# Introduction to JAVA EE (J2EE)

By: Mahboob Ali

## What is J2EE (or JEE)

- Short for ***J**ava **2** Platform **E**nterprise **E**dition*.

- J2EE is a platform-independent, Java-centric environment from Sun/Oracle for developing, building and deploying Web-based enterprise applications online.

- The J2EE platform consists of a set of services, APIs, and protocols that provide the functionality for developing multi-tiered, Web-based applications.

## Differences between Java EE and Java SE

- Java technology is both a programming language and a platform.
- The Java programming language is a high-level object-oriented language that has a particular syntax and style.
- A Java platform is a particular environment in which Java programming language applications run.
- There are several Java platforms.
- Many developers, even long-time Java programming language developers, do not understand how the different platforms relate to each other.

## Java SE

- When most people think of the Java programming language, they think of the Java SE API.
- Java SE's API provides the core functionality of the Java programming language.
- It defines everything from the basic types and objects of the Java programming language to high-level classes that are used for networking, security, database access, graphical user interface (GUI) development, and XML parsing.
- In addition to the core API, the Java SE platform consists of a virtual machine, development tools, deployment technologies, and other class libraries and toolkits commonly used in Java technology applications.

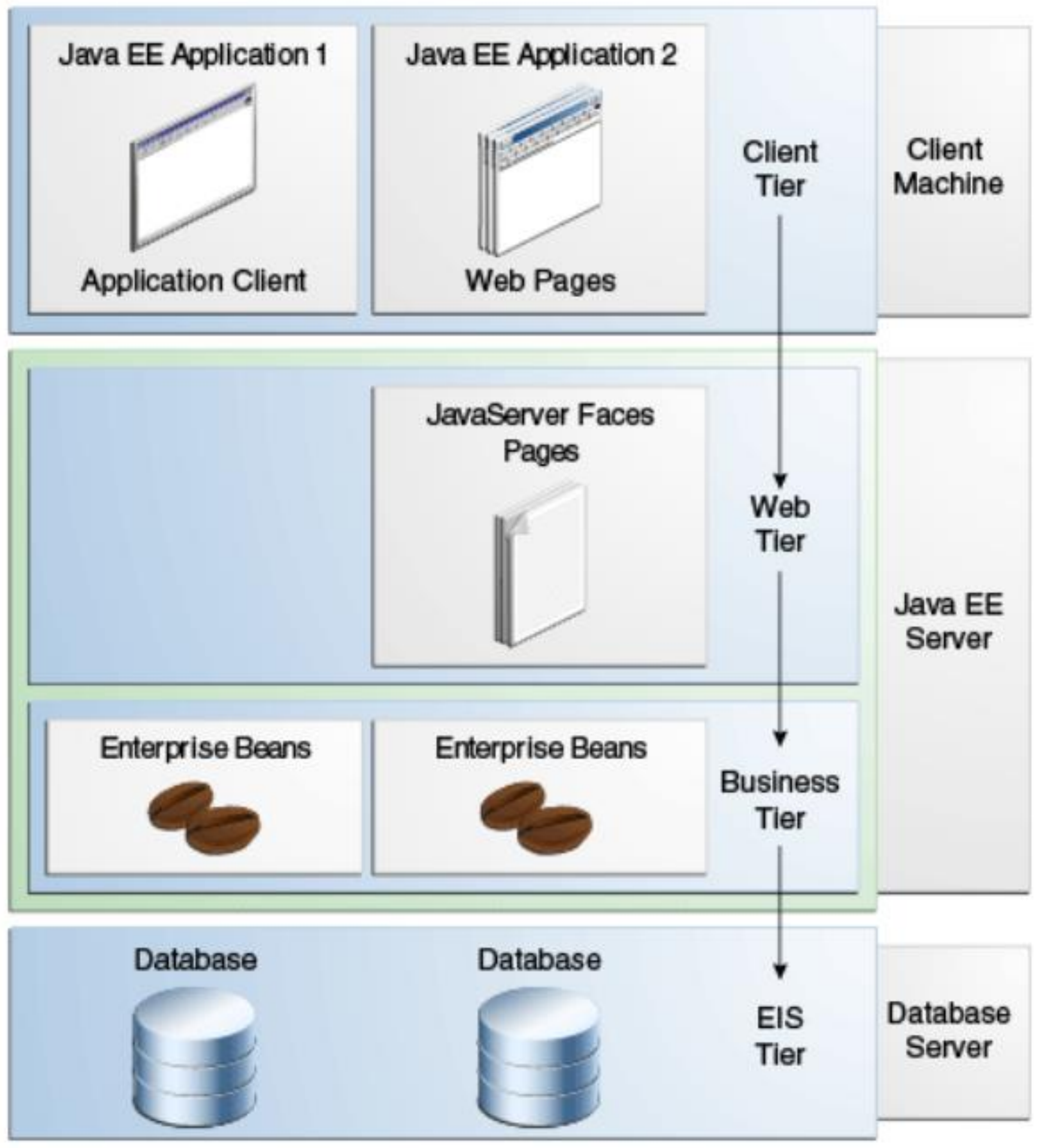
## Java EE

- The Java EE platform is built on top of the Java SE platform.
- The Java EE platform provides an API and runtime environment for developing and running
  - large-scale
  - multi-tiered
  - scalable
  - reliable
  - secure network applications

# Distributed Multi-tiered Applications

- The Java EE platform uses a distributed Multi-tiered application model for enterprise applications.
- Application logic is divided into components according to function
- The application components that make up a Java EE application are installed on various machines depending on the tier in the Multi-tiered Java EE environment to which the application component belongs.

# Multi-tiered Applications



## Three-tiered Applications

- Although a Java EE application can consist of all tiers shown in the prior slide, Java EE Multi-tiered applications are generally considered to be three-tiered applications because they are distributed over three locations:
    1. Client machines
    2. The Java EE server machine, and the database or legacy machines at the back end.
    3. Three-tiered applications that run in this way extend the standard two-tiered client-and-server model by placing a multithreaded application server between the client application and back-end storage.

## Java EE Components

- Java EE applications are made up of components.
- A Java EE component is
  - a self-contained functional software unit that
  - is assembled into a Java EE application with its related classes and files
  - communicates with other components.
- The Java EE specification defines the following Java EE components:
  - Application clients and applets are components that run on the client.
  - Java Servlet, JavaServer Faces, and JavaServer Pages (JSP) technology components are web components that run on the server.
  - EJB components (enterprise beans) are business components that run on the server.

## Java EE Components

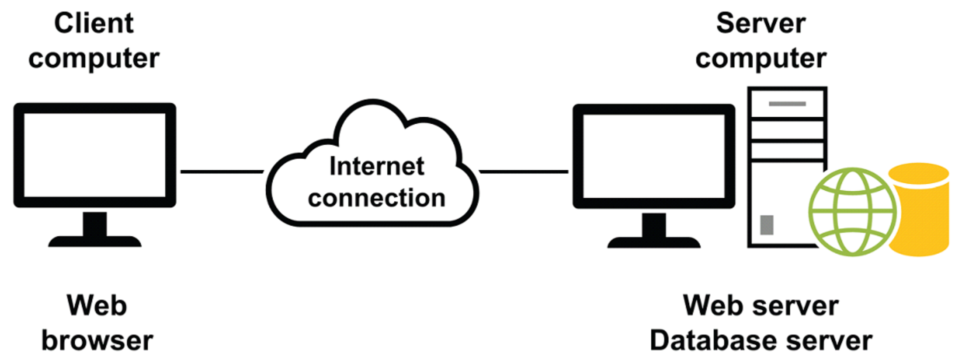
- Java EE components are written in the Java programming language and are compiled in the same way as any program in the language.
- The differences between Java EE components and "standard" Java classes:
  - Java EE components are assembled into a Java EE application
  - they are verified to be well formed and in compliance with the Java EE specification
  - they are deployed to production, where they are run and managed by the Java EE server.

## Java EE Components

- Client-tier components run on the client machine.
- Web-tier components run on the Java EE server.
- Business-tier components run on the Java EE server.
- Enterprise information system (EIS)-tier software runs on the EIS server.

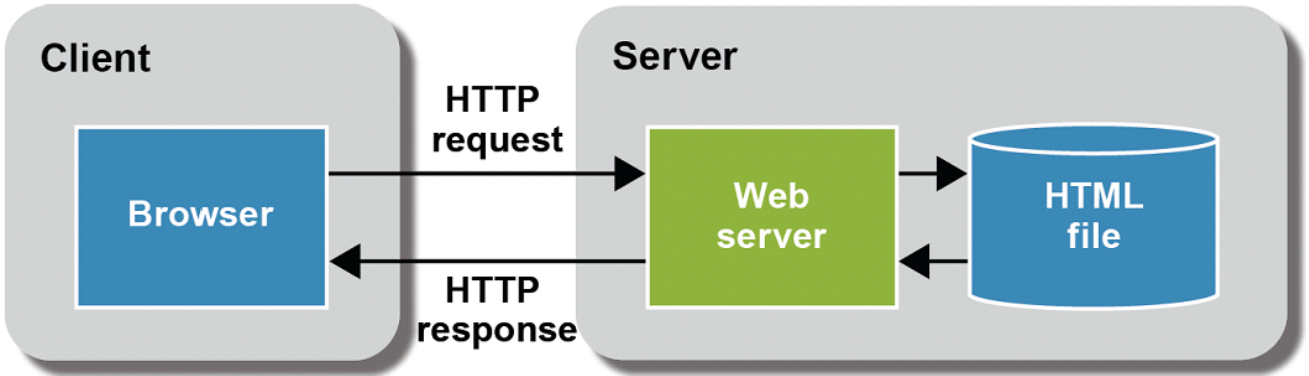
# Web Application

- **<u>Web Application:</u>** set of web pages that are generated in response to user requests.
  - Typical Client/ Server application.
- The user works with a *web browser* at the client computer.
- A web application runs on the server computer under the control of *web server* software. (like Apache)
- Server computer also runs a *database management system (DBMS).*



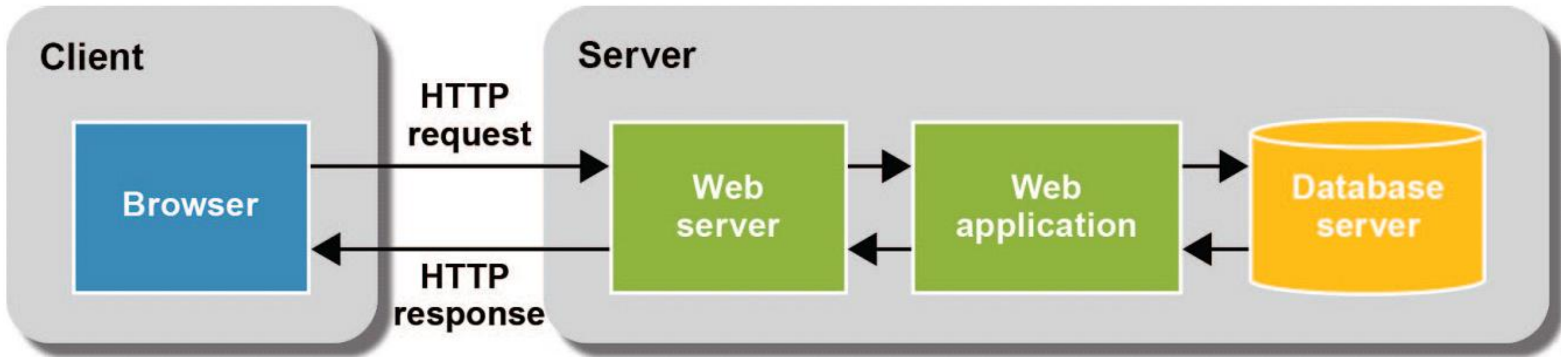
## Static Pages and how they work

- *HTML* is the language that the web browser converts into the web pages of a web application.
- ***Static web page*** is an HTML document stored in a file and doesn't change.
- *Http Request* is a server message sent by the browser for requesting a page.
- *Http Response* is a reply server message sent to the browser against its request.



# Dynamic Pages and how they work

- *Dynamic web page* is an HTML document that's generated by a web application, the web page changes according to the parameters sent to it.
- Server receives the request
- Server passes the request to the application.
- Application generates a response in form of HTML document and returns to web server.
- Web server wraps the HTML document with HTTP response and send it to the browser.

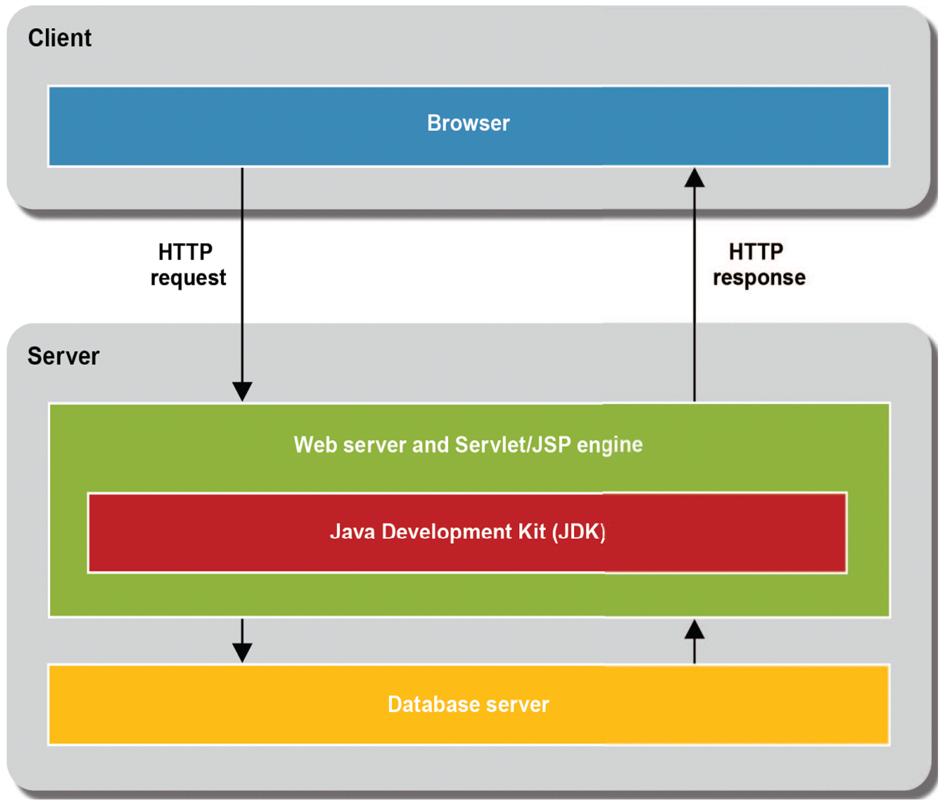


# Approaches for Java Web Application

- Three commonly used approached for Java web application development.
- **<u>Servlet/ JSP</u>**
  - *Servlets* store the java code that does the server-side processing.
  - *JSP (JavaServer Pages)* store the HTML that defines the user interface.
  - This is a low-level API, means developer has more control over different parts of the application like HTML, CSS and JavaScript.
- **<u>JavaServer Faces(JSF)</u>**
  - Next version which is designed to replace both *servlet* and *JSPs*.
  - Higher-level API, which generates lots of code automatically for the programmer.
  - You can also use EJBs (Enterprise JavaBeans) to define server side components.
- **<u>Spring Framework</u>**
  - Another high-level API which helps the programmer.
  - But Spring Framework also gives you the control over HTML, CSS and JavaScript as well.

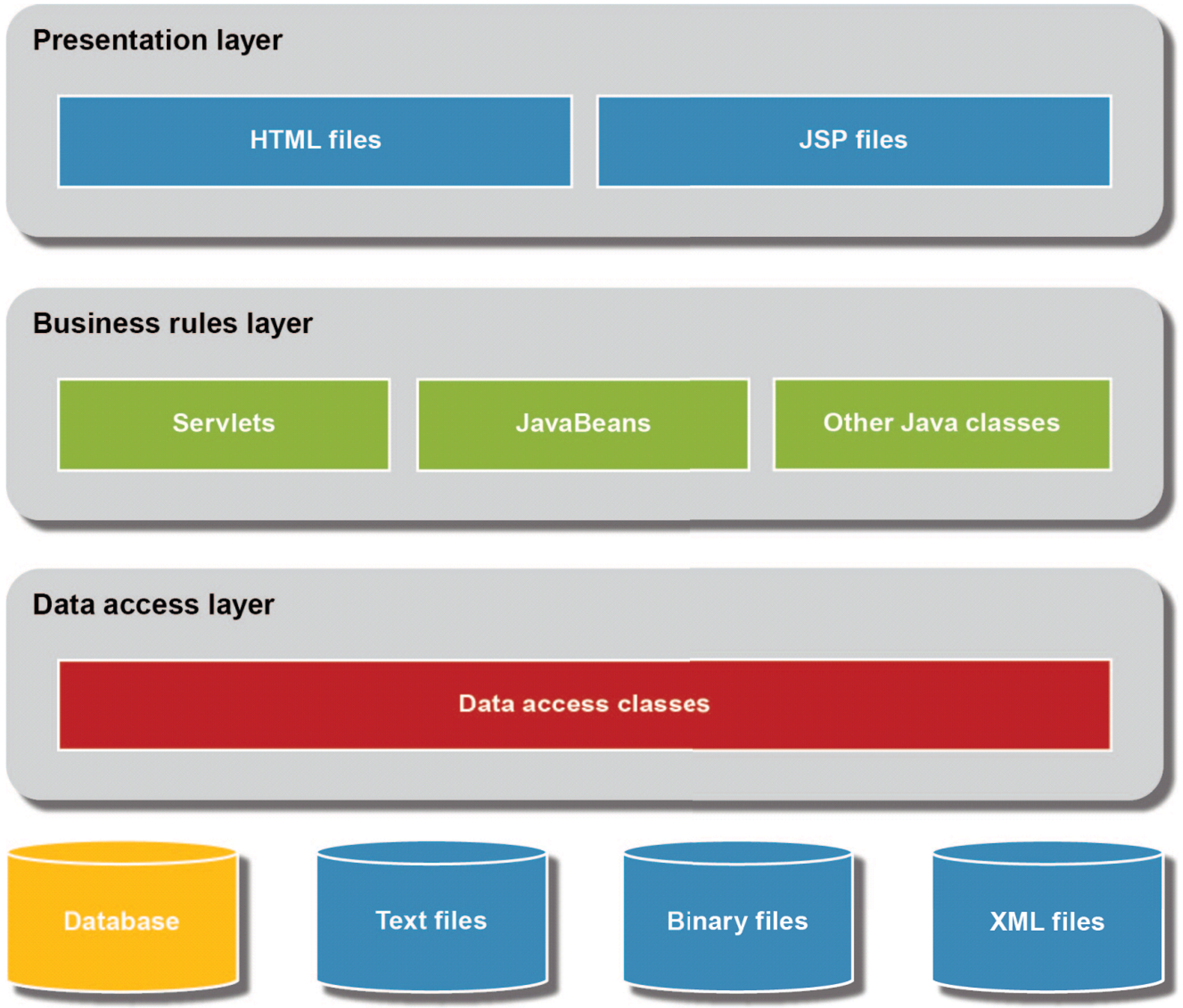
## Introduction to servlet/ JSP

- A servlet/JSP application consist of,
  - A Web Server
  - A Servlet/JSP engine.
- The engine is also know as a *container*.
- Engine is responsible for processing the HTTP request and generating the appropriate HTTP response against it.
- Servlet/JSP engine's required the JDK to work.



## The Architecture

- The architecture layers of typical web application uses servlets and JSP
  - *The Presentation Layer:* also known as *user interface layer.*
  - *The Business rules layer.*
  - *Data Access layer.*



## Web servers for Java Web Application

- Two popular web servers
  - Tomcat
    - Comes with a web server named Coyote
    - Servlet/JSP engine names Catalina.
    - Open-source

  - GlassFish
    - Is a complete Java EE application server.
    - Provides more features than Tomcat.
    - Disadvantage is it uses more resources than Tomcat.
    - Open-source

# Gears required for the examples

- Eclipse for Enterprise Java Developers.

https://www.eclipse.org/downloads/packages/release/2019-03/r/eclipse-ide-enterprise-java-developers

- MySQL

https://dev.mysql.com/downloads/windows/

- Apache Tomcat Server

https://www.eclipse.org/webtools/jst/components/ws/1.5/tutorials/InstallTomcat/InstallTomcat.html (Installation Instructions)