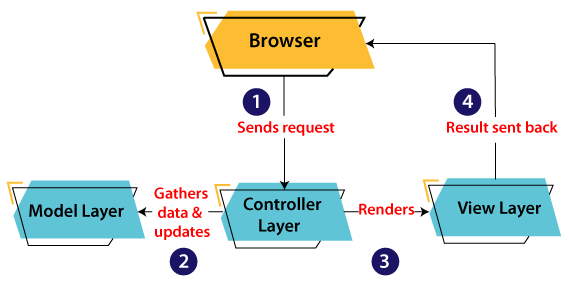
MVC Architecture in Java

* **Model:** It represents the business layer of application. It is an object to carry the data that can also contain the logic to update controller if data is changed.
* **View:** It represents the presentation layer of application. It is used to visualize the data that the model contains.
* **Controller:** It works on both the model and view. It is used to manage the flow of application, i.e. data flow in the model object and to update the view whenever data is changed.

In Java Programming, the Model contains the simple [Java classes](https://www.javatpoint.com/object-and-class-in-java)

, the View used to display the data and the Controller contains the [servlets](https://www.javatpoint.com/servlet-tutorial)

. Due to this separation the user requests are processed as follows:



1. A client (browser) sends a request to the controller on the server side, for a page.
2. The controller then calls the model. It gathers the requested data.
3. Then the controller transfers the data retrieved to the view layer.
4. Now the result is sent back to the browser (client) by the view.

# How to build a Web Application Using Java

 A web application is computer software that utilizes the web browser and technologies to perform tasks over the internet. A web application is deployed on a web server.

[Java](https://www.javatpoint.com/java-tutorial)

provides some technologies like [Servlet](https://www.javatpoint.com/servlet-tutorial)

and [JSP](https://www.javatpoint.com/jsp-tutorial)

that allow us to develop and deploy a web application on a server easily. It also provides some frameworks such as Spring, Spring Boot that simplify the work and provide an efficient way to develop a web application. They reduce the effort of the developer.

We can create a website using static [HTML](https://www.javatpoint.com/html-tutorial)

pages and style them using [CSS](https://www.javatpoint.com/css-tutorial)

, but we need server-side technology when we want to create a dynamic website.

In this section, we will see how to create a website using Java Servlets and HTML. Further, we will see how these technologies are useful for developing a web application.

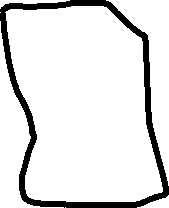
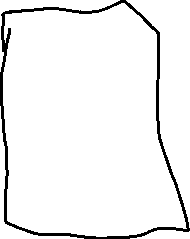
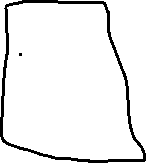
Let's understand the components of a web application:

### **What is a Web Application**

A web application is computer software that can be accessed using any web browser. Usually, the frontend of a web application is created using the scripting languages such as HTML, CSS, and JavaScript, supported by almost all web browsers. In contrast, the backend is created by any of the programming languages such as Java, Python, Php, etc., and databases. Unlike the mobile application, there is no specific tool for developing web applications; we can use any of the supported IDE for developing the web application.



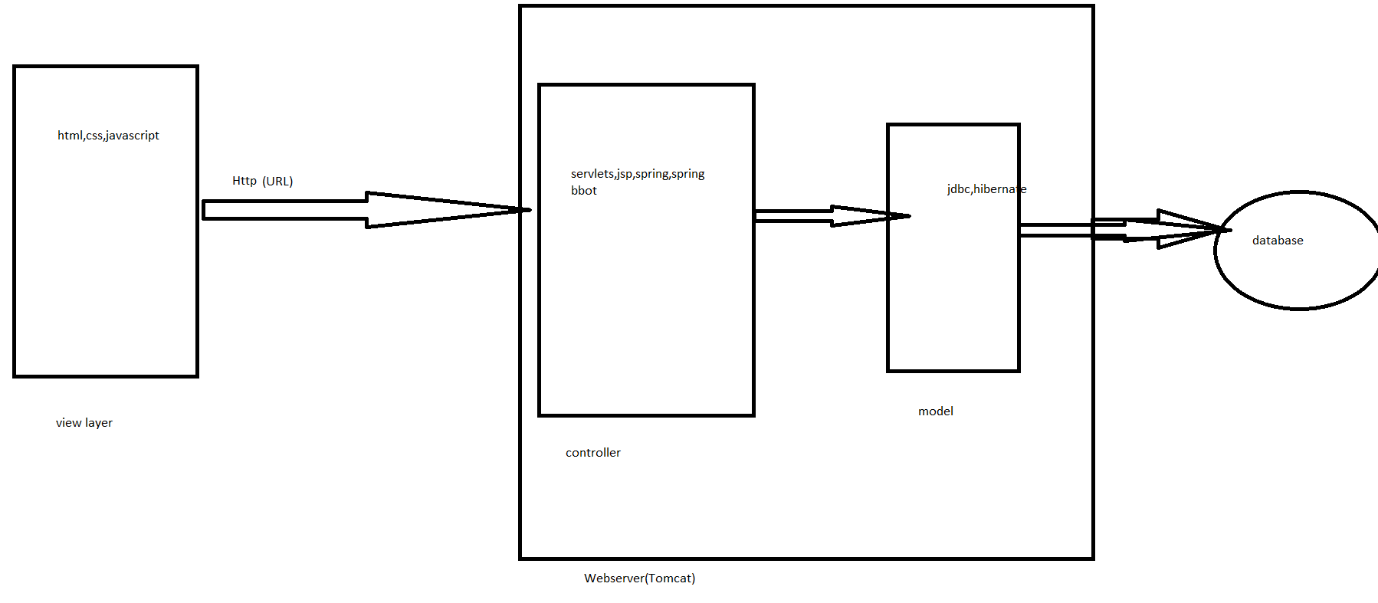
html, servlets,



javascript,

css





### **Web Server and Client**

The web server is a process that handles the client's request and responds. It processes the request made by the client by using the related protocols. The main function of the webserver is to store the request and respond to them with web pages. It is a medium between client and server. For example, Apache is a leading webserver.

A client is a software that allows users to request and assist them in communicating with the server. The web browsers are the clients in a web application; some leading clients are Google Chrome, Firefox, Safari, Internet Explorer, etc.

### **HTML and HTTP**

The HTML stands for HyperText Markup Language; it is a common language for Web Server and Web Client communication. Since both the web server and web client are two different software components of the web, we need a language that communicates between them.

The HTTP stands for HyperText Transfer Protocol; it is a communication protocol between the client and the server. It runs on top of the TCP/IP protocol.

Some of the integral components of an HTTP Request are as following:

**HTTP Method:** The HTTP method defines an action to be performed; usually, they are GET, POST, PUT, etc.

**URL:** URL is a web address that is defined while developing a web application. It is used to access a webpage.

**Form Parameters:** The form parameter is just like an argument in a Java method. It is passed to provide the details such as user, password details on a login page.

### **What is URL**

URL stands for Universal Resource Locator used to locate the server and resource. It is an address of a web page. Every web page on a project must have a unique name.

A URL looks like as follows:

1. http://localhost:8080/SimpleWebApplication/

Where,

**http or https:** It is the starting point of the URL that specifies the protocol to be used for communication.

**Localhost:** The localhost is the address of the server. When we run our application locally, it is called localhost; if we deployed our project over the web, then it is accessed by using the domain name like "javatpoint.com". The domain name maps the server to IP addresses.

**8080:** This is the port number for the local server; it is optional and may differ in different machines. If we do not manually type the port number in the URL, then by default, the request goes to the default port of the protocol. Usually, the port no between 0 to 1023 are reserved for some well-known services such as HTTP, HTTPS, FTP, etc.

We have discussed all the major components of a web application. Let's move towards our main motive How to build a web application in Java.

First, understand servlet:

### **What is Servlet**

A Servlet is a Java program that runs within a web server; it receives the requests and responds to them using related protocols (Usually HTTP). The Servlets are capable enough to respond to any type of request; they are commonly used to make the application functional.

We can create a static website using only HTML and CSS, but when it comes to dynamic, we need a server-side programming language. For these applications, Java provides Servlet technology, which contains HTTP-specific servlet classes.

The **javax.servlet** and **javax.servlet.http** packages contain interfaces and classes for creating servlets. All servlets should implement the Servlet interface, which defines life-cycle methods. To implement a generic service, we can use the GenericServlet class by extending it. It provides **doGet** and **doPost** methods to handle HTTP-specific services.

### **Why are the Servlets Useful?**

Web servers are capable enough to serve static HTML requests, but they don't know how to deal with dynamic requests and databases. So, we need a language for dynamic content; these languages are PHP, Python, Java, Ruby on Rails, etc. In Java, there are two technologies Servlet and JSPs, that deals with dynamic content and database. Java also provides frameworks such as Spring, Spring Boot, Hibernate, and Struts to use the servlet and JSP easily.

The Servlets and JSPs are server-side technologies that extend the functionality of a web server. They support dynamic response and data persistence. We can easily create a web application using these technologies.