

Présentation CodeIgniter

Histoire de CodeIgniter

2006 : Création du framework

Créé par Rick Ellis, PDG d'EllisLab, CodeIgniter est né le 28 février 2006. Issu d'ExpressionEngine, un CMS développé par la même société, CodeIgniter visait à offrir une alternative simple et performante pour le développement d'applications web.

2006-2019 : Mise à jour fréquente de CodeIgniter

Au fil des années, CodeIgniter a connu une adoption rapide et une popularité croissante au sein de la communauté PHP. Sa simplicité d'utilisation, sa documentation claire et sa large communauté de contributeurs ont largement contribué à son succès.

Versions majeures et évolutions notables :

- **2007** : Version 1.0, introduction du pattern MVC et de la bibliothèque de base de données
- **2008** : Version 1.5, ajout de la gestion des sessions, des helpers et de la validation de formulaires
- **2011** : Version 2.0, refonte majeure du framework avec l'intégration de l'autoload, des hooks et des tests unitaires
- **2012-2019** : Versions 2.1 à 3.1, enrichissement continu avec de nouvelles fonctionnalités et améliorations de performance

2020 : Nouvelle ère avec CodeIgniter 4

Sortie le 24 février 2020, cette version majeure représente une refonte complète du framework. CodeIgniter 4 s'appuie sur les principes fondamentaux de ses prédécesseurs tout en intégrant des technologies modernes et des pratiques de développement actuelles.

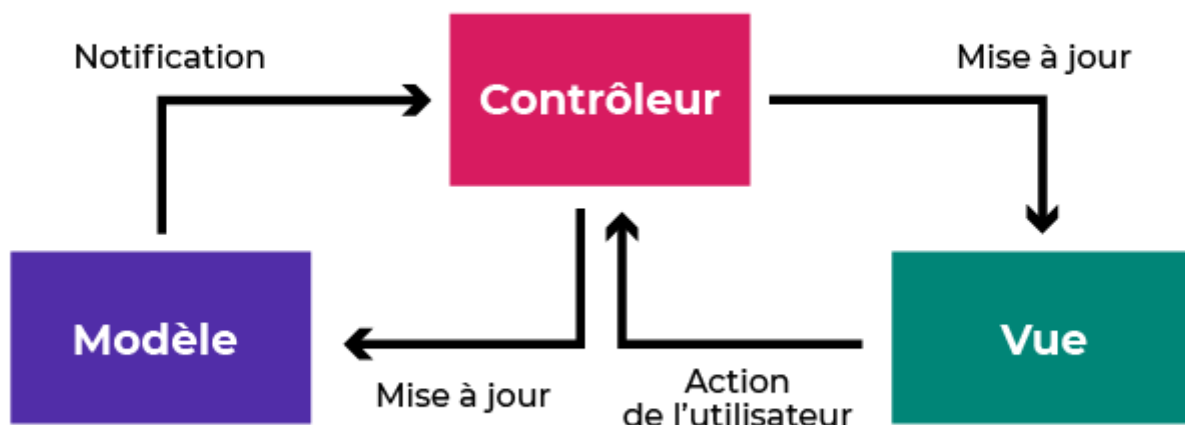
Points clés de la version 4 :

- Architecture MVC améliorée: meilleure séparation des couches et structure plus flexible
- Performances optimisées: code plus léger et utilisation efficace des ressources
- Nouvelles fonctionnalités: support de PSR-4, gestion des routes améliorée, sécurité renforcée
- Compatibilité PHP 7+: utilisation des dernières fonctionnalités du langage PHP

C'est justement cette version 4 de CodeIgniter qui est utilisé pour le développement de la plateforme Castopod.

Présentation de l'architecture MVC

L'architecture MVC (Model-View-Controller) est un modèle de conception logicielle largement utilisé pour le développement d'interfaces utilisateur. Ce modèle vise à séparer les différentes préoccupations d'une application en trois composants distincts :



1. Modèle (Model) :

Le modèle représente la couche métier de l'application. Il encapsule la logique métier, les données et les règles de fonctionnement. Le modèle est responsable de la gestion des données, des calculs et de la validation des informations.

2. Vue (View) :

La vue est la couche de présentation de l'application. Elle est responsable de l'affichage des données et de l'interaction avec l'utilisateur. La vue utilise les données du modèle pour générer l'interface utilisateur et reçoit les interactions de l'utilisateur pour les transmettre au contrôleur.

3. Contrôleur (Controller) :

Le contrôleur agit comme un intermédiaire entre le modèle et la vue. Il reçoit les requêtes de l'utilisateur, les traite en utilisant le modèle et met à jour la vue en conséquence. Le contrôleur est responsable de la gestion du flux de l'application et de la coordination des interactions entre le modèle et la vue.

Organisation d'un projet Codeigniter

Dossier principal :

Le point d'entrée de votre application Codeigniter se trouve dans le dossier principal. Il contient les fichiers suivants :

- `index.php` : Démarre l'application et charge le framework Codeigniter.
- `config.php` : Contient les configurations de base de l'application (par exemple, la base de données, les routes).
- `autoload.php` : Détermine les classes et les helpers à charger automatiquement.

Structure MVC :

- **Dossier app**: Contient les différents composants de l'architecture MVC :
 - `Controllers` : Contient les contrôleurs de l'application. Chaque contrôleur est responsable d'une partie de la logique métier et de l'interaction avec l'utilisateur.
 - `Models` : Contient les modèles de l'application. Chaque modèle représente une entité métier et encapsule la logique d'accès aux données.
 - `Views` : Contient les vues de l'application. Chaque vue est responsable de l'affichage d'une partie de l'interface utilisateur.

Fichiers importants :

- `Routes.php` : Définit les règles de routage des URL vers les contrôleurs et les actions.
- `Application.php` : Classe principale de l'application qui gère le flux de l'application.
- `Controller.php` : Classe de base pour tous les contrôleurs.
- `Model.php` : Classe de base pour tous les modèles.
- `View.php` : Classe de base pour toutes les vues.

Exemple de structure d'un projet basique :

```
app/  
├─ Controllers/  
│   └─ Home.php  
├─ Models/  
│   └─ User.php  
└─ Views/  
    ├─ home/  
    │   └─ index.php  
    └─ user/  
        └─ profile.php  
system/
```

CodeIgniter et Castopod

Vous pouvez retrouver une explication détaillée de l'organisation des fichiers pour Castopod dans la section [Organisation des fichiers](#). Cette section explique comment les fichiers sont organisés selon l'architecture MVC pour le développement de Castopod.

Source

- [Documentation officielle de CodeIgniter](#)
- [Page Wikipédia de CodeIgniter](#)

Revision #6

Created 9 February 2024 14:57:16 by tanguy

Updated 10 February 2024 15:45:08 by Emilien