MORE JAVASCRIPT AND FORMS

Continuing with JavaScript and Forms

Announcements

- At the demo on Monday, we will review Assignments 3, 4, and 5

Standup

Discuss questions with your Scrum Team

Quiz

1. What type of input do you use if you want to send it with the form, but don't want the user to interact with it?
2. Should you use `get` or `post` to send an image to a server?
3. Write PHP code to get the list of files with a .txt extension from a directory, whose name is passed in as `test.php?dir="dir";`

And the answer is …

Quiz (Answers)

1. Form info without user interaction?
   - Hidden
2. `get` or `post` to send an image to a server?
   - Post, urls are limited to about 1024 characters.
3. Write PHP code to get the list of files a directory, whose name is passed in as test.php?dir="dir";

```
<?php
    $directory = $_REQUEST["dir"];
   $reviews = glob("$directory/*.txt")
    ?>
```

Answers (cont)


```
<?php
      $movie = $_REQUEST["film"];
     $reviews = glob("moviefiles/$movie/review*.txt")
      ?>
```

More forms

Reset Buttons


```
Name: <input type="text" name="name" /> <br />
Food: <input type="text" name="meal" value="pizza" />
<br />
<label>Meat? <input type="checkbox" name="meat" /></label>
<br />
<input type="reset" />
```

HTML

- specify custom text on the button by setting its value attribute

Grouping input: <fieldset>, <legend>


```
<fieldset>
      <legend>Credit cards:</legend>
      <input type="radio" name="cc" value="visa"
      checked="checked" /> Visa
      <input type="radio" name="cc" value="mastercard" />
      MasterCard
      <input type="radio" name="cc" value="amex" />
      American Express
</fieldset>
```

HTML

- fieldset groups related input fields, adds a border; legend supplies a caption

Common UI control errors

- "I changed the form's HTML code ... but when I refresh, the page doesn't update!"
- By default, when you refresh a page, it leaves the previous values in all form controls
  - it does this in case you were filling out a long form and needed to refresh/return to it
  - if you want it to clear out all UI controls' state and values, you must do a full refresh
    - Firefox: Shift-Ctrl-R
    - Mac: Shift-Command-R

Styling form controls


```
input[type="text"] {
        background-color: yellow;
        font-weight: bold;
}
```
CSS

- attribute selector: matches only elements that have a particular attribute value
- useful for controls because many share the same element (input)

Hidden input parameters


```
<input type="text" name="username" /> Name <br />
<input type="text" name="sid" /> SID <br />
<input type="hidden" name="school" value="UW" />
<input type="hidden" name="year" value="2048" />
```

HTML

- an invisible parameter that is still passed to the server when form is submitted
- useful for passing on additional state that isn't modified by the user

Submitting data

Problems with submitting data


```
<form action="http://localhost/test1.php" method="get">
<label><input type="radio" name="cc" /> Visa</label>
<label><input type="radio" name="cc" /> MasterCard</label>
<br />
Favorite Star Trek captain:
<select name="startrek">
        <option>James T. Kirk</option>
        <option>Jean-Luc Picard</option>
</select> <br />
</form>
```

HTML

- the form may look correct, but when you submit it...
- `[cc] => on, [startrek] => Jean-Luc Picard`
- How can we resolve this conflict?

The `value` attribute


```
<label><input type="radio" name="cc" value="visa" />
Visa</label>
<label><input type="radio" name="cc" value="mastercard" />
MasterCard</label> <br />
Favorite Star Trek captain:
<select name="startrek">
        <option value="kirk">James T. Kirk</option>
        <option value="picard">Jean-Luc Picard</option>
<input type="submit" value="submit" />
</select> <br />
```

HTML

- value attribute sets what will be submitted if a control is selected
- `[cc] => visa, [startrek] => picard`

URL-encoding

- certain characters are not allowed in URL query parameters:
  - examples: " ", "/", "=", "&"
- when passing a parameter, it is URL-encoded
  - "Xenia's cool!?" → "Xenia%27s+cool%3F%21"
- you don't usually need to worry about this:
  - the browser automatically encodes parameters before sending them
  - the PHP $_REQUEST array automatically decodes them
  - ... but occasionally the encoded version does pop up (e.g. in Firebug)

Submitting data to a web server

- though browsers mostly retrieve data, sometimes you want to submit data to a server
  - Hotmail: Send a message
  - Flickr: Upload a photo
  - Google Calendar: Create an appointment
- the data is sent in HTTP requests to the server
  - with HTML forms
  - with **Ajax** (seen later)
- the data is placed into the request as parameters

HTTP GET vs. POST requests

- GET : asks a server for a page or data
  - if the request has parameters, they are sent in the URL as a query string
- POST : submits data to a web server and retrieves the server's response
  - if the request has parameters, they are embedded in the request's HTTP packet, not the URL

HTTP GET vs. POST requests

- For submitting data, a POST request is more appropriate than a GET
  - GET requests embed their parameters in their URLs
  - URLs are limited in length (~ 1024 characters)
  - URLs cannot contain special characters without encoding
  - private data in a URL can be seen or modified by users

Form POST example


```
<form action="http://localhost/app.php" method="post">
<div>
        Name: <input type="text" name="name" /> <br />
        Food: <input type="text" name="meal" /> <br />
        <label>Meat? <input type="checkbox" name="meat" /
></label> <br />
        <input type="submit" />
<div>
</form>
```

HTML

GET or POST?


```php
if ($_SERVER["REQUEST_METHOD"] == "GET") {
        # process a GET request
...
} elseif ($_SERVER["REQUEST_METHOD"] == "POST") {
        # process a POST request
...
}
```

PHP

- some PHP pages process both GET and POST requests
- to find out which kind of request we are currently processing, look at the global `$_SERVER` array's "`REQUEST_METHOD`" element

Uploading files


```
<form action="http://webster.cs.washington.edu/params.php"
method="post" enctype="multipart/form-data">
      Upload an image as your avatar:
      <input type="file" name="avatar" />
      <input type="submit" />
</form>
```
HTML

- add a file upload to your form as an input tag with type of file
- must also set the `enctype` attribute of the form

Processing form data in PHP

"Superglobal" arrays

Array	Description
$_REQUEST	parameters passed to any type of request
$_GET, $_POST	parameters passed to GET and POST requests
$_SERVER, $_ENV	information about the web server
$_FILES	files uploaded with the web request
$_SESSION, $_COOKIE	"cookies" used to identify the user (seen later)

- PHP superglobal arrays contain information about the current request, server, etc.
- These are special kinds of arrays called associative arrays.

Associative arrays


```php
$blackbook = array();
$blackbook[“xenia”] = "206-685-2181";
$blackbook[“anne”] = "206-685-9138";
...
print “Xenia's number is " . $blackbook[“xenia"] . ".\n";
```

PHP

- associative array (a.k.a. map, dictionary, hash table) : uses non-integer indexes
- associates a particular index "key" with a value
  - key “xenia" maps to value "206-685-2181"

Example: exponents


```php
<?php
        $base = $_REQUEST["base"];
        $exp = $_REQUEST["exponent"];
        $result = pow($base, $exp);
?>
<?= $base ?> ^ <?= $exp ?> = <?= $result ?>
```

PHP

- What should we do to run this with xampp?

Example: Print all parameters


```php
<?php
foreach ($_REQUEST as $param => $value) {
  ?>
  <p>Parameter <?= $param ?> has value <?= $value ?></p>
  <?php
}
?>
```

PHP

- What should we do to run this with xampp?

Processing an uploaded file in PHP

- uploaded files are placed into global array `$_FILES`, not `$_REQUEST`
- each element of `$_FILES` is itself an associative array, containing:
  - `name`: the local filename that the user uploaded
  - `type`: the MIME type of data that was uploaded, such as image/jpeg
  - `size` : file's size in bytes
  - `tmp_name` : a filename where PHP has temporarily saved the uploaded file
    - to permanently store the file, move it from this location into some other file

Uploading files


```
<input type="file" name="avatar" />
```
HTML

- example: if you upload tobby.jpg as a parameter named avatar,
  - $_FILES["avatar"]["name"] will be “tobby.jpg"
  - $_FILES["avatar"]["type"] will be "image/jpeg"
  - $_FILES["avatar"]["tmp_name"] will be something like "/var/tmp/phpZtR4TI"

```
Array
(
    [file1] => Array
        (
            [name] => MyFile.txt (comes from the browser,
so treat as tainted)
            [type] => text/plain  (not sure where it gets
this from - assume the browser, so treat as tainted)
            [tmp_name] => /tmp/php/php1h4j1o (could be
anywhere on your system, depending on your config
settings, but the user has no control, so this isn't
tainted)
            [error] => UPLOAD_ERR_OK  (= 0)
            [size] => 123   (the size in bytes)
        )
   [file2] => Array
        (
            [name] => MyFile.jpg
            [type] => image/jpeg
            [tmp_name] => /tmp/php/php6hst32
            [error] => UPLOAD_ERR_OK
            [size] => 98174
        )
)
```

Processing uploaded file example


```php
$username = $_REQUEST["username"];
if (is_uploaded_file($_FILES["avatar"]["tmp_name"])) {
move_uploaded_file($_FILES["avatar"]["tmp_name"],
"$username/avatar.jpg");
        print "Saved uploaded file as $username/avatar.jpg
\n";
} else {
        print "Error: required file not uploaded";
}
```

PHP

- functions for dealing with uploaded files:
  - is_uploaded_file(filename)
    returns TRUE if the given filename was uploaded by the user
  - move_uploaded_file(from, to)
    moves from a temporary file location to a more permanent file

Including files: `include`


```php
include("header.php");
```

- inserts the entire contents of the given file into the PHP script's output page
- encourages modularity
- useful for defining reused functions needed by multiple pages

DOM and timers

Problems with JavaScript

JavaScript is a powerful language, but it has many flaws:

- the DOM can be clunky to use
- the same code doesn't always work the same way in every browser
  - code that works great in Firefox, Safari, ... will fail in IE and vice versa
- many developers work around these problems with hacks (checking if browser is IE, etc.)

Prototype framework


```
<script src=" https://ajax.googleapis.com/ajax/libs/
prototype/1.7.0.0/prototype.js "
type="text/javascript"></script>                    JS
```

- the Prototype JavaScript library adds many useful features to JavaScript:
  - many useful extensions to the DOM
  - added methods to String, Array, Date, Number, Object
  - improves event-driven programming
  - many cross-browser compatibility fixes
  - makes Ajax programming easier (seen later)

The $ function


```JS
$("id")
```

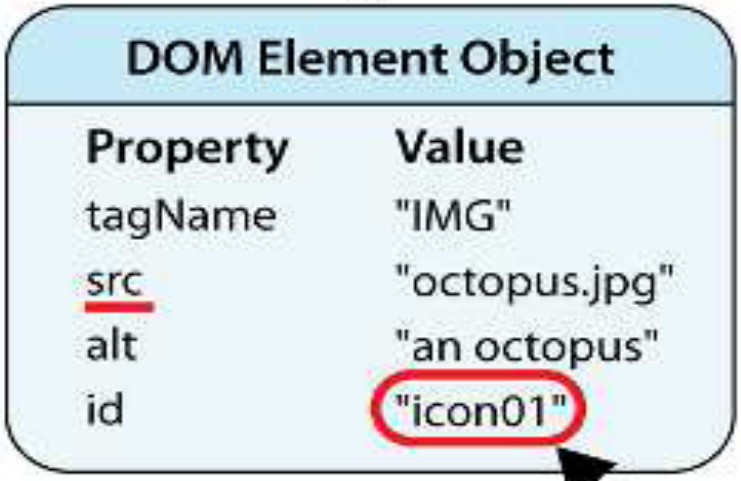
- returns the DOM object representing the element with the given id
- short for document.getElementById("id")
- often used to write more concise DOM code:

```JS
$("footer").innerHTML = $("username").value.toUpperCase();
```

DOM element objects

HTML

```
<p>
  Look at this octopus:
  <img src="octopus.jpg" alt="an octopus" id="icon01" />
  Cute, huh?
</p>
```



JavaScript

```
var icon = document.getElementById("icon01");
icon.src = "kitty.gif";
```

DOM object properties


```
<div id="main" class="foo bar">
<p>Hello, I am <em>very</em> happy to see you!</p>
<img id="icon" src="images/potter.jpg" alt="Potter" />
</div>
```
HTML

Property	Description	Example
tagName	element's HTML tag	$("main").tagName is "DIV"
className	CSS classes of element	$("main").className is "foo bar"
innerHTML	content inside element	$("main").innerHTML is "\n <p>Hello, ve...
src	URL target of an image	$("icon").src is "images/potter.jpg"

DOM properties for form controls


```
<input id="sid" type="text" size="7" maxlength="7" />
<input id="frosh" type="checkbox" checked="checked" />
Freshman?
```
HTML

Property	Description	Example
value	the text in an input control	$("sid").value could be "1234567"
checked	whether a box is checked	$("frosh").checked is true
disabled	whether a control is disabled (boolean)	$("frosh").disabled is false
readOnly	whether a text box is read-only	$("sid").readOnly is false

Abuse of innerHTML


```
// bad style!
var paragraph = document.getElementById("welcome");
paragraph.innerHTML = "<p>text and <a
href="page.html">link</a>";                                    JS
```

- innerHTML can inject arbitrary HTML content into the page
- however, this is prone to bugs and errors and is considered poor style

Adjusting styles with the DOM


```
<button id="clickme">Color Me</button>
```
HTML

```
window.onload = function() {
	document.getElementById("clickme").onclick =
changeColor;
};
function changeColor() {
	var clickMe = document.getElementById("clickme");
	clickMe.style.color = "red";
}
```
JS

- contains same properties as in CSS, but with camelCasedNames
  - **examples:** `backgroundColor`, `borderLeftWidth`, `fontFamily`

Common DOM styling errors

- forgetting to write .style when setting styles:

```
var clickMe = document.getElementById("clickme");
~~clickMe.color = "red";~~
clickMe.style.color = "red";                                    JS
```

- style properties are capitalized likeThis, not like-this:

```
~~clickMe.style.font-size = "14pt";~~
clickMe.style.fontSize = "14pt";                                JS
```

- style properties must be set as strings, often with units at the end:

```
~~clickMe.style.width = 200;~~
clickMe.style.width = "200px";
clickMe.style.padding = "0.5em";                                JS
```

Unobtrusive styling


```
function okayClick() {
        this.style.color = "red";
        this.className = "highlighted";
}
```
JS

```
.highlighted { color: red; }
```
CSS

- well-written JavaScript code should contain as little CSS as possible
- use JS to set CSS classes/IDs on elements
- define the styles of those classes/IDs in your CSS file

Timer events

method	description
setTimeout(function, delayMS);	arranges to call given function after given delay in ms
setInterval(function, delayMS);	arranges to call function repeatedly every delayMS ms
clearTimeout(timerID); clearInterval(timerID);	stops the given timer so it will not call its function

- both `setTimeout` and `setInterval` return an ID representing the timer
  - this ID can be passed `to clearTimeout/Interval` later to stop the timer

setTimeout example


```
<button onclick="delayMsg();">Click me!</button>
<span id="output"></span>
```
HTML

```
function delayMsg() {
        setTimeout(booyah, 5000);
        $("output").innerHTML = "Wait for it...";
}
function booyah() { // called when the timer goes off
        $("output").innerHTML = "BOOYAH!";
}
```
JS

setInterval example


```
<button onclick="delayMsg();">Click me!</button>
<span id="output"></span>
```
HTML

```
var timer = null; // stores ID of interval timer
function delayMsg2() {
	if (timer == null) {
		timer = setInterval(rudy, 1000);
	} else {
		clearInterval(timer);
		timer = null;
	}
}
function rudy() { // called each time the timer goes off
	$("output").innerHTML += " Rudy!";
}
```
JS

Passing parameters to timers


```
function delayedMultiply() {
// 6 and 7 are passed to multiply when timer goes off
      setTimeout(multiply, 2000, 6, 7);
}
function multiply(a, b) {
      alert(a * b);
}
```
JS

- any parameters after the delay are eventually passed to the timer function
- why not just write this?

```
setTimeout(multiply(6 * 7), 2000);
```
JS

Common timer errors


```
setTimeout(booyah(), 2000);
setTimeout(booyah, 2000);
setTimeout(multiply(num1 * num2), 2000);
setTimeout(multiply, 2000, num1, num2);
```
JS

- what does it actually do if you have the () ?
  - it calls the function immediately, rather than waiting the 2000ms!