How RPG Programmers Can Leverage PHP Arrays

Anne Rue
Regional Account Manager
Anne.rue@zend.com
(310) 480 4161

Mike Pavlak
Solutions Consultant
Mike.p@zend.com
(815) 722 3454
Agenda

- Introduce arrays in PHP
- Review RPG arrays
- Compare RPG and PHP array concepts
- More functions for arrays in PHP
- Q&A
Why are we talking about arrays?

- Fastest method for manipulating ordered sets
- Highly leveraged in PHP development
- PHP developers take them for granted
- Available in RPG but long neglected
- Gap that needs to be closed
- Array defined:

  ...a data structure consisting of a group of elements that are accessed by indexing
PHP Array Examples
Data Type Review: 8 Data Types

- **Scalar**
  - String: “the quick brown fox...”, ‘123456’
  - Integer: 860, -9, 57009
  - Floating point: 19.99, 29.99, 3.1412
  - Boolean: true, false

- **Compound**
  - Object: OOP

- **Special**
  - Resource: Handle
  - Null: Something that not nothing (empty set)
Three types of arrays

- **Enumerated**
  - Simple list
  
  ```php
  $arrayone = array("Scooby", "Shaggy", "Daphne", "Fred", "Velma");
  ```

- **Associative**
  - Custom key
  
  ```php
  $arraytwo = array(  
      Cartoon1=>'Scooby',  
      Cartoon2=>'Shaggy',  
      Cartoon3=>'Daphne',  
      Cartoon4=>'Fred',  
      Cartoon5=>'Velma'  
  );
  ```

- **Multidimensional**
  - Array of arrays
  
  ```php
  $arraythree = array(  
      array("Scooby", "Shaggy", "Daphne", "Fred", "Velma"),  
      array("Bugs", "Daffy", "Tweety", "Elmer", "Foghorn")  
  );
  ```
Enumerated array

Code:

```php
$arrayone = array('Scooby', 'Shaggy', 'Daphne', 'Fred', 'Velma');
echo "<BR> <BR> Array one: "; print_r($arrayone);
```

Output:

Associative array

Code:

```php
$arraytwo = array(
    Cartoon1 => 'Scooby',
    Cartoon2 => 'Shaggy',
    Cartoon3 => 'Daphne',
    Cartoon4 => 'Fred',
    Cartoon5 => 'Velma'
);

echo "<BR> <BR> Array two: "; print_r($arraytwo);
```

Output:


If you have trouble, think CL command parameters: Keyword & Values!!!
Multidimensional array

Code:

```php
$arraythree = array(
    array('Scooby', 'Shaggy', 'Daphne', 'Fred', 'Velma'),
    array('Bugs', 'Daffy', 'Tweety', 'Elmer', 'Foghorn')
);

echo "<BR> <BR> Array three: ";print_r($arraythree);
```

Output:

```
```
Adding elements & growing the array

- PHP Arrays are dynamic
- Can be sized on the fly, no need to recompile
- Example adding element:

```php
$arrayone = array('Anne', 'Mark', 'Sabrina', 'Katie', 'Rick');
$e1 = array
$arrayone[] = 'Joe';
$e1 = array
```

```
```

```
```
Removing elements & reducing the array

- `array_pop` removes element from the end
- `unset` removes an element you specify (or entire array!)

```php
$arrayone = array('Anne', 'Mark', 'Sabrina', 'Katie', 'Rick');
echo "<BR> <BR> Array one: "; print_r($arrayone);
array_pop($arrayone);
echo "<BR> <BR> Array one: "; print_r($arrayone);
unset($arrayone[2]);
echo "<BR> <BR> Array one: "; print_r($arrayone);
```


Trivia points

• Really only one type of array, associative
• Data content is non-restrictive, any data types
• Each element can be different
• Array sizes change dynamically
• Supports no known limit of dimensions
  ▪ Memory
  ▪ Humans like 2 or 3 (Think spreadsheet and workbook)
• Used heavily in i/o
• Both keys and content can be dynamic
• Index starts at zero while RPG starts at one
Arrays

- Introduction
- Installing/Configuring
  - Requirements
  - Installation
  - Runtime Configuration
  - Resource Types
- Predefined Constants
- Array Functions
Review RPG Arrays
In the beginning...

- Indicators were the only ordered set
  - Original RPG and RPG II

<table>
<thead>
<tr>
<th>Name</th>
<th>Indicators</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numbered</td>
<td>*IN01-*IN99</td>
<td>Gen purpose</td>
</tr>
<tr>
<td>Command Key</td>
<td>*INKA - *INKY</td>
<td>No “O”</td>
</tr>
<tr>
<td>Halt</td>
<td>H1-H9</td>
<td>Error recovery</td>
</tr>
<tr>
<td>Matching</td>
<td>M1-M9, MR</td>
<td>Matching records</td>
</tr>
<tr>
<td>Control</td>
<td>L1-L9</td>
<td>Level Breaks</td>
</tr>
<tr>
<td>External</td>
<td>U1-U8</td>
<td>Switches</td>
</tr>
<tr>
<td>Cycle</td>
<td>1P, LR, OA-OG, OV</td>
<td>Printing</td>
</tr>
</tbody>
</table>
And then…

- RPG II - Then came simple arrays.
  - Predefined length
  - Single variable data type
  - Built in E-specs

- Op Codes
  - XFOOT – Summing array
  - MOVEA – Move data (Still most extremely powerful)
  - LOKUP – Search the array
  - SORTA – Gee, I wonder what this does?

- Seems like things paused here for a while
Today…

• Compile time tables
  • Great for static content
  • Defined below “O” specs
  • Two dimensional in nature

• RPG III – Multiple Occurrence Data Structure (MODS)
  • Two dimensional feel
  • Still a little clunky

• RPG IV – More Power!
  • V5R1 – BIF’s: %LOOKUP, %LOOKUPGT, etc.
  • V5R2 – DIM for Data Structures; MODS on Steroids!
  • V5R3 – %SUBARR is an attempt at dynamic sizing
  • V5R4 – XML processing
  • i6.1 – DIM up to 16,773,104
How PHP matches up to RPG
Array shootout

- Base functions
  - RPG has about a dozen op-codes and BIF’s (Variations on BIF’s)
  - Many op-codes can manipulate array content
  - PHP has 75 functions [www.php.net/array](http://www.php.net/array)

- Size
  - RPG has limits, 16,773,104 as if i6.1
  - PHP has no practical limits, No “array index overflow” error
  - RPG array must be defined, PHP grows dynamically

- Type
  - RPG uses static typing (one type, one length)
  - PHP is dynamically typed (Each element can be different)
Simple Array Search (Lookup)

```
$arrayone = array('Scooby', 'Shaggy', 'Daphne', 'Fred', 'Velma');

RPG

D ARRAYONE S 8 DIM(5)
D X S 1 0 INZ
/FREE

X=%LOOKUP('DAPHNE':ARRAYONE);

/END-FREE

I found her in position==> 3

PHP

$x= array_search('Daphne', $arrayone);

echo "<BR> <BR> I found her in position==> $x";

I found her in position==> 2
```
Simple traverse

```php
$arrayone = array('Scooby', 'Shaggy', 'Daphne', 'Fred', 'Velma');

for X=1 to (%ELEM(ARRAYONE));
    except REC1;
    endfor;
```

RPG

```
foreach ($arrayone as $key => $x) {
    echo "<BR>$x is the index value " . $key;
}
```

PHP

```
Scooby is the index value 0
Shaggy is the index value 1
Daphne is the index value 2
Fred is the index value 3
Velma is the index value 4
```
# RPG to PHP function map

<table>
<thead>
<tr>
<th>Function</th>
<th>RPG</th>
<th>PHP</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search</td>
<td>%LOOKUP</td>
<td>array_search</td>
<td></td>
</tr>
<tr>
<td>Sum</td>
<td>%XFOOT</td>
<td>array_sum</td>
<td>Array_prod can multiply</td>
</tr>
<tr>
<td>Get portion</td>
<td>%SUBARR</td>
<td>array_slice</td>
<td>Substring an array by chunks</td>
</tr>
<tr>
<td>Sort</td>
<td>SORTA</td>
<td>asort, arsort</td>
<td>PHP sequence dynamic</td>
</tr>
<tr>
<td>Move</td>
<td>MOVEA</td>
<td>array_slice</td>
<td>Substring by character</td>
</tr>
<tr>
<td>Count</td>
<td>%ELEM</td>
<td>count</td>
<td>Get number of elements</td>
</tr>
</tbody>
</table>
More functions in PHP
Interesting functions

• How to move around the array
• Randomize contents
• Array housekeeping
• Move array elements to variables
• Sort two or more arrays at once
• Execute a function on each element with no loop!
• Data file example
Navigate the array...Thanks Jon!

```php
$myArray = array("Car", "Truck","Motorcycle","Plane","Bicycle");
echo "<p>Here's the complete array </p>";
print_r($myArray);
echo "<br>

echo "Current: ", current($myArray), "<br>
echo "Next: ", next($myArray), "<br>
echo "One after that: ", next($myArray), "<br>
echo "Prev: ", prev($myArray), "<br>
echo "End: ", end($myArray), "<br>
reset($myArray);
```

Here's the complete array


Current: Car
Next: Truck
One after that: Motorcycle
Prev: Truck
End: Bicycle
Resetting the array.
Current: Car
Mix it up with a shuffle

```php
$arrayone = array('Scooby', 'Shaggy', 'Daphne', 'Fred', 'Velma');
echo "<BR> <BR> Array one: "; print_r($arrayone);
shuffle($arrayone);
echo "<BR> <BR> Array one: "; print_r($arrayone);
shuffle($arrayone);
echo "<BR> <BR> Array one: "; print_r($arrayone);
```

```html
```
Consolidate, clean and sort arrays

$arrayone = array('Scooby', 'Shaggy', 'Daphne', 'Fred', 'Velma', 'Jerry');
$arraytwo = array('Bugs', 'Daffy', 'Tweety', 'Elmer', 'Foghorn', 'Jerry');
$both = array_merge($arrayone, $arraytwo);
$both = array_unique($both);
$both = asort($both);
$arrayone = array(0 => 'Scooby', 1 => 'Shaggy', 2 => 'Daphne', 3 => 'Fred', 4 => 'Velma', 5 => 'Jerry');
$arraytwo = array(0 => 'Bugs', 1 => 'Daffy', 2 => 'Tweety', 3 => 'Elmer', 4 => 'Foghorn', 5 => 'Jerry');
$both = array_merge($arrayone, $arraytwo);
$both = array_unique($both);
$both = asort($both);


Sort Multiple Arrays at once!

```php
$arrayone = array('Dog', 'Cat', 'Chicken', 'Dolphin');
echo "<BR> <BR> Array one: "; print_r($arrayone);
$arraytwo = array('Puppy', 'Kitten', 'Chick', 'Pup');
echo "<BR> <BR> Array two: "; print_r($arraytwo);
$arraythree = array('Litter', 'Kindle', 'Peep', 'Pod');
echo "<BR> <BR> Array two: "; print_r($arraythree);
array_multisort($arrayone,$arraytwo,$arraythree);
echo "<BR> <BR> Array one: "; print_r($arrayone);
echo "<BR> <BR> Array one: "; print_r($arraytwo);
echo "<BR> <BR> Array one: "; print_r($arraythree);
```


$arrayone = array('Dog', 'Cat', 'Chicken', 'Dolphin');
echo "<BR> <BR> Array one: "; print_r($arrayone);
 strtoupper($arrayone);
echo "<BR> <BR> Array one: "; print_r($arrayone);

function upperCase(&$animal, $key) {
    $animal= strtoupper($animal);
}
array_walk($arrayone, upperCase);
echo "<BR> <BR> Array one: "; print_r($arrayone);
Get data from a file

while($row=db2_fetch_array($stmt)) {
    print_r($row); echo "<BR>";
    list( $CUSTOMER_NUMBER, $CUSTOMER_NAME, $CUSTOMER_STATE)= $row;
    echo("<TR><TD> $CUSTOMER_NUMBER</TD> <TD>$CUSTOMER_NAME </TD>
        <TD> $CUSTOMER_STATE</TD>" );
}

- Loop through data
- List function copies to variables
- Implicit copy, be careful
- Arrays in PHP like Data Structures in RPG: The workhorse of data manipulation!

<table>
<thead>
<tr>
<th>Customer Number</th>
<th>Customer Name</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Jimmy Buffet</td>
<td>IL</td>
</tr>
<tr>
<td>2</td>
<td>Sherlock Holmes</td>
<td>EN</td>
</tr>
<tr>
<td>3</td>
<td>Gregory House</td>
<td>NJ</td>
</tr>
<tr>
<td>4</td>
<td>Dexter Morgan</td>
<td>FL</td>
</tr>
<tr>
<td>5</td>
<td>Mary Shannon</td>
<td>NM</td>
</tr>
</tbody>
</table>
Kevin Schroeder from Zend’s Global Services Group with Jeff Olen, co-author of…

Get yours at MCPressOnline or at fine bookstores everywhere
Thank you!!

Questions?
Anne.rue@zend.com
(310) 480 4161
Mike.pavlak@zend.com