Java Web Programming With Java EE SDK

Markus Veijola 2013

Abbreviations

- API = Application Programming Interface
- CDI = Context and Dependency Injection
- EIS = Enterprice Integration Service
- EJB = Enterprise JavaBeans
- EL = Expression Language
- JAAS = Java Authentication and Authorization Service
- Java EE = (Java) Enterprise Edition
- JDBC = Java Database Connectivity API
- JMS=Java Messaging Service
- JNDI = Java Naming and Directory Interface API
- JPA = Java Persistence API
- JPQL = Java Persistence Query Language
- JSON = JavaScript Object Notation
- JSP = Java Server Pages
- JSF = Java Server Faces
- JTA = Java Transaction API
- ORM = object-relational mapping
- SOAP = Simple Object Access Protocol
- WSDL = Web Services Description Language
- XML = Extensible Markup Language

Content

- Overview
- Application Model
- Distributed Multitiered Applications
- Java EE Components
- Java EE Containers
- Java Server Faces Technology
- Enterprise Beans
- Java Servlet Technology
- Internationalizing and Localizing Web Applications
- Dependency Injection
- The Web Tier
- Web services
- JPA
- Java Persistence Query Language
- Security

Tools

- What you need for completing exercises:
  - Net Beans IDE v. 7.3.1
  - Glass Fish Server 4.0
  - JPA version 2.0
  - Java Server Faces 2.2
  - Maven 2.2.1
- You can download Net Beans IDE with all ingredients mentioned above from here:

  https://netbeans.org/downloads/

References

- JSF HTML Tag Library Docs:JSF Docs
- web.xml documentation:web.xml doc
- Java EE Documentation:Java EE Docs
- JPQL Docs: JPQL
- Net Beans IDE: NetBeans
- Java EE Annotations Reference: Java EE Annotations
- Annotations Tutorial: Annotation Tutorial

Overview

Overview

- The Java EE platform is built on top of the Java SE platform.
- Java EE provides an API and run time environment for developing and running large-scale, multi-tiered, scalable, reliable, and secure *network applications* a.k.a *enterprise applications*.
- Enterprise applications are designed to solve the problems encountered by large enterprises like agencies, governments and big companies.
- Enterprise applications can be useful also to smaller units.

Overview

- The aim of the Java EE platform is to provide developers with a powerful set of APIs while *shortening development time,reducing application complexity, and improving application performance → in short saving the MONEY*

Overview

- The Java EE 7 platform includes the following new features:
  - Batch Applications
  - Concurrency Utilities
  - Java API for JSON processing
  - Java API for WebSockets
  - New features for Servlets
  - New features for EJBs
  - New Features for JSF

Overview

- There are many frameworks, tools and components you can use along with Java EE SDK
  - Maven (project management and comprehension tool)
  - Spring (Framework)
  - Hibernate (ORM)
  - JPA (ORM)
  - Struts (Framework)

Application Model

Application Model

- The Java EE application model consists of:
  - Java Programming Language
  - Java Virtual Machine
- These two main elements forms the basis of the application model: portability, security, and developer productivity.

Application Model

- The Java EE application model *defines an architecture for implementing services as multitier applications that deliver the scalability, accessibility, and manageability needed by enterprise-level applications*.

Application Model

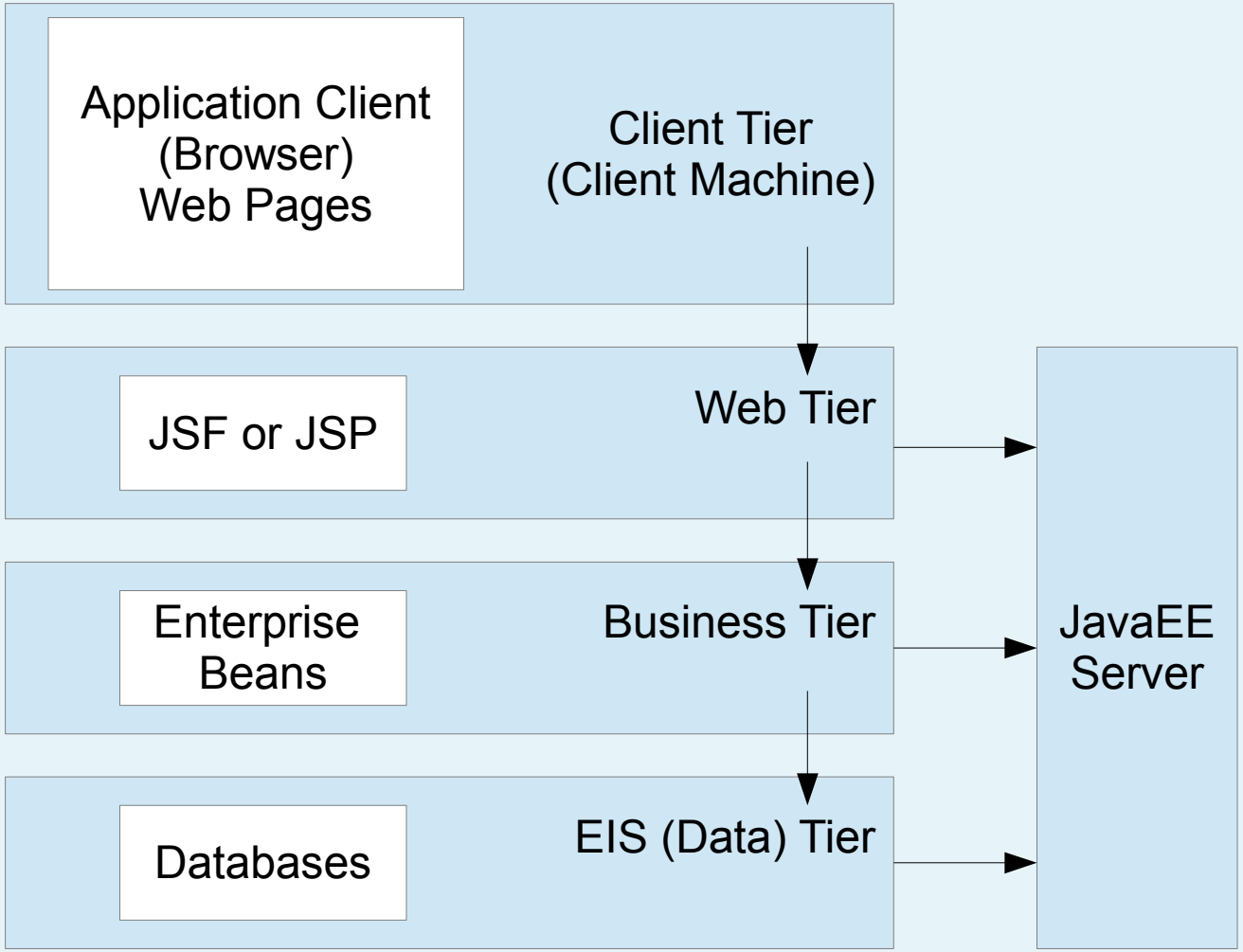
- The application model consists of following parts:
  - The business and presentation logic to be implemented by the developer
  - The standard system services provided by the Java EE platform
- The developer can trust on the platform to provide solutions for the hard systems-level problems of developing a multitier service.

Distributed Multitiered Applications

Tiered Applications

- In a multi-tiered application, *the functionality of the application is separated into isolated functional areas, called tiers*.
- Typically, multi-tiered applications have a client tier, a middle tier,business tier, and a data tier (often called the enterprise information systems tier).

Tiered Applications



Tiered Applications

- 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.

Tiered Applications

- Although a Java EE application can consist of all tiers, Java EE multitiered applications are generally considered to be three-tiered applications because they are distributed over three locations: client machines, the Java EE server machine, and the database at the back end.

The Client Tier

- The client tier consists of application clients that access a Java EE server and that are usually located on a different machine from the server. The clients make requests to the server. The server processes the requests and returns a response back to the client.
- Clients can be a web browser, a stand-alone application, or other servers.
- Client application doesn’t have to be a Java application.

The Web Tier

- The web tier consists of components that handle the interaction between clients and the business tier. Its primary tasks are the following:
  - Dynamically generate content in various formats for the client.
  - Collect input from users of the client interface and return appropriate results from the components in the business tier.
  - Control the flow of screens or pages on the client.
  - Maintain the state of data for a user's session.
  - Perform some basic logic and hold some data temporarily in JavaBeans components

The Web Tier

- The following Java EE technologies are used in the web tier in Java EE applications:
  - Servlets
  - JavaServer Pages
  - JavaServer Faces technology
  - JavaServer Pages tag library
  - JavaBeans components

The Business Tier

- The business tier consists of components that provide the business logic for an application.
- Business logic is code that provides functionality to a particular business domain, like the financial industry, or an e-commerce site.
- In a properly designed enterprise application, the core functionality exists in the business tier components

The Business Tier

- The following Java EE technologies are used in the business tier in Java EE applications:
  - Enterprise JavaBeans (EJB) components, referred to here as enterprise beans
  - JAX-WS web service endpoints
  - Java Persistence API entities

EIS (Data) Tier

- The enterprise information systems (EIS) tier consists of database servers, enterprise resource planning systems, and other legacy data sources, like mainframes. These resources typically are located on a separate machine than the Java EE server, and are accessed by components on the business tier.

EIS (Data) Tier

- The following Java EE technologies are used to access the EIS tier in Java EE applications:
  - The Java Database Connectivity API (JDBC)
  - The Java Persistence API
  - The J2EE Connector Architecture

Java EE Components

Java EE Components

- Java EE applications consists 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 and that communicates with other components which can be reused in different projects*.

Java EE Components

- The Java EE specification defines the following Java EE components:
  - Application clients are components that run on client machine.
  - Java Servlet, JavaServer Faces, and JavaServer Pages components are web components that run on the server.
  - Enterprise JavaBeans (EJB) components (enterprise beans) are business components that run on the server.

Java EE Components

- Each of these components can be implemented and assembled separately by different people or teams. This is a big benefit at least in bigger projects.
- Producing each of these components requires different kind of knowledge of Java EE technologies.

Java EE Containers

Containers

- When you write Java EE applications, there are many things that happens ”under the hood” mostly in Java EE server.
- Multitiered applications usually involve complicated code to handle transaction and state management, multithreading, resource pooling, and other complex low-level details.
- Component based approach makes Java EE applications easy to write because business logic is organized into reusable components.

Containers

- Java EE server provides underlying services in the form of a container for every component type.
- Because you do not have to develop these services yourself, you are free to concentrate on solving the business problem at hand.

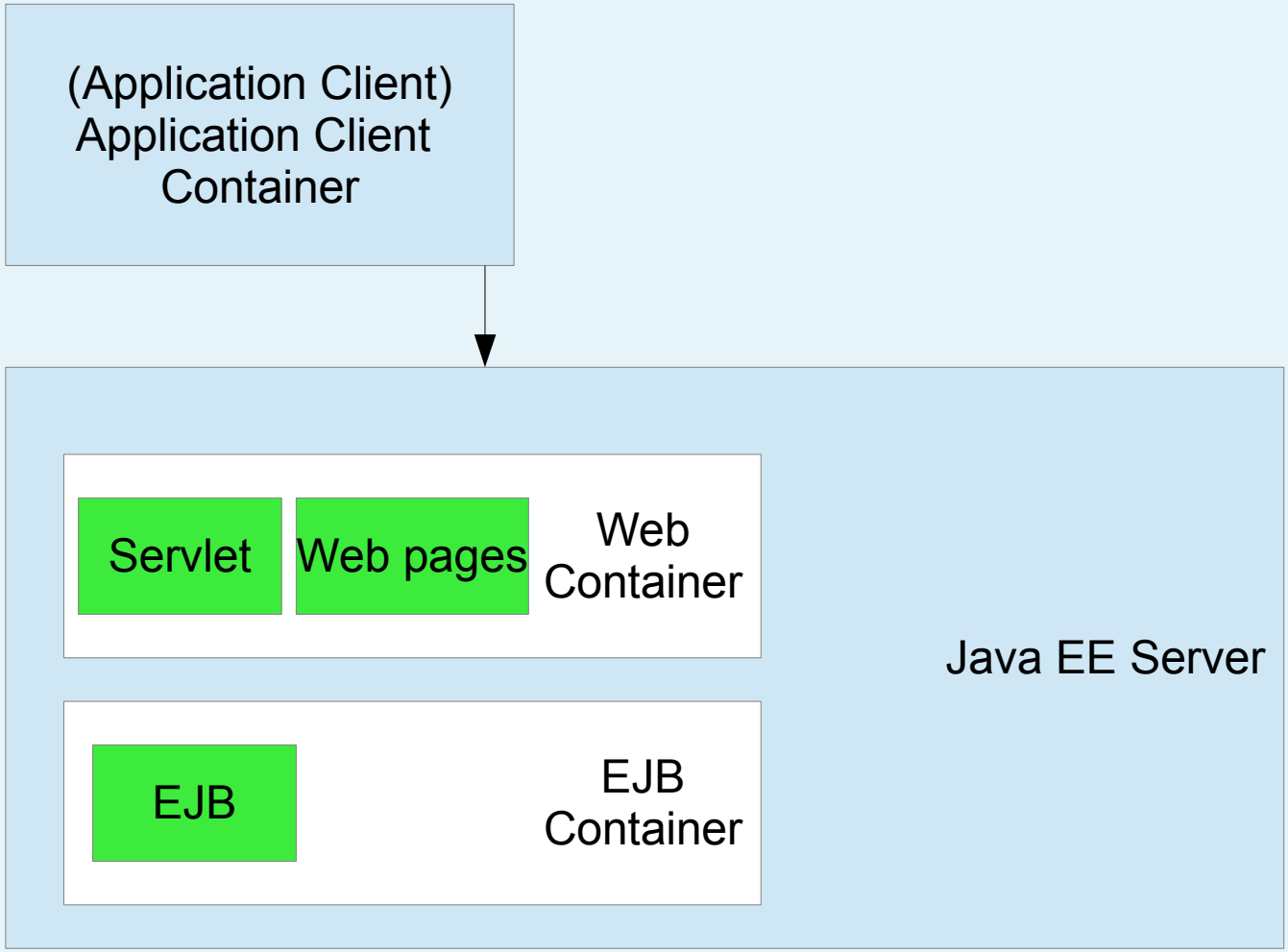
Containers

- Containers are the interface between a component and the low-level platform-specific functionality that supports the component.
- Before it can be executed, *a web, enterprise bean, or application client component must be assembled into a Java EE module and deployed into its container*.

Containers

- The tools we use to create Java EE applications makes these assembly modules for you automatically.
- You can configure how these assembly modules are created, for example you can configure web component security model.
- The components are illustrated in next figure.

Containers


(Application Client)
Application Client
Container
Servlet
Web pages
Web
Container
EJB
EJB
Container
Java EE Server

Containers

- Java EE server: The run time portion of a Java EE product. A Java EE server provides EJB and web containers.
  - EJB container:Manages the execution of enterprise beans for Java EE applications.
  - Web container: Manages the execution of web pages, servlets, and some EJB components.
  - Client container:Manages the execution of application client components.