COMP 3095
Lecture 1
Overview of Servlet and JSP Technology

Objectives: Applied vs Knowledge

- Applied Objectives:
  - These objectives ask you to apply what you have learned as your developing web applications.
  - Represent the critical objectives of programming course.

- Knowledge Objectives:
  - These objective define skills such as identifying, describing and explaining the required concepts, terms, procedures. These objective determine whether you are able to talk intelligently about the topic.

Objectives

Knowledge

1. Name software components that run on the client of a typical web application.
2. Name two software components that run on the server of a typical web application.
3. Distinguish between HTML and HTTP.
4. Distinguish between static web pages and dynamic web pages.
5. Name three approaches to developing Java web applications.
6. Describe components required for developing servlet and JSP applications.
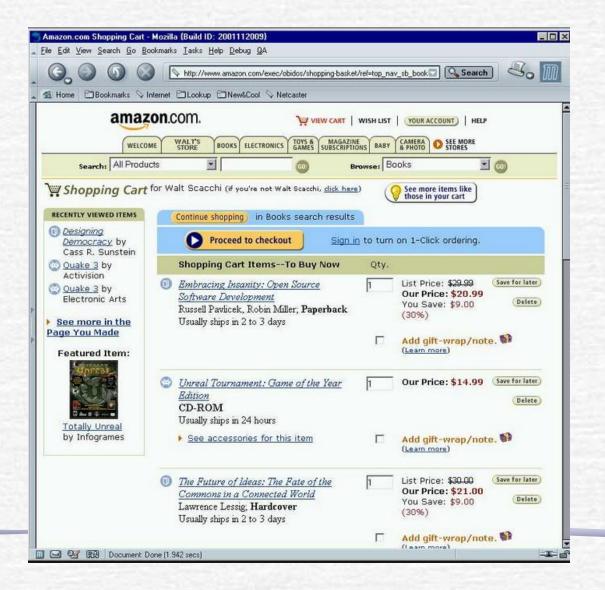7. List and describe the three layers of a typical Java web application.

<u>Knowledge</u>

8. In general describe the use of the following directories as described by J2EE specification.
   - webapps
   - document root
   - WEB-INF
   - WEB-INF\classes

9. Name two IDE's for Java Web Development.

10. Name two web servers for developing Java web applications.

11. Name a popular database server.

12. JSP vs JSF

13. What Servlets and JSPs are all about.
    - Understanding the role of servlets.
    - Building Web Pages dynamically.
    - Evaluating servlets vs. other technologies.
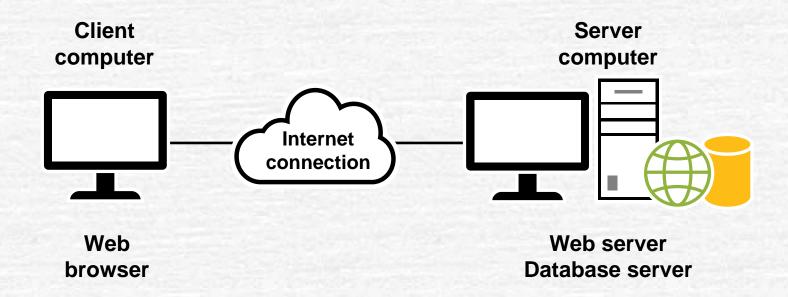   - Understanding the roles of JSP
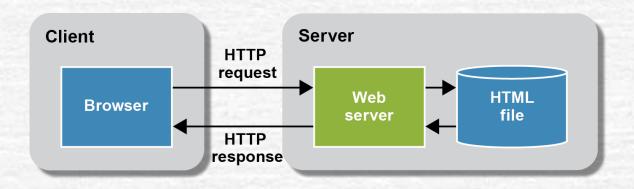
Modern Web Application

Modern Web Application
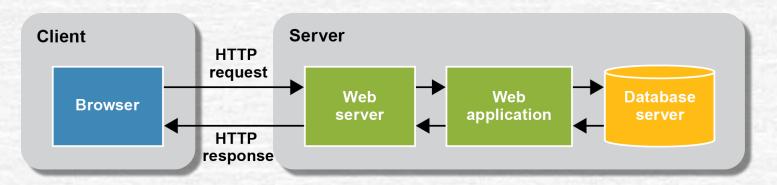
Components of Web Application

Client computer

Internet connection

Server computer

Web browser

Web server Database server

How a web server processes static web pages

How a web server processes dynamic web pages

Approaches for developing Java web apps

- Servlet/JSP:
  - Is a lower-level API that does less work for the programmer.
  - Provides a high degree of control over the HTML/CSS/JavaScript that's returned to the browser.
- JSF:
  - Is a higher-level API that does more work for the programmer.
  - Makes it more difficult to control the HTML/CSS/JavaScript that's returned to the browser.
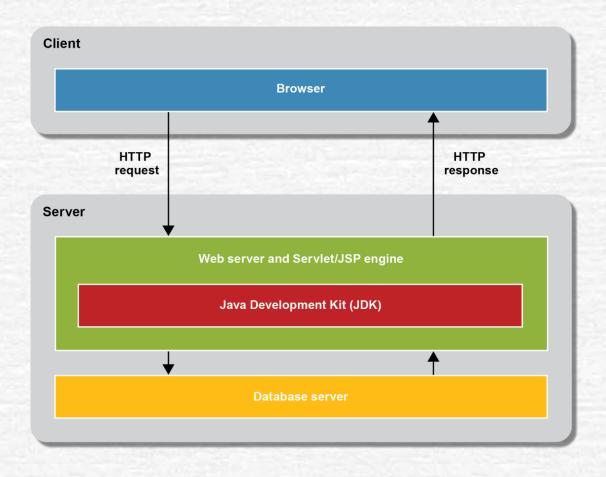- Framework (Spring / Struts):
  - Is an even higher-level API that does more work for the programmer.
  - Provides a high degree of control over the HTML/CSS/JavaScript that's returned to the browser.

Terminology

- **The Java Standard Edition (Java SE):** includes the Java Development Kit (JDK) and the Java Runtime Environment (JRE).

- **The Java Enterprise Edition (Java EE):** specification describes how web servers can interact with all Java web technologies.

- **Servlets:** store the Java code that does the server-side processing.

- **JavaServer Pages (JSPs):** store the HTML that defines the user interface.

- **JavaServer Faces (JSF):** provides a higher-level API that **replaces** both servlets and JSPs.

- **Java Persistence API (JPA):** is an API for working with databases.

The components of a servlet/JSP application

JSP vs JSF

JSP vs. JSF 2

- Servlets and JSP (JavaServer Pages)
  - Original, widely-deployed standard.
  - Used by google.com, ebay.com, Walmart.com.
  - Low-Level by todays standards.

- JSF2 (JavaServer Faces) Version 2
  - An official part of Java EE as of Java 6
  - Higher-level features: integrated Ajax support, field validation, page templating, rich third-party component libraries such as PrimeFaces, etc. Designed around the MVC architecture.
  - Recommended for almost all new projects

JSP vs JSF: When to Use Which?

- **Servlets and JSP**
  - For maintaining and extending legacy projects.

- **Servlets only**
  - For apps with front ends that do not use a server-side framework.
    - HTML with Jquery and Jquery UI.
  - Servlets primarily handle the Ajax requests from Jquery and do not build full pages.

- **JSF 2**
  - For almost all new projects that involve dynamic pages.
  - Usually combined with a rich component toolkit
    - Prime Faces (http://www.coreservlets.com/JSF-Tutorial/primefaces/)
    - Rich Faces (https://docs.jboss.org/richfaces/latest_3_3_X/en/devguide/html/GettingStarted.html)

Technologies Used Internally with JSF?

- Servlets
  - Servlets are still used behind the scenes and javax.faces.webapp.FacesServlet **controls** everything.
  - Many servlet APIs important in JSF.

- Servlets APIs most commonly used with JSF
  - Cookies (especially long-lived ones).
  - Setting **response headers** and **response status codes**.
  - Changing output based on User-Agent
    - String userAgent = request.getHeader("User-Agent");
    - User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10.5; en-US; rv:1.9.0.13) Gecko/2009073021 Firefox/3.0.13
  - Explicit Session manipulation.
  - Security.

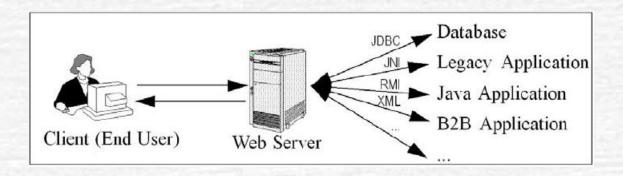# What are Servlets and JSP all about?

Web Applications

- Downside to browser-based apps
  - HTML is okay for static content but lousy for programs or dynamic content.
  - Communication is inefficient
    - HTML is poor protocol for the way we **now** use Web Applications

Why Web Applications?

Why does everyone want Web Apps?

1. Universal access
   - Every computer on a network has access
2. Automatic Updates
   - Content comes from server so data is never out of date.

A Servlets Job

- Read explicit data sent by client (form data)
- Read implicit data sent by client (request headers)
- Generate Results
- Send the explicit data back to client (html)
- Send the implicit data to client (status code + response header).

Why Build Web Pages Dynamically?

- The Web pages are based on data submitted by the User.
  - Example: Results page from search engine
  - Example: Order Confirmation pages from online stores.

- The Web page is derived from data that changes frequently.
  - Example: Weather reports or new headline pages.

- The Web page uses info from databases or other server-side sources.
  - Example: E-commerce site could user servlet to build a web page that lists the current price and availability of each item that is for sale.

Advantages of Servlets over Traditional CGI

- Efficient
  - Threads instead of OS processes, one servlet copy
- Convenient
  - Lots of high-level utilities
- Powerful
  - Sharing data, pooling, persistence.
- Portable
  - Run on virtually all operating systems.
- Inexpensive
  - There are plenty of free and low-cost servers.
- Secure
  - No shell scripts, no buffer overflows.

Mainstream

- Popular
  - JSP/Servlets: Single most common use of Java technology
  - Leading technology for medium /large web applications.
    - Google reports over 650 million Web Pages using JSP.
- Support
  - Apache, Oracle, IBM, Sybase, BEA, Jetty, Caucho, World Wide Wen consortium and many others.
- Runs on
  - Windows, Unix/Linux, MaxOS, VMS and IBM mainframe OSs.
- Used for
  - Airline companies, hotels, e-commerce sites, search engines, banks, financial sites etc.

Web Application Language popularity

- http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html