

Lab 021 : Construire un serveur MCP en C#

Niveau : L200 Parcours : MCP Durée : ~45 min ? Coût : Gratuit (local + Ollama)

Ce que vous apprendrez

Créer un serveur MCP avec le SDK .NET officiel ModelContextProtocol Exposer des outils, des ressources et des prompts depuis C# Tester le serveur avec MCP Inspector Le connecter au mode Agent de GitHub Copilot via mcp.json

Introduction

Python est idéal pour le prototypage rapide MCP, mais .NET est courant dans les environnements d'entreprise. Le package NuGet officiel ModelContextProtocol rend la construction de serveurs MCP en C# nativement supportée.

Prérequis

.NET 8 SDK ou version ultérieure ? gratuit Lab 012 : Qu'est-ce que MCP ? recommandé Node.js (pour MCP Inspector) ? gratuit

? Fichiers d'accompagnement

note "Téléchargez ces fichiers avant de commencer le lab" Enregistrez tous les fichiers dans un dossier lab-021/ dans votre répertoire de travail.

Fichier | Description | Téléchargement | BrokenMcpServer.cs | Exercice de correction de bugs (3 bugs + auto-tests) | ? Télécharger |

Exercice du lab

Étape 1 : Créer le projet

[diagram or code sample]

Étape 2 : Construire le serveur MCP

Remplacez Program.cs par :

[diagram or code sample]

Créez ProductTools.cs :

[diagram or code sample]

Étape 3 : Exécuter et tester avec MCP Inspector

Terminal 1 ? compiler le serveur : [diagram or code sample]

Tester avec MCP Inspector : [diagram or code sample]

Dans Inspector, cliquez sur Tools et testez search_products avec la requête "camping". Vous devriez voir la tente renvoyée.

Étape 4 : Ajouter une ressource

Les ressources exposent des données en lecture seule (fichiers, vues de base de données, etc.). Ajoutez dans ProductTools.cs :

[diagram or code sample]

Mettez à jour Program.cs pour enregistrer les ressources : [diagram or code sample]

Étape 5 : Connecter à GitHub Copilot

Ajoutez dans .vscode/mcp.json de votre espace de travail :

[diagram or code sample]

Activez le mode Agent dans VS Code, puis demandez : *« Quels produits de camping sont en stock ? »*

Différences clés vs SDK Python

| Python | C# | Décorateur | @mcp.tool() | [McpServerTool] | Description | docstring | [Description("...")] | Ressources | @mcp.resource() | [McpServerResource(...)] | Transport | mcp.run(transport="stdio") | .WithStdioServerTransport() | Conteneur DI | ? |

Microsoft.Extensions.Hosting |

? Exercice de correction de bugs : Réparer le serveur MCP cassé

Ce lab inclut un fichier de serveur MCP C# volontairement cassé. Votre défi : trouver et

corriger 3 bugs.

[diagram or code sample]

Configuration : [diagram or code sample]

Les 3 bugs :

| Outil | Symptôme | Type | 1 | list_categories | NullReferenceException au démarrage |
Initialisation null | 2 | search_products | Retourne toujours une liste vide [] | Inversion
logique (!) | 3 | get_product_details | Retourne "not found" pour les IDs en minuscules |
Comparaison sensible à la casse |

Vérifiez vos corrections : Après avoir corrigé les 3 bugs, connectez-vous avec MCP

Inspector et exécutez :

list_categories() ? devrait retourner ["Backpacks", "Sleeping Bags", "Tents"]

search_products(keyword: "tent") ? devrait retourner P001, P002, P003

get_product_details(productId: "p001") ? devrait retourner les détails de TrailBlazer Tent
2P

? Quiz de connaissances

question "Q1 (Exécutez le lab) : Après avoir corrigé les 3 bugs et appelé

list_categories(), que retourne l'outil ? Listez les catégories dans l'ordre où elles
apparaissent dans la sortie."

Corrigez les bugs, démarrez le serveur, connectez-vous avec MCP Inspector et appelez
list_categories().

??? success "? Révéler la réponse" ["Backpacks", "Sleeping Bags", "Tents"]

Les catégories sont retournées par ordre alphabétique car le code original utilise une List
triée. Le bug #1 (categories = null) causait une NullReferenceException avant de retourner
quoi que ce soit ? le corriger révèle la liste correctement triée.

question "Q2 (Exécutez le lab) : Après avoir corrigé le bug #3 (le bug de comparaison
sensible à la casse), quelle valeur de StringComparison remplace StringComparison.Ordinal
dans la correction ?"

Lisez attentivement la description du bug #3, puis regardez la correction que vous avez
appliquée dans ? BrokenMcpServer.cs.

??? success "? Révéler la réponse" StringComparison.OrdinalIgnoreCase

Le code original utilisait StringComparison.Ordinal qui est sensible à la casse, donc
get_product_details("p001") échouait car les IDs stockés sont en majuscules ("P001"). Le
remplacer par StringComparison.OrdinalIgnoreCase fait fonctionner les recherches d'ID quelle que soit la
casse envoyée par le client.

question "Q3 (Choix multiple) : Le bug #2 dans search_products causait un retour de liste
toujours vide. Quelle était la cause racine ?"

- A) Le paramètre keyword était null - B) L'appel Contains() était inversé avec ! ? il
filtrait les correspondances au lieu de les garder - C) La liste de produits n'était pas
initialisée - D) La recherche était sensible à la casse et aucun produit ne correspondait

??? success "? Révéler la réponse" Correct : B ? Inversion logique

Le code avait !product.Name.Contains(keyword) ? le ! inversait la condition, donc les
produits qui contenaient le mot-clé étaient exclus, et les produits qui ne contenaient pas
le mot-clé étaient retournés. Avec une liste de résultats vide, il n'y avait aucun produit
non-correspondant non plus. Supprimer le ! corrige la logique.

Prochaines étapes

Déployer ce serveur dans le cloud : ? Lab 028 ? Déployer MCP sur Azure Container Apps

Version Python du serveur MCP : ? Lab 020 ? Serveur MCP en Python

Modernisation de Phase 3

warning "Verified 2026-10-27" Ciblez MCP 2026-07-28. Préférez serveur stateless, discovery, SDK Tier 1 actuel et état multi round-trip explicite. Ajoutez Registry et autorisation selon le modèle de confiance.

[diagram or code sample]

Model Context Protocol specification MCP authorization MCP Registry

Kit d'accompagnement

Le build du site génère des téléchargements PDF et ZIP stables. Le ZIP contient le guide du lab et tous les fichiers d'accompagnement déclarés.

Télécharger le PDF généré Télécharger le ZIP généré