Fiche de Procédure : Les Bases de Symfony

1. Qu'est-ce que Symfony ?

- Symfony est un framework PHP open source, conçu pour développer des sites et applications web complexes, robustes, évolutifs et performants.
- Il repose sur une architecture MVC (Modèle-Vue-Contrôleur) et regroupe de nombreux composants PHP réutilisables, adoptés dans de nombreux projets et CMS (Drupal, Magento, etc.).
- Développé par la société française SensioLabs, c'est l'un des frameworks PHP les plus utilisés au monde, soutenu par une grande communauté.

2. Fonctionnalités principales

- Composants réutilisables : Symfony propose des dizaines de composants autonomes (formulaires, sécurité, routage, validation, etc.) utilisables ensemble ou séparément.
- Système de bundles : Permet d'ajouter des fonctionnalités modulaires à une application.
- Moteur de templates Twig : Pour séparer la logique métier de l'affichage et faciliter la génération de vues dynamiques.
- Gestion avancée des formulaires et validation : Génération, affichage, traitement et validation des formulaires côté serveur.
- Sécurité : Authentification, autorisation, gestion des utilisateurs, protection contre CSRF et injections SQL.
- ORM Doctrine : Gestion de la base de données via des entités PHP, migrations, requêtes, etc.
- Performances et cache : Système de cache HTTP et bytecode pour accélérer les applications.
- Tests et outils de debug : Outils intégrés pour tester, profiler et déboguer le code.
- Internationalisation : Prise en charge native de la traduction et de la gestion multilingue.
- Automatisation et CI/CD : Intégration facilitée avec des outils d'automatisation, pipelines de tests et déploiement.

3. Comment démarrer un projet Symfony ?

1. Pré-requis :
    a. PHP 8+ (selon la version de Symfony).
    b. Composer (gestionnaire de dépendances PHP).
    c. Un serveur web (Apache, Nginx, ou serveur intégré PHP).
    d. (Facultatif) Docker pour un environnement reproductible.
2. Création du projet :

```
composer create-project symfony/skeleton nom-du-projet
```

3. Structure de base :
    a. `/src` : code source (contrôleurs, entités, services…)
    b. `/templates` : vues Twig
    c. `/config` : configuration (routes, services, sécurité…)
    d. `/public` : point d'entrée web (index.php)
    e. `/var` et `/vendor` : cache, logs, dépendances
4. Lancer le serveur de développement :

```
symfony server:start
```

Ou

```
php -S localhost:8000 -t public
```

4. Bonnes pratiques et astuces

- Respecter la structure et les conventions du framework pour faciliter la maintenance et le travail en équipe.
- Gérer les dépendances avec Composer et maintenir les bundles à jour.
- Utiliser l'ORM Doctrine pour la gestion de la base de données, avec des migrations et des fixtures pour les données de test.
- Automatiser les tests et le déploiement avec des outils CI/CD (GitLab, GitHub Actions…).
- Sécuriser l'application : protéger les informations sensibles, surveiller les dépendances, appliquer les correctifs de sécurité.
- Documenter le code et utiliser les outils de debug et de profiling intégrés.
- Rester à jour avec les dernières versions de Symfony pour bénéficier des nouveautés et des correctifs.

5. Ressources pour progresser

- Documentation officielle Symfony (guides, tutoriels, recettes).
- Communauté active, forums, Slack, Stack Overflow.
- Formations, articles et vidéos spécialisés.
- Exemples de projets open source utilisant Symfony.

Astuce : Symfony est un framework puissant et modulaire. Commencez par un projet simple, explorez les composants, et appuyez-vous sur la documentation et la communauté pour monter en compétence rapidement.