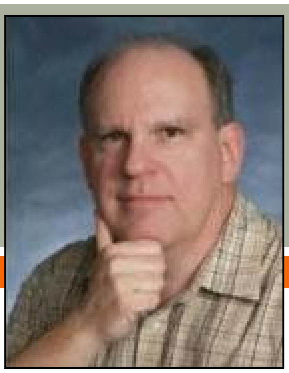
PHP - Beyond the Basics

John Valance

JValance Consulting
johnv@jvalance.com

About John Valance



- Independent consultant
  - Specialty is helping iSeries shops develop web applications, and related skills
  - Training, mentoring, consultation and coding
- 25+ years iSeries/AS400 experience
- 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

Goals of This Presentation

- Cover tips and techniques useful to development of business applications in PHP on System i

- Topics:
  - Database paging
  - File system processing / Writing CSV Content
  - Email
  - Session management / Cookies

Assumptions

- You understand
  - Basic PHP syntax
  - Basics of web application coding in PHP
  - Have done some basic DB2 applications with PHP
- You are ready to go a little deeper
  - Pick up a few ideas and “how-to” tips

Database Paging

Database Paging

- Large Result Sets
  - Don’t load all records on screen at once
  - Show subset page by page
    - i.e. 20 to 50 records per page
- Similar to Subfile
  - But techniques very different, due to HTTP
- Need mechanism / algorithm to allow user-controlled page access

Database Paging

- Need to use a scrollable cursor
  - Option on db2_prepare(): `'cursor' => DB2_SCROLLABLE`
  - This allows us to **read a specific row** number in result set
  - High performance record-level access
- Compute starting row number to retrieve as:

  $rowNum = (PAGE_SIZE * $pageNum) - PAGE_SIZE + 1;
- Retrieve $pageNum from HTML form field
  - If not present, default to page 1
- Add buttons on screen for Next, Previous
  - These will request $page+1, $page-1

Database Paging

Additional Features

- On first page, disable the “Previous” button

```
if ($pageNum == 1) $prevState = "disabled";
```

- On last page, disable the “Next” button
  - After loading page, try to retrieve next record
  - If no more, disable the “Next” button

```
if (!$row = db2_fetch_assoc($stmt, $rowNum))
   $nextState = "disabled";
```

- In HTML, echo $nextState and $prevState within the button tags

Database Paging

Even Fancier…

- ✓ Show Page X of Y
- ✓ Add buttons for “First Page”, “Last Page”

- First Page button is easy…
  - Just request page = 1 when button clicked
  - Disable the button if current page=1
    - Can use **`$prevState`** we calculated earlier
- For X of Y, and last page, we need to know how many pages there are in query result
  - Note: Disable the last page button if currently on last page
    - Can use **`$nextState`** we calculated earlier

Database Paging

Page X of Y and Last Page Link

- We need to know how many pages there are in query result

- We need to do a separate SELECT to count number of rows in result

```
SELECT COUNT(*) as ROW_COUNT
FROM <same table> JOIN <same joins>
WHERE <same conditions>
```

- We can then calculate total pages as:

```
$numberOfPages =
    ceil( (int)$row['ROW_COUNT'] / PAGE_SIZE);
```

- Use ceil() function to round page number up.

IFS Files and CSV Processing

File Processing

- By file, we mean IFS files, not object type *FILE.
  - E.g., text files, PDF, images, Excel, CSV
- PHP core includes numerous file system functions
  - http://us2.php.net/manual/en/ref.filesystem.php
- Two types of file functions:
  - File name-based
    - Dealing with the file as a whole
    - Receive a file name as parameter
  - Resource-based
    - Open a file to read/write portions of the file’s contents
    - Receive a file “handle” resource as input

File Name-based Functions

- Some File Name-based Functions
  - bool **file_exists** ($filename)
  - string **file_get_contents** ($filename)
    - file_get_contents can use a URL as $filename
    - If it contains special characters, encode it with urlencode()
  - int **file_put_contents** ($filename , $data)
  - int **file_size** ($filename)
  - bool **copy** ($source , $dest)
  - bool **unlink** ($filename)
- If no path specified, file is searched for in same folder as php script
  - Some functions allow option to use include_path to search for files

File System Functions
Resouce-based File Handling

- First, open the file using fopen() function
  - fopen() returns a file-handle resource

```
$fh = fopen(‘myfile.txt’, ‘a+’);
```

- Use resource returned by fopen() to call functions to read and write data

```
$text = fread($fh, file_size(‘myfile.txt’));
fwrite($fh, “some new text\n”);
```

- Lastly, close the file resource using fclose()

```
fclose($fh);
```

fopen() and Resource-based functions

- resource **fopen** (string $filename , string $mode)
  - $mode is open mode for file processing:
    - See table on next slide for fopen mode options

- Some resource-based functions
  - string **fread** (resource $handle , int $length)
  - int **fwrite** (resource $handle , string $string [, int $length])
  - int **fseek** (resource $handle , int $offset [, int $whence])
  - bool **feof** (resource $handle)
  - bool **fclose** (resource $handle)

fopen() mode flags

Mode	Function	Read	Write	Pointer	Overwrite	Create
r	open	Yes	No	begin	No	No
r+	open	Yes	Yes	begin	No	No
w	open	No	Yes	begin	Yes	Yes
w+	open	Yes	Yes	begin	Yes	Yes
a	open	No	Yes	end	No	Yes
a+	open	Yes	Yes	end	No	Yes
x	create	No	Yes	begin	No	No
x+	create	Yes	Yes	begin	No	No
c	open/create	No	Yes	begin	No	Yes
c+	open/create	Yes	Yes	begin	No	Yes
b	binary file	Specify b along with above mode flags if binary file				

CSV file handling

- PHP has built-in CSV handling functions
  - Resource-based functions – need to use fopen() to get $handle
- int **fputcsv** (resource $handle , array $fields)
  - Parses the array $fields into a comma-separated string
  - Write the string to the end of the file denoted by $handle
    - Strings automatically quoted if contain blanks or commas
    - Includes newline at end of CSV string
  - Returns number of bytes written
- array **fgetcsv** (resource $handle)
  - Reads one line from $handle
  - Parses CSV content and returns an array containing an element for each value in the CSV string
  - Advances the file pointer to next line for looping

Building CSV files from DB2 content

- Easy to create a CSV file from an SQL query
- Use db2_fetch_array()
  - Returns an array of field values based on an SQL query
- Pass returned array to fputcsv()

```
$conn = db2_connect ( "*LOCAL", "USER", "PSWD" );
$stmt = db2_prepare( $conn, "SELECT * FROM MYTABLE" );
db2_execute( $stmt );
$fh = fopen('mytable.csv', 'w');
while ( $row = db2_fetch_array( $stmt ) ) {
    fputcsv($fh, $row);
}
fclose($fh);
db2_close($conn);
```

Adding Column Headings to CSV File

- Use db2_num_fields() and db2_field_name() functions

- Add the following before reading/writing data rows:

```
for ($col = 0; $col < db2_num_fields($stmt); $col++)
    $headings[] = db2_field_name( $stmt, $col );
fputcsv($handle, $headings); // first line of CSV file
```

- If cryptic DDS field names, use ‘AS’ in SELECT

  -
    ```
    SELECT CSCNUM as “Customer Number”,
           CSNAME as “Customer Name”
    ```

Delivering a File to the Browser

- Instead of writing to IFS, send it to user
  - User will get a “File open/save” dialog

- We can access the PHP output stream as a file resource
  - php://output – Use this as filename in fopen()
  - Specify mode = ‘w’ (write)

```
$handle = fopen("php://output", 'w');
```

- Need to do two other things:
  - Buffer output
    - Want to deliver the file all at once
  - Specify content type and file name
    - Use header() function to set values in HTTP headers sent to browser

Delivering a File to the Browser

```
ob_start(); // start output buffering
// set file type and name in HTTP header
header("Content-type: application/csv;");
header('Content-Disposition: attachment;
       filename="membership.csv"');
... do db2 query execute
$handle = fopen("php://output", 'w');
... write content to $handle as before
... after db2_close() and fclose():
// Flush output buffer - send entire file to browser
ob_end_flush();
```

Sending Email

Sending Email

- PHP mail() function
  - Built-in to PHP core
  - Simple, easy to use
  - Best suited for text-only messages

bool **mail** (string $to , string $subject , string $message [, string $headers])

Example:

```
$to      = 'customer@gmail.com';
$subject = 'Test Email';
$message = 'Testing 1,2,3';
$headers = "From: custserv@ourcompany.com\r\n" .
           "Reply-To: custserv@ourcompany.com\r\n";
mail($to, $subject, $message, $headers);
```

- SMTP server/port is set in php.ini

Adding Email Attachments

- Underlying protocols are complex
  - based on RFC822
- Can be done with `mail()` function, but not easy
  - Requires understanding MIME formats
    - MIME = Multipurpose Internet Mail Extensions
    - http://en.wikipedia.org/wiki/MIME
- Best to use a package that makes it simple
  - PEAR::Mail_Mime
    - http://pear.php.net/package/Mail_Mime
  - Zend Framework: Zend_Mail class
    - http://framework.zend.com/manual/1.11/en/zend.mail.html
    - Zend Framework included with Zend Server (even CE)
    - Very simple interface
    - Great integration with other Zend products

Using Zend_Mail

- To use Zend Framework classes in your code, add these two lines at top of your script:

```
require_once 'Zend/Loader/Autoloader.php';
Zend_Loader_Autoloader::getInstance();
```

Note: path to Zend Framework library folder is already set in your include path by ZS installation

- Example – sending plain text message:

```
$mail = new Zend_Mail();
$mail->setFrom('jvalance@sprynet.com', 'Our Company');
$mail->addTo('jvalance@sprynet.com', 'J. Valance');
$mail->addTo('john.valance@gmail.com', 'John V.');
$mail->setSubject('Test Order Confirm');
$mail->setBodyText('This is to confirm your recent order...');
$mail->send();
```

Add an Attachment

```
$pdf = file_get_contents('some_pdf_file.pdf');
$attach = $mail->createAttachment(
                    $pdf,
                    'application/pdf',
                    Zend_Mime::DISPOSITION_ATTACHMENT,
                    Zend_Mime::ENCODING_BASE64
              );
$attach->filename = 'brochure.pdf';
```

- Note: `application/pdf` = Content-type
  - Tells email client what program to open attachment with
  - *Other examples:*
    - `application/csv` (Excel most likely)
    - `img/jpg`

HTML-formatted Emails

- Use **`$mail->setBodyHtml('…html content…')`**
- Should also **`setBodyText('…text content…')`** for recipients that only receive text
- HTML emails can be tricky…
  - Some email clients don't handle them well / the same
    - Web-based clients
    - PDAs / Smart-phones
  - A lot of companies stick to plain text notification emails
  - Rules of thumb for successful HTML emails:
    - Use <table>s for layout (vs. CSS positioning etc.)
    - Specify CSS attributes inline, vs, style sheet
      - i.e. - as <tag style="…"> attribute, no matter how redundant
  - Images in HTML:
    - better to use external file references for images (vs. image attachments)
    - i.e. <img src=http://www.mycompany.com/logo.gif>

Session Management and Cookies

Session Management

- HTTP protocol is stateless
  - There is no continuous connection to server
  - Each request/response is completely independent of the next
- Web applications need a mechanism to simulate a user session
- PHP makes this easy with session functions and session variables
- Session variables are stored on the server by PHP
  - Session variables are keyed by a session ID
  - Session variables are accessed via the $_SESSION array
- Session ID is stored in a cookie on the client
  - This is triggered by PHP's session_start() function
  - Cookies are automatically sent with request by browser

Session Management

- Login Script:

```
session_start(); // must happen before any output
… validate user/pswd
$_SESSION['userid'] = $_POST['userid'];
… other processing
```

- Application scripts include this at top:

```
session_start();
if (! isset($_SESSION['userid'])
    header('Location: login.php'); // redirect to login
    exit;  // always exit after redirect
else
    echo “Hello “ . $_SESSION['userid'];
```

- Logout:

```
session_start();
session_destroy();
setcookie(session_name(),'',0,'/'); // expire cookie
header('Location: login.php'); // redirect to login
exit; // always exit after redirect
```

Session Persistance

Session variables persist until one of these happens:

- session_destroy() is called
- browser windows are all closed
- session cookie times out
  - based on **session.cookie_lifetime** in php.ini
  - this is unreliable – better to code your own session timeout logic
- session garbage collection takes place
  - based on **session.gc_maxlifetime** in php.ini

Session Timeout Handling

- Session timeout logic

```
if (isset($_SESSION['LAST_ACTIVITY'])
&& (time() - $_SESSION['LAST_ACTIVITY'] > 1800)) {
    // last request was more than 30 minutes ago
    session_destroy(); // destroy session data
    session_unset();   // unset $_SESSION vars

    // expire cookie

    setcookie(session_name(),'',0,'/');
} else {

   // update last activity time stamp
   $_SESSION['LAST_ACTIVITY'] = time();

}
```

Cookies

- To persist user information beyond a session, set a cookie

```
setcookie(
    'mycookie',              // name
    'Oreo',                  // value
    time()+(60*60*24*30),    // expire in 30 days
);
```

- Must call setcookie() before any browser output
  - because cookies are set via response headers
- Retrieve value via $_COOKIES array:

```
$cookieValue = $_COOKIES['mycookie'];   // 'Oreo'
```

Summary

Summary

- Database paging
  - Scrollable cursor
  - Specify starting row on db2_fetch
- File system processing / Writing CSV Content
  - File-based functions vs. Resource-based functions
  - fputcsv() and fgetcsv()
  - $download = fopen("php://output", 'w');
    - Use buffering
    - Specifying file name and type with header() function

Summary

- Email
  - mail() – simple emails without attachments
  - Zend_Mail – attachments / HTML
- Session management / Cookies
  - HTTP = stateless protocol
  - session_start() / session_destroy()
  - $_SESSION array
  - setcookie() – persist information beyond session

More Information

- John Valance
    - JValance Consulting
    - jvalance@sprynet.com
    - 802-355-4024
- Contact me for source code