

## Introduction à la programmation Perl : Travaux Pratiques - 4 jours

*formation 431*

- Vous apprendrez à**
- Créer rapidement des scripts Perl efficaces et réutilisables
  - Tirer parti des nombreux opérateurs de Perl 5 et des fonctions intégrées
  - Structurer du code avec des sous-routines utilisateurs
  - Exploiter les nombreux packages objet supplémentaires disponibles sur Internet
  - Créer des scripts d'administration qui peuvent être utilisés sur des plates-formes UNIX, Linux et Win32
  - Réaliser avec facilité des tâches complexes de manipulation de données

**Objectif** Perl, langage de script orienté objet, offre de nombreuses possibilités et est largement utilisé avec UNIX, Linux, Win32 et Internet. Durant ce cours, vous apprendrez à écrire des scripts réutilisables avec Perl 5. Grâce aux exercices, vous apprendrez à utiliser Perl dans vos environnements système et vos applications et à utiliser les fonctions intégrées dans le langage et les modules externes.

**À qui s'adresse cette formation** Administrateurs d'UNIX, de Linux et de Windows, ingénieurs logiciel, programmeurs et utilisateurs avertis. Une expérience professionnelle d'un langage procédural ou objet est supposée acquise. Une connaissance des outils de scripts UNIX et des expressions régulières est utile.

**Travaux pratiques** Durant ce cours, des exercices vous apportent une expérience approfondie et renforcent chaque concept présenté :

- Analyse et manipulation de texte avec des expressions régulières de Perl
- Lecture et écriture de fichiers de données et des flux E/S standard
- Extraction/organisation d'information à partir de fichiers multiples
- Amélioration de la puissance des scripts en utilisant les modules objet fournis et de contributeurs
- Écriture de scripts réseau pour accéder à la base de données de messagerie et aux serveurs Web

## Introduction à la programmation Perl : Travaux Pratiques - 4 jours

formation 431

### Introduction à Perl

- Points forts : facilité de programmation avec les opérateurs, souplesse, rapidité d'exécution
- Construction d'outils réutilisables : administration système, manipulation de texte, Internet
- Domaines d'application courants : filtrage des textes, applications réseau, programmation d'applications Web

### Concepts fondamentaux de la syntaxe Perl

#### Exécutions de programmes Perl

- Invocation de scripts Perl sous UNIX/Linux avec la syntaxe "shebang"
- Démarrage de scripts Perl sous Win32 par association ou ligne de commande
- Aide avec **perldoc**

#### Types de variables et contextes

- Scalaire, listes et hachages
- Chaînes de caractères, interpolation dans les chaînes
- Variables spéciales
- Intégrer la notion de contexte

#### Composer des structures de données

- Création de références à des variables nommées
- Création de références à des données anonymes
- Construction de tableaux multidimensionnels
- Utilisation de tables de hachage multidimensionnelles

#### Gestion des fichiers et des entrées de l'utilisateur

- Gestion des E/S standard
- Définition et utilisation des descripteurs de fichiers
- Analyse des arguments de ligne de commande
- Lecture et écriture de fichiers de données

#### Concordance de motifs et opérateurs

##### Expressions régulières Perl

- Extraction d'informations textuelles importantes
- Utilisation d'expressions régulières UNIX
- Modification des données avec des substitutions

- Concordances globales et insensibles à la casse

#### Les groupes d'opérateurs de Perl

- Manipulation d'expressions arithmétiques
- Réplication et augmentation des chaînes
- Rapidité grâce aux opérateurs d'affectation
- Obtention d'attributs de fichier
- Prise de décision avec les opérateurs logiques
- Mise en place et utilisation d'intervalles

#### Boucles, décisions et contrôle de flux

##### Constructions de contrôle de flux traditionnelles

- Prise de décisions avec **if/else/elsif**
- Création de boucles avec **do, while, until, for** et **for each**

##### Constructions spécifiques à Perl

- **if** et **unless** comme modificateurs d'instructions
- Contrôle du flux avec **next** et **last**
- Simulation de l'instruction **switch**

#### Sous-routines et modules

##### Écriture de sous-routines

- Définition et appel de sous-routines
- Passage et lecture des paramètres
- Retour de valeurs à l'appelant

##### Rendre les données fonctionnelles

- Localisation des données : **my** et **local**
- Accès aux variables globales
- Extraction de variables locales avec **shift**

#### Fonctions intégrées et ajouts

##### Besoins courants

- Traitement de chaînes
- Traitement des tableaux avec les fonctions de liste
- Organisation des informations avec **sort**
- Tri des données à partir de champs multiples

##### E/S et construction d'outils

- Manipulation des entrées du système de fichiers
- Lecture de fichiers binaires
- Dissection et création d'enregistrements avec **split** et **join**
- Formatage des sorties

#### Perl et la technologie objet

##### Comment Perl implémente l'orienté objet

- Introduction à la technologie objet dans Perl
- Méthodes, classes et constructeurs
- Obtention et suivi de modules tiers du CPAN

##### Accès aux modules orientés objet

- Comment utiliser **use**
- Définition d'un schéma pour employer des modules objet
- Appel de méthodes avec la syntaxe **->**
- Passage de paramètres d'initialisation