

# GitHub Copilotのすべて

## Azure DevOpsオンライン Vol.13 Agentic DevOpsって何？

---

2025年6月21日

**GitHub Copilotの最強なポイントは？**

# GitHub Copilotは支払いやすさが最強！

## 企業での導入が簡単な理由

### 既存の支払い手段を活用

- GitHubの既存契約に追加するだけ
- 新規の支払い手続き不要

### 他の生成AIサービスの課題

- クレジットカード必須
- 企業での新規決済は承認が大変
- 経理処理が . . .

# 本日のゴール

**(ほぼ) すべての機能の概略を知る**

---

デモ中心に解説します

※ 管理機能およびEnterprise限定などの一部機能を除く

## 本日のゴール **2**

本発表内容の学習の仕方を理解する

---

# 本日のゴール

本発表内容の学習コンテンツが自作できるようになる **3**

---



## 自己紹介

- 名前: 中村 充志
- 所属: リコージャパン株式会社
- Microsoft MVP for Development Technologies (2017～)

## 本日の環境

### 使用環境

- GitHub Copilot Pro+ (Microsoft MVP特典)
- VS Code + C#

### Special Thanks

**Microsoftさんありがとう！** 

MVP特典でPro+を使わせていただいています！



# 発表コンテンツについて

## 学習コンテンツ

---

<https://agreeable-island-0c8e4d900.6.azurestaticapps.net/>

## 発表資料

---

<https://github.com/nuitsjp/all-of-github-copilot>

**たのしい話（機能）の前に楽しくない（お金）の話**

# GitHub Copilot プラン比較

プラン	月額	主な対象	プレミアムリクエスト
Free	無料	個人開発者	50/月
Pro	\$10	個人開発者	300/月
Pro+	\$39	AIパワーユーザー	1,500/月
Business	\$19/ユーザー	チーム・組織	300/ユーザー/月
Enterprise	\$39/ユーザー	大規模組織	1,000/ユーザー/月

一部機能の制限とプレミアムリクエストのリミットの相違

# プレミアムリクエストとは

## 基本概念

---

### リクエストの定義

- Copilotへのすべての依頼
- チャットでの質問、コード生成、拡張機能の利用など

### 課金モデル

- **基本:** プランに含まれる月間許可量
- **追加:** 許可量超過時は **\$0.04/リクエスト**

# モデル別プレミアムリクエスト消費量

モデル	Premium リクエスト
基本モデル（現在はGPT-4.1）	0（有料ユーザー）、1（Free）
Claude Sonnet 4	1
Claude Opus 4	10
o3	1
o4-mini	0.33

# BusinessとEnterpriseの共存は可能か？

## よくある疑問

- ユーザーによって利用頻度が異なり、使い分けたいことがある
- コスト最適化を図りたい

 答え: **共存可能です！**

## BusinessとEnterpriseの共存可能

### 混在のメリット

- 同一リポジトリで異なるライセンスユーザーの混在は可能
- ユーザーの役割に応じたライセンス割り当て

### 注意点

- ただ現状のライセンス設計にだいぶ無理がある
- 将来変わる可能性はあり

## 重要なポイント

### Organization制限

- OrganizationにはBusinessまたはEnterprise何れかのみ設定可能
  - 例: BusinessレベルのOrganizationでライセンスを割り当てるとBusinessに

### Team単位の管理

- CopilotライセンスはOrganization内のTeamごとに割り当てられる
  - Copilotが有効なユーザーと無効なユーザーが混在可能

### ライセンス優先度




- ライセンスを重複割当した場合、有効ライセンスは優先度によって決定
  - **Enterprise > Business > Pro+ > Pro**
  - おそらくガバナンスを優先するため



# Biz/Entの混在おすすめ設定

 リポジトリ管理Organizationとライセンス管理Organizationを分ける

## 構成例

Organization	ユーザー	用途
 <b>Business License Org</b>	User B1, User B2	ライセンス管理専用
 <b>Enterprise License Org</b>	User E1, User E2	ライセンス管理専用
 <b>Repository Org</b>	全ユーザー	実際の開発作業

## 設定のポイント

- ライセンス用Organizationにはリポジトリを持たせない
- リポジトリは別Organizationに配置

# 機能紹介

## 機能①

#	機能名	Free	Pro	Pro+	Business	Enterprise
1	Code completion	✓	✓	✓	✓	✓
2	Copilot Chat	✓	✓	✓	✓	✓
3	Copilot coding agent	✗	✗	✓	✗	✓
4	Copilot in the CLI	✓	✓	✓	✓	✓
5	GitHub Copilot code review	✓	✓	✓	✓	✓
6	GitHub Models	✗	✗	✓	✓	✓
7	Repository and personal custom instructions	✓	✓	✓	✓	✓
8	Organization custom instructions	✗	✗	✗	✓	✓

## 機能②

#	機能名	Free	Pro	Pro+	Business	Enterprise
9	Copilot prompt files	✓	✓	✓	✓	✓
10	Copilot pull request summaries	✗	✓	✓	✓	✓
11	Copilot text completion	✗	✗	✗	✗	✓
12	GitHub Copilot Extensions	✓	✓	✓	✓	✓
13	Copilot Workspace	✗	✓	✓	✓	✓
14	Copilot Spaces	✓	✓	✓	✓	✓
15	Copilot knowledge bases	✗	✗	✗	✗	✓

## ⚠️ 最初にお断り

### 🚫 本日デモできない機能

次の機能は本日お見せできません。

1. 🔒 **制限Preview機能** - 現在募集されていないもの
2. 🏢 **Enterprise限定機能** - 高度な管理・統合機能

これらは簡単な解説のみとなります。

# 1. Code Completion

## リアルタイムコード補完

---



### 概要

- AIによる自動的なコード提案
- 文脈を理解した賢い補完
- 複数行の提案も可能

## 2. Copilot Chat

### 対話型コーディングアシスタント

---

#### 3つのモード

モード	用途	特徴
Ask	質問・説明	コードの理解、技術的な質問
Edit	ファイル編集	複数ファイルの制御された編集
Agent	自律的実行	タスクの自動完了、ツール実行

#### コンテキスト指定

#file , #codebase , #selection , @workspace

# Copilot Chat モードの機能比較

## 各モードの機能範囲

機能	Ask	Edit	Agent
質問への回答	✓	✓	✓
単一ファイル編集	✗	✓	✓
複数ファイル編集	✗	✓	✓
ターミナル実行	✗	✗	✓
エラー自動修正	✗	✗	✓
実装計画作成	✗	✗	✓
MCP利用	✗	✗	✓
利用可能モデル	◎	◎	○



# なぜ機能の少ないモードを選ぶのか？

## Ask モードを選ぶ理由

- **最速の応答**: 編集機能がない分、処理が軽い
- **純粋な情報取得**: コードを誤って変更するリスクがゼロ
- **リクエスト消費最小**: 対話のみで追加処理なし

## Edit モードを選ぶ理由

- **予測可能性**: 指定したファイルのみ変更（Agent は探索的）
- **高速処理**: 計画フェーズがないため Agent より速い
- **制御性**: ツール実行なし、コード変更のみに集中
- **中程度のリクエスト消費**: Agent の1/3～1/5程度

## 3. Copilot Coding Agent

### 自動実装エージェント

---

#### 機能

- GitHub IssueからPRまでの自動実装
- コードの自動生成と検証
- テストの作成と実行

#### ワークフロー

1. Issueの理解と分析
2. 実装計画の作成
3. コードの自動生成

## 4. Copilot in the CLI

### コマンドライン支援

---

#### 機能

- コマンドの提案
- コマンドの説明
- エラーの解決支援

#### 使用例

```
gh copilot suggest "ファイルを再帰的に検索"  
gh copilot explain "git rebase -i HEAD~3"
```

# 5. GitHub Copilot Code Review

## AIによるコードレビュー

---

### 機能

- PRの自動レビュー
- 改善提案の生成
- セキュリティ問題の検出

### レビュー内容

- コード品質
- ベストプラクティス
- 潜在的なバグ

## 6. GitHub Models

### AI言語モデルへのアクセス

---

#### 機能

- 業界をリードする大規模・小規模言語モデルへの直接アクセス
- 様々なAIモデルの試用と比較
- プロトタイピングと実験

#### 用途

- モデルの選定
- APIの事前検証
- 性能比較

# 7. Repository and Personal Custom Instructions

## 応答のカスタマイズ

---

### 機能

- 個人の好みに基づく応答調整
- 使用ツール・フレームワークの指定
- プロジェクト固有の要件設定

### 設定例

- コーディング規約の指定
- 優先言語/フレームワーク

## 8. Organization Custom Instructions

### 組織レベルのカスタマイズ

---

#### 機能

- 組織全体での統一された応答設定
- 企業のコーディング規約の適用
- セキュリティポリシーの強制

#### メリット

- 一貫性のあるコード生成
- 組織標準の自動適用
- 品質の統一化

# 9. Copilot Prompt Files

## プロンプトファイルの活用

---

### 機能

- 再利用可能なプロンプトの定義
- プロジェクト固有の設定
- チーム間での共有

### 使用例

- `.github/prompts/foo-prompts.md`
- テンプレートの定義
- ベストプラクティスの共有



# 10. Copilot Pull Request Summaries

## PR要約の自動生成

---

### 機能

- PR変更内容の自動要約
- 影響範囲の分析
- レビューア向けの説明生成

### メリット

- レビュー時間の短縮
- 変更内容の明確化
- コミュニケーションの改善

# 11. Copilot Text Completion

## テキスト補完支援

---

### 機能

- PR説明文の迅速・正確な作成支援

### メリット

- 一貫性のある文書作成
- 時間の節約
- 品質の向上

# 12. GitHub Copilot Extensions

## 外部ツールの統合

---

### 機能

- 外部ツールのCopilot Chatへの統合
- サードパーティサービスとの連携
- カスタム拡張の利用

### 対応ツール例

- Docker
- Azure
- データベース管理ツール

# 13. Copilot Workspace

## 統合開発環境

---

### 機能

- IssueからPRまでの統合フロー
- コードの自動生成と検証
- レビューと改良の統合

### ワークフロー

1. Issueの理解
2. 実装計画の生成
3. コードの自動実装

# 14. Copilot Spaces

## コンテンツ管理・共有

---

### 機能

- 関連コンテンツの整理・集中化
- コンテキストの共有
- チーム間のナレッジ共有

### 用途

- プロジェクトドキュメントの管理
- コードスニペットの共有
- ベストプラクティスの蓄積

# 15. Copilot Knowledge Bases

## 組織知識の統合

---

### 機能

- 組織ドキュメントのコレクション作成・管理
- 内部ナレッジベースとの統合
- カスタムコンテキストの提供

### 用途

- 社内コーディング規約の適用
- ドメイン知識の活用
- プロジェクト固有情報の参照

# 管理者向け機能

## Business/Enterprise プラン

---

### アクセス管理

- メンバーのライセンス管理
- 組織レベルのポリシー設定

### 使用状況分析

- 利用統計の確認
- 効果測定

### セキュリティ

- ファイルの除外設定

# 学習コンテンツの活用方法

## オンライン教材

---

<https://agreeable-island-0c8e4d900.6.azurestaticapps.net/>

## 学習のポイント

- 正直学習コンテンツとしての精度は微妙
- 発表資料のリポジトリをクローンしてCopilot Chatで適宜質問・修正しながら実行するのがお勧め
- 別に完璧な教材なんて必要がない

## 発表資料

 <https://github.com/nuitsjp/all-of-github-copilot>



# 学習コンテンツの作成方法

## 効率的な作成フロー

### 1 公式ドキュメントの活用

- プロンプトで学習教材を自動生成

### 2 `#fetch` で直接参照

- URLを直接指定して効率化

### 3 Markdownでローカル保存

- コンテキスト長の節約
- Chrome拡張（Webpage to Markdown）の活用

# 学習コンテンツ作成のコツ

## 重要な考え方

### 完璧を求めない

- 「こんなんで十分」の精神
- 作り込みより実践を優先

### プロンプトに時間をかけない

- モデルが変われば最適解も変わる
- 学習時間を最優先に

## 参考資料

 <https://github.com/nuitsjp/all-of-github-copilot>

# まとめ

## 本日本日お伝えしたこと

---

### GitHub Copilotの全機能概要

- 15の主要機能
- プラン別の利用可能機能

### 効果的な学習方法

- 段階的アプローチ
- 実践的な練習

### 継続的な活用

- 学習コンテンツの活用

# ありがとうございました！

 **GitHub Copilotで開発を加速しましょう！**

---

## リソース

- 学習コンテンツ: <https://agreeable-island-0c8e4d900.6.azurestaticapps.net/>
- 公式ドキュメント: <https://docs.github.com/copilot>
- VS Code ドキュメント: <https://code.visualstudio.com/docs/copilot/>

 **Happy Coding with GitHub Copilot!**