

GIT&WSL入門講座 座2021

54代 HarrisonKawagoe(@hrsnnkwge_pro)



講座内容

- イン트로ダクション
- GitHubアカウントの作り方
- Gitの導入
- GitとGitHubの使い方
- WSLの導入
- WSLの使い方(Cプログラムのコンパイル)

ソースコード管理の重要性

- チーム開発では機能ごとに仕事を振り分けることが多く、チームメンバーが書いたソースコードを採用したり、添削したりするなど、プロジェクトを効率的に管理することができる
- 履歴が残るので、何が問題があったらロールバックできる

GITとGITHUBの違い



- Git:ローカルでソースコードを管理する為のツール



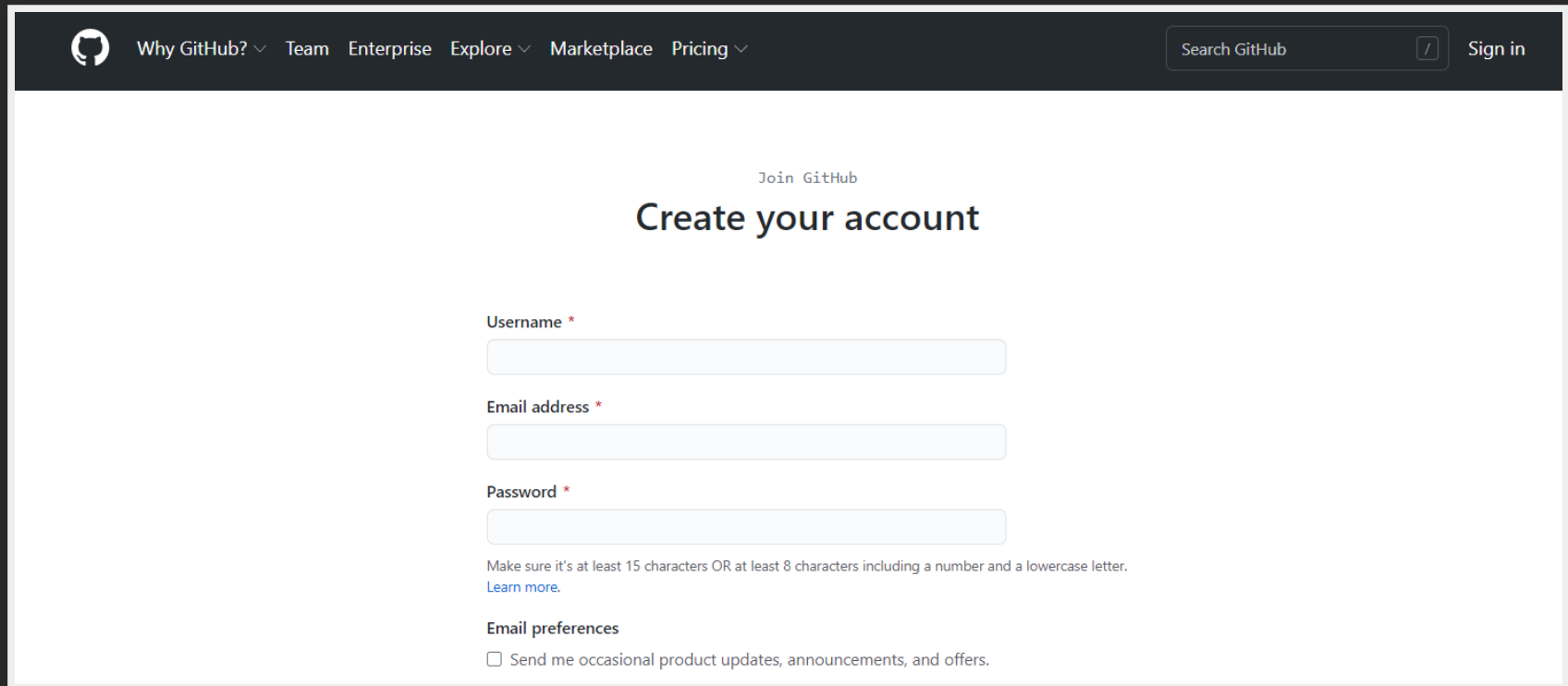
- GitHub:gitと組み合わせて、リモートでソースコード管理する為のプラットフォーム

- リポジトリのソースコード管理だけではなく、サイトを公開したり、Webアプリを自動でデプロイさせたりすることも可能



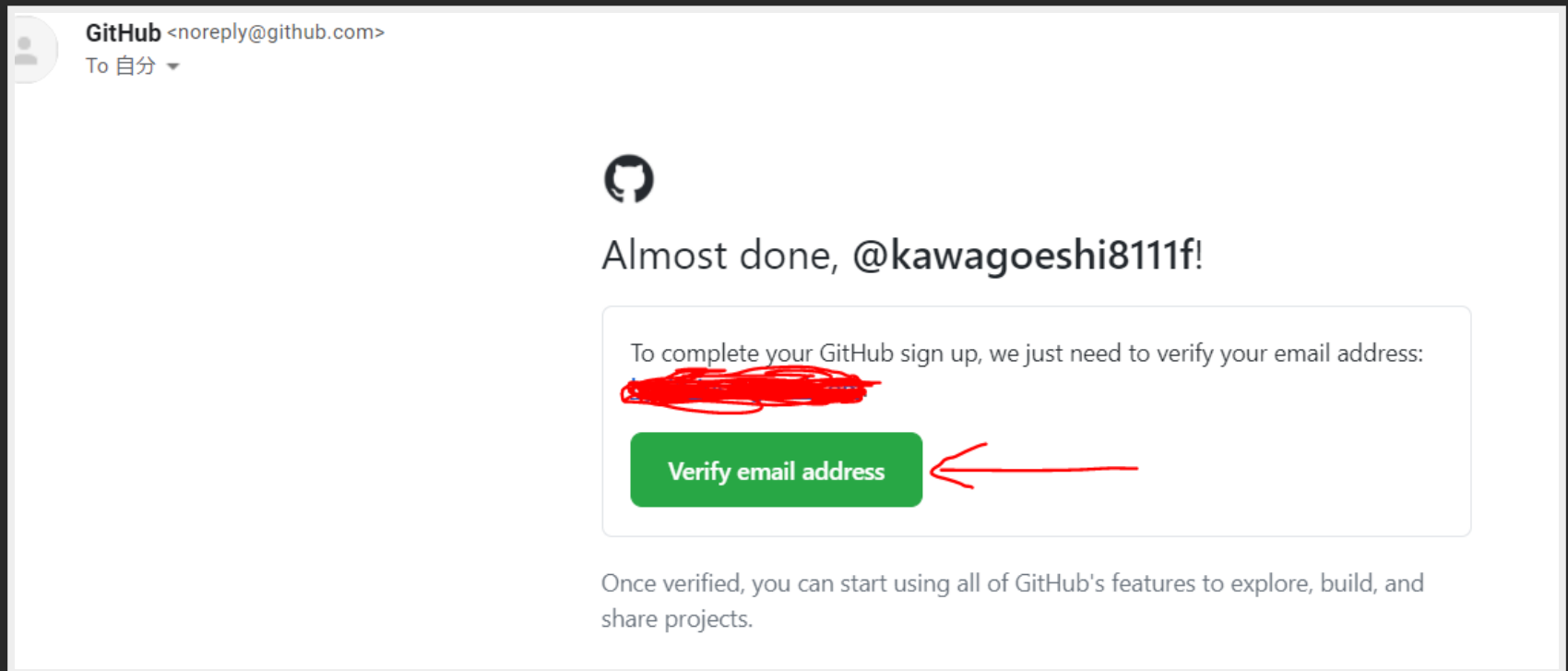
GITHUBアカウントの作り方

- <https://github.com/join>



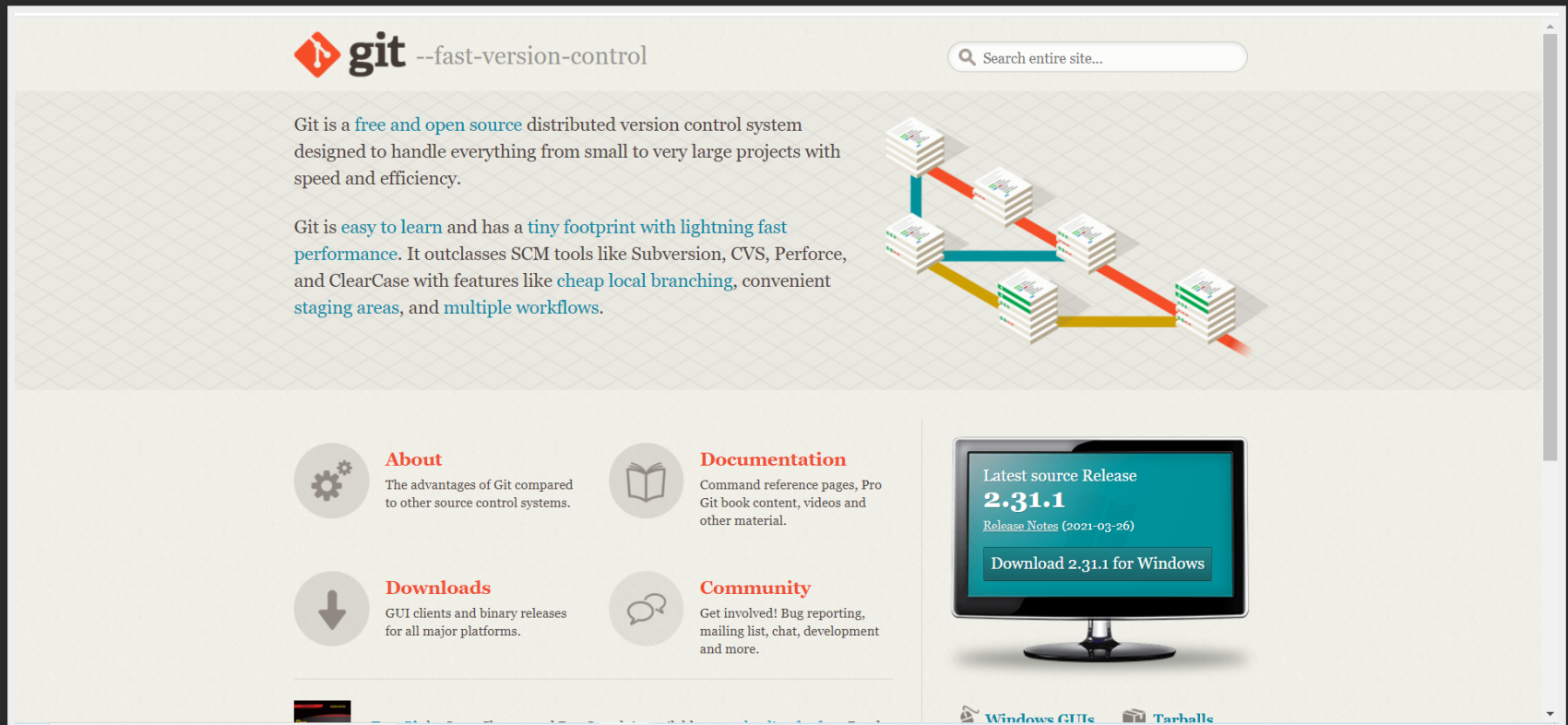
The screenshot shows the GitHub 'Create your account' page. At the top, there is a navigation bar with the GitHub logo, links for 'Why GitHub?', 'Team', 'Enterprise', 'Explore', 'Marketplace', and 'Pricing', a search bar labeled 'Search GitHub', and a 'Sign in' link. The main content area is titled 'Join GitHub' and 'Create your account'. It features three input fields: 'Username *', 'Email address *', and 'Password *'. Below the password field, there is a note: 'Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)' and an 'Email preferences' section with a checkbox labeled 'Send me occasional product updates, announcements, and offers.'

- 認証用のメールが届いたら、メール内の「Verify」ボタンをクリックして、アカウントを有効化します



GITの導入

- <https://git-scm.com/>



The screenshot shows the Git website homepage. At the top left is the Git logo (a red octopus) followed by the text "git --fast-version-control". To the right is a search bar with the placeholder text "Search entire site...". Below the logo and search bar is a large section with a grid background. On the left side of this section, there are two paragraphs of text. The first paragraph describes Git as a free and open source distributed version control system. The second paragraph describes Git as easy to learn and having a tiny footprint with lightning fast performance. On the right side of this section is a 3D diagram showing several stacks of books connected by colored lines (red, blue, yellow) representing a branching model. Below this section are four icons with corresponding text: "About" (gears icon), "Documentation" (book icon), "Downloads" (downward arrow icon), and "Community" (speech bubbles icon). To the right of these icons is a computer monitor displaying the latest source release "2.31.1" and a button to "Download 2.31.1 for Windows". At the bottom right of the monitor area are links for "Windows GUIs" and "Tarballs".

git --fast-version-control

Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.

About
The advantages of Git compared to other source control systems.

Documentation
Command reference pages, Pro Git book content, videos and other material.

Downloads
GUI clients and binary releases for all major platforms.

Community
Get involved! Bug reporting, mailing list, chat, development and more.

Latest source Release
2.31.1
Release Notes (2021-03-26)
Download 2.31.1 for Windows

Windows GUIs Tarballs

- Macの場合、デフォルトでGitが入っていることが多い
- ターミナルで、「git」を叩くと、Gitが入っているかどうかを確認できる

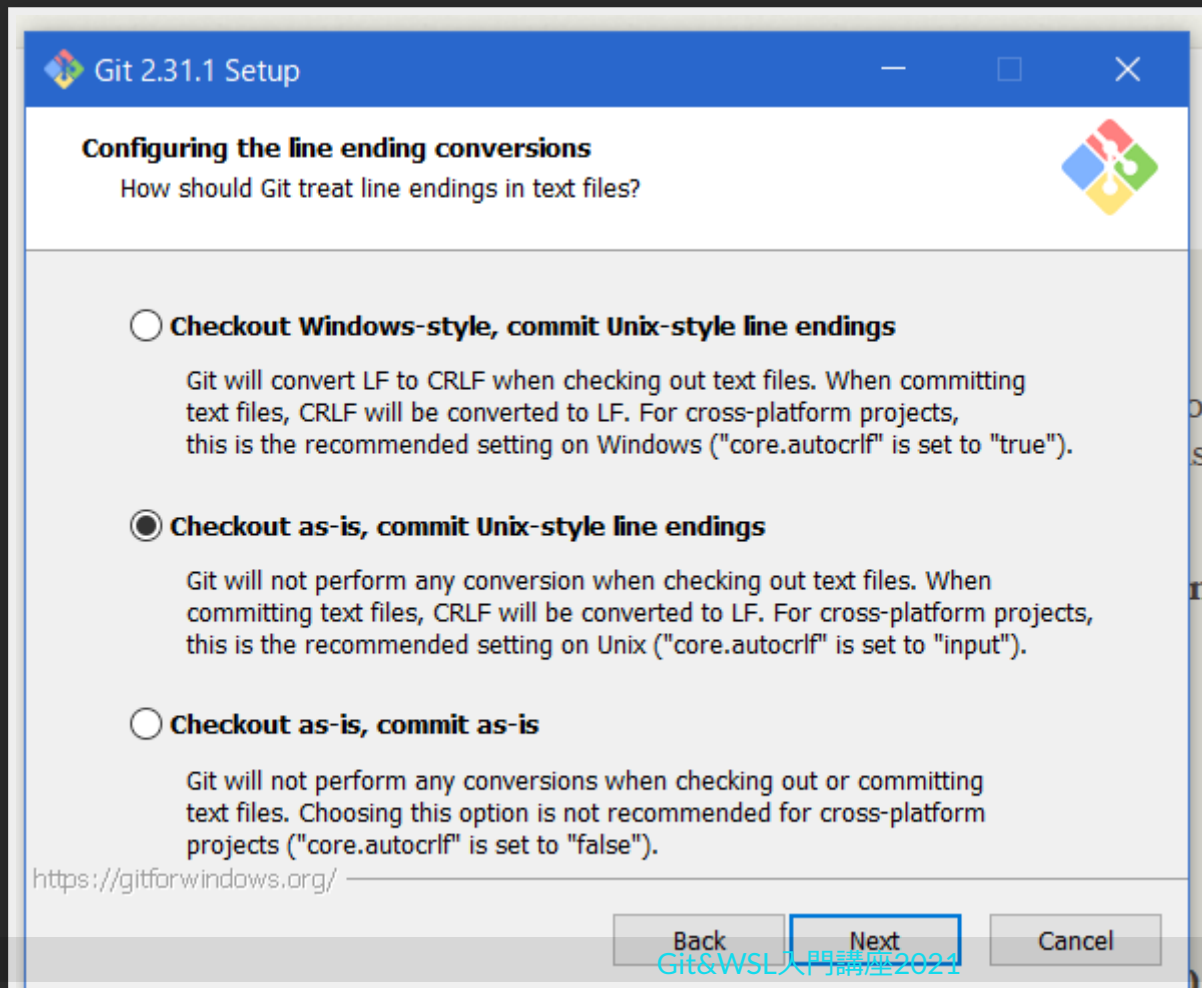
```
hidetomo8111f@DESKTOP-4P7E5SS:/mnt/c/Users/HYZIT/gitwsl-kouza$ git
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        <command> [<args>]
```

These are common Git commands used in various situations:

- Linuxのパッケージマネージャーから導入することも可能
- <https://git-scm.com/download/linux>

```
sudo apt install git
```

- Windowsの場合、以下の画面が出るまでに、ひたすら「Next」ボタンをクリックし続けられたい
- 改行コードはLFにする



Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?



Use MinTTY (the default terminal of MSYS2)

Git Bash will use MinTTY as terminal emulator, which sports a resizable window non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `wintpty` to work in MinTTY.

Use Windows' default console window

Git will use the default console window of Windows ("cmd.exe"), which works with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

<https://gitforwindows.org/>

Back

Next

Cancel

Gitコマンドを叩くと下記のように出力されればOK

```
:\Users\HZIT>git
sage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
      [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
      [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
      [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
      [--super-prefix=<path>] [--config-env=<name>=<envvar>]
      <command> [<args>]
```

these are common Git commands used in various situations:

ソースコードのダウンロード(CLONE)

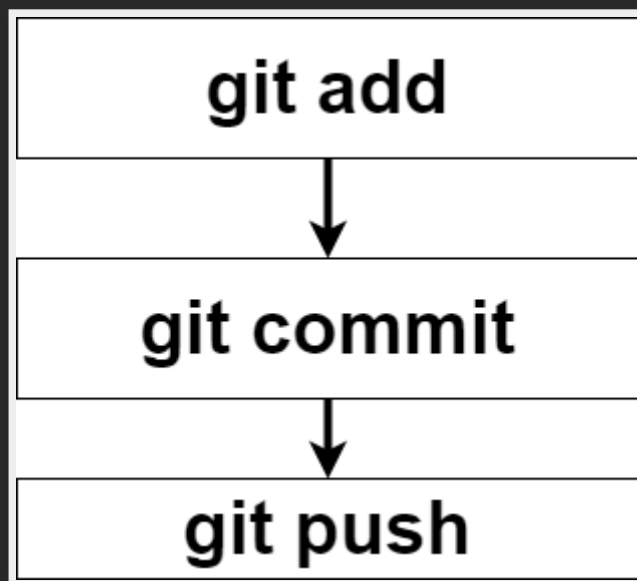
The screenshot shows the GitHub repository page for HarrisonKawagoe3960X / Kouza-Test. The 'Code' button is highlighted with a red circle and labeled '1'. The 'Clone' dropdown menu is open, showing the 'HTTPS' option with the URL `https://github.com/HarrisonKawagoe3960X/Kouza-Test` highlighted with a red circle and labeled '2'. The repository name 'Kouza-Test' is visible in the main content area.

- cloneコマンドを叩くと、リポジトリがダウンロードされる

```
git clone [url]
cd [リポジトリ名]
```

```
C:\Users\HYZIT>git clone https://github.com/HarrisonKawagoe3960X/Kouza-Test.git
Cloning into 'Kouza-Test'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```


ソースコードの変更が反映される まで



GIT ADD

- 変更したファイルを追加する時に使うコマンド

```
git add [ファイル名]
```

- [ファイル名]を.にすると、変更があるファイルを全部追加してくれるので、大半の場合は下記のコマンドが良い

```
git add .
```

GIT COMMIT

- 変更履歴を作成する時に使うコマンド

```
git commit -m "[Message]"
```

Vimなどのエディタの使い方がわかる人は、-mでメッセージを指定せずに、git commitを入力した後の次の画面でコミットメッセージを入力すれば良い

GIT PUSH

- 変更をリモートに反映するコマンド

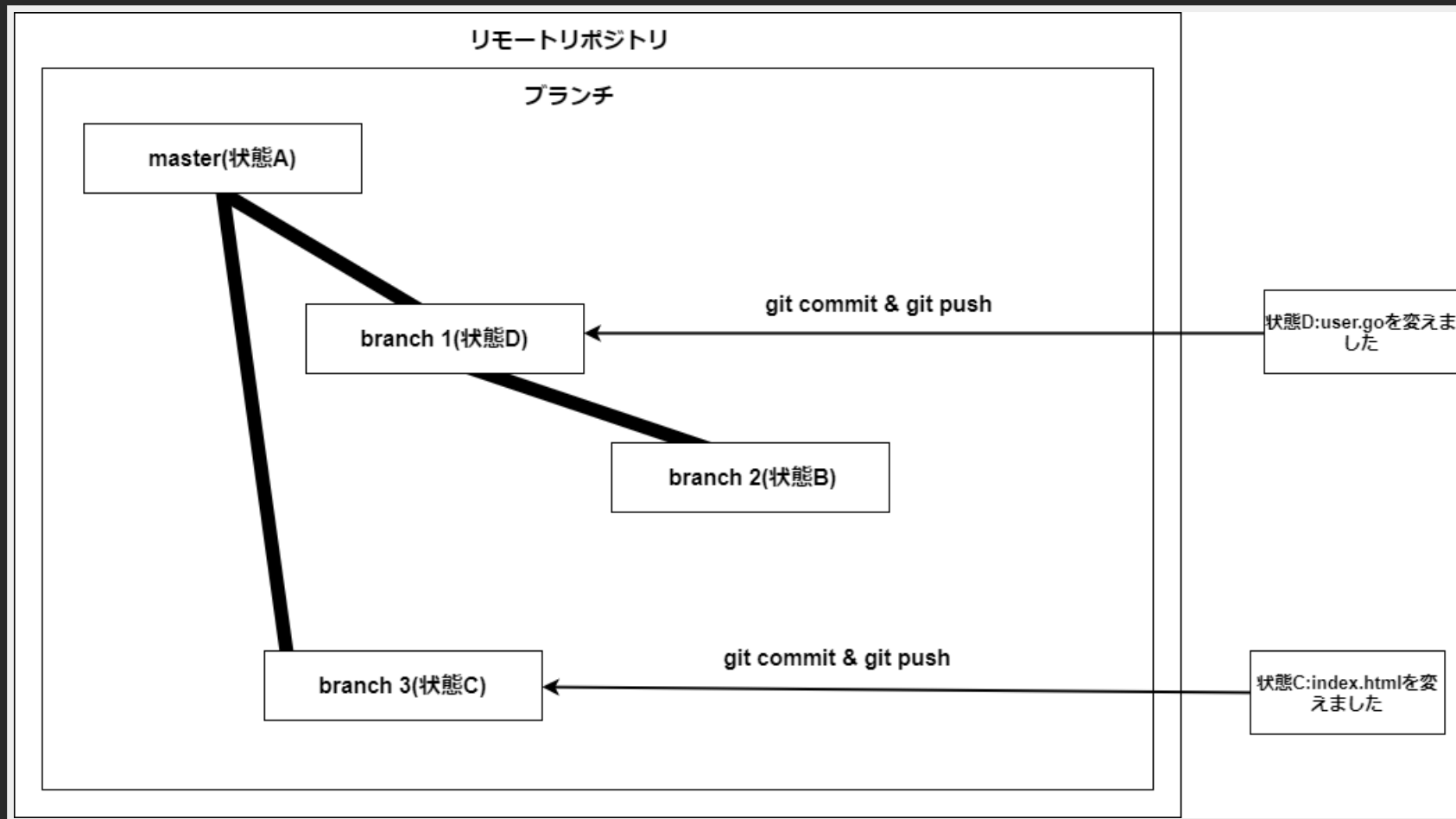
```
git push origin [ブランチ名]
```

既存のリポジトリで作業している場合は、git pushだけでもいける

ブランチ

- チーム開発で機能ごとに仕事を振り分けられる際によく使われる
- master(main)ブランチは本番用のコードを管理するのが一般的で、実際に作業する際はmaster以外のブランチでやったほうが管理しやすい

- 特定のブランチに変更をpushしたとしても、mergeコマンドを使わなければ、他のブランチに影響を与えることはない



- ブランチの作成

```
git checkout -b [ブランチ名]
```

- ブランチの切り換え

```
git checkout [ブランチ名]
```

- ブランチの確認

```
git branch
```

新しく作成したブランチで初めてpushする場合は、
下記のコマンドを入力する:

```
git push --set-upstream origin [ブランチ名]
```


gitとGitHubを試してみたい人は:
<https://github.com/HarrisonKawagoe3960X/Kouza-Test>

リモートの変更をローカルに反映する(PULL)

```
git pull origin [ブランチ名]
```

現在使用しているブランチの変更をローカルに反映する場合は、

```
git pull
```

だけでも問題ない

リモートの変更をローカルに反映する(PULL)

```
git pull origin [ブランチ名]
```

現在使用しているブランチの変更をローカルに反映する場合は、

```
git pull
```

だけでも問題ない

リモートの変更をローカルに保存する(FETCH)

```
git fetch origin [ブランチ名]
```

ただし、変更は現在作業しているローカルブランチに反映されない

他のブランチ(あるいはリモートブランチ)の変更をローカルブランチに反映する(MERGE)

```
git merge origin [ブランチ名]
```

現在使用しているブランチにmergeする場合は:

```
git merge
```

だけでも良い。

PULL, MERGE, FETCHの関係:

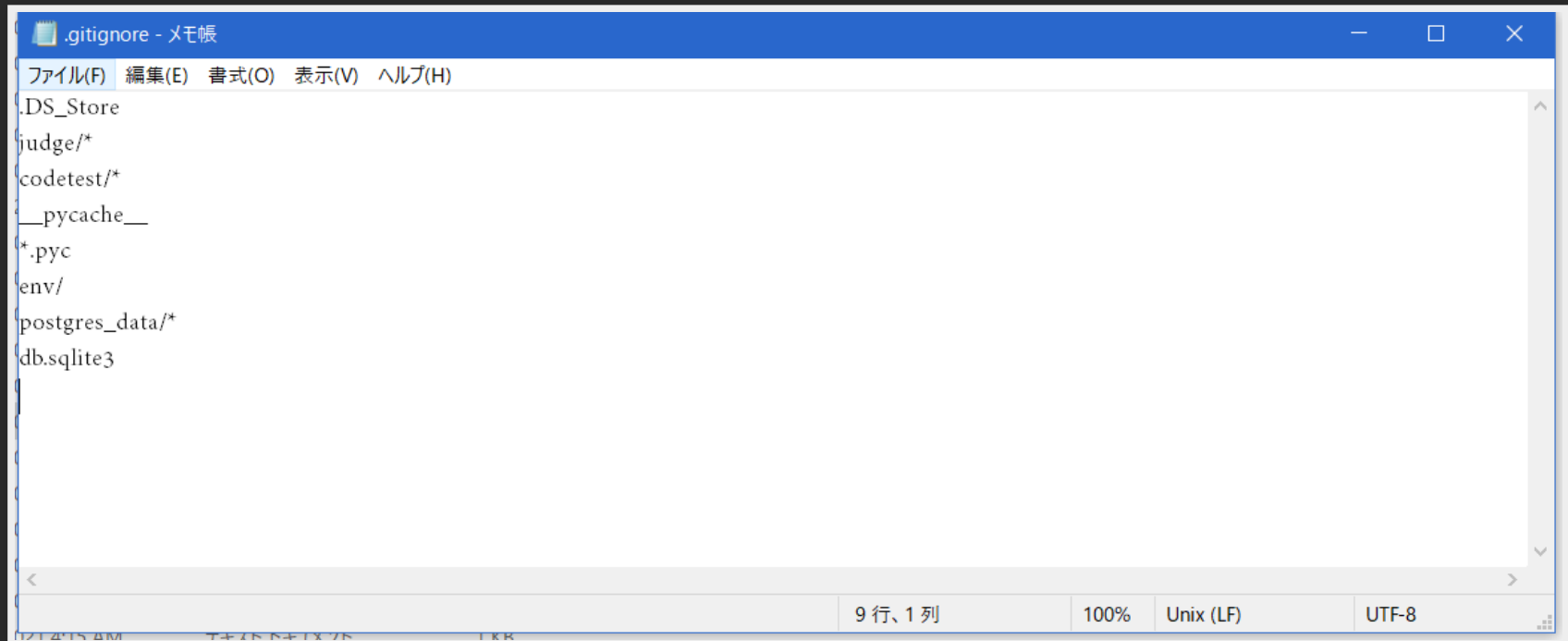
- Pull = Fetch + Merge

ロールバックしたい場合は:

```
git log  
git reset --hard [コミットid]
```

特定のファイルをバージョン管理から除外したい場合:

- .gitignoreファイルを作成して、除外したいファイルとフォルダーを記述すれば良い



```
.gitignore - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
.DS_Store
judge/*
codetest/*
__pycache__
*.pyc
env/
postgres_data/*
db.sqlite3

9行、1列 100% Unix (LF) UTF-8
```


MERGEが失敗した場合の対処法(コンフリクト):

- コンフリクトが起こったファイルの中では、下記のように記述されるので、採用するコードだけの残せばよい:

```
<<<<<<<HEAD
```

```
[ローカルブランチのコード]
```

```
=====
```

```
[マージするブランチのコード]
```

```
>>>>>>> [コミットID]
```

```
Shell
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare C
<<<<<<<HEAD (Current Change)
[ローカルブランチのコード]
=====
[マージするブランチのコード]
>>>>>>>[コミットID] (Incoming Change)
...
```

- エディタを使うと分かりやすい
- コンフリクトが発生したファイルが多い場合、GitHub Desktopを使うと直しやすい
- 必要のないファイルは(キャッシュやビルドファイルなど)はソースコード管理から外す



PULL REQUESTを使おう!

- 特定のブランチから変更をリモートにプッシュした場合、管理者は変更を採用したり、拒否したりすることができる
- 変更を採用した場合は、変更が指定されたブランチにMergeされる

折り畳みボタンの削除 #102

Edit Open with ▾

Merged HarrisonKawagoe... merged 1 commit into `develop` from `feature/hrsnkwge` on 8 Mar

Conversation 0 Commits 1 Checks 0 Files changed 1 +2 -3

HarrisonKawagoe3960X commented on 8 Mar
No description provided.

折り畳みボタンの削除 5900f7d

HarrisonKawagoe3960X merged commit `bdc65f8` into `develop` on 8 Mar [Revert](#)

Pull request successfully merged and closed
You're all set—the `feature/hrsnkwge` branch can be safely deleted. [Delete branch](#)

Write Preview H B I ≡ <> @ ↺ ↻

- Reviewers: No reviews
- Assignees: No one—assign yourself
- Labels: None yet
- Projects: None yet
- Milestone: No milestone



CODE REVIEWでソースコードをチェックしよう!

折り畳みボタンの別味 #102 Edit Open with ▾

Merged HarrisonKawagoe... merged 1 commit into develop from feature/hrsnkwe on 8 Mar

Conversation 0 Commits 1 Checks 0 Files changed 1 +2 -3

Changes from all commits ▾ File filter ▾ Jump to ▾ 0 / 1 files viewed Review changes ▾

templates/home/index.html 5

@@ -27,6 +27,7 @@ <h1 class="jumbotron-heading">みすみミュージアム</h1>	27 <p>検索したいキーワードを入力してください。</p>	27 <p>検索し
28 <input type="search" name="search" id="searchinput" placeholder="キーワードを入力" value="{{keyword}}">	28 <input ty	28 <input ty
29 <input type="submit" name="submit" value="検索">	29 <input ty	29 <input ty
30 <!-- 折り畳み展開ポイント -->	30 + {% comm	30 + {% comm
31 <div onclick="obj=document.getElementById('open').style; obj.display=	31 <!-- 折り	31 <!-- 折り
(obj.display=='none')?'block':'none';">	32 <div onc	32 <div onc
	33 <a sty	33 <a sty
@@ -35,7 +36,6 @@ <h1 class="jumbotron-heading">みすみミュージアム</h1>	36	36
35 <!-- 折り畳まれ部分 -->	37 <!-- 折り畳まれ部分 -->	37 <!-- 折り畳まれ部分 -->
36 <div id="open" style="display:none;clear:both;background-color: #eee">	38 <div id="open" style="display:none;clear:both;background-color: #eee">	38 <div id="open" style="display:none;clear:both;background-color: #eee">
37 {% comment %}	39 <h4>対象機種</h4>	39 <h4>対象機種</h4>
38 - {% comment %}	40 <p>	40 <p>
39 <h4>対象機種</h4>	41 <input type="checkbox" name="kisyu[]" value="windows">Windows	41 <input type="checkbox" name="kisyu[]" value="windows">Windows
40 <p>	42	42
41 <input type="checkbox" name="kisyu[]" value="windows">Windows	43	43
@@ -53,10 +53,9 @@ <h4>ゲームカテゴリ</h4>	53 <input type="radio" name="category" value="パズル">パズル	53 <input type="radio" name="category" value="パズル">パズル
52 <input type="radio" name="category" value="パズル">パズル	54 <input type="radio" name="category" value="音ゲー">音ゲー	54 <input type="radio" name="category" value="音ゲー">音ゲー
53 <input type="radio" name="category" value="音ゲー">音ゲー	55 </p>	55 </p>
54 </p>	56 - {% endcomment %}	56 - {% endcomment %}
55 </p>	57 -	57 -
56 - {% endcomment %}	58 </div>	58 </div>
57 -	59	59
58 </div>	60	60
59 <!-- 折り畳まれ部分 -->	61	61

Finish your review

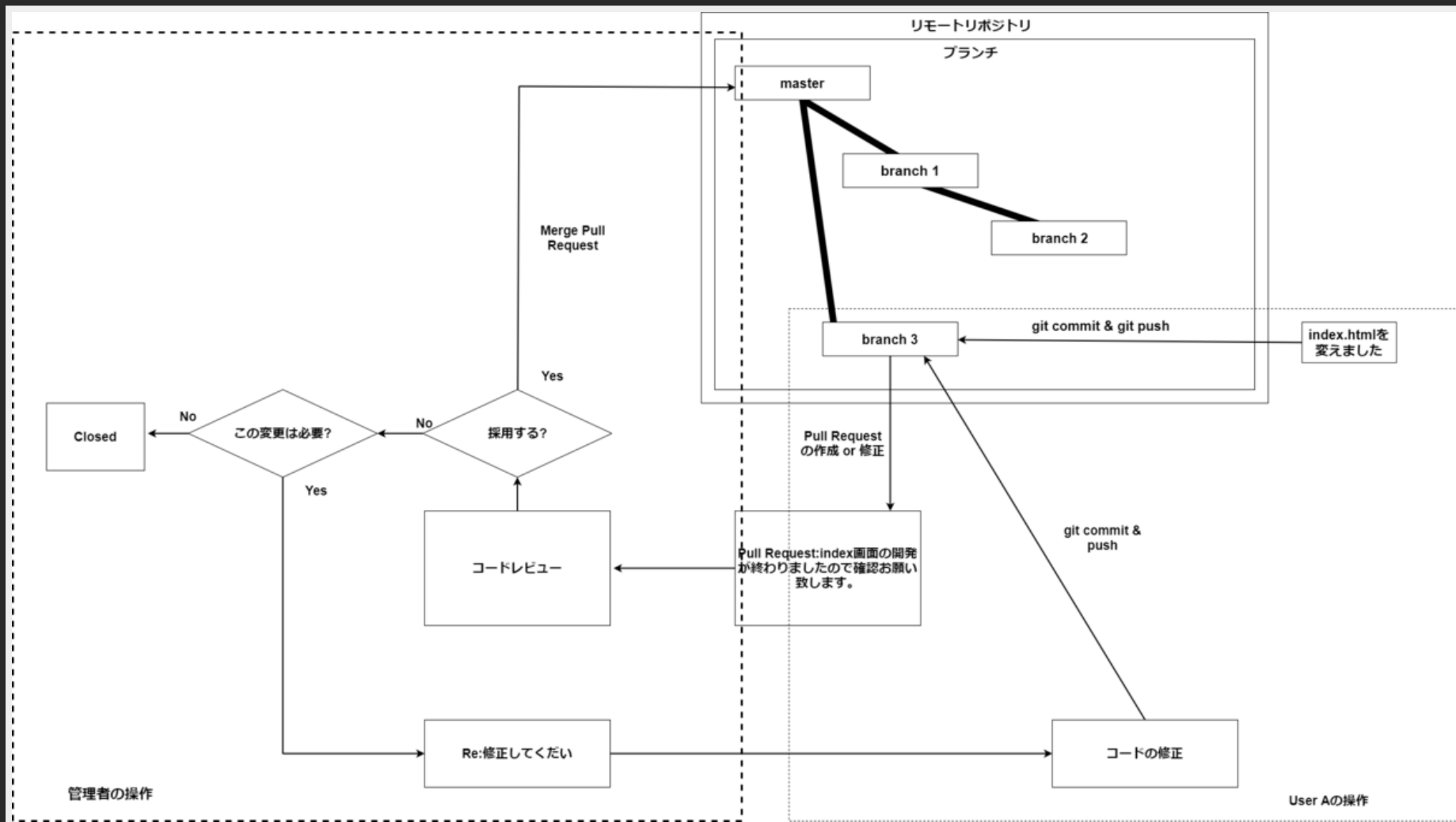
Write Preview H B I ≡ < > @ ↻

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Submit review





バグの報告、要望などがあれば ISSUEを投げよう!

管理者のログ #107 Edit New issue

Open HarrisonKawagoe3960X opened this issue on 8 Mar · 0 comments

HarrisonKawagoe3960X commented on 8 Mar

- 誰が公開設定を変えたのか
- 誰がユーザの情報を変えたのか

HarrisonKawagoe3960X added the backend label on 8 Mar

Write Preview **H B I**

Attach files by dragging & dropping, selecting or pasting them.

Assignees
No one—assign yourself

Labels
backend

Projects
None yet

Milestone
No milestone

Linked pull requests
Successfully merging a pull request may close this issue.
None yet

- オススメのチュートリアル:
<https://backlog.com/ja/git-tutorial/>

WSLは何?

- Windows Subsystem for Linux とは、Linuxのバイナリ実行ファイルをWindows 10およびWindows Server上でネイティブ実行するための実行環境である。(Wikipedia参照)

なぜWSL?

- 最近のWebアプリなどのプログラムは、LinuxあるいはUnixで実行するのが前提となっている
- GCCなど授業で使われるコンパイラやプログラムの多くも、Linux(Unix)環境が無いと実行できない
- 本当はネイティブのLinux環境が望ましいが、OSを入れるのはちょっとハードルが高い
- WSLはWindowsのファイルシステムと連携しているので、ファイルのアクセスが便利



導入方法

<https://docs.microsoft.com/ja-jp/windows/wsl/install-win10>




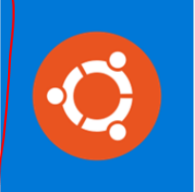







初めてLINUXを使うのであれば、 UBUNTUをPCに入れよう!

- 初心者優しい
- 利用人数が多い
- 情報系の授業でも勧められている

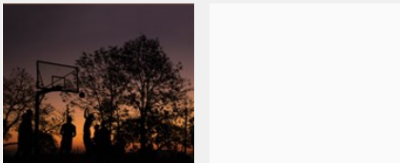
結果: Ubuntu

部門: すべての部署 | 次で使用可能: パーソナル コンピューター

アプリ (22) [すべて表示](#)

								
Ubuntu ★★★★★ 50 📄	Ubuntu 20.04 LTS ★★★★★ 7 📄	Ubuntu 18.04 LTS ★★★★★ 11 📄	¥4,700 割引 X410 ★★★★★ 9 📄	Raft WSL 📄	Virtual Machine. 📄	Linux Cheatsheet 📄	2buntu.com 📄	Bitco Easy 📄
インストール済み	インストール済み	インストール済み	¥5,850 ¥1,150	無料+	¥1,150	無料+	無料	¥700

映画 (1) [すべて表示](#)



よく使われるLINUXコマンド

- root権限の取得

```
sudo -s
```

- 指定したディレクトリへ移動

```
cd [フォルダ名]
```

- ファイルの作成

```
touch [ファイル名]
```

- ファイルの実行

```
./[ファイル名]
```

- <https://qiita.com/kenichit/items/a29e1254f289fd9c7df2>



APTの使い方

- アプリのインストール

```
sudo apt install [パッケージ名]
```

- パッケージ一覧を更新

```
sudo apt update
```

- インストールしたアプリの更新

```
sudo apt upgrade
```

- https://qiita.com/SUZUKI_Masaya/items/1fd9489e

- GCCなどのビルドツールの導入

```
sudo apt install build-essential
```

- Java8 : openjdk-8-jre
- Python3 : python3 python3-pip

オススメのエディタ

- Linuxではコマンドラインでファイルを変更することが多いので、自分に合うエディタを選ぶと作業効率が上がることもある。

- Vim:

<https://qiita.com/okamos/items/c97970ab34ff55ff3>

```
#include <stdio.h>

int main(void){
    printf("Hi!\n");
    return 0;
}
~
~
~
~
~
~
~
~
~
~
:wq!|
```

- nano:

```
GNU nano 4.8                               main.c
#include <stdio.h>

int main(void){
    printf("Hi!\n");
    return 0;
}
```

^G Get Help **^O** Write Out **^W** Where Is **^K** Cut Text **^J** Justify **^C** Cur Pos **M-U** Undo **M-A** Mark Text
^X Exit **^R** Read File **^_** Replace **^U** Paste Text **^T** To Spell **^_** Go To Line **M-E** Redo **M-6** Copy Text



ご清聴ありがとうございました!