

# GH-300

# GitHub Copilot

## part 1

本講義は以下のMicrosoft Learn教材に準拠しています。

[Course GH-300T00-A: GitHub Copilot - Training | Microsoft Learn](#)



# Part 1 目次

- モジュール1 責任あるAI
- モジュール2 GitHub Copilotの概要
- モジュール3 プロンプトエンジニアリング
- モジュール4 高度な機能
- モジュール5 さまざまな環境でのGitHub Copilot
- モジュール6 管理とカスタマイズ

# 責任あるAIの原則

- ・「**責任あるAIの原則**」は、マイクロソフトが定義した、AIシステムの利用者が安心してAIを使えるようにするための具体的なガイドライン
- ・6つの原則で構成される

# 「責任あるAIの原則」に含まれる6つの原則

原則	概要
<b>公平性</b> fairness	AIシステムはすべての人を公平に扱い、同様の状況にあるグループに対して異なる影響を与えないようにするべきである
<b>信頼性と安全性</b> reliability and safety	AIシステムは信頼性が高く、安全に運用されるべきであり、設計通りに動作し、予期しない状況にも安全に対応できるようにするべきである
<b>透明性</b> transparency	AIシステムの決定がどのように行われたかを利用者が理解できるようにするべきである
<b>プライバシーとセキュリティ</b> privacy and security	個人情報や企業情報を保護し、データの収集、使用、保存について透明性を持たせ、消費者がデータの使用方法を選択できるようにするべきである
<b>包括性</b> inclusiveness	AIシステムはすべての人々を含め、排除することなく設計されるべきである
<b>責任</b> accountability	AIシステムの設計者や運用者はシステムの動作に対して責任を持ち、業界標準に基づいた責任の規範を確立するべきである

# GitHub Copilotとの関係

- ・マイクロソフトは2018年にGitHubを買収
- ・GitHub やGitHub Copilotはマイクロソフトの製品の一部となつた
- ・GitHub Copilotも 「**責任あるAIの原則**」に沿って開発されている
  - ・したがつて利用者が安心してGitHub Copilotを利用できると言える

# 参考: rinna株式会社の「責任あるAIの原則」

- 責任あるAI | rinna株式会社
- マイクロソフトの6つの「原則」に「Creativity（創造性）」を加えた「7つの原則」を採用
- 各原則がどのようにrinnaのAIサービスの開発・運用・販売に反映されているのかが、具体的にわかりやすく解説されている
- ※rinna株式会社は、これまでマイクロソフトが開発・運営してきたAI「りんな」を含むチャットボットAI事業を引き継いで2020/6/17に設立された会社。



# モジュール1 まとめ

- ・「**責任あるAIの原則**」は、マイクロソフトが定義した、AIシステムの利用者が安心してAIを使えるようにするためのガイドラインで、6つの原則から構成される
- ・GitHub Copilotもこの原則に沿って開発されているため、安心して利用できる
- ・(応用) 「責任あるAIの原則」は、自社のAIガイドラインを策定する際のベースラインとしても活用できる
  - ・例: rinna株式会社

# Part 1 目次

- ・モジュール1 責任あるAI
- ・モジュール2 GitHub Copilotの概要
- ・モジュール3 プロンプトエンジニアリング
- ・モジュール4 高度な機能
- ・モジュール5 さまざまな環境でのGitHub Copilot
- ・モジュール6 管理とカスタマイズ

# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

いろいろとご質問がお有りかと思いますが、ここで、過去のご受講者様からいただいた多数のご質問とその回答をご紹介します。ここを聞いていただければ、ほとんどの疑問は解消するかと思います！

# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ



# GitHub (ギットハブ) とは？

- ・ソフトウェア開発のプラットフォーム
- ・インターネット上での多数の開発者とのコラボレーションが可能
- ・近年はエンタープライズ向け機能も強化
- ・Git（バージョン管理ツール）を使用してソースコードのバージョン管理を行う
- ・2018/6/4, マイクロソフトはGitHubの買収を発表、同10月に買収を完了。現在GitHubはマイクロソフト傘下となっている。

[マイクロソフト、GitHubの買収を完了 - ZDNET Japan](#)

<https://japan.cnet.com/article/35120250/>

## ■ GitHubのトップページ <https://github.com/>

The screenshot shows the GitHub homepage with a dark blue background. At the top, there's a banner for GitHub Copilot. Below it is the GitHub logo and a 'Sign in' button. The main headline reads 'Build and ship software on a single, collaborative platform'. A subtext below it says 'Join the world's most widely adopted AI-powered developer platform.' There are three buttons at the bottom: a white 'Enter your email' input field, a green 'Sign up for GitHub' button, and a dark grey 'Try GitHub Copilot' button. A yellow callout box at the bottom left contains Japanese text: 'GitHubの利用をこれから開始する場合、ここからアカウントを作成できる'. A small circular arrow icon with an upward arrow is in the bottom right corner.

GitHub Copilot is now available for free. [Learn more](#)

Sign in

# Build and ship software on a single, collaborative platform

Join the world's most widely adopted AI-powered developer platform.

Enter your email

Sign up for GitHub

Try GitHub Copilot

GitHubの利用をこれから開始する場合、  
ここからアカウントを作成できる

# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# GitHub Copilot とは？

- ・プログラマのコードの記述作業を支援する機能
- ・2021/6/29 プレビュー開始、2022/6/22 正式リリース
- ・「AIペアプログラマ」である
  - ・人間のプログラマのコード記述をAIが助けてくれる
  - ・プログラマが効率的にコードを記述するのに役立つ
- ・Visual Studio Codeなど、さまざまな開発ツールと連携が可能
- ・個人で利用するにはGitHubアカウントが必要
  - ・法人（組織）で利用する場合は組織から払い出されるアカウントを使用

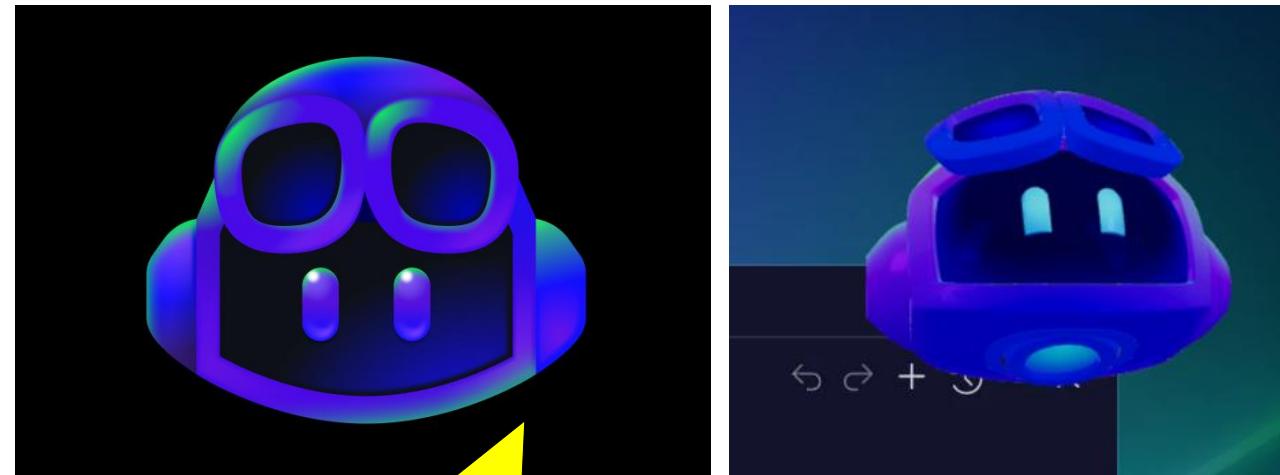
<https://github.blog/jp/2022-06-22-github-copilot-is-generally-available-to-all-developers/>

<https://www.itmedia.co.jp/news/articles/2106/30/news063.html>

# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# GitHubのCopilotのマスコットキャラ 「Copilot」

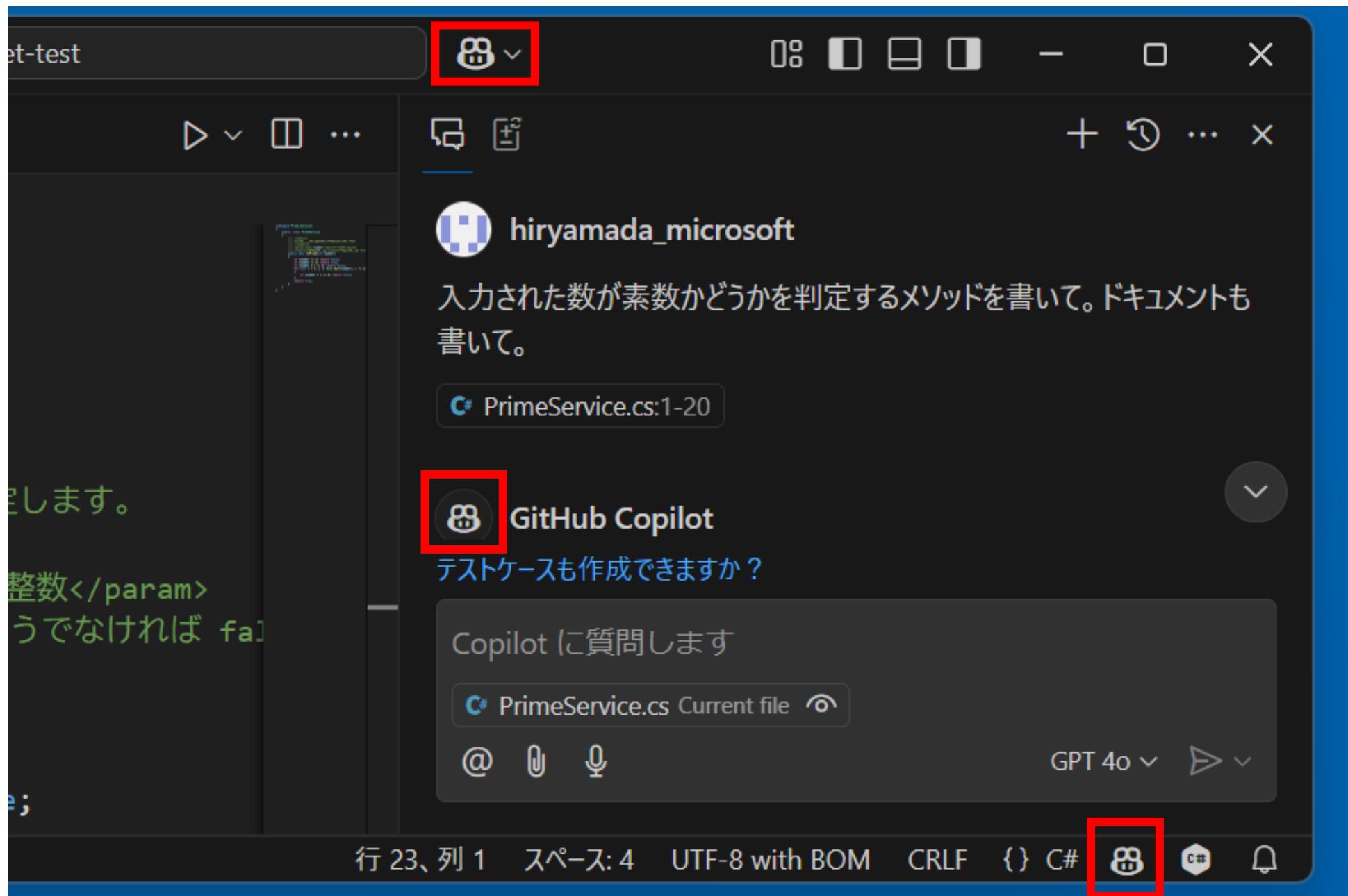


ヘルメットとゴーグルを装着した  
「コパイロット」（副操縦士）の  
イメージ



開発ツールなどに  
表示される白黒アイコン

## ■Visual Studio Codeの中に表示されるCopilotアイコンの例



# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料でも利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# GitHub Copilotは無料でも利用可能

- 2024/12/18～ **GitHub Copilot Free** が利用可能になった
- 個人のGitHubユーザーで、**組織や企業を通じてCopilotを利用していない方**が対象
- 利用制限がある
  - 1ヶ月に 2,000 コード補完が利用可能
  - 1ヶ月に 50 チャットメッセージが利用可能
- 制限なく利用したい場合は GitHub Copilot Pro（月額10ドル）などを契約

# 参考: GitHub Copilotのサブスクリプションプラン

- GitHub Copilot Free
  - 個人ユーザー向け。無料
- GitHub Copilot Pro
  - 個人のプロフェッショナル開発者向け。10ドル/月 or 100ドル/年
  - (以前は「GitHub Copilot **Individual**」と呼ばれていた)
- GitHub Copilot Pro+
  - すべてのAIモデルを利用したい開発者向け。39ドル/月 or 390ドル/年
- GitHub Copilot Business
  - チームや小規模な組織向け。19ドル/ユーザー/月
- GitHub Copilot Enterprise
  - 大規模な企業向け。39ドル/ユーザー/月

[GitHub Copilot のサブスクリプションプラン - GitHub Docs](#)

[GitHub Copilot Enterprise の概要 - Training | Microsoft Learn](#)

詳しくはこちらの  
ページを参照

# ■GitHub Copilot のサブスクリプションの選択画面（個人向け）

## Free

GitHub Copilot の使用を  
すばやく開始する方法。

\$0<sup>USD</sup>

[開始する](#)

[VS Code で開く](#)

^ 特典

- ✓ 1か月あたり 50 件のエージェント モードまたはチャットリクエスト
- ✓ 1か月あたり 2,000 件のコード補完
- ✓ Claude 3.5 Sonnet、GPT-4.1 などへのアクセス

## Pro ベストセラー

無制限のコード補完とチャットを利用して、より多くのモデルにアクセス。

\$10<sup>USD</sup>  
月額または年額 \$100

[30 日間無料で試す](#)

^ Free プランの全特典プラス:

- ✓ GPT-4.1 を使用した無制限のエージェント モードとチャット<sup>1</sup>
- ✓ 無制限のコード補完
- ✓ コード レビュー、Claude 3.7/4 Sonnet、Gemini 2.5 Pro などへのアクセス
- ✓ 最新モデルを使用するためのプレミアム リクエストの件数が GitHub Copilot Free の 6 倍。さらに多くのプレミアム リクエストを購入するオプション付きです<sup>2</sup>
- ✓ コーディング エージェント (プレビュー)

## Pro+

最大レベルの柔軟性とモデル選択肢。

\$39<sup>USD</sup>  
月額または年額 \$390

[開始する](#)

^ Pro プランの全特典プラス:

- ✓ Claude Opus 4、o3 といったすべてのモデルへのアクセス
- ✓ 最新モデルを使用するためのプレミアム リクエストの件数が GitHub Copilot Free の 30 倍。さらに多くのプレミアム リクエストを購入するオプション付きです<sup>2</sup>
- ◆ Access to [GitHub Spark](#)

# ■GitHub Copilot のサブスクリプションの選択画面（企業向け）

## Business

GitHub Copilot でワークフローを高速化。

\$19<sup>USD</sup>  
ユーザー/月

[開始する](#)

[営業担当者へのお問い合わせ](#)

---

△ 特典

- ✓ GPT-4.1 を使用した無制限のエージェント モードとチャット<sup>1</sup>
- ✓ 無制限のコード補完
- ✓ コード レビュー、Claude 3.5/3.7/4 Sonnet、Gemini 2.5 Pro などへのアクセス
- ✓ 最新モデルを使用するためのプレミアム リクエスト  
ト数はユーザーあたり 300 件。さらに多くのプレ  
ミアム リクエストを購入するオプション付きです<sup>2</sup>
- ✓ ユーザー管理と使用状況メトリクス
- ✓ 知財免責とデータ プライバシー
- ✓ コーディング エージェント (プレビュー)

## Enterprise

AI エージェントと包括的なモデルアクセスでスケーリング。

\$39<sup>USD</sup>  
ユーザー/月

[開始する](#)

[営業担当者へのお問い合わせ](#)

---

△ Business プランの全特典プラス:

- ✓ Claude Opus 4、o3 といったすべてのモデルへのアクセス
- ✓ 最新モデルを使用するためのプレミアム リクエストの  
件数が Business プランの 3.33 倍。さらに多くのプレ  
ミアム リクエストを購入するオプション付きです<sup>2</sup>

## ■GitHub Copilot の各サブスクリプションプランで提供される機能の違い (表の一部)

	Copilot Free	Copilot Pro	Copilot Pro+	Copilot Business	Copilot Enterprise
Available models in chat	Copilot Free	Copilot Pro	Copilot Pro+	Copilot Business	Copilot Enterprise
GPT-4.1	✓	✓	✓	✓	✓
GPT-5	✗	✓	✓	✓	✓
o3	✗	✗	✓	✗	✓
o4-mini	✗	✓	✓	✓	✓
Claude Opus 4.1	✗	✗	✓	✗	✓
Claude Opus 4	✗	✗	✓	✗	✓
Claude Sonnet 3.5	✓	✓	✓	✓	✓
Claude Sonnet 3.7	✗	✓	✓	✓	✓
Claude Sonnet 3.7 Thinking	✗	✓	✓	✓	✓
Claude Sonnet 4	✗	✓	✓	✓	✓
Gemini 2.5 Pro	✗	✓	✓	✓	✓
Gemini 2.0 Flash	✓	✓	✓	✓	✓

たとえば、GPT-5、Claude Sonnet 3.7、Gemini 2.5 Proなどの最新モデルは、GitHub Copilot Proプラン以上で利用可能

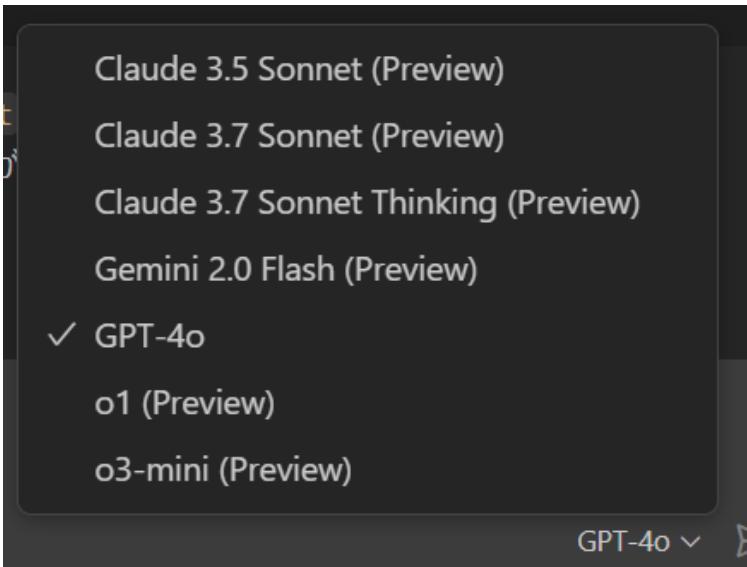
## ■GitHub Copilot の各サブスクリプションプランで提供される機能の違い (表の一部)

	Copilot Free	Copilot Pro	Copilot Pro+	Copilot Business	Copilot Enterprise
Available models in chat	Copilot Free	Copilot Pro	Copilot Pro+	Copilot Business	Copilot Enterprise
GPT-4.1	✓	✓	✓	✓	✓
GPT-5	✗	✓	✓	✓	✓
o3	✗	✗	✓	✗	✓
o4-mini	✗	✓	✓	✓	✓
Claude Opus 4.1	✗	✗	✓	✗	✓
Claude Opus 4	✗	✗	✓	✗	✓
Claude Sonnet 3.5	✓	✓	✓	✓	✓
Claude Sonnet 3.7	✗	✓	✓	✓	✓
Claude Sonnet 3.7 Thinking	✗	✓	✓	✓	✓
Claude Sonnet 4	✗	✓	✓	✓	✓
Gemini 2.5 Pro	✗	✓	✓	✓	✓
Gemini 2.0 Flash	✓	✓	✓	✓	✓

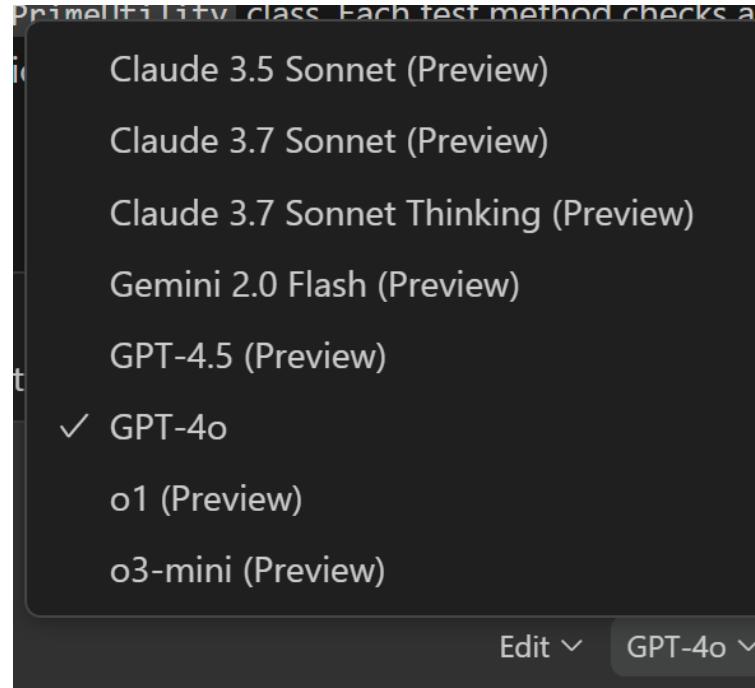
GitHub Copilot Freeでは  
GPT-4.1、Claude Sonnet  
3.5、Gemini 2.0 Flashのみ  
利用可能

## ■ご参考: 選択可能なモデル

2025/2/5時点



2025/3/1時点

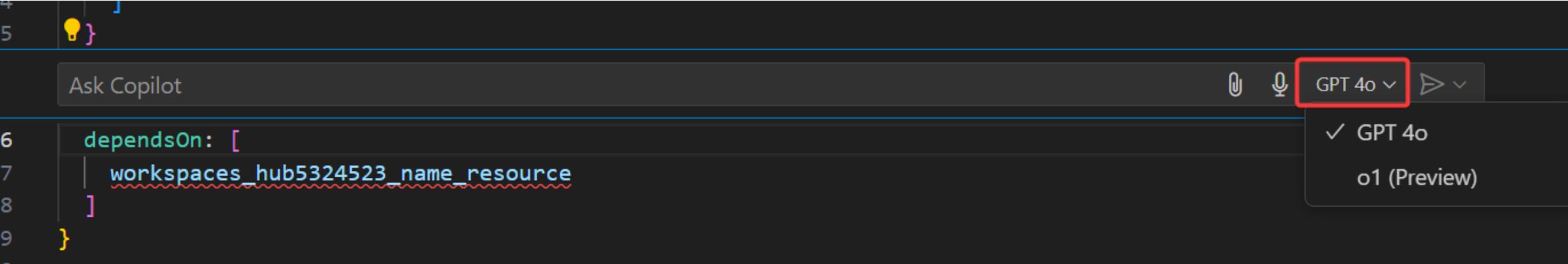


2025/8/14時点

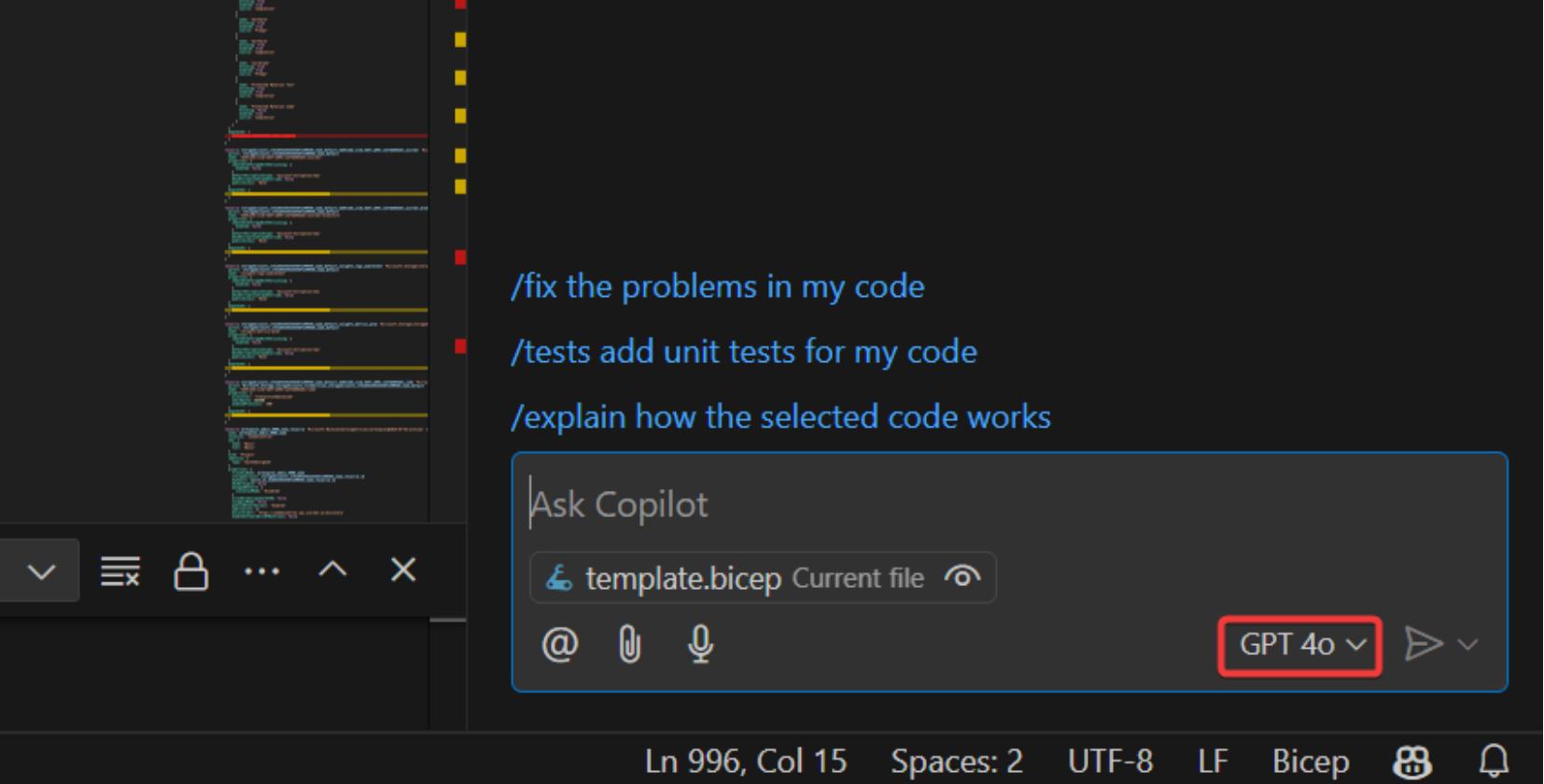
✓ GPT-4.1	0x
GPT-4o	0x
Claude Opus 4	10x
Claude Opus 4.1 (Preview)	10x
Claude Sonnet 3.5	1x
Claude Sonnet 3.7	1x
Claude Sonnet 3.7 Thinking	1.25x
Claude Sonnet 4	1x
Gemini 2.0 Flash	0.25x
Gemini 2.5 Pro (Preview)	1x
GPT-5 (Preview)	1x
o3 (Preview)	1x
o3-mini	0.33x
o4-mini (Preview)	0.33x
<a href="#">モデルの管理...</a>	
GPT-4.1 ▾	

※組織の設定により、プレビューのモデルの利用を禁止している場合があります

## ■ご参考: モデルの切り替えはいつでも可能



```
95 }  
96 dependsOn: [  
97 | workspaces_hub5324523_name_resource  
98 ]  
99 }
```



/fix the problems in my code  
/tests add unit tests for my code  
/explain how the selected code works

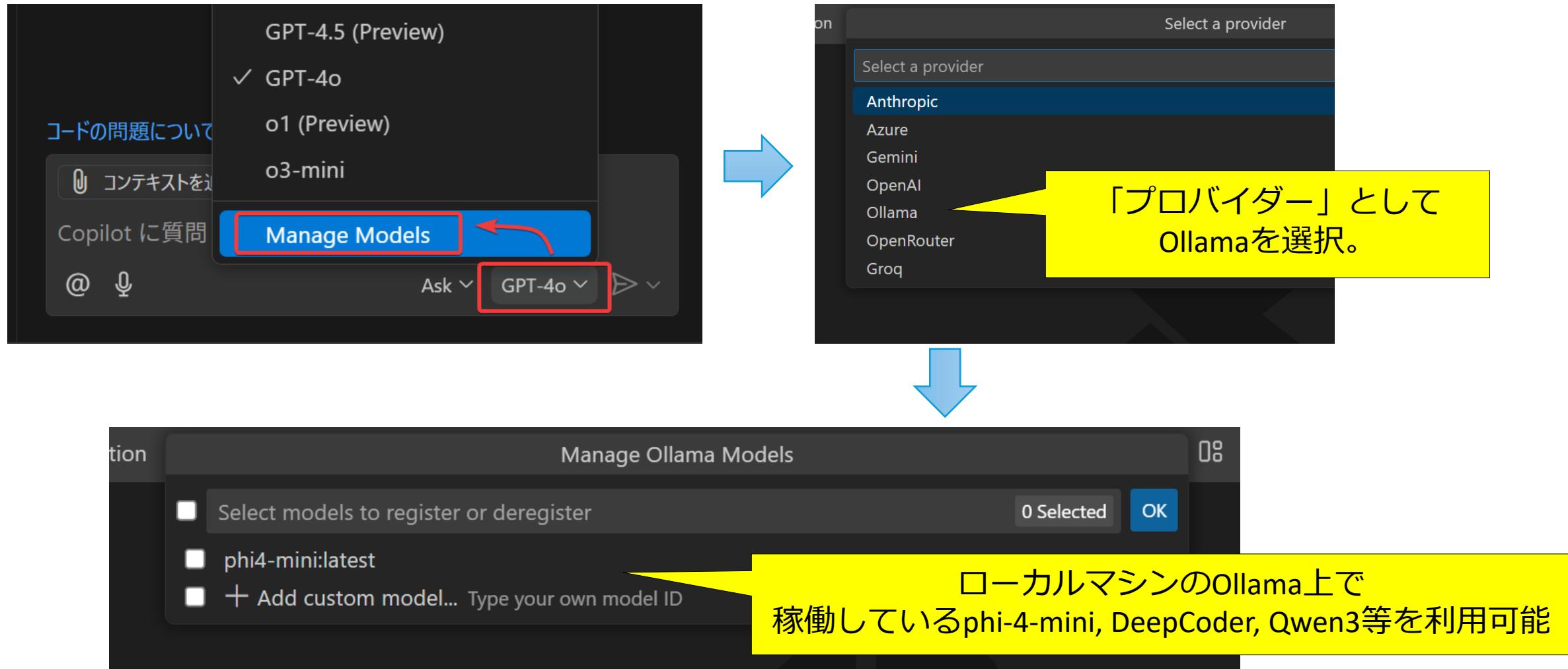
Ask Copilot

template.bicep Current file

@ 0 0 GPT 4o ▶

Ln 996, Col 15 Spaces: 2 UTF-8 LF Bicep 📁 📲

■ご参考: 「Manage Models」メニューが追加され、OllamaやOpenRouterなどの「プロバイダー」の広範なモデルが利用可能に。 (2025/4~)



「Visual Studio Code」と「Ollama」で簡単に始められる、安心・安全なローカルAI活用術：クラウドサービスだけじゃない！ ローカルPCやサーバ、Kubernetesで生成AI (5) - @IT

Github Copilot now supports Ollama and OpenRouter Models 🎉 : r/LocalLLaMA  
AI language models in VS Code

# モジュール2 GitHub Copilotの概要

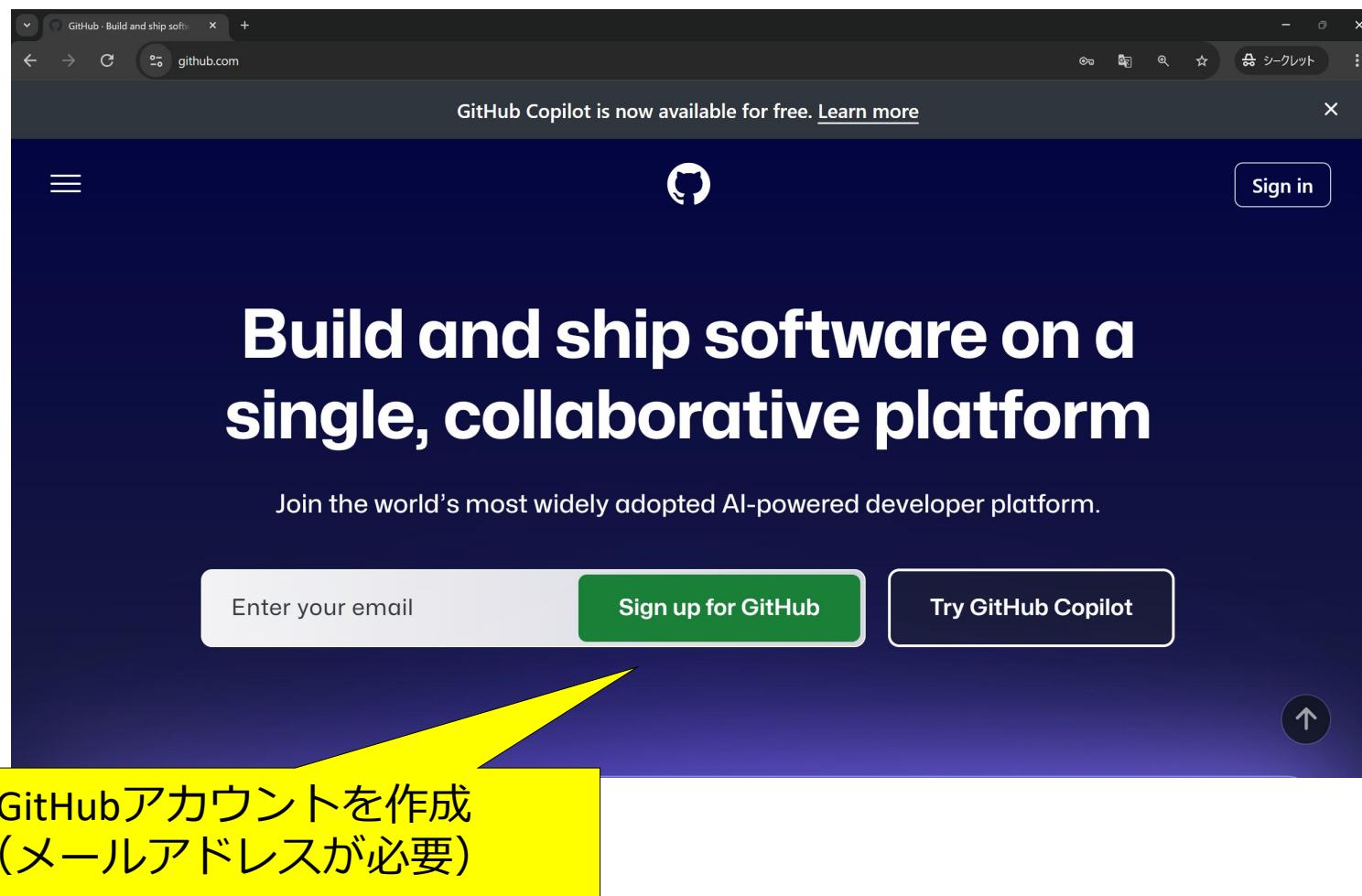
- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# GitHub Copilotの使用を開始する手順

- ・※Visual Studio Code と GitHub Copilot for Free を使う場合
- ・GitHubアカウントを持っていない場合は作成
- ・Visual Studio Code をインストールして起動
- ・Visual Studio Code 画面上部の「Copilot」アイコンをクリック
- ・「Sign in to Use Copilot for Free」をクリック
- ・GitHubアカウントにサインイン

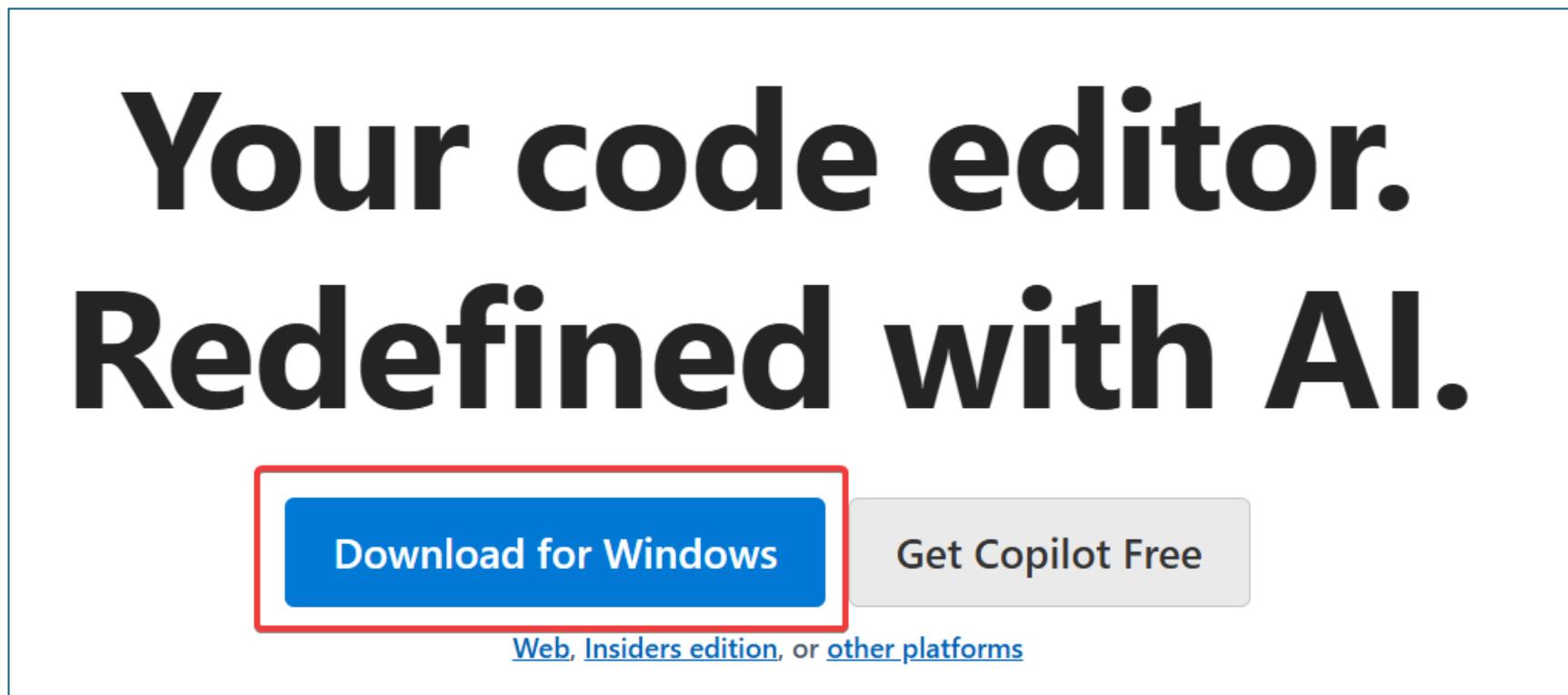
## ■GitHub Copilotの使用を開始する手順(1)GitHubアカウントの作成

<https://github.com/> にアクセス

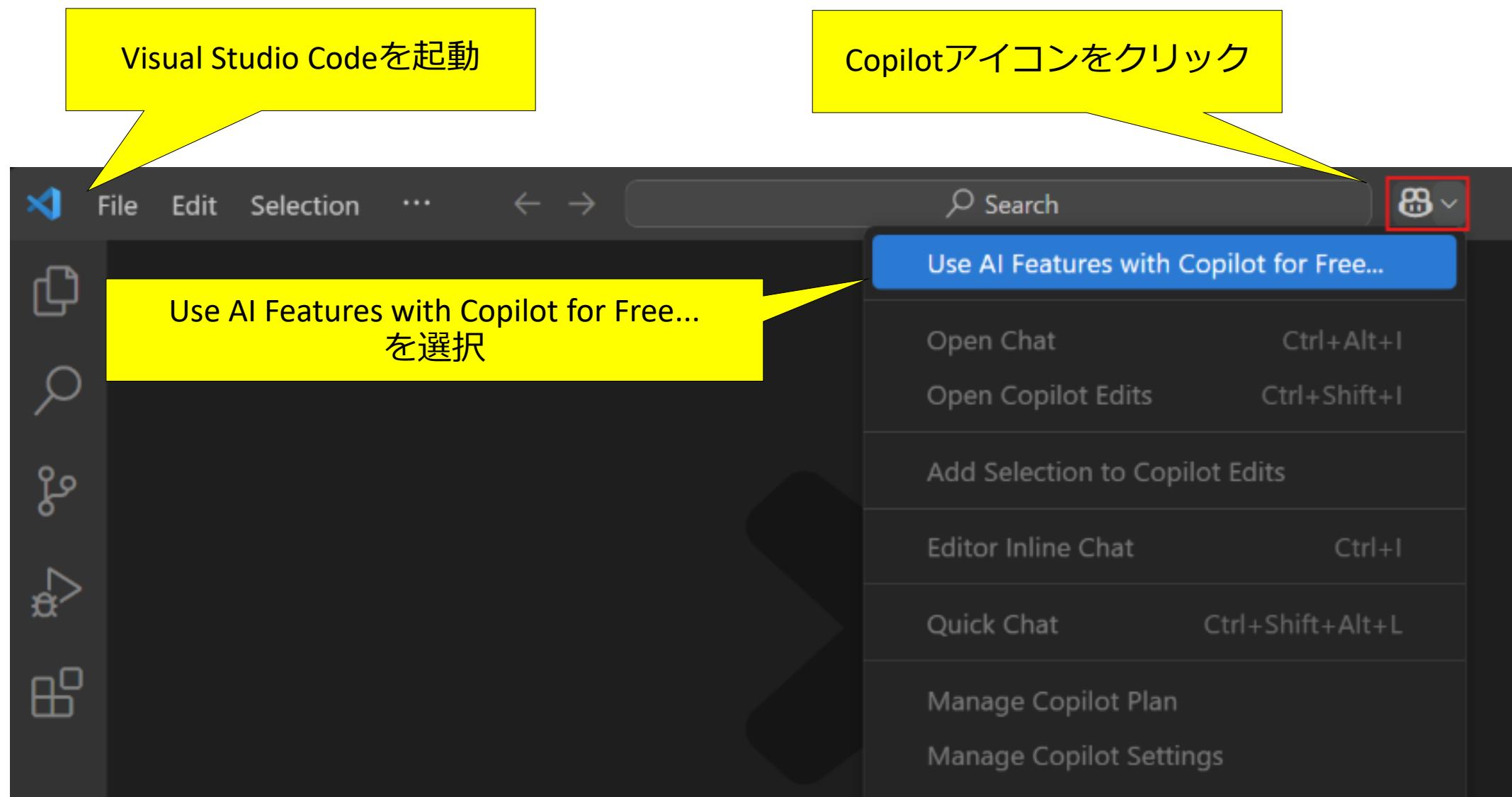


## ■GitHub Copilotの使用を開始する手順(2)Visual Studio Codeのインストール

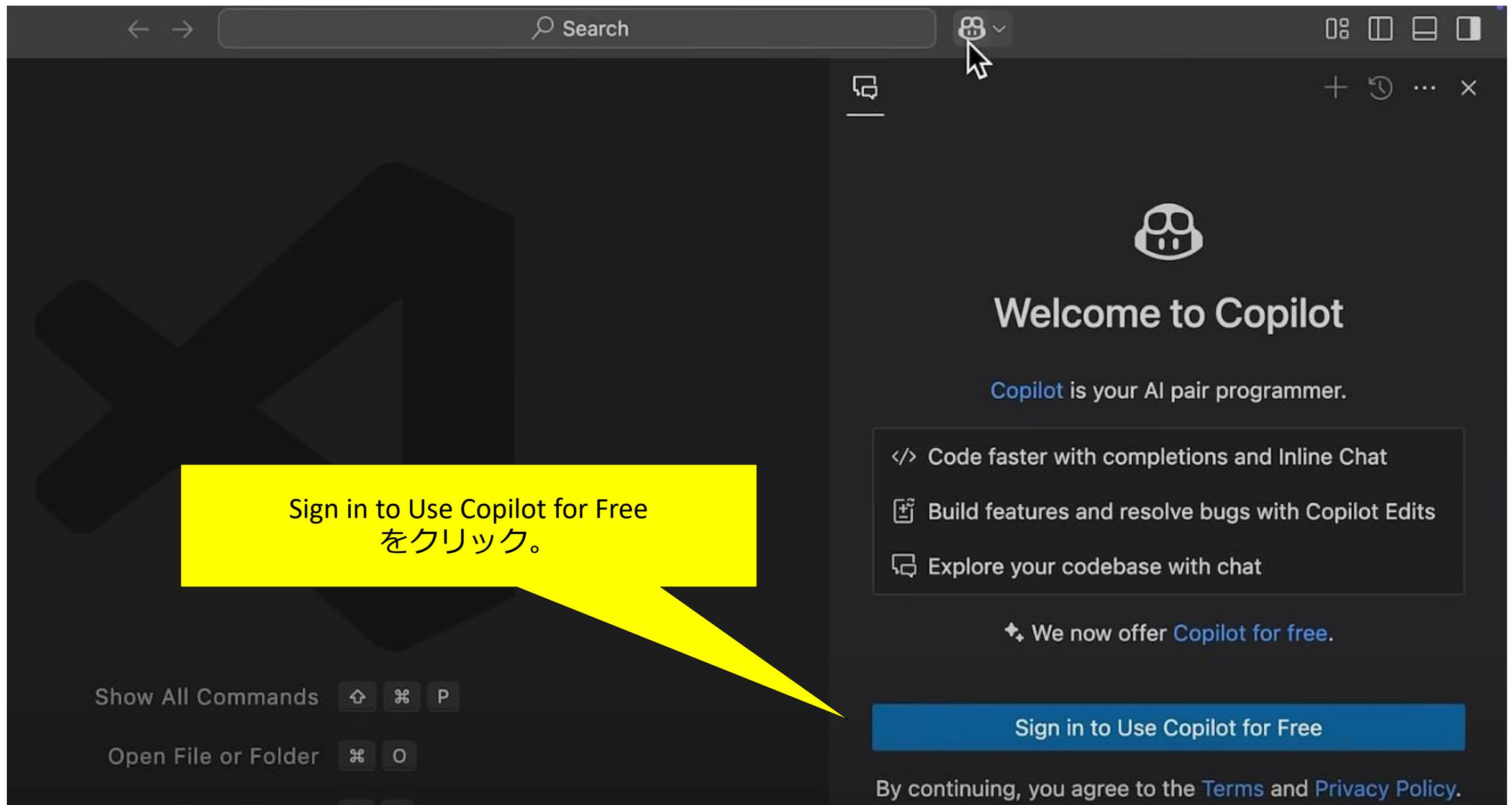
<https://code.visualstudio.com/> から、Visual Studio Code をダウンロードする。Windows/Mac/Linuxに対応。



## ■GitHub Copilotの使用を開始する手順(3)Visual Studio CodeでGitHub Copilot for Freeを選択



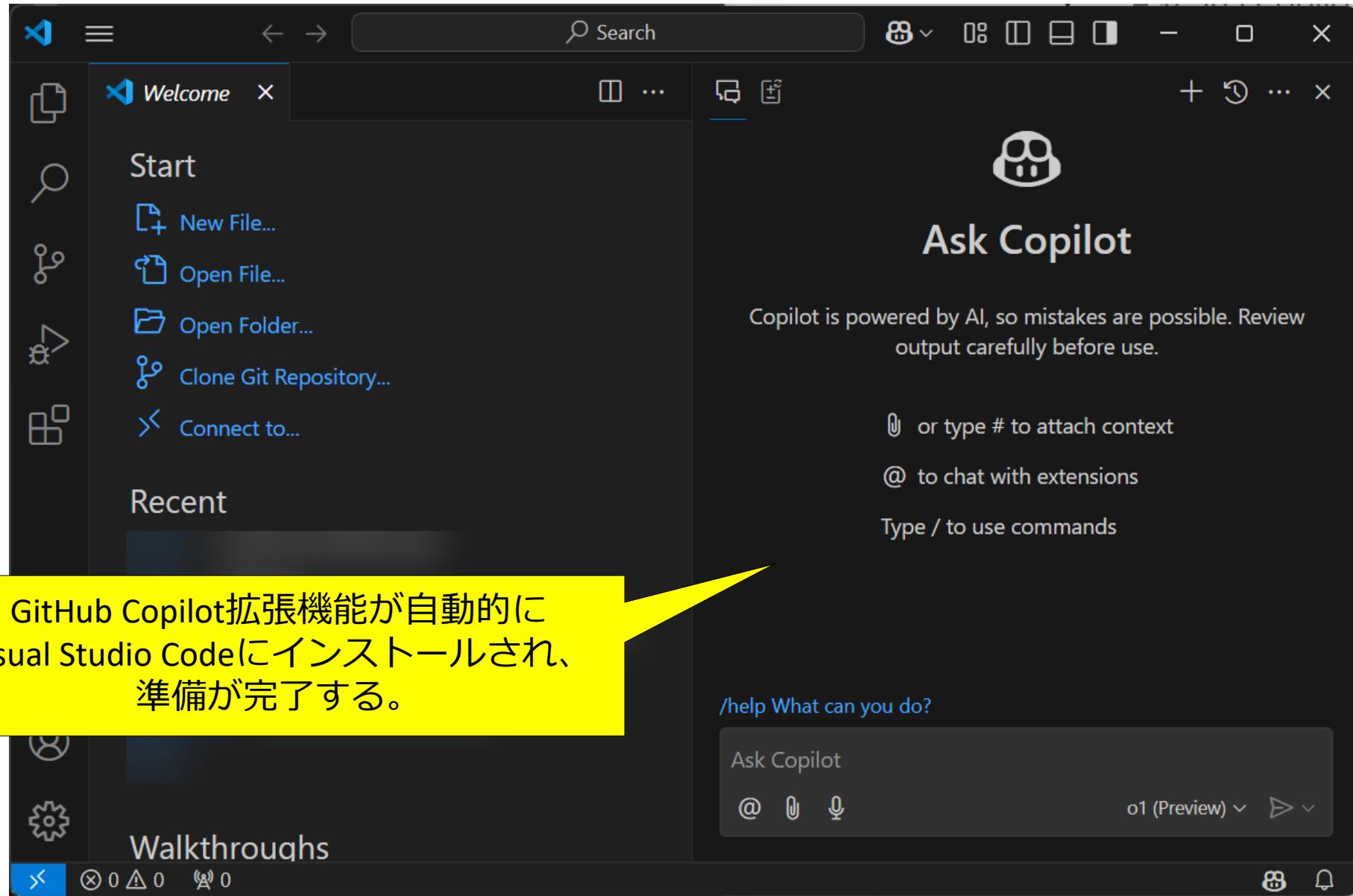
## ■GitHub Copilotの使用を開始する手順(4)GitHubアカウントにサインイン



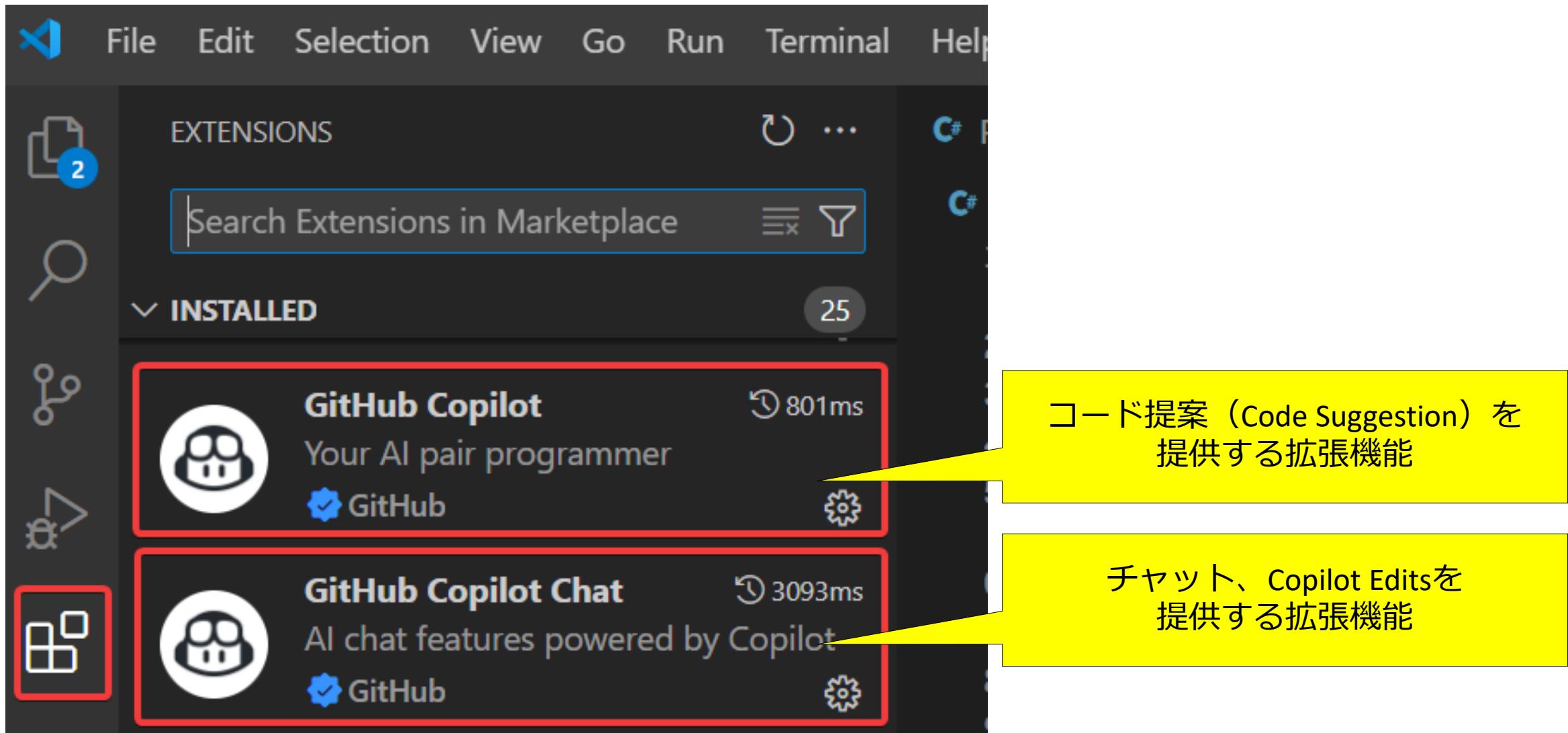
## ■GitHub Copilotの使用を開始する手順(4)GitHubアカウントにサインイン



## ■GitHub Copilotの使用を開始する手順(5)準備完了

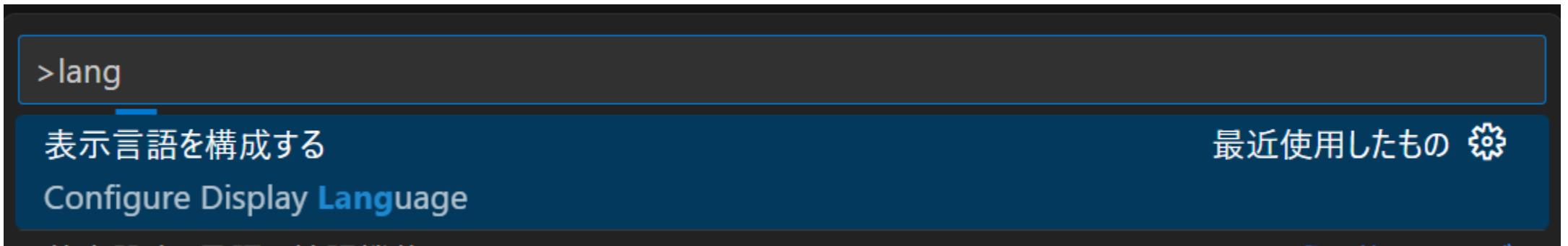


## ■このときVisual Studio Codeに2つの「拡張機能」がインストールされる

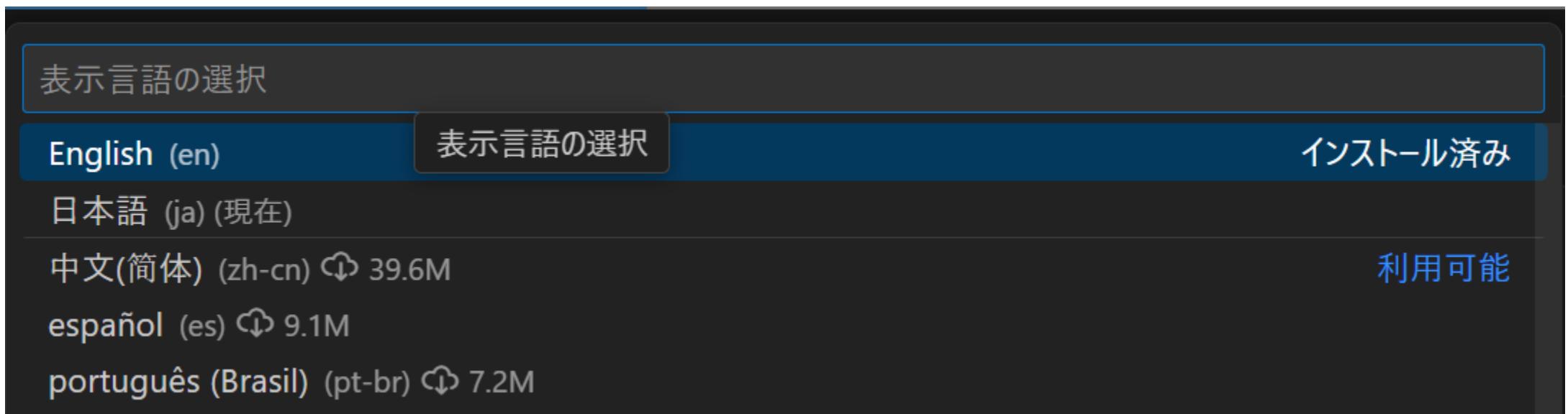


■お好みで、Visual Studio Codeの表示言語を変更します。

[F1], Configure Display Language



お好みの表示言語を選択



# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# GitHub Copilot の基本的な動作: **提案**

- GitHub Copilotは、プログラマが次に打ち込むであろうコードを予測して**提案**してくれる (code **suggestion**)
  - この動作は「**コード補完**」(code completion) とも呼ばれる
- **提案**は少し薄い色で表示される
- **提案**が表示されたら**Tabキー**を押して確定する。
- または**ESCキー**を押して**提案**を却下する。
- GitHub Copilot Freeでは月2000回までコード**提案**を利用できる

## ■GitHub Copilotの基本的な動作

```
60  class IDatabaseServiceImpl : IDatabaseService
61  {
62      public string GetData(int id) => id switch
63      {
64          1 => "りんご",
65          2 => "みかん",
66          _ => "(見つかりませんでした)"
67      };
68  }
69
70 // IDatabaseServiceを実装したクラスで、文房具のデータを返す
71 
```

プログラマが  
コメントを  
書くと・・・

## ■GitHub Copilotの基本的な動作

```
60  class IDatabaseServiceImpl : IDatabaseService
61  {
62      2 references
63      public string GetData(int id) => id switch
64      {
65          1 => "りんご",
66          2 => "みかん",
67          _ => "(見つかりませんでした)"
68      };
69
70 // IDatabaseServiceを実装したクラスで、文房具のデータを返す
71 class StationeryDatabaseService : IDatabaseService
{
    public string GetData(int id) => id switch
    {
        1 => "ボールペン",
        2 => "シャープペンシル",
        _ => "(見つかりませんでした)"
    };
}
```

コメントに沿って、文房具のデータを返すコードが生成される（ここでは、すでに書かれている上のコードを手本として、同じパターンのコードが生成されている）

GitHub Copilotがコードを提案（コメントに続く適切なコードを生成し、候補として表示）

コメントに沿って、インターフェースも実装されている

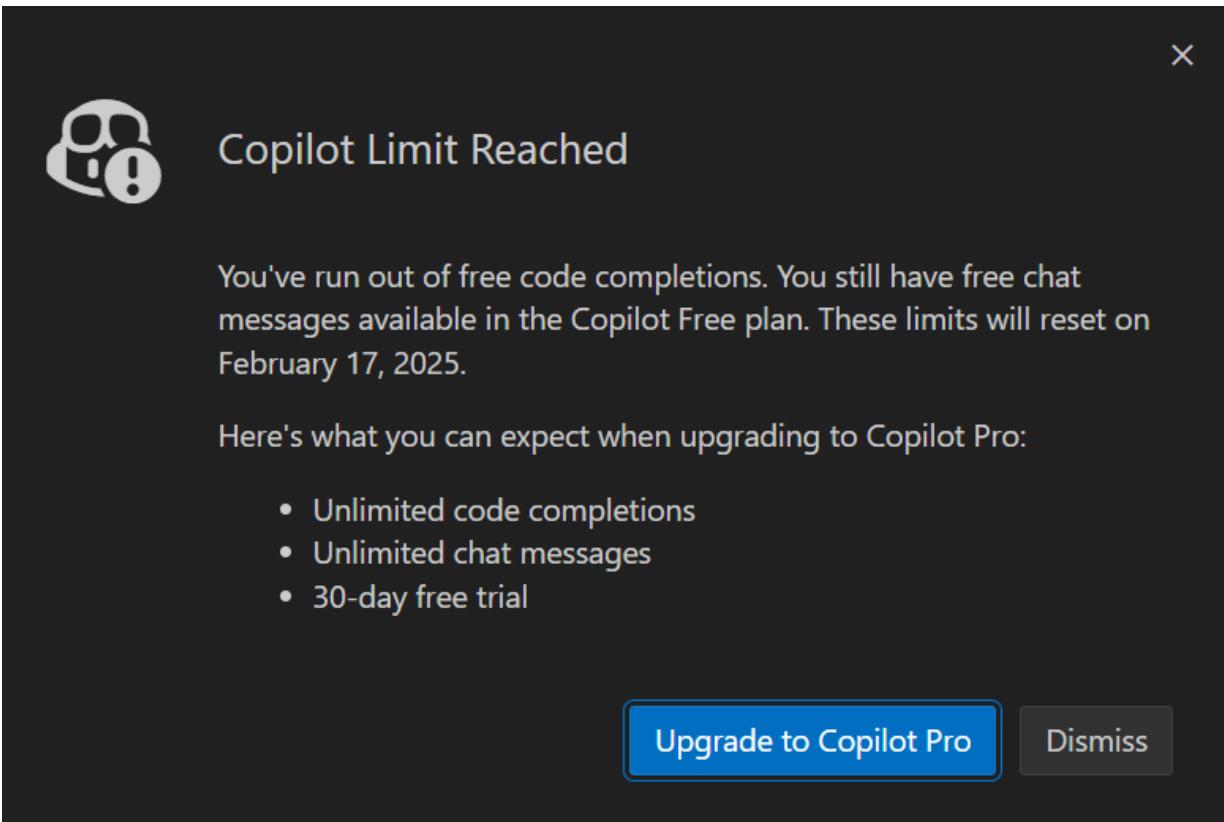
## ■GitHub Copilotの基本的な動作

```
60  class IDatabaseServiceImpl : IDatabaseService
61  {
62      2 references
63      public string GetData(int id) => id switch
64      {
65          1 => "りんご",
66          2 => "みかん",
67          _ => "(見つかりませんでした)"
68      };
69
70 // IDatabaseServiceを実装したクラスで、文房具のデータを返す
71 // 0 references
72 class StationeryDatabaseService : IDatabaseService
73 {
74     2 references
75     public string GetData(int id) => id switch
76     {
77         1 => "ボールペン",
78         2 => "シャープペンシル",
79         _ => "(見つかりませんでした)"
80     };
81 }
```

Tabキーを押して  
提案を確定

## ■GitHub Copilot Freeで利用できる制限に達した場合の表示例

### Copilot の制限に達しました



**無料のコード補完**が利用できなくなりました。この制限は 20XX 年 M 月 N 日にリセットされます。

1ヶ月経過すると  
また2000回の補完が利用できるようになる

GitHub Copilot Pro にアップグレードすると、以下のものが利用できる：

- 無制限のコード補完
- 無制限のチャット メッセージ

GitHub Copilot Pro は月10ドルの費用が発生（最初の30日は無料で利用可能）

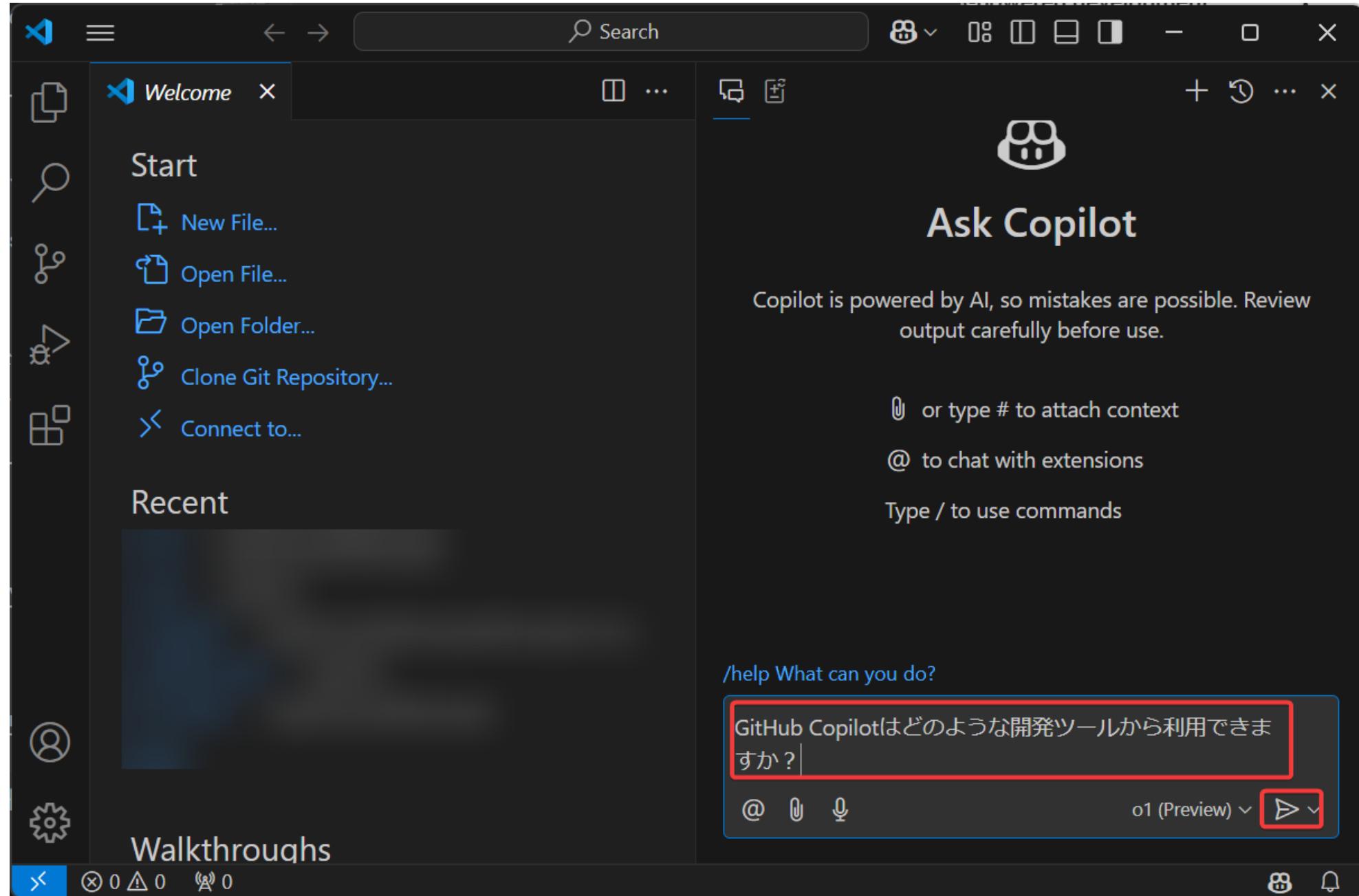
# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

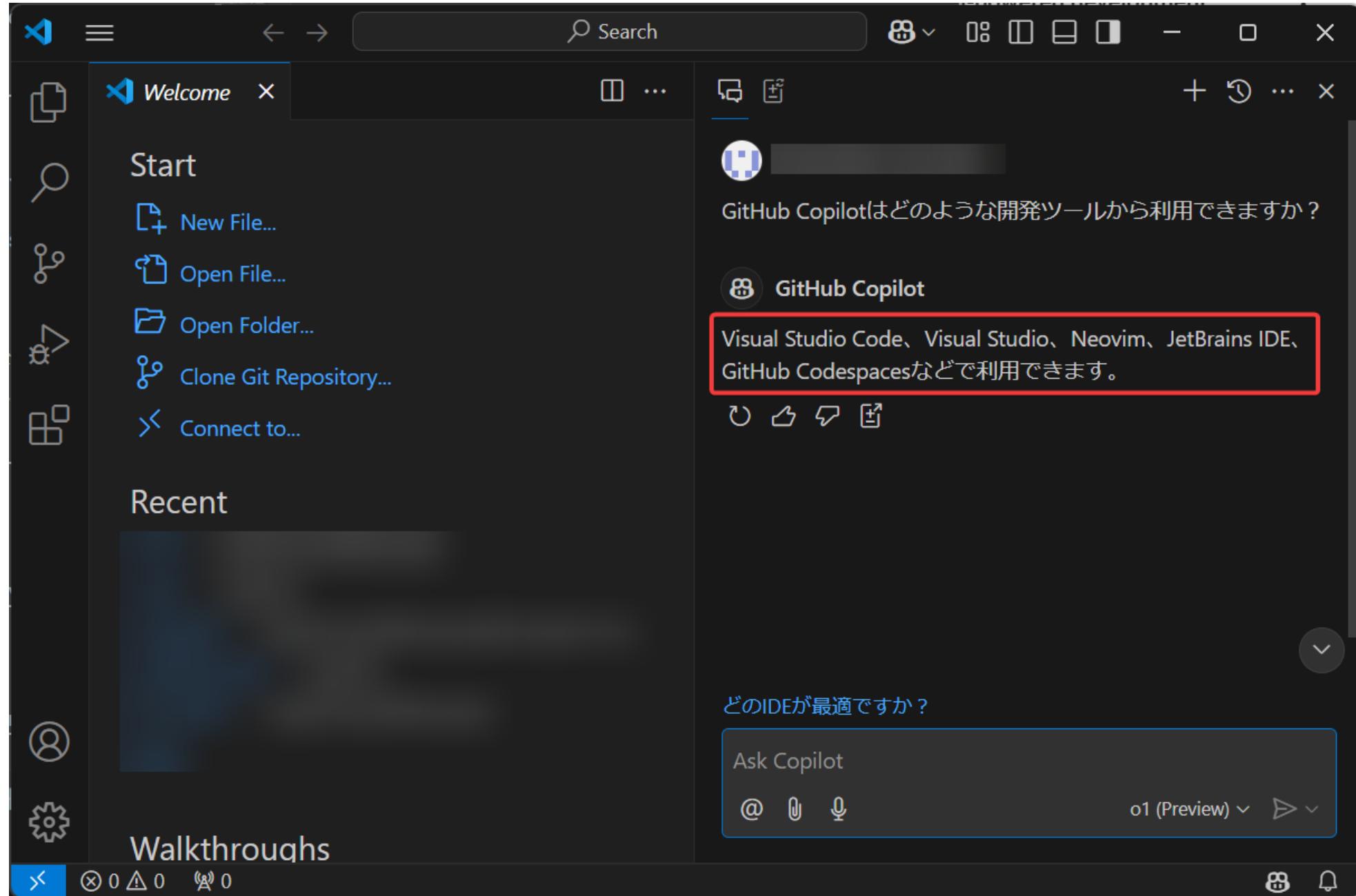
# GitHub Copilot **Chat** とは？

- ・チャット形式でAIに質問したり、コードの説明や修正案をもらったりできる機能
- ・2023/12/29 一般提供開始
- ・GitHub Copilot Freeの場合は月50メッセージまで使用可能
- ・利用方法
  - ・Visual Studio Codeなどの開発ツールから
  - ・GitHubのトップ画面（ダッシュボード）から
  - ・GitHub モバイルアプリから

## ■Visual Studio Code内から利用する例



## ■Visual Studio Code内から利用する例



# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# GitHub Copilot では どのようなことができるのか？（主なもの）

- GitHub Copilot の「コード提案」や、GitHub Copilot チャットの「Askモード」
  - コードのドキュメントを生成する
  - コメントからコードを生成する
  - 単体テストのコードを生成する
  - コードの品質やセキュリティを改善する
- GitHub Copilot チャットの「Editモード」
  - 複数のファイルを同時に編集する
- GitHub Copilot チャットの「Agent モード」
  - コマンドを提案・実行する
  - MCPサーバー経由で外部サービスを利用する
- GitHub Copilot コーディングエージェント
  - イシューの担当者をCopilotに割り当て → Copilotがそのイシューを解決するプルリクエストを作成

## ■ご参考: 以下のような使い方も可能です

- 一般的な質問への回答（開発に関する質問に答えてもらうなど）
- プロンプトの生成、改善
- コードのアップデート（Python 2で書かれたコードをPython 3にアップデートするなど）、コンバート（COBOLのプログラムをJavaに書き換えるなど）
- 選択部分やGitの変更部分のレビューの生成（GitHub Code Review）
- HTML・CSS・JavaScriptの生成
- テストデータの生成（CSVなど）
- 設定ファイルの生成（YAML、JSONなど）
- IaCの定義ファイルの生成（Terraform、ARM テンプレート、Bicepなど）
- GitHub Actionsのワークフローの生成（YAML）
- UMLダイアグラムなどの図の生成（SVG、Mermaidなど）
- SQL文の生成
- コマンドやスクリプトの生成
- プログラムのエラーログや、コマンドのエラーメッセージなどの原因究明と対処方法の提案
- コミットメッセージの生成、プルリクエストのタイトルと説明文の生成
- @azure や @askstackoverflow といった「GitHub Copilot Extensions」の追加
- 独自の「GitHub Copilot Extensions」の開発

# ■ご参考: 「Copilot Chat クックブック」で、GitHub Copilot のチャットを使った作業の例（プロンプト例）が確認できる

The screenshot shows the GitHub Docs interface for the Copilot Chat Cookbook. The left sidebar has a navigation tree with sections like Home, GitHub Copilot, Quickstart, and Copilot Chat Cookbook. The Copilot Chat Cookbook section is expanded, showing a sub-section for "All prompts". The main content area is titled "Copilot Chat クックブック" and describes examples of prompts used in GitHub Copilot Chat. It features a "Spotlight" section with three cards: "Generate unit tests", "Improving code readability and maintainability", and "Debugging invalid JSON". Below this, there's a search bar and filter options for "記事を検索する", "カテゴリ: All", "複雑性: All", and "フィルターのリセット". At the bottom, there are three more cards: "無効な JSON のデバッグ", "API のレート制限の処理", and "考えられる機能実装について確認する". Each card includes a small icon, a title, a brief description, and two tags: "Debugging code" and "Intermediate".

[Copilot Chat クックブック - GitHub Docs](#)

# ■ご参考: 「Copilot Chat クックブック」で、GitHub Copilot のチャットを使った作業の例（プロンプト例）が確認できる



## 無効な JSON のデバッグ

Copilot Chat は、JSON データの構文エラーや構造上の問題を特定して解決できます。

[Debugging code](#) [Intermediate](#)



## API のレート制限の処理

Copilot Chat による再試行ロジックの実装を検出するコードの提案は、API のレート制限を処理するのに役立ちます。

[Debugging code](#) [Intermediate](#)



## 考えられる機能実装について確認する

Copilot Chat は、1 つの機能を実装するためのさまざまなアプローチについて確認するのに役立ちます。

[Functionality analysis](#) [Intermediate](#)



## ユーザー フィードバックの分析と取り込み

Copilot Chat を使うと、ユーザーのフィードバックをプロジェクトに組み込むプロセスを強化できます。

[Functionality analysis](#) [Intermediate](#)



## コードの読みやすさと保守容易性を改良する

Copilot Chat から、コードを理解し、保守しやすくする方法の提案を受けることができます。

[Refactoring code](#) [Simple](#)



## lint エラーの修正

Copilot Chat から、コード リンターによって特定された issue を修正する方法の提案を受けることができます。

[Refactoring code](#) [Intermediate](#)



## パフォーマンスの最適化のためのリファクタリング

Copilot Chat は、実行の遅いコードを高速化する方法を提案できます。

[Refactoring code](#) [Simple](#)



## 設計パターンを実装するためのリファクタリング

Copilot Chat は、コードの改善に使用できる設計パターンを提案できます。

[Refactoring code](#) [Intermediate](#)



## データ アクセス層のリファクタリング

Copilot Chat は、ビジネス ロジックからデータ アクセス コードを切り離し、アプリケーションの保守とスケーリングを容易にする方法を提案できます。

[Refactoring code](#) [Advanced](#)

# ■ご参考: 「Copilot Chat クックブック」で、GitHub Copilot のチャットを使った作業の例（プロンプト例）が確認できる



## ビジネスロジックの UI コンポーネントからの分離

Copilot Chat を使うと、ビジネスロジックをユーザーインターフェイスコードから分離できます。これでアプリケーションの保守と拡張が容易になります。

[Refactoring code](#) [Advanced](#)



## 横断的関心事の処理

Copilot Chat は、コードが配置されているメソッドまたは関数の主要な関心事以外の関心事に関連するコードを回避するのに役立ちます。

[Refactoring code](#) [Intermediate](#)



## 複雑な継承階層の簡略化

Copilot Chat は、コードをリファクタリングして継承の複数のレイヤーにクラスが存在しないようにするために役立ちます。

[Refactoring code](#) [Intermediate](#)



## データベースのデッドロックまたはデータ整合性の問題の解決

Copilot Chat を使うと、コードが原因でデータベースの操作が遅くなったりブロックされたりすることや、テーブルのデータが不足したり正しくなくなったりすることを避けるのに役立ちます。

[Refactoring code](#) [Advanced](#)



## コードを別のプログラミング言語に変換する

Copilot Chat は、同じ操作を別のプログラミング言語で実行するようにコードを書き直すのに役立ちます。

[Refactoring code](#) [Simple](#)



## レガシコードの文書化

Copilot Chat は、レガシコードの文書化に役立ちます。

[Documenting code](#) [Simple](#)



## レガシコードの説明

Copilot Chat は、よくわからないコードの説明に役立ちます。

[Documenting code](#) [Simple](#)



## 複雑なアルゴリズムまたはロジックの説明

Copilot Chat は、複雑なアルゴリズムやロジックに関する明確で簡潔なドキュメントを追加するのに役立ちます。

[Documenting code](#) [Intermediate](#)



## ドキュメントとコードの変更の同期

Copilot Chat は、コードのドキュメントを最新の状態に保つのに役立ちます。

[Documenting code](#) [Intermediate](#)

## ■ご参考: 「Copilot Chat クックブック」で、GitHub Copilot のチャットを使った作業の例（プロンプト例）が確認できる



### 単体テストを生成する

Copilot Chat は、関数の単体テストを生成するために役立ちます。

[Testing code](#)

[Intermediate](#)



### レイヤーを抽象化するモック オブジェクトを作成する

Copilot Chat は、単体テストに使用できるモック オブジェクトの作成に役立ちます。

[Testing code](#)

[Intermediate](#)



### Web ページのエンドツーエンド テストを作成する

Copilot Chat は、エンドツーエンド テストの生成に役立ちます。

[Testing code](#)

[Advanced](#)



### コード内の既存の脆弱性を見つける

Copilot Chat を使うと、コード内の一般的な脆弱性を見つけて、修正の提案を受けることができます。

[Security analysis](#)

[Intermediate](#)

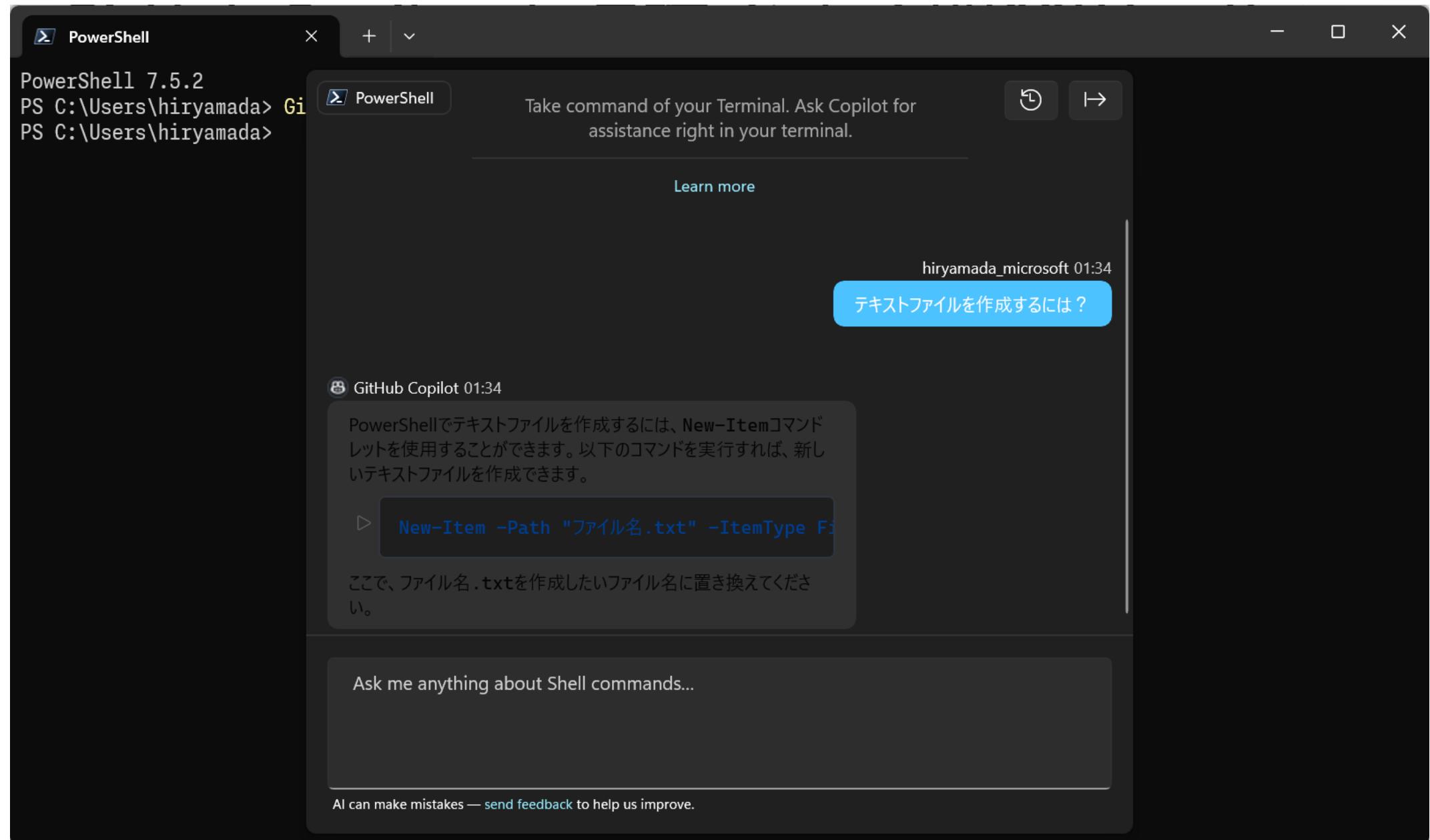
# モジュール1 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# GitHub Copilot と連携できる開発ツール

- Visual Studio / Visual Studio Code 本コースではVisual Studio Codeでの使用方法を解説
- JetBrains IDE（IntelliJ IDEA、PyCharm、Rider、WebStorm等）
- Vim / Neovim
- Xcode
- Eclipse
- Azure Data Studio
- GitHub CLI (`gh copilot explain` / `gh copilot suggest`)
- GitHub Mobile アプリ
- Windows Terminal

## ■参考: Windows Terminal (カナリア版) でのGitHub Copilot Chatの利用例



[GitHub Copilot Chat in Windows Terminal を試してみよう](#)

## ■各開発ツールでの導入方法は公式サイトのドキュメントで確認できる

The screenshot shows the GitHub Docs page for "GitHub Copilot を使用して IDE でコードの提案を取得する". The page header includes the GitHub logo and "GitHub Docs", with a dropdown menu for "Version: Free, Pro, & Team". The main title is "GitHub Copilot を使用して IDE でコードの提案を取得する". Below it, a sub-section title reads "GitHub Copilot を使用して、エディターでコードの提案を取得する". A horizontal navigation bar lists supported tools: Azure Data Studio, JetBrains IDEs, Vim/Neovim, Visual Studio, Visual Studio Code, and Xcode. A green button at the bottom left says "無料で始める". A yellow callout bubble points to the "JetBrains IDEs" link, stating "各開発ツールでの GitHub Copilot 導入方法のドキュメントが利用可能". Another yellow callout bubble points to the bottom right, stating "詳しくはこちらのページを参照". The footer contains the URL "GitHub Copilot を使用して IDE でコードの提案を取得する - GitHub Docs".

GitHub Docs

Version: Free, Pro, & Team

三 GitHub Copilot / GitHub Copilot を使用する / コードの提案を取得する

## GitHub Copilot を使用して IDE でコードの提案を取得する

GitHub Copilot を使用して、エディターでコードの提案を取得する

Azure Data Studio    JetBrains IDEs    Vim/Neovim    Visual Studio    Visual Studio Code    Xcode

無料で始める

各開発ツールでの GitHub Copilot 導入方法のドキュメントが利用可能

詳しくはこちらのページを参照

[GitHub Copilot を使用して IDE でコードの提案を取得する - GitHub Docs](#)

# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# よくあるご質問

- GitHub Copilotによって、プログラマの仕事はなくなりますか？

→いいえ。GitHub Copilotはプログラマの仕事を支援し、生産性を向上させるツールと位置づけられています。プログラマの仕事を完全になくしたり、GitHub Copilotに仕事が奪われてしまうことはありません。（GitHub社の公式見解）

▼ GitHub Copilot は、コード生成を完全に自動化し、開発者を置き換えることを目的としていますか? ☺

いいえ。Copilot は開発者の効率を高めるためのツールです。開発者に代わるものではありません。開発者は、出所不明のサードパーティ コードに適用するのと同じ種類の安全策と注意を継続的に適用する必要があります。

- この製品は「Autopilot」ではなく「Copilot」と呼ばれ、監視なしで提案を生成することを意図したものではありません。Copilot の提案に対しては、サードパーティのコードに使用するのとまったく同じ種類の安全策と注意を払う必要があります。
- サードパーティ コードの使用に関するベスト プラクティスを特定することは、このセクションの範囲外です。ただし、組織が現在使用しているプラクティス(厳格な機能テスト、コードスキャン、セキュリティ テストなど)が何であれ、Copilot の提案に従ってこれらのポリシーを継続する必要があります。さらに、コードエディターまたはエディターが、生成されたコードをレビューする前に自動的にコンパイルまたは実行しないことを確認する必要があります。

# よくあるご質問

## ■ GitHub Copilotが生成するコードは常に正確ですか？

→いいえ。一般的に生成AIの出力は必ずしも正しいとは限りません。

→GitHub Copilotも生成AIを使用しているため同様であり、必ずしも正しいコード（期待したコード）を生成するとは限りません。したがって人間（プログラマ）が、生成されたコードをよく確認する必要があります。

# よくあるご質問

- 「Cursor」、「Cline」、「Roo Code」（Roo-Cline）、「Devin」などのAI開発ツールと比べて、GitHub Copilotはどうですか？

→メリット: Microsoft（GitHub）の公式ツールとして安心して利用できます。企業向けの機能も充実していて、企業での採用実績も多数あります。

→違い: GitHub Copilotは「全自动でコードやアプリを開発するツール」ではないです。あくまでコパイロット（副操縦士）、つまり人間のプログラマの開発作業を支援するツール、という位置付けです。

# よくあるご質問

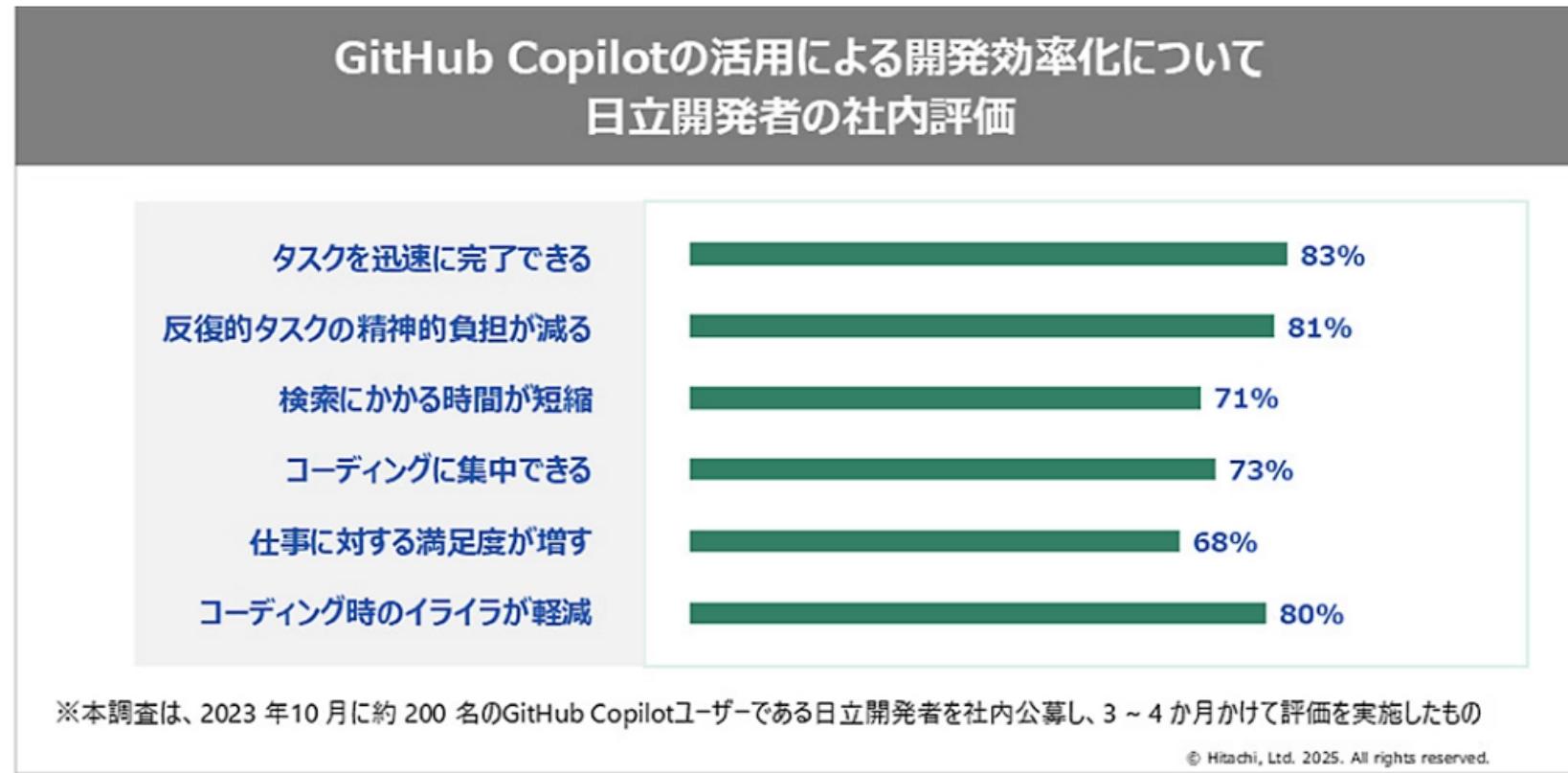
■ GitHub Copilotの企業での採用事例はありますか？  
→非常にたくさんあります。

Github Copilot 採用企業・レポートをまとめてみた (2023年08月07日時点) #GitHub - Qiita

TC3株式会社: GitHub Copilot for Businessを導入  
no plan株式会社: 「ChatGPT & GitHub Copilot手当」を全員に導入  
株式会社M&Aクラウド: GitHub Copilot導入におけるハードルの整理  
STORES株式会社: GitHub Copilot for Businessを導入  
株式会社SmartHR: GitHub Copilotを導入し、開発体験の向上に貢献  
OLTA株式会社: ChatGPTとGitHub Copilotを全社導入  
株式会社Leaner Technologies: GitHub Copilot for Businessを導入  
株式会社Gaudiy: GitHub Copilotを導入して1ヶ月経過  
ミスリル株式会社: ゲーム開発の現場でGitHub Copilotを検証  
NewCyte合同会社: GitHub Copilotを使ったプログラミングにチャレンジ  
株式会社ワントゥーテン: GitHub Copilotを全社導入  
株式会社スマートバンク: GitHub Copilot for Businessを導入  
株式会社カケハシ: ChatGPTやGitHub Copilotの活用に関する考慮点  
株式会社ZOZO: GitHub Copilotの全社導入とその効果  
レバレジーズ株式会社: GitHub Copilotの全社導入過程でのノウハウ  
FastLabel株式会社: 開発社員向け福利厚生としてGitHub Copilotを導入  
ナイル株式会社: GitHub Copilotを全エンジニアに導入

株式会社スカイディスク: GPT-4を搭載したGitHub Copilot Xを導入  
株式会社ツクリバ: GitHub Copilotの利用環境を全エンジニアに提供開始  
ラグナロク株式会社: エンジニア希望者全員にGitHub Copilotを支給  
株式会社TOKIUM: ChatGPT PlusとGitHub Copilotを導入  
ENECHANGE株式会社: GPT-4とGitHub Copilotの導入  
株式会社ヘッドウォータース: GitHub Copilot for Businessの全社導入  
Crezit Holdings株式会社: GitHub Copilotの導入  
株式会社メタップス: GitHub Copilot Xを導入  
株式会社ワンキャリア: GitHub Copilotを導入  
株式会社サーバーワークス: ChatGPT PlusとGitHub Copilotの補助制度を導入  
株式会社Ridge-i: GitHub CopilotとChatGPT Plusを導入  
株式会社アイスリーデザイン: GitHub Copilotを導入  
BRANU株式会社: GitHub Copilotを導入  
株式会社アブリズム: Microsoft 365 CopilotとGitHub Copilotを導入  
株式会社エブリー: ChatGPT PlusとGitHub Copilotを導入  
株式会社スタディスト: AIとのセキュアな相談を可能にし、従業員の生産性を向上  
パーソルホールディングス株式会社: GitHub Copilotを導入

# 日立製作所が GitHub Copilot の活用で開発生産性を向上。社内評価、開発フレームワークとの連携、コミュニティ活動を通して、さらなる生成 AI 活用を推進



日立製作所で行われた GitHub Copilot に対するアンケート調査の結果

[日立製作所が GitHub Copilot の活用で開発生産性を向上。社内評価、開発フレームワークとの連携、コミュニティ活動を通して、さらなる生成 AI 活用を推進 | Microsoft Customer Stories](#)

■マイクロソフトの「お客様事例」ではGitHub Copilot導入事例を数十件確認できる。Kakaku.com様、東芝テック様、パナソニックコネクト様、SBテクノロジー様など。

## お客様事例を検索する

github copilot

次でフィルター

github copilot x

すべてクリア

並べ替え順 v

関連 v

The screenshot shows a search interface with a search bar containing 'github copilot'. Below the search bar are filter options: '次でフィルター' (Next Filter) with 'github copilot' selected, and sorting options: 'すべてクリア' (Clear All), '並べ替え順' (Sort by Recency), and '関連' (Related). Two search results are displayed:

- Kakaku.com**  
業界: 電気通信とメディア  
開発者の生産性向上だけではなく職場満足度向上にも大きな効果、トライアルでの評価を経て開発現場全体に GitHub Copilot を導入
- TOSHIBA**  
業界: 工業および製造業  
新規事業への進出でソフトウェア開発の比重が増大した東芝テック、その生産性向上を目指したPoCで明らかになった「GitHub Copilot の 2つの効果」

Each result card includes icons for GitHub Copilot, Microsoft Teams, and Azure OpenAI, and a blue '事例を読む' (Read Case Study) button.

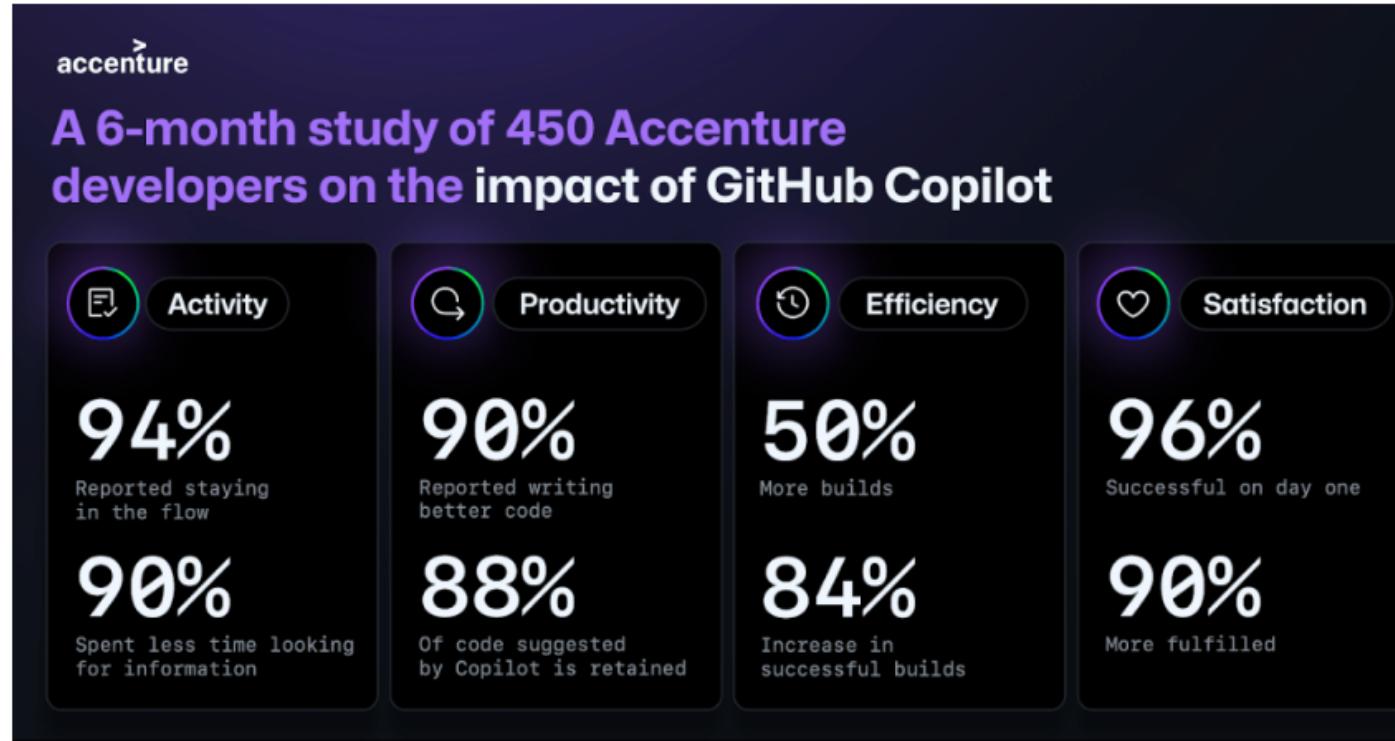
# よくあるご質問

■ GitHub Copilotの導入効果はどのくらいですか？

→右のようなデータがあります。  
(GitHubとAccenture社調べ)

開発者がより集中して、より質の高いコードを、より効率的に開発でき、仕事の満足度も高まります。

<https://github.blog/jp/2024-03-04-github-copilot-enterprise-is-now-generally-available/>



- GitHub Copilotは、Accentureの開発者がフロー状態を維持し、作業の中止を最小限に抑えるのに貢献：開発者の94%は、GitHub Copilotを使用することでフロー状態を維持し、反復的なタスクに費やす労力を減らすことができたと回答しています。また、開発者の90%は、情報検索に費やす時間を短縮できたと回答しています。
- GitHub Copilotは、開発者がより質の高いコードを本番環境にプッシュできるよう支援：GitHub Copilotの提案したコードの88%がエディターに保持されました。また、開発者の約90%は、GitHub Copilotの提案が含まれているコードをコミットしたと回答しています。
- GitHub Copilotは、開発者が質の高いコードを書き、仕事をしながらスキルアップに貢献：開発者の90%はGitHub Copilotを使って質の高いコードを書けたと回答し、開発者の約95%はGitHub Copilotの提案から学びがあったと回答しています。

# よくあるご質問

- GitHub Copilotの導入効果（ROI）を測定する方法はありますか？

→はい。以下の記事が参考になると思います。

<https://resources.github.com/learn/pathways/copilot/essentials/measuring-the-impact-of-github-copilot/>

## GitHub Copilotの影響を測定する



Ryan Salva // 製品担当副社長 // GitHub

# よくあるご質問

## ■ GitHub Copilotは無料でも利用できますか？

→はい、すでにご紹介した通り、無料（Free）プランもありますが、提案・チャットの回数に制限があるため、業務で使用するのであれば有料プラン（Pro、Business、Enterprise）がおすすめです。

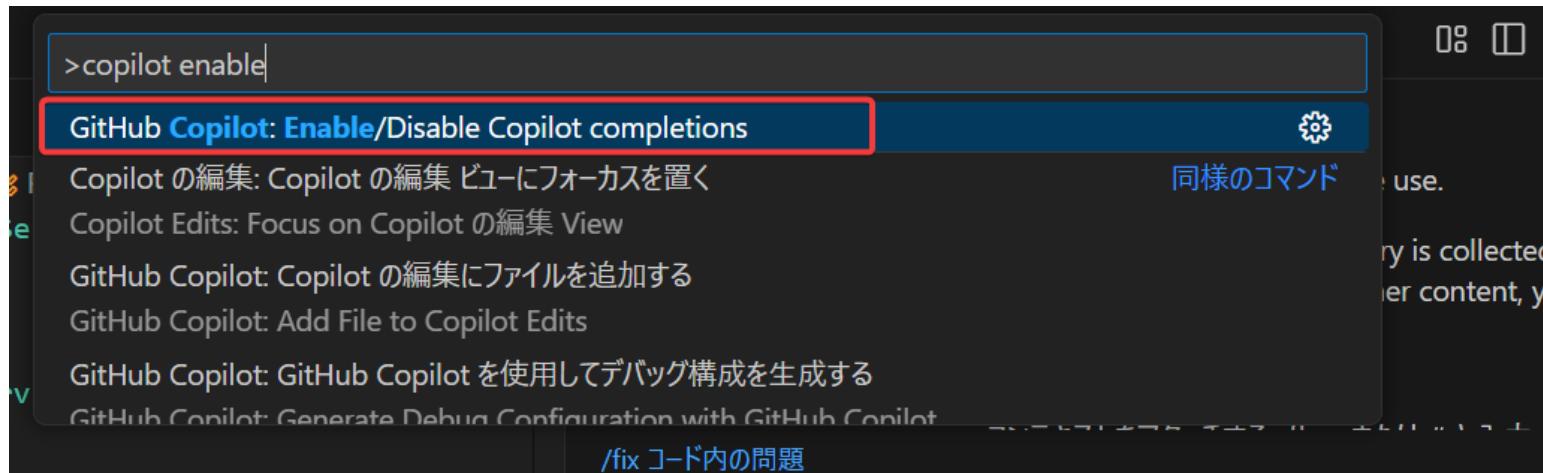
## ■ 有料プランは従量課金ですか？（たくさん使うとよりお金がかかりますか？）

→基本的には定額料金です。ユーザーあたり・月あたりで 10 USD（Proプランの場合）といったように価格が決められています。ただし、高度なモデルを利用したチャットなどでは「プレミアムリクエスト」が利用され、各プランで設定された回数を超えた場合は追加で課金されます。

# よくあるご質問

■ GitHub Copilotのコード提案を普段はOFFにしておき、必要な場合にのみONにして利用できますか？

→可能です。[F1]または[Ctrl + Shift + P]を押してコマンドパレットを表示し、[GitHub Copilot: Enable/Disable Copilot completions]を呼び出すことで、提案（補完）をON/OFFできます

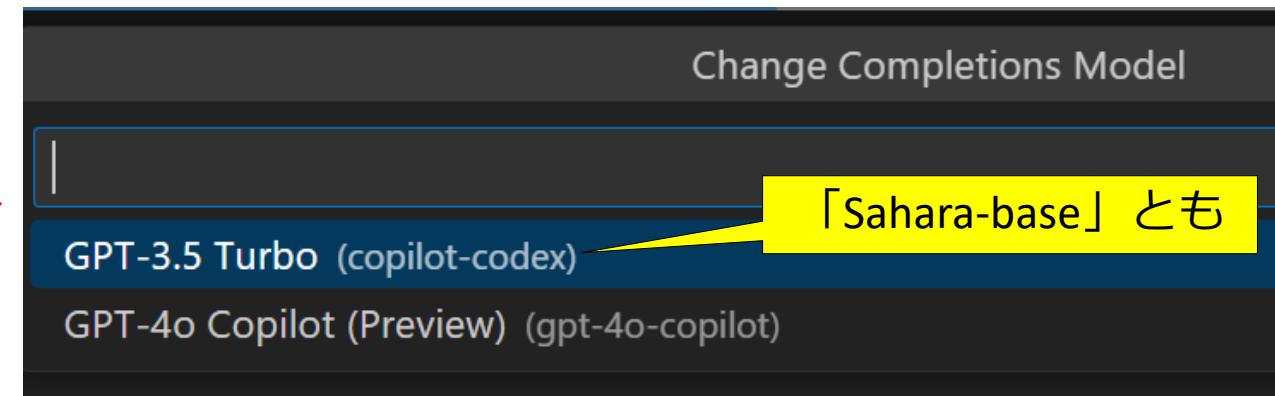
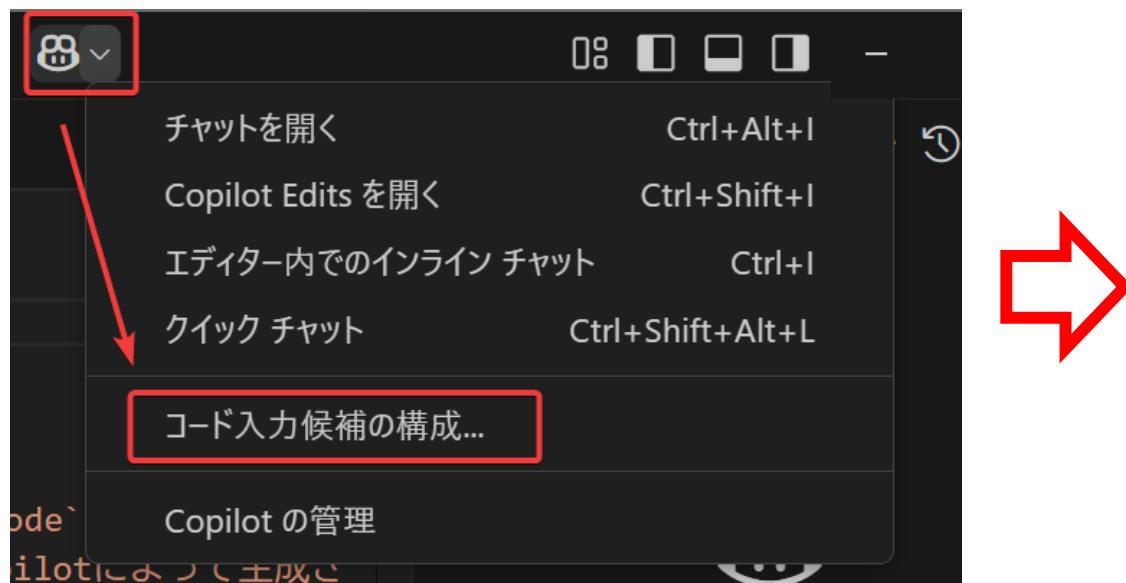


# よくあるご質問



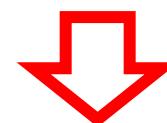
■ GitHub CopilotのチャットやEditsでは、使用するモデルを変更できますが、コード候補（コード補完）で使用するモデルは変更できますか？

→はい、可能です。



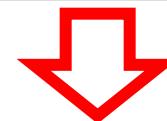
## ■ご参考: コード候補（コード補完）で使用されるモデルの変更の歴史

2021/6/29: GitHub Copilot テクニカルレビュー開始、OpenAI Codex モデルを使用。



2023/3/20: OpenAI社がCodexモデルを廃止 (deprecated)

202?/?/??: GitHub Copilot のモデルが GPT-3.5-Turboの派生版「Sahara-base」に変更



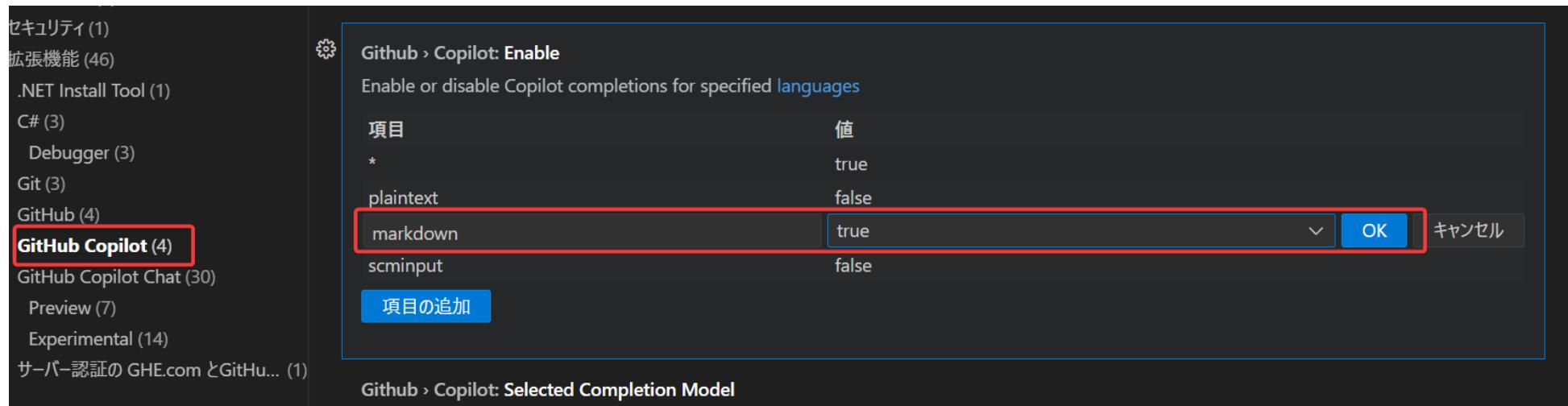
2025/2/18: GPT-4o が選択可能に (プレビュー)

<https://github.blog/news-insights/product-news/introducing-github-copilot-ai-pair-programmer/>  
GitHub・服部氏が語る「GitHub Copilot」の裏側 “エンジニアの開発生産性を上げる”ために重視している2つのポイント | ログミーBusiness

New GPT-4o Copilot code completion model available now in public preview for Copilot in VS Code - GitHub Changelog

# よくあるご質問

■テキストファイルやMarkdownファイルではGitHub Copilotの提案が表示されないです  
→設定で変更できます。



# よくあるご質問

■ GitHub Copilotはどのようなプログラミング言語で使用できますか？

→C#、Java、Python、JavaScript、TypeScriptなど、非常に多くのプログラミング言語に対応しています。

→またプログラミング言語だけではなく、各種のコマンドやスクリプトの生成、設定ファイルの生成、Azure ARMテンプレートやBicep、Terraformといったインフラ定義の生成、テストデータの生成などにも利用できます。

# よくあるご質問

- GitHub Copilotのショートカットキーがたくさんあって覚えるのが難しい  
→ 「チートシート」（ショートカットキーまとめ）を活用しましょう

## GitHub Copilot in VS Code cheat sheet

### Chat with GitHub Copilot

Use natural language to chat with GitHub Copilot and get help with coding tasks. For example, ask Copilot to explain a block of code or a programming concept. Get more information about using [Copilot Chat](#).

Action	Description
<code>Ctrl+Alt+I</code>	Open the <b>Chat</b> view and start a chat conversation with Copilot by using natural language.
<code>Ctrl+Shift+I</code>	Open the <b>Copilot Edits</b> view and start a code editing session across multiple files.

- 一週間くらいGitHub Copilotを使うと指が自然にショートカットを覚えると思います。

# よくあるご質問

- GitHub Copilotのすべての機能は、どのツール（vsCode, Visual Studio, Eclipse, Xcodeなど）でも同じように利用可能ですか？  
→いいえ。
  - ・現在Visual Studio Codeが最も進化が早く、多くの機能がすばやく投入され、すぐに最新機能を利用できます。
  - ・Visual Studio でも多くの機能を利用でき、Visual Studioでしか利用できない「GitHub Copilotによるアプリデバッグ」といった機能もあります。
  - ・その他のツール・IDEについては、「チャットのみ」「補完のみ」など、サポート状況はまちまちで、今後のアップデートにより徐々に改善が進むと思われます。

# よくあるご質問

■ GitHub Copilotは、GitHub Codespaces（GitHubが提供するクラウドベースの開発環境）で使用できますか？

→はい、利用できます。

# よくあるご質問

■ GitHub Copilotは、`vscode.dev`（web版のVisual Studio Code）で使用できますか？

→チャットは利用できませんが、コード補完は利用できます。

# よくあるご質問

- GitHub Enterprise Server（セルフホストバージョンのGitHub）を使用していますが、GitHub Copilotは使用できますか？

→いいえ、GitHub CopilotはGitHub Enterprise Serverの環境では使用できません。

# よくあるご質問

■ GitHub Copilotは、インターネット接続がない環境でも使用できますか？

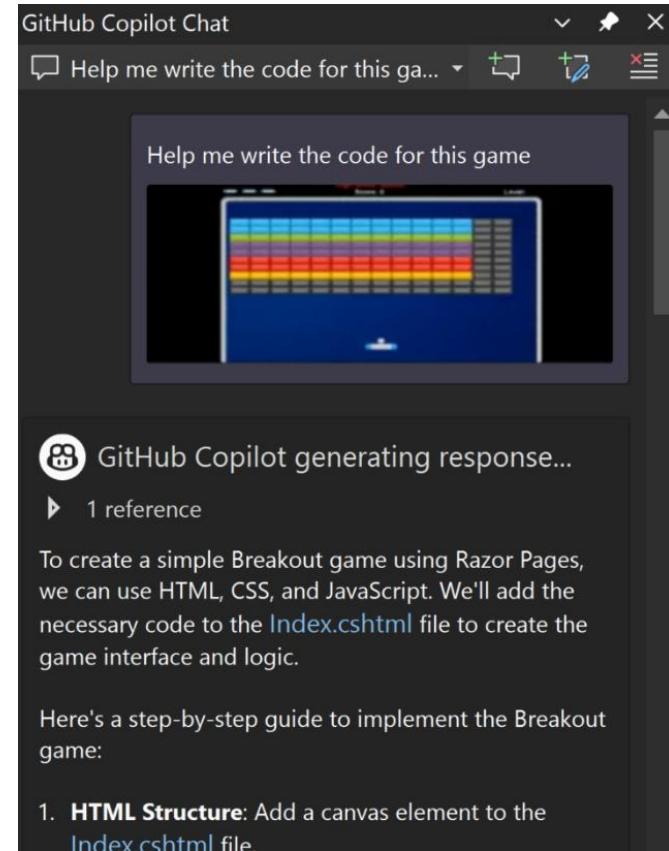
→いいえ、使用できません。インターネット接続（GitHubのサーバーへのHTTPS接続）が必須となります。

HTTPプロキシが利用できる環境では使用できます。

# よくあるご質問

■ GitHub Copilotのチャットに画像を送信できますか？（画像を参考に作業をしてもらう、トラブルシューティングをしてもらうなど）

→はい、画像添付が可能です。



# よくあるご質問

- GitHub Copilotは、音声での入力・出力には対応していますか？

→はい、対応しています。Visual Studio CodeでGitHub Copilotが使える状態にして、さらに「VS Code Speech」拡張機能をセットアップすることにより、マイクから音声入力でGitHub Copilotに指示を出したり、回答を読み上げてもらったりすることが可能です。追加で設定をすることで日本語にも対応できます。

# よくあるご質問

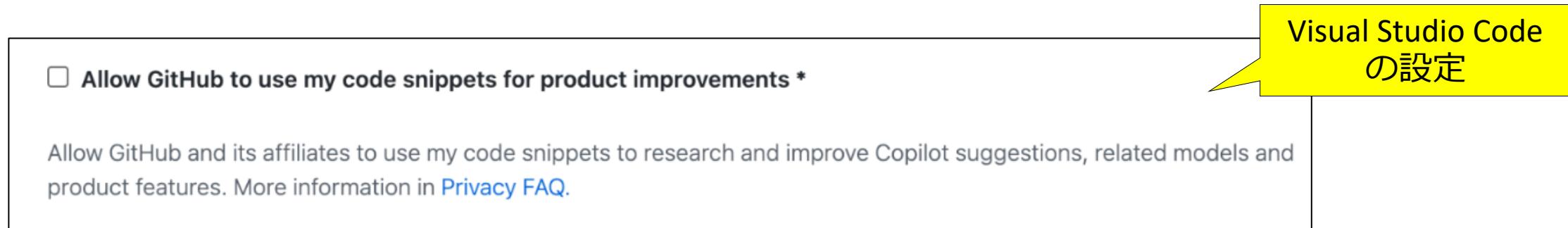
■ GitHub Copilotを使用すると、ユーザーのデータ（コード）はモデルのトレーニングに使用されますか？  
→個人での使用（Free、Proプラン）の場合: ~~設定可能~~ →いいえ

→企業での利用（Business、Enterpriseプラン）の場合: いいえ

**GitHub は、GitHub のモデルをトレーニングするために Copilot Business または Enterprise データを使用しますか？**

いいえ。GitHub はモデルのトレーニングに Copilot Business データも Enterprise データも使用しません。

■ ご参考：個人での使用（Free/Pro/Pro+プラン）の場合、**以前は**、「コードをGitHubで使う許可」が設定可能でした。



現在この設定はなくなり、Business/Enterpriseと同様、**ユーザーのコードがトレーニングに使用されることはありません**。

# よくあるご質問

## ■ GitHub Copilotの応答や生成は英語？日本語？

→ 基本的にはVisual Studio Codeの表示言語設定に従うはずですが、実際動かしてみると必ずしも言語設定に従わない場合があるようです。

.github/copilot-instructions.md で指示したり、  
チャットのプロンプトで「in English」「in Japanese」「日本語で」といったように追加で指示するとよいです。

またVSCodeの設定でチャットの「応答ロケール」を指定することも可能です。



# よくあるご質問

■ GitHub Copilotが提案するコードやチャットの動作はカスタマイズできますか？（規則や指示を追加するなど）

→はい、いくつかの方法でカスタマイズが可能です。

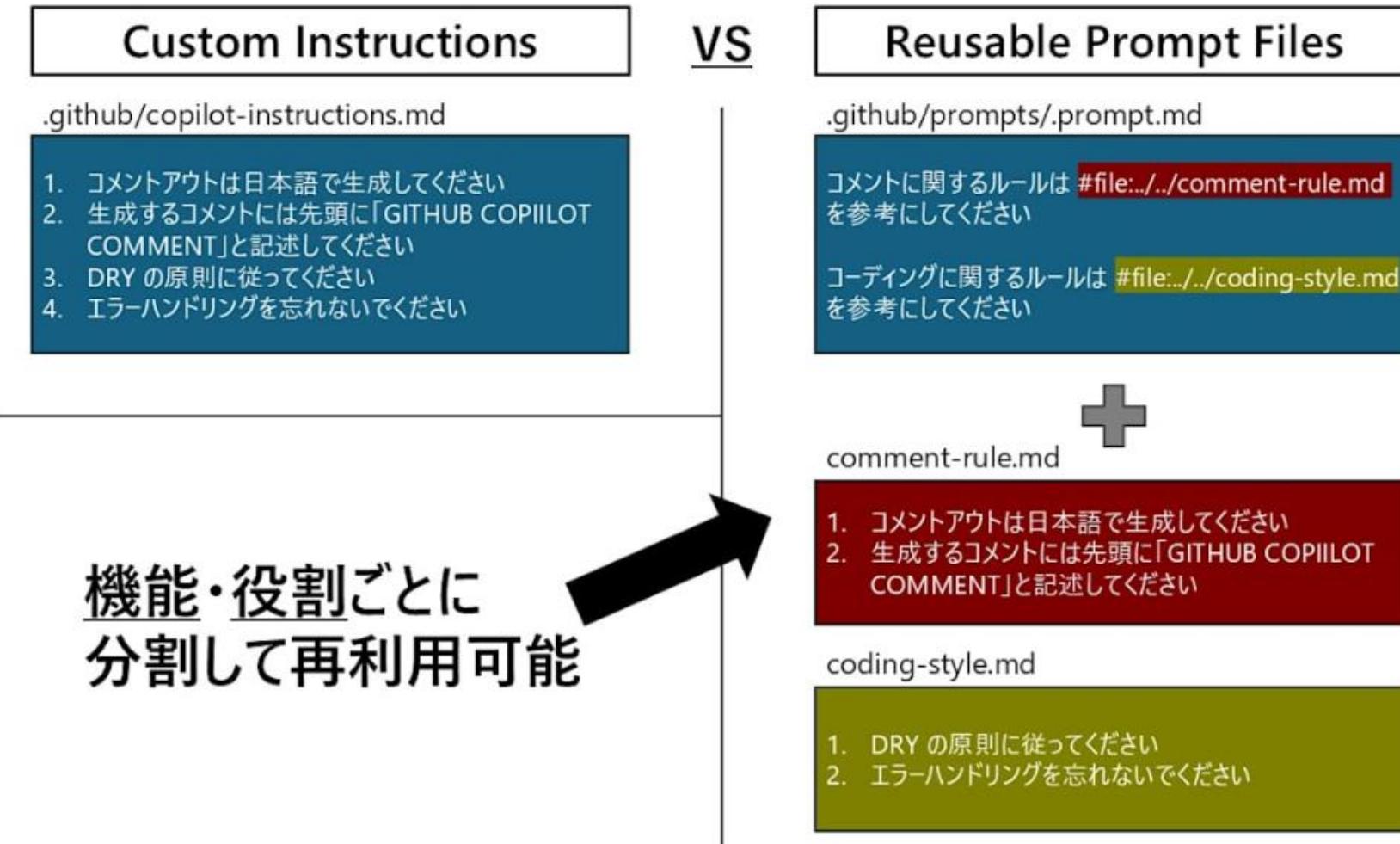
- GitHubリポジトリレベルのカスタマイズ: リポジトリのルートに`.github/copilot-instructions.md`ファイルを作成
- 個人レベルのカスタマイズ: GitHubの画面右上のCopilotアイコンをクリックして「Customize GitHub Copilot > Personal Instructions」から設定
- VSCodeの設定で、コード生成・テスト生成・コードレビュー・コミットメッセージ生成のカスタムインストラクションを設定

[Custom instructions for GitHub Copilot in VS Code](#)

[Copilot のカスタマイズ - GitHub Docs](#)

■参考 「Reusable Prompt Files」により、プロンプトを複数ファイルに分割して定義し、プロンプトファイルを再利用できるようになった。

2025/2/6～



[Vision, agent mode, next edit suggestions, and more for GitHub Copilot in VS Code January release \(v0.24\) - GitHub Changelog](#)

図の出典: カスタムインストラクションの一歩先！GitHub Copilotへの指示を分割管理しよう

# よくあるご質問

## ■ GitHub Copilotのトークン数の制限は？

→選択したプランとモデルによって決まりますが、たとえば入力4000トークン・出力4000トークンなどです。

### 転送率の制限 ☞

プレイグラウンドと無料の API の使用には、1 分あたりの要求数、1 日あたりの要求数、要求あたりのトークン数、同時要求数のレート制限があります。レート制限を受けた場合、さらに要求を行うには、該当したレート制限がリセットされるまで待つ必要があります。

低、高、埋め込みモデルのレート制限は異なります。使用しているモデルの種類を確認するには、GitHub Marketplace のモデルの情報を参照してください。

レート制限レベル	転送率の制限	Copilot Free	Copilot Pro	Copilot Business	Copilot Enterprise
Azure OpenAI o1-mini	1分あたりの要求数	適用なし	2	3	3
	1日あたりの要求数	適用なし	12	15	20
	要求あたりのトークン数	適用なし	4000 入力、 4000 出力	4000 入力、 4000 出力	4000 入力、 4000 出力
同時要求数	適用なし	1	1	1	1

詳しくはこちらの  
ページを参照

■参考 VSCode + GPT-4oで64K / 128K コンテキストウィンドウが使用可能に。

2024/12/6～

- GitHub.com、GitHub Mobile、GitHub CLI、および公式にサポートされている IDE 上の Copilot Chat では、OpenAI GPT-4o を使用する際に **64k トークン ウィンドウ**が利用できるようになりました。
- Visual Studio Code Insiders で GitHub Copilot を使用している場合は、さらに大きな **128k のコンテキスト ウィンドウ**にアクセスできます。これは、OpenAI GPT-4o でサポートされている最大値です。
- ※トークン ウィンドウ = コンテキスト ウィンドウ = 入力トークンと出力トークンの合計値。

# よくあるご質問

## ■ 仕様書に従ってコードを生成させることはできますか？

→はい。以下のような方法でGitHub Copilotに関連ファイル（仕様が書かれたテキストファイルや関連ソースファイルなど）を認識させると、GitHub Copilotはその内容を参考にコードを生成することができます。

- ・関連ファイルをエディターの別タブで開いておく
- ・チャットで #file を使用して関連ファイルを明示的に指定する
- ・クリップアイコンをクリックして関連ファイルを選択する
- ・チャットビューにファイルをドロップする



 GitHub Copilot Chat (Copilot Chat) に添付できるファイルの種類は、主に以下の通りです。

## 1. サポートされているファイル形式

- 画像ファイル
  - PNG (.png)
  - JPEG (.jpg, .jpeg)
  - GIF (.gif)
  - SVG (.svg)
- テキストファイル
  - プレーンテキスト、コードファイル (.txt, .md, .py, .js, .java など)
- ログファイル
  - 一般的なログファイル (.log など)

## 2. 添付できないファイルの例

- 実行形式ファイル (.exe, .bat など)
- アーカイブファイル (.zip, .tar.gz など)
- 特殊なバイナリファイルや大容量ファイル
- セキュリティ上の理由で制限されているファイル

Word/Excel/PowerPoint/PDFは  
アップロードできない。

## 3. 注意点

- ファイルサイズ制限があります (通常数MB程度まで)。

# よくあるご質問

- GitHub Copilotに関する認定試験や資格はありますか？

→はい。GitHub社は、「GitHub Copilot 認定試験」を提供しています。日本語版もあります。99 USD。

※詳しくは本講義の最後に解説

[GitHub Certifications \(GitHub Copilot certification\)](#)

[GitHub Certifications](#)

[GitHub 認定で専門性をアピール - GitHub Resources](#)



# よくあるご質問

- GitHub Copilotが生成したコードの著作権・所有権はどうなりますか？

→GitHub Copilotはただのツールであり、著作権・所有権は主張しません。エディターやIDEを使ってコードを書いた場合と同様に、コードは常にコードを書いた人間に属します。

# よくあるご質問

- GitHub Copilot が生成するコードが、ドキュメントに記載されている例とは少し異なるのですが・・・
- GitHub Copilot の挙動が実行ごとに異なる場合があるのですが・・・

→ はい。そういうことが起こり得ます。GitHub Copilot は**生成AI**を使用しており、**生成AI**の出力は**非決定的 (nondeterministic)**です。つまり、若干のランダム性を持っています。

→ 何かがうまくいかない場合、同じ操作をただ繰り返すか、プロンプトなどを少し変えることで解決できる場合があります。

# よくあるご質問

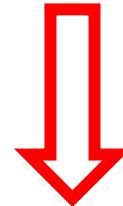
■ GitHub Copilot利用におけるベストプラクティス集はありますか？

→はい。GitHubドキュメント (<https://docs.github.com>) 内に、「ベストプラクティス」や「プロンプトエンジニアリング」（良いプロンプトの書き方）が記載されています。

[Copilot Chat のプロンプトエンジニアリング - GitHub Docs](#)

[GitHub Copilot の使用についてのベストプラクティス - GitHub Docs](#)

その他のご不明点  
(特に企業での利用における懸念点など)  
については、  
「GitHub Copilot Trust Center」に情報がまとまっています。ご確認ください。



[GitHub Copilot Trust Center - GitHub Resources](#)

<https://resources.github.com/ja/copilot-trust-center/>

企業でのGitHub Copilotの導入の推進方法については、GitHubドキュメントの「Copilotの導入を推進する」ページをご確認ください。

The screenshot shows a navigation sidebar on the left with sections like 'GitHub Copilot' and 'Copilot の導入を推進する'. The main content area has a title '会社での Copilot の導入の推進' and text explaining the process for promoting Copilot in an organization.

**会社での Copilot の導入の推進**

Copilot の導入を推進するための効果的な有効化プロセスを計画する方法について説明します。

効果的な有効化プロセスは、organization への Copilot の導入を推進するために不可欠です。このプロセスを organization のニーズと目標に合わせて調整し、チームが Copilot の効果的な使い方を理解するのに役立つよう設計する必要があります。

有効化プロセスは、フィードバックと結果に基づいて進化する可能性があります。プロセスのレビューと更新を定期的に行って、organization のニーズが満たされ続けるようにする必要があります。

GitHub Copilot の有効化プロセスは、次のステージに分ることができます。

- ライセンスの付与
- ユーザーによる環境の設定のサポート
- Copilot の効果的な使用のサポート

# モジュール2 GitHub Copilotの概要

- GitHub とは ?
- GitHub Copilot とは ?
- GitHub Copilot のマスコットキャラクター
- GitHub Copilot は無料で利用可能
- GitHub Copilot の使用を開始する手順
- GitHub Copilot の基本的な動作: 提案
- GitHub Copilot Chat とは ?
- GitHub Copilot ではどのようなことができるのか ?
- GitHub Copilot と連携できる開発ツール
- よくあるご質問
- まとめ

# モジュール2まとめ

- **GitHub Copilot** は「AIペアプログラマ」であり、プログラマのコードの記述作業を支援する。
- Visual Studio Codeなど、さまざまな開発ツールで利用できる。
- 個人で利用するには**GitHubアカウント**が必要。
- 個人での利用の場合、無料でも利用できるが、コード提案が月2000回まで・チャットメッセージが月50回まで、といった制限がある。より上位のサブスクリプションプランに変更が可能。
- 基本的には、**GitHub Copilot** は、コードの続きを提案してくれる。タブキーを押すことで提案を確定できる。
- **GitHub Copilot Chat**では、チャット形式でAIに質問したり、コードの説明や修正案をもらったりできる。

# Part 1 目次

- ・モジュール1 責任あるAI
- ・モジュール2 GitHub Copilotの概要
- ・モジュール3 プロンプトエンジニアリング
- ・モジュール4 高度な機能
- ・モジュール5 さまざまな環境でのGitHub Copilot
- ・モジュール6 管理とカスタマイズ

# プロンプトエンジニアリング

- ・プロンプトエンジニアリングとは？
- ・GitHub Copilot チャットでのプロンプトエンジニアリング例
  - ・テクニック1: まず目標を説明し、それから詳細な要件を与える
  - ・テクニック2: 例を示す
  - ・テクニック3: 複雑なタスクを単純なタスクに分割する
  - ・テクニック4: あいまいさを回避する

# プロンプトエンジニアリングとは？

- ・プロンプトエンジニアリング
  - ・AIに最適な指示（プロンプト）を設計し、望む出力を得る技術
- ・GitHub Copilotでのプロンプトエンジニアリングの重要性:
  - ・GitHub Copilotのチャット（インラインチャットやチャットビュー）では、プロンプトを入力して作業を指示する必要がある
  - ・より具体的で明確なプロンプトを入力することで、精度の高いコード提案や回答を得ることができる

# テクニック1: まず目標を説明し、それから詳細な要件を与える

## よくない例

- 正の整数が入力されたら、素数の場合は `true` を返し、入力が正の整数でない場合はエラーとする関数を実装してください。

## よい例

- 数値が素数かどうかを示す関数を記述してください
- 関数は整数を受け取ります
- 整数が素数の場合は `true` を返します
- 入力が正の整数でない場合、関数はエラーになります

# テクニック2: 例を示す

## よくない例

- 文字列内のすべての日付を検索し、配列で返す関数を記述。

## よい例

- 文字列内のすべての日付を検索し、配列で返す関数を記述。  
例:
  - `findDates("11/14/2023 に歯医者を予約し、 12-1-23 にクラブを予約しています")`
  - 戻り値: `["11/14/2023", "12-1-23"]`

# テクニック3: 複雑なタスクを単純なタスクに分割する

## よくない例

- $10 \times 10$  の文字グリッドを生成し、有効な単語一覧を使ってグリッド内のすべての単語を検索し、少なくとも 10 個の単語を含むグリッドを生成し、さらにそのグリッドと 10 個のランダムな単語を出力する関数をまとめて書いてください。

## よい例

- $10 \times 10$  の文字グリッドを生成する関数を記述します。
- 有効な単語の一覧を指定して、文字グリッド内のすべての単語を検索する関数を記述します。
- 前の関数を使って、少なくとも 10 個の単語を含む  $10 \times 10$  個の文字グリッドを生成する関数を書きます。
- 前の関数を更新して、グリッドから文字のグリッドと 10 個のランダムな単語を出力します。

# テクニック4: 曖昧さを回避する

## よくない例

- この関数ではどのような処理が実行されますか？

## よい例

- CreateUser関数ではどのような処理が実行されますか？

# Part 1 目次

- ・モジュール1 責任あるAI
- ・モジュール2 GitHub Copilotの概要
- ・モジュール3 プロンプトエンジニアリング
- ・モジュール4 高度な機能
- ・モジュール5 さまざまな環境でのGitHub Copilot
- ・モジュール6 管理とカスタマイズ

# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント

# ■ GitHub Copilot **Code Suggestion**: 2022/6/6 一般提供開始

The screenshot shows a code editor window with a dark theme. The file is named 'Class3.cs' and is part of a project 'PrimeService'. The code defines a class 'PrimeService' with a static method 'IsPrime' that checks if a number is prime. A yellow callout box with the text '[Alt + ¥]' points to the cursor position in the code editor, indicating the keyboard shortcut for accepting the suggested code.

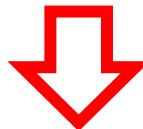
```
1  namespace MyPrimeService
2  {
3  }
4  class PrimeService
5  {
6      public static bool IsPrime(Long number)
7      {
8          if (number <= 1) return false;
9          if (number == 2) return true;
10         if (number % 2 == 0) return false;
11         for (Long i = 3; i <= Math.Sqrt(number); i += 2)
12         {
13             if (number % i == 0) return false;
14         }
15         return true;
16     }
17 }
```

# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント

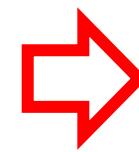
## ■ GitHub Copilot **Next edit suggestions (NES)**: 2025/2/6 プレビュー開始、 2025/3 一般提供開始

```
0 個の参照  
static int Add(int a, int b)  
{  
    return a + b;  
}
```



第3引数 int c を  
手動で追加すると . . .

```
0 個の参照  
💡 static int Add(int a, int b, int c)  
{  
    return a + b;  
}
```



GitHub Copilotがユーザーの  
「次の編集」を予測して提案  
を表示。Tabで確定。

```
4 {  
5  
6 → 0 個の参照  
7 static int Add(int a, int b, int c)  
8 {  
9     return a + b + c;  
}
```

March 2025 (version 1.99)

<https://github.blog/changelog/2025-02-06-next-edit-suggestions-agent-mode-and-prompts-files-for-github-copilot-in-vs-code-january-release-v0-24/>

# コード提案 vs 次の編集提案(NES)

## コード提案

### (code suggestions)

- ・プログラマが次に記述するであろうコードを予測して提案
- ・コードを新規作成する際に使用する
- ・例: コメントから、メソッド(関数)の本体のコードを提案(作成)する

## 次の編集提案

### (Next Edit Suggestions, NES)

- ・プログラマが次に実行するであろうコード編集を予測して提案
- ・コードを編集する際に使用する
- ・例: メソッド(関数)の引数を書き換えると、それに合わせてメソッド本体の修正を提案する

# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント

# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント

# ■ GitHub Copilot Chat: 2023/12/29 一般提供開始

A screenshot of a code editor showing a C# file named Class3.cs. The code defines a namespace `MyPrimeService` and a class `PrimeService`. A cursor is at the start of the opening brace of the class definition. Below the code, there is an inline chat interface with a text input field containing `/generate IsPrime`. The interface includes icons for microphone, speaker, and GPT-4o, along with a dropdown arrow.

```
1 namespace MyPrimeService
2 {
3     0 個の参照
4     class PrimeService
5     {
6
7         }
8     }
9 }
```

オンラインチャット  
[Ctrl + I]

A screenshot of a code editor showing a chat view for a file named sample17. The title bar says "sample17". The main area displays a message from GitHub Copilot: "output carefully before use." followed by "As an internal user, additional telemetry is collected. If you work on a project that contains customer content, you must disable telemetry." Below this, there are three command suggestions: "/fix コード内の問題", "/tests コードの単体テストを追加する", and "/explain 選択したコードのしくみ". A dropdown menu shows "Class3.cs Current file" and "@workspace add IsPrime method". The bottom status bar shows "行 7, 列 1 スペース: 4 UTF-8 CRLF {} C# ⚡".

チャットビュー  
[Ctrl + Alt + I]

<https://github.blog/news-insights/product-news/github-copilot-chat-now-generally-available-for-organizations-and-individuals/>

# GitHub Copilotの高度な機能

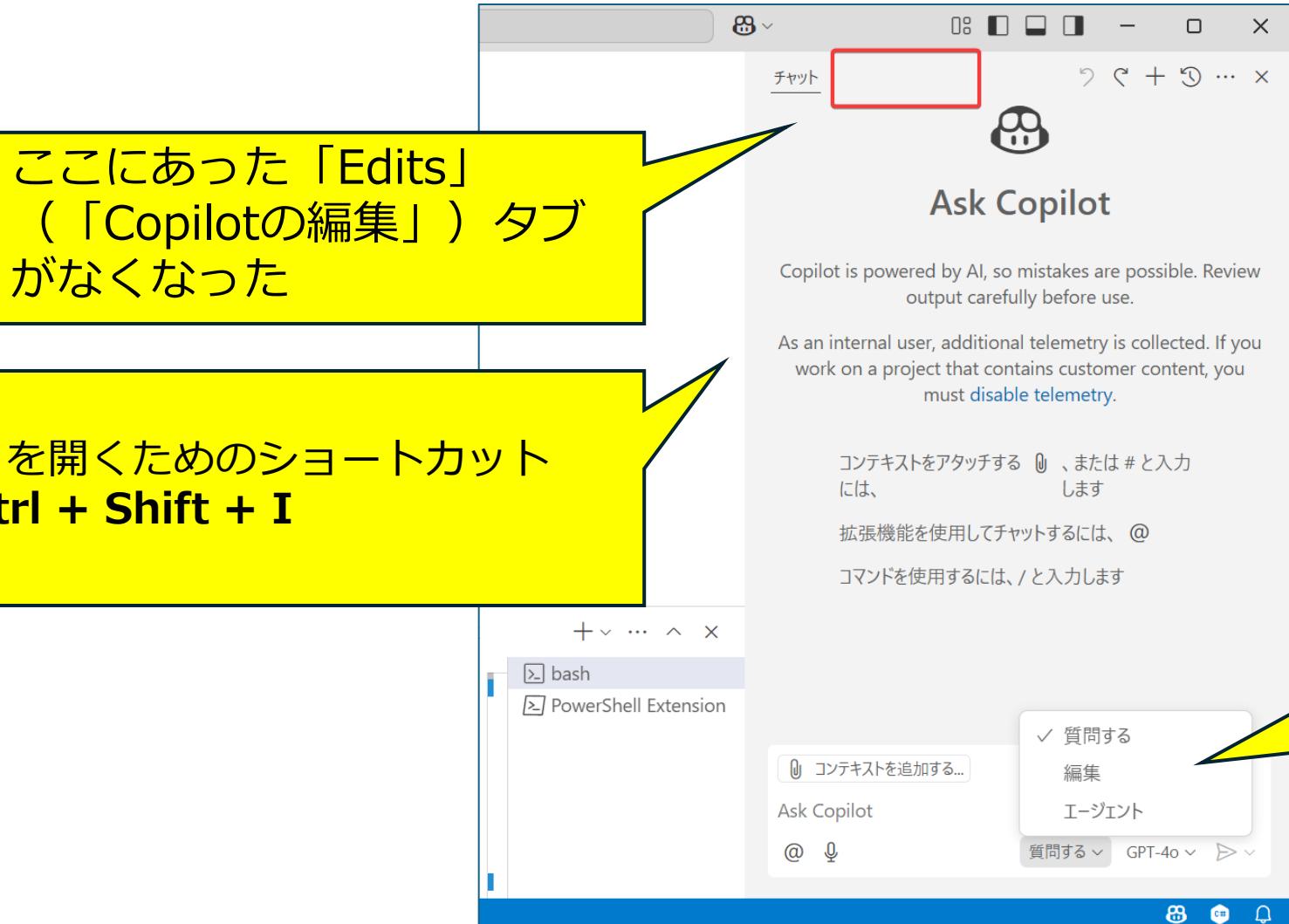
- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント

## ■ GitHub Copilot Edits: 2025/2/6 一般提供開始



<https://github.blog/news-insights/product-news/github-copilot-the-agent-awakens/#copilot-edits-now-ga-in-vs-code-%f0%9f%8e%89>

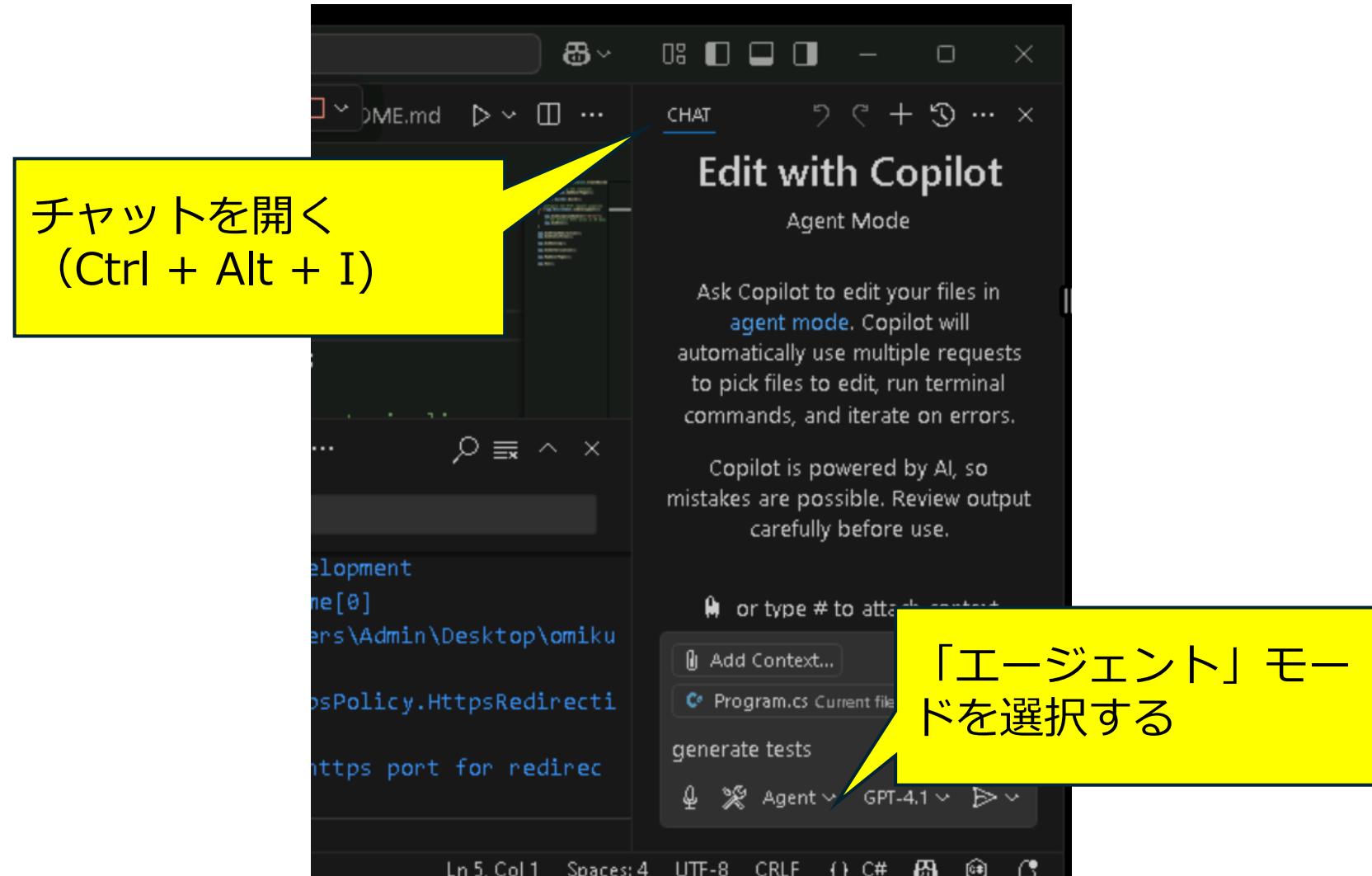
# (2025/4 アップデート) 「Edits」 タブがなくなった。



# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント

■ GitHub Copilot **agent mode**: 2025/2/6 プレビュー開始。  
Editsの機能に加え、**コマンドの提案・実行**ができるようになっている。



<https://github.blog/changelog/2025-02-06-next-edit-suggestions-agent-mode-and-prompts-files-for-github-copilot-in-vs-code-january-release-v0-24/>

■GitHub Copilot **agent mode**で、プロンプトを入力。かなり複雑な手順を指示できる。

The screenshot shows the GitHub Copilot Agent Mode interface. At the top left is a button labeled "Add Files...". The main area contains three paragraphs of Japanese text:

C#のソリューションを作成し、ソリューションに PrimeService クラスライブラリプロジェクトと PrimeService.Tests xunit テストプロジェクトを追加してください。  
PrimeService.Tests は PrimeService プロジェクトを参照します。PrimeService プロジェクトに、入力された整数が素数かどうかを判定する IsPrime メソッドを作成します。  
PrimeService.Tests プロジェクトに、IsPrime メソッドの単体テストを記述します。単体テストでは1から15までの整数をテストしてください。最後に dotnet tests を実行して単体テストを実行し、テストが成功することを確認してください。

At the bottom right of the main panel are three buttons: "エージェント" (Agent), "GPT-4o", and a right-pointing arrow. Below the main panel is a dark footer bar with a bell icon and a user profile icon.

## ■ GitHub Copilot agent mode

The screenshot shows the GitHub Copilot agent mode interface. At the top, there are tabs for "チャット" and "COPilot の編集". On the right side, there are icons for refresh, settings, plus, and close. Below the tabs, there is a small profile picture of a cat.

A red box highlights a block of Japanese text from the AI's response:

C#のソリューションを作成し、ソリューションに PrimeService クラスライブラリプロジェクトと PrimeService.Tests xunit テストプロジェクトを追加してください。PrimeService.Tests は PrimeService プロジェクトを参照します。PrimeService プロジェクトに、入力された整数が素数かどうかを判定する IsPrime メソッドを作成します。PrimeService.Tests プロジェクトに、IsPrime メソッドの単体テストを記述します。単体テストでは1から15までの整数をテストしてください。最後に dotnet tests を実行して単体テストを実行し、テストが成功することを確認してください。

Below this, the GitHub Copilot logo is shown with the text "GitHub Copilot". A yellow callout box points to the "dotnet new sln -n SampleSolution" command in the terminal window, containing the text "必要なコマンドを提案してくる！".

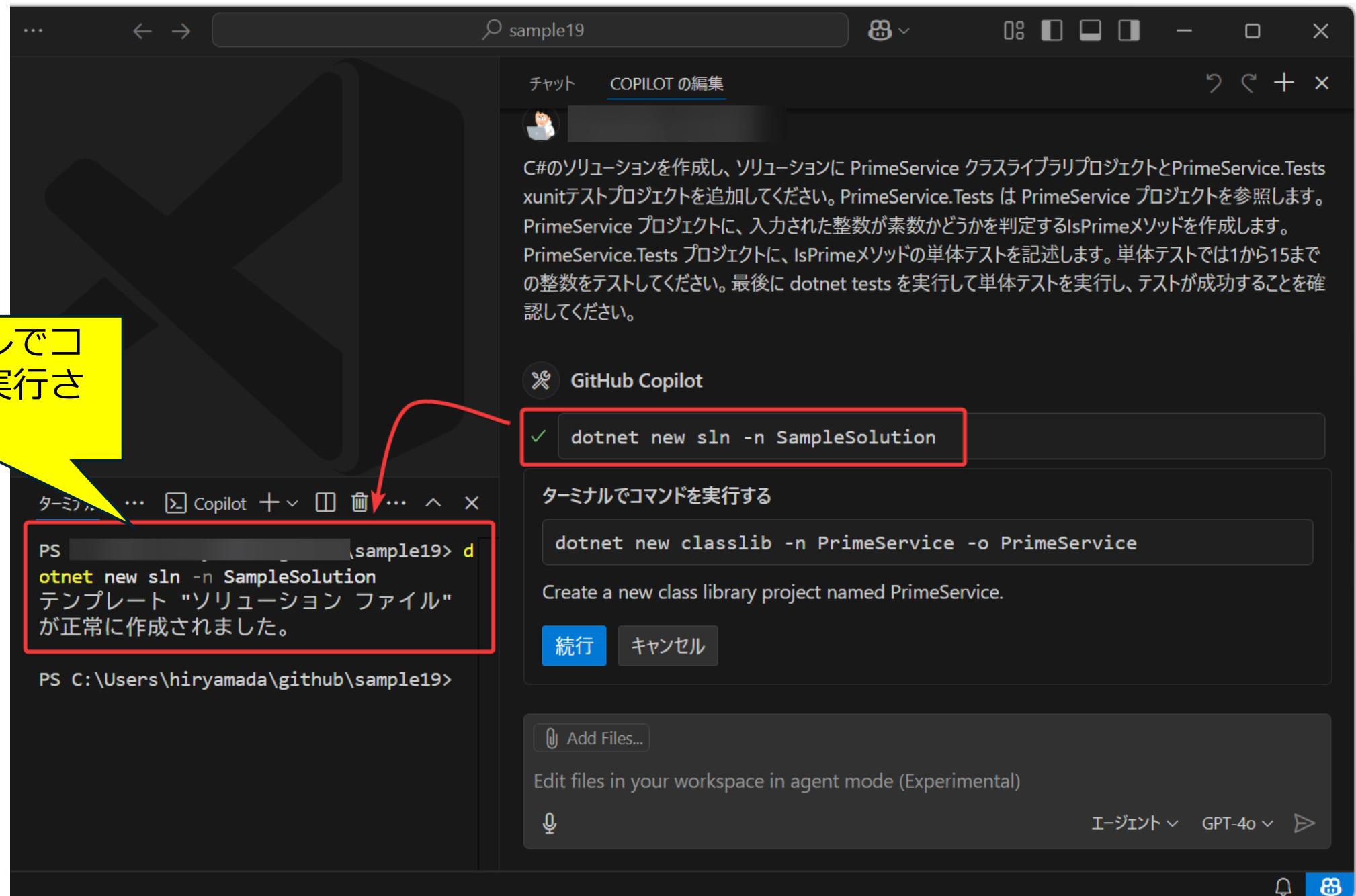
In the terminal window, there is a message: "ターミナルでコマンドを実行する" followed by the command "dotnet new sln -n SampleSolution". Below the command, it says "Create a new C# solution named SampleSolution." There are two buttons: a blue "続行" button with a red border and a grey "キャンセル" button.

A yellow callout box on the left points to the "続行" button with the text "「続行」をクリックして承認".

At the bottom of the interface, there is a "Add Files..." button, a message "Edit files in your workspace in agent mode (Experimental)", and a status bar with "エージェント" and "GPT-4o" dropdowns, along with a right-pointing arrow icon.

## ■ GitHub Copilot agent mode

ターミナルでコマンドが実行された！



## ■ GitHub Copilot agent mode

The screenshot shows the GitHub Copilot agent mode interface. At the top, there's a search bar with the text "sample19". Below it, a sidebar has "チャット" and "COPilot の編集" tabs, with "COPilot の編集" currently selected. A message from GitHub Copilot suggests creating a C# solution with a PrimeService class library and a PrimeService.Tests unit test project. A yellow callout points to this message with the text "さらに次のコマンドを提案してくる！". In the terminal below, the command "dotnet new sln -n SampleSolution" is shown as a suggestion. A yellow callout points to this terminal area with the text "'続行'をクリックして承認". A modal window titled "ターミナルでコマンドを実行する" contains the command "dotnet new classlib -n PrimeService -o PrimeService" and its description "Create a new class library project named PrimeService.". It includes "続行" and "キャンセル" buttons, with "続行" highlighted by a red box. The bottom of the screen shows a workspace for editing files in agent mode, with options like "Add Files...", "Edit files in your workspace in agent mode (Experimental)", and "エージェント".

sample19

チャット COPilot の編集

C#のソリューションを作成し、ソリューションに PrimeService クラスライブラリプロジェクトとPrimeService.Tests xunitテストプロジェクトを追加してください。PrimeService.Tests は PrimeService プロジェクトを参照します。PrimeService プロジェクトに、入力された整数が素数かどうかを判定するIsPrimeメソッドを作成します。PrimeService.Tests プロジェクトに、IsPrimeメソッドの単体テストを記述します。単体テストでは1から15までの整数をテストしてください。最後に dotnet tests を実行して単体テストを実行し、テストが成功することを確認してください。

GitHub Copilot

✓ dotnet new sln -n SampleSolution

ターミナルでコマンドを実行する

dotnet new classlib -n PrimeService -o PrimeService

Create a new class library project named PrimeService.

続行 キャンセル

「続行」を  
クリックして  
承認

Add Files...

Edit files in your workspace in agent mode (Experimental)

エージェント GPT-4o ▶

# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント



ファイル(F) 編集(E) 選択(S) 表示(V) ...

← →



def binary\_search(arr, target): Untitled-1 ●

```
1 def binary_search(arr, target):
2     left, right = 0, len(arr) - 1
3
4     while left <= right:
5         mid = (left + right) // 2
6
7         if arr[mid] == target:
8             return mid
9         elif arr[mid] < target:
10            left = mid + 1
11        else:
12            right = mid - 1
13
14    return -1
15
```





...



```
def binary_search(arr, target):
    def binary_search(left, right = 0):
        while left <= right:
            mid = (left + right) // 2
            if arr[mid] == target:
                return mid
            elif arr[mid] < target:
                left = mid + 1
            else:
                right = mid - 1
        return -1
```

- 定義へ移動 F12
  - 宣言へ移動
  - 型定義へ移動
  - 実装へ移動 Ctrl+F12
  - 参照へ移動 Shift+F12
  - ピーク >
  - すべての参照を検索 Shift+Alt+F12
  - すべての実装の検索
  - 呼び出し階層の表示 Shift+Alt+H
  - 型階層の表示
  - 選択内容をチャットに追加
  - ファイルをチャットに追加
- Copilot >**
- シンボルの名前変更 F2
  - すべての出現箇所を変更 Ctrl+F2
  - リファクター... Ctrl+Shift+R
  - ソース アクション...
  - 切り取り Ctrl+X
  - コピー Ctrl+C
  - 貼り付け Ctrl+V
  - インタラクティブ ウィンドウで実行します >
  - Python の実行 >

- エディター内のインライン チャット Ctrl+I
- 説明
  - 修正
  - レビューとコメント >**
  - ドキュメントを生成する
  - テストを生成する



1 def binary\_search(arr, target): Untitled-1 ●

1 def binary\_search(arr, target):

コード レビュー コメント (1 / 2)

↑ ↓ ⌂ ⌂ ⌂ ⌂ ⌂

**GitHub Copilot**関数 `binary_search` の説明がありません。引数や返り値について簡単なドキュメントを追加してください。

提案する変更:

```
+     """  
+     Perform binary search on a sorted list to find the target value.  
+  
+     Args:  
+         arr (list): A sorted list of elements to search.  
+         target: The value to search for in the list.  
+  
+     Returns:  
+         int: The index of the target if found, otherwise -1.  
+     """
```

適用して次へ移動

破棄して次へ移動

```
2     left, right = 0, len(arr) - 1  
3  
4     while left <= right:  
5         mid = (left + right) // 2  
6  
7         if arr[mid] == target:  
8             return mid  
9         elif arr[mid] < target:
```



```
1 def binary_search(arr, target): Untitled-1 ●  
2     """  
3         Perform binary search on a sorted list to find the target value.  
4     """  
5     Args:  
6         arr (list): A sorted list of elements to search.  
7         target: The value to search for in the list.  
8     Returns:  
9         int: The index of the target if found, otherwise -1.  
10    """  
11    left, right = 0, len(arr) - 1  
12  
13    while left <= right:  
14        mid = (left + right) // 2  
15  
16        if arr[mid] == target:  
17            return mid  
18        elif arr[mid] < target:  
19            left = mid + 1  
20        else:  
21            right = mid - 1  
22  
23    return -1
```

コードレビューコメント (1 / 1)

↑ ↓ ← → ⌂ ⌃ ⌄ ⌅



GitHub Copilot



ターゲットが見つからない場合に -1 を返すことは一般的ですが、その挙動を関数のドキュメントに明記してください。

提案する変更:

+ """

1 def binary\_search(arr, target): Untitled-1 ●

コードレビューコメント (1 / 1)

GitHub Copilot

ターゲットが見つからない場合に -1 を返すことは一般的ですが、その挙動を関数のドキュメントに明記してください。

提案する変更:

```
+     """  
+     Performs binary search on a sorted list.  
+  
+     Args:  
+         arr (list): The sorted list to search.  
+         target: The value to search for.  
+  
+     Returns:  
+         int: The index of the target if found; -1 if not found.  
+     """
```

適用 破棄 |

25

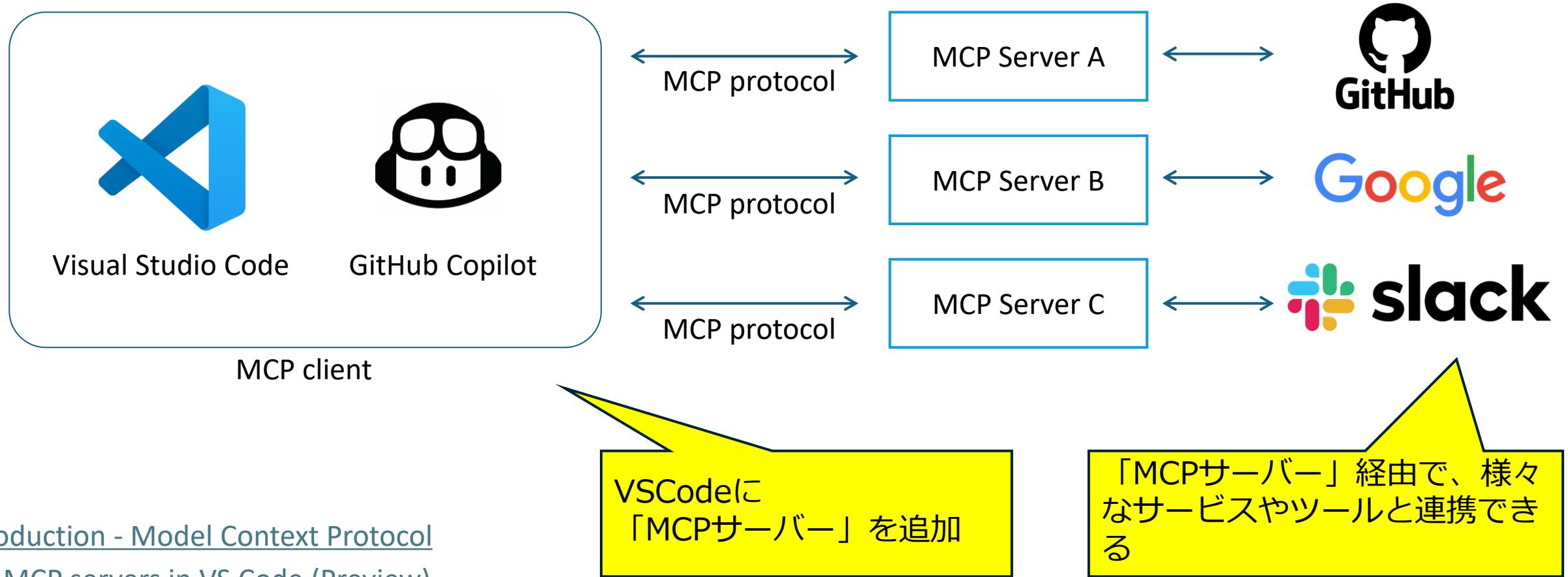
行 24、列 14 (600 個選択) スペース: 4 UTF-8 CRLF {} Python 3.13.7 64-bit No Environment Go Live default

# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント

# VSCodeが「MCPサーバー」に対応、 agent modeから利用可能に！(2025/4)

- 現在プレビュー中



# C#でMCPサーバーを実装する例

- dotnet new worker
- dotnet add package ModelContextProtocol --prerelease
- Program.csにてMCPサーバーツールを実装

```
[McpServerToolType]
public static class TimeTool
{
    [McpServerTool, Description("Get current time")]
    public static string GetcurrentTime(string message) => $"{DateTime.Now}";
}
```

- dotnet release

# PythonでMCPサーバーを実装する例



The screenshot shows the Microsoft Visual Studio Code (VS Code) interface. The title bar includes the VS Code logo, file paths, and standard menu items: ファイル(F), 編集(E), 選択(S), 表示(V), and three dots. Below the title bar is a toolbar with back and forward arrows. The left sidebar contains icons for file operations like new file, search, and refresh, with a count of 1 for the file list. The main editor area displays a Python script titled 'Untitled-1'. The code uses color-coded syntax highlighting:

```
1  from mcp.server.fastmcp import FastMCP
2  import time
3
4
5  # Create an MCP server
6  mcp = FastMCP("Demo")
7
8  # a tool that return current time as a string
9  @mcp.tool()
10 def current_time() -> str:
11     """Return the current time as a string"""
12     return time.ctime()
13
14
```

# VSCodeにMCPサーバーを追加

.vscode/mcp.json  
ファイルを作成

F1, 「MCPサーバーの追加」コマンドで  
サーバーを追加 (MCPサーバーの実行可  
能形式ファイルのパスを指定)

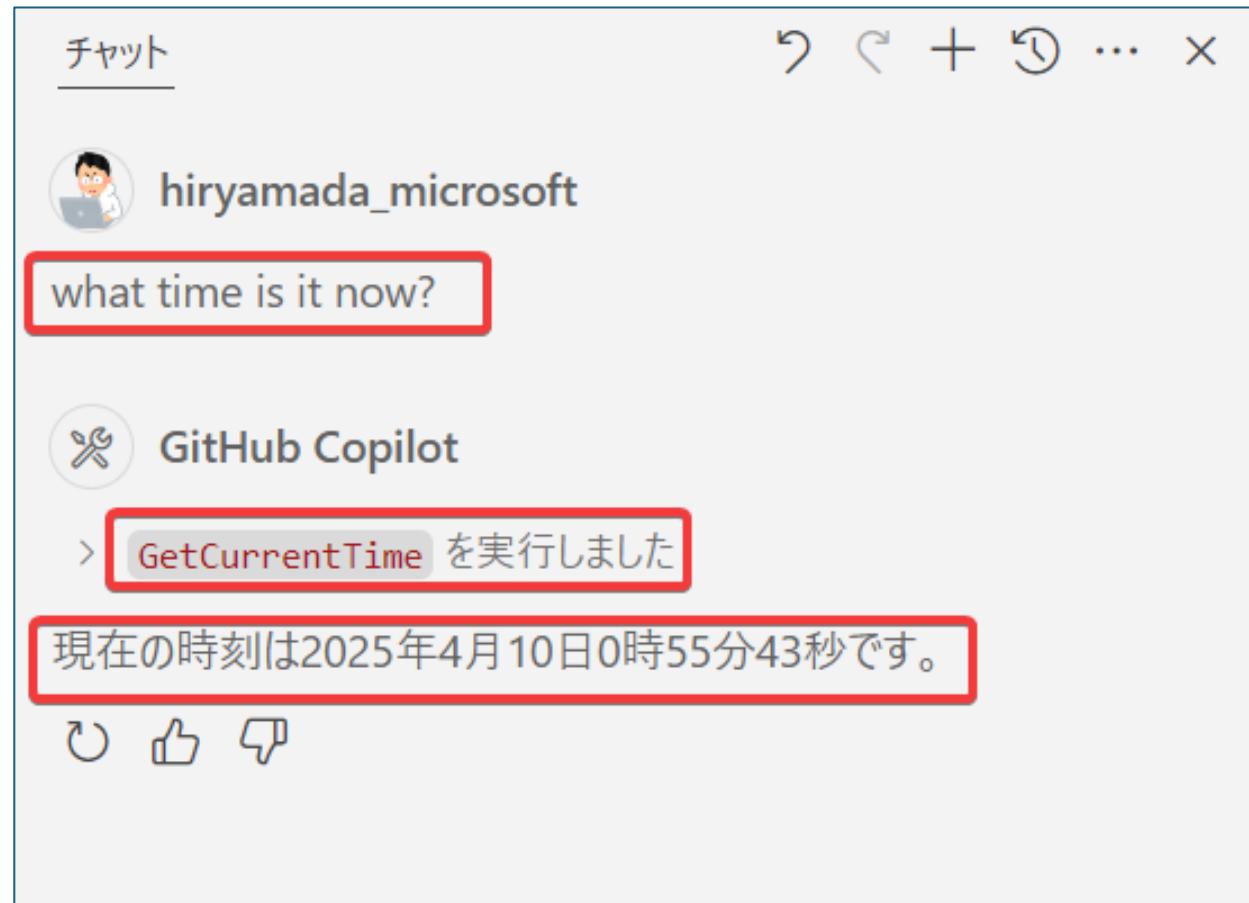
The screenshot shows the VS Code interface with a yellow callout box on the left pointing to the .vscode folder in the Explorer sidebar, containing a mcp.json file. A red box highlights the mcp.json file. Another yellow callout box on the right points to the mcp.json file in the Editor pane, which contains the JSON configuration code shown below.

```
{
  "servers": {
    "my-mcp-server-cc387ad4": {
      "type": "stdio",
      "command": "C:\\work\\proj2025-0409-2231\\bin\\Release\\net9.0\\publish\\mcp.exe",
      "args": []
    }
  }
}
```

# MCPサーバーを起動

```
{  
  "servers": {  
    ▷起動 |1 個のキャッシュされたツール  
    "my-mcp-server-cc387ad4": {  
      "type": "stdio",  
      "command": "C:\\work\\proj2025-0409-2231\\\"  
      "args": []  
    }  
  }  
}
```

チャットビュー（Ctrl + Shift + I）を起動、  
agent modeのチャットから、サーバーの  
機能を利用できる



# GitHub Copilotの高度な機能

- GitHub Copilot Code Suggestion
  - Next edit suggestions (NES)
- GitHub Copilot Chat
  - Askモード
  - Editモード
  - Agentモード
- GitHub Copilotコードレビュー
- MCPサーバー
- コーディングエージェント



GitHub Copilot Coding Agent (Project Padawan) は、GitHubが開発した次世代のAIコーディング支援ツールで、開発者がより高度なタスクを自律的にこなせるように設計されています。従来のGitHub Copilotが「ペアプログラマー」のようにコード補完を行うのに対し、Padawanは「エージェント」として、指示に基づいて複数ステップのタスクを実行できるのが特徴です。

## 「Padawan」という名前の由来

「Padawan」は *Star Wars* に登場する ジェダイ見習い（弟子） を意味します。GitHub はこの名前を、まだ学習中だが将来的に強力な力を持つ存在というコンセプトで採用しました。つまり、Copilot エージェントはまだ「見習い」段階だが、開発者を支援しながら成長することを示唆しています。

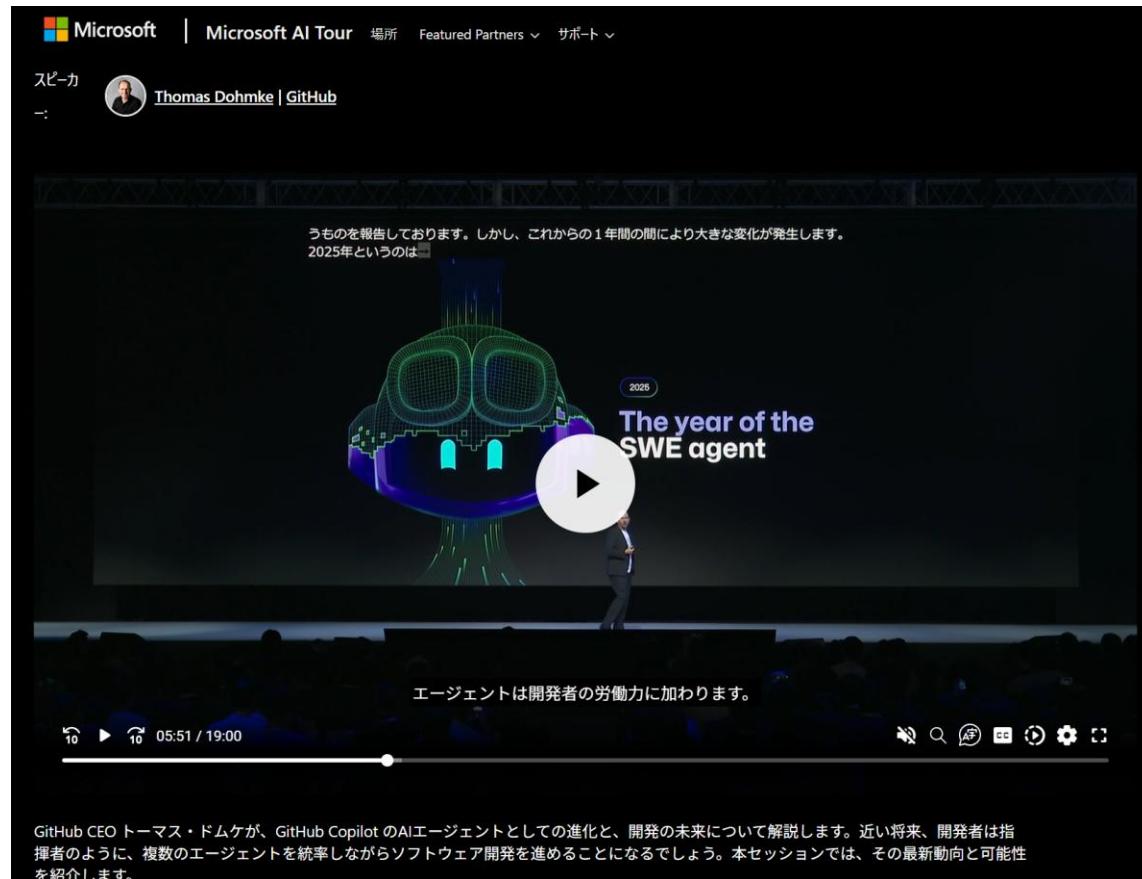
the-decoder

2025/3/27 Microsoft AI Tour 2025 にGitHub CEO トマス・ドムケが登壇。  
「AIピアプログラマー」（Project Padawan）をデモ。

これが、のちに 「**GitHub Copilot Coding Agent**」 となつた。

[GitHub Copilotの新時代：エージェントによる変革 \( GitHub Copilot's Agentic Evolution \)](#)

基調講演動画

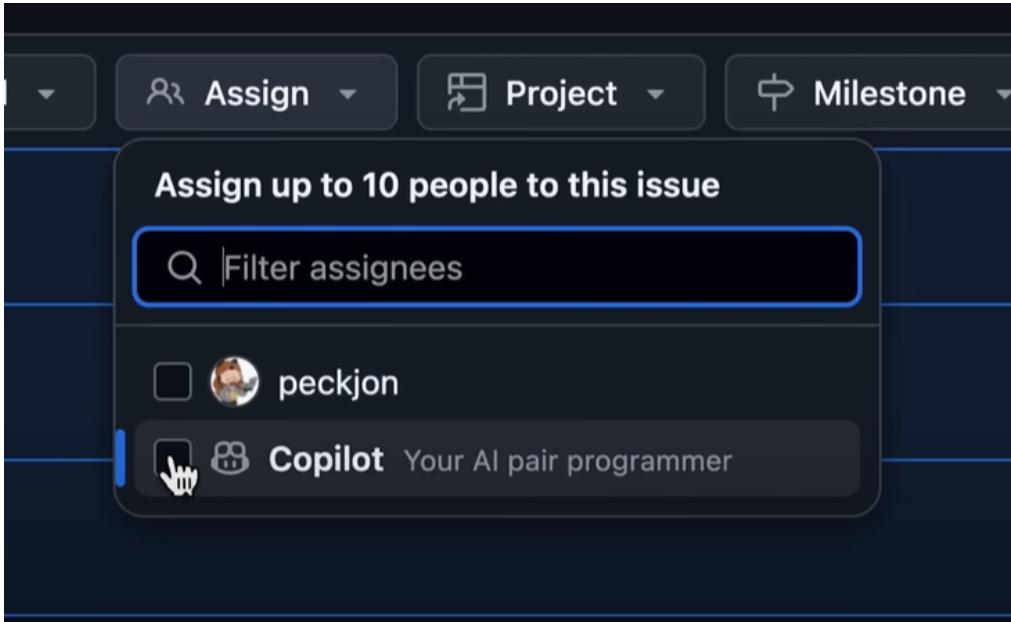


近い将来、開発者は指揮者のように、複数のエージェントを統率しながらソフトウェア開発を進めることになる！

SWE（ソフトウェアエンジニアリング）エージェントの時代がやってくる！

SWEエージェントは「AIピアプログラマー」だ！（ピア peer=同僚）

2025/5/19 Microsoft Build 2025 にて、Project Padawan改め  
「GitHub Copilot Coding Agent」としてプレビュー開始！



GitHub Issuesでイシューを作り、  
担当者（Assignee）として「Copilot」  
を選ぶ！

[WIP] Migrate ORM to Tortoise #19

Draft Copilot wants to merge 1 commit into main from copilot/fix-14

Conversation 0 Commits 1 Checks 2 Files changed 0

Copilot AI commented 4 minutes ago · edited by peckjon

- Update dependencies in requirements.txt
- Create new BaseModel in models/base.py
- Update models/init.py to use Tortoise ORM initialization
- Update Category model to use Tortoise ORM
- Update Publisher model to use Tortoise ORM
- Update Game model to use Tortoise ORM
- Update utils/database.py to support Tortoise ORM
- Update app.py to use the new initialization
- Update routes/games.py to use Tortoise ORM query syntax
- Test the application to verify all functionality works

Fixes #14.

「Copilot」がIssueを解決し、プルリクエストを作成してくれる！

[GitHub Copilotコーディングエージェントのご紹介](#)

[GitHub Copilot: Meet the new coding agent - The GitHub Blog](#)

# Part 1 目次

- ・モジュール1 責任あるAI
- ・モジュール2 GitHub Copilotの概要
- ・モジュール3 プロンプトエンジニアリング
- ・モジュール4 高度な機能
- ・モジュール5 さまざまな環境でのGitHub Copilot
- ・モジュール6 管理とカスタマイズ

# さまざまな環境でのGitHub Copilot

- IDE（統合開発環境）での利用
- GitHub Mobileでのチャットの利用
- GitHub ダッシュボードでのチャットの利用
- GitHub デスクトップアプリでの利用
- GitHub CLIでの利用
- GitHub Codespacesでの利用

# さまざまな環境でのGitHub Copilot

- IDE（統合開発環境）での利用
- GitHub Mobileでのチャットの利用
- GitHub ダッシュボードでのチャットの利用
- GitHub デスクトップアプリでの利用
- GitHub CLIでの利用
- GitHub Codespacesでの利用

# Visual Studio でもGitHub Copilotが利用可能

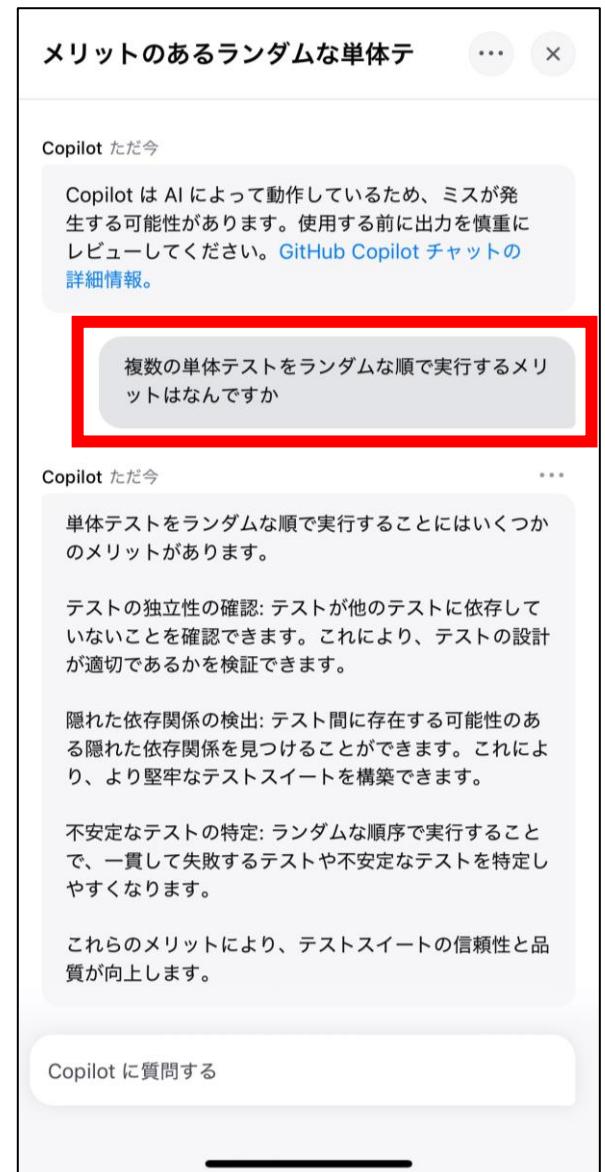
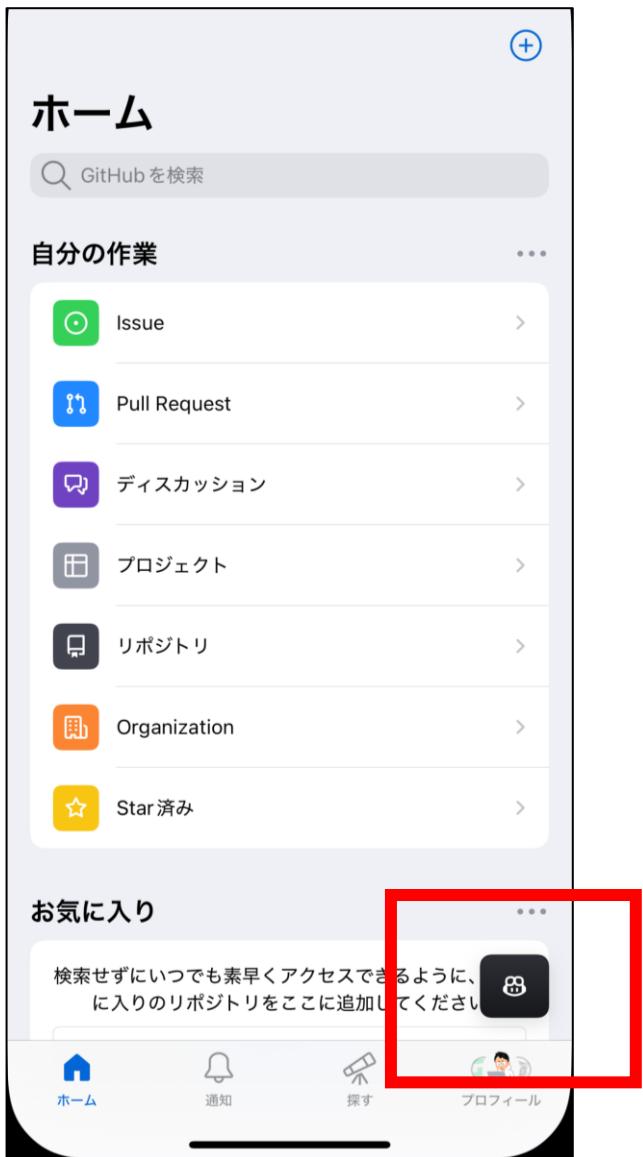
The screenshot shows a Visual Studio 2022 interface with the following details:

- File Explorer:** Shows files like BasketService.cs, Program.cs, Basket.cs, and What's New.
- Solution Explorer:** Shows projects like FabrikamProduct.BikeSharing.Web and BikeSharing.Web.Controllers.HelpController.
- Toolbars:** Standard Visual Studio toolbars for file operations, search, and navigation.
- Status Bar:** Displays "0:13 / 0:37".
- GitHub Copilot Chat:** A sidebar titled "GitHub Copilot Chat" with a message: "Add a wishlist function to the basket".
- Copilot Summary:** A section titled "Copilot" with the following status:
  - Edits generated
  - Found 3 result(s)
  - #currentfile
- Changes Summary:** A section titled "Here's a summary of the changes:" listing 6 items:
  - Create a new Wishlist entity to represent the wishlist.
  - Add methods to the BasketService to add and remove items from the wishlist.
  - Updated the Basket entity to include a wishlist property.
  - Update the BasketService to handle the wishlist operations.
  - Create a new interface IWishlistService to define the wishlist operations.
  - Implement the IWishlistService in a new WishlistService class.
- Accept Changes:** A list titled "Iteration 1" with four items:
  - BasketService.cs
  - Basket.cs (highlighted with a cursor)
  - IWishlistService.cs
  - WishlistService.cs
- Feedback:** A section at the bottom right asking "Was this helpful?" with a thumbs up and thumbs down icon.
- Bottom Bar:** Includes "Ask Copilot: Type / for commands or # to reference code" and icons for Gemini 1, Gemini 2, and Gemini 3.

# さまざまな環境でのGitHub Copilot

- IDE（統合開発環境）での利用
- GitHub Mobileでのチャットの利用
- GitHub ダッシュボードでのチャットの利用
- GitHub デスクトップアプリでの利用
- GitHub CLIでの利用
- GitHub Codespacesでの利用

# ■GitHubモバイルアプリでGitHub Copilotのチャットを利用する例



# さまざまな環境でのGitHub Copilot

- IDE（統合開発環境）での利用
- GitHub Mobileでのチャットの利用
- GitHub ダッシュボードでのチャットの利用
- GitHub デスクトップアプリでの利用
- GitHub CLIでの利用
- GitHub Codespacesでの利用

## ■GitHubのトップページ（ダッシュボード）から利用する例

The screenshot shows the GitHub Dashboard. On the left, there's a sidebar with 'Top repositories' and a 'New' button. Below it is a search bar with 'Find a repository...'. The main area is titled 'Home' and features a prominent 'Ask Copilot' button, which is highlighted with a red rectangular box. Below this are other cards: 'Find issues assigned to me' and 'Python password endpoint'. At the bottom, there's a card for the 'microsoft/semantic-kernel' repository, which was released 5 days ago and includes a 'dotnet-1.34.0' update. A 'Changes:' section is also visible.

## ■GitHubのトップページ（ダッシュボード）から利用する例

The screenshot shows the GitHub Dashboard at <https://github.com>. On the left, there's a sidebar with 'Top repositories' and a 'New' button. Below it is a search bar with 'Find a repository...'. The main area is titled 'Home' and features a prominent red-bordered box containing the Japanese text 'GitHub Copilotはどのような開発ツールから使えますか？'. To the right of this box is a red-bordered '▶' button. Below the box, the text 'Copilot uses AI. Check for mistakes.' is visible. A repository card for 'microsoft/semantic-kernel' is shown, released 5 days ago. The word 'dotnet-1.34.0' is displayed below the repository card. At the bottom, the word 'Changes:' is partially visible.

GitHub Copilotはどのような開発ツールから使えますか？

▶

Copilot uses AI. Check for mistakes.

Home

Filter

microsoft/semantic-kernel released  
5 days ago

dotnet-1.34.0

Changes:

## ■GitHubのトップページ（ダッシュボード）から利用する例

The screenshot shows the GitHub Copilot interface. At the top, there's a navigation bar with icons for three, GitHub logo, Copilot, search, and other options. Below it is a toolbar with icons for refresh, edit, and more. The main area displays a question "GitHub Copilotはどのような開発ツールから使えますか？" (What development tools can GitHub Copilot be used from?) followed by a detailed answer.

**GitHub Copilotは以下の開発ツールから使用できます：**

- 1. Visual Studio Code (VS Code):**
  - GitHub CopilotはVisual Studio Codeの拡張機能として提供されています。インストール後、コーディング中に自動補完やコード提案を受けることができます。
- 2. Visual Studio:**
  - Visual Studio 2022以降のバージョンでGitHub Copilotを使用することができます。VS Codeと同様に、コード補完や提案機能を利用できます。
- 3. JetBrains製品:**
  - IntelliJ IDEA、PyCharm、WebStormなどのJetBrains製品でもGitHub Copilotを使用することができます。JetBrains Marketplaceからプラグインをインストールすることで利用可能です。

これらのツールを使用して、効率的なコーディング体験を得ることができます。

# さまざまな環境でのGitHub Copilot

- IDE（統合開発環境）での利用
- GitHub Mobileでのチャットの利用
- GitHub ダッシュボードでのチャットの利用
- GitHub デスクトップアプリでの利用
- GitHub CLIでの利用
- GitHub Codespacesでの利用

File Edit View Repository Branch Help

Current repository desktop-tutorial

Current branch main

Fetch origin Last fetched 4 minutes ago

Changes 1 History README.md

Filter

1 changed file

README.md

@@ -1,5 +1,7 @@  
- # Welcome to GitHub Desktop!  
+ # GitHub Desktopへようこそ！  
- This is your README. READMEs are where you can communicate what your  
project is and how to use it.  
+ これはあなたのREADMEです。READMEでは、プロジェクトの内容や使い方を伝える  
ことができます。  
- Write your name on line 6, save it, and then head back to GitHub Des  
ktop.  
+ 6行目にあなたの名前を書き込み、保存してからGitHub Desktopに戻ってください。  
+  
+ Hiromichi Yamada

Get started

Install a text editor

Create a branch

A branch allows you to work on different versions of a repository at one time. Create a branch by going into the branch menu in the top bar and clicking "New branch".  
Ctrl + Shift + N

Edit a file

Make a commit

Publish to GitHub

Open a pull request

Update README with Japanese translation and name

Translated the README content to Japanese and added 'Hiromichi Yamada' as instructed on line 6.

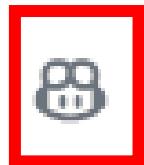
Commit 1 file to main

Exit tutorial



Update README with Japanese translation and name

Translated the README content to Japanese and added  
'Hiromichi Yamada' as instructed on line 6.



Commit 1 file to **main**



Update README with Japanese translation and name

Translated the README content to Japanese and added  
'Hiromichi Yamada' as instructed on line 6.



Commit 1 file to **main**

# さまざまな環境でのGitHub Copilot

- IDE（統合開発環境）での利用
- GitHub Mobileでのチャットの利用
- GitHub ダッシュボードでのチャットの利用
- GitHub デスクトップアプリでの利用
- GitHub CLIでの利用
- GitHub Codespacesでの利用



ファイル(F) 編集(E) 選択(S) 表示(V) ...

← →

検索 [管理者] test1

問題 出力 デバッグコンソール ターミナル ポート

● PS C:\Users\azureuser\Documents\GitHub\test1> **gh copilot explain "sudo apt-get"**

Welcome to GitHub Copilot in the CLI!

version 1.1.1 (2025-06-17)

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code or suggestions. For more information, see <https://gh.io/gh-copilot-transparency>

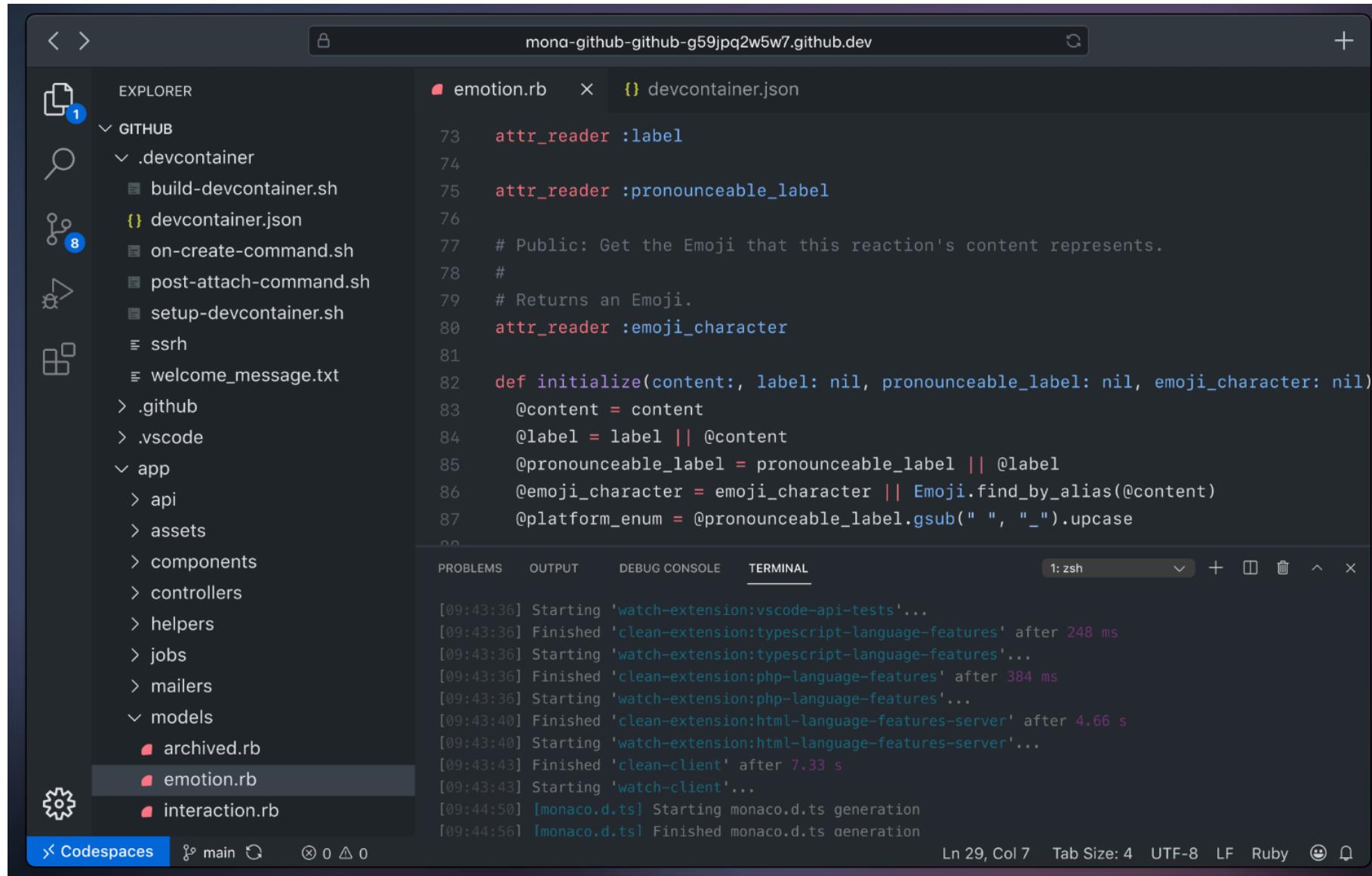
#### Explanation:

- **sudo** is used to run a command with elevated rights, allowing changes to system files.
- **apt-get** is a command-line tool for handling packages in Debian-based systems, such as Ubuntu.
- Common sub-commands include:
  - **install** to install a package.
  - **remove** to uninstall a package.
  - **update** to refresh the list of available packages.
  - **upgrade** to upgrade installed packages to their latest versions.
- You can leverage **apt-get --help** for more information on available options and usage.

# さまざまな環境でのGitHub Copilot

- IDE（統合開発環境）での利用
- GitHub Mobileでのチャットの利用
- GitHub ダッシュボードでのチャットの利用
- GitHub デスクトップアプリでの利用
- GitHub CLIでの利用
- GitHub Codespacesでの利用

GitHub Codespaces: Webブラウザー上から利用できる「クラウド開発環境」。見た目・機能はほぼVisual Studio Code。この環境でもGitHub Copilotが利用できる。



# Part 1 目次

- ・モジュール1 責任あるAI
- ・モジュール2 GitHub Copilotの概要
- ・モジュール3 プロンプトエンジニアリング
- ・モジュール4 高度な機能
- ・モジュール5 さまざまな環境でのGitHub Copilot
- ・モジュール6 管理とカスタマイズ

# GitHub Copilotのカスタマイズ(1) カスタムインストラクション

`.github/copilot-instructions.md` 内に、GitHub Copilotに対する指示を記述。

プロジェクト（またはGitHubリポジトリ）の直下に  
「`.github`」フォルダを作り、そこに  
「`copilot-instructions.md`」というファイルを作り、その中にカスタマイズの指示を（Markdownの文法で）指定。



エクスプローラー

...

copilot-instructions.md ×

...



SAMPLE29

.github

copilot-instructions.md

PrimeService

&gt; bin

&gt; obj

PrimeService.csproj

C# PrimeUtil.cs

PrimeService.Tests

&gt; bin

&gt; obj

PrimeService.Tests.csproj

C# PrimeUtilTests.cs

sample29.sln

.github &gt; copilot-instructions.md &gt; abc # テストコード

## # ドキュメントコメント

- 日本語で出力
- 「だ」「である」調で出力
- `summary`, `param`, `returns` タグを使用
- ドキュメントコメント内のコード部分 (`true`, `false` などのキーワードなど)は `c(``code``) タグを使用
- ドキュメントコメントの最後の部分には `<remarks>` このドキュメントコメントは GitHub Copilot によって生成された`</remarks>` を追加

## # インラインコメント

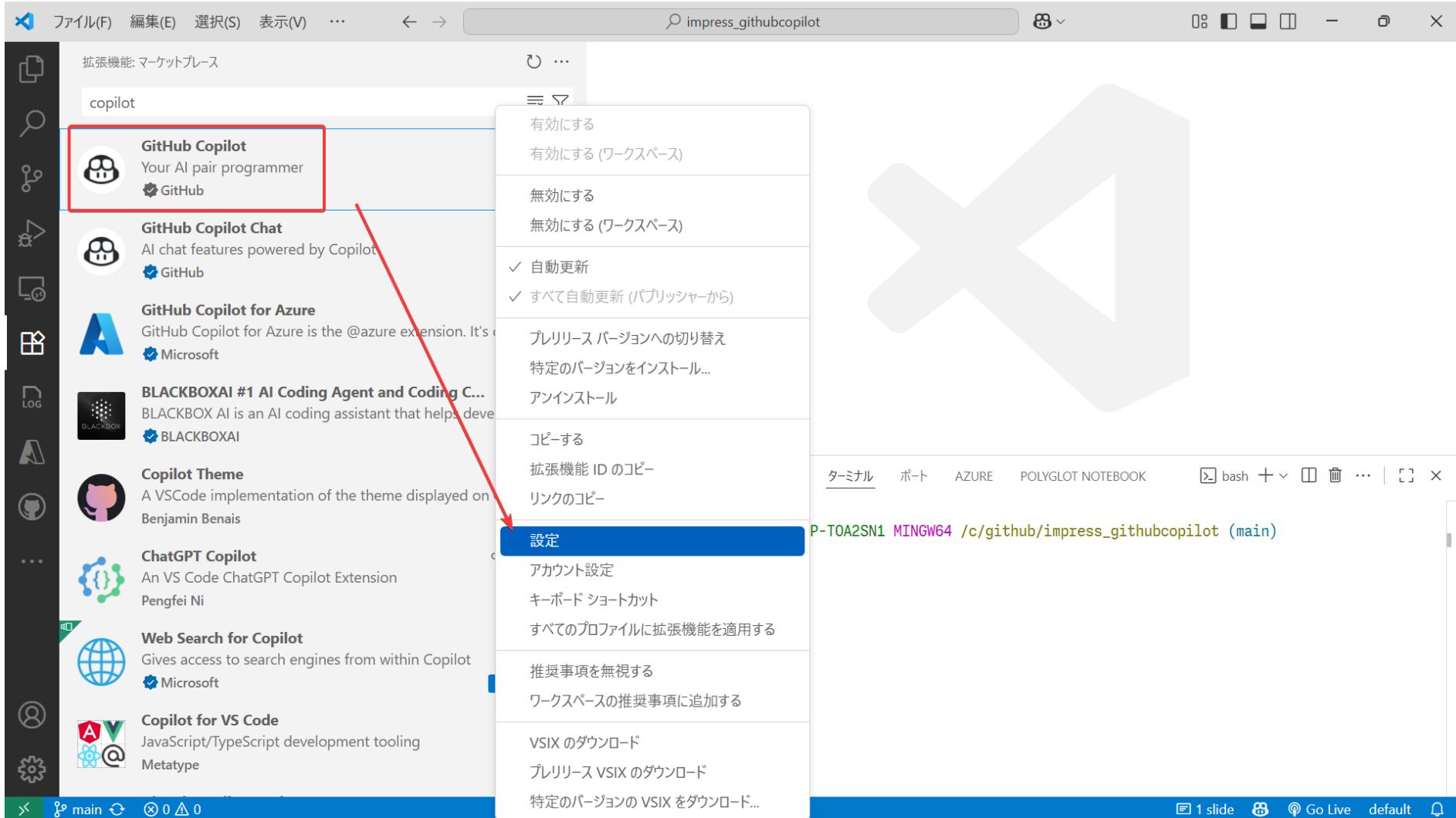
- 日本語で出力
- 可能な限り体言止めを使用して短く書く。例えば「チェックする」ではなく「チェック」と書く。
- 末尾の句点 ( `。` ) は省略
- 必要な場合は「だ」「である」調で出力

## # テストコード

- テストユニットフレームワークとして xUnit を使用
- use File-scoped namespace declaration like 'namespace SampleNamespace;'
- テストソースコードのファイルは `~.Tests` フォルダー以下に配置
- テストソースコードのファイル名は `~Tests.cs` とする
- 特に指示がない場合 `[Fact]` を使用
- 特に指示がない場合 テストケースは5件ほど作成

# GitHub Copilotのカスタマイズ(2)

## Visual Studio CodeのGitHub Copilot拡張機能の「設定」



# GitHub Copilot（コード提案）についてはそれほど設定できる項目はない

The screenshot shows the VS Code settings editor with the search bar set to "impress\_githubcopilot". The left sidebar has icons for file operations, search, and other extensions. The main area shows the results for "GitHub Copilot".

**GitHub > Copilot: Advanced**  
settings.json で編集

**GitHub > Copilot: Enable**  
Enable or disable auto triggering of Copilot completions for specified languages. You can still trigger suggestions manually using `Alt + \`

項目	値
*	true
plaintext	false
markdown	false
scminput	false

**GitHub > Copilot: Selected Completion Model**  
The currently selected completion model ID. To select from a list of available models, use the "Change Completions Model" command or open the model picker (from the Copilot menu in the VS Code title bar, select "Configure Code Completions" then "Change Completions Model"). The value must be a valid model ID. An empty value indicates that the default model will be used.

Bottom navigation bar: main, 0, Go Live, default, etc.

拡張機能: マーケットプレース



...

copilot

**GitHub Copilot**

Your AI pair programmer

**GitHub Copilot Chat**

AI chat features powered by Copilot

**GitHub Copilot for Azure**

GitHub Copilot for Azure is the @azure

**BLACKBOXAI #1 AI Coding Agent**

BLACKBOX AI is an AI coding assistant

**Copilot Theme**

A VSCode implementation of the them

Benjamin Benais

**ChatGPT Copilot**

An VS Code ChatGPT Copilot Extension

Pengfei Ni

**Web Search for Copilot**

Gives access to search engines from wi

**Copilot for VS Code**

JavaScript/TypeScript development tool

Metatype



1690ms



有効にする

有効にする (ワークスペース)

無効にする

無効にする (ワークスペース)

✓ 自動更新

✓ すべて自動更新 (パブリッシャーから)

特定のバージョンをインストール...

アンインストール

コピーする

拡張機能 ID のコピー

リンクのコピー

設定

アカウント設定

キーボード ショートカット

すべてのプロファイルに拡張機能を適用する

ワークスペースの推奨事項に追加する

VSIX のダウンロード

特定のバージョンの VSIX をダウンロード...



すべてのコマンドの表示 Ctrl + Shift + P

ファイルに移動する Ctrl + P

チャットを開く Ctrl + Alt + I

ターミナルの切り替え Ctrl + `

デバッグの開始 F5

1 slide



Go Live

default



# GitHub Copilot チャットについては多数の設定項目がある（41個）

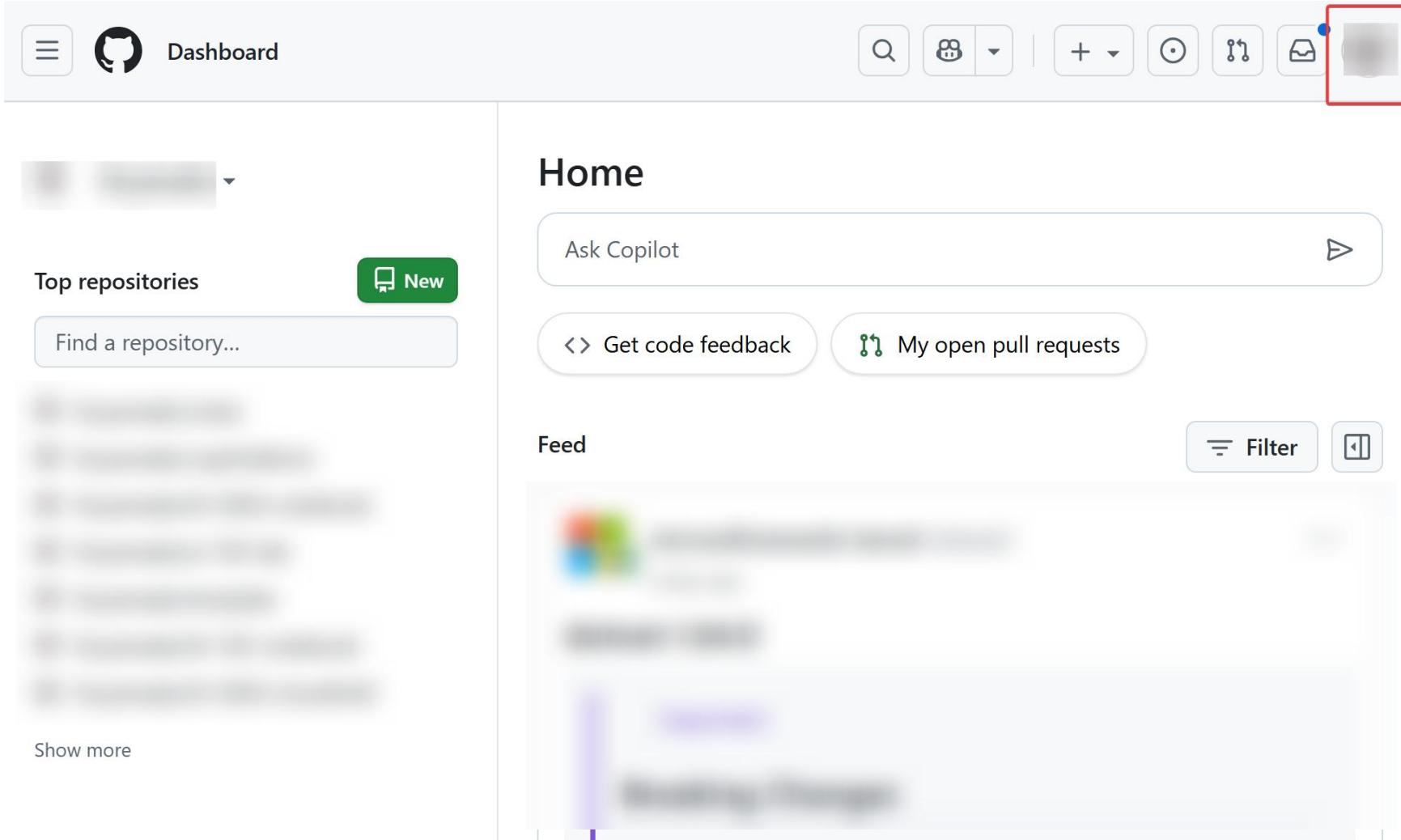
The screenshot shows the VS Code settings editor with the search bar set to "impress\_githubcopilot". The results list contains 41 items under the "GitHub Copilot Chat" category. The items are organized into sections:

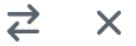
- GitHub > Copilot > Chat > Agent: Auto Fix**:  
A checked checkbox for "編集したファイルに関する診断を自動的に修正します。"
- GitHub > Copilot > Chat > Agent: Run Tasks**:  
A checked checkbox for "Configures whether Copilot Edits can run workspace tasks in agent mode."
- GitHub > Copilot > Chat > Code Generation: Use Instruction Files**:  
A checked checkbox for ".github/copilot-instructions.md"からのコード命令をCopilot要求に追加するかどうかを制御します。

注: 指示は簡潔かつ正確に記述してください。指示が不十分な場合、Copilotの品質とパフォーマンスが低下する可能性があります。Copilotのカスタマイズに関する[詳細情報](#)。
- GitHub > Copilot > Chat: Custom Instructions In System Message**:  
A checked checkbox for "有効にすると、カスタム命令とモード命令が、ユーザー メッセージではなく、システム メッセージに追加されます。"
- GitHub > Copilot > Chat: Locale Override**:  
Copilotが応答するロケール(例: 'en'、'fr')を指定します。既定では、Copilotは、VS Codeの構成された表示言語ロケールを使用して応答します。  
A dropdown menu currently showing "auto".
- GitHub > Copilot > Chat: Scope Selection**:  
A checkbox for "ユーザーが '/explain' を使用していて、アクティブなエディターに選択がない場合に、特定のシンボル スコープを選択するようにユーザーに促すかどうか。"

The left sidebar shows the "Settings" icon selected. The bottom status bar shows tabs for "main" and "default".

# GitHub Copilotのカスタマイズ(3) GitHubのWebサイト上の設定画面





Home

New

Ask Copilot

<> Get code feedback

My

Feed

Try Enterprise Free

Feature preview

Settings

Your profile

Your repositories

Your Copilot

Your projects

Your stars

Your gists

Your organizations

Your enterprises

Your sponsors

# GitHub Copilot で使用するモデル、プレビュー機能の有効・無効などを設定できる

The screenshot shows the GitHub Copilot settings interface. The first section, "OpenAI GPT-5 mini in Copilot", has a red box around its title and a red arrow points from the "Enabled" option in the dropdown menu below it. The "Enabled" option is also highlighted with a red box. The "Disabled" option is shown below it. The second section, "Copilot coding agent", is set to "Disabled". The third section, "MCP servers in Copilot", is set to "Enabled".

<b>OpenAI GPT-5 mini in Copilot</b>	<span>Preview</span>	<span>Select an option ▾</span>
You can use the latest OpenAI GPT-5 model in Copilot. <a href="#">Learn more about how GitHub Copilot selects models</a>		
<div style="border: 1px solid #ccc; padding: 5px; display: inline-block;"><span>Enabled</span> You will have access to the feature</div>		
<b>Dashboard entry point</b>	<b>Disabled</b>	
Allows instant chatting when you click the Copilot icon.	You won't have access to the feature	
<b>Copilot coding agent</b>	<span>Preview</span>	<span>Disabled</span>
Delegate tasks to Copilot coding agent in repositories where it is enabled. <a href="#">Learn more</a> .		
<b>MCP servers in Copilot</b>		<span>Enabled</span>
Connect MCP servers to Copilot in all Copilot editors and Coding Agent. Note that Coding Agent is in public preview. See MCP docs for <a href="#">Copilot Chat</a> and <a href="#">Coding Agent</a> .		

# GitHub Copilotのカスタマイズ(4) 「組織のポリシー」での設定

GitHub Copilot を Business / Enterprise プランで利用する組織では、「ポリシー」を使用して、組織レベルの設定を行うことができる

# 組織でのGitHub Copilot利用におけるポリシー（制限など）を設定できる

The screenshot shows the GitHub Copilot settings page within a GitHub Enterprise interface. The left sidebar includes links for Organizations, People, Policies, Repositories, Copilot (which is selected), Actions, Projects, Teams, Organizations, Code security and analysis, Personal access tokens (Beta), GitHub Connect, Code Security, Settings, and Compliance. The main content area is titled "GitHub Copilot" and contains sections for "Suggestions matching public code" and "Manage organization access to GitHub Copilot". Under "Manage organization access to GitHub Copilot", there are three options: "Disabled", "Allow for all organizations" (which is selected), and "Allow for specific organizations (5 organizations enabled)". A note states that enterprise use is governed by the GitHub Copilot Product Specific Terms. A "Save" button is at the bottom.

Search or jump to... / Pull requests Issues Codespaces Marketplace Explore

## GitHub Copilot

### Suggestions matching public code

GitHub Copilot can allow or block suggestions matching public code. See [the GitHub Copilot documentation](#) to learn more.

No Policy ▾

### Manage organization access to GitHub Copilot

Assign which organizations will have access to GitHub Copilot inside your enterprise. Admins of an organization with approved access will receive an email with setup instructions.

**Disabled**  
Disable GitHub Copilot access for my organizations

**Allow for all organizations**  
Allow access to GitHub Copilot for all organizations, including any created in the future

**Allow for specific organizations (5 organizations enabled)**  
Only specifically-selected organizations may use GitHub Copilot

Your enterprise's use of GitHub Copilot is governed by the [GitHub Copilot Product Specific Terms](#).

Save

たとえば「パブリックコードに一致するコード」の提案をブロックできる

The screenshot shows the GitHub Copilot settings page. On the left is a sidebar with navigation links like Organizations, People, Policies, Repositories, Copilot (which is selected), Actions, Projects, Teams, Organizations, Code security and analysis, Personal access tokens (Beta), GitHub Connect, Code Security, Settings, and Compliance. The main content area has a heading "GitHub Copilot" and a section titled "Suggestions matching public code". This section contains a note about GitHub Copilot's ability to allow or block suggestions matching public code, a link to the GitHub Copilot documentation, and a dropdown menu set to "No Policy". A large red box highlights this "Suggestions matching public code" section. Below it is another section titled "Manage organization access to GitHub Copilot", which includes options for "Disabled", "Allow for all organizations" (selected), and "Allow for specific organizations (5 organizations enabled)". It also mentions GitHub Copilot Product Specific Terms and a "Save" button.

# 「パブリックコードに一致するコードの提案」とは？

- 提案されたコードが、GitHub上で公開されたリポジトリのコード（パブリックコード）に偶然に一致する可能性がある
  - 可能性はかなり低い（1%未満とされる）
- パブリックコードにはライセンスが適用されている場合があり、その場合はそのライセンスを遵守しながらそれを利用する必要がある。
  - 特に、GPL (GNU General Public License) が適用されたソースを使用した派生物を配布する場合は、その派生物のソースコードを公開し、またその派生物もGPLで配布する必要がある。
- コードに適用されているライセンスによっては、提案されたコードを利用するかどうかを、ユーザーが判断する必要がある

## ■既存コードの類似コードが検出された場合、コードのリポジトリとライセンスが表示される

The screenshot shows the GitHub Copilot interface with two main sections:

**Left Panel (Code Citations):**

```
1 # Code Citations
2
3 ## License: 不明
4 https://github.com/JerMej1s/DevBuildLab6_1_PrimeNumbers/tree/
5 636a56738e0bb91f61af4b1328033f61d3e4e635/Program.cs
6
7 ...
8 <= 1) return false;
9     if (number == 2) return true;
10    if (number % 2 == 0) return false;
11    for (int i = 3; i <= Math.Sqrt(number); i += 2)
12    {
13        if (number
14 ...
15
16 ## License: 不明
17 https://github.com/FmoOliveira/myPortal/tree/
18 eb47865e7bce417fbfc581e1258d2d0d2cdfffb14/index.md
19
20 ...
21 == 2) return true;
22     if (number % 2 == 0) return false;
23     for (int i = 3; i <= Math.Sqrt(number); i += 2)
24     {
25         if (number % i == 0) return false;
26     }
27 ...
28
```

**Right Panel (GitHub Copilot Output):**

GitHub Copilot detected a prime number service implementation:

```
namespace Prime.Services
{
    public class PrimeService
    {
        /// <summary>
        /// 入力された数が素数かどうかを判定します。
        /// </summary>
        /// <param name="number">判定する整数</param>
        /// <returns>素数であれば true、そうでなければ false</returns>
        public bool IsPrime(int number)
        {
            if (number <= 1) return false;
            if (number == 2) return true;
            if (number % 2 == 0) return false;
            for (int i = 3; i <= Math.Sqrt(number); i += 2)
            {
                if (number % i == 0) return false;
            }
            return true;
        }
    }
}
```

Annotations highlight the detected code citations and the generated prime service implementation. A red box encloses the citation section on the left and the generated code on the right. Red arrows point from the GitHub Copilot interface to these boxes. A button at the bottom right of the Copilot panel is labeled "一致を表示する" (Show Match).

# 「パブリックコードに一致するコードの提案」をブロックした場合の動作

- GitHub Copilotがコードの提案を生成
- 生成されたコードの提案とその周囲の約 150 文字のコードを、GitHub 上のパブリックコードと照合
- パブリックコードに一致、または、近い一致がある場合、その提案は表示されない