



# LPIC レベル1 技術解説無料セミナー

主催：特定非営利活動法人LPI-Japan

講師：宮原 徹 ((株)びぎねっと)

# 講師プロフィール

---

- 株式会社びぎねっと 代表取締役社長兼CEO
  - 日本仮想化技術株式会社 代表取締役社長兼CEOでもある
- Linux・オープンソースに関するIT技術者教育を中心にビジネスを展開
- 現在は仮想化技術に関するソリューション提案を行っている (VMware・Xenなど)
- LPI-Japan発行 メールマガジン 執筆者

# 本日のアジェンダ

---

1. LPIC (Linux技術者認定試験) の概要
2. ポイント解説
3. Q&A

LPIの概要を理解し、Linuxの学習を  
スタートするためのポイントを掴む

---

# LPIC (Linux技術者認定試験) の概要

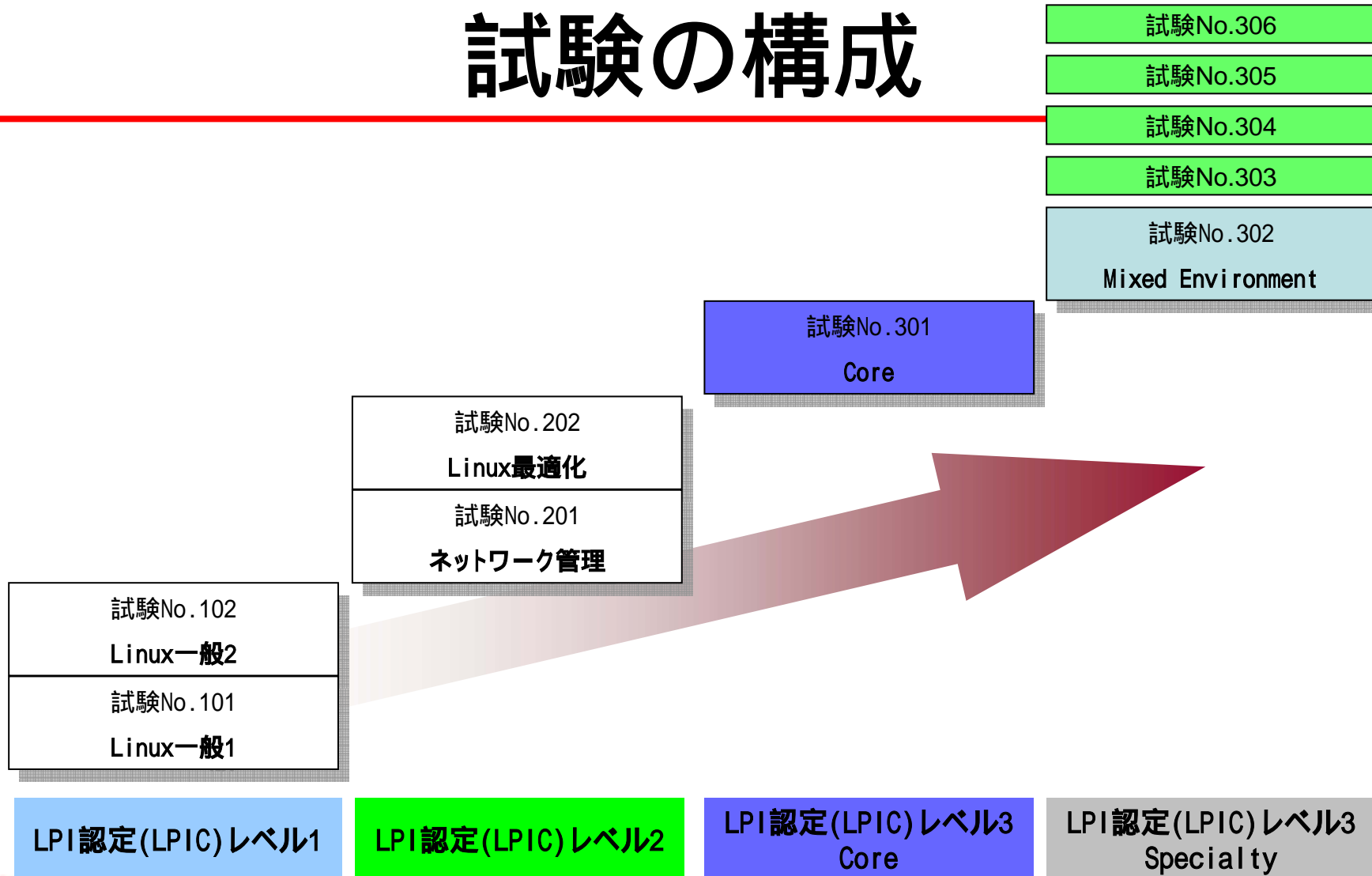


# LPICの特長

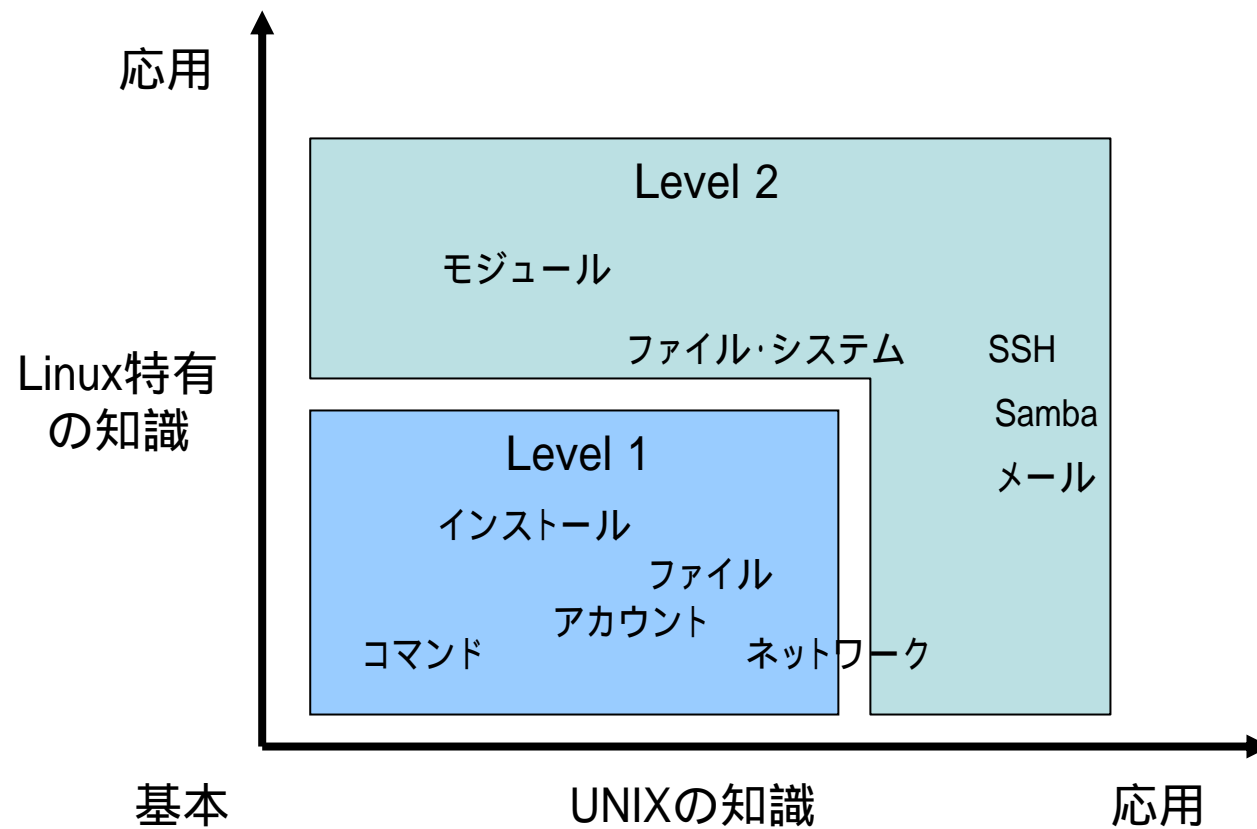
---

- オープンソース
  - 世界的なコミュニティで形成
- ベンダーニュートラル
  - 様々な環境で知識が活かせる
- 本質的な問題
  - 技術的な本質を見極める問題
- 広範囲に渡る出題
  - 技術レベルの再認識

# 試験の構成



# 試験のカバーしている範囲



---

# LPICレベル1 試験出題範囲と 対策



# 101試験の出題範囲

---

主題101: ハードウェアとアーキテクチャ

主題102: Linuxのインストールとパッケージ管理

主題103: GNUとUnixのコマンド

主題104: デバイス、Linuxファイルシステム、ファイルシステム階層標準

主題110: X Window system

# 102試験の出題範囲

---

主題105: カーネル

主題106: ブート、初期化、シャットダウン、ランレベル

主題107: 印刷

主題108: ドキュメンテーション

主題109: シェル、スクリプト、プログラミング、コンパイル

主題111: 管理業務

主題112: ネットワークの基礎

主題113: ネットワークサービス

主題114: セキュリティ

# 出題範囲 バージョン2.0

---

- 2008年12月から改訂予定
- レベル1、レベル2について変更予定
  - サーバー構築についてはレベル2に移動
- レベル1については、現状をベースに学習しても大きな問題はない？
  - レベル2まで学習しないとあまり意味がない

# 全体的な傾向と対策

---

- 出題範囲に幅がある
  - 知らないポイントを無くす
  - 得意なポイントを作る
- うろ覚え、ケアレスミスを減らしたい
  - コマンドのオプションの意味をしっかりと
  - 出題範囲詳細に出てくるファイルやコマンドはmanコマンドなどで調べておく

# 基本Linux学習でカバーされる範囲

---

- 主題102: Linuxのインストールとパッケージ管理
- 主題103: GNUとUnixのコマンド
- 主題104: デバイス、Linuxファイルシステム、ファイルシステム階層標準
- 主題106: ブート、初期化、シャットダウン、ランレベル
- 主題108: ドキュメンテーション
- 主題111: 管理業務
- 主題112: ネットワークの基礎
- 主題113: ネットワークサービス

Linux+インターネットサービスの  
学習でこの辺りまでカバー

# 基本Linux学習でカバーされない範囲

---

主題101: ハードウェアとアーキテクチャ

主題105: カーネル

主題107: 印刷

主題109: シェル、スクリプト、プログラミング、コンパイル

主題114: セキュリティ

このあたりについては追加で  
周辺情報を集める必要がある

# 学習の方法

---

- 出題範囲をしっかりと把握
  - 関連キーワードはすべて調べる
- 基礎的なLinuxの操作方法を学習
  - インストールからコマンド操作、Linuxシステム管理基礎レベル(ユーザ管理等)
  - カバーされない範囲については、別途周辺情報で知識を補う(PC自作なども効果的)

# 学習用ディストリビューション

---

- 今使える環境があるならまずはその環境で学習
  - 基本的にディストリビューションに依存しない
- 最終的な知識の確認は2.4ベースのディストリビューションも含めて
  - 試験内容がLinuxカーネルバージョンが2.4をベースにしている部分もある
  - **個人的には**Red Hat Linux 7.3がオススメ(未だに現役で動いてたりする場合も?)

---

# ポイント解説



# 今回解説するポイント

---

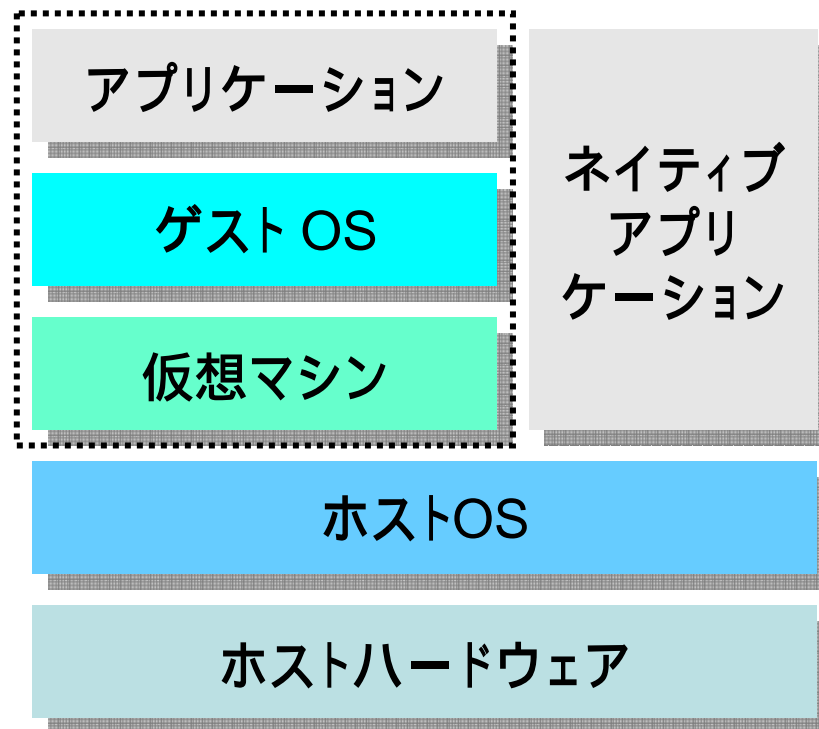
- Linuxのインストール(仮想マシンを使って)
- 1.103.1 コマンド行で操作する
- 1.103.3 基本的なファイル操作を行う
- 1.103.4 ストリーム、パイプ、リダイレクトを使う

# Linuxの学習環境を作る

---

- 理想を言えば、PCを2台～3台用意
  - たとえばWindowsクライアントにLinuxサーバ
- HDD交換可能にする
  - 色々なディストリビューションを試せる
  - 再インストールも簡単
- 物理的な問題から仮想マシンで代替
  - LinuxやWindowsで無料で使えるものとしてVMware ServerやVirtualBoxなど
  - MacならParallelsやVMware Fusion、VirtualBoxなど

# 仮想マシンとは



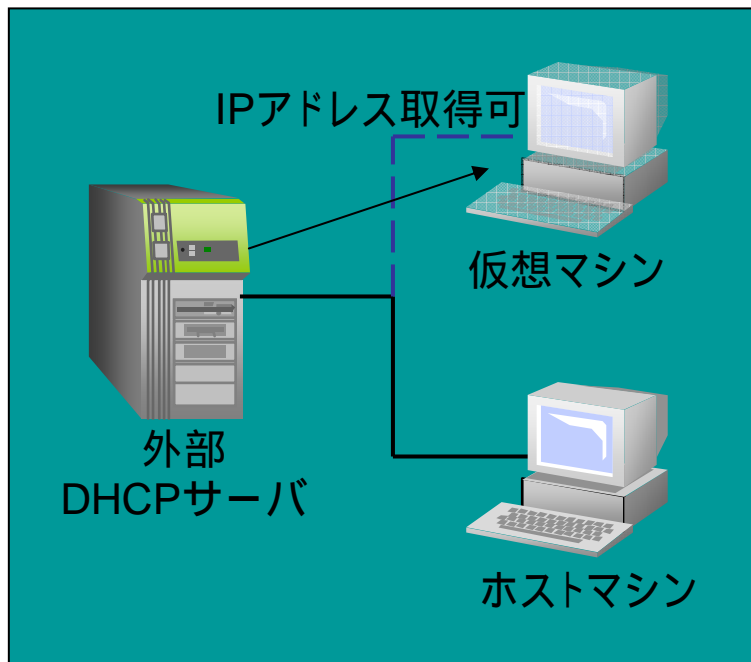
- ソフトウェアでもう1台のPCを再現する技術
- ホストOSにはLinuxやWindowsを使用
- 仮想マシン内で別のOSを実行可能
- HDDやメモリの許す限り、複数の仮想マシンを実行可能

# 仮想マシン利用のポイント

---

- メモリは実際のPC同様に消費
  - ホストマシンにできるだけ沢山のメモリを搭載
- 仮想ハードディスクを使用する
  - ホストOS上のファイル = 仮想マシンのHDD
- ゲストOSのインストール元はISOイメージも可能
  - 光学式ドライブはホストOSと共用
  - ISOイメージからインストールも可能
- 仮想ネットワークの設定は用途に応じて
  - ブリッジ: 外部から接続可能・DHCP設定の場合にはDHCPサーバが必要
  - NAT: 外部から接続不可・DHCP設定の場合には仮想マシンソフトがDHCPサーバを準備

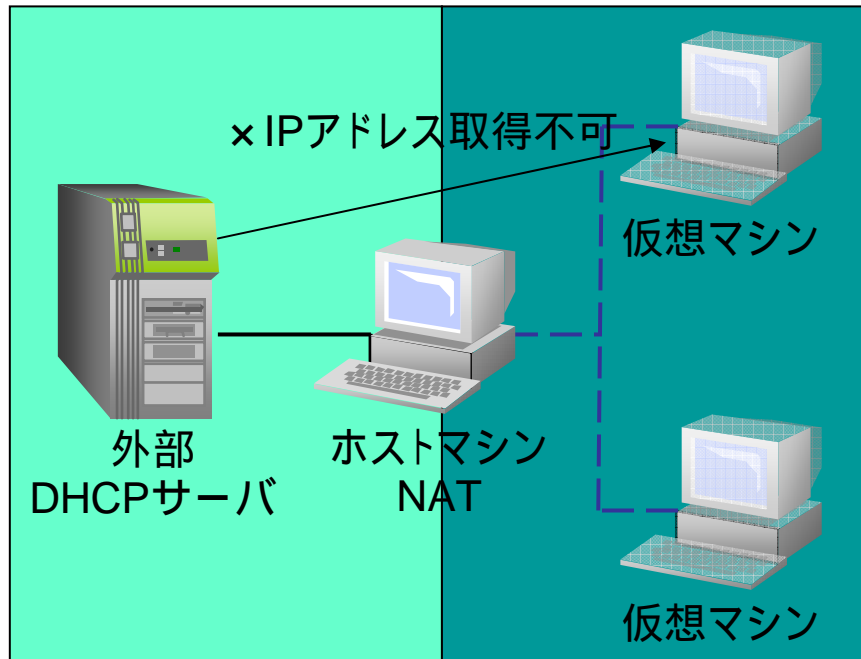
# ブリッジ



同一セグメント

- ホストマシンが接続されているネットワークとブリッジ接続
  - 外部とはLayer2接続
- 外部DHCPサーバからIPアドレス取得可能
- 固定IPアドレスならば物理ネットワークのIPに合わせる

# NAT



別セグメント

- ホストマシンが接続されているネットワークとNAT接続
  - 外部とはLayer3接続
- 外部DHCPサーバからIPアドレス取得不可
  - ホストマシンでDHCPサーバ稼働

## 1.102.1 ハードディスクの配置を設計する

---

### 説明

- Linuxシステムにおけるディスクパーティションの構成を設計する。
- ファイルシステムないしスワップスペースを異なるディスクやパーティションに割り当てる
- システムを適切に使用できるように設計を調整する
- ブートに当たってBIOSの制限を満たすように/bootパーティションを配置する

### 重要なファイル、用語、ユーティリティ

- / (root)ファイルシステム
- /varファイルシステム
- /homeファイルシステム
- スワップスペース
- マウントポイント
- パーティション

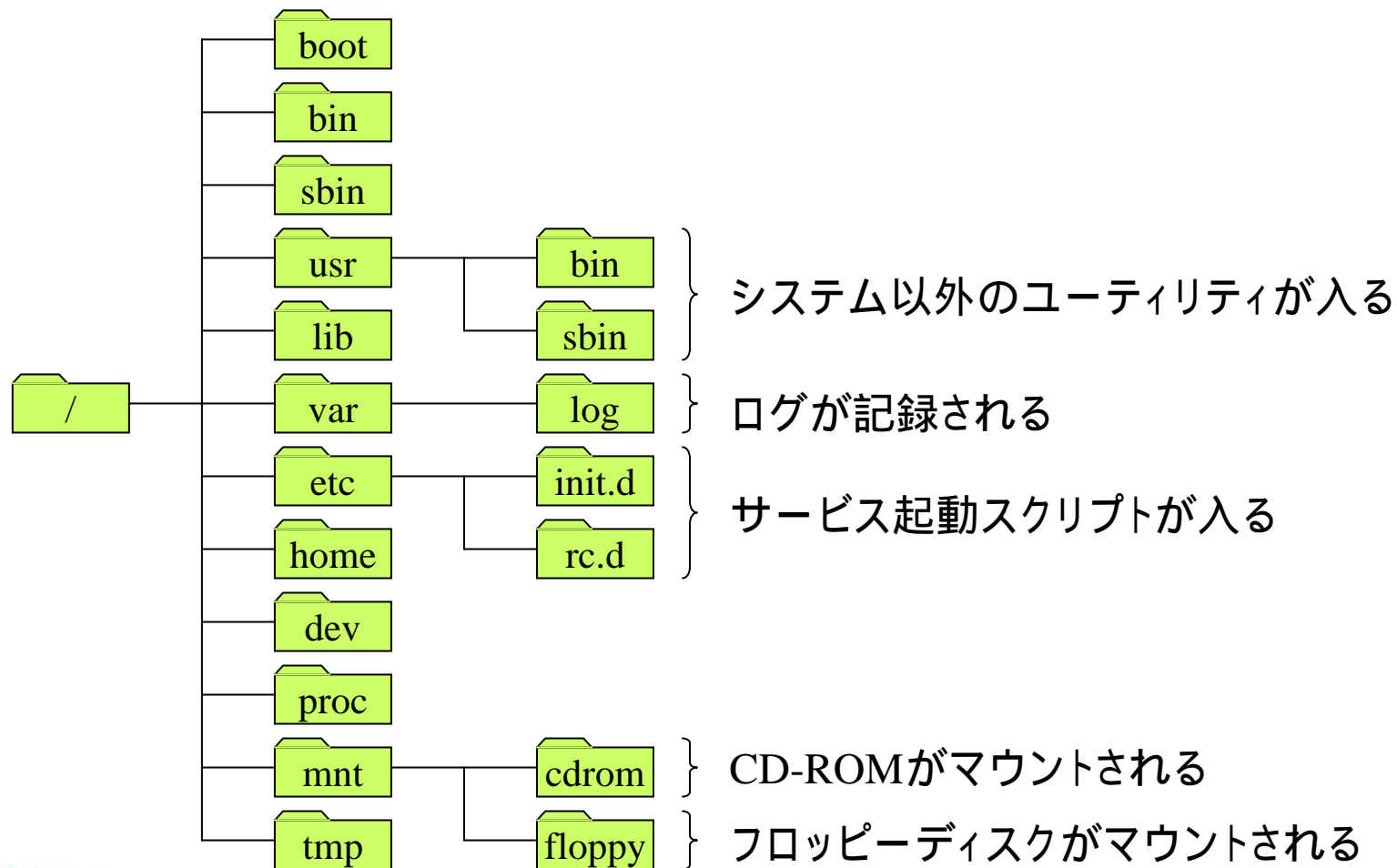
# Linuxのディレクトリ構造

---

## 主なディレクトリ

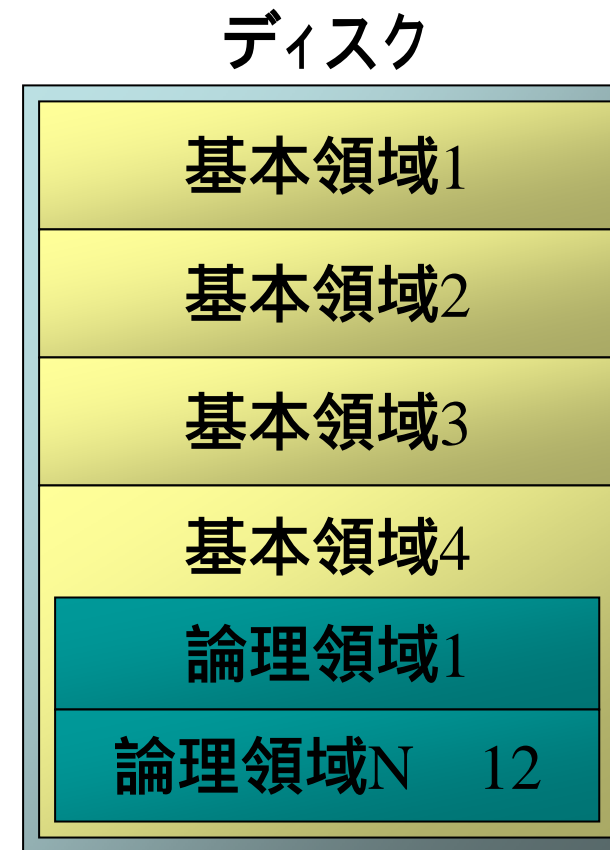
- /boot Linuxカーネルと起動関連ファイルを格納
- /bin Linuxの最も基本的なコマンドを格納
- /sbin Linuxのシステム管理に関するコマンドを格納
- /usr Linux上のアプリケーションソフトのファイルを格納
- /lib 各種ライブラリファイルを格納
- /var ログやメールなどシステムが一時的にデータを格納
- /etc 各種設定ファイルを格納
- /home 一般ユーザーのホームディレクトリ
- /dev デバイスファイルの入っている特殊なディレクトリ
- /proc システムの各種稼動状態をファイルとして保持
- /mnt フロッピーディスクやCD-ROMをマウントする
- /tmp 各種プログラムが利用する一時ディレクトリ

# Linuxディレクトリ構造図



# パーティションとは？

- 1つのディスクを複数の領域に仕切り、別々に使用
- 1つのディスクに4つの基本領域が作成できる
- 基本領域の1つを拡張領域とし、その中に最大12の論理領域を作成できる(上限値無い場合も)
- Windowsは論理領域から起動できない
- Linuxは論理領域から起動できる

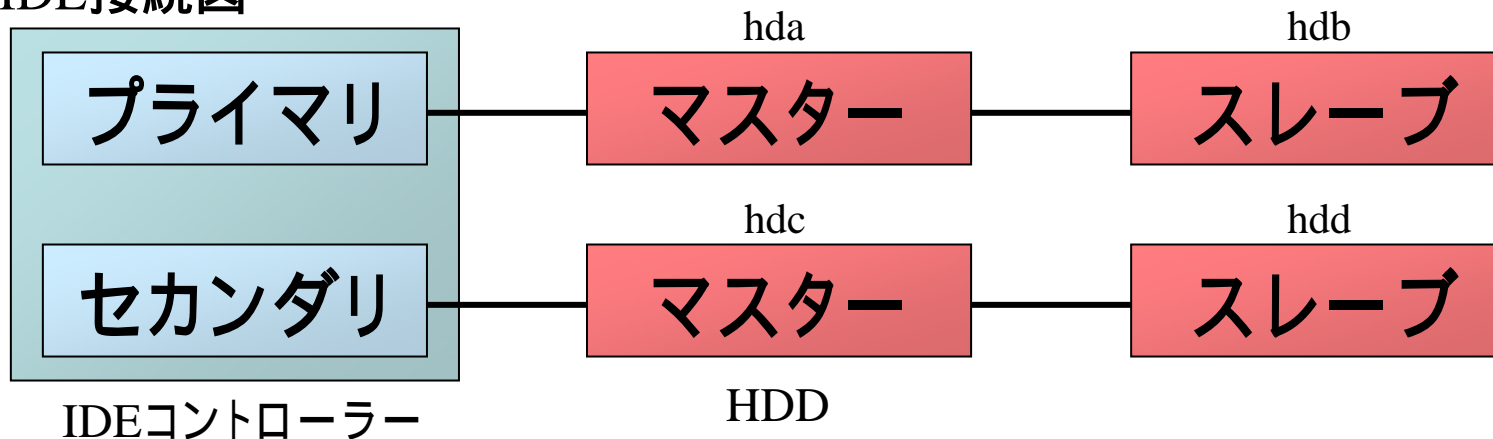


注:基本領域4 拡張領域となる

# HDDとデバイスファイル

- IDE接続HDDのデバイスファイルは、プライマリマスターからセカンダリスレーブまで順に/dev/hda ~ /dev/hddとなる
- SCSI接続HDDのデバイスファイルは、認識順に/dev/sda ~ となる
  - SATA接続HDDはSCSI接続相当になる

IDE接続図



# ファイルシステム構成の検討

---

- /(ルート)ディレクトリ以外を別のディスクやパーティションに分離する
  - /homeを分離して、**ユーザデータ**を分離する
  - /varを分離して、**システムデータ**を分離する
- ディレクトリを分離する理由
  - メンテナンス性の確保(システムとデータの分離)
  - RAID(ミラー)による冗長性の確保
  - パフォーマンスの向上

# 1.103.1 コマンド行で操作する

---

## 説明

- シェルと対話して、コマンド行を使用する。適切なコマンドとコマンド群を入力する、環境変数を参照したりエクスポートしたりする、コマンド履歴と編集機能を活用する、パスに存在する、存在しないにかかわらず、コマンドを起動する、コマンド置換を使用する、ディレクトリツリーに沿ってコマンドを再帰的に実行する、コマンドに関する情報を見つけるためにmanを使用する。

## 重要なファイル、用語、ユーティリティ

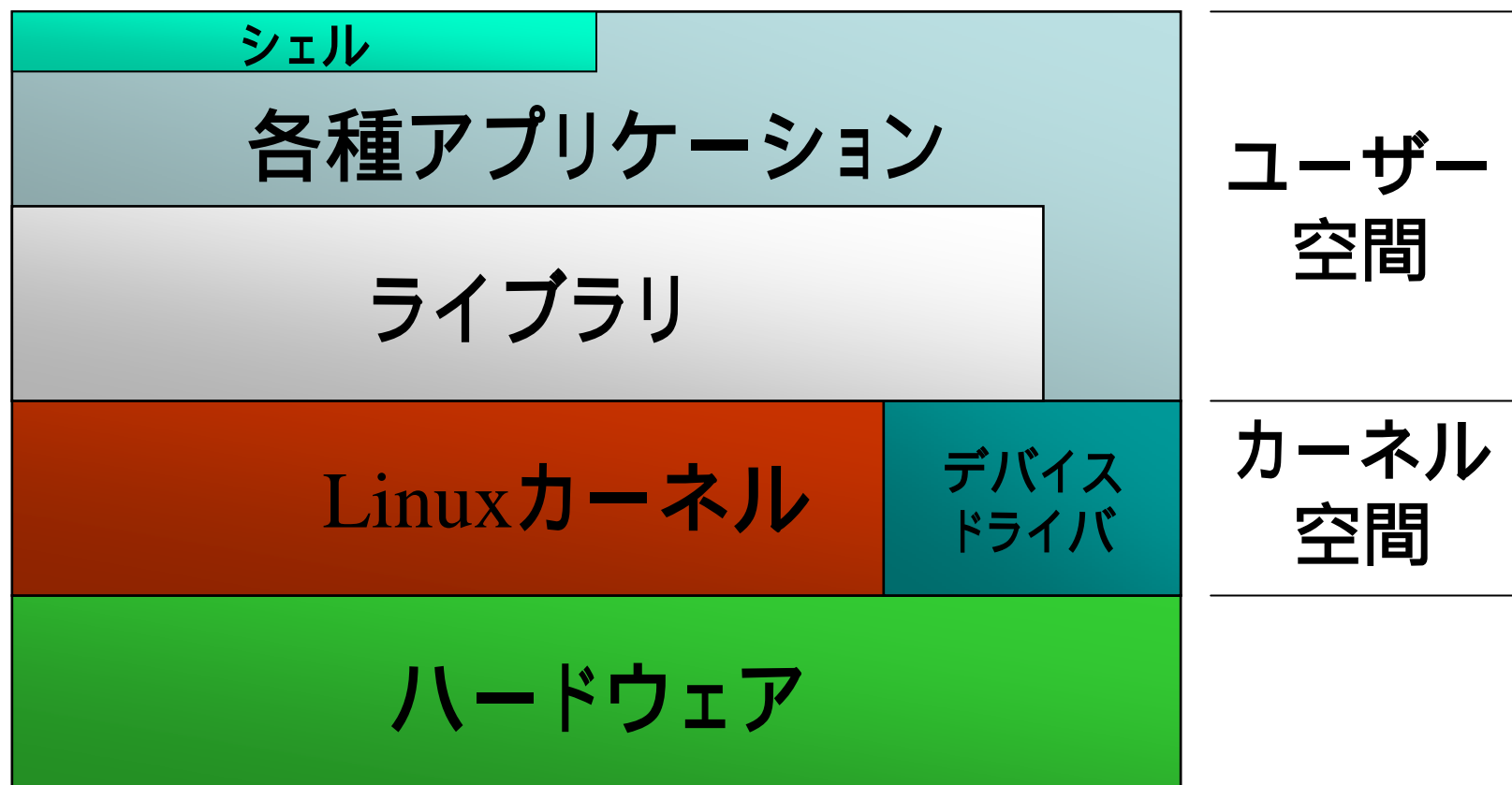
- .
- bash
- echo
- env
- exec
- export
- man
- pwd
- set
- unset
- ~/.bash\_history
- ~/.profile

# シェル

---

- シェルとは、ユーザーの命令をOSに対して伝えるユーザーインターフェースである
- プログラムを実行するには、そのプログラムの名前(コマンド)をシェルプロンプトに入力する
  - ログインシェル
    - ログインすると、画面にはシェルプロンプトが表示される(CUIログインの場合)
  - シェルスクリプト
    - シェルに対する命令をファイルに保存しておき、実行させる

# Linuxの構造



# コマンドオプションと引数

---

- コマンドの動作を細かく設定するためにオプションと引数をコマンドに与えます
  - オプションはコマンドの設定項目を指定します
  - 引数はファイル名などの具体的な値を指定します
- コマンド基本文法
  1. コマンド
  2. コマンド 引数
  3. コマンド 引数1 引数2
  4. コマンド -オプション
  5. コマンド -オプション 値 ← オプションが値を取る時は必ず値が必要
  6. コマンド -オプション 値 引数 ...

# カレントディレクトリ

---

- コマンド実行時にディレクトリを指定しなかった場合、暗黙のうちに指定されるディレクトリ
- 「作業(ワーク)ディレクトリ」とも呼ばれる
- 確認にはpwdコマンドを使用
  - プロンプトにもカレントディレクトリ名が表示される

コマンドラインによる操作上達の近道は  
カレントディレクトリを意識すること

# ホームディレクトリ

---

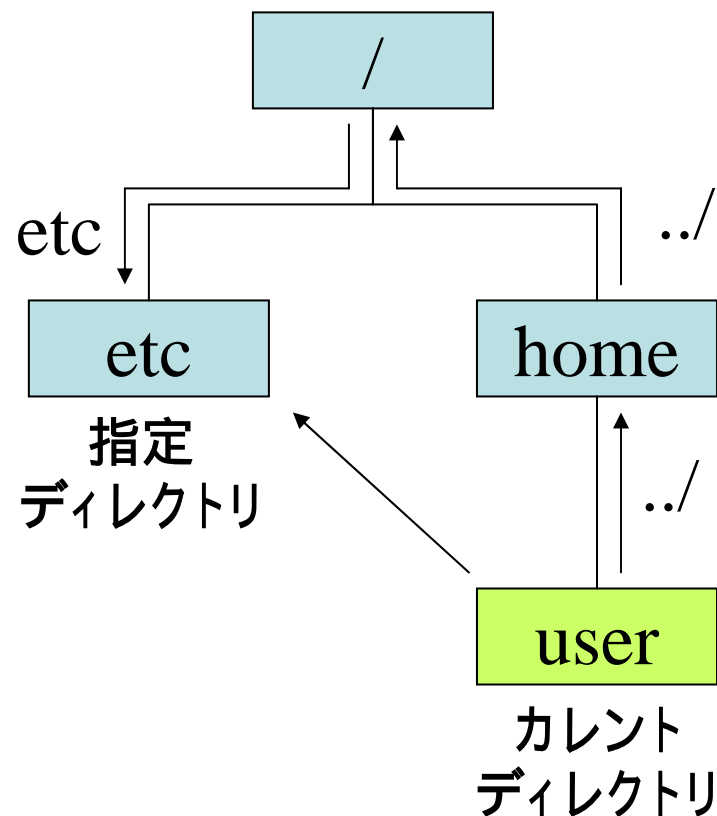
- ホームディレクトリは、ユーザーが自由に利用できるディレクトリ
- ログイン直後のカレントディレクトリとなる
  - ホームディレクトリが存在しない場合には/`ディレクトリ`
- デフォルトでは“/`home/ユーザー名`”となっている
- 様々な設定ファイル(ドットファイル)が格納されている
  - `ls -a`で参照可能

# 絶対指定と相対指定

---

- 絶対指定
  - 「/」(ルートディレクトリ)からの絶対位置を指定する
- 相対指定
  - カレントディレクトリからの相対位置を指定する
  - ファイル名                      カレント内のファイル
  - ディレクトリ名                  カレント内のディレクトリ
  - .                                      カレントディレクトリ
  - ..                                     親ディレクトリ
  - ~(チルダ)                         ホームディレクトリ

# 絶対指定と相対指定



- カレントディレクトリ
  - /home/user
- 指定ディレクトリに対する絶対指定
  - /etc
- 指定ディレクトリに対する相対指定
  - ../../etc

## コマンドライン補完

---

- 長いディレクトリ名入力を省くシェルの機能
- TABキーを押すと、ファイル名が前方一致検索され、候補が1つであれば補完される

例) `cat /etc/sysconfig/network`と入力する

1. `cat /e<TAB>`
  2. `cat /etc/sysco<TAB>`
  3. `cat /etc/sysconfig/ne<TAB>`
  4. `cat /etc/sysconfig/network`
- 補完候補が複数ある場合には、TABキーを2回押すと一覧表示される

## コマンド履歴

---

- シェルはコマンド作業の履歴を取っている
- 呼び出して編集・再実行可能
- 呼び出し方法
  - カーソルの上下      順番に呼び出し
  - historyコマンド      履歴一覧の表示
  - !履歴番号      履歴番号のコマンドを再実行
  - !文字列      文字列で始まる最新コマンド
  - !!      直前のコマンドを再実行
  - コマンド !\*      直前のコマンド引数を再利用

# 環境変数

---

システム全体の環境を設定する値を格納する変数

- 設定方法

- A) シェル変数を設定し、exportする

1. 変数 = 値
2. export 変数

- B) exportで直接設定する

1. export 変数 = 値

- 環境変数の確認方法

- envコマンドで環境変数とその値を一覧できる

# 環境変数PATH

---

- シェルでコマンドを実行する際に、実行するコマンドファイルを環境変数PATHで設定されたディレクトリから検索する(サーチパス)
- ディレクトリの中を、設定されている順に検索を行い、最初に見つかったコマンドファイルを実行する
- コマンドファイルを見つけられなかった場合にはコマンドを実行できない
- whichコマンドで、実際に実行されるコマンドファイルの位置を調べられる

## suによるユーザーの切り替え

---

su [-] [*username*]

- 「-」をつけると、ログインしたのと同じ状態になる  
(環境変数が新たに設定される)
  - 「-」をつけないと、前のユーザーの環境変数を引き継ぐ(PATHなどが変わらない)
- *username*を省略すると、rootへの切り替えとなる
  - /sbin,/usr/sbin等へのPATH設定はrootのみ
  - su -でrootでログインしたのと同じ状態

## コマンドの使用方法を調べる

---

- manコマンド
  - `man [section_number] command`
  - マニュアルの種類によってセクションに分かれている
  - 同じ名前でも複数のセクションに分かれている場合には指定が必要。
  - マニュアルファイルは/usr/man、あるいは/usr/share/manに入っている
  - 環境変数LANGが設定されていると"/usr/share/man/言語名"以下を優先的に参照する
    - LANG=ja\_JP.UTF-8なら/usr/share/man/ja以下を参照
    - LANGを設定しないと英語マニュアルを表示(オススメ)

# 1.103.4 ストリーム、パイプ、リダイレクトを使う

---

## 説明

- ストリームをリダイレクトして、テキストデータを効果的に処理するために接続する。これには、標準入力、標準出力、標準エラー出力をリダイレクトする、あるコマンドの出力を別のコマンドへの入力にパイプする、出力を標準出力とファイルの両方に送るといったことを含む。

## 重要なファイル、用語、ユーティリティ

- tee
- xargs
- <
- <<
- >
- >>
- |
- ``

# 標準入出力

---

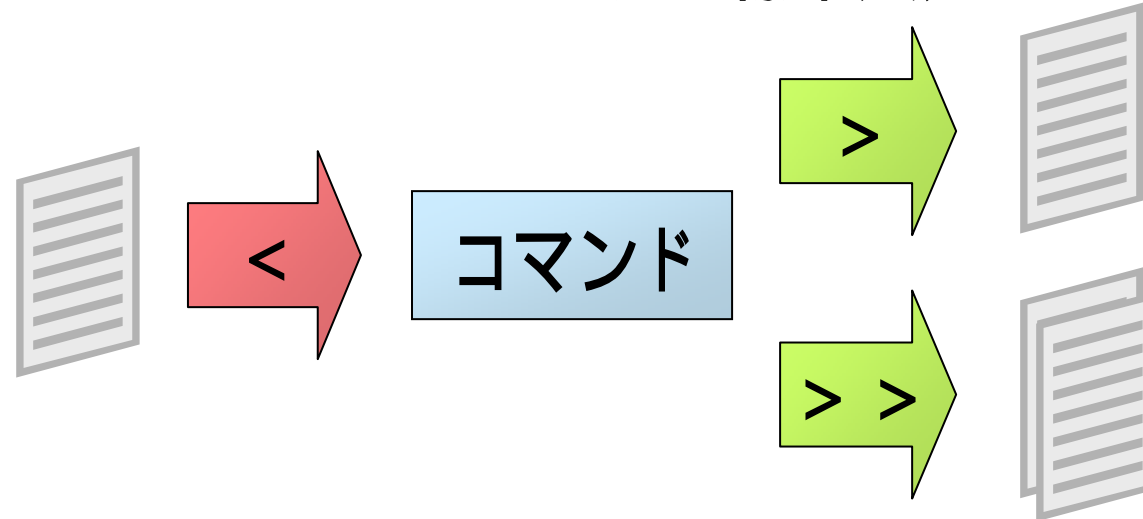
- コマンドは標準入力からデータを受け入れて、処理を行った後に標準出力へと結果を出力する
- エラーが発生した場合には、標準エラー出力へとエラーメッセージを出力する



# リダイレクト

---

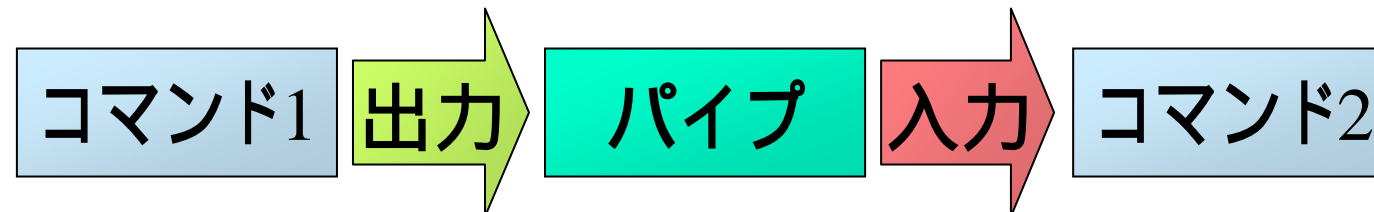
- 標準入出力とファイルの間のやり取りを制御する
  - `command > file` 標準出力をファイルにリダイレクト
  - `command >> file` 標準出力を追加でリダイレクト
  - `command < file` ファイルを標準入力にリダイレクト



# パイプ

---

- あるコマンドの結果を、さらに他のコマンドで処理したい場合に使用
- *command1*の標準出力を*command2*の標準入力へ渡す
  - *command1* | *command2*
  - *command1* > *file*; *command2* < *file*と同等



# 文字列を検索する(grep)

---

- 書式

- `grep key [file]`
- `file`または標準入力から`key`を探し出す
- `key`には正規表現を指定できる

- 正規表現

- `.` 任意の1文字
- `*` 直前の文字の0回以上の繰り返し
- `[文字]` 文字のいずれか1文字
- `^` 行頭
- `$` 行末
- `¥` 続くメタキャラクタを文字として処理(エスケープ)

# まとめ

---

- **学習の目標を定めよう**
  - 例) Linuxを使ったWebサーバー構築
- **できるだけ沢山Linuxに触る**
  - コマンドライン中心に、苦勞なく触れる程度に
  - 色々なディストリビューションを試してみる
- **自分なりに説明できること**
  - 教えることが最高の学習

# LPIメルマガのご紹介

---

- 『LPI通信』

  - Linuxオープンソース関連ニュース

  - Linuxの利用に役立つTipsやテクニック

  - LPI合格者の声

- 『LPIC Level2・Level3を受けてみよう!』

  - Level2・Level3の例題解説

  - Level2・Level3を取得するメリット、活用の事例など

  - Level1をお持ちでない方にも役立ちます

<http://www.lpi.or.jp/mail/>