Dokumentation zur Umstellung von kvwmap auf PHP 7.3 und PHPMapScript SWIG und MapServer 7.4

Erstellt von: Dr. Peter Korduan, GDI-Service
letzte Änderung am: 11.03.2020

Änderungen:

Datum	Änderung
04.01.2020	Migrationshinweise
11.03.2020	Fragen und Antworten

Inhaltsverzeichnis

1 Migrationshinweise

Der kvwmap-server Container beinhaltet:
php 7.3.11-1~deb10u1
mapserver 7.2.2

1.1 MapServer

Mapserver Änderungen von 6.4 auf 7.4:
* OPACITY 70 ersetzen durch COMPOSITE OPACITY 70 END
* FILTER muss MapServer Expression beinhalten oder ersetzen durch SQL-Subselects in Data
* MapFiles müssen in UTF-8 gespeichert werden. Evt. Layerlevel ENCODING verwenden
* GD Graphics Library nicht mehr unterstützt und ist nur noch optional verfügbar.
* Bitmap-Fonts wurden durch TrueType Font ersetzt deshalb haben labelobj keine Eigenschaft type mehr:
Vorkommen von $map->scalebar->label->type = 'truetype'; löschen
* LABEL PARTIALS default ist nun false
* Empfehlung für die Verwendung von SWIG API welche PHP 7+ unterstützt:
  * min SWIG 3.0.11, empfohlen wird aber 4.0.0
  * Funktionen, die in der alten PHP-MapScript API nicht mehr gehen in der SWIG-API suchen und durch diese ersetzen.
  * all of your PHP scripts (that leverage MapServer objects and functions) must now always first include the generated `mapscript.php` file containing MapServer constants

```
include("C:/ms4w/apps/phpmapscriptng-swig/include/mapscript.php");
```

- take note of the change in how to declare your new objects:

```
// open map
$oMap = new mapObj("C:/ms4w/apps/phpmapscriptng-swig/sample.map");
```

instead of the former way:

```
// open map
$oMap = ms_newMapObj("C:/ms4w/apps/phpmapscript/sample.map");
```

1.2 PHP

SWIG ist eine Script-Sprachen-Schnittstelle zu C oder C++ Programmen.
Umstellung von PHP 5.6 auf PHP 7.4

5.6 => 7.0

* Error handling for eval() should now include a *catch*block that can handle this error.

* Reihenfolge der Interpretation von indirekten Ausdrücken hat sich geändert

Expression	PHP 5 interpretation	PHP 7 interpretation
$$foo['bar']['baz']	${$foo['bar']['baz']}	($$foo)['bar']['baz']
$foo->$bar['baz']	$foo->{$bar['baz']}	($foo->$bar)['baz']
$foo->$bar['baz']()	$foo->{$bar['baz']}()	($foo->$bar)['baz']()

Foo::$bar['baz']() Foo::{$bar['baz']}() (Foo::$bar)['baz']()

* Alle Ausdrücke, die indirekte Aufrufe verwenden umschreiben.
* Zuordnungen mit list() werden nun in der Reihenfolge des Erscheinens ausgeführt. Wenn eine Variable mit gleichem Namen zwei mal in list auftaucht wird der erste Wert verwendet. Der zweite Wert wird ignoriert. Prüfen ob list verwendet wird und ob doppelte Namen vorkommen können. Ggf. Weitere Änderungen von list berücksichtigen.
* Nur Variablen sollten by Reference an Funktionen übergeben werden. Keine Funktionen oder anderes in ().
* foreach Verhalten hat sich geändert
  * array-pointer wird nicht mehr modifiziert beim Durchlaufen
  * Eine Kopy wird bei by-value verwendet. Man kann also keine Werte des Array's mehr mit foreach ändern.
  * foreach by-reference hat sich geändert. Prüfen ob so etwas verwendet wird.
* Verwenden wir octal (0128) Literals?
* Verwenden wir bitshifts (1 >> -1)?
* Division by 0 erzeugt jetzt statt einer Warnung ein Error
* hexadezimale Strings werden nicht mehr als Zahlen interpretiert
* nicht mehr unterstützte Funktionen
  * call_user_method, call_user_method_array
  * nutze statt dessen call_user_func, call_user_func_array
* Alle ereg Funktionen wurden entfernt, nutze statt dessen PCRE also preg Funktionen
* mcrypt_generic_end ersetzen durch mcrypt_generic_deinit
*Alle ext/mysql Funktionen wurden entfernt. Ersetzen durch Alternative MySQL API
  * Umstellung von mysql auf mysqli Funktionen
    Siehe User Contributed Notes auf https://www.php.net/manual/de/ref.mysqli.php
* set_magic_quotes_runtime() wird nicht mehr unterstützt
* dl() wird nicht mehr unterstützt
* imageps... Funktionen werden nicht mehr unterstützt
* Diese hier nicht mehr als Variablen namen verwenden
  - resource
  - object
  - mixed
  - numeric
* calls from incompatible context removed, nicht statische methoden sollen nicht in statischen Methoden aufgerufen werden.
* verwenden wir den yield Operator?
* verwenden wir func_get_arg?
* verwenden wir $HTTP_RAW_POST_DATA?
* keine # mehr in ini-Files mehr als Comment erlaubt. Statt dessen ; verwenden
* verwenden wir split() Das wird nicht mehr unterstützt?
* substr("test",4); # false in PHP 5, "" in PHP 7

PHP 7.0 => 7.1

* Zu wenig Argumente beim Aufruf einer Funktion wirft jetzt einen Fehler statt einer Warnung.
* error_log setting in php.ini prüfen.
* Applying the empty index operator to a string (e.g. *$str[] = $x*) throws a fatal error instead of converting silently to array. Variablen mit [] müssen also auch immer vom Typ Array initiert sein.
* DateTime() liefert nun auch microseconds

* Variables bound to a closure via the *use* construct cannot use the same name as any superglobals, $this, or any parameter. For example, all of these function definition will result in a fatal error:

<?php

$f = function () use ($_SERVER) {};

$f = function () use ($this) {};

$f = function ($param) use ($param) {};

* JSON Encoding von Leeren Key Namen ("" => 1) führt jetzt zu einem leeren Property Namen
* Mit JSON_UNESCAPED_UNICODE flag in json_encode U+2028 U+2029 werden escaped
* Whilst *$this* is considered a special variable in PHP, it lacked proper checks to ensure it wasn't used as a variable name or reassigned. This has now been rectified to ensure that *$this* cannot be a user-defined variable, reassigned to a different value, or be globalised.

PHP 7.1 => 7.2

* $additional_headers Parameter in mail und mb_sendmail erfordern array als Argument statt string
* casten wir array zu objekten und umgekehrt? Besseres Handling
* Warnung bei nicht zählbaren Typen
* minimal unterstützte Windows Version Windows 7 / Windows Server 2008 R2
* The json_decode() function option, **`JSON_OBJECT_AS_ARRAY`**, is now used if the second parameter (assoc) is **`NULL`**. Previously, **`JSON_OBJECT_AS_ARRAY`** was always ignored.

PHP 7.2 => 7.3

* heredoc Strings z.B. <<<FOD dürfen reserviertes Wort nicht beinhalten bis FOD; am Ende erscheint. Suchen ob das z.B. Irgendwo z.B. für <<<SQL verwendet wird.
Siehe https://wiki.selfhtml.org/wiki/PHP/Tutorials/Einstieg/Grundlagen#Einfache_Typen_und_ihre_Literale zu m Thema heredoc und nowdoc. Evtl. bei der Gelegenheit mal die SQL-Statements von echo "" . $xy auf heredoc Strings umstellen.
* libcurl ≥ 7.15.5 is now required

PHP 7.3 => 7.4

* fn kann nicht mehr als Classen oder Funktionsname verwendet werden.
* <?php am Ende ohne newline wird anders interpretiert.
* htmlentities() will now raise a notice (instead of a strict standards warning) if it is used with an encoding for which only basic entity substitution is supported, in which case it is equivalent to htmlspecialchars().
* fread() and fwrite() will now return **`FALSE`** if the operation failed. Previously an empty string or 0 was returned.

These functions now also raise a notice on failure, such as when trying to write to a read-only file resource.
* fputcsv und fgetcsv akzeptiert empty str als escape Char
* The bundled libzip library has been removed. A system libzip >= 0.11 is now necessary to build thezip extension

2 Entwicklungsumgebung einrichten

Folgende Einstellungen wurden vorgenommen um die Umstellung durchzuführen und zu testen.
Auf gdi-service.de Server wurde ein neuer Container (web2.0.0) gestartet. Dazu wurde dcm erweitert, so dass man die Version mit angeben kann.

```
dcm [ run | rerun | console ] web 2.0.0
```

Des weiteren wurde die env_and_volumes copiert nach env_and_volumes_2.0.0
Darin wurden Pfade und Namen angepasst.
In etc/apache2/sites-available gibt es Konfig-Dateien mit _2.0.0 am Ende und in sites-enabled werden diese geladen. PHP ist im Container 2.0.0 unter /etc/php/7.3/… Deshalb auch neue Konstante in env_and_volumes. php.ini aus diesem Verzeichnis nach /home/gisadmin/etc/php5/apache2/php_2.0.0.ini und die in env_and_volume_2.0.0 eingebunden. Werte in php_2.0.0.ini angepasst.
Es wurde ein zweites www Verzeichnis angelegt als www2 und dort das wichtigste was man braucht reingepackt.
Die Umgebung ist erreichbar unter:
http://gdi-service.de:8085/kvwmap_pet_dev
Wenn man die Anwendung im web2 Container über Port 80 erreichen möchte bietet sich eine Umleitung im Web Container an, die über einen Eintrag in einer Apache.conf Datei erfolgen kann:

```
ProxyPass /kvwmap2 http://web2-0-0/kvwmap2
ProxyPassReverse /kvwmap2 http://web2-0-0/kvwmap2
```

Für weitere Zugriffe müssten entsprechend weitere Umleitungen angelegt werden.

3 Warnungen

Es werden folgende Warnungen behoben:

- Array Key existiert nicht
- Konstante existiert nicht bei define bei der die Konstante nicht in Anführungsstrichen steht, z.B. in credentials.php

4 Umstellung von mysql auf mysqli

Fatal error: Uncaught Error: Call to undefined function mysql_connect() in /var/www/apps/kvwmap_pet_dev/class/mysql.php:600 Stack trace: #0 /var/www/apps/kvwmap_pet_dev/start.php(33): database->open() #1 /var/www/apps/kvwmap_pet_dev/index.php(143): include('/var/www/apps/k...') #2 {main} thrown in /var/www/apps/kvwmap_pet_dev/class/mysql.php on line 600

Die Variante wie mit mysqli Daten aus mysql abgefragt werden sieht so aus: siehe
https://www.php.net/manual/de/mysqli.quickstart.statements.php

```
<?php
$mysqli = new mysqli("example.com", "user", "password", "database");
if ($mysqli->connect_errno) {
    echo "Failed to connect to MySQL: (" . $mysqli->connect_errno . ") " . $mysqli->connect_error;
}

if (!$mysqli->query("DROP TABLE IF EXISTS test") ||
    !$mysqli->query("CREATE TABLE test(id INT)") ||
    !$mysqli->query("INSERT INTO test(id) VALUES (1), (2), (3)")) {
    echo "Table creation failed: (" . $mysqli->errno . ") " . $mysqli->error;
}

$res = $mysqli->query("SELECT id FROM test ORDER BY id ASC");

echo "Reverse order...\n";
for ($row_no = $res->num_rows - 1; $row_no >= 0; $row_no--) {
    $res->data_seek($row_no);
    $row = $res->fetch_assoc();
    echo " id = " . $row['id'] . "\n";
}

echo "Result set order...\n";
$res->data_seek(0);
```

```
while ($row = $res->fetch_assoc()) {
    echo " id = " . $row['id'] . "\n";
}
?>
```

Für das Suchen nach Vorkommen diesen Befehl verwenden:

```
grep -R --exclude-dir=fast_cases 'suchtext' *
```

4.1 Objektorientierter vs. Prozeduraler Stil

Für einige Funktionen gibt es zwei verschiedene Schreibweisen. Z.B. gibt es für den ehemaligen Befehl mysql_fetch_array mit mysqli folgende zwei Varianten:
$mysqli→mysqli_result→fetch_assoc(); fetch_assoc ist eine Methode des Objektes mysqli_result
$mysqli→mysqli_fetch_assoc($mysqli_result); mysqli_fetch_assoc ist eine Methode des Objektes mysqli

4.2 connect

mysql_connect(host, user, passwd) und mysql_select_db(dbName) => new mysqli(host, user, passwd, dbName) und Fehlerauswertung statt connection_id mysqli->connect_errno auswerten.
Die Variable dbConn wird in mysql-Klasse geändert auf mysqli

4.3 close

mysql_close($dbConn) => $dbConn->close()

4.4 mysql_select_db

mysql_select_db => $dbConn->select_db

4.5 mysql_query

mysql_query($sql) => $dbConn->($sql)
mysql_query($sql, $this->dbConn) => mysqli_query($this->dbConn, $sql)
Das Rückgabeergebnis von mysqli_query ist false im Fehlerfall. Die Auswertung $query == 0 kann also weiterhin verwendet werden um zu testen ob die Anfrage erfolgreich war.

4.6 mysql_error

mysql_error($dbConn) => $mysqli->error

4.7 mysql_fetch_array

mysql_fetch_array($query) => $mysqli→result→fetch_array()
Prüfen ob man nicht gleich auf fetch_assoc() wechselt wenn anschließend sowieso nur auf assoziative Elemente zugegriffen wird.
mysql_fetch_array($query) => $mysqli→result→fetch_assoc()

4.8 mysql_num_rows

mysql_num_rows($ret[1]) => $this→result→num_rows() or $ret[1]->result->num_rows()

4.9 fast_cases

fast_cases neu bilden weil darin auch die alten mysql Aufrufe drin stehen. Das wäre handisch zu aufwendig zu bewerkstelligen.

4.10 mysql_field_type

mysql_field_type($ret[1], $i) => $field = $this→result→fetch_field_direct($i); $field->type
fetch_field_direct liefert objekt mit der Eigenschaft type

5 Umstellungen in kvwmap

- Es gab vorher in der Klasse rolle hist_timestamp als statische Variable, die aber auch als non-static verwendet wurde. Die non-static Variante wird umbenannt in hist_timestamp_de weil darin die

deutsch formatierte Variante gehalten werden soll.
rolle→hist_timestamp => rolle→hist_timestamp_de muss in allen auch den snippets geändert werden.

- Klassenvariablen sollen jetzt vor der Benutzung im Kopf der Klasse deklariert sein mit var $variablenname
- Wenn auf ein Array mit einem Key zugegriffen wird, den es in dem Array noch nicht gibt die Funktion value_of verwenden. Also z.B.
  statt $formvars['dies_und_das'] => value_of($formvars, 'dies_und_das')
- Die Funktion count liefert einen Fehler wenn das Argument null oder keine Array ist.
  $a = 42;
  $num = count($a);
  liefert z.B. einen Fehler. Es sollte also vorher getestet werden ob es sich um ein Array handelt, z.B.:
  $a = 42;
  $num = (is_array($a) ? count($a) : 0);
  Das geht natürlich nur, wenn $num auch 0 sein soll wenn $a kein Array ist. Vielleicht ist es ja ein Fehler wenn $a kein Array ist und es muss hier eine Exception kommen. Besser ist es wenn vorher dafür gesorgt wird, wenn $a von vorherein ein Array ist, also als array deklariert ist.
  $a = array();
  $a = 42;
  num = count($a);
  In dem Fall würde schon die Zeile $a = 42 einen Fehler liefern, was ja auch so sein soll. count($a) liefert dann aber nie einen Fehler.

6 Fragen und Antworten

Es gibt das Verzeichnis /var/docker/www2/apps.... Dort können wir ja dann unsere Skripte hin kopieren um zu testen.

ja

Du hast uns ja die URL https://geoport-lk-mse.de/kvwmap2 als Beispiel gegeben. Wenn ich die teste, bekomme ich einen Fehler.

Am Anfang das waren Warnungen weil die Constructor Methode von Klassen nicht __construct heißt. Das habe ich jetzt mal für alle Klassen in kvwmap angepasst und auf Eurem kvwmap2 gepullt. Die treten jetzt also nicht mehr auf. Das ist gefixt.

Der Fehler Uncaught Error: Call to undefined function mysql_connect() in /var/www/apps/admin/connect_msp.php:8 Stack trace: #0 /var/www/apps/kvwmap/custom/layouts/snippets/login_29.php(6):
ist jetzt genau das woran Ihr arbeiten könnt. mysql_connect muss umgestellt werden auf die mysqli Variante. Siehe dazu den Abschnitt: "Umstellung von mysql auf mysqli" in der Projektdoku.

In der Datei connect_msp.php muss also

```
$db_link=mysql_connect($server,$nutzer,$password);
if (!$db_link) {
   die('Verbindung nicht möglich : ' . mysql_error());
 }

mysql_select_db($dbname,$db_link);
mysql_query("SET NAMES 'utf8'");
```

ersetzt werden durch:

```
$mysqli = new mysqli($server, $nutzer, $password, $dbname);
if ($mysqli->connect_errno) {
   echo "Failed to connect to MySQL Database" . $dbname . ": (" . $mysqli->connect_errno . ") " . $mysqli-
```

```
>connect_error;
}

$mysqli->query("SET NAMES 'utf8'");
```

In der conf-Datei für kvwmap2 verweist du auch auf /var/www/apps. Müsste da nicht auf /var/www2/apps verwiesen werden?

Der Pfad /var/www/ in der config.php ist korrekt, weil im Container web 2.0.0 die Anwendung ja auch in /var/www/ liegt. siehe auch env_and_volumes_2.0.0
-v /var/docker/www2:/var/www
Von www2 wird in den Kontainer auf www gemappt.

Wie komme ich über einen Browser an die Skripte im Verzeichnis /var/docker/www2?

Wenn man eine Anwendung im web2 Container über Port 80 erreichen möchte bietet sich eine Umleitung im Web Container an, die über einen Eintrag in einer Apache.conf Datei erfolgen kann, so auch für kvwmap2 gemacht erreichbar über:
https://geoport-lk-mse.de/kvwmap2/

```
  ProxyPass /kvwmap2 http://web2-0-0/kvwmap2
  ProxyPassReverse /kvwmap2 http://web2-0-0/kvwmap2
```

Für weitere Zugriffe müssten entsprechend weitere Umleitungen angelegt werden in gisadmin/etc/apache2/site-available und in site-enabled gelinkt und in web-Container service apache2 reload.

Wie komme ich über das Skript dcm in den neuen Container oder geht es nur über docker exec -it?

dcm console web 2.0.0

Anlagenverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis