

ファイルシステム関係のFAQ.

• [ファイルシステム関係のFAQ.](#)

- [Linux のファイルシステム構造について知りたい](#)
- [bin と sbin はなにが違うの？](#)
- [/と /usr でインストールされるファイルが分けられているみたいなのはなぜ？](#)
- [/etc と /var は何が違うの？](#)
- [なんでこんなディレクトリ名になったの？](#)
- [/opt はなんのためにあるの？](#)
- [ホームディレクトリにある Desktop というディレクトリは何？](#)
- [Windowsの「ドライブ」がないけど、CD とかを入れたらどうすればいいの？](#)
- [起動スクリプトの配置はどうなってるの？](#)
- [ディストリビューションをインストールした時の記録がどこかにありますか？](#)
- [ファイルやディレクトリを検索したい。\(find, locate\)](#)
 - [検索したファイルにコマンドを適用する](#)
- [ファイルの種類を調べたい。\(file\)](#)
- [ファイル名が「-」ではじまるファイルを消したい](#)
- [データをより確実に消去したい\(shred\)](#)
- [大事なファイルを消してしまったので復元したい。](#)
- [Linuxはデフラグしなくても大丈夫なのですか？](#)

• [ファイル/ディスク操作のコマンド](#)

- [ファイルのリストを表示: ls](#)
- [ディレクトリを移動: cd](#)
- [ファイルをコピー: cp](#)
- [ファイルシステムを検索: find](#)
- [ファイルを移動: mv](#)
- [ディレクトリを作成: mkdir](#)
- [ファイルを削除: rm](#)
- [ファイルを展開: tar](#)
- [ハードディスクの使用状況を表示: df](#)

Linux のファイルシステム構造について知りたい

基本は以下。(/ ディレクトリ配下)

ディレクトリ	解説
/	ルートディレクトリ
ユーザー別	
/home	ユーザーのホームディレクトリ
/root	root のホームディレクトリ
ソフトウェア	
/bin	汎用コマンド
/sbin	管理者用コマンド (root専用)
/usr	アプリケーションなどのパッケージ
管理ファイル	
/etc	設定ファイル
/var	データファイル

/tmp	一時ファイル
デバイス、ハードウェア	
/dev	デバイスファイル
/mnt	デバイスをマウントするディレクトリ
/proc	システム情報
その他	
/lib	共有ライブラリ
/boot	OSのブート用ファイル

/usr の下は更に以下のようになっています。

ディレクトリ 解説

/usr/bin	アプリケーションによる汎用コマンド
/usr/sbin	アプリケーションによる管理者用コマンド (root専用)
/usr/lib	アプリケーションのライブラリ
/usr/share	アーキテクチャに依存しないファイル (ドキュメントなど)
/usr/include	C言語用ヘッダファイル
/usr/local	アプリケーションをソースからコンパイルして入れるなど、管理者が自由に使える領域

ホームディレクトリにはユーザーの個人ファイルの他、ユーザー固有の設定ファイルやデータファイルがドットファイルとして置かれています。

このようになっている理由は、パーティションを分ける際にディレクトリの用途によって別々の領域に分けやすくするためです。(例えば /home とその他のディレクトリを分けておくと、他のパーティションが障害にあっても /home は被害にあわずに済む)

ディストリビューションによっては /usr とは別に /opt があつたり、/mnt がなかつたり、/boot が /initrd だつたりもします。

現在の Linux のファイルシステムは大体 [Filesystem Hierarchy Standard](#) に準拠されてることが多いので、それを読めば分かるかも。

以下は FHS の概要と解説。

- http://linuxjm.sourceforge.jp/html/LDP_man-pages/man7/hier.7.html

bin と sbin はなにが違うの？

/ や /usr にある bin や sbin ディレクトリには、ユーザーやシステムから使われるバイナリ/実行ファイルを置きます。

どんなユーザーからも使われる可能性がある汎用的なプログラムは bin に、システム管理者しか使わないような運用系のプログラムは sbin に置きます。

/ と /usr でインストールされるファイルが分けられているみたいなのはなぜ？

/ にはシステムの動作に最低限必要なファイルとプログラム、/usr にはそれ以外のファイルとプログラムを置きます。

/etc と /var は何が違うの？

どちらもプログラムから使用される設定・データファイルを置きます。ただし、一度設定したら変わらないようなファイルは /etc に置き (*1)、/var は使っているたびにサイズが変化するファイ

ル (*2) を置きます。

なんでこんなディレクトリ名になったの？

以下を参照。

- http://www.gnu.org/software/hurd/hurd/faq/slash_usr_symlink.html
- <http://d.hatena.ne.jp/ytakano/20100715/1279219401>

/opt はなんのためにあるの？

サポート元が違うとか、サイズが大きすぎるとかの理由で /usr から外されたファイルを置く。

/opt の下はパッケージ別に分かれていて、/usr/bin は /opt/kde/bin とかに当たります。

ホームディレクトリにある Desktop というディレクトリは何？

KDE や GNOME を使っている場合、個人のデスクトップ画面に表示されるディレクトリ。

Windowsの「ドライブ」がないけど、CD とかを入れたらどうすればいいの？

ドライブがない代わりに、/mnt や /media ディレクトリにメディア名でマウントされる。 (*3)

自動でマウントされない場合は、手動で /dev からマウント。

起動スクリプトの配置はどうなってるの？

[ServersGeneralFAQs#init](#)に移動しました。

ディストリビューションをインストールした時の記録がどこかにありませんか？

大抵のディストリビューションは、インストール時の記録が後で参照できるように用意されていますが、ディストリビューションによっては記録が保存されている場所が異なっていることがあります。

- Vine Linux

完全な記録が /tmp/install.log にあり、選択結果 (kickstart) が /root/anacocnda-ks.cfg に保存されています。

- Fedora Core 1

どちらの記録も /root ディレクトリに保存されます。

ファイルやディレクトリを検索したい、(find, locate)

ファイルやディレクトリを検索するには find , locate が使える。

検索	コマンド
find (逐次検索)	\$ find 検索するディレクトリ -name ファイル名
locate (データベースから検索)	\$ locate ファイル名

find では逐一ファイルを検索するのに対し、locate ではファイル名のデータベースから検索するのでやや高速。

ただし、locate ではデータベース更新後に作成されたファイルやディレクトリは検索できません。データベースを再更新するには root で `updatedb` コマンドを実行します。大抵の場合、updatedb は cron デーモンによって 自動で定期的に行われる 設定になっています。

- [ファイルやディレクトリを詳細な条件で探すには \(@IT\)](#)
- [ファイルやディレクトリを素早く探すには \(@IT\)](#)
- [ファイルを検索する \(find\)](#)
- [ファイルを検索する \(locate, which,...\)](#)

検索したファイルにコマンドを適用する

findコマンドの-execアクションを使用します。

例: カレントディレクトリの複数のzipファイルを一度にまとめて展開する。

```
$ find . -name '*.zip' -exec unzip {} \;
```

ファイルの種類を調べたい, (file)

あるファイルがどんな種類のファイルかわからない時は less などのページャでとりあえず中身を覗いてみるのも一つの手だけど、file コマンドを使うとマジックナンバーに基づいてファイルの種類を調べることができます。

```
$ file sample.gif
sample.gif: GIF image data, version 89a, 302 x 231
```

ファイル名が「-」ではじまるファイルを消したい

-fのようにファイル名が-ではじまるファイルを作ってしまうと、コマンドラインから `rm -f` で消そうとしたときに -f がオプション指定だと解釈されてしまい、rm コマンドで削除することができません。

こういった場合は `rm ./-f` として引数に与える文字の最初に「-」以外を指定するか、`rm -- -f` としてオプションとして解釈させない(参照: [man rm](#)) ことで削除します。

- [rmで消せないファイル名を作ってしまった](#)
- http://linuxjm.sourceforge.jp/html/GNU_fileutils/man1/rm.1.html#lbAD

データをより確実に消去したい (shred)

shred コマンドを使いましょう。データの削除やシステムファイルの削除をより確実に実行して、その復元を困難にします。

```
shred          コマンド
ファイルを確実に消去 $ shred --remove ファイル
```

メディアやドライブを廃棄したり、他人に譲渡する際活用できます。

- [ファイルを完全に消去するには\(@IT\)](#)

大事なファイルを消してしまったので復元したい。

基本的に復元できません。大事なファイルは普段からどこかにバックアップをとっておこう。

ファイルシステムに ext2 を使っている場合に限り、[mc](#) で復元できるかも。[うっかりと削除してしまったファイルを復活させたい](#) (ITMedia) を参照のこと。

rm コマンドでは -i オプションを付けると削除するかどうか尋ねるようになるので、rm を使う場合は予防的にこのオプションを使うのもいいかもしれません。

- [Linux Ext2fs Undeletion mini-HOWTO](#)

次のような状況を想像してほしい。ここ 3 日間、飲まず食わずでシャワーも浴びずに作業を続け、ようやくその抑えきれないハッキングの衝動が実を結んだ。ついに、プログラムが完成したのだ。世界的な賞賛と名声をもたらすだろうほどの。あとは tar でかためて、Metalab にあげるだけだ。おっと、Emacs のバックアップファイルの削除を忘れていた。ここであなたは、rm * ~ とコマンドを打つ。そして、嗚呼、コマンドに余計なスペースを入れてしまったことに気付くのである。世紀の逸品を削除してしまった！しかし、万策尽きたわけではない。この文書は、Ext2 ファイルシステム上で削除してしまったファイルを復旧する方法について解説している。おそらく、最後にあなたはそのプログラムをリリースできるはず
....

Linuxはデフラグしなくても大丈夫なのですか？

Linux で使われているファイルシステムはひとつのファイルに対し連続したブロックを取るの
で、Windows に比べて断片化しにくいといわれています(一部では「断片化しない」と盲目的に語
られているのも事実ではありますが)。

なので、デフラグはしなくても構いません。

- [JF: Linux Partition HOWTO ファイルシステムとフラグメンテーションに関するいくつかの事実](#)

MS-DOS ファイルシステムはディスクスペースの異常な管理でよく知られてい
ます。MS-DOS のバッファキャッシュの使い方がひどく悪いことと相まってパ
フォーマンスに及ぼすファイルフラグメンテーションの影響は際だっています。
DOS ユーザー達はほとんど毎週ディスクをデフラグすることに慣れていて、デ
フラグすることに関して儀式的な信仰に達している人すらいます。このような習
慣は Linux と ext2 に持ちこむべきではありません。Linux のネイティブなファイ
ルシステムは普通の使い方ではデフラグを必要としません。ディスク上の空き領
域がすくなくとも 5% ある状況なら「普通の使い方」の範疇です。

ファイル/ディスク操作のコマンド

ファイルのリストを表示: ls

カレントディレクトリにあるファイルとディレクトリ一覧を見る。

```
$ ls
```

[ドットファイル](#)まで見たい時は -a オプション、ファイルの詳細を表示するには -l オプションを
つける。

```
$ ls -al
```

ディレクトリ名を指定するとカレントディレクトリ以外を表示する。

\$ ls [表示するディレクトリ名]

ディレクトリを移動: cd

指定ディレクトリに移動する。(カレントディレクトリを変更する)

\$ cd [ディレクトリ名]

[zsh](#)で `setopt auto_cd` を設定すると、ディレクトリを打つだけでcdすることができるので、激しく便利。

ファイルをコピー: cp

ファイルをコピーする。

\$ cp [コピー元のファイル、ディレクトリ名] [コピー先のファイル、ディレクトリ名]

ディレクトリごとコピーする時は `-r` オプション。

ファイルシステムを検索: find

ファイルを検索する

\$ find [探すディレクトリ] -name [名前]

いろんなオプションが使える上、ディレクトリツリーを調べるのにも使えて便利。

ファイルを移動: mv

ファイルを移動する。名前の変更にも使える。

\$ mv [現ファイル、ディレクトリ名] [新ファイル、ディレクトリ名]

ディレクトリを作成: mkdir

新しいディレクトリを作る。

\$ mkdir [ディレクトリ名]

一気に深いディレクトリを作る時は `-p` オプション。

ファイルを削除: rm

ファイル、ディレクトリを削除する。

\$ rm [ファイル、ディレクトリ名]

ディレクトリ以下をまるごと削除するは `-r` オプション。

ファイルを展開: tar

tar.gz または tar.bz2 形式で圧縮されたファイルの展開。

```
$ tar xvf [圧縮されたファイル]
```

ハードディスクの使用状況を表示: df

ハードディスクの残り容量(=使用状況)を表示。

```
$ df
Filesystem          1K-ブロック   使用   使用可  使用% マウント位置
/dev/hda3            14160404    2292192 11148904  18% /
/dev/hda2             101105      6340    89544    7% /boot
none                 123728       0    123728    0% /dev/shm
```

デフォルトだとブロック数で表示するためわかりにくいので、-h オプションをつけるとよいでしょう。-h は単位を MB や GB で表示させるオプションです。

```
$ df -h
Filesystem          サイズ  使用  残り  使用% マウント位置
/dev/hda3           14G   2.2G   11G   18% /
/dev/hda2            99M   6.2M   88M    7% /boot
none                121M     0   121M    0% /dev/shm
```

-T オプションをつけるとファイルシステムの種類(ext2, ext3, Reiserfs, ...etc)を併せて表示します。