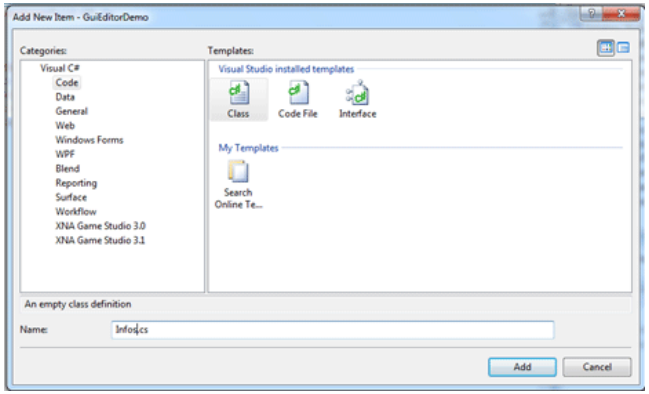
Just create a new Folder in the project and call it Screens.



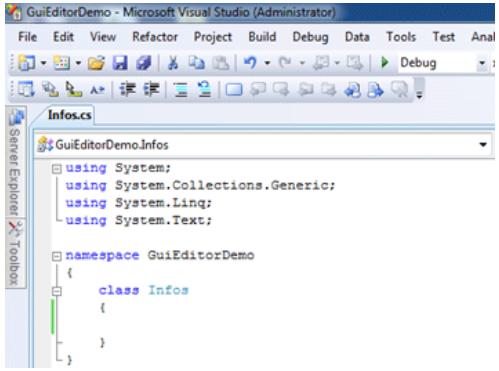


4. For Communication of Windows & XNA, creating a class where we will declare "public static" variables

Create a new Class & name it Infos.cs...



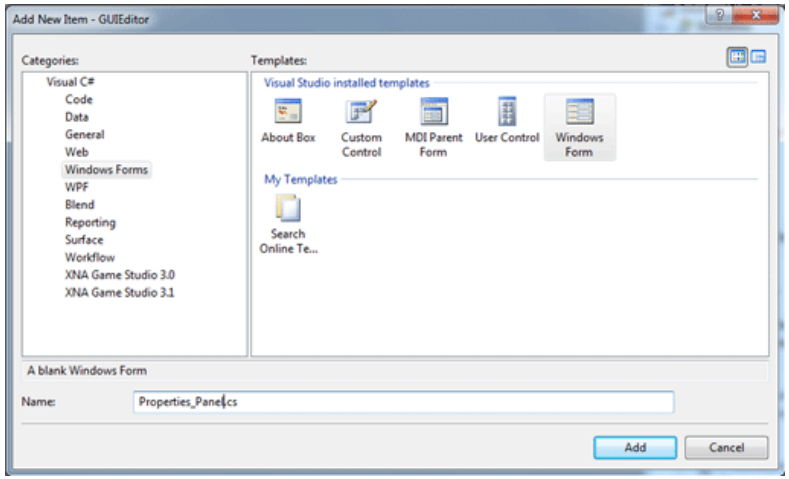
It seems just like this:

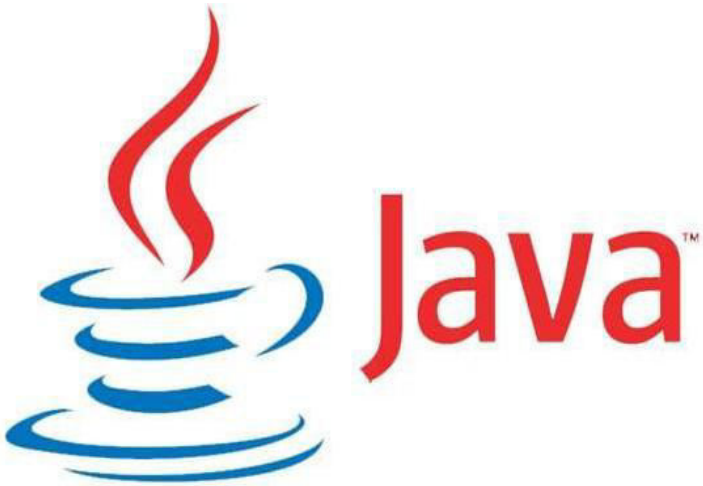


In this point we aren't writing any code.

5. A "Properties Panel" where we can change the properties of selected controls

We need to add a Properties Panel. First we need to create a Windows Forms:





CS-19 Programming with JAVA

B.C.A. Semester – 4

Unit-1

History, Introduction, Basics & OOP

Gaurav K Sardhara

Mo:-9067351366

1) Who invented Java Language?

James Gosling @ Sun Microsystems in 1991.

2) Why Initially Java was known as "Oak"?

Oak is a symbol of strength and chosen as a national tree of many countries like U.S.A., France, Germany, Romania etc. In 1995, Oak was renamed as **"Java"**.

3) List-out characteristics of Java Language?

1. Simple
2. Object-Oriented
3. Platform independent
4. Secured
5. Robust
6. Architecture neutral
7. Portable
8. Dynamic
9. Interpreted
10. High Performance
11. Multithreaded
12. Distributed

4) Why Java is platform independent?

Java code can be run on multiple platforms e.g. Windows, Linux, Sun Solaris, and Mac/OS etc.

Java code is compiled by the compiler and converted into byte code. This byte code is a platform independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere (WORA).

5) What is Byte code in java?

Byte Code is a highly optimized set of instructions designed to be executed by the Java run time system, which is called Java Virtual Machine (JVM).

6) What is JVM?

The JVM is the software platform that provides runtime environment for running Java application and byte code interpretation.

7) List-out various Java Editions.

There are mainly three edition of Java for various purposes.

- J2SE (Java 2 Standard Edition)
- J2EE (Java 2 Enterprise Edition)
- J2ME(Java 2 Micro Edition)

8) List-out various data types in Java.

There are mainly two types of data types available in Java language.

1) ***Primitive data types:***
   a. *Numeric-Type*
      i. Real
         1. Float
         2. Double
      ii. Integer
         1. Byte
         2. Short
         3. Int
         4. Long
   b. *Non-Numeric-Type*
      i. Characters (char)
      ii. Boolean
2) ***Non-Primitive Types***
   a. Array
   b. Class
   c. Interface

9) List-out various Java Tokens.

The java language contains following tokens.

1) Identifier
2) Keyword
3) Literal
4) Whitespace
5) Separator
6) Operator

10) **What is type conversion? *Or***
What is Automatic type conversion in Java?

Java automatically converts variables of lower precision type to higher precision type during the evaluation of expression or during assignment. This is known as automatic type conversation or automatic promotion.

11) What is type casting?

There are many situation occurs that you need that conversion, when a higher precision type is to be assigned to a lower precision type.

12) What is Jagged Array?

In Jagged arrays, each row, in a two-dimensional array, may contain different lengths.

13) What is the role of Garbage Collector?

In Java Garbage collector automatically handles de-allocation of memory. When no references to an object to an object exist, that object is assumed to be no longer needed, and the memory occupied by the object can be reclaimed.

14) List-out various types of access modifiers available in Java language?

These are…

- Public
- Private
- Protected
- Default

15) What is the use of static keyword?

Java is allowed to create a member that can be used itself, without reference to a specific instance. To create such a member, precede its declaration with the keyword **static**.

16) How this keyword works?

It can be used inside any method to refer to the current object.

17) Define the role of finalize() method.

User can write finalize() in any class to execute statements at the time of garbage collection.

18) How final keyword is used?

The final keyword is used to restrict the change in value of variable or does not allow overridden of methods and inheritance of classes.

19) What is command-line argument?

The command-line arguments are the value that pass with the main method during the execution of program.

20) What is constructor overloading?

Constructor having the same name with different parameter list is called constructor overloading.

21) Define Method Overloading.

Overloading methods is the concept of polymorphism. It is possible in java to define two or more methods within the same class that share the same name, with different parameter declaration. This process is known as method overloading.

22) Define Constructor. List-out its types.

A constructor allows objects to initialize themselves as soon as they are created.

· A constructor automatically called when an object is created.

· It has same name as the class in which it resides and is syntactically similar to a method.

There are mainly two types:

- Parameterized
- Non-Parameterized

23) What is the use of new keyword?

An object is created from a class using new operator.

· The new operator after creation of the object returns the object reference. It can be stored in a variable.

Q. ***Explain History of Java Language.***

History of Java

Java history is interesting to know. The history of java starts from Green Team. Java team members (also known as **Green Team**), initiated a revolutionary task to develop a language for digital devices such as set-top boxes, televisions etc.

For the green team members, it was an advance concept at that time. But, it was suited for internet programming. Later, Java technology as incorporated by Netscape.

Currently, Java is used in internet programming, mobile devices, games, e-business solutions etc. There are given the major points that describes the history of java.

1) **James Gosling**, **Mike Sheridan**, and **Patrick Naughton** initiated the Java language project in June 1991. The small team of sun engineers called **Green Team**.

2) Originally designed for small, embedded systems in electronic appliances like set-top boxes.

3) Firstly, it was called **"Greentalk"** by James Gosling and file extension was ".gt".

4) After that, it was called **Oak** and was developed as a part of the Green project.

Why Oak name for java language?

5) **Why Oak?** Oak is a symbol of strength and chosen as a national tree of many countries like U.S.A., France, Germany, Romania etc.

6) In 1995, Oak was renamed as **"Java"** because it was already a trademark by Oak Technologies.

Why Java name for java language?

7) **Why they choose java name for java language?**
The team gathered to choose a new name. The suggested words were "dynamic", "revolutionary", "Silk", "jolt", "DNA" etc. They wanted something that reflected the essence of the technology: revolutionary, dynamic, lively, cool, unique, and easy to spell and fun to say.

According to James Gosling "Java was one of the top choices along with **Silk**". Since java was so unique, most of the team members preferred java.

8) Java is an island of Indonesia where first coffee was produced (called java coffee).

9) Notice that Java is just a name not an acronym.

10) Originally developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995.

11) In 1995, Time magazine called **Java one of the Ten Best Products of 1995**.

12) JDK 1.0 released in (January 23, 1996).

Q. Explain Characteristics of Java Language.

Features of Java / Characteristics of JAVA / JAVA Buzzwords:

There is given many features of java. They are also known as java buzzwords. The Java Features given below are simple and easy to understand.

1. Simple
2. Object-Oriented
3. Platform independent
4. Secured
5. Robust
6. Architecture neutral
7. Portable
8. Dynamic
9. Interpreted
10. High Performance
11. Multithreaded
12. Distributed

✓ **[1] Simple**

There are various features that makes the java as a simple language. According to Sun, Java language is simple because:

- Syntax is based on C++ (so easier for programmers to learn it after C++).
- Programs are easy to write and debug because java does not use the pointers explicitly. Removed many confusing and/or rarely-used features e.g., explicit pointers, operator overloading etc.
- No need to remove unreferenced objects because there is Automatic Garbage Collection in java.
- Java provides the bug free system due to the strong memory management. It also has the automatic memory allocation and de-allocation system.

✓ **[2] Object-oriented**

Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior.

Object-oriented Programming (OOPs) is a methodology that simplify software development and maintenance by providing some rules.

Java is a true object oriented languages. All programs code and data reside within object and classes. The object model in java is simple and easy to extend.

Basic concepts of OOPs are:

1. Object
2. Class
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation

Inheritance: It is the process of creating the new classes and using the behavior of the existing classes by extending them just to reuse the existing code and adding the additional features as needed.

Polymorphism: As the name suggest one name multiple form, Polymorphism is the way of providing the different functionality by the functions having the same name based on the signatures of the methods.

Encapsulation: It is the mechanism of combining the information and providing the abstraction.

As the languages like Objective C, C++ fulfills the above four characteristics yet they are not fully object oriented languages because they are structured as well as object oriented languages. But in case of java, it is a fully Object Oriented language because object is at the outer most level of data structure in java. No standalone methods, constants, and variables are there in java. Everything in java is object even the primitive data types can also be converted into object by using the wrapper class.

✓ **[3] Platform Independent**

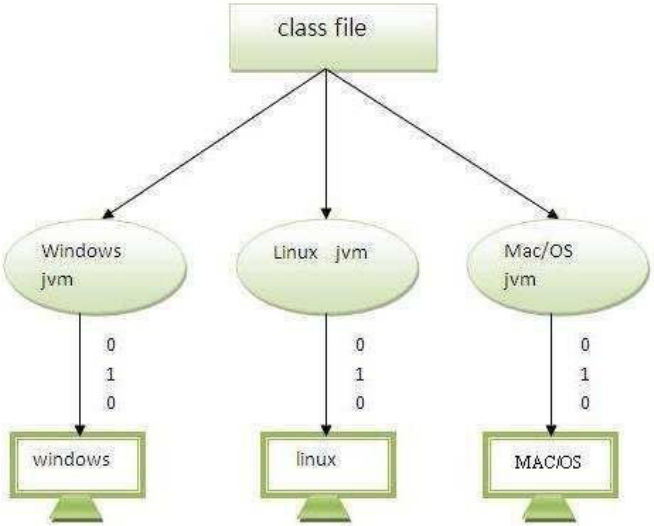
A platform is the hardware or software environment in which a program runs. There are two types of platforms software-based and hardware-based. Java provides software-based platform. The Java platform differs from most other platforms in the sense that it's a software-based platform that runs on top of other hardware-based platforms.

It has two components:

1. Runtime Environment
2. API (Application Programming Interface)

Java code can be run on multiple platforms e.g. Windows, Linux, Sun Solaris, and Mac/OS etc.

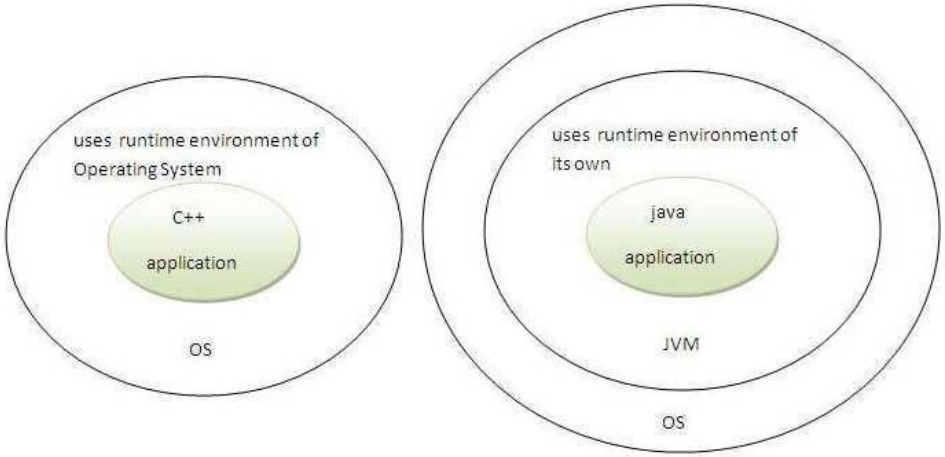
Java code is compiled by the compiler and converted into byte code. This byte code is a platform independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere (WORA).



✓ **[4] Secured**

Java is secured because:

Java does not use memory pointers explicitly. All the programs in java are run under an area known as the virtual machine sand box.



1. **Byte code Verifier:** verifies before it run on user's computer to ensure that it does not violet restriction by checking the validity of the objects, classes, variables & methods used in the java program.

2. **Class loader:** adds security by separating the package for the classes of the local file system from those that are imported from network sources.

   AND

   Applet loader: determines how and when an applet is allowed to use classes in a runtime java environment and applet should not replace important components of java runtime environment.

3. **Security Manager:** determines the accessibility options of a class like reading and writing a file to the local disk.

Java uses the public key encryption system to allow the java applications to transmit over the internet in the secure encrypted form.

These security are provided by java language. Some security can also be provided by application developer through SSL, JAAS, and Cryptography etc.

✓ **[5] Robust**

Robust simply means strong; so Java is healthy & strong compare to another language.

Java has the strong memory allocation and automatic garbage collection mechanism.

It provides the powerful exception handling and type checking mechanism as compare to other programming languages.

Compiler checks the program whether there any error and interpreter checks any run time error and makes the system secure from crash.

All of the above features makes the java language robust.

✓ **[6] Architecture-neutral**

The term architectural neutral seems to be weird, but yes Java is an architectural neutral language as well. A program of JAVA written and compiled in one platform can run on any other platform. Means "write once than run anywhere & any time for life time.

The growing popularity of networks makes developers think distributed. In the world of network it is essential that the applications must be able to migrate easily to different computer systems. Not only to computer systems but to a wide variety of hardware architecture and operating system architectures as well.

The Java compiler does this by generating byte code instructions, to be easily interpreted on any machine and to be easily translated into native machine code on the fly. The compiler generates an architecture-neutral object file format to enable a Java application to execute anywhere on the network and then the compiled code is executed on many processors, given the presence of the Java runtime system. Hence Java was designed to support applications on network. This feature of Java has thrived the programming language.

✓ **[7] Portable**

Java Programs can be easily moved from one system to another system.

Java portability in two ways.

1. Java compiler generates byte code instruction that can be implemented on any machine.
2. The Size of the primitive data types is machine - independent.

The feature Write-once-run-anywhere makes the java language portable provided that the system must have interpreter for the JVM. Java also have the standard data size irrespective of operating system or the processor. These features makes the java as a portable language.

✓ **[8] Dynamic**

While executing the java program the user can get the required files dynamically from a local drive or from a computer thousands of miles away from the user just by connecting with the Internet.

✓ **[9] Interpreted**

We all know that Java is an interpreted language as well. With an interpreted language such as Java, programs run directly from the source code. The interpreter program reads the source code and translates it on the fly into computations. Thus, Java as an interpreted language depends on an interpreter program. The versatility of being **platform independent** makes Java to overtake from other languages. The source code to be written and distributed is platform independent. Another advantage of Java as an interpreted language is its error debugging quality. Due to this any error occurring in the program gets traced. This is how it is different to work with Java.

✓ **[10] High-performance**

Java is faster than traditional interpretation since byte code is "close" to native code still somewhat slower than a compiled language (e.g., C++)

Java uses native code usage, and lightweight process called threads. In the beginning interpretation of byte code resulted the performance slow but the advance version of JVM uses the adaptive and just in time compilation technique that improves the performance.

✓ **[11] Multi-threaded**

As we all know several features of Java like Secure, Robust, Portable, dynamic etc.; you will be more delighted to know another feature of Java which is Multithreaded.

Java is also a multithreaded programming language. Multithreading means a single program having different threads executing independently at the same time. Multiple threads execute instructions according to the program code in a process or a program. Multithreading works the similar way as multiple processes run on one computer.

ultithreading programming is a very interesting concept in Java. In multithreaded programs not even a single thread disturbs the execution of other thread. Threads are obtained from the pool of available ready to run threads and they run on the system CPUs.

The main advantage of multi-threading is that it shares the same memory. Threads are important for multi-media, Web applications etc.

✓ **[12] Distributed**

The widely used protocols like HTTP and FTP are developed in java. Internet programmers can call functions on these protocols and can get access the files from any remote machine on the internet rather than writing codes on their local system.

We can create distributed applications in java. RMI and EJB are used for creating distributed applications. We may access files by calling the methods from any machine on the internet.

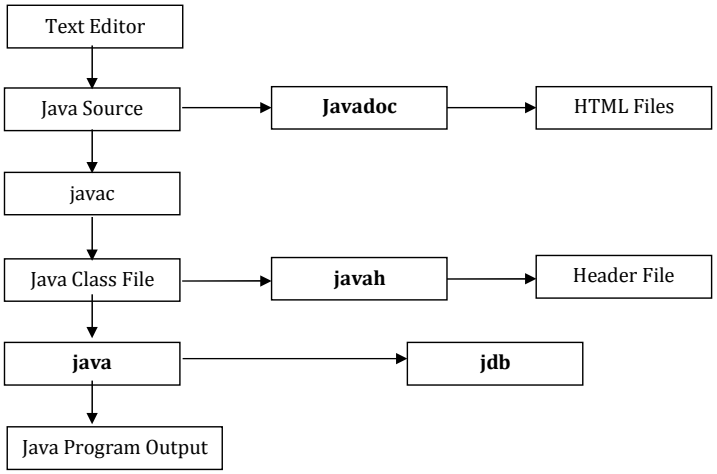
How JAVA changed Internet

1) **JAVA Applet**
   - An applet is a special kind of JAVA program that is designed to be transmitted over the Internet and automatically executed by a JAVA-compatible web browser.
   - If the user clicks a link that contains an applet, the applet will be automatically downloaded & run in the browser.
   - There are two very broad categories of object that are transmitted between the server & client : Passive Information & Dynamic
2) **Security**
3) **Portability**

Q. Explain various components of JDK.

JDK & Its Components:

- The JDK (JAVA Development Kit) provides collection of tools that are used to develop and run JAVA Programs.
  - **appletviewer** (Use : View of java Applets, We can't run java applets)
  - **javac** (Use : java Compiler, It translate source code to byte code files the interpreter can understand)
  - **java** (Use : java interpreter, it runs application by reading and interpreting byte code files)
  - **javap** (Use : java disassembler, it enables one to convert byte code files a program description)
  - **javah** (Use : C header files, provides header files for uses methods)
  - **javadoc** (Use : Create HTML document, it creates HTML documentation from java source code files)
  - **jdb** (Use : java debugger, it helps one to find errors in a programs)
- These different tools are applied to build and run application program as shown below :



- Different classes are available in packages as a part of Java standard Library (JSL) also known as Application Programming Interface (API).
- Different packages available with API are given below :

Language support Package:

- A collection of classes and methods required for implementing basic features of Java is stored in this package. i.e. **java.lang**

Utility Package:

- A collection of classes to provide utility function such as Date, time etc… are stored in utilities package. i.e. **java.util**

Input / Output Package:

- A collection of classes required for input / output is stored in this package. i.e. Java.io

Networking Support Package:

- This package includes classes and methods for communicating with other computers via internet. Data & program code can be set using this. i.e. java.net

AWT Package:

- AWT stands for Abstract Window Toolkit
- The awt package contains classes that implements graphical user interface. i.e. GUI
- Different components of window can be used using the classes of this package. i.e. **java.awt**

Applet Package:

- This includes a set of classes & methods which allows us to create applet. i.e. **java.applet**

Q. Explain Byte Code in Java.

Java's Magic: The Byte Code

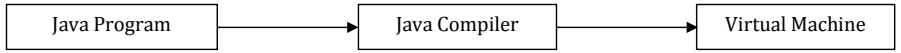
- The key that allows solving the security and portability problems just described is that the output of JAVA compiler is not executable code. Rather it is Byte Code.
- Byte Code is a highly optimized set of instructions designed to be executed by the Java run time system, which is called Java Virtual Machine (JVM).

- The originally JVM designed as interpreter for Byte Code.

Q. What is the role of JVM?

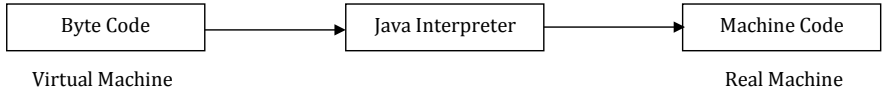
Java Virtual Machine (JVM):

- All language compilers translate source code into machine code for specific computer.
- Java compiler also does the same thing. Java Compiler produces and intermediate code known as byte code for a machine that does not exist.
- This machine is called the Java Virtual Machine & it exits only inside the computer memory.



Process of Complication

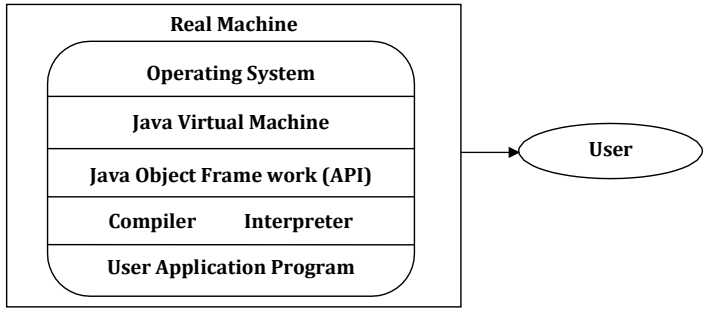
- The virtual machine code (Byte code) is not machine specific. The machine specific code is generated by the Java Interpreter by acting as an n intermediary between the virtual machine and the real machine as shown in bellow figure.



Process of converting Byte Code into Machine Code

Note: Interpreter is different for different machine.

- The following figure shows how user can interact with real machine for Java programming.



Q. Explain types of Java program.

Types of JAVA Program:

- In JAVA Language there are two types of programs can be written
  1) **Application**
  - It is regular program and it is run in a PC (Personal Computer) by the OS (operating System).
  2) **Applet**
  - It is web browser. Applet can also run with a program called applet viewer – It is debugging aid a final environment to run applet.

A First Simple Program

```
/*  this is a simple JAVA program Call the files "Example.java".   */
class Example
{
   // your programs begin with a call to main().
   public static void main(string args[])
   {
         System.out.println ("This is a simple JAVA Program");
   }
}
```

If you want to run a program than follow next steps:

1) First you type a program in Notepad & save with same class name with extension of java.
2) Open Command Prompt
3) First go to your **"*.java"** file folder with the use of command prompt and then set path to

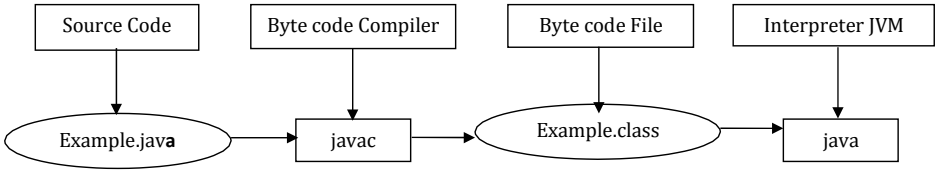
```
set path= C:\Program Files\Java\jdk1.5.0_14\bin
```

4) Compile the JAVA File with **javac** <file Name with extension>
5) Execute it with **java** <file name without extension>

Entering the Program:

- For most computer languages, the name of the file that holds the source code to a program is immaterial.
- About Java is that the name you give to a source file is very important.
- In this example name of source file should be Example.java.
- As you see by looking at the program, the name of the class defined by the program is also Example. This is not a coincidence.
- In Java, all code must reside inside a class. The name of that class should match the name of the file that holds the program.
- You should also make sure that the capitalization of the filename matches the class name.

Compiling & Execution of the Program:



- To compile the Example program. Execute the compiler using **javac**, specifying the name of the source file on the command line as `C:\> javac Example.java`
- The **javac** compiler creates a file called **Example.class** that contains the byte code version of the program (The output of **javac** is not code that can be directly executed.
- To actually run the program, you must use the Java application launcher called **java**. like `C:\> java Example`
- When program is run successfully the following output displayed : This is a simple JAVA Program

A Closer look at the First Simple Program

- Java is case sensitive.
- Java compiler will compile classes that don't contain a main () method but java has no ways to run these classes.

Comments:

There are three types of comments

1) **Single line comment**
```
// this is single line comment
```
2) **Multiline comment**
```
/*    This is multiline comments,
      where we can write more than one line in comments     */
```
3) **Documentation comment**

✓ **class Example**

- This line uses the keyword class to declare that a new class is being defined.
- Example is an identifier that is the name of the class
- The entire class definition including all of its members, will be between the opening curly brace ({) and the closing curly brace (})

Q. Explain main method of Java.

public static void main(String args[])

main ():

- main is a method called when a java application begins.

public:

- The public keyword is an access specifier, which means that the content of the following block accessible from all other classes.

Static:

- The keyword static allows main () to be called without having to a particular instance of a class.

Void:

- The keyword void tells the compiler that main () does not return a value. The methods can return value.

String args [ ]

- Declare a parameter named args, which is an array of instance of the class string.

- args [ ] receives any command-line argument present when the program is executed (arrays are collection of similar objects).

System.out.println ( )

System: It is predefined class that provides access to the system.

out: it is the output stream that is connected to the console.

println ( ): it is displays the string which is passed to it (output is the built-in println() method)

Q. Explain Structure of Java Language.

Structure of JAVA

- A Java program may contain more than one class but out of them one class has a main method.
- Classes may contain many data & methods.
- The structure of java program is

Name	Requirement
Documentation Section / statement	Suggested but optional
Package section / Statement	optional
Import section / statement	optional
Class Definition	compulsory
Main method Class	compulsory

Documentation Section / statement:

- In this section, we can write program definition, developed by, managed by, class name program name type all information.

Package Section / statement:

- In Java program first statement one can write is a package statement.
- This statement declares a package name and tells the compiler that the classes included here belong to this package. This is optional in any java program.
- **Example :** package sybca;

Import section / statement:

- Import statements can be write after package statements and before the class definition.
- Once a package is declared one can write any number of import statements
- **Example :** import sybca.sem4; **OR** Import sypgdca.sem2;

Class Definition

- In Java program there is more than one class.
- In this class definition we have to provide keyword class along with the class name.
- A class can include variable s as well as methods.

Main method Class

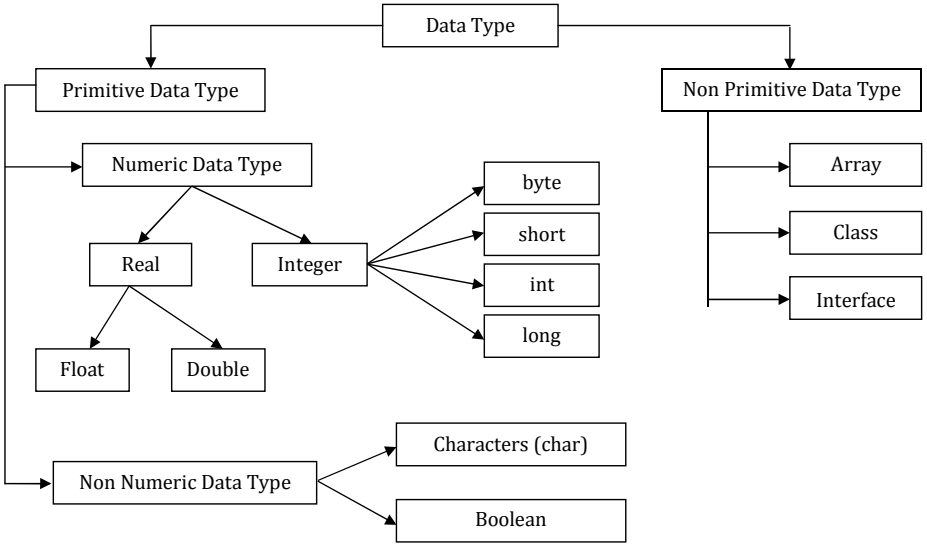
- In Java program one class is always need that is main class. In main method one can create object of different classes & with the use of these objects we can call the methods as well as variables of

those classes. After end of main method. Program terminates and control passes back to the Operating system

Q. Explain various Data types of Java.

Data Types in JAVA

- Every variable has a type, every expression has type & every type has to be defined.
- Data type defined that which type of values is stored in a variable and it also defined the size of a data type.
- Different Data types allow the programmer to use suitable data type for the application.
- All assignments, whether explicit or via parameter passing in method calls, are checked for type compatibility.
- Java compiler checks all expressions & parameters to ensure that the types are compatible.
- Any type mismatches are errors that must be correct before the compiler will finish compiling the class.
- In java mainly two types of data types categories :
  1. Primitive Data Types
  2. Non-Primitive Data Types



Primitive Data Types

- Primitive types represent single value not complex objects.
- The primitive types are defined to have an explicit range & mathematical behavior.
- Java defines eight primitive types of data: byte, short, int, long, char, float, double and Boolean.
- The primitive data types are also refereed as a simple data types. These can be put in four groups :
  1. **Integer :**

- This group includes byte, short, int & long which are for whole valued signed numbers.
- It can hold any whole number.
- The size of values that can be stored depend on the integer data type used for the variable
- Java does not support any unsigned data types and therefore al java values are signed mean positive or negative.
- we can make integer (int) as long by appending l or L after the number i.e. 123l or 123L

Name	Keyword	Width	Range	Example
Byte	byte	8	-128 To 127	byte a, b;
Short	short	16	-32,768 To 32,767	short x, y;
Integer	int	32	-2,14,74,83,648 To 2,14,74,83,647	int c, d;
Long	long	64	-92,23,37,20,36,85,47,75,808 To 92,23,37,20,36,85,47,75,807	long z, dist;

2. Real / Floating – Point :

- Floating point number as known as real numbers
- This group includes float & double which represent numbers with fractional precision.
- It holds numbers which contains fractional part such as 3.77 or -3.77.
- Floating point are treated as double precision numbers. To force them to a single precision mode we must append f or F to the number like 1.23f or 1.23F

Name	Keyword	Width	Range	Example
Float	float	32	1.4e-045 To 3.4e+038	float hightemp, lowtemp;
Double	double	64	4.9e-324 To 1.8e+308	double pi, r, s ;

3. Characters :

- This group includes char, which represents symbols in character set, like letters and numbers.
- To store character like alphabets, special symbols or digits. It can hold only single value at a time.

Name	Keyword	Width	Range
Character	char	2	0 to 127 (Standard Set of characters - ASCII)
			0 to 255 (If Extended 8 bit character set, ISO - Latin - 1)
			0 To 65536 (If System is 16 bit)

4. Boolean :

- This group includes Boolean which is special type for representing true / false values.
- This data type used when we want to test a particular condition during program evaluation.
- There are only two values that a Boolean data type can be take "True" or "False"
- It takes on one bit in the memory.

Q. Explain Variable in Java.

Variable

- The variable is the basic unit of storage in a Java program.
- A variable is defined by the combination of an identifier, a type, and an optional initializer.

- All variables have scope, which defines their visibility and a lifetime.

Declaring a Variable:

- In java all variables must be declared before they can be used.
- The basic form of a variable declaration is shown as :

```
Type identifier [ = value][, identifier [= value]….];
```

- Variable are separated by commas. After declaring variables statement must end with semicolon.

Example:

```
int a, b, c;
double pi = 3.14159;
char x = 'X';
byte z = 22;
```

Q. Explain Scope of variable.

The Scope & life Time of Variable:

- Java variables are classified into three kinds :

Instance variable:

- Instance variable & class variables are declared inside a class. Instance variable are created when the object of a class is created & therefore they are associated with the objects. They take different values for different objects.

Class Variable:

- Class variable are global to a class. It belongs to entire set of objects of a class. Only one memory location is created for class variable.

Local Variable:

- Variable which are declared inside any method is available to that method only.

It is not available to other method, so it is known as local / private variable which is available in a method in which it is declared.

Q. Explain Java Tokens.

Java Tokens

- A Java program is basically a collection of classes.
- A class is defined by a set of declaration statement and methods containing executable statement.
- The smallest unit of a program is called Tokens.
- Different types of tokens are given below :

Identifiers:

- Identifiers are programs designed token, they are used for naming classes, methods, variables, objects, labels packages & interfaces in a program.
- Java identifiers follows the basic rules for identifiers :
- They can have Alphabets, digits, underscore & dollar sign.
- They must not begin with digits.
- Uppercase and lowercase alphabets are distinct.

- They can be of any size.
- The conversation for identifiers are
  - Names of all methods & variables are of lowercase alphabets, **Example:** display (), sum, avg, etc….
  - When more than one word are used in a name, the second word are begin with uppercase alphabets**, Example:** totalMarks, dayTemperature, etc…
  - All classes and interfaces start with an uppercase alphabets & remaining alphabets are lowercase, **Example:** Sample, Student, Employee, Etc…
  - Variable that represent constant values are written in all uppercase alphabets, **Example:** PI, MAX, MIN, etc…

Keywords:

- Keywords are essential of any programming language.
- They implements specific features of any programming language.
- Java language has approx. 60 words as keywords (all keywords are written in lower case letters)
- Some of the keywords of C & C++ are not included in Java & some new keywords are added.
- These keywords combined with the syntax of the operators and separators from the foundation of java language.
- All the keywords are having specific meaning in JAVA (We can't use them as names of variables, class methods, function, etc...

abstract	assert	boolean	break	byte
case	catch	char	class	const
continue	default	do	double	else
extends	Final	finally	float	for
goto	If	implements	import	instanceof
int	interface	long	native	new
package	private	protected	public	return
short	static	strictfp	super	switch
synchronized	this	throw	throws	transient
try	void	volatile	while	enum

Literals:

- Literals in Java are a sequence of characters that represent constant values to be storage in a variable.
- Java has five types of literals as under :

1. **Integer literals**

   In java integer literals means whole numbers.

   There are three types of integer literals:

Name	Range	Values	Example
Decimal	0	0	0, 0326, 027894
Octal	8	0, 1, 2, 3, 4, 5, 6, 7	05, 0776, 0123
Hexa Decimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F	0x38, 0x2A5F, 0x4AB9

2. **Floating point literals**

   Numbers with decimal point and fractional values are called floating point literals

   Floating point are represented on of the two forms standard & Scientific

   In standard form, integer part & fractional part of literal are separated by a decimal point i.e.: 27.75

   In scientific form, the literal represented floating point followed by the letter 'e' or 'E' and signed integer. I.e. 5.248e-35 or 4.8592E28

3. **Char literals**

   Character literals in java are single character literals enclosed in single quotes. i.e. 'a' or 'A'.

   Java supports a set of character literals known as escape sequence which cannot be printed.

Escape Sequence	Description
\ddd	Octal character (ddd)
\uxxxx	Hexadecimal Unicode character (xxxx)
\'	Single quote
\"	Double quote
\\	Backslash
\r	Carriage return
\n	New Line (also known as line feed)
\f	Form feed
\t	Tab
\b	Backspace

4. **String literals**

   A sequence of characters enclosed in a pair of double quote constitute a string literal.

   "Hello World"

   "Two \nlines"

   "\"This is quotes \""

5. **Boolean literal**

   In java Boolean literals take values True or False. Means **True** is **1** & **False** is **0**.

White Space:

- Java is free-form language, means that you do not need to follow any special indentation rules.
- In Java whitespace is a space, tab or a new line.

Separator (Special Character):

- Separators are symbols used to indicate where group of code is divided and arrange.
- Different separators available in java are :

  Parenthesis ( ):

Parentheses are used to contain lists of parameters in method definition and invocation. Parentheses are used for defining precedence in expressions, contaminating expression in control statements and surrounding cast types.

Curly Braces ({ }) :

It is used to contain the values of automatically initialized arrays, Braces are used to define block of code for classes, methods and local scopes.

Square Brackets ([ ]) :

It is used to declare array types, also used when dereferencing array values.

Semicolon (;) :

It is used for terminate statements.

Comma (,) :

Separators consecutive identifiers in a variable declaration, also used to chain statements together inside a for statements.

Period (.) :

It is used to separator package names form sub packages and classes, also used to separate a variable of method form a reference variable.

Q. Explain Operators in Java.

Operators

- An operator is a symbol that used to perform certain mathematical or logical manipulations.
- Operators are used in programs to manipulate date and variables.
- Java operators can be classified into number of related categories as under :

1. Arithmetical Operator	5. Increment & Decrement Operator
2. Relational Operator	6. Conditional Operator
3. Logical Operator	7. Bitwise Operator
4. Assignment Operator	8. Special operator

❖ **Arithmetic Operator :**

○ Java provides the entire basic arithmetical operator.

Operator	Meaning
+	Addition or Unary plus
-	Subtraction or unary minus
*	Multiplication
/	Division
%	Module Division

❖ **Relational Operator :**

○ For comparing two quantities and depending on their relation after we can take certain decision.

Operator	Meaning
<	Is less than

<=	Is less than or equal to
>	Is greater than
>=	Is greater than or equal to
==	Is equal to
!=	Is not equal to

❖ **Logical Operator :**

○ Java has only three logical operator are as under :

Operator	Meaning
&&	Logical AND
||	Logical OR
!	Logical Not

❖ **Assignment Operator :**

○ It is used to assign the value of an expression to a variable.

○ Java has shorthand assignment operators which are used in the form of :

Statement with Simple Assignment Operator	Statement with Shorthand Operator
a = a + 1	a+ = 1
a = a * 1	a* = 1
a = a*(n+1)	a* = n+1
a = a/(x+y)	a/=(x+y)
a = a%b	a% = b

❖ **Increment & Decrement Operator:**

○ Java has two increment & decrement operators.

Operator	Meaning	Used
++	It adds 1 to the operand	++a or a++
--	It subtract 1 to the operand	--a or a--

❖ **Conditional Operator:**

○ The character pair? **:** is a ternary operator available in java. This operator is used to construct conditional expressions of the form Like: Expression-1 ? Expression-2 : Expression-3

❖ **Bitwise Operator:**

○ Java has a distinction of supporting special operators known as bitwise operators for manipulating of data at values of bit level.

○ These operators are used for testing the bits, or shifting them to the right or left. Bitwise operators may not be applied to float or double.

Operator	Meaning
&	Bitwise AND
!	Bitwise OR
^	Bitwise exclusive OR
~	One's complement

<<	Shift left
>>	Shift right

❖ **Special Operator :**

- Java some special operators like instance of operator and member selection operator(.)

Instance of Operator:

- The instance of is an object reference operator and returns true if the object on the left-hand side is an instance of the class given on the right – hand side.
- This operator allows us to determine whether the object to particular class or not.

Dot operator:

- The dot operator (.) is used to access the instance variable and methods of class objects.

Q. Explain Type Conversion & Casting in Java.

Type conversion & Casting

- There may be situations where there is a need to assign a value of one type to a variable of another type variable.
- If two types are compatible then java will perform the conversion automatically.
- But, if two types are incompatible than you must use a cast, which performs an explicit conversion between incompatible types.

Java Automatic Type Conversion:

- Java automatically converts variables of lower precision type to higher precision type during the evaluation of expression or during assignment. This is known as automatic type conversation or automatic promotion.
- Automatic type conversion will occur if the following tow condition are satisfied :
  1. The two variable types are compatible.
  2. The destination type is larger (higher) than the source type.
- Java also does automatic type conversion while assigning values to variables in assignment statement :

```
Destination type = source type;
```

- If precision of destination type be larger than source type, java converts lower precision value to higher precision value.

Form Type	To Convert Type
byte	Short, char, int, long, float, double
Short	Int, long, float, double
Char	Int, long, float, double
Int	long, float, double
Long	float, double
Float	Double

Casting Incompatible Types:

- There are many situation occurs that you need that conversion, when a higher precision type is to be assigned to a lower precision type.
- For Example in your program you want to assign an int value to a byte variable. So that is not automatically possible in that case you must do type casting, because a byte is smaller than int.
- This kind of conversion is called sometimes narrowing conversion. It has general form

```
Type variable = (target – type) value / variable
```

Like:

```
int x;
byte b;
b = (byte) x;
```

Q. Explain Decision Control statements in Java.

Decision Control Statements & Selection Statements

- Java supports two selection statements. If & switch
- These statements allow you to control the flow of your program's execution based upon conditions known only during run time.

If

- The if statement can be used to route program execution through two different paths.

Syntax:

```
if (condition)
{
   Statements;
}
else
{
   Statements;
}
```

Nested If:

- A nested IF is an IF statements that is the target of another IF-Else statements
- When you nest IF, the main thing to remember is that an ELSE statement always refers to the nearest IF statement that is within the same block as the ELSE and that is not already associated with an ELSE.

Syntax:

```
if (condition)
{
   if (condition)
   {
         Statements;
   }
   else
   {
         Statements;
   }
}
else
{
   Statements;
}
```

If-Else-IF Ladder:

- A common programming construct that is based upon a sequence of nested ifs is the IF-ELSE-IF ladder.
- IF statements are executed from top to down.

Syntax:

```
if (condition)
   Statement - 1;
else if (condition)
   Statement - 2;
else if (condition)
   Statement - 3;
else if (Condition)
   Statements - 4 ;
..
..
..
else
   Statement - N;
```

Switch

- The switch statement is Java's multi-way branch statement. It provides easy way to dispatch execution to different parts of your code based on the value of an expression.

Syntax:

```
switch (Expression)
   case value 1 :
         Statements;
         Break;
   case value 2 :
         Statements;
         Break;
   case value N :
         Statements;
         Break;
   default:
         Statements;
```

Q. Explain Looping Statements in Java.

Looping Control Statements:

While

- The WHILE loop is java's most fundamental looping statement. It repeats a statements of block while is controlling expression is true.

Syntax:

```
while (condition)
{
   Statements;
}
```

Do-While

- The main difference between while & do-while loop that do while loop at least execute body of the loop once time if condition is true or false.
- This is happened because condition of the loop is at the bottom of the body loop so loop will execute the first than condition is checked.

Syntax:


```
do
{
   //body of loop
} while (condition)
```

For

- The for loop is the most popular loop. It takes initialization, condition and increment or decrement of variable as arguments

Syntax:

```
for (initialization; condition; iteration)
{
   Statements;
}
```

Simple For Loop	for (i=0;i<=10;i++)
Declaring Variable within the for loop	for (int I = 0; i<=10; i++)
For Loop using the comma	for (a=1, b=10; a<b; a++, b--)
For loop partial empty parts	for (; a<10 ;)

Q. Explain various Jumping Statements in Java.

Jump Statements

- Java supports basic THREE JUMP statements. BREAK, CONTINUE & RETURN
- These Statements transfers control to the another part of the program.

BREAK:

- The Break statements has following use :
  1) It terminates the SWITCH sequence.
  2) It terminates the LOOP.

CONTINUE:

- Some time it is useful to force an early iteration of a loop. That is you might want to continue running loop, but stop processing the reminder of the code in its body for this particular iteration.

RETURN:

- The last control statement is RETURN. This statement is used to explicitly return from a method.
- The return statement immediately terminates the method in which it is executed.

Q. What is an Array? Explain in Brief.

Arrays

- An array is a group of same typed variables that are shared by a common name.
- Array of any type can be created and may have one or more dimensions.
- A specific element in array is accessed by its index.
- Arrays are useful in representing vectors, matrix, string analysis and sorting.

One – Dimensional Arrays

- A list items can be given one variable name using only one subscript and such a variable is called a one-dimensional array.
- In java, an array is created in two steps. An array variable is to be declared and then an array object is to be created using new operator by specifying the number of memory locations required.
- Declaration of one dimensional array is :

```
Data-type variable-name [<size>];
```

Example:
```
int num[10];
```
or
```
int num[];
Num = new int[10];
```

- In short, An array is a two steps process:
- One must declare a variable of the desired array type.
- One must allocate the memory using new, and assign it to the array variable.

Multi-Dimensional Arrays:

- Java supports multi-dimensional arrays. Multi-dimensional arrays are actually arrays of arrays.
- Two-Dimensional array is declared by two pairs of square brackets like [ ] [ ].

  Example: `int num1 [3][3];`
- Three-Dimensional arrays are declared by three pairs of square brackets of [ ] [ ] [ ].

  Example: `int num2 [3][3][3];`

Alternative Array Declaration:

- There is a second form that may be used to declare an array : `type [] var-name;`
- The square brackets follow the type specifier, and not the name of the array variable.

  Example: (The following two declaration are equivalent)

```
int sum [] = new int [4];
int [ ] sum = new int[4];
char matrix [ ] [ ] = new char [3] [4];
char [ ] [ ] matrix = new char [3] [4];
```

(This alternative declaration form offers convenience when declaring several arrays at the same time.)

Command line Program:

```
class CommandLine
{
	public static void main (String args[])
	{
		int l = args.length;
		for (int i=0; i<l; i++)
		{
			System.out.println("Argument [" + i + "] is : " + args[i]);
		}
	}
}
```


C:\...\>Javac Commandline.java
C:\...\>Java Commandline Basic Cobal C Java MyClass OurClass

Output:
```
Argument [0] is Basic
Argument [1] is Cobol
Argument [2] is C
Argument [3] is Java
Argument [4] is MyClass
Argument [5] is OurClass
```

JAVA 2D Array

Example:
```
public class Program
{
	public static void main(String[] args)
	{
		// Use initialize syntax.
		 int[][] values = { { 1, 2 }, { 3, 4 } };
		System.out.println(values[0][0]);
		System.out.println(values[1][0]);
		System.out.println(values[0][1]);
		System.out.println(values[1][1]);
	}
}
```
Output:
```
1
3
2
4
```

Q. What is Jagged Array? Explain in brief.

Jagged Array

It is a new feature supported by Java. In Jagged arrays, each row, in a two-dimensional array, may contain different lengths. Let us design a two-dimensional array with 4 rows where the first row contains 4 elements, the second row with 1 element, the third row with 2 elements and the fourth row with 3 elements.

Following is the graphical representation of an array.

Columns	0	1	2	3
student 0	44	55	66	77
student 1	36			
student 2	87	97		
student 3	68	78	88	

Example:

```
public class JaggedArray
{
      public static void main(String args[])
      {
            int student[][] = new int[4][];
            student[0] = new int[4];
            student[1] = new int[1];
            student[2] = new int[2];
            student[3] = new int[3];

            System.out.println("Row count: " + student.length);
            System.out.println("Third row size: : " + student[3].length);

            // 1st row
            student[0][0] = 44;
            student[0][1] = 55;
            student[0][2] = 66;
            student[0][3] = 77;

            // 2nd row
            student[1][0] = 36;

            // 3rd row
            student[2][0] = 87;
            student[2][1] = 97;

            // 4th row
            student[3][0] = 68;
            student[3][1] = 78;
            student[3][2] = 88;
```



```
            System.out.println("student[3][1] marks: " + student[3][1]);
            System.out.println("\nMatrix Form");
            for(int i = 0; i < student.length; i++)
            {
                  for(int j = 0; j < student[i].length; j++)
                  {     System.out.print(student[i][j] + "\t");    }
                  System.out.println("\n");
            }
      }
}
```

Jagged array with String values

This is like a 2D array but with irregular sub array lengths. Its lengths are not even - they are jagged. Here we create a jagged array of two String arrays.

The first sub array has two elements. The second one has three String elements. We loop over all these values.

Example:

```
public class JaggedArrayString
{
      public static void main(String[] args)
      {
            // Create an array of String arrays: a jagged array.
            String[][] values = new String[2][];

            // Fill first row with 2-element array.
            values[0] = new String[2];
            values[0][0] = "cat";
            values[0][1] = "dog";

            // Use 3-element array for second row.
            values[1] = new String[3];
            values[1][0] = "fish";
            values[1][1] = "bird";
            values[1][2] = "lizard";

            // Display rows and elements.
            for (String[] array : values)
            {
                  for (String element : array)
                  {
                        System.out.print(element);
                        System.out.print(" ");
                  }
                  System.out.println("\n");
            }
      }
}
```

Output:

```
cat dog
fish bird lizard
```

3D Array

Usually two dimensions is enough, but we can use three dimensions in an array. And even more are supported. It is rare to need a 3D array.

Example:

```
public class Program
{
      public static void main(String[] args)
      {
            // Create space cube with 9 points.
            byte[][][] space = new byte[3][3][3];
            space[0][0][0] = 10;
            space[1][1][1] = 20; // Middle of the cube.
            space[2][2][2] = 30;

            // Display points in our space-cube.
            System.out.println(space[0][0][0]);
            System.out.println(space[1][1][1]);
            System.out.println(space[2][2][2]);
      }
}
```

Output:

```
10
20
30
```

Q. What is Class? Explain in detail.

Class

- Java is a pure object oriented programming language.
- Each statement must be written in class.
- Class declared by class keyword along with a class name.
- Class name is user defined name.
- A class may contain data like instance variable, member variable, method which are used to manipulate data.

  Syntax:

```
class <class name>
{
   Data-type instance <variable – 1>;
   Data-type instance <variable – 2 >;
   Type method-name1 (parameter list)
   {
         Body of method;
   }
   Type method-name2 (parameter list)
   {
         Body of method;
   }
}
```


Declaring & Creating Object:

- Object is an instance of class.
- An object is created from a class using the new operator.

  Example: `MyClass m1;`
- The variable m1 is the object reference variable whose type is MyClass which contains null value.

  `m1 = new MyClass ();`
- The new operator creates an object of type MyClass returns the object reference.
- One can create an object in a single line statement like:

  `MyClass m1 = new MyClass();`

Q. Explain new keyword.

The New Keyword:

- An object is created from a class using new operator.
- The new operator after creation of the object returns the object reference. It can be stored in a variable.
- Variable that stores object reference is called object reference variable.
- The class name followed by parentheses specifies the constructor for the class.
- A constructor defines what occurs when an object of a class is created.
- Most of the classes explicitly define their own constructors within their class definition but if explicit constructor is specified then java will automatically supply a default constructor.

  Example: (Output of this program is 6000.00)

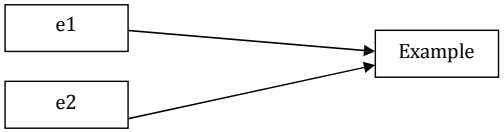
```
class Example
{  double width, height, depth;  }
class Exampledemo
{
   public static void main (String args[])
   {
         Example imp = new Example();
         double vol;
         imp.width=10;
         imp.height=20;
         imp.depth=30;
         vol = imp.width * imp.height * imp.depth;
         System.out.println("Volume is " + vol);
   }
}
```

- Each object contains its own copy of each variable defined by the class.
- Every Example object contains its own copy of the instance variables width, height, & depth.
- To access these variables, we will use the dot (.) operator.

Assigning Object Reference Variable:

```
Example e1 = new Example ();
Example e2 = e1; // assignment of e1 to e2
```

- After the execution of e1 & e2, both will refer the same object of Example.
- The assignment of e1, e2 did not allocate memory or copies any part of the original object.
- It simply makes e2 refer to the same object as e1. It means if you can make any changes through e2 it will affected the object to which e1 is referring because they are same object. Just see the figure :



Data Member

- Data members in a class can be defined into two categories :
  1) **Instance Member**
     - Each instance of the class has its own copy of variables and method in the memory of the computer.
     - These methods & variables are collectively called as instance members.
     - Every object has its own copy of all the instance variables and all the objects of a class share common implementation of the instance methods.
  2) **Static Member**
     - It belong to a class and not to any particular object unlike instance member.
     - All object of class share a common copy of static member.
     - The static keyword is used to declare static variable or methods
     - A static method can only access a static member.

Method:

- There are three types of methods :
  1) Void type – Does not return any value
  2) Return value type – Returning a value
  3) Method which takes parameter

  General Syntax:

```
type name (parameter list)
{
   Body of method;
}
```

- Here type means data return by the method (if method does not return a vale its return must be void), name means specifies the name of the method & parameter list means it is a sequence of type & identifiers pairs separated by comma.
- Method that have return type other than void return a value the calling routing using the return statement like return value;
- Parameters are variables that receive the value of the argument passes to the method when it is called. If the method has no parameters, than the parameters list will be empty.

Void type – Does not return any value

```
class Example
{
   double a,b,c;
   void volume()
   {     System.out.println("Volume is : "+a*b*c);    }
}
class ExampleDemo
{
   public static void main(String args[])
   {
         Example E1 = new Example();
         Example E2 = new Example();
         E1.a=5;
         E1.b=10;
         E1.c=15;
         E2.a=10;
         E2.b=20;
         E2.c=30;
         E1.volume();
         E2.volume();
   }
}
```

Return value type – Returning a value

```
class Example
{
   double a,b,c;
   double volume()
   {     return a*b*c;     }
}

class ExampleDemo
{
   public static void main(String args[])
   {
         Example E1 = new Example();
         Example E2 = new Example();
         double vol;
         E1.a=5;
         E1.b=10;
         E1.c=15;
         E2.a=10;
         E2.b=20;
         E2.c=30;
         vol = E1.volume();
         System.out.println("Volume is : "+vol);
         vol = E2.volume();
         System.out.println("Volume is : "+vol);
   }
}
```

Method which takes parameter

```
class Example
{
   double a,b,c;
   double volume()
```

```
    {     return a*b*c;     }
    void temp(double x, double y, double, z)
    {
          a = x;
          b = y;
          c = z;
    }
}
class ExampleDemo
{
   public static void main(String args[])
   {
          Example E1 = new Example();
          Example E2 = new Example();
          double vol;
          E1.temp(5,10,15);
          E2.temp(10,20,30);
          vol = E1.volume();
          System.out.println("Volume is : "+vol);
          vol = E2.volume();
          System.out.println("Volume is : "+vol);
   }
}
```

Q. Explain Constructor in detail.

Constructor

- A constructor allows objects to initialize themselves as soon as they are created.
- A constructor automatically called when an object is created.
- It has same name as the class in which it resides and is syntactically similar to a method.
- The constructor is automatically called immediately after the object created, before the new operator complete.
- Constructors do not have return type not even void type.
- Two types of constructors :

1) Non Parameterized Constructor

```
class Example
{
      double a,b,c;
      Example ()
      {     a=10, b=10,c=10;  }
      double volume()
      {     return a*b*c;     }
}
class ExampleDemo
{
      public static void main(String args[])
      {
            Example E1 = new Example();
            Example E2 = new Example();
            double vol;
            vol = E1.volume();
            System.out.println("Volume is : "+vol);
            vol = E2.volume();
```



```
            System.out.println("Volume is : "+vol);
      }
}
```

2) Parameterized Constructor

```
class Example
{
      double a,b,c;
      double volume()
      {     return a*b*c;     }
      Example(double x, double y, double z)
      {
            a = x;
            b = y;
            c = z;
      }
}
class ExampleDemo
{
      public static void main(String args[])
      {
            Example E1 = new Example(5,10,15);
            Example E2 = new Example(10,20,30);
            double vol;
            vol = E1.volume();
            System.out.println("Volume is : "+vol);
            vol = E2.volume();
            System.out.println("Volume is : "+vol);
      }
}
```

Q. Explain Method Overloading.

Method Overloading

- A class can contain any number of methods.
- Methods can be with parameter and without parameter.
- The parameters in a method are called type signature.
- It is possible in java to define two or more methods within the same class that share the same name, with different parameter declaration.
- In this situation the method is overloaded, and the process is referred to as method overloading.
- Overloading methods is the concept of polymorphism.

Example:

```
class Overload
{
   void demo()
   {     System.out.println("No Parameter");       }
   void demo(int a)
   {     System.out.println(Value of A : "+a);     }
   void demo(int a, int b)
   {     System.out.println("Value of A is: "+a+" & Value of B is: "+b);    }
   double demo(double a)
   {
         System.out.println("Value of A - Double Data-type : " +a);
```

```
            return a*a;
      }
   }
   class OverloadDemo
   {
      public static void main (String args[])
      {
            Overload od = new Overload();
            double result;
            od.demo();
            od.demo(10);
            od.demo(10,20);
            result = od.demo(50.25)
            System.out.println("Result of od.demo(50.25) : " + result);
      }
   }
```

Q. Explain Constructor overloading in java.

Constructor Overloading

- Constructor having the same name with different parameter list is called constructor overloading.

Example

```
class Example
{
   double a,b,c;
   double volume()
   {     return a*b*c;     }
   Example(double x, double y, double z)
   {
         a = x;
         b = y;
         c = z;
   }
   Example ()
   {     a = b = c = -1;   }
   Example (double len)
   {     a = b = c = len;  }
 }
   class ExampleDemo
   {
         public static void main(String args[])
         {
               Example E1 = new Example(5,10,15);
               Example E2 = new Example();
               Example E3 = new Example (50);
               double vol;
               vol = E1.volume();
               System.out.println("Volume is : "+vol);
               vol = E2.volume();
               System.out.println("Volume is : "+vol);
               vol = E3.volume();
               System.out.println("Volume is : "+vol);
         }
   }
```


Passing objects as a Parameter to Method

```
class Demo
{
      int a, b;
      Demo( int i, int j)
      {
            a = i;
            b = j;
      }
      boolean equals (Demo obj)
      {
            if (obj.a == a && obj.b==b)
                  return true;
            else
                  return false;
      }
}
class PassObjDemo
{
      public static void main (String args[])
      {
            Demo ob1 = new Demo (5,10);
            Demo ob2 = new Demo (5,10);
            Demo ob3 = new Demo (10,5);
            System.out.println("ob1 == ob2" + ob1.equals(ob2));
            System.out.println("ob2 == ob3" + ob2.equals(ob3));
            System.out.println("ob1 == ob3" + ob1.equals(ob3));
      }
}
```

Constructor overloading with object as a parameter

```
class Example
{
      double a,b,c;
      double volume()
      {     return a*b*c;     }
      Example (Example Obj)
      {
            a = Obj.a;
            b = Obj.b;
            c = Obj.c;
      }
      Example(double x, double y, double z)
      {
            a = x;
            b = y;
            c = z;
      }
      Example ()
      {     a = b = c = -1;   }
      Example (double len)
      {     a = b = c = len;  }
}
class OverloadParaDemo
{
      public static void main(String args[])
```

```
        {
                Example E1 = new Example(5,10,15);
                Example E2 = new Example();
                Example E3 = new Example (50);
                Example E4 = new Example (E1);
                double vol;
                vol = E1.volume();
                System.out.println("Volume is : "+vol);
                vol = E2.volume();
                System.out.println("Volume is : "+vol);
                vol = E3.volume();
                System.out.println("Volume is : "+vol);
                vol = E4.volume();
                System.out.println("Volume is : "+vol);
        }
}
```

Returning an Object

A method can return any type of data, including class type that you create.

```
class Test
{
        int a;
        Test(int i)
                a = i;
        Test TempTen()
        {
                Test temp = new Test (a+10);
                return temp;
        }
}
class RetrunobjDemo
{
        public static void main(String args[])
        {
                Test ob1 = new Test (5);
                Test ob2;
                ob2 = ob1.TempTen();
                System.out.println("ob1.a " + ob1.a);
                System.out.println("ob2.a " + ob2.a);
                ob2 = ob2.TempTen();
                System.out.println("ob2.a After Second Time : " + ob2.a);
        }
}
```

Q. Explain Command line Arguments.

Command Line Arguments

- If we want to pass information into a program when we execute it, it is possible only when we use command line arguments to main() method.
- A command line arguments is the information hat directly follows he program name of the commandline when it is executed.
- It is stored as string in the String array passed to main() method.


```
class CommandLineDemo
{
        public static void main(String args[])
        {
                for (int i=0;i<args.length;i++)
                        System.out.println("Args[" + i + " ] : " + args[i]);
        }
}
```

Q. Explain final keyword.

Final Keyword

- We can use final keyword with variable, method or class

Final Keyword with variable:

- When we use final keyword with variable contains of the variable cannot be changed further. It means it is constant for whole programs.

```
final int max = 10;
```

- Final variable behaves like a class variable i.e. only one copy of the final variable is created for the different object of the class. However outside of the class, we can't access final variable with class name. It must be accessed by objet of the class.

Final Keyword with Method:

- In some situation we does not want to override the method of super class in a subclass for this final keyword is used to the method in super class.

```
final void display()
{       ...........         }
```

Final Keyword with class

- Sometimes we want to restrict class to be inherited. This can be done by using final keyword with class declaration.

```
final class <class-name>
{       Body of the class }
```

Q. Briefly explain finalize() method.

Finalize ( ) Method

- Constructor is used to initialize the object when it is declared. However java doesn't support the concept of destructor but it supports **garbage collection** mechanism.
- It automatically de-allocates the memory resources used by the object.
- Sometimes object may have non-object resources like file descriptors or system fonts the garbage collection can't free for these resources.
- The Free, these resources we must use finalize method.
- It can be write any class. User can define any action inside finalize () method.
- This finalize() method can be put inside a class before the object is destroyed. Its run before the object goes for destruction.

- The finalize method is called whenever java is about to reclaim the space for the object.

```
finalize ()
{     Body of the finalize method.  }
```

Example:

```
protected void finalize()
{     System.out.println("object has been finalized.");     }
```

Argument Passing

- We can pass argument in two ways

Call by Reference

```
class Test
{
      int a,b;
      void cal (Test Obj)
      {
            Obj.a = a*2;
            Obj.b = b/2;
      }
      Test (int i, int j)
      {
            a = i;
            b = j;
      }
}
class CallByReferenceDemo
{
      public static void main(String args[])
      {
            Test T1 = new Test(10,20);
            System.out.println("Original A : "+T1.a+" & B :" + T1.b);
            T1.cal(T1);
            System.out.println("After Call By Value  A : "+T1.a+" & B :" + T1.b);
      }
}
```

Call by Value

```
class Test
{
      void cal (int i, int j)
      {
            i = i*2;
            j = j/2;
      }
}
class CallByValueDemo
{
      public static void main(String args[])
      {
            Test T1 = new Test();
            int a = 10, b = 20;
            System.out.println("Original A : "+a+" & B :" + b);
            T1.cal(a,b);
            System.out.println("After Call By Value  A : "+a+" & B :" + b);
      }
}
```


Note: We cannot change the value because when we create an object that time we should pass the value with help of constructor

Q. Explain Recursion.

Recursion

- Recursion is the process of defining something in terms of itself.
- Recursion is the attributes that allows a method to call itself.
- A method that calls itself is said to be recursive.

Example: 1 Write a program to find out factorial value of given number

```
class Factorial
{
      int fact (int n)
      {
            int result;
            if (n==1)
                  return 1;
            result = fact (n-1)*n;
            return result;
      }
}
class RecursionDemo
{
      Public static void main(String args[])
      {
            Factorial F = now Factorial ();
            double temp;
            temp = f.fact(5);
            System.out.println("Factorial of 5 : "+temp);
            temp = f.fact(6);
            System.out.println("Factorial of 5 : "+temp);
      }
}
```

Example: 2 Write a program to print Array elements

```
class Test
{
      int values[];
      Test(int i)
      {
            values = new int[i];
      }
      void printArray(int i)
      {
            if (i==0)
                  return ;
            else
                  printArray(i-1);
                  System.out.println("[" + (i-1) + "]" + values [i-1]);
      }
}
```

```
class RecursionDemo
{
      public static void main(String args[])
      {
            Test obj = new Test(10);
            int i;
            for (i=0;i<10;i++)
            obj.values[i]=i;
            obj.printArray(10);
      }
}
```

Q. Explain this keyword.

This Keyword

- It can be used inside any method to refer to the current object.
- It is always a reference to the object on which the method was invoked.

Example: Write a program to use this keyword

```
class Example
{
      double a = 5.5, b = 6.6, c = 7.7;
      public Example (double x, double y)
      {     this.a = x;
            this.b = y;
      }
      public Example()  {     }
      public Example (double x, double y, double z)
      {  this.a  =  x;
            this.b = y;
            this.c = z;
      }
}
class ThisDemo
{
      public static void main (String args[])
      {
            Example E1 = new Example(10,20,30);
            System.out.println("E1 - B Values is : "+E1.b );
            Example E2 = new Example(5,15);
            System.out.println("E2 - B Values is : "+E2.b );
            Example E3 = new Example()
            System.out.println("E3 - B Values is : "+E3.b );
      }
}
```

Q. Write a note on static keyword & its use.

Static Member

- Java is allowed to create a member that can be used itself, without reference to a specific instance. To create such a member, precede its declaration with the keyword **static**.
- When member is declared static, it can access before any objects of its class are created & without reference to any object.
- We can declare both methods and variables to be static.

- The most common example of a static is main () method.
- Main () is declared as static because it must be called before any objects exits.
- If we can decaled instance variable as static its automatically global variables. When object of its class are declared, no copy of a static variable is made. All class share the same static variable.
- If we declare method as static than we have several restrictions :
  - Only access static data
  - Cannot refer to this or super in any way
  - Cal calls only other static methods.

Note: When we execute programs than first all static keywords are executed than executed other things.

Example:

```
class StaticDemo
{
      static int a = 50, b = 25;
      static void temp()
      {
            System.out.println("Value of A : " + a);
      }
}
class StaticDemofinal
{
      public static void main(String args[])
      {
            StaticDemo.temp();
            System.out.println("Value of B : " + b);
            //  System.out.println("Value of B : " + StaticDemo.b);
      }
}
```

Example :

```
class StaticDemo
{     static int x = 3;
      static int y;
      static
      {     System.out.println("We have Used Static Block");      }
      static void display(int z)
      {
            System.out.println("X : "+x);
            System.out.println("Y : "+y);
            System.out.println("Z : "+z);
      }
      public static void main(String args[])
      {     display(10);      }
}
```

Example: write a program to create static methods for addition, subtraction, multiplication & division.

```
class Staticoperation
{
      Static int add (int a, int b)
      {     return a+b; }
      Static int sub (int a, int b)
      {     return a-b; }
```

```
      Static int multi (int a, int b)
      {     return a*b; }
      Static double div (double a, double b)
      {     return a/b; }
}
class MethodDemo
{
      public static void main (string args[])
      {
            int x = add(10, 2);     // staticOperation.add(10,2);
            int y = sub(50, 10);    // staticMethodDemo(50,10);
            int z = multi (5, 10);
            double total = div (25, 5);
            System.out.println(x);
            System.out.println(y);
            System.out.println(z);
            System.out.println(total);
      }
}
```

Access Modifiers / Controls

- Encapsulation provides Access Control. Though encapsulation, one can control what part of a program can access the member of a class, by controlling access, one can prevent misuse.
- Java provide a set of access **Modifiers** are as below :

Public

- When a member of a class is modified by the public **Modifiers**, then that member can be accessed by any other code.

Private

- When a member of a class is modified by the public private, then that member can only be accessed by other members of its class.

Protected

- Protected applies only when inheritance is involved. If you want to allow an element to be seen outside of your current package, but only to clauses that subclass your class directly, than declared that element protected.

Default

- When a member does not have an explicit access specification, it is visible to subclasses as well as to other cases in the same package.

- When no access **Modifiers** is used, then by default the member of a class is public within its own package, but cannot be accessed outside of its package (Default).

Example:

```
class Test
{
   int a;
   public int b;
   private int c;
   void Data (int i)
```



```
   {     c = i;      }
   int Data1()
   {     return c;   }
}
class Demo
{
   public static void main (String args[])
   {
         Test T1 = new Test();
         T1.a=10;
         T1.b=20;
         T1.c=30;
         System.out.println("A, B & C :" + T1.a + " " + T1.b + " " + T1.Data1();
   }
 }
```

Q. Explain Transient & Volatile keyword.

Volatile & Transient

- Java defines two types of modifiers Transient & Volatile. These modifiers are used to handle specialized situations. When an instant variable declared as transient, then its value need not when an object is stored.

```
class Demo
{
         transient int i ;       // will not saved
         int j;            // it saved
}
```

- The volatile modifiers tell the compiler that the variable modified by volatile can be changed unexpectedly by the other parts of your program. This situation involves multithread programs.

Q. Explain abstract method.

Abstract Method

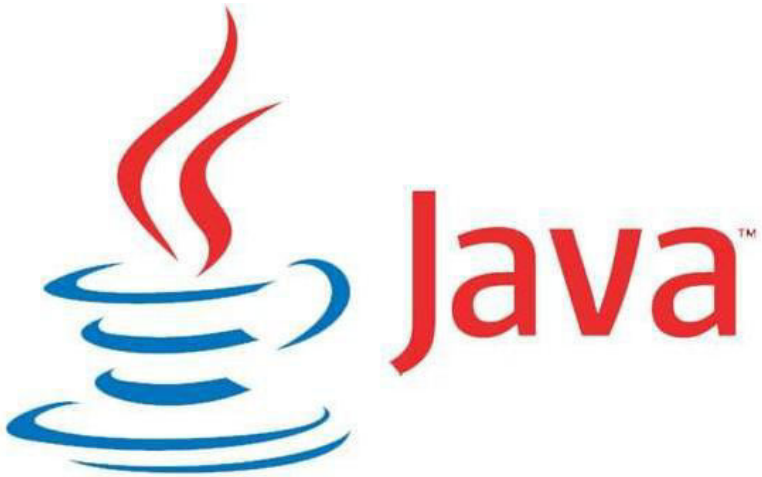
- We know that when we use final keyword with method that method cannot be overwritten same way of we use final keyword with class that class cannot be inherited.
- Sometimes we want to do exactly opposite to above things. That means the method of the super class must be override in the subclass and the class must be inherited. This can be one by using abstract keyword.
- Whenever we are declaring any method as abstract the class of that method class must be declared as abstract.
- When a class is declared as an abstract we cannot create object of that class.

Q. Explain Garbage collection.

Garbage collection

- Since objects are dynamically allocated by using the new operator, you might be wondering how such objects are destroyed and their memory released for later reallocation. In some languages, such as C++, dynamically allocated objects must be manually released by use of a delete operator.

- Java takes a different approach; it handles de-allocation for you automatically. The technique that accomplishes this is called garbage collection. It works like this: when no references to an object to an object exist, that object is assumed to be no longer needed, and the memory occupied by the object can be reclaimed. There is no explicit need to destroy objects as in C++. Garbage collection only occurs sporadically (if at all) during the execution of your program. It will not occur simply because one or more objects exist that are no longer used. Furthermore, different java run-time implementations will take varying approaches to garbage collection, but for the most part, you should not have to think about it while writing your programs.



CS-19 Programming with JAVA

B.C.A. Semester – 4

Unit-2

Inheritance & Use of Java Packages

Gaurav K Sardhara
Mo:- 9067351366

1) What is inheritance? How many types of inheritance supported in java?

Inheritance is the property of object oriented programming that supports code reusability features. The mechanism of deriving new class from the old class is called inheritance.

There are following types of inheritance supported in Java.

1) Single level
2) Multilevel
3) Hierarchical

2) What is the role of super keyword in terms of inheritance?

A subclass constructor is used to construct the instance variable of both subclass & the super class.

· Subclass constructor use the keyword **super** to invoke the constructor of the super class.

· The super keyword must be written in the subclass constructor.

3) What is Method overriding?

When we create one class from another class by using inheritance we would like to define the method of super class in the subclass. This concept is called method overriding.

· In another words we are hiding the method of super class in subclass by redefining it.

4) Define Interface.

Interfaces are similar to abstract classes, but differ in their functionality.

· Interface defines methods without body.

5) What is inner class?

In a nested class situation the class which is inside the other class in known as inner class.

6) What is nested of class?

If a class is defining within another class such type of situation is called nested class.

7) What is abstract class?

An abstract class is a class which contains only declaration of methods or data members, which will be override in sub class.

8) Define Package.

A package is a namespace that organizes a set of related classes and interfaces.

9) List-out various core packages of Java language.

The following are the core packages of java

- Java.lang
- Java.util
- Java.io
- Java.net
- Java.applet

- Java.awt
- Javax.swing
- Java.sql

10) List-out various wrapper classes of java.

The following are the wrapper classes available in java.

- Boolean
- Character
- Double
- Integer
- Float
- Long

11) What is wrapper class?

When there is a requirement to represent primitive data type values in the form of object then wrapper class is used. Wrapper class provides many additional methods to perform various operation with primitive values.

12) How String & StringBuffer classes are different?

Both the class represent the String i.e. sequence of characters. But when object of String class is created then it doesn't allow to add more characters afterwards, while you can modify or add characters in to StringBuffer.

13) _____Class is used to generate values of any type like int, float, and double randomly.

Random class of java.util package

14) Calendar class represents to which package?

Java.util package

15) What is Synchronization in Java?

Synchronization in java is the capability of control the access of multiple threads to any shared resource.

Java Synchronization is better option where we want to allow only one thread to access the shared resource.

16) How HashMap class and HashTable class differ from each other?

HashMap class is not synchronized and thread-safe for element access between multiple threads. While HashTable works fine than HashMap.

17) What StringTokenizer class does?

StringTokenizer class breaks the string from given or specified token.

Q. What is Inheritance? Explain in detail.

Inheritance

- Inheritance is the most important property of any object oriented programming language. Java support the concept of Inheritance
- It is always good to reuse the code instead of writing the same code again and again.
- This can be achieved by creating a new class form the existing class. The mechanism of deriving new class from the old class is called inheritance.
- The old class is called base class or parent class or super class while new class is called derived class or child class or sub class.
- The inheritance allows sub class to inherit all the variables & methods of their parent class.
- The keyword extends is used for inherit variables & methods of super class to the subclass. The subclass will now contain variables & methods of its own as well as of the super-class.

Different form of Inheritance is available

1) **Single inheritance (Only One Super Class)**
2) **Multilevel Inheritance (Derived from derived class)**
3) **Hierarchical Inheritance (One Super Class Many Sub Class)**
4) **Hybrid Inheritance (Mixing of more than one inheritance)**
5) **Multiple inheritance (Several Super class)**

   Note: Java doesn't support multiple Inheritance directly however we can achieve multiple inheritance in Java by Interface.

Syntax:

```
class <sub class name> extends <super class name>
{
      Variable declaration;
      Method declaration;
      Body of class;
}
```

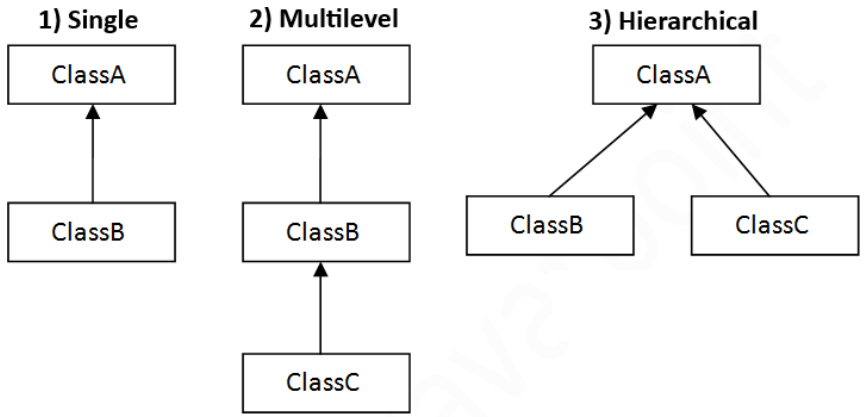
Understanding the simple example of inheritance

```
class Employee
{
      float salary = 40000;
}
class Programmer extends Employee
{
      int bonus = 10000;
      public static void main(String args[])
      {
            Programmer p = new Programmer();
            System.out.println("Programmer salary is:" + p.salary);
            System.out.println("Bonus of Programmer is:" + p.bonus);
      }
}
```


Q. Explain various types of Inheritance in Java.

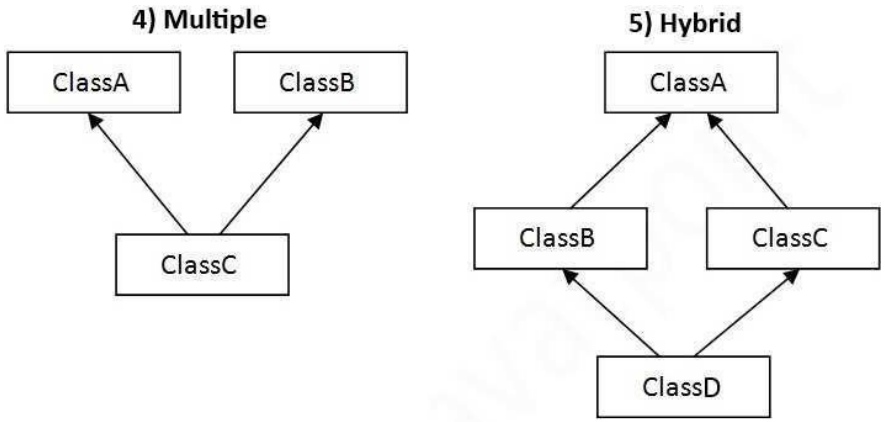
Types of inheritance in java

- On the basis of class, there can be three types of inheritance in java: single, multilevel and hierarchical.
- In java programming, multiple and hybrid inheritance is supported through interface only. We will learn about interfaces later.



Note: Multiple inheritance is not supported in java through class.

When a class extends multiple classes i.e. known as multiple inheritance. For Example:



1) **Single Inheritance**

   Single inheritance is easy to understand. When a class extends another one class only then we can call it a single inheritance. The above flow diagram shows that class B extends only one class which is A. Here A is a parent class of B and B would be a child class of A.

Example:

```
class A
{
```

```
        int a, b;
        void displayab()
        {     System.out.println("A & B : " + a + " " + b);    }
}
class B extends A
{
        int c;
        void display()
        {     System.out.println("C : " + c);      }
        void sum()
        {     System.out.println ("A+B+C : " + (a+b+c));       }
}
class simple-Inheritance
{
        public static void main (String args[])
        {
                A Superob = new A();
                B Subob = new B();
                Superob.a=5;
                Superob.b=10;
                System,out.println("concept of Superob : ");
                Superob.displayab();
                System.out.println();
                Subob.a = 15;
                Subob.b = 20;
                Subob.c = 30;
                System.out.println("Concept of Sub Object : ");
                Subob.display();
                Subob.sum();
        }
}
```

2) Multilevel Inheritance

Multilevel inheritance refers to a mechanism in OO technology where one can inherit from a derived class, thereby making this derived class the base class for the new class. As you can see in above flow diagram, C is subclass or child class of B and B is a child class of A.

Example:

```
class Car
{
        public Car()
        {     System.out.println("Class Car");     }
        public void vehicleType()
        {     System.out.println("Vehicle Type: Car");   }
}
class Maruti extends Car
{
        public Maruti()
        {     System.out.println("Class Maruti");        }
        public void brand()
        {     System.out.println("Brand: Maruti");       }
        public void speed()
        {     System.out.println("Max: 90Kmph");         }
}
public class Maruti800 extends Maruti
{
        public Maruti800()
        {     System.out.println("Maruti Model: 800");   }
        public void speed()
```



```
        {     System.out.println("Max: 80Kmph");         }

        public static void main(String args[])
        {
                Maruti800 obj=new Maruti800();
                obj.vehicleType();
                obj.brand();
                obj.speed();
        }
}
```

Output:

```
        Class Car
        Class Maruti
        Maruti Model: 800
        Vehicle Type: Car
        Brand: Maruti
        Max: 80Kmph
```

3) Multiple Inheritance

"Multiple Inheritance" refers to the concept of one class extending (Or inherits) more than one base class. The inheritance we learnt earlier had the concept of one base class or parent. The problem with "multiple inheritance" is that the derived class will have to manage the dependency on two base classes.

- ✓ **Note 1:** Multiple Inheritance is very rarely used in software projects. Using **Multiple** inheritance often leads to problems in the hierarchy. This results in unwanted complexity when further extending the class.
- ✓ **Note 2:** Most of the new OO languages like **Small Talk, Java, C# do not support Multiple inheritance**. Multiple Inheritance is supported in C++.

Q) Why multiple inheritance is not supported in java?

- ✓ To reduce the complexity and simplify the language, multiple inheritance is not supported in java.
- ✓ Consider a scenario where A, B and C are three classes. The C class inherits A and B classes. If A and B classes have same method and you call it from child class object, there will be ambiguity to call method of A or B class.
- ✓ Since compile time errors are better than runtime errors, java renders compile time error if you inherit 2 classes. So whether you have same method or different, there will be compile time error now.

Example:

```
class A
{
        void msg()
        {     System.out.println("Hello");         }
}
class B
{
        void msg()
        {     System.out.println("Welcome");       }
}
class C extends A,B
{
        //suppose if it were
        public Static void main(String args[])
        {
                C obj = new C();
```

```
            obj.msg();  //Now which msg() method would be invoked?
      }
}
```

Output: `Compile Time Error`

4) Hierarchical Inheritance

In such kind of inheritance one class is inherited by many sub classes. In below example class B, C and D inherits the same class A. A is parent class (or base class) of B, C & D.

Example:

```
class A
{
      public void methodA()
      {     System.out.println("method of Class A");  }
}
class B extends A
{
      public void methodB()
      {     System.out.println("method of Class B");  }
}
class C extends A
{
      public void methodC()
      {     System.out.println("method of Class C");  }
}
class D extends A
{
      public void methodD()
      {     System.out.println("method of Class D");  }
}
class MyClass
{
      public void methodB()
      {     System.out.println("method of Class B");  }

      public static void main(String args[])
      {
            B obj1 = new B();
            C obj2 = new C();
            D obj3 = new D();
            obj1.methodA();
            obj2.methodA();
            obj3.methodA();
      }
}
```

Output:

```
      method of Class A
      method of Class A
      method of Class A
```

5) Hybrid Inheritance

In simple terms you can say that Hybrid inheritance is a combination of **Single** and **Multiple inheritance.** A typical flow diagram would look like above. A hybrid inheritance can be achieved in the java in a same way as multiple inheritance can be!! Using interfaces. Yes you heard it right. By using **interfaces** you can have multiple as well as **hybrid inheritance** in Java.

Member Access & Inheritance / Constructor of Sub class

```
class Example
{
      double a,b,c;
      private double d; // if we declare private than we can't use that variable
with inheritance of new class.
      Example ()
      {     a = b = c = -1;   }
      Example (double len)
      {     a = b = c = len;  }
      Example (double x, double y, double z)
      {     a = x;      b = y;      c = z;      }
      Example (Example Obj)
      {     a = Obj.a;  b = Obj.b;  c = Obj.c;  }
      double volume()
      {     return a*b*c;     }
}
class Temp extends Example
{
      double t;
      Temp (double x, double y, double z, double p)
      {     a = x;      b = y;      c = z;      t = p;      }
}
class ConstructorofSubclass
{
      public static void main(String args[])
      {
            Example E1 = new Example(5,10,15);
            Example E2 = new Example();
            Example E3 = new Example (50);
            Example E4 = new Example (E1);
            double vol;
            vol = E1.volume();
            System.out.println("Volume is : "+vol);
            vol = E2.volume();
            System.out.println("Volume is : "+vol);
            vol = E3.volume();
            System.out.println("Volume is : "+vol);
            vol = E4.volume();
            System.out.println("Volume is : "+vol);
            Temp T1 = new Temp (2,4,6,8);
            T1.volume();
            System.out.println("Volume is - Extend Class :" + vol);
      }
}
```

Q. How super keyword is used in inheritance.

Inheritance with the use of the keyword - Super

- Super has two general forms
  - It calls the super class constructor
  - It is used to access a member of the super class that has been hidden by member of sub class.
- A subclass constructor is used to construct the instance variable of both subclass & the super class.
- Subclass constructor use the keyword **super** to invoke the constructor of the super class.

- The super keyword must be written in the subclass constructor.
- The call to super class constructor must appear as the first statement within the subclass constructor.
- The parameter in the super class must match the type & order of the instance variable defines in the super class.

Example

```
class Bank
{
      int accno;
      double bal;
      Bank (int a, double b)
      {
            accno = a;
            bal = b;
      }
      void display()
      {
            System.out.println (accno);
            System.out.println (bal);
      }
}
class BankInt extends Bank
{
      double rate;
      BankInt(int a, double b, double r)
      {
            super (a, b);
            rate = r;
      }
      void interest ()
      {
            double I = bal * rate * 0.10
            bal = bal + I;
      }
}
class IntDemo
{
      public static void main (String Args[])
      {
            BankInt B1 = new BankInt(11, 10000, 10)
            B1.interest();
            B1.display();
      }
}
```

Example: Write a program to create a student class with rollno, create mark class from student class with marks of three subjects. Create sports & method for calculating total & display data from marks class.

```
class student
{     int rno;
      student (int r)
      {     rno = r;    }
}
class marks extends student
{     int m1, m2 m3;
      marks (int r, int a, int b, int c)
```



```
      {     super (r);
            m1 = a;
            m2 = b;
            m3 = c;
      }
      void display()
      {     System.out.println (rno);     }
}
class sports extends marks
{     int p;
      sports (int r, int a, int b, int c, int q)
      {     super (r, a, b, c)
            p = q;
      }
      int total ()
      {     int sum;
            sum = m1+m2+m3+p;
            return sum;
      }
}
class Multilevel
{
      public static void main (String args[])
      {     Sports S1 = new sports (1, 70, 80, 90, 70);
            S1.display();
            System.out.println (S1.total());
      }
}
```

Visibility Modifier:

- It is possible to inherit all the members of a class by a subclass using extends keywords.
- We know that the variables & methods of a class are visible everywhere in the program. However sometimes we do not want this. We want to put restriction on the access of variable & methods of the class.
- This can be done by using visibility modifier with instance variable or method.
- Visibility modifier is also known as Access Modifiers or Access Control. Java provides three keywords for this purpose they are “Public”, “Private” & “Protected”. If none of the above keyword is specified than the default access is friendly access.
- When we use friendly access we can use instance variable & method of the class. Anywhere in the same package but it is not available is other package.

Access Location	Default (Friendly)	Public	Protected	Private	Private-Protected
Same Class	Yes	Yes	Yes	Yes	Yes
Sub Class in Same Package	Yes	Yes	Yes	No	Yes
Other Class in Same Package	Yes	Yes	Yes	No	No
Sub Class in Other Package	No	Yes	Yes	No	Yes
Other Class in Other Package	No	Yes	No	No	No

Public:

- When we want any variable or method available to all the class in same package or in other package we can use Public visibility modifier.
- When we use this modifier a variable & method is available everywhere means no security is achieved.

Protected:

- When we use protected visibility modifier the access is between public & friendly. i. e. the variable & method visible not only to all classes & sub-classes in the same package but also to the sub classes in other package.

Private:

- When we use Private visibility modifier a variable & method we give the highest degree of protection.
- When we use private visibility modifier with any instance variable & method they are only available in the class in which it is declared. They cannot be inherited by subclass.

Private -Protected:

- A field (instance variable as a member) can be declared with two keywords private & protected. This gives visibility in between protected & private.
- The fields declare as private protected are available in the class in which it is declared and in the sub class in same package and in other package. However it is not available in the other class in same package & in other package.

Q. Explain Method Overriding in java.

Method Overriding

- When we create one class from another class by using inheritance we would like to define the method of super class in the subclass. This concept is called method overriding.
- In another words we are hiding the method of super class in subclass by redefining it.
- When we use method overriding the subclass & super class both must contain method with same name, same type of argument, same no of arguments & same return data type.

Example

```
class Super
{
      int x;
      Super (int x)
      {     this.x = x; }
      void display ()
      {     System.out.println (x); }
}
class Sub extends Super
{
      int y;
      sub (int x, int y)
      {
            super (x);
```



```
            this.y = y;
      }
      void display ()
      {
            System.out.println (x);
            System.out.println (y);
      }
}
class OverideDemo
{
      public static void main (string args[])
      {
            Super S1 = new Super (10);
            S1.display ();
            Sub S2 = new Sub (100, 200)
            S2.display ();
      }
}
```

Q. Difference: Method Overloading & Method Overriding

Method Overloading	Method Overriding
It will work, if in program has only one class	It will work, if in program has inheritance means at least two class are necessary
It has same name but different no. of arguments & types of arguments	It has same name, same no of arguments & same type of arguments.

Q. What is an Interface? Explain in detail.

Interface

- Interfaces are similar to abstract classes, but differ in their functionality.
- Interface defines methods without body.
- Interfaces are syntactically similar to classes, but they lack instance variables and their methods are declare without body.
- Interface can contain final variables which must be initialized with values.
- One class can implements any number of interfaces.
- If we are implementing an interface in a class we must implement all the methods defined in the interface as well as a class can also implement its own methods, by providing the interface Keyword.
- Java allows you to fully utilize the "One interface multiple methods" aspect of polymorphism.

Syntax:

```
Access_Specifier interface <interface_name>
{
      Access_specifier Data_Type <method_name> (parameter List);
      Final Data_Type Variable = Value;
}
```

Here,

Access_Specifier: is either public or not used (means no access specifier then default access result and interface is only available to other members of the same package, if we declare as public than interface can be used by any other code)

Interface_Name: Name of the Interface, it can be any valid identifier.

Example:

```
interface Temp
{
      final int x = 50, y = 60;
      double avg (int x, int y);
}
```

Example:

```
interface AreaCalculation
{
      final double pi = 3.14;
      double AreaCalculation (double r);
}
class Circle implements AreaCalculation
{
      public double AreaCalculation(double x)
      {
            double ar;
            ar = pi * x * x;
            return ar;
      }
      void display()
      {     System.out.println ("Area Calculation");   }
}
class InterfaceDemo
{
      public static void main (String args[])
      {
            double area;
            AreaCalculation ac = new Circle();
            area = ac.AreaCalculation(10);
            System.out.println("Area For Circle is " + area);
      }
}
```

Interface can be extended

Example:

```
interface SuperA
{
      void disp1();
      void disp2();
}
interface SubB extends SuperA
{     void disp3();     }
class ShowDemo implements SubB
{
      public void disp1()
      {     System.out.println ("Display 1");   }
      public void disp2()
      {     System.out.println ("Display 2");   }
      public void disp3()
      {     System.out.println ("Display 3");   }
}
```



```
class InterfaceExtendsDemo
  {
        public static void main (String Args[])
        {
              ShowDemo sd = new ShowDemo();
              sd.disp1();
              sd.disp2();
              sd.disp3();
        }
  }
```

Q. Explain Hybrid inheritance implementation using interfaces.

Example:

```
interface A
{     public void methodA();  }
interface B extends A
{     public void methodB();  }
interface C extends A
{     public void methodC();  }

class D implements B, C
{
      public void methodA()
      {     System.out.println("MethodA");      }
      public void methodB()
      {     System.out.println("MethodB");      }
      public void methodC()
      {     System.out.println("MethodC");      }
      public static void main(String args[])
      {
            D obj1= new D();
            obj1.methodA();
            obj1.methodB();
            obj1.methodC();
      }
}
```

Native

- Java is a dynamic language; it has the capability of dynamically linking in new class libraries, methods and objects. Java program also supports functions written in other languages such as C & C++. These functions are known as native methods. Native methods are linked dynamically.
- **Kindly visit the example:**
  http://www.ict.nsc.ru/win/docs/java/tutorial/native1.1/stepbystep/index.html

Q. Explain Nested & Inner classes in Java.

Nested & Inner Class

- If a class is defining within another class such type of situation is called nested class.
- The scope of nested class is bounded by the scope of its enclosing class like { }
- There are two types of nested classes are ;

Syntax:

```
class OuterClass
{
      ...
      class NestedClass
      {     ...   }
}
```

```
class OuterClass
{
      ...
      static class StaticNestedClass
      {     ...   }
      class InnerClass
      {     ...   }
}
```

Static nested class

- A static nested class is one which has the static modifier applied because it is static.
- If we want to access the member of class with the help of an object. Means it cannot refer to members of its enclosing class directly.
- For that reason static nested classes are rarely used.

Non Static nested class (Inner / Outer Class)

- An inner class is a non-static nested class.
- It has access to all of the variables and methods of its outer class and may refer to them directly in the same way that other non-static member of the outer class.

Example

```
class Outerclass
{
      int out_a = 100;
      void temp()
      {
            Innerclass inner = new Innerclass();
            inner.display();
      }
      void temp1()
      {
            System.out.println("The Value of Inner Class B : "+inner_b);
            // Error for B variable.
      }
      class Innerclass
      {
            int inner_b = 50; // Local Variable of Inner-Class
            void display()
            {
                  System.out.println("The Value of Outer Class A : "+out_a);
            }
      }
}
class NestedDemo
{
      public static void main(String args[])
      {
            Outerclass outer = new Outerclass();
            Outer.temp();
      }
}
```


Q. Explain the use of final keyword with reference to method & class.

Final Keyword with Method & Class

- The keyword Final has three uses.
  1) It can used to create constant variable.
     `final int a = 10;`
  2) Second, it can used to prevent overriding.
     If we declare method as final as it cannot be overridden.
  3) Third, it can used prevent Inheritance.
     If we declare class as final as it cannot be Inheritance.

Example

```
final class A
{
      final void display ()
      {     System.out.println ("This is Final Method");     }
}
class B extends A
{
      void display ()
      {
            System.out.println ("This is Not Possible");
            // Error - cannot override method
      }
}
```

Q. Explain abstract keyword for class & method

Abstract Class & Method

- There are situations in which you want to define a super class that declares the structure of a given abstraction without providing a complete implementation of every method.
- That is, sometimes you want to create a super class that only defines generalized form that will be shared by all of its sub classes, leaving it to each subclass to fill in the detail.
- You can require that certain methods be overridden by sub classes by specifying the abstract type modifier.
- A subclass must override an abstract method.
- To declare an abstract method, use the following form:

  abstract type name (parameter list);

  You can see that there is no method body is present.

- Any class that contains one or more abstract methods must also be declared as abstract. To declare a class as abstract, use the **abstract** keyword at the beginning of the class declaration.

- There can be no objects of an abstract class. That is, an abstract class can't be directly instantiated with the **new** operator.
- You can't declare abstract constructors or abstract static method.

- Any subclass of an abstract class must either implement all of the abstract methods in the super class or be itself declared as abstract.

Example

```
abstract class Shape
{
      double dim1, dim2;
      Shape (double a, double b)
      {
            dim1 = a;
            dim2 = b;
      }
      abstract double area();
}

class Rectangle extends Shape
{
      Rectangle (double a, double b)
      {     super(a, b);      }

      double area ()
      {
            System.out.println ("Inside Are of Rectangle is : ");
            return dim1 * dim2;
      }
}
class Triangle extends Shape
{
      Triangle (double a, double b)
      {
            super (a, b);
      }
      double area ()
      {
            System.out.println ("Inside Are of Triangle is : ");
            return dim1 * dim2 / 2;
      }
}
class AbstractDemo
{
      public static void main (String args[])
      {
            // This is not possible because you have used abstract class
            Shape S = new Shape(10,10);
            Rectangle R = new Rectangle (10,10);
            Triangle T = new Triangle (10,10);
            Shape S1;
            S1 = R;
            System.out.println ("Area is " + S1.area());
            S1 = T;
            System.out.println ("Area is " + S1.area());
      }
}
```

Q. What is Package? Explain in detail.

Java Packages

A package is a namespace that organizes a set of related classes and interfaces. Conceptually you can think of packages as being similar to different folders on your computer. You might keep HTML pages in one folder, images in another, and scripts or applications in yet another. Because software written in the Java

programming language can be composed of hundreds or thousands of individual classes, it makes sense to keep things organized by placing related classes and interfaces into packages.

- A **java package** is a group of similar types of classes, interfaces and sub-packages.
- The most important property of OOP language is versatility of code.
  - One way to achieve the reusability is by extending classes & implementing interfaces but we cannot do this in other program.
  - Second way of achieving reusability is to use package.
- Package in java is similar to class libraries in other programming language.
- Packages are java's way of grouping a variety of class and interfaces together.
- The grouping is done according to functionality.
- The benefits of using packages are given below:
  1) The classes contain in the packages of other programs can be easily reused.
  2) In packages classes can be unique to compare with classes in other packages. I.e. two class with same name can be stored in two different packages.
  3) Packages provide a way to hide classes thus altering other program & packages for accessing.
  4) Package also provides a way for separating design & code.
- Package in java can be categorized in two form, built-in package and user-defined package.
- There are many built-in packages such as java, lang, awt, javax, swing, net, io, util, sql etc.
- For any application java API packages can be used for design purpose and user defined packages can be used for coding purpose.

Defining a Package

To group classes into a package each class must have a package statement defined at the top of it's "***.java**" file. It lets the compiler know which package the class belongs to and must be the first line of code. For example, imagine you're making a simple Battleships game. It makes sense to put all the classes needed in a package called battleships:

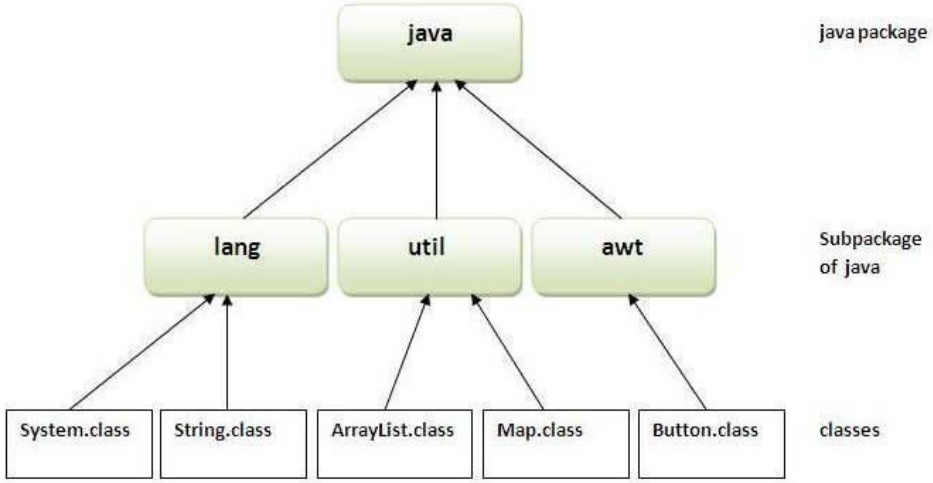
```
package battleships
class GameBoard{}
```

Every class with the above package statement at the top will now be part of the Battleships package.

Typically packages are stored in a corresponding directory on the file system but it is possible to store them in a database. The directory on the file system must have the same name as the package. It's where all the classes belonging to that package are stored. For example, if the battleships package contains the classes GameBoard, Ship, ClientGUI then there will be files called GameBoard.java, Ship.java and ClientGUI.java stored in a directory call battleships.

Advantage of Java Package

1) Java package is used to categorize the classes and interfaces so that they can be easily maintained.
2) Java package provides access protection.
3) Java package removes naming collision.



Q. Explain User-Defined Package.

User Define Packages

- JAVA API provides a large number of classes grouped into different packages according to functionality beside this we can create our own packages.
- We must first declare the name of the package using "**package**" keyword then we can define a class normally.
- The package declaration statement must be first statement in java source code file.

Syntax:

```
package <package name>
<access modifier> class <class name>
{
      Body of class
}
```

Example:

```
package mypack;
public class First
{
      System.out.println("First Class");
}
```

Save file as: `First.java (Save in mypack folder, i.e.:d:\java\mypack\First.java)`

- It creates class "**First**" in package "**mypack**", we must save this file with **First.java** in the sub directory mypack folder.
- Now we can use this class in other package from parent directory of the package by using import statement.
- We can also create hierarchy of package. This can be done by specifying multiple package name separated by dots.

Example: package firstpack.secondpack;

I.e.: d:\java\firstpack\secondpack\ABC.java

- A Java package file can have more than one class definitions in such cases only one of the classes must be declared as public and we must store the file with that class name when a source file with more than one class definition is compiling. Java creates independent class file for those classes.

```
package mypack;
public class A
{     Body of class;    }
class B
{     Body of class;    }
class C
{     Body of class;    }
```

Simple example of java package

The **package** keyword is used to create a package in java.

Save file as Simple.java

```
package mypack;
public class Simple
{
      public static void main(String args[])
      {     System.out.println("Welcome to package"); }
}
```

How to Compile & Run java package program.

Compile: `javac -d . Simple.java`

Run: `java mypack.Simple`

Output: `Welcome to package`

The -d is a switch that tells the compiler where to put the class file i.e. it represents destination. The "." represents the current folder.

Accessing Package:

- After creating a package we can access that package either by specifying a full qualified class name or using "import" statement.
- We can use import statement when there were many references to a particular package or the package name is too long.
- The general form of import statement for Searching a class is as follow:

```
import package1[.package2[.package3……classname;
```

Here "package1" is the name of top level package. "package2" is the name of package included in "package1" and so on… finally the class name is specified.

Example: import firstpackage.secondpackage.Myclass;

Note: if you want to take all classes we use ***** signature.

Example: import firstpackage.secondpackage.*;

How to access package from another package?

There are three ways to access the package from outside the package.

1. import package.*;

2. import package.classname;
3. fully qualified name.

1) Using packagename.*

- If you use "**package.***" then all the classes and interfaces of that package will be accessible but not sub-packages.
- The import keyword is used to make the classes and interface of another package accessible to the current package.

Example of package that import the packagename.*

Save file as A.java

```
package pack;
public class A
{     public void msg()
      {     System.out.println("Hello");  }
}
```

Save file as B.java

```
import pack.*;
class B
{     public static void main(String args[])
      {     A obj = new A();  obj.msg();  }
}
```

Output: `Hello`

2) Using packagename.classname

If you import package.classname then only declared class of this package will be accessible.

Example of package by import package.classname

Save file as A.java

```
package pack;
public class A
{     public void msg()
      {     System.out.println("Hello");  }
}
```

Save file as B.java

```
import pack.A;
class B
{
      public static void main(String args[])
      {     A obj = new A();  obj.msg();  }
}
```

Output: `Hello`

3) Using fully qualified name

- If you use fully qualified name then only declared class of this package will be accessible. Now there is no need to import. But you need to use fully qualified name every time when you are accessing the class orinterface.
- It is generally used when two packages have same class name e.g. java.util and java.sql packages contain Date class.

Example of package by import fully qualified name

Save file as A.java

```
package pack;
public class A
```

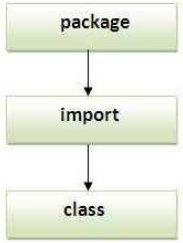


```
{     public void msg()
      {     System.out.println("Hello");  }
}
```

Save file as B.java

```
class B
{     public static void main(String args[])
      {     pack.A obj = new pack.A();    obj.msg();  }
}
```

Output: `Hello`



❖ **Note:** Sequence of the program must be package then import then class.

Package examples

- We can declare a package and class according to our requirement we must store this file in the sub directory as the name of the package after that we can access the class of that package form other program.

Save: mypack\A.java

```
package mypack;
   public class A
   {     public void disp()
         {     System.out.println ("Class A");     }
   }
```

Use class A in our program PackDemo

Save: PackDemo.java

```
import mypack.A;
   class PackDemo
   {     public static void main (String args[])
         {     A a1 = new A();   A1.disp();  }
   }
```

Program: Write a program to create two different packages with a class in each package. The class name must be different and access this two class from third program.

Save: firstpack\First.java

```
package firstpack;
   public class First
   {     public void disp()
         {     System.out.println ("Class First");       }
   }
```

Save: secondpack\Second.java

```
package secondpack;
   public class Second
   {     public void display ()
         {     System.out.println ("Class Second");      }
   }
```

Save: java\mypack\PackDemo.java

```
import firstpack.First;
import secondpack.Second;
class PackDemo
{
      public static void main (String args[])
      {
            First T1 = new First();
            Second T2 = new Second();
            T1.disp();
            T2.display();
```

```
        }
}
```

Program: Write a program to create a package with a class student and access that class for inheritance in other program.

Save: stud\Student.java

```
package stud;
public class Student
{
        int rno;
        String nam = new String();
        public void getdata (int x, String y)
        {       rno = x;    nam = y;    }
        public void disp()
        {
                System.out.println ("Roll No is : " + rno);
                System.out.println ("Name of Student is : " + nam);
        }
}
```

Save: java\result.java

```
import stud.Student;
class Result extends Student
{
        int m1, m2;
        void marks(int s1, int s2)
        {       m1 = s1;    m2 = s2;    }
        void tot()
        {
                int t = m1 + m2;
                System.out.println ("Total = " + t);
        }
}
```

Save: java\StudPackDemo.java

```
import stud.*;
class StudPackDemo
{
        public static void main(String args[])
        {
                Student st = new Student();
                Result re = new Result();
                st.getdata(1, "ABC");
                st.disp();
                re.marks(70,89);
                re.tot();
        }
}
```

Adding Class to an Existing Package

- It is easy to add new class to the exiting package

  Example: We have package p1 as follows

```
package p1;
   public class A
   {     Body of class;    }
```

- The package p1 contains 1 public class A. Now suppose we want to add another public class B to this package, this can be done as follow :

Define the class & make it public.

1) Place the package statement before class definition.

   Example:

```
package p1;
public class B
{     Body of class;    }
```

2) Store this file as "**B.java**" in sub directory **p1**
3) Compile file B.java this will create **B.class** file in directory p1.

- Now, the package p1 will contain two public class, class A & class B. This class can be used in other programs by using import statement.

  Example: `import p1.*;`

Thus, we can create a package with multiple public using following steps:

1) Decide the name of the Package
2) Create a subdirectory with the name under the directory where name source file is stored.
3) Create class in separate source file and put the package declaration statement at the top in the file.
4) Move to the subdirectory and compile all the files.

Hiding Classes

- To hide a class, declare the class without public access modifier in a package

Example:

```
package p1;
public class A
{     Body of class;    }
class C
{     Body of class;    }
```

Now, class c is not available in other program because it is not public, it is friendly.

Example:

```
import p1.*;
class Demo
{     public static void main (String args[])
      {     A a = new a();
            C c = new C();  // invalid
      }
}
```

❖ **Note:** If you import a package, sub-packages will not be imported.

If you import a package, all the classes and interface of that package will be imported excluding the classes and interfaces of the sub-packages. Hence, you need to import the sub-package as well.

Q. Explain package & sub-package concept in detail.

Sub-package in java

Package inside the package is called the sub-package. It should be created to categorize the package further.

The standard of defining package is domain.company.package e.g. com.project.bean

Example of Subpackage

```
package com.javatpoint.core;
class Simple
{     public static void main(String args[])
      {     System.out.println("Hello subpackage");   }
}
```

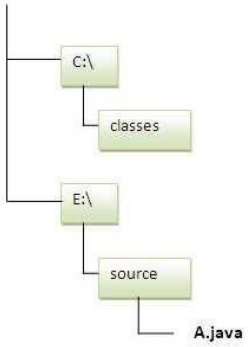
Compile: `javac -d . Simple.java`

Run: `java com.javatpoint.core.Simple`

Output: `Hello subpackage`

How to send the class file to another directory or drive?

There is a scenario, I want to put the class file of A.java source file in classes' folder of c: drive.



For example: //save as Simple.java

```
package mypack;
public class Simple
{
      public static void main(String args[])
      {     System.out.println("Welcome to package"); }
}
```

Compile: `e:\sources> javac -d c:\classes Simple.java`

Run: To run this program from e:\source directory, you need to set classpath of the directory where the class file resides.

```
e:\sources> set classpath=c:\classes;.;
e:\sources> java mypack.Simple
```

Another way to run this program by -classpath switch of java:

The -classpath switch can be used with javac and java tool.

To run this program from e:\source directory, you can use -classpath switch of java that tells where to look for class file.

For example:

```
e:\sources> java -classpath c:\classes mypack.Simple
```

Compile: `Welcome to package`

Ways to load the class files or jar files

There are two ways to load the class files temporary and permanent.

- Temporary
  - By setting the classpath in the command prompt

  - By -classpath switch
- Permanent
  - By setting the classpath in the environment variables
  - By creating the jar file, that contains all the class files, and copying the jar file in the jre/lib/ext folder.

Rule: There can be only one public class in a java source file and it must be saved by the public class name.

//save as C.java otherwise Compile Time Error

```
class A{}
class B{}
public class C{}
```

How to put two public classes in a package?

If you want to put two public classes in a package, have two java source files containing one public class, but keep the package name same.

For example:
//save as A.java

```
package javatpoint;
public class A{}
```

//save as B.java

```
package javatpoint;
public class B{}
```

Static Import

The static import feature of Java 5 facilitate the java programmer to access any static member of a class directly. There is no need to qualify it by the class name.

Advantage of static import:

- Less coding is required if you have access any static member of a class often.

Disadvantage of static import:

- If you overuse the static import feature, it makes the program unreadable and unmaintainable.

Simple Example of static import

```
import static java.lang.System.*;
class StaticImportExample
{     public static void main(String args[])
      {
            out.println("Hello");   //Now no need of System.out
            out.println("Java");
      }
}
```

Output: `Hello`
`Java`

What is the difference between import and static import?

The import allows the java programmer to access classes of a package without package qualification whereas the static import feature allows to access the static members of a class without the class

qualification. The import provides accessibility to classes and interface whereas static import provides accessibility to static members of the class.

What about package class?

The package class provides methods to get information about the specification and implementation of a package. It provides methods such as getName(), getImplementationTitle(), getImplementationVendor(), getImplementationVersion() etc.

Example of Package class: In this example, we are printing the details of java.lang package by invoking the methods of package class.

```
class PackageInfo
{
   public static void main(String args[])
   {
      Package p = Package.getPackage("java.lang");
      System.out.println("package name:" + p.getName());
      System.out.println("Specification Title:" + p.getSpecificationTitle());
      System.out.println("Specification Vendor: " + p.getSpecificationVendor());
      System.out.println("Specification Version:" + p.getSpecificationVersion());
      System.out.println("Implementation Title:" + p.getImplementationTitle());
      System.out.println("Implementation Vendor:" + p.getImplementationVendor());
      System.out.println("Implementation Version:" + p.getImplementationVersion());
      System.out.println("Is sealed:" + p.isSealed());
   }
}
```

Output:

```
package name: java.lang
Specification Title: Java Platform API Specification
Specification Vendor: Sun Microsystems, Inc.
Specification Version: 1.6
Implementation Title: Java Runtime Environment
Implementation Vendor: Sun Microsystems, Inc.
Implementation Version: 1.6.0_30
Is sealed: false
```

Java API Packages

In general, a package can contain the following kinds of types. A package allows a developer to group classes (and interfaces) together. These classes will all be related in some way – they might all have to do with a specific application or perform a specific set of tasks. The Java API is a collection of packages - for example, the javax.xml package. The javax.xml package and its sub packages contain classes to handle XML.

Core packages in Java

java.lang	basic language functionality and fundamental types
java.util	collection data structure classes
java.io	file operations
java.math	Multi precision arithmetic's
java.net	Networking operations, sockets, and DNS lookups.
java.sql	Java Database Connectivity (JDBC) to access databases

java.awt	basic hierarchy of packages for native GUI components
java.awt.event	Provides interfaces and classes for dealing with different types of events fired by AWT components.
java.applet	classes for creating an applet
javax.swing	hierarchy of packages for platform-independent rich GUI components

java.lang Package Classes

Provides classes that are fundamental to the design of the Java programming language. The most important classes are ***Object***, which is the root of the class hierarchy, and ***Class***, instances of which represent classes at run time.

Q. Explain Math Class

- For scientific & engineering calculations, a variety of mathematical functions are needed. The class ***Math*** provides commonly used mathematical functions such as sine, cosine, and square root. Java provides these functions in the Math class available in ***java.lang*** package.

```
import java.lang.*;
class demo
{
      public static void main (String args[])
      {
            double x = 4.7;
            System.out.println (“Rounded By Round = ” + Math.round(x));
            System.out.println (“Rounded By Rint = ” + Math.rint(x));
            System.out.println (“Rounded By Ceil = ” + Math.ceil(x));
            System.out.println (“Rounded By Floor = ” + Math.floor(x));
            System.out.println (“SIN Value = ” + Math.sin(x));
            System.out.println (“COS Value  = ” + Math.cos(x));
            System.out.println (“Tan Value = ” + Math.tan(x));
            System.out.println (“Square Root of = ” + Math.sqrt(x));
            System.out.println (“Log Value of = ” + Math.log(x));
            System.out.println (“Exp. Value = ” + Math.exp(x));
      }
}
```

Q. Explain Wrapper Classes

Frequently it is necessary to represent a value of primitive type as if it were an object. The wrapper classes ***Boolean***, ***Character***, ***Integer***, ***Long***, ***Float***, and ***Double*** serve this purpose. An object of type ***Double***, for example, contains a field whose type is double, representing that value in such a way that a reference to it can be stored in a variable of reference type. These classes also provide a number of methods for converting among primitive values, as well as supporting such standard methods as equals and hash Code. The ***Void*** class is a non-instantiable class that holds a reference to a ***Class*** object representing the primitive type void.

- In vector we cannot store the data of built-in data types like int, float, long, etc… so this primitive data type must be converted into object using Wrapper class contain in java.lang package.
- The following table shows the different data types and their wrapper classes.

Simple Data Type	Wrapper Classes
byte	Byte
short	Short

int	Integer
Long	Long
float	Float
double	Double
Char	Character
Boolean	Boolean

- The wrapper classes have the number of methods for handling primitive data type & object.
- The following are some of them:

Converting primitive numbers to object numbers using constructor methods

Integer val = new Integer (i);

Wrapper class object Operator Constructor int (in-built Data Type)

Float val1 = new Float (j);

Double val2 = new Double (k);

Long val3 = new Long (l);

Note: Built in => Object => Use Constructor

Converting object numbers to primitive numbers

int i = val.intValue ();

float j = val1.floatValue();

double k = val2.doubleValue();

long l = val3.longValue();

Converting numbers to string using toString ( ) method.

String S1 = Integer.toString (i);

String S2 = Float.toString (j);

String S3 = Double.toString (k);

String S4 = Long.toString (l);

Note: Primitive to String Object

Converting String object to numeric objects using the static method value of ().

Double val = Double.valueOf (str); // Converting String Object into Double Object

double k = val.doubleValue ( ); // Converting Double Object into Built in Data Type

Float val1 = val.FloatOf (str);

Integer val2 = Integer.valueOf (str);

Long val3 = Long.valueOf (str);

Converting Numeric Strings to primitive numbers using parse method.

int i = Integer.parseInt(str); // String To Primitive

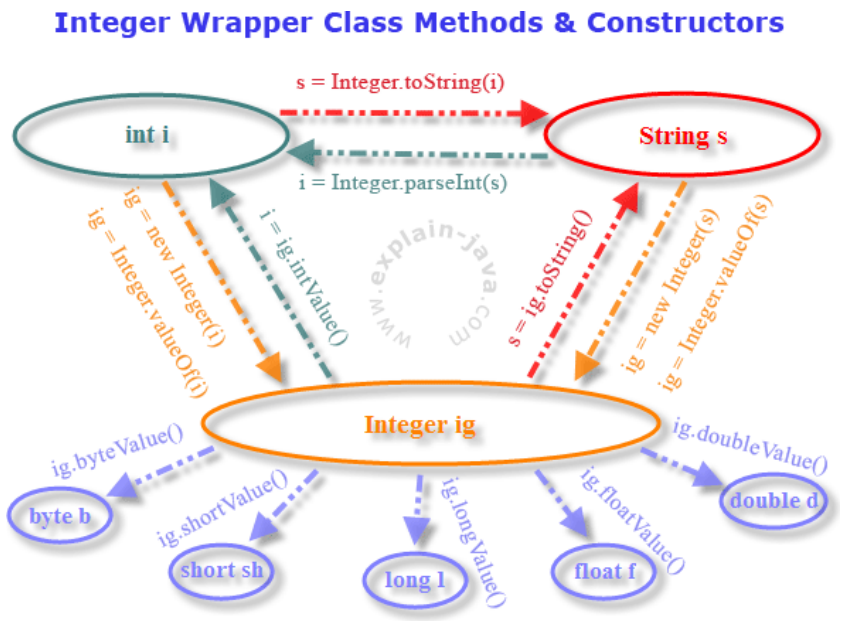
long j = Long.parseLong (str);

double k = Double.parseDouble (str);

float f = Float.parseFloat(str);

Summary:

Constructor Methods	=>	Primitive to Object
typeValue ( ) methods	=>	Object to Primitive
toString ( ) methods	=>	Primitive to String Object
valueOf ( ) methods	=>	String Object to Wrapper Classes Object
parse Method	=>	String to Primitive



Examples of Java Wrapper Classes:

To describe the application of wrapper classes in java, let us create an Integer wrapper class object with primitive int type value and get back the primitive value from the wrapper object.

Ex-1:
```
int i = 26;        // Primitive data type 'int'
// Integer Wrapper class instantiation
Integer i_Obj = new Integer (i);
// unwrapping primitive data 'int' from wrapper object
int i2 = i_Obj.intValue();
```

Ex-2:
```
String str = "S";
char ch = str.charAt(0);
System.out.println("Your 'char' value: " + ch);
Output: Your 'char' value: S
```

Why do we need wrapper classes in java?

Do we really need wrapper classes or wrapper data types in java? Can't we use primitive data types directly instead of wrapper classes? Actually we can use primitive types in many places but not all. In some cases we can't use primitive values as is, so wrapper classes are needed/required. For example, if we need to store numbers in a collection, we can't use primitives because collections such as List, Set, and Map need objects as their elements. In such cases you must use wrapper classes.

What is the difference between primitive data types and wrapper classes in java?

The main difference between primitive data types and wrapper classes (wrapper types) in java is that the primitive types can be used as raw data for operations such as arithmetic, logical, etc. and wrapper classes acts as data holders for these primitive data types.

Important note: The primitive data type values will be stored in Stack Memory whereas wrapper class objects (like any other java objects) are stored in Heap Memory.

Q. Explain String & StringBuffer Class

The classes ***String*** and ***StringBuffer*** similarly provide commonly used operations on character strings.

- A string defines as a sequence of characters.
- String manipulation is the important part of any programming language.
- The easiest way to represent a sequence of character is using character array.
- However it's easy to implement but string manipulation is not possible.
- In java string are not character array which is terminated by null character so it means string is type of object.
- Java handled string by two classes: String class & StringBuffer class.
- StringBuffer class & StringBuilder hold strings that can be modified after they are created.
- StringBuffer class & StringBuilder classes are defined in java.lang, they are available to all programs automatically.
- The difference between two classes is that the String creates strings of fixed length while StringBuffer creates strings of variable length which can be modified in terms of length & content.

1. **String Class :**

The string class support several constructors like:

String()

```
String S1 = new String();
String S1 = new String("Good Morning");
```

String (char chars[])

```
Char chars[] = {'H','E','L','L','O','.','.'};
String S2 = new String (chars);
String S3 = "MYCLASS";
```

String (char chars[], intstartIndex, intnumChars)

```
Char Chars[] = {'H','E','L','L','O','.','.'};
String S4 = new String{chars,2,3);
```

Example:

```
class StringDemo
{
   public static void main (String args[])
   {
```



```
String S1 = "Good Morning";
          Char chars[] = {'H','E','L','L','O','.','.'};
          String S2 = new String (chars);
          String S3 = "MYCLASS";
          Char Chars[] = {'H','E','L','L','O','.','.'};
          String S4 = new String{chars,2,3);
          String S5 = S1 + S2;
          System.out.println(S1+","+S2","+S3+","+S4+"","+S5);
      }
   }
```

- Different method for string class are given below :

No.	Method	Description
1.	S2 = S1.toLowerCase()	It converts the String S1 to Lower Case & Store in to S2.
2.	S2 = S1.toUpperCase()	It converts the String S1 to Upper Case & Store in to S2.
3.	S2 = S1.replace('x', 'y')	It replace all the occurrence of x with y in string S1 & store into S2
4.	S2 = S1.trim()	It removes the white spaces from the beginning & end from the String S1 & Store into S2.
5.	b=S2.equals(S1) (Here b is Boolean Data-type)	If S1 & S2 are equal than it returns True otherwise False (case Sensitive)
6.	b=S2.equalsIgnoreCase(S1) (Here b is Boolean Data-type)	It Compares S1 & S2 and return True if S1 = S2 by ignoring case (Not Case Sensitive)
7.	C = S1.CharAT(N) (Here C is Character & N is numeric value/variable)	It returns nth position character from the String S1
8.	N = S1. length()	It found the length of string S1 & return numeric value.
9.	N = S1.campareTo(S2)	It compare two string & returns negative if S1 < S2 positive if S1 > S2 Zero (0) if S1 = S2
10.	S3 = S1.concat(s2)	It concatenates the string S1 & S2 than store in String S3
11.	S2 = S1.substring(N)	It gives string starting from nth character from string S1 (Starting count from 0 - position in string)
12.	S2 = S1.substring(N,M)	It gives string starting from N^{th} character up-to M^{th} position from string S1 (Starting count from 0 - position in string)
13.	N = S1.indexOf('S')	It returns the position of first occurrence of character S in String S1.
14.	N = S1.indexOf('X', m)	It gives the position of S that occurs after m^{th} position in the string S1.
15.	str = String.ValueOf(P)	It creates a string object of the parameters P where P is of simple type of object.
16.	str = P.toString()	It creates a string representation of the object P.

2. **StringBuffer**

StringBuffer defines these Constructor:

StringBuffer ()

StringBuffer (int size)

StringBuffer (String str)

StringBuffer (CharSequence chars)

- StringBuffer class is a peer class of String.
- String creates strings of fixed length, while StringBuffer creates strings of flexible length that can be modified in terms of both length & content.
- It means String need modification are handled by StringBuffer class.
- In other way we cans say if we require a string of variable length then we can't create the object of string class because string class create an object of fixed length. If we want to create string of variable length then we can use string buffer class.
- The different methods for string manipulation are available for string buffer class object.

No.	Method	Description
1.	S1.SetcharAt(n, 'x')	It modified the nth character of 'x'
2.	S1.length()	Gives the current length of StringBuffer.
3.	S1.Capacity()	Gives the total allocated capacity (Default 16 / 32)
4.	S1.SetLength(n)	Set the length of the buffer within a String Buffer object of S1.
5.	S1.append(S2)	It will append string s2 at the end of String S1
6.	S1.insert(n, S2)	It will insert the string S2 at the position nth of the String S1
7.	S1.reverse()	Reverse the String of S1
8.	S1.deleteCharAt(n)	Delete the nth position character for String S1
9.	S1.delete(n, m)	Delete the nth position for starting to mth position of ending.

String Array:

- We can create an array of string to store more than one string in a variable. It is the concept of array of object. We can create array of string just similar to we are creating array of built in data type.

  Syntax:

  `String name[] = new string [3]` (Means name variable hold three different names.)

Example: write a program to sort given strings in ascending order

```
class stringsort
{
   static string city[]={"Madras","Calcutta","Banglore","Bombay","Delhi"};
   public static void main (String args[])
   {
         int size = city.length;
         string temp = now;
         for (int i=0; i<size; i++)
         {
               for (int j=i+1;j<size;j++)
               {
                     if (city[j].compareToCity(city[i])<0)
                     {
                           temp = city[i];   city[i]=city[j];  city[j]=temp;
                     }
               }
         }
         for (int i = 0; i<size;i++)
```



```
                     System.out.println(city[i]);
               }
   }
```

Explain String vs StringBuffer vs StringBuilder with toString() Example

Many java beginners have a confusion about what is the **difference between String, StringBuffer, and StringBuilder classes** and which class we should use out of those three. Well, I personally prefer to use StringBuilder if we need to do many string concatenations. Let's go in detail and see which one is better among the three classes with the following toString() example.

The following example java class represents employees of a company:

```
class Employee
{
      int empno;
      String ename;
      String job;
      String hireDate;
      int sal;

      public Employee(int empno, String ename, String job, String hireDate, int
sal)
      {
            this.empno = empno;
            this.ename = ename;
            this.job = job;
            this.hireDate = hireDate;
            this.sal = sal;
      }
      public String toString()
      {
            String str = " Employee [ empno=" + empno
                              + " , ename=" + ename
                              + " , job=" + job
                              + " , hireDate=" + hireDate
                              + " , sal=$" + sal
                              + " ]";
            return str;
      }
}
```

The above class uses string concatenation operator '+' which is similar to String.concat() method. Each time when you use '+' operator or String.concat() method, a new String object will be created instead of appending to the existing string object as the String class is immutable.

Now, let's discuss about StringBuffer and StringBuilder classes so that we can decide which one is preferred to use.

Difference between StringBuffer and StringBuilder

The StringBuilder class was added in Java 5 (J2SE 5.0/JDK 1.5). It has exactly the same API methods as the StringBuffer class, except StringBuilder is **not thread safe**. In other words, StringBuilder's methods are not synchronized. Although, both classes have same methods, it is strongly recommended to use StringBuilder instead of StringBuffer whenever possible because **StringBuilder runs faster**. So apart from synchronization, anything we say about StringBuilder's methods holds true for StringBuffer's methods, and vice versa.

Using StringBuilder and StringBuffer methods

The below table contains some useful methods which are available in both StringBuilder and StringBuffer classes:

Method	Description
append(String str)	Appends the specified string to this character sequence.
delete(int start, int end)	Removes the characters in a substring of this sequence.
insert(int offset, String str)	Inserts the string into this character sequence.
replace(int start, int end, String str)	Replaces the characters in a substring of this sequence with characters in the specified String.
reverse()	Causes this character sequence to be replaced by the reverse of the sequence.

All of the StringBuffer/StringBuilder methods operate on the value of the object invoking the method.

Example:

```
StringBuffer sb = new StringBuffer("Hello");
sb.append(" World");
System.out.println(" Result: " + sb);  // ==> sb.toString()
```

Note that toString() will be automatically called on any object which is concatenated with a string using '+' operator in java.

Result: `Hello World`

Here, a call to sb.append(" World") is actually appending " World" to itself (sb), hence there is no need to assign the result back to 'sb'. In fact, these method calls can be chained to each other.

Example:

```
StringBuilder sb2 = new StringBuilder("abc");
sb2.append("def").reverse().insert(3, "---");
System.out.println(" Result: " + sb2);
```

Result: `"fed---cba"`

Notice that in each of the previous two examples, there was a single call to new, i.e., we are creating only one object. So, we can use either StringBuffer or StringBuilder to concatenate multiple strings so that we can avoid creation of many unnecessary objects.

Let us take the example of toString() method from the earlier Employee class and update it using StringBuilder as follows:

```
public String toString()
{
        StringBuilder str = new StringBuilder(" Employee [ empno=").append(empno)
                        .append(" , ename=").append(ename)
                        .append(" , job=").append(job)
                        .append(" , hireDate=").append(hireDate)
                        .append(" , sal=$").append(sal)
                        .append(" ]");
        return str.toString();
}
```

Sample Result:

```
Employee [empno=526, ename=Ram, job=SSE, hireDate=29-Oct-2007, sal=$4000]
```


Here, we have created only one StringBuilder object (str) using 'new' keyword and appending different strings to the same object 'str'. Unlike in earlier code with String object, we are not creating any intermediate strings for concatenation and thus saving lots of objects, in other words, saving more memory.

java.util Package Classes

Contains the collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes (Random, Date, Calendar, GregorianCalendar, Vector, HashTable, StringTokenizer, etc…).

Q. Explain Random Class

- This class allow you to generate random int, float, double or long numbers.
- This can be very valuable if you are building a simulation of a real – world system.
- This class provides the following constructors :

  Random ( )

  Random (long start)

- The Random class provides a number of methods for the manipulations of the Random created as below :

Methods	Description
void nextBytes(byte buffer[])	Fills buffer with random numbers
int nextInt()	Returns a random Integer value.
float nextFloat()	Returns a random Float value.
double nextDouble()	Returns a random Double value.
long nextLong()	Returns a random Long value.

Example:

```
import java.util.*;
class RandomDemo
{
      public static void main (String args[])
      {
            Random temp = new Random ();
            for (int i = 0; i < 5 ; i++)
            {
                  System.out.println("Random Integer Value : " + temp.nextInt());
                  System.out.println("Random Float Value : " + temp.nextFloat());
                  System.out.println("Random Double Value : " + temp.nextDouble());
                  System.out.println("Random Long Value : " + temp.nextLong());
                  System.out.println();
            }
      }
}
```

Q. Explain Date class

Date Class

- The date class encapsulates information about a specific date & time. It provides the following constructors:

Date()

This form returns an object that represents the current date & time. Allocates a Date object and initializes it so that it represents the time at which it was allocated, measured to the nearest millisecond.

Date(int year, int month, int date)

Date(int year, int month, int date, int hrs, int min)

Date(int year, int month, int date, int hrs, int min, int sec)

Date(long date)

This form returns an object that represents the date & time milliseconds after the time. Allocates a Date object and initializes it to represent the specified number of milliseconds since the standard base time known as "the epoch", namely January 1, 1970, 00:00:00 GMT.

Date(String s)

- The Date class provides a number of methods for the manipulations of the Date created as below :

Methods	Description
boolean after (Date d)	Return True of d is after the current date. Otherwise returns false
boolean before (date d)	Return True of d is before the current date. Otherwise returns false
boolean equals (Date d)	Return True if the same value as d current date. Otherwise returns false
long getTime()	Returns the number of milliseconds since the epoch.
void setTime (long msec)	Sets the date & time of the current object to represent msec milliseconds since the epoch.
String toString()	Returns the String equivalent of the date.

Q. Explain Calendar class

Calendar Class

- The Calendar class is an abstract class that provides methods for converting between a specific instant in time and a set of calendar fields such as YEAR, MONTH, DAY_OF_MONTH, HOUR, and so on, and for manipulating the calendar fields, such as getting the date of the next week. An instant in time can be represented by a millisecond value that is an offset from the Epoch, January 1, 1970 00:00:00.000 GMT (Gregorian).
- Like other locale-sensitive classes, Calendar provides a class method, getInstance, for getting a generally useful object of this type. Calendar's ***getInstance()*** method returns a Calendar object whose calendar fields have been initialized with the current date and time:

```
Calendar rightNow = Calendar.getInstance();
```

- This class defines several int constants that are used when you get or set components of the calendar.

Example:

```
import java.util.*;
class CalendarExample
{     public static void main (String args[])
      {     Date Dt = new Date();
            System.out.println ("Current Date : " + Dt);
            Calendar cal = Calendar.getInstance ();
            System.out.println ("Date : " + cal.get(cal.DATE));
            System.out.println ("Month : " + cal.get(cal.MONTH));
            System.out.println ("Year : " + cal.get(cal.YEAR));

            System.out.println ("Hour : " + cal.get(cal.HOUR));
            System.out.println ("Minute : " + cal.get(cal.MINUTE));
            System.out.println ("Second : " + cal.get(cal.SECOND));
            System.out.println ("Millisecond : " + cal.get(cal.MILLISECOND));

            System.out.println ("Hour of Day : " + cal.get(cal.HOUR_OF_DAY));
            System.out.println ("Day of Week : " + cal.get(cal.DAY_OF_WEEK));
            System.out.println ("Day of Month : " + cal.get(cal.DAY_OF_MONTH));
            System.out.println ("Day of Year : " + cal.get(cal.DAY_OF_YEAR));
            System.out.println ("Week of Year : " + cal.get(cal.WEEK_OF_YEAR));
            System.out.println ("Day of Week in Month : " +
cal.get(cal.DAY_OF_WEEK_IN_MONTH));
      }
}
```

Q. Explain Gregorian Class

- The Gregorian calendar class is concrete subclass of Calendar. It provides the logic to manage date & time information according to the rules of the Gregorian calendar.
- This Class provides following constructors :

GregorianCalendar () This form creates an object initialized with the current date & time.

GregorianCalendar (int year, int month, int date)

GregorianCalendar (int year, int month, int date, int hour, int minute, int sec);

GregorianCalendar (int year, int month, int date, int hour, int minute);

All others forms allow you to specify how various date & time components are initialized.

For Leap Year:

boolean isLeapYear() This method returns true if the current year is a leap year otherwise it returns false.

Q. Explain Vector Class

- The Vector class is one of the most important in all of the Java class libraries.
- Programming language like C & C++ allows passing variable numbers of arguments to the function we can pass different no. of arguments to the function according to the situation.
- However java doesn't support the concept of variable arguments to a function.
- We can achieve this concept by using Vector class of the java.util package.

- The Vector class object can be used to hold any object of any class of any number
- Actually this class is used to create dynamic array.
- In static array we can't expand the size of array but in Vector means dynamic array that automatically expands as more elements are added to it.
- All Vectors are created with same initial capacity. When space is needed to accommodate more elements, the capacity is automatically increased.
- We can create Vector just like array (its need to **import java.util.*** in your program when we want to use Vector)

  Vector list = new Vector ( ); OR Vector list = new Vector (3);
- The Vector Class provides the following constructors :

  Vector ( ) This form creates a Vector with an initial capacity of ten elements.

  Vector (int n) This form creates a Vector with an initial capacity of n elements.

  Vector (int n, int m) This form creates a Vector with an initial capacity of n elements that increases by m elements each time it needs to expands
- Vector possess a number of advantages over array
  1) It is convenient to use Vector to store objects
  2) A Vector can be used to store a list of objects that may vary in size.
  3) We can add & delete objects from that list as & when required.
- The main disadvantages of Vector are that we cannot directly store simple data types in a Vector. We can only store objects in the Vector. (However this limitation can be solved by using Wrapper class)
- The Vector class provides a number of methods for the manipulations of the Vector created as below :

Methods	Description
list.addElement (item)	This method adds the specified item to the given list.
list.elementAt(n)	It gives the nth position element form the given list.
list.size()	It givens the total number of elements in the given list.
list.removeElement(item)	It will remove the specified item of the list.
list.removeElementAt(n)	It removes the item form the nth position of the given list.
list.removeAllElements()	It is used for removing all the elements from the given list
list.copyInto(array)	It will copy all the items from the list into array
list.insertElementAt(item, n)	It will add the item at the nth position in the given list

Example

```
import java.util.*;
class VectorExample
{
      public static void main (String args[])
      {
            Vector list = new Vector();
            list.addElement(new Integer(50));
            list.addElement(new Float(10.5));
            list.addElement(new Double(98.76));
            list.addElement(new String("Hello"));
            list.addElement(new Long(9000));
```



```
            System.out.println (list + "\n");

            String S1 = new String ("World");
            list.insertElementAt(S1,3);
            System.out.println (list + "\n");

            list.removeElementAt(1);
            System.out.println (list + "\n");

            list.removeElement(98.76);
            System.out.println (list);
      }
}
```

Program: Write a program to demonstrate the use of the vector class using command line argument

```
import java.util.*;
class VectCmdLine
{
      public static void main (String args[])
      {
            Vector list = new Vector();
            int len = args.length;

            System.out.print("Passed Argument : ");
            for (int i=0 ; i<len ; i++)
            {
                  list.addElement(args[i]);
                  System.out.print(args[i] + " ");
            }
            list.insertElementAt("Cobol",2);
            int size = list.size();

            String listArray[] = new String[size];
            list.copyInto(listArray);

            System.out.println("\n");

            System.out.print("After Insertion : ");
            for (int i=0 ; i<len ; i++)
            {
                  System.out.print(listArray[i] + " ");
            }
            System.out.println("\n");
      }
}
```

C:\…\> java VectDemo C C++ Fortron Pascal Basic

Program: Write a program that accepts shopping list of five items from the command line and store them in a vector & also add the following task

- To delete an item from the list.
- To add an item at the end of the list.
- To add an item of specific location in the list.
- To printthe list.

```
// DigiCam Laptop Bluetooth
import java.util.*;
```

```
class VectorManipulation
{
      public static void main(String args[])
      {
            Vector list = new Vector ();
            int len = args.length;
            for (int i=0 ; i<len ; i++)
            {
                  list.addElement (args[i]);
            }
            Display  (list);
            addatend (list, new String("Pen Drive"));
            addatpos (list, "Tablet", 3);
            remove   (list, "DigiCam");
      }
      static void display(Vector a)
      {     System.out.println(a + "\n");       }
      static void addatend(Vector B, String Str)
      {
            B.addElement(Str);
            display(B);
      }
      static void addatpos(Vector V, String Str, int p)
      {
            V.insertElementAt(Str, p);
            display(V);
      }
      static void remove(Vector V1, String Str)
      {
            V1.removeElement(Str);
            display(V1);
      }
}
```

Q. Explain HashMap Class

- A HashMap contains values based on the key. It implements the Map interface and extends AbstractMap class.
- It contains only unique elements.
- It may have one null key and multiple null values.
- Itmaintains no order.

Hierarchy of HashMap class: Map implements AbstractMap extends HashMap

Example:

```
import java.util.*;
class HashMapExample
{
      public static void main(String args[])
      {
            HashMap<Integer,String> hm = new HashMap<Integer,String>();
            hm.put(100,"Amit");
            hm.put(101,"Vijay");
            hm.put(102,"Rahul");
            for(Map.Entry m:hm.entrySet())
            {     System.out.println(m.getKey() + " " + m.getValue());   }
```



```
}
```

Q. Explain HashTable Class

- A Hashtable is an array of list. Each list is known as a bucket. The position of bucket is identified by calling the hashcode ( ) method. A Hashtable contains values based on the key. It implements the Map interface and extends Dictionary class.
- It contains only unique elements. It may have / not have any null key or value.
- Hashtable was part of the original java.util and is a concrete implementation of a Dictionary.
- However, Java 2 re-engineered Hashtable so that it also implements the Map interface. Thus, Hashtable is now integrated into the collections framework. It is similar to HashMap, but it is synchronized.
- Like HashMap, Hashtable stores key/value pairs in a hash table. When using a Hashtable, you specify an object that is used as a key, and the value that you want linked to that key. The key is then hashed, and the resulting hash code is used as the index at which the value is stored within the table.

The HashTable defines four constructors.

HashTable ( ) The first version is the default constructor

HashTable (int size) The second version creates a hash table that has an initial size specified by size:

HashTable (int size, float fillRatio)

The third version creates a hash table that has an initial size specified by size and a fill ratio specified by fillRatio. This ratio must be between 0.0 and 1.0, and it determines how full the hash table can be before it is resized upward.

HashTable(Map m)

The fourth version creates a hash table that is initialized with the elements in m. The capacity of the hash table is set to twice the number of elements in m. The default load factor of 0.75 is used.

Method: put()

The **put(K key, V value)** method is used to map the specified key to the specified value in this hashtable.

- **key -**This is the hashtable key.
- **value -**This is the value.

Difference between HashMap and Hashtable

HashMap and Hashtable both are used to store data in key and value form. Both are using hashing technique to store unique keys. But there are many differences between HashMap and Hashtable classes that are given below.

#	HashMap	Hashtable
1	HashMap is **non synchronized**. It is not-thread safe and can't be shared between many threads without proper synchronization code.	Hashtable is **synchronized**. It is thread-safe and can be shared with many threads.
2	HashMap **allows one null key and multiple null values**.	Hashtable **doesn't allow any null key or value**.
3	HashMap is **fast**.	Hashtable is **slow**.

4	We can make the HashMap as synchronized by calling this code Map m = Collections.synchronizedMap(hashMap);	Hashtable is internally synchronized and can't be unsynchronized.
5	Iterator in HashMap is **fail-fast**.	Enumerator in Hashtable is **not fail-fast**.
6	HashMap inherits **AbstractMap** class.	Hashtable inherits **Dictionary** class.

Q. Explain Synchronization in Java

- Synchronization in java is the capability of control the access of multiple threads to any shared resource.
- Java Synchronization is better option where we want to allow only one thread to access the shared resource.

HashTable defines the following methods:

Methods with Description
void clear( ) Resets and empties the hash table.
Object clone( ) Returns a duplicate of the invoking object.
boolean contains(Object value) Returns true if some value equal to value exists within the hash table. Returns false if the value isn't found.
boolean containsKey(Object key) Returns true if some key equal to key exists within the hash table. Returns false if the key isn't found.
boolean containsValue(Object value) Returns true if some value equal to value exists within the hash table. Returns false if the value isn't found.
Enumeration elements( ) Returns an enumeration of the values contained in the hash table.
Object get(Object key) Returns the object that contains the value associated with key. If key is not in the hash table, a null object is returned.
boolean isEmpty( ) Returns true if the hash table is empty; returns false if it contains at least one key.
Enumeration keys( ) Returns an enumeration of the keys contained in the hash table.
Object put(Object key, Object value) Inserts a key and a value into the hash table. Returns null if key isn't already in the hash table; returns the previous value associated with key if key is already in the hash table.
void rehash( ) Increases the size of the hash table and rehashes all of its keys.
Object remove(Object key) Removes key and its value. Returns the value associated with key. If key is not in the hash table, a null object is returned.
int size( ) Returns the number of entries in the hash table.
String toString( ) Returns the string equivalent of a hash table.

Ex-1:

```
import java.util.*;
class Hashtable_Simple
{
      public static void main(String args[])
      {
            Hashtable<Integer,String> hm=new Hashtable<Integer,String>();
```



```
            hm.put(100,"Shatrugn");
            hm.put(102,"Laxman");
            hm.put(101,"Bharat");
            hm.put(103,"Ram");

            for(Map.Entry m : hm.entrySet())
            {     System.out.println(m.getKey() + " " + m.getValue());   }
      }
}
```

Output:

```
103 Ram
102 Laxman
101 Bharat
100 Shartugn
```

Ex-2:

```
import java.util.*;
public class HashTableExample
{
      public static void main(String args[])
      {     // create hash table
            Hashtable htable1 = new Hashtable();

            // put values in table
            htable1.put(1, "A");
            htable1.put(2, "B");
            htable1.put(3, "C");
            htable1.put(4, "D");
            System.out.println("Initial hash table value: " + htable1);

            // put value at key 4
            String returnval = (String) htable1.put(4,"TP");
            System.out.println("Return value : " + returnval);
            System.out.println("New hash table value : " + htable1);
      }
}
```

Output:

```
Initial hash table value: {4=D, 3=C, 2=B, 1=A}
Return value: D
New hash table value: {4=TP, 3=C, 2=B, 1=A}
```

Ex-3: Hashtable & Enumeration class example

```
import java.util.*;
public class HashTableDemo
{     public static void main(String args[])
      {     // Create a hash map
            Hashtable balance = new Hashtable();
            Enumeration names;
            String str;
            int bal;

            balance.put("Ram", new Integer(3434));
            balance.put("Laxman", new Integer(123));
            balance.put("Hanuman", new Integer(1378));
            balance.put("Bharat", new Integer(99));
```

```
balance.put("Shatrugn", new Integer(-19));

            // Show all balances in hash table.
            names = balance.keys();
            while(names.hasMoreElements())
            {
                  str = (String) names.nextElement();
                  System.out.println(str + " : " + balance.get(str));
            }
            System.out.println();

            // Deposit 1,000 into Ram's account
            bal = ((Integer)balance.get("Ram")).intValue();
            balance.put("Ram", new Integer(bal + 1000));
            System.out.println("Ram's new balance : " + balance.get("Ram"));
      }
}
```

Output:

```
Laxman: 123
Shatrugn: -19
Bharat: 99
Ram: 3434
Hanuman: 1378
Ram's new balance: 4434
```

Q. Explain StringTokenizer class in detail.

- The java.util.StringTokenizer class allows you to break a string into tokens. It is simple way to break string.
- It doesn't provide the facility to differentiate numbers, quoted strings, identifiers etc. like StreamTokenizer class. We will discuss about the StreamTokenizer class in I/O chapter.

There are 3 constructors defined in the StringTokenizer class.

StringTokenizer (String str)

Creates StringTokenizer with specified string.

StringTokenizer (String str, String delim)

Creates StringTokenizer with specified string and delimiter.

StringTokenizer (String str, String delim, boolean returnValue)

Creates StringTokenizer with specified string, delimiter and returnValue. If return value is true, delimiter characters are considered to be tokens. If it is false, delimiter characters serve to separate tokens.

Public method	Description
boolean hasMoreTokens()	checks if there is more tokens available.
String nextToken()	returns the next token from the StringTokenizer object.
String nextToken(String delim)	returns the next token based on the delimeter.
boolean hasMoreElements()	same as hasMoreTokens() method.
Object nextElement()	same as nextToken() but its return type is Object.
int countTokens()	returns the total number of tokens.

Example: Let's see the simple example of StringTokenizer class that tokenizes a string "my name is khan" on the basis of whitespace.

```
import java.util.StringTokenizer;
public class StringTokenizer_Simple
{     public static void main(String args[])
      {     StringTokenizer st = new StringTokenizer("My Country is INDIA"," ");
            while(st.hasMoreTokens())
            {     System.out.println(st.nextToken()); }
      }
}
```

Output:

```
My
Country
is
INDIA
```

Example: of StringTokenizer class methods

```
import java.util.*;
public class StringTokenizer_Methods
{     public static void main(String[] args)
      {     StringTokenizer st1 = new StringTokenizer("Gujarat");
            System.out.println("Next token is : " + st1.nextToken());
            System.out.println("Count tokens is : " + st1.countTokens());
            System.out.println("-----------------------------------");
            StringTokenizer st = new StringTokenizer("My,Country,is,INDIA");
            System.out.println("Next token (,) is : " + st.nextToken(","));
            System.out.println("Has more token is : " + st.hasMoreTokens());
            System.out.println("Count tokens is : " + st.countTokens() + "\n");

            System.out.println("Next token is : " + st.nextToken());
            System.out.println("Has more elements is : " + st.hasMoreElements());
            System.out.println("Count tokens is : " + st.countTokens() + "\n");

            System.out.println("Next element is : " + st.nextElement());
            System.out.println("Count tokens is : " + st.countTokens() + "\n");

            System.out.println("Next element is : " + st.nextElement());
            System.out.println("Count tokens is : " + st.countTokens() + "\n");
      }
}
```

Hierarchy for All Packages

http://docs.oracle.com/javase/7/docs/api/overview-tree.html

CS-19 Programming with JAVA

B.C.A. Semester – 4

Unit - 3

Exception Handling, Threading & Stream I/O

Gaurav k sardhara
Mo:- 9067351366

1) What is Exception Handling?

The **exception handling in java** is one of the powerful *mechanism to handle the runtime errors* so that normal flow of the application can be maintained.

2) Define Exception.

In java, exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.

3) List-out types of Exception.

There are mainly two types of exception:

- Checked Exception
- Unchecked Exception

4) How throw, throws & Throwable keywords are used?

- The keyword throw is used to explicitly throw an exception. While throws is used to declare an exception.
- The keyword throw is used within the method. While throws is used with the method signature.
- The java.lang.Throwable class is the superclass of all errors and exceptions in the Java language. Only objects that are instances of this class (or one of its subclasses) are thrown by the Java Virtual Machine or can be thrown by the Java throw statement.

5) What is Thread?

Thread is basically a lightweight sub-process, a smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.

6) Define Multithreading.

Multithreading in java is a process of executing multiple threads simultaneously.

7) Which package provides various File handling classes?

Java.io package

8) List-out various thread life cycle stages.

- New Born state
- Runnable state
- Running state
- Blocked state
- Dead state

9) How threads can be created in Java?

There are mainly two methods to create thread in java

- By extending thread class
- By implement Runnable interface

10) List-out various types of System I/O streams in Java.

In java, 3 streams are created for us automatically. All these streams are attached with console.

1. **System.out:** standard output stream
2. **System.in:** standard input stream
3. **System.err:** standard error stream

11) What is Stream?

A stream is a sequence of data. In Java a stream is composed of bytes. It's called a stream because it's like a stream of water that continues to flow.

12) List-out various types of I/O streams in java.

There are two types of stream available in java,

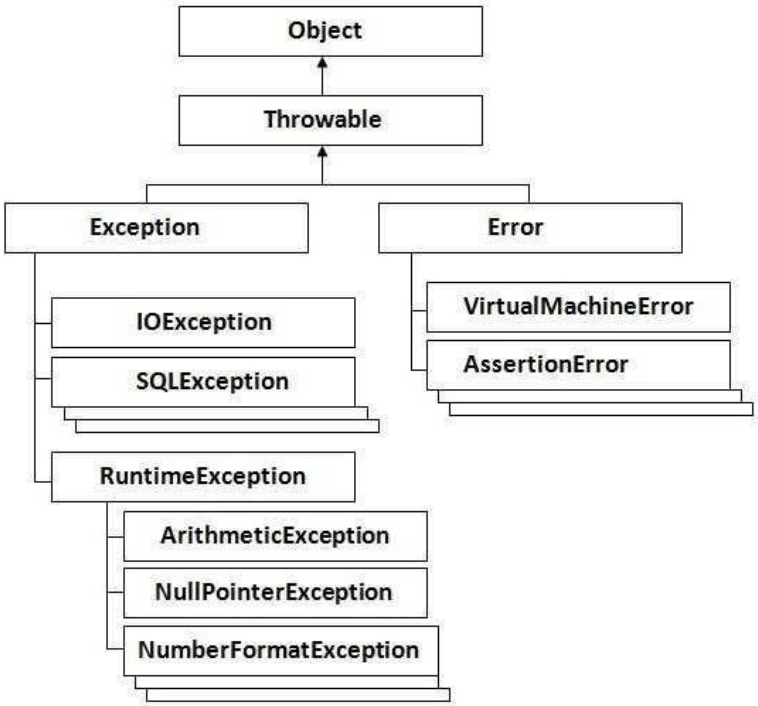
- Byte-based Stream
- Character-based Stream

13) What StreamTokenizer class does?

The StreamTokenizer class takes an input stream and parses it into "tokens", allowing the tokens to be read one at a time.

Exception Handling

- The **exception handling in java** is one of the powerful *mechanism to handle the runtime errors* so that normal flow of the application can be maintained.
- Exception Handling is a mechanism to handle runtime errors such as ClassNotFound, IO, SQL, Remote etc.
- Whenever we are writing programs by mistake we may type something wrong and this will create an error.



Q. What is exception?

Dictionary Meaning: Exception is an abnormal condition.

- In java, exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.
- An exception is a condition that is caused by a runtime error in the program.
- When the java interpreter encounters an error it creates an exception object and throws it.
- If the exception object is not caught and handled properly then interpreter will display an error message and terminate the program.
- If we want the program to continue with remaining execution of code then we should try to catch the exception and then display appropriate message for taking corrective actions. This mechanism is called Exception Handling.
- The purpose of exception handling mechanism is to provide a means to detect and report errors conditions so that appropriate actions can be taken.
- The mechanism suggest the following task :
  1. Find the error or problem (Hit the Exception)
  2. Inform that an error has occurred (Throw the Exception)
  3. Receive the error information (Catch the Exception)
  4. Take corrective Action (Handle the Exception)

Q. Explain various types of Exception in java.

There are mainly two types of exceptions: checked and unchecked where error is considered as unchecked exception. The sun microsystem says there are three types of exceptions:

1. Checked Exception
2. Unchecked Exception
3. Error

Difference between checked and unchecked exceptions

1) Checked Exception

The classes that extend Throwable class except RuntimeException and Error are known as checked exceptions e.g.IOException, SQLException etc. Checked exceptions are checked at compile-time.

2) Unchecked Exception

The classes that extend RuntimeException are known as unchecked exceptions e.g. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc. Unchecked exceptions are not checked at compile-time rather they are checked at runtime.

3) Error

Error is irrecoverable e.g. OutOfMemoryError, VirtualMachineError, AssertionError etc.

The error may be classified in two categories :

1. **Compile time Error :**

- The error which is generated by wrong syntax is known as compile time error.
- When the program is compiled, compiler will shows the error message when the compile time error is generated the compiler will not create the class file. So we have to solve the error & recompile it.
- The common way due to which this type of error occurs are missing semicolon, miss spell, keywords or variables mistake of opening & closing braces etc…

2. **Run time Error :**

- Sometimes our program may compile properly and generate class file but when we execute the program we will get wrong result or program may terminate. This is due to run-time error.
- The common mistakes for causing runtime errors are division by 0, array index out of bound, storing wrong type of data in array passing from arguments to methods etc…

- **Common scenarios where exceptions may occur**

There are given some scenarios where unchecked exceptions can occur. They are as follows:

1) Scenario where **ArithmeticException** occurs

If we divide any number by zero, then occurs an ArithmeticException.

```
int a=50/0;         //ArithmeticException
```

2) Scenario where **NullPointerException** occurs

If we have null value in any variable, performing any operation by the variable occurs a NullPointerException.

```
String s = null;
System.out.println (s.length());    //NullPointerException
```

3) Scenario where **NumberFormatException** occurs

The wrong formatting of any value, may occur NumberFormatException. Suppose I have a string variable that have characters, converting this variable into digit will occur NumberFormatException.

```
String s = "abc";
int i = Integer.parseInt (s);       //NumberFormatException
```

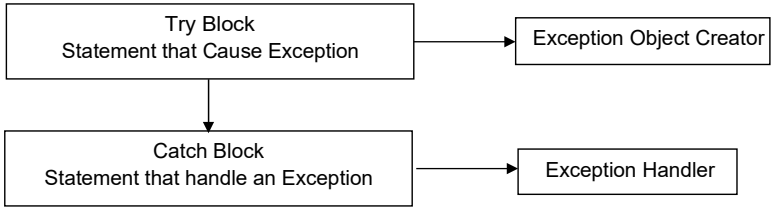
4) Scenario where **ArrayIndexOutOfBoundsException** occurs

If you are inserting any value in the wrong index, it would result ArrayIndexOutOfBoundsException as shown below:

```
int a[] = new int[5];
a[10] = 50;         //ArrayIndexOutOfBoundsException
```

Syntax of Exception Handling:

- The basic concept of Exception handling is throwing an exception & catching it. This is shown in the figure.



- Java uses the keyword try for a block of code that is user interface to cause an error condition and throws and exception.
- A catch block is defined by the keyword catch which "catches" the exception thrown by the try block.
- The catch block is added immediately after the try block. The following syntax shows this :

```
class <class name>
{     Body of class;
      try
      {     Statements;
            cause exception;
      }
      catch (exception_type object_name)
      {     Statements; }
      Statements of class;
}
```

- The try block can have one or more statements that would generate an exception.

- If one statement generates an exception the remaining statement in the block will not be executed at the execution jumps to the catch block that is placed after the try block.
- The catch block can have one or more statements that are necessary to process the exception each try block must be followed by at-least one catch block otherwise compiler will generate an error.

Different types of Built-in Exceptions in JAVA

- Java provides an exception class for handling different types of exceptions occurred during the execution of the program.
- The following list shows the different types of exception commonly occurred during program execution.

No.	Exception Type	Description
1	ArithmeticException	It is caused by math errors such as division by Zero.
2	ArrayIndexOutOfBoundsException	It is caused when pad index of array is given.
3	ArrayStoreException	It is caused when a program tries to store wrong type of data in an array.
4	FileNotFoundException	It is caused when attempt to non-existence file.
5	IOException	It is caused when a general failure in input output operation.
6	NullPointerException	It is caused by referencing null object.
7	NumberFormatException	It is caused when a conversation between strings and numbers fails.
8	OutOfMemoryException	It is caused when there is not enough memory for the allocation of the new object or array.
9	SecurityException	It is caused where an applet tries to perform and action i.e. not allowed by the browsers security.
10	StackOverFlowException	It is caused when the system runs out of stack space.
11	StringIndexOutOfBoundsException	It is caused when a program try to access a no-existence character position from a string.

Q. Explain Java Exception handling in detail.

Java exception handling is managed by using five keywords:

try

Piece of code of your program that you want to monitor for exceptions are contained within a try block. If an exception occurs within the try block it is throwing.

catch

catch block can catch the exception and handle it in some rational manner.

throw

System generated exception are automatically thrown by the java run time system. Now if we want to manually throw an exception, we have to use the ***throw*** keyword.

throws

Any exception that is thrown out of a method must be specified by a ***throws*** clause.

finally

Any code that absolutely must be executed before a method returns will be placed in a finally block.

Let us see in details…

☐ try block

- Java try block is used to enclose the code that might throw an exception. It must be used within the method.
- Java try block must be followed by either catch or finally block.

Syntax of java try-catch

```
try
{  //code that may throw exception     }
catch(Exception_class_Name ref)
{  }
```

Syntax of try-finally block

```
try
{  //code that may throw exception     }
finally
{  }
```

☐ catch block

- Java catch block is used to handle the Exception. It must be used after the try block only.
- You can use multiple catch block with a single try.

Problem without exception handling

Let's try to understand the problem if we don't use try-catch block.

```
public class Testtrycatch1
{
      public static void main(String args[])
      {
            int data=50/0;    //may throw exception
            System.out.println ("rest of the code...");
      }
}
```

Output: `Exception in thread main java.lang.ArithmeticException:/ by zero`

- As displayed in the above example, rest of the code is not executed (in such case, rest of the code... statement is not printed).
- There can be 100 lines of code after exception. So all the code after exception will not be executed.

Solution by exception handling

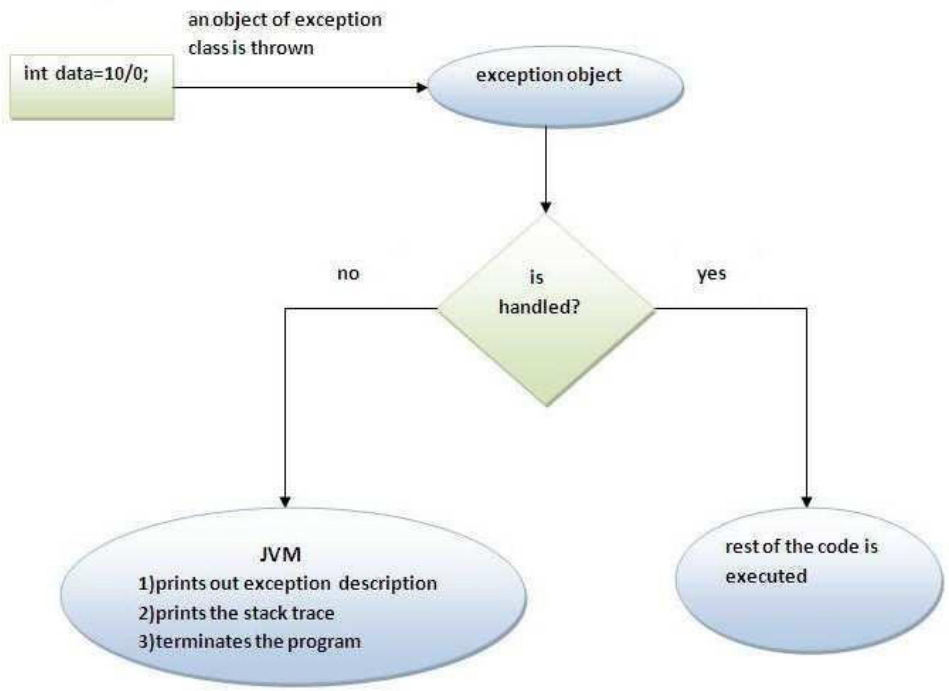
Let's see the solution of above problem by java try-catch block.

```
public class Testtrycatch2
{     public static void main(String args[])
      {     try { int data=50/0;    }
            catch (ArithmeticException e)
            {     System.out.println (e);       }
            System.out.println ("rest of the code...");
      }
}
```

Output:
```
Exception in thread main java.lang.ArithmeticException:/ by zero
rest of the code...
```

- Now, as displayed in the above example, rest of the code is executed i.e. rest of the code... statement is printed.

Internal working of java try-catch block



The JVM firstly checks whether the exception is handled or not. If exception is not handled, JVM provides a default exception handler that performs the following tasks:

- Prints out exception description.
- Prints the stack trace (Hierarchy of methods where the exception occurred).
- Causes the program to terminate.

But if exception is handled by the application programmer, normal flow of the application is maintained i.e. rest of the code is executed.

Program: Write a program to demonstrate the use of try & catch block for exception handling.

```
class TryCatchDemo1
{     public static void main (String args[])
      {     int a=10, b=5, c=5;
            int x, y;
            try
            {     x = a / (b - c);  }
            catch (ArithmeticException e)
            {     System.out.println ("Division By Zero Value " + e);    }
            y = a / (b+c);
            System.out.println ("Value of Y = " + y);
```



```
      }
}
```

Output: `Division by Zero Value` *Errors*
`Value of Y = 1`

Program: Write a program to input data from the command line argument. Display total numbers of valid and invalid data. Data is valid when it is integer number.

```
class TryCatchDemo2
{     public static void main (String args[])
      {     int valid =0, invalid = 0, No;
            for (int i = 0; i < args.length ; i++)
            {     try
                  {
                        No = Integer.parseInt(args[i]);
                        valid++;
                  }
                  catch (NumberFormatException e)
                  {     invalid++;  }
            }
            System.out.println ("Total Valid Integer Number : " + valid);
            System.out.println ("Total In-Valid Integer Number : " + invalid);
      }
}
```

Program: Write a program to input numbers through Command line argument calculate sum and average of only Integer numbers & display it.

```
class TryCatchDemo3
{     public static void main (String args[])
      {     int s = 0, c = 0;
            float avg = 0;
            for (int i=0; i < args.length ; i++)
            {     try
                  {
                        int no = Integer.parseInt(args[i]);
                        c++;        s = s + no;
                  }
                  catch (NumberFormatException e)
                  {
                        System.out.println("Only Integer Digit Operation Allowed");
                  }
            }
            avg = s /(float) c;
            System.out.println ("Total: " + s);
            System.out.println ("Average: " + avg);
      }
}
```

- **Multiple Catch Block:**

- Generally, Java virtual machine throws an exception whenever a statement has an error.
- The try block contains these types of statements and the execution is thrown and the catch block will caught this exception.
- Sometimes we might require more than one catch block for single try block.
- This is because the statement written in the try block may create more than one type of exception.
- To handle this exception we required Multiple Catch Block with single Try block.

- If you have to perform different tasks at the occurrence of different Exceptions, use java multi catch block.

Let's see a simple example of java multi-catch block.

```
public class MultipleCatchBlock
{     public static void main(String args[])
      {     try
            {     int a[]=new int[5];     a[2]=30/0;  }
            catch (ArithmeticException e)
            {     System.out.println("task 1 is completed");      }
            catch (ArrayIndexOutOfBoundsException e)
            {     System.out.println("task 2 is completed");      }
            catch (Exception e)
            {     System.out.println("common task completed");    }
            System.out.println("rest of the code...");
      }
}
```

Output:

```
task 1 is completed
rest of the code...
```

Rule: At a time only one Exception is occurred and at a time only one catch block is executed.

Rule: All catch blocks must be ordered from the most specific to most general i.e. catch for ***ArithmeticException*** must come before catch for ***Exception***.

Program: Write a program to demonstrate the use of multiple catch blocks

```
class MultiCatch2
{     public static void main (String args[])
      {     int a[] = {5, 10};
            int b = 2;
            float x;
            try
            {     x = a[2] / b - a[1];    }
            catch (ArithmeticException e)
            {     System.out.println("Division By Zero");   }
            catch (ArrayIndexOutOfBoundsException e)
            {     System.out.println("Wrong Index Number"); }
            catch (ArrayStoreException e)
            {     System.out.println("Wrong Data Store in Array");      }
      }
}
```

(Note: Don't write catch statement first, because it is a super class and all exception are the sub class of super class. So if you want to write, you must write at last.)

- **Nested try block**

The try block within a try block is known as nested try block in java.

Why use nested try block?

Sometimes a situation may arise where a part of a block may cause one error and the entire block itself may cause another error. In such cases, exception handlers have to be nested.

Syntax:

```
....
try
{     statement 1;
      statement 2;
      try
      {     statement 1;
            statement 2;
      }
      catch (Exception e)
      {     }
}
catch (Exception e)
{     }
....
```

Example:

```
class NestedTryBlock
{     public static void main(String args[])
      {     try
            {     try
                  {     System.out.println("going to divide");
                        int b =39/0;      }
                  catch (ArithmeticException e)
                  {     System.out.println(e);  }
                  try
                  {     int a[]=new int[5];           a[5]=4;     }
                  catch (ArrayIndexOutOfBoundsException e)
                  {     System.out.println(e);  }
                  System.out.println("other statement");
            }
            catch (Exception e)
            {     System.out.println("handled");      }
            System.out.println("normal flow..");
      }
}
```

Q. Explain throw exception

throw exception

- The Java throw keyword is used to explicitly throw an exception.
- We can throw either checked or unchecked exception in java by throw keyword. The throw keyword is mainly used to throw custom exception. We will see custom exceptions later.
- The syntax of java throw keyword is given below.
  - `throw exception;`
- Let's see the example of ***throw IOException***.
  - `throw new IOException("sorry device error");`

throw keyword example: In this example, we have created the validate method that takes integer value as a parameter. If the age is less than 18, we are throwing the ***ArithmeticException*** otherwise print a message welcome to vote.

```
public class TestThrow1
{     static void validate(int age)
      {     if(age < 18)
                  throw new ArithmeticException("not valid");
```

```
            else
                  System.out.println("welcome to vote");
      }
      public static void main(String args[])
      {     validate(13);
            System.out.println("rest of the code...");
      }
}
```

Output: `Exception in thread main java.lang.ArithmeticException:not valid`

Program:

```
class MyException extends Exception
{
      MyException (String msg)
      {     super (msg);      }
}
class MyThrowException
{     public static void main (String args[])
      {     int x=5, y=1000;
            try
            {
                  float z = (float) x / y;
                  if (z < 0.01)
                  {     throw new MyException ("No is too small");      }
            }
            catch (MyException e)
            {     System.out.println("My Exception is Caught [" + e + "]");    }
            finally
            {     System.out.println("I am Always Available");     }
      }
}
```

Q. Explain throws keyword

□ throws keyword

- The Java throws keyword is used to declare an exception. It gives an information to the programmer that they may occur an exception. So, it is better for the programmer to provide the exception handling code so that normal flow can be maintained.
- Exception Handling is mainly used to handle the checked exceptions. If they occurs any unchecked exception such as NullPointerException, it is the programmer's fault that he/she is not performing checkup before the code being used.

Syntax:

```
return_type method_name() throws exception_class_name
{     ...     }
```

Example: Let's see the example of java throws clause which describes that checked exceptions can be propagated by throws keyword.

```
import java.io.IOException;
class TestThrows1
{
```



```
      void m()throws IOException
      {     throw new IOException("device error");    //checked exception     }
      void n()throws IOException
      {     m();  }
      void p()
      {     try
            {     n();  }
            catch (Exception e)
            {     System.out.println("exception handled");  }
      }
      public static void main(String args[])
      {
            TestThrows1 obj = new TestThrows1();
            obj.p();
            System.out.println("normal flow...");
      }
}
```

Output:

```
exception handled
normal flow...
```

Rule: If you are calling a method that declares an exception, you must either caught or declare the exception.

<u>There are two cases:</u>

Case 1: You caught the exception i.e. handle the exception using try/catch.

In case you handle the exception, the code will be executed fine whether exception occurs during the program or not

```
import java.io.*;
class M
{
      void method()throws IOException
      {     throw new IOException("device error");    }
}
public class TestThrows2
{     public static void main(String args[])
      {     try
            {     M m = new M();    m.method();       }
            catch (Exception e)
            {     System.out.println("exception handled");  }
            System.out.println("normal flow...");
      }
}
```

Output:

```
exception handled
normal flow...
```

Case 2: You declare the exception i.e. specifying throws with the method.

A. In case you declare the exception, if exception does not occur, the code will be executed fine.

```
import java.io.*;
class M
{     void method()throws IOException
```

```
        {     System.out.println("device operation performed");     }
}
class TestThrows3
{     public static void main(String args[])throws IOException
      {     //declare exception
            M m = new M();    m.method();
            System.out.println("normal flow...");
      }
}
```

Output:

```
device operation performed
normal flow...
```

B. In case you declare the exception if exception occurs, an exception will be thrown at runtime because throws does not handle the exception.

```
import java.io.*;
class M
{     void method() throws IOException
      {     throw new IOException("device error");    }
}
class TestThrows4
{     public static void main(String args[]) throws IOException
      {     //declare exception
            M m=new M();
            m.method();
            System.out.println("normal flow...");
      }
}
```

Output:

```
Runtime Exception : device error
```

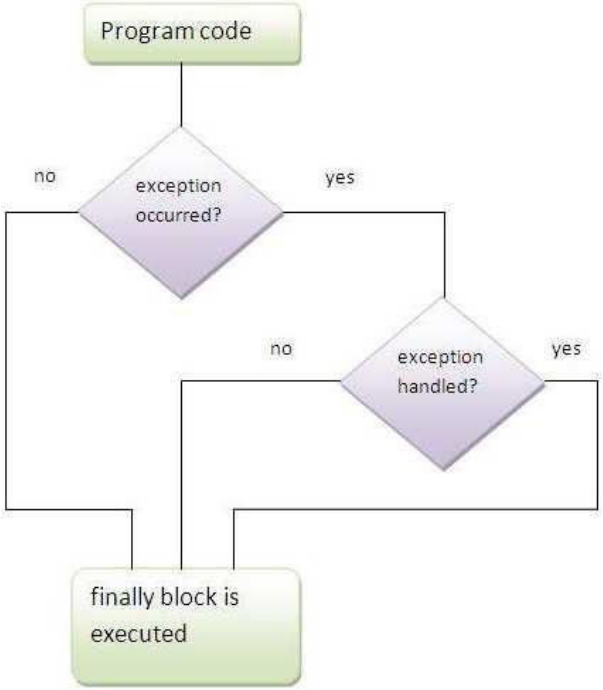
Difference between throw and throws:

#	throw keyword	throws keyword
1	throw is used to explicitly throw an exception.	throws is used to declare an exception.
2	checked exception cannot be propagated without throws.	checked exception can be propagated with throws.
3	throw is followed by an instance.	throws is followed by class.
4	throw is used within the method.	throws is used with the method signature.
5	You cannot throw multiple exception	You can declare multiple exception e.g. public void method ( ) throws IOException, SQLException.

Q. Explain finally keyword.

finally block

- Java finally block is a block that is used to execute important code such as closing connection, stream etc.
- Java finally block is always executed whether exception is handled or not.
- Java finally block must be followed by try or catch block.
- Some statements of the program we want to execute regardless of the previous statements are executed properly or not. This type of statements must be written in the finally block.



Note: If you don't handle exception, before terminating the program, JVM executes finally block (if any).

Why use java finally

Finally block in java can be used to put "cleanup" code such as closing a file, closing connection etc.

Usage of Java finally

Let's see the different cases where java finally block can be used.

Case 1: Let's see the java finally example where ***exception doesn't occur***.

```
class TestFinallyBlock
{     public static void main(String args[])
      {     try
            {     int data=25/5;    System.out.println(data);     }
```

```
            catch (NullPointerException e)
            {     System.out.println(e);  }
            finally
            {     System.out.println("finally block is always executed");     }
            System.out.println("rest of the code...");
      }
}
```

Output:

```
5
finally block is always executed
rest of the code...
```

Case 2: Let's see the java finally example where ***exception occurs and not handled***.

```
class TestFinallyBlock1
{     public static void main(String args[])
      {     try
            {     int data=25/0;    System.out.println(data);     }
            catch (NullPointerException e)
            {     System.out.println(e);  }
            finally
            {     System.out.println("finally block is always executed");     }
            System.out.println("rest of the code...");
      }
}
```

Output:

```
finally block is always executed
Exception in thread main java.lang.ArithmeticException:/ by zero
```

Case 3: Let's see the java finally example where ***exception occurs and handled***.

```
public class TestFinallyBlock2
{     public static void main(String args[])
      {     try
            {     int data=25/0;    System.out.println(data);     }
            catch (ArithmeticException e)
            {     System.out.println(e);  }
            finally
            {     System.out.println("finally block is always executed");     }
            System.out.println("rest of the code...");
      }
}
```

Output:

```
Exception in thread main java.lang.ArithmeticException:/ by zero
finally block is always executed
rest of the code...
```

Rule: For each try block there can be zero or more catch blocks, but only one finally block.

Note: The finally block will not be executed if program exits (either by calling System.exit ( ) or by causing a fatal error that causes the process to abort).

Q. Explain Throwable Class

The java.lang.Throwable class is the superclass of all errors and exceptions in the Java language. Only objects that are instances of this class (or one of its subclasses) are thrown by the Java Virtual Machine or can be thrown by the Java throw statement.

Following is the declaration for java.lang.Throwable class:

```
public class Throwable
```



```
      extends Object

      implements Serializable
```

Constructors

#	Constructor & Description
1	**Throwable()** This constructs a new throwable with null as its detail message.
2	**Throwable(String message)** This constructs a new throwable with the specified detail message.
3	**Throwable(String message, Throwable cause)** This constructs a new throwable with the specified detail message and cause.
4	**Throwable(Throwable cause)** This constructs a new throwable with the specified cause and a detail message of (cause==null ? null : cause.toString()) (which typically contains the class and detail message of cause).

Methods

#	Method & Description
1	**Throwable fillInStackTrace()** This method fills in the execution stack trace.
2	**Throwable getCause()** This method returns the cause of this throwable or null if the cause is nonexistent or unknown.
3	**String getLocalizedMessage()** This method creates a localized description of this throwable.
4	**String getMessage()** This method returns the detail message string of this throwable.
5	**StackTraceElement [ ] getStackTrace()** This method provides programmatic access to the stack trace information printed by printStackTrace().
6	**Throwable initCause(Throwable cause)** This method initializes the cause of this throwable to the specified value.
7	**void printStackTrace()** This method prints this throwable and its backtrace to the standard error stream.
8	**void printStackTrace(PrintStream s)** This method prints this throwable and its backtrace to the specified print stream.
9	**void printStackTrace(PrintWriter s)** This method prints this throwable and its backtrace to the specified print writer.
10	**void setStackTrace(StackTraceElement [ ] stackTrace)** This method sets the stack trace elements that will be returned by getStackTrace() and printed by printStackTrace() and related methods.
11	**String toString()** This method returns a short description of this throwable.

What is the use of printStackTrace () method in Java?

It prints the stack trace of the Exception to System.err.

It's a very simple, but very useful tool for diagnosing an Exception. It tells you what happened and where in the code this happened.

```
import java.lang.*;
public class ThrowableDemo
{     public static void main(String[] args)
      {     try
            {     ExceptionFunc();  }
            catch(Throwable e)
            {     // prints stacktace for this Throwable Object
```

```
                e.printStackTrace();    }
        }
        public static void ExceptionFunc() throws Throwable
        {
                Throwable t = new Throwable("This is new Exception...");
                StackTraceElement[] trace = new StackTraceElement[]
                { new StackTraceElement("ClassName","methodName","fileName",5) };

                // sets the stack trace elements
                t.setStackTrace(trace);
                throw t;
        }
}
```

Output:

```
java.lang.Throwable: This is new Exception...
at ClassName.methodName(fileName:5)
```

Q. Explain user-defined Exception class.

Creating user defined Exception class

User-defined exception class is used to show custom messages to the end users. It is used to hide the other exceptions like *NullPointerException*; *ArithmeticException* etc. User-defined exception class is so common that it will be used in every project.

Example-1: how to create user defined exception by extending Exception Class.

```
class WrongInputException extends Exception
{ WrongInputException(String s)
        {       super(s);   }
}
class Input
{       void method() throws WrongInputException
        {       throw new WrongInputException("Wrong input");   }
}
class UDExcep
{       public static void main(String[] args)
        {       try
                {       new Input().method();   }
                catch (WrongInputException wie)
                {       System.out.println(wie.getMessage());       }
        }
}
```

Example-2:

```
class MyException extends Exception
{       private int ex;
        MyException(int a)
        {       ex = a;     }
        public String toString()
        {       return "MyException[" + ex + "] is less than zero";   }
}
class UDETest
{       static void sum(int a,int b) throws MyException
        {       if(a < 0)
                {       throw new MyException(a);       }
                else
                {       System.out.println(a + b);      }
        }
        public static void main(String[] args)
```



```
        {       try
                {       sum(-10, 10);       }
                catch(MyException me)
                {       System.out.println(me);         }       }
}
```

Q Explain Multithreading in detail

Multithreading

- **Multithreading in java** is a process of executing multiple threads simultaneously.
- Thread is basically a lightweight sub-process, a smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking.
- But we use multithreading than multiprocessing because threads share a common memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process.
- Java Multithreading is mostly used in games, animation etc.
- Multithreading is a programming concept where a program (process) is divided into two or more subprograms (process) which can be executed simultaneously. Means one subprogram can display animation and other sub program is busy with inputting data.
- In most of computers there is a single process and therefore in reality the process is doing one job at a time. However processor switches between the processes so fast that it applets like processing simultaneously.
- A thread is similar to a program that has a single flow of control. It has a beginning, a body and end. Infect all the simple programs are the single threaded programs.
- Java supports multithreading i.e. multiple flow of controls can be developed in program. Each flow of control is known as Thread which runs in parallel to others.
- Create a thread in Java is very simple. Threads are implemented in the form of objects that contains a method called “Run”.
- The Run method is the heart & soul of any thread. It makes up the entire body of a thread and it is the only method in which threads behavior can be implemented.
- The run method should be invoked by an object of the appropriate thread. This can be done by initiating with the help of start method.

Advantage of Java Multithreading

1) It **doesn't block the user** because threads are independent and you can perform multiple operations at sametime.
2) You **can perform many operations together so it saves time**.
3) Threads are **independent** so it doesn't affect other threads if exception occur in a single thread.

Multitasking

Multitasking is a process of executing multiple tasks simultaneously. We use multitasking to utilize the CPU. Multitasking can be achieved by two ways:

- Process-based Multitasking (Multiprocessing)
- Thread-based Multitasking (Multithreading)

1) Process-based Multitasking (Multiprocessing)
   - Each process has its own address in memory i.e. each process allocates separate memory area.
   - Process is heavyweight.
   - Cost of communication between the processes is high.
   - Switching from one process to another require some time for saving and loading registers, memory maps, updating lists etc.

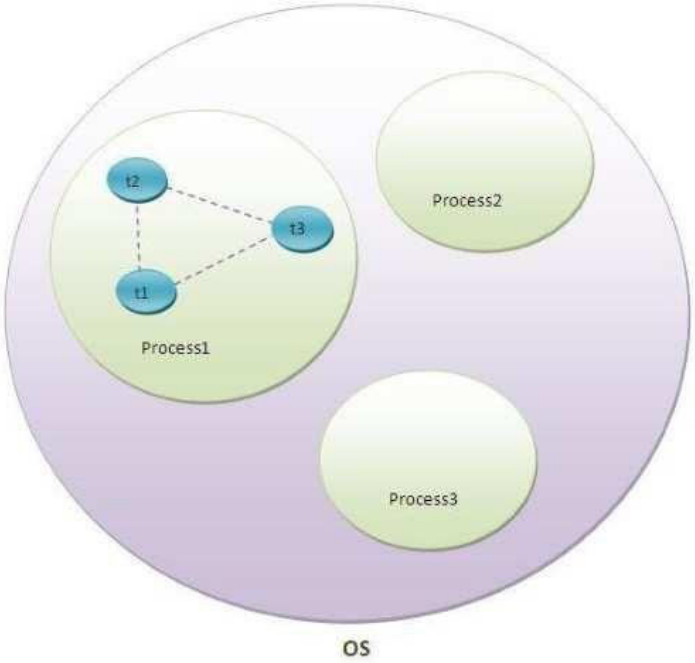
2) Thread-based Multitasking (Multithreading)
   - Threads share the same address space.
   - Thread is lightweight.
   - Cost of communication between the thread is low.

Note: At least one process is required for each thread.

What is Thread?

A thread is a lightweight sub process, a smallest unit of processing. It is a separate path of execution.

Threads are independent, if there occurs exception in one thread, it doesn't affect other threads. It shares a common memory area.



As shown in the above figure, thread is executed inside the process. There is context-switching between the threads. There can be multiple processes inside the OS and one process can have multiple threads.

Note: At a time one thread is executed only.

Q How to create thread in Java?

By creating a Thread class / extending the Thread Class

- We can make our class Runnable as a thread by extending the class Thread. This gives us to access all the methods of the thread class. It includes the following steps :

1. **Declaring the Class**

   Declare the class as extending the Thread class.

```
class Try extends Thread
{
      Body of class;
      Statements;
}
```

2. **Implementing the Run Method :**

   Implement the run method i.e. responsible for executing the sequence of code that the thread will execute.

   The run method is inherited by the class Try. We have to overwrite this method in class Try. This can be done as follow:

```
public void run()
{     Body of run Method;     }
```

3. **Staring a Thread :**

   Create a thread object and call the start method to execute the thread.

   To start a thread we must create object of the class Try & then we can call start method using this object.

```
Try T1 = new Try();
T1.start();
```

Example:

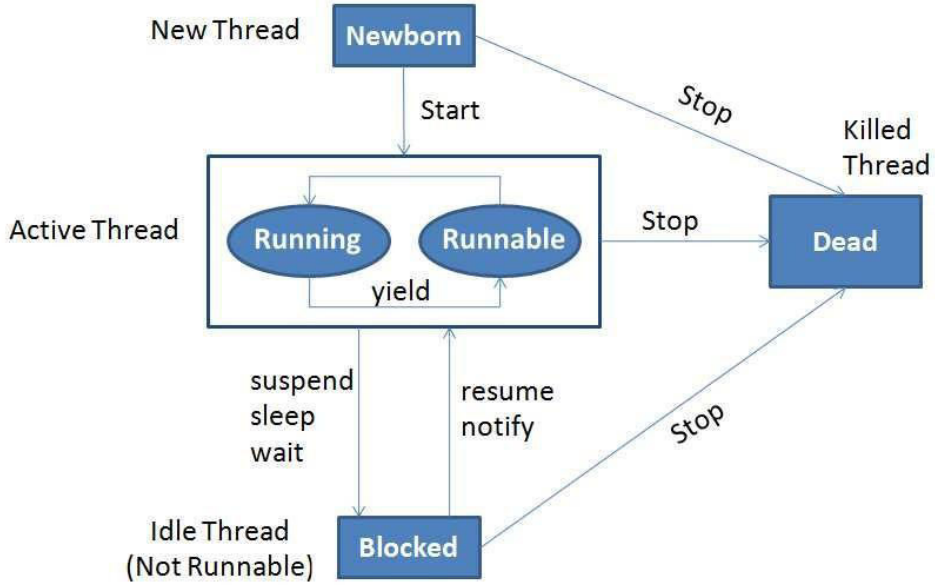
```
class A extends Thread
{
      public void run ()
      {
            for (int i=1 ; i<=5 ; i++)
                  System.out.println ("\t From A : " + i);
            System.out.println ("Exit from A");
      }
}
class B extends Thread
{
      public void run ()
      {
            for (int j=1 ; j<=5 ; j++)
                  System.out.println ("\t From B : " + j);
            System.out.println ("Exit from B");
      }
}
```

```
class ThreadTest
{
      public static void main (String args[])
      {
            A T1 = new A();
            B T2 = new B();
            T1.start();
            T2.start();
      }
}
```

Q. Explain Life-cycle of Thread.

Life cycle of a Thread (Thread States)

- A thread can be in one of the five states. According to sun, there is only 4 states in thread life cycle in java new, runnable, non-runnable and terminated. There is no running state.
- But for better understanding the threads, we are explaining it in the 5 states.
- The life cycle of the thread in java is controlled by JVM. The java thread states are as follows:
  1. New Born State
  2. Runnable State
  3. Running State
  4. Blocked State (Non-Runnable)
  5. Dead State (Terminated)



State transition diagram of a thread

1) New Born State:

- When we create a thread object the thread is born and it is in the New Born state. The thread is not yet schedule for running at this state we can do one of the following :
  a. Scheduled it for running using **start ( )** method.
  b. Kill it using **stop ( )** method.

2) Runnable State

- The Runnable state means that the Thread is ready for execution but it is waiting for the availability of the processor. I.e. The thread has join the queue of threads that are waiting for execution. The processor will be allocated to this thread in first come first serve (FCFS) manners, if all has same priority.

3) Running State

- Running state means the processor has given its time to the thread for its execution.
- The thread runs until the processor is allocated to the next thread in the queue.
- However running thread can give processor to the next thread in the queue using **yield ( )** method.
- We can move thread to the Blocked state using following three methods :
  - ✓ **Suspend ( ) method:**
    - We can use **suspend ( )** method when we does not known where to move From Runnable state to Blocked state.
    - We can use **resume ( )** method when we want to move Blocked state to Runnable state.
  - ✓ **Sleep ( ) method:**
    - If we know the specified time period for Blocked state we can use sleep method. We can specify time in millisecond as Parameter to this method.
    - After that time thread will move automatically from Blocked state to Runnable state.
  - ✓ **Wait ( ) method:**
    - When we want to blocked a thread until certain condition occur then we can use wait method. The thread can be scheduled to run using notify ( ) method.

4) Blocked State:

- A thread is said to be blocked when it is prevented from entering into the Runnable state. This happens where the thread is suspend, sleeping or waiting.
- A Blocked thread is considered as not Runnable thread but not dead. It is qualified to run again.

5) Dead State:

- Every thread has lifecycle a Running thread ends its life when it has completed executing its run method it is a natural death.
- However we can kill the thread by using stop method.

Constructors:

- Thread ( )

- Thread (String name)
- Thread (Runnable r)
- Thread (Runnable r, String name)

Commonly used methods of Thread class:

1. **public void run():** is used to perform action for a thread.
2. **public void start():** starts the execution of the thread. JVM calls the run () method on the thread.
3. **public void sleep(long milliseconds):** Causes the currently executing thread to sleep (temporarily stops the execution) for the specified number of milliseconds.
4. **public void join():** waits for a thread to die.
5. **public void join(long milliseconds):** waits for a thread to die for the specified milliseconds.
6. **public int getPriority():** returns the priority of the thread.
7. **public int setPriority(int priority):** changes the priority of the thread.
8. **public String getName():** returns the name of the thread.
9. **public void setName(String name):** changes the name of the thread.
10. **public Thread currentThread():** returns the reference of currently executing thread.
11. **public int getId():** returns the id of the thread.
12. **public Thread.State getState():** returns the state of the thread.
13. **public boolean isAlive():** tests if the thread is alive.
14. **public void yield():** causes the currently executing thread object to temporarily pause and allow other threads to execute.
15. **public void suspend():** is used to suspend the thread(deprecated).
16. **public void resume():** is used to resume the suspended thread(deprecated).
17. **public void stop():** is used to stop the thread(deprecated).
18. **public boolean isDaemon():** tests if the thread is a daemon thread.
19. **public void setDaemon(boolean b):** marks the thread as daemon or user thread.
20. **public void interrupt():** interrupts the thread.
21. **public boolean isInterrupted():** tests if the thread has been interrupted.
22. **public static boolean interrupted():** tests if the current thread has been interrupted.

Program: Write a program to demonstrate the use of yield (), stop (), sleep () methods

```
class A extends Thread
{     public void run ()
      {     for (int i=1 ; i<=5 ; i++)
            {     System.out.println ("Thread A : " + i);
                  if(i == 1)  {     yield ();   }
            }
            System.out.println ("Exit from A");
      }
}
class B extends Thread
{     public void run ()
      {     for (int j=1 ; j<=5 ; j++)
            {     System.out.println ("Thread B : " + j);
                  if(j == 2)  {     stop();     }
```



```
            }
            System.out.println ("Exit from B");
      }
}
class C extends Thread
{     public void run ()
      {     for (int k=1 ; k<=5 ; k++)
            {     System.out.println ("Thread C : " + k);
                  if(k == 1)
                  {     try   {     sleep (1000);     }
                        catch (Exception e) {   }
                  }
            }
            System.out.println ("Exit from C");
      }
}
class ThreadDemo
{
      public static void main (String args[])
      {
            A T1 = new A();
            B T2 = new B();
            C T3 = new C();
            System.out.println("Thread A Starts");
            T1.start();
            System.out.println("Thread B Starts");
            T2.start();
            System.out.println("Thread C Starts");
            T3.start();
            System.out.println ("Exit from Main Method");
      }
}
```

Q. How to create thread?

There are two ways to create a thread:

1. By extending Thread class
2. By implementing Runnable interface.

By converting a class to a thread.

- Define a class that implements Runnable interface and write code for thread in Run method.
- The approach to be used depends on what the class we are creating required if it requires extends another class than use Runnable interface otherwise Thread class.

1. **By creating a Thread class**

   Define a class that extends the thread class and overwrite its run method.

   Thread class provide constructors and methods to create and perform operations on a thread. Thread class extends Object class and implements Runnable interface.

```
class MyThread_Class extends Thread
{
      public void run()
      {
            System.out.println("thread is running...");
      }
```

```
        public static void main(String args[])
        {
                MyThread_Class t1 = new MyThread_Class ();
                t1.start();
        }
}
```

Output: `thread is running...`

Who makes your class object as thread object?

Thread class constructor allocates a new thread object. When you create object of MyThread_Class class, your class constructor is invoked (provided by Compiler) from where Thread class constructor is invoked (by super() as first statement). So your MyThread_Class class object is thread object now.

2. **By implementing the Runnable interface:**

- The Runnable interface should be implemented by any class whose instances are intended to be executed by a thread. Runnable interface have only one method named **run ( )**.

  public void run(): is used to perform action for a thread

Starting a thread:

start () method of Thread class is used to start a newly created thread. It performs following tasks:

- A new thread starts (with new call stack).
- The thread moves from New Born state to the Runnable state.
- When the thread gets a chance to execute its target, run () method will run

- We can use Runnable interface to create a thread by implementing it.
- The Runnable interface has method which must be overwriting in our class to write the body of thread.
- To do this we must perform the following steps :
  1. Declare the class as implementing the Runnable interface.
  2. Override the run method of Runnable interface.
  3. Create a Thread by defining an object. I.e. instantiated from this Runnable class as the target of the Thread.

Call the threads start () method to run the Thread.

```
class MyThread_Interface implements Runnable
{
        public void run()
        {
                System.out.println("thread is running...");
        }
        public static void main(String args[])
        {
                MyThread_Interface m1 = new MyThread_Interface();
                Thread t1 = new Thread (m1);
                t1.start();
        }
}
```

Output: `thread is running...`

If you are not extending the Thread class, your class object would not be treated as a thread object. So you need to explicitly create Thread class object. We are passing the object of your class that implements Runnable so that your class run () method may execute.

Program: Write a program to demonstrate the use of Runnable Interface

```
class X implements Runnable
{ public void run ()
        {
                for (int j=1 ; j<=10 ; j++)
                        System.out.println ("Thread X : " + j);
                System.out.println ("Exit From X");
        }
}
class Y implements Runnable
{       public void run ()
        {
                for (int k=1 ; k<=5 ; k++)
                        System.out.println ("Thread Y : " + k);
                System.out.println ("Exit From Y");
        }
}
class RunnableDemo
{
        public static void main (String args[])
        {
                X R1 = new X();
                Thread threadX = new Thread(R1);
                threadX.start();

                Y R2 = new Y();
//              Thread threadY = new Thread(R2);
//              threadY.start();
                new Thread (new Y()).start();

                System.out.println("Exit From Main");
        }
}
```

- In main method first we have created an instance of X and then pass this instance initial value of the object threadX of Thread class and then we have called the start method using the object threadX.
- If direct reference to the Thread threadY is not required then we can use a shortcut as follows :

  `new Thread (new Y()).start();`

Difference between Thread and Runnable interface in Java

Thread vs. Runnable in Java

1) Java doesn't support multiple inheritance, which means you can only extend one class in Java so once you extended Thread class you lost your chance and cannot extend or inherit another class in Java.
2) In Object oriented programming extending a class generally means adding new functionality, modifying or improving behaviors. If we are not making any modification on Thread than use Runnable interface instead.

3) Runnable interface represent a Task which can be executed by either plain Thread or Executors or any other means. So logical separation of Task as Runnable than Thread is good design decision.

4) Separating task as Runnable means we can reuse the task and also has liberty to execute it from different means. Since you cannot restart a Thread once it completes. Again Runnable vs. Thread for task, Runnable is winner.

5) Java designer recognizes this and that's why Executors accept Runnable as Task and they have worker thread which executes those task.

6) Inheriting all Thread methods are additional overhead just for representing a Task which can be done easily with Runnable.

Example: Use of Thread class & Runnable Interface for thread execution.

```
public class MyThread_both
{     public static void main (String[] args)
      {     Runner r = new Runner();
            Thread t1 = new Thread(r, "Thread A");
            Thread t2 = new Thread(r, "Thread B");
            Thread s1 = new Strider("Thread C");
            Thread s2 = new Strider("Thread D");
            t1.start();
            t2.start();
            s1.start();
            s2.start();
      }
}
class Runner implements Runnable
{     private int counter;
      public void run()
      {     try
            {     for (int i=1 ; i<=2; i++)
                  {
                        System.out.println(Thread.currentThread().getName() + " ["
+ i + "]: " + counter++ +"\n");
                        Thread.sleep(1000);
                  }
            }
            catch(InterruptedException e)
            {     e.printStackTrace();    }
      }
}
class Strider extends Thread
{     private int counter;
      Strider(String name)
      {     super(name);      }
      public void run()
      {     try
            {     for (int i=1 ; i<=2 ; i++)
                  {
                        System.out.println(Thread.currentThread().getName() + " ["
+ i + "]: " + counter++ +"\n");
                        Thread.sleep(1000);
                  }
            }
            catch(InterruptedException e)
            {     e.printStackTrace();    }
      }
```



```
}
```

The output result is

```
Thread A [1]: 0
Thread D [1]: 0
Thread B [1]: 1
Thread C [1]: 0
Thread C [2]: 1
Thread A [2]: 3
Thread D [2]: 1
Thread B [2]: 2
```

Q. Explain Thread priority.

Thread Priority:

- In Java each thread is assigned a priority which affects the order in which it is scheduled for running. If we do not give any priority to the Thread then all the threads have same priority and these threads are given equal treatment.
- Java permits us to set the priority of a thread using **setPriority ( )** method. We can use this method as follows

  ```
  Threadname.SetPriority (n);
  ```

  Where n is any integer between **1** to **10**;
- The thread class defines several priority constants as follow :

  ```
  MIN_PRIORITY = 1;
  NORM_PRIORITY = 5;
  MAX_PRIORITY = 10;
  ```

Example:

```
a.SetPriority (10);     // it allocates Maximum Priority
Threadname.SetPriority (n);
```

- The default setting for the thread is NORM_ PRIORITY i.e. = 5
- By assigning priority to the threads we unsure that they are given attention that they deserve.

Example: We may need to answer an input as quickly as possible whenever we are using multiple threads we can assign maximum priority to this thread.

```
class A extends Thread
{     public void run()
      {     for (int i=1 ; i<=5 ; i++)
                  System.out.println ("Thread A : " + i);
            System.out.println ("Exit From A");
      }
}
class B extends Thread
{     public void run()
      {     for (int j=1 ; j<=5 ; j++)
                  System.out.println ("Thread B : " + j);
            System.out.println ("Exit From B");
      }
}
class C extends Thread
{     public void run()
      {     for (int k=1 ; k<=5 ; k++)
                  System.out.println ("Thread C : " + k);
```

```
            System.out.println ("Exit From C");
      }
}
class ThreadPrioDemo
{     public static void main (String args[])
      {     A T1 = new A();   B T2 = new B();   C T3 = new C();
            T3.setPriority (Thread.MAX_PRIORITY);
            T1.setPriority (Thread.MIN_PRIORITY);
            T2.setPriority (Thread.NORM_PRIORITY);
//          T2.setPriority (T3.getPriority ()-2);
            T1.start();       T2.start();       T3.start();
      }
}
```

Q. Explain Thread Exception in detail

- We should write sleep method in try block followed by a catch block. This is necessary because sleep method throws an exception which should be caught.
- Java runtime system allows us to throw an exception when we attempt to invoke a method that a thread cannot handle.
- **Example:** A Sleeping thread cannot deal with resume method because when the thread is blocked it cannot receive any instruction.
- The catch statement may take one of the following forms:

```
catch (ThreadDeath e)
{
      Body of catch; // Killed thread
}
catch (InterruptedException e)
{
      Body of catch; // Cannot Handle in Current State
}
catch (IllegalArgumentException e)
{
      Body of catch; // Illegal Method Argument
}
catch (Exception e)
{
      Body of catch; // Any Other
}
```

Q. Explain Thread Synchronization in detail.

Synchronization & Monitor

- Synchronization in java is the capability of control the access of multiple threads to any shared resource.
- Java Synchronization is better option where we want to allow only one thread to access the shared resource.
- If thread that are running are using they own data and methods provided inside they run method than it will be ok.
- But the problem arises when they try to use data & methods outside the run method of themselves.

- In such a situation they are using same resource and may complete for that resource and this will create serious problem.
- **Example:** One thread may try to read a record from the file and another thread is still writing records to the same file depending on the situation we may get strange result.
- Two overcome this problem we can see Synchronization Java supports the technique of Synchronization by using synchronize keyword.

Understanding the concept of Lock in Java

- Synchronization is built around an internal entity known as the lock or monitor. Every object has a lock associated with it. By convention, a thread that needs consistent access to an object's fields has to acquire the object's lock before accessing them, and then release the lock when it's done with them.

Java synchronized method

- If you declare any method as synchronized, it is known as synchronized method.
- Synchronized method is used to lock an object for any shared resource.
- When a thread invokes a synchronized method, it automatically acquires the lock for that object and releases it when the thread completes its task.

  Example: the method that will read information from a file & the method that will Write the file way be declared as synchronized.

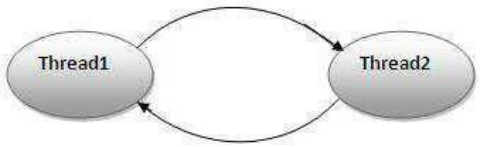
```
Synchronized void read ()
{  Body of loop;     }
Synchronized void Write ()
{  Body of loop;     }
```

- When we declare a method as synchronized Java creates a monitor and hang its over to the thread, that class the method first.
- As long as thread holds the monitor no other thread can enter in the synchronized section of code.
- It is also possible to write a synchronized block, it will lock the particular object of the class. This can be done as follow :

```
Synchronized (lock - Object)
{
   Body of loop;
}
```

Deadlock:

- Deadlock in java is a part of multithreading. Deadlock can occur in a situation when a thread is waiting for an object lock, that is acquired by another thread and second thread is waiting for an object lock that is acquired by first thread. Since, both threads are waiting for each other to release the lock, the condition is called deadlock.



- While using synchronization we must be careful to use synchronized method or synchronized block.
- In some situation two or more threads are waiting for the some resource and due to some reason the condition on which thread waiting will never occur. So the resource will be never allocated to it. This situation is known as Deadlock.
- **Example:** Assume that the Thread A must access the method2 before it can realize method1. At the same time the thread B are using method2 and it will realize method2 before accessing method1.
- So method1 will be never allocated to method2 and method2 to method1.

```
Thread A
Synchronized method1()
{
   Synchronized method2() {   }
}
Thread B
Synchronized method2()
{
   Synchronized method1() {   }
}
```

Java Stream I/O (File Handling)

- Java I/O (Input and Output) is used to process the input and produce the output based on the input.
- Java uses the concept of stream to make I/O operation fast. The ***java.io*** package contains all the classes required for input and output operations.
- We can perform file handling in java by **java IO API.**
- The ***java.io*** package provides a file class that supports for creating files and directories.
- This includes read and writes permission time of last modification, length etc…

Q. Explain File Class

- The Java.io.File class is an abstract representation of file and directory pathnames.
- Instances may or may not denote an actual file-system object such as a file or a directory. If it does denote such an object then that object resides in a partition. A partition is an operating system-specific portion of storage for a file system.
- A file system may implement restrictions to certain operations on the actual file-system object, such as reading, writing, and executing. These restrictions are collectively known as access permissions.

- Instances of the File class are immutable; that is, once created, the abstract pathname represented by a File object will never change.
- Java File class represents the files and directory pathnames in an abstract manner. This class is used for creation of files and directories, file searching, file deletion etc.
- The File object represents the actual file/directory on the disk. There are following constructors to create a File object:

Constructors:

- **File(File parent, String child)** - This method creates a new File instance from a parent abstract pathname and a child pathname string.
- **File(String pathname)** - This method creates a new File instance by converting the given pathname string into an abstract pathname.
- **File(String parent, String child)** - This method creates a new File instance from a parent pathname string and a child pathname string.

- File class defines two characters constant they are separator **Backslash (\)** & Path **Separator char (;).**
- The different methods are available for file to handle the file.

#	Methods with Description
1	**public String getName()** Returns the name of the file or directory denoted by this abstract pathname.
2	**public String getParent()** Returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory.
3	**public File getParentFile()** Returns the abstract pathname of this abstract pathname's parent, or null if this pathname does not name a parent directory.
4	**public String getPath()** Converts this abstract pathname into a pathname string.
5	**public boolean isAbsolute()** Tests whether this abstract pathname is absolute. Returns true if this abstract pathname is absolute, false otherwise
6	**public String getAbsolutePath()** Returns the absolute pathname string of this abstract pathname.
7	**public boolean canRead()** Tests whether the application can read the file denoted by this abstract pathname. Returns true if and only if the file specified by this abstract pathname exists and can be read by the application; false otherwise.
8	**public boolean canWrite()** Tests whether the application can modify to the file denoted by this abstract pathname. Returns true if and only if the file system actually contains a file denoted by this abstract pathname and the application is allowed to write to the file; false otherwise.
9	**public boolean exists()** Tests whether the file or directory denoted by this abstract pathname exists. Returns true if and only if the file or directory denoted by this abstract pathname exists; false otherwise
10	**public boolean isDirectory()** Tests whether the file denoted by this abstract pathname is a directory. Returns true if and only if the file denoted by this abstract pathname exists and is a directory; false otherwise.
11	**public boolean isFile()** Tests whether the file denoted by this abstract pathname is a normal file. A file is normal if it is not a directory and, in addition, satisfies other system-dependent criteria. Any non-directory file

	created by a Java application is guaranteed to be a normal file. Returns true if and only if the file denoted by this abstract pathname exists and is a normal file; false otherwise.
12	**public long lastModified()** Returns the time that the file denoted by this abstract pathname was last modified. Returns a long value representing the time the file was last modified, measured in milliseconds since the epoch (00:00:00 GMT, January 1, 1970), or 0L if the file does not exist or if an I/O error occurs.
13	**public long length()** Returns the length of the file denoted by this abstract pathname. The return value is unspecified if this pathname denotes a directory.
14	**public boolean delete()** Deletes the file or directory denoted by this abstract pathname. If this pathname denotes a directory, then the directory must be empty in order to be deleted. Returns true if and only if the file or directory is successfully deleted; false otherwise.
15	**public void deleteOnExit()** Requests that the file or directory denoted by this abstract pathname be deleted when the virtual machine terminates.
16	**public boolean renameTo(File dest)** Renames the file denoted by this abstract pathname. Returns true if and only if the renaming succeeded; false otherwise.
17	**public boolean setReadOnly()** Marks the file or directory named by this abstract pathname so that only read operations are allowed. Returns true if and only if the operation succeeded; false otherwise.
18	**public boolean equals(Object obj)** Tests this abstract pathname for equality with the given object. Returns true if and only if the argument is not null and is an abstract pathname that denotes the same file or directory as this abstract pathname.
19	**public String toString()** Returns the pathname string of this abstract pathname. This is just the string returned by the getPath() method.

Example: FileDemo.java & FileDemo2.java

Q. Explain RandomAccessFile class

- The **Java.io.RandomAccessFile** class file behaves like a large array of bytes stored in the file system.Instances of this class support both reading and writing to a random access file.
- It is used to read and write data to a File in random positions. The RandomAccessFile class treats the file as an array of Bytes. And you can write your data in any position of the Array.

Constructors:

- **RandomAccessFile(File file, String mode)** - This creates a random access file stream to read from, and optionally to write to, the file specified by the File argument.
- **RandomAccessFile(File file, String mode)** - This creates a random access file stream to read from, and optionally to write to, a file with the specified name.

Methods:

#	Method	Description
1	long getFilePointer()	This method returns the current offset in this file.
2	long length()	This method returns the length of this file.
3	int read(byte[]b)	This method reads up to *b.length* bytes of data from this file into an array of bytes.
4	void seek(long pos)	This method sets the file-pointer offset, measured from the beginning of this file, at which the next read or write occurs.

5	void write(byte[]b)	This method writes b.length bytes from the specified byte array to this file, starting at the current file pointer.

Example: RandomAccessFile_Example.java

Q. Explain various types of I/O streams.

Streams (Input & Output)

- A stream is a sequence of data. In Java a stream is composed of bytes. It's called a stream because it's like a stream of water that continues to flow.
- In java, 3 streams are created for us automatically. All these streams are attached with console.
  1. **System.out:** standard output stream
  2. **System.in:** standard input stream
  3. **System.err:** standard error stream
- Let's see the code to print **output** and **error** message to the console.

```
System.out.println ("simple message");
System.err.println ("error message");
```

- Let's see the code to get **input** from console.

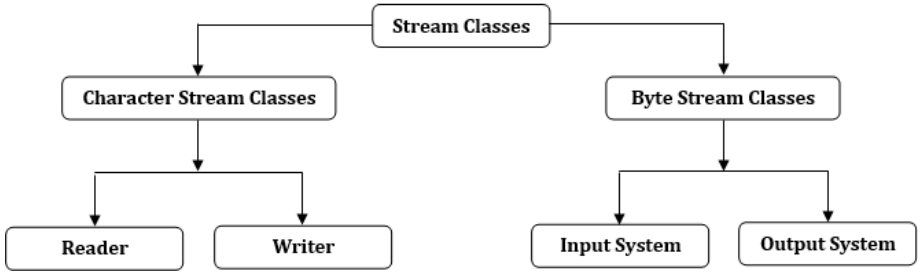
```
int i = System.in.read();           //returns ASCII code of 1st character
System.out.println ((char) i);      //will print the character
```

- A Stream is an ordered sequence of data that is a common characteristics shared by input output devices.
- A stream has a source and a destination of data. A stream is link to a physical device by JAVA IO system for example, an input stream may read data from the keyboard and send it to the memory. While output stream reads the data from the memory and write it to the monitor/printer.

There are two types of streams available:

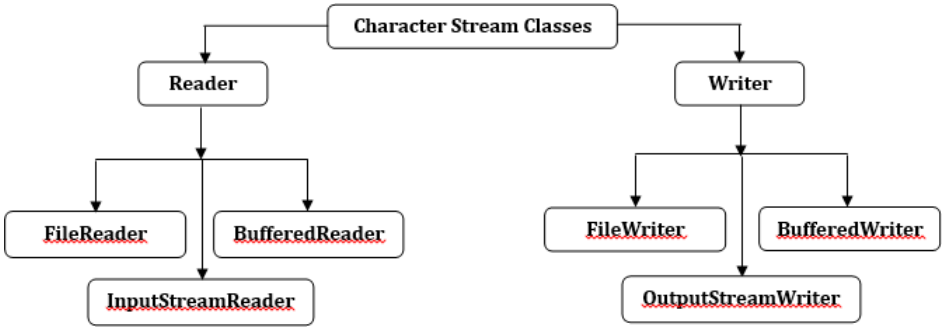
1. **Character Stream**
   - Character stream allows us to read and write characters and strings, the characters are stored and rewrite in human readable format.
   - An input character stream converts bytes to the character and output character stream converts to the byte.
2. **Byte Stream**
   - Byte stream allows us to read and write binary data. In this data are accessed as a sequence of bytes. It is not in the human readable format.

Stream class provides the following hierarchy of classes.



1) Character Stream Class

- Character stream class allows us to read and write characters and streams. The characters are stored in the file in human readable format. This class has following hierarchy of classes.



❖ Reader

- Reader Stream classes are used to read character. This class defines the functionality that is available for all character input streams.
- Reader is an abstract class so we cannot create an instance object of this class. It is used as a base class.

FileReader

- File reader is a subclass of Input Stream reader class. It inputs characters form a file.

  Constructors:
  - ✓ **FileReader (String filepath);**
  - ✓ **FileReader (File obj);**

BufferedReader

- The BufferedReader class is a subclass of reader class and it buffers input from character string.

  Constructors:
  - ✓ **BufferedReader (Reader obj);**

  - ✓ **BufferedReader (Reader obj, int bsize);**

InputStreamReader

- Input stream reader class is subclass of reader class. It converts streams of bytes to streams of characters. This is done according to the specific character and coding.

  Constructors:
  - ✓ **InputStreamReader (InputStream is);**
  - ✓ **InputStreamReader (InputStream is, string encoding);**

❖ Writer

- The writer stream classes are used to perform all output operations on file. The writer class is an abstract class which ads as a base class for all other writer stream classes.

FileWriter

- File Writer class is a subclass of Output Stream Writer class. It output a character to a file.

  Constructors:
  - ✓ **FileWriter (String filepath);**
  - ✓ **FileWriter (File obj);**

BufferedWriter

- The buffered writer class is a sub class of writer class. It buffers output to a character string.

  Constructors:
  - ✓ **BufferedWriter (Writer obj);**
  - ✓ **BufferedWriter (Writer obj, int bsize);**

OutputStreamWriter

- The output stream writer is a sub class of Writer Class. It converts a stream of characters to a stream of bytes. This is done according to specific characters encoding.

  Constructors:
  - ✓ **OutputStreamWriter (Outputstream OS);**
  - ✓ **OutputStreamWriter (Outputstream OS, String Encoding);**

PrintWriter

- The print Writer is a sub class of writer class.
- It displays the String equivalent of Simple types such as int, float, char objects etc..

  Constructors:
  - ✓ **PrintWriter(Outputstream OS);**
  - ✓ **PrintWriter (Output stream OS, Boolean f);**
  - ✓ **PrintWriter (Writer OS);**
  - ✓ **PrintWriter (Writer OS, Boolean f);**

Reading and Writing through Character Stream Classes

FileWriter and FileReader

- Java FileWriter and FileReader classes are used to write and read data from text files. These are character-oriented classes, used for file handling in java.
- Java has suggested not to use the FileInputStream and FileOutputStream classes if you have to read and write the textual information.

FileWriter class

- Java FileWriter class is used to write character-oriented data to the file.

Constructors:

- **FileWriter (String file)** - creates a new file. It gets file name in string.
- **FileWriter (File file) - creates a new file. It gets file name in File object.**

Methods:

#	Method	Description
1	public void write(String text)	writes the string into FileWriter.
2	public void write(char c)	writes the char into FileWriter.
3	public void write(char[] c)	writes char array into FileWriter.
4	public void flush()	flushes the data of FileWriter.
5	public void close()	closes FileWriter.

Example of FileWriter

In this example, we are writing the data in the file abc.txt.

Example: FileWriter_Simple.java

FileReader class

- Java FileReader class is used to read data from the file. It returns data in byte format like FileInputStream class.

  Constructors:

  - ✓ **FileReader (String file) -** It gets filename in string. It opens the given file in read mode. If file doesn't exist, it throws FileNotFoundException.
  - ✓ **FileReader (File file) -** It gets filename in file instance. It opens the given file in read mode. If file doesn't exist, it throws FileNotFoundException.

Methods:

#	Method	Description
1	public int read()	returns a character in ASCII form. It returns -1 at the end of file.
2	public void close()	closes FileReader.

Example of FileReader

In this example, we are reading the data from the file abc.txt file.

Example: FileReader_Simple.java

BufferedWriter and BufferedReader

BufferedWriter

- The Java.io.BufferedWriter class writes text to a character-output stream, buffering characters so as to provide for the efficient writing of single characters, arrays, and strings.
- The buffer size may be specified, or the default size may be used.
- A Writer sends its output immediately to the underlying character or byte stream.
- The buffered writer class is a sub class of writer class. It buffers output to a character string.

Constructors:

- **BufferedWriter (Writer obj) -** This creates a buffered character-output stream that uses a default-sized output buffer.
- **BufferedWriter (Writer obj, int bsize) -** This creates a new buffered character-output stream that uses an output buffer of the given size.

Methods:

#	Method	Description
1	void close()	This method closes the stream, flushing it first.
2	void flush()	This method flushes the stream.
3	void newLine()	This method writes a line separator.
4	void write(char[] cbuf, int off, int len)	This method writes a portion of an array of characters.
5	void write(int c)	This method writes a single character.
6	void write(String s, int off, int len)	This method writes a portion of a String.

Example: BufferedWriter_Test.java

BufferedReader

- The Java.io.BufferedReader class reads text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines.
- The buffer size may be specified, or the default size may be used.
- Each read request made of a Reader causes a corresponding read request to be made of the underlying character or byte stream.
- The BufferedReader class is a subclass of reader class and it buffers input from character string.

Constructors:

- **BufferedReader (Reader obj) -** This creates a buffering character-input stream that uses a default-sized input buffer.
- **BufferedReader (Reader obj, int bsize) -** This creates a buffering character-input stream that uses an input buffer of the specified size.

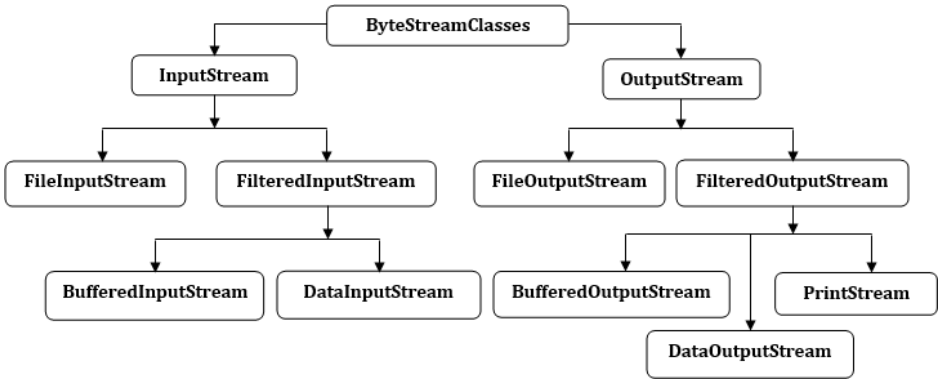
Methods:

#	Method	Description
1	void close()	This method closes the stream and releases any system resources associated with it.
2	void mark(intreadAheadLimit)	This method marks the present position in the stream.
3	boolean markSupported()	This method tells whether this stream supports the mark() operation, which it does.
4	int read()	This method reads a single character.
5	int read(char[] cbuf, int off, int len)	This method reads characters into a portion of an array.
6	String readLine()	This method reads a line of text.
7	boolean ready()	This method tells whether this stream is ready to be read.
8	void reset()	This method resets the stream.
9	long skip(long n)	This method skips characters.

Example: BufferedReader_Test.java

2) Byte Stream Class

- Byte Stream allows a programmer to work with binary data in a file. In this stream data are accessed as a sequence of bytes.
- Byte stream includes output stream, file output stream, data output stream and print stream classes use for output and input stream file input stream, buffered input stream and data input stream class used for input.



❖ Input Stream :

- The abstract input stream class defines the functionality that is available for all byte output streams. It provides different methods for handling input stream.
- This class has following subclasses.

• File Input Stream

- The file input stream class is a sub class of input stream class. It allows us to read binary data from a file.

  Constructors:

  ✓ **FileInputStream (String FilePath);**

  ✓ **FileInputStream (File Object);**

• Filtered Input Stream

- The filtered input stream class is a sub class of input stream class. It filters as input stream. Filter Input Stream is an abstract class, so we cannot create instance of it.

  BufferedInputStream

  - The buffered input stream class is a subclass of Filtered input stream class. It buffers input from a byte stream.

    Constructors:

    ✓ **BufferedInputStream (InputStream is);**

    ✓ **BufferedInputStream (InputStream is, int bsize);**

  Data Input Stream

  - The data Input Stream is a sub class of filtered Input Stream class. It allows to read the simple JAVA types from a byte input stream.

    Constructor:

    ✓ **DataInputStream (InputStream is);**

❖ Output Stream:

- The output stream class defines the functionality for al byte output streams. This class is base class for other output stream classes.
- This class has following subclasses.

• FileOutputStream

- The File Output Stream Class is a subclass of OutputStream. It allows us to write binary data to a file.

  Constructors:

  ✓ **FileOutputStream (String FilePath);**

  ✓ **FileOutputStream (File Object);**

• FilteredOutputStream

- The filtered output stream is a sub class of output stream. It is an abstract class so we can't create object of this class. It has following subclasses :

  Buffered Output Stream

  - The buffered Output Stream class is a subclass of Filtered output stream class. It buffers output from a byte stream.

    Constructors:

    ✓ **BufferedOutputStream (OutputStream OS);**

    ✓ **BufferedOutputStream (OutputStream OS, int bsize);**

DataOutputStream

- The dataOutputStream is a subclass of filteredOutputStream class. It allows us to create byte output stream.

  Constructor:

  ✓ **DataOutputStream (OutputStream OS);**

PrintStream

- The print stream class is a subclass of Filtered Output Class. It provides all the formatting capabilities used with System.out.
- The Static System.out variable is a print stream.

  Constructor:

  ✓ **PrintStream (OutputStream OS);**

  ✓ **PrintStream (OutputStream OS, Boolean F);**

Reading and Writing through Byte Stream Classes

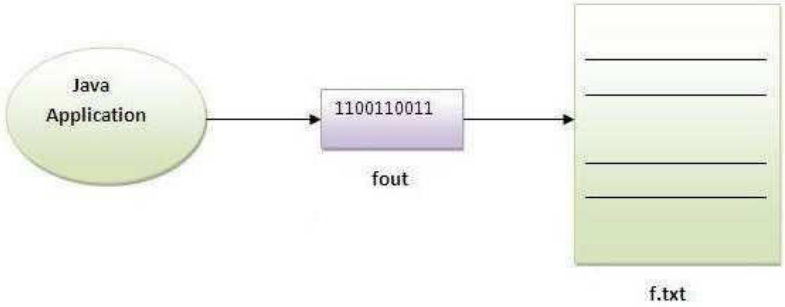
FileInputStream and FileOutputStream

- In Java, FileInputStream and FileOutputStream classes are used to read and write data in file. In another words, they are used for file handling in java.

FileOutputStream class

- Java FileOutputStream is an output stream for writing data to a file.
- If you have to write primitive values then use FileOutputStream. Instead, for character-oriented data, prefer FileWriter.But you can write byte-oriented as well as character-oriented data.

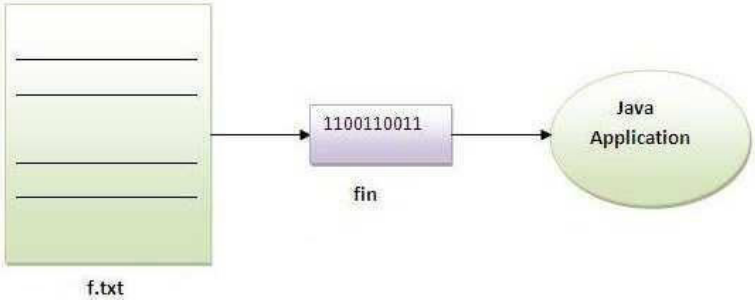
Example: FileInputStream_readTest.java



FileInputStream class

- Java FileInputStream class obtains input bytes from a file. It is used for reading streams of raw bytes such as image data. For reading streams of characters, consider using FileReader.
- It should be used to read byte-oriented data for example to read image, audio, video etc.

Example: FileOutputStream_writeTest.java



Example of Reading the data of current java file and writing it into another file

We can read the data of any file using the FileInputStream class whether it is java file, image file, video file etc. In this example, we are reading the data of File_Copy.java file and writing it into another file File_Paste.java.

Example: File_Copy.java

BufferedOutputStream and BufferedInputStream

BufferedOutputStream class

- Java BufferedOutputStream class uses an internal buffer to store data. It adds more efficiency than to write data directly into a stream. So, it makes the performance fast.

Example of BufferedOutputStream class:

In this example, we are writing the textual information in the BufferedOutputStream object which is connected to the FileOutputStream object. The flush() flushes the data of one stream and send it into another. It is required if you have connected the one stream with another.

Example: BufferedOutputStream_Test.java

BufferedInputStream class

- Java BufferedInputStream class is used to read information from stream. It internally uses buffer mechanism to make the performance fast.

Example of Java BufferedInputStream

Let's see the simple example to read data of file using BufferedInputStream.

Example: BufferedInputStream_Test.java

DataOutputStream and DataInputStream

DataOutputStream class

- The DataOutputStream class enables you to write Java primitives to OutputStream's instead of only bytes. You wrap an OutputStream in a DataOutputStream and then you can write primitives to it.

- ataOutputStream is also useful for creating non-human-readable byte files of data fields such as numbers. Writing number fields into records as Java primitive types is much more efficient than converting them each time and writing them as readable characters.

DataOutputStream class

- The DataInputStream class enables you to read Java primitives from InputStream's instead of only bytes. You wrap an InputStream in a DataInputStream and then you can read primitives from it.
- DataInputStream reads streams of the binary format of data primitive types which were written by DataOutputStream.

Example:

1. DataInputStreamDemo.java
2. DataOutputStreamDemo.java
3. DataIOStream_Simple.java
4. DataIO_Uppercase.java
5. DataIOStreamDemo.java

PrintStream class

- The Java.io.PrintStream class adds functionality to another output stream, the ability to print representations of various data values conveniently.
- The PrintStream class enables you to write formatted data to an underlying OutputStream. For instance, writing int, long and other primtive data formatted as text, rather than as their byte values.
- The PrintStream class contains the powerful format() and printf() methods (they do exactly the same, but the name "printf" is more familiar to C-programmers). These methods allow you to mix text and data in very advanced ways, using a formatting string.

Examples:

1. PrintStreamDemo.java
2. PrintStream_Format.java

Q. Explain StreamTokenizer Class

- The Java.io.StreamTokenizer class takes an input stream and parses it into "tokens", allowing the tokens to be read one at a time. The stream tokenizer can recognize identifiers, numbers, quoted strings, and various comment styles.
- The StreamTokenizer class takes an input stream and parses it into "tokens", allowing the tokens to be read one at a time. The parsing process is controlled by a table and a number of flags that can be set to various states. The stream tokenizer can recognize identifiers, numbers, quoted strings, and various comment styles.

Fields:

Following are the fields for Java.io.StreamTokenizer class:

- **double nval** - If the current token is a number, this field contains the value of that number.
- **String sval** - If the current token is a word token, this field contains a string giving the characters of the word token.
- **static int TT_EOF** - A constant indicating that the end of the stream has been read.
- **static int TT_EOL** - A constant indicating that the end of the line has been read.
- **static int TT_NUMBER** - A constant indicating that a number token has been read.
- **static int TT_WORD** - A constant indicating that a word token has been read.
- **int ttype** - After a call to the nextToken method, this field contains the type of the token just read.

Constructor:

- **StreamTokenizer(Reader r)** - This creates a tokenizer that parses the given character stream.

Methods

#	Method & Description
1	void commentChar(int ch) Specified that the character argument starts a single-line comment.
2	int lineno() This method returns the current line number.
3	void lowerCaseMode(boolean fl) This method determines whether or not word token are automatically lowercased.
4	int nextToken() This method parses the next token from the input stream of this tokenizer.
5	void pushBack() This method causes the next call to the nextToken method of this tokenizer to return the current value in the ttype field, and not to modify the value in the nval or sval field.
6	void quoteChar(int ch) This method specifies that matching pairs of this character delimit string constants in this tokenizer.
7	void resetSyntax() This method resets this tokenizer's syntax table so that all characters are "ordinary." See the ordinaryChar method for more information on a character being ordinary.
8	String toString() This method returns the string representation of the current stream token and the line number it occurs on.
9	void whitespaceChars(int low, int hi) This method specifies that all characters c in the range low <= c <= high are white space characters.
10	void wordChars(int low, int hi) This method specifies that all characters c in the range low <= c >= high are word constituents.

Example:

1. StreamTokenizer_Demo.java
2. StreamTokenizer_nextToken.java

Java ByteArrayOutputStream class

- Java ByteArrayOutputStream class is used to write data into multiple files. In this stream, the data is written into a byte array that can be written to multiple stream.
- The ByteArrayOutputStream holds a copy of data and forwards it to multiple streams.

- The buffer of ByteArrayOutputStream automatically grows according to data.

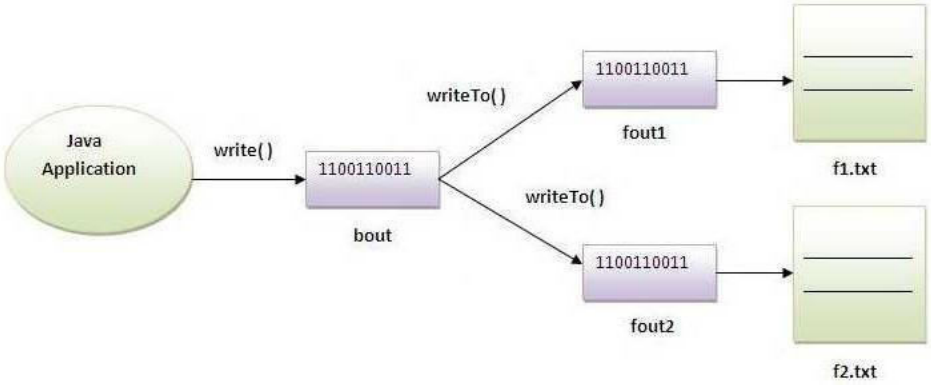
Constructors:

- **ByteArrayOutputStream ()** - creates a new byte array output stream with the initial capacity of 32 bytes, though its size increases if necessary.
- **ByteArrayOutputStream (int size)** - creates a new byte array output stream, with a buffer capacity of the specified size, in bytes.

Methods

#	Method	Description
1	public synchronized void writeTo (OutputStream out) throws IOException	writes the complete contents of this byte array output stream to the specified output stream.
2	public void write(byte b) throws IOException	writes byte into this stream.
3	public void write(byte[] b) throws IOException	writes byte array into this stream.
4	public void flush()	flushes this stream.
5	public void close()	has no affect, it doesn't closes the bytearrayoutputstream.

Example: ByteArrayOutputStream_2Files.java



Java SequenceInputStream class

- Java SequenceInputStream class is used to read data from multiple streams. It reads data of streams one by one.

Constructors:

- **SequenceInputStream (InputStream s1, InputStream s2)** - creates a new input stream by reading the data of two input stream in order, first s1 and then s2.
- **SequenceInputStream (Enumeration e)** - creates a new input stream by reading the data of an enumeration whose type is InputStream.

Simple example of SequenceInputStream class

In this example, we are printing the data of two files f1.txt and f2.txt.

Example: SequenceInputStream_Simple.java

Example of SequenceInputStream that reads the data from two files

In this example, we are writing the data of two files f1.txt and f2.txt into another file named f3.txt.

Example: SequenceInputStream_Concate.java

Example of SequenceInputStream class that reads the data from multiple files using enumeration

If we need to read the data from more than two files, we need to have these information in the Enumeration object. Enumeration object can be get by calling elements method of the Vector class. Let's see the simple example where we are reading the data from the 2 files.

Example: SequenceInputStream_Vector_Enum.java

CS-19 Programming with JAVA

B.C.A. Semester – 4

Unit-4

Applets & Layout Managers

Gaurav sardhara
Mo:-9067351366

1) **What is an Applet?**

Applet is a special type of program that is embedded in the webpage to generate the dynamic content.

It runs inside the browser and works at client side.

2) **List-out various Applet Life cycle stages.**

The applet life cycle involves the following different states.

1. Born state (Initialized)
2. Running state (Started)
3. Display state (Paint)
4. Idle state (Stopped)
5. Dead state (Destroyed)

3) **Which class support drawing of various shapes on applet?**
Graphics class using paint() method of applet.

4) **What is the role of Layout manager?**

The Layout Managers are used to arrange components in a particular manner. Layout Manager is an interface that is implemented by all the classes of layout managers.

5) **List-out various Layout Managers available in Java.**

There are following Layout managers class available in java

- FlowLayout
- BorderLayout
- GridLayout
- CardLayout
- GridBagLayout

6) **List-out various AWT control classes.**

The following are the various AWT control classes.

- Button
- List
- CheckBox
- Label etc.

7) **List-out various AWT container component.**

The following are the container control classes of AWT.

- Applet
- Frame
- Window

- Panel

8) How Canvas is used?

A canvas provides a rectangular area on which we can draw.

· This is valuable because we can graphs and other diagrams on the canvas by using method of the Graphics class.

Q. Explain Applet in detail.

Introduction to Applet Programming

- Applet is a special type of program that is embedded in the webpage to generate the dynamic content. It runs inside the browser and works at client side.
- We can create two types of programs in JAVA
  1. Standard Application Program [Execution using ***java*** command]
  2. Applet Programs [Execution using ***appletviewer*** command]
- An application program can be run directly on the stand alone computer system.
- Applet is a small Java program which is used for internet computing. It can be transported from one computer to another computer using the internet.
- To run applet program we require either **Appletviewer** or **Web browser** that supports JAVA.
- We can embed applet into webpage in two ways :
  1. We can create our own applet and embed it to our web page. This type of applet is called **Local Applet**.
  2. We can download an applet from the remote computer via internet and then embed to our webpage. This type of applet is called **Remote Applet**.
- The **Applet** class is contained in the **java.applet** Package. All applets are subclass of Applets. So all applets must import **java.applet** & applet must import **java.awt (Abstract Window Toolkit)**
- Applet code uses the services of two classes, **Applet & Graphics**.
- Applet is JAVA program that is embedded in A HTML documents and runs in the context of a JAVA capable browser. Java.applet also defines three interfaces: **AppletContext, AudioClip & AppletStub**.

There are some important differences between an applet and a standalone Java application, including the following:

- An applet is a Java class that extends the **java.applet.Applet** class.
- A main() method is not invoked on an applet, and an applet class will not define main().
- Applets are designed to be embedded within an HTML page.
- Any application program can run independently while applet cannot be run independently. They runs from the webpage when it is embedded.
- Applet cannot read & write from the files on the local computer.
- Applet cannot communicate with other servers on the network.
- Applet cannot run any program from the local computer.
- Applets are restricted from using library of other language like c, c++.

- When a user views an HTML page that contains an applet, the code for the applet is downloaded to the user's machine.
- A JVM is required to view an applet. The JVM can be either a plug-in of the Web browser or a separate runtime environment.
- The JVM on the user's machine creates an instance of the applet class and invokes various methods during the applet's lifetime.
- Other classes that the applet needs can be downloaded in a single Java Archive (JAR) file.

Advantage of Applet

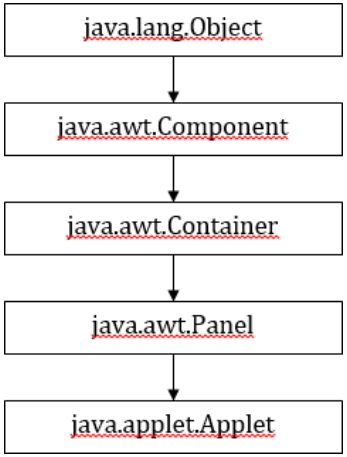
There are many advantages of applet. They are as follows:

- It works at client side so less response time.
- Secured
- It can be executed by browsers running under many platforms, including Linux, Windows, Mac OS, etc.

Drawback of Applet

- Plugin is required at client browser to execute applet

The applet class hierarchy is as follows :



#	Method	Description
1	void destroy()	Destroys the methods
2	AppletContext getAppletContext()	Returns The Applet Context
3	URL getCodeBase()	Returns the code base.
4	URL getDocumentBase()	Returns the documents base
5	Image getImage(URL url)	Returns an Image object for the image at url.
6	Image getImage(URL url, String imgName)	Returns an Image object for the image named imgName relative to url
7	String getParameter (String pName)	Returns the value of parameter pName
8	void init()	Initializes this applet.
9	void showStatus(String str)	Displays str on the status line.

10	void start()	Starts this applet.
11	void stop()	Stops this applet.
12	void paint()	For display something

Creating an Executable Applet Program

We can create an executable applet by using the following steps:

Step-1: Write a source code of applet in .Java file

```
import java.applet.Applet;
import java.awt.Graphics;
public class HelloJava extends Applet
{
	public void paint (Graphics g)
	{
		g.drawString("Hello JAVA", 10, 100);
	}
}
```

Step-2: Save HelloJava.java

Step-3: Compile the source file using Java compiler and store it in the same directory:

```
javac HelloJava.java
```

Step-4: Create a webpage to execute the applet as follows: **"HelloJava.html"**

```
<HTML>
<HEAD>
	<TITLE>Welcome to Applets</TITLE>
</HEAD>
<BODY>
	<Applet Code="HelloJava" width="400" height="100"></Applet>
</BODY>
</HTML>
```

Step-5: Save this file as HelloJava.html in the same folder where HelloJava.java & HelloJava.class file is stored.

Step-6: Execute the **HelloJava.html** Web Page using web browser to view output.

Note:

1. We can also use ***appletviewer*** to display the applet like: **appletviewer HelloJava.html**
2. **appletviewer** only supports the code of applet. It does not support HTML code so it displays only Hello JAVA. It not displays Welcome to Applets.
3. We can directly use <applet> tag just after the import statement as contained in html source code file.

Q. Explain Applet Life Cycle methods in detail.

Every java applet inherits a set of default behaviors from the applet class. The applet life cycle involves the following different states.

1. Born state (Initialized)
2. Running state (Started)
3. Display state (Paint)
4. Idle state (Stopped)
5. Dead state (Destroyed)

Methods:

The java.applet.Applet class provides 4 life cycle methods and java.awt.Component class provides 1 life cycle method for an applet.

- **java.applet.Applet class:**

  For creating any applet java.applet.Applet class must be inherited. It provides 4 life cycle methods of applet.

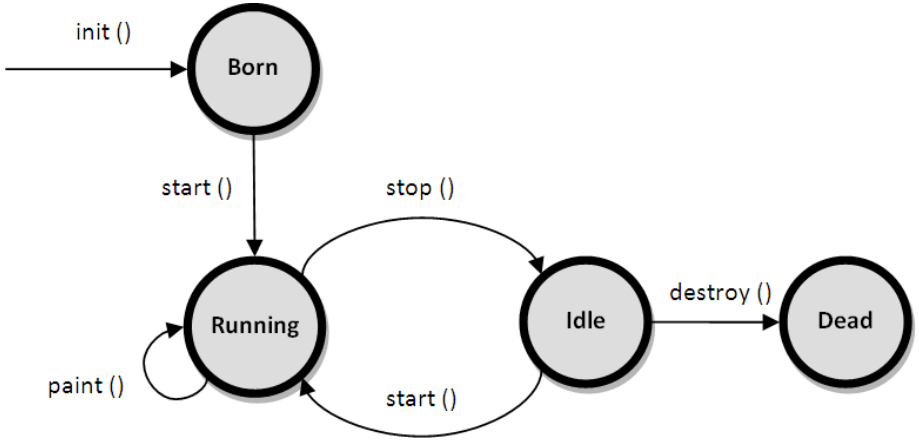
  1. **public void init():** is used to initialized the Applet. It is invoked only once.
  2. **public void start():** is invoked after the init() method or browser is maximized. It is used to start the Applet.
  3. **public void stop():** is used to stop the Applet. It is invoked when Applet is stop or browser is minimized.
  4. **public void destroy():** is used to destroy the Applet. It is invoked only once.

- **java.awt.Component class:**

  The Component class provides 1 life cycle method of an applet.

  5. **public void paint(Graphics g):** is used to paint the Applet. It provides Graphics class object that can be used for drawing oval, rectangle, arc etc.

The following figure shows the state transitions diagram of applet.



Born State (Initialization)

- Applet enters in the initialization state when it is first loaded. This is achieved by calling the **init ( )** method of the applet class. The **Applet** class is born when we perform the following steps to require during this state.
  1. Create object needed by the applet
  2. Set up initial values
  3. Load images & fonts
  4. Set up Colors
- We can override the **init ()** method of Applet class to perform the above task if required in the following way.

```
public void init ()
{
   Statements;
}
```

2) Running State (Started)

- Applet enters in the running state when the system calls the **start ()** method of Applet class. These occurs automatically after the applet is initialize.
- We can also use this method to start the applet which is in ***idle*** state this will restart the applet. Unlike the init ( ) method, the start ( ) method may be called more than once.
- We may override the start ( ) method of applet class as follows

```
public void start ()
{
   Statements;
}
```

3) Display State (Paint)

- When applet moves to the display state it has to perform some output operation on the screen. This happens immediately after the applet enters into running state.
- The **paint ()** method is used to perform this task. Almost every applet will have paint () method. The default version of paint ( ) method does absolute nothing. Thus to display something over the screen we must override the paint ( ) method. It can be override as follows:

```
public void paint (Graphics g)
{
   Display Statements;
}
```

paint ( ) method is inherited from the **Component** class, a super class of **Applet**.

4) Idle State (Stopped)

- An applet becomes idle when it is stopped from running. Stopping occurs automatically when we leave the page containing the currently running applet.
- We can also use **stop ( )** method explicitly. This method can be override as follows:

```
public void stop ()
{
   Statements;
}
```


5) Dead State (Destroyed)

- An applet is said to dead when it is removed from the memory. This occur automatically by invoking **destroy ( )** method. When we quit the browser, like initialization destroying stage is also occurs only once in the applet life cycle.
- We can override destroyed method to clean up the resources as follows:

```
public void destroy ()
{
   Statements;
}
```

Example: Write a program to demonstrate the different method of the applet class

Save file: AppletMethods.java

```
import java.applet.Applet;
import java.awt.Graphics;
/*    <Applet Code="AppletMethods" width="400" height="100"></Applet>    */
public class AppletMethods extends Applet
{
      String State1 = "", State2 = "", State3 = "", State4 = "";
      public void init()
      {
            State1 += "Initialization";
      }
      public void start()
      {
            State2 += "Start";
      }
      public void stop()
      {
            State3 += "Stop";
      }
      public void destroy ()
      {
            State4 += "Destroy";
      }
      public void paint (Graphics g)
      {
            g.drawString (State1, 10, 20);
            g.drawString (State2, 10, 40);
            g.drawString (State3, 10, 60);
            g.drawString (State4, 10, 80);
      }
}
```

Save file: AppletMethods.java

```
<HTML>
<HEAD>
      <TITLE>Welcome to Applet Methods</TITLE>
</HEAD>
<BODY>
      <Applet Code="AppletMethods" width="400" height="100"></Applet>
</BODY>
</HTML>
```

Q. Explain Graphics class in java.

Graphics Programming

- One of the most important features of JAVA is its ability to draw graphics. We can write JAVA applet that draw lines, figures, different shapes, images & text. We can also use different colors for displaying graphics.
- Every applet has its own area of the screen known as "canvas".
- The size for an applet space is decided by the width & height attributes of the <applet> tag.

Methods

There are different methods available in graphics class to draw the different shapes following are some of them.

1. **drawstring()**

- This method is used to draw (display) the specified string on the canvas of an Applet.

Syntax: `public abstract void drawstring (String str, int x, int y)`

Where str is the String content to be draw on the Canvas. Where x & y are the left & top coordinate of the canvas area of an Applet.

Example: `g.drawString ("Hello World !!!", 10, 40);`

2. **drawLine ()**

- This method is used to draw a line on the applet canvas. It draws the line between the points (x1, y1) and (x2, y2).

Syntax: `public abstract void drawLine(int x1, int y1, int x2, int y2)`

Where x1, y1, is the starting co-ordinate & x2, y2 is the ending co-ordinate of the line.

Example: `g.drawLine (10, 100, 50, 250);`

3. **drawRect ()**

- This method is use for drawing Rectangle. It draws a rectangle with the specified width and height.

Syntax: `public void drawRect (int x, int y, int width, int height)`

Where x, y, is the starting co-ordinate, width & height is the size of a rectangle.

Example: `g.drawRect (10, 100, 50, 250);`

4. **fillRect ()**

- This method is used to draw & fill the rectangle with default color and specified width & height.

Syntax: `public abstract void fillRect (int x, int y, int width, int height)`

Example: `g.fillRect (10, 100, 50, 50);`

5. **drawRoundRect ()**

- This method is use for drawing Rectangle which rounded corners.

Syntax: `public abstract void drawRect (int x1,int y1,int w,int h,int xr,int yr)`

Where x1, y1, is the starting co-ordinate, w is the width; h is the height & xr - yr are x rounded & y rounded of the rectangle.

Example: `g.drawRoundRect (10, 100, 50, 50, 10, 10);`

6. **fillRoundRect ()**

- This method is used to draw a filled Rounded rectangle.

Syntax: `public abstract void fillRoundRect (int x1,int y1,int w,int h,int xr,int yr)`

Example: `g.fillRoundRect (10, 100, 50, 50, 10, 10);`

7. **drawOval ()**

- This method is used to draw an Oval with the specified width and height.

Syntax: `public abstract void drawOval (int x, int y, int width, int height)`

Where x, y is the starting point of Oval Rectangle, width & height is the size of the Oval.

Example: `g.drawOval (10, 100, 50, 50);`

8. **fillOval ()**

- This method is used to draw a filled Oval with the default color and specified width and height.

Syntax: `public abstract void fillOval (int x, int y, int width, int height)`

Example: `g.fillOval (10, 100, 50, 50);`

9. **drawArc ()**

- This method is used to draw a circular shape or an Arc.

Syntax: `public abstract void drawArc (int x, int y, int width, int height, int startAngle, int arcAngle)`

Where x, y is the starting point of Arc Rectangle, width & height is the size of the Arc & last two arguments represents starting angle & ending angle of the Arc.

Example: `g.drawArc (10, 10, 100, 50, 0, 90);`

10. **fillArc ()**

- This method is used to fill a circular shape or an Arc.

Syntax: `public abstract void fillArc (int x, int y, int width, int height, int startAngle, int arcAngle)`

Example: `g.fillArc (10, 10, 100, 50, 0, 90);`

11. **drawPolygon ()**

- This method is used to draw Polygons.

Syntax: `public abstract void drawPolygon (int[] xPoints, int[] yPoints, int nPoints)`

- Polygons are shapes with many sides. It may be considered as a set of lines connected together.
- The end of first line is the beginning of second line, the end of second line is the beginning of the third line so on.
- This method takes three arguments first arguments is an array of integers corresponding to the x coordinates. The second arguments is an integer corresponding to the y coordinates and the third arguments is the total number of points.

- This means that we can draw polygon with n sides by using **drawline( )** method n times.
- However we can draw polygons more continently using **drawPolygon( )** method.

Example:
```
int xpoints[] = {25, 145, 25, 145, 25};
int ypoints[] = {25, 25, 145, 145, 25};
int npoints = 5;
g.drawPolygon (xpoints, ypoints, npoints);
```

12. **drawImage ()**

- Applet is mostly used in games and animation. For this purpose image is required to be displayed. The java.awt.Graphics class provide a method drawImage () to display specified image on the canvas.

  Syntax: `public abstract boolean drawImage(Image img, int x, int y, ImageObserver observer)`

How to get the object of Image

- The java.applet.Applet class provides getImage () method that returns the object of Image.

  Syntax: `public Image getImage(URL u, String image){}`

Other required methods of Applet class to display image:

- **public URL getDocumentBase():** is used to return the URL of the document in which applet is embedded.
- **public URL getCodeBase():** is used to return the base URL

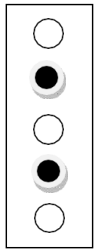
Example:
```
import java.awt.*;
import java.applet.*;
public class funnyImage extends Applet
{     Image picture;
      public void init()
      {     setBackground(Color.lightGray);
            picture = getImage(getDocumentBase(),"funny_face.jpg");
      }
      public void paint(Graphics g)
      {     g.drawImage (picture, 100, 50, this);     }
}
```

Example: write a program to use all above methods
```
import java.awt.*;
import java.applet.*;
public class GraphicsDemo extends Applet
{     public void paint (Graphics g)
      {     g.drawLine (10,10,50,50);     // Draw a Line
            g.drawRect (10,60,40,30);     // Draw a Rectangle
            g.fillRect (60,10,30,80);     // Draw a Filled Rectangle
            g.drawRoundRect (10,100,50,50,10,10);     // Draw a Rounded Rectangle
            g.fillRoundRect (20,110,30,30,5,5); // Draw a Filled Rounded Rectangle
            g.drawOval (80,40,200,200);         // Draw an Oval
            g.fillOval (120,80,40,40);          // Draw a Filled Oval [Right Eye]
            g.fillOval (190,80,40,40);          // Draw a Filled Oval [Left Eye]
            g.drawArc (140,180,80,40,0,-180);   // Draw an Arc
            g.fillArc (150,130,50,90,0,90);     // Draw a Filled Arc
            g.drawLine(100,50,260,50);
            g.drawLine(100,50,180,5);
            g.drawLine(260,50,180,5);
            int xa[] = {270, 320, 370, 320};
            int ya[] = {80, 10, 80, 110};
            int n = 4;
            g.drawPolygon (xa, ya, n);          // Draw a Polygon
```



```
      }
}
```

Example: write a program to display the following applet:
```
import java.awt.*;
import java.applet.*;
/*    <Applet Code=drawCir height=300 width=300></Applet>  */
public class drawCir extends Applet
{     public void paint (Graphics g)
      {     g.drawOval (150,40,50,40);
            g.fillOval (150,90,50,40);
            g.drawOval (150,140,50,40);
            g.fillOval (150,190,50,40);
            g.drawOval (150,240,50,40);
      }
}
```



Q. Explain Color Class

- You can set the color of the pen or the background of the display.
- This class encapsulates colors using the RGB format. In RGB format, the red, blue, and green components of a color are each represented by an integer in the range 0-255. The value 0 indicates no contribution from this primary color. The value 255 indicates the maximum intensity of this color component.
- The **java.awt.Color** class is used to set different colors. Each object of the class represent a particular color with the help of this class we can draw color strings shapes in an applet.
- The **Color** class also defines several constants that represent specific colors. These are black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white & yellow.

Constructor

- **Color (int, int, int)**

  Creates a color with the specified red, green, and blue components.
- **Color (int)**

  Creates a color with the specified RGB value, where the red component is in bits 16-23 of the argument, the green component is in bits 8-15 of the argument, and the blue component is in bits 0-7.
- **Color (float, float, float)**

  Creates a color with the specified red, green, and blue values, where each of the values is in the range 0.0-1.0.

- We can define our own color as follow :`color C = new color (100, 200, 255);`
- Now, we can use this color as background color that is **this.setBackground (C)** ;
- We can use setColor() method for setting foreground color that is **g.setColor (C)**;

No.	Method	Description
1	Color brighter ()	Returns a brighter version of the current object
2	Color darker ()	Returns a darker version of the current object
3	int getRed()	Returns the Red components of the current object.
4	int getGreen()	Returns the green components of the current object.
5	int getBlue()	Returns the blue components of the current object.

6	int getRGB()	Returns an int encoding of the current object.
7	Boolean equals(Object obj)	Returns True if the current object and obj. represent the same colorvalue.

Example: write a program to use Color class methods

```
import java.awt.*;
import java.applet.*;
/*    <applet code="ColorDemo" width="400" height="400"></applet> */

public class ColorDemo extends Applet
{
      Color   c[]   =   {Color.yellow,   Color.black,   Color.darkGray,   Color.gray,
Color.magenta, Color.blue, Color.cyan, Color.green, Color.red, Color.orange};

      public void paint (Graphics g)
      {
            int bar = 250 / c.length;
            for (int i=0; i < c.length ; i++)
            {
                  g.setColor (c[i]);
                  g.fillRect (i * bar, 0, (i + 1) * bar, 300);
            }
            for (int i=0 ; i < c.length ; i++)
            {
                  g.setColor (c[i]);
                  g.fillRect (0, (i + 1) * bar , i * bar, 300);
            }
      }
}
```

Displaying Text

- The drawString ( ) method of graphics class is used to display text. This text will be of default type and color.
- However we can display String by using different modes. This can be done by using Font Class of **awt package**.
- A font determines the size and appearance of the character String.

Constructor

Font (string name, int style, int pt)

Here name is identifies the name of font.

Style may be bold, italic, and plain

Pt is the size of font. After creating Font we can use that font using **setFont ( )** *method of graphics class.*

No.	Method	Description
1	int charWidth(char c)	Returns the width of c
2	int charWidth(int i)	Returns the width of the character in the lowest 16 bits of i.
3	int getAscent()	Returns the Ascent.
4	int getDescent()	Returns the Descent.
5	int getHeight()	Returns the Height.
6	int getLeading ()	Returns the Leading.

7	int stringWidth(String Str)	Returns width of Str.

Example: write a program to use Color class methods

```
import java.awt.*;
import java.applet.*;
public class FontDemo1 extends Applet
{
      public void paint (Graphics g)
      {
            g.setFont (new Font ("serif", Font.BOLD, 36));
            g.setColor (Color.blue);
            g.drawString ("Font Example",5,100);
      }
}
```

Input Data from User:

- Applet works in a graphical environment therefore applet treats input as a text string. We must first create an area of the screen in which user can type and edit the data.
- This can be done by using **TextField** class of the applet package.
- Once TextField are created for receiving input we can type the data in the fields and edit if necessary.
- If we input the data which is numeric that data must be converted to the appropriate data type before it is used for processing.

Example: Write a program to input two numbers in applet and display sum of it.

```
import java.awt.*;
import java.applet.*;
/*    <applet code="SumDemo" width="300" height="100"></applet>   */
public class SumDemo extends Applet
{     TextField text1, text2;
      public void init()
      {     text1 = new TextField (8);
            text2 = new TextField (8);
            add (text1);      add (text2);
            text1.setText("0");
            text2.setText("0");
      }
      public void paint (Graphics g)
      {     int x=0, y=0, z=0;
            String s1, s2, s;
            g.drawString ("Input No in Each Box", 10, 50);
            try
            {
                  s1 = text1.getText();
                  x = Integer.parseInt(s1);
                  s2 = text2.getText();
                  y = Integer.parseInt(s2);
            }
            catch (Exception e)     {     }
            z = x + y;
            s = String.valueOf(z);
            g.drawString ("The sum is : ", 10, 70);
            g.drawString (s, 100, 75);
      }
```

```
      public boolean action(Event e, Object o)
      {     repaint();        return true;      }
}
```

Applet Dimension

- We can dynamically find the dimension of an Applet
- We can display a circle at the center of an applet. If the applet is resize the circle must remain at the center. This can be achieved by using **getSize ( )** method.

Dimension (Dimension d); here d is a Dimension Object

Dimension (int w, int h); here arguments w and h represent the width & height in pixels.

Example:

```
import java.awt.*;
import java.applet.*;
public class DimensionDemo extends Applet
{
      public void paint (Graphics g)
      {
            Dimension d = getSize();
            int x = d.width / 2;
            int y = d.height / 2;
            int  radius  =  (int)  (d.width  <  d.height  ?  0.4  *  d.width  :  0.4  *
d.height);
            g.drawOval(x - radius, y - radius, 2 * radius, 2 * radius);
      }
}
```

Passing Parameter to the Applet

- We can pass parameter to the applet. This can be done by **<PARAM……>** tag as an attribute of the applet tag.

Example-1:

```
import java.awt.*;
import java.applet.*;
// Parameters will be taken from GetParam1.html file using <param> tag
public class GetParam1 extends Applet
{
      String str;
      public void init()
      {
            str = getParameter ("Str");
            if (str == null)
                  str = "[ No parameter passed ]";
            str = "Hello, " + str;
      }
      public void paint (Graphics g)
      {     g.drawString(str, 30, 30);    }
}
<HTML>
<HEAD>
      <TITLE>Passing Parameter to JAVA Applets</TITLE>
</HEAD>
<Body>
<APPLET CODE="GetParam1.class" WIDTH="300" HEIGHT="250">
      <PARAM Name = "Str" value = "Applet!">
</Applet>
```



```
</Body>
</HTML>
```

Example-2:

```
import java.awt.*;
import java.applet.*;
// Parameters will be taken from GetParam2.html file using <param> tag
public class GetParam2 extends Applet
{
      public void paint (Graphics g)
      {
            if (getParameter ("string") == null)
                  g.drawString("[ No parameter passed ]", 20, 20);

            String myfont = getParameter ("font");
            String mystring = getParameter ("string");
            int mysize = Integer.parseInt(getParameter("size"));

            Font f = new Font (myfont, Font.BOLD, mysize);

            g.setFont(f);
            g.setColor(Color.red);
            g.drawString(mystring, 10, 40);
      }
}
<HTML>
<HEAD>
      <TITLE>Passing Parameter to JAVA Applets</TITLE>
</HEAD>
<Body>

<APPLET CODE="GetParam2.class" WIDTH="400" HEIGHT="300">
      <PARAM Name = "font" VALUE = "comic sans ms">
      <PARAM Name = "size" VALUE = "30">
      <PARAM Name = "string" VALUE = "Hello, World... It’s me.">
</Applet>

</Body>
</HTML>
```

Q. Explain various types of Layout manager.

Layout Manager

- The Layout Managers are used to arrange components in a particular manner. Layout Manager is an interface that is implemented by all the classes of layout managers.
- In graphical user interface **add ( )** method is used to add a component on an Applet.
- However it is necessary to specify a location for that element. This cannot be done using add () method.
- We can use the layout manager which will determine the place of the component.
- A layout manager automatically arranges controls within an applet window. If an Applet is resized the layout manager automatically adjust the place of the elements.

- Components may have different size on different platforms. This all is adjusted by the layout manager.
- Advantages of layout manager are: **Placed, Resized, different platform adjusted.**
- A layout manager is associated with each container object

  Layout Manager getLayout ( );

  void setLayout (LayoutManager lm); for getting & setting the current layout method.

  Here, lm is a reference to an object that implements the LayoutManager interface.

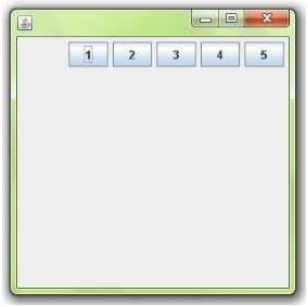
- There are different Layout Manager Classes available in JAVA we can use these according to our application.

1. **FlowLayout**
2. **BorderLayout**
3. **GridLayout**
4. **CardLayout**
5. **GridBagLayout**

1) **java.awt.FlowLayout**
2) **java.awt.BorderLayout**
3) **java.awt.GridLayout**
4) **java.awt.CardLayout**
5) **java.awt.GridBagLay-out**

1) **javax.swing.BoxLayout**
2) **javax.swing.GroupLayout**
3) **javax.swing.ScrollPaneLayout**
4) **javax.swing.SpringLay-out**

Q. Explain Flow Layout



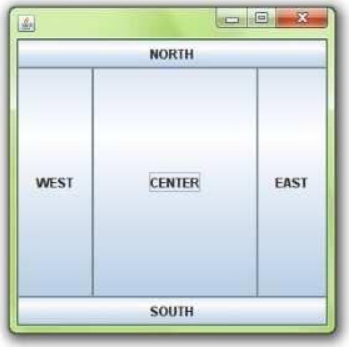
- The FlowLayout is used to arrange the components in a line, one after another (in a flow). It is the default layout of applet or panel.
- It is the default layout Manager. Components are put the upper left corner, left to right and top to bottom.
- When no more components fit on a line, the next one appears on the next line.

Constructors:

- **FlowLayout ():** This is default layout, it creates a flow layout with centered alignment and a default 5 unit horizontal and vertical gap.
- **FlowLayout (int align):** creates a flow layout with the given alignment and a default 5 unit horizontal and vertical gap.
- **FlowLayout (int align, int hgap, int vgap):** creates a flow layout with the given alignment and the given horizontal and vertical gap

#	Fields of FlowLayout class:	Valid Values
1	public static final int LEFT	FlowLayout.LEFT
2	public static final int RIGHT	FlowLayout.RIGHT
3	publicstaticfinalintCENTER	FlowLayout.CENTER
4	public static final int LEADING	FlowLayout.LEADING
5	public static final int TRAILING	FlowLayout.TRAILING

Q. Explain Border Layout



- The BorderLayout is used to arrange the components in five regions: north, south, east, west and center. Each region (area) may contain one component only. It is the default layout of frame or window.
- A border layout class implements a common Layout style for top-level windows.
- It has four narrow regions and one large region in the center. The four sides are referred as North, South, East, West

Constructors:

- **BorderLayout():** creates a border layout but with no gaps between the components.
- **JBorderLayout(int hgap, int vgap):** creates a border layout with the given horizontal and vertical gaps between the components.
- A different form of the ***add()*** method must be used in conjunction with a border layout manager as follows: **void add(Component compobj, Object region);**
- The BorderLayout provides five constants for each region:

#	Fields of BorderLayout class:	Valid Values
1	public static final int NORTH	BorderLayout.NORTH
2	public static final int SOUTH	BorderLayout.SOUTH
3	public static final int EAST	BorderLayout.EAST
4	public static final int WEST	BorderLayout.WEST
5	public static final int CENTER	BorderLayout.CENTER

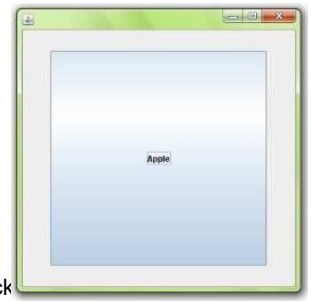
Q. Explain Grid Layout

- The GridLayout is used to arrange the components in rectangular grid. One component is displayed in each rectangle.
- A border layout class automatically arranges components in a grid. All components are created with equal size.

Constructors:

- **GridLayout():** creates a grid layout with one column per component in a row.
- **GridLayout(int rows, int columns):** creates a grid layout with the given rows and columns but no gaps between the components.
- **GridLayout(int rows, int columns, int hgap, int vgap):** creates a grid layout with the given rows and columns along with given horizontal and vertical gaps

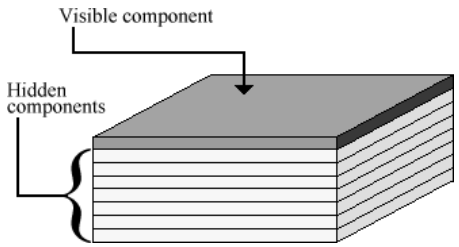
Q. Explain Card Layout



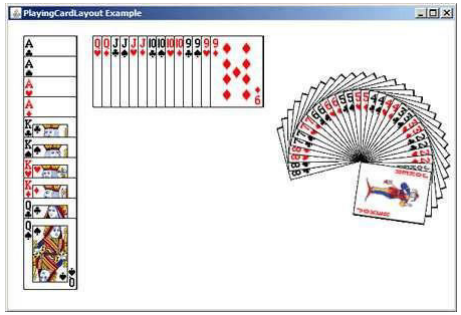
- The CardLayout class manages the components in such a manner that only one component is visible at a time. It treats each component as a card that is why it is known as CardLayout.
- The card Layout is unique among the other Layout managers in that it stores several different layout.
- Each layout can be thought as a separate index card in a deck

that can be shuffled so that any card is on Top at any time.

Constructors:



- **CardLayout():** creates a card layout with zero horizontal and vertical gap.
- **CardLayout(int hgap, int vgap):** creates a card layout with the given horizontal and vertical gap.
- The cards are typically held in an object of type panel. This panel must have card layout selected as a layout managers.



- The cards that form the deck are also typically object of type panel. Thus we must create a panel that contains the deck and a panel for each card in the deck.
- After creating deck and cards of the deck next step is to add components in each card.
- After all this process, we must provide some way for the user to select particular card.
- When card panels are added to a panel they are given a name and ***void add (Component panelObj, Object name)*** method is used to adding carding in the panel.

We can activate a card by calling one of the **methods** defined by ***CardLayout*** class.

- **public void next(Container parent):** is used to flip to the next card of the given container.
- **public void previous(Container parent):** is used to flip to the previous card of the given container.
- **public void first(Container parent):** is used to flip to the first card of the given container.
- **public void last(Container parent):** is used to flip to the last card of the given container.
- **public void show(Container parent, String name):** is used to flip to the specified card with the given name.

Q. Explain GridBag Layout

- Some applet require a more control over how component are arranged within the applet window
- A good way to do this is to use GridBagLayout.
- In GridBagLayout we can specify the relative place of components by specifying their positions within cell inside a grid.

- The key to GridBag is that each component can be a different size and each row in the grid can have different number of columns.
- The general procedure for using GridBag is to first create a new GridBagLayout object and set it as current Layout managers.

- Then set the constraints that apply to each components that will be added to GridBag. Finally add the components to the Layout manager.

Constructor:

- **GridBagLayout():** Creates a grid bag layout manager..
- We can use **setConstraints()** method of GridBagLayout class to define the constraints or the component.

#	Field	Purpose
1	int anchor	It specifies the location of a component within a cell. The default is GridBagConstraints.Center
2	int fill	Specifies how a component is resized of the component is smaller than its cell.
3	int gridheight	It specifies the height of component in term of cell. The default is 1.
4	int gridwidth	It specifies the width of component in term of cell. The default is 1.
5	int gridx	Specifies the X coordinates of the cell to which the component will be added.
6	int gridy	Specifies the Y coordinates of the cell to which the component will be added.

- GridBagConstraints also defines several static fields that contain standard constraint values, such as GridBagConstraints. CENTER & GridBagConstraints.VERTICAL.

- When a component is smaller than its cell, you can use the anchor field to specify where within the cell the component's top-left corner will be located. There are two types of values that you can give to anchor.

- **The first are absolute :**

GridBagConstraints.CENTER
GridBagConstraints.SOUTH
GridBagConstraints.EAST
GridBagConstraints.SOUTHEAST
GridBagConstraints.NORTH
GridBagConstraints.SOUTHWEST
GridBagConstraints.NORTHEAST
GridBagConstraints.WEST
GridBagConstraints.WEST

- The second type of values that can be given to anchor is relative, which means the values are relative to container's orientation, which might differ for non-western languages. The relative values are shown here :

GridBagConstraints.FIRST_LINE_END
GridBagConstraints.LINE_END
GridBagConstraints.FIRST_LINE_START
GridBagConstraints.LINE_START
GridBagConstraints.LAST_LINE_END
GridBagConstraints.PAGE_END
GridBagConstraints.LAST_LINE_START
GridBagConstraints.PAGE_START

Q. Explain Box Layout

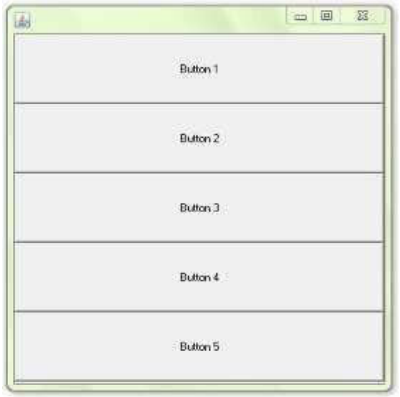
- The BoxLayout is used to arrange the components either vertically or horizontally.
- For this purpose, BoxLayout provides four constants.
  1) public static final int **X_AXIS**
  2) public static final int **Y_AXIS**
  3) public static final int **LINE_AXIS**
  4) public static final int **PAGE_AXIS**

Note: BoxLayout class is found in ***javax.swing*** package.

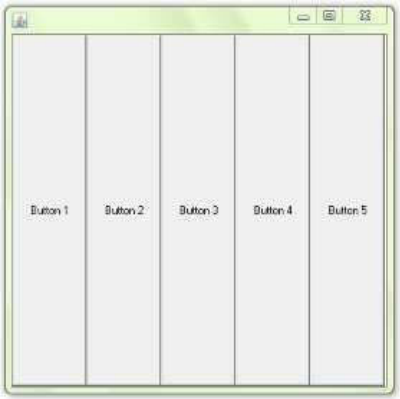
Constructor:

- **BoxLayout (Container c, int axis):** creates a box layout that arranges the components with the given axis.

Y_AXIS

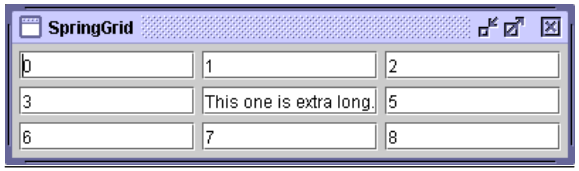


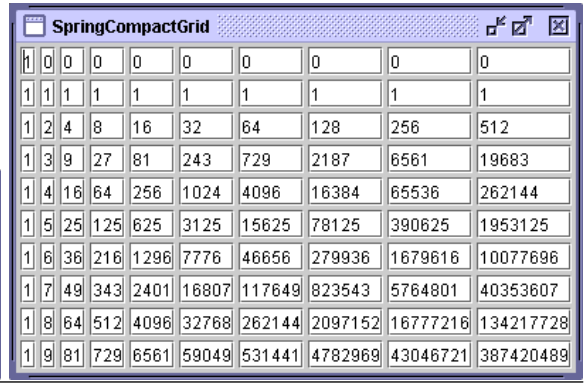
X_AXIS

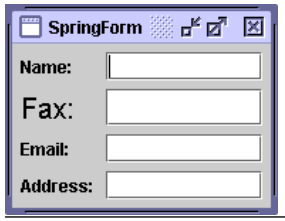


Q. Explain Spring Layout

- The class **SpringLayout** positions the children of its associated container according to a set of constraints.
- **SpringLayout** is a very flexible layout manager that can emulate many of the features of other layout managers.







Examples

http://warhol.wiwi.hu-berlin.de/~berendt/Dsurkov/users/dsurkov/public_html/java/uiswing/layout/spring.html

For more examples of Layout Managers, visit below link...

http://www.java2s.com/Tutorial/Java/0240__Swing/Catalog0240__Swing.htm

CS-19 Programming with JAVA

B.C.A. Semester – 4

Unit - 5

AWT & GUI using Swing & Event Handling

Gaurav k sardhara
Mo:- 9067351366

1) **How AWT components are differ than Swing components?**

Swing components provides more functionality than AWT components. They are light weighted & supports MVC rich set of GUI.

2) **List-out various Swing Component classes.**

The following are the Swing component classes:

- JButton
- JList
- JLabel
- JComboBox etc.

3) **What Event Delegation Model does in Java with GUI component?**

The modern approach to handling events is based on Event Delegation Model which defines standard mechanism to generates and process event.

· It is a standard mechanism for a source to generate an event and send it to the listener.

· Changing the state of an object is known as an event.

4) **Define Event.**

An Event is an object which changes the state where any source is active or inactive. OR An Event is an object that describes a state change in a source.

5) **What is an EventListener.**

A listener is an object that is notified when an event occurs.

6) **List-out various types of Events supported by an EventListner.**

The following are the events that supported by an EventListner.

- Item Event
- Action Event
- Focus Event
- Key Event
- Container Event
- Component Event
- Mouse Event etc.

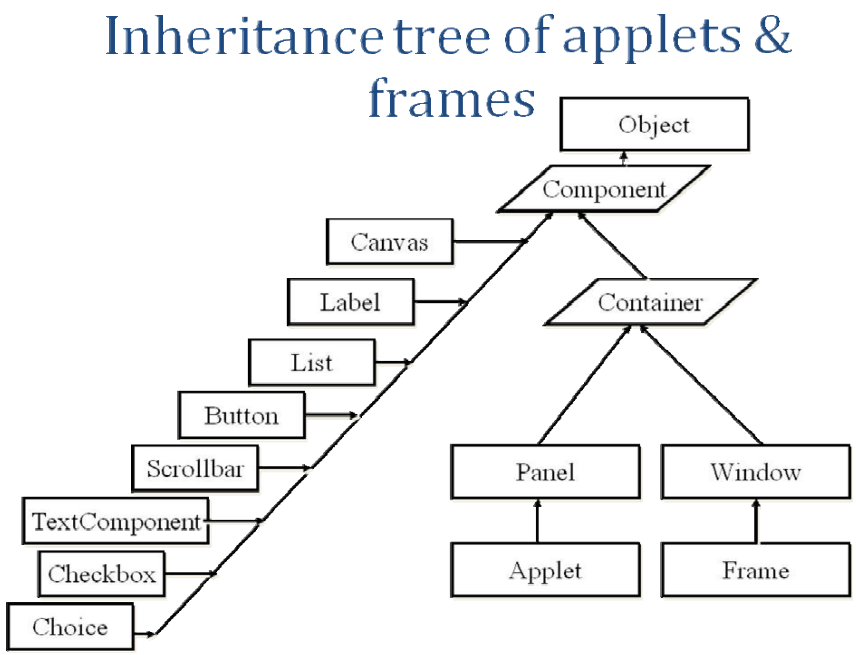
7) **What is the role of Adapter classes in Java?**

· An adapter class provides an empty implementation of all the methods a particular listener interface.

Q. Explain AWT Controls (Abstract Windowing Toolkit)

- JAVA provides AWT controls which is declared in AWT package as a class.
- It provides full supports for user interface components like labels, buttons, checkbox, textfields, scrollbars, menus, etc...

- These all controls can be put into the container. The different container available is panel, window and frame.



Q. What is Container? Explain in detail

A container is an area that can hold elements, can be drawn on and painted. There is a class named *Container* in *java.awt*package, from which two commonly used containers: *Frame* and *Panel* are derived directly or indirectly. The *Container* class is a subclass of the *Component* class.

Frames and panels are by far the most commonly used containers. Frame is a separate window whereas Panel is an area contained within a window. A panel does not have borders and is displayed within the window provided by a browser or the *appletviewer*. *Appletviewer* is a tool provided by the JDK to view *applets*.

1) Label

- A label is a string that appears on a graphical user interface. It is a passive control that does not support any interruption with the user.

Constructors:

Label ( );
Label (String str);
Label (String str, int align);

Where Str is the text which we want to display as a label and align is a constant which is used for left, right or Centeralignment.

Example: Write a program to demonstrate the Use of Label:

```
import java.awt.*;
import java.applet.*;
public class LabelDemo extends Applet
{
```

```
        public void init ()
        {
                String S="My First Label";
                Label L1 = new Label (s, Label.LEFT);
                Label L2 = new Label (s, Label.CENTER);
                Label L3 = new Label (s, Label.RIGHT);
                Add (L1);
                Add (L2);
                Add (L3);
        }
}
```

2) Button

- A button is a component that appears as a push button, when button is pressed it generate as event.
- A button class define the following constructors :
  Button ();
  Button (String Str);
- An action event is generated each time a button is pressed. The method which other objects register and unregister to receive action events generated by this components are given below :
  void addActionListener (ActionListener al)
  void removeActionListener (ActionListener al)
- An action Listener used **getActionCommand()** to obtain a command string associated with an action generated by the button.
- We can **SetActionCommand(String Str)** method to set the string return by the **getActionCommand().**

Example: Write a program to demonstrate the Use of button:

```
import java.awt.*;
import java.applet.*;
```

3) Checkbox

- A checkbox is a component that combines the label and small box. It is control that is used to turn on or turn off a particular option.
- A smallbox can either contain or checkmark or empty (checked or Unchecked)
- The step of the checkbox can be changed while clicking the mouse on the box.
- It has following constructors:
  Checkbox ( );
  Checkbox (StringStr);
  Checkbox (String Str, boolean state);
  Checkbox (String Str, boolean state, CheckboxGroup grp);
  Checkbox (String Str, CheckboxGroup grp, boolean state);
- The methods by which other objects register and unregister to receive these events are as follows :
  void addItemListener (ItemListener al)

void removeItemListener (ItemListener al)

4) Check Box Group

- A checkbox Group is a set of checkbox out of which any one can be selected at a time it is also known as radio buttons.
- For this first we have to define a checkbox group and then we can add checkboxes to this group.
- The CheckboxGroup class defined only one constructor that is : **CheckboxGroup ();**
- We can obtain the group member that is currently set via **Checkbox getSelectedCheckbox()** method.
- We can also set particular checkbox in the group via void **setSelectedCheckbox(Checkbox cb)** method.
- The methods by which other objects register and unregister to receive these events are as follows :
  void addItemListener (ItemListener al)
  void removeItemListener (ItemListener al)

5) Choice (Similar to Combo Box)

- A choice is component that provides a list of menu when user clicks on a choice it will display the list of choice and selection can be made. Only one item can be selected at a time.
- Choice class has only one constructors that is **Choice ( )** and item event is generated when user selects any item form the given list.
- The methods by which other objects register and unregister to receive these events are as follows :
  void addItemListener (ItemListener al)
  void removeItemListener (ItemListener al)

6) List (Similar to List Box)

- A list is a component that allows a user to select one or more items if the list is large then it will automatically provide scrollbar so that user can set the entire list.
- An action event is generated whenever user double clicks on an item in a list.
- An item event is generated when a user selects or deselects an item in a list.
- This list defines the following constructors :
  List (); //default constructor
  List (int rows);
  In this the rows indicates the numbers of items that are visible to the user.
  List (int rows, Boolean Multiple);
  In this, multiple parameters if it is set to true user can select multiple entries in a list otherwise only one item can be selected at a time.
- The methods by which other objects register and unregister to receive these events are as follows :
  void addItemListener (ItemListener al)
  void removeItemListener (ItemListener al)

- The method by which other objects register and unregister to receive action events generated by this components are given below :
  void addActionListener (ActionListener al)
  void removeActionListener (ActionListener al)

7) Scrollbar

- Scrollbar are used to select any Integer values between s specified minimum & maximum values.
- A scrollbar contains a slider that may be dragged to continuously change its value alternatively the user may click on one of the buttons on the scrollbar to move slider. These operations modify the value of scrollbar.
- The Scrollbar class has the following constructors :
  Scrollbar ( );
  Scrollbar (int orientation);
  Scrollbar (int orientation, int value, int width, int min, int max);
- Scrollbar may be oriented horizontally or vertically. It is assigned the value HORIZONTAL & VERTICAL.
- The argument value represents the initial value of the scrollbar.
- The width of the slider is width.
- The range of the scrollbar is determined by the min & max arguments.
- An **AdjustmentEvent** is generated whenever the value represented by a scroll bar changed, the **java.awt.event.AdjustmentEvent** class used.

8) TextField

- A textfield allows a user to enter one line of text.
- Textfield allow the User to enter string and to edit the text using the arrow keys, cut & paste keys, and mouse selection.
- TextField defines the following constructors :
  TextField();
  TextField(String Str);
  TextField(int cols);
  TextField(String Str, int cols);
  Here, str is the text for the field. The argument cols indicate the width of the field in character.
- We can use **void setEchoChar(char ch)** method to set the character which will be echoed in th screen.
- We can use **Char getEchoChar()** method to get the echo character.
- We can check a text field to see if it is in this mode with the **Boolean echoCharIsSet()** method.

9) TextArea

- A TextArea allows a user t input multiple line of text.
- TextArea defines the following constructors :
  TextArea();

  TextArea(String str);
  TextArea(int rows, int cols);
  TextArea(String str, int rows, int cols);
  TextArea(String str, int rows, int cols, int Scrollbars);

- Str is the text for the TextArea.
- Rows and cols represent the no of rows and columns in the component.
- Scrollbar indicates vertical scrollbar, horizontal scrollbar both…

For adding text to the TextArea the following methods are available.

void append (string str);
The append () method append the string specified by str to the end of the current text.
void insert (String str, int index);
insert ()method inserts the string passed in str at the specified index..
void replaceRange (string str, int strartIndex, int endIndex);
To replace text, call replaceRange(). It replace the characters from startIndex to endIndex-1, with the replacements text passed in str.

10) Canvas

- A canvas provides a rectangular area on which we can draw.
- This is valuable because we can graphs and other diagrams on the canvas by using method of the Graphics class.
- Canvas defines this constructors: **Canvas ()**

11) Menu Bars & Menus

- A window can have a menu bar associated with it. A menu bar displays a list of top-level menu choices. Each choice is associated with a drop-down menu. This concept in implemented by the following classes :
  - ✓ **MenuBar**
  - ✓ **Menu**
  - ✓ **MenuItem**
- In short a MenuBar contains one or more menu objects each menu object contains a list of menu objects.
- Each menu item object represents something that can be selected by the user.
- To create a MenuBar first create an instance of MenuBar. This class defines only one constructor :
  MenuBar();
- After creating MenuBar next create the instance of menu that will define the selections displayed on the bar.
- Menu class has following constructors :
  Menu ( );

Menu (String optioName);

Menu (String optionName, Boolean removable);

- After adding different options to the menu individual menu items can be added by MenuItem class. It defines the following Constructors

  MenuItem ( );

  MenuItem (String itemName);

  MenuItem (String itemName, MenuShortcut KeyAcc);

12) Dialog Box

- Dialog boxes are used to obtain user input. They are similar to frame windows, except theat dialog boxes are always child window of a top-level window.
- Dialog box do not have menu bars. You cannot access the other parts of your progress until you have closed the dialog box.
- Constructors for dialog box are :

  Dialog (Frame parentWindow, Boolean mode);

  Dialog (Frame parentWindow, String title, Boolean mode);

- Here parentWindow is the owner of the dialog box. If mode is true the dialog box is modal. Otherwise it is modeless.
- The title of the dialog box can be passed in title.

13) File Dialog

- The FileDialog class extends Dialog and provides a dialog that allows a user to select a file for reading or writing.
- This is very useful when user want to save a data to a file ans later retrieve tha data.
- This class defines two int constants named File.Dialog.LOAD & File.Dialog.Save
- These determine if the file dialog is being used to select a file for reading or writing. Constructors for File Dialog are:

  FileDialog (Frame parent)

  FileDialog (Frame Parent, String str);

  FileDialog (Frame Parent, String str, int rw);

- Here parent is the frame that creates this dialog. The title bar of the file dialog is initialized form str. The final argument indicates if this fie dialog identifies a file for loading or saving. If rw is LOAD, a file is for reading, if rw is SAVE, a file is for writing.

Q. Explain Java Swing components.

- Java Swing tutorial is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.
- Unlike AWT, Java Swing provides platform-independent and lightweight components.
- The ***javax.swing*** package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.
- Java introduces a package called Abstract Window Toolkit (AWT). This package contains a large number of classes and interfaces that supports GUI.
- This component gives different looks or different platform because they are using native code of platform and it is the main limitation of these components.
- To solve this problem JAVA2 introduce a new classes backed in swing. These swing classes are contain in JAVA extension package known as JAVAX.
- The method in swing class does not use the native graphical method directly. The methods in swing classes are purely written in JAVA Thus, it provides same look on different platforms.
  - It is light weight program.
  - Access becomes fast.
  - Fully platform independent
  - It provides special effort on each component.

Difference between AWT and Swing

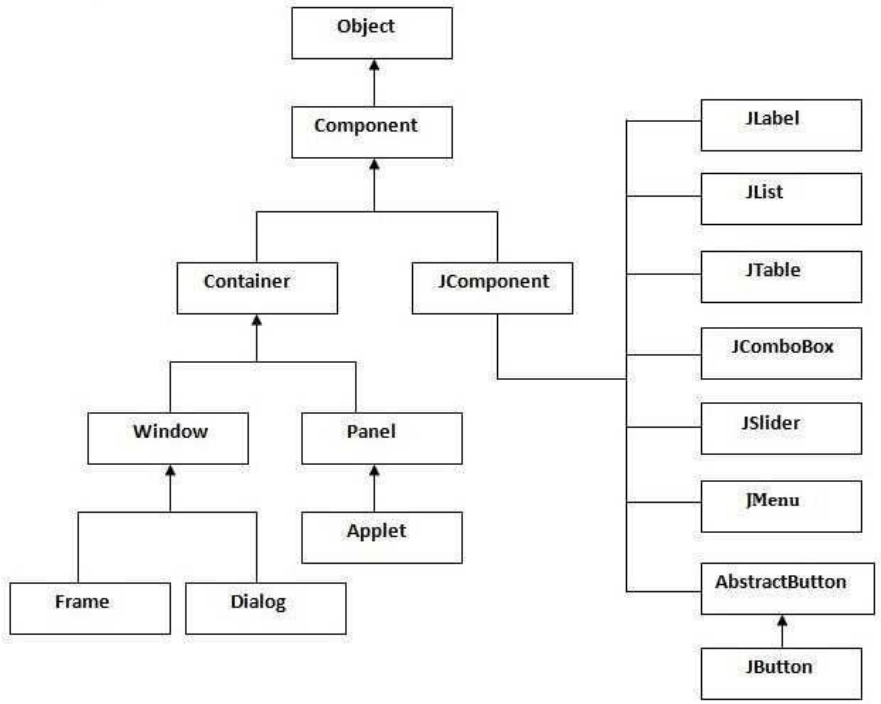
There are many differences between java awt and swing that are given below.

#	Java AWT	Java Swing
1	AWT components are **platform-dependent**.	Java swing components are **platform-independent**.
2	AWT components are **heavyweight**.	Swing components are **lightweight**.
3	AWT **doesn't support pluggable look and feel**.	Swing **supports pluggable look and feel**.
4	AWT provides **less components** than Swing.	Swing provides **more powerful components** such as tables, lists, scrollpanes, colorchooser, tabbedpane etc.
5	AWT **doesn't follows MVC** (Model View Controller) where model represents data, view represents presentation and controller acts as an interface between model and view.	Swing **follows MVC**.

What is JFC?

- The Java Foundation Classes (JFC) are a set of GUI components which simplify the development of desktop applications.

Hierarchy of Java Swing classes



Methods:

The methods of Component class are widely used in java swing that are given below.

#	Method	Description
1	public void add(Component c)	add a component on another component.
2	public void setSize(int width,int height)	sets size of the component.
3	public void setLayout(LayoutManager m)	sets the layout manager for the component.
4	public void setVisible(boolean b)	sets the visibility of the component. It is by default false.

Java Swing Examples

There are two ways to create a frame:

1) By creating the object of Frame class (association)
2) By extending Frame class (inheritance)

We can write the code of swing inside the ***main()***, constructor or any other method.

✓ **Simple Java Swing Example**

Let's see a simple swing example where we are creating one button and adding it on the JFrame object inside the main() method.

Example: FirstSwingExample.java

✓ **Example of Swing by Association inside constructor**

We can also write all the codes of creating JFrame, JButton and method call inside the java constructor.

Example: SimpleConstructorExample.java

✓ **Simple example of Swing by inheritance**

We can also inherit the JFrame class, so there is no need to create the instance of JFrame class explicitly.

Example: SimpleInheritanceExample.java

Window in Swing

- A conventional window is created is swing in a top level window. The top level window is supported by Jframe, JApplet, JWindow & JDialogue.
- The top level window are called RootContainer. These root containers have several layers / panes like JRootPane, JMenubar, JLayeredPane, ContentPane and GlassPane.
- The **add ( )** method is used to add the components to the container.
- General syntax of this method is : ***void add (component);***
- To create a user interface window the following steps are to be followed:
  1. Create a root container using JFrame, JApplet, JWindow or JDialogue.
  2. Get the container using get Content panes ( )
  3. Create a component
  4. Attach the component to a container using add ( ) method.

JFrame

- A JFrame window is subclass of frame class, when a JFrame window is created its size is (0, 0) and it is invisible. It contains different icons like maximize, close, etc… but it is invisible.
- Once a frame is created we can use it for our purpose. We can use window closing event to close the window using System.exit (0) method.

Constructors:

✓ **JFrame ();**

✓ **JFrame (String title);**

JApplet

- JApplet class extends Applet class. Applet Class use swing must be a subclass of JApplet.
- JApplet has several functionality which are not found in applet Class.

Example: JApplet supports various panes like Content pane, Glass pane and root pane.

JPanel

- JPanel is exactly same as a Panel. We can place component to the JPanel and then JPanel can be added to the container.
- When we are using JPanel first we must set the size and then we can use paint method. This can be done suing paint component for drawing.

JComponents

The JComponent class is a subclass of container this class contains a large number of subclass like JButton, JLabel, Jcombobox, Etc…

JLabel

- JLabel is a sub class of Jcomponent. It can display text and icons.

Constructors:

- ✓ **JLabel (icon i);**
- ✓ **JLabel (String S);**
- ✓ **JLabel (String S, int align);**
- ✓ **JLabel (String S, icon i, int align);**

  Here S is string which we want to display as Label, i is the icon & align is the constant which can be left, right, center.

JButton

- The JButton class is used to create a button that have platform-independent implementation.

Constructors:

- ✓ **JButton ( ):** creates a button with no text and icon.
- ✓ **JButton (String s):** creates a button with the specified text.
- ✓ **JButton (Icon i):** creates a button with the specified icon object.

Methods:

1	**public void setText(String s):** is used to set specified text on button.
2	**public String getText():** is used to return the text of the button.
3	**public void setEnabled(boolean b):** is used to enable or disable the button.
4	**public void setIcon(Icon b):** is used to set the specified Icon on the button.
5	**public Icon getIcon():** is used to get the Icon of the button.
6	**public void setMnemonic(int a):** is used to set the mnemonic on the button.
7	**public void addActionListener(ActionListener a):** is used to add the action listener to this object.

Example: ImageButton.java

JRadioButton

- The JRadioButton class is used to create a radio button. It is used to choose one option from multiple options. It is widely used in exam systems or quiz.
- It should be added in ***ButtonGroup*** to select one radio button only.

Constructors:

- ✓ **JRadioButton ():** creates an unselected radio button with no text.
- ✓ **JRadioButton (String s):** creates an unselected radio button with specified text.
- ✓ **JRadioButton (String s, boolean selected):** creates a radio button with the specified text and selected status.

Methods:

1	**public void setText(String s):** is used to set specified text on button.
2	**public String getText():** is used to return the text of the button.
3	**public void setEnabled(boolean b):** is used to enable or disable the button.
4	**public void setIcon(Icon b):** is used to set the specified Icon on the button.
5	**public Icon getIcon():** is used to get the Icon of the button.
6	**public void setMnemonic(int a):** is used to set the mnemonic on the button.
7	**public void addActionListener(ActionListener a):** is used to add the action listener to this object.

Example: RadioSimple.java

<u>ButtonGroup class:</u>

The ButtonGroup class can be used to group multiple buttons so that at a time only one button can be selected.

Example: RadioExample.java

JCheckBox

- JCheckBox class provides the functionality of checkbox. Its immediate super class is JToggleButton which provides support for two state buttons.

Constructors:

- ✓ **JCheckBox (String S);**
- ✓ **JCheckBox (icon i);**
- ✓ **JCheckBox (String S, icon i);**
- ✓ **JCheckBox (String S, Boolean State);**
- ✓ **JCheckBox (icon i, Boolean State);**
- ✓ **JCheckBox (String S, icon i, Boolean State);**

JTextField

- The Swing TextField can be displayed by using JTextField.
- JTextField is a subclass of Jcomponent. It allows entering and editing any text from the user. The Text must ne single line only.

Constructors:

- ✓ **JTextField ();**
- ✓ **JTextField (String S);**
- ✓ **JTextField (int cols);**
- ✓ **JTextField (String S, int cols);**

JPasswordField

- JPasswordField creates a TextField same as JtextField only the difference is insisted of the actual character the * will be displayed.

Constructors:

- ✓ **JPasswordField ( );**
- ✓ **JPasswordField (String s);**
- ✓ **JPasswordField (int size);**
- ✓ **JPasswordField (String s, int size);**

JTextArea

- The JTextArea class is used to create a text area. It is a multiline area that displays the plain text only.

Constructors:

- ✓ **JTextArea ():** creates a text area that displays no text initially.
- ✓ **JTextArea (String s):** creates a text area that displays specified text initially.
- ✓ **JTextArea (int row, int column):** creates a text area with the specified number of rows and columns that displays no text initially.
- ✓ **JTextArea (String s, int row, int column):** creates a text area with the specified number of rows and columns that displays specified text.

Methods:

1	**public void setRows(int rows):** is used to set specified number of rows.
2	**public void setColumns(int cols)::** is used to set specified number of columns.
3	**public void setFont(Font f):** is used to set the specified font.
4	**public void insert(String s, int position):** is used to insert the specified text on the specified position.
5	**public void append(String s):** is used to append the given text to the end of the document.

Example: TArea.java

JScrollbar

- JScrollbar is the subclass of JComponent the scrollbar are available in two orientation vertical and horizontal. They are defined as JScrollbar.HORIZONTAL & JScrollbar.VERTICAL

Constructors:

- ✓ **JScrollbar ( );**
- ✓ **JScrollbar (int Orientation);**
- ✓ **JScrollbar (int Orientation, int value, int width, int min, int max);**

JComboBox

- The JComboBox class is used to create the combo box (drop-down list). At a time only one item can be selected from the item list.

Constructors:

- ✓ **JComboBox ( )**
- ✓ **JComboBox (Object[] items)**
- ✓ **JComboBox (Vector<?> items)**

Methods

1	**public void addItem(Object anObject):** is used to add an item to the item list.
2	**public void removeItem(Object anObject):** is used to delete an item to the item list.
3	**public void removeAllItems():** is used to remove all the items from the list.
4	**public void setEditable(boolean b):** is used to determine whether the JComboBox is editable.
5	**public void addActionListener(ActionListener a):** is used to add the ActionListener.
6	**public void addItemListener(ItemListener i):** is used to add the ItemListener.

Example: Combo.java

JList

- JList is a subclass of Jcomponent. It creates a list of items and allows the user to seect one or more item from the list.
- A list is use when the no of selection for any item is large selection of an item can be done by clicking the item.

Constructors:

- ✓ **JList ( );**
- ✓ **JList (vector v);**
- ✓ **JList (Object obj [ ]);**
- ✓ **JList (Listmodel lm);**
- We can create a list for JList by using array of vector or vector there is no method is the JList to add or to remove Item in the list.
- A default list can be created using default list model and we can add item to default list using ***addelement ()***.

Menus

JMenubar

- The JMenubar is a subclass of JComponent. A menubar holds many menus in the bar.
- A menubar must be attached to a JFrame window or JApplet window constructor for this class is ***JMenubar ( )***;

JMenu

- The JMenu is a container of JMenu item a menu can be attach in several menu item

Constructors:

- ✓ **JMenu ( );**
- ✓ **JMenu (StringS);**

JMenuItem

- A JMenuItem is a part of JMenu we can add different menu items to the menu using JMenuItem.

Constructors:

- ✓ **JMenuItem ( );**
- ✓ **JMenuItem (String S);**
- ✓ **JMenuItem (icon i);**
- ✓ **JMenuItem (String S, icon i);**

JTable

- The JTable class is used to display the data on two dimensional tables of cells.

Constructors:

- ✓ **JTable ( ):** creates a table with empty cells.
- ✓ **JTable (Object [ ] [ ] rows, Object [ ] columns):** creates a table with the specified data.

Example: MyTable.java

JColorChooser

- The JColorChooser class is used to create a color chooser dialog box so that user can select any color

Constructors:

- ✓ **JColorChooser ( ):** is used to create a color chooser pane with white color initially.
- ✓ **JColorChooser (Color initialColor):** is used to create a color chooser pane with the specified color initially.

Methods:

- ✓ **public static Color showDialog(Component c, String title, Color initialColor):** is used to show the color-chooser dialog box.

Example: JColorChooserExample.java

JTabbedPane

- A tabbed Pane is a component that appears as a group of folders on a file cabinet. Each folder has a title.
- When a user selects a folder, its contents become visible. Only one of the folders may be selected at a time.
- Tabbed panes are commonly used for setting configuration options.
- Tabbed panes are encapsulated by the JTabbedPane class, and is subclass of JComponent.
- We will use a default constructor. Tabs are defined via the following method:

  Void addTab(String str, Component comp);

  Here, str is the tile of the tab, and comp is the component that should be added to the tab. Typically a JPanel or a subclass of it is added.
- The general procedure to use a tabbed pane in an applet is outlined here :
  - Create a JTabbedPane object.
  - Call ***addTab ( )*** to add a tab to the pane.
  - Repeat step 2 for each tab.
  - Add the tabbed pane to the content pane of the applet.

JProgressBar

- The JProgressBar class is used to display the progress of the task

Constructors:

- ✓ **JProgressBar ( ):** is used to create a horizontal progress bar but no string text.
- ✓ **JProgressBar (int min, int max):** is used to create a horizontal progress bar with the specified minimum and maximum value.
- ✓ **JProgressBar (int orient):** is used to create a progress bar with the specified orientation, it can be either Vertical or Horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL constants.
- ✓ **JProgressBar (int orient, int min, int max):** is used to create a progress bar with the specified orientation, minimum and maximum value.

Methods:

1	**public void setStringPainted(boolean b):** is used to determine whether string should be displayed.
2	**public void setString(String s):** is used to set value to the progress string.
3	**public void setOrientation(int orientation):** is used to set the orientation, it may be either vertical or horizontal by using SwingConstants.VERTICAL and SwingConstants.HORIZONTAL constants..
4	**public void setValue(int value):** is used to set the current value on the progress bar

Example: MyProgress.java

JSlider

- The JSlider is used to create the slider. By using JSlider a user can select a value from a specific range

Constructors:

- ✓ **JSlider ( ):** creates a slider with the initial value of 50 and range of 0 to 100.
- ✓ **JSlider (int orientation):** creates a slider with the specified orientation set by either JSlider.HORIZONTAL or JSlider.VERTICAL with the range 0 to 100 and initial value 50.
- ✓ **JSlider (int min, int max):** creates a horizontal slider using the given min and max.
- ✓ **JSlider (int min, int max, int value):** creates a horizontal slider using the given min, max and value.
- ✓ **JSlider (int orientation, int min, int max, int value):** creates a slider using the given orientation, min, max and value.

Methods:

1	**public void setMinorTickSpacing(int n):** is used to set the minor tick spacing to the slider.
2	**public void setMajorTickSpacing(int n):** is used to set the major tick spacing to the slider.
3	**public void setPaintTicks(boolean b):** is used to determine whether tick marks are painted.
4	**public void setPaintLabels(boolean b):** is used to determine whether labels are painted.
5	**public void setPaintTracks(boolean b):** is used to determine whether track is painted.

Example-1: SliderSimple.java

Example-2: SliderExample.java

Q. Explain Event Delegation Model

- The modern approach to handling events is based on Event Delegation Model which defines standard mechanism to generates and process event.
- It is a standard mechanism for a source to generate an event and send it to the listener.
- Changing the state of an object is known as an event. For example, click on button, dragging mouse etc. The ***java.awt.event*** package provides many event classes and Listener interfaces for event handling.
- The different things about event Delegation model are given below:

Event

- An Event is an object which changes the state where any source is active or inactive. OR An Event is an object that describes a state change in a source.
- This occurs when a user interactive with an element in graphical user interface some of the activities that cause event to be generate by pressing a button, entering a character, selecting an item, etc…

Event Listener

- A listener is an object that is notified when an event occurs.
- It has two major requirements.
  1. It must have registered with the one or more sources to receive notification about specific types of event.
  2. It must implement methods to receive and process those notifications.

Event classes and Listener interfaces:

#	Event Classes	Listener Interfaces
1	ActionEvent	ActionListener
2	MouseEvent	MouseListener and MouseMotionListener
3	MouseWheelEvent	MouseWheelListener
4	KeyEvent	KeyListener
5	ItemEvent	ItemListener
6	TextEvent	TextListener
7	AdjustmentEvent	AdjustmentListener
8	WindowEvent	WindowListener
9	ComponentEvent	ComponentListener
10	ContainerEvent	ContainerListener
11	FocusEvent	FocusListener

✓ **Steps to perform Event Handling**

Following steps are required to perform event handling:

1. Implement the Listener interface and overrides its methods
2. Register the component with the Listener

✓ **Event Handling Codes:**

We can put the event handling code into one of the following places:

1. Same class
2. Other class
3. Anonymous class

How to get the source of an Event

- A source is an object that generates an event. This occurs when the internal state of the object changes in some way.
- Source may generate more than one type of event. A source must register Listeners in order for the Listeners to real modify about a specific type of event.
- We can add an component to the listener using following method

```
public void add<Type>Listener (<Type>Listener object)
```

- A Source must also provide a method that allows a listener to remove component from the listener.
- This can be done by using following method

```
public void remove<Type>Listener (<Type>Listener object)
```

- The different sources which can generate events are Button, Checkbox, Choice, List, Scrollbar, Text Component and window.

For registering the component with the Listener, many classes provide the registration methods.

#	Components	Listener Methods	Description
1	Button	addActionListener (ActionListener a)	Generate Action Event when the button is pressed.
2	TextField	addActionListener (ActionListener a) addTextListener (TextListener a)	Generate Action Event when an text/item is double clicked; Generate Text Events when the user enters a character.
3	TextArea	addTextListener (TextListener a)	Generate Text Events when the user enters a character.
4	Checkbox	addItemListener (ItemListener a)	Generate Item Event when the check box is selected or deselected
5	Choice	addItemListener (ItemListener a)	Generate Item Event when the choice is changed
6	List	addActionListener (ActionListener a) addItemListener (ItemListener a)	Generate Action Event when an item is double clicked; Generate Item Event when the list value is selected or deselected
7	MenuItem	addActionListener (ActionListener a) addItemListener (ItemListener a)	Generate Action Event when a menu item is selected; Generate Item Event when a checkable menu item is selected or deselected
8	Scrollbar	addAdjustmentListener (AdjustmentListener a)	Generate Adjustment Events when the scrollbar is manipulated.
9	Window	addWindowListener (WindowListener a)	Generate window events when a window is activated, closed, deactivated, deiconified, iconified, opened or quit.

Q. *What is an Event Class? Explain in detail*

EventObject

- The event object class extends **Object** which is in ***java.util*** package.
- It is the super class for all events.

Constructor:

✓ **EventObject (Object Obj);** Here *Obj* is the Object that generates this event.

Methods:

getSource ( ) - Its return the source of the event.

toString ( ) - It is returns the string equivalent of the event.

AWTEvent

- The AWTEvent class is a subclass of EventObject and is part of the java.awt package.
- AWTEvent is an abstract class.
- All of the AWT event types are subclass of AWTEvent.

Constructor:

✓ **AWTEvent (Object Source, int id);**

Here source is the object that generates the event and id identifies the type of the event.

Methods:

int getID ( ) - It returns the type of Event.

String toString ( ) - It returns the string equivalent of the event.

- EventObject is super class of all events. AWTEvent is super class of all AWT events that are handled by the event delegation model.
- The package ***java.awt.event*** defines several types of events that are generated by various user interface elements.
- Java provides a different event Classes and event Listeners Interface for the particular event. The following table show this :

#	Event Type	Event Source	Event Listener	Event Handler
1	Action Event/ <class>Adapter	Button List MenuItem TextField	ActionListener	void **actionPerformed** (ActionEvent ae)
2	ItemEvent/ <class>Adapter	Choice Checkbox CheckboxMenuItem List	ItemListener	void **ItemStateChanged** (ItemEvent ie)
3	TextEvent/ <class>Adapter	TextField TextArea	TextListener	void **textChanged** (TextEvent te)
4	KeyEvent/ <class>Adapter	Components (All controls are capable of generating)	KeyListener	void **KeyPressed** (KeyEvent ke) void **KeyReleased** (KeyEvent ke) void **KeyTyped** (KeyEvent ke)
5	FocusEvent/ <class>Adapter	All Component	FocusListener	void **focusGained** (FocusEvent fe) void **focusLost** (FocusEvent fe)
6	Adjustment Event/ <class>Adapter	Scrollbar	AdjustmentListener	void **adjustmentValueChanged** (AdjustmentEvent ae)

7	ComponentEvent/ <class>Adapter	Scroll Bar	ComponentListener	void **componentResized** (ComponentEvent ce) void **componentMoved** (ComponentEvent ce) void **componentShown** (ComponentEvent ce) void **componentHidden** (ComponentEvent ce)
8	ContainerEvent/ <class>Adapter		ContainerListener	void **componentAdded** (ContainerEvent ce) void **componentRemoved** (ContainerEvent ce)
9	MouseEvent/ <class>Adapter	Component	MouseListener	void **mouseClicked** (MouseEvent me) void **mouseEntered** (MouseEvent me) void **mouseExited** (MouseEvent me) void **mousePressed** (MouseEvent me) void **mouseReleased** (MouseEvent me)
			MouseMotionListener	void **mouseMoved** (MouseEvent me) void **mouseDragged** (MouseEvent me)
10	MouseWheelEvent/ <class>Adapter	Mouse Wheel	MouseWheelListener	void **mouseWheelMoved** (MouseWheelEvent me)
11	WindowEvent/ <class>Adapter	Window	WindowListener	void **windowActivated** (WindowEvent we) void **windowClosed** (WindowEvent we) void **windowClosing** (WindowEvent we) void **windowDeactivated** (WindowEvent we) void **windowOpened** (WindowEvent we)
			WindowFocusListener	void **windowGrainedFocus** (WindowEvent we) void **windowLostFocus** (WindowEvent we)

1. ActionEvent

- The Action Event class defines four integer constraints that can be used to identify any modifiers associated with an action event: **ALT_MASK**, **CTRL_MASK**, **SHIFT_MASK** and **META_MASK**.
- In addition, there is an integer Constant **ACTION_PERFORMED**, which can be identifying the action events.

Constructors:

✓ **ActionEvent (Object obj, int type, String cmd)**

✓ **ActionEvent (Object obj, int type, String cmd, int modifiers)**

✓ **ActionEvent (Object obj, int type, String cmd, long when, int modifiers)**

Here, obj is a reference to the object that generated this event.

The type of the event is specified by type, and its command string is cmd.

The modifiers indicate which modifiers keys were pressed when the event was generated.

2. ItemEvent

- There are two types of item events, which are identified by the following integer constants :
  - **DESELECTED -** the user deselected an item
  - **SELECTED** - the user selected an item.

Constructor:

- ✓ **ItemEvent (ItemSelectable obj, int type, Object entry, int state);**

  Here, *obj* is a reference to the object that generated this event.
  The *type* of event specifies by the type.
  The specified item is generated the item event is passed in *entry*.
  The current state of that item is in *state*.

3. TextEvent

- TextEvent defines the integer constant **TEXT_VALUE_CHANGED**

Constructor:

- ✓ **TextEvent(Object obj, int type);**

  Here, *obj* is a reference to the object that generated this event.
  The type of event specifies by the *type*.

4. KeyEvent

- There are 3 types of key events, which are identified by these integer constants **KEY_PRESED**, **KEY_RELEASED** and **KEY_TYPED**.
- The first two events are generated when a key is pressed or released, the last event occurs only when a character is generated.

5. Focus Event

- These event are identified by the integer constants **FOCUS_GAINED** & **FOCUS_LOST**.
- FocusEvent is a subclass of ComponentEvent and has these **Constructors**:
  - ✓ **FocusEvent (Component orj, int type);**
  - ✓ **FocusEvent (Component orj, int type, boolean tempflag);**
  - ✓ **FocusEvent (Component orj, int type, boolean tempflag, Component other);**

  Here, *obj* is a reference to the object that generated this event.
  The type of event specifies by the *type*.
  The argument *tempflag* is set to true if the focus event is temporary otherwise it is set to false.

6. Adjustment Event

- Three are five types of adjustment events. The adjustmentEvent class defines integer constants that can be used to identify them. The constant are:
  - **BLOCK_DECREMENT:** The User clicked inside the scroll bar to decrease its value.
  - **BLOCK_INCREMENT:** The User clicked inside the scroll bar to increase its value.
  - **TRACK:** The slide was dragged
  - **UNIT_DECREMENT:** The button at the end of the scroll bar was clicked to decrease its value.
  - **UNIT_INCREMENT:** The button at the end of the scroll bar was clicked to increase its value.
- There is an integer constant, **ADJUSTMENT_VALUE_CHANGED**, which indicates that a change has occurred.

Constructor:

- ✓ **AdjustmentEvent(Adjustable obj, int id, int type, int data);**

  Here, *obj* is a reference to the object that generated this event.
  The *id* equals ADJUSTMENT_VALUE_CHANGED.
  The type of the event is specified by *type* & its associated data is *data*.

Methods:

- **Adjustable getAdjustable ( )** - This method returns the object that generated the event.
- **int getAdjustmentType ( )** - This method returns one of the constants defined by AdjustmentEvent.
- **int getValue ( )** - The amount of the adjustment can be obtained from this method.

7. Component Event

- The component event class defines integer constants that can be used to identify them. The constants and their meaning are :
  - **COMPONENT_HIDDEN:** The component has hidden
  - **COMPONENT_MOVED:** The component has moved
  - **COMPONENT_RESIZED:** The component has resized
  - **COMPONENT_SHOWN:** The component became visible

Constructor:

- ✓ **ComponentEvent(component obj, int type);**

  Here, *obj* is a reference to the object that generated this event.
  The type of event specifies by the *type*.
- Component Event is the super class wither directly or indirectly of ContainerEvent, FocusEvent, KeyEvent, MouseEvent &WindowEvent.

Method:

- **Component getComponent ( )** - it returns the component that generated the event.

8. Container Event

- Container Event is a sub class of component event.
- It is generated when q component is added to or removed from a container.
- There are two types of container events **COMPONENT_ADDED** and **COMPONENT_REMOVED**

Constructor:

- ✓ **ContainerEvent (Component obj, int type, Component comp);**

  Here, *obj* is a reference to the object that generated this event.
  The type of event specifies by the *type* and the component that has been added to or removed from the container is *comp*.

9. Mouse Event

- Mouse event is subclass of InputEvent.
- There are eight types of mouse events.

- The MouseEvent class defines the following integer constants that can be used to identify them :
  - **MOUSE_CLICKED** : User clicked the mouse
  - **MOUSE_DRAGGED** : User dragged the mouse
  - **MOUSE_ENTERED** : User entered a component
  - **MOUSE_EXITED** : User exited from a component
  - **MOUSE_MOVED** : Mouse moved
  - **MOUSE_PRESSED** : Mouse was pressed
  - **MOUSE_RELEASED** : Mouse was released
  - **MOUSE_WHEEL** : Mouse Wheel was moved

Constructor:

✓ **MouseEvent (Componet Obj, int type, long when, int modifiers, int x, int y, int clicks, boolean triggersPopup);**

Here, *obj* is a reference to the component that generated this event & the type of event specified *type*.

The system time at which the mouse event occurred is passed in *when*.

The coordinates of the mouse are passed in *x* & *y*.

The *triggersPopup* flag indicates if this event causes a pop-up menu to appear on this platform.

Methods:

- **int getX(); & int getY();** - The most commonly used method in this class are **getX ( )** and **getY ( )**. These return the X & Y coordinates of the mouse when the event occurred.
- **Point getPoint ( )** - It returns a Point object that contains the X, Y coordinates in its integer members: X & Y.
- **translatePoint ( )** - Its changes the location of the event.
- **int getClickCount ( )** - This method obtains the number of mouse clicks for this event.

10. Window Event

- Window Event is a subclass of component Event. There are ten types of window events :

Constructors:

✓ **WindowEvent (window obj, int type);**

✓ **WindowEvent (window obj, int type, Window other);**

✓ **WindowEvent (window obj, int type, int fromstate, int tostate);**

✓ **WindowEvent (window obj, int type, Window other, int fromstate, int tostate);**

Here, *obj* is a reference to the object that generated this event.

The type of event specifies by the *type*.

other specifies the opposite window when a focus event occurs.

The *fromstate* specifies the prior state of the window.

tostate specifies the new state that the window will have when a window state change occurs.

Method:

- **getWindow ( )** - returns the Window object that generated the event.

Q. Explain Adapter Class

- When we implements any interface in our program, we must implements all the method declared in the interface.
- Example: A program use MouseListener interface & only one handler method MouseClicked() is used in the program.
- However we have to overwrite all the methods of the listener interface. This can be done by writing the method name and empty braces (like ()).
- This is not a convenient way the solution of this problem is to use Adapter classes.
- An adapter class provides an empty implementation of all the methods a particular listener interface.
- The following table shows the adapter classes available in ***java.awt.event*** package.

No.	Adapter Class	Listener
1	ComponentAdapter	ComponentListener
2	ContainerAdapter	ContainerListener
3	FocusAdapter	FocusListener
4	KeyAdapter	KeyListener
5	MouseAdapter	MouseListener
6	MouseMotionAdapter	MouseMotionListener
7	WindowAdapter	WindowListener

Example :

```
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
public class MouseAdapterDemo extends Applet
{
      public void init()
      {     setBackground(Color.green);
            addMouseListener(new MyMouseAdapter(this));
      }
}
class MyMouseAdapter extends MouseAdapter
{
      MouseAdapterDemo mad;
      public MyMouseAdapter (MouseAdapterDemo mad)
      {     this.mad = mad;    }
      public void mousePressed(MouseEvent me)
      {     mad.setBackground (Color.red);
            mad.repaint();
      }
      public void mouseReleased (MouseEvent me)
      {     mad.setBackground (Color.green);
            mad.repaint();
      }
}
```

Inner Classes

- We can declare and defines one class the definition of another class such class are called nested classes.
- The scope of nested class is bounded by the scope of its enclosing class. Thus if class B is defined in Class A then B is known to A. Thus the nested class B has access to all type the members including private members of class in which it is nested (class A)
- There are two way of nested classes :

  1) **Static**
     - A static nested class is a class with static modifier. It must access the member of its enclosing class through an object, because of this reason it is rarely used.
  2) **Non Static Nested class (Inner Class)**
     - An inner class is non static nested class. It has access to all the variables and methods of its outer class and it can be accessed directly.

Anonymous Inner Class

- An anonymous inner class is an inner class that does not have the name the syntax to define an anonymous inner class is given below :

```
new className ()
{
      ----
}
```

- Here class name is the name of the super class for our anonymous inner class. The following example shows the general form of an applet that uses an anonymous inner class as an adapter for mouse event.
- This init() method of the applet instantiates an anonymous adapter class an register that object to receive mouseListener event.

	Network Administration	TOPIC	MARKS	TOTAL LECT.
UNIT 1	Basics of Network	Network concepts -What is network - Use of network Network model -peer – to – peer -client – server Network Services -File service, -Print service, -Comm. service, -Data base service, -Security service, -Application service Network Access Methods 1)csma / cd, csma / ca, 2)Token passing 3)Polling Network Topologies Bus, Ring, Star, Mesh, Hybrid Additional Advanced Network Topologies Ethernet,CDDI,FDDI Communication Methods 1. Unicasting 2. Multicasting 3. Broadcasting	14	7
	Network Models and LAN Sharing	OSI reference model with 7 layers TCP/IP network model with 4 layers File And Print Sharing in LAN. Mapping of network drive Disk quota Encryption Compression Net meeting		5

UNIT 2	Transmission Media	Transmission Media -Types of Transmission media 1)Guided media -Co – Axial Cable, -Twisted Pair Cable, -Crimping of Twisted pair cable -Fiber Optic Cable Unguided media -Infrared, Laser, Radio, Microwave, Bluetooth tech.	14	6
	Multiplexing & Switching Concepts	Different Frequency Ranges Multiplexing & Demultiplexing Multiplexing Types FDM, TDM, CDM, WDM Switching Tech. 1)Circuit Switching, 2)Message Switching, 3)Packet Switching		6
	Network devices	CABLE NETWORK DEVICES LAYER1 DEVICES -LAN CARD, -MODEM , -DSL & ADSL -HUB(Active,Passive,Smart hub) -REPEATER LAYER2 DEVICES -SWITCH(Manageable, nonmanagable) -BRIDGE(Source route, Transactional) LAYER3 DEVICES -ROUTER -BROUTER -GATEWAY WIRELESS NETWORK Wireless switching and routing concepts		8
UNIT 3	Network Protocols	Packets &Protocols Conn. Oriented protocols -TCP&	14	5

		connection less protocols-UDP TCP/IP STACK -HTTP, -FTP -SMTP, -POP3 -SNMP, -TELNET -ARP , -RARP IPX/SPX AppleTalk, NetBIOS Name PROTOCOL L2CAP, RFCOMM Protocol		
	Network Routing	What is routing Requirements of routing Types of Routing -static -dynamic -default Routing protocols --Exterior Routing protocol- 1)BGP --Interior Routing protocol 1)Distance vector routing -RIP,-IGRP,-EIGRP 2)Link state routing -OSPF,-IS IS		6
UNIT 4	IP ADDRESSING	What is ip address? Types of ip address 1) ipv4 Class structure subneting,supernetting 2) ipv6 Basic structure of ipv6 Implementation of ipv6	14	4
	Windows 2008 server	Installation of 2008 enterprise server Various editions of windows 2008 server Installation & Configuration of Active Directory -Domains, Trees, Forests concept Accounts(User, Group, Computer) Policy (Security and audit) Logging Events MMC(Microsoft Management console)		8

UNIT 5	Basics of Network Security	Fundamental of Network Security Policies, Standard, Procedures, Baselines, Guidelines Security methods - Encryption - Cryptography - Authentication Security Principle –CIA Model	14	7
	Internet connection & Sharing	Basics of Internat • How internat is connect with computer • Technology related internat - Dial up tech. - ISDN network tech. - Lease line tech. VPN -Types of VPN -VPN protocols(PPTP, L2TP, IPsec.) • Proxy server, Firewall • GPS,GPRS • CCTV tech.		8
			70 marks	70lect.

UNIT 1

Chapter 1: Introduction

❖ What is Network? :

A network is the most cost-effective way to share a collection of communication equipment such as PC's, servers, printers, and modems that has been connected together by cables. A network helps people work collectively, not just individually.

❖ Network Concept :

Networking is the concept of sharing resources and services.
A network of computers is a group of interconnected systems sharing resources and interacting using a shared communications link

All networks must have the following:

- A resource to share (resource)
- A pathway to transfer data (transmission medium)
- A set of rules governing how to communicate (protocols)

Having a transmission pathway does not always guarantee communication. When two entities communicate, they do not merely exchange information; rather, they must understand the information they receive from each other. The goal of computer networking, therefore, is not simply to exchange data but to understand and use data received from other entities on the network.

The two main reasons for using computer networking are to provide services and to reduce equipment costs. Networks enable computers to share their resources by offering services to other computers and users on a network.

The following are specific reasons for networking PCs:

- Sharing files
- Sharing printers and other devices
- Enabling centralized administration and security of the resources within the system
- Supporting network applications such as electronic mail and database services

❖ Types of Networks :

- LAN – Local Area Network
- MAN – Metropolitan Area Network
- WAN – Wide Area Network

❖ LAN (Local Area Network)

- A local area network may serve as few as two or three users (for example, in a home network) or many as thousands of users.
- Local-area networks (LANs) evolved around the PC revolution. LANs enabled multiple users in a relatively small geographical area to exchange files and messages, as well as to access shared resources such as file servers and printers.
- Is a group of computers and associated device that share a common communications line or wireless link and typically share the resources of a single processor or server within a small geographic area (for example, within an College Building).

LANs are characterized by the following:

- They transfer data at high speeds (higher bandwidth).
- They exist in a limited geographical area.
- Connectivity and resources, especially the transmission media, usually are managed by the company running the LAN.

❖ MAN - (Metropolitan-area networks)

- A MAN usually consists of two or more LANs in a common geographic area. Typically, a service provider is used to connect two or more LAN sites using private communication lines or optical services. A MAN can also be created using wireless bridge technology by beaming signals across public areas.

❖ WAN – (Wide Area Network)

- WANs interconnect LANs, which then provide access to computers or file servers in other locations. Because WANs connect user networks over a large geographical area, they make it possible for businesses to communicate across great distances. WANs allow computers, printers, and other devices on a LAN to be shared with distant locations.

WANs are characterized by the following:

- They exist in an unlimited geographical area.
- They usually interconnect multiple LANs.
- They often transfer data at lower speeds (lower bandwidth).

- Connectivity and resources, especially the transmission media, usually are managed by a third-party carrier such as a telephone or cable company.

❖ **Network Model :**

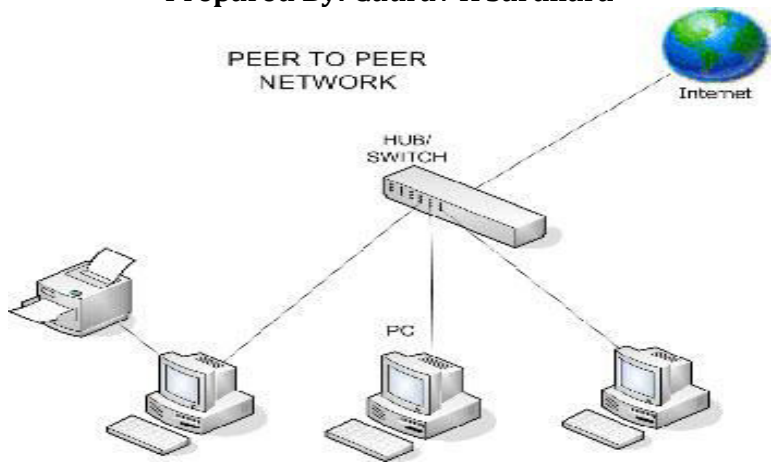
- Networks are all about sharing resources of computers, servers, printers, scanners etc to each other. There are two different types with which network between computers can be formed. Networking formation completely depends on the requirement of the organization scale and usability. We should always study requirements and needs before we decide to choose any type of network. Picking wrong set of options can cost you waste of money, time and resources.

- Two types of networks are Peer to Peer networking also known to be p2p, the other one is Client and Server networks.

- (1) Peer to Peer (2) Client Server

(1) Peer to Peer (p2p) Model:

- P2p networking type is most commonly used computer networks. This type of network is very cost effective but supports lesser number of computers in network. Ten to fifteen computers can be connected to each other using p2p networking model without problem, more number of computers often create problems.
- All computers possesses same status within the network and no computer control any other computer but it self, this network does not have server to control and monitor.
- Security level is not towards higher side and each work station it self is responsible for security. Using p2p models files can be shared among computers. Files like, videos, audios, pictures, spreadsheets and all digital media can be sent or received with in the network.
- Printers, scanners and internet can be shared with in all computers.
- Below is the picture showing three computers connected to each other with hub and switch. All computers are connected to hub through Network adaptor card using Cable and hub or switch is connected to internet to pass it on to connected computers.. You can see there is no server involved in this diagram but all individual computers are connecting to hub forming P2P network.



Limitation of P2P networking model:

Before deciding to implement P2P model one must know the limitations of this type. Getting to know later can be frustrating big time. It would highly be recommended to get your organizational people site together and discuss the needs. Peer to Peer looks very simple, quite cost effective and attractive, yet it can keep progress very limited.

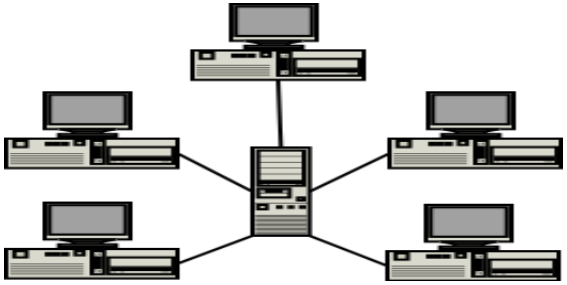
- Peer-To-Peer networks are designed for limited number computers, it will start creating issues when exceed 15 number of computers
- High security levels can not be achieved using p2p networks, so if organization have concerns with security p2p will not be that great.
- Organizational growth will outgrow p2p networks; it will not support growing number of computers when increased above fifteen.
- Regular training is required for computer users of p2p network. p2p network is control by computers and computers are controlled by human, small mistake by one of the user can hold the work for other users on same p2p network.

(2) Client Server Model:

- Choosing right kind of networking model is very important for organization. If you are using lesser number of computer and do not see any need to increase the numbers of computers to more than 15 then you are fine with peer to peer networking model, but if you are bigger organization or seeing growth in network, client and server model is designed for it.

- The difference in p2p and client server model is that p2p does not have any device or computer that controls computers on network whereas; client / server model has one dedicated computer which is called server. It is called dedicated server. All computers are connected to hub and hub is connected to dedicated server.
- Server is responsible to perform according to the request sent to it by clients. For example server can act as print server, if client request a print of document server will send print command to printer and it will be printed.
- Same way all the files are stored on the server and not on client computer, same client can retrieve data by using any other computer on the same network. This concept is known as centralization, this enables server to keep profile of users, data, and software etc completely in tacked and organized.
- Normal computer can also be configured as server and it should be alright and perform server tasks efficiently, but if network growth is on seen and many computers are required to attach to network that's where we might need proper server to take over the network.
- You can see in diagram below. All the workstations (Clients) are attached on server, some times there is hub involved but in this case it is just clients and server.

Client Server model



❖ **Peer-to-Peer versus Client/ Server Networks :**

Advantages

Peer-to-Peer Networks

- No dedicated server.
- Less expensive.
- Easy to install and maintain.
- Good file, printer, and CD-ROM sharing.

Client/Server Networks

- Fast.
- Expandable.
- Will work with any application.
- Handles shared database applications.
- More reliable (dedicated server).
- Highest level of security.

Disadvantages

Peer-to-Peer Networks

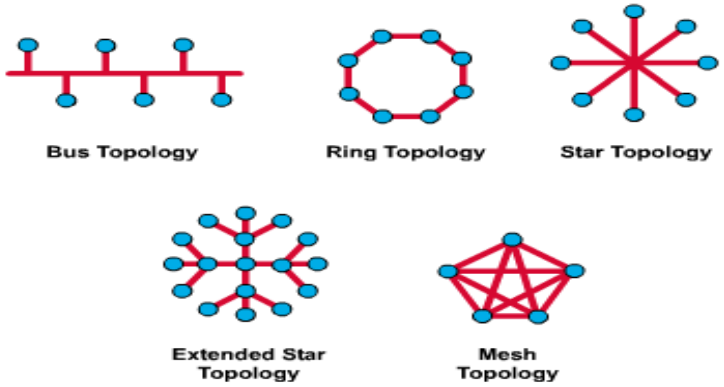
- Slow.
- No good for database applications.
- Less reliable (server is workstation).
- Limited expandability.

Client/Server

- Needs dedicated server.
- More expensive to buy.
- More expensive to maintain.

➢ **Network Topology:**

- The network topology defines the way in which computers, printers, and other devices are connected. A network topology describes the layout of the wire and devices as well as the paths used by data transmissions.



➢ **Bus Topology:**

- All devices are connected to a central cable, called the bus or backbone.
- Bus networks are relatively inexpensive and easy to install for small networks.
- Ethernet systems use a bus topology. In Simple way we can say us (All devices share a common wire to transmit and receive data through using an arbitration method.)
- Commonly referred to as a linear bus, all the devices on a bus topology are connected by one single cable.

Advantages:

- It is easy to handle and implement.
- It is best suited for small networks

Disadvantages:

- The cable length is limited. This limits the number of stations that can be connected.
- This network topology can perform well only for a limited number of nodes.

➢ **Star & Tree Topology :**

- All devices are connected to a central hub.
- Star networks are relatively easy to install and manage, but bottlenecks can occur because all data must pass through the hub.
- This is not much of a problem anymore with the widespread deployment of switches.
- At the central point we usually see a device generically called a hub or switch.
- The star topology is the most commonly used architecture in Ethernet LANs.
- When installed, the star topology resembles spokes in a bicycle wheel.
- Larger networks use the extended star topology also called tree topology.
- This topology significantly reduces the traffic on the wires by sending packets only to the wires of the destination host.

Advantages:

- Due to its centralized nature, the topology offers simplicity of operation.
- It also achieves an isolation of each device in the network.

Disadvantages:

- The network operation depends on the functioning of the central hub. Hence, the failure of the central hub leads to the failure of the entire network.

➢ **Ring topology:**

- All devices are connected to one another in the shape of a closed loop, so that each device is connected directly to two other devices, one on either side of it.
- Ring topologies are relatively expensive and difficult to install, but they offer high bandwidth and can span large distances.
- A ring is created to which each device attaches. A special signal, called a token travels around the ring letting it know that it is that device's turn to transmit.

Advantages:

- The data being transmitted between two nodes passes through all the intermediate nodes. A central server is not required for the management of this topology.

Disadvantages:

- The failure of a single node of the network can cause the entire network to fail.
- The movement or changes made to network nodes affects the performance of the entire network.

➢ **Mesh Topology :**

- The mesh topology connects all devices (nodes) to each other for redundancy and fault tolerance.
- It is used in WANs to interconnect LANs and for mission critical networks like those used by banks and financial institutions.
- Implementing the mesh topology is expensive and difficult.

Advantage:

- The arrangement of the network nodes is such that it is possible to transmit data from one node to many other nodes at the same time.

Disadvantage:

- The arrangement wherein every network node is connected to every other node of the network, many of the connections serve no major

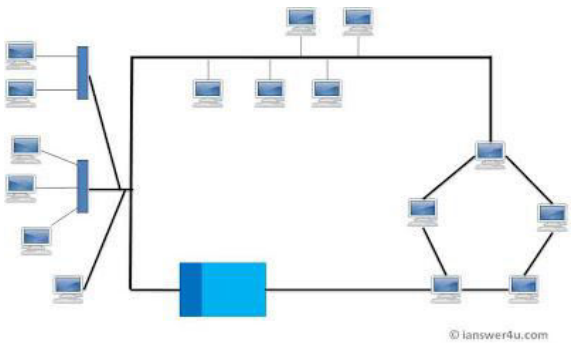
purpose. This leads to the redundancy of many of the network connections.

❖ **What is Hybrid Topology?**

Before starting about Hybrid topology, we saw that a network topology is a connection of various links and nodes, communicating with each other for transfer of data. We also saw various advantages and disadvantages of Star, Bus, Ring, Mesh and Tree topologies.

Now lets discuss what Hybrid Network topology is and why it finds its application in Wide Area Networks. Hybrid, as the name suggests, is mixture of two different things. Similarly in this type of topology we integrate two or more different topologies to form a resultant topology which has good points(as well as weaknesses) of all the constituent basic topologies rather than having characteristics of one specific topology. This combination of topologies is done according to the requirements of the organization.

For example, if there exists a ring topology in one office department while a bus topology in another department, connecting these two will result in Hybrid topology. Remember connecting two similar topologies cannot be termed as Hybrid topology. Star-Ring and Star-Bus networks are most common examples of hybrid network.



Hybrid Network Topology Image

Advantages of Hybrid Network Topology

1) **Reliable:** Unlike other networks, fault detection and troubleshooting is easy in this type of topology. The part in which fault is detected can be isolated from the rest of network and required corrective measures can be taken, WITHOUT affecting the functioning of rest of the network.
2) **Scalable:** Its easy to increase the size of network by adding new components, without disturbing existing architecture.
3) **Flexible:** Hybrid Network can be designed according to the requirements of the organization and by optimizing the available resources. Special care can be given to nodes where traffic is high as well as where chances of fault are high.
4) **Effective:** Hybrid topology is the combination of two or more topologies, so we can design it in such a way that strengths of constituent topologies are maximized while there weaknesses are neutralized. For example we saw Ring Topology has good data reliability (achieved by use of tokens) and Star topology has high tolerance capability (as each node is not directly connected to other but through central device), so these two can be used effectively in hybrid star-ring topology.

- **Disadvantages of Hybrid Topology**

1) **Complexity of Design:** One of the biggest drawback of hybrid topology is its design. Its not easy to design this type of architecture and its a tough job for designers. Configuration and installation process needs to be very efficient.
2) **Costly Hub:** The hubs used to connect two distinct networks, are very expensive. These hubs are different from usual hubs as they need to be intelligent enough to work with different architectures and should be function even if a part of network is down.
3) **Costly Infrastructure:** As hybrid architectures are usually larger in scale, they require a lot of cables, cooling systems, sophisticate network devices, etc.

ADDITIONAL TOPOLOGIES:-

1- ETHERNET TOPOLOGIES
2- CDDI
3- FDDI

❖ Ethernet TOPOLOGY:-

→ Key features of Ethernet topologies:

- Fast, reliable throughput speed-10 Mbps.
- Accurate transmission-CSMA/CD access method.

- Easy compatibility-more LAN components match Ethernet standards than any other.
- Maximum flexibility-two topologies (bus or star) and five kinds of cable (standard or Thin coax; unshielded twisted pair; FOIRL or 10BASE-FL fibre optic).

❖ Overview:
Ethernet is the most widely used network topology. You can choose between bus and star topologies, as well as coax, twisted-pair, or fibre optic cabling. And with the right connective equipment, multiple Ethernet-based LANs can be linked together. In fact, with the right equipment and software, even Token Ring, AppleTalk®, and wireless LANs can be connected to Ethernet. Ethernet uses CSMA/CD (Carrier Sense Multiple Access with Collision Detection).
In this method, multiple workstations access a transmission medium (multiple access) by listening until no signals are detected (carrier sense). Then they transmit and check to see if more than one signal is present (collision detection). Each station attempts to transmit when it "believes" the network is free. If there is a collision, each station attempts to retransmit after a preset delay, which is different for each workstation. When a collision is detected, a "jam" signal is propagated to all nodes. Each station that detects the collision will wait some period of time and then try again.

→ The two possible topologies for Ethernet are bus and star.
The bus is the simplest (and the traditional) topology. Standard Ethernet (10BASE5) and Thin Ethernet (10BASE2), both based on coax cable systems, use the bus. In this one-cable LAN, all workstations are connected in succession (a "bus" arrangement) on a single cable. All transmissions go to all the connected workstations.
Each workstation then selects the transmissions it should receive based on the address information contained in the transmission.

→ In a star topology, all attached workstations are wired directly to a central hub that establishes, maintains, and breaks connections between them (in the event of an error). The advantage of a star topology is that it's easy to isolate a problem node. The disadvantage is that if the hub fails, the entire system is compromised. Twisted-Pair Ethernet (10BASE-T), based on unshielded twisted pair, and Fibre Optic Ethernet (FOIRL and 10BASE-FL), based on fibre optic cable, use the star.

→ Typical applications.
Use a bus topology for a large network with many users and longer segments. With repeaters or media converters, you can easily interconnect to other networks with different topologies.

→ Use a star topology when you want to use twisted-pair cabling (10BASE-T Ethernet) for a multiple-building campus setup (you might already have twisted pair-telephone wire-installed on your premises). Use a star topology for your fiber optic links.

❖ **CDDI** :-

→ Copper Distributed Data Interface (CDDI) is a standard for data transmission in a local area network. It uses optical fiber as its standard underlying physical medium, although it was also later specified to use copper cable, in which case it may be called FDDI.

→ Description

→ CDDI provides a 100 Mbit/s twisted pair standard for data transmission in local area network that can extend in range up to 200 kilometers (120 mi). Although CDDI logical topology is a ring-based token network, it did not use the IEEE 802.5 token ring protocol as its basis; instead, its protocol was derived from the IEEE 802.4 token bus timed token protocol. In addition to covering large geographical areas, CDDI local area networks can support thousands of users. CDDI offers both a Dual-Attached Station (DAS), counter-rotating token ring topology and a Single-Attached Station (SAS), token bus passing ring topology.

→ A FDDI network contains two rings, one as a secondary backup in case the primary ring fails. The primary ring offers up to 100 Mbit/s capacity. When a network has no requirement for the secondary ring to do backup, it can also carry data, extending capacity to 200 Mbit/s. The single ring can extend the maximum distance; a dual ring can extend 100 km (62 mi). CDDI had a small frame size than the standard Ethernet family, which only supports a maximum-frame size of 1,500 bytes, allowing better effective data rates in some cases.

❖ **FDDI :-**

→ **Fiber Distributed Data Interface** (**FDDI**) is a standard for data transmission in a local area network. It uses optical fiber as its standard underlying physical medium, although it was also later specified to use copper cable, in which case it may be called **CDDI** (Copper Distributed Data Interface), standardized as **TP-PMD** (Twisted-Pair Physical Medium-Dependent), also referred to as TP-DDI (Twisted-Pair Distributed Data Interface).

→ **Description**

→ FDDI provides a 100 Mbit/s optical standard for data transmission in local area network that can extend in range up to 200 kilometers (120 mi). Although FDDI logical topology is a ring-based token network, it did not use the IEEE 802.5 token ring protocol as its basis; instead, its protocol was derived from the IEEE 802.4 token bus timed token protocol. In addition to covering large geographical areas, FDDI local area networks can support thousands of users. FDDI offers both a Dual-Attached Station (DAS), counter-rotating token ring topology and a Single-Attached Station (SAS), token bus passing ring topology.

→ FDDI, as a product of American National Standards Institute X3T9.5 (now X3T12), conforms to the Open Systems Interconnection (OSI) model of functional layering using other protocols. The standards process started in the mid 1980s.

→ DDI-II, a version of FDDI described in 1989, added circuit-switched service capability to the network so that it could also handle voice and video signals. Work started to connect FDDI networks to synchronous optical networking (SONET) technology.

→ A FDDI network contains two rings, one as a secondary backup in case the primary ring fails. The primary ring offers up to 100 Mbit/s capacity. When a network has no requirement for the secondary ring to do backup, it can also carry data, extending capacity to 200 Mbit/s. The single ring can extend the maximum distance; a dual ring can extend 100 km (62 mi). FDDI had a larger maximum-frame size (4,352 bytes) than the standard Ethernet family, which only supports a maximum-frame size of 1,500 bytes, allowing better effective data rates in some cases.

❖ **Network Services:**

- **File services :**

File services enable networked computers to share files with each other. This capability was one of the primary reasons networking of personal computers initially came about. File services include all network functions dealing with the storage, retrieval, or movement of data files. File services enable users to read, write, and manage files and data. This includes moving files between computers and archiving files and data.

File services are an important part of client/server and peer-to-peer networks. Computers providing files services are referred to as file servers.

Two types of servers exist:
Dedicated and non-dedicated. Dedicated servers do nothing but fulfill requests to network clients. These servers commonly are found in client/server environments. Non-dedicated servers do double duty. They enable a user to go onto the machine acting as a file server and request the use of files from other machines; at the same time, they give files to users who request them from other computers on the network.

- **Printing Services :**

After file services, printing is probably the second biggest incentive for installing a LAN.

The following are some of the many advantages of network print services:

- Many users can share the same printers. This capability is especially useful with expensive devices such as color printers and plotters.
- Printers can be located anywhere, not just next to a user's PC.
- Queue-based network printing is more efficient than direct printing because the workstation can begin to work again as soon as a job is queued to the network.
- Modern printing services enable users to send facsimile (fax) transmissions through the network to a fax server.

- **Application Services :**
  - Application services enable organizations to install servers that are specialized for specific functions
  - Some of the more common application servers are database servers, messaging/communication servers, groupware servers, and directory servers. Application servers are an effective strategy for making a network more scalable.
  - Additional application servers can be added as new application needs emerge.
  - If more power is necessary for an application, only the application server needs to be upgraded.
  - Application services enable applications to leverage the computing power and specialized capabilities of other computers on a network.

- **Database Services :**
  - Database servers are the most common type of application servers.
  - Because database services enable applications to be designed in separate client and server components, such applications frequently are called client/server databases.
  - With a client/server database, the client and server applications are designed to take advantage of the specialized capabilities of client and database systems,
  - The client application manages data input from the user, generation of screen displays, some of the reporting, and dataretrieval requests sent to the database server.
  - The database server manages the database files; adds, deletes, and modifies records in the database; queries the database and generates the results required by the client; and transmits results back to the client.
  - The database server can service requests for multiple clients at the same time.

A modern database server is a sophisticated piece of software that can perform the following functions:

- Provide database security.
- Optimize the performance of database operations.
- Determine optimum locations for storing data without requiring clients to know where the data is located.
- Service large numbers of clients by reducing the amount of time any one client spends accessing the database.
- Distribute data across multiple database servers.

- **Messaging/Communication Services :**

Messaging/communication services generally transfer information from one place to another.

This communication of information can be broken down into three subareas:

- Email
- Voice mail
- Fax services

Email:
Email systems can service any size group from a local workgroup to a corporation to the world. By installing email routing devices, you can transfer mail smoothly and efficiently among several LANs.
Email also can be routed to and received from the Internet. This enables users in dozens of countries throughout the world to exchange electronic messages.
Early text-based email has given way to elaborate systems that support embedded sound, graphics, and even video data.
Some of the major email packages include Microsoft's Exchange Server, Novell's GroupWise, and Lotus Notes.

Voice Mail:
Voice mail enables you to connect your computer to a telephone system and to incorporate telephone voicemail messages with your PC. The technical term for this is *telephony*. This often involves moving
Your voicemail messages from the phone system to the LAN and enabling the computer network to distribute this information to different clients.

Fax Services:
Fax services enable you to send or receive faxes from your computer. This is similar to printing in that your can "print" the document to a fax device. Fax services, however, can take on more complicated

Features including the capability to send faxes to a central fax server and to receive faxes from the phone system to a central fax device. That device then delivers the fax message to your PC. This all occurs Automatically.

- **Security Services :**

- Another service provided by networks is security. Security is one of the most important elements involved
  In a network.
- When users share resources and data on a network, they should be able to control who can access the
  Data or resource and what the user can do with it. An example of this is a file showing the financial records of a company. If this file is on a file server, it is important to be able to control who has access to the file.
- Security services often deal with a user account database or something like the aforementioned directory services. This database of users often contains a list of names and passwords.
  When a person wants to access the network, he must log on to the network. Logging on is similar to trying to enter an office building with a security guard at the front door. Before you can enter the building, you must verify who you are against a list of people who are allowed access.

  Security services often are intermingled with other services. Some services added to a network can utilize the security services of the system onto which they have been installed. An example of this is Microsoft Exchange Server. This messaging product can utilize the security services of an existing Windows NT Server.

❖ **Network Access Method :**

- A network of computers based on multi-access medium requires a protocol for effective sharing of the media. As only one node can send or transmit signal at a time using the broadcast mode, the main problem here is how different nodes get control of the medium to send data, that is "who goes next?". The protocols used for this purpose are known as Medium Access Control (MAC) techniques

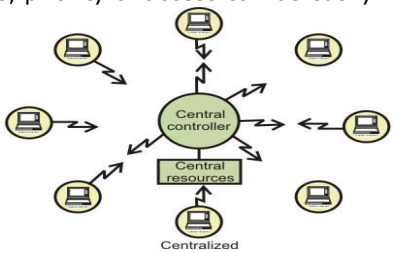
❖ **Round Robin Techniques:**

- In Round Robin techniques, each and every node is given the chance to send or transmit by rotation. When a node gets its turn to send, it may either decline to send, if it has no data or may send if it has got

data to send. After getting the opportunity to send, it must relinquish its turn after some maximum period of time. The right to send then passes to the next node based on a predetermined logical sequence. The right to send may be controlled in a centralized or distributed manner. Polling is an example of centralized control and token passing is an example of distributed control as discussed below.

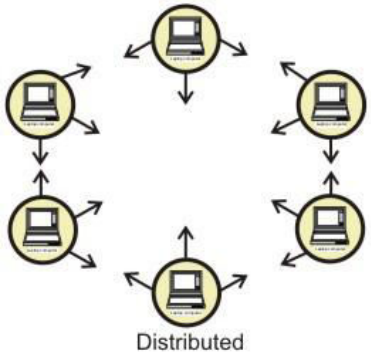
(1) Polling :

- The mechanism of polling is similar to the roll-call performed in a classroom. Just like the teacher, a controller sends a message to each node in turn.
- The message contains the address of the node being selected for granting access.
- Although all nodes receive the message, only the addressed node responds and then it sends data, if any. If there is no data, usually a "poll reject" message is sent back. In this way, each node is interrogated in a round-robin fashion, one after the other, for granting access to the medium.
- The first node is again polled when the controller finishes with the remaining codes.
- The polling scheme has the flexibility of either giving equal access to all the nodes, or some nodes may be given higher priority than others. In other words, priority of access can be easily implemented.



Polling using a central controller

- Polling can also be accomplished without a central controller. Here, all stations receive signals from other stations as shown in Fig. Below. Stations develop a polling order list, using some protocol.

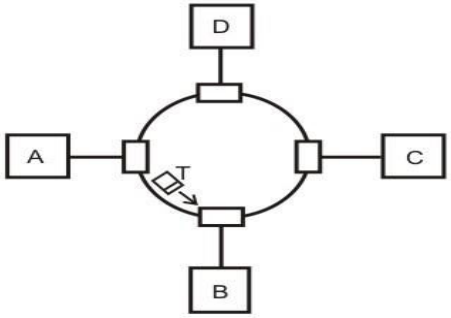


Polling in a distributed manner

(2)**Token Passing**:

- In token passing scheme, all stations are logically connected in the form of a ring and control of the access to the medium is performed using a token. A token is a special bit pattern or a small packet, usually several bits in length, which circulate from node to node. Token passing can be used with both broadcast (token bus) and sequentially connected (token ring) type of networks.
- In case of token ring, token is passed from a node to the physically adjacent node. On the other hand, in the token bus, token is passed with the help of the address of the nodes, which form a logical ring. In either case a node currently holding the token has the 'right to transmit'. When it has got data to send, it removes the token and transmits the data and then forwards the token to the next logical or physical node in the ring. If a node currently holding the token has no data to send, it simply forwards the token to the next node. The token passing scheme is efficient compared to the polling technique, but it relies on the correct and reliable operation of all the nodes. There exists a number of potential problems, such as lost token, duplicate token, and insertion of a node, removal of a node, which must be tackled for correct and reliable operation of this scheme.

- Performance of a token ring network can be represented by two parameters; throughput, which is a measure of the successful traffic, and delay, which is a measure of time between when a packet is ready and when it is delivered.



Token ring network

❖ **Contention-based Approaches**

- Round-Robin techniques work efficiently when majority of the stations have data to send most of the time. But, in situations where only a few nodes have data to send for brief periods of time, Round-Robin techniques are unsuitable.
- Contention techniques are suitable for bursty nature of traffic. In contention techniques, there is no centralized control and when a node has data to send, it contends for gaining control of the medium.
- The principle advantage of contention techniques is their simplicity.
- They can be easily implemented in each node.
- The techniques work efficiently under light to moderate load,
- But performance rapidly falls under heavy load.

(2) Carrier Sense Multiple Access (CSMA):

Procedure

- Listen to medium and wait until it is free (no one else is talking)
- Wait a random back off time then start talking

Advantages

- Fairly simple to implement
- Functional scheme that works

Disadvantages

- Can not recover from a collision
  (Inefficient waste of medium time)

(3) Carrier Sense Multiple Access with Collision Detection (CSMA-CD):

- This refined scheme is known as Carrier Sensed Multiple Access with Collision Detection (CSMA/CD) or Listen-While-Talk.
- CSMA/CD protocol can be considered as a refinement over the CSMA scheme.

Procedure

- Listen to medium and wait until it is free
- Then start talking, but listen to see if someone else starts talking too
- If a collision occurs, stop and then start talking after a random back off time
- This scheme is used for hub based Ethernet

Advantages

- More efficient than basic CSMA

Disadvantages

- Requires ability to detect collisions.

(4) Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA) :

Procedure

- Similar to CSMA but instead of sending packets control frames are exchanged
- RTS = request to send
- CTS = clear to send
- DATA = actual packet
- ACK = acknowledgement

Advantages

- Small control frames lessen the cost of collisions (when data is large)
- RTS + CTS provide "virtual" carrier sense which protects against hidden terminal collisions (where A can't hear B)

Disadvantages:

- Not as efficient as CSMA-CD
- Doesn't solve all the problems of MAC in wireless networks (more to come)

❖ **COMMUNICATION METHODS:-**

→ **Unicast**

- Unicast is the term used to describe communication where a piece of information is sent from one point to another point. In this case there is just one sender, and one receiver.
- Unicast transmission, in which a packet is sent from a single source to a specified destination, is still the predominant form of transmission on

LANs and within the Internet. All LANs (e.g. Ethernet) and IP networks support the unicast transfer mode, and most users are familiar with the standard unicast applications (e.g. http, smtp, ftp and telnet) which employ the TCP transport protocol.

→ **Broadcast**

- Broadcast is the term used to describe communication where a piece of information is sent from one point to all other points. In this case there is just one sender, but the information is sent to all connected receivers.
- Broadcast transmission is supported on most LANs (e.g. Ethernet), and may be used to send the same message to all computers on the LAN (e.g. the address resolution protocol (arp) uses this to send an address resolution query to all computers on a LAN). Network layer protocols (such as IPv4) also support a form of broadcast that allows the same packet to be sent to every system in a logical network (in IPv4 this consists of the IP network ID and an all 1's host number).

→ **Multicast**

- Multicast is the term used to describe communication where a piece of information is sent from one or more points to a set of other points. In this case there is may be one or more senders, and the information is distributed to a set of receivers (theer may be no receivers, or any other number of receivers).
- One example of an application which may use multicast is a video server sending out networked TV channels. Simultaneous delivery of high quality video to each of a large number of delivery platforms will exhaust the capability of even a high bandwidth network with a powerful video clip server. This poses a major salability issue for applications which required sustained high bandwidth. One way to significantly ease scaling to larger groups of clients is to employ multicast networking.
- Multicasting is the networking technique of delivering the same packet simultaneously to a group of clients. IP multicast provides dynamic many-to-many connectivity between a set of senders (at least 1) and a group of receivers. The format of IP multicast packets is identical to that of unicast packets and is distinguished only by the use of a special class of destination address (class D IPv4 address) which denotes a specific multicast group. Since TCP supports only the unicast mode, multicast applications must use the UDP transport protocol.
- The multicast mode is useful if a group of clients require a common set of data at the same time, or when the clients are able to receive and store (cache) common data until needed. Where there is a common need for the same data required by a group of clients, multicast transmission may provide significant bandwidth savings (up to 1/N of the bandwidth compared to N separate unicast clients).

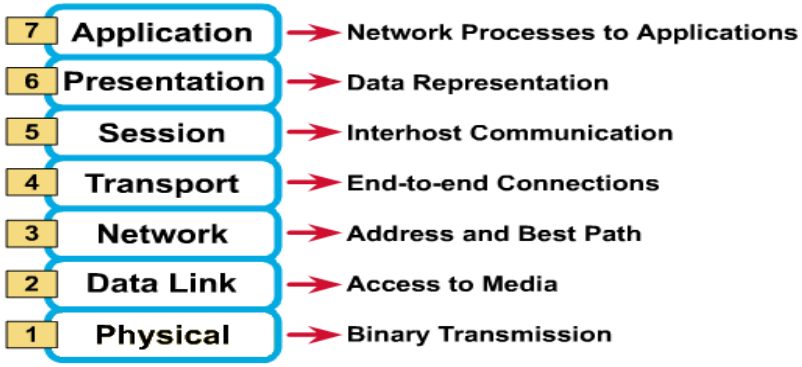
- The majority of installed LANs (e.g. Ethernet) are able to support the multicast transmission mode. Shared LANs (using hubs/repeaters) inherently support multicast, since all packets reach all network interface cards connected to the LAN. The earliest LAN network interface cards had no specific support for multicast and introduced a big performance penalty by forcing the adaptor to receive all packets (promiscuous mode) and perform software filtering to remove all unwanted packets. Most modern network interface cards implement a set of multicast filters, relieving the host of the burden of performing excessive software filtering.

Chapter 2: Network Standards

❖ Open Systems Interconnection (OSI) Model :

- International standard organization (ISO) established a committee in 1977 to develop architecture for computer communication.
- Open Systems Interconnection (OSI) reference model is the result of this effort.
- In 1984, the Open Systems Interconnection (OSI) reference model was approved as an international standard for communications architecture.
- Term "open" denotes the ability to connect any two systems which conform to the reference model and associated standards.
- The OSI model is now considered the primary Architectural model for inter-computer communications.
- The OSI model describes how information or data makes its way from application programmes (such as spreadsheets) through a network medium (such as wire) to another application programme located on another network.
- The OSI reference model divides the problem of moving information between computers over a network medium into SEVEN smaller and more manageable problems.
- This separation into smaller more manageable functions is known as layering.

OSI Reference Model: 7 Layers



➢ **Physical Layer :**

- Provides physical interface for transmission of information.
- Defines rules by which bits are passed from one system to another on a physical communication medium.
- Covers all - mechanical, electrical, functional and procedural - aspects for physical communication.
- Such characteristics as voltage levels, timing of voltage changes, physical data rates, maximum transmission distances, physical connectors, and other similar attributes are defined by physical layer specifications.

➢ **Data Link Layer :**

- Data link layer attempts to provide reliable communication over the physical layer interface.
- Breaks the outgoing data into frames and reassemble the received frames.
- Create and detect frame boundaries.
- Handle errors by implementing an acknowledgement and retransmission scheme.
- Implement flow control.
- Supports points-to-point as well as broadcast communication.
- Supports simplex, half-duplex or full-duplex communication.

➢ **Network Layer :**

- Implements routing of frames (packets) through the network.
- Defines the most optimum path the packet should take from the source to the destination
- Defines logical addressing so that any endpoint can be identified.
- Handles congestion in the network.
- Facilitates interconnection between heterogeneous networks (Internetworking).

- The network layer also defines how to fragment a packet into smaller packets to accommodate different media.

➢ **Transport Layer :**

- Purpose of this layer is to provide a reliable mechanism for the exchange of data between two processes in different computers.
- Ensures that the data units are delivered error free.
- Ensures that data units are delivered in sequence.
- Ensures that there is no loss or duplication of data units.
- Provides connectionless or connection oriented service.
- Provides for the connection management.
- Multiplex multiple connection over a single channel.

➢ **Session Layer :**

- Session layer provides mechanism for controlling the dialogue between the two end systems. It defines how to start, control and end conversations (called sessions) between applications.
- This layer requests for a logical connection to be established on an end-user's request.
- Any necessary log-on or password validation is also handled by this layer.
- Session layer is also responsible for terminating the connection.
- This layer provides services like dialogue discipline which can be full duplex or half duplex.
- Session layer can also provide check-pointing mechanism such that if a failure of some sort occurs between checkpoints, all data can be retransmitted from the last checkpoint.

➢ **Presentation Layer :**

- Presentation layer defines the format in which the data is to be exchanged between the two communicating entities.
- Also handles data compression and data encryption (cryptography).

➢ **Application Layer :**

- Application layer interacts with application programs and is the highest level of OSI model.
- Application layer contains management functions to support distributed applications.
- Examples of application layer are applications such as file transfer, electronic mail, remote login etc.

❖ **ENCRYPTION**:-

→ In cryptography, encryption is the process of encoding messages or information in such a way that only authorized parties can read it. Encryption does not of itself prevent interception, but denies the message content to the interceptor.

→ In an encryption scheme, the intended communication information or message, referred to as plaintext, is encrypted using an encryption algorithm, generating cipher text that can only be read if decrypted. For technical reasons, an encryption scheme usually uses a pseudo-random encryption key generated by an algorithm.

→ It is in principle possible to decrypt the message without possessing the key, but, for a well-designed encryption scheme, large computational resources and skill are required. An authorized recipient can easily decrypt the message with the key provided by the originator to recipients, but not to unauthorized interceptors.

❖ **COMPRESSION**:-

→ In digital signal processing, data compression, source coding, or bit-rate reduction involves encoding information using fewer bits than the original representation. Compression can be either lossy or lossless.

→ Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossy compression reduces bits by identifying unnecessary information and removing it.The process of reducing the size of a data file is referred to as data compression.

→ In the context of data transmission, it is called source coding (encoding done at the source of the data before it is stored or transmitted) in opposition to channel coding.

→ Compression is useful because it helps reduce resource usage, such as data storage space or transmission capacity. Because compressed data must be decompressed to use, this extra processing imposes computational or other costs through decompression; this situation is far from being a free lunch.

→ Data compression is subject to a space–time complexity trade-off. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it is being decompressed, and the option to decompress the video in full before watching it may be inconvenient or require additional storage.

→ The design of data compression schemes involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (when using lossy data compression), and the computational resources required to compress and decompress the data.

❖ **DISK QUOTA:-**

→ A disk quota is a limit set by a system administrator that restricts certain aspects of file system usage on modern operating systems. The function of using disk quotas is to allocate limited disk space in a reasonable way.

→ Types of quotas:-

- There are two basic types of disk quotas. The first, known as a *usage quota* or *block quota*, limits the amount of disk space that can be used. The second, known as a *file quota* or *inode quota*, limits the number of files and directories that can be created.
- In addition, administrators usually define a warning level, or *soft quota*, at which users are informed they are nearing their limit, that is less than the effective limit, or *hard quota*. There may also be a small *grace interval*, which allows users to temporarily violate their quotas by certain amounts if necessary.

❖ **MAPPING NETWORK DRIVE:-**

→ Drive mapping is how operating systems, such as Microsoft Windows, associate a local drive letter (A through Z) with a shared storage area to another computer over a network.

→ After a drive has been mapped, a software application on a client's computer can read and write files from the shared storage area by accessing that drive, just as if that drive represented a local physical hard disk drive.

<u>Drive Mapping</u>

→ Mapped Drives are hard drives (even if located on a virtual or cloud computing system, or network drives) which are always represented by names, letter(s), or number(s) and they are often followed by additional strings of data, directory tree branches, or alternate level(s) separated by a "\" symbol.

→ Drive mapping is used to locate directories, files or objects, and programs or apps, and is needed by end users, administrators, various other operators, and users or groups.

→ Mapped drives are usually assigned a letter of the alphabet after the first few taken, such as A:\, B:\, C:\, and D:\ (which is usually an optical drive unit). Then, with the drive and/or directory (letters, symbols, numbers, names, and all other components) to be mapped and might be entered into the necessary address bar/location(s).

❖ **NETMEETING** :-

→ Microsoft NetMeeting was a VoIP and multi-point videoconferencing client included in many versions of Microsoft Windows (from Windows 95 OSR2 to Windows XP).

→ It used the H.323 protocol for videoconferencing, and was interoperable with OpenH323-based clients such as Ekiga, OpenH323, and Internet Locator Service (ILS) as reflector.

→ It also used a slightly modified version of the T.120 Protocol for white boarding, application sharing (or by extension, desktop sharing), and file transfers.

❖ **FILE AND PRINT SERVICE:-**

File and printer sharing, information retrieval, and data storage are among the most frequently used network services. They are therefore crucial factors to consider when choosing a network operating system.

→ Microsoft built the Windows® 2000 Server operating system from the ground up as an integrated, multipurpose operating system. The operating system design responds to customer demands for sophisticated but easy-to-manage file and print services, for integration of Web and media content with file and print information sharing, and for meeting exponential growth in storage requirements while lowering storage cost. In addition, its open architecture lets third-party developers provide additional functionality in response to ever-changing business requirements.

→ Microsoft developed specific file and print features to meet widespread customer needs:

- **Reduced cost.** Remote Storage migrates infrequently used files to lower-cost secondary storage, yet keeps that data available if needed. Removable Storage helps reduce costs by letting multiple client applications share local libraries and tape or disk drives while ensuring that client applications do not corrupt each other's data.
- **Better manageability.** The improved NTFS file system, distributed file system (Dfs), and Indexing Service make it easier to find and access files across expanding networks. New interfaces make operating system services easier to manage; for example, the new printer interface makes it simpler for both administrators and end-users to configure and manage their printing needs.
- **Increased availability and reliability.** Dfs replication and File Replication service (FRS) synchronization help keep data available to users, even if a server or disk drive fails or a shared folder or file becomes corrupted. Dynamic volumes formatted with NTFS 5 allow fewer reboots when adding disks and creating, extending, or mirroring a volume.
- **Scalability.** The Windows 2000 NTFS version 5 file system and the Windows 2000 storage subsystems let users efficiently store and retrieve ever-larger quantities of data.

→ Organizations that install Windows 2000 file and print servers in their existing network c can take advantage of several new features. When they upgrade to a Windows 2000 network, additional file and print capabilities become available.

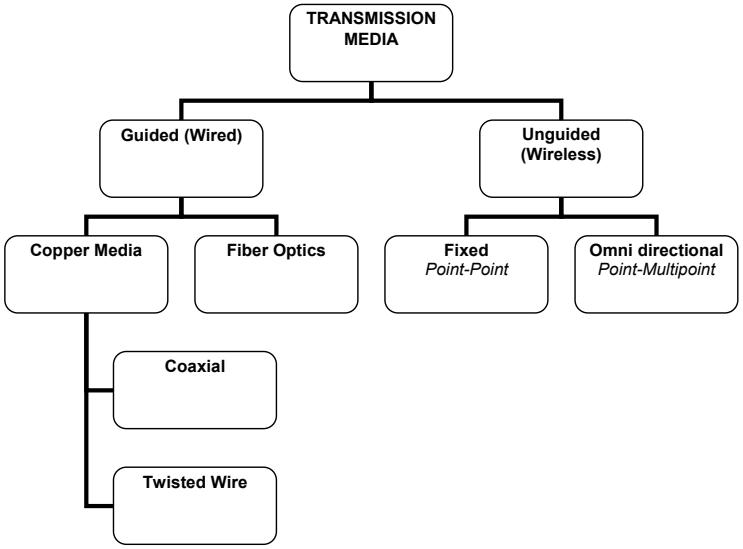
→ This overview focuses primarily on the Windows 2000 Server implementation of the standard file and print services components. However, it includes mention of several Web-related features where they are inextricably bound up with file and print services.

UNIT 2

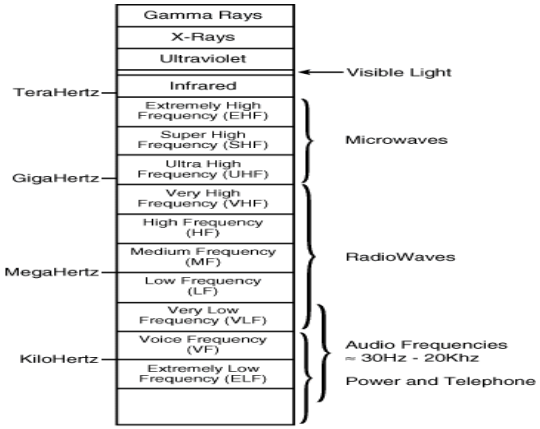
Chapter 3: Transmission Media

❖ Types of Transmission Media :

- Transmission media is divided into two: Wired or Wireless
- **Wired Media** is the most common and is further divided into three different types of cabling: Coaxial, Twisted Pairs, and Fiber Optic Cables
- **Wireless media**, which is, in a sense, no media at all, is also gaining popularity. Wireless transmissions use radio waves or infrared light to transmit data.



❖ TRANSMISSION FREQUENCIES



2GHz to 40GHz
- Microwave
- Highly directional
  Point to point
  Satellite

30MHz to 1GHz
- Omni directional
- Broadcast radio

3 x 1011 to 2 x 1014
- Infrared
- Local

❖ **CHARACTERISTICS OF TRANSMISSION MEDIA:**

Each type of transmission media has special listed as follows:

- Cost
- Installation requirements
- Bandwidth
- Band Usage (Baseband or Broadband)
- Attenuation
- Immunity from electromagnetic interference

➢ **Bandwidth**

- The term Bandwidth refers to the measure of the capacity of a medium to transmit data.

- Data transmission rates frequently are stated in terms of the bits that can be transmitted per second. The bandwidth that a cable can accommodate is determined in part by the cable's length. A short cable generally can accommodate greater bandwidth than a long cable.

➢ **Band Usage**

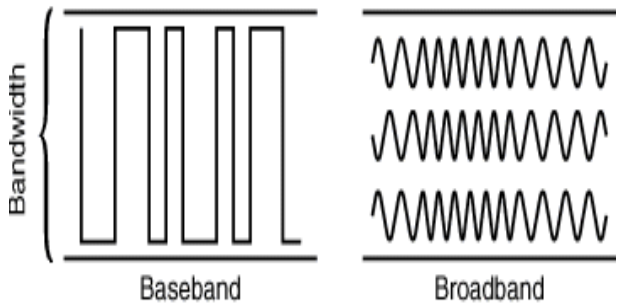
- Band Usage is the allocation of the capacity of transmission media and has two ways: **baseband** and **broadband** transmissions.

➢ **Baseband**

- Baseband is the most common mode of operation and devotes the entire capacity of the medium to one communication channel. Baseband signaling can be accomplished with both analog and digital signals

➢ **Broadband**

- Broadband enables two or more communication channels to share the bandwidth of the communications medium. This technique of dividing bandwidth into frequency bands is called frequency-division multiplexing (FDM) and works only with analog signals. Another technique, called time-division multiplexing (TDM), supports digital signals.



➢ **Attenuation**

- Attenuation is a measure of how much a signal weakens as it travels through a medium.

➢ **Electromagnetic interference (EMI)** consists of outside electromagnetic noise that distorts the signal in a medium. EMI in the

form of noise caused by nearby motors or lightning. Some network media are more susceptible to EMI than others.

- **Crosstalk** is a special kind of interference caused by adjacent wires. Crosstalk is a particularly significant problem because large numbers of cables often are located close together with minimal attention to exact placement.

❖ GUIDED TRANSMISSION MEDIA (CABLE):

- Guided Transmission Media uses a **cabling** system that guides the data signals along a **specific path**.
- Guided Media is also known as **Bounded Media**, since the data signals are a bounded system.
- Cabling technology is not limited to copper wire only. Cables can be any physical or conductive media like **wires**, **coaxial cables** or **fibre optics**

- **COAXIAL CABLE :**

- Coaxial cables were the first cable types used in communications technology.
- It consists of two conductors that share a common axis

- **The components of a coaxial cable are as follows:**

- Center conductor
- Outer conductor
- Insulation layer
- Plastic encasement/jacket

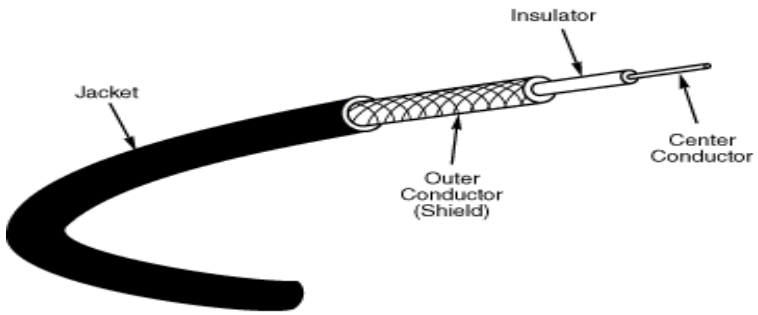


Fig: Coaxial Cable

- **Types of Coaxial Cable**

- **Thinnet:** is a light and flexible cabling medium that is inexpensive and easy to install.
- **Thicknet:** is thicker and does not bend as readily as Thinnet, Thicknet cable is harder to work with. A thicker center core, however, means that Thicknet can carry more signals a longer distance than Thinnet.

- **Coaxial Characteristics**

- **Installation**
- Coaxial cable is reasonably easy to install because the cable is strong and difficult to break. In addition, connectors can be installed with inexpensive tools and a bit of practice. The device-to-device cabling approach can be difficult to reconfigure, however, when new devices cannot be installed near an existing cabling path.
- **Cost**
- The coaxial cable used for Thinnet falls at the low end of the cost spectrum, whereas Thicknet is among the more costly options.
- **Bandwidth**
- Computes that employ coaxial cable typically have a bandwidth between 2.5 Mbps and 10 Mbps. Thicker coaxial cables offer higher bandwidth, and the potential bandwidth of coaxial is much higher than 10 Mbps.
- **EMI Characteristics**
- All copper media are sensitive to EMI, although the shield in coax makes the cable fairly resistant. Coaxial cables, however, do radiate a portion of their signal, and electronic eavesdropping equipment can detect this radiated signal.

- **Coaxial Application**

- Most versatile medium
- Television distribution
- Long distance telephone transmission
- Short distance computer systems links
- Local area networks

❖ TWISTED PAIR CABLE :

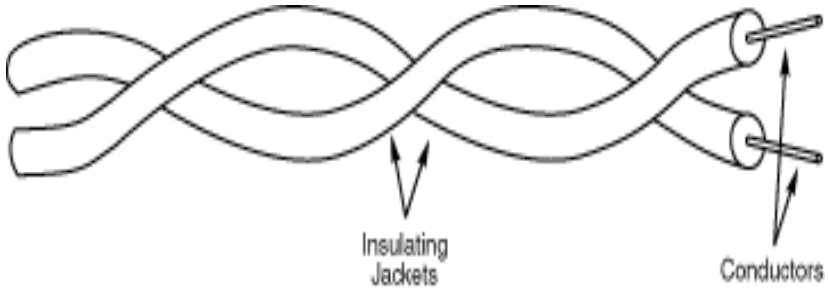
- A basic **twisted-pair cable** consists of two strands of copper wire twisted together.
- This twisting reduces the sensitivity of the cable to EMI and also reduces the tendency of the cable to radiate radio frequency noise that interferes with nearby cables and electronic components.
- Twisted-pair cable is inexpensive to install.
- Offers the lowest cost per foot of any cable type.

Application:

- Most common medium
- Telephone network
  - — Between house and local exchange (subscriber loop)
- Within buildings
  - — To private branch exchange (PBX)
- For local area networks (LAN)
  - — 10Mbps or 100Mbps

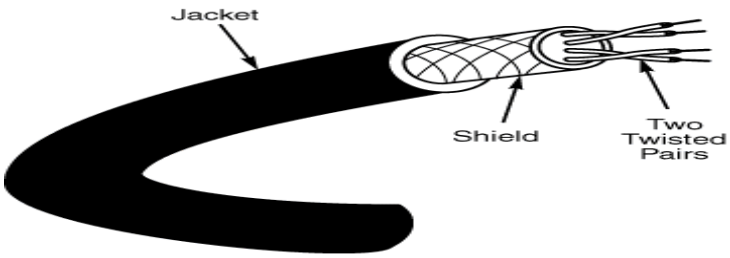
Twisted Pair Cable

- Two types of twisted-pair cable are used: (1) shielded and (2) unshielded.



(1) SHIELDED TWISTED PAIR CABLE:

- Shielded twisted-pair cabling consists of one or more twisted pairs of cables enclosed in a foil wrap and woven copper shielding.
- Metal braid or sheathing that reduces interference
- More expensive
- Harder to handle (thick, heavy)



STP Characteristics

- **Installation**
- Naturally, different network types have different installation requirements. One major difference is the connector used. In many cases, installation can be greatly simplified by using pre-wired cables.
- **Cost**
- STP cable costs more than thin coaxial or unshielded twisted-pair cable. STP is less costly, however, than thick coax or fiber-optic cable.
- **Capacity**
- The most common data rate for STP cable is 16 Mbps, which is the top data rate for Token Ring networks.
- **Attenuation**
- All varieties of twisted-pair cable have attenuation characteristics that limit the length of cable runs to a few hundred meters, although a 100-meter limit is most common.
- **EMI Characteristics**
- The shield in STP cable results in good EMI characteristics for copper cable. Comparable to the all copper cables, STP is sensitive to interference and vulnerable to electronic eavesdropping.

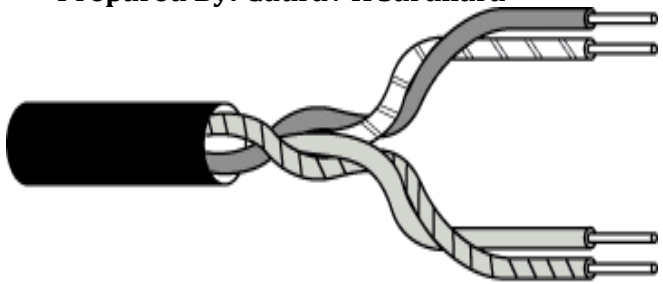
(2) UNSHIELDED TWISTED PAIR CABLE:

- The characteristics of the **Unshielded Twisted Pair cables (UTP)** are similar in many ways to STP, differing primarily in attenuation and EMI. Several twisted-pairs can be bundled together in a single cable. These pairs typically are color coded to distinguish them.
- Ordinary telephone wire
- Cheapest
- Easiest to install
- Suffers from external EM interference

UTP cable is available in the following five grades, or categories:

-
- **Categories 1 and 2** - voice-grade cables are suitable only for voice and for low data rates (below 4 Mbps)
- **Category 3** - generally suited for data rates up to 10 Mbps
- **Category 4** - consists of four twisted-pairs, is suitable for data rates up to 16 Mbps.
- **Category 5** - consists of four twisted-pairs, is suitable for data rates up to 100 Mbps.

UNSHIELDED TWISTED PAIR CABLE

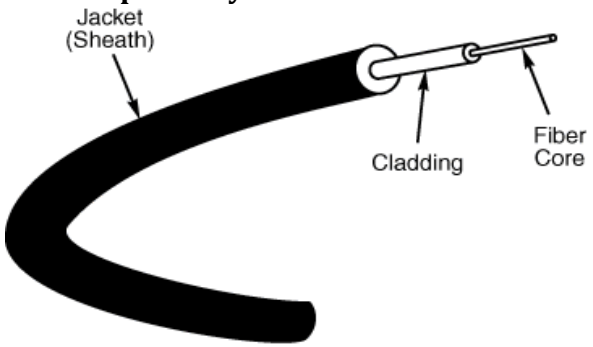


- **UTP Characteristics**

- **Installation**
- UTP cable is easy to install. Some specialized equipment might be required, but the equipment is low in cost.
- Category 5 cable has stricter installation requirements than lower categories of UTP.
- **Cost**
- UTP cable is the less costly, although properly installed Category 5 tends to be fairly expensive. Distance limits for voice cabling are much less severe than for data-grade cabling.
- **Capacity**
- The data rates possible with UTP have pushed up from 1 Mbps, past 4 and 16 Mbps, to the point where 100 Mbps data rates are now common.
- **Attenuation**
- UTP cable shares similar attenuation characteristics with other copper cables. UTP cable runs are limited to a few hundred meters, with 100 meters as the most frequent limit.
- **EMI Characteristics**
- Because UTP cable lacks a shield, it is more sensitive to EMI than coaxial or STP cables. UTP might not be suitable for noisy environments such as factories. Crosstalk between nearby unshielded pairs limits the maximum length of cable runs

❖ **FIBER-OPTIC CABLE :**

- Fiber-optic cable is the ideal cable for data transmission because it accommodates extremely high bandwidths,
- has no problems with EMI, ssupports durable cables and cable runs as long as several kilometers.
- The two disadvantages of fiber-optic, however, are cost and installation difficulty.
- Optical fiber cables don't transmit electrical signals. Instead, the data signals must be converted into light signals.



Fiber Optic Cable

- The center conductor of a fiber-optic cable is a fiber that consists of highly refined glass or plastic. The fiber is coated with a cladding that reflects signals back into the fiber to reduce signal loss. A plastic sheath protects the fiber.
- A fiber-optic network cable consists of two strands separately enclosed in plastic sheaths—one strand sends and the other receives.
- Two types of cable configurations are available: loose and tight configurations.

- **Application :**

- greater capacity (bandwidth of up to 2 Gbps)
- smaller size and lighter weight
- lower attenuation
- immunity to environmental interference
- highly secure due to tap difficulty and lack of signal radiation

- **Fiber-Optics Characteristics**

- **Installation**
- Fiber-optic cable requires greater care because the cables must be treated fairly gently during installation. Every cable has a minimum bend radius, and fibers are damaged if the cables are bent too sharply. It also is important not to stretch the cable during installation.

- **Cost**
- Fiber-optic cable is the most expensive cable type to install.

- **Capacity**
- Fiber-optic cable can support high data rates (as high as 200,000 Mbps) even with long cable runs. Fiber-optic cables can transmit 100 Mbps signals for several kilometers.

- **Attenuation**
- Attenuation in fiber-optic cables is much lower than in copper cables. Fiber-optic cables are capable of carrying signals for several kilometers.

- **EMI Characteristics**
- Because fiber-optic cables don't use electrical signals to transmit data, they are totally immune to electromagnetic interference. The cables also are immune to a variety of electrical effects that must be taken into account when designing copper cabling systems.
- Because the signals in fiber-optic cable are not electrical in nature, they cannot be detected by the electronic eavesdropping equipment that detects electromagnetic radiation. Therefore, fiber-optic cable is the perfect choice for high-security networks.

❖ **UNGUIDED TRANSMISSION MEDIA (Wireless):**

- **Unguided Media or Wireless Communication** consists of a means (e.g. air, space) for the data signals to travel, where there is nothing to guide them along a specific path, like in wires.
- Unbounded media is electromagnetic waves in form of radio, microwave, infrared or others.
- Wireless communication is used where cables are difficult to use or install.
- Wireless transmission media refers to the methods of carrying data through the air or space using infrared, radio, or microwave signals.

- **REASONS FOR WIRELESS TECHNOLOGY:**

- Spaces where cabling would be impossible or inconvenient.
- People who move around a lot within their work environment.
- Temporary installations.
- People who travel outside of the work environment and need instantaneous access to network resources.

- **CLASSIFICATIONS OF WIRELESS TRANSMISSION:**

- **Classification by Propagation :**

- Fixed (Directional)
- Mobile (Omni directional)

- **Classification by Method**

- Infrared
- Laser

- Narrow-band radio
- Spread-spectrum radio
- Microwaves

➢ **Infrared :**

- Infrared is a wireless transmission medium that carries data via light beams.
- Transmitter and receiver must be in line of sight.
- Infrared technology allows computing devices to communicate via short-range wireless signals using infrared lights wherein they are typically are limited to within 100 feet.
- Uses transmitters/receivers (transceivers) that modulate no coherent infrared light.
- Transceivers must be within line of sight of each other (directly or via reflection).
- Unlike microwaves, infrared does not penetrate walls.
- Initially used in remote controls

- **<u>Advantages:</u>**

- Portable - no antennae required
- Inexpensive
- **<u>Disadvantages:</u>**

- Limited range
- Very Sensitive

➢ **Laser Transmission :**

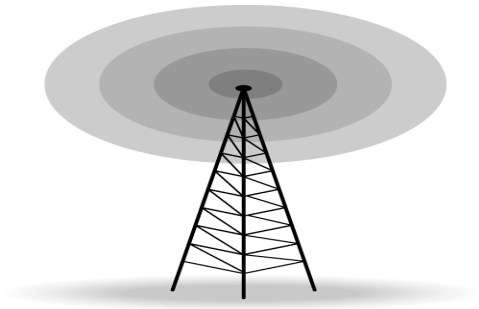
- High-powered laser transmitters can transmit data for several thousand yards when line-of-sight communication is possible.
- Lasers can be used in many of the same situations as microwave links
- Laser light technology is similar to infrared technology.

➢ **Radio :**

- I Radio is a wireless transmission medium that carries data via radio frequency signals.
- Wireless LANs in a home or business are one type of radio technology.
- Radio signals can be long range (between cities or regions) and short range (within a building).
- Radio signals are susceptible to noise and electrical Interference.
- Radio waves are used for multicast communications,
- such as radio and television, and paging systems.

- They can penetrate through walls.
- Highly regulated.
- Use Omni directional antennas
- Radio is a general term often used to encompass frequencies in the range 3 kHz to 300 GHz.

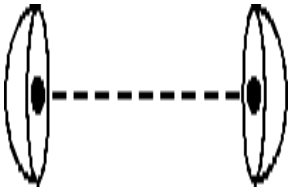
Omni directional antenna



➢ **Microwaves :**

- Microwaves are high frequency radio waves.
- Much of long-distance telephone service is carried by microwaves.
- Microwaves travel in a straight line.
- Microwave relay stations are built about 30 miles apart.
- Microwaves are used for unicast communication.
- Such as cellular telephones, satellite networks, and wireless LANs.
- Higher frequency ranges cannot penetrate walls.
- Use directional antennas - point to point line of sight communications

Directional Point-to-point focused beams employing high frequencies



Applications:
- Television distribution
- Long-distance telephone transmission
- Private business networks

Disadvantages:
- Line of sight requirement

- Expensive towers and repeaters
- Subject to interference such as passing airplanes and rain

❖ **The Concept of Multiplexing:**

- **Multiplexing** refers to the combination of information streams from multiple sources for transmission over a shared medium.
- **Demultiplexing** refers to the separation of a combination of information streams back into separate information streams

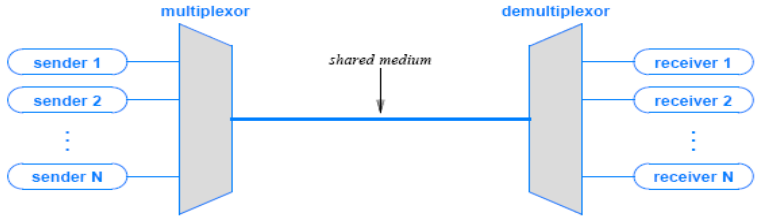
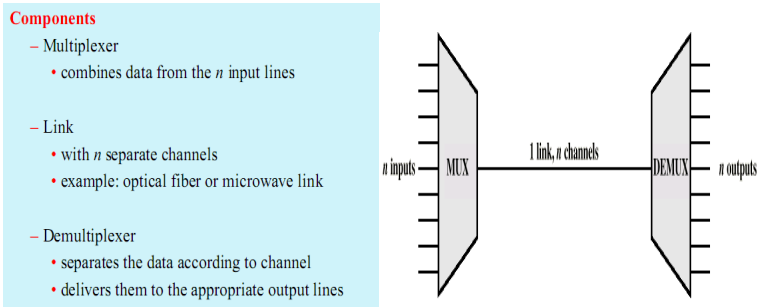


Figure 11.1 The concept of multiplexing in which independent pairs of senders and receivers share a transmission medium.

❖ **Principles of Multiplexing:**



● **Multiplexing : Applications:**

- Telephone systems
- Telemetry
- Satellites
- Broadcasting (radio and TV)

❖ Basic Types of Multiplexing:

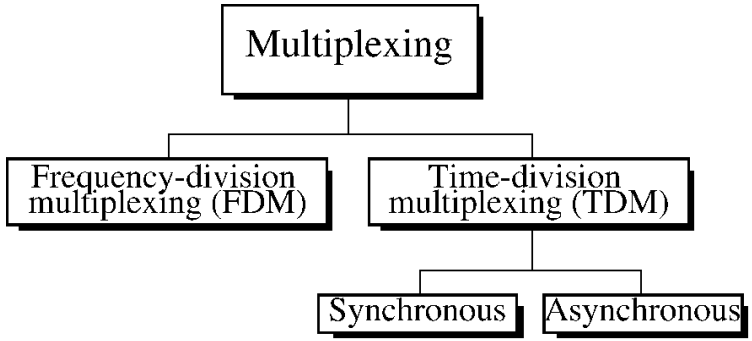
- There are four basic approaches to multiplexing that each have a set of variations and implementations
  1. Frequency Division Multiplexing (FDM)
  2. Wavelength Division Multiplexing (WDM)
  3. Time Division Multiplexing (TDM)
  4. Code Division Multiplexing (CDM)
- TDM and FDM are widely used
- WDM is a form of FDM used for optical fiber
- CDM is a mathematical approach used in cell phone mechanisms.

➢ **The two most common types of multiplexing are :**

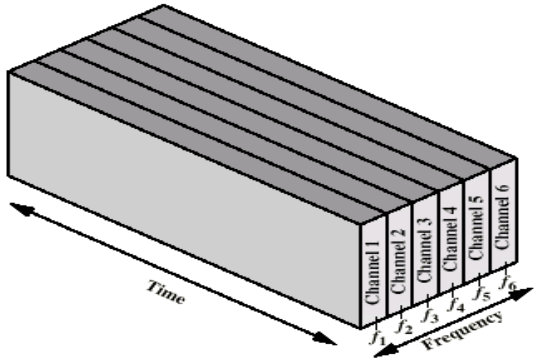
1. Frequency-division multiplexing (FDM)
   - Generally used for analog information.
   - Individual signals to be transmitted are assigned a different frequency within a common bandwidth.

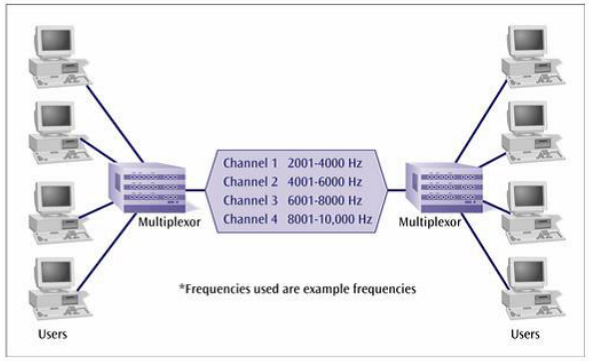
2. Time-division multiplexing (TDM)
   - Generally used for digital information.
   - Multiple signals are transmitted in different time slots on a single channel.

➢ **Types of multiplexing:-**



❖ FrequencyDivisionMultiplexing(FDM) :





FDM: all signals are transmitted at the same time (all the time) but in different frequency bands

- It is possible to send Simultaneously multiple carrier waves over a single copper wire
- A demultiplexor applies a set of filters that each extract a small range of frequencies near one of the carrier frequencies.
- Advantage of FDM arises from the simultaneous use of a transmission medium by multiple pairs of entities
- We imagine FDM as providing each pair with a private transmission path
- as if the pair had a separate physical transmission medium

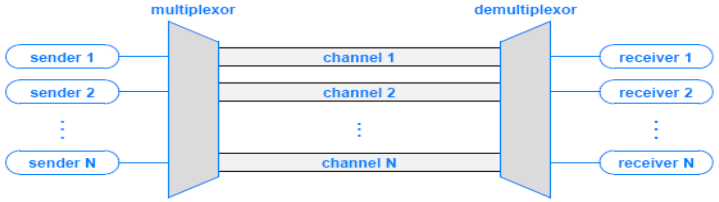


Figure 11.3 The conceptual view of Frequency Division Multiplexing (FDM) as providing a set of independent channels.

- **Limitations Of FDM :**

  In practical FDM systems, there are some limitations:

  - If the frequencies of two channels are too close, interference can occur
  - Furthermore, demultiplexing hardware that receives a combined signal must be able to divide the signal into separate carriers
  - Designers choosing a set of carrier frequencies with a gap between them known as a guard band

Channel	Frequencies Used
1	100 KHz - 300 KHz
2	320 KHz - 520 KHz
3	540 KHz - 740 KHz
4	760 KHz - 960 KHz
5	980 KHz - 1180 KHz
6	1200 KHz - 1400 KHz

Figure 11.4 An example assignment of frequencies to channels with a guard band between adjacent channels.

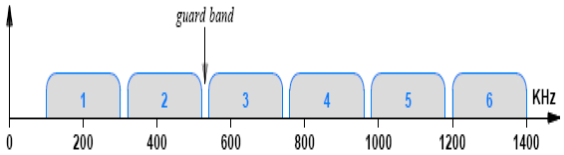


Figure 11.5 A frequency domain plot of the channel allocation from Figure 11.4 with a guard band visible between channels.

- **FDM Advantages & Disadvantages :**

Advantages:

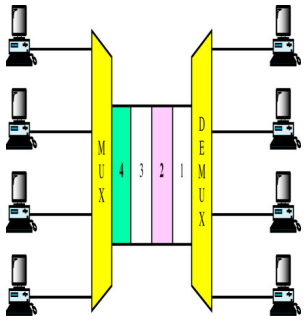
– no dynamic coordination needed

Disadvantages:

– guard spaces

❖ Time Division Multiplexing(TDM):

- Generally used for digital information.
- Multiple signals are transmitted in different time slots on a single channel.



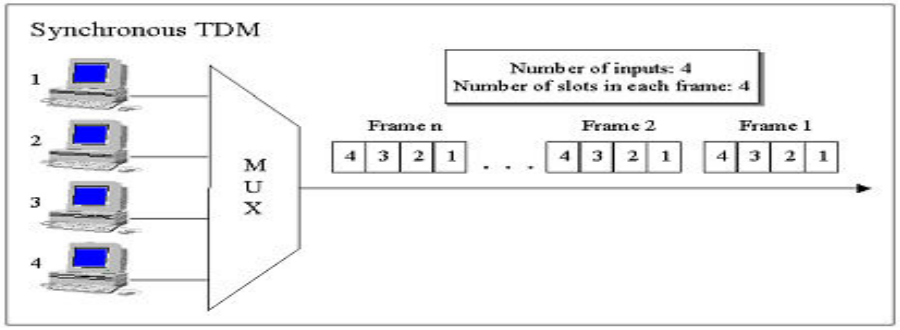
- Sharing of the signal is accomplished by dividing available transmission time on a medium among users.
- Digital signaling is used exclusively.
- Time division multiplexing comes in two basic forms:
  - Synchronous time division multiplexing, and
  - Statistical, or asynchronous time division multiplexing.

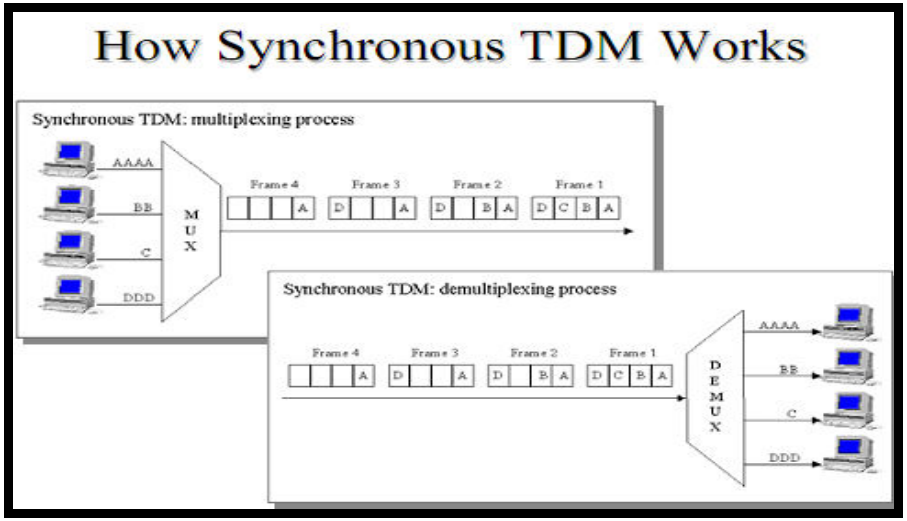
❖ Synchronous TDM :

- It is widely used throughout the Internet

Synchronous TDM

- The multiplexer allocates exactly the same time slot to each device at all times, whether or not a device has anything to transmit
- A frame consists of one complete cycle of time slots. Thus the number of slots in frame is equal to the number of inputs.

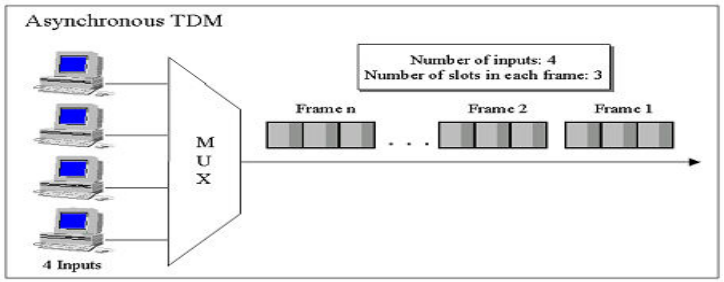


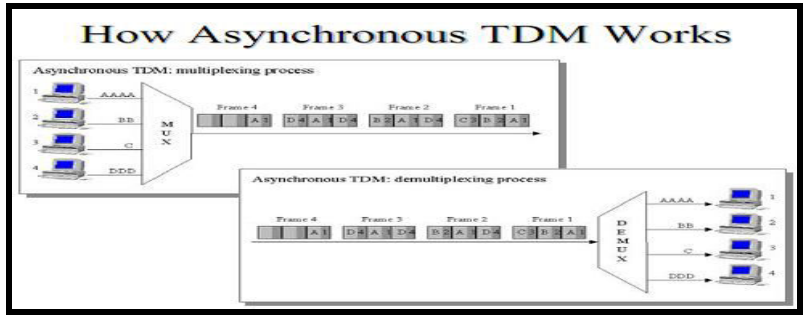


❖ **Asynchronous TDM :**

Asynchronous TDM
(or statistical time-division multiplexing)

- Each slot in a frame is **not** dedicated to the fix device
- The number of slots in a frame is not necessary to be equal to the number of input devices. More than one slots in a frame can be allocated for an input device.





❖ **Switching:-**

→ A network consists of many switching devices. In order to connect multiple devices, one solution could be to have a point to point connection in between pair of devices. But this increases the number of connection. The other solution could be to have a central device and connect every device to each other via the central device which is generally known as Star Topology. Both these methods are wasteful and impractical for very large network. The other topology also can not be used at this stage. Hence a better solution for this situation is SWITCHING. A switched network is made up of a series of interconnected nodes called switches.

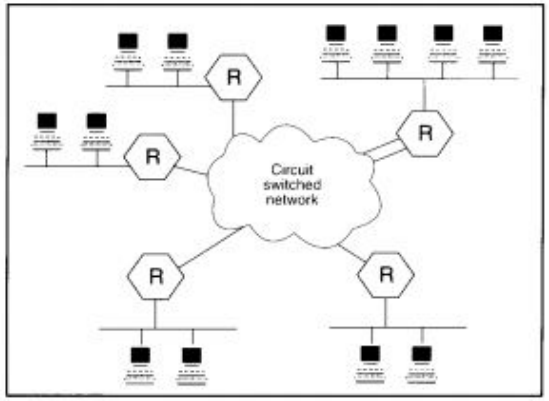
Types of Switching Techniques

→ There are basically three types of switching methods are made available. Out of three methods, circuit switching and packet switching are commonly used but the message switching has been opposed out in the general communication procedure but is still used in the networking application.

1) Circuit Switching
2) Packet Switching
3) Message Switching

Circuit Switching

→ Circuit Switching is generally used in the public networks. It come into existence for handling voice traffic in addition to digital data. How ever digital data handling by the use of circuit switching methods are proved to be inefficient. The network for Circuit Switching is shown in figure.



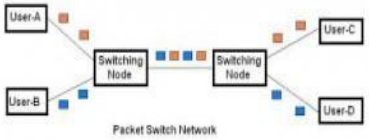
Circuit Switching Network

- Here the network connection allows the electrical current and the associated voice with it to flow in between the two respective users. The end to end communication was established during the duration of call.
- In circuit switching the routing decision is made when the path is set up across the given network. After the link has been sets in between the sender and the receiver then the information is forwarded continuously over the provided link.
- In Circuit Switching a dedicated link/path is established across the sender and the receiver which is maintained for the entire duration of conversation.

Packet Switching

→ In Packet Switching, messages are broken up into packets and each of which includes a header with source, destination and intermediate node address information. Individual Packets in packet switching technique take different routes to reach their respective destination. Independent routing of packets is done in this case for following reasons:

1. Bandwidth is reduces by the splitting of data onto different routes for a busy circuit.
2. For a certain link in the network, the link goes down during transmission the the remaining packet can be sent through the another route.

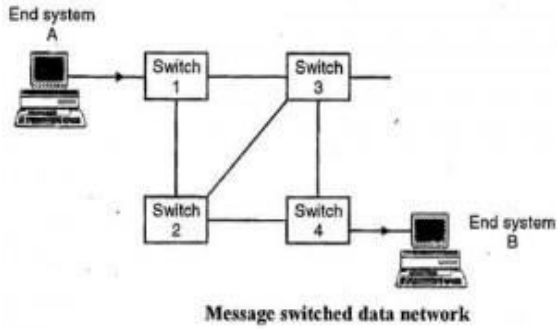


→ Packet Switching Network

- The major advantage of Packet switching is that they they are used for performing data rate conversion.
- When traversing the network switches, routers or the other network nodes then the packets are buffered in the queue, resulting in variable delay and throughput depending on the network's capacity and the traffic load on network.
- Packet switching contrasts with another principal networking paradigm, circuit switching, a method which sets up a limited number of dedicated connections of constant bit rate and constant delay between nodes for exclusive use during the communication session.
- In cases where traffic fees are charged, for example in cellular communication, packet switching is characterized by a fee per unit of information transmitted.

Message Switching

→ In case of Message Switching it is not necessary to established a dedicated path in between any two communication devices. Here each message is treated as an independent unit and includes its own destination source address by its own. Each complete message is then transmitted from one device to another through internetwork.



Message Switching Data Network

- Each intermediate device receive the message and store it until the nest device is ready to receive it and then this message is forwarded to the next device. For this reason a message switching network is sometimes called as Store and Forward Switching.
- Message switches can be programmed with the information about the most efficient route as well as information regarding to the near switches that can be used for forwarding the present message to their required destination.
- The storing and Forwarding introduces the concept of delay. For this reasons this switching is not recommended for real time applications like voice and video.

Chapter 5: Networking Devices and Routing

❖ Layer 1 Devices :

❖ **NIC(NETWROK INTERFACE CARD) :**

- NIC Stands for Network Interface card. It is one of the most important computer network devices that are used for the data communication and to connect the computers with each other. It is plugged inside the computer either in the PCI slot or it is built-in the motherboard.

- A twisted pair UTP/STP with RJ45 connector is used to connect the computer with the Hub or Switch. Fiber optic cable can also be used to connect the computer with the hub or switch.
- A NIC can be wired or wireless and it has digital circuitry and microprocessor. A wireless NIC is used to connect the computers with each other wirelessly. There are different vendors of the NIC such as D-Link, 3Com, Intel, Realtek, Baylan and Baynet.
- Before buying and installing a network interface card in your computer make sure that it is compatible with the other network devices. NIC card operates on the Data Link and physical layer of the OSI layers model.
- For every computer in a network, it is required to have a NIC to communicate with other computers.
- Every NIC has unique MAC address and no two NIC cards from two different vendors can have the same MAC address. NIC has twisted pair, BNC and AUI sockets. The one end of the network cable is used to connect with the NIC and the other end is used to connect with the hub or switch.
  - NIC provides the fulltime connectivity for the data transmission.

NIC (NETWORK INTERFACE CARD)

- Sometimes computers do not communicate with each other due to the malfunctioning of the NIC.

- **The network interface cards problems can be resolved with the following tips.**

- Make sure that you have the updated and correct version of the LAN card's driver.
- Ensure that the LEDs of the NIC are working properly.
- Check that the network cable is properly connected at both ends.
- Right click on the network status icon on the right bottom of the desktop and click repair.
- Ensure that the TCP/IP settings are accurate.
- Disable the antivirus and firewall.

- If the problem still persists then try to replace the NIC with a new one.

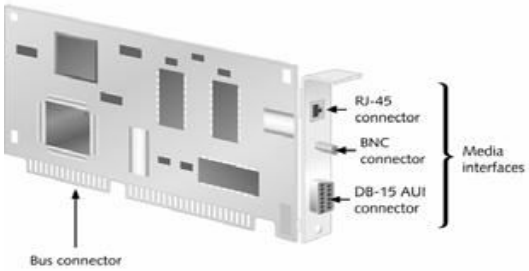


Figure 4-2 Ethernet NIC with interfaces for thinnet (BNC), thicknet (AUI), anc 10BaseT (RJ-45)

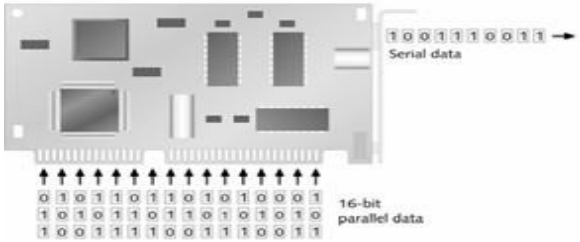
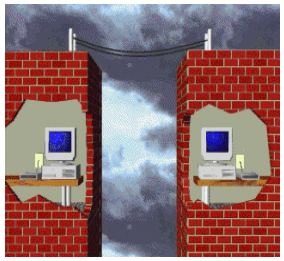
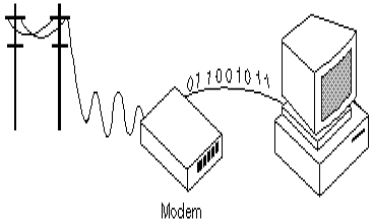


Figure 4-1 NICs mediate communication between a computer and the network cable

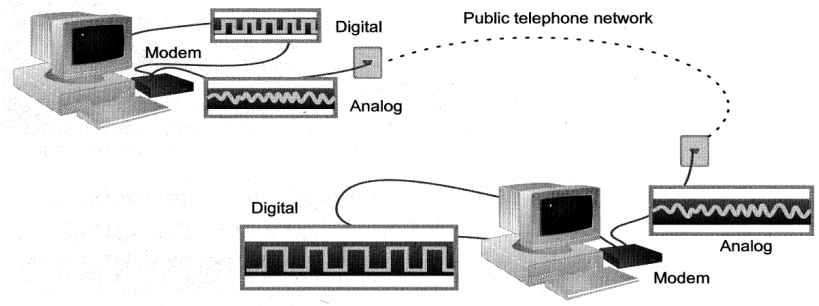
❖ MODEM :





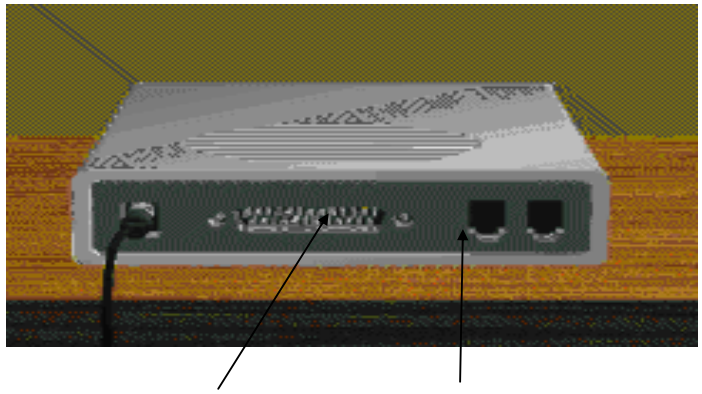
- Short for modulator-demodulator. A modem is a device or program that enables a computer to transmit data over, for example, telephone or cable lines.

- Computer information is stored digitally, whereas information transmitted over telephone lines is transmitted in the form of analog waves. A modem converts between these two forms.
- Allow computers to communicate over a telephone line
- Enable communication between networks or connecting to the world beyond the LAN
- Cannot send digital signal directly to telephone line
- **Sending end**: Modulate the computer's digital signal into analog signal and transmits
- **Receiving end**: Demodulate the analog signal back into digital form
-



- **Modems typically have the following I/O interface:**

- A serial RS-232 communication interface
- An RJ-11 telephone-line interface (a telephone plug)



RS-232 **RJ-11**

- **Modem Performance Measures :**

- **Baud rate -** the number of symbol change per second on the transmission line**.**
- **Bit per second (bps) -** number of bits transmitted per second.

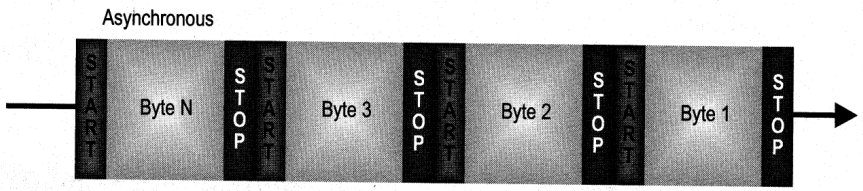
❖ **Types of Modem :**

- Modems are classified according to the transmission method they use for sending and receiving data.

- **The two basic types of modems are as follows:**
- Asynchronous modems
- Synchronous modems

➢ **Asynchronous Modems :**

- No clocking devices
- Commonly used in telephone networks
- Data is transmitted in a serial stream. Each character is turned into a string of 8 bits
- Each of these characters is separated by one start bit and one or two stop bits



- Asynchronous transmission does not use a clocking mechanism to keep the sending and receiving devices synchronized.
- Instead, this type of transmission uses bit synchronization to synchronize the devices for each frame that is transmitted.
- In bit synchronization, each frame begins with a start bit that enables the receiving device to adjust to the timing of the transmitted signal.

- Asynchronous transmission is most frequently used to transmit character data and is ideally suited to environments in which characters are transmitted at irregular intervals, such as when users enter character data.

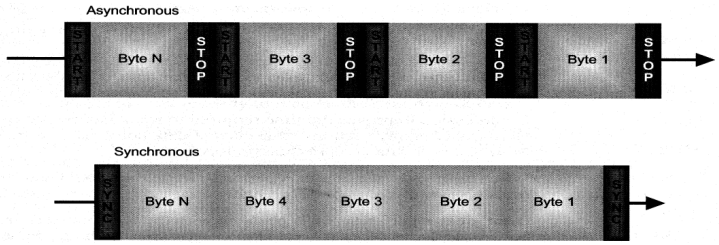
- Asynchronous transmission generally requires less expensive hardware than synchronous transmission.

➢ **Synchronous Modems :**

- Need clocking devices
- Data are transmitted in blocks
- Used in digital networks
- Synchronous transmission eliminates the need for start and stop bits by synchronizing the clocks on the transmitting and receiving devices.

- **Synchronous transmission offers many advantages over asynchronous transmission**.
- The overhead bits (synch, CRC, and end) comprise a smaller portion of the overall data frame, which provides for more efficient use of available bandwidth.
- Synchronization improves error detection and enables the devices to operate at higher speeds.

- **The disadvantage of synchronous transmission is that**
- The more complex circuitry necessary for synchronous communication is more expensive.
- Network adapter cards commonly employ synchronous transmission methods.



- **Comparison : Asynchronous modems Vs Synchronous modems**

- **Asynchronous modems** are relatively simple and economic
  - Large overhead - can be up to 20 to 27% of the data traffic
  - Error control is done by using parity bit or higher layer protocols,

- **Synchronous modems** are relatively complicated and expensive

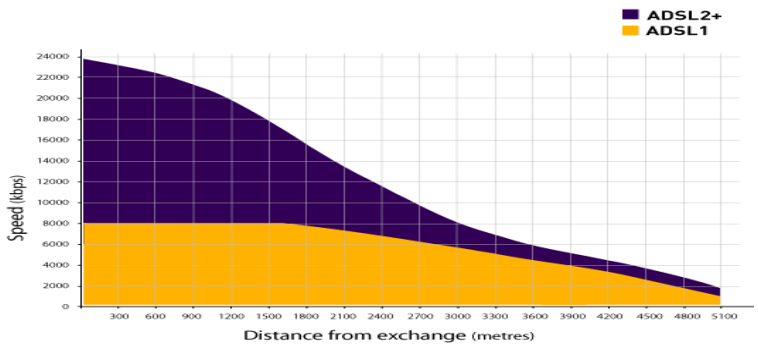
  - Seldom use in home market
  - Less overhead means higher efficiency
  - More sophisticated error control protocol is required.

❖ **DSL AND ADSL**:-

→ DSL (Digital Subscriber Line) is a common technology for bringing high-bandwidth (broadband) information to homes and small businesses over standard (copper) telephone lines. DSL comes in many different flavors, such as SDSL and ADSL2. The most common forms of DSL in the UK, ADSL (Asymmetric DSL) and ADSL2+, splits a single telephone line into separate voice and data channels, allowing you to make a phone call while surfing the Internet at the same time.

→ ADSL offers speeds of up to 8Mbps (Megabits per second) downstream and 448Kbps upstream (832Kbps on business lines). The technology is cheap, fast and easy to install (doesn't require an engineer) and reasonably reliable, although performance can suffer due to ISP congestion, distance from the local exchange (shorter lines are faster but anything over 6.5km is usually slow), poor home wiring and interference from other electrical devices. Each connection is fixed to a specific telephone line.

→ The latest ADSL2+ (ITU G.992.5) technology is capable of pushing download speeds at up to 24Mbps and uploads at up to 1.4Mbps, it also supports port bonding (linking several lines together for faster speeds) and has an improved range over ADSL. Both ADSL and ADSL2+ are "best efforts" broadband services, which means that bandwidth is shared between many users and can be highly variable - especially over long distances and at off-peak times (i.e. busy afternoons will slow the performance). The following graph shows what impact distance can have on speeds.



→ Users of these services should learn to understand common router statistics, which can help in diagnosing line problems. Figures for the

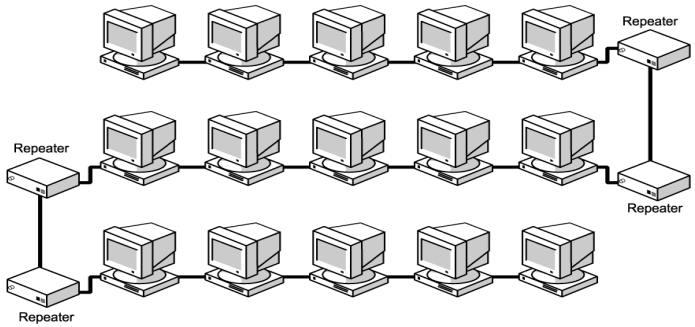
following items will change depending on your line condition (e.g. expect them to be worse during thunderstorms that will cause extra interference):

❖ **Repeaters :**

- The purpose of a repeater is to extend the maximum range for the network cabling.
- A repeater is a network device that repeats a signal from one port onto the other ports to which it is connected (see Figure).
- Repeaters operate at the OSI Physical layer.
- A repeater does not filter or interpret—it merely repeats (regenerates) a signal, passing all network traffic in all directions.
- A repeater doesn't require any addressing information from the data frame because a repeater merely repeats bits of data. This means that if data is corrupt, a repeater will regenerate the signal anyway.
- The advantages of repeaters are that they are inexpensive and simple.

- Some repeaters simply amplify signals. Although this increases the strength of the data signal, it also amplifies any noise on the network. In addition, if the original signal has been distorted in any way, an amplifying repeater cannot clean up the distortion.

- Multi-port repeaters are often called hubs.
- Hubs are very common internetworking devices. Generally speaking, the term hub is used instead of repeater when referring to the device that serves as the center of a star topology network.
- **Disadvantages :**
- it can't filter network traffic. Data, sometimes referred to as bits, arriving at one port of a repeater gets sent out on all other ports

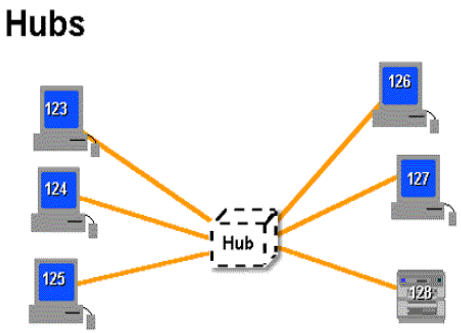


❖ **Hub :**

- Hubs, also called wiring concentrators, provide a central attachment point for network cabling.
- An unintelligent network device that sends one signal to all of the stations connected to it.
- Traditionally, hubs are used for star topology networks, but they are often used with other configurations to make it easy to add and remove computers without bringing down the network.
- Resides on Layer 1 of the OSI model

- **Hubs come in three types:**

- Passive
- Active
- Switching



- **Passive Hub:**

- Passive hubs do not contain any electronic components and do not process the data signal in any way.
- The only purpose of a passive hub is to combine the signals from several network cable segments.
- All devices attached to a passive hub receive all the packets that pass through the hub.
- Because the hub doesn't clean up or amplify the signals (in fact, the hub absorbs a small part of the signal),
- The distance between a computer and the hub can be no more than half the maximum permissible distance between two computers on the network. For example, if the network design limits the distance between two computers to 200 meters, the maximum distance between a computer and the hub is 100 meters.

- As you might guess, the limited functionality of passive hubs makes them inexpensive and easy to configure.
- That limited functionality, however, is also the biggest disadvantage of passive hubs.
- Often small networks use passive hubs, due to the fact that there are few machines on the LAN and small distances between them.

- **Active Hub:**

- Active hubs incorporate electronic components that can amplify and clean up the electronic signals that flow between devices on the network.
- This process of cleaning up the signals is called signal regeneration.

Signal regeneration has the following benefits:

- The network is more robust (less sensitive to errors).
- Distances between devices can be increased.
- These advantages generally outweigh the fact that active hubs cost considerably more than passive hubs.
- Because active hubs function in part as repeaters, they occasionally are called multiport repeaters.

- **Intelligent Hub :**

- Intelligent hubs are enhanced active hubs. Several functions can add intelligence to a hub:

- **Hub management:** Hubs now support network management protocols that enable the hub to send packets to a central network console. These protocols also enable the console to control the hub; for example, a network administrator can order the hub to shut down a connection that is generating network errors.

- **Switching:** The latest development in hubs is the switching hub, which includes circuitry that very quickly routes signals between ports on the hub. Instead of repeating a packet to all ports on the hub, a switching hub repeats a packet only to the port that connects to the destination computer for the packet. Many switching hubs have the capability of switching packets to the fastest of several alternative paths. Switching hubs are replacing bridges and routers on many networks. a switching hub acts like a very fast bridge.

- Switching hubs are the most expensive of hubs on the market. Often they are simply referred to as
  Switches. As for the exam, think of all types of hubs as simply a hub.

❖ **Hub : Advantages And Disadvantages :**

- **Advantages :**
- As an active hubs regenerate signals, it increases the distance that can be spanned by the LAN (up to 100 meters per segment).
- Hubs can also be connected locally to a maximum of two other hubs, thereby increasing the number of devices that can be attached to the LAN.
- Active hubs are usually used against attenuation, which is a decrease in the strength of the signal over distance.
- **Disadvantages :**
- Bandwidth is shared by all hosts i.e. 10Mbs shared by 25 ports/users.
- Can create bottlenecks when used with switches.
- Have no layer 3 switching capability.
- Active hubs are usually used against attenuation, which is a decrease in the strength of the signal over distance.

❖ **Layer 2 Device :**

❖ **SWITCH:-**

→ A **network switch** (also called **switching hub**, **bridging hub**, officially **MAC bridge** is a computer networking device that connects devices together on a computer network, by using packet switching to receive, process and forward data to the destination device. Unlike less advanced network hubs, a network switch forwards data only to one or multiple devices that need to receive it, rather than broadcasting the same data out of each of its ports.

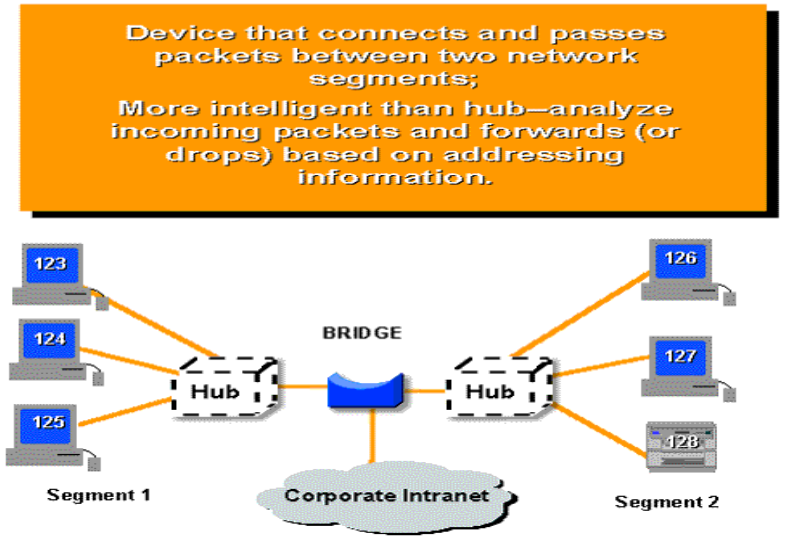
→ A network switch is a multiport network bridge that uses hardware addresses to process and forward data at the data link layer (layer 2) of the OSI model. Switches can also process data at the network layer (layer 3) by additionally incorporating routing functionality that most commonly uses IP addresses to perform packet forwarding; such switches are commonly known as layer-3 switches or multilayer switches. Beside most commonly used Ethernet switches, they exist for various types of networks, including Fibre Channel, Asynchronous Transfer Mode, and InfiniBand. The first Ethernet switch was introduced by Kalpana in 1990.

→ A switch is a device in a computer network that electrically and logically connects together other devices. Multiple data cables are plugged into a switch to enable communication between different networked devices.

Switches manage the flow of data across a network by transmitting a received message only to the one or more devices for which the message was intended. Each networked device connected to a switch can be identified using a MAC address, allowing the switch to regulate the flow of traffic. This maximizes the security and efficiency of the network.

→ Essentially, when replacing a repeater hub with an Ethernet switch, the single large collision domain is split up into smaller ones, reducing the probability and scope of collisions and, as a result, increasing the potential throughput. Because broadcasts are still being forwarded to all connected devices, the newly formed network segment continues to be a broadcast domain.

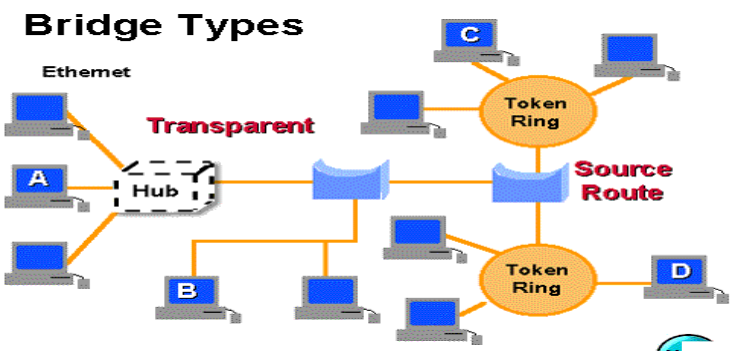
❖ **Bridge :**



- Bridges can extend the maximum size of a network.
- Bridge is a much more flexible device
- Bridges operate at the MAC sub layer of the OSI Data Link layer.

- Bridge is more selective and passes only those signals targeted for a computer on the other side.
- A bridge can make this determination because each device on the network is identified by a unique physical address.

- Bridges accomplish several things. First, they divide busy networks into smaller segments. If the network is designed so that most packets can be delivered without crossing a bridge, traffic on the individual network segments can be reduced.



- **Source Route Bridge :**

  – Used in Token Ring networks.
  – The entire path (ring number and bridge number) is embedded within Packet
  • Search frame
  • Route discovery frame

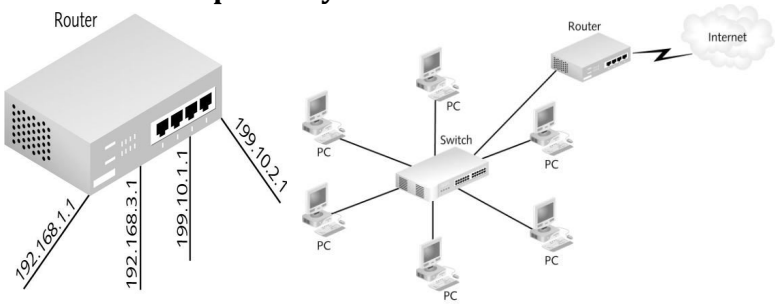
- **Translational bridge :**

  – Used to convert one networking data format to another.
  • For example, from Token

❖ **<u>Layer 3 Device :</u>**

❖ **Router :**

- **Routing: The** process of determining systematically: How to forward messages toward the destination node based on its address.
- Routers are another type of internetworking device.
- These devices pass data packets between networks based on network protocol or layer 3 information.
- Routers have the ability to make intelligent decisions as to the best path for delivery of data on the network.
- Provide filtering and network traffic control
- Used on LANs and WANs
- Connect multiple segments and networks.
- Multiple routers create an "internetwork"
- Operate at the Network layer
- Create a table to determine how to forward packets
- Filtering and traffic control base on logical addresses.



- **Routers come in two general types:**

- **Static Routers**: These routers do not determine paths. Instead, you must configure the routing table, specifying potential routes for packets.

- **Dynamic Routers**: These routers have the capability to determine routes (and to find the optimum path among redundant routes) based on packet information and information obtained from other routers.

- Both static and dynamic routers use routing tables to pass packets on to remote sub networks.

❖ **Router : Advantages And Disadvantages :**

- **Advantages :**
- Can connect networks of different architecture
- Token Ring to Ethernet
- Choose best path through or to a network
- Create smaller collision domains
- Create smaller broadcast domains
- **Disadvantages :**
- Only work with routable protocols
- More expensive than hubs, bridges, and switches
- Routing table updates consume bandwidth.

❖ **Brouters :**

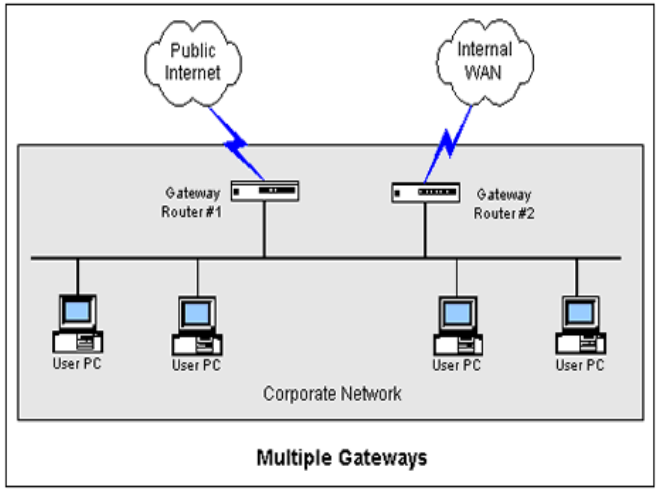
A brouter is a router that also can act as a bridge.

- A brouter attempts to deliver packets based on network protocol information, but if a particular Network layer protocol isn't supported, the brouter bridges the packet using device addresses.

- Hybrid device
- Functions as a router for routable protocols
- Functions as a bridge for non-routable protocols
- Operates at Data Link and Network layers.
- Operate at both the network layer for routable protocols and at the Data Link layer for non-routable protocols
- Handle both routable and non-routable features by acting as routers for routable protocols and bridges for non-routable protocols

❖ **Gateway :**

- A network gateway is an internetworking system capable of joining together two networks that use different base protocols.
- A network gateway can be implemented completely in software, completely in hardware, or as a combination of both. Depending on the types of protocols they support, network gateways can operate at any level of the OSI model.



❖ **WIRELESS NETWORK CONCEPTS:-**

→ A wireless network device is a type of electronic that can be used on or used to help facilitate a wireless Internet network. There are many different devices that can fit within this definition, and the number only

seems to be growing as wireless Internet becomes more and more widespread.

→ One of the most common examples is a router, which converts a fixed Internet signal into a wireless one and effectively creates a network that computers and other technology can connect to without the use of cables or other cords.

→ The things that are actually doing the connecting are also usually considered network devices, though, and this can include anything from computers and phones to e-readers and a wide range of "smart" gadgets. One of the core things that all have in common is the ability to both send and receive a wireless signal, and to connect instantaneously with other components on the same network.

❖ **Basic Concept**

→ In the world of Internet technology, wireless networks are becoming increasingly common. Not only are they growing in use in homes and offices, but they are also more and more commonplace in public areas like cafes, shopping malls, and restaurants.

→ The networks aren't usually worth much without devices that can jump onto them and use the signals provided for some other purpose.

→ Computers and many Internet-enabled phones use the signals primarily for browsing and e-mail capabilities. There are many more things that *can* be done, including storing and capturing data, routing cable television signals, and remotely controlling a range of things from garage door openers to light switches and home alarm systems.

❖ **How They Work**

→ This type of device works in cooperation with very specific wireless signals. The specifics tend to vary based on the engineering at issue as well as what it is the signal is being used to do, but in general, every device no matter its end purpose has both a sender and a receiver. The receiver picks up of the signal in the environment, and the sender converts it into something that can then be used for some other purpose within the network.

→ In order to take advantage of newer wireless signals and technologies, a wireless network device must usually be compatible with and up to date with that specific signal. Technology often changes more rapidly than product development. In many cases this requires upkeep and software installations on the user's part, and regular updating is usually very important.

❖ **Router Capabilities and Signal Division**

→ Wireless network devices can use a range of different signals. Much of this depends on the type of device and its purpose, as well as the sort of

router anchoring the network as a whole. Wireless routers can operate on band signals such as 802.11a, 802.11b, 802.11g, and 802.11n. 802.11n requires a router with specific built-in technology. It offers a stronger signal and greater range compared to other wireless router networks.

→ Different bands and signals are also what allow these sorts of devices to run simultaneously without interfering with each other. Many households have wireless Internet as well as cable TV with a remote control and web-based data storage, both of which can be seen as basic wireless devices. Using the remote to flick to another channel on a TV does not interfere with using the Internet, any more than checking e-mail on a phone while also streaming video on a laptop would. This is because different devices typically use different bands and signals.

❖ **Signal Conversions**

→ Aside from being a device that broadcasts wireless signals, a network device may also be one that receives signals. Many video game consoles are examples of this sort of device when they're able to sync with a home's wireless router to allow online game play and other features.

→ Other examples of these devices include USB wireless adapters, wireless cards, some wireless keyboards and mice, and wireless printers. Any device or peripheral that operates by connecting to a wireless network is usually considered to be a wireless network device.

UNIT 3

Chapter 6 : Network Protocols

❖ Protocol & Packet :

Protocol:

A pre-defined rule for data transfer for example: IP, TCP, UDP, HTTP, FTP, SMTP

- The rules and encoding specifications for sending data. The protocol defines the format and meaning of the data that is exchanged. The

protocols also determine whether the network uses a peer-to-peer or client/server architecture.

- **Protocol Roles:**
- Addressing and routing of messages
- Error detection
- Recovery
- Sequence and flow controls
- Example: HTTP protocol for communication between web browsers and servers.

Packet :

A packet is the unit of data that is routed between an origin and a destination on the Internet or any other packet-switched network.

- When any file (e-mail message, HTML file, Graphics Interchange Format file, Uniform Resource Locator request, and so forth) is sent from one place to another on the Internet, the Transmission Control Protocol (TCP) layer of TCP/IP divides the file into "chunks" of an efficient size for routing. Each of these packets is separately numbered and includes the Internet address of the destination. The individual packets for a given file may travel different routes through the Internet. When they have all arrived, they are reassembled into the original file (by the TCP layer at the receiving end).

- Packet" and "datagram" are similar in meaning. A protocol similar to TCP, the User Datagram Protocol (UDP) uses the term datagram.
- **In general, network packets contain the following:**
  - **Header:** The header signifies the start of the packet and contains a bundle of important parameters, such as the source and destination address and time/synchronization information.
  - **Data:** This portion of the packet contains the original data being transmitted.
  - **Trailer :** The trailer marks the end of the packet and typically contains error-checking (Cyclical Redundancy Check, or CRC) information

❖ TCP/IP :

- A The TCP/IP protocol suite (also commonly called the Internet protocol suite) was originally developed by the United States Department of Defense (DoD) to provide robust service on large internetworks that incorporate a variety of computer types.
- Main purpose of this protocol was for it to be hardware-independent.
- In some literature, the TCP/IP protocol suite is referred to as the DoD model.
- In recent years, the Internet protocols constitute the most popular network protocols currently in use.
- One reason for the popularity of TCP/IP is that no one vendor owns it, unlike the IPX/SPX, DNA, SNA, or AppleTalk protocol suites, all of which are controlled by specific companies.
- TCP/IP evolved in response to input from a wide variety of industry sources.

- TCP/IP is the most open of the protocol suites and is supported by the widest variety of vendors. Virtually every brand of computing equipment now supports TCP/IP.
- This has lead to some problems, though. Because TCP/IP is an open standard, sometimes one vendor's implementation of TCP/IP does not work with another's implementation.

OSI	TCP / IP(DoD)
Application (Layer 7)	Application(Process)
Presentation (Layer6)	
Session (Layer 5)	
Transport (Layer 4)	Host to Host (Transport)
Network (Layer 3)	Internet
Data Link (Layer 2)	Network Access (Subnet)
Physical (Layer 1)	

- The model for the Internet protocol suite has four layers (refer to Figure). From this model, you can see
  The approximate relationships of the layers.

The DoD (TCP/IP) model's layers function as follows :

- **The Network Access layer** corresponds to the bottom two layers of the OSI model. This Correspondence enables the DoD protocols to coexist with existing Data Link and Physical layer standards.
- **The Internet layer** corresponds roughly to the OSI Network layer. Protocols at this layer move data between devices on networks.
- **The Host-to-Host layer** can be compared to the OSI Transport layer. Host-to-Host protocols enable peer communication between hosts on the internetwork. (At the time these protocols were designed, personal computers and workstations didn't exist, and all network computers were host computers. As a result, devices on TCP/IP networks are typically referred to as hosts. The concept of a client/server relationship didn't exist, and all communicating hosts were assumed to be peers.)
- **The Process/Application** layer embraces functions of the OSI Session, Presentation, and Application layers. Protocols at this layer provide network services.

- One huge advantage of using TCP/IP is that TCP/IP is required for communication over the Internet; thus the Internet can be used as a communication backbone.

- A large number of protocols are associated with TCP/IP. These different protocols are grouped into the following unofficial categories:

  General TCP/IP Transport Protocols
  TCP/IP Services
  TCP/IP Routing

Advantages:

- Supports networking services better than the other Windows XP protocols
- Multiple routing protocols
- Good error detection and handling
- Works with most kinds of computers

Disadvantages:

- Not fast
- Not easy to use
- Requires
  - Fair degree of expertise
  - Careful planning
  - Constant maintenance and attention
- Mass of information and detail work.

❖ **FTP:-**

→ The **File Transfer Protocol** (**FTP**) is a standard network protocol used to transfer computer files from one host to another host over a TCP-based network, such as the Internet.

→ FTP is built on a client-server architecture and uses separate control and data connections between the client and the server. FTP users may authenticate themselves using a clear-text sign-in protocol, normally in the form of a username and password, but can connect anonymously if the server is configured to allow it. For secure transmission that protects the username and password, and encrypts the content, FTP is often secured with SSL/TLS (FTPS). SSH File Transfer Protocol (SFTP) is sometimes also used instead, but is technologically different.

→ The first FTP client applications were command-line applications developed before operating systems had graphical user interfaces, and are still shipped with most Windows, UNIX, and Linux operating systems. Many FTP clients and automation utilities have since been developed for desktops, servers, mobile devices, and hardware, and FTP has been incorporated into productivity applications, such as Web page editors.

❖ **HTTP:-**

→ HTTP functions as a request-response protocol in the client-server computing model.
→ A web browser, for example, may be the *client* and an application running on a computer hosting a web site may be the *server*. The client submits an HTTP *request* message to the server. The server, which provides *resources* such as HTML files and other content, or performs other functions on behalf of the client, returns a *response* message to the client.
→ The response contains completion status information about the request and may also contain requested content in its message body.
→ A web browser is an example of a *user agent* (UA). Other types of user agent include the indexing software used by search providers (web crawlers), voice browsers, mobile apps, and other software that accesses, consumes, or displays web content.

❖ **SMTP:-**

→ **Simple Mail Transfer Protocol** (**SMTP**) is an Internet standard for electronic mail (email) transmission. First defined by RFC 821 in 1982, it was last updated in 2008 with the Extended SMTP additions by RFC 5321—which is the protocol in widespread use today.
→ SMTP by default uses TCP port 25. The protocol for mail submission is the same, but uses port 587. SMTP connections secured by SSL, known as SMTPS, default to port 465 (nonstandard, but sometimes used for legacy reasons).
→ Although electronic mail servers and other mail transfer agents use SMTP to send and receive mail messages, user-level client mail applications typically use SMTP only for sending messages to a mail server for relaying. For receiving messages, client applications usually use either POP3 or IMAP.
→ Although proprietary systems (such as Microsoft Exchange and IBM Notes) and webmail systems (such as Outlook.com, Gmail and Yahoo! Mail) use their own non-standard protocols to access mail box accounts on their own mail servers, all use SMTP when sending or receiving email from outside their own systems.

❖ **POP3:-**

→ In computing, the **Post Office Protocol** (**POP**) is an application-layer Internet standard protocol used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection.
→ POP has been developed through several versions, with version 3 (**POP3**) being the last standard in common use before largely made obsolete by the more advanced IMAP.
→ POP supports simple download-and-delete requirements for access to remote mailboxes (termed mail drop in the POP RFC's).Although most POP clients have an option to leave mail on server after download, e-mail clients using POP generally connect, retrieve all messages, store them on the user's PC as new messages, delete them from the server, and then

disconnect. Other protocols, notably IMAP, (Internet Message Access Protocol) provide more complete and complex remote access to typical mailbox operations.
→ In the late 1990s and early 2000s, fewer Internet Service Providers (ISPs) supported IMAP due to the storage space that was required on the ISP's hardware. Contemporary e-mail clients supported POP, then over time popular mail client software added IMAP support.
→ A POP3 server listens on well-known port 110. Encrypted communication for POP3 is either requested after protocol initiation, using the STLS command, if supported, or by POP3S, which connects to the server using Transport Layer Security (TLS) or Secure Sockets Layer (SSL) on well-known TCP port 995.
→ Available messages to the client are fixed when a POP session opens the maildrop, and are identified by message-number local to that session or, optionally, by a unique identifier assigned to the message by the POP server. This unique identifier is permanent and unique to the maildrop and allows a client to access the same message in different POP sessions. Mail is retrieved and marked for deletion by message-number. When the client exits the session, the mail mark.

❖ **SNMP:-**

→ **Simple Network Management Protocol** (**SNMP**) is an "Internet-standard protocol for managing devices on IP networks". Devices that typically support SNMP include routers, switches, servers, workstations, printers, modem racks and more.[1] SNMP is widely used in network management systems to monitor network-attached devices for conditions that warrant administrative attention. SNMP is a component of the Internet Protocol Suite as defined by the Internet Engineering Task Force (IETF). It consists of a set of standards for network management, including an application layer protocol, a database schema, and a set of data objects.
→ SNMP exposes management data in the form of variables on the managed systems, which describe the system configuration. These variables can then be queried (and sometimes set) by Managing applications.

❖ **TELNET:-**

→ **Telnet** is an application layer protocol used on the Internet or local area networks to provide a bidirectional interactive text-oriented communication facility using a virtual terminal connection. User data is interspersed in-band with Telnet control information in an 8-bit byte oriented data connection over the Transmission Control Protocol (TCP).
→ Telnet was developed in 1969 beginning with RFC 15, extended in RFC 854, and standardized as Internet Engineering Task Force (IETF) Internet Standard STD 8, one of the first Internet standards.
→ Historically, Telnet provided access to a command-line interface (usually, of an operating system) on a remote host. Most network equipment and

operating systems with a configuration (including systems based on Windows NT). However, because of serious security concerns when using Telnet over an open network such as the Internet, its use for this purpose has waned significantly in favor of SSH.

→ The term *telnet* is also used to refer to the software that implements the client part of the protocol. Telnet client applications are available for virtually all computer platforms. *Telnet* is also used as a verb. *To telnet* means to establish a connection with the Telnet protocol, either with command line client or with a programmatic interface. For example, a common directive might be: "*To change your password, telnet to the server, log in and run the passed command.*" Most often, a user will be *telnetting* to a Unix-like server system or a network device (such as a router) and obtaining a login prompt to a command line text interface or a character-based full-screen manager.

❖ **ARP:-**

→ The **Address Resolution Protocol** (**ARP**) is a telecommunication protocol used for resolution of network layer addresses into link layer addresses, a critical function in multiple-access networks. ARP was defined by RFC 826 in 1982.It is Internet Standard STD 37. It is also the name of the program for manipulating these addresses in most operating systems.

→ ARP is used for mapping a network address to a physical address like an Ethernet address . ARP has been implemented with many combinations of network and data link layer technologies, like IPv4, Chaosnet, DECnet and Xerox PARC Universal Packet (PUP) using IEEE 802 standards, FDDI, X.25, Frame Relay and Asynchronous Transfer Mode (ATM). IPv4 over IEEE 802.3 and IEEE 802.11 is the most common case.

→ In Internet Protocol Version 6 (IPv6) networks, the functionality of ARP is provided by the Neighbor Discovery Protocol (NDP).

❖ **RARP:-**

→ The **Reverse Address Resolution Protocol** (**RARP**) is an obsolete computer networking protocol used by a client computer to request its Internet Protocol (IPv4) address from a computer network, when all it has available is its Link Layer or hardware address, such as a MAC address. The client broadcasts the request, and does not need prior knowledge of the network topology or the identities of servers capable of fulfilling its request.

→ RARP is described in Internet Engineering Task Force (IETF) publication RFC 903.It has been rendered obsolete by the Bootstrap Protocol (BOOTP) and the modern Dynamic Host Configuration Protocol (DHCP), which both support a much greater feature set than RARP.

→ RARP requires one or more server hosts to maintain a database of mappings of Link Layer addresses to their respective protocol addresses. Media Access Control (MAC) addresses needed to be individually configured on the servers by an administrator. RARP was limited to serving only IP addresses.

→ Reverse ARP differs from the Inverse Address Resolution Protocol (InARP) described in RFC 2390, which is designed to obtain the IP address associated with a local Frame Relay data link connection identifier. InARP is not used in Ethernet.

❖ **IPX/SPX:**

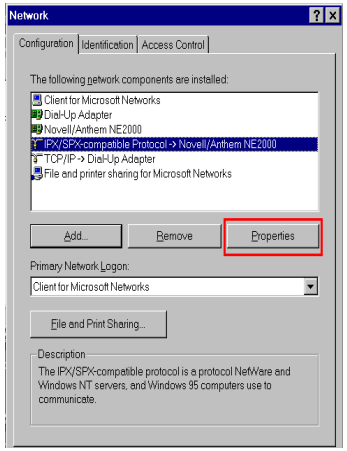
- The NetWare protocols have been designed with a high degree of modularity.
- This modularity makes the NetWare protocols adaptable to different hardware and simplifies the task of incorporating other protocols into the suite.
- Windows NT doesn't use the IPX/SPX suite to communicate with NetWare resources.
- Microsoft instead developed a clone of IPX/SPX called NWLink—IPX/SPX Compatible Transport.
- IPX/SPX is generally smaller and faster than TCP/IP and, like TCP/IP, it is routable.
- However, it operates down to the Data Link layer of the OSI model so it is more dependent upon hardware devices than the TCP/IP protocol.
- IPX/SPX makes up the protocol suite that is used to transfer information on networks running the Novell NetWare operating system.

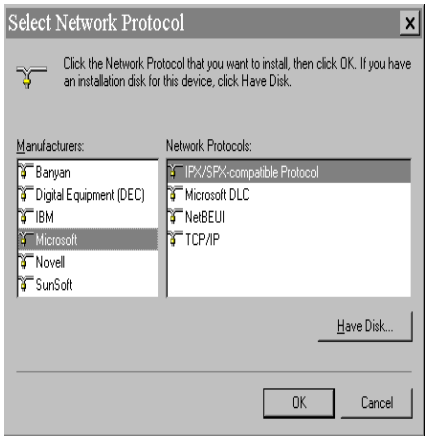
- **Internetwork Packet Exchange (IPX)** -Transfer information between devices.
  - Internetwork Packet Exchange
  - Connectionless protocol
  - Resides at the Network Layer
  - Responsible for network addressing and routing

- **Sequenced Packet Exchange (SPX)** - An extension of the IPX protocol.
  - Sequenced Packet Exchange
  - Connection-Oriented protocol
  - Resides at the Transport Layer
  - Uses services provided by IPX
  - Ensures reliable data delivery and manages sessions

- **IPX/SPX Advantages/Disadvantages:**

  - Used mostly with Novell NetWare networks
  - Common and outperforms TCP/IP
  - Not supported on the Internet
  - Typically found in private networks





IPX/SPX:

❖ **Apple Talk :**

- AppleTalk is the computing architecture developed by Apple Computer for the Macintosh family of personal computers.
- Although AppleTalk originally supported only Apple's proprietary Local Talk cabling system, the suite has been expanded to incorporate both ethernet and token-ring Physical layers.
- Within Microsoft operating systems, AppleTalk is only supported by Windows NT Server. Windows NT Workstation and Windows 95 do not support AppleTalk.
- AppleTalk cannot be used for Microsoft-to-Microsoft operating system communication. It can be used only through Windows NT servers supporting Apple clients.
- AppleTalk originally supported networks of limited scope. The AppleTalk Phase II specification issued in 1989, however, extended the scope of AppleTalk to enterprise networks.

- Used only in Apple networks
- Routable Protocol
- AppleTalk Phase II allows this protocol to work with others
- AppleTalk divides groups of computers into Zones instead of Domains

❖ **Server Messaging Blocks (SMB) :**

One of the most popular protocols for PCs lets you share files, disks, directories, printers, and (in some cases) even COM ports across a network: this protocol is called the SMB (Server Message Block) standard.

- An SMB client or server can communicate with just about any other similar program.

❖ **NETBIOS NAMES :**

- Implement a NetBIOS naming scheme for all computers on a given network.
- NetBIOS is an interface that provides NetBIOS-based applications with access to network resources.
- Every computer on a Windows NT network must have a unique name for it to be accessible through the NetBIOS interface. This unique name is called a computer name or a NetBIOS name.
- NetBIOS (Network Basic Input/output System) is an application interface that provides
  PC-based applications with uniform access to lower protocol layers. NetBIOS was once most closely associated with the NetBEUI protocol—NetBEUI, in fact, is an abbreviation for NetBIOS Extended User Interface.

Assigning NetBIOS Names

- On a NetBIOS network, every computer must have a unique name.
- The computer name must be 15 characters long or fewer.
- NetBIOS name can include alphanumeric characters and any of the following special characters:
- NetBIOS names are not case-sensitive.

❖ **Logical Link Control and Adaptation Protocol(L2CAP):**

- The Logical Link Control and Adaptation Layer Protocol (L2CAP) is layered over the Baseband Protocol and resides in the data link layer.

- L2CAP provides connection-oriented and connectionless data services to upper layer protocols with protocol multiplexing capability, segmentation and reassembly operation, and group abstractions.
- L2CAP permits higher level protocols and applications to transmit and receive L2CAP data packets up to 64 kilobytes in length.

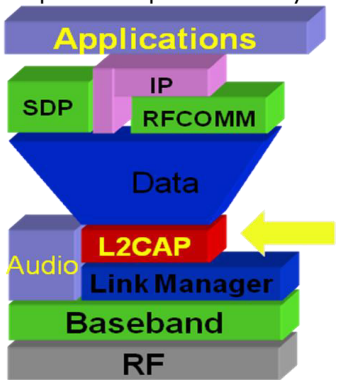


Fig : A hardware/software/protocol description(For Bluetooth)

L2CAP has many functions:

- Multiplexing to allow several higher layer links, possibly based on different protocols.
- Segmentation and reassembly to allow transfer of larger packets than lower layers support.
- Quality of Service (QoS) management for higher layer protocols
- Optional error control and retransmissions.
- All applications must use L2CAP to send data. It is also used by Bluetooth's higher layers such as RFCOMM
- L2CAP provides the facilities needed by higher layer protocols to communicate across a Bluetooth link:

❖ RFCOMM :

- RFCOMM is a simple transport protocol, which provides emulation of RS232 serial ports over the L2CAP protocol.

- The protocol is based on the ETSI (**European Telecommunications Standards Institute)** standard TS 07.10. a standard in the Telecom industry.

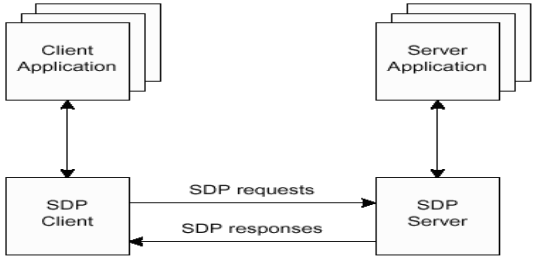
- The RFCOMM protocol supports up to 60 simultaneous connections between two BT devices. The number of connections that can be used simultaneously in a BT device is implementation-specific.

RFCOMM is commonly used:

- For Bluetooth GPS Receivers.
- To provide Bluetooth to legacy devices which used RS232 before. E.g., laser BarCode scanners, early mobile/foldable Bluetooth keyboards, wireless interface for hobby embedded system and robotic projects.
- As a basis for other legacy protocols like OBEX and PPP.

❖ Service Discovery Protocol (SDP) :

- The service discovery protocol (SDP) provides a means for applications to discover which services are available and to determine the characteristics of those available services.
- A specific Service Discovery protocol is needed in the Bluetooth environment, as the set of services that are available changes dynamically based on the RF proximity of devices in motion, qualitatively different from service discovery in traditional network-based environments.
- The service discovery protocol defined in the Bluetooth specification is intended to address the unique characteristics of the Bluetooth environment.
- SDP is a simple protocol with minimal requirements on the underlying transport.
- It can function over a reliable packet transport
- SDP uses a request/response model where each transaction consists of one request protocol data unit (PDU) and one response PDU



❖ Common Address Redundancy Protocol (CARP):

- The Common Address Redundancy Protocol, or CARP allows multiple hosts to share the same IP address.
- CARP is a free alternative to the Virtual Router Redundancy Protocol (VRRP).
- Its primary purpose is to allow multiple hosts on the same network segment to share an IP address. CARP works by allowing a group of hosts on the same network segment to share an IP address.
- This group of hosts is referred to as a redundancy group. The redundancy group is assigned an IP address that is shared among the group members.
- Within the group, one host is designated the master and the rest as back-ups.
- The master host is the one that currently holds the shared IP.
- It responds to any traffic or ARP requests directed towards it. Each host may belong to more than one redundancy group at a time
- CARP is a multicast protocol. It groups several physical computers together under one or more virtual addresses. Of these, one system is the master and responds to all packets destined for the group, the other systems act as hot spares.
- No matter what the IP and MAC address of the local physical interface, packets sent to the CARP address are returned with the CARP information.

CH 7. NETWORK ROUTING

❖ Routing protocol :

- A routing protocol is a protocol that specifies how routers communicate with each other, disseminating information that enables them to select routes then any two nodes on a computer network, the choice of the route being done by routing algorithms.
- Each router has a priori knowledge only of networks attached to it directly.
- A routing protocol shares this information first among immediate neighbors, and then throughout the network.
- This way, routers gain knowledge of the topology of the network.

There are many types of routing protocols; three major classes are in widespread use on IP networks:

- **Interior gateway routing** via **link-state routing protocols**, such as **OSPF** and **IS-IS**

- **Interior gateway routing** via **path vector or distance vector protocols**, such as **RIP**, **IGRP** and **EIGRP**
- **Exterior gateway routing**. **BGP** v4 is the routing protocol used by the public Internet.

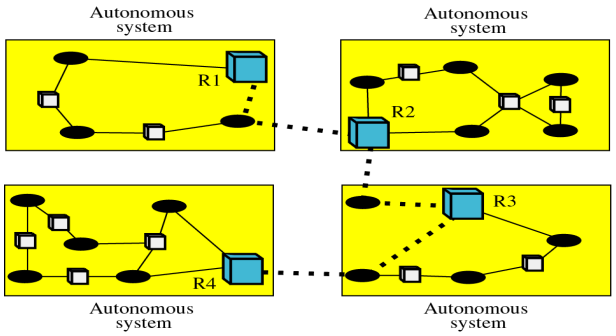
- **Link State Routing (Protocol):**

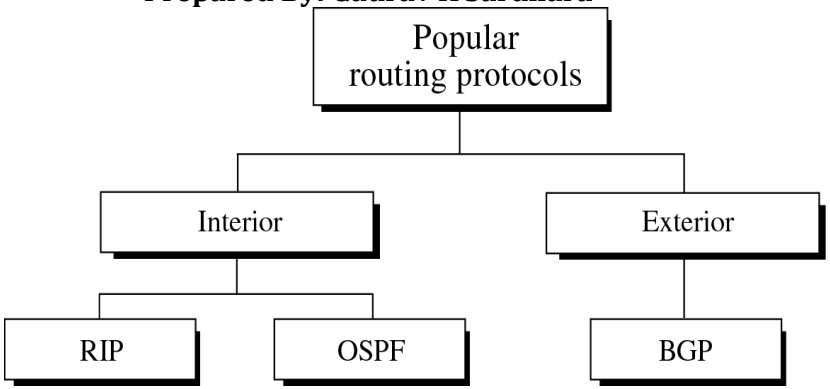
- The link-state protocol is performed by every switching node in the network (i.e. nodes that are prepared to forward packets; in the Internet, these are called routers).
- The basic concept of link-state routing is that every node constructs a map of the connectivity to the network, in the form of a graph, showing which nodes are connected to which other nodes. Each node then independently calculates the next best logical path from it to every possible destination in the network. The collection of best paths will then form the node's routing table.

- **Distance Vector Routing (Protocol):**

- A distance-vector routing protocol requires that a router informs its neighbors of topology changes periodically and, in some cases, when a change is detected in the topology of a network.
- Compared to link-state protocols, which require a router to inform all the nodes in a network of topology changes, distance-vector routing protocols have less computational complexity and message overhead

- **Autonomous System (AS) :** Collection of networks with same policy
  - Single routing protocol. Usually under single administrative control.





❖ **Interior Gateway Protocols (IGPs) :**

- Interior Gateway Protocols (IGPs) exchange routing information within a single routing domain. A given autonomous system can contain multiple routing domains, or a set of routing domains can be coordinated without being an Internet-participating autonomous system.

  Common examples include:

- **IGRP** (Interior Gateway Routing Protocol)
- **EIGRP** (Enhanced Interior Gateway Routing Protocol)
- **OSPF** (Open Shortest Path First)
- **RIP** (Routing Information Protocol)
- **IS-IS** (Intermediate System to Intermediate System)

■ **Interior Gateway Routing Protocol (IGRP) :**

- Interior Gateway Routing Protocol (IGRP) is a distance vector interior routing protocol (IGP) invented by Cisco.
- It is used by routers to exchange routing data within an autonomous system.
- IGRP was created in part to overcome the limitations of RIP (maximum hop count of only 15, and a single routing metric) when used within large networks.
- IGRP supports multiple metrics for each route, including bandwidth, delay, load, MTU, and reliability; to compare two routes these metrics are combined together into a single metric, using a formula which can be adjusted through the use of pre-set constants.
- The maximum hop count of IGRP-routed packets is 255 (default 100), and routing updates are broadcast every 90 seconds (by default).
- IGRP is considered a classful routing protocol. Because the protocol has no field for a subnet mask,.

- Classful protocols have become less popular as they are wasteful of IP address space.

■ **Enhanced Interior Gateway Routing Protocol (EIGRP) :**

- Enhanced Interior Gateway Routing Protocol - (EIGRP) is a Cisco proprietary routing protocol loosely based on their original IGRP. EIGRP is an advanced distance-vector routing protocol, with optimizations to minimize both the routing instability incurred after topology changes, as well as the use of bandwidth and processing power in the router.
- Widely used in many enterprise networks and in some ISP networks
- Multiprotocol (supports more than IP)
- Exhibits good scalability and rapid convergence
- Supports unequal cost load balancing.

■ **Open Shortest Path First (OSPF) :**

- Open Shortest Path First (OSPF) is a dynamic routing protocol for use in Internet Protocol (IP) networks.
- Specifically, it is a link-state routing protocol and falls into the group of interior gateway protocols, operating within a single autonomous system (AS).
- OSPF is perhaps the most widely-used interior gateway protocol (IGP) in large enterprise networks.
- It gathers link state information from available routers and constructs a topology map of the network.
- The topology determines the routing table presented to the Internet Layer which makes routing decisions based solely on the destination IP address found in IP diagrams.
- OSPF was designed to support variable-length subnet masking (VLSM) or Classless Inter-Domain Routing (CIDR) addressing models.
- OSPF detects changes in the topology, such as link failures, very quickly and converges on a new loop-free routing structure within seconds. It computes the shortest path tree for each route using a method based on Dijkstra's algorithm, a shortest path first algorithm.
- OSPF uses multicast addressing for route flooding on a broadcast network link.
- OSPF Version 2 for IPv4. The updates for IPv6 are specified as OSPF Version 3

■ **Routing Information Protocol (RIP) :**

- The Routing Information Protocol (RIP) is a dynamic routing protocol used in local and wide area networks.
- As such it is classified as an interior gateway protocol (IGP).
- It uses the distance-vector routing algorithm.

- It was first defined in RFC 1058 (1988).
- The protocol has since been extended several times, resulting in RIP Version 2 (RFC 2453).
- Both versions are still in use today, however, they are considered to have been made technically obsolete by more advanced techniques such as Open Shortest Path First (OSPF) and the OSI protocol IS-IS.

- RIP has also been adapted for use in IPv6 networks, a standard known as RIPng (RIP next generation), published in RFC 2080 (1997).
- There are three versions of the Routing Information Protocol: RIPv1, RIPv2, and RIPng.

■ **Intermediate system to intermediate system (IS-IS) :**

- Intermediate system to intermediate system (IS-IS), is a protocol used by network devices (routers) to determine the best way to forward diagrams through a packet-switched network, a process called routing.
- IS-IS is a link-state routing protocol, meaning that it operates by reliably flooding Link State information throughout a network of routers. Each router then independently builds a picture of the network's topology. Packets or datagrams are forwarded based on the best topological path through the network to the destination.
- The protocol was defined in ISO/IEC 10589:2002 as an international standard within the Open Systems Interconnection (OSI) reference design.
- IS-IS is not an Internet standard, however IETF republished the standard in RFC 1142 for the Internet community.

❖ **Exterior Gateway Protocol (EGP):**

- Exterior Gateway Protocol (EGP) is a now obsolete routing protocol for the Internet originally specified in 1982 by Eric C. Rosen of Bolt, Beranek and Newman, and David L. Mills.
- EGP is a simple reach ability protocol, and, unlike modern distance-vector and path-vector protocols, it is limited to tree-like topologies.

- examples include:
- **BGP** (Border Gateway Protocol).

■ **Border Gateway Protocol** (**BGP**):

- The Border Gateway Protocol (BGP) is the core routing protocol of the Internet.
- It maintains a table of IP networks or 'prefixes' which designate network reach ability among autonomous systems (AS).
- It is described as a path vector protocol.

- BGP does not use traditional Interior Gateway Protocol (IGP) metrics, but makes routing decisions based on path, network policies and/or rule sets.
- BGP was created to replace the Exterior Gateway Protocol (EGP) routing protocol to allow fully decentralized routing.
- This allowed the Internet to become a truly decentralized system.

Operation:

- BGP neighbors, or peers, are established by manual configuration between routers to create a TCP session on port 179.
- A BGP speaker will periodically send 19-byte keep-alive messages to maintain the connection (every 60 seconds by default). Among routing protocols,
- BGP is unique in using TCP as its transport protocol.
- When BGP is running inside an autonomous system (AS), it is referred to as Internal BGP (IBGP or Interior Border Gateway Protocol).
- When it runs between autonomous systems, it is called External BGP (EBGP or Exterior Border Gateway Protocol).
- Routers on the boundary of one AS, exchanging information with another AS, are called border or edge routers.
- In the Cisco operating system, IBGP routes have an administrative distance of 200, which is less preferred than either external BGP or any interior routing protocol.
- Other router implementations also prefer EBGP to IGPs, and IGPs to IBGP.
- BGP Can be used in two scenarios
  - Internally, inside an Autonomous System
    - Known as **IBGP** (Internal BGP)
  - Externally, between peers
    - Known as **EBGP** (External BGP)

❖ **Types of Routing**

→ There are three basic methods to build a routing table on any Networking Device.

1. Static Routing
2. Dynamic Routing
3. Default Routing

→ **Static Routing:**

→ A **static routing** occurs when you manually add routes in each Router's Routing table, Routing table should create, maintain, and update by a network administrator, manually.

→ A static route to every network must be configured on every router for full connectivity. This provides a granular level of control over routing, but quickly becomes impractical on large networks.

→ Routers will not share static routes with each other, thus reducing CPU/RAM overhead and saving bandwidth. However, static routing is not fault-tolerant, as any change to the routing infrastructure (such as a link going down, or a new network added) requires manual intervention.

→ Routers operating in a purely static environment cannot seamlessly choose a better Route if a link becomes unavailable. Static routes have an Administrative Distance (AD) of **1**, and thus are always preferred over dynamic routes, unless the default AD is changed. A **static route** with an **adjusted AD** is called a **floating static route.**

- The following briefly outlines the advantages and disadvantages of static routing:

→ **Advantages of Static Routing:**

1. Minimal CPU/Memory overhead
2. There is no bandwidth update between Routers, which means you will Save bandwidth on WAN links.
3. It adds security because the administrator can choose Routing access to certain networks only.

→ **Disadvantages of Static Routing:**

1. If any Infrastructure changes must be manually adjust the Configuration in complete network.
2. No "dynamic" fault tolerance if a link goes down
3. Administrator must understand the complete internetwork and how each Router connected to configure properly.
4. Impractical on large network

→ **<u>Dynamic Routing:</u>**

→ A Dynamic Routing is when protocols are used to find networks and update Routing tables on Routers.

→ A **dynamic** routing table is created, maintained, and updated by a routing protocol running on the router. Examples of routing protocols include **RIP** (Routing Information Protocol), **EIGRP** (Enhanced Interior Gateway Routing Protocol), and **OSPF** (Open Shortest Path First).

→ Routers do share dynamic routing information with each other, which increases CPU, RAM, and bandwidth usage. However, routing protocols are capable of dynamically choosing a different (or better) path when there is a change to the routing infrastructure.

❖ **Difference between Routing Protocols and Routed Protocols**

→ Do not confuse **routing protocols** with **routed protocols**:

→ A **routed protocol** is a Layer 3 protocol that applies logical addresses to devices and routes data between networks (such as IP)

→ A **routing protocol** dynamically builds the network, topology, and next hop information in routing tables (such as RIP, EIGRP, etc.)

→ The following briefly outlines the advantages and disadvantages of dynamic routing:

→ **Advantages of Dynamic Routing:**

1. Simpler to configure on larger networks
2. Will dynamically choose a different (or better) route if a link goes down
3. Ability to load balance between multiple links

→ **Disadvantages of Dynamic Routing:**

1. Updates are shared between routers, thus consuming bandwidth
2. Routing protocols put additional load on router CPU/RAM
3. The choice of the "best route" is in the hands of the routing protocol, and not the network administrator.

→ **<u>Default Routing:</u>**

→ A default routing is used only when one exit path from the Router. Default Routing can configure like a static route with the ip route command, but use 0.0.0.0 0.0.0.0 for the IP network and subnet mask followed by the next hop router's IP address or Exit interface of the local Router.

→ Have to also use the ip classless command since there aren't any routes in the routing table. ip classless is enabled by default in IOS version 12.

- **Command Syntax for Default Routing on Cisco Router**
- Router(config)#ip route 0.0.0.0 0.0.0.0 192.168.1.2
  Router (config) #ip classless.

UNIT 4

Chapter 8 : IP Addressing

❖ **IP Address:**

- Unique logical address of any computer consider as a ip address.
- It has 2 types:-

1) IPv4
2) IPv6

❖ **IPv4 :-**

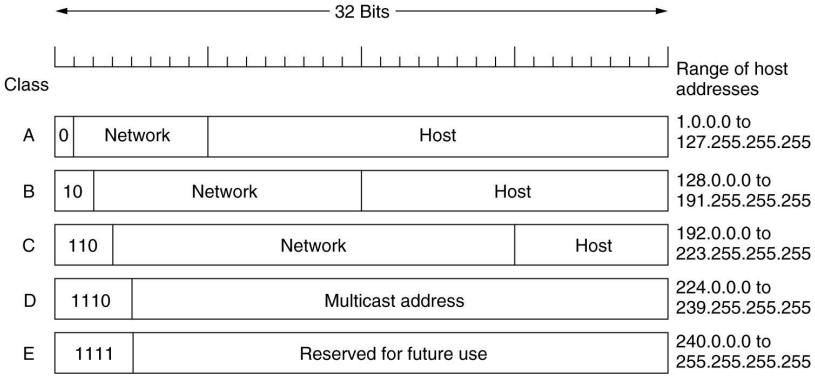
- The IP address is a 32 bit address.
- Identifies the network and the host on a given network.
- Divided into two parts first part identifies the network, second part identifies the host on the network
- The format is not the same for each address.

- The 32 bit number is represented in the following format.
- xxx.xxx.xxx.xxx
- Where xxx is the decimal representation of the binary bit string.
- Example: 142.110.3.4

↓

10001110 01101110 00000011 00000100

❖ Classes of IP Addresses :



■ Class A :

- Used for small number of networks and large number of hosts.
- First byte (8 bits) represents the network address.
- Last 3 bytes (24 bits) represent the host address.
- Class A addresses have a first bit of 0.
- Class A network addresses range from 0 to 127.

■ Class B :

- Provide an equal number of networks and hosts.
- First two bytes are network address and last two bytes are host addresses.
- First two bits of a class B address are 10.
- Network addresses range from 128 to 191.

■ Class C:

- Greater number of network addresses fewer host addresses.
- First three bits are 110.
- Network addresses range form 192-223.

■ Class D :

- Used for special multicast addresses.
- First four bits 1110.

■ Class E :

- Used for experimental purposes
- first four bits 1111

❖ Subnets :

- Subnets are used to divide a large network into smaller networks.
- Each address allows for one network address and many hosts (ie all hosts are on the same network).
- Subnet masks are used to create many subnets within the same network address.

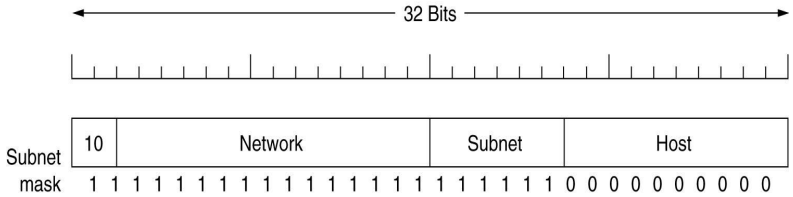
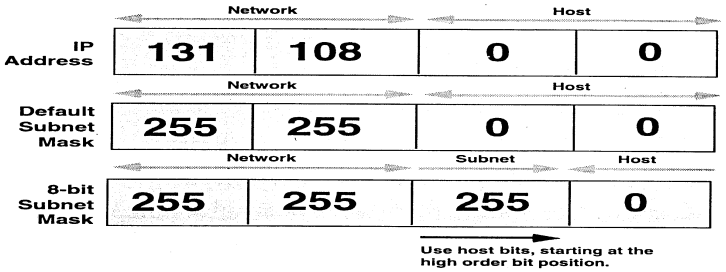


Fig: A class B network sub netted into 64 subnets

❖ Subnet masks :

- A bit string applied to an address.
- If the bit is on the corresponding bit in the address is considered to be a network bit.
- The network mask is known locally only.



- Layout of the subnet mask
  - binary 1 for network bits
  - binary 1 for subnet bits
  - binary 0 for host bits

Fig: Subnet Mask for Class B address

Address Class	High-Order Bits	First Octet Address Range	Number of Bits in the Network Address	Number of Networks	Number of Hosts per Network
Class A	0	0-127	8	126	16,777,216
Class B	10	128-191	16	16,384	65,536
Class C	110	192-223	24	2,097,152	254
Class D	1110	224-239	28	N/A	N/A

❖ Sub netting :

- A network is divided into several smaller networks with each sub network (or subnet) having its sub network address.

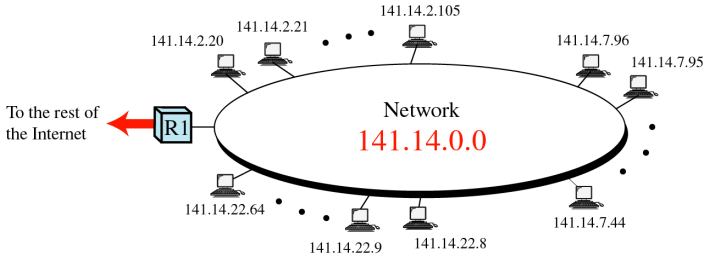


Fig: Not sub netted

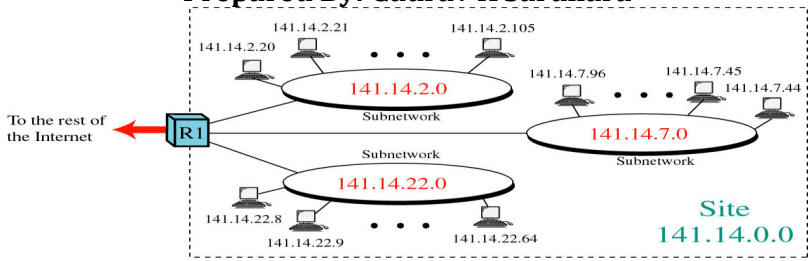


Fig: Sub netted

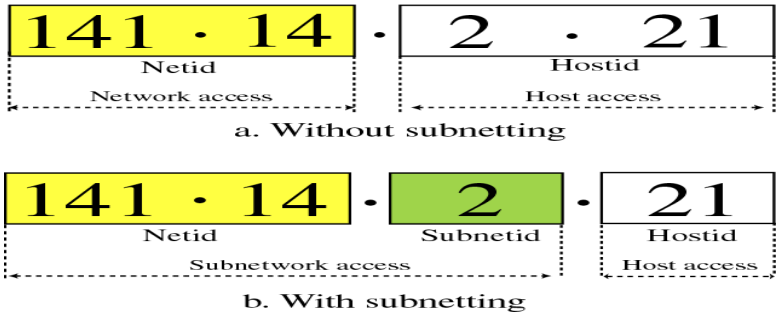


Fig: Three levels of hierarchy : netid, subnetid, and hostid

❖ Super netting :

- Combining several class C addresses to create a larger range of addresses.

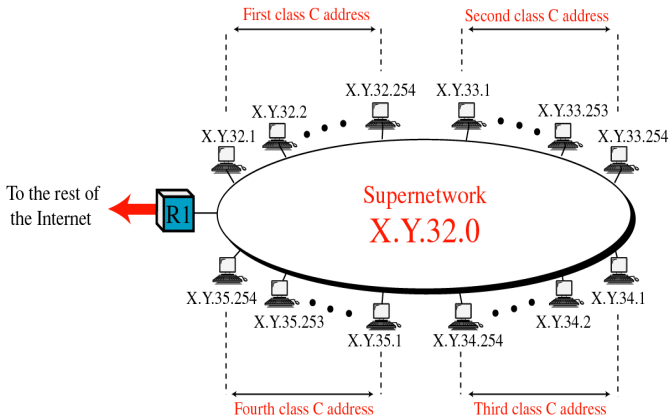
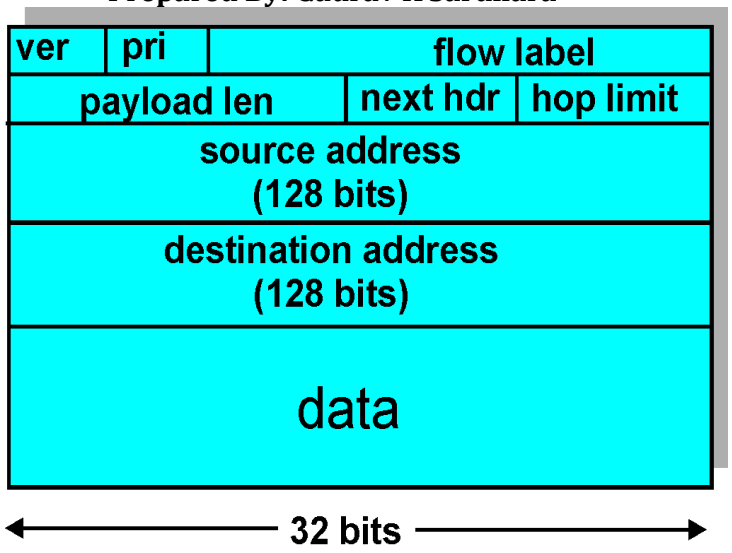


Fig: 4 class C addresses combine to make one super network

❖ IPV6 :

- IPv6 was designed to take an evolutionary step from IPv4.. Functions that work in IPv4 was kept in IPv6.Functions that didn't work were removed. The changes from IPv4 to IPv6 fall primarily into the following categories:
- Header Format Simplification
- Improved Support for Options
- Expanded Routing and Addressing Capabilities
- Quality-of-Service Capabilities
- Authentication and Privacy Capabilities

- IPV6 is Datagram Protocol.
- Routing via RIP, OSPF, IS-IS, BGP
- End-to-end reliability via TCP
- Semantics are very similar to IPv4
- Larger addresses
- More emphasis on security
- IPv6 is a Network Protocol with many more addresses than IPv4:
- 340,282,366,920,938,463,374,607,431,768,211,456 available addresses.
- With so many addresses we can overcome the shortage in IPv4 supply and continuing support the growth of Internet.
- In IPv6 some tasks are simpler than in IPv4: (Auto-configuration, Renumbering, Multicast, IP Mobility, etc.)
- IPv6 Enables Innovation. Particularly for applications without NAT .
- IPv6 addresses are represented by Hexadecimal numbers.
- Example: 2001:DB8:12FF:1231:FFB5::F9DA/64.
- In IPv6 there is not Network Mask, only Prefix Length.
- In IPv6 the header is always 40 bytes long, extensions are listed as "next header".
- In IPv6 there is no Broadcast, only Multicast.
- In IPv6 there is no ARP or IGMP, ICMPv6 takes those jobs.
- IPv6 has 128-bit (16-byte) source and destination IP addresses. Although 128 bits can express over $3.4 \square 10^{38}$ possible combinations, the large address space of IPv6 has been designed to allow for multiple levels of subnetting and address allocation from the Internet backbone to the individual subnets within an organization.



- IPv6 header is simpler than IPv4
  - IPv4: 14 fields, variable length (20 bytes +)
  - IPv6: 8 fields, fixed length (40 bytes)
- Header fields eliminated in IPv6
  - Header Length
  - Identification
  - Flag
  - Fragmentation Offset
  - Checksum
- Header fields enhanced in IPv6
  - Traffic Class
  - Flow Label

Chapter 9: Windows Server 2003

❖ Installing and Configuring Windows Server 2003 :

As an experienced IT professional, you have no doubt spent considerable time installing Windows platforms.

Some of the important and enhanced considerations when installing Windows Server 2003 are :

- **Bootable CD-ROM installation :**
  Windows Server 2003 can be installed directly from the CD-ROM.
  There is no support for starting installation from floppy disks.

- **Improved graphical user interface (GUI) during setup :**
  Windows Server 2003 uses a GUI during setup that resembles that of Windows XP

- **Product activation :**
  Retail and evaluation versions of Windows Server 2003 require that you activate the product.
  Volume licensing programs, such as Open License, Select License, or Enterprise Agreement, do not require activation.

Following Step required installing Windows 2003.

1. Configure the computer's BIOS or the disk controller BIOS to boot from CD-ROM. If you are not sure how to configure your computer or disk controller to boot from CD-ROM, consult your hardware documentation.

2. Insert the Windows Server 2003 installation CD-ROM into the CD-ROM drive and restart the computer.

3. After configuring the system for booting from a CD, the Windows Setup screen appears.
At this point, Setup is loading the driver files it needs to continue with installation.

4. The "Time Limited" warning screen appears with the option of Continuing Setup or Quitting.
Press ENTER to Continue Setup or F3 to Quit and reboot the system.

5. The "Welcome to Setup" screen appears with the option of Continuing Setup, Repair a previous installation, or Quitting.
- Press ENTER to Continue Setup.
- You may also choose R to Repair or F3 to Quit and reboot the system.

6. The "Windows Licensing Agreement" screen, otherwise known as "EULA," displays the legal in's and out's of this particular software package.
- You may press F8 to signify that you agree with the terms, hit ESC if you do not agree and PAGE UP or PAGE DOWN to scroll through each screen. Note: If you do not agree to the terms, setup will quit and reboot the system.

7. Hard drive partition information is now displayed. This varies with each systems hardware configuration.

8. Continue to create partitions until all space is used or the configuration meets your requirements. Note: a small portion will be unavailable to partition. This is normal. In this example, it is 8 MB.

9. Select what format you wish to use by pressing the UP ARROW and DOWN ARROW keys.
Press ENTER to confirm your selection and Continue or ESC to Cancel.

10. Watch the progress bar as Setup formats the partition, It may take awhile.

11. After the partition is a finished formatting, Setup copy various files to support booting from the hard drive and continue on.

12. The first reboot and the end of the blue background has arrived. If you are impatient, press ENTER to Restart before the 15 seconds expire.
ENTER to Restart the Computer is the only option available.

13. The new Windows Server 2003 Family boot screen is displayed.

14. Windows Server 2003 Installation:
Sit back. It may be awhile.

15. Region and Input Languages Option

16. Enter in your Name and optional Organization information, and then select the Next button.
Select the **Next** button to continue.

17. Enter your unique 25 digit Product Key that came with your CD or download, and then select the **Next** button.

18. Configure the Licensing mode.
- Options are Per server or Per seat.
- Per server is usually used for a single-server network.

- Per client is used if all of the concurrent connections is higher than the number of clients or seats that you have.

19. Enter an Administrator Password now. It is very important that you keep this information safe and remember what it is!
Select the **Next** button to continue.

20. If the chosen password does not meet acceptable guidelines, a warning box will appear with suggestions on how to make the system more secure.

21. Configure the proper information for the Date, Time and Time Zone here.
Select the **Next** button to continue.

22. Faster development of applications, but still slow install times. Wait here while the Network is installing.

23. The Network Settings Dialog is next. Under usual circumstances, the **Typical settings** are fine.
Choose your method and select the **Next** button.

24. Custom settings for network (Network Load Balancing).

25. File and Print Sharing for Microsoft Networks has several options not available with Windows XP Home or Professional.

26. TCP/IP Properties contains the standard options. Adjust them for your particular needs as required. Select the **Advanced** button to further configure your TCP/IP options.

27. **Select** Workgroup or Computer Domain.
Select the **Next** button after making your choice.

28. Install screen appear.
The system will reboot after all files have been copied over to the install partition. Now may be a good time to take a break. It may be awhile.

29. The moment we have all been waiting for, Windows 2003 is starting up for the first time.

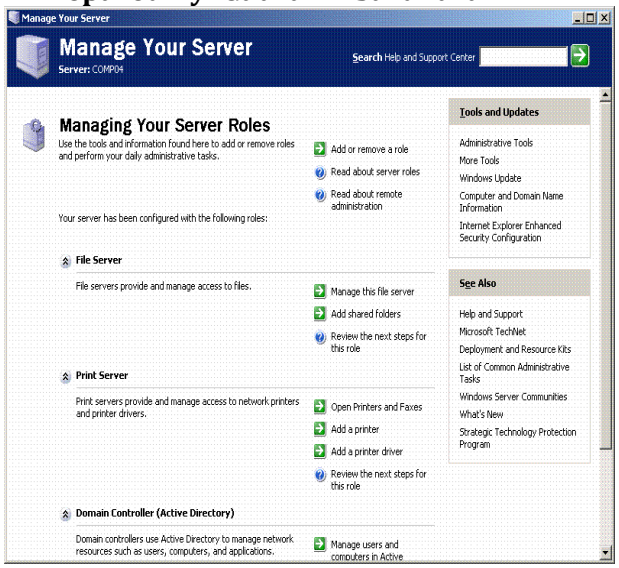
30. Hit the "Three Finger Salute" combination of Ctrl+Alt+Delete to login to the Administrator account.
You did remember your password, right?!?

31. Enter your password to login to the Administrator account.

Select **OK** to continue.

❖ **Windows 2003 Enterprise Server Configuration:**

- After installing and activating Windows, you can configure the server using the Manage Your Server page that launches automatically at logon.
- The page facilitates the installation of specific services, tools, and configurations based on server roles.
- Click Add or Remove a Role and the Configure Your Server Wizard appears.
- If you select Typical Configuration For A First Server, the Configure Your Server Wizard promotes the server to a domain controller in a new domain, installs Active Directory services, and, if needed, Domain Name Service (DNS), Dynamic Host Configuration Protocol (DHCP), and Routing And Remote Access (RRAS) service.
- If you select Custom Configuration, the Configure Your Server Wizard can configure the following roles:



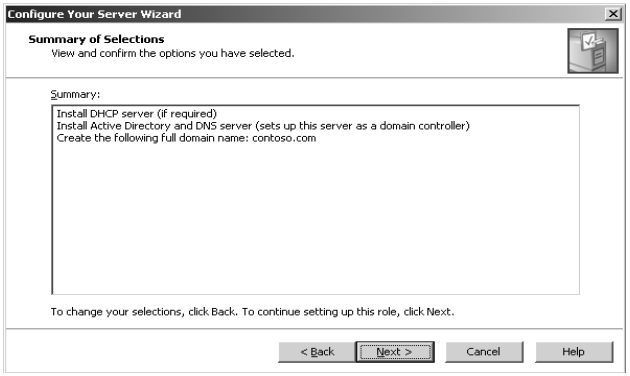
- **Following Step required Configuring Windows 2003 :**

Example: Configure the server as the first domain controller in an Active Directory domain called contoso.com**.**

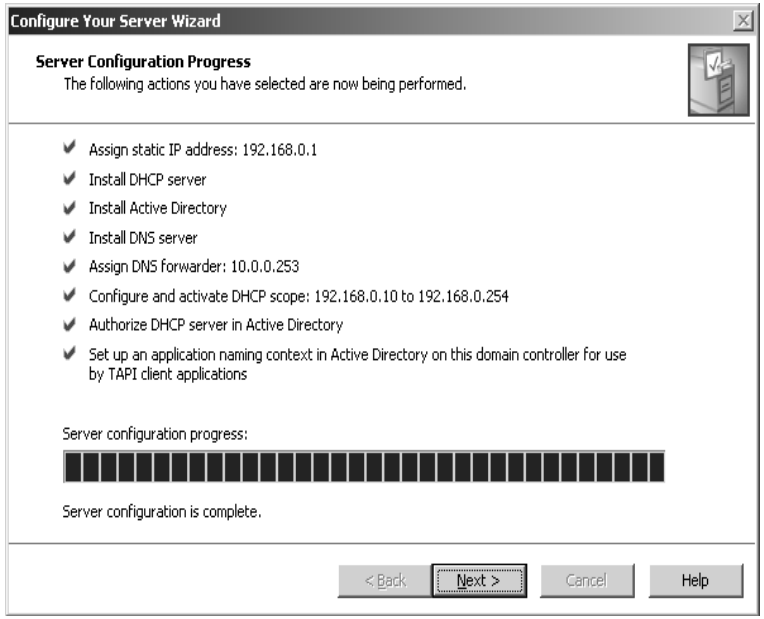
- 1. If it is not already open, open the Manage Your Server page from the Administrative Tools program group.
- 2. Click Add Or Remove A Role. The Configure Your Server Wizard appears.
- 3. Click next and the Configure Your Server Wizard detects network settings.
- 4. Click Typical Configuration For A First Server, and then click Next.
- 5. In Active Directory Domain Name, type contoso.com.
- 6. Verify that NetBIOS Domain Name reads CONTOSO and click Next.
- 7. Verify that the Summary of Selections matches that shown in Figure and click Next.

The Configure Your Server Wizard reminds you that the system will restart and asks you to close any open programs.

- 8. Click Yes.

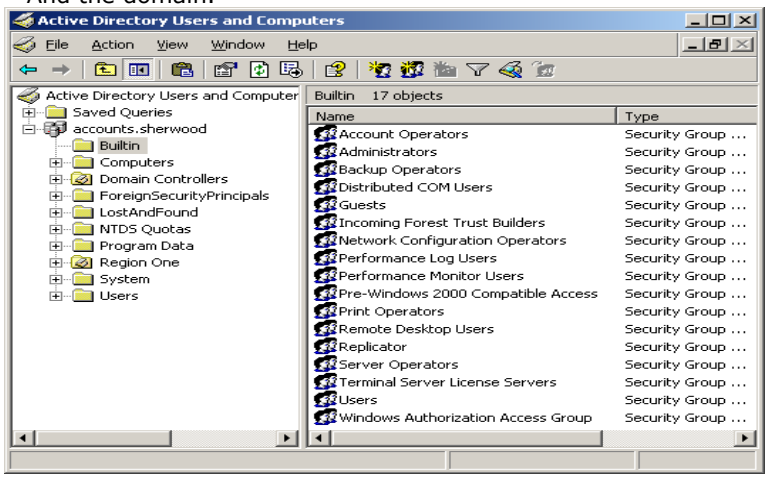


- 9. After the system has restarted, log on as Administrator.
- 10. The Configure Your Server Wizard will summarize its final steps, as shown in Figure.



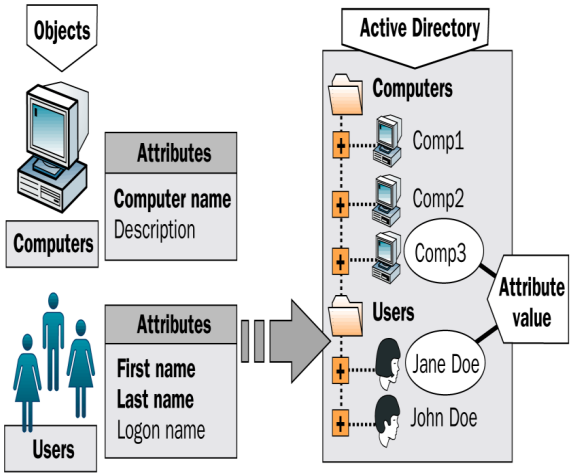
❖ Active Directory :

- Microsoft Windows networks support two directory service models:
- The workgroup
- And the domain.



- The domain model is by far the more common in organizations implementing Windows Server 2003.
- The domain model is characterized by a single directory of enterprise resources that is trusted by all secure systems that belong to the domain.
- Those systems can therefore use the security principals (user, group, and computer accounts) in the directory to secure their resources.
- **Active Directory thus acts as an identity store, providing a single trusted list of Who's Who in the domain.**
- Active Directory itself is more than just a database,
- It is a collection of supporting files including transaction logs and the system volume, or Sysvol, that contains logon scripts and group policy information.
- It is the services that support and use the database, including Lightweight Directory Access Protocol (LDAP), Kerberos security protocol, replication processes, and the File Replication Service (FRS).
- The database and its services are installed on one or more domain controllers.
- A domain controller is a server that has been promoted by running the Active Directory Installation Wizard by running DCPROMO from the command line or by running the Configure Your Server Wizard.
- Once a server has become a domain controller, it hosts a copy, or replica, of Active Directory and changes to the database on any domain controller are replicated to all domain controllers within the domain

❖ Active Directory Objects and Attributes :



❖ Active Directory Components:

Logical Structure

- Domains
- Organizational units
- Trees
- Forests

Physical Structure

- Sites
- Domain controllers

❖ Domains, Trees and Forests :

- **Domain** : Administrative unit of Active Directory
- **Tree :** A collection of one or more domains
- **Forest :** A collection of one or more trees

- Active Directory cannot exist without at least one domain.
- A domain is the core administrative unit of the Windows Server 2003 directory service.

- An enterprise may have more than one domain in its Active Directory.
- Multiple domain models create logical structures called trees when they share contiguous DNS names.
- For example **microsoft.com**, **uk.microsoft.com**, and us.microsoft.com share contiguous DNS namespace, and would therefore be referred to as a tree.
- If domains in an Active Directory do not share a common root domain, they create multiple trees.
- That leads you to the largest structure in an Active Directory: the forest.
- An Active Directory forest includes all domains within that Active Directory.
- A forest may contain multiple domains in multiple trees, or just one domain.
-
- When more than one domain exists, a component of Active Directory called the **Global Catalog** becomes important because it provides information about objects that are located in other domains in the forest and reduces LDAP queries to Active Directory.

Fig: Domain Tree

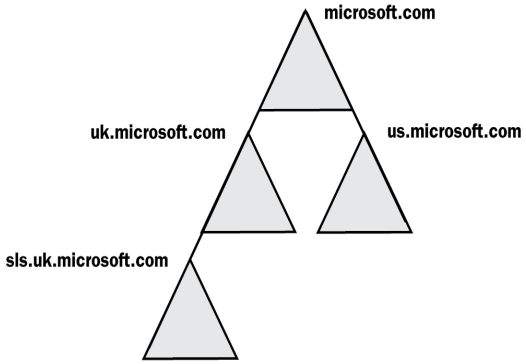


Fig: Domain Forest

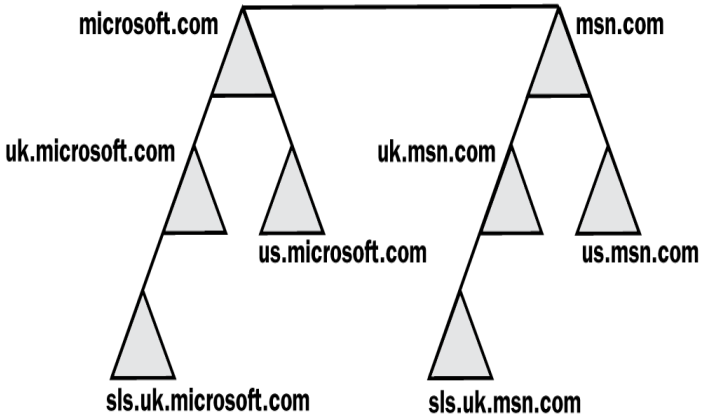
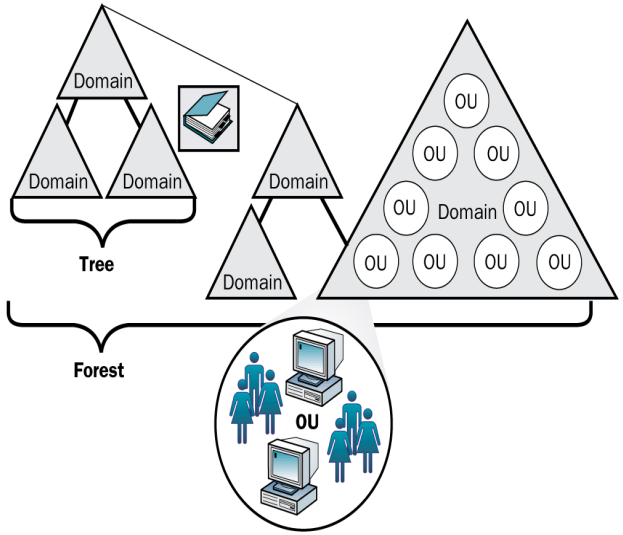


Fig: Logical Hierarchy



❖ Managing Active Directory Objects :

- When you first install Active Directory, a number of Containers are created to hold built-in users and groups, as well as computer accounts by default

- Organizational Units (OUs) allow the assignment of Group Policy and delegation of administrative control to junior administrators
- User accounts are best arranged into Organizational Units and have certain management functions that can be delegated at the OU level and inherited by lower levels.

- **User Account :**

  A user account consists of
  - Username and password
  - Group membership
  - Rights and permissions to access resources.

  Windows Server 2003 Computer configured as a Domain Controller with Active Directory
  - User accounts are managed by Active Directory Users and computers.

  Windows Server 2003 computer member Server (not a Domain Controller) and Windows XP workstations

  - User accounts are managed by Local Users and Groups

- **Computer Account :**

- Computer accounts are created and stored in the Active Directory like User and group accounts.

- Like users and group accounts, computer accounts have their own **specific attributes or properties** by which they can be searched and identified in the Active Directory.

- They can be members of security or distribution groups and inherit permissions from group objects.

- They inherit group policy settings from container objects such as domains, sites and Organizational Units (OUs).

- Computer accounts are used to identify computers in a domain with their security principles – SID

- A user with a valid user account and a password in Active Directory cannot log on to a domain, if the computer is not represented in that domain.

- Each Windows Server 2003 computer, Windows XP, Windows 2000 Server and Professional computer, Windows NT Server and workstation computer must have a computer account in an Active Directory - Domain Controller (DC) to participate in a domain.

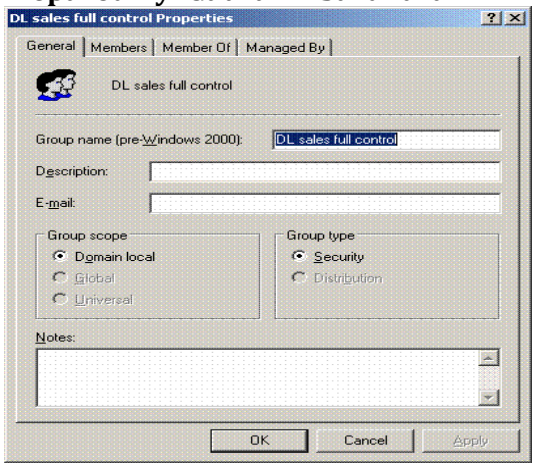
- Windows 95, 98, Me computers must install Active Directory Client software to participate in a domain
- Computer account password is generated automatically by the operating system and kept hidden

- **Group Account :**

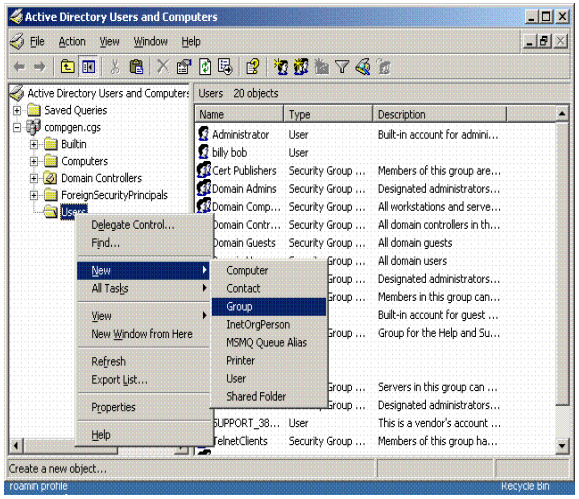
- Groups are a collection of user and computer accounts that you manage as a single unit.
- Groups simplify administration by enabling you to grant permissions to resources to the group rather that to each user individually.
- Are characterized by Scope and Type.
- Groups can be nested (groups can be members of other groups).
- In addition to user accounts, you can add other groups, contacts, and computers to groups
- Windows 2000 provides the ability to create groups:
- in a stand-alone computer security accounts database
- in Active Directory

  Types of Groups:

  - **Security groups :**
    - Windows 2000 itself only uses security groups, which you use to assign permissions to access resources and rights to perform tasks
    - Has all of the same functionality as distribution groups.
    -
  - **Distribution groups :**
    - Applications can use distribution groups as lists for non security-related functions
    - You cannot use distribution groups to assign rights and permissions



- **How to Create a Group :**
  - Active Directory Users and Computers Right click the OU or Container you want to create a group in and click New>>Group



❖ Monitoring Performance :

- One of the most important tasks that should be performed on the network is some form of statistical collecting.
- These statistics can range from the performance of servers, workstations, and other

- Devices on the network to the performance of individual components within a program or service itself.
- Generally Several types of performance monitoring tools:
  - Simple Network Management Protocol (SNMP),
  - Windows NT Performance Monitor, and
  - Windows 95's System Monitor.

● Simple Network Management Protocol (SNMP) :

- Many types of software and hardware on the market enable you to collect statistics on the network.
- One important protocol used within the TCP/IP protocol suite that assists in statistic collecting is the Simple Network Management Protocol (SNMP).
- SNMP is a protocol that is supported by most pieces of hardware and software that support the TCP/IP protocol stack.
- This protocol allows for the collection of statistics of various resources on the network.
  For this information to be collected about a resource, the resource must run an SNMP service, or have some other device run the SNMP service on its behalf.
- The SNMP service collects predefined information. This information is stored in a Management Information Base (MIB).
- An MIB is a database of information that can be read by management software designed to work with SNMP.

Management software issues one of the following three main commands:

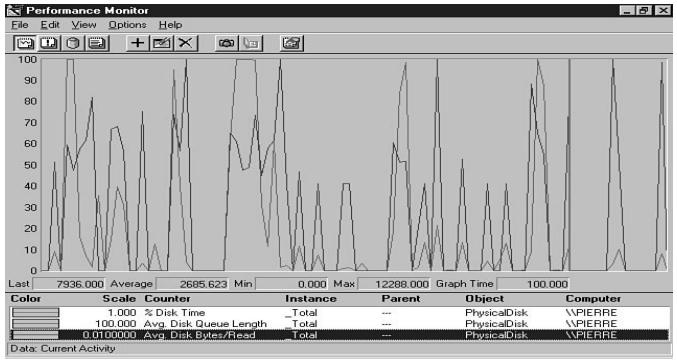
. The get command gathers information within an MIB.
. The get next command gets the next piece of information within the MIB.
. The set command places information within the MIB.

- These devices that have an SNMP service monitoring them can also be configured to issue traps, or system messages, when certain parameters are reached or exceeded.

● Windows NT Performance Monitor :

- Windows NT's Performance Monitor tool lets you monitor important system parameters for the computers on your network in real-time.
- Performance Monitor can keep an eye on a large number of system parameters, providing a graphical or tabular profile of system and network trends.
- Performance Monitor also can save performance data in a log for later reference.

- You can use Performance Monitor to track statistical Measurements (called counters) for any of several hardware or software components (called objects).
- An example of these counters for an object being displayed in a chart format can be seen in Figure.



Some Performance Monitor objects that relate to network behavior are as follows:

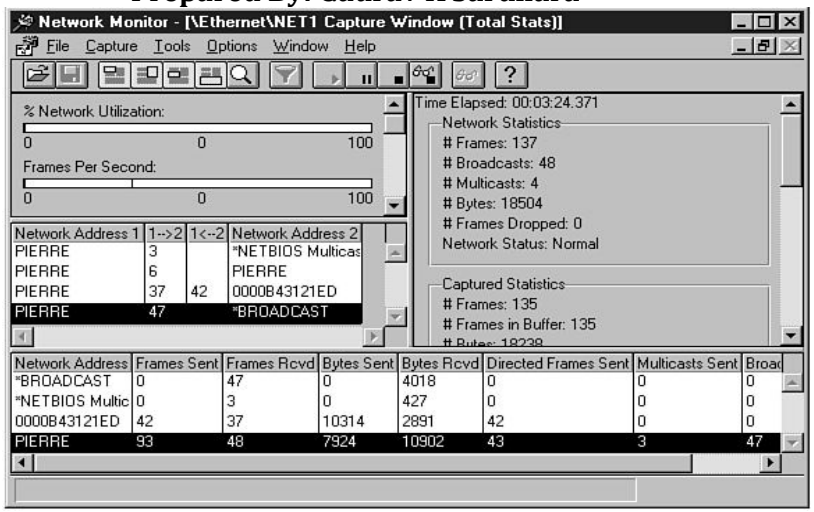
. Network segment
. Server
. Server work queues
. Workstation or other Redirectors
. Protocol-related objects, such as TCP, UDP IP, NetBEUI, NWLink, and NetBIOS
. Service-related objects, such as Browser and Gateway Services for NetWare.

Some Performance Monitor counters that relate to the performance of components or resources on a computer are as follows:

. Processor
. Memory
. Physical Disk

❖ Monitoring Network Traffic :

- Protocol analysis tools monitor network traffic by intercepting and decoding frames.
- Software-based tools, such as Windows NT Server's Network Monitor (see Figure), analyze frames coming and going, in real time, from the computer on which they run.



- Network Monitor records a number of statistics, including the percent of network utilization and the broadcasts per second.
- In addition, Network Monitor tabulates frame statistics (such as frames sent and received) for each network address.
- An enhanced version of Network Monitor, which is included with the Microsoft BackOffice System Management Server (SMS) package, monitors traffic on more than just the traffic between the local computer and other devices.
- It will also monitor traffic that is just between other devices, and also traffic on remote networks, provided a monitor agent is installed on the remote network segment.
- For large networks, or for networks with complex traffic patterns, you might want to use a hardware-based protocol-analysis tool.
- A hardware-based protocol analyzer is a portable device that can be as small as a palmtop PC or as large as a suitcase.

- The advantage of a hardware-based protocol analyzer is that you can carry it to strategic places around the network (such as a network node or a busy cabling intersection) and monitor the traffic at that point.
- A hardware-based protocol analyzer is often a good investment for a large network because it concentrates a considerable amount of monitoring and troubleshooting power into a single, portable unit.

- For a smaller network, however, a hardware-based analyzer might not be worth the initial five-figure expense because less expensive software-based products perform many of the same functions.

❖ Monitoring Network Traffic :

- Some operating systems, such as Windows NT, have the capability to keep a running log of system events.
- That log serves as a record of previous errors, warnings, and other messages from the system.
- Studying the event log can help you find recurring errors and discover when a problem first appeared.
- The event log should also be scanned on a regular basis to look for any indications of potential problems.
- Windows NT's Event Viewer application provides you with access to the event log.

- **You can use Event Viewer to monitor the following types of events:**
- **System events:**. Warnings, error messages, and other notices describing significant system events. Examples of system log entries include browser elections, service failures, and network Connection failures.
- **Security events:** Events tracked through Windows NT's auditing features.
- **Application events**. you can check the application log for an application-related error or warning message, provided the application is programmed to write to the event log. Some NT services such as the JET database engine used by WINS record their information in the application events log rather than the system log.

- **The Windows NT Event Viewer utility contains the following five types of events:**

1. **Information.** These events simply state that something of importance has been done, such as the loading of a protocol. These events are recorded for a matter of information only.

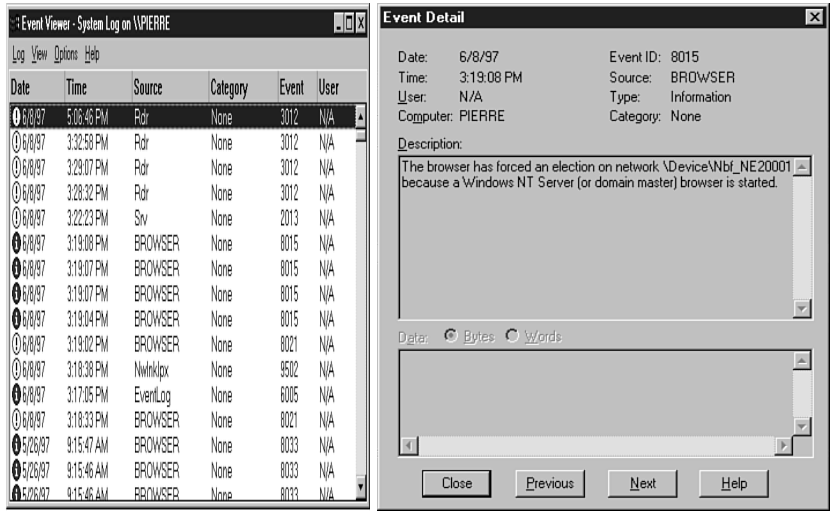
2. **Warning.** These events serve as a warning that some event that may be important has occurred. Often when services are stopped, a warning event is generated.

3. **Stop.** These events occur when something of significance, such as a detrimental event, has occurred. Often when services or hardware fail, a Stop event is generated.

4. **Success**. This event is generated within the auditing log.
   Success events are generated when an object that was audited as

Successful has occurred. You might, for example, audit the successful Logon of users.

5. **Failure.** This event is generated within the auditing log.
   Failure events are generated when an object that was audited as a Failure has occurred, such as the failure of users to log on.



❖ Microsoft Management Console(MMC) :

- MMC is the primary tool used to administer Windows Server 2003.

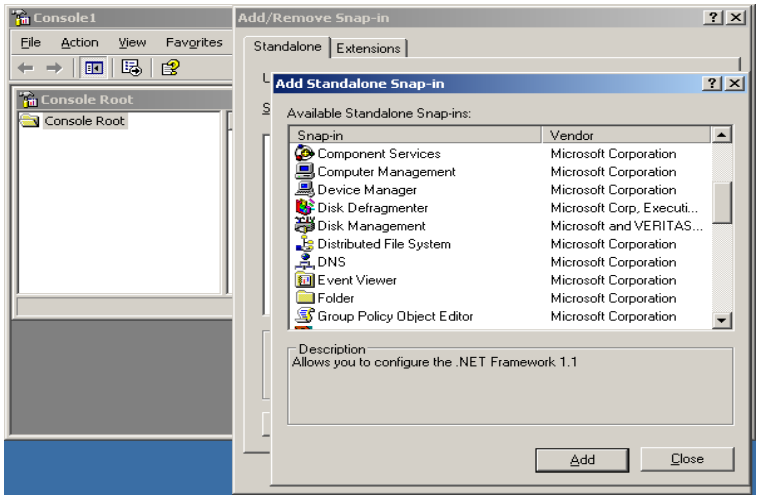
- In a Windows Server 2003 environment, administrator will normally be responsible for more than one server.

- A large number of pre-configured MMC are available in the Administrative Tools menu.
- A useful tool for administrators to manage Windows computers anywhere on the network
  (Remote server and clients) is **Microsoft Management Console (MMC)**

- MMC provides a customizable management framework for hosting multiple management tools (snap-ins)
- MMC with one or more snap-ins is called **console**

- Can add and remove management tools as necessary and save as a custom MMC console file with .msc extension
- Most of the shortcuts in the Administrative Tools program group are preconfigured MMC consoles.
- The executable file for MMC is **Mmc.exe.**
- Run this file from the Run dialog box or command prompt.
- Empty console appears, select Add/Remove snap-in from the **File** menu.
- **Select** and **add** as many stand-alone snap-ins to a console and save it as a custom console with **.msc file extension.**
- Can access a remote computer through selecting Connect to Another Computer from Action menu in the MMC snap-in.
- Also, by using Add/Remove snap in from **File** menu, selecting what computer you want to manage from the list of snap-ins and then clicking **Add** button.

Microsoft Management Console



UNIT- 5

Chapter 10: Network Security

❖ Fundamental of Network Security :

When you are planning, designing, or implementing a network or are assigned to operate and manage one, it is useful to ask yourself the following questions:

1. What are you trying to protect or maintain?
2. What are your business objectives?
3. What do you need to accomplish these objectives?
4. What technologies or solutions are required to support these objectives?
5. Are your objectives compatible with your security infrastructure, operations, and tools?
6. What risks are associated with inadequate security?
7. What are the implications of not implementing security?
8. Will you introduce new risks not covered by your current security solutions or policy?
9. How do you reduce that risk?
10. What is your tolerance for risk?

You can use these questions to pose and answer some of the basic questions that underlie fundamental requirements for establishing a secure network. Network security technologies reduce risk and provide a foundation for expanding businesses with intranet, extranet, and electronic commerce applications.

Solutions also protect sensitive data and corporate resources from intrusion and corruption.

Advanced technologies now offer opportunities for small and medium-sized businesses (SMB), as well as enterprise and large-scale networks to grow and compete; they also highlight a need to protect computer systems against a wide range of security threats.

The challenge of keeping your network infrastructure secure has never been greater or more crucial to your business. Despite considerable investments in information security, organizations continue to be afflicted by cyber incidents. At the same time, management aims for greater

results with fewer resources. Hence, improving security effectiveness remains vital, if not essential, while enhancement of both effectiveness and flexibility has also become a primary objective.

Without proper safeguards, every part of a network is vulnerable to a security breach or unauthorized activity from intruders, competitors, or even employees. Many of the organizations that manage their own internal network security and use the Internet for more than just sending/receiving e-mails experience a network attack—and more than half of these companies do not even know they were attacked. Smaller companies are often complacent, having gained a false sense of security. They usually react to the last virus or the most recent defacing of their website. But they are trapped in a situation where they do not have the necessary time and resources to spend on security.

❖ Security Paradigm :

As the size of networks continues to grow and attacks to those networks become increasingly sophisticated, the way we think about security changes. Here are some of the major factors that are changing the security paradigm:

- **Security is no longer about "products":** Security solutions must be chosen with business objectives in mind and integrated with operational procedures and tools.
- **Scalability demands are increasing:** With the increasing number of vulnerabilities and security threats, solutions must scale to thousands of hosts in large enterprises.
- **Legacy endpoint security Total Cost of Ownership (TCO) is a challenge:** Reactive products force deployment and renewal of multiple agents and management paradigms.
- **Day zero damage:** Rapidly propagating attacks (Slammer, Nimda, MyDoom) happen too fast for reactive products to control. Therefore, an automated, proactive security system is needed to combat the dynamic array of modern-day viruses and worms.

❖ Principles of Security—The CIA Model :

A simple but widely applicable security model is the confidentiality, integrity, and availability (CIA) triad. These three key principles should guide all secure systems. CIA also provides a measurement tool for security implementations. These principles are applicable across the entire spectrum of security analysis—from access, to a user's Internet history, to the security of encrypted data across the Internet. A breach of any of these three principles can have serious consequences for all parties concerned.

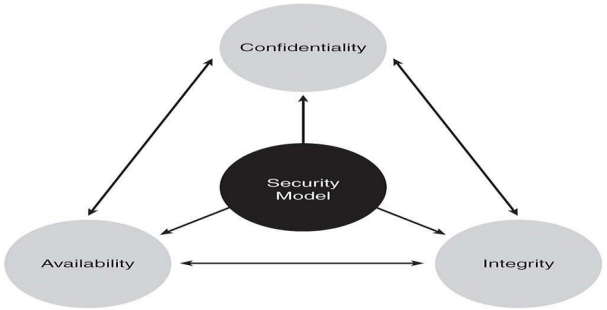


Fig: CIA Triad

Confidentiality:

Confidentiality prevents unauthorized disclosure of sensitive information. It is the capability to ensure that the necessary level of secrecy is enforced and that information is concealed from unauthorized users. When it comes to security, confidentiality is perhaps the most obvious aspect of the CIA triad, and it is the aspect of security most often attacked. Cryptography and encryption methods are examples of attempts to ensure the confidentiality of data transferred from one computer to another. For example, when performing an online banking transaction, the user wants to protect the privacy of the account details, such as passwords and card numbers. Cryptography provides a secure transmission protecting the sensitive data traversing across the shared medium.

Integrity:

Integrity prevents unauthorized modification of data, systems, and information, thereby providing assurance of the accuracy of information and systems. If your data has integrity, you can be sure that it is an accurate and unchanged representation of the original secure information. A common type of a security attack is man-in-the-middle. In this type of attack, an intruder intercepts data in transfer and makes changes to it.

Availability:

Availability is the prevention of loss of access to resources and information to ensure that information is available for use when it is needed. It is imperative to make sure that information requested is readily accessible to the authorized users at all times. Denial of service (DoS) is one of several types of security attacks that attempts to deny access to the appropriate user, often for the sake of disruption of service.

❖ Policies, Standards, Procedures, Baselines, Guidelines:

A *security model* is a multilayered framework made of many integrated entities and logical and physical protection mechanisms, all working together to provide a secure system that complies with industry best practices and regulations

Security Policy:

A *security policy* is a set of rules, practices, and procedures dictating how sensitive information is managed, protected, and distributed. In the network security realm, policies are usually point specific, which means they cover a single area. A security policy is a document that expresses exactly what the security level should be by setting the goals of what the security mechanisms are to accomplish. Security policy is written by higher management and is intended to describe the "whats" of information security.

Trust is one of the main themes in many policies. Some companies do not have policies because they trust in their people and trust that everyone will do the right thing. But, that is not always the case, as we all know. Therefore, most organizations need policies to ensure that everyone complies with the same set of rules.

A policy should define the level of control users must observe and balance that with productivity goals. An overly strict policy will be hard to implement because compliance will be minimal or ignored. On the contrary, a loosely defined policy can be evaded and does not ensure accountability and responsibility. A good policy has to have the right balance.

Standards:

Standards are industry-recognized best practices, frameworks, and agreed principles of concepts and designs, which are designed to implement, achieve, and maintain the required levels of processes and procedures.
Like security policies, standards are strategic in nature in that they define systems parameters and processes.
Standards vary by industry. There are two notable standards in security information management—ISO 17799 and COBIT.

Procedures:

Procedures are low-level documents providing systematic instructions on how the security policy and the standards are to be implemented in a system. Procedures are detailed in nature to provide maximum information to users so that they can successfully implement and enforce the security policy and apply the standards and guidelines of a security program.
Employees usually refer to procedures more often than other policies and standards because procedures provide the actual details of the implementation phase of a security program.

Baselines:

A *baseline* is the minimum level of security requirement in a system. Baselines provide users the means to achieve the absolute minimum security required that is consistent across all the systems in the organization. For example, a company might have a baseline for Windows 2000 servers to have Service Pack 4 installed on each server in the production environment. The procedure document would supplement the baseline by spelling out step-by-step instructions on where to download Service Pack 4 and how to install it to comply with this security level.

Guidelines:

Guidelines are recommended actions and operational guides for users. Similar to procedures, guidelines are tactical in nature. The major difference between standards and guidelines is that guidelines can beused as reference, whereas standards are mandatory actions in most case

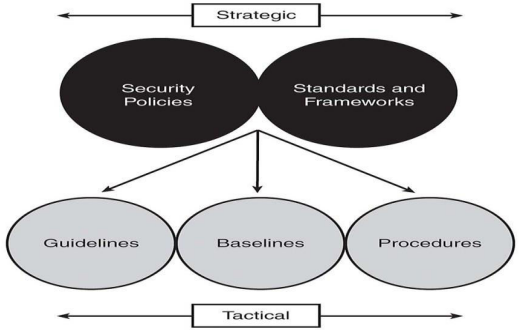


Fig.: Relationships among Security Policies, Standards, Procedures, Baselines, and Guidelines

❖ **Security Models:**

An important element in the design and analysis of secure systems is the security model, because it integrates the security policy that should be enforced in the system. A *security model* is a symbolic portrayal of a security policy. It maps the requirements of the policy makers into a set of rules and regulations that are to be followed by a computer system or a network system. A *security policy* is a set of abstract goals and high-level requirements, and the security model is the do's and don'ts to make this happen.

You should know about several important security models even though describing them in detail is beyond the scope of this book:

- **The Bell-LaPadula Model (BLM)**, also called the multilevel model, was introduced mainly to enforce access control in government and

military applications. BLM protects the confidentiality of the information within a system.

- **The Biba model** is a modification of the Bell-LaPadula model that mainly emphasizes the integrity of the information within a system.

- **The Clark-Wilson model** prevents authorized users from making unauthorized modification to the data. This model introduces a system of triples: a subject, a program, and an object.
- The Access Control Matrix is a general model of access control that is based on the concept of subjects and objects.
- The Information Flow model restricts information in its flow so that it moves only to and from approved security levels.

- **The Chinese Wall model** combines commercial discretion with legally enforceable mandatory controls. It is required in the operation of many financial services organizations.

- **The Lattice model** deals with military information. Lattice-based access control models were developed in the early 1970s to deal with the confidentiality of military information. In the late 1970s and early 1980s, researchers applied these models to certain integrity concerns. Later, application of the models to the Chinese Wall policy, a confidentiality policy unique to the commercial sector, was developed. A balanced perspective on lattice-based access control models is provided.

❖ **ENCRYPTION**:-

→ In cryptography, encryption is the process of encoding messages or information in such a way that only authorized parties can read it. Encryption does not of itself prevent interception, but denies the message content to the interceptor.
→ In an encryption scheme, the intended communication information or message, referred to as plaintext, is encrypted using an encryption algorithm, generating cipher text that can only be read if decrypted. For technical reasons, an encryption scheme usually uses a pseudo-random encryption key generated by an algorithm.
→ It is in principle possible to decrypt the message without possessing the key, but, for a well-designed encryption scheme, large computational resources and skill are required. An authorized recipient can easily decrypt the message with the key provided by the originator to recipients, but not to unauthorized interceptors.

❖ **CRYPTOGRAPHY:**-

→ Cryptography prior to the modern age was effectively synonymous with encryption, the conversion of information from a readable state to apparent nonsense.
→ The originator of an encrypted message (Alice) shared the decoding technique needed to recover the original information only with intended recipients (Bob), thereby precluding unwanted persons (Eve) from doing the same. Since World War I and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.
→ Modern cryptography is heavily based on mathematical theory and computer science practice; cryptographic algorithms are designed around computational hardness assumptions, making such algorithms hard to break in practice by any adversary.
→ It is theoretically possible to break such a system, but it is infeasible to do so by any known practical means. These schemes are therefore termed computationally secure; theoretical advances, e.g., improvements in integer factorization algorithms, and faster computing technology require these solutions to be continually adapted.
→ There exist information-theoretically secure schemes that provably cannot be broken even with unlimited computing power—an example is the one-time pad—but these schemes are more difficult to implement than the best theoretically breakable but computationally secure mechanisms.
→ The growth of cryptographic technology has raised a number of legal issues in the information age. Cryptography's potential for use as a tool for espionage and sedition has led many governments to classify it as a weapon and to limit or even prohibit its use and export.
→ In some jurisdictions where the use of cryptography is legal, laws permit investigators to compel the disclosure of encryption keys for documents relevant to an investigation. Cryptography also plays a major role in digital rights management and piracy of digital media.

❖ **AUTHENTICATION**:-

→ In the context of computer systems, authentication is a process that ensures and confirms a user's identity. Authentication is one of the five pillars of information assurance (IA). The other four are integrity, availability, confidentiality and no repudiation
→ Authentication begins when a user tries to access information. First, the user must prove his access rights and identity. When logging into a computer, users commonly enter usernames and passwords for authentication purposes. This login combination, which must be assigned to each user, authenticates access. However, this type of authentication can be circumvented by hackers.
→ A better form of authentication, biometrics, depends on the user's presence and biological makeup (i.e., retina or fingerprints). This

technology makes it more difficult for hackers to break into computer systems.
The Public Key Infrastructure (PKI) authentication method uses digital certificates to prove a user's identity. There are other authentication tools, too, such as key cards and USB tokens. One of the greatest authentication threats occurs with email, where authenticity is often difficult to verify. For example, unsecured emails often appear legitimate.

❖ Perimeter Security :

Opinions on perimeter security have changed a great deal over the past few years. Part of that change is that the very nature of perimeter security is becoming increasingly uncertain, and everyone has a different view of just what it is. The limits of the perimeter itself are becoming broad and extensive, with no geographic boundaries, and remote access is becoming part of the integral network.

Is Perimeter Security Disappearing?

In essence, the perimeter has been transformed and extended to the various levels within the network. In other words, networks today do not have a single point of entrance; they are multi-entry open environments where controlled access is required from anywhere within the network. This transformation leads us to start thinking in terms of multiperimeter networks.

The Difficulty of Defining Perimeter

Traditional networks are growing with the merging of remote network access. Wireless networks, laptops, mobile phones, PDAs, and numerous other wireless gadgets need to connect from outside the enterprise into the corporate network. To fulfill these needs, the concept of inside versus outside becomes rather complicated. For example, when you connect to the corporate network using a virtual private network (VPN), you are no longer on the outside the network. You are now on the inside of the network, and so is everything that is running on your computer.

Globally networked businesses rely on their networks to communicate with employees, customers, partners, and suppliers. Although immediate access to information and communication is an advantage, it raises concerns about security and protecting access to critical network resources.

Network administrators need to know who is accessing which resources and establish clear perimeters to control the access. An effective security

policy balances accessibility with protection. Security policies are enforced at network perimeters. Often people think of a perimeter as the boundary between an internal network and the public Internet, but a perimeter can be established anywhere within a private network, or between your network and a partner's network.

❖ Security in Layers:

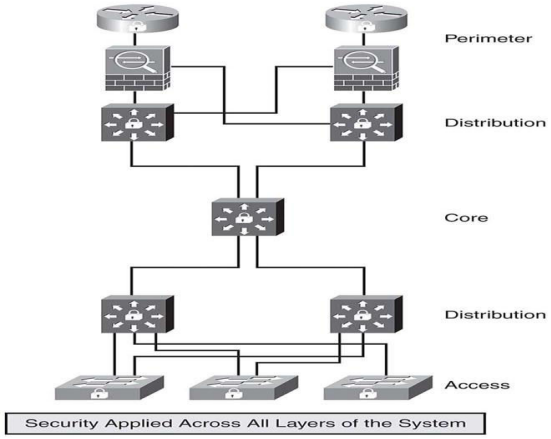
Security in layers is the preferred and most scalable approach to safeguard a network. One single mechanism cannot be relied on for the security of a system. To protect your infrastructure, you must apply security in layers. This layered approach is also called *defense in depth*. The idea is that you create multiple systems so that a failure in one does not leave you vulnerable, but is caught in the next layer. Additionally, in a layered approach, the vulnerability can be limited and contained to the affected layer because of the applied security at varying levels.

Multilayer Perimeter Solution

As stated previously, today's solutions are shifting toward the approach of placing safeguard mechanisms at various layers of the network, not just at the boundary or edge devices. Today, it is recommended to deploy Intrusion Prevention System (IPS) devices on both the inside and outside boundaries of private networks. Firewalls, on the other hand, are placed between various business segments or departments within the same organization, dividing the network into logical groupings and applying perimeter defense at each segment or department. In this multiperimeter model, each segment can have different layers of defense within it.

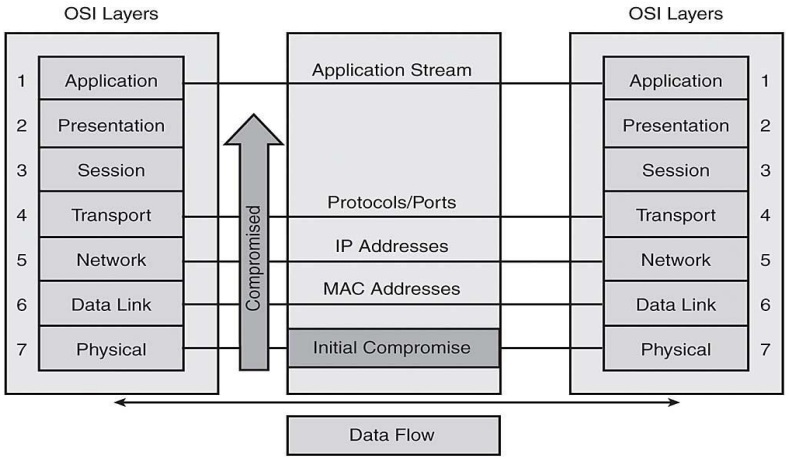
Effective perimeter security has become increasingly important over recent years. Perimeter security cannot be trusted to only the traditional defense mechanisms of firewalls and IDS. Web applications, wireless access, network interconnectivities, and VPNs have made the perimeter a much more complicated concept than it was a couple of years ago.

A layered approach requires implementing security solutions at different spectrums of the network. Another similar concept is *islands of security*. To implement islands of security, do not restrict your thinking to perimeter security. Do not depend on just one method for your security. You should, instead, have layers of protection—perimeter, distribution, core, and access layer. Figure illustrates a basic multilayered security mechanism, which is designed to protect the data flow in the system.



The Domino Effect

The OSI reference model was built to enable different layers to work independently of each other. The layered approach was developed to accommodate changes in the evolving technology. Each OSI layer is responsible for a specific function within the networking stack, with information flowing up and down to the next subsequent layer as data is processed. Unfortunately, this means that if one layer is hacked, communications are compromised without the other layers being aware of the problem. For example, as shown in Figure -, if the physical layer (Layer 1) was compromised, it could cause all other layers to be compromised in succession. Security is only as strong as the weakest link. When it comes to networking, any layer can be the weakest link.



❖ **Security in Wheel:**

Network security is a continuous process built around the corporate security policy. The security wheel depicted in Figure shows a recursive, ongoing process of striving toward perfection—to achieve a secured network infrastructure. The paradigm incorporates the following five steps:

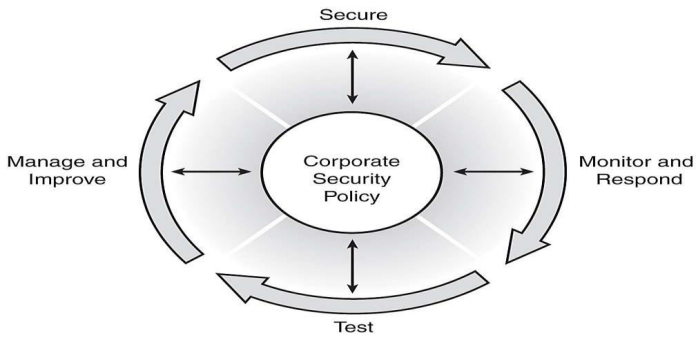


Fig: Security Wheel

Step 1: Develop a security policy

A strong security policy should be clearly defined, implemented, and documented, yet simple enough that users can easily conduct business within its parameters.

Step 2: Make the network secure

Secure the network by implementing security solutions (implement authentication, encryption, firewalls, intrusion prevention, and other techniques) to stop or prevent unauthorized access or activities and to protect information and information systems.

Step 3: Monitor and respond.

This phase detects violations to the security policy. It involves system auditing and real-time intrusion detection and prevention solutions. This also validates the security implementation in Step 2.

Step 4: Test.

This step validates the effectiveness of the security policy through system auditing and vulnerability scanning and tests existing security safeguards.

Step 5: Manage and improve.

Use information from the monitor and test phases to make improvements to the security implementation. Adjust the corporate security policy as security vulnerabilities and risks are identified. Manage and improve corporate security policy.

Lessons learned from Steps 2 through 5 should always be reflected back to the corporate security policy in Step 1, so that the high-level security expectations are being met. This should be an ongoing process, a continuous life cycle.

Chapter 11: Internet Connection and Sharing

Basic of Internet:

By the turn of the century, information, including access to the Internet, will be the basis for personal, economic, and political advancement. The popular name for the Internet is the information superhighway.

Whether you want to find the latest financial news, browse through library catalogs, exchange information with colleagues, or join in a lively political debate, the Internet is the tool that will take you beyond telephones, faxes, and isolated computers to a burgeoning networked information frontier.
The Internet supplements the traditional tools you use to gather

information, Data Graphics, News and correspond with other people. Used skillfully, the Internet shrinks the world and brings information, expertise, and knowledge on nearly every subject imaginable straight to your computer.

The Internet links are computer networks all over the world so that users can share resources and communicate with each other. Some computers have direct access to all the facilities on the Internet such as the universities. And other computers, eg. Privately-owned ones have indirect links through a commercial service provider, who offers some or all of the Internet facilities.

In order to be connected to Internet, you must go through service suppliers. Many options are offered with monthly rates. Depending on the option chosen, access time may vary.

The Internet is what we call a Meta network, that is, a network of networks that spans the globe. It's impossible to give an exact count of the number of networks or users that comprise the Internet, but it is easily in the thousands and millions respectively. The Internet employs a set of standardized protocols which allow for the sharing of resources among different kinds of computers that communicate with each other on the network.

These standards, sometimes referred to as the Internet Protocol Suite, are the rules that developers adhere to when creating new functions for the Internet.
The Internet is also what we call a distributed system; there are no central archives. Technically, no one runs the Internet. Rather, the Internet is made up of thousands of smaller networks. The Internet thrives and develops as its many users find new ways to create, display and retrieve the information that constitutes the Internet.

How internat is connect with computer:

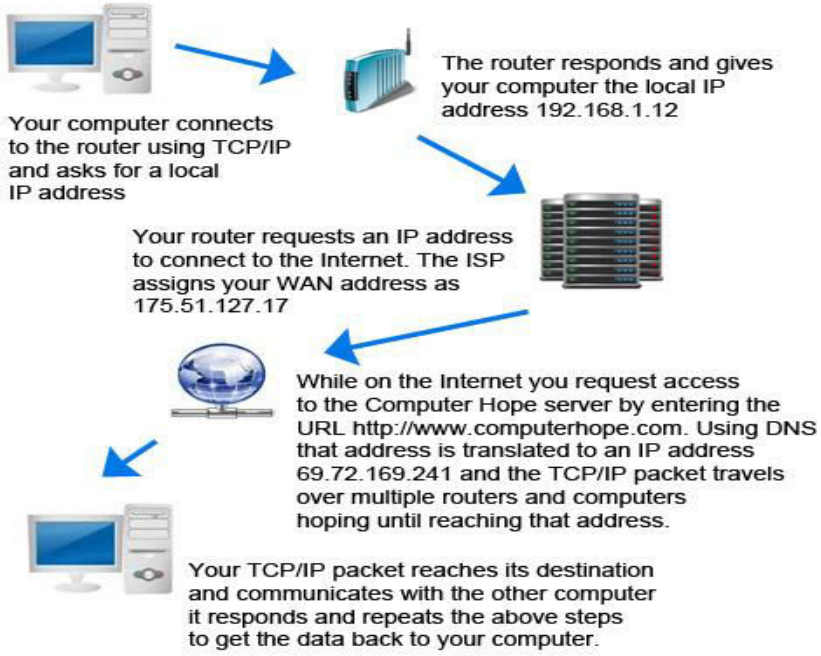
Using a network connection, including connecting to the Internet, computers connect to each other to transmit data between them and communicate with each other using the TCP/IP (Transmission Control Protocol / Internet Protocol). Think of TCP/IP as a book of rules, a step-by-step guide that each computer uses to know how to talk to another computer. This book of rules dictates what each computer must do to transmit data, when to transmit data, how to transmit that data. It also states how to receive data in the same manner. If the rules are not followed, the computer will not be able to connect to another computer, nor send and receive data between other computers.

Internet service providers (ISP), the companies that provide Internet service and connectivity also follow these rules. The ISP provides a bridge between

your computer and all the other computers in the world, which are all a part of the Internet. The ISP uses the TCP/IP protocols to make computer-to-computer connections possible and transmit data between them. When successfully connected to an ISP you will be assigned an IP address, which is a unique address given to your computer or network and allows it to be found while on the Internet.

If you have a home computer network, the computers are also using TCP/IP to connect to each other. This protocol allows each computer to "see" the other computers on the network and share files between them and is what makes it possible for a printer to be shared on a network. When computers connect to each other on the same network, it is called a local area network, or LAN. When multiple networks are connected to each other, it is called a wide area network, or WAN. With this type of network, your home will have a network router that connects to your ISP. The router is given the IP address for your connection to the Internet and then assigns local IP addresses to each device in your network. These local addresses are often 192.168.1.2-255. When accessing a local computer in your own network, your router sends your TCP/IP packets between the local IP addresses. However, when you want to connect to the Internet your router communicates to the Internet with the IP address assigned to it from the ISP. This is why when on the Internet your IP address is not a 192.168.x.x address.

When requesting information from a web page, such as Computer Hope you enter a URL that is easy to understand and remember. In order for your computer to access the computer containing the pages that URL must be converted into an IP address, this is done with DNS. Once DNS has converted the URL into an IP address the routers on the Internet will know how to route your TCP/IP packet. Below is a graphic illustration of everything explained above to help better illustrate the process of your computer communicating with another computer on the Internet?



Technology related internet:

.

- **Dial up Technology:**

  Definition: Dial up networking technology provides PCs and other network devices access to a LAN or WAN via standard telephone lines. Dial up Internet service providers offer subscription plans for home computer users.

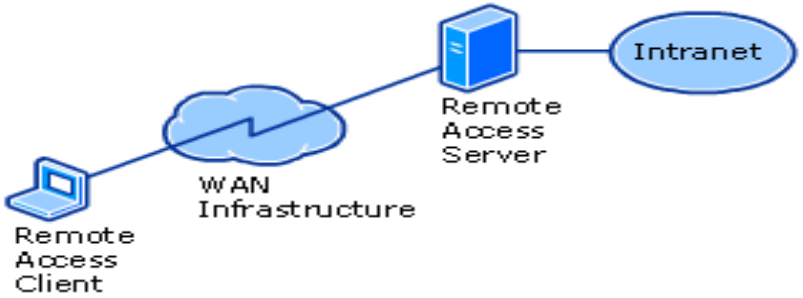
Dial-up Internet access is a form of Internet access that uses the facilities of the public switched telephone network (PSTN) to establish a dialled connection to an Internet service provider (ISP) via telephone lines. The user's computer or router uses an attached modem to encode and decode Internet Protocol packets and control information into and from analogue audio frequency signals, respectively.

The term was coined during the early days of computer telecommunications when modems were needed to connect terminals or computers running terminal emulator software to mainframes, minicomputers, online services and bulletin board systems via a telephone line.

Dial-up connections to the Internet require no infrastructure other than the telephone network. Where telephone access is widely available, dial-up remains useful to travelers. Dial-up is often the only choice available for rural or remote areas, where broadband installations are not prevalent due to low population density, and high infrastructure cost. Dial-up access may also be an alternative for users on limited budgets, as it is offered free by some ISPs, though broadband is increasingly available at lower prices in many countries due to market competition.

Dial-up requires time to establish a telephone connection (up to several seconds, depending on the location) and perform handshaking for protocol synchronization before data transfers can take place. In locales with telephone connection charges, each connection incurs an incremental cost. If calls are time-metered, the duration of the connection incurs costs. Dial-up access is a transient connection, because either the user, ISP or phone company terminates the connection. Internet service providers will often set a limit on connection durations to allow sharing of resources, and will disconnect the user—requiring reconnection and the costs and delays associated with it. Technically-inclined users often find a way to disable the auto-disconnect program such that they can remain connected for days.

A 2008 Pew Internet and American Life Project study states that only 10 percent of US adults still used dial-up Internet access. Reasons for retaining dial-up access include lack of infrastructure and high broadband prices. This has allowed Dial-up providers such as NetZero to continue spending marketing dollars to obtain customers and commit to having U.S. based customer support.



- **ISDN Network Technology (Integrated Services Digital Network)**

ISDN (Integrated Services Digital Network) is a set of CCITT/ITU standards for digital transmission over ordinary telephone copper wire as well as over other media. Home and business users who install an ISDN adapter (in place of a telephone modem) receive Web pages at up to 128 Kbps compared with the maximum 56 Kbps rate of a modem connection. ISDN requires adapters at both ends of the transmission so your access provider also needs an ISDN adapter. ISDN is generally available from your phone company in most

urban areas in the United States and Europe. In many areas where DSL and cable modem service are now offered, ISDN is no longer as popular an option as it was formerly.

There are two levels of service: the Basic Rate Interface (BRI), intended for the home and small enterprise, and the Primary Rate Interface (PRI), for larger users. Both rates include a number of B-channels and a D-channels. Each B-channel carries data, voice, and other services. Each D-channel carries control and signaling information.

The Basic Rate Interface consists of two 64 Kbps B-channels and one 16 Kbps D- channel. Thus, a Basic Rate user can have up to 128 Kbps service. The Primary Rate consists of 23 B-channels and one 64 Kpbs D-channel in the United States or 30 B-channels and 1 D-channel in Europe.

ISDN in concept is the integration of both analog or voice data together with digital data over the same network. Although the ISDN you can install is integrating these on a medium designed for analog transmission, broadband ISDN (BISDN) is intended to extend the integration of both services throughout the rest of the end-to-end path using fiber optic and radio media. Broadband ISDN encompasses frame relay service for high-speed data that can be sent in large bursts, the Fiber Distributed-Data Interface (FDDI), and the Synchronous Optical Network (SONET). BISDN is intended to support transmission from 2 Mbps up to much higher, but as yet unspecified, rates.

- **Leased Line Technology:**

A leased line is a service contract between a provider and a customer, whereby the provider agrees to deliver a symmetric telecommunications line connecting two or more locations in exchange for a monthly rent (hence the term lease). It is sometimes known as a 'Private Circuit' or 'Data Line' in the UK or as CDN (Circuit Direct Number) in Italy. Unlike traditional PSTN lines it does not have a telephone number, each side of the line being permanently connected to the other. Leased lines can be used for telephone, data or Internet services. Some are ring down services, and some connect two PBXes.

Typically, leased lines are used by businesses to connect geographically distant offices. Unlike dial-up connections, a leased line is always active. The fee for the connection is a fixed monthly rate. The primary factors affecting the monthly fee are distance between end points and the speed of the circuit. Because the connection doesn't carry anybody else's communications, the carrier can assure a given level of quality.

An internet leased line is a premium internet connectivity product, delivered over fiber normally, which is dedicated and provides uncondensed, symmetrical speeds, Full Duplex. It is also known as an Ethernet leased line, DIA line, data circuit or private circuit.

For example, a T-1 channel can be leased, and provides a maximum transmission speed of 1.544 Mbit/s. The user can divide the connection into different lines for multiplexing data and voice communication, or use the channel for one high speed data circuit. Increasingly, leased lines are being used by companies, and even individuals, for Internet access because they afford faster data transfer rates and are cost-effective for heavy users of the Internet.

- **VPN:**

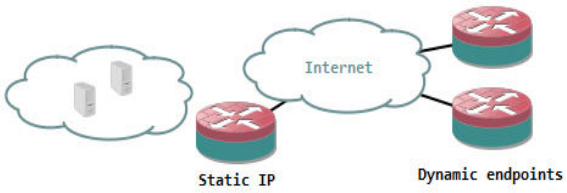
  What is a virtual private network (VPN)?

  A virtual private network (VPN) is a network that uses a public telecommunication infrastructure, such as the Internet, to provide remote offices or individual users with secure access to their organization's network. A virtual private network can be contrasted with an expensive system of owned or leased lines that can only be used by one organization. The goal of a VPN is to provide the organization with the same capabilities, but at a much lower cost.

- **Types of VPN:**

→ **Site-to-site VPN**
Often abbreviated to S2SVPN. It's a connection between two sites and encrypts all traffic between two (or multiple) subnets. There are two types of S2SVPN:

- Policy-based: interesting traffic triggers an ACL and is encrypted and sent to the remote VPN peer.
- Routed: traffic is routed into an encrypted tunnel to the remote VPN peer.
- For a detailed explanation and configuration, Jeremy made some excellent posts about this on Packetlife: Part 1 for policy-based and Part 2 for routed.



→ **DMVPN**

→ A dynamic multipoint VPN is not a protocol but more a technique using different protocols. One or more central hub routers are required, but the remote (spoke) routers can have dynamic IPs and more can be added

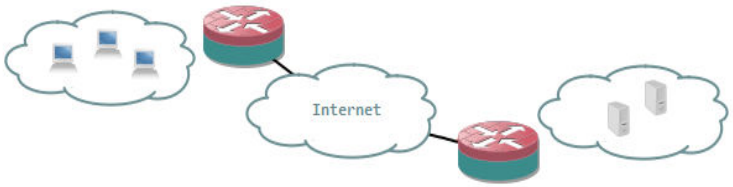
without having to modify the configuration on the hub router(s), or any other spoke routers.

→ The routers use a next-hop resolution protocol, combined with a dynamic routing protocol to discover remote peers and subnets. The VPN itself is a mGRE tunnel (GRE with multiple endpoints) which is encrypted. This way, traffic between spoke routers does not have to go through the hub router but can be sent directly from spoke to spoke.

→ ***Different types of VPN explained.***

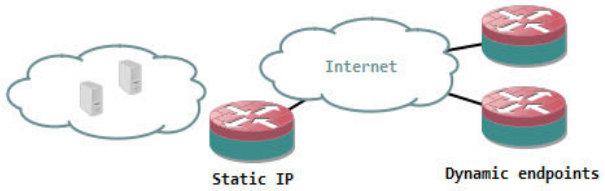
→ Since I'm going to talk more about VPNs in the upcoming weeks, I'm going to explain the different types of VPN here. No configuration guides, but an explanation so it's clear what is what.

→ For those who aren't sure what a VPN is: a **V**irtual Private **N**etwork is an encrypted connection between two or more devices over a public network. Some may argue that it doesn't necessarily has to be encrypted, but when it's not, that's called a tunnel (for me at least). Here's a list of the types:



→ **Site-to-site VPN**
Often abbreviated to S2SVPN. It's a connection between two sites and encrypts all traffic between two (or multiple) subnets. There are two types of S2SVPN:

- Policy-based: interesting traffic triggers an ACL and is encrypted and sent to the remote VPN peer.
- Routed: traffic is routed into an encrypted tunnel to the remote VPN peer.
- For a detailed explanation and configuration, Jeremy made some excellent posts about this on Packetlife: Part 1 for policy-based and Part 2 for routed.



- **DMVPN**
  A dynamic multipoint VPN is not a protocol but more a technique using different protocols. One or more central hub routers are required, but the remote (spoke) routers can have dynamic IPs and more can be added without having to modify the configuration on the hub router(s), or any other spoke routers. The routers use a next-hop resolution protocol, combined with a dynamic routing protocol to discover remote peers and subnets. The VPN itself is a mGRE tunnel (GRE with multiple endpoints) which is encrypted. This way, traffic between spoke routers does not have to go through the hub router but can be sent directly from spoke to spoke.

- **Client VPN**
  A client VPN is an encrypted connection from one device towards a VPN router. It makes that one remote device appear as a member of a local subnet behind the VPN router. Traffic is tunneled from the device (usually a computer or laptop of a teleworker) towards the VPN router so that user has access to resources inside the company. It requires client software that needs to be installed and configured.

- **SSLVPN**
  This type of VPN works like a client VPN. The difference is that the

remote client does not need preconfigured software, but instead the browser acts as VPN software. The browser needs to support active content, which every modern browser supports, either directly or through a plug-in. Traffic is tunneled over SSL (or TLS) to the SSLVPN router. From a networking perspective, traffic is tunneled over layer 4 instead of layer 3. The benefit is that the remote user does not need to configure anything and can simply log in to a web page to start the tunnel. The drawback that you'll likely need a dedicated device as SSLVPN endpoint because this is not a standard feature.

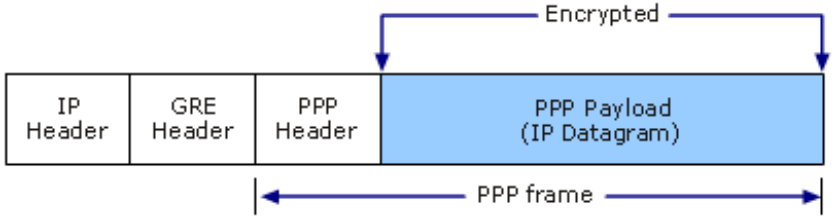
❖ **VPN Protocols (PPTP, L2TP, IPSec.)**
  - PPTP:

PPTP allows multiprotocol traffic to be encrypted and then encapsulated in an IP header to be sent across an IP network or a public IP network, such as the Internet. PPTP can be used for remote access and site-to-site VPN connections. When using the Internet as the public network for VPN, the PPTP server is a PPTP-enabled VPN server with one interface on the Internet and a second interface on the intranet.

Encapsulation

PPTP encapsulates PPP frames in IP datagrams for transmission over the network. PPTP uses a TCP connection for tunnel management and a modified version of Generic Routing Encapsulation (GRE) to encapsulate PPP frames for tunneled data. The payloads of the encapsulated PPP frames can be encrypted, compressed, or both. The following figure shows the structure of a PPTP packet containing an IP datagram.

Structure of a PPTP Packet Containing an IP Datagram



  - L2TP

L2TP allows multiprotocol traffic to be encrypted and then sent over any medium that supports point-to-point datagram delivery, such as IP or asynchronous transfer mode (ATM). L2TP is a combination of PPTP and Layer 2 Forwarding (L2F), a technology

developed by Cisco Systems, Inc. L2TP represents the best features of PPTP and L2F.

Unlike PPTP, the Microsoft implementation of L2TP does not use MPPE to encrypt PPP datagrams. L2TP relies on Internet Protocol security (IPsec) in Transport Mode for encryption services. The combination of L2TP and IPsec is known as L2TP/IPsec.

Both L2TP and IPsec must be supported by both the VPN client and the VPN server. Client support for L2TP is built in to the Windows Vista® and Windows XP remote access clients, and VPN server support for L2TP is built in to members of the Windows Server® 2008 and Windows Server 2003 family.

L2TP is installed with the TCP/IP protocol.

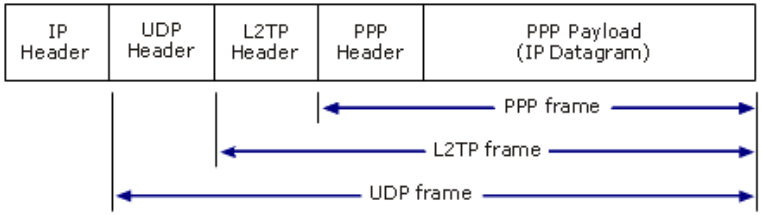
Encapsulation

Encapsulation for L2TP/IPsec packets consists of two layers:

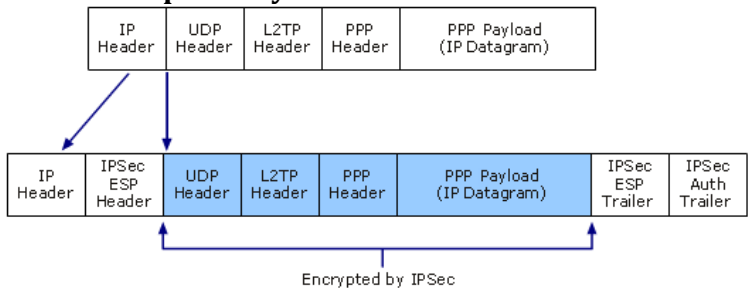
First layer: L2TP encapsulation

A PPP frame (an IP datagram) is wrapped with an L2TP header and a UDP header.

The following figure shows the structure of an L2TP packet containing an IP datagram.



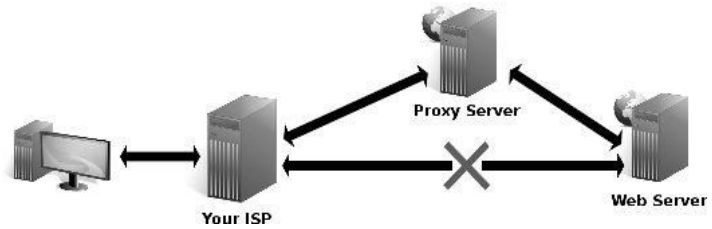
Encryption of L2TP Traffic with IPsec ESP



- **Proxy Server:**

During a HTTP connection, the IP address of the client machine is necessarily transmitted in order to get the information back. This allows a server to identify the source of the web request. Any resource you access can gather personal information about you through your unique IP address - your ID in the Internet. They can monitor your reading interests, spy upon you and log your requests for third parties. Also, owners of the Internet resources may impose some restrictions on users from certain countries or geographical regions.

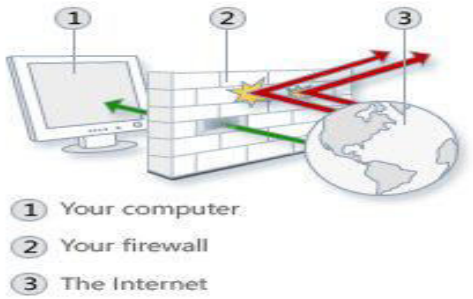
An anonymous proxy server acts as a middleman between your browser and an end server. Instead of contacting the end server directly to get a web page, the browser contacts the proxy server, which forwards the request on to the end server. When the end server replies, the proxy server sends the reply to the browser. No direct communication occurs between the client and the destination server, therefore it appears as if the HTTP request originated from the intermediate server. The only way to trace the connection to the originating client would be to access the logs on the proxy server (if it keeps any). So an anonymous proxy server can protect your identity by stripping a request of all identifying information.



- **Firewall**

A firewall is a software program or piece of hardware that helps screen out hackers, viruses, and worms that try to reach your computer over the Internet.

If you can't start Windows Firewall or you are getting an error, use our free tool to diagnose and fix problems.



- If you use a computer at home, the most effective and important first step you can take to help protect your computer is to turn on a firewall.
- Windows 8, Windows 7, Windows Vista, and Windows XP SP2 or higher have a firewall built-in and turned on by default. (**Note:** Support for Windows XP ends in April 2014.)
- If you have more than one computer connected in the home, or if you have a small-office network, it is important to protect every computer. You should have a hardware firewall (such as a router) to protect your network, but you should also use a software firewall on each computer to help prevent the spread of a virus in your network if one of the computers becomes infected.
- If your computer is part of a business, school, or other organizational network, you should follow the policy established by the network administrator.

GPS

Global Positioning System was developed by the United States'
GPS is often used by civilians as a navigation system. On the ground, any GPS receiver contains a computer that "triangulates" its own position by getting bearings from at least three satellites. The result is provided in the form of a geographic position - longitude and latitude - to, for most receivers, within an accuracy of 10 to 100 meters. Software applications can then use those coordinates to provide driving or walking instructions.

Getting a lock on by the GPS receivers on the ground usually takes some time especially where the receiver is in a moving vehicle or in dense urban areas. The initial time needed for a GPS lock is usually dependent on how the GPS receiver starts. There are three types of start - hot, warm and cold.

The **hot start** is when the GPS device remembers its last calculated position and the satellites in view, the almanac used (information about all the satellites in the constellation), the UTC Time and makes an attempt to lock onto the same satellites and calculate a new position based upon the previous information. This is the quickest GPS lock but it only works if you are generally in the same location as you were when the GPS was last turned off.

The **warm start** is when the GPS device remembers its last calculated position, almanac used, and UTC Time, but not which satellites were in view. It then performs a reset and attempts to obtain the satellite signals and calculates a new position.

The receiver has a general idea of which satellites to look for because it knows its last position and the almanac data helps identify which satellites are visible in the sky. This takes longer than a hot start but not as long as a cold start.

And finally – the **cold start** is when the GPS device dumps all the information, attempts to locate satellites and then calculates a GPS lock. This takes the longest because there is no known information.

The GPS receiver has to attempt to lock onto a satellite signal from any available satellites, basically like polling, which takes a lot longer than knowing which satellites to look for. This GPS lock takes the longest.

In an attempt to improve lock times, cellphone manufacturers and operators have introduced the Assisted GPS technology, which downloads the current ephemeris for a few days ahead via the wireless networks and helps triangulate the general user's position with the cell towers thus allowing the GPS receiver to get a faster lock at the expense of several (kilo)bytes.

- **GPRS:**

Definition: GPRS is a cellular networking service that supports WAP, SMS text messaging, and other data communications. GPRS technology is integrated into so-called *2.5G* mobile phones designed to provide faster data transfer speeds than older 2G cellular networks.

General packet radio service (GPRS) is a packet oriented mobile data service available to users of the 2G cellular communication systems global system for mobile communications (GSM), as well as in the 3G systems. In 2G systems, GPRS provides data rates of56-114kbit/s.
GPRS data transfer is typically charged per megabyte of traffic transferred, while data communication via traditional circuit switching is billed per minute of

connection time, independent of whether the user actually is using the capacity or is in an idle state. GPRS is a best-effort packet switched service, as opposed to circuit switching, where a certain quality of service (QoS) is guaranteed during the connection for non-mobile users.
2G cellular systems combined with GPRS are often described as 2.5G, that is, a technology between the second (2G) and third (3G) generations of mobile telephony. It provides moderate speed data transfer, by using unused time division multiple access (TDMA) channels in, for example, the GSM system. Originally there was some thought to extend GPRS to cover other standards, but instead those networks are being converted to use the GSM standard, so that GSM is the only kind of network where GPRS is in use. GPRS is integrated into GSM Release 97 and newer releases. It was originally standardized by European Telecommunications Standards Institute (ETSI), but now by the 3rd Generation Partnership Project (3GPP).
GPRS was developed as a GSM response to the earlier CDPD and i-mode packet switched cellular technologies.

- **CCTV Technology:**

Closed-circuit television (**CCTV**) is the use of video cameras to transmit a signal to a specific place, on a limited set of monitors. It differs from broadcast television in that the signal is not openly transmitted, though it may employ point to point (P2P), point to multipoint, or mesh wireless links. Though almost all video cameras fit this definition, the term is most often applied to those used for surveillance in areas that may need monitoring such as banks, casinos, airports, military installations, and convenience stores. Videotelephony is seldom called "CCTV" but the use of video in distance education, where it is an important tool, is often so called.[1][2]

In industrial plants, CCTV equipment may be used to observe parts of a process from a central control room, for example when the environment is not suitable for humans. CCTV systems may operate continuously or only as required to monitor a particular event. A more advanced form of CCTV, utilizing digital video recorders[3] (DVRs), provides recording for possibly many years, with a variety of quality and performance options and extra features (such as motion-detection and email alerts). More recently, decentralized IP-based CCTV cameras, some equipped with megapixel sensors, support recording directly to network-attached storage devices, or internal flash for completely stand-alone operation. Surveillance of the public using CCTV is particularly common in many areas around the world

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
1.	Introduction	Meaning of OS Functions of OS Features of OS

Chapter No. 1
Introduction

1.1 Meaning of Operating System:

It is general thing that we are using software and physical parts of computer like word, excel, power-point, printer, scanner etc. these all are software and physical parts of computer but we should think about that while we are booting computer at that time windows or any other software load itself, we are not able to load word, excel or any other application or developing software directly. Yes this is true because user can not deal directly with any physical parts of computer for that user requires a control programs which will control all physical parts of computer and also supporting to load other software and tools as per user requirement. Such a control program is known as operating system.

An operating system is one kind of software it is system software, which will support us to use hardware component and also providing us a platform to launch application software, developing software and system software too which will support us to perform some specific operation. It is a bridge between user and physical parts of computer.

Any computer needs a control program to boot itself without control program it is not possible to work on computer. So it is important that any computer must have its control program to identify user requirement and also providing such a service to satisfy user requirement. There are many operating systems available in market like Dos, Windows, Novel Netware, UNIX etc… All operating system have its own functionality. Which operating system should be chosen by the user it is depends on user's work.

Is Operating System a control Program?

It is true that an operating system is a control program because it controls all physical parts of computer as well as also identify user requirement and as per user requirement it provides support to user to perform specific operation. Any operation of user on computer which is totally controlled by operating system. It works like manager of any organization who is responsible for all works. The same way operating system is working in computer.

****

Summary

An operating system is platform between hardware and software, and it is a control program because it controls all physical parts of computer as well as also identity user requirement.

Function of Operating System

1.2 Functions of Operating System

Any computer needs a control program to boot itself without control program it is not possible to work on computer. So it is important that any computer must have its control program to identify user requirement and also providing such a service to satisfy user requirement. There are many operating systems available in market like Dos, Windows, Novel Netware, UNIX etc… All operating system have its own functionality. Which operating system should be chosen by the user it is depends on user's work.

"An operating system is a system software, control program and a bridge between user and computer that will support us to use hardware resources and also to launch application, development and system software"

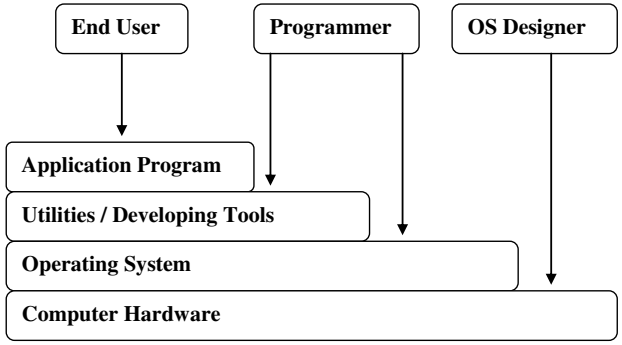
Function of Operating System

Different operating system has different functionality. But we can guess some general purpose or functionality of all operating system which are discussed below.

(1) The operating system as a user/computer interface

The end user generally is not concerned with the details of the computer's hardware. Thus the end user views a computer system in terms of a set of application.

Such a application can be expressed in programming language and it is developed by application programmer. If one were to develop an application program then first he/she will have to use set of machine instruction that is a complex task. To ease this task a set of system program is provided. Such a program is known as operating system. The operating system masks the details of the hardware from the programmer and provides the programmer with a convenient interface for using the system. It acts as mediator, making it easier for the programmer and for application programs to access and user those facilities and services.



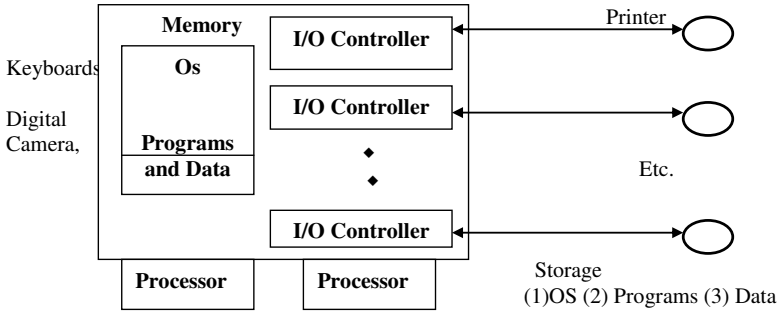
Layers and Views of Computer System

Services of Operating System

(1) Program Development
(2) Program Execution
(3) Access I/O Devices
(4) Control Access To Files
(5) System Access
(6) Error Detection And Response
(7) Accounting

(2) The Operating System As Resource Manager

A computer system is a set of resources for the movement, storage and processing of data and for the control of function. The operating system is responsible for managing these resources.



The Operating System As Resource Manager

Operating system provides services to user that he/she can use computer physical parts and he/she can get his/her work with from computer with high efficiency. Example If we wants to have a print on page then we can use devices like printer this is a sure resource that will support to user to have a print on page. In that way there are many internal and external resources there. To use such resources you need a program which is control all resources and providing you the best service to get work with efficiency. This program known as operating system. Operating system manages all resources of computer it works like a resource manager and manage all resources which in the system.

(3) Ease of Evolution of an Operating Systems

1. Hardware upgrades plus new types of hardware

Operating system is going to be evaluated at hardware point of view itself day by day if we take example in earliest version of os we did have Graphics User Interface environment we were using Typing User Interface but at that time we did have such a hardware which support us to run GUI application but now a day we can…

2. New Services

Operating system provides service to user as his/her demands. Day by day services given in operating system are going to be increased. For example If user found any difficulty to maintain good performance with existing tools, new measurement and control tools may be added to the operating system.

3. Fixes

Any operating system has faults. These are discovered over the course of time and fixes are made. The fix may introduce new faults.

Summary

Functions of O.S. are memory management, I/O management, Process management, File management, Security management, Exception management.

Evaluation of Operating System

Evaluation of Operating System

To use computer resource and to launch application software, developing software and system software we need one control program to control the hardware resource and also able to execute program and understand user requirement and giving response to user such a control program known as operating system. There are many type of operating system available in market. They have their own functionality. Earlier day we had some limited source but day by day updating has been done in operating system.

Evolution:-

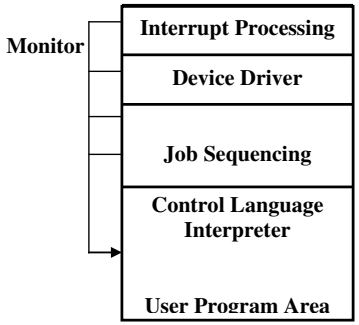
To attempting to understand the key requirements for an operating system and the significance of the major features of operating system, it is useful to consider how operating system has evolved over the year.

1. Batch Processing

Early machines were very expensive therefore it was important to maximize machine utilization. The wasted time due to scheduling and setup time was unacceptable.

To improve utilization a batch operating system was developed. It was developed in mid 1950s. The central idea behind the batch processing operating system scheme was the use of a piece of software known as the monitor. With the use of this type of operating system, the user no longer has direct access to the machine. Rather, the user submits the job on cards or tape to a computer operator, who batches the jobs together sequentially and places the entire batch on an input

device, for the use by the monitor. Each program is constructed to branch back to the monitor when it completes processing at which point the monitor would automatically begin loading the next program.



Boundary

To understand how this scheme works, let us look at it form two point of view: Monitor and Processor.

Monitor Point Of View:

It is monitor that controls the sequence of event. Monitor must always be in main memory and available for execution such a monitor known as resident monitor. The rest of the monitor that are loaded as subroutines to the user program at the beginning of any job device. As it reads in current job is placed in the user program are, and control is passed to the job. When the job is completed, it returns of each job are sent to an output device such as a printer, for delivery to user.

Process Point Of View:

At a certain point, the processor is executing instruction from the portion of main memory containing the monitor. These instructions cause the next job to read into another portion of main memory. Once a job has been read in, the processor will encounter a branch instruction in the monitor that instructs the processor to continue execution at the start of the user program. The processor will then execute the instruction in the user's program until it encounters an ending or error condition. Either event cause the processor to fetch its next instruction from the monitor program. Thus the phrase "control is passed to a job" simply means that the processor is now fetching and executing instructions in a user program is now fetching and executing instructions from the monitor program.

2. Multiprogramming Operating System

Even with the automatic job sequence provided by a batch operating system, the processor is often idle. The problem is that I/O devices are slow compared to processor. So processor is idle for such a time. To solve this problem multiprogramming operating system was developed. This operating system having same sequence of process which batch operating system has but difference is there that in batch processing system there is only one program is loaded in memory and waiting for I/O response but in multiprogramming there is one or more then one programs are loaded in main memory. If one process is going on and job waiting for I/O, the processor can switch to the other job, which likely not waiting for I/O. Furthermore, we might expand memory to hold three, four or more programs and switch among all of them. The process is known as multiprogramming or multitasking. It is the central theme or modern operating system.

3. Time-Sharing System

With the use of multiprogramming, batch processing can be quite efficient. However, for many jobs, it is desirable to provide a mode in which the user interacts directly with the computer. Indeed, for some jobs, such as transaction processing, an interactive mode is essential.

Today, the requirement for an interactive computing facility can be, and often is, met by the use of a dedicated microcomputer. That option was not available in the 1960s, when met computers were big and costly. Instead time sharing was developed.

Just as multiprogramming allows the processor to handle multiple batch jobs at a time, multiprogramming can be used to handle multiple interactive jobs. In this latter case, the technique is referred to as time sharing, because the processor's time is shared among multiple users. In a time-sharing system, multiple users simultaneously access the system through terminals, The time sharing Operating System also be multiprogramming operating system but difference is there compared to batch processing and multiprogramming like that the batch processing and multiprogramming operating system support one user at a time but time sharing system support multiple user and processor giving particular time to each user.

Real Time Sharing Operating System:

Real time operating system is used in environments where a large number of events, mostly external to the computer system, must be accepted and processed in a

short time or within certain deadlines. Such applications include industrial control: telephone switching equipment, flight control and real time simulations. Real time systems are also frequently used in military applications.

A primary objective of real-time systems is to provide quick event-response times, and thus meet the scheduling deadlines. The processor is normally allocated to the higher priority processes usually preempt execution of the lower-priority processes. This form of scheduling called priority-base preemptive scheduling is used by a majority of real-time systems. Memory management in real-time system is comparatively less demanding than in other types of multiprogramming system. The primary reason for this is that many processes permanently reside in memory in order to provide quick response time.

Summary

Types of Operation System:-

Batch O.S, Multiprogramming O.S, Multiprocessing O.S, Time Sharing O.S, Real-time O.S

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
2.	**Process Management**	Introduction of OS process Process State Transition Diagram Process Scheduling

Chapter No. 2
Introduction of OS System Process

1. Introduction Of Operating System Process

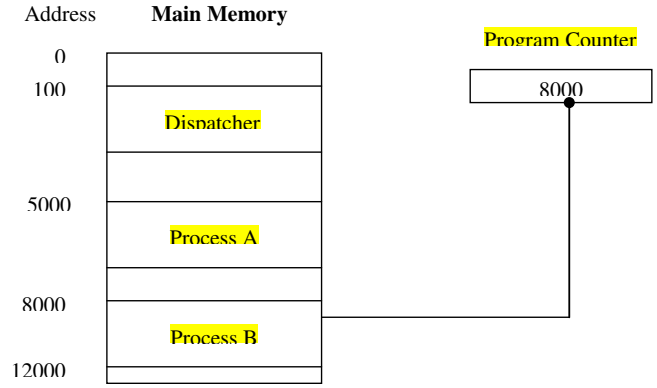
In general life we are doing our routine work that is known as our task which generally we will have to perform daily or routine. As we know that user can't deal directly with computer there a control program which supports to user to perform any specific operation. To perform any operation, operating system has to perform some task that task known as process.

The fundamental of any modern operating system is process management. The operating system must allocate resources to processes, enable processes to share and exchange information, protect the resources of each process from other processes, and enable synchronization among processes. To meet this requirement, the operating system must maintain a data structure for each process that describes the state and resource ownership of the process and that enables the operating system to exert process control.

2. Process States Transaction Diagram

The principal function of a processor is to execute machine instruction residing in main memory. These instructions are provided in the form of program. A process or task is created for the program to execute involves a sequence of instruction within that program. We can characterize the behavior of an individual process by listing

the sequence of instructions that execute for the process. Such a listing is referred to as a trace of the process.

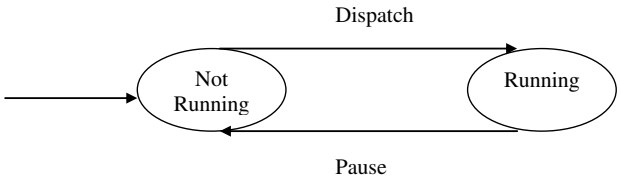


Above figure shows a memo Process C e processes. These three processes are fully loaded in main memory. In addiction, there is a small dispatcher program that switches the processor from one process to another. The address given above figure are the starting address of any process and program counter indicates the process B is going to be executed and A and C are waiting for I/O operation.

A Two-State Process Model:

The operating system's principal responsibility is controlling the execution of processes; this includes determining the interleaving pattern for execution and allocating resources to processes. The first step in designing a program to control processes is to describe the behavior that we would like the processes to exhibit.

We can construct the simplest possible model by observing that, at any time, a process is either being executed by a processor or not. Thus, a process may be in one of two states: Running or Not Running.



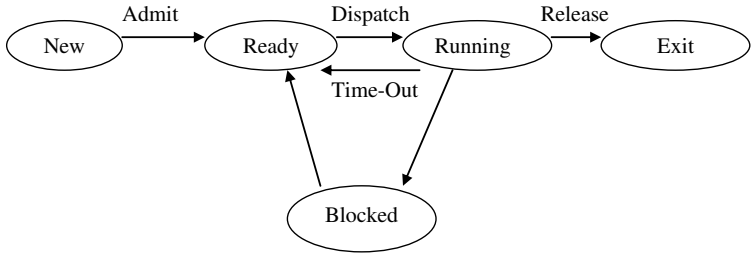
State Transition Diagram

When the operating system creates a new process, it enters that process into the system in the No Running State. The process exists, is known to the operating system, and is waiting for an opportunity to execute. From time to time, the currently running process will be interrupted and the dispatcher portion of the operating system will select a new process to run. The former process moves from the Running state to the Not Running state, and one of the other processes moves to the Running state.

A Five-State Model

If all processes were always ready to execute, then the queuing description is necessary. The queue is a first-in-first-out and the processor operates in round-robin fashion on the available process. However, even with the simple example that we have described that some processes in the Not Running state are ready to execute, while other are blocked, waiting for and I/O operation to complete. Thus, using a single queue, the dispatcher could not just select the process at the oldest end of the queue. Rather, the dispatcher would have to scan the list looking for the process that is not blocked and that has been in the queue the longest.

A more natural way to handle this situation is to split the Not Running state into two states: Ready and Blocked. This is shown in given below figure.

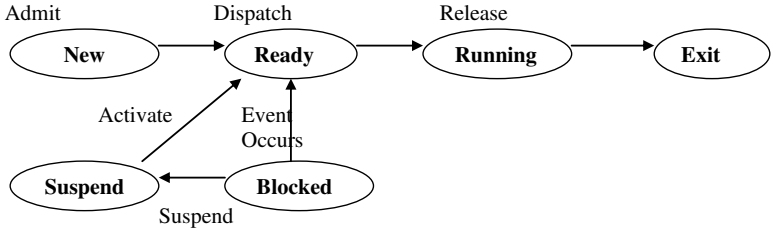


Five State Process Model

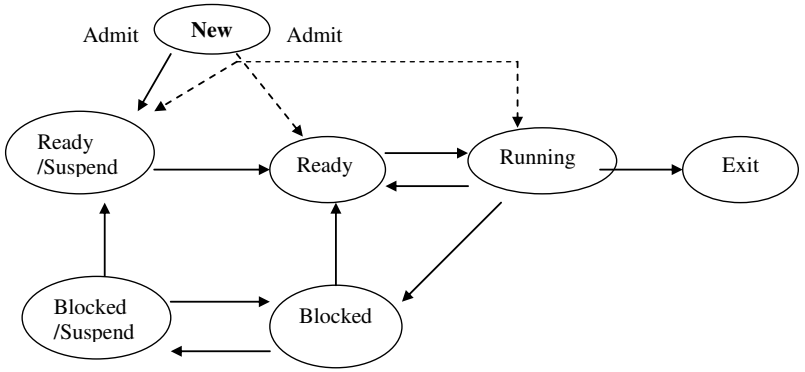
- **Running:** The process that is currently being executed. One process can be in this state at a time
- **Ready:** A process that is prepared to execute when given the opportunity.
- **Blocked:** A process that cannot execute until some event occurs, such as the completion of an I/O operation.
- **New:** A process that has just been created but has not yet been admitted to the pool of executable processes by the operating system. Typically, a new process has not yet been loaded into main memory.
- **Exit:** A process that has been released from the pool of executable processes by the operating system, either it halted or because it aborted for some reason.

The three principal states just described (Ready, Running, Blocked) provide a systematic way of modeling he behavior of processes and guide the implementation of the operating system. Many operating systems are constructed using just these three states.

But many operating systems using other states to justify better. Such a state known as suspended process. Given below diagram will explain well about these new states.



Process State Transition Diagram with suspend state



Process State Transition Diagram with two suspend state

- **Ready:** The process is in main memory and available for execution.
- **Blocked:** The process is in main memory and awaiting an event.
- **Running:** The process is in main memory and going to be executed
- **Blocked/Suspend:** The process is in secondary memory and awaiting an event
- **Ready/Blocked:** The process is in secondary memory but is available for execution as soon as it is loaded into main memory.

◆Blocked -> Blocked/Suspend:

If there are no ready processes, then at least one blocked process is swapped out to make room for another process that is not blocked. This transition can be made even if there are ready processes available, if the operating system determines that the currently running process or a ready process that it would like to dispatch requires more main memory to maintain adequate performance.

◆ Blocked/Suspend -> Ready/Suspend:

A process in the Blocked/Suspend states is moved to the Ready/Suspend state when the event for which it has been waiting occurs. Note that this requires that the state information concerning suspended processes must be accessible to the operating system.

◆ Ready/Suspend->Ready:

When there are no ready processes in main memory, the operating system will need to bring one in to continue execution. In addition, it might be the case that a process in the Ready/Suspend state has higher priority than any of the processes in the Ready state. In that case, the operating system designer may dictate that it is more important to get at the higher-priority process than to minimize swapping.

◆ Ready -> Ready/Suspend:

Normally, the operating system would prefer to suspend a blocked process rather than a ready one, because the ready process can now be executed, whereas the blocked process is taking up main memory space and cannot be executed. However, it may be necessary to suspend a ready process if that is the only way to free up a sufficiently large block of main memory. Also, the operating system may choose the suspend a lower-priority ready process rather than a higher-priority blocked process if it believes that the blocked process will be ready soon.

◆New->Ready/Suspend and New->Ready:

When a new process is created, it can either be added to the Ready queue or the Ready/Suspend queue. In either case, the operating system must build some tables to manage the process and allocate an address space to the process. It might be preferable for the operating system to perform these housekeeping duties at early times, so that it can maintain a large pool of processes that are not blocked. With this strategy, there would often be insufficient room in main memory for a new process; hence the use of the

(new -> Ready/Suspend) transition. On the other hand, we could argue that a just-in-time philosophy of creating processes as late as possible reduces operating system overhead and allows that operating system to perform the process-creation duties at a time when the system is clogged with blocked processes anyway.

◆ Blocked/Suspend->Blocked:

Inclusion of this transition may seem to be poor design. After all, if a process is not ready to execute and is not already in main memory, what is the point of bringing it in? But consider the following scenario: A process terminates, freeing up some main memory. There is a process in the Blocked/Suspend queue with and the operating system has reason to believe that the blocking event for that process will occur soon. Under these circumstances, it would seem reasonable to bring a blocked process into main memory in preference to a ready process.

◆ Running -> Ready/Suspend:

Normally, a running process is moved to the Ready state when its time allocating expires. If, however, the operating system is preempting the process because a higher-priority process on the Blocked/Suspend queue has just become unblocked, the operating system could move the running process directly to the (Ready/Suspend) queue and free some main memory.

◆ Various -> Exit:

Typically, a process terminates while it is running, either because it has completed or because of some fatal fault condition. However, in some operating systems, a process may be terminated by the process that created it or when the parent process is itself terminated. If this is allowed then a process in any state can be moved to the Exit state.

Summary

Process is an event or Program in execution mode.
Process states are:-

1. New
2. Ready
3. Running
4. Block/Waiting
5. Terminated/Stop

Process Scheduling

The aim of processor scheduling is to assign processes to be executed by the processor or processors over time, in a way that meets system objectives, such as response time, throughput, and processor efficiency.

Scheduling affects the performance of the system because it determines which processes will wait and which will progress. Fundamentally, scheduling is a mater of managing queues to minimize queuing delay and to optimize performance in a queuing environment.

FCFS (Firs-Come-First-Served)

The simplest scheduling policy is first-come-first-served (FCFS), also known as first-in-first-out (FIFO) or a strict queuing scheme. As each process becomes ready, it joins the ready queue. When the currently running process ceases to execute, the process that has been in the ready queue the longest is selected for running.

First-Come-First-Served performs much better for long processes than short ones. Consider the following example,

Process	Arrival Time	Service Time	Start Time	Finish Time	Turnaround Time	T_r/T
W	0	1	0	1	1	1
X	1	100	1	101	100	1
Y	2	1	101	102	100	100
Z	3	100	102	202	199	1.99
Mean					100	26

The normalized turnaround time for process Y is way out of time compared to the other processes: The total time that it is in the system is 100 times the required processing time. This will happen whenever a short process arrives just after a long process. On the other hand, even in this extreme example, long processes do not fare poorly. Process Z has a turnaround time that is almost double that of Y, but its normalized waiting time is under 2.0.

Another difficulty with FCFS is that it tends to favor processor-bound processes over I/O-bound processes. Consider that there is a collection of processes, one of which mostly uses the processor (processor bound) and a number of which favor I/O (I/O bound). When a processor-bound process is running, all of the I/O-bound processes must wait. Some of these may be in I/O queue (blocked state) but may move back to the ready queue while the processor-bound process is executing tidally work for them to do. When the currently running process leaves the Running state, the ready I/O-bound processes quickly move through the Running state and become blocked on I/O events. If the

processor-bound process is also blocked, the processor becomes idle. Thus, FCFS may result in inefficient use of both the processor and the I/O devices.

FCFS is not an attractive alternative on its own for a single-processor system. However, it is often combined with a priority scheme to provide an effective scheduler. Thus, the scheduler may maintain a number of queues, one for each priority level, and dispatch within each queue on a first-come-first-served basis.

Priority Base Non Preemptive And Preemptive

The selection function determines which process, among ready processes, is selected next for execution. The function may be on priority, resource requirement, or the execution characteristics of the process. In latter case three quantities are significant.

w = time spent in system so far, waiting and executing
e = time spent in execution so far
s = total service time required by the process, including e; generally, this quantity must be estimated or supplied by the user.

For example, the selection function max[w] indicates a first-come-first-served discipline.

This decision mode specifies the instants in time at which the selection function is exercised. There are two general categories. (1) Non Preemptive (2) Preemptive.

Non Preemptive

In this case, once a process is in the Running state, it continues to execute until (a) it terminates or (b) blocks itself to wait for I/O or to request some operating system service.

Preemptive

The currently running process may be interrupted and moved to the Ready state by the operating system. The decision to preempt may be performed when a new process arrives, when an interrupt occurs that places a blocked process in the Ready state, or periodically based on a clock interrupt.

Preemptive policies incur greater overhead than non preemptive once but may provide better service to the total population of processes, because they prevent any one process from monopolizing the processor for very long. In addition, the cost of preemption may be kept relatively low by using efficient process-switching mechanism and providing a large main memory to keep a high percentage of program in main memory.

Round Robin

Round Robin is particularly effective in a general-purpose time-sharing system or transaction processing system. One drawback to round robin is its relative treatment of processor bound and I/O bound processes. Generally, an I/O bound process has a shorter processor burst (Amount of time spent executing between I/O operation) than a processor-bound process. If there is a mix of processor – bound and I/O bound processes, than the following will happen: An I/O bound process uses a processor for a short period and then is blocked for I/O; it waits for the I/O operation to complete and then joins the ready queue. On the other hand, a processor-bound process generally uses a complete time quantum while executing and immediately returns to the ready queue. Thus, processor-bound processes tend to receive an unfair portion of processor time, which result in poor performance for I/O devices and an increase in the variance of response time.

Summary

Process scheduling Techniques are:

1. Preemptive
   - FCFS (First Come First Serve)
   - Event Driven/Priority Based
2. Non-Preemptive
   - SJF (Shortest Job First)
   - Round Robin

- **Difference between Multitasking/Time-Sharing and Multiprocessing and Multithreading**

(1) Multitasking/Time-Sharing Operating System.

Multitasking is a logical extension of the multiprogramming OS. Here CPU is multiplexed by time among several jobs that are kept in main memory and on disk.

Pseudo-parallelism is achieved by providing rapid CPU scheduling among jobs/processes. Thus, users can have good terminal response. It also provides direct interaction between user and the system.

It also allows sharing of computer among users simultaneously. It gives an illusion/impression to each user as if the entire computer resources are available to him solely.

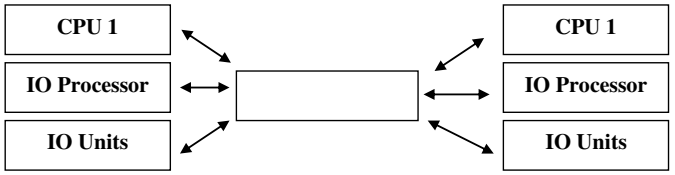
This system is more complex than multiprogramming, and requires following features:

- **Several jobs must be kept in memory simultaneously, which requires memory management and protection.**
- **As memory is limited, concepts like swapping and virtual memory are used.**
- **CPU scheduling should be sophisticated, to provide fairness to all processes.**
- **Disk management and file system is complex now.**
- **Job synchronization, communication among jobs and problems like deadlock should be overcome.**

(2) Multiprocessing Operating System

In multiprocessing operating system it is more then one CPUs are used, those are interconnected or can be independent but they execute together.

The instruction of the program is executed by more then one CPU and their memory.



This inputs are also taken for it CPU for processing. If one CPU breaks down the control is transferred to another CPU.

The main thing to be controlled is scheduling and job priority. If scheduling is not perfect then it can produce wrong result.

Advantages:

1. It improves performance of computer system by allowing parallel processing.
2. It also utilize CPU in method where ideal time is minimum.
3. If one CPU breaks down, the process is completed by another CPU so there is minimum break down.

Disadvantages:

1. The O.S. should be program to schedule balance and co-ordinate input output and processing with multiple CPU.
2. A large main memory is required.
3. It is very expensive.
4. It is difficult to maintain.

(3) Multithreading Operating System:

Multithreading OS supports the concept of multiple threads within single process environment.

Generally, an application is implemented as a separate process with several threads of control. Sometimes, a single application needs to perform more than one task simultaneously. For example, word processor needs to accept input, provide spell checking, and auto-saving. But, how all these are possible at the same time?

There are different threads in single process which perform these different tasks simultaneously. One thread may accept inputs, while other may provide spell checking, and third one may do auto saving.

Operating system, which supports such type of functionality, is called multithreading operating system.

Such systems provide advantages like Good response, Resource sharing, Better economy and Utilization of multiprocessor architectures.

Summary

Multiprocessing O.S:-

In multiprocessing operating system it is more then one CPUs are used, those are interconnected or can be independent but they execute together.

Multithreading O.S:-

Multithreading OS supports the concept of multiple threads within single process environment.

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
3.	**Memory Management**	Physical Memory and Virtual Memory Memory Allocation Paging Segmentation

Chapter No. 3

Q. 1 Physical Memory and Virtual Memory

Ans:

It is a common word while we are dealing with computer industry, without memory we can not deal with computer system, sometime we are comparing computer with human being because computer also having memory power to hold data for future reference. But remember any memory in computer has its own physical parts like Ram, Hard Disk etc.. but we classified some memory with their functionality like temporary memory, permanent memory. We also classified main memory in parts like main memory, virtual memory, cash memory etc…

Here we see the classification of two memory like logical memory and physical memory.

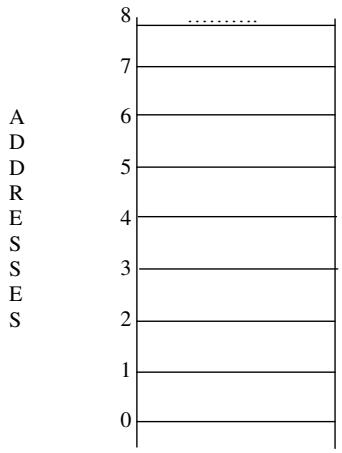
● **Physical Memory**

Physical memory is the workhorse of the memory system. It is also known as the main memory. It is internal to the computer system, and comes in the form of chips. The word main is used to distinguish it form external mass storage devices such as disk drives.

Physical memory constructs the middle layer in the memory hierarchy of a computer system. It is slower and cheaper compared to cache memory, while faster and costly compared to disks.

Physical memory generally consists of Random Access memory (RAM). It is volatile, means it needs to have electrical power in order to maintain its information.

When power is lost, the information is lost too. By word 'random' means that any piece of data from this memory can be accessed quickly in constant time.



Main/Physical Memory

Generally all the information is stored in disks in form of various files. But access from disk is much slower. So, any information to be used must be brought into main memory from disk before using it. Such information can be directly accessed by the CPU.

Physical memory is expensive compared to disks. So it has limited storage capacity. The capacity available for a given price is increasing all the time. For example many home personal computer now have a capacity of 512 MB or 1 GB.

The entire physical memory can be considered as sequential list of bytes. In most modern computers, each byte has an address that is used to locate it. This address is called physical address.

The above figure depicts the small part of a main memory. Each box in this figure represents a single byte. Each byte has an address. In this figure the addresses are the integers to the left of the boxes: 0,1,2,3,4……. and so on. The addresses for most computer memory start at 0 and go up in sequence until each byte has an address.

●Virtual Memory

Definition

"A virtual memory is a technique that allows a process to execute even though it is partially loaded in main memory"

Virtual memory removes the requirement that an entire process should be in main memory for its execution. The main advantage of this is: a process can be larger than the main memory (Remember the question of executing 40-MB process in 32-MB memory? Here, it is possible).

Here logical addresses are referred as virtual addresses. Also, logical address space of a process is referred as virtual address space. Virtual address space can be larger then the physical memory.

Motivations behind this technique:

From real time programs it is observed that the entire program is not needed to be in memory for its execution. Consider the following observations-

1. **Programs contain codes to handle unusual error conditions. Such errors occur very rarely, and so such codes execute in rare situations.**
2. **Generally arrays are allocated more memory than the actual requirement.**
3. **Programs contain certain operations and features which are used very rarely So, such types of code and data part need not to be in memory all the time.**

Also, even in cases where the entire program is needed, it may not be needed at the same time.

These all factors motivate the technique of virtual memory.

Advantages:

There are two main advantages of virtual memory.

1. Programs (and so processes) are not constrained by physical memory size. A process larger than the memory can also execute.
2. Degree of multiprogramming can be varied over a large range. As there is no need to keep an entire process in memory, more and more processes can be accommodated in memory.

Implementation:

The virtual memory can be implemented in one of the following three ways:

1. Demand paging
2. Demand Segmentation
3. Segmentation with Paging

In demand paging, pages are moved to memory as per requirement. Similarly, in demand segmentation, segments are moved only as per requirement. Here, on all pages or all segments are moved to memory together at the same time. In segmentation with paging scheme, segments are divided into pages.

Q. 2 Difference Logical Address V/s Physical Address

● Logical Address

1. The process address space can be considered as a sequential list of bytes. Each byte has an address that is used to locate it. These addresses are called logical addresses.
2. Logical addresses are generated by the CPU. This means that CPU determines the location of each instruction and data in process address space.
3. Logical address space (LAS): It is a set of all logical addresses that can be referenced by a process.
4. Here, addresses in LAS starts from zero and goes up to some maximum value based on the size of the process.
5. Process can read and write addresses of its own logical address space.
6. The logical addresses are limited by the address size of the processor. For example, 32-bit processor can generate up to 232 (4 GB) addresses.

● Physical Address

1. The entire physical memory can be considered as a sequential list of bytes. Each byte has an address that is used to locate it. These addresses are called physical addresses.
2. Physical addresses are seen y the main memory.
3. Physical Addresses Space (PAS): it is a set of all physical addresses occupied by a process in main memory during its execution.
4. Here, address in PAS may or may not be contiguous based upon the memory allocation method.
5. Process can read and write only those physical addresses that belong to its own physical address space.
6. The physical addresses are limited to the amount of installed memory.

For any process to get executed, it should be in main memory. For these reason, whenever process (code, data, and stack) is fetched in main memory, the required storage space is allocated for that process. Now, here the physical addresses allocated to the process may not exactly match the logical addresses of the process. For this reason, there should be some mapping between physical addresses and logical addresses.

Q. 3 Memory Allocation (Contiguous Memory Allocation and Non Contiguous Memory Allocation?

User programs are stored in main memory before execution. Also, any data to be used is stored in main memory. In a similar way, operating system also needs to be stored in main memory.

For all these purposes, there is a need of memory allocation. Memory should be allocated to various processes and files (data) as per requirements. Memory allocation refers to the operations of allocating memory.

Also when there is no need, allocated memory should be freed.

Memory has a limited capacity compared to disk. And, it is possible that more than one process may be executing simultaneously. So, memory should be allocated carefully among all these processes.

While allocating memory, the two goals should be fulfilled:

1. High Utilization

- Maximum possible memory should be utilized
- No any single piece of memory should be wasted.

2. High Concurrency:

- Maximum possible processes should be in main memory
- As there are no more and more processes in memory, CPU will remain busy most of the time in executing one of the process; resulting in better thoughput.

Basically, there are two different ways in which memory is allocated. These are described below.

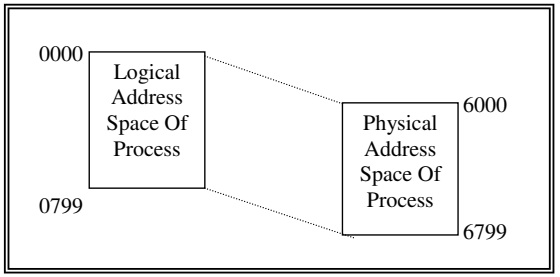
1. Contiguous Memory Allocation
2. Non-Contiguous Memory Allocation

1. Contiguous Memory Allocation

This is simple and old method of memory allocation. Generally, it is not used in modern operating system. Here, each process occupies a block of contiguous memory locations in main memory. Entire process is kept together in a contiguous section of memory. When a process is brought in main memory, a memory is searched to find out a chunk of free memory having enough size to hold a process. Once such chunk is found, required memory is allocated. If a contiguous memory space of the required size is not available in the main memory, the process is made to wait until contiguous space of the

required size is available. Here, logical address space is not divided into any partitions. Also, physical address space will be contiguous, without any gaps.

The following figure clears out this allocation method.



Contiguous Memory Allocation

As described in figure, in contiguous memory allocation, each process occupies a block of contiguous memory locations in main memory. Here, entire process is kept together in a contiguous section of memory.

There are three different way in which contiguous memory allocation method can be used.

1. Single Process Monitor
2. Multiprogramming with Fixed Partitions
3. Multiprogramming with Dynamic Partitions

1. Single Process Monitor

Here, only single process is allowed to run. No more than one process can run simultaneously. This is the simplest possible memory management scheme. Memory is shared between the process and the operating system.

2. Multiprogramming with Fixed Partitions

This method allows multiple processes to execute simultaneously. Here, memory is shared among the operating system and various simultaneously running processes. Multiprogramming increases the CPU Utilization. CPU can be kept busy almost all the time by keeping more than one process simultaneously in the memory.

3. Multiprogramming with Dynamic Partitions

This method is also allows multiple processes to execute simultaneously. Here, memory is shared among operating system and various simultaneously running processes. Here, memory is not divided into any fixed size partitions. Also the number of partitions is not fixed. Process is allocated exactly as much memory as it requires. Initially, the entire available memory is treated as a single free partition.

Advantages

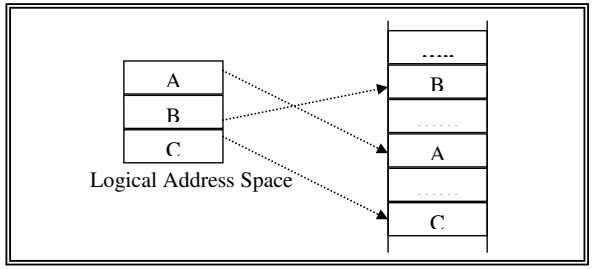
1. It is easy to implement.
2. It is easy to understand.

Disadvantages

1. Having poor memory utilization.

Non-Contiguous Memory Allocation

This method is used by most modern operating system. Here, logical address space of a process is divided into partitions. And, for each partition contiguous chunk of free memory is allocated. Physical address space will not be contiguous now. The following figure clears out this allocation method. Here, logical address space of a process is divided into three partitions (A,B and C) and each partition is allocated separate chunk of memory in physical memory.



Non-Contiguous Memory Allocation

Physical address space of a process will not be contiguous now. Physical addresses will be scattered over entire memory. **There are two kind of allocation techniques used here:** One is paging and other is segmentation.

Paging:

Here, the logical address space of a process is divided into blocks of fixed size, called pages. Also, the physical memory is divided into blocks of fixed size, called Frame or Page Frame.

Segmentation

This technique implements user's view of logical address space. From user's point of view, logical address space of any process is a collection of code, data and stack. Code can be comprise of main function, other user defined functions and library function. Data can be local variables, global variables, arrays, symbol table and other data structures.

Here, the logical address space of a process is divided into blocks of varying size, called segment. Each segment contains a logical unit of a process. Logical unit can be main function, other functions or procedures, stack, array, symbol table etc…

Advantages of Non-Contiguous Memory Allocation

Having better memory utilization

Disadvantages of Non-Contiguous Memory Allocation

It is complex to implement.

Q. 4 Explain Non-Contiguous Memory Allocation: Paging

Most of modern operating systems use this method for allocation memory. In this method, logical address space of a process is divided into partitions. For each partition contiguous chunk of free memory is allocated in physical memory.

Physical address space of a process will not be contiguous now. Physical addresses will be scattered over entire memory.

There are two kinds of allocation techniques used here: One is paging, descried below.

Paging

Here, the logical address space of a process is divided into blocks of fixed size, called Pages. Also, the physical memory is divided into blocks of fixed size, called Frames or Page Frames.

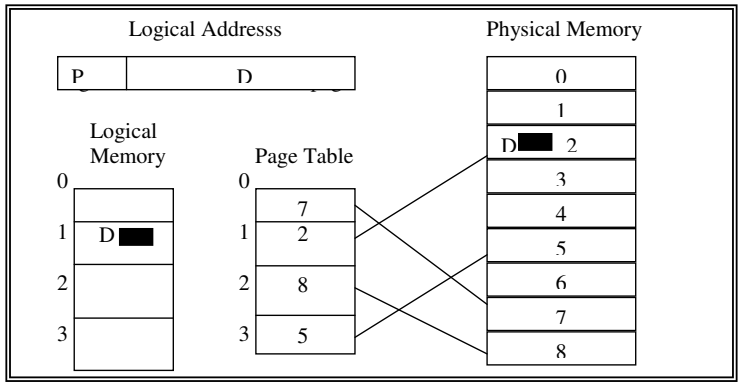
The pages and frames will be of the same size. This size will be typically a power of 2. It varies between 512 Bytes to a few MB.

Whenever a process is to be executed, its pages are moved from secondary storage, i.e. a disk, to the available frames in physical memory. So, memory allocation refers to the task of finding free frames in memory, allocating them to pages, and keeping track of which frame belongs to which page.

Operating system maintains a table, called page table, for each process. The page table is indexed by a page number, which gives the number of pages are stored, is kept in this page table.

Here, logical address is divided into two parts i) page number, which gives the number of a page; and ii) an offset, which gives the actual location within a page.

The following figure gives idea about this method. Here, the logical address space of a process, also called logical memory, has been divided into four pages. Physical memory contains total of 11 frames. Also, a page table is described which shows the mapping between pages and frames.



Basic Concept Of Paging

Implementation

The above figure show how the paging scheme is implemented. It describes how logical addresses are converted to physical addresses.

A table, called page tale, is used to implement paging. When a process is to be executed, its pages are moved to free frames in physical memory. The information about frame number, where a page is stored, is kept in this page table.

During the process execution, a CPU generates a logical address (L) to access instruction or data from a particular location. This logical address is divided into two parts: a page number (p) and an offset (d) within a page.

Logical Address (L)

Page Number (p)	Offset (d)

The page number is used to search the page table. Corresponding frame number (f) is obtained from the page table. This frame number indicates the actual frame on physical memory in which the page is stored.

The physical address (P) of a particular location is obtained by combining the frame number (f) with the offset (d).

Physical Address (P)

Frame Number (f)	Offset (d)

Address Translation:

Let the page size (or frame size) be 2^n bytes.

So, the logical address (L) can be divided into two parts, page number (P) and offset (d), such that –

Page Number (p) = $L / 2^n$, and

Page Offset (d) = $L \% 2^n$.

This page number (p) is used to find out corresponding frame number (f) from page table. Once frame number (f) is found, physical address can be found as –

Physical Address (P) = $f * 2^n + d$.

Advantages of Paging Method

1. **Allocation and De-Allocation** of memory is very fast. For this, only the list of free frames needs to be maintained. No need of any algorithms such as first fit, best fit, etc.
2. **Swap-in and Swap-out** operations are fast. As page size matches disk block size, data can be moved to and from disk very easily.
3. There is no external fragmentation. Each and every frame from memory can be allocated to processes.

Disadvantages of Paging Method

1. **Additional memory reference** is required to read information from page table. Every instruction or data reference requires two memory accesses: One for page table, and one for instruction or data.
2. **Size of page table** may be too large to keep it in main memory. Pages table contains entry for all the pages in logical address space. For large process, page table will be large.

3. **Internal fragmentation :** A process size may not be an exact multiple of the page size. So, some space would remain unoccupied in the last page of a process, page results in internal fragmentation.

Q. 5 Explain Non-Contiguous Memory Allocation: Segmentation

This is one of the techniques of non-contiguous memory allocation: segmentation. This technique implements user's view of logical address space. From user's point of view, logical address space of any process is a collection of code, data and stack. Code can be comprise of main function, other user defined functions and library function. Data can be local variables, global variables, arras, symbol table and other data structures.

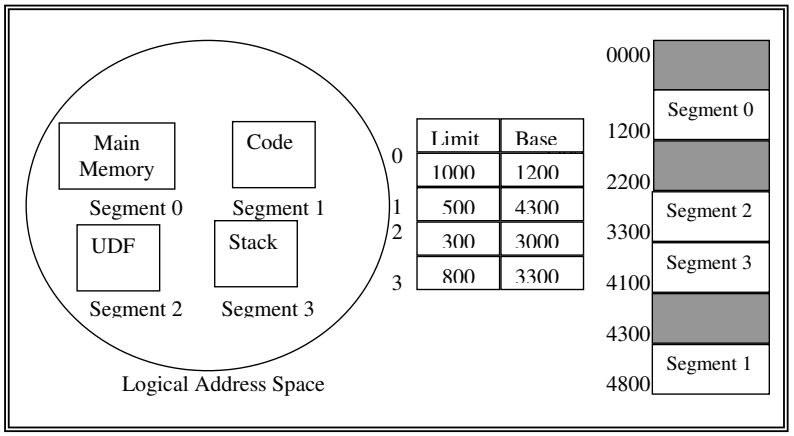
Here, the logical address space of a process is divided into blocks of varying size, called Segments. Each segment contains a logical unit of a process. Logical unit can be main function, other functions or procedures, stack, array, symbol table etc.

Each segment can be considered as a completely independent address space. It consists of linear sequence of addresses starting from '0' to some maximum limit. All segments are of varying lengths and length depends upon the size of a logical unit. All segments are given unique numbers to identify them.

Whenever a process is to be executed, its segments are moved from secondary storage, i.e. a disk, to the main memory. Each segment is allocated a chunk of free memory of the size equal to that segment.

Operating system maintains a table, called segment table, for each process. The segment table contains information about each segment of a process. This information includes size of segment and the location in memory where the segment has been loaded.

Here logical address is divided into two parts: i) segment number, which identifies a segment and ii) and offset, which gives the actual location within a segment



Basic Concept of Segmentation

The above figure gives ideas about this method. Here, the logical address space of a process has been divided into four segments. A physical memory is shown with these four segments. A segment table is described which gives information such as where each segment is loaded in physical memory and what is the size (length) of each segment.

Implementation

A table, called segment table, is used to implement segmentation. Each entry of the segment table has a segment base and segment limit. The segment base contains the starting physical address where the segment resides in memory. The segment limit specifies the length of the segment.

During the process execution, a CPU generates a logical address (L) to access instruction or data from a particular location. This logical address is divided into two parts: First, a segment number (s), and Second, and offset (d) within that segment.

Logical Address (L) :

Segment Number (s)	Offset (d)

The segment number is used as an index into the segment tale. The offset (d) of the logical address must be between 0 and the segment limit. If it is not, illegal address error will be generated. Else, if it is within limit, it is added to the segment base to produce the physical address.

Advantages of Non-Contiguous Memory: Segmentation

1. All segments are independent from each other. So, segments can grow or shrink without affecting other segment.
2. If the procedure (or function) in segment 'n' is modified and recompiled, no other segments need to be changed or recompiled.
3. Sharing of procedures and data among various processes simple.
4. Different segments of a single process can be given different kind of protection. One can be read-only, while other can be writable and so on.
5. There is no Internal Fragmentation. Segments are allocated exactly as much memory as required.

Disadvantages Of Non-Contiguous Memory: Segmentation

1. It is still expensive to allocate contiguous free memory to segments. Some algorithm, such as first fit, is required for allocation of a memory.
2. External Fragmentation is possible, which requires memory de-fragmentation or compaction.

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
4.	Getting Started with UNIX	UNIX Architecture UNIX Features Types Of Shell UNIX File System Types Of Files UNIX File & Directory Permissions

Chapter No. 4
UNIX Architecture

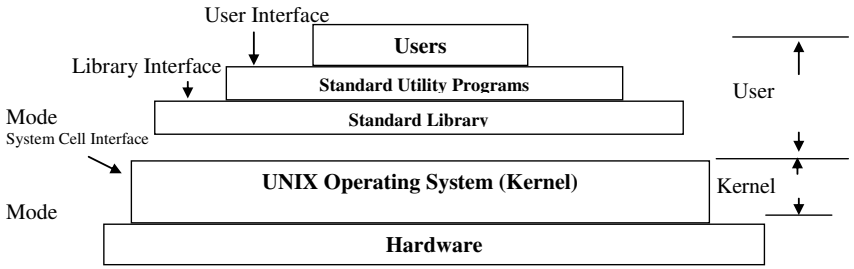
History of UNIX

In 1970, Ken Thompson and Dennis Ritchie at AT&T Bell labs developed UNIX operating system. Earlier this system was known as UNICS (UnIplexed Information and Computer Service), but later it became famous as a UNIX.

In 1973, it was re-written in 'C' programming language (higher-level language). This made UNIX the world's first portable operating system, capable of being easily ported (moved) to any hardware. UNIX became easy to understand and modifiable. This was a major advantage for UNIX. This step led to the wide acceptance of UNIX among users at that time. Today's tremendous popularity of UNIX is also responsible to this step.

The Architecture of UNIX

The various layers depicted in UNIX architecture are shown in below figure and also described below.



1. Hardware

- The bottom layer is the hardware.
- It consists of various physical devices such as CPU, memory, disks, monitor, printer, etc…
- These devices provide various services. For example, printers are used for printout purposes.

2. UNIX Operating System (Kernel):

- The next higher layer is the UNIX operating system.
- It is also called system kernel, or simply, kernel. (It is described next).
- It manages all the underlying hardware.
- It directly interacts with the hardware and provides user programs required services.
- It hides the complex details of hardware also.
- In short, it provides the simple interface between user programs and hardware.
- The main services of an operating system includes process management, memory management, file system management, I/O management etc.

3. Standard Library:

- Above operating system, next layer is for standard library.
- It contains a set of procedures, one procedure per system cell.
- These procedures are written in assembly language and used to invoke various system calls from user programs.

4. Standard Utility Programs:

- In addition to operating system and system call library, all versions of UNIX supply a large number of standard programs.
- Such programs include command processor (Shell), compilers, editors, text processing programs, file manipulation utilities, a variety of commands and so on.

- Such programs make the user tasks simpler. Users interact with them and they, in turn, interact with the operating system to get services from operating system.

5. Users:

- The top most layer is of users.
- User programs come in this layer. They interact with the system either by using library procedures to invoke system calls, or by using utility-program such as shell.

Here, some new terms came in picture such as kernel, shell, and system call. These are described below.

● **Kernel:**

- The kernel is the core of the UNIX operating system.
- UNIX uses microkernel approach, where code from the kernel is moved up in higher layers to keep it as small (thin) as possible.
- Kernel is a program, which is loaded in memory when system is turned on. It stays there and provides various services until the system is turned off.
- Kernel interacts with the hardware directly. When user program needs to use any hardware, it has to use services provided by the kernel. Special functions, called system calls, are used to request kernel. Kernel performs the job on behalf of the user process.
- In addition to providing services to user programs, kernel also provides other services like process management, memory management, file system management and so on.
- In short, kernel manages entire computer system.

● **Shell:**

- The shell is an interface between the user program and the kernel.
- When user logs-in to the system, process for shell starts execution. It terminates when user logs-out from the system.
- Users can directly interact with the shell.
- It works as a command interpreter. It accepts commands from user and translates them into the form, which the kernel can understand easily.
- It is also a programming language. It provides various programming functionalities such as looping, branching and so on.

● **System Call:**

- System calls are special functions.
- They are used to request kernel to provide various services, such as reading from a file stored on hard disk.

- They can be invoked via library procedures, or via command provided by shell, or even directly from C programs in UNIX.
- System calls are similar to user-defined functions. Difference is that they execute in the kernel mode, having fully access to all the hardware; while user defined functions execute in user mode, having no direct access to the hardware.
- Various flavors of UNIX have one thing in common. They all use the same set of system calls provided by POSIX standard. If any operating system is using other system calls, then it will not be a UNIX operating system.

● **Features of UNIX**

Many operating systems are there in market, all operating systems having their own functionality which operating system user should chose that is depends on the task. As being an operating system, UNIX contains all the features that any operating system should have, however, in addition to such common features, UNIX contains some special features, which makes it so much popular. . These features are as given below

1. Files and Process:

- In UNIX, every thing is either a file or a process.
- A file is a collection of data. They are used to store large amount of information permanently. Directories are considered as files. In addition, physical devices are also considered as files in UNIX.
- A process is a program in execution. All commands execute as processes and perform their tasks.

2. A Multi-User System:

-UNIX is a multi user system. (Dos is a single user system.)
- This means that it allows multiple users to work simultaneously on the same system.
- Different users can login from different machines into the same machine by using programs like 'TELNET'.

3. Multi Tasking System.

- UNIX is a multi-tasking system too. (Dos is a single tasking system)
- It allows multiple program to run simultaneously.
- Among simultaneously running processes, one process will be foreground process. User can interact with this process directly. While other processes will be background processes. They execute in background without requiring user interaction.

4. The Building Block Approach:

- UNIX uses the building block approach to perform complex tasks.
- It provides a few hundred commands each of which can perform one simple job to perform complex tasks, such simple commands can be combined using pipes and filters. Thus, the small is beautiful philosophy is implemented here.

5. Portability

- UNIX is written in high-level language, rather than low level assembly language
- This makes it easy to read, understand, change and move to other machine.
- Thus, it provides protability.

6. Consistent format for files

- UNIX does not provide special formats for different file types.
- It treats all files same, and provides consistent format for files in for of simple byte streams.
- It is the responsibility of the application programs to interpret file contents as per requirements.
-

7. Pattern Matching

- UNIX provides very sophisticated pattern matching capabilities.
- There is a set of special character like as '*' that can be used in pattern matching.

8. Programming facility:

- The UNIX shell is also a programming language.
- It supports all the programming features such as variables, control structures, loops so on.
- These features can be used to develop shell programs, called shell scripts. Such programs can be used to control and automate many of the system's functions.
-

9. On-line help

- UNIX provides an on-line help facility for all the commands
- For this purpose, it provides a command named 'man'. By using this command, user can have an instant help on any command.

These features make the UNIX operating system popular among other operating systems.

Types of Shell

The kernel program is usually stored in a file called 'UNIX' whereas the shell program is **un** a file called **'sh'**. For each user working with UNIX at any time different shell programs are running. Thus, at a particular point in time there may be several shells running in memory but only one kernel. This is because; at any instance UNIX is capable of executing only one program as the other programs waits for their turn. And since it's the kernel which executes the program one kernel is sufficient. However, different users at different terminals are trying to seek kernel's attention. And since the user interacts with the kernel through the shell different shells are necessary.

Types of Shells:

Different people implemented the interpreter function of the shell in different ways. This gave rise to various types f shells, **the most prominent of which are outlined below:**

Bourne Shell (Bash)

Among all, Steve Bourne's creation, known after him as the Bourne Shell, is the most popular. Probably that's why it is bundled with every UNIX system. Or perhaps it is the other way round. Because it was bundled with every system it became popular. Whatever the cause and the effect, the fact remains that this is the shell used by many UNIX users.

C Shell (Corn)

This shell is a hit with those who are seriously into UNIX programming. It was created by Bill Joy, then pursuing his graduation at the University of California at Berkeley. It has two advantages over the Bourne Shell.

First, it allows aliasing of commands. That is, you can decide what name you want to call a command by. This proves very useful when lengthy commands which are used time and again are renamed by you. Instead of typing the entire command you can simply use the short alias at the command line.

If you want to save even more on the typing work, C shell has command history feature. This is the second benefit that comes with C Shell. Previously typed commands can be recalled, since the C shell keeps track of all commands issued at the command line. This feature is similar to the one provided by the program DOSKEY in MS-DOS environment.

Korn Shell (Ksh)

If there was any doubt about the cause-effect relationship of the popularity of Bourne Shell and it's inclusion in every package, this adds fuel to it. The not-so-widely-used Korn Shell is very powerful, and is a superset of Bourne Shell. It offers a lot more capabilities and is decidedly more efficient than the other. It was designed to be so by David Korn of AT & T's Bell Labs.

****

Summary

There are 3 types of shells in UNIX:

1) **Bsh**
2) **Korn shell**
3) **C shell**

Explain login commands? 1. passwd 2. logout 3. who 4. who am I 5. clear?

Ans:

1. passwd:

This command is used to change the password for logged in user. If your account doesn't have a password or has one that is already known to others, you should change it immediately. This is done with the **passwd** command.

Syntax

$passwd ↵
Enter login password: *******
New password: *******
Re-enter new password: ********
passwd (SYSTEM): passwd successfully changed for the particular user

passwd expects you to respond three times. First, it prompt for the old password. Next, it checks whether you have entered a valid password, and if you have, it then prompts for the new password.

2. Logout:

This command is used to logout or terminate the session there is other way to logout with exit command and also you can terminate session with the press of **ctrl +d** but these commands may not be work every where to terminate the session the most reliable command is logout.

Syntax:

$logout ↵

3. Who:

This command is used to get information about users (including yourself), you can use several commands. The **who** command reports on users who are presently logged in. The who command normally reports certain information about logged-in users. By using its options, you can specify that it report information about the processes initiated by init, and that it report reboots changes to the system clock, and logoffs. If you invoke who with no option or argument, you get the following output:

Syntax:

$who

Output:

Username	tty00	Date
Username	tty01	Date

Option

-u this option is used to see "time since the last activity" (idle time) and the process ID number for each logged-in user. A "process ID number" is an integer number assigned by UNIX to uniquely identify a given process.

Example

$who –u

Output

Username	tty00	Date	Process No.
Username	tty01	Date	Process No.

-q This option is used to shows login IDs and a count of the logged-in users q stands for quick.

Example

$who –q

Output

User Name	Terminal Name
User Name	Terminal Name

#Users = 2

4. who am i

Sometime we **logged in** from several terminals under different accounts, and we forget that which username or terminal name we are using at that time this command is useful to know username and terminal name.

Syntax:

$who am i

Output

User Name	Terminal Name	Date and Time

5. Clear:

This command is used to **clear** the screen. This command works like **cls** command of dos.

Syntax:

$clear

syntax

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
5.	**UNIX Shell Command**	Connecting UNIX Shell : Telnet **Login Commands:** Passwd Logout Who Who am i Clear **File / Directory Related Command:** ls,cat cd, pwd, mv,cp,ln, rm,rmdir,mkdir,umask,chmod, chown,chgrp,find,pg,more, less,head,tail,wc,touch Redirection & piping **Advance Tools** **Finding Patterns in Files:** grep, fgrep, egrep **Working with columns and fields:** cut , paste, join **Tools for sorting:** sort , uniq **Comparing files :** cmp , comm., diff **Changing Information in Files :** tr , sed **Examining File Contents :** od **Tools for mathematical calculations:** bc , factor **Monitoring Input and Output:**

		tee , script **Tools For Displaying Date and Time:** cal ,date **Communications:** telnet , wall, mtod, write, mail, news, finger **Process Related Command :**ps **Command to run process in background:** nice, kill, at, batch, cron, crontab, wait, sleep **Concept of Mounting a File System:** mount, umount

Chapter No. 5

UNIX File System Overview:

UNIX looks at everything as a file and any UNIX system has thousands of files. If you write a program, you add one more file to the system. When you compile it, you add some more. Files grow rapidly, and if any are not organized properly, you will find it difficult to locate them. Proper file organization requires a directory-based storage system. Just as an office has separate file cabinets to group files of a similar nature, UNIX also organizes its own files in directories and expects you to do that as well.

The file system in UNIX is one of its simple and conceptually clean features. It lets users access other files not belonging to them, and without infringing on security. It also offers an adequate security mechanism so that outsiders are not able to tamper with a file's contents.

The file is a container for storing information. As a first approximation, we can treat it simply as a sequence of characters. If you name a file foo and write three characters a, b and c into it, then foo will contain only the string abc and nothing else.

The Types Of Files:

UNIX treats directories and devices as files as well. A directory is simply a folder where you store file and other directories. All physical devices like the hard disk, memory, CD-ROM, printer and modem are treated as files. The shell is also a file, and

so it the kernel. And if you are wondering how UNIX treats the main memory in your system, it's a file too!

So we have already divided files into three categories:

1. Ordinary (Regular) File:

This is the most common file type in UNIX. All programs you write belong to this type. An ordinary file itself can be divided into to types.

(A) Text File:

A text file contains only printable characters, and you can often view the contents and make sense out of them. All C and Java program sources, shell and perl scripts are text files. A text file contains lines of characters where every line is terminated with the new line character, also known as line feed. When you press enter while inserting text, the LF character is appended to every line. You won't see this character normally, but there is a command which can make it visible.

(B) Binary File:

On the other hand, contains both printable and nonprintable characters that cover the entire ASCII range 0 – 255. Most UNIX commands are binary files, and the object code and executables that you produce by compiling C programs are also binary files. Picture, sound and video files are binary files as well. Displaying such files with a simple eat command produces unreadable output and may even disturb your terminal's settings.

2. Directory File:

A directory contains no data, but keeps some details of the files and subdirectories that it contains. The UNIX file system is organized with a number of directories and subdirectories, and you can also create them as and when you need. You often require to do that to group a set of files pertaining to a specific application. This allows two or more files in separate directories to have the same filename.

A directory file contains an entry for every file and subdirectory that it houses. If you have 20 files in a directory, there will be 20 entries in the directory. Each entry has two components.

- **The file name.**
- **A unique identification number for the file or directory.**

If directory contains an entry for a file named abc, we commonly say that the directory contains the file abc. Though we shall often be using the phrase "contains the

file" rather than "contains the file name", you must not interpret the statement literally. A directory contains the file name and not the file's contents.

3. Device File:

You will also be printing files, installing software from CD-ROM r backing files t tape. All these activities are performed by reading or writing the file representing the device. For instance, you print a file by writing the file representing the printer. When you restore files from tape, you read the file associated with the tape drive. The kernel takes care of this "reflection" by mapping these special files to their respective devices. It is advantageous to treat devices as files as some of the commands used to access an ordinary file also work with device files.

Device file names are generally found inside a single directory structure, ./dev. A device file is indeed special; it's not really a stream of character. In fact, it doesn't contain anything at all. You will soon learn that every file has some attributes that are not stored in the file but elsewhere on the disk. It's the attributes f a device file that entirely govern the operation of the device. The kernel identifies a device from its attributes and then uses them to operate the device.

****

UNIX file & directory permissions:

(A) File permission:

Before we take up file permission, you need to understand the significance of file ownership. When you create a file, your username shows up in the third column of the file's listing; you are the owner of the file. Your group name is seen in the fourth column; your group is the group owner of the file. If you copy someone else's file, you are the owner of the copy. If you can't create files in other users' home directories, it's because those directories are not owned by you.

UNIX has a simple and well-defined system of assigning permissions to files. Issue the ls –l command once again to view the permissions of a few files.

Categories of user		**Types Of Permission**	
User	u	Read	r
Group	g	Write	w
Other	o	Execute	x

```
$ls –l

-rwxr-xr--    1    kumar  metal   20500  may 10 19:21 chap02
-rwxr-xr-x    1    kumar  metal     890  Jan  29 23:17 dateval.sh
-rw-rw-rw-    1    kumar  metal      84  Feb  12 12:30 dept.lst
```

Observe the first column that represents the file permissions. These permissions are also different for the three files. UNIX follows a three-tired file protection system that determines a file's access right. To understand how this system works, let's break up the permissions string of the file chap02 into three groups. The initial – (in the first column) represents an ordinary file and is left of the permissions string:

Each group here represents a category and contains three slots, representing the read, write and execute permission of the file – in that order. r indicates read permission, which means cat can display the file. w indicates write permission; you can edit such a file with an editor. x indicates execute permission; the file can be executed as a program. The – shows the absence of the corresponding permission.

The first group (rwx) has all tree permission. The file is readable, writable and executable by the owner of the file, kumar. But do we know who the owner is? Yes we do. The third column shows kumar as the owner and the first permissions group applies to kumar. You have to log in with the username kumar for these privileges to aplly to you.

The second group (r-x) has a hyphen in the middle slot, which indicates the absence of write permission by the group owner of the file. This group owner is metal, and all users belonging to the metal group have read and execute permissions only.

The third group (r--) has the write and execute bits absent. This set of permissions is applicable to others, i.e., those who are neither the owner kumar nor belong to the metal group. This category (others) is often referred to as the world. This file is not world-writable.

You can set different permissions for the three categories of users – owner, group and others. It's important that you understand them because a little learning here can be a dangerous thing. A faulty file permission is a sure recipe for disaster.

(B) Directory Permission:

Directory also has their own permissions and the significance of these permissions differ a great deal from those of ordinary files. You may not have expected this, but be aware that read and write access to an ordinary file are also influenced by the permission of the directory housing them. It's possible that a file can't be accessed even though it has read permission, and can be removed even when it' write-protected. In fact, it's very easy to make it behave that way.

If the default directory permission are not altered, the chmod theory still applies. However, if they are changed, unusual things can happen. Though directory permissions are taken up later, it's worthwhile to know what the default permissions are on your system:

Categories of user		**Types Of Permission**	
User	u	Read	r
Group	g	Write	w
Other	o	Execute	x

```
$mkdir c_progs

$ls –ld c_progs

drwxr-xr-x   2   kumar metal 512  may  9   09:57  c_progs
```

First character d displays that c_progs is directory, second, third and fourth character rwx is showing that user is having all permission read, write and execute, fifth, sixth and seventh character r-x showing that group is having only read and execute permission, eighth, ninth and tenth character r-x showing that others having read and execute permission.

The default permissions of a directory on this system are rwxr-xr-x (or 755); that is what they should be. A directory must never be writable by group and others. If you find that your files are being tampered with even though they appear to be protected, check up the directory permissions.

Related Command

(1) ls:

This command is used see the list of files and directory at current location or specified path. This command supports us to see the list of file and directory.

Syntex:

$ls [-Option] [Path]

Options:

–x	Multicolumn output
-F	Marks executable with * directories with / and symbolic links with @
-a	Show all filenames beginning with a dot including . and ..
-R	Recursive list
-r	Sorts filenames in reverse order (ASCII collating sequence by default)
-1	One filename in each line
-l	Long listing in ASCII collating sequence showing seven attributes of a file
-d	List only directory
-t	Sorts filenames by last modification time

-lt	Sorts filenames listing by last modification time
-u	Sorts filenames by last access time
-lu	Sort by ASCII collating sequence but listing shows last access time
-lut	As above but sorted by last access time
-i	Display inode number

(2) cat

This command is use to displays files contents, creates new file, or appends data to an existing file.

Syntax 1 To display the content of file

$cat File Name

Example

$cat Example1.txt

Syntax 2 To create a new file

$cat > File Name
Enter the file content
Ctrl + d

Example

$cat > Example2.txt
Enter the file content
Ctrl+d

Syntax 3 To append existing file

$cat >> File Name
Enter the file content want to append
Ctrl+d

Example

$cat >>Example1.txt
Enter the file content want to append
Ctrl+d

(3) cd (Change Directory)

This command is used to change one directory forward, backward and supporting to go to root directory directly.

Syntax

cd	Change working directory to home directory.
cd ..	Change working directory to parent directory.
cd Path	Change working directory to specified directory.
cd /	Change working directory to root directory.

Example

If current directory /home/abc and user home directory is /home/guest then

$cd will take you to home directory

now current directory is /home/guest

$cd .. will take you to home directory

now current directory is /home

$cd /home/abc

now current directory is /home/abc

$cd /

now current directory is / (Root)

(4) pwd: (Path for working directory)

This command is used to see path for working directory in UNIX you can not see the prompt with path at that time if we want to know the what is our current path. The pwd command supports to know path for working directory.

Syntax

$pwd

Example

If current directory is /home guest then

$pwd will show you /home/guest

(5) mv command (Renaming Files and Moving a file from one place to another place)

The mv command renames files. It has two distinct functions:

- It renames a file or directory.
- It moves a group of files to a different directory.

To rename a file or directory

Syntax

$mv Old File Name New File Name

mv doesn't create a copy of the file; it merely renames it. No additional space is consumed on disk during renaming. To rename the file abc.txt to xyz.txt you can write following command.

$mv abc.txt xyz.txt

If the destination file doesn't exist, it will be created. For the above example, mv simply replaces the filename in the existing directory entry with the new name. By default, mv doesn't prompt for overwriting the destination file if it exists. So be careful.

To move a file from one place to another

Syntax

$mv file name Destination folder or path of Destination folder

Example

To move a.txt file from root to home/abc directory then following command is used from root.

$mv a.txt /home/abc
Above command will move file named a.txt from root to /home/abc

(6) cp (Copying a file)

This command is used to copy a file.

Syntax

$cp [option] Source file path or file Target path

Option
-i Asks before overwriting a destination file
-r It copies directory structure.

Example

To copy a a.txt file from /home/abc to /home/guest following command is used.

$cp /home/abc/a.txt /home/guest

Above command will copy a.txt file from /home/abc to /home/guest directory

(7) ln (Link a file)

This command is used to link a file. Links are used for sharing of files from two different locations in directory structure. There are two types of link 1) soft link 2) hard link.

Soft Link:- It links file name if source is deleted then link will not be found

Hard Link:- It links physical storage area of the file if source file is deleted still it will target the physical part of the file and data can be found.

Syntax:

$ln [Option] Source File Name Link File Name

Option
-s Creates soft link if this option is not specified with ln command then hard link will be created.

Example

File Name abc.txt following is the content of the file
Abc abc abc abc abc abc

$ ln –s abc.txt xyz.txt

Soft link will be created if abc.txt file will be deleted then user will not be able to see the content of xyz.txt because it is related to abc.txt file name.

$ ln abc.txt xyz.txt

Hard link will be created if abc.txt file will be deleted still user will be able to work will the content of abc.txt file.

(8) rm (Remove File)

This command is used to remove (delete) a file.

Syntax

$rm [Option] [Path]File Name

Options

-i	Ask before removing a file
-r	Performs recursive deletion in directory
-f	Removes write protected file also

Example

To remove a file abc.txt from /home/guest current directory is root then following command will be written.

$rm /home/guest/abc.txt

Above command will remove abc.txt file.

(9) rmdir: (Remove directory)

This command is used to remove directory form current path and specified path.

Syntex:

$rmdir [Path] Directory Name

This command is useful to remove empty directory the prime condition of this command is this that whatever the directory is, you are going to remove it that must be empty else message will display on the screen that directory is not empty.

Error Message

rmdir: dirname: cannot remove [Directory not empty]

(10) mkdir: (Make Directory)

This command is used to create a directory on current path or specified path.

Syntax:

$mkdir Directory Name

Some time you are giving right command still directory is not going to be created for following reasons.

Error Message

Mkdir: directory name: [file exists]

(1) The directory name is given with mkdir is already exist
(2) There may be a an ordinary file by that name in the current directory
(3) The permission set for the current directory don't permit

(11) Umask

This command is used to set default file permission. Whatever the file is created by the owner at that time which permission is default given to user, group and other that default permission will be set using Umask command. Umask stands for user file creation mask. The term mask implying which permissions to mask or hide. The umask value tells UNIX which of the three permission are to be denied rather than granted. The current value of umask can be easily determined by just typing umask.

To see the default permissions

Syntax

$umask

$umask
0022

Here first 0 indicates that for owner no permission is denied, Second 0 indicates that for user no permission is denied, third number 2 indicates that write permission is denied to group and fourth number two is also indicating that write permission denied to others.

Changing default permission using umask

$umask 111

This would see to it that here onwards any new file that you create would have the permissions 555 and any directory that you create would have the permission 666.

(12) Chmod (Changing File and Directory Permissions)

A file or directory is created with a default set of permissions, and this default is determined by a simple setting called umask. Generally, the default setting write-protecccts a file from all except the user (new name for owner), though all user may have read access. However, this may not be so on your system. To know your system's default, create a file xstart:

$cat > xstart

$ ls –l

-rw-r--r-- 1 kumar metal 1906 Sep 5 23:38 xstart

Syntax Relative Permission

$chmod <<User Category>> <<Operation>> <<Permission>> <<File Name>>

User Category : User, Group, Other
Operation : Assign Or remove a permission using + and –
Permission : Read, Write, Execute

Example

To assign execute permission to the user of the file xstart, we need to write following command

$chmod u+x xstart

$ls –l

-rwxr--r-- 1 kumar metal 1906 May 10 20:30 xstart

To use multiple characters to represent the user category (ugo).

$chmod ugo+x xstart or $chmod a+x xstart or $chmod +x xstart

$ls –l

-rwxr-xr-x 1 kumar metal 1906 May 10 20:30 xstart

To use more then one file at a time

$chmod u+x note note1 note2 note 3

To grant and revoke a permission at a time

$chmod a-x,go+r xstart

Abbreviations Used By chmod

Category	Operation	Permission
u user	+ Assigns Permission	r Read Permission
g group	- Remove Permission	w Write Permission
o Other	= Assigns absolute permission	x Execute permission
a All(ugo)		

Syntax Absolute Permissions

$chmod Combination of Decimal Code

Combination of Decimal Code

- Read Permission - 4 (Octal 100)
- Write Permission - 2 (Octal 010)
- Execute Permission - 1 (Octal 001)

Binary	Octal	Permissions	Significance
000	0	---	No permission
001	1	--x	Executable Only
010	2	-w-	Writable Only
011	3	-wx	Writable and Executable
100	4	r--	Readable Only
101	5	r-x	Readable and Executable
110	6	rw-	Readable and Writable
111	7	rwx	Readable, Writable and Executable

Example 1

$chmod 666 xstart;

$ls –l xstart;

-rw-rw-rw- 1 kumar metal 1906 May 10 20:30 xstart

The 6 indicates read and write permission (4 + 2). To restore the original permissions to the file, you need to remove the write permission (2) from group and others:

Example 2

$chmod 644 xstart

$ls –l xstart

-rw-r--r-- 1 kumar metal 1906 May 10 20:30 xstart

To assign all permissions to the owner, read and write permissions to the group, and only execute permission to the other, use following command.

Example 3

$chmod 761 xstart

-rwxrw---x 1 kumar metal 1906 May 10 20:30 xstart

(13) Chown: (Change Owner)

This command is used to change the owner of file. It transfers ownership of a file to a user, and it seems that it can optionally change the group as well. The command requires the user-id (UID) of the recipient, followed by one or more filenames. Changing ownership requires super user permission, so let's first change our status to that of super user with the su command.

$su
Password:********
#_

After the password is successfully entered, su returns a # prompt, the same prompt used by root. su lets us acquire super user status if we know the root password. To now renounce the ownership of the file note to sharma, use chown in the following way:

Syntax:

Chown < New Owner Name> file(s)

Example

#ls –s note

-rwxr----x 1 kumar metal 347 May 10 20:30 Note

#chown sharma note;

#ls –l

-rwxr----x 1 sharma metal 347 May 10 20:30 Note

#exit

$_

Once ownership of the file has been given away to sharma, the permissions that previously applied to kumar now apply to sharma. Thus, kumar can no longer edit note since there's no write privilege for group and others. He can't get back the ownership either. But he can copy the file to his own directory, in which case he becomes the owner of the copy.

(14) chgrp: (Changing Group Owner):

By default, the group owner of a file is the group to which the owner belongs. **The chgrp (change group) command changes a file's group owner**. A user can change the group owner of a file, but only to a group to which she also belongs. Yes, a user can belong to more than one group.

chgrp shares a similar syntax with chown. In the following example, kumar changes the group ownership of dept.lst to dba (No super user permission required):

Syntax:

chgrp <<New Group Name>> <<File Name>>

$ls –l dept.lst

-rw-r--r-- 1 kumar metal 139 Jun 8 16:43 dept.lst

$chgrp dba dept.lst

-rw-r--r-- 1 kumar dba 139 Jun 8 16:43 dept.lst

If kumar is also a member of the dba group then above command will work, if he is not a member of the dba group then super user can make the command work. Note that kumar can reverse this action and restore the previous group ownership (to metal) because he is still owner of the file and consequently retains all right related to it.

(15) Find (Locating Files)

This command is used to find a particular file.

Syntax

$find [Path]/File Name

Example 1

$find abc.txt

$find /home/guest/abc.txt

User can use wild card character with file name * means all character and ? means 1 character

Example 2

To find the file which having extension name .txt following command is used

$find *.txt

Example 3

To find the file which 2nd character is d then following command is used

$find ?d*.*

(16) Pg commands

This command is used to see the information of the document page wise.

Syntax

$pg Starting Line Number No of Lines Per Page Option File Name

Example

$pg +10 –15 –p "Page No. %d" –s abc.txt

Explanation

Above command starts displaying the contents of abc.txt file, 15 lines at a time from 10th line onwards. At the end of each displayed page a prompt comes which displays the page number on view. This prompt overrides the default ':' prompt of the pg command. The –s option ensures that the prompt is displayed in reverse video.

(17) More Command (Paging out)

This command is used to see the information of the document page wise.

Syntax

$more File Name **Press q to exit**

Example

$more abc.txt

Explanation

You will see the contents of abc.txt on the screen, one page at a time. At the bottom of the screen, you'll also see the filename and percentage of the file that has been viewed:

----More----(17%)

More has a couple of internal commands that don't show up on the screen when you invoke them. q, the command used to exit more, is an internal command.

Navigation of More Command

Irrespective of version, more uses the spacebar to scroll forward a page at a time. You can also scroll by small and large increment of lines or screens. To move forward one page, use

f or spacebar	To move page forward
b	To move back one page
Enter or j	One line forward

k	One line back
p or 1G	Beginning of file
G	End of file
q	Quit

(18) less (Paging Out)

This command is used to see the information of the document page wise. This command work just like more command and it is older version.

Syntax

$less File Name **Press q to exit**

Example

$less abc.txt

Explanation

You will see the contents of abc.txt on the screen, one page at a time. At the bottom of the screen, you'll also see the filename and : that is ready to accept the navigation command those are described below.

Abc.txt :

Less has a couple of internal commands that don't show up on the screen when you invoke them. q, the command used to exit less, is an internal command.

Navigation of less Command

Irrespective of version, less uses the spacebar to scroll forward a page at a time. You can also scroll by small and large increment of lines or screens. To move forward one page, use

f or spacebar	To move page forward
b	To move back one page
Enter or j	One line forward
k	One line back

p or 1G	Beginning of file
G	End of file
q	Quit

At the end of file user has to give q to quit from less command.

(19) head

We are using cat command to view the content of files; however, if the file is bigger than 24 lines then the matter would naturally scroll off the screen. If we want to stop the scrolling we can do so by hitting the pause key and resume it by hitting any other key. This of course needs a bit of a practice otherwise the matter scrolls off the screen before you can reach for the pause key. To exercise a tighter control over the way we can view files UNIX proves several utilities. Out of these the head commands help in viewing lines at the beginning of the file respectively.

Syntax:

$head –Line Number File Name

Unless otherwise specified the head command assumes that you want to display first 10 lines in the file. Should you decide to view first fifteen lines you simply have to say.

$head -15 myfile

Here with head command we can specify the number of lines if we decide to override this default value.

The disadvantage of head is that they cannot display a range of lines. Moreover, what is displayed is final. That is, if we have displayed the first 50 lines in a file we cannot move back and view say the 10th line. UNIX provides two commands which offer more flexibility in viewing files.

(20) tail

We are using cat command to view the content of files; however, if the file is bigger than 24 lines then the matter would naturally scroll off the screen. If we want to stop the scrolling we can do so by hitting the pause key and resume it by hitting any other key. This of course needs a bit of a practice otherwise the matter scrolls off the screen before you can reach for the pause key. To exercise a tighter control over the way we can view files UNIX proves several utilities. Out of these the tail commands help in viewing lines at the ending of the file respectively.

Syntax:

$tail –Line Number File Name

Unless otherwise specified the tail command assumes that you want to display last 10 lines in the file. Should you decide to view last fifteen lines you simply have to say.

$tail -15 myfile

Here with tail command we can specify the number of lines if we decide to override this default value.

The disadvantage of tail is that they cannot display a range of lines. Moreover, what is displayed is final. That is, if we have displayed the last 50 lines in a file we cannot move back and view say the 10th line. UNIX provides two commands which offer more flexibility in viewing files. These commands are pg and more.

(21) wc (Word Count)

This command is used to count number of characters, words and lines in an given input file. If file is not given, it takes input from standard input.

Syntax

$wc [Option] File Name

Options

-l	Counts number of lines
-w	Counts number of words
-c	Counts number of characters

Example

A file abc.txt having following data

1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10

$ wc abc.txt

Output

2 20 42 abc.txt

Note: Spacebar and Enter also known as on character

(22) touch (Changing the time stamp)

This command is used to change the time stamp.

Syntax

$touch MonthDateHoursMinutes File Name

Example

$ls –l

-rw-r--r-- 1 kumar metal 870 Mar 16 14:30 emp.lst

$touch 02281030

-rw-r--r-- 1 kumar metal 870 Feb 16 16:50 emp.lst

****

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
6.	**Text Editing With vi Editor**	Introduction of vi editor Modes in vi Switching mode in vi Cursor movement Screen control commands Entering text, cut, copy, paste in vi editor

Chapter No. 6
VI Editor

✪ Introduction to VI editor

No matter what work you do with the UNIX system, you will eventually write some C programs or shell scripts. You may have to edit some of the system files at times. For all this you must learn to use an editor, and UNIX providers a very versatile one VI editor, VI editor is a screen editor, where a portion of the file is displayed on the terminal screen, and the cursor can be moved around the screen to indicate where you want to make changes. You can select which part of the file you want to have displayed. Screen editors are also called display editors, or visual editors. VI is one of the more popular screen editors that run on the UNIX system.

✪ Invoking VI

It will put filename into a buffer, and display the file on the screen. If the file is larger than the screen can display, the screen will act as a window into the file. At the beginning of a session, the screen will display the first part of the file. If filename does not exist, VI will create it.

✪ Modes in VI editor

One of the most important aspects to remember about **vi** is that most of the commands fall into one of three modes:

(1) **Command mode:**

This is the default mode of VI editor. This mode is used to give some command for navigation, edition and copy or cut. This is the base mode of VI editor if user want to switch this mode to **insert or input mode then user has to press i or I or a or A.** If user wants to switch execute mode from this mode then user has to **press:**

(2) **Input or Insert Mode:**

This mode is used to enter data in VI editor. If user wants to insert some text to the editor then first user has to select this mode. **This mode will be selected from command mode pressing I or i or a or A.**

(3) **Execute Mode:**

This mode is used to save or quit from VI editor. What ever the changes user has done in file using VI editor if user wants to save, find any particular string then using this mode user can save or find the string and also whenever user wants to quit from the VI editor this mode will supports user to quit from the VI editor. **To switch to this mode user has to press: or / on command mode**

✪ Switching mode in VI

While you start **VI** editor at that time you will be at vi mode that will ready to accept defined command on that particular key but not any input. If you want to input text into the file you will have to go to input mode for that you will have to **press "i".**

If you will press **"i" at VI** mode you will be at input mode here you can input any data into the file from this mode if you want to switch to vi mode then you will have to press **"Esc"** key. If you will press **"Esc"** key at input mode you will be able to switch to vi mode.

If you want to switch to command mode from the VI mode then you will have to press **":" or "/"** that will support you to switch you from VI mode to command mode.

If you want to switch to command mode from the input mode then you will have to first come to the VI mode then you will be able to switch to command mode. To switch from input mode to command mode then you first will have to press **"Esc"** key and then after you will have to press **":" or "/"** key that will support you to switch from input mode to command mode via **VI** mode.

✪ Cursor movement in VI editor

Whatever the command you are giving that will work on VI mode only.

To move around in the file, use the arrow keys: the up arrow will move the cursor up one line, the down arrow down one line. The right arrow will cause the cursor to move one position to the right. From the end of a line, it will go to the beginning of the next line.

For terminals with no functional arrow keys, four keys will move the cursor around:

h **left arrow**
j **down arrow**
k **up arrow**
l **right arrow**

✪ Screen control commands

Whatever the command you are giving that will work on **vi** mode only.

ctrl + f
This will move the user forward one screen in the file.

ctrl+b
This will move the user backward one screen in the file.

G
This will move the cursor to the end of the file. Note, again that vi, like any other UNIX utility, is always case sensitive.

num
This will bring the cursor to line defined line number.

✪ Entering, Editing, Copying data into vi editor

Whatever the command you are giving that will work on vi mode only.

x
That will delete on character.

dd

That will delete lines beginning at the current line.

yy
That will copy lines beginning at the current line.

p
That will paste copied line.

u
This is very useful command. It cancels the effect of the previously executed command.

q
When in vi, typing will bring the user into command mode. As the : is typed, it will be displayed n the last line of the screen, and vi will wait for a command to be typed. q is such a command. This will exit vi, if no changes have been made since the last write-to-file command :q! will exit, even if the buffer has not been written to the file.

w
Will cuase the contents of the buffer to be written to the file :wq will write the buffer to the file, and exit t the calling shell.

i
This will support you to switch from VI mode to input mode.

:
This will support you to switch from VI mode to command mode.

esc
This will support you to switch from input mode to VI mode.

/ This will support you to find any particular pattern from the file

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
7.	**Shell Programming**	Shell Keywords Shell Variables ->System Variables ->User Variables ->Decision Statements ->Loop ->test command ->Logical Operators

Chapter No. 7
Concept of Redirection and Piping:

Most UNIX commands expect input to come from a file(s), and produce output to anther file(s). "One of these files is the standard input, and is the place from which a program expects to read its input. Another is called the standard output, and it is the file to which the program writes its results. The third file is the diagnostic output (also called Standard error), and it is a file to which program writes any error responses. Generally, the standard input is taken to be the keyboard (input is typed by the user), the standard output is the terminal screen.

Standard Input, Output, Error

Many UNIX commands get input from what is called standard input and send their output to standard output (often abbreviated as standard input and standard output). Your shell sets things up s that standard input is your keyboard, and standard output is the screen.

Here's an example using the cat command. Normally, cat reads data from all of the files specified by the command line, and sends this data directly to standard output. Therefore, using the command:

/home/bca/cat history master

Display the contents of the file history followed by masters.

However, if you don't specify a filename, cat reads data from standard input and sends it back to standard output. Here's an example

/home/bca/cat > papers

Hello there
Hello there
Bye
Bye
Ctrl +d

Each line that you type is immediately echoed back by cat. When reading form standard input, you indicate the input is "finished" by sending an EOT (end of text) signal, in general, generated by pressing Ctrl + D.

(1) Redirection Output:

Now, let's say that you want to send the output of cat to a file, to save our shopping list on disk. **The shell lets you redirect standard output to a file name, by using the ">" symbol.** Here's how it works:

$cat amg.txt>output.txt

The > operator causes a new file to be created. If you had already created a file named output.txt, it would be deleted and replaced with the new data. **If you wanted to add the new data to the old output, you could use the >> operator.** For example.

$cat amg.txt>>output.txt

This will add content of amg.txt to output.txt file both file must be exists at specified path.

(2) Redirection Input:

There are two possible sources of input for UNIX commands. Programs such as ls and find get their input from the command line in the form of options and filenames. Other programs, such as cat, can get their data from the standard input as well as from the command line. Try the cat command with no options on the command line:

$cat

There is no response. Because n files are specified with the command, cat waits to get its input from your keyboard, the standard input file. The program will accept

input lines from the keyboard until it sees a line which begins with **Ctrl+D**, which is the end of file signal for standard input.

To redirect the standard input, you use the < operator. For example, if you wanted cat to get its input from output.txt you could use the command.

$cat<output.txt

The difference between this command and

$cat output.txt

is a subtitle one. In filenames provided as options to a command, you can use filename substitution. When redirecting input, you must use the name of an existing file or device.

(3) Standard Error:

Whenever the command is given wrongly at prompt there must be an error occurs to store the error message in file because we can see in windows base application that if any system error occurs that generates the log file for that particular error. In that way in UNIX such a file can be generated as standard error.

Example

$ cat > output.txt 2>errorfile

A message cannot open uoutput.txt will be written in errorfile. You can see such message with the help of cat errorfile.

Concept of Piping:

Suppose that you wanted a directory listing that was sorted by the mode file type plus permissions. To accomplish this, you might redirect the output from ls to a data file and then sort that data file. For example.

$ls –l>tempfile

$sort<tempfile

```
-rw-rw-r-    1    marsha    adept   1024    Jan 20 14:14 Lines.dat
-rw-rw-r-    1    marsha    adept   3072    Jan 20 14:14 Lines.idx
-rw-rw-r-    1    marsha    adept   256     Jan 20 14:14 Pages.dat
```


Although you get the result that you wanted, there are three drawback to this method:

You might end up with a lot of temporary file in your directory. You would have to go back and remove them. The sort program does not begin its work until the first command is complete. This is not too significant with the small amount of data used in this example, but it can make a considerable difference with larger files. The final output contains the name of your temp file, which might not be what you had in mind.

Fortunately, there is a better way.

The pipe symbol (|) causes the standard output of the program on the left side of the pipe to be passed directly to the standard input of the program on the right side of the pipe symbol. Therefore, to get the same results as before, you can use the pipe symbol. For example

$ls –l|sort

```
-rw-rw-r-    1    marsha    adept   1024    Jan 20 14:14 Lines.dat
-rw-rw-r-    1    marsha    adept   3072    Jan 20 14:14 Lines.idx
-rw-rw-r-    1    marsha    adept   256     Jan 20 14:14 Pages.dat
```

To connect two or more operation within the same stream at that time pipe sign is used to perform such a operation in UNIX.

Finding patterns (Regular expression, gerp, egrep, fgrep)

UNIX was originally written for people using slow and clumsy teletype terminals. The fewer the number of characters typed the better. For that reason, many UNIX commands are very short two or three letters. The options, called flags, are also very short usually one character.

The syntax for some commands will be described in this session along with some of the options or flags. However, many flags for many commands will not be mentioned. It is the user's responsibility to verify which flags/options are available on their machine. Short commands are usually not very mnemonic. But there are always tricks to help remember their meaning.

⭘Regular Expression

Regular expression as used in VI, grep, etc. Regular expressions are formed from single characters, concatenation of regular expressions, choice between two regular expressions, and zero or more occurrences of a regular expression.

In UNIX, regular expressions are defined using the following Meta symbols and extensions to the regular expression notation.

.	Match any single character.
[xyz]	Match any character enclosed in the brackets.
[A-B]	Match any character in the range.
[^xyz]	Complement: match any character not enclosed in the brackets.
re*	The Kleene star: match zero or more occurrences of the Re.
^re	RE match must occur at the beginning of a line.
re$	RE match must occur at the end of line.
\\	Represent the backslash

✪grep Command

The grep command is used to read from the specified file on the command line.

Syntex:

$grep [option] RE [file(s)]

option

-b	Display, at the beginning of the output line, the number of the block in which the regular expression was found. This can be helpful in locating block numbers by context. (The first block is block zero)
-c	Print the number of lines that contain the pattern, that is, the number of matching lines.
-h	Prevent the name of the file that contains the matching line from being displayed at the beginning of that line. Note:- When searching multiple files, grep normally reports not only the matching line but also the name of the file that contains it.
-i	Ignre distinctions between uppercase and lowercase during comparisons.
-l	Print one time the name of each file that contains lines that match the pattern-regardless of the actual number of matching lines in each file-on separate lines of the screen.
-n	Precede each matching line by its line number in the file.
-s	Suppress error messages about nonexistent or unreadable files.
-v	Print all lines except those that contain the pattern. This reverses the logic of the search.

Examples:

1. $grep –i 'you' pax corn

Here the pax and corn are file name from where you want to find 'you' or 'YOU' because option i is indicating that ignoring case.

2. grep –c 'you' pax corn

Here you want to see that 'you' how many times repeated in the files. The output of above command will be this.

Output

pax : 1
corn: 5

3. grep –h 'you' pax corn

Above command will shows the output for the matching lines without specifying the files from which they came.

4. grep –iv 'you' pax corn

Above command specifies output of "every line in pax and corn that does not have 'you' or 'YOU'.

5. grep '^The' pax cron

The above command finds lines that begin with The.

6. $grep 'n$' pax corn

Above command finds lines that end with n.

✪egrep Command

egrp uses full regular expression in the pattern string. The syntax of egrep is the same as that for grep.

Syntex:

$egrep [option] RE [files]

The egrep command uses the samre regular expressions as the grep command, except for \(and\), and includes the following additional patters:

RE+	Matches on or more occurrence(s) or RE. (Contrast this with grep's RE* pattern, which matches zero or more occurrences of RE).
RE?	Matches zero or one occurrence of RE.
RE1|RE2	Matches either RE1 or RE2. The | acts as a logical OR operator.
(RE)	Groups multiple regular expressions.

The egrep command accepts the same command-line options as grep as well as the following additional command-line options:

-e Search for a special expression (that is, a regular expression that begins with a-)
-f Put the regular expression into file.

Example:

1. $egrep '[A-Z][A-Z]+' pax corn

The above command finds two or more consecutive uppercase letters.

2. $egrep 'DOS|SCO' pax corn

The above command finds each line that contains either DOS or SCO.

3. $egrep 'n(e|o)w' corn

The above command finds either new or now.

✪fgrep Command (Multiple string searching)

The fgrep command searches a file for a character string and prints all lines that contain the string. Unlike grep and egrep, fgrep interprets each character in the search string as a literal character, because fgrep has no meta characters.

Syntex:

$fgrep [options] string [files]

The options you use with the fgrep command are exactly the same as those that you use for egrep, with the addition of –x, which prints only the lines that are matched in their entirety.

Working with column and fields using cut and paste command

(1) cut command

Cut command is used to cut or pick up a given number of character or fields from the specified file. This command is support you to see specified filed. Like you have large database information but out of them you want to see some selected fields than cut command is useful in UNIX.

Syntax:

$cut [Main Options] Field or Character List [Options] File Name

Main Options:

-f This option is used to specify field list separated by TAB
-b This option is used to specify bytes list single character taken as bytes
-c This option is used to specify character list single character taken as character

It is must that you will have to use one main option with cut command.

Options:

-d This option is used to specify the character as field separator
-s This option is used to skip the lines which does not have field separator

Example 1:

```
$cat >> abc.txt
1.  one  two  three  four  five  six
2.  one  two  three  four  five  six
3.  one  two  three  four  five  six
```

(1) **$cut –b 1,2 abc.txt** (This will show you given byte)

```
1.
2.
3.
```

(2) **$cut –b 1-4 abc.txt** (This will show you range 1 to 4 byte)

```
1.  o
2.  o
3.  o
```

(3) **$cut –c 1,2 abc.txt** (This will show you given character)

```
1.
2.
3.
```

(4) **$cut –c 1-4 abc.txt** (This will show you range 1 to 4 character)
4. o
5. o
6. o

Example 2:

$cat >>abc1.txt
1:one:two:three:four:five:six
2:one:two:three:four:five:six
3:one:two:three:four:five:six

$cut –f 1-3 –d":" abc1.txt
1:one:two
2:one:two
3:one:two

Example 3:

(1) **$cat >> abc.txt**
1. one two three four five six
2. one two three four five six
3. one two three four five six
abcdefghijklmnopqrstuvwxyz

(2) **$cut –f 1-4 abc.txt**
$cat >> abc.txt
1. one two three
2. one two three
3. one two three
abcdefghijklmnopqrstuvwxyz

(3) **$cut –f 1-4 –s abc.txt**
1. one two three
2. one two three
3. one two three

Paste Command

This command is used to merges lines of files. This command prints lines consisting of sequentially corresponding lines of each given files, separated by tabs, terminated by a new line. If no files are given, the standard input is used. A filename of -means standard input.

Syntax:

$Paste [Option] File Name1 File Name2

Options:

-s Serial Paste the lines of one file at a time rather then one line from each file.

-d Delimiters tow files

Example 1

$cat >> 1.txt
1. aaa bbb ccc
2. aaa bbb ccc
3. aaa bbb ccc

$cat >> 2.txt
A) AAA BBB CCC
B) AAA BBB CCC
C) AAA BBB CCC

(1) **$paste 1.txt 2.txt**
1. aaa bbb ccc A) AAA BBB CCC
2. aaa bbb ccc B) AAA BBB CCC
3. aaa bbb ccc C) AAA BBB CCC

(2) **$paste -d"|" 1.txt 2.txt**
1. aaa bbb ccc |A) AAA BBB CCC
2. aaa bbb ccc |B) AAA BBB CCC
3. aaa bbb ccc |C) AAA BBB CCC

(3) **$paste -s 1.txt 2.txt**
1. aaa bbb ccc 2. aaa bbb ccc 3. aaa bbb ccc
A) AAA BBB CCC B) AAA BBB CCC C) AAA BBB CCC

(4) **$paste -s -d"|" 1.txt 2.txt**
1. aaa bbb ccc |2. aaa bbb ccc |3. aaa bbb ccc
A) AAA BBB CCC |B) AAA BBB CCC |C) AAA BBB CCC

Join:

join joins the contents of two files, combining common fields of data.

join [OPTION]... FILE1 FILE2

Options

-a FILENUM	also print unpairable lines from file FILENUM, where FILENUM is 1 or 2, corresponding to FILE1 or FILE2
-e EMPTY	replace missing input fields with EMPTY
-i, --ignore-case	ignore differences in case when comparing fields
-j FIELD	equivalent to '-1 FIELD -2 FIELD'
-o FORMAT	obey FORMAT while constructing output line
-t CHAR	use CHAR as input and output field separator
-v FILENUM	like -a FILENUM, but suppress joined output lines
-1 FIELD	join on this FIELD of file 1
-2 FIELD	join on this FIELD of file 2
--check-order	check that the input is correctly sorted, even if all input lines are pairable
--nocheck-order	do not check that the input is correctly sorted
--header	treat the first line in each file as field headers, print them without trying to pair them
--help	display command help and exit
--version	display version information and exit

Tools for sorting using sort and uniq

Sort:

This command is used to sort file with field number. This command is used to arrange your file alphabetically.

Syntax:

$sort [Options] File Name

Options

+Field Number	This option is used to define field on which you want to sort your file
-r	This option is used to sorting data in reverse order.

Note: Field number always start with 0

Example:

```
$cat >> auto.txt
1   h   j   k
4   a   x   d
3   d   l   i
2   j   o   m
```

(1)
```
$sort auto.txt
1   h   j   k
2   j   o   m
3   d   l   i
4   a   x   d
```

(2)
```
$sort +1 auto
4   a   x   d
3   d   l   i
1   h   j   k
2   j   o   m
```

(3)
```
$sort -r    auto
4   a   x   d
3   d   l   i
2   j   o   m
1   h   j   k
```

(4) **$sort -r +1 auto**

2	j	o	m
1	h	j	k
4	a	x	d
3	d	l	i

Uniq Command

This command is used to finding out the duplicate and uniq entry from any particular files, Sometimes we merge the files at that time duplication is the big problem to prevent duplication of the records uniq command will support us to finding out duplication and uniq rows.

Syntax:

$uniq [Options] Filenames

Options:

- -c This will count any particular rows that how many times it is repeated in file
- -u This will remove the duplicate rows from the file and will print only non repeated rows
- -d This will finding out the duplicate rows and only print duplicate rows not uniq rows

Example:

$cat >> new.txt

01	accounts	6213
01	accounts	6213
02	admin	5423
03	marketing	6521
03	marketing	6521
03	marketing	6521
04	personnel	2365
05	production	9876
06	sales	1006

(1) **$uniq new.txt**

01	accounts	6213
02	admin	5423
03	marketing	6521
04	personnel	2365

05	production	9876
06	sales	1006

(2) **$uniq -c new.txt**

2	01	accounts	6213
1	02	admin	5423
3	03	marketing	6521
1	04	personnel	2365
1	05	production	9876
1	06	sales	1006

(3) **$uniq -u new.txt**

02	admin	5423
04	personnel	2365
05	production	9876
06	sales	1006

(4) **$uniq -d new.txt**

01	accounts	6213
03	marketing	6521

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
8.	Getting Started with Linux	->History of Linux ->Features of Linux ->Installation and ->configuration of Linux

Chapter No. 8

Comparing Files:

You have seen UNIX commands that works with a single file at a time. However, often a user must compare two files and determine whether they are different, and if so, just what the differences are UNIX provides commands that can help:

The **cmp** command compares two files and then simple reports the character number and line number where they differ.

The **comm** command compares two files and the report in three column.

The **diff** command compares two files and tells you exactly where the file differ and what you must do to make them agree.

⦿cmp command

The simplest command for **comparing two files,** cmp, simply tells you whether the files are different or not, if they are different, it tells you where in the file it sptted the first difference, if you are use cmp with no option. The command's syntax is:

Syntax:

$cmp [options] file1 file2

options

-l Gives you more information. It displays the number of each character that is different (the first character in the files is number 1), and then prints the octal value f the ASCII code of that character. (You will probably not have any use for the octal value of a character until you become a shell programming expert).

-s Prints nothing but returns an appropriate result code (0 if there are no differences, 1 if there are one or more differences). This option is useful when you write shell scripts.

Examples:

$cat >na1
allen Christopher
babinchak david
best betty
bloom dennis
boelhower joseph
bose anita
cacossa ray
delucia joseph

$cat > na2
allen Christopher
babinchak David
best betty
boelhower joseph
bose
cacossa ray
delucia joseph

Note that the first difference between the two files is on the second lines. The D in David in the second file is the 29th character, counting all new line characters at the ends of lines.

$cmp –s na1 na2
$echo $?

output:

1

The variable ? is the shell variable that contains the result code of the last command, and $? Is its value. The value 1 on the last line indicates that cmp found at

least one difference between the two files. If is there no difference found at the time of comparing two files then output will be 0.

✪comm command (Finding What is Common)

The comm. Command compare two sorted files line by line. comm. prints lines that are common, and lines that are unique, to two input files. The two files must be sorted before comm. can be used. The filename – means the standard input. With no options, comm. produces three column output. Column one contains lines unique to files1, column two contains lines unique to file2, and column three contains lines common to both files.

Syntax:

$comm [options] [-help] [-version] file1 file2

The options -1, -2, and -3 suppress printing of the corresponding columns.

-help Print a usage message and exit with a nonzero status.

-version Print version information on standard output then exit.

✪diff command (Converting One File to Other)

The diff command is much more powerful than the cmp command. It shows you the differences between two files by outputting the editing changes that you would need to make to convert one file to the other.

Syntax:

$diff [options] file1 file2

-b Ignores trailing blanks, and treats all other strings f blanks as equivalent to one another.
-i Ignores uppercase and lowercase distinctions.
-t Preserves indentation level of the original file by expanding tabs in the output.
-w Ignores all blanks (spaces and tabs).

First, let's look at the two files that show what diff does:

Let's apply diff to the files na1 and na2 (the files with which cmp was demonstrated):

Example:

$diff na1 na2

2c2
<babinchak david
-
>babinchak David
4d3
<bloom dennis
6c5
<bose anita
-
>bose

These editor commands are quite different from that diff printed before. The first four lines show

2c2
<babinchak david
-
>babinchak David

Which means that you can change the second line of file1 to match the second line of file 2 by executing the command, which means change line 2 of file 1 to line 2 f file2. Note that both the line from file1 prefaced with <-and the line from file2-prefaced with >- are displayed, separated by a line consisting of three dashes.

The next command sys to delete line 4 from file1 to bring it into agreement with file2 up to but not including line3 of file2. Finally, notice that there is another change command, 6c5, which says change line 6 of file 1 by replacing it with line 5 of file2.

You can use the –i option to tell diff to ignore the case of the characters, as follows:

$diff –i na1 na2
4d3
<bloom dennis
6c5
<bose anita
-
>bose

The –c option causes the differences to be printed in context, that is, the output displays several of the lines above and below a line in which diff finds a difference. Each difference is marked with one of the following:

An exclamation point (!) indicates that corresponding lines in the two files are similar but not the same.
A minus sign (-) means that the line is not in file2.
A plus sign (+) means that the line is in file2 but not in file1.

$diff –c na1 na2

Changing information in files using tr and sed

(1) tr (Translating Characters)

This command is used to translate character for any particular files. The tr filter manipulates individual character in a line.

Syntex:

$tr expression 1 expression2

Example 1

```
$cat >> abc.txt
abcd abcd abcd abcd abcd
abcd abcd abcd abcd abcd
ctrl+d
$tr '[a-z]' '[A-Z]' < abc.txt
ABCD ABCD ABCD ABCD ABCD
ABCD ABCD ABCD ABCD ABCD
$
```

Example 2 with same file

```
$tr '[a|b|c]' '[o]' < abc.txt
oood oood oood oood oood
oood oood oood oood oood
$
```

Example 3 with same file

```
$tr 'abc' 'opq' < abc.txt
opqd opqd opqd opqd opqd
opqd opqd opqd opqd opqd
$
```


option –d Delete given character from the files.

Example 4

```
$tr –d '/a' < abc.txt
bcd bcd bcd bcd bcd
bcd bcd bcd bcd bcd
$
```

sed (Stream editor)

sed is a multipurpose tool which combines the work of several filters. It is derived from ed. The sed performs non interactive operations on a data stream hence its name. It uses very few options but has a host of feature that allows you to select lines and run instructions on them. Learning sed will prepare you well for perl which uses many of these features.

sed uses instructions to act n text. An instructin combines an address for selecting lines, with an action to be taken on them, as shown by the syntax:

Syntex:

sed [option] 'address action' file name

The address and action are enclosed within single quotes. Addressing in sed is done in two ways

By one or two line numbers (like 3, 7)
By specifying a /- enclosed pattern which occurs in a line (like /From:/)

Common files

```
$cat >> abc.txt
1. abcd abcd abcd abcd abcd
2. abcd abcd abcd abcd abcd
3. abcd abcd abcd abcd abcd
4. abcd abcd abcd abcd abcd
5. abcd abcd abcd abcd abcd
6. abcd abcd abcd abcd abcd
```

Example1 Line addressing

```
$sed '3q' abc.txt
1. abcd abcd abcd abcd abcd
2. abcd abcd abcd abcd abcd
```

```
3.      abcd    abcd    abcd    abcd    abcd
$
```

Example2 Print line from anywhere

```
$sed –n '5,6p' abc.txt
5.      abcd    abcd    abcd    abcd    abcd
6.      abcd    abcd    abcd    abcd    abcd
```

Example3 Print line from anywhere (Negative Action)

```
$sed –n '4,$!p' abc.txt
1.      abcd    abcd    abcd    abcd    abcd
2.      abcd    abcd    abcd    abcd    abcd
3.      abcd    abcd    abcd    abcd    abcd
$
```

Example4 Using multiple instructions

```
$sed –n –e '1p' –e '3p' abc.txt
1.      abcd    abcd    abcd    abcd    abcd
3.      abcd    abcd    abcd    abcd    abcd
$
```


od command: (Displaying data in octal)

Many files (especially executables) contain nonprinting characters, and most UNIX commands don't display them properly. The file odfile contains some of these characters that don't show up normally.

More about od command

$cat > odfile
White space includes a
The ctrl+G character rings a bell
The ctrl+L character skips a page
ctrl+d

The apparently incomplete first line actually contains a tab. The od command makes these commands visible by displaying the ASCII octal value of its input. **The –b option displays this value for each character separately.** Here's a trimmed output:

$od –b odfile

```
0000000 127 150 151 164 145 040 163 160 141 143 145 040 151 156 143 154
0000020 165 144 145 163 040 141 040 011 012 124 150 145 040 007 040 143
etc….
```

Each line displays 16 bytes of data in octal, preceded by the offset (position) in the file of the first byte in the line. In the absence of proper mapping it's difficult to make sense out of this output, but when the –b and –c option are combined, the output is friendlier:

$od –bc odfile

```
0000000 127 150 151 164 145 040 163 160 141 143 145 040 151 156 143 154
          W   h   i   t   e       s   p   a   c   e       i   n   c   l
0000020 165 144 145 163 040 141 040 011 012 124 150 145 040 007 040 143
          u   d   e   s       a       \t  \n  T   h   e       007     c
etc……
```

Each line is now replaced with tow. The octal representations are shown in the first line. The printable character and escape sequences are shown in the second line. The first character in the first line is the letter W having the octal value 127. You will recall having used some of the escape sequences with echo. Let's have a look at their various representations.

The tab character Ctrl-i is shown as \t and the octal value 011.
The ball character Ctrl-g is shown as 007. Some systems show it as \a.

The formfeed character, Ctrl-l is shown as \f and 014.
The LF character Ctrl-j is shown as \n and 012. Note that od makes the newline character visible too.

Tools for mathematical calculations

● **bc command**

This command is used to have calculator at prompt.

Syntax

$bc
-

Examples

To do simple calculation

$bc
5+6
11
Ctrl+d

To do multiple calculation

$bc
5+6;5*2
11
10

To have decimal point
$bc
scale=2
5/2
2.50

To have binary to decimal
$bc
ibase=2
1111
15

To have decimal to binary

$bc
obase=2
15
1111

● **factor command**

When factor command is invoked without an argument, it waits for a number to be typed in. If you type in a positive number less then 2^46. It will factories the number and print its prime factors; each one is printed the proper number of times. Then it waits for another number. It exits if it encounters a zero or any non-numeric character.

Example

$factor
15
3
5

28
2
2
7
q to quit

$

If factor is invoked with an argument, it factors the number as above and then exits.

Monitoring input and output command

● **tee**

tee is an external command and not a feature of the shell. It handles a character stream by duplicating its input. It saves one copy in a file and writes the other to standard output. tee can be placed anywhere in a pipeline. It creates new file and stored enter data and also give the output at a time to standard output.

Syntax

$tee [other command] filename

Other command

User can place other command using pipeline.

Example 1

Creating a.txt file using tee command.

	$tee abc
	abc abc abc
Output Displayed to next row	abc abc abc
	xyz xyz xyz
Output Displayed to next row	xyz xyz xyz
	Ctrl + d

Example 2

Using pipeline

$who | tee user.txt

Username Terminal No. Date and Time

Same information will be stored in user.txt file.

● **script (Recording your session)**

To store your login session or any particular session script command is used.

Syntax:

$script **Filename**

Example:

$script abc

script started, output file is abc

$ls
bbb xyz.txt
$wc bbb
1 1 1 bbb
$exit
script done, output file is xyz

$cat
$ls
bbb xyz.txt
$wc bbb
1 1 1 bbb
$exit
script done, Date and Time will be shown here

Tools for displaying date and time

● **cal**

cal is a handy tools that you can invoke any time to see the calendar of any specific moth, or a complete year. To see the calendar for the month of July 2009, provide the month number and year as the two arguments to cal:

Syntax

$cal [Month Number Year]

Month Number Year

This option will show you the calendar for the particular month of particular year. To see the Aug 2009 calendar then following command will be invoked.

$cal 8 2009

Above command will show you the calendar for the month of august 2009.

User can see also all year calendars for the particular year for that user has to write year number after cal command.

Example

$cal 2009

Above command will show you 12month calendar for the year of 2009.

With cal, you can produce the calendar for any month or year between the year 1 and 9999. This should serve our requirements.

● **date (Displaying date and time)**

UNIX has a date command that shows the date and time in the form used on the Internet.

Syntax:

```
$date
Fri     Aug     19      13:28:34        IST     2005
$_
```

date is a valid command, and it displays both the date and time. Notice another security feature of UNIX; the command doesn't prompt you to change either the date or time. This facility is available only to the administrator, and the strange thing is that he uses the same command.

Communication

● **telnet (Remote Login)**

The telnet command will allow you to log on to a remote machine. If yu have an account on a host in a local network (or on the Internet), you can use telnet with the hostname or IP address as argument:

Syntax

```
$telnet IP Address
```

Example

```
$telnet 192.168.35.12….
Trying 192.168.35.12.
Connected to 192.168.35.12.
Escape character is '^]'.

SunOS 5.8
Login:
```

You now have to enter your login name at this prompt, and then the password to gain access to the remote machine. As long as you are logged in, anything you type is sent to the remote machine, and your machine just acts as a dumb terminal. Any files

you use or command that you run will always be on the remote machine. After you have finished, **you can press ctrl+d or type exit to logout and return to your local shell.**

The escape character lets you make a temporary escape t the telnet> prompt so you can execute a command on your local machine. To invoke it, press [Ctrl-]]. YYou can then use the! With a UNIX command, say ls to list files on the local machine:

```
$ctrl+]
telnet>!ls –l
```

A telnet session is closed in the same way as any login session. **Use [Ctrl + d] or the command appropriate to the shell.** If you are at the telnet> prompt, use quit, an internal command of telnet.

● **wall (Communicating with users)**

The wall command addresses all users simultaneously. Most UNIX systems don't permit users to run this command and reserve it for the sole use of the administrator.

#wall
The machine will be shut down today
At 14:30 hrs. The backup will be at 13:30 hrs
Ctrl+d

All users currently logged in will receive this message on their terminal. This command is routinely executed b the administrator—especially before shutdown of the system.

● **mtod (For message of the day)**

The 'message of the day' like the news, is typed by the superuser in the file /etc/mtod. In case of the news command it is entirely the user's discretion whether to read the news or not. As against this the user doesn't have a choice whether to read the message of the day or not. Reading the message of the day is mandatory, as it is displayed as soon as you login. As soon as you login a file /etc/profile file gets executed. This file contains a command cat /etc/motd. As a result the contents of /etc/motd are bound to get displayed on the screen. **When I logged in I found the following message on my screen.**

The air-conditioning system needs a maintenance check! Hence the system would be down between 2.00 to 5.00 PM. See you at 5.00

The /etc/profile is like our good old AUTOEXEC.BAT of DOS. It gets executed every time the user logs in. In addition every user has his own .profile file in his home directory using which he can customize his working environment. The sequence of executing of these files is /etc/profile followed by .profile.

As you must have guessed it is only the superuser who can change the contents of the file /etc/mtod.

● **Write (Communicating with users)**

The write command can be used by any user to write something on someone else's terminal, provided the recipient of the message permits communication.

Syntax

$write **user name**

Example

$write user2
Hey there! I am back to the office
Ctrl+d

On executing this command the message would be relayed to the user whose login name is user2. He would hear a beep on his terminal, followed by the message:

Message from user1 on UNIX (tt3a) [Thu Oct 15 17:13:58]….
Hey there! I am back to the office
(end of message)

There are two prerequisites for a smooth write operation:

1) The recipient must be logged in, else an error message is ineitable.
2) The recipient must have give permission for messages to reach his or her terminal. This is done by saying the $ prompt.

$mesg –y

If you are expecting nothing of consequence and do not wish to be disturbed by Social trivia like the one we just saw, you can deny write permission to your terminal by saying.

$mesg –n

A superuser however can write to any terminal, irrespective of whether mesg has been set –y or –n.

● **mail**

Using mail you can quickly and efficiently circulate memos and other written information to your co-workers, including directions for the out of town party this Saturday and the latest meanderings. You can even send and receive mail from people outside your organization, if you and they use networked computers.

Write command required that the user to whom the message is to be sent not only to be logged in, but also open to messages. Unlike this, mail can be sent to users who have logged in currently or even to users who haven't logged in currently. In case the user has logged in at several terminals the moment mail is sent to this user it becomes available at all the terminals.

Sending mail

Syntax

$mail **user name**
Subject:-UNIX Course
Body of the message
Ctrl+d

Sending mail more then one user

Syntax

$mail **user name1 user name2 user name3**
Subject:-UNIX Course
Body of the message
Ctrl+d

Sending mail from the file

Syntax

$mail **user name1 user name2 user name< abc.txt**

Handling Incoming Mail

The incoming mail received by a user is stored in a mailbox. Each user is given a mailbox whose name is same as the log name of the user. For example, for user aa1 the filename would be aa1 and it would be present in /user/spool/mail directory. All mail

received from different sources is appended to this mailbox file. However while viewing mail the messages are shown separately as if they are being read from separate files.

To read the mail that has been received we simply say mail at the shell prompt.

$mail
mail 1 User Name Day and Date Time Subject
mail 2 User Name Day and Date Time Subject
mail 3 User Name Day and Date Time Subject
&

& displayed at the bottom. This is known as the mail prompt. We can issue several commands at this prompt. If you want to know which, you can type a? and obtain help on these commands.

&2
Mail 2 will be displayed

&q
To quit from the mail box.

● **news**

The system administrator is the sole person who can make news under the UNIX OS. He types the information which he wants everyone on the network to know of in different files in /user/news directory. Whenever we log in if any fresh news has come in since we logged out last time then a message is displayed on our terminal saying.

News: rally tennis appeal

Where rally, tennis and appeal are the names of the files in which the news items are available.

We may either choose to ignore it, or if any news item interests us we may decide to peruse the same. To read the news, we have to say at the dollar prompt:

$news
rally
Content of the files will be displayed here

tennis

Content of the files will be displayed here

appeal
Content of the files will be displayed here

If you want to peruse only a particular item from the several items then you can say.

$news rally
rally
Content of the files will be displayed here

There are two more options available with the news command. The –n option only lists the names of the news from the /user/news directory that have not yet been read by you.

-s option which provides a count of the unread new items in the /user/news directory.

$news –s
5 news items.

If you try news –s after you have gone through the entire news item it promptly displays the message 'No news'.

● **finger**

finger command is used to tell you which users are connected and which, if any, can receive messages. **It displays a list of all those who have logged in** and places a * next to those terminals where mesg is set to –n.

Syntax

$finger –i

Above command will displays following output

Login	TTY	When	Idle
ABC	*tty10	Fri Oct 13 17:25	8 minutes 12 seconds
XYZ	tty3	Fri Oct 13 17:21	49 Seconds
JKL	*tty4	Fri Oct 13 16:21	

Process Related Commands

● **ps (Process Status)**

Displays status of current running processes. By default, **it shows process ID**, terminal with which process is associated, time consumed by the process since it has been started and process name (generally command name or program name).

Syntax

$ps [Option]

-f	Full listing showing the PPID (Parent Process ID) of each process
-e or A	All processes including user and system processes
-u	Process of user usr only
-l	Long listing showing memory-related information
-t	Processes running on terminal term, (say /dev/console)

Example

$ps

Simple ps command will display following output

PID	TT	TIME	COMMAND
36724	tty10	0	ksh.exe
37796	tty10	0	ps.exe

● **To run processes in background**

To run a process in the background, UNIX provides the ampersand (&) symbol. While executing a command, if this symbol is placed at the end of the command then the command will be executed in the background. When you run a process in the background a number is displayed on the screen.

This number is nothing but the PID of the process that you have just executed in the background. Let's understand this with an example.

$sort employee.dat > emp.out &
17653
$

The task of sorting the file employee.dat and storing the output in emp.out has now been assigned to the background, letting the user free to carry out any other task in the foreground.

Advantages and limitation of background processes

(1) On terminal of background process no success or failure is reported on the screen. Then how do we keep track of it? That is where the PID displayed on the screen comes in handy. We can search for this PID in the output of ps to verify whether the process is still running or has been terminated.

(2) The output of a background process should always be redirected to a file. Otherwise you would get a garbled screen showing the output of the background process along with whatever you are doing in the foreground.

(3) With too many processes running in the background the overall system performance is likely to degrade.

(4) If you log out while some of your processes are running in the background all these processes would be abandoned halfway through. This is natural because all your processes are children/grandchildren/great grandchildren of your sh (shell) process. And when we log out the sh process dies along with all its children.

● **nice (changing the process priorities)**

Though all processes are equal, some processes are more equal than others. They can be made so by increasing their priority. The processes with higher priority would obviously get a time slot earlier and would be fired earlier than the other processes in the queue.

The priority of a process is decided by a number associated with it. This number is called 'nice' value of the process. Though paradoxical, higher the nice value of a process lower is its priority.
The nice value of a process can range from 0 to 39, with 20 as the default nice value of a process. Thus, a process with a nice value 25 would execute slower then the one with a nice value 20.

Example 1

$nice cat **employee.dat**

This would increase the nice value of our cat process from 20 to 30 Since, we didn't specify the increment, an increment of 10 got assumed and the nice value got correspondingly incremented. If we so desire, we can specify the value of the increment.

$nice – 15 cat employee.dat

Now the cat process will have a nice value 35 (20+15). Note that the increment can range from 0 to 19. By incrementing the nice value we are putting cat on a lower priority and hence it would be executed slower than what it does normally.

$nice --15 cat employee.dat

Above command will reduce the nice value of a process but this kind operation can be done by only administrator. If any user wants to put his/her process with higher priority then he or she has to request to the administrator to put his or her process in higher priority.

Note that the priority of a process can be changed at the time of firing the process at command prompt. Once the process has been submitted to the process queue its priority cannot be changed.

User can see the nice values of the various process running in memory using ps –l command.

● **kill (Killing the process)**

When kill command is invoked, it sends a termination signal t the process being killed. A signal is a mechanism to communicate with a process. These signals have been given numbers. In the above example we have not communicated the signal number to the process to be killed. Hence, the default signal is same as the one generated when you hit the Del key in the middle of a ls or cat command. This default signal however is not very powerful and may not be able to kill a process at all times. A good example is the sh process which cannot be terminated by this default signal.

Syntax

$kill PID (Process ID)

To delete process forcefully then following syntax is used

$kill -9 PID

User can see the process id using ps cmmand.

Another important point, you can kill only your processes and not those fired by other users. Super user of course enjoys an altogether different status. He can kill any of the user's processes easily.

● **at**

This command is capable of executing UNIX commands at a future date and time. The UNIX command can be specified at the command prompt or can be stored in a file and the at command can use this file to execute the commands. Both these facilities are exemplified below.

$at 17:00
echo "It's 5 PM! Backup your files and logout"
Ctrl + d
Job 853158864.a at Wed Jun 14 17:00:00 IST 1996

At 5 PM in the evening you would see a message on your screen "its 5 PM! Backup your files and logout".

● **batch**

Instead of we specifying that our commands be executed at a precise moment in time sometimes we may let the system decide the best time for executing our commands. The way to achieve this is through a command called batch. When we submit our jobs using this command, UNIX executes our job when it is relatively free and the system load is light. Since the time of execution of our commands is left for the system to decide we don't specify the time while executing the batch command.

$batch
echo "Hello Word"
Cltr + d
Job 692322435.b at Fri Jun 14 17:00:00 IST 1996

Once again note that the 'b' extension given to our job-id signifies that it has been submitted using the batch command.

● **Cron (Running Jobs Periodically)**

At and batch command are meant for one-time execution but corn executes programs at regular intervals. It is mostly dormant, but every minute it wakes up and looks in a control file (the crontab file) in /var/spool/cron/crontabs for instructions to be performed at that instant. After executing them, it goes back to sleep, only to wake up the next minute.

A user may also be permitted to place a crontab file after his/her login name in the crontab directory kumar has to place his crontab commands in the file /var/spool/cron/crontabs/kumar. This location is, however, system-dependent. A specimen entry in the file /var/spool/cron/crontabs/kumar can look like this:

00-10 17 * 3,6,9,12 5 find / -newer .last_time -print backuplist

The first field (legal value 00 to 59) specifies the number of minutes after the hour when the command is to be executed. The range 00-10 schedules execution every minute in the first 10 minutes of the hours. The second field (17, i.e. 5 p.m.) indicates the hours in 24 hour format for scheduling (legal value 1 to 24).

The third field (legal value 1 to 31) control the day of the month. This field (here, an asterisk), read with the other tow, implies that the command is to be executed every minute, for the first 10 minutes, starting at 5 p.m. ever day. The fourth field (3,6,9,12) specifies the month (legal values 1 to 12). The fifth field (5 – Friday) indicates the days of the week (legal values 0 to 6), Sunday having the value 0.

- **crontab**

User can also create your own crontab files with vi in the format shown below

00-10 17 * 3,6,9,12 5 find / -newer .last_time -print backuplist

The first field (legal value 00 to 59) specifies the number of minutes after the hour when the command is to be executed. The range 00-10 schedules execution every minute in the first 10 minutes of the hours. The second field (17, i.e. 5 p.m.) indicates the hours in 24 hour format for scheduling (legal value 1 to 24).

The third field (legal value 1 to 31) control the day of the month. This field (here, an asterisk), read with the other tow, implies that the command is to be executed every minute, for the first 10 minutes, starting at 5 p.m. ever day. The fourth field (3,6,9,12) specifies the month (legal values 1 to 12). The fifth field (5 – Friday) indicates the days of the week (legal values 0 to 6), Sunday having the value 0.

You will then need to use the crontab command to place the file in the directory containing crontab files for cron to read the file again:

crontab cron.txt cron.txt contains cron commands

If kumar runs this command, a file named kumar will be created in /var/spool/cron/crontabs containing the contents of cron.txt. In this way, different users can have crontab files named after their user-ids.

- **wait: (Waiting for process to complete)**

The wait command built into most shells (including all the shells discussed in this book) will wait for completion of all background processes or a specific background process. Usually, wait is used in scripts, but occasionally you may want to use it

interactively to wait for a particularly important background job or to pause until all of your current background jobs complete so you will not load the system with your next job. **The command will wait for a particular PID.** If you are using a job control shell, you can use a job identified instead of a PID.

Example:

$job1 & job2 & wait job2 id (Process Id job1 [01] & job2 [02]

$job 1 & job2 & wait [02]

- **sleep**

Most interactive use of wait in can be replaced by notify. The notify command tells the shell not to wait until issuing a new prompt before telling you about the completion of all or some background jobs. The command notifies will tell shell to give asynchronous notification of job completion. The command notifies job id will tell the shell to give asynchronous notification for a particular job.

Example

$sleep 30 & sleep 10 & notify %2

```
[2] Done      Sleep 10
[1] Running   Sleep 20
$
```

When you do this example, don't type anything after hitting return to enter notify %2.
The notification appears as soon as job 2 finishes.

- **mount & umount**

A file system can be "mounted" on your Linux system interactively or automatically at startup. Then the file system is just as accessible as any other file system on your computer. The command to "attach" a file or directory is **mount**, and the command to "detach" a mounted file or directory is **umount** (see UNIX *man* (manual) pages for the commands mount and umount for more information). Users should detach from the file system when the resource is no longer needed.

Generic Mount Points

A logical place to locate mount points is in the directory **/mnt**. Only the *superuser* (or *root*) can create directories in /mnt. Redhat comes with mount points already set up for a floppy, a CD-ROM, and a parallel port Zip drive. To export a file or directory from a server, item 4 below, it may need to be explicitly specified by *root* in the /etc/fstab file:

Node:Mount_dir	Mount Point	File Type	mount Options
/dev/fd0	/mnt/floppy	auto	noauto 0 1
/dev/cdrom	/mnt/cdrom	iso9660	ro,noauto 0 1
/dev/zip	/mnt/zip	Vfat	auto,exec 0 1
server:/exported	/mnt/server	nfs	Defaults

• The Case Control Structure

Another way of controlling the sequence of execution is using the case control structure. Though if-else constructs can be nested, and abundance of conditions may make tracing the control in a program difficult. Hence, for a program where we are required to select from several alternatives a case control structure is the answer. The general form of the case control instruction is given below.

Syntax

```
case value in

choice 1)
        Statement block;;
choice 2)
        Statement block;;
choice )
        Statement block;;
*)
        Statement block;
case
```

Example

```
Echo "Enter a number from 1 to 3: \c"
read num

case $num in
        1)  echo you entered 1;;
        2)  echo you entered 2;;
        3)  echo you entered 3;;
```



```
        *)  echo you entered not 1,2,3;
esac
```

• The while loop

It is often the case in programming that you want to do something a fixed number of times. The while loop is ideally suited for such cases. Let us look at a simple example which uses a while loop.

Syntax

```
while [ Condition ]
do
        statement block
done
```

Example

```
count=1
while [ $count –le 3 ]
do
        echo $count
done
```

The program executes all commands between do and done 3 times. These commands form what is called the 'body' of the while loop. Immediately following the while is a control command. So long as the exit status of the control command is true, all commands within the body of the while loop keep getting executed repeatedly. To begin with, the variable count is initialized to 1 and every time the simple interest logic is executed the value of count is incremented by one. The variable count is many a times called either a 'loop counter' or an 'index variable'.

• The Until Loop

An until loop looks like this:

```
unitl [ Condition ]
do
        Statement Block
done
```

Example

```
count=1
while [ $count –ge 3 ]
do
```

```
        echo $count
done
```

The statements within the until loop keep on getting executed till the exit status of the control command remains false (1). When the exit status becomes ture (0), the control passes to the first command that follows the body of the until loop (i.e. the first command after done).

There is a minor difference between the working of while and until loops. The while loop executes till the exit status of the control command is true and terminates when the exit status becomes false. Unlike this the until loop executes till the exit status of the control command is false and terminates when this status becomes true. This difference is brought about more clearly by the following programs.

• Using for with command line arguments

The for loop is more frequently used as compared to the while and the until loops. Its working is also different than the other two loops. The for allows us to specify a list of values which the control variable in the loop can take. The loop is then executed for each value mentioned in the list.

Syntax

```
for variable in value1 value2 value3…
do
        Statement block
done
```

Here the for loop would execute the same sequence of commands for values mentioned in the list following the keyword in. Here is a simple example of the for loop.

Example

```
For $word in High on a hill was a lovely mountain
do
        echo $word
done
```

Output

```
High
on
a
hill
was
a
```



```
lovely
mountain
```

• if then fi

Like most programming languages, shell to uses the keyword if to implement the decision control instruction. The simplest form of the if statement looks like this:

Syntax:

```
if [ Condition ];then
        True Block
fi
```

If you are familiar with any high level language you must have used some form of an if statement. The UNIX shell if statement is different from others at least on one count. In most high level languages the if statements are usually concerned with the values of variables – is age greater than 20, is answer equal to yes or some such condition.

Example

```
echo "Enter a Number \c"
read N
if [ $N –eq 10 ]; then
echo "Enter number is 10"
fi
```

• if…then….else….fi

The if statement by itself will execute a single command, or a group of commands, when the exit status of the control command is 0. It does nothing when the exit status is 1. Can we execute one group of commands if the exits status is 0 and another group if the exit status is 1? Of course. This is what is the purpose of the else statement, which is demonstrated in the following shell script.

Syntax:

```
if  [ Condition ];then
        True Block
else
        False Block
fi
```

Example

```
echo "Enter a Number \c"
read N
if [ $N –eq 10 ]; then
echo "Enter number is 10"
else
echo "Enter number is not 10"
fi
```

- **if…then….elif….fi**

Example

```
echo "Enter a Number \c"
read N
if [ $N –eq  0 ]; then
echo "Enter number is 0"
elif [ $N –gt 0 ];then
echo "Enter number is greater then 0"
else
echo "Enter number is less then 0"
fi
```

test Command:

Checks file types and compares values.

Syntax:

test EXPRESSION
[EXPRESSION]

Description

test is used as part of the conditional execution of shell commands.

test exits with the status determined by EXPRESSION. Placing the EXPRESSION between square brackets ([and]) is the same as testing the EXPRESSION with test. To see the exit status at the command prompt, echo the value "$?" A value of 0 means the expression evaluated as true, and a value of 1 means the expression evaluated as false.

Expressions take the following forms:

(EXPRESSION)	EXPRESSION is true
! EXPRESSION	EXPRESSION is false
EXPRESSION1 -a EXPRESSION2	both EXPRESSION1 and EXPRESSION2 are true
EXPRESSION1 -o EXPRESSION2	either EXPRESSION1 or EXPRESSION2 is true
-n STRING	the length of STRING is nonzero
STRING	equivalent to -n STRING
-z STRING	the length of STRING is zero
STRING1 = STRING2	the strings are equal
STRING1 != STRING2	the strings are not equal
INTEGER1 -eq INTEGER2	INTEGER1 is equal to INTEGER2
INTEGER1 -ge INTEGER2	INTEGER1 is greater than or equal to INTEGER2
INTEGER1 -gt INTEGER2	INTEGER1 is greater than INTEGER2
INTEGER1 -le INTEGER2	INTEGER1 is less than or equal to INTEGER2
INTEGER1 -lt INTEGER2	INTEGER1 is less than INTEGER2
INTEGER1 -ne INTEGER2	INTEGER1 is not equal to INTEGER2
FILE1 -ef FILE2	FILE1 and FILE2 have the same device and inode numbers
FILE1 -nt FILE2	FILE1 is newer (modification date) than FILE2
FILE1 -ot FILE2	FILE1 is older than FILE2
-b FILE	FILE exists and is block special
-c FILE	FILE exists and is character special
-d FILE	FILE exists and is a directory
-e FILE	FILE exists
-f FILE	FILE exists and is a regular file
-g FILE	FILE exists and is set-group-ID
-G FILE	FILE exists and is owned by the effective group ID
-h FILE	FILE exists and is a symbolic link (same as -L)
-k FILE	FILE exists and has its sticky bit set
-L FILE	FILE exists and is a symbolic link (same as -h)
-O FILE	FILE exists and is owned by the effective user ID
-p FILE	FILE exists and is a named pipe
-r FILE	FILE exists and read permission is granted
-s FILE	FILE exists and has a size greater than zero
-S FILE	FILE exists and is a socket
-t FD	file descriptor FD is opened on a terminal
-u FILE	FILE exists and its set-user-ID bit is set
-w FILE	FILE exists and write permission is granted

-x FILE	FILE exists and execute (or search) permission is granted

Example:

test 100 -gt 99 && echo "Yes, that's true." || echo "No, that's false."

This command will print the text "Yes, that's true." because 100 is greater than 99.

Use of Logical Operator

Shell allows usage of three logical operations while performing a test. These are:

(a) –a (read as AND)
(b) –o (read as OR)
(c) ! (read as NOT)

The first two operators, -a and –o, allow two or more conditions to be combined to be combined in a test. Let us see how they are used in a program Consider the following problem.

Relational Operator

-lt	Less then
-gt	Greater then
-le	Less then equal to
-ge	Greater then equal to
-eq	Equal to
-ne	Not equal to

Chapter No. 8 Shell Programming

Shell Variables

- **System Variable**

PS2	:-	Secondary prompt string (default >)
PATH	:-	Search path for commands; multiple pathnames are separated with a colon (default /bin: /usr/bin)
HOME	:-	Default argument for the cd command; contains the pathname of the home directory.
LOGNAME	:-	Holds user's login name
MAIL	:-	Name of file to check for incoming mail
IFS	:-	Internal field separator (default space, tab or new line)
SHELL	:-	Pathname of the shell
TERM	:-	Specifies your terminal type
MAILCHECK	:-	Specifies how often to check for mail in $MAIL or $MAILPATH. If set to 0, mail is checked before each prompt (default 600 seconds).

User Variable

Set

The set statement displays all variables available in the current shell. This command will show the list of all variables and their expression.

Unset

To remove the definition of a function from the shell we can use the unset command. For example,

$unset HOME

Now user does't have HOME variable that stored the home directory path.

echo

This command is used to print something on the screen. This command is just like printf in c language. To print value of any variable using echo that variable should have $ followed by variable name.

read

This command is used to input a value of any variable at the time of running program. This command is just like scanf in c language.

Getting Started with Linux

1) History of Linux

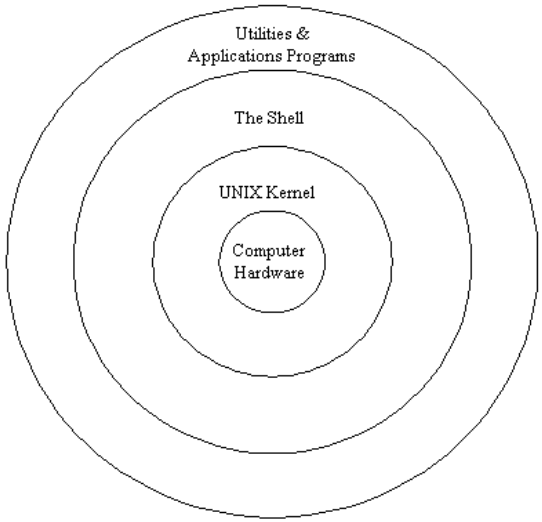
The History of Linux began in **1991** with the commencement of a personal project by a Finnish student, Linus Torvalds, to create a new free operating system kernel.

Since then, the resulting Linux kernel has been marked by constant growth throughout its history. Since the initial release of its source code in 1991, it has grown from a small number of C files under a license prohibiting commercial distribution to its state in 2009 of over 370 megabytes of source under the GNU General Public License.

Summary

LINUX began in 1991; it has small number of c files. UNIX has a long history as an open development environment. More recently, it has become the system of choice for both commercial and some personal uses. UNIX performs the typical operating system tasks, but also includes a standard set of commands and library interfaces. The building-block approach of UNIX makes it an ideal system for creating new applications.

2) Structure of Linux/Unix



Summary

It has contains HARDWARE, KERNEL, SHELL AND UTILITY PROGAMS. It is a heart of Unix/Linux.

3) Features of UNIX

- multi-user:-
  more than one user can use the machine at a time
  supported via terminals (serial or network connection)
- multi-tasking:-
  more than one program can be run at a time
- hierarchical directory structure
  to support the organization and maintenance of files
- portability:-
  only the kernel (<10%) written in assembler
  tools for program development
  a wide range of support tools (debuggers, compilers)

Summary

Multi-user, Multi-tasking and Portability are main Features of Unix.

4) Installation and Configuration of Linux

• In this guide I will cover the installation of Ubuntu Linux 11.10, 32-bit version from a LiveCD. The installation of other 'Ubuntu versions including 64-bit will be very similar to this guide. I will try to explain certain options and provide helpful hints along the way, so rather than just following the guide, you can understand the reasoning behind the decisions.

This guide is drawn from my experience. As some of you are aware, I'm passionate about open-source software and OS', in particular Linux. I'm no professional though, just a person that enjoys spending his spare time using Linux and open-source software in-between dissecting others' computers, and somehow managing to turn my own systems into fireballs of destruction!

If there are any mistakes please draw my attention to them and I will correct as needed. I have tried to make this as simple as possible, whilst covering the vast majority of scenarios users will come across whilst installing this operating system.

Step 1:
The first thing you should do is head to http://www.ubuntu.com/download/ubuntu/download and download Ubuntu 11.10 LiveCD. Just click the big orange Start download box.

Step 2:
Using your disc burning software, burn the .iso you downloaded to a CD.

Step 3:
Before you go any further, ensure all important data is backed up in case of data loss on your drives. This guide assumes you have media backups of your Windows partitioned hard drive and you are safe to proceed.

****Warning: Installing another operating system without first ensuring you have backups of your current files and operating system is a big risk. If you have no data to lose or you've backed up important data, you're ready to proceed. YOU are responsible if you lose data.For those of you using Windows, and installing Linux for the first time I recommend you either use a separate hard disk that does not contain the Windows OS, or create a partition big enough for Linux within Windows using Disk Management in the Administrative Tools menu of the control panel. 30GB of hard disk space is absolutely plenty of space for you to begin exploring Ubuntu whilst at the same time having room to grow.*

Step 4:
Ensure you have a network cable connected, restart your computer, and boot from the CD

drive.

Step 5:
The LiveCD will load up, and you'll be presented by the following box:

For the purposes of this guide, we will assume you've already tried Ubuntu and want to proceed with an installation, so click "Install Ubuntu."

Step 6:
You'll be greeted by the "Preparing to install Ubuntu" screen, exactly as below:

I recommend you select "Install third-party software" as I have done in the screenshot above. I prefer to do system updates once up and running, but if you have the extra time you can also select "download updates while installing" as well. Then click continue.

Step 7:
The next screen you will see is "installation type," what you see will be dependent on whether you have an existing Windows installation or not.

I'm going to split this into three different sub-steps, to make it as simple as possible.

Step 7-A:
For those installing in a virtual machine or to hard disks without an OS you will see the following screen:

You have two choices:

1. Erase the entire disk and use all of it for installation -- Ubuntu will automatically partition your disk and proceed with installation.
2. Select "something else" and manually create your partitions (which is covered in detail in step 7-C).

If you are choosing the first option, select the radio button and then click continue, proceeding to step 8.

Step 7-B:
Those of you that have current Windows installations or are going to dual-boot with another existing OS will be presented with a screen similar to below:

You have three options available:

1. You can choose the first option and install Ubuntu alongside your existing OS.
2. You can opt to replace your Windows installation with Ubuntu, allowing the installer to format your current partitions and automatically create new ones for Linux.
3. You can choose "something else" and create your own partition scheme and sizing (covered in detail in step 7-C).

Once you have selected which route you wish to proceed with click continue and proceed to step 8.

Step 7-C:
Having selected the "something else" option you will be presented with the following window:

Linux recognizes and assigns IDs to drive in a different manner to Windows. In the above image, you can clearly see my hard disk in the list. It is identified by "/dev/sda." Linux recognizes drives in the following way:

- Sda = 1st drive
- Sdb = 2nd drive
- Sdc = 3rd drive and so on

Partitions are also shown after the drive letters. So if I had 2 partitions on my first disk, they would be identified as:

- Sda1 – 1st drive, 1st partition
- Sda2 – 1st drive, 2nd partition

You will not see the common Windows C: label in the disk menu in the above list. You do, however, have key things to help you recognize your Windows C: drive. Both of these can be used to identify which is your Windows disk.

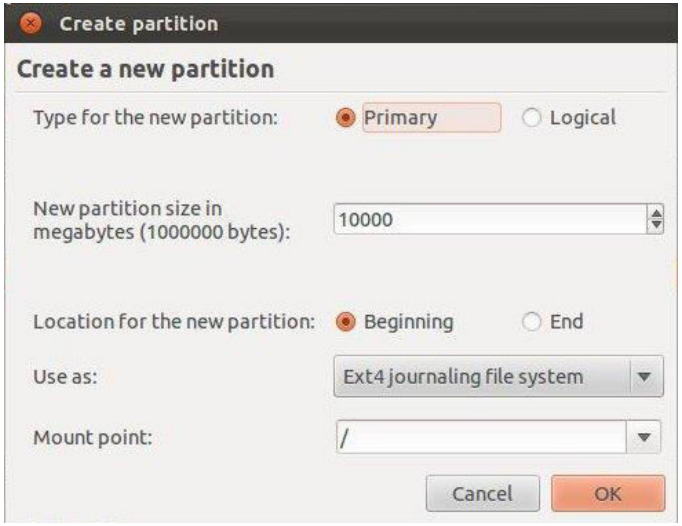
- The size of the disk is shown
- The name of the drive is shown

This is your current partition layout for your hard disks. If you have more than one disk, they will show up as /dev/sda, /dev/sdb etc.

Firstly, identify your Windows installation. In my case, it's sda1 (which is my first hard disk, first partition). What you see depends on how you created the extra space. I just resized the Windows partition from within Windows, and left the free space ready to install Linux. I recommend using Windows or a free utility from within Windows to resize your partition as most beginners will understand it more.

Therefore, we now need create a minimum of two partitions:

Click "Add" and the following box will appear:



You will notice I have already filled out the example above to create a 10GB root partition.

You can have a maximum of 4 primary partitions, or 3 primary partitions and 1 logical (which allows for another 64 partitions)
The size above is 10.00GB. e.g 1,000 = 1GB 10,000 = 10GB (Remember to leave enough free remaining space to create your SWAP partition!)
Location for new partition: e.g. do you want it at the start or end of the free space. Select beginning.
Use as: Ext4 is the recommended file system for Ubuntu, much the same as NTFS is Windows. SWAP is for SWAP space.
Mount point: This is where you want the partition to mount. E.g. we need a root partition, which in Linux is denoted by a "/".

Click OK once you have finished setting the partition information and you will return to your partition screen, now showing the root partition you just created. Using the same methods as before, create a SWAP partition.

I recommend you set the size of your SWAP partition to at least the size of your available RAM. If you have plenty of hard disk capacity I would suggest you use double the size. So if you have 2GB of RAM, set it to either 2GB or 4GB. For best performance it is recommended you have your SWAP partition at the beginning or end of your drive.
Once you have done that, you should be looking at something like below.

So to re-cap the above, (in my example) we have the following:

• /dev/sda1 is your Windows partition.
• /dev/sda2 is your new root partition (Windows equiv. of C🙂.
• /dev/sda3 is your SWAP space.

Once you are happy with the changes you have made, click install now and proceed to the next step.

Step 8:
As the installation starts to copy the required files to the hard disk, you will be presented with a screen to select your locale. It should automatically find where you are, as it has for me already:

Just double check it is correct, and then select continue.

Step 9:
The next screen to appear will be keyboard layout:

Ensure the correct option is selected, above you will see the correct (and default UK) selection has been automatically made for me.

Step 10:
You will now be greeted by the "who are you" screen, ready for you to fill out with your user details:

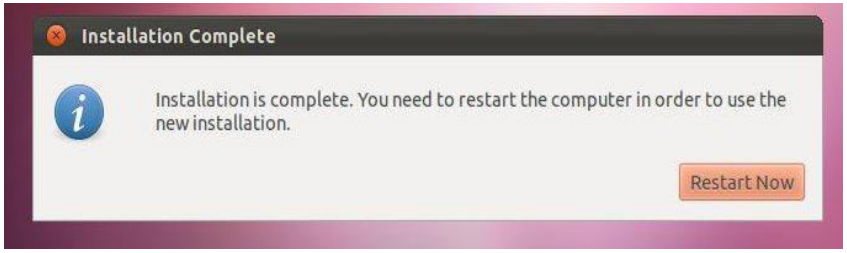
The computers name and username will automatically populate when you type your full name. You can however edit them as you please. Fill in the details and then click continue.

You can opt to have Ubuntu automatically log in for you -- even with a password set -- or you can choose the traditional option requiring a password to log in. You really don't need to choose the encrypt option unless you're installing on a laptop and are dealing with highly secure information.

Step 11:
The installation information screens will now appear as Ubuntu continues the installation:

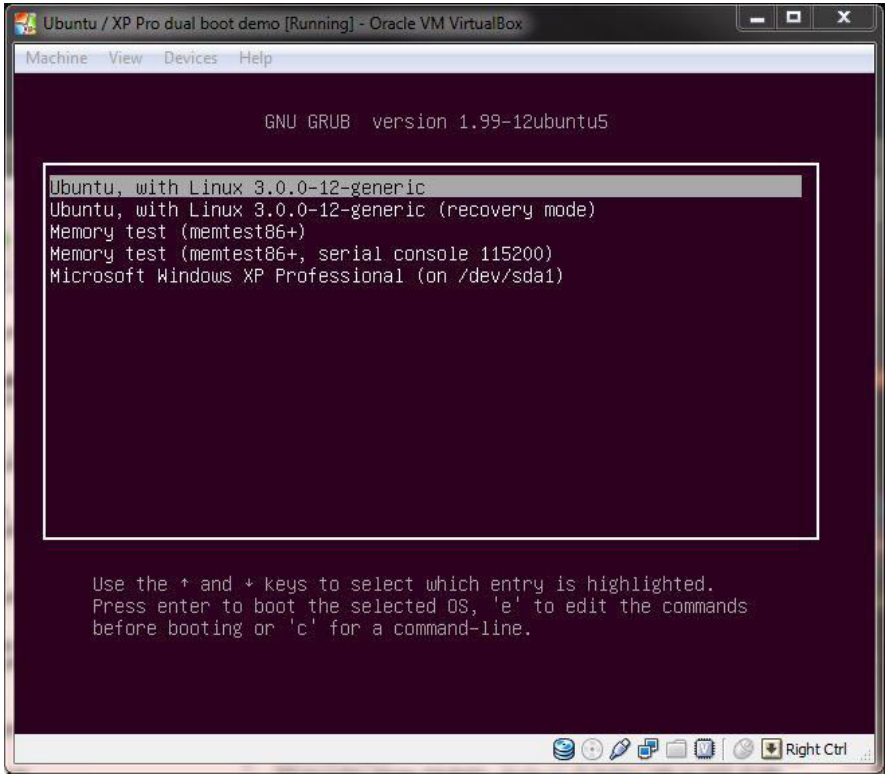
- **Step 12:**

Once installation has finished, you will be presented with the following box:



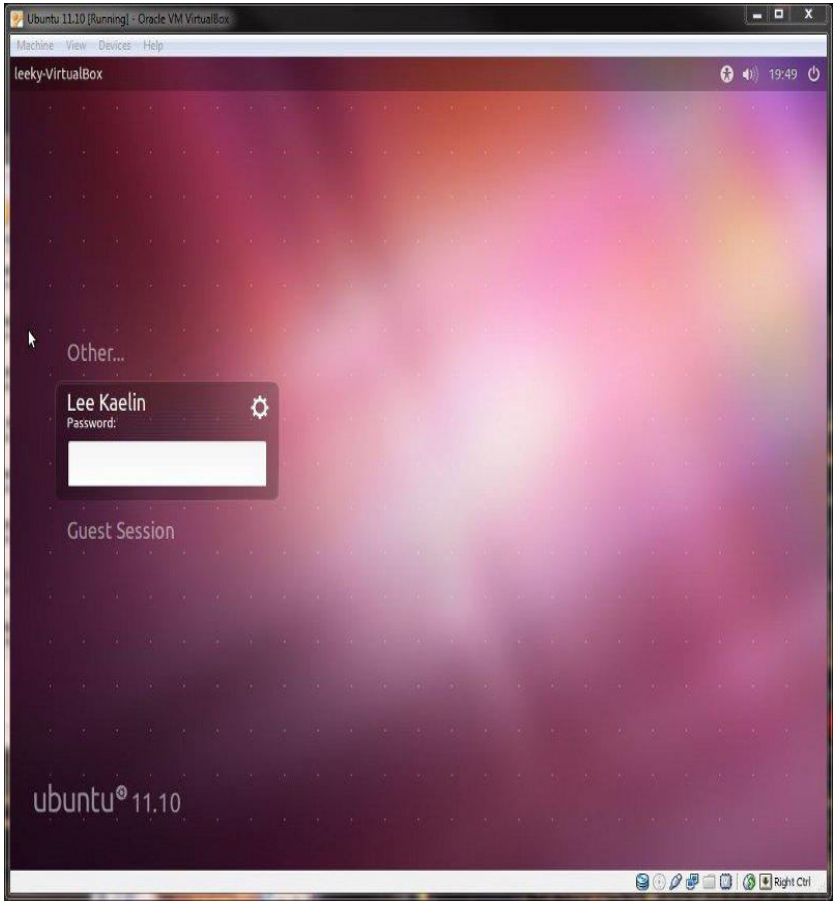
Select "restart now" and when requested, remove your installation CD, then press enter to reboot.

Step 13:
For those of you that have Ubuntu as the only OS the computer will boot directly into Linux. If you're dual-booting, you will see the GRUB menu appear similar to below:



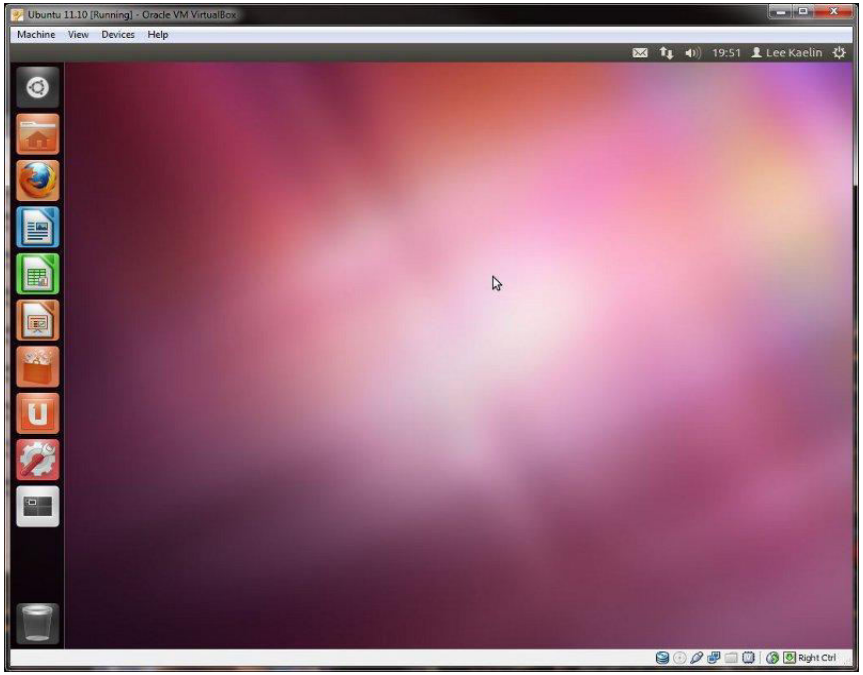
Hit enter, to select the first option and load your newly installed Ubuntu OS.

Step 14:
For those that elected to automatically log into Ubuntu, you will go straight to the desktop in Step 15. For everyone else, you will be greeted with the new login manager for Ubuntu:

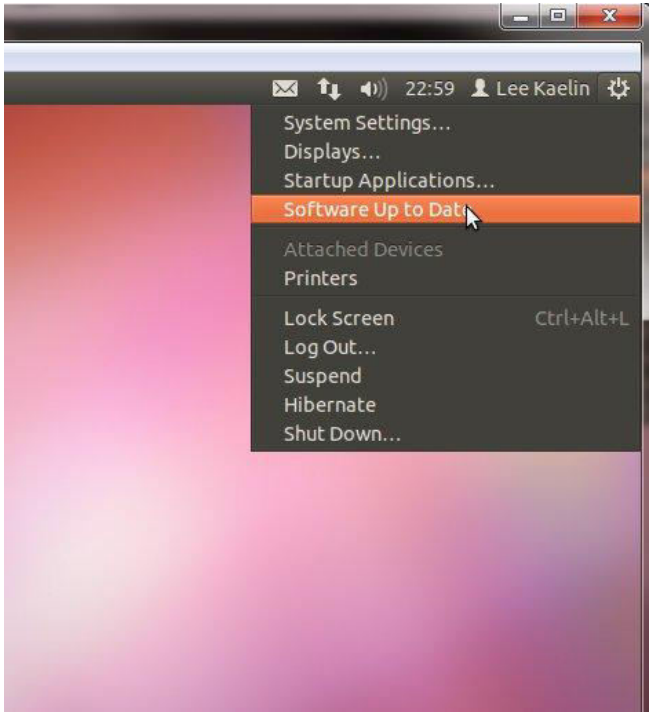


Enter your password, and hit enter to login to your desktop.

Step 15:
Your desktop should look like this:

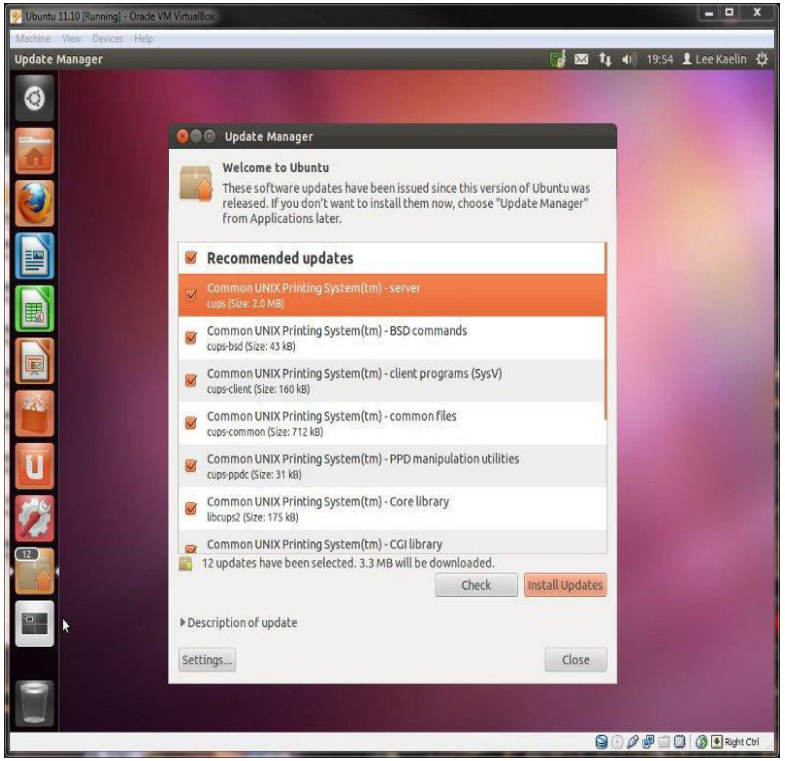


Step 16:
Before we proceed further, let's check for updates. Click on the power button on the top right corner of the screen and select "check for updates," or words to that effect.



I'd already run updates on this install so the example above is displaying "software up to date," but the picture highlights where you need to select anyway.

Upon selecting the update option, the update manager will appear, as below:



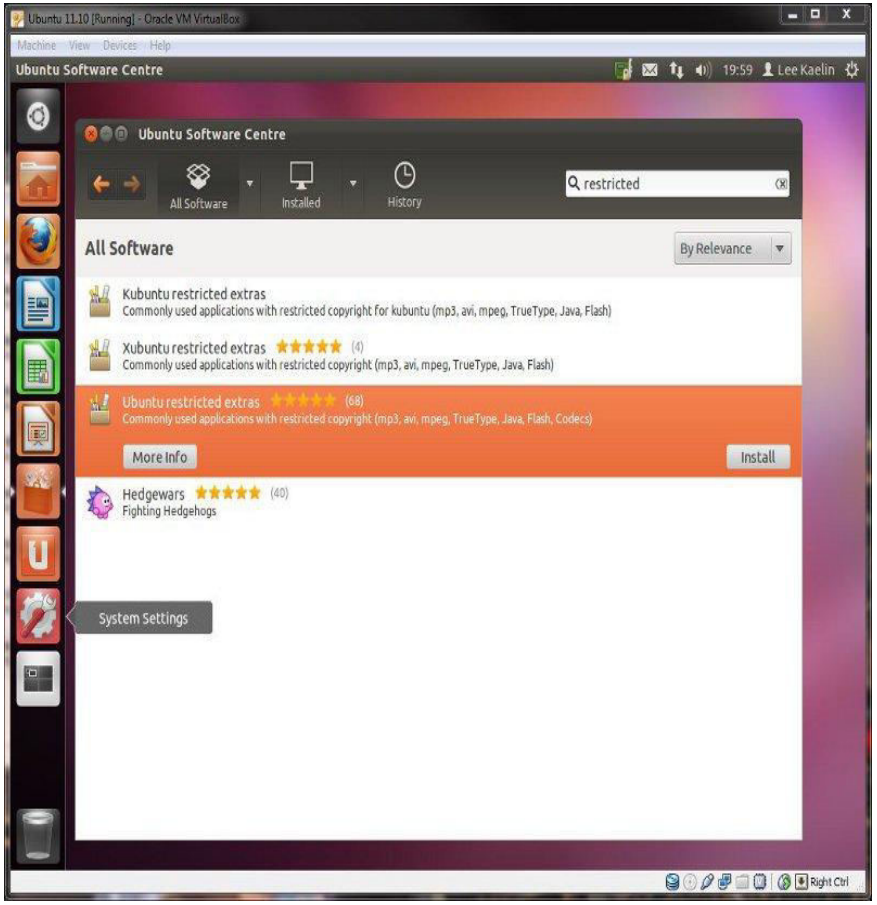
If it comes up with no available updates, just select "check" again to verify that it is correct. Having done the same thing myself, I was presented with the updates you see above. For those that opted to install updates during installation it is unlikely there will be further updates required.

You might be asked to enter your password to confirm changes. If prompted, enter your password and click OK. The same is true of any notifications that may appear during updating the OS.

Once complete select close, and restart Ubuntu. The power button is located on the top right corner of the screen. Click this and select shutdown.

Step 17:
No install is complete without full support for mp3s, core MS fonts, DVD playback codec's, Flash and Java, so let's go ahead and sort this now.

Click the black Ubuntu menu button at the top left corner of the screen and in the menu that appears type "software centre" and select the Ubuntu Software Centre. Once open, click the search bar, type "restricted" and the following should appear:



Select Ubuntu restricted extras, and then click on the install button.

Authentication is much like UAC (user access control) in windows Vista and 7. It is required to elevate your user privileges to that of root (Linux administrator).

Ubuntu restricted extras will now download, sort any dependencies and install. You can check its progress by viewing the progress bar above the install button. Once finished the In Progress tab will disappear -- restart Linux. It's not strictly necessary, but I always do it after installing this package so everything can start up properly.

Now you can enjoy your new OS and begin exploring its features.

Summary

To enjoy your new Linux installation forever without any worries, it is important to take a few simple precautions as above. You must do these as soon as you complete installing Linux on your machine for the first time.

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

No.	Topics	Details
9.	Linux Booting	->LILO Configuration ->GRUB Configuration

Chapter No. 9
Linux Booting

1) LLIO Configuration

The LILO configuration file is `/etc/lilo.conf`. The `/sbin/lilo` commands uses this file to determine what information to write to the MBR.

Warning

Before editing `/etc/lilo.conf`, be sure to make a backup copy of the file. Also, have a working boot disk available so that changes can be made to the MBR if there is a problem. Refer To the man page for `mkbootdisk` for more information on creating a boot disk.

The `/etc/lilo.conf` file is used by the `/sbin/lilo` command to determine which operating system or kernel to load and where it should be installed.

2.9.1. Sample `/etc/lilo.conf`

The following is a sample `/etc/lilo.conf` for a system configured to boot two operating systems, Red Hat Enterprise Linux and DOS:

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
message=/boot/message
lba32
default=linux

image=/boot/vmlinuz-2.4.0-0.43.6
        label=linux
        initrd=/boot/initrd-2.4.0-0.43.6.img
```

```
        read-only
        root=/dev/hda5

other=/dev/hda1
        label=dos
```

The following is a more detailed look at the lines of this file:

- `boot=/dev/hda` — Instructs LILO to be installed on the first hard disk of the first IDE controller.
- `map=/boot/map` — Locates the map file. In normal use, this should not be modified.
- `install=/boot/boot.b` — Instructs LILO to install the specified file as the new boot sector. In normal use, this should not be altered. If the `install` line is missing, LILO assumes a default of `/boot/boot.b` as the file to be used.
- `prompt` — Instructs LILO to show you whatever is referenced in the `message` line. While it is not recommended that you remove the `prompt` line, if you do remove it, you can still access a prompt by holding down the [Shift] key while your machine starts to boot.
- `timeout=50` — Sets the amount of time that LILO waits for user input before proceeding with booting the `default` line entry. This is measured in tenths of a second, with 50 as the default.
- `message=/boot/message` — Refers to the screen that LILO displays to let you select the operating system or kernel to boot.
- `lba32` — Describes the hard disk geometry to LILO. Another common entry here is `linear`. You should not change this line unless you are very aware of what you are doing. Otherwise, you could put your system in an unbootable state.
- `default=linux` — Refers to the default operating system for LILO to boot as seen in the options listed below this line. The name `linux` refers to the `label` line below in each of the boot options.
- `image=/boot/vmlinuz-2.4.0-0.43.6` — Specifies which Linux kernel to boot with this particular boot option.
- `label=linux` — Names the operating system option in the LILO screen. In this case, it is also the name referred to by the `default` line.
- `initrd=/boot/initrd-2.4.0-0.43.6.img` — Refers to the *initial ram disk* image that is used at boot time to initialize and start the devices that makes booting the kernel possible. The initial ram disk is a collection of machine-specific drivers necessary to operate a SCSI card, hard drive, or any other device needed to load the kernel. You should never try to share initial ram disks between machines.
- `read-only` — Specifies that the root partition (refer to the `root` line below) is read-only and cannot be altered during the boot process.
- `root=/dev/hda5` — Specifies which disk partition to use as the root partition.
- `other=/dev/hda1` — Specifies the partition containing DOS.

Summary

LILO (Linux Loader) is a boot loader for Linux and was the default boot loader for most Linux distributions in the years after the popularity of loading. Today, most distributions use GRUB as the default boot loader

2) GRUB Configuration

GRUB is a boot loader designed to boot a wide range of operating systems from a wide range of file systems. GRUB is becoming popular due to the increasing number of possible root file systems that can Linux can reside upon.

GRUB is documented in a GNU info file. Type **info grub** to view the documentation.

The GRUB configuration file is `/boot/grub/menu.lst`. Some distributions use another configuration file; for example, Red Hat Linux uses the file `/boot/grub/grub.conf`.

GRUB configuration files are interpreted. Syntax errors will not be detected until the machine is rebooted, so take care not to make typing errors.

Edit the GRUB configuration file and remove any **splash image** entries. If these entries are not removed GRUB 0.90 behaves very oddly, transferring control between the serial console and the attached monitor and keyboard.

If there is not already a **password** command in the GRUB configuration file then create a hashed password, see Figure 4-4. The password should be good, as it can be used to gain root access.

Figure 4-4. Using md5crypt to create a hashed password for GRUB

```
grub> md5crypt
Password: **********
Encrypted: $1$U$JK7xFegdxWH6VuppCUSIb.
```

Use that hashed password in the GRUB configuration file, this is shown in Figure 4-5.

Figure 4-5. GRUB configuration to require a password

```
password --md5 $1$U$JK7xFegdxWH6VuppCUSIb.
```

Define the serial port and configure GRUB to use the serial port, as shown in Figure 4-6.

Figure 4-6. GRUB configuration for serial console

```
serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
terminal serial
```

`--unit` is the number of the serial port, counting from zero, unit 0 being `COM1`.

Note that the values of `--parity` are spelt out in full: `no`, `even` and `odd`. The common abbreviations `n`, `e` and `o` are *not* accepted.

If there is mysteriously no output on the serial port then suspect a syntax error in the **serial** or **terminal** commands.

If you also want to use and attached monitor and keyboard as well as the serial port to control the GRUB boot loader then use the alternative configuration in Figure 4-7.

Figure 4-7. GRUB configuration for serial console and attached monitor and keyboard console

```
password --md5 $1$U$JK7xFegdxWH6VuppCUSIb.
serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
terminal --timeout=10 serial console
```

When both the serial port and the attached monitor and keyboard are configured they will both ask for a key to be pressed until the timeout expires. If a key is pressed then the boot menu is displayed to that device. Disconcertingly, the other device sees nothing.

If no key is pressed then the boot menu is displayed on the whichever of `serial` or `console` is listed first in the **terminal** command. After the timeout set by the **timeout** the default option set by **default** is booted.

Figure 4-8. GRUB output to default device when configured for serial and attached monitor output

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

    GRUB  version 0.90  (639K lower / 162752K upper memory)
```



```
 +-------------------------------------------------------------------
----+
 | [ Red Hat Linux (2.4.9-21)   ]
|
 |
|
 |
|
 +-------------------------------------------------------------------
----+
      Use the ^ and v keys to select which entry is highlighted.
      Press enter to boot the selected OS or 'p' to enter a
      password to unlock the next set of features.

   The highlighted entry will be booted automatically in 10 seconds.
```

Summary

Installation of grub2 does not change the default Grub Legacy boot loader in any way and may be installed alongside it. There is a README. Mageia in the package which explains it's basic use, and how to add grub2 to a grub legacy menu for testing.

3) User Interfaces (GUI AND CUI)

CUI (Composite User Interface) is a compound type of user interface which incorporates the features of the GUI (Graphical User Interface) and the CLI (Command Line Interface) by realizing both interfaces in one area at the same time. It is absolutely substantial that the CUI contains GUI elements and control language in the same area of the screen at the same time, so the incorporation is complete. CUI is not an optional, alternatively selectable combination of the two communication method but a truly new approach.

An other relevant feature of the CUI method is that it uses generated commands. When the user activated some GUI part of the user interface an automatically generated command appears to complete the task so the user can observe how the CLI part of the user interface can be used.

B.C.A.-4(UNIX.)

Bachelor of Computer Application
(Semester – IV)
Saurashtra University

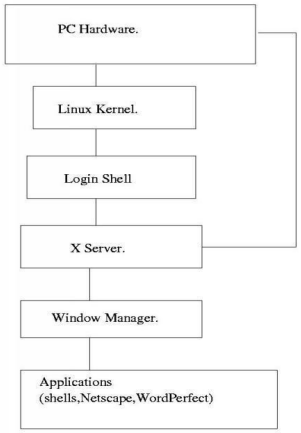
No.	Topics	Details
10.	**Working with X-windows(Ubuntu)**	->Layered Structure of X -Windows Manager -Desktop Environment -User Configuration -Start X command ->Window Manager -GNOME -KDE -Purpose of Window Manager ->KDE DESKTOP

Chapter No. 10
Working With X-Windows (Ubuntu)

- **Layered Structure of X**

1) Window Manager:

A window manager is the piece of software responsible for managing the different windows that appear on your screen. The window manager controls the placement of windows, draws the borders and scrollbars, and is responsible for ensuring that your X-Windows programs get along. A window manager is usually started after the X server finishes loading, which is the application responsible for drawing everything onto your screen. It is the window manager; however, that controls much of the look and feel of your X-Windows desktop. Usually window managers also allow you to launch applications. The attached diagram shows how the pieces fit together. Window managers sit between the X-Server and X-Windows applications, controlling the placements and look and feel of windows. It is possible to run X-Windows applications without a window manager, but it is rather crude and limiting.



Summary

X Window System's architecture, give a better understanding of its design, which components integrate with X and fit together to provide a working graphical environment and what choices are there regarding such components as window managers, toolkits and widget libraries, and desktop environments.

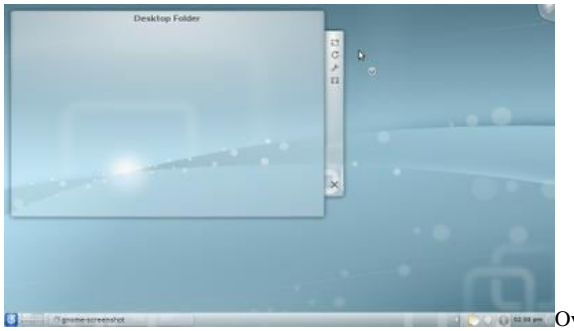
2) Desktop Environment:

The most popular desktop environment currently in use is Gnome; it is the desktop environment that powers the three most popular desktop Linux distros (Ubuntu, Fedora, and Linux Mint). Gnome is a fully developed desktop environment that provides a fully integrated application set. It is easy to use and provides GUI tools for making edits to all the different features that are available within it. It is a very "user friendly" desktop environment that is fantastic for new users.

Gnome's memory foot print is modest for all the features it provides. A default Gnome installs uses around 180megs of RAM. If you like eye candy on your computer odds are you will want to be running compiz (desktop effects) on your Gnome desktop. A default Gnome install with compiz running uses slightly more memory, right about 205megs.

KDE:

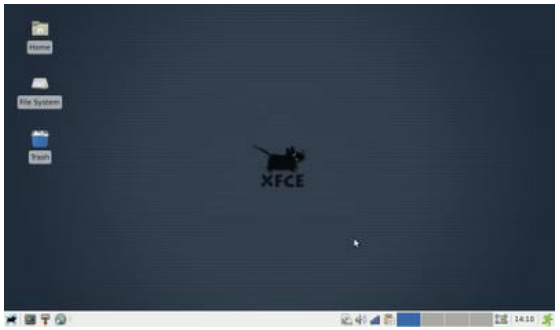
In terms of popularity KDE is the second most popular desktop environment. Like Gnome it is fully matured and provides it's own full application set as well as GUI tools for configuration. KDE also has a wide selection of "plasma widgets", which are handy applets you can place all around your desktop for all sorts of tasks. They range from something as practical as a calculator to as useless as a display from "The Matrix".



Over all KDE is much more customizable than Gnome, but this comes at a cost. A default KDE install uses around 510megs of RAM, if you are looking for eye candy it does not cost as much to run kwin (KDE's built in desktop effects) as it does to run compiz. With kwin enabled a default KDE install uses around 520megs of RAM.

XFCE:

XFCE is designed to be simplistic and quick. It does not provide much in the way of eye candy (although you can run compiz on it) but, it is a decently fast/responsive desktop environment. While XFCE does have some of it's own applications, such as it's file manager Thunar and the XFCE system monitor, it does still borrow some applications from the Gnome environment (such as nm-applet network manager). Don't think XFCE is an immature project though, what it borrows from Gnome is more to save itself from reinventing the wheel than from lack of ability. XFCE does not have quite as many tools for making GUI edits as Gnome or KDE, but it does have a fairly good configuration panel.



Designed to be quick and lightweight XFCE leaves a low memory footprint on the system you have it running on. At defaults XFCE uses around 140megs of RAM.

LXDE:

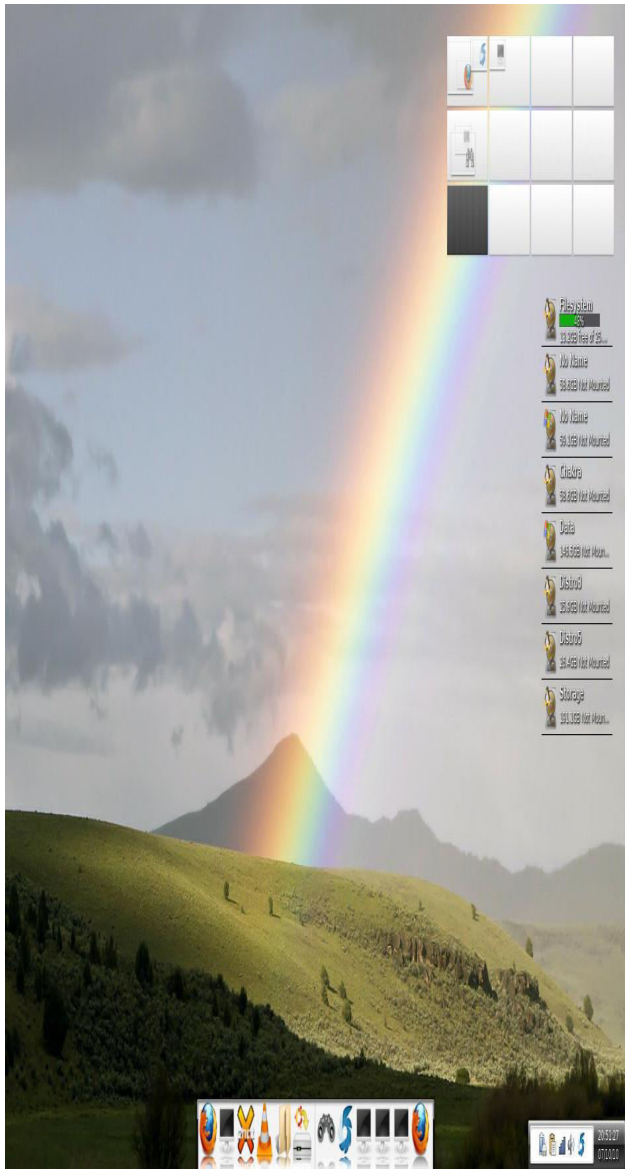
LXDE is a newer project in the world of Linux desktop environments. Similar to XFCE, LXDE's goal is to provide a fast, lightweight desktop environment with little resource usage. LXDE has a few of it's own applications, but those applications it still lacks it borrows from Gnome and XFCE. The age of the LXDE project really shows when you start to look into making customizations to things. Many adjusts have to still be made by manually editing configuration files, not a bad thing if you know your system well (or are willing to learn it), but this can be a giant set back for a beginner who wants things to "just work".

LXDE may be a much younger project than XFCE, but it does a fantastic job of resource conservation. A fresh install of LXDE uses 100megs of RAM, the lowest of all the desktop environments I am reviewing today.

E17:

I cannot do an overview of Linux desktop environments without mentioning E17. E17 is designed to be a lightweight, but elegant desktop environment. It is very successful at both of these tasks.

E17 uses all of its own libraries, which have been built from the ground up for speed and flexibility. E17 is a tinkerer's delight; You can customize and change.



A base install of E17 leaves a memory footprint of around 110megs of RAM. Now, while a base install is functional, half the fun of E17 is playing with widgets, changing transitions, and generally toggling everything you can just to see how shiny you can make your desktop. After I had my E17 fully configured it's memory footprint was increased to a whopping 120megs of RAM.

A few of you may be wondering if E17 is so lightweight, flexible, and flashy why more distros don't opt to use it for their desktop environment. There are two reasons for this. First and foremost is the fact that E17 is very much "beta" software. Compiling the latest version from source at any given point can have piles of crashes/defaults that can make using it a giant headache. Second, if you do take the time to piece together a stable E17 build (check out Elive or Bodhi for two good examples of what E17 can do) the desktop environment takes some getting used to. For many it will feel foreign regardless if they are used to using another Linux desktop environment or a Windows machine.

Summary

Desktop Environment is desktop environment that is simple, fast with good look and feel. It use FLTK2 GUI library. It should work fine on slower computers too.

3) User Configuration:

well, a simple way may be to go to system -> administration -> user and groups.....first, edit the users and check they have permissions, etc.... if they are trying to run things installed only for certain users, they will not have the permissions....such as certain icon themes, etc...

Summary

It is a simple technique to User configuration.

4) Start X Command:

Purpose

It Initializes an X session.

Syntax

startx [**-d** *Display*:*0*] [**-t** | **-w**] [**-x** Startup | [**-r** *Resources*] [**-m** *Window Manager*]] [**-wait**]

Description

The **startx** command streamlines the process of starting an X session.

Summary

The command does the following:

- Sets the user's **DISPLAY** environment variable to identify the X server to the X clients
- When run from a workstation, starts the X server
- Starts the X clients.

- **Windows Managers**

1) GNOME:

GNOME is a desktop environment and graphical user interface that runs on top of a computer operating system. It is composed entirely of free and open source software and is developed by both volunteers and paid contributors, the largest corporate contributor being Red Hat. It is an international project that includes creating software development frameworks, selecting application software for the desktop, and working on the programs that manage application launching, file handling, and window and task management.

Summary

GNOME is part of the GNU Project and can be used with Linux-only operating systems.

2) KDE:

KDE is an international free software community. producing an integrated set of cross-platform applications designed to run on Linux, FreeBSD, Solaris, Microsoft Windows, and OS X systems. It is known for its Plasma Desktop, a desktop environment provided as the default working environment on many Linux distributions, such as opens USE, Mageia and Kubuntu and is default desktop environment on PC-BSD a BSD operating system.

The goal of the community is to provide basic desktop functions and applications for daily needs as well as tools and documentation for developers to write stand-alone applications for the system. In this regard, the KDE project serves as an umbrella project for many standalone applications and smaller projects that are based on KDE technology. These include Calligra Suite, digiKam, Rekonq, K3b, and many others.

KDE software is based on the Qt framework. The original GPL version of this toolkit only existed for the X11 platform, but with the release of Qt 4, LGPL versions are available for all platforms. This allows KDE software based on Qt 4 to also be distributed to Microsoft Windows and OS X.

Summary

KDE Platform consists of the libraries and services needed to run KDE applications.

3) Purpose of Window Manager:

Window managers control the behavior of windows and how they respond to the user. At minimum they at least provide **focus control so that an application** will know you're talking to it. A window manager is also concerned with how to place or move windows and sizes them. A tiling window manager will place windows in a grid and will only allow a user to create more frames and place windows in them, which will result in a highly controlled placement and size for the windows. **Windows cannot typically overlap**. The disadvantage of this approach being a generally **less user-friendly desktop** hostile to using the mouse with windows taking up as much real estate as they can without intersecting other Windows, meaning space will be of short supply unless you don't mind some Windows getting much tinier. A stacking window manager is like what **older Windows** used to be: You're allowed to overlap windows all you like, size them as you wish, and place them where you want. A dynamic window manager allows the user to **switch between a tiling mode** and a stacking mode. Most modern desktops today, however, use a compositing window manager, which enables advanced window management techniques and enhanced visual effects; it usually takes on many of the disadvantages of stacking window managers (Reliance on the desktop's panel for most events.), solving some others (Thumbnail previews, "live" task switching where the windows themselves seem to transform themselves on-screen.), and introducing its own, including: A much heavier memory footprint, requiring somewhat more advanced hardware to work to its fullest. They can be slow if you use more eye candy than window management enhancements. And some compositing window managers on Linux will not work without OpenGL, which can mean trouble with some drivers that doesn't really have a considerable support for OpenGL. Most Linux desktop environments use compositing window managers, but are able to **switch compositing on or off as needed, or load alternative window managers if their default choice is incompatible**.

Summary

- ✓ Window managers control the behavior of windows and how they respond to the user.
- ✓ Windows cannot typically overlap.

- **The KDE Desktop:**

Practical method/Desktop View

THANK YOU