

週末プログラマーのためのGitHub Copilot / Copilot Chatの使い方 -- 『だまって、これだけやれ』

2024/07/31

研開/シスI

江端智一

1. GitHub Copilot / Copilot Chat とは (著者(江端)主観)

GitHub Copilot / Copilot Chat については、ChatGPT等で調べてもらえれば、すぐに分かることだと思いますので、細かい記載はバツサリ省略します。

私の"主観で記載させて貰うのであれば、『**GitHub Copilot / Copilot Chat を使わずにコーディングするなんて、正気か?**』というものです(本書リリース時点での所感)。

ただ、上記の感想は、「**私(江端)は30年くらい、コーディングで(も)飯を喰ってきた**」という実績が背景にあるので、もしかしたら的を外しているかもしれません。

GitHub Copilot / Copilot Chatは、間違ったコードや回答を**自信タツプリ**にしてくることがあります。ですので、それに対して『**それ、変だろう?**』と言い返せる力量は、必要かもしれません。

それでも —— 自分のエンジニアの人生の中で、もっとも優しく、かつ、的確に、技術指導してくれたのは、"GitHub Copilot / Copilot Chat"であった —— と、断言できると思います((以下、"GitHub Copilot / Copilot Chat"の総称として"Copilot"を使うことがあります))

弊社のどの社員(エンジニア)よりも、Copilotは優しかった。

なにしろ、Copilotの回答に対して、私は、『助けて頂いて、本当にありがとうございました』と、御礼のメッセージを返してしまっただけです。

それに対して、Copilotは、『どういたしまして。不明点があれば、また質問してください』と御礼の返礼をしてきました。

このような優しいメッセージに、私は涙が出そうになりました。

2. 本書の目的

本書は、「週末エンジニアのための」という題目を冠しています。これは会社の面倒くさい手順を通している時間と、私の勉強の為に費用対効果を天秤にかけて、個人的に、GitHub Copilot/ Copilot Chatを使うことにし、その時に作成したメモを纏めたものです。

(私は、(金額にもよりますが)技術の取得のために自腹を切るのは、あまり気にしない方です。その金額に見合う「ラク」が手に手に入るのであれば、私は構いません)。

それ故、本書は、さっさとGitHub Copilot/ Copilot Chatを使い始めたい人向けに作成されています。

3. 本書の方針

- 私(江端)のPC環境で、私のやってきたことを記載しているだけのものです。
- もっと良いやり方は(必ず)あると思います。それは気にせず記載しています。
- Visual Studio Codeなど、普段、私が使い倒しているものについての説明は省きます。

4. 前提とする(私(江端)と同じ)環境

この環境に合わせる必要はありませんが、参考にして下さい。もっとも重要な点は、**社内システムからの使用を想定していない**です。ですので、社内から利用する際に必要となりそうな、プロキシ等の設定作業などは、全く触れていません。

私(江端)の環境は、

1. 自宅のPCでインターネット環境を、使い倒している
2. Githubのアカウントをすでに持っていて、使い倒している
3. インターネットに接続している Windows10 Box または "ラズパイ4"を合計4台持っていて、使い倒している
4. 全てのマシンでVisual Studio Code(以下、"VSCode"と称す)をインストール済みで、使い倒している

です。

5. 今の時点で「面倒くさいなあ」「分からんなあ」と思えるかもしれないこと

5.1. GitHub Copilot と Copilot Chat の違いとは何か

5.1.1. GitHub Copilot

- 目的: 開発者がコードを書くのを補助するためのツール
- 機能: コードの補完や提案を行う。開発者が入力したコードやコメントに基づいて次のコード行や関数全体を予測・提案する

5.1.2. Copilot Chat

- 目的: 開発者がコードに関する質問をしたり、説明を求めたり、コードのデバッグや理解を助けるための対話型ツール
- 機能: チャット形式でのインタラクションを通じて、コードに関する質問に答える。特定のコードの説明を提供したり、コードのエラーを解決するための提案を行う

ということになっています。

簡単に言えば"GitHub Copilot"は、コード作成中に、勝手にコードを提案して来たりします(**いきなり20行くらい提案してくることもある**)。しかも、これから自分で作成しようと思っていたものを、**先回りして作ってくれる**、というものです。

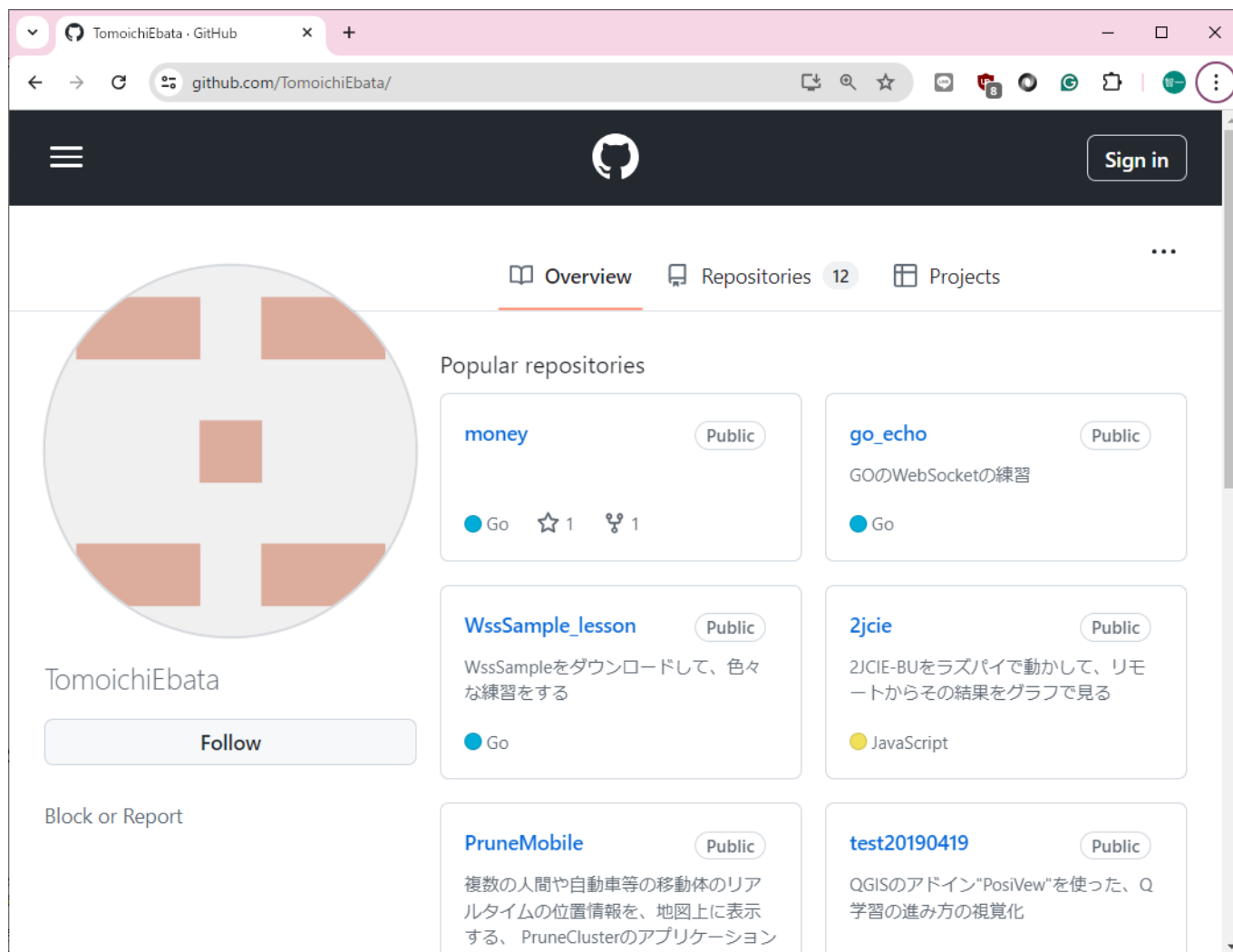
比して、"Copilot Chat"は、**ChatGPTと同じように、言語で質問するもの**、という理解で良いと思います。

5.2. 「Githubのアカウントを持っていること」とはどういうことか

GitHubとは、ソフトウェア開発者がコードをホスティング、管理、共有するためのウェブベースのプラットフォームです。

簡単に言えば「**そこそこ使えるようになった自作のソースコードを、自分の為に退避させておく、または、他人にも使えるようにする、公開用のソースコード格納するサイト**」と思っておけば、大丈夫でしょう。

私(江端)のGitHubのWebサイトは、こんな感じになっています



まずは、自分用のGitHubのサイトは、自力で作成して下さい。ともあれ、**これがないと、GitHub Copilot / Copilot Chatを始めることができません。**

最近、GitHubは、2段階セキュリティがデフォルト(自分のスマホの使用が必須)となっていて、これもかなり面倒です。

新しいPCのVSCodeにGitHub Copilot / Copilot Chat を VSCodeの拡張機能としてインストールする度に、スマホに飛んでくる認証コードを入力しなければなりません。かなり面倒ですが、諦めて下さい。

5.3. VSCodeの他の拡張機能が、「質問の入力」や「コードの確定」を邪魔をしてくるかもしれないかも

私の場合、「GitHub Copilotをvscodeにアドインしたが、tabキーを押しても提案を採用できない」などの問題が発生しました。

この件についての詳細は、

<https://wp.kobore.net/%e6%b1%9f%e7%ab%af%e3%81%95%e3%82%93%e3%81%ae%e6%8a%80%e8%a1%93%e3%83%a1%e3%83%a2/post-13786/> に記載しています。

これ以外についても、色々問題が出てくるかもしれません(というか、出てくだろう)が、自分専用Copilotにするために苦労しながら設定して下さい。

5.4. お金はかかります

個人用ですと**10ドル/月**です(リリース時現在)

Copilot Individual : 個人向けプラン Copilot Business : 企業向けプラン Copilot Enterprise : エンタープライズ向けプラン

The screenshot shows the GitHub Copilot pricing page. It is divided into two main sections: 'For organizations' and 'For individuals'.

For organizations

- Copilot Business** (Recommended): Copilot in the coding environment. \$19 per user / month. Features include: Code completions, Chat in IDE¹ and Mobile², CLI assistance³, Security vulnerability filter, Code referencing, Public code filter, IP indemnity, and Enterprise-grade security, safety, and privacy. Buttons: Buy now >, Contact sales >.
- Copilot Enterprise** (Available Feb 2024): Copilot personalized to your organization throughout the software development lifecycle. Requires GitHub Enterprise Cloud. \$39 per user / month. Features include: Everything in Copilot, plus: Chat personalized to your codebase, Documentation search and summaries, Pull request summaries, Code review skills, and Fine-tuned models⁴. Button: Join waitlist >.

For individuals

- Copilot Individual**: Code completions, Chat, and more for indie developers and freelancers. \$10 per month / \$100 per year. Button: Start a free trial >. Note: Free for verified students, teachers, and maintainers of popular open source projects.

会社が提供しているところもあります(弊社は提供しているようです)が、週末エンジニアが自分のOSS開発用に使うことはできませんので、私(江端)は自腹を切っています。

5.5. いくつかのVSCodeを同時に使うことができます。

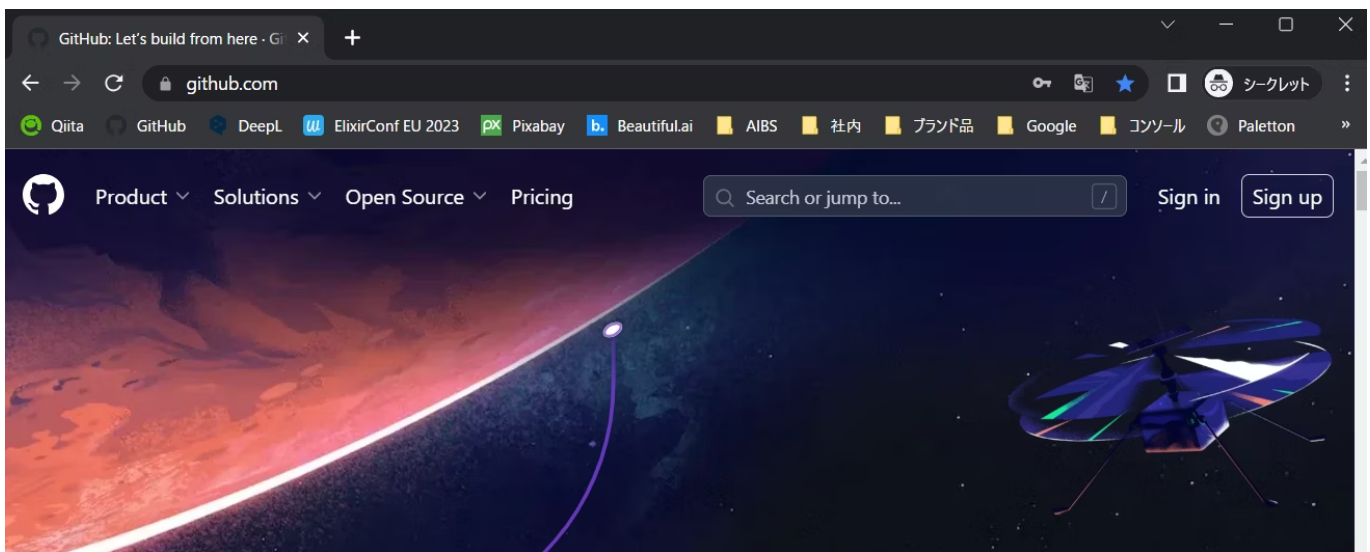
私、自分のPC2台と、ラズパイ4、2台でVSCodeを使っていますが、それぞれに、Copilotをインストールして使い倒しています。

私は、4台同時に使っています。

6. Copilot環境構築

6.1. GitHub アカウントの作成

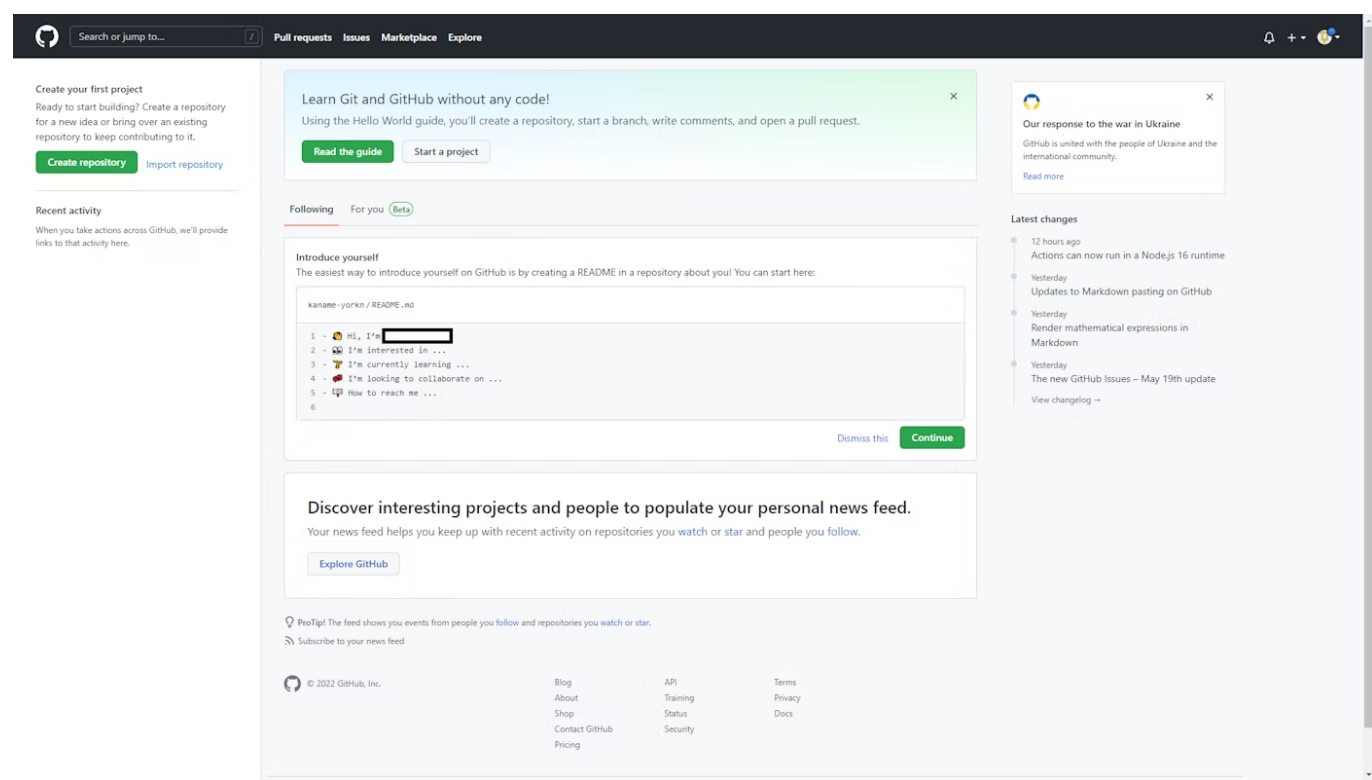
前述のように、GitHub アカウントが必要となります。itHub にアクセスし、Sign up からアカウントを作成してください(アカウントを持っている人はスキップ)。



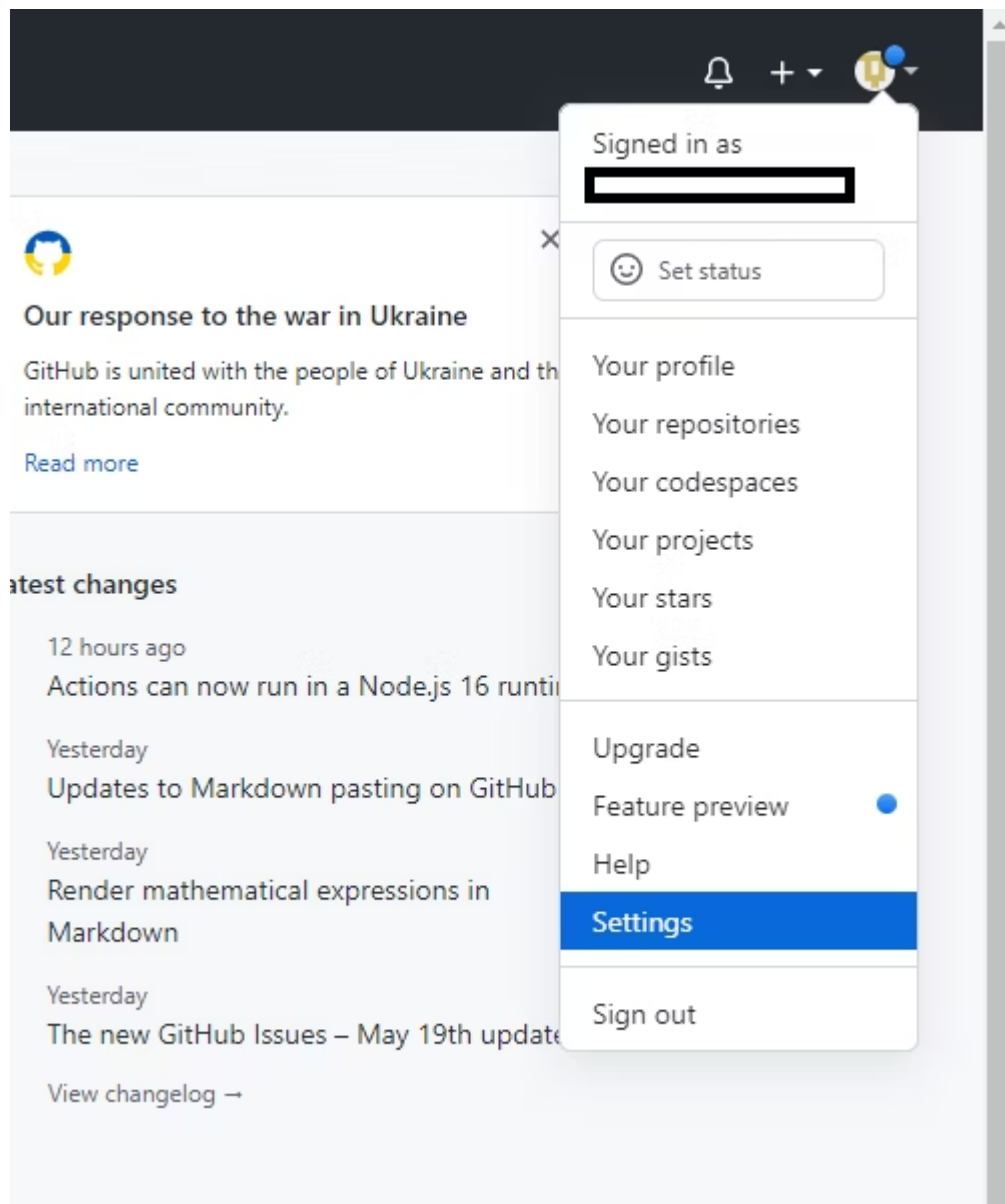
6.2. GitHubサイトにログイン

認証は、2要素認証をデフォルトにしておいて下さい。ここで手を抜くと、後で苦労します(私は苦労しました)。

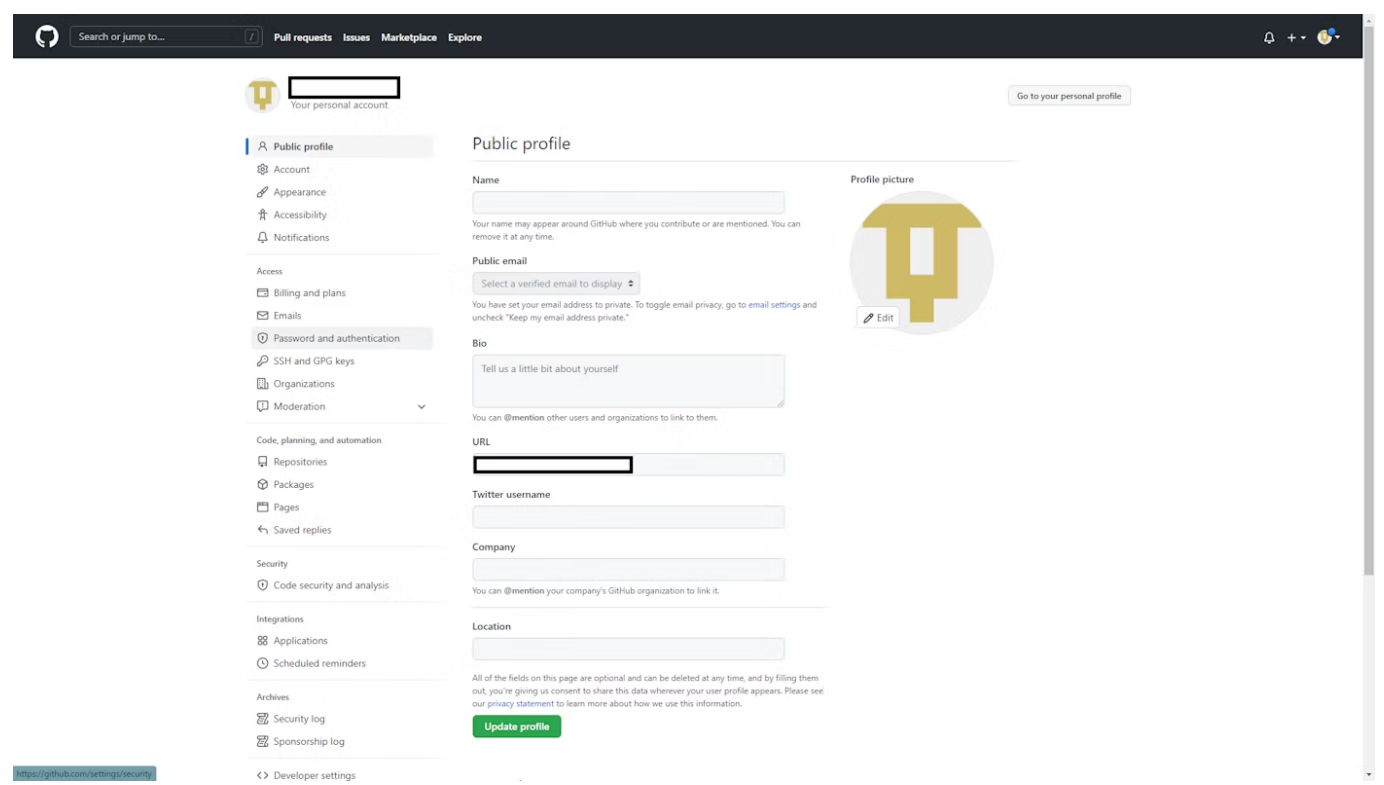
凄く重要なことですが、**ここで設定したパスワードは絶対に忘れないようにして下さい**。これからVSCodeにインストールする度に、(1)あなたのスマホ(電話番号の登録)と、(2)このパスワードが、必要となります。



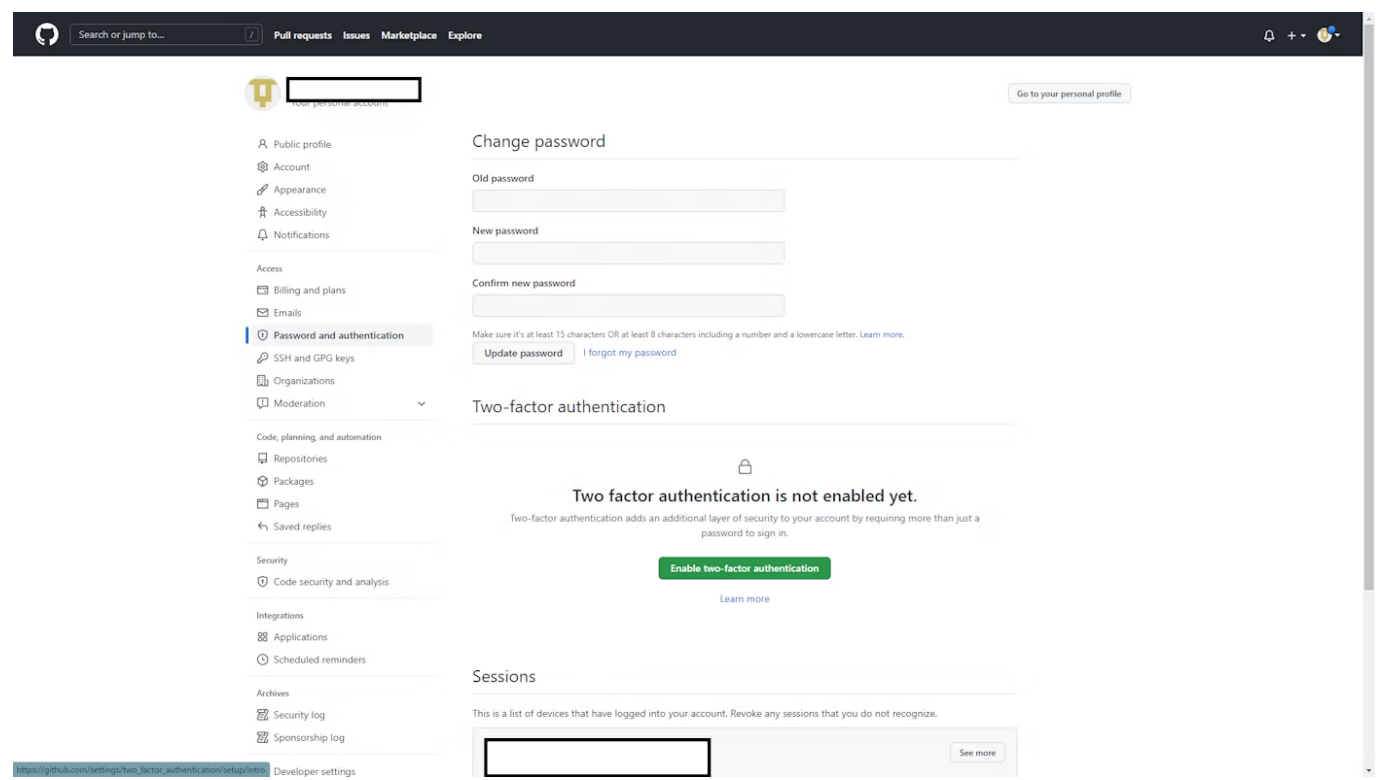
右上のメニューから「Settings」をクリックして下さい。



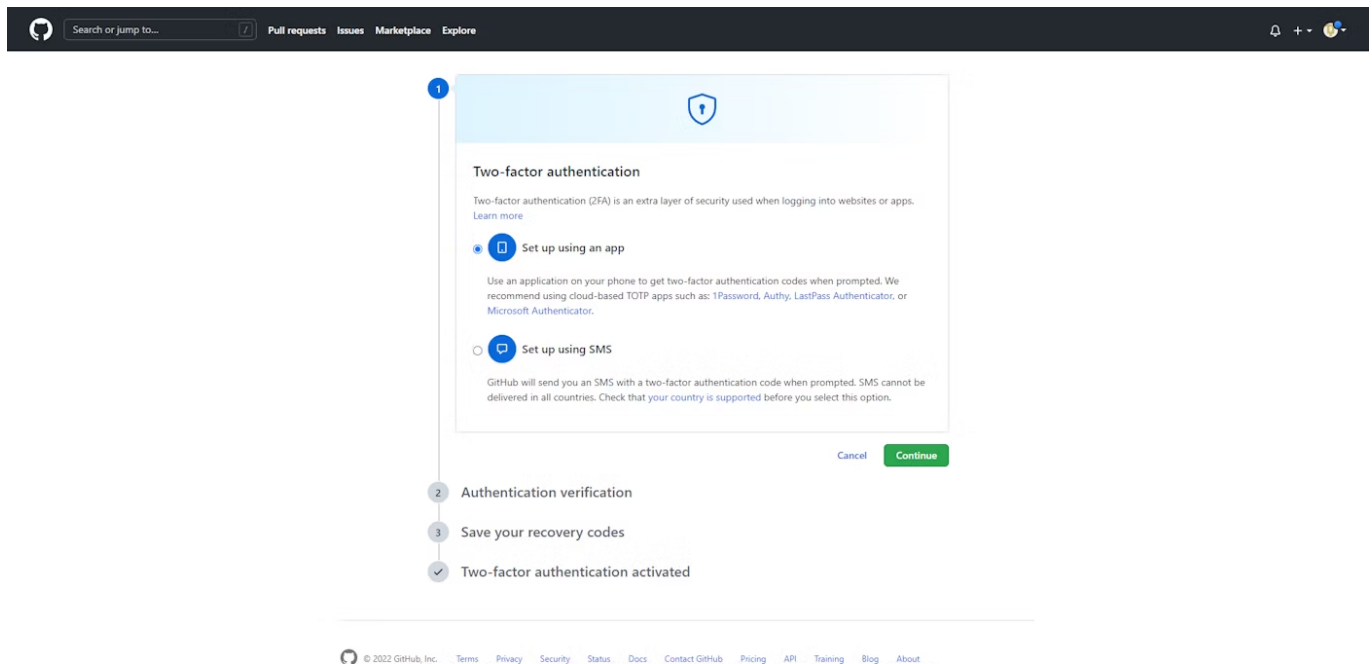
「Password and authentication」をクリックして下さい。



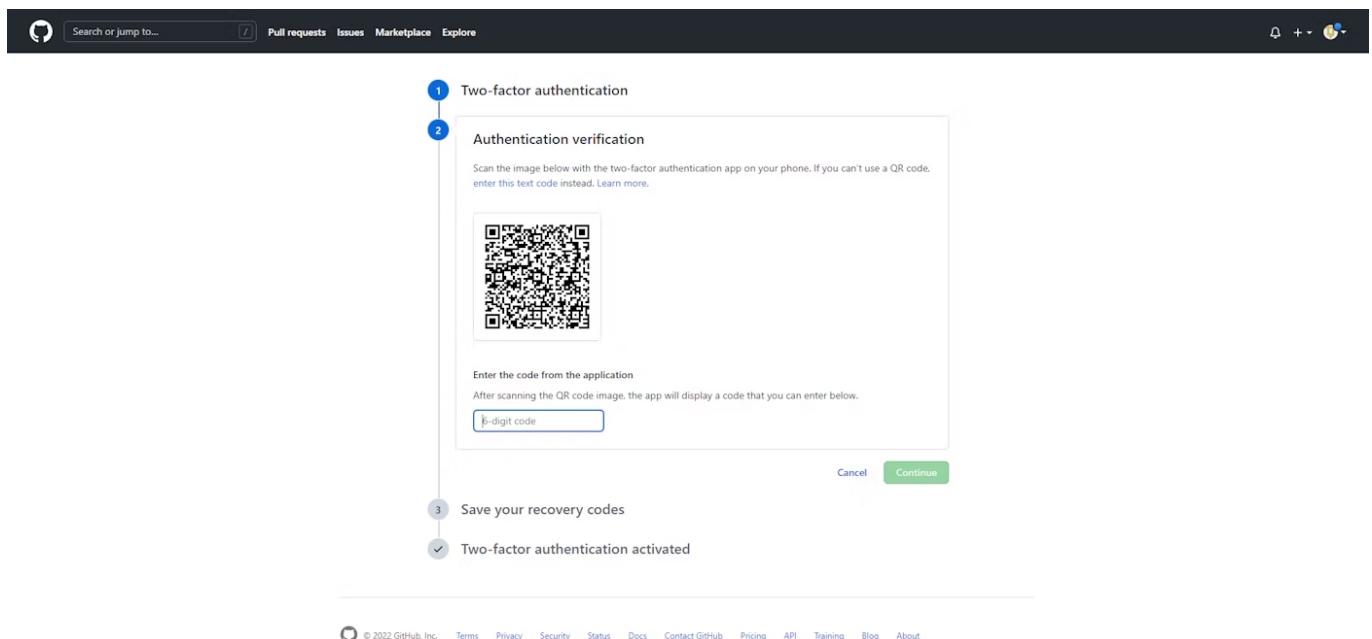
「Enable two-factor authentication」をクリックして下さい。



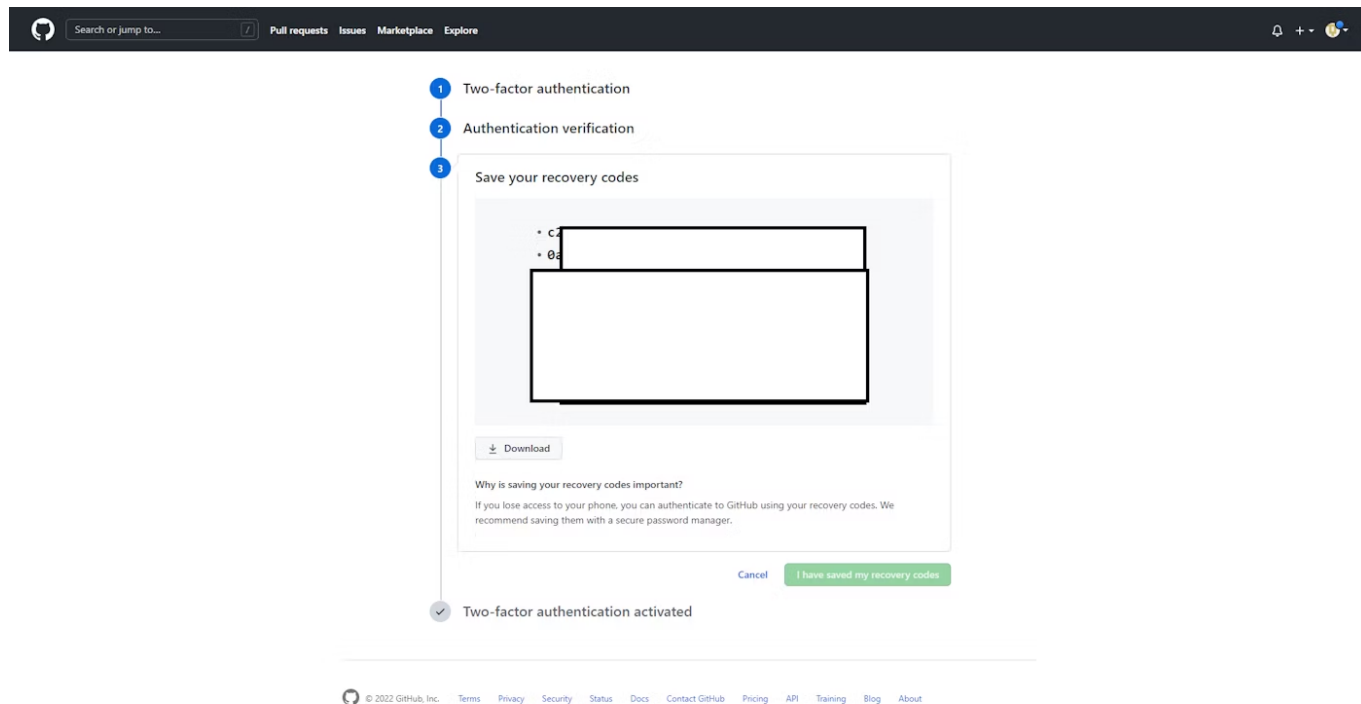
「Set up using an app」を選択した状態で、「Continue」をクリックしてください。



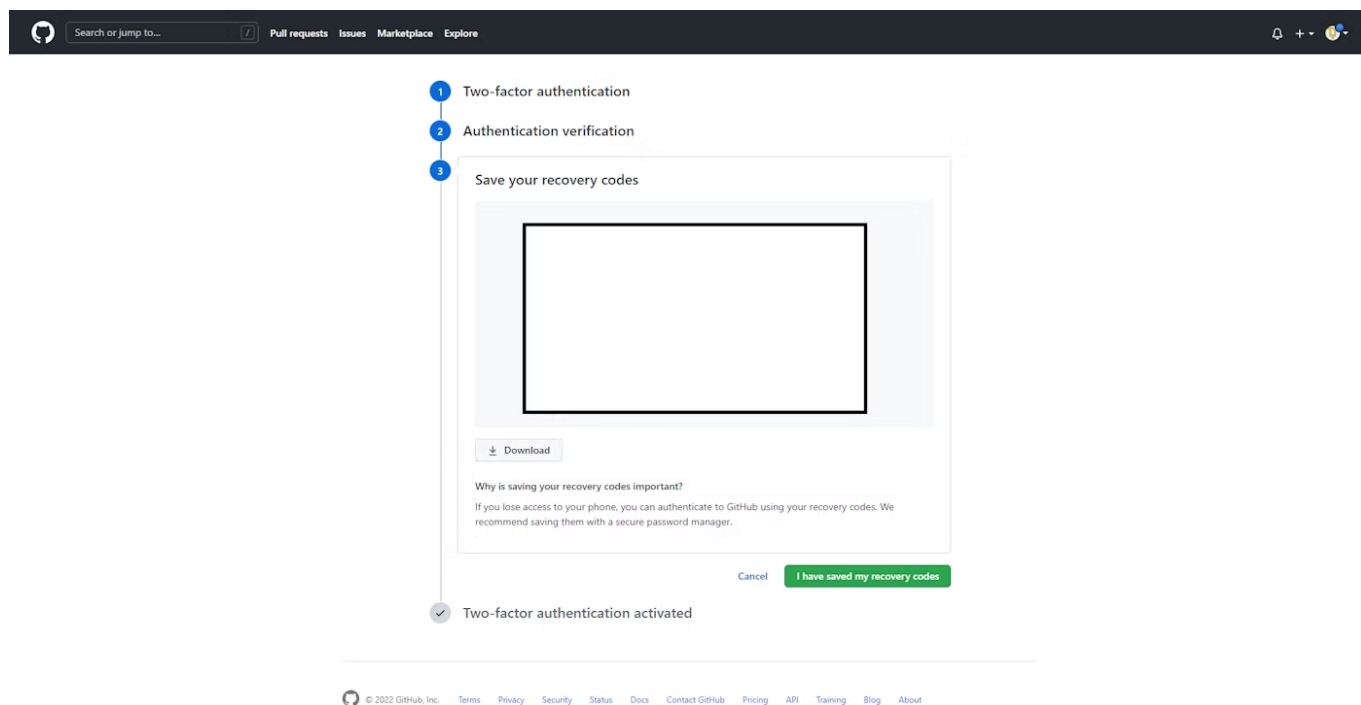
スマートフォンの認証アプリを用いてバーコードを読み取り、表示されたコードをブラウザに入力してください。(私はiPhoneのカメラを使っただけ)



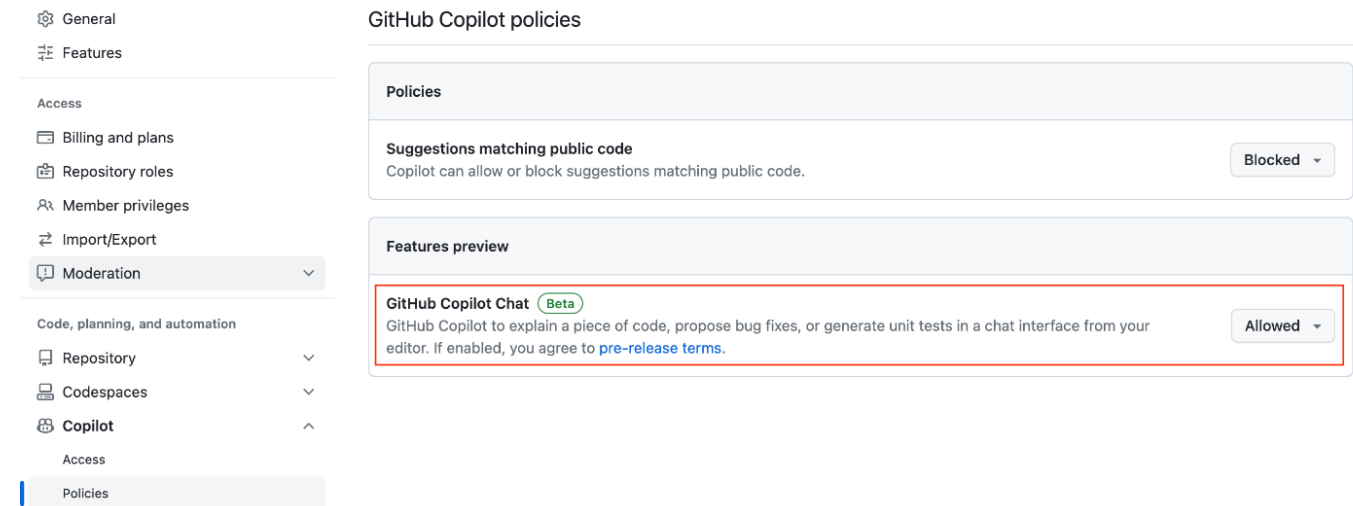
「Download」をクリックし、リカバリーコードのダウンロードを行ってください。



「I have saved my recovery codes」をクリックして、最後に「Done」をクリックして設定は終了します。



最後に、GitHubの「settings/copilot/policies」 から GitHub Copilot Chatを有効化して下さい。



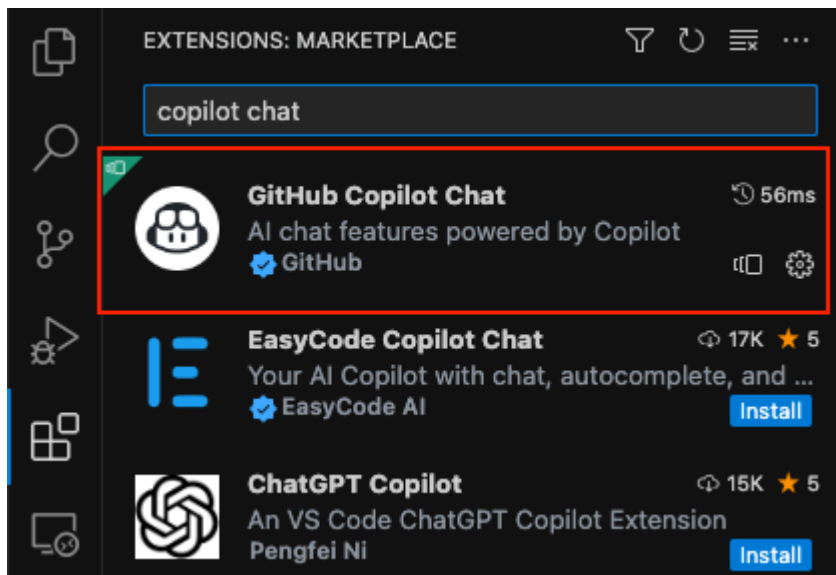
GitHubサイトの作業は、これで終了です。

6.3. VSCodeへの"GitHub Copilot / Copilot Chat"のインストール

VScodeを立ち上げて、拡張機能の「GitHub Copilot」をインストールして下さい。

因みに、拡張機能の「**日本語パック Japanese Language Pack for Visual Studio Code**」をインストールしておくと、ラクできる(英語を読まなくてすむ)と思います。

ちなみに「GitHub Copilot」をインストールすると、「Copilot Chat」も自動的にインストールされるようです。



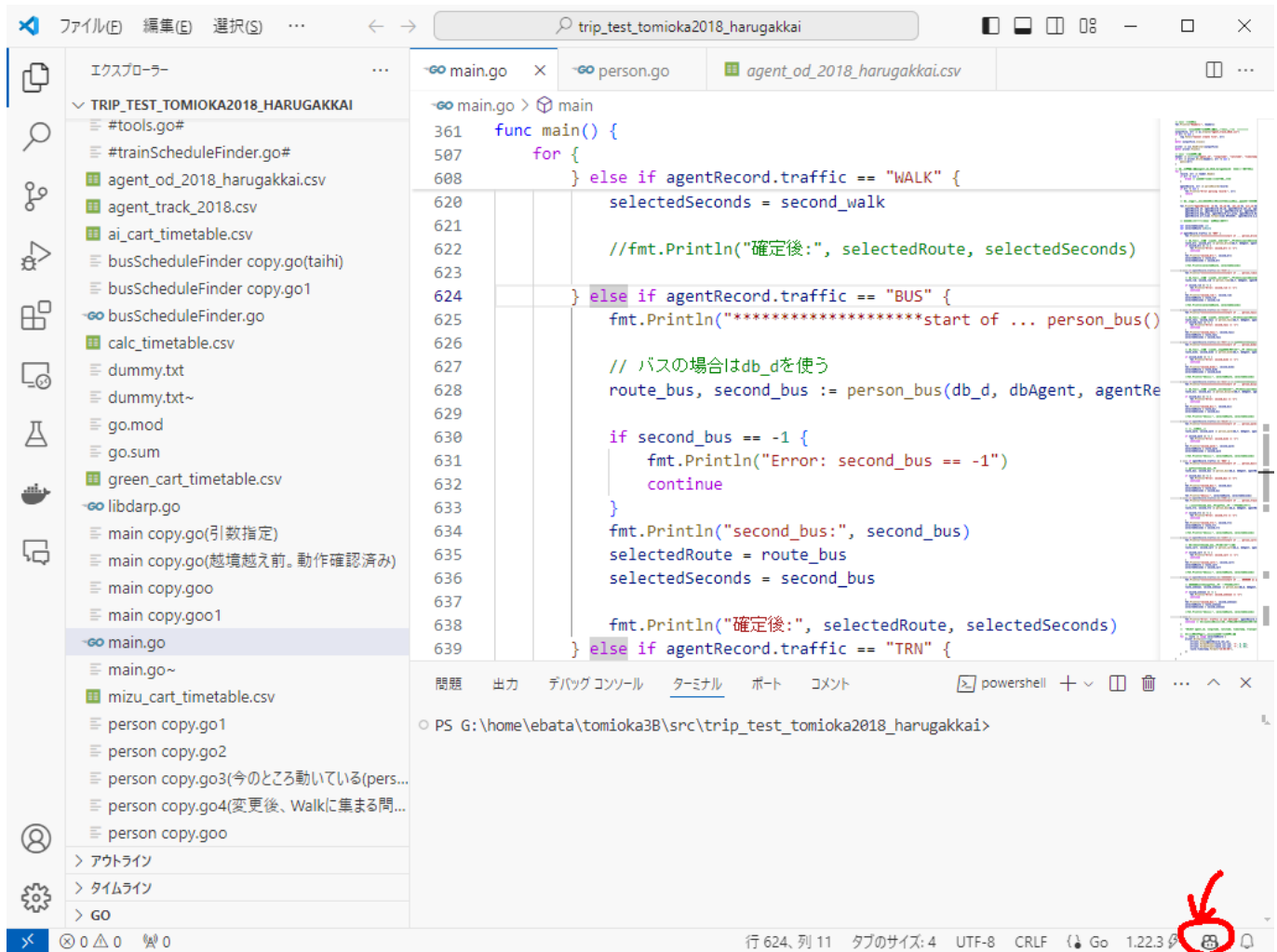
で、このインストールを実施すると、先程設定した"GitHub"のWeb画面から、2段階認証の作業を要求されます。

具体的には、Web画面に飛ばされてログインアカウントとパスワードの入力を要求されます。

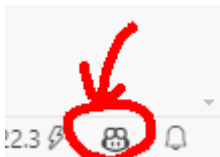
そして、その後、あなたのスマホに、認証用のコード(数字6ケタ?)が飛んでくるので、それを入力することで、やっと、GitHub Copilotが使用できるようになります。

(この作業で私は、何度か失敗しました。)

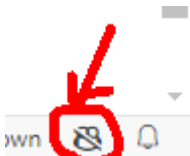
Copilotの起動に成功すると、VSCodeにこのアイコンが出てくるはずですが



拡大します。



失敗すると、こんな感じの表示になります。



失敗したらどうするか？ — 成功するまで、やり直しましょう。

(私、何度か失敗して、今は成功しているのですが、「なぜ上手く行ったのか」を説明できません。

『何度かやっているうちに、成功した』としか言えません。悪しからず。

7. Copilotとは何か(主観的所感)

7.1. Copilot(GitHub Copilot / Copilot Chat)の概要

7.1.1. その11

Go言語で、250人分のエージェントを作成して、次の処理にかかろうとして

```
for
```

と書いた段階で、

```
for i := 0; i < 250; i++ {  
    fmt.Println("person[" + i + "].Origin:", person[i].Origin.Name,  
person[i].Origin.Loc.Lng, person[i].Origin.Loc.Lat)  
    fmt.Println("person[" + i + "].Destination:", person[i].Destination.Name,  
person[i].Destination.Loc.Lng, person[i].Destination.Loc.Lat)  
    fmt.Println("person[" + i + "].Route:", person[i].Route)  
}
```

と、ここまで、自動的にコードを提案してきます。

7.1.2. その2

Go言語で、

```
// 時速4kmの場合、1秒間に移動する距離は、
```

とREM文を書いていたら、Copilotが、その後ろに、

```
4km/h ÷ 3600s/h = 0.001111111111km = 1.11111111m
```

と、文案を提案してきました。

あるいは、Java言語で"/"と記入するだけで、

```
// このテストでは、SumCalculatorクラスのmainメソッドを実行し、標準入力と標準出力をシミュレートして  
います。具体的には、標準入力に5と10を入力し、標準出力が"The sum of 5 and 10 is: 15"になることを検  
証しています。
```

という文章を、一瞬に作成します。

7.1.3. 具体的な使い方

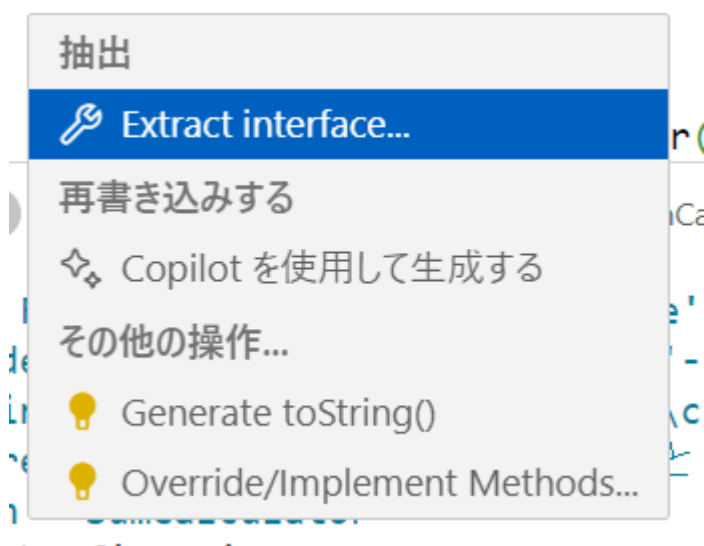
多くの場合、その内容は正しいです。もちろん、コメントの内容が異なる場合もありますが、概ね、そのコメントの文章を援用して使うことができます。

また、私たちがVSCodeエディタを見ていると、

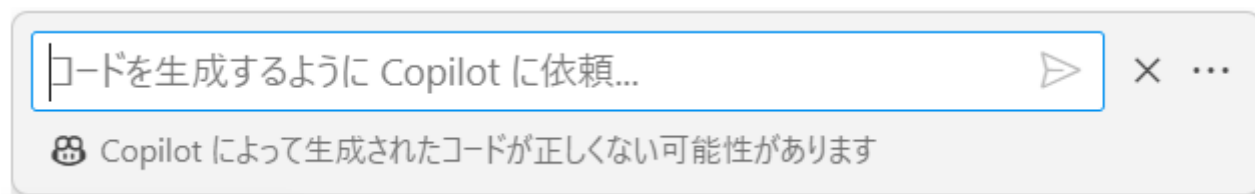
```
36 | assertEquals(5, result);  
37 | }
```

という黄色のマークが出てきます。

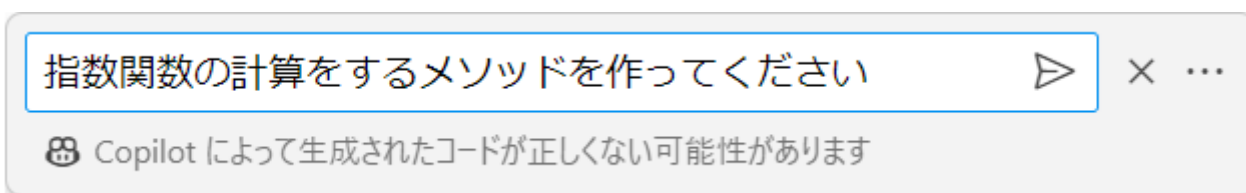
この黄色いマークをクリックすると、



というようなメニューがでできます。ここで「Copilotを作成して生成する」と選択すると、



というような、文字入力を促してきますので、ここに作りたいコードの概要を、適当なフレーズを入れて入力します。



などと入力すると、

指数関数の計算をするメソッドを作ってください

同意する

破棄



1 件の変更



```
27 // 指数関数の計算をするメソッド
28 public static int calculateExponential(int base, int exponent) {
29     int result = 1;
30     for (int i = 0; i < exponent; i++) {
31         result *= base;
32     }
33     return result;
34 }
```

こちらで一瞬でコードを生成してくれます。

私たちプログラムの作成または保守する者にとって、一番ラクできることは、コーディング本体はもちろんですが、変数の型などを、Copilotに丸投げすれば良い、ということです。

しかし、このようなメソッドを丸投げしたら、メインプログラムを作成する際に、結局自分が苦労することになりかねませんが、メソッドの利用方法のコードも作成してくれるので、助けてもらいましょう。

calculateExponentialをテストするプログラムを作ってください

同意する

破棄



2 件の変更



```
56 public class SumCalculatorTest {
57
58     @Test
59     public void testCalculateExponential() {
60         // Prepare input
61         int base = 2;
62         int exponent = 3;
63
64         // Execute the method
65         int result = SumCalculator.calculateExponential(base, exponent);
66
67         // Verify the output
68         assertEquals(8, result);
69     }
70
71     @Test
72     public void testSumCalculator() {
73         // Prepare input
74         String input = "5\n10\n";
75         InputStream inputStream = new ByteArrayInputStream(input.getBytes());
```

これだけでも、Copilotの威力は十分に感じて頂けると思います。

実際に本文を記載している私(江端)は、プログラムコードを1行も書くことなく、「日本語のフレーズ」だけを使って、プログラムを完成させることができました。

7.2. Copilotの弱点

このように、Copilotがプログラムをラクさせてくれる、という点においては、疑いはありませんが、Copilotは、私たちの意図を誤って解釈してコーディングすることがあります。

Copilotの最大の強みであり弱みは、Copilotは「その条件では、コードは作成できません」と絶対に言わない、ということです。Copilotは分からないことがあっても、力付くでコードを作り上げてしまいます。しかもたちの悪いことに、そのプログラムはコンパイルエラーなく、ちゃんと動くおとがあります。

ですので、私たちはCopilotのコードの内容を正しく理解するスキルがあることが、必要な条件となります。

8. GitHub Copilotの使い方

多分、説明の必要はありません。というか、使えば直ぐに分かることなので、説明する必要がありません。

VSCodeで、GitHub Copilotをアクティブになっていることを確認して、適当なサンプルプログラムを書き始めてみて下さい。

例えば、test.c というc言語のプログラムを

```
#include <std
```

ここまで書いてだけで、もうプログラムを勝手に作られて(多分ビックリすると思います)。勝手に色々なコメントやらが出てきます。これが、コード補完機能です。

8.1. コード補完機能

コード補完機能とは、コードを書き始めたり、作成したいコードをコメントで記述したりすることで、コードを自動的に生成してくれる機能です。

提案されたコードをそのまま利用する場合は、「Tab」を押すと、コードが反映されます。

利用しない場合は、「Esc」を押すと、提案されたコードは拒否できます。

また、コメントとして「XXXXXする関数を書いてください。」と記述すると、GitHub CopilotがXXXXXの機能のコードを自動的に提案してくれます。

ザックリと纏めると、こんな感じです。

コマンド操作	VSCode
提案を受け入れる	Tab
提案を拒否する	Esc

これだけで十分だと思いますが、以下のようなものもあるようです(私は、これ以外に使ったことがありませんが)。

コマンド操作	VSCode
次の提案を表示	Alt +]
前の提案を表示	Alt + [

コマンド操作

VSCode

次の単語を受け入れる

Control + →

複数の提案を含む新しいタブを開く

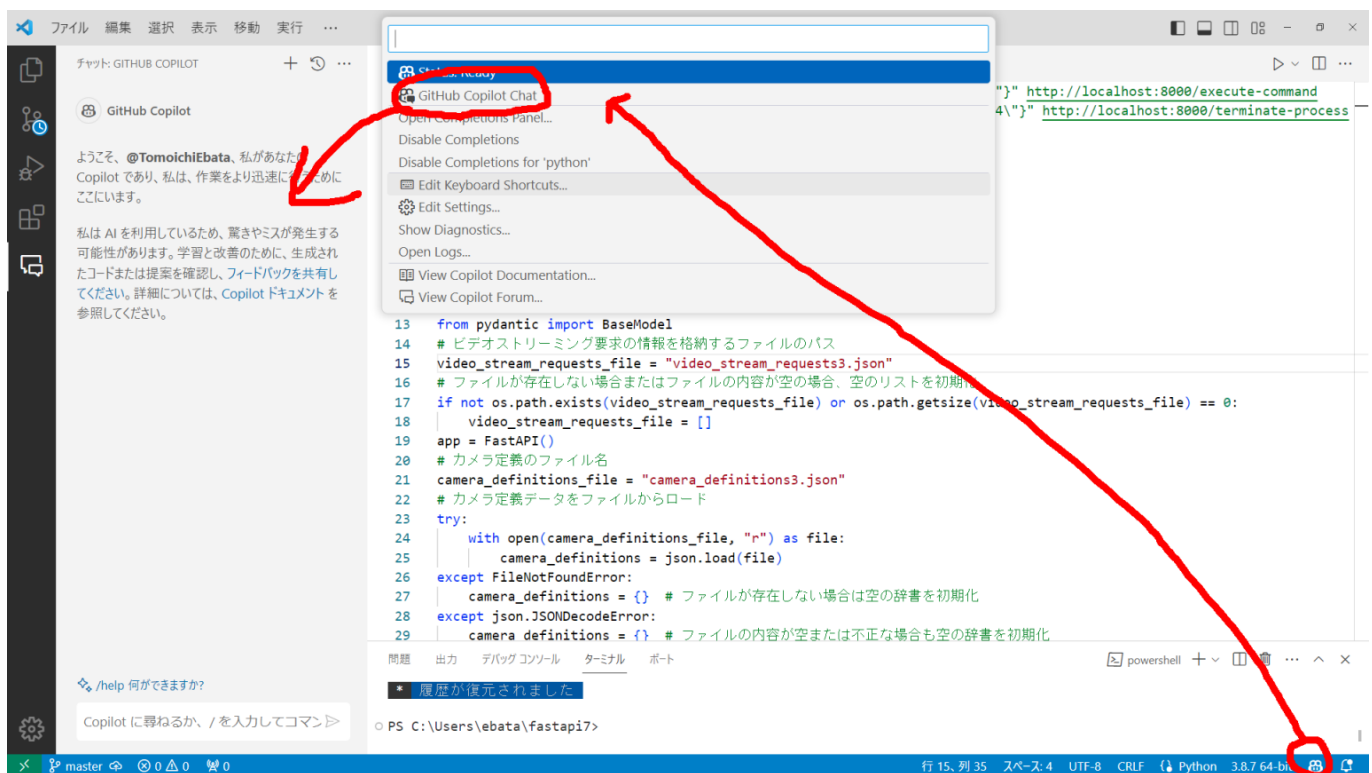
Ctrl + Enter

9. Copilot Chatの使い方

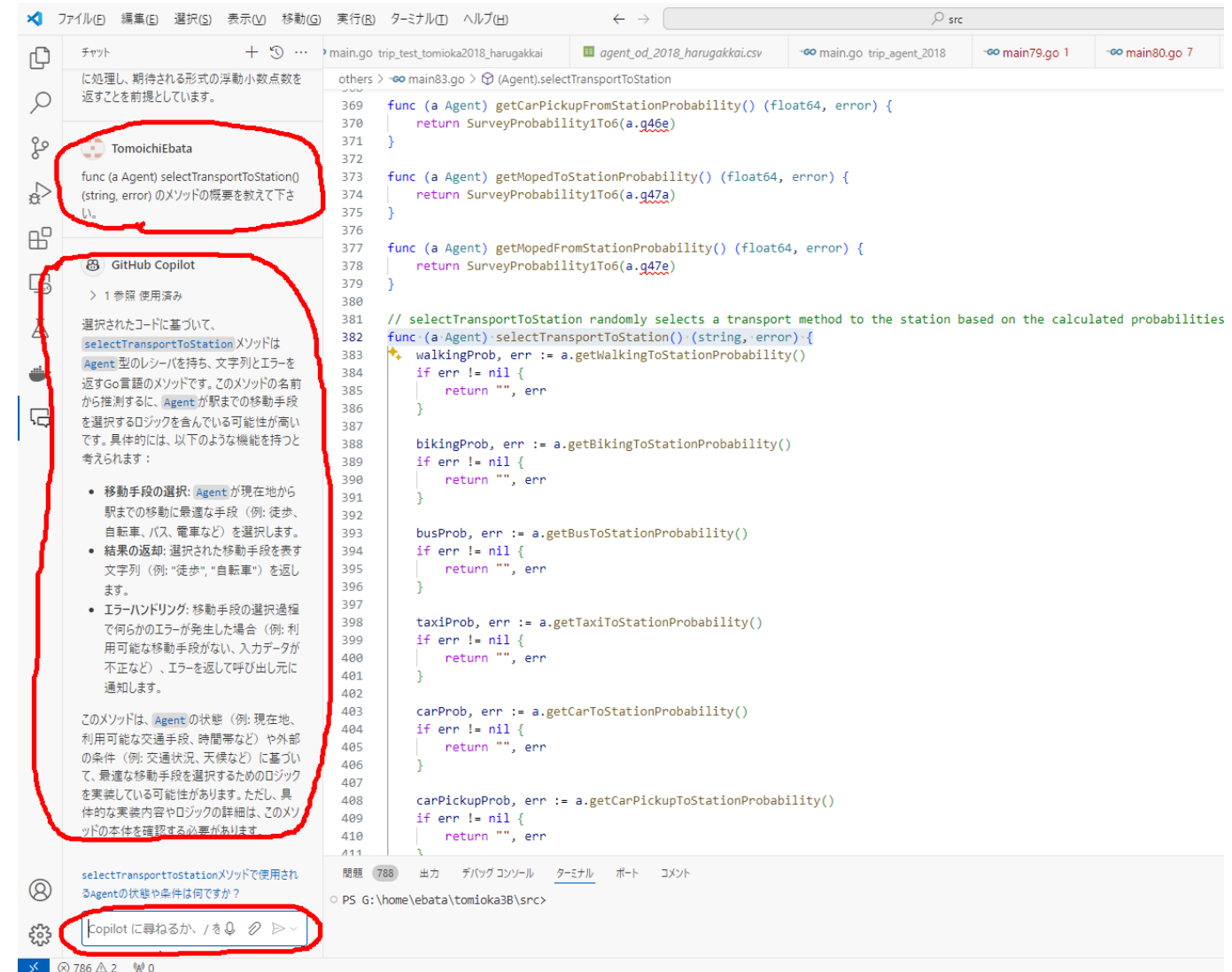
9.1. Chat機能

Copilot Chatは、GitHub Copilotと違って、1アクション必要となります。

アイコンをクリックして、メニュー画面から、"GitHub Copilot Chat"を選択する必要があります(これだけです)



で、あとは空欄に質問を文章で記載して下さい。ここでは、「func (a Agent) selectTransportToStation() (string, error) のメソッドの概要を教えてください」を入力してみました(まあ、私が作ったメソッドなんですけどね)。



前述した通り、ChatGPT的な使い方ができます。Copilot Chatは、ソースコードを教える必要がない分、手を抜けます。

これ以外にも、便利な使い方がありますが、それには言及しません。私が使っていないからです。

10. まとめ

本書では、GitHub Copilot / Copilot Chat について『**だまって、これだけやれ**』を記載しました。

ここで紹介したGitHub Copilot / Copilot Chat機能は、ほんの一部ですが、私にはこれだけでも十分に役に立っています。

もっと勉強すれば良いのですが、現状に満足してしまっています(作成しなければならないコードが山ほどある、というのもその理由の一つです)。

コーディングをしなければならない人は、とっととGitHub Copilot / Copilot Chat を始めることをお勧めします。

得られるメリットを考えれば、10ドル/月はゴミのようなものです(と、私(江端)は思っています)。

以上