よしだともこの

Linux事始めの書

第8回　UNIXコマンド使いへの道
——その6　知っていると便利な記号たち

友人から、足首と手首と首から上だけが出るようになっていて、あとはドーンとつながった冬用の室内着をプレゼントされました。当然、「ペンギンの着ぐるみ」or「なまずの着ぐるみ」への応用を検討中です。

よしだともこ　http://www.tomo.gr.jp/

My“Happy 材木屋 Life”

毎年、新年のスタートと同時に届く年賀状を読むと、幸せな気分になるものです。特に、今年の傾向としては、コンピュータには関係してなかった友人たちからの「私もインターネットを使い始めましたので、メールをください」というコメントが目立ちました。さっそく、「この友人は大学時代のクラブ仲間」、「この友人は○○関係」というふうにグループ分けをして、メーリングリストを作ってしまいました。そうすれば、いつ途切れるか分からない1対1のメール交換とは違い、複数の人たちで長期的に情報交換できますからね。

その結果、10年以上も「今年こそ、会いたいですね……」という年賀状のコメントから、一歩も進まなかった関係が、1月を10日過ぎただけで、具体的に会う日取りの相談にまで発展しているのですから、驚いてしまいます。こんなふうに、電子メール(特にメーリングリスト)は、人間関係を変えていくのですね。しかし、こういうスピーディーな時代の到来に、自分自身ついていけなくなって、のんびりしていた時代が懐かしくなるのでは……などと、年明け早々、考えさせられました。

考えさせられたというと、大学院時代の恩師からの年賀状に書いてあった、以下のコメントでした。

「相変わらず材木屋(そのこころは、キが多い)ですね。まぁ、元気でプロダクティブなのは何より。今年もよろしく。」

キが多いから材木屋かぁ……。あまりに的確な表現で、「○○先生に、座布団5枚！」です。確かに、ここ数年の私は、あるときはコレ、またあるときはアレというふうに、一見、なんのつながりのないことに対して、まるで何かにとり憑かれたように活動してきました。書籍も共著を含めて、昨年は4冊[*1]出したのですが、それぞれ切り口が違う……。さらに、大学で教えている授業内容も教えに行く学校[*2]によって違うし、突発の講義や講演内容にしても、毎回、違ったお題を与えられてる……。

なんでこうなるかというと、つまりは、いろんな興味分野を持っているからで、やはりキが多いとしか言いようがない。ある人が何かしてると、それがキになり、あーだ、こーだと一緒に言っているうちに、気が付くと副担当か何かになっていたり、それに関して講義することになってたりする。そういう状態だというのに、別の人が別の分野で楽しそうに何かしてると、やっぱりそれがキになり、さらに他の人をも巻き込んで、突っ走ってしまう……。人と一緒に走ってるのは楽しくて、つい全速力で走ってしまう。結果的に、元気でプロダクティブな1年を、あっという間に終えた私を、周りの人は、あきれて見てたのかも。

一見何のつながりのないことに対して、夢中になることの欠点は大いに認めますが、逆に、1人が同時に複数のことにのめり込むことで、関連性が発見できたりもします。例えば、1999年11月から12月にかけて私は、自宅の庭を中心としだ“大幅な改善”に関わっていました。

実は、この家に引っ越してきた当初は、庭を耕してトマトやキュウリを作ったり、芝生を植えて庭でバーベキューがで

---

*1　「Wnn6徹底入門」、「ホップ!ステップ！Linux！」を翔泳社から。「インターネット講座」を北大路書房から。「よしだともこのルート訪問記書籍版」をソフトバンクから。

*2　1999年度は、京都ノートルダム女子大学、京都大学、龍谷大学瀬田キャンパスに教えに行ってました。1999年度が終ったころに書いたりして(笑)。

きるようにしてたんですが、これは、予想以上に時間と手間がかかる作業でした。最初は、実家の母が、わき目の取り方や肥料のやり方を指導しに来てくれてたんですが、そのうち「野菜は手間がかかるから、無理に植えなくてもいいよ」と言うようになり、結局、植木やツル性植物は伸び放題、庭に雑草が目立ちだし、暇を見付けて草抜きするぐらいではとうてい追い付かない……という状態になっていました。では、どうすれば、再び美しい庭が得られるのか。

「もっと、メンテしやすい庭にしてしまおう」

ここ数年、ずっとこれを考えていて、今回、やっとこさ実行に移したわけでした。

具体的には、外構工事の会社の方と詳細に話し合い、花壇の部分を比較的少なくし(それ以外はコンクリートなどにし)、水道の口を多くして水やりをしやすくし、伸び放題だったツル性植物と、扱いに手間取った芝生は全面撤廃し、代わりにドーンとウッドデッキを設置してもらいました。その後、自分で花壇に、扱いやすい草花だけを選んで植えました。

その過程で再認識したのは、「もし私が、庭仕事が好きで好きでたまらなくて、時間があると無意識に庭に出て、植木の刈込みや雑草を抜くのを苦にしないタイプなら、メンテしやすい庭にする必要はなかった」ということでした。そして、「今の時点では、雑草はまったくなくなって、非常にすばらしい状態になってるけど、気をゆるめると、すぐに雑草が生えてくる。今後は、メンテを怠らないようにしよう」と決意しました。

また、庭により頻繁に出るようになってみると、近所の人たちと顔を合わす機会も増え、「きれいになりましたねぇ」と多くの方から言われました。それで気が付いたのは、きれいでなかったときの庭は、私自身はあんまり見ないようにしていたのですが(苦笑)、近所の人は、少なくとも私よりは、何倍も庭に出て、うちの庭をしっかり見てたんですよね。そして、「よしださん家の雑草の種が、こっちの庭に飛んできたらイヤだなぁ」と思っておられたのでしょう。伸び放題のツルが、隣の庭に伸びていくことに対して私は「勝手に、切ってくださ〜い」と笑って言ってましたけど、相手にしてみれば、根本的に解決してほしかったに違いありません。

いろいろな面で、庭の管理というのは、会社や学校のサーバ構築や管理、運用に通じることが多いように思えました。サーバ構築が好きな人がいて、その人は知識も経験も豊富で、時間もあって、常に面倒を見ているのなら、メンテしやすくなくていいのです。メンテに技術が必要で時間がかかる方が、かえって好まれるかもしれません。また、1度、メンテしやすくしたとしても、それ以降、最低限の時間をメンテに費やさなければ、せっかくのシステムは、すぐにダメになってしまいます、最低限の愛情を注ぎ続けなければ。また、変な状態のまま運用していると、周りの人に迷惑をかけます。

さて、最近、まったくネットワーク環境のなかった職場(特に学校)に、ほとんどお金や人をかけず、自分たちの力だけで、LANを構築したという話をよく聞きます*3。中古のパソコンにPC UNIXをインストールしたものをサーバにして、LANのケーブルを自分たちで引き回して、環境を作るそうです。これはまさに、運動場に新しく花壇を作るようなものですね。

必要だと思う人たちの努力で、土が掘り起こされ、水やりをするための水道の口が増設され、花の種が蒔かれる……。その中では必ず、「なんで、学校に花壇なんか必要なんだ?」と反対する人への説得も必要でしょうし、せっかく花壇を作っても、水やりを担当する人が自分だけだと、自分がなんらかの理由で水をやれないときには花が枯れてしまいます。そのためには、事前に仲間(説得や水をやるチーム)を結成しておく必要があるのは、花壇もLANも同じですね。

「自分が転勤した後のことも考えて、メンテしやすいネットワークを作るように心がけています。サーバにお金が使えない我々には、手作りパソコンにインストールしたLinuxが安定して動いてくれるのは、大助かりです。職場に仲間を作ることがもっとも大切です。仲間作りには、メーリングリストが多いに役立ちますね。」

そんな話を聞きながら、「庭管理とネットワーク管理は、共通点が多いなぁ……」と、妙にニヤニヤしていた私なのでした。

こういう理由で、年賀状に私のことを材木屋と書かれた○○先生、今年も私は、キが多い材木屋はやめられそうにありません。それに、私が「元気でプロダクティブ」なのは、材木屋をやっているからなんですよ。いつかきっと、材木屋だからこそのアウトプットを出したいと思いますので、それまでは、一貫性が見いだせない私の活動に、どうか目をつむっていてください。

UNIXコマンド使いへの道 〜知っていると便利な記号たち〜

さて。これまでこの連載ではgrepやsedの便利な使い方、そのコマンド行をファイルに保存して、UNIXコマンドのように使う「シェルスクリプト」などを紹介してきました。その過程で、「説明もなく、この記号を使ってもいいのかしら……」

*3 他誌で恐縮ですが、ソフトバンク発行のUNIX USERでの2000年3月号(2月8日発売)と4月号(3月8日発売)の連載記事に、この話をまとめています。

と思うことが、何度かありましたので、今回は、記号をまとめてみます。もっとも基本的なものは、表1のものです。

では、1つ1つ、説明していきましょう。

●入出力の切り換え(リダイレクション)

まずは、入出力の切り換え(redirection ：リダイレクション)の記号の説明から。

UNIXの特長を説明した資料に、以下のように書かれているのを見かけたことがありませんか？

> 入力データの読み込みや出力データの表示をする装置が固定的ではない。コマンドの実行時に指定、切り換えができるように、UNIXでは「標準入出力」と呼ばれる論理上の入出力機構が設けられている。入出力機構には、標準入力、標準出力、標準エラー出力がある。

特に何も指定しなければ「標準入力はキーボード、標準出力は画面、標準エラー出力も画面」と割り当てられるものの、いとも簡単に入力や出力が変更できるのは、もともとそういうふうに作ってあるからだというのです。

で、入力や出力を切り換えることが「redirection」(リダイレクション)と呼ばれて、主に次の記号が使われます。

< ファイル名	入力をファイルからに切り換え
> ファイル名	出力をファイルに切り換え
>> ファイル名	出力をファイルへ追加

まず、「入力はファイルから」って言われても、どんなときに使うのか、ピンと来ないかもしれません。そこで、例を考えてみました。

電子メールを読み書きするコマンドは、「mail」コマンドです。今どき、こういう素のコマンドでメールを書く人はいないと思いますが、実行してみましょう。mailコマンドの後ろに、メールアドレスを書き、リターンとします。

```
$ mail tomo@tomo.gr.jp
Subject:
```

「まずは、Subjectから内容を書いてくださ〜い」というモードになりましたね。Subjectフィールドを書き終ったらリターンし、その後、内容を入力していき、本文を書き終ったら、“Ctrl-d”とします。すると、プロンプトが戻ります。

```
$ mail tomo@tomo.gr.jp
Subject: test
Dear Tomoko,
A Happy New Year!
bye
EOT
$
```

これでメールは書けるのですが、こういう画面でメールを書くと、日本語だって使いにくいですよね。そういう場合に、画面上でリアルタイムにメールを書くのではなく、あらかじめ書いておいた“letter”というファイルを、メールコマンドに“< ファイル名”を使って渡すのです。

```
$ cat letter
Subject: test2
ともこさん
今年はちゃんと原稿の締め切りんを守るやでぇ〜。
$ mail tomo@tomo.gr.jp < letter
$
```

これで、本来ならば、画面上でメールの本文を入力すべきところだけど、あらかじめ書いておいたファイルがあるので、入力をそのファイルから切り換えたことになります。

次の、「出力をファイルに切り換え」は、「画面にだらだら流したいんじゃなくってぇ、結果をファイルに保存したいわけよ」というときに使います。例えば、2000年のカレンダを表示させるコマンドとして、“cal 2000”を実行すると、画面に2000年のカレンダが流れます。

これを、“calendar”という名前のファイルに保存したいときに、次のように実行します。

```
$ cal 2000 > calendar
$
```

calendarファイルの内容は、“cal 2000”とだけ入力したときに、画面に流れた内容です。

最後の、「出力をファイルへ追加」というのは、すでに存在するファイルの後に、出力を追加していくときに使います。つまり、“cal 2000 > calendar”と実行すると、もし、calendarというファイルがすでに存在し、その中に重要な内容が記述されていたとしても、元のcalendarファイルは壊れ

表1

記　号	意　味
< > >>	リダイレクション
|(パイプ)	前の出力を次の入力にする
&	バックグランド処理
;	複数のコマンド列挙

て、“cal 2000”の出力結果が上書きされてしまいます。しかし、“cal 2000 >> calendar”なら、元のcalendarファイルの後に、追加されます。つまり、

```
$ cal 1999 > calendar
$ cal 2000 >> calendar
$
```

のように実行すれば、calendarファイルは、前半が1999年のカレンダー、後半が2000年のカレンダーになります。元々、存在しないファイルに、“>>”で書き込んでみると、“>”で書き込んだのと同じ結果になりました。

すでにご存じの方が多いと思いますが、メーリングリストなどで、「この部分は〜さんへの質問」という意味で、

```
どう思いますか。>よしださん
```

というふうに使うときの“>”マークは、UNIXの“>”からきています。コメント行に“#”を使うのも、シェルスクリプトから派生した習慣ですね。

●前の出力を次の入力にする(|)

前の出力を次の入力にする“|”(パイプ)の説明としてよく使われるのは、「これを使えば、複数の行のコマンド実行を、1行にまとめることができる。しかも、テンポラリファイルを作らずに……」というものです。まず、以下のようにコマンドを実行したとします。

```
$ ls -l > tmp
$ wc -l tmp
241
$ rm tmp
$
```

これは、“ls -l”の出力結果をまず、“tmp”という名前のファイルに入れます。そして、「何行あるか教えておくれ」という意味の“wc -l”コマンドを実行し、241行という結果を得ます。その後、rmコマンドでtmpという名前のファイルを削除しているわけです。これと同じことが、パイプを使うと、1行でできてしまいます。

```
$ ls -l | wc -l
241
$
```

めでたし、めでたし。

●バックグランド処理(&)

これについては、この連載ですでに詳しく書いたことがある(1999年10月号)ので、復習になりますが、コマンド行の最後に“&”をつけると、処理が背後での実行(バックグランド処理)にまわされます。そのため、リターンを押した瞬間に、次の行にプロンプトが戻ってきて、別の作業ができるというわけです。

```
$ cc tomo.c -o tomo &
[1] 705
$
```

プロンプトは戻ってきているものの、その作業は終ったわけではなくて実行中なので、「ジョブ番号は1番、プロセス番号は705番だよん」ということを、知らせてくれているわけです。銀行や郵便局で渡される、待ち行列の札のようですね。

ちなみに、“cc tomo.c -o tomo”という行は、“tomo.c”というC言語のプログラムをコンパイルして、tomoというオブジェクトを作るという意味なので、“&”をつけてEnterキーを押した瞬間には、まだ“tomo”はできていません。別のコマンドを入力して結果を得た後に、「終ったよ」という意味で、“Done”という合図があります。こんな感じです。

```
$ cc tomo.c -o tomo &
[1] 705
$ date
Wed Jan 19 20:01:08 JST 2000
[1]+  Done        cc tomo.c -o tomo
$
```

UNIXでは、このように処理を最初からバックグランドで実行させる以外に、表(フォアグランド)で実行中の通常のプロセスを一時的に中断させたり、途中からバックグランドに回したり、再び表に戻したりと、やりたい放題もて遊ぶこと(制御すること)ができます。これが「ジョブコントロール」と言われる機能です。ただ、最近ではCPUパワーが上がっているので、自分が書いたC言語のプログラムのコンパイルや、TeXのコンパイル程度では、別にわざわざ中断させたり、バックグランドに回したりする必要がなくなってしまったのがちょっと寂しい……なんて書くと、年配者まるだしで恥ずかしいでしょうか……。

しかし、ターミナルからアプリケーションを起動するときに使えば、そのターミナルでどんどんと新しいアプリケーションが起動できるので、非常に便利です。

```
$ netscape &
$ mule &
```

```
$ xv &
$
```

こんな感じ。こちらも、マスターすれば、めでたし、めでたし、ですね。

●複数のコマンド列挙(;)

最後に紹介する記号は、複数のコマンドを列挙するための“;”です。これは、非常に単純な話で、

```
$ cd WWW
$ mule index.html &
```

というふうに実行するときに、

```
$ cd WWW ; mule index.html &
```

というふうに書けば1行で書けるよ、という話です。

私が“;”を使う瞬間というのは、tcshやbashの持つ、「Ctrl-pで、過去のコマンド行を表示させる」を使っているときに、例えば、

```
$ mule index.html &
```

を見つけ出し、「おっと、今は違うディレクトリにいるから、先にディレクトリを移動しておかないとダメなんだ……」と思って、Ctrl-pで表示させたコマンド行の前に、ディレクトリ移動のコマンドを加えることが多いです。こんな感じです。

```
$ cd ~/WWW ; mule index.html &
```

あとは、かなり個人的で汎用性のない話ですが、提出した雑誌原稿の版下を校正していて、「おっと、この行では、先にディレクトリを移動しておかないといけなかった。版下の段階で行数が変わると編集部も困るだろうから、cdの行を新しく加えるのではなく、同じ行に“;”を使って加えておこう」というときに使った経験もあります。

こちらも、CPUパワーが低かった頃に愛用してました。さらに、バックグランド処理も知らなかった時期には、『“cc t1.c -o t1”のコンパイルに約1分かかるから、トイレに行って帰ってくるまでに、3つコンパイルしておこう！　人間様がトイレに行っている間に集中してコンピュータに働かせるなんて、我ながら素晴らしいアイデア！」と思って、以下のような行をまとめて入力してから、トイレに向かったものでした。

```
$ cc t1.c -o t1 ; cc t2.c -o t2 ; cc t3.c -o t3
```

で、席に戻ってきたときにプロンプトがまだ出てなかったら、「セーフ、セーフ、間に合ったぁ」と、内心、嬉しかったです。トイレから走って戻っていたことは、言うまでもありません:-)。

ま、そのうち、こういうのはバックグランド処理に回せばいいとか、シェルスクリプトにしておくといいとか、makeファイルの書き方だの、いろいろなことを知り、こういう行を手入力することはなくなりました。が、「セーフ、セーフ、間に合ったぁ」という、あのころの密かな楽しみを思い出すたびに、「他にもっと楽しいこと、なかったん？　かなり暗い新入社員生活を送ってたのね」と苦笑しています。

今回はここまでです。次回は、便利なシェルスクリプトについて、いくつか紹介しようと思っています。

では、また。

COLUMN

新しくLUGを作ったら……

「新しくLUG(Linux Users Group)を作ったときは、どういうところに広報すれば良いのでしょうか？」という質問を受けたので、私の知っている範囲内でまとめてみました。

①http://www.linux.or.jp/をまとめてくださっている人々(Webmasters)に連絡

メールの宛先：webmaster@linux.or.jp

http://www.linux.or.jp/community/group/ **のページに追加される。**

②ChangeLogのWebページ(http://www.changelog.net/)の発信元に連絡

メールの宛先：news@ChangeLog.net

http://www.changelog.net/ **のページに掲載されると同時に、メールマガジンとして、**ChangeLog**講読者に配信される。**

③Linux Japan 誌の編集部に連絡

メールの宛先：lj-lug@linuxjapan.com

Linux Japan**誌の「**Linux User Group Map**」のページに掲載されると同時に、「**Web**版** Linux User Group Map**」(**http://www.linuxjapan.com/Link/lugmap.html**)に掲載される。**

④Linux Community連絡会(http://www.linet.gr.jp/llc/)の発信元に連絡

メールの宛先：koyama@linet.gr.jp**あるいは**kojima@linet.gr.jp**まで連絡**

(よしだともこ)