

Web Programming/Scripting: PHP and Ajax

Dr. Michele Weigle
Department of Computer Science
Old Dominion University
mweigle@cs.odu.edu

<http://www.cs.odu.edu/~mweigle/CS312-F11>

1

What is PHP?



- ◆ **PHP: Hypertext Preprocessor**
- ◆ **Server-side scripting language**
 - » different from JavaScript (a *client-side* scripting language)
- ◆ **Free alternative to Microsoft's Active Server Pages (ASP)**
- ◆ **Can be directly embedded into HTML**
- ◆ **Syntax is similar to Perl and C**
- ◆ **PHP is often combined with a MySQL database (which we won't cover)**

<http://www.w3schools.com/php/default.asp>

2

What is AJAX?



- ◆ Asynchronous JavaScript And XML
- ◆ Uses JavaScript to send and receive data between a web browser and web server
- ◆ Based on JavaScript, XML, HTML, CSS
 - » there's nothing new for us to learn!
- ◆ Made popular in 2005 by Google
 - » Google Suggest – as you type in words in the search box, suggestions for what you might be looking for appear
- ◆ Uses XMLHttpRequest (XHR)
 - » which we've already discussed

<http://www.w3schools.com/ajax/>

3

PHP and Ajax

Outline

◆ PHP



- » Syntax
- » Strings
- » Conditionals
- » Forms
- » Arrays
- » Loops
- » Functions
- » PHP and XML

◆ Ajax



- » Recalling XMLHttpRequest
- » Examples
 - ❖ Suggest
 - ❖ XML

4

Basic Syntax

- ◆ Start with **<?php**
- ◆ End with **?>**
- ◆ Each code line must end with a semicolon
- ◆ Comments
 - » one-line `//`
 - » multi-line `/* ... */`
- ◆ To output text, use **echo ' '**
- ◆ All variables start with **\$**
 - » data type does not need to be set before declaring variable

```
$text = "Hello World";  
$num = 16;
```

http://www.w3schools.com/PHP/php_syntax.asp

5

Hello World Example

```
<html>  
  <head><title>PHP Test</title></head>  
  <body>  
    <?php echo '<p>Hello World</p>'; ?>  
  </body>  
</html>
```

file extension must be php

<http://www.cs.odu.edu/~mweigle/cs312/php/hello.php>

*must be world-executable
(chmod 755)*

<http://us.php.net/tut.php>

6

PHPInfo Example

- ◆ `phpinfo()` is a built-in function that displays useful information about your system setup (loaded PHP modules, predefined variables, configuration settings)

```
<html>
  <head><title>PHP Info</title></head>
  <body>
    <?php phpinfo(); ?>
  </body>
</html>
```

<http://www.cs.odu.edu/~mweigle/cs312/php/info.php>

<http://us.php.net/tut.php>

7

Displaying Browser Type

- ◆ Let's check what browser the user is using
 - » look at the "User-Agent:" option that is sent in the HTTP request header
 - » `$_SERVER` is a special reserved variable

```
<html>
  <head><title>PHP Browser Check</title></head>
  <body>
    <?php
      echo $_SERVER['HTTP_USER_AGENT'];
    ?>
  </body>
</html>
```

<http://www.cs.odu.edu/~mweigle/cs312/php/browser.php>

<http://us2.php.net/manual/en/reserved.variables.server.php>

8

Strings in PHP

◆ Concatenation operator .

```
$txt1 = "Hello World";  
$txt2 = "1234";  
echo $txt1 . " " . $txt2;
```

Hello World 1234

http://www.w3schools.com/PHP/php_string.asp

9

Useful String Functions

◆ strlen(*String*)

- » returns the *length* of the string
- » number of characters, including spaces

◆ strpos(*String*, *SearchString*)

- » returns the starting position of *SearchString* if found inside *String*
 - ❖ position starts from 0 (i.e., 0 is the first character)
- » returns FALSE if *SearchString* is not found in *String*

http://www.w3schools.com/PHP/php_string.asp

10

More PHP Syntax

◆ Familiar C++/Java/Perl operators

- » comparison: ==, !=, >, >=, <, <=
- » assignment: =, +=, *=, ...
- » logical: &&, ||, !

◆ Familiar C++/Java/Perl conditionals syntax

```
if (condition) {  
    statements;  
} elseif {  
    statements;  
} else {  
    statements;  
}
```

http://www.w3schools.com/PHP/php_if_else.asp

11

Displaying Browser Type Now With Conditionals

*<html>, <head>,
<body> tags removed
in remaining examples
to save space*

```
<?php  
if (strpos ($_SERVER['HTTP_USER_AGENT'], 'MSIE') != FALSE) {  
    echo "You are using Internet Explorer."  
} else {  
    echo "You aren't using Internet Explorer, you're using <br />" .  
    $_SERVER['HTTP_USER_AGENT'];  
}  
?>
```

<http://www.cs.odu.edu/~mweigle/cs312/php/browser2.php>

<http://us.php.net/tut.php>

12

Mixing PHP and HTML

- ◆ Logical flow of the script remains intact even if broken up with HTML statements.

```
<?php
if (strpos($_SERVER['HTTP_USER_AGENT'], 'MSIE') != FALSE) {
?>
<h3>strpos() returned non-false</h3>
<p>You are using Internet Explorer</p>
<?php
} else {
?>
<h3>strpos() returned false</h3>
<p>You are not using Internet Explorer, you're using</p>
<?php
    echo $_SERVER['HTTP_USER_AGENT'];
}
?>
```

<http://us.php.net/tut.php>

<http://www.cs.odu.edu/~mweigle/cs312/php/html-php.php>

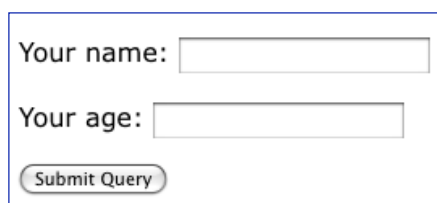
13

PHP and Forms

- ◆ Use a separate PHP file (not embedded in HTML)

```
<form action="action.php" method="post">
  <p>Your name: <input type="text" name="name" /> </p>
  <p>Your age: <input type="text" name="age" /> </p>
  <p><input type="submit" /> </p>
</form>
```

<http://www.cs.odu.edu/~mweigle/cs312/php/php-form.html>



Your name:

Your age:

<http://us.php.net/tut.php>

14

PHP and Forms

```
Hi <?php echo htmlspecialchars($_POST['name']); ?>.
You are <?php echo (int)$_POST['age']; ?> years old.
```

<http://www.cs.odu.edu/~mweigle/cs312/php/action.php>

`htmlspecialchars()` ensures that any special HTML characters are properly encoded so people can't inject HTML tags or JavaScript into your page.

- ◆ `$_POST`
 - » when "post" method is used
- ◆ `$_REQUEST`
 - ◆ when either "get" or "post" is used
- ◆ `$_GET`
 - » when "get" method is used

http://www.w3schools.com/PHP/php_get.asp

15

PHP Numeric Arrays

◆ Creation

```
$names = array("Peter", "Quagmire", "Joe");
```

OR

```
$names[0] = "Peter";
$names[1] = "Quagmire";
$names[2] = "Joe";
```

Length of an array:
`count (array)`

`count ($names)`
3

◆ Usage

```
$names[1]            Quagmire
```

http://www.w3schools.com/PHP/php_arrays.asp

16

PHP Associative Arrays

◆ Creation

```
$ages = array("Peter"=>"32", "Quagmire"=>"30", "Joe"=>"34");
```

OR

```
$ages['Peter'] = "32";
```

```
$ages['Quagmire'] = "30";
```

```
$ages['Joe'] = "34";
```

◆ Usage

```
$ages['Peter']          32
```

http://www.w3schools.com/PHP/php_arrays.asp

17

PHP Loops

◆ Familiar C++/Java syntax

```
while (condition) {  
    statements;  
}
```

```
do {  
    statements;  
} while (condition);
```

```
for (init; condition; increment)  
{  
    statements;  
}
```

◆ Includes Perl-like *foreach* statement

```
foreach (array as item) {  
    statements;  
}
```

```
$arr = array ("one", "two");  
foreach ($arr as $item) {  
    echo $item . "<br />";  
}
```

» Perl syntax is slightly different

http://www.w3schools.com/PHP/php_looping.asp

18

PHP Functions

- ◆ There are over 700 built-in functions available

- » <http://www.w3schools.com/PHP/default.asp>

- ◆ Writing your own functions

- » begin with the word **function**

- » syntax similar to JavaScript

```
function functionName () {  
    statements;  
}
```

http://www.w3schools.com/PHP/php_functions.asp

19

PHP Functions

- ◆ Functions with parameters

```
function functionName (parameters) {  
    statements;  
}
```

- ◆ Functions with return values

```
function functionName (parameters) {  
    statements;  
    return value;  
}
```

http://www.w3schools.com/PHP/php_functions.asp

20

PHP Function Example

```
function add ($x, $y)
{
    $total = $x + $y;
    return $total;
}

echo "1 + 16 = " . add (1,16);
```

1 + 16 = 17

http://www.w3schools.com/PHP/php_functions.asp

21

PHP and Ajax Outline

◆ PHP



- » Syntax
- » Strings
- » Conditionals
- » Forms
- » Arrays
- » Loops
- » Functions
- » PHP and XML

◆ Ajax



- » Recalling
XMLHttpRequest
- » Examples
 - ❖ Suggest
 - ❖ XML

22

PHP and XML

SimpleXML

- ◆ Must know the layout of the XML file
- ◆ Load XML file
 - » `$xml = simplexml_load_file ("url of xml file");`

<http://php.net/manual/en/book.simplexml.php>

23

Simple XML

Functions

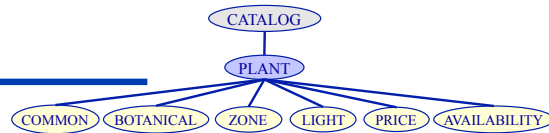
- ◆ Get the name of the first element
 - » `$xml->getName()`
- ◆ Loop through each child
 - » `foreach ($xml->children() as $child)`
 - ❖ to access label: `$child->getName()`
 - ❖ to access data: `$child`
- ◆ To get children of child
 - » `$child->children()`

24

Simple XML

Example

```
<?php
$xml = simplexml_load_file("plant_catalog.xml");
// access each plant
foreach ($xml->children() as $plant) {
    // access each element of the plant
    foreach ($plant->children() as $child) {
        // print label: data
        echo $child->getName() . ": " . $child . "<br />";
    }
}
?>
```



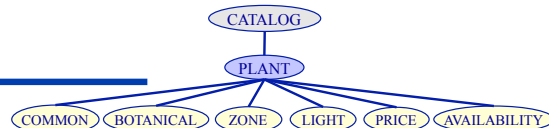
<http://www.cs.odu.edu/~mweigle/cs312/php/plants.php>

<http://www.cs.odu.edu/~mweigle/cs312/php/plants.php.txt>

25

SimpleXML

XPath



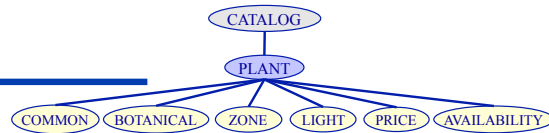
- ◆ After loading the xml file with SimpleXML, use XPath to find particular elements in the XML.
- ◆ Find all PLANT nodes whose common names are Bloodroot
 - » `$plant = $xml->xpath("//PLANT [COMMON='Bloodroot']");`
 - ❖ // - selects nodes in the document from the current node that match the selection no matter where they are
 - ❖ returns an array
- ◆ Access the price of the first plant that matches
 - » `$price = (string) $plant[0]->PRICE;`

http://www.w3schools.com/XPath/xpath_syntax.asp

26

XPath

Example



◆ Find all PLANT nodes with "Mostly Shady" light

```
$plant = $xml->xpath ("//PLANT [LIGHT='Mostly Shady']");
```

◆ Display all plants that match

```
$numPlants = count ($plant);  
for ($i=0; $i<$numPlants; $i++) {  
    echo "<p>"  
    foreach ($plant[$i]->children() as $child) {  
        echo $child->getName() . ": " . $child . "<br />";  
    }  
    echo "</p>";  
}
```

<http://www.cs.odu.edu/~mweigle/cs312/php/plants-xpath.php>
<http://www.cs.odu.edu/~mweigle/cs312/php/plants-xpath.php.txt>

27

Writing to XML

◆ addChild(*label*, *value*)

» creates (and returns) a new node

◆ asXML(*filename*)

» writes the current XML object to a file

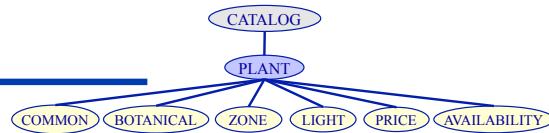
» XML file must be writeable by the web server (anyone)

❖ chmod 666

28

Writing to XML

Example



```
$xml = simplexml_load_file("plant_catalog.xml");
```

```
// add a new plant node (no text value)
```

```
$newPlant = $xml->addChild("PLANT", "");
```

```
$newPlant->addChild("COMMON", "Snakeroot");
```

```
$newPlant->addChild("BOTANICAL", "Cimicifuga");
```

```
$newPlant->addChild("ZONE", "Annual");
```

```
$newPlant->addChild("LIGHT", "Shade");
```

```
$newPlant->addChild("PRICE", "$5.63");
```

```
$newPlant->addChild("AVAILABILITY", "071199");
```

```
// write out entire XML file (with new entry)
```

```
$xml->asXML("plant_catalog.xml");
```

29

PHP and Ajax

Outline

◆ PHP



- » Syntax
- » Strings
- » Conditionals
- » Forms
- » Arrays
- » Loops
- » Functions
- » PHP and XML

◆ Ajax



- » Recalling
XMLHttpRequest
- » Examples
 - ❖ Suggest
 - ❖ XML

30

AJAX

Remember XMLHttpRequest (XHR)?

```
function sendXHR(url)
{
    if (xmlhttp == null) {
        xmlhttp = new XMLHttpRequest();
    }
    if (xmlhttp == null) {
        alert("Your browser does not support XMLHttpRequest.");
        return;
    }
    xmlhttp.onreadystatechange = stateChange;
    xmlhttp.open("GET", url, true);
    xmlhttp.send(null);
}
```

requests the document found at the given URL

xmlhttp is the variable that allows us to make the request

stateChange function is called when the state of the XMLHttpRequest object changes

open() sets up an HTTP request for the URL

send() sends the request to the server

<http://www.cs.odu.edu/~mweigle/cs312/xml/xmlHelperFns.js>

31

AJAX

Remember XMLHttpRequest (XHR)?

```
function stateChange()
{
    if (xmlhttp.readyState == 4) {
        if (xmlhttp.status == 200) {
            // HTTP response code (200 is OK)
            doStuff(); // you must write this function
        } else {
            alert ("Problem retrieving data: " + xmlhttp.statusText);
        }
    }
}
```

readyState

- 0 - request is not initialized
- 1 - request has been set up
- 2 - request has been sent
- 3 - request is in process
- 4 - request is complete

<http://www.cs.odu.edu/~mweigle/cs312/xml/xmlHelperFns.js>

32

PHP and Ajax

Outline

◆ PHP



- » Syntax
- » Strings
- » Conditionals
- » Forms
- » Arrays
- » Loops
- » Functions
- » PHP and XML

◆ Ajax



- » Recalling XMLHttpRequest
- » Examples
 - ❖ Suggest
 - ❖ XML

33

AJAX Suggest Example

Type-Ahead Searching

◆ Like Google Suggest

- » start typing in textbox and suggested words/phrases appear below

First Name:	<input type="text" value="jo"/>
Suggestions: John, Joe, Joseph	

<http://www.cs.odu.edu/~mweigle/cs312/ajax/suggest/suggest.html>

AJAX Suggest Example Components

First Name:
Suggestions: John, Joe, Joseph

- ◆ HTML
 - » provide form with a text box for user to enter a name
 - » provide named element to display suggestions
- ◆ JavaScript (client-side operations)
 - » `showHint()`
 - ❖ take data from HTML form and send to PHP script
 - » `doStuff()`
 - ❖ receive reply from PHP script
 - ❖ display result on webpage in named HTML element
- ◆ PHP script (server-side operations)
 - » create array of available names
 - » search array for those starting with data sent by JavaScript
 - » return the list of matching names

35

AJAX Suggest Example HTML Setup

First Name:
Suggestions: John, Joe, Joseph

- ◆ Load JavaScript helper functions
- ◆ Setup for embedded JavaScript (or the functions can be put into an external JavaScript file)

```
<head>
<title>Ajax Suggest Example</title>
<script type="text/javascript" src="http://www.cs.odu.edu/~mweigle/cs312/xml/
xmlHelperFns.js">
</script>

<script type="text/javascript">
<!--
    put showHint() and doStuff() functions here
// -->
</script>
</head>
```

36

AJAX Suggest Example HTML

First Name:

Suggestions: John, Joe, Joseph

- ◆ Provide form with a text box for user to enter a name
- ◆ Provide named element to display suggestions

```
<body>
```

```
<form action ="">
```

```
First Name:
```

this.value is the text currently in the box

```
<input type="text" id="txt1" onkeyup="showHint(this.value)" />
```

```
</form>
```

```
<p>Suggestions: <span id="txtHint"></span></p>
```

```
</body>
```

37

AJAX Suggest Example JavaScript

First Name:

Suggestions: John, Joe, Joseph

```
var xmlhttp = null; // must be outside function to be global
```

```
function showHint(str)
```

send data from HTML form to PHP script

```
{
```

```
    // check to see if textbox is blank
```

```
    if (str.length==0) {
```

```
        document.getElementById("txtHint").innerHTML="";
```

```
        return;
```

```
    }
```

```
    // build request to run PHP script on server
```

```
    var url = "gethint.php";
```

adds parameter 'q' to URL

```
    url = url + "?q=" + str;
```

adds random number to prevent using cached file

```
    url = url + "&sid=" + Math.random(); // avoid browser cache
```

```
    sendXHR(url);
```

sendXHR() in xmlHelperFns.js

```
}
```

38

AJAX Suggest Example

JavaScript

First Name:

Suggestions: John, Joe, Joseph

- ◆ Receive reply from PHP script
- ◆ Display result in named HTML element

```
function doStuff()
{
    document.getElementById("txtHint").innerHTML = xmlhttp.responseText;
}
```

responseText holds everything printed (echo) by the PHP script

39

AJAX Suggest Example

PHP (gethint.php)

First Name:

Suggestions: John, Joe, Joseph

```
<?php
// Fill up array with names
$names = array ("Anna", "Brittany", "Diana", ...);    rest of names omitted for space

// get the q parameter from the URL
$q = $_GET["q"];

// lookup all hints from array if length of q > 0
if (strlen ($q) > 0) {
    $hint = "";
    for ($i=0; $i<count($names); $i++) {
        if (strtolower($q) == strtolower(substr($names[$i], 0, strlen($q)))) {
            if ($hint == "") {
                $hint = $names[$i];
            } else {
                $hint = $hint . " , " . $names[$i];
            }
        }
    }
}
```

create array of available names

search for names starting with
data from JavaScript

- *count(\$arr)* returns number of elements in \$arr
- *strtolower(\$str)* returns the lowercase version of \$str
- *substr(\$str, \$start, \$end)* returns characters in positions \$start to \$end from \$str

40

AJAX Suggest Example

PHP (gethint.php *continued*)

First Name:

Suggestions: John, Joe, Joseph

```
// Set output to "no suggestion" if no hint was found
// or to correct value
if ($hint == "") {
    $response = "no suggestion";
} else {
    $response = $hint;
}

// output the response
echo $response;

return list of names

?>
```

<http://www.cs.odu.edu/~mweigle/cs312/ajax/suggest/suggest.html>

<http://www.cs.odu.edu/~mweigle/cs312/ajax/suggest/gethint.php.txt>

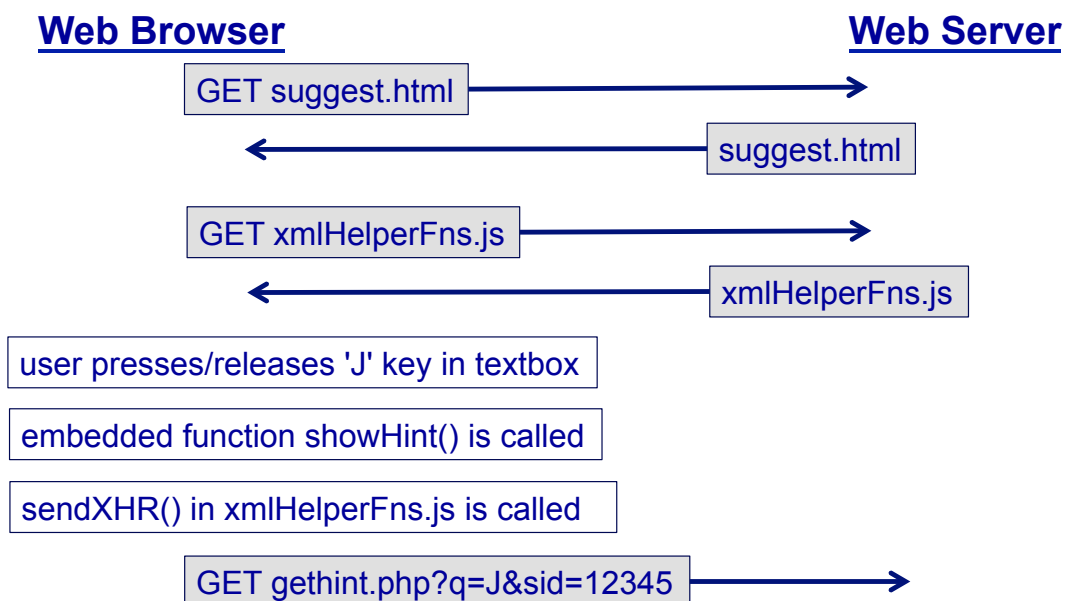
41

AJAX Suggest Example

Data Flow

First Name:

Suggestions: John, Joe, Joseph



42

AJAX Suggest Example

Data Flow (*continued*)

Web Browser

Web Server

First Name:

Suggestions: John, Joe, Joseph

gethint.php executes



Joe John Joseph

event handler stateChange() in xmlHelperFns.js is called

embedded function doStuff() is called

reply from server is written to the webpage

43

PHP and Ajax

Outline

◆ PHP



- » Syntax
- » Strings
- » Conditionals
- » Forms
- » Arrays
- » Loops
- » Functions
- » PHP and XML

◆ Ajax



- » Recalling XMLHttpRequest
- » Examples
 - ❖ Suggest
 - ❖ XML

44

AJAX XML Example

- ◆ Provide drop-down box with list of plants.
- ◆ When user selects a plant, display the information from an XML file for that plant.

Select a plant:

Plant info will be listed here.

Select a plant:

COMMON: Bloodroot
BOTANICAL: Sanguinaria canadensis
ZONE: 4
LIGHT: Mostly Shady
PRICE: \$2.44
AVAILABILITY: 031599

<http://www.cs.odu.edu/~mweigle/cs312/ajax/plants/plants.html>

Reference for this example: http://www.w3schools.com/php/php_ajax_xml.asp

45

AJAX XML Example Components

Select a plant:

Plant info will be listed here.

- ◆ HTML form
 - » provide drop-down box
 - » provide named element for data to be written to
- ◆ XML file
 - » plant_catalog.xml
- ◆ JavaScript (client-side operations, embedded in HTML)
 - » showPlant() and doStuff()
 - » very similar to previous example
- ◆ PHP (server-side operations)
 - » getPlant.php

Reference for this example: http://www.w3schools.com/php/php_ajax_xml.asp

46

AJAX XML Example

HTML Setup

Select a plant: Blue Phlox

Plant info will be listed here.

- ◆ Load JavaScript helper functions
- ◆ Setup for embedded JavaScript

```
<head>
<title>Ajax XML Example</title>
<script type="text/javascript" src="http://www.cs.odu.edu/~mweigle/cs312/xml/
xmlHelperFns.js">
</script>

<script type="text/javascript">
<!--
  put showPlant() and doStuff() functions here
// -->
</script>
</head>
```

47

AJAX XML Example

HTML

Select a plant: Blue Phlox

Plant info will be listed here.

- ◆ Provide form with a drop-down box
- ◆ Provide named element to display information

```
<body>
<form action="">
Select a plant:
<select name="plants" onchange="showPlant(this.value)">
<option value="Bloodroot">Bloodroot</option>
<option value="Jack-In-The-Pulpit">Jack-In-The-Pulpit</option>
<option value="Phlox, Blue">Blue Phlox</option>
<option value="Phlox, Woodland">Woodland Phlox</option>
<option value="Spring-Beauty">Spring-Beauty</option>
<option value="Trout Lily">Trout Lily</option>
<option value="Black-Eyed Susan">Black-Eyed Susan</option>
</select>
</form>

<div id="plantInfo"><strong>Plant info will be listed here.</strong></div>
```

48

AJAX XML Example JavaScript

Select a plant:

Plant info will be listed here.

```
var xmlhttp = null;
```

very similar to previous
example (suggest)

```
function showPlant(str)
{
    // build request to run PHP script on server
    var url = "getPlant.php";
    url = url + "?q=" + str;
    url = url + "&sid=" + Math.random(); // avoid browser cache

    // send request to PHP script
    sendXHR(url);
}
```

49

AJAX XML Example JavaScript

Select a plant:

Plant info will be listed here.

- ◆ Receive reply from PHP script
- ◆ Display result in named HTML element

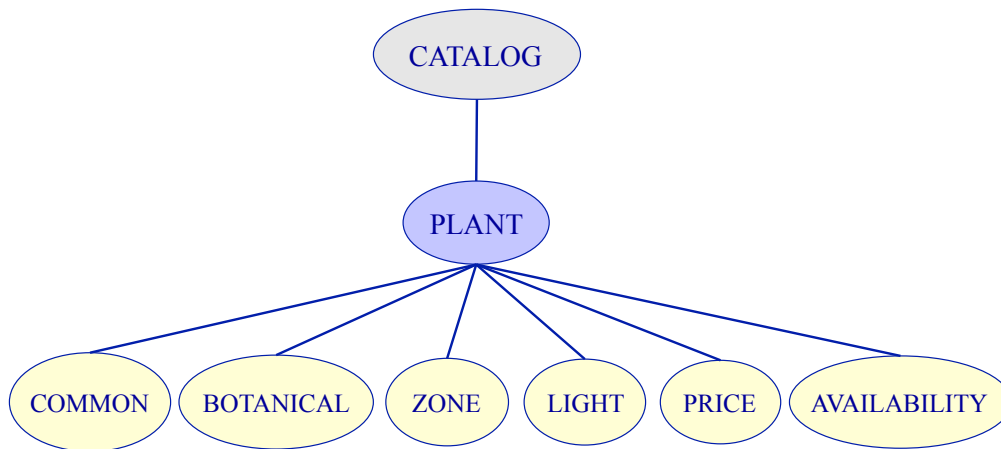
almost exactly the same as
previous example (suggest)

```
function doStuff()
{
    document.getElementById("plantInfo").innerHTML = xmlhttp.responseText;
}
```

50

AJAX XML Example

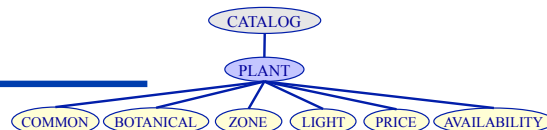
plant_catalog.xml



51

AJAX XML Example

getPlant.php



```
<?php
$q=$_GET["q"]; // get the parameters sent in the URL

// load the XML file
$xml = simplexml_load_file ("/home/mweigle/public_html/cs312/
xml/plant_catalog.xml");

// find the appropriate plant
$plant = $xml->xpath ("//PLANT [COMMON='$q']");

foreach ($plant[0]->children() as $child) {
    // print each child of the plant
    echo $child->getName() . ": " . $child . "<br />";
} http://www.cs.odu.edu/~mweigle/cs312/ajax/plants/plants.html
http://www.cs.odu.edu/~mweigle/cs312/ajax/plants/getPlant.php.txt
```

52