

GitHub Universe Recap

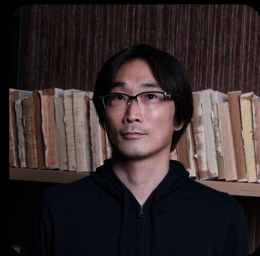
AIが支援する開発者プラットフォーム

@parroty

Kenta Nakase / 中瀬 健太

シニアソリューションズエンジニア

GitHub Japan



UNIVERSE23



AI-powered developer platform



COPILOT



COLLABORATION



PRODUCTIVITY



SECURITY



SCALE



Powered by AI



GitHub Copilotによって
コーディングのスピードが
55%加速

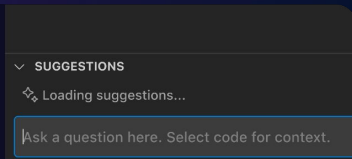


**75%の開発者の
満足度が向上**



20,000以上の
プロジェクトで
利用

コード全体の**46%**が
GitHub Copilotによって
提案されたコード*



**100万人以上の
開発者が利用**

```
def test_parse_expenses_with_empty_input(self):
    expenses_string = ''
    expected_output = []
    self.assertEqual(parse_expenses(expenses_string),
                      expected_output)

def test_parse_expenses_with_comments(self):
    expenses_string = '# This is a comment\n2023-01-02 -34.01 USD\n2023-01-03 2.59 DKK\n2023-01-03 -2.72 EUR'
    expected_output = [(datetime.datetime(2023, 1, 2),
                        -34.01, 'USD'),
                       (datetime.datetime(2023, 1, 3),
                        2.59, 'DKK'),
                       (datetime.datetime(2023, 1, 3),
                        -2.72, 'EUR')]
    self.assertEqual(parse_expenses(expenses_string),
                      expected_output)

if __name__ == '__main__':
    unittest.main()
```

The code assumes that the datetime module is imported. The test cases cover different scenarios such as valid input, empty input, input with comments, invalid date format, and invalid value format.

Analyze and debug

Ask a question or type '/' for topics

```
15         continue
16     date, value, currency = line.split(" ")
17     expenses.append((datetime.datetime.strptime(date, "%Y-%m-%d"),
18                   float(value),
19                   currency))
20
21     return expenses
22
23 expenses_data = '''2023-01-02 -34.01 USD
24                 2023-01-03 2.59 DKK
25                 2023-01-03 -2.72 EUR'''
26
27
28
29
30
31
```

Ln 17, Col 3 Spaces: 2 UTF-8 LF {} TypeScript

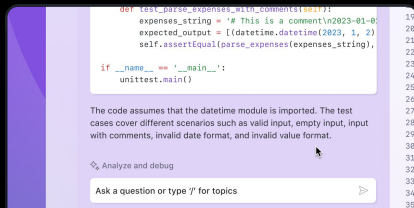
 GitHub Copilot Chat

一般提供開始

2023年12月



GitHub Copilot Business



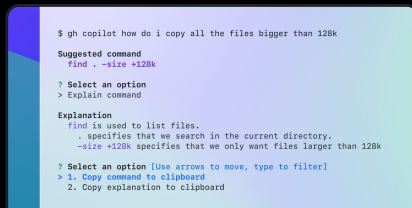
Copilot Chat in the IDE

PRIVATE BETA SET BY ORG 2023/11/8

PUBLIC BETA 2023年7月と9月

AIペアプログラマーとのチャットが可能に

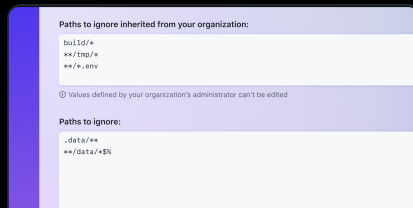
JetBrains系IDEサポートを開始。VS Code及びVisual Studioですでにパブリックベータで提供中



Copilot in the CLI

PUBLIC BETA 2023/11/8

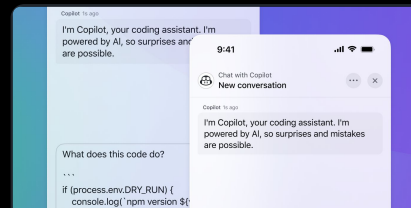
ターミナル内でコマンドの提案や、コマンドを説明



Copilot コンテンツ除外設定

PUBLIC BETA 2023/11/8 - 調整中

Organization / Repositoryに対して、特定のコンテンツをCopilotが使わないよう制御



Copilot in GitHub Mobile

ANNOUNCEMENT 2023/11/8

移動中でもCopilotにアクセス可能



Topics

利用シナリオ

対象ユーザー

コンテキスト / プロンプト

組織内での展開



利用シナリオ

実験・検証

- アイデア出し
- プロトタイピング

既存コード理解

- プロジェクトの解説
- アドホックな質問

新規コード生成

- 繰り返しパターンの生成
- 単体テストやドキュメントのコード生成

評価・修正

- トラブルシューティング
- 修正コードの提案

実験・検証 - (1)

Copilotと会話をしながらアプリケーション機能を拡張

bt-prototyping-sandbox [Codespaces: upgraded spac

@workspace TODOアプリケーションを作成したいです。タスクの一覧表示を

parrotly

@workspace TODOアプリケーションを作成したいです。タスクの一覧表示を実現するコンポーネントとメイン画面から呼び出す修正の実装を提案してください。メイン画面にはタスク一覧だけを表示するようにして、既存のテンプレートにある表示項目は削除してください。

GitHub Copilot used @workspace

Used 10 references

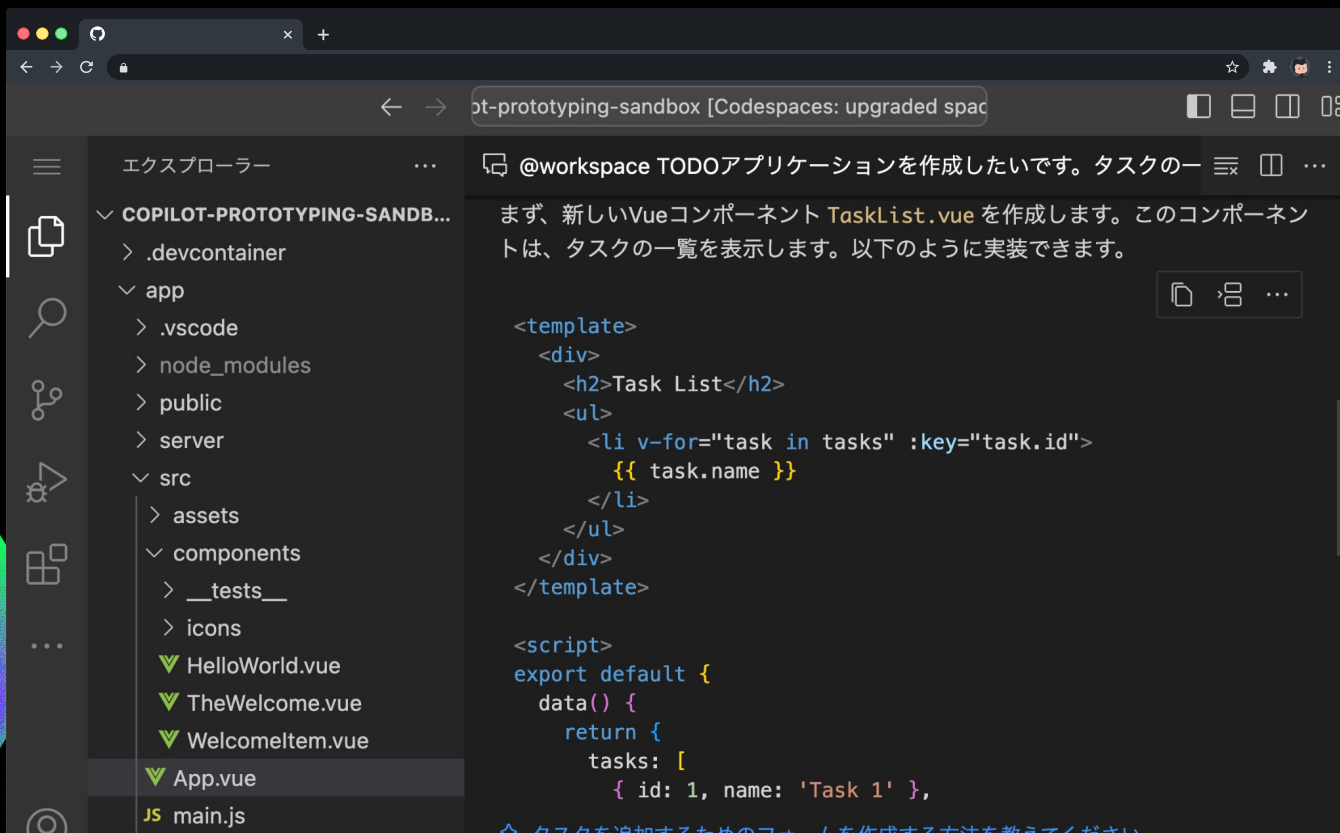
- index.html app
- README.md app
- index.js app/server
- App.vue app/src
- main.css app/src/assets
- HelloWorld.vue app/src/components

File Explorer:

- COPILOT-PROTOTYPING-SANDBOX
 - .devcontainer
 - app
 - .vscode
 - node_modules
 - public
 - server
 - src
 - assets
 - components
 - __tests__
 - icons
 - HelloWorld.vue
 - TheWelcome.vue
 - WelcomItem.vue
 - App.vue
 - main.js

実験・検証 - (2)

Copilotと会話をしながらアプリケーション機能を拡張



Explorer: エクスプローラー

- COPILOT-PROTOTYPING-SANDB...
- └ .devcontainer
- └ app
 - └ .vscode
 - └ node_modules
 - └ public
 - └ server
 - └ src
 - └ assets
 - └ components
 - └ __tests__
 - └ icons
 - └ HelloWorld.vue
 - └ TheWelcome.vue
 - └ Welcomeltem.vue
 - └ App.vue
 - └ JS main.js

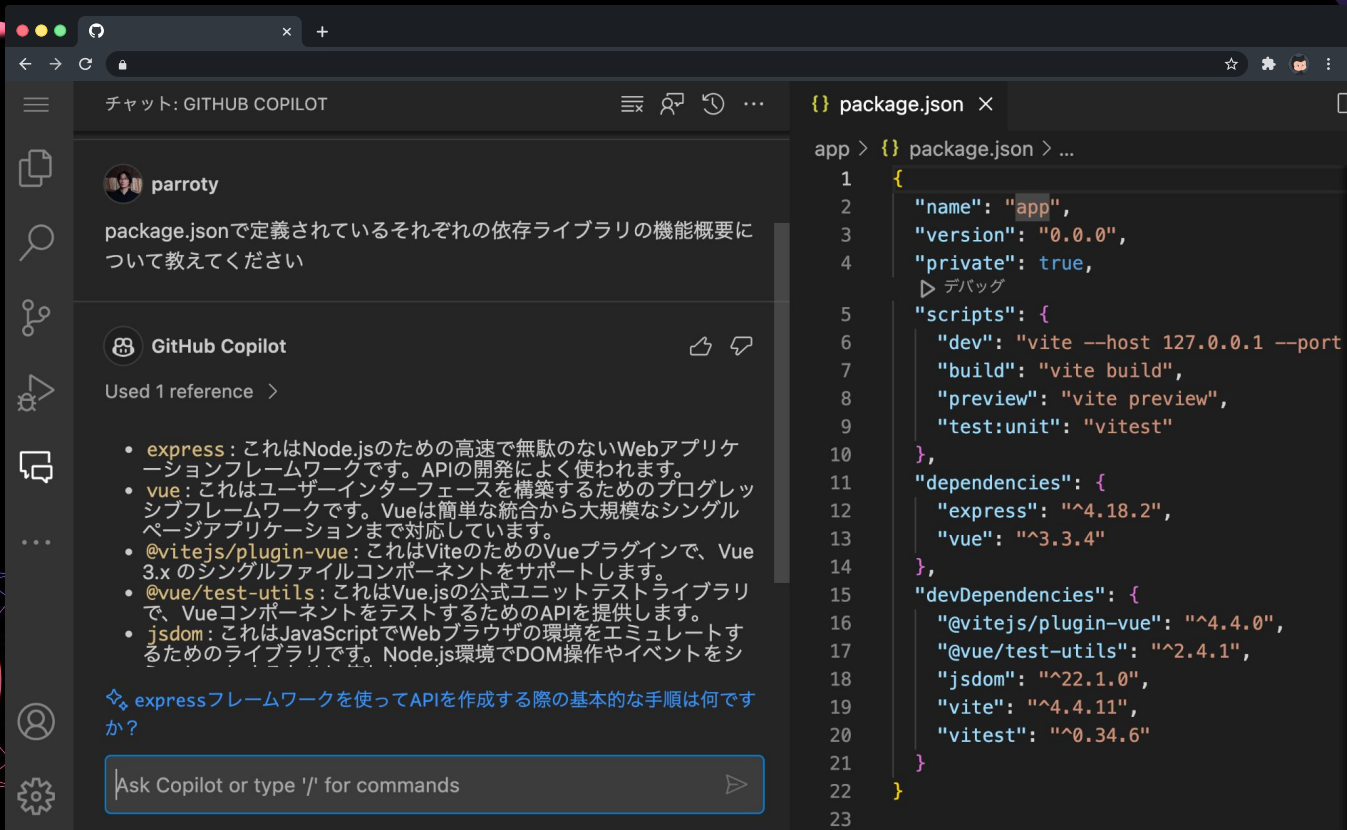
Code Editor: @workspace TODOアプリケーションを作成したいです。タスクの一覧を表示します。以下のよう

```
<template>
  <div>
    <h2>Task List</h2>
    <ul>
      <li v-for="task in tasks" :key="task.id">
        {{ task.name }}
      </li>
    </ul>
  </div>
</template>

<script>
export default {
  data() {
    return {
      tasks: [
        { id: 1, name: 'Task 1' },
      ]
    }
  }
}
```

既存コードの理解 - (1)

Copilotと会話をしながら、様々な疑問点を解消



The screenshot shows a web browser window with a chat interface on the left and a code editor on the right. The chat interface is titled "チャット: GITHUB COPILOT" and shows a conversation with a user named "parrotty". The user asks for an overview of dependencies in package.json. GitHub Copilot responds with a list of dependencies and their descriptions. The code editor shows the content of package.json, with the "dependencies" section highlighted.

チャット: GITHUB COPILOT

parrotty

package.jsonで定義されているそれぞれの依存ライブラリの機能概要について教えてください

GitHub Copilot

Used 1 reference >

- **express**: これはNode.jsのための高速で無駄のないWebアプリケーションフレームワークです。APIの開発によく使われます。
- **vue**: これはユーザーインターフェースを構築するためのプログレッシブフレームワークです。Vueは簡単な統合から大規模なシングルページアプリケーションまで対応しています。
- **@vitejs/plugin-vue**: これはViteのためのVueプラグインで、Vue 3.xのシングルファイルコンポーネントをサポートします。
- **@vue/test-utils**: これはVue.jsの公式ユニットテストライブラリで、VueコンポーネントをテストするためのAPIを提供します。
- **jsdom**: これはJavaScriptでWebブラウザの環境をエミュレートするためのライブラリです。Node.js環境でDOM操作やイベントをシ

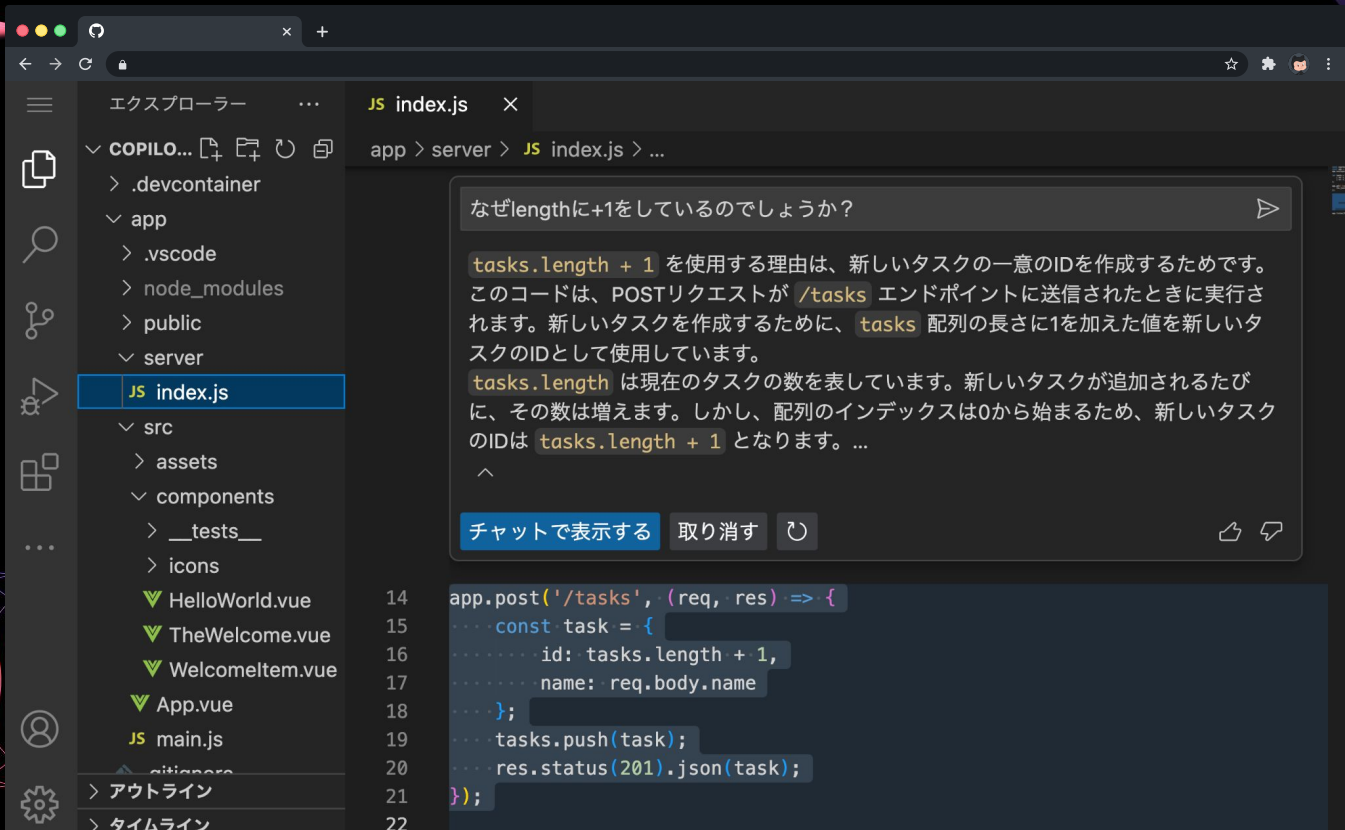
❖ expressフレームワークを使ってAPIを作成する際の基本的な手順は何か?

Ask Copilot or type '/' for commands

```
app > {} package.json > ...
1  {
2    "name": "app",
3    "version": "0.0.0",
4    "private": true,
5    "scripts": {
6      "dev": "vite --host 127.0.0.1 --port",
7      "build": "vite build",
8      "preview": "vite preview",
9      "test:unit": "vitest"
10   },
11   "dependencies": {
12     "express": "^4.18.2",
13     "vue": "^3.3.4"
14   },
15   "devDependencies": {
16     "@vitejs/plugin-vue": "^4.4.0",
17     "@vue/test-utils": "^2.4.1",
18     "jsdom": "^22.1.0",
19     "vite": "^4.4.11",
20     "vitest": "^0.34.6"
21   }
22 }
```

既存コードの理解 - (2)

Copilotと会話をしながら、様々な疑問点を解消



The screenshot shows a VS Code editor with a chat window open. The chat window contains the following text:

なぜlengthに+1をしているのでしょうか？

tasks.length + 1 を使用する理由は、新しいタスクの一意的IDを作成するためです。このコードは、POSTリクエストが /tasks エンドポイントに送信されたときに実行されます。新しいタスクを作成するために、tasks 配列の長さに1を加えた値を新しいタスクのIDとして使用しています。

tasks.length は現在のタスクの数を表しています。新しいタスクが追加されるたびに、その数は増えます。しかし、配列のインデックスは0から始まるため、新しいタスクのIDは tasks.length + 1 となります。...

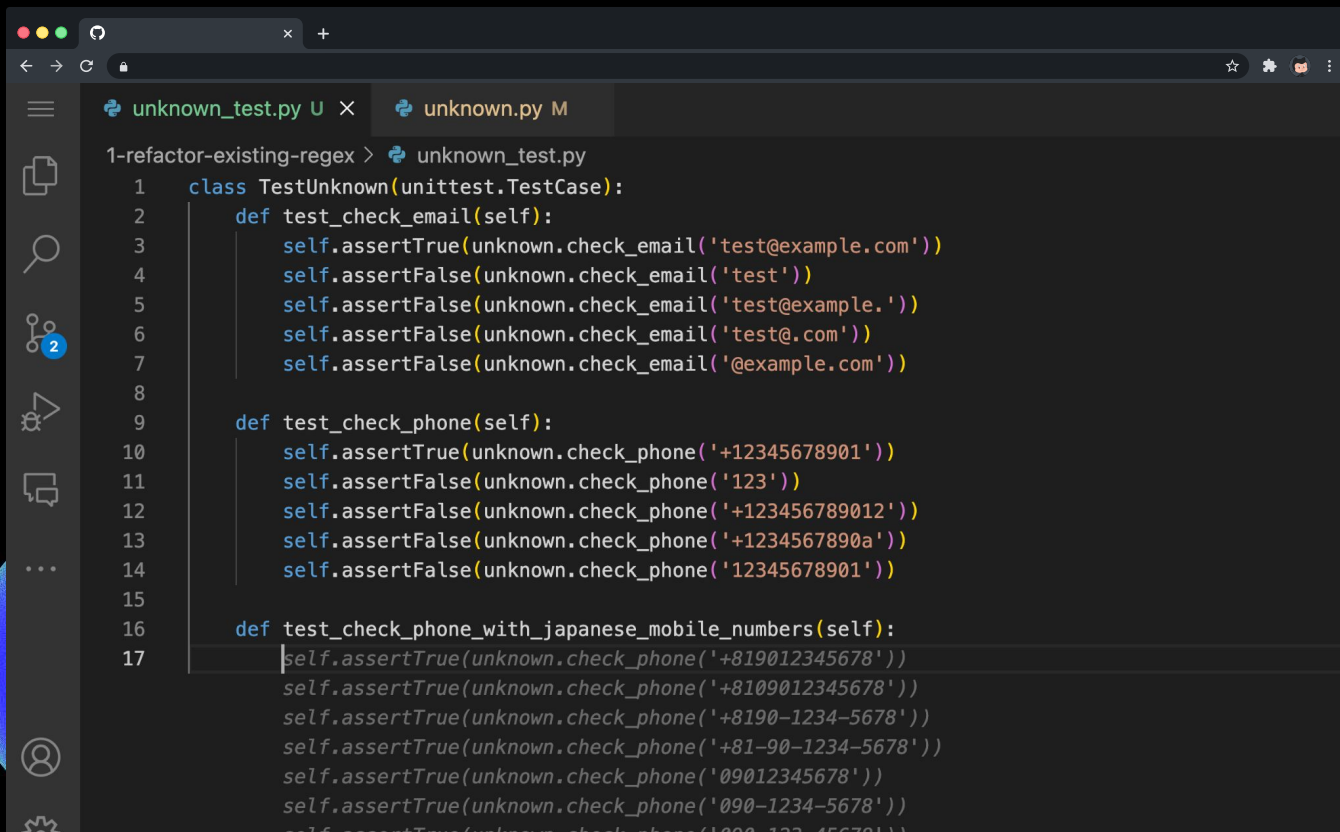
Buttons: チャットで表示する, 取り消す, 刷新

The code snippet in the editor below the chat window is:

```
14 app.post('/tasks', (req, res) => {
15   ... const task = {
16     ... id: tasks.length + 1,
17     ... name: req.body.name
18   };
19   ... tasks.push(task);
20   ... res.status(201).json(task);
21 });
22
```

新規コード生成

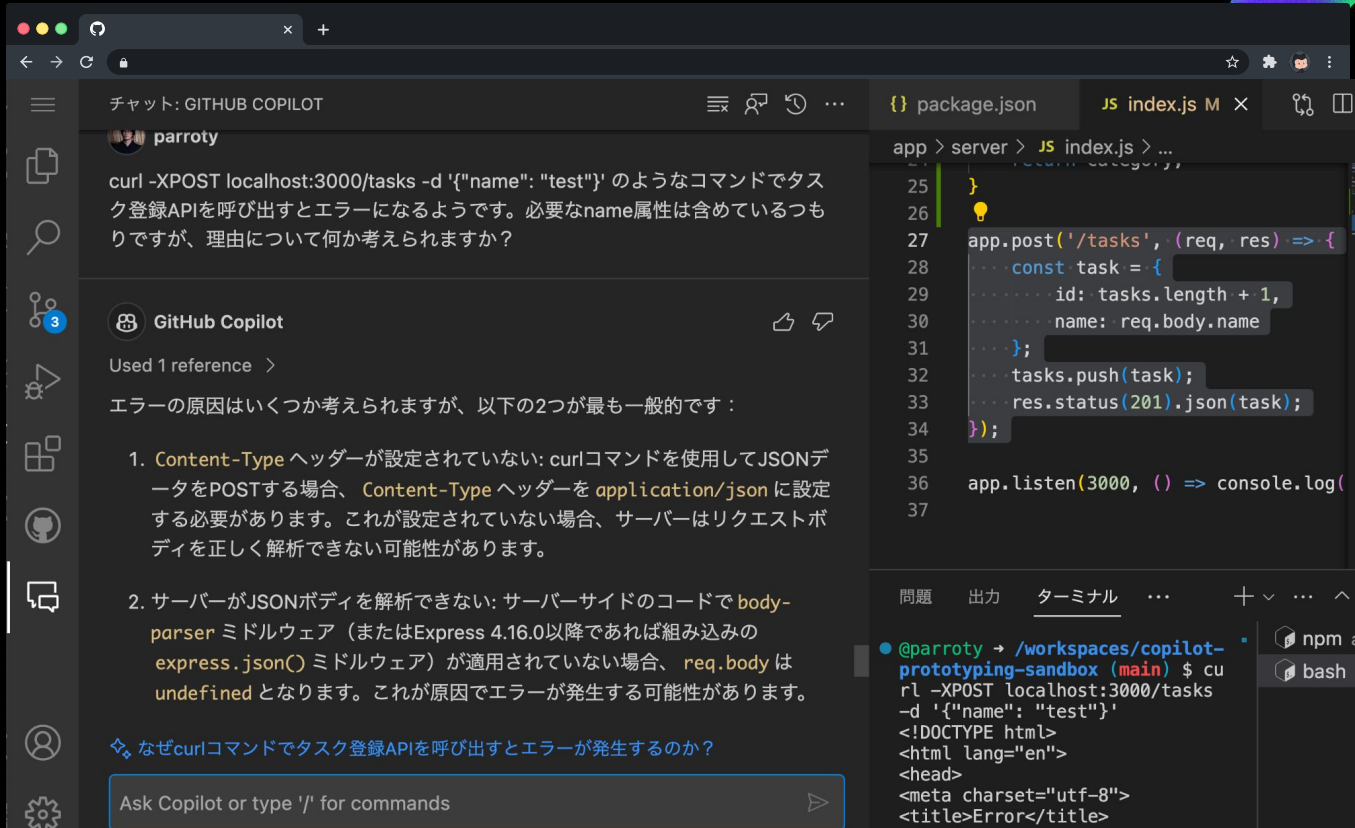
面倒な作業をCopilotに任せて、よりコアな作業に集中



```
1 class TestUnknown(unittest.TestCase):
2     def test_check_email(self):
3         self.assertTrue(unknown.check_email('test@example.com'))
4         self.assertFalse(unknown.check_email('test'))
5         self.assertFalse(unknown.check_email('test@example.'))
6         self.assertFalse(unknown.check_email('test@.com'))
7         self.assertFalse(unknown.check_email('@example.com'))
8
9     def test_check_phone(self):
10        self.assertTrue(unknown.check_phone('+12345678901'))
11        self.assertFalse(unknown.check_phone('123'))
12        self.assertFalse(unknown.check_phone('+123456789012'))
13        self.assertFalse(unknown.check_phone('+1234567890a'))
14        self.assertFalse(unknown.check_phone('12345678901'))
15
16    def test_check_phone_with_japanese_mobile_numbers(self):
17        self.assertTrue(unknown.check_phone('+819012345678'))
18        self.assertTrue(unknown.check_phone('+8109012345678'))
19        self.assertTrue(unknown.check_phone('+8190-1234-5678'))
20        self.assertTrue(unknown.check_phone('+81-90-1234-5678'))
21        self.assertTrue(unknown.check_phone('09012345678'))
22        self.assertTrue(unknown.check_phone('090-1234-5678'))
23        self.assertTrue(unknown.check_phone('1000-1234-5678'))
```

評価・修正 - (1)

様々なシナリオで試行錯誤のサイクルを短縮



The screenshot shows a VS Code editor interface. On the left, a chat window titled "チャット: GITHUB COPILOT" is open, showing a conversation with "parroty". The chat message asks for help with a curl command error. The Copilot response lists two common causes for the error: missing Content-Type header and an incompatible JSON parser. Below the list, a question is asked: "なぜcurlコマンドでタスク登録APIを呼び出すとエラーが発生するのか?". The code editor on the right shows a JavaScript file "index.js" with a REST API endpoint for creating tasks. The code includes a POST handler that pushes a new task to an array and returns a 201 status. The terminal at the bottom shows the command being executed: "curl -XPOST localhost:3000/tasks -d '{"name": "test"}'".

チャット: GITHUB COPILOT

parroty

curl -XPOST localhost:3000/tasks -d '{"name": "test"}' のようなコマンドでタスク登録APIを呼び出すとエラーになるようです。必要なname属性は含めているつもりですが、理由について何か考えられますか？

GitHub Copilot

Used 1 reference >

エラーの原因はいくつか考えられますが、以下の2つが最も一般的です：

1. Content-Type ヘッダーが設定されていない: curlコマンドを使用してJSONデータをPOSTする場合、Content-Type ヘッダーを application/json に設定する必要があります。これが設定されていない場合、サーバーはリクエストボディを正しく解析できない可能性があります。
2. サーバーがJSONボディを解析できない: サーバーサイドのコードで body-parser ミドルウェア (またはExpress 4.16.0以降であれば組み込みの express.json() ミドルウェア) が適用されていない場合、req.body は undefined となります。これが原因でエラーが発生する可能性があります。

なぜcurlコマンドでタスク登録APIを呼び出すとエラーが発生するのか？

Ask Copilot or type '/' for commands

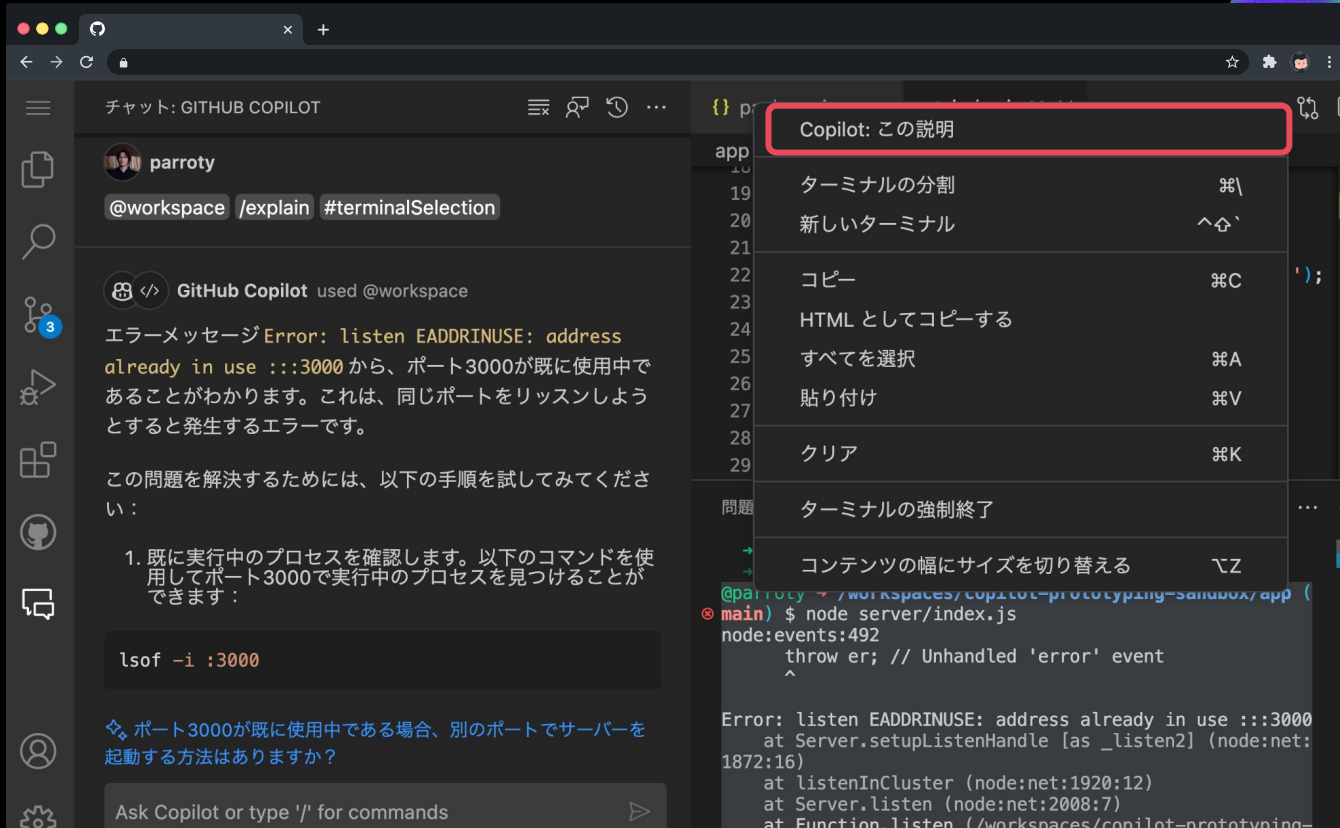
```
app > server > JS index.js > ...
25 }
26
27 app.post('/tasks', (req, res) => {
28   ... const task = {
29     ... id: tasks.length + 1,
30     ... name: req.body.name
31   };
32   ... tasks.push(task);
33   ... res.status(201).json(task);
34 });
35
36 app.listen(3000, () => console.log(
37
```

問題 出力 ターミナル ... + v ... ^

```
@parroty → /workspaces/copilot-prototyping-sandbox (main) $ curl -XPOST localhost:3000/tasks -d '{"name": "test"}'
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Error</title>
```

評価・修正 - (2)

様々なシナリオで試行錯誤のサイクルを短縮



The screenshot shows a VS Code interface with a chat window on the left and a terminal window on the right. The chat window displays a conversation with GitHub Copilot regarding an error message. The terminal window shows the execution of a Node.js server and the resulting error.

Chat Window:

チャット: GITHUB COPILOT

parroty

@workspace /explain #terminalSelection

GitHub Copilot used @workspace

エラーメッセージ `Error: listen EADDRINUSE: address already in use :::3000` から、ポート3000が既に使用中であることがわかります。これは、同じポートをリッスンしようとすると発生するエラーです。

この問題を解決するためには、以下の手順を試してみてください：

- 既に実行中のプロセスを確認します。以下のコマンドを使用してポート3000で実行中のプロセスを見つけることができます：

```
lsof -i :3000
```

◆ ポート3000が既に使用中である場合、別のポートでサーバーを起動する方法はありますか？

Ask Copilot or type '/' for commands

Terminal Window:

```
@parroty /workspaces/copilot-prototyping-sandbox/app (main) $ node server/index.js
node:events:492
  throw er; // Unhandled 'error' event
  ^

Error: listen EADDRINUSE: address already in use :::3000
    at Server.setupListenHandle [as _listen2] (node:net:1872:16)
    at listenInCluster (node:net:1920:12)
    at Server.listen (node:net:2008:7)
    at Function.listen (/workspaces/copilot-prototyping-
```

UPDATES

October 2023

September 2023

August 2023

July 2023

June 2023

May 2023

April 2023

March 2023

February 2023

January 2023

November 2022

October 2022

September 2022

August 2022

July 2022

June 2022

May 2022

April 2022

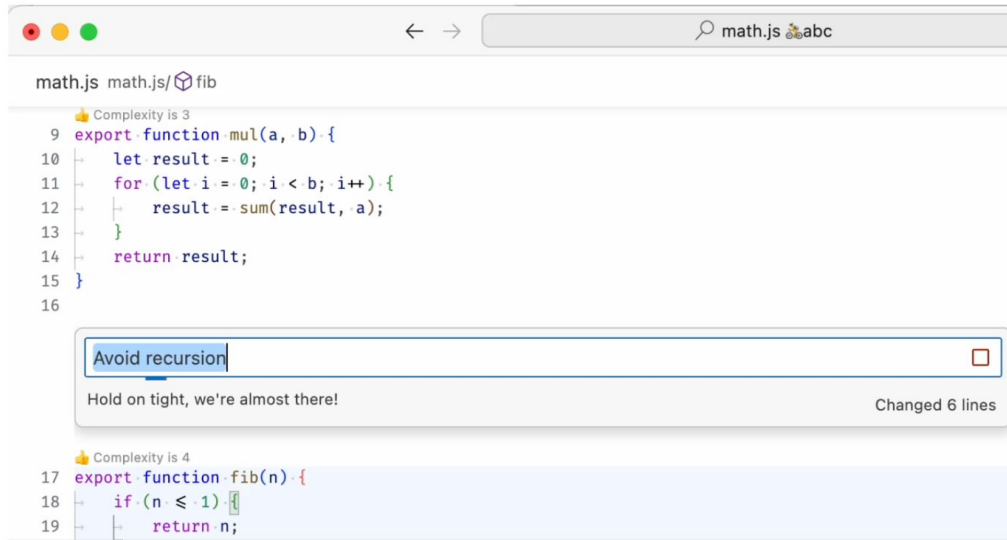
March 2022

Contributions to extensions

GitHub Copilot

Streaming inline chat

The [GitHub Copilot Chat](#) extension's inline chat can now make progressive text edits and "types" at the rate at which a response being received. This is a more natural experience than the previous behavior of waiting for the entire chat response to be received before applying it to the editor.



IN THIS UPDATE

- Accessibility
- Workbench
- Editor
- Source Control
- Notebooks
- Debug
- Remote Development

{ Contributions to extensions

- Preview Features
- Extension authoring
- Proposed APIs
- Engineering
- Extensions and documentation
- Notable fixes
- Thank you

- Subscribe
- Ask questions
- Follow @code
- Request features
- Report issues
- Watch videos



対象ユーザー

👤 Junior Developer

- 短い学習サイクル
- 疑問点の具体化・明確化

👤⁺ Senior Developer

- 定形作業の簡略化
- より難しい課題への集中

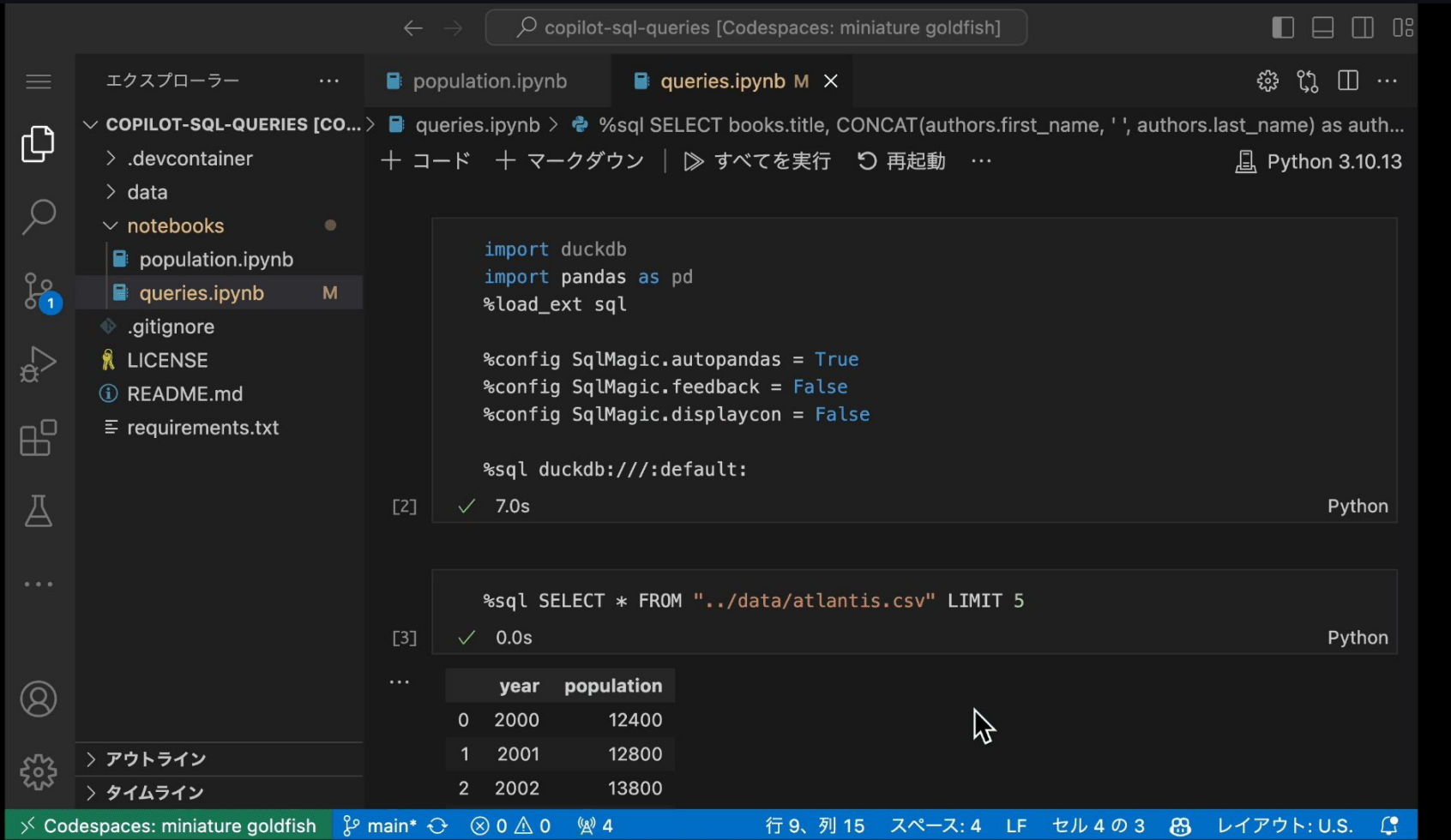
📄 Data Scientist

- Jupyter Notebook
- SQL Queries

🖥️ System Administrator

- CLI Command
- Infrastructure as Code

Data Scientist - Jupyter Notebook



← → copilot-sql-queries [Codespaces: miniature goldfish]

エクスプローラー ... population.ipynb queries.ipynb M ×

COPILOT-SQL-QUERIES [CO...]

- > .devcontainer
- > data
- > notebooks
 - population.ipynb
 - queries.ipynb M
- ◆ .gitignore
- 👤 LICENSE
- 📄 README.md
- ☰ requirements.txt

```
queries.ipynb > %sql SELECT books.title, CONCAT(authors.first_name, ' ', authors.last_name) as auth...
```

+ コード + マークダウン | ▶️ すべてを実行 🔄 再起動 ... Python 3.10.13

```
import duckdb
import pandas as pd
%load_ext sql

%config SqlMagic.autopandas = True
%config SqlMagic.feedback = False
%config SqlMagic.displaycon = False

%sql duckdb:///default:
```

[2] ✓ 7.0s Python

```
%sql SELECT * FROM "../data/atlantis.csv" LIMIT 5
```

[3] ✓ 0.0s Python

...

	year	population
0	2000	12400
1	2001	12800
2	2002	13800

Codespaces: miniature goldfish main* 0 0 4 行 9、列 15 スペース: 4 LF セル 4 の 3 レイアウト: U.S.

System Administrator - Ansible

The screenshot shows a VS Code interface with a dark theme. At the top, the browser address bar shows 'laC-copilot [Codespaces: miniature space zebra]'. The main window is split into two panes. The left pane is titled 'チャット: GITHUB COPILOT' and contains a chat window with the GitHub Copilot logo and the text: 'こんにちは @parroty さん。何かお役に立てることはありますか?'. Below the chat are several 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'. At the bottom of the chat is an input field with the placeholder text 'Ask Copilot or type '/' for commands'. The right pane is titled '! deploy-nginx.yml U X' and shows an Ansible terminal window with the prompt 'ansible > ! deploy-nginx.yml'. The terminal output shows a line of code with a tooltip: '1 % I を押して、GitHub Copilot Chat に何らかの操作を依頼します。 入力を開始して閉じます。'. A green circle highlights the tooltip. At the bottom of the VS Code window, the status bar shows 'Codespaces: miniature space zebra', 'main*', '0 0 0', 'UTF-8 LF YAML', and 'レイアウト: U.S.'.

laC-copilot [Codespaces: miniature space zebra]

チャット: GITHUB COPILOT

GitHub Copilot

こんにちは @parroty さん。何かお役に立てることはありますか?

/fix コード内の問題

/tests コードの単体テストを追加する

/explain 選択したコードのしくみ

Ask Copilot or type '/' for commands

ansible > ! deploy-nginx.yml

1 % I を押して、GitHub Copilot Chat に何らかの操作を依頼します。 入力を開始して閉じます。

Codespaces: miniature space zebra main* 0 0 0 UTF-8 LF YAML レイアウト: U.S.

Introducing:

GitHub Copilot Enterprise

New SKU | GA 2024


GitHub Enterprise Cloudが必要です
Copilot Business に含まれるすべての内容に加えて以下が含まれます

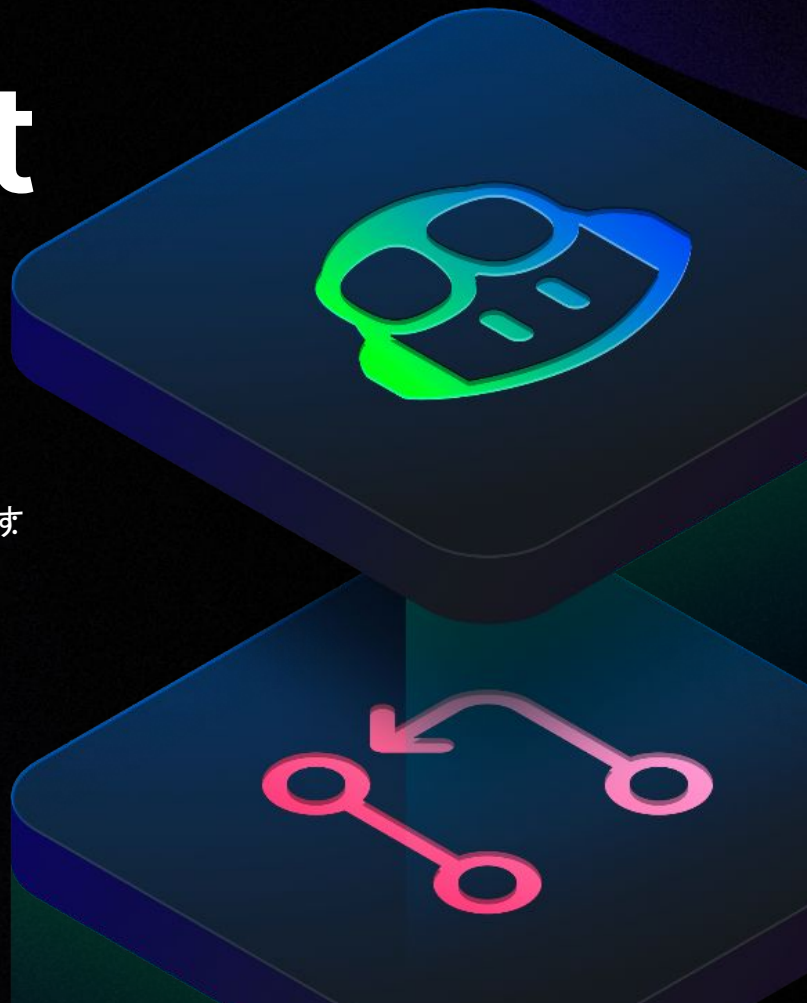
コードベースのパーソナライゼーション

ドキュメント検索

プルリクエスト要約

Copilot カスタマイゼーション

 Add-on





コンテキスト

Open Tabs

- アクティブな周辺コード
- 選択した特定のコード

Personalization / RAG (*)

- リポジトリ内検索による
コンテキスト拡充

Fine-Tuned Models

- リポジトリ内の情報を使っ
たAIモデルの
カスタマイズ

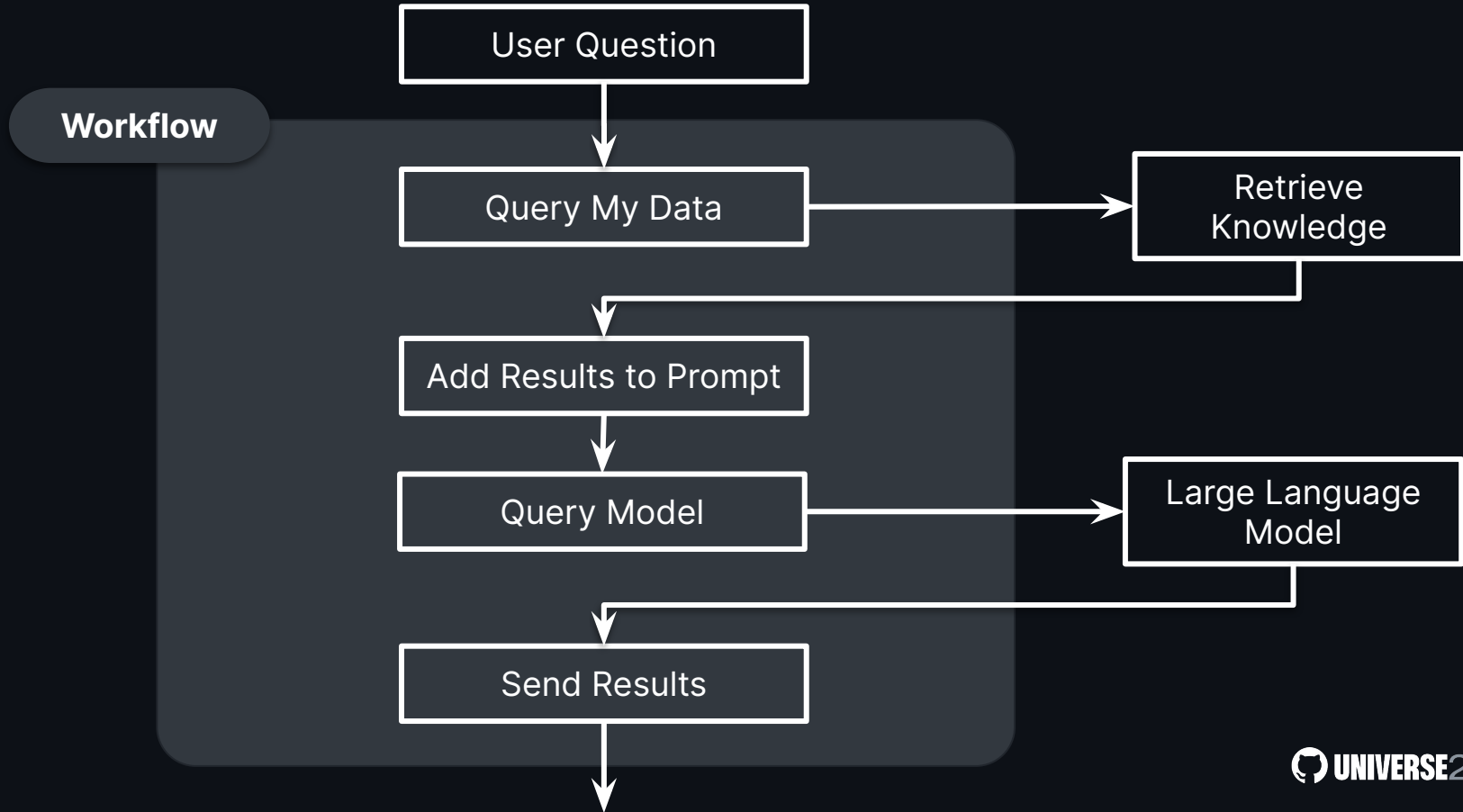
Partner Program

- GitHub外のデータによる
コンテキスト拡充

(*) Retrieval Augmented Generation

Note: Not associated with a product release

RAG (Retrieval Augmented Generation)





Fine-Tuned Models

- ✓ より適切かつ特化されたコードを提案
- ✓ バグの数を抑えてより速く開発が可能に
- ✓ コードの参照文脈・モデル・ホスト・推測は全部プライベート
- ✓ 独自の開発言語や流行でない開発言語に基づいたコード提案生成



Copilot Enterpriseが必要

Custom models

Mona Lisa

Created on 14 Jul

Status	Training data	Training time
Training	5 repositories	0s

Training steps

Estimated time to finish: 36h 21m

- ✓ **Preprocessing repositories**
Completed in 5h 32m
- Building training data**
Started 10 minutes ago · Estimated time to finish: 3h 05m
- **Model fine-tuning**
Waiting for building training data to finish
- **Model evaluation**
Waiting for model finetuning to finish
- **Deploying**
Waiting for model evaluation to finish



テクノロジーパートナー向け: Copilot Partner Program

プログラムへの申請をテクノロジーパートナー向けに公開

✔ Copilotのユースケースや性能を拡張

✔ Copilot向けの新たなプラグインを作成

✔ GitHub外に保存されたデータへ拡大





プロンプト クラフティング

Single, Specific, Short

- 単一の責任範囲
- 明確な意図
- 簡潔なレスポンス

The screenshot shows a browser window displaying a GitHub blog post. At the top left is the GitHub logo and the word 'Blog'. At the top right are 'Menu' and a search icon. Below the header, there are two category tags: 'Engineering' and 'Product'. The main title of the post is 'How to use GitHub Copilot: Prompts, tips, and use cases'. The introductory text reads: 'In this prompt guide for GitHub Copilot, two GitHub developer advocates, Rizel and Michelle, will share examples and best practices for communicating your desired results to the AI pair programmer.' Below the text is a large image featuring the GitHub Copilot logo (a purple robot head) centered within a circular gradient background of blue and green. At the bottom of the screenshot, the word 'Authors' is visible.



組織内での展開

検討

- 言語・フレームワーク
- プロジェクト・ユーザーの選定
(スモールスタート, etc.)

検証

- トライアル・デモ環境
(<https://gh.io/copilot-demo>)
- ハンズオン・ワークショップ
- メトリック・ヒアリング



Thank you