<u>Introduction</u>

In general by using Java technology we are able to design following 2 types of applications.

Java

Standalone Applications   Enterprise Applications

CUI App   GUI App   Web App   Distributed App
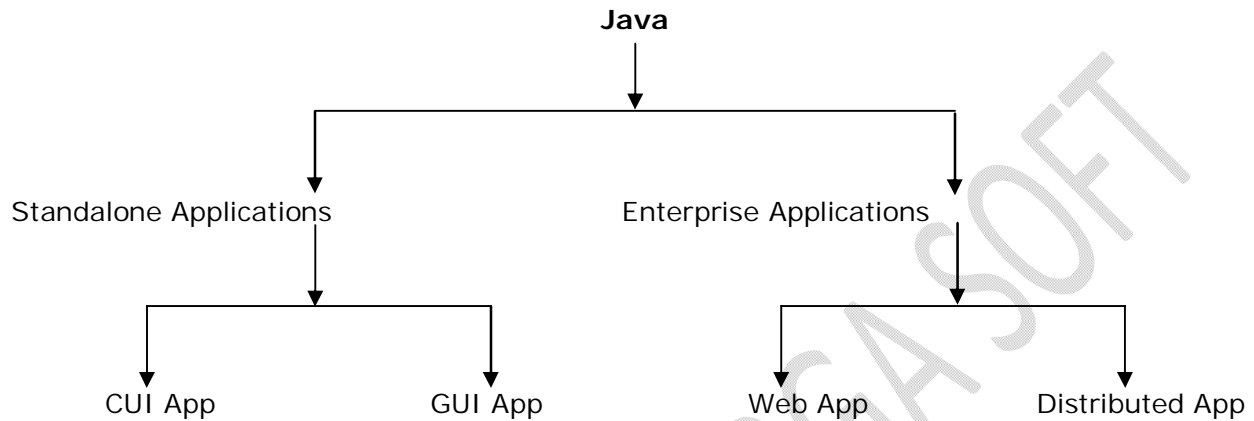
1. Standalone Applications:

These are the java applications, which will be designed without using Client-Server Architecture.

There are 2 types of Standalone applications.

1. CUI Applications
2. GUI Applications

Q: What are the differences between CUI Applications and GUI Applications?

Ans: CUI Applications are the standalone applications, which will be designed in such a way to take input and to provide output by using Command prompt, where Command prompt is acting as user interface and it able to support character data. So that the java application which will design on the basis of command prompt i.e. character user interface is called as **CUI application**.

GUI Applications are the standalone applications, which will be designed in such a way to take input and to provide output by using a collection of Graphic component, where the collection of Graphic components is acting as an user interface and it able to support GUI component so that the java application which will be design on the basis of Graphical user interface is called as **GUI Application**.

2. Enterprise Applications:

These are the java applications, which will be designed on the basis of Client-Server Architecture. There are 2 types of Enterprise applications.

1. Web Applications
2. Distributed Applications

Q: What are the differences between Web Applications and Distributed Applications?

Ans: 1. Web Application is the server side application, it will be designed without disturbing its application logic over multiple number of JVM's.

Distributed application is the server side application it will be designed by disturbing application logic over multiple number of JVM's.

2. In general web applications will be designed by using a set of server side technologies called as Web technologies like CGI, Servlets, JSP's and so on.

Distributed applications will be designed by using a set of technologies called as Distributed technologies like Socket programming, RMZ, EJB's, Webservices and so on.

3. The main purpose of web applications is to generate dynamic response from server machine, but the main purpose of distributed applications is to establish distributed communication between local machine and remote machine in order to access the remote services.

4. In general web applications will provide services for web clients, but distributed applications will provide services for any type of clients.

5. In general web applications will be executed by using both web servers and application servers, but distributed applications will be executed by using only application servers.

1. Web Applications:

Web Application is a collection of web components, it will be executed by using web containers like Servlet Container to execute servlets, Jsp Container to execute JSP's and so on.

2. Distributed Applications:

Distributed Application is a collection of distributed components, it will be executed by using the distributed containers like EJB's Container to execute EJB's.

At the beginning stages of the computer we have Client-Server architecture, where the purpose of server is to hold up some resources and to share that resources to all the clients as per the client request.

In the above context, when we send a request from client to server for a particular resource then server will identify requested resource, pick up the content and send that content as response to client without performing any particular action at server. In this case the response which was generated by server is called as static response.

As per the application requirements we need to generate dynamic response from server, for this we need to execute the application at server called as Web Application. To design

web applications at server side we need a set of server side technologies called as **Web Technologies.**

To design web applications we will use web technologies like CGI, Servlets, Jsp's, Perl, PHP and so on. Therefore, the main purpose of servlets and jsp's is to design web applications at server in order to generate dynamic response from server.

Q: To design web applications at server we have already CGI Technology then what is the requirement to go for servlets?

Ans: CGI is the server side technology designed on the basis of C technology and Scripting. C technology is the process based technology, it will make CGI technology as process based technology.

If we deploy any CGI application at server side then container will create a separate process for each and every request. If we increase number of requests automatically CGI container will generate number of processes at server.

Basically process is a heavy weight component, to handle single process system has to consume a lot of system memory and execution time.

Due to the above reason to handle the multiple number of processes at a time server machine may get burden, due to this reason server may generate delayed responses to the client request, it will reduce the performance of server side application.

In the above context, to increase the performance of the server side application we have to use an alternative server side technology i.e. Servlets.

P1

Client

CGI
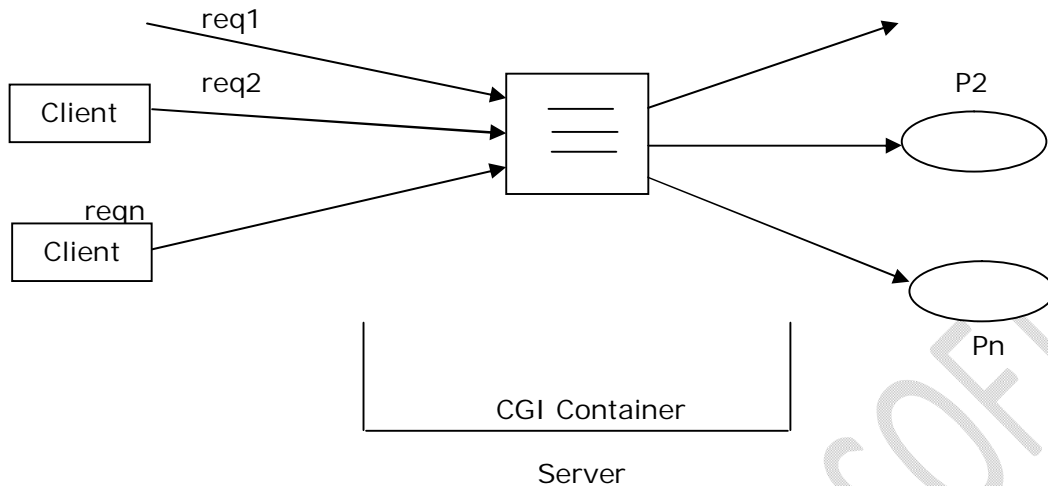
   Servlet is the server side technology designed on the basis of java technology. Java technology is a Thread based technology, it will make servlets as Thread based technologies.

If we deploy any servlet application at server then for every client request servlet container will generate a separate thread on the respective servlet object.

In the above context, if we increase number of requests even container will create number of threads instead of processes.

   When compared to process thread is light weight, to handle multiple number of requests i.e. thread server machine may not get burden, it may provide quick responses for the client request, it may increase the performance of server side application.
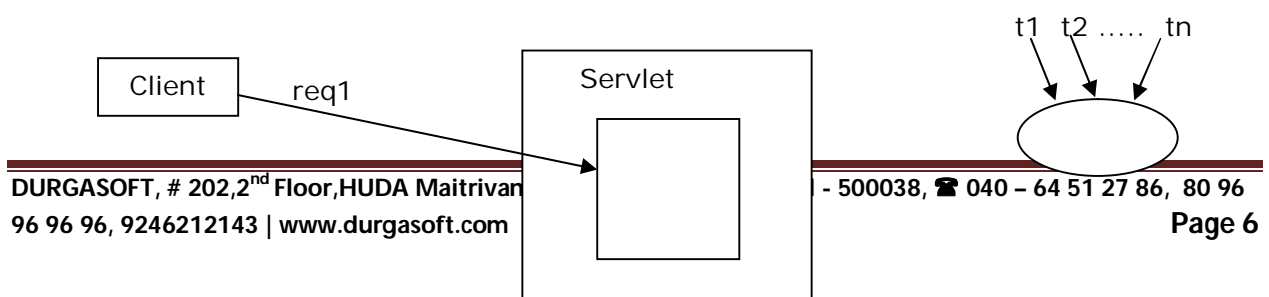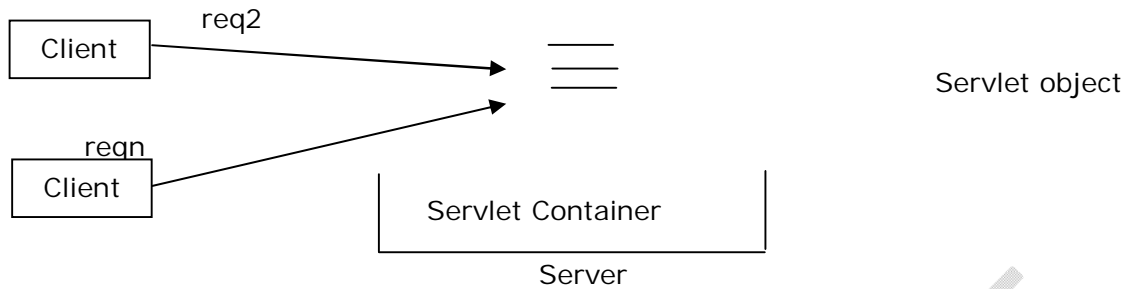
Client ── req2 ──→ ▭ Servlet object

Client ── reqn ──→

Servlet Container

Server

Q: What are the differences between servlets and jsps?

Ans: 1. If we want to design web applications by using Servlets we must require very good java stuff.

   To design web applications by using Jsp technology it is not required to have java knowledge, it is sufficient to have presentation skills.

Note: The main intention to introduce Jsp technology is to reduce java code as much as possible in web applications.

2. In web applications, we will prefer to use Servlets to pick up the request and process the request.

   But we will prefer to use Jsp's to generate dynamic response to the client with very good look and feel.

Note: In web applications, we will utilize Servlets to provide controller logic, integration logic, validation logic, implementing business logic and so on. But we will prefer to use Jsp technology only to provide presentation part.

3. In case of Servlets, we are unable to separate both presentation logic and business logic, but in case of Jsp's we are able to separate presentation logic and business logic because in

Jsp pages we are able to use html tags to prepare presentation logic and we are able to use Jsp tags to prepare business logic.

4. If we perform any modifications on the existed Servlets then we have to perform recompilation and reloading on to the server explicitly.

If we perform any modifications on the existed Jsp's then it is not required to perform recompilation and reloading because Jsp pages are auto compiled and auto loaded.

5. If we want to design any web application on the basis of MVC architecture then we have to use a servlet as controller and a set of Jsp pages as view part.
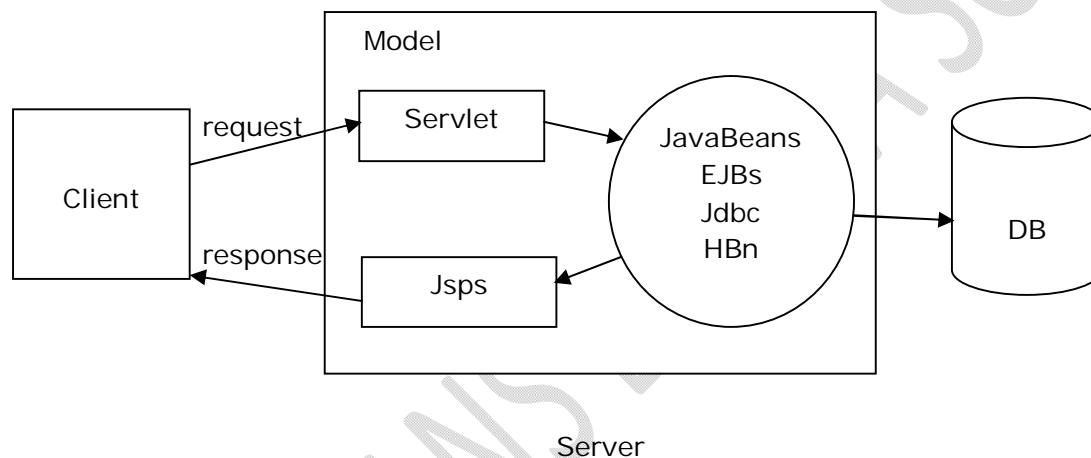
Note 1: Struts is a web application framework designed on the basis of MVC architecture, where we have to use ActionServlet as controller and we have to use a set of Jsp pages as view part.

Note 2: JSF(Java Server Faces) is a web application framework, where we have to use FacesServlet as a controller and a set of Jsp pages as view part.

Enterprise:

Enterprise is a business organization, a group of organizations running under a single label.

Enterprise Application:

It is a software application, which will be designed for a particular enterprise in order to simplify the internal business processing.

To design enterprise applications we have to use the following 3 layers.

1. User Interface Layer
2. Business Processing Layer
3. Data Storage and Access Layer

1. User Interface Layer:

User Interface Layer is the top most layer in enterprise application, it will provide starting point for the customers to interact with enterprise application.

The main purpose of User Interface Layer in enterprise applications is

1. To improve look and feel for the enterprise applications.
2. To accept user details in order to execute server side application.
3. To provide very good environment for client side validations with Javascript functions.
4. To specify different types of requests at client browser like get, post, head and so on.

To prepare User Interface Layer in enterprise application we will use a separate logic is called as **Presentation Logic.**

To prepare presentation logic in enterprise applications we have to use technologies like AWT, SWING, HTML, JSP, Velocity and so on.

2. Business Processing Layer:

Business Processing Layer is the heart of the enterprise application, it will provide very good environment to define and execute all the business rules and regulations which are required by the client.

To prepare Business Processing Layer in enterprise application we will use a separate logic is called as **Business Logic.**

In enterprise application development,to prepare Business logic we have to use technologies like Servlets, Jsp's, JavaBeans, EJBs and so on.

3. Data Storage and Access Layer:

This layer is the bottom most layer in enterprise applications.

The main purpose of this layer is to provide data persistency in enterprise application.

Data Storage and Access Layer will provide very good environment to provide the basic database operations(CRUD) as per the enterprise application requirement.

To prepare Data Storage and Access Layer we will use a separate logic called as **Persistence Logic.**

In enterprise application development,to prepare persistence logic we have to use technologies like Jdbc, EJB-Entity Beans Hibernate, JPA and so on.

To design enterprise application we need to define the degree of enterprise application, for this we have to use system architectures.

1-Tier Architecture

2-Tier Architecture
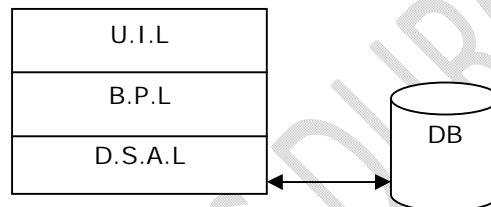
----

n-Tier Architecture

1-Tier Architecture:

To design enterprise application if we use 1-tier architecture then we have to provide User Interface Layer, Business Processing Layer and Data Storage and Access Layer within a single machine.

1-Tier architecture is highly recommended for standalone applications.

In 1-Tier architecture, a single machine we have to use to accommodate the complete application so that a single machine resources may not be sufficient to execute applications, it may reduce the performance of enterprise application.

1-Tier architecture will not provide any environment to handle multiple number requests, it is able to provide less sharability.

1-Tier Architecture

2-Tier Architecture:

To design any enterprise application it is minimum to use 2-Tier architecture. The best example for 2-Tier architecture is Client-Server architecture.

2-Tier architecture will allow to design and execute the application in 2 layers of machines.

In case of 2-Tier architecture, tier-1 will manage User Interface Layer and tier-2 will manage Business Processing Layer and Data Storage and Access Layers.

2-Tier architecture will provide loosely coupled design when compared with 1-Tier architecture.

2-Tier architecture will provide very good environment to handle multiple number of requests and it is able to improve sharability.

Client-Tier/Tier-1   Server-Tier/Tier-2
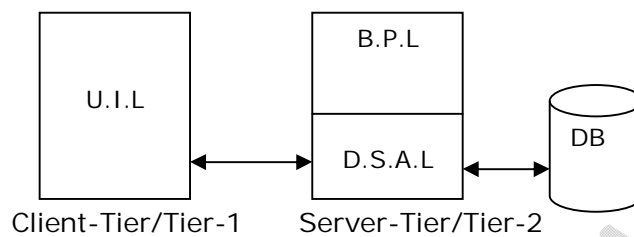
2-Tier Architecture

Note: As part of the enterprise application development always it is suggestible to use Tiered architectures because it will improve sharability, able to provide more loosely coupled design and so on, but we should not increase number of tiers in enterprise applications without having the requirement otherwise maintenance problems will be increased.