Objectives

- Review: HTML Forms
- Intro to Java Server-side Web Technology

Review: HTML Forms

- What attribute is required in a `form` tag?
  - What attribute is optional?
- What attribute do we use to create different types of `input`?
- How do we distinguish between input data?
- How do we "group" radio buttons and checkbox buttons?
- What tag do we use to improve usability of our radio buttons and checkboxes?
- When should we use "get" vs "post"?

Why Requirements/Specifications?

- "Most programmers regard anything that doesn't generate code to be a waste of time. Thinking doesn't generate code, and writing code without thinking is a recipe for bad code. Before we start to write any piece of code, we should understand what that code is supposed to do. Understanding requires thinking, and thinking is hard."
- In the words of the cartoonist Dick Guindon: "Writing is nature's way of letting you know how sloppy your thinking is."

Source: http://www.wired.com/opinion/2013/01/code-bugs-programming-why-we-need-specs

INTRODUCTION TO SERVER-SIDE PROGRAMMING

Architecture of the Web

Presentation-oriented:

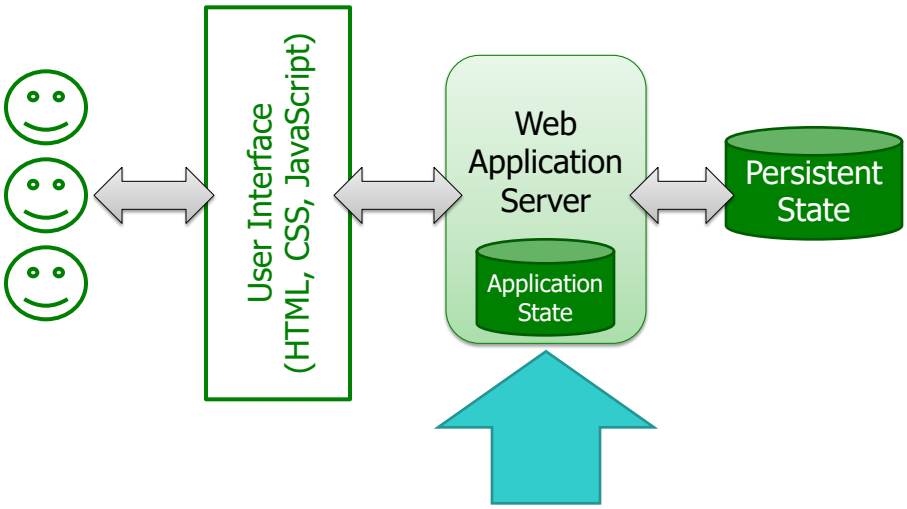
Web Browser
- Makes requests, renders responses
- Executes JavaScript, client-side code

Web Server handles static requests

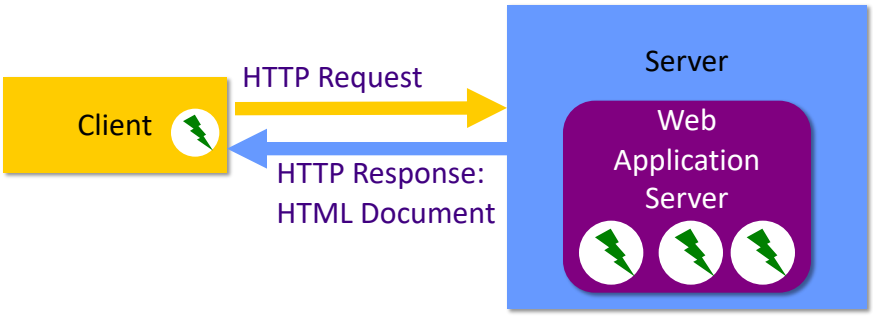
Web Application Server handles ***dynamic*** requests

Service-oriented Alternative:
Client is another server/non-user computer

Web Application Architecture



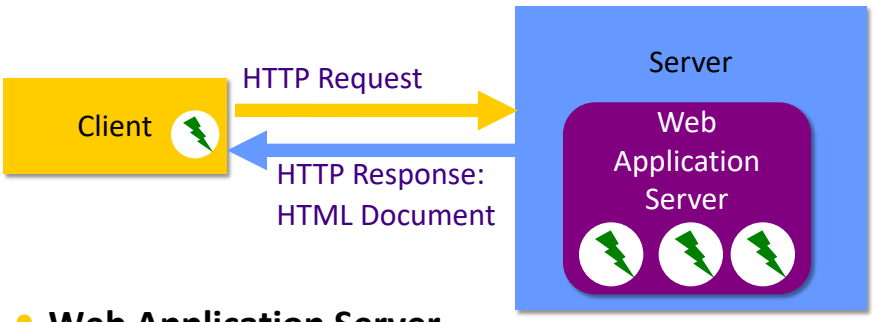
Java-based Web Application Server



Can have multiple Web applications running

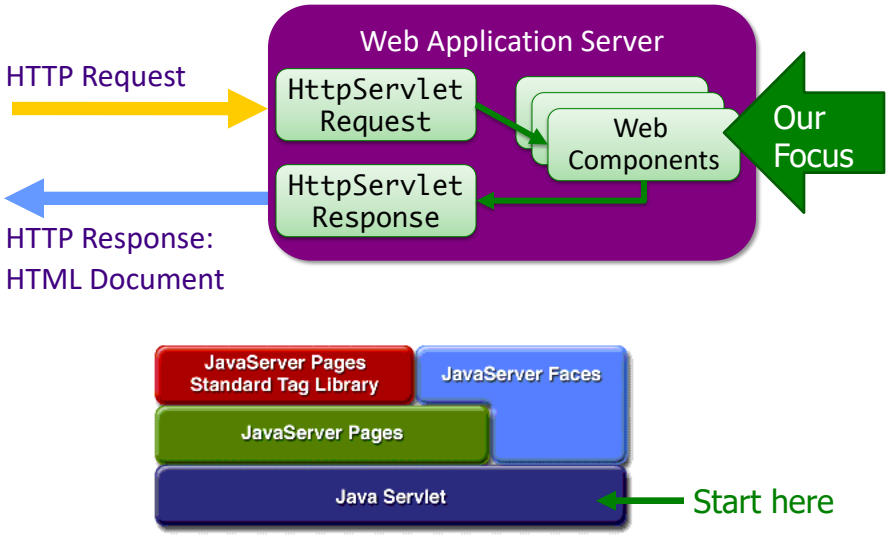
- Web Application Server
  - ***Container*** to run the Java-based web applications
  - Typically listens on port 8080 (rather than 80)

Architecture of the Web



- **Web Application Server**
  - Parses request, including data
  - Executes request
  - Returns response (often an HTML document)
    - May do other things, like send email, …

Request Handling in Java



Servlets

- A Java class that extends the functionality of web servers
  - Processes requests on server
  - Sends results (typically as an HTML file) back to client
- In `javax.servlet.*` packages
  - Part of Java Enterprise Edition (EE), as a separate download
  - Eclipse for EE development (Web Tools Platform)
- Java's answer to CGI (Common Gateway Interface)
- Portable, more secure (no buffer overflows)
- Supported by many major Web servers
  - E.g., Apache Tomcat, Jetty, WebSphere, etc.

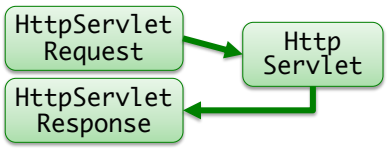
The `Servlet` Interface

Review from CSCI209: What is an *interface*?

- `javax.servlet.Servlet`
- All servlets implement the `Servlet` interface
  - `HttpServlet`, `GenericServlet`, `FacesServlet`
  - Web application server invokes many methods of `Servlet` automatically

The `HttpServlet` Class

- Web-based servlets typically extend `HttpServlet`
  - Implements `Servlet` interface
- `HttpServlet` implements the `service` method
  - Parameters
    - **`HttpServletRequest`** - from the client
    - **`HttpServletResponse`** - to the client
  - `service` calls the respective method (e.g., `doGet` or `doPost`) in response to a HTTP GET or POST request
- Recall:
  - **GET** - data encoded in URL
    - Request a resource (file) or retrieve data
  - **POST** - data encoded in body of message
    - Upload data; processing; hide data from URL



HttpServletResponse

- Provides output streams and methods to write data to the client

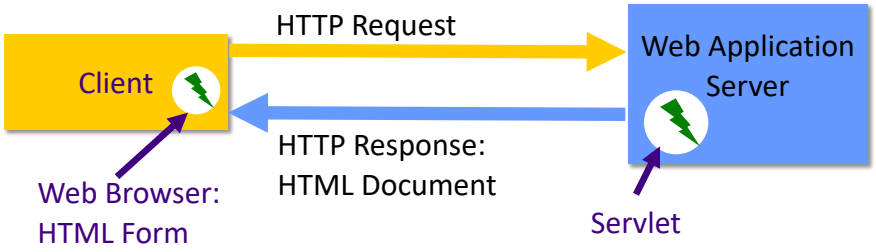
- **Methods:**
  - `ServletOutputStream getOutputStream()`
  - `PrintWriter getWriter()`
  - `void setContentType(String)`
    - Specifies the MIME type of the response
      - Browser knows what it received and how to format it
    - "`text/html`" specifies an HTML document

CSCI209 Flashback: Difference between *streams* and *writers*?

HttpServletResponse methods

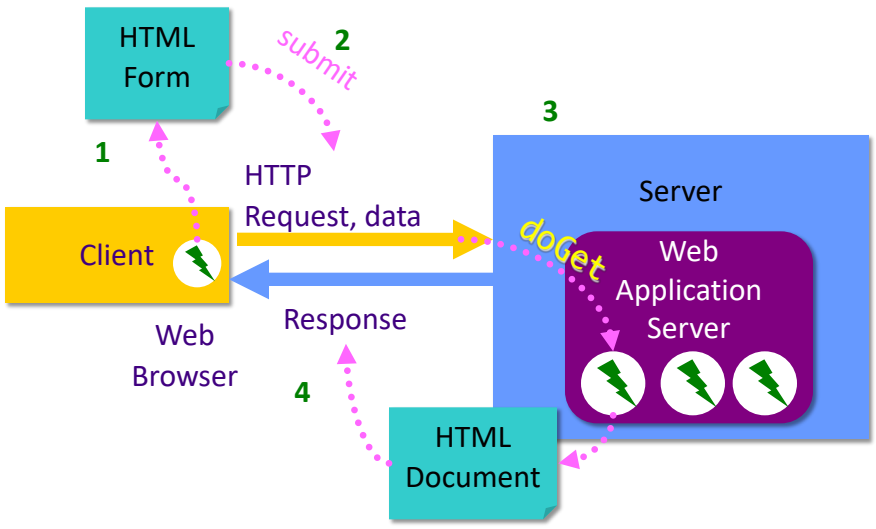
- `ServletOutputStream getOutputStream()`
  - Obtains a byte output stream that enables the servlet to send *binary data* to the client
- `PrintWriter getWriter()`
  - Obtains a text writer that enables the servlet to send *character data* (text) to the client
- `void setContentType(String)`
  - Specifies the MIME type of the response
    - Browser knows what it received and how to format it
  - "`text/html`" specifies an HTML document

Example Communication with Servlet



- HTML page with a form containing a submit button
  - Triggers client's request
- When the button is pressed, browser sends the servlet a GET request

Example Servlet Flow



To Generate a Response

- `doGet` method needs to
  - Obtain an output writer to write back to the client
  - Generate/write the HTML page to the client using the writer
  - Close the writer

How can we implement these steps?

```
void doGet(HttpServletRequest request,
           HttpServletResponse response)
```


Accepting Certain Types of Requests

- We can design the servlet to **only** accept/handle GET requests
  - Override the `doGet` method
  - Could return an error inside of `doPost`
- Could do the opposite: only accept/handle POST requests

HTTP Response Errors

- **HttpServletResponse** has a method for returning errors and fields that define errors codes

```
void sendError(int statusCode [,
String msg])
```

- Example status code fields:
  - SC_HTTP_VERSION_NOT_SUPPORTED
  - SC_METHOD_NOT_ALLOWED
  - SC_NOT_IMPLEMENTED

Tomcat

Eclipse/Tomcat

Using Eclipse

- HTML, CSS
- Start servlets

IN-CLASS WORK

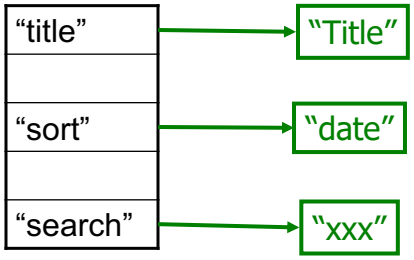
Handling Data

- So far, we haven't done anything with data that comes with the request

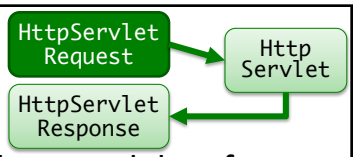
Requests for a digital publication library:

```
GET dspace/search?search=xxx&sort=date&title=Title
```

- Data is stored in a hashtable-like object



HttpServletRequest



- Provides input streams and methods to read data from the client
- Methods:
  - `String getParameter(String paramname)`
    - Returns the value of a request parameter
      - `null` if parameter doesn't exist
  - `String[] getParameterValues(String paramname)`
    - Returns an array of Strings containing the values for a specific request parameter
  - `Enumeration<String> getParameterNames()`
    - Returns the names of all of the parameters passed in the request

`HttpServletRequest` Methods

Requests for a digital publication library:

GET dspace/simple-search?search=xxx&sort=date&title=Title

- `getParameter("title")`
  - Returns "Title"
- `getParameterValues("sort")`
  - Returns ["date"]
- `Enumeration<String> getParameterNames()`
  - Returns the names of all of the parameters passed to the servlet

Note these are Strings

A More Complex Example: Survey

- An HTML form that asks the user for their favorite type of pet
- After user submits the form, the server sends back the current results of the survey
- Uses object serialization to write to/read from file

Deployment: WAR files

- Web Archives
  - Analog to JAR files
  - Bundles together all the code, files for the web application
- Copy into webapps directory of web application server
  - Server will automatically extract files and run
  - Procedure for Apache/Tomcat and Resin
- Can export WAR files from Eclipse
  - For submissions **MUST** include source code

TODO

- Complete Lab 3
  - Due tonight at midnight
- Requirements due tonight at midnight
  - Pick 3 "key" pages to mock up
    - Basis for other pages
- Read "Quality Attributes of Web Software Applications"
  - See Course Schedule for link
  - Writeup on Sakai
  - Due Wednesday @ midnight