

Java Editions



Java Card

Micro Edition (ME)

Standard Edition (SE)

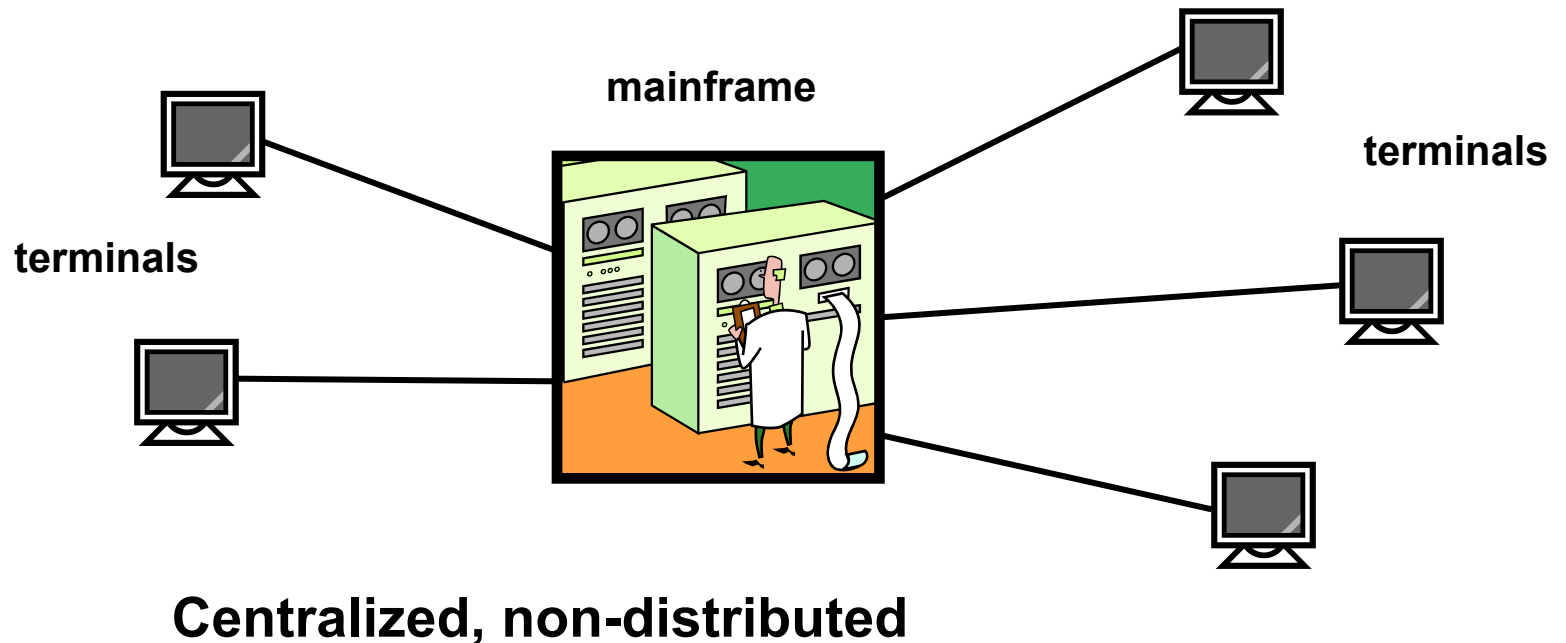
Enterprise Edition (EE)

JavaFX (Merged to Java SE 8)

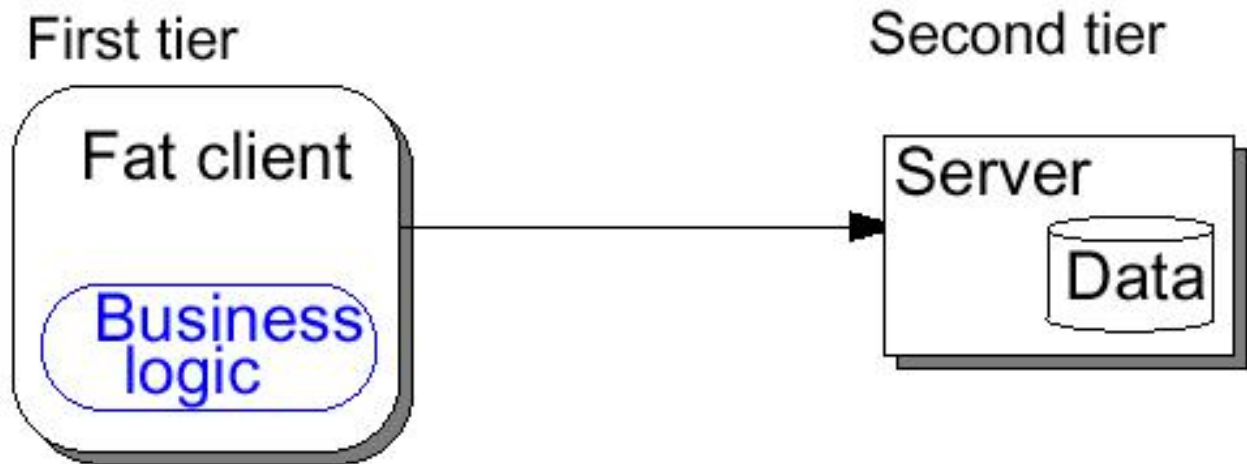
PersonalJava (discontinued)

In the beginning,

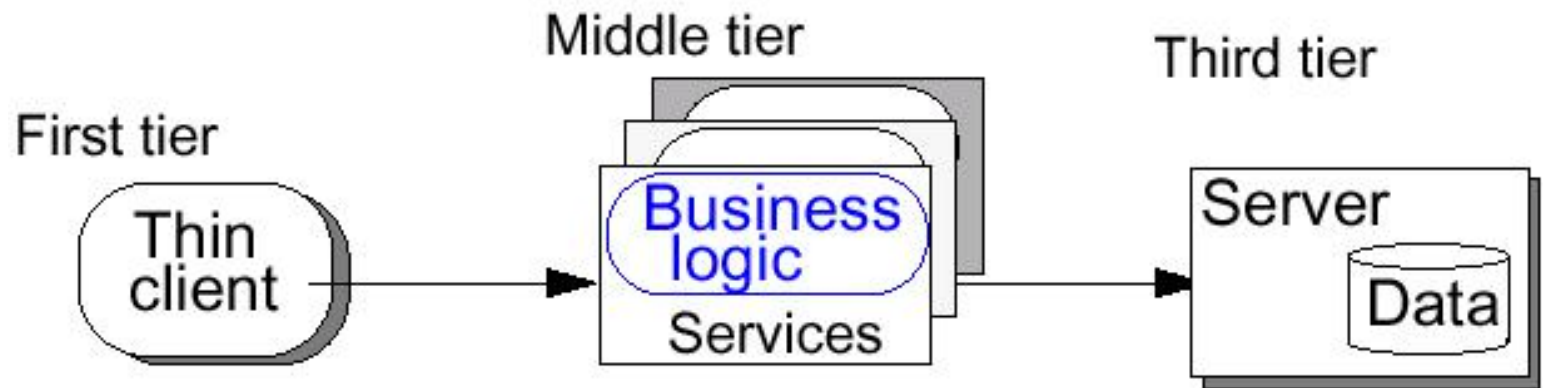
- In the beginning, there was darkness and cold.
Then, ...*



- In the 90's, systems should be *client-server*



- Today, enterprise applications use the *multi-tier* model



Application Servers

- “Multi-tier applications” have several independent components
- An *application server* provides the infrastructure and services to run such applications

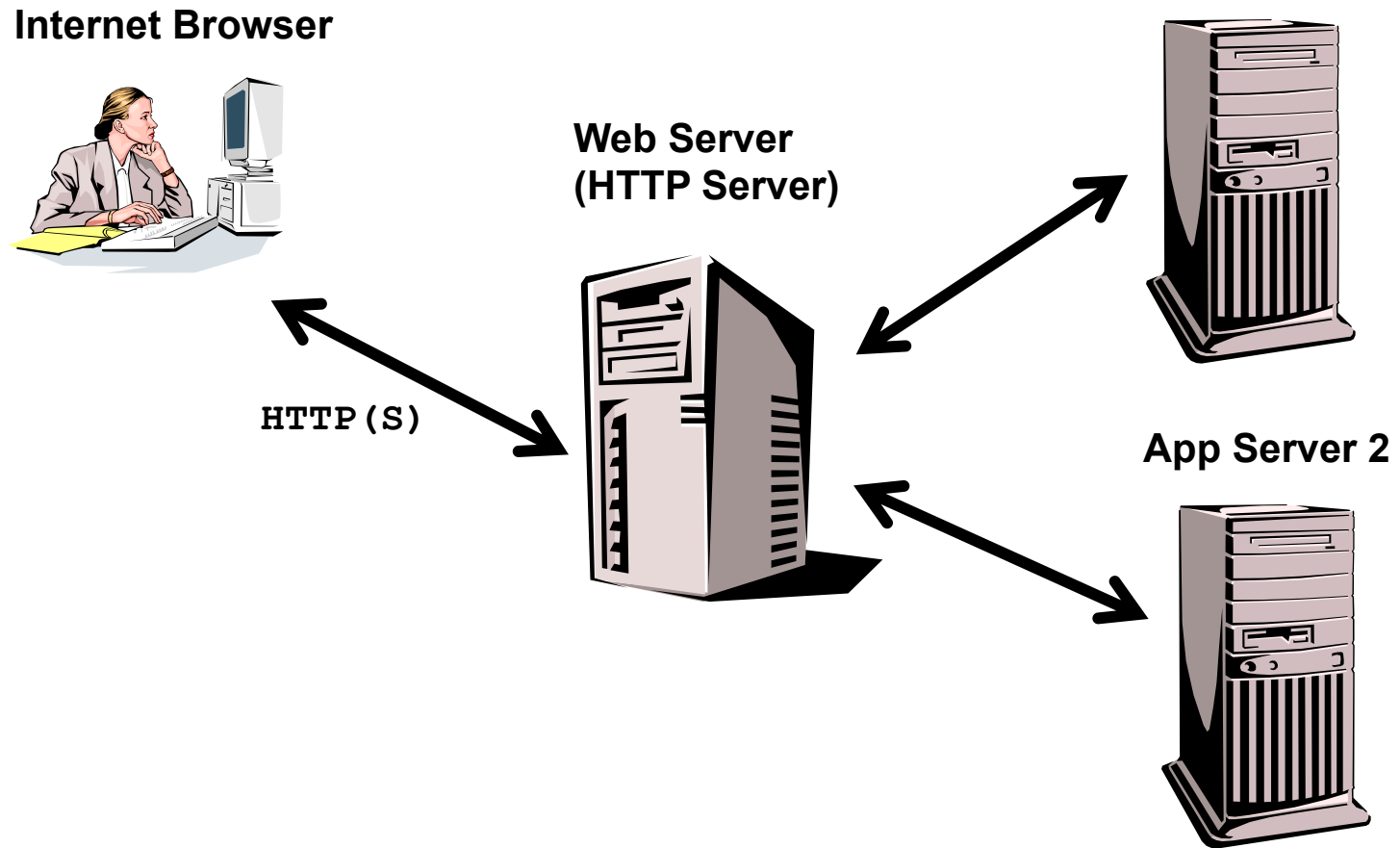
Application Server Products

- Application server products can be separated into 3 categories:
 - J2EE-based solutions
 - Non-J2EE solutions (PHP, ColdFusion, Perl, etc.)
 - And the Microsoft solution (ASP/COM and now .NET with ASP.NET, VB.NET, C#, etc.)

J2EE Application Servers

- Major J2EE products:
 - BEA WebLogic
 - IBM WebSphere
 - Sun iPlanet Application Server
 - Oracle 9iAS
 - HP/Bluestone Total-e-Server
 - Borland AppServer
 - Jboss (free open source)

Web Server and Application Server





What is J2EE (or JEE)

- Short for *Java 2 Platform Enterprise Edition*.
- 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.

What is Java 2 Enterprise Edition?

The platform provides an API and runtime environment for developing and running enterprise software, including network and web services, and other large-scale, multi-tiered, scalable, reliable, and secure network applications.

Current version is J2EE8

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

What & Who?

- 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.
- The Java EE platform is developed through the Java Community Process (JCP), which is responsible for all Java technologies.
- Expert groups, composed of interested parties, have created Java Specification Requests (JSRs) to define the various Java EE technologies.

J2EE Benefits

- High availability
- Scalability
- Integration with existing systems
- Freedom to choose vendors of application servers, tools, components
- Multi-platform



J2EE Benefits

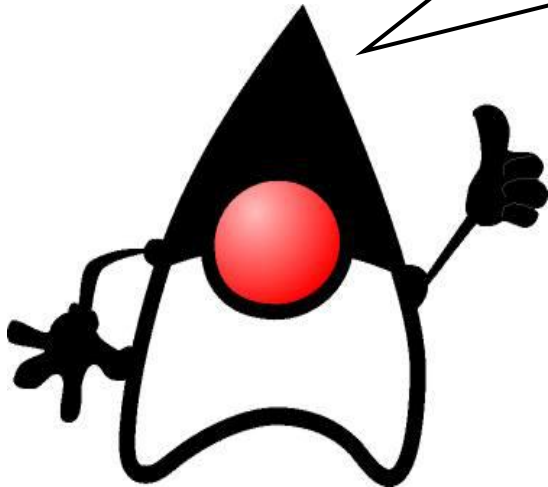
- Flexibility of scenarios and support to several types of clients
- Programming productivity:
 - Services allow developer to focus on business
 - Component development facilitates maintenance and reuse
 - Enables deploy-time behaviors
 - Supports division of labor

Development Roles

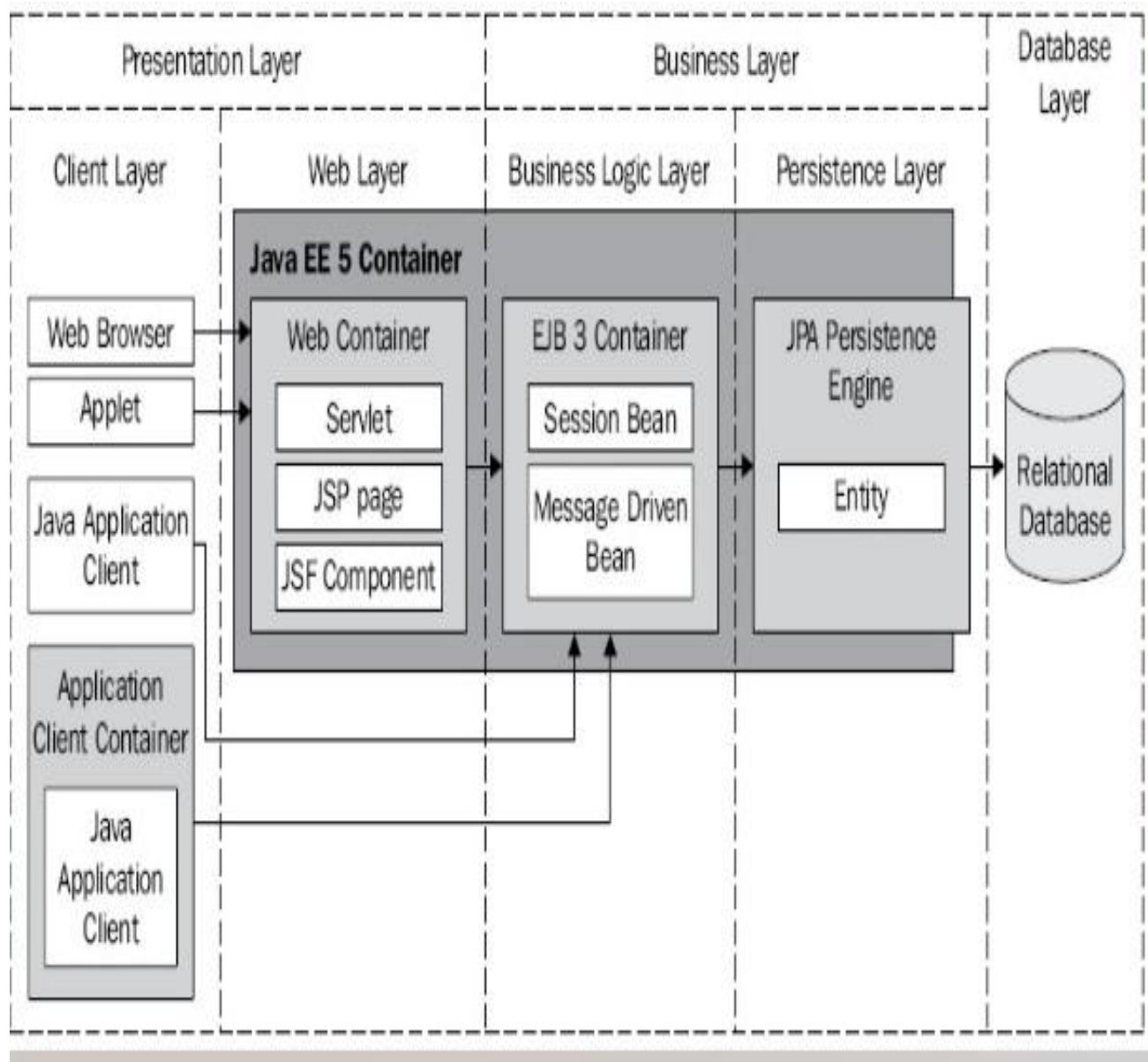
- **Java EE Product Provider:** Product providers are typically application server vendors that implement the Java EE platform according to the Java EE 6 Platform specification.
- **Tool Provider:** who creates development, assembly, and packaging tools used by component providers, assemblers, and deployers.
- **Application Component Provider:** who creates web components, enterprise beans, applets, or application clients for use in Java EE applications.
- **Enterprise Bean Developer:** develops and Packages the .class files and deployment descriptor into the EJB JAR file
- **Web Component Developer:** Develops and Packages the .class, .jsp, and.html files and deployment descriptor into the WAR file
- **Application Client Developer:** Develops and Delivers a JAR file containing the application client
- **Application Assembler:** who receives application modules from component providers and may assemble them into a Java EE application EAR file.
- **Application Deployer and Administrator:** configures and deploys application clients, web applications, Enterprise JavaBeans components, and Java EE applications, administers the computing and networking infrastructure where Java EE components and applications run, and oversees the runtime environment.

J2EE Benefits

Don't forget to say
that Java is cool!

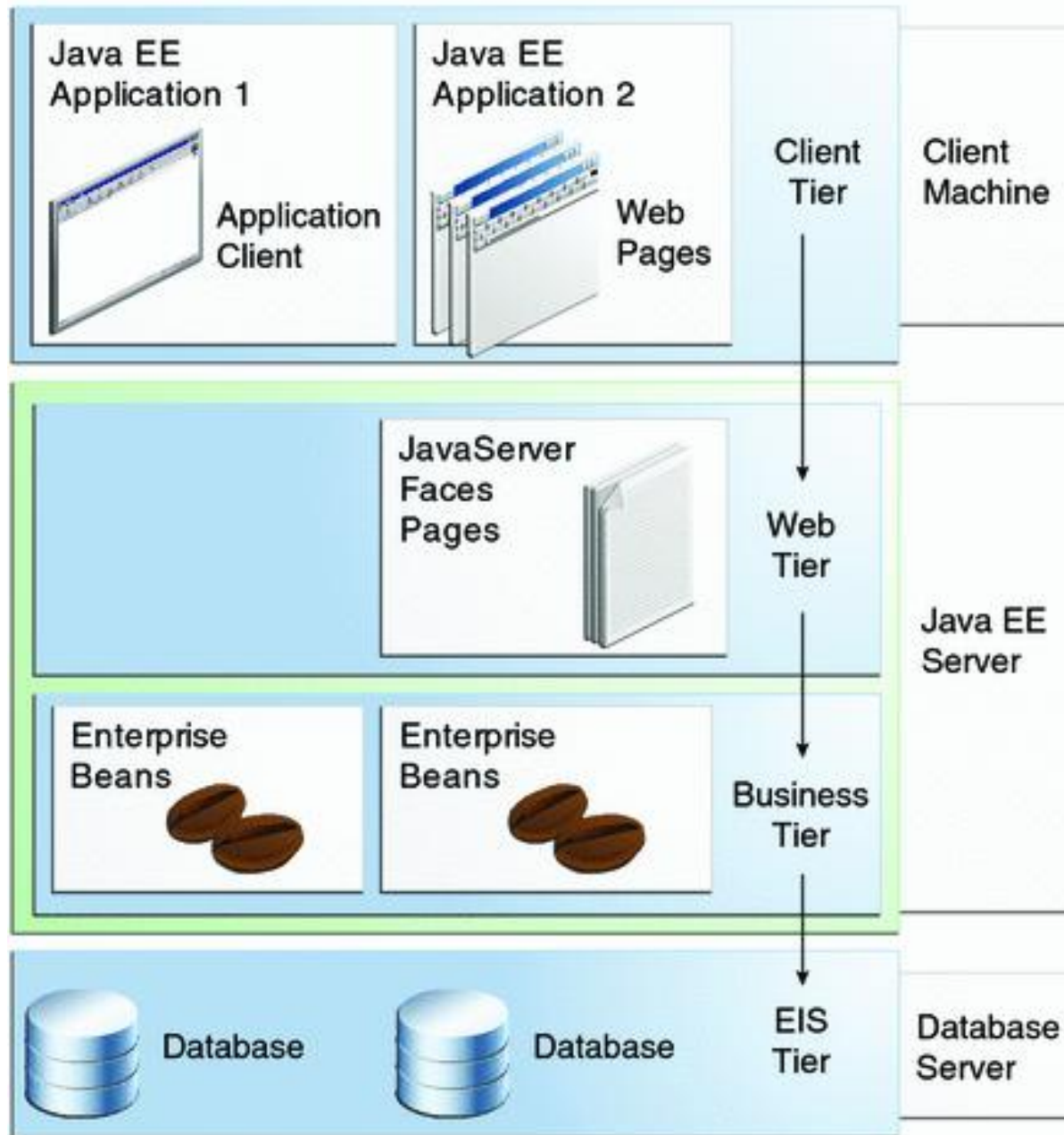


A glance at J2EE architecture



J2EE Application Model

- This model partitions the work needed to implement a multitier service into the following parts:
 - The business and presentation logic to be implemented by the developer
 - The standard system services provided by the Java EE platform



Multi-tiered Applications

J2EE 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 and that 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.
 - Enterprise JavaBeans (EJB) components (enterprise beans) are business components that run on the server.

Java EE Clients

- A Java EE client is usually either a web client or an application client.
- A **web client** consists of two parts:
 - Dynamic web pages containing various types of markup language (HTML, XML, and so on), which are generated by web components running in the web tier
 - A web browser, which renders the pages received from the server
- A web client is sometimes called a **thin client**. Thin clients usually do not query databases, execute complex business rules, or connect to legacy applications.
- When you use a thin client, such heavyweight operations are off-loaded to enterprise beans executing on the Java EE server, where they can leverage the security, speed, services, and reliability of Java EE server-side technologies.

J2EE Clients

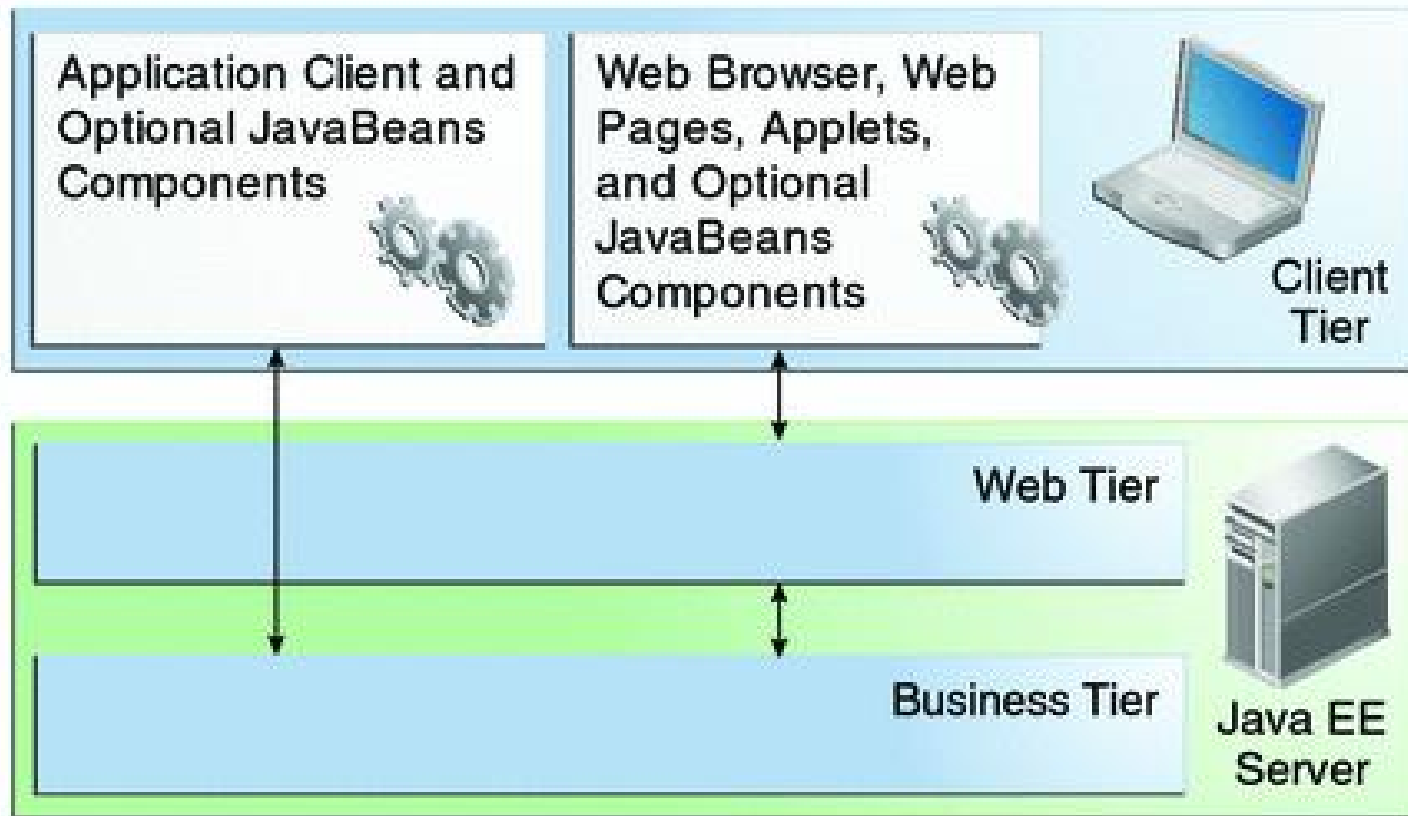
Application Clients:

- An **application client** runs on a client machine and provides a way for users to handle tasks that require a richer user interface than can be provided by a markup language.
- An application client typically has a graphical user interface (GUI) created from the Swing or the Abstract Window Toolkit (AWT) API, but a command-line interface is certainly possible.
- Application clients directly access enterprise beans running in the business tier.
- Application clients written in languages other than Java can interact with Java EE servers, enabling the Java EE platform to interoperate with legacy systems, clients, and non-Java languages.

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.

J2EE Server Communication



The client communicates with the business tier running on the Java EE server either directly or, as in the case of a client running in a browser, by going through web pages or servlets running in the web tier.

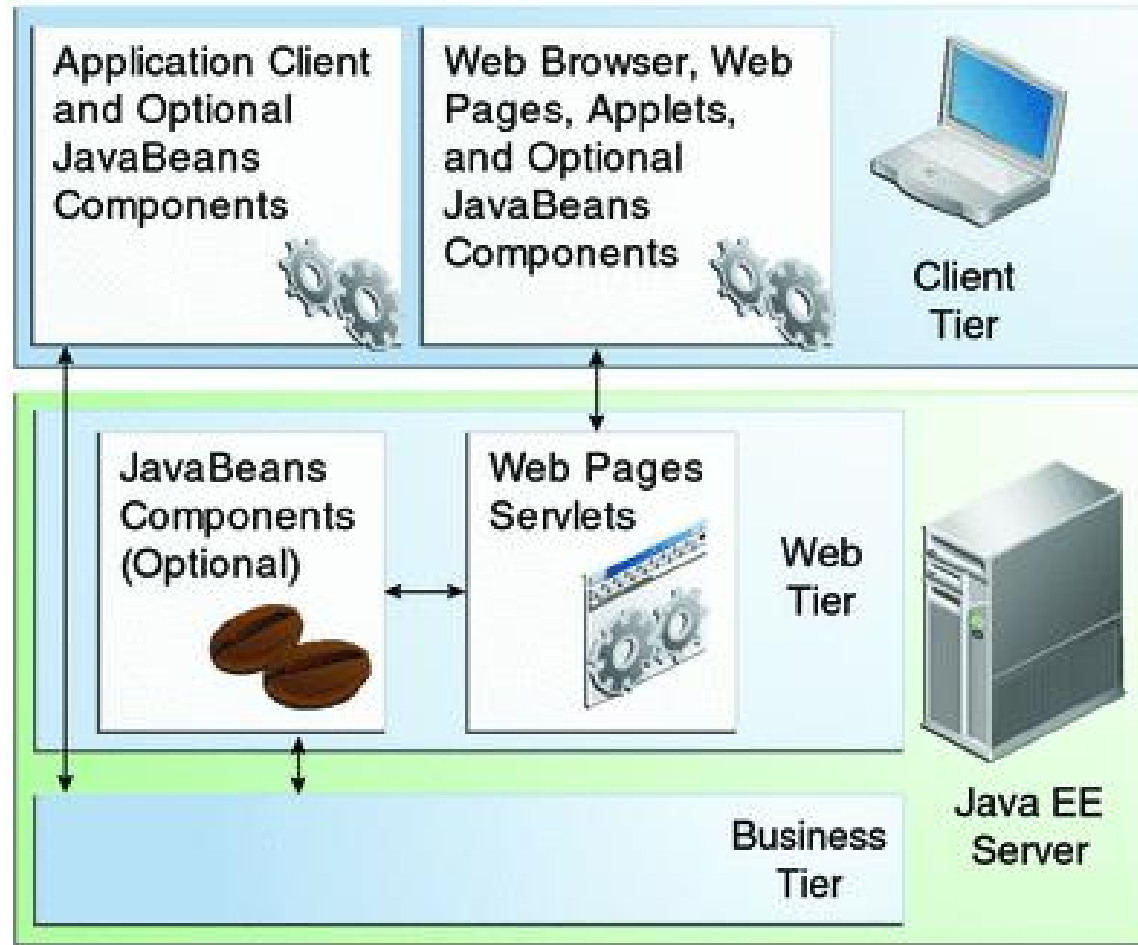
Web Components

- Java EE web components are either servlets or web pages created using JavaServer Faces technology and/or JSP technology (JSP pages).
 - **Servlets** are Java programming language classes that dynamically process requests and construct responses.
 - **JSP pages** are text-based documents that execute as servlets but allow a more natural approach to creating static content.
 - **JavaServer Faces technology** builds on servlets and JSP technology and provides a user interface component framework for web applications.

Web Components

- Static HTML pages and applets are bundled with web components during application assembly but are not considered web components by the Java EE specification.
- Server-side utility classes can also be bundled with web components and, like HTML pages, are not considered web components.

Web Tier and J2EE Applications



The web tier, like the client tier, might include a JavaBeans component to manage the user input and send that input to enterprise beans running in the business tier for processing.

JSP

- Used for web pages with dynamic content
- Processes HTTP requests (non-blocking call-and-return)
- Accepts HTML tags, special JSP tags, and scriptlets of Java code
- Separates static content from presentation logic
- Can be created by web designer using HTML tools

Servlet

- Used for web pages with dynamic content
- Processes HTTP requests (non-blocking call-and-return)
- Written in Java; uses print statements to render HTML
- Loaded into memory once and then called many times
- Provides APIs for session management

EJB

- EJBs are *distributed components* used to implement business logic (no UI)
- Developer concentrates on business logic
- Availability, scalability, security, interoperability and integrability handled by the J2EE server
- Client of EJBs can be JSPs, servlets, other EJBs and external applications
- Clients see *interfaces*

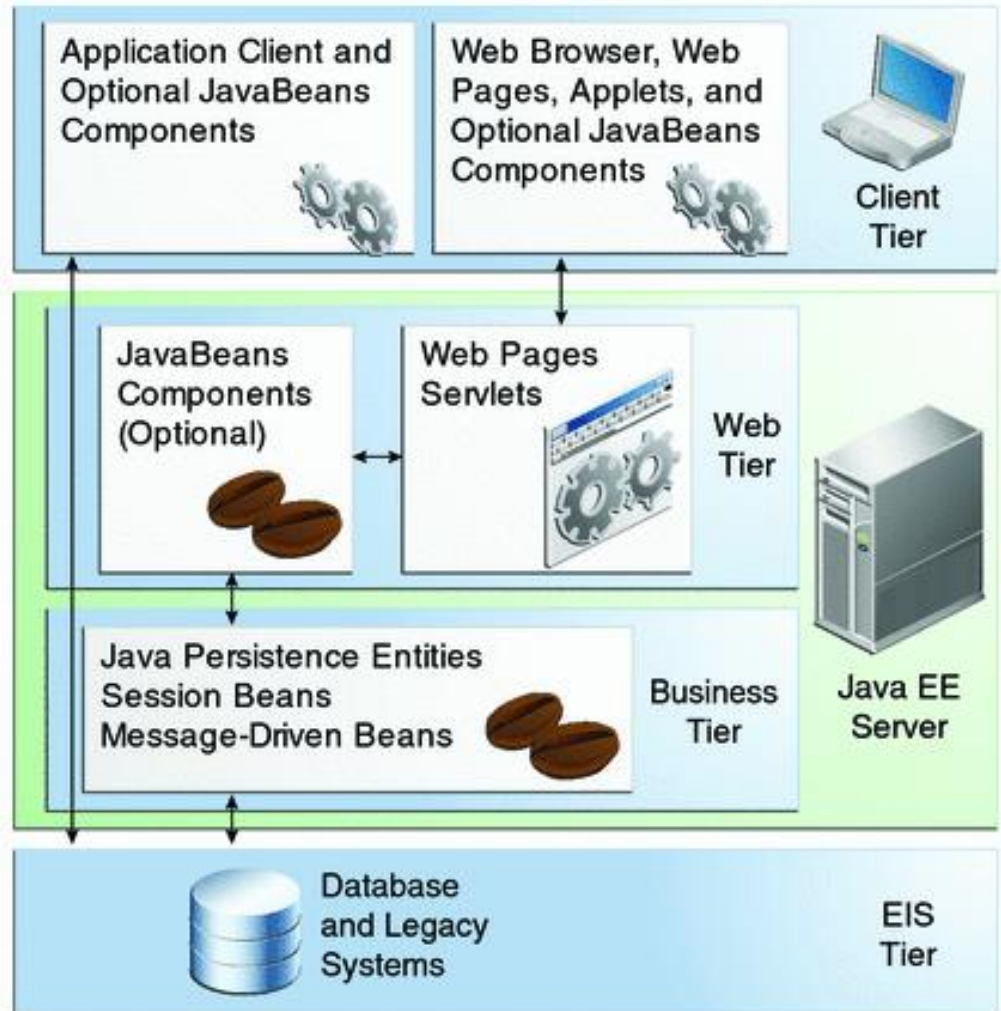
Business Components

Business code, which is logic that solves or meets the needs of a particular business domain such as banking, retail, or finance, is handled by enterprise beans running in either the business tier or the web tier.

Business and EIS tier

Enterprise bean receives data from client programs, processes it (if necessary), and sends it to the enterprise information system tier for storage.

An enterprise bean also retrieves data from storage, processes it (if necessary), and sends it back to the client program.



Enterprise Information System Tier

- The enterprise information system tier handles EIS software and includes enterprise infrastructure systems, such as enterprise resource planning (ERP), mainframe transaction processing, database systems, and other legacy information systems.
- For example, Java EE application components might need access to enterprise information systems for database connectivity

J2EE CONTAINERS

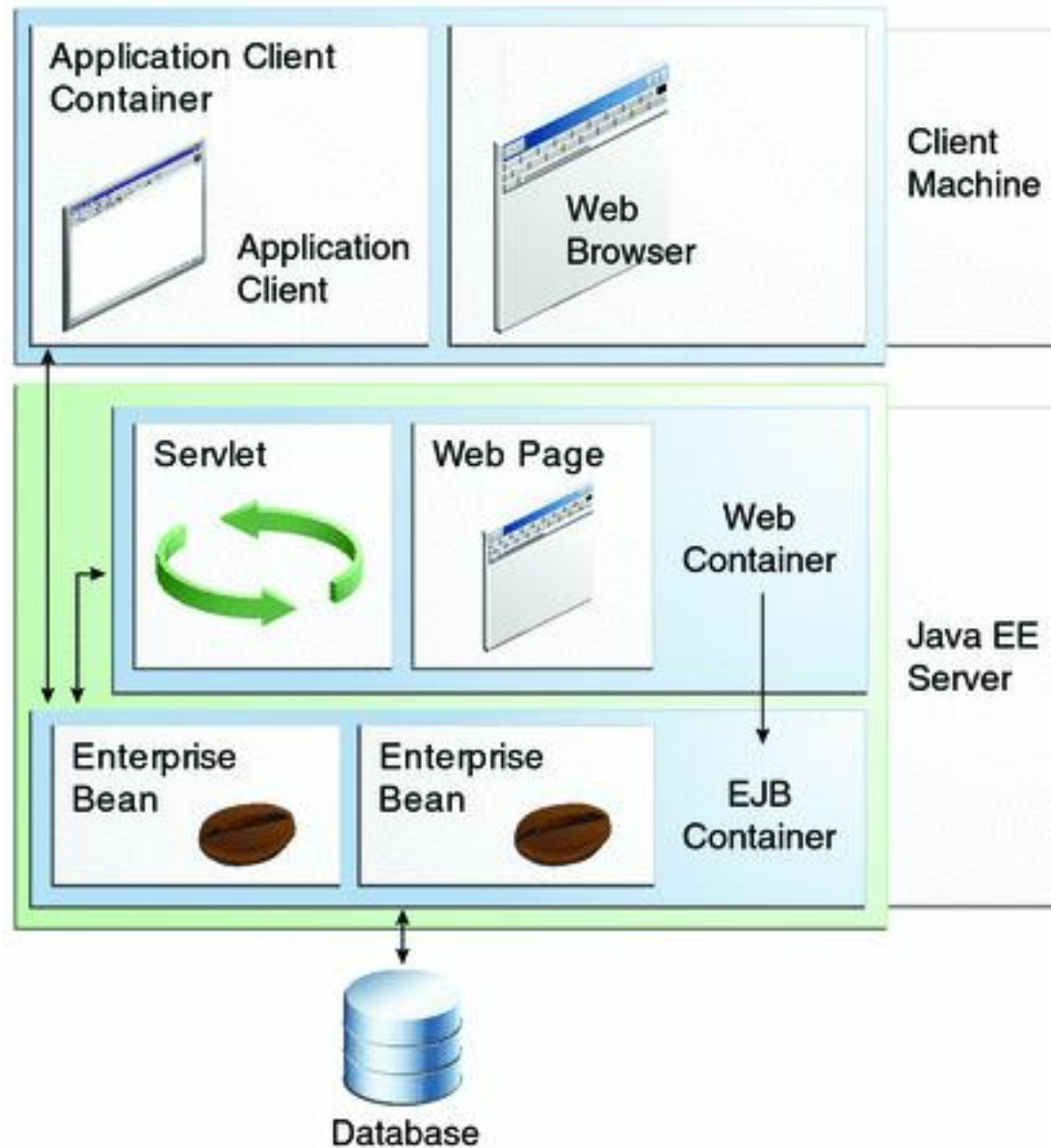
J2EE Containers

- Web components are supported by the services of a runtime platform called a **web container**.
- A web container provides such services as request dispatching, security, concurrency, and lifecycle management. A web container also gives web components access to such APIs as naming, transactions, and email.
- Normally, thin-client multi-tiered applications are hard to write because they involve many lines of intricate code to handle transaction and state management, multithreading, resource pooling, and other complex low-level details.
- The component-based and platform-independent Java EE architecture makes Java EE applications easy to write because business logic is organized into reusable components.

Container Services

- **Containers** are the interface between a component and the low-level platform-specific functionality that supports the component.
 - The Java EE security model lets you configure a web component or enterprise bean so that system resources are accessed only by authorized users.
 - The Java EE transaction model lets you specify relationships among methods that make up a single transaction so that all methods in one transaction are treated as a single unit.
 - JNDI lookup services provide a unified interface to multiple naming and directory services in the enterprise so that application components can access these services.
 - The Java EE remote connectivity model manages low-level communications between clients and enterprise beans.

Java EE Server and Containers



Java EE Servers and Containers

- **Java EE server:** The runtime portion of a Java EE product. A Java EE server provides EJB and web containers.
- **Enterprise JavaBeans (EJB) container:** Manages the execution of enterprise beans for Java EE applications. Enterprise beans and their container run on the Java EE server.
- **Web container:** Manages the execution of web pages, servlets, and some EJB components for Java EE applications. Web components and their container run on the Java EE server.
- **Application client container:** Manages the execution of application client components. Application clients and their container run on the client.
- **Applet container:** Manages the execution of applets. Consists of a web browser and Java Plug-in running on the client together.

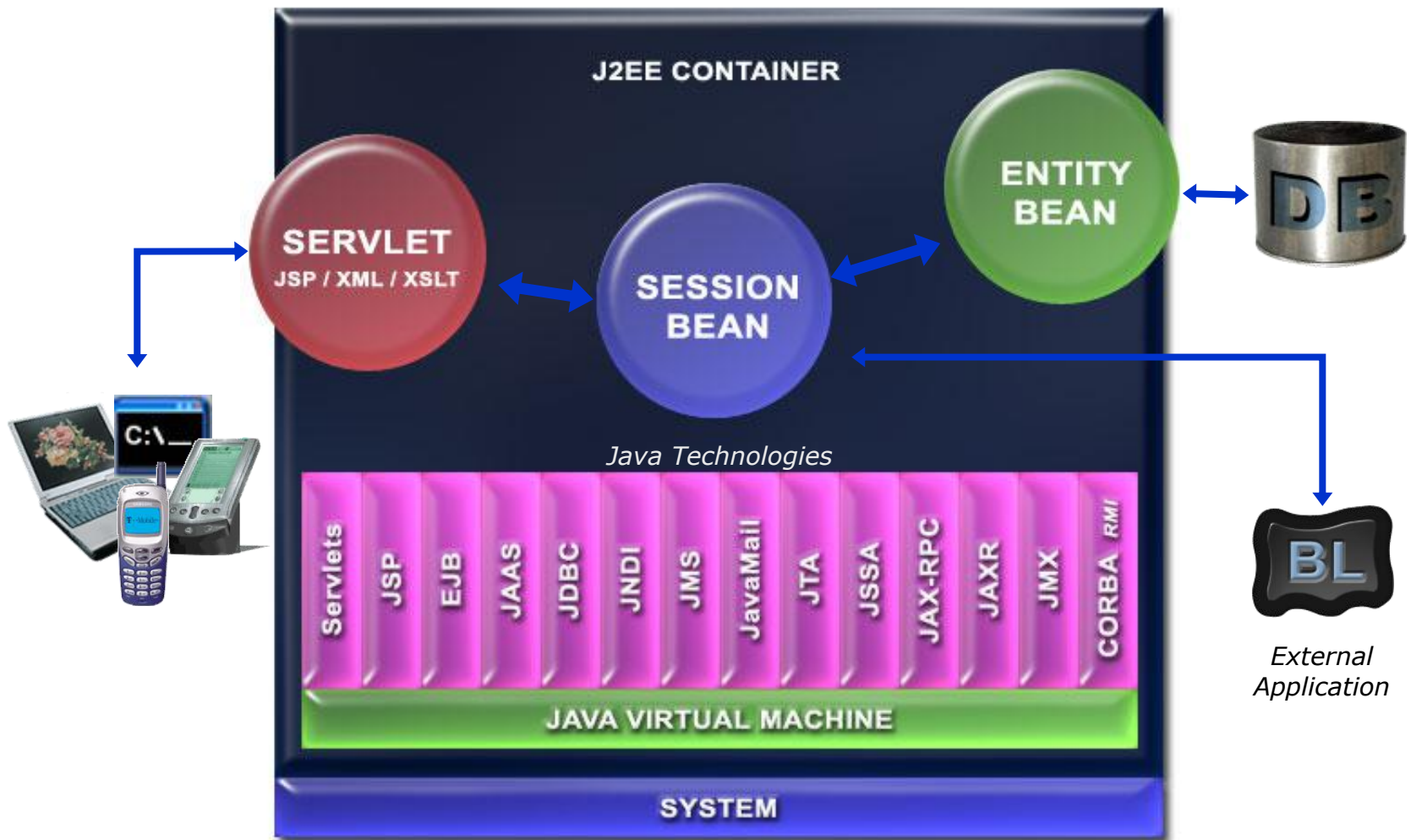
Java EE Application Assembly and Deployment

- A Java EE application is packaged into one or more standard units for deployment to any Java EE platform-compliant system. Each unit contains
 - A functional component or components, such as an enterprise bean, web page, servlet, or applet
 - An optional deployment descriptor that describes its content
- Once a Java EE unit has been produced, it is ready to be deployed. Deployment typically involves using a platform's deployment tool to specify location-specific information, such as a list of local users who can access it and the name of the local database.
- Once deployed on a local platform, the application is ready to run.

Packaging Applications

- A Java EE application is delivered in a Java Archive (JAR) file, a Web Archive (WAR) file, or an Enterprise Archive (EAR) file.
- A WAR or EAR file is a standard JAR (.jar) file with a .war or .ear extension.
- Using JAR, WAR, and EAR files and modules makes it possible to assemble a number of different Java EE applications using some of the same components.
- No extra coding is needed; it is only a matter of assembling (or packaging) various Java EE modules into Java EE JAR, WAR, or EAR files.

The J2EE Framework



WEB APPLICATIONS

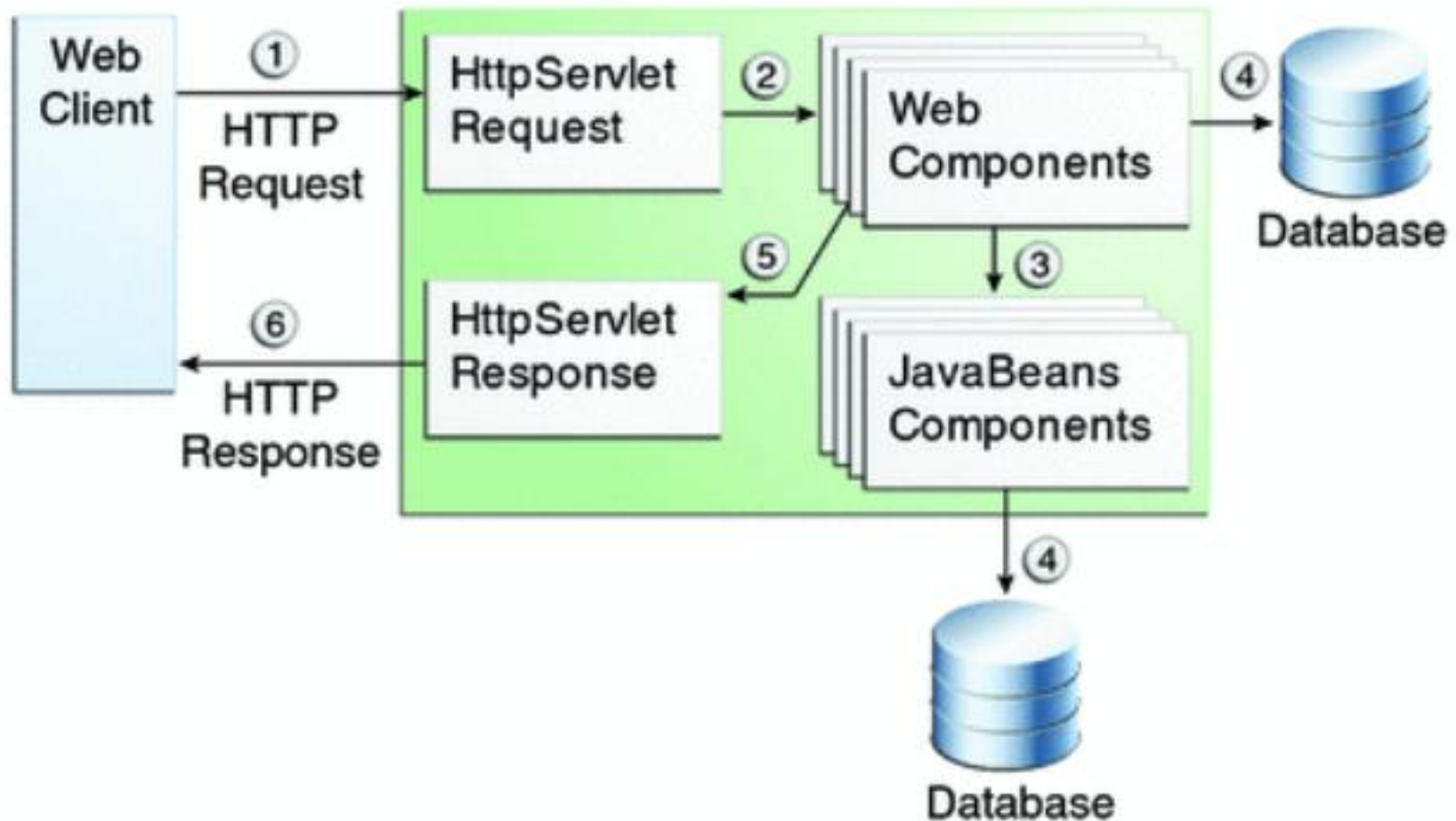
Web Applications

- A **web application** is a dynamic extension of a web or application server.

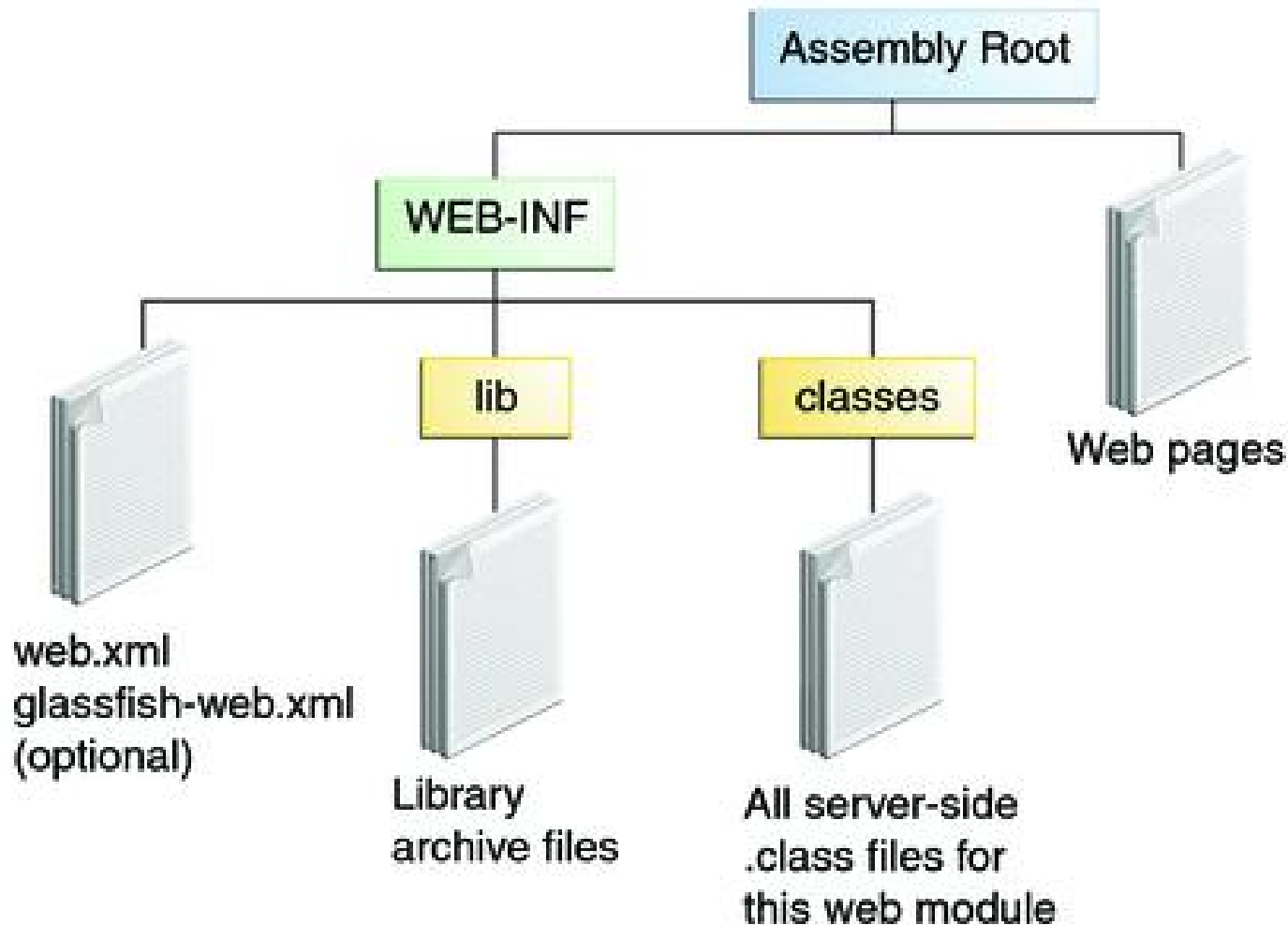
Web applications are of the following types:

- **Presentation-oriented:** A **presentation-oriented web application** generates interactive web pages containing various types of markup language (HTML, XHTML, XML, and so on) and dynamic content in response to requests.
- **Service-oriented:** A **service-oriented web application** implements the endpoint of a web service. Presentation-oriented applications are often clients of service-oriented web applications.

Java Web Application Request Handling



Web Module Structure



Web Services Support

- Web services are web-based enterprise applications that use open, XML-based standards and transport protocols to exchange data with calling clients.
- The J2EE platform provides the XML APIs and tools you need to quickly design, develop, test, and deploy web services and clients that fully interoperate with other web services and clients running on Java-based or non-Java-based platforms.

XML

- XML is a cross-platform, extensible, text-based standard for representing data.
- When XML data is exchanged between parties, the parties are free to create their own tags to describe the data, set up schemas to specify which tags can be used in a particular kind of XML document, and use XML stylesheets to manage the display and handling of the data.

SOAP Transport Protocol

- Client requests and web service responses are transmitted as Simple Object Access Protocol (SOAP) messages over HTTP to enable a completely interoperable exchange between clients and web services, all running on different platforms and at various locations on the Internet.
- HTTP is a familiar request-and response standard for sending messages over the Internet, and SOAP is an XML-based protocol that follows the HTTP request-and-response model.

WSDL & UDDI

- **WSDL Standard Format :** The Web Services Description Language (WSDL) is a standardized XML format for describing network services. The description includes the name of the service, the location of the service, and ways to communicate with the service.
- **UDDI and ebXML Standard Formats:** Universal Description, Discovery and Integration (UDDI) and ebXML, make it possible for businesses to publish information on the Internet about their products and web services, where the information can be readily and globally accessed by clients who want to do business.

