Linuxの基礎知識

@nc30mtd

コンソールの開き方

端末を開く

Raspberry Pi上で端末を開く.
SSHでリモートホストに接続する.
GUIからの場合は画面左上のアイコンから端末を起動する

※このアイコン→ 

コマンドを実行する

SSHでリモートホストにログイン，またはGUIから端末を起動した画面（シェル）上で行う．
起動時は下記の通りコマンドラインが表示されており，コマンドを打ち込むことにより所定の動作や処理を行う．

```
pi@raspberrypi:~ $
```

`pi@raspberrypi:~` は `ユーザー名@コンピュータ名:現在のディレクトリ` を意味する．

コマンドは `コマンド本体` と `オプション`，`引数` で構成され，半角スペースで区切って実行する．

ex.) `/home` ディレクトリの一覧を表示する際にリスト形式で表示する

```
$ ls -l /home
```

上記の場合，`ls` がコマンド本体，`-l` がオプション，`/home` が引数となる

Linuxのユーザー管理について

Linuxは一般的なオペレーティングシステムと同じで、実際に使用するにはユーザとしてログインする必要がある．ここではユーザについて解説する．

一般ユーザと管理者（root）

管理者（root）

`管理者` は `root` と呼ばれる特別なユーザーで，システムの全ての操作が行えるあらゆる管理権限を有しており，「スーパーユーザー」とも呼ばれる．

一般ユーザー

限られた操作のみ実行できるユーザーで，システムに影響を与える可能性のあるコマンドの実行やセキュリティ上で重要なファイルの閲覧が制限されるなどの制約がある

root以外のユーザーはすべてこの一般ユーザーであり，前述の制約に関する作業を行う場合は`sudo`コマンドや`su`コマンドにより一時的に管理者権限を取得して作業を行う

Raspberry pi（Debian，またはUbuntu）は一般ユーザとしてログインしたのち，管理者権限が必要な作業に関しては`sudo`コマンドを用いて行い，rootユーザーとしてログインしないのが一般的である.

どうしてもrootとしてログインしたい場合は一旦一般ユーザとしてログインしたのち`$ sudo su`などによりユーザーを切り替え可能である.

システムアカウント

特定のアプリケーション（apache,sambaなど）を実行する際に利用する特殊なユーザーアカウントが存在するが，ログインアカウントとしては利用できない.

Linuxのディレクトリ構成

一般的なLinuxにおけるディレクトリ構成について下記に示す.
一般ユーザーとして利用するのは基本的に `/home/ユーザー名` のディレクトリとなる

ディレクトリ名	用途
/	ルートディレクトリ
/bin	基本コマンド(cat,cp,ls等)
/boot	起動に必要なファイル（カーネル、初期RAM）
/dev	デバイスファイル（usb,cdrom,disk等）
/etc	設定ファイル
/home	ユーザのホームディレクトリ
/lib	32bit 版のライブラリー（基本コマンドの実行に必要なライブラリ群）
lib64	64bit 版のライブラリー（64bit版はこっち）

ディレクトリ名	用途
/lost+found	破損ファイルの断片が格納される
/media	リムーバブルメディア用マウントポイント（cdrom等のマウントポイント）
/mnt	ハードディスク等の一時的なマウントポイント
/opt	アプリケーションソフトウェアパッケージのインストール先
/proc	カーネルやプロセス情報
/root	root用ホームディレクトリ
/run	実行時の可変データ群。再起動時に消去される
/sbin	システム管理用コマンドなど（ip, shutdown, reboot など）
/srv	HTTP、FTP 用データが置かれている。
/sys	デバイスやドライバーの設定ファイルなど
/tmp	一時的なファイル（再起動時に消去される）
/usr	ユーザーが使用する各種プログラムなど
/var	変更されるデータ（内容が常に変化するファイル群が格納）

絶対パスと相対パス

ファイルなどの所在を書き表すパス（path）の表記法のうち，重要な２つの記述方法について以下に示す.

絶対パス

階層構造の頂点からの位置関係を記述する方式．ルートディレクトリ（`/`）からの位置関係をすべて記述する方法のことで，途中にあるディレクトリを区切り文字`/`で区切った上，すべてつないで並べる.

相対パス

現在位置からの相対的な位置関係を記述する方式。起点となる位置から目的の位置までの道筋にある要素を順に並べて記述する。

カレントディレクトリは `.` で表し、一階層上位のディレクトリは `..` で表現する．`..`を繰り返し記述することでディレクトリ階層の親子関係をたどって上へ移動することができる

ディレクトリ操作

- `pwd` 現在のディレクトリを表示する
- `ls` ディレクトリの一覧を表示する
- `cd` ディレクトリを移動する
- `mkdir` ディレクトリを作成する
- `mv` ディレクトリやファイルを移動する
- `cp` ディレクトリやファイルをコピーする
- `rm` ディレクトリやファイルを削除する
- `shutdown/halt` システムをシャットダウンする
- `apt/apt-get` パッケージをインストールする
- `pip3` pipでモジュールをインストールする
- `python3` pythonを実行する

pwd 現在のディレクトリを表示する

下記コマンドにより現在のディレクトリを絶対パスで表示する.

```
$ pwd
```

実行例（pwd)

```
$ pwd
/home/pi
```

`ls` ディレクトリの一覧を表示する

```
$ ls
```

オプション

利用頻度のオプションを示す

- `-l` リスト表示する
- `-a` すべてのファイルを表示する（ドットからはじまる隠しファイルを含めて表示する）
- `-t` ファイルの更新日が新しい順に表示する

実行例

```
$ ls -la .
```

※引数がない場合，カレントディレクトリの内容を表示する

cd ディレクトリを移動する

下記コマンドにより現在のディレクトリ（カレントディレクトリ）から移動することができる

```
$ cd [移動するディレクトリ]
```

引数には絶対パス，相対パスいずれも指定可能

※引数がない場合，ホームディレクトリ（`/home/ユーザー名`）に移動する

mkdir ディレクトリを作成する

新規ディレクトリを作成する

```
$ mkdir [移動するディレクトリの名前]
```

mv ディレクトリやファイルを移動する・リネームする

ディレクトリやファイルを移動する.
移動先に同名のファイルやディレクトリがある場合は問答無用で上書きする

カレントディレクトリの `source.py` を `source1.py` にリネームする

```
$ mv source.py source1.py
```

カレントディレクトリの `source.py` を１階層上位のディレクトリに `source1.py` にリネームして移動する

```
$ mv source.py ../source1.py
```

cp ディレクトリやファイルをコピーする

ファイルやディレクトリをコピーする

ディレクトリをまるごとコピーする場合は `-r` オプションをつける

ファイルをコピーする

１階層上層のディレクトリにsource.pyをコピーする

```
cp source.py ../
```

ディレクトリをコピーする

１階層上層のディレクトリにsrcをコピーする

```
cp -r src ../
```

`rm` ディレクトリやファイルを削除する

ディレクトリを削除する場合にディレクトリ名を指定した場合，中身のそれぞれのファイルについて削除に関する確認メッセージが表示される.

煩わしい場合は `-r` オプションの付与により再帰的に削除を行うことができる.
消すことができるファイルは問答無用で削除するため実行時には最新の注意を払う.
（特にrootでの実行時など）

カレントディレクトリのhoge.pyを削除する場合

```
$ rm hoge.py
```

カレントディレクトリに存在するディレクトリ `work` を削除する場合

```
$ rm -rf work
```

`shutdown/halt` システムをシャットダウンする

電源を落とす（落とすことができる状態にする）ために実行する.

※当該コマンドを実行せずに電源断した場合，本来ストレージに書き込まなければならないデータや書き込み中のデータが破損することでシステムに予期しない問題を引き起こす場合がある.

`apt/apt-get` パッケージをインストールする

Linuxにはディストリビュションにより様々なパッケージ管理ソフトウェアが存在する.
パッケージ管理ソフトウェアではパッケージのインストールやインストール時の依存関係の解決を行うことができる.

Raspberry Pi OSはDebianと呼ばれるディストリビュションをベースに構築されており, `apt` または `apt-get` コマンドによりパッケージ管理ソフトウェアを利用することができる.

パッケージのインストールにはroot権限を要するので, 管理者権限で以降のコマンドを実行する `sudo` コマンドを併用して実行する

```
$ sudo apt-get install [パッケージ名]
```

インストールの実行に関する確認メッセージを省略する場合は `-y` オプションを付与する.

```
$ sudo apt-get install -y [パッケージ名]
```

`python3` pythonを実行する

pythonを実行する際にはソースを引数としてソースコードを実行する方法と，インタラクティブモード（対話モード）が存在する．

対話モードは `python3` コマンドのみを実行すると，pythonのプロンプトが表示される．
このプロンプトにソースコードを打ち込むことで実行が可能である．

```
$ python3
Python 3.7.3 (default, Apr  3 2019, 05:39:12)
[GCC 8.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print("hoge")
hoge
>>>
```

ソースコードを指定して実行する場合は `python3` コマンドのあとに引数としてソースコードのパスを与える．実行するコードで更に引数を必要とする場合は以降に記述する．

```
$ python3 source.py arg1 arg2
```

pip3 pipでモジュールをインストールする

pythonでは世界中の有志が構築したモジュールを利用することができる.
これらの中にはpipと呼ばれるパッケージ管理システムを用いてインストールできるものも存在し,
下記のコマンドでインストールが可能である.

```
$ pip3 install [パッケージ名]

または

$ python3 -m pip install [パッケージ名]
```