Advanced Java Technology

Mr. Prashant Sahatiya, Assistant Professor
Information Technology Engineering

CHAPTER-5

Java Enterprise Edition

Outline

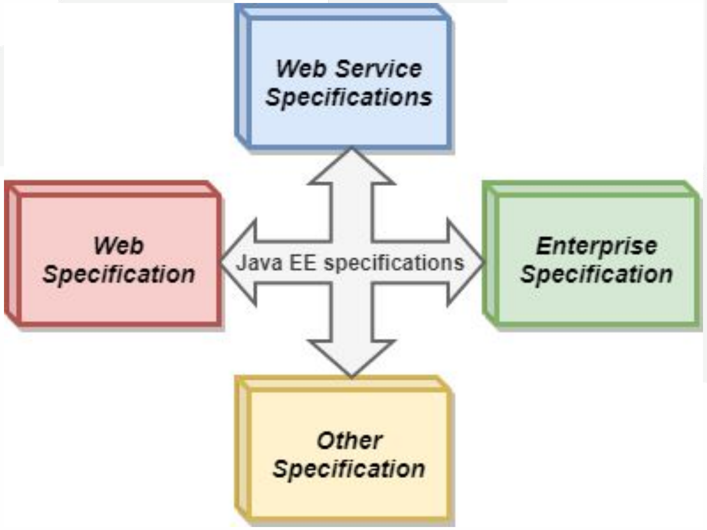
- Java Enterprise Edition
- Architecture
- Containers
- Developing applications
- Changes from Java EE 5 to Java EE 8

Java EE

- The **Java EE** stands for **Java Enterprise Edition**, which was earlier known as J2EE and is currently known as Jakarta EE. It is a set of specifications wrapping around Java SE (Standard Edition). The Java EE provides a platform for developers with enterprise features such as **distributed computing** and **web services**.
- Java EE applications are usually run on reference run times such as **microservers** or **application servers**. Examples of some contexts where Java EE is used are e-commerce, accounting, banking information systems.

Specifications of Java EE

- Java EE has several specifications which are useful in making **web pages**, reading and writing from **database** in a **transactional way**, managing **distributed** queues. The Java EE contains several APIs which have the functionalities of base Java SE APIs such as Enterprise JavaBeans, connectors, Servlets, Java Server Pages and several web service technologies.



1. Web Specifications of Java EE

- **Servlet-** This specification defines how you can manage HTTP requests either in a synchronous or asynchronous way. It is low level, and other specifications depend on it
- **WebSocket-** WebSocket is a computer communication protocol, and this API provides a set of APIs to facilitate WebSocket connections.
- **Java Server Faces-** It is a service which helps in building GUI out of components.
- **Unified Expression Language-** It is a simple language which was designed to facilitate web application developers.

2. Web Service Specifications of Java EE

- **Java API for RESTful Web Services-** It helps in providing services having Representational State Transfer schema.
- **Java API for JSON Processing-** It is a set of specifications to manage the information provided in JSON format.
- **Java API for JSON Binding-** It is a set of specifications provide for binding or parsing a JSON file into Java classes.
- **Java Architecture for XML Binding-** It allows binding of xml into Java objects.
- **Java API for XML Web Services-** SOAP is an xml based protocol to access web services over http. This API allows you to create SOAP web services.

3. Enterprise Specifications of Java EE

- **Contexts and Dependency Injection-** It provides a container to inject dependencies as in Swing.
- **Enterprise JavaBean-** It is a set of lightweight APIs that an object container possesses in order to provide transactions, remote procedure calls, and concurrency control.
- **Java Persistence API-** These are the specifications of object-relational mapping between relational database tables and Java classes.
- **Java Transaction API-** It contains the interfaces and annotations to establish interaction between transaction support offered by Java EE. The APIs in this abstract from low-level details and the interfaces are also considered low-level.
- **Java Message Service-** It provides a common way to Java program to create, send and read enterprise messaging system's messages.

4. Other Specifications of Java EE

- **Validation**- This package contains various interfaces and annotations for declarative validation support offered by Bean Validation API.
- **Batch applications-** It provides the means to run long running background tasks which involve a large volume of data and which need to be periodically executed.
- **Java EE Connector Architecture-** This is a Java-based technological solution for connecting Java servers to Enterprise Information System.

Java SE vs Java EE

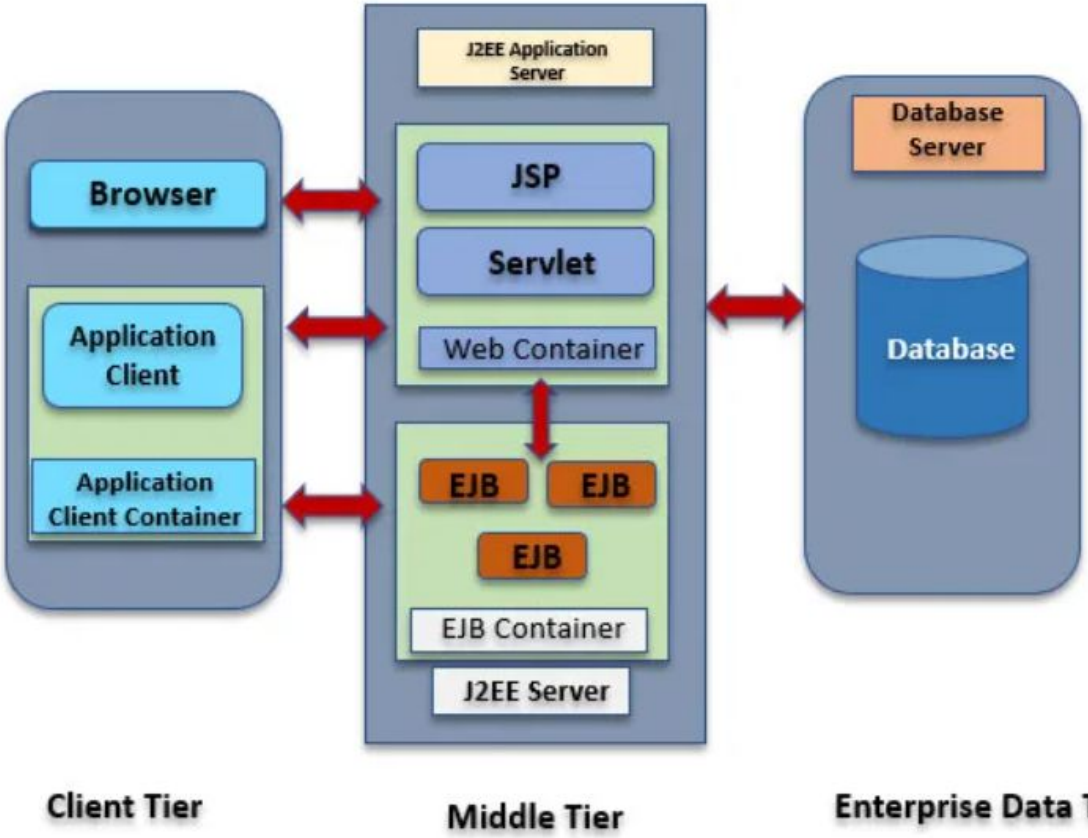
Java SE	Java EE
Java SE provide basic functionalities such as defining types and objects.	Java EE facilitates development of **large scale** applications.
SE is a normal Java specification	EE is built upon Java SE. It provides functionalities like **web applications** , and **Servlets** .
It has features like class libraries, deployment environments, etc.	Java EE is a structured application with a separate client, business, and Enterprise layers.
It is mostly used to develop APIs for Desktop Applications like antivirus software, game, etc.	It is mainly used for developing **web applications** .
Suitable for beginning Java developers.	Suitable for experienced Java developers who build enterprise-wide applications.
It does not provide user authentication.	It provides **user authentication** .

J2EE Architecture

J2EE Uses Three Tiers:

1. **Client Tier:** Client tier consists of user programs that interact with the user for request and response.
2. **Middle Tier:** Middle tier usually contains enterprise beans and web services that distribute business logic for the applications.
3. **Enterprise Data Tier:** Enterprise data is stored in a relational database. This tier contains containers, components and services.

J2EE Architecture



1. Client Tier

- The client tier consists of programs or applications interact with the user. Usually, they are located in a different machine from the server. Client tier prompts the user inputs into user requests then forwarded to the J2EE server then processed result returned back to the client. A client can be a web browser, standalone application or server that runs on a different machine.
- Clients can be classified as a **Web Client** and **Application Client**.

1. Client Tier

Web Client

- Web client consists of dynamic web pages of various mark-up languages that are generated by web components running in web tier or web browser which renders pages received from the server. Web clients are also called as thin clients that usually do not perform things like query database, execute business rules. When using thin client-heavy operations are offloaded to enterprise beans executing in the J2EE server.

1. Client Tier

Applets:

- Web pages received from web tier embedded an Applet these run on a web browser. Web components are APIs for creating a web client program. Web components enable the user to design cleaner and more modular applications. They provide a way to separate application programming.

1. Client Tier

Application Clients

- The application client runs on the client machine and handles the tasks that give richer user interfaces. Typically the GUI is created by Swings or AWT. Application clients can directly access EJBs running in the business tier using an HTTP connection.

2. Middle Tier (Web tier & EJB Tier)

Web Tier /Web Component

- Web components can be **servlets** or **JSP pages**. Servlets can dynamically process the request and generate the responses. Compared to JSP and servlets – servlets are dynamic pages to some extent but JSP pages are static in nature.
- During application assembly process Client's Static HTML programs and applet, codes are bundled in web tier/ Web Component. Actually these HTML and applets are not considered as elements of web components. Server-side utility classes are also bundled with web component but they are not considered as web components.
- Web tier might include EJB components for processing user inputs and sends the input to Enterprise bean running in the business tier.

2. Middle Tier (Web tier & EJB Tier)

EJB Tier /EJB Component

- Enterprise components handle usually business code that is logic to solve particular business domains such as banking or finance are handled by enterprise bean running in the business tier.

- Enterprise Container receives data from client processes if necessary, sends it to the enterprise information system for storage. Enterprise bean also retrieves data from storage, processes it and sends it back to the client.

3. Enterprise Information System

This tier consists of database servers, enterprise resource planning systems and other data sources. Resources are typically located on a separate machine than the J2EE Server and accessed by components on the business tier.

Technologies used in EIS Tier:

1. **Java Database Connectivity API (JDBC).**
2. **Java Persistence API.**
3. **Java Connector Architecture.**
4. **Java Transaction API.**

Containers in J2EE Architecture - Application Client Container

The container includes a set of **classes**, **libraries**, other files that are required to execute client programs in their own JVM. **Manages the execution of client components**. It also provides services that enable java client program to execute. This container is specific to the **EJB container**. Compared to other containers in J2EE this container is lightweight.

Features:

1. Security: responsible for collecting authentication Data such as User Name and Password and sends data over RMI/IIOP to the server. The server then processes the data using the JAAS module. Even though authentication techniques are provided by the client container but these are not under the control of the application client.
2. Naming: Allows the application clients to use Java Naming and Directory Interface (JNDI).

Containers in J2EE Architecture - Web Container

- **Web Container** is a component of the web server that interacts with **Java servlets**. A web container is responsible for managing the servlet lifecycle and mapping URLs. **Web container handles a request from Servlets and JSP files and other files that includes server-side code.**

- Web container implements a web component contract of the J2EE architecture. This provides a runtime environment for additional web components security, transaction, deployment, etc.

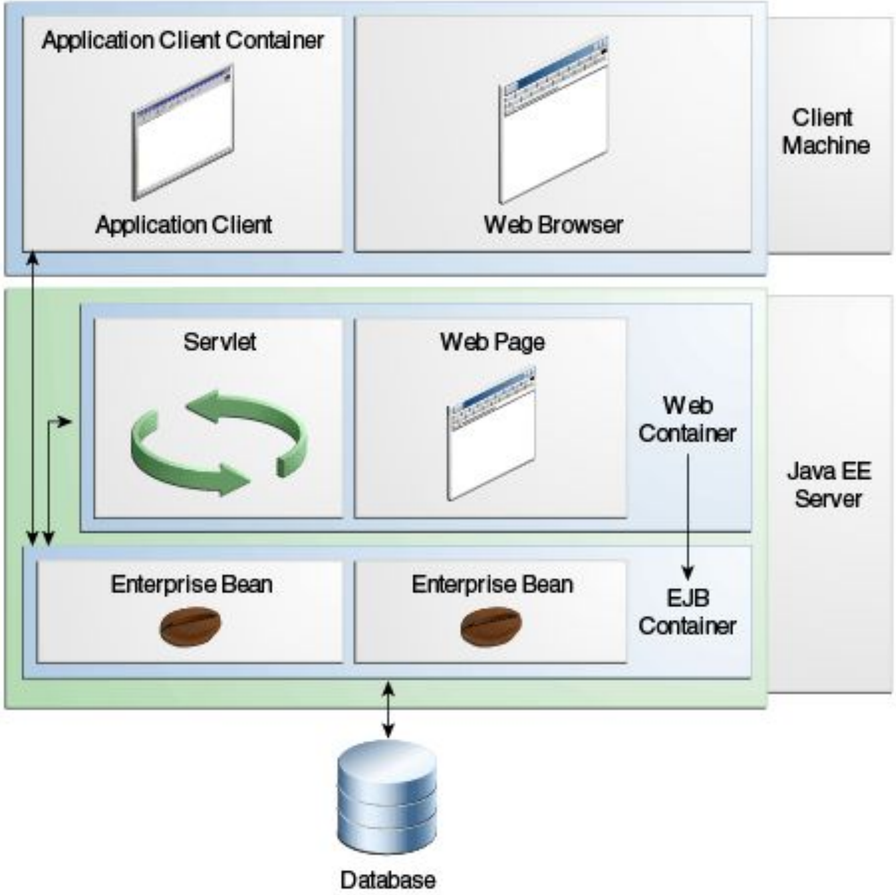
Containers in J2EE Architecture - EJB Container

- **Enterprise Java bean container** consists of server components that contain **business logic.** Provides local and remote access to enterprise beans. EJB container is responsible for creating enterprise bean, binding enterprise bean to the naming services.
- More than one module can be installed within a single EJB container. It performs transactional actions like – Start Transaction, Commits or Rollback transaction, manages various connection pools for database resources, synchronizing bean instance variables with corresponding data items that are stored in the database.

Containers in J2EE Architecture - Applet Container

- The container where the client’s Applet programs will run may be in a web browser or other application programs that support applet programing. Applets are subject to more restrictions due to the sandbox security model this limits access to the client machine. These are normal web pages downloaded from the web servers and executes on the client browser.

Containers in J2EE Architecture



Conclusion

- J2EE Three Tier Architecture which composed of three tiers logical computing. This helps in developing specific client-server based applications. It also helps in development by separating the User Interface, business logic, and data storage layer. Gives greater flexibility for development for updating specific parts of the application without affecting the rest of the parts. This flexibility can improve overall development and upgrading and replacing data from a specific tier without affecting the whole system.

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversity.ac.in