|  |
| --- |
| Développeur Web & Web Mobile  PHP et SQL – Niveau 2  Medoo – Votre 1er framework PHP |
| <https://formation.astroware-conception.com>  [contact@astroware-conception.com](mailto:contact@astroware-conception.com) |

# Medoo, un framework ultra léger

Jusqu’à présent vous avez appris à écrire vous-mêmes, à la main, vos requêtes à la base de données depuis PHP, en utilisant soit mysqli soit PDO. C’est la base et ces connaissances sont donc indispensables. Néanmoins, pour se faciliter la tâche et pour fiabiliser cette étape, on a pour habitude d’utiliser des framework. Les gros framework MVC que nous utiliserons plus tard dans la formation (CodeIgniter, Laravel ou Symfony) embarquent leur propre **ORM** (Mapping Objet-Relationnel) servant d’interface entre la base de données et l’application que vous écrivez, vous apprendrez à les utiliser car ils participent grandement au succès de ces frameworks.

**Dans le cadre de cet exercice nous allons apprendre à utiliser un micro-framework ultra léger dédiés aux accès à la base de données – Medoo.**

Présentation de Medoo :

**On parle de framework, mais un framework, ça consiste en quoi ?**

Dans le cas présent, l’intégralité de **Medoo** tient en un seul fichier pesant moins de 40ko. En incluant ce fichier dans votre projet PHP vous profiterez de toute la puissance de **Medoo**. Du fait qu’il soit hyper spécialisé dans un seul domaine (les bases de données), ses 40ko font de **Medoo** l’un des frameworks les plus légers existants, en comparaison des frameworks MVC habituels pesant souvent plusieurs Mo.

Lien vers la page **Get started** de **Medoo** : <https://medoo.in/api/new>

Commencez donc par copier le fichier Medoo.php dans votre projet, puis, dans votre fichier PHP, vous ajouterez les lignes suivantes :

include\_once ("Medoo.php");

use Medoo\Medoo;

A partir de là, vous pourrez faire appel aux classes définies par **Medoo**.

L’étape suivante sera de créer la connexion avec la base de données que vous ferez de la manière suivante :

// Initialisation

$database = new Medoo([

'database\_type' => 'mysql',

'database\_name' => 'name',

'server' => 'localhost',

'username' => 'your\_username',

'password' => 'your\_password'

]);

Votre objet **$database** vous permettra ensuite (comme avec *PDO*) d’exécuter les requêtes, que vous construirez désormais en **programmation orientée objets**.

Par exemple, pour *insérer* un nouvel utilisateur dans une table *users*, on procèdera de la manière suivante :

$database->insert (users, [

'user\_name' => 'tristan\_le\_debutant',

'email' => 'tristandu06@gmail.com'

]);

Pour faire un *select* des champs **user\_name** et **email** de l’utilisateur ayant **l’id 50**, on écrira :

$data = $database->select ('users', [

'user\_name',

'email'

], [

'user\_id' => 50 **// lire : WHERE user\_id = 50**

]);

Vous trouverez toute l’aide et les exemples nécessaires pour écrire n’importe quelle requête sous cette forme dans la documentation de Medoo : <https://medoo.in/doc>

Le bénéfice de ce framework, vous devez déjà le pressentir : en n’écrivant pas vous-même le code **SQL** des requêtes et en laissant **Medoo** le générer pour vous, vous ajoutez une barrière de protection supplémentaire contre les **injections SQL**. C’est à la charge du framework de vous garantir que les paramètres que vous donnerez à vos requêtes seront utilisés sans le moindre risque d’injection. Pas besoin de traiter les paramètres, vous pouvez les utiliser tels quels depuis un formulaire, **Medoo** vous protégera de ce type d’injections.

Le fait d’encapsuler le requêtage propose un autre avantage immédiat – en utilisant le modèle objet de **Medoo**, vous n’avez pas à vous inquiéter de la syntaxe SQL. Si vous utilisez correctement les fonctions issues de **Medoo**, les requêtes que vous écrirez seront syntaxiquement correctes, et si des erreurs sont générées par le SGBD, vous pourrez facilement les traiter via **Medoo**.

Enfin, un détail auquel vous n’aurez pas forcément pensé, mais d’un SGBD à un autre, la syntaxe **SQL** utilisée peut varier un peu. Ainsi un site développé en local avec une base de données MySQL ne fonctionnera probablement pas tel quel si le serveur ne dispose que de PostgreSQL. Il faudra probablement adapter quelques requêtes. En passant par la couche d’abstraction offerte par **Medoo**, vous vous affranchissez de cette étape car c’est le framework qui va s’assurer de générer les requêtes compatibles avec le SGBD installé, tout simplement.

Travail à faire :

Le travail à réaliser est simple. Si vous avez terminé les exercices du niveau 2, faîtes en une copie et reprenez-les.

Vous commencerez par écrire un fichier **database.php** qui importera **Medoo** et qui initialisera la connexion à votre base de données via un objet **$database** (voir exemple ci-dessus).

Ensuite vous reprendrez chacun des exercices que vous avez réalisés, et vous les migrerez vers **Medoo**. Vous inclurez le fichier **database.php** que vous avez écrit, et ensuite vous remplacerez toutes les requêtes PDO/mysqli que vous avez écrites par l’équivalent proposé par **Medoo**. Vous ne devez utiliser **QUE** **Medoo** pour faire toutes les requêtes.

Si vous n’aviez pas terminé les exercices du niveau 2, mettez la fin de cet exercice en pause et faîtes dès maintenant la migration vers **Medoo**. Une fois la migration terminée, vous terminerez les exercices du niveau 2, en utilisant exclusivement **Medoo** bien entendu.

Dans le cadre de vos prochains exercices ou projets, vous serez libres, si vous le souhaitez, de continuer à utiliser **Medoo** à chaque fois. Son utilisation ne se justifiera pas forcément quand vous utiliserez un autre framework **MVC** proposant déjà un **ORM**, mais le reste du temps, n’hésitez vraiment pas ! **Les frameworks sont là pour ça.**