

# Programme symfony Workshop Débutant

## ***Du PHP « à plat » à la structure MVC***

- Comprendre la séparation en couches
- Mettre la logique métier dans des classes
- Lire les paramètres d'une requête et les passer à la vue
- Utiliser la syntaxe PHP alternative pour les templates
- Découper un projet en niveaux application, module, action
- Naviguer dans l'arborescence des fichiers d'un projet symfony
- Utiliser un décorateur pour factoriser les éléments récurrents de la vue
- Lier des pages entre elles
- Comprendre la théorie du design pattern MVC

## ***Le templating avec symfony***

- Utiliser un helper et un helper group
- Construire un formulaire avec des helpers
- Concevoir des templates modulaires avec les partials, slots, components, et le layout
- Explorer les helpers disponibles (texte, nombre, date)
- Utiliser le view.yml et les méthodes de la réponse pour modifier la vue

## ***L'environnement de travail symfony***

- Utiliser plusieurs environnements de développement
- Comprendre le déroulement d'une requête avec la web debug toolbar
- Trouver plus d'information sur les requêtes avec les fichiers de log
- Comprendre cascade de configuration
- Apprendre la syntaxe YAML
- Comprendre le cache de la configuration et symfony cc
- Explorer les fichiers de configuration (settings.yml, config.php, app.yml)
- Utiliser la ligne de commande symfony
- Débugger une application symfony (traces, log\_message, die)

## ***Le routing et le protocole HTTP***

- Comprendre ce qu'est une URL
- Distinguer un get et un post
- Utiliser Apache et le rewriting d'URL pour les requêtes entrantes
- Utiliser un helper pour formater les requêtes sortantes
- Comprendre le front controller
- Comprendre le routing bidirectionnel
- Savoir écrire un routing.yml (ordre, tokens, valeurs par défaut, requirements)
- Distinguer un redirect et un forward
- Ajouter une extension à une page
- Utiliser les règles nommées
- Utiliser le routing pour gérer une arborescence de navigation

## ***L'abstraction Objet-Relationnel***

- Manipuler des objets métier
- Se protéger des injections SQL
- Abstraire le SQL pour pouvoir changer de base de données
- Ecrire un schéma relationnel pour Propel (schema.yml)
- Connecter une application à une base de données (propel.ini, databases.yml)
- Générer un modèle objet (propel-build-model, propel-build-sql, propel-build-schema)
- Utiliser les classes générées par Propel (getters, setters, méthodes peer)
- Utiliser les raccourcis de Propel pour les relations entre objets
- Utiliser l'objet Criteria pour requêter la base de données
- Traduire une requête SQL en requêtage objet
- Refactoriser le code d'une action pour le mettre dans le modèle

## ***L'installation du framework***

- Installer symfony depuis une sandbox, PEAR, SVN
- Comprendre l'organisation du repository SVN de symfony
- Initialiser un projet, un module
- Configurer un serveur web
- Initialiser un suivi de version
- Mettre à jour symfony
- Installer symfony sur un serveur de production
- Synchroniser deux installations

## ***L'admin generator***

- Générer ou initier un module
- Comprendre la différence entre un scaffolding et une administration
- Initier une administration
- Explorer le code généré
- Utiliser le generator.yml pour modifier le code généré
- Modifier la vue list (champs affichés, pagination, filtres, tooltips, actions, partial columns)
- Modifier la vue edit (admin-tags, action)
- Modifier les classes générées pour gérer un edit non standard
- Personnaliser des templates
- Utiliser des thèmes
- Comprendre les templates de templates

## ***Les tests***

- Valider un élément de code avec un test unitaire
- Utiliser lime et la ligne de commande symfony test
- Alimenter une base de test avec des fixtures
- Valider une fonctionnalité avec un test fonctionnel
- Utiliser l'objet sfTestBrowser pour simuler une navigation entre des pages
- Vérifier un élément de page avec le Dom CSS Selector
- Automatiser les tests

## ***Le cache***

- Mettre le résultat d'une requête (la vue) en cache
- Mettre en cache une page avec ou sans layout
- Mettre en cache un composant de la vue (partial, component)
- Explorer l'arborescence des fichiers cache

## ***L'internationalisation et la localisation***

- Identifier les textes de l'interface à traduire (\_\_( ))
- Traduire une interface avec un dictionnaire XLIFF
- Changer la culture de l'utilisateur
- Formater automatiquement une date, un nombre, un montant
- Stocker des informations différentes selon la localisation