

ML36 - 3 JOURS

PHP : développer des applications web dynamiques

Créer des applications PHP modernes, sécurisées et maintenables : formulaires, objets, sessions, API et bonnes pratiques de développement.

1 Langage

2 Données

3 Sécurité

4 API

DURÉE

3 jours (21h)

NIVEAU

Débutant et +

FORMAT

Présentiel / Visio

PARCOURS ACTUALISÉ 2026

PHP 8.5 • Composer & autoload • PDO / MySQL • POO • sécurité web • sessions • API REST • qualité & maintenance

Introduction

PHP reste l'un des langages les plus utilisés pour développer des applications web côté serveur. Le langage a fortement évolué : typage plus précis, programmation orientée objet, gestion moderne des dépendances, performances améliorées et exigences de sécurité renforcées font désormais partie des pratiques attendues d'un développement professionnel.

Cette formation de trois jours permet de construire une application dynamique de bout en bout : traitement de formulaires, manipulation de données, accès à une base MySQL via PDO, sessions, authentification simple, organisation orientée objet, échanges JSON et consommation d'API. Le parcours privilégie les pratiques transférables plutôt que la mémorisation de syntaxe.

Le programme est actualisé pour les branches PHP 8.x maintenues, avec PHP 8.5 comme version stable de référence en septembre 2026. Les nouveautés du langage sont présentées lorsqu'elles améliorent la lisibilité, la sécurité ou la maintenabilité, sans transformer la formation en cours de migration de version.

RÉFÉRENCE

ML36

DURÉE

3 jours (21h)

NIVEAU

Débutant et +

LIEU

Toute la France Présentiel / visio

TYPE

Collective Intra / inter-entreprise

ACCÈS

À la demande Délai indicatif : 3 semaines

POSITIONNEMENT DU PARCOURS

Une formation PHP moderne orientée pratique : comprendre le langage, structurer le code, sécuriser les entrées, connecter une base de données et exposer ou consommer des données via API.

Objectifs pédagogiques

Des compétences concrètes pour développer et maintenir une application PHP dynamique en respectant les bonnes pratiques actuelles.

- 01** Maîtriser les fondamentaux PHP
Écrire des scripts clairs avec variables, types, conditions, boucles, fonctions, tableaux et gestion des erreurs.
- 02** Traiter les entrées web
Récupérer, valider et filtrer les données issues de formulaires, requêtes HTTP et fichiers.
- 03** Structurer le code en POO
Utiliser classes, objets, propriétés, méthodes, namespaces, exceptions et autoloading.
- 04** Travailler avec MySQL
Lire et modifier des données avec PDO et des requêtes préparées, gérer transactions et erreurs.
- 05** Sécuriser une application
Prévenir injections SQL, XSS, CSRF, mauvaise gestion des sessions et fuites de données sensibles.
- 06** Échanger avec des API
Consommer et produire des données JSON, comprendre les principes REST, statuts HTTP et authentification.
- 07** Adopter un workflow professionnel
Utiliser Composer, conventions PSR, logs, configuration séparée et premiers réflexes de qualité / maintenance.

POUR QUI ?

Développeurs web débutants, intégrateurs, webmasters techniques, profils IT ou toute personne souhaitant développer des fonctionnalités serveur en PHP.

PRÉREQUIS

Connaître HTML et CSS et être à l'aise avec les notions de base du Web. Une première expérience en programmation est utile mais non obligatoire.

Écrire du PHP moderne et traiter les requêtes web

Construire une base solide : syntaxe, types, fonctions, formulaires, validation et outils de développement.

1 Environnement & syntaxe

- Comprendre le rôle de PHP côté serveur et le cycle requête / réponse.
- Installer ou utiliser un environnement local : PHP, serveur web, éditeur et terminal.
- Variables, constantes, types, opérateurs et conversions.
- Conditions, boucles, tableaux et fonctions.
- Typage des paramètres / retours et lecture des messages d'erreur.
- Découvrir les fonctions récentes utiles des branches PHP 8.x.

2 HTTP, formulaires & fichiers

- Comprendre GET, POST, en-têtes et codes de statut.
- Récupérer les données d'un formulaire et contrôler leur présence.
- Valider, filtrer et normaliser les entrées utilisateur.
- Gérer upload de fichiers, limites et contrôles essentiels.
- Manipuler chaînes, dates et données structurées.
- Éviter l'affichage direct de données non échappées.

ATELIER PRATIQUE

Créer un formulaire complet avec validation serveur, messages d'erreur, conservation des valeurs saisies et traitement sécurisé des données.

- 01** Variables d'environnement
Séparer configuration, secrets et code applicatif.

- 02** Erreurs & exceptions
Afficher en développement, journaliser en production.

- 03** Lisibilité
Nommer clairement, découper les fonctions et éviter le code dupliqué.

Organiser son code avec Composer et la programmation objet

Passer du script isolé à une application structurée, réutilisable et plus facile à maintenir.

3 Composer & autoload

- Comprendre dépendances et gestion de paquets.
- Initialiser un projet Composer et lire composer.json.
- Utiliser l'autoloading PSR-4.
- Organiser dossiers src, public, config et ressources.
- Installer une bibliothèque adaptée à un besoin simple.
- Verrouiller les versions avec composer.lock et mettre à jour avec prudence.

4 Programmation orientée objet

- Créer classes, objets, propriétés et méthodes.
- Constructeurs, visibilité et propriétés typées.
- Héritage, interfaces et composition : quand les utiliser.
- Namespaces et organisation du domaine applicatif.
- Exceptions pour gérer les situations anormales.
- Découvrir readonly, enums et autres outils modernes lorsque le cas le justifie.

ATELIER PRATIQUE

Refactorer le formulaire du matin en petites classes : validation, service métier et représentation d'une entité simple, chargées automatiquement via Composer.

01

Responsabilité unique - Une classe ou fonction doit avoir un rôle clair.

02

Dépendances - Éviter les variables globales et rendre les besoins explicites.

03

Exceptions - Remonter une erreur exploitable sans exposer d'informations sensibles.

04

Standards - S'appuyer sur PSR-4 et PSR-12 pour une base de code cohérente.

Connecter PHP à MySQL avec PDO

Lire, écrire et fiabiliser les échanges avec une base de données sans exposer l'application aux injections SQL.

5 PDO & requêtes préparées

- Créer une connexion PDO et configurer les erreurs.
- Exécuter SELECT, INSERT, UPDATE et DELETE.
- Préparer les requêtes et lier les paramètres.
- Récupérer les résultats sous une forme exploitable.
- Gérer les erreurs sans révéler les détails de connexion.
- Comprendre pourquoi concaténer une entrée utilisateur dans SQL est dangereux.

6 Transactions & couche d'accès

- Comprendre transaction, commit et rollback.
- Utiliser une transaction pour plusieurs opérations cohérentes.
- Séparer accès aux données, logique métier et affichage.
- Créer un repository / service simple sans sur-architecture.
- Gérer pagination, recherche et filtres basiques.
- Identifier les problèmes classiques de performance : N+1, requêtes inutiles et index manquants.

ATELIER PRATIQUE

Créer un mini-CRUD sécurisé : liste, création, modification et suppression d'enregistrements avec PDO, requêtes préparées, validation et messages d'erreur adaptés.

- 01** Préparer
Paramètres séparés du SQL : aucune concaténation de données utilisateur.

- 02** Valider
Contrôler les types, longueurs et règles métier avant l'écriture.

- 03** Journaliser
Tracer les erreurs techniques sans les afficher à l'utilisateur.

- 04** Transaction
Regrouper les opérations qui doivent réussir ou échouer ensemble.

Sessions, authentication et protections indispensables

Appliquer les réflexes de sécurité attendus dans une application web PHP en production.

7 Sessions & authentication

- Démarrer et configurer une session correctement.
- Créer un flux de connexion / déconnexion simple.
- Hasher et vérifier les mots de passe avec les fonctions dédiées.
- Régénérer l'identifiant de session après authentication.
- Gérer rôles ou autorisations simples côté serveur.
- Éviter de stocker des informations sensibles inutilement en session.

8 Risques web courants

- Injection SQL : requêtes préparées et validation.
- XSS : encodage de sortie selon le contexte HTML.
- CSRF : jetons sur les actions sensibles.
- Upload : type, taille, nommage et stockage hors exécution.
- Secrets : variables d'environnement, jamais dans le dépôt.
- Logs : ne pas exposer mots de passe, tokens ou données personnelles.

ATELIER PRATIQUE

Ajouter à l'application une authentication simple, protéger une zone privée et sécuriser une action de modification avec contrôle d'autorisation et jeton CSRF.

01

Entrée - Valider et normaliser ce qui entre dans l'application.

02

Traitement - Utiliser requêtes préparées et règles métier explicites.

03

Sortie - Échapper les données au moment où elles sont rendues.

04

Secret - Séparer clés, mots de passe et configuration du code source.

Échanger des données et préparer une application maintenable

Manipuler JSON, comprendre REST et adopter les réflexes de qualité nécessaires avant déploiement.

9 JSON & API REST

- Encoder et décoder du JSON proprement.
- Comprendre ressources, méthodes HTTP et codes de statut.
- Consommer une API avec cURL ou une bibliothèque HTTP.
- Envoyer headers, paramètres et jetons d'authentification.
- Créer un endpoint JSON simple dans l'application.
- Valider les entrées et produire des réponses d'erreur cohérentes.

10 Qualité & maintenance

- Appliquer une convention de code cohérente (PSR-12).
- Séparer configuration développement / production.
- Utiliser logs et gestion centralisée des erreurs.
- Comprendre les bases des tests automatisés et de l'analyse statique.
- Verrouiller et auditer les dépendances Composer.
- Préparer une checklist de mise en production et de mise à jour.

ATELIER PRATIQUE

Consommer une API externe puis exposer une ressource JSON issue de l'application. Contrôler statuts HTTP, erreurs, authentification éventuelle et données renvoyées.

- 01** API externe
Récupérer des données, gérer délai, erreur et format inattendu.

- 02** Endpoint interne
Retourner une réponse JSON avec statut HTTP adapté.

- 03** Dépendances
Contrôler les paquets installés et éviter les versions obsolètes.

- 04** Production
Désactiver l'affichage des erreurs, journaliser et protéger les secrets.

Assembler une mini-application PHP professionnelle

Mettre en pratique l'ensemble du parcours dans un cas complet et directement réutilisable comme trame de travail.

- 01** Structurer le projet
Composer, autoload PSR-4, configuration et organisation des dossiers.

- 02** Créer le modèle de données
Connexion PDO, requêtes préparées et opérations CRUD.

- 03** Gérer les formulaires
Validation serveur, messages d'erreur et protection des entrées.

- 04** Sécuriser les accès
Session, connexion, autorisation simple et protection CSRF.

- 05** Ajouter une API
Consommer ou exposer une ressource JSON avec codes HTTP cohérents.

- 06** Préparer la production
Logs, secrets, dépendances, checklist sécurité et maintenance.

LIVRABLES REMIS AUX PARTICIPANTS

- 01** Support PHP 8.x - Syntaxe, POO, PDO, sécurité, sessions, API et bonnes pratiques.
- 02** Checklist sécurité PHP - SQL, XSS, CSRF, sessions, secrets, uploads et logs.
- 03** Starter project Composer - Arborescence de référence avec autoload et configuration séparée.
- 04** Fiche PDO & API - Requêtes préparées, transactions, JSON et codes HTTP essentiels.
- 05** Checklist mise en production - Erreurs, configuration, dépendances, secrets et contrôles avant déploiement.

Organisation pédagogique

Des modalités pédagogiques de formation, de suivi et d'évaluation pour un parcours technique fortement orienté pratique

1

Prérequis

Connaître HTML / CSS et les principes de base du Web. Une première expérience en programmation est utile mais non obligatoire. Chaque participant doit disposer d'un poste permettant d'exécuter PHP et MySQL ou d'un environnement fourni pour les exercices.

2

Moyens pédagogiques et techniques

Présentiel : salle équipée, connexion Internet, projection et exercices sur poste. Distanciel : visioconférence, partage d'écran et environnement de développement accessible. Alternance d'explications, démonstrations, code en direct et exercices.

3

Supports et outils

PHP 8.x maintenu, éditeur de code / IDE, Composer, serveur local ou conteneur selon contexte, MySQL/MariaDB, navigateur, client API, dépôt d'exercices et fichiers de démarrage.

4

Suivi et évaluation

Questionnaire de positionnement, émargements, exercices pratiques, revues de code, quiz ou QCM, projet final, validation de la checklist sécurité, enquêtes de satisfaction à chaud et à froid et certificat de réalisation.

ÉQUIPE PÉDAGOGIQUE & LES PLUS

Intervenant développeur PHP / backend ou consultant web disposant d'une pratique professionnelle du développement serveur, des bases de données, des API et de la sécurité applicative.

Les plus : PHP 8.x actuel • 70 % pratique / code • PDO & sécurité au cœur du parcours • Composer & standards PSR • API REST • projet final complet

CONTACT & ACCESSIBILITÉ

Julien Petitjean - Responsable administratif / référent handicap | contact@target-formation.fr | 05 25 33 18 04

Délai d'accès : à la demande du client, sous un délai indicatif de trois semaines. Formation à distance ou en présentiel, en inter ou intra-entreprise. Adaptation possible selon les besoins identifiés en situation de handicap.