

2022

# Foundation Course

## Table of Contents

Table of Contents.....	2
Learning Outcome .....	5
<b>Introduction .....</b>	<b>6</b>
<b>Chapter 1: Introduction to Linux Operating System.....</b>	<b>11</b>
1.1 Introduction to Operating System.....	11
1.2 Introduction to Linux .....	13
1.3 Installation of Ubuntu.....	20
1.4 Linux Commands.....	30
1.5 Working on different text editors: nano, vi .....	35
1.6 Shell Scripting .....	39
1.7 Managing Linux Files.....	42
1.8 SSH Command.....	46
<b>Chapter 2: Introduction to Database.....</b>	<b>49</b>
2.1 Introduction to DBMS .....	49
2.2 Introduction to RDBMS.....	54
2.3 Database Schema .....	56
2.4 What is MySQL? .....	62
2.5 What is DML? .....	65
<b>Chapter 3: Building Front Face for the Web .....</b>	<b>68</b>
3.1 Understanding the Web.....	68
3.2 HTML for web Layout .....	78
3.3 CSS for Page Design .....	108
3.4 JS for Client-side scripting.....	130
<b>Chapter 4: Introduction to JAVA Programming Language.....</b>	<b>148</b>
4.1 Introduction to JAVA Programming.....	148
4.2 Understanding the OOPS Concepts .....	167
4.3 Java Fundamentals .....	177
4.4 Java Control Statements .....	198
4.5 Java Arrays .....	210
4.6 Java Class and Objects.....	214
4.7 Important Topics of OOPS Concept .....	244
4.8 Exception Handling.....	273
4.9 Additional Topics .....	282
<b>Chapter 5: Understanding the Cloud Computing .....</b>	<b>293</b>
5.1 Introduction to cloud computing .....	294

5.2 Introduction to Microsoft Azure Cloud Platform .....	295
5.2 Fundamentals of Networking and Networking Protocols .....	316
5.3 Virtual Private Network (VPN) .....	330
5.4 DHCP .....	340
5.5 Azure Virtual Networks .....	342
5.6 Market and Job Trends.....	355

**This course booklet has been designed by Edunet Foundation for the Tech-Saksham programme in partnership with Microsoft and SAP**

## Learning Outcome

After completing this handbook, the learner will have the knowledge on:

- Basics of working with Linux OS
- Shell Scripting in Linux
- Fundamentals of Databases
- Internet, Domains, Networking
- Understanding architecture of a website
- Core JAVA Programming and Coding Skills
- Cloud Computing environment and its usage in various application areas
- Networking, how to configure and manage cloud native networking services

# Introduction

Artificial Intelligence (AI) is the defining technology of our time. It is the branch of computer science that deals with the study and design of intelligent agents that perceive their environment and take actions that maximize their chances of success. John McCarthy, also known as the father of AI has described AI as “*The science and engineering of making intelligent machines, especially intelligent computer programs*”. It is human ingenuity that has made AI powerful. AI today enables us to build amazing software, which can improve health care, enable people to overcome their physical disadvantages, create incredible entertainment experiences, and empower smart infrastructure to name a few.

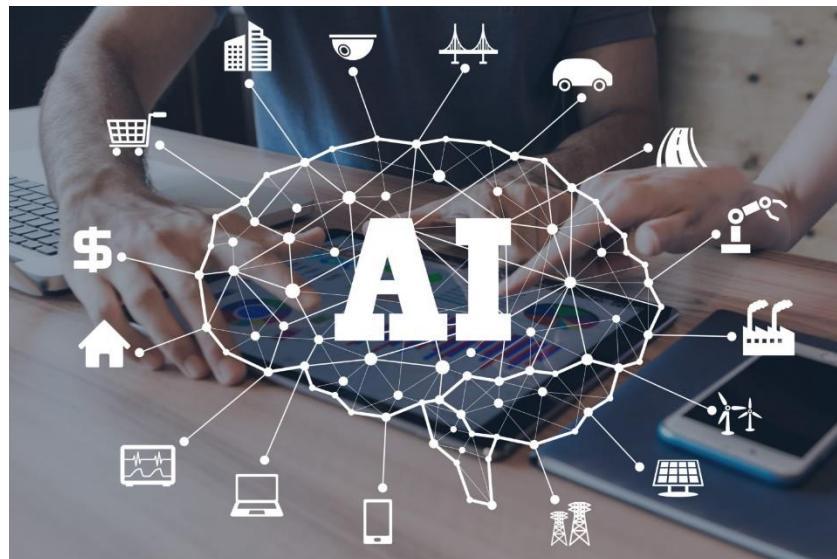


Image Source: <https://www.webopedia.com>

Some common examples of AI are facial recognition which grants machines the ability to identify human faces and objects in pictures and videos. Another example is Natural Language Processing, which gives the computer the capability to interpret written or spoken language, and respond in kind. This is what we encounter in Alexa or Cortana.

Data is the most important ingredient of AI. We need to have rich datasets to run our models. AI practitioners need to have an infrastructure environment, i.e., CPU, memory, disk, so that they don't lose time waiting for an infrastructure team to prepare it. This is where cloud computing comes in, as it helps AI data scientists to use data science services including jupyter notebooks, store the large data sets on the cloud to work on them efficiently. Cloud computing provides developers with storage servers applications and programs on an on-demand basis via the internet and they get these resources kind of similar to how we get our electricity.

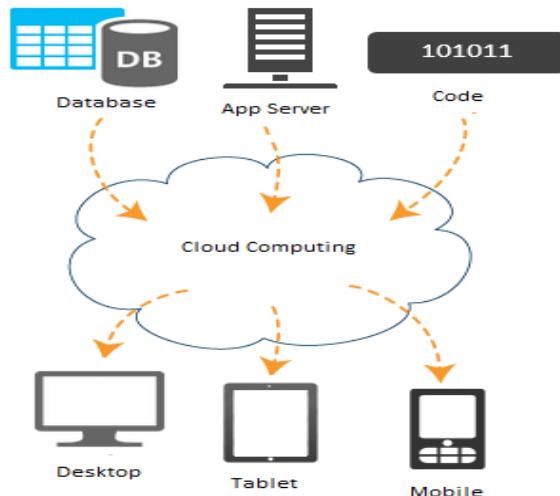


Image Source: <https://www.dotnettricks.com>

We do not build massive power structures in our backyards, instead we rely on a centralized power grid. We use whatever electricity we need as per our demand basis and pay accordingly. Similarly, developers/users use whatever resources they need on a pay-as-you-go on demand basis. The resources include servers, storage, databases, networking, software, and analytics.

AI capabilities in the cloud computing environment will help in making different operations (business, healthcare, education, entertainment) more efficient, strategic, and insight-driven while also providing additional flexibility, agility, and cost savings. The services offered by these operations have to be moved online, as we are all living in a digital world. Having a website is very crucial in the field of education, medical, finance, small businesses, and mid-sized or large-sized businesses because each website type has its own role to play in the current market in order to establish business and attract prospects. The more information users have about an organization or service, the more likely they are to trust that company and be loyal to it. This is where web development comes in. Web development is an overall term used for all the work that goes into building a website.

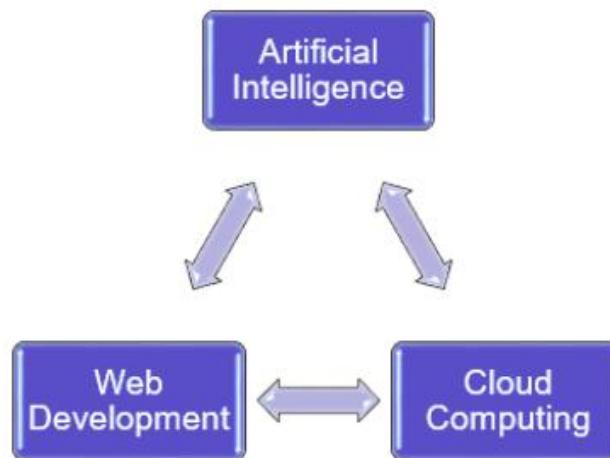


Image Source: <https://www.clariwell.in/>

Web Development includes everything from web markup and coding, to website development including all related developmental tasks, such as client-side scripting,

server-side scripting, etc., to make a fast, secure, and engaging website. The web developers aim to provide the customers with a highly personalized and enhanced experience. They have to work on the front-end (client-side) of the website allowing users to interact with a website and play videos, expand or minimize images, highlight text, and more. Web development also involves working on the back-end (server-side), which involves building and maintaining the mechanisms that process data and perform actions on websites. It involves data storage, security, and other server-side functions that we cannot see.

In our day-to-day scenarios, the three technologies interplay with each other leading to the smooth functioning of different platforms, be it business, education, or healthcare. The most famous example will be the e-commerce website, which uses AI tools like voice-based search results and AI-based chatbots to improve user experience. The first task goes into building the websites, working on the front as well as the back-end. The next task is implementing AI tools, whether it is a product recommendation software or an AI chatbot assisting in responding to queries of users, etc. The applications and services related to the website are developed, without worrying too much about scalability and deployment issues, i.e., whether the e-tracking website will crash with heavy usage or not be available in a particular location. This is where cloud computing platforms provide us with a stack of solutions that can ease and speed up the development cycle letting the developers consume resources as a service and pay only for what they use most of the time. The applications developed using cloud computing offer inherent scalability and also ensure high availability.



In the general media, AI, web development, and cloud computing are often associated with technologies like bitcoin, or some futuristic robots, but the potential of this combination is really much more and today it has mostly revolutionized the business and humanitarian aspects of our society. Let us take another example of Airbnb. Airbnb is one of the world's largest hoteling services but they don't own a single piece of property and they do this by creating an application that uses cloud computing to link their customers with hosts that have space to share. The official website of Airbnb helps increase the company's credibility and helps establish an image letting the audience know who they are and what they represent or offer. AI is used by Airbnb to perform background checks on guests in order to ensure the safety of its hosts. Airbnb

also uses smart pricing machine-learning algorithms to help its hosts set competitive rates that maximize occupancy rates. This idea of Airbnb was started by two guys in an apartment, but now it is a multibillion-dollar company and can compete with some of the largest hoteling services in the world. They get all the data and storage from the cloud and don't have to build their own data centers to store all their data for the smooth transition of their business. These technologies have given the people associated with small businesses to show off their AI-implemented technology with little resources on their website, as cloud services work on a pay-as-you-go basis.

This course will prepare you with the foundation required to learn either of the three technologies discussed above and then implement them. The first module of this course will be covering Linux, which is one of the most popular operating systems for Information Technology employees, whether you are working as a system administrator, network engineer, or IT technician. Linux with its powerful shell is a friendly operating system for software developers.

Data, as we know, is the new oil. It is the base of AI, cloud as well as web. Data is valuable, but if unrefined, it cannot really be used. A database is an organized collection of structured information or data. Databases are probably the **most** important part of an application and are discussed in Module 2. A professional full-stack developer should definitely know a lot more about databases and their different types.

In module 3, we will learn about the basics of web development. HTML or Hyper Text Markup Language is basically the backbone of the world wide web. Every page you see while surfing the internet has HTML involved. HTML provides structure to your pages. CSS or Cascading Style Sheets supplements HTML by controlling how a page should appear. CSS is mainly used for setting colors, backgrounds, and appearance of the page. Without CSS, building web pages can become almost impossible. While HTML and CSS together help in defining what a web page should look like, JavaScript adds life to those pages. JavaScript allows you to make your web pages interactive and is really essential to making fully featured web applications.

In module 4, core java will be discussed in details. Java is a vast language, with lots of variants and frameworks to choose from. Core Java is referred to the central components of the Java language—the thing that people use to write the frameworks and has developed the cottage industry around Java. Understanding the basics of Java will give you a significant advantage when learning all of the related tools built on top of it, such as Android app development, video game development, desktop GUI applications, or just general software development.

Knowledge about cloud platforms will allow you to manage your servers and run-time environments for applications. It will also greatly increase your demand in the market. So, in Module 5 we will study the popular cloud platform that is Microsoft Azure. Azure is the cloud computing service offered by Microsoft that helps to build, test, deploy and manage applications and services through its globally present datacenters. Nowadays companies are managing their worldwide business through the cloud, applications can be hosted near to their user base and users can access the application with good performance and low latency. For instance, you can live in India and manage your business anywhere in the world using cloud computing. Therefore, you will learn

various skills that are needed to start your journey in cloud computing. You will be learning about networking concepts that are the foundation for creating cloud-native virtual networks. You will learn to create and prepare your virtual machine for the development environment that can be used to develop web applications.

# Chapter 1: Introduction to Linux Operating System

## Learning Outcomes:

- Understand Linux Operating System and Kernel
- Learn the Basic Linux Commands
- Explore the nano, and vi editors
- Understand key concepts of shell scripting and ssh

## 1.1 Introduction to Operating System

An operating system (OS) is a software program that acts as an interface between the computer hardware components and the user. It helps in the management of the computer's memory and processes and also handles the data generated by the software. It helps us to communicate with the computer without us having to know, how to speak the computer's language. Modern operating systems use a graphical user interface (GUI), where everything is displayed on the screen and lets us use our mouse to click icons, menus, and buttons.

The operating system is mainly designed to serve two basic purposes:

- 1) It mainly controls the allocation and use of the computing system's resources among the various user and tasks.
- 2) It provides an interface between the computer hardware and the programmer that simplifies and makes feasible for coding, creation of application programs and debugging.

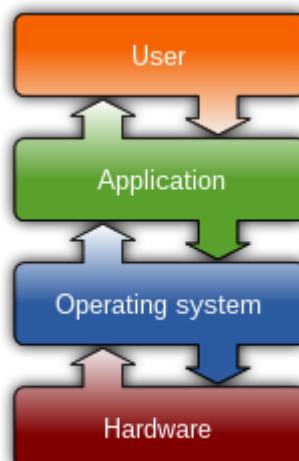


Image source: [Wikipedia](#)

The operating system provides services to programs and the users of the system. The common management tasks that an operating system perform include:

- 1) **Process management:** It involves putting the tasks in order and pairing them into manageable size, before they go to the CPU.
- 2) **Memory management:** It consists of coordinating data to and from the Random Access Memory and determines the necessity for virtual memory.
- 3) **Device management:** It provides an interface between the connected devices. It also performs the task of allocation and de-allocation of the devices.
- 4) **File management:** It manages all the file-related activities such as organization storage, retrieval, naming, sharing, and protection of files.
- 5) **Storage management:** Systems have several levels of storage which include primary storage, secondary storage, and cache storage. It directs permanent data storage.
- 6) **User Interface management:** It allows you to communicate with your computer. It also sees to the coordination and assignment of compilers, interpreters, and another software resource of the various users of the computer systems.
- 7) **Security management:** The security module protects the data and information of a computer system against malware threat and authorized access.
- 8) **Job accounting:** It keeps track of time and resource used by various job and users.

## History of Operating System

Operating systems were first developed in the late 1950s to manage tape storage. The computers before them were generally used to solve simple math calculations and did not need any operating system. All programming was done in absolute machine language, by wiring up plugboards to control the machine's basic functions. Operating systems in the 1950s were called single-stream batch processing systems because the data was submitted in groups. These new machines were called mainframes, and they were used by professional operators in large computer rooms. By the late 1960s operating systems designers were able to develop the system of multiprogramming in which a computer program was able to perform multiple jobs at the same time. The operating systems from the 1980s saw the creation of personal computing. These personal computers would cost a very small fraction of what mainframe computers cost. A personal computer was so affordable that it made it possible for a single individual to own one. The first OS built by Microsoft was DOS. It was built in 1981 by purchasing the 86-DOS software from a Seattle company. The present-day popular OS Windows first came to existence in 1985 when a GUI was created and paired with MS-DOS. Today all of our electronic devices run off of operating systems, from our computers and smartphones, to ATM machines and motor vehicles. And as technology advances, so do operating systems.

## Types of Operating System

We discuss below some of the popular types of operating systems.

- 1) **Batch Operating System:** In a Batch Operating System, the similar jobs are grouped together into batches with the help of some operator and these batches are executed one by one. The Batch OS can be shared between multiple users. Every user prepares his or her job on an offline device like a punch card and submit it to the computer operator. The overall time taken by the system to execute all the programmes is reduced.

- 2) **Time-Sharing Operating System:** A Time-sharing or multi-tasking OS system enables people located at a different terminal to use a single computer system at the same time. Here more than one processes are being executed at a particular time with the help of the time-sharing concept. Since equal time is given to each process, each process gets equal opportunity to execute.
- 3) **Distributed Operating System:** It uses many processors located in different machines to provide very fast computation to its users. These systems are connected to each other using a shared communication network. The best part about these Distributed Operating Systems is remote access i.e., one user can access the data of the other system and can work accordingly. The systems are connected with each other, therefore the failure of one system does not stop the execution of processes as resources are shared between each other. The load on the host computer also gets distributed and this, in turn, increases the efficiency.
- 4) **Real-time Operating System:** Real-time Operating Systems are used in the situation where we are dealing with some real-time data. The time interval to process and respond to inputs is very small. For example, the details of the temperature of the petroleum industry are very crucial and this should be done in real-time and in a very short period of time. A small delay can result in a life-death situation. So, this is done with the help of a Real-time OS.
- 5) **Embedded Operating System:** This OS is designed to perform a specific task for a particular device that is not a computer. For example, the software used in elevators is dedicated to the working of elevators only and nothing else. It is fast, costs less, consumes less memory and other resources as it is dedicated only to a particular job.

To summarize, we found operating system mainly acts as an interface between the hardware and the software. It is mainly used to hide the complexity of the hardware and makes the assessment to hardware easier without have to write large codes. The three most commonly used operating systems for personal computers are Microsoft Windows, macOS, and Linux. We will discuss about the Linux operating system in the next section.

## 1.2 Introduction to Linux

Linux is a Unix-like open-source OS. An operating system is said to be Unix-like if it supports multiple users, has a hierarchical file system, and behaves similar to UNIX OS based on the Linux kernel. Linux is freely available to everyone, due to it being an open-source operating system. It was created in the early 1990s by Linus Torvalds, a Finnish software engineer. Linus originally developed Linux as a free OS for Intel x86-based platforms but has since then been ported to more computer hardware platforms than any other operating system. Linux has today become one of the leading operating systems on servers and large-scale systems such as supercomputers and mainframe computers. The Android system which is widely used for smartphones is also built on the Linux kernel. Apart from these, Linux also works on embedded systems, i.e., the devices whose operating system is implemented in firmware, such as video games, watches, routers, television controllers, etc.

### Why we need Linux?

The main advantage Linux has over other operating systems is that it is free. Not only is the operating system free, but all the related applications are also available free of

cost. Linux being an open-source operating system, it allows a user to modify its source (even source code of applications) itself as per the user requirements.

Linux is more reliable when compared to other OS. Traditionally UNIX-like systems are known for running for years without a single failure or having a situation that demands a restart. Linux with its top-notch design and built-in security results in an un-parallel up-time. It is more secure and almost non-vulnerable. Linux does not require the use of commercial anti-virus/anti-malware packages. The Linux community is much more active and major as well as minor updates are released from time to time.

## Linux Basic Features

Following are some of the important features of Linux Operating System.

**Portable** - Portability means software can work on different types of hardware in same way. Linux kernel and application programs support their installation on any kind of hardware platform.

**Open Source** - Linux source code is freely available, and it is community-based development project. Multiple Teams works in collaboration to enhance the capability of Linux operating system and it is continuously evolving.

**Multi-User** - Linux is a multiuser system means multiple users can access system resources like memory/ ram/ application programs at same time.

**Multiprogramming** - Linux is a multiprogramming system means multiple applications can run at same time.

**Hierarchical File System** - Linux provides a standard file structure in which system files/ user files are arranged.

**Shell** - Linux provides a special interpreter program which can be used to execute commands of the operating system. It can be used to do various types of operations, call application programs etc.

**Security** - Linux provides user security using authentication features like password protection/ controlled access to specific files/ encryption of data.

## Linux Advantages

**1. Low cost:** You do not need to spend time and money to obtain licenses since Linux and much of its software come with the GNU General Public License. You can start to work immediately without worrying that your software may stop working anytime because the free trial version expires. Additionally, there are large repositories from which you can freely download high-quality software for almost any task you can think of.

**2. Stability:** Linux does not need to be rebooted periodically to maintain performance levels. It does not freeze up or slow down over time due to memory leaks and such. Continuous up-times of hundreds of days (up to a year or more) are not uncommon.

**3. Performance:** Linux provides persistent high performance on workstations and on networks. It can handle unusually large numbers of users simultaneously and can make old computers sufficiently responsive to be useful again.

**4. Network friendliness:** Linux was developed by a group of programmers over the Internet and has therefore strong support for network functionality; client and server systems can be easily set up on any computer running Linux. It can perform tasks such as network backups faster and more reliably than alternative systems.

**5. Flexibility:** Linux can be used for high performance server applications, desktop applications, and embedded systems. You can save disk space by only installing the components needed for a particular use. You can restrict the use of specific computers by installing for example only selected office applications instead of the whole suite.

**6. Compatibility:** It runs all common UNIX software packages and can process all common file formats.

**7. Choice:** The large number of Linux distributions gives you a choice. Each distribution is developed and supported by a different organization. You can pick the one you like best; the core functionalities are the same; most software runs on most distributions.

**8. Fast and easy installation:** Most Linux distributions come with user-friendly installation and setup programs. Popular Linux distributions come with tools that make installation of additional software very user friendly as well.

**9. Full use of hard disk:** Linux continues work well even when the hard disk is almost full.

**10. Multi-tasking:** Linux is designed to do many things at the same time; e.g., a large printing job in the background won't slow down your other work.

**11. Security:** Linux is one of the most secure operating systems. —Wall\$! and flexible file access permission systems prevent access by unwanted visitors or viruses. Linux users have the option to select and safely download software, free of charge, from online repositories containing thousands of high-quality packages. No purchase transactions requiring credit card numbers or other sensitive personal information are necessary.

**12. Open Source:** If you develop software that requires knowledge or modification of the operating system code, LINUX's source code is at your fingertips. Most Linux applications are Open Source as well.

## Components of Linux System

Linux Operating System has primarily three components.

- **Kernel** – Kernel is the core part of Linux. It is responsible for all major activities of this operating system. It consists of various modules and it interacts directly

with the underlying hardware. Kernel provides the required abstraction to hide low level hardware details to system or application programs.

- **System Library** – System libraries are special functions or programs using which application programs or system utilities access Kernel's features. These libraries implement most of the functionalities of the operating system and do not require kernel module's code access rights.
- **System Utility** – System Utility programs are responsible to do specialized, individual level tasks.

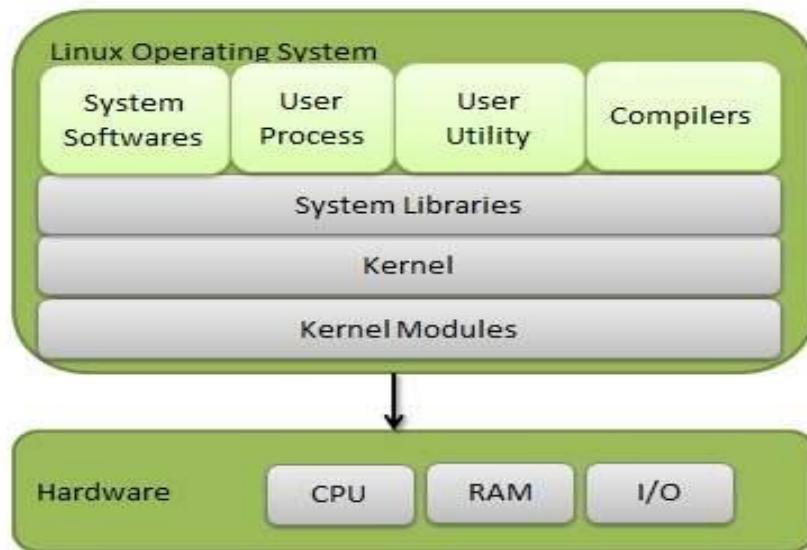


Image: LINUX Operating System [11]  
Reference: [https://www.tutorialspoint.com/operating\\_system/os\\_linux.htm](https://www.tutorialspoint.com/operating_system/os_linux.htm)

## Kernel Mode vs User Mode

Kernel component code executes in a special privileged mode called **kernel mode** with full access to all resources of the computer. This code represents a single process, executes in single address space and does not require any context switch and hence is very efficient and fast. Kernel runs each process and provides system services to processes, provides protected access to hardware to processes.

Support code which is not required to run in kernel mode is in System Library. User programs and other system programs work in **User Mode** which has no access to system hardware and kernel code. User programs/ utilities use System libraries to access Kernel functions to get system's low-level tasks.

## Linux Distribution (Operating System) Names

Linux has spawned several distributions over time, and has ensured that everyone from desktop users to Enterprise environments had something to be happy about. While the basic system/kernel is the same for all the distributions, the look and feel and the software ecosystem driving one Linux distribution to the other may be quite different. Each distribution includes the Linux kernel, the GNU shell utilities, the desktop environment, a package management system, an installer and other services. Many components are developed independently from each other and are distributed

in source code form. Linux being an open-source software, anyone can make their own Linux distribution by assembling it from the source code themselves, or by modifying an existing distribution. Currently, more than 300 Linux distributions are actively maintained. Here we list a few popular Linux distributions:

1. Redhat Enterprise Linux
2. Fedora Linux
3. Debian Linux
4. Suse Enterprise Linux
5. Ubuntu Linux

### *Common things between Linux & UNIX*

UNIX is an operating system developed around 1969 at AT&T Bell Labs by Ken Thompson and Dennis Ritchie. UNIX is not free and was created by Bell Labs for its own use, but it was licensed to other tech companies over the years. Linux was developed to work much like UNIX, but it is open source and freely available, unlike UNIX. Unix is not as flexible as Linux. It has less compatibility with different types of hardware. Unix installation requires strict and well-defined hardware machinery and works only on a specific CPU. Below we list some of the common applications shared by both Linux and UNIX:

1. GUI, file, and windows managers (KDE, Gnome)
2. Shells (ksh, csh, bash)
3. Various office applications such as OpenOffice.org
4. Development tools (perl, php, python, GNU c/c++ compilers)
5. Posix interface

## LINUX File system

We will in this section explore the Linux file system, getting to know how the directories are organised. Linux file structure files are grouped according to purpose. Ex: commands, data files, documentation. Parts of a Unix directory tree are listed below. All directories are grouped under the root entry "/". That part of the directory tree is left out of the below diagram.

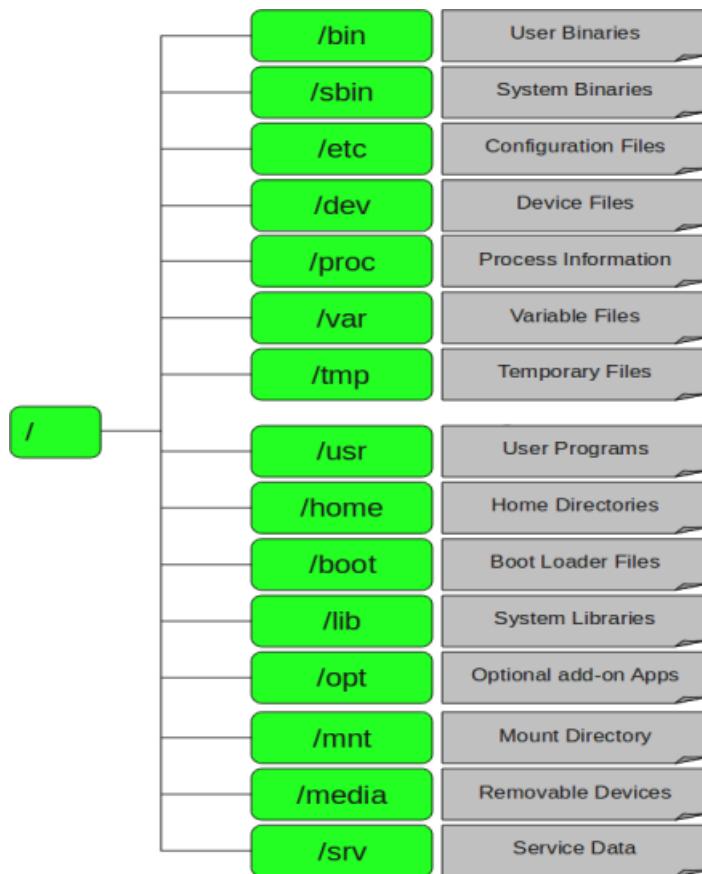


Image: LINUX File Structure

Reference: [https://www.tutorialspoint.com/operating\\_system/os\\_linux.html](https://www.tutorialspoint.com/operating_system/os_linux.html)

## 1. / – Root

- Every single file and directory starts from the root directory.
- Only root user has write privilege under this directory.
- Please note that /root is root user's home directory, which is not same as /.

## 2. /bin – User Binaries

- Contains binary executables.
- Common linux commands you need to use in single-user modes are located under this directory.
- Commands used by all the users of the system are located here.
- For example: ps, ls, ping, grep, cp.

## 3. /sbin – System Binaries

- Just like /bin, /sbin also contains binary executables.
- But, the linux commands located under this directory are used typically by system administrator, for system maintenance purpose.
- For example: iptables, reboot, fdisk, ifconfig, swapon

## 4. /etc – Configuration Files

- Contains configuration files required by all programs.

- This also contains startup and shutdown shell scripts used to start/stop individual programs.
- For example: /etc/resolv.conf, /etc/logrotate.conf

#### 5. */dev – Device Files*

- Contains device files.
- These include terminal devices, usb, or any device attached to the system.
- For example: /dev/tty1, /dev/usbmon0

#### 6. */proc – Process Information*

- Contains information about system process.
- This is a pseudo filesystem contains information about running process. For example: /proc/{pid} directory contains information about the process with that particular pid.
- This is a virtual filesystem with text information about system resources. For example:  
/proc/uptime

#### 7. */var – Variable Files*

- var stands for variable files.
- Content of the files that are expected to grow can be found under this directory.
- This includes — system log files (/var/log); packages and database files (/var/lib); emails (/var/mail); print queues (/var/spool); lock files (/var/lock); temp files needed across reboots (/var/tmp);

#### 8. */tmp – Temporary Files*

- Directory that contains temporary files created by system and users.
- Files under this directory are deleted when system is rebooted.

#### 9. */usr – User Programs*

- Contains binaries, libraries, documentation, and source-code for second level programs.
- /usr/bin contains binary files for user programs. If you can't find a user binary under
- /bin, look under /usr/bin. For example: at, awk, cc, less, scp
- /usr/sbin contains binary files for system administrators. If you can't find a system binary under /sbin, look under /usr/sbin. For example: atd, cron, sshd, useradd, userdel
- /usr/lib contains libraries for /usr/bin and /usr/sbin
- /usr/local contains users programs that you install from source. For example, when you
- install apache from source, it goes under /usr/local/apache2

#### 10. */home – Home Directories*

- Home directories for all users to store their personal files.
- For example: /home/john, /home/nikita

**11. /boot – Boot Loader Files**

- Contains boot loader related files.
- Kernel initrd, vmlinuz, grub files are located under /boot
- For example: initrd.img-2.6.32-24-generic, vmlinuz-2.6.32-24-generic

**12. /lib – System Libraries**

- Contains library files that supports the binaries located under /bin and /sbin
- Library filenames are either ld\* or lib\*.so.\*
- For example: ld-2.11.1.so, libncurses.so.5.7

**13. /opt – Optional add-on Applications**

- opt stands for optional.
- Contains add-on applications from individual vendors.
- add-on applications should be installed under either /opt/ or /opt/ sub-directory.

**14. /mnt – Mount Directory**

- Temporary mount directory where sysadmins can mount filesystems.

**15. /media – Removable Media Devices**

- Temporary mount directory for removable devices.
- For examples, /media/cdrom for CD-ROM; /media/floppy for floppy drives;
- /media/cdrecorder for CD writer

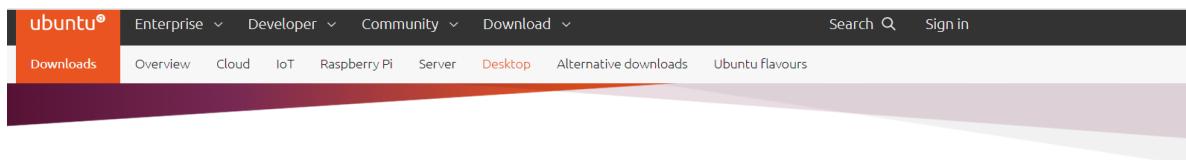
**16. /srv – Service Data**

- srv stands for service.
- Contains server specific services related data.
- For example, /srv/cvs contains CVS related data.

## 1.3 Installation of Ubuntu

### Downloading

Visit this link to <http://www.ubuntu.com/download/desktop> download Ubuntu.



The screenshot shows the Ubuntu website's download section. The top navigation bar includes links for Enterprise, Developer, Community, Download, Search, and Sign in. The 'Downloads' link is highlighted in orange. Below the navigation is a secondary menu with links for Overview, Cloud, IoT, Raspberry Pi, Server, Desktop (which is highlighted in orange), Alternative downloads, and Ubuntu flavours.

## Download Ubuntu Desktop



This screenshot shows the Ubuntu 20.04.2.0 LTS download page. It features a large green 'Download' button at the top right, which is outlined in red in the image. To the left of the button, there is descriptive text about the LTS version and its long-term support period. Below the text, there are 'Ubuntu 20.04 LTS release notes' and 'Recommended system requirements'. The system requirements list includes items like a 2 GHz dual core processor or better, 4 GB system memory, and 25 GB of free hard drive space.

You can select 32/64-bit versions as per your choice.

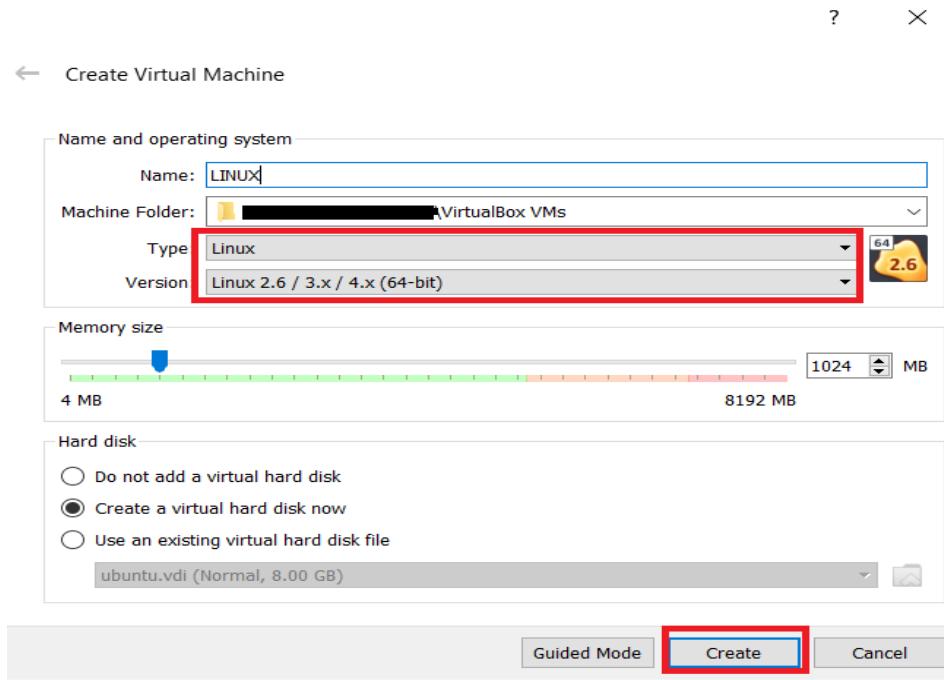
*Create a Machine in Virtual Box*

**Step-1** Open Virtual box and click on new button



**Step-2** In next window, give the name of your OS which you are installing in virtual box. And select OS like Linux and version as Ubuntu 32 bit. And click on next.

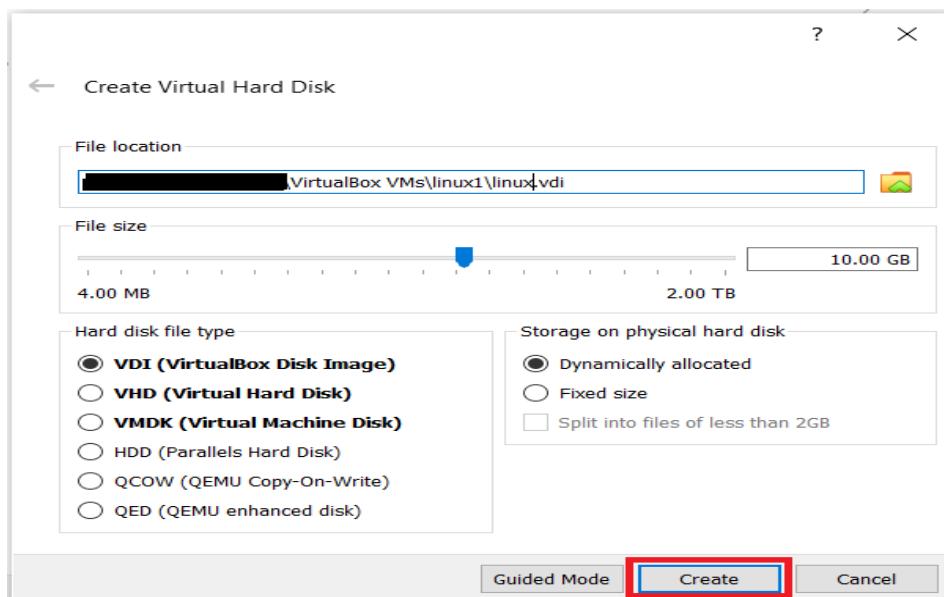
**Step-3** Now Allocate Ram Size To your Virtual OS. I recommended keeping 1024mb (1 GB) ram to run Ubuntu better. And click on next.



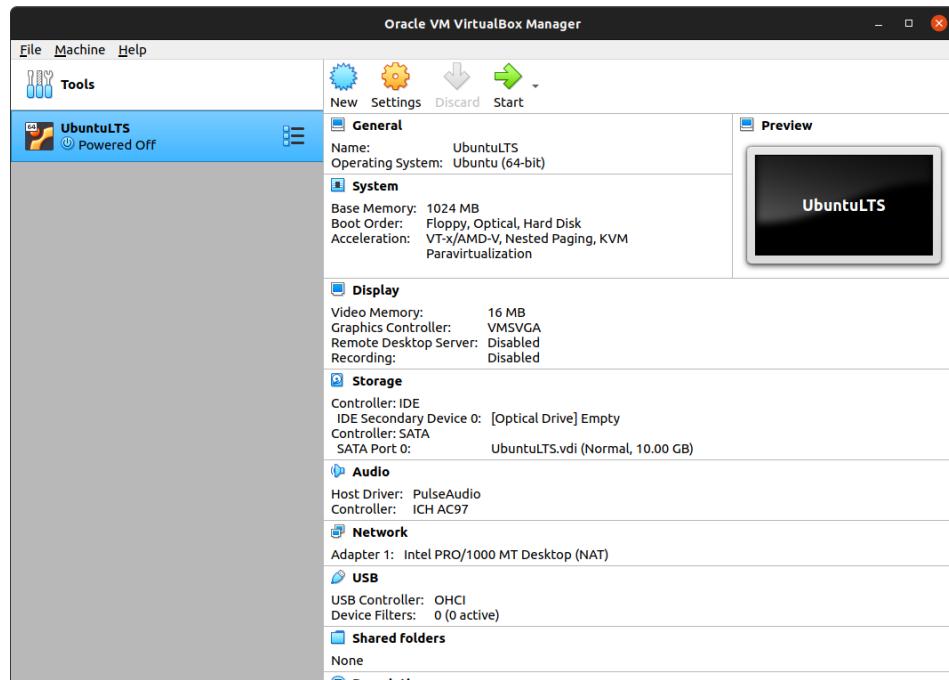
**Step-4** Allocate memory to your virtual hard drive, minimum 10 GB recommended.

**Step-5** Select VDI (virtual hard disk) option.

**Step-6** Click on dynamic allocated. This means that the size of the disk will increase dynamically as per requirement. and click on create button.



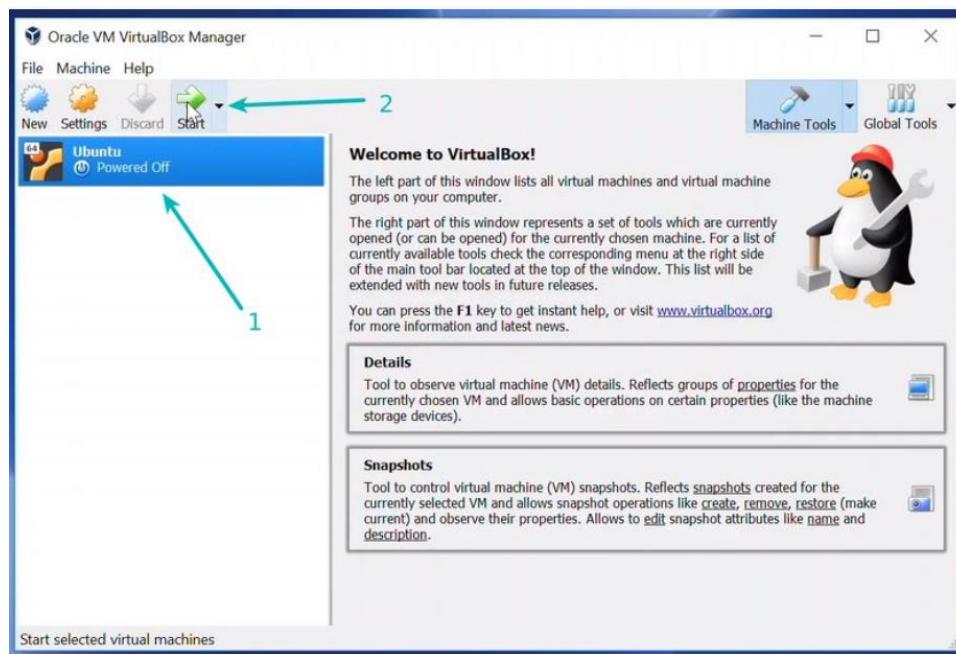
**Step-7** Now you can see the machine name in left panel



So, a Machine (PC) with 8GB Hardisk, 1GB RAM is ready.

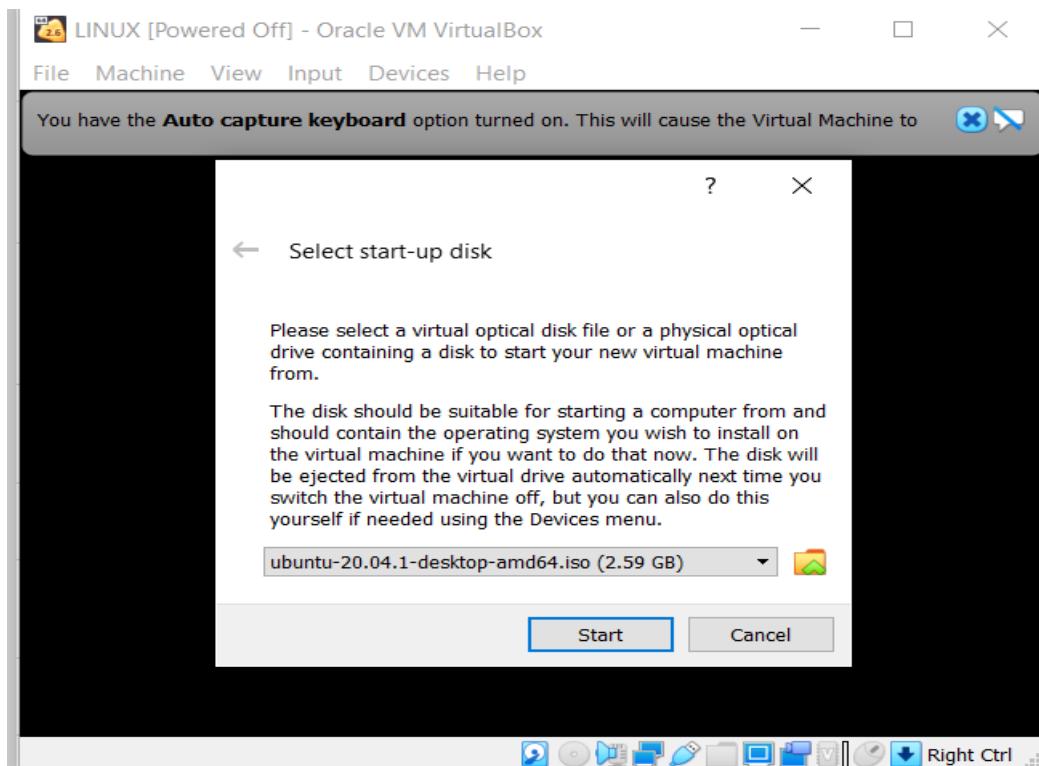
### *How to Install Ubuntu*

#### **Step-1** Select the Machine (1) and Click on Start (2)

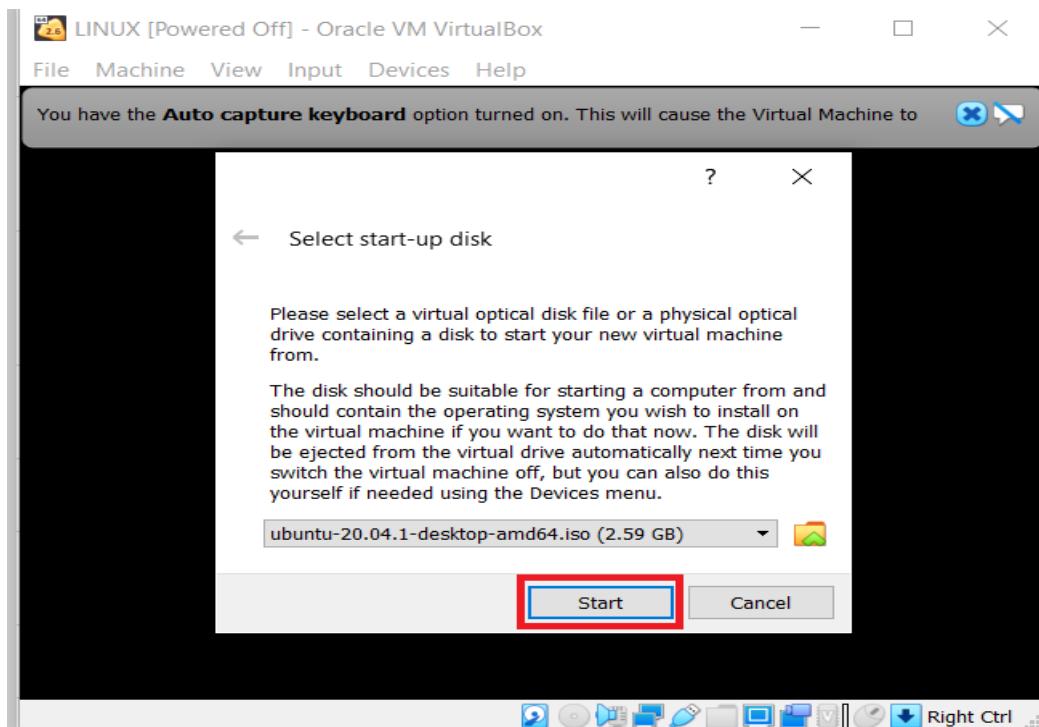


#### **Step-2** Select the Folder Option

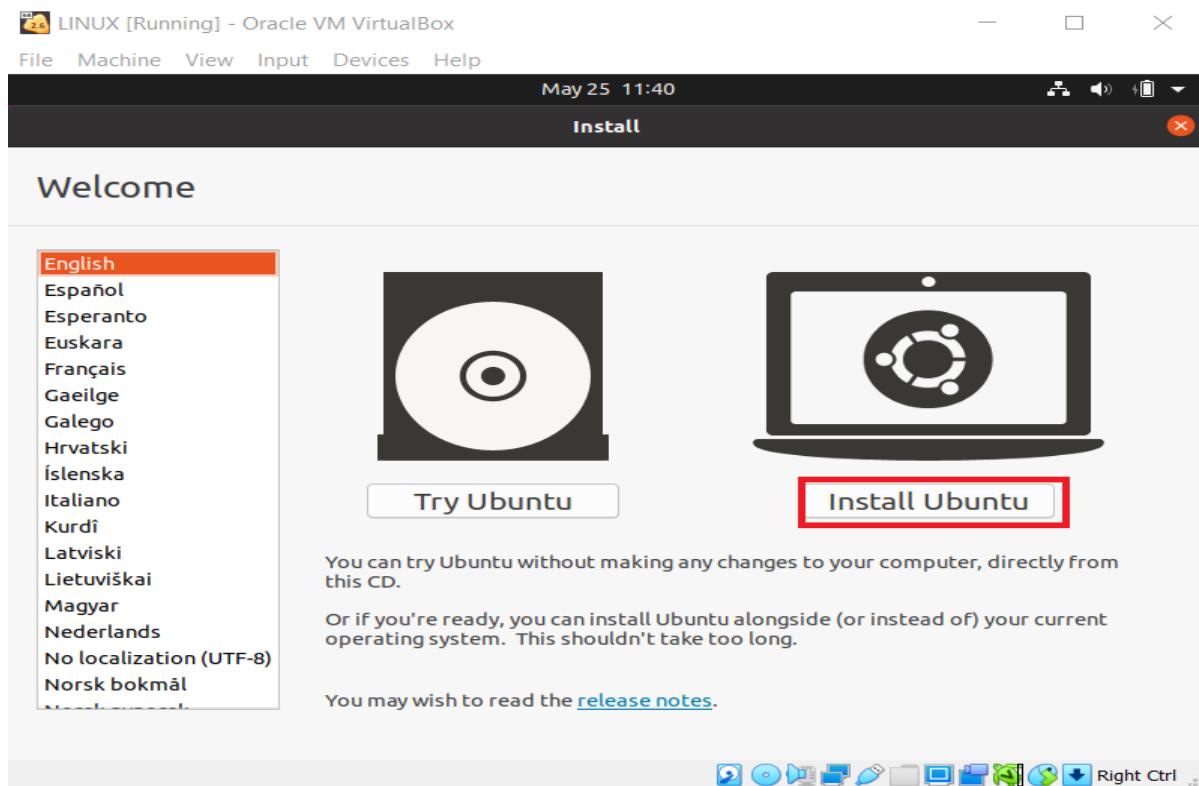
#### **Step-3** Select the Ubuntu iso file



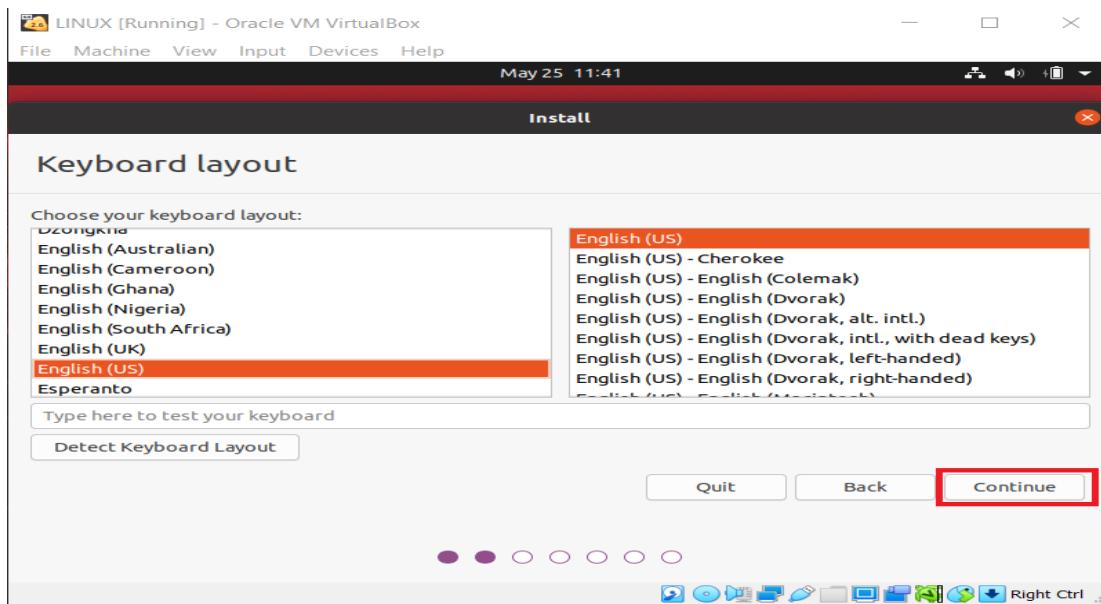
#### Step-4 Click Start



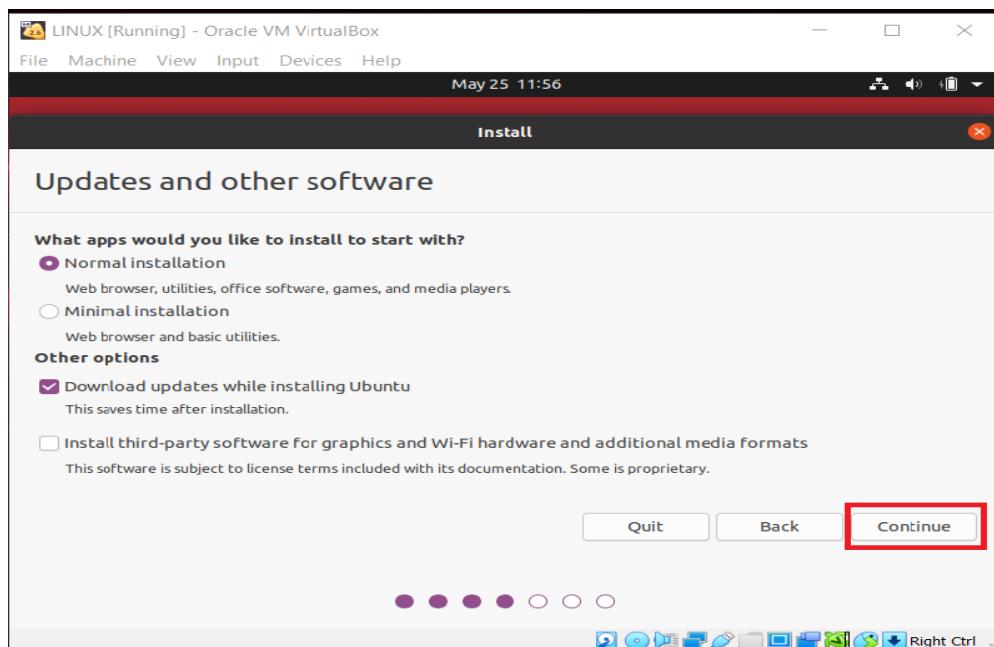
**Step-5** You have an option to Run Ubuntu WITHOUT installing. In this tutorial we will install Ubuntu



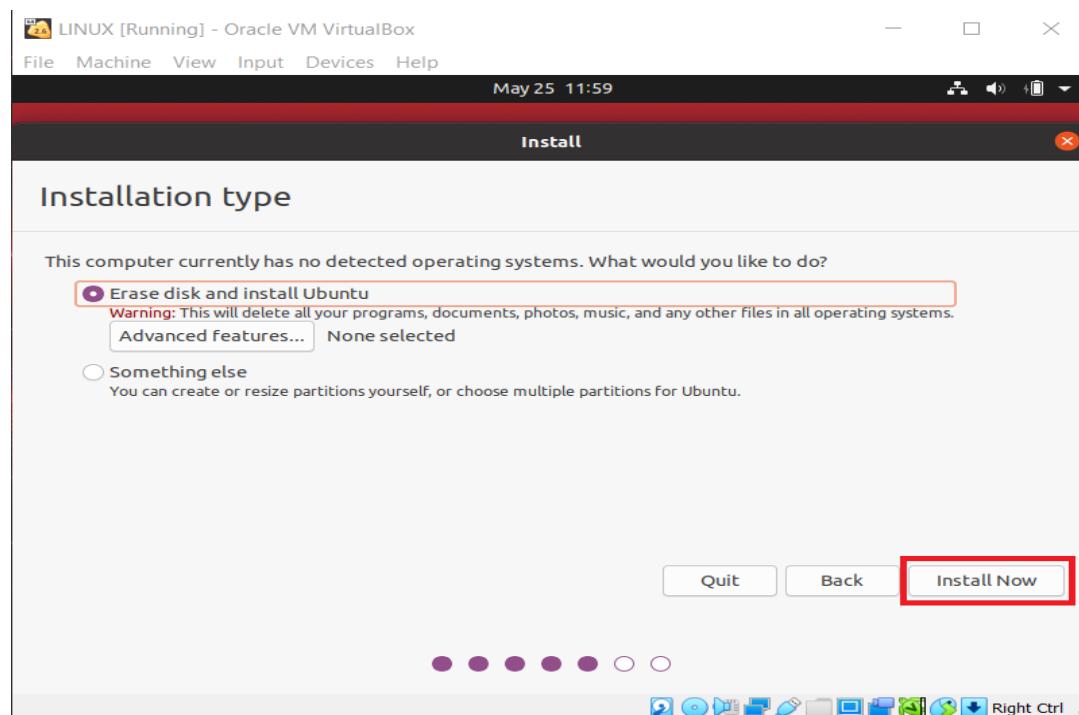
**Step-6** Select your keyboard layout, by default English (US) is selected but if you want to change then, you can select in the list. And click on continue



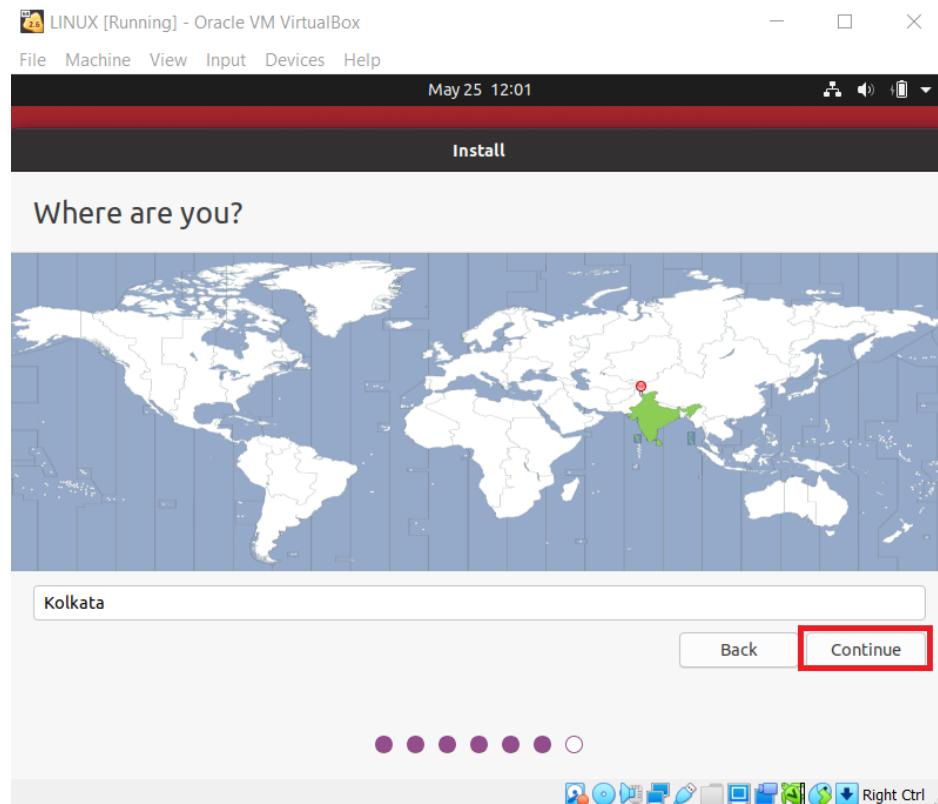
**Step-7** Click continue.



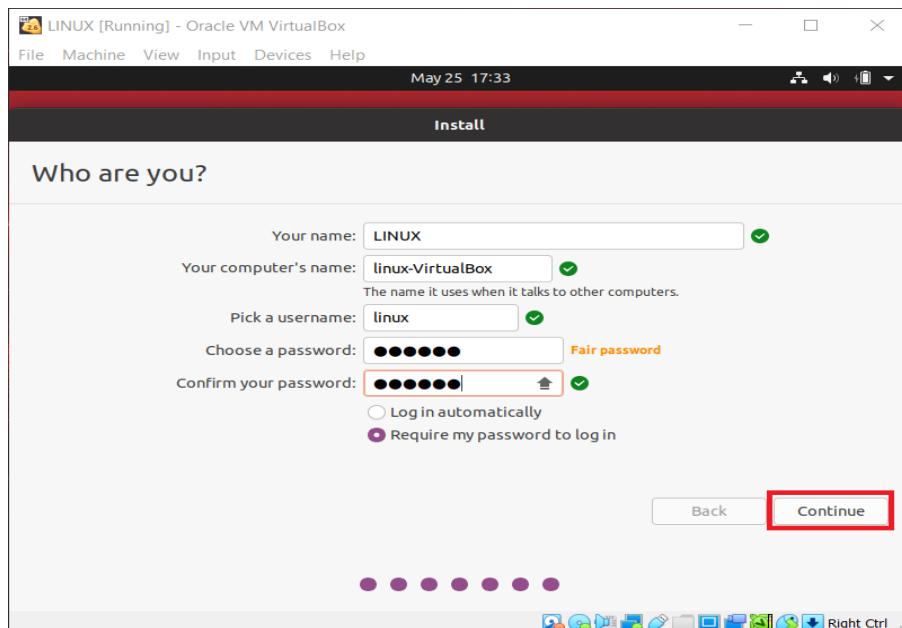
**Step-8** Select option to erase the disk and install Ubuntu and click on install now. This option installs Ubuntu into our virtual hard drive which is we made earlier. It will not harm your PC or Windows installation



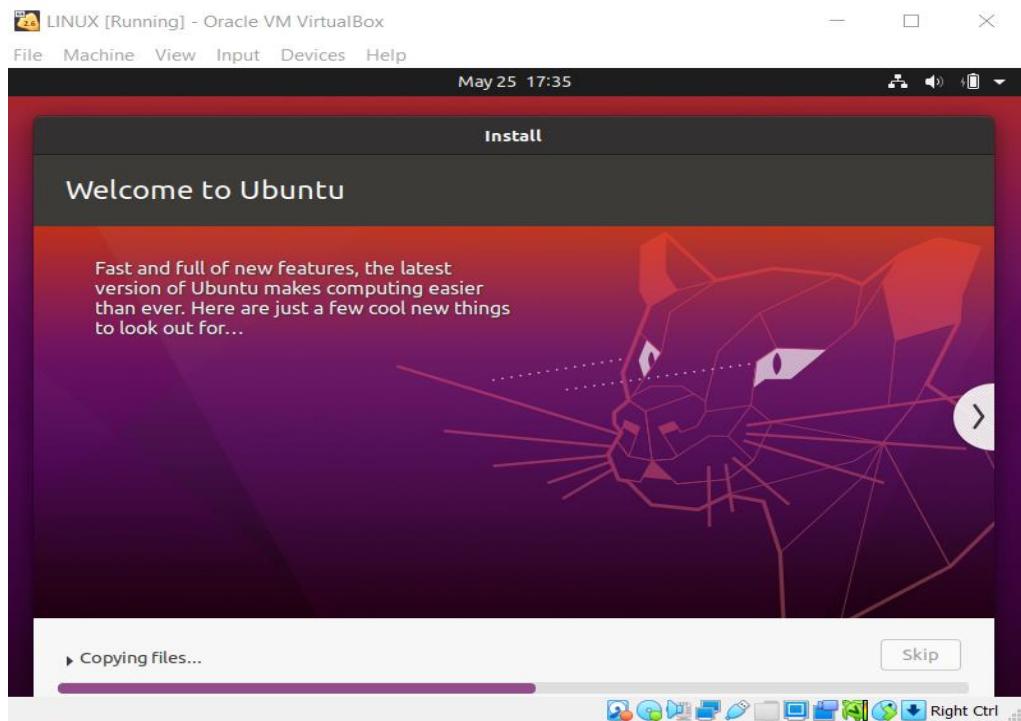
**Step-9** Select your location for setting up time zone, and click on continue



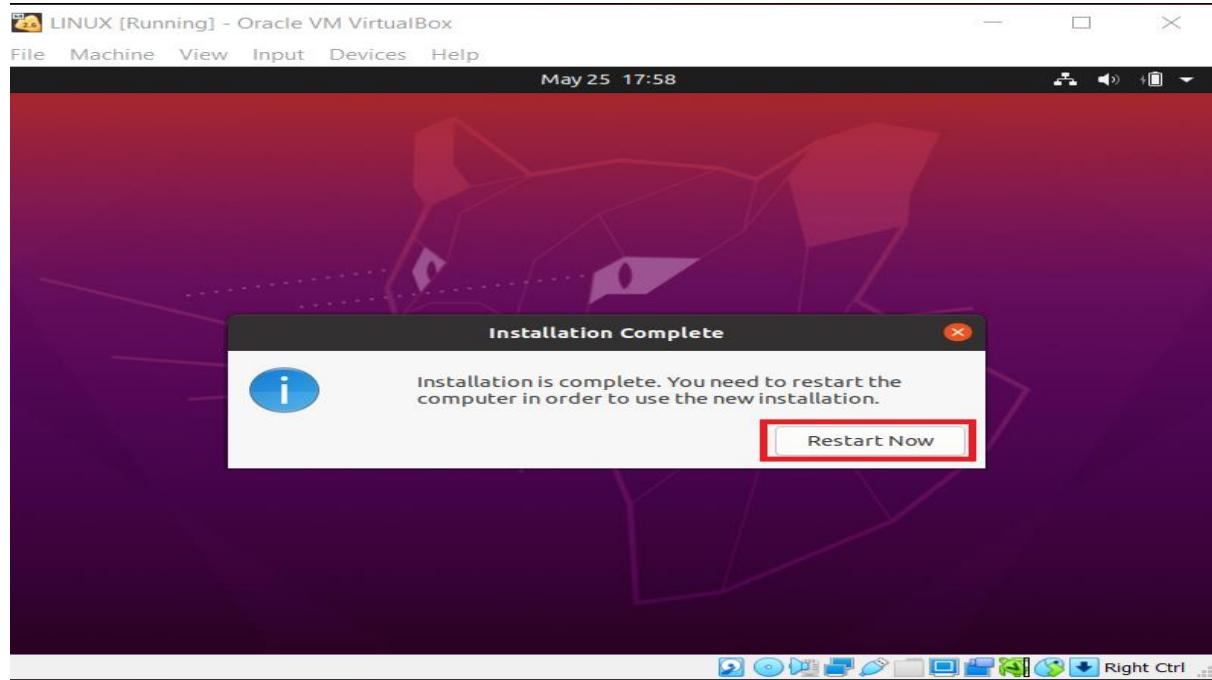
**Step-10** Select your username and password for your Ubuntu admin account. This information has been needed for installing any software package into Ubuntu and also for login to your OS. Fill up your details and tick on login automatically to ignore login attempt and click on continue



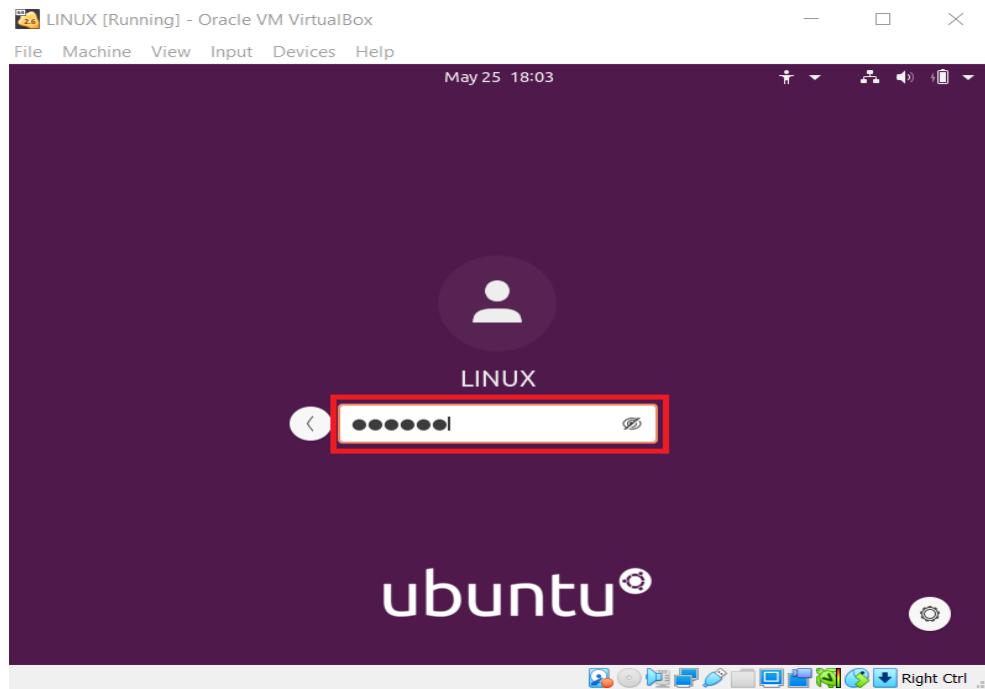
**Step-11** Installation process starts. May take up to 30 minutes. Please wait until installation process completes.



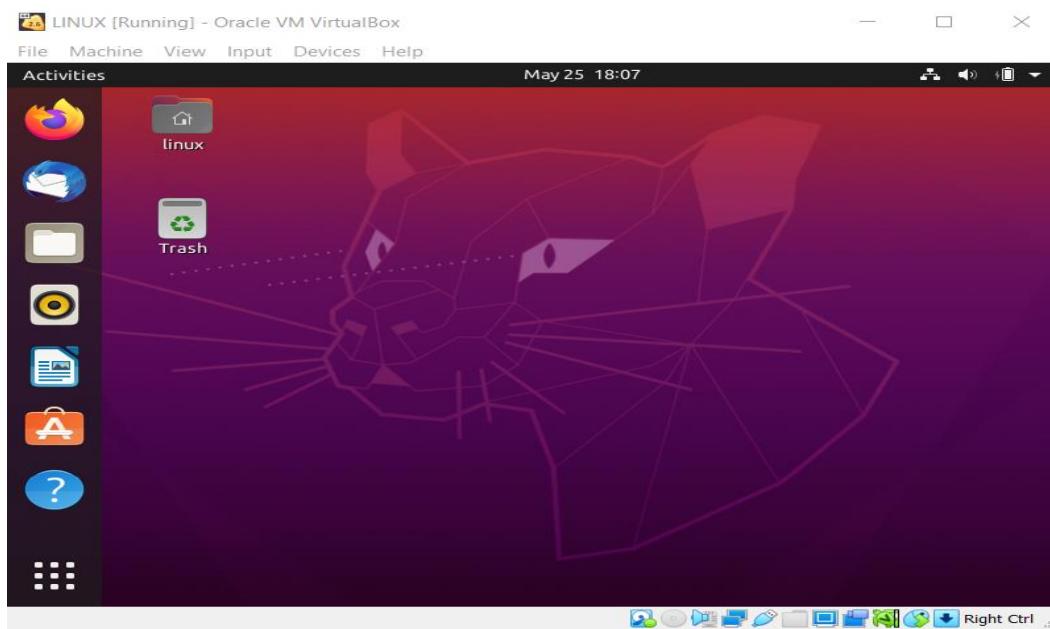
## Step-12 Click on Restart Now



## Step-13 After finishing the Restarting, you will see Ubuntu Desktop. Enter Password



#### Step-14 Ubuntu Desktop page



After installing OS, let's get started with some basic commands used in Linux.

## 1.4 Linux Commands

Here is a list of basic Linux commands:

### 1. **pwd command:**

Use the **pwd** command to find out the path of the current working directory (folder) you're in. The command will return an absolute (full) path, which is basically a path of all the directories that starts with a forward slash (/). An example of an absolute path is **/home/username**.

### 2. **cd command:**

To navigate through the Linux files and directories, use the **cd** command. It requires either the full path or the name of the directory, depending on the current working directory that you're in. Let's say you're in **/home/username/Documents** and you want to go to Photos, a subdirectory of Documents. To do so, simply type the following command: **cd Photos**.

Another scenario is if you want to switch to a completely new directory, for example, **/home/username/Movies**. In this case, you have to type **cd** followed by the directory's absolute path: **cd /home/username/Movies**.

There are some shortcuts to help you navigate quickly:

**cd ..** (with two dots) to move one directory up

**cd** to go straight to the home folder

**cd-** (with a hyphen) to move to your previous directory

On a side note, Linux's shell is case sensitive. So, you have to type the name's directory exactly as it is.

### 3. **ls command:**

The **ls** command is used to view the contents of a directory. By default, this command will display the contents of your current working directory.

If you want to see the content of other directories, type **ls** and then the directory's path. For example, enter **ls /home/username/Documents** to view the content of **Documents**.

There are variations you can use with the **ls** command:

**ls -R** will list all the files in the sub-directories as well

**ls -a** will show the hidden files

**ls -al** will list the files and directories with detailed information like the permissions, size, owner, etc.

### 4. **cat command:**

**cat** (short for concatenate) is one of the most frequently used commands in Linux. It is used to list the contents of a file on the standard output (stdout). To run this command, type **cat** followed by the file's name and its extension. For instance: **cat file.txt**.

Here are other ways to use the **cat** command:

**cat > filename** creates a new file

**cat filename1 filename2>filename3** joins two files (1 and 2) and stores the output of them in a new file (3)

to convert a file to upper- or lower-case use, **cat filename | tr a-z A-Z >output.txt**  
We list below some of the most frequently used cat commands.

- 1) To view a single file: **cat filename**. Output will show content of the given filename
- 2) To view multiple files: **cat file1 file2**. Output will show the content of file1 and file2.
- 3) To view contents of a file preceding with line numbers: **cat -n filename**. Output will show content with line number.
- 4) Copy the contents of one file to another file: **cat [filename-whose-contents-is-to-be-copied] > [destination-filename]**. The content will be copied in destination file.
- 5) Cat command can suppress repeated empty lines in output: **cat -s Language.txt**. Output will suppress repeated empty lines in output.
- 6) Cat command can append the contents of one file to the end of another file: **cat file1 >> file2**. Output will append the contents of one file to the end of another file.
- 7) Cat command can display content in reverse order using tac command: **tac filename**. Output will display content in reverse order.
- 8) Cat command can highlight the end of line: **cat -E "filename"**. Output Will highlight the end of line.
- 9) Cat command to display the content of all text files in the folder: **cat \*.txt**. Output will show the content of all text files present in the folder.
- 10) Cat command to write in an already existing file: **cat >> Languages.txt The newly added text**. Output will append the text "The newly added text." to the end of the file.

#### **5. cp command:**

Use the **cp** command to copy files from the current directory to a different directory. For instance, the command **cp scenery.jpg /home/username/Pictures** would create a copy of scenery.jpg (from your current directory) into the Pictures directory.

#### **6. mv command:**

The primary use of the **mv** command is to move files, although it can also be used to rename files.

The arguments in mv are similar to the cp command. You need to type **mv**, the file's name, and the destination's directory. For example: **mv file.txt /home/username/Documents**.

To rename files, the Linux command is **mv oldname.ext newname.ext**

#### **7. mkdir command:**

Use **mkdir** command to make a new directory — if you type **mkdir Music** it will create a directory called **Music**. There are extra **mkdir** commands as well:

To generate a new directory inside another directory, use this Linux basic command **mkdir Music/Newfile**

use the **p** (parents) option to create a directory in between two existing directories. For example, **mkdir -p Music/2020/Newfile** will create the new "2020" file.

**8. rmdir command:**

If you need to delete a directory, use the **rmdir** command. However, rmdir only allows you to delete empty directories.

**9. rm command:**

The **rm** command is used to delete directories and the contents within them. If you only want to delete the directory — as an alternative to rmdir — use **rm -r**.

**Note:** Be very careful with this command and double-check which directory you are in. This will delete everything and there is no undo.

**10. touch command:**

The **touch** command allows you to create a blank new file through the Linux command line. As an example, enter **touch /home/username/Documents/Web.html** to create an HTML file entitled **Web** under the **Documents** directory.

**11. locate command:**

You can use this command to **locate** a file, just like the search command in Windows. What's more, using the **-i** argument along with this command will make it case-insensitive, so you can search for a file even if you don't remember its exact name. To search for a file that contains two or more words, use an asterisk (\*). For example, **locate -i school\*note** command will search for any file that contains the word "school" and "note", whether it is uppercase or lowercase.

**12. find command:**

Similar to the **locate** command, using **find** also searches for files and directories. The difference is, you use the **find** command to locate files within a given directory.

As an example, **find /home/ -name notes.txt** command will search for a file called **notes.txt** within the home directory and its subdirectories.

Other variations when using the **find** are:

To find files in the current directory use, **find . -name notes.txt**

To look for directories use, **/ -type d -name notes. txt**

**13. grep command:**

Another basic Linux command that is undoubtedly helpful for everyday use is **grep**. It lets you search through all the text in a given file.

To illustrate, **grep blue notepad.txt** will search for the word blue in the notepad file. Lines that contain the searched word will be displayed fully.

**14. sudo command:**

Short for “**SuperUser Do**”, this command enables you to perform tasks that require administrative or root permissions. However, it is not advisable to use this command for daily use because it might be easy for an error to occur if you did something wrong.

**15. df command:**

Use **df** command to get a report on the system’s disk space usage, shown in percentage and KBs. If you want to see the report in megabytes, type **df -m**.

**16. du command:**

If you want to check how much space a file or a directory takes, the **du** (Disk Usage) command is the answer. However, the disk usage summary will show disk block numbers instead of the usual size format. If you want to see it in bytes, kilobytes, and megabytes, add the **-h** argument to the command line.

**17. head command:**

The **head** command is used to view the first lines of any text file. By default, it will show the first ten lines, but you can change this number to your liking. For example, if you only want to show the first five lines, type **head -n 5 filename.ext**.

**18. tail command:**

This one has a similar function to the head command, but instead of showing the first lines, the **tail** command will display the last ten lines of a text file. For example, **tail -n filename.ext**.

**19. diff command:**

Short for difference, the **diff** command compares the contents of two files line by line. After analyzing the files, it will output the lines that do not match. Programmers often use this command when they need to make program alterations instead of rewriting the entire source code.

The simplest form of this command is **diff file1.ext file2.ext**

**20. tar command:**

The **tar** command is the most used command to archive multiple files into a **tarball** — a common Linux file format that is similar to zip format, with compression being optional.

This command is quite complex with a long list of functions such as adding new files into an existing archive, listing the content of an archive, extracting the content from an archive, and many more.

**21. chmod command:**

**chmod** is another Linux command, used to change the read, write, and execute permissions of files and directories. As this command is rather complicated

**22. chown command:**

In Linux, all files are owned by a specific user. The **chown** command enables you to change or transfer the ownership of a file to the specified username. For instance, **chown linuxuser2 file.ext** will make **linuxuser2** as the owner of the **file.ext**.

**23. jobs command:**

**jobs** command will display all current jobs along with their statuses. A job is basically a process that is started by the shell.

**24. kill command:**

If you have an unresponsive program, you can terminate it manually by using the **kill** command. It will send a certain signal to the misbehaving app and instructs the app to terminate itself.

There is a total of **sixty-four signals** that you can use, but people usually only use two signals:

**SIGTERM (15)** — requests a program to stop running and gives it some time to save all of its progress. If you don't specify the signal when entering the **kill** command, this signal will be used.

**SIGKILL (9)** — forces programs to stop immediately. Unsaved progress will be lost. Besides knowing the signals, you also need to know the process identification number (PID) of the program you want to **kill**. If you don't know the PID, simply run the command **ps ux**.

After knowing what signal you want to use and the PID of the program, enter the following syntax:

**kill [signal option] PID.**

**25. ping command:**

Use the **ping** command to check your connectivity status to a server. For example, by simply entering **ping google.com**, the command will check whether you're able to connect to Google and also measure the response time.

**26. wget command:**

The Linux command line is super useful — you can even download files from the internet with the help of the **wget** command. To do so, simply type **wget** followed by the download link.

**27. uname command:**

The **uname** command, short for Unix Name, will print detailed information about your Linux system like the machine name, operating system, kernel, and so on.

**28. top command:**

As a terminal equivalent to Task Manager in Windows, the **top** command will display a list of running processes and how much CPU each process uses. It's very useful to monitor system resource usage, especially knowing which process needs to be terminated because it consumes too many resources.

**29. history command:**

When you've been using Linux for a certain period of time, you'll quickly notice that you can run hundreds of commands every day. As such, running **history** command is particularly useful if you want to review the commands you've entered before.

**30. man command:**

Confused about the function of certain Linux commands? Don't worry, you can easily learn how to use them right from Linux's shell by using the **man** command. For instance, entering **man tail** will show the manual instruction of the tail command.

**31. echo command:**

This command is used to move some data into a file. For example, if you want to add the text, "Hello, my name is John" into a file called name.txt, you would type **echo Hello, my name is John >> name.txt**

**32. zip, unzip command:**

Use the **zip** command to compress your files into a zip archive, and use the **unzip** command to extract the zipped files from a zip archive.

**33. hostname command:**

If you want to know the name of your host/network simply type **hostname**. Adding a -I to the end will display the IP address of your network.

**34. useradd, userdel command:**

Since Linux is a multi-user system, this means more than one person can interact with the same system at the same time. **useradd** is used to create a new user, while **passwd** is adding a password to that user's account. To add a new person named John type, **useradd John** and then to add his password type, **passwd 123456789**. To remove a user is very similar to adding a new user. To delete the users account type, **userdel UserName**

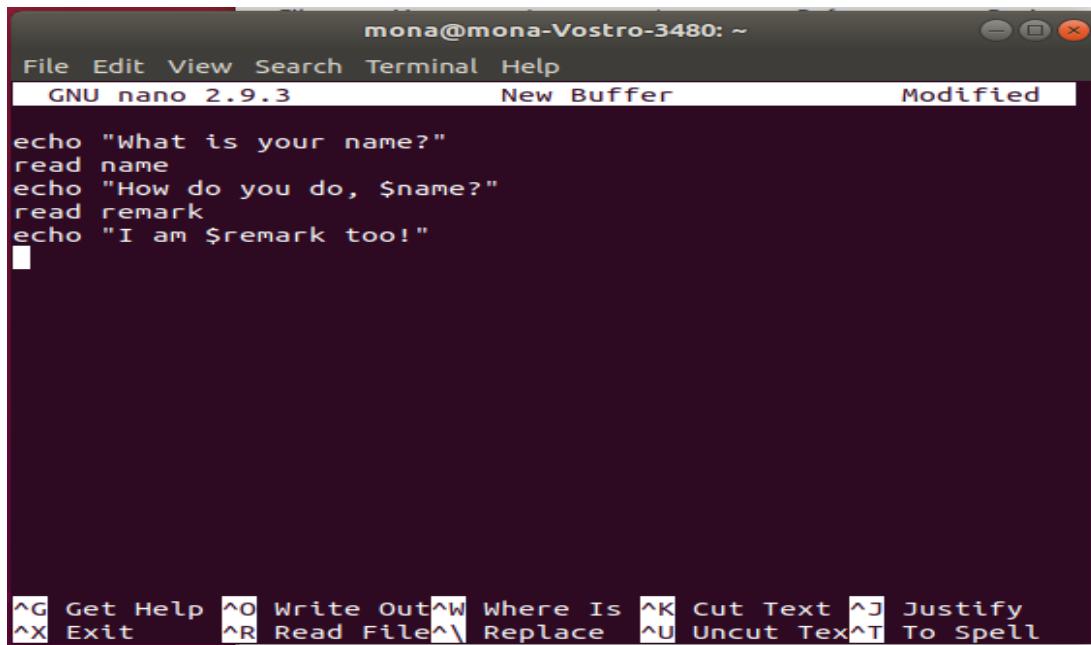
## 1.5 Working on different text editors: nano, vi

### Nano Editor

GNU nano is easy to use command line text editor for Linux operating systems. It includes all the basic functionality you'd expect from a regular text editor, like syntax highlighting, multiple buffers, search and replace with regular expression support, spellchecking, UTF-8 encoding, and more.

**Install NANO editor**

Open terminal and type in the following command: **sudo apt-get -y install nano**  
You will be prompted to enter the Password. After the installation is finished type the command: **nano** to be redirected to GNU Nano interface. Type in the following lines:



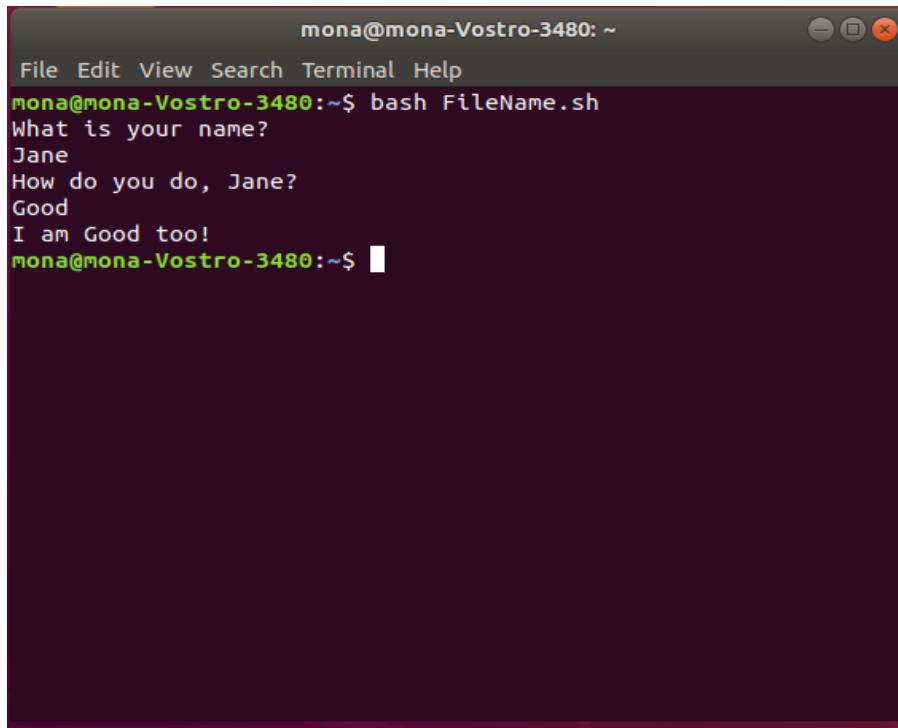
```
mona@mona-Vostro-3480: ~
File Edit View Search Terminal Help
GNU nano 2.9.3          New Buffer          Modified
echo "What is your name?"
read name
echo "How do you do, $name?"
read remark
echo "I am $remark too!"
```

^G Get Help ^O Write Out^W Where Is ^K Cut Text ^J Justify  
^X Exit ^R Read File^L Replace ^U Uncut Tex^T To Spell

After that Click **CRTL+X**. Enter **Y**. After that enter the filename **FileName.sh**  
Click **Enter**

After that use the command: **bash FileName.sh**

**Output:**



```
mona@mona-Vostro-3480: ~
File Edit View Search Terminal Help
mona@mona-Vostro-3480:~$ bash FileName.sh
What is your name?
Jane
How do you do, Jane?
Good
I am Good too!
mona@mona-Vostro-3480:~$
```

## VI Editor

The VI editor is the most popular and classic text editor in the Linux family. It is available in almost all Linux distributions and works the same across different platforms and distributions. There are different versions of the VI editor available, but the most popular one is VIM [ Vi IMproved]. In order to work with VI editor, we need to understand its operation mode which is divided into two parts.

### 1) VI Command mode

The VI editor opens in this mode and it only understands commands. This mode saves the changes you have made in the file, also you can move the cursor and cut, copy, and paste the text. The commands are case sensitive, so we have to use the right letter case.

### 2) VI Editor Insert mode:

This mode is for inserting text in the file. You can switch from the command mode to the insert mode by pressing “i” on the keyboard. Once in the insert mode, any key would be taken as input for the file. In order to return back to the command mode and save the changes made in the file you have to press the Esc key.

#### **Install VI editor**

You have to open the terminal and type the command: **sudo apt-get -y install vim**  
This will install vi editor on your system.

#### **How to use vi editor**

In order to launch the Vi editor, you open the terminal and type:

**vi <NEW\_filename> or <EXISTING\_filename>**

If you are opening an existing file, the editor will open it for you to edit.

Step 1: Creating a new file

```
mona@mona-Vostro-3480:~$ vi new_file
```

Step 2: The vi editor opens in the command mode. ‘~’ in the figure shows the unused lines.



"new\_file" [New File] 0,0-1 All

Command Mode

'~~ shows unused lines

Step 3: Next we have to press ‘i’ to enter the insert mode



-- INSERT -- 0,1 All

Insert Mode

Step 4: Add content to your new file.

```
Welcome To Tech Saksham
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

```
~
```

- o – Open a new line (goes into insert mode)
- dd – Delete line
- 3dd – Delete 3 lines.
- D – Delete contents of line after the cursor
- C – Delete contents of a line after the cursor and insert new text. Press ESC key to end insertion.
- dw – Delete word
- 4dw – Delete 4 words
- cw – Change word

So, summarizing we can say that vi editor works in two modes, command and insert. The command mode is used to take the user commands, and the Insert mode is for editing text. The vi editor is available in all the Linux Distributions, and is the most popular and commonly used Unix text editor.

## 1.6 Shell Scripting

The Linux operating system is made of two main components, the kernel and the shell. The shell in Linux serves as an interface between the user and the kernel and executes programs called commands. It is the Linux command line interpreter. Suppose, a user enters `pwd` then the shell executes the `pwd` command. It accepts human-readable commands into the system and executes those commands, which then run automatically and gives the output. The shell apart from a single command can also execute other programs such as scripts, and applications. When the terminal is opened in Linux, the Shell issues a command prompt (mostly \$), where the input is typed and then executed once the Enter key is used. The output of the result is thereafter displayed on the terminal. There are a number of shells available to a Linux user, you can select the one according to your convenience:

- `sh`: commonly known as the Bourne shell, it was developed by Steve Bourne. This is the original shell.
- `csh`, `tcsh` – `csh` (Command SHell) is a C shell and is easily able to read file commands. The command prompt for a C shell is %. `tcsh` is an advanced version of `csh` and behaves like `csh` with some additional utilities.
- `ksh` – this is the Korn SHell, which carries out commands either interactively.
- `bash` – The Borne Again SHell is the most popular shell used for Linux and is developed by GNU. It is the default shell for the latest Linux versions.

You can check the shell, configured for your login account by typing `echo $SHELL` in your terminal. If the default is not the `bash` shell, you can always change the shell using the CHange SHell (`chsh`) command. You can get a list of shells available on your system using the command `cat /etc/shells`

```
mona@mona-Vostro-3480:~$ cat /etc/shells
# /etc/shells: valid login shells
/bin/sh
/bin/bash
/bin/rbash
/bin/dash
mona@mona-Vostro-3480:~$ echo "My current shell is $SHELL ($0)"
My current shell is /bin/bash (bash)
mona@mona-Vostro-3480:~$ chsh
Password:
Changing the login shell for mona
Enter the new value, or press ENTER for the default
    Login Shell [/bin/bash]:
```

We will now discuss about the shell script, which is basically a text file containing a sequence of commands for a UNIX based operating system like Linux. A shell script is mostly used for repetitive tasks that may be time consuming to execute by typing one line at a time. It is mostly used for executing routine backups, linking existing programs together, running a program or automating the code compiling process. It combines lengthy and repetitive sequences of commands into a single and simple script that can be stored and executed anytime when required.

The basic steps involved with shell scripting include first writing the script, then making the script accessible to the shell and giving the shell execute permissions. The content of the script can be a series of commands in a language that can be interpreted by the shell. The functions that the shell scripts support include loops, variables, arrays and if/then/else statements. We will discuss below the steps needed to create a Shell Script:

- First create a file using your favourite text editor, and name it filename.sh
- Start the script with **#!/bin/sh** where **#!** Directs the script to the interpreter location and **/bin/sh** directs it to the bourne shell.
- Write the code and save the file. Remember the extension of the file should be sh
- In order to execute the script, use the command **bash filename.sh**

Let us now start with a simple script, where we will take the input from the user and display the value by combining other string value. We first create the script.

```
#!/bin/bash
echo "Enter your name"
read name
echo "We Welcome $name to the Tech Saksham Program"
```

4,51

All

The second step involves running the script file using bash command where we will be prompted to enter the input.

```
mona@mona-Vostro-3480:~$ bash script1.sh
Enter your name
```

We enter the input, and will see that the script reads the name.

```
mona@mona-Vostro-3480:~$ bash script1.sh
Enter your name
Jane
We Welcome Jane to the Tech Saksham Program
mona@mona-Vostro-3480:~$
```

In the next example we are using the while loop. In this example the while loop will execute 7 times and the value of the count variable will increment by one in each step. The while loop will terminate when the value of the count variable will be 7.

```
#!/bin/bash
count=1
while [ true ]
do
    echo $count
    if [ $count -eq 7 ];
    then
        break
    fi
    ((count++))
done
~
~
~
~
~
~
~
<example1.sh" 11L, 109C          11,4      All
```

We run the file using the bash command and get the following output.

```
mona@mona-Vostro-3480:~$ vi example1.sh
mona@mona-Vostro-3480:~$ bash example1.sh
1
2
3
4
5
6
7
```

We will next consider the example of a for loop, where after writing a script with a for loop, the loop will iterate for 10 times and print all the values of the x variable in a single line.

```
#!/bin/bash
for (( x = 10; x>0; x-- ))
do
    echo -n "$x "
done
printf "\n"
~
6,11      All
```

The file is run using bash command and the following output is generated:

```
mona@mona-Vostro-3480:~$ bash for-example.sh
10 9 8 7 6 5 4 3 2 1
mona@mona-Vostro-3480:~$
```

In this final example, we will write a script using if condition. Here the starting and ending block of the statement is defined by 'if' and 'fi'. We will write a simple script, where, we take the age as input and compare if it is greater than equal to 18.

```
#!/bin/bash
echo "Enter your age"
read n

if [ $n -ge 18 ];
then
    echo "You are eligible to vote"
else
    echo "Wait till you are 18 years to vote"
fi
```

10,2

All

The script is run using the bash command and we get the following input

```
mona@mona-Vostro-3480:~$ bash if_example.sh
Enter your age
4
Wait till you are 18 years to vote
mona@mona-Vostro-3480:~$ bash if_example.sh
Enter your age
25
You are eligible to vote
mona@mona-Vostro-3480:~$
```

We see from the above examples the simplicity and efficiency of the shell scripts and how shell scripting can help us create complex programs containing conditional statements, loops, and functions. The syntax on the script is the same as it is on the shell command line, removing any interpretation issues.

## 1.7 Managing Linux Files

In Linux operating system, each entity is regarded as a file. There is a popular saying 'Everything is a file in Linux'. Let us first get an overview of the Linux file system:

1. Regular files: These are the most common file types and contain human readable text, program instructions and ASCII characters. They can be simple text files, pdf files, binary files, zipped or compressed files and multimedia files.
2. Special files: These are the files that represent physical devices such as printers, CD drives, mounted volumes and any input and output device.
3. Directories: A directory is a special file type that stores both regular and special files in a hierarchical order, starting from the root directory and then branching out to other directories.

The linux files can be managed using some of the commands listed below:

pwd command, cd command, ls command, touch command, cat command, mv command, cp command, mkdir command, rmdir command, rm command, find and locate command.

All the above commands are discussed before.

Linux as we know, is a multi-user operating system which can be accessed by many users simultaneously. This in turn can raise security issues, where a user can corrupt, change or remove crucial data. Therefore, for effective security, Linux has divided the authorization into 2 levels:

- Ownership

- Permission

### Linux File Ownership

Linux assigns three types of owners, to every file and directory. They are

User: A user is the owner of the file. By default, a person becomes the owner as soon as he/she creates a file.

Group: A user-group contains multiple users. All the users belonging to a particular group will have the same Linux group permissions to access the file.

Other: Any other user, who has neither created the file nor belongs to a user-group belongs to this category. When some settings are set for Others, it may also refer to settings for the world.

The Permission system on Linux helps it to distinguish between the three user types. Let us now understand the permission and how do they define the user behaviour.

### Linux File Permissions

Every file or directory in the Linux system has three permissions, which are defined for all the three owners discussed above:

Read: This permission gives the respective owner the authority to open and read a file. When given read permission on a directory, it gives the user the ability to list its content.

Write: The write permission gives the respective user the authority to modify the contents of the file. The write permission on a directory allows the user to add, remove and rename files stored in the directory.

Execute: In Linux, a user cannot run a program unless the execute permission is set. You can easily run an executable program in Windows, which has an extension .exe. We will understand file permissions in Linux with an example, **ls -l** command on terminal gives

```
mona@mona-Vostro-3480:~$ ls -l
total 52
drwxr-xr-x 2 mona mona 4096 Jul 28 10:36 Desktop
drwxr-xr-x 2 mona mona 4096 Jul 10 13:47 Documents
drwxr-xr-x 3 mona mona 4096 Jul 27 17:36 Downloads
-rw-r--r-- 1 mona mona 8980 Jul 10 13:44 examples.desktop
-rw-r--r-- 1 mona mona 102 Jul 26 11:19 FileName.sh
drwxr-xr-x 2 mona mona 4096 Jul 10 13:47 Music
-rw-r--r-- 1 mona mona 0 Jul 28 08:47 newfile
-rw-r--r-- 1 mona mona 24 Jul 28 08:52 new_file
drwxr-xr-x 2 mona mona 4096 Jul 10 13:47 Pictures
drwxr-xr-x 2 mona mona 4096 Jul 10 13:47 Public
drwxr-xr-x 2 mona mona 4096 Jul 10 13:47 Templates
```

The terms listed in the first row, 'drwxr-xr-x' are the ones that tells us about the permission given to the owner, user group and others.

```
-rwxr--r--  
-rwxr--r--  
-
```

In the above figure, the first '-' implies that we have selected a file. In case it is a directory, it will start with 'd' as shown below:

```
drwxr-xr-x  
drwxr-xr-x  
drwxr-xr-x
```

The characters shown above are easy to remember:

r = read permission

w = write permission

x = execute permission

-= no permission

-rwxr--r--

The first part ‘-’ of the above code says it is a file. Then ‘rw-’ suggests that the owner can read the file, write or edit the file, but has no permission to execute the file, as the execute bit is set to ‘-’. The second part is ‘r--’ for the user group, who can only read the file, but cannot write or execute the file. The third part is also ‘r--’, which means any user can only read the file.

Changing permissions in Linux using ‘chmod’ command

We will now discuss how to change the file permissions. We can use the ‘chmod’ command which stands for ‘change mode’. With this command we can set permissions, i.e., read, write and execute on a file or directory for the owner, group and the others.

The syntax for that is **chmod permissions filename**

We can use the command in either the absolute or the symbolic mode. In the numeric mode the file permissions are represented as a three-digit octal number. We list below the numbers for all permission types.

Number	Permission Type	Symbol
0	No permission	-
1	Execute	--x
2	Write	-w-
4	Read	r--

The other permissions are calculated by summing over the above numbers. So, if we want to give a write and execute permission to a user it will be  $1 + 2 = 3$ . Similarly, you can work out the numbers for the other permissions. We list them all in the following table:

Number	Permission Type	Symbol
3	Execute + Write	-wx
5	Read + Execute	r-x
6	Read + Write	rw-
7	Read + Write + Execute	rwx

Let us now see the chmod permissions command in action. We will first check the current file permissions of a file:

```
mona@mona-Vostro-3480:~/Desktop$ ls -l
total 4
-rwxr--r-- 1 mona mona 24 Jul 28 08:52 new_file
```

We will now use the chmod command and check the permissions again:

```
mona@mona-Vostro-3480:~/Desktop$ chmod 764 new_file
mona@mona-Vostro-3480:~/Desktop$ ls -l
total 4
-rwxrw-r-- 1 mona mona 24 Jul 28 08:52 new_file
```

We have changed the permissions of the file to 764, which says, owner can read, write and execute. The user-group can read and write and the users can only read. Therefore, the final code will show as ‘-rwxrw-r--’. This is how you can change user permissions in Linux on file, by assigning a number.

We can change permissions for all the owners, in the symbolic mode, where it uses mathematical symbols to modify the file permissions.

The “+” operator adds a permission to a file or directory. The “-” operator removes the permission. The “=” operator sets the permission and overrides the permissions set earlier. The various owners are represented as ‘u’ for the user, ‘g’ for the group, ‘o’ for other, and ‘a’ for all.

Step 1: Checking current file permissions

```
mona@mona-Vostro-3480:~/Desktop$ ls -l
total 4
-rw-r--r-- 1 mona mona 24 Jul 28 08:52 new_file
```

Step 2: Setting permissions to the ‘other’ users

```
mona@mona-Vostro-3480:~/Desktop$ chmod o=rwx new_file
mona@mona-Vostro-3480:~/Desktop$ ls -l
total 4
-rw-r--rwx 1 mona mona 24 Jul 28 08:52 new_file
```

Step 3: Adding ‘execute’ permission to the ‘user-group’.

```
mona@mona-Vostro-3480:~/Desktop$ chmod g+x new_file
mona@mona-Vostro-3480:~/Desktop$ ls -l
total 4
-rw-r-xrwx 1 mona mona 24 Jul 28 08:52 new_file
```

Step 4: Removing ‘read’ permission for ‘user’

```
mona@mona-Vostro-3480:~/Desktop$ chmod u-r new_file
mona@mona-Vostro-3480:~/Desktop$ ls -l
total 4
--w-r-xrwx 1 mona mona 24 Jul 28 08:52 new_file
```

The file /etc/group contains all the groups defined in the system. You can use the command ‘groups’ to find all the groups you are a member of. The ‘chgrp’ command can change the group ownership **chgrp group filename**. Similarly, the ‘chown’ command can change the ownership of a file/directory. Use the following commands: **chown user file** or **chown user:group file**.

We next discuss about the three types of accounts on a Unix system –

#### *Root account*

This is also called **superuser** and would have complete and unfettered control of the system. A superuser can run any commands without any restriction. This user should be assumed as a system administrator.

#### *System accounts*

System accounts are those needed for the operation of system-specific components for example mail accounts and the **sshd** accounts. These accounts are usually needed for some specific function on your system, and any modifications to them could adversely affect the system.

#### *User accounts*

User accounts provide interactive access to the system for users and groups of users. General users are typically assigned to these accounts and usually have limited access to critical system files and directories.

Unix supports a concept of *Group Account* which logically groups a number of accounts. Every account would be a part of another group account. A Unix group plays important role in handling file permissions and process management.

### Managing Users and Groups

There are four main user administration files –

- **/etc/passwd** – Keeps the user account and password information. This file holds the majority of information about accounts on the Unix system.

- **/etc/shadow** – Holds the encrypted password of the corresponding account. Not all the systems support this file.
- **/etc/group** – This file contains the group information for each account.
- **/etc/gshadow** – This file contains secure group account information.

Check all the above files using the **cat** command.

## 1.8 SSH Command

The SSH command stands for “Secure Shell” and it provides a secure encrypted connection between two hosts over an insecure network. It uses the client and server applications to establish a remote connection. The SSH command allows you to gain access and remotely manage other computers, transfer files, and do virtually anything you can do while physically sitting in front of the machine.

When a secure SSH connection is established, a shell session is started and we are able to manipulate the server by typing commands within the client on our local computer.

### How does SSH work?

We need two components, a client and the corresponding server-side component in order to establish an SSH connection. An SSH client is an application we install on the computer which we will later use to connect to another computer/server. The SSH client uses the provided remote host information to initiate the connection and if the credentials are verified, establishes the encrypted connection.

On the server's side, there is a component called an SSH daemon that is constantly listening to a specific TCP/IP port for possible client connection requests. Once a client initiates a connection, the SSH daemon will respond with the software and the protocol versions it supports and the two will exchange their identification data. If the provided credentials are correct, SSH creates a new session for the appropriate environment. Linux uses the OpenSSH client. To check if the client is available on your Linux-based system, you will need to:

- Load an SSH terminal. You can either search for “terminal” or press **CTRL + ALT + T** on your keyboard.
- Type in **ssh** and press Enter in the terminal.

If the client is installed, you will get an output like:

```
mona@mona-Vostro-3480:~$ ssh
usage: ssh [-46AacfGgKkMnqsTtVvXxYy] [-b bind_address] [-c cipher_spec]
           [-D [bind_address:]port] [-E log_file] [-e escape_char]
           [-F configfile] [-I pkcs11] [-i identity_file]
           [-J [user@]host[:port]] [-L address] [-l login_name] [-m mac_spec]
           [-o ctl_cmd] [-o option] [-p port] [-Q query_option] [-R address]
           [-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]]
           [user@]hostname [command]
```

If not installed, you have to type **sudo apt-get install openssh-client**

You then type in your superuser password and press enter to complete the installation. You will now be able to SSH into any machine, which has server-side application on it. Added to that, you also need to have the necessary privilege to gain access, along with the hostname or IP address.

The next step is to check if OpenSSH server is available on the Ubuntu system of the remote computer that needs to accept SSH connections. If we type **ssh localhost** and press enter

- For the systems without the SSH server installed, the error will be  
**ssh: connect to host localhost port 22: Connection refused**

In case OpenSSH server is not installed, we have to install it with the command  
**sudo apt-get install openssh-server**

After the files are installed, you can check if the SSH server is running on the machine by typing

```
mona@mona-Vostro-3480:~$ sudo service ssh status
[sudo] password for mona:
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Thu 2022-07-28 14:22:08 IST; 31min ago
    Main PID: 761 (sshd)
      Tasks: 1 (limit: 4915)
     CGroup: /system.slice/ssh.service
             └─761 /usr/sbin/sshd -D

Jul 28 14:22:08 mona-Vostro-3480 systemd[1]: Starting OpenBSD Secure Shell server...
Jul 28 14:22:08 mona-Vostro-3480 sshd[761]: Server listening on 0.0.0.0 port 22.
Jul 28 14:22:08 mona-Vostro-3480 sshd[761]: Server listening on :: port 22.
Jul 28 14:22:08 mona-Vostro-3480 systemd[1]: Started OpenBSD Secure Shell server.
Jul 28 14:30:36 mona-Vostro-3480 sshd[2391]: Connection closed by 127.0.0.1 port 59728 [preauth]
```

Another way to test is, typing the command

```
mona@mona-Vostro-3480:~$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:Pl4VbpuJz50FcLR76uXZHxn5xSOC4i7VvHHAV7oRoEg.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
mona@localhost's password: 
```

Once you enter the password, you will be connected to the remote computer. If you have trouble connecting to a remote server, make sure that

- The IP address of the remote machine is correct.
- The port SSH daemon is listening to is not blocked by a firewall or forwarded incorrectly.
- Your username and password are correct.
- The SSH software is installed properly.

## SCP Command

We will now discuss about the scp command, which is an easy way to transfer files between the Linux systems. Scp stands for secure copy and is a part of the SSH tool.

With scp, you can copy a file or directory:

- From your local system to a remote system.
- From a remote system to your local system.
- Between two remote systems from your local system.

The scp command syntax takes the following form:

**scp [OPTION] [user@]SRC\_HOST:]file1 [user@]DEST\_HOST:]file2**

where,

- OPTION - scp options such as cipher, ssh configuration, ssh port, limit, recursive copy ...etc.
- [user@]SRC\_HOST:]file1 - Source file.
- [user@]DEST\_HOST:]file2 - Destination file

Local files should be specified using an absolute or relative path, while remote file names should include a user and host specification.

- To copy a file from a local to a remote system run the following command:

**scp file.txt remote\_username@10.10.0.2:/remote/directory**

Where file.txt is the name of the file we want to copy, remote\_username is the user on the remote server, 10.10.0.2 is the server IP address. The /remote/directory is the path to the directory you want to copy the file to. If you don't specify a remote directory, the file will be copied to the remote user home directory.

- 2) To copy a file from a remote to a local system, use the remote location as a source and local location as the destination.

**scp remote\_username@10.10.0.2:/remote/file.txt /local/directory**

- 3) The following command will copy the file /files/file.txt from the remote host host1.com to the directory /files on the remote host host2.com.

**scp user1@host1.com:/files/file.txt user2@host2.com:/files**

We discussed about the Linux Operating System, which is a Unix-like, open source and community-developed operating system. The concept of database will be discussed in the next chapter.

# Chapter 2: Introduction to database

## Learning Outcomes:

- Introduction to Data Base Management System
- Learn about the structure and terminology of RDBMS
- Explore MySQL with different SQL commands
- Understanding key concepts of CRUD operations

## 2.1 Introduction to DBMS

In computerized information system data is the basic resource of the organization. So, proper organization and management for data is required for the organization to run smoothly. Database management system deals with the knowledge of how data is stored and managed on a computerized information system. In any organization, it requires accurate and reliable data for better decision making, ensuring privacy of data and controlling data efficiently. The examples include deposit and/or withdrawal from a bank, hotel, airline or railway reservation, purchase items from supermarkets in all cases, a database is accessed.

### What is data?

Data are the known facts or figures that have implicit meaning. It can also be defined as it is the representation of facts, concepts or instructions in a formal manner, which is suitable for understanding and processing. Data can be represented in alphabets (A-Z, a-z), digits (0-9) and using special characters (+, -, ., #, \$, etc) e.g.: 25, "ajit" etc.

**Information:** Information is the processed data on which decisions and actions are based. Information can be defined as organized and classified data used to provide meaningful values, e.g.: "The age of John is 18"

**File:** File is a collection of related data stored in secondary memory.

**File Oriented Approach:** The traditional file-oriented approach to information processing each application has a separate master file and its own set of personal files. In file-oriented approach the program is dependent on the files and files are dependent upon the programs.

### Disadvantages of the file-oriented approach:

**1) Data redundancy and inconsistency:** The same information may be written in several files. This redundancy leads to higher storage and access cost. It may lead data inconsistency that is the various copies of the same data may present at multiple places for example a changed customer address may be reflected in single file but not elsewhere in the system.

**2) Difficulty in accessing data:** The conventional file processing system does not allow data to be retrieved in a convenient and efficient manner according to user choice.

**3) Data isolation:** Because data are scattered in various files and files may be in different formats, retrieving the appropriate data is difficult with new application programs.

**4) Integrity Problems:** Developers enforce data validation in the system by adding appropriate code in the various application programs. However, when new constraints are added, it is difficult to change the programs to enforce them.

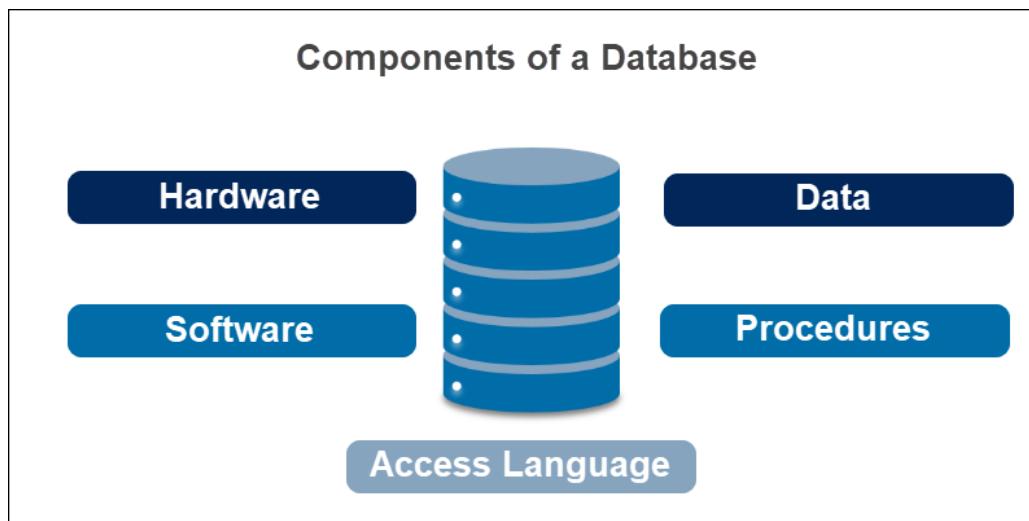
**5) Atomicity:** It is difficult to ensure atomicity in a file processing system when transaction failure occurs due to power failure, networking problems etc. (atomicity: either all operations of the transaction are reflected properly in the database or none are)

**6) Concurrent access:** In the file processing system it is not possible to access the same file for transaction at the same time.

**7) Security problems:** There is no security provided in the file processing system to secure the data from unauthorized user access.

## What is a Database?

A **Database** is a collection of related data organized in a way that data can be easily accessed, managed, and updated. The Database can be software-based or hardware-based, with one sole purpose, storing data. During the early computer days, data was collected and stored on tapes, which were mostly write-only, which means once data is stored on it, it can never be read again. They were slow and bulky, and soon computer scientists realized that they needed a better solution to this problem. Larry Ellison, the co-founder of Oracle, was amongst the first few who realized the need for a software-based Database Management System.



Reference: <https://phoenixnap.com/kb/what-is-a-database>

## Why do we use a database?

There are multiple reasons behind it such as:

- Databases can store very large numbers of records efficiently, It is very quick and easy to find information.
- It is easy to add new data and to edit or delete old data.
- Data can be searched & sorted easily.
- Data can be imported into other applications.
- More than one person can access the same database at the same time - multi-access.
- Security may be better than in paper files.

## Why do we need a database?

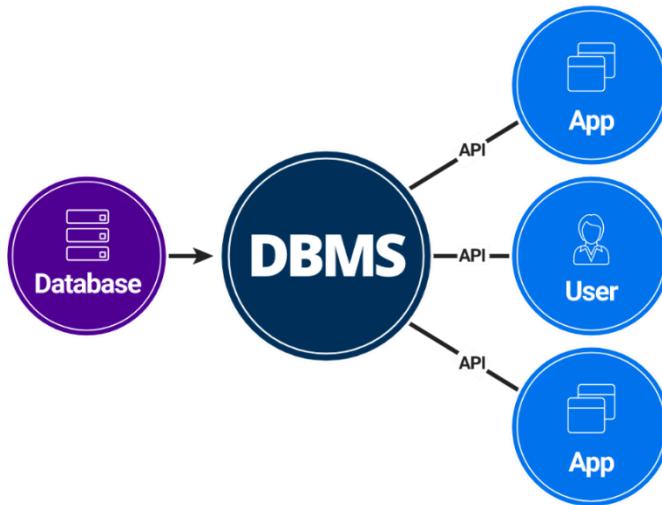
1. **Manages large amounts of data** - A database stores and manages a large amount of data on a daily basis. This would not be possible using any other tool such as a spreadsheet as they would simply not work.
2. **Accurate**- A database is pretty accurate as it has all sorts of built-in constraints, checks, etc. This means that the information available in a database is guaranteed to be correct in most cases.
3. **Easy to update data**- In a database, it is easy to update data using various Data Manipulation languages (DML) available. One of these languages is SQL.
4. **Security of data**- Databases have various methods to ensure security of data. There are user logins required before accessing a database and various access specifiers. These allow only authorized users to access the database.
5. **Data integrity**- This is ensured in databases by using various constraints for data. Data integrity in databases makes sure that the data is accurate and consistent in a database.
6. **Easy to research data**- It is very easy to access and research data in a database. This is done using Data Query Languages (DQL) which allows searching of any data in the database and performing computations on it.

## What is DBMS?

DBMS stands for Database Management System. We can break it like this DBMS = Database + Management System. Database is a collection of data and Management System is a set of programs to store and retrieve those data. Based on this we can define DBMS like this: DBMS is a collection of inter-related data and set of programs to store & access those data in an easy and effective manner.

- The Database management system is software that is used to manage databases. For example, MySQL, Oracle, etc. are very popular commercial databases which are used in different applications.
- DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more.

- It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.



Reference: <https://www.smartsheet.com/sites/default/files/ic-database-management-dbms-qatekeeper.svg>

### DBMS allows users the following tasks:

**Data Definition:** It is used for creation, modification, and removal of definition that defines the organization of data in the database.

**Data Updation:** It is used for the insertion, modification, and deletion of the actual data in the database.

**Data Retrieval:** It is used to retrieve data from the database which can be used by applications for various purposes.

**User Administration:** It is used for registering and monitoring users, maintaining data integrity, enforcing data security, dealing with concurrency control, monitoring performance, and recovering information corrupted by unexpected failure.

### Characteristics of DBMS

- It uses a digital repository established on a server to store and manage the information.
- It can provide a clear and logical view of the process that manipulates data.
- DBMS contains automatic backup and recovery procedures.
- It contains ACID properties which maintain data in a healthy state in case of failure.
- It can reduce the complex relationship between data.
- It is used to support manipulation and processing of data.
- It is used to provide security of data.
- It can view the database from different viewpoints according to the requirements of the user.

### Advantages of DBMS

1. **Controls database redundancy:** It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.

2. **Data sharing:** In DBMS, the authorized users of an organization can share the data among multiple users.
3. **Easily Maintenance:** It is easily maintainable due to the centralized nature of the database system.
4. **Reduce time:** It reduces development time and maintenance need.
5. **Backup:** It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.
6. **Multiple user interface:** It provides different types of user interfaces like graphical user interfaces, application program interfaces.

## Disadvantages of DBMS

1. **Cost of Hardware and Software:** It requires a high speed of data processor and large memory size to run DBMS software.
2. **Size:** It occupies a large space for disks and large memory to run them efficiently.
3. **Complexity:** Database system creates additional complexity and requirements.
4. **Higher impact of failure:** Failure is highly impacted in the database because in most of the organizations, all the data is stored in a single database. If the database is damaged due to electric failure or database corruption then the data may be lost forever.

## DBMS applications

Applications where we use Database Management Systems are:



Reference- <https://www.thecscience.com/2021/07/DBMS-applications.html>

**Telecom:** There is a database to keep track of the information regarding calls made, network usage, customer details etc. Without the database systems it is hard to maintain that huge amount of data that keeps updating every millisecond.

**Industry:** Whether it is a manufacturing unit, warehouse, or distribution center, each one needs a database to keep the records of the ins and outs. For example, the distribution center should keep track of the product units that are supplied into the

center as well as the products that get delivered out from the distribution center each day; this is where DBMS comes into the picture.

**Banking System:** For storing customer info, tracking day-to-day credit and debit transactions, generating bank statements, etc. All this work has been done with the help of Database management systems. Also, banking systems need the security of data as the data is sensitive, this is efficiently taken care of by the DBMS systems.

**Sales:** To store customer information, production information and invoice details. Using DBMS, you can track, manage and generate historical data to analyze the sales data.

**Airlines:** To travel through airlines, we make early reservations, this reservation information along with the flight schedule is stored in a database. This is where the real-time update of data is necessary as a flight seat reserved for one passenger should not be allocated to another passenger, this is easily handled by the DBMS systems as the data updates are in real-time and fast.

**Education sector:** Database systems are frequently used in schools and colleges to store and retrieve the data regarding student details, staff details, course details, exam details, payroll data, attendance details, fees details etc. There is a large amount of inter-related data that needs to be stored and retrieved in an efficient manner.

**Online shopping:** You must be aware of online shopping websites such as Amazon, Flipkart, etc. These sites store product information, your addresses and preferences, and credit details and provide you with the relevant list of products based on your query. All this involves a Database management system. Along with managing the vast catalog of items, there is a need to secure the user's private information such as bank & card details. All this is taken care of by database management systems.

## 2.2 Introduction to RDBMS

### What is RDBMS?

RDBMS stands for Relational Database Management System. RDBMS is the basis for SQL, and for all modern database systems like MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access. A Relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as introduced by E. F. Codd. A relational database is a collection of data items with pre-defined relationships between them. These items are organized as a set of tables with columns and rows. Tables are used to hold information about the objects to be represented in the database. E.g., in a ticket processing system, details about us (e.g., age, gender) and our journey (e.g., source, destination), are collected, and the ticket is provided to us.

### Benefits of Persistent Storage

**Simplicity:** Persistent storage helps developers provision their storage, without necessarily needing any expertise as storage experts. It simply allows them to provision volumes for both on-premise/ public cloud services.

**Security:** When it comes to the security and encryption aspects of storage solutions, persistent storage scores high. It fulfills the security requirements of most enterprises

in terms of volume-level encryption, self-encrypting disks, and key management, among others to protect them against any kind of data loss and security breaches.

**Flexibility:** Persistent storage offers you great flexibility over traditional storage and lets you use the same software across different virtual machines, clouds and containers. Further, developers also enjoy the flexibility to choose the storage interfaces for their workload, including file, block or object storage. It also gives developers the ability to deliver data services with one system, irrespective of protocol, thus boosting productivity, offering more freedom, and leading to more effective application development.

**Portability:** Today's cloud-native world requires organizations to adopt a hybrid cloud approach to be able to combine the benefits of public and on-premises clouds. Persistent storage makes it easy to migrate your stateless applications across multiple clouds and migrate your data from one cloud to other clouds.

**Efficiency:** Persistent storage makes application development much more efficient. It eliminates the need to rewrite applications when you want to port them from one cloud provider to another and you can simply move applications without expensive or time-consuming rewrites whenever you want.

**Cost-effectiveness:** With persistent storage, you only have to pay for the storage and computer you use. It scales on-demand with no disruptions, growing and shrinking automatically as you add and remove files.

### *DBMS vs RDBMS*

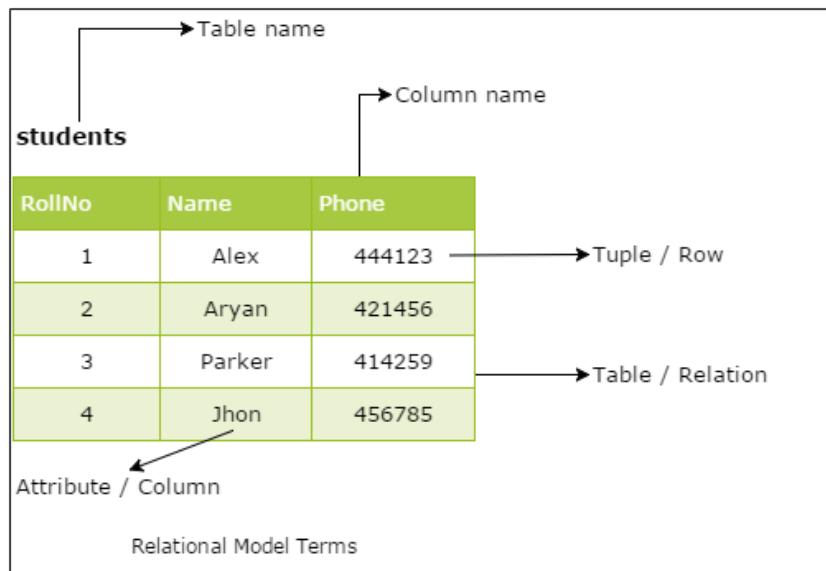
DBMS	RDBMS
<ul style="list-style-type: none"><li>• DBMS stands for "Database Management System".</li></ul>	<ul style="list-style-type: none"><li>• RDBMS stands for "Relational Database Management System".</li></ul>
<ul style="list-style-type: none"><li>• DBMS technology stores the data in the form of files.</li></ul>	<ul style="list-style-type: none"><li>• RDBMS stores the data in the form of tables.</li></ul>
<ul style="list-style-type: none"><li>• DBMS is designed to handle small amounts of data.</li></ul>	<ul style="list-style-type: none"><li>• RDBMS is designed to deal with vast amount of data.</li></ul>
<ul style="list-style-type: none"><li>• DBMS provides support only for a single user at a time.</li></ul>	<ul style="list-style-type: none"><li>• RDBMS provides support for multiple users at a time.</li></ul>

Reference: <https://www.tutorialsmate.com/2021/02/difference-between-dbms-and-rdbms.html>

## Terminology in RDBMS

1. **Table-** A table is a set of data represented by columns and rows. A table in a database looks like a simple spreadsheet.
2. **Row-** It is a combination of column values and is referred to as a record. A record is also called as a row of data is each individual entry that exists in a table.

3. **Column-** It is referred to as a field. A column is a vertical entity in a table that contains all information associated with a specific field in a table.
4. **Relation-** It defines database relationships in the form of tables.
5. **Tuples-** A single row of a table, which contains a single record for that relation. For example, if we have a table to store book details for a shop, then each tuple is an individual book the shop sells, and it will store all the data on that individual book such as the ISBN, book title and price.
6. **Attributes-** An individual piece of data in a record is known as a field, or attribute.
7. **Degree-** A degree of relationship represents the number of entity types that associate in a relationship
8. **Cardinality-** It refers to the uniqueness of a column in a table. For example, the primary key field will have a completely unique value for every record. Where there is a large percentage of unique values, this is known as “High Cardinality”. Where there are a lot of repeated values across the entity’s tuples, this is known as having “Low Cardinality”.
9. **Domain-** It is a unique set of values permitted for an attribute in a table. For example, a field may have an integer number data type, which defines that it can only allow whole numbers to be entered. However, there may be additional rules applied, such as that the number must be between 1 & 10. The domain would therefore be this range of whole numbers.



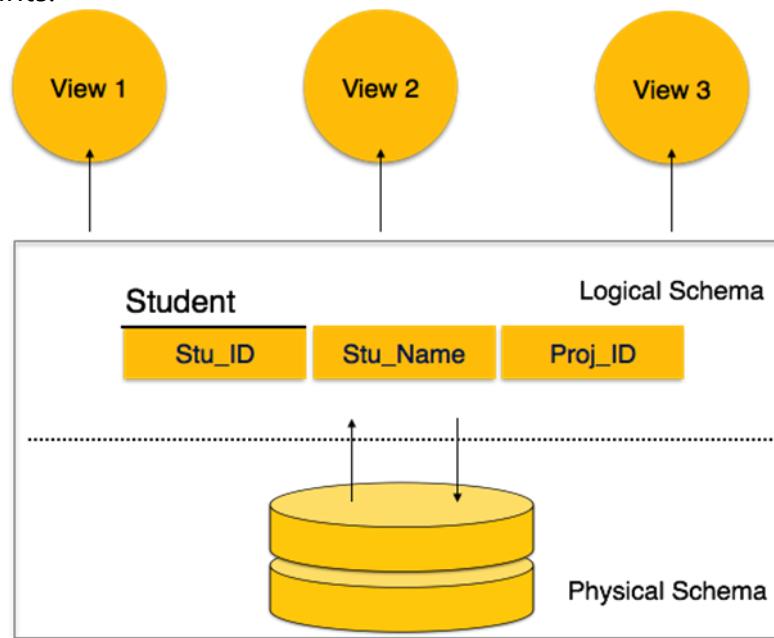
Reference- <https://www.w3schools.in/wp-content/uploads/2016/08/Relational-Model-Terms.png>

## 2.3 Database Schema

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied to the data. A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.

A database schema can be divided broadly into two categories –

1. **Physical Database Schema** – This schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.
2. **Logical Database Schema** – This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

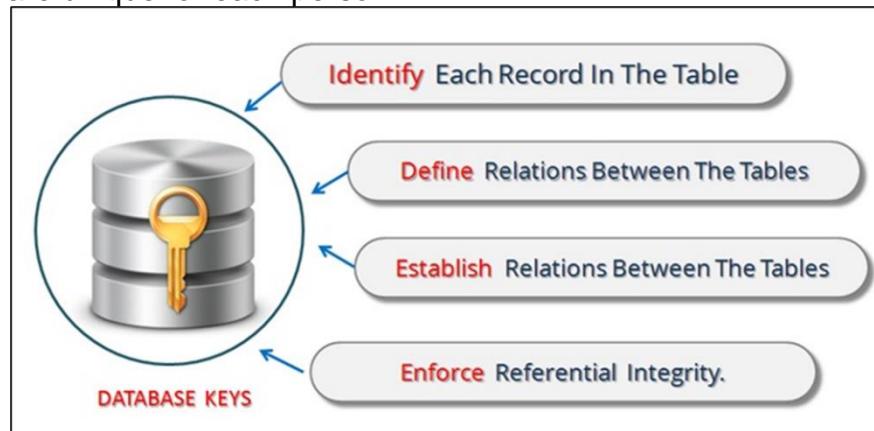


Reference: [https://www.tutorialspoint.com/dbms/dbms\\_data\\_schemas.htm](https://www.tutorialspoint.com/dbms/dbms_data_schemas.htm)

## What is the use of Database Keys?

Keys play an important role in the relational database. It is used to uniquely identify any record or row of data from the table. It is also used to establish and identify relationships between tables.

For example, ID is used as a key in the Student table because it is unique for each student. In the PERSON table, passport\_number, license\_number, SSN are keys since they are unique for each person.



Reference: <https://www.learncomputerscienceonline.com/database-keys/>

## Types of Keys in DBMS (Database Management System)

There are mainly eight different types of Keys in DBMS and each key has its different functionality:

1. Super Key
2. Primary Key
3. Candidate Key
4. Alternate Key
5. Foreign Key
6. Compound Key
7. Composite Key
8. Surrogate Key

### *What is the Super key?*

A super key is a group of single or multiple keys which identifies rows in a table. A Super key may have additional attributes that are not needed for unique identification.

Example:

EmpSSN	EmpNum	Empname
9812345098	AB05	Shown
9876512345	AB06	Roslyn
199937890	AB07	James

In the above-given example, EmpSSN and EmpNum name are superkeys.

### *What is a Primary Key?*

PRIMARY KEY in DBMS is a column or group of columns in a table that uniquely identifies every row in that table. The Primary Key can't be a duplicate meaning the same value can't appear more than once in the table. A table cannot have more than one primary key.

### **Rules for defining Primary key:**

- Two rows can't have the same primary key value
- It must for every row to have a primary key value.
- The primary key field cannot be null.
- The value in a primary key column can never be modified or updated if any foreign key refers to that primary key.

Example:

In the following example, `<code>StudID</code>` is a Primary Key.

StudID	Roll No	First Name	Last Name	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

### ***What is the Alternate key?***

ALTERNATE KEYS is a column or group of columns in a table that uniquely identifies every row in that table. A table can have multiple choices for a primary key but only one can be set as the primary key. All the keys which are not primary keys are called Alternate Key.

Example:

In this table, StudID, Roll No, and Email are qualified to become a primary key. But since StudID is the primary key, Roll No, Email becomes the alternative key.

StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

### ***What is a Candidate Key?***

CANDIDATE KEY in SQL is a set of attributes that uniquely identify tuples in a table. Candidate Key is a super key with no repeated attributes. The Primary key should be selected from the candidate keys. Every table must have at least a single candidate key. A table can have multiple candidate keys but only a single primary key.

### **Properties of Candidate key:**

- It must contain unique values
- Candidate key in SQL may have multiple attributes
- Must not contain null values
- It should contain minimum fields to ensure uniqueness
- Uniquely identify each record in a table

Example: In the given table Stud ID, Roll No, and email are candidate keys which help us to uniquely identify the student record in the table.

StudID	Roll No	First Name	LastName	Email
1	11	Tom	Price	abc@gmail.com
2	12	Nick	Wright	xyz@gmail.com
3	13	Dana	Natan	mno@yahoo.com

### ***What is the Foreign key?***

FOREIGN KEY is a column that creates a relationship between two tables. The purpose of Foreign keys is to maintain data integrity and allow navigation between two different instances of an entity. It acts as a cross-reference between two tables as it references the primary key of another table.

Example:

DeptCode	DeptName
001	Science
002	English
005	Computer

Teacher ID	Fname	Lname

B002	David	Warner
B017	Sara	Joseph
B009	Mike	Brunton

In this key in dbms example, we have two tables, teacher and department in a school. However, there is no way to see which search works in which department.

In this table, adding the foreign key in Deptcode to the Teacher name, we can create a relationship between the two tables.

Teacher ID	DeptCode	Fname	Lname
B002	002	David	Warner
B017	002	Sara	Joseph
B009	001	Mike	Brunton

This concept is also known as **Referential Integrity**.

#### *What is the Compound key?*

COMPOUND KEY has two or more attributes that allow you to uniquely recognize a specific record. It is possible that each column may not be unique by itself within the database. However, when combined with the other column or columns the combination of composite keys become unique. The purpose of the compound key in database is to uniquely identify each record in the table.

Example:

OrderNo	ProductID	Product Name	Quantity
B005	JAP102459	Mouse	5
B005	DKT321573	USB	10
B005	OMG446789	LCD Monitor	20
B004	DKT321573	USB	15
B002	OMG446789	Laser Printer	3

In this example, OrderNo and ProductID can't be primary keys as it does not uniquely identify a record. However, a compound key of OrderNo and ProductID could be used as it uniquely identified each record.

#### *What is the Composite key?*

COMPOSITE KEY is a combination of two or more columns that uniquely identify rows in a table. The combination of columns guarantees uniqueness, though individual uniqueness is not guaranteed. Hence, they are combined to uniquely identify records in a table.

The difference between a compound and a composite key is that any part of the compound key can be a foreign key, but the composite key may or maybe not be a part of the foreign key.

#### *What is a Surrogate key?*

SURROGATE KEYS is an artificial key that aims to uniquely identify each record. It is called a surrogate key. This kind of partial key in dbms is unique because it is created when you don't have any natural primary key. They do not lend any meaning to the data in the table. The surrogate key in DBMS is usually an integer. A surrogate key is a value generated right before the record is inserted into a table.

Fname	Lastname	Start Time	End Time
Anne	Smith	09:00	18:00
Jack	Francis	08:00	17:00
Anna	McLean	11:00	20:00
Shown	Willam	14:00	23:00

The above example shows the shift timings of the different employees. In this example, a surrogate key is needed to uniquely identify each employee.

Surrogate keys in sql are allowed when

- No property has the parameter of the primary key.
- In the table when the primary key is too big or complicated.

**Relationships-** A relational database contains tables that relate to another table by using a relationship. The tables are connected by a common field. The relationships are defined as:

- **One to Many:** This is the most common type of table relationship. For every record in Table A, there are multiple records in Table B.  
Example: There is a one-to-many relationship between the Customers table and Orders table. A customer may have many orders in the Order table.
- **Many to Many:** For every record in Table A, there are multiple records in Table B, and vice versa. In this case, a third table called a Join Table is created to which will contain only unique values  
Example: Many to Many relationships between Orders and Products with the table ProductsOrders functioning as the Join Table. The table ProductsOrders holds all the details about each order and what products it contains. Its primary key is a combination of the primary key of the Orders and Products table.
- **One to One –** The rarest type of table relationship. It is used for security and organization to avoid empty fields that only apply to some records in a table.

## Difference Between Primary key & Foreign key

Following is the main difference between primary key and foreign key:

Primary Key	Foreign Key
Helps you to uniquely identify a record in it is a field in the table that is the primary key of another table.	
Primary Key never accepts null values.	A foreign key may accept multiple null values.
The primary key is a clustered index and data in the DBMS table are physically organized in the sequence of the clustered index.	A foreign key cannot automatically create an index, clustered or non-clustered. However, you can manually create an index on the foreign key.
You can have the single Primary key in a table.	You can have multiple foreign keys in a table.

## 2.4 What is MySQL?

- MySQL is released under an open-source license.
- MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
- MySQL works very quickly and works well even with large data sets.
- MySQL supports large databases, up to 50 million rows or more in a table.
- MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.
- As with other relational databases, MySQL stores data in tables made up of rows and columns. Users can define, manipulate, control, and query data using Structured Query Language, more commonly known as SQL.

### Features of MySQL

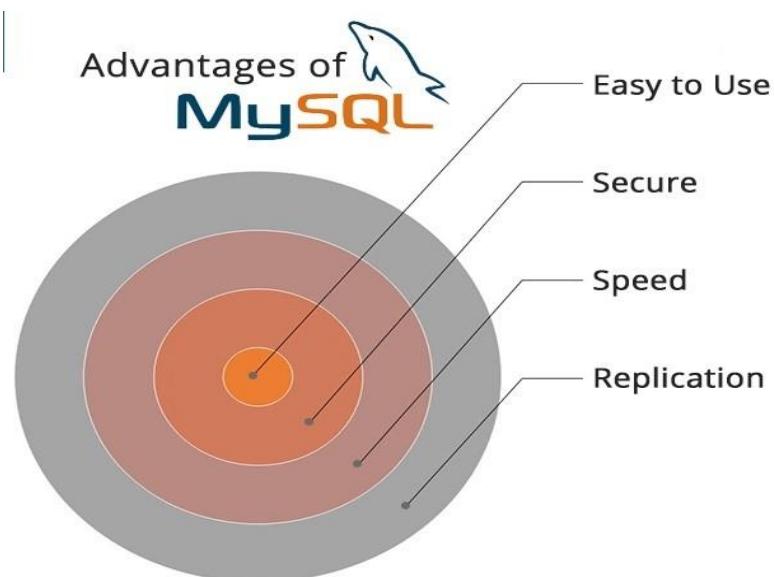
1. **Speed:** Of course, the speed at which a server-side program runs depends primarily on the server hardware. Given that the server hardware is optimal, MySQL runs very fast. It supports clustered servers for demanding applications.
2. **Ease of use:** MySQL is a high-performance, relatively simple database system. From the beginning, MySQL has typically been configured, monitored, and managed from the command line.
3. **MySQL Administrator:** This tool makes it possible for administrators to set up, evaluate, and tune their MySQL database server. This is intended as a replacement for mysqladmin.
4. **MySQL Query Browser:** Provides database developers and operators with a graphical database operation interface. It is especially useful for seeing multiple query plans and result sets in a single user interface.
5. **Configuration Wizard:** Administrators can choose from a predefined list of optimal settings, or create their own.
6. **MySQL System Tray:** Provides Windows-based administrators a single view of their MySQL instance, including the ability to start and stop their database servers.
7. **Cost:** MySQL is available free of cost. MySQL is an "Open Source" database. MySQL is part of LAMP (Linux, Apache, MySQL, PHP / Perl / Python) environment, a fast-growing open-source enterprise software stack. More and more companies are using LAMP as an alternative to expensive proprietary software stacks because of its lower cost, reliability, and documentation.
8. **Query Language Support:** MySQL understands standards-based SQL (Structured Query Language).
9. **Capability:** Many clients can connect to the server at the same time. Clients can use multiple databases simultaneously. You can access MySQL using several interfaces such as command-line clients, Web browsers.
10. **Connectivity and security:** MySQL are fully networked, and databases can be accessed from anywhere on the Internet, so you can share your data with anyone, anywhere. Connectivity could be achieved with Windows programs by using ODBC drivers. By using the ODBC connector to MySQL, any ODBC-

aware client application (for example, Microsoft Office, report writers, Visual Basic) can connect to MySQL.

11. **Portability:** MySQL runs on many varieties of UNIX, as well as on other non-UNIX systems, such as Windows and OS/2. MySQL runs on hardware from home PCs to high-end server. MySQL can be installed on Windows XP, Windows Server 2003, Red Hat Fedora Linux, Debian Linux, and others.

## Benefits of MySQL

1. **Flexible and easy to use:** Modify source code according to your own needs and expectations. The installation process is relatively simple, and doesn't take much time.
2. **High performance:** Even if you are storing massive amounts of big e-Commerce data or doing heavy business intelligence activities, MySQL can assist you smoothly with optimum speed.
3. **A mature DBMS:** Developers have been using MySQL for years, which means that there are abundant resources for them. It has evolved over the years and has very little margin for any kind of bug.
4. **Secure database:** Data security is the basic need of every web app. With its Access Privilege System and User Account Management, MySQL sets the security bar high. It offers both Host-based verification and password encryption as well.
5. **Free installation:** The community edition of MySQL is free to download. There are some prepaid options but if your company is too small to pay for them, the free-to-download model is the most suitable for a fresh start.
6. **Simple syntax:** MySQL's structure is very simple and plain. That's why developers even consider MySQL a database with a human-like language. MySQL is easy to use, and most of the tasks can be executed right in the command line, reducing development steps.



Reference: [https://www.janbasktraining.com/blog/uploads/images/image\\_750x\\_5dcd28ff0dfd4.jpg](https://www.janbasktraining.com/blog/uploads/images/image_750x_5dcd28ff0dfd4.jpg)

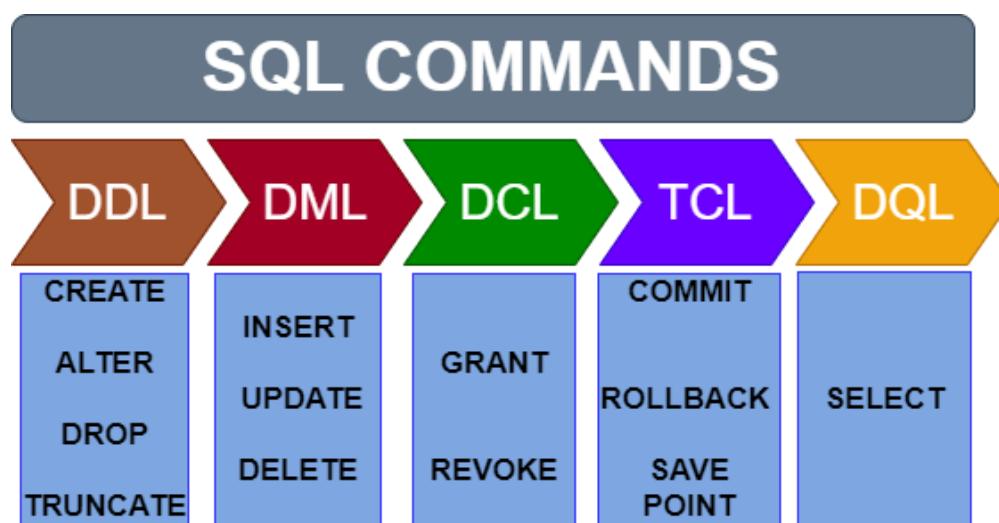
## Cons of MySQL

1. **Owned by Oracle:** MySQL used to be open source but now it's not completely open source. It's mostly now under Oracle's license which limits the MySQL community in terms of improving the DBMS.
2. **Scalability issues:** MySQL is not as efficiently scalable as the NoSQL database. It will need more engineering effort to make it scalable. So, if you have apps for which your database will increase substantially, you should choose another DBMS option.
3. **Limited support for SQL standards:** MySQL doesn't completely compliance with SQL standards. It does not provide support for some standard SQL features as well as it has some extensions and features that don't match the Structured Query Language standards.

## Types of SQL

Here are five types of widely used SQL queries.

1. Data Definition Language (DDL)
2. Data Manipulation Language (DML)
3. Data Control Language (DCL)
4. Transaction Control Language (TCL)
5. Data Query Language (DQL)



Reference: <https://powerbidocs.com/wp-content/uploads/2019/12/SQL-Commands-2018664638-1576589804123.png>

Let's see each of them in detail:

### What is DDL?

Data Definition Language helps you to define the database structure or schema. Let's learn about DDL commands with syntax.

Five types of DDL commands in SQL are:

- **CREATE** - CREATE statements is used to define the database structure schema:

Syntax:

```
CREATE      TABLE      TABLE_NAME      (COLUMN_NAME
DATATYPES[,....]);
```

For example:

```
Create database university;
Create table students;
Create view for_students;
```

- **DROP**- Drops commands remove tables and databases from RDBMS.

Syntax

```
DROP TABLE;
```

For example:

```
Drop object_type object_name;
Drop database university;
Drop table student;
```

- **ALTER**- Alters command allows you to alter the structure of the database.

Syntax:

To add a new column in the table

```
ALTER TABLE table_name ADD column_name COLUMN-definition;
```

To modify an existing column in the table:

```
ALTER TABLE MODIFY(COLUMN DEFINITION....);
```

For example:

```
Alter table guru99 add subject varchar;
```

- **TRUNCATE**- This command is used to delete all the rows from the table and free the space containing the table.

Syntax:

```
TRUNCATE TABLE table_name;
```

Example:

```
TRUNCATE table students;
```

## 2.5 What is DML?

Data Manipulation Language (DML) allows you to modify the database instance by inserting, modifying, and deleting its data. It is responsible for performing all types of data modification in a database. There are three basic constructs which allow database program and user to enter data and information are. Here are some important DML commands in SQL:

- INSERT
- UPDATE
- DELETE

- **INSERT**: This is a statement is a SQL query. This command is used to insert data into the row of a table.

Syntax:

```
INSERT INTO TABLE_NAME (col1, col2, col3,... col N)
VALUES (value1, value2, value3, .... valueN);
```

Or

```
INSERT INTO TABLE_NAME  
VALUES (value1, value2, value3, .... valueN);
```

For example:

```
INSERT INTO students (RollNo, FirstName, LastName) VALUES ('60',  
'Tom', Erichsen');
```

- **UPDATE:** This command is used to update or modify the value of a column in the table.

Syntax:

```
UPDATE table_name SET [column_name1= value1,...column_nameN  
= valueN] [WHERE CONDITION]
```

For example:

```
UPDATE students  
SET FirstName = 'Jhon', LastName= 'Wick'  
WHERE StudID = 3;
```

- **DELETE:** This command is used to remove one or more rows from a table.

Syntax:

```
DELETE FROM table_name [WHERE condition];
```

For example:

```
DELETE FROM students  
WHERE FirstName = 'Jhon';
```

## What is DCL?

DCL (Data Control Language) includes commands like GRANT and REVOKE, which are useful to give “rights & permissions.” Other permission controls parameters of the database system. Commands that come under DCL:

- Grant
  - Revoke
- **Grant:** This command is used to give user access privileges to a database.  
Syntax:  

```
GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER,  
ANOTHER_USER;
```

For example:  

```
GRANT SELECT ON Users TO'Tom'@'localhost';
```

- **Revoke:** It is useful to back permissions from the user.

Syntax:

```
REVOKE privilege_nameON object_nameFROM {user_name |PUBLIC  
|role_name}
```

For example:

```
REVOKE SELECT, UPDATE ON student FROM BCA, MCA;
```

## What is TCL?

Transaction control language or TCL commands deal with the transaction within the database.

- **Commit:** This command is used to save all the transactions to the database.  
Syntax:  
    Commit;  
For example:  
    DELETE FROM Students  
    WHERE RollNo =25;  
    COMMIT;
- **Rollback:** Rollback command allows you to undo transactions that have not already been saved to the database.  
Syntax:  
    ROLLBACK;  
Example:  
    DELETE FROM Students  
    WHERE RollNo =25;
- **SAVEPOINT:** This command helps you to set a savepoint within a transaction.  
Syntax:  
    SAVEPOINT SAVEPOINT\_NAME;  
Example:  
    SAVEPOINT RollNo;

## What is DQL?

Data Query Language (DQL) is used to fetch data from the database. It uses only one command:

- **SELECT:** This command helps you to select the attribute based on the condition described by the WHERE clause.  
Syntax:  
    SELECT expressions  
    FROM TABLES  
    WHERE conditions;  
For example:  
    SELECT FirstName  
    FROM Student  
    WHERE RollNo > 15;

Summarizing, data, to any business, is an asset. How we input, retrieve and use that data can be simply a time-consuming expense or make all the difference to the efficiency and cost saving of the business. A database is a collection of processed information related to a particular subject or purpose. Database management systems are programs that enable you to manage data on your computer. A computerised database management helps a user to store, change, find and present information from the database. The software used to store, manage, query, and retrieve data stored in a relational database is called a relational database management system. We also learned about MySQL, which is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company.

# Chapter 3: Building Front Face for the Web

## Learning Outcomes:

- Understand and use basic internet functionalities
- Understanding the architecture of a website
- Design front-end of a web page and modify its content through HTML and CSS
- Demonstrate understanding of client-side scripting

## 3.1 Understanding the Web

### Internet

*What is Internet?*

The Internet is essentially a global network of computing resources. You can think of the Internet as a physical collection of routers and circuits as a set of shared resources. Some common definitions given in the past include –

- A network of networks based on the TCP/IP communications protocol.
- A community of people who use and develop those networks.

The **Internet** is the biggest world-wide communication network of computers. The short form of **internet** is the 'net'. ... It is used by billions of people all over the world. It enables the users to send, receive, collect, store, update, delete, and many other operations of the data across the world. You can do all of this by connecting a computer to the Internet, which is also called **going online**. When someone says a computer is online, it's just another way of saying it's connected to the Internet.

### ***How does the Internet work?***

It's important to realize that the Internet is a global network of **physical cables**, which can include copper telephone wires, TV cables, and fibre optic cables. Even wireless connections like Wi-Fi and 3G/4G rely on these physical cables to access the Internet. When you visit a website, your computer sends a request over these wires to a **server**. A server is where websites are stored, and it works a lot like your computer's hard drive. Once the request arrives, the server retrieves the website and sends the correct data back to your computer.

### ***Internet-Based Services***

Some of the basic services available to Internet users are –

- **Email** – A fast, easy, and inexpensive way to communicate with other Internet users around the world.
- **Telnet** – Allows a user to log into a remote computer as though it were a local system.
- **FTP** – Allows a user to transfer virtually every kind of file that can be stored on a computer from one Internet-connected computer to another.
- **UseNet news** – A distributed bulletin board that offers a combination news and discussion service on thousands of topics.
- **World Wide Web (WWW)** – A hypertext interface to Internet information resources.

### ***History and Evolution of the Internet***

The Internet completely revolutionised communication and technology across the Globe. Initially, computerised devices were only used for large industries but later its usage increased massively. It is also mandatory for people to know that it is not possible for a single person to develop something as broad and wide as the Internet all by himself/herself. It was a combined effort of multiple researchers and programmers that the Internet was discovered.

Given below are a few important points which played an extremely important role in the development of the Internet and making it one of the most widely used resources across the world.

- The first development was the introduction of host-to-host network interactions. This was first observed in ARPANET in 1969. It was developed by Advanced

Research Projects Agency (APRA) of the Department of Defence, U.S. It was one of the first general usage of computer networks.

- Next step was commercialising the usage and making the transistors and transmitters fit in smaller devices for convenient Internet usage for the general public. This was introduced in the 1970s
- Moving forward, satellites and wireless communication was the main target. Defence Advanced Research Projects Agency (formerly ARPA), supported satellite-based radio packets for mobile usage of networks
- The next was the development of Transmission Control Protocol (TCP). This enabled different machines and networks across the world to assemble data packets. It was in the 1980s that the TCP/IP approach was adapted by researchers and technologists, following the footsteps of the U.S. Department of Defence
- With the introduction of personal computers, the demand for commercial Internet usage increased. This was the time when Ethernet and other Local Area Networks came in the foreground
- In 1993, the web browser was introduced, which followed the point-and-click approach and is now a widely used operation for Internet users
- The late 1990s was the time when thousands of Internet Service Providers has taken up the market and most of them were from the U.S.
- And then the 21st century brought in an amalgamation of technology and wireless Internet accessibility for its users. Wherein, wireless broadband services came in as a boon for Internet users

### ***Understanding HTTP Basics***

**HTTP** stands for hypertext transfer protocol and is used to transfer data across the Web. It is a critical protocol for web developers to understand and because of its widespread use it is also used in transferring data and commands in IOT applications.

### **How It Works**

Like most of the Internet protocols **http** it is a **command and response text-based** protocol using a **client server** communications model.

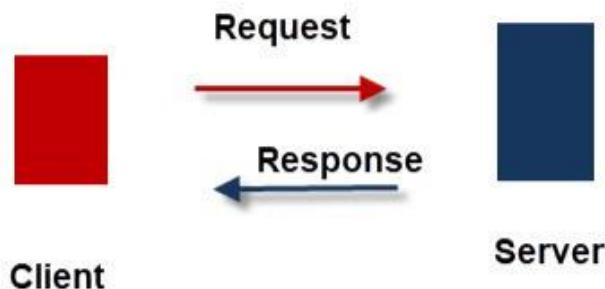


Image: HTTP protocol  
Reference: <http://www.steves-internet-guide.com/http-basics/>

The client makes a request and the server responds. The **HTTP protocol** is also a **stateless protocol** meaning that the server isn't required to store session information, and each request is independent of the other. This means:

- All requests originate at the client (your browser)
- The server responds to a request.
- The requests(commands) and responses are in readable text.
- The requests are independent of each other and the server **doesn't need to track** the requests.

### ***Pros and Cons of Using the Internet***

Intentionally or unintentionally, Internet usage is a part in the day to day lives of every individual. The Internet has made lives easy and comfortable, but at the same time made human being dependable for the smallest or biggest of information. Discussed below are the uses of the internet, along with a few cons that it brings along.

#### ***Pros of Internet***

- **Easy Access to Information** – Information on anything and everything are available online. The Internet makes it convenient to learn about new things at any point in time and get details on various subjects, irrespective of time and place
- **Platform for Online Education** – With the advanced technology, even students and adults wish to learn new things and gaining knowledge at various online portals has become more accessible
- **Job Hunting** – Employers can look for employees on the internet and the job seekers can apply online for jobs using the Internet
- **Platform to become an entrepreneur** – Today, thousands of people have started their own websites and getting good business and users/customers by making their own websites and selling products or services. This has become accessible due to Internet connectivity
- **Visual and Graphical Representation of Things** – Various researches have shown that a person tends to get more engaged with a graphical representation of things. Internet has made this facility also convenient for both user and creator
- **Reduced the parameter of Distance** – Social-media has reduced the distance between people as communication has become much easier because of Internet connection

With the Internet being an extremely essential part of daily life, it is important that a person is well aware of the disadvantages of the Internet and its excess usage.

#### ***Cons of Internet***

- **Dependency** – The dependency of people for looking things and information online has increased massively since the introduction of Internet and its easy access
- **Cyber Crime** – People do not just use internet for learning purposes, cybercrime has also been at a distinctive high because of effortless availability of resources

- **Distraction** – People can easily find online games, interesting information, etc. online which may be a cause of distraction for many.
- **Bullying and Trolls** – Online platforms are being used for unethical practises like bullying people and trolling them.

## Web Page

Web page is a document available on world wide web. Web Pages are stored on web server and can be viewed using a web browser. A web page can contain huge information including text, graphics, audio, video and hyperlinks. These hyperlinks are the link to other web pages. Collection of linked web pages on a web server is known as website. There is unique Uniform Resource Locator (URL) is associated with each web page. A **web page** is a simple document displayable by a browser. Such documents are written in the HTML language. A web page can embed a variety of different types of resources such as:

- *style information* — controlling a page's look-and-feel
- *scripts* — which add interactivity to the page
- *media* — images, sounds, and videos.

**Note:** Browsers can also display other documents such as PDF files or images, but the term **web page** specifically refers to HTML documents. Otherwise, we only use the term **document**.

All web pages available on the web are reachable through a unique address. To access a page, just type its address in your browser address bar:

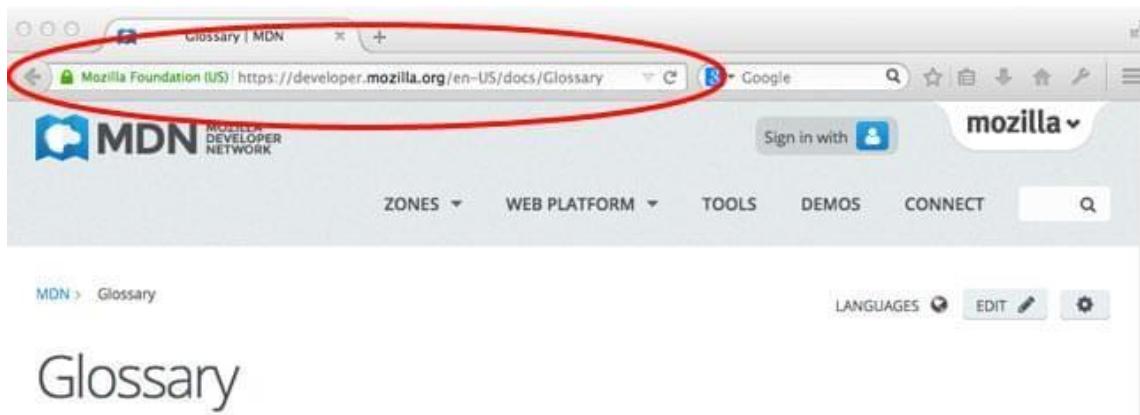


Image: Web Page

Reference:<https://developer.mozilla.org/enUS/docs/Learn/Commonquestion/Pagessitesserversandsearchengines>

## What is Hyperlink?

A hyperlink or simply a link is a selectable element in an electronic document that serves as an access point to other electronic resources. Typically, you click the hyperlink to access the linked resource. Familiar hyperlinks include buttons, icons, image maps, and clickable text links.

## Type of Web Pages

### Static Web page

Static web pages are also known as flat or stationary web page. They are loaded on the client's browser as exactly they are stored on the web server. Such web pages contain only static information. User can only read the information but can't do any modification or interact with the information.

Static web pages are created using only HTML. Static web pages are only used when the information is no more required to be modified.

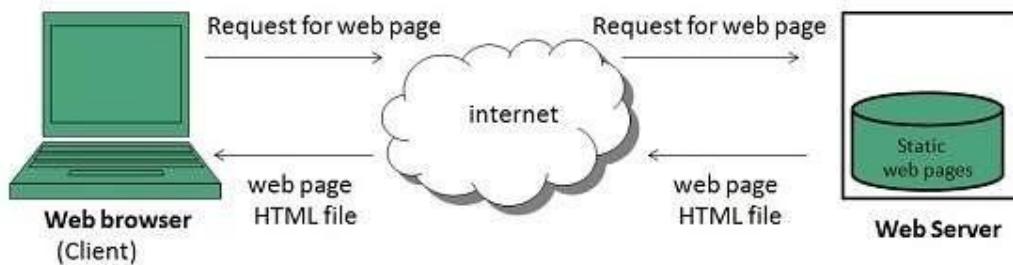


Image: Static Web Page

Reference: [https://www.tutorialspoint.com/internet\\_technologies/web\\_pages.htm](https://www.tutorialspoint.com/internet_technologies/web_pages.htm)

### Dynamic Web page

Dynamic web page shows different information at different point of time. It is possible to change a portion of a web page without loading the entire web page. It has been made possible using Ajax technology.

#### Server-side dynamic web page

It is created by using server-side scripting. There are server-side scripting parameters that determine how to assemble a new web page which also includes setting up of more client-side processing.

#### Client-side dynamic web page

It is processed using client-side scripting such as JavaScript. And then passed in to Document Object Model (DOM).

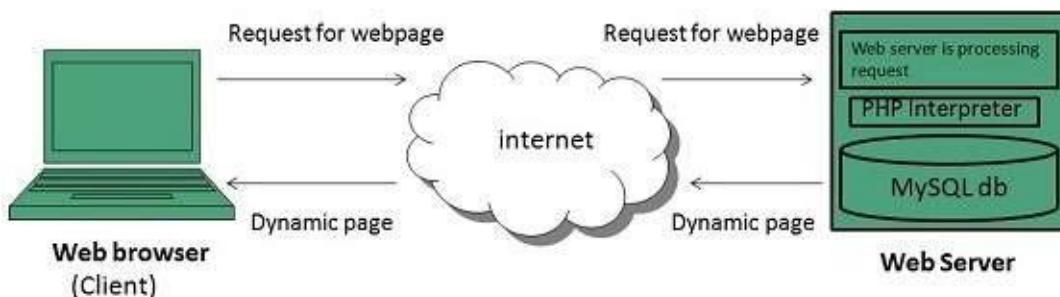


Image: Dynamic Web Page

Reference: [https://www.tutorialspoint.com/internet\\_technologies/web\\_pages.htm](https://www.tutorialspoint.com/internet_technologies/web_pages.htm)

## Website

A collection of web pages which are grouped together and usually connected together in various ways. Often called a "web site" or a "site".

A *website* is a collection of linked web pages (plus their associated resources) that shares a unique domain name. Each web page of a given website provides explicit links—most of the time in the form of clickable portion of text—that allow the user to move from one page of the website to another.

To access a website, type its domain name in your browser address bar, and the browser will display the website's main web page, or *homepage* (casually referred as "the home"):

### What is ISP?

ISP stands for Internet Service Provider. They are the companies who provide you service in terms of internet connection to connect to the internet. You will buy space on a Web Server from any Internet Service Provider. This space will be used to host your website.

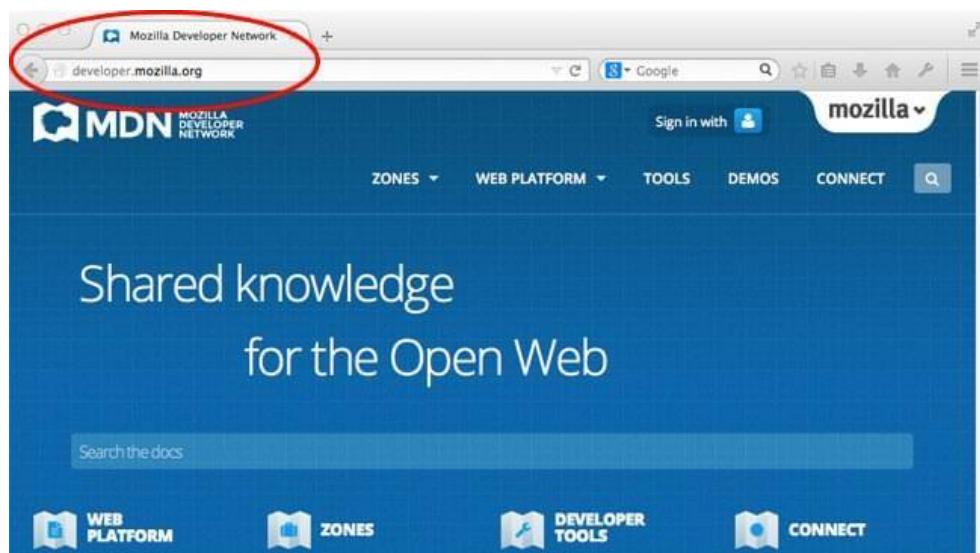


Image: Internet Service Provider

Reference:[https://developer.mozilla.org/enUS/docs/Learn/Common\\_questions/Pages\\_sites\\_servers\\_and\\_search\\_engines](https://developer.mozilla.org/enUS/docs/Learn/Common_questions/Pages_sites_servers_and_search_engines)

The ideas of a *web page* and a *website* are especially easy to confuse for a *website* that contains only one *web page*. Such a website is sometimes called a *single-page website*.

A website is a collection of publicly accessible, interlinked Web pages that share a single domain name. Websites can be created and maintained by an individual, group, business or organization to serve a variety of purposes. Together, all publicly accessible websites constitute the World Wide Web. Although it is sometimes called "web page," this definition is wrong, since a website consists of several webpages. A website is also known as a "web presence" or simply "site".

### Why you need a Website?

Here, are prime reasons why you need a website:

- An effective method to showcase your products and services
- Developing a site helps you to create your social proof

- Helps you in branding your business
- Helps you to achieve your business goals
- Allows you to increase your customer support.

## Web Application

A web application is a software or program which is accessible using any web browser. Its frontend is usually created using languages like HTML, CSS, Javascript, which are supported by major browsers. While the backend could use any programming stack like LAMP, MEAN, etc. Unlike mobile apps, there is no specific SDK for developing web applications.

### Why you need a Web Application?

Web applications are more popular because of the following reasons:

- Compared to desktop applications, web applications are easier to maintain as they use the same code in the entire application. There are no compatibility issues.
- Web applications can be used on any platform: Windows, Linux, Mac... as they all support modern browsers.
- Mobile App store approval not required in web applications.
- Released any time and in any form. No need to remind users to update their applications.
- You can access these web applications 24 hours of the day and 365 days a year from any PC.
- You can either make use of the computer or your mobile device to access the required data.
- Web applications are a cost-effective option for any organization. Seat Licenses for Desktop software are expensive where SaaS, are generally, pay as you go.
- Web-Based Apps are Internet-enabled apps that are accessed through the mobile's web browser. Therefore, you don't require downloading or installing them.

Here is how a web application works:

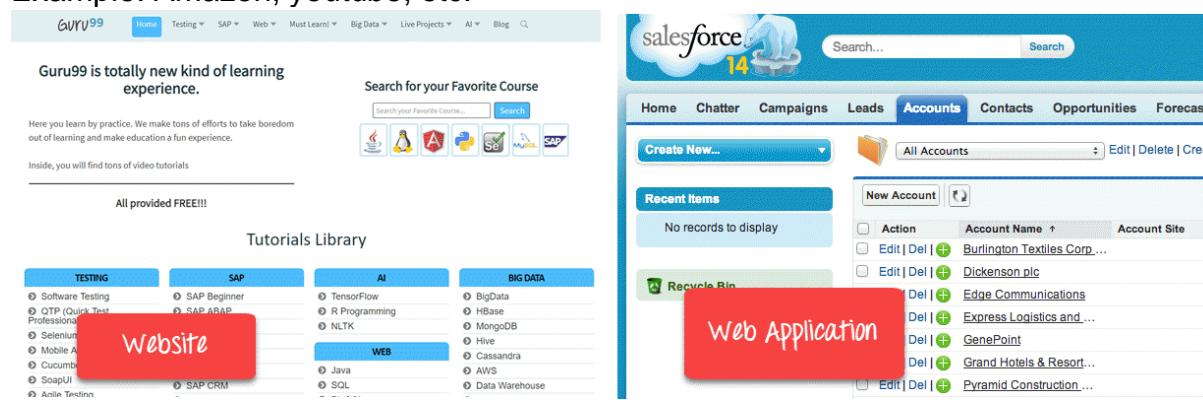
1. The user creates a request to the web server over the Internet through the application's user interface.
2. The web server sends this request to the web application server.
3. The web application server executes the requested task, then generates the results of the required data.
4. The web application server sends those results back to the web server (requested information or processed data).
5. The web server carries the requested information to the client (tablet, mobile device or desktop).
6. The requested information appears on the user's display.

### Difference Between Web application and Website

**Web Application:** Web application is a piece of software that can be accessed by the browser. A Browser is an application that is used to browse the internet.

Web application needs authentication. The web application uses a combination of server-side scripts and client-side scripts to present information. It requires a server to manage requests from the users. Example: Google Apps

**Website:** Website is a collection of related web pages that contains images, text, audio, video, etc. It can consist of one page, two pages, and n number of pages. A website provides visual and text content that users can view and read. To view a website requires a browser (chrome, firefox). There are many types of websites like Archive website, Blog, Community website, Dating website, etc. Example: Amazon, youtube, etc.



The image shows two side-by-side screenshots. On the left is the Guru99 website, which features a navigation bar with links like 'Home', 'Testing', 'SAP', 'Web', 'Must Learn', 'Big Data', 'Live Projects', 'AI', 'Blog', and 'Search'. Below the navigation is a section titled 'Guru99 is totally new kind of learning experience.' followed by a brief description and a 'Search for your Favorite Course' bar. A 'Tutorials Library' section lists categories: TESTING, SAP, AI, and BIG DATA, each with a list of sub-topics. A red box highlights the 'Website' category under TESTING. On the right is the Salesforce application interface, showing a navigation bar with 'Home', 'Chatter', 'Campaigns', 'Leads', 'Accounts', 'Contacts', 'Opportunities', and 'Forecast'. The 'Accounts' tab is selected, displaying a list of accounts such as 'Burlington Textiles Corp...', 'Dickenson plc', 'Edge Communications', 'Express Logistics and...', 'GenePoint', 'Grand Hotels & Resort...', and 'Pyramid Construction...'. A red box highlights the 'Web Application' entry in the list.

Image: Web Site vs Web Application  
Reference: <https://www.guru99.com/difference-web-application-website.html>

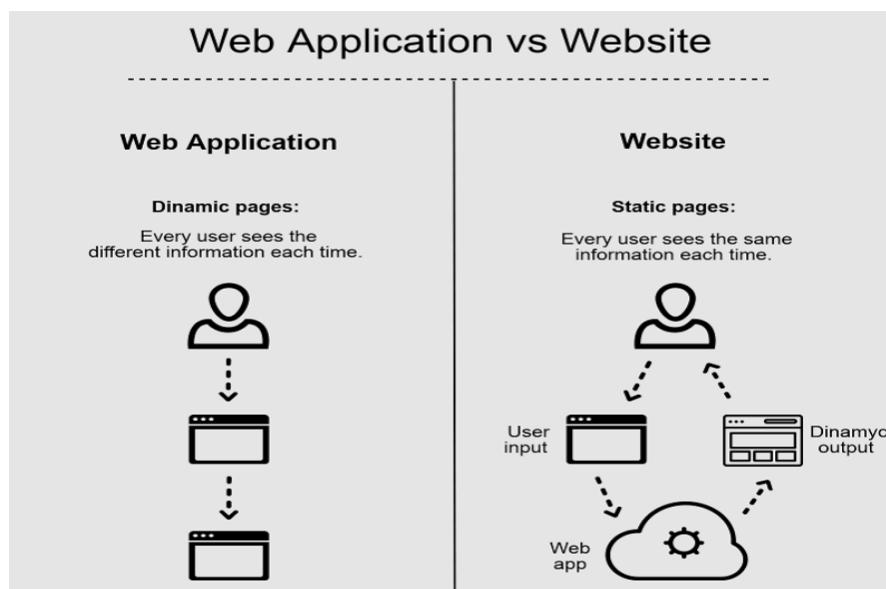


Image: Web Site vs Web Application  
Reference: <https://www.guru99.com/difference-web-application-website.html>

Parameter	Web Application	Website
Created for	A web application is designed for interaction with the end user.	A website mostly consists of static content. It is publicly accessible to all the visitors.
User interaction	In a web application, the user not only read the page content but also manipulate the restricted data.	A website provides visual & text content which user can view and read, but not affect its functioning.
Authentication	Web applications need authentication, as they offer a much broader scope of options than websites.	Authentication is not obligatory for informational websites. The user may ask to register to get a regular update or to access additional options. This feature is not available for the unregistered website visitors.
Task and Complexity	Web application functions are quite higher and complex compared to a website.	The website displays the collected data and information on a specific page.
Type of software	The web application development is part of the website. It is itself not a complete website.	The website is a complete product, which you access with the help of your browser.
Compilation	The site must be precompiled before deployment.	The site doesn't need to be pre-compiled.
Deployment	All changes require the entire project to be re-compiled and deployed.	Small changes never require a full re-compilation and deployment. You just need to update the HTML code.

Image: Web Site vs Web Application

Reference: <https://www.guru99.com/difference-web-application-website.html>

## Practical Activities

**Open/run the HTML file in a web browser to check the output.**

Steps to run the HTML file in a web browser:

- 1) Press "Windows-E" to launch Windows Explorer.
- 2) Navigate to the folder that contains your HTML file.
- 3) Double-click the file. Your default browser displays the HTML document. If the browser is not open, Windows launches it.

**Program:**

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>Black Goose Bistro</title>
</head>
<body text="blue">
<h1>Black Goose Bistro</h1>
<hr color="green">
<h2 text="blue">The Restaurant</h2>
<p>The Black Goose Bistro offers casual lunch and dinner fare in a hip atmosphere. The menu changes regularly to highlight the freshest ingredients.</p>
<h2>Catering</h2>
```

You have fun... we'll do the cooking. Black Goose catering can handle events from snacks for bridge club to elegant corporate fundraisers.

```
<h2>Location and Hours</h2>
Seekonk, Massachusetts; Monday through Thursday 11am to 9pm, Friday and Saturday, 11am to midnight
```

```
</body>
</html>
```

**Output/Results snippet:**



## Black Goose Bistro

### The Restaurant

The Black Goose Bistro offers casual lunch and dinner fare in a hip atmosphere. The menu changes regularly to highlight the freshest ingredients.

### Catering

You have fun... we'll do the cooking. Black Goose catering can handle events from snacks for bridge club to elegant corporate fundraisers.

### Location and Hours

Seekonk, Massachusetts; Monday through Thursday 11am to 9pm, Friday and Saturday, 11am to midnight

## 3.2 HTML for web Layout

### HTML Basic Components

**HTML** stands for Hyper Text Markup Language, is the standard markup language used to create web pages. Along with CSS, and JavaScript, HTML is a cornerstone technology used to create web pages, as well as to create user interfaces for mobile and web applications.

- ✓ HTML stands for Hyper Text Markup Language
- ✓ HTML elements are the building blocks of HTML pages
- ✓ HTML describes the structure of Web pages using HyperText markup language
- ✓ HTML elements are represented by tags
- ✓ When you test your page on browsers. Browsers do not display the HTML tags, but use them to render the content of the page.
- ✓ HTML page extension always will be .html (example.html)

### HTML Files

Every web page is actually a HTML file. Each HTML file is just a plain-text file, but with a (.html) file extension instead of (.txt,) and is made up of many HTML tags as well as the content for a web page. A web site will often contain many html files that link to each other. You can edit HTML files with your favourite editor. Like Dreamweaver, Notepad etc.

### Creating HTML Document:

Creating an HTML document is easy. To begin coding in HTML you need only two things: a simple-text editor and a web browser. Notepad is the most basic of simple-text editors and you will probably code a fair amount of HTML with it.

Open Notepad or another text editor.

- At the top of the page type <html>.
- On the next line, indent spaces and now add the opening header tag: <head>.
- On the next line, indent ten spaces and type <title> </title>.
- Go to the next line, indent five spaces from the margin and insert the closing header tag: </head>.
- Five spaces in from the margin on the next line, type <body>.
- Now drop down another line and type the closing tag right below its mate: </body>.

- Finally, go to the next line and type </html>.
- In the File menu, choose Save As.
- In the Save as Type option box, choose All Files.
- Name the file template.htm.
- Click Save.

### HTML Example

```
<!DOCTYPE html>
<html>
<body>
<h1>HTML Tutorial</h1>
<p>HTML Introduction</p>
</body>
</html>
```

### HTML Page Structure

```
<html>
<head>
<title>Page title</title>
</head>
<body>
<h1>This is a heading</h1>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
</body>
</html>
```

#### <DOCTYPE! html>:

This tag is used to tell the HTML version. This currently tells that the version is HTML 5.

#### <html>:

This is called HTML root element and used to wrap all the code.

#### <head>:

Head tag contains metadata, title, page CSS etc. All the HTML elements that can be used inside the <head> element are:

- <style>
- <title>
- <base>
- <noscript>
- <script>
- <meta>
- <title>

#### <body>:

Body tag is used to enclose all the data which a web page has from texts to links. All of the content that you see rendered in the browser is contained within this element.

Example: HTML page can be created using any text editor (notepad). Then save that file using .htm or .html extension and open that file in browser. It will get the HTML page response.

### Why we need to learn HTML?

- It is a simple mark-up language. Its implementation is easy.
- It is used to create a website.
- Helps in developing fundamentals about web programming.
- Boost professional career.

### Features of HTML:

- It is easy to learn and easy to use.
- It is platform independent.
- Images, video and audio can be added to a web page.
- Hypertext can be added to text.
- It is a markup language.

### Advantages:

- HTML is used to build websites.
- It is supported by all browsers.
- It can be integrated with other languages like CSS, JavaScript etc.

### Disadvantages:

- HTML can create only static webpages so for dynamic web page other languages have to be used.
- Large amount of code has to be written to create a simple web page.
- Security feature is not good.

## List

HTML List Tags are used to specify information in the form of list.

HTML Lists are very useful to group related information together. Often List items look well-structured and they are easy to read for users. A list can contain one or more list elements.

<b>HTML List Types</b>		
<b>List Type</b>	<b>Description</b>	<b>Tags used</b>
Unordered List	used to group a set of items without any order.	<ul>,<li>
Ordered List	used to group a set of items, in a specific order.	<ol>,<li>
Definition List	used to display some definition term (dt) & definition's description (dd)	<dl>,<dt>,<dd>

### Unordered lists

Unordered lists are used to list set of items when they have no special order or sequence. It is also called as bulleted list. Unordered list is created using HTML <ul> tag. Each item in the list start with the <li> tag

### Example of Unordered List

```
<ul>
<li>Red</li>
<li>Green</li>
<li>Blue</li>
</ul>
```

## List

## Tags

**<li>** tag is used to display list elements and it is used in ordered and unordered list. Above example will list items with bullets (small black circles) by default. There are different list styles available such as bullet, circle etc.

### Unordered List Style Type Attribute

Like ordered list, **type** attribute is used to customize bullet style for the list of elements. By default, a solid circle is used as bullets.

Type value: Numbering style

Disc: A solid circle

Square: A solid square

Circle: An unfilled circle

HTML List Types		
List Style Type	Description	Example & Syntax
disc	Starts a list using discs type bullets (default)	<ul type="disc">
circle	Starts a list using circle type bullets	<ul type="circle">
square	Starts a list using square type bullets	<ul type="square">
none	Starts a list without bullets	<ul type="type:none">

### Example of Unordered List with Different List Styles

```
<html>
<title> Unordered List Example Test - Tutorials Class </title>
<body>
```

```
<h2>Unordered List of Fruits with Disc Bullets</h2>
<ul type="disc">
<li>Apple</li>
<li>Banana</li>
<li>Mango</li>
</ul>
```

```
<h2>Unordered List of Colors with Circle Bullets</h2>
<ul type="circle">
<li>Red</li>
<li>Green</li>
<li>Blue</li>
</ul>
```

```
<h2>Unordered List of Fruits with Square Bullets</h2>
<ul type="square">
<li>Apple</li>
```

```
<li>Banana</li>
<li>Mango</li>
</ul>

<h2>Unordered List of Colors without bullets</h2>
<ul type="none">
<li>Black</li>
<li>Green</li>
<li>Blue</li>
</ul>

</body>
</html>
```

### Ordered lists

Ordered list is used to list related items in a numbered or other specific order. This is useful when you want to show counts of items in some way. Ordered list is created using HTML `<ol>` tag. Each item in the list start with the `<li>` tag.

#### Example of Ordered List

```
<ul>
<li>Red</li>
<li>Green</li>
<li>Blue</li>
</ul>
```

Above example will list colors items with numbers by default. There are different list styles available for ordered list such as numbers, letters etc.

#### Ordered List Style Type Attribute

There are two attributes can be used to customize ordered list, they are

- (1) **Type - changing numbering style**
- (2) **Start - changing numbering order.**

**Type** – is used to change the number style. The default number style is standard Arabic numerals (1, 2, 3, .....).

**Start** – is used to specify the number of letters with which start the list. The default starting point is 1. The value of the start attribute should be a decimal number, regardless of the numbering style being used

<b>HTML Ordered List Style Type Attribute</b>		
<b>List Style Type</b>	<b>Description</b>	<b>Example and Syntax</b>
Numbers	Starts a list using numbers (default)	<code>&lt;ol type="1"&gt;</code>
Uppercase letters	Starts a list using uppercase letters	<code>&lt;ol type="A"&gt;</code>
Lowercase letters	Starts a list using lowercase letters	<code>&lt;ol type="a"&gt;</code>

Uppercase numbers	roman	Starts a list using uppercase roman numbers	<ol type="I">
Lowercase numbers	roman	Starts a list using lowercase roman numbers	<ol type="i">

```
<html>
<title> Ordered List Example - Tutorials Class </title>
<body>

<h2>Ordered List of Fruits with Numbers </h2>
<ol type="1">
  <li>Banana</li>
  <li>Apple</li>
  <li>Grapes</li>
</ol>

<h2>Ordered List of Fruits with Uppercase letters</h2>
<ol type="A">
  <li>Apple</li>
  <li>Banana</li>
  <li>Mango</li>
</ol>

<h2>Ordered List of Colors with Lowercase letters</h2>
<ol type="a">
  <li>Red</li>
  <li>Green</li>
  <li>Blue</li>
</ol>

<h2>Ordered List of Colors with Uppercase roman numbers</h2>
<ol type="I">
  <li>Red</li>
  <li>Green</li>
  <li>Blue</li>
</ol>

<h2>Ordered List of Colors with Lowercase roman numbers</h2>
<ol type="i">
  <li>Red</li>
  <li>Green</li>
  <li>Blue</li>
</ol>

</body>
</html>
```

## Definition Lists

Definition list is different from other two types of lists. No bullet or number is provided for the list items. In this list type, the list element has two parts.

- (1) A definition terms
- (2) The definition description

Definition list is surrounded within <DL> ..... </DL> tags.

Definition term is presented in between <DT> ..... </DT> tag and

Definition description should be surrounded within <DD> ..... </DD> tag.

```
<dl>
  <dt>Computer</dt>
  <dd>Computer is an electronic device that is designed to work with Information.
</dd>
  <dt>HTML</dt>
  <dd>HyperText Markup Language (HTML) is the standard markup language for
creating web pages and web applications. </dd>
</dl>
```

## Tables

You can create tables for your website using the <table> tag in conjunction with the <tr>, <td> and <th> tags. The HTML tables allow displaying the data (e.g. image, text, link) in columns and rows of cells. Table rows may be grouped into a head, foot, and body sections (via the <thead>, <tfoot> and <tbody> elements, respectively).

The <caption> tag defines a table caption. The <caption> tag must be inserted immediately after the <table> tag.

**Note:** You can specify only one caption per table.

Cellpadding - The **cell padding** attribute places **spacing** around data within each **cell**.

Cellspacing - The **cell spacing** attribute places space around each **cell** in the table

Th	- Table Head
Tr	- Table Row
Td	- Table Data

### thead, tbody, and tfoot elements

```
<!DOCTYPE html>
<html>
<head>
<style>
thead {color:green;}
tbody {color:blue;}
tfoot {color:red;}

table, th, td {
  border: 1px solid black;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1>The thead, tbody, and tfoot elements</h1>
```

```
<table>
```

```
<thead>
```

```
<tr>
```

```
    <th>Month</th>
```

```
    <th>Savings</th>
```

```
</tr>
```

```
</thead>
```

```
<tbody>
```

```
<tr>
```

```
    <td>January</td>
```

```
    <td>$100</td>
```

```
</tr>
```

```
<tr>
```

```
    <td>February</td>
```

```
    <td>$80</td>
```

```
</tr>
```

```
</tbody>
```

```
<tfoot>
```

```
<tr>
```

```
    <td>Sum</td>
```

```
    <td>$180</td>
```

```
</tr>
```

```
</tfoot>
```

```
</table>
```

```
</body>
```

```
</html>
```

Month	Savings
January	\$100
February	\$80
Sum	\$180

The `<td>` elements are the data containers of the table. They can contain all sorts of HTML elements; text, images, lists, other tables, etc.

## Simple Table

```
<!DOCTYPE html>
<html>
  <head>
    <title>Title of the document</title>
    <style>
      table, th, td {
        border: 1px solid #666;
      }
    </style>
  </head>
  <body>
    <table style="width:80%;">
      <tr>
        <th>Month</th>
        <th>Date</th>
      </tr>
      <tr>
        <td>January</td>
        <td>10.01.2014</td>
      </tr>
      <tr>
        <td>February</td>
        <td>10.01.2014</td>
      </tr>
    </table>
  </body>
</html>
```

Month	Date
January	10.01.2014
February	10.01.2014

ISBN	Title	Price
3476896	My first HTML	\$53
5869207	My first CSS	\$49

## Graphics

### What are Graphics?

Web graphics are visual representations used on a Web site to enhance or enable the representation of an idea or feeling, in order to reach the Web site user. Graphics may entertain, educate, or emotionally impact the user, and are crucial to strength of branding, clarity of illustration, and ease of use for interfaces.

Examples of graphics include maps, photographs, designs and patterns, family trees, diagrams, architectural or engineering blueprints, bar charts and pie charts, typography, schematics, line art, flowcharts, and many other image forms.

Graphic designers have many tools and technologies at their disposal for everything from print to Web development, and W3C provides many of the underlying formats that can be used for the creation of content on the open Web platform.

## What are Graphics Used For?

Graphics are used for everything from enhancing the appearance of Web pages to serving as the presentation and user interaction layer for full-fledged Web Applications.

Different use cases for graphics demand different solutions, thus there are several different technologies available. Photographs are best represented with PNG, while interactive line art, data visualization, and even user interfaces need the power of SVG and the Canvas API. CSS exists to enhance other formats like HTML or SVG. WebCGM meets the needs for technical illustration and documentation in many industries.

### Graphic Elements in HTML5

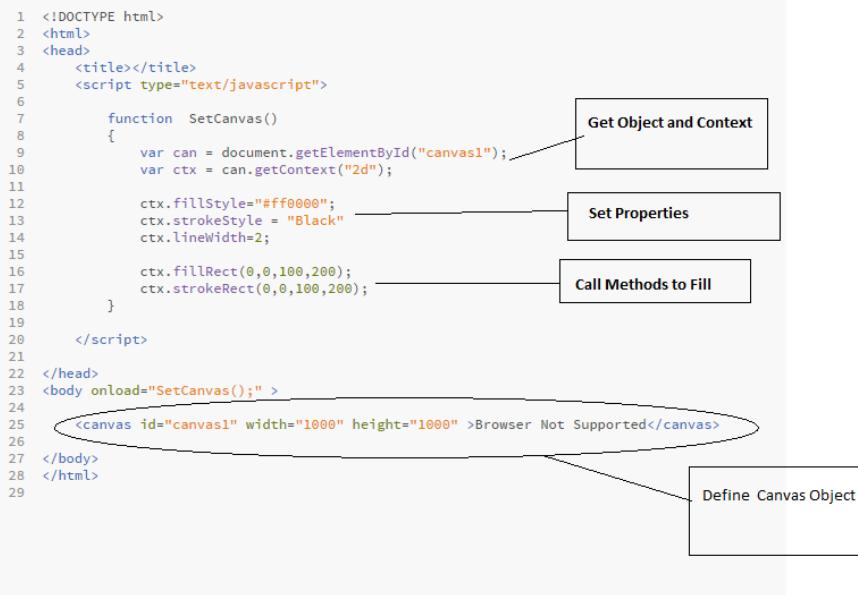
- Canvas
- Scalable Vector Graphics (SVG)

#### Canvas Element

The <canvas> element:

- helps the browser to draw shapes and images without any plugin.
- is used to draw graphics, on the fly, via scripting.
- has several methods for drawing paths, boxes, circles, characters and adding images.

The following is a sample of drawing a rectangle using a Canvas Element:



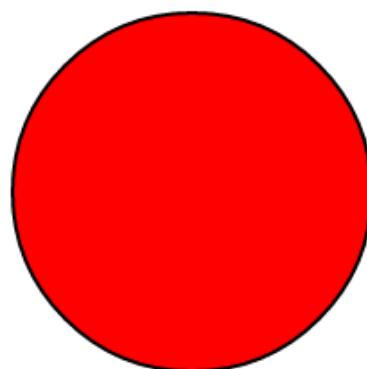


The following is a sample of drawing a circle using a Canvas Element:

```
Ch1:
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <title></title>
5      <script type="text/javascript">
6
7          function SetCanvas()
8          {
9              var can = document.getElementById("canvas1");
10             var ctx = can.getContext("2d");
11
12             ctx.fillStyle="#ff0000";
13             ctx.strokeStyle = "Black"
14             ctx.lineWidth=2;
15
16
17             ctx.arc(200,200,100,0,2* Math.PI);
18             ctx.fill();
19             ctx.stroke();
20
21         }
22
23     </script>
24
25 </head>
26 <body onload="SetCanvas();">
27     <canvas id="canvas1" width="1000" height="1000" >Browser Not Supported</canvas>
28
29 </body>
30 </html>
31
32
```

Diagram annotations:

- A callout box labeled "Get Object and Context" points to the line `var can = document.getElementById("canvas1");`.
- A callout box labeled "Set Properties" points to the line `ctx.fillStyle="#ff0000";`.
- A callout box labeled "Call methods to fill" points to the lines `ctx.arc(200,200,100,0,2* Math.PI);`, `ctx.fill();`, and `ctx.stroke();`.
- A callout box labeled "Define Canvas Object" points to the line `<canvas id="canvas1" width="1000" height="1000" >Browser Not Supported</canvas>`.



## Scalable Vector Graphics (SVG)

### A <SVG> element:

- is based on vector-based family of graphics.
- defines graphics in XML Format.
- creates graphics that does not lose any quality when zoomed or resized

The following is a sample of drawing a rectangle using an <SVG> Element:

```
1  <!DOCTYPE HTML>
2  <html>
3  <head>
4
5  <title>Scalar Vector Graphics
6  </title>
7  </head>
8
9  <body>
10
11 <svg height="500" width="500" >
12   <rect x="50" y="100" height="100" width = "200" fill="red" stroke="black" stroke-width="2" />
13 </svg>
14
15 </body>
16
17 </html>
```

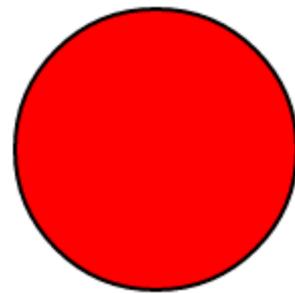
This create object with inbuilt properties. No Need for any Java Script



The following is a sample of drawing a circle using an SVG Element:

```
1  <!DOCTYPE HTML>
2  <html>
3  <head>
4
5  <title>Scalar Vector Graphics
6  </title>
7  </head>
8
9  <body>
10
11 <svg height="500" width="500">
12   <circle cx="100" cy="100" r="70" fill="red" stroke="black" stroke-width="2" />
13 </svg>
14
15 </body>
16
17 </html>
```

This create object with inbuilt properties. No Need for any java script.



## Differences between Canvas and SVG elements

Canvas	svg
Canvas draws 2D graphics, on the fly (with a JavaScript).	SVG defines the graphics in XML format.
Resolution dependent.	Resolution independent.
Canvas is rendered pixel by pixel.	In SVG, each drawn shape is remembered as an object.

## Multimedia

### What is Multimedia?

Multimedia comes in many different formats. It can be almost anything you can hear or see, like images, music, sound, videos, records, films, animations, and more. Web pages often contain multimedia elements of different types and formats.

### Browser Support

The first web browsers had support for text only, limited to a single font in a single color. Later came browsers with support for colors, fonts, images, and multimedia!

### Multimedia Formats

Multimedia elements (like audio or video) are stored in media files. The most common way to discover the type of a file, is to look at the file extension. Multimedia files have formats and different extensions like: .wav, .mp3, .mp4, .mpg, .wmv, and .avi.

### Audio format



Image: HTML Audio Format

Reference: <https://www.sitesbay.com/html5/html5-audio-tag>

### Video format

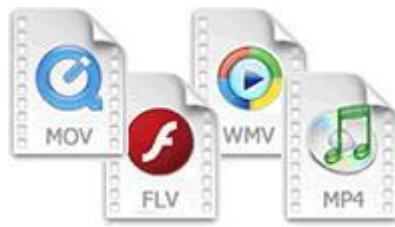
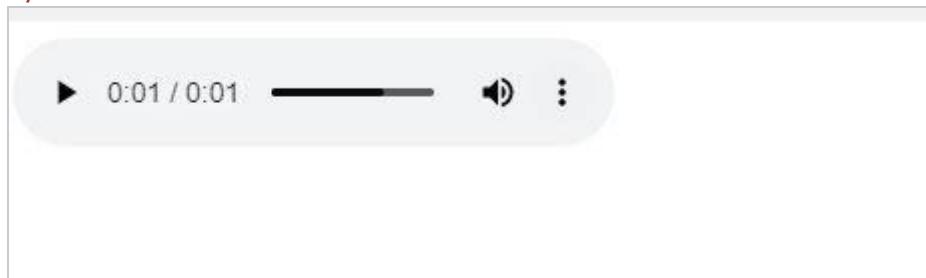


Image: HTML Video Format  
Reference: <https://www.sitesbay.com/html5/html5-video-tag>

The HTML5 `<audio>` element has specified a standard method to embed audio in a web page.

```
<audio controls>
  <source src="horse.ogg" type="audio/ogg">
  <source src="horse.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio>
```



The **controls** attribute is used to add audio controls, like play, pause, and volume. Text between the `<audio>` and `</audio>` tags will be displayed in browsers which does not support the `<audio>` element.

Multiple **<source>** elements can be linked to different audio files. The browser will be using the first recognized format.

The HTML5 `<video>` element is used to specify a standard way of embedding a video in a web page.

```
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
Your browser does not support the video tag.
</video>
```



The **controls** attribute is used to add video controls, like play, pause, and volume. It is a good method to always use **width** and **height** attributes. If height and width are not mentioned, the browser will not know the size of the video. The effect of this is that page will be changing (or flicker) while the video gets loaded.

Text between the `<video>` and `</video>` tags will only get displayed in browsers which do not support the `<video>` element. Multiple `<source>` elements can be used to link different video files. The browser will be using the first recognized format.

## Forms

HTML Forms are required to collect different kinds of user inputs, such as contact details like name, email address, phone numbers, or details like credit card information, etc.

Forms contain special elements called controls like inputbox, checkboxes, radio-buttons, submit buttons, etc. Users generally complete a form by modifying its controls e.g., entering text, selecting items, etc. and submitting this form to a web server for further processing.

### FORM ATTRIBUTES

#### Action Attribute

The action attribute defines the action to be performed when the form is submitted. Usually, the form data is sent to a page on the server when the user clicks on the submit button.

In the example above, the form data is sent to a page on the server called "/action\_page.php". This page contains a server-side script that handles the form data:

```
<form action="/action_page.php">
```

If the `action` attribute is omitted, the action is set to the current page.

#### Target Attribute

The target attribute specifies if the submitted result will open in a new browser tab, a frame, or in the current window. The default value is "`_self`" which means the form will be submitted in the current window.

To make the form result open in a new browser tab, use the value "`_blank`". Here, the submitted result will open in a new browser tab:

```
<form action="/action_page.php" target="_blank">
```

### Method Attribute

The method attribute specifies the HTTP method (**GET** or **POST**) to be used when submitting the form data.

```
<form action="/action_page.php" method="get">  
<form action="/action_page.php" method="post">
```

### EXAMPLE:

```
<form action="/action_page.php" target="_blank" method="post">  
  <label for="fname">First name:</label><br>  
  <input type="text" id="fname" name="fname" value="John"><br>  
  <label for="lname">Last name:</label><br>  
  <input type="text" id="lname" name="lname" value="Doe"><br><br>  
  <input type="submit" value="Submit">  
</form>
```

### Name Attribute

Each input field must have a `name` attribute to be submitted.

If the `name` attribute is omitted, the data of that input field will not be sent at all.

```
<form> <label>Username: <input type="text"></label>  
<label>Password: <input type="password"></label>  
<input type="submit" value="Submit"> </form>
```

### Input Element

This is the most commonly used element within HTML forms. It allows you to specify various types of user input fields, depending on the `type` attribute. An input element can be of type *text field*, *password field*, *checkbox*, *radio button*, *submit button*, *reset button*, *file select box*, as well as several new input types introduced in HTML5.

`<input>` element can be displayed in several ways, depending on the `type` attribute. If the `type` attribute is omitted, the input field gets the default type: "text".

```
<form>  
  <label for="fname">Firstname:</label><br>  
  <input type="text" id="fname" name="fname"><br>  
  <label for="lname">Lastname:</label><br>  
  <input type="text" id="lname" name="lname">  
</form>
```

The `for` attribute of the `<label>` tag should be equal to the `id` attribute of the `<input>` element to bind them together.

## <label> Element

The `<label>` tag defines a label for many form elements.

The `<label>` element is useful for screen-reader users, because the screen-reader will read out load the label when the user is focused on the input element.

## Text Fields

Text fields are one-line areas that allow the user to input text.

```
<form> <label for="username">Username:</label>
<input type="text" name="username" id="username"> </form>
```

## Password Field

Password fields are similar to text fields. The only difference is; characters in a password field are masked, i.e., they are shown as asterisks or dots. This is to prevent someone else from reading the password on the screen.

```
<form> <label for="user-pwd">Password:</label>
<input type="password" name="user-password" id="user-pwd"> </form>
```

Username:

## <button> Element

The `<button>` element defines a clickable button:

```
<button type="button" onclick="alert('Hello World!')>Click Me!</button>
```

## Radio Buttons

Radio buttons are used to let the user select exactly one option from a pre-defined set of options.

```
<form> <input type="radio" name="gender" id="male">
<label for="male">Male</label>
<input type="radio" name="gender" id="female"> <label for="female">Female</label>
</form>
```

Male  Female

```
<form>
<input type="radio" id="male" name="gender" value="male">
<label for="male">Male</label><br>
<input type="radio" id="female" name="gender" value="female">
<label for="female">Female</label><br>
<input type="radio" id="other" name="gender" value="other">
<label for="other">Other</label>
</form>
```

## Checkboxes

Checkboxes allows the user to select one or more option from a pre-defined set of options. It is created using an `<input>` element whose `type` attribute has a value of `checkbox`.

```
<form> <input type="checkbox" name="sports" id="soccer"> <label for="soccer">Soccer</label> <input type="checkbox" name="sports" id="cricket"> <label for="cricket">Cricket</label> <input type="checkbox" name="sports" id="baseball"> <label for="baseball">Baseball</label> </form>
```

Soccer  Cricket  Baseball

```
<form> <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike"> <label for="vehicle1"> I have a bike </label><br> <input type="checkbox" id="vehicle2" name="vehicle2" value="Car"> <label for="vehicle2"> I have a car </label><br> <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat"> <label for="vehicle3"> I have a boat </label> </form>
```

## File Select box

The file fields allow a user to browse for a local file and send it as an attachment with the form data.

```
<form> <label for="file-select">Upload:</label> <input type="file" name="upload" id="file-select"> </form>
```

Upload:  No file chosen

## Textarea

Textarea is a multiple-line text input control that allows a user to enter more than one line of text. Multi-line text input controls are created using an `<textarea>` element.

```
<form> <label for="address">Address:</label> <textarea rows="3" cols="30" name="address" id="address"></textarea> </form>
```

Address:

The `rows` attribute specifies the visible number of lines in a text area.  
The `cols` attribute specifies the visible width of a text area.

## Select Boxes

A select box is a dropdown list of options that allows user to select one or more option from a pull-down list of options. Select box is created using the `<select>` element and `<option>` element.

### `<select>` element defines a drop-down list

To define a pre-selected option, add the `selected` attribute to the option

```
<form> <label for="city">City:</label>
<select name="city" id="city">
<option value="sydney">Sydney</option>
<option value="melbourne">Melbourne</option>
<option value="cromwell">Cromwell</option>
</select> </form>
City:  ▾
```

<option value="fiat" selected>Fiat</option>

### Visible Values:

Use the `size` attribute to specify the number of visible values:

```
<select name="cars" size="3">
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="fiat">Fiat</option>
<option value="audi">Audi</option>
</select>
```

`multiple` attribute to allow the user to select more than one value:

```
<select name="cars" size="4" multiple>
```

### Submit and Reset Buttons

#### Submit Button

`<input type="submit">` defines a button for **submitting** the form data to a form-handler.

The form-handler is typically a page on the server with a script for processing input data.

The form-handler is specified in the form's **action** attribute.

```
<form action="/action_page.php">
<label for="fname">First name:</label><br>
<input type="text" id="fname" name="fname" value="John"><br>
<label for="lname">Last name:</label><br>
<input type="text" id="lname" name="lname" value="Doe"><br><br>
```

```
<input type="submit" value="Submit">  
</form>
```

A submit button is used to send the form data to a web server. When submit button is clicked the form data is sent to the file specified in the form's action attribute to process the submitted data.

A reset button resets all the forms control to default values.

```
<form action="action.php" method="post">  
<label for="first-name">First Name:</label>  
<input type="text" name="first-name" id="first-name">  
<input type="submit" value="Submit">  
<input type="reset" value="Reset"> </form>
```

First Name:

## Grouping Form Controls (**<fieldset>** and **<legend>**)

The **<fieldset>** element is used to group related data in a form.

The **<legend>** element defines a caption for the **<fieldset>** element.

You also group logically related controls and labels within a web form using the **<legend>** element. Grouping form controls into categories makes it easier for users to locate a control which makes the form more user-friendly.

```
<form>  
<fieldset>  
<legend>Contact Details</legend>  
<label>Email Address: <input type="email" name="email"></label>  
<label>Phone Number: <input type="text" name="phone"></label>  
</fieldset> </form>
```

## **<datalist>** Element

The **<datalist>** element specifies a list of pre-defined options for an **<input>** element.

Users will see a drop-down list of the pre-defined options as they input data.

The **list** attribute of the **<input>** element, must refer to the **id** attribute of the **<datalist>** element.

```
<form action="/action_page.php">  
<input list="browsers">  
<datalist id="browsers">  
  <option value="Internet  
                                Explorer">  
  <option value="Firefox">  
  <option value="Opera">
```

```
<option value="Safari">
</datalist>
</form>
```

## <output> Element

The `<output>` element represents the result of a calculation (like one performed by a script).

```
<form action="/action_page.php"
      oninput="x.value=parseInt(a.value)+parseInt(b.value)">
      0
      <input type="range" id="a" name="a" value="50">
      100
      +
      <input type="number" id="b" name="b" value="50">
      =
      <output name="x" for="a
      <br><br>
      <input type="submit">
      </form>
```

## Formatting

HTML also defines special **elements** for defining text with a special **meaning**.

The formatting tags are divided into two categories:

**physical tags**, used for text styling (visual appearance of the text)

**logical or semantic tags** used to add semantic value to the parts of the text (for example, let the search engines know what keywords should be web page ranked for).

HTML uses elements like `<b>` and `<i>` for formatting output, like **bold** or *italic* text. Formatting elements were designed to display special types of text:

- `<b>` - Bold text
- `<strong>` - Important text
- `<i>` - Italic text
- `<em>` - Emphasized text
- `<mark>` - Marked text
- `<small>` - Small text
- `<del>` - Deleted text
- `<ins>` - Inserted text
- `<sub>` - Subscript text
- `<sup>` - Superscript text

### <b> and <strong> Tags

The `<b>` tag is a physical tag that stands for bold text, whereas the `<strong>` tag being a logical tag is used to emphasize the importance of the text.

```
<b>This text is bold</b>
<strong>This text is strong</strong>
```

## <i> and <em> Tags

The `<i>` and `<em>` tags define italic text. The `<i>` tag is only responsible for visual appearance of the enclosed text, without any extra importance. The `<em>` tag defines emphasized text, with added semantic importance.

`<i>This text is italic</i>`

`<em>This text is emphasized</em>`

## <pre> Tag

The `<pre>` tag is used to define preformatted text. The browsers render the enclosed text with white spaces and line breaks.

`<pre>Spaces`

and line breaks

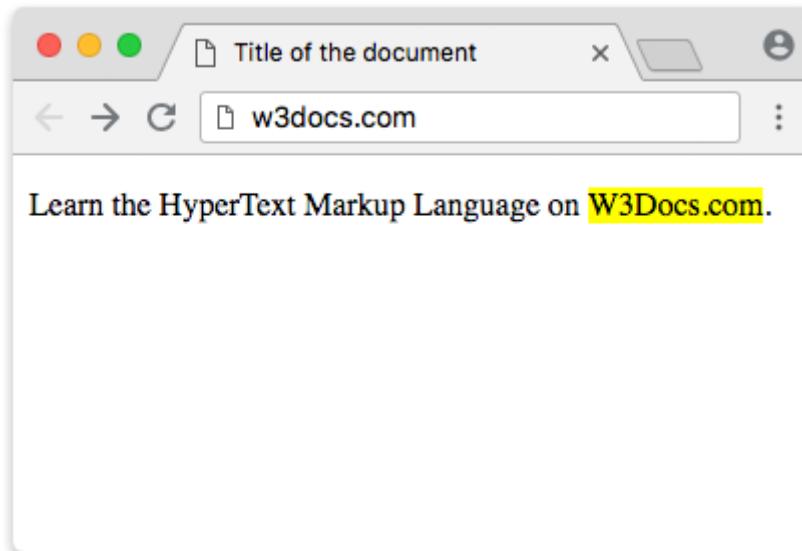
within this element

are shown as typed.

`</pre>`

## <mark> Tag

The `<mark>` tag is used to present a part of text in one document as marked or highlighted for reference purposes.



## <small> Tag

The `<small>` tag decreases the text font size by one size smaller than a document's base font size (from medium to small, or from x-large to large).

The tag usually contains the items of secondary importance such as copyright notices, side comments, or legal notices.

`<p>The interest rate is only 10%*</p>`  
`<small>* per day</small>`

## <big> Tag

The <big> tag defines bigger text.

<p><big>Bigger text</big></p>

## <del> and <s> Tags

The <del> tag specifies a part of the text that was deleted from the document. Browsers display this text as a strikethrough.

<p> She likes <del>violets</del> snowdrops.</p>

<p><s>I am studying in high school.</s></p>

The <s> tag defines text that is no longer correct, or relevant.

The content of both tags is displayed as strikethrough. However, despite the visual similarity, these tags cannot replace each other.

## <ins> and <u> Tags

The <ins> tag defines the text that has been inserted (added) to the document. The content of the tag is displayed as underlined.

<p>She likes <del>violets</del> <ins>snowdrops</ins>.</p>

The <u> tag specifies text that is stylistically different from normal text, i.e., words or fragments of text that need to be presented differently. This could be misspelled words or proper nouns in Chinese.

<p>Here we used <u>the &lt;u&gt; tag</u>.</p>

## <sub> and <sup>

The <sub> defines subscript texts. Subscript text is under the baseline of other symbols of the line and has smaller font. The <sup> tag defines superscript, that is set slightly above the normal line of type and is relatively smaller than the rest of the text. The baseline passes through upper or lower edge of the symbols.

<p>The formula of water is H<sub>2</sub>O, and the formula of alcohol is C<sub>2</sub>H<sub>5</sub>OH </p>

<p>E = mc<sup>2</sup>, where E — kinetic energy, m — mass, c — the speed of light. </p>

## <dfn> Tag

The <dfn> tag is used to define the term, that is mentioned for the first time. The content of the tag is displayed in italic.

<p><dfn>HTML</dfn> (HyperText Markup Language) — The standardized markup language for documents on the World Wide Web. Most web pages contain a description of the markup in the language HTML</p>

## <p>, <br> and <hr> Tags

The <p> tag defines the paragraph. Browsers automatically add 1em margin before and after each paragraph.

```
<p>The first paragraph</p>
```

The <br> tag inserts a single line break. Unlike the <p> tag an empty indent is not added before the line.

```
<h1>How to use the &lt;br&gt; tag</h1>
```

```
<p> We can insert the &lt;br /&gt; tag inside the paragraph, <br /> to transfer a part of the text to another line if necessary.</p>
```

In HTML5 the <hr> tag defines a thematic change between paragraph level elements in an HTML page. In previous versions of HTML it was used to draw a horizontal line on the page visually separating the content. In HTML5 the element has semantic meaning.

```
<h1>Football</h1>
```

```
<p>Team sports that involve, to varying degrees, kicking a ball with a foot to score a goal.</p>
```

```
<hr>
```

```
<h1>Basketball</h1>
```

```
<p>A game played between two teams of five players in which goals are scored by throwing a ball through a netted hoop fixed at each end of the court.</p>
```

## Block components

HTML Blocks are grouping items of HTML elements. Each HTML element has its own display based on the its type. However, HTML blocks also have display but can group the other HTML elements also as block.

By default, all block level elements are starts in new line. The elements added after block level elements will start from new line. Block level elements can contain other block line elements as nested block line elements. Block level elements can stretch to the browser full width.

Examples of block-level elements -

Tags	Description
<address>	Specifies the contact information for a page or document
<form>	A Facility to submit the input data by user
<blockquote>	used to define a quote section
<p>	Paragraph
<pre>	Preformatted text in HTML document
<h1> - <h6>	Used to define HTML headings
<dd>	Used to specify the term/description/name in a description list
<div>	used to create block-level elements
<table>	represents a table in an HTML document

<ol>	Ordered list in html documents
<ul>	Unordered list in HTML document
<nav>	Specifies the navigation area

Block elements can be divided into two types.

1. Inline Elements
2. Group Elements

## Inline elements

The inline elements do not start at newline. The inline elements always start in the middle of the line and part of another element. The below list are the inline elements.

Tags	Description
<b>	used to display the text as bold
<i>	Alternate Voice or different quality of text
<u>	Unarticulated text/underlined text in HTML document
<p>	Paragraph
<address>	Specifies the contact information for a page or document
<form>	A Facility to submit the input data by user
<li>	used to represent a list item in HTML document
<ins>	Editorial Insertion
<del>	Used for editorial deletion
<code>	Used to define the computer code

## Group elements

The Group elements always start with newline. The Group elements used to group other elements. <div> and <span> are the majorly used Group elements.

### <div>

<div> is mostly used to create block-level elements. In other words, <div> tag used to define a section in HTML documents. Except <div>, all other have its own display.

The tag can be coded like <div></div> with its contents/other HTML tags inserted between the opening and closing tags. <div> also acts as block-level element that is used as container for other HTML elements. The <div> element has no additional attributes by default.

style and class are commonly provided with DIV element. When <div> element used together with CSS, the style blocks can be used together. All browsers always place a line break before and after the <div> tag by default.

Syntax -

```
<div>.....</div>
```

Example -

```
<!DOCTYPE html>
<html>
    <head>
        <title>div tag text example.. </title>
    </head>
    <body>
```

```
<div style="background-color:green;">
    <h2>Heading</h2>
    <p>This is paragraph</p>
</div>
</body>
</html>
```

### Output -

#### Heading

This is paragraph

#### <span>

Generic container for text or group inline-elements HTML. The `<span>` tag used to specify the generic container for inline elements and content. There no visual change that can be done by `<span>` tag.

It doesn't represent anything other than its children. The tag can be specified like `<span class=""></span>` with the content between the opening and closing tags.

#### Syntax -

```
<span>.. text here.. </span>
```

#### Example -

```
<!DOCTYPE html>
<html>
<head>
    <title>span tag example.. </title>
</head>
<body>
    <p> <span style="color:red;font-weight:bold">Tutorials</span>
        objective is to deliver the point to point online content on various
        technologies(including technical and non-technical)to encourage the
        reader to learn and gain expertise on their desired skills without
        any conditions and restrictions. </p>
</body>
</html>
```

### Output

**Tutorials** objective is to deliver the point to point online content on various technologies (including technical and non-technical) to encourage the reader to learn and gain expertise on their desired skills without any conditions and restrictions.

## Practical Activity

### Lists

#### Program:

```
<!DOCTYPE html>
<html>
<body>
```

```
<h2>A Nested List</h2>
```

```
<ul>
<li>Coffee</li>
<li>Tea
    <ul>
        <li>Black tea</li>
        <li>Green tea</li>
    </ul>
</li>
<li>Milk</li>
</ul>

</body>
</html>
```

**Output/Results snippet:**

## A Nested List

- Coffee
- Tea
  - Black tea
  - Green tea
- Milk

**Tables**  
**Program:**

```
<Html>
<head>
    <title>Sample</title>
</head>
<body bgcolor="#ff88ff">
<table border="4">
    <thead>
        <tr>
            <th>Month</th>
            <th>Savings</th>
        </tr>
    </thead>
    <tbody>
        <tr style="color:red;">
            <td>January</td>
            <td>$100</td>
        </tr>
```

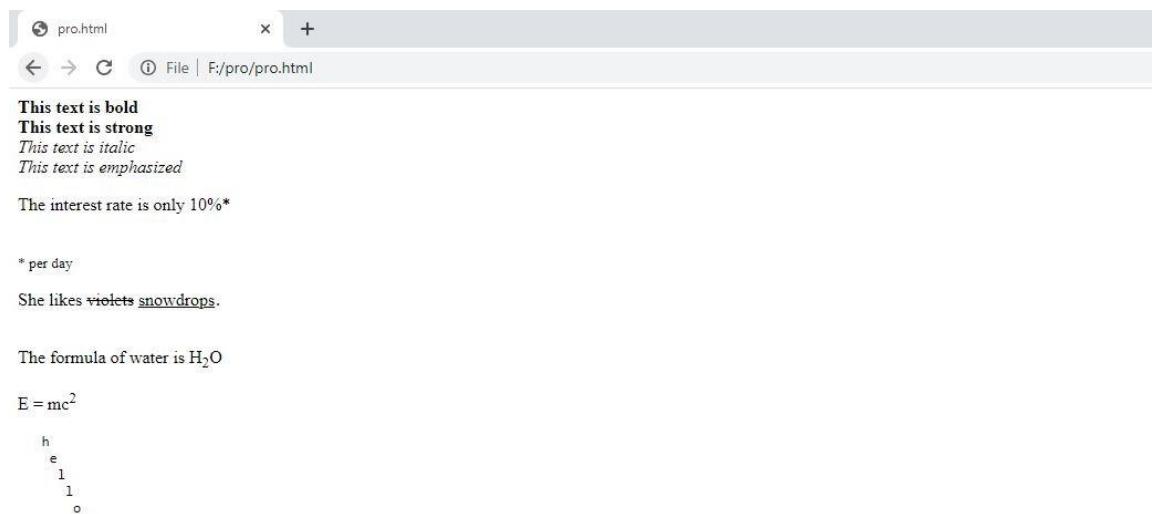
```
<tr>
  <td>February</td>
  <td>$80</td>
</tr>
</tbody>
<tfoot>
<tr>
  <td>Sum</td>
  <td>$180</td>
</tr>
</tfoot>
</table>
</body>
</Html>
```

**Output/Results snippet:****Formatting  
Program:**

```
<!DOCTYPE html>
<html>
<body>
<b>This text is bold</b> <br>
<strong>This text is strong</strong><br>
<i>This text is italic</i><br>
<em>This text is emphasized</em><br>
<p>The interest rate is only 10%*</p><br>
  <small>* per day</small><br>
<p>She likes <del>violets</del> <ins>snowdrops</ins>. </p><br>
The formula of water is H<sub>2</sub>O<br>
<p>E = mc<sup>2</sup></p>
<pre>
h
e
l
```

```
|  
o  
  
</pre>  
</body>  
</html>
```

### Output/Results snippet:



### Block-Components Program:

```
<!DOCTYPE html>  
<html>  
<head>  
<style>  
.nav {  
background-color: yellow;  
list-style-type: none;  
text-align: center;  
margin: 0;  
padding: 0;  
}  
  
.nav li {  
display: inline-block;  
font-size: 20px;  
padding: 20px;  
}  
</style>  
</head>  
<body>
```

## <h1>Horizontal Navigation Links</h1>

<p>By default, list items are displayed vertically. In this example we use display: inline-block to display them horizontally (side by side).</p>

<p>Note: If you resize the browser window, the links will automatically break when it becomes too crowded.</p>

```
<ul class="nav">
<li><a href="#home">Home</a></li>
<li><a href="#about">About Us</a></li>
<li><a href="#clients">Our Clients</a></li>
<li><a href="#contact">Contact Us</a></li>
</ul>

</body>
</html>
```

### Output/Results

snippet:

## Horizontal Navigation Links

By default, list items are displayed vertically. In this example we use display: inline-block to display them horizontally (side by side).

Note: If you resize the browser window, the links will automatically break when it becomes too crowded.



Home    About Us    Our Clients    Contact Us

## Forms

### Program:

```
<!DOCTYPE html>
<html>
<head>
<style>
input[type=text] {
  width: 100%;
  padding: 12px 20px;
  margin: 8px 0;
  box-sizing: border-box;
}
</style>
</head>
<body>
```

<p>Padded text fields:</p>

```
<form>
  <label for="fname">First Name</label>
  <input type="text" id="fname" name="fname">
  <label for="lname">Last Name</label>
  <input type="text" id="lname" name="lname">
<input type="button" value="Click">
</form>
</body>
</html>
```

**Output/Results snippet:**

First Name

Last Name

Click

## 3.3 CSS for Page Design

### CSS design principles

**Forward and backward compatibility:** CSS 2.2 user agents will be able to understand CSS1 style sheets. CSS1 user agents will be able to read CSS 2.2 style sheets and discard parts they do not understand. Also, user agents with no CSS support will be able to display style-enhanced documents. Of course, the stylistic enhancements made possible by CSS will not be rendered, but all content will be presented.

**Complementary to structured documents:** Style sheets complement structured documents (e.g., HTML and XML applications), providing stylistic information for the marked-up text. It should be easy to change the style sheet with little or no impact on the markup.

**Vendor, platform, and device independence:** Style sheets enable documents to remain vendor, platform, and device independent. Style sheets themselves are also vendor and platform independent, but CSS 2.2 allows you to target a style sheet for a group of devices (e.g., printers).

**Maintainability:** By pointing to style sheets from documents, webmasters can simplify site maintenance and retain consistent look and feel throughout the site. For example, if the organization's background color changes, only one file needs to be changed.

**Simplicity:** CSS is a simple style language which is human readable and writable. The CSS properties are kept independent of each other to the largest extent possible and there is generally only one way to achieve a certain effect.

**Network performance:** CSS provides for compact encodings of how to present content. Compared to images or audio files, which are often used by authors to achieve certain rendering effects, style sheets most often decrease the content size. Also, fewer network connections have to be opened which further increases network performance.

**Flexibility:** CSS can be applied to content in several ways. The key feature is the ability to cascade style information specified in the default (user agent) style sheet, user style sheets, linked style sheets, the document head, and in attributes for the elements forming the document body.

**Richness:** Providing authors with a rich set of rendering effects increases the richness of the Web as a medium of expression. Designers have been longing for functionality commonly found in desktop publishing and slide-show applications. Some of the requested rendering effects conflict with device independence, but CSS 2.2 goes a long way toward granting designers their requests.

**Alternative language bindings:** The set of CSS properties described in this specification form a consistent formatting model for visual and aural presentations. This formatting model can be accessed through the CSS language, but bindings to other languages are also possible. For example, a JavaScript program may dynamically change the value of a certain element's 'color' property.

**Accessibility:** Several CSS features will make the Web more accessible to users with disabilities. Properties to control font appearance allow authors to eliminate inaccessible bit-mapped text images. Positioning properties allow authors to eliminate mark-up tricks (e.g., invisible images) to force layout.

The semantics of important rules mean that users with particular presentation requirements can override the author's style sheets. The 'inherit' value for all properties improves cascading generality and allows for easier and more consistent style tuning.

## property:values

### Properties

CSS properties are the styles used on specified selectors. They are written before values in the CSS ruleset and are separated from property values by a colon. Different HTML selectors and elements have different properties. Some properties are universal and can be used on every selector. Others work only on specific selectors and under particular conditions. An example of that is grid-template-columns, which is used to style the layout of a page. It works mainly with *divs* with their *display* property set to the grid (display: grid).

There are many properties and their values for HTML selectors. According to CSS

Tricks, there are "520 distinct property names from 66 technical reports and 66 editor's drafts".

Here are four common properties to work with

- List properties
- Font properties
- Border properties
- Text properties

These properties are common because they are frequently used in all CSS documents and can be applied to different selectors. One unique thing about properties is that they have more than one value attached to them. Text-transform, for example, a property that controls a text's capitalization, can be set to uppercase, lowercase, capitalize, or none. But this also poses a restraint. You must specify a value to the right property else nothing happens. If we have 'text-transform: underline;' nothing will change on the text part because underline is a value for text-decoration.

Below are some properties, their descriptions, and the values they accommodate.

## TEXT

Properties	Description	Values
color	Sets the color of a text	Hex, RGB, keyword
text-transform	Sets the capitalization of the text	uppercase, lowercase, capitalize, none
text-align	Sets the alignment of the text on	right, left, center, justify
	the screen	
letter-spacing	Sets the spacing between text	normal, length
	characters	
text-decoration	Sets the decoration added to the text	none, underline, line-through, overline

## BORDER

Properties	Description	Values
border	Sets the shorthand combination for border-width, border-style, border-color	border-width, border-style, border-color
border-color	Sets the color for the border	Keyword, RGB, transparent, inherit
border-radius	Sets the radius of the four corners of an element's border	length, percentage, initial, inherit
border-style	Sets the style for an element's border	none, hidden, dotted, solid, dashed..., i
border-image	Sets an image as an element's border	border-image-source, border-image-width...

## FONT

Properties	Description	Values
font	Sets the shorthand for all the font specifications	font-style, font-variant, font-weight, font-size/line-height

font-weight	Sets the weight of a font	<i>normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900, inherit</i>
font-style	Sets the style of a font	<i>Normal, italic, oblique, initial, inherit</i>

## LIST

Properties	Description	Values
list-style	Shorthand combination for list-style-type, list-style-position, and list-style-image	<i>list-style-type, list-style-position, list-style-image, inherit</i>
list-image	Sets an image as the <i>none, url, initial, inherit</i> list-item marker	
list-type	This sets the type of <i>none, disc, circle, square, decimal, decimal-leading-zero.....</i> list-item marker	

### Values

Values are written immediately after the colon that separates them from CSS properties. CSS values aren't just text; they come in different forms - URLs, units, measurements, integers, strings, inherit, auto, none, etc. We will look at different CSS values and how to implement them.

## TEXT

Text values are prevalent in CSS. They are usually written without quotes, unlike strings. Every valid CSS property has a text as a value. Margin takes units as values, but it also has auto, which in this case, is taken as text.

```
h1 {  
    color: red;  
    text-align: center;  
}
```

Red and center are text values of color and text-align.

## INTEGERS

Integers are numbers from one to zero or zero to nine. Some CSS properties make allowance for their values to be defined as integers, even negative integers. The values for properties like column-count and z-index are defined using integers.

```
div {  
    column-count: 4;  
}  
div {  
    z-index: -1;  
}
```

## UNITS/MEASUREMENTS

Because CSS must be used to position items on a web page, general layout, and media queries, many properties take units and measurements as their value. There are many property-specific units for CSS values, but available units like px, em, fr and percentages are standard.

```
h3 {  
    font-size: 24px;  
}  
Div {  
    grid-template-columns: 1fr 1fr 1fr 1fr;  
}  
div {  
    width: 100%;  
    font-size: 2em; }
```

## URLS

Properties like background-image take an actual URL as their value. This URL could be absolute or relative. Meaning, you could copy the address of an image online and use it as your background image, or you could get images from the project directory/folder you are working with on your computer.

```
div{  
background-  
image:url("https://cdn.pixabay.com/photo/2015/04/23/22/00/tree736885__340.jpg");  
}  
div{  
    background-image: url("./Images/landscape.jpg");  
}
```

## STRINGS

Strings are text within quotes. The content property which you can use to insert generated content allows strings within its parameter.

```
.div :: after {  
    content: "This is me";  
}
```

## COLORS

The color/background-colors are properties used to set the color of HTML elements. Their values accept either a hexadecimal color combination value, an RGB specification, or a defined color.

```
h2 {  
    color: red;  
}  
div{  
    background-color: #000000;  
}
```

## AUTO, NONE, INHERIT

*Auto*, *None*, and *Inherit* are keywords values in CSS. Auto value allows the property to fill or adjust according to the content of the element.

```
div {  
    width: auto;  
    margin: auto;  
}
```

None adds no value to the specified property.

```
div{  
    border: none;  
}
```

Inherit keyword allows the property to inherit the same value from the parent element. All CSS properties accept *inherit*.

```
<div>  
    <p> this is me </p>  
    <p id="pickme"> this is not me </p>  
</div>  
p{  
    color: red;  
}  
#pickme p {  
    color: inherit;  
}
```

Although all the examples above are of properties with one value, CSS values can have more than one value, and you can manipulate it to write shorthand. Shorthand in CSS is a shorter, more compact way of setting values for similar but multiple properties.

These values,

```
div{  
    border-style: solid;  
    border-color: red;  
    border-width: 5px;  
}
```

can be rewritten in shorthand as

```
div{  
    border: 5px solid red;  
}
```

Other properties with multiple values include, but not limited to  
margin: 1px 3px 6px 2px; /\* defining the top, left, bottom and left of a margin \*/  
font: italic small-caps bold 12px Georgia, serif; /\*defining font-style, font-variant-caps, font-weight, font-size, and font-family. \*/

```
box-shadow: 5px 2px 8px #aaaaaa; /* defining the horizontal offset, vertical offset, blur, and color values */
```

## Dynamic CSS3

### How to create Dynamic CSS?

While writing CSS sometimes we need to declare variables, CSS custom properties are like CSS's own implementation of variables.

### What are the CSS custom properties?

CSS custom properties allow you to assign arbitrary values to a property with a name of your choice. It allows you to use the **var()** function to use these values in other properties.

### Prerequisites

- Basic knowledge of CSS
- You should have knowledge of **currentColor** refers to the current color value of an element

### Implement Custom CSS Properties

How to create a pre/postprocessor variables

How to use Variable

### How to Create pre/post processor Variables

- CSS pre/postprocessors are allowed to store values in a keyword.
- It also scopes the keyword to a certain selector if required.
- This enables you to store color codes, sizes with all of the known units.
- **var()** function has been used to get the value of the custom property.
- We cannot use **var()** functions for selectors, property names, or media query declarations.
- CSS pre/postprocessor variables are only used at compilation-time.
- CSS variables can be used and updated dynamically.

The syntax for CSS custom properties is a bit weird compared to other languages, for example:

```
:root {  
    currentColor: #D3D3D3;  
}
```

```
.section {  
    background: var(currentColor);  
}
```

CSS variables can be used and updated dynamically, illustrate with below example

```
:root {  
    $value: 10px;  
}
```

```
.radius {  
    border-radius: $value;  
}  
  
@media screen and (min-width: 600px) {  
    $value: 50px;  
}
```

This snippet of SASS declarations and rules compiles to CSS as follows:

```
.radius {  
    border-radius: 10px;  
}
```

### ***How to use Variable***

Some Basic CSS custom variables we are using in CSS since a long time is **currentColor** keyword. It can be used on any declaration that accepts a color value, and it cascades perfectly.

Example

```
.section {  
    color: red;  
    border: 1px solid currentColor;  
}  
.section .item {  
    background: currentColor;  
}  
  
.section .item.blue {  
    color: blue;  
}
```

### **Explanation of above Example:**

Create element with color: red

Sets a red background color for every span child of .element, unless a color property is declared in this same block

### ***Transform***

The **transform** property applies a 2D or 3D transformation to an element. This property allows you to rotate, scale, move, skew, etc., elements.

```
div.a {  
    transform: rotate(20deg);  
}  
  
div.b {  
    transform: skewY(20deg);  
}  
  
div.c {
```

```
        transform: scaleY(1.5);  
    }  
  
<style>  
div.a {  
    width: 150px;  
    height: 80px;  
    background-color: yellow;  
    -ms-transform: rotate(20deg); /* IE 9 */  
    transform: rotate(20deg);  
}  
  
div.b {  
    width: 150px;  
    height: 80px;  
    background-color: yellow;  
    -ms-transform: skewY(20deg); /* IE 9 */  
    transform: skewY(20deg);  
}  
  
div.c {  
    width: 150px;  
    height: 80px;  
    background-color: yellow;  
    -ms-transform: scaleY(1.5); /* IE 9 */  
    transform: scaleY(1.5);  
}  
</style>
```

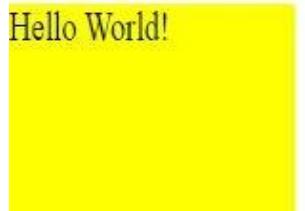
**transform: rotate(20deg);**



**transform: skewY(20deg);**



**transform: scaleY(1.5);**



## ***Transitions***

CSS transitions allows you to change property values smoothly, over a given duration.

### **How to Use CSS Transitions?**

To create a transition effect, you must specify two things:

- the CSS property you want to add an effect to
- the duration of the effect

**Note:** If the duration part is not specified, the transition will have no effect, because the default value is 0.

```
div {  
    width: 100px;  
    height: 100px;  
    background: red;  
    transition: width 2s;  
}
```

### **Change Several Property Values**

The following example adds a transition effect for both the width and height property, with a duration of 2 seconds for the width and 4 seconds for the height:

```
div {  
    transition: width 2s, height 4s;  
}
```

## Animations

CSS allows animation of HTML elements without using JavaScript or Flash!

### What are CSS Animations?

An animation lets an element gradually change from one style to another. You can change as many CSS properties you want, as many times you want. To use CSS animation, you must first specify some keyframes for the animation. Keyframes hold what styles the element will have at certain times.

#### The @keyframes Rule

When you specify CSS styles inside the `@keyframes` rule, the animation will gradually change from the current style to the new style at certain times.

To get an animation to work, you must bind the animation to an element.

The following example binds the "example" animation to the `<div>` element. The animation will last for 4 seconds, and it will gradually change the background-color of the `<div>` element from "red" to "yellow":

```
/*           The           animation           code           */
/*@keyframes                                         example {*/
  from {background-color: red;}                      to {background-color: yellow;}
}

/*     The     element     to     apply     the     animation     to     */
div {                                              width: 100px;
                                               height: 100px;
                                               background-color: red;
                                               animation-name: example;
                                               animation-duration: 4s;
}
```

#### Delay an Animation

The `animation-delay` property specifies a delay for the start of an animation.

The following example has a 2 seconds delay before starting the animation:

```
div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
  animation-delay: 2s;
}
```

## CSS Box Model

In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:

Explanation of the different parts:

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

The box model allows us to add a border around elements, and to define space between elements.

```
div {  
    width: 300px;  
    border: 15px solid green;  
    padding: 50px;  
    margin: 20px;  
}
```

### Width and Height of an Element

In order to set the width and height of an element correctly in all browsers, you need to know how the box model works.

**Important:** When you set the width and height properties of an element with CSS, you just set the width and height of the **content area**. To calculate the full size of an element, you must also add padding, borders and margins.

```
div {  
    width: 320px;  
    padding: 10px;  
    border: 5px solid gray;  
    margin: 0;  
}
```

Here is the calculation:

320px	(width)
+ 20px	padding)
+ 10px	border)
+ 0px	margin)
<b>= 350px</b>	

The total width of an element should be calculated like this:

Total element width = width + left padding + right padding + left border + right border + left margin + right margin

The total height of an element should be calculated like this:

Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

Web design refers to the design of websites that are displayed on the internet. It usually refers to the user experience aspects of website development rather than software development..

## Design layout controls

### *display*

The **display** property specifies the display behavior (the type of rendering box) of an element.

Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is **block** or **inline**.

#### Inline

The default value for elements. Think of elements like `<span>`, `<em>`, or `<b>` and how wrapping text in those elements within a string of text doesn't break the flow of the text.

Pellentesque **inline element** morbi tristique senectu  
Donec eu libero sit amet quam egestas semper. Ac

The `<em>` element has a 1px red border. Notice it sits right *inline* with the rest of the text.

An inline element will accept margin and padding, but the element still sits inline as you might expect. Margin and padding will only push other elements horizontally away, not vertically.

Pellentesque **inline element** morbi tristique senectus et netus et  
malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae,  
ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas  
semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

An inline element will not accept height and width. It will just ignore it.

## Inline Block

An element set to inline-block is very similar to inline in that it will sit inline with the natural flow of text (on the “baseline”). The difference is that you are able to set a width and height which will be respected.



senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

## Block

A number of elements are set to block by the browser UA stylesheet. They are usually container elements, like `<div>`, `<section>`, and `<ul>`. Also text “blocks” like `<p>` and `<h1>`. Block level elements do not sit inline but break past them. By default (without setting a width) they take up as much horizontal space as they can.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

*hi*

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

## Flexbox

The `display` property is also used for new fangled layout methods like Flexbox.

```
.header {  
  display: flex;  
}
```

## Grid

Grid layout will also be initially set by the `display` property.

```
body {  
  display: grid;  
}
```

## ***float Property***

The **float** property is used for positioning and formatting content e.g. let an image float left to the text in a container.

The **float** property can have one of the following values:

- **left** - The element floats to the left of its container
- **right** - The element floats to the right of its container
- **none** - The element does not float (will be displayed just where it occurs in the text). This is default
- **inherit** - The element inherits the float value of its parent

In its simplest use, the **float** property can be used to wrap text around images.

**Example - float: right;**



The following example specifies that an image should float to the **right** in a text:

Lore ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac...

```
img {  
    float: right;  
}
```

## **Positioning techniques**

Positioning allows you to move an element from where it would be placed when in normal flow to another location. Positioning isn't a method for creating your main page layouts, it is more about managing and fine-tuning the position of specific items on the page. There are however useful techniques for certain layout patterns that rely on the position property. Understanding positioning also helps in understanding normal flow, and what it is to move an item out of normal flow. There are five types of positioning you should know about:

- **Static positioning** is the default that every element gets — it just means "put the element into its normal position in the document layout flow — nothing special to see here".

- **Relative positioning** allows you to modify an element's position on the page, moving it relative to its position in normal flow — including making it overlap other elements on the page.
- **Absolute positioning** moves an element completely out of the page's normal layout flow, like it is sitting on its own separate layer. From there, you can fix it in a position relative to the edges of the page's `<html>` element (or its nearest positioned ancestor element). This is useful for creating complex layout effects such as tabbed boxes where different content panels sit on top of one another and are shown and hidden as desired, or information panels that sit off screen by default, but can be made to slide on screen using a control button.
- **Fixed positioning** is very similar to absolute positioning, except that it fixes an element relative to the browser viewport, not another element. This is useful for creating effects such as a persistent navigation menu that always stays in the same place on the screen as the rest of the content scrolls.
- **Sticky positioning** is a newer positioning method which makes an element act like `position: static` until it hits a defined offset from the viewport, at which point it acts like `position: fixed`.

## position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with `position: static;` is not positioned in any special way; it is always positioned according to the normal flow of the page:

```
div.static {  
  position: static;  
  border: 3px solid #73AD21;  
}
```

## position: static;

An element with `position: static;` is not positioned in any special way; it is always positioned according to the normal flow of the page:

This div element has `position: static;`

## position: relative;

An element with `position: relative;` is positioned relative to its normal position. Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

```
div.relative {  
    border: 3px solid #73AD21;  
}  
  
position: relative;
```

An element with **position: relative;** is positioned relative to its normal position:

This div element has position: relative;

## position: fixed;

An element with **position: fixed;** is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element. A fixed element does not leave a gap in the page where it would normally have been located.

```
div.fixed {  
    border: 3px solid #73AD21;  
}  
  
position: fixed;  
bottom: 0;  
right: 0;  
width: 300px;  
#73AD21;
```

This div element has position: fixed;

## position: absolute;

An element with **position: absolute;** is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

**Note:** A "positioned" element is one whose position is anything except **static**.

```
div.relative {  
    position: relative;  
}  
  
border: 3px solid #73AD21;  
  
width: 400px;  
height: 200px;  
#73AD21;
```

```
div.absolute {
```

```
position: absolute;  
top: 80px;  
right: 0;  
width: 200px;  
height: 100px;  
border: 3px solid #73AD21;  
}
```

This div element has position: relative;

This div element has position: absolute;

## position: sticky;

An element with **position: sticky;** is positioned based on the user's scroll position. A sticky element toggles between **relative** and **fixed**, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like **position:fixed**).

```
div.sticky {  
    position: -webkit-sticky; /* Safari */  
    position: sticky;  
    top: 0;  
    background-color: green;  
    border: 2px solid #4CAF50;  
}
```

I am sticky!

Scroll back up to remove the stickyness.

Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudianda nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.

Some text to enable scrolling.. Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudianda nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.

## Practical Activities

**Program:****Create Web-Layout Page Design**

```
<!DOCTYPE html>
<html>
<head>
<link href="style.css" rel="stylesheet">
</head>
<body>

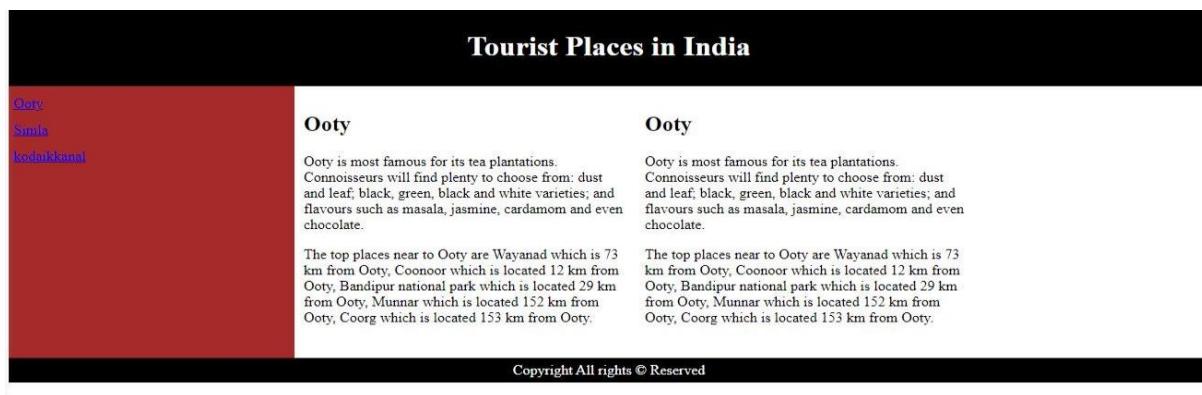
<header>
<h1>Tourist Places in India</h1>
</header>

<nav>
<a href="#">Ooty</a><br>
<a href="#">Simla</a><br>
<a href="#">kodaikanal</a><br>
</nav>
<main>
<section>
<h1>Ooty</h1>
<p>Ooty is most famous for its tea plantations. Connoisseurs will find plenty to choose from: dust and leaf; black, green, black and white varieties; and flavours such as masala, jasmine, cardamom and even chocolate.</p>
<p>The top places near to Ooty are Wayanad which is 73 km from Ooty, Coonoor which is located 12 km from Ooty, Bandipur national park which is located 29 km from Ooty, Munnar which is located 152 km from Ooty, Coorg which is located 153 km from Ooty.</p>
</section>
<section>
<h1>Ooty</h1>
<p>Ooty is most famous for its tea plantations. Connoisseurs will find plenty to choose from: dust and leaf; black, green, black and white varieties; and flavours such as masala, jasmine, cardamom and even chocolate.</p>
<p>The top places near to Ooty are Wayanad which is 73 km from Ooty, Coonoor which is located 12 km from Ooty, Bandipur national park which is located 29 km from Ooty, Munnar which is located 152 km from Ooty, Coorg which is located 153 km from Ooty.</p>
</section>
</main>
<footer>
Copyright All rights &#169; Reserved
</footer>
</body>
</html>
```

**CSS**

```
<style>
  body
  {
    font-size:20px;
  }
  header {
    background-color:black;
    color:white;
    text-align:center;
    padding:5px;
  }
  nav {
    line-height:30px;
    background-color:white;
    height:300px;
    width:300px;
    float:left;
    padding:5px;
  }
  section {
    width:350px;
    float:left;
    padding:10px;
  }
  footer {
    background-color:black;
    color:white;
    clear:both;
    text-align:center;
    padding:5px;
  }
</style>
```

## Output/Results snippet:



The screenshot shows a web page with a dark header bar containing the title "Tourist Places in India". Below the header, there is a sidebar on the left with links to "Ooty", "Munnar", and "Kodaikkalai". The main content area is divided into two columns. The left column is titled "Ooty" and contains text about Ooty's tea plantations and nearby places like Wayanad, Coonoor, Bandipur national park, Munnar, and Coorg. The right column is also titled "Ooty" and contains similar information. At the bottom of the page, there is a footer bar with the copyright notice "Copyright All rights © Reserved".

## Animation

### Program:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
    width: 100px;
    height: 100px;
    background-color: red;
    animation-name: example;
    animation-duration: 4s;
}

@keyframes example {
    from {background-color: red;}
    to {background-color: yellow;}
}
</style>
</head>
<body>

<div></div>

</body>
</html>
```

### Output/Results snippet:

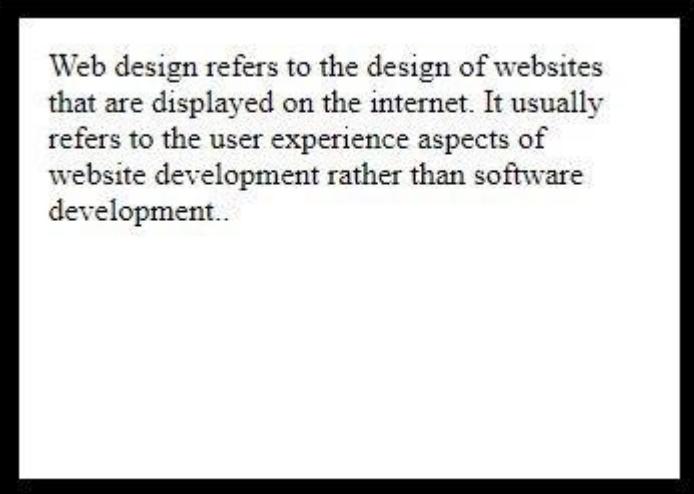


## Box Model

### Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>CSS Box Formatting Model</title>
<style>
div {
    width: 300px;
    height: 200px;
    padding: 15px;
}
```

```
border: 10px solid black;
margin: 20px auto;
}
</style>
</head>
<body>
  <div>Web design refers to the design of websites that are displayed on the internet. It usually refers to the user experience aspects of website development rather than software development.</div>
</body>
</html>
```

**Output/Results snippet:**

Web design refers to the design of websites that are displayed on the internet. It usually refers to the user experience aspects of website development rather than software development..

**Float  
Program:**

```
<!DOCTYPE html>
<html>
<head>
<style>
img {
  float: right;
}
</style>
</head>
<body>
```

<h2>Float Right</h2>

<p>In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.</p>

```
<p>
Web design refers to the design of websites that are displayed on the internet. It usually refers to the user experience aspects of website development rather than software development..Klientboost. This is another fantastic example of mobile responsive web design. Their website loads remarkably fast at four seconds using 3G connections. More importantly, the look and feel of Klientboost's website stays consistent across all devices, yet they've managed to tailor their user experience to each device.</p>

</body>
</html>
```

### Output/Results snippet:

#### Float Right

In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.

Web design refers to the design of websites that are displayed on the internet. It usually refers to the user experience aspects of website development rather than software development..Klientboost. This is another fantastic example of mobile responsive web design. Their website loads remarkably fast at four seconds using 3G connections. More importantly, the look and feel of Klientboost's website stays consistent across all devices, yet they've managed to tailor their user experience to each device.



## 3.4 JS for Client-side scripting

### Handling HTML Events

#### *What is an Event?*

JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page. When the page loads, it is called an event. When the user clicks a button, that click too is an event. Other examples include events like pressing any key, closing a window, resizing a window, etc. Developers can use these events to execute JavaScript coded responses, which cause buttons to close windows, messages to be displayed to users, data to be validated, and virtually any other type of response imaginable.

Events are a part of the Document Object Model (DOM) Level 3 and every HTML element contains a set of events which can trigger JavaScript Code.

### <body> and <frameset> Level Events

There are only two attributes which can be used to trigger any javascript or vbscript code when there is any event occurs on document level.

Attribute	Value	Description
onload	script	Script runs when a HTML document loads
onunload	script	Script runs when a HTML document unloads

**NOTE** – Here script refer to any VBScript or JavaScript function or piece of code.

### <form> Level Events

There are following six attributes which can be used to trigger any javascript or vbscript code when there is any event occurs on form level.

Attribute	Value	Description
onchange	script	Script runs when the element changes
onsubmit	script	Script runs when the form is submitted
onreset	script	Script runs when the form is reset
onselect	script	Script runs when the element is selected
onblur	script	Script runs when the element loses focus
onfocus	script	Script runs when the element gets focus

### Keyboard Events

There are following three events which are generated by keyboard. These events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, and title elements.

Attribute	Value	Description
onkeydown	script	Script runs when key is pressed
onkeypress	script	Script runs when key is pressed and released
onkeyup	script	Script runs when key is released

### Other Events

There following other 7 events which are generated by mouse when it comes in contact of any HTML tag. These events are not valid in base, bdo, br, frame, frameset, head, html, iframe, meta, param, script, style, title elements.

Attribute	Value	Description
onclick	script	Script runs when a mouse click
ondblclick	script	Script runs when a mouse double-click
onmousedown	script	Script runs when mouse button is pressed

onmousemove	script	Script runs when mouse pointer moves
onmouseout	script	Script runs when mouse pointer moves out of an element
onmouseover	script	Script runs when mouse pointer moves over an element
onmouseup	script	Script runs when mouse button is released

## onclick Event Type

```
<script>
    function showAlert() {
        alert('Clicked!');
    }
</script>
<input type="button" value="Save" onclick="showAlert()">
```

## Animations

JavaScript animations are done by programming gradual changes in an element's style. The changes are called by a timer. When the timer interval is small, the animation looks continuous.

```
<!DOCTYPE html>
<html>
<body>

<h1>My First JavaScript Animation</h1>

<div id = "myContainer">
    <div id = "myAnimation">My animation will go here</div>
</div>

</body>
<html>
```

## Styling the Elements

To make an animation possible, the animated element must be animated relative to a "parent container". The container element should be created with style = "position: relative". The animation element should be created with style = "position: absolute".

```
#myContainer {
    width: 400px;
    height: 400px;
    position: relative;
    background: yellow;
}
```

```
#myAnimation {  
    width: 50px;  
    height: 50px;  
    position: absolute;  
    background: red;  
}
```

## Create the Animation Using JavaScript

```
var id = setInterval(frame, 5);  
  
function frame()  
    if /* test for finished */ {  
        clearInterval(id);  
    } else {  
        /* code to change the element style */  
    }  
}  
  
var id = null;  
function myMove()  
    var elem = document.getElementById("myAnimation");  
    var pos = 0;  
    clearInterval(id);  
    id = setInterval(frame, 10);  
    function frame()  
        if (pos == 350) {  
            clearInterval(id);  
        } else {  
            pos++;  
            elem.style.top = pos + 'px';  
            elem.style.left = pos + 'px';  
        }  
    }  
}
```

[Click Me](#)



## Reading element state & data

### *What is DOM in JavaScript?*

JavaScript can access all the elements in a webpage making use of Document Object Model (DOM). In fact, the web browser creates a DOM of the webpage when the page is loaded. The DOM model is created as a tree of objects like this:

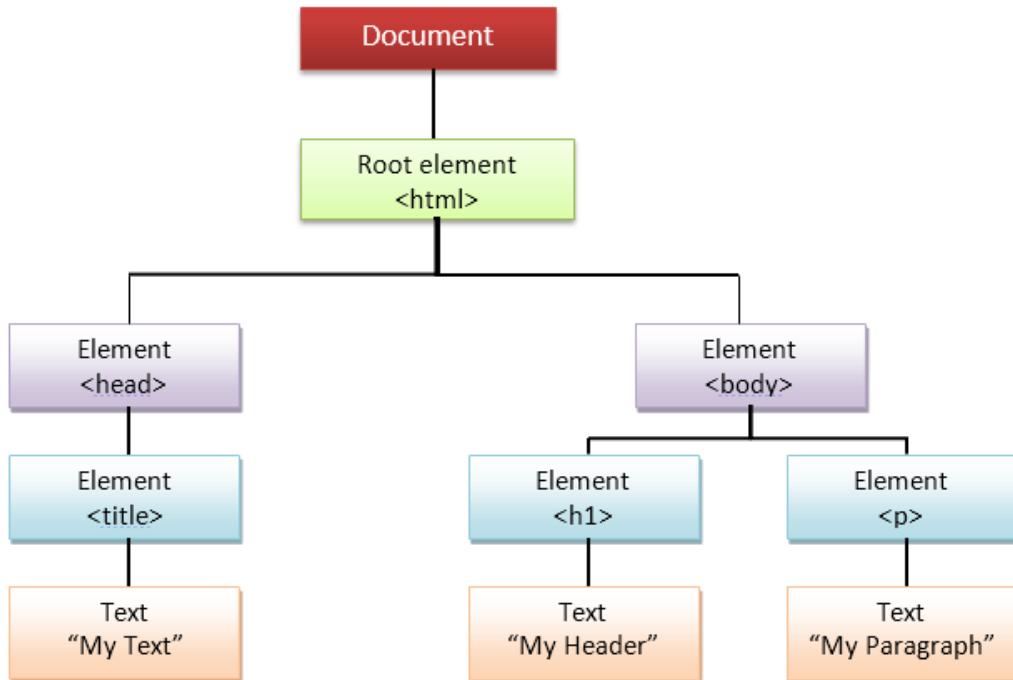


Image: DOM in Javascript  
Reference: [https://www.guru99.com/images/JavaScript/javascript8\\_1.png](https://www.guru99.com/images/JavaScript/javascript8_1.png)

### *How to use DOM and Events*

Using DOM, JavaScript can perform multiple tasks. It can create new elements and attributes, change the existing elements and attributes and even remove existing elements and attributes. JavaScript can also react to existing events and create new events in the page.

### ***getElementById, innerHTML Example***

1. getElementById: To access elements and attributes whose id is set.
2. innerHTML: To access the content of an element.

Top of Form  
Bottom of Form

### **Finding HTML Elements**

Often, with JavaScript, you want to manipulate HTML elements.

To do so, you have to find the elements first. There are several ways to do this:

- Finding HTML elements by id
- Finding HTML elements by tag name
- Finding HTML elements by class name
- Finding HTML elements by CSS selectors

### **Finding HTML Element by Id**

The easiest way to find an HTML element in the DOM, is by using the element id. This example finds the element with `id="intro"`:

```
var myElement = document.getElementById("intro");
```

If the element is found, the method will return the element as an object (in `myElement`).

If the element is not found, `myElement` will contain `null`.

### **Finding HTML Elements by Tag Name**

This example finds all `<p>` elements:

```
var x = document.getElementsByTagName("p");
```

This example finds the element with `id="main"`, and then finds all `<p>` elements inside "main":

```
var x = document.getElementById("main");
var y = x.getElementsByTagName("p");
```

### **Finding HTML Elements by Class Name**

If you want to find all HTML elements with the same class name, use `getElementsByClassName()`.

This example returns a list of all elements with `class="intro"`.

```
var x = document.getElementsByClassName("intro");
```

Finding elements by class name does not work in Internet Explorer 8 and earlier versions.

## Finding HTML Elements by CSS Selectors

If you want to find all HTML elements that match a specified CSS selector (id, class names, types, attributes, values of attributes, etc), use the `querySelectorAll()` method. This example returns a list of all `<p>` elements with `class="intro"`.

```
var x = document.querySelectorAll("p.intro");

<html>
<head>
    <title>DOM!!!</title>
</head>
<body>
    <h1 id="one">Welcome</h1>
    <p>This is the welcome message.</p>
    <h2>Technology</h2>
    <p>This is the technology section.</p>
    <script type="text/javascript">
        var text = document.getElementById("one").innerHTML;
        alert("The first heading is " + text);
    </script>
</body>
</html>
```

## Form handling and validations

### *Form handling*

Forms are an extremely important part of HTML and the Web Platform. They allow users can interact with the page and search something on the site trigger filters to trim result pages send information and much much more.

By default, forms submit their content to a server-side endpoint, which by default is the page URL itself:

```
<form>
    <input type="submit">
</form>
```

We can override this behavior by setting the `action` attribute of the form element, using the HTML method defined by the `method` attribute, which defaults to GET:

```
<form action="/contact" method="POST">
    <input type="submit">
</form>
```

Upon clicking the submit input element, the browser makes a POST request to the /contact URL on the same origin (protocol, domain and port).

Using JavaScript we can intercept this event, submit the form asynchronously (with XHR and Fetch), and we can also react to events happening on individual form elements.

## Working with input element events

We have a number of events we can listen for in form elements

- **input** fired on form elements when the element value is changed
- **change** fired on form elements when the element value is changed. In the case of text input elements and textarea, it's fired only once when the element loses focus (not for every single character typed)
- **cut** fired when the user cuts text from the form element
- **copy** fired when the user copies text from the form element
- **paste** fired when the user pastes text into the form element
- **focus** fired when the form element gains focus
- **blur** fired when the form element loses focus

## Validation

HTML form validation can be done by JavaScript.

If a form field (fname) is empty, this function alerts a message, and returns false, to prevent the form from being submitted:

```
function validateForm()
{
    var x = document.forms["myForm"]["fname"].value;
    if (x == "") {
        alert("Name must be filled out");
        return false;
    }
}
```

The function can be called when the form is submitted:

```
<form name="myForm" action="/action_page.php" onsubmit="return validateForm()" method="post">
Name: <input type="text" name="fname">
<input type="submit" value="Submit">
</form>
```



## Automatic HTML Form Validation

HTML form validation can be performed automatically by the browser:  
If a form field (fname) is empty, the **required** attribute prevents this form from being submitted:

```
<form action="/action_page.php" method="post">
<input type="text" name="fname" required>
<input type="submit" value="Submit">
</form>
```

## Data Validation

Data validation is the process of ensuring that user input is clean, correct, and useful. Typical validation tasks include:

- has the user filled in all required fields?
- has the user entered a valid date?
- has the user entered text in a numeric field?
- Most often, the purpose of data validation is to ensure correct user input.

Validation can be defined by many different methods, and deployed in many different ways.

**Server side validation** is performed by a web server, after input has been sent to the server.

**Client side validation** is performed by a web browser, before input is sent to a web server.

## Handling Cookies and Session Data

### *Cookies in JavaScript*

#### What are Cookies?

A cookie is a piece of data that is stored on your computer to be accessed by your browser. You also might have enjoyed the benefits of cookies knowingly or unknowingly. Have you ever saved your Facebook password so that you do not have to type it each and every time you try to login? If yes, then you are using cookies. Cookies are saved as key/value pairs.

#### Why do you need a Cookie?

The communication between a web browser and server happens using a stateless protocol named HTTP. Stateless protocol treats each request independent. So, the server does not keep the data after sending it to the browser. But in many situations, the data will be required again. Here come cookies into a picture. With cookies, the web browser will not have to communicate with the server each time the data is required. Instead, it can be fetched directly from the computer.

Cookies are a plain text data record of 5 variable-length fields –

**Expires** – The date the cookie will expire. If this is blank, the cookie will expire when the visitor quits the browser.

**Domain** – The domain name of your site.

**Path** – The path to the directory or web page that set the cookie. This may be blank if you want to retrieve the cookie from any directory or page.

**Secure** – If this field contains the word "secure", then the cookie may only be retrieved with a secure server. If this field is blank, no such restriction exists.

**Name=Value** – Cookies are set and retrieved in the form of key-value pairs

## Javascript Set Cookie

You can create cookies using document.cookie property like this.

```
document.cookie = "cookiename=cookievalue"
```

You can even add expiry date to your cookie so that the particular cookie will be removed from the computer on the specified date. The expiry date should be set in the UTC/GMT format. If you do not set the expiry date, the cookie will be removed when the user closes the browser.

```
document.cookie = "cookiename=cookievalue; expires= Thu, 21 Aug 2014 20:00:00 UTC"
```

You can also set the domain and path to specify to which domain and to which directories in the specific domain the cookie belongs to. By default, a cookie belongs to the page that sets the cookie.

```
document.cookie = "cookiename=cookievalue; expires= Thu, 21 Aug 2014 20:00:00 UTC; path=/ "
```

```
//create a cookie with a domain to the current page and path to the entire domain.
```

## JavaScript get Cookie

You can access the cookie like this which will return all the cookies saved for the current domain.

```
var x = document.cookie
```

## JavaScript Delete Cookie

To delete a cookie, you just need to set the value of the cookie to empty and set the value of expires to a passed date.

```
document.cookie = "cookiename= ; expires = Thu, 01 Jan 1970 00:00:00 GMT"
```

```
<html>
<head>
    <title>Cookie!!!</title>
    <script type="text/javascript">
        function createCookie(cookieName,cookieValue,daysToExpire)
    {
        var date = new Date();
```

```

        date.setTime(date.getTime()+(daysToExpire*24*60*60*1000));
        document.cookie = cookieName + "=" + cookieValue + "; expires=" +
date.toGMTString();
    }
    function accessCookie(cookieName)
{
    var name = cookieName + "=";
    var allCookieArray = document.cookie.split(';");
    for(var i=0; i<allCookieArray.length; i++)
    {
        var temp = allCookieArray[i].trim();
        if (temp.indexOf(name)==0)
            return temp.substring(name.length,temp.length);
    }
    return "";
}
function checkCookie()
{
    var user = accessCookie("testCookie");
    if (user!="")
        alert("Welcome Back " + user + "!!!");
    else
    {
        user = prompt("Please enter your name");
        num = prompt("How many days you want to store your name on your
computer?");
        if (user!="" && user!=null)
        {
            createCookie("testCookie", user, num);
        }
    }
}
</script>
</head>
<body onload="checkCookie()"></body>
</html>

```

Application

- Manifest
- Service Workers
- Storage

Storage

- Local Storage
- Session Storage
- IndexedDB
- Web SQL
- Cookies

<https://www.guru99.com>

Filter

Name	Value	Domain	Path	Expires...	Size	HttpOnly	Secure	SameSite	SameP...	Priority
1P_JAR	2021-04-13-16	.gstatic...	/	2021-0...	19	✓	✓	None		Medium
NID	213=Ykilcv90Vj7EkESNiMHepzY2QFqY7zsUkqRU...	.google...	/	2021-0...	217	✓	✓	None		Medium
OTZ	5925065_34_34_	www.g...	/	2021-0...	21	✓	✓			Medium
ANID	AHWqTUmjznBZN83q2YPAguzE28x0SWXL1_55v3...	.google...	/	2022-1...	68	✓	✓	None		Medium
testCookie	hbk	www.g...	/	2021-0...	13					Medium
DV	M8_8Q6bGAeQqcNh6fbLcG-jm_vq9jtc_b0uhmj4i...	www.g...	/	2021-0...	49					Medium

Only show cookies with an issue

Select a cookie to preview its value

## SessionStorage

The sessionStorage object stores data for only one session (the data is deleted when the browser tab is closed).

**Tip:** Also look at the localStorage property which stores data with no expiration date. The data will not be deleted when the browser is closed, and will be available the next day, week, or year.

**Syntax for SAVING data to sessionStorage:**

```
sessionStorage.setItem("key", "value");
```

**Syntax for READING data from sessionStorage:**

```
var lastname = sessionStorage.getItem("key");
```

**Syntax for REMOVING saved data from sessionStorage:**

```
sessionStorage.removeItem("key");
```

**Syntax for REMOVING ALL saved data from sessionStorage:**

```
sessionStorage.clear();
```

```
if (sessionStorage.clickcount) {  
  
    sessionStorage.clickcount = Number(sessionStorage.clickcount) + 1;  
} else {  
    sessionStorage.clickcount = 1;  
}  
document.getElementById("result").innerHTML = "You have clicked the button " +  
sessionStorage.clickcount + " time(s) in this session.";
```

```
<!DOCTYPE html>
<html>
<head>
<script>
function clickCounter() {
    if(typeof(Storage) !== "undefined") {
        if (sessionStorage.clickcount) {
            sessionStorage.clickcount = Number(sessionStorage.clickcount)+1;
        } else {
            sessionStorage.clickcount = 1;
        }
        document.getElementById("result").innerHTML = "You have clicked the button
" + sessionStorage.clickcount + " time(s) in this session.";
    } else {
        document.getElementById("result").innerHTML = "Sorry, your browser does not
support web storage...";
    }
}
</script>
</head>
<body>
<p><button onclick="clickCounter()" type="button">Click me!</button></p>
<div id="result"></div>
<p>Click the button to see the counter increase.</p>
<p>Close the browser tab (or window), and try again, and the counter is reset.
</p>
</body>
</html>
```

**Click me!**

You have clicked the button 1 time(s) in this session.

Click the button to see the counter increase.

Close the browser tab (or window), and try again, and the counter is reset.

## Practical Activities

### Event Handling Program:

```
<html>
<head>
<script type = "text/javascript">
<!--
function over() {
    document.write ("Mouse Over");
-->
```

```
        }
        function out() {
            document.write ("Mouse Out");
        }
        //-->
    </script>
</head>

<body>
    <p>Bring your mouse inside the division to see the result:</p>
    <div onmouseover = "over()" onmouseout = "out()">
        <h2> This is inside the division </h2>
    </div>
</body>
</html>
```

**Output/Results snippet:**

Bring your mouse inside the division to see the result: \_\_\_\_\_

**This is inside the division**      **Mouse Over**

**DOM  
Program:**

```
<!DOCTYPE html>
<html>
<body>

<p id="demo" onclick="myFunction()">Click me to change my HTML content
(innerHTML).</p>

<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Paragraph changed!";
}
</script>

</body>
</html>
```

**Output/Results snippet:**

Paragraph changed!

**Animations  
Program:**

```
<!DOCTYPE HTML>
<html>

<head>
<style>
#train {
    position: relative;
    cursor: pointer;
}
</style>
</head>

<body>



<script>
train.onclick = function() {
    let start = Date.now();

    let timer = setInterval(function() {
        let timePassed = Date.now() - start;

        train.style.left = timePassed / 5 + 'px';

        if (timePassed > 2000) clearInterval(timer);
    }, 20);
}
</script>

</body>

</html>
```

**Output/Results snippet:**Result

index.html



## Cookies & Session Handling

### Program:

```
<!DOCTYPE html>
<html>
<head>
<script>
function setCookie(cname,cvalue,exdays) {
    var d = new Date();
    d.setTime(d.getTime() + (exdays*24*60*60*1000));
    var expires = "expires=" + d.toGMTString();
    document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";
}

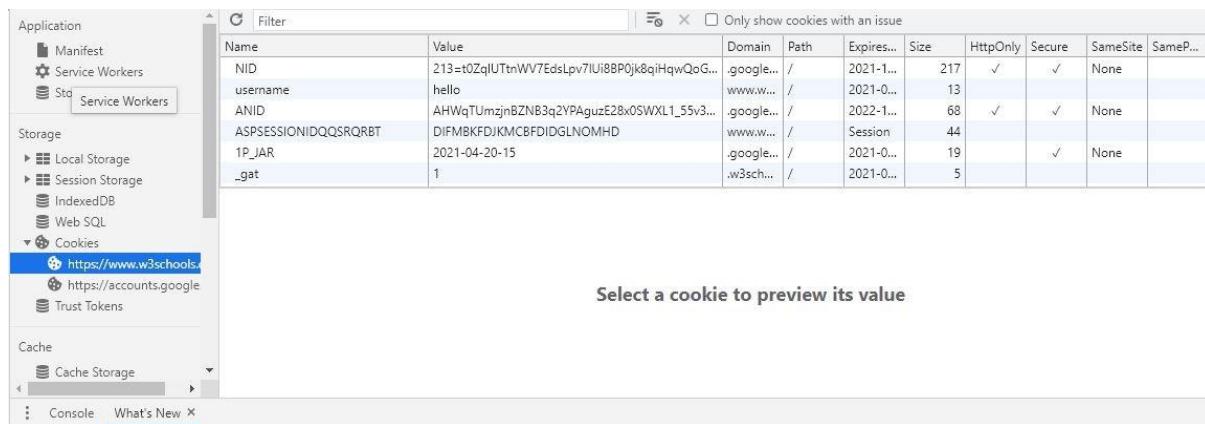
function getCookie(cname) {
    var name = cname + "=";
    var decodedCookie = decodeURIComponent(document.cookie);
    var ca = decodedCookie.split(';');
    for(var i = 0; i < ca.length; i++) {
        var c = ca[i];
        while (c.charAt(0) == ' ') {
            c = c.substring(1);
        }
        if (c.indexOf(name) == 0) {
            return c.substring(name.length, c.length);
        }
    }
    return "";
}

function checkCookie() {
    var user=getCookie("username");
    if (user != "") {
        alert("Welcome again " + user);
    } else {
        user = prompt("Please enter your name:","");
        if (user != "" && user != null) {
            setCookie("username", user, 30);
        }
    }
}
</script>
</head>

<body onload="checkCookie()"></body>

</html>
```

### Output/Results snippet:



Name	Value	Domain	Path	Expires...	Size	HttpOnly	Secure	SameSite	SameP...
NID	213=t0ZqjUTnWV7EdsLpv7IUi8BP0jk&qiHqwQoG...	.google...	/	2021-1...	217	✓	✓	None	
username	hello	www.w...	/	2021-0...	13				
ANID	AHWqTUmzjnBZNB3q2YPAguzE28x0SWXL1_55v3...	.google...	/	2022-1...	68	✓	✓	None	
ASPSESSIONIDQQSRQRTB	DIFMBKFDJMKCBFDIDGLNOMHD	www.w...	/	Session	44				
1P_JAR	2021-04-20-15	.google...	/	2021-0...	19		✓	None	
_gat	1	.w3sch...	/	2021-0...	5				

Select a cookie to preview its value

## Session

### Program:

```
<!DOCTYPE html>
<html>
<head>
<script>
function clickCounter() {
  if(typeof(Storage) !== "undefined") {
    if (sessionStorage.clickcount) {
      sessionStorage.clickcount = Number(sessionStorage.clickcount)+1;
    } else {
      sessionStorage.clickcount = 1;
    }
    document.getElementById("result").innerHTML = "You have clicked the button " +
    sessionStorage.clickcount + " time(s) in this session.";
  } else {
    document.getElementById("result").innerHTML = "Sorry, your browser does not
support web storage... ";
  }
}
</script>
</head>
<body>
<p><button onclick="clickCounter()" type="button">Click me!</button></p>
<div id="result"></div>
<p>Click the button to see the counter increase.</p>
<p>Close the browser tab (or window), and try again, and the counter is reset.</p>
</body>
</html>
```

### Output/Results snippet:

**Click me!**

You have clicked the button 4 time(s) in this session.

Click the button to see the counter increase.

Close the browser tab (or window), and try again, and the counter is reset.

We discussed in this module about HTML, CSS, and JavaScript, which are often called the building blocks of the Web. These three tools dominate web development. We saw HTML as a markup language used to structure our page into elements such as paragraphs, sections, headings, navigation bars, and so on. CSS, on the other hand is a design language used to make our web page look nice and presentable. We learned how to use CSS to layout elements by positioning them in specified areas of our page. Finally, we discussed JavaScript as the programming language, which makes our webpage to think and act. Together we use these three languages to format, design and program web pages.

# Chapter 4: Introduction to JAVA Programming Language

## Learning Outcomes:

- Introduction to JAVA Programming
- Java Fundamentals and Java Control Statements
- Java Class and Objects
- Important Topics of OOPS Concept
- Exception Handling

## 4.1 Introduction to JAVA Programming

### What is Java?

Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented and secure programming language. Java was developed by Sun Microsystems (which is now the subsidiary of Oracle) in the year 1995. James Gosling is known as the father of Java. Before Java, its name was Oak. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java.

**Platform:** Any hardware or software environment in which a program runs, is known as a platform. Since Java has a runtime environment (JRE) and API, it is called a platform.

**Application:** According to Sun, 3 billion devices run Java. There are many devices where Java is currently used. Some of them are as follows:

1. Desktop Applications such as acrobat reader, media player, antivirus, etc.
2. Web Applications such as irctc.co.in, javatpoint.com, etc.
3. Enterprise Applications such as banking applications.
4. Mobile
5. Embedded System
6. Smart Card
7. Robotics
8. Games, etc.

### Types of Java Applications

There are mainly 4 types of applications that can be created using Java programming:

**1) Standalone Application:** Standalone applications are also known as desktop applications or window-based applications. These are traditional software that we need to install on every machine. Examples of standalone application are Media player, antivirus, etc. AWT and Swing are used in Java for creating standalone applications.

**2) Web Application:** An application that runs on the server side and creates a dynamic page is called a web application. Currently, [Servlet](#), [JSP](#), [Struts](#), [Spring](#), [Hibernate](#), [JSF](#), etc. technologies are used for creating web applications in Java.

**3) Enterprise Application:** An application that is distributed in nature, such as banking applications, etc. is called an enterprise application. It has advantages like high-level security, load balancing, and clustering. In Java, [EJB](#) is used for creating enterprise applications.

**4) Mobile Application:** An application which is created for mobile devices is called a mobile application. Currently, Android and Java ME are used for creating mobile applications.

## Java Platforms / Editions

There are 4 platforms or editions of Java:

**1) Java SE (Java Standard Edition):** It is a Java programming platform. It includes Java programming APIs such as `java.lang`, `java.io`, `java.net`, `java.util`, `java.sql`, `java.math` etc. It includes core topics like OOPs, String, Regex, Exception, Inner classes, Multithreading, I/O Stream, Networking, AWT, Swing, Reflection, Collection, etc.

**2) Java EE (Java Enterprise Edition):** It is an enterprise platform that is mainly used to develop web and enterprise applications. It is built on top of the Java SE platform. It includes topics like Servlet, JSP, Web Services, EJB, JPA, etc.

**3) Java ME (Java Micro Edition):** It is a micro platform that is dedicated to mobile applications.

**4) JavaFX:** It is used to develop rich internet applications. It uses a lightweight user interface API.

## Why Use Java?

- Java works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.)
- It is one of the most popular programming languages in the world
- It is easy to learn and simple to use
- It is open-source and free
- It is secure, fast and powerful
- It has a huge community support (tens of millions of developers)
- Java is an object-oriented language which gives a clear structure to programs and allows code to be reused, lowering development costs
- As Java is close to C++ and C#, it makes it easy for programmers to switch to Java or vice versa

## Simple Java Program

A source program is a text file that contains a program (such as above) written in a programming language. Since it contains ordinary text (stored as bytes) it cannot be directly executed (run) by the computer system. As a text file, you can print it, display it on the monitor, or alter it with a text editor.

```
class Hello
{
    public static void main ( String[] args )
    {
        System.out.println("hello");
    }
}
```

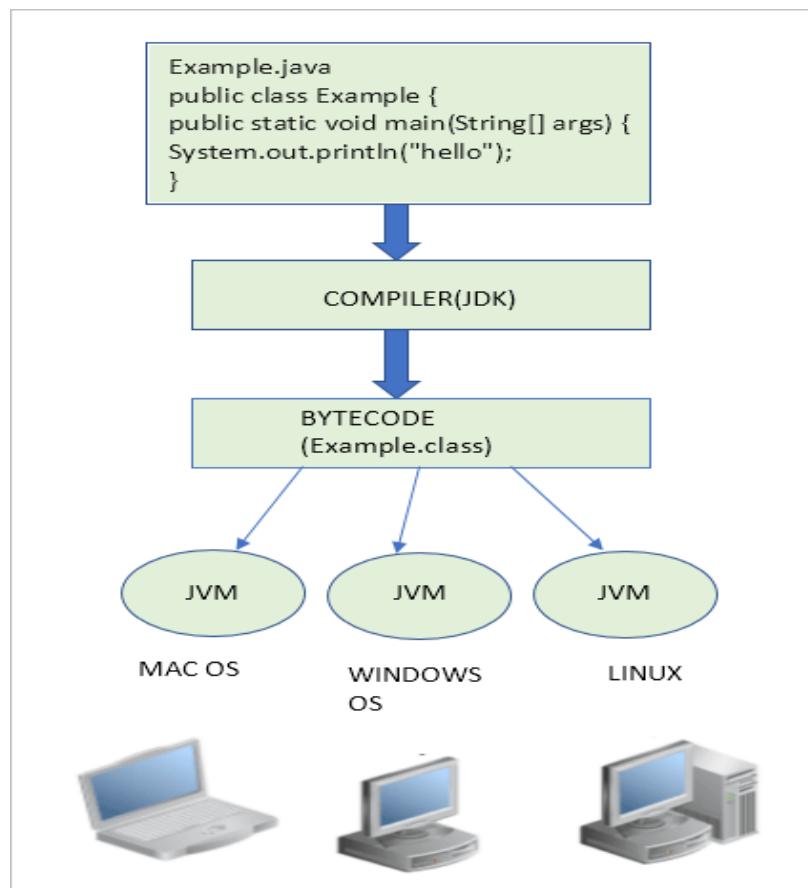


Image: Simple Java Program Execution Steps  
Reference: <https://www.softwaretestinghelp.com/java/java-introduction-installation/>

## Explanation:

**Comments:** Comments are used for explaining code and are used in a similar manner in Java or C or C++. Compilers ignore the comment entries and do not execute them. Comments can be of a single line or multiple lines.

## SingleLineComments:

Syntax:

```
// Single line comment
```

### **Multi-line comments:**

Syntax:

```
/* Multi line comments */
```

**import java.io.\*:** This means all the classes of io package can be imported. Java io package provides a set of input and output streams for reading and writing data to files or other input or output sources.

**class:** The class contains the data and methods to be used in the program. Methods define the behavior of the class. Class GFG has only one method Main in JAVA.  
**static void Main():** static keyword tells us that this method is accessible without instantiating the class.

**void:** keywords tell that this method will not return anything. The main() method is the entry point of our application.

**System.in:** This is the standard input stream that is used to read characters from the keyboard or any other standard input device.

**System.out:** This is the standard output stream that is used to produce the result of a program on an output device like the computer screen.

**println():** This method in Java is also used to display text on the console. It prints the text on the console and the cursor moves to the start of the next line at the console. The next printing takes place from the next line.

**String []args:** This is the argument passed to the main function which is an array of strings with the array name args. One can choose their own flexible name but this name is used by many developers.

## **Features of Java**

The primary objective of Java programming language creation was to make it portable, simple and secure programming language. Apart from this, there are also some excellent features which play an important role in the popularity of this language. The features of Java are also known as Java buzzwords. A list of the most important features of the Java language is given below.

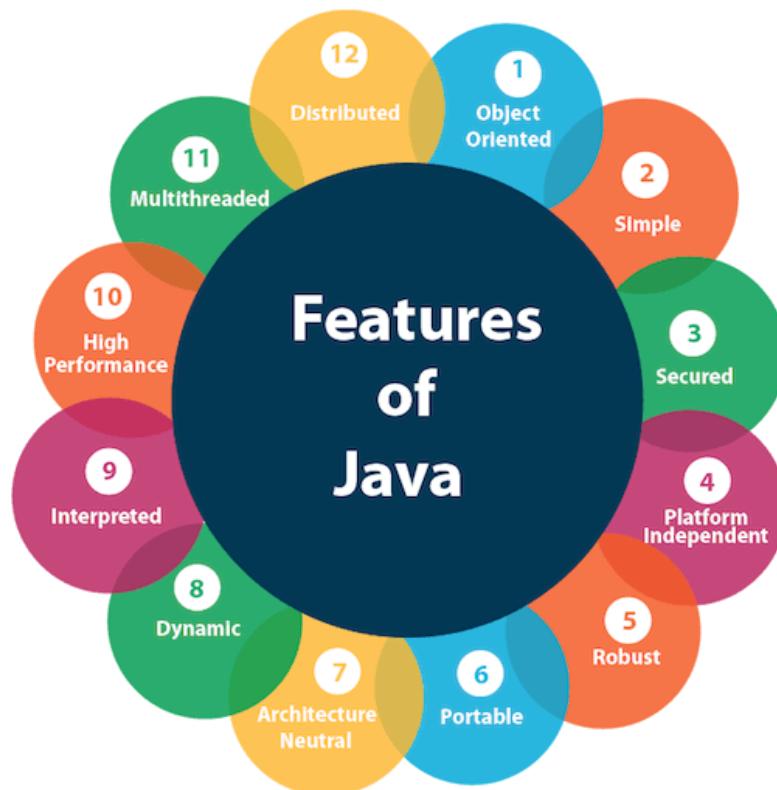


Image: Java Features

Reference: <https://www.javatpoint.com/features-of-java>

- Simple
- Object-Oriented
- Portable
- Platform independent
- Secured
- Robust
- Architecture neutral
- Interpreted
- High Performance
- Multithreaded
- Distributed
- Dynamic

## Simple

Java is very easy to learn, and its syntax is simple, clean and easy to understand. According to Sun Microsystem, Java language is a simple programming language because:

- Java syntax is based on C++ (so easier for programmers to learn it after C++).
- Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.
- There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.

## Object-oriented

Java is an object-oriented programming language. Everything in Java is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporate both data and behavior.

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

Basic concepts of OOPs are:

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

## Platform Independent

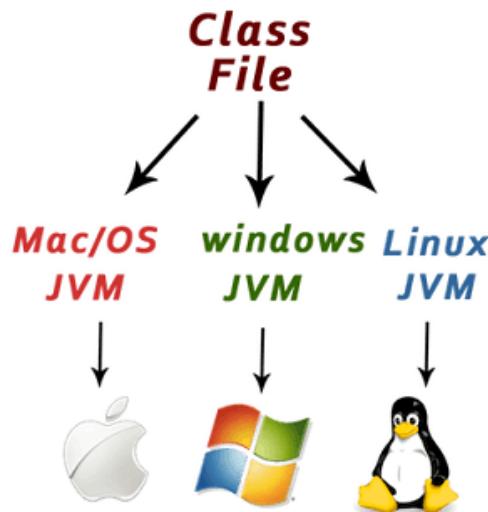


Image: Platform Independent  
Reference: <https://www.javatpoint.com/features-of-java>

Java is platform independent because it is different from other languages like C, C++, etc. which are compiled into platform specific machines while Java is a write once, run anywhere language. 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 a software-based platform.

The Java platform differs from most other platforms in the sense that it is 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 executed on multiple platforms, for example, Windows, Linux, Sun Solaris, Mac/OS, etc. Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform-independent code because it can be run on multiple platforms, i.e., Write Once and Run Anywhere (WORA).

## Secured

Java is best known for its security. With Java, we can develop virus-free systems. Java is secured because:

- No explicit pointer
- Java Programs run inside a virtual machine sandbox

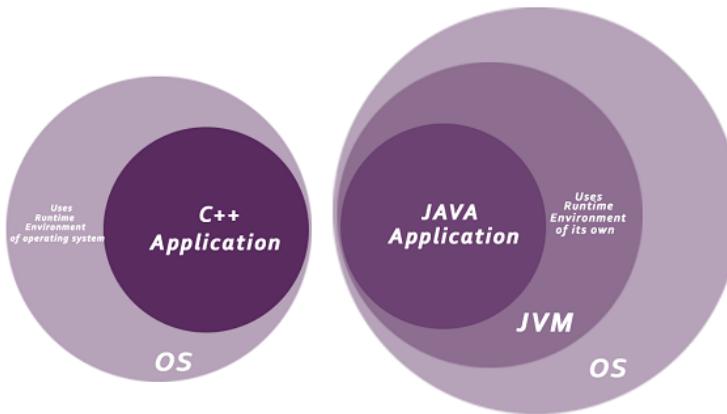


Image: Secured

Reference: <https://www.javatpoint.com/features-of-java>

## ClassLoader

ClassLoader in Java is a part of the Java Runtime Environment (JRE) which is used to load Java classes into the Java Virtual Machine dynamically. It adds security by separating the package for the classes of the local file system from those that are imported from network sources.

## Bytecode Verifier

It checks the code fragments for illegal code that can violate access rights to objects.

## Security Manager

It determines what resources a class can access such as reading and writing to the local disk.

Java language provides these securities by default. Some security can also be provided by an application developer explicitly through SSL, JAAS, Cryptography, etc.

## Robust

The English meaning of Robust is strong. Java is robust because:

- It uses strong memory management.
- There is a lack of pointers that avoids security problems.
- Java provides automatic garbage collection which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.
- There are exception handling and the type checking mechanism in Java. All these points make Java robust.

## Architecture-neutral

Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.

In C programming, int data type occupies 2 bytes of memory for 32-bit architecture and 4 bytes of memory for 64-bit architecture. However, it occupies 4 bytes of memory for both 32 and 64-bit architectures in Java.

## Portable

Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.

## High-performance

Java is faster than other traditional interpreted programming languages because Java bytecode is "close" to native code. It is still a little bit slower than a compiled language (e.g., C++). Java is an interpreted language that is why it is slower than compiled languages, e.g., C, C++, etc.

## Distributed

Java is distributed because it facilitates users to create distributed applications in Java. RMI and EJB are used for creating distributed applications. This feature of Java makes us able to access files by calling the methods from any machine on the internet.

## Multi-threaded

A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads. The main advantage of multi-threading is that it doesn't occupy memory for each thread. It shares a common memory area. Threads are important for multi-media, Web applications, etc.

## Dynamic

Java is a dynamic language. It supports the dynamic loading of classes. It means classes are loaded on demand. It also supports functions from its native languages, i.e., C and C++. Java supports dynamic compilation and automatic memory management (garbage collection).

## History of Java

The history of Java is very interesting. Java was originally designed for interactive television, but it was too advanced technology for the digital cable television industry at the time. The history of Java starts with the Green Team. Java team members (also known as Green Team), initiated this project to develop a language for digital devices such as set-top boxes, televisions, etc. However, it was best suited for internet programming. Later, Java technology was incorporated by Netscape. The principles

for creating Java programming were "Simple, Robust, Portable, Platform-independent, Secured, High Performance, Multithreaded, Architecture Neutral, Object-Oriented, Interpreted, and Dynamic". Java was developed by James Gosling, who is known as the father of Java, in 1995. James Gosling and his team members started the project in the early '90s.

### James Gosling - founder of java

Currently, Java is used in internet programming, mobile devices, games, e-business solutions, etc. Following are the given significant points that describe 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.

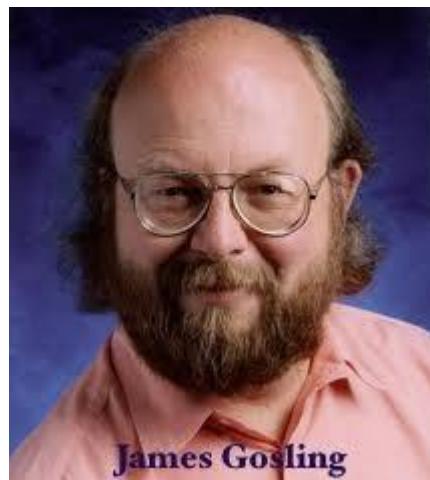


Image: Founder of JAVA  
Reference: <https://www.javatpoint.com/history-of-java>

2. Initially it was designed for small, embedded systems in electronic appliances like set-top boxes.
3. Firstly, it was called "Greentalk" by James Gosling, and the file extension was .gt.
4. After that, it was called Oak and was developed as a part of the Green project.
5. Why Oak? Oak is a symbol of strength and chosen as a national tree of many countries like the U.S.A., France, Germany, Romania, etc.
6. In 1995, Oak was renamed as "Java" because it was already a trademark by Oak Technologies.

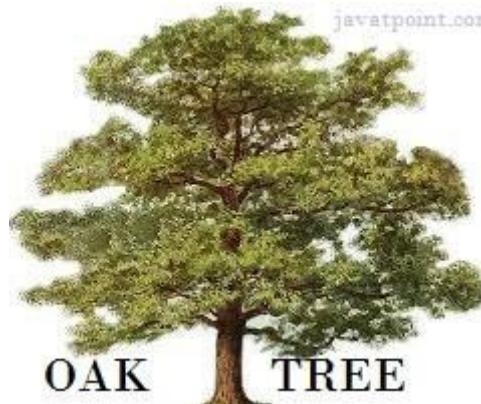


Image: OAK Tree

Reference: <https://www.javatpoint.com/history-of-java>

7. Why had they chose the name Java 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.
8. 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 than other names.
9. Java is an island in Indonesia where the first coffee was produced (called Java coffee). It is a kind of espresso bean. Java name was chosen by James Gosling while having a cup of coffee nearby his office.
10. Notice that Java is just a name, not an acronym.
11. Initially developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995.
12. In 1995, Time magazine called Java one of the Ten Best Products of 1995.

JDK 1.0 was released on January 23, 1996. After the first release of Java, there have been many additional features added to the language. Now Java is being used in Windows applications, Web applications, enterprise applications, mobile applications, cards, etc. Each new version adds new features in Java.

## Java Version History

Many java versions have been released till now. The current stable release of Java is Java SE 10.

1. JDK Alpha and Beta (1995)
2. JDK 1.0 (23rd Jan 1996)
3. JDK 1.1 (19th Feb 1997)
4. J2SE 1.2 (8th Dec 1998)
5. J2SE 1.3 (8th May 2000)
6. J2SE 1.4 (6th Feb 2002)
7. J2SE 5.0 (30th Sep 2004)
8. Java SE 6 (11th Dec 2006)
9. Java SE 7 (28th July 2011)

10. Java SE 8 (18th Mar 2014)
11. Java SE 9 (21st Sep 2017)
12. Java SE 10 (20th Mar 2018)
13. Java SE 11 (September 2018)
14. Java SE 12 (March 2019)
15. Java SE 13 (September 2019)
16. Java SE 14 (Mar 2020)
17. Java SE 15 (September 2020)
18. Java SE 16 (Mar 2021)
19. Java SE 17 (September 2021)
20. Java SE 18 (to be released by March 2022)

JDK 1.0 released in (January 23, 1996). **Java SE 18** is a current stable release of Java, and many other previous Java versions are also available.

## Java Terminology

**1. Java Virtual Machine (JVM):** This is generally referred to as JVM. There are three execution phases of a program. They are written, compile and run the program.

- Writing a program is done by a java programmer like you and me.
- The compilation is done by the **JAVAC** compiler which is a primary Java compiler included in the Java development kit (JDK). It takes the Java program as input and generates bytecode as output.
- In the Running phase of a program, **JVM** executes the bytecode generated by the compiler.

Now, we understood that the function of Java Virtual Machine is to execute the bytecode produced by the compiler. Every Operating System has a different JVM but the output they produce after the execution of bytecode is the same across all the operating systems. This is why Java is known as a **platform-independent language**. JVMs are available for many hardware and software platforms (i.e., JVM is platform dependent).

**2. Bytecode in the Development process:** As discussed, the Java compiler of JDK compiles the java source code into bytecode so that it can be executed by JVM. It is saved as **.class** file by the compiler. To view the bytecode, a disassembler like javap can be used.

**3. Java Development Kit (JDK):** While we were using the term JDK when we learned about bytecode and JVM. So, as the name suggests, it is a complete Java development kit that includes everything including compiler, Java Runtime Environment (JRE), java debuggers, java docs, etc. For the program to execute in java, we need to install JDK on our computer in order to create, compile and run the java program.

**4. Java Runtime Environment (JRE):** JDK includes JRE. JRE installation on our computers allows the java program to run, however, we cannot compile it. JRE includes a browser, JVM, applet supports, and plugins. For running the java program, a computer needs JRE.

**5. Garbage Collector:** In Java, programmers can't delete the objects. To delete or recollect that memory JVM has a program called Garbage Collector. Garbage Collectors can recollect the objects that are not referenced. So Java makes the life of a programmer easy by handling memory management. However, programmers should be careful about their code whether they are using objects that have been used for a long time. Because Garbage cannot recover the memory of objects being referenced.

**6. ClassPath:** The classpath is the file path where the java runtime and Java compiler look for **.class** files to load. By default, JDK provides many libraries. If you want to include external libraries they should be added to the classpath.

### What is Bytecode in Java and it's working?

Bytecode in Java is an intermediate machine-independent code. It is a set of instructions for Java Virtual Machine and it acts pretty similar to the assembler in C++. In general, bytecode is a code that lies between low-level and high-level language. The bytecode is not processed by the processor. It is processed by the Java Virtual Machine (JVM). The job of the JVM is to call all the required resources to compile the Java program and make the bytecode independent. It is the biggest reason why java is known as a platform-independent language. The intermediate code can run on any of the platforms such as Windows, macOS, and Linux.

Byte Code can be defined as an intermediate code generated by the compiler after the compilation of source code(JAVA Program). This intermediate code makes Java a platform-independent language.

### Working of Java Bytecode

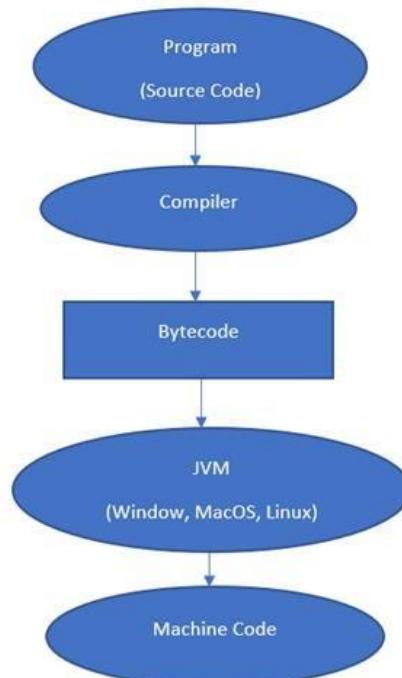


Image: Working of JAVA ByteCode  
Reference: <https://tutorialslink.com/Articles/What-is-Bytecode-in-Java/1552>

**Program:** Program is a .java file. It consists of the code that you have written.

**Compiler:** Compiler complies the .java file and generate .class file.

**Bytecode:** The .class file contains the bytecode. Now, we can run the .class file in any of the other platforms.

**JVM:** JVM runs the bytecode without considering a processor.

**Machine Code:** Now, the machine generates its own machine code in which the byte code is running. That means, its own machine-dependent code to run the .java file.

The only essential requirement for running the bytecode is the installation of basic java in any platform.

### Advantages of bytecode:

- It helps in achieving the platform-independent goal with the help of bytecode.
- The set of instructions for JVM may differ from one system to another but all systems can run the bytecode.
- Bytecode runs only when the interpreter is available.
- It runs on the Java virtual machine only.
- It gives flexibility by giving a quote ‘Write code once, run code anywhere’.
- It also saves a lot of time for a programmer.
- It is of low cost however it gives a high return.

### Disadvantages of Bytecode:

- It takes more time to run the bytecode than the machine code which is machine specific.
- It is difficult to use some platform-specific features because java is platform-independent.
- Mandatory installation of Java interpreter to run the byte code.

## JVM, JRE, JDK

### *JVM: Java Virtual Machine*

- JVM is responsible for taking .class file and converting that .class file in machine code instructions that can be executed by microprocessor.
- It's not a machine, it's a program.
- JVM verifies the code before execution (Sandbox Security).
- JVM is platform dependent and it is heart of Java language.

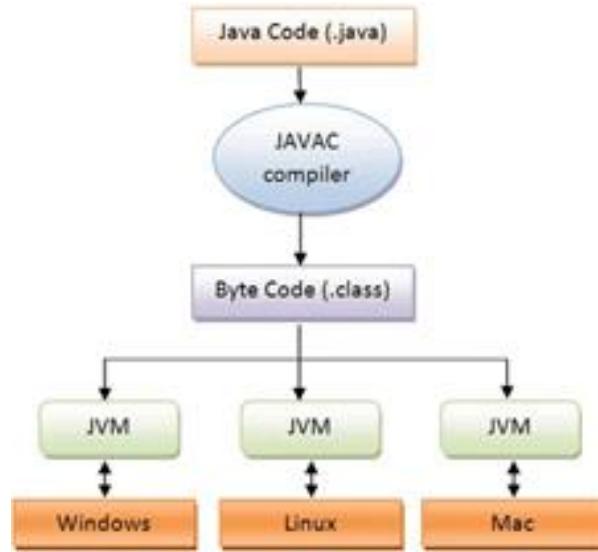


Image: Working of JAVA Bytecode

Reference: <https://www.thecrazyprogrammer.com/ezoimgfmt/3.bp.blogspot.com/-02ntSdF6Y3I/U5DMshp7s6I/AAAAAAAACZA/K7sAbw8Urc/s1600/java+program+execution.png?ezoimgfmt=rs:400x369/rscb1/ng:webp/ngcb1>

The JVM performs the following main tasks:

- Loads code
- Verifies code
- Executes code
- Provides runtime environment

### **JVM Architecture**

It contains class loader, memory area, execution engine etc.

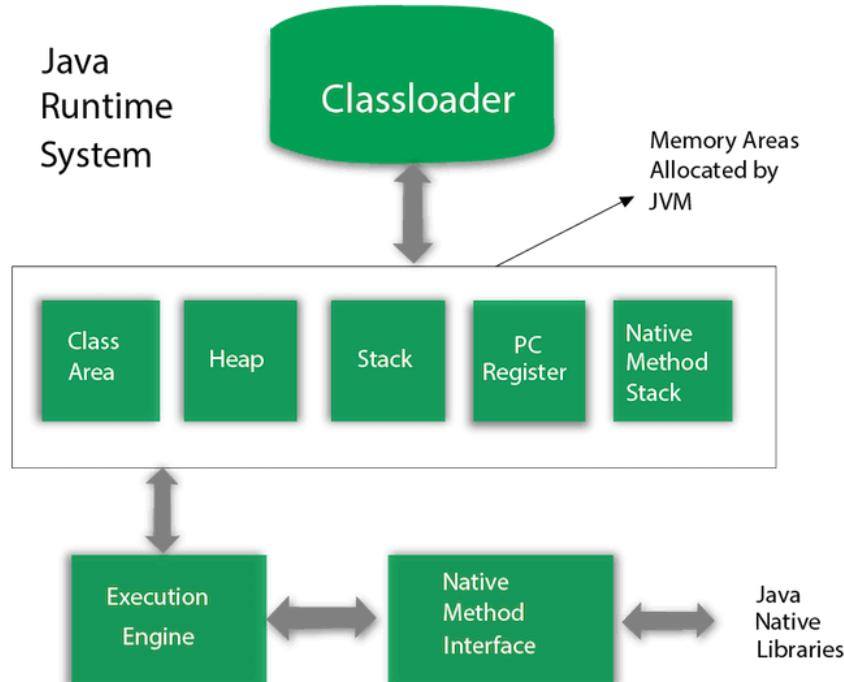


Image: JVM Architecture

Reference: <https://www.javatpoint.com/jvm-java-virtual-machine>

## Classloader

Classloader is a subsystem of JVM which is used to load class files. Whenever we run the java program, it is loaded first by the classloader. There are three built-in classloaders in Java.

- **Bootstrap ClassLoader:** This is the first classloader which is the super class of Extension classloader. It loads the *rt.jar* file which contains all class files of Java Standard Edition like *java.lang* package classes, *java.net* package classes, *java.util* package classes, *java.io* package classes, *java.sql* package classes etc.
- **Extension ClassLoader:** This is the child classloader of Bootstrap and parent classloader of System classloader. It loads the jar files located inside *\$JAVA\_HOME/jre/lib/ext* directory.
- **System/Application ClassLoader:** This is the child classloader of Extension classloader. It loads the classfiles from classpath. By default, classpath is set to current directory. You can change the classpath using "-cp" or "-classpath" switch. It is also known as Application classloader.

### //Let's see an example to print the classloader name

```
public class ClassLoaderExample
{
    public static void main(String[] args)
    {
        // Let's print the classloader name of current class.
        //Application/System classloader will load this class
        Class c=ClassLoaderExample.class;
        System.out.println(c.getClassLoader());
        //If we print the classloader name of String, it will print null because it is
        an
        //in-built class which is found in rt.jar, so it is loaded by Bootstrap
        classloader
        System.out.println(String.class.getClassLoader());
    }
}
```

### Output:

```
sun.misc.Launcher$AppClassLoader@4e0e2f2a
null
```

These are the internal classloaders provided by Java. If you want to create your own classloader, you need to extend the *ClassLoader* class.

**Class(Method) Area:** Class(Method) Area stores per-class structures such as the runtime constant pool, field and method data, the code for methods.

**Heap:** It is the runtime data area in which objects are allocated.

**Stack:** Java Stack stores frames. It holds local variables and partial results, and plays a part in method invocation and return. Each thread has a private JVM stack, created

at the same time as thread. A new frame is created each time a method is invoked. A frame is destroyed when its method invocation completes.

**Program Counter Register:** PC (program counter) register contains the address of the Java virtual machine instruction currently being executed.

**Native Method Stack:** It contains all the native methods used in the application.

**Execution Engine:** It contains a virtual processor

**Interpreter:** Read bytecode stream then execute the instructions.

**Just-In-Time (JIT) compiler:** It is used to improve the performance. JIT compiles parts of the byte code that have similar functionality at the same time, and hence reduces the amount of time needed for compilation. Here, the term "compiler" refers to a translator from the instruction set of a Java virtual machine (JVM) to the instruction set of a specific CPU.

**Java Native Interface:** Java Native Interface (JNI) is a framework which provides an interface to communicate with another application written in another language like C, C++, Assembly etc. Java uses JNI framework to send output to the Console or interact with OS libraries.

### **JRE: Java Runtime Environment**

JRE is an acronym for Java Runtime Environment. It is also written as Java RTE. The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.

JRE has an instance of JVM with it, library classes, and development tools. Once you write and compile Java code, the compiler generates a class file having bytecode. Here are the important components of JRE:

- **Class loaders:** The class loader loads various classes that are necessary for running a Java program. JVM uses three class loaders called the bootstrap class loader, extensions class loader, and system class loader.
- **Byte code verifier:** Byte code verifier verifies the bytecode so that the code doesn't disturb the interpreter.
- **Interpreter:** Once the classes get loaded, and the code is verified, the interpreter reads the code line by line.
- **Run-time:** Run-time is a system used mainly in programming to describe time period during which a particular program is running.
- **Hardware:** Once you compile Java native code, it runs on a specific hardware platform.

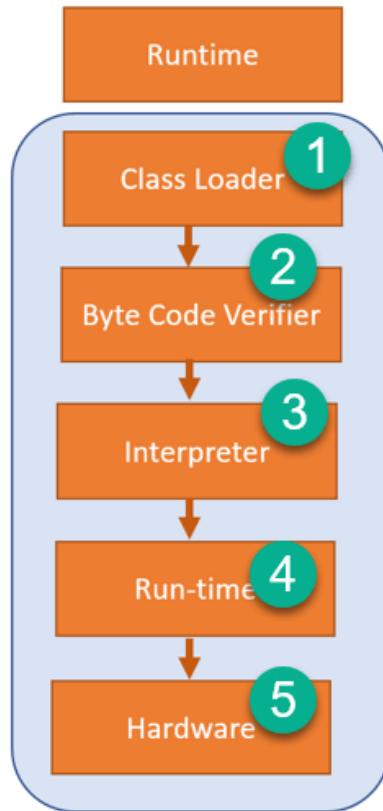
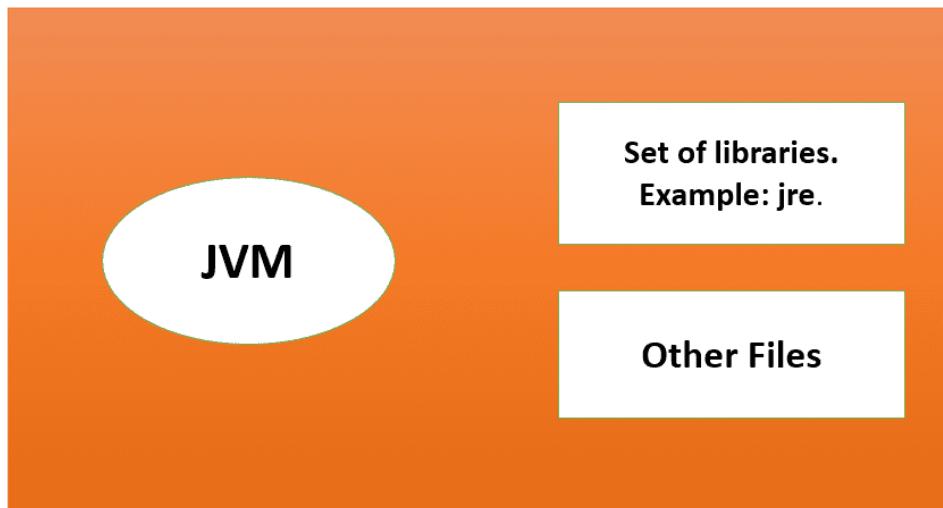


Image: JRE

Reference: <https://www.guru99.com/difference-between-jdk-jre-jvm.html>



## JRE

Image: JRE

Reference: <https://www.guru99.com/difference-between-jdk-jre-jvm.html>

## JDK

JDK is an acronym for Java Development Kit. The Java Development Kit (JDK) is a software development environment which is used to develop Java applications and applets. It physically exists. It contains JRE + development tools. JDK is an

implementation of any one of the below given Java Platforms released by Oracle Corporation:

- Standard Edition Java Platform
- Enterprise Edition Java Platform
- Micro Edition Java Platform

The JDK contains a private Java Virtual Machine (JVM) and a few other resources such as an interpreter/loader (java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc), etc. to complete the development of a Java Application. Here are the important components of JDK:

- **JDK and JRE:** The JDK enables programmers to create core Java programs that can be run by the JRE, which included JVM and class libraries.
- **Class Libraries:** It is a group of dynamically loadable libraries that Java program can call at run time.
- **Compilers:** It is a Java program that accepts text file of developers and compiles into Java class file. It is the common form of output given by compiler, which contains Java byte code. In Java, the primary compiler is Javac.
- **Debuggers:** Debugger is a Java program that lets developers test and debug Java programs.
- **JavaDoc:** JavaDoc is documentation made by Sun Microsystems for the Java. JavaDoc can be used generating API documentation in HTML file from the source program

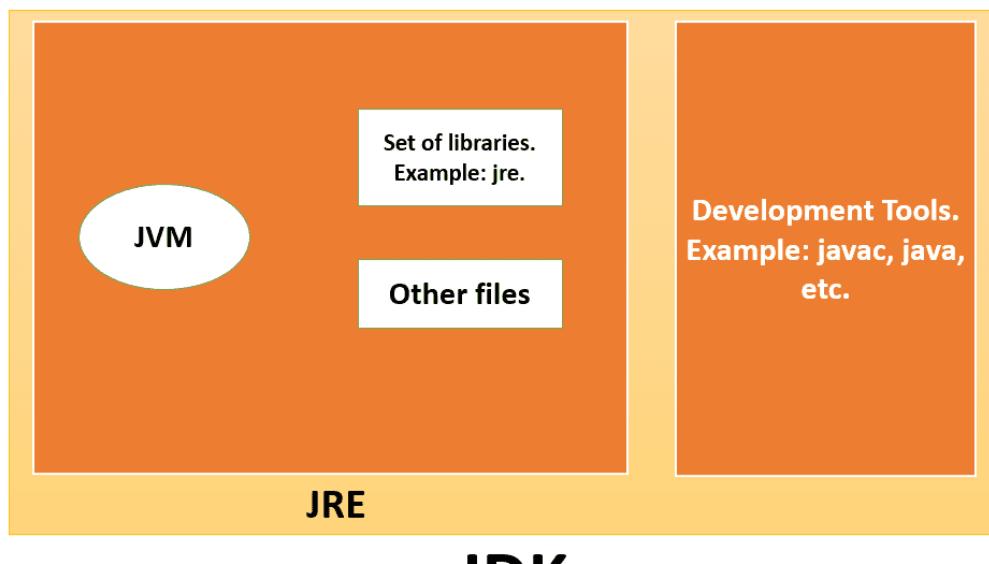


Image: JDK  
Reference: <https://www.guru99.com/difference-between-jdk-jre-jvm.html>

## Components of JDK in Java

The fundamental components of JDK in Java are listed below.

- 1) `java`: It acts as the deployment launcher in the older SUN `java`. It loads the class files and interprets the source code compiled by the `javac` compiler.
- 2) `javac`: The `javac` specifies the `java` compiler to convert the source code into bytecode.

- 3) javadoc: The javadoc generates documentation for the comments added in the source code.
- 4) jar: The jar helps the archives to manage the jar files in the package library.
- 5) jps: The jps stands for Java Virtual Machine Process Status Tool. It manages the active JVMs for the currently executing program.
- 6) appletviewer: The appletviewer is designed to run and debug Java applets without the help of an internet browser.
- 7) idlj: An IDL-to-Java compiler generates Java bindings from a given Java IDL file.
- 8) javap: The javap acts as a file disassembler.
- 9) JConsole: JConsole acts as a Java Management and Monitoring unit.
- 10) javah: The javah is a stub-generator, and C-Header is employed to write native methods.
- 11) javaws: The javaws acts as the Web Start launcher for JNLP applications.
- 12) jhat: The jhat is a heap analysis tool.
- 13) jmc: The jmc stands as an abbreviation for Java Mission Control.

### **Features of JDK**

Here are the important features of JDK:

- It enables you to handle multiple extensions in a single catch block.
- JDK includes all features that JRE has.
- It contains development tools such as a compiler, debugger, etc.
- JDK provides the environment to develop and execute Java source code.
- It can be installed on Windows, Unix, and Mac operating systems.
- Diamond operator can be used in specifying a generic type interface instead of writing the exact one.

### **Features of JRE**

Here are the important features of JRE:

- Java Runtime Environment is a set of tools using which the JVM actually runs.
- JRE contains deployment technology, including Java Web Start and Java Plug-in.
- Developers can easily run the source code in JRE, but he/she cannot write and compile the Java program.
- It includes integration libraries like Java Database Connectivity (JDBC), Remote Method Invocation (RMI), Java Naming and Directory Interface (JNDI), and more.
- JRE has JVM and Java HotSpot virtual machine client.

### **Features of JVM**

Here are the important features of JVM:

- It enables you to run applications in a cloud environment or in your device.
- Java Virtual Machine converts byte code to the machine-specific code.
- It provides basic java functions like memory management, security, garbage collection, and more.

- JVM runs the program by using libraries and files given by Java Runtime Environment.
- JDK and JRE both contain Java Virtual Machine.
- It can execute the java program line by line hence it is also called as interpreter.
- JVM is easily customizable for example, you can allocate minimum and maximum memory to it.
- It is independent from hardware and the operating system. So, you can write a java program once and run anywhere.

### **Why use JDK?**

Here are the important reasons of using JDK:

- JDK contains tools required to write Java programs, and JRE to execute them.
- It includes a compiler, Java application launcher, Appletviewer, etc.
- Compiler converts code written in Java into byte code.
- Java application launcher opens a JRE, loads the necessary class, and executes its main method.

### **Why use JRE?**

Here are the important reasons of using JRE:

- JRE contains class libraries, JVM, and other supporting files. It does not contain any tool for Java development like a debugger, compiler, etc.
- It uses important package classes like math, swingetc, util, lang, awt, and runtime libraries.
- If you have to run Java applets, then JRE must be installed in your system.

### **Why JVM?**

Here are the important reasons of using JVM:

- JVM provides a platform-independent way of executing Java source code.
- It has numerous libraries, tools, and frameworks.
- Once you run Java program, you can run on any platform and save lots of time.
- JVM comes with JIT(Just-in-Time) compiler that converts Java source code into low-level machine language. Hence, it runs more faster as a regular application.

## **4.2 Understanding the OOPS Concepts**

### **What is OOPS?**

Object-Oriented Programming System (OOPs) is a programming concept that works on the principles of abstraction, encapsulation, inheritance, and polymorphism. It allows users to create objects they want and create methods to handle those objects. The basic concept of OOPs is to create objects, re-use them throughout the program,

and manipulate these objects to get results. It refers to a type of programming in which programmers define the data type of a data structure and the type of operations that can be applied to the data structure. It is a methodology or paradigm to design a program using classes and objects. It simplifies software development and maintenance by providing some concepts:

- Object
- Class
- Inheritance
- Polymorphism
- Abstraction
- Encapsulation

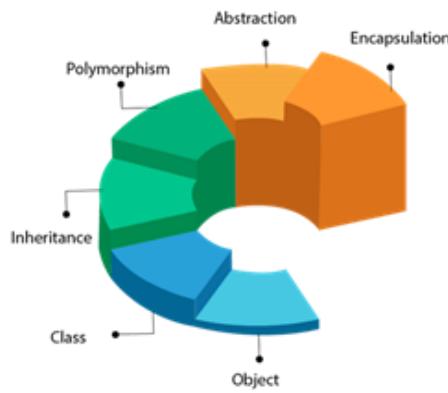


Image: OOPS

Reference: <https://www.javatpoint.com/java-oops-concepts>

## Java Object

Any entity that has state and behavior is known as an object. For example, a chair, pen, table, keyboard, bike, etc. It can be physical or logical. An Object can be defined as an instance of a class. An object contains an address and takes up some space in memory. Objects can communicate without knowing the details of each other's data or code. The only necessary thing is the type of message accepted and the type of response returned by the objects.

**Example:** A dog is an object because it has states like color, name, breed, etc. as well as behaviors like wagging the tail, barking, eating, etc.

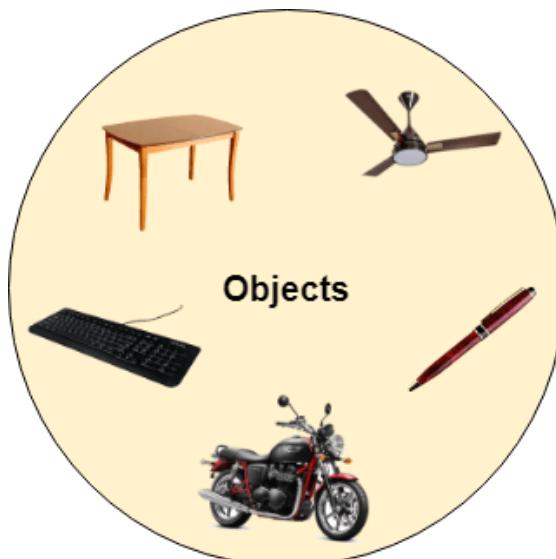


Image: Objects

Reference: <https://www.javatpoint.com/java-oops-concepts>

**Class:** Collection of objects is called class. It is a logical entity. A class can also be defined as a blueprint from which you can create an individual object. Class doesn't consume any space.

**Inheritance:** When one object acquires all the properties and behaviors of a parent object, it is known as inheritance. It provides code reusability. It is used to achieve runtime polymorphism.

**Polymorphism:** If one task is performed in different ways, it is known as polymorphism. For example: to convince the customer differently, to draw something, for example, shape, triangle, rectangle, etc. In Java, we use method overloading and method overriding to achieve polymorphism.

Another example can be to speak something; for example, a cat speaks meow, dog barks woof, etc.



Image: Polymorphism

Reference: <https://www.javatpoint.com/java-oops-concepts>

**Abstraction:** Hiding internal details and showing functionality is known as abstraction. For example, phone call, we don't know the internal processing. In Java, we use abstract class and interface to achieve abstraction.

**Encapsulation:** Binding (or wrapping) code and data together into a single unit are known as encapsulation. For example, a capsule, it is wrapped with different medicines. A java class is the example of encapsulation. Java bean is the fully encapsulated class because all the data members are private here.



Image: Encapsulation

Reference: <https://www.javatpoint.com/java-oops-concepts>

## Features of OOPS

Some features of object-oriented programming in java are:

- Emphasis is on data than procedures
- Programs are divided into objects
- Data Structures are designed to characterize objects.
- Methods operating on the data of an object are tied together in the data structure.
- Data is hidden, and external functions cannot access it.
- Objects communicate with each other through methods
- New methods and data can be easily added whenever necessary
- Follows the bottom-up approach in program design

## JAVA Installation

Check if Java Is Installed. Before installing the Java Development Kit, [check if a Java version is already installed on Windows](#). Follow the steps below:

1. Open a command prompt by typing **cmd** in the search bar and press **Enter**.
2. Run the following command:

**java -version**

## Download Java for Windows 10

Download the latest Java Development Kit installation file for Windows 10 to have the latest features and bug fixes. Using your preferred web browser, navigate to the [Oracle Java Downloads page](#).

On the *Downloads* page, click the **x64 Installer** download link under the **Windows** category. At the time of writing this article, Java version 17 is the latest long-term support Java version.

Linux	macOS	Windows
Product/file description	File size	Download
x64 Compressed Archive	170.66 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip</a> (sha256)
x64 Installer	152 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe</a> (sha256)
x64 MSI Installer	150.89 MB	<a href="https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi">https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi</a> (sha256)

**Wait for the download to complete.**

### Install Java on Windows 10

After downloading the installation file, proceed with installing Java on your Windows system.

#### Follow the steps below:

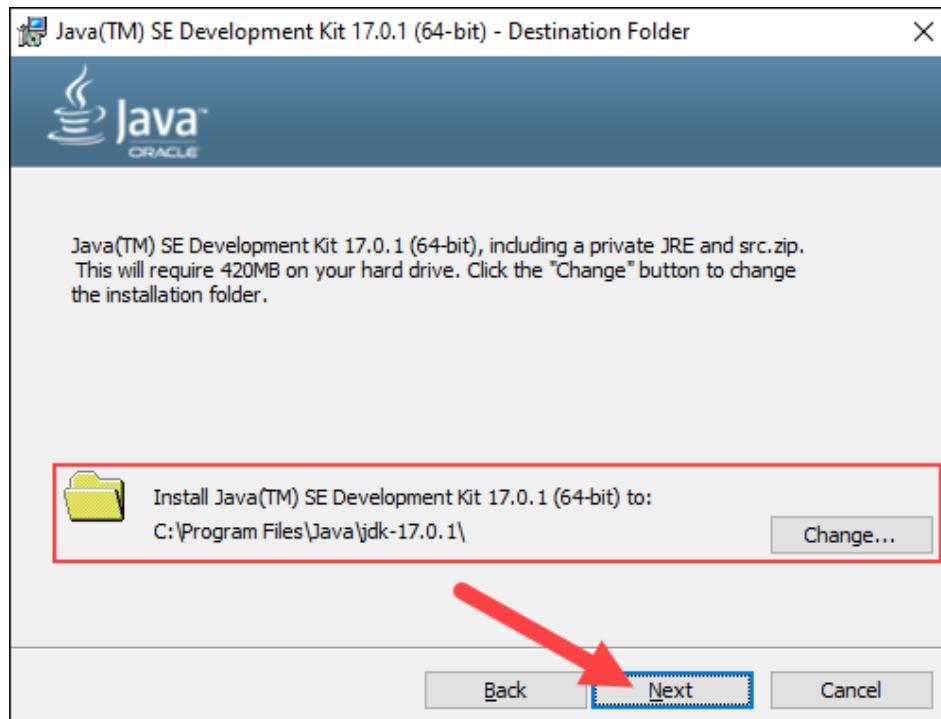
**Step 1:** Run the Downloaded File. Double-click the **downloaded file** to start the installation.

**Step 2:** Configure the Installation Wizard. After running the installation file, the installation wizard welcome screen appears.

1. Click **Next** to proceed to the next step.



2. Choose the destination folder for the Java installation files or stick to the default path. Click **Next** to proceed.



3. Wait for the wizard to finish the installation process until the Successfully Installed message appears. Click **Close** to exit the wizard.



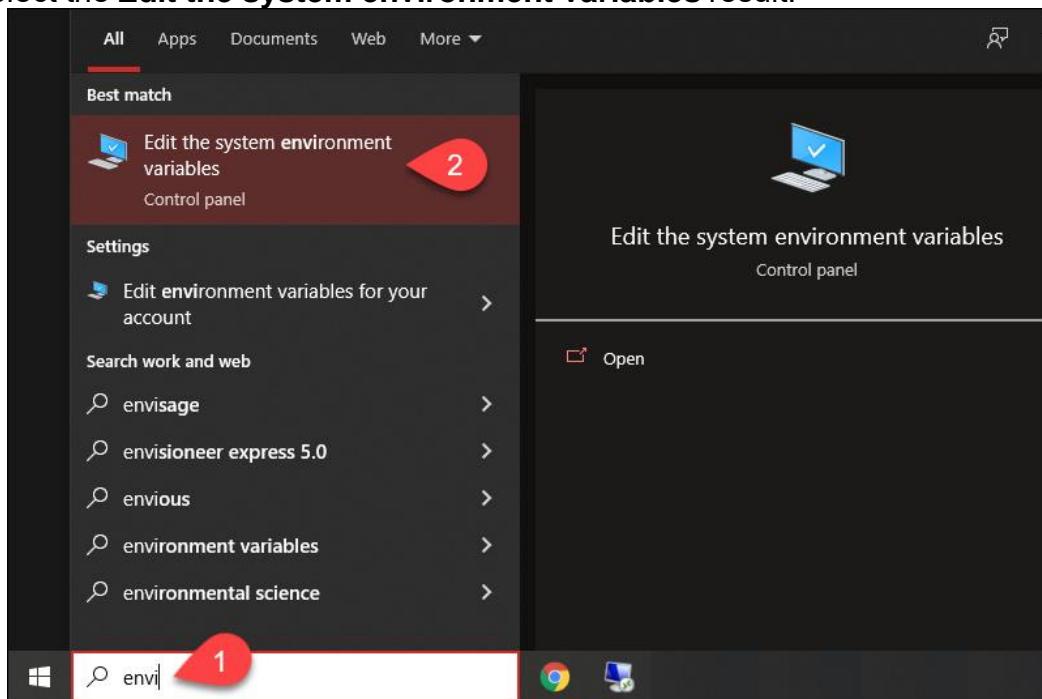
### ***Set Environmental Variables in Java***

Set Java [environment variables](#) to enable program compiling from any directory. To do so, follow the steps below:

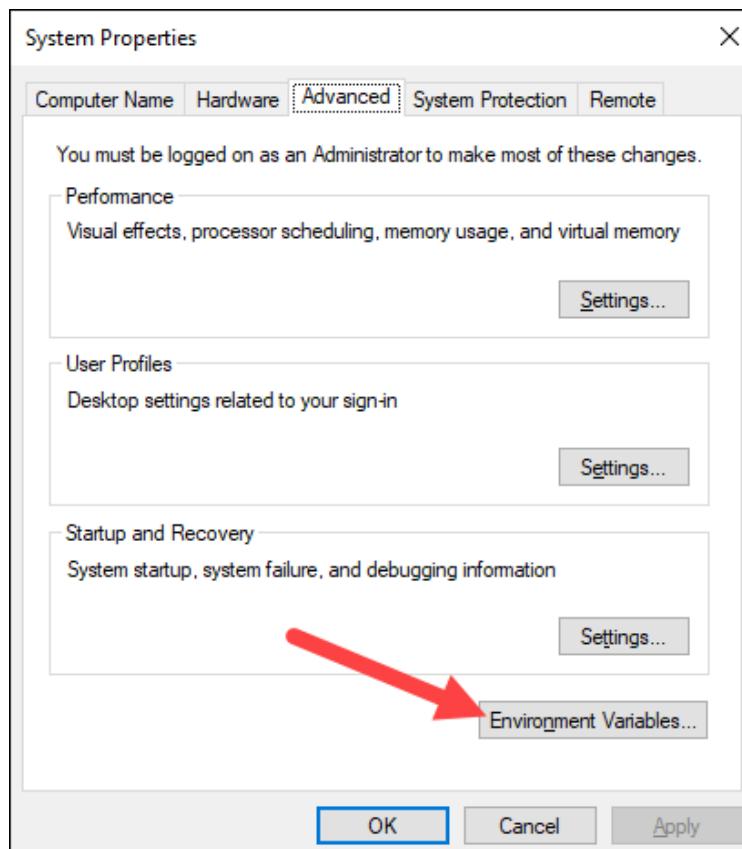
#### **Step 1: Add Java to System Variables**

1. Open the **Start** menu and search for *environment variables*.

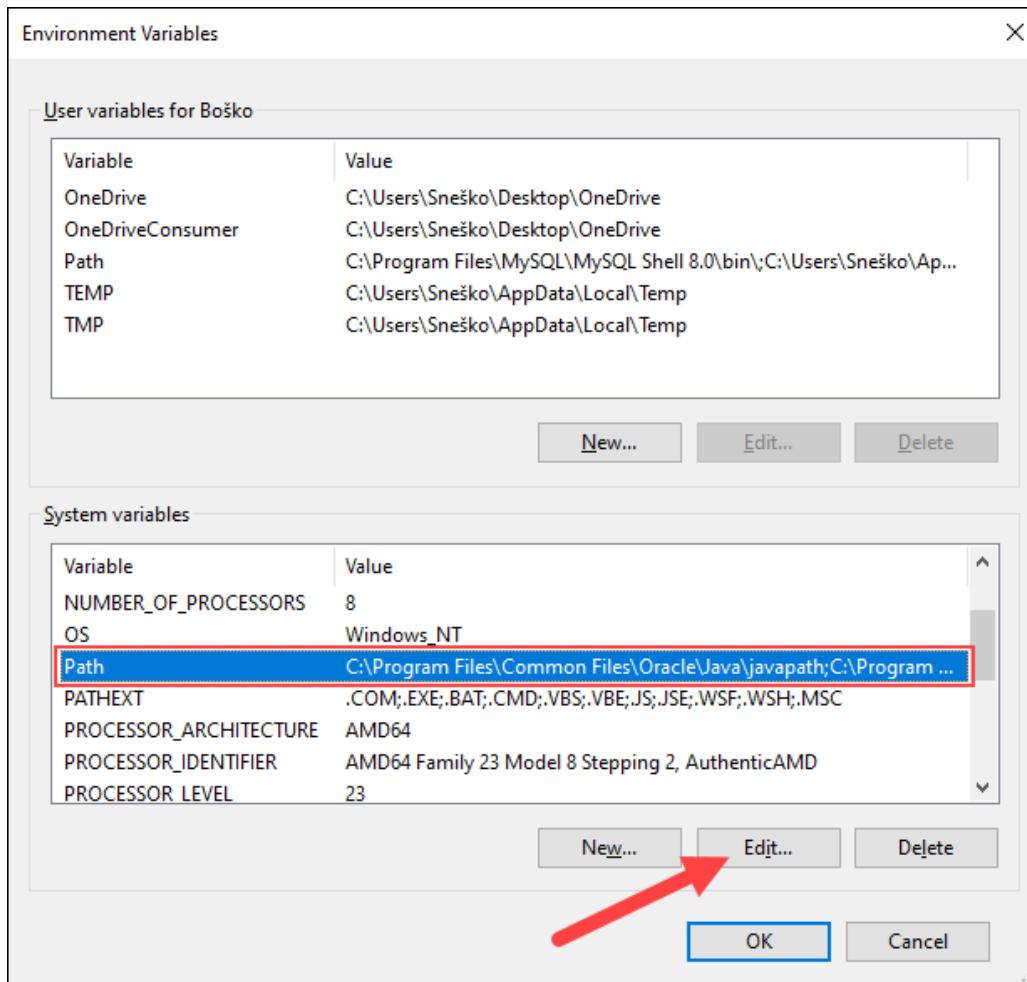
2. Select the **Edit the system environment variables** result.



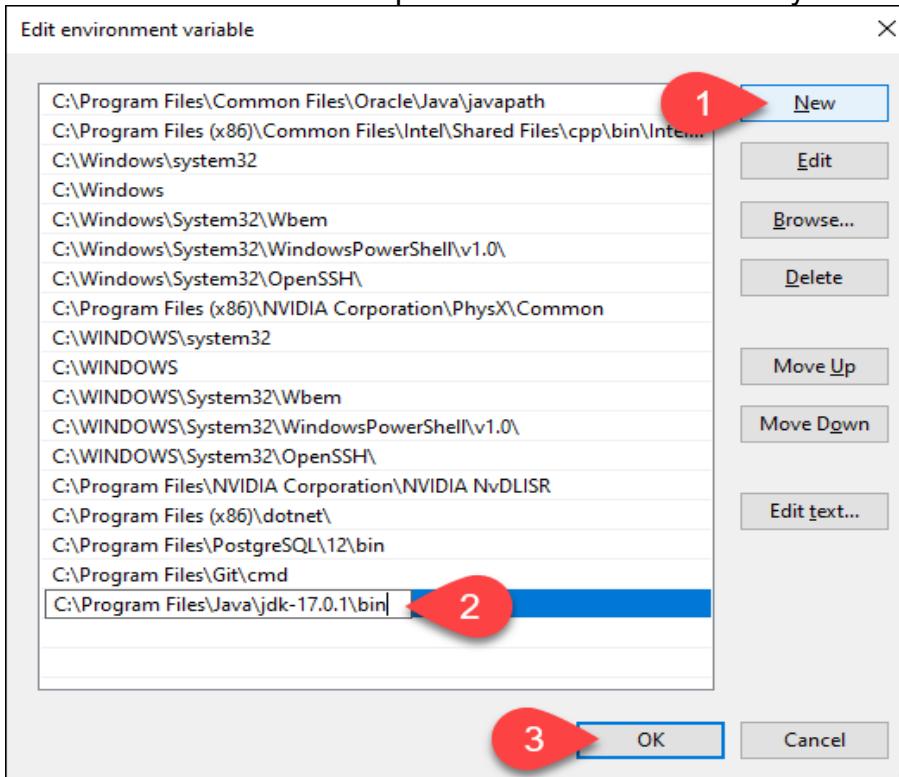
3. In the System Properties window, under the Advanced tab, click **Environment Variables...**



4. Under the System variables category, select the **Path** variable and click **Edit**:



5. Click the **New** button and enter the path to the Java bin directory:



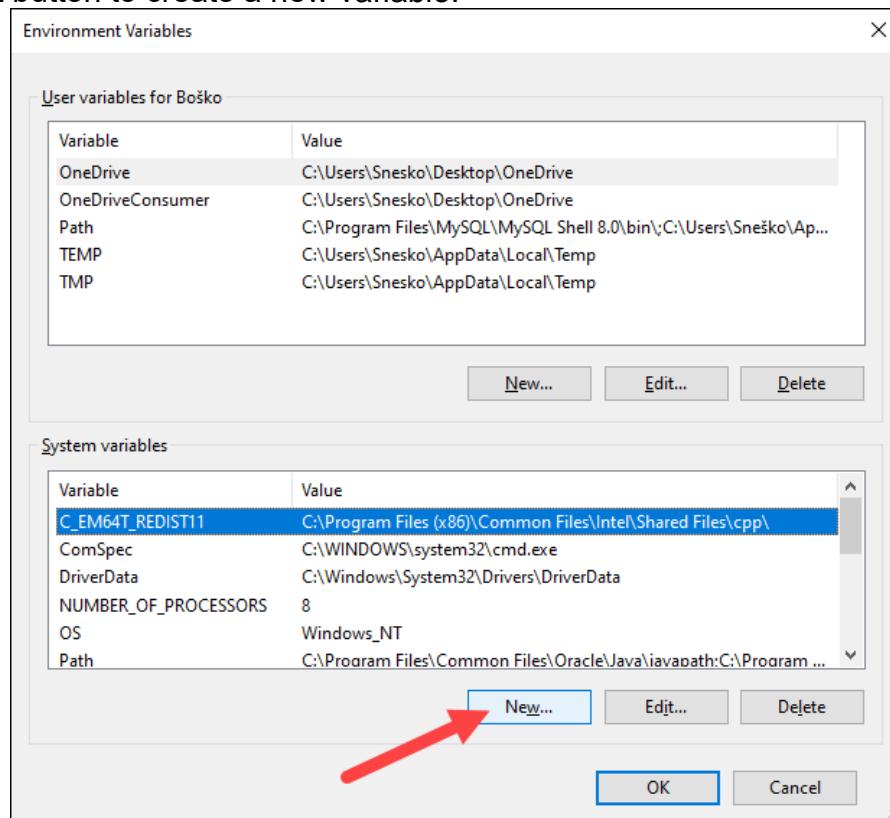
**Note:** The default path is usually **C:\Program Files\Java\jdk-17.0.1\bin**.

6. Click **OK** to save the changes and exit the variable editing window.

**Step 2:** Add **JAVA\_HOME** Variable.

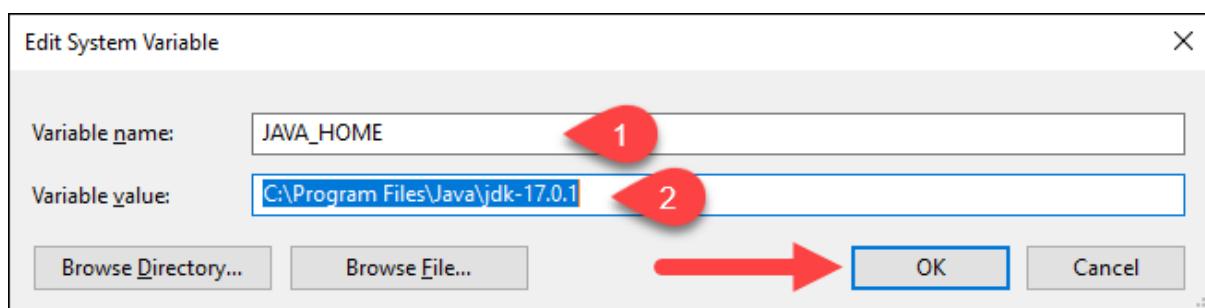
Some applications require the **JAVA\_HOME** variable. Follow the steps below to create the variable:

1. In the Environment Variables window, under the System variables category, click the **New...** button to create a new variable.



2. Name the variable as **JAVA\_HOME**.

3. In the variable value field, paste the path to your Java jdk directory and click **OK**.



4. Confirm the changes by clicking **OK** in the Environment Variables and System properties windows.

### Execution of the Program

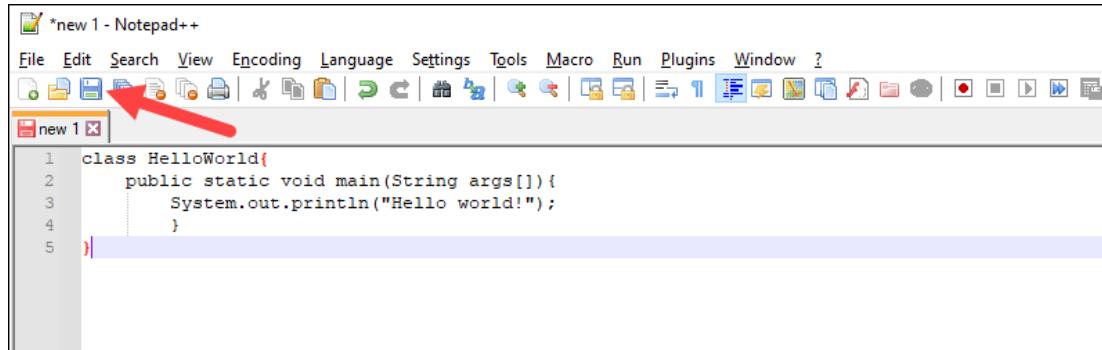
If installed correctly, the command outputs the Java version. Make sure everything works by writing a simple program and compiling it. Follow the steps below:

#### Step 1: Write a Test Java Script

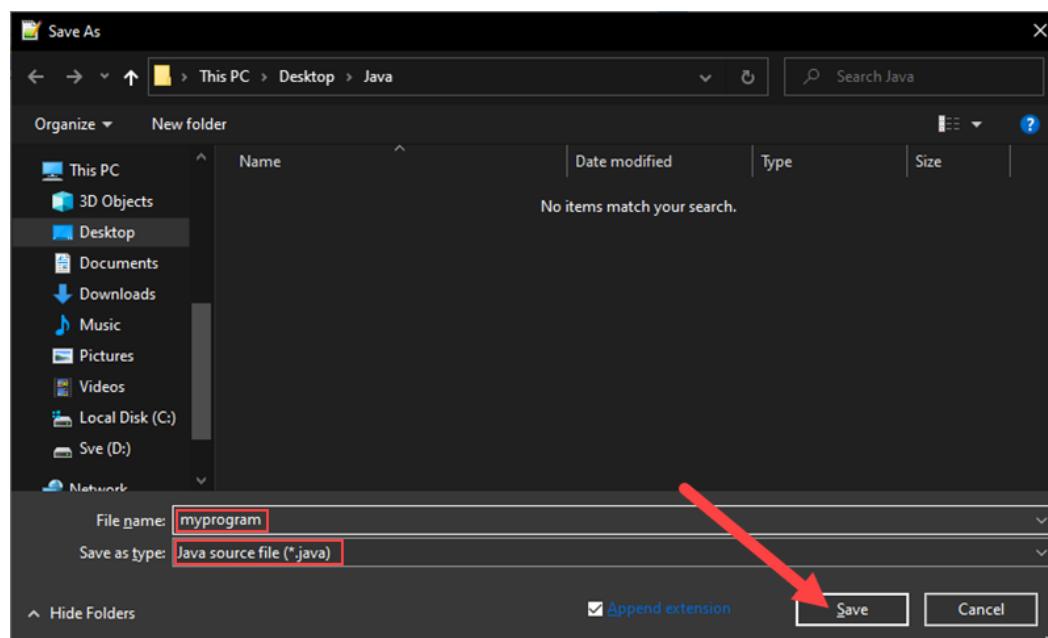
1. Open a text editor such as Notepad++ and create a new file.

2. Enter the following lines of code and click **Save**:

```
class HelloWorld{
    public static void main(String args[]){
        System.out.println("Hello world!");
    }
}
```



3. Name the file and save it as a **Java source file (\*.java)**.



**Note:** When using Notepad, select **All files** for the Save as type option and add the .java extension to the file name.

#### Step 2: Compile the Test Java Script

1. In the command prompt, change the directory to the file's location and use the following syntax to compile the program:

**Javac [filename].java**

For example:

```
C:\Users\boskom\Desktop\Java>javac myprogram.java  
C:\Users\boskom\Desktop\Java>
```

After a successful compilation, the program generates a .class file in the file directory.

2. Run the program with the following syntax:

**java [filename]**

```
C:\Users\boskom\Desktop\Java>java HelloWorld  
Hello world!
```

The output shows that the program runs correctly, displaying the **Hello world!** message.

## 4.3 Java Fundamentals

### Java Variables

A Variable is a container which clasp the value while the Java program is executed. A variable is allocated with a data type. Variable is a name of a memory location. There are three types of variables in java, these are:

- Local
- Instance
- Static

# Types of Variables

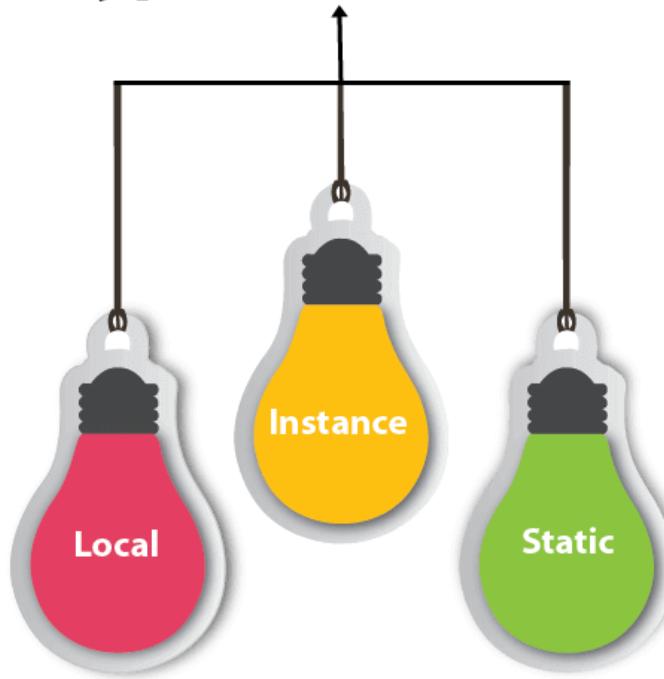


Image: Types of Variables  
Reference: <https://www.javatpoint.com/java-variables>

## Variable

A variable is the name of a reserved area allocated in memory. In other words, it is a name of the memory location. It is a combination of "vary + able" which means its value can be changed.

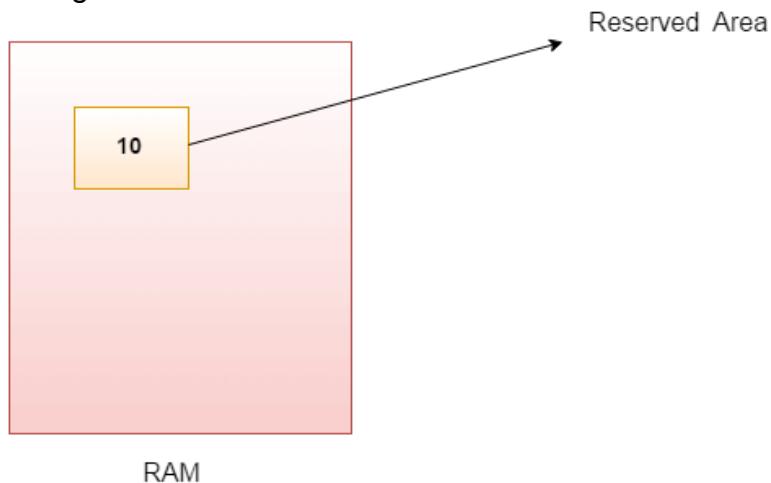


Image:Variables  
Reference: <https://www.javatpoint.com/java-variables>

### What is Local Variable?

A variable declared within the body of the strategy is named native variable. You'll use this variable solely at intervals that technique and also the different strategies within the category are not even aware that the variable exists.

### Local Variable Example:-

```
import java.io.*;
public class A {
    public void TeacherAge(){
        // local variable age
        int age = 25;
        System.out.println("a = " + age);
    }
    public static void main(String args[]){
        Teacher obj = new TeacherAge();
        obj.TeacherAge();
    }
}
```

#### Output:

a = 25

### What is Instance Variable?

A variable declared within the category however outside the body of the tactic, is named instance variable. It's not declared as static. It is referred to as instance variable as a result of its worth is instance specific and isn't shared among instances.

- As instance variables are declared within a class, these variables are created when an object of the class is created as well as destroyed when the object is destroyed.
- Unlike local variables, you may use access specifiers for instance variables. If you don't specify any access specifier then the default access specifier will be used.
- Initialization of Instance Variable isn't Mandatory. By default, its value is 0
- Instance Variable can be accessed only by creating its object.

### Instance Variable Example: -

```
import java.io.*;
class Marks {
    int banMarks;
}
class MarksDemo {
    public static void main(String args[]){
        // first object
        Marks obj1 = new Marks();
        obj1.engMarks = 50;
        System.out.println("Marks = " + obj1.banMarks);
    }
}
```

#### Output:

Marks = 50

## What is Static Variable?

A variable which is declared as static is termed static variable. It cannot be native. You can produce a single copy of static variable and share among all the instances of the category. Memory allocation for static variable happens just once when the category is loaded in the memory.

- These variables are declared same as instance variables, the main difference is that static variables are declared using the static keyword within a class outside any method, constructor, or a block.
- Unlike instance variables, you can only have one copy of a static variable per class irrespective of how many objects you create.
- Static variables are created at the beginning of program execution and destroyed automatically when its execution ends.
- Initialization is not Mandatory for Static Variable. By default its value is 0.
- If you try to access the static variable as like as Instance variable (through an object), the compiler will show the warning message and it will not halt the program. The compiler will replace the object name to the class name automatically.
- If you try to access the static variable without the class name, Compiler will automatically add the class name.

To access static variables, you don't need to create an object of that class, you can simply access the variable as `class_name.variable_name;`.

### **Static Variable Example**

```
import java.io.*;  
class Emp {  
    public static double salary;  
    public static String name = "Tuhin";  
}  
  
public class EmpDemo {  
    public static void main(String args[]){  
        Emp.salary = 1000;  
        System.out.println(Emp.name);  
        System.out.println(Emp.salary);  
    }  
}
```

#### **Output:**

Hari  
1000

## **Data Types**

Data types indicate the specific sizes and values that can be stored in the flexible. There are two variety of data variety in Java programming:

- **Primitive data types:** The primitive data types include boolean, char, byte, short, int, long, float and double.

- **Non-primitive data types:** The non-primitive data types include Classes, Interfaces, and Arrays.

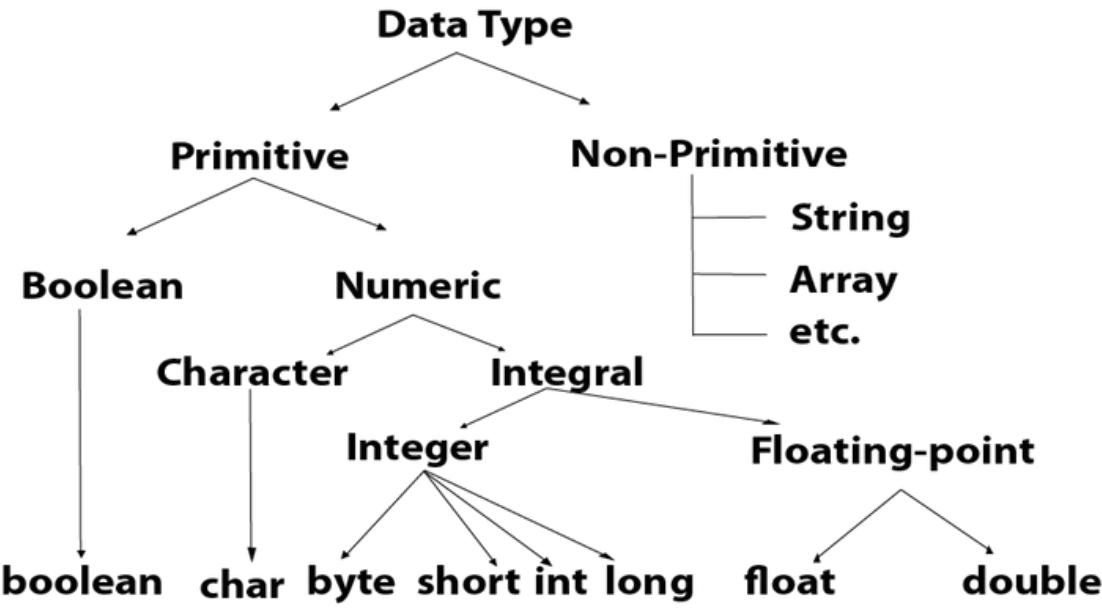


Image: Data Types

Reference: <https://www.javatpoint.com/java-data-types>

## What are Primitive Data Types In Java?

In Java language, untrained data variety are the building blocks of data consumption. These are the most vital data variety available in Java language. Java is a statically-typed programming language. It means, all flexible must be announced before its use. That is why we need to reveal flexible variety and name. There are 8 variety of untrained data variety:

- boolean
- byte
- char
- short
- int
- long
- float
- double

Data Type	Default Value	Default size
boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte
short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

### **What is Boolean Data Type?**

The Boolean data variety is used to store only two probable values: true and false. This data variety is used for elementary flags that track true/false conditions. The Boolean data variety indicate one bit of information, but its "size" can't be prominent precisely.

#### **Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        boolean a = true;
        boolean b = false;
        System.out.println(a);
        System.out.println(b);
    }
}
```

#### **Output:**

true  
false

### **What is Byte Data Type?**

The byte data variety is an example of untrained data type. It is an 8-bit signed two's accompaniment integer. Its amount- variety lies between -128 to 127 (comprehensive). Its minimum amount is -128 and maximum amount is 127.

The byte data variety is used to recover memory in big arrays where the memory savings is most required. It recovers capacity because a byte is 4 times smaller than an integral. It can also be preloved in place of "int" data variety.

#### **Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        byte a = 2;
        byte b = -4;
        System.out.println(a);
        System.out.println(b);
    }
}
```

#### **Output:**

2  
-4

### **What is Short Data Type?**

The short data variety is a 16-bit signed two's complement integer. Its amount variety lies between -32,768 to 32,767 (comprehensive). Its minimum amount is -32,768 and maximum amount is 32,767. Its default amount is 0. The short data variety can also be used to recover memory just like byte data variety. A, abrupt data variety is 2 times smaller than an integer.

**Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        short a = 2000;
        short b = -2000;
        System.out.println(a);
        System.out.println(b);
    }
}
```

**Output:**

```
2000
-2000
```

**What is Int Data Type?**

The int data type is a 32-bit signed two's complement integer. Its amount types lies between - 2,147,483,648 ( $-2^{31}$ ) to 2,147,483,647 ( $2^{31-1}$ ) (inclusive). Its minimum amount is - 2,147,483,648 and maximum amount is 2,147,483,647. The int data types is generally preloved as a default data type for integral amount unless if there is no problem about memory.

**Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        int a = 200000;
        int b = -300000;
        System.out.println(a);
        System.out.println(b);
    }
}
```

**Output:**

```
200000
-300000
```

**What is Long Data Type?**

The long data types is a 64-bit two's complement integer. Its amount type lies between -9,223,372,036,854,775,808( $-2^{63}$ ) to 9,223,372,036,854,775,807( $2^{63-1}$ )

1)(comprehensive). Its minimum amount is - 9,223,372,036,854,775,808 and maximum amount is 9,223,372,036,854,775,807. Its default amount is 0. The long data variety is used when you need a range of values more than those distribute by int.

**Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        long a = 200000L;
        long b = -300000L;
        System.out.println(a);
        System.out.println(b);
    }
}
```

**Output:**

200000L  
-300000L

**What is Float Data?**

The float data types is a single-precision 32-bit IEEE 754 floating point. Its amount range is unlimited. It is recommended to use a float (instead of double) if you need to recover memory in large arrays of floating point numbers. The float data types should never be preloved for precise values, such as currency. Its default amount is 0.0F.

**Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        float a = -2.00;
        float b = 2.00;
        System.out.println(a);
        System.out.println(b);
    }
}
```

**Output:**

-2.00  
2.00

**What is Double Data Type?**

The double data types is a double-precision 64-bit IEEE 754 floating point. Its amount range is endless. The double data variety is generally preloved for decimal amount just like float. The double data type also should never be preloved for precise amount, such as currency. Its default amount is 0.0d.

**Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        double a = -2.00;
        double b = 2.00;
        System.out.println(a);
        System.out.println(b);
    }
}
```

**Output:**

-2.00  
2.00

***What is Char Data Type?***

The char data type is a single 16-bit Unicode character. Its amount type lies between '\u0000' (or 0) to '\uffff' (or 65,535 inclusive). The char data is type preloved to store characters.

**Example:**

```
import java.io.*;
class Simple{
    public static void main(String args[]){
        char a = 'D';
        char b = 'C';
        System.out.println(a);
        System.out.println(b);
    }
}
```

**Output:**

D C

## Java Keywords

Java keywords are also known as reserved words. Keywords are particular words which acts as a key to a code and these are predefined words by Java so it cannot be used as variable or object name.

- **abstract:** Java abstract keyword is employed to declare abstract category. Abstract category will offer the implementation of interface. It will have abstract and non-abstract strategies.
- **boolean:** Java boolean keyword is used to declare a variable as a boolean type. It can only hold True and False values only.

- **break:** Java break keyword is employed to interrupt loop or switch statement. It breaks this flow of the program at specified condition.
- **byte:** Java computer memory unit keyword is employed to declare a variable that may hold associate 8-bit information values.
- **base:** Java case keyword is employed to with the switch statements to mark blocks of text.
- **catch:** Java catch keyword is employed to catch the exceptions generated by attempt statements. It should be used once the attempt block solely.
- **char:** Java char keyword is employed to declare a variable that may hold unsigned 16-bit Unicode characters
- **class:** Java category keyword is employed to declare a category.
- **continue:** Java continue keyword is employed to continue the loop. It resumes this flow of the program and skips the remaining code at the desired condition.
- **default:** Java default keyword is employed to specify the default block of code in an exceedingly switch statement.
- **do:** Java do keyword is employed au fait statement to declare a loop. It will ingeminate an area of the program many times.
- **double:** Java double keyword is employed to declare a variable which will hold a 64-bit floating-point numbers.
- **else:** Java else keyword is employed to point the choicebranches in associate degree if statement.
- **enum:** Java enum keyword is employed to outline a hard and fast set of constants. Enum constructors square measure perpetually personal or default.
- **extends:** Java extends keyword is employed to point that category|a category} springs from another class or interface.
- **final:** Java final keyword is employed to point that a variable holds a continuing worth. it's applied with a variable. it's accustomed limit the user.
- **finally:** Java finally keyword indicates a block of code in a very try-catch structure. This block is usually dead whether or not exception is handled or not.
- **float:** JJava float keyword is employed to declare a variable which will hold a 32-bit number.
- **for:** Java for keyword is employed to start out a for loop. it's accustomed execute a collection of instructions or functions repeatedly once some conditions become true. If the amount of iteration is fastened, it's counseled to use for loop.
- **if:** Java if keyword tests the condition. It brings of the if block if condition is true.
- **implements:** Java implements keyword is employed to implement associate interface.
- **import:** Java import keyword makes categories and interfaces on the market and accessible to this ASCII text file.
- **instanceof:** Java instanceof keyword is employed to check whether or not the thing is associate instance of the desired category or implements associate interface.
- **int:** Java int keyword is employed to declare a variable which will hold a 32-bit signed whole number.
- **interface:** Java interface keyword is employed to declare associate interface. It will have solely abstract strategies.
- **long:** Java long keyword is employed to declare a variable which will hold a 64-bit whole number.

- **native:** Java native keyword is employed to specify that a technique is enforced in native code exploitation JNI (Java Native Interface).
- **new:** Java new keyword is employed to form new objects.
- **null:** Java null keyword is employed to point that a reference doesn't ask something. It removes the rubbish worth.
- **package:** Java package keyword is employed to declare a Java package that has the categories.
- **private:** Java non-public keyword is associate access modifier. it's accustomed indicate that a technique or variable is also accessed solely within the category within which it's declared.
- **protected:** Java protected keyword is associate access modifier. It will be accessible among package and out of doors the package however through inheritance solely. It cannot be applied on the category.
- **return:** Java public keyword is associate access modifier. it's accustomed indicate that associate item is accessible anyplace. it's the widest scope among all different modifiers.
- **short:** Java short keyword is employed to declare a variable which will hold a 16-bit whole number.
- **static:** Java static keyword is employed to point that a variable or technique may be a category technique. The static keyword in Java is employed for memory management principally.
- **strictfp:** Java strictfp is employed to limit the floating-point calculations to confirm movability.
- **super:** Java super keyword may be a reference variable that's accustomed refer parent category object. It will be accustomed invoke immediate parent category technique.
- **switch:** The Java switch keyword contains a switch statement that executes code supported take a look at worth. The switch statement tests the impartiality of a variable against multiple values.
- **synchronized:** Java synchroinal keyword is employed to specify the important sections or strategies in multithreaded code.
- **this:** Java this keyword will be accustomed refer this object in an exceedingly technique or creator.
- **throw:** Java throw keyword is employed to expressly throw associate exception. The throw keyword is principally accustomed throw custom exception. it's followed by associate instance.
- **throws:** The Java throws keyword is employed to declare associate exception. Checked exception will be generated with throws.
- **transient:** Java transient keyword is employed in publishing. If you outline any knowledge member as transient, it'll not be serialized.
- **Try:** Java attempt keyword is employed to begin a block of code which will be tested for exceptions. The attempt block should be followed by either catch or finally block.
- **Void:** Java void keyword is employed to specify that a technique doesn't have a comeback worth.
- **Volatile:** Java volatile keyword is employed to point that a variable could modification asynchronously.
- **While:** Java whereas keyword is employed to begin a jiffy loop. This loop iterates a neighborhood of the program many times. If the amount of iteration isn't fastened, it's counseled to use whereas loop.

## Java Operators

Operator in Java is a symbol or a sign which is used for performing operations. For example: -, +, /, \* etc. In Java there are many different types of operators which are given below: -

- Assignment Operator
- Arithmetic Operator
- Relational Operator
- Logical Operator
- Bitwise Operator
- Unary Operator
- Ternary Operator

### *Assignment Operators*

Assignment = operators are used to assigns the value on its left variable to the right-side variable or data. Assignment operators in Java are given below:

Operator	Example	Equal To	Explain
=	a = b;	a = b;	It takes the value of b and assigns into a.
+=	a += b;	a = a + b;	It takes the value of a + b and assigns into a.
-=	a -= b;	a = a - b;	It takes the value of a - b and assigns into a.
*=	a *= b;	a = a * b;	It takes the value of a * b and assigns into a.
/=	a /= b;	a = a / b;	It takes the value of a / b and assigns into a.
%=	a %= b;	a = a % b;	It takes the value of a % b and assigns into a.

### **Example:**

```
import java.io.*;
Class Main{
    Public static void main(String args[]){
        int a = 12, b = 5;
        int c = 12, d = 5;

        a += b;
        c %= d;

        System.out.println(a);
        System.out.println(c);
    }
}
```

### **Output**

17  
2

## Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations on two or more variables. Arithmetic operators in Java are given below:

Operator	Operation
+	It adds two or more variables / values one with others.
-	It subtracts two or more variables / values on from another
*	It multiplies two or more variables / values one with others.
/	It division two or more variables / values one by another. [N:B If you use the division operator with two integers, then the result will also be an integer. And, if atleast one of the operands is a floating point number, you will get the result will also be in floating point.]
%	Modulo Operation is basically keep the remainder.

### Example:

```
import java.io.*;
Class Main{
    Public static void main(String args[]){
        int a = 12, b = 5;
        float c = 12, d = 5;
        int Addition = a + b; // addition operator
        int Substract = a - b; // subtraction operator
        int Multiplication = a * b; // multiplication operator
        float Division = c / d; // division operator
        int Modulo = a % b; // modulo operator

        System.out.println("a + b = "+ Addition);
        System.out.println("a - b = "+ Substract);
        System.out.println("a * b = "+ Multiplication);
        System.out.println("c / d = "+ Division);
        System.out.println("a % b = "+ Modulo);
    }
}
```

### Output

```
a + b = 17
a - b = 5
a * b = 60
c / d = 2.4
a % b = 2
```

### Relational Operators

Relational operators are used to check the relationship or comparison between two operands or variables. Relational operators in Java are given below:

Operator	Description	Example
==	Is Equal To	5 == 5 will return true.
!=	Is Not Equal To	5 != 4 will return true.
>	Greater Than	5 > 4 will return true.

<b>&lt;</b>	<b>Less Than</b>	5 < 4 will return false.
<b>&gt;=</b>	<b>Greater Than Equal To</b>	5 >= 4 will return true.
<b>&lt;=</b>	<b>Less Than Equal To</b>	5 <= 4 will return false.

### Example:

```
import java.io.*;
Class Main{
    Public static void main(String args[]){
        int a = 5, b = 4;
        System.out.println(a == b);
        System.out.println(a >= b);
    }
}
```

### Output

false  
true

### *Logical Operators*

Logical operators are used to check 2 or more expression is true or not. Logical operators in Java are given below:

Operator	Example	Description
<b>&amp;&amp; (Logical AND)</b>	<b>5 &gt; 4 &amp;&amp; 9 &gt; 3</b>	If both side expression of && (AND) is true, then it returns true.
<b>   (Logical OR)</b>	<b>5 &gt; 4    9 &lt; 3</b>	If any one side expression of    (OR) is true, then it returns true.
<b>! (Logical NOT)</b>	<b>!expression</b>	If expression is true, then it returns false and if expression is false then it returns true and vice versa.

### Example:

```
import java.io.*;
Class Main{
    Public static void main(String args[]){
        int a = 5, b = 4, c = 1;
        System.out.println(a > b && b > c);
        System.out.println(a > b || b < c);
        System.out.println( !(a > b) );
    }
}
```

### Output

true  
true  
false

## Bitwise Operators

Bitwise operators in Java are used to perform operations on every bits. Bitwise operators in java are:

Operator	Description
&	Bitwise AND
	Bitwise OR
^	Bitwise XOR
~	Bitwise Complement
<<	Bitwise Left Shift
>>	Bitwise Right Shift
>>>	Unsigned Right Shift

## Unary Operators

Unary operators are worked with one variable or operands. Unary operators in Java are given below:

Operator	Description	Example
++	Increment Operator	If a = 4 then a++ will be 5.
--	Decrement Operator	If a = 4 then a-- will be 3.

## Example:

```
import java.io.*;
Class Main{
    Public static void main(String args[]){
        int a = 5, b = 4;
        System.out.println(a++);
        System.out.println(b--);
    }
}
```

## Output

```
6
3
```

## Ternary Operators

Ternary operators is basically a conditional operator and sign is "?". For an example:

```
variable = expression? value1: value2;
```

In the above example if expression is true then value1 will be stored in variable and if expression is false then value2 will be stored in variable.

**Example:**

```
import java.io.*;
Class Main{
    Public static void main(String args[]){
        int a = 5, b;
        b = a < 3 ? 2 : 4
        System.out.println(b);
    }
}
```

**Output**

4

***Input and Output*****Java Output**

In Java, you can simply use

**System.out.println(); or**  
**System.out.print(); or**  
**System.out.printf();**

to send output to standard output (screen).

Here, System is a class out is a public static field: it accepts output data.

**Example:**

```
class Program{
    public static void main(String[] args) {
        System.out.println("Java programming is interesting.");
    }
}
```

**Output:**

Java programming is interesting. Here, we have used the println() method to display the string.

**Difference between println(), print() and printf()**

- **print()** - It prints string inside the quotes.
- **println()** - It prints string inside the quotes similar like print() method. Then the cursor moves to the beginning of the next line.
- **printf()** - It provides string formatting (similar to printf in C/C++ programming).

**Example: print() and println()**

```
class Output {
    public static void main(String[] args) {
```

```
    System.out.println("1. println ");
    System.out.println("2. println ");
    System.out.print("1. print ");
    System.out.print("2. print");
}
}
```

**Output:**

1. println  
2. println  
1. print 2. Print

**Java Input**

Java provides different ways to get input from the user. However, in this tutorial, you will learn to get input from user using the object of Scanner class.

In order to use the object of Scanner, we need to import java.util.Scanner package.

```
import java.util.Scanner;
```

To learn more about importing packages in Java, visit Java Import Packages. Then, we need to create an object of the Scanner class. We can use the object to take input from the user.

```
// create an object of Scanner
Scanner input = new Scanner(System.in);
// take input from the user
int number = input.nextInt();
```

**Example: Get Integer Input From the User**

```
import java.util.Scanner;
class Input {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int number = input.nextInt();
        System.out.println("You entered " + number);
        // closing the scanner object
        input.close();
    }
}
```

**Output:**

Enter an integer: 23 You entered 23

In the above example, we have created an object named input of the Scanner class. We then call the nextInt() method of the Scanner class to get an integer input from the user. Similarly, we can use nextLong(), nextFloat(), nextDouble(), and next() methods to get long, float, double, and string input respectively from the user.

**Note:** We have used the close () method to close the object. It is recommended to close the scanner object once the input is taken.

### Example: Get float, double and String Input

```
import java.util.Scanner;
class Input {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        // Getting float input
        System.out.print("Enter float: ");
        float myFloat = input.nextFloat();
        System.out.println("Float entered = " + myFloat);
        // Getting double input
        System.out.print("Enter double: ");
        double myDouble = input.nextDouble();
        System.out.println("Double entered = " + myDouble);
        // Getting String input
        System.out.print("Enter text: ");
        String myString = input.next();
        System.out.println("Text entered = " + myString); }}
```

#### Output:

```
Enter float: 2.343
Float entered = 2.343
Enter double: -23.4
Double entered = -23.4
Enter text: Hey!
Text entered = Hey!
```

## Java Expressions & Blocks

### Java Expression

An expression is a construct which is made up of literals, variables, method calls and operators following the syntax of Java. Every expression consists of at least one operator and an operand. Operand can be either a literal, variable or a method invocation.

Following are some of the examples for expressions in Java:

```
int a = 10; //Assignment expression
System.out.println("Value = "+x);
int result = a + 10; //Assignment exp
if(val1 <= val2) //Boolean expression
b = a++; //Assignment exp
```

### How expressions are evaluated?

It is common for an expression to have more than one operator. For example, consider the below example:

$$(20 * 5) + (10 / 2) - (3 * 10)$$

So, how is the above expression evaluated? Expression evaluation in Java is based upon the following concepts:

- Type promotion rules
- Operator precedence
- Associativity rules

### *Operator precedence*

All the operators in Java are divided into several groups and are assigned a precedence level. The operator precedence chart for the operators in Java is shown below:

Highest Precedence	Operators
	<code>++ (postfix), -- (postfix)</code>
	<code>++ (prefix), -- (prefix), ~, !, +(unary), -(unary), (type-cast)</code>
	<code>*, /, %</code>
	<code>+, -</code>
	<code>&gt;&gt;, &gt;&gt;&gt;, &lt;&lt;</code>
	<code>&gt;, &gt;=, &lt;, &lt;=, instanceof</code>
	<code>==, !=</code>
	<code>&amp;</code>
	<code>^</code>
	<code> </code>
	<code>&amp;&amp;</code>
	<code>  </code>
	<code>?:</code>
Lowest Precedence	<code>=, op=</code>

Image: Data Types

Reference: <https://www.javatpoint.com/java-data-types>

Now let's consider the following expression:

**10 – 2 \* 5**

One will evaluate the above expression normally as, 10-2 gives 8 and then 8\*5 gives 40. But Java evaluates the above expression differently. Based on the operator precedence chart shown above, \* has higher precedence than -. So, 2\*5 is evaluated first which gives 10 and then 10 – 10 is evaluated which gives 0.

### *Associativity Rules*

When an expression contains operators from the same group, associativity rules are applied to determine which operation should be performed first. The associativity rules of Java are shown below:

Operator Group	Associativity	Type of Operation
<code>! ~ ++ -- + -</code>	right-to-left	unary
<code>* / %</code>	left-to-right	multiplicative
<code>+ -</code>	left-to-right	additive
<code>&lt;&lt; &gt;&gt; &gt;&gt;&gt;</code>	left-to-right	bitwise
<code>&lt; &lt;= &gt; &gt;=</code>	left-to-right	relational
<code>== !=</code>	left-to-right	relational
<code>&amp;</code>	left-to-right	bitwise
<code>^</code>	left-to-right	bitwise
<code> </code>	left-to-right	bitwise
<code>&amp;&amp;</code>	left-to-right	boolean
<code>  </code>	left-to-right	boolean
<code>?:</code>	right-to-left	conditional
<code>= += -= *= /= %= &amp;=</code> <code>^=  = &lt;&lt;= &gt;&gt;= &gt;&gt;&gt;=</code>	right-to-left	assignment
<code>,</code>	left-to-right	comma

Image: Data Types

Reference: <https://www.javatpoint.com/java-data-types>

Now, let's consider the following expression:

**10-6+2**

In the above expression, the operators `+` and `-` both belong to the same group in the operator precedence chart. So, we have to check the associativity rules for evaluating the above expression. Associativity rule for `+` and `-` group is left-to-right i.e., evaluate the expression from left to right. So, `10-6` is evaluated to 4 and then `4+2` is evaluated to 6.

### ***Use of parenthesis in expressions***

Let's look at our original expression example:

**`(20 * 5) + (10 / 2) – (3 * 10)`**

You might think that, what is the need of parenthesis (and) in the above expression. The reason I had included them is, parenthesis have the highest priority (precedence) over all the operators in Java.

So, in the above expression, `(20*5)` is evaluated to 100, `(10/2)` is evaluated to 5 and `(3*10)` is evaluated to 30. Now, our intermediate expression looks like:

**`100 + 5 – 30`**

Now, we can apply the associativity rules and evaluate the expression. The final answer for the above expression is 75. There is another popular use of parenthesis. We will use them in print statements. For example, consider the following piece of code:

```
int a=10, b=20;
System.out.println("Sum of a and b is: "+a+b);
```

One might think that the above code will produce the output: "Sum of a and b is: 30". The real output will be:

```
Sum of a and b is: 1020
int a=10, b=20;
System.out.println("Sum of a and b is: "+(a+b));
```

Now `(a+b)` is evaluated first and then concatenated to the rest.

### **Java Blocks**

A block is a group of statements (zero or more) that is enclosed in curly braces `{ }`. For example,

```
class Main {  
    public static void main(String[] args) {  
  
        String band = "Beatles";  
  
        if (band == "Beatles") { // start of block  
            System.out.print("Hey ");  
            System.out.print("Jude!");  
        } // end of block  
    }  
}
```

#### **Output:**

Hey Jude!

In the above example, we have a block if `{....}`.

Here, inside the block we have two statements:

```
System.out.print("Hey ");  
System.out.print("Jude!");
```

However, a block may not have any statements. Consider the following examples,

```
class Main {  
    public static void main(String[] args) {  
        if (10 > 5) { // start of block  
        } // end of block  
    }  
}
```

This is a valid Java program. Here, we have a block if `{...}`. However, there is no any statement inside this block.

```
class AssignmentOperator {  
    public static void main(String[] args) { // start of block } // end of block  
}
```

Here, we have block `public static void main() {...}`. However, similar to the above example, this block does not have any statement.

### **Comment**

Comments can be used to explain Java code, and to make it more readable. It can also be used to prevent execution when testing alternative code.

#### **Single-line Comments**

Single-line comments start with two forward slashes `//`. Any text between `//` and the end of the line is ignored by Java (will not be executed).

This example uses a single-line comment before a line of code: Example

```
// This is a comment
System.out.println("Hello World");
System.out.println("Hello World"); // This is a comment
```

### ***Java Multi-line Comments***

Multi-line comments start with /\* and ends with \*/. Any text between /\* and \*/ will be ignored by Java.

This example uses a multi-line comment (a comment block) to explain the code:

```
public class Main {
    public static void main(String[] args) {
        /* The code below will print the words Hello World
         * to the screen, and it is amazing */
        System.out.println("Hello World");
    }
}
```

## **4.4 Java Control Statements**

### ***If Else Statement***

The java “If statement” is employed to check the condition. It is a boolean condition checker: true or false. There are different types of if statement in Java.

- if statement
- if else statement
- if else if ladder
- nested if statement

### **If Statement in Java:**

The Java “If statement” tests the condition. It executes the “If block” if the condition is true.

#### **Syntax:**

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

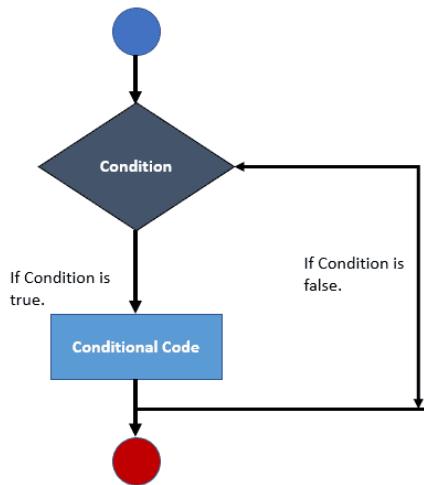


Image: if statement

## Example

```
import java.io.*;
public class Main {
    public static void main(String args[]){
        int age = 20;
        if(age > 18) {
            System.out.println("Over 18");
        }
    }
}
```

## Output:

Over 18

## If Else Statement in Java:

The Java “If else statement” also tests the condition. It executes the “If block” if the condition is true otherwise else block is executed.

### Syntax:

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

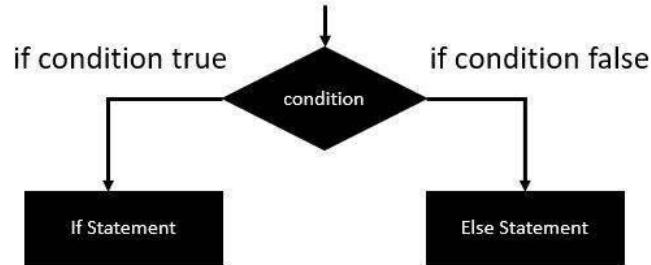


Image: if-else statement

**Example:**

```
import java.io.*;
public class Main {
    public static void main(String args[]){
        int age = 16;
        if(age > 18) {
            System.out.println("Over 18");
        }
        else{
            System.out.println("Under 18");
        }
    }
}
```

**Output:**

Under 18

**If Else If Ladder Statement in Java:**

The “If else if ladder statement” executes one condition from multiple statements.

**Syntax:**

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

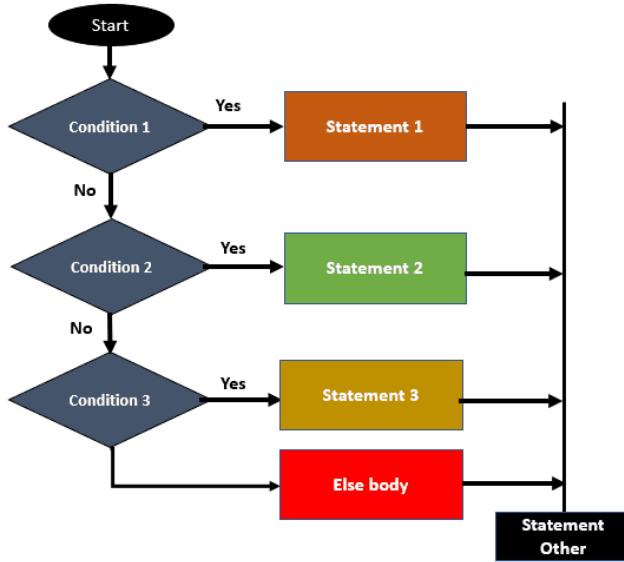


Image:Else\_if ladder statement

**Example:**

```

import java.io.*;
public class Main {
    public static void main(String args[]){
        int age = 18;
        if(age > 18) {
            System.out.println("Over 18");
        }
        if(age == 18) {
            System.out.println("Just 18");
        }
        else{
            System.out.println("Under 18");
        }
    }
}
  
```

**Output:**

Just 18

***Nested if statement Statement in Java:***

The “Nested if statement” represents the “If block” within another “If block.” Here, the inner “If block” condition executes only when outer “If block” condition is true.

**Syntax :**

```

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

```

}
else {
    // code to be executed
}

```

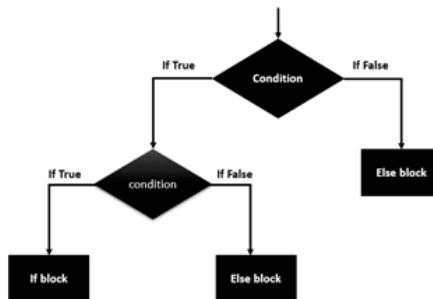


Image: Nested if statement

### **Example:**

```

import java.io.*;
public class Main {
    public static void main(String args[]){
        int age = 19;
        if(age > 18) {
            if(age == 19) {
                System.out.println("Just 19");
            }
        }
        else{
            System.out.println("Under 18");
        }
    }
}

```

### **Output:**

Just 19

### **Switch Case**

The Java switch case statement conducts one statement from multiplex agreements. It is like if-else-if ladder assertion. The switch case statement pursues with byte, short, int, long, enum types, String and few wrapper types like Byte, Short, Int, and Long. Since Java 7, you'll be able to use strings in the switch statement. In another words, the switch case statement tests the equilibrium of a variable opposed to multiple values.

#### **Important Points:**

- There be able one or N number of case values for a switch manifestation.
- The case value need to be switch manifestation type only. The case value must be verbal or constant. It doesn't allow changeable.
- The case values must be unique. In case of similar value, it provides compile-time error.

- The Java switch manifestation must be of byte, short, int, long (with its Wrapper type), enums and string.
- Every case statement be able to break statement which is optional. When control touches to the break statement, it jumps the control after the switch manifestation. If a break statement is not found, it performs the next case.
- The case value can have a default label which is elective.

### Syntax:

```
switch(expression){
    case value1:
        //code to be executed;
        break; //optional
    case value2:
        //code to be executed;
        break; //optional
    default:
        //if all cases are not matched;
}
```

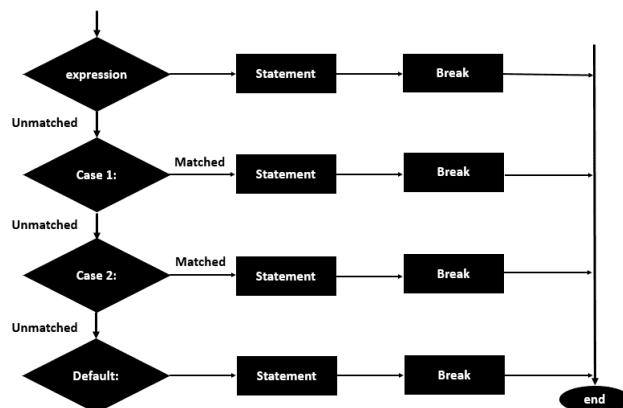


Image: Switch Case

### Example:

```
import java.io.*;
public class Example {
    public static void main(String[] args) {
        int month = 2;

        switch(month){
            case 1: System.out.println("First Month");
            break;
            case 2: System.out.println("Second Month");
            break;
            default: System.out.println("Invalid Month!");
        }
    }
}
```

```
}
```

**Output:**

Second Month

## Loops in Java

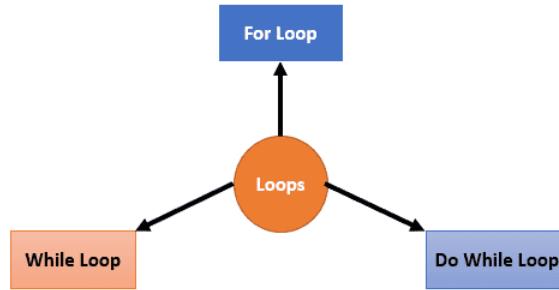


Image: Loops

Loops are used to execute a set of instructions that needs to execute n number of times. In Java, there are their types of loops and these are:

### **For Loop:**

For loop in java is used to iterate a portion of the program in a number of times. If the of the number of iterations is fixed, then is good to go with for loop.

#### **Syntax:**

```
for(variable init; condition; incr/decr){
    // code to be executed
}
```

### **While Loop:**

While loop in java is used to iterate a portion of the program in number of times and If the of the number of iterations is not fixed or unknown, then is good to go with While loop.

#### **Syntax:**

```
while(boolean condition){
    // code to be executed
    incr/decr;
}
```

### **Do While Loop:**

Do While loop in java is used to iterate a part of the program several times and If you need to iterate atleast one time, then is good to go with Do While loop.

#### **Syntax:**

```
do{
    //code to be executed
    incr/decr;
}while(condition);
```

### **for loop**

The Java for loop is employed to repeat a region of the program many times. If the quantity of iteration is known, then it's suggested to use for loop. A simple for loop is that the same as C/C++. we will initialize the variable, check condition and increment/decrement price. It consists of 4 parts:

1. Initialization: It's the initial condition that is dead once once the loop starts. Here, we will initialize the variable, or we will use associate already initialized variable. it's associate optional condition.
2. Condition: It's the second condition that is dead every time to check the condition of the loop. It continues execution till the condition is fake. It should come back Boolean price either true or false. it's associate optional condition.
3. Statement: The statement of the loop is dead every time till the second condition is fake.
4. Increment/Decrement: It increments or decrements the variable price. it's associate optional condition.

### Syntax:

```
for( initialization; condition; incr/decr) {
    //statement or code to be executed
}
```

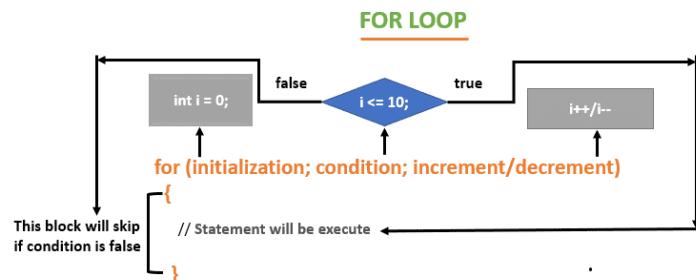


Image: for loop

### Example:

```
public class SimpleLoop {
    public static void main(String[] args) {
        //Code of Java for loop
        for(int i=1; i<=5; i++){
            System.out.println(i);
        }
    }
}
```

### Output:

```
1
2
3
4
5
```

### Nested For Loop

**Syntax:**

```
for( initialization; condition; incr/decr) {  
    //inner for loop  
    for( initialization; condition; incr/decr) {  
        //statement or code to be executed  
    }  
}
```

**Example:**

```
public class exampleLoop {  
    public static void main(String[] args) {  
        //Code of Java for loop  
        for(int i=1; i<=2; i++){  
            for(int j=1; j<=3; j++){  
                System.out.println(i+ " " +j);  
            }  
        }  
    }  
}
```

**Output:**

```
1 1  
1 2  
1 3  
2 1  
2 2  
2 3
```

**Java for-each Loop**

The for-each loop is employed to traverse array or assortment in java. it's easier to use than easy for loop as a result of we do not got to increment worth and use subscript notation. It works on parts basis not index. It returns part one by one within the outlined variable.

**Syntax:**

```
for(Type var:array){  
    //code to be executed  
}
```

**Example:**

```
public class ForEachLoopExample {  
    public static void main(String[] args) {  
        //Array declaration  
        int num[]={ 10, 20, 30, 40, 50};  
        //Printing array using for-each loop  
        for(int i:num){  
            System.out.print(i+ " ");  
        }  
    }  
}
```

## Output:

10 20 30 40 50

### While loop

To iterate a part of the program, the java while loop is operated. If the number of iterations is not stabilized, it is suggested to use while loop.

#### Syntax:

```
while(condition) {
    //statement or code to be executed
    //increment or decrement
}
```

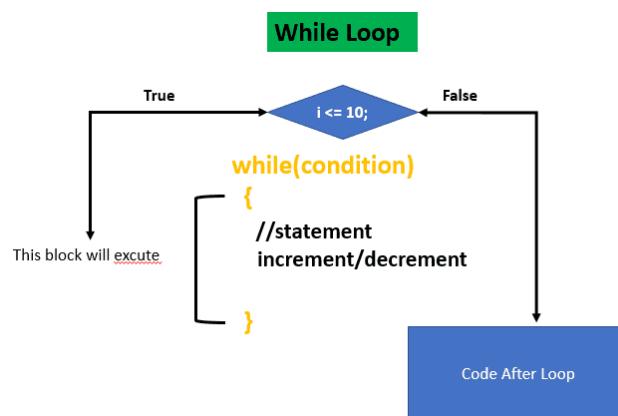


Image: while loop

**There are mainly two parts in while loop.**

- **Condition:** Condition is an expression which is tested. If the expression is true, the loop body will be executed and control goes to increment or decrement part. When the expression becomes false, we exit the while loop.
- **Incr/Decr:** Each iteration loop body is executed, this expression updates (increments or decrements) loop's variable.

Here, the important thing about while loop is that, sometimes it may not even execute. If the expression to be tested outcomes into false, the loop body is skipped and first statement after the while loop will be executed.

#### Example:

```
public class WhileLoop {
    public static void main(String[] args) {
        //Code of Java for loop
        int i = 1;
        while(i<=3){
            System.out.println("Hello From Java");
            i++;//incremented by 1
        }
    }
}
```

}

**Output:**

Hello From Java  
Hello From Java  
Hello From Java

**Do While loop**

The Java do-while loop is used to iterate a part of the program repeatedly, until the specified condition is true. If the number of iterations is not fixed and you must have to execute the loop at least once, it is recommended to use a do-while loop.

Java do-while loop is called an exit control loop. Therefore, unlike while loop and for loop, the do-while check the condition at the end of loop body. The Java do-while loop is executed at least once because condition is checked after loop body.

**Syntax:**

```
do{  
//code to be executed / loop body  
//update statement  
}while (condition);
```

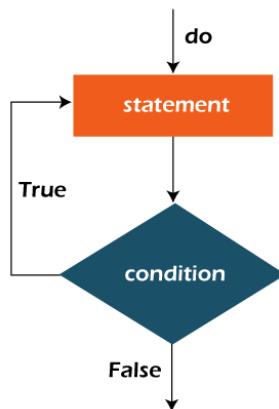


Image: do-while loop

**The different parts of do-while loop:**

- Condition: It is an expression which is tested. If the condition is true, the loop body is executed and control goes to update expression. As soon as the condition becomes false, loop breaks automatically.

Example:

i <=100

- Update expression: Every time the loop body is executed, this expression increments or decrements loop variable.

Example:

i++;

**Example:**

```
public class DoWhileExample {
```

```
public static void main(String[] args) {  
    int i=1;  
    do{  
        System.out.println(i);  
        i++;  
    }while(i<=3);  
}  
}  
Output:  
1  
2  
3
```

### ***Break Statement***

When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop. The Java break statement is used to break loop or switch statement. It breaks the current flow of the program at specified condition. We can use Java break statement in all types of loops such as for loop, while loop and do-while loop.

#### **Syntax:**

```
break;
```

#### **Example:**

```
public class BreakExample {  
    public static void main(String[] args) {  
        //using for loop  
        for(int i=1;i<=10;i++){  
            if(i==5){  
                //breaking the loop  
                break;  
            }  
            System.out.println(i);  
        }  
    }  
}
```

#### **Output:**

```
1  
2  
3
```

### ***Continue Statement***

The continue statement is used in loop control structure when you need to jump to the next iteration of the loop immediately. It can be used with for loop or while loop.

The Java continue statement is used to continue the loop. It continues the current flow of the program and skips the remaining code at the specified condition.

We can use Java continue statement in all types of loops such as for loop, while loop and do-while loop.

**Syntax:**

```
continue;
```

**Example:**

```
public class ContinueExample {  
    public static void main(String[] args) {  
        //for loop  
        for(int i=1;i<=5;i++){  
            if(i==3){  
                //using continue statement  
                continue;//it will skip the rest statement  
            }  
            System.out.println(i);  
        }  
    }  
}
```

**Output:**

```
1  
2  
4  
5
```

## 4.5 Java Arrays

Array is a collection of elements of same type. For example, an int array contains integer elements and a String array contains String elements. The elements of Array are stored in contiguous locations in the memory. This is how an array looks like:

```
int number [] = new int [10];
```

Here number is the array name. The type of the array is integer, which means it can store integer values. The size of the array is 10.

Array works on an index-based system. In the above array, number [0] represents the first element of the array, number [1] represents the second element of the array and so on. The index of array starts from 0 and ends at array\_size-1. In the above example, the index of first element is 0 and index of 10th element is 9.

### Advantages of Array

- **Better performance:** Since array works on a index based system, it is easier to search an element in the array, thus it gives better performance for various operations.
- **Multidimensional:** Unlike ArrayList which is single dimensional, arrays are multidimensional such as 2D array, 3D array etc.
- **Faster access:** Accessing an element is easy in array.

### Disadvantages of Array:

- **Fixed Size:** The size of the array is fixed, which cannot be increased later.
- **Allows only similar type elements:** Arrays are homogeneous, they don't allow different type values, for example an int array cannot hold string elements, similarly a String array cannot hold integer elements. Array is a collection of elements of same type.

For example, an int array contains integer elements and a String array contains String elements. The elements of Array are stored in contiguous locations in the memory.

### Insertion and delegation require shifting of elements.

### Declaration, Instantiation and Initialization of Array in Java

This is how we declare, instantiate and initialize an array.

```
int number[]; //array declaration
number[] = new int[10]; //array instantiation
number[0] = 10; //array Initialization
number[1] = 20; //array Initialization
```

We can also declare an array like this: All the three following syntax are valid for array declaration.

```
int[] number;
int []number;
int number[];
```

### Example:

The following example demonstrates, how we declared an int array, initialized it with integers and print the elements of the array using for loop.

Note: You can see that we have used length property of array to find the size of the array. The length property of array returns the number of elements present in the array.

```
public class JavaExample{
    public static void main(String args[]){
        //array declaration, instantiation and initialization
        int number[] = {11, 22, 33, 44, 55};

        //print array elements
        //length property return the size of the array
        for(int i=0;i<number.length;i++)
            System.out.println("number["+i+"]: "+number[i]);
    }
}
```

### Output:

```
number[0]:11
number[1]:22
number[2]:33
number[3]:44
number[4]:55
```

## Types of array in Java

1. Single Dimensional Array
2. Multidimensional Array

### 1. Single dimensional array

```
public class JavaExample{  
    public static void main(String args[]){  
        //array declaration  
        String names[] = new String[3];  
  
        //array initialization  
        names[0]="Rani";  
        names[1]="Raja";  
        names[2]="Reeta";  
        //print array elements  
        for(int i=0;i<names.length;i++)  
            System.out.println("names["+i+"]: "+names[i]);  
    }  
}
```

#### Output:

```
names[0]: Chaitanya  
names[1]: Ajeet  
names[2]: Rahul
```

### 2. Multidimensional array

#### Multidimensional array declaration:

This is how you can declare a multidimensional array: All the four syntax are valid multidimensional array declaration.

```
int[][] arr;  
int [][]arr;  
int arr[][];  
int []arr[];
```

#### Instantiate Multidimensional Array in Java

Number of elements in multidimensional array = number of rows\*number of columns.

The following array can store upto  $2 \times 3 = 6$  elements.

```
int[][] arr=new int[2][3]; //2 rows and 3 columns
```

#### Initialize Multidimensional Array in Java

```
arr[0][0]=11;  
arr[0][1]=22;  
arr[0][2]=33;  
arr[1][0]=44;  
arr[1][1]=55;  
arr[1][2]=66;
```

#### Example:

```
public class JavaExample{  
    public static void main(String args[]){
```

```
//two rows and three columns
int arr[][]={{11,22,33},{44,55,66};

//outer loop 0 till number of rows
for(int i=0;i<2;i++){
    //inner loop from 0 till number of columns
    for(int j=0;j<3;j++){
        System.out.print(arr[i][j]+" ");
    }
    //new line after each row
    System.out.println();
}
```

**Output:**

```
11 22 33
44 55 66
```

### Print an Array elements using for-each loop

There is another way to print Array elements without using array length property.

```
public class JavaExample{
    public static void main(String args[]){
        //String array
        String names[]{"Chaitanya", "Ajeet", "Rahul", "Hari"};

        //print array elements using for-each loop
        for(String str:names)
            System.out.println(str);

        //int array
        int numbers[]={1, 2, 3, 4, 5};

        //print array elements using for-each loop
        for(int num:numbers)
            System.out.println(num);
    }
}
```

**Output:**

```
Chaitanya
Ajeet
Rahul
Hari
1
2
3
4
5
```

### Exception: **ArrayIndexOutOfBoundsException**

ArrayIndexOutOfBoundsException occurs when we access an array with an invalid index. This happens when the index is either negative or greater than or equal to the size of the array.

```
public class JavaExample{  
    public static void main(String args[]){  
        int number[]={1, 5, 7, 9, 11};  
        for(int i=0;i<=number.length;i++){  
            System.out.println(number[i]);  
        }  
    }  
}
```

**Output:**

ArrayIndexOutOfBoundsException in Java

## 4.6 Java Class and Objects

### Object in Java

An entity that has state and behavior is known as an object e.g., chair, bike, marker, pen, table, car, etc. It can be physical or logical (tangible and intangible). The example of an intangible object is the banking system.

**An object has three characteristics:**

- State: represents the data (value) of an object.
- Behavior: represents the behavior (functionality) of an object such as deposit, withdraw, etc.
- Identity: An object identity is typically implemented via a unique ID. The value of the ID is not visible to the external user. However, it is used internally by the JVM to identify each object uniquely.

### Objects: Real World Examples



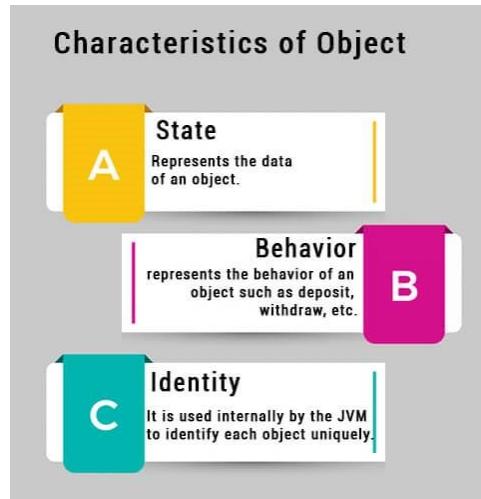


Image: Object

Reference: <https://www.javatpoint.com/object-and-class-in-java>

For Example, Pen is an object. Its name is Reynolds; color is white, known as its state. It is used to write, so writing is its behavior. An object is an instance of a class. A class is a template or blueprint from which objects are created. So, an object is the instance(result) of a class.

### Object Definitions:

- An object is a real-world entity.
- An object is a runtime entity.
- The object is an entity which has state and behavior.
- The object is an instance of a class.

### *Class in Java*

A class is a group of objects which have common properties. It is a template or blueprint from which objects are created. It is a logical entity. It can't be physical.

### Examples of states and behaviors

#### Example 1:

**Object:** House

**State:** Address, Color, Area

**Behavior:** Open door, close door

So, if I had to write a class based on states and behaviours of House. I can do it like this: States can be represented as instance variables and behaviours as methods of the class. We will see how to create classes in the next section of this guide.

```
class House {  
    String address;  
    String color;  
    double area;  
    void openDoor() {
```

```
//Write code here
}
void closeDoor() {
    //Write code here
}
...
...
}
```

**Example 2:****Object:** Car**State:** Color, Brand, Weight, Model**Behavior:** Break, Accelerate, Slow Down, Gear change.**Syntax:**

```
class <class_name>{
    field;
    method;
}
```

Here, fields (variables) and methods represent the state and behavior of the object respectively.

- fields are used to store data
- methods are used to perform some operations

For our bicycle object, we can create the class as

```
class Bicycle {
// state or field
private int gear = 5;

// behavior or method
public void braking() {
    System.out.println("Working of Braking");
}
}
```

In the above example, we have created a class named Bicycle. It contains a field named gear and a method named braking(). Here, Bicycle is a prototype. Now, we can create any number of bicycles using the prototype. And, all the bicycles will share the fields and methods of the prototype.

**Note:** We have used keywords private and public. These are known as access modifiers.

### **Java Objects**

An object is called an instance of a class. For example, suppose Bicycle is a class then MountainBicycle, SportsBicycle, TouringBicycle, etc can be considered as objects of the class.

## Creating an Object in Java

Here is how we can create an object of a class.

```
className object = new className();
// for Bicycle class
```

```
Bicycle sportsBicycle = new Bicycle();
Bicycle touringBicycle = new Bicycle();
```

Here, sportsBicycle and touringBicycle are the names of objects. We can use them to access fields and methods of the class. As you can see, we have created two objects of the class. We can create multiple objects of a single class in Java.

**Note:** Fields and methods of a class are also called members of the class.

## Access Members of a Class

We can use the name of objects along with the **. operator** to access members of a class. For example,

```
class Bicycle {
```

```
// field of class
int gear = 5;
// method of class
void braking() {
    ...
}
```

```
// create object
Bicycle sportsBicycle = new Bicycle();
// access field and method
```

```
sportsBicycle.gear;
```

In the above example, we have created a class named Bicycle. It includes a field named gear.

Here, we have created an object of Bicycle named sportsBicycle. We then use the object to access the field of the class.

## Methods

A method in Java or Java Method is a collection of statements that perform some specific tasks and return the result to the caller. A Java method can perform some specific tasks without returning anything. Methods in Java allow us to reuse the code without retyping the code.

### Method Declaration

In general, method declarations have **six components**:

- Modifier: It defines the access type of the method i.e. from where it can be accessed in your application. In Java, there 4 types of access specifiers.
  - public: It is accessible in all classes in your application.
  - protected: It is accessible within the class in which it is defined and in its subclass/es
  - private: It is accessible only within the class in which it is defined.

- default: It is declared/defined without using any modifier. It is accessible within the same class and package within which its class is defined.
- The return type: The data type of the value returned by the method or void if does not return a value.
- Method Name: the rules for field names apply to method names as well, but the convention is a little different.
- Parameter list: Comma-separated list of the input parameters is defined, preceded with their data type, within the enclosed parenthesis. If there are no parameters, you must use empty parentheses () .
- Exception list: The exceptions you expect by the method can throw; you can specify these exception(s).
- Method body: it is enclosed between braces. The code you need to be executed to perform your intended operations.

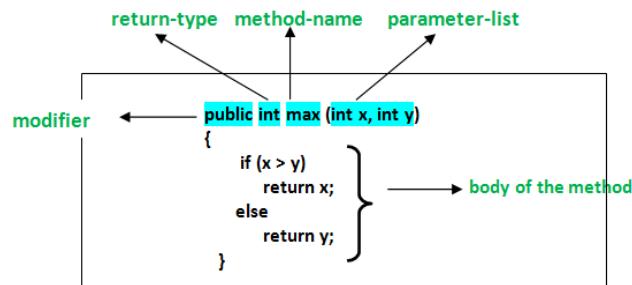


Image: Method Declaration

## Calling a Method

```
// calls the method
addNumbers();
```

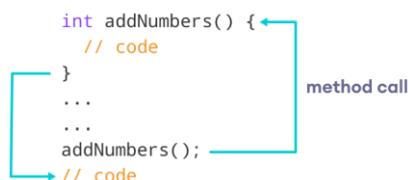


Image: Calling a Method

## Example

```
class Main {
    // create a method
    public int addNumbers(int a, int b) {
        int sum = a + b;
        // return value
        return sum;
    }
}
```

```
public static void main(String[] args) {  
  
    int num1 = 25;  
    int num2 = 15;  
  
    // create an object of Main  
    Main obj = new Main();  
    // calling method  
    int result = obj.addNumbers(num1, num2);  
    System.out.println("Sum is: " + result);  
}  
}
```

### Output

Sum is: 40

In the above example, we have created a method named addNumbers(). The method takes two parameters a and b. Notice the line,

```
int result = obj.addNumbers(num1, num2);
```

Here, we have called the method by passing two arguments num1 and num2. Since the method is returning some value, we have stored the value in the result variable.

**Note:** The method is not static. Hence, we are calling the method using the object of the class.

## Method Return Type

A Java method may or may not return a value to the function call. We use the return statement to return any value. For example,

```
int addNumbers() {  
    ...  
    return sum;  
}
```

Here, we are returning the variable sum. Since the return type of the function is int. The sum variable should be of int type. Otherwise, it will generate an error.

### Example:

```
class Main {  
// create a method  
    public static int square(int num) {  
        // return statement  
        return num * num;  
    }  
    public static void main(String[] args) {  
        int result;  
        // call the method  
        // store returned value to result  
        result = square(10);  
  
        System.out.println("Squared value of 10 is: " + result);  
    }  
}
```

}

**Output:**

Squared value of 10 is: 100

```
int square(int num) { ←  
    return num * num;  
}  
...  
...  
result = square(10); ↓  
// code
```

A diagram illustrating method return type. A blue bracket labeled "method call" encloses the line "result = square(10);". A blue arrow labeled "return value" points from the line "return num \* num;" to the variable "result".

Image: Method Return Type

In the above program, we have created a method named `square()`. The method takes a number as its parameter and returns the square of the number.

Here, we have mentioned the return type of the method as `int`. Hence, the method should always return an integer value.

## Types of Methods in Java

There are two types of methods in Java:

1. Predefined Method: In Java, predefined methods are the method that is already defined in the Java class libraries known as predefined methods. It is also known as the standard library method or built-in method. We can directly use these methods just by calling them in the program at any point.

**Example:**

```
public class Main {  
    public static void main(String[] args) {  
  
        // using the sqrt() method  
        System.out.print("Square root of 4 is: " + Math.sqrt(4));  
    }  
}
```

**Output:**

Square root of 4 is: 2.0

2. User-defined Method: The method written by the user or programmer is known as a user-defined method. These methods are modified according to the requirement.

Example

```
import java.util.Scanner;  
public class EvenOdd  
{  
    public static void main (String args[])  
    {
```

```
//creating Scanner class object
Scanner scan=new Scanner(System.in);
System.out.print("Enter the number: ");
//reading value from user
int num=scan.nextInt();
//method calling
findEvenOdd(num);
}
//user defined method
public static void findEvenOdd(int num)
{
//method body
if(num%2==0)
System.out.println(num+" is even");
else
System.out.println(num+" is odd");
}
```

**Output:**

Enter the number: 12

12 is even

Output 2:

Enter the number: 99

99 is odd

## Static Method

The static keyword is used to construct methods that will exist regardless of whether or not any instances of the class are generated. Any method that uses the static keyword is referred to as a static method.

**Features of static method:**

- A static method in Java is a method that is part of a class rather than an instance of that class.
- Every instance of a class has access to the method.
- Static methods have access to class variables (static variables) without using the class's object (instance).
- Only static data may be accessed by a static method. It is unable to access data that is not static (instance variables).
- In both static and non-static methods, static methods can be accessed directly.

**Syntax:**

```
Access_modifier static void methodName()
{
    // Method body.
}
```

The name of the class can be used to invoke or access static methods.

**Syntax to call a static method:**

```
className.methodName();  
  
Example  
class JavaExample{  
    static int i = 100;  
    static String s = "book";  
    //Static method  
    static void display()  
{  
    System.out.println("i:"+i);  
    System.out.println("i:"+s);  
}  
  
//non-static method  
void funcn()  
{  
    //Static method called in non-static method  
    display();  
}  
//static method  
public static void main(String args[])  
{  
    JavaExample obj = new JavaExample();  
    //You need to have object to call this non-static method  
    obj.funcn();  
  
    //Static method called in another static method  
    display();  
}  
}
```

**Output:**

```
i:100  
i:book  
i:100  
i:book
```

## Constructors

In Java, a constructor is a block of codes similar to the method. It is called when an instance of the class is created. At the time of calling constructor, memory for the object is allocated in the memory.

It is a special type of method which is used to initialize the object. Every time an object is created using the `new()` keyword, at least one constructor is called.

It calls a default constructor if there is no constructor available in the class. In such case, Java compiler provides a default constructor by default. There are two types of constructors in Java: no-arg constructor, and parameterized constructor.

Note: It is called constructor because it constructs the values at the time of object creation. It is not necessary to write a constructor for a class. It is because java compiler creates a default constructor if your class doesn't have any.

Rules for creating Java constructor

There are two rules defined for the constructor.

- Constructor name must be the same as its class name
- A Constructor must have no explicit return type
- A Java constructor cannot be abstract, static, final, and synchronized

## Types of Java constructors

There are two types of constructors in Java:

- Default constructor (no-arg constructor)
- Parameterized constructor

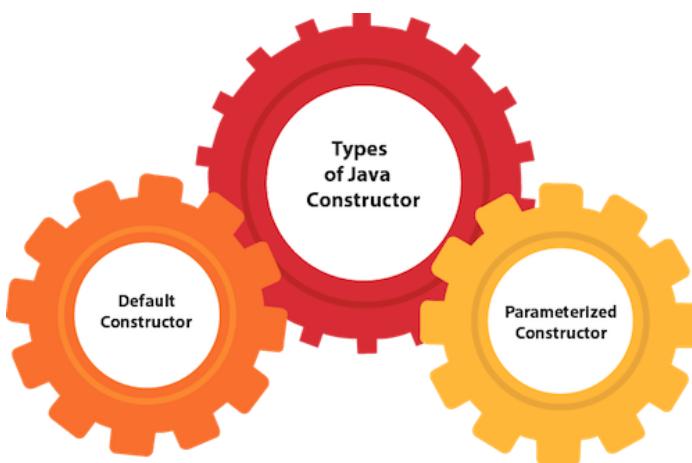


Image: Types of Constructors

### Java Default Constructor

A constructor is called "Default Constructor" when it doesn't have any parameter.

#### Syntax:

```
<class_name>(){}  
                  |  
                  +-->
```

#### Example:

In this example, we are creating the no-arg constructor in the Bike class. It will be invoked at the time of object creation.

```
//Java Program to create and call a default constructor  
class Bike1{  
    //creating a default constructor  
    Bike1(){System.out.println("Bike is created");}  
    //main method  
    public static void main(String args[]){  
        //calling a default constructor  
        Bike1 b=new Bike1();  
    }  
}
```

**Output:**

Bike is created

**Java Parameterized Constructor**

A constructor which has a specific number of parameters is called a parameterized constructor.

In this example, we have created the constructor of Student class that have two parameters. We can have any number of parameters in the constructor.

//Java Program to demonstrate the use of the parameterized constructor.

```
class Student4{  
    int id;  
    String name;  
    //creating a parameterized constructor  
    Student4(int i, String n){  
        id = i;  
        name = n;  
    }  
    //method to display the values  
    void display(){System.out.println(id+" "+name);}  
  
    public static void main(String args[]){  
        //creating objects and passing values  
        Student4 s1 = new Student4(111, "Raja");  
        Student4 s2 = new Student4(222, "Rani");  
        //calling method to display the values of object  
        s1.display();  
        s2.display();  
    }  
}
```

**Output:**

111 Raja  
222 Rani

***Constructor Overloading***

In Java, a constructor is just like a method but without return type. It can also be overloaded like Java methods. Constructor overloading in Java is a technique of having more than one constructor with different parameter lists. They are arranged in a way that each constructor performs a different task. They are differentiated by the compiler by the number of parameters in the list and their types.

**Example:**

```
//Java program to overload constructors  
class Student5{  
    int id;  
    String name;
```

```
int age;
//creating two arg constructor
Student5(int i,String n){
id = i;
name = n;
}
//creating three arg constructor
Student5(int i,String n,int a){
id = i;
name = n;
age=a;
}
void display(){System.out.println(id+" "+name+" "+age);}

public static void main(String args[]){
Student5 s1 = new Student5(111,"Raja");
Student5 s2 = new Student5(222,"Rani",25);
s1.display();
s2.display();
}
```

**Output:**

111 Raja 0  
222 Rani 25

**Difference between constructor and method**

Java Constructor	Java Method
A constructor is used to initialize the state of an object.	A method is used to expose the behavior of an object.
A constructor must not have a return type.	A method must have a return type.
The constructor is invoked implicitly.	The method is invoked explicitly.
The Java compiler provides a default constructor if you don't have any constructor in a class.	The method is not provided by the compiler in any case.
The constructor name must be same as the class name.	The method name may or may not be same as the class name.

## Strings

A string is an immutable object representing a sequence of characters in Java. The immutable property does not allow you to modify a single character of the string, you have to delete the whole string or make a new one. String is a sequence of characters. But in Java, string is an object that represents a sequence of characters. The `java.lang.String` class is used to create a string object.

### How to create a string object?

There are two ways to create String object:

- By string literal
- By new keyword

#### Method 1: Using a string literal

A string literal is the most common practice being followed to create a new string in Java. The first syntax provided below refers to creating a string using a string literal:

**String s=<value>"**

The instances in the above syntax are:

- String is the keyword used to create string literals
- s is the string object's name
- the <value> is the sequence of characters

Whenever the string object is created using the string literal method, JVM matches the string(being created) in the existing list of strings (from string constant pool). If the string already exists, this method will not create a new string, it will refer to the already stored string.

#### Java String literal is created by using double quotes.

##### For Example:

String s="welcome";

Each time you create a string literal, the JVM checks the "string constant pool" first. If the string already exists in the pool, a reference to the pooled instance is returned. If the string doesn't exist in the pool, a new string instance is created and placed in the pool. **For example:**

String s1="Welcome";

String s2="Welcome";//It doesn't create a new instance

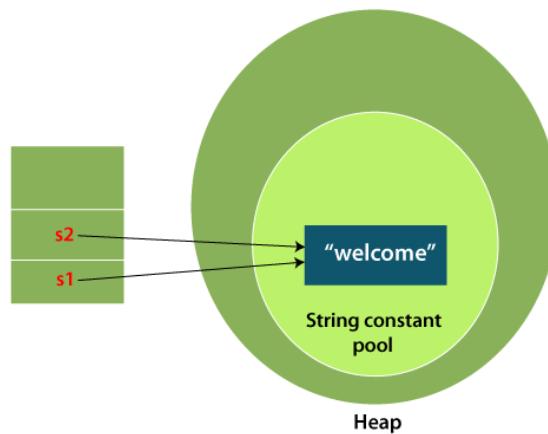


Image: String literal  
Reference: <https://www.javatpoint.com/java-string>

In the above example, only one object will be created. Firstly, JVM will not find any string object with the value "Welcome" in string constant pool that is why it will create a new object. After that it will find the string with the value "Welcome" in the pool, it will not create a new object but will return the reference to the same instance.

**Note: String objects are stored in a special memory area known as the "string constant pool".**

## Method 2: Using the new operator

The following syntax can be followed to create a string in Java using the new keyword.

**String = new String("<value>")**

The new operator always creates a new object rather than referring to the already stored string. Thus, it is recommended to create a string using the string literal as this method optimizes the memory as well.

String s=new String("Welcome");//creates two objects and one reference variable  
In such case, JVM will create a new string object in normal (non-pool) heap memory, and the literal "Welcome" will be placed in the string constant pool. The variable s will refer to the object in a heap (non-pool).

### Example

```
public class StringExample{  
    public static void main(String args[]){  
        String s1="java";//creating string by Java string literal  
        char ch[]={‘s’,’t’,’r’,’i’,’n’,’g’,’s’};  
        String s2=new String(ch);//converting char array to string  
        String s3=new String("example");//creating Java string by new keyword  
        System.out.println(s1);  
        System.out.println(s2);  
        System.out.println(s3);  
    }  
}
```

### Output:

```
java
```

strings  
example

### Immutable String in Java

A String is an unavoidable type of variable while writing any application program. String references are used to store various attributes like username, password, etc. In Java, String objects are immutable. Immutable simply means unmodifiable or unchangeable. Once String object is created its data or state can't be changed but a new String object is created.

#### Example

```
class Testimmutablestring{  
    public static void main(String args[]){  
        String s="Sachin";  
        s.concat(" Tendulkar");//concat() method appends the string at the end  
        System.out.println(s);//will print Sachin because strings are immutable objects  
    }  
}
```

#### Output:

Sachin

Here Sachin is not changed but a new object is created with Sachin Tendulkar. That is why String is known as **immutable**.

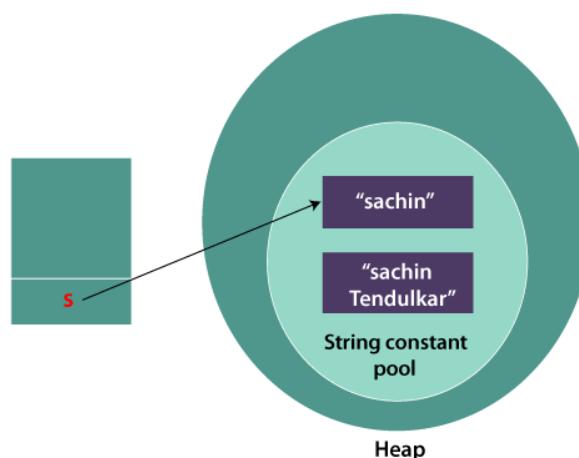


Image: immutable string

Reference: <https://www.javatpoint.com/immutable-string>

As you can see in the above figure that two objects are created but s reference variable still refers to "Sachin" not to "Sachin Tendulkar". But if we explicitly assign it to the reference variable, it will refer to "Sachin Tendulkar" object.

```
class Testimmutablestring1{  
    public static void main(String args[]){  
        String s="Sachin";  
        s=s.concat(" Tendulkar");  
        System.out.println(s);  
    }  
}
```

}

**Output:**

Sachin Tendulkar

In such a case, s points to the "Sachin Tendulkar". Please notice that still Sachin object is not modified.

## Why String objects are immutable in Java?

As Java uses the concept of String literal. Suppose there are 5 reference variables, all refer to one object "Sachin". If one reference variable changes the value of the object, it will be affected by all the reference variables. That is why String objects are immutable in Java. Following are some features of String which makes String objects immutable.

- **ClassLoader:** A ClassLoader in Java uses a String object as an argument. Consider, if the String object is modifiable, the value might be changed and the class that is supposed to be loaded might be different. To avoid this kind of misinterpretation, String is immutable.
- **Thread Safe:** As the String object is immutable we don't have to take care of the synchronization that is required while sharing an object across multiple threads.
- **Security:** As we have seen in class loading, immutable String objects avoid further errors by loading the correct class. This leads to making the application program more secure. Consider an example of banking software. The username and password cannot be modified by any intruder because String objects are immutable. This can make the application program more secure.
- **Heap Space:** The immutability of String helps to minimize the usage in the heap memory. When we try to declare a new String object, the JVM checks whether the value already exists in the String pool or not. If it exists, the same value is assigned to the new object. This feature allows Java to use the heap space efficiently.

### *String Methods*

**length():** This method returns the length of this string. The length is equal to the number of 16-bit Unicode characters in the string.

**Syntax:**

```
public int length()
```

**Example**

```
public class Sample_String{  
    public static void main(String[] args){  
        String str_Sample = "RockStar";  
        //Length of a String  
        System.out.println("Length of String: " + str_Sample.length());}}}
```

**Output:**

Length of String: 8

**indexOf()**

The indexOf() method returns the position of the first occurrence of specified character(s) in a string

**Syntax:**

```
public int indexOf(String str)
public int indexOf(String str, int fromIndex)
public int indexOf(int char)
public int indexOf(int char, int fromIndex)
```

**Example**

```
public class Sample_String{
    public static void main(String[] args){//Character at position
String str_Sample = "RockStar";
System.out.println("Character at position 5: " + str_Sample.charAt(5));
//Index of a given character
System.out.println("Index of character 'S': " + str_Sample.indexOf('S'));
```

**Output:**

Character at position 5: t

Index of character 'S': 4

**charAt()**

The charAt() method returns the character at the specified index in a string. The index of the first character is 0, the second character is 1, and so on.

**Syntax:**

```
public char charAt(int index)
```

**Example**

```
public class Sample_String{
    public static void main(String[] args){//Character at position
String str_Sample = "RockStar";
System.out.println("Character at position 5: " + str_Sample.charAt(5));}
```

**Output:**

Character at position 5: t

**CompareTo()**

The compareTo() method compares two strings lexicographically. The comparison is based on the Unicode value of each character in the strings. The method returns 0 if the string is equal to the other string.

Use the method “compareTo” and specify the String that you would like to compare. Use “compareIgnoreCase” in case you don’t want the result to be case sensitive.

The result will have the value 0 if the argument string is equal to this string; a value less than 0 if this string is lexicographically less than the string argument; and a value greater than 0 if this string is lexicographically greater than the string argument.

**Syntax:**

```
public int compareTo(String string2)  
public int compareTo(Object object)
```

**Example**

```
public class Sample_String{  
    public static void main(String[] args){//Compare to a String  
String str_Sample = "RockStar";  
        System.out.println("Compare      To      'ROCKSTAR':      "      +  
str_Sample.compareTo("rockstar"));  
        //Compare to - Ignore case  
        System.out.println("Compare      To      'ROCKSTAR'      -      Case      Ignored:      "      +  
str_Sample.compareToIgnoreCase("ROCKSTAR"));}}
```

**Output:**

```
Compare To 'ROCKSTAR': -32  
Compare To 'ROCKSTAR' - Case Ignored: 0
```

**Contains()**

The contains() method checks whether a string contains a sequence of characters. Returns true if the characters exist and false if not.

Use the method “contains” and specify the characters you need to check. Returns true if and only if this string contains the specified sequence of char values.

**Syntax:**

```
public boolean contains(CharSequence chars)
```

**Example**

```
public class Sample_String{  
    public static void main(String[] args){ //Check if String contains a sequence  
String str_Sample = "RockStar";  
    System.out.println("Contains sequence 'tar': " + str_Sample.contains("tar"));}}
```

**Output:**

```
Contains sequence 'tar': true
```

**startsWith() & endsWith()**

The endsWith() method checks whether a string ends with the specified character(s). Returns true if the character sequence represented by the argument is a suffix of the character sequence represented by this object.

**Syntax:**

```
public boolean endsWith(String chars)  
public boolean startsWith(String chars)
```

**Example**

```
public class Sample_String{  
    public static void main(String[] args){ //Check if ends with a particular sequence  
String str_Sample = "RockStar";  
    System.out.println("EndsWith character 'r': " + str_Sample.endsWith("r"));}}
```

**Output:**

EndsWith character 'r': true

***replaceAll() & replaceFirst()***

Java String Replace, replaceAll and replaceFirst methods. You can specify the part of the String you want to replace and the replacement String in the arguments.

Example

```
public class Sample_String{  
    public static void main(String[] args){//Replace Rock with the word Duke  
String str_Sample = "RockStar";  
System.out.println("Replace 'Rock' with 'Duke': " + str_Sample.replace("Rock",  
"Duke"));}  
Output:
```

Replace 'Rock' with 'Duke': DukeStar

***toLowerCase() & Java touppercase()***

Just use the “toLowerCase()” or “ToUpperCase()” methods against the Strings that need to be converted.

**Syntax:**

```
public String toLowerCase()  
public String toUpperCase()
```

**Example**

```
public class Sample_String{  
    public static void main(String[] args){//Convert to LowerCase  
String str_Sample = "RockStar";  
System.out.println("Convert to LowerCase: " + str_Sample.toLowerCase());  
//Convert to UpperCase  
System.out.println("Convert to UpperCase: " + str_Sample.toUpperCase());}
```

Output:

Convert to LowerCase: rockstar  
Convert to UpperCase: ROCKSTAR

## Access Modifiers in Java

Access modifiers are keywords in Java that are used to set accessibility. An access modifier restricts the access of a class, constructor, data member and method in

another class. Java language has four access modifiers to control access level for classes and its members.

- Default: Default has scope only inside the same package
- Public: Public has scope that is visible everywhere
- Protected: Protected has scope within the package and all sub classes
- Private: Private has scope only within the classes

Java also supports many non-access modifiers, such as static, abstract, synchronized, native, volatile, transient etc.

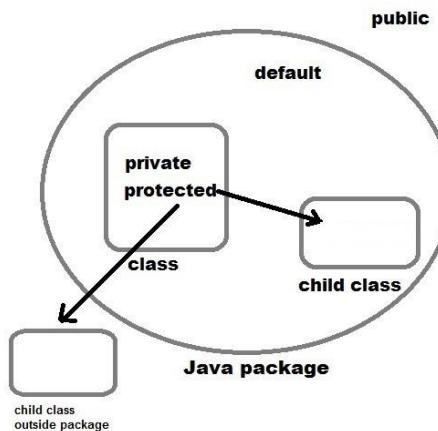


Image: Access modifiers

Reference: [https://miro.medium.com/max/605/0\\*4k3XjYwtO7k5FHuP.jpg](https://miro.medium.com/max/605/0*4k3XjYwtO7k5FHuP.jpg)

Access Modifier	Within Class	Within Package	Same Package by subclasses	Outside Package by subclasses	Global
Public	Yes	Yes	Yes	Yes	Yes
Protected	Yes	Yes	Yes	Yes	No
Default	Yes	Yes	Yes	No	No
Private	Yes	No	No	No	No

Image: Access modifiers

Reference: [https://1.bp.blogspot.com/-GCbzAxJ3\\_t8/XeqabcPeFCI/AAAAAAAABRY/LTJz83SB0zw9Ur7SNaEb2wMq3-QiEmuACLcBGAsYHQ/s640/Access\\_Modifier.png](https://1.bp.blogspot.com/-GCbzAxJ3_t8/XeqabcPeFCI/AAAAAAAABRY/LTJz83SB0zw9Ur7SNaEb2wMq3-QiEmuACLcBGAsYHQ/s640/Access_Modifier.png)

## Default Access Modifier

If we don't specify any access modifier then it is treated as default modifier. It is used to set accessibility within the package. It means we cannot access its method or class from outside the package. It is also known as package accessibility modifier.

### Example:

In this example, we created a Demo class inside the package1 and another class Test by which we are accessing show() method of Demo class. We did not mentioned

access modifier for the show() method that's why it is not accessible and reports an error during compile time.

**//Demo.java**

```
package package1;
public class Demo {
    int a = 10;
    // default access modifier
    void show() {
        System.out.println(a);
    }
}
```

**//Test.java**

```
import package1.Demo;
public class Test {
    public static void main(String[] args) {
        Demo demo = new Demo();
        demo.show(); // compile error
    }
}
```

**Output:**

The method show() from the type Demo is not visible

**Public Access Modifier**

Public access modifier is used to set public accessibility to a variable, method or a class. Any variable or method which is declared as public can be accessible from anywhere in the application.

**Example:**

Here, we have two class Demo and Test located in two different package. Now we want to access show method of Demo class from Test class. The method has public accessibility so it works fine. See the below example.

**//Demo.java**

```
package package1;
public class Demo {
    int a = 10;
    // public access modifier
    public void show() {
        System.out.println(a);
    }
}
```

**//Test.java**

```
package package2;
import package1.Demo;
public class Test {
    public static void main(String[] args) {
        Demo demo = new Demo();
        demo.show();
    }
}
```

**Output:**

10

### ***Protected Access Modifier***

Protected modifier protects the variable, method from accessible from outside the class. It is accessible within class, and in the child class (inheritance) whether child is located in the same package or some other package.

#### **Example:**

In this example, Test class is extended by Demo and called a protected method show() which is accessible now due to inheritance.

```
//Demo.java
package package1;
public class Demo {
    int a = 10;
    // public access modifier
    protected void show() {
        System.out.println(a);
    }
}
```

```
//Test.java
package package2;
import package1.Demo;
public class Test extends Demo{
    public static void main(String[] args) {
        Test test = new Test();
        test.show();
    }
}
```

#### **Output:**

10

### ***Private Access Modifier***

Private modifier is most restricted modifier which allows accessibility within same class only. We can set this modifier to any variable, method or even constructor as well.

#### **Example:**

In this example, we set private modifier to show() method and try to access that method from outside the class. Java does not allow to access it from outside the class.

```
//Demo.java
class Demo {
    int a = 10;
    private void show() {
        System.out.println(a);
    }
}
```

```
//Test.java
public class Test {
    public static void main(String[] args) {
```

```
Demo demo = new Demo();
demo.show(); // compile error
}
}
```

**Output:**

The method show() from the type Demo is not visible

### ***Non-access Modifier***

Along with access modifiers, Java provides non-access modifiers as well. These modifiers are used to set special properties to the variable or method.

Non-access modifiers do not change the accessibility of variable or method, but they provide special properties to them. Java provides following non-access modifiers.

- Final
- Static
- Transient
- Synchronized
- Volatile

### ***Final Modifier***

Final modifier can be used with variable, method and class. If variable is declared final then we cannot change its value. If method is declared final then it cannot be overridden and if a class is declared final then we cannot inherit it.

### ***Static modifier***

Static Modifier is used to make field static. We can use it to declare static variable, method, class etc. static can be used to declare class level variable. If a method is declared static then we don't need to have object to access that. We can use static to create nested class.

### ***Transient modifier***

When an instance variable is declared as transient, then its value doesn't persist when an object is serialized.

### ***Synchronized modifier***

When a method is synchronized it can be accessed by only one thread at a time. We will discuss it in detail in Thread.

### ***Volatile modifier***

Volatile modifier tells to the compiler that the volatile variable can be changed unexpectedly by other parts of a program. Volatile variables are used in case of multi-threading program. volatile keyword cannot be used with a method or a class. It can be only used with a variable.

## Java this keyword

In Java, this is a keyword which is used to refer current object of a class. we can it to refer any member of the class. It means we can access any instance variable and method by using this keyword. The main purpose of using this keyword is to solve the confusion when we have same variable name for instance and local variables.

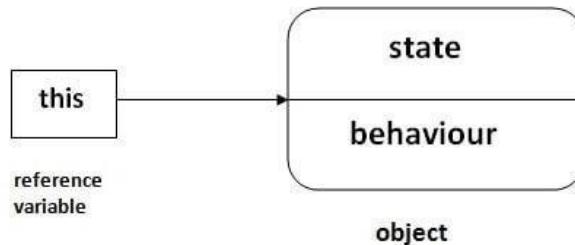


Image: this keyword

Reference: [https://miro.medium.com/max/605/0\\*4k3XjYwtO7k5FHuP.jpg](https://miro.medium.com/max/605/0*4k3XjYwtO7k5FHuP.jpg)

We can use this keyword for the following purpose.

- this keyword is used to refer to current object.
- this is always a reference to the object on which method was invoked.
- this can be used to invoke current class constructor.
- this can be passed as an argument to another method.

### Example:

In this example, we have three instance variables and a constructor that have three parameters with same name as instance variables. Now, we will use this to assign values of parameters to instance variables.

```
class Demo
{
    Double width, height, depth;
    Demo (double w, double h, double d)
    {
        this.width = w;
        this.height = h;
        this.depth = d;
    }
    public static void main(String[] args) {
        Demo d = new Demo(10,20,30);
        System.out.println("width = "+d.width);
        System.out.println("height = "+d.height);
        System.out.println("depth = "+d.depth);
    }
}
```

### Output:

```
width = 10.0
height = 20.0
depth = 30.0
```

Here this is used to initialize member of current object. Such as, this.width refers to the variable of the current object and width only refers to the parameter received in the constructor i.e the argument passed while calling the constructor.

### ***Calling Constructor using this keyword***

We can call a constructor from inside another function by using this keyword

#### **Example:**

In this example, we are calling a parameterized constructor from the non-parameterized constructor using this keyword along with argument.

```
class Demo
{
    Demo ()
    {
        // Calling constructor
        this("Hi");
    }

    Demo(String str){
        System.out.println(str);
    }
    public static void main(String[] args) {
        Demo d = new Demo(); }
}
```

#### **Output:**

Hi

### ***Accessing Method using this keyword***

This is another use of this keyword that allows to access method. We can access method using object reference too but if we want to use implicit object provided by Java then use this keyword.

#### **Example:**

In this example, we are accessing getName() method using this and it works fine as works with object reference. See the below example

```
class Demo
{
    public void getName()
    {
        System.out.println("Hi");
    }
    public void display()
```

```
{  
    this.getName();  
}  
public static void main(String[] args) {  
    Demo d = new Demo();  
    d.display();  
}  
}
```

**Output:**

Hi

***Return Current Object from a Method***

In such scenario, where we want to return current object from a method then we can use this to solve this problem.

**Example:**

In this example, we created a method display that returns the object of Demo class. To return the object, we used this keyword and stored the returned object into Demo type reference variable. We used that returned object to call getName() method and it works fine.

```
class Demo  
{  
    public void getName()  
    {  
        System.out.println("Hi");  
    }  
    public Demo display()  
    {  
        // return current object  
        return this;  
    }  
    public static void main(String[] args) {  
        Demo d = new Demo();  
        Demo d1 = d.display();  
        d1.getName();  
    }  
}
```

**Output:**

Hi

## Static Keyword

The static keyword in Java is used for memory management mainly. It is a keyword that is used to share the same variable or method of a given class. Basically, static is

used for a constant variable or a method that is the same for every instance of a class. The static keyword can be used with class, variable, method, and block. Static members belong to the class instead of a specific instance, this means if you make a member static, you can access it without an object. If we want to access class members without creating an instance of the class, we need to declare the class members static. In Java programming language, static keyword is a non-access modifier and can be used for the following:

- Static Block
- Static Variable (also known as class variable)
- Static Method (also known as class method)
- Static Classes (Nested Classes)

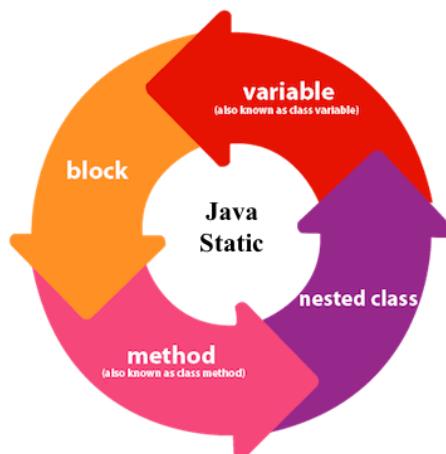


Image: Static Keyword  
Reference: <https://www.javatpoint.com/static-keyword-in-java.jpg>

## Static Block in Java

It is used to initialize the static data member. It is executed before the main method at the time of class loading. If you need to do the computation in order to initialize your static variables, you can declare a static block that gets executed exactly once, when the class is first loaded. We can't access non-static variables in the static block. A class can have multiple Static blocks, which will execute in the same sequence in which they have been written into the program.

### Syntax:

```
static{
    //variable initialization
}
```

### Example:

```
package Demo;
import java.util.*;
public class StaticBlockDemo
{
    //static variable
```

```
static int j = 10;
static int n;

//static block
static
{
    System.out.println ("Static block initialized.");
    n = j * 8;
}
public static void main (String[]args)
{
    System.out.println ("Inside main method");
    System.out.println ("Value of j : " + j);
    System.out.println ("Value of n : " + n);
}
```

**Output:**

Static block initialized  
Inside main method  
Value of j: 10  
Value of n: 80

## Static Variable in Java

If you declare any variable as static, it is known as a static variable. When a variable is declared as static, then a single copy of the variable is created and shared among all objects at the class level. It doesn't matter how many times we initialize a class; there will always be only one copy of a static field belonging to it. The value of this static field will be shared across all objects of either same or any different classes. Static variables are, essentially, global variables. A static variable is common to all the instances (or objects) of the class because it is a class-level variable. Local variables cannot be declared static. If the static variable is not private, we can access it with `ClassName.variableName`

**Syntax:**

static datatype variable-name

**Example:**

```
package Demo;
class Counter {
    static int count = 0;
    Counter(){
        count++;
    }
    public void getCount() {
        System.out.printf("Value of counter: %d \n", count);
    }
    public static void main( String args[] ) {
```

```
Counter c1 = new Counter(); //count incremented to 1
c1.getCount();
Counter c2 = new Counter(); //count incremented to 2
c2.getCount();
Counter c3 = new Counter(); //count incremented to 3
c3.getCount();
}
}
```

**Output:**

Value of counter: 1  
Value of counter: 2  
Value of counter: 3

**Note:**

- We can create static variables at the class level only.
- Static block and static variables are executed in the order they are present in a program.
- It makes your program memory efficient (i.e., it saves memory).

***Static Methods in Java***

A method declared with the static keyword. A static method can access only static variables of a class and invoke only static methods of the class. Usually, static methods are utility methods that we want to expose to be used by other classes without the need of creating an instance. One of the basic rules of working with static methods is that you can't access a non-static method or field from a static method because static methods do not use any instance variables of any object of the class they are defined in. Static methods take all the data from parameters and compute something from those parameters, with no reference to variables.

**Note:** The best-known static method is the main, which is called by the Java runtime to start an application. The main method must be static, which means that applications run in a static context by default.

**Syntax:**

```
static return_type method_name();
```

**Example:**

```
class StaticTest
{
    // non-static method
    int multiply (int a, int b)
    {
        return a * b;
    }
    // static method
    static int add (int a, int b)
    {
        return a + b;
    }
}
```

```
    }
}
public class StaticMethodDemo
{
    public static void main (String[]args)
    {
        // create an instance of the StaticTest class
        StaticTest st = new StaticTest ();
        // call the nonstatic method
        System.out.println (" 5 * 5 = " + st.multiply (5, 5));
        // call the static method
        System.out.println (" 5 + 3 = " + StaticTest.add (5, 3));
    }
}
```

**Output:**

5\*5=25  
5+3=8

**Restrictions for the static method in Java**

There are two main restrictions for the static method. They are:

- The static method cannot use non-static data members or call the non-static method directly.
- this and super cannot be used in a static context.

**Why is the Java main method static?**

It is because the object is not required to call a static method. If it were a non-static method, JVM creates an object first and then calls the main() method which will lead to the problem of extra memory allocation.

**Why it is not required to create an instance of a class?**

Usually, static methods are utility methods that we want to expose to be used by other classes without the need of creating an instance. The static keyword is used to create methods that will exist independently of any instances created for the class. As static methods take all the data from parameters and compute something from those parameters, with no reference to variables. And Class variables and methods can be accessed using the class name followed by a dot and the name of the variable or method. Therefore, the Static Method doesn't require instance creation, so it's generally faster and provides better performance. That's why utility class methods in Wrapper classes, System class, and Collections class are all static methods.

**Note:** The best-known static method is the main, which is called by the Java runtime to start an application. The main method must be static, which means that applications run in a static context by default.

**When to create static methods in Java?**

It's possible to write fluent code when static imports are used. When your method only depends on its parameters, the object state has no effect on the method behavior. Then you can create the method as static.

### Static Class in Java

A class can be made static only if it is a nested class. Java programming language allows us to create a class within a class. It provides a compelling way of grouping elements that are only going to be used in one place, this helps to keep our code more organized and readable.

**Basically, nested classes are of two types:**

1. **Static Nested Classes:** nested classes that are declared static are called static nested classes
2. **Not-static Nested Classes:** nested classes that are non-static are called inner classes or non-static nested classes

**Syntax:**

```
class OuterClass {  
    static class InnerClass {  
        //code  
    }  
}
```

**Example:**

```
class StaticClassDemo  
{  
    //Outer Class  
    static String message = "Hello World!";  
    static class InnerClass  
    {  
        //Inner Class  
        static void getMessage ()  
        {  
            System.out.println (message);  
        }  
    }  
    public static void main (String args[])  
    {  
        StaticClassDemo.InnerClass.getMessage ();  
    }  
}
```

**Output:**

Hello World!

## 4.7 Important Topics of OOPS Concept

## Encapsulation

### *What is Encapsulation in Java?*

The meaning of Encapsulation is to make sure that “sensitive” data is hidden from users. Encapsulation is defined as the wrapping up of data under a single unit. It is the mechanism that binds together code and the data it manipulates. If a data member is private, it means it can only be accessed within the same class. No outside class can access private data members (variable) of other classes.

### **Need for Encapsulation in Java:**

- Better control of class attributes and methods.
- Encapsulation helps us to keep related fields and methods together, which makes our code cleaner and easy to read.
- Helps us to achieve a loose couple.
- An object exposes its behavior by means of public methods or functions.
- Increased security of data.

### **What is Data Hiding in Java?**

Data hiding was introduced as part of the OOP methodology, in which a program is segregated into objects with specific data and functions. This technique enhances a programmer’s ability to create classes with unique data sets and functions, avoiding unnecessary penetration from other program classes. Encapsulation refers to the bundling of related fields and methods together. This allows us to achieve data hiding. Encapsulation in itself is not data hiding. Please have a look at the following image for a better understanding.

### **Difference Between Data Hiding and Encapsulation in Java:**

- Data hiding only hides class data components, whereas data encapsulation hides class data parts and private methods.
- Data hiding focuses more on data security and data encapsulation focuses more on hiding the complexity of the system.

### **How to achieve Encapsulation in Java?**

To achieve encapsulation in Java, you need to:

- declare class variables/attributes as private
- provide public getter and setter methods to access and update the value of a private variable

### **Getter and Setter Methods in Java:**

- The get method returns the variable value, and the set method sets the value.
- If a data member is declared “private”, then it can only be accessed within the same class. No outside class can access data members of that class. If you

need to access these variables, you have to use public “getter” and “setter” methods.

- Getter and Setter’s methods are used to create, modify, delete, and view the values of the variables.
- The syntax for both is that they start with either get or set, followed by the name of the variable, with the first letter in upper case.
- In Java getters and setters are completely ordinary functions. The only thing that makes them getters or setters is a convention.
- A getter for demo is called getDemo and the setter is called setDemo .
- It should have a statement to assign the argument value to the corresponding variable.

**Example:**

```
public class EncapsulationDemo
{
    // private variables declared
    // these can only be accessed by
    // public methods of class
    private String Name;
    private int RollNo;
    private int Age;
    // get method for age to access
    // private variable Age
    public int getAge ()
    {
        return Age;
    }

    // get method for name to access
    // private variable Name
    public String getName ()
    {
        return Name;
    }
    // get method for roll to access
    // private variable RollNo
    public int getRoll ()
    {
        return RollNo;
    }
    // set method for age to access
    // private variable Age
    public void setAge (int newAge)
    {
        Age = newAge;
    }
    // set method for name to access
    // private variable Name
    public void setName (String newName)
```

```
{  
    Name = newName;  
}  
// set method for roll to access  
// private variable RollNo  
public void setRoll (int newRollNo)  
{  
    RollNo = newRollNo;  
}  
public static void main (String[]args)  
{  
    EncapsulationDemo obj = new EncapsulationDemo ();  
    // setting values of the variables  
    obj.setName ("Harsh");  
    obj.setAge (19);  
    obj.setRoll (51);  
    // Displaying values of the variables  
    System.out.println ("Student's Name: " + obj.getName ());  
    System.out.println ("Student's Age: " + obj.getAge ());  
    System.out.println ("Student's RollNo: " + obj.getRoll ());  
}  
}
```

**Output:**

Student's Name: Harry  
Student's Age: 19  
Student's RollNo: 51

## Inheritance with Types

Inheritance in Java is a mechanism in which one object acquires all the properties and behaviors of a parent object. It is an important part of OOPs (Object Oriented programming system). The idea behind inheritance in Java is that you can create new classes that are built upon existing classes. When you inherit from an existing class, you can reuse methods and fields of the parent class. Moreover, you can add new methods and fields in your current class also. Inheritance represents the IS-A relationship which is also known as a parent-child relationship.

### Why use inheritance in java?

- For Method Overriding (so runtime polymorphism can be achieved).
- For Code Reusability.

### Terms used in Inheritance

- **Class:** A class is a group of objects which have common properties. It is a template or blueprint from which objects are created.
- **Sub Class/Child Class:** Subclass is a class which inherits the other class. It is also called a derived class, extended class, or child class.
- **Super Class/Parent Class:** Superclass is the class from where a subclass inherits the features. It is also called a base class or a parent class.

- **Reusability:** As the name specifies, reusability is a mechanism which facilitates you to reuse the fields and methods of the existing class when you create a new class. You can use the same fields and methods already defined in the previous class.

### Syntax:

```
class Subclass-name extends Superclass-name
{
    //methods and fields
}
```

The `extends` keyword indicates that you are making a new class that derives from an existing class. The meaning of "extends" is to increase the functionality.

In the terminology of Java, a class which is inherited is called a parent or superclass, and the new class is called child or subclass.

### Example:

```
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);
    }
}
```

### Output:

Programmer salary is:40000.0  
Bonus of programmer is:10000

## Types of inheritance in java

- Single
- Multiple
- Multilevel
- Hybrid
- Hierarchical

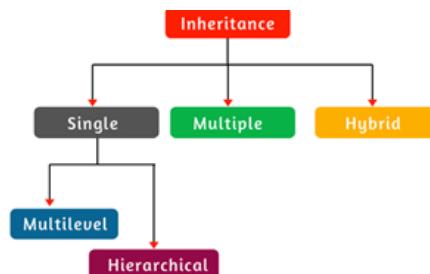


Image: Inheritance with Types

Reference: [https://www.scientecheeasy.com/2020/07/types-of-inheritance-in-java.html/](https://www.scientecheasy.com/2020/07/types-of-inheritance-in-java.html/)

- **Single Inheritance:** When a derived class or subclass inherits from only one base or superclass then it is single inheritance.

- **Multilevel Inheritance:** In Multilevel Inheritance, we have more than one level wherein a class inherits from a base class and the derived class in turn is inherited by another class.
- **Hierarchical Inheritance:** An inheritance hierarchy is formed in this type of inheritance when a superclass is inherited by more than one class.
- **Multiple Inheritance:** Multiple inheritance is the one in which a class can inherit properties and behavior from more than one parent.
- **Hybrid Inheritance:** When one or more types of inheritance are combined, then it becomes a hybrid inheritance.

**Note: Multiple inheritance is not supported in Java through class.**

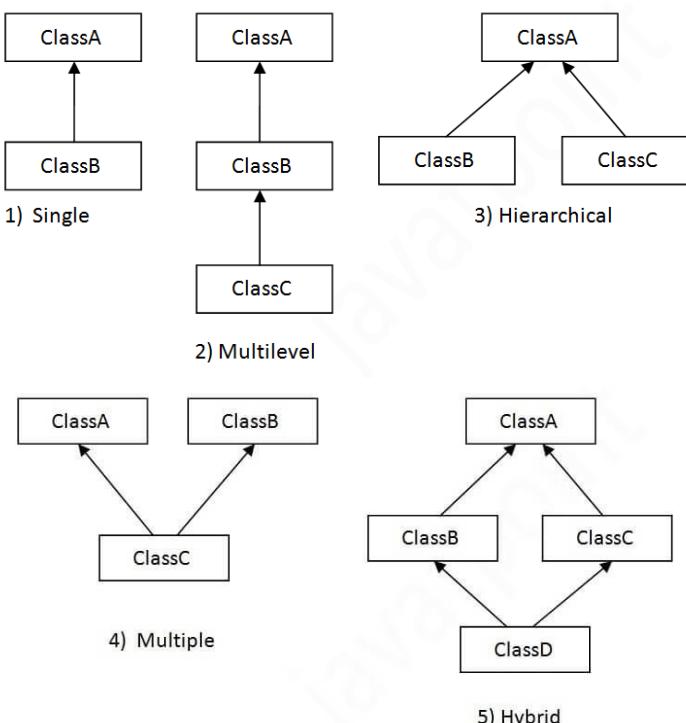


Image: Inheritance with Types

Reference: <https://www.javatpoint.com/inheritance-in-java>

## Single Inheritance

When a class inherits another class, it is known as a single inheritance. In the example given below, Dog class inherits the Animal class, so there is the single inheritance.

### Example:

```
//TestInheritance.java
class Animal{
void eat(){System.out.println("eating...");}
}
class Dog extends Animal{
void bark(){System.out.println("barking...");}
}
```

```
class TestInheritance{  
public static void main(String args[]){  
Dog d=new Dog();  
d.bark();  
d.eat();  
}}
```

**Output:**

barking...  
eating...

## Multilevel Inheritance

When there is a chain of inheritance, it is known as multilevel inheritance. As you can see in the example given below, BabyDog class inherits the Dog class which again inherits the Animal class, so there is a multilevel inheritance.

**Example:**

```
//TestInheritance2.java  
class Animal{  
void eat(){System.out.println("eating...");}  
}  
class Dog extends Animal{  
void bark(){System.out.println("barking...");}  
}  
class BabyDog extends Dog{  
void weep(){System.out.println("weeping...");}  
}  
class TestInheritance2{  
public static void main(String args[]){  
BabyDog d=new BabyDog();  
d.weep();  
d.bark();  
d.eat();  
}}
```

**Output:**

weeping...  
barking...  
eating...

## Hierarchical Inheritance

When two or more classes inherits a single class, it is known as hierarchical inheritance. In the example given below, Dog and Cat classes inherits the Animal class, so there is hierarchical inheritance.

**Example:**

```
//TestInheritance3.java  
class Animal{  
void eat(){System.out.println("eating...");}
```

```
}

class Dog extends Animal{
void bark(){System.out.println("barking...");}
}

class Cat extends Animal{
void meow(){System.out.println("meowing...");}
}

class TestInheritance3{
public static void main(String args[]){
Cat c=new Cat();
c.meow();
c.eat();
//c.bark();//C.T.Error
}}
Output:
meowing...
eating...
```

### Why multiple inheritance is not supported in java?

In java, multiple inheritance is not supported because of ambiguity problem. We can take the below example where we have two classes Class1 and Class2 which have same method display(). If multiple inheritance is possible than Test class can inherit data members (properties) and methods (behaviour) of both Class1 and Class2 classes. Now, Test class have two display() methods inherited from Class1 and Class2. Problem occurs in method call, when display() method will be called with Test class object which method will be called, will it be of Class1 or Class2. This is ambiguity problem because of which multiple inheritance is not supported in java.

#### Example:

```
class Class1{
    public void display(){
        System.out.println("Display method inside Class1.");
    }
}
class Class2{
    public void display(){
        System.out.println("Display method inside Class2.");
    }
}
//let multiple inheritance is possible.
public class Test extends Class1, Class2{
    public static void main(String args[]){
        Test obj = new Test();
        //Ambiguity problem in method call which class display() method will be called.
        obj.display();    }}
}
```

#### Output:

Exception in thread "main" java.lang.Error:

Unresolved compilation problem:

## Method Overloading & Overriding

Method overloading in Java means having two or more methods (or functions) in a class with the same name and different arguments (or parameters). It can be with a different number of arguments or different data types of arguments.

**For instance:**

- void function1(double a) { ... }
- void function1(int a, int b, double c) { ... }
- float function1(float a) { ... }
- double function1(int a, float b) { ... }

In the above example, function1() is overloaded using a different number of parameters and different data types of parameters.

**Example:**

```
class Method_Overloading {  
    double figure(double l, int b) //two parameters with double type  
    {  
        return (l*b);  
    }  
    float figure(int s) //one parameter with float return type  
    {  
        return (s*s);  
    }  
    public static void main(String[] args) {  
        Method_Overloading obj = new Method_Overloading();  
        System.out.println("Area of Rectangle: " +obj.figure(5.55, 6));  
        System.out.println("Area of Square: " +obj.figure(3)); }}
```

**Output:**

Area of Rectangle: 33.3  
Area of Square: 9.0

## By changing the Number of Parameters

A method can differ with the number of parameters passed. The below example shows how we can implement method overloading with the different number of parameters in method declaration and definition.

**Example:**

```
class Method_Overloading {  
    //Method Overloading by changing the number of arguments (or parameters)  
  
    //Method 1  
    double figure(double l, double b) //two arguments or parameters  
    {
```

```
        return (l*b);
    }
double figure(double s) //one argument or parameter
{
    return (s*s);
}

//Method 2
public static void main(String[] args) {
    Method_Overloading obj = new Method_Overloading();
    System.out.println("Area of Rectangle: " +obj.figure(5.55, 6.78));
    System.out.println("Area of Square: " +obj.figure(3.45));
}
```

**Output:**

Area of Rectangle: 37.629  
Area of Square: 11.902500000000002

### Changing the Data Type of Parameters

Another way to do method overloading is by changing the data types of method parameters. The below example shows how we can implement method overloading with the different data types of parameters in method declaration and definition.

**Example:**

```
class Method_Overloading {
    //Method Overloading by changing the data type of arguments (or parameters)
    double figure(double l, double b) //method 1
    {
        return (l*b); //returns area of rectangle
    }
    double figure(int b, int h) //method 2
    {
        return ((b*h)/2); //returns area of right triangle
    }
    double figure(int b, double h) //method 3
    {
        return (b*h); //returns area of parallelogram
    }

    public static void main(String[] args) {
        Method_Overloading obj = new Method_Overloading();
        System.out.println("Area of Rectangle: " +obj.figure(5.55, 6.78));
        System.out.println("Area of Right Triangle: " +obj.figure(3,5));
        System.out.println("Area of Parallelogram: " +obj.figure(4,6.3));
    }
}
```

**Output:**

Area of Rectangle : 37.629

Area of Right Traingle : 7.0

Area of Parallelogram : 25.2

## Can we overload main() in Java?

Yes, evidently main() method can also be overloaded. The below example shows a simple implementation to overload the main() method with a different set of parameters.

### Example:

```
class Main_Overloading {

    //main 1
    public static void main(String[] args)
    {
        System.out.println("Overloading from main 1");
        Main_Overloading.main("User!");
    }

    //main 2
    public static void main(String arg1)
    {
        System.out.println("Hello, " + arg1 + " Overloading main 2");
        Main_Overloading.main("Are you learning", " Main Method
overloading!");
    }

    //main 3
    public static void main(String arg1, String arg2)
    {
        System.out.println("Overloading main 3");
        System.out.println("Hi, " + arg1 + ", " + arg2);
    }
}
```

### Output:

Overloading from main 1

Hello, User! Overloading main 2

Overloading main 3

Hi, Are you learning, Main Method overloading!

## Method Overriding

Method Overriding is redefining the method from the superclass into the subclass by adding new functionality or code. In Java, using the overriding concept superclass provides the freedom to the subclass for writing their own code instead of binding on superclass behavior or code.

## Usage of Method Overriding in Java:

Method overriding is used to provide the specific implementation of a method that is already provided by its superclass. Method overriding is used for runtime polymorphism.

### Rules for Method Overriding in Java:

- The method name must be the same.
- Method parameter types must be the same.
- The method return type must be the same.

### Example:

```
class connection
{
    void connect ()
    {
    }
}

class oracleconnection extends connection
{
    void connect ()
    {
        System.out.println ("Connected to Oracle");
    }
}

class mysqlconnection extends connection
{
    void connect ()
    {
        System.out.println ("Connected to MySQL");
    }
}

class Overriding
{
    public static void main (String args[])
    {
        oracleconnection a1 = new oracleconnection ();
        mysqlconnection b1 = new mysqlconnection ();
        a1.connect ();
        b1.connect ();
    }
}
```

### Output:

Connected to Oracle  
Connected to MySQL

### Dynamic Binding in Java:

When the type of the object is determined at run time, it is known as dynamic binding. In Dynamic binding compiler doesn't decide the method to be called. Overriding is a

perfect example of dynamic binding. In overriding both parent and child classes have the same method.

**Example:**

```
public class Main
{
    public static class superclass
    {
        void print ()
        {
            //print in superclass
            System.out.println ("Hi!");
        }
    }
    public static class subclass extends superclass
    {
        @Override void print ()
        {
            //print in subclass
            System.out.println ("Hello!");
        }
    }
    public static void main (String[]args)
    {
        superclass A = new superclass ();
        superclass B = new subclass ();
        A.print ();
        B.print ();
    }
}
```

**Output:**

Hi!  
Hello!

## Abstract Class

A class which is declared with the abstract keyword is known as an abstract class in Java. It can have abstract and non-abstract methods (method with the body).

### Abstraction in Java

Abstraction is a process of hiding the implementation details and showing only functionality to the user. Another way, it shows only essential things to the user and hides the internal details, for example, sending SMS where you type the text and send the message. You don't know the internal processing about the message delivery. Abstraction lets you focus on what the object does instead of how it does it.

### Ways to achieve Abstraction

There are two ways to achieve abstraction in java

- Abstract class (0 to 100%)
- Interface (100%)

### Abstract class in Java

A class which is declared as abstract is known as an abstract class. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.

- An abstract class must be declared with an abstract keyword.
- It can have abstract and non-abstract methods.
- It cannot be instantiated.
- It can have constructors and static methods also.
- It can have final methods which will force the subclass not to change the body of the method.

**Note:** A method that does not have a body is called an abstract method and the class that is declared by using the abstract keyword is called an abstract class. If a class contains an abstract method, then it must be declared as abstract.

### Abstract Method in Java

A method that does not have a body is called an abstract method in Java. It is declared with the modifier abstract. The following are the properties of the java abstract method,

- It can only be used in an abstract class, and it does not have a body.
- An abstract method contains a method signature, but no method body.
- An abstract method is a method that is declared without implementation.
- Instead of curly braces, an abstract method will have a semicolon (;) at the end.
- A method-defined abstract must always be redefined in the subclass, thus making overriding compulsory OR either making the subclass itself abstract.

### Rules of Abstract Class and Abstract Methods in Java:

**Rule1:** If the method does not have a body it should be declared as abstract using the abstract modifier else it leads to CE: “missing method body or declared abstract”

```
public class Example
{
    void m1(); //CE: missing method body or declared abstract
}
CE: missing method body or declared abstract
```

**Rule2:** If a class has an abstract method, it should be declared as abstract by using the keyword abstract else it leads to CE: class is not abstract and does not override the abstract method in the class.

```
public class Example
```

```
{  
    abstract void m1();  
}
```

CE: Example is not abstract and does not override abstract method m1() in Example.

The correct syntax is given below.

abstract class Example

```
{  
    abstract void m1();  
}
```

**Rule3:** If a class is declared as abstract it cannot be instantiated violation leads to compile-time Error.

abstract class Example

```
{  
    abstract void m1();  
    public static void main(String args[])  
    {  
        Example e = new Example();  
    }  
}
```

CE: Example is abstract, cannot be instantiated

**Rule4:** The subclass of an abstract class should override all abstract methods or it should be declared as abstract else it leads to CE:

```
abstract class Example  
{  
    abstract void m1();  
    abstract void m2();  
}  
class Sample extends Example  
{  
    void m1()  
    {  
        System.out.println("m1 method");  
    }  
}
```

### **Example:**

```
abstract class Bank  
{  
    abstract int getRateOfInterest ();  
}  
class SBI extends Bank  
{  
    int getRateOfInterest ()  
    {  
        return 7;  
    }  
}
```

```
class PNB extends Bank
{
    int getRateOfInterest ()
    {
        return 8;
    }
}
public class Main
{
    public static void main (String[]args)
    {
        Bank b;
        b = new SBI ();
        System.out.println ("SBI Rate of Interest is: " + b.getRateOfInterest () + " %");
        b = new PNB ();
        System.out.println ("PNB Rate of Interest is: " + b.getRateOfInterest () + " %");
    }
}
```

**Output:**

SBI Rate of Interest is: 7%  
PNB Rate of Interest is: 8%

## Interfaces

An interface in Java is a blueprint of a class. It has static constants and abstract methods. The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java. In other words, you can say that interfaces can have abstract methods and variables. It cannot have a method body. Java Interface also represents the IS-A relationship. It cannot be instantiated just like the abstract class.

Since Java 8, we can have default and static methods in an interface.

Since Java 9, we can have private methods in an interface.

### Why use Java interface?

There are mainly three reasons to use interface. They are given below.

- It is used to achieve abstraction.
- By interface, we can support the functionality of multiple inheritance.
- It can be used to achieve loose coupling.

### How to declare an interface?

An interface is declared by using the interface keyword. It provides total abstraction; means all the methods in an interface are declared with the empty body, and all the fields are public, static and final by default. A class that implements an interface must implement all the methods declared in the interface.

### Syntax:

```
interface <interface_name>{
    // declare constant fields
    // declare methods that abstract
    // by default.
}
```

**Example:**

```
interface printable{
void print();
}
class A6 implements printable{
public void print(){System.out.println("Hello");}
public static void main(String args[]){
A6 obj = new A6();
obj.print();
}
}
```

**Output:**

Hello

**Example: Drawable**

In this example, the Drawable interface has only one method. Its implementation is provided by Rectangle and Circle classes. In a real scenario, an interface is defined by someone else, but its implementation is provided by different implementation providers. Moreover, it is used by someone else. The implementation part is hidden by the user who uses the interface

**Example:**

```
//TestInterface1.java
//Interface declaration: by first user
interface Drawable{
void draw();
}
//Implementation: by second user
class Rectangle implements Drawable{
public void draw(){System.out.println("drawing rectangle");}
}
class Circle implements Drawable{
public void draw(){System.out.println("drawing circle");}
}
//Using interface: by third user
class TestInterface1{
public static void main(String args[]){
Drawable d=new Circle(); //In real scenario, object is provided by method e.g.
getDrawable()
d.draw();
}}

```

**Output:**

drawing circle

### Example: Bank

Let's see another example of java interface which provides the implementation of Bank interface.

```
//TestInterface2.java
interface Bank{
float rateOfInterest();
}
class SBI implements Bank{
public float rateOfInterest(){return 9.15f;}
}
class PNB implements Bank{
public float rateOfInterest(){return 9.7f;}
}
class TestInterface2{
public static void main(String[] args){
Bank b=new SBI();
System.out.println("ROI: "+b.rateOfInterest());
}}
```

#### Output:

ROI: 9.15

### Multiple inheritance in Java by interface

If a class implements multiple interfaces, or an interface extends multiple interfaces, it is known as multiple inheritance.

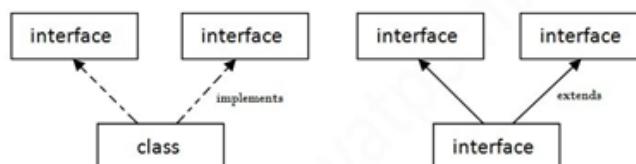


Image: Multiple inheritance in java interface  
Reference: <https://www.javatpoint.com/interface-in-java>

#### Example

```
interface Printable{
void print();
}
interface Showable{
void show();
}
class A7 implements Printable,Showable{
public void print(){System.out.println("Hello");}
public void show(){System.out.println("Welcome");}
}

public static void main(String args[]){
A7 obj = new A7();
obj.print();
```

```
    obj.show();
}
}
```

**Output:**

Hello  
Welcome

## Package

### *What are Packages in Java?*

The package is a collection of classes and interfaces in the form of .class files. It helps organize your classes into a folder structure and makes it easy to locate and use them. More importantly, it helps improve re-usability. Each package in Java has its unique name and organizes its classes and interfaces into a separate namespace, or name group. Although interfaces and classes with the same name cannot appear in the same package, they can appear in different packages.

### **Packages are used for:**

- Preventing naming conflicts.
- Making searching and usage of classes, interfaces, enumerations, and annotations easier.
- Offers access protection such as protected classes, default classes, and private classes.
- Packages can be considered as data encapsulation (or data-hiding).
- With packages, you can organize your project better and easily locate related classes.

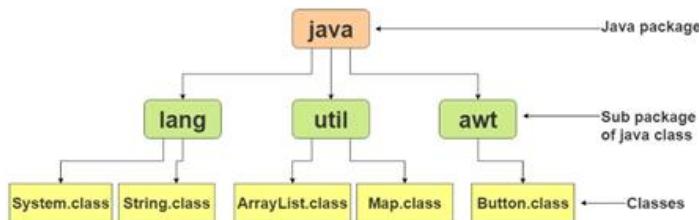


Image: Package

Reference: <https://simplesnippets.tech/packages-in-java/>

## Types of Packages in Java

We have the following two types of packages in Java:

- In-Built Packages
- User-defined Packages

### *In-Built Packages:*

In-Built Packages are the packages that are given by Sun Microsystems or some other companies as a part of Java. We have the following three types of in-built packages.

- **Core Packages:** Core Packages are predefined packages given by Sun MicroSystems which begin with “java”.
- **Extended Packages:** Extended packages are also predefined packages given by Sun Microsystems which begin with “javax”.
- **Third-Party Packages:** Third-Party Packages are also predefined packages that are given by some other companies as a part of Java Software. Example: oracle.jdbc, com.mysql, etc.

### Some In-Built Packages in Java are:

- java.lang: This package is the collection of classes and interfaces using which we can perform basic operations like parsing, storing strings, etc. This package is by default available for every java program. Example: Object, String, Integer, etc.
- java.io: This package is the collection of classes and interfaces using which we can perform basic input and output operations. Example: BufferedReader, Printstream, FileReader, etc.
- java.util: This package is the collection of classes and interfaces using which we can display the date, store the group of objects(Collection API). Example: Date, Collection, ArrayList, etc.
- java.text: This package is the collection of classes and interfaces using which we can perform operations like formatting the date, number, etc. Example: NumberFormat, DateFormat, etc.
- java.net: This package is the collection of classes and interfaces using which we can develop network-related applications. Example: Socket, ServerSocket, etc.
- java.sql: This package is the collection of classes and interfaces using which we can perform JDBC-related operations like connecting to DB, creating the tables, inserting records, etc. Example: Connection, Driver, DriverManager, etc.
- java.awt: This package is the collection of classes and interfaces using which we can develop GUI-based applications. Example: Frame, Label, Button, etc.
- javax.swing: This package is the collection of classes and interfaces using which we can develop better GUI-based Applications. Example: JFrame, JLabel, JButton, etc.
- java.applet: This package is the collection of classes and interfaces using which we can develop applications that run in the browser. Example : Applet, AppletContext, etc.

### User-Defined Packages in Java:

In Java, we can also create user-defined packages according to our requirements. To create the user-defined packages we have to use a java keyword called “package”. User-defined packages contain only user-defined classes or interfaces in the form of .class files. The syntax is given below.

```
package packagename;
```

**Rules:**

- While writing the package name we can specify packages in any number of levels but specifying one level is mandatory.
- The package statement must be written as the first executable statement in the program.
- We can write at most one package statement in the program

**Example:**

```
package Demo;  
public class PackageDemo {  
    public static void main(String args[]) {  
        System.out.println("Have a Nice Day...!!!");  
    }  
}
```

**How to Compile Java Package?**

If you are not using any IDE, you need to follow the syntax given below:

**javac -d directory javafilename**

Example:

**javac -d . PackageDemo.java**

Or

**javac -d E: PackageDemo.java**

Or

**javac -d D:\practice PackageDemo.java**

Here,

**-d:** This option is used to create the package based on the name specified in the program using the package statements and locate the generated class into the package. E: or D: – Specify the location where to locate the created package.

**How to execute the Java Package Program?**

You need to use a fully qualified name to run the class.

**To Compile:** javac -d . PackageDemo.java

**To Run:** java mypack.PackageDemo

**Output:** Have a Nice Day...!!!

**Note:** In Java, for every java program following two packages are available by default:

1. java.lang
2. Current working directory

## How to access package from another package?

There are three ways to access the package from outside the package.

- import package.\*;
- import package.classname;
- fully qualified name.

### Using packagename.\*

If you use package.\* then all the classes and interfaces of this package will be accessible but not subpackages.

The import keyword is used to make the classes and interface of another package accessible to the current package.

#### Example:

```
//save by A.java
package pack;
public class A{
    public void msg(){System.out.println("Hello");}
}

//save by B.java
package mypack;
import pack.*;

class B{
    public static void main(String args[]){
        A obj = new A();
        obj.msg();
    }
}
```

#### Output:

Hello

### Using packagename.classname

If you import package.classname then only declared class of this package will be accessible.

#### Example:

```
//save by A.java
package pack;
public class A{
    public void msg(){System.out.println("Hello");}
}
```

```
//save by B.java
package mypack;
import pack.A;

class B{
    public static void main(String args[]){
        A obj = new A();
        obj.msg();
    }
}
```

**Output:**

Hello

### 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 or interface.

It is generally used when two packages have same class name e.g. java.util and java.sql packages contain Date class.

**Example:**

```
//save by A.java
package pack;
public class A{
    public void msg(){System.out.println("Hello");}
}

//save by B.java
package mypack;
class B{
    public static void main(String args[]){
        pack.A obj = new pack.A(); //using fully qualified name
        obj.msg();
    }
}
```

**Output:**

Hello

### Super Keyword in Java

The super keyword in Java is a reference variable which is used to refer immediate parent class object. Whenever you create the instance of subclass, an instance of parent class is created implicitly which is referred by super reference variable.

#### Usage of Java super Keyword

- super can be used to refer immediate parent class instance variable.
- super can be used to invoke immediate parent class method.

- `super()` can be used to invoke immediate parent class constructor.

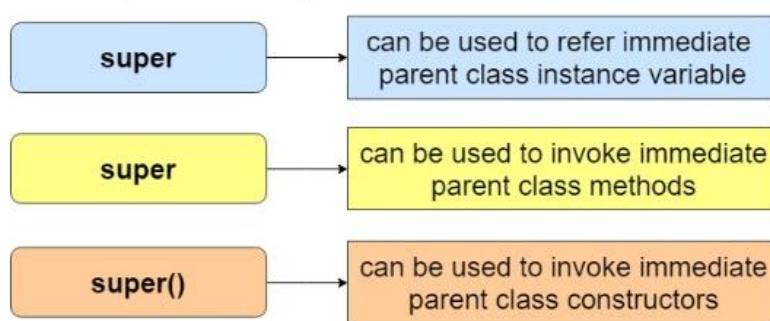


Image: Super keyword  
 Reference: <https://simplesnippets.tech/super-keyword-in-java/>

### Use of super with variables:

We can use super keyword to access the data member or field of parent class. It is used if parent class and child class have same fields. This scenario occurs when a derived class and base class has same data members. In that case there is a possibility of ambiguity for the JVM.

### Example

```

/* Base class vehicle */
class Vehicle
{
    int maxSpeed = 120;
}
/* sub class Car extending vehicle */
class Car extends Vehicle
{
    int maxSpeed = 180;
    void display()
    {
        /* print maxSpeed of base class (vehicle) */
        System.out.println("Maximum Speed:" + maxSpeed );
        System.out.println("Maximum Speed: " + super.maxSpeed);
    }
}
/* Driver program to test */
class Test
{
    public static void main(String[] args)
    {
        Car small = new Car();
        small.display();
    }
}
  
```

### Output:

Maximum Speed: 120  
Maximum Speed: 180

### Use of super with methods:

This is used when we want to call parent class method. So whenever a parent and child class have same named methods then to resolve ambiguity we use super keyword. This code snippet helps to understand the said usage of super keyword.

### Example

```
/* Base class Person */
class Person
{
    void message()
    {
        System.out.println("This is person class");
    }
}
/* Subclass Student */
class Student extends Person
{
    void message()
    {
        System.out.println("This is student class");
    }
    // Note that display() is only in Student class
    void display()
    {
        // will invoke or call current class message() method
        message();

        // will invoke or call parent class message() method
        super.message();
    }
}
/* Driver program to test */
class Test
{
    public static void main(String args[])
    {
        Student s = new Student();
        // calling display() of Student
        s.display();
    }
}
```

### Output:

This is student class  
This is person class

In the above example, we have seen that if we only call method message() then, the current class message() is invoked but with the use of super keyword, message() of superclass could also be invoked.

### Use of super with constructors:

super keyword can also be used to access the parent class constructor. One more important thing is that, "super" can call both parametric as well as non parametric constructors depending upon the situation. Following is the code snippet to explain the above concept:

#### Example:

```
/* superclass Person */
class Person
{
    Person()
    {
        System.out.println("Person class Constructor");
    }
}
/* subclass Student extending the Person class */
class Student extends Person
{
    Student()
    {
        // invoke or call parent class constructor
        super();
        System.out.println("Student class Constructor");
    }
}
/* Driver program to test*/
class Test
{
    public static void main(String[] args)
    {
        Student s = new Student();
    }
}
```

#### Output:

Person class Constructor

Student class Constructor

In the above example we have called the superclass constructor using keyword 'super' via subclass constructor.

### Other Important points:

Call to super() must be first statement in Derived(Student) Class constructor.

If a constructor does not explicitly invoke a superclass constructor, the Java compiler automatically inserts a call to the no-argument constructor of the superclass. If the

superclass does not have a no-argument constructor, you will get a compile-time error. Object does have such a constructor, so if Object is the only superclass, there is no problem.

If a subclass constructor invokes a constructor of its superclass, either explicitly or implicitly, you might think that a whole chain of constructors called, all the way back to the constructor of Object. This, in fact, is the case. It is called **constructor chaining**.

## Final Keyword

### *What is the Final Keyword in Java?*

In the Java programming language, the final keyword is used in several contexts to define an entity that can only be assigned once. The final keyword is used in different contexts. First of all, the final is a non-access modifier applicable only to a variable, a method, or a class. Following are the different context of using the final keyword:

The final keyword in java is used to restrict the user. In Java, the final keyword can be used while declaring an entity. Using the final keyword means that the value can't be modified in the future.

Final can be:

- Variables
- Methods
- Classes

## Final Variables in Java

When a variable is declared with the final keyword, its value can't be modified, essentially, a constant. This also means that you must initialize a final variable. The final variable cannot be reinitialized with another value. However, the data within the object can be changed. So, the state of the object can be changed but not the reference. With variables, the final modifier often is used with static to make the constant a class variable.

### Note:

The variable does not necessarily have to be initialized at the time of declaration. If it's declared but not yet initialized, it's called a blank final variable. This approach is the most common. It is recommended to use uppercase to declare final variables in Java.

### How to initialize a blank final variable?

Below are the two ways to initialize a blank final variable:

A blank final variable can be initialized inside the instance-initializer block or inside the constructor. If you have more than one constructor in your class then it must be initialized in all of them, otherwise, a compile-time error will be thrown.

A blank final static variable can be initialized inside a static block.

### When to use Final Variable?

Final Variables must be used only for the values that we want to remain constant throughout the execution of the program because a final variable is that we can re-assign value to a normal variable but we cannot change the value of a final variable once assigned.

**Example:**

```
public class FinalVariableDemo
{
    final int speedlimit = 90; //final variable
    void run ()
    {
        speedlimit = 400;
    }
    public static void main (String args[])
    {
        FinalVariableDemo obj = new FinalVariableDemo ();
        obj.run ();
    }
}
```

**Output:**

Compile-Time Error

There is a final variable speed limit, we are going to change the value of this variable, but it can't be changed because the final variable once assigned a value can never be changed.

## Final Methods in Java

A final method cannot be overridden or hidden by subclasses which means even though a subclass can call the final method of parent class without any issues but it cannot override it. This is used to prevent unexpected behavior from a subclass altering a method that may be crucial to the function or consistency of the class. We must declare methods with the final keyword for which we required to follow the same implementation throughout all the derived classes. The main intention of making a method final would be that the content of the method should not be changed by any outsider.

**Example:**

```
class Bike
{
    final void run()
    {
        System.out.println ("running");
    }
}
class Honda extends Bike
```

```
{  
    void run()  
    {  
        // COMPILE-ERROR! Can't override.  
        System.out.println ("running safely with 100kmph");  
    }  
  
    public static void main (String args[])  
    {  
        Honda honda = new Honda ();  
        honda.run();  
    }  
}
```

**Output:**

Compile-Time Error

**Final Class in Java**

In Java, the final class cannot be inherited by another class. The main purpose of using a class being declared as final is to prevent the class from being subclasses. If a class is marked as final then no class can inherit any feature from the final class. When an anonymous inner class is defined within the body of a method, all variables declared final in the scope of that method are accessible from within the inner class. For scalar values, once it has been assigned, the value of the final variable cannot change. For object values, the reference cannot change.

**Example:**

```
final class FinalClass  
{  
    // create a final method  
    public void display ()  
    {  
        System.out.println ("This is a final method.");  
    }  
}  
class Main extends FinalClass  
{  
    // try to override final method  
    public void display ()  
    {  
        System.out.println ("The final method is overridden.");  
    }  
    public static void main (String[]args)  
    {  
        Main obj = new Main ();  
        obj.display ();  
    }  
}
```

**Output:**

Compile-Time Error

In the above example, we have created a final class named Final class. Here, we have tried to inherit the final class from the Main class. When we run the program, we will get a compilation error

**Note:**

- The final method can be inherited but you cannot override it.
- The blank Final Variable can be initialized only in the constructor.
- Constructors cannot be declared as final because they cannot be inherited.

**Example:**

```
class Bike10{  
    final int speedlimit;//blank final variable  
  
    Bike10(){  
        speedlimit=70;  
        System.out.println(speedlimit);  
    }  
  
    public static void main(String args[]){  
        new Bike10();  
    }  
}
```

**Output:**

70

## 4.8 Exception Handling

### Error vs Exception

#### *What are Errors?*

The error signifies a situation that mostly happens due to the absence of system resources. The system crash and memory errors are an example of errors. It majorly occurs at runtime.

#### *What are Exceptions?*

The exceptions are the issues that can appear at runtime and compile time. It majorly arises in the code or program authored by the developers. There are two types of exceptions: Checked exceptions and Unchecked exceptions.

We have the following two types of errors:

- Compile Time Error
- Run Time Error

## Compile Time Error

Errors that occur at the time of compilation of the program are called compile-time errors. Compile-time errors occurred because if we don't follow the java syntaxes properly, java programming rules properly, etc. Compile-time errors are identified by the java compiler. So, in simple words, we can say that compile-time errors occur due to a poor understanding of the programming language. These errors can be identified by the programmer and can be rectified before the execution of the program only. So, these errors do not cause any harm to the program execution.

## Run Time Error

Errors that occur at the time of execution in the program are called runtime errors. Run-Time Errors are also called Exceptions. Exceptions may occur because programmer logic fails or JVM fails. Exceptions are identified by JVM.

**Note:** The Runtime errors are dangerous because whenever they occur in the program, the program terminates abnormally on the same line where the error gets occurred without executing the next line of code.

S.No	Errors	Exceptions
1.	The error indicates trouble that primarily occurs due to the scarcity of system resources.	The exceptions are the issues that can appear at runtime and compile time.
2.	It is not possible to recover from an error.	It is possible to recover from an exception.
3.	In java, all the errors are unchecked.	In java, the exceptions can be both checked and unchecked.
4.	The system in which the program is running is responsible for errors.	The code of the program is accountable for exceptions.
5.	They are described in the <code>java.lang.Error</code> package.	They are described in <code>java.lang.Exception</code> package

## Hierarchy of Java Exception classes

The `java.lang.Throwable` class is the root class of Java Exception hierarchy inherited by two subclasses: `Exception` and `Error`. The hierarchy of Java Exception classes is given below:

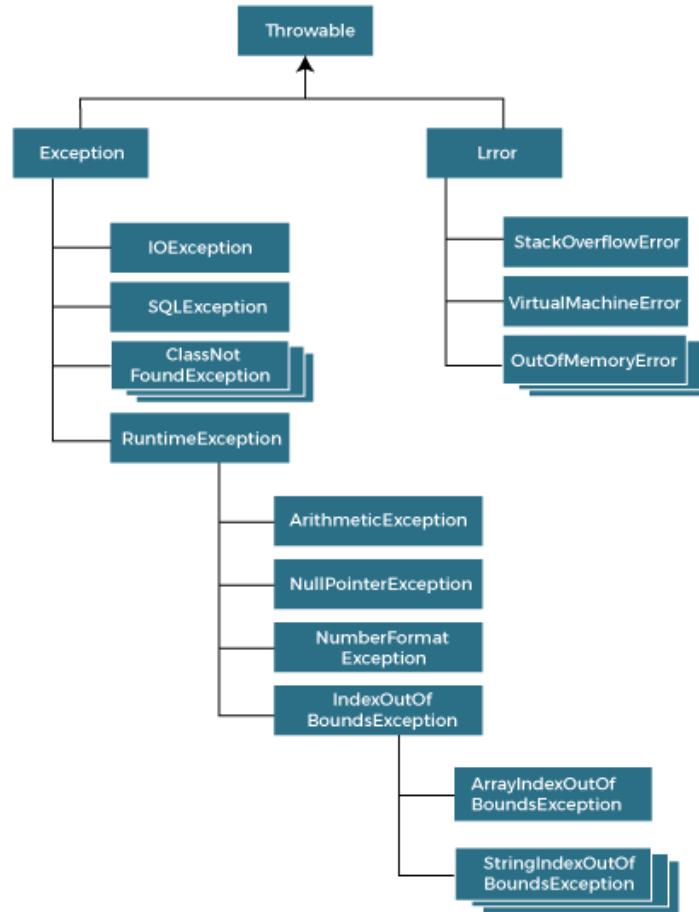


Image: Hierarchy of Java Exception

Reference: <https://static.javatpoint.com/core/images/hierarchy-of-exception-handling.png>

## Types of Java Exceptions

There are mainly two types of exceptions: checked and unchecked. An error is considered as the unchecked exception. However, according to Oracle, there are three types of exceptions namely:

- Checked Exception
- Unchecked Exception
- Error

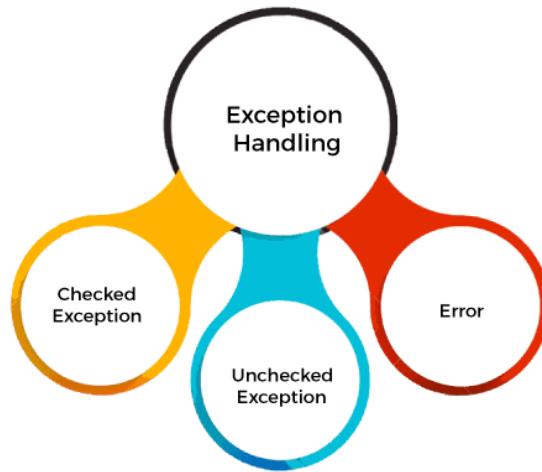


Image: Types of Java Exceptions

Reference: <https://static.javatpoint.com/core/images/hierarchy-of-exception-handling.png>

### Checked Exceptions in Java:

Exceptions that are identified at compilation time and occurred at runtime are called checked exceptions. These checked exceptions are also called Compile Time Exceptions. An exception said to be checked exception whose exception handling is mandatory as per the compiler. Example: IOException, ClassNotFoundException, CloneNotSupportedException, etc.

### Unchecked Exceptions in Java

Exceptions that are identified and occurred at run-time are called Unchecked Exceptions. These Unchecked Exceptions are also called Runtime Exceptions. An exception is said to be an unchecked exception whose exception handling is optional as per the compiler. Example: Arithmetic Exception, NumberFormatException, NoSuchElementException, etc.

**Note:** All child classes of Error and Runtime Exception classes are called the unchecked exception and the remaining classes are called checked exceptions.  
In Java, Exception Handling can be done by using five Java keywords:

- Try
- Catch
- Finally
- Throw
- throws

#### try-block:

- The code which might raise exception must be enclosed within try-block
- try-block must be followed by either catch-block or finally-block, at the end
- If both present, it is still valid but the sequence of the try-catch-finally matters the most

- Otherwise, compile-time error will be thrown for invalid sequence
- The valid combination like try-catch block or try-catch-finally blocks must reside inside method
- **Note:** code inside try-block must always be wrapped inside curly braces, even if it contains just one line of code;
- Otherwise, compile-time error will be thrown
- Compile-time Error : “Syntax error on token “)”, Block expected after this token”

**catch-block:**

- Contains handling code for any exception raised from corresponding try-block and it must be enclosed within catch block
- catch-block takes one argument which should be of type Throwable or one of its sub-classes i.e.; class-name followed by a variable
- Variable contains exception information for exception raised from try-block
- Note: code inside catch-block must always be wrapped inside curly braces, even if it contains just one line of code;
- Otherwise, compile-time error will be thrown
- Compile-time Error: “Syntax error on token “)”, Block expected after this token”

**finally-block:**

- finally block is used to perform clean-up activities or code clean-up like closing database connection & closing streams or file resources, etc
- finally block is always associated with try-catch block
- With finally-block, there can be 2 combinations
- One is try-block is followed by finally-block and other is try-catch-finally sequence
- The only other possible combination is try block followed by multiple catch block and one finally block at the end (this is case of multiple catch blocks)
- Advantage: The beauty of finally block is that, it is executed irrespective of whether exception is thrown or NOT (from try-block)
- Also, it gets executed whether respective exception is handled or NOT (inside catch-block)

**Note:** finally block won't get executed if JVM exits with System.exit() or due to some fatal error like code is interrupted or killed

**throw clause:**

- Sometimes, programmer can also throw/raise exception explicitly at runtime on the basis of some business condition
- To raise such exception explicitly during program execution, we need to use throw keyword

**Syntax: throw instanceOfThrowableType**

- Generally, throw keyword is used to throw user-defined exception or custom exception

- Although, it is perfectly valid to throw pre-defined exception or already defined exception in Java like IOException, NullPointerException, ArithmeticException, InterruptedException, ArrayIndexOutOfBoundsException, etc.

```
try {  
    // some valid Java statements  
    throw new RuntimeException();  
}  
catch(Throwable th) {  
  
    // handle exception here  
    // or re-throw caught exception  
}
```

### **throws keyword or throws clause:**

- throws keyword is used to declare the exception that might raise during program execution
- whenever exception might thrown from program, then programmer doesn't necessarily need to handle that exception using try-catch block instead simply declare that exception using throws clause next to method signature
- But this forces or tells the caller method to handle that exception; but again caller can handle that exception using try-catch block or re-declare those exception with throws clause

**Note:** use of throws clause doesn't necessarily mean that program will terminate normally rather it is the information to the caller to handle for normal termination

- Any number of exceptions can be specified using throws clause, but they are all need to be separated by commas (,)
- throws clause is applicable for methods & constructor but strictly not applicable to classes
- It is mainly used for checked exception, as unchecked exception by default propagated back to the caller (i.e.; up in the runtime stack)

**Note:** It is highly recommended to use try-catch for exception handling instead of throwing exception using throws clause

### **Syntax:**

```
try  
{  
    //code to be monitored for an exception  
    throw exception_object  
    //throw an exception object to the Java Runtime  
}  
catch(exception_object)  
{  
    //exception handler for catching the exception_object  
}  
finally  
{
```

```
//The code will get executed in any case
}
```

## try...catch block:

### try

The statements susceptible to a runtime error fall under the try block. A try block can't exist independently. It has to accompany by either catch or finally block.

### catch

The catch block contains the statements of action to be taken when an exception occurs in the try block. Catch used without a try block leads to a compile-time error.

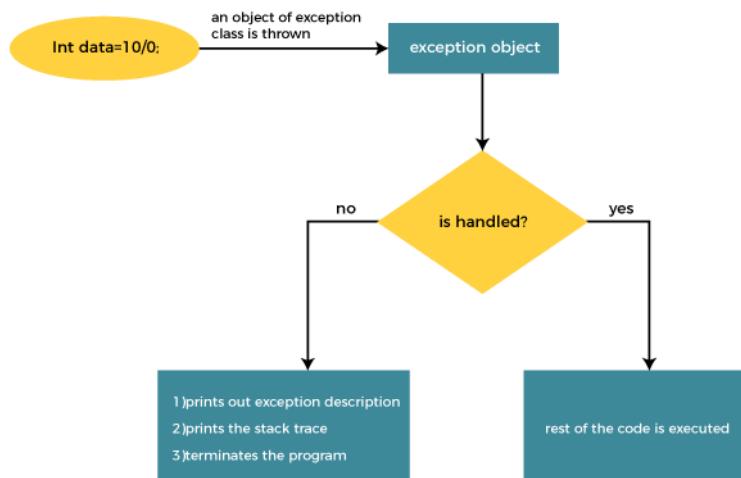


Image: try block  
 Reference: <https://www.javatpoint.com/try-catch-block>

### Example:

```

class Main_class {
    public static void main(String[] args) {
        int var1 = 32;
        double result;
        try {
            result = var1/0;
            System.out.println("Result of division: " +result);
        }
        catch (ArithmaticException e)
        {
            System.out.println("Exception caught! " +e );
        }
    }
}
  
```

## Output:

Exception caught! java.lang.ArithmaticException: / by zero

**Explanation:** The moment an exception occurs in the try block, the flow of execution jumps to the matching catch block. For the time, all the statements in between are skipped, and statements under the catch block get executed. Afterward, the control goes back to the try block and continues normal execution.

## Multiple catch block

A try block can be followed by one or more catch blocks. Each catch block must contain a different exception handler. So, if you have to perform different tasks at the occurrence of different exceptions, use java multi-catch block.

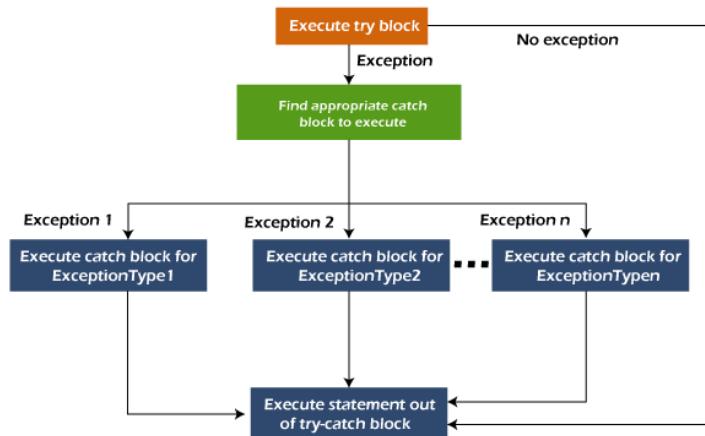


Image: Multiple catch block  
 Reference: <https://www.javatpoint.com/multiple-catch-block-in-java>

## Example

```

public class MultipleCatchBlock1 {
    public static void main(String[] args) {
        try{
            int a[]={};int b=5;
            a[5]=30/b;
        }
        catch(ArithmaticException e)
        {
            System.out.println("Arithmatic Exception occurs");
        }
        catch(ArrayIndexOutOfBoundsException e)
        {
            System.out.println("ArrayIndexOutOfBoundsException occurs");
        }
        catch(Exception e)
        {
    
```

```

        System.out.println("Parent Exception occurs");
    }
    System.out.println("rest of the code");
}
}

```

## Output

Arithmetic Exception occurs  
rest of the code

### **finally block:**

This block contains a set of statements that executes whether or not an exception is thrown.

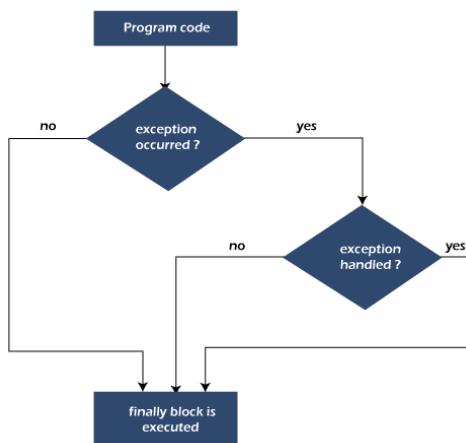


Image: finally block  
Reference: <https://www.javatpoint.com/finally-block-in-exception-handling>

### Why use Java finally block?

finally block in Java can be used to put "cleanup" code such as closing a file, closing connection, etc. The important statements to be printed can be placed in the finally block.

### **throw and throws Keyword**

**throw:** The throw keyword utilizes to throw the exception explicitly.

### **Example:**

```

public class ThrowExample{

    void Votingage(int age){
        if(age<18)
            throw new ArithmeticException("you can't vote as not Eligible to vote");
        else
            System.out.println("Eligible for voting");
    }
}

```

```
}

public static void main(String args[]){
    ThrowExample obj = new ThrowExample();
    obj.Votingage(13);
    System.out.println("End Of Program");
}
}
```

**Output:**

Exception in thread "main" java.lang.ArithmeticException: you can't vote as not Eligible to vote at ThrowExample.Votingage(ThrowExample.java:5) at ThrowExample.main(ThrowExample.java:11)

**throws:** This keyword is used with the method prototype which indicates the type of exceptions that the method might throw to the java runtime.

**Example:**

```
public class ThrowsExample{
    int divion(int a, int b) throws ArithmeticException{
        int intet = a/b;
        return intet;
    }
    public static void main(String args[]){
        ThrowsExample obj = new ThrowsExample();
        try{
            System.out.println(obj.divion(15,0));
        }
        catch(ArithmaticException e){
            System.out.println("Division cannot be done using ZERO");
        } }
```

**Output:**

Division cannot be done using ZERO

## 4.9 Additional Topics

### Java Scanner Class

In Java, we can input with the help of the Scanner class. Java has several predefined classes which we can use. We will learn more about the classes later. The predefined classes are organized in the form of packages. Java Scanner class is found in the java.util package.

#### Java Scanner Class Declaration

```
public final class Scanner
    extends Object
    implements Iterator<String>
```

If we need to import a class or a package, add one of the following lines to the very beginning of your code.

```
import java.util.Scanner; // This will import just the Scanner class
import java.util.*; // This will import the entire java.util package
```

We can use either of the above lines. The first line only imports the Scanner class and the second line imports the whole java.util package. After importing the class, we need to write the following statement in our program that will create an object of Scanner class.

**Scanner s = new Scanner (System.in);**

At this moment writing Scanner s, we are declaring s as an object of Scanner class. System.in within the round brackets tells Java that this will be System Input, i.e., input will be given to the system.

### **Example:**

```
// Java program to read data of various types using Scanner class.
import java.util.Scanner;
public class ScannerDemo1
{
    public static void main(String[] args)
    {
        // Declare the object and initialize with
        // predefined standard input object
        Scanner sc = new Scanner(System.in);

        // String input
        String name = sc.nextLine();

        // Character input
        char gender = sc.next().charAt(0);

        // Numerical data input
        // byte, short and float can be read
        // using similar-named functions.
        int age = sc.nextInt();
        long mobileNo = sc.nextLong();
        double cgpa = sc.nextDouble();

        // Print the values to check if the input was correctly obtained.
        System.out.println("Name: "+name);
        System.out.println("Gender: "+gender);
        System.out.println("Age: "+age);
```

```
        System.out.println("Mobile Number: "+mobileNo);
        System.out.println("CGPA: "+cgpa);
    }
}
```

**Output:**

Harry  
M  
25  
9812342420  
9.7  
Name: Harry  
Gender: M  
Age: 25  
Mobile Number: 9812342420  
CGPA: 9.7

## Wrapper Class

### *What are Wrapper Classes in Java?*

Wrapper classes are Java predefined classes that are responsible to convert the given string type the numerical value into equivalent primitive data type and vice-versa. A wrapper class is bundled default with the Java library and it is located in (jre/lib/rt.jar file). When we create an object to a wrapper class, it contains a field and in this field, we can store primitive data types

The wrapper class in Java provides the mechanism to convert primitive into object and object into primitive.

Primitive Types	Wrapper class
boolean	Boolean
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double

### 8 Wrapper Classes in Java

Image: Wrapper Classes in Java  
Reference:<https://4.bp.blogspot.com/>

### Need for Wrapper Classes:

- The wrapper objects hold much more memory compared to primitive types.
- Wrapper Class will convert primitive data types into objects. The object is needed to support synchronization in multithreading.
- Wrapper class objects allow null values while primitive data type doesn't allow it.

### Example:

```
public class wrapperdemo
{
    public static void main (String[]args)
    {
        String i = "10";
        int k = Integer.parseInt (i);
        double m = Double.parseDouble ("10");
        System.out.println (k);
        System.out.println (m);
        boolean status = Boolean.parseBoolean ("20");
        System.out.println (status);
    }
}
```

**Output:**

10  
10.0  
false

**How to Implement a Wrapper Class in Java?**

Wrapper class can be implemented in Java in the following two ways:

- Autoboxing
- Unboxing

In general, autoboxing and unboxing take place whenever a conversion into an object or from an object is required. Thus, autoboxing/ unboxing might occur when an argument is passed to a method, or when a value is returned by a method.

**Autoboxing in Java:**

“Boxing” refers to converting a primitive value into a corresponding wrapper object. Because this can happen automatically, it’s known as autoboxing. In addition to the simple case of assignments, autoboxing automatically occurs whenever a primitive type must be converted into an object. With autoboxing, it is no longer necessary to manually construct an object in order to wrap a primitive type. You need only assign that value to a type-wrapper reference. Java automatically constructs the object for you.

**Example:**

```
public class AutoBoxing
{
    public static void main (String args[])
    {
        int a = 100; // Primitive data type
        Integer I = a; // Autoboxing will occur internally.
    }
}
```

}

### Example: Primitive to Wrapper

```
//Java program to convert primitive into objects
//Autoboxing example of int to Integer
public class WrapperExample1{
public static void main(String args[]){
//Converting int into Integer
int a=20;
Integer i=Integer.valueOf(a);//converting int into Integer explicitly
Integer j=a;//autoboxing, now compiler will write Integer.valueOf(a) internally

System.out.println(a+" "+i+" "+j);
}}
```

#### Output:

20 20 20

### Unboxing in Java:

Auto-unboxing takes place whenever an object must be converted into a primitive type. Auto-unboxing is the process by which the value of a boxed object is automatically extracted (unboxed) from a type wrapper when its value is needed. There is no need to call a method such as intValue( ) or doubleValue( ).

#### Example:

```
public class AutoBoxing
{
    public static void main (String args[])
    {
        Integer a = new Integer (15); // Wrapper class object
        int l = a; // Unboxing will occur internally.
    }
}
```

### Example: Wrapper to Primitive

```
//Java program to convert object into primitives
//Unboxing example of Integer to int
public class WrapperExample2{
public static void main(String args[]){
//Converting Integer to int
Integer a=new Integer(3);
int i=a.intValue();//converting Integer to int explicitly
int j=a;//unboxing, now compiler will write a.intValue() internally

System.out.println(a+" "+i+" "+j);
}}
```

**Output:**

3 3 3

**Need for Autoboxing and Unboxing in Java:**

- The addition of autoboxing and auto-unboxing greatly streamlines the coding of several algorithms, removing the tedium of manually boxing and unboxing values.
- It also helps prevent errors.
- Moreover, it is very important to generics, which operate only on objects.

## Type Casting

### *What is Type Casting in Java?*

The process of converting the value of one data type (int, float, double, etc.) into another data type is known as Type Casting. Type Casting is the temporary conversion of a variable from its original data type to some other data type, like being cast for a part in a play or movie.

With primitive data types if a cast is necessary from a less inclusive data type to a more inclusive data type it is done automatically. If a cast is necessary from a more inclusive data type to a less inclusive data type the cast must be done explicitly by the programmer.

### Conversion between Primitive Data Types: Cast Operator

The Cast Operator manually lets you convert a value of the data type into another data type. It can be done by Writing the cast operator in front of the expression that needs to be converted.

**Syntax:** (dataTypeName) expression;

When casting from a double to an int, the fractional part will be discarded.

### **Example:**

```
int x;  
double y = 2.5;  
x = (int)y;
```

Here, int is the **Cast Operator**.

Java compiles the code with the cast operator with no problems. In this case, the variable y has a value of 2.5 (the floating-point value) which must be converted to an integer. The value that is returned and stored in variable x would be truncated, which means the fractional part of the number is lost to accommodate the integer data type. Thus, x=2 and, the value of the variable y is not changed at all: y=2.5

## Types of Type Casting in Java:

- Widening or Automatic Type Casting
- Narrowing or Explicit type Casting

### Widening Type Casting in Java:

Widening Type Casting is also known as Automatic Type Casting. When you assign the value of one data type to another, the two types might not be compatible with each other. If the data types are compatible, then Java will perform the conversion automatically known as Automatic Type Conversion.

- Widening Casting(Implicit)



- Narrowing Casting(Explicitly done)



Image: Types of Typecasting

Reference: <https://www.datasciencecentral.com/wp-content/uploads/2021/10/2808315407.png>

### Example: Widening or Automatic Type Casting in Java:

```
package Demo;
public class AutomaticTypeConversion
{   public static void main (String args[])
{
    int intVariable = 100;
    long longVariable = intVariable;
    float floatVariable = longVariable;
    System.out.println ("Integer Value is : " + intVariable);
    System.out.println ("Float Value is : " + floatVariable);
    System.out.println ("Long Value is : " + longVariable);
}}
```

#### Output:

Integer Value is : 100  
Float Value is : 100.0  
Long Value is: 100

### Narrowing Type Casting in Java:

Narrowing Type Casting is also known as Explicit type Casting. This is useful for incompatible data types where automatic conversion cannot be done. It happens when the two data types are not compatible and when we want to assign a value of a larger data type to a smaller data type.

### **Example:Narrowing or Explicit Type Casting in Java:**

```
package Demo;
public class ExplicitTypeCasting
{
    public static void main (String[]args)
    {
        double doubleVariable = 100.04;
        long longVariable = (long) doubleVariable;
        int intVariable = (int) longVariable;
        System.out.println ("Double Value is : " + doubleVariable);
        System.out.println ("Long Value is : " + longVariable);
        System.out.println ("Integer Value is : " + intVariable);
    }
}
```

#### **Output:**

Double Value is: 100.04  
Long Value is: 100  
Integer Value is: 100

### **Command Line Arguments**

The java command-line argument is an argument i.e., passed at the time of running the java program. The arguments passed from the console can be received in the java program and it can be used as an input.  
So, it provides a convenient way to check the behavior of the program for the different values. You can pass N (1,2,3 and so on) numbers of arguments from the command prompt.

#### **Example:**

```
class CommandLineExample{
public static void main(String args[]){
System.out.println("Your first argument is: "+args[0]);
}
}
```

**compile by >** javac CommandLineExample.java  
**run by >** java CommandLineExample Harry

#### **Output:**

Your first argument is: Harry

### **Example: command-line argument that prints all the values**

We are printing all the arguments passed from the command-line. For this purpose, we have traversed the array using for loop.

```
class A{  
    public static void main(String args[]){  
        for(int i=0;i<args.length;i++)  
            System.out.println(args[i]);  
  
    }  
}  
  
compile by > javac A.java  
run by > java A Harry Eliza 1 3 abc
```

#### Output:

Harry  
Eliza  
1  
3  
abc

## Recursion

### *What Is Recursion in Java?*

Recursion is a process by which a function or a method calls itself again and again. This function that is called again and again either directly or indirectly is called the “recursive”

#### Recursion Syntax:

Any method that implements Recursion has two basic parts:

- Method **calls** which can call itself i.e., recursive
- A precondition that will stop the recursion.

Note that a precondition is necessary for any recursive method as, if we do not break the recursion then it will keep on running infinitely and result in a stack overflow.

#### The general syntax of recursion is as follows:

```
methodName (T parameters...)  
{  
    if (precondition == true)  
    //precondition or base condition  
    {  
        return result;  
    }  
    return methodName (T parameters...);  
    //recursive call
```

}

Note that the precondition is also called base condition

## Understanding Recursion in Java

In this section, we will try to understand the recursion process and see how it takes place. We will learn about the base condition, stack overflow, and see how a particular problem can be solved with recursion and other such details.

### Recursion Base Condition

While writing the recursive program, we should first provide the solution for the base case. Then we express the bigger problem in terms of smaller problems. As an example, we can take a classic problem of calculating the factorial of a number. Given a number n, we have to find a factorial of n denoted by n!. Now let's implement the program to calculate the n factorial (n!) using recursion.

#### Example:

```
public class Main{  
    static int fact(int n)  
    {  
        if (n == 1)  
        // base condition  
            return 1;  
        else  
            return n*fact(n-1);  
    }  
    public static void main(String[] args) {  
        int result = fact(5);  
        System.out.println("5! = " + result);  
    }  
}
```

#### Output:

5! =120

### Recursion base condition

In this program, we can see that the condition ( $n \leq 1$ ) is the base condition and when this condition is reached, the function returns 1. The else part of the function is the recursive call. But every time the recursive method is called, n is decremented by 1. **Thus**, we can conclude that ultimately the value of n will become 1 or less than 1 and at this point, the method will return value 1. This base condition will be reached and the function will stop. Note that the value of n can be anything as long as it satisfies the base condition.

## Problem-Solving Using Recursion

The basic idea behind using recursion is to express the bigger problem in terms of smaller problems. Also, we need to add one or more base conditions so that we can come out of recursion.

This was already demonstrated in the above factorial example. In this program, we expressed the n factorial ( $n!$ ) in terms of smaller values and had a base condition ( $n \leq 1$ ) so that when n reaches 1, we can quit the recursive method.

### Fibonacci Series Using Recursion

The Fibonacci series is given by,

1,1,2,3,5,8,13,21,34,55,...

The above sequence shows that the current element is the sum of the previous two elements. Also, the first element in the Fibonacci series is 1.

So in general if n is the current number, then it is given by the sum of (n-1) and (n-2). As the current element is expressed in terms of previous elements, we can express this problem using recursion.

### Implement the Fibonacci series:

```
public class Main
{
    //method to calculate fibonacci series
    static int fibonacci(int n) {
        if (n <= 1) {
            return n;
        }
        return fibonacci(n-1) + fibonacci(n-2);
    }
    public static void main(String[] args) {
        int number = 5;
        //print first 5 numbers of fibonacci series
        System.out.println ("Fibonacci Series: First 10 numbers:");
        for (int i = 1; i <= number; i++)
        {
            System.out.print(fibonacci(i) + " ");
        }
    }
}
```

### Output:

Fibonacci Series: First 10 numbers:

1 1 2 3 5 8 13 21 34 55

We discussed in this module Java, which is one of the best programming language due to its small language vocabulary, portability and simplicity. It is mostly used by web developers and is considered flexible, due to its object-oriented features. Java has exception-handling that requires a programmer to handle error-conditions such as Input/Output errors. Code compiled on one Java platform can be run on other platforms that support Java without modification of either the source-code nor the byte-code. This means that a person can make a Java program for a Windows computer

and have it run a Linux computer or a Mac computer. This makes Java platform independent.

# Chapter 5: Understanding the Cloud Computing

## Learning Outcomes:

- Introduction to Cloud Computing as a domain.
- Learning about Microsoft Azure Portal and offered services.
- Working knowledge of networking protocols and integration to Azure services.
- Real time implementation of services like virtual machines and virtual networks.
- Understand the future prospects with Cloud Computing.

## 5.1 Introduction to cloud computing

We live in the world of internet. Internet or internet like networks are rapidly becoming part of our personal, professional and social lives. In 20<sup>th</sup> century, we have entered industrial revolution 4.0 (IR 4.0). The key technologies in IR 4.0 are Artificial Intelligence, Internet of Things (IoT), web technologies, cyber security, blockchain, data analytics, Big Data, and so on. Cloud Computing is a technology that acts as backbone for harnessing the power of all these technologies.

In simple words, cloud computing means providing software and hardware services over some network, mostly internet. These software and hardware services include compute services, storage, databases, and network services. All these services can be scaled up or down, as per user requirement, and are available on demand over some network.



Image: What is Cloud Computing?

Reference: <https://www.thinkitsolutions.com/wp-content/uploads/2019/04/what-is-cloud-computing-with-example-300x187.png>

For different people, cloud computing could mean different things.

1. For a professional who is working remotely, cloud computing is able to access work resources from a remote location, using internet.
2. When we are using email services, watching shows and movies on Netflix or amazon prime, or editing some content online on a shared drive, we are in fact using the software services provided by some cloud service provider.
3. Software developers also use cloud services to write their code, deploy it online, and debug/update it. Cloud services can automate the software development lifecycle.
4. A businessman, who wants to open his online retail store, can hire cloud services for his website development, storage servers and other IT infrastructure. This enables him to concentrate on business logic and marketing strategies, and leave the technical part onto a cloud service provider, who will charge him for the services and resources used for the e-business.

There are number of cloud service provider available in the market that are helping us in utilizing the on-demand cloud components according to our requirements. One such prominent cloud service provider is Microsoft Azure. In the next segment we will be discussing about the same.

## 5.2 Introduction to Microsoft Azure Cloud Platform

Microsoft Azure is a cloud platform from Microsoft. It is a continually expanding set of cloud services that help your organization meet your current and future business challenges. Azure gives you the freedom to build, manage, and deploy applications on a massive global network using your favorite tools and frameworks.

### ***Why chose Microsoft Azure?***

Microsoft Azure provides a variety of advantages over an on-premise system. These benefits include:

- **High availability:** Depending on the service-level agreement (SLA) that you choose, your cloud-based apps can provide a continuous user experience with no apparent downtime, even when things go wrong.
- **Scalability:** Apps in the cloud can scale *vertically* and *horizontally*:
  - Scale vertically to increase compute capacity by adding RAM or CPUs to a virtual machine.
  - Scaling horizontally increases compute capacity by adding instances of resources, such as adding VMs to the configuration.
- **Elasticity:** You can configure cloud-based apps to take advantage of autoscaling, so your apps always have the resources they need.
- **Agility:** Deploy and configure cloud-based resources quickly as your app requirements change.

- **Geo-distribution:** You can deploy apps and data to regional datacenters around the globe, thereby ensuring that your customers always have the best performance in their region.
- **Disaster recovery:** By taking advantage of cloud-based backup services, data replication, and geo-distribution, you can deploy your apps with the confidence that comes from knowing that your data is safe in the event of disaster.

### ***What can you do with Microsoft Azure?***

Azure provides more than 100 services that enable you to do everything from running your existing applications on virtual machines, to exploring new software paradigms, such as intelligent bots and mixed reality.

Many teams start exploring the cloud by moving their existing applications to virtual machines that run in Azure. Migrating your existing apps to virtual machines is a good start, but the cloud is much more than a different place to run your virtual machines.

### ***What is the Azure Portal?***

- The Azure portal is a web-based, unified console that provides an alternative to command-line tools. With the Azure portal, you can manage your Azure subscription by using a graphical user interface.
- The Azure portal is designed for resiliency and continuous availability. It maintains a presence in every Azure datacenter. This configuration makes the Azure portal resilient to individual datacenter failures and avoids network slowdowns by being close to users. The Azure portal updates continuously and requires no downtime for maintenance activities.
- Using Azure Portal, You can:
  - Build, manage, and monitor everything from simple web apps to complex cloud deployments.
  - Create custom dashboards for an organized view of resources.
  - Configure accessibility options for an optimal experience.

### ***What is the Azure Marketplace?***

Azure Marketplace helps connect users with Microsoft partners, independent software vendors, and start-ups that are offering their solutions and services, which are optimized to run on Azure. Azure Marketplace customers can find, try, purchase, and provision applications and services from hundreds of leading service providers. All solutions and services are certified to run on Azure.

### ***Azure Command Line Tools***

Azure command-line tools automate routine operations, standardize database failovers, and pull data that provide powerful insight. Command line tools gives you the ability to scale your tasks in Azure and also make sharing, an easier task. Because sharing a script is easy as compared to lengthy pages.

We will discuss each of these topics in the upcoming subsections of this chapter.

## Cloud Deployment Models and Service Delivery Models according to Azure

Cloud is the future of computing. It is about outsourcing of IT services and infrastructure to make them accessible remotely via the Internet. Utilizing cloud-computing models boosts not only productivity but also provide a competitive edge to organizations. The growing popularity of cloud computing has given rise to different types of cloud service deployment models and strategies. Therefore, today there exists a variety of enterprise cloud solutions depending on the degree of desired outsourcing needs.

It is along with their customization flexibility, control, and data management within the organization. Further, it involves the pooling of specialized human and technical resources to effectively manage existing systems and applications as it helps in meeting the requirements of organizations and users.

### *Different Types of Cloud Computing Deployment Models*

Most cloud hubs have tens of thousands of servers and storage devices to enable fast loading. It is often possible to choose a geographic area to put the data “closer” to users. Thus, deployment models of cloud computing are categorized based on their location. To know which deployment model would best fit the requirements of your organization, let us first learn about the types of cloud deployment models.

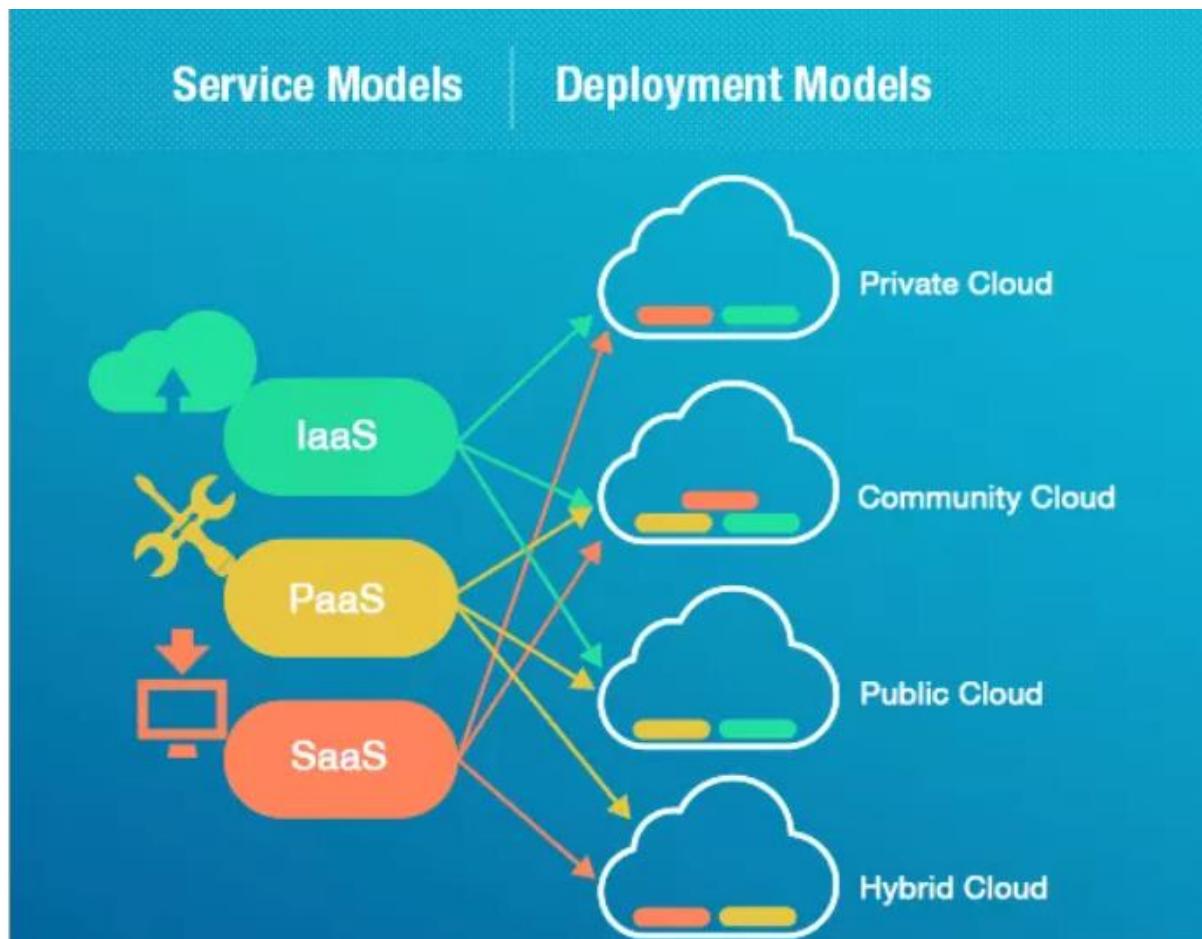


Image: Cloud Service Models

Reference: <https://www.oreilly.com/library/view/the-enterprise-cloud/>

### *Public Cloud*

This type of cloud services is provided on a network for public use. Customers have no control over the location of the infrastructure. It is based on a shared cost model for all the users, or in the form of a licensing policy such as pay per user. Public deployment models in the cloud are perfect for organizations with growing and fluctuating demands. It is also popular among businesses of all sizes for their web applications, webmail, and storage of non-sensitive data.

Azure provides a plethora of services which are deployed in the form of public cloud. User can create their own resources deployed as public cloud models. However, the services will be in the form of multitenancy, since in public cloud models, resources are allocated from a pool of resources. Public clouds can save companies from the expensive costs of having to purchase, manage, and maintain on-premises hardware and application infrastructure.

### *Private Cloud*

It is a cloud-based infrastructure used by stand-alone organizations. It offers greater control over security. The data is backed up by a firewall and internally, and can be hosted internally or externally. Private clouds are perfect for organizations that have

high-security requirements, high management demands, and availability requirements. One drawback is that the company's IT department is held responsible for the cost and accountability of managing the private cloud. So private clouds require the same staffing, management, and maintenance expenses as traditional datacenter ownership.

### ***Community Cloud***

It is a mutually shared model between organizations that belong to a particular community such as banks, government organizations, or commercial enterprises. Community members generally share similar issues of privacy, performance, and security. This type of deployment model of cloud computing is managed and hosted internally or by a third-party vendor.

### ***Hybrid Cloud***

This model incorporates the best of both private and public clouds, but each can remain as separate entities. Further, as part of this deployment of cloud computing model, the internal, or external providers can provide resources. A hybrid cloud is ideal for scalability, flexibility, and security. A perfect example of this scenario would be that of an organization who uses the private cloud to secure their data and interacts with its customers using the public cloud.

For example, let's say you are a CTO of a company. Your company maintains all resources of their own. For a new project you need some expensive resources but for a very short time. You are also worried that the quantities of the resources might increase or decrease any day. You don't want any unnecessary expense. Here, hybrid cloud model can ease your concerns. Instead of buying new resources, you can rent these from cloud service providers such as Microsoft Azure.

### ***Cloud Service Delivery Models***

There are the following three types of cloud service models –

1. [Infrastructure as a Service \(IaaS\)](#)
2. [Platform as a Service \(PaaS\)](#)
3. [Software as a Service \(SaaS\)](#)

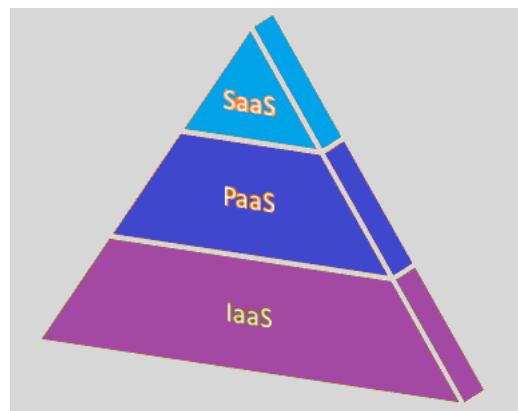


Image: Cloud Service Delivery Models

## 1. Infrastructure as a Service (IaaS)

IaaS is also known as **Hardware as a Service (HaaS)**. It is a computing infrastructure managed over the internet. The main advantage of using IaaS is that it helps users to avoid the cost and complexity of purchasing and managing the physical servers.

### Characteristics of IaaS

There are the following characteristics of IaaS –

- Resources are available as a service
- Services are highly scalable
- Dynamic and flexible
- GUI and API-based access
- Automated administrative tasks

**Example:** Microsoft Azure, DigitalOcean, Linode, Amazon Web Services (AWS), Google Compute Engine (GCE), Rackspace, and Cisco Metacloud.

IaaS is offered in three models: public, private, and hybrid cloud. The private cloud implies that the infrastructure resides at the customer-premise. In the case of public cloud, it is located at the cloud computing platform vendor's data center, and the hybrid cloud is a combination of the two in which the customer selects the best of both public cloud or private cloud.

IaaS provider provides the following services –

- **Compute:** Computing as a Service includes virtual central processing units and virtual main memory for the Vms that is provisioned to the end- users.
- **Storage:** IaaS provider provides back-end storage for storing files.
- **Network:** Network as a Service (NaaS) provides networking components such as routers, switches, and bridges for the Vms.
- **Load balancers:** It provides load balancing capability at the infrastructure layer.



Image: Cloud IaaS model components

Reference: <https://encrypted-tbn0.gstatic.com/images>

### ***Advantages of IaaS cloud computing layer***

There are the following advantages of IaaS computing layer -

- **Shared infrastructure:** IaaS allows multiple users to share the same physical infrastructure.
- **Web access to the resources:** IaaS allows IT users to access resources over the internet.
- **Pay-as-per-use model:** IaaS providers provide services based on the pay-as-per-use basis. The users are required to pay for what they have used.
- **Focus on the core business:** IaaS providers focus on the organization's core business rather than on IT infrastructure.
- **On-demand scalability:** On-demand scalability is one of the biggest advantages of IaaS. Using IaaS, users do not worry about upgrading software and troubleshoot the issues related to hardware components.

### ***Disadvantages of IaaS cloud computing layer***

- **Security:** Security is one of the biggest issues in IaaS. Most of the IaaS providers are not able to provide 100% security.
- **Maintenance & Upgrade:** Although IaaS service providers maintain the software, but they do not upgrade the software for some organizations.
- **Interoperability issues:** It is difficult to migrate VM from one IaaS provider to the other, so the customers might face problems related to vendor lock-in.

### ***Some important point about IaaS cloud computing layer.***

IaaS cloud computing platform cannot replace the traditional hosting method, but it provides more than that, and each resource which are used are predictable as per the usage.

IaaS may not eliminate the need for an in-house IT department. It will be needed to monitor or control the IaaS setup. IT salary expenditure might not reduce significantly, but other IT expenses can be reduced. Breakdowns at the IaaS cloud computing platform vendor's can bring your business to the halt stage. Assess the IaaS cloud computing platform vendor's stability and finances. Make sure that SLAs (i.e., Service Level Agreement) provide backups for data, hardware, network, and application failures. Image portability and third-party support is a plus point. The IaaS cloud computing platform vendor can get access to your sensitive data. So, engage with credible companies or organizations. Study their security policies and precautions.

### **Services provided in Azure as IaaS**

Azure provides a variety of IaaS resources. Such as Compute, Storage, Networking, Azure Arc, Azure Stack HCI, Azure Stack Edge, Azure Stack Hub, Azure IoT, Azure Virtual Machines, Azure Kubernetes Service (AKS), Azure Red Hat OpenShift and many more.

### ***Platform as a Service (PaaS)***

PaaS cloud computing platform is created for the programmer to develop, test, run, and manage the applications.

### **Characteristics of PaaS**

There are the following characteristics of PaaS –

- Accessible to various users via the same development application.
- Integrates with web services and databases.
- Builds on virtualization technology, so resources can easily be scaled up or down as per the organization's need.
- Supports multiple languages and frameworks.
- Provides an ability to "**Auto-scale**".

**Example:** AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos, Magento Commerce Cloud, and OpenShift.

### **Platform as a Service / PaaS**

Platform as a Service (PaaS) provides a runtime environment. It allows programmers to easily create, test, run, and deploy web applications. You can purchase these applications from a cloud service provider on a pay-as-per use basis and access them using the Internet connection. In PaaS, back-end scalability is managed by the cloud service provider, so end-users do not need to worry about managing the infrastructure. PaaS includes infrastructure (servers, storage, and networking) and platform (middleware, development tools, database management systems, business intelligence, and more) to support the web application life cycle.

**Example:** Azure App Service, Google App Engine, Force.com, Joyent.

PaaS providers provide the Programming languages, Application frameworks, Databases, and other tools:



Image: Cloud PaaS Components

Reference: <https://encrypted-tbn0.gstatic.com/images>

**1. Programming languages:** PaaS providers provide various programming languages for the developers to develop the applications. Some popular programming languages provided by PaaS providers are Java, PHP, Ruby, Perl, and Go.

**2. Application frameworks:** PaaS providers provide application frameworks to easily understand the application development. Some popular application frameworks provided by PaaS providers are Node.js, Drupal, Joomla, WordPress, Spring, Play, Rack, and Zend.

**3. Databases:** PaaS providers provide various databases such as ClearDB, PostgreSQL, MongoDB, and Redis to communicate with the applications.

**4. Other tools:** PaaS providers provide various other tools that are required to develop, test, and deploy the applications.

### **Advantages of PaaS**

There are the following advantages of PaaS -

**1. Simplified Development:** PaaS allows developers to focus on development and innovation without worrying about infrastructure management.

**2. Lower risk:** No need for up-front investment in hardware and software. Developers only need a PC and an internet connection to start building applications.

**3. Prebuilt business functionality:** Some PaaS vendors also provide already defined business functionality so that users can avoid building everything from very scratch and hence can directly start the projects only.

**4. Instant community:** PaaS vendors frequently provide online communities where the developer can get the ideas to share experiences and seek advice from others.

**5. Scalability:** Applications deployed can scale from one to thousands of users without any changes to the applications.

### **Disadvantages of PaaS cloud computing layer**

**1. Vendor lock-in:** One has to write the applications according to the platform provided by the PaaS vendor, so the migration of an application to another PaaS vendor would be a problem.

**2. Data Privacy:** Corporate data, whether it can be critical or not, will be private, so if it is not located within the walls of the company, there can be a risk in terms of privacy of data.

**3. Integration with the rest of the systems applications:** It may happen that some applications are local, and some are in the cloud. So, there will be chances of increased complexity when we want to use data which in the cloud with the local data.

### **Services provided in Azure as PaaS**

Some of the services provided by Microsoft Azure as PaaS are Azure WebJobs, Azure Event Grid, Azure Service Bus, Azure Cosmos DB, Azure Cognitive Services, Azure CDN, Azure Redis Cache, Azure API Management.

### ***3. Software as a Service (SaaS)***

SaaS is also known as "**on-demand software**". It is a software in which the applications are hosted by a cloud service provider. Users can access these applications with the help of internet connection and web browser.

### ***Characteristics of SaaS***

There are the following characteristics of SaaS -

- Managed from a central location
- Hosted on a remote server
- Accessible over the internet
- Users are not responsible for hardware and software updates. Updates are applied automatically.
- The services are purchased on the pay-as-per-use basis

**Example:** Microsoft 365, BigCommerce, Google Apps, Salesforce, Dropbox, ZenDesk, Cisco WebEx, ZenDesk, Slack, and GoToMeeting.

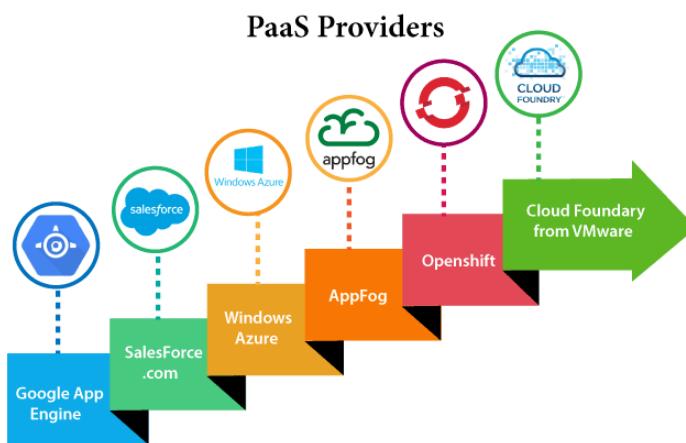


Image: PaaS Providers

### Software as a Service / SaaS

SaaS is also known as "**On-Demand Software**". It is a software distribution model in which services are hosted by a cloud service provider. These services are available to end-users over the internet so, the end-users do not need to install any software on their devices to access these services.

There are the following services provided by SaaS providers -

**Business Services** - SaaS Provider provides various business services to start-up the business. The SaaS business services include **ERP** (Enterprise Resource Planning), **CRM** (Customer Relationship Management), **billing**, and **sales**.

**Document Management** - SaaS document management is a software application offered by a third party (SaaS providers) to create, manage, and track electronic documents.

**Example:** Slack, Samepage, Box, and Zoho Forms.

**Social Networks** - As we all know, social networking sites are used by the general public, so social networking service providers use SaaS for their convenience and handle the general public's information.

**Mail Services** - To handle the unpredictable number of users and load on e-mail services, many e-mail providers offering their services using SaaS.



Image: Cloud SaaS Model

Reference: <https://encrypted-tbn0.gstatic.com/images>

### *Advantages of SaaS cloud computing layer*

- 1. SaaS is easy to buy:** SaaS pricing is based on a monthly fee or annual fee subscription, so it allows organizations to access business functionality at a low cost, which is less than licensed applications.  
Unlike traditional software, which is sold as a licensed based with an up-front cost (and often an optional ongoing support fee), SaaS providers are generally pricing the applications using a subscription fee, most commonly a monthly or annually fee.
- 2. One to Many:** SaaS services are offered as a one-to-many model means a single instance of the application is shared by multiple users.
- 3. Less hardware required for SaaS:** The software is hosted remotely, so organizations do not need to invest in additional hardware.
- 4. Low maintenance required for SaaS:** Software as a service removes the need for installation, set-up, and daily maintenance for the organizations. The initial set-up cost for SaaS is typically less than the enterprise software. SaaS vendors are pricing their applications based on some usage parameters, such as a number of users using the application. So, SaaS does easy to monitor and automatic updates.
- 5. No special software or hardware versions required:** All users will have the same version of the software and typically access it through the web browser. SaaS reduces IT support costs by outsourcing hardware and software maintenance and support to the IaaS provider.
- 6. Multidevice support:** SaaS services can be accessed from any device such as desktops, laptops, tablets, phones, and thin clients.
- 7. API Integration:** SaaS services easily integrate with other software or services through standard APIs.

**8. No client-side installation:** SaaS services are accessed directly from the service provider using the internet connection, so do not need to require any software installation.

#### *Disadvantages of SaaS cloud computing layer*

**1. Security:** Actually, data is stored in the cloud, so security may be an issue for some users. However, cloud computing is not more secure than in-house deployment.

**2. Latency issue:** Since data and applications are stored in the cloud at a variable distance from the end-user, there is a possibility that there may be greater latency when interacting with the application compared to local deployment. Therefore, the SaaS model is not suitable for applications whose demand response time is in milliseconds.

**3. Total Dependency on Internet:** Without an internet connection, most SaaS applications are not usable.

**4. Switching between SaaS vendors is difficult:** Switching SaaS vendors involves the difficult and slow task of transferring the very large data files over the internet and then converting and importing them into another SaaS also.

#### Services provided in Azure as PaaS

All the services or webapps which you are using from Microsoft are examples of SaaS e.g., Outlook, Onedrive, Onenote, Word Online, Excel Online, Powerpoint Online and similar.

#### Difference between IaaS, PaaS, and SaaS

The below table shows the difference between IaaS, PaaS, and SaaS

IaaS	PaaS	SaaS
It provides a virtual data center to store information and create platforms for app development, testing, and deployment.	It provides virtual platforms and tools to create, test, and deploy apps.	It provides web software and apps to complete business tasks.
It provides access to resources such as virtual machines, virtual storage, etc.	It provides runtime environments and deployment tools for applications.	It provides software as a service to the end-users.
It is used by network architects.	It is used by developers.	It is used by end users.
IaaS provides only Infrastructure.	PaaS provides Infrastructure +Platform.	SaaS Infrastructure +Platform +Software.

## Demo on Getting Started with an Azure Cloud and Exploring the Azure Service offerings

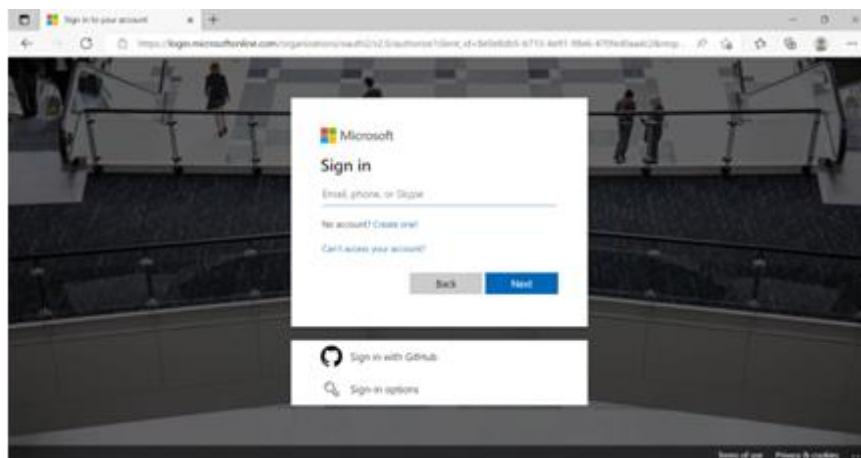
**Activity Details:** This activity is to enable learners to create their first Microsoft Azure Cloud account on the Azure portal and login to the dashboard environment on Azure to check and confirm login and get familiar with the Azure cloud dashboard elements. You need an email id to create the new account. Azure provides free credits to new accounts under various schemes like, free trial accounts, student accounts, etc. Apart from free credits, Azure also provides several free services quota for each account.

### Creating an Account

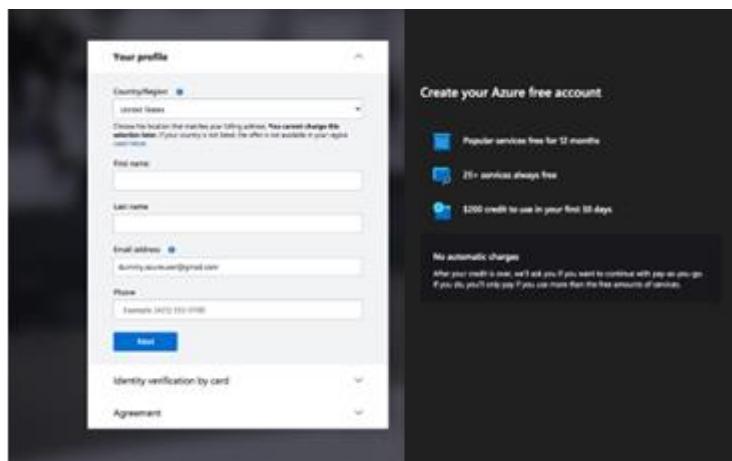
Microsoft offers a free account and you can get popular services free for 12 months. Let's start with creating an Azure free account. Use below link to access Microsoft azure portal. URL: <https://azure.microsoft.com/en-us/free/>



- You can click on the link and login with your Microsoft account or you can create one.



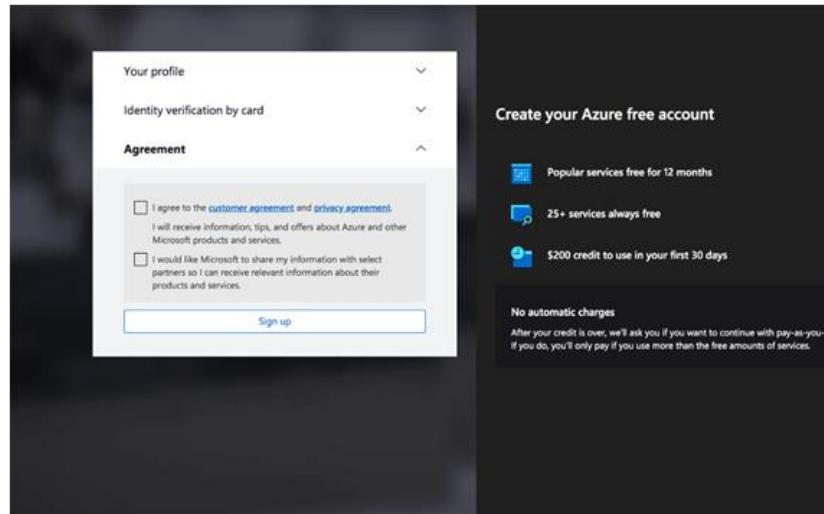
If you are creating the account you are asked to verify your account with the code. Once you provide the code and set up the account, it will take you to the profile page where you enter your profile information, card details for the verification and agreement as below.



- In card Payments Section supports only VISA and MASTER CARDS, RUPAY CARDS are not supported.

### Providing Profile information

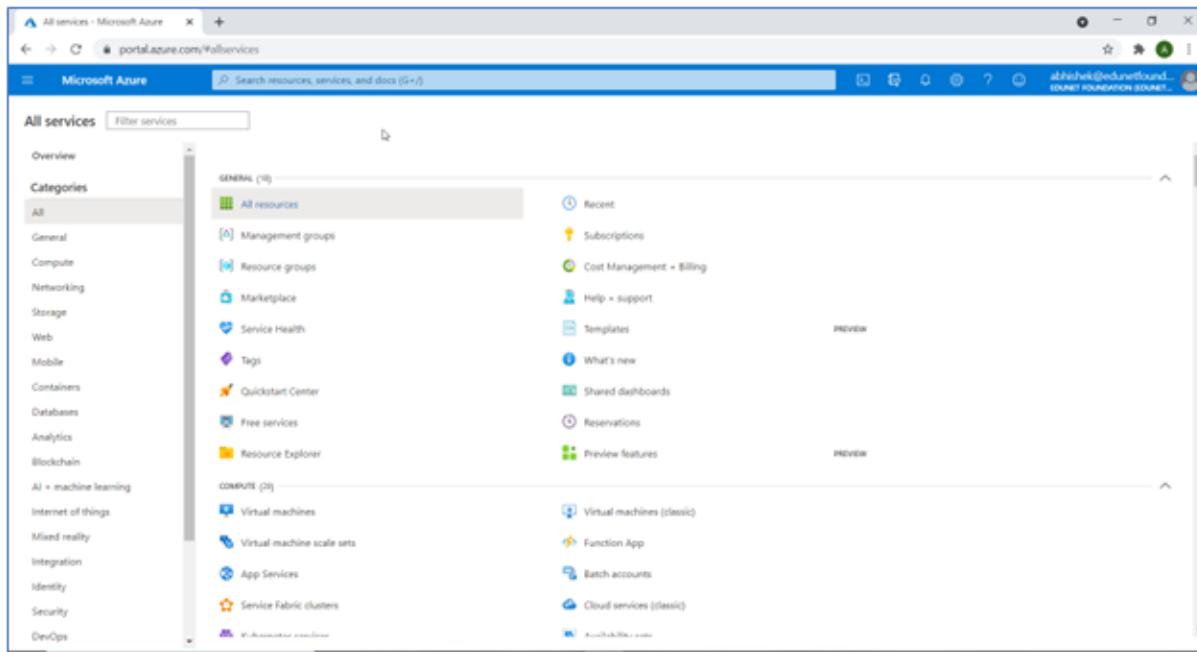
Once you provide all this information, you can check the agreement checkboxes and hit the sign-up button.



Once done, you can see the page below and click on the Build in the portal button and you will be redirected to the portal dashboard as below.

## Portal Dashboard

- When you click on the home and you can see the commonly used resource's dashboard and you can navigate through on the left blade as well.



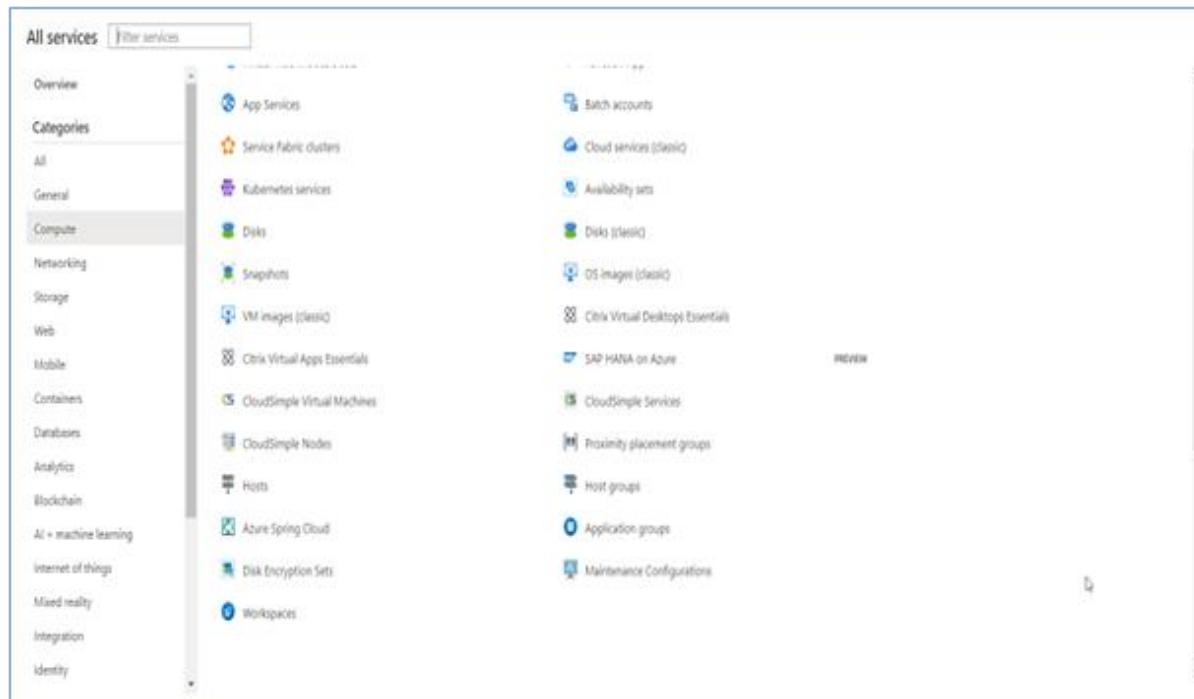
## Microsoft Azure Services

You can also search services according to categories of your choice.

The **Azure** cloud platform is more than 200 products and cloud **services** designed to help you bring new solutions to life—to solve today's challenges and create the future. Build, run and manage applications across multiple clouds, on-premises and at the edge, with the tools and frameworks of your choice.

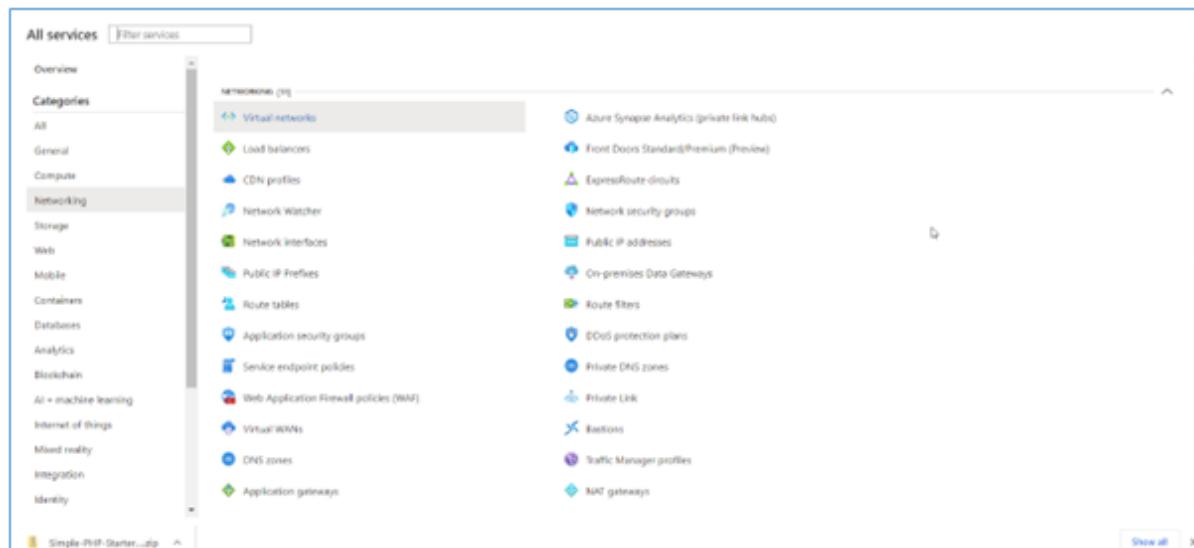
Let us explore some of the most prominent Azure Services.

## Azure Compute Services



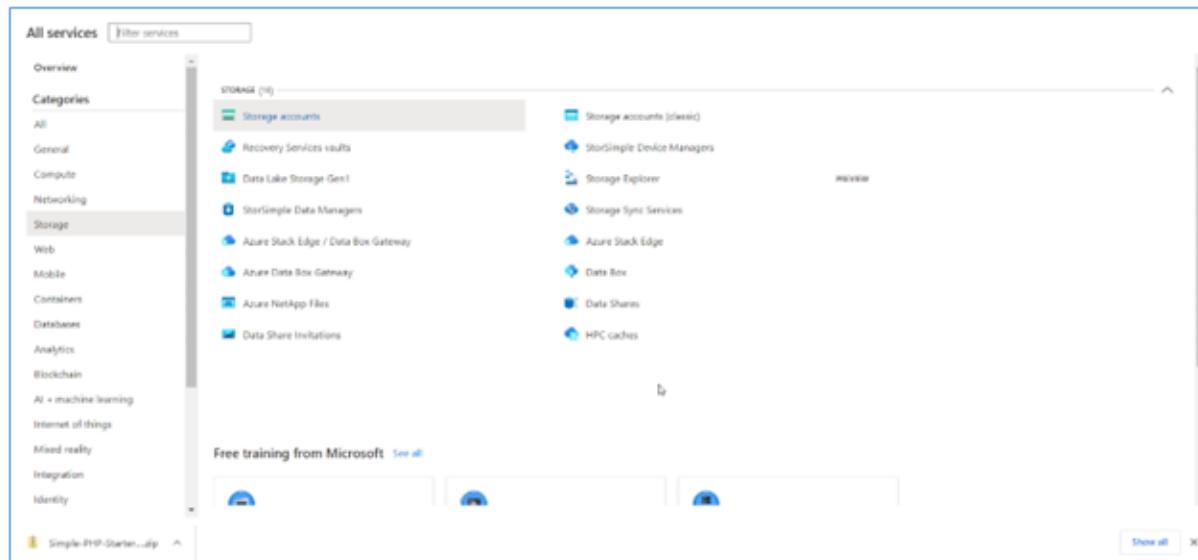
- App Services
- Service Fabric clusters
- Kubernetes services
- Disk
- Snapshots
- VM images (classic)
- Citrix Virtual Apps Essentials
- CloudSimple Virtual Machines
- CloudSimple Nodes
- Hosts
- Azure Spring Cloud
- Disk Encryption Settings
- Workspaces
- Batch accounts
- Cloud services (classic)
- Availability sets
- Disk (classic)
- OS images (classic)
- Citrix Virtual Desktop Essentials
- SAP HANA on Azure
- CloudSimple Services
- Proximity placement groups
- Host group
- Application groups
- Maintenance Configurations

## Azure Networking Services



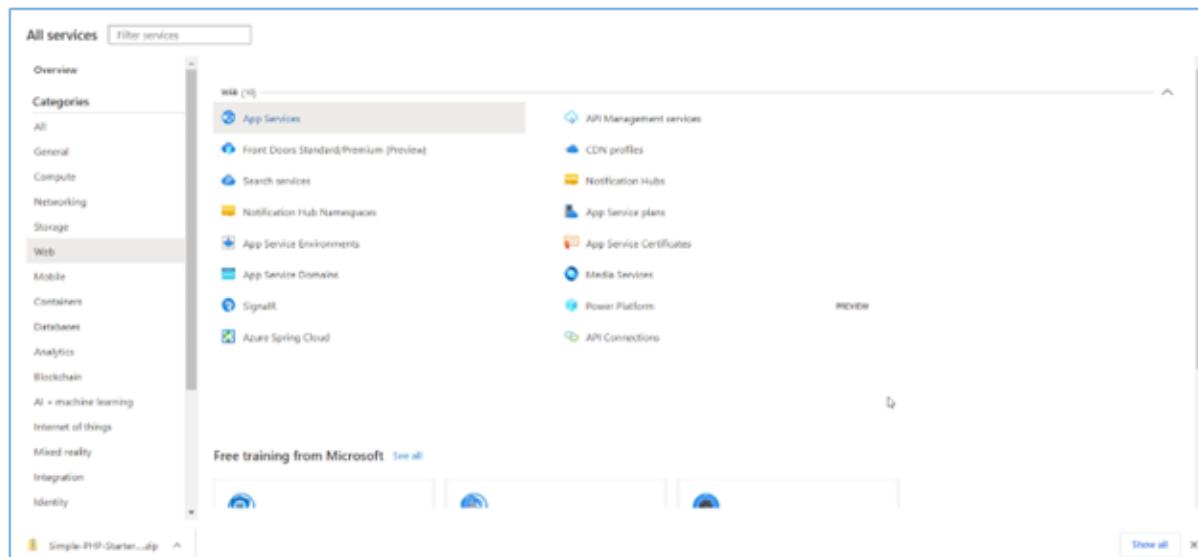
- Virtual networks
- Load balancers
- CDN profiles
- Network Watcher
- Network interfaces
- Public IP prefixes
- Route tables
- Application security groups
- Service endpoint policies
- Web Application Firewall policies (WAF)
- Virtual WANs
- DNS zones
- Application gateways
- Azure Synapse Analytics (private link hub)
- Front Door Standard/Premium (Preview)
- ExpressRoute circuits
- Network security groups
- Public IP addresses
- On-premises Data Gateways
- Route filters
- DDoS protection plans
- Private DNS zones
- Private Link
- Failover
- Traffic Manager profiles
- NAT gateways

## Azure Storage



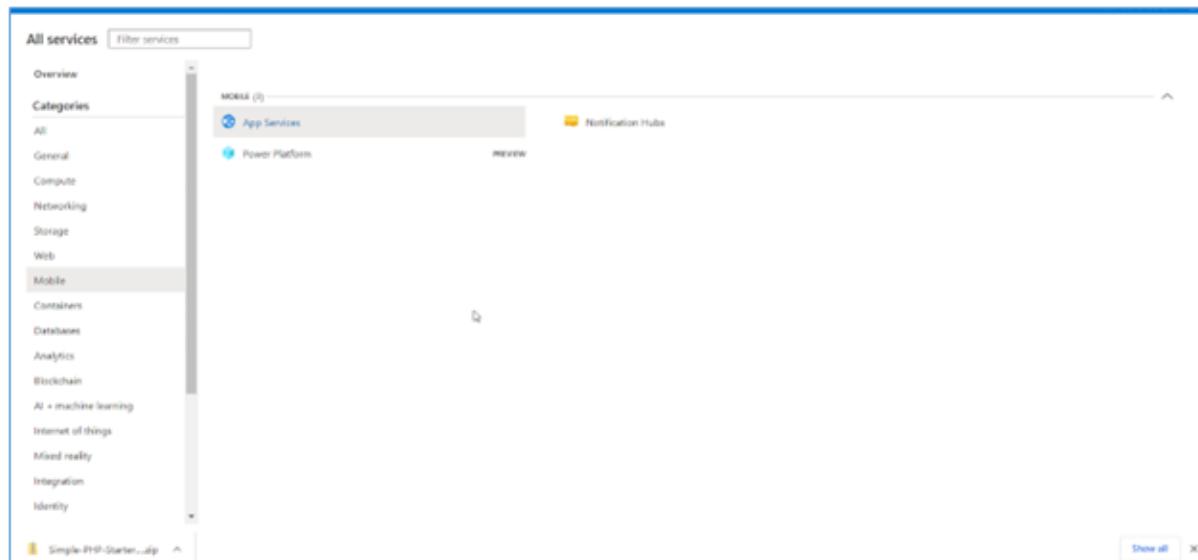
The screenshot shows the Azure portal's service catalog. The left sidebar has 'Storage' selected under the 'Categories' section. The main area displays the 'STORAGE (16)' category, which includes options like Storage accounts, Recovery Services vaults, Data Lake Storage Gen1, and Azure Stack Edge / Data Box Gateway. A 'Free training from Microsoft' banner is visible at the bottom.

## Azure Web



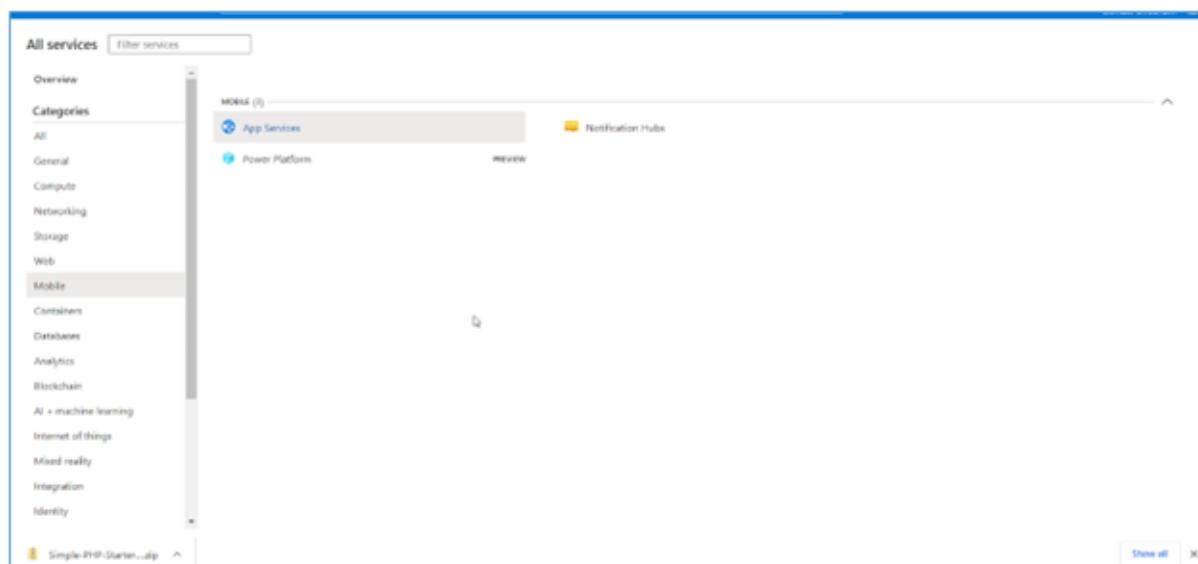
The screenshot shows the Azure portal's service catalog. The left sidebar has 'Web' selected under the 'Categories' section. The main area displays the 'WEB (10)' category, which includes App Services, Front Doors Standard/Premium (Preview), Search services, and App Service Environments. A 'Free training from Microsoft' banner is visible at the bottom.

## Azure Mobile



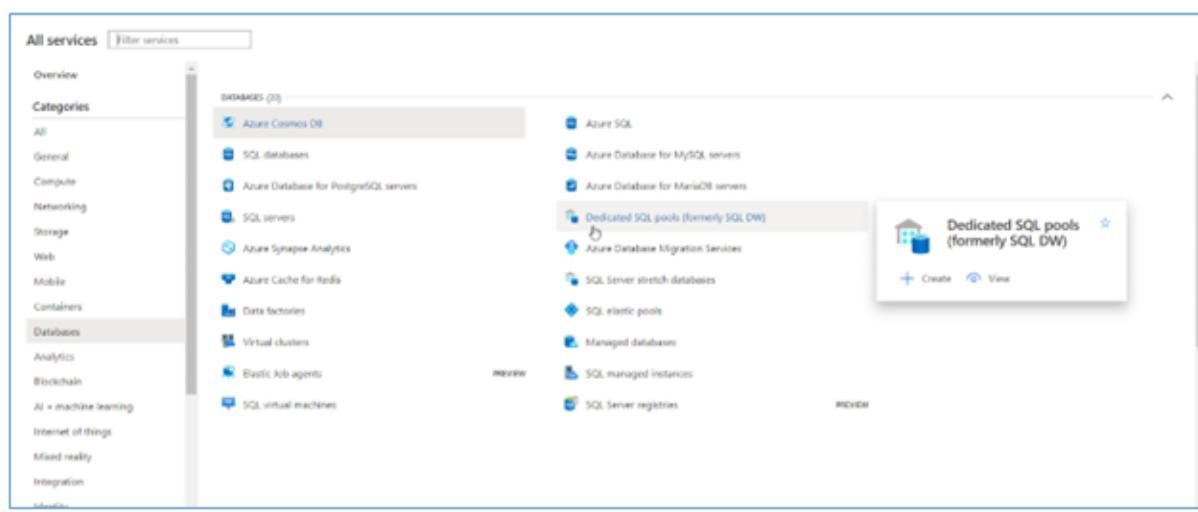
The screenshot shows the Azure portal's 'All services' blade. The left sidebar has 'Mobile' selected under the 'Categories' section. The main area displays a list of services: 'App Services' (selected), 'Power Platform', and 'Notification Hubs'. A preview pane for 'App Services' is visible on the right.

## Azure Containers



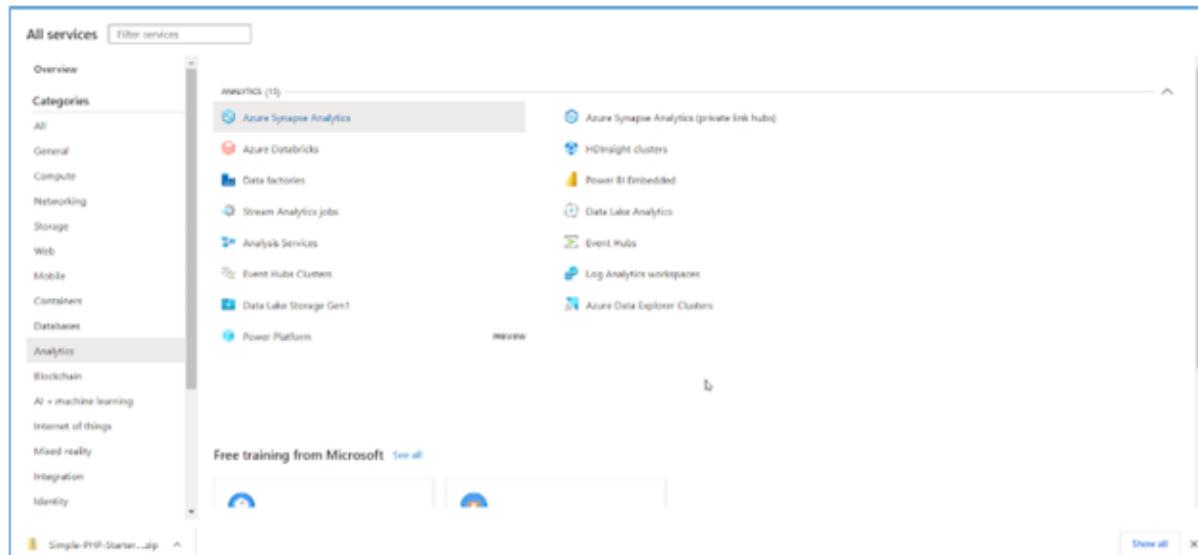
This screenshot is identical to the one above, showing the 'All services' blade with 'Mobile' selected in the sidebar. It displays the same list of services: App Services, Power Platform, and Notification Hubs, with a preview pane for App Services.

## Azure Database



The screenshot shows the 'All services' blade with 'Databases' selected in the sidebar. The main area lists various database services: Azure Cosmos DB, SQL databases, Azure Database for PostgreSQL servers, Azure Database for MariaDB servers, Dedicated SQL pools (formerly SQL DW), Azure Database Migration Services, SQL Server stretch databases, SQL elastic pools, Managed databases, SQL managed instances, and SQL Server registries. A callout box highlights 'Dedicated SQL pools (formerly SQL DW)' with a 'Create' button and a 'View' link.

## Azure Analytics



All services Filter services

Overview

Categories

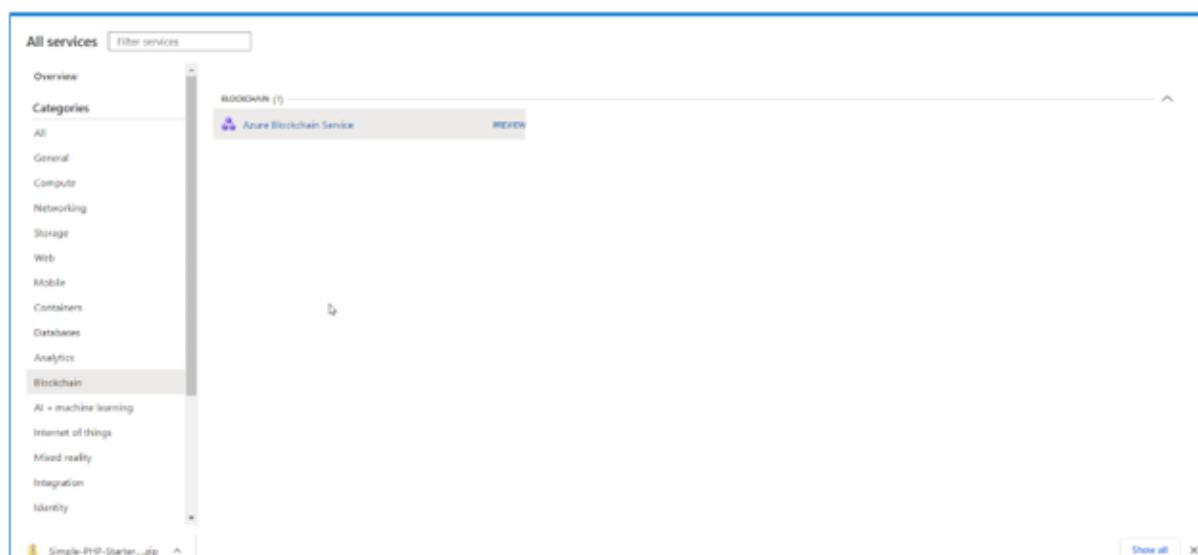
- All
- General
- Compute
- Networking
- Storage
- Web
- Mobile
- Containers
- Databases
- Analytics**
- Blockchain
- AI + machine learning
- Internet of things
- Mixed reality
- Integration
- Identity

Free training from Microsoft See all

Simple-PHP-Starter.zip

Preview

## Azure Blockchain Service



All services Filter services

Overview

Categories

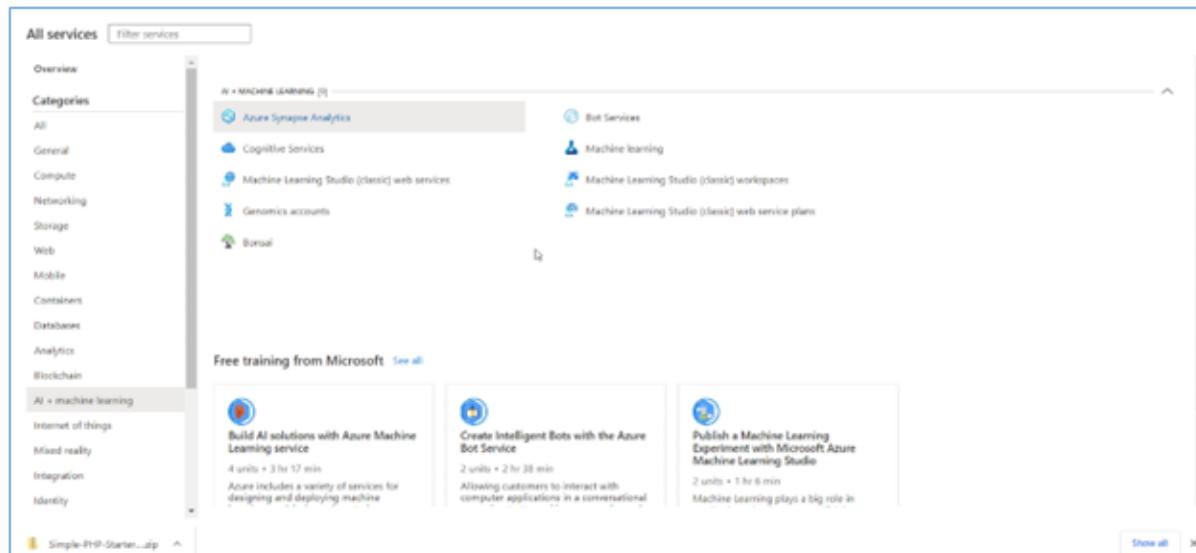
- All
- General
- Compute
- Networking
- Storage
- Web
- Mobile
- Containers
- Databases
- Analytics
- Blockchain**
- AI + machine learning
- Internet of things
- Mixed reality
- Integration
- Identity

Simple-PHP-Starter.zip

BLOCKCHAIN

PREVIEW

## Azure AI + Machine Learning



**All services** Filter services

**Categories**

- All
- General
- Compute
- Networking
- Storage
- Web
- Mobile
- Containers
- Databases
- Analytics
- Blockchain
- AI + machine learning**
- Internet of things
- Mixed reality
- Integration
- Identity

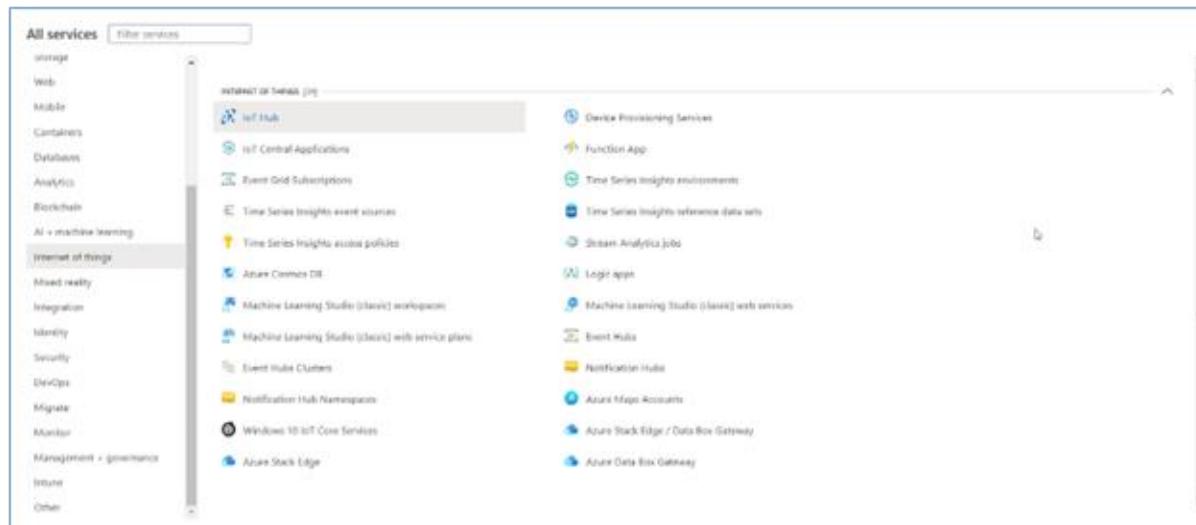
**AI + machine learning**

Free training from Microsoft See all

- Build AI solutions with Azure Machine Learning service** 4 units • 3 hr 17 min Azure includes a variety of services for designing and deploying machine learning models.
- Create Intelligent Bots with the Azure Bot Service** 2 units • 2 hr 38 min Allowing customers to interact with computer applications in a conversational way.
- Publish a Machine Learning Experiment with Microsoft Azure Machine Learning Studio** 2 units • 1 hr 6 min Machine learning plays a big role in...

Show all X

## Internet of Things



**All services** Filter services

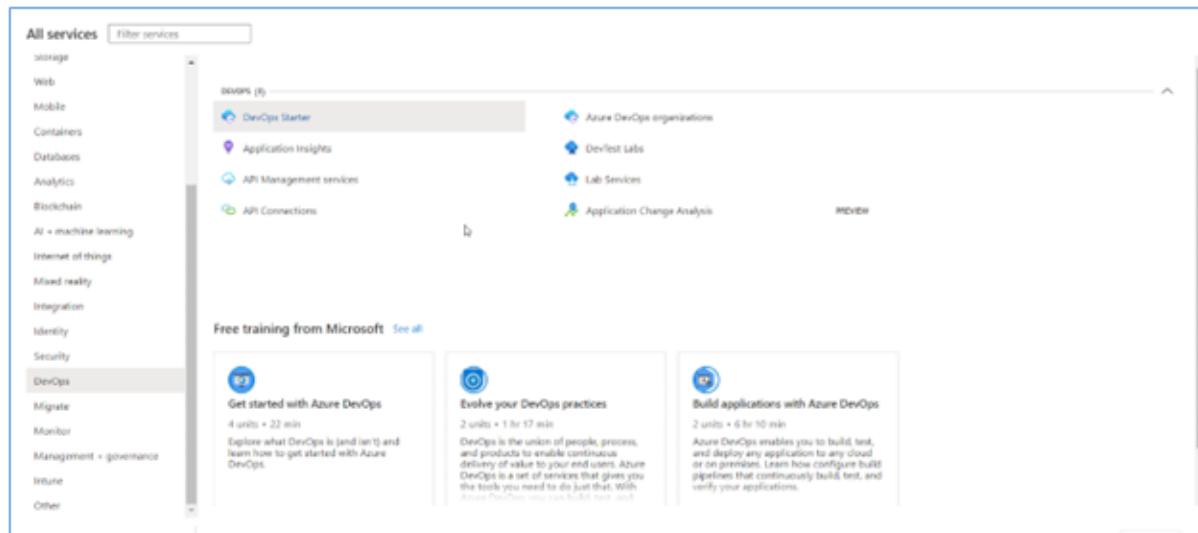
**Categories**

- Storage
- Web
- Mobile
- Containers
- Databases
- Analytics
- Blockchain
- AI + machine learning**
- Internet of things**
- Mixed reality
- Integration
- Identity
- Security
- DevOps
- Migrate
- Monitor
- Management + governance
- Intune
- Other

**Internet of things**

- IoT Hub**
- IoT Central Applications**
- Event Grid Subscriptions**
- Time Series Insights event sources**
- Time Series Insights access policies**
- Azure Cosmos DB**
- Machine Learning Studio (classic) workspaces**
- Machine Learning Studio (classic) web service plans**
- Event Hubs Clusters**
- Notification Hubs Namespaces**
- Windows 10 IoT Core Services**
- Azure Stack Edge**
- Device Provisioning Service**
- Function App**
- Time Series Insights environments**
- Time Series Insights reference data sets**
- Stream Analytics jobs**
- Log Analytics**
- Machine Learning Studio (classic) web services**
- Event Hubs**
- Notification Hubs**
- Azure Maps Accounts**
- Azure Stack Edge / Data Box Gateway**
- Azure Data Box Gateway**

## DevOps



All services Filter services

Storage Web Mobile Containers Databases Analytics Blockchain AI + machine learning Internet of things Mixed reality Integration Identity Security DevOps Migrate Monitor Management + governance Intune Other

DEVOPS (3)

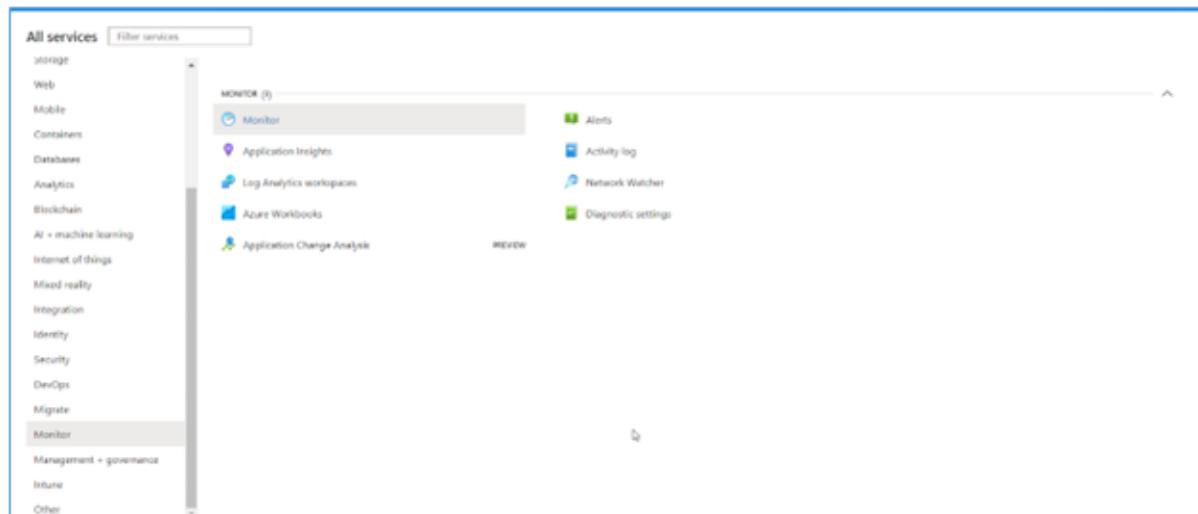
- DevOps Starter
- Application Insights
- API Management services
- API Connections

Azure DevOps organizations Dev/Test Labs Lab Services Application Change Analysis

Free training from Microsoft See all

- Get started with Azure DevOps 4 units • 22 min Explore what DevOps is (and isn't) and learn how to get started with Azure DevOps.
- Evolve your DevOps practices 2 units • 1 hr 17 min DevOps is the union of people, process, and products to enable continuous delivery of value to your end users. Azure DevOps is a set of tools that gives you the tools you need to do just that. With
- Build applications with Azure DevOps 2 units • 6 hr 10 min Azure DevOps enables you to build, test, and deploy any application to any cloud or on-premises. Learn how to configure build pipelines that automatically build, test, and verify your applications.

## Monitor



All services Filter services

Storage Web Mobile Containers Databases Analytics Blockchain AI + machine learning Internet of things Mixed reality Integration Identity Security DevOps Migrate Monitor Management + governance Intune Other

MONITOR (4)

- Monitor
- Application Insights
- Log Analytics workspaces
- Azure Workbooks
- Application Change Analysis

Alerts Activity log Network Watcher Diagnostic settings

**In this activity you have learnt about how to create a free account on the Azure Platform and explored all the services that are present on Azure Cloud Platform.**

**Now, we will study about the Fundamental of Networking that will give you the foundational knowledge on Networking concepts that will help in creating your own Virtual Network on the Azure Cloud.**

## 5.2 Fundamentals of Networking and Networking Protocols

When you connect two or more than two computing devices together, they become connected by a network. This connection may be wired or wireless. The wired medium may be an ethernet cable or fiber connection. Similarly, wireless medium may be Wi-Fi, Bluetooth or any other radio wave based. The connection, maintenance and administration of this network is called Networking.

The network can be of different types based on characteristic properties.

## Types of Networks

There are various types of networks. This is based on coverage. The types are as follows:

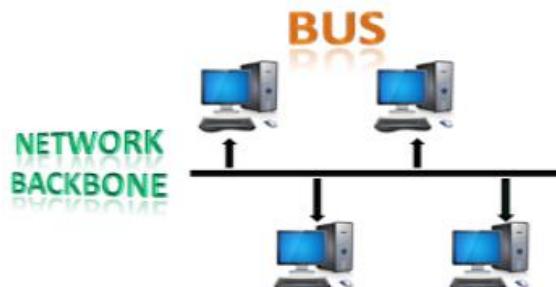
- LAN: LAN aka Local Area Network. This is the smallest form of Network. Usually, two or more devices may be connected via ethernet cable or wireless network device such as hub or switch. In our labs, homes, colleges the network that we have is of LAN type. The radius of such a network is generally not more than 2KM.
- MAN: MAN, aka Metropolitan Area Network. This type of network is larger than LAN. This may spread in a larger area such as a metropolitan city or in our case a colony. This is more of an abstract concept because when it was designed it was meant to support a city. However, nowadays instead of opting for a MAN we usually go for a more commonly used type of network that we called as the Internet. The Internet is also an example of WAN.
- WAN: WAN aka Wide Area Network. The best example of this type is the World Wide Web, also known as the Internet. Whenever you are using any app to connect to a server, you would probably be using the Internet.

For the general population, the Internet has become a synonym with the network. However, one should not confuse the two terms. These networks can not only have different connection mechanisms but also different patterns of connections or topologies.

## Topology

Topology is about the logical pattern of connection. To understand the things better, let us see the types and examples of the topologies.

**Bus Topology:** The systems are connected with a common data transfer line. The layout is cheap but vulnerable to failure and only suitable for low traffic volumes. This is not used for colleges/office networks today, but can still be found within some consumer products.



Bus Topology

Source: <https://static.javatpoint.com/tutorial/computer-network/images/computer-network-topologies-bus-topology.png>

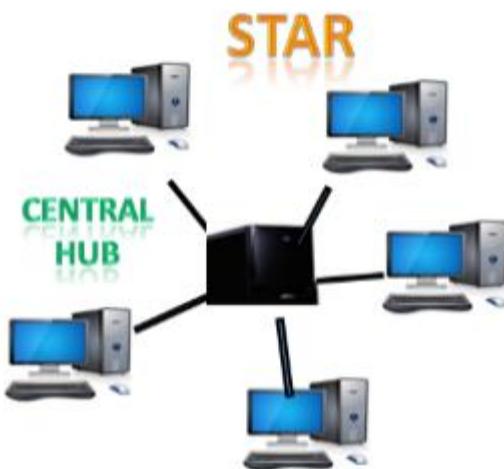
**Ring Topology:** This is an advanced version of Bus topology where devices are connected in a logical circle or ring format. It is easy to manage and with a low risk of collision but reliant on all nodes being powered up and in full working order. Rarely used today.



Ring Topology

Source: <https://static.javatpoint.com/tutorial/computer-network/images/computer-network-topologies-ring-topology.png>

**Star Topology:** All devices are connected to a central switch, which makes it easy to add new nodes without rebooting all currently connected devices. This topology makes efficient use of cable and is easy to administer. On the other hand, the health of the switch is vital. This topology requires monitoring and maintenance. However, it is a commonly used topology.

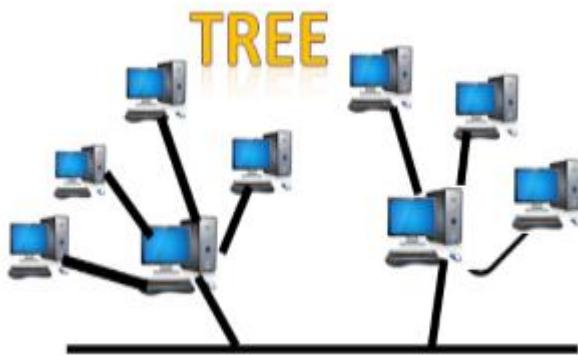


Ring Topology

Source: <https://static.javatpoint.com/tutorial/computer-network/images/computer-network-topologies-star-topology.png>

**Tree Topology:** A hierarchical layout that links together groups of nodes. Creates parent-child dependencies between root nodes and regular nodes. This layout can be

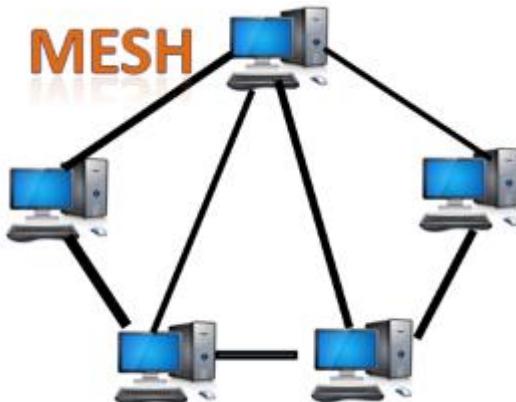
vulnerable to failure if a root node has a problem. This topology is complicated and difficult to manage and it uses a lot of cable.



Tree Topology

Source: <https://static.javatpoint.com/tutorial/computer-network/images/computer-network-topologies-tree-topology.png>

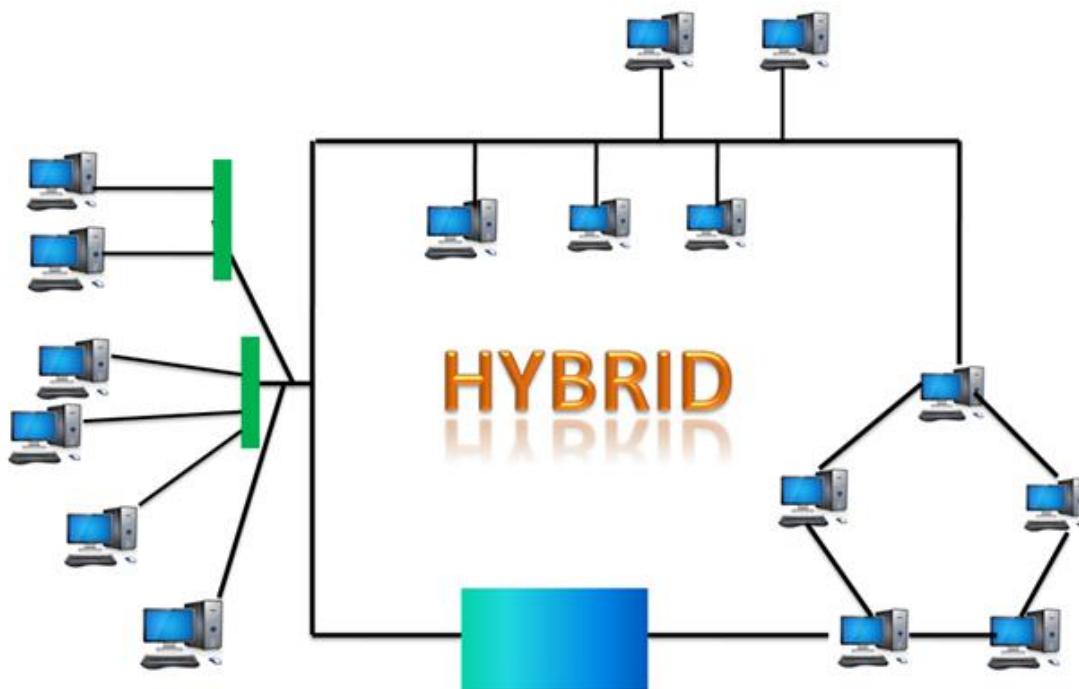
**Mesh Topology:** Each node is connected to every other mode with a direct link. This topology is also known to have peer2peer connections. This topology creates a very reliable network, but requires a large amount of cable and is difficult to administer. Wifi networks make this topology more feasible.



Mesh Topology

Source: <https://static.javatpoint.com/tutorial/computer-network/images/computer-network-topologies-mesh-topology.png>

**Hybrid Topology:** Combines two or more of the standard topologies. This can be a good solution to create quickly link together different existing networks into a unified system.



Hybrid Topology

Source: <https://static.javatpoint.com/tutorial/computer-network/images/computer-network-topologies-hybrid-topology.png>

## Networking Protocols

### IP (IP Addressing)

#### *What is the Internet Protocol (IP)?*

The Internet Protocol (IP) is a protocol, or set of rules, for routing and addressing packets of data so that they can travel across networks and arrive at the correct destination. Data traversing the Internet is divided into smaller pieces, called packets. IP information is attached to each packet, and this information helps routers to send packets to the right place. Every device or domain that connects to the Internet is assigned an IP address, and as packets are directed to the IP address attached to them, data arrives where it is needed. Once the packets arrive at their destination, they are handled differently depending on which transport protocol is used in combination with IP. The most common transport protocols are TCP and UDP.

#### **What is a network protocol?**

In networking, a protocol is a standardized way of doing certain actions and formatting data so that two or more devices are able to communicate with and understand each other.

To understand why protocols are necessary, consider the process of mailing a letter. On the envelope, addresses are written in the following order: name, street address, city, state, and zip code.

If an envelope is dropped into a mailbox with the zip code written first, followed by the street address, followed by the state, and so on, the post office won't deliver it. There is an agreed-upon protocol for writing addresses in order for the postal system to work. In the same way, all IP data packets must present certain information in a certain order, and all IP addresses follow a standardized format.

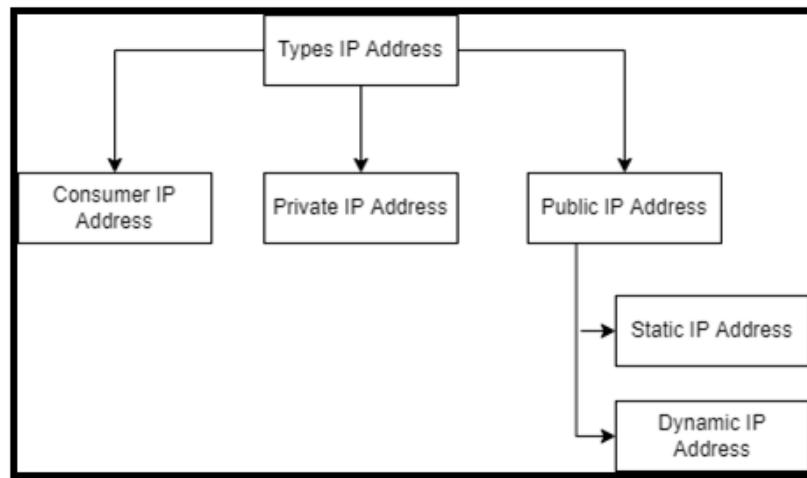
### ***IP Address in Networking-***

In networking,

- IP Address is short for **Internet Protocol Address**.
- It is a unique address assigned to each computing device in an IP network.
- ISP assigns IP Address to all the devices present on its network.
- Computing devices use IP Address to identify and communicate with other devices in the IP network.

### **Types Of IP Address-**

IP Addresses may be of the following two types-



There are different categories of IP addresses, and within each category, different types.

#### ***Consumer IP addresses***

Every individual or business with an internet service plan will have two types of IP addresses: their private IP addresses and their public IP address. The terms public and private relate to the network location — that is, a private IP address is used inside a network, while a public one is used outside a network.

#### ***Private IP addresses***

Every device that connects to your internet network has a private IP address. This includes computers, smartphones, and tablets but also any Bluetooth-enabled devices like speakers, printers, or smart TVs. With the growing internet of things, the number of private IP addresses you have at home is probably growing. Your router needs a way to identify these items separately, and many items need a way to recognize each other. Therefore, your router generates private IP addresses that are unique identifiers for each device that differentiate them on the network.

### ***Public IP addresses***

A public IP address is the primary address associated with your whole network. While each connected device has its own IP address, they are also included within the main IP address for your network. As described above, your public IP address is provided to your router by your ISP. Typically, ISPs have a large pool of IP addresses that they distribute to their customers. Your public IP address is the address that all the devices outside your internet network will use to recognize your network. Public IP addresses come in two forms – dynamic and static.

#### **1. Static IP Address-**

Static IP Address is an IP Address that once assigned to a network element always remains the same.

- They are configured manually.

#### **NOTE**

- Some ISPs do not provide static IP addresses.
- Static IP Addresses are more costly than dynamic IP Addresses.

#### **2. Dynamic IP Address-**

Dynamic IP Address is a temporarily assigned IP Address to a network element.

- It can be assigned to a different device if it is not in use.
- DHCP or PPPoE assigns dynamic IP addresses.

#### **How does IP address work?**

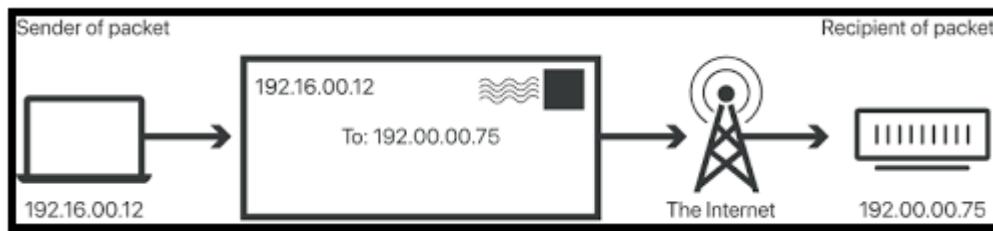
If you want to understand why a particular device is not connecting in the way you would expect or you want to troubleshoot why your network may not be working, it helps understand how IP addresses work.

Internet Protocol works the same way as any other language, by communicating using set guidelines to pass information. All devices find, send, and exchange information with other connected devices using this protocol. By speaking the same language, any computer in any location can talk to one another.

The use of IP addresses typically happens behind the scenes. The process works like this:

1. Your device indirectly connects to the internet by connecting at first to a network connected to the internet, which then grants your device access to the internet.

2. When you are at home, that network will probably be your Internet Service Provider (ISP). At work, it will be your company network.
3. Your IP address is assigned to your device by your ISP.
4. Your internet activity goes through the ISP, and they route it back to you, using your IP address. Since they are giving you access to the internet, it is their role to assign an IP address to your device.
5. However, your IP address can change. For example, turning your modem or router on or off can change it. Or you can contact your ISP, and they can change it for you.
6. When you are out and about – for example, traveling – and you take your device with you, your home IP address does not come with you. This is because you will be using another network (Wi-Fi at a hotel, airport, or coffee shop, etc.) to access the internet and will be using a different (and temporary) IP address, assigned to you by the ISP of the hotel, airport or coffee shop.



### **How to look up IP addresses**

The simplest way to check your router's public IP address is to search "What is my IP address?" on Google. Google will show you the answer at the top of the page.

Other websites will show you the same information: they can see your public IP address because, by visiting the site, your router has made a request and therefore revealed the information. The site IP Location goes further by showing the name of your ISP and your city.

Generally, you will only receive an approximation of location using this technique — where the provider is, but not the actual device location. If you are doing this, remember to log out of your VPN too. Obtaining the actual physical location address for the public IP address usually requires a search warrant to be submitted to the ISP. Finding your private IP address varies by platform:

#### **In Windows:**

- Use the command prompt.
- Search for "cmd" (without the quotes) using Windows search
- In the resulting pop-up box, type "ipconfig" (no quote marks) to find the information.

#### **On a Mac:**

- Go to System Preferences
- Select network – and the information should be visible.

#### **On an iPhone:**

- Go to Settings
- Select Wi-Fi and click the "i" in a circle () next to the network you are on – the IP address should be visible under the DHCP tab.

If you need to check the IP addresses of other devices on your network, go into the router. How you access the router depends on the brand and the software it uses. Generally, you should be able to type the router's gateway IP address into a web browser on the same network to access it. From there, you will need to navigate to something like "attached devices," which should display a list of all the devices currently or recently attached to the network — including their IP addresses.

### ***IP address classes***

With an IPv4 IP address, there are five classes of available IP ranges: Class A, Class B, Class C, Class D and Class E, while only A, B, and C are commonly used. Each class allows for a range of valid IP addresses, shown in the following table.

Class	Address range	Supports
<b>Class A</b>	1.0.0.1 to 126.255.255.254	Supports 16 million hosts on each of 127 networks.
<b>Class B</b>	128.1.0.1 to 191.255.255.254	Supports 65,000 hosts on each of 16,000 networks.
<b>Class C</b>	192.0.1.1 to 223.255.254.254	Supports 254 hosts on each of 2 million networks.
<b>Class D</b>	224.0.0.0 to 239.255.255.255	Reserved for <a href="#">multicast</a> groups.
<b>Class E</b>	240.0.0.0 to 254.255.255.254	Reserved for future use, or research and development purposes.

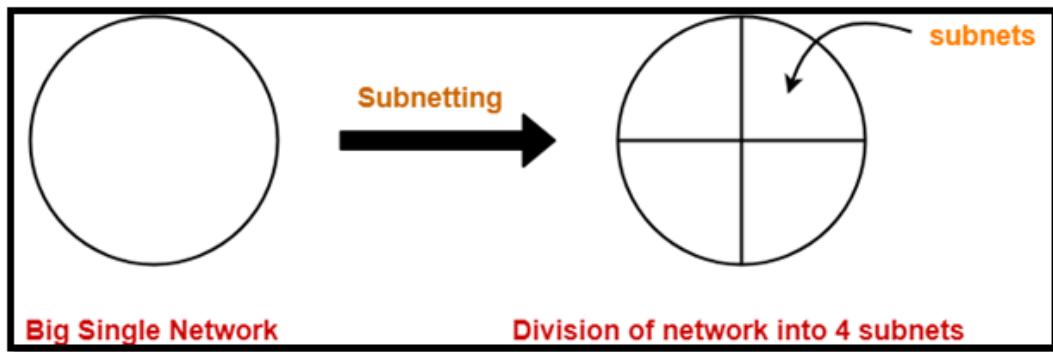
### ***Subnetting in Networking-***

In networking,

- The process of dividing a single network into multiple sub networks is called as **subnetting**.
- The sub networks so created are called as **subnets**.

### **Example-**

Following diagram shows the subnetting of a big single network into 4 smaller subnets-



### Advantages-

The two main advantages of subnetting a network is:

- It improves the security.
- The maintenance and administration of subnets is easy.

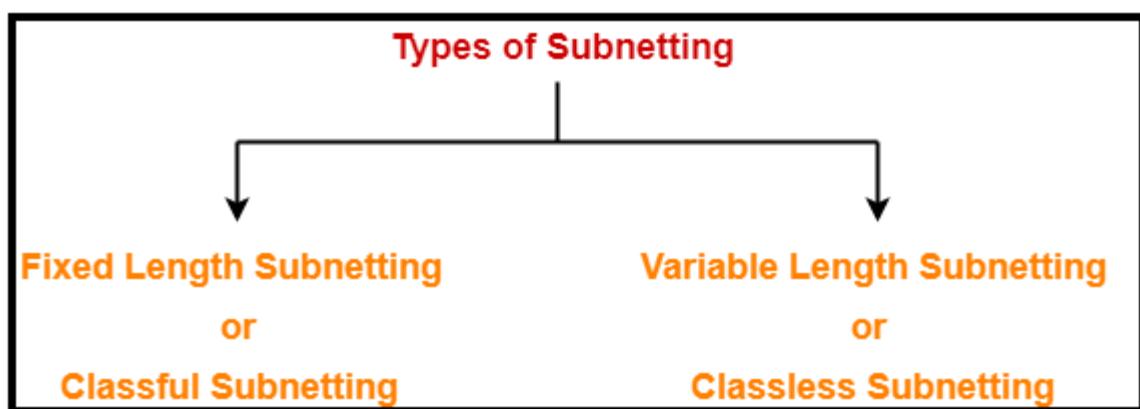
### Subnet ID-

Each subnet has its unique network address known as its **Subnet ID**.

- The subnet ID is created by borrowing some bits from the Host ID part of the IP Address.
- The number of bits borrowed depends on the number of subnets created.

### Types of Subnetting-

Subnetting of a network may be carried out in the following two ways-



- Fixed Length Subnetting
- Variable Length Subnetting

#### 1. Fixed Length Subnetting-

Fixed length subnetting also called as **classful subnetting** divides the network into subnets where-

- All the subnets are of same size.

- All the subnets have equal number of hosts.
- All the subnets have same subnet mask.

## 2. Variable Length Subnetting-

Variable length subnetting also called as **classless subnetting** divides the network into subnets where-

- All the subnets are not of same size.
- All the subnets do not have equal number of hosts.
- All the subnets do not have same subnet mask.

### Subnetting Examples-

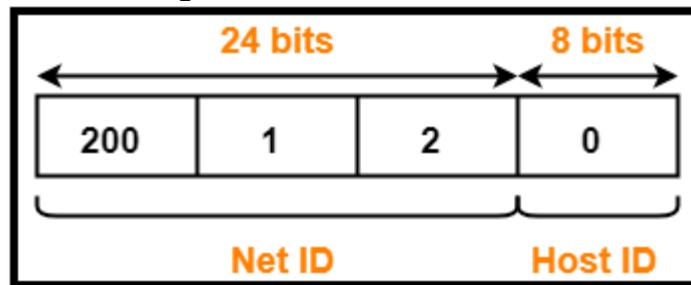
Now, we shall discuss some examples of subnetting a network-

Example-01:

Consider-

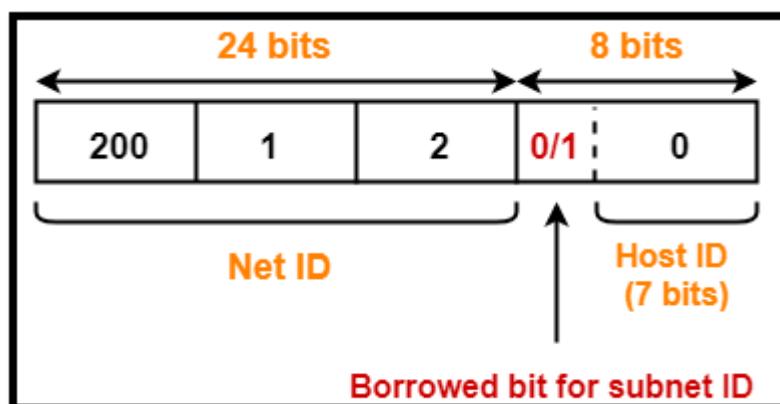
- We have a big single network having IP Address 200.1.2.0.
- We want to do subnetting and divide this network into 2 subnets.

Clearly, the given network belongs to class C.



For creating two subnets and to represent their subnet IDs, we require 1 bit. So,

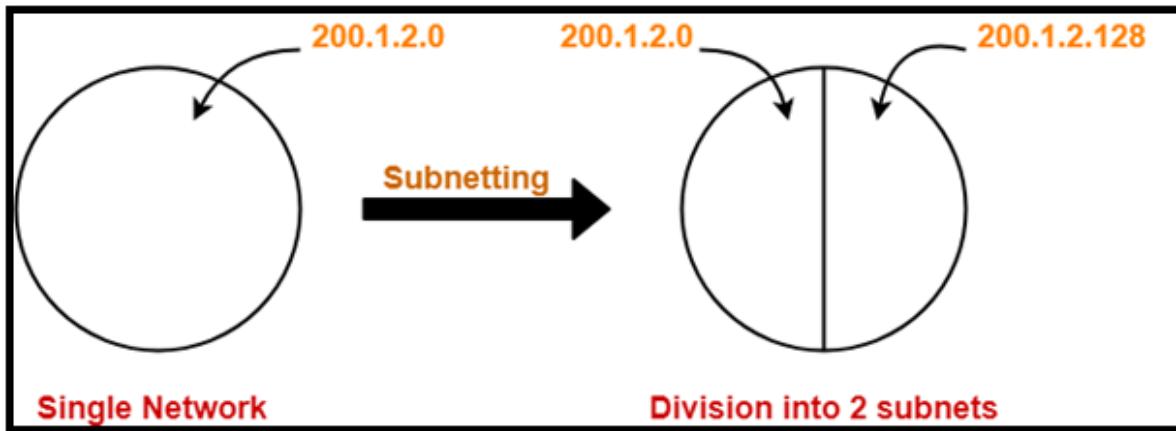
- We borrow one bit from the Host ID part.
- After borrowing one bit, Host ID part remains with only 7 bits.



- If borrowed bit = 0, then it represents the first subnet.
- If borrowed bit = 1, then it represents the second subnet.

IP Address of the two subnets are-

- $200.1.2.00000000 = 200.1.2.0$
- $200.1.2.10000000 = 200.1.2.128$



For 1st Subnet-

- IP Address of the subnet = 200.1.2.0
- Total number of IP Addresses =  $2^7 = 128$
- Total number of hosts that can be configured =  $128 - 2 = 126$
- Range of IP Addresses =  $[200.1.2.00000000, 200.1.2.01111111] = [200.1.2.0, 200.1.2.127]$
- Direct Broadcast Address =  $200.1.2.01111111 = 200.1.2.127$
- Limited Broadcast Address = 255.255.255.255

For 2nd Subnet-

- IP Address of the subnet = 200.1.2.128
- Total number of IP Addresses =  $2^7 = 128$
- Total number of hosts that can be configured =  $128 - 2 = 126$
- Range of IP Addresses =  $[200.1.2.10000000, 200.1.2.11111111] = [200.1.2.128, 200.1.2.255]$
- Direct Broadcast Address =  $200.1.2.11111111 = 200.1.2.255$
- Limited Broadcast Address = 255.255.255.255

### *Disadvantages of Subnetting-*

Point-01: Subnetting leads to loss of IP Addresses.

During subnetting,

- We have to face a loss of IP Addresses.
- This is because two IP Addresses are wasted for each subnet.

- One IP address is wasted for its network address.
- Other IP Address is wasted for its direct broadcasting address.

Point-02: Subnetting leads to complicated communication process.

After subnetting, the communication process becomes complex involving the following 4 steps-

1. Identifying the network
2. Identifying the sub network
3. Identifying the host
4. Identifying the process

Classless Addressing-

- Classless Addressing is an improved IP Addressing system.
- It makes the allocation of IP Addresses more efficient.
- It replaces the older classful addressing system based on classes.
- It is also known as **Classless Inter Domain Routing (CIDR)**.

CIDR Block- When a user asks for specific number of IP Addresses,

- CIDR dynamically assigns a block of IP Addresses based on certain rules.
- This block contains the required number of IP Addresses as demanded by the user.
- This block of IP Addresses is called as a **CIDR block**.

Rules For Creating CIDR Block- A CIDR block is created based on the following 3 rules-

Rule-01: All the IP Addresses in the CIDR block must be contiguous.

Rule-02:

- The size of the block must be presentable as power of 2.
- Size of the block is the total number of IP Addresses contained in the block.
- Size of any CIDR block will always be in the form 2<sup>1</sup>, 2<sup>2</sup>, 2<sup>3</sup>, 2<sup>4</sup>, 2<sup>5</sup> and so on.

Rule-03: First IP Address of the block must be divisible by the size of the block.

**CIDR Notation-**

CIDR IP Addresses look like-

**a.b.c.d / n**

- They end with a slash followed by a number called as IP network prefix.
- IP network prefix tells the number of bits used for the identification of network.

- Remaining bits are used for the identification of hosts in the network.

Example-

An example of CIDR IP Address is-

182.0.1.2 / 28

It suggests-

- 28 bits are used for the identification of network.
- Remaining 4 bits are used for the identification of hosts in the network.

## Transmission Control Protocol (TCP)

Transmission Control Protocol (TCP) – a connection-oriented communications protocol that facilitates the exchange of messages between computing devices in a network.

It is the most common protocol in networks that use the Internet Protocol (IP); together they are sometimes referred to as TCP/IP.

TCP takes messages from an application/server and divides them into packets, which can then be forwarded by the devices in the network – switches, routers, security gateways – to the destination. TCP numbers each packet and reassembles them prior to handing them off to the application/server recipient.

Because it is connection-oriented, it ensures a connection is established and maintained until the exchange between the application/servers sending and receiving the message is complete.

- TCP is connection-oriented, and a connection between client and server is established before data can be sent.
- The server must be listening (passive open) for connection requests from clients before a connection is established.
- Three-way handshake (active open), retransmission, and error-detection adds to reliability but lengthens latency.
- Applications that do not require reliable data stream service may use the User Datagram Protocol (UDP), which provides a connectionless datagram service that prioritizes time over reliability.
- TCP employs network congestion avoidance. However, there are vulnerabilities to TCP including denial of service, connection hijacking, TCP veto, and reset attack.

## What is User Datagram Protocol (UDP)?

User Datagram Protocol (UDP) is a communication protocol that is primarily used to establish low-latency and loss-tolerating connections between applications on the internet. UDP speeds up transmissions by enabling the transfer of data before an agreement is provided by the receiving party. As a result, UDP is beneficial in time-sensitive communications, including voice over IP (VoIP), domain name system (DNS) lookup, and video or audio playback.

UDP is an alternative to Transmission Control Protocol (TCP). Both UDP and TCP run on top of IP and are sometimes referred to as UDP/IP or TCP/IP. However, there are

important differences between the two. For example, UDP enables process-to-process communication, while TCP supports host-to-host communication. TCP sends individual packets and is considered a reliable transport medium. On the other hand, UDP sends messages, called *datagrams*, and is considered a best-effort mode of communications. This means UDP doesn't provide any guarantees that the data will be delivered or offer special features to retransmit lost or corrupted messages. UDP provides two services not provided by the IP layer. It provides port numbers to help distinguish different user requests. It also provides an optional checksum capability to verify that the data arrived intact.

### ***User Datagram Protocol features***

User Datagram Protocol has attributes that make it beneficial for use with applications that can tolerate lost data. Below are some examples:

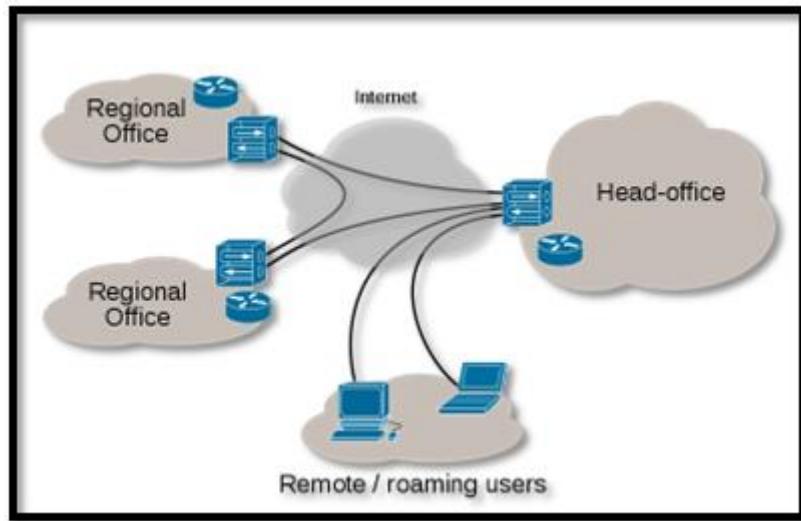
- It allows packets to be dropped and received in a different order than they were transmitted, making it suitable for real-time applications where latency might be a concern.
- It can be used for transaction-based protocols, such as DNS or Network Time Protocol (NTP).
- It can be used where a large number of clients are connected and where real-time error correction isn't necessary, such as gaming, voice or video conferencing, and streaming media.

## **5.3 Virtual Private Network (VPN)**

A virtual private network, or VPN, is an encrypted connection over the Internet from a device to a network. The encrypted connection helps ensure that sensitive data is safely transmitted. It prevents unauthorized people from eavesdropping on the traffic and allows the user to conduct work remotely. VPN technology is widely used in corporate environments.

### **How does a virtual private network (VPN) work?**

A VPN extends a corporate network through encrypted connections made over the Internet. Because the traffic is encrypted between the device and the network, traffic remains private as it travels. An employee can work outside the office and still securely connect to the corporate network. Even smartphones and tablets can connect through a VPN.

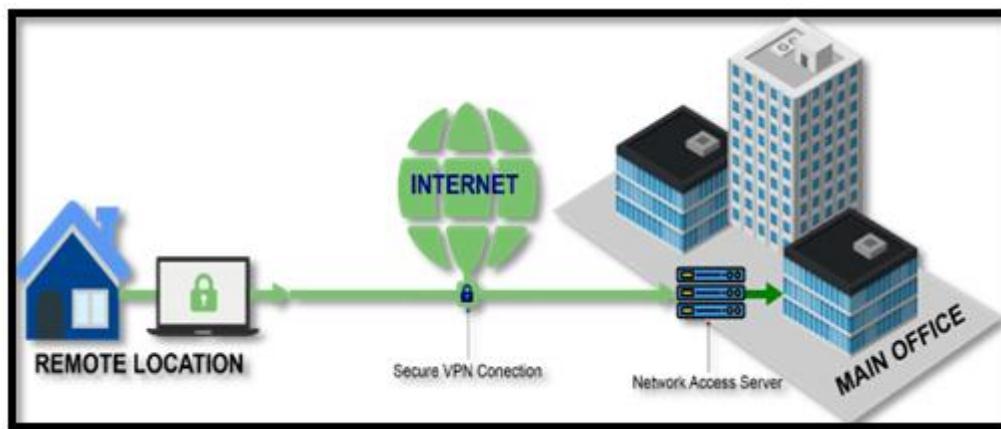


### ***What is secure remote access?***

Secure remote access provides a safe, secure way to connect users and devices remotely to a corporate network. It includes VPN technology that uses strong ways to authenticate the user or device. VPN technology is available to check whether a device meets certain requirements, also called a device's posture, before it is allowed to connect remotely.

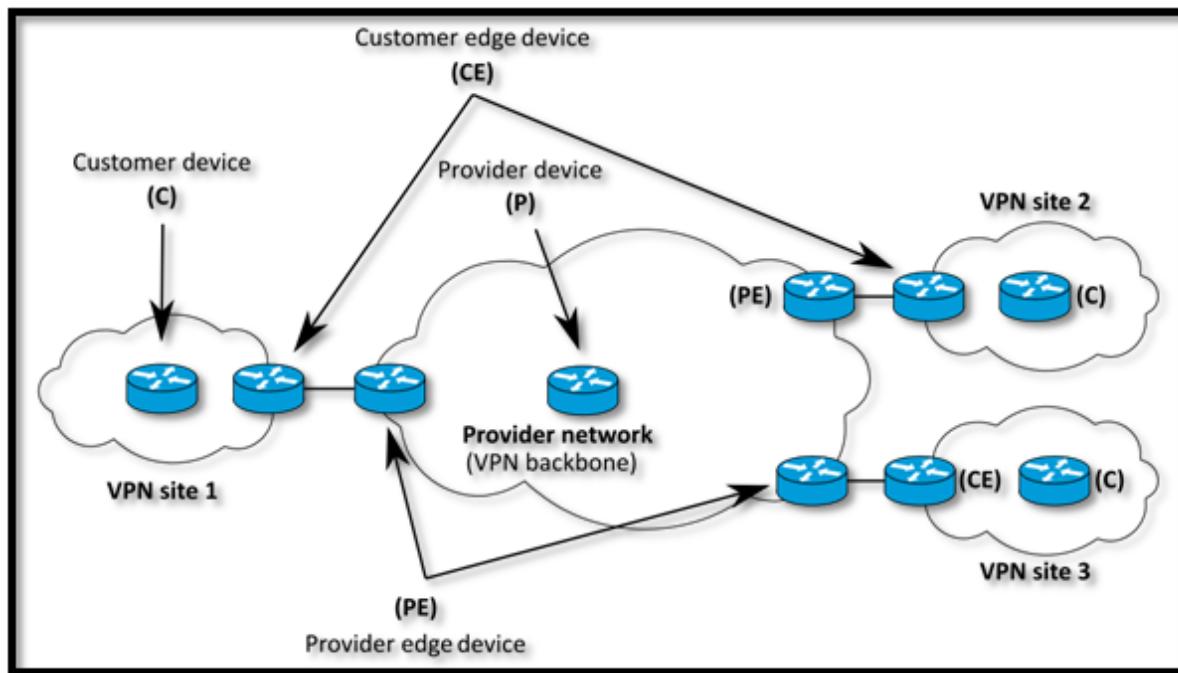
### ***What is Site-to-site?***

A site-to-site VPN connects the corporate office to branch offices over the Internet. Site-to-site VPNs are used when distance makes it impractical to have direct network connections between these offices. Dedicated equipment is used to establish and maintain a connection. Think of site-to-site access as network to network.



### ***Is VPN traffic encrypted?***

Yes, traffic on the virtual network is sent securely by establishing an encrypted connection across the Internet known as a tunnel. VPN traffic from a device such as a computer, tablet, or smartphone is encrypted as it travels through this tunnel. Offsite employees can then use the virtual network to access the corporate network.



### **What are the benefits of a VPN connection?**

A VPN connection disguises your data traffic online and protects it from external access. Unencrypted data can be viewed by anyone who has network access and wants to see it. With a VPN, hackers and cyber criminals can't decipher this data.

- **Secure encryption:** To read the data, you need an *encryption key*. Without one, it would take millions of years for a computer to decipher the code in the event of a brute force attack. With the help of a VPN, your online activities are hidden even on public networks.
- **Disguising your whereabouts:** VPN servers essentially act as your proxies on the internet. Because the demographic location data comes from a server in another country, your actual location cannot be determined. In addition, most VPN services do not store logs of your activities. Some providers, on the other hand, record your behavior, but do not pass this information on to third parties. This means that any potential record of your user behavior remains permanently hidden.
- **Access to regional content:** Regional web content is not always accessible from everywhere. Services and websites often contain content that can only be accessed from certain parts of the world. Standard connections use local servers in the country to determine your location. This means that you cannot access content at home while traveling, and you cannot access international content from home. With **VPN location spoofing**, you can switch to a server to another country and effectively “change” your location.
- **Secure data transfer:** If you work remotely, you may need to access important files on your company’s network. For security reasons, this kind of information requires a secure connection. To gain access to the network, a VPN connection is often required. VPN services connect to private servers and use encryption methods to reduce the risk of data leakage.

## ***How to protect and hide your IP address***

Hiding your IP address is a way to protect your personal information and online identity. The two primary ways to hide your IP address are:

1. Using a proxy server
2. Using a virtual private network (VPN)

A proxy server is an intermediary server through which your traffic is routed:

- The internet servers you visit see only the IP address of that proxy server and not your IP address.
- When those servers send information back to you, it goes to the proxy server, which then routes it to you.

A drawback of proxy servers is that some of the services can spy on you — so you need to trust it. Depending on which one you use, they can also insert ads into your browser.

### ***VPN offers a better solution:***

- When you connect your computer – or smartphone or tablet – to a VPN, the device acts as if it is on the same local network as the VPN.
- All your network traffic is sent over a secure connection to the VPN.
- Because your computer behaves as if it is on the network, you can securely access local network resources even when you are in another country.
- You can also use the internet as if you were present at the VPN's location, which has benefits if you are using public Wi-Fi or want to access geo-blocked websites.

### ***When should you use VPN***

Using a VPN hides your IP address and redirects your traffic through a separate server, making it much safer for you online. Situations where you might use a VPN include:

#### ***When using public Wi-Fi***

When using a public Wi-Fi network, even one that is password-protected, a VPN is advisable. If a hacker is on the same Wi-Fi network, it is easy for them to snoop on your data. The basic security that the average public Wi-Fi network employs does not provide robust protection from other users on the same network.

Using a VPN will add an extra layer of security to your data, ensuring you bypass the public Wi-Fi's ISP and encrypting all your communication.

#### ***When you are traveling***

If you are traveling to a foreign country – for example, China, where sites like Facebook are blocked – a VPN can help you access services that may not be available in that country.

The VPN will often allow you to use streaming services that you paid for and have access to in your home country, but they are not available in another because of international rights issues. Using a VPN can enable you to use the service as if you were at home. Travelers may also be able to find cheaper airfare when using a VPN, as prices can vary from region to region.

### *When you are working remotely*

This is especially relevant in the post-COVID world, where many people are working remotely. Often employers require the use of a VPN to access company services remotely for security reasons. A VPN that connects to your office's server can give you access to internal company networks and resources when you are not in the office. It can do the same for your home network while you are out and about.

### *When you just want some privacy*

Even in the comfort of your own home, using the internet for everyday purposes, using a VPN can be a good idea. Whenever you access a website, the server you connect to logs your IP address and attaches it to all the other data the site can learn about you: your browsing habits, what you click on, how long you spend looking at a particular page. They can sell this data to advertising companies who use it to tailor ads straight to you. This is why ads on the internet sometimes feel oddly personal: it's because they are. Your IP address can also be used to track your location, even when your location services are turned off. Using a VPN prevents you from leaving footprints on the web.

Don't forget your mobile devices, either. They have IP addresses too, and you probably use them in a wider variety of locations than your home computer, including public Wi-Fi hotspots. It is advisable to use a VPN on your mobile when connecting to a network you may not fully trust. You should rely on your VPN to perform one or more tasks. The VPN itself should also be protected against compromise. These are the features you should expect from a comprehensive VPN solution:

- **Encryption of your IP address:** The primary job of a VPN is to hide your IP address from your ISP and other third parties. This allows you to send and receive information online without the risk of anyone but you and the VPN provider seeing it.
- **Encryption of protocols:** A VPN should also prevent you from leaving traces, for example, in the form of your internet history, search history and cookies. The encryption of cookies is especially important because it prevents third parties from gaining access to confidential information such as personal data, financial information and other content on websites.
- **Kill switch:** If your VPN connection is suddenly interrupted, your secure connection will also be interrupted. A good VPN can detect this sudden downtime and terminate preselected programs, reducing the likelihood that data is compromised.
- **Two-factor authentication:** By using a variety of authentication methods, a strong VPN checks everyone who tries to log in. For example, you might be

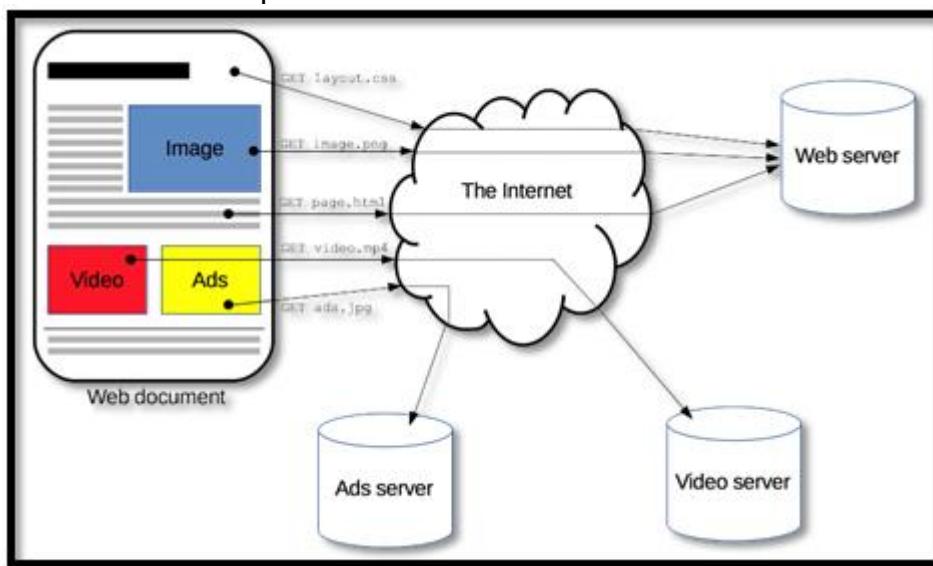
prompted to enter a password, after which a code is sent to your mobile device. This makes it difficult for uninvited third parties to access your secure connection.

## HTTP

Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents, such as HTML.

It was designed for communication between web browsers and web servers, but it can also be used for other purposes. HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response.

HTTP is a stateless protocol, meaning that the server does not keep any data (state) between two requests. Though often based on a TCP/IP layer, it can be used on any reliable transport layer, that is, a protocol that doesn't lose messages silently like UDP does. RUDP — the reliable update of UDP — is a suitable alternative.



### An overview of HTTP

HTTP is a protocol which allows the fetching of resources, such as HTML documents. It is the foundation of any data exchange on the Web and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser.

A complete document is reconstructed from the different sub-documents fetched, for instance text, layout description, images, videos, scripts, and more.

Clients and servers communicate by exchanging individual messages (as opposed to a stream of data).

The messages sent by the client, usually a Web browser, are called requests and the messages sent by the server as an answer are called responses.

### HTTP Cache

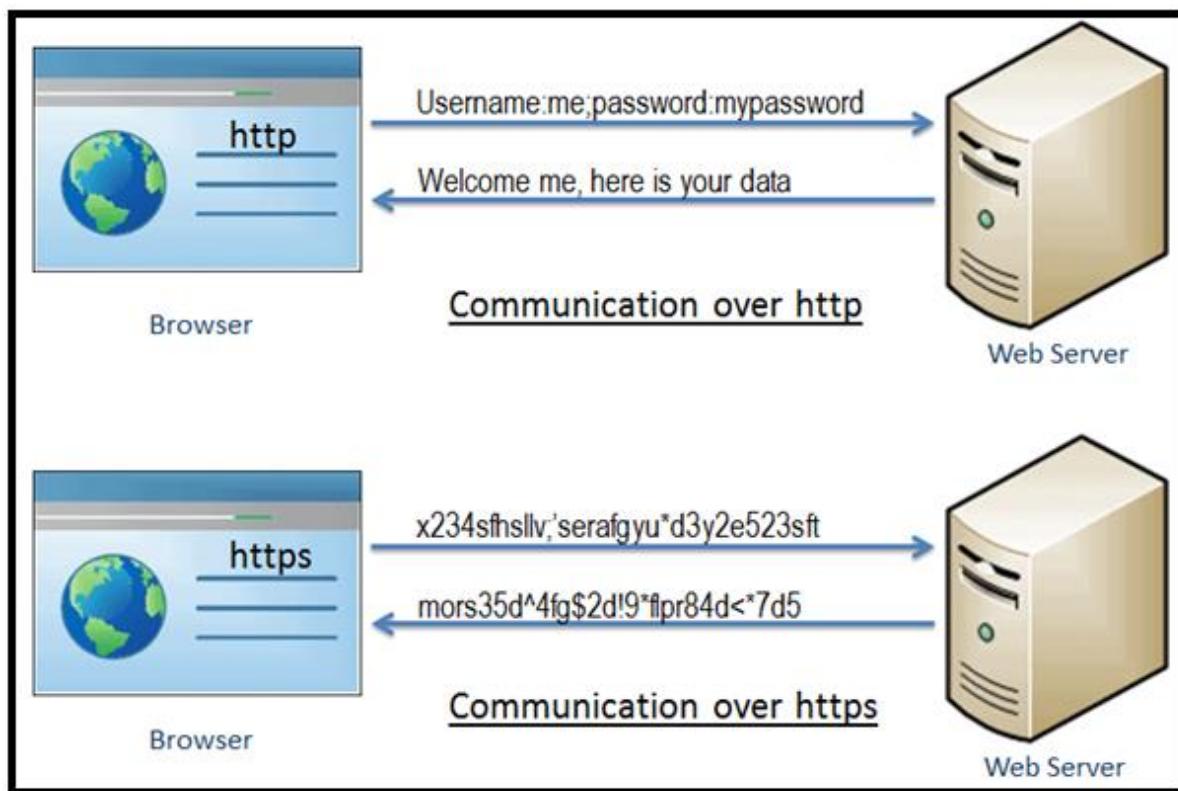
Caching is a technique that stores a copy of a given resource and serves it back when requested. When a web cache has a requested resource in its store, it intercepts the request and returns its copy instead of re-downloading from the originating server.

## HTTP cookie

An HTTP cookie (web cookie, browser cookie) is a small piece of data that a server sends to the user's web browser. The browser may store it and send it back with later requests to the same server.

## HTTPS

Hypertext Transfer Protocol Secure (https) is a combination of the Hypertext Transfer Protocol (HTTP) with the Secure Socket Layer (SSL)/Transport Layer Security (TLS) protocol. TLS is an authentication and security protocol widely implemented in browsers and Web servers. SSL works by using a public key to encrypt data transferred over the SSL connection. Most Web browsers support SSL. It allows you to communicate securely with the web server.



HTTPS ensures data security over the network - mainly public networks like Wi-Fi. HTTP is not encrypted and is vulnerable to attackers who are eavesdropping and can gain access to website database and sensitive information. By virtue, HTTPS encryption is done bi-directionally, which means that the data is encrypted at both the client and server sides. Only the client can decode the information that comes from the server. So, HTTPS does encryption of data between a client and a server, which protects against eavesdropping, forging of information and tampering of data. But how do you ensure if you are seeing an HTTPS-enabled web page? Just check the address bar that carries the site name against different background colours with a lock icon at the left corner. However, this design can be different for different browsers. For

example, consider going to a bank website, say hdfcbank.com. A non-secured HTTP will open up. But when we go to the login page, we can see an HTTPS in the address bar with some specific design. Implementation: HTTPS is mainly used by those websites which deal with monetary transactions or transfer user's personal data which could be highly sensitive. Banking websites are common examples. In layman's terms, HTTPS ensures that users watch websites that they want to watch. Data exchanged between the user and the website is not read, stolen or tampered with by a third party. But it can't encrypt everything - it has some limitations too. For example, HTTPS can't encrypt host addresses and port numbers.

## File Transfer Protocol

FTP is short for **File Transfer Protocol**. It is an application layer protocol.

### *File Transfer Protocol-*

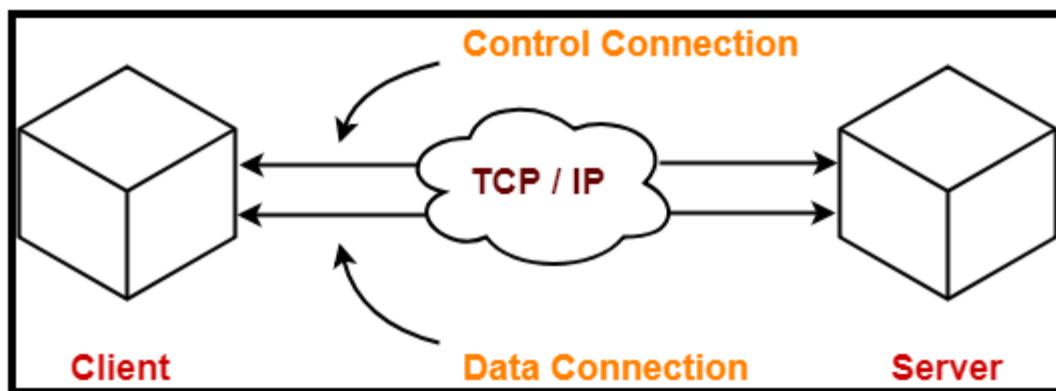
- FTP is short for **File Transfer Protocol**.
- It is an application layer protocol.

### *Purpose-*

- It is used for exchanging files over the internet.
- It enables the users to upload and download the files from the internet.

### *Working-*

FTP establishes two TCP connections between the client and the server.



### *File Transfer Protocol-*

- FTP is short for **File Transfer Protocol**.
- It is an application layer protocol.

### *Purpose-*

- It is used for exchanging files over the internet.
- It enables the users to upload and download the files from the internet.

## ***Working-***

FTP establishes two TCP connections between the client and the server.

- One connection is used for transferring data.
- Other connection is used for transferring control information.

## ***Characteristics of FTP-***

- FTP uses TCP at the transport layer.
- FTP uses port number 21 for control connection.
- FTP uses port number 20 for data connection.
- FTP uses persistent TCP connections for control connection.
- FTP uses non-persistent connections for data connection.
- FTP is a connection-oriented protocol.
- FTP is an out-of-band protocol as data and control information flow over different connections.
- SMTP is a stateful protocol.

## **SSH**

### ***Typical uses of the SSH protocol***

The protocol is used in corporate networks for:

- providing secure access for users and automated processes
- interactive and automated file transfers
- issuing remote commands
- managing network infrastructure and other mission-critical system components.

### ***How does the SSH protocol work***

The protocol works in the client-server model, which means that the connection is established by the SSH client connecting to the SSH server. The SSH client drives the connection setup process and uses public key cryptography to verify the identity of the SSH server. After the setup phase the SSH protocol uses strong symmetric encryption and hashing algorithms to ensure the privacy and integrity of the data that is exchanged between the client and server.

The figure below presents a simplified setup flow of a secure shell connection.



### ***Strong authentication with SSH keys***

There are several options that can be used for user authentication. The most common ones are passwords and public key authentication.

The public key authentication method is primarily used for automation and sometimes by system administrators for single sign-on. It has turned out to be much more widely used than we ever anticipated. The idea is to have a cryptographic key pair - public key and private key - and configure the public key on a server to authorize access and grant anyone who has a copy of the private key access to the server. The keys used for authentication are called SSH keys. Public key authentication is also used with smartcards, such as the CAC and PIV cards used by US government.

The main use of key-based authentication is to enable secure automation. Automated secure shell file transfers are used to seamlessly integrate applications and also for automated systems & configuration management.

We have found that large organizations have way more SSH keys than they imagine, and managing SSH keys has become very important. SSH keys grant access as user names and passwords do. They require a similar provisioning and termination processes.

In some cases, we have found several million SSH keys authorizing access into production servers in customer environments, with 90% of the keys actually being unused and representing access that was provisioned but never terminated. Ensuring proper policies, processes, and audits also for SSH usage is critical for proper identity and access management. Traditional identity management projects have overlooked as much as 90% of all credentials by ignoring SSH keys. We provide services and tools for implementing SSH key management.

### ***SSH provides strong encryption and integrity protection***

Once a connection has been established between the SSH client and server, the data that is transmitted is encrypted according to the parameters negotiated in the setup. During the negotiation the client and server agree on the symmetric encryption algorithm to be used and generate the encryption key that will be used. The traffic between the communicating parties is protected with industry standard strong encryption algorithms (such as AES (Advanced Encryption Standard)), and the SSH protocol also includes a mechanism that ensures the integrity of the transmitted data by using standard hash algorithms (such as SHA-2 (Standard Hashing Algorithm)).

## 5.4 DHCP

Dynamic Host Configuration Protocol (DHCP) is a network protocol that is used to configure network devices to communicate on an IP network. A DHCP client uses the DHCP protocol to acquire configuration information, such as an IP address, a default route, and one or more DNS server addresses from a DHCP server. The DHCP client then uses this information to configure its host. Once the configuration process is complete, the host is able to communicate on the Internet.

The DHCP server maintains a database of available IP addresses and configuration information. When it receives a request from a client, the DHCP server determines the network to which the DHCP client is connected, and allocates an IP address or prefix appropriate for the client, and sends configuration information appropriate for that client.

The DHCP server and DHCP client must be connected to the same network link. In larger networks, each network link contains one or more DHCP relay agents. These DHCP relay agents receive messages from DHCP clients and forward them to DHCP servers. DHCP servers send responses back to the relay agent, and the relay agent then sends these responses to the DHCP client on the local network link.

DHCP servers typically grant IP addresses to clients for a limited interval called a lease. DHCP clients are responsible for renewing their IP address before that interval has expired, and must stop using the address once the interval has expired, if they have not been able to renew it.

DHCP is used for IPv4 and IPv6. While both versions serve the same purpose, the details of the protocol for IPv4 and IPv6 are sufficiently different that they should be considered separate protocols.

### Why use DHCP?

Every device on a TCP/IP-based network must have a unique unicast IP address to access the network and its resources. Without DHCP, IP addresses for new computers or computers that are moved from one subnet to another must be configured manually; IP addresses for computers that are removed from the network must be manually reclaimed.

With DHCP, this entire process is automated and managed centrally. The DHCP server maintains a pool of IP addresses and leases an address to any DHCP-enabled client when it starts up on the network. Because the IP addresses are dynamic (leased) rather than static (permanently assigned), addresses no longer in use are automatically returned to the pool for reallocation.

The network administrator establishes DHCP servers that maintain TCP/IP configuration information and provide address configuration to DHCP-enabled clients in the form of a lease offer. The DHCP server stores the configuration information in a database that includes:

- Valid TCP/IP configuration parameters for all clients on the network.
- Valid IP addresses, maintained in a pool for assignment to clients, as well as excluded addresses.
- Reserved IP addresses associated with particular DHCP clients. This allows consistent assignment of a single IP address to a single DHCP client.
- The lease duration, or the length of time for which the IP address can be used before a lease renewal is required.

A DHCP-enabled client, upon accepting a lease offer, receives:

- A valid IP address for the subnet to which it is connecting.
- Requested DHCP options, which are additional parameters that a DHCP server is configured to assign to clients. Some examples of DHCP options are Router (default gateway), DNS Servers, and DNS Domain Name.

### *When to use a router/switch as your DHCP Server*

There are many enterprise companies who are still using DHCP for IPv4 on their routers/switches. This is typically done by the network administrator who needs to get a DHCP capability up and running quickly but does not have access to a DHCP server. Most routers/switches have the ability to provide the following DHCP server support:

- A DHCP client and obtain an interface IPv4 address from an upstream DHCP service
- A DHCP relay and forward UDP DHCP messages from clients on a LAN to and from a DHCP server
- A DHCP server whereby the router/switch services DHCP requests directly. However, there are limitations to using a router/switch as a DHCP server
- Running a DHCP server on a router/switch consumes resources on the network device. These DHCP packets are handled in software (not hardware accelerated forwarding). The resources required make this practice not suitable for a network with a large number (> 150) of DHCP clients.
- Does not support dynamic DNS. The router/switch DHCP server cannot create an entry into DNS on behalf of the client based on the IPv4 address that was leased to the client.
- No ability to easily manage the scope and see the current DHCP bindings and leases across multiple routers. Administrator must log into the switch/router individually to get information about DHCP bindings.
- No high availability or redundancy of the DHCP bindings. This could cause problems if the current DHCP server and default gateway fails.
- It is more difficult to configure DHCP options on router/switch platform.
- The DHCP service running on a router/switch is not integrated with IP address management (IPAM) for address tracking and scope utilization or security forensics.

## Benefits of DHCP

DHCP provides the following benefits.

- **Reliable IP address configuration.** DHCP minimizes configuration errors caused by manual IP address configuration, such as typographical errors, or address conflicts caused by the assignment of an IP address to more than one computer at the same time.
- **Reduced network administration.** DHCP includes the following features to reduce network administration:
  - Centralized and automated TCP/IP configuration.
  - The ability to define TCP/IP configurations from a central location.

- The ability to assign a full range of additional TCP/IP configuration values by means of DHCP options.
- The efficient handling of IP address changes for clients that must be updated frequently, such as those for portable devices that move to different locations on a wireless network.
- The forwarding of initial DHCP messages by using a DHCP relay agent, which eliminates the need for a DHCP server on every subnet.

## 5.5 Azure Virtual Networks

Azure Virtual Network (VNet) is the fundamental building block for your private network in Azure. VNet enables many types of Azure resources, such as Azure Virtual Machines (VM), to securely communicate with each other, the internet, and on-premises networks. VNet is similar to a traditional network that you'd operate in your own data center, but brings with it additional benefits of Azure's infrastructure such as scale, availability, and isolation.

Azure virtual network enables Azure resources to securely communicate with each other, the internet, and on-premises networks. Key scenarios that you can accomplish with a virtual network include - communication of Azure resources with the internet, communication between Azure resources, communication with on-premises resources, filtering network traffic, routing network traffic, and integration with Azure services.

### Demo on Azure Virtual Network

#### Creating Custom Azure Virtual Network in Azure

**Activity:** This activity allows learners to understand the concept of Azure Virtual Network in Azure cloud. This is the first step towards building a custom network for collection of resources and maintaining basic level security in cloud.

Learn how to create a virtual network using the Azure portal. You deploy two virtual machines (VMs). Next, you securely communicate between VMs and connect to VMs from the internet. A virtual network is the fundamental building block for your private network in Azure. It enables Azure resources, like VMs, to securely communicate with each other and with the internet.

#### Creating Custom VPC in Azure and add subnets to created VPC and deploy VM in subnets and set up communication between the VMs.

**Activity:** This activity allows learners to understand the concept of virtual private cloud network and create VPC in Azure cloud. This is the first step towards building a custom network for collection of resources and maintaining basic level security in cloud.

#### *Create a virtual network using the Azure portal*

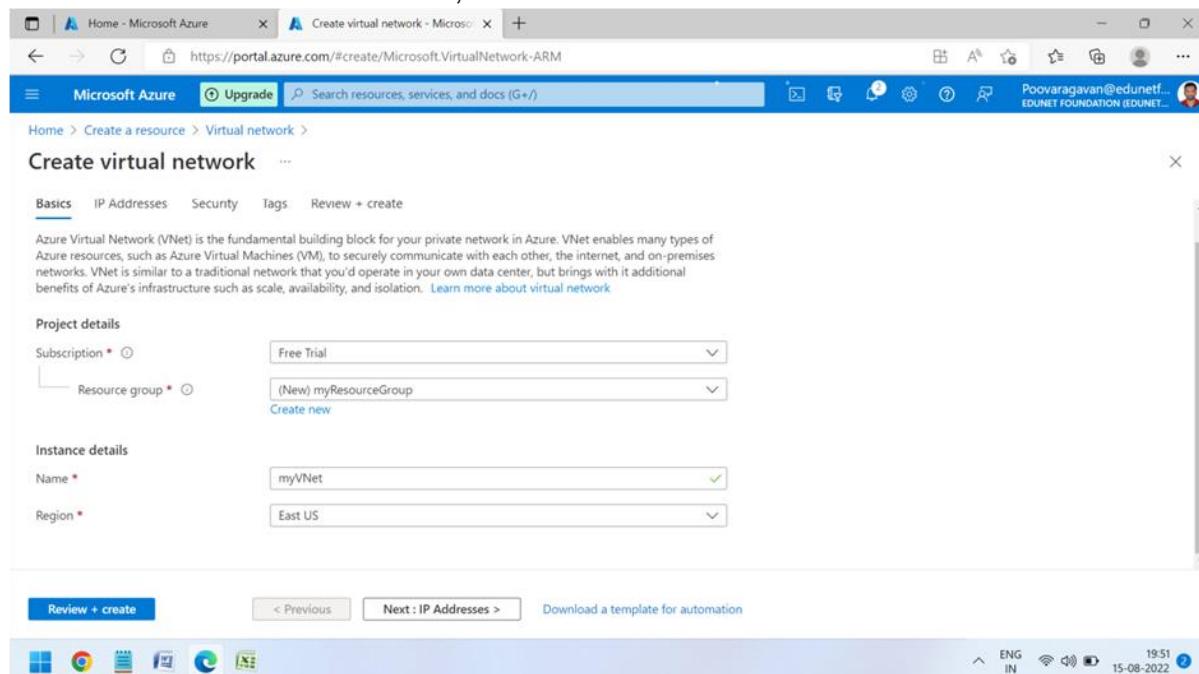
Learn how to create a virtual network using the Azure portal. You deploy two virtual machines (VMs). Next, you securely communicate between VMs and connect to VMs from the internet. A virtual network is the fundamental building block for your private network in Azure. It enables Azure resources, like VMs, to securely communicate with each other and with the internet.

### Step 1: Sign in to Azure

Sign in to the <https://portal.azure.com/>

### Step 2: Create a virtual network

- Select create a resource in the upper left-hand corner of the portal.
- In the search box, enter **Virtual Network**. Select **Virtual Network** in the search results
- In the **Virtual Network** page, select **Create**.
- In **Create** virtual network, enter or select this information in the **Basics** tab:



Setting	Value
<b>Project details</b>	
Subscription	Select your subscription.
Resource group	Select <b>Create new</b> . Enter <b>myResourceGroup</b> . Select <b>OK</b> .
<b>Instance details</b>	
Name	Enter <b>myVNet</b> .
Region	Select <b>(US) East US</b> .

- Select the **IP Addresses** tab, or select the **Next: IP Addresses** button at the bottom of the page and enter in the following information then
- Select **Add**:

[Home](#) > [Create a resource](#) > [Virtual network](#) >

## Create virtual network ...

[Basics](#) [IP Addresses](#) [Security](#) [Tags](#) [Review + create](#)

The virtual network's address space, specified as one or more address prefixes in CIDR notation (e.g. 192.168.1.0/24).

**IPv4 address space**

 Add IPv6 address space ⓘ

The subnet's address range in CIDR notation (e.g. 192.168.1.0/24). It must be contained by the address space of the virtual network.

[+ Add subnet](#) [Remove subnet](#)
 Subnet name

Subnet address range

NAT gateway

 MySubnet

10.1.0.0/24

[Review + create](#)
< Previous
Next : Security >
Download a template for automation

Setting	Value
IPv4 address space	Enter <b>10.1.0.0/16</b> .
<b>Add subnet</b>	
Subnet name	Enter <b>MySubnet</b> .
Subnet address range	Enter <b>10.1.0.0/24</b> .
Select	<b>Add.</b>

- Select the **Security** tab, or select the **Next: Security** button at the bottom of the page
- Under **BastionHost**, select **Enable**. Enter this information:

[Home](#) > [Create a resource](#) > [Virtual network](#) >

## Create virtual network ...

 BastionHost ⓘ

 Disable

 Enable

 Bastion name \*

 AzureBastionSubnet address space \*

10.1.1.0 - 10.1.1.255 (256 addresses)

 Public IP address \*

[Create new](#)
 DDoS Protection Standard ⓘ

 Disable

 Firewall ⓘ

 Standard

 Basic

 Dynamic

 Static

[OK](#)
[Cancel](#)
Download a template for automation

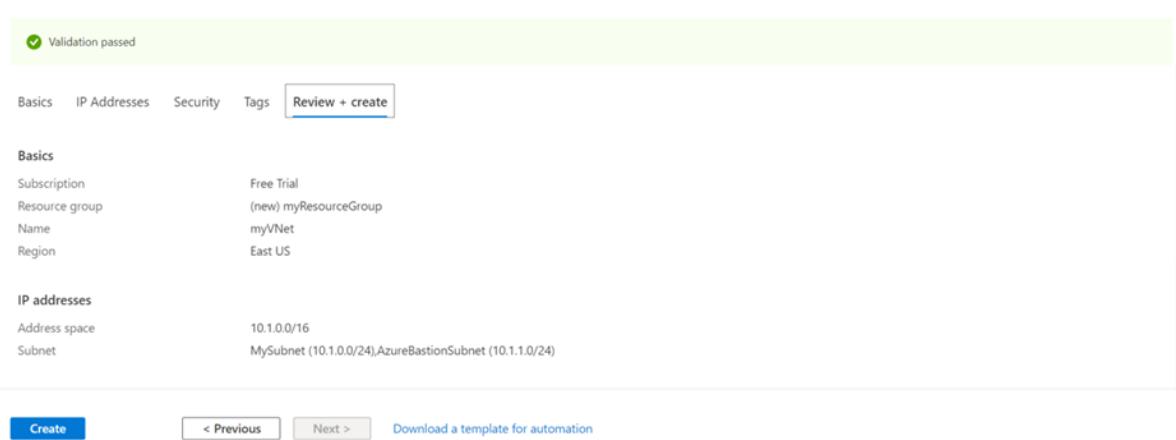
Setting	Value
Bastion name	Enter <b>myBastionHost</b>
AzureBastionSubnet address space	Enter <b>10.1.1.0/24</b>

Public IP Address	Select For Name, enter myBastionIP. Select OK.	Create new.
-------------------	---	-------------

- Select the **Review + create** tab or select the **Review + create** button.
- Select **Create**.

Home > Create a resource > Virtual network >

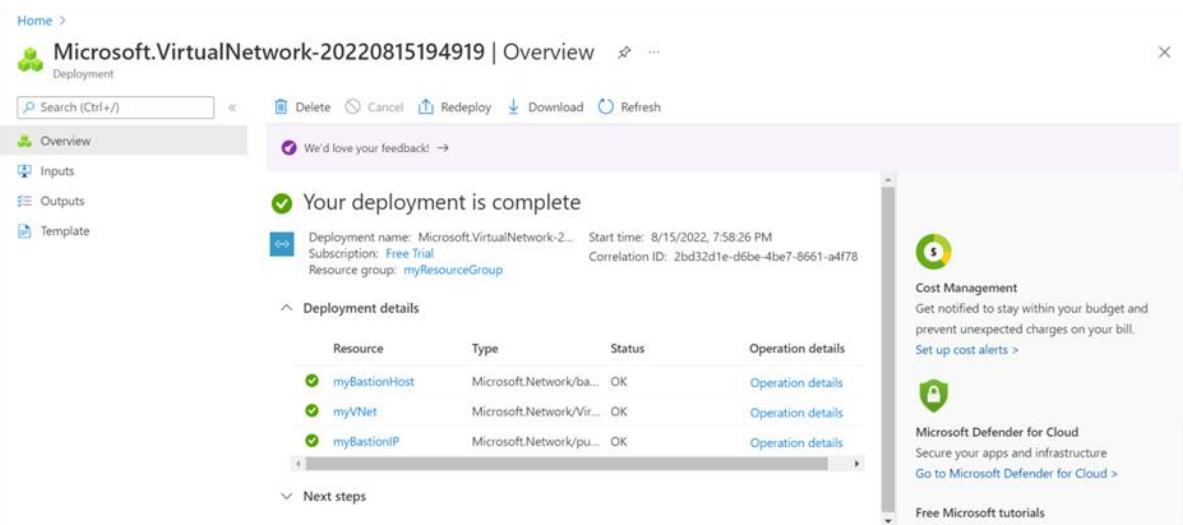
Create virtual network ...



After that, click "create" and wait for a few minutes to verify the deployment. See below the images for reference if successfully deployed.

Home >

Microsoft.VirtualNetwork-20220815194919 | Overview



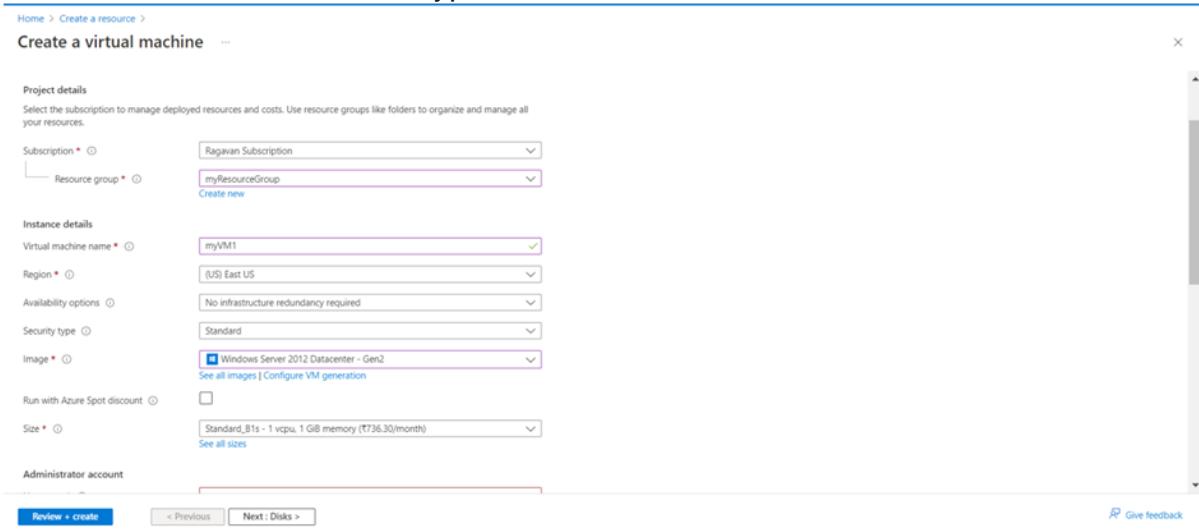
## Create virtual machines

Create two VMs in the virtual network:

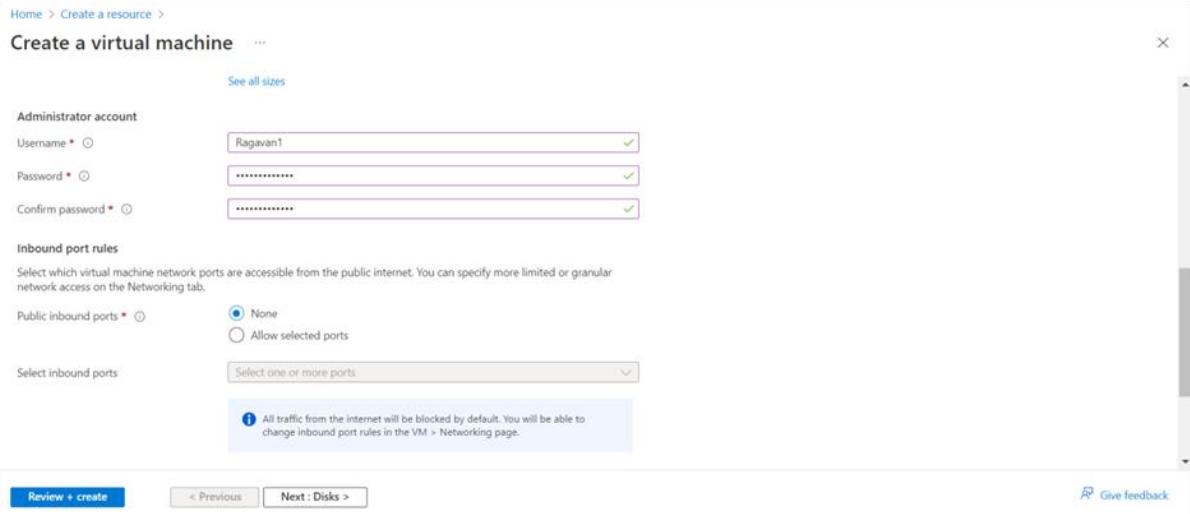
### Step 3: Create the first VM

- On the upper-left side of the portal, select **Create a resource > Compute > Virtual machine**.

- In **Create a virtual machine**, type or select the values in the **Basics** tab:



Setting	Value
<b>Project Details</b>	
Subscription	Select your Azure subscription
Resource Group	Select <b>myResourceGroup</b>
<b>Instance details</b>	
Virtual machine name	Enter <b>myVM1</b>
Region	Select <b>(US) East US</b>
Availability Options	Select <b>No infrastructure redundancy required</b>
Image	Select <b>Windows Server 2019 Datacenter - Gen2</b>
Azure Spot instance	Select <b>No</b>
Size	Choose VM size or take default setting

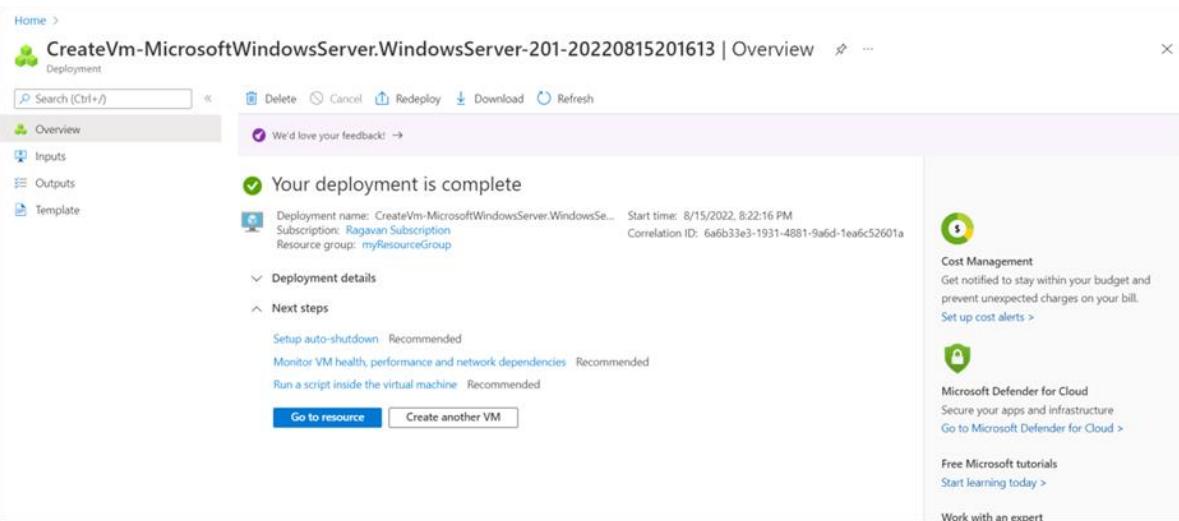
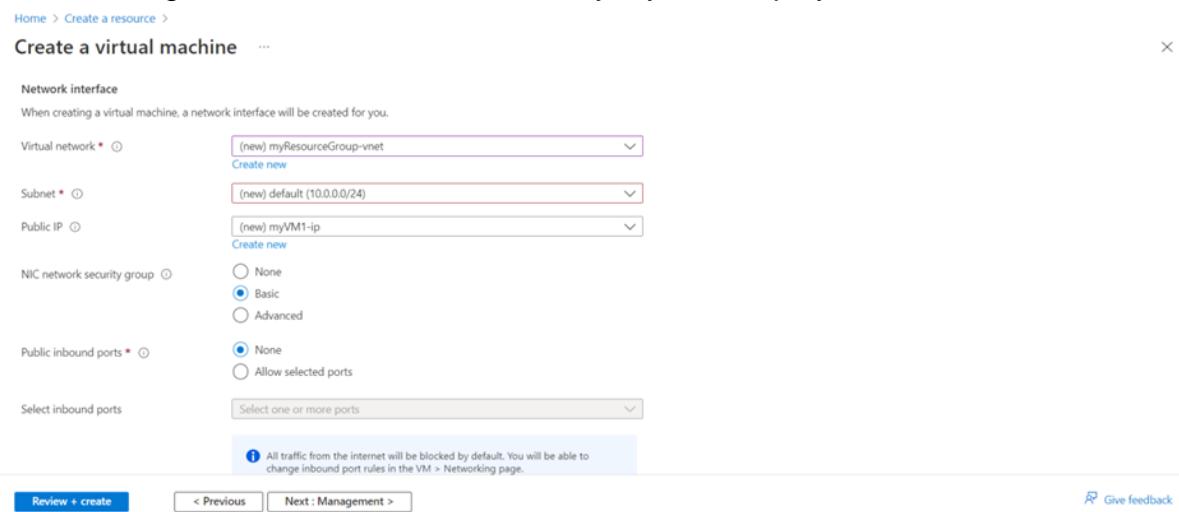


- Select the **Networking** tab, or select **Next: Disks**, then **Next: Networking**

- In the Networking tab, select or enter:

Setting	Value
<b>Network interface</b>	
Virtual network	Select <b>myVNet</b> .
Subnet	Select <b>mySubnet</b>
Public IP	Select <b>None</b>
NIC network security group	Select <b>Basic</b>
Public inbound ports network	Select <b>None</b> .

- After that, click "create" and wait for a few minutes to verify the deployment. See below the images for reference if successfully myVM1 deployed.



## Step 5: Create the second VM

- On the upper-left side of the portal, select **Create a resource > Compute > Virtual machine**
- In **Create a virtual machine**, type or select the values in the **Basics** tab:

Setting	Value
<b>Project Details</b>	
Subscription	Select your Azure subscription
Resource Group	Select <b>myResourceGroup</b>
<b>Instance details</b>	
Virtual machine name	Enter <b>myVM2</b>
Region	Select <b>(US) East US</b>
Availability Options	Select <b>No infrastructure redundancy required</b>
Image	Select <b>Windows Server 2019 Datacenter - Gen2</b>
Azure Spot instance	Select <b>No</b>
Size	Choose VM size or take default setting
<b>Administrator account</b>	
Username	Enter a username
Password	Enter a password
Confirm password	Reenter password
<b>Inbound port rules</b>	
Public inbound ports	Select <b>None.</b>

- Select the **Networking** tab, or select **Next: Disks**, then **Next: Networking**.
- In the Networking tab, select or enter

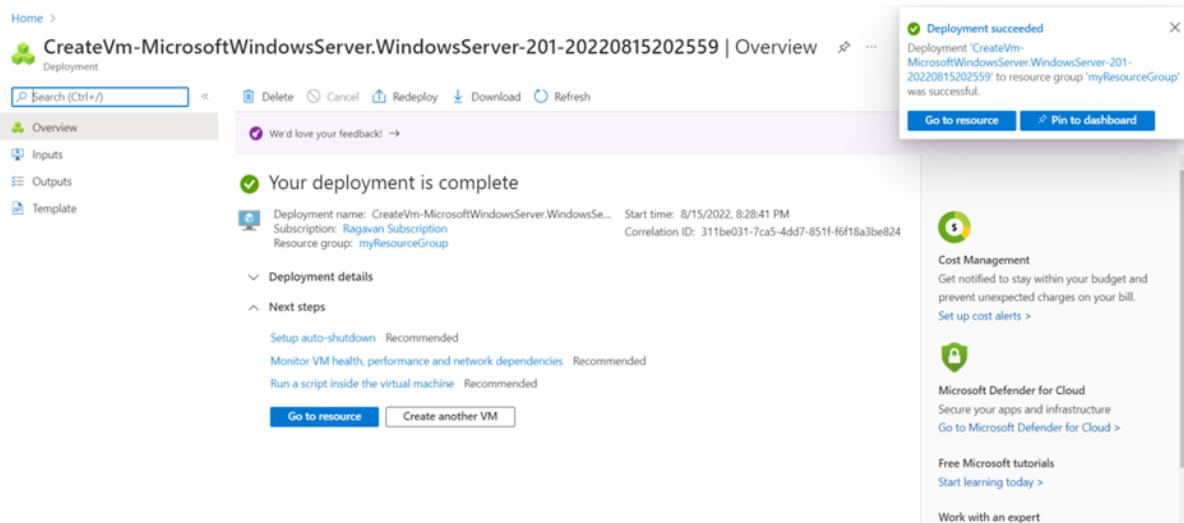
Home > Create a resource >  
Create a virtual machine ...

Network interface  
When creating a virtual machine, a network interface will be created for you.

Virtual network *	<input type="text" value="myResourceGroup-vnet"/>	<input type="button" value="Create new"/>
Subnet *	<input type="text" value="default (10.0.0.0/24)"/>	<input type="button" value="Manage subnet configuration"/>
Public IP	<input type="text" value="(new) myVM2-ip"/>	<input type="button" value="Create new"/>
NIC network security group	<input type="radio"/> None <input checked="" type="radio"/> Basic <input type="radio"/> Advanced	
Public inbound ports *	<input type="radio"/> None <input type="radio"/> Allow selected ports	
Select inbound ports	<input type="text" value="Select one or more ports"/>	

**Review + create**    < Previous    Next : Management >    

- After that, click "create" and wait for a few minutes to verify the deployment. See below the images for reference if successfully myVM2 deployed.



Your deployment is complete

Deployment name: CreateVm-MicrosoftWindowsServer.WindowsSe... Start time: 8/15/2022, 8:28:41 PM  
 Subscription: Ragavan Subscription Correlation ID: 311be031-7ca5-4dd7-851f-f6f18a3be824  
 Resource group: myResourceGroup

Deployment details

Next steps

Setup auto-shutdown Recommended  
 Monitor VM health, performance and network dependencies Recommended  
 Run a script inside the virtual machine Recommended

[Go to resource](#) [Create another VM](#)

**Cost Management**  
 Get notified to stay within your budget and prevent unexpected charges on your bill.  
[Set up cost alerts >](#)

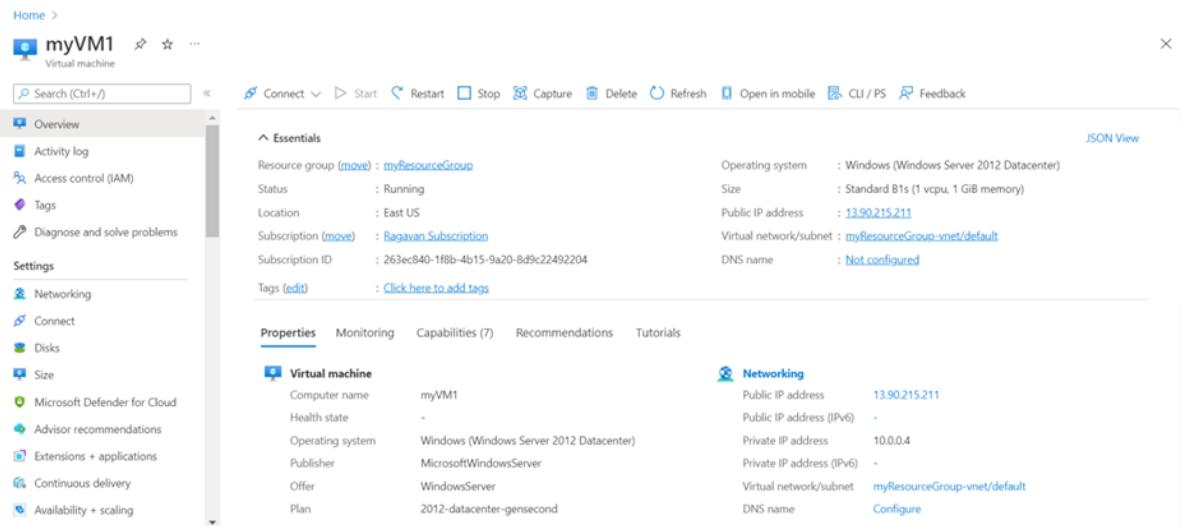
**Microsoft Defender for Cloud**  
 Secure your apps and infrastructure  
[Go to Microsoft Defender for Cloud >](#)

**Free Microsoft tutorials**  
[Start learning today >](#)

**Work with an expert**

## Step 6: Connect to myVM1

- Go to the [Azure portal](#) to manage your private VM. Search for and select **Virtual machines**.
- Pick the name of your private virtual machine **myVM1**.
- In the VM menu bar, select **Connect**, then select **Bastion**.



**myVM1** Virtual machine

Search (Ctrl+ /)

Connect Start Restart Stop Capture Delete Refresh Open in mobile CLI / PS Feedback

Overview Activity log Access control (IAM) Tags Diagnose and solve problems

Settings Networking Connect Disks Size Microsoft Defender for Cloud Advisor recommendations Extensions + applications Continuous delivery Availability + scaling

**Essentials**

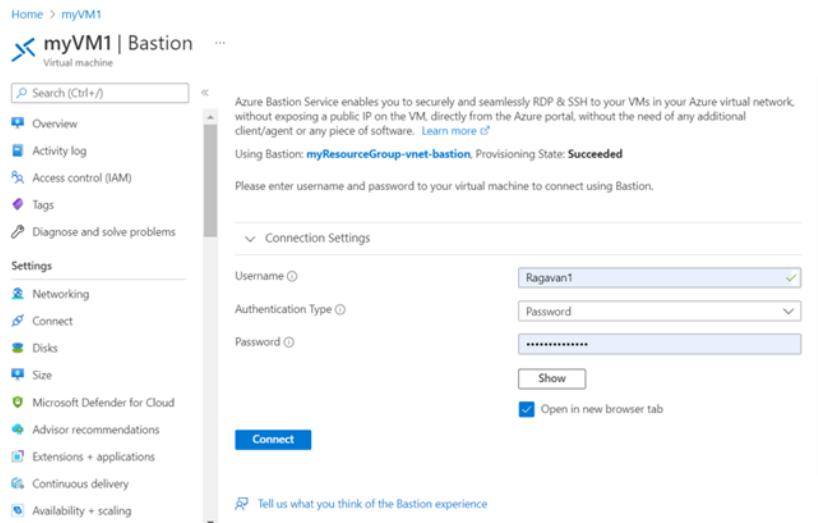
Resource group ( <a href="#">move</a> )	: myResourceGroup	Operating system	: Windows (Windows Server 2012 Datacenter)
Status	: Running	Size	: Standard B1s (1 vcpu, 1 GiB memory)
Location	: East US	Public IP address	: <a href="#">13.90.215.211</a>
Subscription ( <a href="#">move</a> )	: Ragavan Subscription	Virtual network/subnet	: myResourceGroup-vnet/default
Subscription ID	: 263ec840-1fb8-4b15-9a20-8d9c22492204	DNS name	: Not configured
Tags ( <a href="#">edit</a> )	: <a href="#">Click here to add tags</a>	JSON View	

**Properties** **Monitoring** **Capabilities (7)** **Recommendations** **Tutorials**

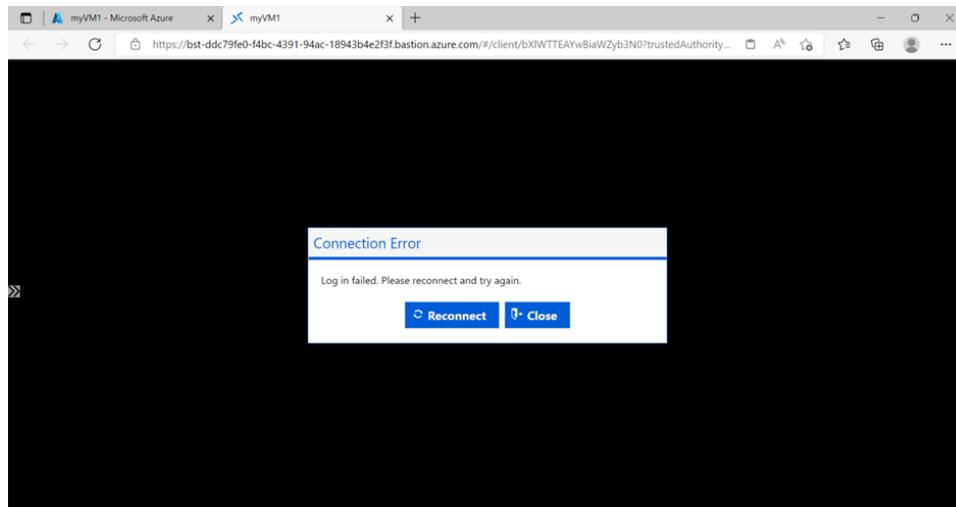
**Virtual machine**

Computer name	myVM1	Networking	
Health state	-	Public IP address	<a href="#">13.90.215.211</a>
Operating system	Windows (Windows Server 2012 Datacenter)	Public IP address (IPv6)	-
Publisher	MicrosoftWindowsServer	Private IP address	10.0.0.4
Offer	WindowsServer	Private IP address (IPv6)	-
Plan	2012-datacenter-gensecond	Virtual network/subnet	myResourceGroup-vnet/default
		DNS name	<a href="#">Configure</a>

- In the **Connect** page, select the blue **Use Bastion** button.
- In the **Bastion** page, enter the username and password you created for the virtual machine previously.
- Select **Connect**.



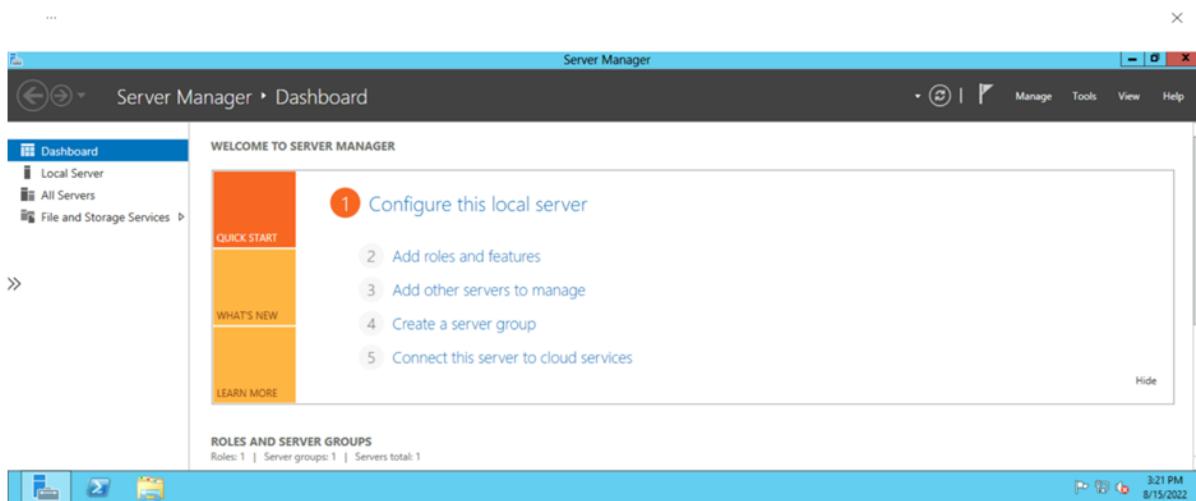
- After Connect enable bastion service is required to Access Virtual machines with registered credentials.
- After Click Connect Popup window opens in browser. In browser we can see virtual machine and access it.



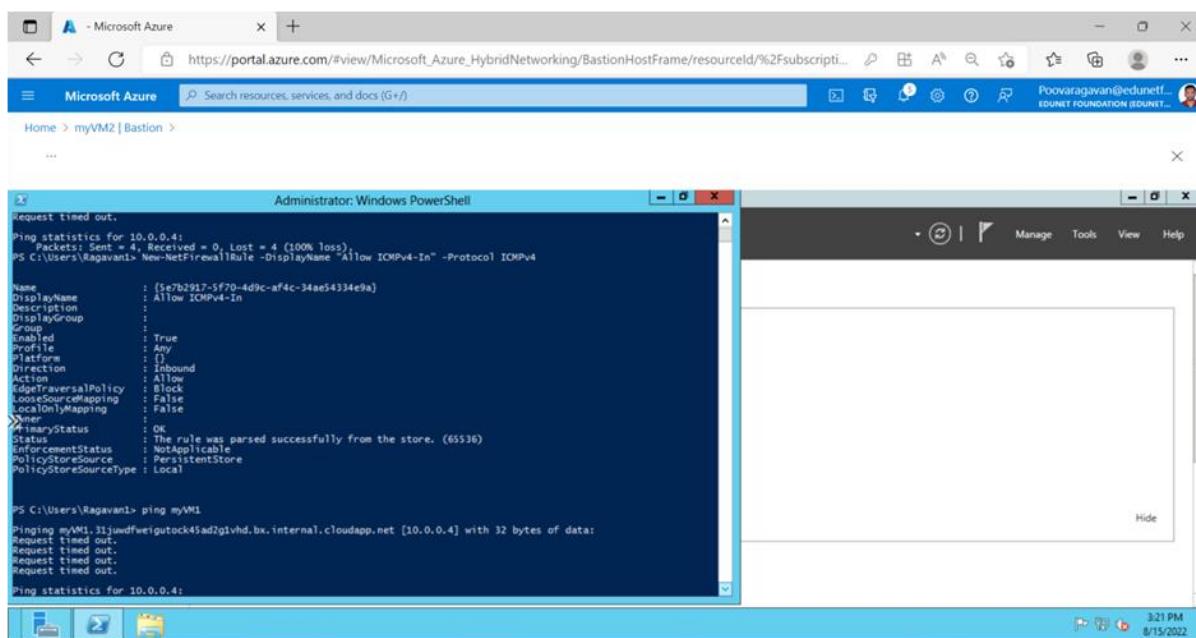
## Step 7: Communicate between VMs

- In the bastion connection of **myVM1**, open PowerShell.
- Enter ping myvm2.
- You'll receive a message similar to this output:

Home > myVM2 | Bastion >



- Close the bastion connection to **myVM1**.
- Complete the steps in [Connect to myVM1](#), but connect to **myVM2**.
- Open PowerShell on **myVM2**, enter ping myvm1.
- You'll receive something like this message:

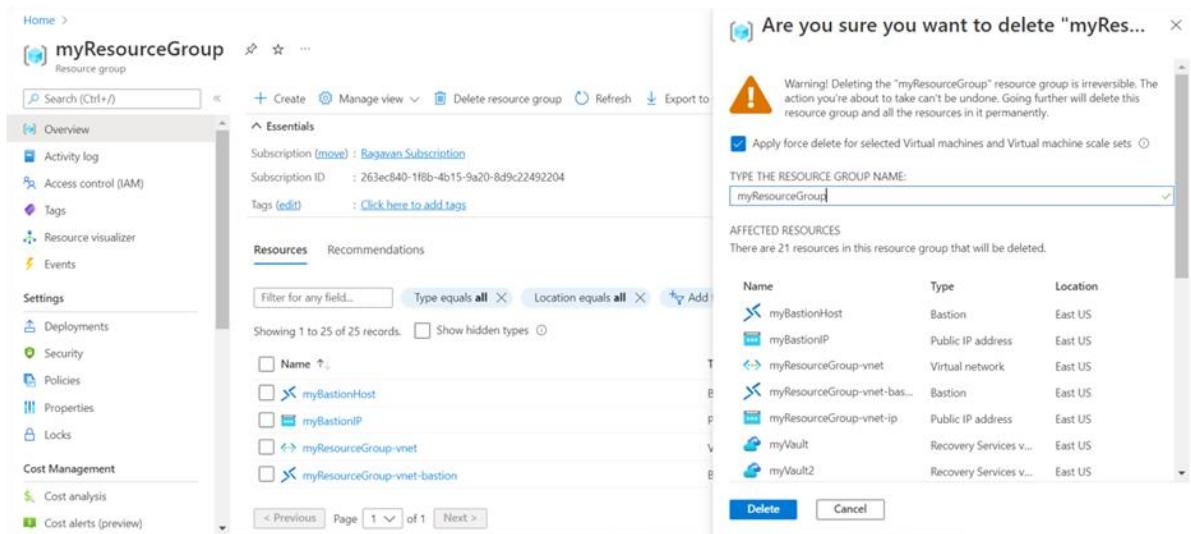


- Close the bastion connection to **myVM2**.

## Step 8: Clean up resources

In this QuickStart, you created a default virtual network and two VMs. You connected to one VM from the internet and securely communicated between the two VMs. When you're done using the virtual network and the VMs, delete the resource group and all of the resources it contains:

- Search for and select **myResourceGroup**.
- Select **Delete resource group**.
- Enter **myResourceGroup** for **TYPE THE RESOURCE GROUP NAME** and select **Delete**.



The screenshot shows the Azure portal interface. On the left, the 'myResourceGroup' resource group is selected. The main area displays various resources within the group. A modal window titled 'Are you sure you want to delete "myRes..."' is open, containing a warning message: 'Warning! Deleting the "myResourceGroup" resource group is irreversible. The action you're about to take can't be undone. Going further will delete this resource group and all the resources in it permanently.' Below the warning, there is a checked checkbox labeled 'Apply force delete for selected Virtual machines and Virtual machine scale sets'. A text input field shows 'myResourceGroup'. At the bottom of the modal, there are 'Delete' and 'Cancel' buttons. The 'Affected Resources' section lists 21 resources, including myBastionHost, myBastionIP, myResourceGroup-vnet, myResourceGroup-vnet-bast..., myVault, and myVault2.

- Click Delete and wait for few minutes. Resource clean-up Successfully.

## Demo on Creating Your First Linux VM in Azure cloud and prepare it for Development Environment

Azure virtual machines (VMs) can be created through the Azure portal. The Azure portal is a browser-based user interface to create Azure resources. This quick start shows you how to use the Azure portal to deploy a Linux virtual machine (VM) running Ubuntu 18.04 LTS. To see your VM in action, you also SSH to the VM and install the NGINX web server.

### *Sign in to Azure*

Sign in to the [Azure portal](#) if you haven't already.

### *Create virtual machine*

1. Type **virtual machines** in the search.
2. Under **Services**, select **Virtual machines**.
3. In the **Virtual machines** page, select **Add**. The **Create a virtual machine** page opens.
4. In the **Basics** tab, under **Project details**, make sure the correct subscription is selected and then choose to **Create new resource group**. Type *myResourceGroup* for the name.\*.

**Project details**

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	Pay-As-You-Go
Resource group *	(New) myResourceGroup
<a href="#">Create new</a>	

- Under **Instance details**, type *myVM* for the **Virtual machine name**, choose *East US* for your **Region**, and choose *Ubuntu 18.04 LTS* for your **Image**. Leave the other defaults.

**Instance details**

Virtual machine name *	myVM
Region *	(US) East US
Availability options	No infrastructure redundancy required
Image *	Ubuntu Server 18.04 LTS
<a href="#">Browse all public and private images</a>	
Size *	<b>Standard D2s v3</b> 2 vcpus, 8 GiB memory <a href="#">Change size</a>

- Under **Administrator account**, select **SSH public key**.
- In **Username** type *azureuser*.
- For **SSH public key source**, leave the default of **Generate new key pair**, and then type *myKey* for the **Key pair name**.

**Administrator account**

Authentication type	<input checked="" type="radio"/> SSH public key <input type="radio"/> Password
Username *	azureuser
SSH public key source	Generate new key pair
Key pair name *	myKey

- Under **Inbound port rules > Public inbound ports**, choose **Allow selected ports** and then select **SSH (22)** and **HTTP (80)** from the drop-down.

Inbound port rules

Select which virtual machine network ports are accessible from the public internet. You can specify more limited or granular network access on the Networking tab.

Public inbound ports \* ⓘ

None  Allow selected ports

Select inbound ports \*

HTTP (80), SSH (22)

**⚠️** This will allow all IP addresses to access your virtual machine. This is only recommended for testing. Use the Advanced controls in the Networking tab to create rules to limit inbound traffic to known IP addresses.

10. Leave the remaining defaults and then select the **Review + create** button at the bottom of the page.
11. On the **Create a virtual machine** page, you can see the details about the VM you are about to create. When you are ready, select **Create**.
12. When the **Generate new key pair** window opens, select **Download private key and create resource**. Your key file will be download as **myKey.pem**. Make sure you know where the .pem file was downloaded, you will need the path to it in the next step.
13. When the deployment is finished, select **Go to resource**.
14. On the page for your new VM, select the public IP address and copy it to your clipboard.

Operating system	:	Linux (ubuntu 18.04 LTS)
Size	:	Standard D2s v3 (2 vCPUs, 8 GB memory)
Public IP address	:	<a href="#">10.111.12.123</a> 

## Connect to virtual machine

Create an SSH connection with the VM.

1. If you are on a Mac or Linux machine, open a Bash prompt. If you are on a Windows machine, open a PowerShell prompt.
2. At your prompt, open an SSH connection to your virtual machine. Replace the IP address with the one from your VM, and replace the path to the .pem with the path to where the key file was downloaded.

Console

```
ssh -i .\Downloads\myKey1.pem azureuser@10.111.12.123
```

### Tip

The SSH key you created can be used the next time you create a VM in Azure. Just select the **Use a key stored in Azure for SSH public key source** the next time you

create a VM. You already have the private key on your computer, so you won't need to download anything.

### **Install web server**

To see your VM in action, install the NGINX web server. From your SSH session, update your package sources and then install the latest NGINX package.

Bash

```
sudo apt-get -y update  
sudo apt-get -y install nginx
```

When done, type exit to leave the SSH session.

### **View the web server in action**

Use a web browser of your choice to view the default NGINX welcome page. Type the public IP address of the VM as the web address. The public IP address can be found on the VM overview page or as part of the SSH connection string you used earlier.

### **Clean up resources**

When no longer needed, you can delete the resource group, virtual machine, and all related resources. To do so, select the resource group for the virtual machine, select **Delete**, then confirm the name of the resource group to delete.

## **5.6 Market and Job Trends**

**Market Trends:** Cloud computing is a newly developing paradigm of distributed computing. Virtualization in combination with utility computing model can make a difference in the IT industry and as well as in social perspective. Though cloud computing is still in its infancy but it's clearly gaining momentum. Organizations like Microsoft, Google, Yahoo, and Amazon are already providing cloud services.

The products like Microsoft Azure, Google App-Engine and Amazon EC2 are capturing the market with their ease of use, availability aspects and utility computing model. Users don't have to be worried about the hinges of distributed programming as they are taken care of by the cloud providers. They can devote more on their own domain work rather than these administrative works.

Business organizations are also showing increasing interest to indulge themselves into using cloud services. There are many open research issues in this domain like security aspect in the cloud, virtual machine migration, dealing with large data for analysis purposes etc. In developing countries like India cloud computing can be applied in the e-governance and rural development with great success. Although as we have seen there are some crucial issues to be solved to successfully deploy cloud computing for these social purposes.

The cloud services market size was valued at \$264.80 billion in 2019, and is projected to reach \$927.51 billion by 2027, growing at a CAGR of 16.4% from 2020 to 2027.

Cloud computing refers to the model or network where a program or applications run, which can be accessed by many devices or servers at a time.

Cloud computing technology is a shift in the tradition of computing, which has given newer and faster methods to provide computing solutions, infrastructure solutions, and application layers. The cloud services market report focuses on the emerging cloud computing technology and its application.

It also gives a comparative analysis of the cloud computing technology with the conventional technology and describes how the cloud computing technology scores an upper hand than the conventional technology. The cloud computing technology comprises of both hardware as well as the software through which the services are delivered. This report contains only the services category and excludes the hardware.

Hence, the report also focuses on the cloud services market opportunities. There lies a great potential in the cloud computing services market due to several benefits such as access to broader network, on demand service, pay as you go benefits, resource

pooling, business agility, rapid elasticity, cost cutting, and others. The global adoption of cloud computing services in various sectors such as medical & healthcare, banking financial services & insurance, and educational sector with the help of various deployment models determines the scope of further increase in the global cloud computing services market.



Image: Market Trends of Cloud Computing  
Reference: [T811175985\\_g.jpg \(650x405\) \(openpr.com\)](#)

### Job Trends

There are number of job roles available if you get skilled on the cloud computing. Let us discuss one by one.

#### Cloud Administrator:

- A cloud admin should be well versed with system management, troubleshooting and virtualization.
- Along with windows OS, Linux is a must
- Along with that, you also need to be familiar with some configuration management tools, monitoring tools and scripting languages.
- Cloud administrators should have strong leadership and people skills.

### Cloud Architect:

- A cloud architect deals with infrastructure design and configuration.
- This is not an entry level job. The candidate should bring eight to 10 years of experience.
- You should be able to build a roadmap for the organization's existing and future cloud assets.
- New technologies can affect the company's cloud infrastructure. A cloud architect must see how changing technologies and trends will affect their systems.
- It is helpful to have few certifications on a certain product like Microsoft Azure.

### Cloud Engineer:

- This is a technical responsibility job.
- Companies look for someone with three to five years of cloud services experience as a fit for the position.
- You should be well familiar with open-source technology, scripting languages, multi-cloud environments, system engineering and software development.
- Familiarity with APIs, orchestration and automation, DevOps and databases are considered as plus points for cloud engineers on top of their computer science or engineering degrees.
- The main responsibilities include the design, planning and management of the cloud infrastructure.

### Cloud Security Manager:

- Security is a big concern for both private and public clouds.
- Because of this, cloud security roles are crucial for IT teams within companies.
- Cloud security managers should have completed formal training and acquired vendor-neutral certifications.
- To be competitive, a cloud security manager candidate should have a strong understanding of compliance issues and IT governance related to the cloud.
- A cloud security manager typically designs, implements and maintains security strategies.
- The role is often incorporated into software development.

### Cloud Application Developer:

- A cloud application developer has to be proficient in most -- if not all -- major scripting languages. Software tools knowledge is also a must.
- Cloud application developers need to know the back-end system integrations with the major cloud platforms.
- A candidate will be responsible to build, test and deploy applications in a company's cloud environment. These tasks often implement DevOps practices, as well as CI/CD tools.

### Cloud Network Engineer:

- A cloud network engineer is a multi-tasking role.
- A candidate is primarily responsible for the implementation, configuration, maintenance and support of the entire cloud network.
- Cloud network engineers may also be in charge of the administration, monitoring, documentation, security and integration of cloud services.

**Cloud Automation Engineer:**

- This role, is not for inexperienced candidates. This is usually a senior position
- An automation engineer has to have experience from software development or IT operations positions.
- Cloud automation engineer role requires detailed understanding of hardware and software, as well as data center and cloud infrastructure.
- A cloud automation engineer implements, optimizes and supports an infrastructure.

All the above job roles require hands on experience of cloud platforms.

Apart from these, it is always beneficial to know cloud application development and deployment, DevOps and integrating various applications and services with cloud. In every field, there is a good chance that your organization may migrate to cloud, to reap the benefits of cloud revolution.