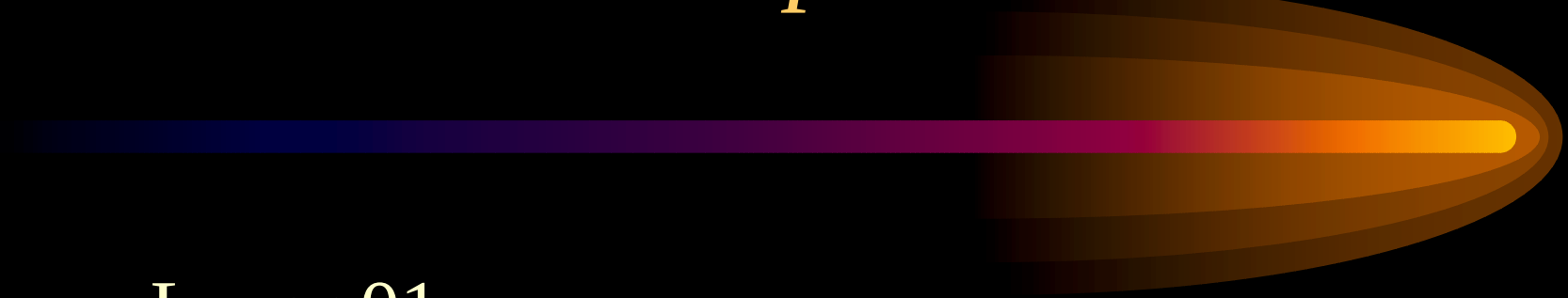
Enterprise JavaBeans



Layer:01

Overview

Agenda

- Course introduction & overview.
- Hardware & software configuration.
- Evolution of enterprise technology.
- J2EE framework & components.
- EJB framework & components.

Course Overview

Introduction

- During this course we'll explore Sun Microsystems' *Enterprise JavaBeans* component object model.
- Because of its complexity, this course can't provide complete coverage over all aspects of the EJB framework.
- This course will serve as a foundation upon which later knowledge can be based.

Prerequisites

- Because this course involves enterprise Java, I assume the following knowledge:
  - Servlet, JSP, and JDBC development
  - Ability to perform installation, configuration, administration, and troubleshooting tasks.
- If you haven't met the official prerequisites for this course, I strongly suggest you drop.

Required Texts

- *Enterprise JavaBeans, 2nd Edition*. Richard Monson-Haefel. O'Reilly & Associates. 2000.
- *Enterprise JavaBeans 1.1 specification*.

Grading

- Your final grade will be calculated using the following weights:

Homework	*30%*
Midterm	*30%*
Final	*30%*
Participation	*10%*

Hardware & Software

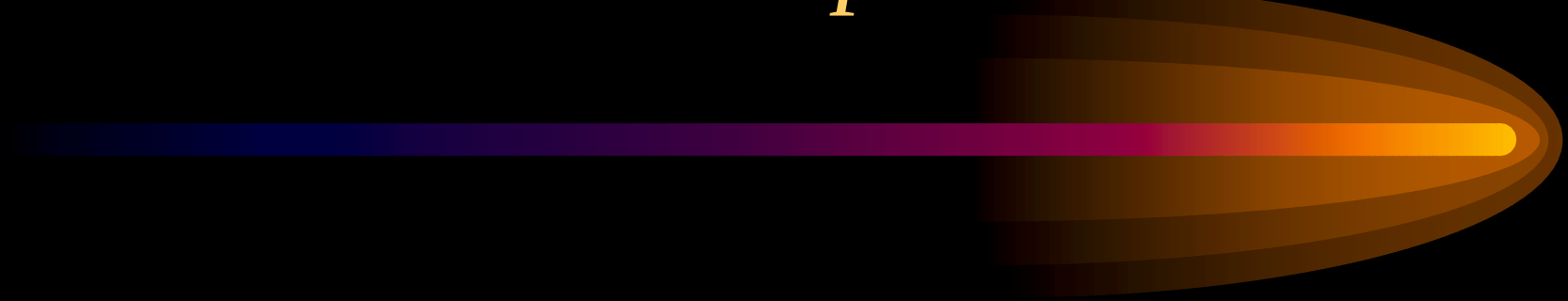
Operating System

- For this course I strongly recommend the following operating systems:
  - Windows NT 4.0
  - Windows 2000
  - Linux
- If you're running Windows 95/98/ME, I won't guarantee or support the behavior of the EJB server.

Required Software

- The following is the required software for this course:
  - Jakarta-Tomcat 3.2 or later
  - JBoss 2.2.2
  - Cloudscape 3.6.4
  - Ant 1.3
  - JUnit 3.7
- This list is non-negotiable. I'll accept no substitutions or alterations.

Enterprise Evolution



Goals

- Modern companies have specific goals for the enterprise information systems:
  - Time-to-market
  - Portability
  - Interoperability
  - Lower cost

Architectures

- Architectures have undergone three (3) key evolutions:
  - 1-tier: This is the traditional mainframe architecture.
  - 2-tier: This is the client-server model.
  - N-tier: This is the current model whereby we distribute our software over a set of machines all of which comprise a part of the application.

Issues

- Each architecture has specific issues in the areas of:
  - Maintainability
  - Manageability
  - Performance
  - Availability

J2EE Framework

What is J2EE? 1 of 3

- The Java 2 Enterprise Edition (J2EE) is a Java-based service-oriented framework.
- The goal is to allow developers to focus on solving business problems, rather than on developing system services.
- This provides for separation of business logic from system services.

What is J2EE? 2 of 3

- The J2EE technology components can be arranged by the service it provides:

Service	J2EE Technology
Web	Servlets, JSP
Database	JDBC
Naming & Directory	JNDI
Messaging	JMS
Email	JavaMail, JAF
Distributed Objects	JavaIDL, RMI, RMI-IIOP
Transactions	JTA

What is J2EE? 3 of 3

- J2EE consists of four (4) main elements:
  - Specification
  - Reference implementation
  - Compatibility test suite
  - Blueprints

Specification

- The J2EE specification outlines what services *must* be provided by a vendor that wishes to be J2EE compliant.
- Specifications typically outline *minimum* requirements.
- The specification often defines the API to be provided to a developer.

Reference Implementation

- The J2EE reference implementation is a fully functional J2EE server that provides web and EJB containers.
- It's used to demonstrate that a product can be built that implements the specification as written.
- *It isn't meant to be production-grade software*.

Compatibility Test Suite

- The J2EE compatibility test suite is used to evaluate a vendor product to ensure that it meets the requirements of J2EE as defined by the specification.
- Any vendor that wishes to market its product as "J2EE compliant" must submit that product for testing.

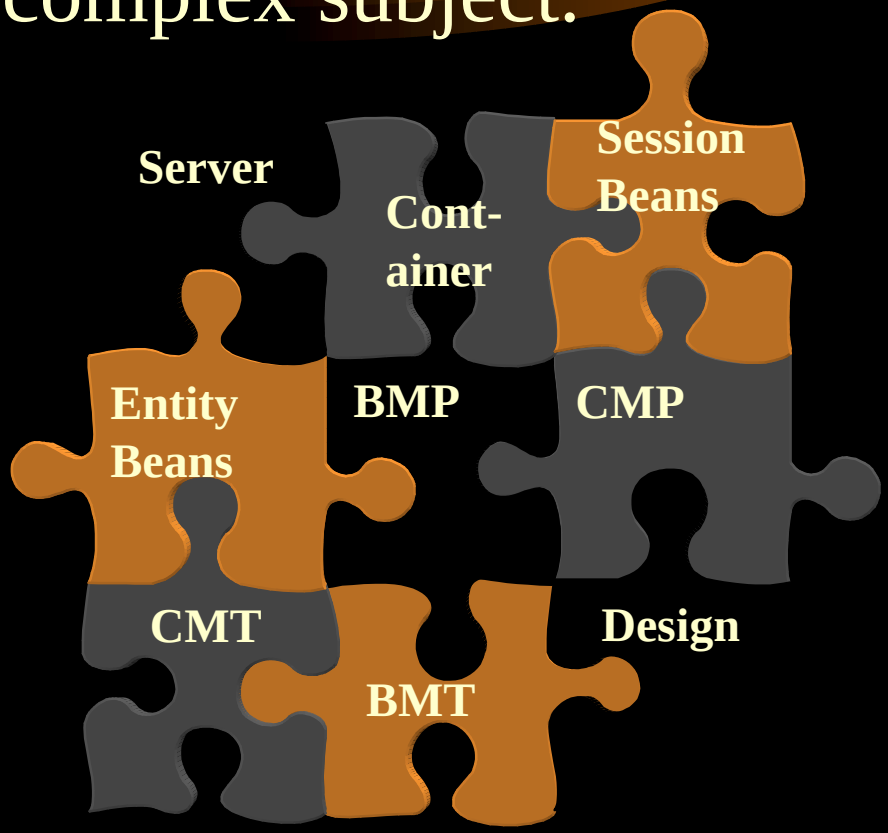
Blueprints

- The J2EE blueprints are an effort by the Sun Java Development Center to document the best practices involving the use of J2EE components.
- In many ways the blueprints are similar to the design patterns you studied in SE455.

EJB Framework

Overview

- EJB technology is a complex subject.
- While these aren't the only topics of interest within the EJB framework, they're the most critical.



What are Enterprise JavaBeans?

- *Enterprise JavaBeans* (EJB) is an architecture for component-based distributed computing.
- Typically EJBs are elements of business processing that could be shared across applications as well as across a network.

EJB Components

- There are four (4) main components to EJB technology:
  - EJB server
  - EJB container
  - EJB
  - Deployment descriptor

EJB Server

- The EJB server provides common services available to all EJBs.
- The server's job is to hide the complexity of these services from the applications that require them.

Services

- The EJB specification outlines six(6) services that must be provided by each server:
  - Naming
  - Transaction
  - Security
  - Persistence
  - Concurrency
  - Lifecycle

EJB Container

- The EJB container integrates with the EJB server and acts as an intermediary between the server and EJBs.
- It provides services such as:
  - EJB instance lifecycle management
  - EJB instance identification

EJB Roles

- The EJB specification outlines six (6) key roles required for EJB development:
  - Bean provider
  - Application assembler
  - Deployer
  - Server provider
  - Container provider
  - System administrator

Bean Provider

- The *bean provider* is a developer.
- They're responsible for translating business requirements into physical code within an EJB.
- The final product of the bean provider is an EJB-JAR file containing the appropriate classes as well as the structural content of the deployment descriptor.

Application Assembler

- The *application assembler* is often a team lead or senior developer.
- They're responsible for packaging all of the EJB-JAR files generated by the bean providers into a complete application.
- Their final product is an EAR file containing the appropriate EJB-JAR files.

Deployer

- The *deployer* is usually a senior developer or architect.
- They're responsible for installing the application into the target runtime environment.
- They've got to be familiar with all aspects of the environment including security and transaction support.

Server Provider

- The *server provider* is usually an external vendor.
- Most clients don't have the time, money, or interest to develop their own EJB servers.
- The job of the server provider is to produce an EJB server that's compliant with the EJB specifications.

Container Provider

- The *container provider* is usually an external vendor and is often the same as the server vendor.
- Most clients don't have the time, money, or interest to develop their own EJB containers.
- The container provider produces an EJB container that's compliant with the EJB specifications.

System Administrator

- The *system administrator* is generally a member of the organization into which an application is being installed.
- Their task to ensure that the runtime environment is configured in such a way that the application can function correctly and integrate with all of the required external components.

Summary

- Course introduction & overview.
- Hardware & software configuration.
- Evolution of enterprise technology.
- J2EE framework & components.
- EJB framework & components.

Next Steps

- Make sure that you install, configure, and test the software required for this class.
- Read chapters 1-2 in your text.