


UW  BUSINESS SCHOOL

J2EE Overview

MSIS 531 – Spring 2006


Learning Goals

- Introduction to J2EE
- Discuss the key technologies within J2EE that we'll consider:
 - Servlets
 - JSP/JSF
 - EJB
 - JDBC
- Brief mention of related technologies
- J2EE applications and packaging

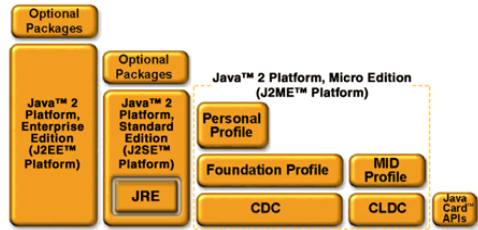
© UW Business School, University of Washington 2006 

The Java 2 Platform

- Platform introduced June, 1999
- J2SE – Java 2 Standard Edition
 - Java for the desktop / workstation
 - <http://java.sun.com/j2se>
- J2ME – Java 2 Micro Edition
 - Java for the consumer device
 - <http://java.sun.com/j2me>
- J2EE - Java 2 Enterprise Edition
 - Java for the server
 - <http://java.sun.com/j2ee>

© UW Business School, University of Washington 2006 

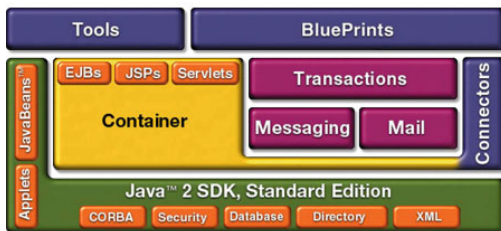
The Java 2 Platform



J2EE Technologies

- Java Servlets
- JSP
- EJB
- JMS
- JDBC
- JNDI
- JTA / JTS
- JavaMail
- JAAS
- XML
- ...

J2EE Components



Java Servlets

7

- Our primary user interface for MSIS531 will be the web browser
 - Most Java courses start from the command line
 - We'll do a couple of examples from there, then use the web for everything else
- Servlets are the Java technology for extending and enhancing web servers
 - Component-based, platform-independent method for building web-based applications
 - Access to all Java APIs
- As we will see, a servlet is just a Java class with special attributes to produce HTML as output

© UW Business School, University of Washington 2006



JSP – JavaServer Pages

8

- JavaServer Pages technology uses XML-like tags and scriptlets written in Java to encapsulate the logic that generates the content for the page
- Any and all formatting (HTML or XML) tags are passed directly back to the response page
- Separates page logic design and display
- We'll use JavaServer Faces, the follow-on technology to JSP; more on this next time
- Behind the scenes, JSP/JSP pages are converted to servlets

© UW Business School, University of Washington 2006



EJB – Enterprise Java Beans

9

- Enterprise JavaBeans™ is the server-side J2EE component architecture
 - Enables rapid, simplified development of distributed, transactional, secure and portable Java applications.
 - EJBs are components that are deployed into containers
 - The container provides services:
 - Loading / Initialization
 - Transactions
 - Persistence
 - Communication with EJB clients
 - Enterprise Naming Context (JNDI name space)

© UW Business School, University of Washington 2006



The Spring Framework – EJB “lite” (?)

10

- EJB downside: complex technology stack
 - Get it all, even if not used
 - May be required to implement unused portions
 - Spring was developed (in part) out of frustration w/EJB
 - Lightweight “container” for Java objects
 - Abstraction of data access
 - Full transaction support
 - MVC web framework
 - Simplified access to other J2EE APIs
 - Support for AOP
 - An amazing technology

© UW Business School, University of Washington 2006



JMS – Java Message Service

11

- JMS provides a reliable, flexible service for the asynchronous exchange of critical business data and events.
 - API adds a common provider framework for developing portable, message based applications
- Useful for:
 - Loosely-coupled systems
 - Publish / Subscribe metaphor
 - Integration with other messaging systems

© UW Business School, University of Washington 2006



JDBC – Data Access API

12

- Java’s answer to ODBC
- JDBC is an API that lets you access virtually any tabular data source from a Java program
 - Cross-DBMS connectivity to a wide range of SQL databases
 - We’ll use MySQL for MSIS531, but all of the big vendors (including Microsoft) support JDBC access
 - Access to other tabular data sources, such as spreadsheets or flat files
- We’ll begin by implementing a JDBC solution
 - Discover both benefits and shortcomings

© UW Business School, University of Washington 2006



JDBC shortcomings & Hibernate

13

- Big issue for Java programmers: mismatch between Java objects and RDBMS tables
 - Ideal outcome: persist an object without concern for underlying implementation, SQL, etc.
- Enter Hibernate: so-called ORM (Object/Relational Mapping) tool
 - Uses XML descriptors as a “glue” layer between classes and tables
 - Support for:
 - Transactions (CRUD)
 - Persistence
 - Lifecycle management

© UW Business School, University of Washington 2006



JNDI – Java Naming and Directory Interface

14

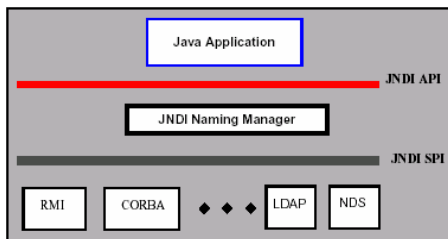
- JNDI provides naming and directory functionality
- Designed using Java's object model
- Purposes:
 - Java applications can store and retrieve named Java objects of any type
 - JNDI provides methods for performing standard directory operations, such as associating attributes with objects and searching for objects using their attributes
 - Allows applications to take advantage of information in a variety of existing naming and directory services, such as LDAP, NDS, DNS, and NIS(YF)

© UW Business School, University of Washington 2006



JNDI - Layers

15



© UW Business School, University of Washington 2006



JTA / JTS – Transactions

16

- The Java Transaction API (JTA) and the Java Transaction Service (JTS) allow J2EE application servers to take the burden of transaction management off of the component developer
- Developers can define the transactional properties of Enterprise JavaBeans™ technology based components during design or deployment using declarative statements in the deployment descriptor
- The application server takes over the transaction management responsibilities

© UW Business School, University of Washington 2006



JavaMail

17

- The JavaMail API provides a set of abstract classes that model a mail system
- The API provides a platform independent and protocol independent framework to build Java technology-based mail and messaging applications
- J2EE contains JAF – JavaBeans Activation Framework since it is required by JavaMail
- Support offered for common mail protocols:
 - IMAP
 - POP
 - SMTP
 - MIME

© UW Business School, University of Washington 2006



JAAS – Java Authentication and Authorization Service

18

- JAAS provides:
 - *Authentication* of users, to reliably and securely determine who is currently executing Java code, regardless of whether the code is running as an application, an applet, a bean, or a servlet; and
 - *Authorization* of users to ensure they have the access control rights (permissions) required to do the actions performed.
- Sample authentication modules using:
 - Java™ Naming and Directory Interface (JNDI)
 - Unix Operating Environment
 - Windows NT
 - Kerberos, Keystore, etc.

© UW Business School, University of Washington 2006



J2EE and XML

19

- The J2EE tech stack includes support for a wide (and ever-increasing) range of XML technologies
 - J2EE includes JAXP 1.1 support, as well as Servlet Filters and XML JSP documents
 - The Java API for XML Processing ("JAXP") supports processing of XML documents using DOM, SAX, and XSLT
 - Wide range of open source solutions also available
- We'll look in detail at XML support in the web services section
 - Development goal: "bolt on" XML support without worrying too much about implementation details

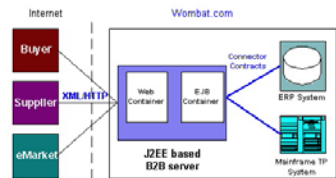
© UW Business School, University of Washington 2006



J2EE Connectors

20

- The J2EE Connector architecture defines a standard architecture for connecting the J2EE platform to heterogeneous EISs (Enterprise Information Systems)
 - EIS examples: ERP, mainframe transaction processing, database systems, legacy applications not written in Java

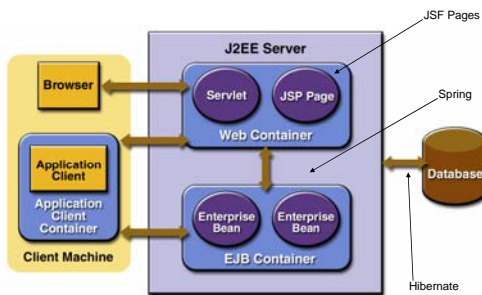


© UW Business School, University of Washington 2006



J2EE Applications

21



© UW Business School, University of Washington 2006



J2EE Deployment

22

- JAR – Java ARchive
 - Java class file
 - EJBs
- WAR - Web ARchive
 - Servlets
 - JSPs
- EAR - Enterprise ARchive
 - Contains other JARs and WARs to form an entire application
- Deployment descriptors
 - XML
 - Required for JARs, WARs, EARs
