

1. 導入編

このセクションのゴール

- 環境構築を完了する
- Web サイトの基本的な構造を理解する
- HTML/CSS が書けるようになる
- GitHub Pages で Web サイトを公開できるようになる

目次

1. 準備
2. 最低限必要な知識
3. HTML とは
4. CSS とは
5. GitHub Pages での公開

01-1. 準備

事前準備

必要なソフト

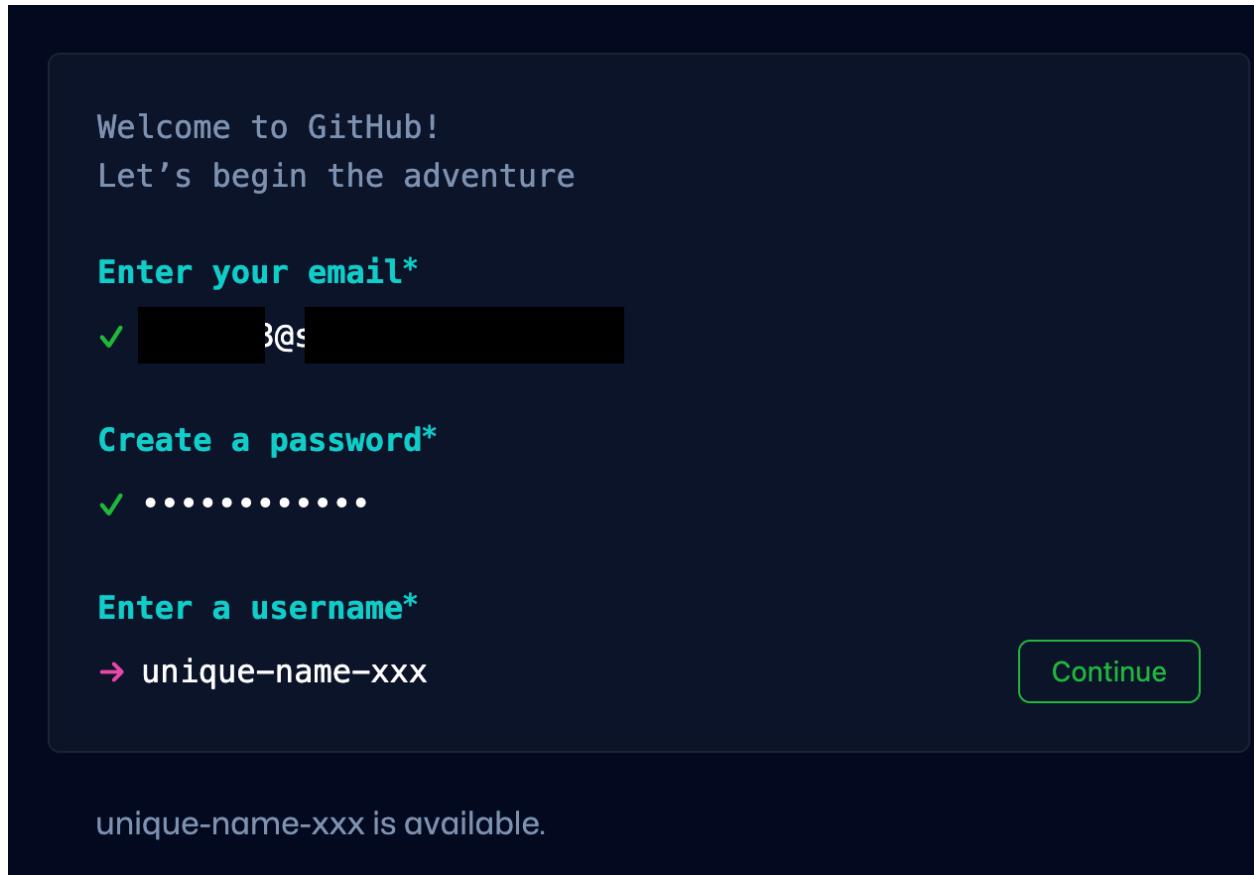
- [Git](#)
 - バージョン管理ツール
- [Visual Studio Code](#)
 - コードエディタ

必要なアカウント

- [GitHub](#)
 - リモートリポジトリのホスティングサービス

GitHub アカウントの作成

1. GitHub にアクセス
2. 右上の `Sign up` をクリック
3. メールアドレス、パスワード、ユーザ名を設定
4. bot テストのミニゲームをクリア
5. メールで送られた認証コードを入力
6. プランは Free を選択
 - メールアドレスは個人のものを推奨
 - ユーザ名は他人と被っている場合利用できません



Git のダウンロード

Windows の場合

1. [Git for Windows](#) にアクセス
2. [Click here to download](#) をクリック

Mac の場合

1. [XCode](#)にアクセス
2. 右上の Download をクリック
3. App Store が開くので、 XCode をインストール

The screenshot shows the official Git website at git-scm.com. The top navigation bar includes links for About, Documentation, Downloads, and Community. The main content area features a large "git" logo and the tagline "--distributed-is-the-new-centralized". A search bar is located in the top right corner. The "Downloads" section is highlighted with a red box around the "Click here to download" button, which links to the latest 64-bit version of Git for Windows. Below this, there are sections for Other Git for Windows downloads (Standalone Installer, 32-bit and 64-bit Setup), Pro Git book information, and instructions for using winget tool. The "Now What?" section provides a call to action for new users.

Download for Windows

[Click here to download](#) the latest (2.46.2) 64-bit version of **Git for Windows**. This is the most recent [maintained build](#). It was released **2 days ago**, on 2024-09-24.

Other Git for Windows downloads

Standalone Installer
[32-bit Git for Windows Setup](#).
[64-bit Git for Windows Setup](#).

Portable ("thumbdrive edition")
[32-bit Git for Windows Portable](#).
[64-bit Git for Windows Portable](#).

Using winget tool

Install [winget tool](#) if you don't already have it, then type this command in command prompt or Powershell.

```
winget install --id Git.Git -e --source winget
```

The current source code release is version **2.46.2**. If you want the newer version, you can build it from [the source code](#).

Now What?

Now that you have downloaded Git, it's time to start using it.

[Read the Book](#)
Dive into the Pro Git book and learn at your own pace.

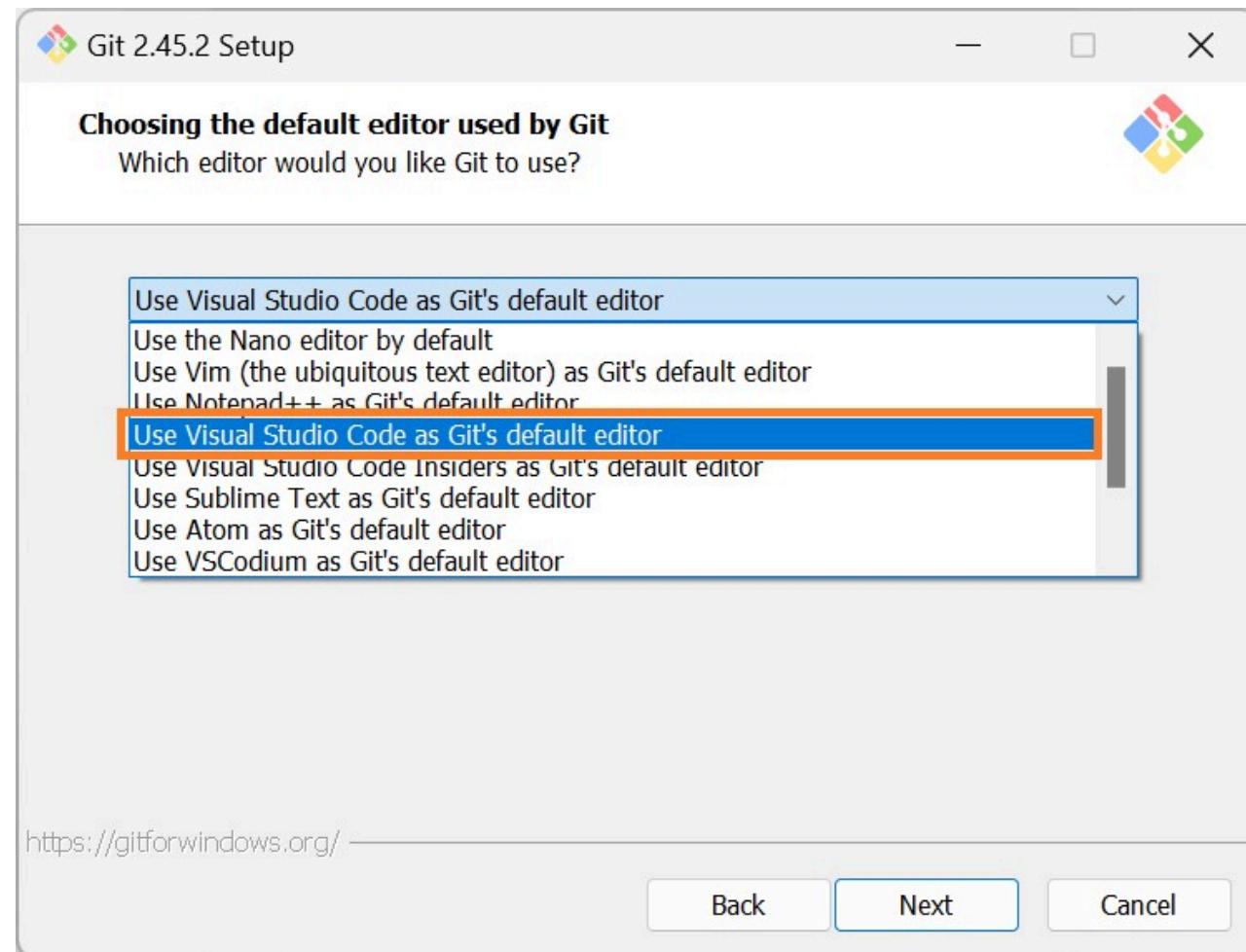
[Download a GUI](#)
Several free and commercial GUI tools are available for the Windows platform.

[Get Involved](#)
A knowledgeable Git community is available to answer your questions.

Git のインストール

Windows の場合

1. ダウンロードフォルダにある exe ファイルをダブルクリック
2. ウィザードが立ち上るので、基本的に右下の「Next」をクリックし続ける
3. 「Choosing the default editor used by Git」画面で、Use Visual Studio Code as Git's default editor を指定する



Git の設定

- スタートメニュー(左下 Windows アイコン)から Git Bash を起動
 - Mac の場合は「アプリケーション」の「ユーティリティ」フォルダにあるターミナルを起動
- 下記のコマンドを入力して、ユーザ名とメールアドレスを設定

```
git config --global user.name "ここにGitHubのユーザ名"
```

```
git config --global user.email "ここにGitHubのメールアドレス"
```

- 特にエラーが出ていなければ設定完了

Visual Studio Code のダウンロード

1. [Visual Studio Code](#) にアクセス
2. OS に合わせてインストーラをダウンロード
3. インストール

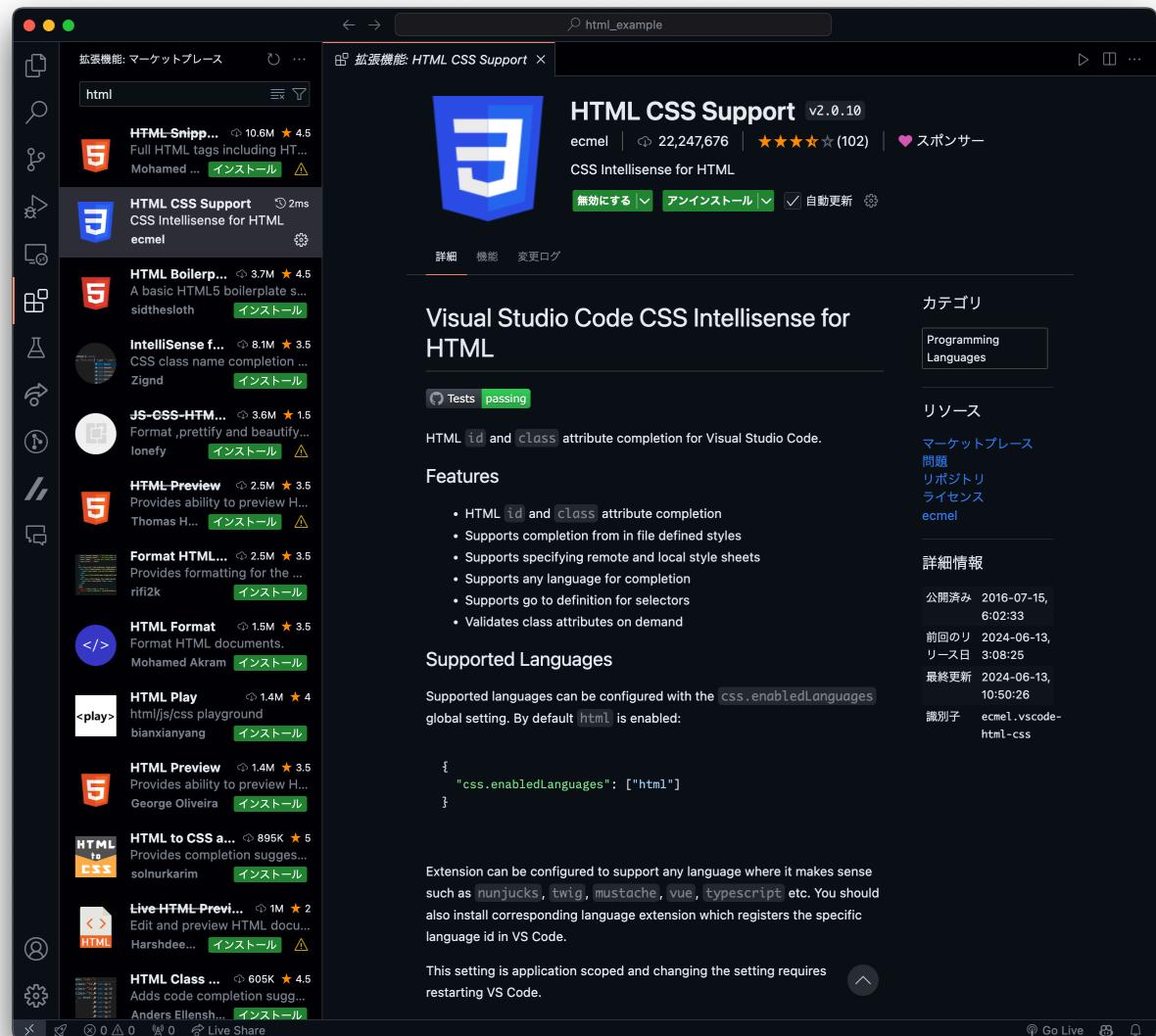
補足

- Microsoft が主導して開発しているオープンソースのコードエディタ
- 一般的に **VSCODE** と呼ばれるので、本資料でも以降は VSCODE と表記します

Visual Studio Code の設定

拡張機能のインストール

- VSCode を起動し、左側のアイコンから拡張機能を検索
- 「HTML CSS Support」と「Live Server」をインストール



1-2. 最低限必要な知識

心構え

書き方を覚えようとしないでください。

量が膨大というのもありますが、特に Web 技術は日々進化しているため、数年前の常識が通用しないことがあります。個人的には写経（サンプルコードを書き写すこと）にも意味がないと思っているので、どんどんコピペしてください。

そもそもこれはテストではないので、「勉強」という意識を持たないほうがラクです。

心構え

書き方を覚えるのではなく、調べ方を覚えてください。

詰まったときは信頼できるドキュメントを見るのが一番ですが、英語であることが多いので、ハードルを感じる場合は以下に挙げるサイトがおすすめです。

- [MDN Web Docs](#)
- [HTML&CSS 入門：イチから Web デザインを習得する講座](#)

ChatGPT や GitHub Copilot などの生成 AI を活用するのも良いでしょう。

各言語の役割

- HTML: 文書の構造を記述
 - 「ここが見出し」「ここが段落」など
- CSS: デザインを記述
 - 「見出しが赤色」「段落のフォントサイズは 16px」など
- JavaScript: 動的な挙動を記述
 - 「ボタンをクリックしたらローディングアニメーションを表示」など

ウェブページを開いたときに表示されているのは殆どの場合 HTML です。

基本的に CSS や JavaScript は HTML から呼び出されることが多いです。

01-3. HTML とは

HTML とは

- HyperText Markup Language
 - 「マークアップ言語」であって「プログラミング言語」ではない
- 文書の構造を記述するための言語
- タグで囲まれた要素を使って構造を表現
- `index.html` というファイル名に特別な意味を持つ
 - `https://example.com/index.html`へのアクセスは `https://example.com/` と同じ

HTML5は 2014 年 10 月に W3C によって勧告された HTML のバージョン。2021 年 1 月に廃止され、以降は **HTML Living Standard** が有効な規格となっている

HTML の基本構造

```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="UTF-8" />
    <title>ページのタイトル</title>
  </head>
  <body>
    <!-- これはコメントで、実際には表示されません -->
    <h1>見出し</h1>
    <p>段落</p>
  </body>
</html>
```

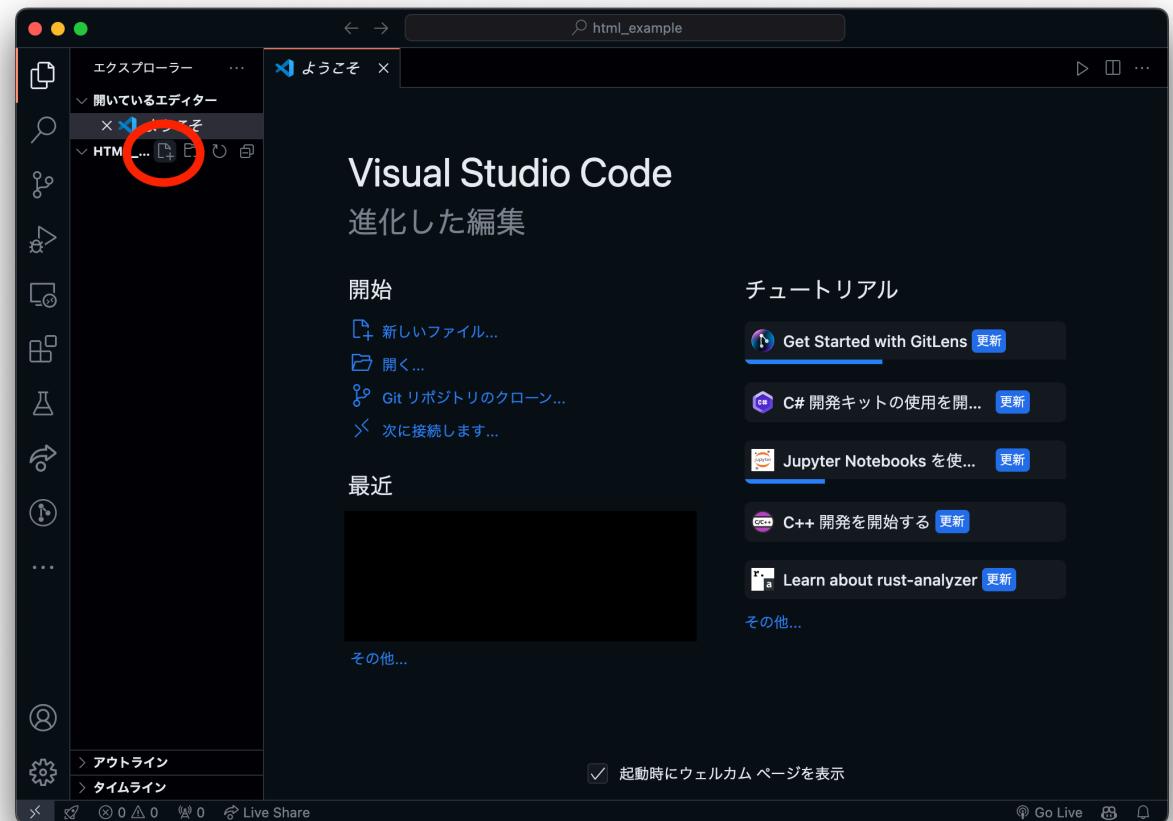
これだけで表示することができる！

HTML の基本的なルール

- 開始タグを書いたら必ず終了タグを書く
 - < で始まり > で終わっている部分がタグで、とくに </> で始まっているものが終了タグ
 - タグや
 タグなど、一部例外あり
- Python などとは異なり、インデントの大きさは問わない
- コメントは <!--> と <--> で囲む
- 1 行目の <!DOCTYPE html> は 必ず書く
- <html> タグの中に <head> と <body> がある（この 3 つは必須）
- <head> タグの中にはページの情報を書く
- <body> タグの中にはページの内容を書く
 - ページを開いた時に表示されるのはこの部分

HTML ファイルを作成してみよう

1. 作業用のフォルダ (portfolio など) を作成
2. VSCode を起動し、左上の「ファイル」→「フォルダを開く」から作成したフォルダを開く
3. 新しいファイルを作成 (右図の赤丸)
4. ファイル名を `index.html` として保存
 - フォルダ作成ボタンと間違えないように注意
 - もしくは左上の「ファイル」→「新しいファイル」



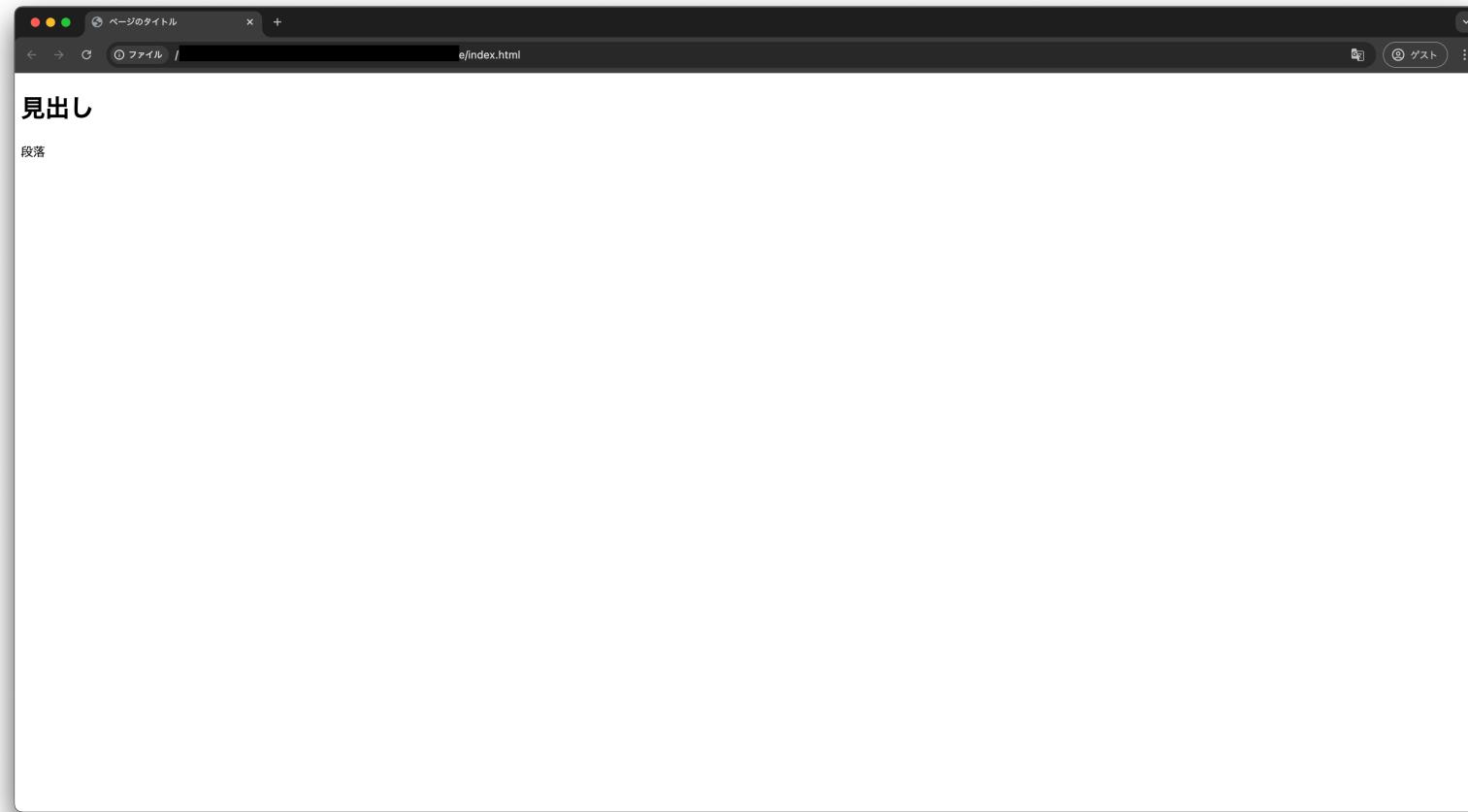
HTML をブラウザで表示してみよう

- 作成した HTML ファイルに先ほどのコードを書き込む

```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="UTF-8" />
    <title>ページのタイトル</title>
  </head>
  <body>
    <h1>見出し</h1>
    <p>段落</p>
  </body>
</html>
```

- ファイルを保存し、VSCode 右下の「Go Live」をクリック
 - エクスプローラーから HTML ファイルをダブルクリックすることでも表示できますが、この方法を用いることでリアルタイムで（再読みすことなく）変更を確認できます

HTML をブラウザで表示してみよう



おもに使うタグ

タグ	説明
<h1> - <h6>	見出しを表現するタグ
<p>	段落を表現するタグ
<a>	ハイパーアリンクを作成するタグ
	画像を表示するタグ
 , , 	リストを表現するタグ
<table> , <tr> , <td>	表を表現するタグ
<div> , 	ブロック要素とインライン要素を表現するタグ

別のページにリンクさせる

- VSCode で新しく `about.html` を作成し、以下のコードを書き込む

```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="UTF-8" />
    <title>aboutページ</title>
  </head>
  <body>
    <h1>aboutページ</h1>
    <p>これはaboutです。</p>
    <a href=". ./index.html">トップページに戻る</a>
  </body>
</html>
```

- `index.html` の `body` タグ内に以下のコードを追加

```
<a href=". ./about.html">aboutページへ</a>
```

画像を表示させる

- 画像をダウンロードして、HTML ファイルと同じフォルダに保存
 - 分かりやすくするために `images` フォルダなどを作成して配置することが多い
- `index.html` の `body` タグ内に以下のコードを追加
 - `your-image-name.png` はダウンロードした画像のファイル名

```

```

- 必ず `alt` 属性をつけること
 - 画像が表示できない場合に代替テキストとして表示される

箇条書きを作る

- index.html の body タグ内に以下のコードを追加

```
<ul>
  <li>リスト1</li>
  <li>リスト2</li>
  <li>リスト3</li>
</ul>
```

- ul は unordered list の略で、順序のないリストを表現
- li は list item の略で、リストの各項目を表現

```
<ol>
  <li>リスト1</li>
  <li>リスト2</li>
  <li>リスト3</li>
</ol>
```

- ol は ordered list の略で、順序のあるリストを表現

表を作る

- index.html の body タグ内に以下のコードを追加

```
<table>
  <thead>
    <tr>
      <th>見出し1</th>
      <th>見出し2</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>1行目1列目</td>
      <td>1行目2列目</td>
    </tr>
    <tr>
      <td>2行目1列目</td>
      <td>2行目2列目</td>
    </tr>
  </tbody>
</table>
```

- table は表を表現
- thead は表の見出し部分
- tbody は表の本体部分
- tr は table row の略で、表の行を表現
- th は table header の略で、表の見出しを表現
- td は table data の略で、表のデータを表現

補足: タグのネスト

- これまでの例にもあったように、タグは入れ子にすることができる
- 例えば、リストの中にリストを入れることもできる

```
<ul>
  <li>リスト1</li>
  <li>
    リスト2
    <ul>
      <li>リスト2-1</li>
      <li>リスト2-2</li>
    </ul>
  </li>
</ul>
```

- ただし、許容されていない組み合わせもあるので注意
 - 例えば、`a` タグの中に `a` タグを入れたり、`p` タグの中に `div` タグを入れたりするのは避ける

01-4. CSS とは

CSS とは

- Cascading Style Sheets
 - スタイルシート言語
- 文書のデザインを記述するための言語
- セレクタとプロパティを使ってデザインを記述

CSS の基本構造

```
h1 {  
  color: red;  
  font-size: 32px;  
}  
p {  
  background-color: lightgray;  
  text-align: center;  
}
```

- セレクタ { プロパティ: 値; } の形式で記述
- セレクタはタグ名や id、class など

h1 タグの文字色を変えてみよう

- `style.css` というファイルを作成
- `index.html` を以下のようにする
- `style.css` に以下のコードを書き込む

```
<!DOCTYPE html>
<html lang="ja">
  <head>
    <meta charset="UTF-8" />
    <title>ページのタイトル</title>
    <link rel="stylesheet" href=".style.css" />
  </head>
  <body>
    <h1>見出し</h1>
    <p>段落</p>
  </body>
</html>
```

```
h1 {
  color: red;
}
```

- `head` タグ内に `<link rel="stylesheet" href=".ファイル名.css" />` を追加することで CSS ファイルを読み込むことができる

id と class

- id はページ内で一意の要素を指定するための属性
 - 1 ページで 1 回しか使うことができない
- class は複数の要素に同じスタイルを適用するための属性
 - 複数の要素に同じスタイルを適用したい場合に使う

```
<h1 id="title">文字色が赤になる</h1>
<p class="description">ここの背景がグレーになる</p>
```

```
#title {
  color: red;
}
.description {
  background-color: lightgray;
}
```

- CSS では名前の前に、 id は # 、 class は . をつける
 - バグを防ぐため id は使わず class を使うことが多い

詳細度 ①

```
<h1 id="title_id" class="title_class">見出し</h1>
```

```
#title_id {  
    color: blue;  
}  
.title_class {  
    color: green;  
}  
h1 {  
    color: red;  
}
```

このとき、見出しへ何色になる？

詳細度 ②

- 複数のセレクタが同じ要素を指定している場合、詳細度が高いものが優先される
- 基本的には最後に書かれたスタイルが優先されるが、詳細度によっては逆転する場合がある
- 基本的に id セレクタ > class セレクタ > タイプセレクタ の順に詳細度が高い

実際のところ詳細度の計算はより複雑で、CSSが難しいとされる理由の一つだったりします。

おもに使うプロパティ

プロパティ名	説明
color	文字色を変更
background-color	背景色を変更
font-size	文字サイズを変更
padding	
margin	
border	枠線に関する設定

padding と margin

- padding は要素の内側の隙間、margin は要素の外側の隙間

```
<div>一つ前の要素</div>
<div class="target">要素</div>
<div>一つ後の要素</div>
```

- 何も指定しないとき

```
// 見やすさのため色のみ設定
.target {
  background-color: red;
  color: white;
}
```

一つ前の要素

要素

一つ後の要素

- padding だけ指定したとき

```
.target {
  background-color: red;
  color: white;
  padding: 16px;
}
```

一つ前の要素

要素

一つ後の要素

- margin だけ指定したとき

```
.target {
  background-color: red;
  color: white;
  margin: 16px;
}
```

一つ前の要素

要素

一つ後の要素

色に関するプロパティ

- `color`、`background-color` など
- 指定方法はさまざま
 - キーワード: `red`, `blue`, `green` など
 - RGB: `rgb(255, 0, 0)`
 - HEX(いわゆるカラーコード): `#ff0000`
 - HSL: `hsl(0, 100%, 50%)`

色に困ったときはMaterial Colorsなどのカラーパレットサイトを使うと便利です。
<https://materialui.co/colors>

数値を扱うプロパティ

- `px` ピクセル
- `%` パーセント
- `em` 親要素のフォントサイズに対する相対的な大きさ
 - `1em` は親要素のフォントサイズと同じ
- `rem` ルート要素のフォントサイズに対する相対的な大きさ
 - `1rem` は `<html>` タグのフォントサイズと同じ

最初は `px` を使っても良いが、レスポンシブデザインを考えると `%` や `rem` を使うほうが良い

レスポンシブデザイン：デバイスの画面サイズに合わせて（パソコンやスマホに応じて）デザインを変えること

枠線をつける

- border: 線の太さ 線の種類 線の色 で指定できる

```
.target {  
  border: 1px solid black;  
}
```

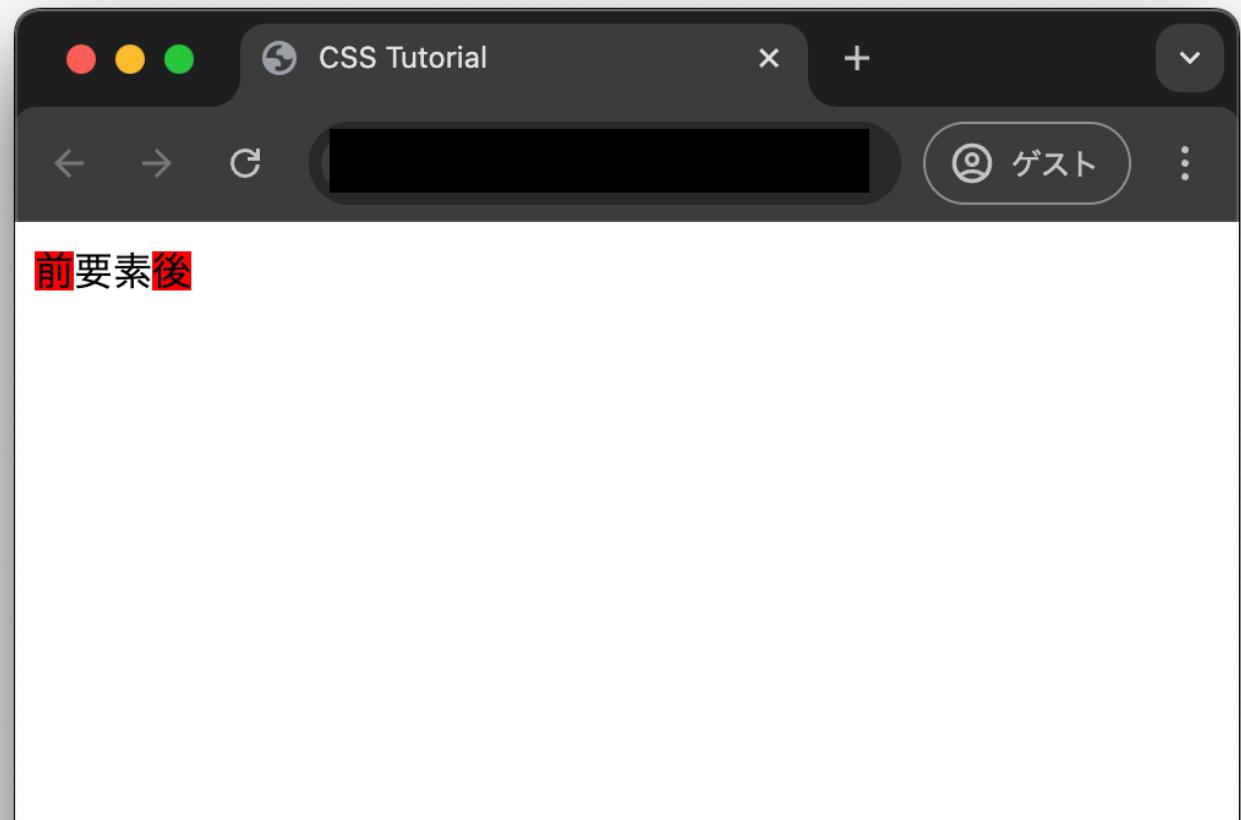
- 線の種類
 - solid : 実線
 - dotted : 点線
 - dashed : 破線
 - double : 二重線
 - none : 線なし

疑似要素

- `::before`、`::after`などの疑似要素を使うことで、要素の前後に要素を追加できる
- メリットとしては HTML を変えることなく装飾を加えることができる
- 実際には `content` の値を空にして、`<div>` タグのように装飾することが多い

```
<div class="target">要素</div>
```

```
.target::before {  
  content: "前";  
  background-color: red;  
}  
.target::after {  
  content: "後";  
  background-color: red;  
}
```



実例: 見出しのデザイン

```
h1 {  
    font-size: 3rem;  
    text-align: center;  
    padding: 1rem;  
    margin: 1rem;  
    color: white;  
    background: linear-gradient(to bottom right, blue, pink);  
    border-radius: 1.5rem;  
}
```



- `text-align: center` で中央揃え
- `linear-gradient` でグラデーションを作成
 - 角度、起点の色、終点の色を指定
 - 詳細は[linear-gradient\(\) - CSS: カスケーディングスタイルシート | MDN](#)を参照
- `border-radius` で角を丸く

01-5. GitHub Pages での公開

GitHub Pages とは

- GitHub が提供する静的 Web ホスティングサービス
- 以前はレンタルサーバーを契約し、FTP クライアントなどを使って公開したいファイルをアップロードする必要があったが、近年は Git でソースコードを管理し、GitHub に push するだけで Web サイトを開けるサービスが増えている
- 同様の他のサービスとしては Netlify や Vercel, Firebase Hosting, Cloudflare Pages などがある（いずれも無料で公開可能）

Git と GitHub の違い

- Gitはプログラムのバージョンを管理するためのソフトウェア
- GitHubは Git で管理されたプログラムをクラウド上で管理するためのWeb サービス
 - 同様の Web サービスとして GitLab, Bitbucket などもある
- GitHub は Git の機能に加え、Issue(プロジェクトの問題やタスク) を管理する機能や共同開発を支援する機能などを提供している
- GitHub Pages は GitHub が提供するサービスの一つ

GitHub でリポジトリを作成する

- GitHub にログインし、右上の「+」をクリックして「New repository」を選択
- Repository name にはお好きなものを
 - [https://username.github.io/\[ここがリポジトリ名になる\]/](https://username.github.io/[ここがリポジトリ名になる]/)
- Public/Private は自由
 - 認証情報を載せないよう注意！
- Create repository をクリック
- 作成後に遷移したページの URL をコピー

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Repository template

No template ▾

Start your repository with a template repository's contents.

Owner *



newt239 ▾

Repository name *

portfolio

portfolio is available.

Great repository names are short and memorable. Need inspiration? How about [scaling-octo-guide](#) ?

Description (optional)

Public

Anyone on the internet can see this repository. You choose who can commit.

Private

You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Git でコミット＆プッシュする

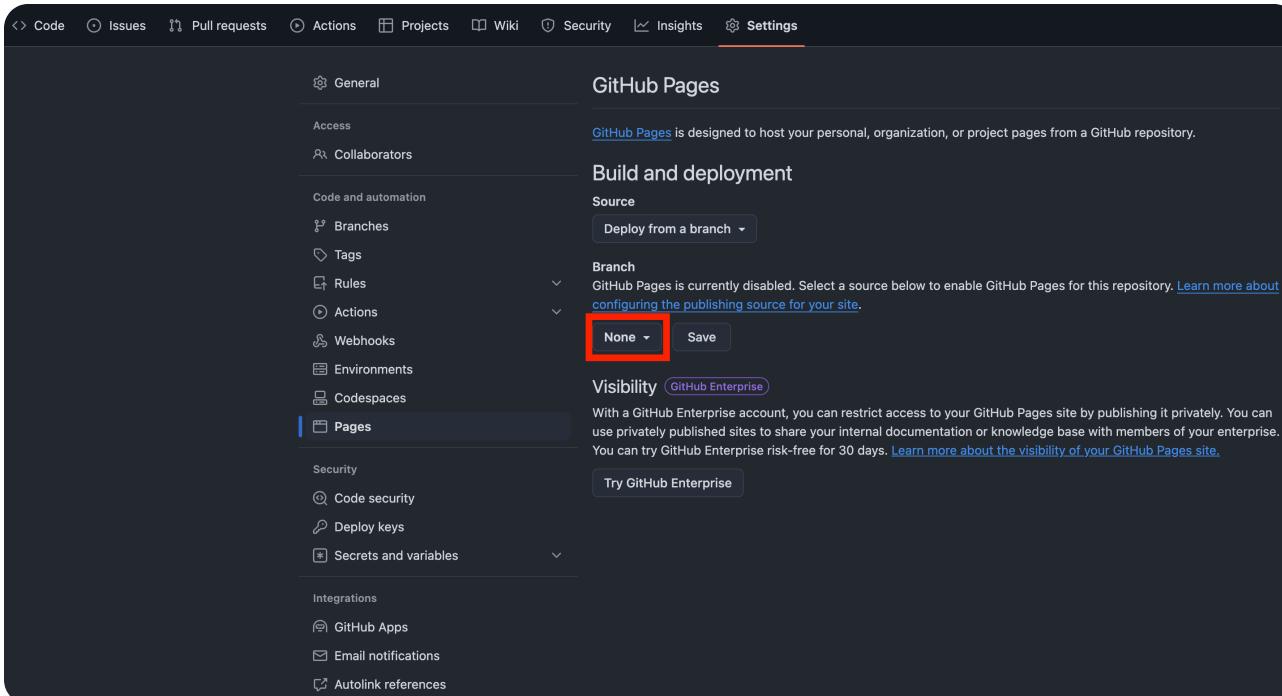
- VSCode 左上のメニューから「ターミナル」→「新しいターミナル」を選択
- 以下のコマンドを入力

```
git init  
git add .  
git commit -m "first commit"  
git remote add origin さっきコピーしたURL  
git push origin main
```

- コマンドは 1 行ずつ実行 (Enter)
- 最後のコマンドに失敗する場合、 `main` を `master` に変更
- いまのところコマンドの意味を理解する必要はありません！

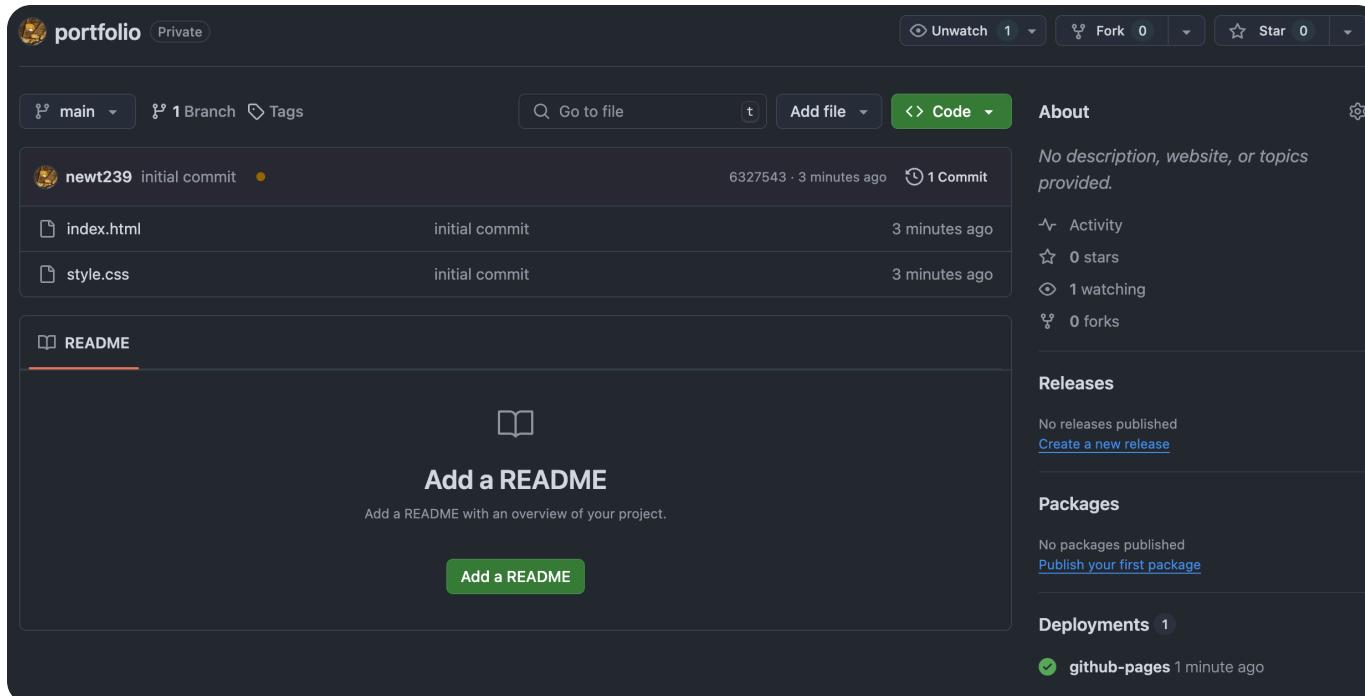
GitHub Pages で公開する

- ・コマンド実行後、GitHub で開いているページをリロード
- ・「Settings」 → 「Pages」 を選択
- ・Branch を main (または master) にして Save



公開できたことを確認

- Code タブに戻り、黄色い丸が緑のチェックマークになるまで待つ
 - 30 秒程度でリロードしてみてください
- 右下の Deployments の github-pages をクリックし、URL をクリック！



第1回の内容は以上です。お疲れ様でした！