

シェル

- ・ シェルとは「入力コマンドを解釈し、適当なプロセスを起動する」もの
- ・ 大別してsh系シェルとcsh系シェルがあり、具体的にはsh, csh, tcsh, ksh, bash, zshなどがある
- ・ Linuxではbashが標準になりつつある
- ・ 使いやすいように個々人がさまざまなカスタマイズをしている

102

bash に関わる重要なファイル

- ・ `~/.bash_profile` ログイン時に実行される
- ・ `~/.bash_logout`
ログインシェル終了時に実行される
- ・ `~/.bashrc` bash起動時に実行される

~/.bash_profile に ~/.bashrc を起動するように設定しておくのがいいでしょう

103

エイリアス機能 (alias)

- コマンドに別名をつける機能
例 `alias m=more`
`alias ll='ls -lF | more'`
(複数語をエイリアスするときは引用符が必要)
- エイリアス一覧の表示
`alias`
- エイリアスの取り消し
`unalias ls`
- エイリアスの一時的回避
`¥ls`

104

オプション設定

- シェルの振舞いを設定するもの
- `set -o`
で、すべての設定状態の一覧がわかる
- 設定するときは
`set -o オプション名`
設定解除するときは
`set +o オプション名`

105

オプション設定(続き)

- 設定しておいた方がいいオプション
 - ignoreeof (ctrl-d でシェルを終了しない)
 - noclobber (既存ファイルに>による上書きを許さない)
 - emacs (emacs形式のコマンド編集を使用する)
 - history (コマンド履歴を有効にする)

106

シェル変数

シェルの動作環境を設定するもの

- 変数名 = 値
で自由に変数を定義できる
(=の前後に空白を入れてはいけない!)
- 組込みのシェル変数は大文字
- 変数の値を引用する時はドル記号(\$)を付ける
例 echo \$PATH
- 変数を削除するには unset 変数名

107

コマンド履歴

- シェルは前に実行したコマンドを覚えているので、それを呼び出して、そのまま再度実行したり、一部を修正して実行できる
→ 上向き矢印キーやctrl-p で呼び出せる
- 履歴は history コマンドで見られる

108

コマンドの履歴(続き)

- 直前に実行したコマンドをもう一度実行するには !! と打つ
- historyコマンドの表示で、左端にある数字を指定して再度実行することができる
例 !13
- 以前実行したコマンドで、指定した文字列で始まるものを再度実行することができる
例 !pw (pwdというコマンドを実行したことがあればpwdと全部打たなくてよい)

109

コマンドの履歴(続き)

- 直前コマンドのささいな打ち間違いを修正したい時は
^文字列1^文字列2
とするのが便利
例 cat ~/aa091048/10238734readng.txt
と打ってしまった時に、
^dng^ding
とすれば全部打ち直さないで済む

110

コマンドラインの編集

emacsのキーバインディングで前に実行したコマンドを呼び出し、編集できる
(C- は ctrl- のこと)

- C-p 1つ前のコマンドを表示する
(C-p と C-n で行き来する)
- C-b や C-f や C-d で編集する
- C-k で削除し、C-y で貼り付け

111

コマンド履歴に関する組込み変数

- HISTSIZE
コマンド履歴に記録する最大コマンド数
(暗黙値は500)
- HISTFILESIZE
履歴リストに保存するコマンド数の最大値
(暗黙値は500)
- HISTFILE
履歴リストが保存される履歴ファイルの名前
(暗黙値は ~/.bash_history)

112

コマンド検索パス

- 組込み変数PATHで、ユーザが入力したコマンドの実行可能ファイルを探す
⇒ 検索を行うディレクトリのリスト
例 echo \$PATH をやってみよう
- 変数PATHの変更は注意しながらすること！
例 現在のPATHのリストの末尾に ~/bin/ を追加するには
PATH=\$PATH:~/bin/”

113

カレントディレクトリにパスを通す例 (あまりお勧めしません)

```
$ touch ./abc (まずabcというファイルを作成)
$ chmod +x ./abc (実行権限を与える)
$ abc (コマンドが見つかりませんというエラー)
$ PATH=$PATH":."
$ abc (今度はエラーが出ず終了)
```

114

環境変数

- ・ (シェルではなく) UNIXで使われるすべてのプロセスがそれぞれ保有する変数
- ・ 子プロセスには親プロセスの環境変数がコピーされる
- ・ PATHやTERMは環境変数
- ・ どんな変数も
export 変数名
で環境変数とすることができる
- ・ 現時点の環境変数の設定状況は
printenvコマンドや(引数なしでの)exportコマンドで確認できる

115

代表的な変数

- HOME ホームディレクトリの絶対パスによる記述
- LANG ロケール(ソフトウェアに内蔵される、言語や国ごとに異なる単位、記号、通貨、日付などの表記規則の集合)の値
- PATH コマンドプログラムを検索するディレクトリのリスト
- USER 自ユーザのユーザ名
- PWD カレントディレクトリの値

116

設定ファイルの再読み込み

```
$ source ~/.bashrc  
のようにする
```

117

プロンプトをカスタマイズしよう

- ・ 組込み変数PS1を設定する
→ 現在のPS1の値を見よう
- ・ 役に立つプロンプト変数
 - ¥u 現在のユーザ名
 - ¥H ホスト名
 - ¥h 最初の . までのホスト名
 - ¥W 現在の作業ディレクトリのベース名
 - ¥w 現在の作業ディレクトリ

118

- ・ プロンプト変数(つづき)
 - ¥! 現在のコマンドの履歴番号
 - ¥¥ バックスラッシュの表示
 - ¥# 現在のコマンドのコマンド番号
 - ¥A HH:MMによる現時刻(24時間表記)
 - ¥T HH:MM:SSによる現時刻(12時間表記)
 - ¥t HH:MM:SSによる現時刻(24時間表記)
 - ¥@ 午前・午後表記付きの12時間表記での現在の時刻
- ・ これらを組み合わせて
PS1="¥w@¥h \$ "
などとする。決まったら ~/.bashrc に追加する

119

コマンド入力中の補完

- ・ コマンド入力中にtabキーを押すと、起動可能なコマンドが列挙される
- ・ 引数となるファイル名入力中にtabキーを押すと、ファイル名を(あいまいでない箇所まで)補完して表示する