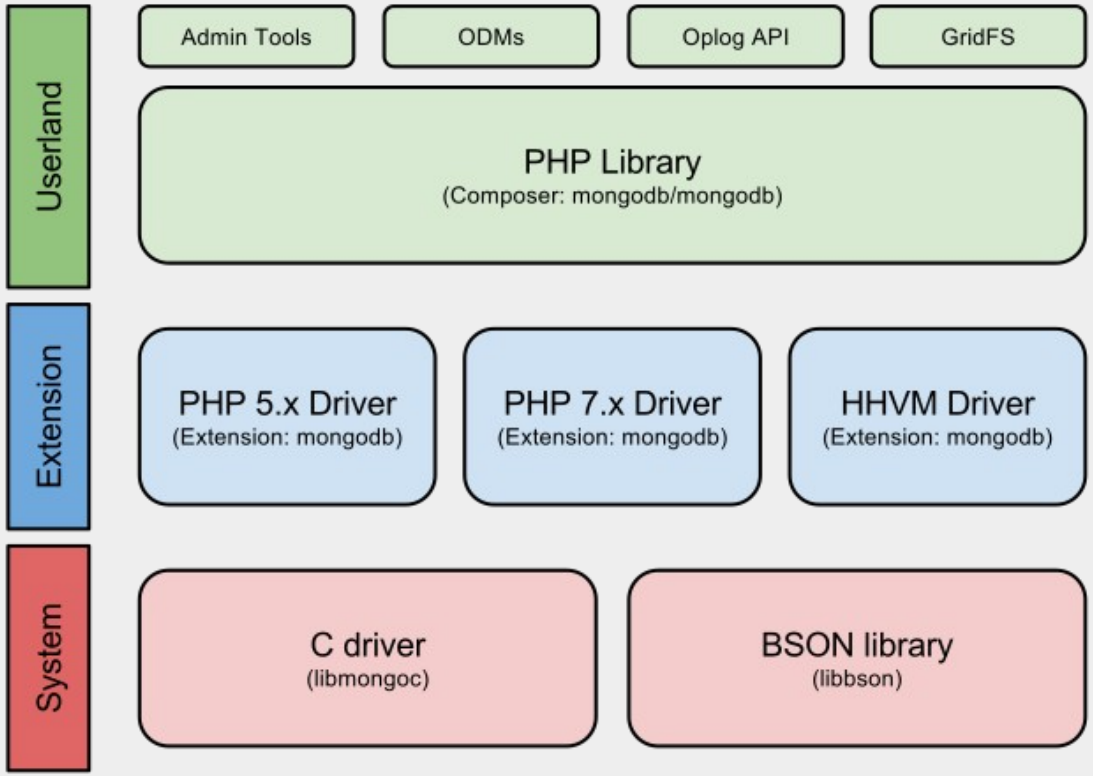
Janusz Starczewski

MongoDB – PHP Driver



Installation the MongoDB PHP driver

Installation with PECL (Unix, Linux, MacOS)

PECL https://pecl.php.net/package/mongodb

1. Installing the MongoDB PHP driver:

```
pecl install mongodb
```

2. Adding the following line to the php.ini file for each environment in which the driver is intended to be used:

extension=mongodb.so

3. Connecting to the MongoDB server using the `MongoDB\Client` class:

```
$mongoClient = new MongoDB\Client('mongodb://localhost:27017');
```

This creates a new instance of the `MongoDB\Client` class, and connects to the MongoDB server running on localhost at the default port 27017.

Installation using Composer

1. Creating a new PHP project and initializing Composer, eg.:

```
composer init
```

This will create a new `composer.json` file in the project directory.

2. Adding the MongoDB PHP driver dependency to `composer.json` file:

```
{
    "require": {
        "mongodb/mongodb": "^1.11"
    }
}
```

This will add the MongoDB PHP driver dependency to the project, specifying version 1.11 or later.

3. Install the MongoDB PHP driver:

```
composer install
```

This will download and install the MongoDB PHP driver and any other dependencies required by the project.

4. Use the MongoDB PHP driver in PHP code, eg.:

```
require_once 'vendor/autoload.php';


use MongoDB\Client;


$mc = new Client('mongodb://localhost:27017');
$db = $mc->selectDatabase('my_database');
$coll = $db->selectCollection('my_collection');
```

Installing Github releases (Windows)

1. Determine the correct archive for your environment and extract the php_mongodb.dll file to the PHP extension directory ("ext" by default).
2. Add the following line to the php.ini:

```
extension = php_mongo.dll
```

Building the MongoDB PHP Driver from source

https://www.php.net/manual/en/mongodb.installation.manual.php

Connection and Selecting a Database

While making a connection `db`, if the database doesn't exist, MongoDB creates it automatically.

Selection is equivalent to `use mydb` in mongosh.

```php
<?php
   // connect to MongoDB
   $mc = new MongoClient();

   echo "Database connection successful";
   // select a database
   $db = $mc->mydb;

   echo "Database selected";
?>
```

Result:

```
Database connection successful
Database selected
```

Creating a Collection

equivalent to `db.createCollection()` in monghosh

```php
<?php
   // connect to MongoDB
   $mc = new MongoClient();
   echo "Database connection successful";

   // select a database
   $db = $mc->mydb;
   echo "Database selected";
   $mycollection = $db->createCollection("mycol");
   echo "Collection created succsessfully";
?>
```

Result:

```
Database connection successful
Database selected
Collection created succsessfully
```

Inserting a Document

equivalent to `db.collection.insertOne()` in monghosh

```php
<?php
   // connect to MongoDB
   $mc = new MongoClient();
   echo "Database connection successful";

   // select a database
   $db = $mc->mydb;
   echo "Database selected";
   $mycollection = $db->mycoll;
   echo "Collection selected";
```

```
   $document = array(
      "title" => "Artificial Intelligence",
      "description" => "paper",
      "score" => 1000,
      "url" => "https://kisi.pcz.pl"
   );

   $mycollection->insert($document);
   echo "Document inserted successfully";
?>
```

Result:

```
Database connection successful
Database selected
Collection selected
Document inserted successfully
```

insertone, insertmany is more appreciated

Selecting All Documents

equivalent to `db.collection.find()` in monghosh

```php
<?php
   // connect to MongoDB
   $mc = new MongoClient();
   echo "Database connection successful";

   // select a database
   $db = $mc->mydb;
   echo "Database selected";
   $mycollection = $db->mycoll;
   echo "Collection selected";
   $cursor = $mycollection->find();
   // iterate cursor to display title of documents

   foreach ($cursor as $document) {
      echo $document["title"] . "\n";
```

```
   }
?>
```

Result:

```
Database connection successful
Database selected
Collection selected {
   "title": "Artificial Intelligence"
}
```

Alternatively you can select a particular field, eg.

```
   $cursor = $mycollection->find(["description" => "paper"]);
```

Updating a Document

equivalent to `db.collection.updateOne( { title: "Artificial Intelligence" },`

```
{
  $set: {
    title: "Machine Learning"
  }
})
```

in monghosh

```php
<?php
   // connect to MongoDB
   $mc = new MongoClient();
   echo "Database connection successful";

   // select a database
   $db = $mc->mydb;
   echo "Database selected";
   $mycollection = $db->mycoll;
   echo "Collection selected";
```

```php
   // to update the document
   $mycollection->update(array("title"=>"Artificial Intelligence"),
      array('$set'=>array("title"=>"Machine Learning")));
   echo "Document updated successfully";

   // to display the updated document
   $cursor = $mycollection->find();

   // iterate cursor to display title of documents
   echo "Updated document";

   foreach ($cursor as $doc) {
      echo $doc["title"] . "\n";
   }
?>
```

Result:

```
Database connection successful
Database selected
Collection selected
Document updated successfully
Updated document {
   "title": "Machine Learning"
}
```

Deleting a Document

equivalent to `db.collection.deleteOne()` in monghosh

```php
<?php
   // connect to MongoDB
   $mc = new MongoClient();
   echo "Database connection successful";

   // select a database
   $db = $mc->mydb;
   echo "Database selected";
   $mycollection = $db->mycoll;
   echo "Collection selected";

   // to remove the document
   $mycollection->remove(array("title"=>"Machine Learning"),false);
   echo "Documents deleted successfully";
```

the second parameter is boolean type
and used for **justOne** field of **remove()** method

```php
    // to display the available documents
    $cursor = $mycollection->find();

    // iterate cursor to display title of documents
    echo "Updated document";

    foreach ($cursor as $doc) {
        echo $doc["title"] . "\n";
    }
?>
```

Result:

```
Database connection successful
Database selected
Collection selected successfully
Documents deleted successfully
```

Remaining MongoDB methods

findOne(),
save(),
limit(),
skip(),
sort() etc.

Libraries for the mongo Extension

Stand-alone Libraries
Mongator ODM
 is an easy, powerful, and ultrafast ODM for PHP and MongoDB. It is a fork of the
Mandango ODM
.

MongoFilesystem
 implements a hierarchical file system using MongoDB as a storage engine. The library uses GridFS for storing the files and a standard collection for the folder information. There is an object-oriented representation of the folders and files in the filesystem and rich API for performing operations. The library also implements renderers for JSON, HTML, and XML.

Mongofill
 is a pure PHP implementation of the mongo extension, which means that it can be used with HHVM. A separate
mongofill-hhvm
 package provides a
libbson
 extension for HHVM, which allows for more performant BSON encoding and decoding.

MongoQueue
 is a PHP queue that allows for moving tasks and jobs into an asynchronous process for completion in the background. The queue is managed by MongoDB.

MongoRecord
 is a PHP MongoDB ORM layer built on top of the mongo PECL extension.

PHPMongo ODM
 is an ODM with support for validation, relations, events, document versioning, and database migrations. Although it is written for the legacy mongo extension, it is tested to work with the mongodb extension using
Mongo PHP Adapter
.

Yamop
 is yet another MongoDB ODM for PHP. It works like the standard MongoDB PHP extension interface but returns objects instead of arrays (as ODM). An integration with
Laravel
 is also available.