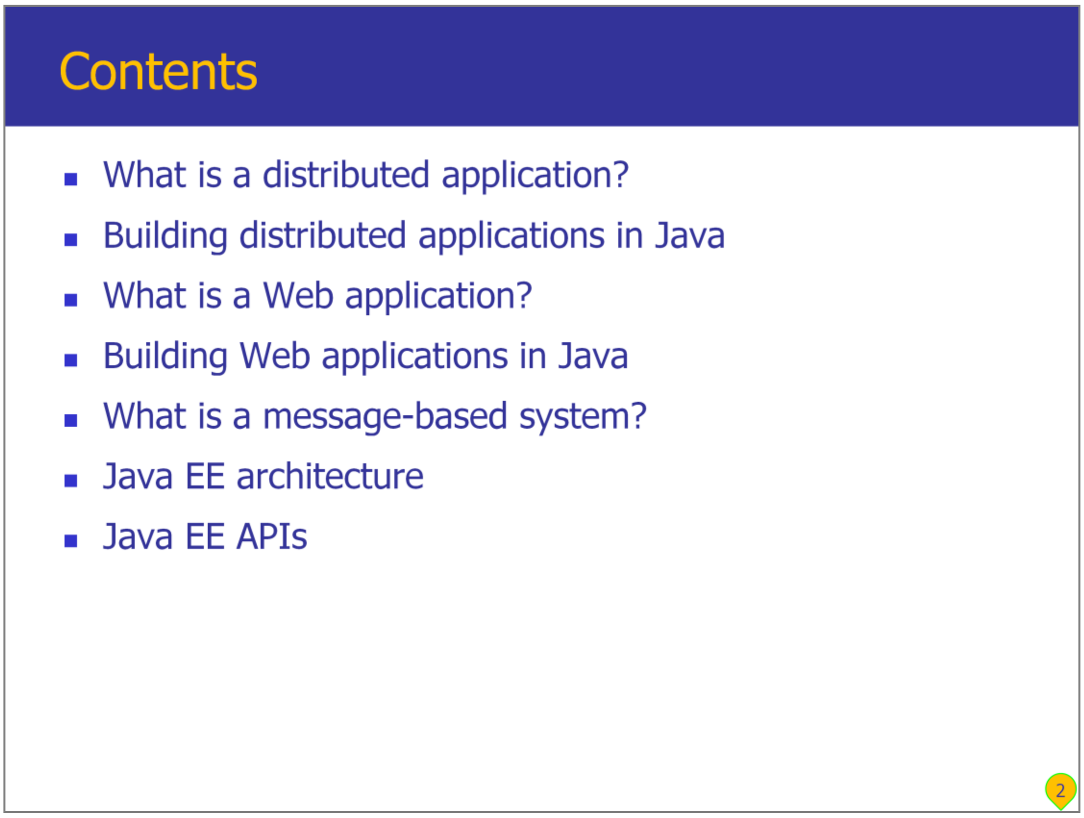
Overview of Java EE

olsen software ltd

This chapter describes the architecture of an enterprise Java application, and introduces essential Java EE concepts and terminology.

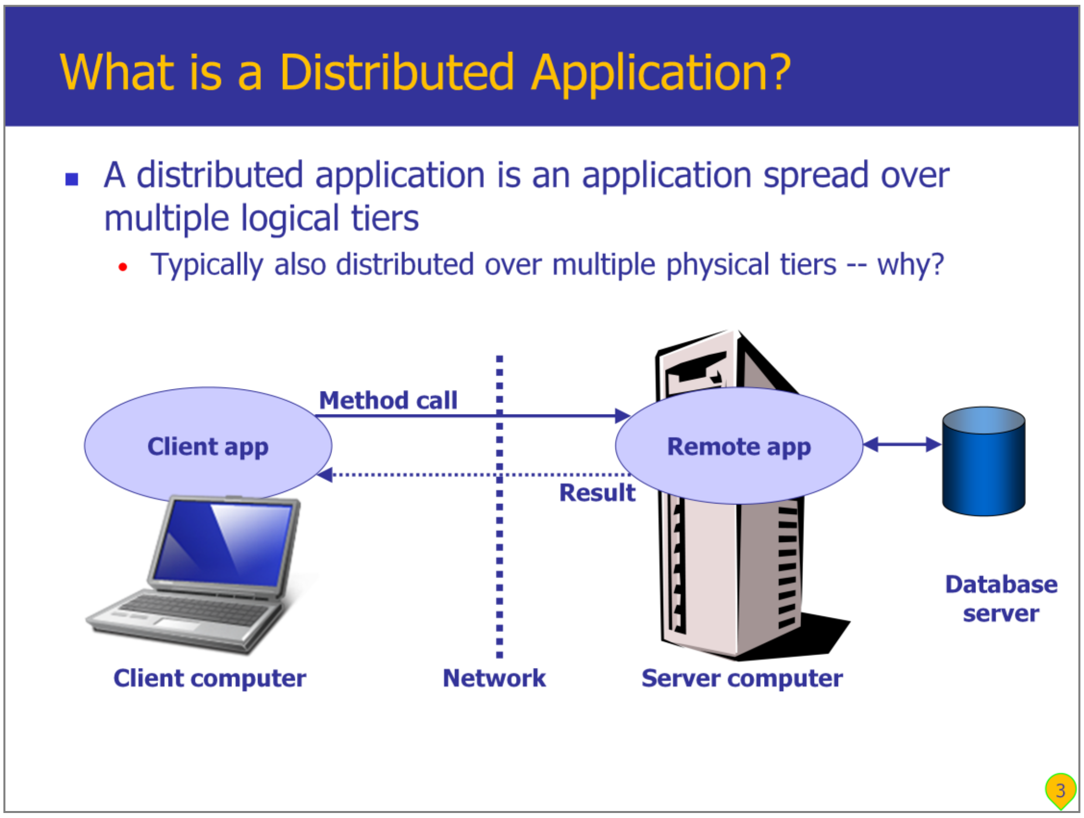


Contents

- What is a distributed application?
- Building distributed applications in Java
- What is a Web application?
- Building Web applications in Java
- What is a message-based system?
- Java EE architecture
- Java EE APIs

2

This chapter describes the different types of components that you can create in an enterprise Java application.

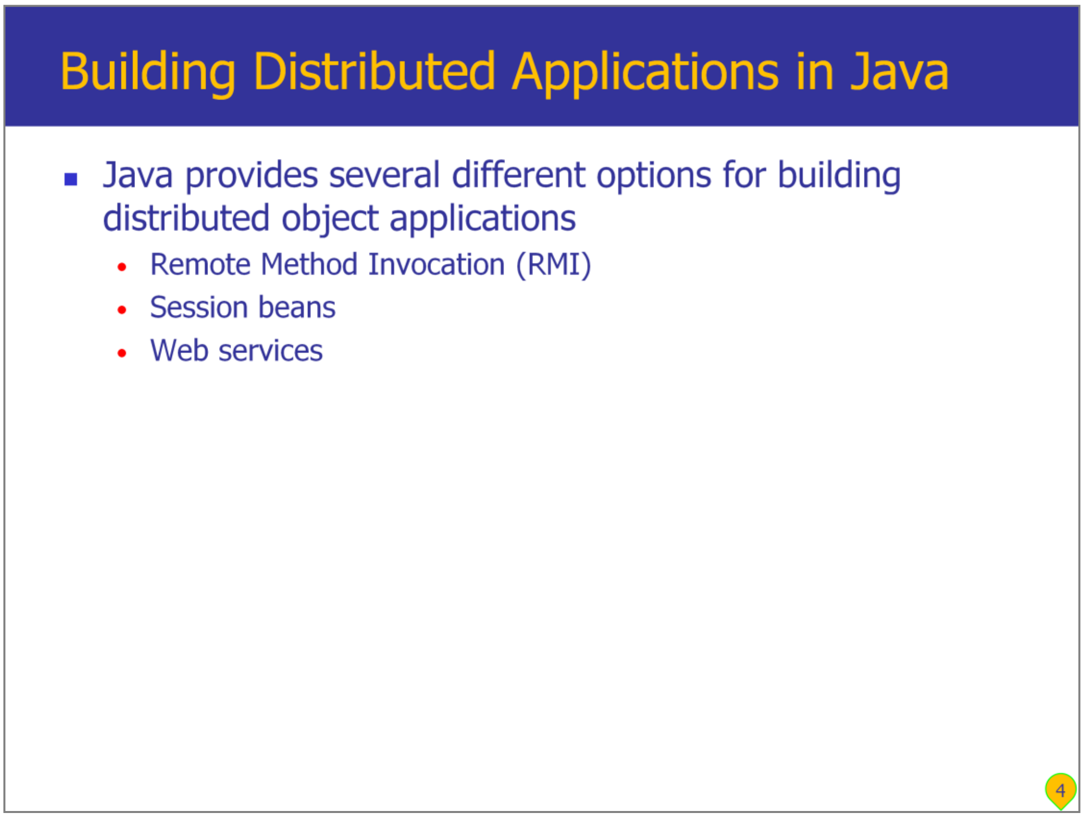


What is a Distributed Application?

- A distributed application is an application spread over multiple logical tiers
  - Typically also distributed over multiple physical tiers -- why?

Method call
Client app
Remote app
Result
Database server
Client computer
Network
Server computer
3

A distributed application comprises objects (typically) that interact with each other across a network or across a JVM boundary. Applications typically distribute functionality to achieve scalability at the server; to place functionality close to the data that it accesses; to place functionality into different security zones; to reuse centralized business rules; and for various other beneficial reasons.

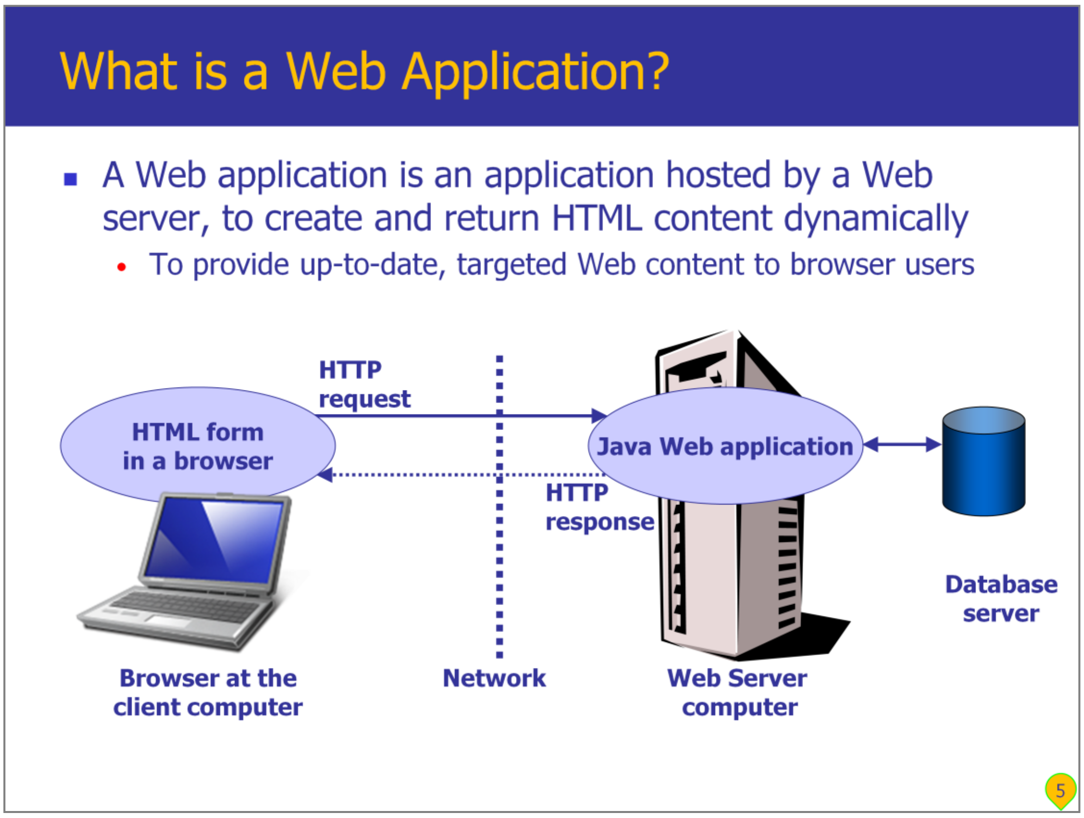


Building Distributed Applications in Java

- Java provides several different options for building distributed object applications
  - Remote Method Invocation (RMI)
  - Session beans
  - Web services

4

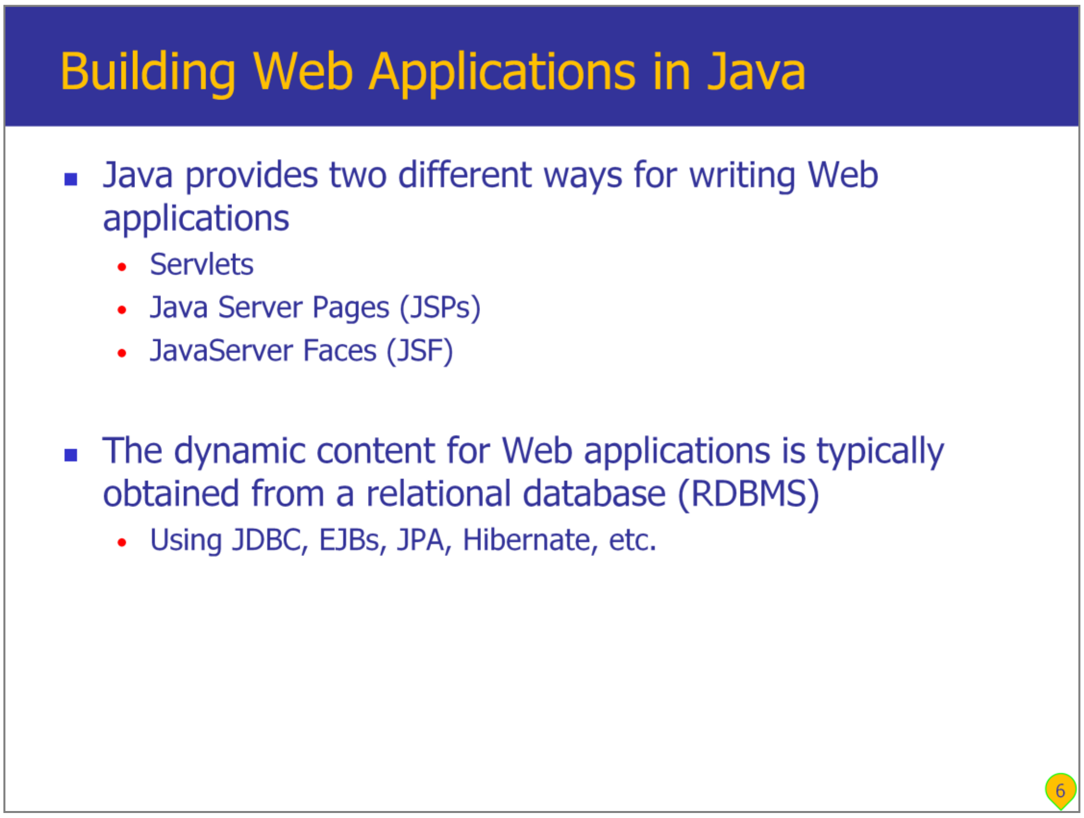
There are several ways to create distributed applications in Java. The contemporary choices are session beans (EJBs) and Web services.



What is a Web Application?

- A Web application is an application hosted by a Web server, to create and return HTML content dynamically
  - To provide up-to-date, targeted Web content to browser users

Web applications are commonplace and vital to many organizations. As you are doubtless aware, Web application are installed on a Web server and are invoked from a browser (or other client, possibly) over an HTTP connection.



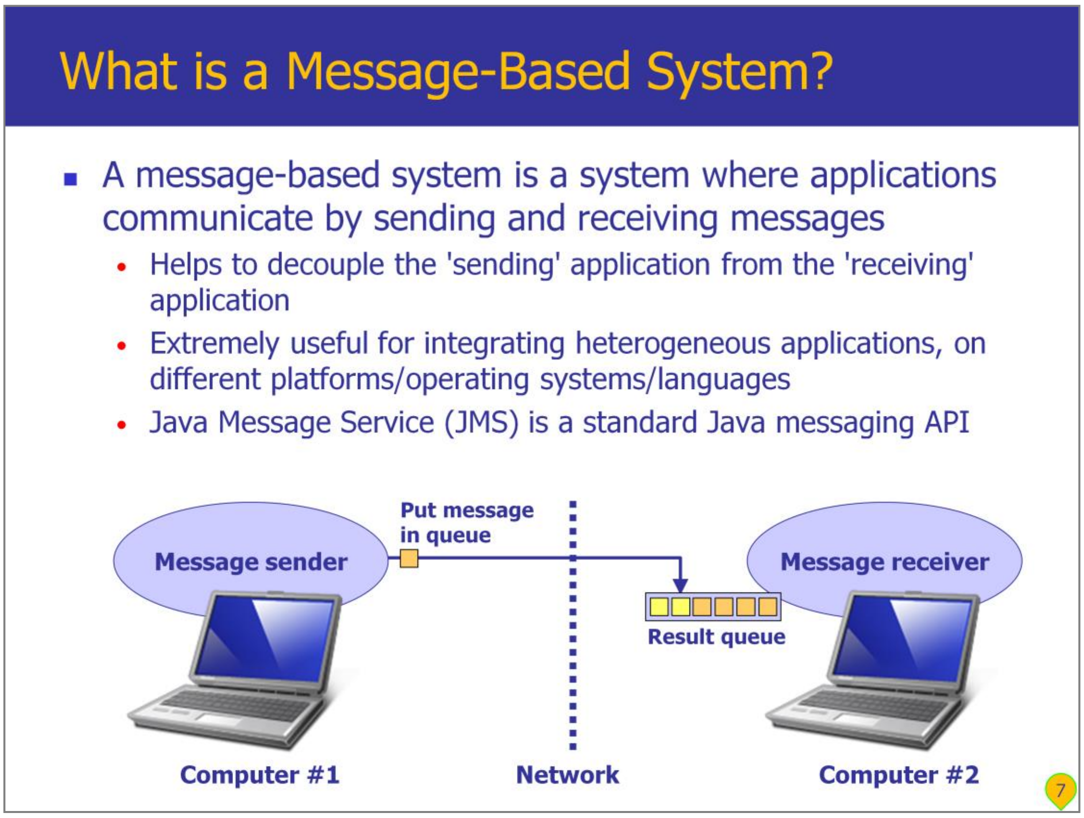
Building Web Applications in Java

- Java provides two different ways for writing Web applications
  - Servlets
  - Java Server Pages (JSPs)
  - JavaServer Faces (JSF)
- The dynamic content for Web applications is typically obtained from a relational database (RDBMS)
  - Using JDBC, EJBs, JPA, Hibernate, etc.

6

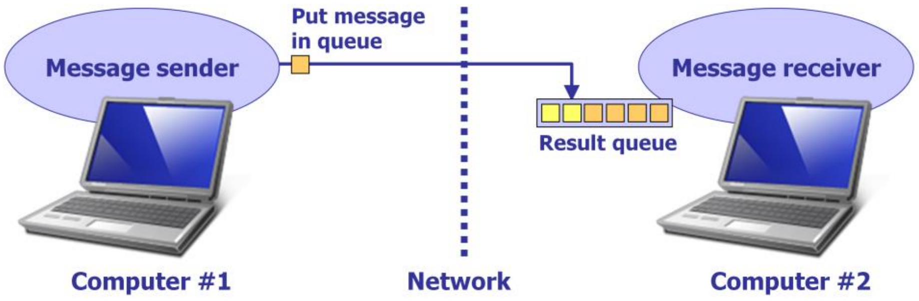
Java EE provides a wealth of Web-related technologies, including servlets and Java ServerPages (JSPs).

Web applications often need to get data dynamically from a database, and can do so by using JDBC or alternative technologies such entity EJBs, the Java Persistence API, and/or Hibernate.



What is a Message-Based System?

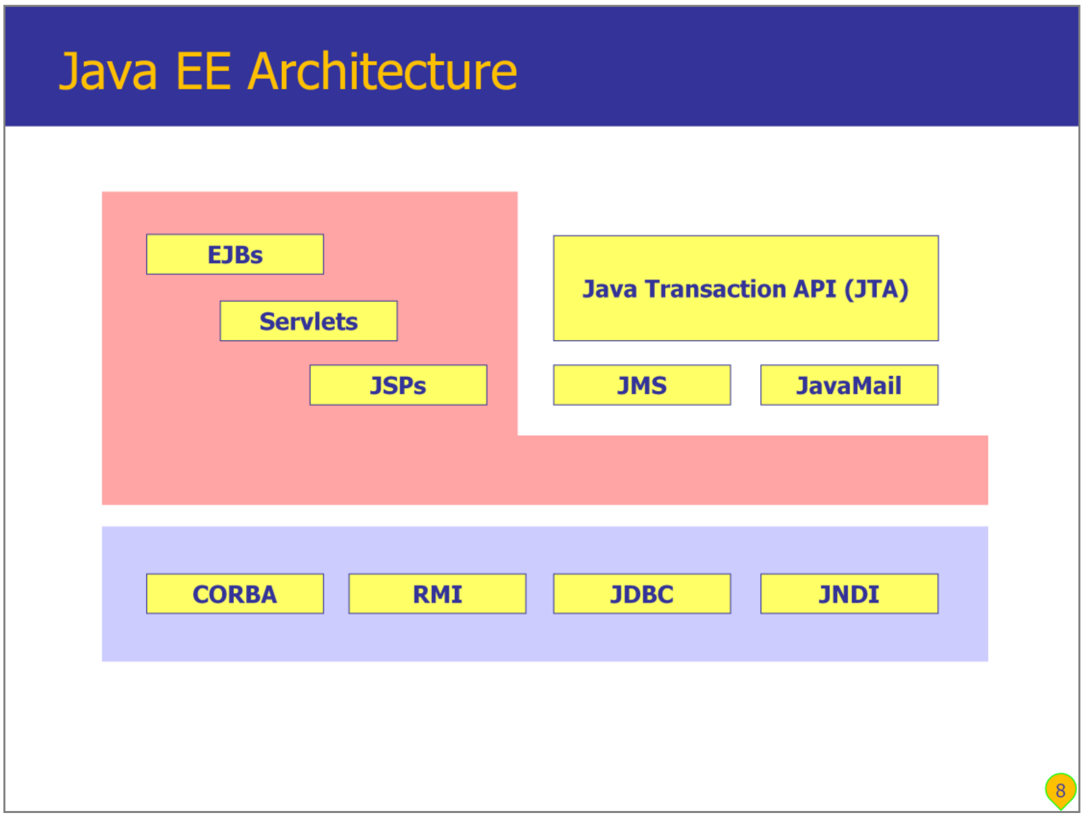
- A message-based system is a system where applications communicate by sending and receiving messages
  - Helps to decouple the 'sending' application from the 'receiving' application
  - Extremely useful for integrating heterogeneous applications, on different platforms/operating systems/languages
  - Java Message Service (JMS) is a standard Java messaging API



7

Many large corporate firms use message queueing products (such as IBM WebSphere) as the backbone for application communication. Message queuing provides a reliable, robust, and scalable approach to distributed computing.

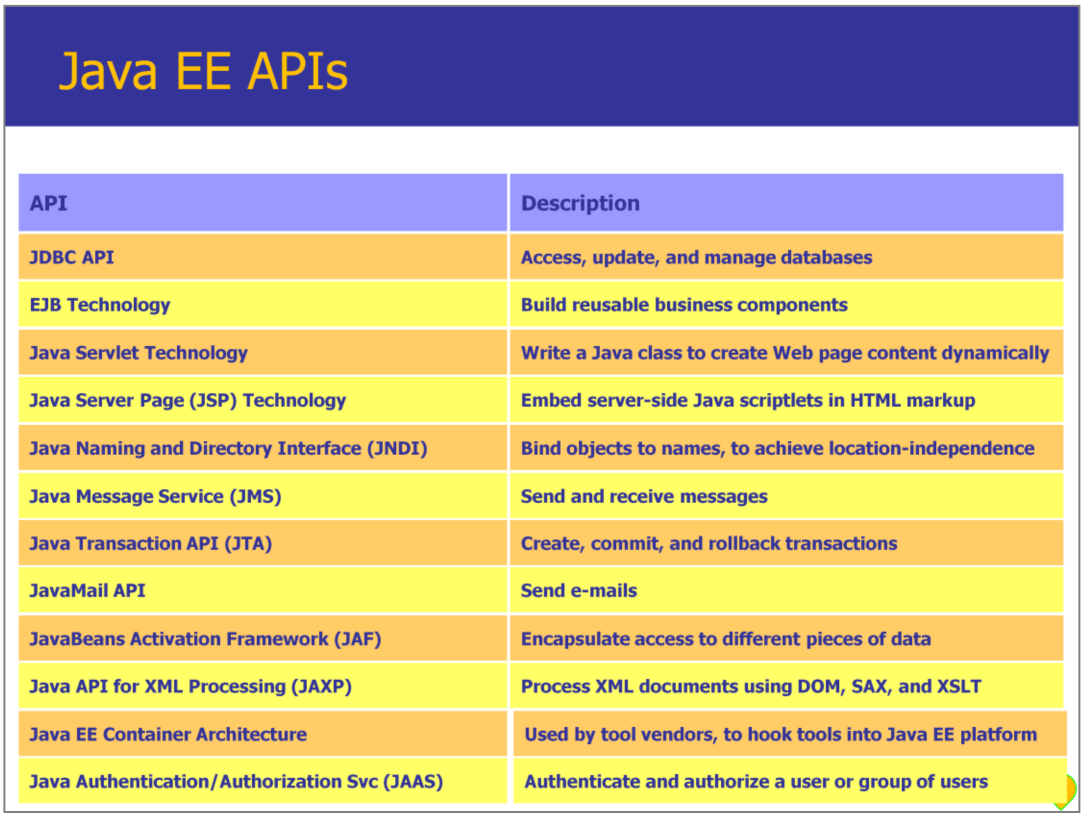
To implement a message-based system in Java EE, you can use the Java Message Service (JMS) API directly, or you can define message-driven beans (MDBs) to encapsulate the underlying plumbing.



Java EE Architecture

This slide summarizes Java EE architecture.

EJBs, servlets, and JSPs must run in the managed environment that is provided by a Java application server, such as JBoss, BEA WebLogic, or IBM WebSphere. In contrast, you can write CORBA, RMI, JDBC, and JNDI code in standalone applications that are independent of an application server if you want to. Likewise, you can use JTA, JMS, and JavaMail outside of an application server if appropriate.



Java EE APIs

API	Description
JDBC API	Access, update, and manage databases
EJB Technology	Build reusable business components
Java Servlet Technology	Write a Java class to create Web page content dynamically
Java Server Page (JSP) Technology	Embed server-side Java scriptlets in HTML markup
Java Naming and Directory Interface (JNDI)	Bind objects to names, to achieve location-independence
Java Message Service (JMS)	Send and receive messages
Java Transaction API (JTA)	Create, commit, and rollback transactions
JavaMail API	Send e-mails
JavaBeans Activation Framework (JAF)	Encapsulate access to different pieces of data
Java API for XML Processing (JAXP)	Process XML documents using DOM, SAX, and XSLT
Java EE Container Architecture	Used by tool vendors, to hook tools into Java EE platform
Java Authentication/Authorization Svc (JAAS)	Authenticate and authorize a user or group of users

This slide summarizes the technologies that make up Java Enterprise Edition. As a Java EE developer you can use some or all of these technologies as you see fit.

Any Questions?