Objectives

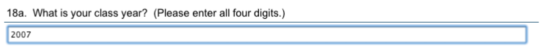
- Review Testing
- Error Handling
- Filters
- Ajax

https://www.facebook.com/ads/preferences

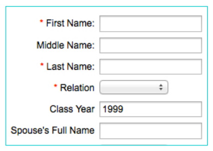
Testing Review

- Describe the general testing process
  - What validates/verifies the output from a program?
- How does the process change for Web applications?
- What are some of the testing frameworks?
  - How do they work?
- What is a tool that we will use for testing?
  - What are its advantages?
- What is the difference between testing and debugging?

Usability Discussion


18a. What is your class year? (Please enter all four digits.)

2007



* First Name:	
Middle Name:	
* Last Name:	
* Relation	
Class Year	1999
Spouse's Full Name	

Error Handling Discussion

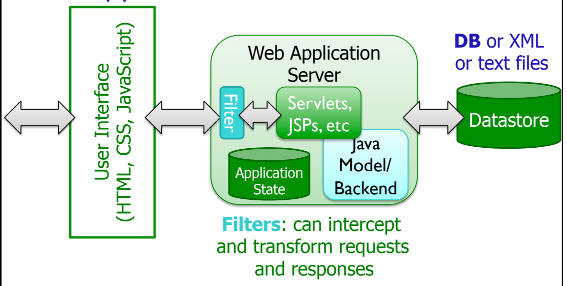
- What types of errors do web applications need to handle?
- Give several examples and describe how you can handle and/or prevent them?

Error Handling

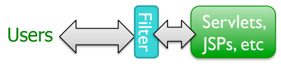
- User input/data
  - Client-side – JavaScript
  - Server-side – last line of defense
    - Handle no input (i.e., parameter is null)
- User navigation
  - Make sure user has permission to be where s/he is
  - WEB-INF – protects user from going to JSP page directly
  - Check permissions (e.g., kept in session)
- Security
  - More next week

SERVLET FILTERS

Web Application Architecture Overview



Filters



- Transform requests or responses
  - Ex: apply some encoding to all responses
- Can act as a function, attached to any servlet/JSP
  - Good modularization: reuse code for multiple servlets

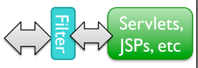
Example: Logging Accesses

- Want to log all accesses to the web application
- Created a `Filter` that intercepts all requests to an application and records them in a log
- Use the filter on multiple applications
  - Simply change the initialization parameters in web.xml

Example: Authorized Access

- We don't want users to get to various pages unless they are authorized to see those pages
- Instead of making every Servlet check authorization, we'll use a **Filter**
  - Requests go through Filter before going to the Servlet

Filter: AuthorizationFilter



- Verifies that user has the authority (permission) to access certain pages
- Requests go through filter before going to the requested Servlet
- `web.xml` file says which Servlets need to be filtered

Filter Implementation

- Implements `Filter` interface
- `doFilter` method implementation does the filter work, called for each attached servlet
- Use annotation to specify configuration parameters and which servlets to filter

Can also be specified in web.xml file

```
@WebFilter(
  urlPatterns = {
      "/My*"
  },
  initParams = {
      @WebInitParam(name = "myparam", value =
"myvalue", description = "example parameter")
})
```

All URLs that start with My

AJAX

Ajax: Asynchronous JavaScript + XML

- Not a programming language
- A way of using JavaScript
- Provides more responsive Web pages
  - Get data from a server without reloading your page
- Allows dynamically displaying data or updating the page without disturbing the user experience
- Aids in the creation of rich, user-friendly web sites
  - Examples: Google suggest tool and maps, Facebook, Flickr, A9

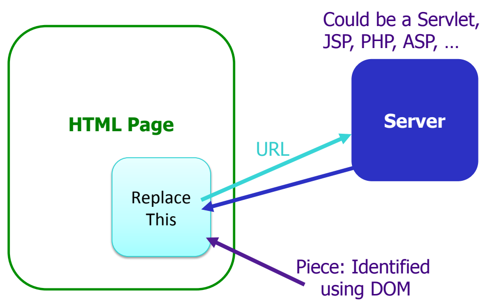
Ajax Difference

- In normal request/response HTTP cycles, the browser locks, waiting for the response and an entire page must be displayed
- With Ajax, asynchronous requests are made and responses update part of a page
  - User can continue to interact with a page while request is in progress
  - Less data needs to be transmitted
  - Page update is quicker because only part of a page is modified

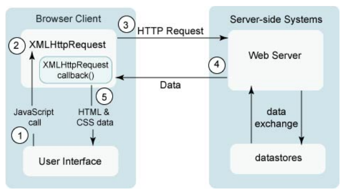
Core Ajax Concepts

- JavaScript's `XMLHttpRequest` object can fetch files from a web server
  - Supported in IE7+, Firefox, Safari, Opera, Chrome
- JavaScript can execute `XMLHttpRequest` asynchronously
  - In the background, transparent to user
- Contents of fetched file can be put into current web page using DOM
  - Reminder: Document Object Model
- Result: user's web page updates dynamically without a page reload

Ajax Request Overview

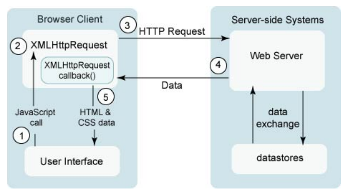

Could be a Servlet, JSP, PHP, ASP, …
HTML Page
Server
URL
Replace This
Piece: Identified using DOM

Typical Ajax Request



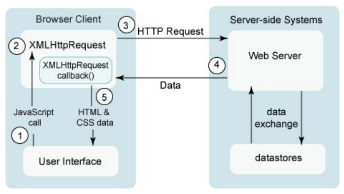
1. User clicks, invokes event handler
2. Handler's JS code creates `XMLHttpRequest` object
3. `XMLHttpRequest` object requests information from a web server

Typical Ajax Request



4. Server retrieves appropriate data, sends it back
5. `XMLHttpRequest` fires event when data arrives
   - Called a **callback**
6. Can attach a handler to be notified when data arrives to parse data, update web page

Typical Ajax Request



- Data can be any text format: HTML, XML, Text, …

How Does This Change Communication Patterns?

- Leads to smaller, more frequent communication with server

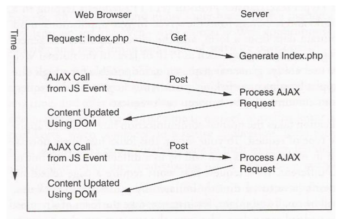


FIGURE 1-2
AJAX application request flow

`XMLHttpRequest` Object

- Methods
  - `abort, getAllResponseHeaders, getResponseHeader, open, send, setRequestHeader`
- Properties
  - `onreadystatechange, readyState, responseText, responseXML, status, statusText`

Using `XMLHttpRequest`

- Attach event handler to `onreadystatechange` event
  - Handler called when request state changes, e.g., *finishes*
  - `function` contains code to run when request completes
- Replace `url` with file you want to download
- Send the request

In an onscreen control's event handler:

```
var ajax = new XMLHttpRequest();
ajax.onreadystatechange = function;
ajax.open("GET", url, true);
ajax.send(null);
```

XMLHttpRequest's `readyState` Property

- Holds the status of the `XMLHttpRequest`
- Changes value from 0 to 4 during a request cycle:
  - 0: not initialized
  - 1: connection established
  - 2: request sent
  - 3: processing
  - 4: finished and response is ready
- `readyState` changes → `onreadystatechange` handler (*callback* function) runs
- Usually we are only interested in `readyState` of 4

Callback Function Example

Only do something when `readyState` is 4

Http Status Code

```
function processChange() {
    // 4 means the response has been returned, ready to be processed
    if (ajax.readyState == 4) {
        // 200 means "OK"
        if (ajax.status == 200) {
           // process whatever has been sent back here …

          // any other status means a problem
        } else {
            alert("There was a problem in the returned data:");
        }
    }
}
```

Ajax `XMLHttpRequest` template

- Most Ajax code uses an **anonymous function** as the event handler
  - A function declared inside another and not given a name
  - Useful because it can access the surrounding local variable

```
var ajax = new XMLHttpRequest();
ajax.onreadystatechange = function() {
    if (ajax.readyState == 4) {
        do something with ajax.responseText;
    }
};
ajax.open("GET", url, true);
ajax.send(null);
```

What does this code do?

Browser Compatibility

```
function processXML(url) {
    if (window.XMLHttpRequest) {
        // obtain new object
        req = new XMLHttpRequest();
        // set the callback function
        req.onreadystatechange = processChange;
        // do a GET with the url; "true" for asynch
        req.open("GET", url, true);
        // null for GET with native object
        req.send(null);
    // IE/Windows ActiveX object for IE5, 6
    } else if (window.ActiveXObject) {
        req = new ActiveXObject("Microsoft.XMLHTTP");
        if (req) {
            req.onreadystatechange = processChange;
            req.open("GET", url, true);
            // don't send null for ActiveX
            req.send();
        }
    } // else browser does not support Ajax
    return req;
}
```

callback function

Pet Survey Example

Project Discussion

- Any places where Ajax could be useful?
- How far did you get?
- What are your next steps?

Looking Ahead

- Tuesday a.m. – Checkpoint with client
  - Increasingly close to "done"
- Final implementation
  - Friday, 12-2 p.m., Spring Fest (final implementation)