Optimisation des applications .NET en langage C#

Cours Pratique de 4 jours - 28h
Réf : OPA - Prix 2024 : 2 390CHF HT

Ce cours vous propose une méthodologie pour améliorer l'efficacité de vos applications .NET et .Net Core. Il vous apprendra à utiliser les principaux outils d'analyse de performances et de diagnostic et vous permettra de maîtriser les différentes techniques d'optimisation en matière de codage du langage C#.

OBJECTIFS PÉDAGOGIQUES

À l'issue de la formation l'apprenant sera en mesure de :

Maîtriser l'architecture des applications .Net

Diagnostiquer les allocations mémoire et éviter les fuites mémoire

Optimiser le code C# en référence aux dernières versions C# 7, 8 et 9

Maîtriser les opérations sur les tableaux et les collections

Optimiser et mesurer les gains de performances via le parallélisme

Coder en C# pour résoudre des problématiques réelles

LE PROGRAMME

dernière mise à jour : 07/2021

1) Introduction

- Évolution de la plateforme .Net et de son écosystème.
- Optimiser : quoi, comment, quand ? Objectifs.

Travaux pratiques : Analyse de l'exécution d'une application .Net via l'outils WinDebug.

2) Le ramasse-miettes GC, optimisation des allocations et diagnostic

- La gestion de la mémoire par le ramasse-miettes et API GC.
- Outils de diagnostic de Visual Studio, de JetBrains et BenchmarkDotNet.
- Le Pattern Dispose et l'implémentation de IDisposable.
- Recommandations pour éviter les fuites mémoires potentielles d'une application .Net.
- Garde-fous au design time via les Règles d'Analyse de code.
- Outils de l'écosystème .Net pour tracer les erreurs de boxing, de closure et références nulles.

Travaux pratiques : Détection et correction des fuites de mémoire. Règles d'analyse et outils de diagnostic.

3) Méthodes C# et constructions du langage

- Méthodes C# et l'évolution du codage.
- Méthodes d'extension et architecture LINQ.
- Méthodes asynchrones et utilisation des mots clés async/await.
- Mesures de performance des allocations des types Value comparés aux types Reference.
- Nouveau type valeur ValueTuple de C# 7.0.
- Passage de paramètres et retour par référence de C# 7.2.
- Avantages de la programmation fonctionnelle en C# et comparaison au langage fonctionnel F#.

PARTICIPANTS

Développeurs, ingénieurs, architectes, chefs de projet.

PRÉREQUIS

Bonnes connaissances du langage C#. Expérience requise.

COMPÉTENCES DU FORMATEUR

Les experts qui animent la formation sont des spécialistes des matières abordées. Ils ont été validés par nos équipes pédagogiques tant sur le plan des connaissances métiers que sur celui de la pédagogie, et ce pour chaque cours qu'ils enseignent. Ils ont au minimum cinq à dix années d'expérience dans leur domaine et occupent ou ont occupé des postes à responsabilité en entreprise.

MODALITÉS D'ÉVALUATION

Le formateur évalue la progression pédagogique du participant tout au long de la formation au moyen de QCM, mises en situation, travaux pratiques…
Le participant complète également un test de positionnement en amont et en aval pour valider les compétences acquises.

MOYENS PÉDAGOGIQUES ET TECHNIQUES

• Les moyens pédagogiques et les méthodes d'enseignement utilisés sont principalement : aides audiovisuelles, documentation et support de cours, exercices pratiques d'application et corrigés des exercices pour les stages pratiques, études de cas ou présentation de cas réels pour les séminaires de formation.
• À l'issue de chaque stage ou séminaire, ORSYS fournit aux participants un questionnaire d'évaluation du cours qui est ensuite analysé par nos équipes pédagogiques.
• Une feuille d'émargement par demi-journée de présence est fournie en fin de formation ainsi qu'une attestation de fin de formation si le stagiaire a bien assisté à la totalité de la session.

MODALITÉS ET DÉLAIS D'ACCÈS

L'inscription doit être finalisée 24 heures avant le début de la formation.

ACCESSIBILITÉ AUX PERSONNES HANDICAPÉES

Vous avez un besoin spécifique d'accessibilité ? Contactez Mme FOSSE, référente handicap, à l'adresse suivante psh-accueil@orsys.fr pour étudier au mieux votre demande et sa faisabilité.

- Pattern Matching en C#.

Travaux pratiques : Exercices et démonstrations concrètes sur les éléments de codage et la programmation fonctionnelle.

4) Tableaux et collections

- Performance des opérations sur les données.
- Collections du .Net et caractéristiques.
- Implémentation de l'interface IEquatable.
- Pointeur IntPtr et le mot clé C# 7.2 stackalloc pour allouer sur la pile.

Travaux pratiques : Exercices et démonstrations sur les collections et allocations sur la pile.

5) La librairie TPL du .Net 4.7 : asynchronisme et parallélisme

- Évolution de la gestion des appels asynchrones via les mots clés async/await.
- Les nouvelles classes de System.Threading.Tasks.
- Paralléliser les itérations for et foreach. Design pattern en matière de parallélisme.
- Utilisation de l'infrastructure PLinq.
- Diagnostic de performance via le NuGet BenchmarkDotNet et DotTrace de JetBrains.

Travaux pratiques : Exercices et démonstrations sur le parallélisme. Diagnostics via BenchmarkDotNet.

6) Conclusion

- Incidence des fonctionnalités C# 8 et C# 9 sur le code et les applications .Net.
- Une synthèse et quelques conseils.
- Recommandations d'ouvrages et références internet.

LES DATES

CLASSE À DISTANCE
2024 : 16 juil., 05 nov.