CMP 436/774

Introduction to Java Enterprise Edition

Fall 2013
Department of Mathematics
and Computer Science
Lehman College, CUNY

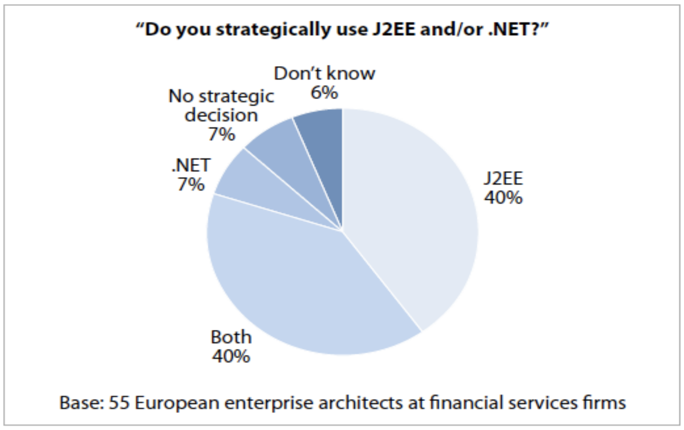
Java Enterprise Edition

- **Developers today increasingly recognize the need for distributed, transactional, and portable applications that leverage the speed, security, and reliability of server-side technology.**
  - Enterprise applications provide the business logic for an enterprise. They are centrally managed and often interact with other enterprise software.
  - In the world of information technology, enterprise applications must be designed, built, and produced for less money, with greater speed, and with fewer resources.
- **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.**

Java Enterprise Edition (cont'd)

- The Java Platform, Enterprise Edition (Java EE) builds upon the Java SE platform and provides a set of technologies for developing and running portable, robust, scalable, reliable and secure server-side applications.
- EE technologies are loosely divided into two categories:
  - Web application technologies
  - Enterprise application technologies
- Depending on your needs, you may want to use certain technologies from either category. For example, makes use of Servlet, JSP/EL, and JSTL "web" technologies (or JSF), as well as EJB and JPA "enterprise" technologies.

Java Enterprise Edition (cont'd)



Java EE Components

- Applets: GUI app's executed in a web browser. They use the Swing API to provide powerful user interfaces.
- Applications: programs executed on a client. Typically GUIs or batch-processing programs that have access to all the facilities of the Java EE middle tier.
- Web applications: applications executed in a web container and respond to HTTP requests from web clients.
  - Made of servlets, servlet filters, JSP pages, and JSF.
  - Servlets also support web service endpoints
- Enterprise Java Beans: container-managed components for processing transactional business logic. They can be accessed locally and remotely through RMI (or HTTP for SOAP and RESTful web services).

5

Java EE Application

- In a Java EE application:
  - The model -- business layer functionality represented by JavaBeans or EJBs
  - The view -- the presentation layer functionality represented by JSPs or JSFs in a web application
  - The controller -- Servlet mediating between model and view
- Must accommodate input from various clients including HTTP requests from web clients, and…
  - WML from wireless clients
  - XML documents from suppliers
  - etc.
- **➔ MVC Architecture (or termed Design Pattern)**

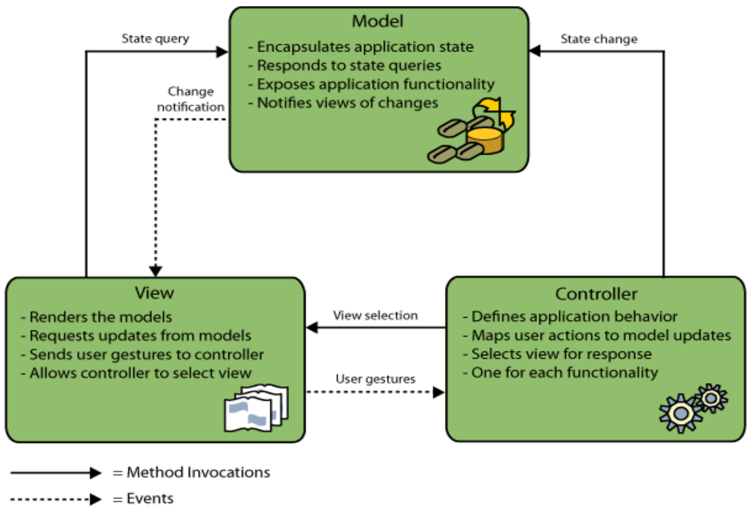
6

MVC Architecture

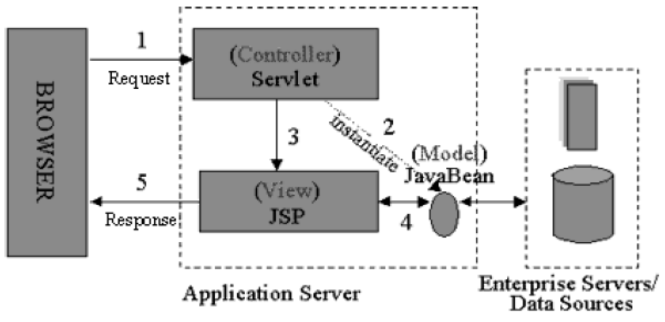
- **Model:**
  - Represents the business data and any business logic that govern access to and modification of the data. The model notifies views when it changes and lets the view query the model about its state. It also lets the controller access application functionality encapsulated by the model.
- **View:**
  - The view renders the contents of a model. It gets data from the model and specifies how that data should be presented. It updates data presentation when the model changes. A view also forwards user input to a controller.
- **Controller:**
  - The controller defines application behavior. It dispatches user requests and selects views for presentation. It interprets user inputs and maps them into actions to be performed by the model. In a web application, user inputs are HTTP GET and POST requests. A controller selects the next view to display based on the user interactions and the outcome of the model operations.

Source: Java BluePrints - J2EE Patterns, MVC
http://java.sun.com/blueprints/patterns/MVC-detailed.html

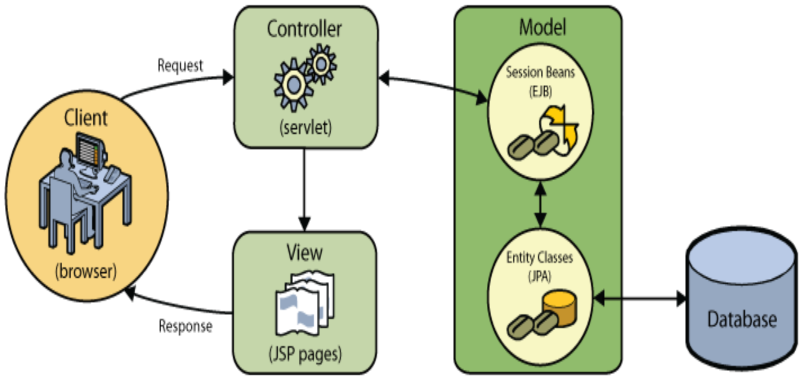
MVC Architecture (cont'd)


Model
- Encapsulates application state
- Responds to state queries
- Exposes application functionality
- Notifies views of changes
State query
State change
Change notification
View
- Renders the models
- Requests updates from models
- Sends user gestures to controller
- Allows controller to select view
View selection
User gestures
Controller
- Defines application behavior
- Maps user actions to model updates
- Selects view for response
- One for each functionality
= Method Invocations
= Events

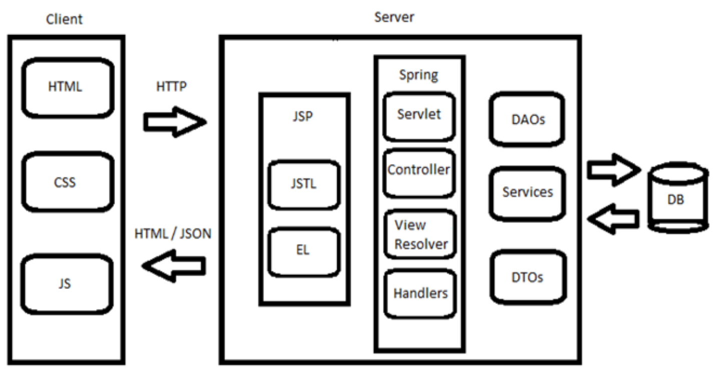
MVC- an example (based on JDBC)



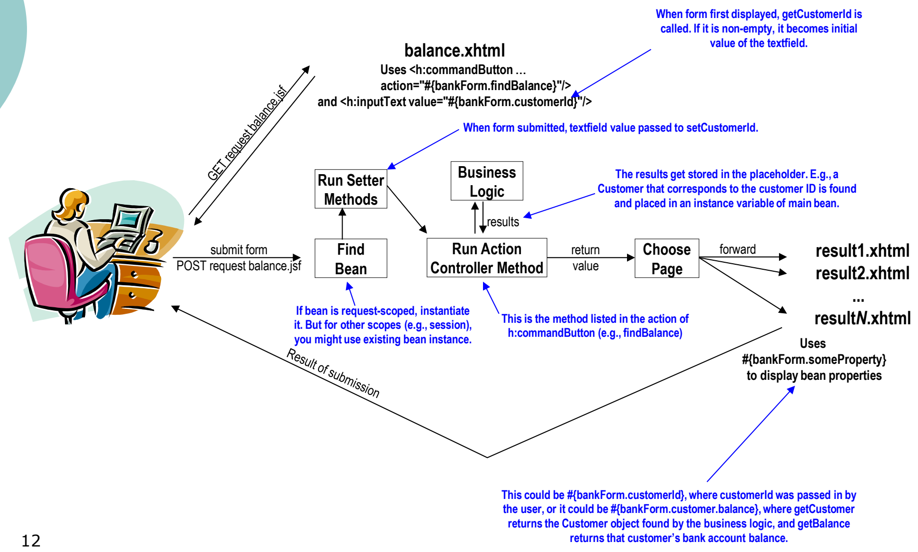
MVC- an example (based on JPA)



MVC- an example (Spring framework)



JSF Flow of Control



Uses <h:commandButton ...
action="#{bankForm.findBalance}"/>
and <h:inputText value="#{bankForm.customerId}"/>

When form first displayed, getCustomerId is called. If it is non-empty, it becomes initial value of the textfield.

When form submitted, textfield value passed to setCustomerId.

The results get stored in the placeholder. E.g., a Customer that corresponds to the customer ID is found and placed in an instance variable of main bean.

If bean is request-scoped, instantiate it. But for other scopes (e.g., session), you might use existing bean instance.

This is the method listed in the action of h:commandButton (e.g., findBalance)

Uses #{bankForm.someProperty} to display bean properties

This could be #{bankForm.customerId}, where customerId was passed in by the user, or it could be #{bankForm.customer.balance}, where getCustomer returns the Customer object found by the business logic, and getBalance returns that customer's bank account balance.

Web Application

- A web application is a dynamic extension of a web or application server. Types of web applications:
  - Presentation-oriented
    - 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.
- In Java EE platform, web components provide the dynamic extension capabilities for a web server.
  - Web components are either Java servlets, web pages, web service endpoints, JSP pages, or JSFs

13

View layer in Web Application

- Display information according to client types
- Display result of business logic (Model)
- Not concerned with how the information was obtained, or from where (since that is the responsibility of Model)

14

Model layer in Web Application

- Models the data and behavior behind the business process
- What it’s responsible for:
  - Performing DB queries
  - Calculating the business process
  - Processing orders
- Encapsulation of data and behavior which are independent of presentation

15

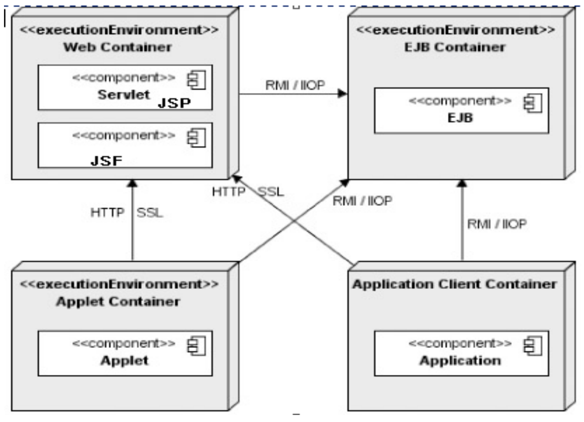
Controller in Web Application

- Serves as the logical connection between the user's interaction and the business services on the back end servers
- Responsible for making decisions among multiple presentations
  - e.g. User's language, locale or access level dictates a different presentation.
- A request enters the application through the control layer, which will decide how the request should be handled and what information should be returned

16

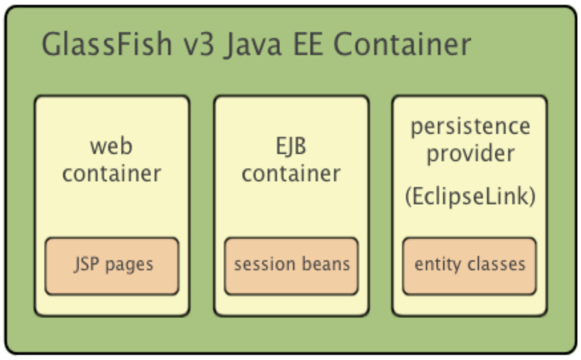
Containers

- Java EE is a set of specifications implemented by different containers.
- Containers are Java EE runtime environments that provide certain services to the components they host such as lifecycle management, dependency injection, security, etc.



<<executionEnvironment>> Web Container
<<component>> Servlet JSP
<<component>> JSF
<<executionEnvironment>> EJB Container
<<component>> EJB
RMI / IIOP
HTTP SSL
RMI / IIOP
HTTP SSL
RMI / IIOP
<<executionEnvironment>> Applet Container
<<component>> Applet
Application Client Container
<<component>> Application

GlassFish AS Container



GlassFish v3 Java EE Container
web container
JSP pages
EJB container
session beans
persistence provider (EclipseLink)
entity classes

JSP, Servlets, and JSF

- **JavaServer Pages (JSP) technology allows you to easily create web content that has both static and dynamic components.**
  - JSP technology makes available all the dynamic capabilities of Java Servlet technology but provides a more natural approach to creating static content.
- Servlets are Java classes that dynamically process requestsand construct responses.
  - Servlets are best suited for service-oriented applications (web service endpoints are implemented as servlets) and the control functions of a presentation-oriented application, such as dispatching requests and handling non-textual data.
- JavaServer Faces (JSF) and Facelets are used for building interactive web applications.
  - Java Server Faces and Facelets pages are more appropriate for generating text-based markup, such as XHTML, and are generally used for presentation–oriented applications.

19

JAVA EE Services and APIs

- JPA: Standard API for object-relational mapping (ORM).
- JMS: allows components to communicate asynchronously through messages.
- Java Naming and Directory Interface (JNDI): used to access naming and directory systems.
- JTA: a transaction API
- JAX-WS (SOAP based), JAX-RS (RESTful HTTP based)

20

JPA

- Objects vs. Entities
  - Objects are instances that just live in memory.
  - Entities are objects that live shortly in memory and persistently in a database.
- JPA maps objects to a database via metadata that can be supplied using annotations or in an XML descriptor
- Annotations: The code of the entity is directly annotated with all sorts of annotations described in the javax.persistence package.
- Entity example in the next page

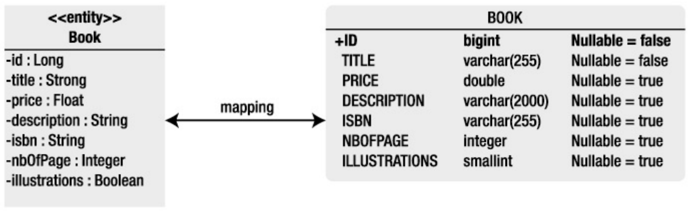
```
@Entity
@Table(name = "category")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Category.findAll", query = "SELECT c FROM Category c"),
    @NamedQuery(name = "Category.findById", query = "SELECT c FROM Category c WHERE c.id = :id"),
    @NamedQuery(name = "Category.findByName", query = "SELECT c FROM Category c WHERE c.name =
:name")})
public class Category implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Basic(optional = false)
    @Column(name = "id")
    private Short id;
    @Basic(optional = false)
    @NotNull
    @Size(min = 1, max = 45)
    @Column(name = "name")
    private String name;
    @OneToMany(cascade = CascadeType.ALL, mappedBy = "categoryId")
    private Collection<Product> productCollection;

    public Category() {
    }

    public Category(Short id) {
        this.id = id;
    }
```

JPA cont’d

- Relational model (i.e. RDBMS) vs. Object Oriented model (i.e. Java)
- Object-Relational Mapping (ORM)
- Java Persistence API (JPA)
  - An API on top of JDBC
  - Can access and manipulate relational data from Enterprise Java Beans (EJBs), web components, and Java SE applications
  - Includes an entity manager API to perform DB-related operations like CRUD
  - Includes JPQL, an object-oriented query language


<<entity>>
Book
-id : Long
-title : Strong
-price : Float
-description : String
-isbn : String
-nbOfPage : Integer
-illustrations : Boolean
mapping
BOOK
+ID bigint Nullable = false
TITLE varchar(255) Nullable = false
PRICE double Nullable = true
DESCRIPTION varchar(2000) Nullable = true
ISBN varchar(255) Nullable = true
NBOFPAGE integer Nullable = true
ILLUSTRATIONS smallint Nullable = true

EJB

- Server-side components
- Encapsulate business logic
- Take care of transactions and security
- Used in building business layers to sit on top of the persistence layer and as an entry point for presentation-tier technologies such as JSP, JSF
- Can be built by annotating a POJO that will be deployed into a container

Type of EJBs

- Session beans and Message-driven Beans (MDBs)
- Session Beans are used to encapsulate high-level business logic and can be
  - Stateful: the state of the bean is maintained across multiple method calls. The "state" refers to the values of its instance variables. Because the client interacts with the bean, this state is often called the *conversational* state. Stateful session bean contains conversational state, which must be retained across method invocations for a single user
  - Stateless: contains no conversational state between invocations, and any instance can be used for any client
  - Singleton: A single session bean is shared between clients and supports concurrent access