

warning "Mise à jour du backend (octobre 2026)" L'inférence GitHub Models a été retirée le 30 juillet 2026. Les références ci-dessous sont conservées uniquement comme contexte historique pendant la migration de ce lab. Utilisez le Lab 013 ? Backends LLM gratuits pour configurer Foundry Local ou Ollama avant de continuer.

Lab 003 : Choisir le Bon Outil

Niveau : L50 Parcours : Tous les parcours Durée : ~15 min ? Coût : Gratuit ? Aucun compte requis

Ce Que Vous Allez Apprendre

Un cadre de décision pratique pour choisir votre outil d'agent IA Comprendre les compromis clés (contrôle vs. simplicité, coût vs. puissance) Des parcours d'apprentissage suggérés en fonction de votre rôle et de vos objectifs

Introduction

Après avoir examiné le panorama dans le Lab 002, la question naturelle est : par où commencer ?

Utilisez le diagramme de décision et les guides par profil ci-dessous pour trouver votre chemin.

Diagramme de Décision

Diagramme de Décision

question "? Vérifiez Votre Compréhension" Selon le diagramme de décision, quel outil devriez-vous utiliser si votre objectif principal est de connecter une base de données ou une API existante à des agents IA ?

??? success "Réponse" Vous devriez construire un MCP Server. Le MCP (Model Context Protocol) fournit un standard de connecteur universel afin que tout agent IA compatible MCP puisse accéder à votre outil ou source de données via une interface commune.

Par Profil

? Analyste Métier / Utilisateur Avancé Objectif : Automatiser les flux de travail, créer des agents sans écrire de code.

Parcours recommandé : 1. Lab 001 ? Lab 011 ? Lab 069 ? Lab 075

Outils : Copilot Studio, Declarative Agents, Power BI Copilot, M365 Copilot

??? Développeur (Python / C#) Objectif : Écrire des agents en code, intégrer avec des systèmes existants.

Parcours recommandé : 1. Lab 013 ? Lab 076 ? Lab 020 ? Lab 082 ? Lab 084

Outils : Agent Framework (SK), MCP, Guardrails, GitHub Models

? Ingénieur Intégration / Plateforme Objectif : Exposer les systèmes existants (bases de données, API) aux agents IA.

Parcours recommandé : 1. Lab 012 ? Lab 020 ? Lab 031 ? Lab 054 ? Lab 064

Outils : MCP, A2A Protocol, pgvector, Azure API Management

?? Architecte de Solutions Objectif : Concevoir des systèmes multi-agents en production avec gouvernance et observabilité.

Parcours recommandé : 1. Lab 076 ? Lab 049 ? Lab 050 ? Lab 074 ? Lab 084

Outils : Agent Framework, Foundry Agent Service, OpenTelemetry, A2A + MCP

? Ingénieur / Analyste de Données Objectif : Construire des analyses alimentées par l'IA, des agents de données et des pipelines d'enrichissement.

Parcours recommandé : 1. Lab 047 ? Lab 052 ? Lab 053 ? Lab 067 ? Lab 075

Outils : Fabric IQ, Work IQ, GraphRAG, Power BI Copilot

? Administrateur d'Entreprise / Gouvernance IT Objectif : Gouverner, sécuriser et surveiller les déploiements d'agents IA à travers l'organisation.

Parcours recommandé : 1. Lab 063 ? Lab 065 ? Lab 066 ? Lab 064 ? Lab 046

Outils : Entra ID, Purview DSPM, Copilot Studio Governance, APIM, Agent 365

? Étudiant / Apprenant Objectif : Comprendre les agents IA et construire quelque chose de réel, gratuitement.

Parcours recommandé : 1. Lab 001 ? Lab 004 ? Lab 013 ? Lab 078 ? Lab 076 ? Lab 022

Outils : GitHub Models, Foundry Local, Agent Framework ? tout est gratuit !

question "? Vérifiez Votre Compréhension" Un architecte de solutions doit concevoir un système multi-agents en production avec observabilité et gouvernance. Quelle combinaison d'outils ce lab recommande-t-il ?

??? success "Réponse" Foundry, Semantic Kernel, AutoGen et App Insights. Le parcours d'apprentissage recommandé est : Foundry Agent MCP ? Agent Observability ? Multi-Agent SK ? AutoGen Multi-Agent. Cela couvre le runtime managé, la logique des agents, l'orchestration multi-agents et la surveillance.

question "? Vérifiez Votre Compréhension" Que signifie « plus de contrôle = plus de responsabilité » dans le compromis contrôle vs. simplicité ?

??? success "Réponse" Les outils pro-code comme AutoGen et Semantic Kernel offrent une flexibilité totale sur la logique de l'agent, mais vous devez gérer davantage de choses ? gestion des erreurs, déploiement, sécurité, mise à l'échelle. Les outils no-code comme Copilot Studio sont plus rapides à construire mais moins personnalisables. Le bon choix dépend des compétences et des exigences de votre équipe.

Les Deux Compromis Clés

Contrôle vs Simplicité, Gratuit vs Payant

Plus de contrôle = plus de flexibilité + plus de responsabilité. Plus de simplicité = plus rapide à construire + moins personnalisable.

question "? Vérifiez Votre Compréhension" Un étudiant sans abonnement Azure et sans budget peut-il quand même construire un agent IA fonctionnel en utilisant les outils de ce hub ?

??? success "Réponse" Oui ! GitHub Models et Semantic Kernel sont entièrement gratuits. Les labs conceptuels L50 et les labs L100?L200 utilisant GitHub Models ne nécessitent aucun abonnement Azure. Les étudiants peuvent construire de vrais agents, exécuter des MCP servers localement et apprendre l'ensemble du cycle de développement d'agents à coût zéro.

2. Gratuit vs. Payant

Le SVG ci-dessus inclut la comparaison complète Gratuit vs. Payant. Commencez gratuitement ? ajoutez Azure uniquement lorsque vous avez besoin de fonctionnalités de production.

? Test de Connaissances

question "Q1 (Choix Multiple) : Un développeur souhaite construire une extension VS Code qui répond à @mybot dans GitHub Copilot Chat. Quel outil/API devrait-il utiliser ?"

- A) Copilot Studio - B) VS Code Chat Participant API (Lab 025) - C) Microsoft Foundry Agent Service - D) Azure Bot Service

??? success "? Révéler la Réponse" Correct : B ? VS Code Chat Participant API

La Chat Participant API enregistre un participant @yourextension directement dans l'interface Copilot Chat de VS Code. Elle fonctionne entièrement dans VS Code ? pas d'abonnement Azure, pas de serveur requis. Copilot Studio est destiné aux agents sans code dans Teams/M365. Foundry est destiné aux agents hébergés côté serveur avec une mise à l'échelle cloud complète.

question "Q2 (Choix Multiple) : Quel facteur est le PLUS important lors du choix entre Copilot Studio et Semantic Kernel ?"

- A) Le langage de programmation que vous préférez (Python vs C#) - B) Si vous avez besoin d'un déploiement cloud ou d'un déploiement local - C) Votre rôle et le niveau de contrôle du code dont vous avez besoin ? développeur citoyen vs. développeur professionnel - D) Le

fournisseur de LLM (OpenAI vs Anthropic)

??? success "? Révéler la Réponse" Correct : C

L'axe de décision principal est contrôle du code vs. rapidité. Copilot Studio cible les développeurs citoyens et les professionnels IT qui ont besoin d'un agent fonctionnel rapidement sans code. Semantic Kernel cible les développeurs professionnels qui ont besoin d'un contrôle total sur la logique, les schémas d'outils, les modèles de mémoire et le comportement en production. Les deux supportent plusieurs LLMs et le déploiement cloud. question "Q3 (Choix Multiple) : Le principe du « moindre privilège » stipule que votre agent ne doit avoir accès qu'à ce dont il a besoin ? rien de plus. Laquelle de ces options viole le moindre privilège ?"

- A) Un agent de recherche de produits qui peut appeler search_products() et get_product_details() - B) Un agent de service client avec un accès en lecture seule à la base de données - C) Un agent de suivi de commandes avec des identifiants d'administrateur complets pour la base de données des commandes - D) Un agent météo qui ne peut appeler que l'API météo publique

??? success "? Révéler la Réponse" Correct : C ? Des identifiants d'administrateur complets violent le moindre privilège

Un agent de suivi de commandes n'a besoin que de *lire* les enregistrements de commandes. Lui donner des identifiants d'administrateur signifie qu'une attaque par injection de prompt ou une erreur de logique pourrait supprimer des commandes, modifier des prix ou accéder à toutes les données clients. La configuration correcte est un utilisateur de base de données en lecture seule limité aux tables spécifiques dont l'agent a besoin. Les options A, B et D respectent correctement le moindre privilège.

Résumé

Il n'y a pas d'outil unique « parfait » ? cela dépend de votre rôle, de vos objectifs et de vos contraintes. La bonne nouvelle : tout dans ce hub commence gratuitement, et vous pouvez toujours monter en niveau. Le cadre de décision ci-dessus vous oriente vers le parcours le plus efficace.

Prochaines Étapes

Choisissez votre parcours et lancez-vous !

Parcours sans code : ? Lab 011 ? Copilot Studio Parcours développeur (gratuit) : ? Lab 013 ? GitHub Models Parcours MCP : ? Lab 012 ? What is MCP? Parcours Azure complet : ? Lab 030 ? Foundry + MCP Vous voulez d'abord comprendre les LLMs ? ? Lab 004 ? How LLMs Work Modernisation de Phase 3

warning "Verified 2026-10-27" Utilisez Microsoft Agent Framework 1.0 pour les nouveaux agents. Semantic Kernel reste une source de migration héritée et AutoGen vise la maintenance. Remplacez les planners obsolètes par workflows, outils, mémoire, checkpoints et arrêts explicites.

[diagram or code sample]

Microsoft Agent Framework 1.0 GA AutoGen to Agent Framework migration

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

Supporting file: CHEATSHEET.md

Fichiers d'accompagnement: Lab 003 : Choisir le Bon Outil

Commencez par le parcours local ou gratuit documenté. Consignez les hypothèses et les

preuves de validation. Gardez les services hébergés facultatifs sauf exigence explicite du lab.

Supporting file: README.md

Lab 003 : Choisir le Bon Outil

Utilisez ces fichiers avec le guide du lab et le kit généré.