Fundamentals of Java Enterprise Components

Fundamentals of Java Enterprise Components

APTECH LIMITED
Contact E-mail: ov-support@onlinevarsity.com

Edition 1 - 2014

Dear Learner,

We congratulate you on your decision to pursue an Aptech course.

Aptech Ltd. designs its courses using a sound instructional design model – from conceptualization to execution, incorporating the following key aspects:

- Scanning the user system and needs assessment

Needs assessment is carried out to find the educational and training needs of the learner

Technology trends are regularly scanned and tracked by core teams at Aptech Ltd. TAG* analyzes these on a monthly basis to understand the emerging technology training needs for the Industry.

An annual Industry Recruitment Profile Survey# is conducted during August - October to understand the technologies that Industries would be adapting in the next 2 to 3 years. An analysis of these trends & recruitment needs is then carried out to understand the skill requirements for different roles & career opportunities.

The skill requirements are then mapped with the learner profile (user system) to derive the Learning objectives for the different roles.

- Needs analysis and design of curriculum

The Learning objectives are then analyzed and translated into learning tasks. Each learning task or activity is analyzed in terms of knowledge, skills and attitudes that are required to perform that task. Teachers and domain experts do this jointly. These are then grouped in clusters to form the subjects to be covered by the curriculum.

In addition, the society, the teachers, and the industry expect certain knowledge and skills that are related to abilities such as *learning-to-learn, thinking, adaptability, problem solving, positive attitude etc*. These competencies would cover both cognitive and affective domains.

A precedence diagram for the subjects is drawn where the prerequisites for each subject are graphically illustrated. The number of levels in this diagram is determined by the duration of the course in terms of number of semesters etc. Using the precedence diagram and the time duration for each subject, the curriculum is organized.

- Design & development of instructional materials

The content outlines are developed by including additional topics that are required for the completion of the domain and for the logical development of the competencies identified. Evaluation strategy and scheme is developed for the subject. The topics are arranged/organized in a meaningful sequence.

The detailed instructional material – Training aids, Learner material, reference material, project guidelines, etc.- are then developed. Rigorous quality checks are conducted at every stage.

- Strategies for delivery of instruction

Careful consideration is given for the integral development of abilities like thinking, problem solving, learning-to-learn etc. by selecting appropriate instructional strategies (training methodology), instructional activities and instructional materials.

The area of IT is fast changing and nebulous. Hence considerable flexibility is provided in the instructional process by specially including creative activities with group interaction between the students and the trainer. The positive aspects of web based learning –acquiring information, organizing information and acting on the basis of insufficient information are some of the aspects, which are incorporated, in the instructional process.

- Assessment of learning

The learning is assessed through different modes – tests, assignments & projects. The assessment system is designed to evaluate the level of knowledge & skills as defined by the learning objectives.

- Evaluation of instructional process and instructional materials

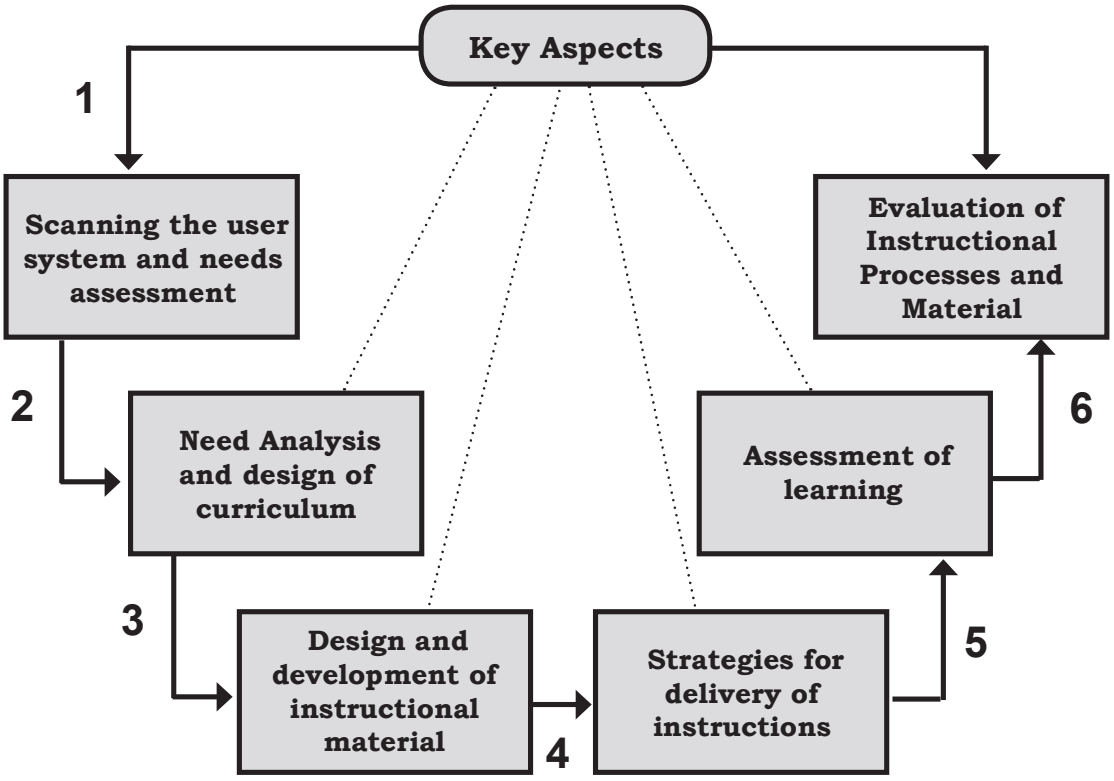
The instructional process is backed by an elaborate monitoring system to evaluate - on-time delivery, understanding of a subject module, ability of the instructor to impart learning. As an integral part of this process, we request you to kindly send us your feedback in the reply pre-paid form appended at the end of each module.

*TAG – Technology & Academics Group comprises of members from Aptech Ltd., professors from reputed Academic Institutions, Senior Managers from Industry, Technical gurus from Software Majors & representatives from regulatory organizations/forums.

Technology heads of Aptech Ltd. meet on a monthly basis to share and evaluate the technology trends. The group interfaces with the representatives of the TAG thrice a year to review and validate the technology and academic directions and endeavors of Aptech Ltd.

Industry Recruitment Profile Survey - The Industry Recruitment Profile Survey was conducted across 1581 companies in August/September 2000, representing the Software, Manufacturing, Process Industry, Insurance, Finance & Service Sectors.

Aptech New Products Design Model



Preface

Web and enterprise applications have become very popular today due to their efficiency and distributed nature. They can be used for different types of transactions and online activities. The use of Enterprise applications allows distribution of components at different levels that helps better management and troubleshooting in case of application errors.

This book has been designed to equip you with the knowledge required to develop distributed and efficient Web and enterprise applications. After reading this book, you will be able to identify and create components of Web and enterprise applications. It also introduces Web Services and security features available in Java EE applications.

The knowledge and information in this book is the result of the concentrated effort of the Design Team, which is continuously striving to bring to you the latest, the best and the most relevant subject matter in Information Technology. As a part of Aptech's quality drive, this team does intensive research and curriculum enrichment to keep it in line with industry trends and learner requirements.

We will be glad to receive your suggestions. Please send us your feedback, addressed to the Design Centre at Aptech's corporate office.

Design Team

Onlinevarsity

your e-way to learning

BI g

Balanced Learner-Oriented Guide

for enriched learning available @

www.onlinevarsity.com

Table of Contents

Sessions

ASK to LEARN

Onlinevarsity
your e-way to learning

Questions *in your* mind?

are here to HELP

Post your questions in the ASK to LEARN section for solutions.

Session - 1

Introduction to Java EE 7

Welcome to the Session, **Introduction to Java EE 7**.

This session provides an introduction to the Java enterprise model. It explains about multitier applications, application servers, and Web services. Further, it explains about deployment of enterprise applications on Web containers. Lastly, it explains the various APIs used in Java EE7.

In this Session, you will learn to:

- Explain Java Application model
- Explain multitier applications
- Describe various components of multitier applications
- Describe containers and services provided
- Describe Web services
- Describe various APIs used in Java EE 7 for enterprise and Web applications

1.1 Java Enterprise Application Model

The Java enterprise application model defines how various components of an enterprise application are organized. This organization is responsible for creating a portable, secure application with high developer productivity. The portability and security of the application are due to the Java programming language and Java Virtual Machine.

Java EE enables the creation of an enterprise application which caters to the needs of the customers, employees, suppliers, partners, and so on. These applications are very complex, where data is accessed from a variety of sources and has to be distributed among various targets. In order to manage this complex flow of data, a middle tier of the application is created.

Middle tier is run on a different server which has access to every component of the enterprise application. This can also be termed as the business logic of an application where it knows all about the organization of the data, processes of the enterprise (business logic), and client requirements.

Figure 1.1 shows an abstract representation of the Java enterprise application model.

Figure 1.1: Application Model of Java EE

The client tier represents various users of the enterprise application such as customers, employees, suppliers, and so on.

The middle tier handles the enterprise business logic. The enterprise business logic refers to the processes in the enterprise and this layer is aware of enterprise data as well as the client requirements. The middle tier processes the data received from the client with the help of the Data tier. That is, developers implement the business logic in this layer. The processed data is then sent back to the client and presented using the presentation logic on the client tier.

The enterprise application model divides the work required to implement the multitier service into two categories:

1. The business logic and presentation logic which are supposed to be implemented by the developer.
2. The standard system services which are generic for any application such as connecting to the database, establishing a session, and so on.

Concepts

The Standard system services are provided by Java EE 7 platform. There can be multiple tiers in an application based on requirement; in which case, the architecture is termed as n-tier architecture.

1.2 Multitier Applications

A multitier application has the application logic divided into multiple components based on their functionality. These multitier applications are distributed where each of the application components may be installed on the same or on different machines depending on the tier to which it belongs. In a typical enterprise application there are three tiers. Figure 1.2 illustrates the multitier architecture and its respective components.

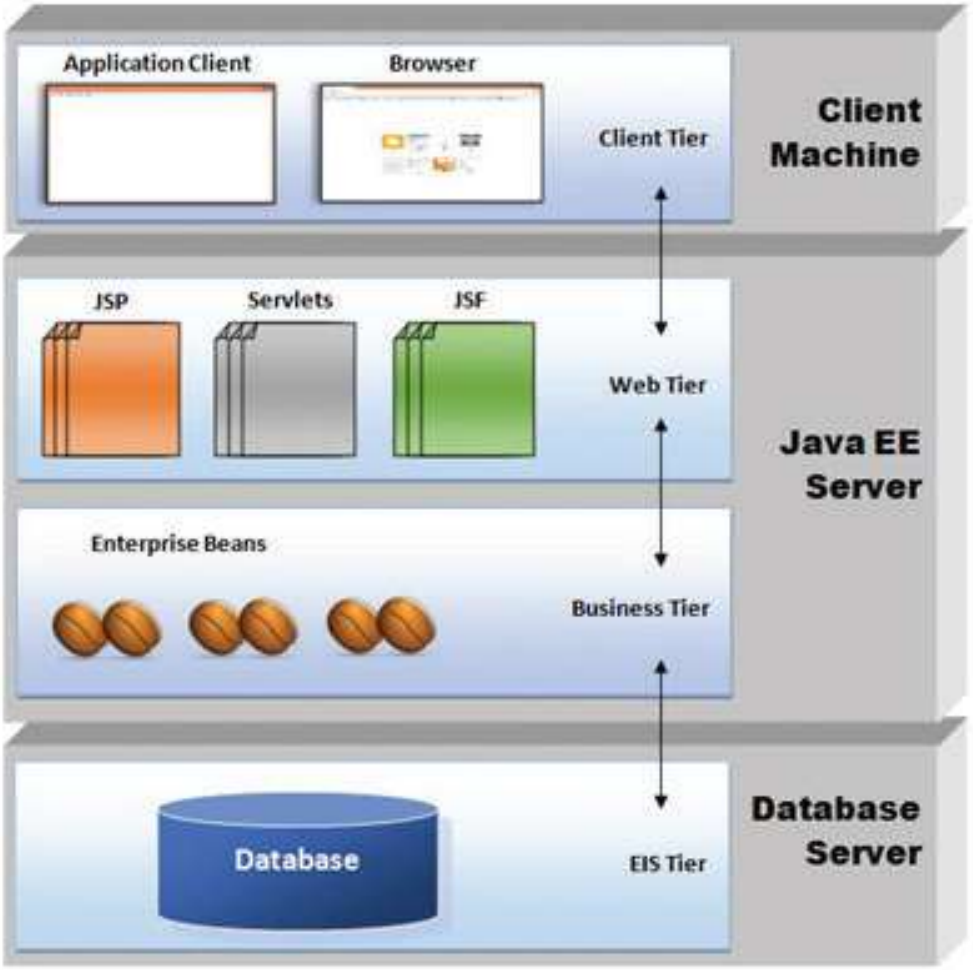


Figure 1.2: Multitier Applications

Various components of Java EE multitier application are as follows:

- Java EE clients
- Web components

Concepts

- Java EE components
- Database components
- Security components

A Java EE application comprises various components that function together. A component can be defined as a fully functional software unit which can communicate with other components.

There are three essential components that are widely used in applications:

1. Application clients
2. Web components
3. Enterprise components

Application clients form the interface for the usage of the application. Examples for application clients are Web browsers for Web applications, user forms for enterprise applications, and so on.

Web components refer to client side scripts such as JavaScript, server side scripts such as Servlets, Java Server Faces, and so on.

Enterprise components refer to the business components such as Enterprise Java Beans (EJB) which run on the server.

1.2.1 Java EE Clients

Java EE clients can be:

- Web clients
- Application clients
- Applets
- Java Bean components

A Web client consists of two parts – dynamic Web pages and Web browsers. Dynamic Web pages are generated by the Web components in the Web tier through markup languages such as HTML and XHTML. The Web browser is responsible for rendering the Web pages.

An application client runs on the client machine and has a user interface such as forms developed through Abstract Window Toolkit (AWT) or Swing. These application clients communicate with the business logic which in turn is implemented through EJB.

Applets can be defined as Web clients that are written in Java language. These are stand alone applications executed through the Web browser.

The multitier application model is an extension of the basic client server application model with additional middle tier and other tiers such as security tiers. Java bean components can be present in both client and server tiers. These bean components manage the data flow between the application client and components running on the Java EE server and also, the data flow between the server components and a database.

1.2.2 Web Components

Web components in an application can either be a Web page at the client end or a script running at the server end to service client requests. Servlets are the server side scripts and on the client side there are Web pages generated through Java Server Pages (JSP) and Java Server Faces (JSF).

Servlets are invoked in response to a user request and construct a response page after processing the input. JSP pages are text based documents which execute as servlets. JSFs are built on top of both JSP and Servlets, primarily used to create the user interface.

Static HTML pages and applets as well as server side utility classes are not considered as Web components according to the Java EE specification.

1.2.3 Java EE Components

Java EE Components are business components that implement the logic of application domains such as banking, health care, mobile services, and so on. These components are essentially enterprise beans running in the application tier and client tier.

The enterprise bean is capable of receiving data from both the client tier and the data tier. The bean receives the data from the client tier processes it and stores it in the storage. Similarly, it can access data from the data storage, process it, and send it to the client.

1.2.4 Enterprise Information System Tier

This is the data repository of the application. This tier includes management of enterprise infrastructure systems such as resource planning, mainframe transaction processing, database systems, and so on.

1.2.5 Security

Java EE enables definition of the security constraints at the time of deployment. The application developers are shielded from the complexity of implementing these security features. The developer only defines the security features through a set of declarative access control rules. The implementation of these rules is taken care by the Java EE platform. Java EE also provides various login mechanisms and therefore, the developer does not require making efforts in implementing these login mechanisms.

1.3 Web and Application Servers

The Java EE components must be packaged and deployed on a server to make them available on the Web.

Figure 1.3 shows the Web and Application servers.

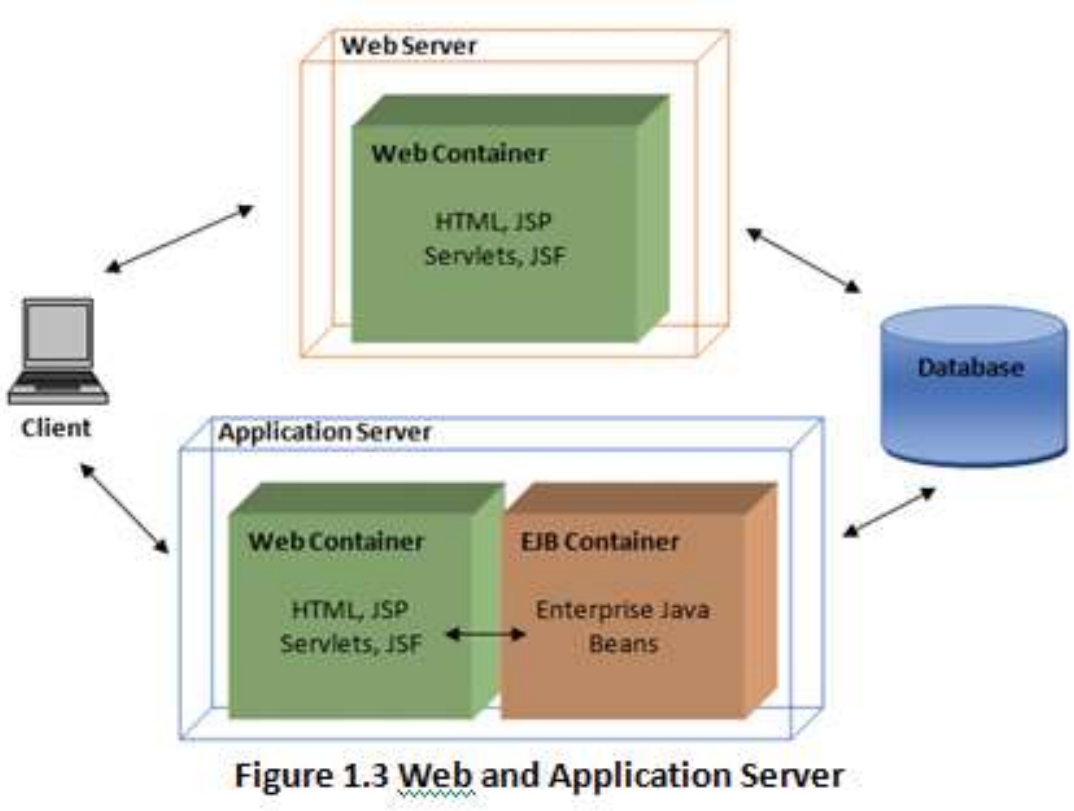


Figure 1.3 Web and Application Server

The Web server consists of a Web container that holds all the Web components. The application server consists of Web containers as well as an EJB container that holds all the enterprise Java bean components. Based on the requirement of the application, a Web server or an application server can be used for deployment.

1.3.1 Web Servers

Web servers accept requests from clients through Web pages and respond to these requests with the appropriate Web page. Web servers can be implemented through hardware or software and are primarily responsible for delivering Web content to the clients.

The communication between the client and Web server takes place through Hyper Text Transfer Protocol (HTTP) and the response to the client request is built in the form of HTML pages. The request from the client is interpreted through server side scripts such as Servlets.

1.3.2 Application Servers

An application server is a software component which is responsible for all the operations of an enterprise and for implementing the business logic of the enterprise.

Concepts

For example, consider an enterprise application for banking domain, the application server should implement the functions required by the employees of the bank where employees interact with the database and carry out various transactions.

The application server of a bank should also function along with a Web server which provides the facilities of Internet banking to its customers. The application server will be aware of the business logic of the bank. The business logic in this case can be the interest rate applicable to certain accounts, the penalty for less than minimum balance situation, and so on. The application server is responsible for implementing all these domain specific issues.

1.4 Containers

Java EE simplifies the application development process by providing certain services to the developer through containers and components.

A container contains various components which together provide some services to the application. The services include transaction management, state management, resource pooling, thread pooling, security, and so on, which are utilized by the developers in majority of the applications. These services help in reducing the effort of developing these functionalities from scratch. These services such as transaction management, resource pooling, and so on can be implemented through different components and put together in a single container according to the requirement of the application. The support and services provided by containers and components enable the developer to concentrate on implementing the actual business logic. Figure 1.4 displays containers in Java EE.

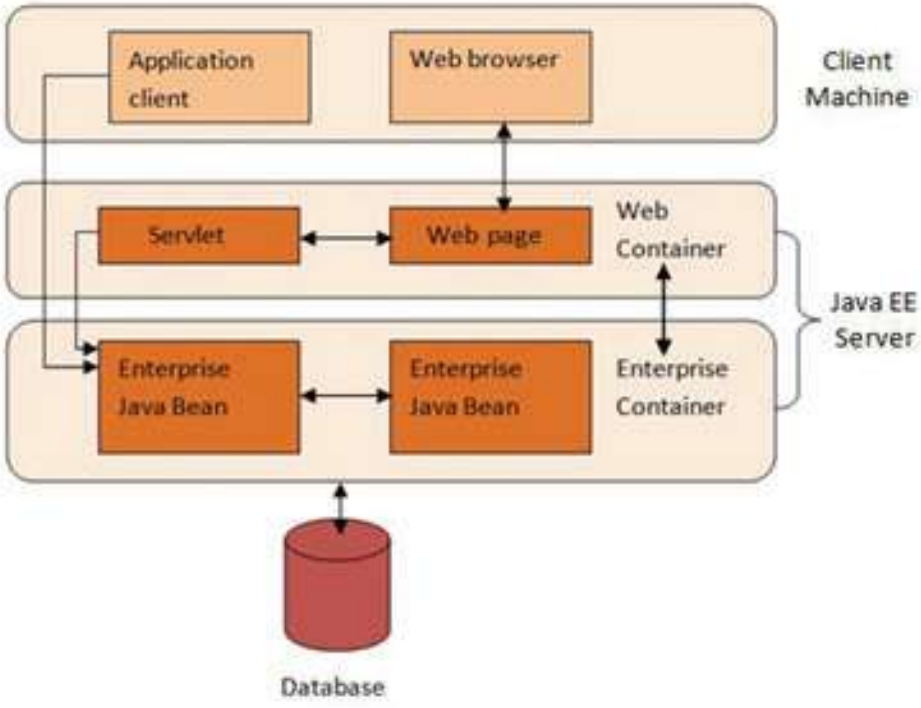


Figure 1.4: Containers in Java EE

1.4.1 Container Services

In order to use the container according to the requirement of the application, it is essential to configure the container settings. Following is a list of services provided by containers in a Java enterprise application:

- When an enterprise or Web application requires security mechanisms, Java EE enables you to configure the components to define access to various resources of the applications.
- Various methods can be put together to form a Java transaction in which case, Java EE transaction model enables you to establish the link between various methods.
- In scenarios where the server and client are located at different geographical locations, Java EE remote connectivity manages communication between the client and the server. Java EE client ensures that the client is unaware of the actual location of the server.
- Java Naming and Directory Interface (JNDI) provides naming service to access different objects and components in the enterprise application domain.

The container allows different configurations of the same application component based on the context of the application. The container manages non-configurable services such as enterprise beans, servlet lifecycles, database access, and so on. The different containers and their interaction are shown in figure 1.4.

1.4.2 Container Types

Various components of the enterprise application are installed in different containers during deployment. They are:

- Web Container
- Enterprise Java Beans Container
- Applet Container
- Application Client Container

The Web container is installed on the Java EE server, which comprises all the components of the Web applications. The Web container is responsible for handling HTTP communication, execution of servlets, managing the life-cycle of Web components, and so on.

The EJB container is also present on the Java EE server, which consists of the EJB components implementing the business logic of the application.

The Applet container manages the execution of applets. This container comprises the Web browser and plug-ins.

The Application Client container manages various application clients.

1.5 Development and Deployment

Developing a Java enterprise application involves various stages as follows:

- ➔ Procuring Java platform and tools for development
- ➔ Development of components by application component providers
- ➔ The components generated by the application providers are assembled by the assemblers and further deployed by the deployers

Product providers are the vendors who enable purchase of Java EE APIs and other features defined in the Java EE specification.

Tool providers are those vendors who provide application development tools such as Eclipse, NetBeans, and other tools for assembly, deployment, and packaging.

Application components include the bean component for enterprise applications, Web component for the Web applications and application clients. Following are the application component providers:

1. A Bean developer writes and compiles the source code, creates the deployment descriptor, and packages the .class file as an EJB JAR file.
2. A Web component developer will write source codes for servlets, JSPs, JSFs, and HTML files. The .class files, .jsp files, and .html files created are packaged as WAR files. As in case of Bean, the source code of the application client is written by an application client developer and compiled to generate a .class file along with the deployment descriptor for the class file. The .class files are then packaged as a JAR file.
3. An Application assembler receives all the components from the application component providers and assembles the respective JAR and WAR files to EAR (Enterprise Archive) file. The assembler may or may not specify the deployment descriptor by adding appropriate XML tags.
4. An Application deployer is responsible for deploying the application in the operational environment. The deployer configures the operational environment based on the specifications provided by the application component provider to resolve external dependencies, specify security settings, and assign transaction attributes.

A Java enterprise application is packaged as one or more standard units for deployment. Each unit has a functional component/components and deployment descriptor. A single developer can play all the mentioned roles or these roles can be divided among individuals and teams. This kind of division is possible because each stage outputs a portable file which is an input to the subsequent stage.

1.5.1 APIs in Java EE7

Java EE 7 provides a set of APIs that can be used for development of Web and enterprise applications.

Figure 1.5 displays the APIs used in EJB containers. The Concurrency utilities, Batch processing, and JSON-P APIS are new inclusions in Java EE7.

Figure 1.5: APIs Used in EJB Container

Figure 1.6 displays the APIs used in Web containers.

Figure 1.6: APIs Used in a Web Container

Concepts

Apart from components such as EJB, Servlets, Java Server Pages, and Java Server Faces, various APIs and libraries are required in Java EE application development. Following is a brief description of the APIs and libraries that are widely used in application development.

1.5.2 Java Server Pages Standard Tag Library

JSTL is a set of JSP tags that are widely used in JSP applications. The tags in JSP have varying functionalities such as tags used for formatting the Web page, tags for iterating and managing the flow control, tags for accessing database, and so on. Java EE 7 uses JSTL 1.2.

1.5.3 Java Persistence API

Java Persistence API is used for bridging the gap between object-oriented interpretation of data and relational data in databases. A persistence entity is a lightweight class representing a table in the relational database. The instances of the class represent rows of the table.

The Persistence API has a query language for the entities in the database. Java EE7 uses Java Persistence API 2.1.

1.5.4 Java Transaction API

Java Transaction API is responsible for transaction management in the application database. It is responsible for commit and rollback operations on tables of the database and maintains database integrity. It ensures that every transaction reads appropriate and accurate data.

1.5.5 Java API for RESTful Web Services

REST stands for Representational State Transfer, which is an architectural style to develop Web services. REST defines certain architectural properties for the application such as performance, scalability of the application, simplicity of the interface, ability to modify the components, portability of component deployment, and so on. The API provides means to develop Web services with the given qualities. Java EE 7 requires JAX-RS2.0.

1.5.6 Managed Beans

Managed Beans are light weight objects running on the Java Virtual Machine, responsible for management related tasks of the application. They are responsible for reporting events such as faults, state changes, performance of the application, and so on. These managed beans can be used in both Web applications and enterprise applications. They are part of Java EE 7 specification that uses Managed Beans 1.0.

1.5.7 Contexts and Dependency Injection for Java EE

CDI is a framework which is an integral part of Java EE platform. It provides an architecture that enables various components of an application such as servlets, beans, and so on to exist only within the life cycle of the application. It defines the context of each component. It allows integration of components in a loosely–coupled but type safe manner.

The components are injected (activated) into the context of the application life cycle based on certain events. The components when added to context, CDI ensures that it is done in a type safe manner holding all the dependencies intact.

1.5.8 Bean Validation

Bean Validation process is responsible for validating the data in beans in various tiers of application. Instead of distributing the task of validation across multiple layers an API is defined along with the metadata model for validation of beans. The API enables defining the validation constraints in one location which is further used across layers. Java EE7 requires Bean Validation 1.1.

1.5.9 Java Messaging Service API

The Java Messaging Service API is used by the components to create, send, receive, and read messages. This messaging standard is used to enable distributed communication which is reliable and asynchronous.

1.5.10 Java EE Connector Architecture

The Connector Architecture is responsible for creating resource adapters that will access Enterprise Information Systems. These resource adapters are used by Java EE applications. A resource adapter is a specific resource manager. One of the common resources in the EIS layer is database.

The connector architecture also integrates the Web services with the existing EISs and ensures that the integration is performance oriented, secure, and scalable.

1.5.11 JavaMail API

Java Mail API enables sending mail notifications. The API has an interface used by application components that are used to send mail and a service provider interface, where service provider refers to the network carrier of the mail.

1.5.12 Java Authorization Contract for Containers

The JACC defines a contract between the Java EE components of the application and authorization policy providers. Based on the authorization policy java.Permission.security classes are defined by the JACC. This is in the case of third party policy providers.

The JACC provides a set of standard permission types relevant for all enterprise and Web applications. It also provides a standard model for role-based authorization, grouping certain permissions along with the roles defined.

1.5.13 Java API for Web Socket

Web Socket refers to an application protocol which provides full duplex communication between two communicating entities. The Web Socket API helps in creating the communication end point and allows specifying the configuration parameters. This is a new API introduced in Java EE 7.

1.5.14 Java API for JSON Processing

JSON refers to a text based data exchange format, which is used in Web services and other applications. The JSON data exchange format is derived from JavaScript. The API enables Java EE applications to parse, transform, and query data from JSON using the object model or streaming model.

1.5.15 Concurrency Utilities for Java EE

The Concurrency Utilities of Java are responsible for providing asynchronous capabilities to enterprise application components. These utilities include managed executor service, managed scheduled executor service, managed thread factory, and so on.

1.5.16 Batch Applications for the Java Platform

Batch applications refer to the set of applications which do not have extensive input and output. The Batch Applications framework for Java enables creating and executing batch jobs. The batch framework has a batch runtime, job specification language based XML, an API to interact with the Java runtime, and an API to implement batch artifacts.