LPIC-1
Preparation
Webinar

「新入社員研修のプロが伝授！
LPIC-1学習ポイント」

実施日：2020/06/07

■ 講師プロフィール

山本　篤美　　群馬県出身

2006年 ALJ創業期に技術職として未経験で入社。

2010年 4月から大規模開発を専門とするIT事業本部の主任に就任。

2011年 9月フリーランスに転身。2013年合同会社プラスアイ設立。
スマートホンアプリ開発及びスクール事業を開始。

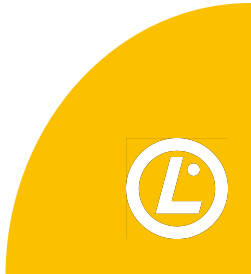
2015年10月 IT教育事業専門会社、ALJ Education Plus(株)代表取締役就任。
設立から4期連続増収増益達成。（現任）

2018年7月 (株)ALJ 営業本部兼IT事業本部本部長に就任。
約100名のエンジニアの社員マネジメントを経験。（現任）

2020年4月 ALJ DX Tech(株)設立、代表取締役就任。（現任）

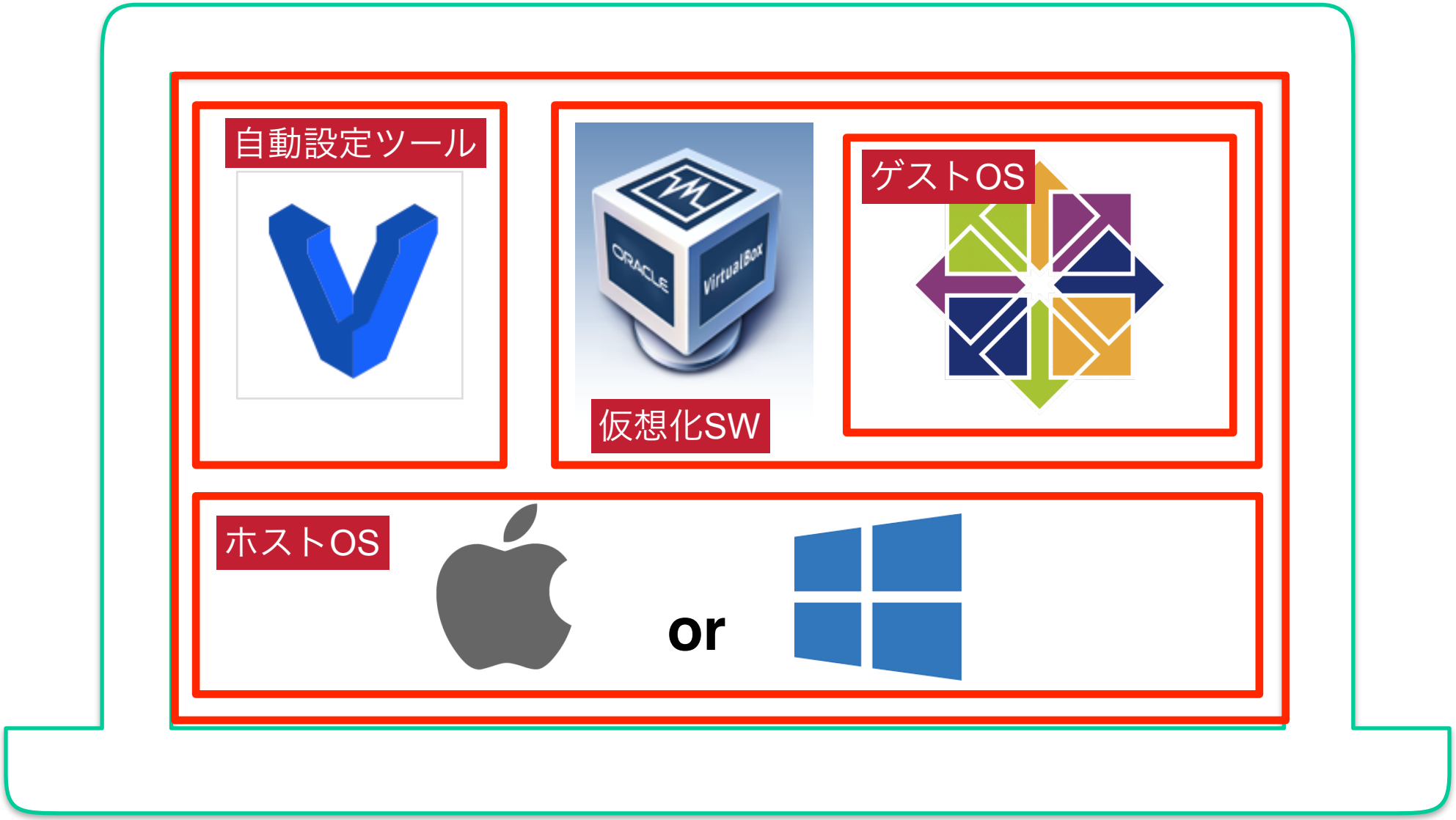
■ 本日のアジェンダ

- 学習環境構築
- LPIC-1の概要
- Linuxの基本的な操作方法
- GNUとUnixコマンド



学習環境構築

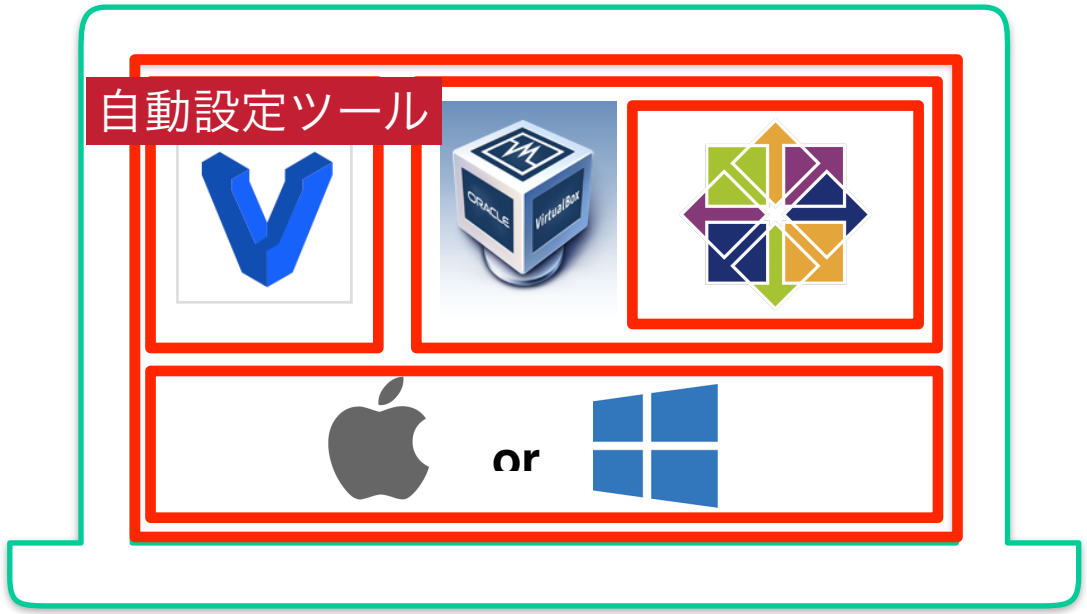
■ 学習環境に必要なSW



■ 仮想化SW

https://www.virtualbox.org/

■ 仮想化SW 自動設定ツール



HashiCorp Vagrant

https://www.vagrantup.com

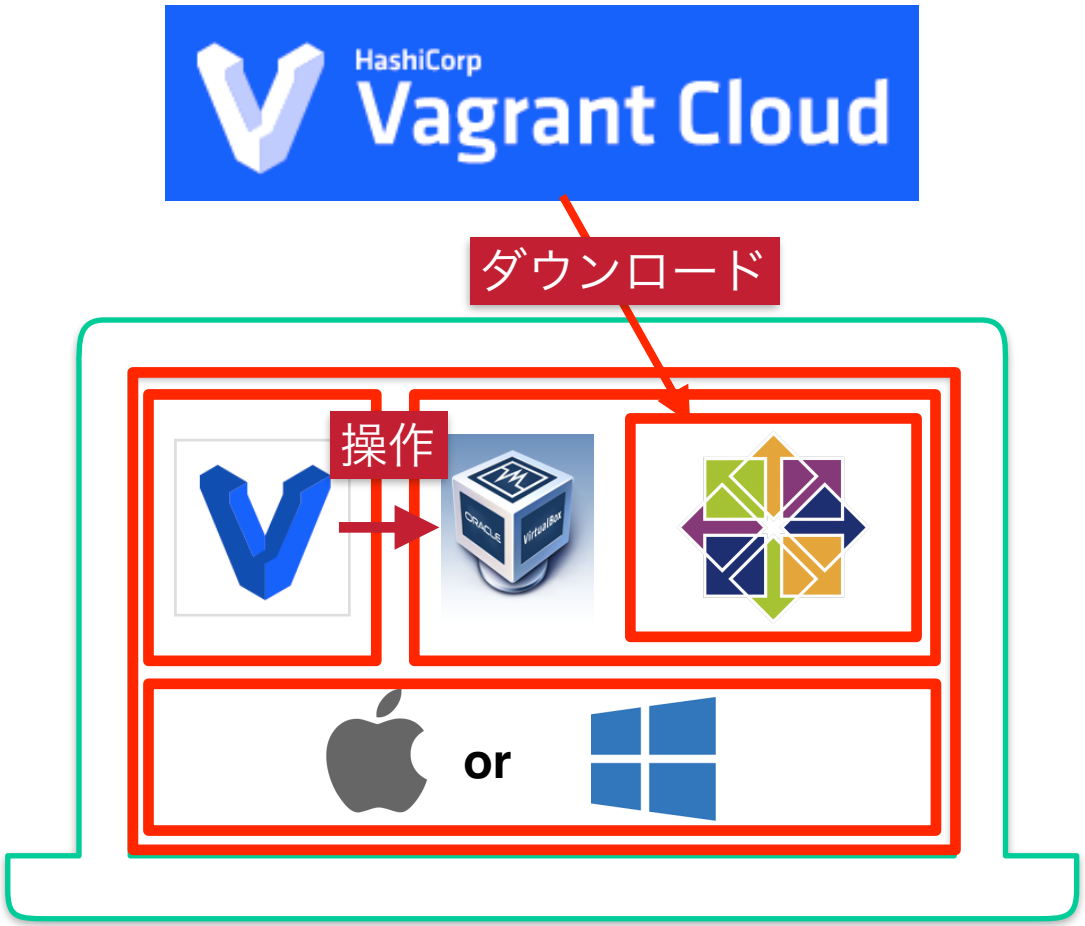
■ ゲストOS

CentOS

https://www.centos.org

■ イメージファイルをダウンロード

https://app.vagrantup.com/centos/boxes/7



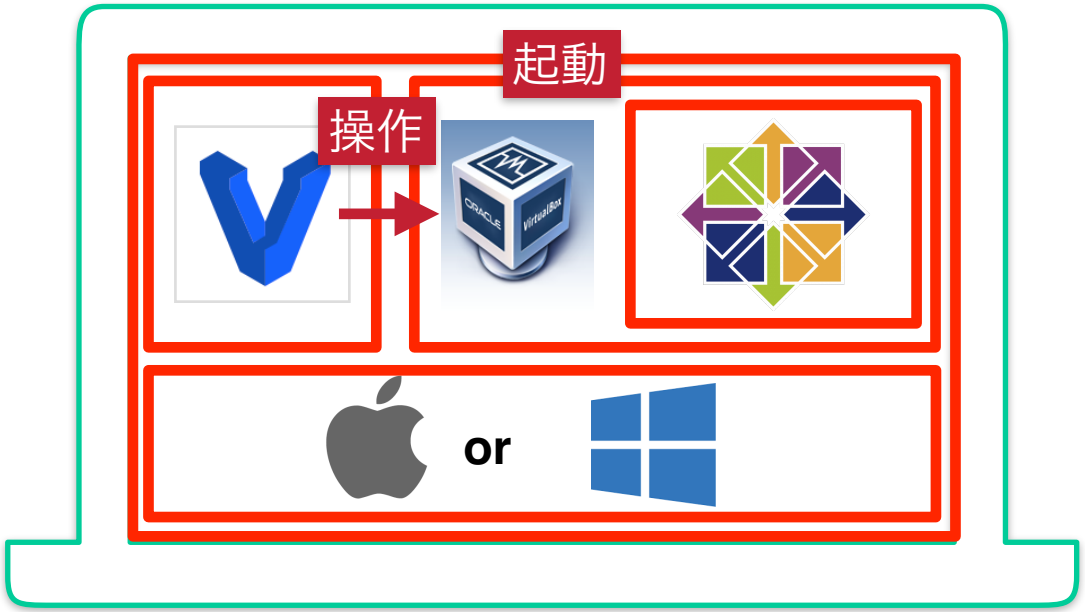

```
$ VBoxManage -v
6.1.8r137981
$ vagrant -v
Vagrant 2.2.9
$ mkdir centos7 && cd centos7
$ vagrant box list
There are no installed boxes! Use `vagrant box add` to add some.
$ vagrant box add centos/7
==> box: Loading metadata for box 'centos/7'
    box: URL: https://atlas.hashicorp.com/centos/7
This box can work with multiple providers! The providers that it
can work with are listed below. Please review the list and choose
the provider you will be working with.

1) hyperv
2) libvirt
3) virtualbox
4) vmware_desktop

Enter your choice: 3
==> box: Adding box 'centos/7' (v1708.01) for provider: virtualbox
省略
$ vagrant box list
centos/7 (virtualbox, 1708.01)
```

■ 初期設定

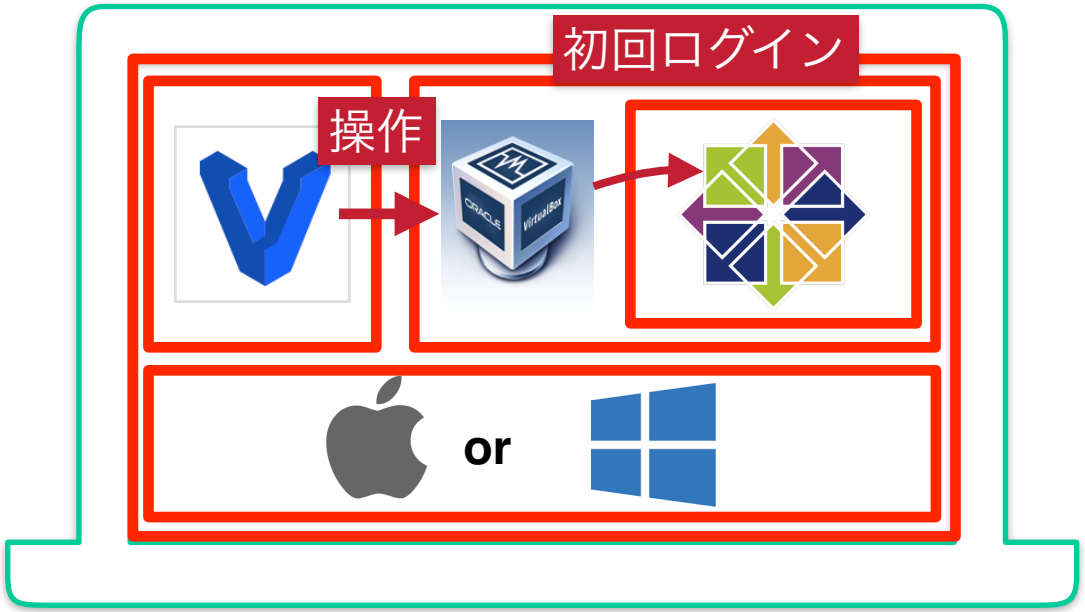
Vagrantfile




```
$ vagrant init centos/7
$ ls
Vagrantfile
$vagrant status
Current machine states:
default                   not created (virtualbox)
The environment has not yet been created. Run `vagrant up` to
省略
then the machine is not created for that environment.
$ vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Checking if box 'centos/7' is up to date...
$ vagrant status
Current machine states:
default                   running (virtualbox)
The VM is running. To stop this VM, you can run `vagrant halt` to
省略
simply run `vagrant up`.
```

■ 初回ログイン

vagrant ssh




```
$ vagrant ssh
[vagrant@localhost ~]$ id
uid=1000(vagrant) gid=1000(vagrant) groups=1000(vagrant)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[vagrant@localhost ~]$ pwd
/home/vagrant
[vagrant@localhost ~]$
```

LPIC-1の概要

■ LPIC-1

- Linux管理者

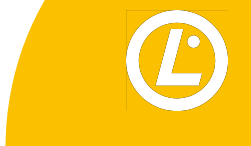
  → コマンドラインで保守作業を実行する能力

LPIC-1 101-500

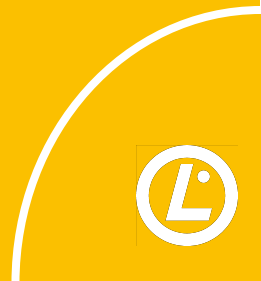
https://www.lpi.org/ja/our-certifications/exam-101-objectives

LPIC-1 102-500

https://www.lpi.org/ja/our-certifications/exam-102-objectives



Linuxの基本的な操作方法



■ 管理者と一般ユーザー

Linuxのユーザーは「管理者」と「一般ユーザー」、「システムアカウント」の3つに分けられる

・ 管理者

→ rootユーザー（スーパーユーザー）とも呼ばれる

→ Linuxシステムの全ての操作ができる

→ 管理者は1つのLinux上に1アカウントのみ存在する

・ 一般ユーザー

→ Linuxシステムの限られた操作のみできる

→ rootユーザーのみが作成可能

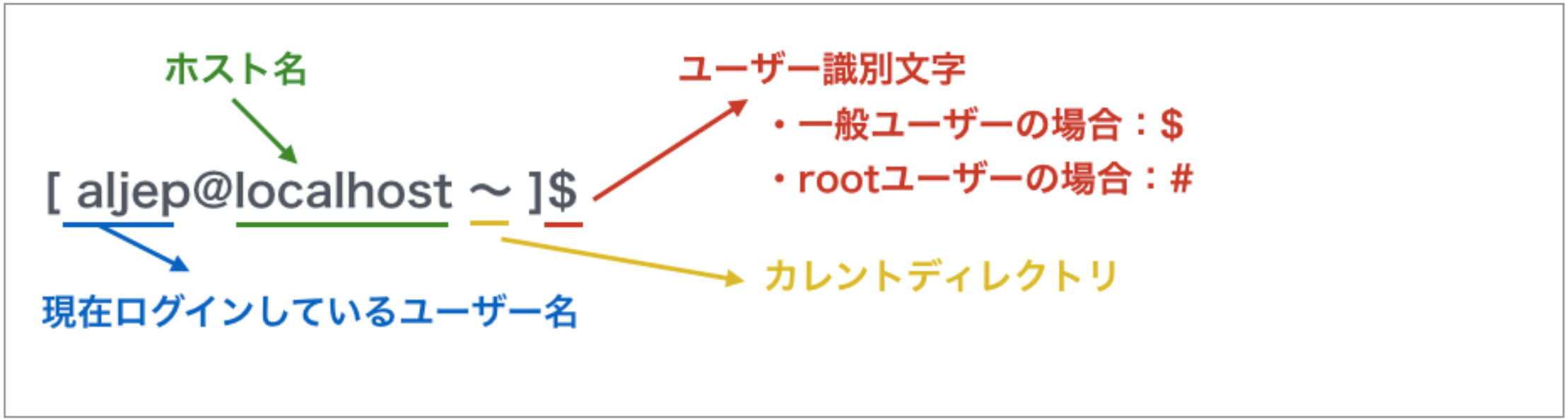
・ システムアカウント

→ 特定のアプリケーション（apache、smb）を実行する際に利用される

→ 特殊なユーザーアカウント

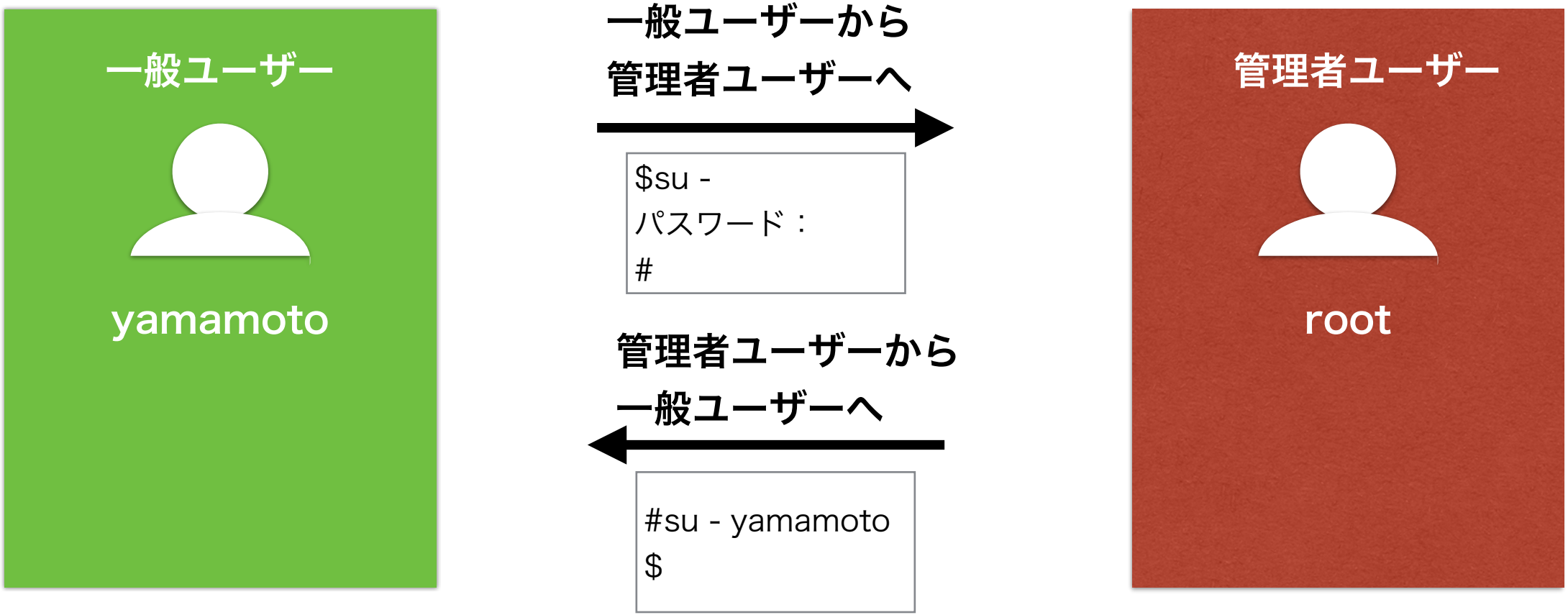
■ プロンプト

・ユーザーからの入力を待ち受ける際に表示される文字列
・ログインしているユーザーが管理者なのか一般ユーザーなのか一目で確認できる



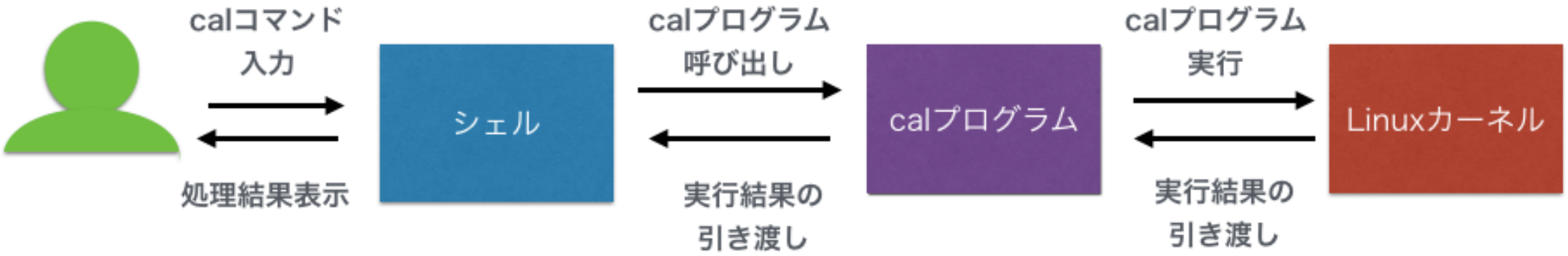
■ ユーザーの切り替え

ユーザーを切り替える場合には suコマンド (Substitute User) を利用する



■ コマンドとは

- あらかじめ機能が決められた英文字をいくつか組み合わせた文字列
- 処理内容はコマンドごとに決められている
- ほとんどのコマンドは英単語の省略形
- 大文字小文字は全て区別される
- コマンドはシェルによってLinuxカーネルへ渡され、実行される
- 実行結果はシェルを介してユーザーへ伝えられる



■ オプションと引数

- コマンドを実行する場合、必要に応じて「オプション」や「引数」を指定する
- オプションは、対象のコマンドに特殊な動きをさせる記号のこと
- 通常「- (ハイフン)」をつけて指定する
- 引数は対象のコマンドの処理に必要となる値や実行対象を指定するもの
- オプションと引数の間には「半角スペース」 を入れる

calコマンドのオプションと引数の例

■ 複数のオプション指定

・ オプションは**複数同時**に指定することができる

・ lsコマンドの複数オプションの指定例①

```
[ aljep@localhost ～ ]$ ls -a -l
```

オプションを別々に指定

・ lsコマンドの複数オプションの指定例②

```
[ aljep@localhost ～ ]$ ls -al
```

オプションを一緒に指定

■ コマンド履歴の利用

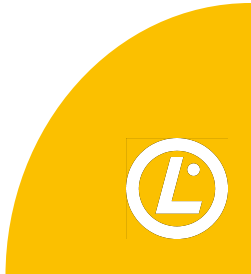
- シェルは入力されたコマンドを履歴として記録する
- 入力したコマンドを履歴から呼び出すことができる

→ 履歴機能

→ キーボードの方向キーの上下矢印キー（[↑]、[↓]）で実行

→historyコマンドを実行

→ オプションを指定しないとデフォルトで1,000個前までの履歴が表示する



■ historyコマンド

- コマンド履歴を表示するコマンド

historyコマンドの書式

```
history ［ オプション ］［ 表示するコマンド数]
```

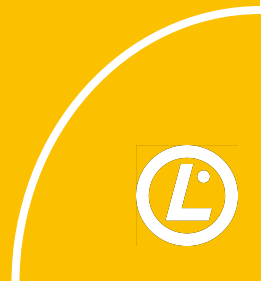
historyコマンドのオプション

オプション	説明
-c	コマンド履歴を全て消去
-d[番号]	指定した番号のコマンド履歴を消去する

参考

- 環境変数「HISTFILE」→ コマンド履歴を格納するファイル名
- 環境変数「 HISTFILESIZE」→「.bash_history」ファイルに記録できるコマンド履歴の件数
- 環境変数「 HISTSIZE」→使用中のbashの履歴数の設定

GNUとUnixコマンド



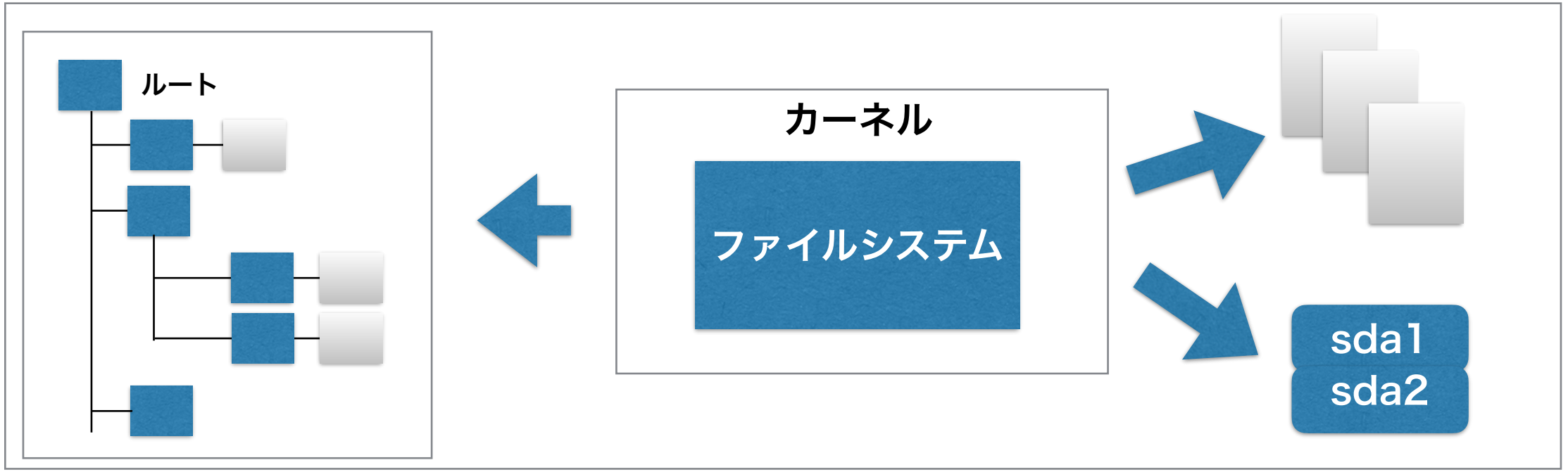
■ ファイルシステム

・ 記憶装置上でデータがどのように格納されているかを管理する仕組みのこと

→ ファイルやディレクトリの作成、削除、移動を行う方法

→ データを記録する方式

→ 管理領域の場所



■ ディレクトリ構成

- **Linuxのディレクトリ構成**

  →「/」(ルート)を頂点としたディレクトリツリーで構成されている

  → 標準化されている

  → FHS(Filesystem Hierarchy Standard)

ディレクトリ	概要
/	ルートディレクトリ。全てのディレクトリは親ディレクトリをたどって行くと、このディレクトリにたどり着きます。
/bin	binary(2進数)という意味を持つ言葉で、Linuxで用いる様々なコマンドが入っている。
/boot	bootは起動という意味です。Linuxが起動するためのファイルが入っています。
/dev	device(周辺機器)の略です。周辺機器を表す特殊ファイル(ディバイスファイル)が格納されています。
/etc	ET Cateraの略。システム環境設定ファイルが格納されています。

■ ホームディレクトリとカレントディレクトリ

- **ホームディレクトリ**
  - → ログインした際に、最初にいる場所(ディレクトリ)
  - → 一般ユーザーの場合は「/home」ディレクトリ配下
    - → ユーザーごとの専用ディレクトリが用意されている
  - → rootユーザーの場合は「/root」がホームディレクトリ

- **カレントディレクトリ**
  - → 現在操作を行っているディレクトリ
  - → 端末を起動した直後は、ログインしたユーザーのホームディレクトリがカレントディレクトリ

■ 相対パスと絶対パス

- **パス**

  → 数あるファイルの中から、一つのファイルを指定する記述方法を「パス」という

- **絶対パス**

  → ルートを起点としてパスを記述する方法

- **相対パス**

  →カレントディレクトリを起点としてパスを記述する方法

ディレクトリ記号	意味
/ (スラッシュ)	パスの先頭に記述すると、ルートディレクトリを表す。パスの途中で記述するとディレクトリを表す。
./ (ドット・スラッシュ)	カレントディレクトリを表す
../ (ドット・ドット・スラッシュ)	一つ上のディレクトリを表す(親ディレクトリを表す)
～ (チルダ)	ホームディレクトリを表す
～- (チルダ・ハイフン)	直前にいたディレクトリ
～[ユーザー名] (チルダ・ユーザー名)	指定したユーザーのホームディレクトリ

■ カレントディレクトリの確認

・ pwd (Print Working Directoroy)
→ カレントディレクトリの場所を確認するコマンド

・ コマンド書式

```
pwd
```

・ pwdコマンドの実行例

```
$ pwd
/home/yamamoto
```

```
# pwd
/root
```

■ カレントディレクトリの移動

・ cd (Change Directory)
→ カレントディレクトリを移動するためのコマンド

・ コマンド書式

```
cd　[ 移動先ディレクトリ]
```

・ cdコマンドの実行例1 (絶対パスで「/etc」ディレクトリに移動)

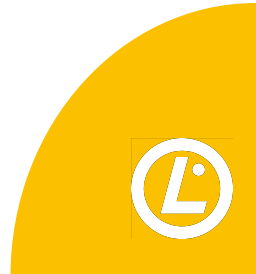
```
$ cd  /etc
```

・ cdコマンドの実行例2 (ホームディレクトリに移動)

```
$ cd
$ cd  ~
```

・ cdコマンドの実行例1

```
$ cd  ~yamamoto
```



■ ファイルやディレクトリの情報を表示する

・ ls (List)コマンド

→ ディレクトリの内容一覧を表示するコマンド

・ コマンド書式

```
ls　[ オプション ] [ パス名 ]
```

・ 主なオプション

オプション	説明
-a (—a)	隠しファイルを含む全てのファイルやディレクトリを表示する
-l (—format=long)	ファイルやディレクトリ詳細情報を表示する
-R (—recursive)	サブディレクトリも含めて表示する
-d (—directory)	指定したディレクトリ自身の情報を表示する
-i (—inode)	ファイル名の左にインデックス番号をつけて表示する

■ ファイルやディレクトリの情報を表示する

・lsコマンドの実行例1（カレントディレクトリ内を対象とした場合）

```
$ ls          ← オプションなし
$ ls -a -l    ← 「-a」オプションと「-l」オプションを別々に指定
$ ls -al      ← 「-a」オプションと「-l」オプションをまとめて指定
```

・lsコマンドの実行例2（参照するディレクトリを指定する場合）

```
$ ls  mydir/                    ← 相対パスで指定
$ ls  /home/yamamoto/mydir/     ← 絶対パスで指定
```

■ ディレクトリの作成

・ mkdir (Make Directory) コマンド

→ 新規にディレクトリを作成するためのコマンド

・ コマンド書式

```
mkdir　[ オプション ] [ ディレクトリ名 ]
```

・ 主なオプション

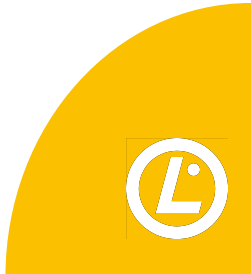
オプション	説明
-p (—parents)	指定されたディレクトリの上位のディレクトリも作成する

・ mkdirコマンドの実行例1 (カレントディレクトリ配下に「LPIC」ディレクトを作成)

```
$ mkdir LPIC
```

・ mkdirコマンドの実行例2 (カレントディレクトリ配下に「saitama」、その配下に「oomiya」ディレクトリを作成)

```
$ mkdir -p saitama/oomiya
```



■ 空のディレクトリの削除

・ rmdir (ReMove empty Directoryes) コマンド

→ 空のディレクトリを削除するためのコマンド

・ コマンド書式

```
rmdir　[ オプション ] [ ディレクトリ名 ]
```

・ 主なオプション

オプション	説明
-p (—parents)	指定されたディレクトリの上位のディレクトリも削除する

・ rmdirコマンドの実行例1 (空のカレントディレクトリ「olddir」)

```
$ rmdir older
```

・ rmdirコマンドの実行例2 (「olddir2」ディレクトリ配下に空のディレクトリ「data」が存在する場合)

```
$ rmdir -p olddir2/data
```

■ ファイルのタイムスタンプの修正、空ファイルの作成

・ touchコマンド

→ 指定したファイルが保持しているタイムスタンプを変更するコマンド

→ オプションを指定しないで実行した場合は、最終更新日時が現在の日付に変更される

→ 指定したファイルが存在しない場合は、空ファイルが作成される

・ コマンド書式

```
touch　[ オプション ] [ ファイル名 ]
```

・ 主なオプション

オプション	説明
-a (—time=atime、 —time=access)	最終アクセス日付のみを変更する
-m (—time=mtime、 —time=modify)	最終更新日時のみを変更する (デフォルト)
-t ([[CC]YY]MMDDhhmm[.ss])(time)	指定した時間にタイムスタンプを変更する CCは西暦の上2桁、YYは西暦の下2桁、MMは月、DDは日、hhは時、mmは分、ssは秒

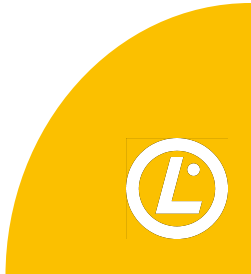
■ ファイルのタイムスタンプの修正、空ファイルの作成

・ touchコマンドによる空ファイルの作成例

```
$ touch yellow                ← ファイル名「yellow」で空ファイルを作成
$ ls -l                       ← 作成ファイルの確認
-rw-rw-r--. 1 yamamoto yamamoto 0 Feb  6 02:53 yellow
```

・ touchコマンドによるファイルのタイムスタンプの変更例

```
$ touch -t 201501010000.00 yellow
-rw-rw-r--. 1 yamamoto yamamoto 0 Jan  1  2015 yellow
```



■ ファイルやディレクトリのコピー

・ cp (Copy)コマンド

→ ファイルやディレクトリのコピーを作成するためのコマンド

・ コマンド書式

```
cp　[ オプション ] [ コピー元ファイル名 ]　[ コピー先ファイル名 ]
```

・ 主なオプション

オプション	説明
-r (—recursive)	ディレクトリ内を再帰的にコピーする(ディレクトリごとコピする)
-i (—interactive)	(上書きが発生する場合に)　コピーする際に確認する
-p (—preserve)	ファイルの属性を維持したままコピーする

■ ファイルやディレクトリのコピー

・cpコマンドによるコピーの例

```
$ cp file1 file2          ←「file1」を同一ディレクトリ内に「file2」の名前でコピー
$ cp file1 mydir2         ←「file1」を「mydur2」ディレクトリ内に同一ファイル名でコピー
$ cp file1 mydir2/file2   ←「file1」を「mydir2」ディレクトリ内に「file2」の名前でコピー
$ cp -r mydir3 newdir     ←「mydir3」ディレクトリを丸ごと「newdir」ディレクトリ内に
                            同一名でコピー
```

■ ファイルやディレクトリのコピー

・確認メッセージの表示
　→ コピー時に、同名のファイルやディレクトリが存在する場合、警告なしで上書きされる
　→ オプション「 -i 」をつけると確認メッセージが表示される

・「 -i 」オプションによる確認メッセージの表示例

```
$ cp -i file1 file2　← 「file1」を同一ディレクトリ内に「file2」の名前でコピー
cp: overwrite ‘file2’?
```

■ ファイルやディレクトリのコピー

・ファイル属性のコピー
　→ オーナー（所有者）情報やグループ情報などの属性情報を維持したままファイルのコピーを行う場合は、オプション「-p」をつける

・「 -p 」オプションを指定しない場合のコピー例

```
# ls -l /home/yamamoto/file1
-rw-rw-r--. 1 yamamoto yamamoto 0 Feb  6 03:19 /home/yamamoto/file1
# cp /home/yamamoto/file1  /home/file10
# ls -l /home/file10
-rw-r--r--. 1 root root 0 Feb  6 03:25 /home/file10
```

■ ファイルやディレクトリのコピー

・「 -p 」オプションを指定した場合のコピー例

```
# ls -l /home/yamamoto/file1
-rw-rw-r--. 1 yamamoto yamamoto 0 Feb  6 03:19 /home/yamamoto/file1
# cp -p /home/yamamoto/file1 /home/file20
# ls -l /home/file20
-rw-rw-r--. 1 yamamoto yamamoto 0 Feb  6 03:19 /home/file20
```

■ ファイルやディレクトリの移動、名前の変更

・ mv (Move)コマンド

→ ファイルやディレクトリを移動したり名前を変更するためのコマンド

・ コマンド書式

```
mv　[ オプション ] [ 移動元のパス名 ]　[ 移動先のパス名 ]
```

・ 主なオプション

オプション	説明
-f (—force)	移動先にある同名のファイルを強制的に上書きする
-i (—interactive)	(上書きが発生する場合に)　移動する際に確認する

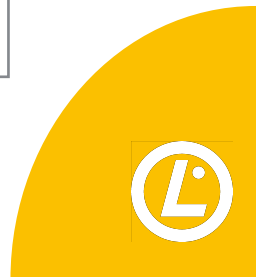
■ ファイルやディレクトリの移動、名前の変更

・ mvコマンドでファイルを移動する例

```
$ mv mvfile1 mvdir1/mvfile1
$ ls mvdir1/
mvdir1
```

・ mvコマンドでファイル名を変更する例

```
$ mv ken2 ken
$ ls
ken
```



■ ファイルやディレクトリの削除

・ rm (ReMove)コマンド

→ ファイルやディレクトリを削除するためのコマンド

・ コマンド書式

```
rm　[ オプション ] [ ファイル名 ]
```

・ 主なオプション

オプション	説明
-f (—force)	ユーザーへの確認なしに、強制的に削除を行う
-i (—interactive)	削除前に確認メッセージを表示する
-r (—recursive)	サブディレクトリも含めて、再帰的にディレクトリ全体を削除する

■ ファイルやディレクトリの削除

・rmコマンドでファイルを削除する例1

```
$  rm rmfile1
```

・rmコマンドでファイルを削除する例2 (複数ファイルを指定して削除)

```
$  rm rmfile2 rmfile3 rmfile4
```

・rmコマンドでディレクトリを削除する例 (空ではないディレクトリ「rmdir2」を指定して削除)

```
$  rm rmdir2          ← オプションなしで実行
rm: cannot remove ‘rmdir2’: Is a directory
$  rm -r rmdir2          ← オプション「 -r 」を指定して実行
```

■ ファイルの種別を表示する

・ fileコマンド

→ ファイルの種類を調べることができるコマンド

・ コマンド書式

```
file ［ ファイル名 ］
```

・ fileコマンドの実行例

```
$  file file1
file1: ASCII text
$  file /home/yamamoto/
$ file /bin/bash
/bin/bash: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically
linked (uses shared libs), for GNU/Linux 2.6.32,
BuildID[sha1]=ab347e897f002d8e3836479e2430d75305fe6a94, stripped
```

■ ファイルの内容を表示する

・ cat (Catinate)コマンド

→ ファイル内容を表示するためのコマンド

・ コマンド書式

```
cat ［ ファイル名 ］
```

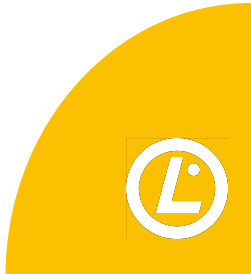
・ 主なオプション

オプション	説明
-n (—number)	各行の左端に、先頭から空白も含めて行番号を付与する
-b (—number-nonblank)	空白でない行に、先頭から行番号を付与する

■ ファイルの内容を表示する

- catコマンドでファイル内容を表示する例1(「/etc/passwd」ファイルの内容を表示する)

```
$  cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
~ 途中省略 ~
ken-nw:x:1001:1002::/home/ken-nw:/bin/bash
yamamoto:x:1002:1003::/home/yamamoto:/bin/bash
```



■ ファイルの内容を表示する

・ catコマンドでファイル内容を表示する例2(「/etc/passwd」ファイルを行番号付きで表示する)

```
$  cat -n /etc/passwd
     1  root:x:0:0:root:/root:/bin/bash
     2  bin:x:1:1:bin:/bin:/sbin/nologin
     3  daemon:x:2:2:daemon:/sbin:/sbin/nologin
~ 途中省略 ~
    25  ken-nw:x:1001:1002::/home/ken-nw:/bin/bash
    26  yamamoto:x:1002:1003::/home/yamamoto:/bin/bash
```

・ 存在しないファイルを指定してcatコマンドを実行した例

```
$  cat @@@
 cat: @@@: そのようなファイルやディレクトリはありません
```

■ ファイルの先頭部分を表示する

・ head コマンド

→ ファイルの先頭行のみを表示する

・ コマンド書式

```
head ［ オプション ］［ ファイル名 ］
```

・ 主なオプション

オプション	説明
-n [行数] (—lines=[行数])	先頭から指定した行数だけを表示する
-c[バイト数]	出力するバイト数を指定する

■ ファイルの先頭部分を表示する

・headコマンドでファイルの先頭部分を表示する例1

```
$  head /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
```

■ ファイルの先頭部分を表示する

・ headコマンドでファイルの先頭部分を表示する例2

```
$  head -n 3 /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
```

■ ファイルの末尾部分を表示する

・ tail コマンド

→ ファイルの末尾部分を表示する

・ コマンド書式

```
tail ［ オプション ］［ ファイル名 ］
```

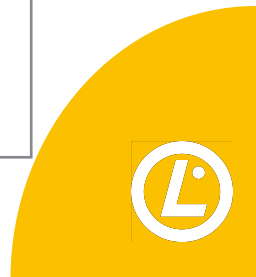
・ 主なオプション

オプション	説明
-n [行数] (—lines=[行数])	末尾から指定した行数だけを表示する
-c[バイト数]	出力するバイト数を指定する
-f (—follow)	ファイルの末尾に動的に追記された行を表示し続ける

■ ファイルの末尾部分を表示する

- tailコマンドでファイルの末尾部分を表示する例1

```
$  tail /etc/passwd
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
chrony:x:998:995::/var/lib/chrony:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
vagrant:x:1000:1000:vagrant:/home/vagrant:/bin/bash
vboxadd:x:997:1::/var/run/vboxadd:/bin/false
ken-nw:x:1001:1002::/home/ken-nw:/bin/bash
yamamoto:x:1002:1003::/home/yamamoto:/bin/bash
```



■ ファイルの末尾部分を表示する

- tailコマンドでファイルの末尾部分を表示する例2

```
$  tail  -n 5 /etc/passwd
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
vagrant:x:1000:1000:vagrant:/home/vagrant:/bin/bash
vboxadd:x:997:1::/var/run/vboxadd:/bin/false
ken-nw:x:1001:1002::/home/ken-nw:/bin/bash
yamamoto:x:1002:1003::/home/yamamoto:/bin/bash
```

■ ファイルの末尾部分を表示する

・ ログをリアルタイム表示

```
$  tail  -f /var/log/messages
Feb  6 04:28:36 localhost chronyd[677]: Selected source 150.95.187.128
Feb  6 04:41:22 localhost su: (to root) vagrant on pts/0
Feb  6 04:47:12 localhost systemd: Started Session 9 of user vagrant.
Feb  6 04:47:12 localhost systemd-logind: New session 9 of user vagrant.
Feb  6 04:47:12 localhost systemd: Starting Session 9 of user vagrant.
Feb  6 04:47:59 localhost systemd: Started Session 10 of user vagrant.
Feb  6 04:47:59 localhost systemd-logind: New session 10 of user vagrant.
Feb  6 04:47:59 localhost systemd: Starting Session 10 of user vagrant.
Feb  6 05:01:02 localhost systemd: Started Session 11 of user root.
Feb  6 05:01:02 localhost systemd: Starting Session 11 of user root.
```

■ ページャ

・ページャ（Peger）とは

→ 1画面(1ページ)に収まりきらないような大きなテキストファイルを分割して表示する機能

→ 「more」や「less」コマンドがある

■ ページャ

・ more コマンド

→ テキストの内容をページ単位で表示する

・ コマンド書式

```
more ［ ファイル名 ］
```

・ 主な操作

操作	説明
[Enter]キー	1行ずつ改行する
[Enter]キー	出力するバイト数を指定する
-f (—follow)	ファイルの末尾に動的に追記された行を表示し続ける

■ ページャ(moreコマンド)

・ moreコマンドの実行例

```
$  more /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
~省略～
rpc:x:32:32:Rpcbind Daemon:/var/lib/rpcbind:/sbin/nologin
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nol
--続ける--(70%)
```

■ ページャ

・ less コマンド

→ テキストの内容をページ単位で表示する（moreコマンドよりも機能が豊富）

・ コマンド書式

```
less ［ オプション ］ ［ ファイル名 ］
```

・ 主な操作

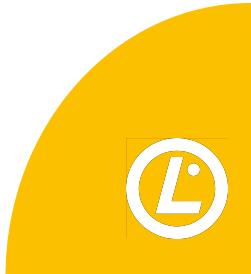
操作	説明
-M (—LONG-PROMPT)	現在表示されている行数とパーセンテージを表示する
-N (—LINE-NUMBERS)	行番号を付与して表示する

■ ページャ(lessコマンド)

・ lessコマンドの実行例

```
$  less /etc/passwd
~途中省略～
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nol
/etc/passwd
```

```
~途中省略～
vboxadd:x:997:1::/var/run/vboxadd:/bin/false
ken-nw:x:1001:1002::/home/ken-nw:/bin/bash
yamamoto:x:1002:1003::/home/yamamoto:/bin/bash
(END)
```



■ ページャ(lessコマンド)

・ 主なlss内部コマンド

キー	説明
[Enter]	次の一行を表示する
[↑]	同上
[↓]	前の一行を表示する
[Space]	次の1画面を表示する
b	前の1画面を表示する
G	ファイルの末尾へ移動する
g	ファイルの先頭へ移動する
/文字列	ファイル内を終端方向に向かって文字列を検索する
?文字列	ファイル内を先頭方向に向かって文字列を検索する
n	検索後、次に一致するパターンを終端方向に向かって検索する
N	検索後、次に一致するパターンを先頭方向に向かって検索する
q	lessコマンドを終了する

■ メタキャラクタの利用

- ワイルドカード
  →メタキャラクタの一種
  →任意の文字列を指定するための特殊な記号
  →複数のファイルを効率的に操作することができる

- 主なワイルドカード

ワイルドカード	説明
*	0文字以上の任意の文字または文字列。ただし、ドットファイルのドット(. (ドット))は除く
?	任意の1文字。ただし、ドットファイルのドット(. (ドット))は除く
[ ]	指定文字のいずれかを表す「 - (マイナス)」により、文字範囲を指定できる
{ }	指定した文字列のいずれかを表す文字列を「, (カンマ) 」により複数列挙できる

■ メタキャラクタの利用

・メタキャラクタを使ってファイルを表示する例1

```
$  ls  /etc/host*
/etc/hosts        /etc/hosts~orig        /etc/hosts.equiv
```

・メタキャラクタを使ってファイルを表示する例2

```
$  ls  a???
a123  abcd
```

・メタキャラクタを使ってファイルを表示する例3

```
$  ls [Tt]est.txt
Test.txt  test.txt
```

■ メタキャラクタの利用

・メタキャラクタを使ってファイルを表示する例4

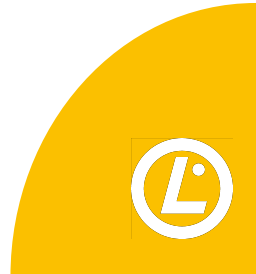
```
$  ls [0-9][0-9].txt
22.txt  92.txt
```

・メタキャラクタを使ってファイルを表示する例5

```
$  ls  {yamada,aoki,inoue}
aoki  inoue  yamada
```

・メタキャラクタの意味を打ち消す
　→ 該当のメタキャラクタの直前に「 \ 」を記述する

```
$  ls  yamada\*
yamada*
```



■ コマンドの使い方を調べる

・ コマンドヘルプの利用（ --helpコマンド ）
　→ 使用するコマンドの書式がわからない場合に利用する

・ 「--help」オプションの利用方法

```
[ 対象とするコマンド ] --help（または「-h」）
```

・ catコマンドについて「--help」オプションで調べる場合

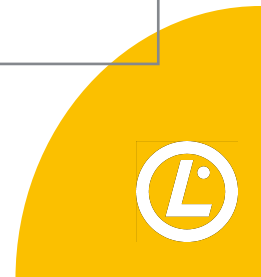
```
$  cat --help
使用法: cat [オプション]... [ファイル]…
ファイル、または標準入力を連結し、標準出力に出力します。
  -A, --show-all           -vETと同じ
  -b, --number-nonblank    空行を除いて行番号を付け加える。-n より優先される
～以下省略～
```

■ コマンドの使い方を調べる

- コマンド実行時にオプションや引数の指定を間違えた場合などにも、ヘルプが表示されることがある

- fileコマンドを引数なしで実行した場合

```
$ file
Usage: file [-bchikLlNnprsvz0] [--apple] [--mime-encoding] [--mime-type]
            [-e testname] [-F separator] [-f namefile] [-m magicfiles] file ...
       file -C [-m magicfiles]
       file [--help]
```



■ コマンドの使い方を調べる

・オンラインマニュアルの利用（manコマンド）

→　詳しく調べたい場合にはmanコマンド(オンラインマニュアル)を利用する

→ オンラインマニュアルの操作方法はlessコマンドと同じ

→ 9種類のセクションに分かれている

→ マニュアルの配置場所

ディストリビューションに含まれる場合は「/usr/share/man」に配置される

個別でインストールした場合は「/usr/local/share/man」に配置される

・manコマンドの書式

```
man ［ オプション ］［ セクション番号 ］　［ コマンド名など ］
```

■ コマンドの使い方を調べる

・主なオプション

オプション	説明
-a (—all)	全ての一致したマニュアルページを探し出す
-f (—whatis)	指定されたキーワード（完全一致）を含むドキュメントの表示
-k (—apropos)	指定されたキーワード（部分一致）を含むドキュメントの表示

・セクション

セクション	説明
1	一般コマンド
2	システムコール(カーネル関数、プログラミングに必要)
3	ライブラリ関数（C言語の関数で、プログラミングに必要）
4	デバイスファイル（プログラミングに必要）
5	ファイル形式
6	ゲーム
7	その他
8	システム管理コマンド
9	カーネル開発（LInux独自のカーネルルーチン用ドキュメント）

■ コマンドの使い方を調べる

- lsコマンドのオンラインマニュアルを表示した例

```
$ man ls
LS(1)                    ユーザーコマンド                    LS(1)

名前
       ls - ディレクトリの内容をリスト表示する

書式
       ls [オプション]... [ファイル]...

説明
       FILE                         に関する情報を一覧表示します
~省略~
```

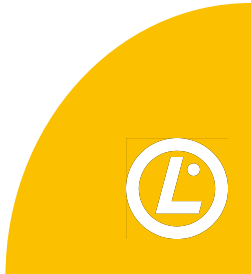
■ コマンドの使い方を調べる

・セクション5の「/etc/passwd」ファイルに関するマニュアルを表示した例

```
$ man 5 passwd
PASSWD(5)          File Formats Manual          PASSWD(5)

名前
       passwd - パスワードファイル

説明
       passwd
       ファイルには各ユーザアカウントの様々な情報が記録されている。
       書かれているのは次の通り。
~ 省略 ~
```



■ コマンドの使い方を調べる

・キーワード「passwd」を含むマニュアルを表示する例

```
$ man -k passwd
afppasswd (1)         - netatalk パスワード管理ユーテ...
passwd (5)            - パスワードファイル
passwd2des (3)        - RFS パスワード暗号化
pwupdate (8)          - NIS マップ passwd および shadow ...
rpc.yppasswdd (8)     - NIS パスワード更新デーモン
yppasswd (1)          - NIS データベースのパスワー...
yppasswdd (8)         - NIS パスワード更新デーモン
grub2-mkpasswd-pbkdf2 (1) - Generate a PBKDF2 password hash.
```

■ コマンドの使い方を調べる

- 日本語マニュアルのインストール方法

```
$ ls /usr/share/man | grep ja
ja          ← 「ja」の表示がされれば、日本語manページはインストール済みです
```

- インストール操作

```
# yum -y install man-pages-ja
```

■　標準入出力

・　標準入力 / 標準出力 /　 標準エラー出力

→ コンピュータの世界にはデータの「入力」、「出力」という流れがある

→ キーボードからの「入力」を「標準入力」という

→ ディスプレイへの「出力」を「標準出力」という

→ エラー（プログラムが正常に処理されなかった場合）の出力を「標準エラー出力」という

→ データの入出力に関する流れを「ストリーム」という

■ 標準入出力

・ ファイルディスクリプタ

→プログラムがアクセスするファイルや標準入出力などOSが識別するために用いる識別子

→ 0から順番に整数の値が割り当てられる

番号	入出力名	デフォルト
0	標準入力	キーボード
1	標準出力	ディスプレイ
2	標準エラー出力	ディスプレイ

■ パイプ

・ パイプとは

→ コマンドやプログラムの処理結果を、別のコマンドやプログラムの入力として引き渡し、
複数のコマンドを組み合わせて使う際に利用する記号

→コマンドとコマンドを繋ぐために使われる記号

→ [Shift] + [¥] キーで入力する

→ 複雑な処理を一行で実行することが可能

・ 「dmesg」コマンドの出力結果をパイプで「less」コマンドに引き渡す例

```
$ dmesg | less
```

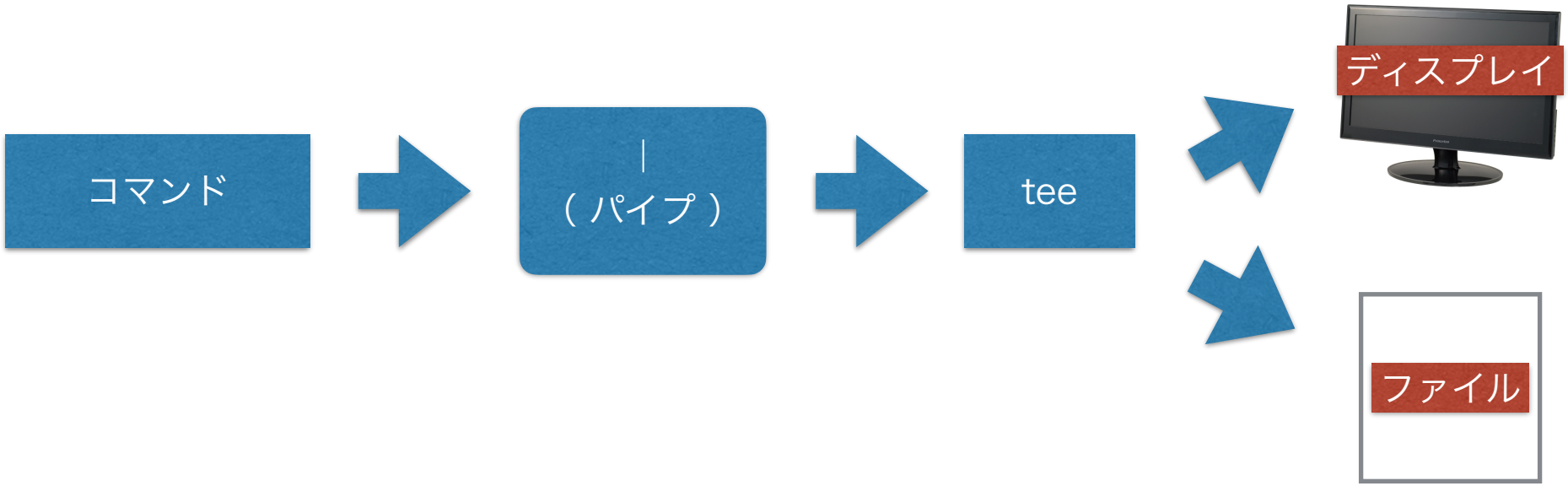
■ 処理結果を画面とファイルに出力する

・ teeコマンド

→ コマンドの処理結果を標準出力としてディスプレイに表示するとともにファイルに出力する

・ コマンド書式

[コマンド] ｜ tee ［ オプション ］［ ファイル名 ］



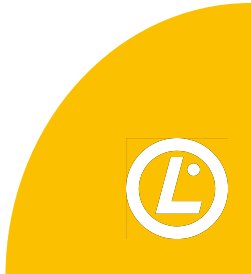
■ 処理結果を画面とファイルに出力する(teeコマンド)

・主なオプション

オプション	説明
-a (—append)	ファイルの内容を上書きせずに保存する

・「ls」コマンドの出力結果をパイプで「tee」コマンドに引き渡す例

```
$ ls -a | tee ls_log.txt
```

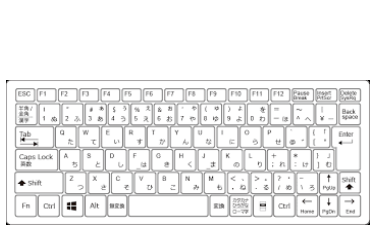


■ リダイレクト

・ リダイレクトとは

→ 標準入力先や標準出力先を変更する記号のこと

→ 「＜」（標準入力）、「＞」（標準出力）、「＞＞」（標準エラー出力）



標準入力

標準エラー出力

標準出力

0

コマンドのプロセス

1

2

ファイル

「＜」
（リダイレクト）

「＞」
（リダイレクト）

ファイル

■ リダイレクト

・主なリダイレクトの書式

リダイレクト書式	説明
コマンド　>　ファイル	コマンドの標準出力をファイルに書き込む（上書き）
コマンド　>>　ファイル	コマンドの標準出力をファイルに追記する
コマンド　<　ファイル	ファイルの内容をコマンドの標準入力へ送る
コマンド　2>　ファイル	コマンドの標準エラー出力をファイルに書き込む（上書き）
コマンド　2>>　ファイル	コマンドの標準エラー出力をファイルに追記する
コマンド　>　ファイル　2>&1	標準エラー出力を標準出力へ設定する（標準出力と標準エラー出力をファイルへ上書きする）
コマンド　>>　ファイル　2>&1	標準エラー出力を標準出力へ設定する（標準出力と標準エラー出力をファイルへ追記する）
コマンド　>　/dev/null　2>&1	標準エラー出力を標準出力へ設定する（コマンドが出力するメッセージを画面に表示しない）

※「/dev/null」は特殊ファイルで入力された全てのメッセージを削除する

■ リダイレクト

・lsコマンドの出力結果をファイル(ls_file.txt)へ上書きする例

```
$ ls  >   ls_file.txt
```

・lsコマンドの出力結果をファイル(ls_file.txt)へ追記する例

```
$ ls  >>   ls_file.txt
```

・ファイルの内容を標準入力として「 wc 」コマンドに引き渡す例

```
$ wc  <   ls_file.txt
```

■ リダイレクト

・ファイルの内容を標準入力として「 wc 」コマンドに引き渡し、結果をファイル(ls_file_output.txt)へ上書きする

```
$ wc  <   ls_file.txt  >  ls_file_output.txt
```

■ catコマンドによるテキストファイルの作成

・ cat コマンド

→ ファイル表示するためのコマンド

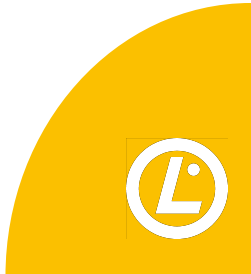
→ リダイレクトと組み合わせると複数行のテキストファイルを作成することも可能

・ コマンド書式

```
cat  <  [ ファイル名 ]
```

・ catコマンドによるテキストファイルの作成例

```
$ cat  <   cat_file.txt
AAA              ← 文字列入力後[ Enter ]キーで改行
BBB
CCC
$                ← [ Ctrl ]  +  [ D ]キーで入力終了
```



■ catコマンドによるテキストファイルの作成

・「 >> 」記号を利用したテキストファイルへの追記例

```
$ cat  >>  cat_file.txt
XXX
YYY
ZZZ
$                    ← [ Ctrl ]  +  [ D ]キーで入力終了
```

・標準エラー出力をテキストファイル(error.txt)へ追記する例

```
$ cat  @@@@  2>  error.txt
$ cat  error.txt
cat: @@@@: そのようなファイルやディレクトリはありません
```

■ catコマンドによるテキストファイルの作成

- echo コマンド
  → 引数に指定した文字列を標準出力としてディスプレイへ表示する
  → リダイレクトと組み合わせると複数行のテキストファイルを作成することも可能

- コマンド書式

```
echo [ 文字列 ]
```

- echoコマンドによる指定した文字列をディスプレイへ標準出力する例

```
$ echo  Hello
Hello
```

■ catコマンドによるテキストファイルの作成

- コマンド書式 (リダイレクトと組み合わせ)

```
echo [ 文字列 ] > [ ファイル ]
```

- echoコマンドによるファイル(echo_file.txt)の作成例

```
$ echo  Hello  >  echo_file.txt
$ cat  echo_file.txt
Hello
```

- echoコマンドによる一行ファイル(echo_file.txt)の追記例

```
$ echo  Good  >>  echo_file.txt
$ cat  echo_file.txt
Hello
Good
```

■ Linuxのテキストエディタ

・ vi エディタ

→ Linuxで標準で実装されているテキストエディタ

→ CUIで動作するテキストエディタ

→ viコマンドで操作する

→ コマンドモードと挿入モードがある

・ viコマンド書式

```
vi [ ファイル名 ]
```

※ファイルを指定しないと新規でファイルを作成できる状態で起動する

・ 新規ファイル名「vifile.txt」を指定してviを起動する例

```
$ vi　vifile.txt
```

■ Linuxのテキストエディタ

・ コマンドモード

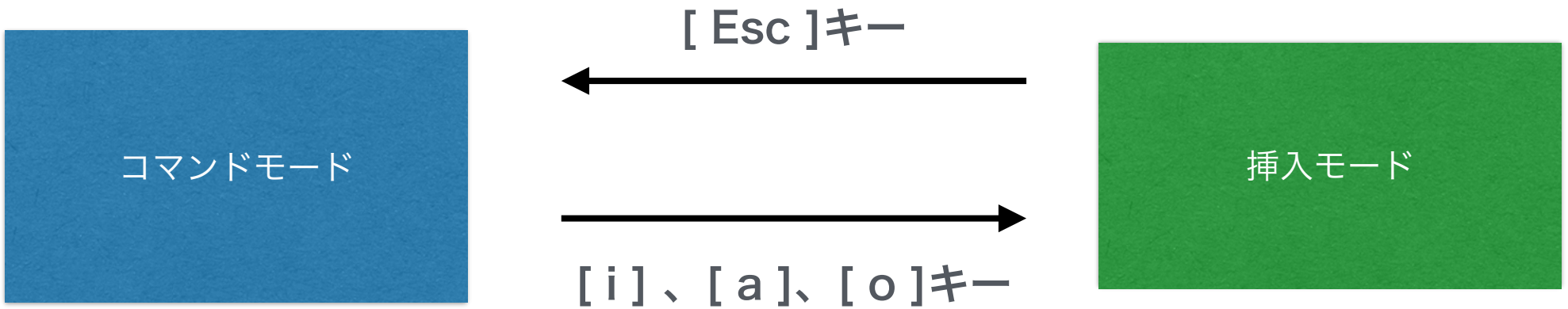
→ カーソルの移動や文字の編集、検索、ファイル保存といったことが行えりモード

→[i] 、[a]、[o]キーのいずれかを押すと挿入モードへ切り替えることができる

・ 挿入モード

→ 文章の入力ができるモード

→[Esc]キーを押すとコマンドモードへ切り替えることができる



■ Linuxのテキストエディタ

・viモードの切り替え

キー操作	説明
i	カーソルの前にテキストを入力する
a	カーソルの後ろにテキストを入力する
A	行末にカーソルを移動し、その直後にテキストを入力する
o	現在の行の下に空白行を挿入してテキストを入力する
O	現在の行の上に空白行を挿入してテキストを入力する
Esc	挿入モードからコマンドモードへ切り替える

■ Linuxのテキストエディタ

・viコマンドのカーソル移動

キー操作	説明
h	1文字左に移動
l	1文字右に移動
k	1行上に移動
j	1行下に移動
0(ゼロ)	現在の行の先頭に移動
$	現在の行の末尾に移動
H	画面の一番上の行頭に移動する
L	画面の一番下の行頭に移動する
gg	ファイルの先頭行に移動する
G	ファイルの最終行に移動する
nG	ファイルのn 行目に移動する（「:n」でも同様の操作が可能）

※コマンドの前に数値を入力することにより、指定した回数分コマンドが繰り返し実行される

■ Linuxのテキストエディタ

・文字の編集操作

キー操作	説明
x	カーソルの位置の文字を削除
X	カーソルの位置の手前の文字を削除
dd	現在の行を削除（切り取り）
D	カーソル位置より右側の文字を削除する
dw	カーソル位置から次の単語までを削除する
y	カーソル位置の文字をバッファにコピーする
yy	現在行をバッファにコピーする（「Y」でも同様の動作をする）
p	現在行の下にバッファの内容を貼り付ける
P	現在行の上にバッファの内容を貼り付ける
U	編集した操作を元に戻す（ando操作を行う）（「ZZ」でも可能）
[Ctrl] + r	ando操作を取り消す

※「x」「X」「dd」などのコマンドは、複数の文字または複数の行をまとめて指定可能

■ Linuxのテキストエディタ

- 主な検索操作

キー操作	説明
/[パターン]	カーソル位置から後方に向かって指定したパターンを検索する
? [パターン]	カーソル位置から前方に向かって指定したパターンを検索する
n	次に一致するパターンを検索する
N	次に一致するパターンを検索する（逆方向）
:noh	パターンに一致した候補の検索マーカーを解除する
:%s/A/B/	最初に見つかった文字列A を文字列B に置換する
:%s/A/B/g	すべての文字列A を文字列B に置換する

- 主な設定変更操作

キー操作	説明
:set nu	行番号を表示する
:set none	行番号を非表示にする
:set ts=タブ幅	タブ幅を数値で指定する

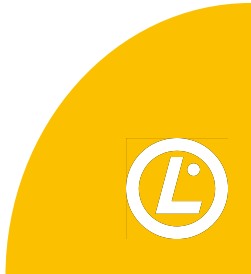
■ Linuxのテキストエディタ

・vi終了、ファイルの保存操作

キー操作	説明
:q	ファイルに保存せずに終了する
:q!	編集中の内容をファイルに保存せずに強制終了する
:wq	編集中の内容を保存して終了する（ZZ（大文字）でも同じ操作が可能）
:w	編集中の内容をファイルに上書き保存する
:e!	最後に保存した内容に戻す
:r [ファイル名]	ファイルの内容を現在行以降に読み込む

・シェルコマンドの実行操作

キー操作	説明
:!コマンド	viを終了せずにシェルコマンドを実行する
:r!コマンド	指定したシェルコマンドの実行結果を挿入する



■ 文字を置換する

- tr (Translate)
  → 標準入力から読み込んだ内容を置換するコマンド
  → 引数を指定することはできない

- コマンド書式 (リダイレクトと組み合わせ)

```
tr [ オプション ] [ 文字列1 ] [ 文字列1 ] < [ ファイル名 ]
```

- 主なオプション

オプション	説明
-d（—delete）	一致した文字列を削除する
-c（--complement）	一致した文字列以外を変換対象とする
-s（—squeeze-repeaats）	連続するパターン文字列を1 文字に圧縮する

■ 文字を置換する（trコマンド）

・対応する文字を置換する例1

```
$ tr  abcd  ABCD  <  tr_file.txt
```

・英小文字を英大文字に変換する例2

```
$ tr  a-z  A-Z  <  tr_file.txt
```

・異なる文字数を指定した置換の例

```
$ tr  ab  ABCD  <  tr_file.txt          ←  置換前の文字列が短い場合
→  tr  ab  AB  <  tr_file.txt                 余った分は無視される

$ tr  abcd  AB  <  tr_file.txt          ←  置換後の文字列が短い場合
→  tr  abcd  ABBB  <  tr_file.txt                余った分は無視される
```

■ 文字を置換する（trコマンド）

・空白文字を指定する場合

```
$ tr a-c " "  <  tr_file.txt
```

・空白を削除する場合

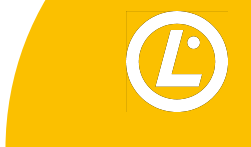
```
$ tr  -d " "  <  tr_file.txt
```

・数字（0〜9）以外をピリオド「.」に置換する例

```
$ tr  –c 0-9  .  <  tr_file.txt
```

・連続する空白（スペース）を一つの空白に置換する

```
$ tr -s  " "  <  tr_file.txt
```



■ 文字を置換する（trコマンド）

- ファイル内のスペースをタブに置換する例

```
$ tr " " "\11 " < tr_file.txt
```

- ファイル内の連続する複数のスペースを1 つのタブに置換する例

```
$ tr -s " " "\11" < tr_file.txt
```

- 文字クラス

  → 文字クラスと呼ばれる特殊な記述を使用して、文字列を指定することも可能

文字クラス	説明
[:alpha:]	英字
[:lower:]	英小文字
[:upper:]	英大文字
[:digit:]	数字
[:alnum:]	英数字
[:space:]	空白文字（スペース）

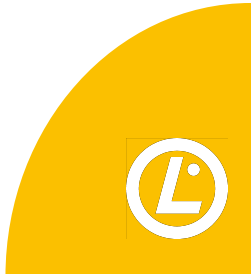
■ 文字を置換する（trコマンド）

- ファイル内の英小文字を英大文字に置換する例

```
$ tr [:lower:] [:upper:]  <   tr_file.txt
```

- ファイル内の空白文字（タブ、空白行含む）を削除する例

```
$ tr –d [:space:] <   tr_file.txt
```



■　テキストファイルを分割する

・ cut コマンド

　→ テキストファイルを指定した区切り文字で分割し、特定のフィールドを表示する

・ コマンド書式

```
cut ［ オプション ］［ ファイル名 ］
```

・ 主なオプション

オプション	説明
-c 文字数	抽出する文字位置を指定する
-d 区切り文字 （--delimiter=[区切り文字]）	フィールドの区切り文字（デリミタ）を指定する （デフォルトでは「タブ」が指定されます）
-f フィールド番号 （--fields=[フィールド番号]）	抽出するフィールド番号を指定する

■　テキストファイルを分割する（ cutコマンド ）

- 「cut」コマンドを使ってファイルから指定したフィールドを抽出する例1

```
$ cut –d: -f 6 /etc/passwd
/root
/bin
/sbin
/var/adm
～途中省略～
/var/empty/sshd
/
/home/ken-nw
```

■ テキストファイルを分割する（ cutコマンド ）

- 「cut」コマンドを使ってファイルから指定したフィールドを抽出する例2

```
$ cut –d: -f 1,3 /etc/group
root:0
bin:1
daemon:2
～途中省略～
sshd:74
tcpdump:72
ken-nw:1000
```

ご清聴ありがとうございました