

# Connecting the Dots

---

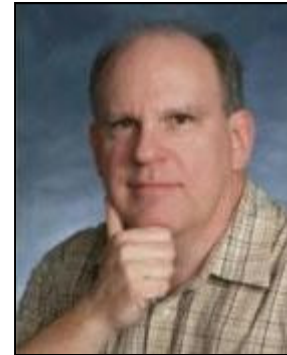
Building Web Applications with  
PHP, HTML, CSS, and JavaScript

**John Valance**  
division 1 systems  
johnv@div1sys.com

<div1>

[www.div1sys.com](http://www.div1sys.com)

# About John Valance



- Independent Consultant
  - Founder and CTO of Division 1 Systems
  - Specialty is helping IBM shops develop web applications and related skills
  - Training, mentoring, project management, consultation and coding
- 30+ years IBM midrange experience (S/38 thru IBM i)
- 12+ years of web development experience
  - Web scripting language of choice = PHP
- Frequent presenter on web development topics
- Trainer for Zend Technologies
  - Teaches Intro to PHP for RPG programmers
  - Zend Certified Engineer

# Goals of Presentation

- Introduce web development concepts to web beginner (experienced RPG programmer)
- Introduce major technical concepts and how components interact
- Introduce language syntax
- Show-and-tell demos and code examples (fun stuff)
- Prepare you for labs on HTML, CSS, PHP and JavaScript
- Come away with an idea of how to start

# What We Will Cover

- Overview of web application architecture
- HTML Basics
- PHP Basics
- Getting input from the browser
- Creating a database listing application
- Introduction to CSS
- Introduction to JavaScript

## Languages Involved in a PHP Database Application

### *Server side (IBM i):*

- PHP
- SQL (accessing DB2 tables)
- Possibly RPG & CL
  - Called via stored procedures or Zend Toolkit for IBMi

### *Client side (web browser):*

- HTML
- CSS
- JavaScript

# HTTP Request/Response Cycle

## *Non-PHP File*

### REQUEST:

- User types URL in browser
  - `http://www.mydomain.com/index.html`
- Browser connects to server and requests file
- Apache server listens for requests – default port 80

### RESPONSE:

- If found, Apache retrieves file from specified folder
  - Sub-folder of doc root, as configured in Apache
  - For Zend Server – doc root = `/www/zendsvr6/htdocs`
- Apache sends file back to browser

### Done!

- Connection is dropped

# HTTP Request/Response Cycle

## *PHP File*

- Client requests file `myApp.php` from web server
- Apache sees '.php' file request
- File is retrieved and handed to PHP processor
  - PHP file may *combine HTML with embedded PHP code.*
  - Embedded PHP code is executed, which *may retrieve information from database.*
  - PHP can merge database content with HTML
- Apache receives document (HTML) back from PHP
- Apache sends HTML back to browser
- **Done!**

# HTML / PHP

---



# HTML Basics

- HTML = tag-based language
  - Paired tags
    - `<HTML> content </HTML>`
    - Machine-readable code (unfortunately)
  - Hierarchical structure
  - Head and Body sections
    - `<head> ... </head>`
      - Information to the browser
      - Contents of `<head>` not visible to user
    - `<body> ... </body>`
      - This is the visible part of the page

# HTML Sample Structure

```
<html>
  <head>
    <title>Static Hello World</title>
  </head>

  <body>
    <h1>Hello, World Wide Web!</h1>
  </body>
</html>
```

# HTML Tags

Visual samples at:

[http://jvalance.com/webdemos/html\\_elements.html](http://jvalance.com/webdemos/html_elements.html)

Other important tags for web apps:

- Table tags:

- For formatting table of data (like subfile)

```
<table> <tr> <td>
```

- Form and input tags:

- For collecting input from user

```
<form action="someScript.php">
```

```
<input type="xxxxx" name="myVar">
```

# HTML Tables

**<table>** - Defines entire table

**<tr>** - One for each table row

**<td>** - One for each column in each row

Tables can be nested

```
<table>
  <tr>
    <td>Col 1</td> <td>Col 2</td> <td>Col 3</td>
  </tr>
  <tr>
    <td>Col 1</td> <td>Col 2</td> <td>Col 3</td>
  </tr>
</table>
```

# Some Features of PHP

- Scripting language
  - interpreted, not compiled
- Specialized for web applications
  - especially database oriented
- Free – open source
- Runs on server
  - typically embedded within HTML
  - end result is to generate HTML dynamically
- Procedural or Object Oriented coding
- Most widely used server-side web scripting language, worldwide!
- Key technology for IBM i

## More Features of PHP

- Dynamically typed variables
  - Eg: Variable can change from string to integer
- Very robust function set
- Arrays are very important
  - Many features of PHP implemented as arrays
  - Over 60 array handling functions
  - Associative arrays = character index
    - “key => value” lists
  - Multi-dimensional arrays
  - Mixed data types in one array

# Anatomy of a Request URL

`http://www.mydomain.com/pubapps/myScript.php?cust=10357`

<code>http://www.mydomain.com/</code>	Protocol // domain
<code>pubapps/</code>	Path to the script (relative to the web root folder)
<code>myScript.php</code>	Script file name
<code>?</code>	Delimiter (separates script name from the query string)
<code>cust=10357</code>	Query string (i.e. parameters the script can access)

# Query String - Multiple Parameters

Name/Value Pairs, Separated by '&'

```
script.php?name1=value1&name2=value2...
```

```
http://www.myComp.com/  
myScript.php?cust=12345&action=update
```

PHP parses query string into `$_GET` array

```
$custNo = $_GET['cust']; // 12345  
$action = $_GET['action']; // update
```



# Form Tag

```
<form action="myScript.php" method="post">  
    input tags...  
</form>
```

---

`<form>` - defines a group of input fields

Makes user input easier than typing query string in URL

- `action` attribute
  - tells what PHP script will receive input values
- `method` attribute
  - defines how values are delivered to action script
  - `method="get"` - send inputs on URL, as a query string
  - `method="post"` - send inputs separately
    - *Allows more data to be sent*
    - *Typically used when updating the server*

# Form Example

```
<form method="get" action="form_process.php">
  Enter your name:
  <input type="text" name="nameFld" value="John" />
  <br>
  <input type="submit">
</form>
```

Looks like this in browser:

Enter your name:

Clicking Submit button creates request for:

`/mydomain.com/form_process.php?nameFld=John`

```
<?php
  $name = $_REQUEST['nameFld'];
  echo "Hello $name! <br>";
?>
```

➔ PHP can read form inputs via the `$_REQUEST` array

# Input Tags

- Must be enclosed within `<form>` `</form>`
- Name attribute is used in PHP to access the value
- `Type="submit"` creates a button to submit the form

```
<form name="form1" action="myScript.php" method="post">
  <input type="text" name="myText">
  <input type="radio" name="myRadio">
  <input type="checkbox" name="myCheckBox">
  <select name="myDropDown">
    <option value="A">Option A</option>
    <option value="A">Option A</option>
  </select>
  <input type="submit" value="Submit Form">
</form>
```

# PHP Database Access

*List all records from DB table*

```
$conn = db2_connect ( "*LOCAL", "PHPUSER", "PSWD1" );

$query = "SELECT * FROM PHPTEST.MEMBERSHIP";
$stmt = db2_prepare( $conn, $query );
db2_execute( $stmt );

while ( $row = db2_fetch_assoc( $stmt ) ) {
    $memberId = $row['MEMBERID'];
    $name = "{$row['FIRST_NAME']} {$row['LAST_NAME']}";
    echo "Member ID $memberId; $name<br>";
}

db2_close ( $conn );
```

# CSS

---

# Styling with CSS

- CSS = Cascading Style Sheets
- Extension to HTML as of HTML v 4
- Allows fine-grained control of visual elements on a page
- Simple, intuitive syntax

# CSS Syntax

```
selector {  
    property: value;  
    property: value;  
    ...  
}
```

- **Selector**: identifies a part of the document to be styled  
HTML tag name, Class name, or a Unique ID
- **Property**: A specific presentation attribute to be styled  
color, font-weight, border attributes, visibility
- **Value**: How the presentation attribute should be styled  
`color: red;`  
`font-weight: bold;`  
`border: 2px solid blue;`

# Examples of CSS Selectors

- HTML Tag Name:

```
CSS: BODY { font: arial; font-size: 12pt; color: navy }
```

- *Can use any HTML tag name*
- *Applies to all occurrences of the tag throughout a document*

- Class Name - precede with period (.):

```
CSS: .error { color: red; font-weight: bold}
```

```
HTML: <p class="error">Invalid email address</p>
```

- *Can specify the same class on many different HTML tags*

- Unique ID – precede with hash (#):

```
CSS: #shipto { visibility: hidden }
```

```
HTML: <div id="shipto"> <table>... </div>
```

- *ID name should only occur once in HTML document*



# CSS Style Sheet Example

```
1 body {
2     font-family: arial, verdana, sans-serif;
3     font-size: 12pt;
4 }
5
6 h1, h2, h3 {
7     color: #2E529C;
8     font-family: verdana;
9 }
10 .error {
11     color: red;
12     background-color: yellow
13 }
14 p.big {
15     font-size: 16pt;
16 }
```

# Where Can Styles Be Defined?

- Inside a single HTML element

```
<table style="border:none; color:blue">
```

- Inside the <head> element of an HTML page

```
<head>
```

```
  <style type="text/css">
```

```
    table { border:none; color:blue }
```

```
  </style>
```

```
</head>
```

- In an external CSS file

```
<head>
```

```
  <link rel="stylesheet" type="text/css"
```

```
    href="siteStyle.css" />
```

```
</head>
```

# JAVASCRIPT

---

# What is JavaScript?

- It isn't Java! (but similar syntax, based on C).
- Runs on the client-side (usually) i.e. in browser
- Scripting language for web browsers
- All browsers have built-in JavaScript interpreter – you don't buy it or install it.
- Interpreted at run-time (as page loads)
- JavaScript code is downloaded with the HTML document, but only runs in the browser.

# JavaScript Sample

```
<html>
<head>
<title>JavaScript Example</title>

<script>
  function checkCustNo() {
    if (document.myForm.custNo.value == '') {
      alert('Customer number is required.');
```

```
    } else {
```

```
      alert('Customer No. entered was: ' +
        document.myForm.custNo.value);
```

```
    }
```

```
  }
```

```
</script>
```

```
</head>
```

# What Can JavaScript Do?

- Validate input data
- Handle events
  - e.g.: mouse clicks or cursor movement into/out of fields
- Control Dynamic HTML
  - make things move around, appear and disappear
- Read and alter document elements, including HTML tags and CSS attributes
- Open & close windows, and communicate between windows.
- *Key technology in Ajax and Web 2.0 applications*

# Where Is JavaScript Coded in HTML?

- Can be inserted just about anywhere, but must be enclosed in `<script> </script>` tag
- Typically, functions are defined in `<head>` section.
- Can also be included as external file
  - Function libraries, Frameworks
  - Linked to document in `<head>` section
- Can also be included as action in certain HTML tags:

```
<form action="checkInputs();">
```

```
<button onclick="alert('You clicked me.')">
```

```
<a href="javascript:openHelpWindow();">
```

# CURRENT STATE OF WEB DEVELOPMENT

---



## Present/Future State of Web Development

- Mobile is King
  - HTML 5 / CSS 3
- Responsive Design – multiple devices
- Ajax – Asynchronous JavaScript and XML
- Service Oriented Architecture (SOA)
- Application control
  - Shift from server side (PHP) to Client side (JavaScript)
  - Single-page Apps

# Frameworks

- Collection of predefined components for working in a specialized programming domain
- Object Oriented (PHP/JavaScript)
- Simplify web development
  - Don't reinvent the wheel
- Allow greater integration between systems
- Build upon solid foundation
  - Inherit from existing classes and extend

# Examples of Web Frameworks

- PHP: Zend Framework (ZF2 = current)
  - MVC = Model / View / Controller
    - complex set of classes that work together
    - manages relationship between data model and user interactions
  - ZF also includes many useful classes that are easy to use
    - db / authentication / mail / xml / ldap / log / Etc....
- JavaScript: jQuery
  - select document sections to manipulate using CSS selectors
  - **greatly** simplifies JavaScript development
- CSS: Twitter Bootstrap
  - provides a set of css classes to give a clean, modern design
  - handles responsive design for mobile devices

## Summary - Elements of Web App Development

- HTTP protocol / Stateless nature of web requests
- Client-Side (browser) technologies
  - HTML: basic markup and document structure
  - CSS: styling of document elements
  - JavaScript: make web pages dynamic
- Server-Side technologies
  - PHP : dynamically generate document contents based on elements of the request
  - SQL : retrieve / update DB (merge DB content with HTML markup)

## More Information

- Examples used in this presentation available at:  
<http://jvalance.com/webdemos/>
- Email me if you would like the source code:  
[johnv@div1sys.com](mailto:johnv@div1sys.com)
- Attend the hands-on Labs for more details!

*Thank you!*

# Contact Info

- John Valance  
division 1 systems  
106 Main St., Burlington VT 05401
- [johnv@div1sys.com](mailto:johnv@div1sys.com)
- 802-355-4024
- <http://www.div1sys.com>

