# 概要

Linuxのソフトウェア情報の内，GUI（デスクトップアプリ）色の強いものはこちらに載せる。

（それ以外は基本的に別ドキュメント Linux に載せるものとする）

2023-05-18　新たにOSSをいう別ドキュメントを作成。ソフトウェア情報に関してはこのドキュメントと重複する可能性がある

日本語版 isoのダウンロード： <https://www.ubuntulinux.jp/products/JA-Localized/download>

# セットアップ

## インストール

### UbuntuでBootableUSBを作る

方法はいくつかあるらしい

方法１：DDを使う

・.isoイメージをダウンロードしてddもしくはEtherで書き込む

方法２：startupを使う

（アプリケーションの表示）→ 検索ボックスに Startupと入力

またはTerminalからusb-creator-gtkと入力しても良い。

その他 → ISOイメージを選択 → ブータブルUSBの作成

イメージはrootパーティションと同じパーティションに置く必要がある？仮想マシンからホストOSのドライブにあるISOイメージを選択したら作成時にエラーが出て先に進めなかった。

（公式情報より）[https://ubuntu.com/tutorials/create-a-usb-stick-on-ubuntu#1-overview](https://ubuntu.com/tutorials/create-a-usb-stick-on-ubuntu" \l "1-overview)

方法３：

ユーティリティ → ディスク →  → ディスクイメージをアタッチ → .isoイメージを選択

2021年10月14日

実験１ LiveUSBでファイルシステムを大きくする

・ 何もしなくても最初から8GBのパーティションができた。但しUSB総容量は32GBなので，なぜ8GBのなのかは不明。

・作成されたLiveUSBのファイルシステムはiso9660。ファイルシステムの変更はできない？

・

実験２　仮想マシンでインストール

Virtual Machineでは画面が見切れて、先に進むボタンが押せなかった。

実験３

### Windows環境での使い方

#### 方法１．Virtual Boxを使う

操作：

Virtual BoxでUbuntuマシンを作って，ISOファイルを仮想CDドライブにマウントして起動し直す。

デメリット：

OSの上にOSを乗せているので，当然パフォーマンスが落ちる。

ホストOS（Windows）とのファイル共有，クリップボードやUSBメモリのマウントにひと手間必要。

（補足）スペックの高いパソコンであれば，これが一番手軽かつ便利な方法。

#### 方法２．LiveUSBを利用する

別のPCにUbuntuを入れたり，故障からの復旧に使われるLiveUSBを利用する。

操作

LiveUSB作成

書き込みツールUNetbootin：<https://unetbootin.github.io/>

Space used to preserve files across reboots (Ubuntu only) を0以外にする。※最大4096MB

※このソフトではisoダウンロード後に書き込む事もできるが，isoは先に入手しておいた方が良い。

（補足）Linux機で使う場合

sudo add-apt-repository ppa:gezakovacs/ppa

sudo apt update

sudo apt install unetbootin

設定ファイルの更新

/boot/grub/grub.cfg

menuentry "Try Ubuntu without installing" {

linux /casper/vmlinuz.efi ~ quiet splash --- **persistent**

起動

PC本体を再起動 → F12とかDELとかでBIOS(UEFI) 画面を出す → USBブートできるように設定。

USBブート後もUbuntu(GNU GRUB) で何を起動するか聞かれる ⇒ Try Ubuntu without installing

デメリット：

ファイルシステムが最大4Gbしか対応できないので，アプリ（パッケージ）などが入れられない。⇒ GPartedで自身のパーティションを変更する事はできないのか？

#### 方法３．デュアルブート

初めに：

一番良いのはWindowsとは別ストレージを用意してUbuntuをインストール。

*⇒ 同一ストレージの別パーティションにインストールした場合，Windowsの機能アップデート後にトラブルが発生する。*

ノートPCは内部ストレージが１つである事が多いので， USBメモリにUbuntuをインストールして常にそのUSBメモリを接続しっぱなしにしてするしか方法しかない。但し，この方法では実質USBポートが常に埋まってしまう。

→ 対策：トラブル覚悟で，別パーティションにインストール。もしくはVirtualBox などを用いる。

操作：

LiveUSBでUbuntuを立ち上げる。GPartedなどでスワップ領域と（必須ではない）ext4でルートディレクトリ（/） に設定されたパーティションを用意。その後，デスクトップの右上にある「インストール」

注意！大分後ろの方に出てくる，「*コンピュータにはUbuntu\*\*とWindows Boot Managerがインストールされています*」のところで一番下の**「それ以外」を選択**する事！

その後，インストール先に先ほど作成した /dev/sdb などを選択する。

画面が見切れる問題：Altキーを押しながら画面をドラッグ

ブートローダの変更：

Ubuntuをインストールすると，パソコンを起動した時のブートローダがgrubになる。

このgrubは優先順位がWindows Boot Managerよりも高くなっていて，更にgrub内でもWindowsよりもUbuntuのほうが優先度が高いという二重構造になっている。場合に応じた設定方法を下に残す。

補足：UEFIを使ったシステムの場合あまり問題にならないらしい。ただ、USBへUbuntuをインストールを行う際は念のため仮想マシンなどで行った方が無難に思える。

grub内での優先順位を変更する：

grep -e "^\(menuentry\)\|\(submenu\)" /boot/grub/grub.cfg # grub内のOSの順番を確認

/boot/grub/grub.cfg nano, vim などで編集

GRUB\_DEFAULT=2 # grub内で標準で呼び出すOSを変更する。※上から0番，1番... となる

> sudo update-grub で編集内容を適用する

Ubuntuを使ってブートローダをWindows Boot Managerに戻す：

sudo efibootmgr # ブートローダの優先順位が分かる

sudo efibootmgr -o '000,0003,2001,2002' # 例 --bootorderでも可

但し，Windows Boot Manager ではUbuntuは起動できない（※）のでこの方法ではUbuntuは起動できなくなる 。LiveUSBで起動して，もう一度ブートローダの優先順位を変える。

※厳密にはできるらしいが，めんどくさそうなのでやってない。

Windowsを使って再びgrubに変更する：

管理者権限でコマンドプロンプトを開く（ファイル名を指定して実行 →cmd【Ctrl+Shift + Enter】）

> bcdedit /enum firmware

ファームウェア アプリケーション (101fffff :: 例 ＋ 情報抜粋

--------------------------------

path \EFI\ubuntu\shimx64.efi

description ubuntu

> bcdedit /set {bootmgr} path \EFI\ubuntu\shimx64.efi

～Ubuntuデュアルブート 続き

この情報は要確認：20.07.17

また，ブートローダをインストールするデバイスをUSBにする。初期設定ではパソコン本体のHDDになっており，そのまま行ってしまうとかなり面倒な事になる。参考）[grub対応](#_grub対応：)

（参考）<https://blog.mktia.com/how-to-make-ubuntu-live-usb/>

##### UEFIで対応する方法

起動直後にF12など → UEFI BIOSでSecure Bootを無効に

Win10起動 → パーティション操作

1. Win10用パーティションを縮小：使用領域＋Update用32Gb＋α以上とする。

2. Ubuntuインストール用FAT32の領域を作る。（未フォーマットのパーティションで良い）

3. LiveUSB でUbuntuを起動。デバイス名を確認してインストール。

※パーティション操作にはEaseUS Partition MasterやMiniTool Partition Wizardを使う。

（コンピュータ → 管理 → ディスクの管理　ではパーティションの移動ができない為。

但し，ボリュームの縮小　で対応できるらしいので好みとなる。20.07.21）

### 初期設定

#### 時間のずれの解消

sudo hwclock -D --systohc --localtime

sudo timedatectl set-local-rtc true

sudo timedatectl set-local-rtc 1 --adjust-system-clock

概要

Windows側のシステム時刻(OSが管理している時刻)がLocal Time(ローカル時)、Linux側がUTC(協定世界時)に設定されている。

Windows側で変えるとレジストリの変更が必要なので、Linux側で変えたほうが楽

[ terminal ]

sudo hwclock --verbose --systohc --localtime

方法２

sudo ntpdate ntp.ubuntu.com

過去の方法

$ sudo timedatectl set-local-rtc 1 --adjust-system-clock

#### 日本語入力：

アクティビティ → lang → 言語サポート

## 設定

### 設定に関する情報

#### ホームディレクトリ

特に日本語版のUbuntuを入れるとターミナル起動直後のホームディレクトリに少し戸惑う。

cd ../.. で２階層あがった後にls をすれば通常のファイル構成を確認できる

#### PATH

#terminal

echo $PATH # 現在PATHに設定されているパスを表示

nano .profile # 編集

# 末尾にこれを追加

export PATH=$PATH:/usr/local/pgsql

source ~/.profile # 実行

### Windows側の共有フォルダを利用

#### smbクライアントを利用する

ファイル → サーバーへ接続 の文字列に “smb://192.168.1.1” と言った形で入力する

#### マウントする

sudo apt install cifs-utils

### ファイアウォール

Ubuntuはデフォルトではiptablesが入っていない。

代わりにufw；Uncomplicated FireWall を使用している

設定ファイル：/etc/default/ufw

### X Window System関連

#### 概要

X Window Systemは、ユーザー入力（マウスやキーボード）や出力を受け付ける「Xサーバー」と、Xサーバーの入力情報を受け取って処理し、Xサーバーに対して画面の表示要求を行ったりする「Xクライアント」で成り立っている。

GNOMEやKDEに利用されている。

別名：X11?

#### 関連コマンド

##### startx

startx x windowサーバーを開始する

##### xdpyinfo

情報元：https://qiita.com/kakkie/items/c6ccce13ce0beaefaad1

##### xwininfo

ウィンドウの位置などの情報も得られる

##### xeyes

LinuxのX Window クライアントで有名なソフト。実行すると目玉が表示される。

##### xauth

xauth list # アクセス権限を調べる

##### xorg

sudo apt install xorg

(X.Org)

主流のX Windowsクライアントソフト

###### 設定ファイル

[ /etc/X11/xorg.conf ]

Section "Device"

Identifier "HDMI"

Driver "modsetting"

Option "Monitor-HDMI-0" "HDMI"

Option "Primary" "True"

EndSection

グラフィックリソースの設定？見たことない。

#### 関連する環境変数

XAUTHORITY

X Windowシステムの認証トークンの指定。

X Windowシステムを使用するアプリを正常に実行する為に必要な場合がある

補足

Cannot open display:0 といったエラーが発生する際には設定が必要らしい。

Unitファイルで対応する事も出来る

[ /etc/systemd/system/sample.service ]

[Service]

Environment=XAUTHORITY=$XXAUTHORITY #こんな事で良いらしい

### Wayland

「X11を置き換えるもの」として期待されているGUIシステム

Westonは、Waylandコンポジタのリファレンス。

Waylandにおいては、クライアントがサーバーに描画指示を行い、それを受けたサーバーが描画処理を行う。

この通信プロトコルがWaylandであり、Waylandプロトコルを実装したサーバーをWaylandコンポジタと呼びます。このコンポジタの1種がWestonです。

# アプリ

## 概要

最初に：

「Ubuntuソフトウェア」でだいたいのアプリはインストールできるので、それ以外の方法が必要な場合にのみここに記述する。

### .shでのインストール方法

先にTerminalを開いておいて，Terminalから実行する。

chmod +x *script\_file.sh* # 「実行」のアクセス許可を付与（大体付いているが）

./*script\_file.sh*　 # 実行する

Windowsの感覚で行くと，.shファイルをTerminalで開けば良さそうだが，そういった事はできない。

### .debファイルのインストール

dpkg -i *package*.deb #

apt ./*package*.deb # 実はapt でもインストールできる

### MakeFileでのインストール

にTerminalを開いておいて，Terminalから実行する。

sudo apt install build-essential # ビルドツールが無ければインストール(make コマンドの為)

cd *target\_directory* # MakeFileがあるディレクトリへ移動

sudo make install # インストール

### アンインストール

方法１：設定

 → アプリケーション → “ソフトウェア”で開く → 削除

方法２：コマンド

sudo apt remove google-chrome-stable

但しこの方法だと、アプリの正式名称が必要

方法３：Ubutntu Software

 → インストール済 から対象アプリの「削除」を選択。

※Ubuntu Software を使ってインストールしたアプリしか削除できない。

## 管理

### OSバックアップ

#### 概要

Timeshift スナップショットを利用したシステム復元ツール

Clonezilla ディスクイメージでシステム管理するツール

#### Timeshift

スナップショットを利用したシステム復元ツール

#### Clonezilla

ディスクイメージでシステム管理するツール

### 圧縮・解凍

#### 7zip

用法：7za command archive\_name [file\_names]...

sudo apt -y install p7zip-full

7za --help # ヘルプの表示

7za x target.7z # 解凍（展開）例

コマンド

a file\_name ファイルをアーカイブに追加

x 展開

オプション(switches)

## 開発

### python

少し紛らわしいが、python3が最初から入ってる。

python3

で起動できる。

但し仮想環境(venv) やpipのインストールが必要

apt install -y python3-pip

apt install -y python3-venv

入ってなければ以下の方法でインストール

sudo apt install python3 python3-pip # インストール

sudo apt install python-is-python2 # 近年python2は入っていないことが多い2021-07-08

### Android Studio

<https://developer.android.com/studio/install>

上記サイトにちゃんと説明があるが，念のため。

64bitの場合は先に以下のパッケージをインストール

sudo apt-get install libc6:i386 libncurses5:i386 libstdc++6:i386 lib32z1 libbz2-1.0:i386

### VSCodium

ターミナル：code

100%オープンソース版のVSCode

※また，VSCodeとVisual Studio Codeは別物らしい。ややこしい。

公式ページで.deb ファイルをダウンロードしてインストールする。

sudo apt install ./<file>.deb

（参考）<https://code.visualstudio.com/docs/setup/linux>

### Git

各ディレクトリで働くアプリで，ディレクトリ内に含まれるファイルの変更履歴などを取り扱う。

リポジトリ：各ディレクトリ（プロジェクト）に割り当てられる，管理データファイルの事

ローカルリポジトリ：通常，ローカルクライアントにあるこのリポジトリに更新などを行う。

リモートリポジトリ：サーバー上にあるリポジトリ。ロカールリポジトリが同期される。

#### オンライン操作(GitHub)

##### アカウント作成

まずはGitのオンラインサービスであるGitHubでアカウントを作成する。

##### レポジトリの削除

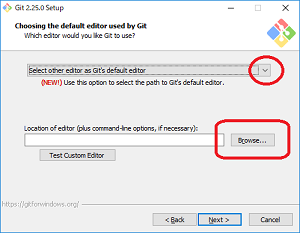
レポジトリのSetting の下の方にDangerZoneというグループがある。

その下の方にDelete this repositoryがある。（右図参照）

#### 最初の準備(Win)

git bash linuxライクなCUI

git cmd 通常のcmdだが，bashへのPATHが通っていて，ハイブリッドなcmd



・インストール時にエディタを聞かれるので，ここだけはしっかりと選択しておく。

（左図参照）

・恐らくBashだけで良いので，UseBashOnly を選択しておく。

（GUIは使うにせよ，サードパーティのもので良いと思う）

##### 最初のチュートリアル

ここではGitBashでの例。GUIなどでも基本的にやる事は同じ。

ローカルとリポジトリの新規作成とリモートとの同期の例：

setx HOME *”C:\data\hogehoge”* # 必要があればhomeディレクトリの変更

git config --global user.name *username* # ユーザー名の設定（先にwebで作っておいたもの）

git config --global user.email *m**ail\_address* # メールアドレス

# エディタの変更（インストール中に設定し忘れたり，別のエディタを使いたい場合のみ）

git config --global core.editor "*'C:\Program Files\Sublime Text 3\subl.exe'* --new-window --wait"

# ローカル環境の設定

mkdir “*directory\_name*” # 必要あればディレクトリを作成する

cd “*directory\_name*” # 対象ディレクトリに移動しておく。

git init # 対象ディレクトリでのGitの開始。（既存ファイルは削除されない）

# 更新を開始（先にリモートレポジトリを作っておく）

git add -A # フォルダ内にあるすべてのファイルを追加（または更新）

git commit -m “*short commit message*” # 変更をcommit

git push -u https://github.com/*user*/*reposit* master # リモートへpush -u：--set-upstreamの事

# あとは更新毎にadd -A 以下を繰り返す。

# 標準のリモートレポジトリを示す変数originを設定しておくと後々楽

git remote -v # 現在のorigin の設定値　空白ならば設定が必要

git remote add origin https://github.com/*user*/*reposit* #

Configに関して：

local 各リポジトリごとの設定 global 現在ユーザーの共通設定 system システム内の設定

pull（ダウンロード）する場合：

git clone https://github.com/*user*/*reposit* # 他人のプロジェクトへの初参加時など，**最初はクローン**する

git clone https://github.com/*user*/*reposit dir\_path* # ディレクトリ名指定バージョン

# 注意：cloneコマンドでリポジット名のディレクトリが自動作成されるので，homeで実行するか，下側のディレクトリ指定バージョンを使う。

# リモートリポジトリで何かしらの変更があった

git pull origin master # 変更の適用。※一度cloneするとorigin という変数が使える

<https://qiita.com/gold-kou/items/7f6a3b46e2781b0dd4a0>

##### branchのチュートリアル

git branch -a # branchの一覧を見る

git checkout -b *branch\_name* # 新branchを作成し，更に切り替える。

# -b オプション＝ git branch branch\_name + checkout branch\_name

git push -u origin *branch\_name* # remote も新ブランチに切り替えつつpushする

git checkout -D *wrong\_name*  # ブランチの削除。名前を打ち間違えた時など。

# 何かしらの変更やテストをここで行う

# 新ブランチbranch\_name に問題が起こり，前の状態に戻りたいとき

git checkout master # masterに切り替える（前の状態に戻す）

# 新ブランチbranch\_nameの変更をmasterに適用するとき(merge)

git checkout master # 一度masterに切り替える

git merge *branch\_name* # branchの変更を適用する。master ← branch\_name　となる。

git branch -d *branch\_name* # 必要なくなったbranchを削除する。（当然，.残しておいても良い）

# リモートのブランチ

git push -d origin branch\_name # リモートのbranchを削除　--delete でも良い

git fetch -p # 削除されたリモートbranchをローカルに適用する　--pruneでも良い

補足）

・よく用いられるbranch名として，issue, develop などがある。

##### ssh公開鍵

# GitBash

# 公開鍵認証

ssh-keygen -t rsa -C "メールアドレス" # ~/.ssh/id\_rsa が作成される

# ssh-keygen -t rsa -b 4096 -C "メールアドレス" -Cと言う情報もある

# Enter file in which… ()内のパスがSSH keyの保存先となる。問題なければそのままEnter

xclip -selection c [target\_path] # 公開鍵の内容をクリップボードにコピー xclip は要インストール

# GitHub で公開鍵を登録

# 1.GitHubにログインし，右上のプロフィールをクリック -> Settingsにて設定画面に遷移

# 2.左のメニューからSSH keysをクリック

# 3. Add SSH Keyをクリック

# 4. Titleに識別できる文字列を入力

# 5. Bodyに先ほどクリップボードに入れた値をペースト

# 6. Add Keyをクリック

# 秘密鍵のパーミションを変更

chmod 600 (秘密鍵のファイル名)

# /.ssh/config ファイルの作成

# 以下内容

Host github.com

HostName github.com

IdentityFile ~/.ssh/id\_rsa # ここで指定するのは秘密鍵の方 .pub の付いていない方のファイル

User git

# ssh を用いて接続できればOK

<https://qiita.com/jun6231jp/items/5596ef0c912c7f57ef0a> Webアプリの情報

##### GitBash操作

ショートカットキー

|  |  |
| --- | --- |
| Ctrl+A 行の先頭へカーソルを移動 | 行の最後へカーソルを移動 |
| Alt+←(Alt+→) 単語単位でカーソル移動 | Atrl+I 画面をクリアする |
| ↑ 一動作前のコマンドを表示 | ↓ 一動作後のコマンドを表示 |
| Tab 入力中のコマンド補完 | Ctrl+R 履歴検索 |

#### Github

##### README.md

![](file\_name) ファイルを選択

#### メモ

他にもSVN(Subversion)やTFS(Team Foundation Server)というツールがある。

##### GUI

Source Tree

git GUI

GitHub Desktop リモートレポジトリにのみアクセスする？

Tortoise Git Windowsのエクスプローラー上で動く拡張機能。←微妙だった20.02.10

##### 自作サーバー

自作サーバーをリモートリポジトリにする事も可能

sudo apt install git # git をインストール（ない場合）

mkdir -p /var/www/git # リポジトリを置くためのディレクトリを作成

#### コマンドリファレンス（抜粋）

add # レポジトリにファイルを追加 ※commitするまでは変更は適用されない

add *filename*

-A # 新規作成/変更/削除されたファイル全てを追加(AllのA)

-u # 変更/削除されたファイルを追加。新規作成は対象外。(updateのu)

-n # 追加されるファイルを調べる

. # 新規作成/変更されたファイルを追加。削除は対象外。

(関連) reset

reset # 変更のリセット

※commitするまでは変更は適用されない

--soft # 作業ツリーがそのまま

--hard # 作業ツリーも削除

(関連) commit

commit # 変更点のcommit

※オプション無しだと，メッセージ入力の為にエディタが立ち上がる。ハッシュ記号(#)でコメントアウトされた最後の行の次から書き込む。

１行目に要約。２行目は空白。３行目に詳細　などとルールを決めておく。

コメントを空白にすると，コミットを取り消せる。また，エディタで「保存」を行わずに閉じると同様にコミットの取り消しになる。

-m *”message”* # コミットメッセージと呼ばれる，要約をつける

-a # 変更のあったファイルすべて

--amend # 直前のcommitを修正。commit時とほぼ同様の動作となる。

(関連) reset

push # ローカルの変更をリモートへ適用

push origin mastar # ブランチを指定してpush（ブランチ名はmasterがデフォルト）

# ※origin はcloneやpull を使った後に使える変数

# : *branch\_name* とするとリモートブランチを削除できる

-u # upstream（メインのブランチ）としてpushする --set-upstream

-d # --delete

git push --delete origin *”branch\_name”* # リモートブランチの削除

branch # branch操作

branch *branch\_name* # branchを作成

-a # 全てのbranchを表示する --all

-d *branch\_name* # マージ済のブランチの削除（マージ済でないと消せない） --delete

-D *branch\_name* # 強制的なブランチの削除

--remote # リモートブランチの一覧を表示

checkout # 作業branchを切り替える。

git checkout *branch\_name*  # branch\_name に切り替え。基本的な使い方。

-b *branch\_name* # 新branchを作成し，更に切り替える。

-b *new\_branch base\_branch* # base\_branch を起点にbranchを作成

-B *branch\_name* # branch\_nameが存在すれば移動，存在しなければ作成し，更に切り替える。

-f *branch\_name* # エラーを無視して強制的に切り替え

-m # --merge

checkout *”file\_path* # 指定したcommitの状態を現在の作業ツリーに展開する

#

merge # 現在のブランチの変更を指定したブランチに適用する

git merge *branch\_name* # 新bran

-b *branch\_name* # 新branchを作成し，更に切り替える。

--no-ff

clone # 他ユーザのリポジトリの完全コピー

e.g.1 git clone https://github.com/libgit2/libgit2 mylibgit # ディレクトリを新規に作成する場合

e.g.2 git clone user@server:/path/to/repo.git # ユーザー名とサーバー指定

rm # ファイル／ディレクトリを管理対象から外す

rm *”file\_name”* #

-d *”directory\_path”* # ディレクトリを削除。空でないと消せない。

-r *”directory\_path”* # ディレクトリごと削除。

--cached *”file\_name”* # ファイルを残す時

（関連）.gitignore

fetch # 変更情報を取得

e.g. git fetch # origin/master が最新になる

diff # 変更差分を表示　引数が必要？

init # 指定ディレクトリにリポジトリを作成

#

log　ログを表示

pull # fetch とmerge を実行するショートカット。実行前にローカルレポジトリがcommitされている必要がある。

status # commitに関する状態の確認

remote # 今までに使用したリモートサーバーを表示する

add *shrortname url* # リポートリポジトリの追加

add origin ssh;//git@github.com:user/reposit # よくある例。originの値を変更する。

-v # 書き込み用と読み取り用のアドレス(url) を表示

※commitするまでは変更は適用されない

#### 用語（Git）

commit 変更をリポジトリに反映する事。

push ローカルの変更内容をリモートに反映する事。

pull リモートの変更内容をローカルに反映する事。

fetch リモートリポジトリの内容を確認する

Bareリポジトリ 変更対象となるファイルを持たない管理用のリポジトリ。＝リモートリポジトリ

origin リポジトリの場所(URL)を格納した変数のようなもの

branch 本作業に影響を与えない為の分岐。変更に問題がなければ本流のbranchとmargeできる。

master initで作成されるbranch名で，

head 現在の作業branchの先頭commit

#### 資料

オンラインマニュアル<https://git-scm.com/book/ja/v2>

### VirtualBox

※Ubuntu上でVirtualBoxを使う為の情報

公式サイト<https://www.virtualbox.org/> からパッケージファイル（.deb）をダウンロード

sudo dpkg -i virtualbox-<...>\_amd64.deb

## 常用

### vlc media player

sudo apt -y install vlc

Setting→詳細→Default Applications→Video→

### anki

公式ページから .tar.bz2でダウンロードされるので，tar で解凍する

tar xjfv anki-2.1.28-linux-amd64sudo .tar.bz2 --directory=./ # 解凍

cd anki-2.1.28-linux-amd64

~~sudo make install # makeファイルのインストール ない場合はbuild-essentialを入れる~~

#↓

sudo ./install.sh # 2021.10.01現在

sudo apt -y install mpv #音声や動画を再生する為にmpvをインストールする必要がある。

### Thunderbird

インストール

以下の３つの方法がある。

普通にaptでインストール。

sudo apt install thunderbird



ソフトウェアセンターを利用

公式サイトから .tar.bz2ファイルをダウンロード　<https://support.mozilla.org/ja/products/thunderbird>

tar xjf thunderbird-\*.tar.bz2

※但し、この方法はいわゆるポータブル版になる。メールが重複したりおかしかった事がある

## Office系

### LibreOffice

MS Officeを使う方法

Microsoft365を使う

### mono

[https://www.mono-project.com/download/stable/#download-lin](https://www.mono-project.com/download/stable/" \l "download-lin)

### rclone （各種オンラインストレージ）

#### 概要

対応：Amazon Drive，Amazon S3，Box ，Google Drive， Dropbox，Google Cloud Storage， Microsoft，Nextcloud，One Driveなど

#### チュートリアル

OneDriveの設定

sudo apt install rclone # 普通にaptでインストールできる。

sudo dpkg -i rclone-v\*.\*\*.\*-linux-amd64.deb # 公式版をDLしてインストールする場合

OneDriveの設定

rclone config

n/s/p> n # 新規作成

name> *my**\_strage* # 自由に名前を付けて良い

Storage> *21*  # サービスのtypeを入力。例：2021-06-27時点では oneDrive は21

client\_id> # 入力する必要がない → なにも入力せずにEnter

client\_secret> # 入力する必要がない → なにも入力せずにEnter

Edit advanced config? (y/n)> # n 任意だがyでも良い。

Use auto config? # y

# ここで自動起動したWebブラウザでMicrosoft OneDriveにログイン。「Success!」と表示されたら、端末に戻る。

Choose a number from below, or type in an existing value

Your choice> *1*  # 大体の場合，Personal

Chose drive to use: # 今作成したドライブの番号を入力

# マウント

mkdir ~/*onedrive* # マウント用のディレクトリ作成

rclone --vfs-cache-mode writes mount *my\_strage*: ~/*onedrive*/ & # マウント

# 補足：最後の& はバックグラウンドで動かす事を指す。表示される数字はプロセスのID番号（PID）

ps -f -p 1234 # PID=1234 の時、こうするとプロセスの状態がみられる

2023-03-11追記：[https://rclone.org/#providers](https://rclone.org/" \l "providers) からconfig を見たほうが早いかも

Linuxの起動で自動的にマウントしたい場合はcrontab -e で以下のように記述

（以下のどちらでも良い）

@reboot /usr/bin/rclone mount *my\_box*: ~/*box*/

@reboot /usr/bin/rclone --vfs-cache-mode writes mount *my\_onedrive*: ~/*onedrive*/

補足

/etc/rc.local で実行するとユーザーがrootな為、~root/.config/rclone/rclone.conf を読みに行ってしまいうまく行かない。ユーザー権限で実行するcronの方が適切。

また、cronはもともとバックグラウンドで動作するため、スクリプト最後の「&」は不要。

#### Windowsでの使用

##### インストール（chocolatey

Chocolatyをインストール（管理者権限でpowershellを実行）

Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))

rcloneを含めた関連パッケージのインストール

choco install rclone winsfp sshfs

任意

choco install rclonetray

補足：あるアカウントにすでにインストール済だと、インストールに--forceオプションが必要

choco install rclone winsfp sshfs --force

rclone.confのありか

|  |  |
| --- | --- |
| 標準 | %USERPROFILE%\.config\rclone |
| 全体 | C:\ProgramData\rclone\rclone.conf |
| それ以外１ | %USERPROFILE%\AppData\Roaming\rclone |
| それ以外１ |  |
| それ以外１ |  |

<https://qiita.com/fuk101/items/f5245b741e236cc9c13f>

#### 詳細

rclone概要

rclone [フラグ] [コマンド]

フラグ （公式資料）<https://rclone.org/flags/>

--vfs-cache-mode 詳細不明だがOneDriveには推奨らしい

writes

full

コマンド rclone -h でコマンドのヘルプが表示される。

mount

rclone mount name:[path/to/remote] /path/to/local

copy

rclone copy /path/to/local name:path/to/remote

rclone mount remotebox: ~/rclone/remotebox/ & # 例：

listremotes リモートの設定ファイルを全て表示

config

オンラインストレージの設定を行う。

なお，設定は/home/user1/.config/rclone/rclone.conf に保存される

オプション

--daemon デーモン（バックグラウンドモード）として起動。Windowsでは使えない。

#### 補足

#### 情報

・rclone-browser という，完全GUIのパッケージも存在するらしい。

・rclonetrayというツールもある

（参考）<https://qiita.com/afj/items/61ca815aa37d87bd391d>

（公式サイト）<https://rclone.org/>

FUSE；Filesystem in Userspace

## セキュリティ

アンチウイルスソフトウェア

補足：

公式が言うには、そもそもUbuntuは攻撃対象として選ばれる事が稀、OSのセキュリティが高い、点からアンチウィルスソフトウェアはそれほど必要とされていない（らしい）

### ClamTk

### ClamAV

sudo apt install -y clamav

sudo clamscan --infected --recursive --remove / # ウィルススキャン

Comodo Antivirus for Linux

Windowsも対応

：Sophos Antivirus

F-Secure Linux Security

## 画像系

### Pinta

お絵かきソフト

#sudo add-apt-repository ppa:pinta-maintainers/pinta-stable #Pintaのレポジトリ取得（多分不要）

sudo apt install pinta

### jp2a

jpegからASCIIアートに変換する

Github：<https://github.com/cslarsen/jp2a>

## 動画編集

### kdenlive

非常に高機能なアプリ

但し、使い方が分かりにくかった

### OpenShot

公式：https://www.openshot.org/ja/

### Simple Screen Recorder

PC画面の録画ができる。

sudo apt install simplescreenrecorder

補足

OS標準の「スクリーンレコーダー」があるらしい。起動方法：Alt+Ctrl+Shift+R

## ネットワーク関連

### Samba

概要

smbd ファイル共有であるSMB/CIFS サービスをクライアントへ提供

nmbd NetBIOS over TCP/IP クライアントへの NetBIOS ネームサーバー機能の提供

#### コマンド

pdbedit # SAM データベース (Samba ユーザーのデータベース) を管理する

-a # ユーザーをデータベースに追加する。

(e.g.) pdbedit -a -u source

#### 設定ファイル

# smb.conf

[global]

dos charset = CP932 # 効果不明

unix charset = UTF-8 # 効果不明

[share] # 共有名

comment = Share #

path = /home/samba/share #

read only = no #

browsable = yes # ブラウズリスト中に現れる利用可能な共有の一覧

force user = pi

public = yes #

guest ok = yes # ゲストユーザー(パスワードなし)での接続を許可。

create mode = 0777 #

directory mode = 0777 #

writeable = yes #

<https://www.samba.org/samba/docs/current/man-html/smb.conf.5.html>

FAT

### gFTP

aptやyumでインストールする。

sudo apt install gftp

鍵認証をしたい場合

メニュー：FTP→Preferences（設定）→SSH

SSH引数を以下のようにする

ssh -i /home/user\_name/.ssh/id\_rsa

補足：Ubuntuのファイルブラウザでは普通にリモートのファイルが開ける。

→  で sftp://192.168.1.1 などと入力。

鍵認証の場合は~/.ssh/config の設定名を使う　例：sftp://my\_config

Nautilusというのもあるとか

### ufw

ファイアウォール。やや扱いやすい。（らしい）

ログ /var/log/ufw.log

### VNC

vinagre 、 krdc 、 RealVNC 、 TightVNC などが利用可能。

VNCには大きくUltraVNCとRealVNCがある。

TightVNC 動作、容量が軽い。但し画面の表示だけらしい。

# サーバー側

sudo apt install tigervnc-common

sudo apt install tigervnc-standalone-server

sudo apt install tigervnc-scraping-server

tigervncserver -localhost no # VNCサーバーの起動

# 閲覧（クライアント）側

sudo apt install tigervnc-viewer

2022-06-23うまく行かなかったので保留

VNC Viewer

公式サイトから.debファイルをダウンロードする形式。

（ダウンロード先）<https://www.realvnc.com/en/connect/download/viewer/>

### Remmina

VNC, RDPでの接続が可能。

RDP（Remote Desktop Protocol）：Windowsに使用されている。

### iptables

パケットフィルタリング型のファイアウォール。

## SNS系

### zoom

.deb ファイルをダウンロードする形式。（2021-07-09現在）

（ダウンロード先）[https://zoom.us/download#client\_4meeting](https://zoom.us/download" \l "client_4meeting)

apt install ./zoom\_amd64.deb

sudo dpkg -i zoom\_amd64.deb

### discord

## Html系

### chromium

オプション

--app=URL ツールバー無しで表示

--window-size =width, height

--window-position =0,0

## データベース

### Postgresql

クライアント／サーバーモデルで、サーバー側のプログラムはpostgres

## ストレージ

## 仮想化など

### wine

#### 概要

Windowsの仮想環境を構築してWindows系のアプリを実行する

公式インストール方法

<https://wiki.winehq.org/Ubuntu>による公式インストール方法

sudo dpkg --add-architecture i386

sudo mkdir -pm755 /etc/apt/keyrings

sudo wget -O /etc/apt/keyrings/winehq-archive.key https://dl.winehq.org/wine-builds/winehq.key

sudo wget -NP /etc/apt/sources.list.d/ https://dl.winehq.org/wine-builds/ubuntu/dists/focal/winehq-focal.sources

sudo apt install --install-recommends winehq-stable

推奨：

（新しいバージョン）<https://wiki.winehq.org/Ubuntu>

（参考）<https://qiita.com/yakitatata/items/12a6e1fba33cdaa814d9>

新しいwinw(6.18)は初回起動時に.NET(mono)をダウンロードするか聞いてくれる。そこでダウンロードすれば，その先は.NETアプリも普通に動く。

テスト

wine notepad # メモ帳が起動

wine winemine # mineスリーパーが起動

補足：ここでメモ帳が開くのは、仮想windowsにメモ帳がインストールされているから

メモ

・ホストOSの\home\sdaがC:¥users¥sdaとして仮想的に動く。

# インターネットエキスプローラー

winetricks wine-gecko # これが必要

wine iexplore

#### winetricks

概要

必要な追加ライブラリを管理する

インストール

sudo apt install winetricks

winetricks --

文字化け対応をする

ファイルブラウザを開く

Select the default wineprefic → Run explorer

Webブラウザを開く

#### Windowsアプリのインストール

mkdir ~/wine/SomeWinApp # 書き込み権限があるディレクトリが必要

wine ./SomeWinAppInstaller.exe # wineからWinアプリのインストーラーを起動するとインストール

wine regedit # レジストリエディタの起動

wine uninstaller # アンインストールする時

.NET Framework

sudo dpkg --add-architecture i386 # 32bitを有効にする

Officeのインストール

Firefoxのインストール

#### 別パーティションのアプリを実行する例

補足　勘違いから発生した無駄な調査情報だった 2023-0204

前提条件

・ドライブまたはパーティションが２つ以上あり、windows、ubuntuのブートローダーそれぞれ入っている

・そのwindowsパーティションは、物理的に接続されている

・そのwindowsパーティションにOfficeがインストールされている

事前準備：

# 必要ライブラリのインストール　不要かも？

sudo apt install winbind

# winetricksの必要ライブラリのインストール　この内いくつかは不要かも？　注意:2時間くらいかかる

winetricks fontsmooth-rgb gdiplus mfc42 msxml3 riched20 vcrun6

実行ファイルをさがして実行：

# 対象パーティションのUUIDを調べる

# 補足：egrepが入っていない場合は grep 'vfat' などとしても良い

lsblk --output NAME,PARTLABEL,UUID,FSTYPE,SIZE | egrep 'vfat|ntfs'

# > SIZEやPARTLABEL="Basic data partition" などで判断する

# /mntに仮マウントする

sudo mount -t ntfs UUID={UUID} /mnt

# あとは普通にlsコマンドでofficeのインストール先のパス構成を調べる

# ・Windowsシステムのパス名には空白が含まれる為''で囲う必要がある

ls -l '/mnt/Program Files/Microsoft Office/root/Office16' | grep '.EXE'

# WORDの実行

wine '/mnt/Program Files/Microsoft Office/root/Office16/WINWORD.EXE'

資料

参考：http://linux.ikoinoba.net/index.php?mode=page&aim=wine

参考：[https://www.yukkuriikouze.com/2019/05/25/2940/#toc\_id\_3\_1](https://www.yukkuriikouze.com/2019/05/25/2940/" \l "toc_id_3_1)

公式wiki　https://wiki.winehq.org/Download

### Docker

別ドキュメントソフトウェア情報.docxに移動。

#### 補足

但しラズパイの場合Arm，IntelのCPU違いでDocker公式イメージがほとんど使えない為，あまりメリットはないとか？

これら，本当に必要か？

sudo apt install apt-transport-https ca-certificates curl gnupg-agent software-properties-common

## トラブル対応

### aptがロックしてしまった場合

sudo rm /var/lib/apt/lists/lock

sudo rm /var/cache/apt/archives/lock

### Release file for xxx is not valid yet

Dockerや仮想マシンとホストOSとで時間のずれがある場合に発生する。

大概、再起動などで時間のずれは解消する。

# 基本

## ショートカットキー

### ファイルブラウザ

F9 ナビゲーションウィンドウの表示／非表示

Alt+F2→R GNOMEの再起動

## Ubuntu Pro

### esm-apps

Extended Security Maintenance for Applications の略。

Ubuntu Proのサブスクリプションに含まれる特別なセキュリティメンテナンスサービス。

値段：

<https://ubuntu.com/pricing/devices>

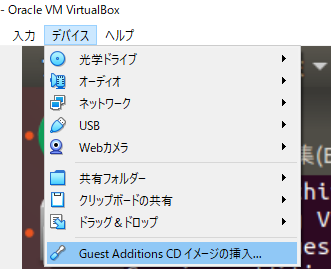
esm-appsは、Ubuntu Proユーザーに特定のアプリケーションに関連するセキュリティアップデートを提供することを目的としています。これにより、ユーザーは最新のセキュリティ修正を受け取り、セキュリティ上の問題から保護される。

# VirtualBox

概要：

VirtualBox で使用する際の注意点を残す。

## 画面サイズ変更

デバイス → Guest AddtionansCDイメージの挿入

→ ターミナルに”Press Return to close this window…” と表示されたら終了

→ VirtualOS(Ubuntu) の再起動

→ 右上から設定 → デバイス → ディスプレイ

→任意倍率のスケーリング

## ホストOSとのファイル共有

VirtualBoxの基本機能で共有フォルダが作成できるが，Ubuntuでアクセス権をうるさく言われるので，ホストOSでネットワーク共有フォルダを作成した方が早い。

# 具体例

## DVD

再生

VLCメディアプレーヤー

# DVDの再生に必要なパッケージ

sudo apt install libdvd-pkg

sudo dpkg-reconfigure libdvd-pkg

.isoファイルを右クリックして

DVDのデバイスは、だいたい、以下のいずれか

/dev/dvd

/dev/sr0

dd if=[デバイス] of=sample.iso

メモ

・BraseroというGUIソフトを使う手もあるが、ddの結果とあまり変わらないっぽい

・mkisofsというCLIコマンドもあるっぽい

Handbrake

flatpakのインストール

apt -y install flatpak

# ひつようなパッケージ

sudo apt install gnome-software-plugin-flatpak

# リポジトリ情報の追加

flatpak remote-add --if-not-exists flathub https://dl.flathub.org/repo/flathub.flatpakrepo

Handbrakeのインストールと起動

# インストール

flatpak install flathub fr.handbrake.ghb

# 起動

flatpak run fr.handbrake.ghb # 普通にGUI上からも起動ができる

補足

別々のディスクイメージのキューも作成できる

## CD

Asunder

sudo apt install asunder #

sudo apt install lame # mp3圧縮に必要

# Debian

2021-09-30に使ってみる機会があったのでメモを残す。

入っていないパッケージ

git

デフォルトではPCのUSBなど，デバイスにアクセスできない模様。(WSL2)

# その他

## メモ

ユーザー識別子

|  |  |  |  |
| --- | --- | --- | --- |
|  | 意味 | idコマンド | 備考 |
| effective UID  （実効UID） | アクセス権限上のユーザー | デフォルト | ファイル作成とファイルアクセスに影響 |
| real UID  （実UID） | プロセスやファイルの実際の所有者 | -r | シグナル送信の権限に影響 |
| saved UID  （保存UID） | 特権を一時的に高める際に使用 |  |  |
| file system UID | file systemに対するアクセス権を得たいが、実UIDなどを変更したくない場合に使用 |  | 変更していない限り実効UIDと同じ |

## ディスプレイマネージャ

### GDM

### LightDM