MoodleORM Documentation

Release 1.0.0

Andrew Caya

May 15, 2023

CONTENTS

A simple ORM, with a Unit of Work, to easily save cascading data from Moodle Forms, using Moodle Entities, via single database transactions.

MoodleORM Home Page

CHAPTER

ONE

INSTALLATION

1.1 Prerequisites

- Minimum: PHP 7.3, recommended: PHP 8.0, or greater.

1.2 Moodle Plugin Installation

To add the **MoodleORM local plugin** package to your Moodle project, please follow the standard Moodle instructions to install a local plugin.

CHAPTER

TWO

CONFIGURATION

The plugin's configuration is set by passing the following parameters to the `unitofwork`'s constructor upon object instantiation. The constructor's function signature is as follows:

```
public function __construct(\moodle_database $DB, array $classmap, string $maintablename,
↪ int $maintablekey, string $childparentidcolumnname = null, bool $enableconstraints =␣
↪false) {}
```

The first parameter is an instance of the Moodle database abstraction layer (global $DB).

The second parameter is an array containing a classmap of the tables that the Unit of Work should manage for you. The array should have the following layout: the array must contain an array for each database table that should be managed. Each one of these arrays should have an index with the name of the table, that is associated to an array containing these two elements: the fully-qualified class name (FQCN) of the Moodle entity that is to be mapped to the table, and an optional element containing the FQCN of its parent entity.

Here is an example of a classmap for four tables with three parent/child relationships, in concatenation, where one child becomes the parent of the next child:

```
$this->classmap = [
    'simulation_wave' => [
        'class_fqn' => '\local_moodleorm\tests\wave',
        'parent_class_fqn' => '',
    ],
    'simulation_circuit' => [
        'class_fqn' => '\local_moodleorm\tests\circuit',
        'parent_class_fqn' => '\local_moodleorm\tests\wave',
    ],
    'simulation_station' => [
        'class_fqn' => '\local_moodleorm\tests\station',
        'parent_class_fqn' => '\local_moodleorm\tests\circuit',
    ],
    'simulation_substation' => [
        'class_fqn' => '\local_moodleorm\tests\substation',
        'parent_class_fqn' => '\local_moodleorm\tests\station',
    ],
];
```

By default, an entity that does not have a parent is considered to be the fully-managed (CRUD) child of the main table, which, in turn, should always be read-only. But, if need be, one could easily create an entity for the main table and designate it as its own child, thus allowing for single table management (complete CRUD) of the main table. This being said, this would be a limit use case, since accessing the main table through the standard Moodle DBAL might be

a better choice in this case. Again, the full potential of the ORM becomes clear when managing complex relationships between elements of a Moodle form, for example.

Note: The ORM manages 1 -> 1 subordination in its version 1.0.0, and 1 -> N parent to child relationships will be added eventually.

The third parameter is the main table name. This is the read-only reference table that is usually the anchor for a data structure. For example, usually, a course module, or a particular activity, is the anchor point of any form element or data structure in Moodle.

The fourth parameter is the primary key (`id` field) that should be fetched from the main table.

The fifth parameter is optional, and makes it possible to determine the nomenclature of the index that is to be considered as a foreign key to the parent's id for the first level child (an entity with no direct parent FQCN). This is particularly useful when Moodle core tables do not use any particular pattern in their index nomenclature. For second level children and beyond, the nomenclature must be the parent table's name element that follows the last underscore, followed by the expression 'id', without any spaces, underscores, or hyphens. For example, if the parent table's name is 'simulation_circuit', then the name of child's foreign key to the parent id would be 'circuitid'. This is also the default behaviour for first level children if nothing is specified for this parameter.

The sixth and final paramater is also optional and allows the user to enable constraints on all of the children's foreign keys. It is recommended to avoid using constraints with Moodle tables, since it is not considered to be a standard way of handling tables in the Moodle community for now. By default, this parameter is set to `false`.

CHAPTER

THREE

ORM METHODS

MoodleORM has twelve (12) public methods that will help you to work with complex data.

The main public methods are:

```
* export_data()
* save() and,
* commit()
```

3.1 export_data() Method

The `export_data()` method allows you to extract an array of the data that was read from the database, into a format that is perfectly compatible with the ORM's `save()` method.

Note: The `export_data()` will only return data if the ORM's `save()` method was not previously invoked, and the ORM's registry is clean.

The format of the data is fairly straightforward. Each data stucture has a `repositoryname` index, which must match the name of the table in the classmap array. The `data` index contains the fields as defined in the table schema and in the corresponding properties of the Moodle entity. Finally, the array must either contain an `entityid` index, that holds the id of the existing entity, or an `entityuuid` index if the entity is to be inserted into the database. In order to avoid collisions, it is recommended to use the `bin2hex(random_bytes(20))` PHP functions to generate the UUID. If a new entity has a newly created parent entity, it will also be necessary to create a `parentuuid` index and insert the corresponding UUID at this index of the array. Here is an example of a new entity that must be created:

```
$simulationid = $course->cmid;

$uuid = bin2hex(random_bytes(20));

$data[] = [
    'repositoryname' => 'simulation_wave',
    'entityuuid' => $uuid,
    'parentuuid' => null,
    'data' => [
        'name' => 'Wave 2',
        'description' => 'Second wave.',
        'simulationid' => $simulationid,
    ],
];
```

An example of the array's structure when extracting the data from the database would be something like the following example:

```
$data[] = [
    'entityid' => 4,
    'repositoryname' => 'simulation_wave',
    'data' => [
        'name' => 'Wave 2',
        'description' => 'Second wave.',
        'simulationid' => 1,
    ],
];
```

Updating is a question of modifying elements of an entity in the array. Deleting is simply a question of removing the appropriate element from the array.

3.2 save() Method

Once the data array is set, it will be necessary to invoke the `save()` method, in order for the ORM to start tracking changes in the entities. Its Unit of Work will start building a registry of "dirty" entities, that will then be used to define the elements of the transaction that will be sent to the database when invoking its `commit()` method.

3.3 commit() Method

The `commit()` method will clean the dirty registry of entities by running an SQL transaction, with all of the required queries, in order to save the new data to the database.

Note: The `commit()` method will automatically be invoked by the ORM's destructor method if it falls out of scope of the current PHP script.

3.4 Other ORM Methods

The ORM comes with the following helper methods:

- `get_maintable_settings()`, which gives access to the main read-only table's data, based on the given `id`,
- `get_classmap()`, which makes it possible to get the ORM's currently used classmap,
- `get_registry()`, which will return an array containing all of the data, including Moodle entities, that were initially read from the database,
- `get_dirty()`, which will return an array of all of the keys of the new data that require a CRUD action in order to save them to the database,
- `get_stage()`, which will return an array of all of the data, including Moodle entities, that were included in the commit (database transaction),
- `is_committed()`, which will return true if a database transaction was attempted, and false if no persistence action was taken, and
- `is_dirty()`, which will return true if the `save()` method was invoked with new data.

There are also two helper methods for adding or removing foreign key constraints on the foreign keys. These methods are:

- `try_add_cascade_delete_check()`, and
- `try_remove_cascade_delete_check()`.

Note: For a working example on how to start building a Moodle Form with MoodleORM on the back end (without the templates, the CSS, or the JS), please see the 'dist' folder included with each release of MoodleORM.

CHAPTER

FOUR

WHAT'S NEW IN VERSION 1.0.0 (2023-02-27)

- Initial version of the MoodleORM local plugin.

CHAPTER

FIVE

MOODLEORM LICENSE

5.1 Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an

original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   You must give any other recipients of the Work or Derivative Works a copy of this License; and You must cause any modified files to carry prominent notices stating that You changed the files; and You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or

product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

CHAPTER

SIX

INDICES AND TABLES

- genindex
- search

INDEX