

02 Linuxの利用

02 Linuxの利用

前回の補足

追加作業の補足（昨年度発生した事例 ubuntu22.04）

sudoが初期ユーザで利用できない場合

正常に使えない場合

端末でよく使用するコマンド

日本語入力の仕方

アップデート・インストール時の注意

手動アップデート

パッケージの検索

パッケージのインストール

試しに入れてみよう

パッケージのアンインストール

その他apt利用時のメッセージ

必須プログラムのインストール（git）

gitの初期設定

開発環境の準備

エディタ

vscodeの拡張機能インストール

Linuxコマンドの練習

lsコマンド

lsは非常によく使用する

C言語のサポート規格を調べる

前回の補足

追加作業の補足（昨年度発生した事例 ubuntu22.04）

sudoが初期ユーザで利用できない場合

(1) 利用しているユーザに、sudo権限（一時的に管理者権限でコマンドを実行する権限）を与える。

- 端末（Terminal）を起動する。
- 以下、ターミナル内で入力し実行（Enter）する。

```
yoshimura@ubuntu2204:~$ su -
パスワード:
root@ubuntu2204:~# gpasswd -a yoshimura sudo ←各自のユーザ名を記述
Adding user yoshimura to group sudo
root@ubuntu2204:~# exit ←もしくは [ctrl]+[d]
logout
yoshimura@ubuntu2204:~$
```

- この作業により、sudo コマンドが使用可能になる。
 - 一度、LogOut & Login を行う → 再起動の方が良い感じ
- うまく行かない場合は、仮想マシンの再起動を行う。

正常に使えない場合

- 上記操作で、正常に `sudo` が利用できる人と、利用できない人が存在している。
 - 別の方法で、強制的に利用可能にする
 - 現時点での原因は不明。
（原因が分かったら教えてほしい）

(2) 以下の作業を行う

```
$ su -
# visudo
```

- これにより、sudo設定ファイルが直接編集できる。
- 最下行までスクロール（マウスは使用できないので、カーソルキー↓等）する。

```
# See sudoers(5) for more information on "@include" directives:
```

```
@includedir /etc/sudoers.d
```

ユーザ名 ALL=(ALL:ALL) ALL ←このように入力する

- [ユーザ名]は、各自が作成したユーザ名を設定する。
- `ctrl` + `s` で保存
- `ctrl` + `x` で終了

```
# exit ←もしくは [ctrl]+[d]
```

```
$ sudo command etc.
```

- この直後から、`sudo` コマンドが使用可能になる。
- この方法は、ユーザを直接登録する方法だが、100人とか1000人とかを個別に登録するのは非現実的。
 - そこで、本来の方法はグループを作成し、そのグループのユーザに対して許可を与える方法を用いている。

端末でよく使用するコマンド

- 設定する時、設定後、良く使用するもの、一連の作業で利用するコマンドをまとめておきます。

【通常よく使用するコマンド】

コマンド	意味	注意事項
sudo	管理者権限で実行	パスワード入力時に何も表示されない 一定時間、パスワードの再入力省略される
apt	ソフトウェアパッケージの管理ツール (現方式)	関連するパッケージのダウンロード・インストールを行なってくれる。 削除にも使用する
dpkg	ソフトウェアパッケージの管理ツール (旧方式)	指定したパッケージのみ操作 (インストール・削除等) を行う。 関連するパッケージは自動で取得しない。
ls	ファイルのList (一覧) を表示する	Windowsでのdirコマンドに相当 ファイルやディレクトリの一覧を表示する
exit	端末 (Terminal) を抜ける	端末での処理を終えて、ウインドウを閉じる 文字未入力時に <code>ctrl + d</code> でも同じ動作

コマンド	意味	注意事項
cd	ディレクトリ場所の変更 (change directoryの略)	現在居る場所を変更する cd のみで homeディレクトリに移動する (/home/username/) <code>~</code> (チルダ) はhomeディレクトリを意味する。
cp	ファイル・ディレクトリのCopy	Windowsでのcopyコマンドに相当。コピー
mv	ファイル・ディレクトリのMove、名前変更	Windowsでのmoveコマンドに相当。移動・名前の変更
rm	ファイル・ディレクトリのRemove	Windowsでのdelコマンドに相当。削除
df	ディスク(Disk)の空き容量(Free)を確認	たまには使って確認してほしい

コマンド	意味	注意事項
grep	文字列のフィルター	フィルターコマンド。指定した条件の文字列を含む行を出力する
less	ページャ（表示ツール）	入力されたデータを表示する（スクロール・検索なども可） 抜けるには <code>q</code> をtypeする
history	コマンドの実行履歴	1000行分記録する。記録から実行することもできる。
man	マニュアル表示	コマンドやファイルの説明書を見ることができる

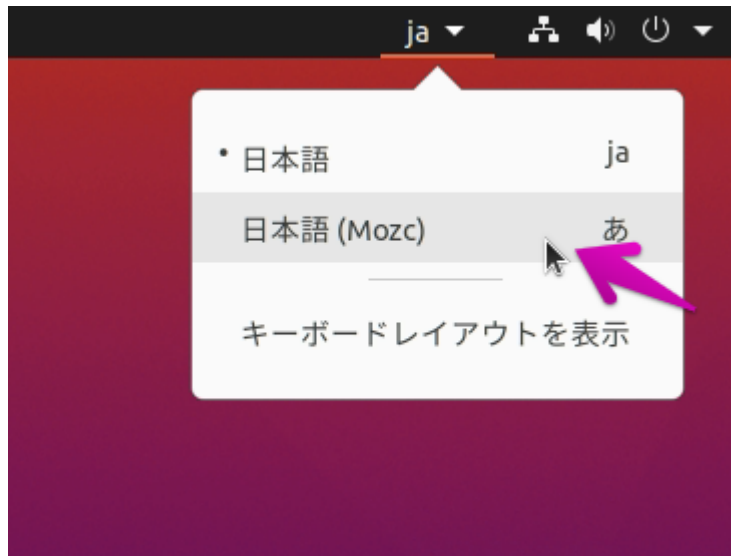
- 詳しい使い方は、各コマンドの後ろに `--help` を付けることで、表示されるものが大半
- `--help` で表示されない場合は、`man` を使用する

【1度設定すると以後あまり使用しないコマンド】

コマンド	意味	注意事項
passwd	パスワードの変更	パスワードを設定・変更する
ssh-keygen	公開鍵暗号方式の鍵を生成する	秘密鍵と公開鍵を生成する。 秘密鍵は絶対に人に見せない
ssh-copy-id	公開鍵をサーバに登録する	パスワード認証が可能な時しか使えない

日本語入力の仕方

- 画面右上に「ja」が表示されている場合は、キーボード通りの入力（英数字等）のみとなる。
- 「あ」「A」と表示されている場合は、日本語変換プログラム Mozcが動作している。
 - [半角/全角]キーで日本語入力のOFF/ON切り換えが可能
 - `Win` (Windows Key)+ `Space` (スペース)で、「日本語」・「日本語(Mozc)」の切り換えが可能
 - 詳細設定はGUIツールが用意されている。



アップデート・インストール時の注意

- 同時に複数のアップデートやインストールは実行できない。
 - アップデートに時間がかかっているからといって、別の端末でインストールすることはできない。
- 起動後しばらくの間は、裏で自動的に更新をチェック&インストールしている。
 - この間はアップデートやインストールができない。
- 学内で実施するときは、回線への負荷を考え、一斉に更新作業等を行わないようにする。
- 気になるパッケージ等があれば、調べてインストールして試してみると良い
 - Ubuntu上 (vm) の空きディスク容量に注意
 - Windows側 (host) の空きディスク容量にも注意

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           391M  1.5M  390M   1% /run
/dev/sda3       49G   14G   33G  31% /      ← 【ディスク全体】
tmpfs           2.0G    0  2.0G   0% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
/dev/sda2       512M  6.1M  506M   2% /boot/efi
tmpfs           391M   88K  391M   1% /run/user/128
tmpfs           391M  112K  391M   1% /run/user/1000
```

- 「-h」は人間に読みやすい単位で表示するオプション
 - /dev/sd〜 は、物理的なドライブを表している。
 - tmpfs は、OSが作成した一時作業領域

【結果から必要な行だけを表示する】

```
$ df -h | grep "dev"
/dev/sda3          49G   14G   33G   30% /
tmpfs              2.0G    0   2.0G    0% /dev/shm
/dev/sda2          512M   6.1M  506M    2% /boot/efi
```

- `"` は、空白を含まない場合省略可（以下同じ結果）

```
$ df -h | grep dev
/dev/sda3          49G   14G   33G   30% /
tmpfs              2.0G    0   2.0G    0% /dev/shm
/dev/sda2          512M   6.1M  506M    2% /boot/efi
```

【結果から不必要な行を非表示にする】

- `-v` は指定した文字列を含まない行（Notの意味）

```
$ df -h | grep -v tmpfs
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3        49G   14G   33G   30% /
/dev/sda2        512M   6.1M  506M    2% /boot/efi
```

手動アップデート

<https://atmarkit.itmedia.co.jp/ait/articles/1708/31/news017.html>

```
$ sudo apt update
$ sudo apt upgrade
```

- 1行目で、ローカルに保存されている各パッケージの一覧情報（カタログ）を、ネット上の最新の情報に更新する。
- 2行目で、マシンにインストールされているパッケージと、カタログの一覧情報から、最新のパッケージに更新する。
 - 2行目を実行しないと、マシンの状況は変更されない。
- 随時行なっておいください

パッケージの検索

```
$ apt search キーワード
```

- 検索は一般ユーザで実行可能（sudoが不要）
- 出力行が多い場合は、lessを使用すると良い

```
$ apt search game | less
```

- 出力結果をlessプログラムに渡す
- lessの終了は `q` を押す。
- `↓` , `↑` , `PageUp` , `PageDown` でスクロールできる。 (スペース) で1画面ずつスクロールする。
 - lessの使い方は別途説明する

パッケージのインストール

```
$ sudo apt install パッケージ名
```

- ただし、正しく入力してもパッケージ名が見つからないようなメッセージが出る場合は、一覧情報が古い場合がある。
 - install前に、手動アップデート `sudo apt update` を実施する癖を付けると良い

```
$ sudo apt update
$ sudo apt install パッケージ名
```

試しに入れてみよう

- ASCII ARTを生成するコマンド `figlet` を入れてみる

```
$ sudo apt install figlet
$ figlet Hello
```

パッケージのアンインストール

- aptでインストールした場合は、簡単に削除できる

```
$ sudo apt remove パッケージ名
```

- なお、たまにautoremoveを実行するよう表示が出る場合

```
これを削除するには 'sudo apt autoremove' を利用してください。
```


- すでに不必要になったパッケージを実際に削除する

```
$ sudo apt autoremove
```

その他apt利用時のメッセージ

aptを実行したときに、様々なメッセージが表示されるので良く確認して欲しい

表示するには 'apt list --upgradable' を実行してください。

- そのまま、上記を実行すれば良い

```
$ sudo apt list --upgradable
```

apt-get update を実行するか --fix-missing オプションを付けて試してみてください。

- apt-get は apt の昔の名前なので、以下のコマンドをどちらか実行する

```
$ sudo apt-get update --fix-missing
$ sudo apt update --fix-missing
```

```
yoshimura@x390:~$ sudo apt upgrade
[sudo] yoshimura のパスワード:
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is he
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is he
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is he
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is he
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is held by
process 3463 (apt)... 4s
```

- 他のプロセスでaptが実行されているので、それが終わるまで待っている旨のメッセージ。
- このまま、待ち続けるか、`ctrl + c` で中断する。
 - 中断した場合は、後ほど同じ命令を再実行する

その他、見なれないメッセージが出た場合は、直接相談するか、ググってください。

必須プログラムのインストール (git)

- バージョン管理ツールgitをなるべく利用しよう

```
$ git
コマンド 'git' が見つかりません。次の方法でインストールできます:
sudo apt install git
```

- このように、利用できない場合はインストールパッケージを教えてくれる。

```
$ sudo apt install git
```

【実行例】

```
$ sudo apt install git
[sudo] yoshimura のパスワード:
パッケージリストを読み込んでいます... 完了
依存関係ツリーを作成しています... 完了
状態情報を読み取っています... 完了
以下の追加パッケージがインストールされます:
  git-man liberror-perl
提案パッケージ:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-
mediawiki git-svn
以下のパッケージが新たにインストールされます:
  git git-man liberror-perl
アップグレード: 0 個、新規インストール: 3 個、削除: 0 個、保留: 2 個。
4121 kB のアーカイブを取得する必要があります。
この操作後に追加で 20.9 MB のディスク容量が消費されます。
続行しますか? [Y/n]
取得:1 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 liberror-perl all 0.17029-1
[26.5 kB]
取得:2 http://gb.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git-man all 1:2.34.1-
1ubuntu1.8 [953 kB]
取得:3 http://gb.archive.ubuntu.com/ubuntu jammy-updates/main amd64 git amd64 1:2.34.1-
1ubuntu1.8 [3141 kB]
4121 kB を 3秒 で取得しました (1433 kB/s)
以前に未選択のパッケージ liberror-perl を選択しています。
(データベースを読み込んでいます ... 現在 217076 個のファイルとディレクトリがインストールされています。)
.../liberror-perl_0.17029-1_all.deb を展開する準備をしています ...
liberror-perl (0.17029-1) を展開しています...
以前に未選択のパッケージ git-man を選択しています。
.../git-man_1%3a2.34.1-1ubuntu1.8_all.deb を展開する準備をしています ...
git-man (1:2.34.1-1ubuntu1.8) を展開しています...
以前に未選択のパッケージ git を選択しています。
.../git_1%3a2.34.1-1ubuntu1.8_amd64.deb を展開する準備をしています ...
git (1:2.34.1-1ubuntu1.8) を展開しています...
liberror-perl (0.17029-1) を設定しています ...
git-man (1:2.34.1-1ubuntu1.8) を設定しています ...
```

```
git (1:2.34.1-1ubuntu1.8) を設定しています ...
man-db (2.10.2-1) のトリガを処理しています ...
```

gitの初期設定

```
$ git config --global user.name "自分の名前"
$ git config --global user.email 自分のEメールアドレス
$ git config --global init.defaultBranch main
```

- 名前にスペースが含まれない場合は「"」は省略可
- 初期のブランチ名を「main」に設定する
 - 少し前までは「master」で固定（というルール）されていた。
世界の言葉に対する反応への対応。

<https://phoeducation.work/entry/20210720/1626735480>


開発環境の準備

エディタ


- 授業では、vscodeを利用する。
 - 他のエディタを使用したければ使用しても良い。（自己責任で）
 - vi (or neovim)
 - emacs
 - nano (ubuntu標準の簡易Editor：ターミナル用)
 - gedit (ubuntu標準のEditor：GUI用)
 - <https://code.visualstudio.com/download> からダウンロード

Download Visual Studio Code


Free and built on open source. Integrated Git, debugging and extensions.


↓ Windows
Windows 8, 10, 11

User Installer [x64](#) [x86](#) [Arm64](#)
System Installer [x64](#) [x86](#) [Arm64](#)
.zip [x64](#) [x86](#) [Arm64](#)
CLI [x64](#) [x86](#) [Arm64](#)

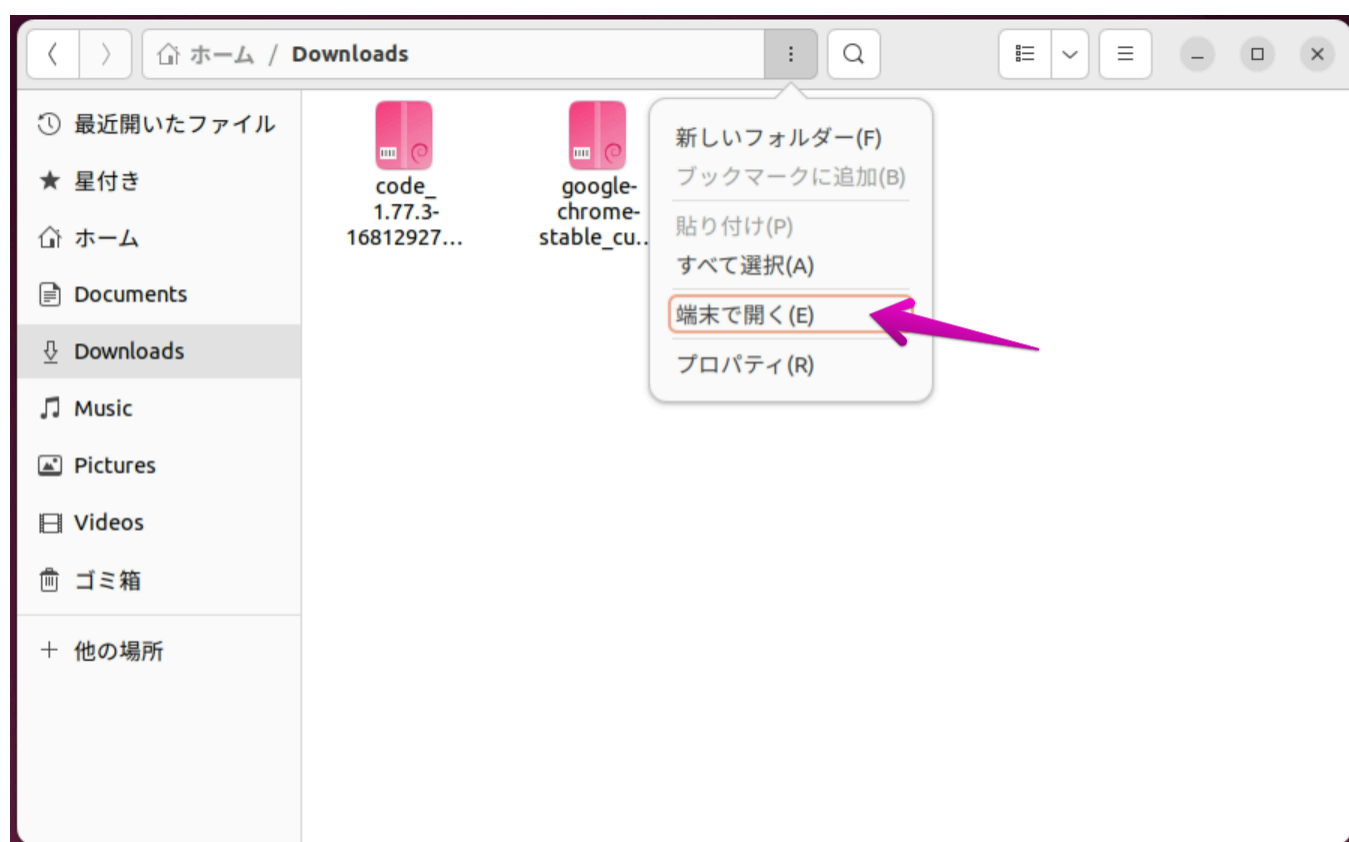
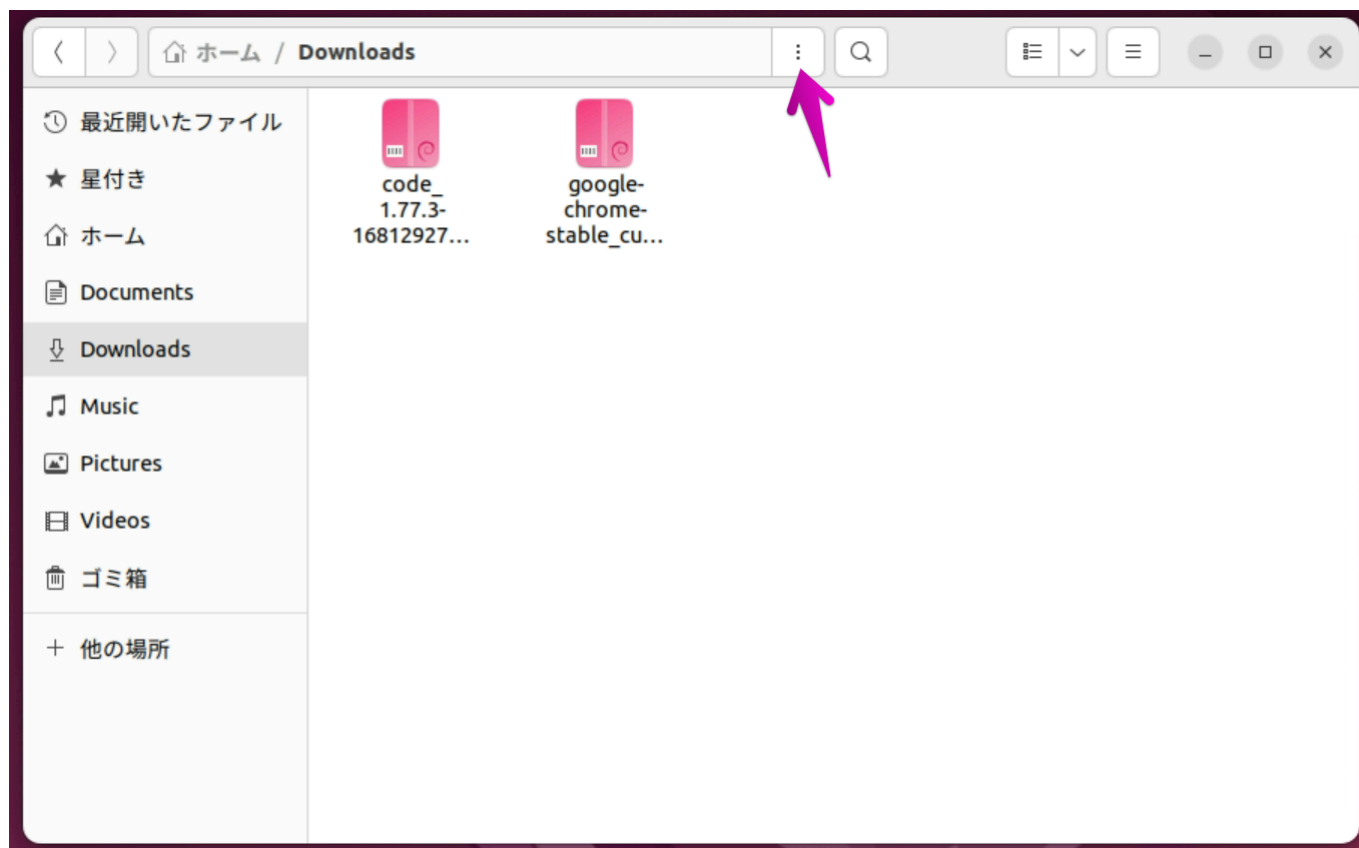

↓ .deb **↓ .rpm**
Debian, Ubuntu Red Hat, Fedora, SUSE

.deb [x64](#) [Arm32](#) [Arm64](#)
.rpm [x64](#) [Arm32](#) [Arm64](#)
.tar.gz [x64](#) [Arm32](#) [Arm64](#)
Snap [Snap Store](#)
CLI [x64](#) [Arm32](#) [Arm64](#)


↓ Mac
macOS 10.11+

.zip [Intel chip](#) [Apple silicon](#) [Universal](#)
CLI [Intel chip](#) [Apple silicon](#)

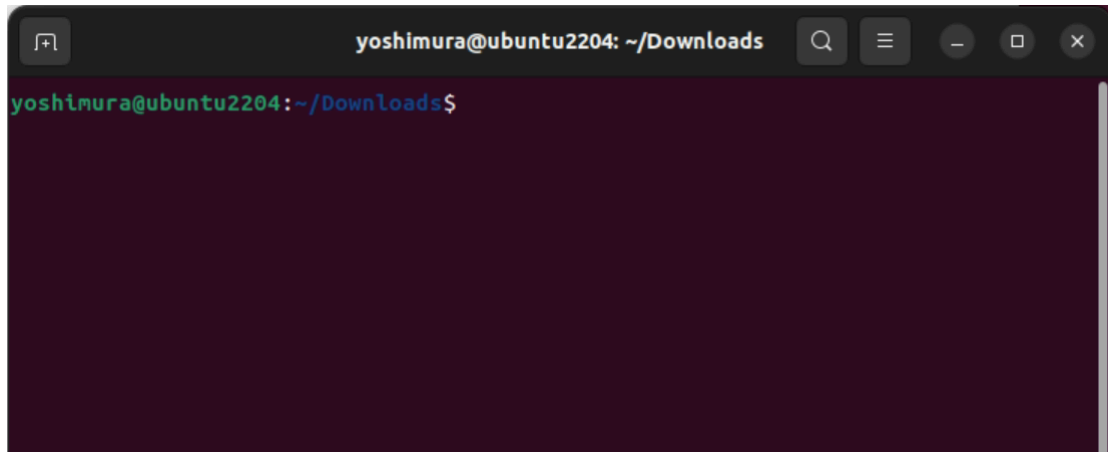
- Linux用には、複数の形式が用意されていることが多い
 - Ubuntuの場合は、`.deb` 形式を選択する
- 【方法1】「ファイル」を利用し該当のファイルを確認する。
 - 「ダウンロード」フォルダに保存されている
 - 上部「Downloads」右の「:」から「端末で開く」を選択



- 【方法2】「ファイル」を利用し該当のファイルを確認する。
 - 「ダウンロード」フォルダに保存されている

- 何もないところで右クリック「端末で開く」を選択

- 端末が開かれた状態



- 【方法3】ダウンロードファイルを保存後、直接端末内でコマンドを使って移動しても良い（上図と同じ状態になる）

```
$ cd ~/Downloads
```

- 「~」は特別な記号で、自分のhomeディレクトリを意味する。
- 「~/Downloads」は、具体的には、「/home/username/Downloads」を意味する

```
$ ls
```

- ちゃんとダウンロードしたファイルが保存されているかの確認をする。

```
$ ls
code_1.99.0-1743632463_amd64.deb  google-chrome-stable_current_amd64.deb
```

- 実際のインストール

```
$ sudo apt install ./code_1.99.0-1743632463_amd64.deb
```

- 入力は補完機能を利用する（手入力すると間違えることが多く、エラーを生じやすい）
 - `TAB` キーを利用する
- 【操作手順】

```
$ sudo apt in[TAB]
```

↓

```
$ sudo apt install
```

↓

```
$ sudo apt install ./c[TAB]
```

↓

```
$ sudo apt install ./code_1.99.0-1743632463_amd64.deb
```

- 最後に `Enter` キーを押して実行する。

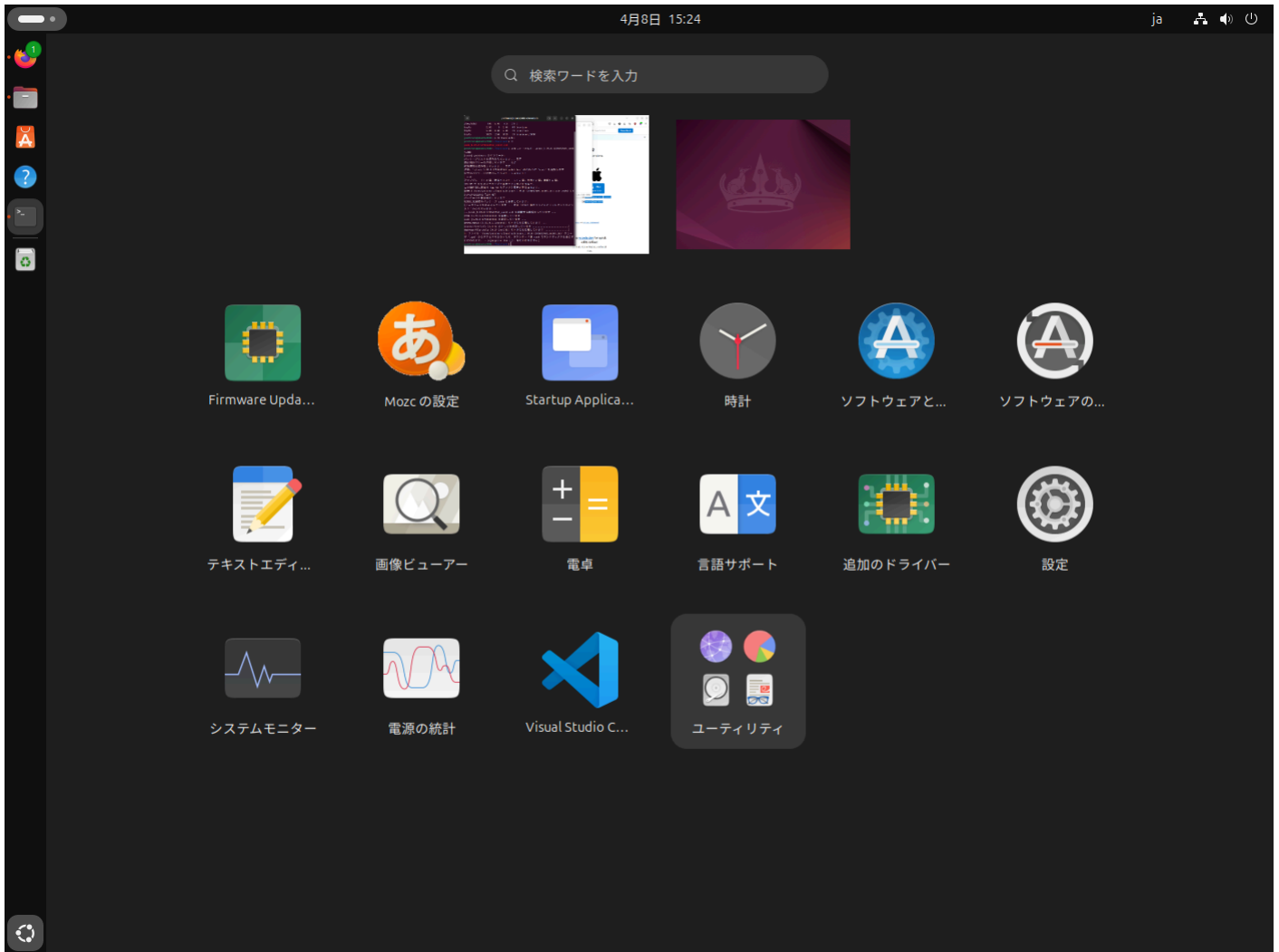
。以下、実行例

```
yoshimura@ubuntu2404:~/Downloads$ sudo apt install ./code_1.99.0-1743632463_amd64.deb
[sudo] yoshimura のパスワード:
パッケージリストを読み込んでいます... 完了
依存関係ツリーを作成しています... 完了
状態情報を読み取っています... 完了
注意、 './code_1.99.0-1743632463_amd64.deb' の代わりに 'code' を選択します
以下のパッケージが新たにインストールされます:
  code
アップグレード: 0 個、新規インストール: 1 個、削除: 0 個、保留: 1 個。
104 MB 中 0 B のアーカイブを取得する必要があります。
この操作後に追加で 423 MB のディスク容量が消費されます。
取得:1 /home/yoshimura/Downloads/code_1.99.0-1743632463_amd64.deb code amd64
1.99.0-1743632463 [104 MB]
パッケージを事前設定しています ...
以前に未選択のパッケージ code を選択しています。
(データベースを読み込んでいます ... 現在 153047 個のファイルとディレクトリがイン
ストールされています。)
.../code_1.99.0-1743632463_amd64.deb を展開する準備をしています ...
code (1.99.0-1743632463) を展開しています...
code (1.99.0-1743632463) を設定しています ...
gnome-menus (3.36.0-1.1ubuntu3) のトリガを処理しています ...
shared-mime-info (2.4-4) のトリガを処理しています .....]
desktop-file-utils (0.27-2build1) のトリガを処理しています .....]
N: ファイル './code_1.99.0-1743632463_amd64.deb' がユーザ
'_apt' からアクセスできないため、ダウンロードは root でサンドボックスを通さずに行われます。 -
pkgAcquire::Run (13: 許可がありません)
```

- 。表示されるメッセージにエラー等無ければOK。
- 。以下のように表示される場合があるが、これは無視してOK

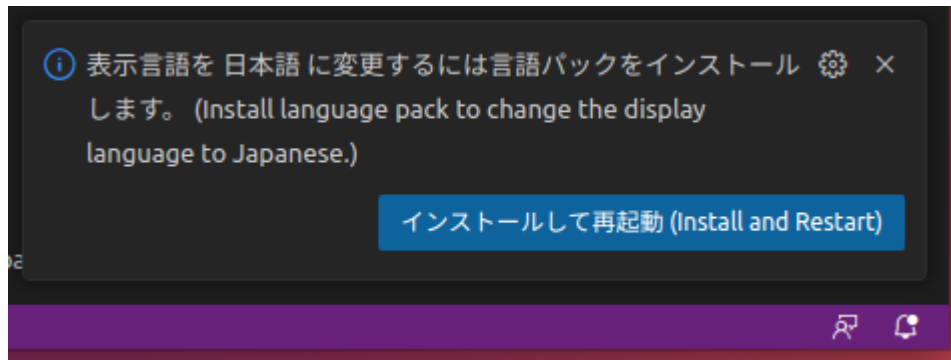
```
N: ファイル '/home/yoshimura/Downloads/code_1.88.0-1712152114_amd64.deb' がユーザ '_apt' からアクセスできないため、ダウンロードは root でサンドボックスを通さずに行われます。 - pkgAcquire::Run (13: 許可がありません)
```

- このメッセージは、Notice で、エラーではなく、無視して良い
(2016年ころからの不具合だが、実質問題がないため放置されている)
- 次のように「Visual Studio Code」のアイコンが表示される→起動できる



vscodeの拡張機能インストール

- 起動後「Extentions」を選択する
- 「日本語パッケージ」のインストール確認が表示されるので、インストールする。
 - メッセージを消してしまった場合は、手動で入れることが可能（次項）



- 以下の拡張機能を導入（最低限の拡張機能）
 - Japanese Language Pack for Visual Studio Code（上記日本語と同じ）
 - これを入れると「Restart」（再起動）が必要になる→Restartする
 - 起動すると最初にインストールされるはず
 - Indent-rainbow
 - Git Graph
 - C/C++
- さまざまな拡張機能が存在するので、少しずつ試して欲しい。
 - 【参考】吉村がC言語用プロファイルに入れている拡張機能
 - Bookmarks
 - Cmake Tools
 - Doxygen Documentation Generator
 - Edit CSV
 - Hex Editor
 - Material Icon Theme
 - Zenkaku-Hankaku
 - AI用
 - Gemini Code Assist
 - GitHub Copilot
 - Phind
 - Markdown用
 - Markdown All in One
 - Markdown PDF
 - Markdown Table

- Markuplint
- Marp for VS Code
- SSH（リモート接続用）
 - Remote SSH
 - Remote - SSH: Editing Configuration Files
 - Remote Explorer

Linuxコマンドの練習

lsコマンド

- ファイルの一覧を表示する

```
yoshimura@ubuntu2204:~$ ls
Desktop    Downloads  Pictures   Templates  snap
Documents  Music      Public     Videos
```

- `-l` オプションの利用
 - long format（詳細情報）で表示する

```
yoshimura@ubuntu2204:~$ ls -l
合計 36
drwxr-xr-x 2 yoshimura yoshimura 4096 Apr 14 13:30 Desktop
drwxr-xr-x 2 yoshimura yoshimura 4096 Apr 14 13:30 Documents
drwxr-xr-x 3 yoshimura yoshimura 4096 Apr 19 18:35 Downloads
drwxr-xr-x 2 yoshimura yoshimura 4096 Apr 14 13:30 Music
drwxr-xr-x 2 yoshimura yoshimura 4096 Apr 14 13:30 Pictures
drwxr-xr-x 2 yoshimura yoshimura 4096 Apr 14 13:30 Public
drwxr-xr-x 2 yoshimura yoshimura 4096 Apr 14 13:30 Templates
drwxr-xr-x 2 yoshimura yoshimura 4096 Apr 14 13:30 Videos
drwx----- 4 yoshimura yoshimura 4096 Apr 14 13:41 snap
```

- `-a` オプションの利用
 - all（隠しファイルも全て）を表示する
 - Linuxでは、先頭が`.`（ドット）のファイル・ディレクトリは隠し属性となる。

```
.          .gitconfig      .sudo_as_admin_successful  Downloads
..         .gnupg        .vboxclient-clipboard.pid   Music
.bash_history .lessht       .vboxclient-draganddrop.pid  Pictures
.bash_logout .local        .vboxclient-seamless.pid     Public
.bashrc      .pam_environment .vboxclient-vmsvg-session-tty2.pid Templates
.cache       .pki          .vscode                      Videos
.config      .profile      Desktop                      snap
.dotnet      .ssh          Documents
```

- `-l` と `-a` を両方指定する

【パターン1】

```
yoshimura@ubuntu2204:~$ ls -l -a
合計 120
drwxr-x--- 19 yoshimura yoshimura 4096 Apr 19 18:35 .
drwxr-xr-x  3 root      root      4096 Apr 14 13:26 ..
-rw-----  1 yoshimura yoshimura 3154 Apr 19 18:28 .bash_history
-rw-r--r--  1 yoshimura yoshimura  220 Apr 14 13:26 .bash_logout
-rw-r--r--  1 yoshimura yoshimura 3771 Apr 14 13:26 .bashrc
:
```

【パターン2】

```
yoshimura@ubuntu2204:~$ ls -la
合計 120
drwxr-x--- 19 yoshimura yoshimura 4096 Apr 19 18:35 .
drwxr-xr-x  3 root      root      4096 Apr 14 13:26 ..
-rw-----  1 yoshimura yoshimura 3154 Apr 19 18:28 .bash_history
-rw-r--r--  1 yoshimura yoshimura  220 Apr 14 13:26 .bash_logout
-rw-r--r--  1 yoshimura yoshimura 3771 Apr 14 13:26 .bashrc
:
```

lsは非常によく使用する

- そのため、別の名前が用意されている（コマンドではなく、別名：エイリアスが設定されている）

コマンド	エイリアス	補足
ls -CF	l	リストを常に複数の列で出力する
ls -A	la	-aから「.」「..」を除く
ls -alF	ll	-F：名前の後にタイプ識別子（ <code>* / => @ </code> のいずれか）を付けて出力する

<https://atmarkit.itmedia.co.jp/ait/articles/1606/27/news018.html>

C言語のサポート規格を調べる

- Linuxでは、gccもしくはclangが使用できる。
 - gccは標準でインストール済み

<https://ja.wikipedia.org/wiki/GNU%E3%82%B3%E3%83%B3%E3%83%91%E3%82%A4%E3%83%A9%E3%82%B3%E3%83%AC%E3%82%AF%E3%82%B7%E3%83%A7%E3%83%B3>

```
$ gcc -v --help 2>/dev/null | grep -E "^\s+\-std=.*$" | less
```

- `less` の終了は `q`