Java and Web Development

Starring:
Majrul Ansari
and no one else

- 12+ years of IT experience
- 8+ years of Corporate Training experience
- Struts, JSF, Hibernate, Spring, EJB, WebServices, HTML5/CSS3, jQuery, Android, iOS, … are some of the technologies I am comfortable with
- *Wish you all a very happy learning!*

My Small Profile

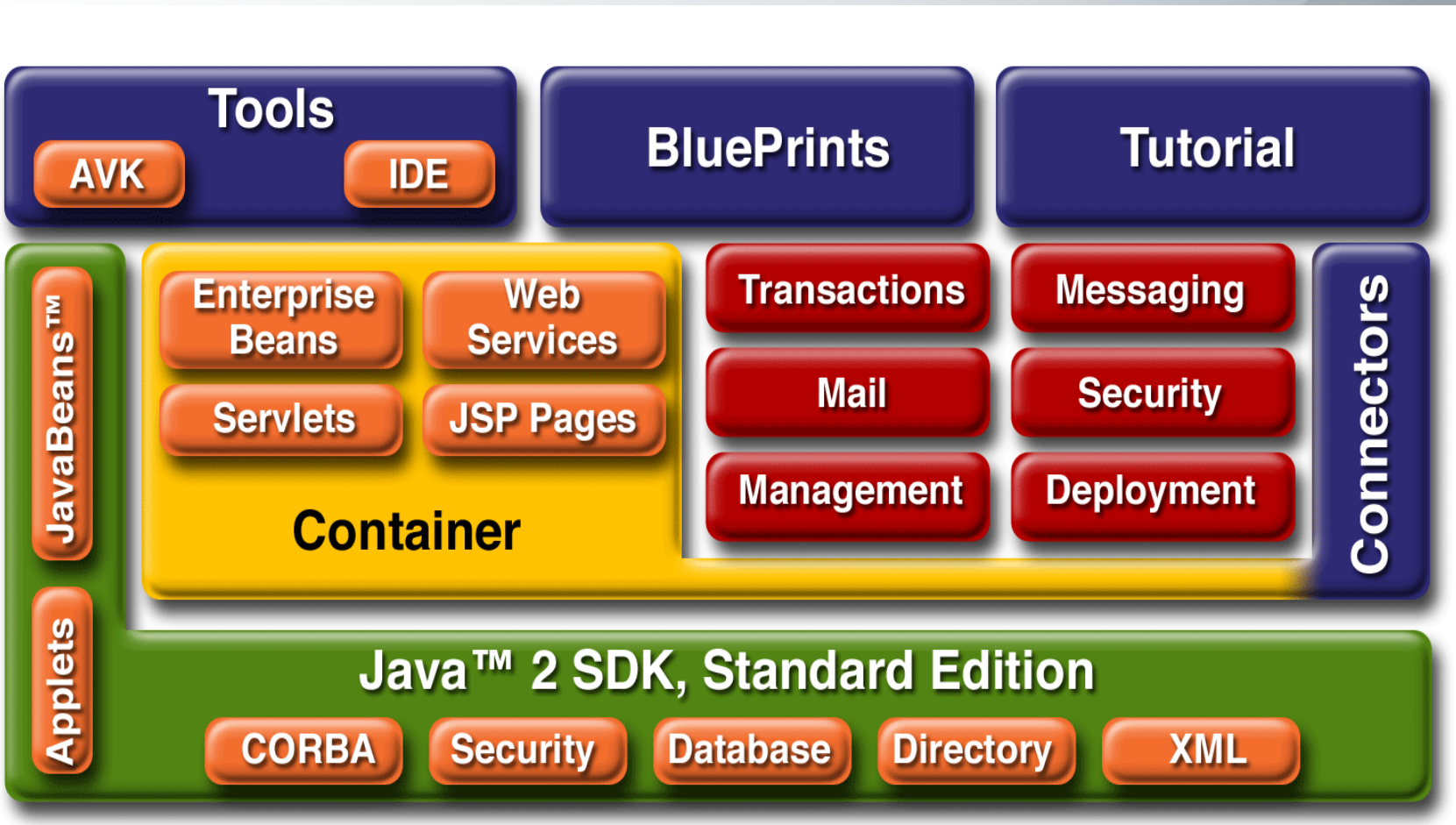
Introduction

- Let's discuss about the J2EE/JEE specification
- Role of Servlets/JSP
- Role of EJB
- …

Java and its flavors

- J2SE (Standard)
- J2EE (Enterprise)
- J2ME (Micro)

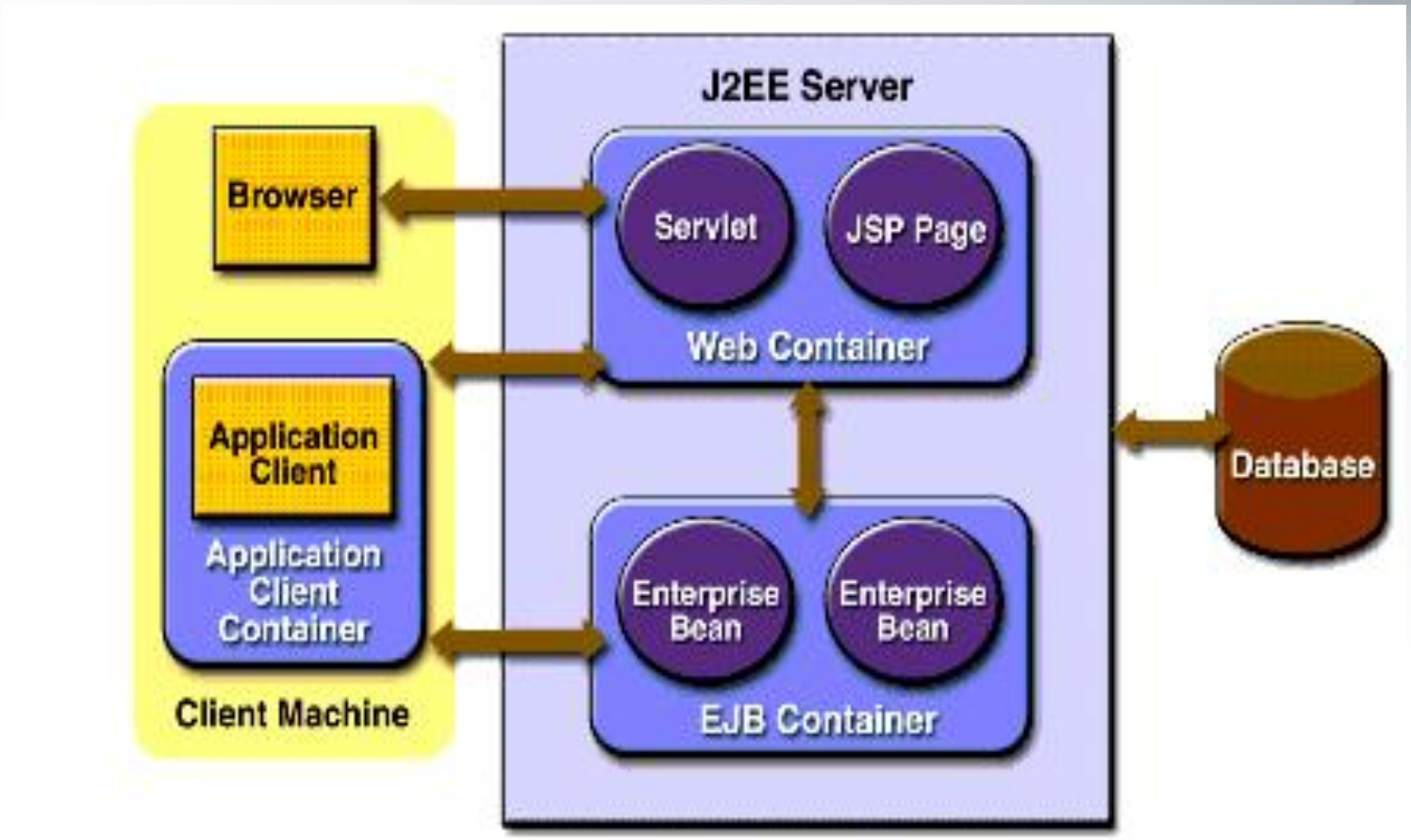
J2EE Architecture overview



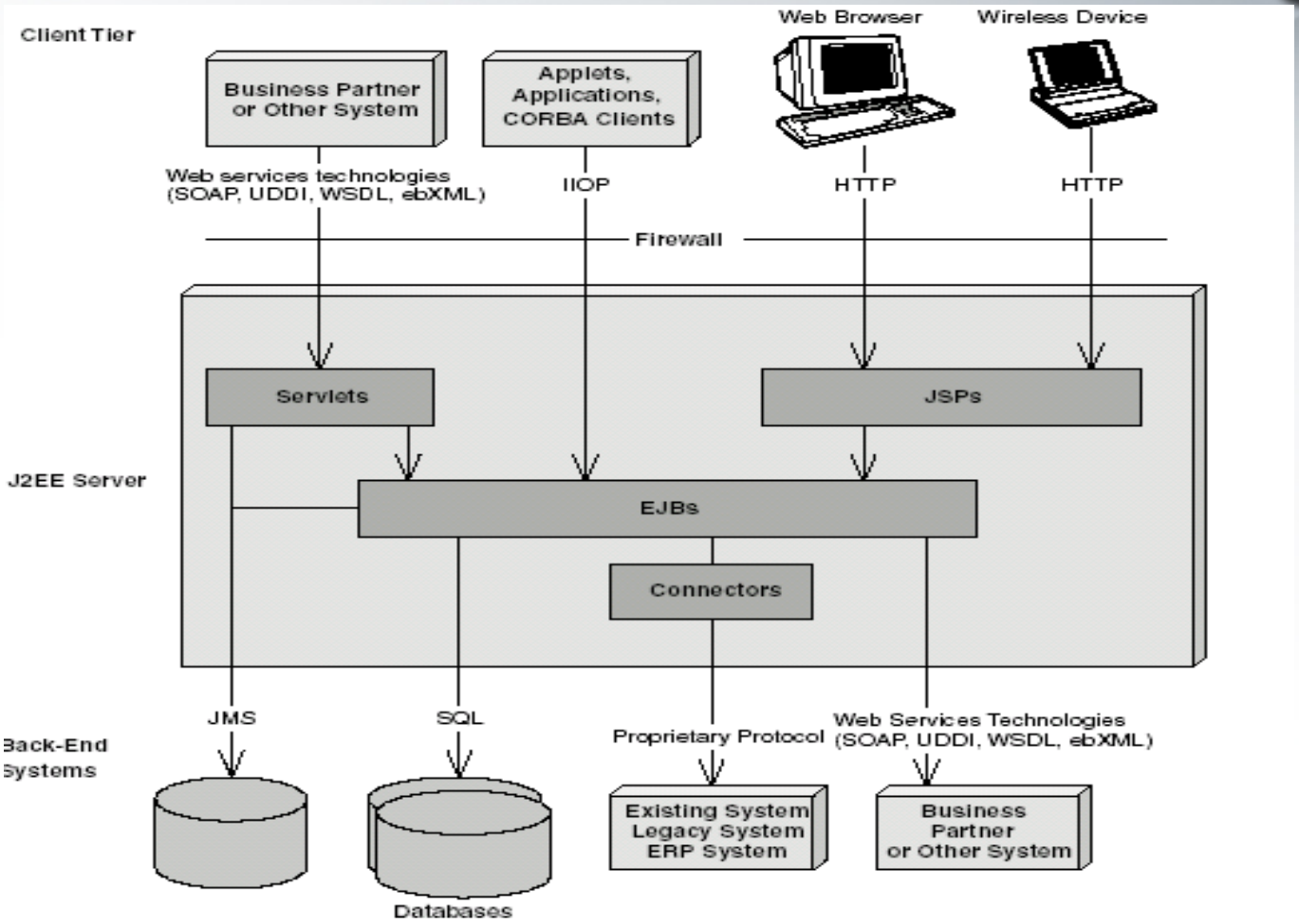
Introduction to J2EE

- J2EE addresses issues persisting to the usage of Java on the server side
- Provides a standard specification to develop portable server-side applications
- Access to low-level services like transactions, security, pooling, concurrency, and other external resources in a standard way

J2EE Overview continued

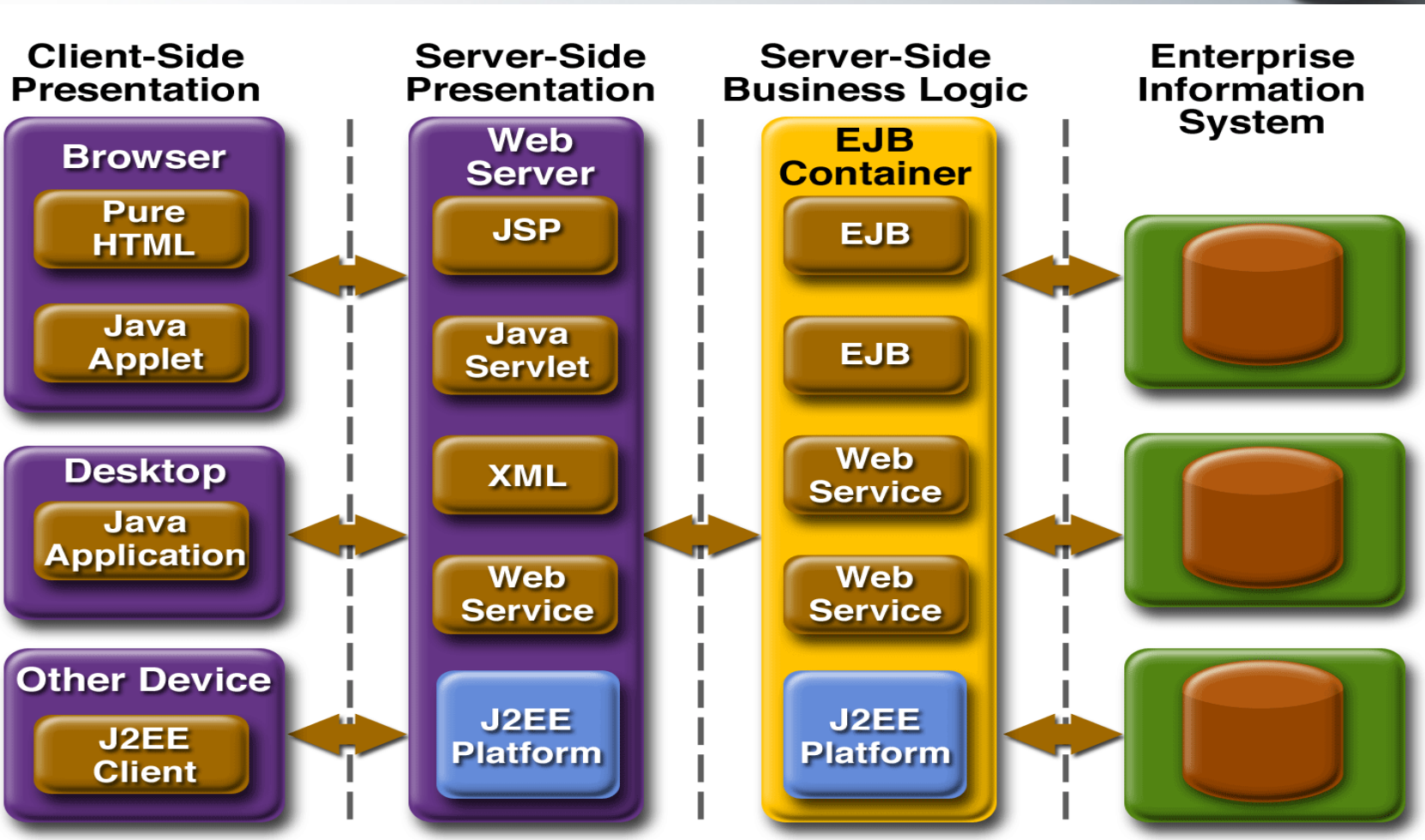


J2EE Overview continued



J2EE Application Layers

Application Layer continued



J2EE Specification

- Servlet, JSP
- EJB
- JNDI
- JTA
- JMS
- JCA
- JAAS
- JMX
- Java Mail
- RMI-IIOP
- Web Services

Servlet

- Introduction
- Example Servlets
- Handling Request and Response
- Cookies
- Session Management

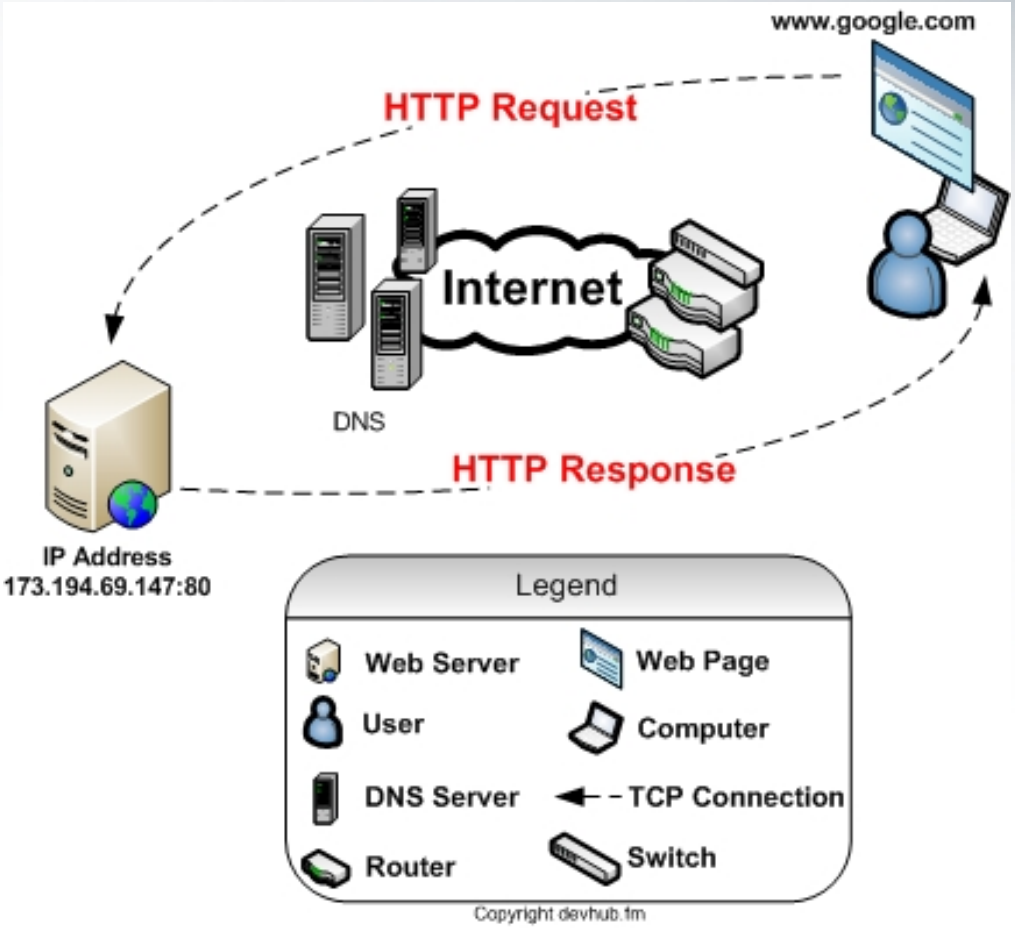
Features of a Web/HTTP Server

- Serve files stored on the server to the client using a browser
- Logically, a Web server is similar to an FTP server except the difference of the protocol used
- HTTP protocol is stateless, which means everytime we send a request, there is no trace of the previous request on the server
- To download a file from the server, we need to mention the file path in the URL itself, for ex:
  - **http://www.whatisallthis.com/whydoineedthis.pdf**

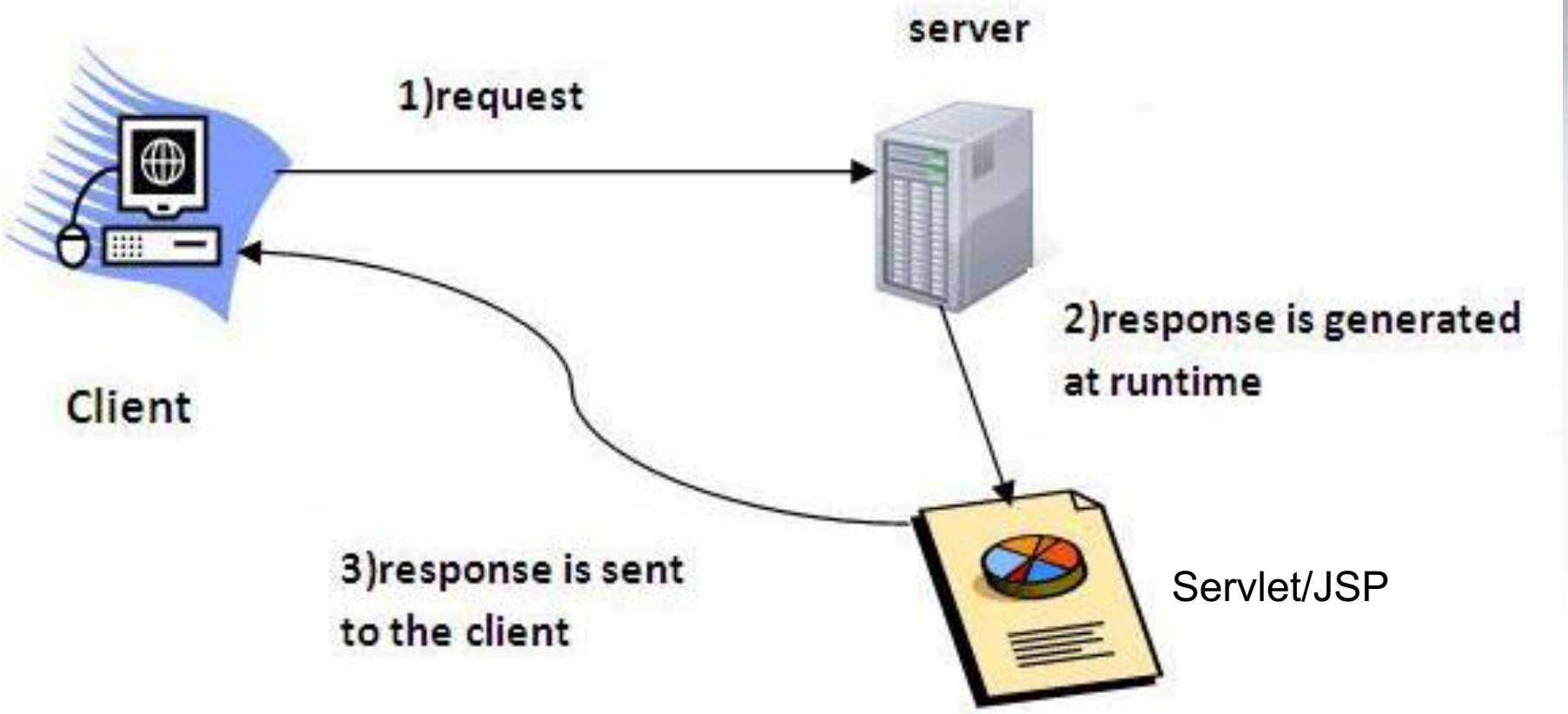
Cont'd…

- The most important feature of a Web Server is, it allows us to extend it's capability by writing server side extensions
- These extensions are written using the API provided by the Web Server
- Servlets give us the same power of Java on the server side whereby allowing us to write code independent of any web server
- This is achieved using a separate software called as Web Container/Servlet Container
  - Tomcat/JRUN, …

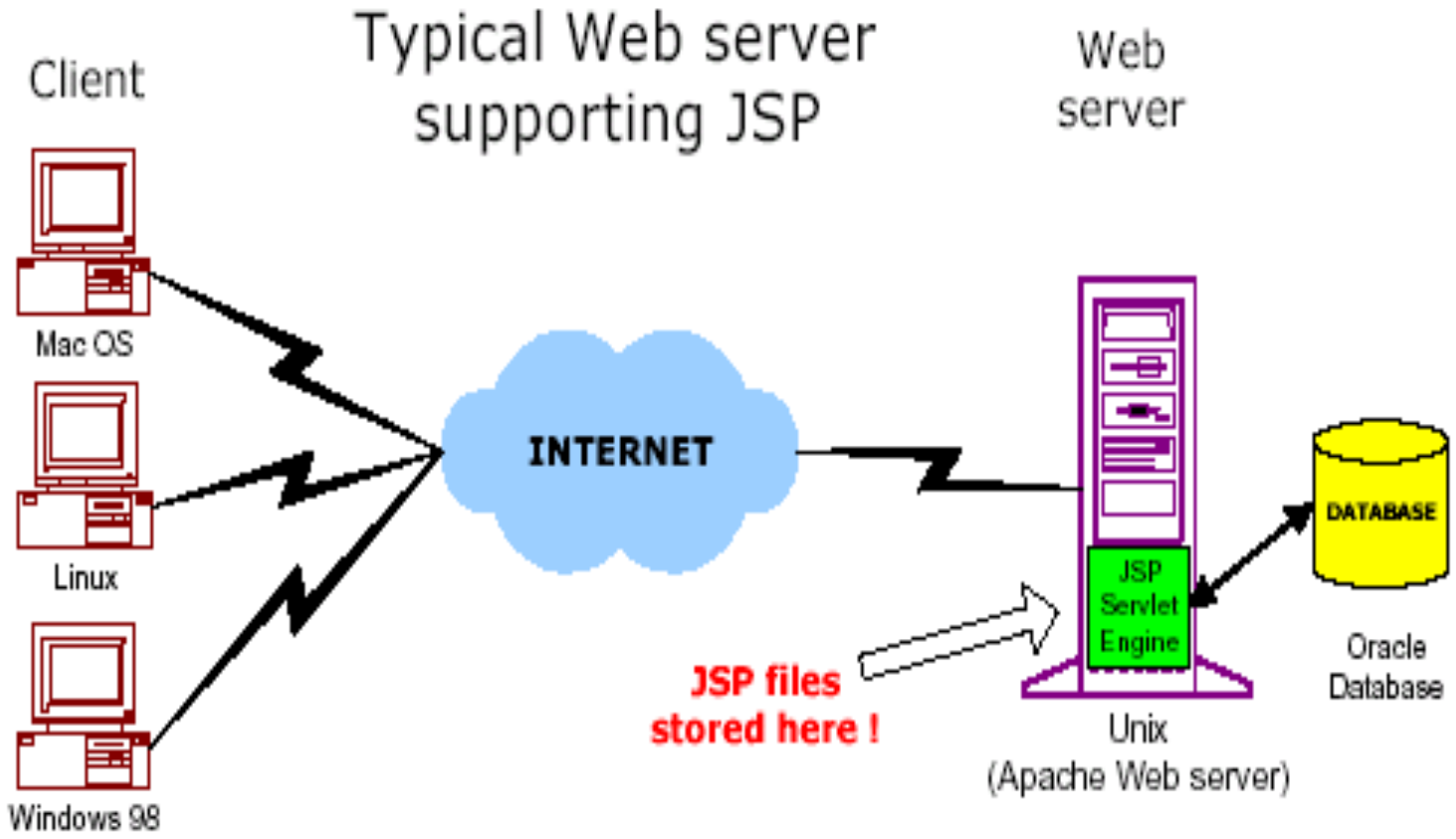
Role of a Web Server



In case of Servlet/JSP



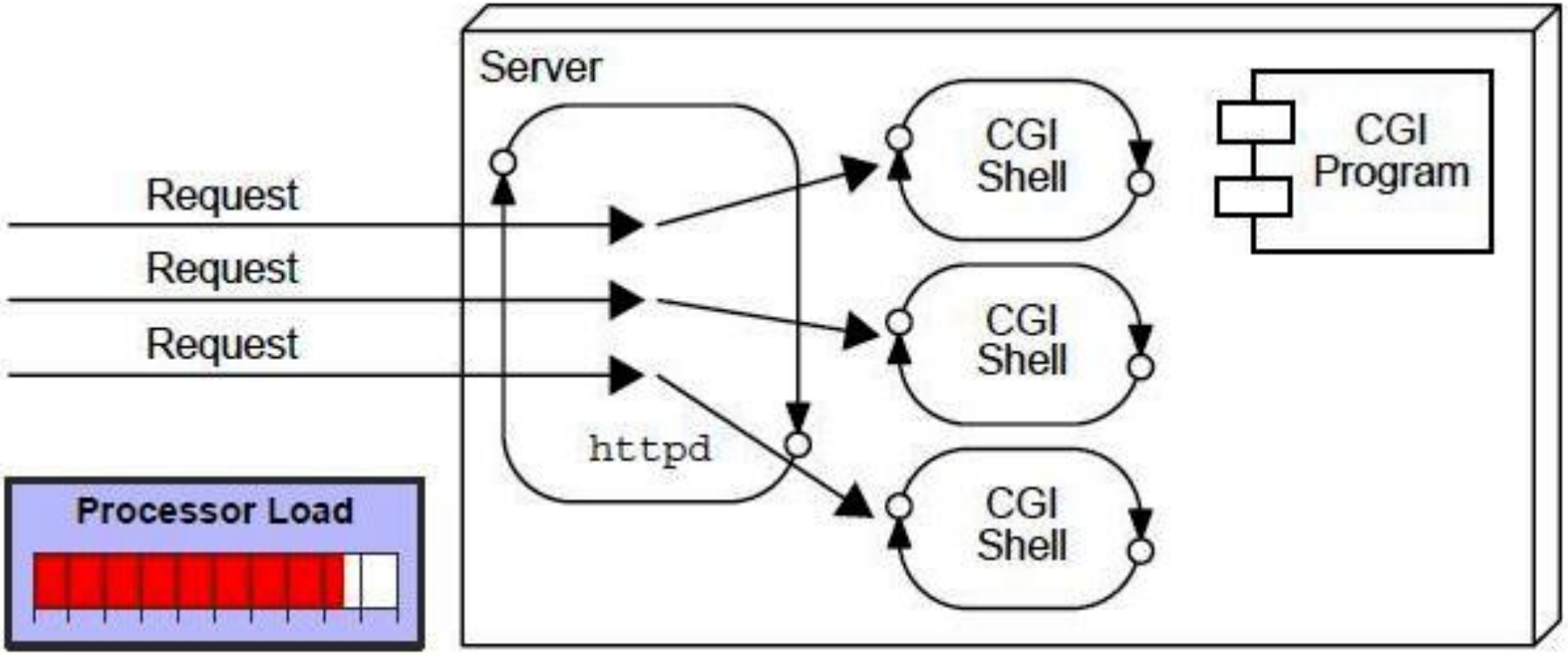
Web Server and Servlets / JSP



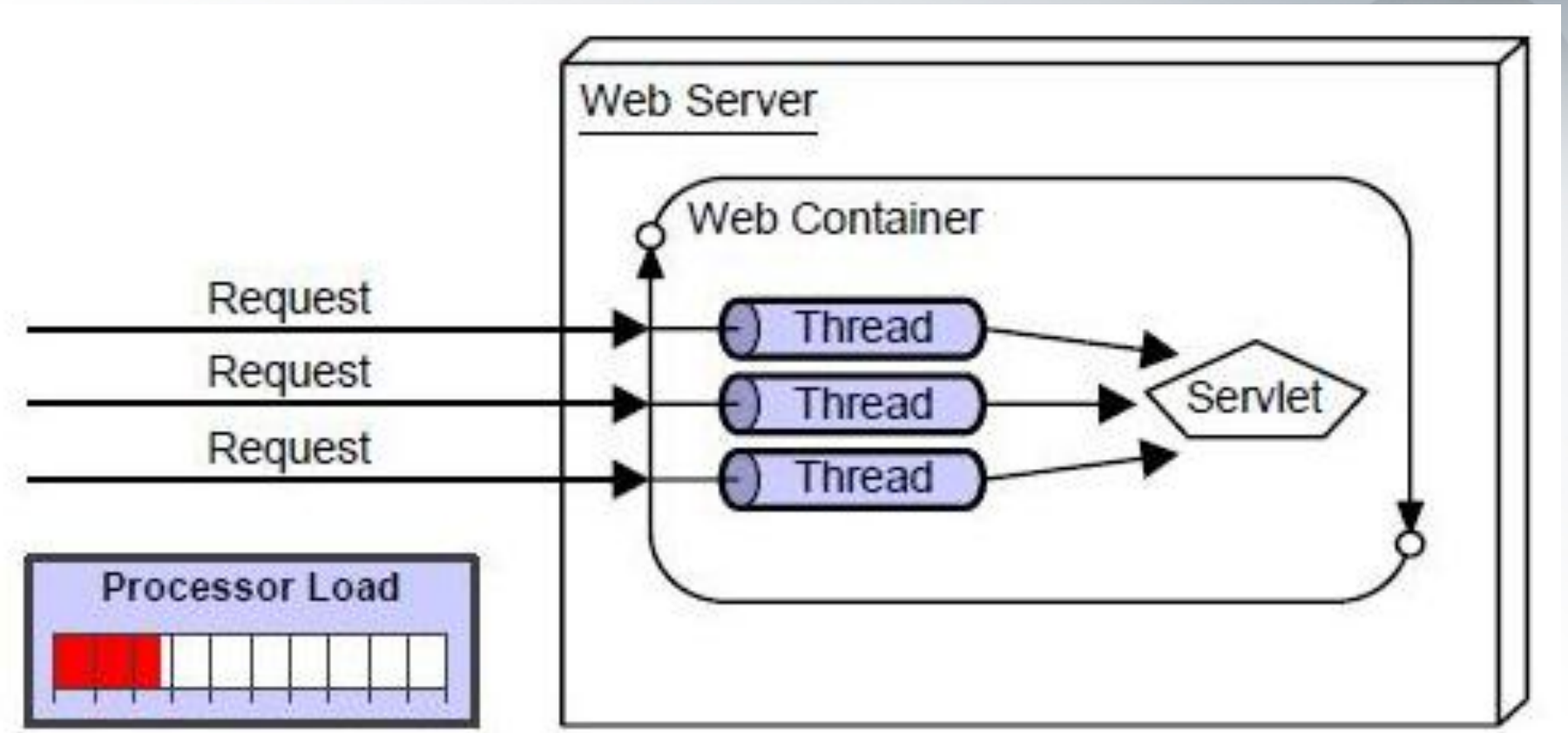
Servlet cont'd…

- Extensions to Web Servers
- Responsibilities of a Servlet
  - Reading explicit data sent by client (form data)
  - Reading implicit data sent by client (request headers)
  - Generate the results
  - Send the explicit data back to the client (HTML)
  - Send the implicit data to the client (status codes and response headers)

Problem with CGI

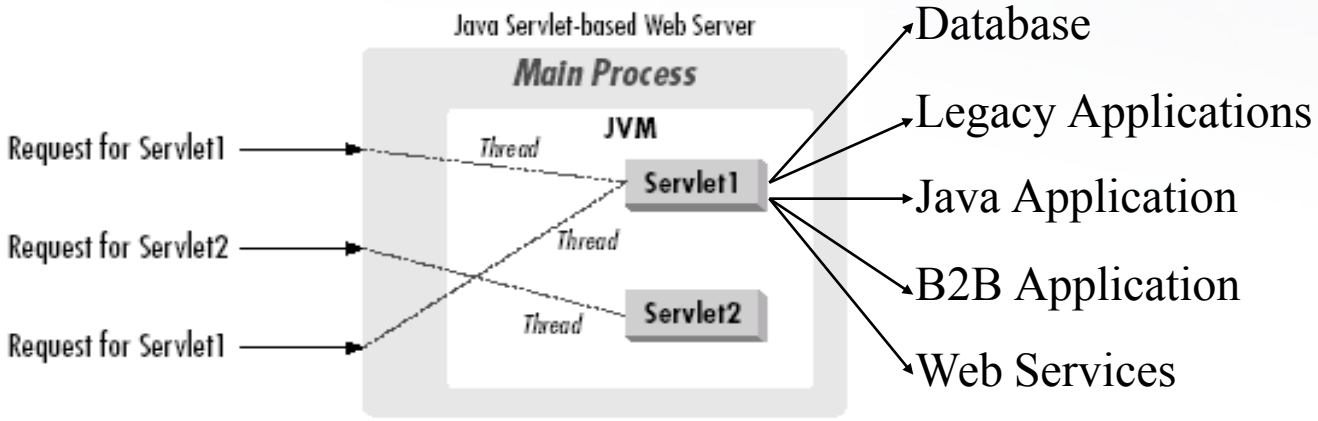


Servlet/JSP


Web Server
Web Container
Request
Request
Request
Thread
Thread
Thread
Servlet
Processor Load

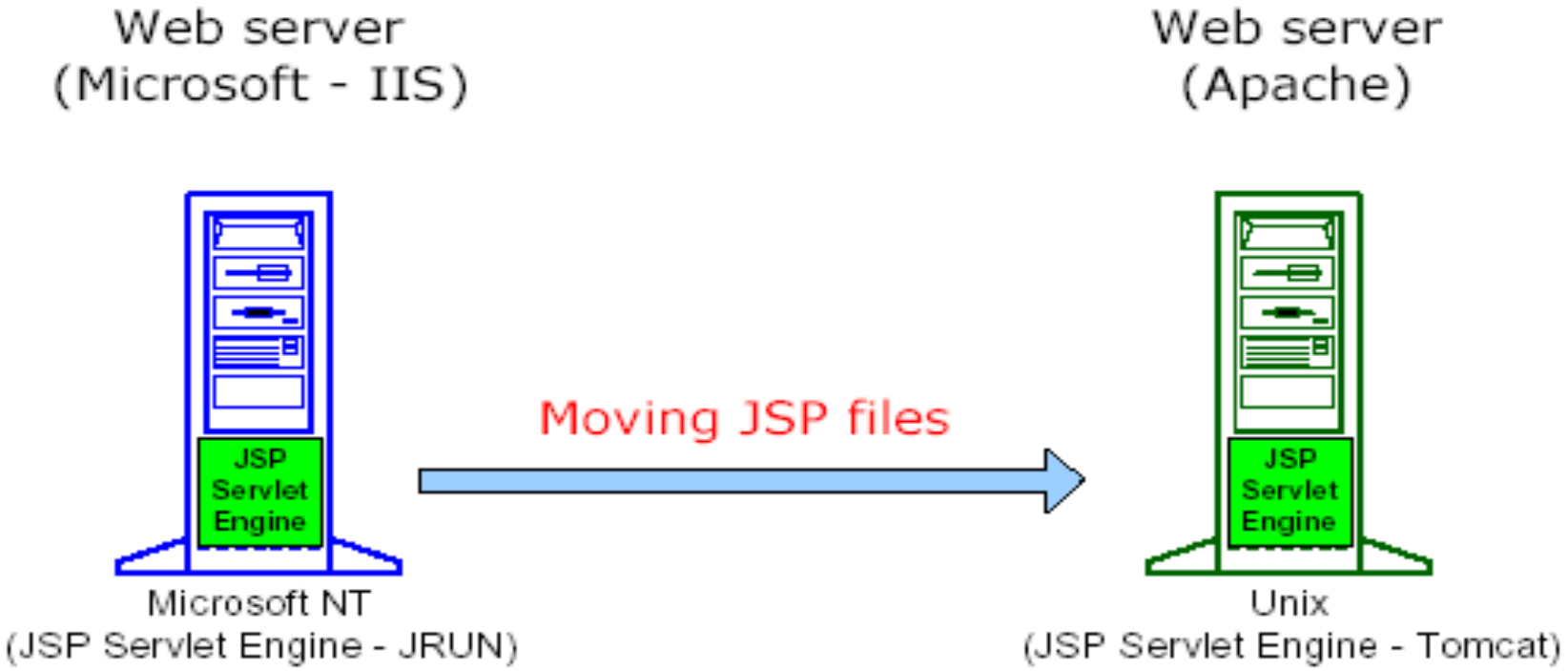
Why Servlets and JSP ?

- Performance, Powerful, Portable
  - Threads instead of OS processes, one servlet copy, persistence
  - Huge Java API's can be exploited
  - Runs on virtually all OS and servers



Servlets and JSP are portable

Moving JSP file from one platform to another.



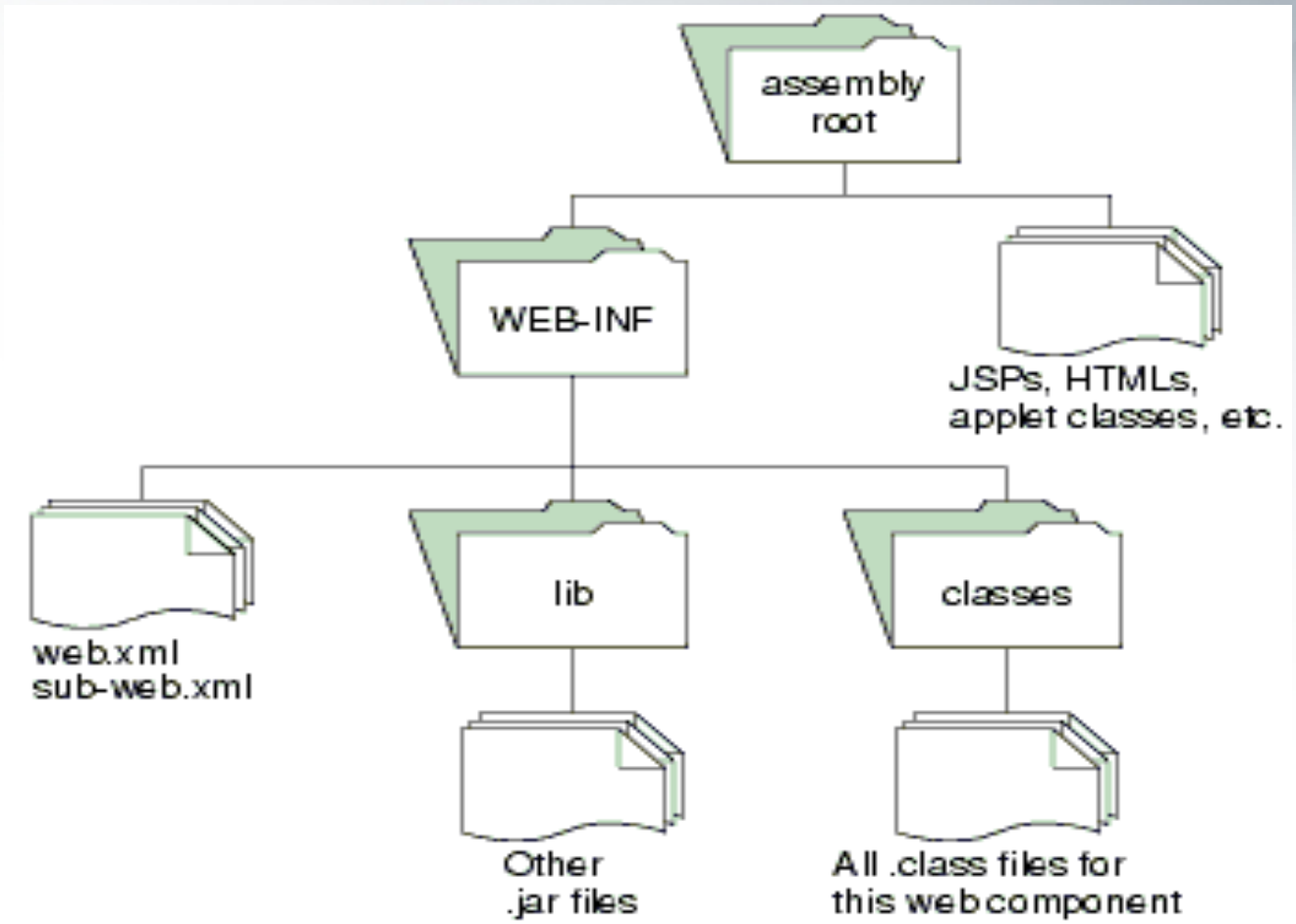
Why build pages dynamically ?

- The Web page is based on data submitted by the user
  - E.g., results page from search engines and order-confirmation pages at online stores
- The Web page is derived from data that changes frequently
  - E.g., a weather report or news headline page
- The Web page uses information from databases or other server side resources
  - E.g., an e-commerce site could use a servlet to build a Web page that lists the current price and availability of each item that is for sale

Technologies used for developing Web Applications

- HTML is the primary technology used over the Web
- To add dynamism in HTML pages, we use Javascript, DOM and DHTML
- To manage the layout and appearance of Web pages, we use CSS
- To dynamically generate HTML pages, we use Server side technologies like Servlets, JSP, PHP, ASP.NET and others
- To asynchronously communicate with the server, we use AJAX

Structure of a Web Module



Servlet Life Cycle

- init
  - Executed once when the servlet is first loaded. Not called for each request
- service
  - Called in a new thread by server for each request. Dispatches to doGet, doPost, etc…
- doGet, doPost, etc…
  - Handles HTTP GET, POST,etc… requests
- destroy
  - Called when server deletes the servlet instance. Not called after each request

Servlet examples

- Basic Hello World Servlet
- Generating data in HTML format
- Generating data in Excel format
- Reading input from the client
- Reading init parameters from the web.xml file

HTTP Headers

- Reading HTTP request headers from the client
- Using the “Content-Encoding” response header
- Using the “Refresh” response header

Java Server Pages(JSP)

- Alternative to Servlets
- Servlets are cumbersome as it's a pure Java class with HTML embedded in it
- JSP is an HTML or an XML or any other markup file with Java code embedded in it
- When requested, JSP is compiled into a Servlet
- Tag libraries are a big plus
- JSTL, JSF, Struts, EL, etc… add more power

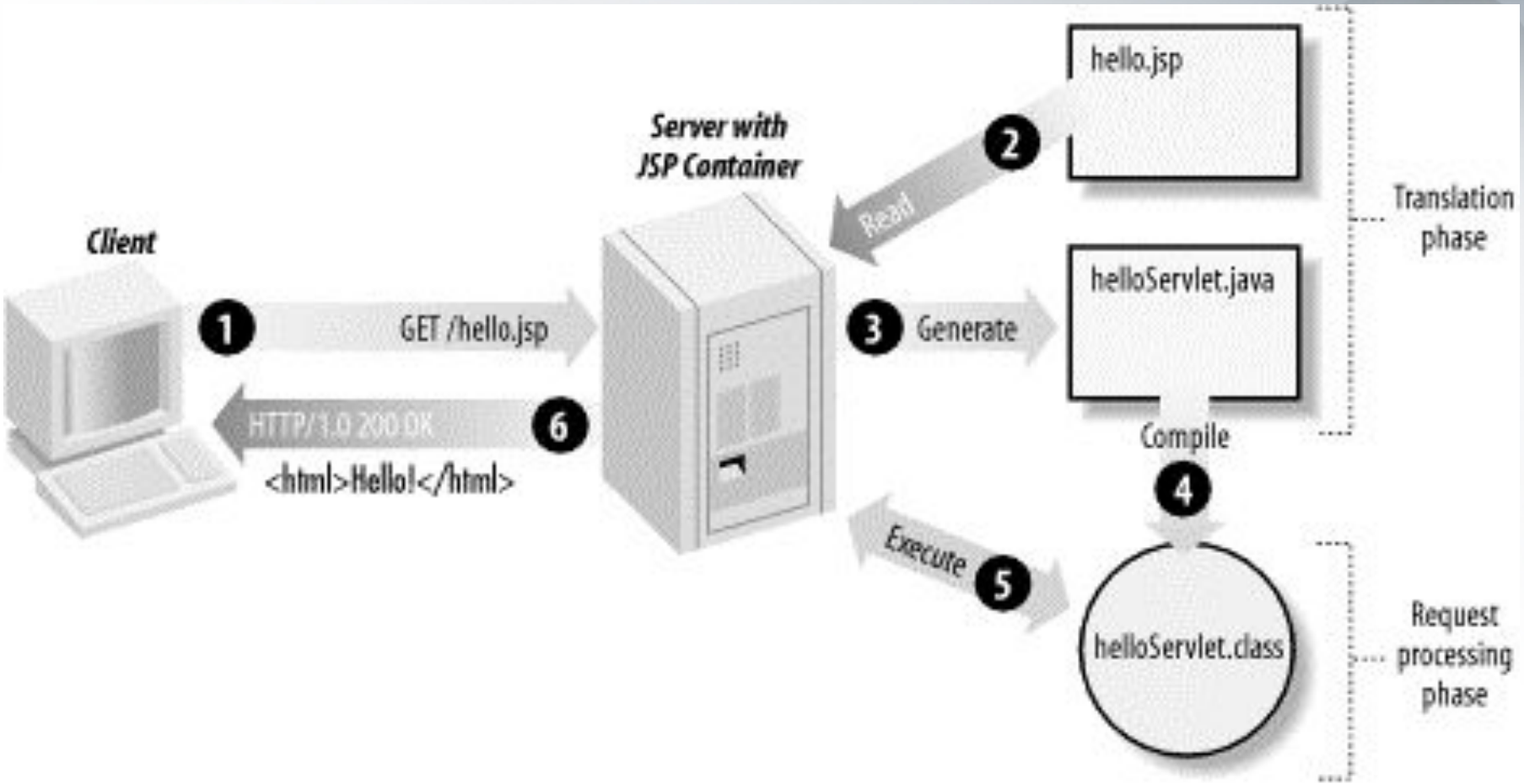
JSP Examples

- Using elements
  - <%= %> expression
  - <% %> script-let
  - <%! %> declarative
  - <%@ %> directive
  - <%-- --%> comments
- Using action elements
  - <jsp:forward />
  - <jsp:include />
  - <jsp:useBean />

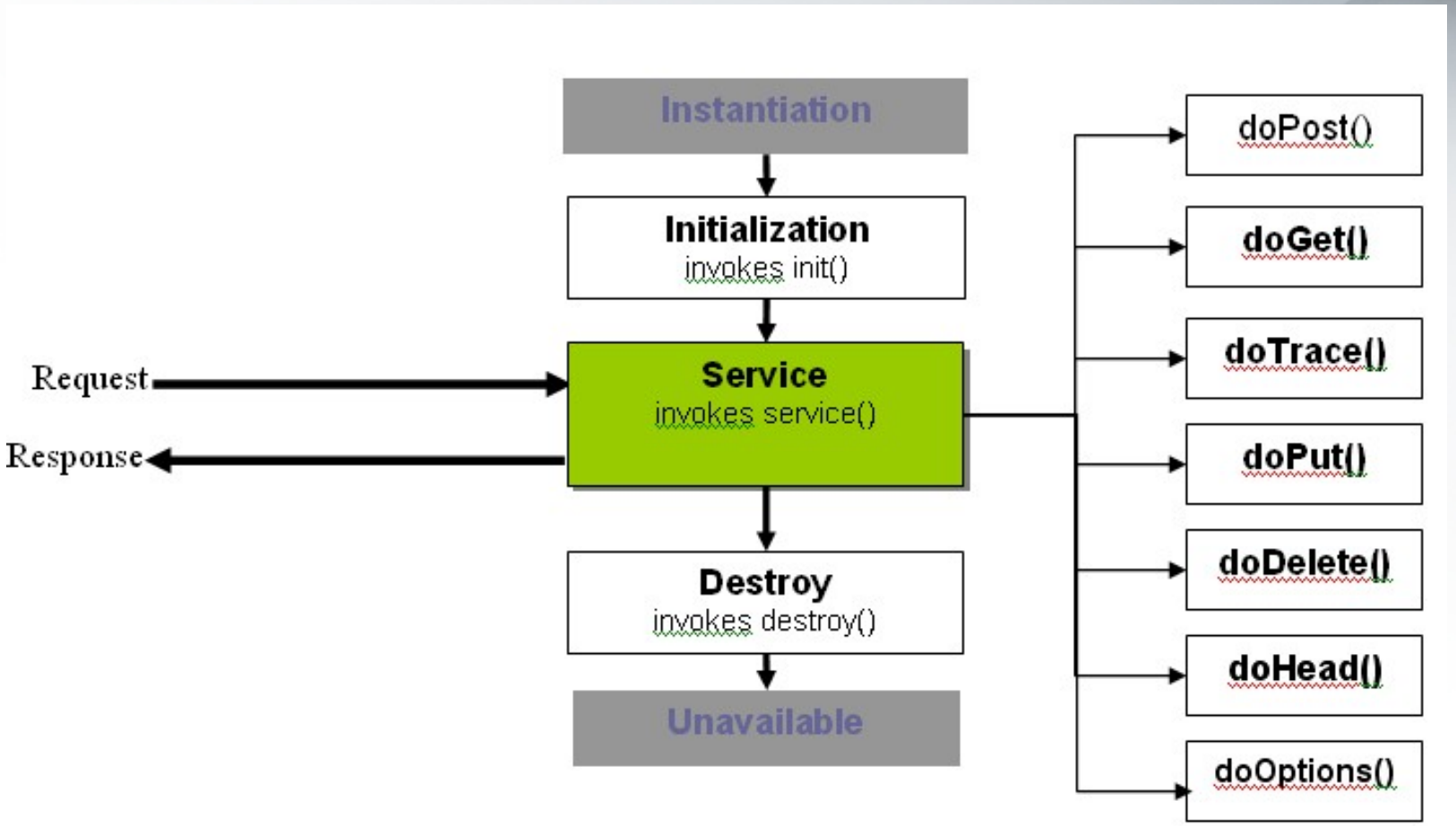
JSP Examples cont'd…

- Session management
  - Cookies
  - Session objects
  - URL rewriting
  - Hidden fields
- Error Handling
- Filters and Listeners
- Custom Tags

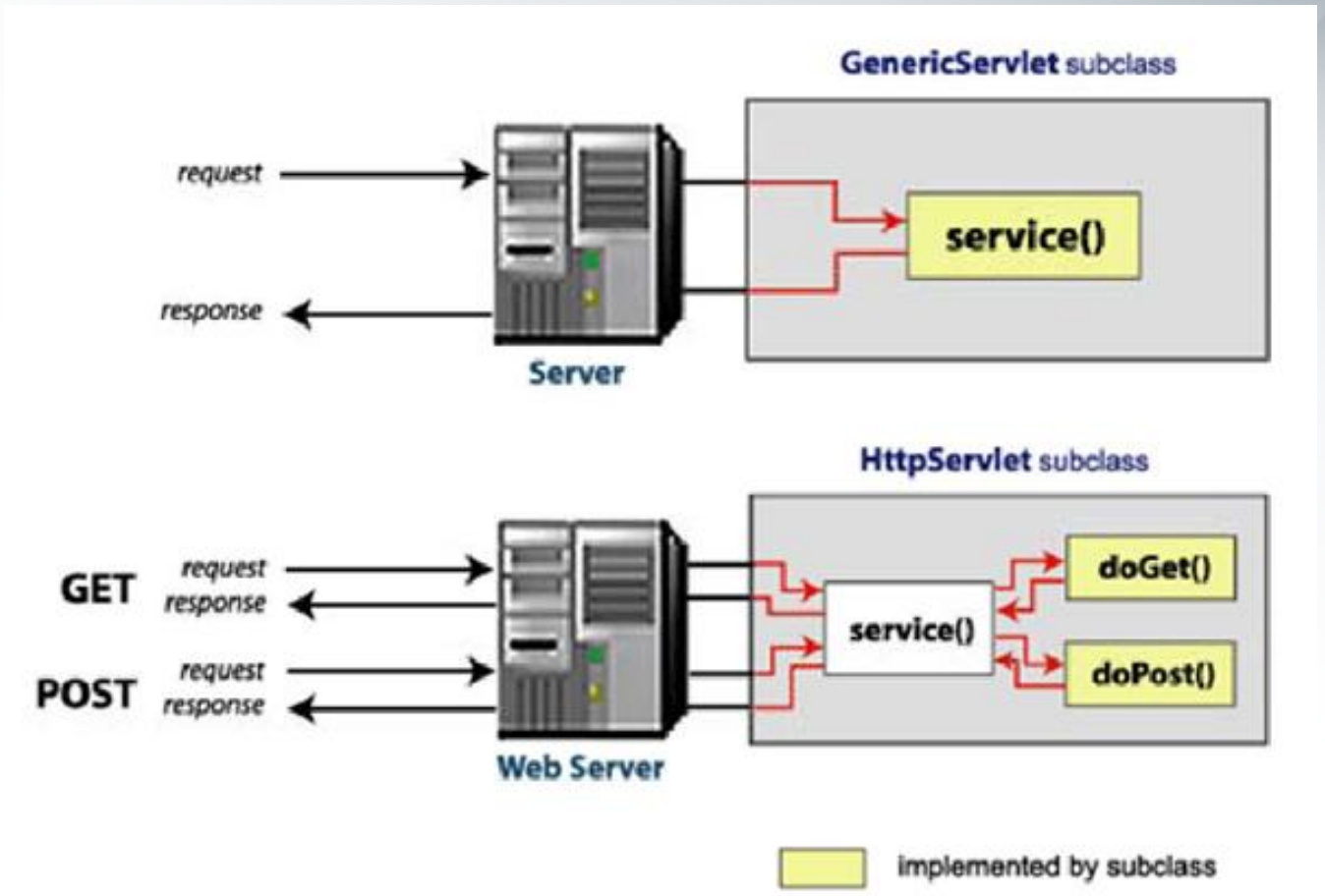
JSP Architecture



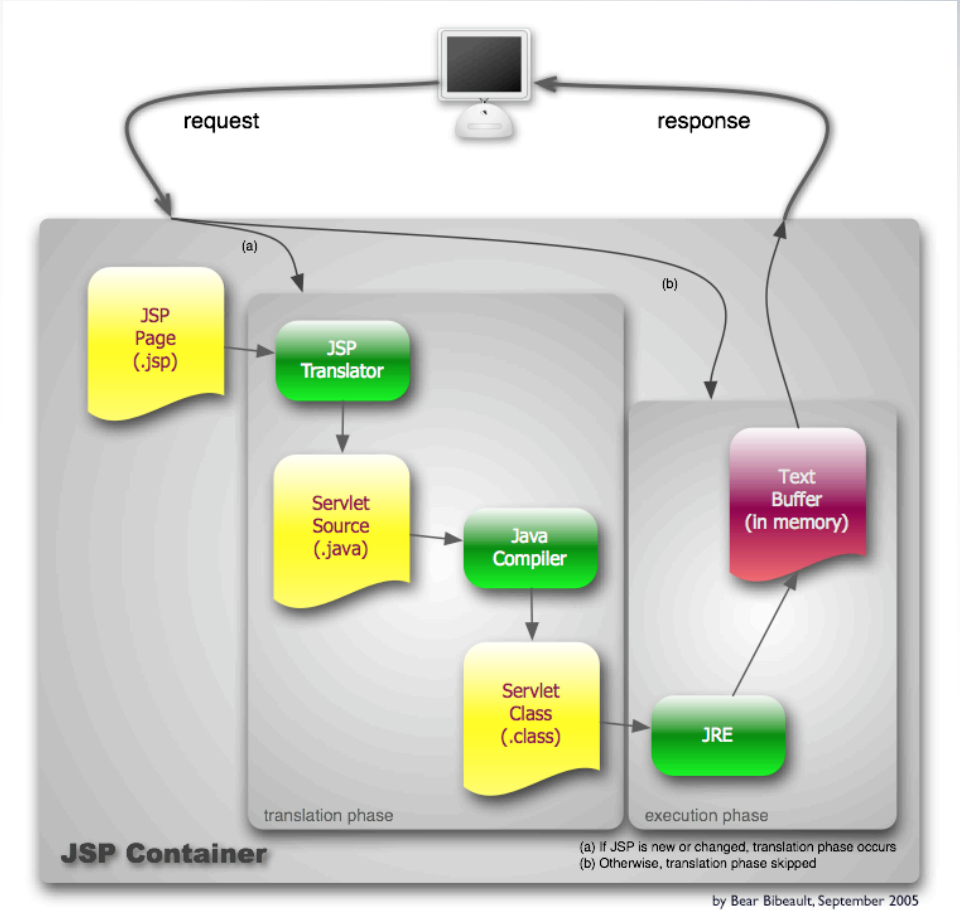
Servlet Lifecycle



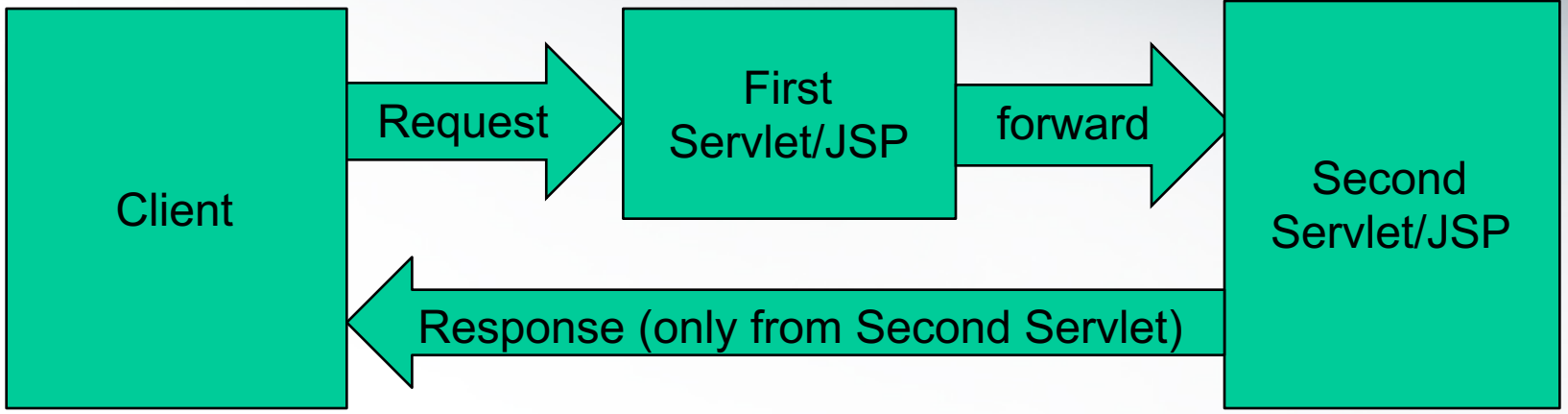
Cont'd..


GenericServlet subclass
request
response
service()
Server
HttpServlet subclass
GET
request
response
POST
request
response
service()
doGet()
doPost()
Web Server
implemented by subclass

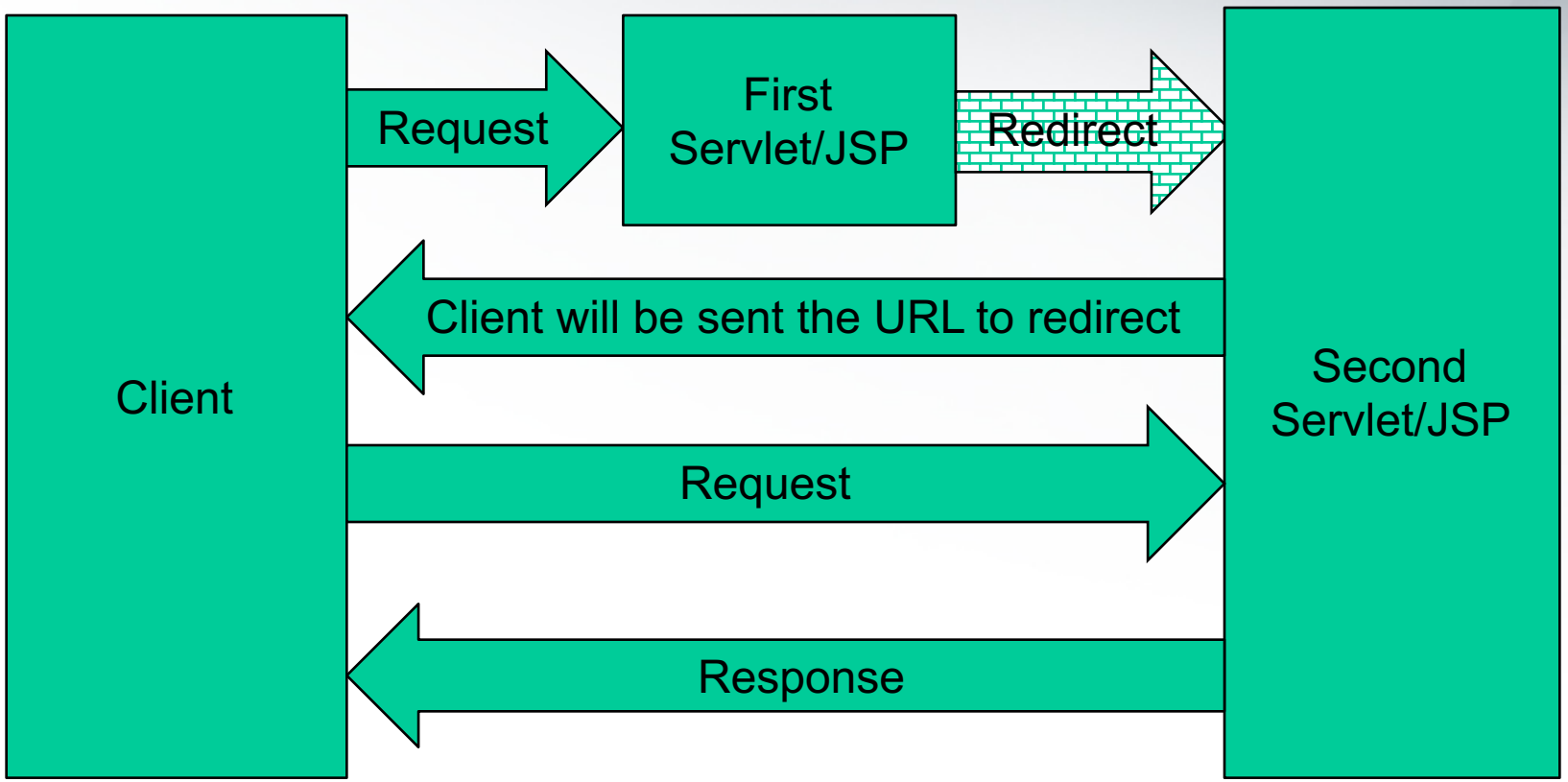
JSP Lifecycle



Forward


Client
Request
First
Servlet/JSP
forward
Second
Servlet/JSP
Response (only from Second Servlet)

Redirect

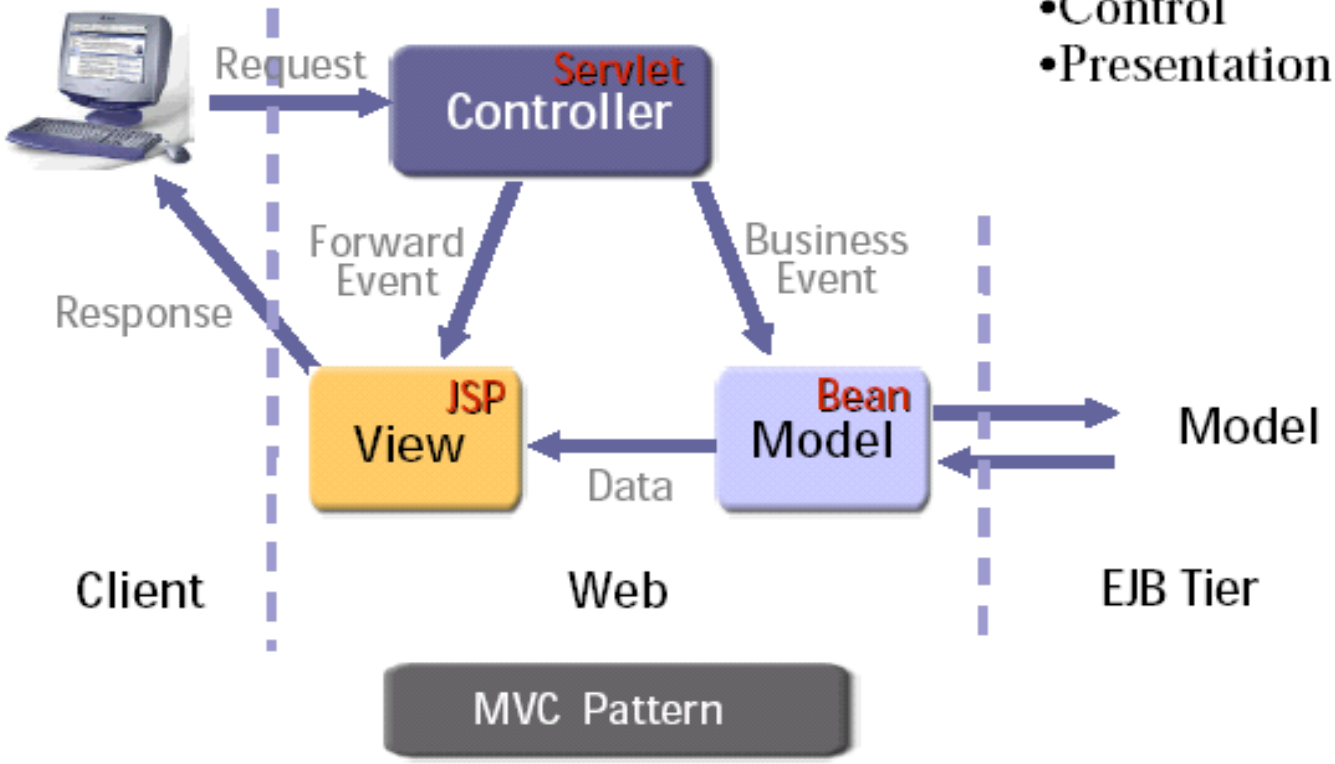

Client
Request
First
Servlet/JSP
Redirect
Second
Servlet/JSP
Client will be sent the URL to redirect
Request
Response

MVC

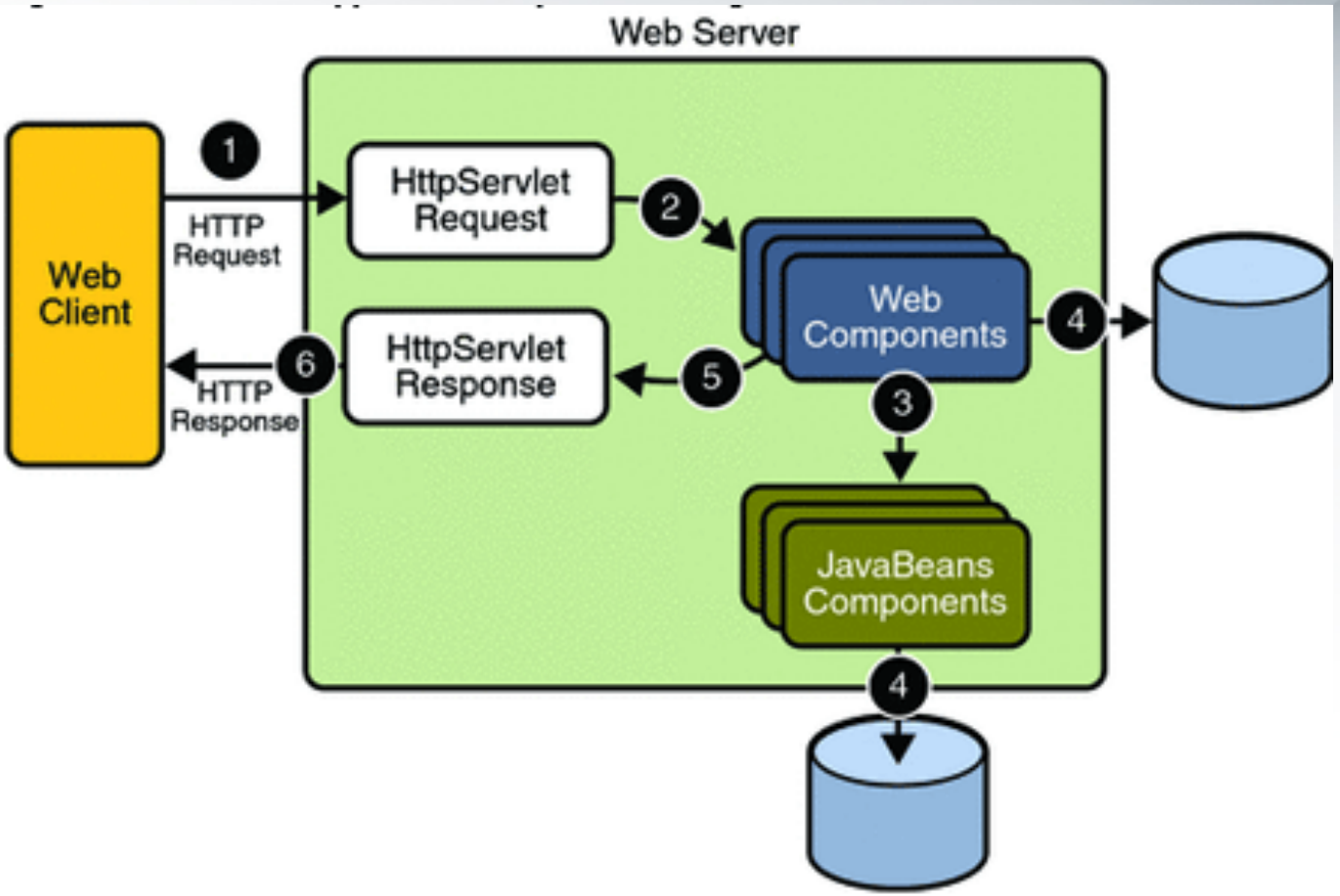
- **Model =** business data and rules, shared by all clients
- **Controller =** defines the application behavior, interprets user actions and maps them into actions performed on the model
- **View =** renders the contents of the model

Servlet, JSP and MVC

- Separate:
  - Model data
  - Control
  - Presentation of view



Cont'd…


Web Server
Web Client
1
HTTP Request
HttpServlet Request
2
Web Components
4
3
HttpServlet Response
5
6
HTTP Response
JavaBeans Components
4

Conclusion