

TUG

TEC 20i8

“TEC it to the limit”

TEC 25th edition -- Wednesday May 30 & Thursday May 31, 2018

FRESCHHE SOLUTIONS

30+ PHP Tips in 60 Minutes

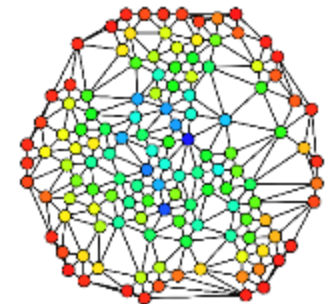


Tip 1 – RTFM – php.net

- Don't waste money on books that summarize functions
 - ▶ Becomes out of date QUICKLY
 - ▶ Online manual is free!
 - ▶ Books worth buying:
 - Cookbooks
 - Design patterns
 - Advanced PHP, etc
- Open Source community “assumes” you have done this
 - ▶ Good questions get answered quickly
 - ▶ Stupid questions tend to get ignored or ridiculed
- Install a browser you rarely use (like IE) and set initial page to php.net
- DB2 Extension is documented quite extensively here

Tip #2 – Keep Current

- PHP is NOT RPG.
 - ▶ Only RPG supports 40 years of backward compatibility.
- PHP needs updating
 - ▶ Security patches (www)
 - ▶ New features
 - ▶ Deprecated functions
- Long Term Support comes to IBM i community FOR FREE!!!
- PHP is an “edge” technology and these require frequent updating



Tip #3 – Master Web Development

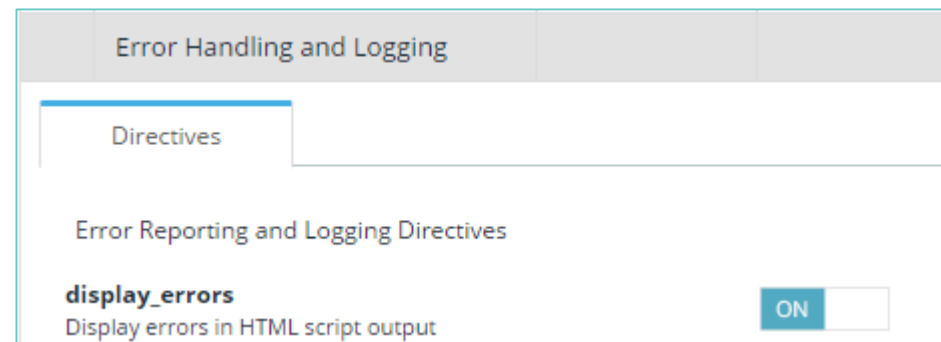
- HTML 5
 - ▶ The glue or DDS of the web
 - ▶ Easy to start
- Cascading Style Sheets
 - ▶ Bring organization to site
 - ▶ Powerful feature set (csszengarden.com)
- JavaScript
 - ▶ Power browser functions
 - ▶ Many good code libraries
 - jQuery
 - ExtJS-Sencha
 - Angular



Tip #4 – Develop with `display_errors` on

- PHP is best suited for iterative development model
 - ▶ Code → Save → Refresh → Repeat
- ALSO: Find errors as you go at the script level
 - ▶ `error_reporting(-1); // report all error levels`
 - ▶ `ini_set('display_errors', '1'); // *show me* the errors`
- Installation of Zend Server for Production turns this off
 - ▶ COMMON for this to be off in single LPAR environments
 - ▶ Get a developer LPAR (later)
- Admin GUI

- ▶ PHP → Extensions →



Tip #5 – Get a Development LPAR

- No more excuses!
 - ▶ IBM has made this MUCH easier (i on i)
 - ▶ IBM has made this much less expensive
 - Couple thousand, Disk, RAM
- Only one instance of Zend Server supported per LPAR
 - ▶ Settings are global (php.ini)
 - ▶ Library list can be manipulated, but isolation requires CHROOT, etc.
- New PHP developers don't get having dev & prod on same LPAR
 - ▶ “off the street” struggle with one server, let alone libl, etc...
 - ▶ Can do dangerous things like issue ALTER TABLE to CUSTMAST
- You are doing more with your IBM i now than ever before
 - ▶ “I never had a dev LPAR before” is not a valid excuse
 - ▶ Give your devs the freedom to experiment

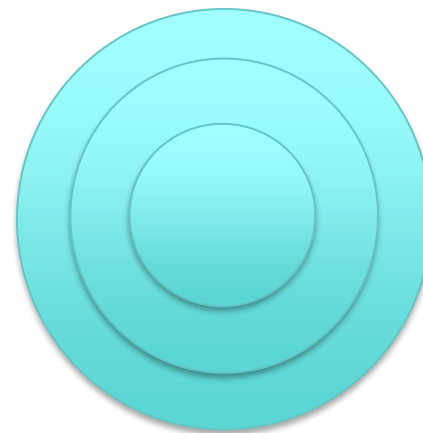
Tip #6 – Don't develop PHP like RPG

- RPG applications often are built with main file then chains
 - ▶ SETLL then READE ORDDetail
 - CHAIN PRODUCT CHAIN INVENT CHAIN ONPLAN
- Consider a single SQL statement
 - ▶ Join files are more efficient than chains, in most cases
 - ▶ SQL optimizers are genius!
 - ▶ Learn good indexing strategy
 - ▶ Go to SQL sessions at conferences
 - ▶ Ask Db2 guys, they'll tell you!
- Great Redbooks:
 - ▶ Database modernization
 - ▶ RPG and Application Modernization



Tip #7 – Bit Twiddling 101

- Consider this VERY carefully, what is the most expensive resource?
- Single quotes vs. double quotes
- Echo vs. Print
- Sprint() instead of ""
- Unset variables for memory management
- Close database connection if you are done unless pconnect
- \$row['id'] is faster than \$row[id];
- Static pages (html) are OK!!!
- Not everything has to be a Framework



Fastest to slowest:
Construct
Function
Static method
Object
Framework

Tip #8 – Understand modern development

■ Tools

▶ Git

▶ SSH

▶ HTTP / HTTPS

▶ Terminal Usage

■ Your role

▶ Front / Back / Full stack developer

▶ Dev Ops

▶ Test / Scrum master

The 2018 Web Developer Roadmap

An illustrated guide to becoming a Frontend or Backend Developer with links to courses







<https://codeburst.io/the-2018-web-developer-roadmap-826b1b806e8d>



Tip #9 - Performance of logs

- PHP logfile captures EVERY PHP error (can be adjusted)
- “clip” the logfile periodically
 - ▶ Rename or delete
 - ▶ Add this to weekly/monthly CL process
 - ▶ PHP just creates a new one when it needs it!
- Don't forget to read them, too!
 - ▶ Notice errors & deprecated functions



Directory				
/usr/local/zendphp7/var/log				
				
	Name	^	Size (KB)	Last Modified
	monitor.log			4 27 October, 2017 5:25:28 PM CDT
	pagecache.log			1 6 August, 2017 10:56:48 AM CDT
	php.log		2147483	17 September, 2017 3:22:18 PM CDT
	statistics.log		497	10 April, 2018 7:04:39 AM CDT
	tkit_debug.log			2 6 August, 2017 12:03:09 PM CDT

Tip #10 – Get vs. Post

- GET is great for embedded URL
 - ▶ Take you directly to a detailed page
 - ▶ Routing through application
 - ▶ Benign data
- POST
 - ▶ Better for more sensitive content
 - ▶ Always encrypt
 - ▶ Always filter and escape

Tip #11 – Use opcode cache

- PHP is interpreted
- Each script is read from disk, then parsed to opcode, then run
- Opcode cache can keep parsed copy in memory
- All or nothing purge

Administration

- Components
- Audit Trail
- Users
- Web API Keys
- License
- Import/Export
- Settings

Zero: OPcache Loaded Optimizer Plus Cache provides faster PHP execution through opcode caching and...

OPcache, aka Optimizer Plus Cache, improves PHP performance by storing precompiled script bytecode in shared memory, thereby removing the need for PHP to load and parse scripts on each request. This extension is bundled with PHP 5.5.0 and later, and is available in PEAR for PHP versions 5.2, 5.3 and 5.4.

opcache.blacklist_filename The location of the OPcache blacklist file.	<input type="text" value=""/>	opcache.consistency_checks Check the cache checksum every N requests.	<input type="text" value="0"/>
opcache.dups_fix This back should only be enabled to work around "Cannot redeclare class" errors.	<input type="checkbox" value="OFF"/>	opcache.enable_file_override Allow file existence caching.	<input type="checkbox" value="ON"/>
opcache.error_log The location of the OPcache error log file.	<input type="text" value="/usr/local/zendsvr6/var/log"/>	opcache.fast_shutdown If enabled, a fast shutdown sequence is used for the shutdown logic.	<input type="checkbox" value="OFF"/>
opcache.file_update_protection Prevents caching files that are too flat (their number of seconds old).	<input type="text" value="2"/>	opcache.force_restart_timeout Time duration (in seconds) for waiting for a scheduled restart to begin if the cache is not being accessed.	<input type="text" value="180"/> seconds
opcache.interned_strings_buffer The amount of memory used to store interned strings, in megabytes. This configuration directive is ignored in PHP > 5.5.0.	<input type="text" value="4"/> MBytes	opcache.log_verbosity_level The verbosity of the OPcache log.	<input type="text" value="1"/>
opcache.max_accelerated_files The maximum number of keys persisted in the OPcache hash table.	<input type="text" value="2000"/>	opcache.max_file_size The maximum file size that will be cached, in bytes. If this is 0, all files will be cached.	<input type="text" value="0"/>
opcache.max_wasted_percentage The maximum percentage of "wasted" memory until a restart is scheduled.	<input type="text" value="5"/> %	opcache.memory_consumption The OPcache shared memory usage size. The amount of memory for storing precompiled PHP code in Mbytes.	<input type="text" value="64"/> MBytes
opcache.optimization_level A bitmask where each bit enables or disables the appropriate OPcache passes.	<input type="text" value="0x7FFFFFFF"/>	opcache.preferred_memory_model Shared memory model to use.	<input type="text" value=""/>
opcache.protect_memory Protects shared memory from unapproved writes while executing scripts. This is useful for internal debugging only.	<input type="checkbox" value="OFF"/>	opcache.restrict_api Use OPcache API only from path starting the specified string.	<input type="text" value=""/>
opcache.revalidate_freq How often to check file timestamps for changes to the shared memory storage allocation.	<input type="text" value="2"/> seconds	opcache.revalidate_path Enables or disables file search in include_path.	<input type="checkbox" value="OFF"/>
		opcache.validate_timestamps If enabled, OPcache checks the file timestamps and updates the cache accordingly.	<input type="checkbox" value="ON"/>

Tip #12 – Data Cache

- Green screen lookup window vs. drop down
- Data Cache (Zend or APC) saves data in memory
- Performance
- White paper
- Better performance for frequently accessed lists
- Less stress on DB2
- Aggregate values (BI)
- Look at examples...



Standard DB2 Call

```

16 function getOrders()
17 {
18     $DB2handle = Connect();
19
20     $sql = 'select * from classicmodels.orders';
21     $stmt = db2_exec($DB2handle, $sql);
22
23     if (! $stmt) {
24         echo 'Connection failed: ' . db2_stmt_error() . ' : ' . db2_stmt_errormsg();
25         return FALSE;
26     }
27     var_dump($stmt);
28     while (($row = db2_fetch_assoc($stmt)) !== FALSE) {
29         $orders[] = $row;
30     }
31
32     return $orders;
33 }
    
```

Function	Count	Duration Inclusive (ms)
{main}	2 rows are not displayed	
Conn {main}	1	572.896
db2_ getOrders()	1	76.508
get0 db2_exec()	1	27.182
db2_ Connect()	1	17.922
db2_ db2_connect()	1	17.855
db2_ db2_fetch_assoc()	327	16.845

DB2 Call w/Data Cache

```

35= function getOrders2()
36 {
37     $orders = zend_shm_cache_fetch('my-orders');
38
39     if ($orders === FALSE) {
40
41         $DB2handle = Connect();
42
43         $sql = 'select * from classicmodels.orders';
44         $stmt = db2_exec($DB2handle, $sql);
45
46         if (!$stmt) {
47             echo 'Connection failed: ' . db2_stmt_error() . ' : ' . db2_stmt_errormsg();
48             return FALSE;
49         }
50         while (($row = db2_fetch_assoc($stmt)) !== FALSE) {
51             $orders[] = $row;
52         }
53
54         // Store results...326 orders...
55         zend_shm_cache_store('my-orders', $orders, 24 * 3600);
56     }

```

Function	Count	Duration Inclusive (ms)
3 rows are not displayed		
{main}	1	533.768
getOrders2()	1	75.485
db2_exec()	1	27.003

Function	Count	Duration Inclusive (ms)
2 rows are not displayed		
{main}	1	427.947
getOrders2()	1	2.088

Tip #13 – That one connection

- PHP can support multiple connections to DB2
 - ▣ Also MySQL, MSSQL, Oracle, etc.
- Only one connection required per DB, more costs performance
 - ▣ Multiple SQL statements and result sets can share a connection
 - ▣ Connection consumes memory, don't waste
 - ▣ Connection startup & shutdown
- Sometimes you need another connection
 - ▣ If you need different *LIBL
 - ▣ Connecting to second partition



Tip #14 – Prepared SQL

- Test in ACS Database View...
- DB2_Exec
- DB2 Execute

```
23 /* Construct the SQL statement */
24 $sql = "SELECT * FROM ZENDSVR.SP_CUST WHERE CUST_ID > ? FOR FETCH ONLY";
25
26 /* Prepare, bind and execute the DB2 SQL statement */
27 $stmt= db2_prepare($conn_resource, $sql);
28 $lower_limit = 1220; //from the CUST_ID value
.
45 //Execute statement , uses a binding of parameters
46 db2_bind_param($stmt, 1, "lower_limit", DB2_PARAM_IN);
47 $result = db2_execute($stmt);
48
49     if (!$result) {
50         echo 'The db2 execute failed. ';
51         echo 'SQLSTATE value: ' . db2_stmt_error();
52         echo ' Message: ' . db2_stmt_errormsg();
53     }
54     else
55     {
56         while ($row = db2_fetch_array($stmt))
```

Tip #15 – Use Stored Procedures

- The fastest method for accessing data and business logic
- Faster than the toolkit
- Ubiquitous
- If you got ‘em, use ‘em.

```
CREATE PROCEDURE sales_price(
    IN Customer# CHAR(8),
    IN Product# CHAR(8),
    OUT price DECIMAL(12,2))
LANGUAGE SQL
BEGIN
    SELECT CustPrice          FROM priceTable
    WHERE CUSTNO=Customer# AND
    PRODNO=Product# AND
    active = 'Y'
END;
```

```
tn5250 - zend.idevcloud.com
File Edit View Help
Columns . . . : 1 71 Browse MPAVLAK/QSQLSRC
SEU=> MPEX1
FMT ** ...+... 1 ...+... 2 ...+... 3 ...+... 4 ...+... 5 ...+... 6 ...+... 7
***** Beginning of data *****
0001.00 DROP PROCEDURE MPAVLAK/MPEX1;
0002.00 CREATE PROCEDURE MPAVLAK/MPEX1 ( ) RESULT SETS 1 LANGUAGE SQL
0003.00 SQLPROC: BEGIN
0004.00 DECLARE CUR1 SCROLL CURSOR FOR
0005.00 SELECT * FROM ZENDSVR/SP_CUST WHERE CUST_ID > 1220;
0006.00 OPEN CUR1;
0007.00 SET RESULT SETS CURSOR CUR1;
0008.00 END
***** End of data *****

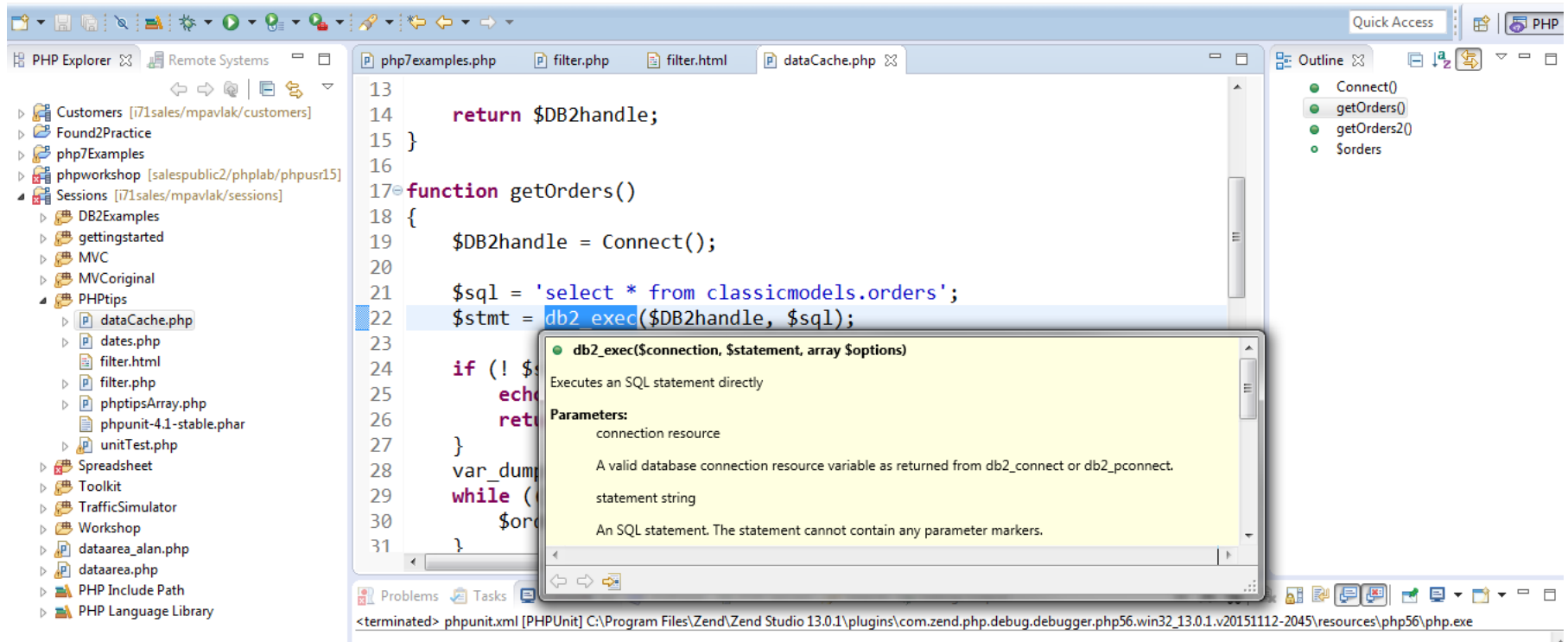
F3=Exit F5=Refresh F9=Retrieve F10=Cursor F11=Toggle F12=Cancel
F16=Repeat find F24=More keys
(C) COPYRIGHT IBM CORP. 1981, 2007.
5250 MW 009/002
```

Tip #16 – Use a good editor

- Zend Studio is an IDE
 - ▣ Syntax check, Variable prompting (case sensitive)
 - ▣ HTML, JavaScript, CSS support
 - ▣ Free for first year
- Notepad is painful
- Other editors
 - ▣ Notepad++
 - ▣ Netbeans
 - ▣ PHP Storm



Good editor



Tip #17 – System Naming

- Gives scripts a library list like logon screen
- 3 ways to do library list (see other presentation DB2 Examples)

```

3 function Connect()
4 {
5     // Standard DB connection to DB2...
6     $conn = "*LOCAL";
7     $name = "";
8     $pwd = "";
9     $DB2handle = d 9 $options = array("i5_lib"=>"ZENDSVR6");
10    if (! $DB2hand 10
11        echo 'Conn 11 $con1 = db2_connect($db, $user, $pwd, $options);
12                                db2_conn_errormsg());
13
14    return $DB2handle 9 $options = array( 'i5_naming' => DB2_I5_NAMING_ON,
15 } 10 'i5_lib1'=>"MPAVLAK ZENDSVR6 AMAZON");
11
12 $con1 = db2_connect($db, $user, $pwd, $options);
13 if (!$con1) {
14     echo 'Error connecting to ' . $db . ' - ' .
15         db2_conn_error() . ' - ' .
16         db2_conn_errormsg();
17     exit();
18 }

```

Tip #18 – Global Variables

- Don't use them
 - ▶ Globals are the antithesis of functional development
 - ▶ They are the GOTO of the variable world
 - ▶ Remember RPG III? Yep, you get it...

- Register Globals

- ▶ Security hole
- ▶ Deprecated in 5.4
- ▶ Removed in 5.6
- ▶ Replace with `extract()`

- Avoid the Global Namespace

- ▶ Prevent collisions

```
3 function formatDate() {  
4  
5     global $date;  
6  
7     $formattedDate = date("Y/m/d");  
8     return $formattedDate;  
9  
10 }  
11  
12 $date = 'Herman';  
13  
14 $displayDate = formatdate();
```

Tip #19 – Static Typing

- PHP 7 introduces many new features and scalar type hinting is one...

▶ <http://php.net/manual/en/migration70.php>

```
function add(float $a, float $b) {  
    return $a + $b;  
}  
  
$returnValue = add(1.5, 2.5); // int(4)  
// Works  
  
$returnValue = add("1 foo", "2");  
// PHP 5.6 and below gives a Notice  
// PHP7 TypeError  
  
$returnValue = add(1, 2); // int(3)  
// Widening
```

Static Typing (cont.)

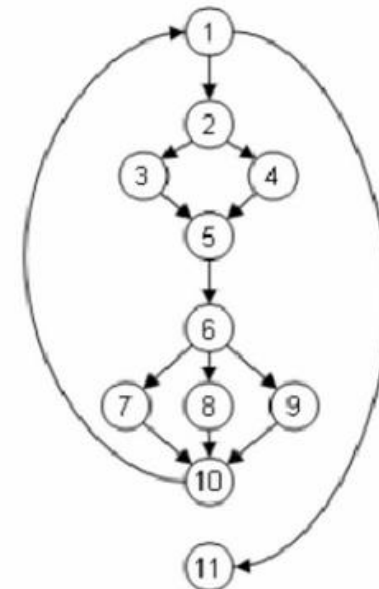
- So is return typing... (Thanks Cal!)

```
function foo(): array {  
    return [];  
}
```

```
Class MyOtherClass extends MyClass {  
  
    function make(): MyOtherClass  
    {  
        return new MyOtherClass();  
    }  
}
```

Tip #20 - Manage Cyclomatic Complexity

- How deeply nested is the logic?
- Impacts all languages
- Gets worse over time (Maintenance)
- Use phpcloc



Laravel	1.62
Symfony	1.88
Slim	2.40
Zend	2.76
CakePHP	3.30

[*Taylor Ottwell](#)

< 10 Easy to maintain
11-20 Harder to maintain
21+ Candidates for refactoring/redesign

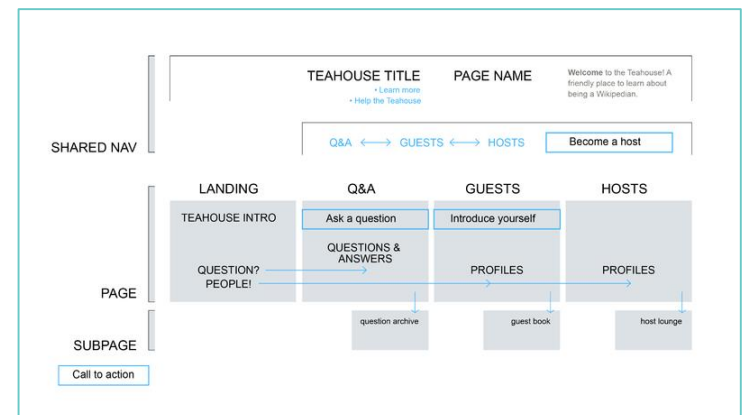
Tip #21 – Agile & Wireframe

■ Agile

- ▶ Deliver a little bit, more often
- ▶ Don't need full blown scrum for agile techniques

■ Wireframe

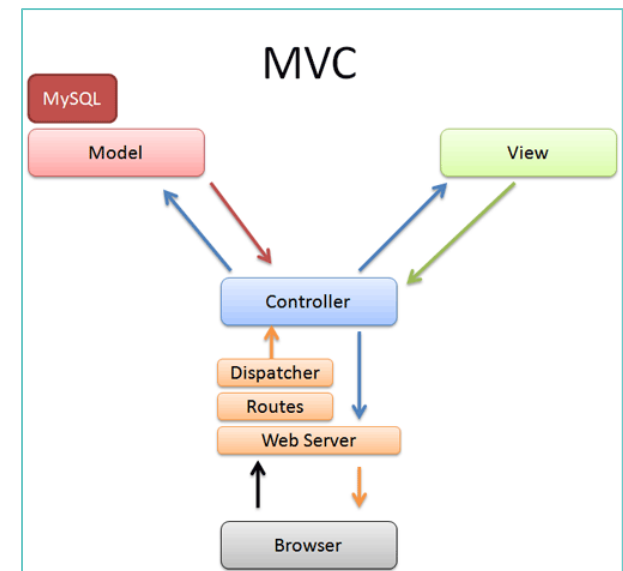
- ▶ Design, mock up app before coding
- ▶ Modern design document



Tip #22 – Study Design Patterns

- Foundation of most frameworks
- Developed initially for traffic management
- Supports many methodologies

- ZF1 – Singleton
- ZF2 – Factory
- Symfony

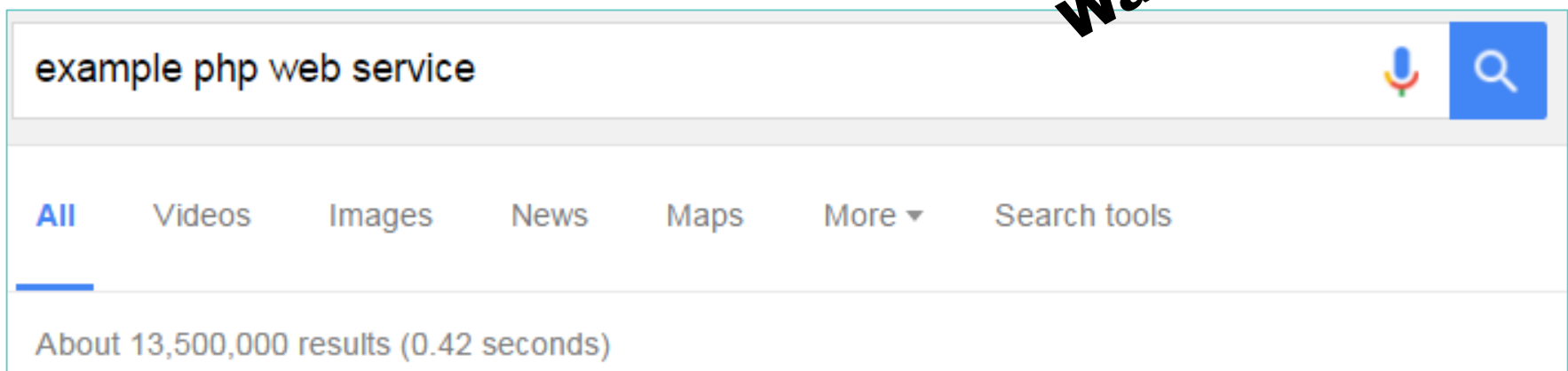


- No, MVC is a design architecture...not a design pattern!

Tip #23 – Google and PHP.net

- Your two best friends in PHP development
- When learning a new skill, oftentimes don't know question to ask
- Google heuristics can estimate what you are looking for
 - ▣ Often can lead you back to php.net
 - ▣ May lead to examples

Watch the Dates!



Tip #24 – Just do it!

- Don't wait around for your boss to tell you
- Find an excuse to play
- Build a website (Go Daddy is \$10 a month)
- Volunteer
 - ▣ Church
 - ▣ VFW
 - ▣ Local rock band
- Internal department project
 - ▣ IT help desk
 - ▣ Intranet
 - ▣ Media wiki for documentation

Tip #25 - OPC (or as Jon likes to say OPO)

- Other Peoples Code
 - ▶ It's how you learned RPG/COBOL
 - ▶ Community loves to share, but be careful, not all good code
- Google can help find resources
 - ▶ Hotscripts.com, Phpclasses.org, Sourceforge
- Sort by date, descending...for most current stuff!

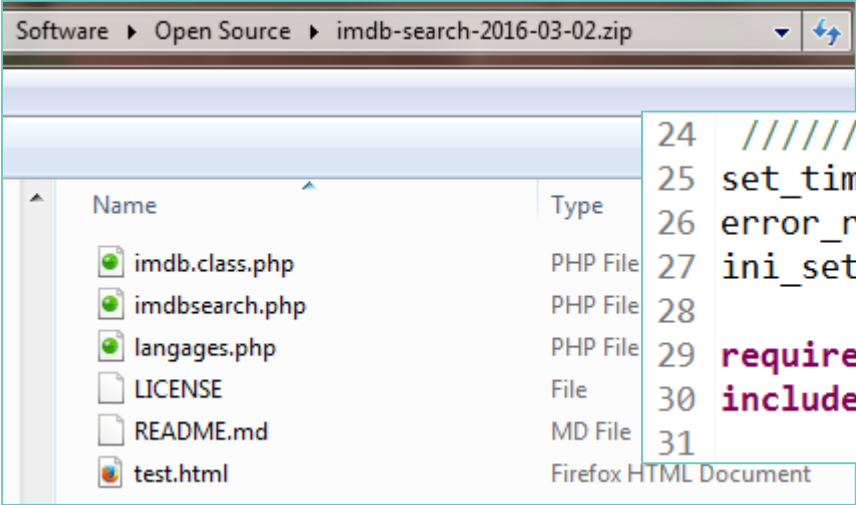
sourceforge



The screenshot shows the PHPClasses website interface. At the top, there's a navigation bar with 'Login' and 'Register' links. Below that, the site title 'PHP CLASSES' is visible. A search bar is present with the text 'PHP IMDB search: Search movies and retrieve information from IMDB'. The main content area displays details for a class named 'indb-search 1.0'. The details are organized into several sections: 'Info', 'Example', 'View files (6)', 'Download zip', 'Reputation', 'Support forum', 'Blog', and 'Links'. The 'Reputation' section shows 'Total: 47' and 'This week: 47'. The 'Download Rankings' section shows 'All time: 8,318' and 'This week: 13'. The 'Version' section shows 'indb-search 1.0'. The 'License' section shows 'The PHP License'. The 'PHP version' section shows '5.0'. The 'Categories' section shows 'PHP 5, Searching, Web services, Video'.



OPC (cont.)



```

24 ///////////////////////////////////////////////////////////////////
25 set_time_limit(0);
26 error_reporting(E_ERROR | E_WARNING | E_PARSE);
27 ini_set("memory_limit" , -1);
28
29 require('langages.php');
30 include("imdb.class.php");
31

```

```

15 class Imdb
16 {
17     // Scrape movie information from IMDb page and return results in an array.
18     // this function becomes public by ArouG and has been modified !
19     // I prefer use the IMDb search engine and not ASK, GOOGLE and the others ! Much better ^^
20     public function scrapeMovieInfo($imdbUrl, $getExtraInfo = true, $myip, $Country = '')
21     {
22         require('langages.php');
23         $arr = array();
24         $html = $this->geturl("${imdbUrl}combined", $myip);
25         $title_id = $this->match('/<link rel="canonical" href="http:\\\\www.imdb.com\\title\\(tt\d+)\combined" \\/>ms',
26             if(empty($title_id) || !preg_match("/tt\d+/i", $title_id)) {
27                 $arr['error'] = "No Title found on IMDb!";
28                 return $arr;
29             }
30         $arr['imdb_id'] = $title_id;
31         $tmp= str_replace("'", '', trim($this->match('/<title>(IMDb \- )*(.*) \\.?</title>ms', $html, 2)));
32         $arr['title'] = html_entity_decode($tmp, ENT_COMPAT, 'UTF-8');
33         $arr['year'] = trim($this->match('/<title>.*?\\(.*?\\d{4}\\).*?</title>ms', $html, 1));
34

```

Tip #26 – Consider test driven development

- Continuous Integration / Delivery is the goal
- Requires company to automate EVERY step in the software life cycle
- Automate Testing, start slow
 - ▶ Create test scripts for functions

- Phpunit

```
23 {
24     $this->angle = null;
25     parent::tearDown();
26 }
27
28 public function testAttributesAreOfCorrectType()
29 {
30     $this->assertAttributeInstanceOf('clarkphp\TrafficSimulator\Magnitude', 'magr
31     $this->assertAttributeInstanceOf('clarkphp\TrafficSimulator\Unit\AngularUnit'
32 }
33
34 /**
35  *
36  * @todo Find clean way of testing all implemented angular unit types.
37  */
38 public function testToStringYieldsProperValue()
39 {
40     $this->assertSame('2.5 deg', '' . $this->angle);
41 ..
```

The screenshot shows a code editor with PHP code for a test class. The code includes a constructor, a tearDown method, and three test methods: testAttributesAreOfCorrectType, testToStringYieldsProperValue, and testGetUnitYieldsAngularUnitObject. A comment in testAttributesAreOfCorrectType indicates a TODO for finding a clean way to test all implemented angular unit types. The testToStringYieldsProperValue method asserts that the string representation of the angle is '2.5 deg'.

On the right side of the IDE, there is a file explorer showing the project structure with the following files and folders:

- import declarations
 - clarkphp\TrafficSimulator\Angle
 - clarkphp\TrafficSimulator\Magnitude
 - clarkphp\TrafficSimulator\Unit\Degrees
 - clarkphp\TrafficSimulator\Unit\AngularUnit
- AngleTest
 - Angle : Distance
 - setUp()
 - tearDown()
 - testAttributesAreOfCorrectType()
 - testToStringYieldsProperValue()
 - testGetMagnitudeYieldsMagnitudeObject()
 - testGetUnitYieldsAngularUnitObject()

At the bottom of the IDE, there is a PHPUnit console window showing the execution results. The status bar indicates 'Runs: 56/56', 'Errors: 0', and 'Failures: 0'. The console window shows the following output:

```
Trace Failure
Object Diff
Code Coverage
```

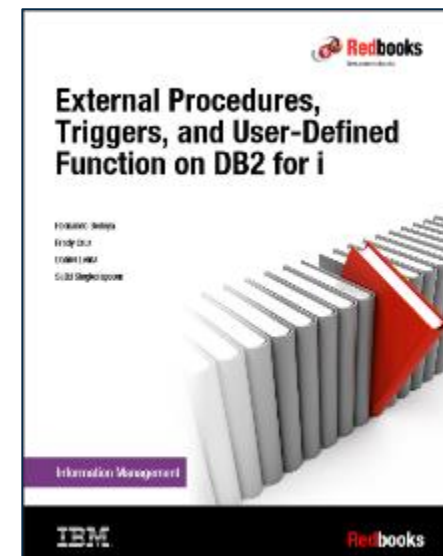
Tip #27 – Source Control

- Existing commercial products handle PHP
 - ▶ Rocket, Arcad, Remain, etc.
- Consider a code respository
 - ▶ SVN, GiT (on IBM i)
- You are doing more with this source code than ever before
 - ▶ Maybe it's time for a change
 - ▶ Consider all the elements that make up a website
 - CSS, HTML, JavaScript
- New resources require new rules...



Tip #28 – Master SQL

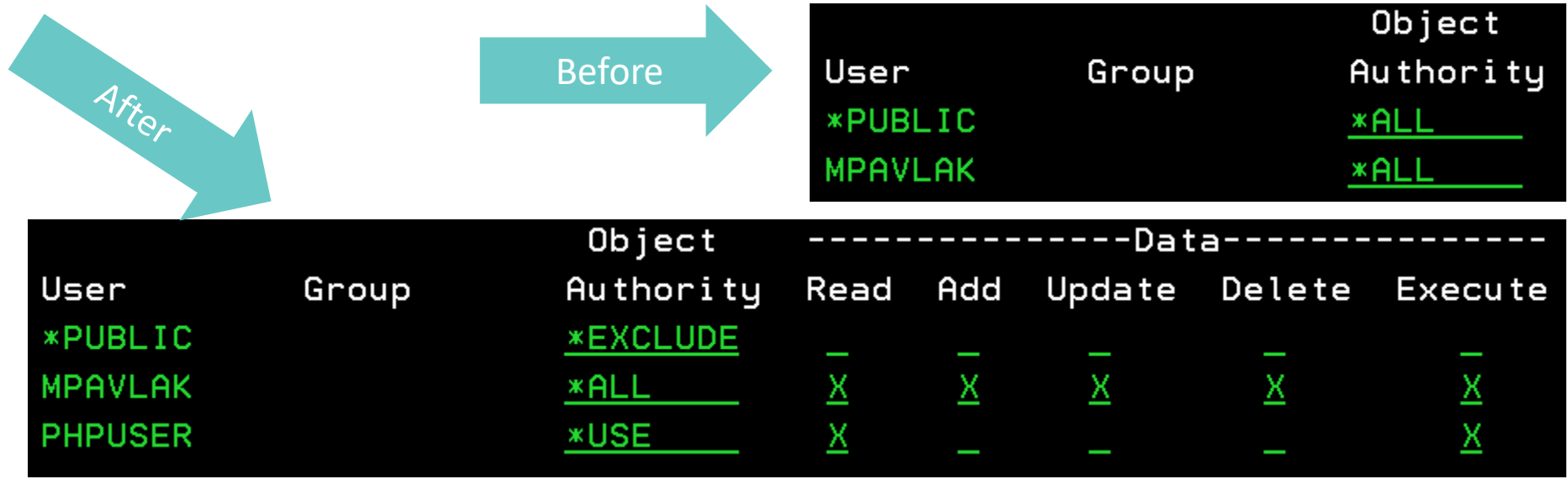
- IBM DB2 Extension uses same DB2 CLI as the native environment
- Any changes to DB2 are immediately available to PHP
- Every Technology Refresh contains changes to DB2 and often SQL
- SQL is the primary language for data access in PHP
 - ▢ Query optimizer
 - ▢ iNav interface
 - ▢ Index advisor



- New <http://publib-b.boulder.ibm.com/redpieces/abstracts/sg246503.html>

Tip #29 – Master Security

- You are embarking on a new world (www)
- Consider access to your data
- Consider access to your IBM i
 - ▢ Have you looked at your datasets?
- Take a networking class (the network/internet is the computer)



Tip #30 – Date Handling

- Formatting dates in PHP is all about time stamps
- DB2 installations usually store date as integers or date fields
- To leverage PHP, must master conversions

```
3 $today = time();
4
5 echo $today;
6
7 $dateIn = '02/05/2016';
8
9 $dateWrk = strtotime($dateIn);
10
11 echo '<br /><br />' . $dateWrk;
12
13 $formattedDate = date('l F d, Y', $dateWrk);
14
15 echo '<br /><br />' . $formattedDate;
```

1457571317

1454659200

Friday February 05, 2016

Tip #31 – Error suppression

- Embrace your mistakes
- Use of the @ to suppress errors is still supported in PHP 7
 - ▶ THIS IS BAD!!!! Equivalent of MONMSG CPF0000
- Learn to handle errors
 - ▶ Try & Catch (example in samples!)
 - ▶ Exception processing

```
try {  
    $ToolkitServiceObj = ToolkitService::getInstance($db, $user, $pass, $extension);  
}  
  
catch (Exception $e)  
{  
    echo $e->getMessage(), "\n";  
    exit();  
}
```

Practical guide to PHP error from RPG call

Monitoring for Errors in PHP Script Called from CLLE

APRIL 5, 2018 BY BRUCE GUETZKOW · 0 COMMENTS



<http://club.alanseiden.com/blog/2018/04/05/monitoring-for-errors-in-php-script-called-from-clle/>

Tip #32 – Never trust external data

- Developer is agent of security in web app design
- Filter inputs (Frameworks do this but so does PHP!)
- JavaScript may do this, but that is not enough...

- [Validate filters](#)
- [Sanitize filters](#)
- [Other filters](#)
- [Filter flags](#)

```
3 $redisplayForm = FALSE;
4 $errorCount = 0;
5
6 $quantityOK = filter_input(INPUT_POST, 'quantity', FILTER_VALIDATE_INT);
7
8 if ($quantityOK == TRUE) {
9     $quantityClean = $_POST['quantityIn'];
10    echo "Good Quantity!!!";
11 } else {
12    $errorCount++;
13    $errorMessage[] = 'Bad Quantity Entered, please try again...';
14    $redisplayForm = TRUE;
15    echo "Bad Quantity!!!";
16 }
```

Tip #33 – Who are you?

- Netcraft knows all
- Hide phpinfo
- Never show a path, Clean URL's
- Display errors in development ONLY
- Expose_php in PHP.ini file to reduce visibility
- Gone in PHP7 (Yay!)

common.org/?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000

PHP Credits

PHP Group

Thies C. Arntzen, Stig Bakken, Shane Caraveo, Andi Gutmans, Rasmus Lerdorf, Sam Ruby, Sascha Schumann, Zeev Suraski, Jim Winstead, Andrei Zmievski

Language Design & Concept

Andi Gutmans, Rasmus Lerdorf, Zeev Suraski, Marcus Boerger

PHP Authors

THANK YOU

Mike Pavlak

Mike.Pavlak@freschesolutions.com

FRESCHESOLUTIONS

