3章 GNUとUnixのコマンド

3-1. コマンドラインの動作

3-1-1. シェルとは

- シェルは入力されたコマンドを受け付けて、カーネルに対して命令を行う
- デフォルトはbash
- ユーザにログインシェルが設定されている

```
echo $SHELL
```

3-1-2. ディレクトリの指定方法とカレントディレクトリ

- ホームディレクトリ
  - root : /root/
  - 一般ユーザ : /home/ユーザ名/
- カレントディレクトリ
  - pwd
- 相対パス、絶対パス
- 「.」　「..」　「~」　「/」

3-1-3. 変数

- システムに用意されている定義済みの変数（例）
  - LANG : 利用している言語、文字コード
  - PATH : コマンドの実行ファイルの格納先
  - PS1 : プロンプトに表示する内容
  - SHELL : 現在利用しているシェル

```
echo [-n] 文字列 #-nで改行せずに文字列を出力する
```

- 変数の定義「変数名=値」

```
#特定のコマンド実行時のみ変数を変える場合
[root@localhost work]# LANG=C ls 1234
ls: cannot access 1234: No such file or directory
[root@localhost work]# ls 1234
ls: 1234 にアクセスできません: そのようなファイルやディレクトリはありません
```

- PATH変数に新しいディレクトリを追加する

```
PATH=$PATH:追加するディレクトリ
```

- 変数の種類
  - シェル変数：子プロセスには引き継がれない
  - 環境変数：子プロセスに引き継がれる（exportコマンド）

```
export 変数名=値   #指定したシェル変数を環境変数としてエクスポート

set [option] #定義済みの変数と値、関数などを一覧表示。

env #定義済みの環境変数と値を一覧表示。

unset 変数名 #定義済みの変数を削除する
```

3-1-4. 特殊文字の抑止

- 「'」（シングルコーテーション）：囲まれた範囲を抑止
- 「"」（ダブルコーテーション）：囲まれた範囲を抑止
- 「\」（バックスラッシュ：）：次の文字を抑止

3-1-5. コマンドの連続実行

- command1; command2 -> command1に続いて、command2を常に実行
- command1 && command2 -> command1が成功したら、command2を実行
- command1 || command2 -> command1が失敗したら、command2を実行

3-1-6. コマンド履歴

```
history [option] [履歴数]
-c : 履歴の消去
```

histryコマンドで呼び出されるのは、シャットダウン前なども。
ホームディレクトリにある「.bash_history」ファイルに履歴が残る
「$HISTFILESIZE」「$HISTSIZE」

3-1-7. bashシェルのキー操作

- Ctrl + c　：　処理の停止
- Ctrl + z　：　処理の一時停止
- Ctrl + a　：　カーソルを行頭に移動
- Ctrl + e　：　カーソルを行末に移動
- Ctrl + s　：　画面への出力を停止
- Ctrl + q　：　画面への出力を再開
- Ctrl + r　：　コマンド履歴の後方検索

- Ctrl + l　:　画面のクリア

3-1-8. マニュアルの活用

```
man [option] [セクション] キーワード
```

- セクション
  - 1 : 一般コマンド
  - 2 : システムコール
  - 3 : ライブラリ関数
  - 4 : ファイルフォーマット
  - 8 : 管理コマンド

```
man passwd #passwdコマンド
man 5 passwd #/etc/passwdファイル

    -a : 用意されたすべてのマニュアルを表示
    -f(whatis) : 指定したキーワードでマニュアルを検索(完全一致)
    -k(apropos) : 指定したキーワードでマニュアルを検索(部分一致)
```

```
whatis passwd #whatisデータベースを利用して検索
#whatisデータベースがない場合は
makewhatis
```

3-2. テキストストリームの処理

- cat
  - 指定したファイルの内容を表示

```
cat [option] [ファイル]
    -n : 行数をつけて表示（空行にも採番）
```

- nl
  - 指定したファイルの内容を行番号をつけて表示（cat -n）（空行には採番しない）

```
nl [option] [ファイル名]
```

- head
  - 指定したファイルの先頭部分を表示

```
head [option] ファイル
    -行数 : 指定した行数を表示（規定は10行）
```

- tail
    - 指定したファイルの末尾部分を表示

```
tail [option] ファイル
    -行数 : 指定した行数を表示（規定は10行）
    -f : 末尾をリアルタイムで表示
```

- fmt
    - 指定したファイルを段落に整形して表示

```
fmt [option] ファイル
    -w 文字数 : 指定した文字数の段落として整形（規定は75文字）
```

- pr
    - 指定したファイルを印刷用整形

```
pr [option] ファイル
```

- cut
    - ファイル内の各行から指定した列のみを表示

```
cut [option] ファイル
    -d"記号" : 指定した記号を区切り文字として扱う
    -f 列1[,列2***] : 指定した列を表示

$ cut -d":" -f1,6 /etc/passwd
root:/root
bin:/bin
```

- paste
    - ファイルを行単位で結合し手表示

```
paste [option] ファイル1 ファイル2
    -d"記号" : 指定した記号を区切り文字として扱う

[root@localhost com]# paste 1 2
1:a:s   1:n:m
```

```
2:d:f   2:v:b
3:t:y   3:g:v
```

- join
    - 指定した2つのファイルの中から共通の列に基づき結合

```
join [option] ファイル1 ファイル2
    -t"記号" : 指定した記号を区切り文字として扱う(規定はスペース、タブ)
    -1 列 : ファイル1の指定した列を共通の列として扱う(規定は1列目)

[root@localhost com]# join -t":" 1 2
1:a:s:n:m
2:d:f:v:b
3:t:y:g:v
```

- sort
    - ファイル内容を並べて表示

```
sort [option] ファイル
    -t"記号" : 指定した記号を区切り文字として扱う(規定はスペース、タブ)
    -k 列1 [,列2***] : 指定した列を基準に並べ替える
    -r : 降順で並べ替える
    -n : 基準列を数値として扱う
```

- uniq
    - 重複している行を排除して表示

```
uniq [option] ファイル
    -c : 重複している数を表示
```

- wc
    - ファイル内の行数、単語数、文字数をカウントして表示

```
wc [option] ファイル
    -l : 行数を表示
```

- od
    - ファイルを8進数や他の形式で表示

```
od [option] ファイル
    -t c : ASCII文字で表示(規定は8進数)
```

- tr
    - 文字の置換、削除
    - tr -d 削除する文字
    - tr 置換前の文字 置換後の文字
    - 英子文字:[:lower:]
    - 英大文字:[:upper:]

```
tr [option] 文字1 [文字2]
    -d : 指定された文字を削除

[root@localhost com]# cat 1 | tr -d "a"
1::s
2:d:f
3:t:y

[root@localhost com]# cat 1 | tr [:lower:] [:upper:]
1:A:S
2:D:F
3:T:Y
```

3-3. 基本的なファイル管理の実行

- ls
    - ディレクトリの内容を参照

```
-a : ドットファイルも含めたすべてのファイル/ディレクトリを表示
-d : ディレクトリ自身の内容を表示
-h : ファイルサイズをKB、MBといった単位で表示
-i : iノード番号を表示
-l : パーミッションやタイムスタンプなどの情報も表示
-R : サブディレクトリも展開して表示
```

- touch
    - ファイルのタイムスタンプを更新/空のファイルを作成

```
touch [option] ファイル
    -t MMDDHHmm : 指定した日時にタイムスタンプを更新
```

- ワイルドカードによるファイル指定
    - 「*」：指定した位置に0文字以上の任意の文字
    - 「?」：指定した位置に1文字の任意の文字
    - [n],[n-n]　：指定した位置に[]内に含まれる文字。ハイフンで連続文字の指定
    - [!n],[!n-n]　：指定した位置に[]内に含まれない文字
- cp

    - ファイルのコピー

```
cp [option] ファイル1 ファイル2/ディレクトリ
    -r : ディレクトリをコピー
    -f : 上書き時、確認メッセージを非表示にする
    -i : 上書き時、確認メッセージの表示を強制する
```

- mv
    - ファイルの移動/名前の変更

```
mv [option] ファイル1 ファイル2/ディレクトリ
    -f : 上書き時、確認メッセージを非表示にする
    -i : 上書き時、確認メッセージの表示を強制する
```

- rm
    - ファイルの削除

```
rm [option] ファイル
    -r : ディレクトリを削除
    -f : 確認メッセージを非表示にする
    -i : 確認メッセージの表示を強制する
```

- mkdir
    - ディレクトリの作成

```
mkdir [option] ディレクトリ
    -p : 親ディレクトリも一緒に作成
    -m パーミッション値 : 作成するディレクトリのパーミッションを指定
```

- rmdir
    - 空のディレクトリの削除

```
rmdir [option] ディレクトリ
    -p : 親ディレクトリも一緒に削除
```

- split
    - 指定したファイルを分割

```
split [option] ファイル [分割後の名前]
    -l 行数 : 分割する行数(規定では1000行で分割)
```

- tar
    - ファイルのアーカイブ/展開

```
tar [option] ファイル
    -c : アーカイブファイルの作成
    -x : アーカイブファイルの展開
    -t : アーカイブファイルの内容を参照する
    -v : 処理したファイルの一覧を表示
    -f ファイル名 : アーカイブファイル名の指定
```

- 圧縮ファイルの種類
    - gzip形式　　gunzip　（tar -z）
    - bzip2形式　　bunzip2　（tar -j）
    - xz形式　　unxz　　（tar -J）
- cpio
    - cpioによるアーカイブ処理を実行

```
cpio option
    -i : コピーインモード（アーカイブからファイルを取り出す）
    -o : コピーアウトモード（アーカイブの作成）
    -p : コピーパスモード（異なるディレクトリ領域にファイルを出力）
※パイプやリダイレクトを利用してアーカイブファイルの指定を行う
```

- dd
    - ddによるアーカイブ処理を実行
    - パーテーション領域をまるごとアーカイブ操作する

```
dd [option] if=出力元 of=出力先
```

- file
    - ファイルの種類の特定

```
file [option] ファイル
```

3-4. リダイレクトとパイプ

3-4-1. リダイレクト

```
コマンド > ファイル     #上書きでリダイレクト
コマンド >> ファイル     #追記でリダイレクト

コマンド > ファイル     #標準出力をリダイレクト
コマンド 2> ファイル     #標準エラー出力をリダイレクト

コマンド > ファイル 2>&1     #標準出力と標準エラー出力をリダイレクト
コマンド &> ファイル

コマンド < ファイル       #標準入力をリダイレクト
コマンド << 終了文字列     #ヒアドキュメント、終了文字列により処理を終了
[root@localhost ~]# cat << EOF > nn.txt
> 1
> 2
> 3
> EOF


※リダイレクトは出力先と出力元に同じファイルは設定できない
```

- tee
  - 標準入力の内容を標準出力とファイル両方に送る

```
tee [option] [ファイル]
    -a : ファイルに上書きではなく、追記をする
```

- パイプ（|）

3-5. プロセスの生成、監視、終了

- ps
  - プロセスの一覧表示
    - PID : プロセスID
    - TTY : 端末
    - STAT : プロセスの状態
    - TIME : CPUの稼働時間
    - COMMAND : プロセスの実行内容

```
ps [option]

    #BSD形式
    a : ユーザが実行したすべてのプロセスを表示
    f : 親子関係も表示する
    l : nice値などを含めた詳細情報を表示
    x : 制御端末のないサービスなどのプロセスを表示
```

```
    #Unix形式
    -e : すべてのプロセスを表示
    -f : 起動時間などを含めた詳細情報を表示
    -l : nice値などを含めた詳細情報を表示
```

- pstree
    - 動作しているプロセスをツリー状に表示

```
pstree [option]
    -p : PIDを表示
```

- pgrep
    - 動作しているプロセスを検索して表示

```
pgrep [option] キーワード
    -l : PIDのほか、プロセス名も表示
    -u ユーザ : 指定したユーザ権限で実行されているプロセスを表示
```

- top
    - 現在動作しているプロセスやシステムリソースの状態をリアルタイムに表示

```
top [option]
```

- uptime
    - 現在動作しているシステムリソースの状態を表示
- free
    - システムのメモリの状態を表示

ジョブ関連

バックグラウンドジョブとしての処理

◎末尾に「&」をつけて実行する

- jobs
    - 実行中のジョブ一覧を表示
- fg ジョブ番号
    - バックグラウンドジョブをフォアグラウンドに切り替え
- bg ジョブ番号
    - フォアグラウンドジョブをバックグラウンドに切り替え
- nohup コマンド構文
    - ログアウトしてもハングアップシグナルを無視して、コマンドを実行
- screen
    - スクリーンによる操作を管理

```
screen [option]
    -ls : 起動中のスクリーン一覧を表示
    -r PID : 指定したスクリーンに再接続

    Ctrl+a -> d : スクリーンを切断
    Ctrl+a -> \ : スクリーンを終了
```

- kill
    - 指定したプロセスもしくはジョブにシグナルを送信

```
kill [option] PID | %ジョブ番号

    - シグナル名/シグナル番号 : 対象のシグナルを実施
    -s シグナル名/シグナル番号  :  送信するシグナルを指定(規定は15)
    -l                        : シグナルの一覧を表示
```

シグナル番号	シグナル番号	意味
1	HUP	再起動
2	INT	割り込み(Ctrl+c)
9	KILL	強制終了
15	TERM	終了
20	TSTP	サスペンド(Ctrl+z)

- killall [option] 名前
    - 指定した名前のプロセスにシグナルを送信
- pkill [option] キーワード
    - 指定したキーワードに一致するプロセスにシグナルを送信

3.6 プロセスの実行優先度の変更

メモリに読み込まれたプロセスは、実行優先度の高いものがより多くCPUによって処理される。 その優先度を定めたものが「nice値」といわれ、値が低いほうが優先度が高くなる。

- PRIが優先度
- NIがnice値

```
[root@localhost ~]# ps axl
F   UID   PID  PPID PRI  NI    VSZ   RSS WCHAN  STAT TTY  TIME COMMAND
1     0     2     0  20   0      0     0 kthrea S    ?    0:00 [kthreadd]
1     0     4     2   0 -20      0     0 worker S<   ?    0:00 [kworker/0:0H]
```

- nice
  - nice値を指定してコマンドを実行

```
nice [option] コマンド構文

    -n nice値(-10) : nice値の指定
    nice値(--10)
```

- renice [option] nice値 PID
  - 実行中のプロセスのnice値を変更する

正規表現を使用したテキストファイルの検索

- grep

```
-e : 検索キーワードの指定（複数）
-E : 拡張正規表現の利用（egrep）
-v : 否定条件
```

- 正規表現

```
* : 前の文字の0回以上の繰り返し
+ : 前の文字の1回以上の繰り返し
. : 指定した位置に1文字の任意の文字
[n],[n-n] : 指定した位置に[]内に含まれる文字
[^n],[^n-n] : 指定した位置に[]内に含まれない文字
^ : 行頭
$ : 行末
```

```
grep -E '(80|443)/tcp' /etc/services
    tcc-http        24680/tcp               # TCC User HTTP Service
    ethercat        34980/tcp               # EtherCAT Port
```

```
fgrep 'bash$' /etc/passwd
```

- sed
  - 文字列の編集を実行

```
#置換
 sed [option] 's/検索文字列/置換後の文字列/' ファイル
```

```
#削除
 sed [option] 'd/削除する文字列' ファイル

    -iでファイルの内容を直接編集

s/~~/~~/g : すべての文字列
```

- expand
  - タブをスペースに変換
  - -i : 行頭のタブのみを変換する
- unexpand
  - 行頭にある2文字以上連続したスペースもしくはタブをタブに変換
  - -a : 行頭以外の連続スペースもすべて変換

補足

viコマンドを実行しているとき、エディタ上でコマンド実行できる -> **[:!ls]**