AI Tools für Entwickler

Vienna, September 26th, 2024

Overview AI Code Assistants

Beispiele für Code Assistants

- Github Copilot
- Gitlab Duo
- Jetbrains AI Pro
- Tabnine
- Google Gemini Code Assist
- Amazon Q Developer
- Continue / LM Studio
- ...

Overview AI Code Assistants

Assistant	Company	Model	location	price / user / month	Visual Studio (Code)	IntelliJ	Eclipse
Github Copilot	Github Inc.	Open AI Codex	cloud & local	$10 - $19	yes	yes	3rd party
Gitlab Duo	Gitlab Inc.	Google Vertex AI, Anthropic Claude	cloud	$19 - $39	yes	yes	no
Jetbrains AI Pro / Jetbrains AI Assistant	Jetbrains	Jetbrains Proprietary, Google Vertex AI	cloud	$10	no	yes	no
Google Gemini Code Assist	Google	Google Vertex AI	cloud	$19-$23	yes	yes	no
Tabnine	Codota	proprietary	both	$12-$39	yes	yes	yes
Codeium	Exafunction	proprietary	local	$0 - $ 19			
LM Studio / Continue Plugin		multiple like: Mixtral-8x7B, CodeLlama-7B, OpenHermes-2.5-Mistral-7B	local	free	yes	yes	no

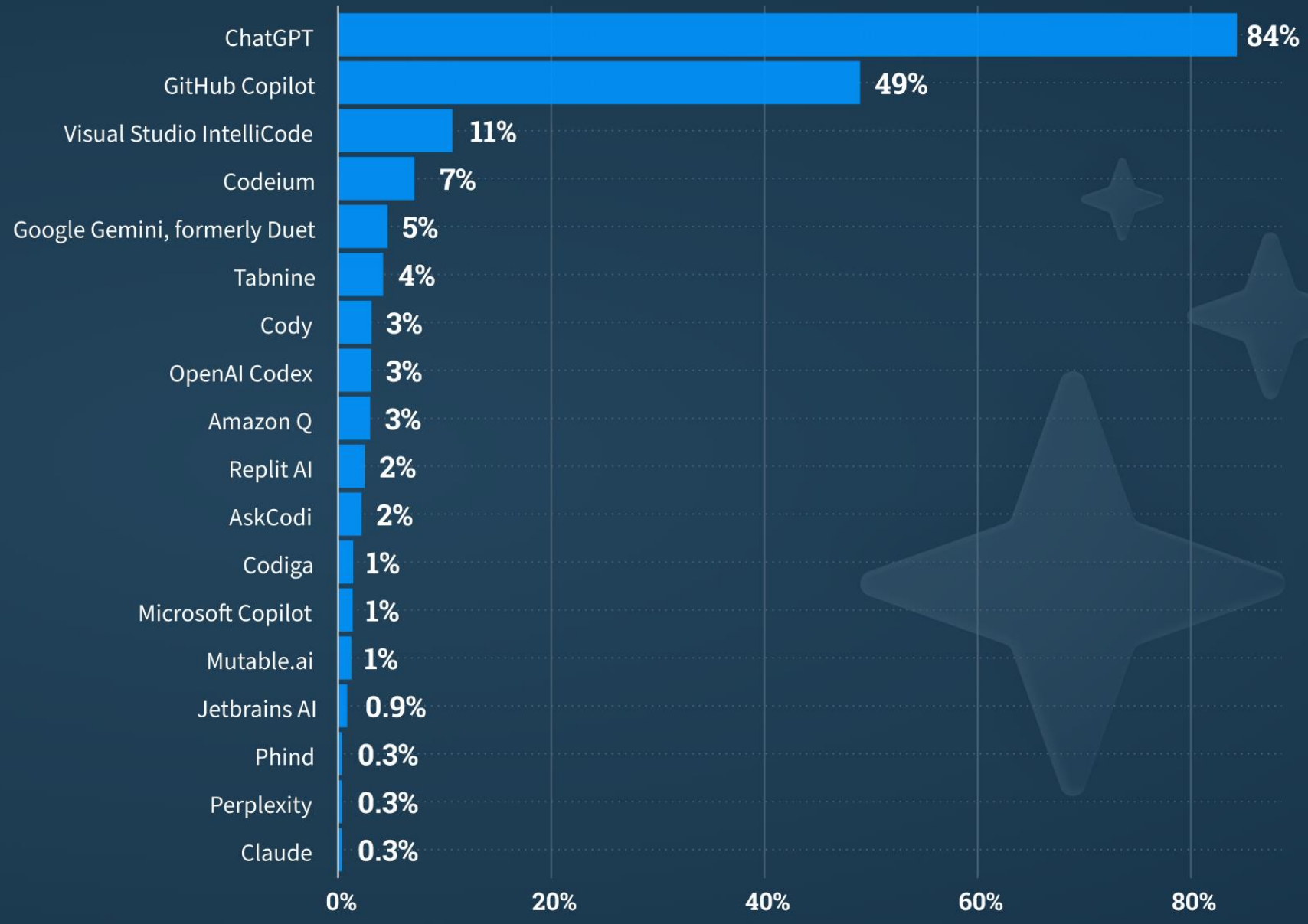
Einschränkungen

Einige Assistenten entwickeln ihr volles Feature Set nur mit Projekten in spezifischen Umgebungen. Beispiele:

- Google Gemini / Cloud Code ist für Google Cloud Projekte ausgelegt
- Gitlab Duo ist ausgelegt auf Projekte, die in Gitlab Enterprise Repositories verwaltet werden

Code Assistants

What is the primary code assistant tool professional developers use?

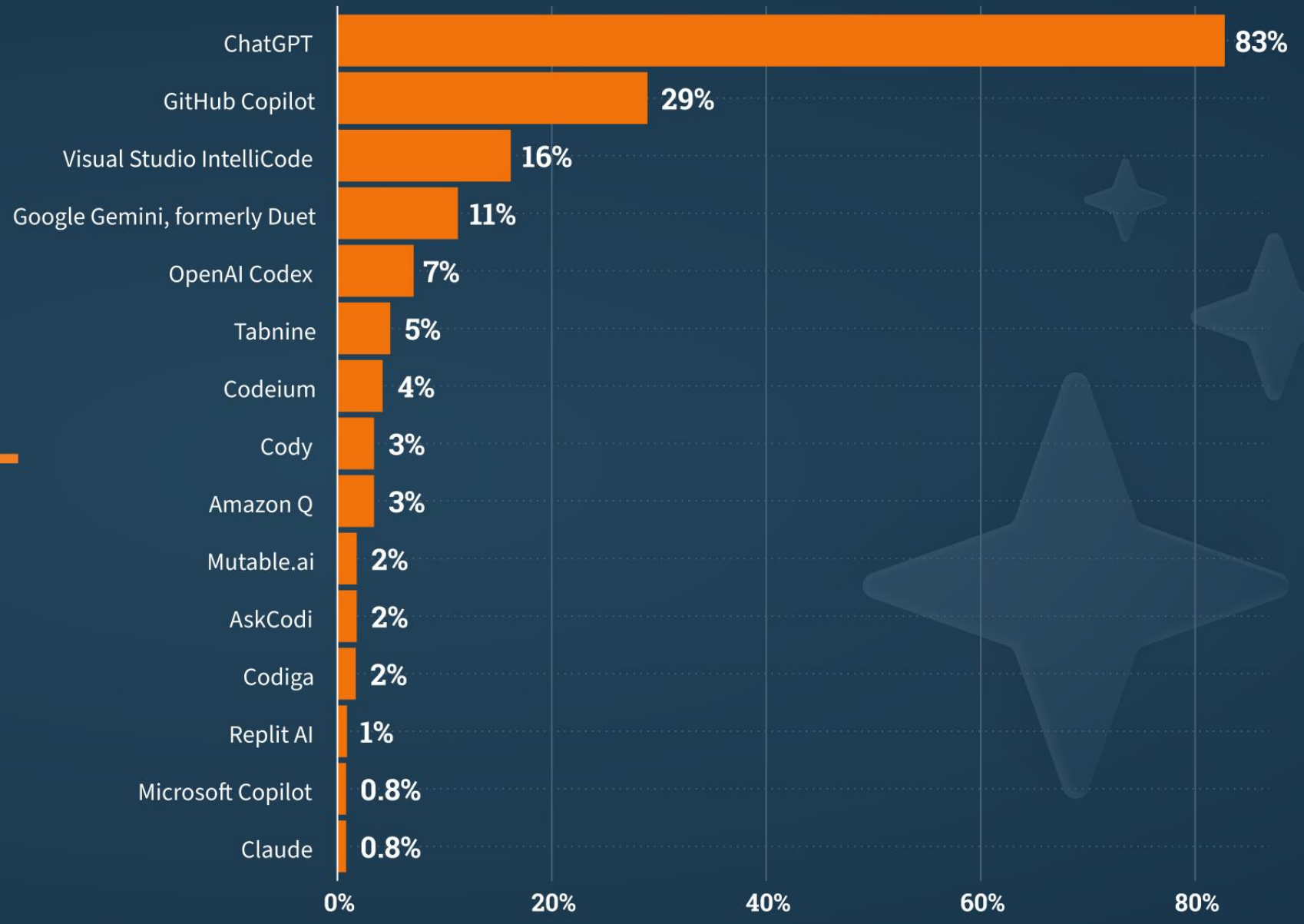


stackoverflow

Source: stackoverflow.com survey May 2024

Code Assistants

What is the primary code assistant tool developers-in-training use?



stackoverflow

Source: stackoverflow.com survey May 2024

AI Assistant Features

- Code Generierung basierend auf sprachlicher Beschreibung
- Code Generierung über Autocompletion anhand momentanen Kontexts
- Generierung von Unit Tests
- Code Migration in andere Sprache
- Analyse von Laufzeitfehlern oder Compile Fehlern
- Analyse von Schwachstellen im Code
- Refactoring von Code
- Generierung von (Javadoc) Kommentaren
- Generierung von Commit Messages
- Generierung von Command Line Befehlen

Assitenten im Detail

- Jetbrains AI Assistant
- Github Copilot
- LM Studio / Continue Plugin

Was wir nicht behandeln:

Code Generatoren, die gesamte Anwendungen generieren, wie z.B.

- make.com (Prozess- und Integrationsautomation)
- bubble.io (Generierung von Webapps)
- Flutterflow.io (Mobile Apps Generator)

DEMO

Beobachtungen

Probleme bei der Verwendung:

- Generierter Code ist sehr oft fachlich inkorrekt -> man übernimmt Fehler
- Generierter Code enthält oft Vorschläge mit veralteten Bibliotheken
- Generierte Dokumentation enthält oft nur Dinge, die aus dem Code ersichtlich sind. Das resultiert auf Dauer zu einer Verarmung der Dokumentation
- Nach Refactoring hat sich die Funktionalität des Codes geändert
- Lokale Modelle haben größte Ungenauigkeiten

Ultimatives Feature:

- Generierung von Commit Messages
- Generierung von Unit Tests

Beobachtungen

Unterschiede der Code Qualität der einzelnen Assistenten

- Ergebnisse ähneln einander bei ChatGPT, Copilot und Jetbrains AI Assist sehr stark
- Google Gemini generiert schlechtesten Code

Ultimatives Feature:

- Generierung von Commit Messages
- Generierung von Unit Tests

Beste Integration in IDE

- Jetbrains AI Assist

Meine Sieger

Produkt	Integration IntelliJ	Integration VS Code	Ergebnisse
Jetbrains AI Assistant	* * * * *	-	* * *
Github Copilot	* * *	* * * *	* * * *
LM Studio + Continue Plugin	* *	* *	*
ChatGPT	-	-	* * * * *

Beste Ergebnisse: ChatGPT

Beste Integration in IntelliJ: Jetbrains AI

Beste Integration in VS Code: Github Copilot

Github Copilot

Dokumentation:

- Allgemeine Dokumentation: https://docs.github.com/de/copilot
- Copilot in VS Code https://code.visualstudio.com/docs/copilot/overview
- Prompt Engineering for Github Copilot https://docs.github.com/en/copilot/using-github-copilot/prompt-engineering-for-github-copilot

Jetbrains Code Assist

Dokumentation:

- Allgemeine Dokumentation: https://www.jetbrains.com/help/idea/ai-assistant.html

THANK YOU!