

エンジニア集まれ！  
**GitHub Copilot** の  
オンラインワークショップ開催！

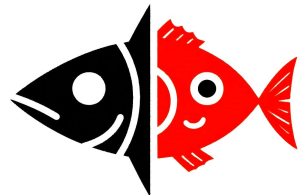
株式会社クラベス

2025.3.13

15:00開始



# 自己紹介



## 岩嶋大 (Iwashima Dai)

- 前職ではドライブレコーダーの Web サービス開発を行っていました。
- 現在はShopifyを用いたECサイト構築や、勤怠、給与システムと会計システムを繋ぐ連携システムの開発や認証基盤の開発業務に携わっています。
- GitHub Copilot導入チームに参画。



## 今村佳吾 (Imamura Keigo)

- 前職では創業メンバーとして、九州でシステム開発サービスを展開。  
ふるさと納税管理 / 医療機関の予診票 / LINE連携顧客管理など  
10以上のプロジェクトにPM/エンジニアとして携わる。
- 更なるスキルUPとクラベスの社風に惹かれクラベスに入社。  
GitHub Copilot導入チームに参画。

# 株式会社クラベスについて

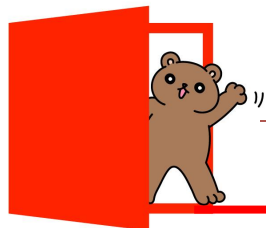
会社名	株式会社クラベス
代表者	堀内 文雄 (Horiuchi Fumio)
本社	東京都墨田区両国 1-3-9 ムラサワ第二ビル 3F
事業内容	ECサイト構築および周辺システムの構築 OMO基盤構築(認証基盤/会員基盤構築) ポイント管理システム構築 ライセンス管理システム構築 データ連携サービスの開発 学習塾向けAPIシステム構築



弊社HP



wantedly



## 22世紀の話をしよう

Let's talk about the 22nd century!

# アジェンダ

**1** 質問先について

**2** GitHubCopilotについて

**3** 各機能の紹介

**4** 動画によるデモンストレーション

**5** AIツールとの向き合い方

**6** クラベスでの導入事例

# 1 質問先について

---



# 質問先について

## 「Slido」にて質問を受け付けています

ワークショップ内で疑問点等ございましたら、  
ぜひお気軽にSlidoにてご投稿ください！

只今よりTeamsのメッセージにて、Slidoのリンク先をお知らせいたします。

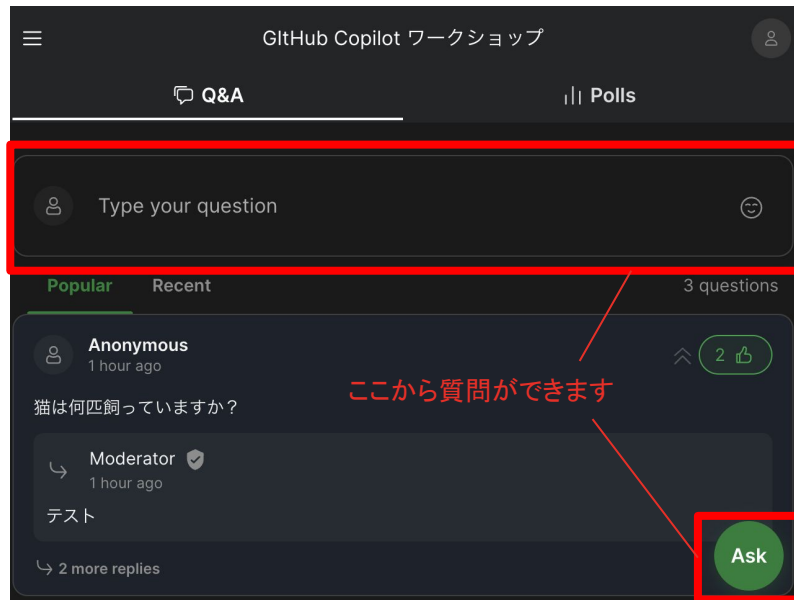
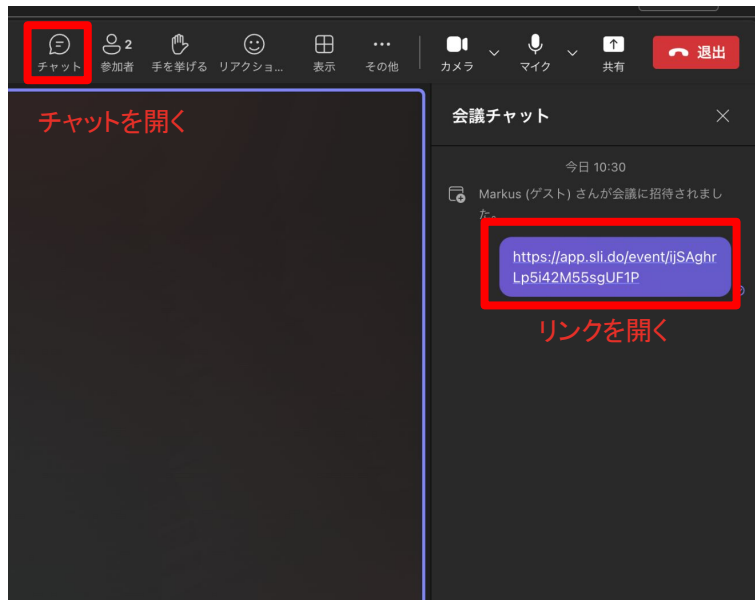
※Teamsのメッセージからの質問はお控えください

回答スタッフが待機しておりますので、順次ご回答させていただきます。



# 質問先について

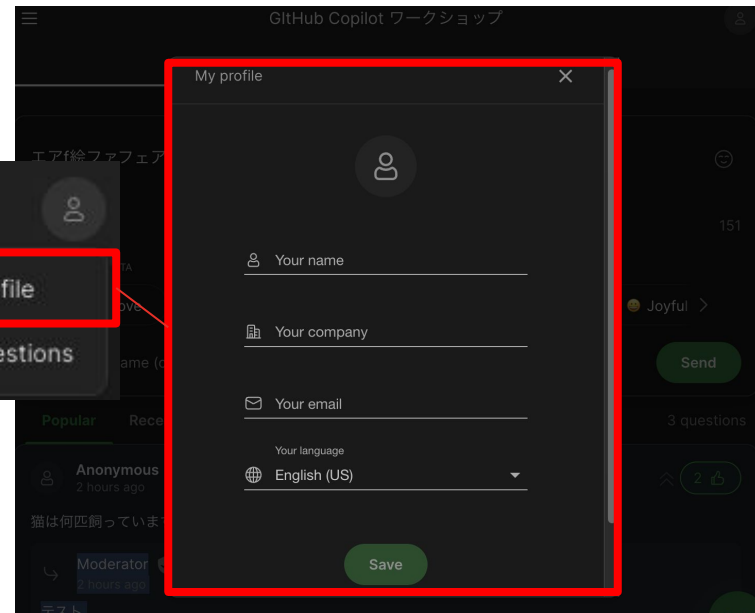
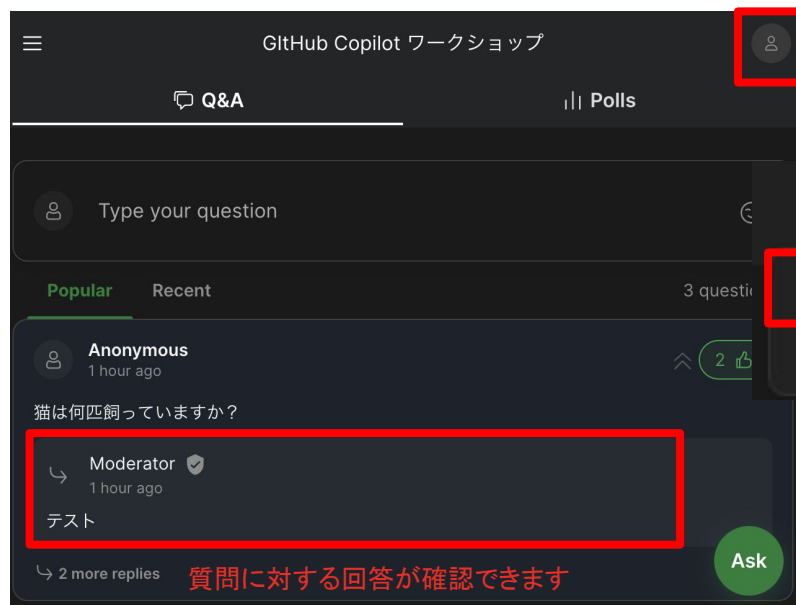
## Slidoの使い方①



# 質問先について

## Slidoの使い方②

右上のアイコンを選択し「My Profile」を選ぶと  
プロフィール情報を登録できます（登録しない場合は匿名）





**2**

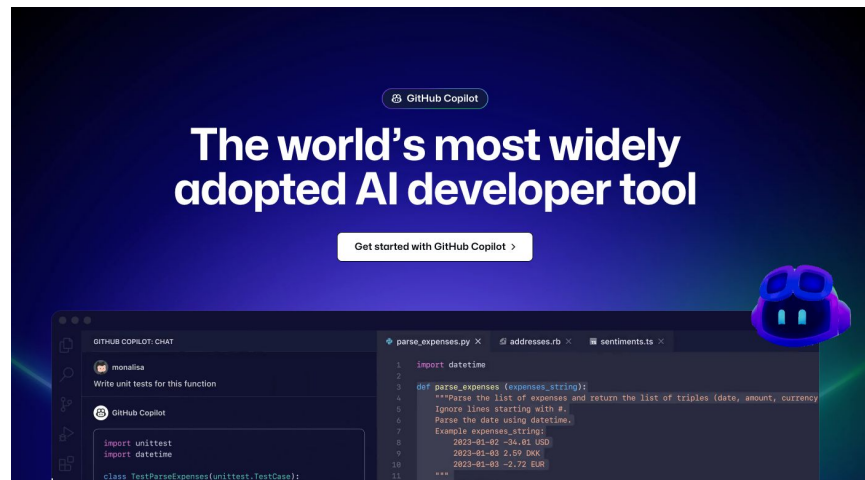
# GitHubCopilotについて



# 世界で 最も広く 採用されている AI開発ツール

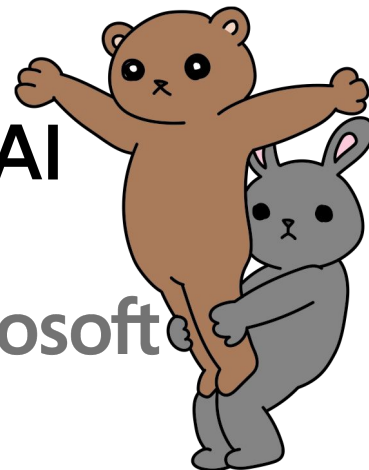
 **GitHub Copilot**

<https://github.com/features/copilot>



# デジタルと イノベーションの 融合

 **GitHub Copilot**



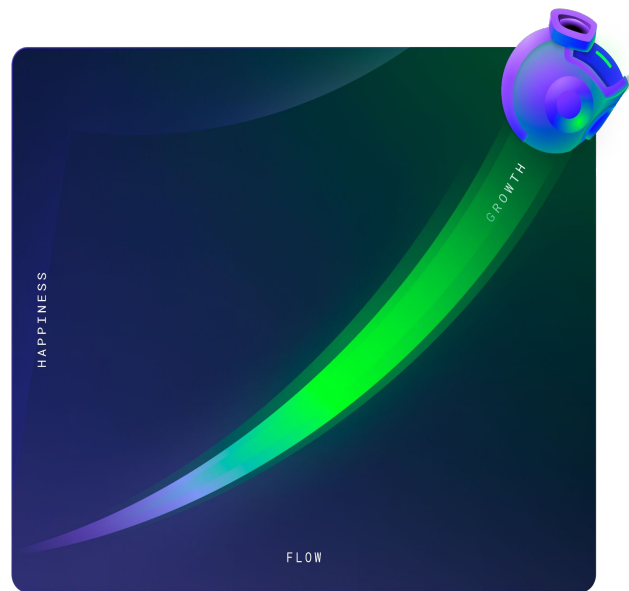
# DevExとは

## ■ Developer Experience

- 開発者が日々経験している業務  
言い換えれば「エンジニアの業務体験や環境」そのもの

マッキンゼーの調査によると ...

生成AIベースのツールを使用している開発者は、  
「全体的な幸福感、充実感、没頭状態」を報告する確率が2倍以上」  
であったと報告されている。



引用: <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/unleashing-developer-productivity-with-generative-ai>

## DevEX向上に必要な3要素

### フィードバックループ

❗ フィードバックの遅い開発は  
フラストレーションが溜まる



#### 改善効果

コードレビューの応答時間の短縮

**革新性20%UP**

緊密なフィードバックループ

**技術的負債50%軽減**

### 認知負荷の軽減

❗ キャッチアップコストは  
開発者の大きな負担



#### 改善効果

理解度の高い開発者

**生産性42%UP**

直感的なツール・プロセス

**革新性50%UP**

### 没入状態の実現

❗ 開発の本質以外の作業は  
なるべく減らしたい



#### 改善効果

深い作業に注力

**生産性50%UP**

魅力的な仕事をしている開発者

**生産性30%UP**

# AIは DevExにおける 次世代の 強力な一手



# 開発体験の改善メリット

1

## 収益までの時間を短縮

DevExの改善による

収益の拡大

- ・市場投入までの時間短縮
- ・収益性の向上

2

## 業務効率の向上

業務効率の向上による

業績アップ

- ・生産性
- ・効率性
- ・製品の品質

3

## 人材定着率の向上

優秀なエンジニアを

惹きつけ、維持する

- ・高いチームモラル
- ・やる気とやりがいのある仕事

# コパイロットによる効果

世界で最も広く採用されているツール

開発者の幸福を最大化

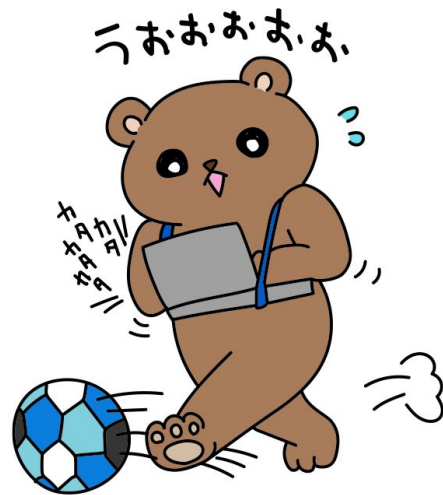
開発者の生産性向上

ソフトウェア開発を加速する





# ワークフローと ソフトウェア開発の 俊敏性を加速させる 実証済みのAI



# 開発者の中で1位指名



開発者の75%  
がより充実  
している



AIコーディング  
ツールを使用する  
開発者の55%が  
GitHub Copilotを選択

開発者の75%が  
来年も試したいと  
考えている

Copilotにより  
コーディングが  
55%高速化

150万人以上  
の開発者

confidential.

引用: <https://github.blog/2022-09-07-research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/>  
<https://github.blog/2023-06-13-survey-reveals-ais-impact-on-the-developer-experience/>

© 2024 CLAVES Co.,Ltd.

# Copilotによる 開発者の生産性は、 スピードだけでは ありません

73%

開発者の73%がより**没頭状態**にあると回答

74%

開発者の74%がより**満足度**の高い仕事に集中できると回答

87%

開発者の87%が反復作業における**精神的労力**が**軽減**したと回答

引用: <https://github.blog/2022-09-07-research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/>  
<https://github.blog/2023-06-13-survey-reveals-ais-impact-on-the-developer-experience/>

Copilotによる  
開発者の生産性は  
開発者が  
最も重要なことに  
集中できることを  
意味します



多くの時間

- 企画立案
- 共同作業
- プレスト
- デザイン

- 脆弱性の発見
- ドキュメント検索
- 変更点とコメント要約
- 既存コード読解
- Gitコマンド学習
- 構文修正
- テストや繰り返しコード
- デバッグ

少ない時間

# Copilotを使用した 開発者の生産性は 企業に利益をもたらします

リリースサイクルの俊敏性

55%



Copilotを使用した  
開発者は55%早く  
タスクを完了

開発者のオンボーディング

57%



開発者の57%がAIがス  
キルUPに役立ち日常的  
に使用したいと回答

開発者の保持

41%



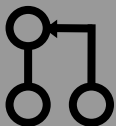
開発者の41%が  
やる気を失う状態になるこ  
とが軽減したと回答

引用: <https://github.blog/2022-09-07-research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/>  
<https://github.blog/2023-06-13-survey-reveals-ais-impact-on-the-developer-experience/>

# GitHub Platform



AI搭載



## Collaboration

Discussions

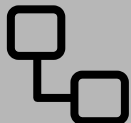
Merge queue

Pull request

Projects

Issues

Search



## Productivity

Copilot

Codespaces

npm

CI/CD with Actions

Automation

Runners



## Security

Secret scanning

Dependabot

Advanced Security

Security overview

Supply chain



## Scale

100M+ Developers

Source code management

3

## 各機能の紹介



# 各機能の紹介

Suggest

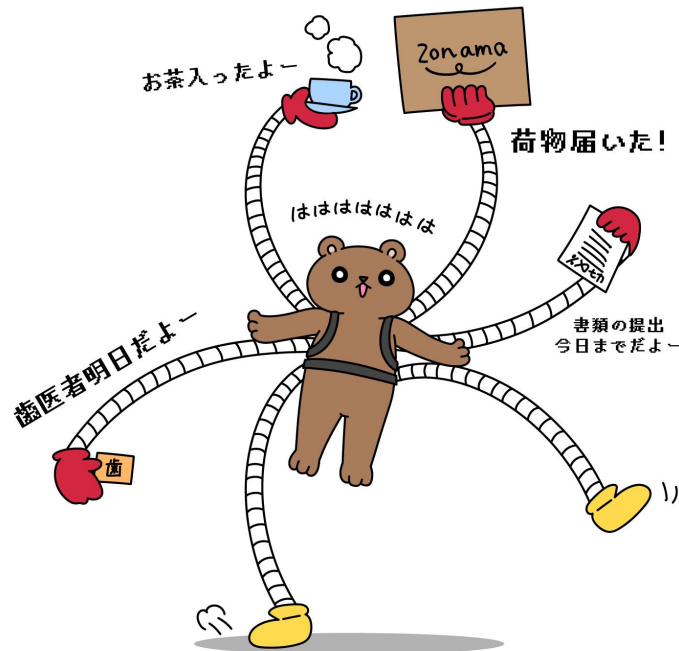
Chat

CLI

PullRequest

CustomPrivateModels

Documentation





# 各機能の紹介

## Suggest

- コメントをコードに変換

コメントからコードを生成し、コメントの情報量などプロンプト次第で生成物の精度を向上できます。

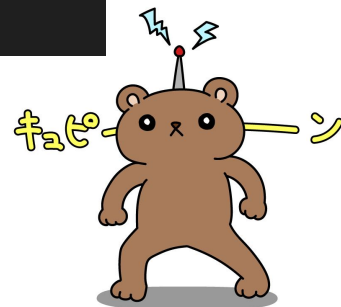
- 繰り返し作業の半自動化

既存ソースを解析し、次にコーディングする内容や定型文を素早く提案します。

- 提案の豊富さ

複数の提案を受けることができ既存ソースの代替案の提案やリファクタリングも可能です。

```
// 1から100まで順に3の倍数なら「Fizz」、5の倍
const fizzBuzz = () => {
  for (let i = 1; i <= 100; i++) {
    if (i % 15 === 0) {
      console.log("Fizz Buzz");
    } else if (i % 3 === 0) {
      console.log("Fizz");
    } else if (i % 5 === 0) {
      console.log("Buzz");
    } else {
      console.log(i);
    }
  }
};
```



# 各機能の紹介

## Chat

- **コーディングサポート全般に対応**

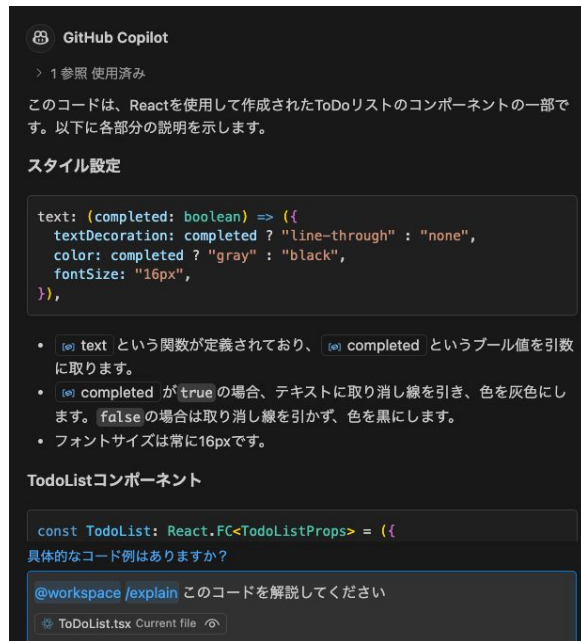
プロンプトやこれまでのコンテキストをもとにコーディングのサポートを行います。

- **コードブロックの説明**

/explainコマンドにより、理解が難しいコードブロックの内容を詳しく説明します。

- **エラー解決、脆弱性の指摘**

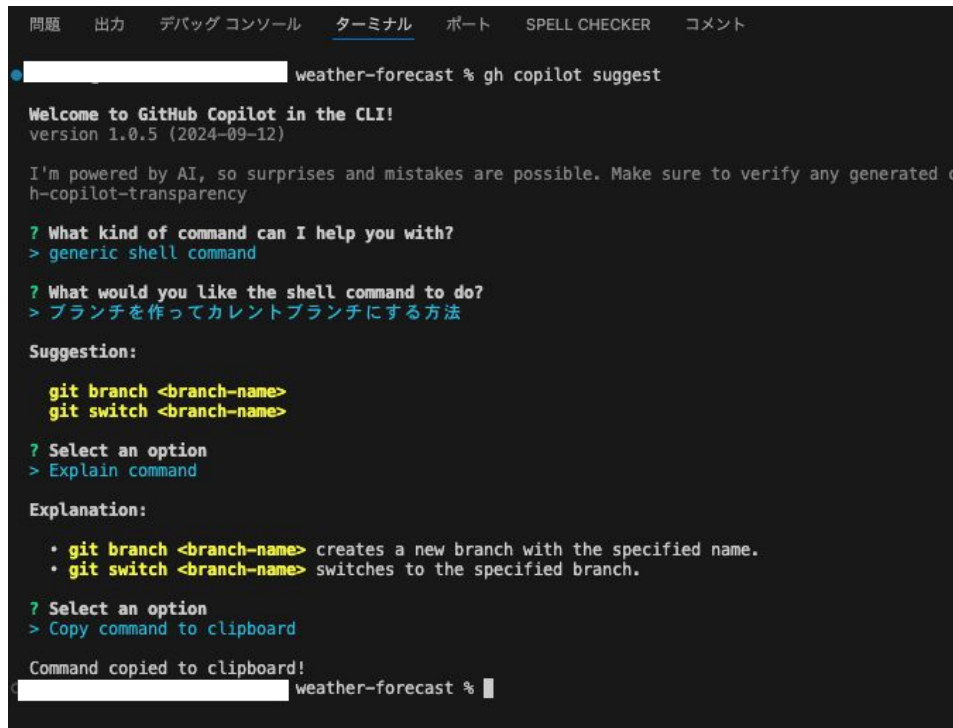
エラー文をそのままプロンプトとして送ることで解決策の提案や、コード内に潜む脆弱性を指摘し、適切な修正案を提案します。



# 各機能の紹介

## CLI

- **ターミナルでのコマンドの支援**  
対話形式でコマンドの説明やコマンドの提案を受けることができます
- **ターミナルでのエラー分析**  
ターミナルでのエラー出力を分析して、失敗したシェルコマンドを修正します



```
問題 出力 デバッグ コンソール ターミナル ポート SPELL CHECKER コメント
weather-forecast % gh copilot suggest

Welcome to GitHub Copilot in the CLI!
version 1.0.5 (2024-09-12)

I'm powered by AI, so surprises and mistakes are possible. Make sure to verify any generated code.
h-copilot-transparency

? What kind of command can I help you with?
> generic shell command

? What would you like the shell command to do?
> ブランチを作ってカレントブランチにする方法

Suggestion:

git branch <branch-name>
git switch <branch-name>

? Select an option
> Explain command

Explanation:

• git branch <branch-name> creates a new branch with the specified name.
• git switch <branch-name> switches to the specified branch.

? Select an option
> Copy command to clipboard

Command copied to clipboard!
weather-forecast %
```

# 各機能の紹介

## PullRequest

### Public Preview

- Pull Requestの要約の自動生成

コードの変更内容を要約し、概要を自動生成します。

- コードレビューの支援

レビュー時に変更箇所を解析し、提案や指摘を提供します。

- コードスキャンで検出された問題の修正提案 (Public or Enterprise)

CodeQLを使用し既存のソースコードの脆弱性を分析し検出された問題に対する具体的な修正案をPR形式で提示します。

# 各機能の紹介

## CustomPrivateModels

Alpha

- **カスタムモデルの作成**

独自に指定したリポジトリ内のコードに基づき、  
組織のニーズに合わせた特定のコーディングスタイルやパターンが組み込まれた  
カスタムAIモデルを作成します。

- **独自のプログラミング言語に対応**

独自のプログラミング言語または  
あまり一般的ではないプログラミング言語用に作成すると効果的です。

# 各機能の紹介

Documentation

Private Beta

- **時間をかけない信頼できる回答**

ライブラリのメンテナーによって書かれた最新の情報を使用し、元のドキュメント内の引用を含む回答をサポートします。

- **パーソナライズされた回答**

開発者の経験レベル、ライブラリの理解度に応じた最適な回答を提供します。

- **ライブラリ間での情報結合(今後の方向性)**

両方のライブラリで関連情報を検索し、それを 1 つの回答にまとめます。

# 各機能の紹介

Suggest

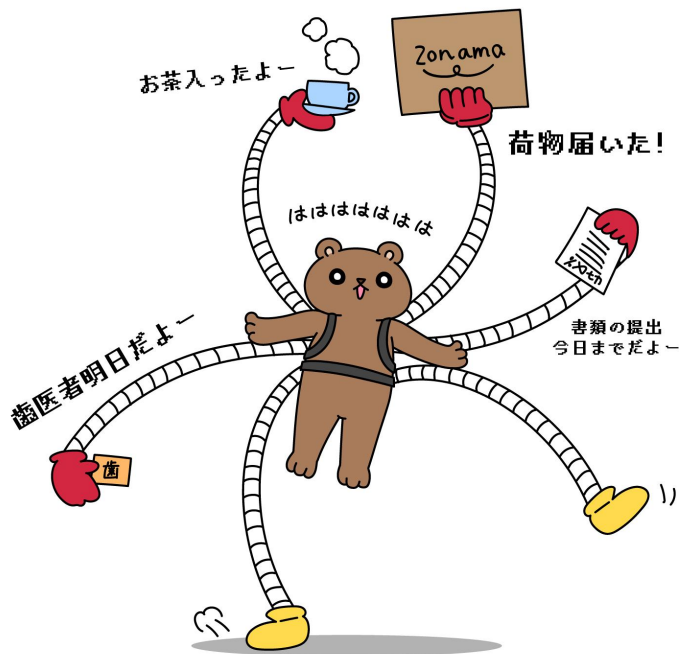
Chat

CLI

PullRequest

CustomPrivateModels

Documentation



4

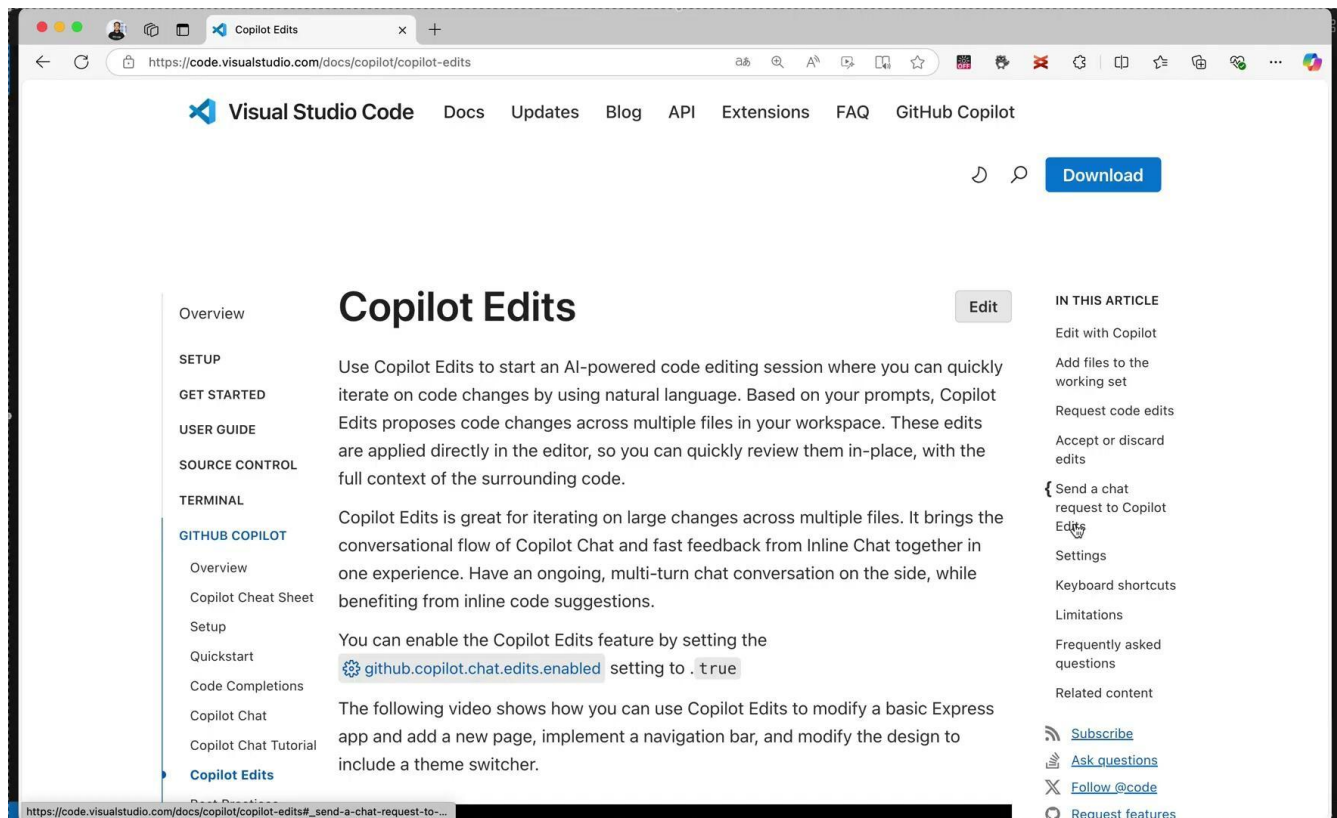
## 動画によるデモンストレーション





# 動画によるデモンストレーション

## Copilot Editsを使用した複数ファイル編集



5

## AIツールとの向き合い方



# AIツールとの向き合い方

## ■ メリット

- 作業効率の向上

Suggest、Chatにより開発スピードが上昇する。

- デバッグ時のエラー調査

エラー文をそのまま Copilotに投げることで、  
エラーの解説と解決策を提示する。

- 技術力の向上

/explainコマンドにより、  
わからないことに対し遠慮なく聞ける環境を作れる。



# AIツールとの向き合い方

## ■ デメリット

- 誤った提案を採用

生成されたコードの良し悪しを判断できないと、  
想定外の開発を行ってしまう可能性がある。

だからこそ/explainや、Copilotとの対話を駆使して、  
生成物の理解、技術力の向上を図る必要がある。

- 独創的なコードが生まれにくくなる

Copilotは既存のソースコードを学習した上で提案を行なっている。  
提案する生成物もこれまで他の誰かが作成したもの踏襲に近いため、  
新規で独創的なコードを書くには、やはり人の手が必要になる。



# AIツールとの向き合い方

## ■ セキュリティ

- ソースコードの漏洩

プロンプトの内容や Copilotが読み取った内容が、  
学習モデルとして利用される可能性がある。

情報が漏洩しないよう GitHub上での設定を正確に行わなければならない。

- 第三者からの著作権侵害

Copilotが提案する情報は、  
GitHubのパブリックリポジトリを元にした学習モデルから生成している。

そのため、1%未満の確率で第三者が開発した内容と相違ない提案が行われる可能性がある。  
今後の機能でチェック機能が追加される。

また、Copilotの出力に対する著作権保護プログラム「**Copilot Copyright Commitment**」  
があります。

# AIツールとの向き合い方

## ■ 利用者の確認

- 統計情報からの確認

組織やチーム単位で Copilotの使用状況を取得できる  
REST APIエンドポイントが設けられているため、  
逐一利用率の確認が可能。

- アクションの取り方

利用頻度が低いから解約しようではなく、  
使うことで効率よく開発できることが既の実証されているため、  
利用頻度が改善するようにアクションを起こすことが重要。



# AIツールとの向き合い方

## Copilotの自動レビュー機能

- Copilotのレビューのみでマージしない

Copilotはあくまで「副操縦士」という立場で、メインの操縦士は Copilotを使用している開発者自身。自動レビューの結果を過信するのではなく、あくまで最終レビューは「人」が行う必要がある。

```
packages/copilot4prs/app/models/pull_requests/copilot/code_review_creator.rb
```

...	...	@@ -56,7 +56,8 @@ def create
56	56	generated_comment_results << true
57	57	elsif pull.user&.feature_enabled?(:copilot_pr_reviews_submit_empty_reviews)
58	58	# Reviews currently can't be submitted with no body, so we need to add a comment.
59	-	review.body = NO_COMMENTS_BODY
	59	+ random_greeting = ["Hello!", "Howdy!", "Hey there!"].sample
	60	+ review.body = [random_greetin, NO_COMMENTS_BODY].join(" ")



**Copilot** (AI) 27 minutes ago

Beta ...

The variable name 'random\_greetin' is misspelled. It should be 'random\_greeting'.

Suggested change

60	-	review.body = [random_greetin, NO_COMMENTS_BODY].join(" ")
60	+	review.body = [random_greeting, NO_COMMENTS_BODY].join(" ")

# AIツールとの向き合い方

## ■ エンジニアの成長へのアプローチ

- 理解できないソースコードの解説

/explainコマンドを使用することでソースの解説を依頼できるため、  
わからないことを後回しにする必要がなくなる。

- リファクタリング

今までの実装に対してリファクタリングの提案を依頼することができるため、  
より良い開発方法を教わることができる。

- 正規表現の生成や物理名の提案

初心者を悩ませる正規表現も生成可能であるため、  
日本語でルールをきちんと記載すればよくなった。  
変数名や、メソッド名など英語でどう表現すればいいか悩む時間も、  
Copilotを使用することで削減可能に。





# AIツールとの向き合い方

## ■ パソコンやスマホが登場した時と同じ現象

- 業務の変革

パソコンや、携帯電話、スマートフォン、クラウドが登場した時と同じように、AIの登場でパラダイムシフトが起きている。AIで処理できる業務が格段に増加する。

- 活用できていない企業への信頼

活用できない企業は成長速度が鈍化してしまうことが予想され、優秀な人材はその必要性を認識し、活用できていない企業を避けることも予想される。営業や採用の面にも影響がでる。

- 使い方の重要性

新しい技術は多くの人にとって良くも悪くも未知の技術。  
そのリスクに対応しながらうまく活用し、成長してきた企業が存在する。  
つまり大事ななのは、正しい理解と活用方法をきちんと定義すること。

6

## クラブスでの導入事例



# クラブスでの導入事例

## 社内全体としてのルール

1

### Copilot導入チームの 発足

社内ルールの制定

Copilotの普及活動

- ・安全性の担保
- ・開発の効率化
- ・利用促進

2

### 最終チェックは「人」

AIに任せきりにしない

熟練の技術者としての視点

- ・品質の向上
- ・学習に繋がる

3

### Copilot未経験者に対し ペアプロ実施

理解できないソースは

使用しない

- ・不具合の混入を阻止
- ・最適な使い方をマスター

# クラブスでの導入事例

## ■ 効果測定方法

- REST APIにてメトリクスを取得する

日ごとの使用回数を計測でき、  
使用回数の遷移から使用が定着しているかが判断できる。

# クラブスでの導入事例

## 効果測定方法

採用率推移（件数ベース）（%）

