2000年はJava＋LinuxでE-Commerce？

Javaを開発し、これまでJavaの標準化を進めていたSun Microsystems（以下Sun）が、国際標準化団体のECMAからJavaを引き上げてしまったというニュースが入ってきた。SunはJavaに関して新しい方向に動きだしたようだ。

Java対応の開発ツールJBuilderを開発しているInpriseは、Sunと共同でLinuxに対応した「Java 2 Platform Standard Edition」を開発していると発表した。そしてSunは同社のWebサイトで、そのプレビュー版「Java 2 SDK 1.2.2RC1」を公開し、正式版は2000年上期を予定しているという。

Javaは動作するプラットフォームを選ばないことが特徴になっている。しかし、同じプログラムを処理能力の低いマシンで実行すれば当然レスポンスが遅くなるわけで、Webブラウザでアクセスするアプリケーションでは、サーバ側で処理するのが主流になっている。

そして、そのサーバプログラムとしてJavaが一番使われている分野にE-Commerce市場がある。1999年7月に幕張メッセで行われたLinux Exhibition '99基調講演で、IBMのLinux戦略担当Vice PresidentのRichard J. Sulivan氏は、e-businessのフレームワークの中にLinuxプラットフォームが加わっていくことを発表していた。そのe-businessのポータビリティを実現しているのは、Javaである。

WebサーバではかなりのシェアをもつLinuxマシンが占めている。その応用分野でもあるE-Commerce市場は急激に成長しているため、多くのシステムにLinuxが使われていくのが自然である。Javaを利用したE-Commerceは、Linuxサーバ市場の発展に大きく貢献するだろう。

Software　発売日　1999年12月3日

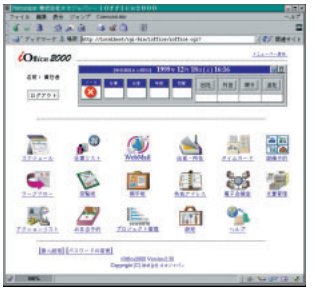
Web対応グループウェア「iOffice2000」

URL　http://www.neo.co.jp/ioffice/

発売　株式会社ネオジャパン
TEL　045-912-5971
価格　3万9800円（5ユーザー）～

ネオジャパンのイントラネットWeb対応グループウェアである「iOffice2000」がバージョンアップされて、2.30となった。価格は5ユーザーで3万9800円と据え置き。前バージョンを購入している場合は、同一のライセンスキーで新バージョンを使用できる。またiOffice2000の機能を拡張するiモード対応版も2.0にバージョンアップされている。iOffice2000本体を購入したユーザーは無償で利用できる。

サーバマシンには、Linux、FreeBSD、WindowsNTなど多くのOSが利用可能だ。クライアントは、Netscape Navigator、Internet ExplorerなどのWebブラウザを用いる。



バージョン2.30は、新たに「プロジェクト管理」、「お弁当予約」機能を追加し、全部で15の機能を提供する。ユーザーは、必要な機能のみを選択して運用が可能だ。またアドインであるiモード対応版も「タイムカード」、「設備予約」、「仕事リスト」、「ワークフロー」、「回覧板」の5つの機能を追加し、全部で9つの機能を提供している。

Software　発売日　1999年12月20日

Wordドキュメントからテキスト抽出ができるテキストエディタ「デ変研TEXT」

URL　http://village.infoweb.ne.jp/~dehenken/

発売　有限会社データ変換研究所
TEL
価格　4600円

Microsoft Wordなどのドキュメントから、テキスト部分を抽出する機能を持ったテキストエディタ「デ変研TEXT」が、データ変換研究所より発売された。Linux、FreeBSD、NetBSDに対応する。価格は4600円。同社では、1年間で2000本の販売を見込んでいる。

デ変研TEXTは、従来のJIS／EUC／SJISに加え、2種のユニコード（UCS-2／UTF-8）を自動判別し、読み込み／保存が可能だ。

また、Windowsアプリケーションで作製したドキュメントから、テキスト部分を抽出し、読み込むことができる。対応するアプリケーションは、Word（95～2000）、Excel（95～2000）、PowerPoint（97、2000）、一太郎 9、OASYS 6。

Software　発売日　1999年11月30日

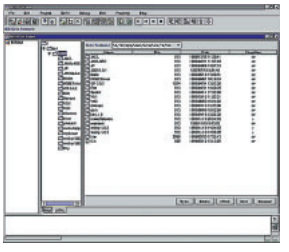
Linux用開発環境C／C++コンパイラ「C++ Package」

URL　http://www.fqs.co.jp/

発売　株式会社富士通九州システムエンジニアリング
TEL　092-852-3126
価格　3万円

富士通九州システムエンジニアリングから、インテルx86 PCのLinux上で動作するC／C++コンパイラ「C++ Package」が発売された。1999年7月より発売中の「Fortran＆C Package」からFortranコンパイラを除いて、低価格化したもの。価格は一般ユーザー向けが3万円で、教育／研究機関向けは2万円。今後1年間で、約1万本の発売を予定している。

C++ Packageは、富士通が開発した、C／C++コンパイラとGUI開発支援ツール「Relaxin」をセットにしたもの。出力されるメッセージやマニュアル（man形式、HTML形式）はすべて日本語になっている。Relaxinの機能により、プロジェクト単位のビルド／デバッグが行える。またデバッガは対話形式を採用している。



Hardware　発売日　1999年12月1日

モバイルメールサーバ機能を持つサーバ「MM QUBE」

URL　http://www.nttdocomo.co.jp/

発売　NTT移動通信網株式会社
TEL　0120-774-360
価格　29万8000円

NTT移動通信網は、モバイルメールサーバ機能を搭載したオールインワンサーバ、「MM QUBE（エムエムキューブ）」を発売した。本体価格は、29万8000円。　米コバルトネットワーク社の「Cobalt Qube2」をベースにしており、サイズは184mm（W）×197mm（H）×184mm（D）とコンパクト。

Cobalt Qube2のWebサーバ、FTPサーバ、DNSなどのインターネットサービス機能に加え、携帯電話やPHSを用いた、電子メールの発信や受信を可能にする「VALUE-MAIL」機能を搭載している。携帯電話の番号によって接続前に認証とメールチェックを行うため、従来のプロバイダ経由のPPP接続と比較して通信時間を短縮でき、コストダウンできる。またIDやパスワードの漏洩があっても、発信者の電話番号で認証するので、不正アクセスが防止できる。

Hardware

データベースサーバ「NL Server with UniSQL」シリーズ

発売日　1999年12月13日

発売　ノーザンライツ株式会社
TEL　044-850-2391
価格　17万4000円〜

URL　http://www.nlights.co.jp/

ノーザンライツは、データベース「Infrover/UniSQL」をプリインストールしたLinuxサーバ「NL Server with UniSQL」を発売した。価格は、CPUがPentium II 400 MHz、メモリが128Mバイト（最大1Gバイト）、HDDがUltra2 Wide SCSIの9.1Gバイトという構成のエントリーモデルで17万4000円となっている。モデルによってVine Linux 1.1またはRed Hat Linux 6.0を採用している。

Infrover/UniSQLは、次世代のオブジェクト指向技術を統合したリレーショナルデータベースで、画像や音声など、マルチメディアデータや複合文書管理に適している。SQL言語の完全なサポートに加え、オブジェクト指向データベースの機能を拡張したSQL/X言語仕様を提供する。将来的には、Java、CORBAに対応する予定。

Infrover/UniSQLの開発者向けの技術サポートサービスは、有償でNTTデータが行う。料金は年間30万円。

Hardware

小型エントリーサーバ用UPS「Smart-UPS 500」

発売日　2000年1月12日

発売　株式会社エーピーシージャパン
TEL　03-5434-2021
価格　6万9800円

URL　http://www.apc.co.jp/

エーピーシージャパンは、小型エントリーサーバ向けのUPS「Smart-UPS 500」を1月12日から発売する。価格は6万9800円。Smart-UPS 500は、Smart-UPSシリーズの中で容量が最少のモデルで、小規模なエントリークラスのサーバに適している。

ラインインタラクティブ方式という構造を採用しており、平常時にも±20%程度の電圧変動を自動的に補正する。また停電時の切り換えも短時間で行える。バッテリはホットスワップ可能で、交換時にシステムを停止する必要がない。

電源管理ソフトウェアは、標準でWindowsNT用の「PowerChute plus」が添付されている。UPSの遠隔監視、停電時の自動シャットダウン、スケジュールによるサーバの停止／起動などが可能だ。Linux対応の「PowerChute plus for Linux」は、1999年11月末から同社のWebサイトでダウンロードできるようになっており、WindowsNT版と同様の機能が利用できる。

Software

日本語入力システム「VJE-Delta Ver.3.0 for Linux／BSD」

発売日　2000年1月29日

発売　株式会社バックス
TEL
価格　5800円

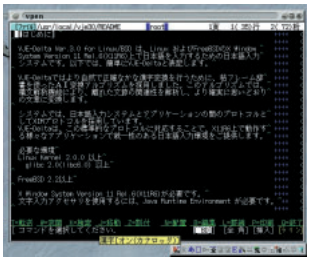
URL　http://www.vacs.co.jp/

バックスは、Linux／FreeBSDに対応した日本語入力システム「VJE-Delta Ver.3.0 for Linux／BSD」を発売する。価格は5800円。

現在販売中のVer.2.5をバージョンアップしたもので、新たにWindows版と同等のツールバーを採用し、操作性を向上している。変換エンジンには、格フレーム辞書を用いたAI変換を採用している。構文解析機能により離れた文節の関連性を判断し、より正確な変換を実現する。

VJE-Deltaとアプリケーション間のプロトコルは、標準的なXIMを用いており、X Window System（X11R6）上のさまざまなアプリケーションで使用できる。

1999年11月より、ベータ版が同社のFTPサイト（ftp://ftp.vacs.co.jp/pub/unix/vje30/Linux）から入手可能になっている。Linux版は、tarボール、RPM、debパッケージで提供されている。LASER5 Linux 6.0、TurboLinux日本語版4.2、Debian GNU/Linux 2.1などで動作することが確認されている。



Software

テキストエディタ中心の統合ソフト「XZ for Linux」

発売日　2000年3月

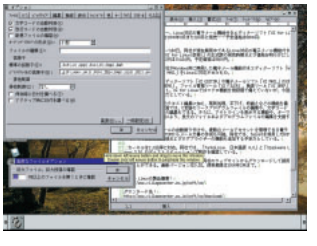
発売　株式会社ビレッジセンター
TEL
価格　9800円

URL　http://www.villagecenter.co.jp/

ビレッジセンターは、Linux対応の電子メール機能付きエディタ「XZ for Linux（仮称）」を2000年3月に発売する。価格は9800円の予定。

XZ for Linuxは、同社が開発したWindows版の電子メール機能付きエディタ「WZ EDITOR 4.0 with WZ MAIL」をLinuxに対応させたもの。テキストエディタ「XZ EDITOR」、電子メールソフト「XZ MAIL」、メモツール「XZ MEMO」、ファイル管理ツール「XZ FILER」、検索ツール「XZ GREP」からなる。WZ EDITORが持つマクロ機能は含まれないが、今後サポートされる予定。

XZ for Linuxのα版（0.52）が同社のWebサイト（http://www.villagecenter.co.jp/soft/xz/）よりダウンロード可能。2000年2月まで試用できる。



News Headline from 日刊アスキーLinux

www.Linux24.com

レーザーファイブ、でんさテクノ東京アクシスソフトウェアが、ASP事業で提携

1999年12月15日

レーザーファイブは、ASP（アプリケーション・サービス・プロバイダ）事業の分野で、でんさテクノ東京およびアクシスソフトウェアとの提携を発表した。でんさテクノ東京は、ネットワークシステムの構築やサポートなどのサービスを提供している、日立電子サービスの子会社。アクシスソフトウェアはWebアプリケーションなどを開発しているソフトウェアメーカー。

この提携により、3社は中小企業やSOHO事業者を主なターゲットとしたサービス「L-Servo（エル・セルボ）」を提供する。L-ServoのOSにはLASER5 Linux 6.0を、グループウェアにはアクシスソフトウェアのiモード対応のWebグループウェア「SuperVIP」を採用。システムの構築や運用およびサポートは、でんさテクノ東京が行う。

レーザーファイブ
(http://www.laser5.co.jp/)
でんさテクノ東京
(http://www.dtts.co.jp/)
アクシスソフトウェア
(http://www.axissoft.co.jp/)

米TurboLinuxが、「Linux-SNA接続技術」をオープンソース化へ

1999年12月15日

ターボリナックス ジャパンは、米TurboLinuxが「Linux-SNA接続技術」をオープンソース化することを発表した。これは、IBMのメインフレームのシステムネットワーク「SNA」とLinuxシステムとをTCP/IPで接続するもの。同技術は、Linuxのカーネル2.3.21で開発されており、「Microsoft SNA Server」や「IBM eNetwork Communications Server」とLinuxマシンの接続が可能となる。

TurboLinux (http://www.turbolinux.com/)

「NetFront for Linux/GTK」評価版の無償配布開始

1999年12月15日

アクセスが、情報家電向けブラウザ「NetFront」のLinux版である「NetFront for Linux/GTK」評価版の無償配布を開始した。

同社によると、今後市場の要求を見て2000年中に製品版をリリースする計画で、価格や開発キットのリリースは現時点では未定。

「NetFront」は、セガ・エンタープライゼスのDreamcastやカシオ計算機のCASIOPEIAなどの情報家電に採用されており、Cookie、Javascriptなどに対応している。動作環境は、インテルx86を採用したLinux（カーネル2.0以上）で、ほかにXFree86 3.3、glibc 2.0または2.1、GTK+ 1.2、zlib 1.1が必要。LASER5 Linux 6.0、Vine Linux 1.1での動作が確認されている。

アクセス
(http://www.access.co.jp/)

Blackdown、Linux Java 1.2.2 RC3をリリース

1999年12月9日

以前からLinux環境へのJava移植を手掛けてきたBlackdown Teamから「Java Platform 2 SDK Version 1.2.2-RC3 (Release Candidate 3)」が発表された。

Blackdownは、Linux環境にJavaを移植するために結成されたボランティアの組織。Java2 SDK 1.2.2-RC3の動作に必要とされる条件は、カーネルが2.2.0以上、glibcが2.1.2以上となっている。

Java Linux
(http://www.blackdown.org/)

Linux版Quake III Golden Master完成

1999年12月9日

米Lokiは、Linux版Quake III ArenaのGolden Masterが米id Softwareの認可を受けたことを発表した。Lokiは、「Civilization」や「Myth II」などの人気ゲームタイトルをLinuxプラットフォームに移植しているソフトウェア会社で、12月2日には、id SoftwareとQuake III Arenaの販売契約を発表している。

Loki Entertainment Software
(http://www.lokigames.com/)
id Software
(http://www.idsoftware.com/)

米3dfxがGlideとハードウェア情報を公開

1999年12月8日

Voodooシリーズのビデオカードで知られる米3dfxは、3DグラフィックスAPIであるGlide APIと、同社のビデオカードのハードウェア情報をオープンソースとして公開すると発表した。

これにより、3Dグラフィックスを使用したソフトウェア、特にゲームのLinux版の開発が進むと期待される。

Glide Source Kitsは同社のWebサイトよりダウンロード可能。

Linux.3dfx.com
(http://linux.3dfx.com/open_source.htm)

XFree86 4.0のリリース時期発表

1999年12月8日

XFree86 Projectは、XFree86の次のプリ4.0（ベータ版に相当）である3.9.17が本年中にリリースされることと、4.0はその約2カ月後にあたる2000年第1四半期中頃になると発表した。

XFree86 4.0は、TrueTypeフォントや3Dグラフィックのサポート、グラフィックカードに依存しないローダブルモジュールによる実装などの新機能を持ち、長

く待ち望まれている新バージョン。

また、現在の3.3系列のバグフィクス版であるXFree86 3.3.6については、3.9.17のリリースと同じ頃になるという。

XFree86 Project
(http://www.xfree86.org/)

米Inpriseと米Sunが Java 2 Platform Standard Edition Linux版を共同開発

1999年12月8日

米Inpriseは12月7日に、Linux対応のJava言語開発ツール「Java 2 Platform Standard Edition（J2SE）」を米Sun Microsystemsと共同で開発していることを発表した。これにより、Linux上でJavaベースのアプリケーションの開発および運用が可能になる。

同製品のベータ版である「Java 2 SDK v 1.2.2 Release Candidate 1 for Linux」が同社のWebサイトに12月8日から公開されており、無償でダウンロードできる。対象はJDC（Java Developer Connection）の会員だが、名前やメールアドレスなどの登録だけで誰でも会員になれる。会費は無料。

なお、正式版のダウンロードも無償で、2000年上半期に公開の予定。同製品には米InpriseのJava開発ツールが同梱される見込み。日本法人のインプライズによれば、同梱製品は「Borland JBuilder 3 Standard Edition」のサブセット（基本機能版）だという。

インプライズ
(http://www.inprise.co.jp/)

グループウェアソフト「TeamWARE Office for Linux V5.3」を発売

1999年12月7日

富士通南九州システムエンジニアリング（MQS）は、Linux対応のグループウェアソフト「TeamWARE Office for Linux V5.3」を12月1日に発売した。出荷開始は2000年1月末。同製品は、MQSと富士通が共同開発したもの。また、北米では富士通の子会社である米Fujitsu Software、ヨーロッパでは同子会社のフィンランドTeamWARE Groupが2000年第1四半期に販売を開始する。

TeamWARE Officeの対応OSは、Windows NT、Solaris、およびUXP／DS（富士通の商用UNIX）だったが、企業において導入が進んでいるLinuxにも対応した。対応ディストリビューションは、「TurboLinux Server 日本語版 6.0」および「Caldera OpenLinux 2.3」。対応ディストリビューションは順次拡大される予定。

富士通南九州システムエンジニアリング
(http://www.mqs.co.jp/product/twolnx/)

Lotus Domino R5 for Linuxの完成版ダウンロード開始

1999年12月3日

米Lotus DevelopmentのWebアプリケーションサーバ「Lotus Domino R5 for Linux」の、最終フィックスバージョンであるGold版が、Lotus DominoのWebページからダウンロード可能になった。ファイルサイズは約90Mバイト。90日間、無料で試用できる。対応するディストリビューションは、Red Hat Linux 6.0と、Caldera OpenLinux。

Lotus Domino R5 for Linuxは、XML機能を強化したLotus Domino R5.0.2に準拠したもの。日本での出荷は2000年第1四半期が予定されている。

Lotus Development
(http://www.lotus.com/)
Notes.net (http://notes.net/)

開発ツール「TIPPLER for Linux」をキャンペーン価格で販売

1999年12月1日

日本ユニシスは、GUIアプリケーション開発ツール「TIPPLER for Linux」のキャンペーン価格販売を開始した。通常価格（開発環境基本構成）30万円を、約4割引の19万円で販売する。期間は2000年3月末まで。

TIPPLERは、オブジェクト指向のスクリプト言語「Uniscript」を核とした開発ツール。CやC++よりも、1/10以下の工数でGUIアプリケーションを開発することができるという。対応するディストリビューションは、LASER5 Linux 6.0と、TurboLinux日本語版4.0となっている。

日本ユニシス
(http://www.unisys.co.jp/)

ぷらっとホームが、Webメール型メールサーバを発売

1999年12月1日

ぷらっとホームは、Webメール環境を容易に構築できるソフトウェア「WallEdge MailServer スターターパック」を発売した。

WallEdge MailServerを利用すれば、Webブラウザからのメール確認が可能になる。Apache-SSL対応の認証証明、ベリサイン・セキュア・サーバID（別途購入必要）や、MIME、HTML、MHTML、cc:Mailの添付ファイルに対応している。

バージョンがVer.1.5に上がり、従来のFreeBSDに加えてLinux（Red Hat Linux 5.2／6.0英語版）にも対応した。また仮想フォルダ機能の装備や、Perl 5.005での動作が可能となっている。

ぷらっとホーム
(http://www.plathome.co.jp/)

サーバ管理ツール HDE Linux Controller 1.0が発売

1999年11月30日

ホライズン・デジタル・エンタープライズは、2000年1月31日より、LinuxサーバをWebブラウザで管理できるツール「HDE Linux Controller 1.0」を発売する。価格は未定。

HDE Linux Controller 1.0は、同社が無料配布している「HDE-WUI」の商用パッケージ版で、店頭販売されるもの。HDE-WUIの機能に加え、ipchainsやルーティングの設定が可能になっている。

Red Hat Linux、TurboLinux、LASER5 Linuxなどに対応している。また12月15日から発売される「TurboLinux Server 日本語版 6.0 SOHO Edition」や、2000年発売予定の「LASER5 Linux 6.0 Rel.2に

は、HDE Linux Controller 1.0がバンドルされる。

ホライズン・デジタル・エンタープライズ
(http://www.hde.co.jp/)

SGIがIRIS Performer 2.3の無償ダウンロードを開始

1999年11月30日

米SGIは、「IRIS Performer 2.3 for Linux」の無償ダウンロードを開始した。IRIS Performer 2.3 for Linuxは、高性能な3Dレンダリングツールキット。

IRIS Performer 2.3 for Linuxは、すでに存在するIRIS Pertformerと完全に互換性のあるAPIを搭載し、ランタイムライブラリ、ファイルローダ、開発用ヘッダファイル、サンプルソースコード、マニュアルなど開発において必要なものすべてを揃えている。また、WebページにはFAQやバグ情報、パッチ情報が掲載され、さらにメーリングリストも用意されている。

SGI (http://www.sgi.com/)

SGIがItaniumプロセッサベースのクラスタリングデモを公開

1999年11月26日

米SGIは、米Oregon州Portlandで開催されたSupercomputing '99 Conferenceで、Intel Itaniumプロセッサベースのクラスタリングのデモを公開した。SGIによると、このデモに使われたソフトはすべて、オープンソースのもので構成されているという。

デモの内容は、Project Trillianによりポーティングされた64ビットの Linuxカーネルを、Itaniumプロセッサマシン上で動作させ、汎用シミュレーションツールキットのCactusをクラスタリングにより実行、結果をIA-32ノード上で動作する数値解析ソフトAmiraにより表示するというもの。

SGI (http://www.sgi.com/)

米IBMがViaVoiceのLinux版をβリリース

1999年11月25日

米IBMは、音声認識ソフトViaVoice SDK for Linux V2.0（Beta）とRun Time Kit for Linux V2.0（Beta）をリリースした。SDKには文法解析ツール、サンプルプログラム、ドキュメントが、Run Time Kitには音声認識エンジンと、セットアップユーティリティが含まれる。

対応するディストリビューションはRed Hat Linux 6.0。そのほかにSuSE、Caldera OpenLinux、Debian/GNU Linuxで動作報告があったという。

SDKとRun Time Kitは、IBMのWebサイトからダウンロードすることができる。また、FAQのページ、メーリングリストも用意されている。

IBM ViaVoice SDK for Linux
(http://www.ibm.com/software/speech/dev/sdk_linux.html)

WordPerfect Office2000 for Linuxのベータテスト受け付けを開始

1999年11月25日

カナダのCorelは、同社のオフィス製品「Corel WordPerfect Office2000 for Linux」のベータテスト受け付けを開始した。同社は、先日Corel LINUX OSをリリースしている。

ベータテスターは、同社のWebページで登録後、抽選で決定される。ベータ1の配布は12月末から1月始め頃の予定。

Corel WordPerfect 2000 for Linuxは、米国で開催されたCOMDEX Fall/ '99にてデモンストレーションされ、Byte.com の「Best of Show Award COMDEX/Fall '99」賞も受賞した。

北米英語版は2000年春に発売予定。

Corel (http://www.corel.com/)

UPS管理ソフトの無償ダウンロードを開始

1999年11月18日

エーピーシー・ジャパンは、「PowerChute plus for Linux」の無償ダウンロードを11月29日より開始する。PowerChute plus for Linuxは、ネットワークに対応した「シャットダウン機能」、「スケジュール運転機能」、「ログファイル機能」を搭載した、Linux用のUPS管理ソフト。

「スケジュール運転機能」では、システムが使われていない時間に、自動的に電源を切り、使用する時間になると自動的に起動するように設定することが可能になっている。また「ログファイル機能」では、定期的に入出力の電源管理や電源周波数などを記録し、トラブルの傾向や問題点を分析することも可能だ。さらに、ネットワーク上に点在するUPSを管理端末から監視することにより、管理性を向上させたとしている。

対応するUPSは同社のSmart-UPSシリーズとMatrix UPS。対応OSは、「TurboLinux 日本語版 3.0／4.0」、「TurboLinux Server 1.0」「Red Hat Linux 5.2日本語版」。なお、PowerChute plus for Linuxを利用するためには、専用ケーブルの「Interface Kit for Linux」が必要。Interface Kit for Linuxの定価は6000円。

エーピーシー・ジャパン
(http://www.apc.co.jp/)

Secure Virtual Network for Linuxを発表

1999年11月18日

ネットワークセキュリティ関連製品を手掛けるチェック・ポイント・ソフトウェア・テクノロジーズは、同社のセキュア・バーチャル・ネットワーク（SVN）を、Linuxプラットフォームへ拡張すると発表した。対応するディストリビューションはRed Hat Linux。

SVNは、VPN（Virtual Private Network）のセキュリティをに強固にしたもの。今回Linuxに対応するのは、「VPN-1／FireWall-1 Version4.1」、「Account Management module」、「VPN-1 SecureServer」、「FloodGate-1」の4製品。

チェックポイント・ソフトウェア・テクノロジーズ
(http://www.checkpoint.co.jp/)

Distribution ▶▶▶

LASER5 LinuxのAlphaプロセッサ版発売

レーザーファイブは、「LASER5 Linux 6.0 Server Edition for Alpha」を1999年12月下旬から出荷すると発表した。同社から発売中の「日本語redhat Linux 6.0 Server Edition」（インテルx86用）をAlphaプロセッサに移植し、さらに独自の機能拡張を加えたもの。

専用の日本語インストーラを採用しており、最小限の入力を行うだけで、高度なセキュリティ機能を持つ安定したサーバ環境の構築が可能だ。サーバ用途のソフトウェアとして、データベースのPostgreSQL（日本語化済み）、日本語全文検索システムのMitakeSearchが付属する。

対象機種は、コンパックの「Linux-ready」シリーズのAlphaプロセッサ搭載機種と、ビジュアルテクノロジーのVT-Alphaシリーズ。

価格は、サポート権なしが1万円、180日間のサポート権付きが4万9800円。サポート内容は、インストールからサーバ設定までを電話、FAX、Webサイトでサポートするというもの。なおサポート権付きの対象は、コンパックのAlphaプロセッサ搭載の「Linux-ready」モデルのみとなっている。

レーザーファイブ（http://www.laser5.co.jp/）

Vine Linux 2.0、3月上旬に発売

技術評論社は、「Vine Linux 2.0 CR Official製品版」（以下Vine 2.0）を2000年3月上旬より発売すると発表した。Vine 2.0は、Red Hat Linux 6.1をベースにProject Vineが開発中のディストリビューション。現在発売中の「Vine Linux 1.1 CR Official製品版」、「Vine Linux 1.1 CR with Wnn6」の登録ユーザーに対するバージョンアップ版の販売も同時に行われる（直販のみ）。

Vine 2.0は、定評の日本語インストーラ、独自開発のツールなどはそのままに、各コンポーネントのアップグレード、デスクトップ環境のKDE／GNOMEの採用、付属アプリケーションの充実をはかったもの。カーネルは2.2.13、Cライブラリはglibc 2.1.2を採用する予定。

Vine 2.0からは、日本語入力システム「Wnn6 Ver.3」を標準装備とした。また商用フォントのDynaFontも1書体増やして、5書体付属とし、多彩な日本語表示／印刷が可能になる。

体験版ソフトウェアなどのパッケージが増えたため、CD-ROMの枚数は4枚になった。価格は未定となっている。

技術評論社（http://www.gihyo.co.jp/）
Vine Linux（http://www.vinelinux.org/）

Storm Linux 2000、北米で発売開始

カナダのStormix Technologiesは、1999年12月15日に北米でStorm Linux 2000の発売を開始した。パッケージはStandard Editionの1種類で、49.95USドル。商用ソフトウェアを除いたダウンロード版は、同社のFTPサイトから取得できる。

Storm Linux 2000は、Debian GNU/Linuxをベースにしたディストリビューションで、カーネルは2.2.13。XFree86 3.3.5、KDE 1.1.2、GNOME（October Release）を採用しており、最新のデスクトップ環境を利用可能。Storm Administration System（SAS）という独自の管理システムを用いている。インストーラはGUIスタイルのものを用いており、初心者にも分かりやすくなっている。

Standard Editionには、Applixware Office 4.4.2（デモ版）、StarOffice 5.1a、Partition MagicSE、BRUバックアップツール（デモ版）、VMWare 1.1（デモ版）、Kriloなどの商用ソフトウェアが付属している。

また同社では、日本語Webページも開設しており、Storm LinuxやSASの説明文を日本語で読むことができる。Storm Linuxの日本語版は、2000年中の発表を目標に準備中。

Stormix Technologies（http://www.stormix.com/）

SuSE Linux 6.3発売開始

ドイツのSuSE社は、1999年12月1日にSuSE Linux 6.3の出荷を開始した。価格は、49.95USドル。CD-ROM 6枚組とDVD-ROM 1枚のパッケージがあり、内容は同じだ。1500以上のソフトウェアパッケージを収録している。

インストールは、YaST2というGUIを用いたインストーラで行う。YaST2はハードウェアの自動認識機能を持ち、初心者でも20分程度でX Window Systemまで含めてインストールが終了するという。

カーネル2.2.13、XFree86 3.3.5、GNOME（October Release）、KDE 1.1.2と最新のコンポーネントを揃えている。商用製品のデモ版として、Vmware 1.1.1、Hummingbird Exceed（Windows用Xサーバ）、Railroad Tycoon II（ゲーム）などを収録している。

SuSE, Inc.（http://www.suse.com/）

Products

CATV／xDSLインターネット接続に使うIPルータ

プレステージ310

最近広まり始めたCATV／xDSLインターネット接続では、専用モデムを介して1台のPCを接続するのが基本だが、LAN環境の普及した今日ではいささか役不足だ。本機を用いれば、こうした環境でも複数台のPCから同時にインターネットにアクセスできるようになる。

製品名　プレステージ310
価格　6万8000円
問い合わせ先　株式会社ブレーン
TEL 042-582-0228
http://www.brain-tokyo.co.jp/

本機を開発したZyXEL Communicationsというメーカーは、汎用CPUを使って高性能なモデムやISDN TAを開発してきたことで知られる。現在はルータも手がけており、本機はMbps単位の高速なケーブル／xDSLモデムをEthernetで接続して使えるルータとして開発された。日本ではブレーンが販売する。

複数のPCでケーブル／xDSLモデムを共有する

本機はIPパケットをルーティングする機器で、**写真1**にあるWAN（インターネット）側とLAN側のEthernetインターフェイスの間で、IPパケットの橋渡しやフィルタリングを行う。インターネット側にはケーブル／xDSLモデムを、またLAN側にはPCなどモデムを共有したいネットワーク機器を接続する*1。ルーティングプロトコルとしてはRIP-1／2をサポートし、RIP-2はマルチキャストにも対応する。

最近の市販ルータとしては一般的な、IPマスカレード相当のIPアドレス／ポート変換機能も本機はサポートしている。SUA（Single User Account）/NATと呼ばれるこの機能により、インターネット側のIPアドレス1つで、LAN側の複数のPCから同時にインターネットにアクセスできる。また同時に、LAN側のサーバをインターネットに公開することも可能だ。本機がポート番号をもとに複数のサーバへパケットを割り振れるので、たとえばメールサーバやWebサーバなどを別々のマシンで運用できる。ポート／アドレス変換機能のせいで動作しないネットワークゲームなども、同様の設定で利用できる。

本機の特徴としては、LAN側イン

*1　CATVやxDSLによるインターネット接続プロバイダでは、複数のPCを接続することを許可していなかったり、別の契約が必要だったりする場合がある。

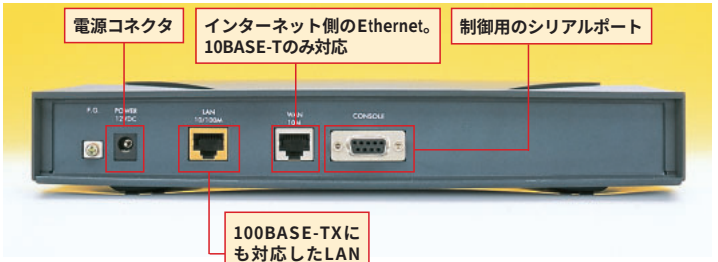


写真1　本機の背面パネル

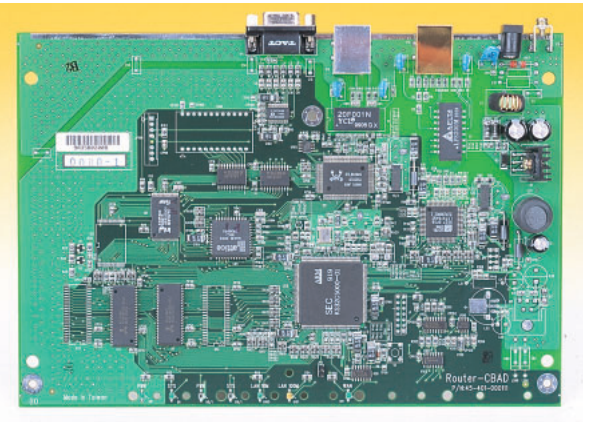


写真2　本機の内蔵コントロール基板
中央やや下にある大きめのチップが、ルーティング処理をつかさどるCPUだ。チップ表面の印刷からARMプロセッサを組み込んだASICと思われる。基板の集積度はそれほど高くなく、もう少し基板のサイズを縮小して小型化できそうだ。

ターフェイスが100BASE-TXであることが挙げられるだろうか。インターネット側は10BASE-Tだが、モデムのスピードを考えると順当といえる。LAN側を100BASE-TX対応機器で統一できるというメリットもあるが、現在では10/100両対応ハブを安価に入手できるのであまり重要ではない。むしろ、インターネット側EthernetのMACアドレスを、LAN側に接続した任意の機器のMACアドレスと同一にできる機能のほうが便利に感じた。ケーブル／xDSLモデムを使ったインターネット接続では、プロバイダに届け出たMACアドレスを持つ機器しか通信できないことがあり、MACアドレスを変更するとプロバイダにその旨を伝える必要がある。しかし、本機ならその手間を省ける。また、接続機器のホスト名で認証される場合でも、本機のホスト名を変更して対処できる。

詳細な設定はテキストベースで行う

本機の各種設定をLinuxマシンから変更するには、**写真1**の制御用シリアルポートにPCを接続してターミナルエミュレータから操作するか、あるいはtelnetを利用する（Windowsマシンからなら GUIの設定ツールが使える）。対話形式とはいえ、設定は**画面1**のようにテキストベースであり、最近主流のWebブラウザによる設定方式に比べると敷居は高い。特にオンラインヘルプが参照できないのが難点といえよう。ただし設定項目自体は豊富で柔軟性はある。

ルータの設定のなかでも、特に複雑なパケットフィルタリングについては、対話形式でよくまとめられているほうだ。フィルタのセット数は12個まで、また1セットのルール数は6個までである。さらにインターネット側／LAN側の入出力ポートそれぞれに、4セットまでフィルタを設定できるので、よほど凝ったフィルタでない限り数は十分だろう。ただ実際に設定してみた限りでは、より具体的なフィルタ設定例のドキュメントがほしいと感じた。Ethernetパケットでのフィルタリングも可能で、特定のMACアドレスだけブロックすることもできる。

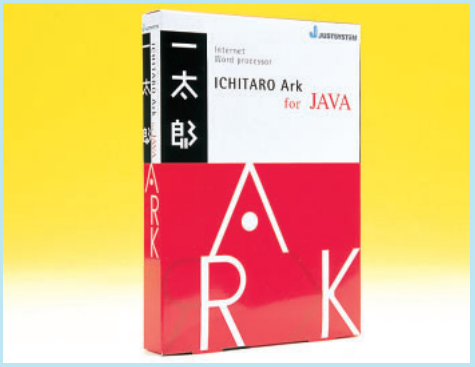
プロバイダごとに仕様の差が大きい現在のCATV／xDSLインターネット接続では、ルータが仕様の差を吸収せざるを得ない面がある。本機ではファームウェアのアップデートがZyXELによって繰り返されており、こうした仕様のバラツキの問題に対処している。また最新ファームウェアではPPP over Ethernetなどの新機能が追加されており、今後のさらなる高機能化にも期待が持てそうだ。

```
Copyright (c) 1994 - 1999 ZyXEL Communications Corp.

            Prestige 310 Main Menu

Getting Started             Advanced Management
 1. General Setup           21. Filter Set Configuration
 2. WAN Setup
 3. LAN Setup               23. System Password
 4. Internet Access Setup   24. System Maintenance

Advanced Applications
 12. Static Routing Setup
 15. SUA Server Setup
```

画面1　システム管理端末（SMT）の画面
対話形式で本機のほぼ全機能を設定できる。シリアル接続の場合、XMODEMで設定ファイルのバックアップ／リストアが可能。telnet接続ならtftpで実現できる。ファームウェアのアップデートも可能

Ethernet	インターネット側： 10BASE-T LAN側： 10BASE-T／100BASE-TX（自動切換え・全二重対応）
IPアドレス割り当て	インターネット側：固定またはDHCPクライアントとして自動設定 LAN側： DHCPサーバとしてIPアドレス配布可能
ポート／アドレス変換	SUA/NAT機能（1対多の変換をサポート）
同時セッション数	256セッション
ファイアウォール	IPおよびEthernetパケットのフィルタリング
ログ機能	LinuxなどUNIX系OSのsyslogdで収集可能
LEDインジケータ	電源／システム状態／LAN側（10／100）／インターネット側
外形寸法（mm）・重量	267（W）×178（D）×37（H）、717g
電源	外付け（ACアダプタ）
付属品	Ethernetツイストペアケーブル（ストレートとクロス各1本）、シリアルケーブル／変換アダプタ、日本語マニュアル2冊など

表1　プレステージ310の主な仕様



一太郎Ark for Java

XMLを基本のデータフォーマットとして採用していることや、Javaで書かれていることなど異色のワープロソフトだが、WindowsやSolarisだけでなく、Java環境が整えばLinux用の本格的な日本語ワープロソフトとしても期待できそうだ。

製品名	一太郎Ark
価格	9800円
問い合わせ先	株式会社ジャストシステム TEL 03-5412-3939 http://www.justsystem.co.jp/

ジャストシステムから12月3日に、インターネットワープロ「一太郎Ark」が発売された。1999年4月より「一太郎Ark for Java Technology Preview」という機能限定版を、無償ダウンロードの形で配布していたが、それを製品化したものである。なお、日本語入力システムは付属していない。

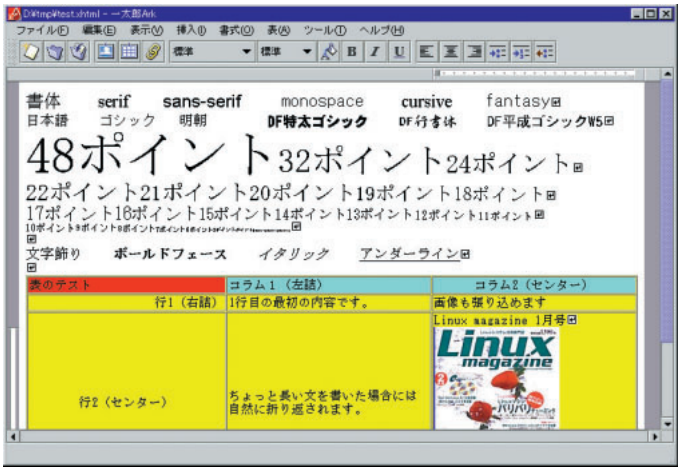
プラットフォームに依存しない100% Pure Javaアプリケーションとなっているため、Windows、Solaris、Linux、Macintoshで同じプログラムが動作可能である。動作環境として「Java 2 Runtime Environment Version 1.2.2以降」が必要である。今回発売された製品では、Windows 98／95／NT Workstation用と、SPARC版Solaris用のJRE（Java Runtime Environment）が付属し、動作確認がなされている。それ以外の動作環境（すなわちLinuxも）についてはサポートを受けられないので注意していただきたい。

まずは、Windows 98にインストールして使い勝手を確認してみた。普通のWindowsアプリケーションと同じように専用インストーラが付属しているので、エクスプローラでCD-ROMのinstall.exeを起動すればよい。インストールが終了すると、デスクトップ上に一太郎Arkのアイコンができるので、マウスでダブルクリックして起動するだけだ。

画面1の上の部分を見てわかるようにメニューやツールバーは、シンプルながらワープロとして標準的なレイアウトになっている。付属のマニュアルはインストール手順だけしか書かれていないが、マニュアルレスですぐに使い始められるだろう。

Linuxで一太郎Arkを動かす

次に、一太郎ArkをLASER5 Linux 6.0で動作させてみた。現時点でLinux用Java 2の環境は正式リリースされていない。従来からLinuxにJavaを移植しているBlackdown Term（Java Linux、http://www.blackdown.org/）から、12月にJava2SDK 1.2.2-RC3がリリースされたところである。LASER5 Linux 6.0に標準で搭載されているJavaは、jdk 1.1でありJava 2ではないため、そのJava2SDK 1.2.2-RC3をインストールした。RPMパッケージで配布されているわけではないため、インストールや環境設定に手間がかかるが、ここでは詳細は割愛する。一太郎Arkのインストールは、

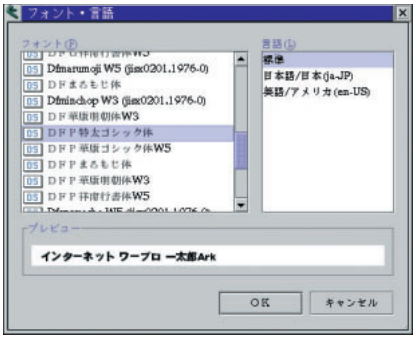


画面1
一太郎Arkで、いろいろ書式を変えて入力してみたところ。メニューやツールバーを見ただけでは、ほかのワープロソフトと大きな違いはない。

画面2
HTMLファイルはURLを指定することで読み込むことが可能でフレームにも対応している。

ark10.zipファイルをunzipコマンドで展開するだけである。

Xのターミナル画面から“java -jar ark10.jar”と入力して起動する。フォントを変えてみたり、文字の大きさをポイント単位で指定したり、表を入れたりして使ってみた。少し不便に感じたのは、文字の大きさの指定で、ツールバーからは「標準、最大、大、やや大、中、やや小、小、最小」しか選べない。HTMLの表現規則に準じたもので、ブラウザの標準の文字に対する相対的な大きさを指定するわけだが、このあたりがワープロというよりHTMLエディタを感じさせる。しかし、メニューから［書式］—［文字サイズ］—［サイズ指定］と選ぶことで、ポイント単位で指定することができる。



画面3
フォントの設定。Linux上でもこのようにOSに登録されたフォントを目で確認しながら選んでいく。

なお、URLを指定してWebサイトやFTPサイトから直接ファイルを読み込むことも可能になっている（**画面2**）。この機能はHTMLエディタとして使う場合に便利だろう。

ファイルの保存は、HTMLとXHTMLというフォーマットを採用し、使用できる文字セットとして、シフトJIS、日本語EUC、ISO2022-JP、Unicodeといったコードに対応しているので、インターネット上でのデータ交換が容易になっている。

一太郎Arkはマルチリンガル対応になっていて、文章中に複数の言語／フォントを混在することが可能だ。フォントはOSに組み込まれたものから選択する（**画面3**）。Linuxでの日本語フォントの表示はギザギザが目立つが、メニューからテキストアンチエイリアスをONにするとなめらかになって読みやすい。

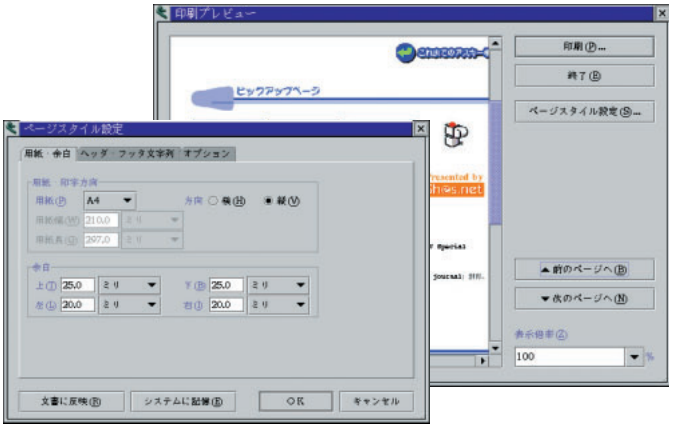
ファイルの読み込みは、一太郎（Ver.5～10）、MS-WORD（Ver.6／95／97／98／2000）、RTFに対応しているが、罫線データの読み込みは一太郎（Ver.7～10）に限られている。ほかのファイルフォーマットからはテキスト部分と文字装飾属性の一部が抽出されるだけなので、残念ながら同じような体裁にはならない。

印刷機能だが、プレビューも可能で、用紙や印字方向、余白も細かく設定もできる（**画面4**）。LIPS系のレーザープリンタを使ってプリントアウトしたところ、文章だけだとすぐに出力されるが、画像含んだHTMLドキュメントは非常に遅かった。プリンタに送られるデータの量が数十Mバイトにもなってしまったのだが、これはLinuxの印刷の仕組みが原因だろう。

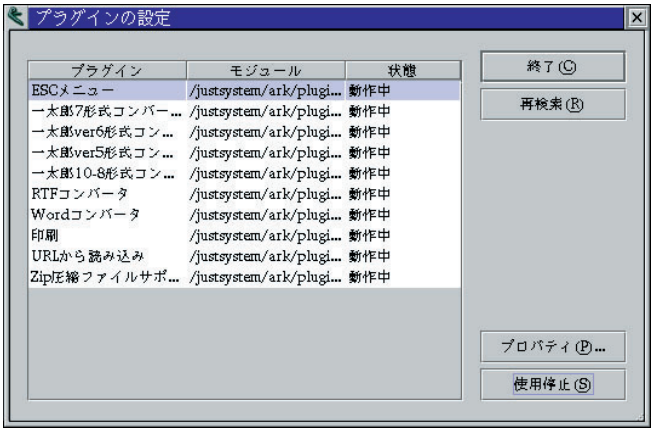
プラグインで機能を拡張できユーザーにAPIを公開予定

一太郎Arkでは、プラグインによって機能を追加できるようになっていて（**画面5**）、同社のWebサイトにあるプラグインのサンプルを利用して自分で拡張することができる。

今回、Linuxでの操作中に一太郎Arkがハングアップすることがあった。Windows上では安定して動作したので、LinuxのJava環境が不完全なためと思われる。いまのところLinuxできちんと使える日本語ワープロがないという状況なので、一刻も早く正式に対応することを期待したい。



画面4
印刷プレビュー画面の表示が行え、ページスタイル設定では用紙の選択や、余白を細かく設定できる。



画面5
プラグインで機能を拡張できる。標準搭載のプラグインは各種フォーマットに対応したファイル読み込み機能が中心だ。

Distribution

新着ディストリビューション

Corel LINUX OS

Linuxといえばサーバと思われがちだが、デスクトップ環境を主なターゲットにしたものも増えてきた。独自の機能を多く盛り込んだデスクトップ向けディストリビューションを紹介する。

Slackware Linux 7.0

昔からのLinuxユーザーなら、一度は使ったことがあるはずのSlackwareが、メジャーバージョンアップをはたして帰ってきた。最新のデスクトップ環境を備えた、老舗のディストリビューションを紹介する。

Linux MLD 4

Windowsがインストールされているマシンに Linuxを追加するのは、けっこう大ごとだ。そんなときに役に立つ、Windows環境との親和性を持ったディストリビューションを紹介する。

Kondara MNU/Linux 1.0

メーカー主体のディストリビューションが増えてきたなか、コミュニティの力強さを感じさせるニューカマーが登場した。「伊達と酔狂」を自称する先鋭的なディストリビューションを紹介する。

Corel LINUX OS

「Corel LINUX OS」（以下、Corel LINUX）は、WordPerfectやCorelDRAWなどのWindowsアプリケーションで知られるCorelによって、Debian GNU/Linuxをベースに開発されたLinuxディストリビューションである。初心者にもわかりやすいグラフィカルインストーラとWindowsとの高い親和性を武器に、主にデスクトップ市場への進出を目指している。

2種類の製品

Corel LINUXは、カーネルバージョンは2.2.12、XFreee86はバージョン3.3.5、標準デスクトップ環境にはKDE 1.1.2を搭載している。ただし、後述するとおり、このKDEにはCorel独自の拡張がなされている。Corel LINUXは、Debian GNU/Linuxをベースに開発されており、パッケージ管理方式に強力なパッケージ管理機構を持つ、deb方式を採用している。

CorelのWebサイトから1999年11月15日より、無料ダウンロード版が公開されている。また、同11月30日よりスタンダード版（59.95USドル）とデラックス版（89.95USドル）が発売された。ただし、デラックス版については北米のみの販売となっている。国内販売は、メディアヴィジョン（http://www.mvi.co.jp/）が行う。

3ステップしかない簡単インストーラ

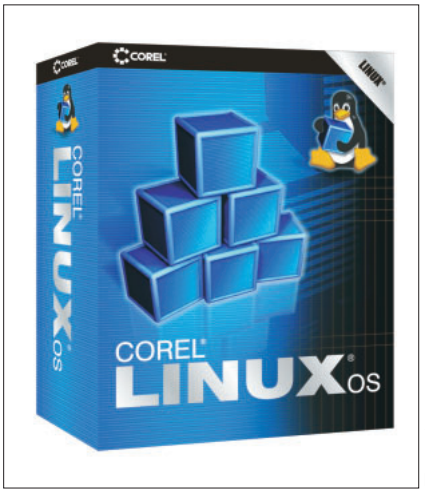
Corel LINUXで、まず目を引くのがグラフィカルインストーラ「Corel Install Express」である。ブート後にいきなり起動するグラフィック画面もさることながら、ユーザーの入力フェーズを、ユーザー名、インストールタイプ、パーティション設定という、3ステップだけに絞り込んでいる。あまりに設定項目が少ないため、かえって心配になってしまうユーザーもいるかもしれないが、試用した限りではハードウェアの認識やDHCPを利用したネットワーク設定など問題はほとんどなかった（ただし、キーボードが101キーボードに誤設定されてしまったが）。また、ブートローダはMBRに書き込まれるが、既存の環境もしっかり起動リストに登録されており、使用上の問題はなかった（しかも、起動リストには、「MS Windows」や「TurboLinux」といった具合に既存環境の名前まで正確に認識されていた）。

独自拡張されたKDE

Corel LINUXは、標準のデスクトップ環境としてKDE（バージョンは1.1.2）を採用している（**画面1**）。ただ、Control Centerにsamba関連の設定項目を追加するなど、細かいところがいろいろカスタマイズされている。また、次のようなCorel独自のツールがいくつか搭載されている。

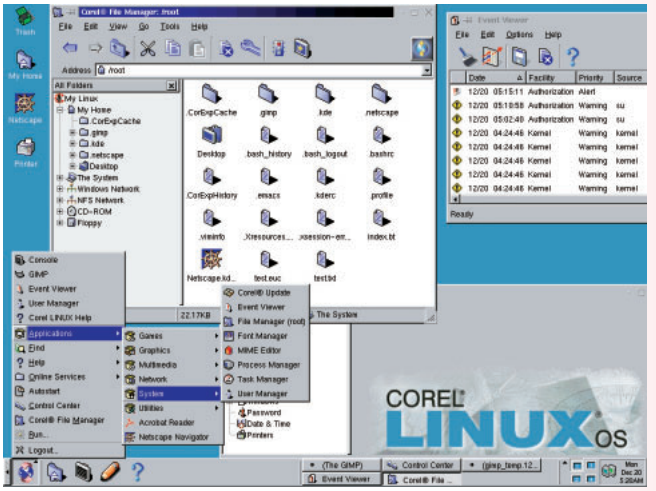
■Corel File Manager

KDEの標準ファイルマネージャであるkfmは、ローカルファイルとリモートファイルへシームレスにリソースアクセスできるという特徴を持っている。しかし、Corel LINUXでは、このkfmではなく「Corel File Manager」という独自のファイルマネージャを標準で使用している（**画面2**）。このCorel



Corel Linux OS
スタンダード版：59.95USドル
デラックス版：89.95USドル
Corel Corporation
http://linux.corel.com/

画面1　Corel Linuxのデスクトップ環境
Corel Linuxのデスクトップ環境は、メインメニューなどがよりWindowsに近い感じだ。また、アプリケーションもKDEの標準環境より少なくあっさりとしているが、かえって使いやすいような印象を受ける。アイコンも上品だ。



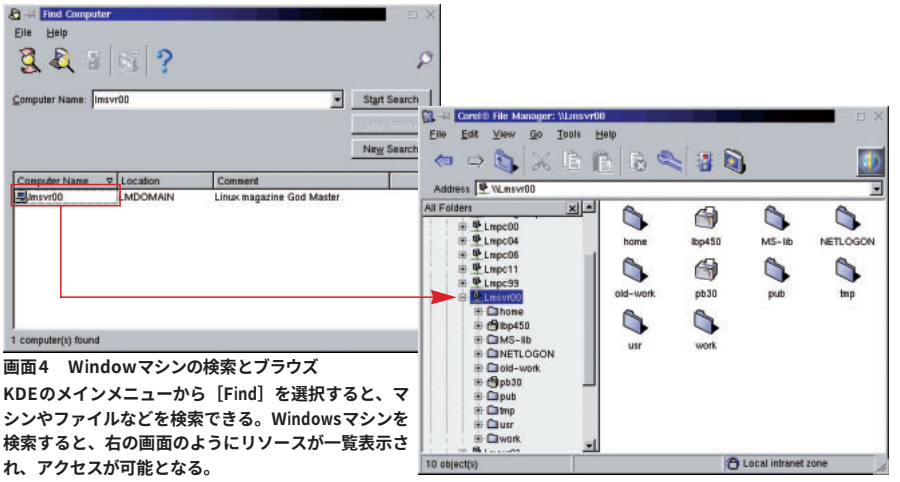
File Managerも、シームレスなリソースアクセスが可能なうえ、Windowsネットワークのブラウズも可能となっており、よりWindowsのExplorerに近い操作性を持っている。

■Corel Update

Corel Updateは、Corel LINUXのパッケージ管理を行うGUIツールである（**画面3**）。deb方式のパッケージ管理コマンドであるdpkgやapt-getなどの機能を利用して、CD-ROMや指定したFTPサイトやWebサイトからdebファイルをダウンロード／インストールすることで、Corel LINUXを常に最新の状態に保つことができる。基本的なスタンスは、Windowsの「Windows Update」とよく似た機能といえるだろう。

Windowsとの親和性

Corel LINUXのもうひとつの特徴が、Windowsとの親和性である。たとえば、Corel File Managerの項でも述べたとおり、smbクライアントの機能を利用して、ネットワーク上にあるWindowsマシンの検索、表示、アクセスがGUIベースで行えるようになっている（**画面4**）。これにより、Windowsネットワークとシームレスな環境を構築できる。これもWindowsが圧倒的なシェアを誇るデスクトップ市場へ進出していくためには、必須の要件といえるだろう。



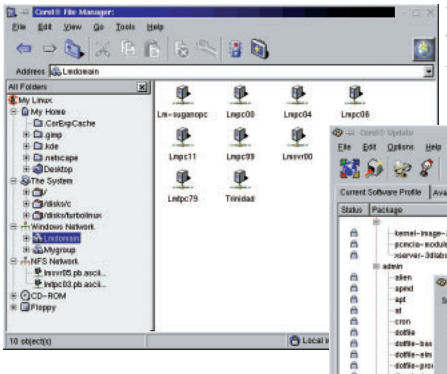
画面4　Windowマシンの検索とブラウズ
KDEのメインメニューから［Find］を選択すると、マシンやファイルなどを検索できる。Windowsマシンを検索すると、右の画面のようにリソースが一覧表示され、アクセスが可能となる。

日本語は？

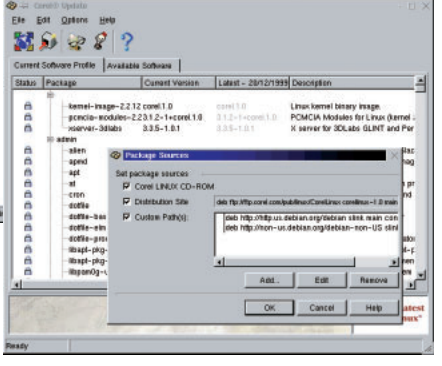
Corel LINUXは海外で作成されたディストリビューションであるため、日本語環境についてはまったく考慮されていない。しかし、dselectのFTPアクセス機能を利用して、Debian GNU/LinuxのFTPサイトから日本語表示や入力に必要なパッケージ（Canna、kinput2、XEmacsなど）をダウンロード／インストールしたところ、まったく問題なく日本語環境を構築することができた（**画面5**）。これも、強力なパッケージ管理機構を持つdeb方式だからこそ成せるワザといえるかもしれない。

将来が楽しみなディストリビューション

全体的に見るとCorel LINUXはよくまとまっており、完成度は高い。デスクトップ市場への進出をふまえたWindows類似の機能も、ある程度の成果を出しているように思う。さらに、Corelには、WordPerfectやCorelDRAWといった優れたアプリケーションがある。そういったアプリケーション群との連携など、Corel LINUXの今後の動向に期待したい。



画面2　Corel File Manager
Windowネットワークの表示など、kfmに比べ、よりExplorerに近い操作性を持つ。



画面3 Corel Update

画面5　日本語対応となったCorel LINUX
debパッケージのインストールにより、日本語の表示／入力が「とりあえず」できるようになった。

Corel LINUX OSのインストール

CD-ROMブートが可能なマシンでは、Corel LINUX OSのCD-ROMをドライブに入れて起動します。CD-ROMブートできない場合は、起動用フロッピーを以下の手順で作ります。

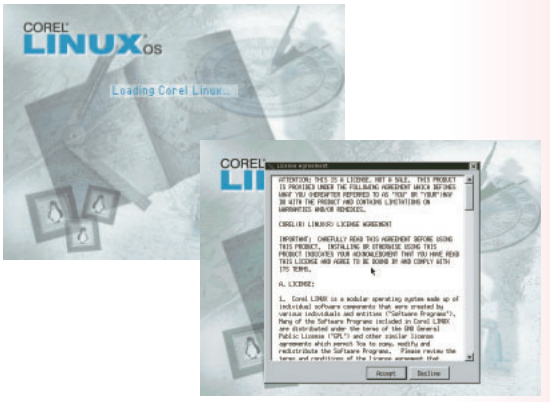
（1）Windows 9x／NTが起動しているPCのドライブに、Corel LINUX OSのCD-ROMを入れます。（2）自動的に起動するCorel Linux Autorunウィザードで起動用のフロッピーを作成します。（3）自動的にAutorunウィザードが起動しない場合は、CD-ROMのルートディレクトリにあるAutorun.exeを、エクスプローラから起動します。

1 インストーラの起動

CD-ROM、または上記の手順で作製したフロッピーから起動すると、Corel LINUX OSの起動画面が表示されます。ここで、ハードウェアの自動認識を行っているので、多少時間がかかります。

その後、ライセンス関係のメッセージが表示されますので、［Accept］ボタンをクリックします。

Corel LINUX OSのブートローダは、MBRに書き込まれます。すでにほかのLinuxディストリビューションがインストールしてあって、LILOがMBRに書き込まれている場合や、MBRに市販のブートセレクタを入れている場合には、この点に注意してください。



2 ユーザー名の入力

最初はユーザー名の入力をします。ふだんCorel LINUX OSを使う際のユーザー名（User id）を指定し、［Next>］ボタンを押します。

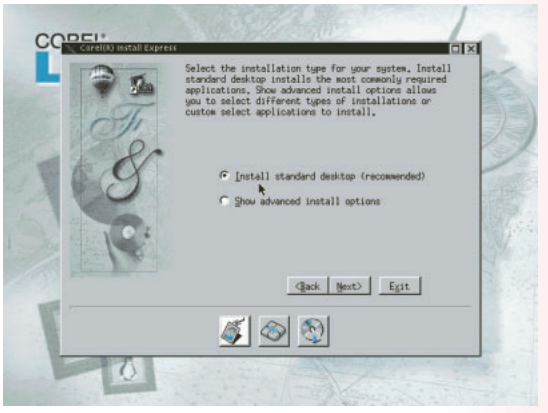
この時点では、パスワード設定を行いませんが、インストールが成功して最初にログインする際に、設定を求められます。

なお、インストーラ画面下部に並んでいる3つのアイコンは、左から順に［User id］、［Package］、［Partition］を表しています。設定が終了した項目は、色が白に変わります。また、このアイコンをクリックすると、設定が終了した項目に戻れますので、あとからの修正が可能です。

3 インストール内容の設定（1）

どのようなパッケージをインストールするか設定します。とりあえずCorel LINUX OSを試してみたいなら、デフォルトの［Install standard desktop］のままでいいでしょう。そのまま［Next>］ボタンを押します。

インストール内容を自分で選びたい場合は、［Show advanced install options］を選択して［Next>］ボタンを押します。



インストール内容の設定（2）

1つ前の画面で［Show advanced install options］を選んだときは、この画面が表示されます。デフォルトの［Desktop］に加えて、ソフトウェア開発用のツールなどを含んだ［Desktop Plus］や［Server］、個別のアプリケーションを自分で選択できる［Custom］を選択できます。

4

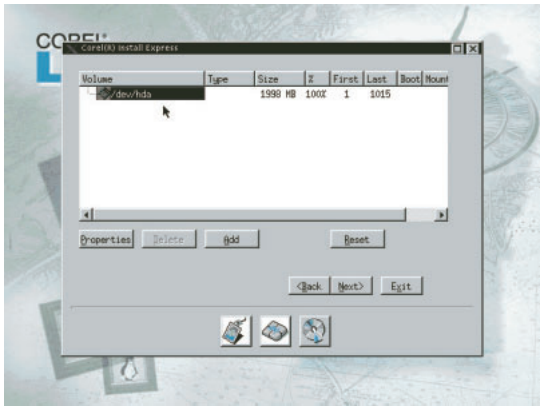
パーティションの設定（1）

Corel LINUX OSで使用するパーティションを設定します。Corel LINUX OS専用のマシンとして使うなら、［Take over disk］を選択し、確認のダイアログで［Yes］ボタンをクリックします。ただし、これを選択すると、ハードディスクの内容はすべて消去されます。Windowsなど、ほかのOSとのデュアルブートを行うなら、絶対に選択してはいけません。

ハードディスクに未使用の領域があり、そこにCorel LINUX OSをインストールするなら、［Use free disk space］を選択します。

現在のパーティション設定を変更してCorel LINUX OS用のパーティションを作成するなら、［Edit partition table］を選びます。

5



パーティションの設定（2）

1つ前の画面で［Edit partition table］を選ぶと、この画面が表示されます。ここでパーティションのサイズ変更や削除、追加を行います。サイズ変更や削除したパーティションの内容は失われます。本当に消してもいいか、よく確認してから作業を行ってください。

6

インストール開始!

画面下のアイコンが、3つとも白くなっていれば、インストール準備は完了です。［Install］ボタンを押せば開始します。［Scan for bad blocks while formatting］にチェックを入れておくと、ハードディスクのフォーマット時に不良ブロックの検査をしてくれます。多少時間はかかりますが、チェックしておいたほうがいいでしょう。

進行状況は棒グラフで表示されますので、あとは100%になってインストールが完了するまで待つだけです。

7

Slackware Linux 7.0

Slackware Linuxの最新版「Slackware Linux 7.0」が、1999年10月29日にリリースされた。FTPで入手できるほか、Walnut Creek（http://www.cdrom.com/）からCD-ROM4枚組の公式パッケージを39.95USドルで購入することもできる。

Slackwareは、1992年に最初のバージョンが発表された老舗のディストリビューションだ。Slackwareと組み合わせて使うJE（Japanese Extensions）パッケージが日本の有志によって開発されたこともあり、数年前は日本でデファクトスタンダードとなっていた。複雑なパッケージ管理システムを持たず、tarとgzipでアーカイブしたバイナリパッケージを採用するSlackwareは、シンプルな構成が特徴で、現在でも、ベテランユーザーを中心として根強いファンが多い。また、Slackwareをもとに、日本語化したディストリビューションPlamo Linuxも作られている（http://www.linet.gr.jp/~kojima/Plamo/）。

しかし、前作のSlackware Linux 4.0では、カーネル2.2を採用したものの、カーネルと並んでシステムを支えるCライブラリには、従来のlibc5を採用し、当時主流になりつつあったglibc2はランタイムサポートのみにとどまっていた。

このように、他の主力ディストリビューションに後れをとっていた感のあるSlackwareだったが、今回リリースされた7.0では、最新のカーネル2.2.13、glibc2.1.2で足回りを固め、X Window System環境も、XFree86 3.3.5、KDE 1.1.2、GNOME 1.0.53と最新バージョンを揃えた（**画面1**）。

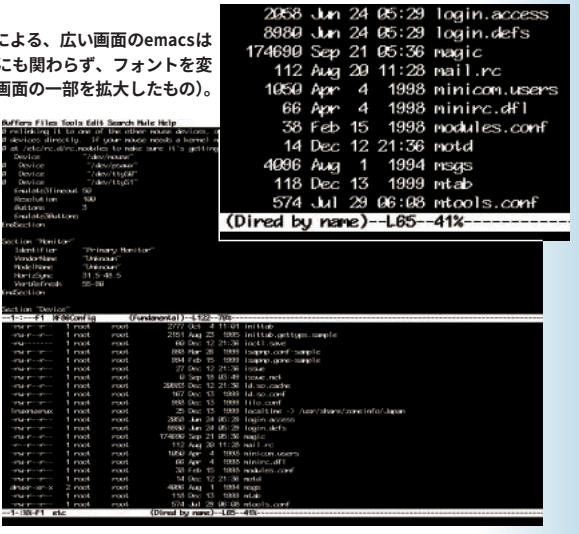
統合デスクトップ環境を提供するKDEとGNOMEは、ごく標準的な構成となっており、特別な追加機能は盛り込まれていないようだが、オリジナルならではの安心感と美しさを持つ。

インストーラは、従来のバージョンと同様のテキストベースGUIのものだが、インストール途中でフォントを選ぶ画面があるのには驚いた。Slackware 7.0では、カーネル2.2で採用されたフレームバッファコンソールをデフォルトで利用しており、コンソール画面にも関わらずフォントを変えたり、高解像度の広い画面を利用できるのだ（**画面2**）。まるで、DOS/Vで流行したHiTextやV-Textのようだ。仮想コンソールとあわせれば、わざわざXを起動しなくとも、テキストベースのアプリケーションをさくさくと使える。Linux本来の“軽さ”を再認識できるだろう。

現在は、英語版しかリリースされていないが、日本語化パッケージの発表を期待するのは私だけではあるまい。腕に覚えのある方は、日本語化されたアプリケーションを組み込んでみてはいかがだろうか。

Slackware Linux 7.0
39.95USドル
http://www.slackware.com/

画面1
XFree86は最新の3.3.5なので、多くのビデオカードに対応する。デスクトップ環境は、KDEとGNOMEが利用できる。



画面2
フレームバッファコンソールによる、広い画面のemacsは快適だ。さらに、コンソールにも関わらず、フォントを変更することもできる（右上は画面の一部を拡大したもの）。

Linux MLD4

「Linux MLD4」（以下、MLD4）は、メディアラボから1999年12月10日より発売されたLinuxディストリビューションである。本製品は、Windowsとの親和性に優れた簡単インストーラが特徴である。価格は8800円で、旧バージョンのMLD、および同社のもうひとつのLinuxディストリビューション製品「Live Linux」の登録ユーザーなら、優待価格の3900円で購入できる。

簡単インストール

MLD4の最大の特徴は、インストールが非常に簡単なことである。インストーラCD-ROMをWindowsマシンに挿入すると、Auto Runでインストール画面が起動する（**画面1**）。ここで、次の2つのインストールタイプのうち、いずれかを選択する。

・標準版

本格的にLinuxを利用する人向けで、KDE、Wnn6、XEmacsといったアプリケーションがインストールされる。空きディスク容量として650Mバイト以上が必要。

・コンパクト版

空きディスク容量があまりない人や、とりあえず試してみたいという人向けで、fvwm2、Canna、Netscapeなどがインストールされる。空きディスク容量として、250Mバイト以上が必要。

ここで、インストールタイプとディスク容量を選択すると、あとはインストール終了まで何もする必要はない。

さらに、MLD4はインストールだけでなくアンインストールも簡単である。Windowsのコントロールパネルの［アプリケーションの追加と削除］アプレットから、［Linux Media Lab Distribution 4.0］を選択し、［追加と削除］ボタンをクリックするだけである（**画面2**）。

簡単インストールのワケ

MLD4のインストールがこれほど簡単なのには、次のような理由がある。

■パーティションの切り直し不要

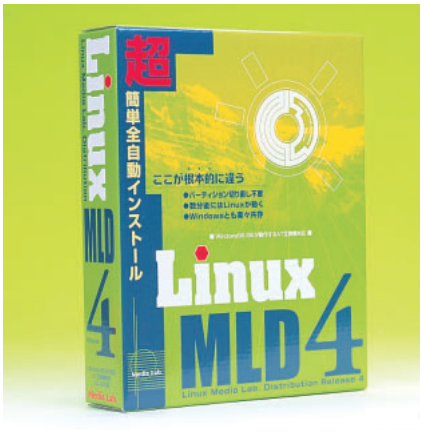
MLD4は、Windows 95/98がインストールされているFAT領域にそのままインストールすることができる（**画面3**にあるようにWindowsからは1つの大きなファイルに見える）。パーティション操作は難しい作業なので、このメリットは大きいだろう。

■レジストリからの設定情報取得

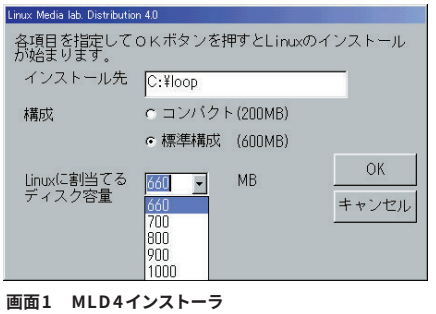
MLD4のインストーラは、インストール時にWindowsが保持しているデバイス情報やネットワーク設定などをWindowsのレジストリファイルから取得するようになっている。そのため、インストールの際、まったく設定項目の入力が必要ないのだ。

起動方法

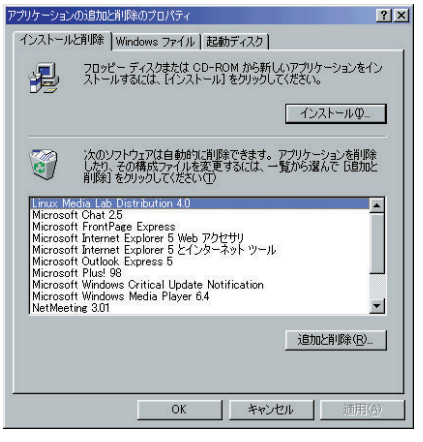
MLD4は、LOADLINという、DOS上で動作するブートローダを利用してカーネルをロードしてから、FAT上に作ったMLD4用の領域（前述の1つの大きなファイル）を「ループバックデバイス」という仕組みによってマウントし、起動する。



Linux MLD4
8800円
メディアラボ株式会社
03-5294-7255
http://www.mlb.co.jp/



画面1　MLD4インストーラ
インストールは簡単で、インストールタイプとMLD4用の領域のサイズを指定するだけだ。



画面2
Windowsの［アプリケーションの追加と削除］アプレット

ここまで読むと難しいように思えるが、実際はMS-DOSモードからlinux.batというファイルを実行するだけである。ショートカットを作れば、ダブルクリックだけで簡単だ。

Linuxとしての特徴

MLD4は、カーネルに2.2.12、XFree86は3.3.5-XTTを搭載しており、KDE、XEmacs、Netscapeといった約400のパッケージがバンドルされている。また、Red Hat Linuxをベースにしており、パッケージ管理にはRPM方式を採用している。そのため、rpmコマンドを利用すれば、アプリケーションを追加することも可能である。

■KDEを採用したデスクトップ

MLD4は標準デスクトップ環境として日本語化されたKDE 1.1.2を採用している（ただし、インストール時に［コンパクト版］を選択したときは、fvwm2）。さらに、付属の「フリーソフトパッケージ」CD-ROMからGNOME、Window Maker、Enlightenmentといった、おなじみのウィンドウマネージャをインストールすることも可能である。

■日本語環境もバッチリ

MLD4では日本語表示／入力の基本項目についてはあらかじめ設定されているため、インストールしてすぐに日本語入力が可能となっている。また、商用かな漢字変換ソフトウェア「Wnn6 Ver.3」や商用TrueTypeフォントである「DynaFont」が5書体バンドルされている。

MLD4はインストールと起動は少々特殊だが、起動後はまったく「普通の」Linuxとして利用できる。速度面に関してもほとんど気にはならないレベルである。さらに、/bootfsというディレクトリからWindowsのファイルへもアクセスできる。

Windowsが動くマシンならほぼ問題なし

MLD4を動作させるには、CPUはPentium以上、メモリは16Mバイト以上（推奨64Mバイト以上）のWindows 95/98が動作するPC/AT互換機となっている。また、インストールには空きハードディスク容量は250Mバイト以上必要となる（ただし、圧縮ドライブは不可）。そのほか、USBやFM/TV機能のサポートなど、対応デバイスは他のディストリビューションと遜色はない。

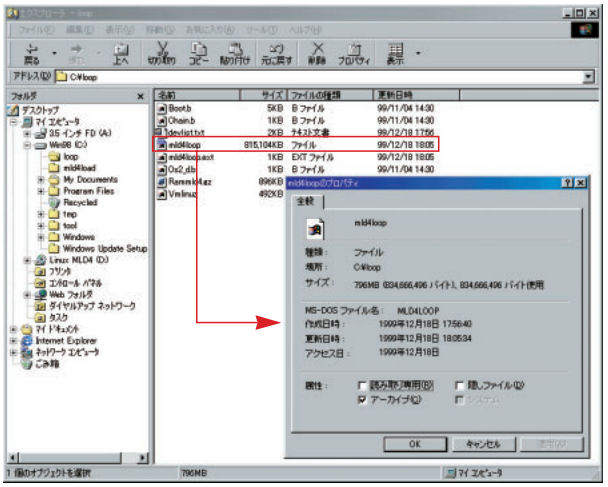
マニュアルもまずまず

MLD4には190ページ程度のマニュアルが1冊添付されている。インストールにあまりページを割く必要がないためか、KDEの使い方、MLD4固有の部分の解説はもとより、簡単なコマンドリファレンスやX Window Systemやシステム管理の簡単な説明があり、ページ数のわりには詳しい内容となっている。

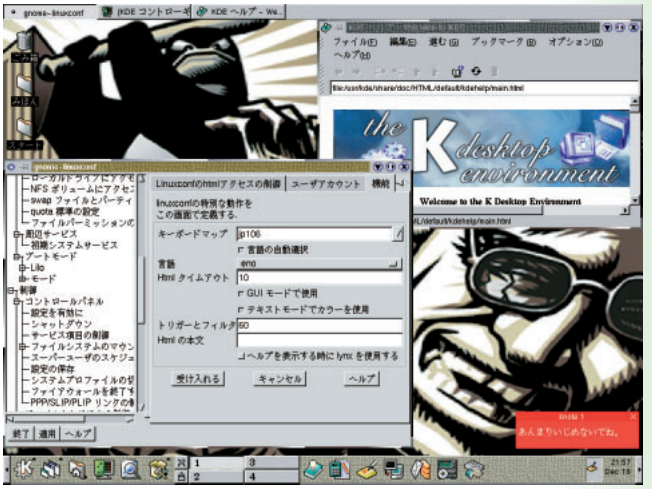
残念ながら、開発元のメディアラボでは個別サポートは行われておらず、メーリングリストが唯一の情報源となっている。入会方法などは、メディアラボのWebサイトを参照してほしい。

初心者は「買い」

インストールや起動方法だけ見ると、MLD4は異色のディストリビューションといえるかもしれない。しかし、このインストーラは、そんなことを気にさせないほどの素晴らしい出来である。ビクビクしながら、パーティション設定やブート設定をしている初心者なら、間違いなく「買い」のディストリビューションといえるだろう。



画面3 Windowsから見えるMLD4の領域



画面4　MLD4の標準デスクトップ環境、KDE 1.1.2

Kondara MNU/Linux 1.0

デジタルファクトリジャパンから1999年12月1日に、Kondara MNU/Linux 1.0（以下Kondara）が発売された。開発は、Kondara Projectが行った。米国Red Hat社の開発版バージョンである「Rawhide」をベースに、日本語対応などの拡張を加えたディストリビューションである。

パッケージは1種類で、インテルx86 PC用、Alphaマシン用のバイナリCD-ROM各1枚とソースCD-ROMの計3枚が含まれている。赤が基調の派手なパッケージを開けると、いきなり熱いメッセージが目に入るだろう。

価格は6800円で、商用のDynaFont（Dynalab）が5書体と90日間3件までのサポートが含まれている。サポートの内容は、インストールからX Window System起動までを、FAXまたは電子メールでサポートという標準的なものだ（インテル版のみ）。

Rawhide＋α＝Norika

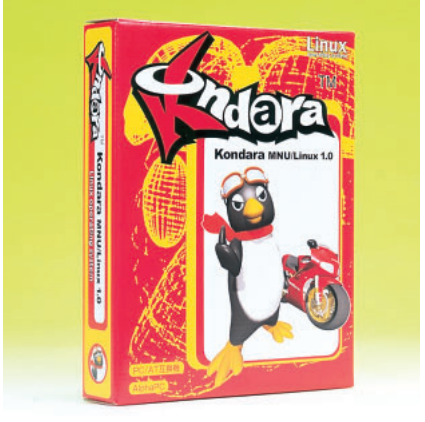
米Red Hat社のディストリビューションは、6.1が「Cartman」、6.0が「Hedwig」というように、バージョンごとにニックネームがつけられているが、開発版は常にRawhideを名乗ることになっている。Rawhideは、万人向けとはいえないが、最新の技術を追い求める先鋭的ユーザーには魅力的だ。Kondara MNU/Linux 1.0は、Red Hat Linux 6.1に相当するRawhideをベースにしている。そのためRed Hat Linuxと互換性があり、Red Hat Linuxに対応した商用アプリケーションが動作する。

余談だが、Red Hat社にならってRPM系の各ディストリビューションもニックネームをつけている。Kondara MNU/Linux 1.0は「Norika」だ。

Anacondaとkudzuを採用

Kondaraでは、Red Hat Linux 6.1と同様に、「Anaconda」というグラフィカルなインストーラが採用されている。起動時にビデオカードの種類を自動で識別し、適切なビデオカード用のXサーバを起動する。画面の左側に常に説明が表示されているので（**画面1**）、各画面で何をするのか確認しながらインストールが行える。手持ちのビデオカードがAnacondaに対応していなかったとしても、テキストベースのインストーラからインストール可能だ。

残念ながらAnacondaには不具合がいくつか見つかっている。編集部内で試した範囲では、ネットワークカードの認識に失敗することが何回かあった。Kondaraには、ハードウェアの自動検出／設定プログラム「kudzu」も採用されている。システムを起動する際にたくさんのメッセージが表示されるが、その中に「Checking for new hardware」という行が見つかるはずだ。これがkudzuの出力だ。ビデオカードやハードディスクなどを取り換えると、上記のメッセージの後、ハードウェア環境の変更を検出したというメッセージと新しい設定内容が表示される。その後古い設定を消すか、残したうえで使用しないことにするか選べばよい。もちろん「再起動しますか？」などというメッセージが出ることもなくマシンが起動する。初めて体験するとけっこう感動だ。PCを自作した結果、余ってしまったハードウェアを持っているなら、ぜひ一度試してみることをお勧めする。



Kondara MNU/Linux 1.0
6800円
デジタルファクトリジャパン株式会社
http://www.digitalfactory.co.jp/

画面1
GUIインストーラAnaconda。左側のヘルプを参照しながらインストールが行える。

最新のコンポーネントを収録

主要コンポーネントには、カーネルが2.2.13、Cライブラリがglibc 2.1.2と最新バージョンが採用されている。アプリケーションは、極端に不安定なもの以外は最新のものを採用するというKondara Projectのポリシーに基づいて選択されている。たとえばNetscape Communicatorは4.7、Gimpは1.1.11が収録されている。

またXFree86は、3.3.5にX-TTのパッチを当てたものが用いられている。ウィンドウマネージャやデスクトップ環境は、KDE、WindowMakerなど多数含まれているが、推奨されているのは、Enligtenment（**画面2**）だ。

日本語環境への対応が十分でないとの判断で、GNOMEは収録されていない。だが1999年10月末にKondara Project内で「Gnome野郎Aちーむ」が結成されており、GNOME対応が開始された。1カ月後の11月には、早くもその成果がKondara MNU/Linuxのスナップショットに取り込まれている。

常に継続する開発

Kondaraは、Kondara Projectというコミュニティベースで開発されており、12月1日のパッケージ発売以降も開発は続けられている。バグの修正や新たに見つかったセキュリティホールへの対応、パッケージのアップデートの情報は、Kondara ProjectのWebサイト（http://www.kondara.org/）を通じてアナウンスされている。特にセキュリティホールへの対応は非常に迅速で、国内ではトップクラスだ。セキュリティ問題への感度が高いということは、Kondaraが個人利用だけでなく、業務用途にも適していることを示している。

またWebサイトには、掲示板が用意されており、プロジェクトのメンバーに直接バグレポートを出したり、質問したりできる。よほど身勝手な質問でなければ、答えがもらえるだろう。さらに「バグ宙太」という名前のバグトラッキングシステム（BTS）（**画面3**）も提供されている。BTSは、多くのユーザーから報告されるバグレポートを効率よく扱うシステムで、ここを参照すれば、すでに報告されているバグの一覧、対処方法の確認などができる。バグレポートの前にはバグ宙太を眺めておこう。

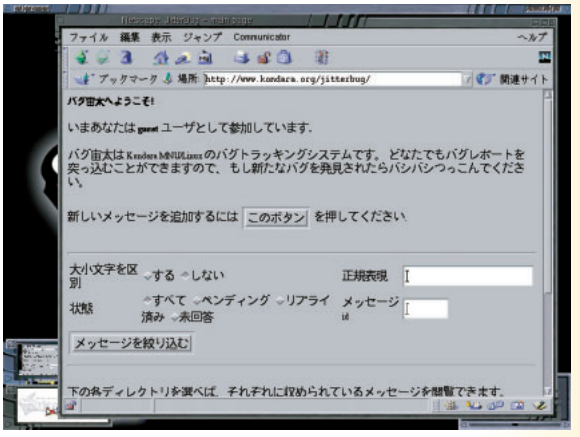
目指すは「ディストリビューションの星」

Kondaraには、商用のアプリケーションは付属していない。そのため、LinuxでOffice系のソフトを使ってみたいとか、賢いかな漢字変換システムを利用したいと考えているユーザーには適していないだろう。どちらかというと開発者指向のユーザー、しかも積極的に新しいソフトにチャレンジしていくユーザーにはうってつけのディストリビューションだ。

「コンダラ」とは、TVアニメ「巨人の星」のオープニングで主人公が引っ張っているローラーのことだ。「思い込んだら」という歌詞を「重いコンダラ」と聞き違えたために、そういうことになってしまった。また「MNU」は、「Mount is Not Umount」の略で、「ムニュウ」と読む。「ペンギンに触ったときの感じ」なのだそうだ。

Kondara ProjectのWebサイトは、このようなウィットにあふれている。一見ふざけているようにも見えるが、このような言葉遊びは、オープンソースのコミュニティと切り離せないものだ（GNUはGNU is Not Unixの略なのだ！）。Kondara MNU/Linuxには、そのようなコミュニティの雰囲気がよく表れている。見た目やノリは軽めでも、中身は正統派ストロングスタイル、目指すは、もちろん「ディストリビューションの星」だ。

画面2
Kondara Project推奨ウィンドウマネージャ、Enlightenmentのデスクトップ。No.1ペンギンは所帯持ちだった!?



画面3
バグトラッキングシステム「バグ宙太」。バグ報告の前に、こちらを確認しよう。

ツール、グラフィックス、サウンド、ゲーム……ぜんぶまとめて

フリーソフト大全 怒濤の100本!

「Linux 主な用途は インストール」(詠み人知らず)
という川柳があるように、新しいディストリビューションが出ると、ついインストールしてしまうという人も多いという。だが、ただディストリビューションをインストールしただけではつまらない。インストール後に好みの環境を作り上げていくことこそ、正しいLinuxの楽しみ方だ。今回紹介するフリーソフトを活用して、自分だけのオリジナル環境を目指そう。お楽しみはインストールのあとからなのだ。

文:出井 一、中井幸博、塙 雅典、山岸典将
Text : Hajime Dei, Yukihiro Nakai, Masanori Hanawa, Norimasa Yamagishi

photo : Takashi Shinohara(Dee)

今回紹介するソフトウェアの中には、Xで動作するものやコンソールで動作するもの、別途ライブラリやスクリプト言語が必要なものがある。ソフトウェアごとの動作条件は、以下のアイコンで示しているので、パッケージに同梱のドキュメントをよく読んで、動作環境や利用条件を確認してほしい。

フリーソフトウェアは、作者の善意から提供されていることを理解しよう。

X	X Window System上で動作する
コンソール	コンソール上で動作する
GTK+	GTK+ツールキットが必要
GNOME	GNOMEライブラリが必要
Qt	Qtライブラリが必要
Tcl/Tk	Tcl/Tk実行環境が必要
Perl	Perl実行環境が必要
Python	Python実行環境が必要
Mesa	OpenGLライブラリMesaが必要
日本語可	日本語が利用可能、もしくは日本語化パッチが存在する

フリーソフトをインストールするには

Linux上で動作するソフトの配布形態は、ソースファイルやドキュメントををtarで1つのファイルにまとめ、gzipなどで圧縮する昔ながらのソース配布（「tarボール」などと呼ばれる）と、Red Hat系ディストリビューションなどでおなじみのパッケージ配布に大別できる。

そこで、まずは代表的なパッケージシステムのRPM（Red Hatパッケージマネージャ）を利用したバイナリパッケージのインストール方法について解説する。続いて、ソースパッケージを利用したリビルドの方法、tarボールのソースからコンパイルしてインストールする方法、パッチファイルを利用したソースの修正方法などについて順番に説明する。

■RPMを利用したバイナリパッケージのインストール

RPMのバイナリパッケージは、拡張子が「i386.rpm」や「noarch.rpm」などのファイルだ。これらのファイルには、ソースからビルドされた実行ファイルなどが含まれているので、そのままインストールするだけで、すぐにソフトを実行可能だ。インストール時にはライブラリなどの依存関係をチェックしてくれるし、後でパッケージをアンインストールすることも容易なので、プログラミングの知識がない人でも安心して利用できる。

●rpmコマンドを利用する方法

RPMのパッケージのインストール、アンインストール、中身の確認などを行うには、コマンドライン版のrpmコマンドか、GUI版のGnoRPMやKpackageなどを用いる。まずは、rpmコマンドの使い方について簡単に説明しよう。

(1) インストール済みでないか調査

インストール作業を始める前に、すでにそのソフトがインストールされていないか確かめてみよう。たとえば、パッケージ「hoge」がインストールされているかどうかは、

```
$ rpm -q hoge
```

で調べられる。インストール済みの場合は「hoge-1.2.3-4」のように、バージョンとリリース番号付きのパッケージ名が出力される。一方、インストールされていない場合は「パッケージhogeはインストールされていません」などと表示される。

インストール済みの全パッケージのリストを表示するには、パッケージ名を省略して、-qaオプションを指定すればいい。パッケージ名の一部しか分からないときは、rpmコマンドの出力を、以下のようにパイプでgrepに接続して、文字列検索するといいだろう。

```
$ rpm -qa | grep hoge
```

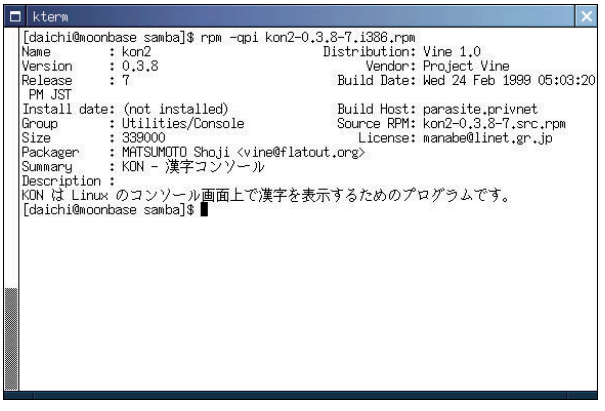
(2) バイナリパッケージを取得

インストールするバイナリパッケージを、WebサイトやFTPサイトから取得しよう。ファイル名は「hoge-1.2.3-4.i386.rpm」のように、「パッケージ名－バージョン番号－リリース番号.アーキテクチャ.rpm」という書式で統一されている。

インストール前のパッケージの情報は、-qpオプションに情報の種類を指定するオプションを付けたrpmコマンドで調べられる。たとえば、

```
$ rpm -qpi hoge-1.2.3-4.i386.rpm
```

と-iオプションを付けると、そのパッケージの概要が表示される（**画面1**）。このほか-lオプションを付けると、パッケージに含まれる全ファイルのフルパスが表示されるし、-Rオプション（大文字）では依存するパッケージ名などが表示される。どのようなファイルがインストールされるか、必要なライブラリは何かといった点を前もって調べられるわけだ。



画面1 パッケージの概要を表示する

(3) インストールとアンインストール

パッケージのインストール、アンインストールは、さまざまなディレクトリのファイルを作成／削除する都合上、スーパーユーザー（root）で行わなくてはならないパッケージが多い（一般ユーザーのままインストールできるよう工夫されたものもある）。

インストールの際は、インストール用の-iオプションではなく、更新用の-Uオプションを使うのがお勧めだ。古いパッケージがインストールされているかどうかを調べて、新規インストールか更新かを判断してくれる。インストール経過を「#」のグラフで表示する-vhオプションと組み合わせて、

```
$ rpm -Uvh hoge-1.2.3-4.i386.rpm
```

とする。さらに、コマンドラインに複数のパッケージファイル名を並べて、それらをまとめてインストールすることも可能だ。

依存関係のチェックに失敗すると、この時点で「○○○が必要です」などのメッセージが表示されてインストールに失敗する。この問題を解決するには、必要とされるパッケージを先にインストールすればいい。依存関係を無視する--nodepsオプションや、ほかのパッケージからインストールされたファイルを無条件で置き換える--forceオプションは、安易に使ってはならない。

一方、パッケージをアンインストールする場合は、-eオプションとパッケージ名を指定して、

```
$ rpm -e hoge
```

とすればいい。パッケージファイル名ではなく、パッケージ名を指定する点に注意されたい。

●GUIによるバイナリパッケージのインストール

GUIを使ってRPMのパッケージを扱うソフトとしては、GNOME上で動作する「GnoRPM」、KDE上で動作する「Kpackage」、X上で動作する「glint」などがある。ここでは、GnoRPMとKpackageを使ったインストール方法について簡単に説明しよう。

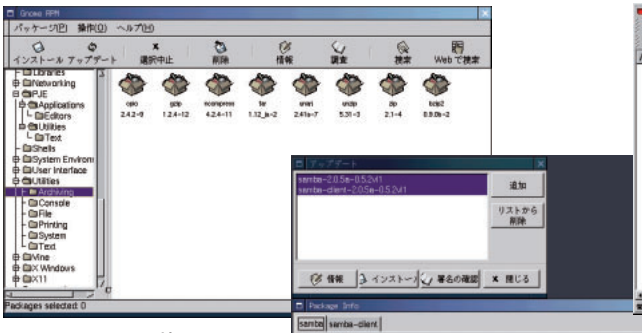
GnoRPMのウィンドウ（**画面2**）には、インストール済みのパッケージがグループ別のフォルダに分類されている。各パッケージは右側の領域にアイコンで一覧表示され、右クリックメニューから[情報（クエリー）]を選択すると、パッケージに関する説明や含まれているファイルを確認できる。

パッケージをインストールするには、[アップグレード]ボタンを押してダイアログを開き、[追加]ボタンでインストール／更新するパッケージを選択する。[情報]ボタンで各パッケージの説明や含まれるファイルを確認でき、一度に複数のパッケージを追加できる（**画面3**）。[インストール]ボタンを押すと、パッケージのインストールが行われる。

一方、Kpackageのウィンドウ（**画面4**）にもインストール済みのパッケージがグループ別にツリー表示される。こちらは、ツリー中にパッケージ名そのものが表示されており、これらをクリックすると、右側の領域にそのパッケージに関する説明や含まれているファイルが表示される。

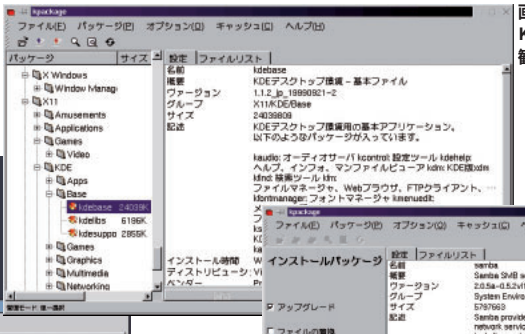
パッケージをインストールするには、ツールバー左端の[パッケージを開く]ボタンを押す。インストール／更新するパッケージを選択すると、ウィンドウの表示がインストール用に切り替わる（**画面5**）。ここでは、パッケージの説明や含まれるファイルを確認でき、[インストール]ボタンを押すと、そのパッケージのインストールが行われる。

このように、GUIを利用したソフト

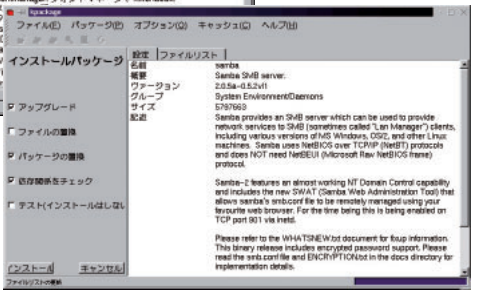


画面2 GNOME環境ではGnoRPMを使おう

画面3 GnoRPMによるインストール



画面4 KDEではKpackageがお勧め



画面5 Kpackageによるインストール

では、インストール済みのパッケージに関する情報を簡単に得られるのが特徴だ。インストール時の手順そのものは、rpmコマンドと比べてそれほど違いがあるわけではない。

■ソースパッケージを利用する

RPMはGPLで配布されているため、Red Hatだけでなく、SuSE／Open／Mandrakeなどの海外ディストリビューション、Vine／Turbo／LASER5／Kondaraといった国産ディストリビューションなどで広く採用されている。

ここで注意すべき点は、「特定のディストリビューション用に作られたバイナリパッケージが、他のディストリビューションでも動作するとは限らない」ということだ。基本的に、あなたが使っているディストリビューション用のパッケージでない場合は、正しく動作する保証はない。

たとえば、Red Hatの6.0以降ではシステムライブラリにglibc 2.1が採用されている。このため、6.0用に作成されたバイナリパッケージを従来のglibc2.0を採用したシステム（Red Hat 5.2系など）にインストールしようとしても、ライブラリの依存関係のチェックではねられてしまう（**画面6**）。

また、TurboLinuxとVine Linuxでは日本語マニュアルを格納するディレクトリ名が微妙に異なる（jaとja_JP.ujis）ため、TurboLinux用のパッケージをVine Linuxにインストールすると（あるいはその逆でも）、日本語マニュアルが正しくインストールされない。

●ソースパッケージからバイナリパッケージをリビルドする

この問題を解決するには、あなたが使っているディストリビューション用のバイナリパッケージを自分で作成してしまえばいい。とはいえ、なにもない状態からパッケージを作るのは結構大変なので、他のディストリビューション用に作られたソースパッケージを流用するといいだろう。

ソースパッケージは、拡張子が「src.rpm」のファイルで、ソースのtarボールや修正用のパッチファイル、バイナリパッケージを作成する手順がかかれたspecファイル（仕様書）などが含まれている。

ソースパッケージからバイナリパッケージを生成（リビルド）するには、--rebuildオプションを使って、

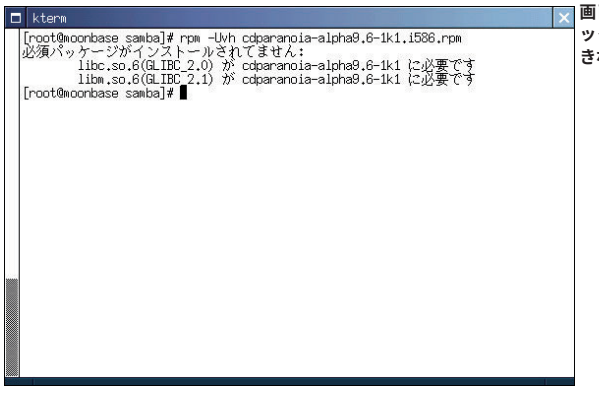
```
$ rpm --rebuild hoge-1.2.3.4.src.rpm
```

とする。specファイルの記述に基づいて、ソースtarボールが展開され、パッチファイルにより修正され、コンパイルとリンクが行われ、バイナリパッケージが作成される（**画面7**）。あとは、/usr/src/redhat/RPMS/i386などに置かれたバイナリパッケージをインストールすればいい。場合によっては、1つのソースパッケージから複数のバイナリパッケージが作成されることもある。

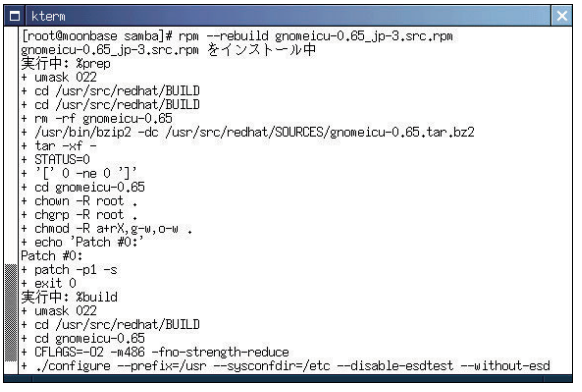
読者の中には、リビルドするだけでライブラリの問題などが解決するのを不思議に思われる方がいるかもしれない。これは、現在配布されているソフトの多くに設定を自動化するスクリプト（configure）が使われており、リビルドの際に実行されたconfigureが、あなたのマシンのライブラリなどを調べて、正しい設定を行ってくれるためだ。

もっとも、パッケージによっては、configureに特定のオプションを指定しなければならなかったり、Makefileなどを直接書き換えなければ動作しないこともある。こうした場合はちょっとやっかいだ。ソースパッケージをインストールして（specファイルなどが展開される）、specファイルの変更やパッチファイルの追加などを行い、ソースパッケージを作り直す必要がある。詳細は、JFのWebページ（http://www.linux.or.jp/JF/）から入手できる「RPM_HOWTO」や「RPM_BUILD_HOWTO」などを参照されたい。

こうした作業にはある程度プログラミングの知識が必要になるので初心者には難しいかもしれない。リビルドだけで動かなかった場合は、とりあえず



画面6 バイナリパッケージが導入できないことも…



画面7 ソースパッケージをリビルドしてバイナリパッケージを作成する

ソースのtarボールから作成する方法（次ページ以降で説明）を利用したほうがいいだろう。

■ソースのtarボールから作成する

ソース配布の利点は、自分のシステムに合わせて設定を変えたり、パッチを当てて不具合の修正や変更を行える自由度の高さだ。今回紹介するソフトの中には、ソースのみで配布されているものも少なくない。

(1) ソースコードを取得し展開する

インストールしたいソフトのtarボールをWebやFTPから取得する。ファイル名は「hoge-1.2.3.tar.gz」のように、「ソフト名－バージョン番号.tar.gz」という書式が多い。

tarボールの内容を調べるには、tarコマンドをtzfオプション付きで実行する。たとえば、

```
$ tar tzf hoge-1.2.3.tar.gz
```

とすると、tarボールに含まれるファイルのリストが表示される。

ほとんどの場合、tarボール中のファイルはディレクトリ付きでアーカイブされている。このため、xvzfオプションをつけて、

```
$ tar xvzf hoge-1.2.3.tar.gz
```

とすると、カレントディレクトリにサブディレクトリ（「hoge-1.2.3」など）が作成され、そこにファイル一式が展開される。スーパーユーザー（root）になって作業を行う場合は/usr/srcに、一般ユーザーで作業する場合はホームディレクトリ以下の適当なディレクトリに展開するといいだろう。

(2) 作成方法の調査とMakefileの作成

ソースの中には、READMEやINSTALLといったファイルが含まれている。これらを閲覧してビルドとインストールの方法を調べよう。説明を精読する必要はなく、「build」や「install」などの単語を探せばいい。

最近のソフトでは、automake／autoconfを利用したスクリプト「configure」によって設定の大半が自動化されているものが多い。この場合、

```
# ./configure
```

とするだけで、コンパイルやリンクに必要な情報が自動的に取得され、Makefileが作成される。もし、必要なライブラリが見つからなかったり、バージョンが古くて対応していないと、警告メッセージを表示して終了する。こうした場合は、まずライブラリのインストールを行ってしまおう。

このほか、「Imakefile」が含まれている場合は「xmkmf」とすればMakefileが作成される。また、Makefileやヘッダファイルなどに書かれた設定を、手動で変更しなくてはならないソフトも存在する。

(3) ビルドからインストールまで

Makefileの設定が適切ならば、

```
# make
```

とするだけでいい。エラーが出ることなく終了すれば、コンパイルとリンクは成功だ。なお、カレントディレクトリに作成された実行ファイルをインストール前に動作確認する場合は、

```
# ./hoge
```

のように、ファイル名の前に「./」を付けることに注意されたい。

たいていの場合、インストールの際もmakeを使って、

```
# make install
```

とすればいい（rootになって行うこと）。バイナリパッケージのインストール先と重ならないように、たいていは/usr/local以下にインストールされる。一方、アンインストールは、「make uninstall」で行えるソフトもあるが、ほとんどの場合は手動で削除する必要がある。

●パッチファイルによる修正

日本語への対応やバグの修正、バージョンアップなどのために配布されるパッチファイルには、オリジナルと修正後の相違点だけが書かれている。

修正（「パッチを当てる」という）は、patchコマンドを使って行う。パッチを当てたいtarボールを展開し、展開先のディレクトリに移動した後でpatchコマンドを使う。パッチファイルは、無圧縮のまま配布されている場合（拡張子patch）と、gzipで圧縮されている場合（拡張子patch.gz）がある。それぞれ、

```
$ patch -p1 < hoge.patch
$ zcat hoge.patch.gz | patch -p1
```

とすればいい。

なお、パッチファイルが対象とするバージョンより新しいファイルにパッチを当てると、patchが修正個所を見つけられないことがある。この場合は、修正できなかった内容が書かれたファイル（拡張子rej）が作成されるので、エディタなどを使って手作業で修正を行う。

高機能な日本語オフィスソフト

USI（JPW／JSIAG／JEGON）

バージョン：1.0.6　　種別：GPL

http://www2.tky.3web.ne.jp/~metal/vMacj.html

X　日本語可

ワードプロセッサJPW、表計算ソフトJSIAG、アニメーション作成ソフトJEGONで構成される日本語オフィスソフト。元来は、ソースが公開されている英語版オフィスソフト「SIAG」（http://www.edu.stockholm.se/siag/）から分岐したものだが、各部の日本語化やキーバインドを含めたインターフェイスの大幅な変更などにともない、別のソフトといえる仕上がりになっている。

●日本語ワープロJPW

JPWは日本語の編集が可能なワープロソフト（**画面1**）。メニューをはじめ、チップヘルプ、ダイアログにいたるまですべて日本語化されている。日本語入力の切り替えはShift＋Spaceキーで行い、任意の位置で文章を折り返すことも可能だ。文字修飾やレイアウトについては、ボールドや上付き／下付き、センタリングなどHTML程度のレベル。フォントの種類・サイズ・色の組み合わせを「スタイル」として登録してまとめて切り替えられる。

文書の管理はExcel風で、それぞれの文書（バッファ）に複数の「シート」が含まれる。シートやバッファの切り替えはウィンドウ下のタブをクリックすればいい。ウィンドウの分割表示も可能だ。ファイルの読み込みと保存については、独自形式のほか、テキスト形式やリッチテキストフォーマット（RTF）、HTML形式などをサポート。読み書きとも日本語EUCとシフトJISに対応しており、読み込み時にDOS／Mac形式の改行コードを自動的にUNIX形式に変換する。印刷については、Postscript形式の出力にのみ対応している。

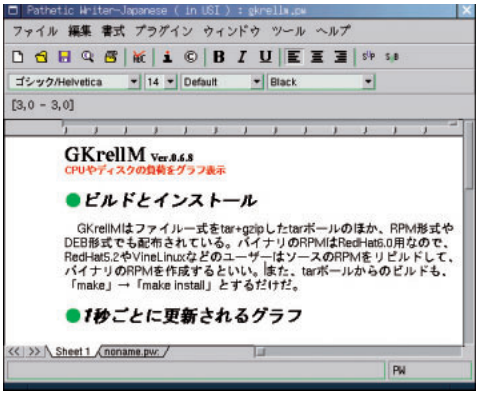
●表計算ソフトJSIAG

JSIAGは日本語が使用できる表計算ソフト（**画面2**）。JPWと同様、メニューやダイアログなどはすべて日本語化されている。セルを囲む黒い枠線がカーソル、背景が灰色のセルが選択領域を表している。バッファやシートの扱いはJPWと同様で、ウィンドウ分割による複数シートの同時表示も可能だ。セルに入力するデータは、ラベル、数値、演算式の3種類。数値以外のデータは自動判別できないため、セルにデータを入力する前に、ラベルの場合はスペースキー、演算式の場合は「=」キーを押す必要がある（セルの編集はスペースキーで統一）。セルの位置指定は、Excelなどでおなじみ A4形式と、行と列の数字をカンマ区切りで表記する形式の両方が使える。このほか、便利な演算式や関数も多数用意されている。

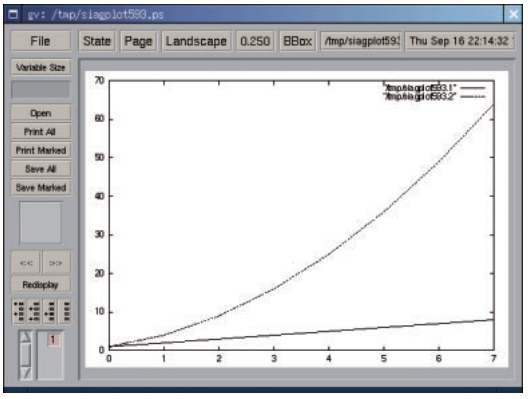
独自のファイル形式以外に、カンマ区切りのCSV形式やテキスト形式、HTML形式、ロータス123形式などにも対応している。印刷に関してはJPWと同様、Postscript形式で出力される。表をそのままプレビュー／印刷するだけでなく、選択範囲のセルの数値をGnuplotでグラフ化してgvで表示することも可能だ（**画面3**）。

●アニメーション作成ソフトJEGON

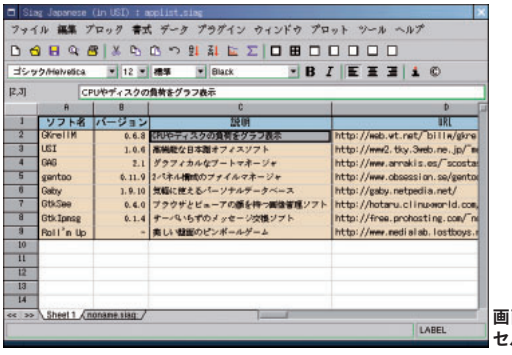
JEGONは、画像や文字、図形などを使ったアニメーションを簡単に作成できるソフト（**画面4**）。エディタとアニメータの2つのウィンドウで構成されている。複数のステップにおけるオブジェクトの位置とサイズを指定すると、その間の動きを自動的に補間したアニメーションが作成される。アニメーションは独自形式で保存するだけでなく、アニメーションGIF形式の画像やCのソースファイルとして出力することも可能だ。なお、アニメーションGIF画像の作成には、USIのパッケージに付属するgifmergeというツールを使う。



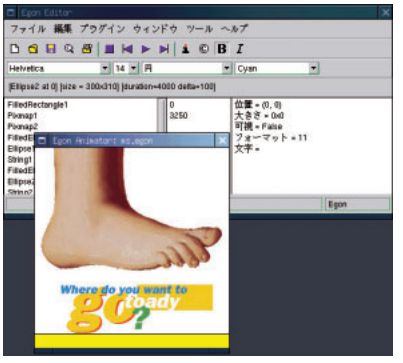
画面1 日本語ワープロJPW。メニューはすべて日本語表示。



画面3　選択範囲の数値をGnuplotでグラフ化しgvで印刷する。



画面2 日本語表計算ソフトJSIAG。セルに日本語を入力可能だ。）



画面4 アニメーション作成ソフトJEGONの2つのウィンドウ。

なんでもこなす驚異のファイルマネージャ

KFM

バージョン：1.167.2.21　　種別：GPL

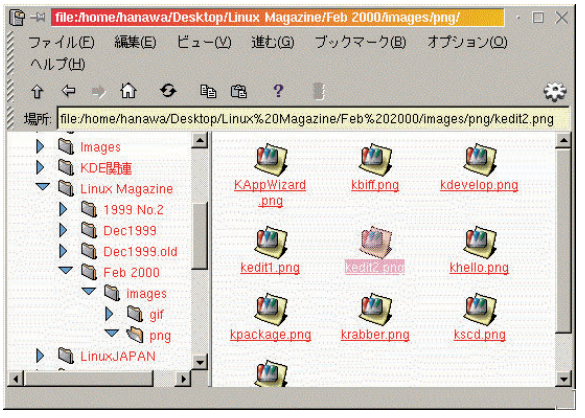
http://www.kde.gr.jp/~komaba/kfmtools.html

X　Qt　日本語可

kfmはkpanelと並んでKDEのルック＆フィールを決定している、まさにKDEの中核アプリケーションだ。その最大の特徴は、ローカルホストのディスク上にあるローカルファイルと、ネットワーク上にあるリモートファイルを区別することなく取り扱えるネットワーク透過性だ。

kfmは基本的にはファイルマネージャだが、Webブラウザ、FTPクライアント、アーカイブブラウザとしても機能する。HTMLファイルをkfmウィンドウ内で表示でき、リンクをクリックするかロケーションバーにURLを入力すれば、そのままWebサーフィン開始だ。FTPクライアントにもなるので、FTPサイトへもシームレスに接続できる。この時、FTPサイトのファイルをデスクトップやローカルフォルダにドラッグ＆ドロップするだけでコピーができるのはもちろん、anonymousでない通常のFTPで接続してあれば、リモートホスト上のファイルコピーも可能だ。tar＋gzipなどのアーカイブファイルも通常のフォルダと同様に取り扱えるので、特定のファイルだけをtarボールから取り出したい時などに、いちいちアーカイブを解凍する必要もない。ほかにも画像ファイルのサムネイルが表示できるので、画像ファイルの取り扱いにも便利だ。

kfmをさらに便利に使うためのユーティリティ集KFM Toolsが、日本KDEユーザ会の井上氏により公開されている。これによりkfmの利用範囲がさらに広がる。まだ使ったことがない方はぜひ試してみて欲しい。



画面1 通常はファイルマネージャ

画面2 WWWブラウザにもFTPクライアントにもなる。

簡単に使えるテキストエディタ

Kedit

バージョン：1.2.2　　種別：GPL

http://www.kde.org/

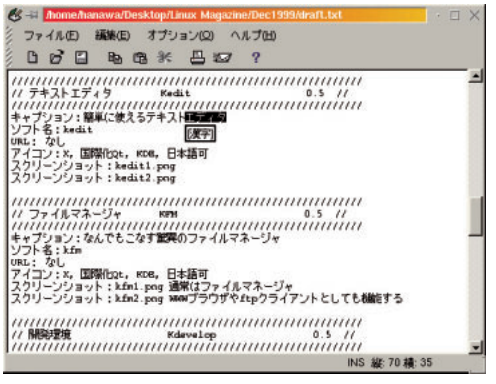
X　Qt　日本語可

KDEの標準エディタ。kfmで.txtや.html等のテキスト形式のファイルのアイコンをクリックするか、パネル上の「簡易テキストエディタ」アイコンをクリックすることで起動する。小さいためすぐに立ち上がるので、ちょっとしたメモをとりたい場合や、設定ファイルをちょっと修正したい場合に重宝する。

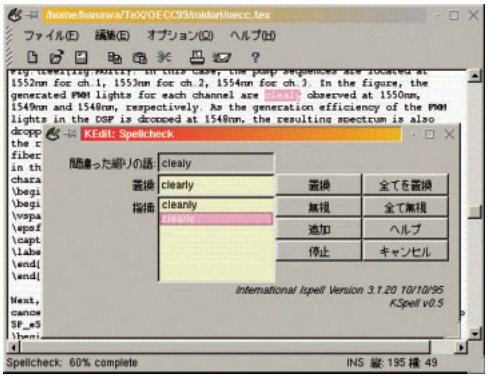
手軽に使えるからといって、低機能と馬鹿にしてはいけない。確かにEmacsほどの高い機能は実装されておらず、プログラミングには向かないが、自動行折り返し、自動バックアップ、バッファ内容のメール送信、指定したURLの読み込みと保存、kspellによる英文スペルチェック機能など、キラリと光る小粋な機能を備えている。

ほかのKDEアプリケーションと同じく、keditもkfmと組み合わせて利用する場合に本領を発揮する。keditはKDEドラッグ＆ドロップに対応しているので、デスクトップやkfmウィンドウ内のファイルをkedit上にドラッグ＆ドロップして開くこともできる。kfmウィンドウがFTPクライアントとして働いているときも、FTPサーバ上のファイルをローカルファイルと同じように編集することが可能だ。このネットワーク透過性こそが、KDEアプリケーションの醍醐味といえるだろう。

スペルチェック機能は、ispellのKDE用フロントエンドであるkspellを外部アプリケーションとして呼び出すことで実現している。このため必要十分なスペルチェック機能が簡単な操作で利用できる。



画面1 日本語の入力ももちろん可能。



画面2 スペルチェッカを呼び出せる。

統合開発環境

KDevelop

バージョン：1.0beta4.1　　種別：GPL

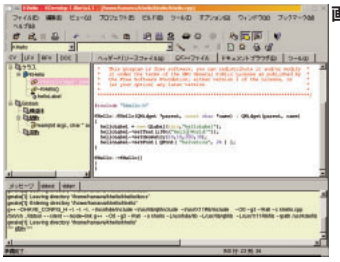
http://www.kdevelop.org/

X　Qt　日本語可

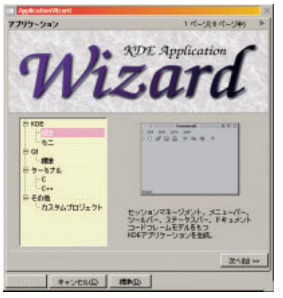
KDevelopはUNIX用ソフトウェア開発のための統合開発環境だ。エディタ、コンパイラ、リンカ、automake、autoconfなどを統合しているのはもちろんのこと、以下の豊富な機能が組み込まれている。

・KAppWizardによる自動ひな型生成
・Classgeneratorによる容易なクラス作成と既存コードへの組み込み
・プロジェクト内のソース／ヘッダ／ドキュメントの一元管理
・SGMLフォーマットのユーザハンドブック作成支援とKDE形式のHTMLヘルプファイル出力の自動生成
・プロジェクトに含まれるクラスAPIのHTMLドキュメント自動生成
・アプリケーションの国際化支援
・組み込みダイアログボックスエディタによるダイアログボックスのWYSINWYG（What You See Is NEARLY What You Get）生成
・CVSによるプロジェクト管理のための使いやすいフロントエンド
・KDbgとの統合によるデバッグ
・KIconEditによるプロジェクト用アイコンピックスマップの作成
・各種開発用ツールのToolsメニューへの組み込み

KDE用アプリケーションの開発にはもちろん、C++を用いたGUIプログラムの開発全般で利用できる。KDevelopは開発を容易かつ楽しくしてくれるデベロッパー向け統合ソフトといえるだろう。



画面1 開発中の画面。



画面2 アプリケーションの雛型は自動的に生成。

画面3 国際化アプリケーションも簡単に作成可能。

日本語が使えるプレゼンテーションツール

MagicPoint

バージョン：1.0.6a　　種別：フリーソフト

http://www.Mew.org/mgp/

日本語可

Linuxではメジャーなプレゼンテーションツール。ユーザーは .mgpという拡張子のテキストを編集してプレゼンテーションを作る。出来上がったプレゼンテーションはPostScriptに変換して印刷することもできる。MagicPointは、文字表示にVFlibを使っているため、日本語フォントで大きな文字を書いてもギザギザが出ることなく表示できる。WindowsのPowerPointのようにビジュアルに編集できないが、テキスト処理を応用してプレゼンテーションを作れるという、UNIXらしいコンビネーションで使いこなすことができる。最近、MagicPointでプレゼンテーションを作るときに役立つテンプレートを集めたサイトがオープンした。(http://puchol.com/cpg/software/mgp/)

日本語対応のMS Wordドキュメントのビューア

wv

バージョン：0.5.42　　種別：GPL

http://www.wvWare.com/

日本語可

Microsoft Wordのドキュメントを解析するためのライブラリ。Wordのドキュメントでは日本語がUnicodeで出力されるため、wvでもある程度までは日本語を扱うことができる。Linux用の商用ワードプロセッサであるAbiWordは、このwvを利用してWordファイルを読み込む機能を備える。

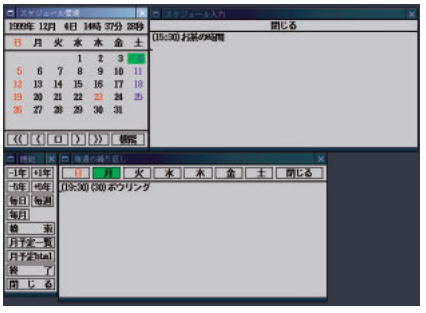
Windowsのオフィスツールで作ったファイルをLinuxでも扱えれば、多くのユーザーがWindowsを使う必要がなくなり、Linuxの普及が進むと期待されている。マルチバイト語圏の日本では、日本語化や国際化の壁があり、なかなかそこまではいかないが、シングルバイト語圏の欧米諸国では、すでにそうした状況になりつつある。

スケジュール管理が可能な日本語カレンダー
nittei

バージョン：1.80　　**種別：GPL**

http://member.nifty.ne.jp/seto-yoneji/nittei.html

X　日本語可



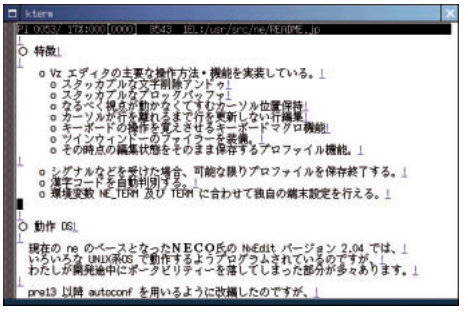
スケジュール管理を行えるカレンダーソフト。国産アプリなのでパッチなどを当てなくても最初から日本語が使える点がうれしい。2000年から施行された祝日月曜化法による成人の日、体育の日の移動に対応しているほか、ユーザーによる休日や記念日の追加も可能。スケジュールは分単位で設定でき、予定時刻（あるいは指定した何分前か）になるとウィンドウが開いてユーザーに通知してくれる。また、指定時刻にプログラムを自動実行したり、毎日、毎週、毎月の繰り返しスケジュールを設定することも可能だ。予定時刻を記述しなかった場合、通知などのスケジュール管理からは外れるものの、入力した内容はそのまま保存されるので、簡単な日記代わりにも使える。

Vzによく似たインターフェイスを持つエディタ
NxEdit（ne）

バージョン：3.00pre17　　**種別：GPL**

http://www3.justnet.ne.jp/~ele/

コンソール　日本語可



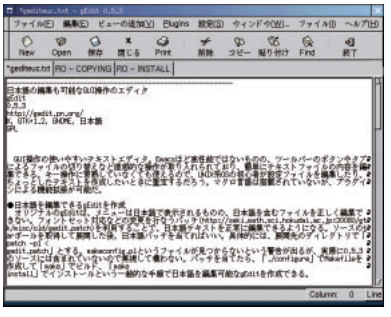
コンソールやktermなどの端末画面で軽快に動作する日本語対応のテキストエディタ。MS-DOS用の高速テキストエディタとして一世を風靡したVzエディタによく似たインターフェイスが特徴だ。たとえば、カット（コピー）＆ペースト用のスタッカブルなブロックバッファ、編集していたファイル名や行番号を保存するプロファイル機能、ファイルオープンの際に利用できるファイラなど。モードレスなWordstar系のエディタをDOSやWindowsで愛用している人には、viやEmacsよりもはるかに使いやすいだろう。なお、初期設定のキーバインドはVzエディタ風だが、テキスト形式の設定ファイルを変更することで柔軟にカスタマイズできる。

日本語の編集も可能なGNOMEのエディタ
gEdit

バージョン：0.6.1／0.5.3（日本語版）　　**種別：GPL**

http://gedit.pn.org/
http://seki.math.sci.hokudai.ac.jp:20080/gtk/misc/obsolete/

X　GTK+　GNOME　日本語可



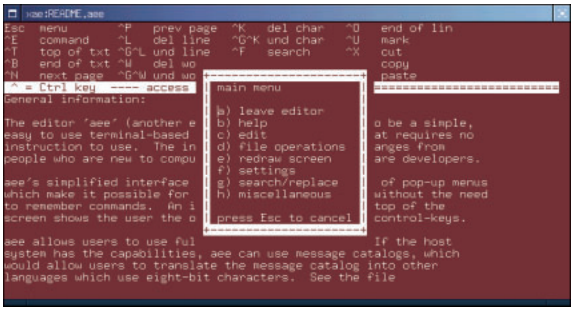
GUI操作の使いやすいテキストエディタ。Emacsほど高性能ではないものの、ツールバーのボタンやタブによるファイルの切り替えなど直感的な操作が取り入れられており、簡単にテキストファイルの内容を編集できる。キー操作に習熟していなくても使えるので、UNIX系OSの初心者が設定ファイルを編集したり、ちょっとしたテキストを作成したいときに重宝するだろう。日本語EUCのテキストを正しく編集するには、フォントセット対応などの変更を行う日本語化パッチを当てる必要がある。マクロ言語は搭載されていないが、プラグインによる機能拡張が可能で、編集中のテキストの単語数などを調べる「Word Count」などのプラグインが付属する。

Escキーでメニューが表示される小型エディタ
aee、xae、ee

バージョン：2.2.3　　**種別：Artistic**

http://www.users.uswest.net/~hmahon/

X　コンソール



コンソールとXの両方で使える小型（150Kバイト程度）のテキストエディタ。コンソール版のインターフェイスをaee、X用のインターフェイスをxaeと呼ぶ。日本語には対応していないが、小型軽量な利点を生かして設定ファイルの書き換えなどで使えるだろう。使い方はどちらも同じで、Ctrl－英字キーで編集を行うモードレスタイプ。キーバインドは独特だが、主要なキー操作が画面上部に表示されているので、初めて使う場合でも戸惑うことはない。また、Escキーで開くポップアップメニューでは、各種設定の変更や使い方の表示、ファイルの読み込みや保存、編集終了といった操作を行える。このほか、さらに小型（80Kバイト程度）のeeも配布されている。

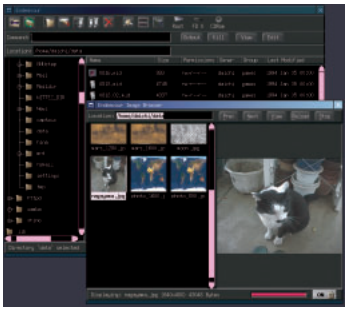
画像閲覧に便利なファイルマネージャ

Endeavour

バージョン：1.07　　種別：GPL

http://fox.mit.edu/xsw/edv.htm

X



黒い背景がクールなファイルマネージャ。作者グループ（WolfPack）によるゲーム「XShipWars」（http://fox.mit.edu/xsw/）のウィンドウ部品（ウィジェット）が使われているため、サイバーな香りが漂う。こうした外見に似合わず、ファイルの削除などを一律に禁止する「マスターライトプロテクト」機能や、削除ファイルを復活できる「ゴミ箱」を装備するなど親切な設計だ。また、JPEG／GIF／TIFF／BMP形式などに対応したイメージビューアや、サムネイル表示を行うイメージブラウザを内蔵しており、画像の閲覧にも向いている。なお、Endeavour自体は日本語には対応していないが、外部ソフトを使えばファイルの内容表示や編集は問題ない。

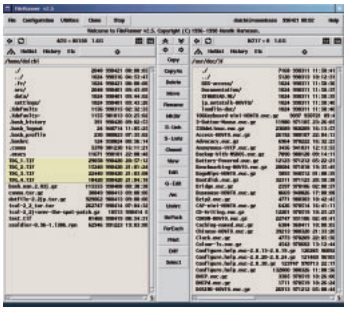
FTPクライアントにもなるファイルマネージャ

FileRunner

バージョン：2.5　　種別：GPL

http://www.cd.chalmers.se/~hch/filerunner.html

X　Tcl/Tk　日本語可



2つのディレクトリのファイル一覧が表示され、コピーや移動の結果をその場で確認できるファイルマネージャ。日本語対応のTcl/Tkであれば日本語ファイルの閲覧や編集も可能だ。操作は簡単で、ファイルをクリックやドラッグで選択した後、中央部に並んだボタンを押せばいい。たとえば、[View]ボタンを押すと、テキストやアーカイブの内容は内蔵ビューアで表示され、画像ファイルの内容はxvで表示されるなど、ファイルの拡張子に応じた閲覧が行われる。また、FTPクライアントソフトとしての機能も備えており、ディレクトリの代わりに接続先URLを入力することで、ローカルなディスクと同じ感覚でFTPサーバ上のファイル処理を行える。

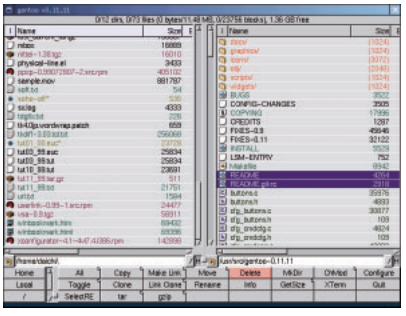
2ペイン構成の使いやすいファイルマネージャ

gentoo

バージョン：0.11.11　　種別：GPL

http://www.obsession.se/gentoo/

X　GTK+　日本語可



2つのディレクトリのファイル一覧が同時に表示され、コピーや移動の結果をその場で確認できる2ペイン構成のファイルマネージャ。使い方は簡単で、どちらかのファイルリストのファイルをクリックなどで選択後、ウィンドウ下部のボタン（[Copy]や[Delete]など）を押せばいい。左右のファイルリストの表示をはじめ、コマンドボタンの内容やキーバインド、拡張子などに応じたダブルクリック時の動作など、gentooに関するあらゆる設定を設定ダイアログから変更できるのが特徴だ。なお、ソース（gentoo.c）中の「gtk_init();」の前に「gtk_set_locale();」を1行追加してビルドすると、ファイル一覧や内蔵ビューアで日本語を正常に扱えるようになる。

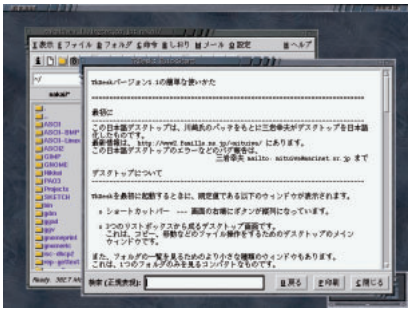
Tcl/Tkを用いたファイルマネージャ

TkDesk

バージョン：1.2　　種別：GPL

http://people.mainz.netsurf.de/~bolik/tkdesk/

Tcl/Tk　日本語可



TkDeskで日本語デスクトップ （tkdesk.bmp）

Tcl/Tkを使ったファイルマネージャを中心としたデスクトップ環境である。ファイルマネージャは高速で、テキストファイルをクリックすると、軽快な動作でエディタが起動して表示される。ファイルマネージャは多機能な上にメーラ（TkMailer）と連動しており、統一感がある。

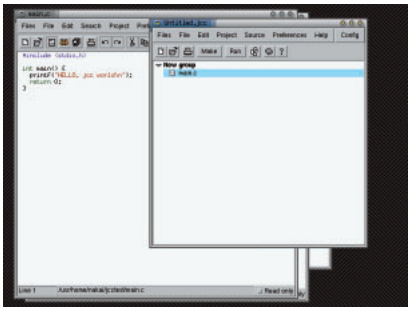
KDEやGNOMEのように、デスクトップ全体でアプリケーションの統一性をはかるということを考えなければ、デスクトップ環境とはTkDeskのようなあり方も正しかったといえるだろう。テキストの編集、印刷においてもちゃんと日本語を使うことができる。Tcl/Tkによるデスクトップ環境では、ほかにもTkStepが日本語化されている。

Core Warrior風の多言語開発環境
Code Crusader

バージョン：2.1.4 **種別：フリーソフト**

http://www.newplanetsoftware.com/jcc/

X



MetroWorksの商用開発環境Code Warriorに触発されて作られたというツール。VisualBasicなどのようにGUIアプリケーションを作るための機能はないが、ビジュアルな開発環境を提供する。エディタはよく作られており、Cのソースコードや、HTMLのソースの構造を理解して、タグなどを強調表示してくれる。特に日本語対応を必要としないユーザーなら、大変便利に使うことができるだろう。

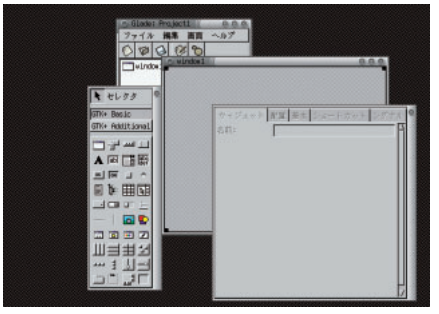
GUIライブラリにはJXというものを使用している。JXは非商用に限って無償のアプリケーションフレームワークである。日本語のサイトもあり、QtやGtk+などのメジャーなGUIライブラリに比べて、JXがいかに優れているかが記載されている。

GTK+用のインターフェイスビルダ
Glade

バージョン：0.5.5 **種別：GPL**

http://glade.pn.org/

X GTK+ 日本語可



GTK+アプリケーションをビジュアルに作ることのできる開発ツール。GTK+向けの開発ツールはほかにもあるが、Gladeは頻繁に改良が続けられ、最も人気があるもののひとつとなっている。アプリケーション全体、または一部を、Gladeで作ったアプリケーションは結構ある。

基本的には、プロジェクトを開き、ウィンドウを作り、ウィジェットを貼りつけて表示する文字列や細かなプロパティを編集していって、最後にソースコード書き出すというスタイルになる。Gladeで作ったアプリケーションは、出力したソースコードからバイナリを作って単体で動作させるわけだが、libgladeを使うと、Gladeのプロジェクトをコンパイルなしでランタイムのようにして実行できる。

OpenGLを利用したリアルな3D時計
glclock

バージョン：5.0 **種別：フリー**

http://www.daionet.gr.jp/~masa/glclock/index.html

X Mesa

OpenGL（あるいは互換ライブラリのMesa）を利用して、リアルな3D時計を表示するソフト。起動時オプションの指定や同梱のシェルスクリプトにより、ソフトウェアによるシミュレーションでも快適に表示されるシンプルなデザインから、ハードウェアの3Dアクセラレータが必須のゴージャスなものまで、バリエーションに富んだ外観を楽しめる。また、さまざまな設定で時計を表示し、フレームレートを計測するベンチマーク用スクリプトも用意されている。さらに、ボタンを押したままマウスを動かすことで、時計を移動、回転させたり、フタを開け閉めできる。画面全体を時計が回転しつつ移動するというスクリーンセーバ的な使い方も可能だ。

円形のウィンドウを使った美少女コスプレ時計
Emi Clock for X11

バージョン：2.0.2 **種別：フリーソフト**

http://www.plaza.hitachi-sk.co.jp/~masa-k/EmiClock/
http://www02.u-page.so-net.ne.jp/momo/yuna-k/index-jp.html

X 日本語可

円形のウィンドウにかわいい女の子の絵が表示されるアナログ時計。ウィンドウマネージャ側で、タイトルバーやリサイズバーを表示しないように設定して使うといいだろう。X版のEmi Clock for X11は古場氏が作成したもので、グラフィックや音声データはオリジナルのMac版やWindows版と同じ物が使われている。チャイムやアラームはもちろん、1時間ごとに女の子が着替えたり、3種類の文字盤を選べるなど楽しい機能が満載だ。ウィンドウをクリックすると、オプションや着替えの設定を行うためのメニューがポップアップする。なお、サブメニューが表示されない場合は、メニューの左右の端ぎりぎりまでカーソルを移動させるといいようだ。

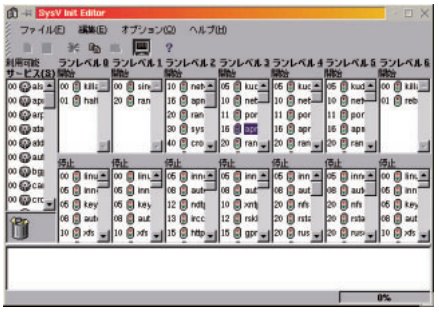
GUIベースのSysV Initエディタ

KDE SysV Init Editor（ksysv）

バージョン：0.4.99　　種別：GPL

http://eclipt.uni-klu.ac.at/projects/ksysv/

X　Qt



ksysvは、SysV InitのGUIフロントエンドだ。SysV Initはシステムの状態をランレベル（0～6）に分け、ランレベルごとに稼働／停止するサービスを管理するツールだ。たとえばRed Hat系では/etc/rc.d/rc?.d/（？はランレベル）にSまたはKで始まるファイル名のスクリプトを置き、サービスを制御する。Sで始まるのはそのランレベルに入ったときに稼働するサービス、Kで始まるのは停止するサービスだ。

左端の「利用可能サービス」内から各ランレベルの開始／停止エリアにドラッグ＆ドロップするだけで、サービスの設定ができる。不要なサービスはゴミ箱に入れればいい。rootでしか利用できないので、suコマンドでroot権限を得てから起動しよう。

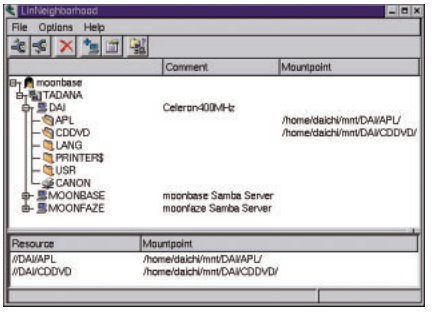
ディレクトリのサイズを一覧で表示

LoST

バージョン：0.97　　種別：GPL

http://www.geocities.com/SiliconValley/8036/lost_page.html

X



指定したディレクトリ以下の、全サブディレクトリのサイズを表示するソフト。コンソール用コマンドの「du」をGUI化し、拡張したものといえる。使い方は簡単で、メニューやツールバーの「choose dir」から調べたいディレクトリを選ぶだけだ。各ディレクトリは、サイズに応じて「Kb」「Mb」の単位付きで表示される。チェックする必要のないサブディレクトリは「Exclude dirs」で指定しておけば、計算されない。ファイルやディレクトリのアイコンを右クリックすると、Delete／Copy／Pasteなどのメニューが表示される。現バージョンではDeleteは可能だが、CopyやPasteによるファイルの移動は実装されていないようだ。

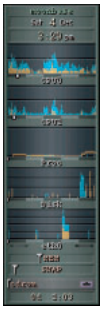
CPUやディスクの負荷をグラフ表示

GKrellM

バージョン：0.7.4　　種別：GPL

http://web.wt.net/~billw/gkrellm/gkrellm.html

X



CPUやディスクの負荷、プロセス数、PPPやイーサネットのパケット流量、メモリやスワップの使用量などを縦型の小さなウィンドウにグラフやメータで表示するモニタリングソフト。Linuxカーネル2.2以降でマルチプロセッサ（SMP）を使っている場合は、CPUの数だけCPUグラフを表示する。グラフは1秒ごとに更新され、縦方向のスケールは測定値に応じて自動的に調整される。また、各グラフともシアンとオレンジの2色で異なるデータを表示しており、たとえばCPUグラフではユーザータイムとシステムタイム、ディスクグラフではリード量とライト量といった具合だ。表示内容の選択、計測単位量やグラフの大きさは設定により変更できる。

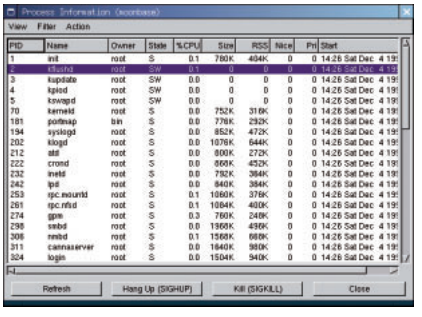
GTK+によるグラフィカルなプロセス情報表示

gPS

バージョン：0.4.1　　種別：GPL

http://www.gps.seul.org/

X　GTK+



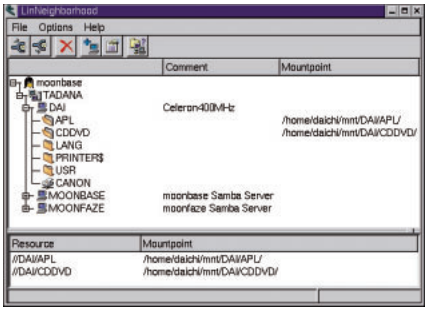
従来はpsやtopコマンドを使って表示していたプロセス情報を、GTK+を利用して表形式やグラフで表示するソフト。topコマンド風の表形式で、プロセスID、プロセス名、オーナー名、状態、CPU占有率、メモリ使用量、実メモリ使用量、nice値、優先度、開始時間といった情報が表示される。項目名をクリックするだけで、その内容をキーとしてプロセス一覧をソートできるほか、フィルタ機能を使って条件に合うプロセスだけに絞り込むことも可能だ。CPUとメモリの使用状況をグラフで表示する機能も用意されている。なお、KDE向けの同様のソフトとしては、Qtツールキットを利用した「qps」（http://www.student.nada.kth.se/~f91-men/qps/）がある。

Sambaクライアントの表示とマウントを行う
LinNeighborhood

バージョン：0.4.2 **種別：GPL**

http://www.bnro.de/~schmidjo/

X GTK+



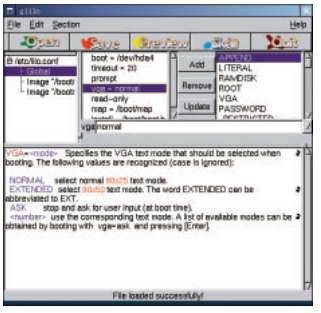
Sambaを利用して、LinuxマシンやWindowsマシンなどの共有ディレクトリをマウント／アンマウントするソフト（smbclient／nmblookup／smbmount／smbumountが必要）。ウィンドウには、Sambaからアクセス可能なマシン名と、それぞれの共有リソースがツリー形式で表示されており、共有リソースをダブルクリックするとマウント用のダイアログが開く。初期設定では、実行ユーザーのホームディレクトリ以下の「mnt/マシン名/リソース名」にマウントされる（変更可）。続いて、指定したファイルマネージャが起動され、マウントしたディレクトリの内容を表示する。ウィンドウ下には、マウントしているリソース名とマウントポイントが一覧表示される。

LILOの設定ファイルをGUIで編集
GLilo

バージョン：0.4 **種別：GPL**

http://www.student.hig.se/~nd96pwt/glilo/

X GTK+



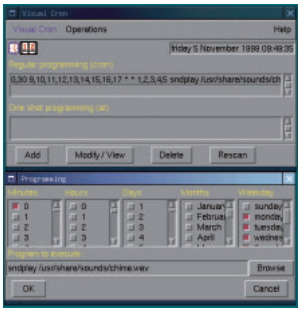
Linuxで広く使われているブートローダ「LILO」の設定ファイル（/etc/lilo.conf）の内容をGUIを使って編集するソフト。LILOの設定ファイルで指定するオプションやその意味についての詳細は意外に知られていない。GLiloを使うと、設定ファイルの内容を読み込んで、セクション別にオプションを分類表示してくれる。さらに、オプションの内容変更、削除、追加の際には、画面下にそのオプションに関するヘルプが表示され、現在指定しているオプションの意味や、追加するオプションの指定方法などが簡単にわかるようになっている。なお、変更した設定を実際に反映させるには/sbin/liloを実行する必要があることをお忘れなく。

時刻を指定した繰り返し処理をGUIで設定
vcron

バージョン：1.4 **種別：フリーソフト**

http://www.linux-kheops.com/pub/vcron/vcronGB.html

X Tcl/Tk



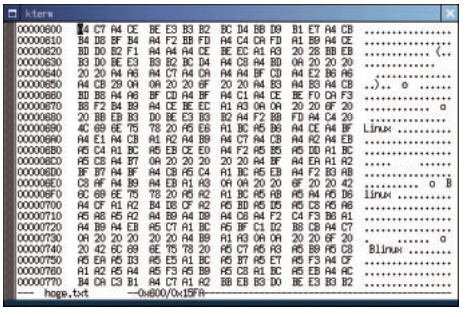
指定時刻に繰り返し処理を行うcronシステムをGUIを使って設定するソフト。指定時刻や一定時間後に1回だけ処理を行うatコマンドの設定も可能だ。UNIX系OSでは、crondやatdといったデーモンを利用して、指定時刻や一定時間後のコマンドの自動実行が容易に行える。特に、Linuxのディストリビューションに広く採用されているVixie cronは、BSD風の/etc/crontabによる設定と、SysV風のcrontabコマンドによるユーザーごとに独立した設定を同時に利用でき、crondの再起動も必要ないという優れものだ。しかし、設定ファイルの書式は「0,30 9-17 * * Mon-Fri ...」などと呪文のようなものなので、慣れるまではこうした設定をvcronで行うといい。

Emacs風のキーバインドで使えるバイナリエディタ
Hexedit

バージョン：1.1.0 **種別：GPL**

http://www.chez.com/prigaux/hexedit.html

コンソール



ファイルの内容を16進数とASCII文字列で表示し、バイナリファイルを直接書き換えることのできるバイナリエディタ。カーソルキーやファンクションキーでカーソル移動やファイルの読み込み、保存を行えるほか、Emacs風のキーバインドでカーソル移動や検索、アンドゥ、カット（コピー）＆ペーストなどの操作が可能だ。Tabキーで、カーソルが16進数表示部とASCII表示部に交互に切り替わる。一方、「ふつ〜vi」という人には、vi風のバイナリエディタ「bvi」（http://bvi.linuxbox.com/）がお勧めだ。画面表示はHexeditに似ているが、キーバインドはviに準拠しており、hjklキーでのカーソル移動はもちろん、「:」で始まるex系のコマンドも用意されている。

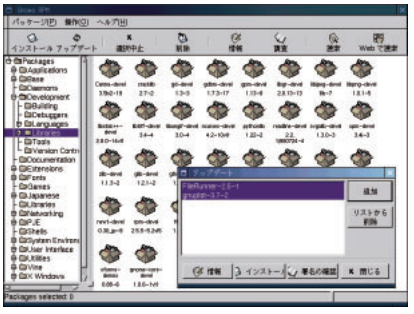
GNOME採用のグラフィカルなRPMマネージャ

GnoRPM

バージョン：0.9　　種別：GPL

http://www.daa.com.au/~james/gnome/

X　GTK+　日本語可



Red Hat系Linuxで使われているパッケージシステム「RPM」のパッケージをGUIで操作するソフト。複雑なオプションをコマンドラインで指定することなくパッケージを扱える。ウィンドウには、インストール済みのパッケージのアイコンがジャンル別にフォルダで分類され、アンインストールや情報の取得、調査などが可能。新たにパッケージをインストールしたり、新しいバージョンに更新する場合は、ツールバーの[インストール]または[アップデート]ボタンを押してダイアログを開き、[追加]ボタンで目的のパッケージを選択すればいい。一度に複数のパッケージをインストールしたり、インストール前に各パッケージの情報を確認することも可能だ。

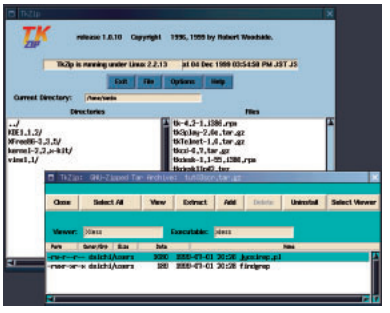
RPMやZIP／LHAも扱えるアーカイブマネージャ

TkZip

バージョン：1.0.15　　種別：フリーソフト

http://www.pcnet.com/~proteus/TkZip/TkZip.html

X　Tcl/Tk　日本語可



Tcl/Tkで書かれたアーカイブマネージャ。tar＋gzされたtarボールだけでなく、RPM形式のパッケージや、Windowsで使われるZIP／LHA／RAR形式のアーカイブも扱える。実際には、各アーカイブ形式に対応したコマンド（tar／gzip／zip／unzip／lhaなど）をTkZipが自動的に検索して内部で実行している。いずれの形式の場合も、アーカイブ内のファイル一覧がウィンドウに表示され、ファイルの展開やビューアによる閲覧といった処理を統一されたGUI操作で行える。もちろん、アーカイブへのファイルの追加や新規アーカイブの作成も可能だ。また、閲覧に使われるビューアはユーザーが自由に設定でき、画面表示色や動作も細かくカスタマイズ可能だ。

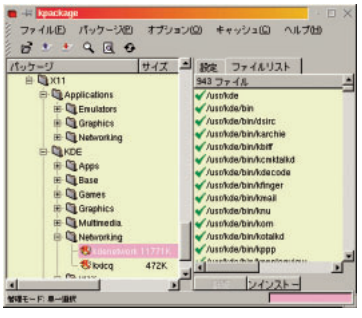
汎用パッケージマネージャ

Kpackage

バージョン：1.3　　種別：GPL

http://www.general.uwa.edu.au/u/toivo/kpackage/

X　Qt　日本語可



RPM／Debian／Slackware／BSDの各パッケージを統合的に管理するためのGUIインターフェイスだ。実際の処理を行うrpm／dpkgなどのコマンドが別途必要となる。起動するとウィンドウの左側ではインストール済みのパッケージがRPMの場合、階層表示され、まるでファイルマネージャのような操作感覚でパッケージを扱うことができる。

KDEドラッグ＆ドロップにも対応しており、Kpackageのウィンドウ上にデスクトップやkfmウィンドウからパッケージをドラッグ＆ドロップするだけで、パッケージ情報の閲覧／インストール／アップグレード／テストなどが行える。なおパッケージのインストールには、root権限が必要になる。

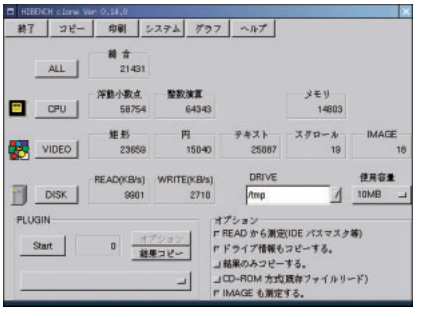
CPU／ビデオ／ディスクのベンチマークを計測

HDBENCH clone

バージョン：0.14.0　　種別：GPL

http://www.enjoy.ne.jp/~gm/program/hdbench/index-ja.html

X　GTK+　日本語可



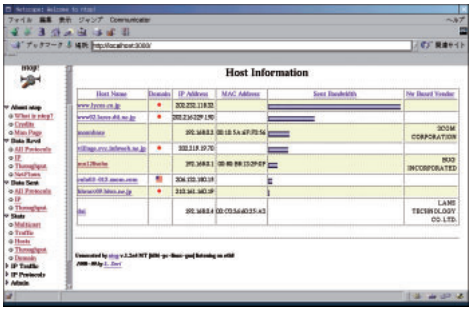
有名なWindows用国産ベンチマークソフト「HDBENCH」にそっくりのインターフェイスを持つ、UNIX系OS用の国産ベンチマークソフト。大きく分けて3種類（CPU、ビデオ、ディスク）のベンチマークを計測できる。これらは、それぞれ独立して計測することもできるし、まとめて順番に計測することも可能だ。CPUに関しては、1秒間あたりの浮動小数点演算および整数演算の計算回数、メモリ転送速度を計測する。ビデオでは、1秒間あたりの矩形、円、テキストの描画回数、3秒間のスクロールでの描画回数、5秒間のイメージの描画回数を計測する。ディスクについては、1秒間の書き込み、読み込み速度を計測する。計測結果のコピーや印刷も可能だ。

ネットワークトラフィックをWebブラウザで監視
ntop

バージョン：1.2a9　　種別：GPL

http://www-serra.unipi.it/%7Entop/

コンソール



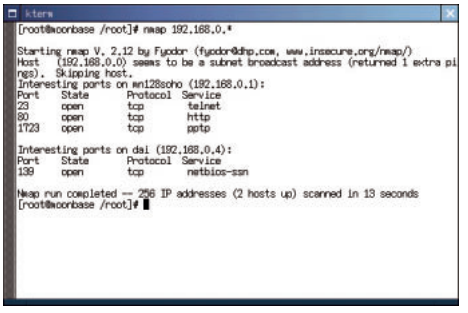
ネットワークトラフィックを監視して、端末画面やWebページに表示するソフト。ntop（network top）という名は、端末画面を利用するインタラクティブモードがtopコマンドに似ていることに由来する。一方、Webページに監視結果を表示するWebモードも用意されており、起動時に-wオプションとポート番号（3000など）を指定することでHTTPサーバとして動作する。ブラウザでホスト名とポート番号のURL（ローカルホストなら「http://localhost:3000/」）を指定するとトップページが表示され、左側の階層メニューをクリックすると、送受信データのデータ量をドメイン別に表示するページや、IPトラフィックの統計を表示するページなどに切り替わる。

不要なサービスが実行されていないか調べる
nmap

バージョン：2.12（安定版）/2.3beta8　種別：GPL

http://www.insecure.org/nmap/

コンソール



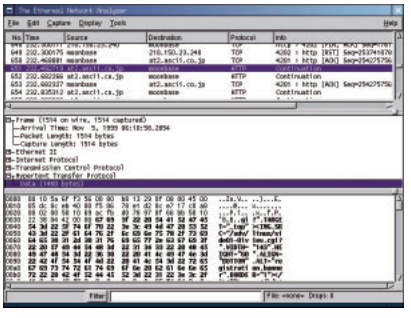
各種のポートスキャンを行うツール。ホスト名やアドレスを指定したマシン（あるいはゾーン）を対象に、広範囲のポート番号への接続を試み、開放されているポート番号とサービス名を表示する。管理者が気づかないまま不要なサービスが有効になっていると、他のプロセスの実行速度に悪影響を与えるし、セキュリティホールの原因ともなる。そこで、自分の管理しているマシンが開放しているポートをnmapでチェックし、不要なサービスが有効になっていれば、対応するサーバを起動しないよう/etc/inetd.confを修正するなどの対策をとる。なお、管理外のマシンに対してポートスキャンを行うとクラックと見なされることがあるので注意されたい。

パケットを監視するネットワークアナライザ
Etherreal

バージョン：0.7.9　　種別：GPL

http://ethereal.zing.org/

X　GTK+



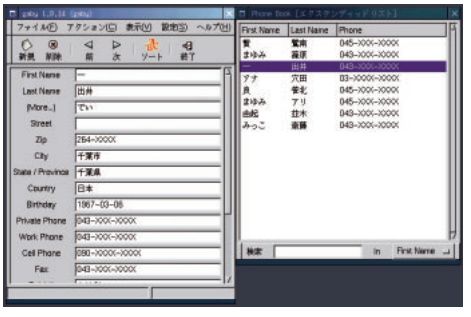
イーサネットやPPP接続用のデバイスを監視し、そこを流れるTCPなどのパケット情報を取得するネットワークアナライザ。ウィンドウ上部には、取得したパケットの取得日時／送信元／送信先／プロトコル／簡単な説明が一覧表示される。強力な表示フィルタを使って、表示するパケットを絞り込むことも可能だ。一覧中のパケットをクリックすると、そのパケットに関する詳細な情報がウィンドウ下側に表示される。TCPデータストリームの内容（テキスト）を別ウィンドウに表示することも可能だ。得られたデータをファイルに保存して後で読み込んだり、tcpdumpなど他のツールの出力を取り込んで解析する機能も用意されている。

気軽に使えるパーソナルデータベース
Gaby

バージョン：1.9.14　　種別：GPL

http://gaby.netpedia.net/

X　GTK+　GNOME　日本語可



住所録やCD目録などを管理できるパーソナルデータベースソフト。サーバ・クライアント方式のデータベースソフトを使うほど大規模ではない個人的なデータの管理に適している。データの入力や閲覧は、各種の「ビュー」で行う。日本語も問題なく入力、編集可能だ。基本的なビューとして、データの入力・編集に適した「フォーム」と、多数のレコードを同時に閲覧できる「リスト」の2つがあり、このほか「拡張リスト」や「キャンバス」などのバリエーションがある。データベースの構造を定義している記述ファイルを書き換えることで、項目の表示順の変更や新たな項目の追加を柔軟に行えるほか、プラグインによる機能拡張にも対応している。

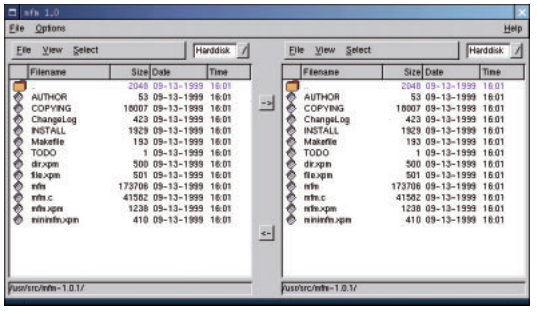
DOS形式のフロッピーを簡単に扱える

mfm

バージョン：1.2　　種別：GPL

http://www.core-coutainville.org/mfm/

X　GTK+



DOSやWindowsで作成されたフロッピーを、mtools（mdir／mcopy／mdelなど）を呼び出すことによりGUIで扱うファイル管理ソフト。フロッピーをマウントすることなくファイルリストを表示し、コピーや削除を行える。ウィンドウにはファイルリストの表示エリアが2つ用意されている。DOSのフロッピーをFDDに挿入した後で、どちらかのエリアの右上のリストボックスを「Hardisk」から「a:」に切り替えると、ファイルリストがフロッピーの内容に切り替わる。VFATに対応しているため、Windows 9xの長いファイル名も表示可能だ。コピーや削除の際は、対象となるファイルをクリックで選択してから、中央の[←]／[→]ボタンを押すか、[File]メニューから選択する。

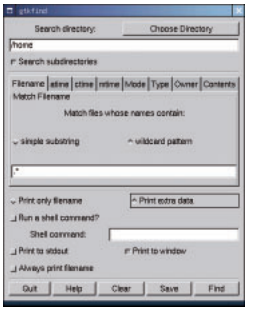
強力なファイル検索をGUIで行なう

gtkfind

バージョン：1.1　　種別：GPL

http://www.oz.net/~mattg/download.html

X　GTK+



ファイル名、日時、属性、サイズなど、さまざまな条件によるファイル検索を行なうソフト。初心者でもGUIを使って複雑な検索条件を設定できる。ウィンドウには、上部に検索ディレクトリ、中央部にさまざまな検索条件を設定する「カード」、下部には検索後の処理の選択や検索開始用ボタンなどが並んでいる。それぞれの「カード」では、findで指定可能な条件のほか、「指定した文字列を含むファイルを検索する」というgrep風の条件も設定できる。検索結果に対しては、ファイル名などの情報をウィンドウに一覧表示するだけでなく、外部コマンドを実行したり、標準出力に出力して他のコマンドとパイプで接続するなど、さまざまな使い方が可能だ。

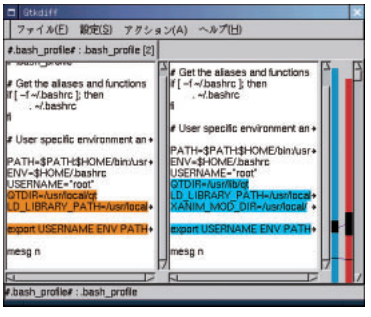
2つのファイルの違いをカラー表示

gtkdiff

バージョン：1.0.1　　種別：GPL

http://www.ainet.or.jp/~inoue/software/gtkdiff/index.html

X　GTK+　GNOME　日本語可



2つのファイルの内容を並べて表示し、内容の異なる行をオレンジとシアンの背景色で表示するソフト。ウィンドウ右側には、ファイル全体での相違点の位置と対応関係がオレンジ／シアンのバーとそれを結ぶ線で図示される。また、画面が狭い環境に向いた1画面表示では、2つのファイルの内容を1つにまとめ、相違点だけ背景色を変えて上下に並べて表示してくれる。プログラムのソースや設定ファイルなど、内容の一部を追加・変更することの多いファイルに対して使うと便利だ。このほか、2つのディレクトリを比較して、片方にだけ含まれるファイルや、同名で内容の異なるファイルのリストを左右に並べて表示することも可能だ。

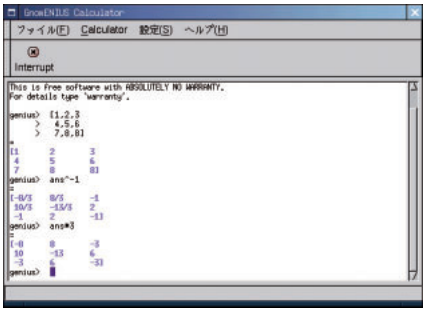
行列計算などが可能な高精度計算ソフト

Genius

バージョン：0.4.6　　種別：GPL

http://www.5z.com/jirka/genius.html

X　GTK+　コンソール　日本語可



有名なCADソフトと同名だが、こちらは高精度な計算ソフトだ。コンソールで動作するgeniusのほか、GUIのフロントエンドgnome-geniusも用意されている。実行にはGNUの数値計算用ライブラリ「GMP」（http://www.swox.com/gmp/）が必要だ。整数や浮動小数点数（精度指定可）はもちろん、無理数や複素数、行列計算にも対応している。たとえば、3÷9の結果は0.333…という浮動小数ではなく3分の1のまま保持されるし、行列aの逆行列は「a^-1」とするだけで求められる。このほか、内蔵のGEL（Genius Extention Language）と呼ばれる言語には豊富な組み込み関数や条件分岐命令などが用意されており、ユーザーが自由に関数を追加できる。

擬似透明化できるターミナル
Eterm

バージョン：0.8.10 **種別：GPL**

http://www.eterm.org/

日本語可

rxvtを元にして、バックグラウンドに画像を貼りつけたり、透明にできるようにしたターミナルエミュレータだ。0.8.9からの違いは主にバグフィックスにとどまっている。－Oオプションをつけて起動すると、デスクトップのバックグラウンドが表示されるので、透明になったような気分になれる（厳密に考えれば透明ではない）。

透明化したEtermの中でviやmuleを起動すると透明エディタになる。また、muleを起動してさらにmewなどのメーラを使うと、透明メーラができあがる。通常muleの背景は色を変えることはできるが、画像を指定するようなことはできないので、この透明エディタや透明メーラは不思議な感じがしてしまう。

グラフィカルなブートマネージャ
GAG

バージョン：2.1 **種別：GPL**

http://raster.cibermillennium.com/gageng.htm

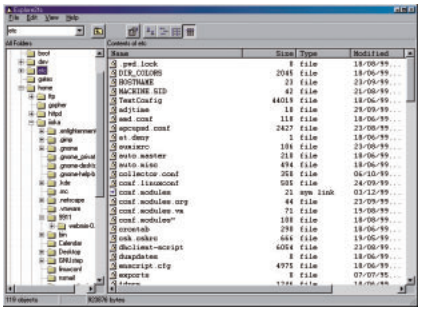
日本語可

Windows／Linux／BeOSなど、9種類のOSに対応したグラフィカルなブートマネージャ。専用のフロッピーやハードディスクのMBR（マスターブートレコード）にインストールすることで、OSの切り替えをアイコンを含んだメニューから行えるようになる。2台目以降のハードディスクからのブートも可能だ。8Gバイトを超えるハードディスクには対応していないので注意されたい。説明は14カ国語版が用意されており、その中にはローマ字表記の日本語版まである。なお、GAG本体はフロッピーディスクイメージとして配布されているので、ハードディスクのMBRにインストールする場合にも、いったん専用フロッピーでGAGを起動してMBRにコピーする必要がある。

WindowsからExt2ファイルシステムを参照する
Expore2fs

バージョン：1.00 pre2 **種別：GPL**

http://uranus.it.swin.edu.au/~jn/linux/explore2fs.htm



今回の特集で唯一のWindowsプログラムだ。Explore2fsは、Windows 9x／NTからLinuxの標準ファイルシステムであるExt2ファイルシステムを参照するプログラムだ。LinuxとWindowsのデュアルブートマシンでWindowsを使っているときに、Linuxパーティション内のファイルが必要になることは多いが、そんな場合に重宝する。

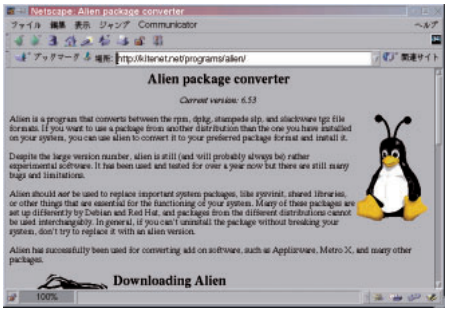
インターフェイスはWindowsのエクスプローラと変わらない。必要なファイルを指定して、マウスの右クリックで表示されるメニューで[Properties]を選ぶと、Ext2上での詳細な情報が表示される。また[Export file]でWindowsパーティションへの保存、[View]で拡張子に応じたプログラムを起動してファイルを見ることができる。

各種パッケージ間の変換を行う
Alien

バージョン：6.53 **種別：GPL**

http://kitenet.net/programs/alien/

コンソール　Perl



Linuxで用いられているパッケージを相互に変換するプログラムだ。Red Hat系のrpm、Debian GNU/Linuxのdeb、Slackwareのtarボールのほか、日本ではあまりなじみのないStampede Linux用のslpパッケージに対応している。

スクリプト言語のPerlで記述されており、実行にはPerlの5.004より新しいバージョンが必要になる。またrpmパッケージを扱うにはrpmコマンドが、debパッケージを扱うにはdpkgなどのコマンドが必要だ。変換されたパッケージをそのままインストールし、終了後に消去することも可能なので、あたかもRed Hat Linuxに直接debパッケージをインストールする感覚で使用できる。

GUIベースの高機能メールクライアント

kmail

バージョン：1.0.11　　種別：GPL

http://www.kde.org/

X　Qt　日本語可

kmailはKDEの標準メールクライアントで、すべての操作をGUIにより行うことができる。MacintoshやWindowsで用いられている一般的なGUIメールクライアントと同様の画面構成となっており、非常に直観的に操作できるので、誰にでも簡単に使えるだろう。

●kmailの特徴

kmailの主な機能としては、

・フォルダ自動振り分け機能
・kfmとの協調動作
・住所録
・複数メールアカウント対応
・PGP暗号化

などが備えられており、通常の使用にはまったく問題がないまでのレベルに仕上がっている。一方で、以下のような問題点も存在している。

1.対応プロトコルがPOP3とローカルスプールのみでIMAP4には非対応
2.検索機能が一切搭載されていない
3.メールのスレッド表示ができない
4.サーバにメールを残した場合、サーバ側の設定次第では何度も同じメールを受信してしまう

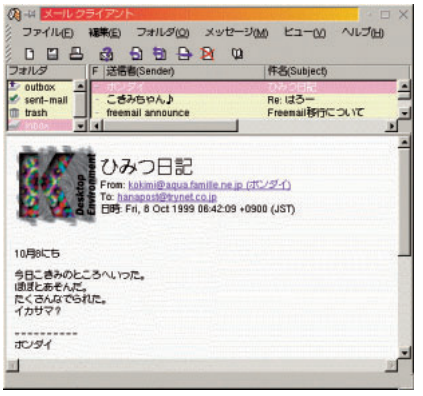
特に4番目は、kmail以外のメーラでもメールを読みたい場合に深刻な問題となりうる。条件によってはkmailの使用を断念せざるをえないこともあるだろう。

●kmailのカスタマイズ

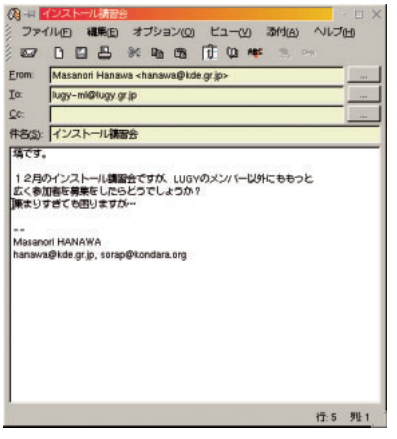
kmailの利用の際にはいくつか注意しなければならないことがある。まず最初にしなければいけないことは、メール送信設定を[MIME対応]に切り替えることだ。初期設定では[8ビットに従う]になっているが、このままではサブジェクトが日本語で書かれている場合に文字化けしてしまう。kmailの最初の起動時の設定ダイアログボックスで[メール作成]タブから、[メールの送信設定]－[送信メッセージ]－[MIME対応(Quoted Printable)]を選択して[MIME対応]に切り替えよう。そこで設定し忘れた場合には、ウィンドウ上部のメニューバーから[ファイル]－[設定]と選べば設定ダイアログボックスが表示されるので、後は上述の通り設定する。ちなみに12月1日に発売されたKondara MNU／Linux 1.0に含まれているKDEパッケージでは、デフォルトでMIME対応になるようにパッチが当てられている。

初期設定では、メール作成ウィンドウでテキストを入力していくと80文字（バイト）で自動的に改行するようになっているが、日本語場合には不具合があるようで、逆にせっかく入力したメールテキストが乱れてしまうことがある。これを避けるためには、設定ダイアログボックスの[メール作成]－[外観]－[1行の折り返し文字数（英文）]のチェックを外しておけばよい。

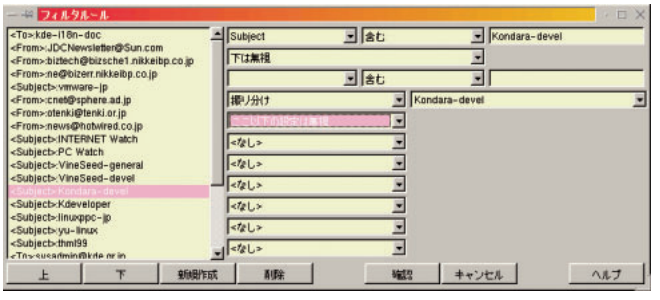
ほかにも、ノートPCを使ったモバイル環境でkmailを利用する場合には、[メール作成]－[メールの送信設定]－[送信の仕方]－[後で送信]をチェックしておけば、オフラインでメールを作成できる。たくさんのフォルダにメールを振り分けて保存している場合には[外観]－[長いフォルダリスト]をチェックして、一度に表示されるフォルダが多くなるようにしておこう。



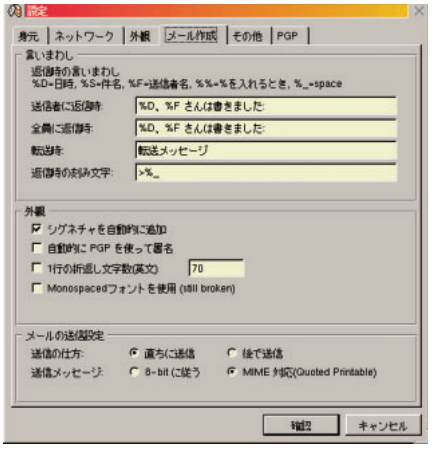
画面1 kmailウィンドウ。



画面2 メール作成ウィンドウ。



画面3 フィルタ設定。



画面4 設定ダイアログボックス。

Emacs上で動作する先進的なメールクライアント

Wanderlust

バージョン：1.0.3（安定版）／2.2.10（ベータ版）　種別：GPL

http://www2.tky.3web.ne.jp/~metal/vMacj.html
http://www2.tky.3web.ne.jp/~metal/vMacj.html

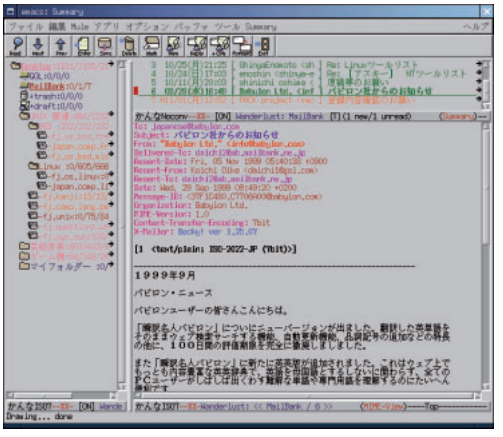
X　日本語可

Emacs／XEmacs／Mule上で動作する先進的なメッセージ管理ソフト。POP3／IMAP4やMH形式対応のメールクライアントとして、またNNTP対応のニュースリーダとして日本語を含むメッセージを統一的に扱える。XEmacsではツールバーやフォルダアイコンなどがグラフィックで表示され、マウスによる操作も可能だ。なお、Emacs-lispのみの実装でMHなどの外部ソフトは不要だが、MIMEモジュール（SEMIかtm）をあらかじめインストールしておく必要がある。

Emacs上で「M-x wl」としてWanderlustを起動すると、美しいロゴが表示された後で「フォルダモード」に移行する。ここでは、メールやニュースなどのメッセージが格納されるフォルダがツリー状に並べられている。POP3／IMAPのメールスプールもフォルダとして扱われるので、いちいちローカルなフォルダ（+inboxなど）にメールを取り込む必要はない。また、ユーザーがフォルダ構成を変更したり、新規フォルダを追加することも可能だ。

フォルダ上でスペースキーを押すと「サマリーモード」に移行する。ここでは、フォルダ内のメッセージがスレッド順に一覧表示され（読んでいるスレッドのみ開く）、スペースキーを押すだけで未読メッセージを読み進められる。また、マルチパートMIMEによる添付ファイルにも対応しており、XEmacsでは添付画像がインライン表示される。このほか、メッセージの振り分け（リファイル）の際には、同じ差出人や同一スレッドのメッセージを以前移動したフォルダに簡単に移動できる学習機能が備わっている。一方、メッセージの作成時にはEmacsの強力な編集機能をそのまま利用できる。もちろん、マルチパートMIMEによるファイルの添付も可能だ。なお、オフライン状態で作成したメッセージはキュー（+queue）に保存され、オンラインになった時点で一括送信される。

このほか、ZIP／LHAなどのアーカイブを通常のフォルダのように扱うアーカイブフォルダや、古くなったメッセージを自動的に削除（またはアーカイブフォルダに移動）するエクスパイア、差出人などに応じて各メッセージに点数を付けてSPAMの排除を行うスコアリングなど、便利な機能が豊富に用意されている。



画面1 XEmacsではツールバーやフォルダのアイコンが表示される。

2パネル構成の高機能なFTPクライアント

IglooFTP PRO

バージョン：0.9.8　　種別：シェアウェア（29.95ドル）

http://www.littleigloo.org/

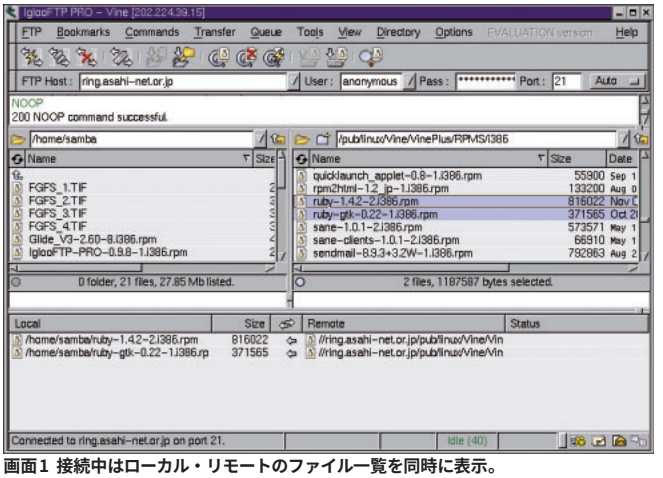
X　GTK+　日本語可

ローカルとリモートのファイル一覧が同時に表示されるFTPクライアントソフト（「イグルー」はエスキモーが作る氷のドームのこと）。ツールバーや右クリックメニューによる操作、FTPサイトのブックマークによる管理、あらゆる設定をGUIで変更できる設定ダイアログなど、とても使いやすいソフトに仕上がっている。試用期間30日のシェアウェアだが、機能制限はいっさい設けられていない（フリー版のIglooFTPも配布されている）。ソースが提供されないと、日本語の表示や入力への対応が気になるところだが、GTK+のlocale設定が正しく行われており、そのままの状態でファイル一覧や内蔵のビューア、エディタで日本語を扱える。

ウィンドウは複数のパネルで構成されており、左側のパネルにはローカルなディレクトリのファイル一覧、右側のパネルにはFTPサイトのブックマークがフォルダで分類されてツリー表示されている。すでに、各種のディストリビューションやGnomeプロジェクトなどのFTPサイトのブックマークが登録済み。もちろん、ユーザーによるFTPサイトの登録やフォルダの追加も可能だ。

登録済みのFTPサイトについては、ブックマークをダブルクリックするだけで接続できる。一時的に訪れるFTPサイトの場合は、ホスト名やアカウントなどの情報をツールバー下のエリアに直接入力すればいい。クリップボードにコピーされたURLに接続する機能も用意されている。FTPサイトへの接続が成功すると、ウィンドウ右側のパネルがブックマークのツリー表示から接続先のFTPサイト（リモート）のファイル一覧表示に切り替わる。

転送したいファイルを左右のパネルから選択してツールバーのボタンを押す（あるいは右クリックメニューから選択する）とファイル転送が行われる。また、転送したいファイルのリストを画面下の「キュー」に溜めておき、後でまとめて一気に転送することもできる。このほか、ローカル・リモートどちらのファイルに対しても、右クリックメニューを使ってファイルの閲覧や編集、削除、属性変更などのファイル操作が可能だ。ちょっとしたファイルの編集には十分に使えるエディタも内蔵されている。



画面1 接続中はローカル・リモートのファイル一覧を同時に表示。

日本語対応のGNOME版ICQクライアント

GnomeICU

バージョン：0.67／0.65ja5（日本語版）種別：GPL
http://gnomeicu.gdev.net/
http://northeye.org/gnomeicu-ja/（日本語化）

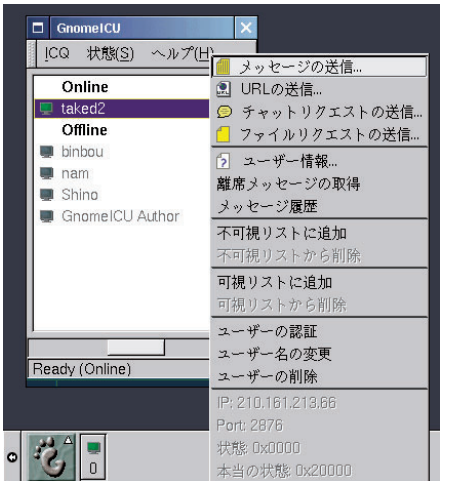
X　GTK+　GNOME　日本語可

ユーザー数が2000万人を超えるコミュニケーションソフト「ICQ」互換のクライアントソフト。メッセージ交換やチャット、ファイル転送など、ICQの機能をほぼサポートしている。ふだんはGNOMEのパネルに常駐しており、ダブルクリックするとユーザー一覧を含むウィンドウが開く。なお、オリジナルのGnomeICUは日本語を想定していないため、日本語のメッセージを日本語EUCのまま送信してしまう。WindowsのICQクライアントと日本語でメッセージをやりとりするには、日本語化パッチをソースに当てて、送受信時の文字コード変換などの修正を行う必要がある。

起動すると、GNOMEのパネルに小さなボタンが表示される。GnomeICUのウィンドウを開くには、このボタンをダブルクリックすればいい。初めて起動した場合は、UIN（ユーザー識別番号）の入力を求められる。ICQの特徴はこのUINによるユーザー管理にあり、どこから接続しても同じUINなら同一ユーザーだと判断され、同じマシンからでもUINを切り替えれば別のユーザーとして扱われる。GnomeICUでは、すでに持っているUINを設定するだけでなく、ICQの新規ユーザーとして新たなUINを取得することも可能だ。

続いての作業は、ICQを使っている友人などをコンタクトリストに登録すること。GnomeICUのウィンドウには、コンタクトリストに登録したユーザーだけが表示され、メッセージの交換やチャットなどを行える。相手のUINを直接入力するほか、（少し時間がかかるが）名前やメールアドレスによる検索も可能だ。登録が完了すると、そのユーザーのUINがウィンドウに表示され（しばらくするとニックネームに変わる）、ICQを使用中なら[Online]に、そうでなければ[Offline]に並んで接続状況を知らせてくれる。

コンタクトリストに登録したユーザーとは、自由にメッセージの交換やチャット、ファイル転送を行える。こうした操作には、マウスの右クリックメニューを利用する。チャットやファイル転送については、相手が了承した場合にのみ実際の処理が行われる。なお、ICQのシステム自体の制限により、チャットでは日本語が使えないので注意されたい。



画面1 友達などが現在オンラインかどうか一目でわかる。

GTK+を利用した高機能なIRCクライアント

X-Chat

バージョン：1.2.1（安定版）／1.1.3（日本語版）／1.3.8（開発版）　種別：GPL
http://xchat.linuxpower.org/
http://www.on.cs.keio.ac.jp/~yasu/jp_gnome.html（日本語化）

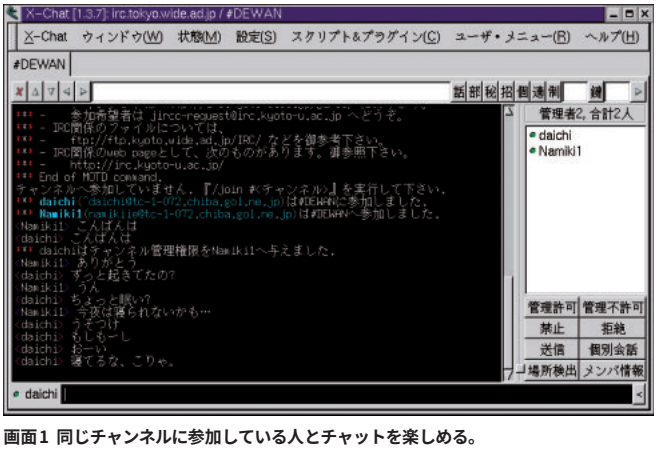
X　GTK+　日本語可

GTK+を利用してX上で動作する最新のIRCクライアントソフト。「IRC（Internet Relay Chat）」は、TCP／IP環境で動作するサーバクライアント方式の分散型マルチユーザーチャットシステム。X-ChatなどのIRCクライアントを利用して、インターネットやイントラネットに用意されたIRCサーバに接続し、チャンネルに参加（あるいは自分でチャンネルを作成）して、同じチャンネルに参加している複数の相手と文字による会話を楽しめる。

X-Chatでは、接続先のIRCサーバはグループ別に分類管理されており、ダブルクリックするだけで接続できる。初期設定では海外のサーバしか登録されていないので、国内のIRCサーバ（irc.tokyo.wide.ad.jpなど）を新規登録して接続するといい。なお、こうしたサーバの多くは互いに連携してひとつのシステムとして動作するため、国内のサーバに接続しても世界中の人間と会話できる。また、本来なら、チャンネルへの参加や参加者の確認、ニックネームの変更などの際に「/」で始まるコマンド（/joinなど）を入力する必要がある。しかし、X-Chatでは、ツールバーのボタンや右クリックメニューでコマンドを入力でき、チャンネルの参加者一覧は常にウィンドウ右側に表示されているので、IRCに慣れていない人にも使いやすいだろう。

なお、IRCでは日本語の文字コードにJISを使うことになっているため、X-Chatを使って日本語チャットを楽しむには日本語パッチ（1.1.3用）を当てる必要がある。Kondara MNU/Linuxなど、1.3.x系のソースにパッチ済みのRPMパッケージを持つディストリビューションもあるようだ。なお、日本語パッチは単に文字コードの変換機能を追加するだけではなく、メニューなどの日本語化も行ってくれる。

このほか、IRCサーバを介さずに他のIRCクライアントと直接接続し、セキュリティ的に安全な状態でチャットやファイル転送を行える「DCC（Directly Connect Client）」、複数のIRCコマンドを組み合わせて新しいコマンドを作成できるユーザー定義コマンド、Perlスクリプトやプラグインモジュールによる機能拡張など、上級者向けの機能も用意されている。設定によっては、ウィンドウの背景表示を透明化することも可能だ（ZVTが必要）。



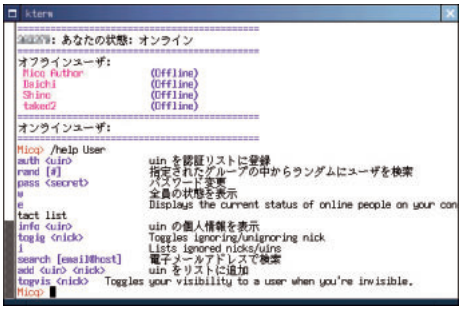
画面1 同じチャンネルに参加している人とチャットを楽しめる。

テキストベースで軽快に動作するICQクライアント
micq

バージョン：0.4.3／0.4.0（日本語版）　種別：フリーソフト

http://phantom.iquest.net/micq/
http://www.tahoo.org/~taku/unixicq.html （日本語化）

コンソール　日本語可



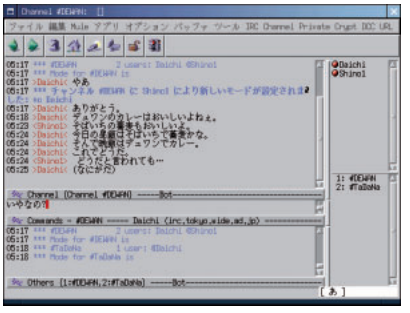
ktermなどの端末画面で動作する軽量なICQクライアントソフト。画面制御は行わずメッセージが流れるだけなので、非力なマシンでも軽快に動作する。自分の状態の変更、リストに表示するユーザーの登録、メッセージ交換などの機能はすべてコマンドラインからのコマンド入力で行う。また、初回起動時にUIN（アカウント用の番号）を入力しなかった場合、ICQユーザーとしての新規登録を自動的に行ってくれる。なお、ktermをシフトJIS漢字モードに設定すれば、オリジナルのまま日本語メッセージのやりとりが可能。日本語EUCモードでmicqを利用したいなら、送受信時の文字コード変換とメッセージを日本語化するパッチを当てる必要がある。

Emacs上で動作する国産のIRCクライアント
Liece

バージョン：1.3.7（開発版）／1.2.8.25（安定版）　種別：GPL

http://www.unixuser.org/~ueno/liece/index.ja.html

日本語可



Emacs／XEmacs／Mule上で動作する国産のIRCクライアントソフト。最初から文字コード変換機能が備わっていて、パッチなどで修正することなく日本語を扱える点がうれしい。Lieceの特徴は、画面（バッファ）の分割を「スタイル」と呼ばれる機構で柔軟に設定できることにある。付属の3スタイルを切り替えるほか、自分でスタイルをカスタマイズすることも可能。XEmacsでは、ツールバーのボタンやマウスの右クリックメニューによってIRC用のコマンド入力を行えるので、Emacs系エディタのキー操作に慣れていない人でも簡単にチャットを楽しめる。もちろん、「C-c j」でチャンネルに参加といったキー操作を駆使して、バリバリにチャットすることも可能だ。

サーバいらずのメッセージ交換ソフト
GtkIpmsg

バージョン：0.2.1　　種別：GPL

http://free.prohosting.com/~nobori/

X　GTK+　日本語可

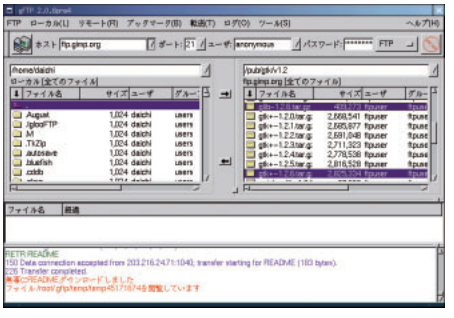
シンプルなメッセージ送受信ソフト「IP Messenger」のGTK+版。専用のサーバを用意する必要がなく気軽に使えるのが特徴で、イントラネットでのメッセージ交換に威力を発揮する。また、Windows版、Mac版、X11R4版、Java版など各種OSのIP Messengerと相互にメッセージをやりとりできる。ふだんはアイコン風のウィンドウが表示されており、これをクリックするとIP Messengerを使用中のユーザーの一覧が表示される。操作は簡単で、送信先ユーザーを選択してメッセージを入力し、送信ボタンを押せばいい。相手の画面には、あなたのニックネームや送信日時とともにメッセージが表示される。その場で返信を書くことも可能だ。

2パネル構成のシンプルなFTPクライアント
gFTP

バージョン：2.0.6pre4　　種別：GPL

http://gftp.seul.org/

X　GTK+　日本語可



ローカルとリモートのファイル一覧が左右に表示されるシンプルな2パネル構成のFTPクライアント。日本語カタログが用意されており、日本語環境ではメニュー表示やオプション設定ダイアログ、各種のメッセージに至るまで日本語で表示される。FTPサイトに接続するには、ホスト名などを直接ツールバーのエリアに入力するか、あらかじめブックマークに登録しておいてメニューから選択する。ブックマークはフォルダによる分類が可能だ。接続後は、一覧からファイルを選択して中央部の矢印ボタンを押すだけで、即座にダウンロード・アップロードが行われる。このほか、右クリックメニューなどにより、閲覧・編集・削除といったファイル操作も可能だ。

ファイル名補完やカラーls機能付きのFTPクライアント

yafc

バージョン：0.5.0　　**種別：GPL**

http://www.stacken.kth.se/~mhe/yafc/

コンソール　日本語可

コンソールベースで動作する軽快なFTPクライアントソフト。ftpコマンドに似たシンプルなインターフェイスながら、ファイル名補完やエイリアス、カラーlsなどの便利な機能・コマンドが満載だ。たとえば、リモート（FTPサーバ上）のファイルに対してもタブキーによるファイル名補完が働くし、「ls --color」とすると、リモートのファイル名がカラー表示される。「ls」とするだけでカラー表示を行うには、エイリアス機能を使って「alias ls ls --color」とすればいい。指定したディレクトリ以下のファイルをまとめて送受信することも可能だ。このほか、接続手続きを簡略化するブックマークやオートログイン機能が用意されている。

WebやFTPサーバからファイルを取得する

wget

バージョン：1.5.3　　**種別：GPL**

http://www.gnu.org/software/wget/wget.html

コンソール

HTTP／FTPプロトコルを利用してファイルを自動的にダウンロードするコンソールベースのツール。Webページのリンクを再帰的にたどって関連したページや画像をまとめて取得したり、FTPサーバのディレクトリ内の全ファイルを一気に取得できる。ファイルはURLを反映したディレクトリツリーに格納される。再帰的な取得の有無やその際にたどるリンク数、対象とする（あるいは外す）ドメインや拡張子、ファイルの保存先、同名ファイルを上書きするかなど、設定はすべてコマンドラインオプションで指定する。また、こうしたオプションをGUIで設定する「gwget」（http://usuarios.meridian.es/frimost1/gwget/gwget.htm）も作られている。

Webサイトのページをまとめて取得する

WWWcp

バージョン：1.91　　**種別：フリーソフト**

http://www.ff.iij4u.or.jp/~rewsirow/WWWcp/WWWcp.html

コンソール　Perl

HTTPプロトコルを利用してWebページを自動巡回するPerlスクリプト。コマンドラインで指定したURLのファイルを取得した後、そこに含まれるリンクを再帰的にたどって、そのサイトのページや画像をすべて取得する。初期設定では、最初に指定したURLのファイル名の直前までを「基準パス」とし、それを含まないリンク先は無視するため、外部サイトへのリンクはたどらない（変更可）。一方、基準パスにマッチするパスはすべて取得し、URLを反映したディレクトリツリーに保存する。起動時オプションや環境変数により、対象外とするファイル、プロキシサーバの利用、漢字コード変換、受信を中断した部分から再開するかなどを設定できる。

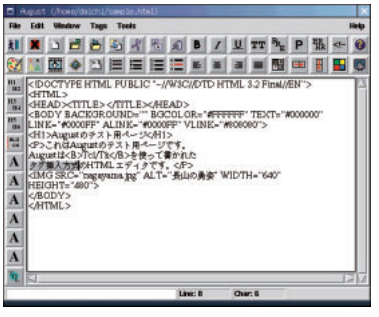
Tcl/Tkで書かれたHTMLエディタ

August

バージョン：0.50　　**種別：GPL**

http://august.seul.org/

X　Tcl/Tk　日本語可



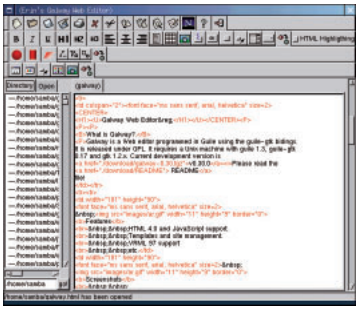
Tcl/Tkで書かれたタグ挿入型のシンプルなHTMLエディタ。日本語化されたTcl/Tk8.0jpを使用すれば、ソースを変更することなく日本語を入力・編集できる。カラー構文表示などは備わっていないものの、ツールバーのボタンでタグを効率よく入力でき、ブラウザを起動して実際の表示を確認できる。使い方は一般的なタグ挿入型のHTMLエディタと同じで、ツールバーのボタンを押したり、[Tags]メニューの項目を選択すると、対応するタグがカーソル位置に挿入される。一部のボタンは、タグで囲むテキストを選択した状態で押さないと機能しないので注意されたい。画像やテーブルなど属性の指定が複雑なタグについては専用ダイアログが用意されている。

カラー構文表示のHTMLエディタ
galway

バージョン：0.30　　**種別：GPL**

http://erin.netpedia.net/galway.html

X　GTK+



GNUのscheme言語「guile」で記述されたタグ挿入型のHTMLエディタ。実行にはguileとguile-gtk、GTK+が必要だ。日本語は扱えないものの、最新のHTML4.0に対応しており、カラー構文表示によりタグなどが別色で表示される。左側にはカレントディレクトリや編集中のファイルの一覧が表示され、クリックするだけでオープンや切り替えが可能だ。使い方は一般的なタグ挿入型のHTMLエディタと同じで、ツールバーのボタンを押すと、対応するタグがカーソル位置に挿入される。また、画像やテーブル、リンクなどについては、専用ダイアログで属性を指定できる。このほか、JavaScriptやVRML2.0もサポートしており、スタイルシート（CSS）にも対応する予定だ。

Webページのリンク切れをチェック
LinBot

バージョン：1.0　　**種別：GPL**

http://starship.python.net/crew/marduk/linbot/

コンソール

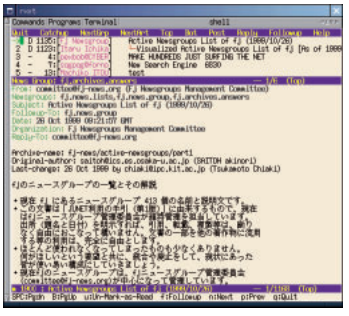
Webページのリンク切れなどを調べてくれるツール。使い方は簡単で、コマンドラインでチェックするURLを指定してLinBotを起動すればいい。自動的にリンクをたどって複数のWebページを調べ、それぞれのリンク先のページが存在するか、作成日時が古いものはないか、受信速度が遅いものはないかなどを調べてくれる。リンクをたどる数や、外部リンクのチェックの有無、外部リンクとみなす文字列パターン、チェック結果の保存先などの設定はすべてコマンドラインオプションで行う。チェック結果はHTMLファイルで出力されるので、リンク切れ、古いページ、新しいページ、遅いページなどの状況を、Webブラウザで閲覧できる。

Gnusに似たインターフェイスを持つ高速ニュースリーダ
slrn

バージョン：0.9.5.7　　**種別：GPL**

http://space.mit.edu/~davis/slrn.html
http://kondara.sdri.co.jp/~kikutani/slrn.html （日本語化）

コンソール



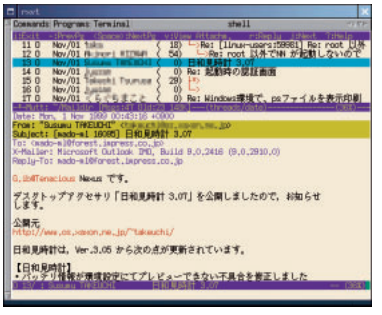
テキストベースで動作する高速なニュースリーダ。画面表示はEmacs上で動作するGnusに似ており、記事のスレッド表示にも対応している。S-langと呼ばれるC風のマクロ言語を利用して、端末ソフトごとの文字色指定や、細かなキー設定のカスタマイズが可能だ。基本的にはGnus準拠のキーバインドで、スペースキーだけで記事を読み進められる（マウス操作にも対応）。また、日本語化パッチを当てることにより、日本語の記事の表示、ヘルプメッセージの日本語化、マルチパートMIME対応、クリックカブルURLなど、日本語対応と各種の機能拡張が行われる。このほか、SPAMメッセージを排除するのに便利なスコアファイル機能や、ニュースサーバから記事をまとめて取得するツール

柔軟にカスタマイズできるメールクライアント
mutt

バージョン：1.0/1.0pre3i（日本語版）　　**種別：GPL**

http://www.mutt.org/
http://kondara.sdri.co.jp/~kikutani/mutt.html （日本語化）

コンソール



テキストベースで動作する高速なメールクライアント。POP3のほかIMAP4プロトコルにも対応し、マルチパートMIMEやPGPによる暗号化など高機能なのが特徴だ。画面表示はslrnと似ており、サブジェクトなどの一覧が表示されるインデックス画面と、メールの内容が表示されるページャ画面で構成される。日本語を扱うには、日本語対応パッチを当てたmuttと日本語化されたS-Langを組み合わせる必要がある。S-langを利用して、スレッド表示や各部の表示色、キーバインド、マクロなどの柔軟なカスタマイズが可能だ。なお、muttでは従来のmbox／MH形式も扱えるが、クラッシュや排他制御、セキュリティなどの問題が解決されるMaildir形式の利用をお勧めする。

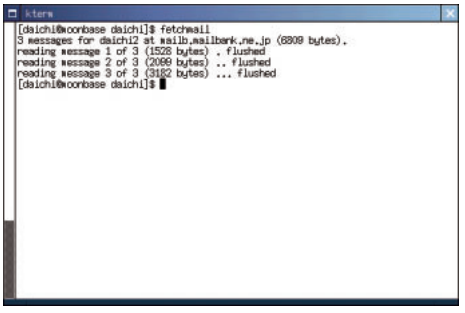
通信

リモートのメールサーバからメールを一括転送
fetchmail

バージョン：5.2.0　　種別：GPL

http://www.tuxedo.org/~esr/fetchmail/

コンソール



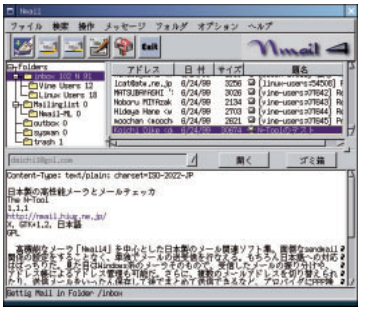
リモートのメールサーバからPOP3／IMAP4プロトコルなどを使ってメールを取得し、ローカルのメールスプールや別のメールサーバに転送してくれるツール。一定時間ごとにメールを取得するデーモンモードも用意されている。プロバイダにPPP接続しているユーザーがオフラインでメールを読む場合や、複数のメールサーバのメールを1つのメールサーバに集める場合などに便利だ。設定ファイルには、メールサーバ名やプロトコル、サーバ用アカウント・パスワード、メールの送付方法などを指定する。ローカルのメールスプールに転送する際に、MTA（sendmailなど）は不要だ。サーバにメールを残したり、複数のサーバからメールを取り込むこともできる。

Windowsライクな日本製メールクライアント
The N-Tool

バージョン：1.1.2　　種別：GPL

http://nmail.hiug.ne.jp/

X　GTK+　日本語可



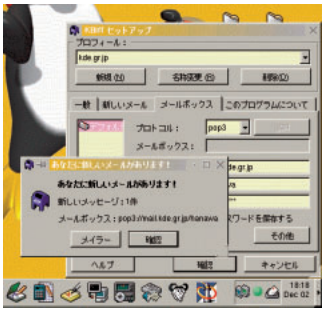
多機能なメールクライアント「Nmail4」を中心とした日本製のメール関連ソフト集。画面はWindowsなどでよく見られる3ペイン方式で、受信したメールのフォルダへの振り分けや、アドレス帳によるアドレス管理も可能だ。さらに、複数のメールアドレスを切り替えて使ったり、送信メールをいったん保存してまとめて送信するなど、プロバイダにPPP接続してインターネットを利用するユーザーに便利な機能が用意されている。メールの作成に関しては、同梱の簡易エディタ「マルチメディアエディタ」のほか、Emacsなどの外部エディタを利用することも可能。このほか、POPサーバに接続して新着メールをチェックするpopcheckerなどが含まれる。

新規メールの到着を絵と音で通知
KBiff

バージョン：2.3.10　　種別：GPL

http://www.pobox.com/~kurt_granroth/kbiff/

X　Qt



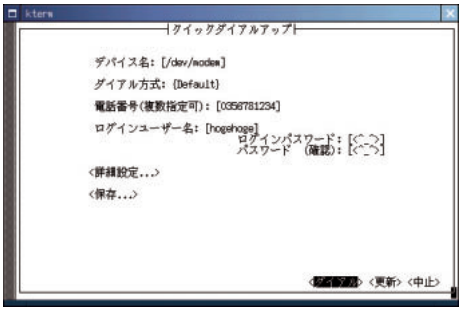
サーバ（またはローカルホスト）に新しいメールが到着すると、知らせてくれるソフトだ。ダイアログボックスにメッセージが表示され、パネル上のドックアイコンが変化しする。ほかにアニメーション表示や、サウンドファイル（WAVファイル）の再生も行える。ドックアイコンをクリックすれば登録してあるメーラ（デフォルトはkmail）が起動し、直ちにメールを読める。対応しているメールボックス形式は、ローカルmbox、qmailのmaildir、MH、POP3、IMAP4、NNTPと、通常用いられている形式はすべてサポートされており、同時に複数のメールボックスを監視することも可能。KDEのセッションマネジメントにも対応しているので、常にパネル上に常駐させておこう。

多機能で使いやすい国産PPP接続ツール
PPxP

バージョン：0.99072807　　種別：フリーソフト

http://www.dsl.gr.jp/~manabe/PPxP/

コンソール



多機能で使いやすいPPP接続ツール。通信が必要になった時点でPPP接続を確立するオンデマンド機能や、IPマスカレード機能、IPパケットフィルタ機能などを備えている。PPPを実装したppxpdデーモンと、インターフェイスを受け持つコンソールが独立しているのが特徴だ。標準コンソールのppxpコマンドのほか、Xpm版のxppxp、Tk版のtkppxp／qdialなど各種のコンソールが用意されている。ppxpコマンドのクイックダイヤルアップ機能を使うと、画面上で電話番号やPPP接続用のログイン名、パスワードなどを設定するだけでPPP接続を行える。設定をスクリプトファイルに保存しておけば、次回からは起動時にそのファイルを指定するだけでいい。

世界各地を飛行できるフライトシミュレータ

Flight Gear Flight Simulator

バージョン：0.7.1　　種別：GPL

http://www.flightgear.org/

X　Mesa

Linuxや＊BSD、Windows 9x／NT、Macintoshなどマルチプラットホームで動作するオープンソースのフライトシミュレータ。ちゃんとしたフライトモデルを用いるシミュレータなので操縦は結構難しいが、そのぶんリアルな機体の挙動を楽しめる。また、オプションとして世界各地のシーナリーデータが用意されている。OpenGL（または互換ライブラリのMesa）を利用するため、快適な速度でプレイするには、3Dアクセラレータカードが必須だ。

●パッケージとライブラリ

Flight Gear Flight Simulatorはソースファイル一式と、実行に必須のベースパッケージ、ドキュメントがそれぞれ別のtarボールで配布されている。このほかに3DグラフィックスライブラリのOpenGL（またはMesa、GLUT）、ポータブルなゲームライブラリplib（http://www.woodsoup.org/projs/plib/）が必要だ。なお、Red Hat 5.2系のディストリビューションで正常にビルドするには、libstdc++を2.9.0以降に更新しておく必要がある。

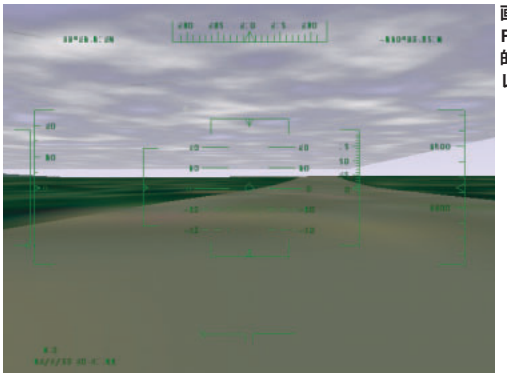
●キーボードで機体を操作

起動すると、現在時刻のアリゾナの空港からシミュレーションが開始される（**画面1**）。飛行機の操作には、キーボードかジョイスティックを利用する（キーボードのみの操作も可能）。よく使うキーはテンキー周辺に配置されており、NumLockを外した状態で使用する。基本的には、↑／↓キーでエレベータ（上下の向き）、←／→キーでエルロン（左右の傾き）、Insert／Enterキーでラダー（左右の向き）を調整する。スロットル（加減速）の調整はPage Up／Downキーで行う。現在の状況はHUD（ヘッドアップディスプレイ）のインジケータ、あるいはパネルの計器で確認できる。このほか、視点の変更やオートパイロットなどの機能も用意されている。

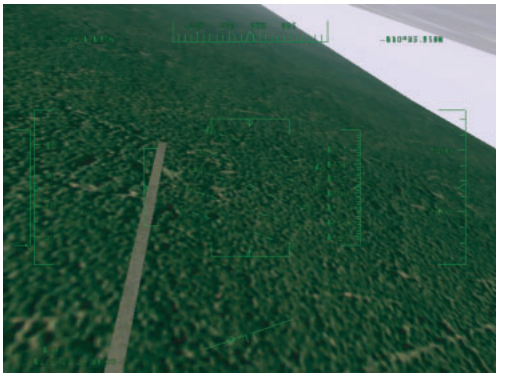
空港から離陸するには、スロットルを開いてスピードを上げつつ、ラダーを操作して滑走路に並行に走るように調整、離陸可能なスピードに達したらエレベータを少しだけ操作して機首を上に向け、あとはこれを維持すればいい。こう書くと簡単そうだが、最初のうちは離陸するだけでも結構難しい（**画面2**）。地面に墜落すると機体が逆さになって操作不能になるので、[File]－[Reset]を選択してやり直そう。なお、手軽に空の散歩を味わいたいなら、機体の高度や速度、日時などを起動時オプションで指定して、昼間に飛行中の状態でゲームを始めるといいだろう。

●豊富に用意されているシーナリーデータ

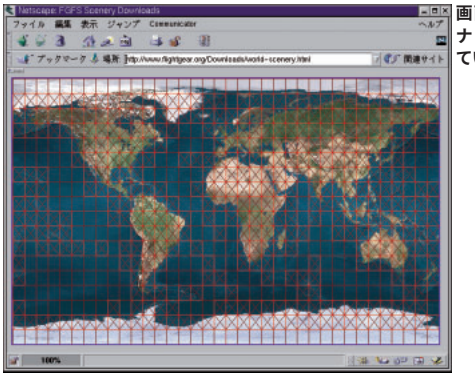
Flight GearのWebサイトには、全世界を緯度・経度でメッシュ状に分割したシーナリーデータが用意されており（**画面3**）、世界各地でのフライトを楽しめる（**画面4**）。ベースパッケージに付属するシーナリーデータはごく狭い地域のものなので、飛んでみたい地域のデータをダウンロードして追加しよう。たとえば、日本の場合はe130n30、e140n30、e140n40あたりをダウンロードして（合計17Mバイト程度）、/usr/local/lib/FlightGear/Scenery以下に追加する。あとは、プレイ開始時の位置を決める空港コードや緯度・経度を起動時オプションで指定すればいい。たとえば、成田空港から開始する場合は「--airport-id=RJAA」とする。ビルド時に作成されるツールを用いて、新たなシーナリーデータを作成することも可能だ。



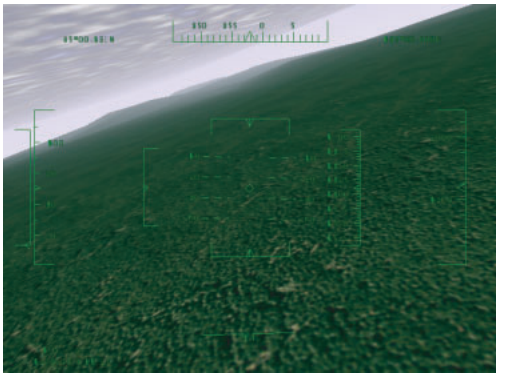
画面1
Flight Gearは本格的なフライトシミュレータだ。



画面2　離陸したアリゾナの空港を上空から眺める。



画面3　世界各地のシーナリーデータが用意されている。



画面4　日本のシーナリーデータを追加して飛行中。

雪山を舞台にした3Dレースゲーム

XRACER

バージョン：0.94　　種別：GPL

http://xracer.annexia.org/

X　Mesa

雪山に設置されたコースを浮遊式クラフトで駆け巡る3Dレースゲーム。コースやクラフトのデザインはシンプルだが、低い視線がもたらすスピード感はなかなかのもので、自動操縦や加速といったパワーアップアイテムも用意されている。なお、OpenGL互換ライブラリのMesa 3.0を利用して3D表示を行うため、実用的な速度でプレイするには、3Dアクセラレータ（Voodoo系など）が必須だ。

XRACERは、ソースとデータが別々に配布されている。実行の際は、環境変数XRACER_HOMEにデータを展開したディレクトリを「export XRACER_HOME=/usr/local/lib/xracer」などとして設定する必要がある。ライブラリ関係では、Mesa 3.0本体とGLUTが必要だ。これらのライブラリは、手持ちの3Dアクセラレータを利用する状態でビルドされている必要がある。詳細は「The Mesa 3D Graphics Library」（http://www.mesa3d.org/）などを参照されたい。

起動するとメニュー画面が表示され、Sキーでプレイ開始だ。画面サイズ（初期設定は640×480ドット）は、起動時に-sオプションで変更できる。たとえば、1024×768ドットの画面でプレイするには、「xracer -s 1024x768」とすればいい。なお、プレイ直前に異常終了してしまう場合は、サウンドドライバ（esd）の有無を確認しよう。

操作にはキーボード・マウス・ジョイステックを利用できる。細かなハンドル操作を行うにはマウスかジョイステックが必須だ。マウスを使うには、プレイ中の画面でDキーを一度押せばいい（再度押すとキーボード操作に戻る）。マウスの左右の動きがハンドル操作、左ボタンがアクセル、右ボタンがブレーキだ。

降りしきる雪の中を猛スピードで走り抜けるには、コースを熟知していることはもちろん、コース内にいくつか設けられている加速ゾーンやアイテムゾーンをうまく通過し、それらを活用する技術も重要だ。アイテム（シールド・加速・自動操縦など）を取得すると、そのシンボルが画面上部に表示され、スペースキーを押した時点で使われる。レースは同じコースを3周するまで続けられ、ゴール時にあなたの順位や合計タイム、ラップタイムなどが表示される。今後はネットワークへの対応なども行われる予定だ。

画面1 激しい吹雪の中、猛スピードでコースを駆け抜ける。

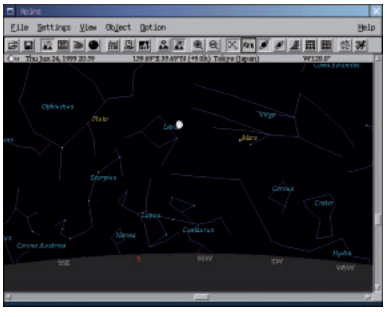
天文シミュレータ

Xplns

バージョン：3.0.1　　種別：フリーソフト

http://www.astroarts.co.jp/products/Xplns/index-j.html

X



商用の高性能天文シミュレータ「ステラナビゲータfor Windows」の開発元、アストロアーツの社員である安喰氏が作製した天文シミュレータ。商用コードを含んでいるため、バイナリ配布のみ行われている。実行にはMotifまたはLesstifが必須なので、これらをインストールしていない場合はMotifを含んだスタティックリンク版Xplnsを利用しよう。

6.5等星までの恒星や約1万個の星雲や星団、すべての星座、惑星、太陽、月、四大小惑星、彗星などの天体の運行を正確にシミュレートする。指定した地点の任意の日時の星空を表示したり、一定時間ごとの星空の変化を次々に表示するアニメーションも可能だ。

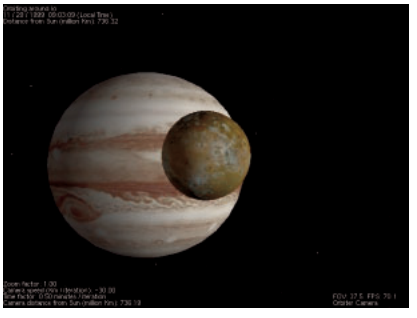
太陽系の惑星や衛星などを3D表示するシミュレータ

Ssystem

バージョン：1.6　　種別：GPL

http://www1.las.es/%7Eamil/ssystem/english.html

X　Mesa



OpenGL（あるいは互換ライブラリのMesa）を利用して、太陽、惑星、衛星、小惑星などの天体を3D表示する太陽系シミュレータ。ソフトウェアのみでも表示できるが、3Dアクセラレータカード（Voodooなど）があればさらにスムーズな表示が可能だ。天体写真を利用したきれいなテクスチャが貼られた天体を、Orbit（軌道上を周回）など4種類のカメラモードで眺められる。初期設定では、一定時間で自動的に天体が切り替わるデモモードで起動するが、キー操作で対象やカメラモードの切り替え、スナップショットの作成などを行うこともできる。

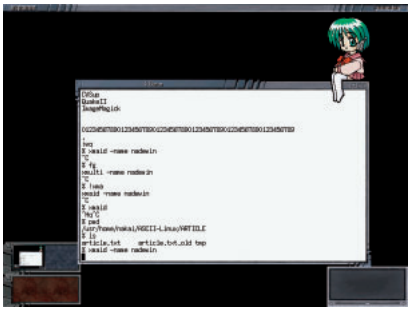
なお、開発中のバージョン2.0では、宇宙船や宇宙ステーションが追加され、さらにテクスチャがグレードアップするようだ。

キュートなデスクトップマスコット
xmulti

バージョン：2.00b4 **種別：GPL**

http://www.denpa.org/~go/xmulti/

X 日本語可



©Leaf

Leafの「To Heart」の人気No.1キャラ「マルチ」が、マウスカーソルを追いかけてくれる。他にもモップを持ってデスクトップを走り回ってくれるおそうじマルチや、アクティブなウィンドウの上に座ってくれるおすわりマルチなどのモードがある。起動するとカレンダーが表示され、もし時間が遅かった場合には「遅くまでたいへんですね」とねぎらいの言葉をかけてくれるのが嬉しい。

なでなでモードでしばらくなで続けるとセリフが変わり、目つきも変わる。隠しコマンドではないが気がつきにくいものだ。

最近では作者が多忙なせいか、バージョンアップが止まっているが、海外のLinuxユーザーにも大変人気のあるソフトウェアである。

デスクトップにペンギンを！
Penguineyes

バージョン：0.9 **種別：GPL**

http://members.xoom.com/nhowie/penguineyes.html

X GTK+

Xeyes（マウスカーソルを見つめる目玉）のGTK+版。Linuxのマスコットキャラとしておなじみ「Tux君」がデスクトップに常駐し、マウスカーソルを目で追いかける。中ボタンでドラッグするとTux君の幅と高さを自由に変更できる。また、右クリックメニューからの選択により、Tux君の目玉の形をハートに変更したり、デザインそのものをGnuのヌーやLinus氏の生首（！）に切り替えることも可能だ。

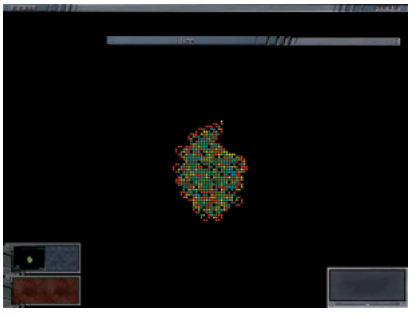
一方、デスクトップ環境としてKDEを使っている人は、ほぼ同様の機能をQtを使って実現した「tuXeyes」（http://kopje.koffie.nu/~ivo/tuXeyes/）を使おう。こちらのデザイン変更はコマンドラインオプションで指定する。

xlockをさらにバリエーション豊かに
XLockMore

バージョン：4.15 **種別：BSDスタイル**

http://www.tux.org/~bagleyd/xlockmore.html

X



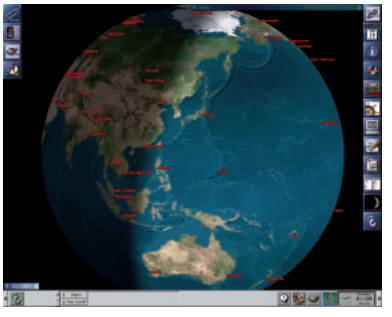
画面ロック機能を持つスクリーンセーバ。起動しておくと、パスワードを入れないとデスクトップに戻らなくなる。いたずら好きな同僚がいる職場に最適だ。画面パターンは数十種類あるので、ランダムで表示させておけば、そのうち気に入ったパターンが見つかるだろう。泡が昇っていったり迷路を自動でたどっていってくれたりというコミカルなものから、幾何学模様を延々と描き続けてくれるものまで揃っており、コマンドラインオプションで指定することができる。ほかにもオプションはいろいろ用意され、ロックをしない「-nolock」もある。また「xlock -inroot」ではかなりCPUパワーを消費するが、デスクトップのバックグラウンドを彩る環境ソフトとなる。

ルートウィンドウにリアルな地球を表示
Xplanet

バージョン：0.42 **種別：GPL**

http://www.alumni.caltech.edu/~hari/xplanet/

X Mesa



Xのルートウィンドウ（背景）に地球の画像を表示するソフト。この手のソフトとしてはxearthが有名だが、Xplanetでは各種のマップ画像を利用して、平面や球面に投影された地球（またはその他の惑星、衛星など）を夜の影や雲も含めてリアルに表示する。使用する画像などはコマンドラインオプションで指定する。なお、アーカイブにはマップ画像は含まれないので、「XGLOBE & XPLANET MAPS」（http://www.radcyberzine.com/xglobe/）などから取得しよう。一定時間ごとにXPlanetを呼び出して再描画を行うxplanetbgコマンドが同梱されているほか、Mesaライブラリを使って通常のウィンドウに表示し、マウス操作で画像を回転させることも可能だ。

美しい盤面のピンボールゲーム

Roll'm Up

バージョン：1.0.6　　種別："as is"

http://www.medialab.lostboys.nl/shortcuts/showcase.html

X

カフェの店内を模した美しい盤面と派手なサウンドが特徴のピンボールゲーム。オランダのLost Boys media labが技術プレビューとして公開しているもので、Linux版のほか、Windows版、BeOS版、Mac版などがバイナリ配布されている。操作にはキーボードを使用する。Nキーでゲームを開始し、Enterキーをしばらく押してプランジャーでボールを打ち出す。左右のフリッパーはそれぞれZキーと／キー、台を揺らすにはスペースキーを押せばいい（揺らしすぎるとティルトになる）。盤面のピアノやビール缶などを指示された順番（ウィンドウ右側中央に表示）で倒していくと、最後には同時に3個のボールを使うマルチボールプレイが可能になる。

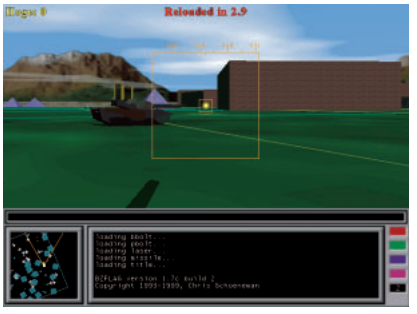
ネットワーク対応の3Dタンクバトルゲーム

BZFlag

バージョン：1.7c　　種別：フリー

http://groundhog.pair.com/bzflag/

X　Mesa



サーバ・クライアント方式のマルチプレイヤー3Dタンクバトルゲーム。TCP/IP接続されたマシンに1人ずつプレイヤーがエントリーし、障害物の置かれたフィールド内で戦闘を繰り広げる。サーバソフトでは、プレイするゲームを「Free for all」（自分以外はすべて敵）と「Capture the Flag」（チーム戦で他チームのフラッグを取り合う）から選択する。また、連続発射可能弾数や各種アイテムの有無も設定可能だ。一方、クライアント側では、画面の品質をオプション設定で調整できるので、ソフトウェアのみの描画でも十分プレイできる。Linux版のほか、Windows 9x／NT版やIRIX版が配布されており、異なるOSのマシン同士で対戦することも可能だ。

1人称型アクションゲーム

QuakeII

バージョン：3.20　　種別：GPL

http://www.idsoftware.com/

有名な商用の3Dバイオレンスアクションゲーム。プレイするには、メインのプログラム（Linux用はフリー）とエリアデータが必要だ。さらにX上で使うには、Mesa（OpenGL互換ライブラリ）がいる。エリアデータは、発売されているものを使うか、デモ用の第1ステージのみのデータを製造元のFTPサーバからダウンロード（約39Mバイト）しよう。最初のステージでもかなりのテクニックを要求されるが、クリアできれば商品を買ってきても楽しめるだろう。こうしたアクションゲームの基本は、自分でパターンをコツコツと開発していくことにある。最初は難しく感じても、地道に攻略していくとクリアできるはずだ。なお、車酔いのする人はプレイ中に酔ってしまうかもしれない。

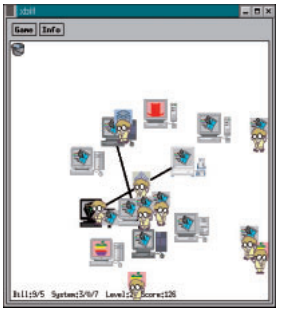
ネットワークを守れ!

xbill

バージョン：2.0　　種別：GPL

http://sakura.sfc.wide.ad.jp/pub/linux/JG/JG-beta7/other-sources/

X



シンプルなジョーク系のアクションゲーム。画面上には、さまざまなOSのコンピュータが表示されている。ゲームをスタートすると、「Wingdows」というOSを持った、どこかで見たようなキャラクタ「bill」がたくさん現れ、すべてのコンピュータに「Wingdows」を入れようとする。インストールされた「Wingdows」は、ネットワークを伝わってほかのコンピュータにも伝染するので、油断するとすべてが「Wingdows」化されてしまう。それを防ぐために、「bill」を退治するのがゲームの目的だ。操作は簡単で「bill」の上でマウスをクリックすればよい。仕事中にWindowsでトラブったら、xbillで気分をリフレッシュしよう。

Pythonで書かれたバリエーション豊かなカードゲーム
PySol

バージョン：3.00 **種別：GPL**

http://wildsau.idv.uni-linz.ac.at/mfx/pysol.html

X Python Tcl/Tk

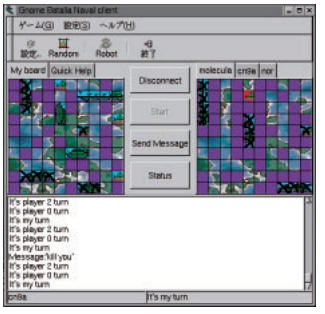
PythonとTcl/Tkを使って書かれたバリエーション豊かなカードゲーム。Windowsでおなじみの「クロンダイク」「フリーセル」「スパイダー」など、計111種類にも及ぶトランプの一人遊び（ソリティア）をプレイ可能だ。左ボタンのクリックでタロン（置札）から新しいカードを引き、ドラッグでタブロー（カード置き場）に移動するのが基本的な操作だ。また、なじみのないゲームでも、ルールの表示（英文）をはじめ、自動的にカードが動くデモ、動かせるカードが示されるヒントなどを参考にして、プレイしながらルールを覚えられるようになっている。このほか、カードの裏表のデザインやサイズを変更するカードセットが別に配布されている。

昔懐かしいネットワーク対戦ゲーム
Batalla Naval

バージョン：0.74.0 **種別：GPL**

http://www.pjn.gov.ar/~rquesada/batnav.html

X コンソール GNOME 日本語可 GTK+



サーバ／クライアント方式のネットワーク対戦型ゲーム。プレイヤーは互いに相手の見えない軍艦の位置を当てていく。そう、昔は潜水艦ゲームと言われていたゲームだ。最高7人まで同時にプレイでき、対戦相手としてAIも用意されている。

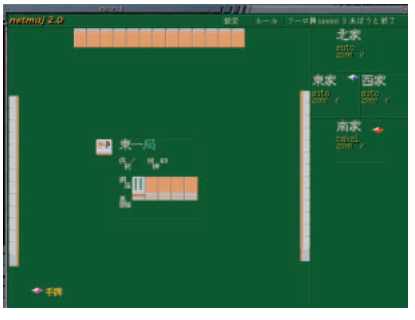
プレイするには、専用のサーバを起動させておく必要がある。GTK+1.2やGNOMEがあれば、グラフィカルな画面でプレイできるが、ncursesライブラリを用いたコンソールでのプレイも可能だ。また、Windows用のクライアントも存在する。サーバ、クライアントともに、バイナリのパッケージは提供されていないので、自分でコンパイルし、インストールする必要がある。

ネットワーク対戦麻雀
netmaj

バージョン：2.0.7 **種別：フリー**

http://www.sfc.wide.ad.jp/~kusune/netmaj/

日本語可



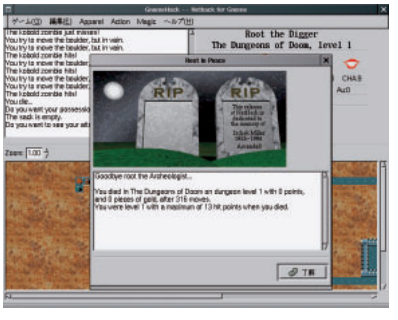
ネットワーク通信対戦麻雀ゲーム。国産のLinuxゲームを集めているグループとしてJGがあるが、netmajはその中でも秀逸なゲームだ。ないた時にもタンヤオがつく「なきタン」や、つもったときにピンフがつく「ピンヅモ」などのローカルルールをいくつもサポートしている。麻雀サーバを立ち上げてネットワーク対戦もできるし、一人でコンピュータ相手に遊ぶことも可能だ。最近はバージョンアップされていないようだが、安定していて楽しめる。通信対戦を含めた麻雀の機能とGUIとは、別のプログラムで実現されているので、Xを使わなくても日本語の表示ができるコンソールならプレイすることができる。コンピュータは強くないので、初心者の練習などにも適している。

グラフィック版NetHack
GnomeHack

バージョン：1.0.4 **種別：GPL**

http://www.xmission.com/~andersen/erik/gnomehack/gnomehack.html

X GTK+ GNOME



キャラクタベースのロールプレイングゲームとして人気のあるゲームNethackのGNOMEバージョン。GNOMEアプリケーションとなり、アイテム表示などがGUI化されている。ゲームオーバー時の墓碑も当然、グラフィックで表示される。操作は矢印のついたカーソルキーで行う。デフォルトではh、j、k、lが別のコマンドに割り当てられてしまっているので、昔からのNethackファンには納得がいかない点があるかもしれないが、コンソールから起動すれば文字だけのNethackがプレイできる。

正式リリース後も少しずつバージョンを上げている。最新版ではどういうわけかLaTeXがないとコンパイルできないようになっている。

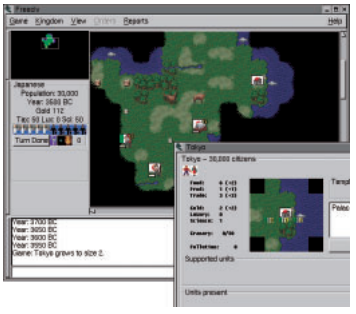
エンターテイメント

戦略シミュレーションゲームCivilizationIIのクローン
Freeciv

バージョン：1.9.0　　**種別：GPL**

http://www.freeciv.org/

X　GTK+



有名なマルチプレイヤー戦略シミュレーションゲームCivilizationIIのクローン。プレイヤーは、文明を発展させながら、他民族との外交や抗争を行い、世界制覇を目指す。最大14人が同時にプレイでき、対戦相手としてAIも用意されている。まずcivserverを起動しておき、プレイヤーはcivclientで接続する。そして、civserverにstart命令を与えれば、ゲームが始まる。

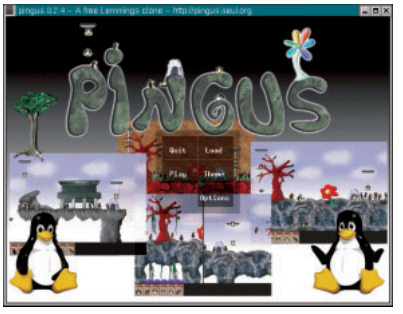
バイナリのパッケージも用意されているが、ディストリビューションによっては古いバージョンのものもある。LASER5 Linuxでソースからコンパイルをしてみたが、特に問題は起きなかった。クライアントはLinuxだけでなく、WindowsやAmiga用もある。

アクションパズルゲーム、レミングスのクローン
Pingus

バージョン：0.2.4　　**種別：GPL**

http://pingus.seul.org/

X



昔懐かしいゲーム、レミングスのクローン。ただし、今回の主人公はペンギンだ。次々に現われるペンギンに縦穴横穴を掘らせ、階段を作らせ、自爆させ、制限時間内に一定数のペンギンを出口から脱出させる。かわいいペンギンだが、かわいそうだなんて思っていると、全員助からないことにもなりかねない。みんなが助かるためには、何人かを犠牲にするのもやむを得ない、という社会の縮図……なんていうことを考える余裕もなくはまってしまうゲームだ。

プレイには、ClanLib、Hermesというライブラリが必要になる。また、起動後に出力先（X／svgalib／フレームバッファ）を選ぶので、＆を付けずに起動しなければならない。

X上の定番MP3プレーヤ
xmms

バージョン：0.9.5.1　　**種別：GPL**

http://www.xmms.org/

GTK+　日本語可

GTK+を使用した、大変人気の高いMP3プレーヤ。最新バージョンでは、Window Maker用のwmxmmsや、GNOMEパネルから起動できるようになるGNOMEアプレットgnomexmmsも付属している。また、ブラウザなしでWebサイト上にあるMP3ファイルを演奏することもでき、プラグインも充実している。

ビジュアル面については、スキンという、画像を変更して雰囲気を一新する機能がある（LinuxユーザーにはEnlightenmentやGNOMEの「テーマ」のようなものだと言ったほうが分かりやすいかもしれない）。xmms用に作られたスキンは、http://www.xmms.org/skins.htmlから30種類ほどダウンロードできる。またxmmsでは、Windowsで人気の高いMP3プレーヤWinAmpのスキンも、展開したり編集したりすることなくそのまま使うことができる。http://www.winamp.com/には3000以上のスキンが登録されているので使ってみるといいだろう。ダウンロードしたスキンは、~/.xmms/Skins/（または/usr/local/share/xmms/Skins/や/usr/share/xmms/Skins/）に置くと自動で認識してくれる。あとは起動して[Options]－[Skin Browser]（またはALT＋S）でスキンを選べるようになっている。チェックボタンをオンにすると、ランダムにスキンを変更することもできる。

LinuxのX環境が向上して、画面を高解像度で使うユーザーが多くなった今では小さすぎるような操作ウィンドウであるが、そのコンパクトさがxmmsの良いところでもある。

画面1 GNOMEアプレットからxmmsが起動できる。

画面2 スキンによって違った雰囲気を演出できる。

高機能なソフトウェアシンセサイザ
TiMidity++

バージョン：2.8.1　　種別：GPL

http://www.goice.co.jp/member/mo/timidity/index-jp.html

X　コンソール　日本語可

MIDI音源を持たないマシンでも、MIDIデータの再生を可能にするソフトウェアシンセサイザ。あらかじめ用意した音源データとMIDIデータの情報から実際の音の波形を生成し、リアルタイムに音楽を再生してくれる。生成した音をWAVEデータとしてファイルに保存することも可能だ。MIDIデータの形式は、標準MIDI形式のほか、レコンポーザ形式などにも対応している。

なお、音源データとして用いるGUS互換のパッチファイルは同梱されていないので、TiMidity++のリンクページで紹介されている配布サイトから適当なパッチファイルを取得する必要がある。たとえば、作者の出雲正尚さんのページには33Mバイトの高品質なパッチファイルが用意されている。これでは大きすぎる場合は、松本庄司さんのページ（http://www.i.h.kyoto-u.ac.jp/~shom/timidity/）にある10Mバイトや4Mバイトのパッチファイルを利用するといい。

TiMidity++を使ってMIDIデータを再生するには、

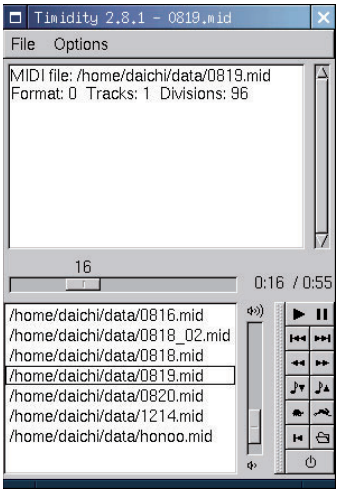
```
$ timidity ファイル名
```

とすればいい。ファイル名は複数指定可能だ。また、これらのファイルをtar＋gzipした状態のまま再生することもできる。

```
$ timidity hoge.tar.gz#hoge.mid
```

のように、「アーカイブファイル名#ファイル名」という書式で指定する。#以下を省略した場合は、アーカイブ内の全ファイルを再生する。HTTPプロトコルなどを利用して、ネットワーク上のファイルを再生する機能も用意されている。

このようにコンソールベースで使うだけでなく、Tcl/TkやGTK+などを利用した各種インターフェイスも用意されている。これらを利用するには、-iオプションに続けてインターフェイスの種類を指定する。たとえば-ikでTcl/Tkインターフェイス、-igでGTK+インターフェイスが使われる。



画面1
GTK+インターフェイスによる表示（-igオプション）。

ブラウザとビューアの顔を持つ画像管理ソフト
GtkSee

バージョン：0.5.0　　種別：GPL

http://hotaru.clinuxworld.com/gtksee/

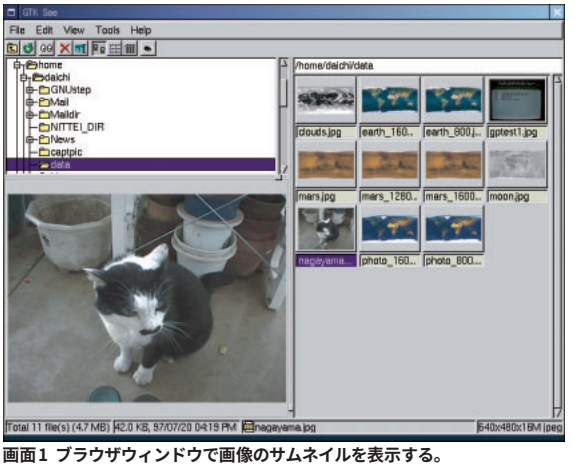
X　GTK+　日本語可

GTK+を利用した画像ビューア。GtkSeeという名は、Windows用の高速画像ビューア「ACDSee」にインターフェイスがよく似ていることに由来する。ファイル管理を行うイメージブラウザと、各画像を表示するイメージビューアの2つの顔を持ち、簡単に両者を切り替えて使える。また、キーボードショートカットが豊富に用意され、ほとんどの操作をキーボードから行える。

日本語表示に対応するには、ビルドする前にgtksee.cの「gtk_init(&argc, &argv);」の前に「gtk_set_locale();」という行を追加すればいい。また、単独でBMP／XBM／GIF／ICO形式に対応しているほか、libjpeg／libtiff／libpngといったライブラリを利用してJPEG／TIFF／PNG形式にも対応可能だ。これらのライブラリをあらかじめをインストールした状態でビルドするといい。

起動すると、イメージブラウザのウィンドウが開いて、カレントディレクトリのファイル一覧が表示される。左上がディレクトリツリー、左下が画像のプレビュー、右側にファイル一覧という構成だ。プレビュー画像は、画像ファイル名をクリックで選択すると表示される。ファイル一覧は、サムネイル（縮小画像）、小アイコン、詳細の3種類の表示方法を、ツールバーのボタンやF8～F10キーで選択できる。詳細表示の場合、画像ファイルは画像形式ごとに異なる背景色が使われ、画像サイズ・色数・形式といった情報もあわせて表示される。一方、サムネイル表示では画像ファイルだけが表示対象となり、それぞれの縮小画像が一覧表示される。

ツールバーの[View]ボタン（あるいはEnterキー）を押すと、ウィンドウの表示がイメージビューアに切り替わり、プレビューしていた画像が実際のサイズで表示される。なお、ディレクトリ内の画像を順番に眺める場合は、いちいちブラウザウィンドウに戻る必要はなく、ツールバーの[←][→]ボタン（あるいはPage Up／Downキー、スペースキー）を押すだけいい。このほか、画像以外は何も画面に表示されない「フルスクリーンモード」や、一定時間（初期設定は4秒）ごとに次の画像に自動的に切り替わる「スライドショー」といった機能も用意されている。



画面1 ブラウザウィンドウで画像のサムネイルを表示する。

さまざまなデジカメの画像を扱える画像ソフト

gPhoto

バージョン：0.4.1　　種別：GPL

http://www.gphoto.org/

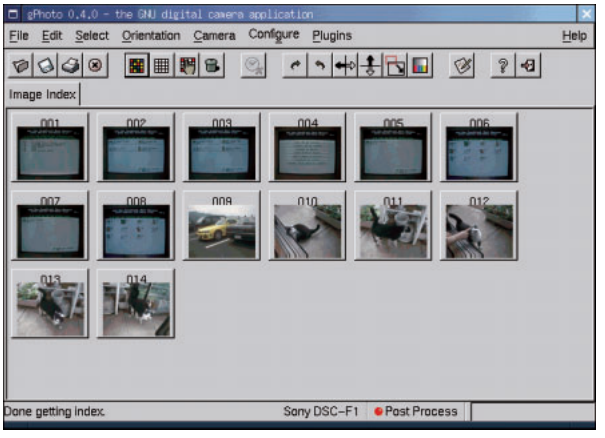
X　GTK+　日本語可

主にデジタルカメラの画像を扱うことを目的に作られた画像ソフト。対応しているデジカメは、現時点でなんと90機種近くにも及ぶ。パソコンと専用ケーブルで接続されたデジカメに対し、ハードディスク上のディレクトリを扱う感覚で、サムネイル（縮小画像）や画像の取得、削除などを行える。なお、日本語表示に対応するには、ビルドする前に、src/main.cの「gtk_init(&argc,&argv);」の前に「gtk_set_locale();」という行を追加する必要がある。

初めて起動したときには、デジカメの機種と使用するシリアルポートを選択して設定を保存する必要がある。設定によっては、ディレクトリのサムネイル一覧から選択した画像を表示するという、一般的な画像管理ソフトとして使うことも可能だ。続いて、パソコンとデジカメをケーブルで接続し、デジカメの電源を入れた状態で[Camera]－[Camera Summary]を選択する。カメラの状態が表示されれば正常に接続されている。「Error Opening Camera」と表示される場合は、再度設定を確認してみよう。

デジカメで撮影した画像をgPhotoで取得するには、最初にサムネイル（縮小画像）の一覧を取得してインデックスとして使用し、取得したい画像を選択するという手順を踏む。サムネイル一覧は、ツールバーの[Get Thumbnail Index]ボタン（またはCtrl－Iキー）で取得できる。しばらく待っていると、撮影画像のサムネイルがインデックス内部に表示されるはずだ。なお、取得したい画像のインデックスがあらかじめわかっている場合は、サムネイルなしのインデックスを使って素早く作業を行うこともできる。

これらのサムネイルを見ながら（あるいはインデックス番号を頼りに）、フルサイズ画像を取得したいものを選択する（複数可）。続いて、ツールバーの[Get Selected Images]ボタンを押すと、選択したインデックスのフルサイズ画像がデジカメから取得され、独立したページに表示される。ツールバーのボタンを使って、左右・上下反転や90度回転、サイズ変更、カラーバランス調整などの画像処理が可能だ。JPEG／GIF／TIFF／PNG／BMPなどの画像形式でファイルに保存したり、プリンタで印刷することもできる。



画面1 デジカメのサムネイル一覧が表示される。

パワフルな画像処理ソフト

Gimp

バージョン：1.1.12　　種別：GPL

http://www.gimp.org/

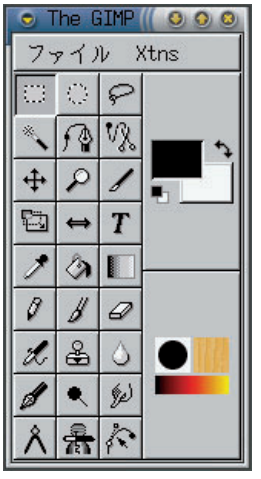
GTK+　日本語可

知名度の高い、高機能なペイントツールである。1.1系列ではさまざまな機能が追加され、Win32用のパッチもマージされた。前回の特集から比べると、マスコットキャラクターのWilber君も日本語をしゃべるようになり、ツールパレットも多少変わっている（起動時の画像はすごいことになってしまっているが）。

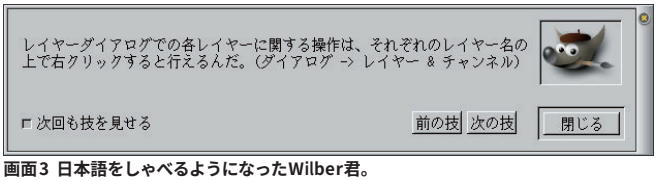
Gimpの主な特徴としては、プラグインがある。特定の画像処理を行うためのフィルタを追加したいなど、機能拡張したい場合にはGimpのソースコードをいじらなくてもプラグインを作ることで対処することができる。自分でプラグインを作れなくても、世界中から提供されるプラグインの中から目的に合ったものを使うことができる。

プラグインのほかに、楽しいロゴを簡単に作れるScript-Fuがある。Script-Fuはたくさんのパターンが用意され、うまくいかないこともあるものの、日本語もScript-Fuにかけることができる。Script-Fuはツールパレットの Xtnsメニューから選択で、Script-fuをさらにネットワーク越しに扱えるようにしたのがNet-Fuである。このNet-Fuを使ってブラウザからロゴの作成や画像処理をしてくれるサービスサイトとしてhttp://www.genroku.com/がある。開設者は日本人留学生とのことで、日本語によるコンテンツもあるそうだ。興味のある人は、一度訪れてみてはどうだろうか。

画面1 なぜかスタートはおどろおどろしい画面。



画面2 パレットのレイアウトも変更があった。



画面3 日本語をしゃべるようになったWilber君。

CDリッパなどと連動するCDプレーヤ
Grip／GCD

バージョン：2.9 **種別：GPL**

http://www.nostatic.org/grip/

X GTK+ 日本語可

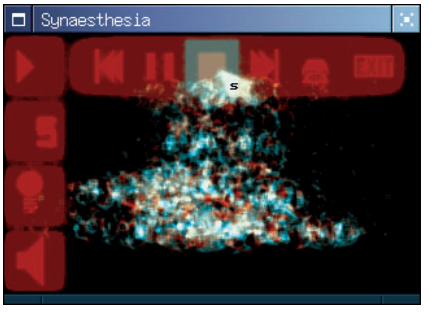
CDリッパやMP3エンコーダと組み合わせて、MP3作成のフロントエンドとして使えるCDプレーヤソフト。CDリッパなどのフロントエンド部分を持たないGCDも用意されている。いずれもCDDBに対応しており、インターネット上のCDDBサーバから音楽CDのタイトルや曲目情報を取得可能だ。画面表示の各部分は必要に応じて表示をオン／オフできる。MP3作成時には、コンソールベースのMP3エンコーダ（「午後のこ〜だ」など）とID3タグツール（mp3info）が別途必要だ。なお、曲目などに日本語を利用するには、ソースを書き換える必要がある。grip.cの「gtk_init(&argc,&argv);」の前に、「gtk_set_locale();」を1行追加してビルドすればいい。

演奏中のCDの音をグラフィカルに表示
Synaesthesia

バージョン：2.0 **種別：GPL**

http://yoyo.cc.monash.edu.au/~pfh/synaesthesia.html

X コンソール



再生している音をきらめくグラフィックで視覚化してくれる一風変わったCDプレーヤ。X上で動作するxsynaethesiaのほか、svgalibを利用したsynaethesiaなどが用意されている。グラフィックは、音の位置や周波数成分などに応じて変化する。バージョン2.0からは、従来のスターに加え、ウェーブやフレームなどエフェクトが追加され、細かなパラメータも調整可能になった。再生開始や停止、トラック移動、エフェクトやボリュームの変更などの操作はウィンドウ上のアイコンで行い、一定時間操作しないとアイコンは自動的に隠される。このほか、他のソフト（mpg123など）の出力をパイプ経由で再生したり、サウンドカードのライン入力をモニタすることも可能だ。

ネットワーク対応のCDプレーヤ
Kscd

バージョン：1.2.7 **種別：GPL**

http://www.kde.org/

X Qt

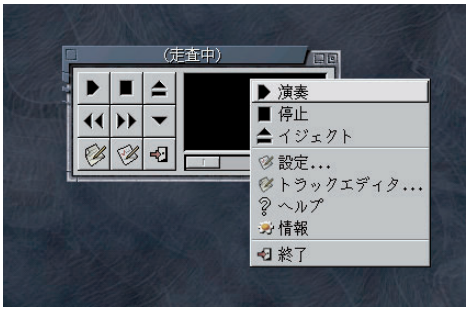
インターネットに接続された状況で本領を発揮するネットワーク対応のCDプレーヤ。その最大の特徴はリモートCDDBに対応していることだ。リモートCDDB機能により、再生中のCDのCDDBエントリ（アーティスト名や曲名など）をインターネット上にあるリモートCDDBサーバから取得できる。リモートCDDBサーバにエントリが見つからない場合には、自分でCDDBエントリを編集し、リモートCDDBサーバに登録することも可能だ。リモートCDDBサーバへはHTTPプロキシを経由して接続できるので、HTTPしか通さないファイアウォールの内部からでもリモートCDDB機能が利用できる。これらのアーティスト情報をWebサイトで検索することも可能だ。

GTK+ベースで、CDDBに対応したCDプレーヤ
gtcd

バージョン：1.0.9 **種別：GPL**

http://www.gnome.org/

GTK+ GNOME 日本語可



GNOME標準のCDプレーヤアプリケーション。マルチメディアアプリケーションというと、派手で凝った作りのものになりがちだ。しかしLinuxで音楽CDを聴く場面を考えると、何か作業をしながらということになるので、こうしたシンプルで作業の邪魔にならないものがよい。

gtcdは、インターネット上のCDDBというCDの曲名を集めたデータベースサイトに接続する機能を持っている。ローマ字表記ではあるが、日本のアーチストや曲名も表示できる。邦楽のCDを聞いている際に、曲名を海外のサーバから教えてもらっているのは、少し不思議な気分だ。なおCDDBのデータは、Webブラウザを用いて、http://www.cddb.com/から検索することもできる。

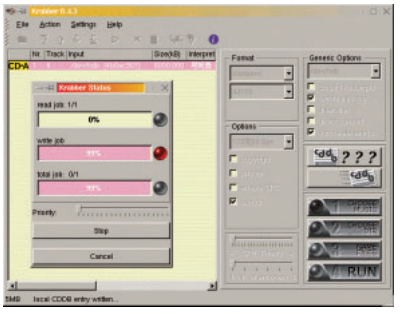
各種オーディオユーティリティのKDE用フロントエンド

Krabber

バージョン：0.4.3a（安定版）／0.4.4　種別：GPL

http://krabber.automatix.de

X　Qt



テキストベースのオーディオユーティリティのGUIフロントエンド。面倒なコマンドライン操作なしで、MP3ファイルを作成したりオーディオCDを作成できる。Krabberを使うには、cdparanoia（CDリッパ）、sox（データ形式の変換）、mpg123（mp3ファイルのデコード）、cdrecord（オーディオCDの作成）が別途必要で、さらにmp3のエンコードを行うには、8hz、bladeenc、ISO、lame、xing、l3enc、mp3encのいずれかが必要だ。使用方法は非常に簡単で、画面右下にある４つのボタン（曲の選択、作業用ディレクトリの指定、ファイル名の指定、実行）を順番に押していくだけで、MP3ファイルが簡単に作成できる。リモートCDDBにも対応している。

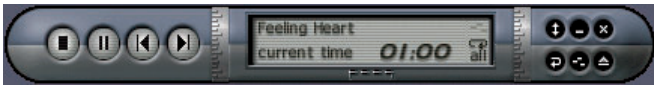
UIを切り替えられる安定したMP3プレーヤ

FreeAmp

バージョン：1.3.1/2.0beta4　種別：GPL

http://www.freeamp.org/

X　コンソール



シンプルなパネルが特徴のMP3プレーヤ。ユーザーインターフェイス部分はプラグインとして本体から分離されており、起動時に組み込まれる。画面で紹介しているX用UIのほか、キャラクタベースの独自UIやmpg123コンパチUIなどが用意されている。HTTPのユニキャストおよびRTPのマルチキャストによるストリーム再生を行ったり、受信データをファイルに保存することも可能だ。また、現在ベータテスト中の2.0では、WinAmpやXmmsなどでおなじみのスキンが導入され、プレーヤの外観を大幅に変更できるようになったほか、プレイリストなども含めWindows版との機能的な違いが解消された。ただし、動作自体はまだ不安定なところがあるようだ。

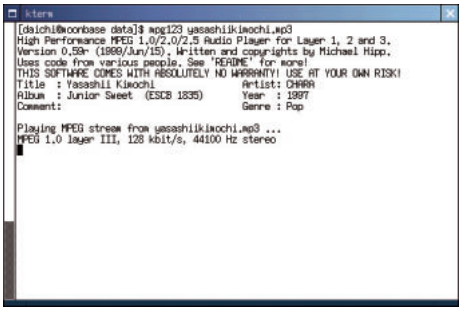
コマンドラインで動作するMP3プレーヤ

mpg123

バージョン：0.59r　種別：フリー

http://www.mpg123.de/

コンソール



コンソールベースで動作する軽くて高性能なMP3プレーヤ。MPEG1（および2）のAudio Layer1〜3に対応している。HTTP経由でのストリーム再生も可能だ。使い方は簡単で、ktermなどのコマンドラインで「mpg123 ファイル名」とするだけ。複数のファイル名を指定すると、それらを順番に再生する（中断はCtrl－Cキー）。なお、通常の再生ではファイル名や形式、ビットレート、サンプリング周波数などが表示される。バックグラウンドで再生する場合は、メッセージを抑制する-qオプションを使おう。X上で動作する「GQmpeg」（http://gqview.netpedia.net/mpeg-over.html）など、各種フロントエンドソフトが豊富に用意されているのも特徴だ。

超高速なMP3エンコーダ

午後のこ〜だ

バージョン：2.23　種別：GPL

http://www.kurims.kyoto-u.ac.jp/~shigeo/

コンソール　日本語可

```
[テストの例。一番下の'time'でベンチを競う]
% gogo -nopsy -test
GOGO-no-coda ver. 2.23 （Nov 24 1999）
Copyright （C） 1999 PEN@MarineCat and shigeo
        Special thanks to Keiichi SAKAI, URURI, Noisyu and Kei
 *** at bench in the early afternoon [benchmark mode] ***
MPEG 1, layer 3 j-stereo
inp sampling-freq=44.1kHz out sampling-freq=44.1kHz bitrate=128kbps
input  file `stdin'
output file `default.mp3'
{ 22943/ 22968} 99.9% （11.78x） re:[00:00:00.05] to:[00:00:50.93]
End of encoding
time= 50.930sec
```

国産の高速MP3エンコーダ。一時は開発中止になりそうだったが、元になっているエンコーダに対するパッチ（GPL）として公開することでライセンス問題にも決着をつけた。最新バージョンでは、今までの3D Now!に加え、MMX、Enhanced 3D Now!（AMD Athlon）、SSE（Pentium III）などへの最適化、そしてSMPへの対応など、フリーのツールながらかなり品質の高い開発が続けられている。Dual Pentium IIIでLinuxカーネルをSSE対応にすると、音楽CDを30倍速で読み出してその速度でエンコードすることができるそうである。

'gogo -test'の「午後べんち」では、ホームページに結果とマシンスペックを申請して競うことができる。

マルチメディアプレーヤxanimのフロントエンド

aKtion!

バージョン：0.3.6　　種別：GPL

http://www.geocities.com/SiliconValley/Haven/3864/aktion.html

X　Qt

Qtライブラリを使った、マルチメディアプレーヤ「xanim」（http://xanim.va.pubnix.com/home.html）のフロントエンド。xanim 2.70.7.0以降に対応している。数多くの画像、動画フォーマットに対応し、さまざまなオプションを指定可能なxanimが、GUIで操作できるようになる。

再生するファイルをGUIのダイアログから指定でき、ボリューム調整、カラーマッピングなどのxanimのさまざまなオプションも、GUIで設定できる。また、再生中に「c」キーを押すことにより、動画の各フレームのキャプチャもできるようになっている。Qtライブラリを使っているため、他のKDE用のソフトとの操作性の統一もとれている。

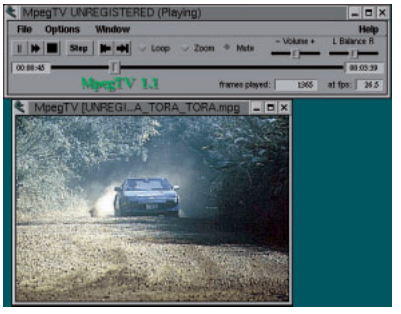
多機能なMPEGビデオプレーヤ

MpegTV Player

バージョン：1.0　　種別：シェアウェア

http://www.mpegtv.com/index.html

X



MPEG1形式の動画、音声またVIDEO CDを再生するプレーヤ。拡大やフルスクリーンモードでの再生も可能になっている。また、コントラスト、ガンマ値や、音声のクオリティなどのさまざまなオプションも設定可能。入力データは、ファイルだけでなく、パイプやネットワークURLによる指定もできる。

実行形式はglibc、libc5、LinuxPPC、Alphaといった幅広いプラットフォーム向けに、RPM形式、deb形式、tarボールといった形で用意されている。同梱のコマンドライン版のmtvpのみ、個人用、教育用の非営利使用に限りフリー。GUIのコントローラを利用するmtvはシェアウェア（10USドル）。

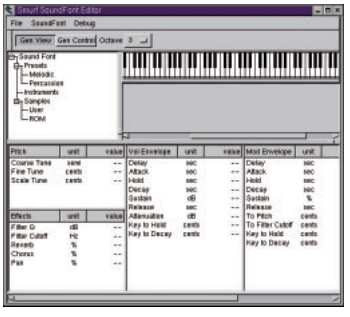
Sound Blaster用のSoundFontエディタ

Smurf Sound Font Editor

バージョン：0.45　　種別：GPL

http://www.resonance.org/smurf/

X　GTK+



Sound Blaster用SoundFontを自作および加工するためのエディタ。Creative LabのSound Blaster AWE32／64／Liveシリーズは、MIDIデータを演奏できるが、それらの音色は、SoundFontと呼ばれるウェーブ音源のテーブルファイルにより定義されている。このファイルを自分で作成し読み込ませることにより、Sound Blasterを独自の音色を持ったMIDI機器にすることができるわけだ。最近はWindows用のゲームなどでもSoundFontを利用しているものがある。

サウンド用のドライバがインストールされていれば、画面上の鍵盤をマウスでクリックして実際に音を聞きながら、さまざまなパラメータを調整して、独自のSoundFontを作ることができる。

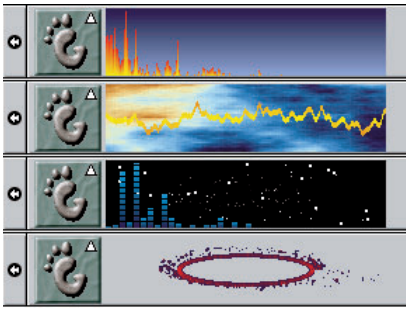
GNOME用のサウンドアナライザアプレット

Visual Sound Analyzer

バージョン：0.9　　種別：GPL

http://vsa.linuxcore.com/

X　GNOME



サウンドをビジュアル化してGNOMEのパネルに表示するアプレット。実行にはGNOME、サウンドデーモンEsounD、高速フーリエ変換ライブラリFFTW（http://www.fftw.org/）が必要だ。EsounDを利用して再生中のサウンドを高速フーリエ変換（FFT）し、スペクトラムアナライザ風のグラフィックをリアルタイムに表示する。背景やグラフィックの表示、フィルタなどがプラグイン形式になっており、好みに合わせて自由に組み合わせられるのが特徴だ。なお、初期設定ではディケイ（減衰）が有効になっているのだが、この機能は正常に動作していないようなので、プロパティダイアログの[General]ページの[Enable Spectrum decay]のチェックを外しておこう。

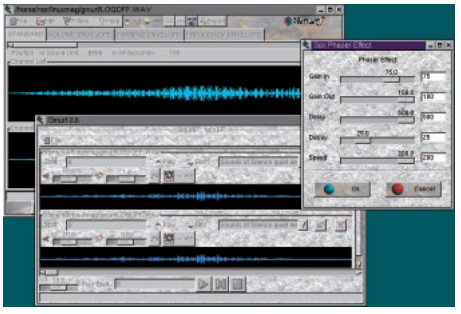
WAVとMP3に対応したウェーブエディタ

Gmurf

バージョン：0.5　　種別：GPL

http://www.epita.fr/~sam-lo_p/

X　GTK+



WindowsのWAVと、MP3フォーマットに対応したウェーブエディタ。ミキサー機能も付いており、ラインやマイクからの入力やボリューム、バランスの調整もできる。　ウェーブエディタとしては、波形のズームイン、アウト、波形を見ながらの削除、コピー、ペーストはもちろん、プラグインを利用することにより、エコーやリバース、周波数の変更等のさまざまなエフェクトをかけることができる。作者のWebページにはプラグインの作成方法も解説されているので、自分の欲しいエフェクトを作ることも可能だ。サウンドデバイスがEsounDのバージョン0.2.12の場合は、うまく動作しないので、0.2.8の利用を推奨している。

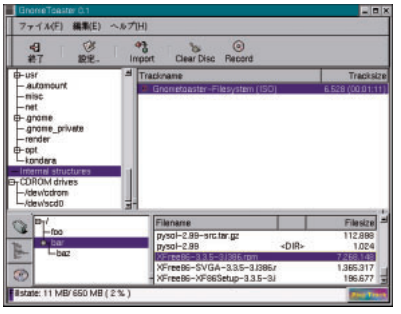
CD-R作成用統合ソフト

Gnome Toaster

バージョン：0.1-11-30-1999　　種別：GPL

http://gnometoaster.rulez.org/

X　GTK+



CD-R作成用のプログラム群を統合して操作するためのソフト。mkisofs／mkhybridによるCD-ROMイメージファイルの作成、cdrecordによる焼き込み、cdda2wavによる音楽CDのwavファイル化、mpg123によるMP3ファイルの作成などが同一のインターフェイスから行える。最近の高性能なマシンならば、イメージファイルを作らない「オンザフライ」でのCD-R焼きも可能だ。またGnome Toaster独自の機能として、MIDI、wavファイル、MP3などの各種データを音楽CD用のデータに変換する機能を持つ。変換されたデータは、音楽CDとして焼くだけでなく、Gnome Toaster上から聞くこともできる。現状ではOpen Sound System、Enlightenment sound daemonに対応している。

CD-R焼きをまとめてお世話する

gcombust

バージョン：0.1.25　　種別：GPL

http://www.abo.fi/~jmunsin/gcombust/

X　GTK+

Linux環境でCD-Rを作製するには、mkisofsやmkhybridでイメージファイルを作製し、cdrecordでメディアに焼くという手順が一般的だ。gcombustは、これらのプログラムや、音楽CDのデータをwavファイルに変換するcdda2wav、CD-Rのファイルと元のファイルを比較するdiffコマンドなどをまとめて扱えるGUIフロントエンドだ。localeを正しく認識し、日本語によるメニュー表示が行える。また設定項目の上にマウスカーソルを置くと簡潔な説明が表示されるため、マニュアルなしでも十分に使いこなせる。マシンの性能が高ければ、mkisofsの出力とcdrecordの入力をパイプでつなぎ、「オンザフライ」でCD-ROMを焼くことが可能だ。

画像処理用ツール集

ImageMagick

バージョン：4.2.9　　種別：フリー

http://www.wizards.dupont.com/cristy/ImageMagick.html

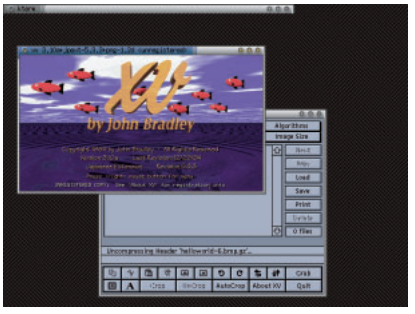
X

非常に多くのフォーマットを扱えるイメージツール。displayコマンドでは画像を表示し、簡単な編集もできる。convertコマンドでは画像のフォーマットを変換することができる。よく知られた画像ライブラリにImlibというのがあるが、あまりメジャーでないフォーマットの解析にはこのImageMagickのコマンドを使用している。多くなりすぎた画像フォーマットを統一しようとして、またもうひとつ画像フォーマットが増えるという現象は昔からよくあるので、ImageMagickのように強力な変換ツールは欠かせない。インターフェイスは前世代的だが、メニューの機能は充実している。現在も開発は続けられており、最近もまたいくつかバージョンアップしている。

定番の画像ビューア
xv
バージョン：3.10a　　種別：シェアウェア

http://www.trilon.com/xv/xv.html

X



昔からある定番グラフィックビューア。シェアウェアであるが、あらゆるディストリビューションには評価版が常識のようについており、twm以外のウィンドウマネージャならデフォルトのメニューから起動できるようになっているのが普通である。何といっても高機能なのに動作が軽く、コンパイルしてインストールするのにも時間がかからない。最近はバージョンアップされていないが、その分安定しており、安心して使える。

日本固有のMAGフォーマット対応など、機能拡張のためのパッチも出回っているので、今後もグラフィックビューアとしてのメジャーな地位をキープしていくだろう。

GNOMEのクールな画像ビューア
Electric Eyes
バージョン：0.3.11　　種別：GPL

http://www.labs.redhat.com/ee.shtml

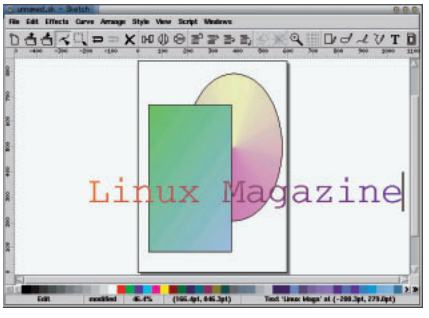
X　GTK+　GNOME　日本語可

グラフィックライブラリImlibを使用した、GPLな多機能グラフィックビューア。スクリーンショットはウィンドウの階層を判断してくれるので便利だ。GNOMEプロジェクトでは、最近グラフィック用のライブラリをImlibからgdk-pixbufに変更し、グラフィックビューアもeog（Eye of Gnome）というものを新しく作ってリリースした。したがってElectric Eyesは次第にGNOME標準ではなくなっていくが、eogはまだ扱えるフォーマットも少なく、画像の拡大・縮小くらいしかできないので、当分はElectricEyesのほうが機能が充実しているだろう。メッセージの翻訳追加やバグフィックスなど、今後もメンテナンスは続けられていくようである。

ベクターグラフィック用ドローソフト
sketch
バージョン：0.6.2（安定バージョン）　　種別：GPL

http://www.online.de/home/sketch/

X　Tcl/Tk　Python



Adobe IllustratorやGYVEのような、ベクターグラフィックを描くためのドローツール。ベクター系のツールは、Gimpなどのビットマップ系のツールで拡大／縮小、回転などの操作を繰り返した際に生じる画像の乱れが起きないため、高品質を求められる用途に向いている。

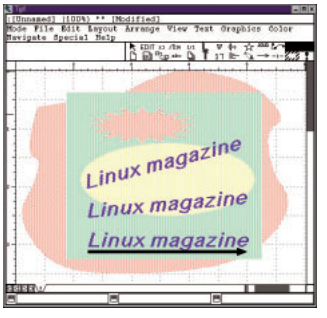
安定バージョンのsketchは日本語に対応していないので、年賀状の出力などには使えないが、できあがったイメージはPostScript形式に対応したプリンタで印刷することができる。

開発バージョンとしてリリースされている0.7.xでは、ツールキットがTkからGTK+への移行が行われるので、そちらで日本語化が一気に加速することが望まれる。

定番のドローツール
Tgif
バージョン：4.1pl25　　種別：フリー

http://bourbon.cs.umd.edu:8001/tgif/

X　日本語可



昔からX Window System上で使われている、Xfigと相壁をなすドローツール。基本のファイル形式は独自のものだが、Postscript、EPS、Xbitmap、GIFといった形式での入出力が可能となっている。

実行に必要なファイルは1つで、RPMやdebになっているわけではない。必要なライブラリ（glibc2）さえあれば、特にインストール作業をしなくても使うことができる。インターフェイス自体は昔から変わっていないので、最近のGTK＋やQt、Tcl/Tkなどを使ったものに比べると見劣りはするが、逆に、個々のライブラリに依存する部分が少ないため、どのような環境でも安心して動作させることができると言えるだろう。

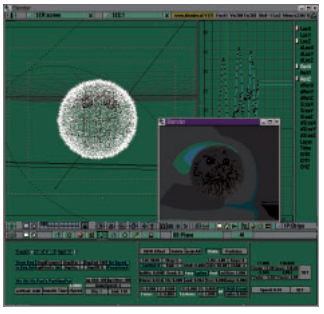
高機能3Dモデリング／アニメーションツール

Blender

バージョン：1.71　　種別：フリーソフト

http://www.blender.nl/
http://www.dims.or.jp/blender/stuff/bl_manjp.html

X　Mesa



Not a Number製の高機能な3Dモデリング／アニメーションツールのBasic版。Basic版とはいえ、市販のモデリングツールに遜色のない、個人で使うには十分な機能がある。3Dモデリングに興味があるのならば、まず、このフリーのツールを使っていろいろ試してみるのもよいだろう。もちろん、VRML、DXFといった汎用的な形式での出力をサポートしているので、Blenderで作ったデータを、他のツールで活用することも可能だ。

英語マニュアルだけでは不安なら、国内のBlenderユーザー有志によって日本語化されたオンラインマニュアル（バージョン1.5用）を参考にさせてもらおう。

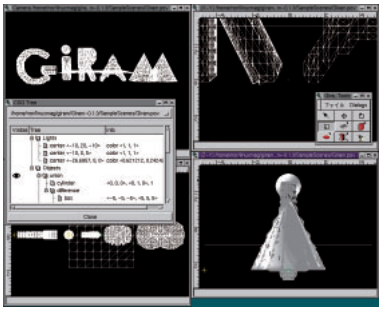
入門者向け3Dモデリングソフト

Giram

バージョン：0.1.3　　種別：GPL

http://www.minet.net/giram/

X　GTK+



3Dレンダリングソフト POV-Rayを利用した、3Dモデリングソフト。高機能とはいえないが、そのぶん一般のモデリングソフトにありがちな難解さはなく、3Dモデリング入門者にもわかりやすいインターフェイスになっている。POV-Rayのフロントエンドを呼び出し、レンダリングを行うことができる。実行形式は用意されていないので、自分でコンパイルする必要がある。

まだ未完成な部分が多いとドキュメントにもあり、実際にメニューを選ぶと「機能はまだ実装されていません」というメッセージが出ることもあった。ただし、開発は継続されており、将来が楽しみなソフトである。

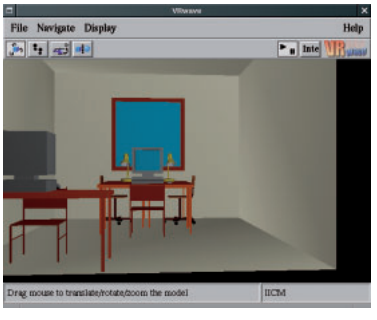
VRML表示ツール

VRwave

バージョン：0.9　　種別：フリーソフト

http://www.iicm.edu/vrwave

X　GTK+　Tcl/Tk



VRML2.0を理解するVRMLブラウザ。 Netscape用プラグインもある。付属のサンプルの中で派手で楽しいものとしてはoffice、teapotくらいだが、それ以外のサンプルもオフィシャルサイトからダウンロード可能だ。最近ではこうした3D関係の分野においても、Linuxは重要になってきている。PCとLinuxを使うことで、安価で高速で、信頼性の高いクラスタリング環境を得ることができる。そのため、特殊効果を使った映画を作る際に、好んで使われることがあるようだ。

VRwaveが動作する環境を作るのは、結構大変だ。今回、動作を確認した組み合わせは、Vine-1.1、JDK-1.0.2（staticリンク版）、VRWave 0.9（JDK1.0.2用）バイナリである。

3Dモデルをさまざまな角度から眺められるビューア

Morpheus

バージョン：0.2　　種別：GPL

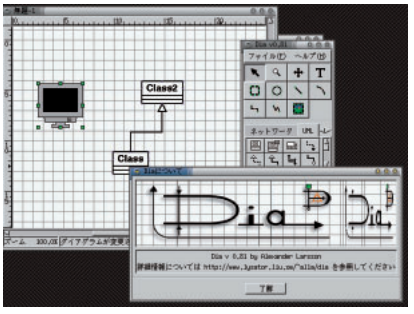
http://wine.sexcity.pl/morpheus/

X　Mesa　GTK+　GNOME

3Dモデルを表示し、さまざまな角度から眺められるGNOME用の3Dモデルビューア。Morpheus本体とライブラリのlibmorphで構成される。このほかツールキットのGtkGLArea（http://www.student.oulu.fi/~jlof/gtkglarea/）も必要だ。現時点で対応している3Dモデルは、LightWaveのデータ形式（拡張子lwob）と3D Studioのデータ形式（拡張子3ds）の2種類だ。フリーな3Dモデルのデータは、3Dカフェ（http://www.3dcafe.com/asp/meshes.asp）などから入手できる。こうしたデータを読み込むと3Dモデルがウィンドウに表示される。表示されたモデルは、左ボタンで上下にドラッグすると拡大／縮小、中ボタンでドラッグすると回転させられる。

Visioのようなダイアログエディタ
dia
バージョン：0.81 **種別：GPL**
http://www.lysator.liu.se/~alla/dia/dia.html

GTK+ 日本語可

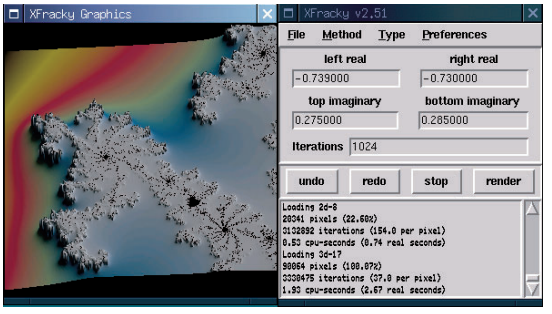


diaはダイアグラムを編集するための専用アプリケーションである。コンピュータ業界では、ネットワークの構成やソフトウェアの構造を人に伝えたり、ドキュメントとして残したりする際に、こうしたツールは大変有用である。Windowsの世界では、同じような目的のツールとしてVisioがあるが、最近Microsoftに巨額で買収された。

日本語環境下での文字化けは、時間とスキルのあるユーザーが直せるので問題ではないが、専門用語に詳しい翻訳者が見つからず、日本語訳のできがもの足りないのが残念だ。diaはGNOMEプロジェクトが推奨するオフィスツール「GNOMEワークショップ」のひとつとして位置づけられている。

さまざまなフラクタル図形を高速に表示
XFrackey
バージョン：2.51 **種別：フリー**
http://www.gk.dtu.dk/~hwj/xfracky/index.html

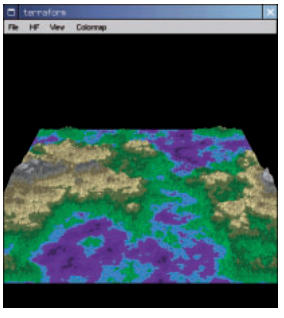
X



マンデルブロート集合やジュリア集合などを利用したさまざまなフラクタル図形を瞬時に表示するソフト。よく見かける2D平面図だけでなく、3D俯瞰図表示も可能だ。メインウィンドウで表示範囲を数値で設定して[render]ボタンを押すか、グラフィックスウィンドウ上で直接範囲を指定すると、その範囲のフラクタル図形が描画される。描画方法は[Method]メニュー、フラクタルの種類は[Type]メニューで変更可能だ。また、デモモードにすると、5秒（変更可）ごとにいろいろなフラクタル図形を順番に表示してくれる。気に入った図形のパラメータをファイルに保存したり、図形をTGA形式の画像ファイルとして書き出す機能も用意されている。

フラクタル地形を作成して3D表示
Terraform
バージョン：0.4.7 **種別：GPL**
http://212.187.12.197/RNG/terraform/

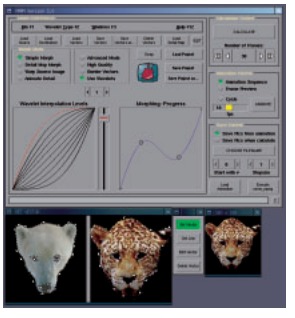
X GTK+



フラクタル3D地形図を生成するソフト。実行にはGTK+のC++ラッパであるGTK--のライブラリが必要だ。最初にパラメータ（ランダム設定可）を設定すると、フラクタル地形が自動生成されて3Dワイヤーフレーム表示される。数値地図（DEM）や各種画像ファイルを読み込んで地形図に変換することも可能だ。表示は3Dワイヤーフレームのほか、2D平面図や3D等高線図などに切り替えられ、複数のカラーマップが用意されている。侵食（Erode）などの効果による地形の変形も可能だ。気に入った地形ができたら、平面図の形で画像ファイルなどに保存できる。なお、レイトレーシング機能は持っていないため、別途POV-Rayなどを利用する必要がある。

形と模様の変化を個別に制御できるモーフィングソフト
xmrm
バージョン：2.0 **種別：GPL**
http://www.cg.tuwien.ac.at/~xmrm/

X



2つの画像から指定された数の中間画像を自動的に生成してアニメーションを作成するモーフィングソフト。最初に2つの画像の対応する部分をベクトルで設定する。次に、適当なフレーム数を設定して中間画像を生成すると、アニメーションを表示できるようになる。アニメーションの速度を変更したり、中間画像をコマ送り表示することも可能だ。モーフィングに関しては、模様などの細部の特徴と、形などの大まかな特徴の変化の度合いを、それぞれグラフ形式で個別に設定できるのが特徴。このほか、模様の変化速度を部分的に変える「Detail Map Morph」や、1つの画像だけを利用して形を変化させる「Warp Source Image」など、多彩な機能が用意されている。

これで解決！
Linux 100の疑問

あんな疑問もこんな疑問もコレで解決。あなたの疑問にすべてお答えします

Linuxを使っていると、さまざまな疑問がわいてくる。あんなことはできないのだろうか？　なんでこうなっているんだろう？　こんなときは、いったいどうすればいいんだ？

ちょっとしたことなんだろうけど、気になって夜も眠れない。そんな、誰もが持つ疑問にお答えしよう。100を超えるQ&Aを読めば、あなたのスキルも大幅アップ。

photo : Takashi Shinohara(Dee)


A

基礎知識

Q Linuxってなに？

A 一言でいえば、LinuxとはUNIX類似OS（Operating System）です。1991年、フィンランド大学の学生だったLinus Torvaldsさんは、既存のUNIXのコードは使わず、i386 CPUを搭載したIBM PC/AT互換機で動作するこのOSを作りました。その後、Linusさんを中心として世界中の開発者が移植・改良を重ね、現在も開発が続けられています。

現在配付されているLinuxには次のような特徴があります。

- **i386以上のx86 CPUを搭載したPC、Macintosh、SPARC、MIPS、Alphaマシンなどで動作する。**
- **真のマルチタスク、仮想メモリ、共有ライブラリ、SMP対応、TCP/IPによるネットワーク機能などを実現。**
- **GCC、Emacs、X Window System、数多くのUNIX用プログラムを利用可能。**
- **Linuxカーネルは GPL（GNU General Public License Version 2）に従って配布されており、誰でもソースコードを入手できる。**

画面1　LinusさんはLinuxをどう発音するか聞いてみよう

Q Linuxってなんて読めばいいの

A 文書などで公式に発表されている答えはありません。日本では、「リナックス」、「リヌクス」、「ライナックス」などと発音する人が多いようです。

なお、開発者のLinusさん自身がスウェーデン語で発音したオーディオクリップが、http://www.linux.org/info/sounds/swedish.auに、英語のものがhttp://www.linux.org/info/sounds/english.au（**画面1**）にありますので、興味ある方はどうぞ。

Q Linuxって著作権フリーなの？　複製して、人にあげてもいいの？

A Linuxカーネルの著作権は、Linus Torvaldsさんが保持しています。また、カーネルの一部は、そのコードを書いた開発者が著作権を持ちます。したがって、PDS（Public Domain Software、著作権を放棄したソフトウェア）ではありません。

カーネルの配布はGPLに基づいて行われています。GPLとは、FSF（Free Software Foundation）によって策定されたライセンスです。GPLでは、複製・変更・配布をすることが許されています。ただし、再配布を制限してはいけませんし、ソースコードが入手できることを保証しなければなりません。

正確な内容を知るためには、原文（http://www.gnu.org/copyleft/gpl.html）や、カーネルのソースに含まれるCOPYINGというファイルを参照してください。公式な文書は英語で書かれた原文のみということになりますが、英語が苦手な方には、日本語訳（ftp://ftp.sra.co.jp/pub/gnu/local-fix/GPL2-j/gpl.text.gz）も参考になるでしょう。

カーネル以外のプログラムの配布条件は、それぞれのプログラムのライセンスに従います。ディストリビューションに含まれるプログラムの多くはGPLに従っており、これらは再配布しても問題ありませんが、商品版ディストリビューションに含まれる商用フォントや商用ソフトウェアは複製できませんので気をつけましょう。

Q Windows／MacintoshからLinuxに乗り換えるべきですか

A この質問に対する答えはありません。なぜならば、LinuxとWindowsは設計思想がまったく違うOSであり、それぞれに良い点と悪い点があるからです。GUI操作のみで、ある程度は直感的に使えるWindowsやMacintoshとは違い、コマンドによる操作が多いLinuxは、決して取っつきやすいものとはいえないでしょう。しかし、ユーザーがスキルを身につけていくにつれ、その分、できることがどんどん広がるのがLinuxのおもしろいところです。

乗り換えるのではなく、まずは、両方を使ってみてはいかがでしょう。そ

うすれば、それぞれの得手不得手もわかります。そして、やりたいことにあったOSを使い分ければよいのです。ひとつのOSにこだわらず、適材適所で使い分けるのが賢い方法です。

1台のマシンに両方をインストールして、デュアルブートで使うのもよいのですが、OSを切り替えるのがちょっと面倒です。複数のマシンがあるなら、イーサネットなどでつないで、WindowsマシンやMacintoshからLinuxマシンを利用することもできます。ネットワークの設定をしなければなりませんが、充実したネットワーク機能はLinuxの魅力のひとつですので、インストールに慣れたらぜひ挑戦してみてください。

Q なぜペンギンがマスコットなの?

A 1996年の初頭、linux-kernelメーリングリストで、Linuxにはどのようなロゴやマスコットが合うかが議論されていました。ほかのOSロゴのパロディがよいとか、サメやワシといった強くて気高いものがよいという意見が数多く出されましたが、Linusさんがペンギンが好きだと発言したことで、ペンギンに決まりました。

そして、ペンギンロゴコンテストが開かれ、Larry EwingさんがGimpで描いたTuxがロゴとして採用されたのです(**画面2**)。

この数年前、Linusさんは、オーストラリアの動物園で小さなペンギンにかまれ、“ペンギン中毒”に感染してしまったのです。これに感染すると、ペンギンのことばかりを考え、ペンギンラブラブ状態に陥るのだそうです。Linuxにマスコットが必要だということになったとき、ペンギン中毒のLinusさんの頭のなかにまず浮かんだのがペンギンだったというわけです。Tuxをロゴキャラクターに決めたのは、面白みを持ったキャラクタだからとのことで、これはLinuxにとっても不可欠の要素なのだそうです。

Tuxに関しては、Steve BakerさんのWebページ(http://www.woodsoup.org/~sbaker/tux/doc/)に詳しく説明されています。

Q スイッチ切りたいんだけど、どうすればいいの?

A 正しい方法で終了しないと、ディスクに未保存のデータが消えたり、二度と起動しなくなったりするかもしれません。電源を切る前に必ずシャットダウンが必要ですので、次のコマンドだけは覚えておきましょう。

```
# shutdown -h now
```

なお、オプションの-hを-rにすると、システムを終了させるのではなく、再起動させます。

Q コマンドの使い方がわかりません

A Linux(UNIX)で使われるコマンドには、ヘンテコな名前のものが数多くあり、しかもそれぞれのコマンドにはたくさんのオプションがあります。まるでユーザーに使われることを拒絶しているかのようですが、実際にそのすべてを覚えている人はいません。では、みんなどうやって使っているのでしょう? 実は、よく使うコマンドはそんなに多くないのです。

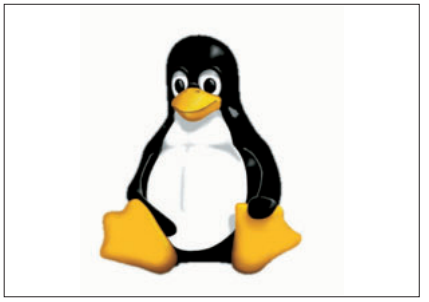
lsやcpのようによく使うコマンドを解説した入門書が数多く出版されているので、自分にあったものを1冊持っていると便利です。Linuxを利用しているうちに、これらのコマンドの使い方はすぐに覚えてしまいます。では、あまり使わないコマンドやオプション、ましてや初めて使うコマンドはどうすればいいのでしょうか? 答えは簡単、調べながら使えばいいのです。Linux(UNIX)には、manというオンラインマニュアルシステムがあります。“man <コマンド名>”と入力すれば、コマンドの使い方とオプションの説明が表示されます。決してわかりやすくはないと思いますが、なにもないのとは全然違います。日本語版のディストリビューションには、たいてい日本語化されたmanが付属していますので、まずは、

```
$ man man
```

として、manコマンドそのものを調べてみましょう。

Q 名前のわからないコマンドはどうする?

A manコマンドにはキーワード検索機能があります。“man -K <キーワード>”でキーワードを含むmanページを表示します。単純な検索なので、この結果のみですべてのコマンドが表示されるわけではありませんが、これはこれで結構便利です。



画面2 ちょっと太めのTux

ディストリビューション

Q ディストリビューションとはなんですか？

A 一言で答えるなら「配布パッケージ」ということになるでしょう。ご存じの方も多いと思いますが、厳密な意味でのLinuxは「カーネル」と呼ばれるコア部分（システムの基本サービスを提供する）だけのことを指します。しかし、それだけではユーザーが利用することはできないため、「ディストリビューター」と呼ばれる個人、法人や団体などがそれぞれのポリシーに合わせて、ライブラリ、基本コマンド、アプリケーションなど、さまざまなソフトウェアと組み合わせて「ディストリビューション」という形にして配布、もしくは販売しています。具体的なものとして、Slackware、TurboLinux、Vine Linux、Red Hat Linux、Debian GNU/Linuxなどがあります。それぞれのディストリビューションについての情報は、http://www.linux.or.jp/distributions/index.html（日本語）やhttp://www.linux.org/dist/index.html（英語）などに詳しく掲載されています。

Q どのディストリビューションが一番いいですか？

A ディストリビューションは、それぞれ特徴を持って作られており、どれも一長一短あります。したがって、Linuxを利用する目的がなにかによってお勧めできるディストリビューションも変わってきます。たとえば、業務用のサーバとしての利用を考えているなら、サポート体制のしっかりしたRed Hat LinuxやTurboLinuxなどが候補になりますし、Linuxというシステムの勉強が目的なら、システムの見通しがよいSlackwareやPlamo Linuxが候補になるでしょう。また、たくさんアプリケーションをインストールして使ってみたいならパッケージ管理機能のしっかりしたDebian GNU/Linux、しっかりした日本語対応を求めるならVine LinuxやLASER5 Linux、さらにWindowsも手放せないならMLD4という具合に、それぞれのディストリビューションが持つ特徴をまず知ることが必要でしょう。

画面3　日本における Linuxerの総本山「linux.or.jp」

Q 雑誌の付録としても配布されていますが、製品との違いはなんでしょうか？

A どのようなものを無料で配布するかという方針は、ディストリビューションによって異なりますが、大まかにいって製品版から商用アプリケーションやサポートなどを除いたものが雑誌の付録となっている場合が多いです。ただし、フリーでFTPサイトからダウンロードできるものであっても、機能が限定されているというわけではなく、Linuxシステムが持っている特徴などは製品版と変わりはありません。

Q パッケージとはなんですか？

A ある1つのアプリケーションに関連するプログラムや設定ファイル、ドキュメントなどをひとまとめにしたものです。ディストリビューションのことをパッケージと呼ぶこともあるためまぎらわしいのですが、ディストリビューションは、パッケージの集合体ともいえるので、そのように呼ばれるのでしょう。

Q RPMとはなんですか？

A RPM（Red Hat Package Manager）、Red Hat Softwareによって開発されたパッケージ管理方式のことです。RPMは、Red HatLinuxを始めとして、TurboLinux、Vine

Linux、 LASER5 Linux、 Open Linux、SuSE Linuxといった数多くのディストリビューションによって採用されており、次に述べるような機能を持っています。

- **パッケージ間の依存関係のチェック機能**
- **強力なクエリー機能**
- **パッケージの検査機能**

これらの機能は、rpmコマンドによって利用することができ、パッケージのインストール、アップグレード、削除を容易にしてくれます。rpmコマンドを利用する際、最低限必要なオプションは次のとおりです。

```
# rpm -i (パッケージのインストール)
# rpm -U (パッケージのアップグレード)
# rpm -e (パッケージのアンインストール)
# rpm --help (rpmコマンドの簡単なヘルプ)
```

これら以外にも便利なオプションはたくさんありますので、一度マニュアルやヘルプを見てみることをお勧めします。

また、RPMはパッケージのファイル名を見ただけでも、さまざまな情報がわかるようになっています（**図1**）。

さらに、最近ではコマンドラインからrpmコマンドを利用しなくても、Red Hat Linuxのパッケージ管理コマンド「glint」やGNOME標準のパッケージ管理コマンド「GnoRPM」といったGUIツールが登場していますので、コマンドラインが苦手な初心者ユーザーは、このようなツールを利用するのもよいでしょう。

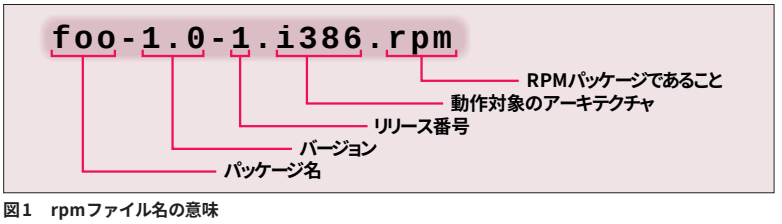


図1　rpmファイル名の意味

Q パッケージ管理方式はRPMだけでしょうか？

A Debian GNU/Linuxによって開発されたパッケージ管理方式として、deb方式があります。強力なパッケージ管理能力が特徴です。これまでは、Debian GNU/Linuxだけがこのdeb方式を利用してきましたが、Storm LinuxやCorel LINUXなどdeb方式を採用するディストリビューションが登場してきています。

deb方式のパッケージは、dpkgコマンドによって利用することができますが、dpkgのインターフェイスとしてdselectやaptなどを使うと便利です。

また、厳密にはパッケージ管理方式とは呼べませんが（依存関係のチェックなどを行わないため）、Slackwareや Plamo Linuxでは、バイナリにインストールシェルスクリプト加えてtar.gz形式（このファイル形式は、「tarボール」と呼ばれることもあります）でまとめたファイルを利用しています。これらのファイルは、pkgtoolコマンドなどを利用してインストール／アンインストールを行います。

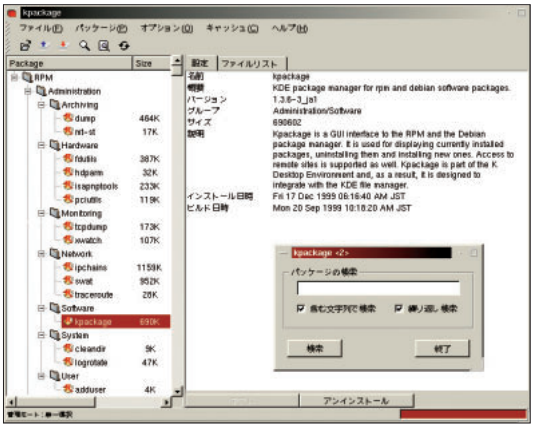
Q RPMとdebでは互換性はあるのですか？

A それぞれは別の管理方式ですので互換性はありません。ただし、Debian GNU/LinuxにはRPM形式を始めとするさまざまなパッケージ方式に変換するalienコマンドがあります。また、KDEのパッケージ管理コマンド「kpackage」はRPMとdebの両方をサポートしています（**画面4**）。

Q rpmファイルなら、rpm系のどのディストリビューションでもインストールできますか？

A これは、半分YESで、半分Noです。というのは、バイナリレベルでは互換性を持っていても、各ディストリビューションによってディレクトリ構成が異なっていたり、ライブラリのバージョンが異なっていることがあるからです。したがって、Red Hat Linux用に作られたrpmファイルをTurboLinuxにインストールした場合、うまく動作しないこともあります。また、その逆もあります。問題ない場合も多いのですが、トラブルを避けるためにも、極力対象ディストリビューション向けに作られたrpmファイルを利用することをお勧めします。

画面4　KDEのパッケージ管理ツール「kpackage」



ハードウェア

Q グラフィックスカードをインストーラが正しく認識しません

A Linuxではグラフィックス出力に、フリーのXサーバソフトウェアのXFree86を採用しています。Voodoo3やMillennium G400、RIVA TNT2などのグラフィックスカードは、XFree86 3.3.4以降でサポートされました。TurboLinux 4.0やLASER5 Linux 6.0、Vine Linux 1.1では、XFree86 3.3.3.1を使っているので、それらのグラフィックスカードを認識できません。インストールの時には、Xの設定をしないで、その後XFree86 3.3.5以降にアップグレードしてから、Xを設定すればよいでしょう。

アップデート用のXFree86は、それぞれのディストリビューションのFTPサイトに、RPMパッケージで用意されています。XFree86-3.3.5-*というファイルを見つけて、XFree86で始まるRPMファイルをダウンロードしておきます。次に“RPM -Uvh ＜パッケージ名＞”でアップグレードして、Xのセットアッププログラムを起動します。

セットアップは、XF86Setupコマンドで行います（**画面5**）。ディストリビューションによっては、Xconfiguratorコマンドやturboxcfgコマンドを使う場合もあります。

画面5　XF86Setupのグラフィックスカード選択ボタン

Red Hat Linux 6.1やKondara MNU/Linux、OpenLinux 2.3では、XFree86 3.3.5を採用していますので、インストーラが自動認識します。

お使いのグラフィックスカードがXFree86に対応しているかどうかは、マニュアルを読むか、それぞれのディストリビューションやXFree86.org（http://www.xfree86.org/）のWebサイトで調べてください。

Q 内蔵モデムが使えません

A 最近のノートPCに採用されている内蔵モデムは、DSPを使いソフトウェアによってモデム機能を実現しているいわゆる「Winmodem」というものです。Winmodemはソフトウェアでモデム機能が実現されていますので、現時点ではWindows上でしか使用できません。

しかし、Linux上でこのWinmodemを使えるようにドライバを開発する「linmodem」というプロジェクトが立ち上がっています。興味のある人は、http://www.linmodems.org/をチェックしてみてください。

また、Lucent社のチップを使ったWinmodemのLinux用ドライバに関して、バイナリパッケージがLucent社からリリースされているほか、「LTModem」（http://www.close.u-net.com/ltmodem.html）というプロジェクトも活動しています。

Q サウンドカードの設定はどうするのですか？

A Linuxのカーネルには、フリーのサウンドドライバが含まれています。しかしサウンドカードによっては、インストーラでは自動的に検出されない場合があります。そのときには手動で設定します。

Red Hat系のLinuxでは、sndconfigコマンドで設定します。root権限で、まず/usr/sbin/sndconfigを実行してください。もしインストールされていない場合には、RPMパッケージのsnd-xxx.i386.rpmとsndconfig-xxx.i386.rpmをインストールしてからsndconfigを実行します。またTurboLinuxでは、turbosoundcfgコマンドで設定します。

lsmodコマンドで、soundモジュールが組み込まれているか確認しておきましょう。

なお、設定された結果は、**リスト1**のように/etc/conf.modulesファイルに記録されます。ISAバスのサウンドカードなどで、I/OアドレスやIRQ、DMAポートなどが合っていないために動作しない場合には、conf.modulesファイルを正しい値にエディタで変更します。修正したら、

▼リスト1　/etc/conf.modulesファイルのサウンド設定部分（LASER5 Linuxの場合、内容はカードによって異なる）

```
alias sound sb
pre-install sound insmod sound dmabuf=1
alias midi opl3
options opl3 io=0x388
options sb io=0x220 irq=5 dma=1 dma16=5 mpu_io=0x330
```

```
# /etc/rc.d/init.d/sound restart
```

と行って、サウンドドライバを再起動します。

CreativeのSound Blaster（Live!を除く）やESS系サウンドチップ、YAMAHAのFM音源チップOPL3などを使用しているのならば、カーネル標準のドライバで動作します。

しかし、Sound Blaster互換製品やYAMAHAのYMF724、YMF744などを使用している場合には、商用のOSS（Open Sound System）ドライバを組み込む必要がありますので、サウンドドライバの入手先（**表1**）を参照してください。

Webサイト	内容
OSS/Free（The Linux Sound System） http://www.linux.org.uk/OSS/	フリーのサウンドドライバで、Linuxのカーネルに標準で含まれているため、新たに入手する必要はない
ALSA（Advanced Linux Sound Architecture） http://www.alsa-project.org/	Linuxのサウンドドライバを GPLに従って開発しているグループ。対応しているサウンドカードのドライバは、OSS/Freeとほぼ同様だが、サウンドカードによっては（特にGUSのドライバ）、OSS/Freeよりも拡張されている場合がある。APIはOSSと互換性がある。最新バージョンは、0.4.1h
商用OSS（Open Sound System） http://www.opensound.com/	4Front Technologies社が開発し、販売しているサウンドドライバ。プラットフォームはLinuxに限らず、広く主要なUNIX環境をサポートしており、デバイスドライバは、各OSプラットフォームで統一したAPIを提供する。販売されているのはバイナリのコードのみ。ドライバの機能はハードウェアごとに異なるが、OSS/Freeよりも拡張されているものが多く、新しいハードウェアのサポートも行われている
Creative Open Source page http://opensource.creative.com/	Sound Blaster Live!のLinuxドライバを、GPLライセンスに基づきソースコードで公開している。現時点ではバイナリは配布されていないので、自分でコンパイルしてインストールする必要がある。1999年11月より正式版となったが、0.3βバージョンから機能の追加はない

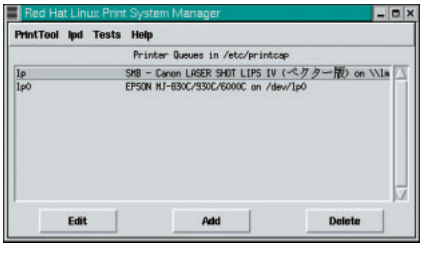
表1　サウンドドライバの入手先

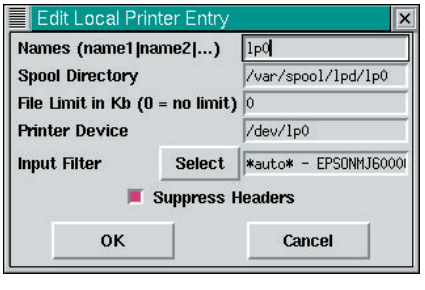
Q プリンタに出力したいのですが？

A TurboLinux、Vine Linuxなど、ディストリビューションによっては、インストール時にプリンタの設定ができるものがあります。

Red Hat LinuxやLASER5 Linuxでは、GNOMEのメニューからコントロールパネルを開き、Printtoolを使用して、プリンタの設定を行うことができます（**画面6**）。

パラレルポートに接続したローカルプリンタや、LANで接続されたネットワークプリンタはもちろんのこと、Windowsマシンで共有プリンタに設定したプリンタにもSMBプロトコルでプリントアウトすることができます。



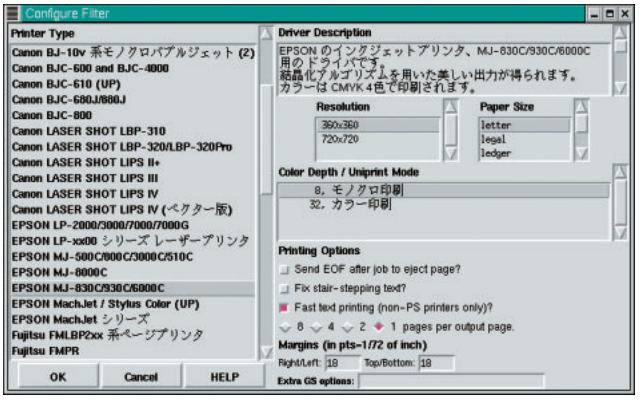


画面6
Red Hat系のディストリビューションに採用されているPrinttool。プリンタの設定をGUIで行うことができます。左上の画面のTestsメニューから「Print Postscript test page」を実行してテストページが正しく印刷ができれば設定は完了です。

Q CD-R/RWドライブにデータをバックアップするには？

A LinuxでCD-R/RWにデータを書き込む手順は、大きく3つの手順に分けることができます。まず、適当なディレクトリにCD-R/RWへ書き込むためのファイルを全部コピーしておきます。次に、mkisofsコマンドで、先ほど用意したディレクトリから、ハードディスク上に仮想ファイルシステムのイメージを作成します。念のため、そのファイルシステムをマウントして、内容を確認しておくとよいでしょう。最後に、そのファイルシステムのイメージを、cdrecordコマンドでCD-R/RWに書き込みます。

コマンドラインでは操作が面倒だという人は、X Window System上で動作するX-CD-Roastを使うとGUIで操作



することができます。X-CD-Roastは内部でmkisofsやcdrecordを呼び出しています。

詳しい情報は、CD-Writing HOWTO（http://www.linux.or.jp/JF/JFdocs/CD-Writing-HOWTO.html）をご覧ください。

Q UltraDMA/66対応のハードディスクは使えますか？

A 現在のディストリビューションに使われているカーネル2.2では、UltraDMA/66はサポートされていません。UltraDMA/33で使うことをお勧めします。

しかし、どうしてもというのでしたら開発版カーネル2.3ではサポートされていますし、カーネル2.2でもパッチを当てることで、UltraDMA/66も使うことが可能になっています。カーネルの再構築が必要ですから、ある程度知識と経験がないと大変な作業です。具体的なインストールの方法は、今月号131ページからの記事「高速ハードディスクをLinuxで使いこなす」を参考にしてください。

Q なんだか、ハードディスクのアクセスが遅いような気がします

A IDEハードディスクのアクセス設定が、PIO転送になっている場合があります。ハードディスクとマザーボードがDMA転送に対応しているのなら、DMA転送に設定することで高速にアクセスできる可能性があります。

dmesgコマンドで、Linux起動時のメッセージを見てみましょう。**リスト2**のような部分が見つけられたら、「BIOS settings」の右側にある「hda:pio」の部分に注目します。この場合、PIO転送になっています。もし「hda:DMA」になっていたならDMA転送になっています。なおhdaはドライブ名で、複数のドライブがついているならhdbやhdcのところも見ます。

または、下記のように行って、「using_dma = 0 (off)」と表示されたらDMA転送は使われていないことがわかります。

```
# hdparm -t -d /dev/hda
```

もし、DMAがオフの場合には、

```
# hdparm  -t -d 1 /dev/hda
```

とDMAをオンにテストしてみます。エラーが出力されずにテスト時間が速くなったなら、/etc/rc.d/rc.localの最下行に、「hdparm -d 1」を加えてブート時に設定するように変更します。

Q 64Mバイト以上のメモリが認識されません

A Linuxのカーネルが2.0.35以前のバージョンの場合は、64Mバイト以上のメモリを自動的に認識しません。その場合は、ブート時にカーネルにオプションをつけることでメモリの容量を指定することができます。たとえば、128Mバイトのメモリを指定するにはLILOと表示された時点で、

```
LILO: linux mem=0x8000000
```

または、

```
LILO: linux mem=128M
```

というように指定します。

毎回ブート時に指定するのは面倒なので、通常は/etc/lilo.confにオプションを書いて、liloを設定しておくほうがよいでしょう（**リスト3**）。lilo.confを書き換えたら忘れずに、

```
# /sbin/lilo
```

と行って、ブートセクタのliloに反映させます。これで次回のブート時から新しい設定になります。なお、起動後に認識されたメモリ容量を表示するには、freeコマンドを使います（**リスト4**）。

Q SCSIカードはなにが使えますか？

A SCSIホストアダプタは、業界標準ともいえるアダプテック社をはじめとして多くのメーカーの製品に対応しています。有力なディストリビューションのWebサイトには、そのディストリビューションが対応しているSCSIホストアダプタのリストが掲載されています。

なお、Linuxへの対応は、ハードウェアメーカーが保証やサポートを行うことはほとんどありません。しかし、Webページに動作確認情報を掲載したり、FTPサイトでLinux用ドライバを提供しているメーカーもありますので、製造メーカーのWebサイトを探してみてください。

▼リスト2 Linux起動時のハードディスク認識部分のメッセージをdmesgで見る

```
PIIX3: IDE controller on PCI bus 00 dev 39
PIIX3: not 100% native mode: will probe irqs later
    ide0: BM-DMA at 0xffa0-0xffa7, BIOS settings: hda:pio, hdb:pio
    ide1: BM-DMA at 0xffa8-0xffaf, BIOS settings: hdc:pio, hdd:pio
hda: IBM-DTTA-350640, ATA DISK drive
hdc: FX120T, ATAPI CDROM drive
```

Q ハードディスクのRAID対応は？

A COMPAQ SMART2やMylex DAC960といったRAID専用コントローラを使うことができます。カーネル2.2からソフトウェアRAIDが実用的に動作するようになりました。ソフトウェアRAIDを実現するためには、RAID toolsユーティリティをFTPサイトから手に入れる必要があります。

なお、RAIDサブシステムのように、ホスト側からは1台のハードディスクドライブに見せるタイプのものは、そのままでもちろん使えます。

Q USBデバイスは使えますか？

A LinuxのUSBサポートは、バージョン2.2.7からカーネルのソースに含まれるようになりました。しかし、USB接続のマウスとキーボードをサポートするためのコードは含まれていますが、カーネルのコンパイル時のオプションがコメントアウトされているのを見ればわかるように、現時点では開発途上であり、一般ユーザーが使えるレベルではありません。

しかし、自分でカーネルをコンパイルしてUSBマウスを実際に使っているユーザーもいるようです。開発系カーネル2.3系では、USB接続のデジカメやプリンタ、ZIPドライブなどのストレージデバイスをサポートするためのコードなどが追加されています。USBデバイスはカーネル2.4が正式にアナウンスされるまで待ったほうがよいでしょう。

LinuxのUSBサポートに関する情報は「Linux USB」（http://www.linux-usb.org/）を参照してください。

Q IrDAは使えますか？

A IrDA（赤外線ポート）のサポートは、2.2系カーネルから含まれるようになっています。シリアルポートのエミュレーションを行うプロトコルであるIrCOMMは安定して動作しているようです。IrCOMMを使うと、IrDA対応のISDN公衆電話やIrDAポートが装備されている携帯電話とやり取りができるようになります。また、IrDAのプロトコルの一つであるIrOBEXを使うことで、PalmIIIとデータ交換ができるようになるとドキュメントに記述があります。詳しくは「The Linux IrDA Project」（http://www.cs.uit.no/linux-irda/）を参照してください。

Q MOドライブを接続したいのですが

A SCSI接続のMOドライブを使うことができます。ただし、540Mバイト以下のMOと640Mバイト以上のMOでは、物理フォーマットが異なっているために、注意しなければいけないことがあります。

物理的なセクタサイズが、540MバイトまでのMOでは512バイトだったの対して、640Mバイト以上のMOでは2048バイトに変更されています。そのため、カーネル2.0系を使っているディストリビューションでは、カーネルにパッチを当てる必要があります。2.2系のカーネルでは問題ありません。

なお、MOではないですがDVD-RAMを使う場合にも、セクタサイズが2048バイトのため同様の注意が必要です。

カーネル2.0.x用の2048バイト／セクタ対応パッチについては、「640MB MO on Linux」のWebサイト（http://liniere.gen.u-tokyo.ac.jp/2048.html）を参照してください。

もちろん、Windows標準であるFATフォーマットのMOを読み書きすることもできますので、サイズが大きめのデータを交換する際には役に立つでしょう。

▼リスト3　メモリ容量128Mバイトに設定する場合の/etc/lilo.confの例

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
image=/boot/vmlinuz
        label=linux
        root=/dev/hda1
        initrd=/boot/initrd
        append="mem=0x8000000"        ← ここに追加する
        read-only
```

▼リスト4　メモリ、スワップの使用量はfreeコマンドで調べることができる

```
# free
             total       used       free     shared    buffers     cached
Mem:        128076      72972      55104      29896       5812      36508
-/+ buffers/cache:      30652      97424
Swap:       128484          0     128484
```

システムの起動

Q Linuxのインストール CD-ROMから起動できない

A 古いマシンやCD-ROMドライブだと、CD-ROMからのブートはサポートされていません。マザーボードによっては、BIOSをアップデートすればCD-ROMブートが可能になるかもしれません。SCSI接続のCD-ROMドライブの場合、SCSIアダプタもCD-ROMブートに対応していなければなりません。

また、マザーボードのBIOS設定で、CD-ROMの起動順位をハードディスクより先にしておかないと、ハードディスクにあるOSが起動してしまいますので注意しましょう。

CD-ROMブートができなくても、悲観することはありません。インストールフロッピーを作り、これで起動すればよいのです。インストールCD-ROMの中に、インストールフロッピーのイメージファイルと、このイメージをフロッピーディスクに書き込むDOSプログラム（rawrite.exe）が入っているはずです。これらを使い、DOSかWindowsでインストールフロッピーを作成しましょう。

Q LILOプロンプトがLIで止まって起動しない

A LILO（LInux LOader）が、表示途中で止まるのにはいくつかの原因が考えられますが、Linuxを何度かインストールするうちに、不要になったLILOがMBR（マスターブートレコード）に残り、悪さをすることがあります。この場合はDOSやWindows 95/98で“ FDISK /MBR ”を実行して古いLILOを消します。

LILOは、Linuxを起動するときに、BIOSを使ってカーネルを読み出しますが、いにしえからの伝統を引きずったBIOSは、ハードディスクの1024シリンダ以降にアクセスできないのです。このため、カーネルが1024シリンダ以降の部分に配置されていると、Linuxを起動できません。近頃の大容量ハードディスクでは、この問題が起きることがあります。

ただし、シリンダが1024以上あるハードディスクでも必ず起動できなくなるわけではありません。カーネルが1023シリンダまでに収まっていれば起動しますが、カーネルの再構築をして、新しいカーネルが1024シリンダ以降に配置されるとLILOからの起動ができなくなります。

この問題を解決するには、カーネルを1023シリンダ以内に置くか、LILOを使わずにブートするしかありません。Red Hat Linuxなどでは、カーネルを/bootディレクトリに置くので、ハードディスクの前のほう（1023シリンダ以内）に10～15Mバイト程度のパーティションを作り、これを/bootにマウントすれば問題を回避できます。

このほかにも、LILOではなく、BIOSの制限を受けないブートセレクタを使う、あるいはフロッピーディスクからカーネルを読み出して起動するという回避方法もあります。

このような時のために、起動フロッピーは、できるだけ作るようにしましょう。

Q ときどき起動にもの凄く時間がかかるのはなぜ？

A Linuxでは、fsckというファイルシステムのチェックプログラムが使われています。これは、Windowsのスキャンディスクのようなものです。このfsckは、一定の回数ディスクをマウントすると起動時に自動実行されます。たとえば、Red Hat Linux 6.1では、ファイルシステムを20回マウントするごとにfsckを自動実行します。また、前回の終了時に正しくシャットダウンされていない場合も実行されるようになっています。チェックするパーティションが小さければすぐに済みますが、パーティションが大きくなるとそれなりに時間がかかるようになります。

また、メール配送プログラムのSendmailは、起動する際にDNS（Domain Name System）サーバに接続しようとします。このときにDNSサーバが見つからないとSendmailがタイムアウトするまで数分間待たされることになります。Sendmailを使わないなら、自動的に起動しないようにしましょう。

Q rootのパスワードを忘れてしまったのですが、再インストールするしかないですか？

A このようなときは、メンテナンス用の“ シングルユーザーモード ”というモードで起動してパスワードを再設定することができます。

シングルユーザーモードで起動するには、LILOの“ boot： ”プロンプトで“ linux 1 ”と入力します。この“ linux ”は起動するOSを選ぶためのラベルで、環境によっては違うかもしれ

ません。その場合は、“ boot： ”プロンプトに対してTabキーを押すと、登録されているラベルが一覧表示されるので、その中からLinuxを起動するものを選びます。

シングルユーザーモードで起動するとシェルのコマンド画面になりますので、passwdコマンドでパスワードを再設定します。

さて、rootのパスワードがわからないと、シャットダウンもできないことが多いでしょう。この場合、まずsyncコマンドを実行して、ディスクキャッシュに残っているデータをディスクに書き込みます。次にCtrl＋Alt＋Deleteキーを押して終了します。

Q ランレベルってなに？

A Linuxには、ランレベルという複数の動作モードがあり、0～6のランレベルによって、機能の範囲を変えられるようにしています。たとえば、Red Hat LinuxやTurboLinuxでは、**表2**のようになっています。

通常のテキストコンソール起動時は、ランレベル3で動作し、X Window System起動時はランレベル5で動作しているというわけです。Linuxは、マルチユーザーOSなので、同時に複数のユーザーで利用できますが、管理者がディスククラッシュを復旧する際などは、ほかのユーザーやデーモンがファイルを読み書きすると非常にマズいことになります。このようなときには、ほかのユーザーやデーモンを排除するためにシングルユーザーモードで作業をします。管理者はマシンの前に座り、一人寂しく作業をするのです。

起動時のデフォルトランレベルは、/etc/inittabで設定されています。このファイル中に、id:3:initdefaultと書かれていればデフォルトのランレベルは3になります。これを書き換えるか、起動時にLILOの“ boot： ”プロンプトで“ linux 5 ”のようにすれば、ランレベルを指定することができます。

一方、起動してからランレベルを切り替えるのがtelinitコマンドです。たとえば、“ telinit 6 ”を実行すると、すぐにランレベル6となり、Linuxが再起動します。

Q 起動フロッピーを作りたい

A ハードディスクからLinuxを起動できなくなったときなどに活躍するのが起動フロッピーです。

ディストリビューションによっては、起動フロッピーを作るためのシェルスクリプトが用意されているものもありますが、ここでは標準的なコマンドを使い、シンプルな起動フロッピーを作る方法を紹介します。

作成手順は、たったの3ステップだけです。まず、フロッピーディスクをフォーマットします。

```
# fdformat /dev/fd0H1440
```

次に、カーネルのイメージを書き込みます。

```
# dd if=/boot/vmlinuz of=/dev/fd0
```

最後に/（ルート）ファイルシステムのパーティションを指定して終了です。

```
# rdev /dev/fd0 /dev/hda1
```

このフロッピーで起動すると、カーネルをフロッピーディスクから読み出し、/dev/hda1のパーティションを/（ルート）ファイルシステムとしてマウントします。

しかし、カーネルに渡すオプションが入力できないので、ランレベルの指定などはできません。そこで、今度は、オプションを指定できる、もう少し便利な起動フロッピーを作ってみます。このフロッピーには、LILOとカーネルイメージを入れ、フロッピー内のカーネルとハードディスクのカーネルを選べるようにします。

下準備として、フロッピーディスクに書き込むLILOのために、設定ファイル/etc/lilo.flopを作ります（**リスト5**）。/etc/lilo.confをコピーして雛形にすると簡単です。

次に、起動フロッピーを作成します。最初にフォーマットし、ext2ファイル

0	停止
1	シングルユーザーモード
2	マルチユーザーモード（ネットワーク機能なし）
3	フルマルチユーザーモード（テキスト）
4	未使用
5	マルチユーザーモード（X Window System）
6	再起動

表2　ランレベルによるLinuxの機能

▼リスト5　/etc/lilo.flopの例

```
boot=/dev/fd0
#LILOはフロッピーに置く
map=/mnt/floppy/lilo-map
prompt
timeout=50
default=linux

image=/boot/vmlinuz-2.2.13
#ハードディスクのカーネル
        label=linux
        read-only
        root=/dev/hda1

image=/mnt/floppy/vmlinuz-
2.2.13
#フロッピーディスクのカーネル
        label=rescue
        read-only
        root=/dev/hda1
```

システムを作成します。

```
# fdformat /dev/fd0H1440
# mke2fs /dev/fd0
```

作成したディスクをマウントし、ハードディスクからチェインローダとカーネルをコピーします。

```
# mount -t ext2 /dev/fd0 /mnt/floppy
# cp -p /boot/chain.b /mnt/floppy
# cp -p /boot/vmlinuz-2.2.13 /mnt/floppy
```

最後に、フロッピーディスクにLILOをインストールし、アンマウントすればできあがりです。

```
# lilo -C /etc/lilo.flop
# umount /mnt/floppy
```

この起動フロッピーでブートすると、LILOの“boot：”プロンプトが表示されるので、ここで“linux”と入力すればハードディスクのカーネルが、“rescue”と入力すればフロッピーディスクのカーネルが起動します。さらに、“rescue 1”などと入力すればランレベルの指定も可能です。

ただし、ブートメディアのドライバをモジュールにしている場合は、これらの方法で起動ディスクを作ることはできません。ブートに必要なドライバをカーネルに組み込んでしまえば、簡単に起動フロッピーを作成できます。

Q デーモン（daemon）とはなんですか？

A ゲームのように、ユーザーと対話しながら表舞台で活躍するアプリケーションとは違い、人目につかないところでひっそりと仕事をする、「縁の下の力持ちプログラム」の総称です。“ps ax”コマンドを実行するとわかるように、Linuxでは数多くのプログラムがデーモンとして動作しています。たとえば、仮想記憶を実現するkswapd、ログを集めてファイルに記録するklogdやsyslogd、各種のネットワークサーバプログラムなどはデーモンとして動作しています。

デーモンは、ユーザーと直接対話する制御端末を持ちません。ですから、一般のユーザーは、デーモンが動いているかどうかを気にかけないでしょう。

起動されたデーモンは、すぐに活動するわけではなく、特定のリクエストを待ち、リクエストを受信するとそれに対する処理を行います。そして処理が済むと、次のリクエストが来るまでまた休むのです。たとえば、Webサーバのhttpdは、ユーザーから接続されるまで休んでいて、接続されると活動を始めます。また、決められた時刻や一定時間ごとにプログラムを起動するcrondも代表的なデーモンです。

Linux、UNIXの世界において、デーモンは、悪魔や悪霊ではなく、ギリシャ神話の「神と人の間に位置する超自然的存在」に由来するものとされています。

Q デーモンの起動、停止の方法を教えてください

A システムの起動時に、自動的に起動するデーモンを設定したい場合はツールを利用するのが便利です。TurboLinuxではturboserviceコマンド、Red Hat Linuxならntsysvコマンドが利用できます。また、X Window Systemで動作するLinuxconfやksysvコマンドが使えれば、グラフィカルなユーザーインターフェイスで設定できます。

では、今すぐデーモンを起動したいときはどうすればよいのでしょう。Red Hat Linuxなどでは、/etc/rc.d/init.dディレクトリ以下にデーモンの起動スクリプトが用意されていますので、起動したいデーモン名のファイルに、引数としてstartをつけて実行します。たとえば、httpd（Webサーバ）を起動したいなら、

```
# /etc/rc.d/init.d/httpd start
```

とするとhttpdが起動されます。引数にはstopとrestartも使え、それぞれ停止と再起動を行います。

一方、Slackwareなどでは/etc/rc.d/ディレクトリにrc.httpdのようなファイルが用意されているので、これを実行すれば、そのデーモンが起動します。

また、止めたい場合はps axを実行して、止めたいデーモンプロセスのPID（プロセスID）を調べ、killコマンドを使います。例として、PIDが567のデーモンを止めてみましょう。

```
# kill -TERM 567
```

Red Hat Linuxなどでは、/var/runディレクトリに、主要なデーモンのPIDを記録したファイルがあるので、これを利用して次のようにすることもできます。

```
# kill -TERM `cat /var/run/httpd.pid`
```

再起動する場合は、killコマンドのオプションを-TERMの代わりに-HUPにします。デーモンの設定ファイルを変更したら、このようにしてデーモンを再起動し、変更内容を動作に反映させます。

ルートユーザー

Q rootで使い続けるのはよくないといわれますが、どうしてでしょうか？

A 大きく分けて理由は3つあります。まず、rootユーザーは、システムに関するあらゆる権限を持っています。そのため、システムクラッシュ、ファイルの破壊などが行えてしまいます。そして、LinuxのようなOSは、それが簡単にできてしまいます。たとえば、“ rm -rf ＊ ”を実行すると、カレントディレクトリ以下にあるファイルとディレクトリを問答無用ですべて削除してしまいます。

次に、セキュリティ確保の問題があります。rootユーザーとしてログインしているときに離席する場合は、マシンにイタズラをされないよう、なんらかの防護策をとらないといけません。防護策が必要なのは一般ユーザーであっても同じですが、前述の権限などの理由により、イタズラされた場合にシステムが受けるダメージはケタ違いに大きくなります。

最後に、Linuxの便利なリモートツールが利用できません。最近のディストリビューションの多くは、セキュリティ上の理由からrootユーザーがtelnetやFTPを利用してアクセスできないようにしてあります。これでは、Linuxの持つメリットが半減します。

これらの問題を防ぐために、一般ユーザーを作っておき、ふだんはそのユーザーで作業するようにしておきます。そして、rootユーザーの権限が必要な作業が発生したときだけ、suコマンドを利用して、一時的にrootユーザーとなり、作業が終わったらexitして、すぐに一般ユーザーに戻ることをお勧めします。

Q 一般ユーザーはどのようにして作るのですか？

A 新しいユーザーを作る方法はいくつかあります。LASER5 Linux 6.0やVine Linux 1.1などのユーザーなら、Linuxconfを使って作成できます。また、TurboLinux 4.xのユーザーならturbousercfgコマンドで作成することができます。また、useraddコマンドを利用してコマンドラインから作成することもできます。

Q suコマンドを実行してrootになったのに、管理者コマンドが使えない

A オプションをつけずにsuコマンドを実行すると、直接rootユーザーとしてログインしたときとは違い、そのユーザーの環境を引き継いだままroot権限を持ちます。これにより、PATHの設定も元のユーザーと同じになるので、管理者コマンドが置かれた/sbinや/usr/sbinなどにはパスが通っていません。

管理者コマンドを利用する場合は、絶対パス指定でコマンドを実行するか、suコマンドに－（ハイフン）オプションをつけてrootユーザーになります。このオプションをつけると、直接rootユーザーとしてログインしたときと同じ環境に初期化されますので、/sbinや/usr/sbinなどにもパスが通るようになります。

Q rootでtelnetやFTPログインができない

A これは、セキュリティを確保するための措置です。

telnetやFTPでは、ユーザー認証の際にパスワードをそのままのテキスト（平文）のままで回線に流します。これはとても危険です。

一般ユーザーとしてtelnetログインし、その後suコマンドでrootになることは可能です。しかし、インターネットに接続したマシンでは、telnetを使うべきではありません。

それでは不便ですから、telnetの代わりにSSH（Secure Shell）を使いましょう。SSHは、ホスト間の通信とユーザー認証を公開鍵暗号によって暗号化します。これにより、リモートログインや、ホスト間でのファイルコピーを安全に行えます。ただし、商用利用する場合は有償のライセンスを受ける必要があります。詳しくは、SSHのWebページ（http://www.cs.hut.fi/ssh/）を参照してください。また、SSHをフリーで実装するOpenSSHプロジェクト（http://www.openssh.com/）もあります（**画面7**）。

画面7　OpenSSHプロジェクトのWebページ

シェル

Q シェルとはなんですか？

A 広義には「ユーザーとシステムの橋渡しをするインターフェイス」ということになります（**図2**）。ただし、Linuxでは「コマンドラインに渡された文字列を解釈して、その命令をシステムに渡すもの」（コマンドラインインタープリタ）の意味で使われることがほとんどです。元々コマンドラインでの利用が主流だったUNIXでは、コマンドラインインタープリタとしてのシェルは数多く作られており、Bourneシェル（sh）、Cシェル（csh）、Kornシェル（ksh）、tcsh、bash、zshなどの種類があります。

Q それぞれのシェルはどんな特徴がありますか？

A これらのシェルは、sh系とcsh系に大別することができます。sh系の大元になるのがBourneシェルです。Bourneシェル（sh）は、AT&Tベル研究所のSteven Bourne氏によって作られたもので、古くからUNIXでは標準的にバンドルされてきたシェルです。Bシェルと呼ばれることもあります。このBourneシェルはUNIXでは標準的な存在ながら、非常にプリミティブな機能しか持っていません。

それとは別にcsh系の大元となるのがCシェルです。Cシェルは、UCB（カリフォルニア大学バークレー校）で、viやBSDの開発で知られるBill Joyによって作られました。ジョブ制御やエイリアスなどBourneシェルにない機能を持っています。また、C言語に似た文法でスクリプトを記述できるという特徴を持っています。

これらの2つシェルを中心に、拡張版としていくつかのシェルができました。sh系の後継として、Kornシェルやbashが開発されました。Kornシェル（ksh）はベル研究所のDavid Korn氏によって開発されました。Kornシェルは、BourneシェルとCシェルの優れた機能を取り込んだシェルで、商用製品です（以前のバージョンを基にパブリックドメインとして提供されているものはある）。

一方のcsh系には、tcshがあります。tcshは、Cシェルの機能に加えて、コマンド行編集が改善されています。

Linuxで標準的に利用されているbashは、*Bourne Again Shell*の略称で、GNUプロジェクトによる成果物です。Bourneシェルとの互換性だけでなく、CシェルやKornシェルの機能も取り込んでいます。基本機能だけでもあらゆるシェルの特徴を持っていますが、カスタマイズ力も高くなっています。

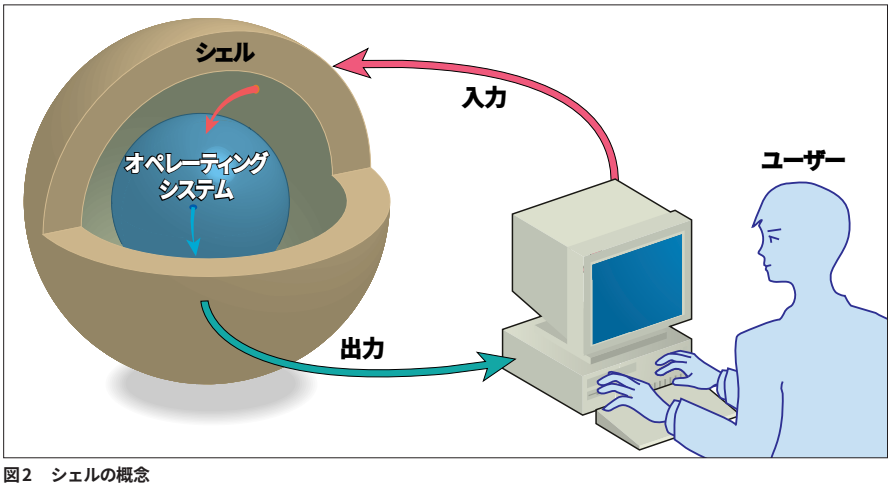


図2　シェルの概念

Q 自分が使っているシェルを知りたいときは？

A コマンドラインから、

```
$ echo $SHELL
```

と入力します。Linuxではほとんどの場合、bashを使用しているでしょう。

Q シェルを変更するには？

A コマンドラインから、chshコマンドを実行することでシェルを変更できます。さらに、LinuxconfやturbousercfgなどのGUIツールからも変更可能です。

Q ホームディレクトリにある.bash_profile、.bashrc、.bash_logoutは、それぞれどのような働きをするのでしょうか？

A 簡単にいうと、.bash_profileはシステムにログインする際に読み込まれる環境定義ファイルです。.bash_loginや.profileなども.bash_profileと同様の働きをして、ホームディレクトリに.bash_profileがなけれ

ば、.bash_loginを探し、.bash_loginがなければ、.profileをという順番で探していきます。特別な理由がない限り、.bash_profileを利用すればいいでしょう。

.bashrcはサブシェルが起動されるたびごとに読み込まれる環境定義ファイルです。ただし、多くのディストリビューションでは、ログインシェルとサブシェルを区別なくコマンドを実行できるよう、.bashrcを.bash_profileの中で読み込むような設定になっています。

なお、起動時には、これらのファイルの前に、/etc/profileがあれば、このファイルが一番先に読み込まれます。

.bash_logoutは、ログアウト時に読み込まれて実行されるファイルで、テンポラリファイルの削除やログの書き込みコマンドなどを書いておけばいいでしょう。

Q lsしても.bash_profile、.bashrc、.bash_logoutが表示されません

A 標準ではlsコマンドは、先頭に「.」（ドット）のついたファイル（ドットファイル）は表示しません。表示させたいときは、

```
$ ls -a
```

として、-a（all）オプションつきで実行します。

Q .bash_profileを変更したのに、有効になりません

A .bash_profileはログイン時に一度しか読み込まれません。変更した設定を有効にしたいなら、

```
$ source ~/.bash_profile
```

か、

```
$ . ~/.bash_profile
```

として、再度読み込ませることで、変更が有効になります。

Q bashで変数を設定するときexportを実行するものとしないものがありますが、違いがわかりません

A 次のように、

変数名=値

として設定する変数は、「シェル変数」と呼ばれる変数の定義法です。これは、複雑なシェルスクリプトを作るときに値の保存に使います。また、組み込み変数と呼ばれるbashが参照する値を設定しておいて、シェルをカスタマイズしたり、シェルの状態を取得したりします。

一方、

```
export 変数名=値
```

のように設定する変数は、「環境変数」と呼ばれる変数の定義法です。環境変数は、コマンドの動作環境を構築するためのもので、環境変数だけがサブプロセスにも伝わります。逆にいえば、このことはサブプロセスにも伝えたい変数は環境変数として定義すべきであることを意味します。具体的には、環境変数EDITORなどがそれに当たります。

Q コマンドラインからの入力をよく間違えます

A bashの補完機能を利用することをお勧めします。コマンドライン上で、コマンド名やファイル名などを途中まで入力したあと、TABキーを押せばbashが検索パス上から該当する文字列を補ってくれます。また、lsをksと打ち間違えるようなら、

```
$ alias ks=ls
```

とすれば、それ以降はksと打ち間違えても、正しいコマンド（ls）が実行されます。ただし、これはあまりお勧めできる方法ではありません。

Q コマンドを入力するとき文字列は'と"のどちらで囲めばいいのでしょうか？

A 単なる文字列をコマンドに渡すだけなら、どちらでも構いません。しかし、変数などをコマンドに渡すときに'（シングルクォート）と"（ダブルクォート）では、その挙動は異なります。たとえば、

```
$ editor=emacs
```

としておいて

```
$ echo "I like $editor"
```

とダブルクォートでくくると、

```
$ I like emacs
```

と表示されますが、

```
$ echo 'I like $editor'
```

とシングルクォートでくくると、

```
$ I like $editor
```

となり、変数は展開されません。

ディスク・ファイル

Q フロッピーディスクやCD-ROMをマウントするってなに？

A WindowsやDOSとは違い、Linuxにはドライブレターという概念はなく、すべてのファイルがひとつの木構造に収まっています。すなわち、/（ルート）ディレクトリの下にすべてのディレクトリ、サブディレクトリ、ファイルがあります。

フロッピーディスクやCD-ROMを使うときは、この木構造の一部にこれらをくっつけて（マウント）使います。マウントするにはmountコマンドを使います。例として、/mnt/cdromというディレクトリ（これをマウントポイントという）にCD-ROMをマウントしてみましょう。

```
# mount -t iso9660 -r /dev/cdrom
    /mnt/cdrom
```

-t iso9660というオプションは、CD-ROMで使われるISO-9660というファイルシステムを使うという指定です。-rは、読み出し専用でマウントするというオプションです。マウントしたあとで、“ ls /mnt/cdrom ”と入力すると、CD-ROMの内容が表示されます。

CD-ROMを取り出す時には、木構造からはずす（アンマウント）作業が必要です。これにはumountコマンドを使います。

```
# umount /mnt/cdrom
```

このようにして、ひとつの木構造を保ったまま、リムーバブルメディアを利用できるのです。

リムーバブルメディア以外にも、ハードディスクをマウントすることももちろん可能です。実際に、Linuxが起動するときには、ハードディスクのパーティションを/（ルート）にマウントします。環境によっては、/bootや/homeなどを別々のパーティションにマウントしていることもあります。

Q rootユーザーしかマウントできないので不便です

A ほとんどのディストリビューションでは、rootユーザーにしかマウント／アンマウントを許可していません。

一般のユーザーにもCD-ROMのマウント／アンマウントを許可する場合は、/etc/fstabファイルをエディタで開き、CD-ROMデバイスの行の第4フィールドに“ user ”、または“ users ”を加えます。たとえば、第4フィールドが“ noauto,ro ”になっているなら、“ noauto,ro,user ”のようにします。フロッピーディスクなどでも同じ要領で設定できます。

“ user ”を加えた場合は、誰でもマウントできますが、そのデバイスをマウントしたユーザー以外はアンマウントができなくなります。一方、“ users ”を加えた場合は、ほかのユーザーでもアンマウントできるようになります。

安全のため、ハードディスクのマウント／アンマウントは、rootユーザーだけが行なえるようにします。

Q アンマウントできないのですが

A umountコマンドやejectコマンドを使おうとすると“ device is busy ”と表示されてアンマウントできないことがあります。このようなときは、アンマウントしようとしているファイルシステムを利用しているユーザーやプログラムがあるということです。たとえば、/mnt/cdromにCD-ROMをマウントしているときに、カレントディレクトリが/mnt/cdrom/fooなどになっているとアンマウントできません。

Q いちいちCD-ROMをマウントするのが面倒です

A automountデーモンを使ってCD-ROMなどを自動的にマウントすることができます。/misc/cdにCD-ROMデバイスを自動マウントする設定を説明します。

エディタで/etc/auto.masterを開き、次のように記述します。

```
/misc  /etc/auto.misc --timeout 60
```

続いて、エディタで/etc/auto.miscを開いて次のように記述します。

```
cd -fstype=iso9660,ro :/dev/cdrom
```

2つのファイルを用意したら、automountデーモンを起動します。Red Hat LinuxやTurboLinuxでは、/etc/rc.d/init.d/autofs startを実行すれば起動します。この設定例では、CD-ROMをドライブに入れるとすぐに/misc/cdにマウントされ、その後、60

秒間使われないと自動的にアンマウントされます。そして、またアクセスされると自動的にマウントします。

Q ディスクの使用状況を知りたいのですが

A dfコマンドを実行すると、各パーティションの利用状況が表示されます。

画面8の例では、3つのパーティションが、それぞれ/（ルート）、/boot、/varにマウントされていることと、パーティションごとの利用している容量、全体容量と利用率がわかります。

またduコマンドは、ディレクトリごとのファイル容量を表示します。

```
# du /usr/man/
9956    /usr/man/man3
9364    /usr/man/man1
1196    /usr/man/man2
472     /usr/man/man4
1404    /usr/man/man5
120     /usr/man/man6
704     /usr/man/man7
2104    /usr/man/man8
8       /usr/man/man9
2832    /usr/man/mann
28344   /usr/man
```

数字は、各ディレクトリのファイル容量です。デフォルトではKバイト単位となるので、/usr/man以下に全部で28344Kバイトのファイルがあることがわかります。

```
# df
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/hda7              5644996    973560   4384680  18% /
/dev/hda1                23302      4013     18086  18% /boot
/dev/hda6               401548    152060    249488  38% /var
```

画面8　dfコマンドの出力結果

Q 128Mバイトのメモリを搭載しています。スワップはどれくらいが適当ですか？

A 利用するプログラムの種類、数、負荷によって最適なスワップの大きさは変わるので、コレが正解というものはありません。一般には、64〜127Mバイトもあれば十分です。古いカーネルでは、1パーティションで128Mバイト以上のスワップスペースは利用できません。いずれにしても、このスワップを使い切るようであれば、パフォーマンスは著しく低下していますから、素直にメモリを増設しましょう。

Q 消したファイルを復活できますか？

A いいえ、できません。rmコマンドに-iオプションをつけると、ファイルを消す前に、本当に消すかシステムが確認してきますので、活用しましょう。

Q DOSのフロッピーディスクを読み書きしたい

A DOSのフロッピーディスクは、Linuxではmsdosファイルシステムとして認識されますが、カーネルがmsdosファイルシステムをサポートするようにコンパイルされていないと利用できません。利用しているカーネルがサポートするファイルシステムは、/proc/filesystemsに書かれています。

次の例では、DOSのフロッピーディスクを/mnt/floppyにマウントしています。

```
# mount -t msdos /dev/fd0 /mnt/floppy
```

取り出す前には、アンマウントしてデータがすべて書き込まれるのを確認します。アンマウントしないで取り出してしまうとデータが破損します。

```
# umount /mnt/floppy
```

ディストリビューションによっては、DOSのフロッピーディスクを使うためのmtoolsというツールが収録されています。これは、UNIX系のOSでDOSのフロッピーディスクを使うためのコマンドラインツール集です。マウントすることなく、ファイルのコピーやフォーマットができます。

使っているディストリビューションに収録されていなければ、http://mtools.linux.lu/から入手することができます。

Q Macintoshのフロッピーを読み書きしたいんだけど

A カーネルがhfsファイルシステムをサポートしていれば、マウントして使うことができます。デフォルトでhfsをサポートしているPC用ディストリビューションはほとんどありませんので、モジュールを作るか、カーネルの再構築をしてhfsを使えるようにします（**画面9**）。

モジュールにしたときは、使う前にモジュールをインストールします。

```
# insmod hfs
```

画面9
xconfigによるカーネルのコンフィグレーション画面。Linuxはいろいろなファイルシステムをサポートしている。

hfsファイルシステムを使えるようになったら、マウントします。

```
# mount -t hfs /dev/fd0 /mnt/floppy
```

ただし、Macintoshの2DDフロッピーディスク（800Kバイトフォーマット）はPCで利用できません。これはハードウェアの制限なので、ソフトウェアではどうしようもありません。

Q Windowsパーティションはどうしたら読み書きできますか？

A カーネルがvfatファイルシステムをサポートしていれば、Windows 95/98のパーティションを利用できます。しかし、そのままでは日本語のファイル名を扱えませんので、パッチを当ててカーネルを再構築する必要があります。このパッチは、小柳雅明さんが作成したものを、川口 浩さん、松嶋 明宏さんが改良したものです。最新版は松嶋さんのWebページ（http://www.jaist.ac.jp/~amatsus/）で公開されています。

ディストリビューションによっては、vfatファイルシステムで日本語のファイル名を扱えるカーネルを採用しているものもあります。

Q ディスクを増設して/homeディレクトリの容量を増やしたいのですが？

A 新しいパーティションを適当なところにマウントして、/homeからシンボリックリンクを張ります。

まず、増設するディスクを接続してLinuxを起動します。増設したディスクを/dev/hdbとすると、次のようにfdiskコマンドを起動して必要な容量のLinuxパーティションを作ります。

```
# fdisk /dev/hdb
```

ここでは、新しいパーティション/dev/hdb1を作ったとして説明を続けます。このパーティションにext2ファイルシステムを構築します。

```
# mke2fs /dev/hdb1
```

利用する前に、fsckを実行してファイルシステムを検査します。

```
# fsck /dev/hdb1
```

ファイルシステムに問題がなければ、/disk2というマウントポイントを作り、/dev/hdb1を/disk2にマウントします。

```
# mkdir /disk2
# mount -t ext2 /dev/hdb1 /disk2
```

次に、/homeの内容を/disk2にコピーします。

```
# cd /disk2
# tar -cf - -C / home | tar -xvpf -
```

この例では、カレントディレクトリを/disk2に移し、tarコマンドを使って/にあるhomeディレクトリ以下を/disk2にコピーしています。コピーが終わったら、/homeの中にあるディレクトリとファイルを消し、/homeから/disk2/homeへシンボリックリンクを張ります。

```
# rm -rf /home
# lm -s /disk2/home /home
```

最後に、/etc/fstabに次の行を追加して、起動時に/dev/hdb1がマウントされるようにします。

```
/dev/hdb1 /disk2 ext2 default 1 1
```

Q Linuxのパーティションが削除できない

A Linuxのfdiskなどで作ったパーティションには、DOSやWindowsのFDISKでは消せないものがあります。Linuxのインストールディスクで起動して、Linuxのfdiskを使って消しましょう。

Q Linuxを削除したけど、LILOが消えない

A MBR（マスターブートレコード）に居座ったLILOを消すには、DOSかWindows 95/98のコマンドラインから“FDISK /MBR”を実行します。これで、MBRが新しいものに上書きされるのでLILOを消すことができます。Windows NTしかないときは、Windows NTの起動ディスクでブートして［修復セットアップ］を行います。また、OS/2では、コマンドラインから“FDISK /NEWMBR”を実行します。

日本語

Q 日本語ディストリビューションって、どこが「日本語」なんですか？

A 日本語化が必要になる要素にはさまざまあります。主なものには、次のようなものがあります。

- **マルチバイト対応のライブラリ**
- **インストーラ**
- **マルチバイト対応コマンド**
- **アプリケーション**
- **ドキュメント（manも含む）**
- **環境定義ファイルの用意**

Slackwareが主流だったころは、英語環境が標準で、インストーラも含めてすべて英語表示されたもので一度環境を構築しておいて、そのうえに日本語の入力／表示ができるアプリケーションをつけ足すという作業を行っていました（これらの日本語対応パッケージは、初期にはJE、のちにPJEと呼ばれていました）。しかし、現在日本語ディストリビューションと呼ばれるもののほとんどは、これらの要素を満たしています。Vine Linuxにいたっては、上記に加えて、さらにコマンド出力まで日本語化するという徹底した日本語対応をウリにしています（**画面10**）。

なお、サーバ専用という用途で使用する場合は、これらの項目は必ずしも必要になるわけではないので（マシンリソースを考慮すると、むしろないほういい場合もある）、英語版のディストリビューションを利用しているユーザーも多いようです。

Q 日本語のファイル名は使えますか？

A Windowsに慣れたユーザーには、日本語のファイル名を利用したいという方も多いでしょう。結論からいうと、「使えるが、使わないほうがいい」という回答になります。日本語ファイルそのものは、cat、cp、mvといった各種コマンドは対応しており、問題なく利用することができます。また、bashならファイル名補完も行うことができます。以前は、うまく表示できなかったlsコマンドも、最近のGNU fileutilsのものがインストールされていれば問題なく表示できます（**画面11**）。しかし、GUIでファイルを扱うことが多いWindowsと異なり、CUIでの作業が多いLinuxでは、ファイル名入力のたびごとにかな漢字変換システムオン／オフするのはかなり煩雑な作業で、あまりお勧めはできません。

Q コンソールで日本語が化けてるんですが

A コンソールが英語モードになっているために日本語が表示できない場合には、kon（漢字コンソールエミュレータ）コマンドを実行します。

また、環境変数LANGが設定されていないために正しく表示できないこともあります。その場合には“export LANG=ja_JP.ujis”（LANG=jaが正しいディストリビューションもある）とします。逆に英語表示にするなら“export LANG=C”とします。

表示が正しくできるようになったら、.bash_profileにその設定を記述しておけば次回からも有効になります。

```
kterm
[root@localhost /root]# ipconfig
bash: ipconfig: command not found
[root@localhost /root]# ifconfig
lo        リンク方法:ローカルループバック
          inetアドレス:127.0.0.1マスク:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RXパケット:0 エラー:0 損失:0 オーバラン:0 フレーム:0
          TXパケット:0 エラー:0 損失:0 オーバラン:0 キャリア:0
          衝突(Collisions):0

[root@localhost /root]# ping
使用法: ping [-LRdfnqrv] [-c カウント] [-i ウェイト] [-l preload]
        [-p パターン] [-s パケットサイズ] [-t ttl] [-I インターフェースアドレス]
 ホスト名
[root@localhost /root]# cp
cp: 引数が間違っています
詳しくは 'cp --help' を実行して下さい。
[root@localhost /root]# w
 12時46分(午後)  稼働   7時24分,  2 ユーザ,  負荷平均率: 0.00, 0.04, 0.07
ユーザー 端末     場所           Login時刻   待機   JCPU   PCPU  現プロセス名
root     tty1                     5:23am  6:08m  1:30   0.07s  sh /usr/X11R6/b
root     ttyp0    :0.0           12:41pm  0.00s  0.78s  0.09s  w
[root@localhost /root]#
```

画面10　日本語化されているVine Linuxのメッセージ

```
kterm
[mitsug-k@latpc08 mitsug-k]$ ls
[mitsug-k@latpc08 japanese]$ cat >締め切りの最後通告が来たよ.txt
だけど、無理なものは無理。
海外逃亡かな？

[mitsug-k@latpc08 japanese]$ ls 締め*
締め切りの最後通告が来たよ.txt
[mitsug-k@latpc08 japanese]$ ls -la
total 3
drwxrwxr-x   2 mitsug-k mitsug-k     1024 Dec 18 12:57 ./
drwxrwxr-x   8 mitsug-k mitsug-k     1024 Dec 18 12:55 ../
-rw-rw-r--   1 mitsug-k mitsug-k       43 Dec 18 12:59 締め切りの最後通告が来たよ.txt
[mitsug-k@latpc08 japanese]$ mv 締め切りの最後通告が来たよ.txt さようなら.txt
[mitsug-k@latpc08 japanese]$ ls -la
total 3
drwxrwxr-x   2 mitsug-k mitsug-k     1024 Dec 18 13:00 ./
drwxrwxr-x   8 mitsug-k mitsug-k     1024 Dec 18 12:55 ../
-rw-rw-r--   1 mitsug-k mitsug-k       43 Dec 18 12:59 さようなら.txt
[mitsug-k@latpc08 japanese]$ rm さようなら.txt
[mitsug-k@latpc08 japanese]$ ls
[mitsug-k@latpc08 japanese]$ cat > あ連R浴
```

画面11　日本語のファイル名も問題なく使える（TurboLinux 4.2）

Windowsからtelnetで接続したときに文字化けする場合には、漢字コードにEUCを使うようにtelnet（Tera Termなど）コマンドを設定します。

Q X上のターミナルウィンドウで日本語が表示できない

A ktermを使っていますか？xtermでは日本語は表示できません。ktermを起動するには、コマンドラインからktermと入力するか、日本語化されたGNOMEのメニューなどから［ユーティリティ］—［日本端末］を選びます。ファイルマネージャで/usr/X11R6/binディレクトリを表示し、ktermのアイコンをマウスでドラッグ＆ドロップすれば、デスクトップや下段のパネル上にショートカットを作ることができるので、作成しておけば便利に使えるでしょう。

Q それでも漢字が文字化けして読めません

A 元のファイルの漢字コードが日本語EUCでないことが考えられます。Windowsで作られた日本語のテキストはシフトJIS（MS漢字コード）になっています。またメールの本文にはISO2022-JP（JIS）コードが使われています。それぞれ違うコード体系なので、それらのファイルを単に、catなどで表示すると文字化けしてしまいます。

Q シフトJISのファイルをEUCに変換したいのですが

A nkf（ネットワーク漢字コード変換フィルタ）という漢字コード変換コマンドがありますので、それを使ってみましょう。たとえば、シフトJISで書かれたsample.txtというファイルを日本語EUCにするには、

```
% nkf -e  sample.txt
```

と行います。nkfは標準入力から受け取って、変換した結果を標準出力にアウトプットできますので、パイプ（|）を使ってほかのプログラムとの入出力をつないでいくことができます。

nkfコマンドは**表3**に示したオプションをつけることで、各種のコード変換を行うことができます。

Q 日本語のmanが表示されない

A LASER5 Linux 6.0などでは、日本語のmanページを表示するためのプログラムとしてjmanが用意されていますので、“ jman ＜コマンド名＞ ”と実行します。

jmanがないディストリビューションでは、環境変数LANGの設定を確認しましょう。“ env ｜grep LANG ”と行って“ LNAG=jp_JP.ujis ”（LANG=jaが正しいディストリビューションもある）が表示されないなら、“ export LANG=ja_JP.ujis ”を設定します。

なお、それでも日本語表示されない場合は、日本語のmanページがインストールされていない可能性もあるので、/usr/manディレクトリに、ja_JP.ujisディレクトリがあるかを確認します。

Q 日本語の入力をするには？

A ほとんどの日本語ディストリビューションにはフリーのかな漢字変換システム（CannaやWnn4）、Xのインプットメソッド（kinput2）がバンドルされており、しかもすぐに入力ができるように設定されています。したがって、X Windows Systemでcannaを使って日本語を入力するには、Shift＋Spaceキーを同時に押すことで、かな漢字変換モードに入ります。入力した単語はスペースキーで変換／次候補、Enterキーで確定します。

Shift＋Spaceキーを押しても変換モードにならない場合には、“ ps ax ”で表示される実行中のプロセスをチェックします。jserver、cannaserver、kinput2といったプログラムが起動されていないか、日本語入力のできない端末（xtermなど）で利用しようといる可能性がありますので、確認してみてください。

Q フリーで利用できるLinuxのかな漢字変換システムにはどのようなものがありますか？

A フリーで利用できるものにはWnn4、Canna、SKK、sj3、T-Codeなどがあります。これらの中でも特に人気の高い、Wnn4、Canna、SKKについて、それぞれの特徴を紹介します。

・Wnn4

Wnn（うんぬ）は、京都大学、オムロン、アステックによって共同開発された、クライアント／サーバ方式のかな漢字変換システムです。Wnnという名前が、「Watashino Namaeha Nakano desu（私の名前は中野です）」を一発で変換できるシステムということから、この文の文節頭のアルファベットをとって名づけられたという話は有名です。かな漢字変換サーバは「jserver」が担当し、フロントエンドプロセッサ「uum」、Xのインプットメソッド

「kinput2」「Xwnmo」、マルチリンガルエディタ「mule」などから利用します。Wnn4は、1989年にリリースされたもので、各種ヨーロッパ言語、日本語、中国語（簡体字、繁体字）、韓国語に対応した変換サーバ（日本語変換サーバ部分はjserver）と各言語用の辞書が含まれており、マルチリンガル入力システムという特徴を持っています。なお、Wnn4.2をGPLに即した形に再整備したFreeWnn 1.0が、FreeWnnプロジェクト（http://www.freewnn.org/）によって1999年4月に公開されています。

オプション	説明
-j	JISコードを出力する（デフォルト）
-e	EUCコードを出力する
-s	シフトJISコードを出力する
-m	MIME を解読する。ISO-2022-JP(base64)とISO-8859-1(Q encode)のみを解読する。ISO-8859-1 (Latin-1) を解読するときは、-lフラグも必要である -mB　MIME base64 stream を解読する。ヘッダなどは取り除くこと -mQ　MIME quoted stream を解読する
-l	0x80-0xfeのコードをISO-8859-1 (Latin-1)として扱う JISコードアウトプットとの組合せみのみ有効 -s, -e, -xとは両立しない
-f?	一行?文字になるように簡単な整形を行う。デフォルトは60文字である
-J -E -S -X -B	期待される入力コードの性質を指定する -J　ISO-2022-JPを仮定する -E　日本語EUC(AT&T)を仮定する -S　シフトJISを仮定する。X0201仮名も仮定される -X　シフトJIS中にX0201仮名があると仮定する -B　壊れた(Broken)JISコード。ESCがなくなったと仮定する -B1 ESC-(, ESC-$ のあとのコードを問わない -B2 改行のあとに強制的にASCIIに戻す

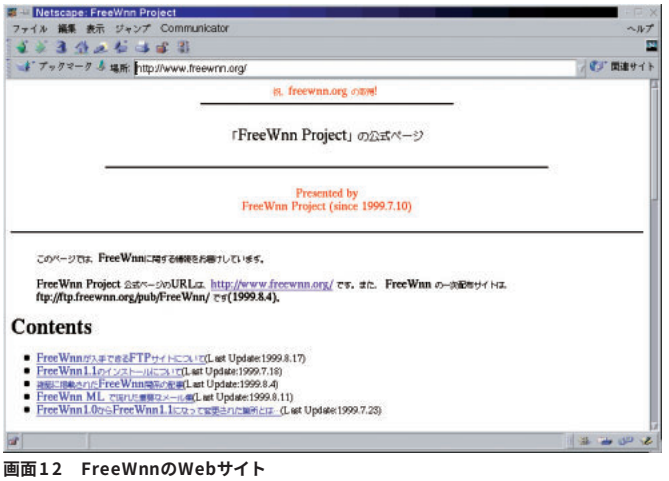
表3　nkfコマンドの主要なオプション

・Canna

Canna（かんな）は、NECによって開発されたクライアント／サーバ方式のかな漢字変換システムです（http://www.nec.co.jp/japanese/product/computer/soft/canna/）。かな漢字変換サーバは「cannaserver」が担当し、フロントエンドプロセッサ「canuum」、Xのインプットメソッド「kinput2」、マルチリンガルエディタ「mule」などから利用します。どのような環境からでも統一的なインターフェイスによって入力ができるという特徴を持っています。また、カスタマイズファイルである.cannaも一元化されています。

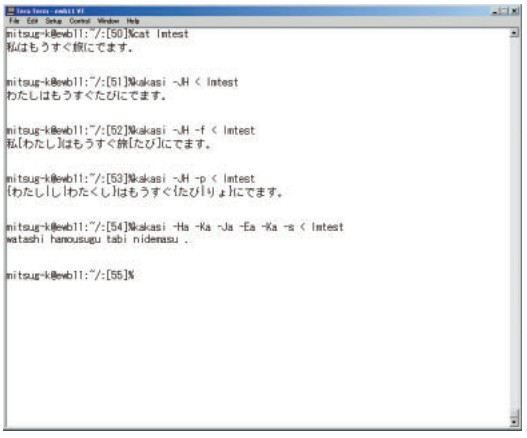
・SKK

SKKは、現京都大学の佐藤雅彦氏によって開発されたかな漢字変換システムです（http://skk.kuis.kyotou.ac.jp/skk/）。Emacsのマクロ言語「Emacs Lisp」だけで記述されており、Emacs系エディタから利用することが可能です。SKKの名前は、「Simple Kana Kanji conversion program」の頭文字をとって名づけられたもので、GPLに則ったフリーソフトウェアです。simpleの名前のとおり、文法解析をいっさい行わず、漢字変換部分や送り仮名の始まりを、ユーザーが大文字で入力することにより指定するという、一風変わった特徴を持っています。一見、面倒な印象を持つかもしれませんが、ユーザー自身が変換を完全にコントロールできるため、ストレスが少なく、根強い人気があります。開発当初は、EmacsLispだけで記述されたものでしたが、現在は共有辞書にアクセスするサーバ「skkserv」も作られており、こちらも利用できるようになっていま



画面12　FreeWnnのWebサイト

画面13　CannaのWebサイト



画面16　kakasiの実行例
各種オプションの指定により、ひらがなへ（-JH）、ふりがなつき（-JH -p）、複数読みの表示（-JH -p）、ローマ字（-Ha -Ka -Ja -Ea -ka -s）など、いろいろな変換が行える。

す。また、フロントエンドプロセッサ「skkfep」やXのインプットメソッド「skkinput」なども開発が進められています。

Q LinuxにはWindows並の変換精度のかな漢字変換システムはないのですか？

A かな漢字変換システムの変換精度は、とても開発に手間がかかるものであり、フリーで開発されることが多いUNIXのかな漢字システムは、メーカーが開発している製品ほどの変換精度がなく、UNIXの弱点といわれていました。しかし、状況は好転し、次に述べるような商用の変換システムがLinuxにバンドルされるようになっており、変換精度もWindowsに引けをとらないレベルに達しています。

・ATOK12 SE for Linux

ATOK12 SE for Linuxは、ジャストシステム社によって開発されているかな漢字変換システムで、WindowsやMacintosh環境で人気の高かったATOK（Advanced Technology of Kana Kanji Transfer）のLinux版です。ただし、単体販売されずOEM提供だけとなっているので、利用するには、バンドルされている TurboLinux 4.x、公認 Red Hat Linux 6.1、LASER5 Linux 6.0、OpenLinux 2.3日本語版などを購入するしかありません。

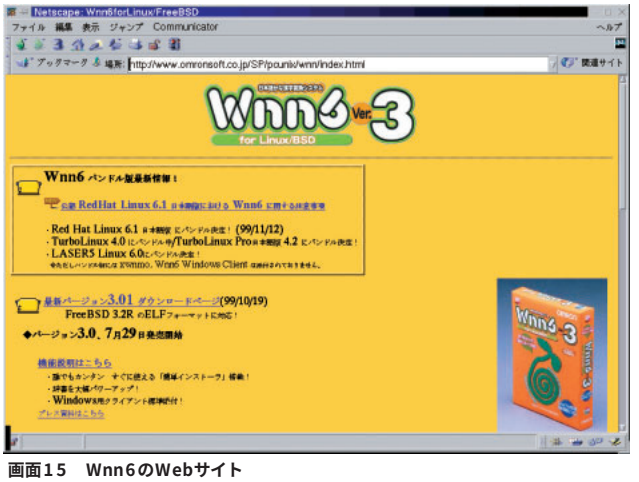
ATOK12 SE for Linuxは、WindowsなどのATOK12に比べ、省入力のため機能がないなど、いくつかの機能が限定されているものの、変換効率や入力校正機能（「ら」抜き言葉の指摘など）はそのまま利用できます。

・Wnn6 for Linux

Wnn6 for Linuxは、オムロンソフトウェア社によって販売されているクライアント／サーバ方式の商用かな漢字システムです（http://www.omronsoft.co.jp/SP/pcunix/wnn/index.html）。最新バージョンは3.0。フリーで配布されていたWnn4からさらに変換効率やカスタマイズ機能が強化されています。具体的には、Wnn4に文節区切りを判断させるアルゴリズムを追加して、文節区切りの間違いによる誤変換を減らしたことや、「FI関係辞書」という用例を記述した辞書によって、同音異義語の選択を最適化した「FI変換」と呼ばれる機能、さらに、ユーザーの確定情報を学習しておく「FI学習」という機能が追加されたことにより、変換効率がWnn4よりも飛躍的によくなっています。

個人向けのパッケージは、「1ユーザー、同時に5セッションまでの接続」というライセンスで、9800円で単体販売されています。また、TurboLinux 4.x、公認 Red Hat Linux 6.1、LASER5 Linux 6.0、OpenLinux 2.3日本語版、Vine Linux 1.1CRW、Linux MLD4といった数多くのディストリビューションの製品版にバンドルされています。

画面14　ATOK12 SE for LinuxのWebサイト



画面15　Wnn6のWebサイト

Q 日本語に関連した便利なツールを教えてください

A 漢字かな混じり文をひらがな／カタカナ／ローマ字のテキストに変換するKAKASI（KAnji KAna Simple Inverter）というソフトウェアがあります。テキストファイル中に読めない漢字がある場合や、漢字にふりがなをつける場合などに便利に利用できるでしょう（**画面16**）。

また、日本語テキストを全文検索するシステム「Namazu」があります。Namazuは、Vine LinuxやPlamo Linuxのドキュメント検索システムに利用されており、個人ベースのデータ管理に利用すれば、より快適な環境を構築できるでしょう。

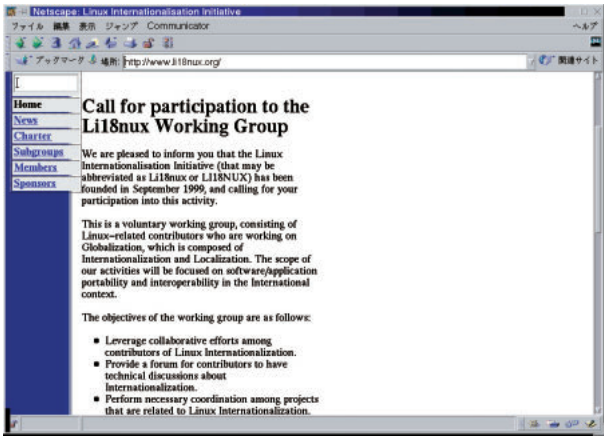
Q 日本語ロケールってなんですか？

A ロケール（locale）とは言語環境を指定するもので、プログラムはあらかじめ用意されているロケールデータベースを参照することによって、その国の言語や文化に合わせた動作をすることができます。漢字などの文字コードの違いはもちろんですが、プログラム内部では単純な数値として計算したものを、表示する時にその国に合わせて出力することもあります。たとえば、通貨は“US$99.80”や“￥9,980”のように表示したり、日付は“20 December, 1999”や“平成11年12月20日”のように表記したりすることがあります。

日本語に対応したLinuxでは通常、環境変数LANGやLC_ALLに“ja_JP.ujis”（日本語EUC）を指定します。しかし「Linux における日本語ロケールに関する指針（1999年12月15日）」の中で、“ja_JP.eucJP”（日本語EUC）を使うことが推奨されましたので、今後は“ja_JP.ujis”の代わりに“ja_JP.eucJP”が使われるようになるでしょう。

よく使われるロケールには、**表4**のようなものがあります。

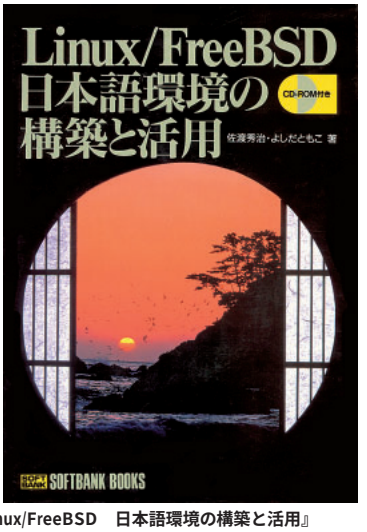
画面17　Li18nuxのWebサイト
Li18nuxは、Linuxを始めとするオープンソースのソフトウェアの国際化に関連するプロジェクトの協業を図る団体。ディストリビューションごとに日本語への対応を行うのではなく、国際化されたAPIを推奨し、ディストリビューション間のソフトウェアの可搬性や相互運用性の確保を目指す。



Q I18Nとはなんですか？

A Internationalization（国際化）の先頭と末尾のアルファベットの間の文字数を数字で表記したもので、システムやアプリケーションなどを国際化することを表します。ここでいう国際化というのは、「複数の国や地域の言語や習慣に対応させること」を指します。またL10Nは、「Localization」を意味し、「ある特定の国や地域の言語や習慣に対応させること」を指します。さらにM17Nは、「Multilingualization）を意味し、「国際化を言語だけの観点からとらえた」ものです。I18NとM17Nは、よく混同している人もいるようですので、注意してください。

お勧めの書籍



『Linux/FreeBSD　日本語環境の構築と活用』
佐渡秀治／よしだともこ著
ソフトバンク刊／2600円／ISBN4-7973-0480-4

ロケール	言語	地域	コード名	備考
ja_JP.ujis	日本語	日本	日本語EUC	Linuxでよく使われる
ja_JP.eucJP	日本語	日本	日本語EUC	
ja_JP.SJIS	日本語	日本	シフトJIS	
ja_JP.JIS7	日本語	日本	JIS	
C	英語	アメリカ	ISO 8859-1	Linuxでよく使われる
en_US.ISO8859-1	英語	アメリカ	ISO 8859-1	
en_GB.ISO8859-1	英語	イギリス	ISO 8859-1	
fr_FR.ISO8859-1	フランス語	フランス	ISO 8859-1	
fr_CA.ISO8859-1	フランス語	カナダ	ISO 8859-1	
de_DE.ISO8859-1	ドイツ語	ドイツ	ISO 8859-1	

表4　よく使われるロケール

X Window System

Q X Window Systemとはなんですか？

A X Window System（もしくはX）は、UNIXでデファクトスタンダードとなっているウィンドウシステムのことです。元々、MIT（マサチューセッツ工科大学）とDECによって始められたAthenaプロジェクトよって開発されたもので、参考にした「W」という名のウィンドウシステムの次という意味から「X」という名がつけられています。最新のバージョンはX11R6.4で、このバージョンよりThe Open Groupが開発し、X.Orgが管理を行っています。ちなみに、「Xという名前のウィンドウシステム」なので、X Windowとは呼びません。

Xの特徴は数多くあり、詳細は専門書に譲りますが、「クライアント／サーバシステム」とすることで柔軟なシステム構成となっていること、ネットワーク経由の接続を可能にした「ネットワーク透過性」などが挙げられます（**図3**）。

Q XFree86とX Window Systemはどう違うのですか？

A 元々XFree86は、X Window Systemをi386系のCPU上で動作するUNIX用に移植したものです。したがって、XFree86は「X Window Systemの一形態」ということができるでしょう。ただし、以前のXFree86はX Window Systemの追加コードという形で開発されていましたが、X11R6からはソースコードが統合され、XFree86もX11のソースコード一式を持つようになっており、現在は追加コードという位置づけではなくなっています。最新バージョンは、X11R6.3をベースにしたXFree86 3.3.5ですが、それとは別にX11R6.4をベースにしたXFree86 4.xの開発も続けられています。XFree86 4.xは、デバイス依存部を動的に組み込む「ローダブルサーバ」や複数のディスプレイへの描画を可能にした「マルチヘッド対応」など、さまざまな試みがなされており、リリースが期待されています。

Q X-TTというのもあるようなのですが

A X-TTは、TrueTypeフォントのレンダリングエンジンであるFreeTypeを利用して、TrueTypeフォントを扱えるようにしたXサーバのことです。このX-TTを利用す

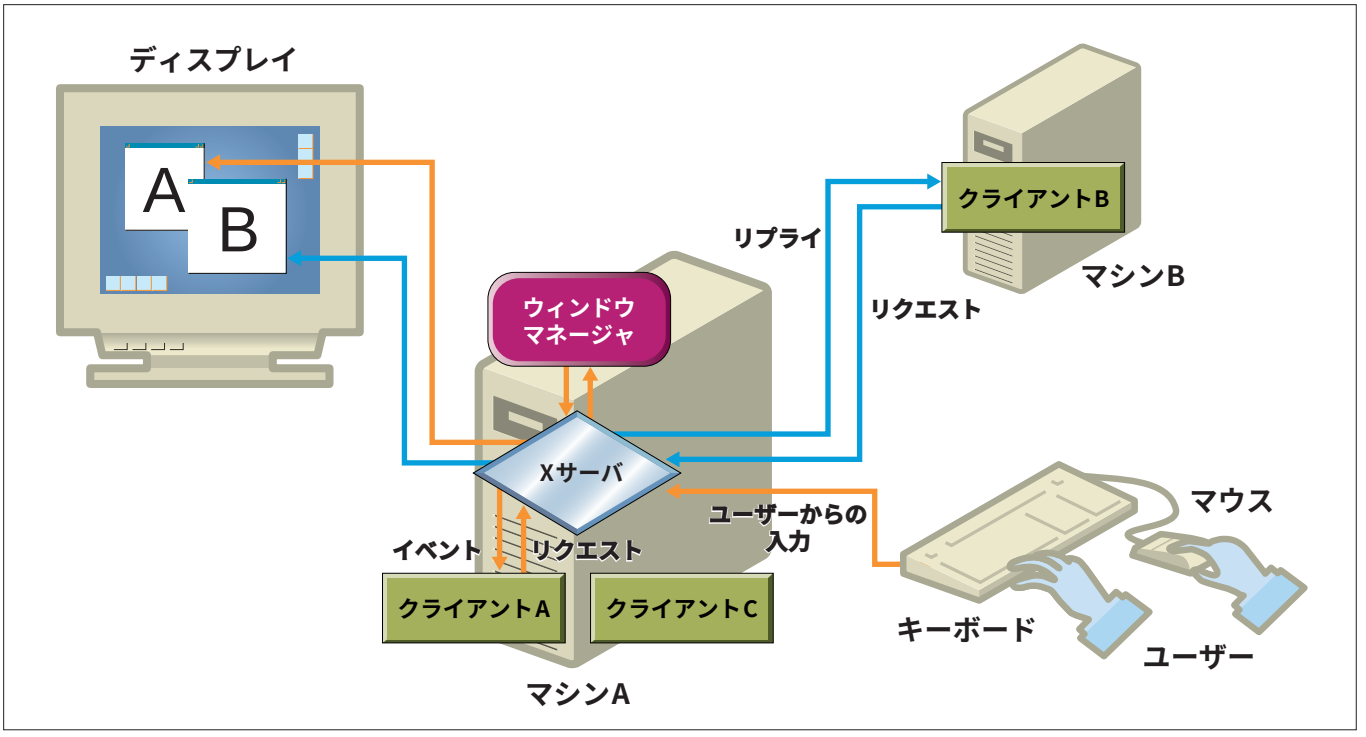


図3　クライアント／サーバモデルのX Window System

ると、これまでガタガタのフォントでしか表示できなかったNetscape Navigatorの表示をきれいにすることができます。最近のほとんどのディストリビューションがX-TTをサポートしていますが、万一使用しているディストリビューションに含まれるXFree86がX-TTに未対応のものであっても、Xフォントサーバを利用すれば同様の効果を得ることができます。なお、現在開発中のXFree86 4.0では、このTrueTypeフォントのサポートが標準で組み込まれる予定です。X-TTに関する詳細な情報は、http://x-tt.dsl.gr.jp/index-ja.htmlを参照してください。

Q ウィンドウマネージャとは？

A ウィンドウマネージャは、ウィンドウの移動、サイズ変更、装飾などを受け持っている特殊なXクライアントのことです。特殊といわれるのは、1つのXサーバ上に1つしか動作しないことや、ユーザー操作に応じてほかのウィンドウの管理を行う点が、ほかのXクライアントと異なるからです。このウィンドウマネージャを変更することによって、さまざまな見栄えや操作感（Look & Feelといいます）を変更することができます。そのため、好みに応じてたくさんのウィンドウマネージャが作られています。ウィンドウマネージャには、X11に標準でバンドルされているtwm、柔軟なカスタマイズが可能なfvwm2、Windows 95ライクなqvwm、NEXTSTEPのLook & Feelを追求したAfterStep、GUI設定ツールを装備したWindow Maker、テーマ機能が充実したEnlightenmentなど数多くの種類があります（**画面18**）。

Q ウィンドウマネージャを変更したいのですが

A Xの起動スクリプト「startx」を実行すると、/usr/X11R6/lib/X11/xinit/xinitrcが実行されます。ディストリビューションによっては、このxinitrcの中でさらにさまざまな条件分岐がなされているものもありますが、大まかにいってこのような流れになっています。ということは、xinitrcを書き換えれば目的は達成できるのですが、これではそのLinuxシステムを利用しているユーザー全員のウィンドウマネージャが変更されてしまいます。startxコマンドの中を見ると、$HOME/.xinitrcが用意されている場合は、それが優先されるようになっていますので、/etc/X11/xinit/xinitrcを自分のホームディレクトリにコピーして、それを参考にカスタマイズすればよいでしょう。ただし、ウィンドウマネージャを最後に起動するようにしておかないと.xinitrcの実行が終わると、X Windows Systemが終了してしまうことがあるので注意が必要です。

なお、TurboLinux 4.xを使っているなら、turbowmcfgというウィンドウマネージャ変更用のツールが用意されているので、これを利用するのが簡単でしょう。このturbowmcfgは、rootユーザーが--systemオプションつきで起動し、ウィンドウマネージャを選択すると、ログインする際のディスプレイマネージャもいっしょに変更してくれます。

Q KDEやGNOMEはウィンドウマネージャじゃないの？

A 結論からいうと、KDE（**画面19**）やGNOMEはウィンドウマネージャではありません。というよりも、ウィンドウマネージャを土台に、アプリケーションや各種設定ツールま

画面18　いろいろなウィンドウマネージャ
左：twm、中：AfterStep、右：Window Maker

で提供し、統一的な環境を提供することを目的としたものです。そのため、「統合デスクトップ環境」と呼ばれることもあります。ちなみに、KDEのウィンドウマネージャはkwmであり、GNOMEのウィンドウマネージャはEnlightenmentやWindow Makerなど、GNOME対応のウィンドウマネージャなら特定はしません。

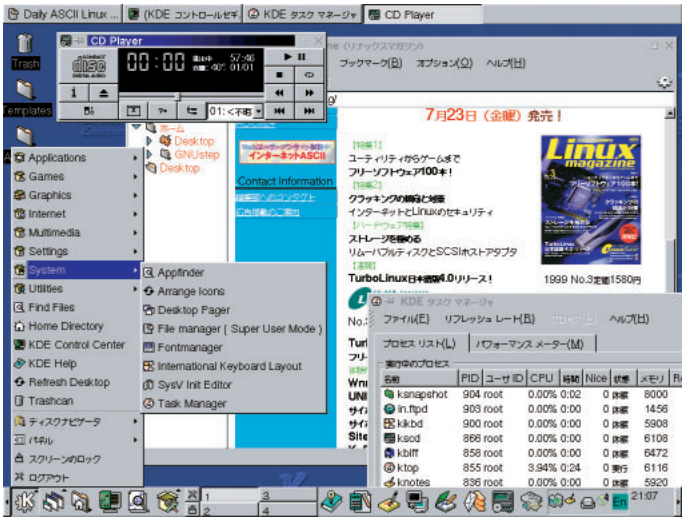
Q MotifやQtとかGTK+ってなんですか？

A 説明の前に、まずXのアプリケーションプログラム（Xクライアント）の仕組みを簡単に説明しましょう。Xクライアントは、**図4**に示すような構成となっています。最下層にあるのが、ウィンドウの生成や消滅、基本図形の描画や各種情報の管理、イベントの管理などを行っている「Xlib」です。このXlibを利用すれば、Xクライアントを作成することは可能ですが、Xlibはあまりにも基本的な機能しか持っていないため、ボタンひとつ描画するにもとても手間がかかってしまいます。そこで、Xlibの上によりユーザーが使いやすい形で利用可能な上位ライブラリが用意されています。これが、「ツールキット」です。

「ツールキット」は、さらにツールキットの核となる機能を提供する「イントリンシクス」（Intrinsics）と、それにかぶせて「見た目」を提供する部品の集まりである「ウィジェットライブラリ」（Widget Library）の2つに分けることができます。Xでは、Xlibまでを標準機能と定義しており、ツールキットは自由に選択できるようになっています。これは、アプリケーションの見せ方は定義しない、というXの設計思想が反映された結果です。

Motifは、OSF（Open Software Foundation）によって提供されているツールキット（正確にはウィジェットライブラリ）で、有料です。SolarisやHP-UXといった商用UNIXのデスクトップ環境として利用されているCDEは、このMotifをベースに作られています。

GTK+（GIMP Tool Kit+）は、元々高機能の画像処理ソフトウェア「Gimp」を記述する目的で設計されたツールキットで、Motifに似たLook & Feelを提供しています。GTK+は、LGPLにしたがって配布されているフリーなソフトウェアで、統合デスクトップ環境「GNOME」でも利用されています。



画面19 KDE

Qtは、ノルウェーのTroll Tech社によって開発されているツールキットで、Windowsともソースレベルの互換性を持つという特徴があります。以前のバージョンのQtは商用で開発を行う際は有料となっていましたが、Qt-2.0からはQPLという新しいライセンスになり、実質的にフリーなソフトウェアといえるでしょう。Qtは、統合デスクトップ環境「KDE」やOpenLinuxのインストーラ「LIZARD」などで利用されています。

Q リモートマシンからローカルマシンにXアプリケーションを表示させたいのですが、どうしたらいいですか？

A まずは、ローカルマシン（local）のXサーバに他のマシン（remote）からの接続を許可する設定をします。

```
local$ xhost remote
```

さらに細かい制御を行いたい場合はxauthコマンドがありますので、詳細はマニュアルを参照してください。次に、リモートマシンに描画させたいマシンの情報を教えます。具体的には、次の手順で環境変数DISPLAYを設定します。

```
remote$ export DISPLAY=local:0.0
```

0.0は、複数のXサーバと複数のスクリーンを特定するための番号ですが、ほとんどの場合このままで問題はありません。あとは、表示させたいアプリケーションをリモートマシン側で起動させるだけで、ローカルマシンに表示されるようになります。なお、シェルスクリプトxonが利用できれば、環境変数の指定がいらず便利に利用できます。

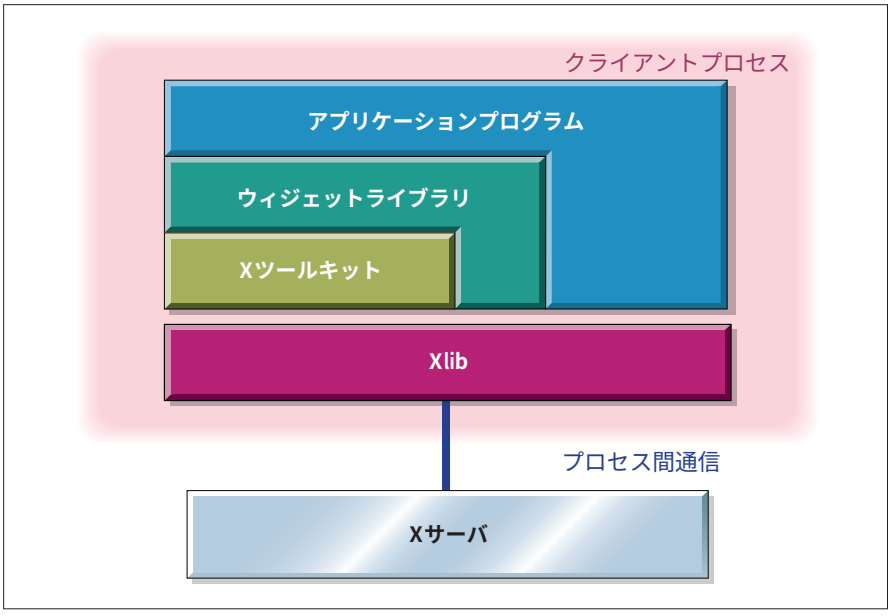


図4　Xクライアントのソフトウェア構成

Q X Window Systemが反応しなくなりました。リブートするしかないでしょうか?

A Ctrl＋Alt＋BackSpaceキーを押せば、X Window Systemは強制終了されます。

Q テキストログインをグラフィカルログインに変更したいのですが

A ランレベルでこのログイン方法が決まっています（現在のランレベルはrunlevelコマンドでわかります）。ディストリビューションによっても異なりますが、多くのディストリビューションは、テキストログインの場合はランレベルが3になっているはずです。そこで、rootユーザーになってから、標準のランレベルを指定してある/etc/xinitrcをエディタで開いて、

```
id:3:initdefault:
```

という行の“3”を“5”に変更して、rebootすると、次回からグラフィカルログインとなるはずです。また、一時的にログイン方法を変更するだけなら、起動時のliloプロンプトから

```
LILO Boot: linux 5
```

として、引数にランレベルの5を指定すれば、グラフィカルログインとなります。

Q グラフィカルログインから抜けたいのですが

A ログインのモードをグラフィカルログイン（xdm）を起動しているときは、何度X Window Systemを終了させても、グラフィカルログインの画面が出てきて、テキストログインの画面に移ることができません。こんなときはCtrl＋Rキーを押すと、グラフィカルログインを終了して、テキストログインの画面になります。しかし、GNOMEのディスプレイマネージャ「gdm」を使用しているときは、Ctrl＋Rキーを押してもgdmから抜けることができないことがあります。そのときは、

1.再起動させ、LILOプロンプトに“linux 3”と入力する
2.inittabを編集し、ランレベルを“5”から“3”へ変更する
3.仮想コンソールを利用する

などがあります。1.は一時的なログインモードの変更に、2.は定常的なログインモードの変更に、3.は一時的にコンソールを利用する際に使用します。それぞれ目的に応じて使い分けてください。

Q Xconfiguratorを実行するとブラックアウトしてしまいます

A Red Hat Linux 6.0ではXconfiguratorで設定に失敗すると、マシンを起動してもなにも表示されず、再びXconfiguratorを実行しても設定を行うことができません。また、Ctrl+Rでグラフィカルログインから抜けることもできません。この場合は、別のマシンからtelnetで接続し、そのマシンをshutdownします。そして、前述の方法でシステム起動方法をテキストログインに変更し、コマンドラインベースでX環境を設定できるxf86configコマンドで設定します。

Q Xでスクリーンショットをとりたいのですが

A これは使用しているディストリビューションやウィンドウマネージャなどにも左右されるので一概にはいえませんが、xv、Gimp、ksnapshotなどで全画面、ウィンドウ単位などで画面をとることが可能です。また、Xに標準添付されているxwdコマンドを利用すれば、コマンドラインからも画面をとることができます。

ネットワーク

Q ネットワークカードが認識されない

A まず、ドライバがちゃんと組み込まれているのか、ifconfigコマンドで確認しましょう。正常に動作している場合は**画面20**のようになります。しかし「eth0」の行から7行が表示されずに、「lo」の行から始まっている場合には、ネットワーク機能は正常に動作していません。

PCIバスの10BASE-T／100BASE-TX対応ネットワークカードの場合、普通はインストーラの自動検出によって認識されますが、もしも自動認識されなかった場合には、手動での設定が必要になります。

ディストリビューションをインストールした直後のカーネルでは、実装されているハードウェアを自動的に検出し、モジュールごとにドライバを組み込んでいます。手動でモジュールを読み込むには、

```
# modprobe <ドライバ名>
```

という書式で、ドライバを設定します。I/Oアドレスや10BASE-T／100BASE-TXの切り替えの指示は、ドライバ名のあとに“eth0=＜パラメータ＞”として指定します。

また、ISAバスのネットワークカードでは、割り込みやDMAアドレスがほかのカードと衝突してしまい動作していない可能性があります。その場合にはpnpdumpコマンドで状況を調べ、isapnpコマンドでカードの使用アドレスを変更するなどして、衝突を回避する必要があります。

Q IPアドレス、BOOTP/DHCPのどちらを選べばいいの？

A BOOTP、DHCPは、ともにIPアドレスを動的に割り当てる仕組みのことで、利用するにはLinuxをつなぐ、BOOTP／DHCPサーバが必要です。

DHCPサーバがない場合は、IPアドレスを直接指定します。専用線などで常時インターネットに接続する場合を除き、自由に使えるIPアドレスである192.168.0.0～192.168.255.255（プライベートアドレスという）の中から適当な値を指定すればよいでしょう。

Q pingを実行しても相手のホストから返事がありません

A pingは、ホストが稼働しているかを確認するためのコマンドです。

まず自分のハードウェアがきちんと動作しているのか、ifconfigコマンドで確認します（**画面20**）。特に2行目に注目しましょう。“inet addr:”がIPアドレス、“Bcast:”がブロードキャスト、“Mask:”がネットワークマスクを表しています。

次に、どこに原因があるのかを調べるため、近くのネットワークから順番にpingコマンドで試験していきます。最初はホスト名を指定せずにIPアドレスを直接入力します。

(1) 同一セグメント内のホスト
(2) GATEWAYアドレス
(3) 別セグメントのホスト

IPアドレスでpingコマンドを実行してみて返事があるようならば、次にホスト名を指定しましょう。ホスト名を指定するとダメなようならDNSの名前解決ができていないと考えられますの

```
# /sbin/ifconfig
eth0      Link encap:Ethernet  HWaddr 00:48:54:3A:FC:76
          inet addr:192.168.1.10  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:130 errors:0 dropped:0 overruns:0 frame:0
          TX packets:331 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:10 Base address:0xa800

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:860 errors:0 dropped:0 overruns:0 frame:0
          TX packets:860 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

画面20　/sbin/ifconfigの結果

```
# nslookup www.ascii.co.jp
Server:  <DNSサーバ名>
Address: <DNSサーバIPアドレス>

Name:    at2.ascii.co.jp
Address:  210.140.231.23
Aliases:  www.ascii.co.jp
```

画面21　nslookupでDNSサービスの動作を調べる

で、ネームサーバに問い合わせするコマンドで確認します。nslookup ＜ホスト名＞を実行して正しいIPアドレスに設定されているか調べます（**画面21**）。

Q 「unkonwn host」となってしまうのですが

A ホスト名からIPアドレスへの変換ができていない可能性があります。IPアドレスがわかっている場合には、IPアドレスを直接指定してみるとうまくいくことがあります。

```
# nslookup <ホスト名>
```

と行ってみてください（**画面21**）。

最初のServerが正しく表示されない場合には、DNSサーバに接続できていません。/etc/resolv.confファイルの「nameserver」行にDNSサーバが登録されていますので、内容を確認してください。

「can't find ＜ホスト名＞: Non-existent host/domain」と表示された場合は、そのホスト名はDNSに登録されていません。/etc/hostsファイルに、ホスト名とIPアドレスを登録すれば解決します。

nslookupコマンドでは、ホスト名の代わりにIPアドレスを指定すると、DNSの逆引きが行われ、ホスト名が表示されます。

Q ネットワークカードを2枚挿していますが、1枚しか認識されません。

A ネットワークカードのドライバは、モジュールとして組み込む場合と、カーネルに組み込む場合の2通りがあります。モジュール組み込みの時には、/etc/conf.modulesファイルに1枚目（eth0）のカードの設定が「alias eth0 ＜モジュール名＞」のように書いてあるはずです。

テストのために、2枚目のカードを、

```
# modprobe eth1 <モジュール名>
```

と行って手動で組み込んでみましょう。正しくロードされたかは、lsmodコマンドでモジュール名が表示されたことで確認できます。この時のモジュール名は、2枚目のネットワークカードに対応するドライバ名を指定します。

次に、linuxconfプログラムなどでeth1のネットワークドライバを設定し、ifconfigコマンドで、eth0とeth1の設定を確認します。うまく動作するようなら、エディタで2枚目のカードの設定を/etc/conf.modulesファイルに、

```
alias eth1 <モジュール名>」
```

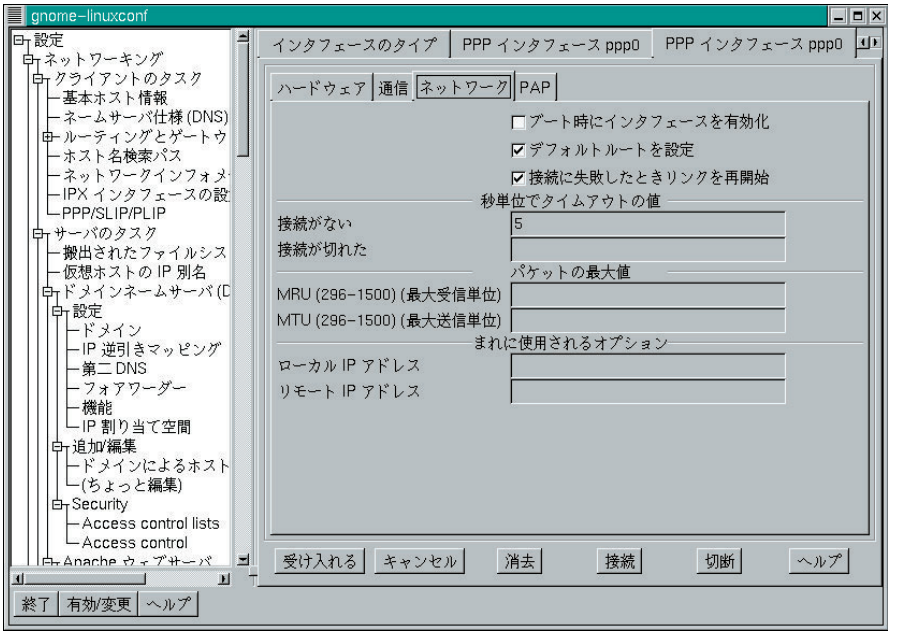
のように書き加えます。

カーネルに組み込む場合は、/usr/src/linuxディレクトリで、make xconfig（コンソール用はmake menuconfig）と行って、使用しているネットワークカードの項目を「Y」に設定し、カーネルを再構築します。モジュールではないので、/etc/conf.modulesファイルは変更する必要はありません。

2枚のカードが同じカードだったり、同じなくとも同じドライバを使う場合には、注意が必要です。使っているネットワークカードのドライバ名を調べるには、Linux Ethernet-Howto（http://www.linux.or.jp/JF/JFdocs/Ethernet-HOWTO.html）を読むとよいでしょう。

Q ISDNルータを使う設定は？

A Linux側の設定は、LANに接続する時と特に変わりはありません。ただし不要な発呼を行わないように、マニュアルを参考にしてWindowsなどでの使用方法と同様にISDNルータを設定しましょう。



画面22 LinuxconfでPPPの設定を行う

Q モデム／TAを使う設定は？

A 電話回線を通じてインターネットサービスプロバイダに接続するには、PPP（ポイント・ツー・ポイントプロトコル）を使います。LinuxではPPP機能を提供するプログラムとして、pppdまたはPPxPがよく使われています。PPPの設定はLinuxconfで設定が簡単に行えるようになっています（画面22）。

なお、PPP接続のための設定ファイルは、/etc/pppディレクトリにあります。

▼リスト6 /etc/sysconfig/networkファイルの例

```
NETWORKING=yes
FORWARD_IPV4="yes"
HOSTNAME="linuxmag.hoge.com"
GATEWAY="192.168.1.10"
GATEWAYDEV="eth0"
```

▼リスト7 /etc/sysconfig/network-script/ifcfg-eth0ファイルの例

```
DEVICE="eth0"
BOOTPROTO="none"
ONBOOT="yes"
IPADDR="192.168.1.10"
NETMASK="255.255.255.0"
IPXNETNUM_802_2=""
IPXPRIMARY_802_2="no"
IPXACTIVE_802_2="no"
IPXNETNUM_802_3=""
IPXPRIMARY_802_3="no"
IPXACTIVE_802_3="no"
IPXNETNUM_ETHERII=""
IPXPRIMARY_ETHERII="no"
IPXACTIVE_ETHERII="no"
IPXNETNUM_SNAP=""
IPXPRIMARY_SNAP="no"
IPXACTIVE_SNAP="no"
```

Q IPマスカレードとは？

A ISDNルータには標準的についている機能で、外部のネットワークから内部のネットワークを見せないようにしてセキュリティを高める仕組みのことです。内部のネットワークから出されるパケットのアドレスを、外部に面するIPマスカレードを導入したホストのIPアドレスに置き換えて、データをやり取りします。そのため、内部ネットワークに複数台のマシンがあっても、外部から見えるのはIPマスカレードを導入した1台だけになります。外部との通信は、ISDNやモデムを使う場合と、LANや専用線を使う場合の両方があります。

Linuxマシンをゲートウェイとして IPマスカレードを実現することができます。LAN同士で接続する場合、ネットワークカードを2枚挿して、外部用と内部用のネットワークアドレスの設定を行います。kernel 2.0とkernel 2.2では設定方法が違うので注意が必要です。なお、Kernel 2.2ではipchainsというプログラムが必要になるのでインストールしておきましょう。

Linux IP Masquerade mini HOWTO（http://www.linux.or.jp/JF/JFdocs/IP-Masquerade.html）が参考になるでしょう。

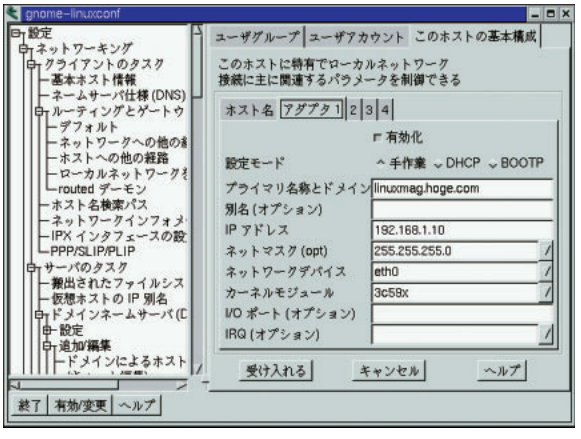
Q IPアドレスやホスト名を変更したいが、どこを変えればいい？

A Red Hat系のディストリビューションなら、linuxconf（**画面23**）ユーティリティを使うのが簡単です。TurboLinuxなら、turbonetcfg（**画面24**）ユーティリティを使いましょう。

設定ファイルをエディタで変更するなら、ホスト名は/etc/sysconfig/network（**リスト6**）ファイルの「HOSTNAME＝」を変更します。IPアドレスは/etc/sysconfig/network-script/ifcfg-eth0（**リスト7**）の「IPADDR＝」を変更します。そのほかのネットワークパラメータも同様に変更できます。

設定を変更したら、

```
# /sbin/ifdown eth0     <---停止
# /sbin/ifup eth0       <---起動
```



画面23 Linuxconfでネットワークの設定を変更する

画面24 turbonetcfgでネットワークの設定を変更する

と行って、ネットワークカードを停止／再起動してください。また、

```
# /etc/rc.d/init.d/network restart
```

と行ってネットワークを再起動することもできます。

Q Windowsマシンとファイル／プリンタを共有したい

A Sambaというユーティリティを使うと、Linuxマシンを Windowsマシンのファイルサーバや、プリンタサーバとして使うことができます。Red Hat系のLinuxでは、Sambaの設定ファイルは、/etc/smb.confです。初期設定ではほとんど“；”でコメントアウトされていますので必要な部分を修正します“［ ］”で囲まれた名前が共有名となっています。プリンタは［printers］セクションで設定します。

なお、Sambaに接続する時にユーザー認証が行われるため、あらかじめパスワードを設定しておく必要があります。Linuxで使われる/etc/passwdではなく、/etc/smbpasswdファイルを使いますので、smbpasswdコマンドでパスワードを設定しておきましょう。なお、smb.confを変更したあとに、

```
# /etc/rc.d/init.d/smb restart
```

と行ってSambaを再起動すると、変更後の設定になります。

Samba 2.xから、SWATというWebブラウザベースで動作する、Samba設定ツールがあります。これを使うとSambaの各種設定が楽に行えます（**画面25**）。

また、Sambaに含まれるsmbclientプログラムで、LinuxからWindowsマシンにあるファイルにアクセスすることもできます。

Sambaについての情報は、Japanized SAMBA Web Pages（http://samba.bento.ad.jp/）をご覧ください。

Q Macintoshとファイル／プリンタを共有したい

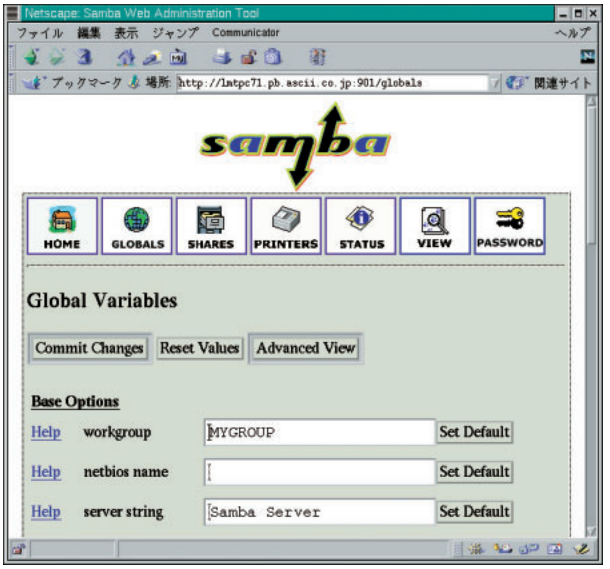
A netatalkというユーティリティプログラムを使うと、LinuxマシンがAppleTalkのファイルサーバとして使うことができるようになります。

netatalkについての情報は、netatalk（http://www.umich.edu/~rsug/netatalk/）を参照してください。

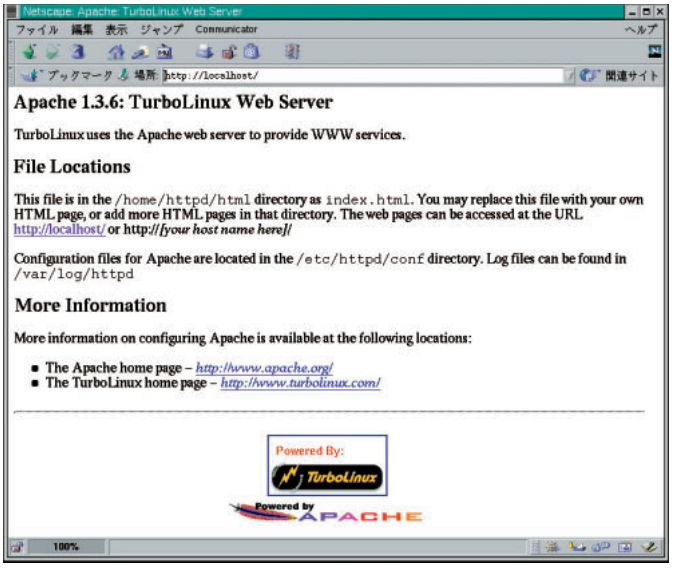
Q Webサーバを起動したい

A ほとんどのLinuxディストリビューションには、Webサーバソフト Apacheが含まれています。“ps ax”でプロセスの実行状態を見て、httpdという行が何行も表示されるようなら、すでにApacheは起動されています。

設定ファイルは/etc/httpd/conf（あるいは/usr/local/apache/conf/）ディレクトリに、そしてHTMLファイルは、/home/httpd/html（あるいは/usr/local/apache/htdocs/）ディレクトリにあります。試しに、Netscapeを起動してみましょう。URLに、localhostと入力することでローカルマシンにアクセスできます（**画面26**）。



画面25　SWATはSambaの設定を、Webブラウザを使って変更できる



画面26　Apacheの動作確認を行う。URLにlocalhostを使うと自アドレスをあらわす。

miscellaneous

Q Xを使わずに複数のコンソールを使いたい

A Linuxには仮想コンソールという機能があります。コンソール画面で、Alt＋ファンクションキーを押すと、それぞれ別のコンソール画面を使うことができます。使える画面の数はディストリビューションによって多少違いますが、普通はF1～F6までの6画面が使えるでしょう。

仮想コンソールは、X Window System環境からも利用できます。X Window Systemからは、Ctrl＋Alt＋ファンクションキーで画面を切り替えます。ほとんどのディストリビューションでは、Alt＋F7でX Window Systemの画面に戻ります。

Q 変な名前のファイルができてしまって消すことができません

A －（ハイフン）で始まるファイル名のものを消そうとして、普通にrmコマンドを実行すると、rmコマンドは引数をファイル名だと解釈せず、オプションと勘違いします。たとえば、次のように-fileというファイルを消そうとしてもエラーになって消すことができません。

```
# rm -file
rm: invalid option -- l
```

そこで、ファイル名の前に./をつけて相対パスでファイル名を指定します。

```
# rm ./-file
```

また、ハイフンを重ねると、それ以降にオプションはないという意味になりますので、“ -file ”をファイル名だと解釈します。

```
# rm -- -file
```

次はファイル名にスペースが入っている場合です。この場合、ダブルクォーテーションかシングルクォーテーションマークでファイル名を囲めば消すことができます。

```
# rm "blank file"
```

最後は、？マークの入った、得体の知れないファイルです。Windowsなどから日本語の名前のついたファイルをFTPでLinuxマシンに転送すると、シフトJISが表示できないLinux上では化けてしまいます。たとえば、“ 業務報告 ”というファイルは“ ???????? ”のように見えます。ktermなど日本語の表示できるターミナル上で、

```
# ls | nkf -e
```

として、漢字コードをEUCに変換すれば本当のファイル名を見ることができます。そこで、この情報とワイルドカードを組み合わせ、慎重に消すことになります。“ 業務報告 ”は8バイトですので、1バイトのキャラクタ1個にマッチするワイルドカード“ ？ ”を8個並べれば消すことができます。同じディレクトリに8バイトの長さの名前を持つファイルがあると、すべてマッチしてしまうので、rmコマンドには必ず-iオプションをつけて対話形式で使いましょう。

```
# rm -i ????????
```

また、消したいファイルが固有の拡張子をもっていればそれを利用することもできます。たとえば“ ?y?[?W??.xls ”などと化けてしまった“ ページ数.xls ”を消すとき、同じディレクトリにxlsという拡張子を持つファイルがなければ、

```
# rm -i *.xls
```

として消すことが可能です。念のため、rmには-iオプションをつけておきましょう。

Q カレントディレクトリにあるプログラムが実行できません

A LinuxやUNIXではDOSとは違い、カレントディレクトリにパスを通さないのが普通です。これはセキュリティを確保するためです。

たとえば、悪意を持った人がトロイの木馬やクラックプログラムを/rootにlsという名前で置いたとします。このとき、rootユーザーのPATH変数の先頭にカレントディレクトリが含まれていたらどうなるでしょう。rootユーザーがログインして、ホームディレクトリ（/root）でlsを実行すると、本当のlsではなく、悪者が置いた悪のプログラムが実行されてしまいます。

このようなことが起こらないよう、カレントディレクトリにパスを通してはいけません。カレントディレクトリにあるプログラムが安全であることが

確認できていれば、/（ルート）ディレクトリからの絶対パスでプログラムを指定するか、

```
# ./hoge
```

のように、.（カレントディレクトリ）からの相対パスを指定してプログラムを実行します。

また、/usr/local/binなどに一般のユーザーが実行ファイルを置けるようにしているなら、rootユーザーのPATH変数に設定するディレクトリの順序にも気をつけましょう。できれば、これらのディレクトリにはパスを通さないのが安全です。

Q キートップとは違う記号が入力されてしまう

A 日本語キーボードを使っているのに、英語キーボードの設定になっているなど、設定が間違っているのでしょう。

キーボードの設定は、コンソール環境とX Window Systemで別々に行う必要があります。

コンソール環境では、loadkeysコマンドに、引数としてキーマップファイルを指定することでキー配置を設定しています。キーマップファイルとは、キーボードごとのキー配置を記述してあるファイルです。

LASER5 Linux 6.0やTurboLinux 4.xでは、/etc/sysconfig/keyboardというファイルで、キーマップファイルを指定します。たとえば、このファイルに、“ KEYTABLE="jp106" ”と記述すると、次回起動時から日本語106キーボードのキー配置になります。また、Slackware 7.0では、/etc/rc.d/rc.keymapのなかでキーマップファイルを指定しています。

どのようなキーマップファイルがあるのかは、キーマップファイルのある/usr/lib/kbd/keymaps/i386/qwerty/ディレクトリを見ればわかります。

X Window Systemでは、/etc/X11/XF86Configというファイルの中で指定します。この中の“ Section keyboard ”で“ XkbModel "jp106" ”と、“ Xkb Layout "jp" ”を記述すれば、日本語106キーボードの設定になります。

Q Caps Lockキーと左Ctrlキーの機能を入れ替えたい

A ひとつ前の質問と同様に、コンソール環境とXの環境を別々に設定する必要があります。ここでは、TurboLinux 4.xとLASER5 Linuxを例に説明します。

コンソール環境では、jp106などのキーマップファイルをもとにして、Caps Lockキーと左Ctrlキーを入れ替えたキーマップファイルを作り、これを使うように設定をします。

作業は次のようになります。まず、キーマップファイルのあるディレクトリをカレントにします。

```
# cd /usr/lib/kbd/keymaps/i386/qwerty
```

もととなるjp106.map.gzをjp106_CC.map.gzにコピーします（LASER5 Linuxでは、以下“ map ”を“ kmap ”に読み替えてください）。

```
# cp jp106.map.gz jp106_CC.map.gz
```

gzipで圧縮されたjp106_CC.map.gzを展開します

```
# gunzip jp106_CC.map.gz
```

展開されたjp106_CC.mapをエディタで開き、次の2カ所を書き換えます。

```
keycode 29 = Control → Caps_Lock
```

```
keycode 58 = Caps_Lock → Control
```

エディタを終了して、gzipで圧縮します。

```
# gzip -9 jp106_CC.map
```

新しいキーマップファイルができましたので、/etc/sysconfig/keyboardで、キーマップファイルの指定を“ jp106 ”から“ jp106_CC ”に変更します。変更したら、キーマップファイルの変更をシステムに反映させます。

```
# /etc/rc.d/init.d/keytable restart
```

これで、コンソール環境ではCaps Lockキーと左Ctrlキーの機能が入れ替わりました。

X環境での設定は比較的簡単です。エディタで、/etc/X11/XF86Configを開き、“ Section "Keyboard" ”の中に、“ XkbOptions "ctrl:swapcaps" ”を追加します。このあと、Xサーバを再起動すれば、Caps Lockキーと左Ctrlキーの機能が入れ替わります。

Q システムの時刻を合わせたい。

A 時刻の設定にはdateコマンドを使います。西暦2000年1月8日22時33分に合わせるなら、次のように指定します。

```
# date 0108223300
```

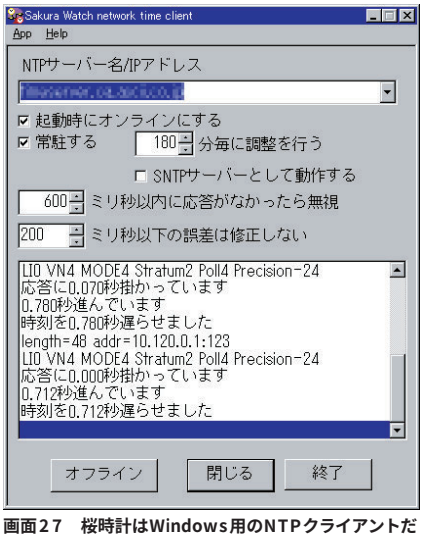
ネットワークを通じてTIMEサーバを利用できるなら、rdateコマンドを使って、TIMEサーバ（仮にtime server.hoge.netとします）と時刻設定を同期させることができます。

```
# rdate -s timeserver.hoge.net
```

秒以下のオーダーで厳密に時刻を合わせるには、TIMEプロトコルではなく、NTP（Network Time Protocol）を使います。NTPを利用するなら、xntp3（http://www.eecis.udel.edu/~ntp/）をインストールします。ダイヤルアップ環境なら、xntp3に含まれるNTPクライアントntpdateでNTPサーバにつないで時刻を同期させましょう。常時接続の環境であれば、デーモンのxntpdを走らせておけば、ほとんど誤差なく同期をとることができます。

xntp3は、NTP Ver.3に対応しており、最新版はNTP Ver.4に対応したntpになりますが、現状では互換性の問題が発生しないようにxntp3を使ったほうがよいかもしれません。

xntp3やntpは、標準ではインストールされないことも多いので、CD-ROMやFTPサイトからファイルを入手してインストールします。



画面27　桜時計はWindows用のNTPクライアントだ

また、Linux上でxntpdを動作させておけば、そのマシンがNTPサーバとなります。LANでつながったWindowsマシンで“桜時計”などのNTPクライアントを実行すれば、このマシンも正確な時刻に同期させることができます。

Q 起動時にエラーが出ているのですが、早すぎて読み取れません

A 起動後、dmesgコマンドを実行してください。これでもわからない場合は、起動してからShift＋PageUPを利用してください。また、Ctrl＋Sでスクロールをストップさせることもできます（再開はCtrl＋Q）。

Q Vine Linuxで、パスワードをシャドウパスワードにするにはどうしたらいいでしょうか？

A LinuxやUNIXでは、ユーザーのパスワードは暗号化され、/etc/passwdに記録されるのが普通でした。/etc/passwdは、パスワード情報以外のユーザー情報も記録されており、さまざまなプログラムから参照されるため、ファイルのパーミッションは644、すなわち誰でも見られるようにせざるをえないのです。

パスワード情報は不可逆な変換による暗号化をしてあるため、従来は安全とされていたのですが、コンピュータの速度が飛躍的に向上し、膨大な単語辞書を用いて総当たり攻撃をするクラッカーも現れ、現在では十分なセキュリティを保てなくなりました。

そこで考えられたのがシャドウパスワードです。シャドウパスワードは、/etc/passwdから、パスワード情報を抜き出し、rootユーザーしか見ることのできない/etc/shadowに移してパスワードの安全性を高める仕組みです。

Vine Linuxには、シャドウパスワードが標準ではインストールされていませんが、rootユーザーになって、pwconvコマンドを実行すればシャドウ化が実現できます。

Q システムが反応しません。リセットしていい？

A キーボードがロックアップして使えない場合は、ネットワークログインを試してみましょう。それでもだめなときは、リセットボタンを押す、あるいは電源を切るしかありません。この場合、しばらく待ってハードディスクのアクセスランプが点灯していないことを確認します。ディスクキャッシュに残っているデータも、一定の時間がたつとハードディスクに書き込まれるので、こうすることで、ディスクデータの破壊を最小限に押さえることができます。

次回の起動時に、fsckが実行され、ファイルシステムが壊れていないかチェックされます。

Q ディスクとファイルシステムのチェックをしたい

A 新しいハードディスクにファイルシステムを構築したときは、念のためチェックをしてから使いましょう。

チェックを行うにはfsckコマンドを使います。/（ルート）ファイルシステム以外は、マウントしていない状態でfsckを実行します。/（ルート）ファイルシステムにfsckを実行するときは、シングルユーザーモードで行います。

いまが買い時！１３Ｇバイトが１万円台前半！

ハードディスク大増設指南

いまどき６Ｇや８ＧでLinuxしてる時代じゃない！ あきれるほど安くなった高速・大容量ハードディスクの購入・セットアップからLiunxでの活用まで懇切丁寧に伝授します

photo : Takashi Shinohara(Dee)

IDEのハードディスクを増設しよう

文：上間俊雄　Text: Toshio Uema

2年ほど前には、サーバ用途が主だったLinuxのハードディスクは、ファイルアクセスが速いSCSIが普通だった。CPUによるPIO転送しかできないIDEだと、ハードディスクのアクセスがCPUを占有するため、低速なCPUではほかのプロセスに対するレスポンスまで悪くなってしまうからである。

しかし、マシンの全体的なパフォーマンスが急速に上がり、大容量のメモリが使え、さらにバスマスタ方式が登場したりと、IDEのハードディスクでも遅く感じなくなってきたのだ。現在購入できるIDEのハードディスクは、10～15Gバイトで1万5000円程度。SCSIでは、9Gバイトで3万円程度だ。

確かにアクセスの多いサーバでは、SCSIのドライブのほうが現在でも信頼性はずっと高い。また、10000rpmというとてつもなく高回転でアクセスの速いドライブもSCSIには存在している。だが、SCSIのハードディスクは高価で、高速なドライブになれば、マシンが暴走してしまいそうなくらい発熱が大きく、しかもうるさい。さらに、SCSIデバイスを接続するためのアダプタカードも別途用意しなければならない。

IDEなら同じ金額で、安価なハードディスクと、最近安定性がかなり上がったATAPIのCD-RドライブやMOドライブなどのバックアップデバイスを購入してもお釣りがくる。しかも、同じ予算ならIDEのハードディスクでは、SCSIよりも2～5倍の容量のドライブが購入できる。

また、最近では一般的になってきたUltra-ATA66のIDEのドライブでは、最大転送速度が66Mバイト／秒となり、Ultra2 SCSIのドライブの40Mバイト／秒と比べてみても高速だ。確かにこれらの数値は理論値にしかすぎないが、同等程度かそれより少し遅いくらいの速度が実現できれば、IDEのほうがコストパフォーマンスが高いのは明らかだ。

社内のファイルサーバやルータマシンなら、IDEのハードディスクでもまったく問題ない。それに、サーバの負荷を気にするのなら、IDEやSCSIよりもメモリを増やしたほうがいいかもしれない。

データをサーバ上に置いて共有する

さて、SambaやNetatalk、NFSなどを利用できるようにしたLinuxのサーバがあれば、Windows、MacOS、LinuxとOSに左右されずにデータの共有ができるので、データを何でもサーバ上に置きたくなる。

データをサーバに置いておけば、いろいろなソフトウェアをインストールして不安定になりがちなクライアントマシンがたとえダウンしたとしても、大事なデータは大丈夫である。さらに、一括してバックアップできるところも便利なところだ。そうなると、Linuxのサーバにも大容量のハードディスクが欲しい。

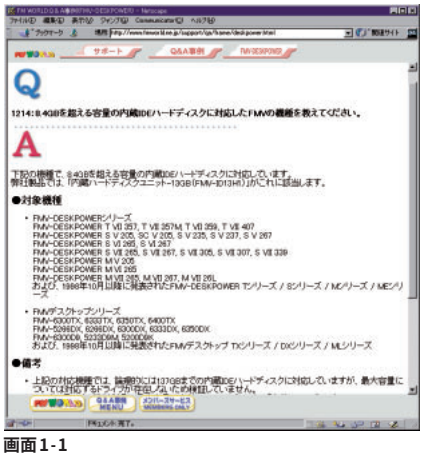
以前、本誌で「5万円で作るLinuxマシン」という記事で実際にマシンを組み立てたが、そのときにはIDEの4Gバイトのハードディスクを使った。当分は問題ない考えていた4Gバイトのハードディスクも、その後またたく間に容量不足に陥ってしまった。画像や音声データ、日々大量に届くメールマガジンの数々をサーバ上に置いていたら、あっという間に容量が足りなくなってしまったのだ。

そこで、大容量のIDEのハードディスクを増設して、容量不足を一気に解消したい。Linuxのハードディスク増設では、最後のマウント作業以外は、Windowsで増設するときとやることはそれほど変わらない。今回は13GバイトのIDEハードディスクを増設してみることにする。

IDEのハードディスクの場合には、マザーボード側がその容量に対応しているかどうかが最初のポイントになる（**画面1-1**）。もし、少し古いPCやマザーボードなどで、8.4Gバイト以上に未対応のときには、BIOSのアップデートで対応できることもあるので、PCメーカーやマザーボードメーカーのホームページを調べてみよう。

Linuxから見たIDEのデバイス

AT互換機のマザーボードのチップセットにはIDEディスクコントローラが内蔵されていて、プライマリとセカンダリにそれぞれ2台ずつ、合計4台の



画面1-1
富士通のFM WORLDのサポートページには、8.4Gバイトを超える容量に対応している機種が掲載されている

IDEデバイスを接続することができる。

また、プライマリに接続できる2台のデバイスは、マスタとスレーブというように分かれている。セカンダリについても同様だ。ATAPI（IDE）のCD-ROMドライブは、プライマリかセカンダリのスレーブ側に接続されていることが多い。筆者の環境では、プライマリのマスタに4Gバイトのハードディスク、プライマリのスレーブにCD-ROMドライブが接続されている。今回の13Gバイトのハードディスクは、セカンダリのマスタに接続することにする。

さて、Linuxから見たIDEのデバイスは、「/dev/hd?」というような表し方になっている。「?」の部分には「a」「b」「c」「d」が入り、**表1-1**のようになる。さらにハードディスクの内部を区切ってパーティションに分ける場合、「/dev/hda1」というように番号が割り振られる。

増設の手順

IDEハードディスクの増設には、いくつかのステップを踏んで行う。途中まではAT互換機のすべてのOSに共通だ。まず、ハードディスクのジャンパ設定をして、マスタかスレーブのどちらかのジャンパピンを止める。

ハードディスクは説明書のない状態で販売されていることが多いので、メーカーのWebサイトを参考にして設定するとよい（**表1-2**）。今回は、マスタに設定した。ジャンパピンの設定が終われば、IDEケーブルをマザーボードにあるIDEのセカンダリポートとハードディスクにそれぞれ接続して、あとは電源コネクタを接続する。

プライマリのマスタ	/dev/hda
プライマリのスレーブ	/dev/hdb
セカンダリのマスタ	/dev/hdc
セカンダリのスレーブ	/dev/hdd

表1-1 IDEのデバイス

■BIOSの設定

接続が終わったなら、さっそく電源を起動しよう。このときにBIOSでハードディスクを認識させるためにマザーボードのBIOSメニューを起動する。BIOSメニューの起動の仕方はマザーボードによって違うが、たいていは起動時のメモリチェックが終了するまでにキーボードの［Del］キーか［F1］キーを押す。ここでは、AWARD BIOSを例に話を進めることにしよう。

AWARDでは起動時に［Del］キーを押すことでBIOSが起動する。まずは、「STANDARD CMOS SETUP」で「HARD DISK」の「Secondary Master」の「TYPE」を“Auto”にする。次に、右端の「MODE」は“LBA”とした。

このLBA（Logical Block Address）はハードディスクへアクセスするときに、論理ブロックアドレスでその位置を指定する。“CHS”というモードもあって、ハードディスクの物理的なシリンダ、ヘッド、セクタという3つのパラメータで位置を指定する。CHSでは8.4GB以上のアドレスを指定できない。

さて、これが終わったなら、ハードディスクを認識するかどうか「IDE HDD AUTO DETECTION」でチェックしてみよう。Secondary Masterのハードディスクが認識されているようであれば、BIOSの設定を書き換えてLinuxを起動する。ここまではWindowsでも同様だ。

■パーティションの設定

増設ハードディスクの設定を行うた

メーカー	URL
Quantum	http://www.quantum.com/support/hdd/hdd_support_ata.htm
IBM	http://www.storage.ibm.com/storage/techsup/hddtech/instgde.htm
Western Digital	http://www.wdc.com/service/FAQ/jumpers.html
Maxtor	http://www.maxtor.com/products/products.html
Seagate	http://www.seagate.com/support/disc/specs/interface_ata_st3.html
Fujitsu	http://www.fujitsu.co.jp/hypertext/hdd/drive/disk.html

表1-2 ハードディスクメーカーのジャンパ設定のWebページ

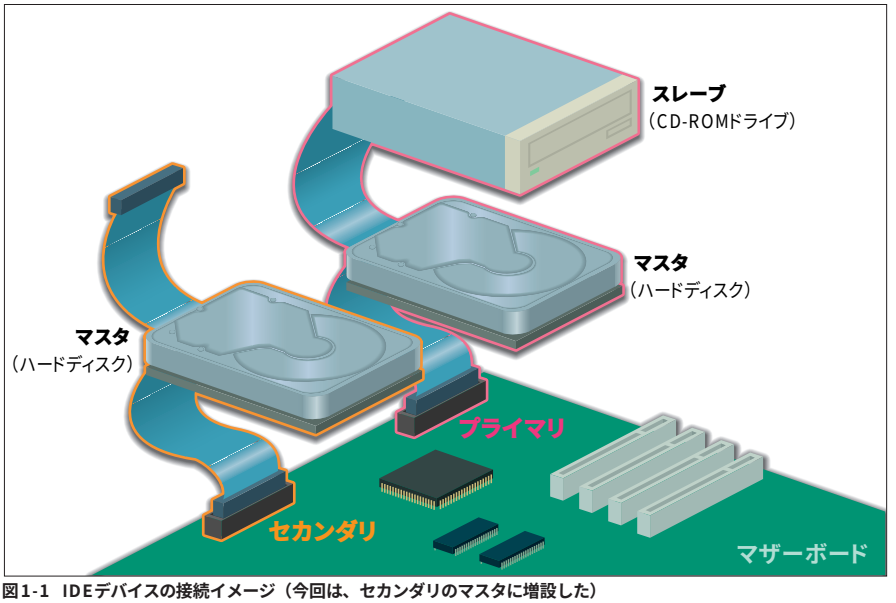


図1-1 IDEデバイスの接続イメージ（今回は、セカンダリのマスタに増設した）

めにLinuxを起動するときには、できるだけシングルユーザーモードで起動させよう。このシングルユーザーモードとは、文字通りスーパーユーザー1人だけがコンソールからアクセスする方法で、外部からtelnetでアクセスしたりできず、スーパーユーザーが指示しない限りファイルを書き換えたりもしない。そのため、バックアップや、ログなどの情報を厳密に管理できるモードだ。マシンへのアクセスの多いサーバの場合には、とくにこのモードで設定するようにしたい。起動時にシングルユーザーモードで起動する場合には、liloのブート指定の「LILO boot:」の状態で、「linux 1」とか「linux -s」とオプションを付けて起動する。なお、"linux"の部分は「/etc/lilo.conf」でLinux起動時のlabelに書かれたものを入力する。すでにLinuxが起動しているときには、スーパーユーザーで「/sbin/shutdown -a now」とすることでシングルユーザーモードに移行する。

Linuxではfdiskなどを使ってLinux用のパーティションを切る。Debian GNU/Linuxなどでは、cfdiskというパーティションを設定するツールが用意されている。fdiskはパーティションを切りたいドライブをオプションで指定して起動する。今回はセカンダリのマスタのパーティションを区切るので、「fdisk /dev/hdc」とする。fdiskの使い方は、インストールのときと同様だ。今回は、13Gバイトを丸々1パーティションとして、Linuxのパーティションを作成した（**画面1-2**）。

■ファイルシステムの作成

Linux用のパーティションを作成したら、次はLinuxのファイルシステム（Ext2）の作成だ。これはDOSやWindowsでいうところのFORMATコマンドだ。ファイルシステムの作成には、mke2fsコマンドにパーティション名を指定して使う。

```
# mke2fs /dev/hdc1
```

通常はオプションを指定せずにファイルシステムを作成してよいが、オプションを指定すれば細かく設定できる。なお、不良ブロックがないかどうかテストをしてからファイルシステムを作成するときには、「-c」オプションを付ける。その場合、大容量のハードディスクではチェックの時間がかなりかかる。

■ファイルシステムをチェック

MS-DOSやWindowsではscandiskというファイルシステムをチェックするプログラムがあるが、Linuxでもfsckというプログラムが用意されていて、Ext2ファイルシステム版のe2fsckというプログラムが用意されている。

e2fsckはExt2のファイルシステムをチェックして、問題があれば修復してくれる。使い方は、e2fsckにパーティション名を指定する。

```
# e2fsck /dev/hdc1
```

もし問題が起きているときには、修復するかどうか確認してくるので（yes/no）のどちらかを入力する。これが面倒な場合には、オプションに「-p」を指定することで、自動的に修復する。

■ファイルシステムをマウントする

Linuxではいくらファイルシステムを作成しても、マウントしなければ実際に利用することはできない。そこでファイルシステムをマウントしよう。あらかじめマウントするディレクトリを作成しておく。

```
# mkdir /home/data
```

ファイルシステムのマウントは、フロッピーディスクやCD-ROMと同様にmountコマンドを利用する（**画面1-3**）。

たいていの場合はマウントするファイルシステムのタイプを指定しなくても、プログラム側でスーパーブロックからファイルシステムを判別するので、

```
# mount /dev/hdc1 /home/data
```

と行ってもよいだろう。

アンマウントするには、

```
# fdisk /dev/hdc
    ⟨
Command (m for help): n          ←― 新パーティションの作成

Command action
   l   logical (5 or over)
   p   primary partition (1-4)
p                                ←― プライマリパーティションの作成
Partition number (1-4): 1        ←― 作成するパーティション番号の指定
First Cylinder: 1                ←― シリンダ範囲の指定（スタート）
Last Cylinder: 1582              ←― シリンダ範囲の指定（エンド）

Command (m for help): w          ←― パーティションテーブルを書き換えて終了
```

画面1-2 fdiskコマンドでパーティションを設定する

```
# umount /dev/hdc1
```

とする。ただし、アンマウントしたいファイルシステムにあるファイルが開かれた状態であるとアンマウントできない。カレントディレクトリをそこに移動している場合も同様でアンマウントできないので注意しよう。

■起動時にファイルシステムをマウントする

マウントするにはコマンドを入力しなければならない。だとすると起動時にはマウントできないはずだが、「/etc/fstab」というファイルに情報を書き込んでおくことで起動時にマウントするようになる。fstabはスーパーユーザーの状態からエディタで編集する。

```
# vi /etc/fstab
```

ここでは、行ごとに1つのマウント設定となっている。1番目のパラメータにファイルシステムのパーティション、2番目のパラメータにマウントするディレクトリ、3番目のパラメータにファイルシステムのタイプを入力する。

4番目のパラメータには、マウントするときのオプションを指定する。ハードディスクの場合「defaults」と設定すればよい。たとえばCD-ROMなどは「user」を設定すると、一般ユーザーがマウントできるようになる。オプションの詳細は「man mount」で確認してほしい。

5番目のパラメータは、ダンプといって、削除したファイルを復元できるようにするかどうかを設定する。ここでのパラメータは、「0」（ダンプしない）、「1」（ダンプする）のどちらか。たとえば、「/tmp」や、プロキシキャッシュのディレクトリはダンプする必要がないので、「0」にする。

最後に6番目のパラメータは、起動時にfsckを実行するかどうかを設定する。ここでの設定は3種類あり、数値で表わす。「0」はfsckをしない。「1」は「/」（ルート）のファイルシステムでfsckをする。「2」は、「/」（ルート）以外のファイルシステムでfsckをする（**画面1-4**）。

設定したら、あとはマシンを再起動すると自動的にマウントされるようになる。

```
# mount -t ext2 /dev/hdc1 /home/data
                |         |         |
                |         |         マウントするディレクトリ
                |         マウントするパーティション
                ファイルシステムのタイプ
```

画面1-3 マウントコマンドの指定

```
# <file system> <mount point>   <type>  <options>              <dump>  <pass>
/dev/hda1       /               ext2    defaults,errors=remount-ro   0      1
/dev/hda2       none            swap    sw                     0       0
/dev/hda3       /usr            ext2    defaults               1       2
/dev/hda5       /var            ext2    defaults               1       2
/dev/hda6       /home           ext2    defaults               1       2
proc            /proc           proc    defaults               0       0

/dev/hdc1       /home/data      ext2    defaults               1       2
```

画面1-4 /etc/fstabを設定して、増設したハードディスクを起動時にマウントする

```
# mv /home/samba/ /home/samba.old/ ←― 現sambaのディレクトリ名を変更
# mkdir /home/samba/ ←― 新sambaのディレクトリ名を作成
# mount -t ext2 /dev/hdc1 /home/samba/ ←― マウント
# df ←― マウントしているファイルシステムを調べる。
Filesystem           1k-blocks      Used Available Use% Mounted on
/dev/hda1              3973832   1184452   2583750  31% /
/dev/hdc1             12229366         1  11593996   0% /home/samba
```

画面1-5 現在のSambaディレクトリはリネームしておき、/dev/hdc1を/home/sambaにマウントする

ファイルのコピー方法

増設したハードディスクがLinux上で使えるようになったら、ハードディスクを圧迫していたデータを移動させよう。もちろん、このときもデータが確実に移動できるようにシングルユーザーモードで作業をする。また、先ほど設定した/etc/fstabの内容を反映させないように「/dev/hdc1」をコメントアウトし、すでにマウントしているなら、umountコマンドでアンマウントしておこう。

まず、Windowsで共有ディスクとして利用しているsambaのディレクトリが一番圧迫していそうなので、/home/samba/のディレクトリの容量をduコマンドで調べてみた。

```
# du -ms /home/samba/
```

オプションに「-m」と付けることで、Mバイト単位で容量を表示してくれる。調べてみると、かなりハードディスクを圧迫しているので、/home/samba/ディレクトリを新しいハードディスク

に移行させることにする。

さて、マウントしなければLinuxで利用できないので、ハードディスクをマウントさせるが、このときに移動した後にも/home/samba/を利用したい。そうすることで、あとで/etc/smb.confを書き換えたりしなくても済むからだ。まず、/home/samba/のディレクトリ名を/home/samba.old/に変更した。次に、新しいハードディスクで使用する/home/samba/というディレクトリを作成する（**画面1-5**）。

これで、新しくマウントするための環境ができたので、mountコマンドで、新しいハードディスクを/home/samba/のディレクトリにマウントする。実際にマウントできているかどうか調べるためにdfコマンドを使用した（**画面1-5**）。

マウントできているようなので、cpコマンドを利用して実際にファイルをコピーしよう（**画面1-6**）。このときに注意して欲しいのが、読み書き実行の権限と所有者についてだ。コピーするときには、cpコマンドに「-a」とオプションをつけて実行する。このオプションの「-a」はファイルの内容から権限や所有者について忠実に保ったコピーとなる。シンボリックリンクは、リンク先のファイルをコピーするのではなく、シンボリック自体をコピーする。オプションの「-a」自体は「-dpR」としたときと同じだ。

コピーが終わったなら、あとは/etc/fstabの内容をviなどのエディタを使って書き換え、再起動した後も新しいハードディスクに/home/samba/をマウントさせるようにする（**画面1-7**）。

これで、共有ディスクを移行することができた。少し使ってから問題がないようであれば古いsambaディレクトリを削除しよう。

```
# rm -rf /home/samba.old/
```

■tarを利用した方法

今回はcpコマンドを利用した方法を紹介した。Linuxのほぼすべてが GNUのcpを利用しているため、問題は起きないのだが、本来、GNUでないcpでは純粋にファイルのコピーしか行うことができないものがあり、そのような場合、シンボリックリンクやファイルのパーミッションが反映されない。そこで、ディレクトリを含めた複数ファイルをひとつアーカイブにしたり、展開したりするtarがよく利用されてきた。先程のcpコマンドでコピーするところで、**画面1-8**のような使い方でもファイルをコピーできる。コマンドで「()」の部分は1つのプロセスで動かすように命令している。一見するとどこか別のテンポラリにtarでアーカイブをしているようだが、ここでは、メモリ上でアーカイブをしつつ、「¦（パイプ）」でそのまま展開している。

■rsyncを利用した方法

rsyncは、サーバ間の2つのディレクトリやファイルの同期を取るソフトであるが、性質上ファイルのコピーとしても使うことができる。本来は、sshなどを併用して暗号化した情報のやり取りとして利用されている。利用方法は、tarと同様にcpコマンドの代わりに利用すればよい（**画面1-9**）。

```
# cp -a /home/samba.old/. /home/samba        ←コピー
```

画面1-6 以前のsambaでディレクトリから新しいファイルをコピーする

```
/dev/hdc1       /home/samba      ext2    defaults                1       2
```

画面1-7 /etc/fstabを書き換えて、起動時からマウントされるようにした

```
# (cd /home/samba.old/ ; tar cf - . ) | ( cd /home/samba/ ; tar xvfp - )
```

画面1-8 tarプログラムでファイルをコピーする方法

```
# rsync -auvb --exclude '*~' /home/samba.old/ /home/samba/
                              元データ          出力先
```

画面1-9 rsyncプログラムでファイルをコピーする方法

Column

容量を確認する

ハードディスクを増設したら古いハードディスクにあるディレクトリを新しいハードディスクに移動することもあるだろう。この場合、「du」というコマンドを利用して、どこにどれくらいの容量が利用されているのか調べることができる。

「du」のあとには、ディレクトリを指定すると、そのディレクトリの階層以下の容量が表示される。このときに、“-b”とオプションを指定すればバイト単位で、“-k”とするとkバイト単位、“-m”でMバイト単位での表示される。しかし、「du」で見た場合、サブディレクトリ全部が見えてしまう。これだと、見にくくなるので、「du -s」とすることで、ディレクトリ全体の容量が表示される。

また、既存のファイルシステムの利用状況を確認するには、「df」というコマンドを利用する。「df」コマンドでも「df -k」とするとkバイト単位、「df -m」とするとMバイト単位で確認することができる。

最新IDEハードディスクのメリットを引き出す

122～126ページでは、IDEハードディスクをLinux PCに増設する手順を紹介した。しかし、PC互換機の古くからの仕様にどっぷりはまったIDEには、ユーザーにとって「はまり」やすいワナが多く存在する。ここでは、そうしたIDEに絡む細かい問題とその解決方法を解説していく。

なお対象はデスクトップPCであり、ノートPCについては当てはまらない場合があることに留意していただきたい。またハードディスクの容量は、1Kバイト＝1000バイトで換算されるのが慣例なので、記事内でもこれに従っている。1Kバイト＝1024バイトではないので注意されたい。

「容量の壁」を克服する

IDEハードディスクには、歴史的な理由からその全記録容量が使えない「容量の壁」という障害が生じる場合がある。容量の一部でも使えればよいのだが、最悪の場合、起動時にハングアップしてしまうことすらあり、厄介な問題といえる。

■ジオメトリパラメータ

図2-1は、ソフトウェアからハードディスク内の各セクタを指し示す（アドレッシングする）2種類の方法を示している。古いIDEハードディスクは、**図2-1**の左図のように、アクセス対象のセクタを3つのパラメータで指し示す必要があった。すなわち、記録トラックを選択する「シリンダ（Cylinder）」と、何枚かあるプラッター*1の記録面を選択する「ヘッド（Head）」、そして1シリンダ中の記録箇所を選択する「セクタ（Sector）」である。セクタの物理的な位置を指し示すこれらはジオメトリパラメータと呼ばれる。また3つのパラメータの頭文字からCHSパラメータなどともいわれる。

一方、**図2-1**の右図のLBA（Logical Block Addressing）は、ディスク内の全セクタ（ブロックとも呼ばれる）に0から順番に論理的な番号を割り当てておき、その論理番号でセクタを指定するという方式だ。CHSパラメータによるアドレッシングに比べるとLBAは、ソフトウェアから見ると、ディスクの物理的な構造を意識する必要がなく、またパラメータが1次元なので扱いやすい。そのため、現在ではLBAでアクセスするのが一般的になっている。

しかしPCの世界では、旧来のソフトウェアとの互換性を維持するのが重要視されるため、今でもそこかしこにCHSパラメータの呪縛が残ってしまっている。たとえば画面**2-1**のようにBIOSセットアップでは、IDEハードディスクのCHSパラメータの最大値を設定する必要がある。ハードディスクの先頭セクタに書き込まれるパーティションテーブルも、CHSパラメータでパ

Glossary

＊1　プラッター
ハードディスクに内蔵される記録ディスクの1つを指してこう呼ぶ。通常のハードディスクでは、内部に複数枚のプラッターが格納されており、それらの表／裏の両面に磁気記録を行なう。

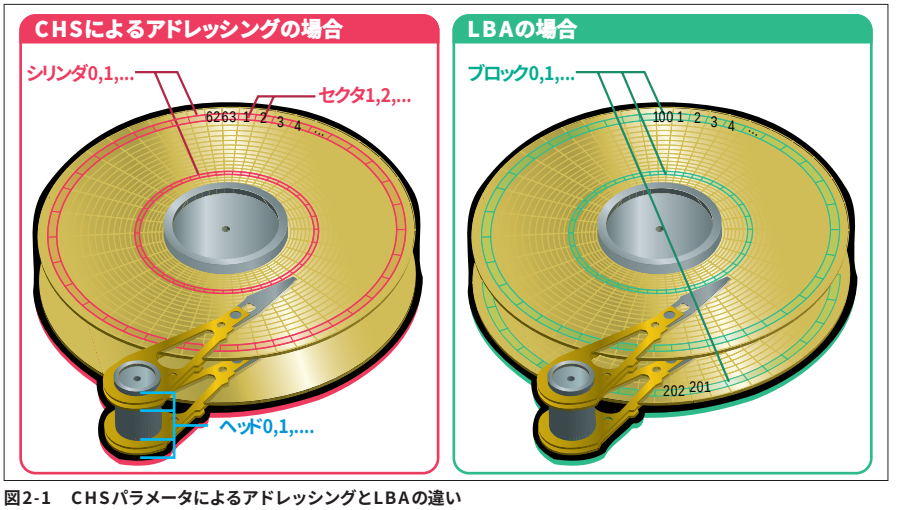
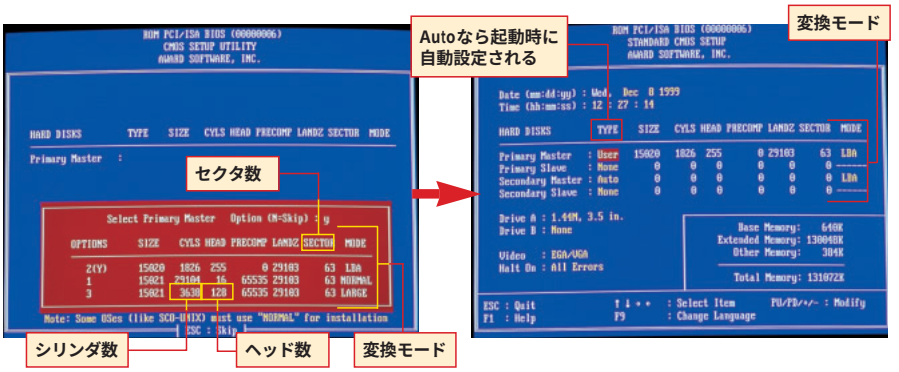


図2-1　CHSパラメータによるアドレッシングとLBAの違い



画面2-1　AWARD BIOSにおけるCHSパラメータの設定
左画面はCHSパラメータを自動検出したところ。BIOS内部でのパラメータ変換方式が3種類あり、CHSパラメータの値もそれぞれ異なる。この検出結果を有効にすると右画面のように実際にパラメータが設定される。

ーティションの位置やサイズが表現される。

ちなみにCHSパラメータが実際のディスク構造と一致していたのは、容量が100Mバイト以下とまだまだ小さかったころまでの話で、**画面2-1**左側にあるCHSパラメータのように、ヘッド数が実際に16あるいは255もあるわけではない。過去との互換性を維持するために、ディスクの全容量を表すCHSパラメータが必要なので、帳尻を合わせて各パラメータの数値が設定されているだけだ。

容量の壁がある原因は、ほとんどがCHSパラメータ絡みである。

■8.4Gバイトの壁

PCの電源が入ってOSが起動するまでは、ディスクBIOSが提供するInt13Hファンクションを使ってディスクをアクセスするのが普通である。これまでInt13Hファンクションでのアドレッシングには CHSパラメータが使用されてきた。その際、シリンダは0～1023、ヘッドは0～255、セクタは1～63の範囲でしか指定できない。そのため最大で約1652万セクタ、つまり約8.4Gバイトまでしかアドレッシングできないのだ。OSが起動すればBIOSは使わずにすむので、この8.4Gバイトの壁は直接関係しない。しかしOS自体はハードディスクの先頭から8.4Gバイト以内に配置しないと起動できない、といった制限が生じてしまうのだ（133ページのコラムにあるようにLinuxも該当する）。またDOSやWindows 3.1／9xのfdiskのように、ディスクBIOS経由でパーティションを設定するツールでも、この制限に引っかかるので、8.4Gバイトを超える領域は扱えない。

現在は拡張Int13Hファンクションが定義されており、64ビットの論理番号を扱えるLBAが利用できる。もちろんそれには、ディスクBIOSに拡張Int13Hファンクションが実装されている必要があるし、ディスクBIOSを呼び出す側も、この新しいファンクションをサポートしなければならない。非常に大雑把だが、BIOSの日付が1997年以前なら拡張Int13Hファンクションが実装されていない可能性が高いので注意したい。

■そのほかの容量の壁

8.4Gバイト以外にも容量の壁は存在する。もっとも古いのは528Mバイトで、IDEハードディスク自身とInt13Hファンクションがそれぞれ受け入れ可能なCHSパラメータの上限が異なっているのが原因だった。この問題を解決すべくIDEハードディスクに実装されたのが前述のLBAである。

一部のBIOSでは、2.1Gバイトまたは4.2Gバイトを超えるハードディスクを接続すると、全容量が認識できないばかりか、PCがハングアップするという障害が生じた。どちらもBIOS内部でCHSパラメータの取り扱いに問題があったのが原因だった。またつい最近では、33.8Gバイトを超えるハードディスクをAWARD BIOS 4.5xのPCに接続すると、やはり起動時にハングアップするという問題も出ている。

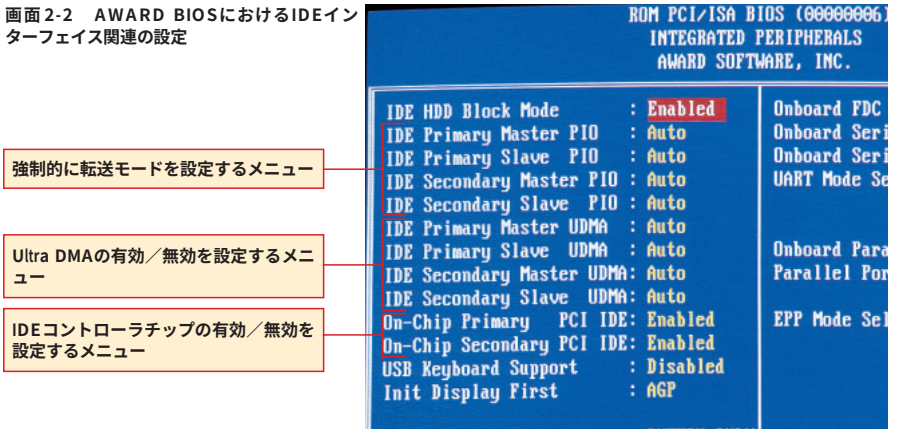
■容量の壁を解消する方法

以上は、すべてディスクBIOSに由来する障害であり、このほかにもOSやユーティリティのせいで容量の壁が生じる場合もある。Linuxに関しては133ページのコラムを参照していただきたい。ここではディスクBIOSによる容量の壁を解消する方法を示そう。

根本的に解決するにはディスクBIOSをアップデートすることである。マザーボード上のIDEコントローラにハードディスクを接続しているなら、バグの解消されたバージョンまでシステムBIOSをアップデートする。ところが古いマザーボードではメーカーによるBIOSアップデートが滞っている場合も珍しくない。そこで、拡張Int13Hファンクションなどに対応した最新のIDEカードを購入して、IDEインターフェイスごとディスクBIOSまで代替してしまうという手がある。今ならUltraDMA/66に対応したPCIカードを選ぶのが順当だろう。131～133ページでは、PCI-IDEカードをLinuxに導入する手順を紹介しているので参照していただきたい。

ソフトウェアでディスクBIOSを入れ替えるという手もなくはない。海外のIDEハードディスクメーカーは、PCが起動してOSがロードされる直前に、ディスクBIOSをソフトウェア的に差し替

画面2-2　AWARD BIOSにおけるIDEインターフェイス関連の設定



えるユーティリティを無償配布している（ほとんどはOnTrack製DiskManagerがもとになっているようだ）。ほとんど追加投資がないのがメリットだが、パーティションの管理方法が特殊なので、OSによってはインストールできないことがある。DOSやWindows 3.1／9xだけでPCを使うならよいが、ほかのOSまで使うのはかなり面倒なので、Linuxユーザーにはお勧めできない。

IDEの高速転送モードを使いこなす

最近のハードディスクは容量だけではなく速度も高められている。しかしIDEインターフェイスの転送レートが遅いと、せっかくの高性能も発揮しきれない。そこで現在最速の転送モードUltraDMA/66を使う条件を知っておこう。

■転送モードとは？

PC側のIDEインターフェイスとIDEデバイス（ハードディスクやCD-ROMドライブ）の間は、規格で定められた転送モードでデータのやり取りを行っている。IDEの規格はANSI（米国規格協会）で規定されており、最新のドラフト（正式規定前の仕様書）はATA/ATAPI-5である。その中では、PIOモードとマルチワードDMAモード、そしてUltra DMAモードという3種類の転送モードが規定されている。

PIOモードは、x86 CPUのループ命令などでデータを転送するのに使われる。最上位のPIOモード4で16.6Mバイト/秒まで発揮できるが、実質は4～7Mバイト/秒前後で頭打ちになることが多い。また他のデバイスの割り込み要求を邪魔するといったデメリットもあり、互換性維持のために存在するといってもよいだろう。

マルチワードDMAモードは一度に複数のデータをDMA転送できるモードで、CPUパワーをあまり必要としないメリットがある。最大転送レートはモード2で16.6Mバイト/秒だ。この速度をさらに高めたのがUltra DMAモードで、現在はモード0～4まで規定されている。モード2がUltraDMA/33（33Mバイト/秒）、モード4がUltraDMA/66（66Mバイト/秒）に該当する。

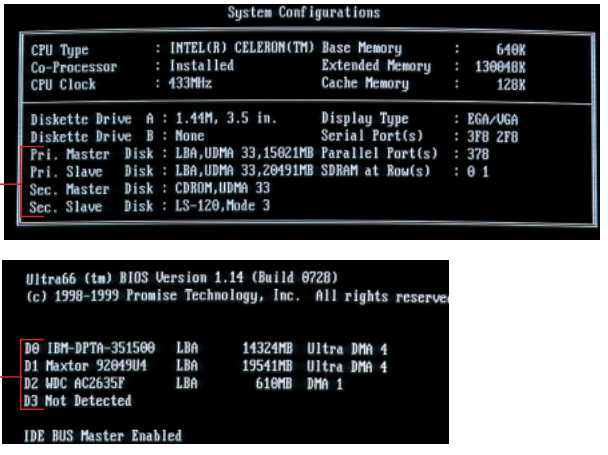
■UltraDMA/66を使うのに必要な条件

最新のIDEハードディスクはどれもUltraDMA/66に対応しているので、今ハードディスクを購入するならUltraDMA/66モードで利用したいところだ。しかし、それにはハードディスク以外にBIOSとIDEコントローラ、IDEケーブル、OS用ドライバのすべてがUltraDMA/66をサポートしていなければならない。

Ultra DMAモードをサポートしているマザーボードの場合、**画面2-2**のようにBIOSセットアップでUltra DMAモードを有効／無効にする、あるいは強制的にPIOモードに設定する、といったことが可能だ。通常はUltra DMAを有効にし、モードは自動判別させるよう設定する。するとOSが起動する直前、**画面2-3**のようにIDEハードディスクの検出状況が表示される（ここでPauseキーを押せばスクロールして消えるのを防げる）。このマザーボードはUltraDMA/66非対応なので、マザーボード上のIDEインターフェイスにつないだ2台のハードディスクは、**画面2-3**上のように「UDMA 33」、つまりUltra DMA/33としか認識されない。一方、

メーカー名	URL
IBM	http://www.storage.ibm.com/techsup/hddtech/welcome.htm
Maxtor	http://www.maxtor.com/library/main.html
Quantum	http://www.quantum.com/support/csr/software/csr_software.htm
Seagate Technology	http://www.seagate.com/support/disc/drivers/index.html
Western Digital	http://www.wdc.com/translate/jp/drives.html

表2-1　UltraDMA/66を有効／無効にするユーティリティ



画面2-3　IDEハードディスクの状態確認
上がAWARD BIOS、下がPromise製Ultra66カードのIDEデバイス検出結果。ここでIDEデバイスの状態が正しく表れないと、OSからも使えない可能性が高い。

ドライブの種別、容量、転送モードが確認できる

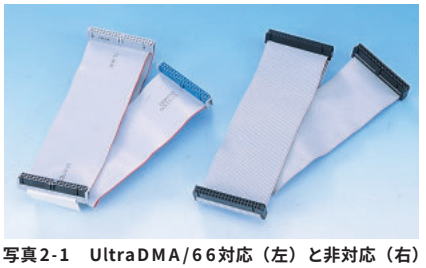


写真2-1　UltraDMA/66対応（左）と非対応（右）のケーブル
非対応ケーブルはケーブル中の信号線（40本）とコネクタのピンが1対1に接続されている。一方、UltraDMA/66対応ケーブルは、ケーブル中の信号線数を80本に増やし、うち半分をグラウンド（基準電位、0V）につなぐことで、残りの40本の信号線同士が干渉しあって信号の波形が乱れるのを防いでいる。

UltraDMA/66対応IDEカードであるPromise製Ultra66では、同じハードディスクが**画面2-3**下のように「Ultra DMA 4」、つまりUltraDMA/66と認識される。

IDEケーブルもUltraDMA/66専用のものが必要だ（**写真2-1**）。高い転送レートでも信号を確実に伝送するには、こうした「特殊」なケーブルを使わなければならない。UltraDMA/66対応IDEコントローラは、ケーブルが対応品かどうか検出できるので、非対応品を使うとUltraDMA/33以下のモードに変更してしまう場合がある。

UltraDMA/66対応IDEコントローラには、メーカー製のUltraDMA/66対応ドライバが用意されているのが普通だから、ほかのコンポーネントに比べると敷居は低い。ただしLinuxは例外で、131～133ページで解説しているような設定が必要である。また、BIOSでUltraDMA/66利用可能と設定するかあるいは自動検出されないと、OSレベルでもドライバの初期化で失敗してUltraDMA/66モードにならないことがある。

必要条件はまだある。UltraDMA/66対応ハードディスクのなかには、UltraDMA/66非対応のシステムでトラブルが生じないよう、出荷時にUltraDMA/33までしか利用できないようにされている製品もある（今回購入したなかではSeagate Technology製Barracuda ATAというハードディスクがそうだった）。これを解決するには、メーカーのWebサイト(**表2-1**)からUltraDMA/66を有効／無効にするユーティリティをダウンロードして設定し直すことだ。実行にはDOS環境が必要なのでLinuxユーザーは注意しよう。

IDEハードディスクを購入する

表2-2は、1999年12月時点で秋葉原のPCパーツショップ店頭に並んでいた主なIDEハードディスクの一覧である。これからIDEハードディスクを購入する際の参考にしていただきたい。備考に「一世代前」などと記してある製品でも、発売されたのは今年の後半からだったりするので、決して価値のない旧式というわけではない。ただ、IDEハードディスクの世代交代が非常に速いだけだ。世代間でそれほど性能が変わらないハードディスクもあるので、安価な旧モデルを購入するのもよいだろう。

最新モデルの性能については、136～138ページのベンチマークテスト結果を参考にしていただきたい。

製品名	型番	回転速度	記録容量	キャッシュ容量	備考
IBM					
Deskstar 25GP	DJNA-35xxx0	5400RPM	10.1G～15.2Gバイト	512Kバイト	DPTAより一世代前の価格重視モデル
			20.3G～25Gバイト	2Mバイト	
Deskstar 37GP	DPTA-35xxx0	5400RPM	15G～22.5Gバイト	512Kバイト	最新の価格重視モデル
			30G～37.5Gバイト	2Mバイト	
Deskstar 22GXP	DJNA-37xxx0	7200RPM	9.1G～22Gバイト	2Mバイト	DPTAより一世代前の性能重視モデル
Deskstar 34GXP	DPTA-37xxx0	7200RPM	13.6G～34.2Gバイト	2Mバイト	最新の性能重視モデル
Maxtor					
DiamondMax 36	9xxxxUx	5400RPM	13.6G～36.5Gバイト	2Mバイト	40より一世代前のメインストリーム・モデル
DiamondMax 40	9xxxxUx	5400RPM	10.2G～40.9Gバイト	2Mバイト	最新のメインストリーム・モデル
DiamondMax 6800	9xxxxUx	5400RPM	6.5G～27.2Gバイト	512Kバイト	40より二代前のメインストリーム・モデル
DiamondMax VL 17	9xxxxUx	5400RPM	4.3G～17.4Gバイト	512Kバイト	VL 20より一世代前の価格重視モデル
DiamondMax VL 20	9xxxxUx	5400RPM	10.2G～20.5Gバイト	512Kバイト	最新の価格重視モデル
DiamondMax Plus 6800	9xxxxUx	7200RPM	6.8G～27.3Gバイト	2Mバイト	6800の性能重視モデル
Quantum					
Fireball CX	FBCXxx.xAT	5400RPM	6.4G～20.4Gバイト	512Kバイト	lctより一世代前の5400RPMモデル
Fireball lct	FBLCTxx.xAT	5400RPM	4.3G～26Gバイト	512Kバイト	最新の価格重視モデル
Fireball Plus KA	FBKAxx.xAT	7200RPM	6.4G～18.2Gバイト	512Kバイト	Plus KXより一世代前の性能重視モデル
Fireball Plus KX	FBKXxx.xAT	7200RPM	6.8G～27.3Gバイト	512Kバイト	最新の性能重視モデル
Seagate Technology					
Medalist（UltraDMA/66）	ST3xxxxxA	5400RPM	8.4G～17.2Gバイト	512Kバイト	Barracuda ATA以前の性能重視モデル
U4	ST3xxxxxA	5400RPM	4.3G～8.4Gバイト	256Kバイト	U8より一世代前の価格重視モデル
U8	ST3xxxxxA	5400RPM	4.3G～17.2Gバイト	512Kバイト	最新の価格重視モデル
Barracuda ATA	ST3xxxxxA	7200RPM	10.2G～28Gバイト	512Kバイト	最新の性能重視モデル
Western Digital					
Caviar	WDxxxAA	5400RPM	4.3G～30.7Gバイト	2Mバイト	価格重視モデル
Expert	WDxxxBA	7200RPM	10.2G～27.3Gバイト	2Mバイト	性能重視モデル

表2-2　市販されている主なIDEハードディスク（1999年12月秋葉原にて調査）

高速ハードディスクをLinuxで使いこなす

最新のUltraDMA/66に対応したハードディスクを手に入れたなら、やはり転送モードは最速のUltraDMA/66で使いたいのが人情というものだろう。Linuxの場合、カーネル2.2以降ならUltraDMA/33までは標準的にサポートされており、多くのディストリビューションでは何も設定しなくてもUltraDMA/33でハードディスクを駆動できる。たとえばCeleron／Pentium II／IIIシステムにもっとも多く使われているIntel 440BXチップセットのIDEコントローラ（**写真3-1**）なら、そのままUltraDMA/33が利用できるはずだ。

しかし、ここで扱うPromise製Ultra66カード（**写真3-2**）をはじめとするUltraDMA/66に対応したドライバは、Linuxの開発版カーネル2.3.xから収録されており、2.2では標準サポートされていない。ただし、パッチをあてることで安定版カーネル2.2.xのままでUltraDMA/66に対応することはできる。ここでは、440BX搭載マザーボードのPCにUltraDMA/66対応ハードディスクを組み込んでLinuxをインストールした状態を想定し、そこからIDEコントローラをUltra66カードに差し替える方法を探ってみる。

UltraDMA/66対応がまだ標準リリースに取り込まれていない機能である以上、ある程度のリスクは覚悟しなければならない。少なくともパッチと共に配布されているドキュメント類はかならず目を通すようにしよう。

UltraDMA/66ドライバのインストール

Linuxに新しいドライバをインストールする場合、多くはカーネルのリコンパイルが必要なので、手順はちょっと面倒だ。このUltraDMA/66対応ドライバの場合、カーネルの設定も変更が必要になるので注意深く作業したい。

■カーネルとパッチの入手

今回用いたドライバは、カーネル2.2のブロックデバイスサポートを開発版カーネル2.3と同等にするUnified IDEパッチである。Promise Technologyも自社Webページにて、Ultra66カードのLinuxドライバ（ベータ版）を配布している。しかし、Unified IDEパッチのほうが、リリース予定のカーネル2.4により近いと判断して採用することにした。

カーネル2.2.13のソースコードはkernel.orgからダウンロードできる。Unified IDEパッチも同様にkernel.orgから入手可能だ。kernel.orgは日本のRingServerにミラーされているので、たとえば**表3-1**のring.gr.jpなどからダウンロードできる。

原稿執筆時における安定版カーネルはバージョン2.2.13が最新だったので、今回のテストではディストリビューション付属のカーネルではなく、バージョン2.2.13をインストールした。Unified IDEパッチはカーネルのバージョンごとに用意されているので、間違えずに2.2.13対応の最新版を選ぶこと。

■カーネルとパッチのインストール

ダウンロードが終わったら、カーネルを展開して、Unified IDEパッチを適用する。実際には**画面3-1**の手順で実行する。ここでは/tmpにLinuxカーネルおよびUnified IDEパッチをダウンロードしたものとしている。

■カーネルの設定変更

画面3-1の最後でmake xconfigを実行すると、カーネルの設定画面が表示される。**画面3-2**は、Unified IDEパッチをあてた後でカーネル設定の「Block

Ultra-DMA Mini-Howto	http://pobox.com/~brion/linux/Ultra-DMA.html
Unified IDEパッチ	ftp://ring.gr.jp/pub/linux/kernel.org/kernel/people/hedrick/
カーネル2.2.13	ftp://ring.gr.jp/pub/linux/kernel.org/kernel/v2.2/linux-2.2.13.tar.bz2
util-linux-2.10d.tar.gz	ftp://ring.gr.jp/pub/linux/kernel.org/utils/util-linux/util-linux-2.10d.tar.gz

表3-1　Linux IDEドライバのアーカイブファイルと情報

写真3-1　IDEコントローラチップのIntel PIIX4E
正確にはIDEコントローラだけではなく、USBインターフェイスなども集積されているチップで、Intel 440BXチップセットの一部だ。Ultra DMA/33対応のIDEインターフェイスを2ポート内蔵する。

写真3-2　IDEコントローラカードのPromise Ultra66（実売価格：約6000円）
UltraDMA/66対応のIDEインターフェイスを2ポート装備するPCIカード。コントローラチップは同社製PDC20262で、市販のマザーボードにはこれを装備してUltraDMA/66対応とした製品もある。

Device」セクションを表示したところだ。Unified IDEパッチを適用する前と比べると、ABIT ComputerのBP6マザーボードが搭載するHighPoint Technologies製IDEコントローラのHPT366や、Ultra66カードが採用するPromise製PDC20262チップなど、UltraDMA/66に対応したドライバが追加されていることがわかる。またExperimental（実験的）なサポートではあるが、VIA82CXXX（VIA Technologies製チップセット）などSuper 7マザーボードへの対応も始まっていることがわかる。

画面3-2でわかるように、各項目ごとに「y」「m」「n」の3つの選択肢（「y」と「n」のみの場合もある）が用意されている。「y」はドライバをカーネルイメージ自体に組み込む場合、「m」はドライバをモジュールとしてインストールする場合に選択する。今回のセッティングは**表3-2**のとおりだ。

表3-2では、必要なドライバをすべてカーネルに組み込んでいる。IDEドライバはモジュールにもできるが、ここではIDEハードディスク自身からLinuxを起動する予定なので、起動時にドライバが必要になる。そのためカーネルイメージに組み込むようにしている。またCMD 640やRZ1000（いずれもIDEコントローラの一種）のサポートを外したのは、この構成では必要ないことがわかっているからだ。

「PROMISE PDC20246/PDC20262 support」がUltra66カードのデバイスドライバなので、これは「y」に変更する。またCD-ROMドライブなどATAPIデバイスの接続にはマザーボード上のPIIX4Eを用いるので、「Intel PIIXn chipsets support」も「y」としておく。

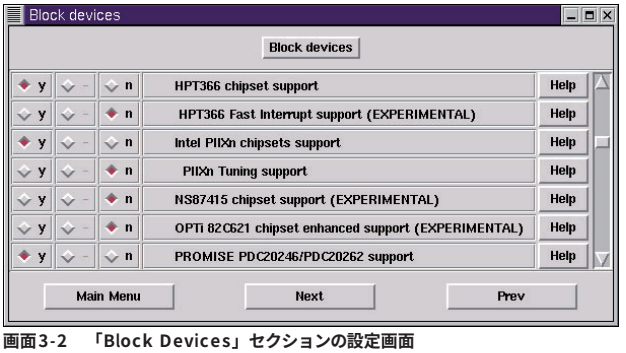
「Use DMA by default when available」は、Linuxの起動時にIDEのDMA転送（Ultra DMAモードを含む）を自動的に有効にするという設定だ。「y」にしておけば、後述のhdparmコマンドで起動後にいちいちUltra DMA/66を有効にしなくてもすむ。またUltra66カード以外のIDEコントローラのなかには、Unified IDEパッチで有効になるUltraDMA/66対応ドライバがあっても、hdparmコマンドではUltraDMA/66を有効にできないものもある。前述のHPT366もこの設定と「HPT366 chipset support」の両方を有効にしないといけない。

今回のテスト環境のように、マザーボード上のIDEコントローラ以外に接続されたハードディスクからLinuxを起動する場合、必ず「Boot off-board chipsets first support」は「y」にする。Ultra66カードなどのIDEコントローラからブートするには、これに加えてPCIデバイスの検出順序を変更するオプションをliloなどで指定する必要があるので注意したい（詳細は後述）。

今回用いたUnified IDEパッチは、「Block Device」以外のセクションには影響しないので、そのほかのセクションの設定はカーネル2.2.13標準と変わらない。ハードウェアの構成をよく調べたうえで、ほかのセクションの設定を修正されたい。

```
# cd /usr/src
# rm linux
# bzip2 -cd /tmp/linux-2.2.13.tar.bz2 | tar xvf -
# bzip2 -cd /tmp/ide.2.2.13.19991213.patch.bz2 | patch -p0
# mv linux linux-2.2.13
# ln -s linux-2.2.13 linux
# cd linux
# make distclean ; make xconfig
```

画面3-1　パッチをあてるためのコマンド実行手順



画面3-2　「Block Devices」セクションの設定画面

項目名	「y」	「m」	「n」
Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy	○		
Incluse IDE/ATA-2 DISK support	○		
Include IDE/ATAPI CDROM support	○		
Generic PCI IDE chipset support	○		
Generic PCI bus-master DMA support	○		
Use DMA by default when available	○		
Boot off-board chipsets first support	○		
Intel PIIXn chipsets support	○		
PROMISE PDC20246/PDC20262 support	○		
CMD 640 chipset bugfix/support			○
RZ1000 chipset bugfix/support			○

表3-2　「Block Devices」セクションで設定すべき項目

```
# cd /usr/src/linux
# make dep ; make bzImage
# make modules ; make modules_install
# cp arch/i386/boot/bzImage /boot/vmlinuz-udma66
# cp System.map /boot/System.map-udma66
# cd /boot
# rm System.map
# ln -s System.map-udma66 System.map
```

画面3-3　カーネルのコンパイルとインストール手順

■カーネルのリコンパイル

設定が終わったら「Save and Exit」を選択すると、カーネルコンフィグレーションファイル/usr/src/linux/.configが作成される。あとは**画面3-3**の手順でカーネルをコンパイルしてカーネルイメージを作成し、/bootディレクトリに必要なファイルをコピーする。

Ultra66カードからLinuxを起動する

以上の手順で、UltraDMA/66対応ドライバのインストールはほぼ完了だ。あとは新しいカーネルでLinuxを起動すればよい。

■新しいカーネルでの起動を試す

まずはハードウェアの構成を変えずに、新しいカーネルでのLinuxの起動を確かめてみる。それには/etc/lilo.confを**リスト3-1**のように書き替えてから/sbin/liloを実行する。再起動してliloのプロンプト（LILO boot:）が表示されたら、新しいカーネルイメージにつけた「linux-udma66」というラベル名（**リスト3-1**参照）を入力してLinuxを起動する。

これで問題なく新しいカーネルが動作したら、次はUltra66カードを装着して同じ手順でLinuxを起動する。正常ならdmesgコマンドの出力にUltra66カードが認識されたことを示すメッセージが残っているはずだ。

■Ultra66カードからLinuxを起動する

まずPIIX4Eに接続していたハードディスクを、すべてUltra66カードへつなぎ替える（UltraDMA/66対応の80芯ケーブルに交換するのを忘れずに）。またCD-ROMドライブなどATAPIデバイスはPIIX4Eのプライマリ IDEに接続しなおして、マスタ／スレーブのジャンパ設定も正しく変える。PCを起動したら、システムBIOSセットアップにてIDEハードディスクを無効にし、必要ならATAPIデバイスだけが認識されるよう設定する。これでもう一度起動しなおすとUltra66につないだハードディスクからブートするはずだ。liloのプロンプトが表示されたら、

```
linux-udma66 pci=reverse
```

と入力してLinuxを起動する。カーネルがパニックを生じずに無事起動したら、正しく設定できたということだ。

なお、CD-ROMドライブのデバイス名が変わるので、

```
# rm /dev/cdrom
# ln -s /dev/hde /dev/cdrom
```

と実行しておけば、これまで通りCD-ROMを扱える。CD-ROMドライブのデバイス名はdmesgコマンドの出力で確認できる（上記では/dev/hdeに変わったと想定している）。

問題なく動作したら、lilo.confのlinux-udma66ブロックに、

```
append="pci=reverse"
```

という1行を追加し、default行のラベルをlinux-udma66に書き替えよう。これでブートのたびにliloのプロンプトでラベルなどを入力せずにすむようになる。

▼リスト3-1　/etc/lilo.confの修正

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
default=linux
image=/boot/vmlinuz-2.2.5
  label=linux
  root=/dev/hda1
  read-only
image=/boot/vmlinuz-udma66
  label=linux-udma66
  root=/dev/hda1
  read-only
```

このブロックを追加する

Column

Linuxにおける8.4Gバイトの壁

Linuxにも大容量ハードディスクで注意が必要な容量の壁が存在する。よく知られているのは起動コード限界で、Linuxのカーネルイメージなどが存在する/bootディレクトリは、ブートディスクの先頭8.4Gバイトの範囲にないと、Linuxはブートできない。Windowsなどとマルチブートする場合は、Linuxのパーティションが8.4Gバイト以降に押しやられることが多いので注意したい。対策としては、Linuxのインストール時に/usrや/homeなど容量のかさむディレクトリだけ別のパーティションにし、/（ルート）のパーティションを8.4Gバイト以内に小さめのサイズで確保すればよい。

また比較的新しいディストリビューションでも、fdiskやcfdiskで8.4Gバイト以上の容量を認識できない場合がある。コマンドラインから、

```
# chsh -v
```

と実行して表示されるバージョンが2.10より低い場合は、fdisk／cfdiskが含まれるutil-linuxをアップデートする必要がある。Red Hat Linuxの場合、アップデート用FTPサイトからRPMやSRPMを入手できるが、LASER5 Linux 6.0には組み込めなかった。kernel.orgにあるアーカイブファイル（**表3-1**のutil-linux-2.10d.tar.gz）では問題なくコンパイル／インストールできたので、試してみるとよいだろう。

今回テストした最新IDEハードディスク

現在デスクトップPCは、インターネット接続端末ともとれる非常に安価なタイプと、ハイエンドの多機能・高速化追求タイプに二極化しつつある。こうした動向にあわせてIDEハードディスクのラインアップも、ローエンド（普及価格帯）とハイエンドの2シリーズからなるのが一般的だ。

今回テストしたIDEハードディスクは、各メーカーの普及価格帯またはハイエンドのシリーズのどちらかから1台ずつ、記録容量15Gバイト～21Gバイトの範囲で選んでみた。各製品紹介にあるアイコンのうち、「5400RPM」「7200RPM」が回転速度を意味しており、前者が普及価格帯で後者がハイエンド向けといえる。また、そのシリーズの最大容量をアイコン（～Gバイト）で記している。

■ジャンパ設定について

各製品紹介の下には、ジャンパの設定方法を記した。「マスタ（1台のみ）」はハードディスクを1台だけケーブルに接続する際の設定で、「マスタ（スレーブあり）」は2台接続するときのマスタ側の設定である。ただ、スレーブ側のハードディスクの仕様によっては、マスタ側で特別なジャンパ設定をする必要があり、それを「マスタ（スレーブあり2）」で記している。「ケーブルセレクト」とは、接続コネクタでマスタかスレーブかを選べる特殊なケーブルのための設定だ。「容量制限」は記録容量を2G／32Gバイトに制限するジャンパ設定で、マスタ／スレーブの設定と組み合わせて使う。ヘッド数を16から15に減らす「ヘッド数制限」も同様なので注意されたい。

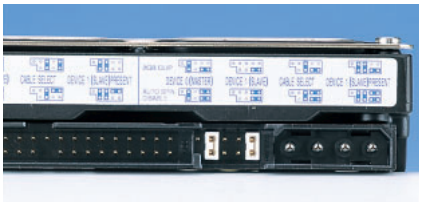
IBM

Deskstar 37GPシリーズ

実売価格：1万4500円前後（15.0Gバイト）
URL：http//www.jp.ibm.com/oemj/storage/

5400RPM　～37.5Gバイト

IBMのIDEハードディスクのラインアップは、回転速度7200RPMで性能重視のDeskstar xxGXPと、5400RPMでより安価なDeskstar xxGPに大別される。どちらも現在の秋葉原では流通量が多いほうで、人気のあるブランドだ。今回購入したのは、5400RPMのシリーズで最新のDeskstar 37GPのうち、15.0GバイトのDPTA-351500である。このシリーズでは、15.0G／22.5Gバイトのモデルは内蔵キャッシュメモリ容量が512Kバイトであるのに対し、それ以上の記録容量では2Mバイトに増量されており、記録容量とキャッシュ容量が連動しているので注意したい。



Maxtor

DiamondMax 40シリーズ

実売価格：　1万8700円前後（20.5Gバイト）
URL：http://www.maxtor.com/

5400PRM　～41.0Gバイト

Maxtorの最新IDEハードディスクは、ハイエンド向けのDiamondMax Plus 40、ミッドレンジのDiamondMax 40、そして普及価格帯のDiamondMax VL 20からなる。どのシリーズも1プラッター当たりの記録容量が10Gバイトを超える高密度を実現している。今回購入したのは、DiamondMax 40シリーズのうち20.5Gバイトの92049U4というモデルである。DiamondMaxシリーズでは、ヘッドの耐衝撃性を高めるEnhanced ShockBlockやデータ保護を強化するMaxSafeといった技術が投入されており、これらによってドライブの信頼性を高めているという。

Quantum

Fireball lctシリーズ

実売価格： 1万7100円前後（17.3Gバイト）
URL： http://www.quantum.co.jp/

5400RPM　~26.0Gバイト

QuantumのIDEハードディスクはFireballというブランドで統一されており、最新のラインナップはハイエンド向けのFireball Plus KXと普及価格帯のFireball lctからなる。今回購入したのはFireball lct 17.3ATという記録容量17.3Gバイトのモデルで、1プラッタあたりの容量は8.6Gバイトだ。すでに10.2Gバイト／プラッターのFireball lct10が発表されているため、現行のlctシリーズはlct08と呼ばれる場合があることに注意したい。Fireballシリーズでは、Quiet Drive Technologyという技術で騒音を抑えることも重視されている。

マスタ（1台のみ）	マスタ（スレーブあり1）
	
スレーブ	ケーブルセレクト
	
容量制限	マスタ（スレーブあり2）
なし	

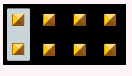
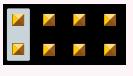
Seagate Technology

Barracuda ATAシリーズ

実売価格： 1万9600円前後（20.4Gバイト）
URL： http://www.seagate.com/

7200RPM　~28.0Gバイト

Seagate TechnologyのIDEハードディスクは、普及価格帯のU4／U8とミッドレンジのMedalist、そしてハイエンドのBarracuda ATAという3シリーズからなる。Barracuda（魚のかます）とは、同社が得意とするハイエンドSCSIハードディスクで長年培われてきたブランドであり、その高性能のイメージをIDEでも利用したい同社の意志が表れている。今回購入したのは20.4GバイトのST320430Aだ。1プラッターあたりの記録容量は6.8Gバイトとやや小さめだ。本機にはG-Force ProtectionやSeaShieldなど各種のドライブ保護技術が施されている。

マスタ（1台のみ）	マスタ（スレーブあり1）
	
スレーブ	ケーブルセレクト
	
容量制限	マスタ（スレーブあり2）
	

Western Digital

Expert BAシリーズ

実売価格： 2万2800円前後（20.5Gバイト）
URL： http://www.wdc.com/translate/jp/

7200RPM　~27.3Gバイト

Western DigitalのIDEハードディスクは、回転速度7200RPMでハイエンド向けのExpertシリーズと、回転速度5400RPMで普及価格帯のCaviarシリーズからなる。今回購入したのはExpert BAシリーズのうち、記録容量20.5GバイトのWD205BAだ。Expert BAの外観やスペックはIBM Deskstar 34GXPと極めてよく似ているが、メディア転送速度など細かい部分で違いが見られる。また、Western Digital独自のData Lifeguardと呼ばれるハードウェア機能とソフトウェアユーティリティを統合したデータ保護機能が提供されている。

マスタ（1台のみ）	マスタ（スレーブあり）
	
スレーブ	ケーブルセレクト
	
容量制限	
なし	

最新IDEハードディスクの性能を調べる

ここでは134～135ページで紹介した最新IDEハードディスクが、Linux環境でどれくらいのパフォーマンスを発揮するのか調べてみた。

ベンチマークテスト環境について

テストに用いたPCのハードウェア仕様は**表5-1**のとおりだ。Linux本体をインストールしたハードディスク（システムドライブ）は別途用意し、テスト対象のハードディスクにアクセスするのはベンチマークプログラムだけに限定している。IDEインターフェイスにはPromise製Ultra66カードを用い、2つのポートに両方のハードディスクを別々に接続した。Linuxのデバイス名としては、/dev/hdaがシステムドライブ、/dev/hdcがテスト対象のドライブに割り当てられる。またマザーボード上にあるPIIX4Eには、Ultra66カードと比較するときだけテスト対象のハードディスクを接続している。

テスト対象のハードディスクには、先頭から2000Mバイトでパーティションを設け、デフォルトのブロックサイズのまま、Linux標準のExt2でフォーマットしている。ベンチマークプログラムはそこにアクセスして速度を測定した。パーティションサイズが小さめなのは、比較のために用意した2Gバイトの古いIDEハードディスクに合わせたためである。

テスト用PCのソフトウェア環境については、131～133ページでUltra66カードをUltraDMA/66モードで稼働させたときの設定と同じで、カーネルは最新の安定版（2.2.13）をインストールした。また、ベンチマークプログラムの実行に必要ないデーモンプログラムは、なるべく停止させ、さらにEthernetカードのドライバも外している。

マザーボード	ASUSTeK Computer P2B-D
プロセッサ	Intel Celeron-433MHz×1
チップセット	Intel 440BX AGPset
メインメモリ	SDRAM 64Mバイト（100MHz）
テスト用IDEインターフェイス	Promise Ultra66（UltraDMA/66対応IDEカード）
ハードディスク（Linux本体用）	IDE 10Gバイト（IBM DTTA-351010）
SCSIホストアダプタ	Adaptec AHA-2940U2W（Ultra2 Wide SCSI）
グラフィックスカード	Matrox Millenium G400 AGP

表5-1　ベンチマークテストに用いたPCのハードウェア仕様

hdparm 3.6（SRPM）	ftp://rawhide.redhat.com/pub/rawhide/SRPMS/SRPMS/hdparm-3.6-1.src.rpm
hdparm 3.6（ソースコード）	ftp://metalab.unc.edu/pub/Linux/system/hardware/hdparm-3.6.tar.gz
Bonnie	http://www.textuality.com/bonnie/

表5-2　使用したアーカイブファイル

hdparmコマンドによるIDEの設定変更

Linuxでは、IDEハードディスクのパラメータを変更したり、ステータスを確認したりするのにhdparmコマンドが利用できる。ベンチマークテストの前に、まずhdparmについて学んでおこう。

■hdparmをアップデートする

現在のディストリビューションに付属のhdparmは、ほとんどがバージョン3.4だ。しかしこのバージョンではUltraDMA/66関連のステータスが表示できない。最新のバージョン3.6ならこの問題は解決されているので、hdparmをアップデートしてからテストしよう。

Red Hat Linux系列のディストリビューションなら、SRPMとしてhdparm-3.6-1.src.rpmを入手できるので、これからバイナリを作成するのがよいだろう。**表5-2**に記したサイトからダウンロードしたら、以下のようにしてhdparmをアップデートする

```
# rpm --rebuild hdparm-3.6-1.src.rpm
# rpm -Uvh /usr/src/Red Hat/RPMS/
i386/hdparm-3.6-1.i386.rpm
```

Debian GNU/Linuxでは、開発版Debian 2.2でバージョン3.6のhdparmが提供されているので、aptを用いたアップデートでコンパイル済みのバイナリを入手できる。Slackwareなどはhdparmのソース自体を入手してコンパイルするのがもっとも容易だろう。**表5-2**に記したサイトからhdparm-3.6.tar.gzをダウンロードしてコンパイルすればよい。

■hdparmの使い方

hdparmのオプションは多岐にわたるので詳細はmanを参照されたい。最もパフォーマンスに影響するのは転送モードの変更である。131～133ページで解説したように、Ultra66カードの場合、カーネル側でDMA転送を利用す

（なし）	設定済みのオプションを表示する
-i	ハードディスクからステータスを得る
-t	プラッターからの読み出し速度を測定する
-T	キャッシュからの読み出し速度を測定する
-d1	DMA転送を有効にする
-d0	DMA転送を無効にする
-d0 -X12	PIOモード4に設定する
-d1 -X66	UltraDMA/33モードに設定する
-d1 -X68	UltraDMA/66モードに設定する

表5-3　テストに使ったhdparmのオプション

るよう指定すれば、最適の転送モードが自動的で選択されるはずだ。しかし、まれにうまくいかないこともあるのでその場合はhdparmを用いる必要がある。hdparmでよく使うオプションを、各モードへの切り替え方法とともに**表5-3**に記す。今回のベンチマークテストも基本的にhdparmで転送モードを切り替えている。

Bonnieによるベンチマークテスト

今回ベンチマークに用いたのはBonnieというファイルシステムのベンチマークテストプログラムである。BonnieはPOSIX標準のC関数を用いてファイルシステムにアクセスし、その性能を転送レートとCPU占有率で示すプログラムだ。評価に用いたのは、Bonnieが出力する結果のうち、ディスク性能が大きく影響するブロック単位のシーケンシャル読み出し／書き込みとランダムシークだけである。

Bonnieは**表5-2**のサイトから入手できる。コンパイル自体は容易で、プログラムを最適化する-O2オプションをMakefileで指定しただけだ。実行ファイルが作成されたらcpコマンドで/usr/binなどにコピーすればよい。

以下のようにBonnieを実行すると、測定結果が標準出力に表れる。

```
# Bonnie -d /mnt/disktest
-m logname -s 256
```

lognameにはテストしたハードディスク名などを指定し、測定結果のログを区別するのに用いる。また-sオプションはテスト領域のサイズをMバイト単位で指定するためのものだ。ここでは、Linuxのディスクキャッシュによる影響を抑えるため、メインメモリの4倍である256Mバイトとした。また**グラフ**に掲載したのは、同条件でBonnieを3回実行して平均した値である。

各ハードディスクの性能を比較する

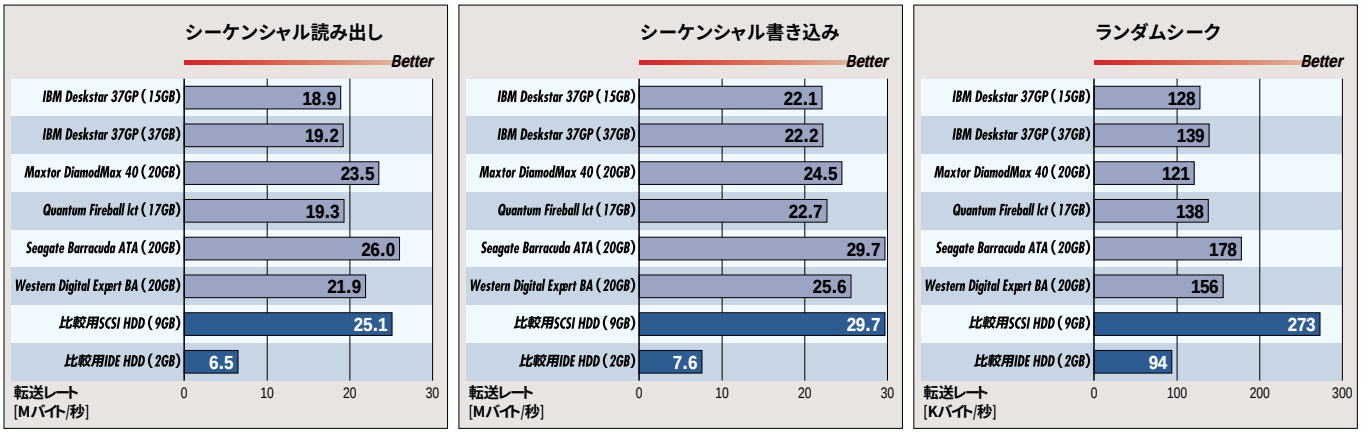
グラフ5-1は、134～135ページで紹介した最新IDEハードディスク5機種のほか、比較のために用意した次の3台のハードディスクを加えて、ベンチマークテストを行った結果である。「比較用IDE HDD（2GB）」は、3年近く前に購入したIBM Deskstar 3（DAQA-32160）という2GバイトのIDEハードディスクだ。回転速度は5400RPMでマルチワードDMAモード1まで対応している。特に性能重視ではないモデルだった。逆に「比較用SCSI HDD」はIBM Ultrastar 18LZXというハイエンドSCSIハードディスクを選んでいる（詳細は**写真5-1**参照）。そのほか、容量による性能差を調べるべく、Deskstar 37GPシリーズの37Gバイトモデルもテストに加えた。

■IDEではBarracuda ATAが最速

グラフ5-1からはっきり読みとれるのは、Seagate Barracuda ATAの良好な結果だ。同じ7200RPMのWestern Digital Expert BAと比べても明らかに速い。内部転送レート323Mビット/秒という高スペックは伊達ではないようだ。5400RPMのなかでは、Maxtor DiamondMax 40の健闘が光る。シーケンシャル読み出しでは7200RPMの

写真5-1　比較対象のSCSIシステム
ハードディスクはIBM製Ultrastar 18LZXの9.1Gバイトモデル、ホストアダプタはAdaptec製AHA-2940U2Wだ。SCSIバスの転送レートは最大80Mバイト/秒とUltraDMA/66より高速である。

グラフ5-1　各ハードディスクの性能比較



Expert BAより高速である。Barracuda ATAに続く295Mビット/秒という内部転送レートの高さが影響しているようだ。

シーケンシャル読み出しで奮わなかったExpert BAだが、ランダムシークは5400RPMのハードディスクに対して明らかに高い。ここに7200RPMであることの価値があるといえる。

■ハイエンドSCSIとの性能差はわずか！？

次に比較用SCSIハードディスク（Ultrastar 18LZX）と性能とコストを比べてみよう。**グラフ5-1**によるとBarracuda ATAとUltrastar 18LZXは、シーケンシャル読み出し／書き込みでほとんど差がなく、ランダムシークだけ前者が引き離されている。ランダムシークが高速なのは、Ultrastar 18LZXの回転速度が10000RPMと高く、磁気ヘッドの回転待ち時間も短いのが影響しているはずだ。

一方、コストはBarracuda ATAが20Gバイトで約2万円、Ultrastar 18LZXは9.1Gバイトで約4万6000円と段違いに高い。SCSIホストアダプタのコストも考慮すると、このクラスのSCSIハードディスクをデスクトップPCに導入するのは極めて割高と言わざるをえない。ランダムシークが多発する高負荷のサーバ環境ならば、コスト分の価値はあるかもしれないが。

■古いIDEハードディスクは交換すべき！？

古いIDEハードディスクの代表として用意したDeskstar 3は、やはり最新IDEハードディスクに比べてかなり性能が落ちることがわかる。たとえばBarracuda ATAとはシーケンシャル読み出しで約4倍も差が開いている。これだけ差が開くと体感でも違いが実感できるだろう。幸いUltra66カードなどを使えば多少古いPCでもUltraDMA/66に対応できるので、手頃なアップデートの選択肢として検討する価値は十分にある。

転送モードやインターフェイスの性能を比較する

IDEハードディスクでは最も高速だったBarracuda ATAを用いて、転送モードの違いによる性能差と、Ultra66カードとマザーボード上のPIIX4Eという2つのコントローラの性能差をそれぞれ探ってみた。その結果が**グラフ5-2**である（hdparmで転送モードを変更しながら測定した）。

■UltraDMA/66対応の優位は明らか！？

グラフ5-2のうち、Ultra66カードにおけるUltraDMA/66と同33を比べると、シーケンシャル読み出しでは約6Mバイト/秒と顕著な差が見られる。この性能差にはいろいろな解釈が可能で、Barracuda ATAがUltraDMA/66モードのみに最適化しているために生じた性能差とも推測できる。実際、グラフには記していないが、DiamondMax 40でもテストしたところ、その性能差は非常に小さかった。しかし、たとえそうだとしてもUltraDMA/66モードの優位性が揺らぐわけではない。

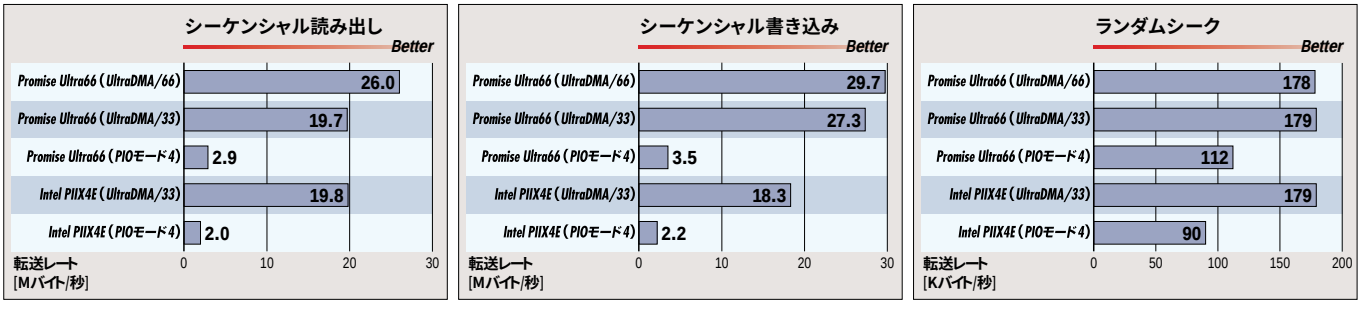
一方、PIOモード4ではどちらのIDEコントローラでも劇的に遅くなってしまった。これでは最新IDEハードディスクの性能をまったく発揮できない。もはやPIOモードは、DMA転送で何かトラブルが生じたときに安全な設定でシステムを起動する際に使われる程度の価値しかない。

■PIIX4Eは高性能なIDEコントローラか！？

グラフ5-2で次に注目されるのは、UltraDMA/33モードでのUltra66カードとPIIX4Eの性能差だ。シーケンシャル書き込みでは9Mバイト/秒と無視できない差が生じている。これはドライバを含むIDEコントローラの性能差によるものとみなせる。PIIX4Eが登場してすでに1年以上経つのに対して、Ultra66カードが搭載するPDC20262は比較的新しく、しかもIDE専用コントローラという点でも優位だ。たとえUltraDMA/33対応ハードディスクを接続する場合でも、PIIX4Eより性能の高い選択肢があることは憶えておきたい。

（127～138ページ：編集部）

グラフ5-2　IDEの転送モードとコントローラの性能比較

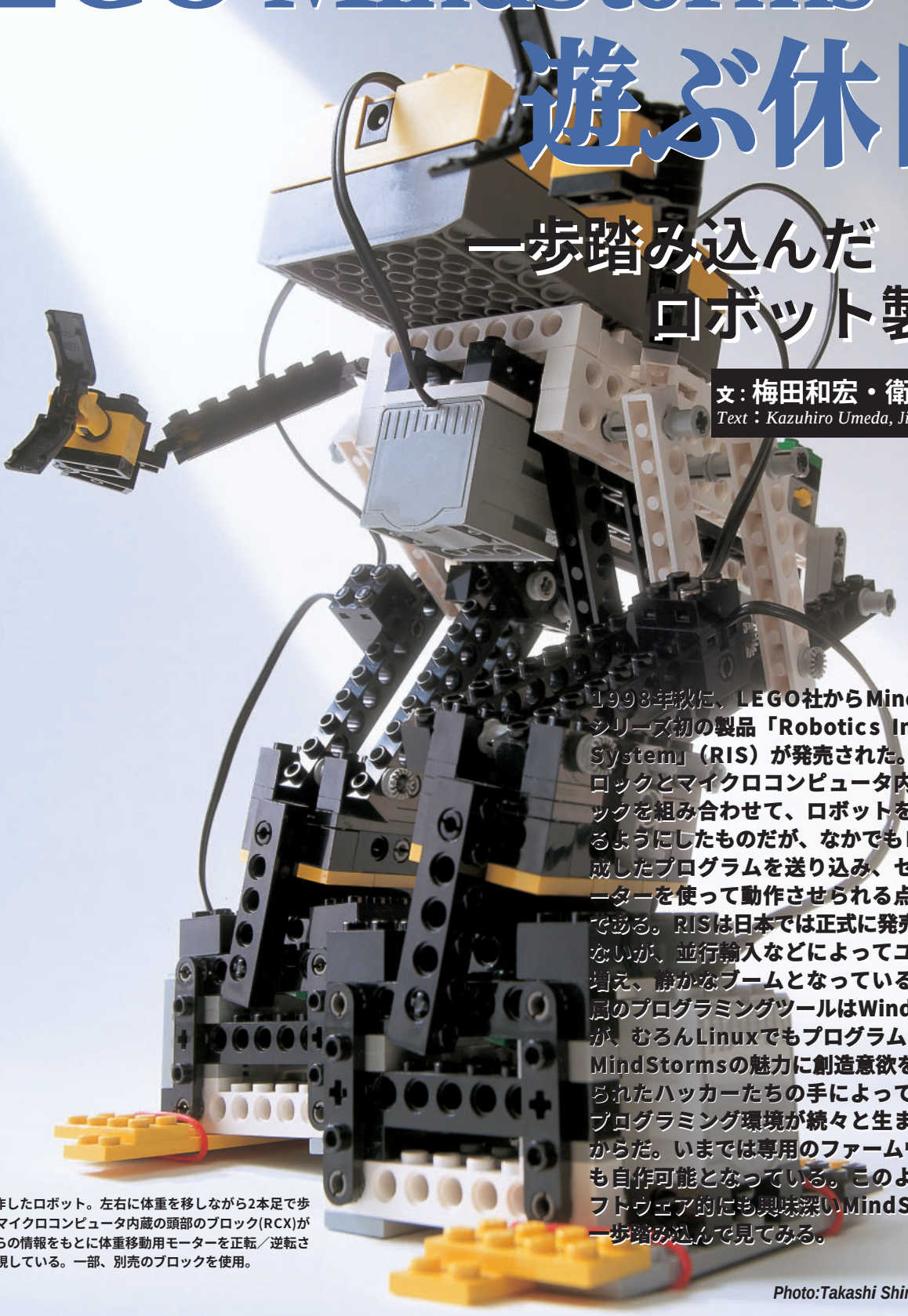


Linuxでも楽しめる

LEGO MindStormsで遊ぶ休日

一歩踏み込んだロボット製作

文：梅田和宏・衛藤仁郎
Text：Kazuhiro Umeda, Jiro Eto



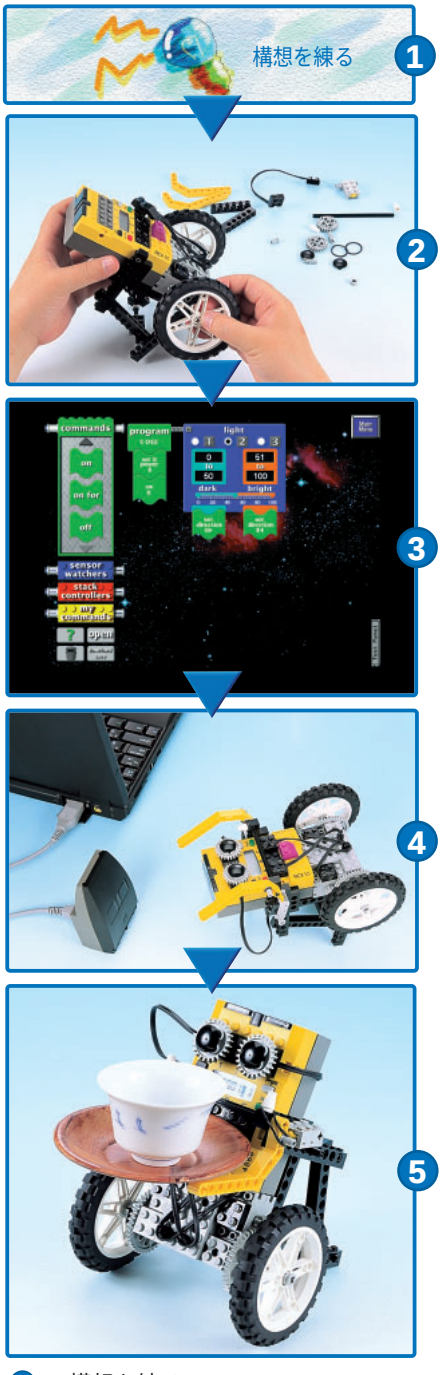
RISで製作したロボット。左右に体重を移しながら2本足で歩行する。マイクロコンピュータ内蔵の頭部のブロック(RCX)がセンサからの情報をもとに体重移動用モーターを正転／逆転させて、実現している。一部、別売のブロックを使用。

1998年秋に、LEGO社からMindStormsシリーズ初の製品「Robotics Invention System」（RIS）が発売された。従来のブロックとマイクロコンピュータ内蔵のブロックを組み合わせて、ロボットを作成できるようにしたものだが、なかでもPC上で作成したプログラムを送り込み、センサやモーターを使って動作させられる点は画期的である。RISは日本では正式に発売されていないが、並行輸入などによってユーザーが増え、静かなブームとなっている。製品付属のプログラミングツールはWindows用だが、むろんLinuxでもプログラムは可能だ。MindStormsの魅力に創造意欲をかきたてられたハッカーたちの手によって、新しいプログラミング環境が続々と生まれているからだ。いまでは専用のファームウェアOSも自作可能となっている。このように、ソフトウェア的にも興味深いMindStormsを一歩踏み込んで見てみる。

Photo:Takashi Shinohara (Dee)

MindStormsを知る

ロボットの製作手順



1……構想を練る
2……ブロックを組み立てる
3……プログラムを作成する
4……プログラムをRCXにダウンロード
5……RCXのRunボタンを押す

LEGOからMindStormsへ

LEGOというと四角いブロックを想像するかもしれないが、実はいくつかのシリーズがある。**図1**のLEGO SYSTEMシリーズは、ブロックを積み上げてモデルを作るので、モーターで駆動すると強度が足りず、バラバラになる。

そこでTechnicシリーズでは、側面に穴の空いたTechnicビームというブロックとペグでブロックを固定する方法が考え出された（**図2**）。この方法によってLEGOブロックでも頑強なフレームを構成できるようになり、ギア、シャフト、チェーン、モーター、エアシリンダと組み合わせてさまざまなメカを創り出せるようになった。

Technicシリーズにセンサとマイクロコンピュータ内蔵のブロック（RCX）を与えたものが、LEGO MindStromsシリーズである。MindStormsシリーズは現在、「Robotics Invention System（RIS）」「Robotics Discovery Set（RDS）」「Droid Developer Kit（DDK）」の3種類がある。このなかでもRISは最も高機能で自由度が高い。

ロボットを製作する

RISを使用したロボットの製作手順は、およそ次のようなものである。

1. **構想を練る**
   どんなものを作るかはアイデア次第だが、最初はマニュアルやCD-ROMのサンプルを作ってみるのがお勧めだ。
2. **ロボットを組み立てる**
   モーターやセンサ、ギヤのおおまかな配置を考えてブロックを組む。
3. **PCでプログラムを作成する**
   付属のRCXコードを使い、LEGOブロックを組み立てるような感じでコマンドブロックを並べる（左写真3）。
4. **プログラムをRCXに転送する**
   シリアルポートにつないだIRタワーから赤外線通信で送る（左写真4）。
5. **ロボットを動かす**
   RCXのRunボタンを押す。

残念ながらRCXコードが使用できるのはWindows95／98のみだが、後述するように、LinuxでもRISのプログラミングを楽しめる。

パーツを観察する

RISのパッケージには、RCX、ブロックやギア、タイヤなどのパーツ、プログラムをロボットに送り込むのに使うIRタワーなど、727個の部品が入っ

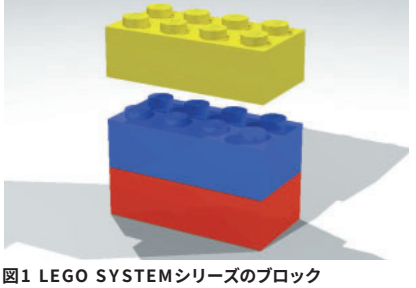


図1 LEGO SYSTEMシリーズのブロック

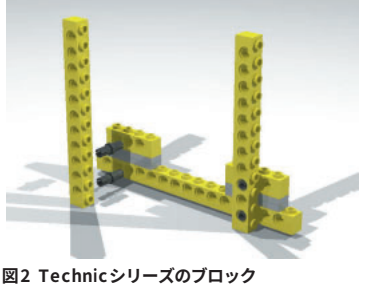


図2 Technicシリーズのブロック

ている（**写真2**）。また、付属のCD-ROMには、RIS標準のプログラム開発環境「RCXコード」やムービーを使用したわかりやすいチュートリアルが収容されている。部品のほとんどはLEGO Technic用に作られたものなので、ロボットの製作過程で部品が不足しても、LEGO SYSTEMのブロックやLEGO Technicの部品を追加購入できる。

RISの頭脳ともいえるのは、RCXである（**写真1**、**表1**）。PCで作成したプログラムを送り込み、つないだセンサとモーターを制御してロボットを動作できる。

新しいプログラミング環境

以上、RISの概要について簡単に解説した。RISにはいろいろな魅力があるが、今回は一歩踏み込んだソフトウェア開発を紹介することにする。

RCXコードは、MindStormsの持つフィロソフィを体現した重要な要素で、C言語バリバリのLinuxユーザーにも、本当はぜひ体験してもらいたいと思っている。しかし筆者はふだん、「RCXコードより優れている点があるから」もしくは「RCXコードでは速度的、機能的に間に合わないので」C言語ふうのNQCやlegOSを使用している。どちらもLinuxで使用可能である。

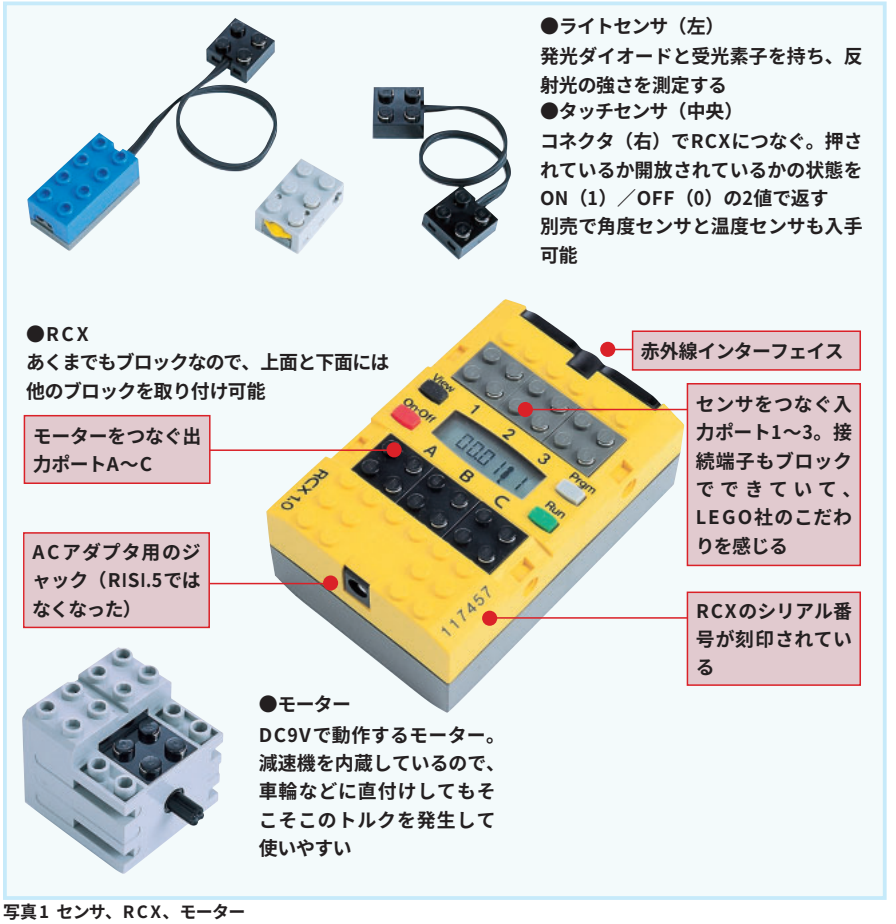


写真1 センサ、RCX、モーター

写真2　RISのパッケージ
ほかにCD-ROM（RCXコードとチュートリアルを収容）とマニュアル、ロボットのテスト走行に使えるテストパッドが付属

CPU	H8/3292（日立製8ビットマイコン） 動作モードは16MHz/5V 10ビットA/Dコンバータ内蔵
ROM	16Kバイト
RAM	32Kバイト
入出力	入力ポート×3 出力ポート×3 赤外線通信ポート 液晶ディスプレイ
赤外線通信	38kHz変調/2400bps
外形寸法(mm)	63（W）×95（D）×40（H）

表1 RCXの仕様

NQC—チャレンジャーのプログラミング環境

Linuxでも OK!

MindStormsが気に入った世界中のハッカーたちは、すぐにRCXコードの限界に気づき、このカラフルなプログラミング環境ではもの足りなくなった。そこで、RCXを解析して多くの新しいツールを作成したのである。そのなかでもNQCは特にユーザーが多いと思われる。

NQCについて

NQC（Not Quite C）はDave Baum氏によって開発されたRCX専用のプログラム言語で、MPL（Mozilla Public License）に基づいて配付されているフリーソフトウェアである（http://www.enteract.com/~dbaum/nqc/）。“Not Quite C”は「C言語とまったく同じではないが、それに近い言語」といったところであろうか。

RCXコードが生成するプログラムは、ハードウェアでは直接実行できないバイトコード（中間コード）であり、RCXのファームウェアによって解釈・実行される。バイトコードは発売後まもなく詳しく解析されたので、新しい開発環境の多くはこのバイトコードを生成するようになっている。NQCもそのひとつである。

NQCの特徴は次のとおりである。

- C言語ふうの言語仕様
- LEGO社純正ファームウェアで動作
- マルチタスクに対応
- Linux／Windows／Macintoshに対応
- RCXコードより高速
- RCXコードよりもRCXの機能を引き出せる

もう少し詳しく見ていこう。

■C言語ふうの言語仕様

まずは、プログラムリストを見るのが早いだろう。リスト1はNQCのプログラム例だ。

▼リスト1 エッジ探索虫のプログラム（NQC用）

```
#define TH       700                          // しきい値の設定

task main()
{
SetSensorType(SENSOR_2, SENSOR_TYPE_LIGHT); // 入力2ポートにライトセンサを接続
SetSensorMode(SENSOR_2, SENSOR_MODE_RAW);   // 入力値の取りうる範囲は0〜1023
SetPower(OUT_B, 7);                         // Bポートの出力を最大にする

  while(true)                  // 無限ループ
  {
    if (SENSOR_2 < TH)         // センサ入力値がしきい値より小さければ
    {
      OnFwd(OUT_B);            // モータを正転
    }
    else
    {
      OnRev(OUT_B);            // モータを逆転
    }
  }
}
```

このプログラムは、白と黒の境界をライトセンサで探索する。どうだろう？　C言語の知識があれば容易に動作が推測できたのではないだろうか。このように、NQCはC言語に非常によく似ていて、ひらめいたアルゴリズムを手早くプログラムできるため、RCXコードを使えないWindows以外のユーザーのみならず、C言語経験者に愛用されている。

■純正のファームウェアを使用する

NQCで作成したプログラムはRCXコードと同様にLEGO社の純正ファームウェアで動作するので、サウンド機能や通信機能など、ファームウェアが備えているすべてのサービスが利用可能であり、そのための関数が多数用意されている。ユーザーはこれらの関数を呼び出すだけで手軽にサービスを享受できる。「お手軽」にという点は、NQCのとても重要なポイントである。

■マルチタスクをサポート

NQCはmain()を含め最大10個までのマルチタスク処理が可能である。ひとつの処理をしている最中でも、並行して別の処理を実行できる。

たとえば、「10秒前進したら左折を4回繰り返して元の位置に戻ってくる」プログラムを「動作中に何か障害物にぶつかって（タッチセンサが押されたら）その場で停止する」ようにするには、「センサを常に監視してセンサが検知したら停止するプログラム」を別タスクで追加して並列に動作させれば、簡単に実現できる。

■マルチプラットフォーム対応

NQCは、Linux、Windows95／98／NT4.0、Macintoshで動作する。このため、Windows以外のユーザーでもRCXのプログラムを作成できるようになった。

■標準のRCXコードより高速

同じ動作をするプログラムをRCXコードとNQCで作成して比べてみると、明らかにNQCのほうが速い。NQCで生成するコードはRCXコードと同じもので、潜在的な能力としては両者とも同じはずなので、RCXコードはファームウェアの能力を限界までは使用しておらず、かなりのマージンをみているのではないかと推測される。

エッジ探索虫を作る

以上、簡単にNQCの特徴について述べたところで、実際にメカを作ってNQCの使い方を説明しよう。

■テストメカを組み立てる

まず、**リスト1**のプログラムで遊べる簡単なテストメカを考えてみた（**写真3、4**）。モーターをRCXのポートBに、ライトセンサをポート2につなぐ。使用するパーツはわずか18なので、RISを持っていたらぜひ作ってほしい。

■プログラムを作成する

ここではリスト1のプログラムを使用する。好みのエディタで記述したら、list1.nqcという名前で保存しよう。

■コンパイルとダウンロード

さてこれからNQCを使用するのだが、Linuxの場合は145ページのコラムを参照してインストールしておこう。ここではLinuxシステムでの手順を示すが、Windows環境ならMark Overmars氏によるRcxCC（RCX Command Center）というGUIツールを使うのが便利だろう（http://www.cs.uu.nl/people/markov/lego/）。どちらでも作成するプログラムや実行時の動作は同じになる。

コンパイルをするには、コンソールから次のコマンドを実行する。

```
% nqc -d list1.nqc
```

-dオプションを指定すれば、コンパイルとダウンロードがまとめてできる。

■動かしてみる

右が白、左が黒の紙を用意し、この境界あたりにテストメカを置く。RCXのRunボタンを押す前に黒のViewボタンを2回押してポート2の入力値を液晶ディスプレイに表示させよう。紙の黒い部分にライトセンサを持ってくると700以上になり、白の上では700未満になることを確認する。そうでなかったらプログラムの1行目、

```
#define TH  700   // しきい値の設定
```

の“700”を2つの値の中間に変更して、nqcコマンドを再度実行する。これで準備は完了だ。モーターをしっかり固定して緑色のRunボタンを押そう。

ライトセンサが白を検知すると左に、黒を検知すると右に首を振る。常に白と黒の境目を探索するように高速で首を振り続ければ成功だ。ちゃんと動いたら、メカの位置を左右に変えてみよう。センサのヘッドは常に境界線を探し続ける。うまく動かない場合はしきい値の値、コネクタの方向を確認していただきたい。このテストメカでは「先端は白の上か黒の上か」という情報がモーターの回転方向の制御にフィードバックされている。

ちなみにこのテストメカ、モーターから手を放すとくねくねと虫のように動く。紙のほうを動かすとちゃんとエッジを追いかけてくるのでおもしろい。

NQCの言語仕様

NQCの言語仕様はその名前が示す通りC言語にとてもよく似ている。そこでC言語と比較した場合の、NQCのお

写真3　エッジ探索虫の部品

写真4　エッジ探索虫

もな制限事項をまとめてみた。

- ●構造体がない
- ●ポインタが使用できない
- ●変数は32個まで
- ●すべての値がsignedのint型（16ビット -32768〜32767）で取り扱われる
- ●関数はvoid型で、返り値がない
- ●サブルーチンの数は8つまで

これらの制限事項が“Not Quite C”と呼ばれる所以である。32Kバイトという限られたメモリでは、しかたのないことかもしれない。逆にCと同様に、コメント／関数（引数可）／サブルーチン／グローバル変数／ローカル変数／演算子（論理演算子、ビット演算子を含む）／制御文（if、else、while、do while)、プリプロセッサ（#include、#define、#ifdef、#ifndef、#if、#elif、#else、#endif、#undef）などが利用できる。

また、RCX専用の言語なので制御用の関数が多数用意されている。

- ●モーター制御用―SetPower()、OnFwd()、OnRev()など
- ●センサ設定用―SetSensorMode()、SetSensorType()など
- ●サウンド制御用―PlaySound()、PlayTone()など
- ●メッセージ通信用―ClearMessage()、SendMessage()
- ●データロギング用―CreateDatalog()、AddToDatalog()

リスト1のSetPower(OUT_B, 7)は「ポートBの出力を最大にセット」し、OnFwd(OUT_B)は「ポートBのモーターを正転方向に回す」コマンドである。また、SetSensorType(SENSOR_2, SENSOR_TYPE_LIGHT)はポート2に接続されたセンサがライトセンサであることを宣言し、SetSensorMode(SENSOR_2, SENSOR_MODE_RAW)はポート2に接続されたセンサからのデータをrawモード（10ビットA/D変換による0〜1023の値）で取得することを表している。RCXの入力ポートはどのセンサでも取り付けられるので、どんなセンサがつながっているかをプログラムで最初に宣言してやる必要がある。RCXはそれにもとづき、ライトセンサであればセンサに電源を供給し、発光ダイオードが光るようにする。

通信機能を使う

さて、RCXは赤外線インターフェイスを使用してPCや他のRCXと通信し合うことができる。この機能を使えば、何台かのロボットが協調して動いたり、何台かのRCXを組み合わせて複雑な動作をする1台のロボットを作成できる。そこで、メッセージ通信やデータロギングを使用した簡単な例をもうひとつ紹介しよう。**リスト2**は、複数のRCXがそれぞれ乱数を振り出し、その数の大きで勝負するゲームである。

▼リスト2　RCX間通信プログラム例

```
task main()
{
  ClearMessage(); // メッセージ記憶領域をクリア
  Wait(200);            // すべてのロボットがRUNされるまで待機
  Wait(Random(400));    // 0〜4秒の範囲でランダムな時間待機
  if (Message() > 0)    // 待ち時間にほかのロボットから通信があった
  {
    start slave;  // 自分はスレーブであることを認識し処理開始
  }
  else                    // 待ち時間に誰からも通信がなかった
  {
    SendMessage(1);     // 自分がマスタになることを他のロボットに宣言
    Wait(400);          // 他のロボットのスレーブ処理開始を待つ
    start master;       // マスタ処理開始
  }
}

task master()           // マスタ処理関数
{
  int  rnd1;
  while(true)
  {
    rnd1 = Random(2) + 1;
    SendMessage(rnd1);   // 1〜3までの乱数を送信
    Wait(200);
  }
}

task slave()            // スレーブ処理関数
{
  int rnd2;
  CreateDatalog(100);    // データログを初期化

  while (true)
  {
    rnd2 = Random(2) + 1;
    ClearMessage();
    until (Message() != 0);     // マスタのメッセージを待つ
    if ((Message() - rnd2) == 0)     // あいこの場合
    {PlaySound(0);AddToDatalog(0);}  // 音を鳴らして0を記録
    if ((Message() - rnd2) <  0)     // スレーブの勝ち
    {PlaySound(1);AddToDatalog(-1);} // 音を鳴らして-1を記録
    if ((Message() - rnd2) >  0)     // マスタの勝ち
    {PlaySound(2);AddToDatalog(1);}  // 音を鳴らして1を記録
  }
}
```

まず、一方のRCX（マスタ）からもう一方のRCX（スレーブ）に赤外線通信を用いて乱数を送信する。スレーブは受信した乱数と自らが発生した乱数とを比較してその結果を音で知らせ、メモリのログに対戦結果を追加する。

どちらのRCXでも同じプログラムですむように、マスタ／スレーブは乱数を用いて最初に自動的に決定する方式とした（コリジョンでデータが消失したときには一定時間を置いて再送すべきだが、リストを簡略にするため省略した）。何回か対戦したらRunボタンを押してプログラムを停止し、ログをPCに転送する。スレーブのRCXをIRタワーと向き合わせ、PC側で**リスト3**のようにコマンドを実行すればRCX内に蓄えられた対戦結果がPCに転送される。

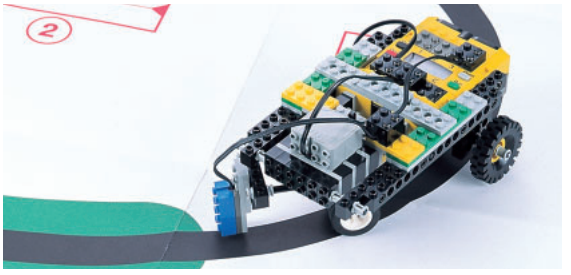


写真5　NQCの応用例―エッジ君
黒い楕円コースを高速に周回する。「MindStorms情報局」（http://www.mi-ra-i.com/JinSato/MindStorms/）がインターネット上で開催している「お気軽コンテスト」の周回タイムトライアル競技（1999年春開催）用に製作した。テストメカとまったく同じ原理のステアリング機構を搭載している。
(http://www.bea.hi-ho.ne.jp/meeco/)

もちろん、PC側でプログラムを作ればこの過程も自動化可能である。

NQCの機能は多彩でまだまだ説明したいことがあるが、誌面の都合上これ以上の解説はほかに譲る。ドキュメントの整備が進んでいるのもNQCの特徴で、そのなかでもMark Overmars氏のチュートリアルはとても完成度が高い。Alberto Palacios Pawlovsky氏によって日本語に翻訳されているので、一読をお勧めする（http://www.cc.toin.ac.jp/tech/sysd/ft07/NQCj.html）。

▼リスト3　対戦結果をPCに送る

```
% nqc -datalog
Uploading Datalog..
0
-1
1
-1
-1
0
```

Column

Linux上にNQC環境を作成する

①NQCをダウンロードする

まずは、NQCをhttp://www.enteract.com/~dbaum/nqc/からダウンロードする。現時点の最新バージョンはVer2.0.2である。Linux版のバイナリ（nqc_lnx_2_0_2.tar.gz）もあるが、動作しない場合を考えてソース（nqc_src_2_0_2.tar.gz）をダウンロードしておこう。

②起動をチェックする

アーカイブファイルを展開してパスを設定したら、コンソールから“nqc”とだけ入力してみる。NQCの説明が表示されればOKだ。エラーが出る場合には、makeし直す必要がある。ちなみにLaser5 Linux 6.0ではバイナリのままで動作したが、日本語Red Hat Linux 5.2ではmakeが必要であった。

③ファームウェアをダウンロードする

ファームウェアはRCXコードを初回に使用したときにPCからRCXにダウンロードされるが、Linux環境からRCXへ手動で送り込むことも可能だ。RCXの電源を入れてディスプレイに00.00と表示されれば、ファームウェアは転送済みでこの作業は不要である。

まず、付属CD-ROMのfirm/firm0309.lgoファイルをNQCと同じディレクトリにコピーする。次にIRタワーをシリアルポートに接続し（デフォルトのポートは/dev/ttyS0）、RCXと向かい合わせて以下のコマンドを実行する。

```
% nqc -firmware firm0309.lgo
```

■NQCコマンドの使い方

```
nqc [Options] [Actions] [Filename] [Actions]
```

Options:

-1	:NQC 1.x互換モード
-d	:プログラムをコンパイル後ダウンロード
-E[*filename*]	:コンパイルエラーをファイルかstdoutに出力
-I*path*	:インクルードファイルを指定されたパスで検索
-L[*filename*]	:生成したコードリストをファイルかstdoutに出力
-O*outfile*	:ファイルへ出力
-S*portname*	:シリアルポートを指定

Actions:

-run	:プログラムを起動
-pgm *number*	:プログラム番号を選択
-datalog|-datalog_full	:データログをPCへアップロード
-near	:IRタワーを近距離モードにセット
-far	:IRタワーを遠距離モードにセット
-watch *time*|now	:RCXに時間をセット
-firmware *filename*	:ファームウェアをダウンロード
-sleep *timeout*	:RCXの自動電源OFF時間をセット
-msg *number*	:RCXへ向けてメッセージを送信
-clear	:RCX内のプログラムとデータログを消去

legOS—自分専用のファームウェアOSを作る

NQCを使いこなすにつれて「ロボットをもっと高速にしたい」という欲が出てくる。たとえばNQCの応用例として挙げた「エッジくん」（**写真5**）は、車速を上げると制御が間に合わなくなってコースアウトしてしまう。これ以上の高速化は、純正ファームウェアでは限界があるのだ。できないことは、できるようにしてしまえ、ということでハードウェアを直接叩けるファームウェアOSを自作した人たちがいる。

legOSについて

legOSは、Markus L. Noga氏らのグループによって開発されたRCX用のオペレーティングシステムであり、MPL（Mozilla Public Licens）にもとづいてライセンスフリーで配布されている（http://www.noga.de/legOS/）。この解説は、1999年11月現在の最新バージョンlegOS-0.2.2をもとに書いている。蛇足ながら、Markus L. Noga氏の開発メインプラットフォームはLinuxである。

RISにはもともとLEGO社純正のファームウェア（Firm0309.lgo）が付属しており、RCXに電源を投入するといつもこのプログラムが動作する。ファームウェアはデバイス（モーター、センサ、赤外線通信など）とユーザーインターフェイス（ボタン操作、プログラムのダウンロードなど）を制御する。RCXコードや前述のNQC言語で記述されたロボットプログラムは、純正ファームウェアの上で動作している。

legOSはこの純正ファームウェアの代替となるOSである。本誌の読者なら、お仕着せのものではなく、自分で使いやすいように不要なものを除いてカーネルを再構築したことがあるかもしれない。同様に、legOSユーザーは自分でファームウェアをコンパイルして作成し、RCXの純正ファームウェアと置き換える。自分用にmakeするのだから、当然必要な機能だけを搭載した軽量なカーネルになる。legOSを使用する目的のひとつは、このように必要な機能に特化したカーネルでRCXを動かすことにある。legOSを使用してAV機器のリモコン機能を実現した例もあり（http://www02.so-net.ne.jp/~abuku/）、純正ファームウェアの壁を飛び越えた応用が期待できる。

特徴

legOSの特徴を簡単に列挙すると、

1. **タイムスライス方式のプリエンプティブマルチタスク**
2. **POSIX標準1003.1b準拠の計数セマフォ**
3. **赤外線通信を用いたネットワークプロトコルLNP（LegOS-Network-Protocol）**
4. **ファームウェアの高速ダウンロードユーティリティが付属**

といったところだが、1. および2. を実現するためには動的なメモリの確保／解放やタスク間の排他制御などが必要であり、メモリマネージャやタスクマネージャは不可欠である。また、3. を実現するために階層的プログラミングモデルの導入が想像され、legOSがかなり本格的なOSであることがわかる。とはいえ、RCXの搭載メモリは32Kバイトと、PCと比較すると非常に狭いので、カーネルもきわめて小さい。純正ファームウェアのダウンロード前のファイルサイズが44Kバイトなのに対し、デフォルトでmakeしたlegOSカーネルは約3分の1の16Kバイトにすぎず、たいへん軽快に動作する。

開発環境

legOSはC言語とわずかなアセンブラコードで記述されている。Linuxのカーネルをmakeする場合、/usr/src/linux/.configを編集して構成要素やオプションを選ぶが、それと同様にlegOS/boot/config.hで必要な機能を定義してmakeする。

```
    :
#define CONF_TIME      // システムタイム
#define CONF_MM        // メモリマネジメント
#define CONF_TM        // タスクマネジメント
#define CONF_SEMAPHORES    // セマフォ
#define CONF_PROGRAM  // プログラムローダ
    :
```

VxWorksなど、組込み機器用のOSを使用したことがあれば、おなじみの作業だろう。

コンパイル作業はLinux上で行い、コンパイラにはgccを用いる。といっても、Linuxのディストリビューションに付属しているgccではない。カーネルはRCXに搭載しているH8チップで動作させるから、H8用のgccを用いるのである。このように、コンパイル環境（この場合Linux）とは別の環境で動作するプログラム（legOSカーネル）を生成する開発環境をクロス開発環境と呼び、そのコンパイラをクロスコンパイラと呼

んでいる。H8用のクロスコンパイラは、http://www.noga.de/legOS からバイナリを入手できるが、gccのソースコードから自分で作成したほうが確実だろう。legOS開発環境の構築方法は、http://www.ylw.mmtr.or.jp/~arcadia/やhttp://arthurdent.dorm.duke.edu/legos/HOWTO/HOWTO.htmlを参照していただきたい。

makeしたカーネル（legOS.srecファイル）は、util/のfirmdl3ユーティリティでIRタワーからRCXにダウンロードする。firmdl3はダウンロードが非常に高速なため、巨大な純正ファームウェアのダウンロードにもお勧めできる。

ユーザープログラムはC言語で記述してmakeする。以前のlegOSでは、OSと一緒に構成し、ファームウェアの一部としてRCXにダウンロードしていたが、legOS-0.2.x以降はユーザープログラムを独立させ、RCXコードのようにダウンロードできるようになった。makeしたプログラムはカーネルのあとから最大8個までダウンロード／消去可能だ。ダウンロードしたプログラムは、Prgmボタンで選択してRunボタンで実行開始／停止できる。

legOSのタスクマネージメント

ここで、legOSのマルチタスキング方式について少し解説する。legOSはタイムスライス方式のプリエンプティブなマルチタスクを採用している。タイムスライス方式は、一定時間（タイムスライス）ごとにタスクスケジューラが起動して、どのタスクを実行させるか判断して実行させる。

タイムスライスの間隔はデフォルトで20msecであり、include/sys/time.hに定義されている。legOSユーザーは、もちろんこの値を自由に変更できる。

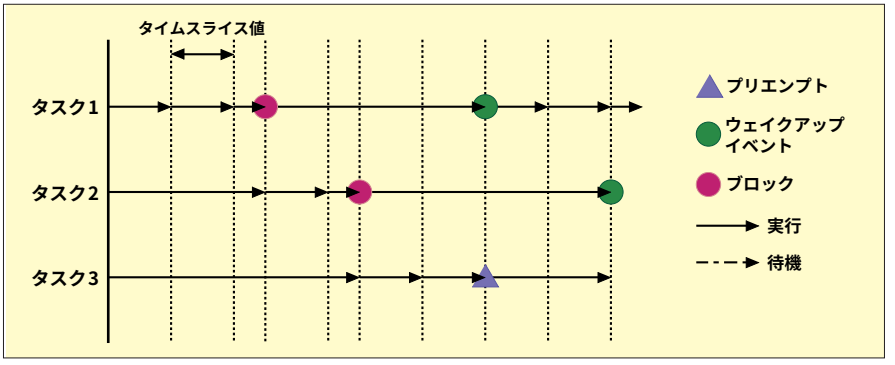


図3　legOSのタスクスケジューリング

タスクが1つだけでイベント待ちがなく、できるだけ高速な処理が必要な場合にはタイムスライス値を大きく設定し、複数タスクで処理するときにタスクスイッチのオーバヘッドが増えてでも高速なタスクスイッチが必要であればタイムスライス値を小さく設定する。

図3に優先順位の高い順にtask1、task2、task3が存在する場合のタスクの実行状態を示した。優先順位の高いタスクがwait()のコールなどでSLEEP状態になると、即座にタスクスケジューラが起動されて次に優先順位の高いタスクが実行状態になる。優先順位の低いタスクが実行状態にあるとき優先順位の高いタスクが待っていたイベント（タッチセンサが押された、光センサが明るくなった、など自由に設定可能）が発生すると、実行権の横取り（プリエンプション）によって優先順位の高いタスクが実行状態になる。

このように複数のタスクが代わるがわるCPUなどの資源を共有して実行する環境では、タスク間の排他制御が必須である。legOSはセマフォを提供しており、ユーザーはこれを用いてクリティカルセクションを管理できる。たとえば、task1とtask2が以下のようにグローバル変数criticalDataを使用すると、criticalDataの整合性は保証されない（task1がcriticalData < 20を評価したとたんにtask2に実行が移る場合を考慮しなくてはならない）。

```
task1
if (criticalData < 20)
   motor_a_dir(rev); /* モーター後退 */
else
   motor_a_dir(fwd); /* モーター前進 */

task2
criticalData = 30;
```

しかし次のようにセマフォを使用すれば、task1が①でsem_postするまでtask2は②のsem_waitで実行がブロックされるため、安全にデータ（クリティカルセクション）を扱える。

```
task1
  sem_wait(&semaphore);
  if (criticalData < 20)
     motor_a_dir(rev); /* モーター後退 */
  else
     motor_a_dir(fwd); /* モーター前進 */
  sem_post(&semaphore); ……①

task2
  sem_wait(&semaphore); ……②
  criticalData = 30;
  sem_post(&semaphore);
```

通信機能

LNP（Legos Network Protocol）は、赤外線通信を用いてTCP/IPのよ

うなコネクション指向と、UDP/IPのようなブロードキャスト特性の通信を可能にする。IPアドレスに相当するLNPアドレスは4ビットで0～15が使用でき、RCXが15台とホストPC1台でネットワークが構成できる。論理的なポート番号は4ビットで0～15ポートが使用できるので、それぞれのRCXで16個のLNPクライアントプログラムが動作できる。デフォルトでは、ホストPCのLNPアドレスは8、RCXは0となっている。複数台のRCXを用いる場合は、適宜config.hに定義されたHOST（RCX）アドレスを変更してカーネルを作り直さなくてはならない。つまりLNPアドレスは動的に設定できない。

各RCX内部の論理通信ポートへは、

```
lnp_addressing_set_handler(1,
&myProg);
```

のようにlnp_addressing_set_handler() APIを用いてパケット到着後の処理を登録することによって、クライアントプログラムを実装できる。この例では、1番ポートに到着したパケットを処理するためにmyProg()を登録している。このように、LNPを用いれば特定のRCXの特定のプログラムと通信することができるし、全部のRCXに同報通信することもできる。また、myProg()の内容によってはより高レベルなプロトコルも簡単に実装できる。これらのしくみは、Ethernet－IP－TCPのような階層的なネットワーク参照モデルとよく似ていて面白い。

なお、ユーザープログラムのダウンロード機能は、0番ポートを使用する上位プロトコルとして実装されている。LNPの上位プロトコルの作成はこれを参考にするとよいだろう。詳しくはソースコードを参照してほしい。

legOSのプログラム例

NQCの例と同様のエッジ探索プログラムが**リスト4**だ。このプログラムを動作させてすぐにわかるのは、モーターがものすごい速さで反応することである。legOSの速さが体感できたのではないだろうか？　コード中のmotor_b_dir関数やLIGHTマクロは、インクルードファイルdsensor.h、dmotor.hに定義されているAPIで、ユーザーはこれらをインクルードすればいつでも使用できる。

また、ユーザープログラムは必ずmain()という名前の関数を1つ含んでいなければならない。legOSカーネルのプログラムローダは、プログラム番号が選択された状態でRunボタンが押されたとき、新しいタスクを生成してmain()をエントリーポイントとして処理を開始するからである。再びRunボタンを押すことで、カーネルはこのタスクを消去してプログラムが停止する（プログラムイメージは消えないのでRunボタンで再スタートできる）。

この例ではごく単純なコードを示したが、C言語で記述するので、当然多数の関数を持っていてもいいし、コンパイルスイッチも利用できる。また、ソースコードは複数のファイルに分けることも可能で、よく使う有用なコードをライブラリにして再利用することもできる。また、ユーザープログラムからさらに新しいタスクを生成して、多重処理やLNPを用いたネットワーク通信によるインテリジェントな処理を行うこともできる。

▼リスト4　エッジ探索プログラム（legOS用）

```
#include <dsensor.h>
#include <dmotor.h>

#define TH    60

int
main()
{
  ds_active((unsigned *const)&SENSOR_2); /* ライトセンサ初期化 */

  motor_b_speed(255);    /* モーター速度設定 */
  while (1) {            /* 無限ループ */
    if (LIGHT_2 < TH) {  /* 暗いとき */
      motor_b_dir(fwd);  /* モーター正転 */
    } else {             /* 明るいとき */
      motor_b_dir(rev);  /* モーター逆転 */
    }
  }
  return 0;
}
```

OSの世界を覗く楽しさ

legOSを使用するのは、RCXコードやNQCに比べてかなり敷居が高い。C言語の知識だけでなくマルチタスキングやハードウェアの知識が要求される。また、legOSを構成する機能にどのようなものがあり、ほかの機能とどのようにかかわっているかを理解しなければ、最適な機能の選択などできない。しかし、C言語が少しでも読めたら試

してみる価値がある。legOSのソースコードを追いかけているうちに「なーんだ、コンピュータってこんなふうに動いてるんだ」と発見できてウキウキすること請け合いである。

legOSはOSの勉強の教材としても好適である。カーネルコードが7000行程度とコンパクトだし、ロボットを動かしながらプログラミングの勉強ができれば楽しいではないか。

■LETroプロジェクト

NQCとlegOSはいずれも海外で開発されたものであり、ドキュメントもほとんどが英文である。一方、日本ではいしかわきょーすけ氏のLETroプロジェクト（http://www.asahi-net.or.jp/~qx5k-iskw/lego/letro/index.html）が登場した。このプロジェクトの目的は、NQCの手軽さでlegOSのようにRCXの能力を限界まで引き出すことにある。C言語とμITRONライクなOS、さまざまなツールを組み合わせて、プログラムを開発できるようにするという。大いに期待したい。

最後に

筆者は長い間、PC上でソフトウェアを作っても決してモニタ画面から出て行くことができない（触ることのできない）もどかしさを感じて満足しきれなかった。自分でデザインして作り上げた作品がソフトウェアで思いどおりに動作することから得られる感覚は、シミュレーションだけでは決して得られない貴重な体験であると思う。この記事を通して1人でも多くの方にMindStormsの愉しさを知ってもらうことが筆者の願いである。

最後にMindStormsの名付け親、Seymour Papert博士の言葉でお別れしよう。

“Knowledge is only part of understanding. Genuine understanding comes from hands-on experience.” ―知識は理解の一部にすぎない。本当の理解は自分の手による経験なくしては得られないものである。

Column

MindStorms関連情報

■「ロボティクスセット」と「ドロイドキット」

MindStormsシリーズの新しい製品、「LEGO MindStorms Robotics Discovery Set（RDS）」と「LEGO MindStorms Droid Developer Kit（DDK）」の日本語版「ロボティクス セット」「ドロイド キット」の販売が開始された。

ロボティクスセットにはRCXの代わりにスカウトと呼ばれる青いユニットが付属し、本体のボタンを押してLED上でアイコンを組み合わせてプログラムを作成する（写真左）。ドロイドキットはStarWarsに登場するドロイド（ロボット）を作成できるセットで、マイクロスカウトと呼ばれる白いユニットが付属し、用意されている7つのプログラムをボタンで選択する（写真右）。

いずれもPCなしで使用できる反面、RISと比べて機能が限定されるため、オリジナルロボットを作りたければPC上で自由にプログラミングできるRISがお勧めである。

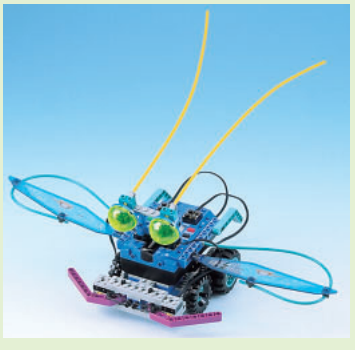
■RISの入手方法

ロボティクスセット、ドロイドキットと違って、RISは残念なことに日本国内では正式に販売されていない。しかし、並行輸入した製品が一部の販売店や以下を始めとする国内のオンラインショップから比較的容易に入手可能だ。現在の価格はおおよそ3万5000円程度である。OSAMU氏のWebサイト、MINDSTORMS Japan（http://www.osamu.nop.or.jp/mindstorms/）などを参考にするとよいだろう。

e-sekai　http://www.e-sekai.com/

■コンテストに参加する

MindStormsでオリジナルな作品を作ったら、LEGO社のMindStormsオフィシャルサイト（http://www.legomindstorms.com/）に登録しよう。ユーザー登録をすると、Webスペースが与えられ、作ったロボットの情報をアップロードして世界中に公開できる。このサイトでは毎月「HALL OF FAME」というコンテストが開催されている。LEGO社が登録されたロボットのなかから優れたものを選んでノービス部門、エキスパート部門それぞれに4台ずつノミネートし、登録ユーザーの投票にかけるものだ。優勝すれば殿堂入りする。



ロボティクスセット（左）とドロイドキット（右）

隠喩としてのコンピュータ

GUIをめぐる反時代的考察

文：豊福 剛
Text : Tsuyoshi Toyofuku

コンピュータユーザーにとって、テキストエディタほど好みが分かれるものはない。昔からUNIXユーザーの間では、viとEmacsの優劣をめぐって聖戦（*holy wars*）が繰り返されてきた。

viは*visual interface*の略なのだが、これが意味するインターフェイスとは、ラインエディタexに対するインターフェイスである。viで実行可能な編集コマンドは、exのそれと等価であり、viとexの違いは、ビジュアルであること、すなわちスクリーン表示にするかどうかにある。vi/exが提供する編集コマンドは必要十分なものとはいえ、複数のファイルを扱うこともできるし、UNIXのシェルコマンドを実行して、その結果を挿入することもできる。編集コマンドの置き換えレベルとはいえ、簡単なマクロ機能も提供されているなど、その機能には奥深いところがあるのは、案外知られていないようだ。ただし、viの操作性そのものは直感的ではなく、インサートモードとコマンドモードの区別に慣れない間は、ビープ音でうるさく警告されることになる。

viがミニマリズム的なのに対して、Emacsはバロック的であり、たくさんの樹木が生い茂る森のようでもある。viと同様にUNIXのシェルコマンドを実行できるのは当然だが、画面を分割して複数のウィンドウを表示できる利点は大きい*。さらに、各種プログラミング言語に対応したインデント処理やカーソル移動などのプログラミング支援機能、アウトライン処理などの文書作成支援機能もあるし、電子メールやネットニュースの読み書き、Diredコマンドでのディレクトリ／ファイルの編集など、およそテキストとして扱うことができる処理であれば、すべてEmacsの環境下で実行可能なのである。

しかし、テキストエディタとしてviとEmacsのどちらが優れているかという問いは不毛であり、用途に応じて使い分けるというのが現実的だろう。ただし、Emacsを狭い意味でのテキストエディタの枠に閉じ込めることはできない。ユーザーにとって、EmacsはUNIXに対する統合された操作環境でもある。EmacsはUNIXのCUIを拡張して、よりインタラクティブなユーザーインターフェイスを提供する。

GUIはコンピュータの進化を妨げた

一般にGUI（*Graphical User Interface*）とCUI（*Character User Interface*）は、互いに対立するものとし

て考えられている。歴史的にはCUIのあとにGUIが発明されたわけで、それはユーザーインターフェイスの進歩として評価されている。そして、GUIはCUIよりも直観的で、初心者でも操作できるという主張が繰り返されてきた。

プログラムを起動するには、CUIではコマンドをタイプしなければならないが、GUIではアイコンをクリックするだけでよく、コマンドを覚える必要がない。アイコンというスクリーン上の図像（イメージ）は、ある隠喩（メタファー）によって成立する象徴（シンボル）である。スクリーン全体が机の隠喩として構成される（デスクトップメタファー）。この隠喩においては、ディレクトリは「フォルダ」として、ファイルは「書類」として、ファイルの削除機能は「ごみ箱」として、それぞれ象徴的な図像として提示される。これらの象徴体系を支える隠喩は、日常的な道具を模倣しているので、コンピュータの初心者にとって直感的に機能を理解することができる、と主張されている。

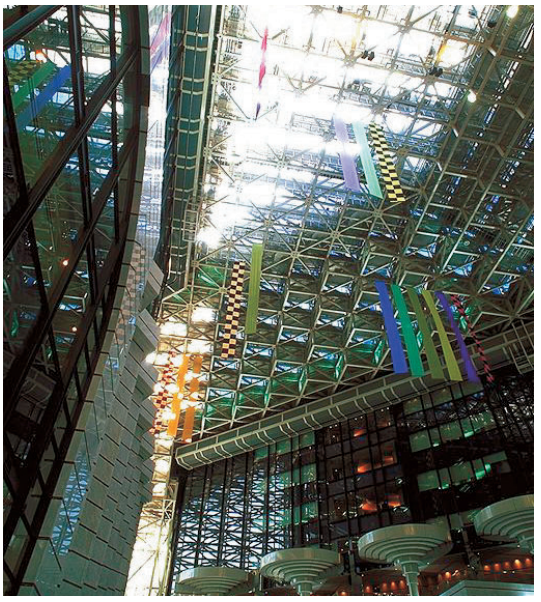
しかし、GUIとCUIを二項対立としてとらえる図式は、少し疑ってかかったほうがいい。マウスの発明者として有名なダグラス・エンゲルバートは、'68年にSRI（Stanford Research Institute）でNLS（oNLine System）を開発した。GUIの起源をたどっていくと、このNLSに行き着く。エンゲルバートは'98年のインタビューで次のように発言している（http://www.ascii.co.jp/ascii24/call.cgi?file=issue/1998/1208/keyp01.html）。

――'68年のプレゼンテーションから30年が経ちましたが、この間のコンピューターの歴史において、予想していなかったことはありましたか？

Engelbart：「進化のスピードが思ったより遅かったのは予想外でした。その妨げとなったのは、GUIとWYSIWYGにあると思います。GUIは初心者には向いていますが、あくまで簡単で自然なインターフェイスが優れているのならば、人は大人になっても三輪車に乗っているはずでしょう。WYSIWYGは、コンピューター革命以前の書類体裁を模倣したものです。ですが、これからの時代において、印刷するということがどれだけ重要な意味を持つというのでしょうか。」

人間がコンピュータに歩み寄るのではなく、コンピュータが人間に歩み寄らなければならない、という信念がGUIを発明した。しかし、CUIで可能な操作のすべてがGUIにおいても実行できるわけではない以上、GUIはCUIを乗り

＊Linuxで一般的に利用されているviである、vimはウィンドウ分割が可能である。



越えたものではない。GUIとCUIは相補的に捉えられるべきだろう。問題は、人間がコンピュータに歩み寄るための回路を遮断すべきかどうかにある。遮断すべきだ、と主張しているコンピュータの典型がMacintoshである。

Macintoshは光の牢獄

Macintoshには、UNIXのようなCUIは存在しない（WindowsにはMS-DOSプロンプトというCUIが存在するとはいえ、それはUNIXと比較すればあまりに貧困なCUIである）。MacintoshのGUI思想は、CUIとGUIを対立的にとらえたうえで、CUIを否定している。Macintoshユーザーにとって、コンピュータとはスクリーンに提示されている「目に見えるもの」こそがすべてであり、それは永遠に追放されることのないエデンの園なのかもしれない。皮肉なことに、そこにはイブをそそのかす蛇はいないし、イブもアダムも知恵の樹の実を食べることはないのだ。

これに対して、人間がコンピュータに歩み寄るための回路は開かれているべきだと考えるのが「啓蒙」である。啓蒙は蛇であらねばならない。蛇は「目に見えるもの」が「目に見えないもの」によって作り出された表層に過ぎないことを知っている。蛇は「目に見えないもの」が言語によって記述されていることを知っている。蛇は「目に見えないもの」を見るために、ソースコードを開示する。

啓蒙は*enlightenment*の訳語である。これと同じ名前のウィンドウマネージャが存在するが、LinuxにおけるGUIは啓蒙と対立するものではないので、このネーミングは必ずしも皮肉ではないのだ。

Macintosh崇拝者には、シェルプロンプトが表示されているだけのLinuxの画面は、あたかも闇のように感じられるだろう。プロンプトに向かってタイプされるコマンドは、闇に向かって異教徒が発する呪文のように感じられるだろう。しかし、ここでEnlightenmentを起動すれば、闇はGUIの光で満たされる。しかも、その光は一切の闇を消し去ったわけではなく、闇を伺うための窓、xterm／ktermを光の中に現出させることもできる。この闇を支配するのは言語である。「はじめにことばがあった」のであれば、言語を遮断してしまったMacintoshは、光の牢獄なのだ。

「ユーザー」の誕生

スクリーンには「遮断する」という意味がある。

MacintoshのGUI思想は、光から闇を遮断する。それはCUIを排除するだけでなく、ユーザープログラミングも排除する。スクリーンの向こう側にアプリケーション開発者がいて、スクリーンのこちら側にユーザーがいる。プログラミングは闇に属する行為であり、それは専門家という司祭に委ねられるべき行為なのだ。司祭は、ユーザーにアプリケーションを提供する。ユーザーにとって、Macintoshを使うことは、すなわちアプリケーションを使うことと同義である。

ユーザーはGUIに従属し、アプリケーションに従属する。もちろん、これはMacintoshに限ったことではなく、Windowsにおいても同様である。パーソナルコンピュータがこれだけ普及したのは、GUIがあり、アプリケーションがあったからだ。Macintosh、そしてWindowsのGUIは、もともとアプリケーションのユーザーインターフェイスを統一するために導入された。GUI以前のアプリケーションでは、そのユーザーインターフェイスはアプリケーションごとに独自なスタイルで提供されていた。GUIをOSが提供することで、アプリケーションの開発者は独自にGUIを開発する必要がなくなり、開発効率が上がる。一方、ユーザーにはアプリケーションの操作法に統一的なスタイルがもたらされる。GUIはアプリケーションの進化によって必然的に要請されたのだ。

現在、LinuxにおけるGUI環境はGNOMEやKDEによって整備され、ユーザーのためのアプリケーションが開発されつつある。ただし、GUIが導入され、アプリケーションが提供されたとしても、そのことによってLinuxがMacintoshやWindowsのような牢獄的OSに変貌することはないはずだ。LinuxでGUIベースのアプリケーションが使えるとしても、GUIだけがユーザーに強要されるわけではなく、Emacsのような拡張されたCUIを使うこともできる。Linuxは、CUIとGUIを二項対立としてはとらえない。用途に応じてCUIとGUIを使い分ければよいのだ。

LinuxやEmacsにおいては、人間がコンピュータに対して主体的に関与するための回路が開かれている。とりわけEmacsにおけるユーザーインターフェイスは、ユーザーによるプログラミングを支援するものとして構成されている。このようなユーザーインターフェイス思想は、Emacsに特有のものではなく、実はMacintoshの誕生に大きな影響を与えたアラン・ケイにも共有されていた。次回、アラン・ケイとMacintoshの間の「連続性」ではなく「断絶性」について考えてみたい。

Profile

とよふく つよし

1962年生まれ。メディアデザイン研究所技術顧問。訳書に『Javaプログラムクイックリファレンス』『Java分散コンピューティング』（オライリージャパン）『GIMPパーフェクトガイド』（エムディーエヌコーポレーション）などがある。

オープンソース
・
アクティビズム

韓国のコピーレフトな人たちの話

文：安田幸弘
Text : Yukihiro Yasuda

先日、「国際労働メディア」という集まりに参加するために、ソウルに行って来た。名前からもわかるように、韓国の労働運動グループが企画した集まりで、海を隔てた労働者たちが暴虐の鎖を断ち、圧政の壁を打ち破るために、コンピュータやインターネット、ビデオなどのメディアを使いこなしていこうという試みである。

フリーランサーのぼくがなんでこんな会議に出かけたのかというと、ひとつは「情報化時代のプライバシー」と題して、最近になって世界的な規模で問題になっているインターネットの規制や盗聴・検閲がテーマになっていたからだ。ぼくはそれに関するセッションで、最近、日本でもじわじわと強まってきたインターネットの規制や盗聴法の話と、セキュリティシステムなどを紹介したのだが、同じセッションでは韓国の住民データベースや当局の検閲、そしてイギリスからは最近話題の地球規模の盗聴システムであるエシュロンや、イギリスで計画されているインターネットの盗聴制度などが紹介されていた。インターネットの世界も、だんだん住み難くなってきたものである。

さて、そしてもうひとつが、Linuxを始めとするオープンソースソフトウェアの利用と意義に関するセッションだった。このセッションは、オープンソースソフトウェアの利用可能性と普及がテーマで、韓国でLinuxがどのように利用されているのか知りたかったというのも理由のひとつだった。

Wordもかなわない国民的ワープロ

そもそもアジアの国々の間では、ソフトの違法なコピーは日常茶飯事だ。この集まりにトルコから参加したある参加者は「トルコでは道端で野菜や果物を売ってる隣でパソコンのソフトを売ってるよ。え？ もちろん海賊版さ。だいたいどれでも1ドルから2ドル」なんて言ってた。ビル・ゲイツがイライラするのも無理はない。

しばらく前まで、韓国でもこの種の違法コピーは堂々と売られていたそうだが、今回見てきた「韓国の秋葉原」と呼ばれる竜山電子商店街では見かけなかった。1年ほど前までは、竜山でパソコンを買うとWindowsは無料でコピー品をインストールするのがあたりまえだったそうだが、マイクロソフトから強力なクレームがついて、次々と販売店が訴えられたという。

そこで浮上したのが無料のLinuxだ。特に韓国では、最近、日本の通産省にあたる産業資源部という役所が中心となって、21世紀の情報化に対応するためと称して低価格な「国民PC」の普及を進めている。竜山では日本円で5万円程度で売られている「国民PC」にもLinux版があり、マスコミでも頻繁に話題になっている。

ちなみに、韓国でも、やはりいくつかの韓国語バージョンのLinuxパッケージが売られている。その中で、メジャーなパッケージはMIZI LINUXというRed Hadをベースにして韓国語のKDE、GNOMEを乗せたパッケージだ。ぼくも韓国みやげにこのパッケージを買ってきたのだが、実によくできている。パッケージとしての完成度の高さもさることながら、Linux版の「アレアハングル」というワープロソフトが付属しているのも大きな魅力だ。

「アレアハングル」は、日本でいえば一太郎に相当する韓国語のワープロソフトだが、約70%以上という一太郎とは比べものにならないほど高い市場占有率を誇る国民的ワープロだ。事実上、このソフトのファイル形式は韓国ワープロのデファクトスタンダードとなっていて、アレアハングルを持っていないと韓国人との文書の交換ができない。ぼくも、韓国語を書くわ

けではないのだが、アレアハングルフォーマットで送られたファイルを開くのにいつも苦労していたので、MIZI LINUXのX版「アレアハングル」は嬉しかった。

面白いのは、この「アレアハングル」がマイクロソフトを悩ませているという話。MS Wordがトップシェアを取れないのは、世界で唯一、韓国だけだそうで、MSは「アレアハングル」の買収工作やらOfficeの無料配布やら値下げやら、ありとあらゆる手段で食い込みをはかっているそうだが、いまのところ「アレアハングル」の優位は揺らぎそうもない。

ただし、「アレアハングル」の開発元は一時、倒産の危機に瀕した。いくらシェアが高くても、ほとんどが違法コピーだからだ。政府や学校でもコピー品が使われていたそうで、これでは儲からないのはあたりまえだ。「アレアハングル事件」は、それだけで1冊の本になりそうな面白い（？）事件だったのだが、この過程でオープンソース化も議論された。オープンソース化は実現しなかったものの、アレアハングル互換のオープンソースソフトの開発は、現在も続いている。

オープンソースはパクリの文化

ところで、ソフトの違法なコピーや偽ブランドなど、アジアでの知的所有権侵害の問題は、国際的な通商問題でもある。日本もこの問題については決して胸を張れる状態ではないのだが、韓国の場合、コピーやパクリはひとつの文化とも言える状態だ。日本のテレビ番組や歌、マンガのパクリは韓国に蔓延していて問題になっているし、インターネットのコンテンツの転載も、最近は規制ができたそうだが、事実上、ほとんど野放しという、なかなかすごい状態である。

労働メディアの集まりでは、「マイクロソフト対オープンソース」的な二項対立の図式を前提として、Linuxを始めとするオープンソースの利用拡大が、情報の独占に対する代案となり得る、というような形で進められたのだが、こうした二項対立的な議論を聞きながら、韓国の—あるいはアジア的な—「パクリの文化」を生かす道もあるのではないかと考えた。

現在のアジア的パクリが問題になるのは、それがルール違反だからなのだが、「いくらパクってもかまわない」というルールだったらどうだろう？ GNUのコピーレフトやオープンソースのさまざまなライセンスは、いわば「パクリのルール」だが、NCSA httpdをパクったApacheを筆頭に、オープンソースソフトの世界はほとんどがそんなパクリのルールの下で成長してきたソフトである。

また、香港映画はそれこそ仁義なきパクリのルールの下で、お互いにパクりパクられながら、パクりに負けない斬新なアイデアを育ててきたことを思い起こすと、知的所有権なんて、パクられたら価値を失う程度のつまらない製品を保護するためのどうでもいいような仕組みなんじゃないかとさえ思えてくるほどだ。

知的所有権をなくしてしまえというのは極論かもしれない。だけど、既存のソフトウェアメーカーの主張だけがすべてではないということを、もっと広く世間に知らせる努力が求められているのではないだろうか。

Profile

やすだ ゆきひろ

生業はテクニカルライター。原稿書きのかたわら、(株)市民電子情報網のボランティア社長兼技術スタッフとして、NGO向けプロバイダのネットワーク運営に携わる。

ドクターShiodaの

ギョーカイSnapShot

PC業界今月の話題

文：塩田紳二
Text : Shinji Shioda

11・29 Microsoft Millenniumのβ2
11・25 Windows2000 日本語版は、来年2月18日
11・25 Microsoft 国内CATV大手タイタス買収か
11・24 AMD 750MHz Athlon発表
11・17 VIA Celeronクローン発表
11・15 Red Hat Cygnusを買収
11・15 Sony、Palm Comuptingと提携。来年PDAを出荷

11月はComdexがあったので、その関連で意外にニュースが多かった。また、Microsoftに対する事実認定が出たあとも、いろいろとニュースが出た。Linux関係では、Red Hatがいろいろと話題になったようだが……。

パソコンの値段は下がるが……

米国の無料パソコンブームの火付け役Free-PC社と低価格パソコンの火付け役、eMachines社（ソーテックのe-Oneを米国で扱って、Appleともめたので有名）が合併するのだとか。Free-PC社は、コンパックのパソコンを無料提供する代わりに、広告や通販からのリベートで、ハードウェア代をまかなう予定だったようだが、どうもうまくいかなかったようである。もっとも、ずさんな管理などというウワサが出ていたので、かならずしも、ビジネスモデルとして失敗したかどうかは不明である。

eMachinesは、プロバイダ契約などを組みにして低価格パソコンを売り出し、いまやPCメーカーとしてはベスト5に入るまでになったが、こっちのほうは、そんなにも調子が悪いわけではなさそうだ。それに、いまでも低価格パソコンは人気である。

感じとしては、Free-PCは、値段を下げすぎたという感じだろうか。ちゃんとした定価を持つ低価格PCに、プロバイダ契約などによる、リベートや各種払い戻しを組み合わせて、値段を除々に下げていくやり方のほうが確実だったということである。もっとも、eMachinesのマシンも、場合によっては、タダ同然に売られているところもあるようなので、限りなく無料に近いとはいえる。結局、金の管理をちゃんとしたほうが勝ちということか。

日本も2月出荷のWindows 2000

いろいろともめていたようだが、日本でも、Windows 2000は、2月に出荷されることに決まったようである。日付は、2月18日。つまり、米国での出荷の翌日ということになる。もっとも、米国、ヨーロッパ、日本で2月17日とすると、時差の関係から、日本での発売が最も早くなるのだとか。この18日は、米国での発売が終わった次という意味。

今回の発表に合わせて、いろいろと「ハデ」なイベントが企画されているということなので、ニュースバリューを米国内で高めるには、先に日本で発売しては「まずい」ということなのであろう。

とりあえず、「年度」内の出荷が確定し、ちょっと安心した人もいるかもしれない。年度またいじゃうと、予算とか大変だものねぇ。でも、Y2Kで大変な次期にWindows 2000（W2K）を入れるなんて、ちょっと無謀ではありませんか？

日本のマイクロソフトにしても、日本では、4月末からはゴールデンウィークだし、たしかマイクロソフトの期末は6月だったので、あまり遅くなると、発表イベントもやりにくいだろうし（春は新製品の時期でもある）、今期の売上げにも響くというところでしょうか。

Red Hatも買収好き？

アンチMicrosoftの波にいま一番乗っているのは、Red Hatだと言われているが、株式公開してから、いろいろと調子がいいようである。つい最近も、GNU開発ツールメーカーのCygnus

Solutionsを買収した。このほかにLinuxCareの買収失敗の話とか、Corel買収のウワサもあるようだ。

国内では、話題になった、LASER5との契約打ち切りのあと、日本法人を作り、ハードメーカーとの提携を進めているようだ。

Red Hatもそういう意味では、いまや、MicrosoftやSUNなどと並んで、競争を繰り広げる立場に立った（と思っている）のだろうが、SUNがそうだったように、競争に一生懸命になるあまりに、“Microsoft化”しないことを祈るばかりである。

ソニーも話題多し

ノートが好調なソニーだが、今回は、米国で人気ナンバーワンのPDA（Hand heldデバイス）であるPalm Computingと提携した。なんでも、Palmシリーズでメモリスティックをサポートし、Palm OSにAV機能を入れるのだとか。ソニーは、来年、Palm-OS搭載のHand Heldデバイス（簡単にいえばPalmシリーズ互換機）を出荷する予定。

いままで、市場での成功は別として、国内でソニーが先鞭を付けた分野に、松下など国内家電メーカーが追従するのが「ならわし」だったが、これで、家電メーカーもまたこぞって、PDAに注目することになるのかも（かつて、電子手帳ブームのときに、松下とか日立は、シャープの電子手帳をOEMしたことがある）。

また、Microsoftは、WindowsCEにテコ入れを考えていて、Palm-Size PC版WindowsCEのバージョンアップなどが計画されている（名前も変えるのだとか）。だから、同じComdexで行われた発表にビルゲイツは激怒していたとか。でも、国内メーカーがソニー追従に入ると、Microsoftが絡む可能性（特に松下とかWindowsCEをソニーへの対抗上担ぐ可能性もあり）があり、そんなに悪いことでもないと思うけどねぇ（それに、ソニーだって、WIndowsCE搭載のプロジェクタなんかも作っているようだし）。

Microsoftは、PlayStation2に刺激されて、PC技術をベースにしたゲーム専用機（X-BOX）を計画中という話だが、今回のSONY-Palm Computingとの提携で、ソニー vs. Microsoftという構図も今後の展開によってはありえるだろう。最も、米国流ビジネスだと、右手で握手しながら、左手で殴り合うなんてことは少なくないので、米国的対決ってやつでしょうか。

さて、Microsoftだが

裁判関係でいろいろとニュースネタになっているMicrosoftだが、製品面では、来年発表のWindows 2000のほかには、Windows 98の後継であるMillennium（コードネーム）のβ2を出したようだ。このMillennium、レガシーデバイスやDOSプログラムのサポートをやめるという話だが、Microsoftの製品って、バージョン3からと言われており、これがWindows 95から数えて3つ目なので、意外に良くなっているのかも。だが、筆者は、すでにWindows 95で、仕事用のデスクトップOSとしては、見切りをつけてしまったので、どうなろうが、個人的には関心のないところ。Windows 98が静かなので、今度は、Windows 95のような爆発的人気を期待したいところだが、裁判の影響なんかもあって、どうなることやら。

では、今月はこのへんで。

無料パソコンの火付け役、Free-PC社。合併後は無料PCの提供をやめることになった。これまでに提供したPCは3万台と言われる。
http://www.free-pc.com/

Linux界の出世頭、Red Hat社。株式公開で得た豊富な資金をもとに、Cygnus Solutions社の買収や800万ドルの基金設立など話題は多い。
http://www.redhat.com/

正しきタコへの道

初級Linuxer養成講座

第5回　LANのないLinuxなんて……(その2)

前回はLinuxマシンをLANに接続する方法を紹介した。家庭内でも小規模のLANを引けば、「これぞLinuxの醍醐味」といった使い方がいろいろできるのだが、面倒な設定が必要になる場合も多い。今回は難しい設定をしなくても利用できるネットワークアプリケーションの世界を見てみることにしよう。

文：竹田善太郎
Text : Zentaro Takeda

正直に白状するが、ここ2、3年は、Linuxを自分の仕事に使うということをあまりしていなかった。記事のネタとして実験をしたり、他人のためにインストールや各種の設定の手伝いをしたり、といったことばかりやっていて、肝心の原稿仕事ができる環境を整備する暇がなかったのだ。

雑誌や書籍の原稿を書くという仕事では、WYSIWYG機能を完備したワープロソフトは**無用の長物**であって、使いやすいテキストエディタと賢い日本語入力システムだけあればいいのだが、実はLinuxでは日本語の商用原稿を書くのに適したエディタが見当たらないのだ。**Muleやviがあるじゃないか**という声が聞こえてくるようだが、実はこれらのエディタは、デフォルトの状態では日本語の原稿を作成するうえでの重大な欠点があるのだ。

このように書くと、不安な気持ちに襲われる読者もいるかもしれない。**Linuxは使い物にならないと言いたいのか？**と詰問したくもなるだろう。実際には、メールを書いたり短い私的な文章を書いたり、あるいはTeXで論文を書くなどの**アカデミックな**使い方をしたりする分にはまったく問題はない。また、MuleやEmacsならば、きちんと設定さえすれば商用の原稿を書くのにも問題はないし、へたなワープロソフトよりも快適に作業できるようになる。

しかし、現在流通しているLinuxのディストリビューションでは、インストール直後のデフォルトの状態で、たとえばMuleを使おうとすると、**3分でストレスが極大**に高まって投げ出したくなる。たとえば、Backspaceキーの扱いが全く不統一な状態になっていて、通常のテキスト入力時にはBackspaceでなくDeleteキーを使わないと1文字消去ができないと思ったら、日本語入力モードではそれが逆になっている、といった具合である。

このへんは、Linuxお得意の設定ファイルを書いてやればきれいに解決できるのだが、その書き方はかなり面倒で、おそらく昨日今日Linuxを使い始めたようなユーザーには、手も足も出ないだろう。こんなときには、**設定ファイルを他人からいただく**という方法もあるのだが、周りにLinuxのパワーユーザーがいないことも多いし、たとえいたとしても、**変態的**な使い方をしているようなユーザーだったりして、何の参考にもならない場合も結構多い。少なくとも、ユーザーからお金をとるようなディストリビューションなら、参考用というスタンスでかまわないから、なんらかの常識的な設定ファイルをつけておいてくれてもよいのではないだろうか。

話を元に戻すと、「Linuxをどんどん使おう」というような原稿を書いている以上、少しは自分の仕事でもLinuxを使わないと、なんだか読者を**だましている**ようで申し訳ないかなと思い、昔、UNIX上で原稿仕事をしていたころに使っていたMule（Emacs）の設定ファイルが残っていないかどうか探してみたのだ。ところが、自宅のマシンのディスクの中をいくら探し回っても、見つからなかった。どうやら、ある時期にOSをインストールした際に、上書きして消してしまったようなのだ。もう見つからないかとあきらめかけていたとき、ひょんな場所からファイルが出てきた。7年以上前からずっと契約してきたインターネットプロバイダのホストの、自分のホームディレクトリに置いてあったのだ。

このISPを選んだ際の条件のひとつが、「telnetとFTPでログインでき

るホストのあるところ」だったのだが、契約して使い始めたときに、少しでも設定を楽にしたかったので、会社で使っていた各種の設定ファイルをまとめてコピーしてあったのだ。実際にはtelnetなどでログインすることはそれほど多くなく、ましてや遅いくせに料金が高い電話回線経由では、Emacsなどのフルスクリーンエディタを使うことは少なかったので忘れていたのだが、おかげで**FTP**を使ってローカルのマシンにコピーするだけで、昔の実用的なエディタ設定を取り戻すことができたのだ。

そんなわけで、この原稿は久しぶりにLinux上のMuleを使って書いている。なかなか快適である。ネットワークのアプリケーションの中ではかなり歴史が古いのに、現在ではその存在さえ知らないユーザーがいるかもしれない**telnet**と**FTP**に救われた、という形に（強引に）落したところで、前回の続きに移ろう。Muleやviの問題点と実用的に使うためのコツについては後日あらためて触れることにしたい。

いざというときの「telnet」

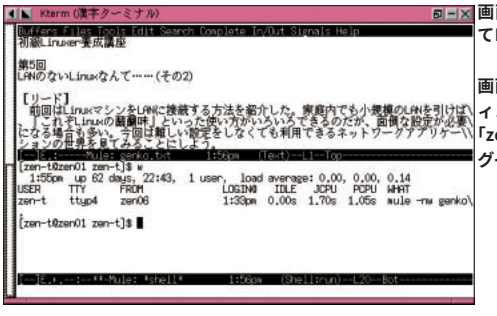
LinuxをLANにつなぐと、すぐに**ApacheでWebサーバだ！**とか**Sambaでファイル共有だ！**ということになるのだが、これらのネットワークアプリケーションはLinuxをインストールしても、そのままではまともに使える状態にはなっていない。Sambaなどは、以前はすぐにでも使えたのだが、Windows 95がWindows 98になった時点で、設定ファイルを変更したり、Samba専用のユーザー管理作業をしないと使えないようになってしまったのだ。自分たちの都合で勝手に仕様を変えるMicrosoftも悪いのだが、ともかくLinuxの敷居が少し高くなってしまったことに違いはない。

ともあれ、Sambaを使えるようにする設定は、実際にはそれほど面倒ではないし、雑誌記事や書籍などで**いや**というほど解説されているので、ちょっと暇があればだれにでも使えるようになるだろう。しかし、「その時間すら惜しいけど、LANがちゃんと機能しているかどうか今すぐに確かめたい」という**せっかち**なユーザーも少なくあるまい。そんなときに、真っ先に試すのは前回触れたpingコマンドなのだが、このコマンドは実用性の面では**まったくの無用**である。そこで、次に試してみるべきなのが**telnet**である。

telnetというのは、LinuxやUNIX系OSに限らず、TCP/IPプロトコルによる接続が可能なOSなら、ほとんどの製品が備えている機能の一種で、ローカルマシンで動かす「telnetクライアント」プログラムを使って、ネットワーク（LANでもダイヤルアップ接続でも、TCP/IPプロトコルが使われていればなんでもよい）で接続された別のマシン（リモートマシン）にログインして、あたかもそのマシンのコンソールを使っているかのように操作できる仕組みである。telnetクライアントはWindows 95／98などの一般ユーザー向けOSに附属しているほか、フリーソフトウェアやシェアウェアとして流通している「ターミナルソフト」にもtelnetクライアントの機能を備えているものがある。

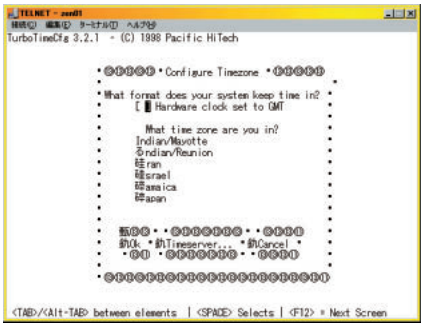
一方で、telnetクライアントの接続先には「telnetサーバ」ともいえるプログラムが必要になる。Windows 95／98にはないが、Windows NTにはtelnetサーバ機能がある。ただし、Windows NTにtelnetで接続しても、使えるのはごく一部の管理者用コマンドだけで、**実用的な作業はできない**と思ったほうがよい。

Linuxでは、ほとんどすべてのディストリビューションで、telnetサーバ機能がすぐに使えるようになっている。Linuxの場合は、Windows NTとは異なり、およそ「コマンドライン」でできることは、どんなことでもtelnetでログインして実行できる。たとえば、sedやawkなどのフィルタ型のコマンドだけでなく、Muleやviなどのスクリーンエディタを起動して作業することも可能だ（**画面1**）。ただし、**Xのアプリ**やコンソールで実行するアプリケーションの中でも、グラフィカルな画面を駆使したようなものは、telnet経由



画面1 telnetを使って他のLinuxマシンにログインしてMuleを使っている例

といっても、ローカルのマシンでMuleを使っている画面と全く同じで区別できないので、Muleのシェルウィンドウで「w」コマンドを実行してみた。ここでは、「zen06」というマシンから「zen01」というマシンにログインしているのだが、おわかりいただけるだろうか？



画面2 TurboLinuxの「turbotimecfg」をtelnet経由で使ってみる

Windowsのtelnet.exeで行った例。Linuxマシンのtelnetからは問題なく使えるが、Windowsからだと画面が正しく表示できない（使えないことはないが……）。

では使用できない場合がある（**画面2**）。

■**万能ターミナル**

telnetが有用なのは、Linuxやその他のUNIX系OSにとどまらず、Windows、Macintosh、DOSから、一部の携帯端末（PDA）まで、TCP/IPのネットワーク機能を持ったコンピュータなら、ほとんどの製品でtelnetを利用できる点にある。たとえば、Windows 98ではc:\windows\telnet.exeというコマンドを使えば、LANやダイヤルアップ接続したネットワーク上のLinux（UNIX）マシンにログインすることができる（**画面3**）。もちろんLinuxにもtelnetクライアントは存在して、そのものずばり**telnet**というコマンド名になっている。たとえば、「orange」という名前のマシンにtelnetを使ってログインしたければ、コマンドラインで、

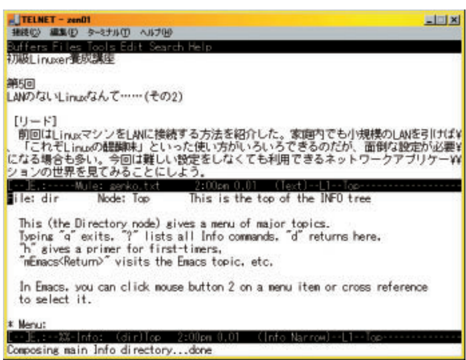
```
$ telnet orange
```

と入力してやればよい。

「個人が1人だけで使っているマシンに、別のマシンからログインできて**何がうれしいの？**」と疑問に思うかもしれないが、たとえば書斎に置いているLinuxマシンへ、リビングに置いたWindowsマシンからログインして作業するなどの使い方ができる。なにより便利なのが、Linuxマシンになんらかのトラブル、たとえばXの設定を間違えて、コンソールに何も表示されなくなったり、Linuxマシンに接続したキーボードがハングアップしたなどの理由で、コンソールからの操作が一切できなくなってしまったような場合、別のマシンからログインできれば、シャットダウンやリブートなどの処理をすることができるのだ。実際のところ、telnetを使用する場面としてもっとも多いのが、このような**トラブル解決の手段**としての使い方でなないかと思う。

余談だが、以前、仕事でよく使っていた某有名ワークステーションメーカーのマシンは、キーボードがハングアップして操作不能になることが頻繁にあって、この**telnetでログイン&リブート**という「技」のお世話になることはしょっちゅうだった。いまのところ、PC互換機でLinuxを使っていて、キーボードのハングアップが原因で操作不能になったことは記憶にないが、いざというときはこのような手段もある、ということだけでも覚えておいたほうがよいだろう。

ちなみに、Linuxマシンにtelnetでログインして、マシンの管理作業を行いたい場合、ほとんどのディストリビューションでは、「rootユーザー」として直接ログインすることはできないようになっているはずだ。いったん一般ユーザーとしてログインしてから、**su**コマンドを使ってrootユーザーになる必要がある。



画面3 Windowsのtelnet.exeでログインしてMuleを使う
画面2のturbotimecfgはダメだったが、MuleやViといった「地味な」アプリケーションなら、Windowsからも問題なく使える。ただし、本文中で説明している「おまじない」が必要になる。telnet.exeのメニューから、「ターミナル」−「基本設定」を選択し、「エミュレーション」で「VT-100/漢字」を、「漢字コードセット」で「日本語EUC」を選択しておけば、日本語の表示やWindowsのFEPを使った漢字入力も可能になるはずだ。

もうひとつ、知っておいたほうがよいのは、Windows 95／98に付属するtelnet.exeを使う場合、そのままではMule、vi、lessなどの逆スクロール可能なアプリケーションはうまく表示できない。このようなコマンドを使いたい場合は、telnetでログインした後で、次の**おまじない**（コマンド）を実行するとよい。コマンドの意味については深く考えないこと。

・csh系のシェル（tcshなど）を使っている場合

```
% setenv TERM vt100
% resize
```

・bashを使っている場合

```
$ export TERM=vt100
$ resize
```

これは、Windowsに付属するtelnet.exeがtelnetプロトコルの一部を**省略**してしまっているのが原因らしい。Linuxのtelnetクライアントを使って別のLinuxマシンにログインするような場合は、このような操作は必要ない。Windowsの場合でも、Windows標準のtelnet.exe以外のターミナルエミュレータを使っている場合は、この操作は不要なことが多い。ちなみに、この「おまじない」の2行めの「resize」コマンドは、telnet使用時に限らず、Linuxのコマンドラインを使っていて、**どうも表示がおかしい**と感じたときに試してみると、症状が改善することがあるので、覚えておくと便利である。

manによるとresizeコマンドは本来、ターミナル設定の値を変更するシェルスクリプトを出力するだけで、本当はそれをファイルにしてシェルに喰わせてやるなどしないと設定が変わらないはずなのだが、なぜか私の環境ではう

まくいってしまう。興味のある方はman resizeをしてみてほしい。

万能ファイル転送「FTP」

コンピュータ同士がLANでつながって、リモートログインできるようになると、次にやりたくなるのが、お互いの間でのファイルのコピーだろう。もう少し気取った言い方をすれば、**ファイル転送**や**ファイル共有**ということになる。LinuxとWindowsマシンでのファイルの共有や転送というと、最初に触れたように**Sambaの出番**ということになるのだが、現在のLinuxとWindowsでは、前述のように多少の作業をしないと、Sambaは使える状態にはならない。また、Macintoshとの間でもファイル共有をしたい場合には、Sambaとは別にMacintosh用の設定もしなければならない。

もちろん、Sambaの設定を行って、WindowsからLinuxマシンのファイルを自由に使えるようにしたほうが便利だし、ぜひともそうすべきであるが、telnetの場合と同様に、何らかの理由でSambaが使えなくなってしまった場合や、Sambaで共有できるように設定していないディレクトリにあるファイルが欲しい場合にはどうしようもない。実際、Windows NTやWindows 2000をインストールしたマシンをLANに接続すると、設定の具合によってはWindows NT側からSambaのサーバにアクセスできなくなったり、最悪の場合、他のWindowsマシンからもSambaにアクセスできなくなることがある。Windows 2000の最新のベータ版では改善されているようだが、筆者の自宅の環境では、どうも安定しない気がする。このような場合でも、必要な最低限のファイルは転送できるようになっていると助かる。

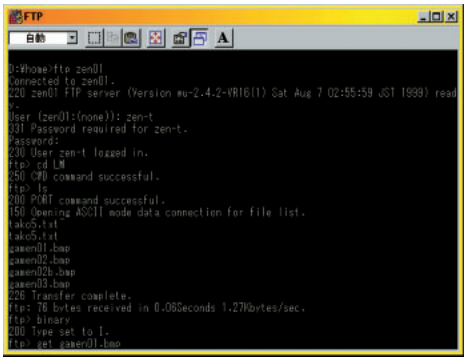
そこで登場するのが**FTP**である。FTPもtelnetと同じく、TCP/IP上で使えるネットワークプロトコルの一種で、FTPクライアントを使ってFTPサーバに接続し、クライアントマシンからサーバにファイルをコピーしたり、逆に、サーバマシン上のファイルをクライアントマシンにコピーできる。FTPのクライアントプログラムについても、telnetと同様に、ほとんどのOSに存在し、WindowsでもOSに標準で付属している（c:\windows\ftp.exeというコマンドファイルがそれ）。FTPサーバについては、Linuxでは標準の状態で使えるようになっており、特別な設定なしに使えるようになっている。前述のようなWindows NTとの相性の問題もなく、IPアドレスの設定さえまちがっていなければ、LANにつながっているだけで使えるので、非常時のファイル転送にも便利だ。

■テキストファイルとバイナリファイル

Windowsの場合、FTPのクライアントとして使えるのは前述のftp.exe（**画面4**）のほか、エクスプローラに似たGUIインターフェイスをもつフリーソフトウェアなどたくさんある。また、Internet ExplorerでもFTPによるファイル転送ができるのだが、こちらは「anonymous FTP」専用と考えたほうがいい（普通のFTPアクセスをする方法もあるのだが、手順が面倒なので実用的ではない）。

FTPでファイルを転送するときに注意すべきなのは、テキストファイルとバイナリファイルの扱いの違いだ。ご存知のようにLinux、Windows、Macintoshでは、それぞれのOSごとにテキストファイルの形式が違っている。このため、単純にファイルを転送しただけでは、テキストファイルが正しく読めないことがある。これに対応するため、FTPにはテキストファイルの形式を、転送先のOSに合わせて自動的に変換してくれる機能がある。FTPサーバのプロンプトで**ascii**と入力すると、テキストファイルを転送するモードになり、**binary**と入力するとテキストファイル以外（便宜的に「バイナリファイル」と呼ばれる）を転送するモードになる。もし、バイナリファイルをテキストモードの状態で転送すると、ファイルが途中で途切れたり、ファイルの内容が変更されてしまうことがあるので注意が必要だ。

なお、テキストモードでの自動変換も万能ではなく、漢字コードの種類やファイルの内容によっては、うまく変換できないこともあるようだ。FTPのテキストファイル変換機能にはたよらず（つまり、テキストファイルもバイナリモードで転送する）、ワープロやエディタのファイル形式変換機能を利用したほうが確実かもしれない。



画面4 WindowsのftpでLinuxマシンとファイルをやりとりする
Linuxマシンに設定してあるユーザー名とパスワードを使ってログインしてから、cd、put、getなどのコマンドを入力して必要なファイルをコピーする。ファイルをまとめてコピーするmput、mgetなどのコマンドもあるが、使い方にちょっとコツがいるので、これらについてはまたの機会にしたい。

日刊アスキーLinux on Linux magazine

日刊アスキー Linuxの裏舞台

～日刊アスキー Linux Weekly Newsletter～

先月号の当コーナーでお知らせした日刊アスキーのメールサービスが、ついに始まった。名称は「日刊アスキー Linux Weekly Newsletter」（以下Weekly）。毎週月曜日に、1週間分のニュースや日刊アスキー Linux内の更新情報をお伝えしようというものだ。もちろん無料である。

今回は、このメールサービスの内容と、システムの構成についてご紹介したいと思う。メールの内容をどうするか、どのようなメールが見やすいのか、といった企画面の話から、多数のメールアドレスへの配信といったシステム面まで、話の範囲は広い。まずは企画面にまつわる話から。

1週間のインデックスを

名前から見てもわかるとおり、Weeklyは週単位のメールサービスだ。“日刊”アスキーのものなのに“Weekly”というのもナニだが、1週間分のニュースをまとめ、カテゴリーごとに分類、整理したものなので、インデックスとして十分役に立つはずだ。これを読めば1週間のLinuxの動きがわかるし、資料として取っておき、将来調べものがある時はリンクをクリックしていただければ、該当記事が掲載されているWebページにジャンプしてもらうことができる。毎日毎日Webサイトにアクセスしている暇なんてない、といった方にもご利用いただけるだろう。ニュースは日々流れていくが、Weeklyはちょっと立ち止まって1週間を眺めるとともに、保存版として活用できるのである。

具体的なメニューは**表1**をご覧いただきたいのだが、各コーナーとも、Web上の日刊アスキー Linuxのメニューを基本とし、メールで見やすいように工夫している。たとえば、日刊アスキー Linuxの場合はニュースのカテゴライズはなされていないが、メールの場合は【一般】や【ソフトウェア】といったようにカテゴライズされている。実は、このカテゴライズは内部ではすでに記事作成時（XML記述時）に行われているのだ。日刊アスキー Linuxの記事がXMLによって作成され、最終的にはHTMLに変換、公開されているのは以前お話ししたとおりだが、<category>～</category>というエレメントが存在し、それがWeeklyのカテゴライズに反映されるのである。

これらのメニューについては、もともと日刊アスキー Linuxのものを踏襲するだけなので、それほど考えることはなかったのだが、記事1本1本の配置については、いろいろと議論したり、ほかのメールサービスなどを比較検討した。

まず、基本的にはニュース記事すべてを載せるのではなく、要約文を載せていこうというもの。当サイトのニュース記事すべてを載せていくと、とても長いメールになって一覧性が落ちる。よって、1つの記事は、

①タイトル
②該当記事のURL
③要約文

というシンプルな構成になった。タイトルや要約を読んで、もっと詳しく知

1　今週の注目記事	その週に掲載した注目記事の要約
2　ニュースダイジェスト	【一般】企業の提携や合併、方針説明など 【ソフトウェア】新製品情報やβ版情報など 【イベントレポート】イベントの取材記事
3　連載コラム	その週に掲載されたコラムの要約
4　Linux関連書籍ベスト10	「Linux関連書籍 今これが売れている！」 のベストテンと要約
5　イベントカレンダー	近々行われるイベントのご案内
6　アスキーの雑誌のLinux情報	Linux magazineをはじめとした、アスキーの雑誌のLinux情報

表1 日刊アスキー Linux Weekly Newsletterの主なメニュー

りたい人は、直接②のURLに飛んでWebページを参照していただけばよいし、1週間のニュースをザッと眺めたい方は、メールのみを読んでもらえばいい。

実は、要約文については、元の記事作成時（Web用記事作成時）に、<summary>～</summary>というエレメントに要約を書いておき、これが紹介文としてメールに記載されるという仕組みをとっている。この<summary>エレメントに書かれた文章は、いままで一部のコンテンツにしか反映されていなかった。あくまでも「将来のための」エレメントで、想定される使い方は別にあった（とシステム担当は語っている）のだが、今回のWeeklyで大々的に使われるようになったのである。

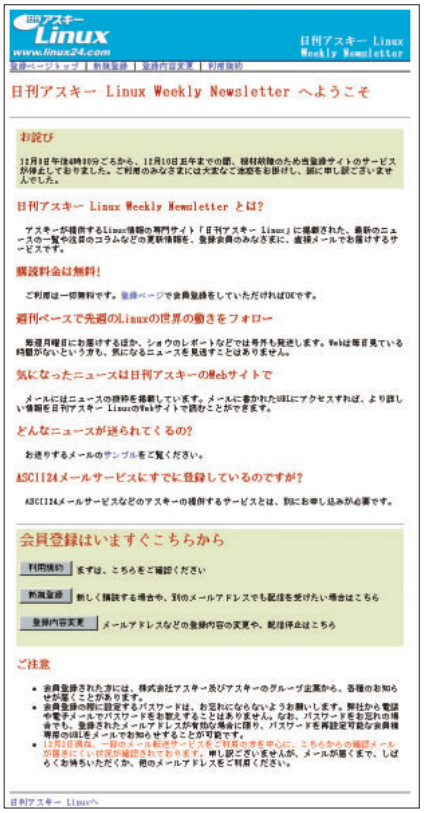
これまで外に出ない（Webページに掲載されない）要素だったので、記事本文のコピー&ペーストでまかなっていたり、場合によっては記述しなかったこともあった。今後はメールニュース充実のためにも、短いながらも的確にニュースの全容がわかるような<summary>にしていきたいと考えているが、実はこれがくせ者で、なるべく多くの情報を的確に書こうとすると、せいぜい80文字くらいの文章なのに、長時間考え込んでしまうこともある。

原稿執筆時点では、ちょうどWeeklyの創刊号を出し終えたところである。作業そのものは、日曜日の深夜に行った。フジテレビがK-1グランプリ'99決勝戦を放映していた時間だ。武蔵が転倒のアクシデントでフィリポビッチの足蹴りを顔面にくらった頃に作業を始め、ピーター・アーツとJ・レ・バンナのアツい一戦の頃には、編集作業はほぼ終わっていた。そして、フィリポビッチがホーストの一撃でマットに沈んだ時はすでに、メールの配信が始まっていたのだった。それでは、順を追って作業の内容をご紹介しよう。

まず、システムが各記事の要約を抜き出し、メール本文が作られる（**画面1**）。今回は初回なので、システム担当者からの指示に従って、編集者がメール本文をチェック。要約文の見直しなど、内容をところどころ手直しした。本来ならば、元の原稿を書いた段階でメール本文の内容も完成しているはずだが、やはり要約文（<summary>）に誤字脱字があったり、要約文そのものがない記事もあり、手直しした。今後はこうしたことは減るだろう。

一方、システム担当者はメールに広告を挿入し、メール本文の再チェックをした。以前お話ししたとおり、日刊アスキーのシステム担当者はもともと編集者であり、さらに技術的にも詳しいのでありがたい。あとは、メールの配送を始めるだけである。

画面1
サンプル画面。これも部内で使用するWebアプリケーションとなっている。



画面2
メールアドレス登録Webサイト。日刊アスキーLinux Weekly Newsletterは、本体の日刊アスキーLinuxと同じく無料である。

こうして、日刊アスキーLinuxが贈る「日刊アスキー Linux Weekly Newsletter 創刊号」が完成した。今後は、毎週月曜日に皆さんのお手元に届くようになるので、Linux関連ニュースに興味のある方は、ぜひ登録していただきたい。

メールニュースシステムの中身

それでは次に、メールニュースシステムの中身について触れよう。

システムの構築は、すべて部内で行われた。正確にいうと、システム担当者が1人で作り上げたものだ。彼がこの原稿用に、とメモを作成してくれたので、それを元にシステムの構成から、どのようなアクシデントが起こったのか、といったお話をご紹介する。大規模メール配送システムの事例として、参考にしていただければ幸いである。

メールニュースシステムは、3つのシ

ステムに分けて考えることができる。ユーザーがWeekly配信の申込登録をする「メールアドレス登録Webサイト」、編集部員がWeeklyを編集するための「メール生成／編集システム」（先ほどの**画面1**を生成するものだ）、そして最後に、Weeklyを配信する「メール配信システム」である。これらはすべてオープンソースソフトウェア（フリーソフト）で構築されている。

●メールアドレス登録Webサイト
http://www.linux24.com/mailnews/

Weeklyの登録システムである。新規申し込みだけではなく、配送先メールアドレスの変更／中止もここで行われる（**画面2**）。我々が留意しなければならなかった点として、いたずらで他人のメールアドレスを勝手に登録されてしまうことだ。また、ユーザーが誤ったメールアドレスを登録してしまい、到達不能なメールアドレスへWeeklyを出し続ける、といった事態も防がねばならない。メールアドレスの登録ミスはかなりの頻度で起こっているのだ。

こうした点に対しては、システムが新規申込時やメールアドレスの変更を受け取った時点で、ユーザーに対して確認のメールを送信する仕組みを作ることで対処している。具体的な手順としては、

1.ユーザーがメールアドレス登録サイトで申し込みを行う
2.ユーザー宛てに確認メールが届く
3.ユーザーは確認メールに記述されたURLにアクセスし、最終確認を行う

というステップになっている。確認メールには、各ユーザー固有のURLが記述されており、このURLにアクセスしない限り、登録は完了しない。逆に、アクセスさえすれば、登録は完了になる。登録の覚えがないユーザーは、この時点で登録作業を中断すればいいし、間違ったメールアドレスを登録してしまった場合は、確認メールが到達しないので、入力ミスがわかるわけだ。

このメールアドレス登録Webサイトには、以下のソフトウェアが使用されている。

- **Apache 1.3.9（Webサーバ）**
- **PHP 3.0.12jp-beta 4（アプリケーション開発言語／処理系）**
- **PostgreSQL 6.5.3（データベース管理システム）**

アプリケーション開発言語／処理系にPHP3を選択したことによって、DNSへの問い合わせなども容易にコーディング可能となり、入力されたメールアドレスの正当性のチェックなども簡単に実装できている。また、フォームなどのデータ処理も容易である。

現在使用しているPHP 3.0.12jp-beta 4は、有志の手によるPHP3の国際化バージョンだ。日刊アスキー Linuxでは、国際化（日本語）バージョンが提供する半角カナ→全角カナ変換や、全角英数→半角英数変換などの機能を利用している。http://php.jpnnet.com/に詳しい紹介があるので、参照していただきたい。

システムのポイントとしては、アドレスの入力ミスに可能な限り対応策をとっていることだ。ユーザーがメールアドレスを入力した時点で、自動的に全角→半角変換を行い、“ @ ”以降について、MX、Aリソースレコードの存在を確認するようになっている。こうすることで、確認メールを送付する前に、メールアドレスの入力ミスを極力少なくしているのだ（“ @ ”以前に関しては、フォームの入力時にチェックは行わない）。信じられないかもしれないが、全角でメールアドレスを入力してしまう人も少なくない。また、パスワードを忘れた場合は、ユーザー専用のURLを生成しメールで送付する。このURLでパスワードの再設定や送信先の変更などが可能になる。この生成したURLは、いつまでも残していたらセキュリティ上問題があるので、一定時間を過ぎると利用できなくなるようになっている。

メールサービスで、登録後に「あなたのパスワードは“ abcdefg ”です」というメールが送られてくるサイトを見かけるが、やはりユーザーとしては自分のパスワードがメールで送られてくるのはイヤなものだろう。Weeklyのシステムでは、こうした事態を防ぐために、ユーザーそれぞれに固有のURLを生成し、対処しているのである。

だが、システム担当によると、ここに落とし穴があったそうだ。サービス開始当初、生成したURLは、“ ？ ”“ ＆ ”“ ＝ ”といった文字を使用してパラメータを記述していたが、Windows版Eudora Proを使っているユーザーから、「メールに書かれたホームページ（生成されたURL）にアクセスできない」といったメッセージをいただいたのである。担当者が調べたところ、どうやらWindows版 Eudora Proの場合、“ ？ ”“ ＆ ”“ ＝ ”といった文字列が記述されている場合、Webブラウザにの一部しか渡さないような仕様になっているらしい。そこで、ユーザーに渡すURLの型式を“ / ”のみを使った、単なるURL指定と同じ型式に変更した。

これには、Apacheの「Rewrite Engine（mod_rewrite）」といった機

能が使われている。RewriteEngineは、受け取ったURLを置換する働きをするので、いったん“/”を使ったURLをApacheが受け取り、RewriteEngineでもとの“？”“＆”“＝”などを使ったスタイルに置き換え、アクセスさせる。こうすれば、いままでの型式との互換性も保たれる。最初のフォーマットのメールでの登録完了処理も問題なく行えるわけだ。

メール生成／編集システム

これは編集部内からしかアクセスできない。システム構成としては、先ほどと同じApache、PHP3、PostgreSQLを使っている。

PHP3は、国際化（日本語）バージョン）で用意されている日本語の文字列処理機能を使い、簡単な禁則処理を行っている。PostgreSQLは日刊アスキーの記事のデータを管理する中枢部分ともいえ、記事本文／画像を外部のファイルとして保存し、要約文を抜き出す用途に使用される。

メール配送システム

これは、登録された送信先に対し、メールを送信する仕組みだ。送信先リストは、PerlによりPostgreSQLからデータを引き出して生成するようになっている。使われているソフトウェアは、

- **sendmail＋SMTPfeed**
- **Perl**

である。

SMTPfeed（SMTP Fast Exploding External Deliverer）は、SMTPの配送のみを行うエージェントで、メーリングリストのように、あて先が多数ある場合に使う。通常のsendmailに、複数の送信先を指定してメールを送ると、送信処理を送信先メールサーバごとに順次行っていく。そのため、送信先の一部にサーバの不調などがあると、その宛先の処理がタイムアウトするまで次の宛先の処理に移らないため、大量の配信には思いがけない時間がかかることがある。SMTPfeedは、LMTP（Local Mail Transfer Protocol、RFC2033）によって、sendmailから呼ばれる外部メーラとして動作する。SMTPfeedは、DNSの問い合わせを同時に行い、DNSへの問い合わせ時間を短縮するとともに、SMTPの送信処理も複数ホストに対して同時に行うことで、配送時間を短縮している。詳しくは、SMTPfeedのWebページをご覧いただきたい（http://www.kyoto.wide.ad.jp/smtpfeed/）。

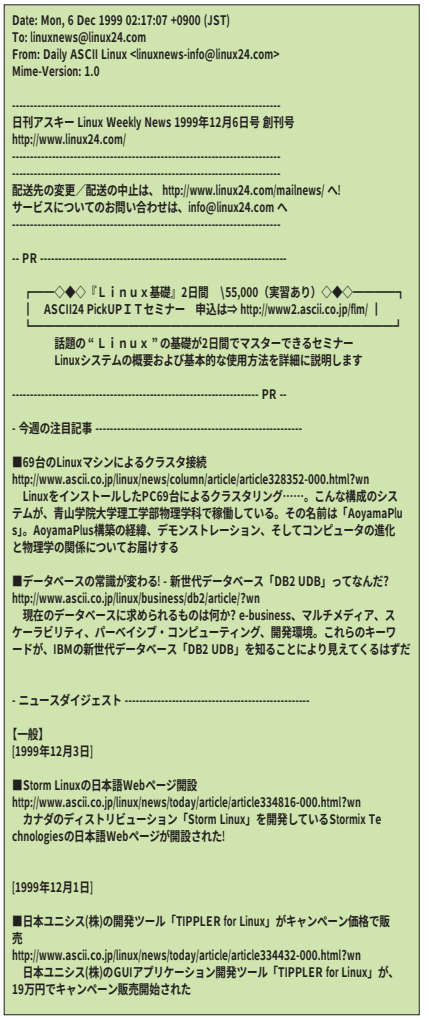
PerlはPostgreSQLに問い合わせをして配送リストを生成し、送信などを行う処理に利用している。

以上が、Weeklyのシステムの概要だ。これで万事うまくいったし、問題もなかったのだが、運用開始後しばらくたった12月8日に、メールアドレス登録Webサイトにアクセスできないというトラブルが発生した。実は、この不調はシステムとは何ら関わりがなく、HDDの故障が原因だった。どうやらHDDのケースの放熱に問題があったようだ。もともと弊社で多数導入していた外付けHDDの1台なのだが、同時に購入したほかのHDDも多数故障した実績があったのである。ただ、登録データは定期的にバックアップされており、HDDの故障自体には影響を受けなかったので、まったく無事である。ご登録いただいた方にWeeklyが届かない、といった事態はないのでご安心いただきたい。現在はすべて復旧し、メールアドレス登録Webサイトも元気に稼働中である。

●

もちろん日刊アスキー Linuxの本体はあくまでもWebサイトであるが、このメールサービスによって、ますます日刊アスキー Linuxの、情報提供サイトとしての強化が図れたと思う。今後は大きなニュースが出た際には号外としても活用するつもりだ。1週間の動きを一覧できるインデックスとして、重大ニュースを見逃さないためのチェック機構として、ぜひ活用していただければ幸いである。

（日刊アスキー Linux編集部・吉川）



```
Date: Mon, 6 Dec 1999 02:17:07 +0900 (JST)
To: linuxnews@linux24.com
From: Daily ASCII Linux <linuxnews-info@linux24.com>
Mime-Version: 1.0

--------------------------------------------------------------------
日刊アスキー Linux Weekly News 1999年12月6日号 創刊号
http://www.linux24.com/
--------------------------------------------------------------------
--------------------------------------------------------------------
配送先の変更／配送の中止は、http://www.linux24.com/mailnews/ へ!
サービスについてのお問い合わせは、info@linux24.com へ
--------------------------------------------------------------------

-- PR ----------------------------------------------------------

  ┌──◇◆◇『Ｌｉｎｕｘ基礎』2日間　\55,000（実習あり）◇◆◇──┐
  |  ASCII24 PickUP ＩＴセミナー　申込は⇒ http://www2.ascii.co.jp/ftm/  |
  └──────────────────────────────────┘

      話題の“Ｌｉｎｕｘ”の基礎が2日間でマスターできるセミナー
      Linuxシステムの概要および基本的な使用方法を詳細に説明します

---------------------------------------------------------- PR --

- 今週の注目記事 ---------------------------------------------

■69台のLinuxマシンによるクラスタ接続
http://www.ascii.co.jp/linux/news/column/article/article328352-000.html?wn
　LinuxをインストールしたPC69台によるクラスタリング……。こんな構成のシステムが、青山学院大学理工学部物理学科で稼働している。その名前は「AoyamaPlus」。AoyamaPlus構築の経緯、デモンストレーション、そしてコンピュータの進化と物理学の関係についてお届けする

■データベースの常識が変わる! - 新世代データベース「DB2 UDB」ってなんだ?
http://www.ascii.co.jp/linux/business/db2/article/?wn
　現在のデータベースに求められるものは何か? e-business、マルチメディア、スケーラビリティ、バーベイシブ・コンピューティング、開発環境。これらのキーワードが、IBMの新世代データベース「DB2 UDB」を知ることにより見えてくるはずだ


- ニュースダイジェスト ---------------------------------------

【一般】
[1999年12月3日]

■Storm Linuxの日本語Webページ開設
http://www.ascii.co.jp/linux/news/today/article/article334816-000.html?wn
　カナダのディストリビューション「Storm Linux」を開発しているStormix Technologiesの日本語Webページが開設された!


[1999年12月1日]

■日本ユニシス(株)の開発ツール「TIPPLER for Linux」がキャンペーン価格で販売
http://www.ascii.co.jp/linux/news/today/article/article334432-000.html?wn
　日本ユニシス(株)のGUIアプリケーション開発ツール「TIPPLER for Linux」が、19万円でキャンペーン販売開始された
```

日刊アスキー Linux創刊号

日出づる国のグループウェア

http://cybozu.co.jp/

新バージョン登場!

サイボウズOffice3 for Linux

Webグループウェアで仕事をスムーズに

1人1台のPC環境とネットワーク化が進み、仕事の生産性を高めるためにもインターネットが使われ始めている。これまでは、ビジネス資料を集めるWebサーフィン、コミュニケーションの電子メールが主流であった。それをさらに進めて、グループで効率的に仕事をこなすためのツール、「グループウェア」を導入する企業が増えている。なかでも、プラットフォームを限定しない、Webベースのグループウェアが注目を集めている。そのひとつ、サイボウズがOffice2からOffice3に大きくバージョンアップされた。もちろんLinuxサーバでも動作する。

Webブラウザさえあればok

サイボウズOffice3は特別なクライアントソフトを必要とせず、ふだん使っているWebブラウザだけで情報のやりとりができる。オフィスの自分のデスクからだけでなく、ほかの部署のマシンからでも利用可能だ。ダイヤルアップサーバを用意すれば、自宅や携帯端末からのアクセスだって可能だ。

インターフェイスがWebなので、操作も簡単。クライアントのOSやWebブラウザの種類を選ばないため、既存環境を変えずに導入できることで、新たなマシンやソフトの導入コストをゼロにできる。これらのメリットは2000社への導入実績が裏付けている。

紙やホワイトボードをデジタルに

サイボウズOffice3が持つ機能は多彩だ。今回は数ある機能の中で、個々人の作業効率を高めるソリューションを中心に見ていこう。

・スケジュール

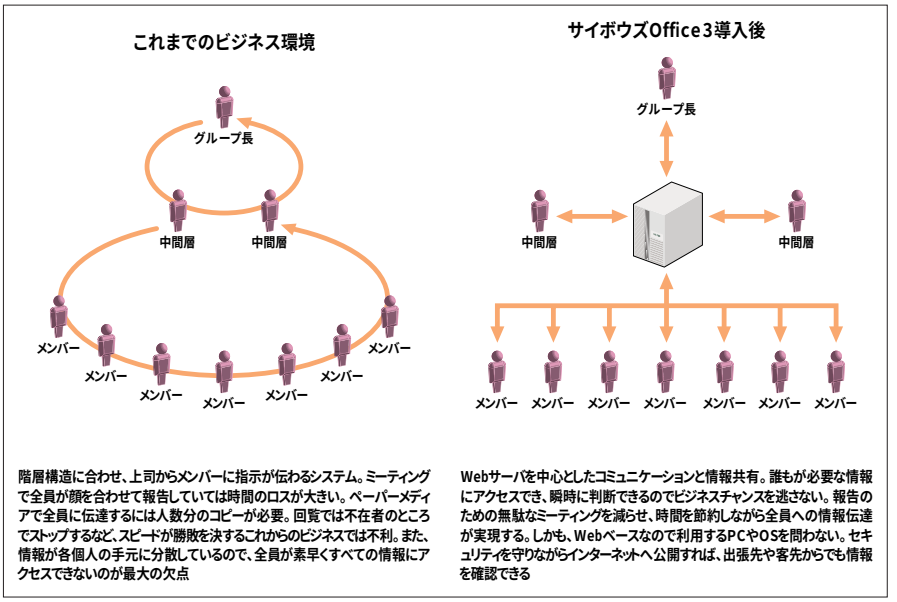
個人スケジュールとグループ全体のスケジュールを管理、確認できる共有スケジュール帳だ。会議のスケジューリングなどに重宝する。個人的な内容は、非公開項目に設定できるため、プライバシーも守れる設計だ。

・行き先案内版

行き先案内板をクリックして、メンバーの行き先を閲覧できる機能だ。ホワイトボードの代わりをなす機能だが、ここで着目したいのは、不在者に対して電話などのメモを残せることだ。行き先案内版で行動を確認したら、そのままWebで担当者宛にメモを残しておける。書き込んだメモは、電話メモ機能で閲覧する以外に、メールでも受け取れる。メモの送り先を携帯電話のアドレスにしておけば、どこからでも不在時の電話に対応できるだろう。

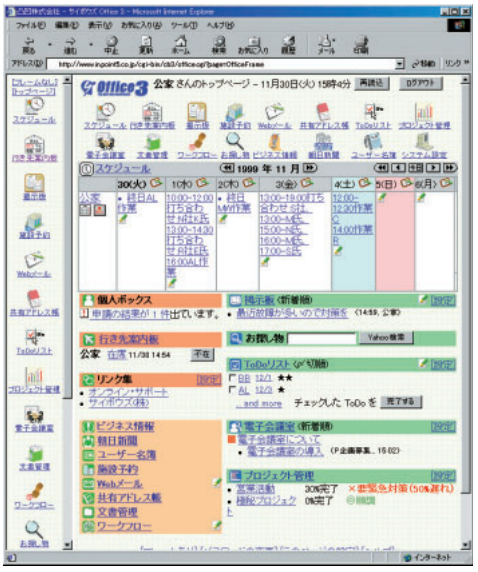
・掲示板、施設予約、プロジェクト管理

グループ内で共有したい機能が掲示板と施設予約である。掲示板を使って部内回覧をすれば、過去の掲示内容も検索できる。また、施設予約は、会議室や打ち合わせスペースを予約できる仕組みだ。自分のPCから目的の施設の利用状態がわかり、使いたいところが空いていればすぐに予約ができる。打

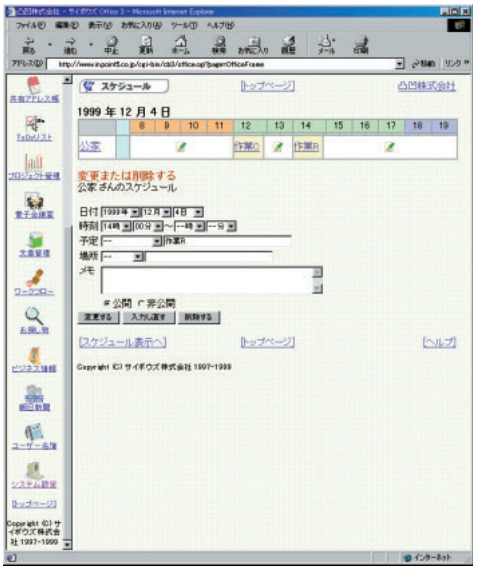


階層構造に合わせ、上司からメンバーに指示が伝わるシステム。ミーティングで全員が顔を合わせて報告していては時間のロスが大きい。ペーパーメディアで全員に伝達するには人数分のコピーが必要。回覧では不在者のところでストップするなど、スピードが勝敗を決するこれからのビジネスでは不利。また、情報が各個人の手元に分散しているので、全員が素早くすべての情報にアクセスできないのが最大の欠点

Webサーバを中心としたコミュニケーションと情報共有。誰もが必要な情報にアクセスでき、瞬時に判断できるのでビジネスチャンスを逃さない。報告のための無駄なミーティングを減らせ、時間を節約しながら全員への情報伝達が実現する。しかも、Webベースなので利用するPCやOSを問わない。セキュリティを守りながらインターネットへ公開すれば、出張先や客先からでも情報を確認できる



個人トップページの例
右フレームの掲示板や電子会議室のタイトル右側の［設定］をクリックすると、表示内容を個人単位でカスタマイズできる。小さな画面なら、左側のフレームをオフにして、必要な項目だけを表示するような設定にするといい



スケジュール入力画面
グループウェアとはいえども、守らなければならない個人情報は守る。スケジュールには、非公開項目があり、自分以外は見られない。デートや飲み会だってモチベーションアップには必須なのだ

ち合わせがキャンセルになったときでも、自分のPCから予約解除ができるため、共有スペースの効率利用につながるはずだ。そして、プロジェクト管理は、グループ全体で動いている大きなプロジェクトを閲覧、管理する機能である。プロジェクトの達成度が目で確認できる点がもっとも大きい。

・共有アドレス帳

あいうえお順や検索機能を持ったアドレス帳だ。住所、電話番号やメールアドレスは、これまでは各個人が名刺やメールソフトで管理していたが、取引先アドレスなど共用できるものはできるだけシェアして、入力の手間や名刺をひっくり返す時間を節約しよう。

・ToDoリスト、Webメール

個人ベースでの利用に限定した機能が、ToDoリストとOffice3で加わったWebメールだ。ToDoリストは自分が「忘れずにやらなければならないこと」のリストで、いくつもの仕事を並行して進めているビジネスマンにとってはありがたい。一方のWebメールは、特定のメールソフトを使わずに、Webブラウザからメールを読み書きする機能だ。どの端末からでもメールを読み書きできるのが最大のメリットだ。なお、WebメールにはSMTP/POP3に対応したメールサーバを別途用意する必要がある。

・利用者に合わせたカスタマイズもできる

Office3ではいろいろな機能が加わっている。もっとも大きな変化は、個人のトップページで、スケジュールやメールの有無など表示する項目を自由にカスタマイズできることだ。自分が使いやすいように設定することで、ホームページを開くのも楽しくなる。

まず体験版で実力を知る

サーバの組み込みというと、難しそうな感覚がある。しかし、サイボウズOffice3は、HTTPサーバの下で動作するCGIだ。インストールもわずかな作業ですむ。インストールが終われば、ブラウザから詳細を設定していくだけで、いつでもグループウェアサービスが始められる。

また、多種多様の機能のすべてを購入したりインストールする必要はなく、必要に応じて選べるところもいい。

Webベースのグループウェアの機能やセキュリティに不安を感じるならば、実際にダウンロードして体験版を試すといい。おそらく、予想以上の便利さを実感できるはずだ。

・ダウンロードURL

http://cybozu.co.jp/download/unix.html

サイボウズ株式会社
〒530-0001 大阪府大阪市北区梅田2-4-13阪神産経桜橋ビル 5F
製品の詳細情報
http://cybozu.co.jp/cb3/

サイボウズOffice3標準価格（税別）

※「サイボウズOfficeパックEX3」「サイボウズOffice SOHO」はA～Jのセットパッケージです（ワークフロー3は含まれません）。
※「サイボウズ Officeパック3」はA～Eのセットパッケージです。
※同一商品でユーザー数を増やされる場合は、差額のみで対応いたします。

製品名	50ユーザー	100ユーザー	ユーザー数無制限
サイボウズOfficeパックEX3	198,000円	380,000円	880,000円
サイボウズOffice SOHO	79,800円 ※10ユーザー		
サイボウズOfficeパック3	99,800円	198,000円	480,000円
A.サイボウズ スケジュール3	49,800円	99,000円	198,000円
B.サイボウズ 行き先案内板3	29,800円	49,000円	98,000円
C.サイボウズ 掲示板3	49,800円	99,000円	198,000円
D.サイボウズ 施設予約3	49,800円	99,000円	198,000円
E.サイボウズ Webメール3	49,800円	99,000円	198,000円
F.サイボウズ 共有アドレス帳3	29,800円	49,000円	98,000円
G.サイボウズ ToDoリスト3	29,800円	49,000円	98,000円
H.サイボウズ プロジェクト管理3	59,800円	99,000円	198,000円
I. サイボウズ 電子会議室 3	49,800円	99,000円	198,000円
J. サイボウズ 文書管理3	49,800円	99,000円	198,000円
K.サイボウズ ワークフロー3	99,800円	198,000円	480,000円

viバージンに贈る超入門講座

viはじめました

最終回　いよいよ千秋楽！

風邪など引かずにフリー系UNIXしてますか?さてこの連載も第6回、いよいよ最終回となってしまいました。駆け足ながら、viの本当の初歩から、それなりに高度な使い方までをひと通り見てきました。最終回の今回は、各種viで共通に備えている機能のうち未紹介のもの、そしてvim特有の機能のうちマルチウィンドウによる編集、そして実用的な用法のいくつかを『詰めviスペシャル』の形でお届けします。

文：佐々木太良
Text : Taroh Sasaki

illustration ; Manami Kato

●どうでもよさそう（ごめん）なコマンド

今までの回で言及しなかったコマンドのうち、少しでも役に立ちそうなもの、面白いものを取り上げましょう。

・大文字・小文字の変換

viコマンド[~]でアルファベットの大文字・小文字を相互変換できます。viのバージョンによっては範囲指定コマンドと組み合わせて使うことができた（ように筆者は記憶している）のですが、現在のviの多くのバージョンでは『数値＋[~]』の組み合わせしかできないようです。

・行内の変更（次のページ図1）

viコマンド[c]は、特定範囲の文字を入力文字列（[ESC]をタイプするまで）と置き換えるものです。特によく使うのは[c][c]のパターンでしょうが、[d][d][i]と変わりません。また[c][$]（つまり行末までを変更）は[C]とタイプしても同様です。

[r]系のコマンドはちょっと[c]系とは異なります。単体で[r]を打った場合、カーソル位置の文字が[r]の直後に打った文字と入れ替わるだけです（が、これが場合によっては結構便利です）。数字＋[r]では同じ文字が何文字かに置き換わってしまうだけです。それより便利なのが[R]コマンドで、このコマンドではとにかく[ESC]を打つまでの入力で、既存テキストを置き換えていきます（つまり上書き入力になる）。書き換え語の文字数が書き換え前と同じであることが直感的にわかっている場合など、筆者は多用します。

・タブのシフト

これはプログラマー向け機能でしょう。『数字[>]範囲』で『範囲』のタブが1段深くなり、『数字[<]範囲』で浅くなります。もちろん行に作用するコマンドですので、範囲指定は異なる行でなければ意味がありません（[>][l]としても[>][>]と同じ意味になるということです）。

・対応するカッコへの移動

これもプログラマー向け（それもLispプログラマー？）です。カッコの上で『%』を打ってみてください。見ての通りの単純な機能なのですが、かなり使いではあります。括弧として認識されるのは『(…)』、『{…}』、『[…]』（nviなどでは『<…>』も）です。

実はこのコマンドが便利なのは、他のコマンドとの組み合わせの際の移動先としてです（**図2**）。

・exコマンドの連続実行

exコマンドの行内に複数のコマンドを書きたい場合、たとえば、

```
[:]w|ne[Enter]
```

のようにして、

```
[:]w[Enter][:]ne[Enter]
```

を連続で実行したのと同じ効果を得ることができます。確かに便利なのですが、このためコマンドに『|』文字が含まれるものが実行できないという不利な点もあります。前回の『詰めvi』地図エディタでは縦線を小文字の『L』で表わしていましたが、これは、

```
map J r|j
```

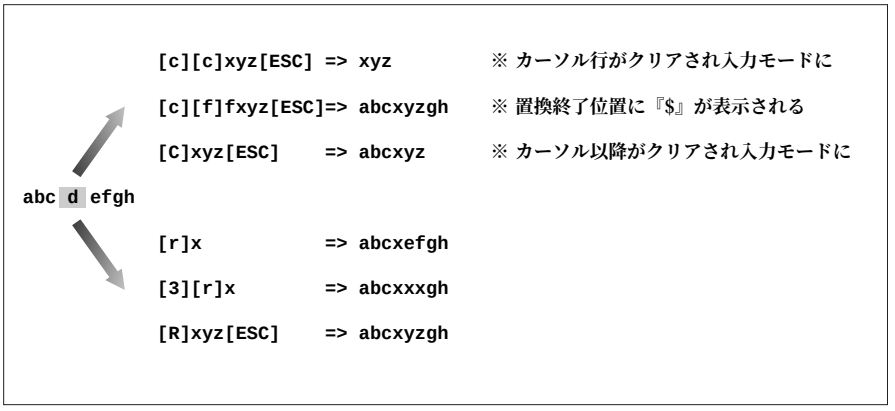


図1 [c]系コマンドと[r]系コマンド

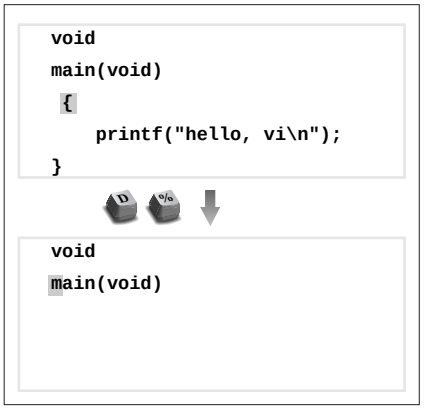


図2 %による範囲の指定

のようにキーマップをしようとすると、これが『map J r』『j』という2つのコマンドを連続実行したものとみなされてしまうからです（.exrcファイルに書かれたものは、exコマンド行で実行したのと同じ効果を持ちます）。

●マルチウィンドウなvi

Linuxの多くのディストリビューションで採用しているvimでは、マルチ

Column

番外編：いざとなったらed

連載の最初のころ『viは十分「重い」エディタなので、非常時に使うのは「?」である』と書きました。では本当にviが使えない場合とはどんな場合で、その場合どうしたらよいのでしょうか。

簡単にいえばviが使えない場合とは、viのコマンド実体があるディレクトリがマウントできない場合のほか、一時ファイルを作る場所がマウントできない場合などがあります。

ただし現在のLinuxでのインプリメントでは、viは/bin/vi（すなわちルートディレクトリにある場合が多い）ですし、一時ファイルも/tmpに作ることができます。

もしマウントができないといっても、実際にファイルシステムが壊れているとは限りません。たとえば/etc/fstabを壊してしまって、シングルユーザーモードで立ち上げ直してviを使った後、元に戻したい場合、手動で、

```
# mount -a
```

または、

```
# mount /dev/sdaX /bin
# mount /dev/sdaY /var
```

などのようにすればviは使えるようになるでしょう（OSのインストール方法によって異なります）。

なおルートファイルシステム（/）がマウントできず、読み出し専用でマウントされている場合は（/etc/fstabの書き換えができません）、

```
# mount -w /
```

が有効でしょう。

さて何らかの理由で本当にviが使えない場合、viの祖先である『ed』を試してみましょう。edはスクリーンエディタではなく、ラインエディタです。どういうことかというと、編集対象が全部画面上に表示されているわけではなく、カーソルが画面上を自由自在に動き回って……というわけにもいきません。

ではどうすればいいの？ と言われそうですが、この連載最終回のレベルまでくれば、皆さんにはedの操作方法がviの中のexコマンドと酷似していることに気付くはずです（ふだんexモードも使っている人なら完璧ですね）。

rootにならずに一般ユーザーで演習をしてみましょう（セーブできないので安全です）。

```
# ed /etc/fstab
242
```

242というのは、バイト数を表現しています。頭の中に/etc/fstabを描いて、現在カーソルが最終行にあるところを思い浮かべてください。まず『1』を入力して1行目にカーソルを移動します。この状態でリターンを打っていくと、カーソルが2行目、3行目……と移動して移動先の行が表示されるのですが、たとえば『wda4』という文字列を含む行に移動したいなら、いきなり『[/]wda4』（コマンドの最後の[Enter]は省略しています）と打てばOKです。viと同じですね。

```
/dev/wda4   /uar    ufs rw  2   2
```

おやおや、ここは『/usr』にマウントしたかったのに、『/uar』になっていますね。現在カーソルはこの行にあるので、viのexコマンド『s』と同様『s/uar/usr/』とタイプして置換します。置換後の結果は表示されないので、不安ならば『.』をタイプして、

```
/dev/wda4   /usr    ufs rw  2   2
```

のような表示を得ます。あとは『w』『q』を実行してセーブ・終了してください。

詳しくはmanを見ていただきたいのですが、ちょっと覚えておけばどうです、exコマンドのノリで使えるということが分かるでしょう。

ウィンドウによる編集ができます。これまでの回では「ちょっと邪道かな」と思われるので取り上げてきませんでしたが、最終回なので軽く触れてみましょう。なお本章の内容だけは他のバージョンのviに当てはまらないので気をつけてください。

まずマルチウィンドウということについてですが、今さらウィンドウがたくさん開くアプリケーションといっても珍しくないでしょう。ただし基本的にvimのマルチウィンドウは、『1つのウィンドウを横に複数分割する』ということを指していると考えてください。たとえばファイル/etc/passwdを編集中（**図3**-a）にウィンドウを2つに分割する（[^W][s]。^はCtrl）と、**図3**-bのようになります。ほかのウィンドウに移動するのは[^W][^W]です（**図3**-c）。ウィンドウ操作のためのviコマンドははすべて[^W]で始まっているのでわかりやすいですね。

「今やウィンドウシステムの時代だよ。わざわざエディタが複数ウィンドウ扱えなくったって」。ごもっともであります。X上でktermを2つ開いているならば、それぞれのウィンドウでviを別々に立ち上げてもよさそうですよね。

まあviを立ち上げるのはXなどのウィンドウシステムが動作していない場合（コンソール上や、telnet先など）もあるので、1つのウィンドウが2つに分割できるのはありがたい場合もあります。

ただし同一ファイルを2つに分割するのは非常に意味があります。異なるファイルAとファイルBを2つ編集するならば、ktermを縦に2つ並べて別々に『vi A』『vi B』と起動したほうが、なんぼか使い良いでしょう（ウィンドウも狭くならないし）。

しかし**図3**-eのように、同一ファイル

a.『vi /etc/passwd』直後（アミがけはカーソルのある位置）

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
/etc/passwd [+][RO]
```

b.分割すると

Ctrl + W s

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
/etc/passwd [+][RO]
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
/etc/passwd [+][RO]
```

c.もう1つのウィンドウに移ろう

Ctrl + W　Ctrl + W

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
/etc/passwd [+][RO]
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
/etc/passwd [+][RO]
```

d.複数ウィンドウ間の編集内容の同期

S b c w sh ESC

```
root:x:0:0:root:/root:/bin/sh  ←― この行も同時に変更されている
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
/etc/passwd [+][RO]
root:x:0:0:root:/root:/bin/sh
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
/etc/passwd [+][RO]
```

e. 別の場所を見よう

G

```
root:x:0:0:root:/root:/bin/sh
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
/etc/passwd [+][RO]
xfs:x:100:233:X Font Server:/etc/X11/fs:/bin/false
fml:x:506:506::/home/fml:/bin/bash
postgres:x:40:234:PostgreSQL Server:/var/lib/pgsql:/bin/bash
/etc/passwd [+][RO]
```

図3 vimのマルチウィンドウのようす（端末は10行表示と仮定しています）

の複数の場所を一覧したい場合は、2枚のktermでは真似できません。すでに述べたように後から起動したviがロックしてしまいます。**図3**-dを見てください。同一ファイルを別々のviで編集していては、このように修正箇所が他方に反映されるということはありません。

さて、**図3**の最下行はコマンド行（カーソルがどのウィンドウにいても共用される）、その上の行はステータス行（ウィンドウごとに用意される）です。このデザインはemacs系のエディタを意識しているといえますね。

主なウィンドウ操作のコマンドは、末尾の表に掲載しています。これらのコマンドはすべて、exコマンドでも対応するものがあるのですが、表では省略しています。

また複数ウィンドウを開いている場合、エディタの終了（[:]qなど）がウィンドウを閉じる操作になってしまうなど、微妙に動作が異なってくるので注意が必要です。まあ、使っているうちにすぐに慣れることだとは思いますが。

これ以上の機能については本連載では取り上げません。マニュアルやヘルプ（[:]help）を使って自分で調べてみてください。

●さよならの前に

今まで取り上げてきたコマンド（実は取り上げていないものも含まれていますが、働きはすぐに推測できるでしょう）を最後に掲載します。本連載はマニュアルと異なるので、すべてのコマンドを網羅しているわけではありません。また、随所で述べた理由によって、あるviのバージョン特有のコマンドはあえて飛ばしたりした部分もあります。

しかしツールというのは何でも、使っているうちに「お、こんな機能もあるんだな」と気付くこともありますし、それが便利だと思ったら使いこなすように工夫することを忘れないでください。

では皆さん、happy viingを忘れずに‼

＜詰めvi＞

●詰めviスペシャル

今回はかなり実用的な『詰めvi』をいくつかお贈りします。本文中で詳細な解説もしますが、正解を見る前にまず、自分で考えてみてください。

（問題1）編集中のファイルを『hoge@moge.com』さんに送りたいとします。これができればO＊tloo＊要らず？
（ヒント）もちろんセーブしてメールしてもよいのですが……。
（問題2）四角く切り取りましょう。図中の空白はスペースだとします。せめてフィールド（空白文字で区切られた文字列）に分かれていれば、正規表現やawkの併用も可能なのですが……（こんなデータベースを作る奴が悪い？）。
（ヒント）man cut

（問題3）cshのマニュアル中の単語の出現頻度ベスト100の表を作ってください（起動直後の状態からviのバッファ中に作るところまで。セーブ不要）。
（ヒント）
・ステップ1：『man csh』の結果を空のviバッファに読み込みます。
・ステップ2：cshのマニュアル中には、『_^HA』として『A』にアンダーラインを引いたり、『c^Hc』として『c』の強調文字を表示したりしていますが、それらの装飾も取り払ってください（タイプライター時代の名残です。lessではこの装飾表示が太字になったり、色付きになったりというフェイク機能がサポートされています）。
・ステップ3：単語数のカウントはUNIXの定番を使います。まず単語を辞書順に並べ替え、重複をカウントしつつ削除し（uniq -c）、数字の大きい順に並べ換えて（man sortを自分で調べてください）多い順に並べ替えます。
・ステップ4：あとは上位100を残す整形ですね。

（問題4）上の問題では、大文字小文字を違う単語としてカウントしてしまいます。どうしたら同一視できるでしょうか。
（ヒント）コマンドtrで大文字を小文字に変換（『tr A-Z a-z』）します。はさむにはどこがよいでしょうか。

```
#name   tel         attend
taroh   045-123-2222o
hanako  03-3123-4567x
ichiro  06-123-4567 o
hachirou011-123-4567o
```

```
#name   attend
taroh   o
hanako  x
ichiro  o
hachirouo
```

＜詰めvi・問題2図版＞

詰めvi 答え
（問題1）最後にundoするのがミソ
[:][!]mail hoge@moge.com[Enter][u]

（問題2）まあこれは『cut』コマンドを知っているかどうか、みたいなものですが。ちなみに、シェルプログラミングをしたりするときのためにpasteコマンド（ファイルを『横に』つなげる）も知っておくと便利です。
[:]!cut -c 1-8,21-

（問題3）ステップ2は置換ですが、難しい正規表現（『バックスペースの前後に同じ文字がある』というパターン）を使わなくても、単なる決まった文字列の削除でOKです。
ステップ3は、viの使い方というより、いかにUNIX系コマンドをよく知っているかにかかっています。これはシェルスクリプトを組む場合でも同じです。よく初心者ユーザーの方から「シェルスクリプトというのが作れれば仕事が楽になるのは知っている。だがどうすればいいの?」という声をききますが、何のことはない、ふだんコマンドラインから入力しているコマンドをファイルに取っておいて、それをキーボードから入力する代わりにファイルからシェルに流し込んだものが『シェルスクリプト』なのです。気張らず構えず、かといって調べもの（manを活用しましょう）を怠らず、徐々に覚えていきましょう。
とはいえ、『sort→uniq→sort』の連結は、多いものをカウントするときの黄金の定石手順です。

・ステップ1:
[:]!man csh[Enter]

・ステップ2:
[:]%s/.[^V][^H]//g[Enter][:]%s/[[TAB]]/[^V][^M]/g[Enter]

・ステップ3:
[1][G][!][G]sort¦uniq -c¦sort -nr[Enter]

・ステップ4:
[d][d][:]101,$d[Enter]

（問題4）
・ステップ3を次のように変更:
[1][G][!][G]tr A-Z a-z¦sort¦uniq -c¦sort -nr[Enter]

コマンド一覧 (今まで取り上げたものすべて)

重要度
★★★ = 覚えるまではviを起動しないほうがよい　☆☆☆ = 知らなきゃ死ぬでしょ　☆☆= 知っていると編集が速くなる　☆= 知っていると自慢できる
(なし) = 知っていてもあまり自慢できない
※重要度は筆者（ちなみにvi歴15年）の主観です。人によっては操作の宗派、いや流儀が異なる場合もあります（たとえば[i][A]の組を頻用せずに[a][I]を使う人もいるかも）

●viコマンド

・絶対必須系

コマンド	重要度	動作
:	★★★	(exコマンドへ)
u	★★★	アンドゥ
u..…	☆☆☆	アンドゥ繰り返し

コマンド	重要度	動作
Q	☆	exモードへ
U	☆	行内すべてアンドゥ
uu]	☆	リドゥ

・カーソル移動系

※ 組み合わせ可能なviコマンド
以下の表で『◎<動作>』となっているものは、繰り返し回数(など)が指定できることを示します
回数を指定しても意味がないものは（原理的に可能でも）書いていません

コマンド	重要度	動作
◎h	☆☆☆	カーソル←
◎j	☆☆☆	カーソル↓
◎k	☆☆☆	カーソル↑
◎+		カーソル下の行の行頭文字へ
◎l	☆☆☆	カーソル→
0 (ゼロ)	☆☆	カーソル行頭へ
_ (アンダーバー)	☆☆	カーソル行頭の文字へ
^U	☆☆	半画面上へ
◎^B	☆☆	1画面上へ
◎w	☆☆☆	次のword先頭まで移動
◎W	☆☆	big word先頭まで移動
◎e	☆	次のword末尾まで移動
◎{	☆	次の段落(1行空き)まで移動
%	☆☆	対応するカッコに移動
◎f<文字>	☆☆	行内次の<文字>に移動
◎t<文字>	☆☆	行内次の<文字>の手前に移動
/文字列<Enter>	☆☆☆	後方検索
n	☆☆☆	前回と同じ方向に再検索
/<Enter>	☆☆	後方に再検索
◎G	☆☆	指定行へ
^G	☆☆	行番号その他の情報表示
'<文字>		ラベル位置へ移動
''	☆☆	直前の移動先へ移動

コマンド	重要度	動作
H	☆☆	表示画面トップへ
◎J	☆☆☆	2行(数行)を接続
M		表示画面中央へ
◎—		カーソル上の行の行頭文字へ
L	☆☆	表示画面末尾へ
$	☆☆	カーソル行末へ
^D	☆☆	半画面下へ
◎^F	☆☆	1画面下へ
◎b	☆☆☆	前のword先頭まで移動
◎Bl	☆☆	big word先頭まで移動
◎E	☆	次のbig word末尾まで移動
◎}	☆	前の段落まで移動
◎F<文字>	☆☆	行内前の<文字>に移動
◎T<文字>	☆☆	行内前の<文字>の手前に移動
?文字列<Enter>	☆☆☆	前方検索
N	☆☆☆	逆方向に再検索
?<Enter>	☆	前方に再検索
G	☆	最終行へ
`<文字>		ラベル位置へ移動
``	☆☆	直前の移動先へ移動

・画面再描画系

コマンド	重要度	動作
^L	??	画面再描画
z.	☆☆	カーソル行を中央として再描画
z+		1画面進む、再描画

コマンド	重要度	動作
^R	??	画面再描画
z—	☆	カーソル行を最下行として再描画
z^		1画面バック、再描画

・文字挿入系

コマンド	重要度	動作
◎i	☆☆☆	カーソル位置の前に挿入
◎a	☆☆	カーソル位置の後に挿入
◎c	☆	指定範囲の変更
◎r	☆	1文字を変更

コマンド	重要度	動作
I	☆☆	行頭に挿入
A	☆☆☆	行末に挿入
C	☆	行末まで変更
R	☆☆	[ESC]入力まで変更

・カット & ペースト系

コマンド	重要度	動作
◎yy	☆☆☆	1行コピ ー
◎x	☆☆☆	カーソル位置の文字削除
◎dd	☆☆☆	1行削除
◎p	☆☆☆	カーソルの後ろにペース

コマンド	重要度	動作
◎Y	☆	1行コピー
◎X	☆☆	カーソル前の文字削除
D	☆☆	行末まで削除
◎P	☆☆☆	カーソルの前にペースト

・マルチウィンドウ系（vimのみ）

コマンド	重要度	動作
[^W][s]	☆☆☆	ウィンドウを2つに分ける(split)
[^W][n]	☆☆	新しいウィンドウを開く(new)
[^W][c]	☆☆☆	新しいウィンドウを閉じる(close)
[^W][q]	☆☆	ウィンドウを閉じて編集を終える(quit)
[^W][o]	☆☆	現在編集中のウィンドウ１個にする(only)
[^W][^W]	☆☆☆	次のウィンドウに移る(wrap)
[^W][+]	☆	現在のウィンドウサイズを大きくする
[^W][—]	☆	現在のウィンドウサイズを小さくする

・その他

コマンド	重要度	動作
ZZ	☆	セーブして終了
◎<	☆	タブを1段浅く
◎>	☆	タブを1段深く
!	☆	外部コマンドの実行
◎~	☆☆	大文字・小文字の入れ替え
<ESC>	★★★	コマンドモードに戻る
^W	☆☆	行頭まで、あるいは新たに入力した部分削除（シェル設定に依存）

● コマンドモード、exモード

※ exコマンドモードに入るときの[:]および最後の<Enter>（または<Esc>）は省略

※ (…)内は直前のキーワードが省略できることを示す

[…]内は付加する・しないで動作が変わることを示す（オン・オフなど）

・絶対必須系

コマンド	重要度	動作
^W ※1	☆☆	行頭まで削除(コマンド入力の中止)

※1 シェルの設定が引き継がれるので異なる場合がある。

・tag操作のコマンド（vimバージョンのvi）

コマンド	重要度	動作
pop	☆☆	指定行にジャンプする
tags	☆☆	最終行にジャンプする
di(splay)		最後から2行目にジャンプする

(nex/nviバージョンのvi)

コマンド	重要度	動作
tagp(op)	☆☆	tagスタックを1個戻す(元の場所に飛ぶ)
di(splay) t(ag)	☆☆	tagスタックを見る
di(splay) b(uffer)	☆	カットバッファの内容を見る

・ファイル操作系

コマンド	重要度	動作
wq	★★★	セーブして終了（名前がすでについている場合）
w	☆☆☆	セーブ（名前がすでににについている場合）
w!	☆☆	強制セーブ
w ファイル名	☆☆☆	名前を付けてセーブ（名前がまだない場合）
		一時的に他の名前でセーブ（名前がある場合）
q	☆☆☆	終了（変更後セーブされていれば）
q!	★★★	強制終了
vi	☆	（viモードへ）

・カーソル移動系

コマンド	重要度	動作
数字	☆☆☆	指定行にジャンプする
$	☆☆	最終行にジャンプする
$-2		最後から2行目にジャンプする
ne(xt)	☆☆	次のファイルを編集
ne(xt) ファイル名	☆	『ファイル名』を編集
ne(xt)#	☆	直前のファイルを編集
ne(xt)! (同上)		現在の編集を放棄して(同上)

※ n(ext) (nのみでne)と同じバージョンのviもあります

・編集コマンド

コマンド	重要度	動作
範囲！外部コマンド	☆☆	外部コマンドの実行結果と置き換え
範囲s/置換前/置換後/	☆☆☆	正規表現を使った一括置換
範囲d	☆☆	削除
範囲m行番号	☆☆	移動
範囲gグローバルコマンド	☆	複数行にまたがる操作

・set・mapコマンドによるviのカスタマイズ

コマンド	重要度	動作
set all	☆☆	setパラメータの表示
set [no]number (nu)	☆	行番号の[非]表示
set [no]ruler (ru)		目盛りの[非]表示
set [no]autowrite (aw)	☆	自動セーブオン／オフ
set [no]autoindent (ai)	☆	自動インデントオン／オフ
set [no]showmatch (sm)	☆	カッコの対応を示す
set [no]wrapscan (ws)		自動ラップのオン／オフ
set [no]readonly (ro)		書き込み保護のオン／オフ
set tabstop (ts) ＝数値	☆	タブ位置設定
set wrapmargin (wm)＝数値		自動ラップ位置の設定
mk(exrc)[!] ファイル名	☆	exrcファイル型式で設定を保存
map 操作1 操作2 ……	☆	『操作1』を『操作2』に置き換え
unm(ap) 操作		『操作』のmapを解除する

Linux日記

第5回　プロセスの実行

第4回では、プロセスの生成から終了までの過程について説明しました。今回は、システム負荷をみる指標となるロードアベレージと、実際のプロセス実行のお話です。

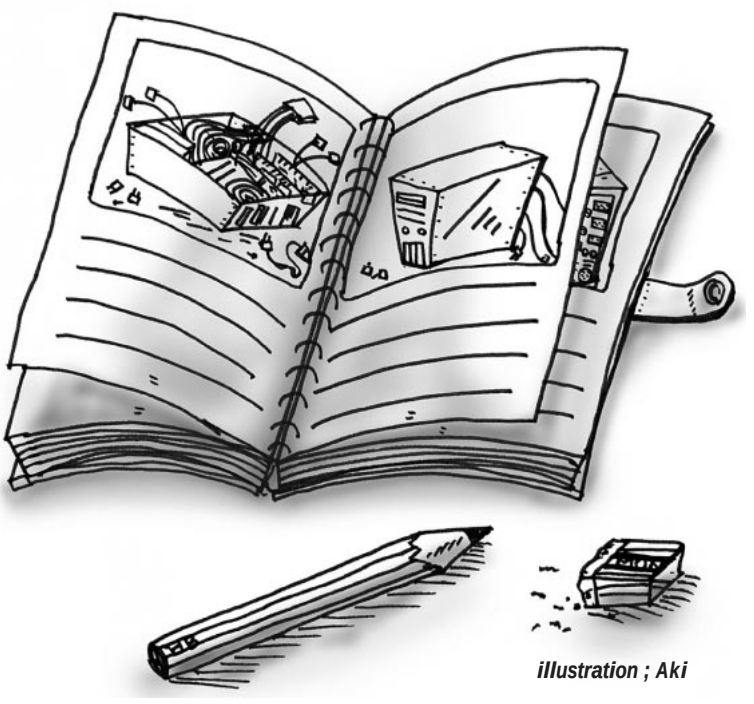


illustration ; Aki

文：榊 正憲
Text : Masanori Sakaki

前回は、プロセスの誕生と終了、プロセスの実行の話だったが、例によって中ぶらりんで終わってしまった。今回は実行中プロセスの話の続きである。

■ロードアベレージ

前回、スケジューラは実行可能プロセスにCPUタイムを割り当て、実際の処理を行うという話をした。実行中のプロセスの数は、システムのCPUの数を超えることはないが、プロセスが実行可能であるかどうかということは、CPUの数には関係ない。処理が進んで、I/Oなどが完了すれば、プロセスは実行可能状態になる。

実行待ちのプロセスが多いということは、簡単に言えば、CPUがシステムに求められている要求を処理しきれていないということである。

システムの負荷の重さを測るうえで、もっとも一般的に使われているのがuptimeコマンドである。これはその名の示す通り、もともとはシステムが連続して稼働している時間を表示するためのコマンドである。つまり、アップしている時間を示すからアップタイムなのだ（これに対して、システムが動いていないことをダウン状態といい、落ちていた時間をダウンタイムという）。だがuptimeが便利なのは、稼働時間と共に表示されるロードアベレージという数値が、システムの負荷の度合を教えてくれるからである。

uptimeの出力を見ると、最後に3つの数字が表示されている。これが過去1分、5分、15分の間のロードアベレージである。この数字が大きいほど、システムの負荷が大きいということになる。3種類の時間スパンでの平均値を見ることで、システムが慢性的に重いのか、一時的に重いのかといったこともわかる（**リスト1**）。

さて、そもそもロードアベレージとは何を表している数字なのだろうか？簡単に言ってしまえば、実行待ちキューの中にあるプロセスの数の時間平均である。つまり、過去1分、5分、15分の間、実行待ちキュー中にあった平均プロセス数、つまり実行可能プロセス数の平均である。正確に言えば、この数には、実行中のプロセスも含まれている。スケジューラによるプロセスの管理という点で見ると、実行可能で実行を待っているプロセスと、実行中のプロセスにさほどの差があるわけではない。マルチプロセッサシステムでない限り、スケジューラが動作している間は、実行中のプロセスが存在しないからだ（実行中なのはシステムのスケジューラである）。そして、プロセスが実行を始めてしまえば、それを観測できる実体というのは存在しなくなるのだ（**図1**）。

▼リスト1　uptimeコマンドの出力

```
3:08pm  up 8 min,  1 user,  load average: 0.05, 0.07, 0.06
```

暇にしているシステムであれば、ロードアベレージの数値は0に近い。実行待ちのプロセスはほとんどないということである。つまり、実行可能になったプロセスにはすぐにCPUが割り当てられ、そのプロセスもすぐに実行を終了するか待ち状態になってしまうという状態だ。1つのプロセスだけが忙しく動いている場合は、ロードアベレージはおおよそ1.0になる。忙しく動作するプロセスが2つあり、常に1つのプロセスが実行待ちになっている状態では、ロードアベレージは約2になる。また、1.0未満の場合、CPUがアイドルしている時間があるということになる。

要するに、実行待ちのプロセス数プラス実行中のプロセス数がロードアベレージとなる。シングルプロセッサシステムであれば、実行中のプロセスの数は0～1だ。ロードアベレージが1近辺であれば、リクエストに対して、CPUは相応の能力で処理していることになる（その処理の実行が速いか遅いかはここでは問題ではない）。それより大きい数値になるなら、CPUの処理能力に対してリクエストされているプロセスの数が多い（待っているプロセスが多い）ということになる。この状態は、一般に負荷が重いということになるだろう。もちろん、CPU負荷は時々刻々変化していくので、ロードアベレージが1より大きいからマシンのパワーアップが必要というわけではない。単に、暇な時より、ユーザーに応答を返すまでの時間が長くなる、あるいは、一定時間に処理できるデータ量が少なくなるというだけのことだ。

ところで、CPUが複数あるマルチプロセッサシステムの場合はどうなるのだろうか？　残念ながら、今のところ手元にマルチプロセッサシステム上で動作しているLinuxがないので、詳しいことはわからない。そのうちマルチプロセッサシステムを動かした時にでも説明できるだろう。たぶんその時には、SMP（対称マルチプロセッシング）とか実行スレッドの話もすることになると思う。

■割り込みとシグナル

プロセスの話からちょっと脱線するが、プロセスの状態遷移を理解するうえで、UNIXのシグナルをある程度知っておく必要がある。シグナルはある種の割り込み処理なので、ハードウェア割り込みについても簡単に説明しておこう。

ハードウェア割り込みというのは、ハードウェアの状態の変化（キーが押されたとか、ディスクI/Oやネットワークの I/Oが完了した、あるいはエラーの発生など）により、実行中のコードが強制的に中断され、別の処理（キー入力ルーチンとかディスクドライバなど）に制御が移ることである。この処理が完了すると、それまで実行されていたコードが再開し、何事もなかったかのように処理を継続する（エラーの場合は再開できないこともある）。つまり、外部イベントにより引き起こされるサブルーチン呼び出しのようなものである（実際にはCPUの動作モードの遷移なども絡むため、もっと複雑だが）（**図2**）。

割り込みを使うことによって、I/Oが関係しないプログラムを、I/O待ちのプログラムと並行して実行させることができる。あるプログラムが入出力などを待っている間に、別のプログラムが処理を続けることができるのだ。これは、複数のプログラムの同時実行をサポートするオペレーティングシステムでは重要な点である。

このようなハードウェアによる割り込みは、一般にシステムのデバイスドライバで処理される。ユーザープロセスは、単にオペレーティングシステムに入出力のリクエストを送れば、I/Oが完了した時点で、つまりシステムとデバイスドライバがいろいろな割り込みやI/Oを処理した後で、処理が完了したことが通知される。一般のプロセスは、この種のハードウェア割り込み

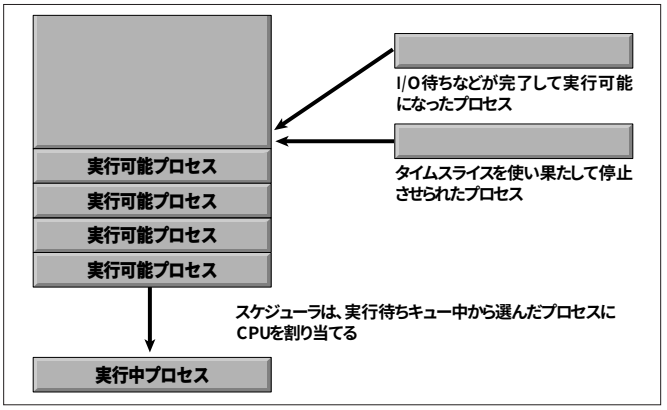


図1　実行待ちキュー

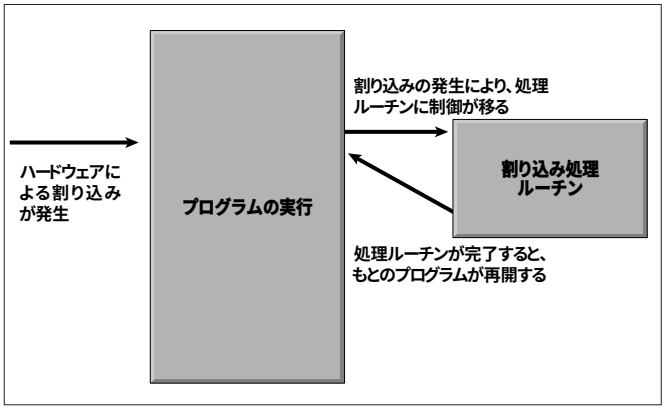


図2　割り込み

を直接検知することはない（**図3**）。応答するとしても、割り込みの結果として入出力が完了し、待ち状態から実行可能状態に遷移するとか、次に説明するシグナルという形になる。

UNIXでは、このようなハードウェア割り込みとは別に、シグナルというある種の割り込みメカニズムも提供されている。これはユーザープロセスからも可視であり、影響を受けるイベントである。たとえばコマンドの実行中にCtrl＋Cをタイプしてコマンドを強制終了させるといった処理は、シグナルメカニズムを使っている。

シグナルは、単純なプロセス間通信の1種である。プロセスは、別のプロセスにシグナルを送ることができる。また、プロセスの状態に基づいて、ある種のシグナルがそのプロセス自身に送られることもある。これはおもにエラーの発生（算術処理のエラーやメモリアクセス違反など）を通知するものである。エラー発生はハードウェア割り込みでトラップされるが、それがオペレーティングシステムによって、プロセスから可視なシグナルに変換されるのである。

プロセスは、シグナルを受信した時に特定の処理を実行する。この処理の内容は、プログラムコードのレベルで設定される。また、何も設定していないときのデフォルト処理も規定されている。もっとも単純な処理は、無視する、あるいはプログラムを終了するというものだ。Ctrl＋Cのタイプで終了するというのはこの例である。プログラムの終了に際して、クリーンアップ処理を行いたい場合（一時ファイルの削除や画面モードの復旧など）は、終了シグナルに対する処理ルーチン（これをシグナルハンドラという）を定義し、定義したルーチン内でその処理を

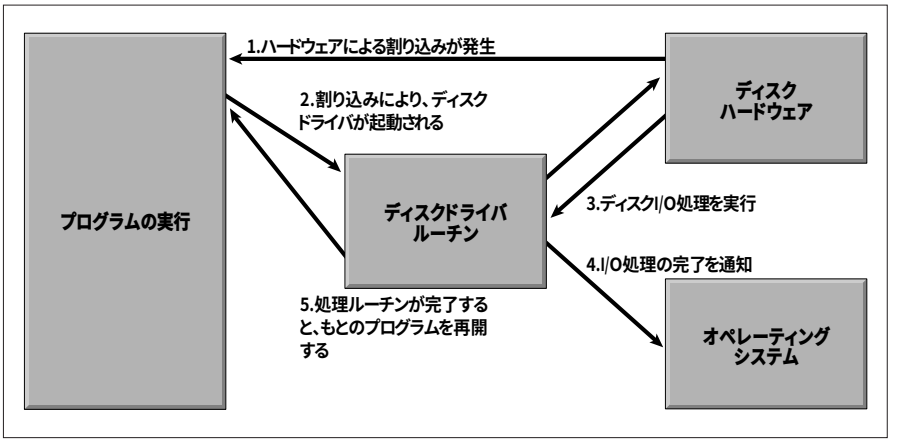


図3　デバイスドライバの動作

Column

ダウンタイムを表示できるか？

uptimeコマンドがあるなら、downtimeコマンドもあるかというと、もちろんない。ダウンしているマシン上でコマンドを走らせろといっても無理である（ログを調べて、アップ後に表示することは可能だが）。しかし、あるシステムがダウンしている時間を調べる方法がないわけではない。ネットワーク環境であれば、ほかのマシンから観察することができる。rwhodというデーモンを動かすと、そのマシンにログインしているユーザーの一覧と、そのマシンのアップタイム、ロードアベレージなどの情報がほかのマシンにブロードキャストされる。この情報はネットワーク上の各マシンでスプールされ（この処理を行うのもrwhodである）、rwho、ruptimeコマンド（それぞれwho、uptimeのネットワークワイド版）でその内容を表示することができる。

このユーザー、アップタイム情報を収めたパケットが一定時間届かないと、そのマシンはダウンしているものとみなされ、ruptimeコマンドでdownと表示され、そしてどれだけの時間ダウンしているかも表示されるのである。

もちろん、本当にダウンしているのではなく、ネットワークの障害かもしれないし、rwhodが死んでいるだけかもしれない。それでも何らかの障害があることは確かだから、要調査である。多数のサーバを管理する場合などは便利なコマンドだ。

ホスト名変更やマシンの撤去などを行うと、その存在しないマシンはずっとdownとして表示される。これは気持ち悪いので、表示しないようにしたいだろう。このような場合は、/var/spool/rwhoの下にある（位置はシステムによって異なることもある）、そのホスト名を含んだファイルを削除すればよい。

rwhodはブロードキャストを使っているので、マシンが増えるとネットワークのトラフィックが増加する。これを嫌って使わない管理者も多い。まぁ、Windowsのネットワークに比べれば誤差みたいなものだが。

と、ここまで書いて、わが家のredhat 5.2日本語版にはrwhoはあるものの、ruptimeコマンドがないことがわかった。もっとも、ほかのruptimeが動くマシン上では、このマシンがアップかダウンかはきちんと表示される（**リスト2**）。

▼リスト2　ruptimeコマンドの出力（pc003というマシンはダウンしている）

```
pc001         up     5:09,     2 users,  load 0.01, 0.06, 0.00
pc002         up  1+06:32,     6 users,  load 0.03, 0.08, 0.06
pc003       down     0:22
```

行うようにすればよい（**図4**）。

どのような種類のシグナルがあるかという詳細は、man 7 signalで見ることができるが、この中にはユーザー定義のものもある。シグナルをうまく使うことで、簡単なプロセス間通信を行うことができる。たとえばttyがアタッチされていないデーモンプロセスなどを制御するために、シグナルを使うことができる。

プロセスの解説でシグナルを紹介したのは、プロセス終了に限らず、プロセスの状態を遷移させるシグナルがいくつかあるからだ。

■入出力待ちの停止

プロセスの実行が停止するもっとも明らかな（そしてもっとも頻度の高い）理由は、入出力待ちだろう。ユーザーがキー入力するのを待つ、ディスクやネットワークなどのI/Oを待つといったことだ。基本的に、プログラムが入力を求めるということは、その入力データを使いたいということである。それゆえ、入力が完了するまで、プログラムは処理を続けることができないのである。

入力待ちより頻度は少ないものの、出力待ちで停止することもある。入出力はバッファリングされるので、小容量の出力なら、たとえ出力先にすぐに書き出すことができなくても、バッファに書き込むことで、プログラム側は書き込みオペレーションを終えることができる。しかし、バッファがいっぱいになってしまうと、バッファが空くまで待たされることになる。

このように、何らかのイベント（この場合は入出力の完了）を待って停止することを、「ブロックされている」という。プロセスをブロックするのは、入出力だけではない。リソースへの排他的アクセスなどに使うロックも、ブロックの原因となる。データベースシステムなどでは、あるデータが同時に複数のプロセスから書き換えられてしまったり、修正途中の不完全なデータが読み出されるのを避けるために、ロックを使ってアクセスを制御している。他のプロセスがロックしているリソースに対してアクセスを試みると、そのプロセスはロックによってブロックされるのである。

psコマンドによって表示される数多くのプロセスの大半は、（よほど忙しいサーバでもない限り）入力待ちで停止している。ネットワーク関係のデーモンであれば、ネットワークの読み込み待ちや接続待ちで停止しており、シェルやエディタなどは、キー入力待ちで止まっているだろう。また、ディスクI/Oを頻繁に行うプロセス（データベースサーバやファイルサーバ用デーモンなど）であれば、ディスクI/O待ちで止まっていることも多いだろう。

psコマンドを使って表示した時、このような停止状態にあるプロセスは、STATフィールドにS、Dが表示される。SとDはどのように違うのか？　簡単に言ってしまえば、シグナルに応答できる停止状態がS、応答できない停止状態がDである。言い方を変えると、シグナル割り込み可能な状態がS、不可能な状態がDである。一般に、Dはディスク I/Oを行うシステムコールの実行中など、途中で停止して割り込み処理を行うのが困難な状態である。それに対してSは、ttyのI/O待ちなど、たとえシステムコール中であっても、シグナル割り込みに対処できる状態である。一般にD状態はディスクI/Oが多いプロセスに見られるもので、プロセスは実行中ではないものの、ユーザーから見れば忙しく動いているプログラムであり、アイドル中のシェルやエディタなどとはまったく異なるものである。また、D状態はずっと続くわけではなく（I/Oの完了によりSかRになる）、普通はR、S、Dを遷移することになる。

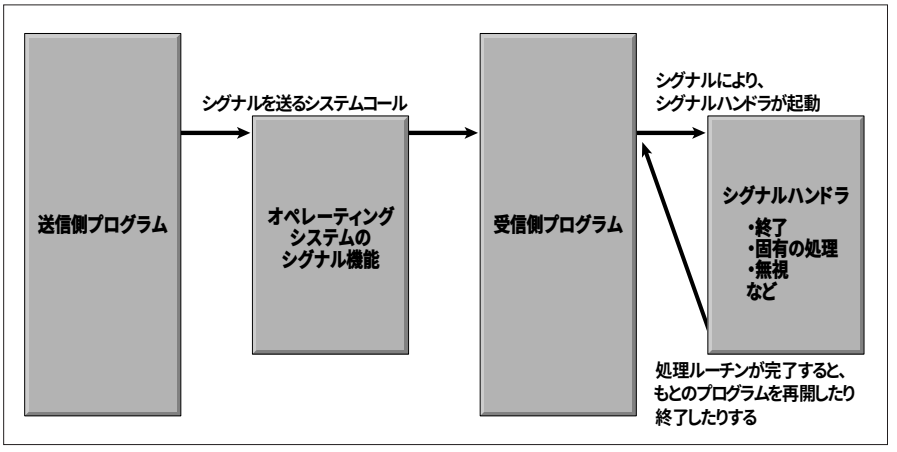


図4　シグナル

■スリープ

プログラムによっては、入出力やイベントを待つわけでもなく、意図的に停止するものがある。たとえば定期的に何らかの処理を行うデーモンプログラムなどは、このような形で停止することになる。このような停止状態をスリープという。

1時間に1回スプールディレクトリを調べて、そこに置かれているファイルを処理するといったプログラムであれ

ば、1時間スリープし、ファイル処理を行うというサイクルを繰り返すことになるだろう。たとえばバッチ処理でメール配信を行うメールデーモンなどは、このような形態で動作する。日常的なところでは、指定時刻にバッチ処理を行うcrondデーモンもスリープと処理を繰り返している。

オペレーティングシステムによっては、スリープするためのシステムコールを用意しているものもあるが、UNIXの場合はライブラリ関数によって提供されている。タイマーにより自身に通知するという仕組みである。

スリープによる停止中のプロセスは、シグナル割り込みを受理できる。つまりスリープ中であっても、シグナルに応答してハンドラが起動されるということだ。これにより、スリープ中のデーモンプロセスでも、シグナルを使った操作や、シグナルによるイベントの通知に即座に応答することができる。この停止状態は割り込み可能なので、スリープ状態のプロセスは、psのSTATフィールドがSである。

厳密にいえば、スリープ状態は、スリープ時間の完了、あるいはシグナルというイベントを待って、割り込み可能状態でブロックされている状態のことである。従って、オペレーティングシステム側から見る限り、キー入力待ちなどでブロックされているプロセスと同じ状態である。実際、割り込み可能状態のI/O待ちはスリープ状態と呼ばれる。しかし、プログラムを使う側、あるいは作る側から見ると、入力待ちのブロックと意図的なスリープは明らかに異なる状態である。ここではI/O待ちと意図的なスリープをあえて分けて説明したが、オペレーティングシステム側から見ると同じであるということを覚えてくことが大事だ。

■ジョブコントロールによる停止

I/Oやロックを待つブロックでもなく、スリープでもない停止もある。それは、ジョブコントロール機能による停止である。エディタなどのプログラムの実行中にCtrl＋Zをタイプすると、そのプロセスは動作を停止し、シェルに戻る。このプロセスは終了したわけではなく、後でfgなどのシェルコマンドを使って再開することができる。ジョブコントロールによるサスペンド状態は、I/O待ちでもなく、実行可能でもないという点で、ちょっと特殊な状態である。しかしこれは、再開を通知するシグナルを待ってブロックされている状態である。

ジョブコントロールによる停止状態のプロセスは、psのSTATフィールドでTと表示される。また、プログラマーでない限りほとんど関係ないが、デバッガから実行されたプログラムもT状態になる。シングルステップ実行やブレークポイントによる停止状態はTである。

ジョブコントロールは、シェルやエディタなどに固有の機能ではなく、オペレーティングシステムがシグナルを使ってサポートしている、つまり、すべてのプログラムに共通の機能である。

Column

killコマンド

シェルのkillコマンドは、一般にプロセスの強制終了を行うためのコマンドだが、オプションとしてシグナルの名前か番号を指定して、任意のシグナルを送ることができる。

たとえばデーモンプロセスの操作（正しい手順による終了、設定ファイルの再読み込み、診断情報のダンプなど）を行うために、管理者はkillコマンドでデーモンに各種シグナルを送ることになる。これはもちろんデーモンを殺す（止める）ためのものではない。デーモンは各種シグナル用のハンドラを用意しており、シグナルを受信するとそのハンドラが起動され、必要な処理を行うのである。どのシグナルがどのような処理を行うかは、デーモンのmanに記述されている。

一般ユーザーでも、とりあえず覚えておくと便利なのは、SIGHUP（1）とSIGKILL（9）である。Xやネットワーク環境で複数のシェルセッションを開いていると、正常終了できずにシェルのプロセスが残ってしまうことがある。このシェルは、ただのkillコマンド（SIGTERMを送る）では殺せないことが多いのだ。このような時は、SIGHUPを送るとたいてい殺せる。SIGHUPは、モデムなどを使って接続している時の回線切断で送られるシグナルである。シェルは、回線切断で終了するようになっているので、SIGHUPはSIGTERMより強力である。

シェルから起動されているプロセスが残っている場合は、シェルを殺すことを試みる前に、そのコマンドのkillを先に行うこと。普通はこれでシェルもいなくなるはずだ。それでもシェルが死なない時に、SIGHUPを試みるようにしよう。

どうしても死なないプロセスは、SIGKILLで殺せる。これはプロセス側でトラップできないシグナルなので、一部のシステムプロセスを除いてどんなプロセスでも殺せる（もちろん、ユーザーの権限内でだが）。ただし、SIGKILLを使った場合は、クリーンアップ処理が行われないので、画面モードがおかしいままになったり、作業用ファイルが残ったままになるので注意すること。

D状態（割り込み不可）状態の停止プロセスは、シグナルでは殺せない。しかし、D状態からSやRに移行しないというのは、普通はシステムのバグやハードウェア障害が原因なので、ユーザーレベルではどうにもならないだろう。

しかしその性質上、ユーザーと対話する形式のプログラムでよく使われているのである。ttyは適当なキー入力（Ctrl＋Zなど）によってプロセスに停止シグナルを送り、シェルはfgやbgなどのコマンドにより、ジョブコントロール用のシグナルをプロセスに送る。シグナルを受信したプロセスは、それに応じて停止／再開する。プログラムによっては、ジョブコントロール用のシグナルハンドラを用意し、停止、再開の前にクリーンアップ処理や準備作業を行うものもある。また、停止したくない場合は、シグナルを無視することもできる。

■プロセスの状態遷移

ここまで解説してきたことをまとめておこう。forkのリターンから始まったプロセスが終了するまでの間に取り得る状態を**図5**に示しておく。

この図は厳密なものではないということを断っておこう。状態の遷移の仕方によっては（あるいはシステムの内部では）、この図の通りに遷移するのではなく、過渡的に別の状態を経由したり、逆に図に示されている途中の状態をスキップすることもある。しかし、プロセスはおおむね実行可能→実行中→停止→実行可能という状態遷移を繰り返しながら処理を進めていくということだけは覚えておいて損はない。

■筆者近況

連載の最初にちょろっと紹介したT氏（この連載をやれといったえらい人）が、またもや筆者のジョブキューに新しい仕事をぶちこんでくれた（毎度ありがとうございます）。前の大きい仕事がひと段落したところだったので、やっと筆者のロードアベレージも2を切るかな（これでもいろいろな仕事をしているのだ）と思ったところなのだが、それは許してもらえそうにない。今のところ新しいジョブはリソース待ちでブロックされているので、ロードアベレージにはまだ影響は出ていないが、筆者にも裏で回しているプライベートなジョブがいくつかあるし（これらはかなりniceされていたのである）、デーモンもあるから、新しい仕事が実行可能になると、ロードアベレージは3に近くなってしまうかもしれない。

T氏は、ほかにもいろいろやらせたい仕事があるなどとブツブツいっていたが、もしこれらが一斉に投入されたら、ロードアベレージはすぐにも4とか5になってしまうだろう。しかしT氏は筆者をこき使うことについては良心的なユーザーで、まとめてジョブを突っ込んで、終わった奴から片付けていくという方法は取らないようである。これをやると信頼性の低い筆者が落ちることを知っているのだろう。しかも筆者のスケジューラは、ロードアベレージが上がるほど、プライオリティの低いジョブを動かしたがる傾向があるということもばれているに違いない。

これはちょっと極端だが、このような話を無意識にしてしまうようになったら、そろそろ気を付けたほうがいい。重要なことは、こういう言い回しは、相手を選んでしなければならないということだ。相手を誤ると、単に話が通じないだけではなく、怪しい奴と思われ、その後の人生が狂ってしまう可能性がある。もちろん筆者はこんな言葉使いはしない（ようにしている）。

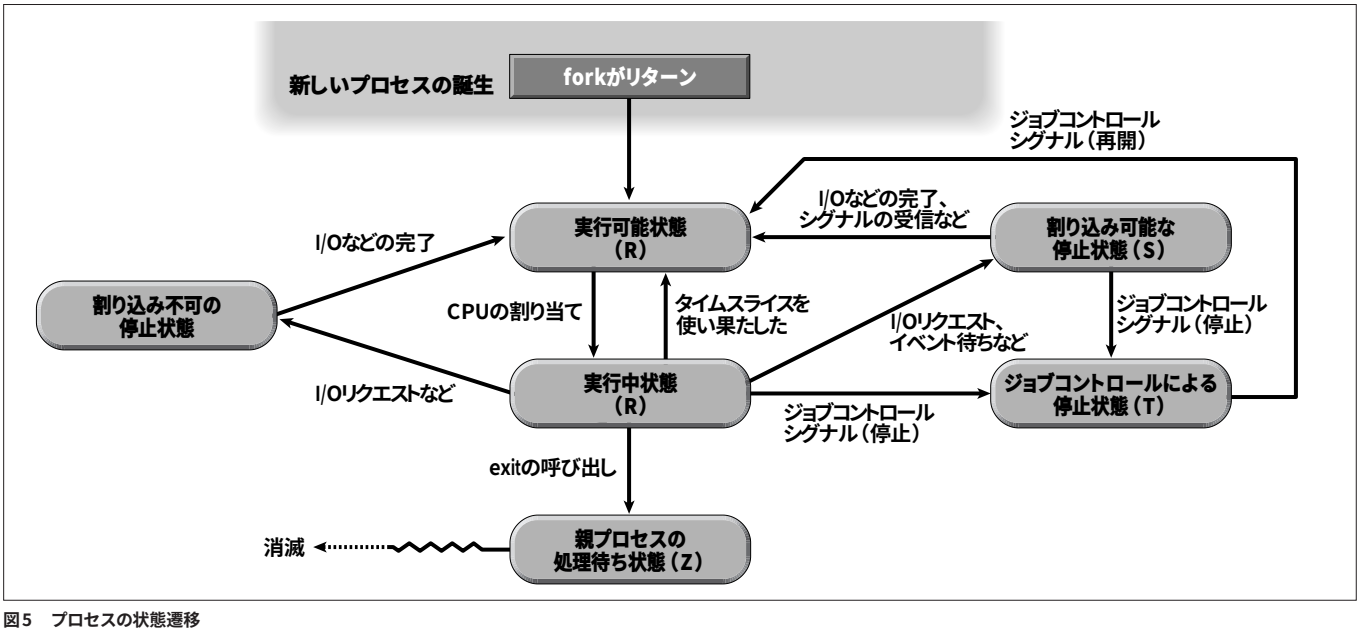
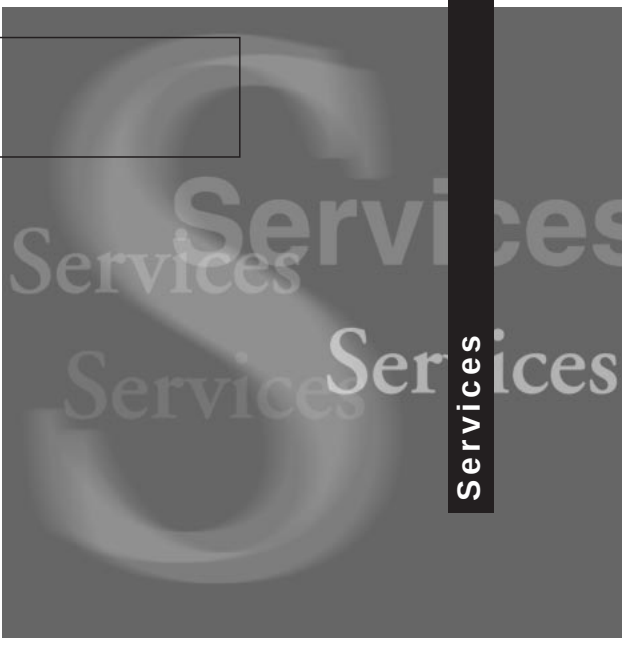


図5　プロセスの状態遷移

Webサーバ構築術（第6回）

Apacheに限らず、サーバソフトにはひとつ厄介な問題がある。それは、電源を入れ続ける限り増えていくログファイルの処理だ。今回はApacheのログを管理する方法と、Analogというアクセスログ解析ツールの使い方を紹介しよう。



Apacheのアクセスログを管理する

文：中島昌彦
Text : Masahiko Nakajima

前回、Apache 1.3.9をソースからインストールしたため、ディレクトリの位置がRed Hatのデフォルトと異なっている。httpd.confなどの設定ファイルは、今回より/usr/local/apache/confディレクトリにあるが、Red Hatのデフォルトでは /etc/httpd/confであるので注意していただきたい。

/usr/local/apache/logsにログファイルがある

ApacheのWebアクセスを記録するログファイルは、デフォルト設定では/usr/local/apache/logsに作られる。だが、ファイルシステムの都合や管理が繁雑になることから、/var/log/以下に作るように設定することも多い。たとえば、Red HatのRPMをインストールした場合には、/var/log/httpd内にApacheのログファイルが作られる。

デフォルト状態で作られるファイルは、access_log、error_logの2つだ。このファイルさえ管理していれば、ファイルシステムをフルにしてしまった巨大なログファイルにあわてることがなくなる。

access_log、error_logは、それぞれファイル名通りのログを記録する。access_logは、Webサーバにアクセスしてきたすべての記録を、そしてerror_logは、正常なアクセス以外のものを記録する仕組みだ。access_logの場合、すべてのアクセスを記録するという点に注意してほしい。すべてのアクセスということは、access_logには正常なアクセス以外の記録も含まれている。正常アクセス以外のアクセスは、error_logとaccess_logの2つにダブって書き込まれる。このため、純粋に、

```
# wc -l /usr/local/apache/logs/access_log
```

としても、純粋なヒット数は得られない。

一方のerror_logは、正常アクセス以外のすべての記録が残されている。Webサイト内のドキュメントやイメージのリンク外れや、CGIのエラーは、error_logを見ながら修正していけば、数は減っていく。error_logは、access_logからエラーだけを抜粋したファイルという位置付けだ。

Red Hat版RPMは、ログのローテートがかかる

ApacheをRPMでインストールしたときには、ログファイルの管理にはそれほど神経質になることもない。Red HatのApacheでは、ログファイルが/var/log/httpd/にあるが、ここは1週間単位でローテートされる。1週間単位でファイル名に.1、.2、.3、…と番号が付け加えられたバックログが保存され、1カ月以上前のものは削除される仕組みになっている（**リスト1**）のだ。これを実現しているのが、/usr/sbin/logrotateだ。

logrotateの仕組みを解説すると、/etc/cron.dailyにあるlogrotateが毎日起動される。/etc/cron.daily/logrotateは、/etc/logrotate.confの内容に従い、/usr/sbin/logrotateを動かす。logrotate.confでは、週単位で動き、バックログは4週ぶん保存するという設定

になっている。管理するログファイルの具体的な処理は、/etc/logrotate.dディレクトリ内のファイルに記述する。Apacheであれば、/etc/logrotate.d/apacheが該当するファイルだ。

この仕組みにより、Red Hatのデフォルトでは、Apacheのログが巨大化することを防いでいる。ちなみに、/etc/crontabを見ると、**リスト2**のように毎日午前4時2分に/etc/cron.dailyを動かす設定になっている。

ここまで仕組みができ上がっているので、ログの切り出しは週単位で、時間は4時2分というスケジュールで何も問題がなければ、この仕組みをそのまま利用したほうが手軽だ。Apacheをソースからインストールしたときなど、ログファイルは/usr/local/apache/logsにある。その場合には、/etc/logrotate.d/apacheファイルの“/var/log/httpd”という文字列を、“/usr/local/apache/logs”に置き替えればよい。

logは/var/logの下にあってほしい

ファイルシステムとして、ちゃんと/varや/var/logを切り分けている管理者にとって、Apacheが/usr/local/apacheの下に山ほどファイルを作ってくれることに不満を感じる人もいるだろう。ログエリアとして十分な容量のファイルシステムを/var/logにマウントしているようなときには、確かにApache 1.3以降のこの仕組みは不便だ。これをもっとも手軽に解決するには、

```
# rm /usr/local/apache/logs
# ln -s /var/log/httpd /usr/local
/apache/logs
```

とするのがベストだ。シンボリックリンクによって、/usr/local/apache/logsへのアクセスが、/var/log/httpdへのアクセスに置き替わる。この方法を使えば、Apacheのコンフィグレーションファイルを書き換える必要もない。さらには、/etc/logrotate.d/apacheも書き換える必要がなくなる。シンボリックリンクを張ったあと、/usr/local/apache/logsディレクトリ内のログファイルへの書き込みが、/var/log/httpd以下に書き込まれることを確認したうえで、Apacheをリスタートする。

```
# /etc/rc.d/init.d/httpd restart
```

シンボリックリンクを張りまくることはしたくないという管理方針ならば、httpd.conf中のCustomLogとErrorLogの行を書き直し、/var/log/httpdの下にログファイルを置く設定にするといい。修正する部分は、**リスト3**のようにCustomLogとErrorLogという行を見つけ出し、フルパスを変更する。ErrorLogとCustomLogは並んではいないので、それぞれ探さないといけない。また、CustomLogの行には、最後にcommonという文字が入っている。これはそのまま残しておく。

ログを取る項目を増やす

access_logとerror_logの具体的な中身が**リスト4、5**だ。一部データが入っていないのでわかりにくいが、並び順はaccess_logの先頭に書いてあるとおりである。

データの中身が含まれていないリモートユーザー名と認証ユーザー名は、セキュリティに密接に関係したデータで、ident（RFC1413）認証に対応したクライアントでなければ、リモートユーザー名は記載されない。さらに、デフォルト状態でApacheはIdentityをチェックしないようになっている。Identityをチェックさせるには、/usr/local/apache/conf/httpd.confに、

▼リスト1　/var/log/httpdディレクトリ。Apacheのログが1週間ごとにローテートされている

```
   0 -rw-r--r--   1 root     root            0 Nov 28 04:02 access_log
   0 -rw-r--r--   1 root     root            0 Nov 21 04:02 access_log.1
   0 -rw-r--r--   1 root     root            0 Nov 14 04:02 access_log.2
   0 -rw-r--r--   1 root     root            0 Nov 12 10:09 access_log.3
   0 -rw-r--r--   1 root     root            0 Nov 28 04:02 error_log
   0 -rw-r--r--   1 root     root            0 Nov 21 04:02 error_log.1
   0 -rw-r--r--   1 root     root            0 Nov 14 04:02 error_log.2
   1 -rw-r--r--   1 root     root          180 Nov 12 10:38 error_log.3
```

▼リスト2　/etc/crontabファイルの内容。毎日午前4時2分にcron.dailyが実行される

```
# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

▼リスト3　httpd.confの設定ファイル

```
#ErrorLog /usr/local/apache/logs/error_log
ErrorLog /var/log/httpd/error_log

#CustomLog /usr/local/apache/logs/access_log common
CustomLog /var/log/httpd/access_log common
```

```
IdentityCheck on
```

という1行を加える。そして、クライアント側でidentdを動かしておけば、**リスト4**のaccess_logの例にある2〜3行目のように、リモートユーザー名が記録される。クライアント側でidentdに相当するものが動いていなければ、unknownと記される仕組みだ。

これは一見すると便利な機能のように見えるが、IdentityCheckをオンにすると、サーバからクライアントマシンにidentチェックのためのアクセスがかかり、サーバとネットワークの負荷が若干ながらも増加するため、IdentityCheckを行うことはほとんどない。

認証ユーザー名はユーザー認証を通過してきて入ったときに有効になる。**リスト4**のaccess_logの4行目がその例だ。ユーザー認証を通過したユーザー名が記録される。

ところが、アクセスしてきたクライアントがApacheに渡す情報はこれだけではない。リンク元や、ブラウザ情報なども含まれる。つまり、アクセスログを集計し、Webサイトの今後に役立つ資料を作りたいならば、デフォルトのログ情報だけでは情報が少なすぎるのだ。そこで、ログファイルのフォーマットを変更し、必要な情報を収集することにする。

もちろんApacheはこうした対応も、/usr/local/apache/conf/httpd.confを変更していくことで可能だ。

access_logのフォーマットを整える

では、access_logの設定を変更しよう。よく加えられるものとして、閲覧者のWebブラウザ、リンク元がある。

/usr/local/apache/conf/httpd.confの中で、ログフォーマットに関する設定は、**リスト6**の部分である。Apacheでは、LogFormatとして、いくつかの形式を設定できる。**リスト6**では、1行目のログフォーマットをcombined、2行目をcommon、3行目をreferer、4行目のログファイルをagentとしてLogFormatで定義している。

こう定義したLogFormatに対して、CustomLogで、実際のログファイル名と、ログフォーマットを定義する。**リスト6**では、CustomLogを使って、/usr/local/apache/logs/access_logというファイルにログを記録する。そのときのフォーマットは、commonの形式を使えという設定だ。書式としては、

```
LogFormat "記録形式" フォーマット名
```

で記録する内容を定義して、

```
CustomLog ファイル名 フォーマット名
```

で書き込むファイル名とフォーマット名を設定する。

そこで、/usr/local/apache/logs/access_log内のCustomLog行の最後にあるcommonをcombinedに変更し、apacheを再起動すると、そのときから/usr/local/apache/logs/access_logにはcombinedの形式でログが記録されるようになる。その状態が**リスト7**だ。行末に、リンク元（この場合はリンク元がないため「−」）と利用ブラウザ名「Lynx/2.8.1pre.9 libwww-FM/2.14」が加わっている。

ざっと、こんな仕組みでApacheのログ書き込み機能は動作している。

access_logで使えるマクロの書式

アクセスログのフォーマット設定はだいたいわかってもらえたはずだ。ところが、自由に書き換えようとするなら、LogFormat内で使えるマクロ形式がわからないといけない。そこで、combinedで設定されている例をもとに解説していこう。

▼リスト4 access_logの例

```
192.168.1.180 - - [12/Nov/1999:20:23:05 +0900] "GET / HTTP/1.1" 200 1622
192.168.1.180 naka - [02/Dec/1999:18:05:45 +0900] "GET / HTTP/1.0" 200 1622
192.168.1.180 unknown - [02/Dec/1999:18:06:19 +0900] "GET / HTTP/1.1" 200 1622
192.168.1.180 - testuser [12/Nov/1999:20:25:14 +0900] "GET /auth/ HTTP/1.0" 200 3497
```

ホスト／リモートユーザー名／認証ユーザー名／時間／最初のリクエスト行／ステータス／バイト数

▼リスト5 error_logの例

```
[Tue Nov 30 01:08:42 1999] [error] [client 192.168.64.180] File does not exist:
/usr/local/apache/htdocs/cb3/office3.gif
```

▼リスト6 httpd.confのCustomLog行でログフォーマットを設定する

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

CustomLog /usr/local/apache/logs/access_log common
```

combinedでは、"%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""という形式でログフォーマットが定義されている。一番外側のダブルクォーテーションでくくられた中身がフォーマット形式のため、

```
%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"
```

という文字列が、ログのマクロということになる。

実際のログファイルと対応させたものが、**図1**だ。リクエスト文字列の最初の1行を示す%rは、ダブルクォーテーションで区切って記録するように、\"%r\"という指定になっている。この「\」は、その次のダブルクォーテーション「"」を文字として渡すためのエスケープキャラクタである。リクエスト行には空白文字が含まれることから、あとでログ処理をするときに迷わないようにするためにはこの処理が必要だ。

たとえばExcelのような表計算ソフトで読み込んで集計するときに、ダブルクォーテーションで区切らないと、スペースがセルの区切り文字として認識される。**図1**であれば、「GET」、「/」、「HTTP/1.0」という3つのセルに分割されてしまうわけだ。同じ処理は、%rだけではなく、リンク元を示す%{Referer}iや、%{User-Agent}iのマクロ部分もダブルクォーテーションで囲む指定になっている。

%>sというマクロは%sというマクロがベースになっている。%sはステータスコードを記録するのだが、単純に%sとすると、リダイレクトがあった場合に、リダイレクト時のステータスコードが記録されてしまう。そこで、「処理最後のステータスコード」の意味で、%>sを使っている。こうした代表的なマクロの一覧を、**表1**にまとめておいた。

ログファイルを拡張するときには、行の最後に付け加えていくようにするといい。ログファイルの巨大化を防ぐために、必要最低限のことだけを記録したいところだが、後述するアクセスログ解析ソフトのAnalogを使うときには、フィールドの並びに合わせてAnalogの設定も変更しなくてはいけない。また、あまりにも独特なログ記録方法を使っていると、ログ解析ソフトの設定変更も必要になる。

%b	転送量（バイト）
%f	ファイル名
%{環境変数名}e	環境変数名
%h	リモートホスト
%{コンテンツ名}I	サーバに送られたリクエストのヘッダ行
%l	リモートユーザー名 (ident認証時)
%p	ポート番号
%P	プロセス ID
%r	最初のリクエスト行
%s	ステータスコード。最後のステータスコードを記録するときは%>s
%t	時間(ロングフォーマット:([日/月/年:時間:分:秒 タイムゾーン])
%{フォーマット}t	時間(書式はstrftimeに準じる)
%T	時間(秒単位)
%u	認証ユーザー名(認証を通った場合)
%U	URL
%v	サーバ名

表1 ログフォーマット指定のマクロ

Analogを使ったログ解析

Apacheが書き出したログを解析するために、LinuxではAnalogというツールがよく使われる。Analogのいいところは、Webサーバの解析ツールだけに、Webブラウザを使ってログの解析結果を閲覧できることだ。また、速度も十分に速く、リアルタイムに解析することも可能だ。

最新版は、バージョン4.0が11月に出たばかりだ。すでにAnalogのRPMを使っているときにはそれをそのまま使えばいいが、4.0をインストールするならば、http://www.statslab.cam.ac.uk/~sret1/analog/から国内のミラーサイトへ行き、そこから入手するといいだろう。makeの手順は、**リスト8**のとおりだ。また、analog-4.0-1.i386.rpmと

▼リスト7 access_logの内容。ログフォーマットをcombinedに変更すると、リンク元とブラウザ名が記録される

```
127.0.0.1 naka - [04/Dec/1999:00:52:49 +0900] "GET /apache_pb.gif HTTP/1.1" 404 295
127.0.0.1 naka - [04/Dec/1999:00:53:13 +0900] "GET / HTTP/1.0" 200 1622 "-" "Lynx/2.8.1pre.9 libwww-FM/2.14"
```

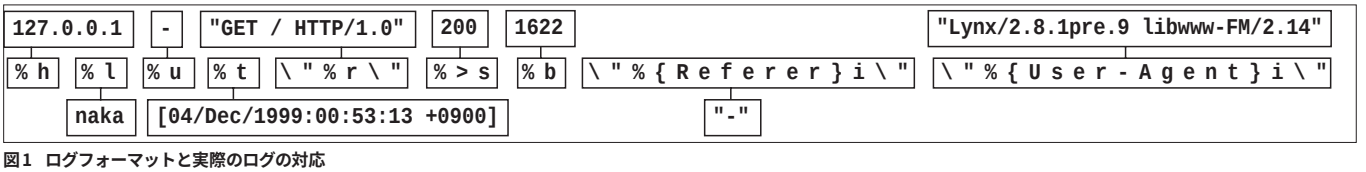


図1 ログフォーマットと実際のログの対応

いうRPMも配布されているので、これを使うのもひとつの方法だ。ここでは、ソースからmakeしたAnalog 4.0をベースに解説していく。

makeしたあとには、anlgform.htmlというファイルができ上がっている。これを使うと、Webブラウザからanalogの解析ができる。まずはこれを試してみよう。

Analogのインストール先は、/usr/local/etc/httpd/のため、本来のドキュメントディレクトリである/usr/local/apache/htdocsからシンボリックリンクを張る。そのあとに、必要なファイルに対してパーミッションを設定し、解析cgiとなるanlgform.plをcgi-binに移動する（**リスト9**）。

この段階では、ソースやオブジェクトファイルがそのまま残っているので、公開サーバに組み込むときには、さらに不要なファイルを整理しておいたほうがよいだろう。

Analogのセットアップ自体は、この作業で終わる。ただし、これだけでは動作しない。さらに、anlgform.htmlとanlgform.plに必要な修正を加えて動くようにする。修正した箇所は、**リスト10、11**のとおりだ。あとは、ブラウザでhttp://ホスト名/analog/というURLを指定すれば**画面1**が出てくる。必要な部分をセットしてProduce statisticsボタンをクリックすると、**画面2**が出てくる。ちゃんと**画面2**が表示されれば、Analogが正しく動作していることになる。

Analog処理つきlogrotateを作る

anlgformは手軽にアクセス集計ができる反面、ひとつ大きな欠点がある。それは、動作するたびに毎回指定したaccess_logファイルを集計し直すという点だ。1日に10万件ものアクセスがあるようなWebサイトであれば、そのたびに集計計算をされていてはたまったものではない。また、Apacheのログは、logrotateによって毎週切り出されている。つまり、週をまたいだ解析ができないわけだ。さらに、解析のたびに集計計算作業が入る。1日あたり数千件のアクセスであれば、毎回集計したところでたいした時間はかからないが、数十万件のアクセスがあるようなサイトでは、ログを見るたびに集計されていてはたまったものではない。

そこで、いちばんベストな方法は、logrotateが動いてログファイルを切り出したときに、同時に集計作業もして集計ファイルのみ別途保存しておくことだ。もしくは、logrotateでApacheのログを切り出すことをやめ、独自のcronを動かして、ログの切り出しと同時に集計もさせるという方法もある。

独自cronであれば、切り出すタイミングも自由に設定できるため、ここではより汎用的という意味から、独自のcronを使った作業を紹介しよう。

まず、logrotateでApacheのログの切り出しを禁止する。

```
# rm /etc/logrotate.d/apache
```

次に、ログファイルの切り出し部分を考えよう。ログファイルを単純に、

```
# cp /usr/local/apache/logs/access_log /tmp/httpd_log-9912.txt
# rm /usr/local/apache/logs/access_log
```

▼リスト8 アクセスログ解析ツールAnalogのmake手順

```
# cd /usr/local/src/analog
# tar xvfz analog4.0.tar.gz
# cd analog4.0
# make
# rm -f *.[och]
# mkdir /usr/local/etc/httpd
# cd ..
# mv analog4.0 /usr/local/etc/httpd/
```

▼リスト9 Analogのセットアップ手順

```
# ln -s
/usr/local/etc/httpd/analog4.0
/usr/local/apache/htdocs/analog
# cd /usr/local/apache/htdocs/analog
# chmod -R o+rx .
# chmod +x lang
# chmod +x anlgform.pl
# cp anlgform.pl
/usr/local/apache/cgi-bin
# ln -s anlgform.html index.html
```

▼リスト10 anlgform.plを追加修正する

```
# 1) uncomment (remove everything before $analog) and edit one of the next two
#    lines to give the location (full pathname) of the analog executable.
#       Unix: $analog = '/usr/local/etc/httpd/analog4.0/analog';
#       Windows: $analog = 'C:\program files\analog 4.0\analog.exe';

$analog = '/usr/local/etc/httpd/analog4.0/analog';    <-- 追加
```

▼リスト11 anlgform.htmlの23～30行目。下記のように修正する

```
<form action="/cgi-bin/anlgform.pl" method="POST">    <-- 変更

<!-- Many systems will want the IMAGEDIR to be different on the form... -->
<!-- than from the commandline, because it should not be within /cgi-bin/ -->
<input type=hidden name="IMAGEDIR" value="/analog/images/">    <-- 変更

<!-- Some users will want to set their logfile like this -->
<input type=hidden name="LOGFILE" value="/usr/local/apache/logs/access_log">
```

としてしまうと、ファイルIDがずれるため、Apacheはログを記録できなくなる。一度access_logを別ファイルにcpして、Apacheの再起動なしに、ログファイルを別ファイルにcpしてから、/dev/nullをaccess_logにcpするという手段もなくはないが、アクセス数が多いときには、別ファイルにcpする時間が無視できない。そこで、ログファイルを動かしたあとに、一度Apacheを再起動して、access_logを正しく処理させるという手順をとる。

一度待避したaccess_logは、analogで処理して、出力された集計結果だけをhtmlファイルとして特定のディレクトリに入れる。こうすれば、過去のログファイルはバックアップするなり削除するなりすればいい。スクリプトとしては、**リスト12**のようなものだ。これに、エラー時の処理と、待避したアクセスログをバックアップもしくは削除する処理を加え、cronで動かす。

logrotateに依存せずに、自分でcron処理することにより、さまざまな処理が加えられる。**リスト12**を/usr/local/apache/cron/logrotate.plとして作り、rootになってから、

```
# cromtab -e
```

と行って、cronテーブルに、

```
0 4 * * * /usr/local/apache/cron
/logrotate.pl
```

を加えておけば、毎日午前4時にログを切り出し、前日4時からの1日分のアクセス記録を作ってくれる。

Analogが作り出したログ集計結果は、/usr/local/apache/htdocs直下に「日付.html」というファイルになっている。これを直接アクセスすれば、その日のログ集計結果がすぐにわかる。ただし、このままでは画像のリンクがすべて外れているのと、ホスト名も正しく記録されていない。そこで、/usr/local/etc/httpd/analog4.0/analog.cfgに、

```
HOSTNAME "[www.ho.point5.co.jp]"
IMAGEDIR /analog/images/
```

を加えてグローバル設定をしておく。HOSTNAMEにはホスト名を、IMAGEDIRとして、analogのイメージファイルのあるドキュメントディレクトリを指定する。これで、画像もホスト名も正しく表示できるようになる。

画面1　Analogのレポートを設定するフォーム

画面2　Analogのアクセスログ解析レポート出力

▼リスト12　/usr/local/apache/cron/logrotate.plプログラム。cronで毎日午前4時に動かす

```
#!/usr/bin/perl

$accesslog='access_log';
$logdir='/usr/local/apache/logs';
$analog='/usr/local/etc/httpd/analog4.0/analog';
$apachectl='/usr/local/apache/bin/apachectl';
$outdir='/usr/local/apache/htdocs';

@dtmp=localtime (time);
if ($dtmp[5]>=100) {
        $dtmp[5] -= 100;
}
++$dtmp[4];

$today=sprintf ("%2.2d%2.2d%2.2d",$dtmp[5],$dtmp[4],$dtmp[3]);
print "start Analog($today)\n";

while (glob("$logdir/*log")) {
        system ("mv $_ $_.$today");
}

system ("$apachectl restart;$analog
$logdir/$accesslog.$today>$outdir/$today.html");

exit;
```

Linuxの日本語環境

文：富樫 秀昭
Text：Hideaki Togashi

日本語の印刷

この連載の冒頭でこのところ連続して触れているTrueTypeフォント・ラスタライザライブラリのFreeTypeだが、1999年8月4日付で、Appleの特許を侵害する可能性のあるコードが見つかったと報告されている（http://www.freetype.org/）。FreeTypeプロジェクトは、Appleの法務部門に問い合わせているが、現時点では未回答のようだ。FreeTypeは、X-TT、 VFlibなどで使われている、TrueTypeフォントを使ううえでキーとなる重要なラスタライザであり、今後の動向が注目される。

今回取り上げる印刷の局面でもFreeTypeは重要な働きをすることになるが、そこにいく前に、まずはLinuxマシンにローカルプリンタをつないで印刷を行う場合を簡単に概観して、その後日本語の印刷についての問題点や対処法などを考えてみたい。今回は、現在一般的に使われているインクジェットカラープリンタの中でも、最新機種であるEPSON PM-800Cを借用できたので、これをLinuxマシンにつないだ環境を見ていくことにしよう。

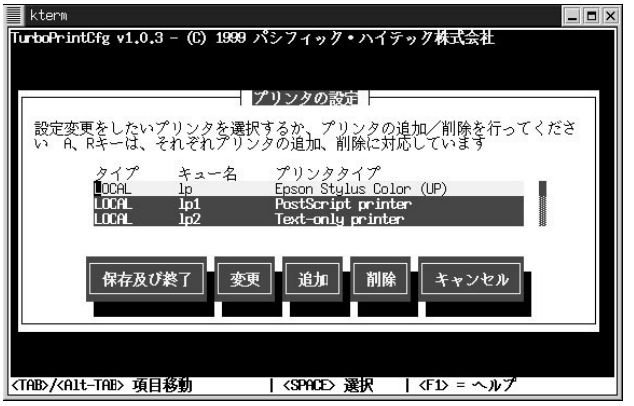
Linuxでの印刷の仕組み

Red Hat Linuxのlinuxconf、control-panel、またはTurboLinuxのturboprintcfg（**画面1**）のように、多くのLinuxディストリビューションでは、プリンタ関連の設定は対話式で自動で行われ、ユーザーが直接設定ファイルを触ることのないようになっているが、UNIX一般の知識として、その仕組みの概略を知っておくことも大切なことだろう。

Linuxに限らずUNIXでは、印刷に際して、lpdというプリンタスプールデーモンが使われるのが一般的だ。プログラムは主に、

- `lpd` — **プリンタスプールデーモン本体**
- `lpr` — **プリントデータ送出**
- `lpq` — **スプールキュー問い合わせ**
- `lprm` — **スプール削除**
- `lpc` — **スプールコントローラ**

からなり、これらは「/etc/printcap」という設定ファイルをもとに動作している。対話式のセットアッププログラムは、まず第一にこのprintcapを自動で設定してくれるものと考えてよいだろう。



画面1　TurboLinuxのプリンタ設定ツールTurboPrintCfg

●printcap

リスト1は、TurboLinuxのturboprintcfgでEPSON PM-800Cのために設定した時の、printcapの該当部分だ。ローカルプリンタを使う際の基本的な設定は、おおむねこのようなものになるだろう。

記述形式は、基本的にtermcapなどと同等で、最初にエントリ名が入り、「：」が区切り文字に使われる。各項目は、値を持つ場合は「＝」でその値が指定される。行が連続する場合は、行末に「＼」が置かれ、1行に1項目を書くのが慣例となっている。「＃」以降はコメントだ。

最初の行のコメント部分に、対話式に指定した内容が書かれている。2行目が指定の始まりで、最初はエントリ名、つまり各プログラムがプリンタ名として認識する名前を指定している部分である。3行目の「sd」はスプールディレクトリ（Spool Directory）の指定で、「/var/spool/lpd」にサブディレクトリが作られて、それが使われるのが普通である。4行目の「mx」はファイルの大きさの指定で、0が指定されると制限なしになる。このように数字の指定の場合は、「＃」が「＝」の代わりに使われる。5行目にある「sh」は、suppress headerを意味し、指定するとカバーページを印字しないようになる。6行目の「lp」がプリンタの接続されているデバイス名の指定である。1つ目のパラレルポートが使われているので、「/dev/lp0」が指定されている。そして最後の行の「if」（Input Filter）は、フィルタとして使われるプログラムの指定である。ここでは、スプールディレクトリの「filter」が指定されている。

つまり、このprintcapの意味するところは、lprが受け取ったデータを/var/spool/lpd/lp/filterで処理し、その結果を/var/spool/lpd/lp/に置き、最終的に/dev/lp0に渡すということになる。その他の指定については、man printcapなどをご覧いただきたい。printcapを変更した場合は、root権限のまま、

```
# ps aux | grep lpd
```

としてlpdのプロセス番号を見て、

```
# kill -9 <lpdのプロセス番号>
```

としてlpdをいったん終了させたうえで、

```
# lpd
```

とlpdを起動し直す必要がある。

●標準はPostScript形式

lpdは、基本的に受け取ったデータをそのままプリンタに渡すものだ。たいていのプリンタは改行などの基本的な制御コードは認識してくれるので、テキストをベタで印刷するぶんには困らないのが普通だが、最近のWindowsプリンタは日本語フォントを持っていないために、日本語を含むテキストはベタでは印刷できないことになる。

これがテキストではなくて画像を印刷するとなると、ビットマップのプリンタ制御コードは各プリンタメーカー、各モデルさまざまで、これまたいろいろある画像フォーマットをいろいろな制御コードに合わせて変換するのは、かなりやっかいである。

そこで、PS（PostScript）プリンタでも非PSプリンタでも、PSを標準のフォーマットとみなして処理するのが、事実上UNIX標準となっている。画像をPSに変換するツールとしては、netpbmという画像変換プログラム群が一般に使われている。たとえば、GIFをPSに変換するには、

```
$ cat file01.gif | giftopnm | pnmtops > file01.ps
```

と変換することになる。ここでは、GIFファイルfile01.gifの内容をパイプでgiftopnmコマンドに渡してpnm（portable anymap）フォーマットに変換し、その出力をpnmtopsコマンドに渡してPSに変換している。このデータは、ファイルにリダイレクトされ、file01.psが作られる。

▼リスト1　turboprintcfgで作られたprintcap

```
##PRINTTOOL3## LOCAL uniprint NAxNA a4 {} U_EpsonStylusColor Default {}
lp:\
        :sd=/var/spool/lpd/lp:\
        :mx#0:\
        :sh:\
        :lp=/dev/lp0:\
        :if=/var/spool/lpd/lp/filter:
```

2段階に分けて変換するのは無駄に思えるかもしれないが、pnmを介することによって、変換プログラムの数をあまり増やさずに、たくさんの画像フォーマットに対応できる。ちなみに、pnmとは、pbm（portable bitmap、2値画像用）、pgm（portable graymap、モノクロ画像用）、ppm（portable pixmap、カラー画像用）の3フォーマットを総称したものである。

JPEGの場合は、djpegというJPEGの圧縮を解くプログラムがあるので、これを使って、

```
$ cat file02.jpg | djpeg -pnm | pnmtops > file02.ps
```

とすることになる。PSプリンタの場合は、最終出力のPSファイルをそのまま出力すればよいのだ。

●Ghostscript

非PSプリンタの場合は、PSファイルをGhostscriptに渡して、処理することになる。Ghostscriptには、元来PSを出力するためにさまざまなプリンタを含むデバイスがサポートされており、それを用いて出力するのだ。現在使っているGhostscriptにどのようなデバイスが組み込まれているかは、

```
$ gs --help
```

と実行すると、その一覧が表示される（**画面2**）。たとえば、escpデバイスが組み込まれていて、ESC/Pプリンタを使うなら、root権限で、

```
# gs -q -dSAFER -dNOPAUSE -sOutputFile=- \
    -sDEVICE=escp filename.ps > /dev/lp0
```

のように実行すると、filnename.psが印刷される。この場合、printcapのifに指定する最も簡単なフィルタは、

```
#!/bin/sh
gs -q -dSAFER -dNOPAUSE -sOutputFile=- \
    -sDEVICE=escp -
```

というものになるだろう。この内容でスクリプト（たとえば/usr/sbin/lf_filter）を作り、root権限に移行して、

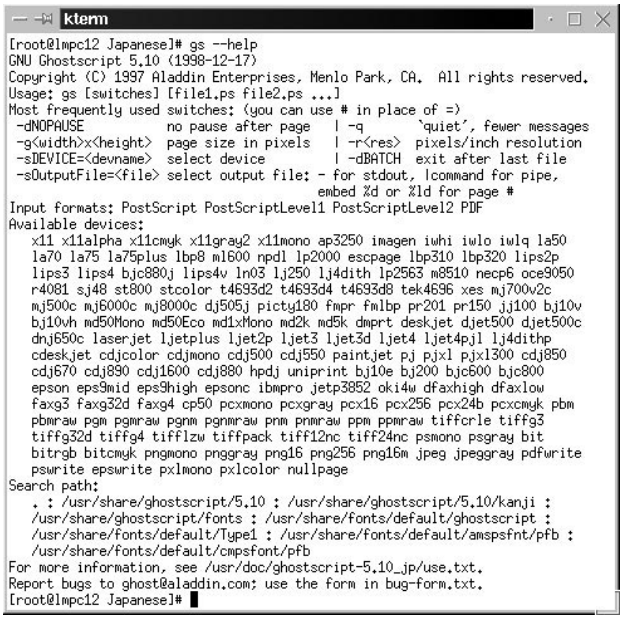
```
# chmod 755 /usr/sbin/lf_filter
# chgrp daemon /usr/sbin/lf_filter
# chown root /usr/sbin/lf_filter
```

とパーミッションを変えて、root権限を与え、printcapのifにフルパスで指定しておけば、このフィルタで印刷できるようになる。printcapを変更した場合は、lpdを再起動するようにしよう。

これまでいろいろなユーザーによっていろいろなプリンタデバイスが次々と開発され、付け加えられてきたが、そういった新しいプリンタデバイスを新たに組み込もうとすると、Ghostscriptそのものをコンパイルし直す必要があり、少々面倒であった。増え続けていく新製品のすべてに対応していくには、膨大な作業量も必要となる。Ghostscriptバージョン5になって、それら各種のプリンタの微妙な指定の違いをパラメータで指定するだけで、いろいろなプリンタで共用できる「uniprint」というデバイスが新たに組み込まれている。uniprintがサポートする範囲であれば、Ghostscript本体の再コンパイルをせずに、必要に応じてパラメータを指定するだけで、各種プリンタが使えるようになったのだ。

●rhs-printfilters

以上で出力に必要な設定の基本的な部分はほぼ終わりである。ユーザーは適切なコマンドでファイルをPSにして、lprに渡してやれば、すべて印刷可能となるわけだが、ど



画面2　Ghostscriptに組み込まれたデバイスを表示

うせならファイルを判別してPSに自動的に変換してくれたほうが、使い勝手は断然よい。

実は、lpdにはこの機能があり、いくつかのファイルを判別して、フィルタを選ぶことができるのだが、判別対象が現在の使われ方にそぐわなくなってしまっている。となると、ifで指定したフィルタの側でファイルの判別をすべてやってしまったほうが効率的だ。

先ほどのturboprintcfgで作られたprintcapに指定されている/var/spool/lpd/lp/filterは、この処理をしてくれるものだ。同時に、ユーザーがUIで指定したプリンタの設定（たとえばGhostscriptのオプションなど）を、そのフィルタの中で生かす仕組みになっている。

このフィルタの本体は、/usr/lib/rhs-printfilters/というディレクトリにある。Red Hatで開発したのでこの名前になっているようだが、TurboLinux 日本語版 4.x、Red Hat Linux 6.1日本語版、LASER5 Linux 6.0、Vine Linux 1.1では、UIはともかく、システムそのものは、すべてこのrhs-printfiltersを使っていた。もちろんRPMパッケージもある。

/usr/lib/rhs-printfilters/にあるのは、フィルタ本体と、設定用ファイルの雛形である。printerdbがプリンタデータベースで、各UIはこのデータベースからプリンタの情報を得、ユーザーの選択に従って/var/spool/lpd/の下の該当ディレクトリに、master-filterからfilterを、general.cfg.in、textonly.cfg.in、postscript.cfg.inからgeneral.cfg、textonly.cfg、postscript.cfgをそれぞれ作っている。printerdbをより充実させて、日本の製品事情にも合ったものが作られることを願っている。

master-filterから作られるfilterは、fileコマンドで入力ファイルの種類を分け、そのファイルに応じた処理をしている。画像ファイルについては、bmp、gif、jpeg、tiffなど通常使われるものはカバーしているようだ。それらの処理そのものは/usr/lib/rhs-printfilters/にあるフィルタ群が行っているが、基本的にはnetpbmのコマンド群を使ってPSに変換するというものだ。

日本語の処理に注目して見てみると、fileコマンドはEUC、Shift-JISを「International language text」と認識し、7ビットJISを「ASCII text（with escape sequences)」と認識するためだろうか、通常のasciiと同等に扱い、テキストをそのままプリンタに渡す設定になっている場合は、日本語テキストもそのままプリンタに渡されるようになっていた。日本語フォントを持たないプリンタでは、日本語は出力されないだろう。さらに、テキストをPSに変換して渡すような設定になっている場合に使われるmpageというプログラムは、そもそも日本語に対応していないようだ。

日本語印刷の問題点

Linuxの印刷の仕組みを一通り概観したところで、日本語を扱ううえでの問題点を改めて見てみよう。

何よりもまず考慮に入れなければならないのは、Linux側の問題というよりも、プリンタそのものだ。

最近の米国ではPSプリンタもかなり値下がりし、数百ドル程度でPSを出力できるレーザープリンタが手に入る。そもそもPS指向であったUNIX環境としては、そういった安価なPSプリンタを入手して、それにそのままPSデータを渡したほうが、好都合である。

PSプリンタでなくても、テキストファイル程度なら以前のプリンタならせいぜい適切な文字コードに変換してやるくらいで、そのままプリンタに渡せば日本語もそれがそのまま印刷されていた。

しかし、ここにきて日本語フォントを搭載しない、いわゆるWindowsプリンタが多く流通するようになってきて、UNIXユーザーにとっては事情が少し複雑になっている。このプリンタは、PSはもちろん受け付けてくれないし、プリンタが日本語フォントを持っていないので、日本語テキストをそのまま渡しても日本語を印刷してくれないのだ。今回試用しているEPSON PM-800Cもまさにそういったプリンタのひとつだ。

上に説明したように、lpdは渡されたものを（スプールして）ほとんどそのままプリンタに渡すというのがその本来の機能であり、もし何らかの変換が必要な場合は、指定したフィルタで変換作業を行う必要がある。

Linuxの場合、そのフィルタはRed Hatのrhs-printfiltersが使われることが多い。その場合、Windowsプリンタでは上記のように、テキストをPSに変換しない設定では日本語が印刷されず、テキストをPSに変換する設定でもmpageが日本語を扱ってくれないために、やはり日本語が印刷されないということになってしまう。つまり、設定したままの状態ではWindowsプリンタではどんな場合でも、

```
$ lpr japanese.txt
```

と、日本語テキストを渡すと、日本語が印刷されないということになるのだ。

日本語テキストの印刷

標準で付属するフィルタに期待する部分は大きいが、そのフィルタでやるにせよ、ユーザーが明示的にやるにせよ、結局は日本語の印刷はPSに変換して行うというのが、現状では最善の策となる。

プリンタが非PSプリンタの場合は、フィルタでそのプリンタに合ったコードにして送り直してやることになるが、そこで使われるのが前述のようにGhostscriptとなる。日本語を処理するのに必要な設定としては、フォントの設定が挙げられるが、Ghostscriptで日本語を処理する場合は、VFlibを用いており、そのあたりのフォントの扱いについては、1999年11月号の本連載のVFlibについての章で扱ったので、ここでは触れない。冒頭にご紹介したFreeTypeを用いている。

そして、PC UNIXを使っていると、ユーザーが自分でテキストをPSに変換してlprに送り込まねばならない場面は、今のところ避けられないことが多いので、この技はぜひとも習得しておきたいところである。そこで、ここでは日本語テキストをPSに変換して印刷するのに便利なツールをいくつか紹介しておきたい。

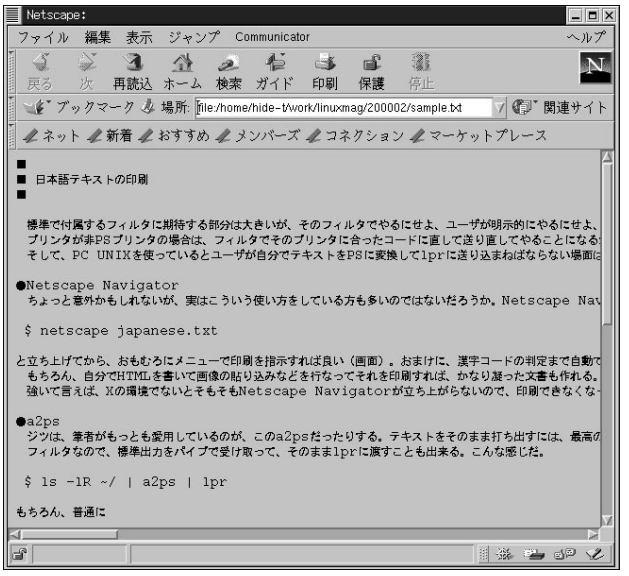
●Netscape Navigator

ちょっと意外かもしれないが、実はこういう使い方をしている方も多いのではないだろうか。Netscape Navigatorで印刷を指示すると、PSで印刷してくれる。つまり、日本語も問題なく印刷してくれるということなのだ。Netscape Navigatorは今は、たいてい標準で付属しており、印刷したいファイルの実体がある場合には、これは便利な方法である。印刷の設定さえちゃんとしてあれば、これで問題なく印刷できる。通常のテキストを引数にして、

```
$ netscape japanese.txt
```

と立ち上げてから、おもむろにメニューで印刷を指示すればよい（**画面3**）。おまけに、漢字コードの判定まで自動でやってくれるので、漢字コードを意識する必要もない。

もちろん、自分でHTMLを書いて画像の貼り込みなどを行ってそれを印刷すれば、かなり凝った文書も作れる。



画面3　Netscape Navigatorは印刷ツールとしても利用可能

強いて言えば、Xの環境でないとそもそもNetscape Navigatorが立ち上がらないので、印刷できなくなってしまうということはある。しかし、現在では、起動するとxdmが最初に立ち上がるディストリビューションも多く、たぶん最も簡単な解ということになるだろう。PSが事実上の標準となっているUNIX環境のアプリケーションならではである。事実上の標準がある程度定まっていると楽ができることの見本といってよいだろう。

●a2ps

実は、筆者がもっとも愛用しているのが、このa2psだったりする。テキストをそのまま打ち出すには、最高のツールだ。Miguel Santana氏のa2psを、Naoki Kanazawa氏がPSの漢字拡張を行い、Kazumasa Utashiro氏がPerlスクリプトにしたものが現在出回っている。他の処理系を一切使わずに、Perlスクリプトだけでテキストを PSに変換してくれるので、使い勝手もよい。

フィルタなので、標準出力をパイプで受け取って、そのままlprに渡すこともできる。こんな感じだ。

```
$ ls -lR ~/ | a2ps | lpr
```

もちろん、普通に、

```
$ a2ps sample.txt | lpr
```

と打ち出すこともできる（**図1**）。

図1　a2psで出力したテキストファイル（1ページ目）

デフォルトでは2段組で左右に出力してくれるが、「-p」オプションでPortraitを指定すれば、普通に1枚に1ページずつ出力してくれるようにもなる。また、「-n」オプションで、cat -nと同じように行番号を表示してくれる。A4がデフォルトになっているが、「-us」「-b4」オプションでサイズを変えることもできる。

残念ながら、ディストリビューションによってはデフォルトでは入っていないことも多いが、その場合はJRPM（Linux Japanese RPM Project、http://jrpm.linux.or.jp/）のUtilities／Textカテゴリーにあるので、ぜひお試しいただきたい。

●TeX、LaTeX

凝った出力をするなら、結局はこれに勝るものはないだろう。Knuth氏の開発した組版ソフトTeXである。使い方によっては、商業出版にも耐えられる品質の出力を期待できる。plain TeX（素のままのTeX）を使うのはちょっと大変だろうから、Lamport氏が開発し、Mittelbach氏らによってバージョンアップされたTeXのマクロパッケージであるLaTeX2eを使うのが簡単だ。TeXのシステムは大規模なもので、以前ならばインストールにいろいろ苦労する羽目になったものだが、現在ではRPMなどの普及によってインストールも簡単になり、そもそもたいていのディストリビューションでは、Linuxのインストール時にTeXもインストールできるので、それをそのまま使えばよい。

```
$ platex text.tex
```

で中間ファイルのDVIファイル（text.dvi）を生成し、それをPSファイルに変換するのだ（**リスト2**）。

PSファイルにする前に出力結果を確かめるには、

```
$ xdvi text.dvi
```

と、xdviでプレビューする。このあたりの操作が面倒なら、Emacs／MuleにAUC TeX、YaTeXなどの便利な統合環境があるので、その環境を満喫するのもよいだろう。

DVIをPSに変換するツールはいくつかあるが、筆者としてはdvipsをお勧めしたい。その機能の豊富さ、対応するマクロの多さなどによって、事実上PSへの変換ツールとして標準の地位を築いている。しかし、残念ながら何故か日本のディストリビューションではdvi2psがインストールされることが多いようだ。dvi2psもdvipsの機能を取り込む形でいろいろと拡張されてきている。dvipsなら、

```
$ dvips text.dvi | lpr
```

とすると、プレビューした通りの印字結果が得られるはずだ（**図2**）。

TeXについては、良書が多く出版されている。それらを参考に美しい組版にTRYされてはいかがだろうか。

●plain2

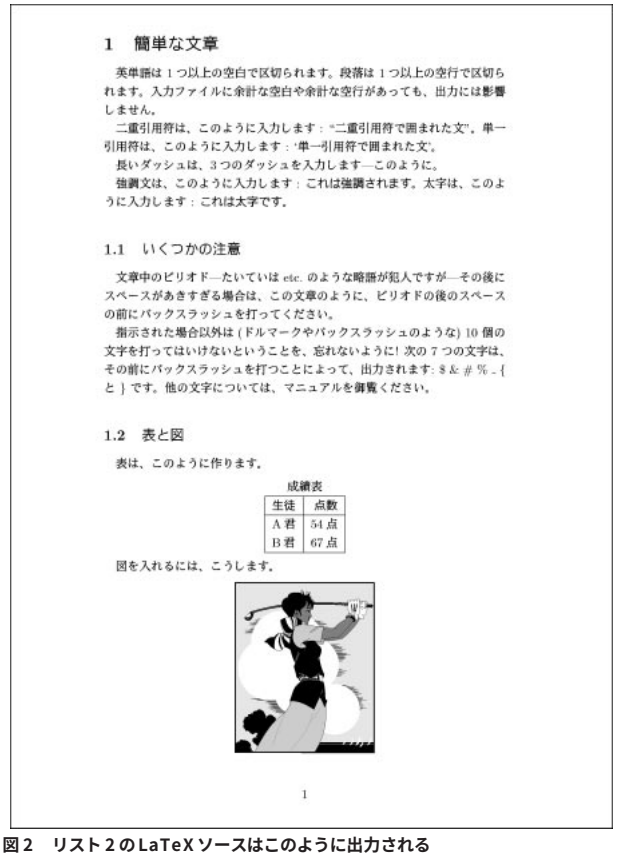
しかしどうしてもLaTeXの入力は面倒だという方には、plain2という選択肢もある。NECの内田明宏氏による、通常のテキストを解析してroff、ないしはLaTeXのソースに自動的に変換してくれるプログラムだ。通常のテキストを、

```
$ plain2 -tex text.txt > text.tex
```

とすると、LaTeXソースに変換してくれる（**リスト3**）。通常のテキストを入力する際に慣用的に使われる記法を基

▼リスト2　簡単なLaTeXのソースの例

```
% '%' の右側に書いてある事は注意書きで、LaTeX は無視します。
%
% 注意!!  次の 10 個の文字は、指示された場合以外は、使ってはいけません。
%               &   $   #   %   _   {   }   ^   ~   \

\documentclass{jarticle}       % 入力ファイルの冒頭にこの行が必要です。
\usepackage{graphics}          % 図を使う指定です。
\begin{document}               % 文書は、この行で始まり、最後の\end{document}で終わります。

\section{簡単な文章}           % このコマンドは、章の見出しを作ります。

英単語は 1つ以上の空白で区切られます。段落は 1つ以上の空行で区切られます。
入力ファイルに余計な空白や余計な空行があっても、出力には影響しません。

二重引用符は、このように入力します : ``二重引用符で囲まれた文''。
単一引用符は、このように入力します : `単一引用符で囲まれた文'。

長いダッシュは、3つのダッシュを入力します---このように。

強調文は、このように入力します : \emph{これは強調されます}。
太字は、このように入力します    : \textbf{これは太字です}。

\subsection{いくつかの注意}    % このコマンドは、節の見出しを作ります。

文章中のピリオド---たいていは etc.\ のような略語が犯人ですが---その後にスペースがあきすぎる場合は、この文章のように、ピリオドの後のスペースの前にバックスラッシュを打ってください。

指示された場合以外は(ドルマークやバックスラッシュのような) 10 個の文字を打ってはいけないということを、忘れないように! 次の 7 つの文字は、その前にバックスラッシュを打つことによって、出力されます: \$  \&  \#  \%  \_  \{  と  \} です。他の文字については、マニュアルを御覧ください。

\subsection{表と図}

表は、このように作ります。

\begin{center}                 % 中央揃えにします。
\begin{tabular}{|l|r|}         % 表のはじまりです。
\multicolumn{2}{c}{成績表}\\
\hline
\textbf{生徒} & \textbf{点数} \\
\hline
A君 & 54点\\
B君 & 67点\\
\hline
\end{tabular}
\end{center}

図を入れるには、こうします。

\begin{center}
\resizebox{!}{5cm}{            % 高さを5cmにして、横のサイズはそれに合わせます。
\includegraphics{golfer.ps}    % 画像を読み込みます。
}
\end{center}

\end{document}                 % 入力ファイルはこのコマンドで終ります。
```



1　簡単な文章

英単語は 1 つ以上の空白で区切られます。段落は 1 つ以上の空行で区切られます。入力ファイルに余計な空白や余計な空行があっても、出力には影響しません。

二重引用符は、このように入力します：“二重引用符で囲まれた文”。単一引用符は、このように入力します：‘単一引用符で囲まれた文’。

長いダッシュは、3 つのダッシュを入力します—このように。

強調文は、このように入力します：これは強調されます。太字は、このように入力します：これは太字です。

1.1　いくつかの注意

文章中のピリオド—たいていは etc. のような略語が犯人ですが—その後にスペースがあきすぎる場合は、この文章のように、ピリオドの後のスペースの前にバックスラッシュを打ってください。

指示された場合以外は (ドルマークやバックスラッシュのような) 10 個の文字を打ってはいけないということを、忘れないように! 次の 7 つの文字は、その前にバックスラッシュを打つことによって、出力されます: $ & # % _ { と } です。他の文字については、マニュアルを御覧ください。

1.2　表と図

表は、このように作ります。

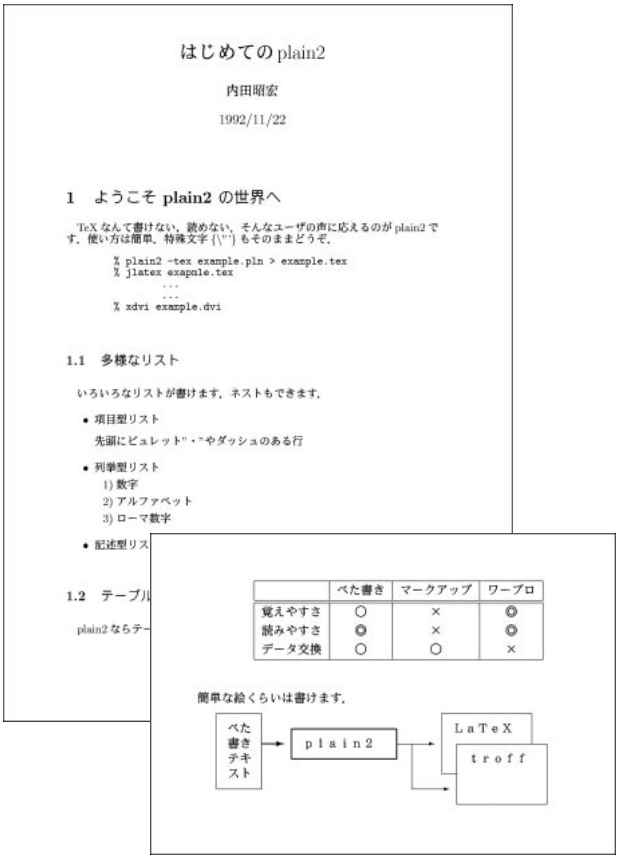
成績表

生徒	点数
A 君	54 点
B 君	67 点

図を入れるには、こうします。

1

図 2　リスト 2 の LaTeX ソースはこのように出力される



はじめての plain2

内田昭宏

1992/11/22

1　ようこそ plain2 の世界へ

TeX なんて書けない，読めない．そんなユーザの声に応えるのが plain2 です．使い方は簡単．特殊文字 {\"'} もそのままどうぞ．

```
% plain2 -tex example.pln > example.tex
% jlatex exapmle.tex
      ...
      ...
% xdvi example.dvi
```

1.1　多様なリスト

いろいろなリストが書けます．ネストもできます．

- 項目型リスト
  先頭にビュレット"・"やダッシュのある行
- 列挙型リスト
  1) 数字
  2) アルファベット
  3) ローマ数字
- 記述型リス

1.2　テーブル

plain2 ならテー

	べた書き	マークアップ	ワープロ
覚えやすさ	○	×	◎
読みやすさ	◎	×	◎
データ交換	○	○	×

簡単な絵くらいは書けます．

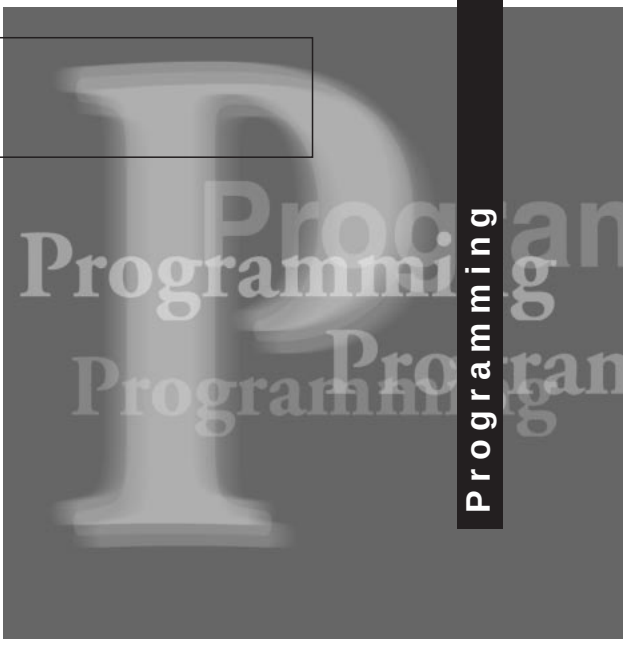
図 3　plain2 を使えば、リスト 3 のテキストがきれいに印刷できる

に解析を行っているので、かなりの文書が結構正しく変換されるようだが、やはり意図通りに変換されないケースもある。このあたりのことについては、plain2 のマニュアル plain2.ps を、ぜひ打ち出してお読みいただきたい。plain2 の記法を理解して書いた文書ならば、箇条書き、表、図など、驚くほどの表現力が発揮される（**図 3**）。そもそも plain2.ps そのものが、plain2 で作られたもので、そのソースも同じディレクトリにあるはずだ。

▼リスト 3　plain2 に入力するテキストファイルの例

```
『はじめての plain2』
                by 内田昭宏
                1992/11/22

1．ようこそ plain2 の世界へ

TeX なんて書けない，読めない．そんなユーザの声に応える
のが plain2 です．使い方は簡単．特殊文字{\"'}もそのままどうぞ．

      % plain2 -tex example.pln > example.tex
      % jlatex exapmle.tex
            ...
            ...
      % xdvi example.dvi

1．1．多様なリスト

いろいろなリストが書けます．ネストもできます．
   ・項目型リスト
      先頭にビュレット"・"やダッシュのある行
   ・列挙型リスト
      1）数字
      2）アルファベット
      3）ローマ数字
   ・記述型リスト

1．2．テーブルと線画の機能

plain2 ならテーブルも簡単に書けます
   ------------------------------------------
   |          | べた書き | マークアップ | ワープロ |
   ==========================================
   | 覚えやすさ |   ○   |     ×     |   ◎   |
   | 読みやすさ |   ◎   |     ×     |   ◎   |
   | データ交換 |   ○   |     ○     |   ×   |
   ------------------------------------------
簡単な絵くらいは書けます．
   ┌──┐  ┌──────┐  ┌──────┐
   │べた│  │      │  │LaTeX │
   │書き│→ │plain2│→ │      │
   │テキ│  │      │  │troff │
   │スト│  │      │  │      │
   └──┘  └──────┘  └──────┘
```

プログラミング工房



前回に引き続き、クロスコンパイラ「Mingw32」を利用して、Linux上からWindowsプログラミングを行う。今回は実践的な例として、GPGを利用したGUIベースの暗号化アプリケーションを作成する。少々難しいところもあるが、知っておけばいろいろと応用が利くはずだ。ぜひ挑戦してみてほしい。

第4回　Windowsプログラミング（2）

文：藤沢敏喜
Text : Toshiki Fujisawa

今回は、前回紹介したクロスコンパイル環境「Mingw32」を利用して、Linux上で実用的なWindowsアプリケーション「wingpg.exe」を作成する。このwingpg.exeは、GNUの暗号化プログラム「GnuPG」（GPG）の共有鍵（symmetric）モードを、Windows上からGUIベースで利用できるように筆者が改造したものである。

多くの読者はWindowsプログラミングに慣れていないと思うので、今回は実用の追求よりも、Windowsプログラミングの雰囲気をつかんでもらうことに主眼に置き、ベーシックなプログラミングスタイルで記述してある。

ただ、今回紹介する手法を参考にすれば、他の多くのプログラムも、大規模な変更をすることなく、Windows上で使えるように改造することもできるようになるだろう。

GPGとは？

GPG（http://www.gnupg.org/）は、GNUによって提供されている暗号化プログラムで、最新バージョンは1.0.0である。PGPの完全な代替プログラムとして利用可能で、PGPから機能の改良とセキュリティに関連した拡張がなされている。DSA、3DES、MD5、SHA-1といった多くの暗号化アルゴリズムをサポートしているが、IDEAやRSAといった特許上の問題があるアルゴリズムを利用していないので、フリーで提供することが可能となっている。

GPGは、コマンドラインからの利用を前提として作成されており、LinuxやFreeBSDを始めとする多くのプラットフォーム上で動作する。Webサイトには Win32版の実行ファイル「gpg.exe」も用意されており、MS-DOSプロンプトからGPGを利用することができる。

gpg.exeの呼び出し

今回作成するプログラムは、ユーザーの便宜を図るため、GUIベースのアプリケーションとしたい。しかし、前述のとおり、gpg.exeはコマンドラインで利用するプログラムである。このようなCUIベースのプログラムを、GUIで使えるようにするにはどうしたらよいだろうか？

まず考えられるのは、ポップアップウィンドウを開いて、ユーザーにファイル名やパスワードを入力させるようなプログラム（wingpg.exe）を作成する方法である。つまり、入力された内容にしたがって、wingpg.exeの中からコマンドライン版のgpg.exeを呼び出すわけである。

C言語では、system関数を使って“system("ls -l")”という記述をすれば、外部プログラムを呼び出すことができる。Win32環境でも、たとえば“system("notepad.exe c:\\config.sys")”とすれば、Cドライブのルートディレクトリにあるconfig.sysをメモ帳で開くことが可能である。これを利用して、wingpg.exeの中からsystem関数でgpg.exeを呼び出せば、当初の目的は（とりあえず）達成

できる（**図1**）。

この方法は、作成そのものはとても簡単なのだが、wingpg.exeとgpg.exeという2つのバイナリを用意する必要があることや、2つのバイナリのインストールやバージョン管理など、管理上のデメリットが大きい。

必要な関数だけのリンク

使用ユーザーが多いと、前述の方法のデメリットは無視できないコストとなってくる。では、このデメリットを軽減できるような方法はないだろうか？

たとえば、必要となる関数だけをGPGのソースコードから抜き出して、直接wingpg.exeにリンクするという方法も考えられる（**図2**）。たとえば、GPGのsymmetricモードでファイルを暗号化するためには、“gnupg-1.0.0/g10/encode.c”の中で定義されている、encode_symmetric関数を呼び出せばよい。この方法なら、コンパクトなバイナリが1つだけになり、エラー処理などの自由度も高い。

しかしこの方法は、GPGのソースコードを詳細に理解していることが前提となる。特に、必要とする関数がグローバル変数や多くの関数に依存していたりすると、その初期化方法を綿密に調べなければならない。

GPGのソースコードは、Cのプログラムだけでも6万行を超える大きさで、ヘッダファイルやドキュメンドまですべて含めると、全部で17万行もある。このサイズのソースコードをすべて理解するのは、かなりたいへんな作業となる。また、バージョンアップなどで、関数呼び出しなどのGPGの内部構造が変更されると、wingpg.exeから呼び出す部分もそれに合わせて変更しなければならず、あとあとのメンテナンスもたいへんである。

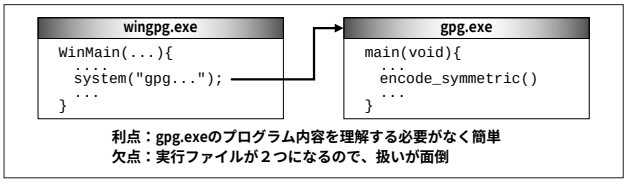


図1　wingpg.exeからのgpg.exeの呼び出し

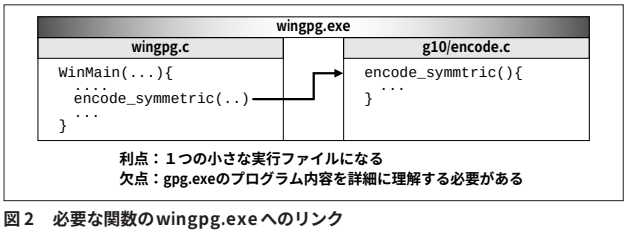


図2　必要な関数のwingpg.exeへのリンク

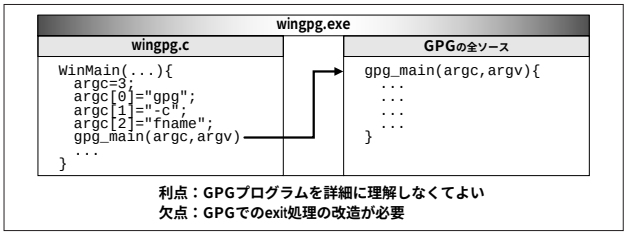


図3　GPGの全関数のリンクとGPGのmain関数の呼び出し

main関数の呼び出し

上記で述べたように、gpg.exeをsystem関数で呼び出す方法と、必要とする関数（encode_symmetric）をリンクする方法にはそれぞれ欠点がある。そこで、GPGのmain関数をwingpg.exeから直接呼び出す方法を考えてみる（**図3**）。

つまり、gnupg-1.0.0/g10/g10.cの519行目にある、

```
int main( int argc, char **argv )
```

という関数を

```
int gpg_main( int argc, char **argv )
```

というように書き換え、このgpg_mainをwingpg.exeから呼び出すようにするわけである。

この場合、gpg.exeがコマンドラインで、

```
% gpg -c filename.txt
```

というように起動するような仕様とすると、wingpg.exeから呼び出す部分は次のようなコードを記述すればよい。

```
argc = 3;
argv[0] = "gpg";
argv[1] = "-c";
argv[2] = "filename.txt";
gpg_main(argc,argv);
```

argcはコマンドライン引数の数（コマンド名も含む）で、argvはコマンドライン引数となる文字列へのポインタである。つまり、このようにしておけば、GPGのmain関数にはあたかもコマンドラインから引数を渡されたのと同じように見えるわけである。

この方法なら、GPG側のプログラムの詳細を深く理解する必要はない。また、GPGがバージョンアップされても、コ

マンドラインシンタックスが変更されない限り、呼び出し側のプログラムを変更する必要はない。さらに、GPGで使用されている6万行のソースコードは、すべてwingpg.exeの中にリンクされるので、ユーザーはwingpg.exeだけをインストールすればプログラムを使用できる。

exit処理

この「wingpg.exeからGPGのmain関数を呼ぶ」方法なら、すべて問題なくいけそうに思える。しかし、実はこの方法にも問題がある。それはexit関数の扱いである。

多くのプログラムでは、ある関数内でエラーが起きたときは、その関数中で直接exit関数を呼び出して終了する。つまり、今回のようにwingpg.exeからgpg_main関数を呼び出しているような場合、exit関数が呼び出されるとwingpg.exe自体が終了してしまうことになる。これでは非常に都合が悪い。エラーが起こっても、gpg_main関数を呼び出した側に制御が戻るようにしなければならない。

このようなエラー処理の常套手段として、C言語の標準ライブラリにあるsetjmp／longjmpを使うことが考えられる。setjmp／longjmpは、簡単にいうと「関数間を越えるGOTO文」のようなもので、setjmpで設定した場所にlongjmpで飛ぶことができる。

このsetjmpを利用して、gpg_main関数の最初の部分にその関数からリターンする（wingpg.cの呼び出し部分へ戻る）コードを書いておく。そして、GPGでexit関数を呼んでいるすべての場所で、exit関数の代わりに「EXIT」という名前の関数を用意しておき、その中でlongjmpを呼べば、上記で述べた問題は解決することになる。なお、このsetjmp／longjmpの詳細については、204ページの「ステップアップC言語」を参照してほしい。

コンソール出力

実は、exit関数の扱い以外にも問題がもうひとつある。それは、コンソールへのメッセージ出力だ。

コマンドラインで使うことを前提として書かれたプログラムは、エラーメッセージはprintf文などを利用してコンソールに出力している。しかし、GUIプログラムではprintf文を使うことはできない。したがって、メッセージ出力を行うために何らかの変更が必要となる。

大規模なプログラムでは、このエラーメッセージの部分はまとまった関数になっていることが多い。GPGも同様で、“gnupg-1.0.0/util/ttyio.c”というファイルでまとめて定義されている。この中のtty_printf関数などを、ポップアップウィンドウを開いて表示するものに置き換えればよい。

パスワードの受け渡し方法

バッチモードの場合、gpg.exeは標準入力を使ってパスフレーズを受け渡すようになっている（これは、セキュリティ上の理由による）。たとえば、パスフレーズを記述した「pass.txt」というファイルを用意しておき、

```
$ gpg --passphrase-fd 0 abc.gpg < pass.txt
```

としてパスフレーズを受け渡す。しかし、Windowsはファイルのセキュリティが貧弱であるため、この方法をとるのは適切ではない。そこで、今回はgpg_main関数の第3引数を使ってパスワードを渡すことにした。つまり

```
int gpg_main( int argc, char **argv, char *pass )
```

として、“g10/passphrase.c”と“g10/g10.c”の2つのファイルにある“--passphrase-fd”を処理する部分を改造することで対応している。

WinMain関数

さて、GPGの呼び出し方法、exit処理、エラーメッセージ処理といった、要の部分がわかったところで、wingpg.exeのメインの流れを見てみよう。

wingpg.exeは、起動するとまずWinMain関数が実行される（**リスト1**）。ここで書いたWinMain関数は、Windowsプログラムの教科書に載っているような標準的なもので、前回説明したインスタンスハンドル（プロセスIDのようなもの）や、コマンドライン引数lpCmdLineなどがWindowsから渡される。

■WinMain関数に渡された引数の処理

まず、**リスト2**で定義されているグローバル変数「global_hInstance」にWindowsのシステムから渡されたインスタンスハンドルを保存する。

Windowsでは、ファイルをドラッグしてプログラムを起動すると、そのファイル名がWinMain関数のコマンドライン引数lpCmdLineとして渡される。しかし、この

lpCmdLineは単純な文字列で、C言語におけるargc、argvのような処理しやすい形ではない。さらに、渡されるファイル名も、MS-DOS時代の遺物である8+3文字のショートファイル名である。これではあまりにも扱いにくいので、cmdarg_setup関数を用いてロングファイル名に変換し、通常のmain関数で渡されるような解析しやすい形でglobal_argc変数とglobal_argv変数に格納する。なお、このcmdarg_setup関数は、付属CD-ROM中のcmdarg.cの中で定義されている。

■ウィンドウクラスの登録

次にウィンドウクラスを登録する。ここでは、最初に

▼リスト1　WinMain関数

```
int PASCAL
WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                                        LPSTR lpCmdLine, int nCmdShow)
{
        HWND            hWnd;
        MSG             msg;
        WNDCLASS        wc;
        static char     *MainWindowClassName = "WinGPG";

        global_hInstance = hInstance;
        if( (global_argc = cmdarg_setup( lpCmdLine, global_argv )) == -1 ){
                mbox("ERROR", "illegual command line [%s]", lpCmdLine );
                return 0;
        }

        if( ! FindWindow(MainWindowClassName, NULL) ){
                wc.style         = 0;
                wc.lpfnWndProc   = MainWndProc;
                wc.cbClsExtra    = 0;
                wc.cbWndExtra    = 0;
                wc.hInstance     = hInstance;
                wc.hIcon         = NULL;
                wc.hCursor       = NULL;
                wc.hbrBackground = (HBRUSH)(COLOR_WINDOW+1);
                wc.lpszMenuName  = NULL;
                wc.lpszClassName = MainWindowClassName;
                if( ! RegisterClass(&wc) ){
                        return FALSE;
                }
        }

        if( ! (hWnd = CreateWindow(MainWindowClassName, "WinGPG",
                WS_OVERLAPPEDWINDOW,
                CW_USEDEFAULT,  CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
                NULL, NULL, hInstance, NULL )) ) {
                return FALSE;
        }

        ShowWindow(hWnd, SW_HIDE);
        UpdateWindow(hWnd);

        while(GetMessage(&msg, NULL, (UINT)NULL, (UINT)NULL )){
                TranslateMessage(&msg);
                DispatchMessage(&msg);
        }
        return msg.wParam;
}
```

ウィンドウハンドル
ウィンドウメッセージ
ウィンドウクラス
コマンドラインを解析
コマンドラインの文字列
引数のクラスが登録されているかを調査
コールバック関数
NULLを指定するとメニューを使わない
ウィンドウクラスの登録
ウィンドウの作成
ウィンドウの位置とサイズをデフォルト設定
ウィンドウの表示制御
メインウィンドウを表示しないようにする
WM_PAINTメッセージを送信し、画面を更新
仮想キーコードの処理
ウィンドウに送られるメッセージを受け取る
コールバック関数へメッセージを送信

FindWindow関数でウィンドウクラスが登録されていないことを確認してから、ウィンドウクラスを定義するwc変数に各種のパラメータをセットし、RegisterClass関数でそのクラスを登録している。ここで重要なのは、

```
wc.lpfnWndProc   = MainWndProc;
```

の行で、これはウィンドウを処理するコールバック関数の定義である。コールバック関数とは、Windowsが呼び出す関数である。たとえば、あるウィンドウに対し、キーボードやマウスのイベントが送られると、そのウィンドウに対応するコールバック関数が呼ばれる。

■ウィンドウの作成

ウィンドウクラスで一般的な特性を定義しておくと、そのウィンドウクラスを基に、CreateWindow関数を使って、表示位置などが異なるウィンドウを作成することができる。

なお、CreateWindow関数で作成された直後のウィンドウは、可視状態になっていない。これらのウィンドウは、ShowWindow関数とUpdateWindow関数を利用して初めてユーザーに見える状態になる。しかし今回は、メインウィンドウを表示しないため、ShowWindow関数の引数として“SH_HIDE”を指定し、不可視の状態にしてある。

■メッセージの処理

上記のような一連の初期化作業が終了すると、プログラムはWindowsからのメッセージ待ちとなる。そして、マウス入力といった種々のイベントが送られると、GetMessage関数でそれを受け取り、TranslateMessage関数やDispatchMessage関数でそれを処理する。つまり、送られたメッセージをWindowsが処理して、そのメッセージによって先ほど登録したコールバック関数がWindowsによって呼び出されるわけである。

コールバック関数MainWndProc

コールバック関数MainWndProcは、「WM_CREATE」と「WM_DESTROY」という2つのメッセージを処理している（**リスト3**）。WM_CREATEは、ウィンドウが最初に作成されたときに送られるメッセージで、ここではexec_main関数を呼び出し、暗号化／復号化などの処理を行っている。

▼リスト2　ヘッダファイルのインクルードとグローバル変数定義

```
#include <windows.h>
#include "wingpg.rh"
#include "cmdarg.h"
static HINSTANCE        global_hInstance;
static int              global_argc;
static char             *global_argv[4096];
static char             global_passwd[128] = "\0";
```

Win32の標準ライブラリ／リソースファイルのヘッダ／コマンドライン解析のヘッダ／インスタンスハンドル／コマンドライン引数の数／コマンドライン引数の文字列配列／パスワードの入力用文字列バッファ

▼リスト3　コールバック関数MainWndProc

```
static LPARAM CALLBACK
MainWndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
        switch(message){
        case WM_CREATE:
                exec_main(global_hInstance, hWnd, global_argc, global_argv );
                PostQuitMessage(0);
                return 0;
        case WM_DESTROY:
                PostQuitMessage(0);
                return 0;
        default:
                return DefWindowProc(hWnd, message, wParam, lParam);
        }
}
```

ウインドウ作成時に送られてくるメッセージ／インスタンスハンドル／メイン（親）ウインドウを示すハンドル／ウインドウを閉じる／ウインドウを閉じる操作をしたときに送られてくる／さまざまなメッセージのデフォルト処理

一方、WM_DESTROYは、ウィンドウの右上にある「×」ボタンを押したときなどに送られるもので、そのウィンドウを閉じる際の処理を行う。また上記の2つ以外のメッセージ（、ウィンドウの最少化など）が送られた場合は、DefWindowProc関数がそれを処理することになる。

exec_main関数

exec_main関数は、今回のプログラムのメイン処理部分である（**リスト4**）。ここではリストにしたがって順に説明していこう。

1.ファイル名をinp_fname変数へセットする

まず、暗号化／復号化を行うファイル名を取得する。wingpg.exeが起動されるのは、wingpg.exeのアイコンがダブルクリックされた場合と、wingpg.exeのアイコン（またはショートカット）へ暗号化するファイルをドラッグした場合の2種類がある。

ダブルクリックされた場合、argcは“0”となるので、get_fname関数を用いてWindows標準の［ファイル選択］ダイアログボックスを開いてユーザーにファイル名を選択させ、inp_fname変数にそのファイル名をセットする。

また、ドラッグして起動した場合はargcが“1”となり、ドラッグしたファイル名が保存されているargv[0]の内容を、strcpy関数を用いてinp_fname変数に保存する。

なお、今回作成するプログラムは、構造を単純化するため、argcが2以上、つまり複数のファイルがドラッグされた場合の対応は実装していない。したがって、もしも2つ以上のファイルがドラッグされた場合は、wingpg.cで定義してあるmbox関数を用いて、メッセージボックスウィ

▼リスト4　msg_xxxとexec_main関数

```
char *msg_title        = "暗号化、または復号化するファイルを選択して下さい";
char *msg_not_found    = "%s が見つかりません";
char *msg_exec_fail[2] = {"暗号化に失敗しました",
                          "復号化に失敗しました" };
char *msg_make_fail[2] = {"暗号化できませんでした",
                          "復号化できませんでした" };
char *msg_success[2]   = {"正常に暗号化して、 %s を作成しました",
                          "正常に復号化して、 %s を作成しました" };
char *msg_exist_err[2] = {"%s が既に存在するので暗号化できません",
                          "%s が既に存在するので復号化できません"};

static int
exec_main(HINSTANCE hInstance, HWND hWnd, int argc, char **argv)
{
        char    *passwd;
        char    inp_fname[MAX_PATH_LEN];
        char    out_fname[MAX_PATH_LEN];
        enum    { ENCODE=0, DECODE=1 } mode;
        char    *opt[2] = {"-c", "" };

        /* (1) ファイル名を inp_fname へセットする  */
        if( argc == 0 ){
                if(get_fname(inp_fname, hInstance, hWnd, msg_title) == NULL){
                        return 0;
                }
        }else
        if( argc == 1 ){
                strcpy(inp_fname, argv[0] );
        }else{
                mbox("ERROR", "too many files");
                return 0;
        }
        /* (2) ファイル名の確認(存在しなければエラー */
        if( ! is_file_exist( inp_fname ) ){
                mbox("ERROR", msg_not_found,inp_fname);
```

ユーザーが入力するパスワード文字列
入力のファイル名
出力のファイル名
ENCODE、DECODE時にそれぞれ指定するオプション
定数を定義する（#define ENCODE 0と等価）
ファイル名選択ウインドウを開く
ウインドウのタイトル表示文字列
選択したファイル名をこの変数に保存
マウスでドラッグされたファイル名

右ページへ続く↗

ンドウを開いて、ユーザーに警告する。

2.ファイル名の確認

1.で暗号化／復号化するファイル名をinp_fname変数に取得できたので、次に実際にそれが存在するかを調べる。存在しなかった場合はエラーとする。

3.出力ファイル名の生成

is_gpg_fname関数を用いて、inp_fname変数にセットされているファイル名の拡張子を確認する。拡張子が“.gpg”の場合は、mode変数に複号化を意味する「DECODE」をセットし、“.gpg”を削除した出力ファイル名をout_fname変数へ保存する。

一方、拡張子が“.gpg”ではない場合は、mode変数に「ENCODE」をセットし、“.gpg”を付加した出力ファイル名をout_fname変数へ保存する。

4.出力ファイルが存在していないかを確かめる

3.でout_fname変数にセットされたファイル名が実際に存在するかを調べ、存在した場合は上書きしないで、エラーメッセージを示して終了する。

5.パスワードの入力

次にget_passwd関数を呼び、パスワード入力のためのウィンドウを開き、ユーザーにパスワードを入力させる。なお、このget_pwsswd関数では、入力した文字を「＊」（アスタリスク）で表示している。

6.GPGを起動

ここまでで得られた暗号化／複号化、ファイル名、パス

```
        return 0;
    }
    /* (3) 出力ファイル名の生成 */
    if( is_gpg_fname(inp_fname) ){
        mode = DECODE;                          ← 復号化するモード
        strcpy( out_fname, inp_fname );
        out_fname[ strlen(out_fname) - 4 ] = '\0';
    }else{                                      ← 後ろから4文字目以降を削除
        mode = ENCODE;                          ← 暗号化するモード
        sprintf( out_fname, "%s.gpg", inp_fname );
    }                                           ← 後ろに".gpg"を付加する
    /* (4) 出力ファイルが存在していないかを確かめる */
    if( is_file_exist( out_fname ) ){
        mbox("ERROR", msg_exist_err[mode], out_fname);
        return 0;                               ← 引数のファイルが存在するかを調べる
    }
    /* (5) パスワードの入力 */
    if( (passwd = get_passwd(hInstance, hWnd)) == NULL ){   ← パスワード入力ウインドウを開く
        mbox("ERROR", "PASSWD ERROR");
        return 0;
    }                                           ← ウインドウで入力されたパスワード文字列
    /* (6) gpgを起動 */
    if( call_gpg( opt[mode], inp_fname, passwd ) != 0 ){
        mbox("ERROR", msg_exec_fail[mode] );
        return 0;
    }                                           ← gpg_main関数を呼び出す
    /* (7) 正常に作成できたかを確認 */
    if( ! is_file_exist( out_fname ) ){         ← 正常に終了すればこのファイルが作成される
        mbox("ERROR", msg_make_fail[mode] );
        return 0;
    }
    /* (8) 正常に作成できたというメッセージを表示 */
    mbox("OK", msg_success[mode], out_fname);
    return 0;                                   ← このファイルが正常に作成されたことを報告
}
```

ワードという3つを引数として、call_gpg関数（**リスト5**）を呼び出す。前述したように、この関数ではあたかもコマンドラインから呼び出したかのようにargc、argvを設定してから、gpg_main関数を呼び出す。

7.正常に作成できたかを確認

正常に暗号化／復号化できた場合は、out_fname変数に示す名前のファイルが作成されているので、それをチェックする。

8.正常に作成できたというメッセージを表示

最後に、正常に暗号化／復号化したことをユーザーに報告をする。

wingpg.exeのコンパイル

さて、プログラムの流れがわかったところで、実際にwingpg.exeをコンパイルして作成してみよう。

このプログラムをコンパイルするには、前号で紹介したクロスコンパイラ環境「Mingw32」が必要となる。もしも、テストするシステムにこの環境が構築されていない場合は、Linux magazine 1月号掲載の195ページからの記事を参考に、あらかじめMingw32が動作する環境を構築しておいてほしい。また、コンパイルには、/var/tmp/に200Mバイト以上の空き容量が必要になるので、あらかじめ確認したうえで実行してほしい。

まず、今月号の付属CD-ROMをマウントし、CD-ROMの/Linuxmag/Programming/にあるwingpg-1.0.0aディレクトリをカレントディレクトリとしてから、makeコマンドを実行すれば自動的にコンパイルが始まる（コンパイルはroot権限がなくても可能）。最近の速いマシンであれば、数分で/var/tmp/workにgpg.exeとwingpg.exeが生成される。

このプログラムは、LASER5 Linux 6.0、TurboLinux 4.2、FreeBSD-3.2で正常にコンパイルが行えることを確認してある。また、コンパイルがうまくいかなかった読者もwingpg.exeを体験できるよう、付属CD-ROMにFreeBSDでコンパイルしたgpg.exeとwingpg.exeを収録してある。

コンパイルの詳細

makeを実行すると、まずgnupg-1.0.0.tgzを展開し、次のパッチを自動的に当てる。

- **Mingw32でコンパイルできるようにするパッチ**
- **Win32用のパッチ**
- **main関数をgpg_mainに変更（第3引数の追加）**
- **exit関数をlongjmpを呼び出すEXIT関数に変更**

そのあと自動的にconfigureが実行され、必要なライブラリを作成してからgpg.exeの生成を始める。gpg.exeの生成が終わると、次にwingpg.c、ttyio.c、cmdarg.cをコンパイルする。なお、gpg.exeのmain関数はgnupg-1.0.0/g10/g10.cで定義されているが、このmain関数は先ほど当てたパッチにより、次のように変更されている。

```
#ifdef USE_WINMAIN
   gpg_main( int argc, char **argv, char *passwd )
#else
```

▼リスト5　call_gpg関数

```
static int
call_gpg( char *opt, char *fname, char *passwd )
{
        int             argc = 0;
        char            *argv[10];
        extern int      gpg_main(int argc, char **argv, char *passwd);

        argv[argc++] = "gpg";
        argv[argc++] = "--no-default-keyring"; /* not make keyring */
        argv[argc++] = "--passphrase-fd";
        argv[argc++] = "0";
        if( *opt != '\0' ){
                argv[argc++] = opt;
        }
        argv[argc++] = fname;
        return gpg_main( argc, argv, passwd );
}
```

```
    main( int argc, char **argv )
#endif
```

また、同様にg10.cで定義されているEXIT関数も、単純にexitするか、longjmpで戻るかをUSE_WINMAINマクロによって切り替えるようにしてある。したがって、gccのコマンドラインで、“ gcc - DUSE_WINMAIN -c g10.c ”として、USE_WINMAINというマクロ定義をしてg10.cをコンパイルすると、wingpg.exe用のg10.oが作成されることになる。

ここまでで、wingpg.exeを生成するために必要なwingpg.o、cmdarg.o、g10.o、ttyio.o が作成されたので、これらのオブジェクトとそのほかのオブジェクト（gpg.exeと共通に使うもの）をリンクし、wingpg.exeを作成する。

wingpg.exeの実行テスト

コンパイル作業が終了したら、/var/tmp/work/に生成されているwingpg.exeをftpやsambaなどを利用してWindowsマシンのデスクトップなどにコピーする。

このwingpg.exeのアイコン（南京錠の形）をダブルクリックすると、先月号で説明したようにファイル選択のダイアログボックスが表示され、暗号化するファイルを選択できる。

一方、エクスプローラなどから暗号化するファイルをwingpg.exeのアイコンへドラッグするとパスワードを聞いてくるので、適当なパスワードを入力すれば暗号化が可能になる（**画面1**）。ただし、今回のプログラムでは複数ファイルのドラッグはサポートしていない。

ここで入力するパスワードは、マシンにログインするためのパスワードのことではなく、暗号化するためのものなので、任意の文字列でよい（ただし、復号化するときまでそのフレーズを覚えていないと、元に戻らなくなってしまう）

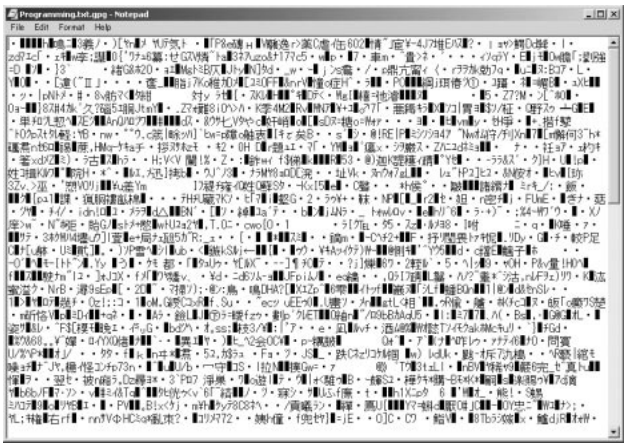
なお、暗号化（または復号化）してできるファイルと同一のファイル名を持つファイルが同じディレクトリに存在する場合は、エラーとなり処理は中断される。このプログラムでは、暗号化したファイルは暗号化する元のファイルがあったディレクトリ内に作成されるので、何度か試すときはこのことに注意してほしい。

また、拡張子“ .gpg ”をwingpg.exeに割り当てておくと、暗号化されたファイルをダブルクリックするだけで、自動的にwingpg.exeを立ち上げることができるので、より便利に使えるだろう。

さらなる応用へ

前回はLinuxやFreeBSD上でWindowsプログラムを作成する方法を紹介し、今回はその具体的な応用方法について述べた。今回は、コマンドラインで使うプログラムをGUIプログラムから扱う方法がメインとなったため、Win32パッチの詳細や、パスワード入力やリソースの話など、Windowsプログラミングの詳細については解説する余裕がなかった。このあたりについては付属CD-ROMに収録してある実際のソースコードや、Windowsプログラミングの入門書などを参考にしていただきたいと思う。また、今回の記事で解説できなかった部分は、http://fujisawa.gr.jp/でフォローする予定である。

なお、今回はソースを簡単にするため、暗号化時のパスワードを1回だけしか入力しないようになっている。これも入力するテキストボックスを2つ用意するなどの改造をしたり、より本格的に公開鍵を扱えるようにするといった応用事例が考えられるので、読者のみなさんもぜひ挑戦してみてほしい。



画面1　元のテキストファイル（左）と暗号化されたファイル（右）

特別講座

ステップアップC言語

setjmp/longjmp

大規模なプログラムでは、エラー処理をどのように実現するかを考える必要がある。特に、親関数から子関数を呼び出し、さらに孫関数を呼び出している場合には、孫関数でエラーが起きたときにどのような対処をするかが問題となる。

■さまざまなエラー処理

この対処には次のような方法が考えられる。

1.孫関数でエラーが起きたらexit関数でプログラムを終了する。
2.孫関数でエラーが起きたらエラーコードを返し、子関数はそれを見て親関数にエラーコードを返し、親関数は子関数からのエラーを調べてプログラムを終了する。
3.孫関数でエラーが起きたら、親関数へ一気に戻り、親関数はエラー処理をして終了する。

ここに挙げた1.の方法は簡単だが、複雑なエラー処理には向かない。一方、2.の方法はうまくプログラムを書くと、どの関数をたどってエラーになったのかという履歴ログを作成するといった高度なエラー処理ができるが、コードはかなり複雑で面倒なものになってしまう。

3.の方法は、ジャンプ命令が使える言語では簡単に記述でき、統一したエラー処理を行えるという利点がある。

■goto文

C言語で用意されているgoto文は、ラベルを設定しておくことで、プログラム内の任意の場所へ無条件にジャンプするもので、リストc-1に示したような形式で利用する。このgoto文は、多用するとプログラムの流れがわかりづらくなるので、C言語の入門書にはあまり使わないように注意書きされていることが多いが、多重ループからの脱出時のように、使い方によっては非常に便利なものである。

ここまで読むと、今回のテーマである「関数呼び出し時のエラー処理」にも使えそうにも思えるが、goto文がジャンプできるのは関数内だけで、関数を越えたジャンプができないので、今回のような場合には使えない。そこで、登場となるのがsetjmp関数とlongjmp関数である。

■setjmp/longjmp

setjmp関数はジャンプする場所を設定する関数で、longjmp関数はその設定した場所へジャンプする関数である。この2つの関数は**リストc-2**のような形式で利用すればよい。

ここでsetjmpを呼び出すと、現在のスタックポインタなどの状態が、**リストc-2**の2行目でjmp_buf型の構造体として定義されているexit_envに保存され、setjmp自身は戻り値として“0”を返す。

そして、子関数、孫関数が呼ばれ、孫関数でlongjmpを呼び出すと、longjmp関数内では、第1引数で渡されたexit_env構造体に保存されているスタックの状態がリストアされるため、あたかも最初にsetjmpが実行されたときと同じ状態になる。ここで、アセンブラレベルでのRETURN命令を実行すると、スタックに積まれている戻り番地、つまりsetjmpを実行した関数へとリターンする。

ここでのスタックとは、サブルーチンがコールされるときに、ローカル変数やその関数が呼ばれた親関数のプログラムの戻りアドレスを保存するメモリ領域のことである。CPUには、現在どのアドレスまで使用されているかという状態を示すレジスタがあり、「スタックポインタ」と呼ばれている。

なお、このリターンの際の戻り値としては、longjmpの第2引数が用いられるため、longjmp実行時における

```
if( setjmp( exit_env ) != 0 ){
    エラー処理();
    return -1;
}
```

での、setjmpの戻り値は“1”となるため、エラー処理が行われることになる。

なお、ここまでの説明でわかるように、longjmpはあらかじめsetjmpを実行した場所にしか飛ぶことはできず、無制限なジャンプができるわけではないことに注意が必要である。

つまり、関数を越えたジャンプが可能とはいっても、自分の祖先の関数が歩いた足跡がある道にしかジャンプできないのである。

▼リストc-1　C言語におけるgoto文の例

```
void
main(void)
{
    ......
    for(;;){
        .....
            if(エラー){
                goto label_err;
                .....
            }
    }

    label_err:
        エラー処理
}
```

▼リストc-2　setjmp/longjmpの使用例

```
#include <setjmp.h>
jmp_buf exit_env;

void 孫関数(void)
{
    if(エラー){
        longjmp( exit_env, 1 );
    }
}

void 子関数(void)
{
    孫関数();
}

void 親関数(void)
{
    if( setjmp( exit_env ) != 0 ){
        エラー処理();
        return -1;
    }
    子関数();
}
```

PostgreSQLを極める

データベースの利用が本格的になってくると、データの変化をリアルタイムに知ることが重要になってくるが、頻繁なデータアクセスは、サーバに負荷をかけてしまう。今回紹介する「クライアント間通知機能」を利用すれば、サーバに負荷をかけずにデータの変化を知ることが可能となる。

第4回 クライアント間通知機能

文：片岡裕生
Text : Hiroki kataoka

本号は、西暦2000年の大台になってから発売される最初のLinux magazineとなるわけですが、読者のみなさんは無事に新年を迎えられたでしょうか？　ちなみに、この記事を書いている時点では、まだ1999年です。今はまだ2000年問題への憶測が盛んに飛び交っている段階ですので、本号が発売になったころにはどうなっているのか、個人的には興味が絶えません。

さて前置きはこのへんにしまして、今回はPostgreSQLの機能のひとつ、クライアント間通知機能（NOTIFY／LISTEN）を紹介します。これはその名のとおり、データベースに接続しているクライアント間で「通知」を行うことができる機能です。特にこれは、複数のプログラムが同じデータベースにアクセスするようなシステム（マルチクライアントのシステム）では素晴らしい効果を発揮します。

どのような場合に利用できるか？

たとえば、あるデータベースの内容を更新するプログラムがあり、それとは別に参照するプログラムもあるとします。そして、この参照するプログラムは、常に最新のデータベースの内容を表示する「モニタプログラム」のようなものだとします。具体的には、このようなシステムとして「在庫管理システム」などが思いつきます。在庫管理システムの場合、注文による出荷や仕入れによる入庫などで在庫量が増減するわけですが、過剰な在庫を避けるために微妙な仕入れ量の調整なども行う必要があるでしょう。

この在庫の流れをモニタするような在庫量監視プログラムを作成する際、あまり最新の在庫量を表示することにこだわると、データベースに頻繁にアクセスすることになってしまいます。それでも、在庫量の表示などのようにデータベース内のレコード数を数えるだけなら、サーバの負荷もたかが知れているでしょう。しかし、食料品の賞味期限までの平均残存日数や最小日数などを表示させようとしたら、サーバの負荷は無視できないほど高くなってしまいます。かといって、この負荷を軽減するためにデータベースへアクセスする頻度を減らすと、今度は最新の情報を表示することができなくなります。これでは急な在庫の減少などを見逃してしまい、補充のための発注が遅れるなどの影響が出るかもしれません。

この問題を解決するひとつの方針として、在庫量監視プログラムが在庫量の変化を瞬時に知ることができるようにすることが考えられます。つまり、在庫量を変化させた入出庫処理プログラムなどが、在庫量が変化したことを在庫量監視プログラムに「通知」してあげればよいわけです。そうすれば、在庫量監視プログラムは在庫量が変化したときにだけデータベースへアクセスすればよいわけですから、サーバの負荷は最小限に抑えられます。

ただし、「通知」といっても難しいことを考える必要はありません。PostgreSQLには、このような目的のために便利に利用できる機能があるからです。それが、今回紹介

する「クライアント間通知機能」です。それでは、この機能の使い方を具体的に紹介していきましょう。

クライアント間通知機能の使い方

PostgreSQLで、このクライアント間通知機能を利用するためには、NOTIFY、LISTEN、UNLISTENという3つのSQL文を利用します（**図1**）。

まず、通知を行うNOTIFYコマンドについて説明します。

NOTIFY 《通知名》

《通知名》には、通知の種類を区別するための任意の名称を指定します。この名称には通知する側と通知を受ける側で同じものを指定しなければ意味がありません。また、《通知名》にはテーブル名やカラム名などと同じ文法上の規則が適用されますので、大文字／小文字を区別したり、空白文字を含めたい場合には“"”（ダブルクォーテーション）で囲む必要があります。《通知名》に利用できる文字数は、最大で31文字までです。

このNOTIFYコマンドを実行するたびに、同じデータベースに接続して同じ通知名で受信を待っているすべてのクライアントに、通知メッセージが送られることになります。

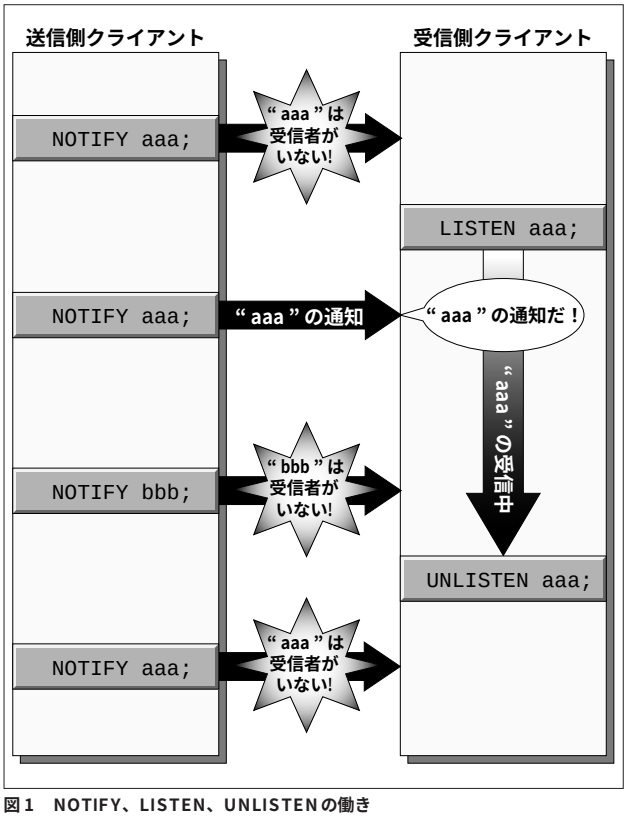


図1 NOTIFY、LISTEN、UNLISTENの働き

なお、デフォルトではどのクライアントも通知はいっさい受けつけない状態です。したがって、通知を受けたい場合には次のコマンドを実行して、そのことをPostgreSQLサーバに伝えます。

LISTEN 《通知名》

《通知名》には、受信したい通知の名称を指定します。NOTIFYの《通知名》と同様な文法上の規則が適用されます。

LISTENコマンド実行後は、同じデータベースに接続しているクライアントが同じ通知名でNOTIFYコマンドを実行するたびに、通知が受信されるようになります。また、同時に複数の名称の通知を受信したければ、受信したいすべての通知名について、それぞれLISTENコマンドを実行します。

通知の受信をやめるには、次のUNLISTENコマンドを実行します。

UNLISTEN 《通知名》

《通知名》には、受信をやめたい通知の名称を指定します。NOTIFYの《通知名》と同様な文法上の規則が適用されます。

注意事項

さて、さっそく実例の紹介といきたいところですが、その前にPostgreSQLのクライアント間通知機能を利用する際の注意事項についていくつか知っておく必要があります。

まず、自分で受信待ちをしている通知を自分が発信した場合は、自分自身にも通知が届きます。自分が発した通知は届かない、というような例外扱いは行われません。

また、トランザクション内でNOTIFYコマンドを実行した場合は、すぐにはほかのクライアントに通知は行われません。実際に通知が行われるタイミングはトランザクションが正常に終了した直後となります。あるいは、トランザクションが取り消されたときは、通知も取り消されてしまいます（つまり、通知されません）。通知は、ほかのクライアントにテーブルが更新されたことを伝えるのが一般的な目的です。だとしたら、これは十分に納得のできる動作でしょう。なお前述のとおり、トランザクション完了まで通知が遅れるため、長いトランザクションを多用してい

る場合には、どうしても通知機能のリアルタイム性は損なわれてしまいます。

そして最後の注意事項は、同じ種類の通知が短時間に連続して行われたときには、クライアントへは一度しか通知が行われない場合があるということです。つまり、通知の回数が重要となる目的には利用できないことを意味します。

psqlコマンドによる使用例

それでは実際に通知を送受信する様子を、PostgreSQLの対話型インターフェイスであるpsqlコマンドを利用して見てみましょう。

画面1は、psqlコマンドを起動して“ascii4”データベースに接続し、“notice1”という通知の受信を開始したところです。これで、同じ“ascii4”データベースに接続しているクライアントから“notice1”という通知が行われれば、このpsqlコマンド上にも受信されるはずです。まずは自分自身に通知を行ってみましょう。

画面2は、**画面1**で起動した状態になっているpsqlコマンドから、2種類のNOTIFYコマンドによる通知を行ってみたところです。

最初のNOTIFYコマンドでは、自分が待ち受けている“notice1”という名称の通知を行っています。その次の行の“NOTIFY”というメッセージは、NOTIFYコマンドが正常に実行されたことを表しています。そして、その直後には次のようなメッセージが表示されています。

```
ASYNC NOTIFY of 'notice1' from backend pid '5146' received
```

これは通知が届いたことを意味しており、その内容からは“notice1”という通知がプロセスID“5146”のPostgreSQLサーバから届いたことがわかります。

2番目のNOTIFYコマンドでは“notice2”という名称の通知を行っていますが、この通知は待ち受けていませんでしたので、そのあとには何も表示されていません。

それでは端末をもう1つ用意し、そちらでもpsqlコマンドを起動してマルチクライアントの環境を作ってみましょう。

画面3は、新しく起動したpsqlコマンドのほうから“notice1”の通知を行ったところです。そして**画面4**が、先に起動していたpsqlコマンドでこの通知を受信したところです。**画面4**ではなぜか無意味なSELECT文を実行していますが、これには理由があります。というのもpsqlコマンドでは、通知が到着したかどうかのチェックをSQL文が実行されたときにだけ行っているのです。ですから、**画面4**では無意味なSELECT文を実行させることにより、通知が到着しているかをpsqlコマンドに強制的にチェックさせているわけです。

もちろん、これはpsqlコマンド特有の制限で、実際のアプリケーション開発時にも同様な制限があるわけではあり

```
% psql ascii4
Welcome to the POSTGRESQL interactive sql monitor:
  Please read the file COPYRIGHT for copyright terms of
POSTGRESQL
[PostgreSQL 6.5.0 on i586-pc-linux-gnu, compiled by gcc
2.7.2.3]

   type \? for help on slash commands
   type \q to quit
   type \g or terminate with semicolon to execute query
 You are currently connected to the database: ascii4

ascii4=> LISTEN notice1;
LISTEN
```

画面1　“notice1”という通知を受信開始

```
ascii4=> NOTIFY notice1;
NOTIFY
ASYNC NOTIFY of 'notice1' from backend pid '5146'
received
ascii4=> NOTIFY notice2;
NOTIFY
```

画面2　自分で通知、自分で受信

```
% psql ascii4
        :
(psqlコマンドのオープニングメッセージ)
        :
ascii4=> NOTIFY notice1;
NOTIFY
```

画面3　異なるクライアントからの通知

```
ascii4=> SELECT 1;
?column?
--------
       1
(1 row)

ASYNC NOTIFY of 'notice1' from backend pid'5163'received
```

画面4　異なるクライアントからの通知を受信

ませんので、誤解のないようにしてください。

より便利な使い方

ここまではpsqlコマンドからの使用例を紹介しました。これらの例で、NOTIFYコマンドを使えば、待ち受けているクライアントに通知が送られることはわかっていただけたと思います。では次にもう少し実用的な例で、より賢いクライアント間通知機能の利用法を紹介しましょう。

前述のとおり、この通知機能はテーブルの更新があったことをほかのクライアントに伝えることを目的として利用することが多いと思います。つまり、「テーブルが更新されたら通知を行う」という利用パターンになるというわけです。

ところで、この「テーブルが更新されたら何かをする…」といえば、前回紹介した「トリガ」や「ルール」と目的が似ています。それなら、この通知機能をトリガやルールとうまく組み合わせることができそうに思えませんか？　実はそのとおりなのです。

ここで、ルールを使った通知の自動送信方法を紹介しましょう。なお、トリガを使った方法も考えられるのですが、PostgreSQLのトリガはテーブルが1行更新されるごとに起動されてしまいます。そのため、もし今回のような目的にトリガを利用すると、無駄な通知が頻繁に発生してしまいます（ひとまとめの更新に対しては通知が一度あれば十分ですから）。またどういうわけか、トリガプロシージャ内でNOTIFYコマンドを使用するとエラーになってしまいます。よって、今回は迷わずルールを利用することにします。

例によって、今回の題材も「顧客名簿」です。今回の例ではテーブル構成はあまり重要ではありませんが、復習の意味も兼ねて顧客テーブル（customer）の構成を**表1**に示しておきます。それでは、この顧客テーブルの更新時に自動的に通知を行わせるルールを作成してみましょう。

まず、通知の名称を決めます。顧客テーブル（customer）の更新を通知するわけですから、ここでは通知名を単純に“customer”とします。よって、通知を行うSQL文は次のようになります。

```
NOTIFY customer
```

次に、顧客テーブルを更新した際に自動的に上記SQL文を実行するルールを登録します。顧客テーブルの更新には“INSERT”、“UPDATE”、“DELETE”と3種類のイベントがありますので、それぞれのイベントに対してルールを登録します。SQL文は**リスト1**のようになります。

さあ、これで顧客テーブルが更新されると自動的に“customer”という名称の通知が発生するようになりました。さっそくpsqlコマンドでテストしてみましょう。読者がテストする際には、付属CD-ROMに含まれている「setup.sql」というファイルを利用してください。このファイルには、必要なテーブルやルールなどを作成するSQL文がすべて含まれています。詳しくは、同封のREADME.txtファイルを参照してください。

画面5がそのテストの様子です（setup.sqlは事前に実行しておきます）。

まず、更新通知を受けるためにLISTENコマンドを実行しています。その後、UPDATE文を用いて顧客テーブル（customer）を更新していますが、その直後に“ASYNC NOTIFY～”という通知が届いている様子がわかるでしょう。

このようにルールを利用すると、ほかのクライアントに

カラム名	データ型	説　明
ccode	integer	顧客コード
name1	text	姓
name2	text	名
addr1	text	住所1（都道府県）
addr2	text	住所2
addr3	text	住所3
zip	text	郵便番号
email	text	電子メールアドレス

表1　顧客テーブル（customer）の構成

▼リスト1　自動的に更新を通知するルールの登録

```
CREATE RULE customer_insert_rule AS ON INSERT TO
customer
    DO NOTIFY customer;
CREATE RULE customer_update_rule AS ON UPDATE TO
customer
    DO NOTIFY customer;
CREATE RULE customer_delete_rule AS ON DELETE TO
customer
    DO NOTIFY customer;
```

```
ascii4=> LISTEN customer;
LISTEN
ascii4=> update customer set ccode = ccode where ccode = 1;
UPDATE 1
ASYNC NOTIFY of 'customer' from backend pid '5163'
received
```

画面5　自動的に発生する更新通知

通知を送る作業をPostgreSQL内だけですませることができるので、アプリケーション作成時の作業量も削減でき、とても効率的です。

顧客名簿アプリケーションによる例

psqlコマンドなどによる簡単な利用例を紹介しました。さらに、ここでは実際にサンプルアプリケーションを作成してみましょう。このアプリケーションの機能は次のようなものです。

- **おなじみの「顧客名簿」アプリケーション**
- **顧客情報の表示や更新機能**
- **更新記録の自動保存機能**
- **更新記録の表示機能**

まず、顧客情報を保管するテーブルが必要です。これは、先ほどの「より便利な使い方」の項で出てきた「顧客テーブル（customer）」をそのまま利用することにします。さらに、更新記録を保存するテーブルや関連機能も必要です。こちらは、前回紹介した「更新記録テーブル（updatelog）」と「顧客テーブルの更新記録を自動取得するトリガ」をそのまま流用することにします。詳しい解説は前回の記事を参照してもらうこととして、更新記録テーブルの構成を**表2**に、更新記録を自動取得するトリガプロシージャを**リスト2**に、このトリガを登録するSQL文を**リスト3**に示して

カラム名	データ型	説　明
uuser	name	更新者
umethod	text	更新種別
udatetime	datetime	更新日時
old_ccode	integer	更新前の内容
old_name1	text	更新前の内容
old_name2	text	更新前の内容
old_addr1	text	更新前の内容
old_addr2	text	更新前の内容
old_addr3	text	更新前の内容
old_zip	text	更新前の内容
old_email	text	更新前の内容
new_*		更新後の内容

表2 更新記録テーブル（updatelog）の構成

▼リスト3　更新記録トリガの登録SQL文

```
CREATE TRIGGER customer_update_tgr
AFTER INSERT OR UPDATE OR DELETE ON customer
FOR EACH ROW
EXECUTE PROCEDURE customer_update();
```

▼リスト2　更新記録を自動取得するトリガプロシージャ

```
CREATE FUNCTION customer_update() RETURNS OPAQUE
AS '
BEGIN
    IF TG_OP = ''INSERT'' THEN
        INSERT INTO updatelog (
            uuser, umethod, udatetime,
            new_ccode, new_name1, new_name2,
            new_addr1, new_addr2, new_addr3, new_zip,
            new_email
        ) VALUES (
            USER, ''INSERT'', ''now'',
            new.ccode, new.name1, new.name2,
            new.addr1, new.addr2, new.addr3, new.zip,
            new.email
        );
        RETURN new;
    ELSE
        IF TG_OP = ''UPDATE'' THEN
            INSERT INTO updatelog (
                uuser, umethod, udatetime,
                old_ccode, old_name1, old_name2,
                old_addr1, old_addr2, old_addr3, old_zip,
                old_email,
                new_ccode, new_name1, new_name2,
                new_addr1, new_addr2, new_addr3, new_zip,
                new_email
            ) VALUES (
                USER, ''UPDATE'', ''now'',
                old.ccode, old.name1, old.name2,
                old.addr1, old.addr2, old.addr3, old.zip,
                old.email,
                new.ccode, new.name1, new.name2,
                new.addr1, new.addr2, new.addr3, new.zip,
                new.email
            );
            RETURN new;
        ELSE
            INSERT INTO updatelog (
                uuser, umethod, udatetime,
                old_ccode, old_name1, old_name2,
                old_addr1, old_addr2, old_addr3, old_zip,
                old_email
            ) VALUES (
                USER, ''DELETE'', ''now'',
                old.ccode, old.name1, old.name2,
                old.addr1, old.addr2, old.addr3, old.zip,
                old.email
            );
            RETURN old;
        END IF;
    END IF;
END;
'
LANGUAGE 'plpgsql';
```

▼リスト4　顧客管理プログラム（抜粋）

```
# 顧客名簿アプリケーション 顧客管理プログラム

#### プロシージャ宣言
## プロシージャ：アプリケーションの終了
proc exitApp {} {
        :
    pg_disconnect $dbHandle
        :
}
## プロシージャ：顧客一覧の読み込み
proc loadList {} {
        :
    # 顧客テーブルからレコードを取得
    set sql "select ... from customer ...
    pg_select $dbHandle $sql record {
        :
    }
        :
}
## プロシージャ：顧客情報の読み込み
proc loadData {} {
        :
    # 顧客テーブルからレコードを取得
    set sql "select * from customer where ...
    pg_select $dbHandle $sql record {
        :
    }
        :
}
## プロシージャ：顧客情報の保存
proc saveData {} {
        :
    # 新規レコードの挿入
    set sql "insert into customer ...
    set resultHandle [pg_exec $dbHandle $sql]
    pg_result $resultHandle -clear
        :
    # 既存レコードの更新
    set sql "update customer ...
    set resultHandle [pg_exec $dbHandle $sql]
    pg_result $resultHandle -clear
        :
}
## プロシージャ：顧客情報の削除
proc deleteData {} {
        :
    # 既存レコードの削除
    set sql "delete from customer ...
    set resultHandle [pg_exec $dbHandle $sql]
    pg_result $resultHandle -clear
        :
}

#### 処理開始
## データベースへ接続
set dbHandle [pg_connect ascii4]
## メインウィンドウの作成
        :
```

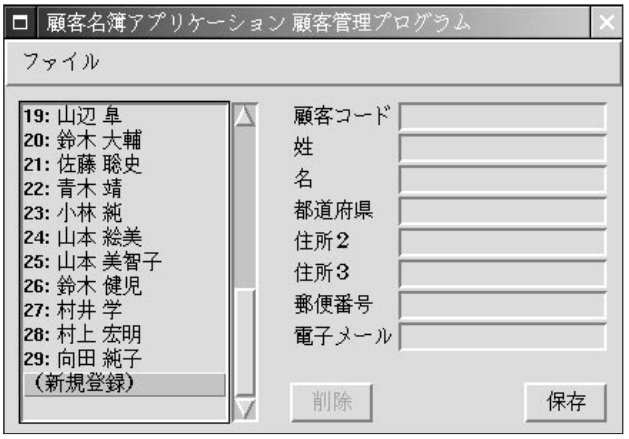
おきます。

これらのテーブル定義や各種SQL文なども、すべて今月号の付属CD-ROMに収録されています。読者がこのサンプルを実際に試すときは、同封のREADME.txtファイルに従って「setup.sql」というファイルを実行するだけで、データベース上での準備がすべて整うように用意してあります。なお、前述の「より便利な使い方」の項での解説に合わせて、すでにsetup.sqlを実行している場合は、すでにトリガなども含めたデータベースの準備はすべて整っているはずです。

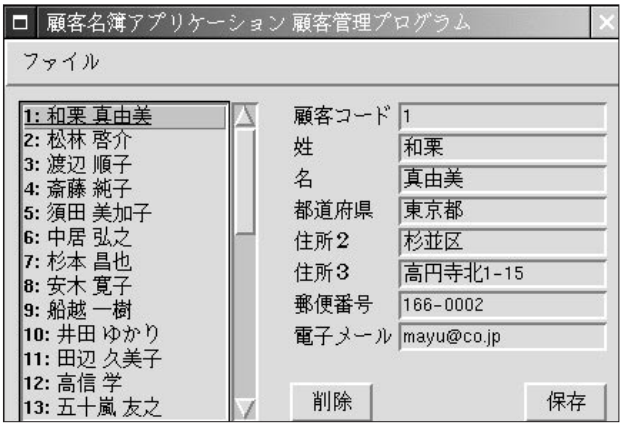
さて、データベースの準備が整ったところで、次にアプリケーション本体の説明をしておきましょう。ただし、ここで本格的なアプリケーションを作成するのでは時間も紙幅も足りませんので、今回はXアプリケーションを簡単に作成できる「Tcl/Tk」というスクリプト言語を利用して、手早く（かなり手抜きをして）作成したものを用意しました。とはいっても、十分参考にはなるはずです。なお、アプリケーションのプログラムリストをすべて掲載していてはこれもまた紙幅が尽きてしまいますので、ここからはデータベースに関連する部分だけを抜粋して紹介していきま

▼リスト5　更新記録表示プログラム（抜粋）

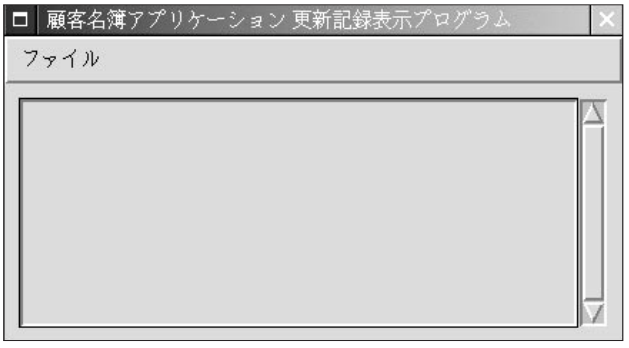
```
# 顧客名簿アプリケーション 更新記録表示プログラム

#### プロシージャ宣言
## プロシージャ：アプリケーションの終了
proc exitApp {} {
        :
    pg_disconnect $dbHandle
        :
}
## プロシージャ：更新記録一覧の読み込み
proc loadList {} {
        :
    # 更新記録テーブルからレコードを取得
    set sql "select ... from updatelog ...
    pg_select $dbHandle $sql record {
        :
    }
        :
}

#### 処理開始
## データベースへ接続
set dbHandle [pg_connect ascii4]
## メインウィンドウの作成
        :
## 更新通知の受信準備
pg_listen $dbHandle customer "loadAll"
        :
```



画面6　顧客管理プログラム



画面8　顧客情報の変更



画面7　更新記録表示プログラム

画面9　更新内容の表示

す。もちろん、付属CD-ROMにはすべてのリストが含まれています。

まずアプリケーションの構成です。機能を大きく分けると「顧客情報の表示や更新」と「更新記録の表示」の2つになりますので、今回はこの2つの機能をそれぞれ単独のプログラムとしました。というのも、今回の題材である「クライアント間通知機能」を利用することにより、まったく別個である2つのプログラムが連携動作する様子をお見せしたいからなのです。

リスト4は、顧客情報の表示や更新を行う「顧客管理プログラム」のプログラムリストからの抜粋です。この顧客管理プログラムでは、データベースに関する処理は以下の4種類しか行っていません（項目はリスト順）。

- **データベースを切断（pg_disconnect）**
- **SELECT文による顧客テーブルの読み込み（pg_select）**
- **INSERT、UPDATE、DELETE文による顧客テーブルの更新（pg_exec）**
- **データベースに接続（pg_connect）**

顧客管理プログラム内では、クライアント間通知機能の明示的な呼び出しは行っていません。これは、あらかじめデータベースに登録してあるルールが、顧客テーブルの更新時に自動的な通知を行ってくれるため、アプリケーション側で特別な処理を行う必要がないからです。

画面6は、このプログラムの起動直後の画面です（このプログラムの起動方法は付属CD-ROMのREADME.txtを参照してください）。

次に、「更新記録表示プログラム」を紹介しましょう。**リスト5**に示したのが、この「更新記録表示プログラム」のプログラムリストからの抜粋です。「更新記録表示プログラム」が行っているデータベースに関する処理は、以下の4種類です（項目はリスト順）。

- **データベースを切断（pg_disconnect）**
- **SELECT文による更新記録テーブルの読み込み（pg_select）**
- **データベースに接続（pg_connect）**
- **更新通知の受信準備（pg_listen）**

更新記録表示プログラムが顧客管理プログラムと異なる

もっとも大きな点は、（リストの最後のあたりで）顧客テーブルの更新通知を受信するための準備を行っているというところです。これにより「更新記録表示プログラム」は、誰かが顧客テーブルを更新したことをリアルタイムに知ることができるようになります。そして、そのタイミングでデータベースにアクセスすることにより、最低限のサーバ負荷で常に最新の更新記録を表示することができるようになります。

画面7が、「更新記録表示プログラム」の起動直後の画面です。最初は更新記録がありませんので、何も表示されず非常に素っ気ない画面となっています。

さあ、それでは先ほどの「顧客管理プログラム」と「更新記録表示プログラム」の両方とも起動した状態で、「顧客管理プログラム」から顧客テーブルを更新してみましょう。まず、「顧客管理プログラム」の画面の左側、顧客一覧リストから、1番目の顧客「和栗 真由美」さんをマウスで選択します。すると、右側に詳細情報が表示されます（**画面8**）。ここで名前の「真由美」を「由美」に変更して［保存］ボタンをクリックしてみます。これで顧客テーブルは更新されました。この直後、あらかじめ起動しておいた「更新記録表示プログラム」の画面はどうなっているでしょうか？　自動的に表示が更新されて、今行った更新の内容が表示されています（**画面9**）。

2つのプログラムはまったく別のプログラムとして作成したにも関わらず、「顧客管理プログラム」が行ったデータベースの更新に対し、「更新記録表示プログラム」が瞬時に反応する様子がおわかりいただけたと思います。

PostgreSQLの高い実力

さて、紙幅も尽きてきました。PostgreSQLのクライアント間通知機能は、いかがでしたでしょうか？　いろいろな使い道がありそうな気がしませんか？

たとえば、最近ではWebとデータベースの連携が当たり前のように利用されています。しかし、Webサイトへの訪問者がデータベースにアクセスしている状況などは、管理者からはなかなか手に取るようにはわからないものです。それでも最新情報の取得が必要なシステムの場合は、従来なら自動的に管理者へメールを発信するような仕組みをシステムに組み込んでいたことだと思います。

しかし、PostgreSQLではそんなことをしなくても、クライアント間通知機能を利用すれば、管理用アプリケーションに常に最新状況が表示されるようにすることもできるのです。

PostgreSQLは、誰もが自由に利用できるフリーなソフトウェアであるにも関わらず、このようにさまざまな可能性を秘めているリレーショナルデータベース管理システムなのです。使い込めば使い込むほどに、その能力は予想以上だと気づいてもらえるに違いないと確信しています。

次回は、これまでとちょっと趣向を変えて、PostgreSQLの運用に関する話など紹介したいと思っています。

Column

PostgreSQLの誤解？

PostgreSQLはよく「遅い」といわれます。雑誌などのデータベースの比較記事などでも、PostgreSQLは他のデータベースに大差をつけられて、最下位の座を与えられていることがほとんどです。この「遅いデータベース」というのが、PostgreSQLの本当の実力なのでしょうか？　私はそうは思っていません。次に述べるような工夫を施すことでチューニングすることができるからです。

PostgreSQLの場合、VACUUMを実行しなければ、よいパフォーマンスは得られません（コマンド名はvacuumdb）。利用するデータにもよりますが、VACUUMを実行するだけで他のデータベースにつけられていた大差が小さくなることがあります。

さらにPostgreSQLは、初期設定では「安全重視」の設定になっています。たとえば、トランザクションが完了するたびに、更新されたデータを物理的にハードディスクへ書き込みます（fsync機能）。これは、万一OSレベルでシステムが停止したときに、データベースの被害を最小限に食い止めるためです。しかし、実際このような状況が起こるのは、安定バージョンのPC-UNIXなら停電時くらいしかありません。したがって、パフォーマンスを重視するなら、このfsync機能を解除することもパフォーマンスの向上につながります（具体的には、起動オプションに -o '-F' を追加してPostgreSQLを起動する）。これは、特に更新処理においては劇的なパフォーマンスアップが期待できます。たとえ、このような運用をしていたとしても、UPSなどと併用すればさほど危険ではないでしょう。

最後に、PostgreSQLは明示的にトランザクションの開始（BEGIN TRANSACTION）を指定しない限り、1つのSQL文ごとにトランザクションをコミットします。更新処理が続くような場合、これはパフォーマンス低下の原因となります。したがって、複数のSQL文を1つのトランザクション内で実行するようにすれば、これもまた劇的なパフォーマンスアップにつながります。

正しい使い方、よりよい使い方を知り、読者の方々もPostgreSQLの性能をギリギリまで引き出してみてください。

Rubyで行こう

いかがですか？ そろそろRubyに慣れてきましたか？ 分厚いRuby本も少しは進みましたか？ 今回はRubyのもっとも得意な分野であるテキスト処理のうち、難解と思われている「正規表現」について解説しましょう。

第3回　スキャナーズ

文：赤松智也
Text : Tomoya Akamatsu

コンピュータと人間では得意分野が異なります。コンピュータは、人間のように自由で柔軟な発想をすることはできませんが、単純な繰り返しを間違えずに超高速で行うことができます。

たとえば、「Rubyのリファレンスマニュアルのうち、Perlという単語を含むファイルがいくつあるか」という質問をしたとします。人間に数えさせると、なかなか正確な答は得られないでしょう。しかし、コンピュータなら次の方法で簡単にわかります。

```
% grep -l Perl *.html
```

これで目的は果たせましたが、これではこの連載の目的が果たせません（笑）。この連載の目的!!　そう、Rubyでやるなら次のようなプログラムになります。

```
ARGV.each{|a|
  n = 0
  open(a){|f|
    f.each{|line| n+=1 if /Perl/ =~ line}
  }
  print a, "\n" if n > 0
}
```

このコマンドを、つい先日リリースされたRuby 1.4.3のリファレンスマニュアル（日本語版）に実行してみると、次のファイルに「Perl」という単語が含まれていることがわかります。

```
function.html
glossary.html
options.html
preface.html
socket.html
syntax.html
variable.html
```

このように、コンピュータはスキャンするのが得意です。昔から餅は餅屋というように、コンピュータが得意なことはコンピュータにやらせるべきです。そして、コンピュータが得意な仕事をうまくコンピュータに押しつける道具が、Rubyのようなプログラミング言語なのです。

正規表現

ここまでの例は「Perl」のような単純な文字列を探すだけでしたが、実際にはある「パターン」を使って文字列を指定したい場合があります。具体的には、「Pで始まってlで終わる」といった表現です。このように「パターンを表現する指定方法」が、今回紹介する「正規表現（Regular

Expression)」です。Rubyでは、正規表現は2つの“ / ”（スラッシュ）にはさまれて表現されています。

```
/FOO/    # FOOという文字列
/P.*l/   #「P」で始まり「l」で終わる文字列
```

正規表現は、このような「パターンを表現するミニ言語」です。いわば、Rubyに埋め込まれた別の言語だと考えることができます。

UNIXでは、正規表現はいろいろなツールで使われています。Rubyはもちろん、perl、awk、sed、ed、vi、emacs、lessなどがそうです。ただし、これら個々のツールで独立に正規表現が実装されているため、ツールによって少しずつ文法が異なっています（いわゆる「方言」があるのです）。ここでは、Rubyで使われる正規表現について説明しますが、基本は同じですから他のツールでも応用は利くでしょう。ちなみに、Rubyの正規表現度は、一言で表現すると「Perl5.004とほぼ同じ」ということになります。

メタ文字	説　明
.	任意の一文字とマッチ
[]	[a-z]はaからzまでのいずれか
[^]	[^a-z]はaからz以外の文字
\w	単語を構成する文字
\W	単語を構成する文字以外
\s	空白文字。[\t\n\r\f]と同じ
\S	非空白文字
\d	数字。[0-9] と同じ
\D	非数字。[^0-9] と同じ
*	0回以上の繰り返し
+	1回以上の繰り返し
?	0または1回の繰り返し
{m,n}	m回からn回の繰り返し
*?	0回以上の繰り返し(非欲張り型)
+?	1回以上の繰り返し(非欲張り型)
??	0または1回の繰り返し(非欲張り型)
{m,n}?	m回からn回の繰り返し(非欲張り型)
^	行頭にマッチ
$	行末にマッチ
\A	文字列の先頭にマッチ
\Z	文字列の末尾(改行があればその直前)にマッチ
\z	文字列の末尾にマッチ
\b	バックスペース(0x08)([]内)
\b	語境界文字([]外)
\B	非語境界文字
|	選択
( )	表現のグループ化(後方参照あり)
\1,\2	後方参照(n番目の括弧に対応する)
(?:)	表現のグループ化(後方参照なし)
(?=)	パターンによる位置指定(幅を持たない)
(?!)	パターンの否定による位置指定(幅を持たない)
(?#)	コメント

表1　正規表現のメタ文字

正規表現の文法

正規表現は、「メタ文字」と呼ばれる特殊機能を持つ文字と、それ以外の文字から構成されます。メタ文字は、プログラミング言語における制御構造のようなもので、正規表現で利用されるメタ文字には**表1**に示したようなものがあります。

メタ文字以外の文字は、その文字そのものと一致します。ですから、“ /FOO/ ”は“ FOO ”という文字の並びと一致します。

正規表現の文法の詳細は『Ruby本』にお任せすることにして、ここでは実際の正規表現の例を見ながら、その意味を解釈してみます。

- **FOO**

  「FOO」。つまり「F」「O」「O」という文字の並び。

- **P.*l**

  「P」に続く任意の文字の0回以上の繰り返しのあと、「l」が続く。「Pl」にもマッチする。

- **[A-Z]\w***

  Rubyの定数の識別子のパターン。

- **(Ruby)+**

  「Ruby」の1回以上の繰り返し、つまり、「Ruby」や「RubyRuby」や「RubyRubyRuby」...。

- **Dec,?**

  「Dec」のあと「,」の0または1回の繰り返し。つまり、「Dec」または「Dec,」。

- **De(c,)?**

  「De」または「Dec,」。複数の文字を繰り返すにはカッコでまとめることが必要。

- **Sun|Mon|Tue|Wed|Thu|Fri|Sat**

  （英語の）曜日にマッチ。選択“ | ”（パイプ）は結合強度が弱いので、直前の文字でなく、文字列が対象。

正規表現の落とし穴

正規表現によるパターンの表現方法を、多少はわかってもらえたでしょうか？　Rubyに限らず、PerlでもAWKでも正規表現は難解だとよくいわれます。それには、それなりのワケがあります。すでに述べた「言語の中のミニ言語」というのも理由のひとつです。そして、それ以外にも次のような理由が考えられます。

■記号が多く密度が高い表現

正規表現は記号によってパターンを表現するため、全体に記号が多く含まれ、文字がぎっしりつまった「ノイズ」のような外見になりがちです。Perlといっしょですね。また、文字にすべて意味があるので、読みやすい表現にするという選択肢が提供されないのも問題です。もっとも、これについてはあとで述べる「拡張正規表現」で対応できます。

■0回以上の繰り返し

私が正規表現を書いていて一番ひっかかるのはこれです。AWKの頃からずいぶん正規表現を書いてきたのですが、今でも間違えます。たとえば、「“a”の0回以上の繰り返しに続く“bb”」というパターンを表す

```
a*bb
```

という正規表現を

```
ccbbaabb
```

という文字列にマッチさせると、どの部分にマッチすると思いますか？

```
ccbbaabb
```

だと思うでしょう？ でも、実は答えは

```
ccbbaabb
```

です。つまり、「0個のaに続くbb」は単なる「bb」だからです。正規表現のマッチは、文字列を左側からスキャンして、最初にマッチした時点でスキャンをやめてしまいます。ですから、その先にもっとふさわしい（ように人間に感じられる）ものがあっても、そこまで探しに行きません。「a」が1個以上あることが保証されていれば、

```
a+bb
```

というパターンにすればすむことですけど、なかなかそうはいかないんですよねえ。

■欲張りマッチ

もうひとつひっかかりやすいのは「正規表現のスキャンは欲張りだ」という点です。これも例を挙げましょう。次は「任意の文字列の並びに“:”（コロン）が続く」というパターンです。

```
.*:
```

この正規表現を

```
http://www.ruby-lang.org:80/ja/index.html
```

にマッチさせると

```
http://www.ruby-lang.org:80/ja/index.html
```

にマッチしそうですよね。でも、実は

```
http://www.ruby-lang.org:80/ja/index.html
```

なんです。つまり、正規表現マッチは基本的に「欲張り」なので途中で“:”を見つけても、そこでスキャンをやめずに、もっとも長い文字列に対してマッチしようとします（難しい言葉では「最左最長一致」といいます）。ですから、“:”を探してどんどん先の方まで進んで、一度文字列の終端まで進んでから、「あ、なかった」と思って、今まで進んできた文字列を引き返すのです（ちなみに、この引き返すことを「バックトラック」と呼びます）。そのため、一番後ろの“:”までマッチすることになるのです。この問題を回避する方法は2つあります。ひとつはパターンを

```
[^:]*:
```

のようにすることです。これは「“:”以外の文字の並びに

“:”が続く」という意味です。これで「http:」の部分にマッチします。もうひとつは、Ruby（とPerl）の正規表現でだけ有効な方法で、

```
.*?:
```

というパターンです。この“*?”というのは「非欲張りマッチ」と呼んで、すでに説明した欲張りなスキャンを行いません（「最左最短一致」とも呼びます）。非欲張りマッチでは、スキャンしていく途中で“:”を見つけると、そこで満足してそれ以上探しに行きません。これで、めでたく「http:」の部分が得られるわけです。

正規表現オブジェクト

純粋なオブジェクト指向言語であるRubyでは、すべてのデータはオブジェクトです。ですから、正規表現もオブジェクトです。Rubyのプログラム中では正規表現オブジェクトは以下のように書きます。ちなみに、「RE」は正規表現（Regular Expression）のことを表しています。

```
/RE/
```

しかし、これではパスのような“/”（スラッシュ）を多く含むパターンを書こうと思うと、毎回、

```
/\/usr(\/local)?\/bin/
```

のように、\（バックスラッシュ）でエスケープしなければならず、かなり繁雑になってしまいます。しかし、柔軟な記述が得意なRubyでは、このような場合のために、

```
%r!/usr(/local)?/bin!
```

オプション	説　明
i	大文字小文字を無視(Ignore case)
o	式展開は最初の1度だけ(Once)
x	拡張正規表現(eXtension)
p	改行も.にマッチ(Posix)
s	シフトJIS
e	EUC
u	UTF-8
n	マルチバイト文字を解釈しない(None)

表2　正規表現のオプション

という表現も用意しています。つまり、“%r”を使って表現すれば“!”の部分には任意の文字が使えるのです。この区切り文字がカッコのときは、対応するカッコが使われます。たとえば、

```
%r{/usr(/local)?/bin}
```

のようになります。

正規表現のオプション

Rubyの正規表現の後ろには、その正規表現オブジェクトの性質を決めるオプションを指定することができます。

```
/RE/i
```

のように書きます。このオプションの意味は**表2**のとおりです。

Rubyでは正規表現ごとに対応するマルチバイト文字のエンコーディングを選ぶことができ、それが大きな特長になっています。標準でマルチバイト文字（日本語）を扱えるのは国産言語のメリットですね。

コメントとインデント

正規表現にオプション“x”（エックス）を指定すると「拡張正規表現」を使えるようになります。拡張正規表現を指定すると、正規表現のパターンの中で、空白を無視して、コメントを置けるようになるので、正規表現を読みやすく書くことができます。たとえば、

```
/\w\w\w+,?\s*(\w\w\w+)\s*(\d\d?)\s*[\d:]+\s*(\d{4})/
```

というパターン（これは日付にマッチするパターンです）を書くのと、

```
/\w\w\w+,?\s*          # 曜日
 (\w\w\w+)\s*          # 月
 (\d\d?)\s*            # 日
 [\d:]+\s*             # 時刻
 (\d{4})               # 年
/x
```

と書くのとでは、コメントやインデントがあるほうが読みやすいでしょう。もっとも、違いはわずかのようにも思えますが…。

使ってみよう、正規表現

正規表現がどのようにパターンを表現するかわかったところで、実際に使ってみましょう。Rubyで正規表現を利用する局面はいろいろありますが、もっとも重要と思われるものを3つだけ紹介します。

■マッチ演算子（=~、!~）

マッチ演算子は、「正規表現マッチを行うための演算子」です（厳密には“=~”はメソッドであるなどいろいろあるのですが、ここではそれで十分です）。

正規表現 =~ 文字列

も、

文字列 =~ 正規表現

も有効で、マッチが成功したときには何バイト目でマッチしたかという整数値を、失敗したときにはnil（偽）を返します。正規表現を前に置くのが、Ruby的だといわれています。

文字列 =~ 文字列

という記述も（お勧めではありませんが）可能です。この場合には**後ろの文字列がパターンになる**ことに注意してください。マッチ演算子は、前回のgrepプログラムでも使われていましたね。

```
while line = gets()
  print line if /pattern/ =~ line
end
```

■sub／gsub

sub／gsubは、「置換を行うメソッド」です。subは先頭のマッチだけを置換し、gsubはマッチしたすべての場所を置換します。

```
"abcabc".sub(/a/, 'A')   # => "Abcabc"
"abcabc".gsub(/a/, 'A')  # => "AbcAbc"
```

sub／gsubは、文字列に置換を行ったコピーを返しますが、sub!／gsub!は、文字列そのものを書き換えます（前回少し説明した「破壊的メソッド」というやつです）。

```
a = "abcabc"
a.sub(/a/, 'A')          # => "Abcabc"
p a                      # => "abcabc"
a.sub!(/a/, 'A')         # => "Abcabc"
p a                      # => "Abcabc"
```

このsub!を使った例を挙げておきます。

Column

Rubyの正規表現の実装

Rubyの正規表現の実装は、GNU AWK（gawk）に用いられていた正規表現ルーチンに改造を加えたものです。この正規表現は、元々GNU Emacs用に開発されたものです。

RubyではGNUの正規表現ルーチンに対して、次の改造が行われています。

- t^2(Takahiro Tanimoto)さんによるシフトJISおよびEUC対応
- 吉田正人さんによるUTF-8対応
- まつもとさんによるPerl5正規表現対応

結果として、ほとんど原型をとどめない程度まで改造されています。このRubyの正規表現ルーチンは、UTF-8を含むマルチバイト文字に対応し、Perl5の正規表現を理解するという、世界でも珍しいものとなっています。 Perl5の正規表現を使いたいなら、始めからPerlの正規表現ルーチンをもらってくれば、と考えるのが自然でしょう。そうすれば、Perlでどんどん追加される新機能に苦労して対応する必要もないですし、日本語化についてもjperlというプロジェクトがあるので問題なさそうです。

しかし、Perlの正規表現ルーチンはPerlのインタプリタと分離していないので、それだけを使うというわけにはいかなかったようです。使おうと思うとPerlのインタプリタをまるごとリンクすることになるそうで、それではちょっと使えません。逆にRubyの正規表現ルーチンは、Rubyとは独立していますから、全文検索ツール「Namazu」に組み込まれたりしています。

なお、この正規表現ルーチンはまつもとさんがゼロから書いたものではないので、時々まつもとさんにも直すのが難しいバグが発見されることがあります。そういうときには「正規表現（ルーチン）は鬼門だ」とか「ゼロから書き直そうかなぁ」などと、愚痴が聞こえてきます。(笑)

```
wc = 0
while line = gets()
  while line.sub!(/\w+/, '')
    wc += 1
  end
end
print wc, "\n"
```

このプログラムはファイル中の単語の数を出力します。

■scan

scanは、「文字列の先頭から正規表現にマッチする部分を切り出してくるメソッド」です。正規表現がカッコを含まないときはマッチした文字列を、カッコを含むときにはカッコに対応する文字列の配列を次々に渡します。

たとえば、以下のようなファイルから品名と値段を切り出すプログラムを考えます。

```
トマト:100円 キュウリ:80円 ネギ:110円
ミカン:298円 イチゴ:320円
トーフ:180円
```

scanを使うと以下のようになります。

```
while line = gets()
    line.scan(/([^\s]*?):(\d+)円/) do |item, price|
      printf "品名: %s 価格: %s円\n", item, price
    end
end
```

このプログラムを、上の例に実行してみると、

```
品名: トマト 価格: 100円
品名: キュウリ 価格: 80円
品名: ネギ 価格: 110円
品名: ミカン 価格: 298円
品名: イチゴ 価格: 320円
品名: トーフ 価格: 180円
```

となります。

また、subのところで説明した単語の数を数えるプログラムをscanを使って書くと、以下のようになります。

```
wc = 0
while line = gets()
  line.scan(/\w+/) do
    wc += 1
  end
end
print wc, "\n"
```

さらに知りたいなら

今回は、スキャンの名手「正規表現」について紹介しました。年末進行の関係で（笑）、あまり具体例を出せなかったのですが、一度正規表現の気持ちになってどのようにマッチを行うのか1文字ずつ考えてみると面白いかもしれません。「あ、失敗した。やりなおし」とか考えていると、どのような正規表現を書けばよいのかわかってくると思います。日常茶飯事になってしまうと、ちょっとアブナイかもしれませんが…。

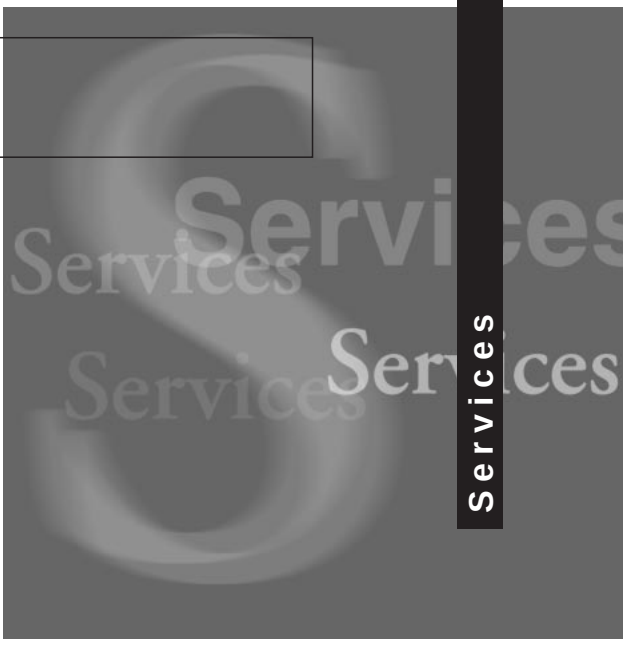
もし、正規表現についてもっと知りたいと思ったら、オライリーの『詳説 正規表現』を読むことをお勧めします。「正規表現とはこんなに奥の深いものであったか」と驚くことでしょう。

『詳説 正規表現』
Jeffrey E. F. Friedl 著
歌代 和正 監訳
春遍 雀来、鈴木 武生 共訳
オライリー・ジャパン発行
4300円
ISBN4-900900-45-1

まさか正規表現だけで1冊作ってしまうとはだれも思わなかっただろう。しかし、「どこでマッチングするか」ということだけを紹介した退屈なものではなく、2つの正規表現エンジン（NFAとDFA）による処理メカニズムの違いや、バックトラックの詳細な検証など、正規表現の利用法というよりも、仕組みの解説に焦点が当てられており読み物としても面白い。まさにオライリー本の面目躍如といった仕上がりである。Rubyの正規表現とほぼ同じ能力を持つ、Perlの正規表現について130ページ以上にもわたって解説されているので、Rubyユーザーにも十分役に立つはずだ。（編集部）

Sambaと闘う（第4回）

今回取り組むのは環境設定だ。Sambaを細かくチューニングして、トラブルを解決したり、より使いやすくしたりしていこう。



環境設定と日本語ファイル名の扱い

文：梅原 系
Text : Kei Umehara

前回の作業で、最新のSamba環境を手に入れることができた。運がよければ、しばらくはこのバージョンを使い続けることができるはずだ。SWATのインストールも行い、気軽にSambaの設定を変えられるようにもなった。ようやくSambaを本格的に使いこなす準備ができたということだ。

SWAT操作の基礎

前回インストールしたWebベースのSamba設定ツール、SWATだが、使い方についてはまったく説明していなかった。実際に細かな環境設定に取り組む前に、基本的な使い方を説明しておこう（SWATの使い方を間違えていては、環境設定もできないからね）。

■ページの切り替え

SWATのウィンドウの上のほうには、**画面1**のようなアイコンやボタンが並んでいる。これは、SWATのさまざまな設定ページを切り替えるためのものだ。作業に応じて、目的のアイコンをクリックすれば、そのページに切り替わるようになっている。

それぞれのアイコンボタンの概要は**画面1**にまとめておくが、これらのなかで特に重要なのは、Sambaの基本設定を行う【GLOBALS】と、共有リソースの管理を行う【SHARES】／【PRINTERS】の3つだ。

■コマンドボタン

各設定ページには、いくつかのボタンが並んでいる。ページごとにボタンは異なっているが、特に大事なボタンについて、ここで説明しておこう。これ以外のボタンについては、おいおい説明していく。

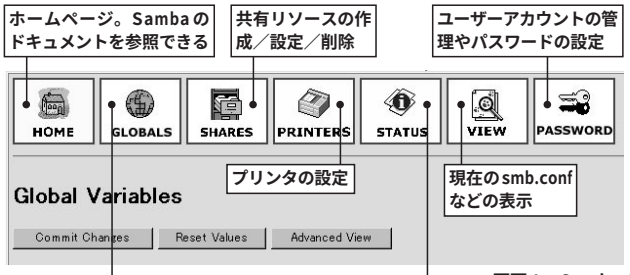
- **【Advanced View】**

SWATでは、基本ページ（アイコンのクリック直後に表示されるページ）の構成をすっきりとさせるために、デフォルトでは頻繁に変更する項目だけを表示するようになっている。Sambaの細かな設定を変更したい場合には、このボタンをクリックして、すべての項目を表示させるようにしよう。今回扱う、日本語ファイル名の設定もデフォルトでは表示されないので、最初にこのボタンをクリックして、関連パラメータを表示させる必要がある。

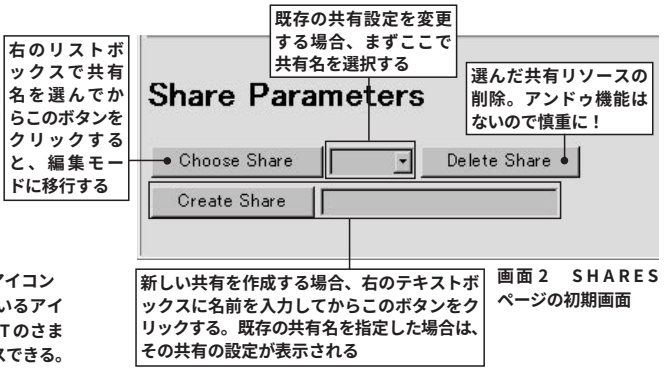
- **【Commit Changes】**

管理者が行った変更を、smb.confに反映させるボタンだ。設定変更を行ったあとは、忘れずにこのボタンを押さないと、せっかくの作業内容が失われてしまうので注意してほしい。

ただし、このボタンをクリックしても、基本的にはsmb.confに書き込まれるだけで、現在実行中のSambaの設定が変更されるわけではない。つまり、テキストエディタでsmb.confを編集した場合と同じように、Sambaの再



画面1　Sambaのアイコンページ上部に並んでいるアイコンを使って、SWATのさまざまな機能にアクセスできる。



画面2　SHARESページの初期画面

起動が必要になる。これは、ページ上部の【STATUS】アイコンをクリックし、【Restart smbd】ボタンをクリックすれば、新しい設定でSambaが再起動するようになっている。

万が一、編集中に混乱したりして、作業を最初からやり直したい場合には、隣の【Reset Value】ボタンをクリックすればよい。

- **【Set Default】**

各エントリに用意されているボタンで、これをクリックすると、現在使用しているSambaのデフォルト設定がテキストボックスに入力される。ページ全体を編集開始時の状態に戻すのであれば、そのページの上のほうに表示されている、【Reset Value】ボタンをクリックすればよい。

■SWATが作成するsmb.confについて

前回も簡単に触れたことだが、SWATを使うとsmb.confが大幅に変更される。人によっては、せっかく作り上げたsmb.confがまるで別物になってしまって、がっかりする羽目に陥るかもしれない。SWATを起動する前に、smb.confのバックアップは必ず行っておこう。

SWATは、起動時にsmb.confの内容をすべて読み込む。そして、【Commit Changes】ボタンがクリックされた時点で、その内容をsmb.confに書き出す。しかし、実際に書き出すのは、「Sambaのデフォルト設定とは違うパラメータだけ」だ。

このような仕組みになっているため、smb.confに記述したコメントや、デフォルトと同じパラメータ指定などはすべて削除されてしまう。もちろん、smb.confでinclude文を使っている場合なども、その指定はなくなってしまう。一時的に共有エントリを無効にしたくて、その指定をコメントアウトしていたりすると、SWATで編集した際に記述がきれいさっぱり消えてしまうことになる。特にコメントアウトを使っている場合には注意してほしい。ちなみに、共有エントリを一時的に無効化するのに、そのエントリに“available=No”パラメータを指定していれば、SWATがエントリを削除することは避けられる。

コメントアウト以外にもうひとつ注意してほしいのは、「デフォルトから変更されているパラメータのみが書き出される」という仕様だ。たとえば、[global]セクションで“workgroup=WORKGROUP”と指定すると、たいていの場合、そのパラメータはsmb.confに出力されなくなる。なぜなら、たいていのSambaでは、workgroupパラメータのデフォルトが“WORKGROUP”だからだ。つまり、人間に読みやすいようにと記述したパラメータなどが、片端から削除される可能性があるのだ。

また、複数のバージョンのSamba間でsmb.confをコピーする場合にも注意が必要だ。あるバージョンのSambaではデフォルト値になっているものが、別のSambaでは非デフォルト値だったりすると、同じsmb.confを使っているはずなのに、2つのSambaの挙動が異なることもある。

■共有エントリの作成

SWATを使っての共有リソースの作成や設定変更は、ディスク共有の場合は【SHARES】アイコン、プリンタ共有の場合は【PRINTERS】アイコンをクリックし、表示されるページで行う。**画面2**は、【SHARES】ページの初期画面だ。ここから以下のような手順で作業すれば、新しい共有を作成することができる。

①新しい共有名を【Create Share】ボタンの隣のテキストボックスに入力する。日本語は使わないこと（詳しくは後述）。

②【Create Share】ボタンをクリックすると、画面3のように共有リソースの設定が表示され、編集できるようになる。

③pathパラメータの設定を、共有するディレクトリのパスに書き換える。
④使い勝手を考えれば、commentパラメータ（ブラウズ時に表示されるコメント）も設定しておいたほうがいい。共有名と同じく、ここでも日本語を使わないこと。
⑤read onlyパラメータはデフォルトで“yes”になっていることが多いので、書き込みを行いたい場合は“No”に修正する。
⑥その他のパラメータについても、必要に応じて修正・設定する。【Advanced View】ボタンをクリックすれば、設定可能な全パラメータが表示される。

設定が終わったら、【Commit Changes】ボタンをクリックしてsmb.confを更新するのを忘れないこと。これを忘れると、すべての作業がパーになる。

必要な設定変更を終えたら、【STATUS】ページを表示させてsmbdをリスタートすれば、新しい共有が使えるようになる。

Sambaと日本語

Sambaは海外で作られたソフトウェアだが、日本語ユーザーのために、シフトJISコードを扱うための機能が用意されている。つまり、日本語化されたSambaだけでなく、本家Sambaでも、日本語ファイル名が扱えるようになっているのだ。

ただし、そのためにはsmb.confで、ちょっとした設定を行わなくてはならない。ここまでは、説明のチャンスがなかったこともあって、日本語ファイル名については一切考慮してこなかった（初回に紹介したsmb.confも、日本語ファイル名がきちんと扱えるものではない）のだが、SWATのインストールを済ませて環境設定に取りかかる準備もできたことだし、このへんで、Sambaで日本語ファイルを扱う場合について、きちんと説明しておこう。

Samba（＋Linux）で日本語ファイルを扱う場合については、考慮しなくてはならない要素が3つある。Samba本体、Linux（UNIX）のファイルシステム、そしてLinuxのユーティリティだ。

■ファイルの内容

取り上げるほどのことではないかもしれないが、最初に説明しておくのは、ファイルの内容についてだ。Sambaとファイルシステムは、どちらもファイルの内容に関しては一切関知しない。

Samba	**非関知**
ファイルシステム	**非関知**
ユーティリティ	**コマンドによる**

つまり、実行ファイルだろうが、テキストファイルだろうが、クライアントから渡されたデータをそのままディスクに書き込み、あるいはディスクから読み出したデータをそのままクライアントに渡す。日本語テキストを含むデータをSambaで扱っても、文字化けなどのトラブルは一切起こらない。

Sambaクライアントから書き込んだファイルをLinux上のユーティリティでアクセスする場合には、ちょっとした注意が必要だ。Linuxなど、UNIX系OSでは標準日本語文字セットとしてEUCと呼ばれるコードを使っている。それに対してWindows系OSでは、シフトJISコードが標準だ。シフトJIS非対応のコマンドを使うと、期待どおりの処理が行われないことが多い。当たり前のことだが、英語版など、日本語非対応のLinuxを使っている場合、付属コマンドも日本語を考慮していないので、うまく処理できないことが多い。

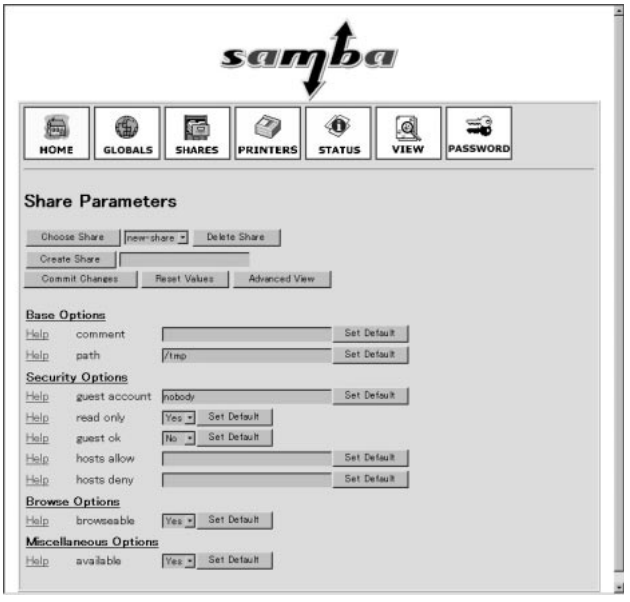
■日本語ファイル名の問題点

Sambaを使う場合に、一番問題になるのが日本語を含むファイル名のトラブルだ。これについては、Sambaの設定によって、ある程度対応できる。

Samba	**設定に応じた文字コードとして解釈**
ファイルシステム	**使える文字は限定**
ユーティリティ	**コマンドによる（通常はそのOSの標準文字セットにのみ対応）**

本家Samba（英語版Samba）をデフォルト設定で使っている場合、ファイル名の変換は一切行われない。クライアントから受け取ったファイル名は、そのままファイルシステムに渡される。ファイルシステムはその文字列を、自分がサポートしている文字セットだと見なして解析し、有効なファイル名であればディスクにアクセスする。

ここで問題になるのが、シフトJISでは、日本語文字（2バイト文字の一部に特別な意味を持つ文字コードが使われている、ということだ。次の例を見てほしい。



画面3　SHARESページの編集画面
共有名の作成、または選択を行うと、このような画面が表示される。パスやコメント、各種パラメータを入力してから【Commit Changes】ボタンをクリックしよう。

（EUCの場合）

```
# echo '予感' | od -t x1
0000000 cd bd b4 b6
```

（シフトJISの場合）

```
# echo '予感' | od -t x1
0000000 97 5c 8a b4
```

odコマンドは、入力を16進数などで表示するためのユーティリティだ。このコマンドラインによって、“予感”という文字列が、実際にはどのようなコードなのかを確認できる。

EUCの場合、漢字などの2バイト文字は、すべて0x80以上のコードが使われる。“予感”の場合は、0xcd 0xbdが“予”の文字に、0xb4 0xb6が“感”の文字に対応している。これらのコードは、英語版など、日本語非対応のLinuxだとしても、すべて通常の文字（2つの1バイト文字）として扱われるため、ファイル名として扱うことに何の問題もない。日本語フォントが用意されていないので、lsなどのコマンドで見るとファイル名が化けて見えるし、bashなどのユーティリティも2バイト文字に対応していないため、コマンドライン編集の時には不都合が生じることがあるものの、アプリケーション（この場合はSamba）が指定したファイル名がそのままディスク中に書き込まれ、ディスクから読み出したファイル名がそのままアプリケーションに渡されるという点では、問題ないわけだ。

一方、シフトJISの場合はどうだろうか？　上の実行例を見ればわかるように、“予”の文字コードは 0x97 0x5c、“感”のコードは0x8a 0xb4となっている。シフトJISでは、2バイト文字の1バイト目には0x80以上のコードを使うが、2バイト目には0x40以上の文字を使う。そのため、指定されたファイル名を英語のファイル名のつもりで解釈すると、2バイト目のコードの一部が、「特殊文字」とされているコードになってしまうことがあるのだ。この場合、“予”の2バイト目の0x5cが、“\”（英語の場合は“＼”）、つまりバックスラッシュとなっているが、これはWindowsのパスの区切りだ。当然、これをパスの区切りとして処理しようとするプログラムは、エラーになる。

Windowsクライアントがシフトjisコードを使い、日本語文字の2バイト目にパスの区切り文字など、特殊なコードが含まれてしまうこと、これが「英語版Sambaで日本語ファイル名が使えない」というトラブルの原因だ。

■Sambaの設定でファイル名を変換する

Sambaが、シフトJISの文字列をそのままファイルシステムに渡しているのが問題の原因とわかれば、その解決方法もすぐにわかるだろう。シフトJISのファイル名をSambaが別の文字コードに変換してやればいいのだ。

Sambaには、この機能が最初から組み込まれている。つまり、英語版Linuxで動作する英語版Sambaであっても、smb.confの設定だけで、シフトJIS（日本語Windows）対応になるのだ。

2バイト文字コードのWindowsクライアントに対応するためには、smb.confの[global]セクション中で、以下のように指定する。

```
coding system = euc
client code page = 932
```

SWATで設定する場合は、まず【GLOBAL】アイコンを、さらに【Advanced View】ボタンをクリックする。【Filename Handling】というセクションの下に、この2つのエントリが用意されているので、それぞれ“euc”、“932”と指定すればいい。前回紹介した日本語Sambaであれば、この2つがデフォルト値としてコンパイルされているため、smb.confに何も書かなくても日本語ファイル名が正しく処理される。

client code pageパラメータで指定しているのが、「Windowsクライアントで使っている文字セット」だ。こ

こで指定している“932”というのが、Windowsの日本語文字コード、シフトJISを意味している。一方のcoding systemパラメータで指定しているのが、「2バイト文字を、どのようなファイル名に変換するか」だ。ここでは、英語版Linuxでも一応問題が起こらない、EUCを指定している。SWATの【STATUS】ページや、Linuxのコマンドラインからのlsコマンドなどでは表示が化けるが、Windowsクライアントからは正しいファイル名が確認できるだろう。ご存じのとおり、日本語対応のLinuxでも標準漢字コードとしてEUCを使っているので、この指定をすることで、Linuxのlsコマンドなどでもファイル名が正しく表示されるはずだ。

Sambaでは、EUC以外にも何種類かの変換文字セットを用意している。詳しくはコラム「Sambaのコーディングシステム」を参照してほしい。

蛇足だが、smb.confには“character set”というパラメータも用意されている。これは特殊な1バイト文字セット用のもので、日本語を扱う場合にはまったく関係がない。下手に指定するとトラブルの原因になるので、何も指定しないこと。

■日本語の共有名

Sambaを使う場合、もうひとつ考慮しなくてはならないのが、2バイト文字を含む共有名とコメントだ。これについては、Sambaの日本語サポートはまだ不完全だ。かなり限定された条件以外では、共有名やそのコメントに、日本語は使えないと思ったほうがいい。

まず、SWATの日本語サポートが不完全なために、2バイト文字を含む共有名やコメントが入力できない。新規の作成も無理だし、smb.confを直接編集して日本語を指定した場合も、文字化けなどが起こって正しく表示できない。

smb.confを直接編集してやれば、日本語文字を含む共有名などは作成できるのだが、クライアントがNTやWindows 2000の場合には文字化けが起こって正しく表示されない。

つまり、以下の条件が揃っている場合のみ、辛うじて日本語の共有名やコメントが使える（可能性がある）ということだ。

- **クライアントはWindows 9xのみ（NTやWindows 2000は使っていない）**
- **smb.confの編集にSWATを使わない（テキストエディタで編集する）**

SWATの対応状況やNTクライアントのサポート状況を見てもわかるように、現在のSambaでは、共有名やコメントでの2バイト文字の利用は考慮されていないと考えたほうがいい。エンドユーザーからは文句が出るかもしれないが、日本語を使わないのが無難というものだ。

ちなみにこのような使い方をする場合、smb.conf中の日本語文字はEUCで記述する。Windowsクライアント側のテキストエディタを使うと、文字コードがシフトJISになってしまったり、改行コードがLinuxではなくWindowsのものになってしまったりするので注意すること。

Column

「Sambaのコーディングシステム」

本文でも説明したとおり、日本語Windowsで日本語を含むファイル名を指定すると、Linux上では正しいファイル名として扱えないことがある。この問題に対処するために用意されているのが、smb.confのcoding systemパラメータだ。

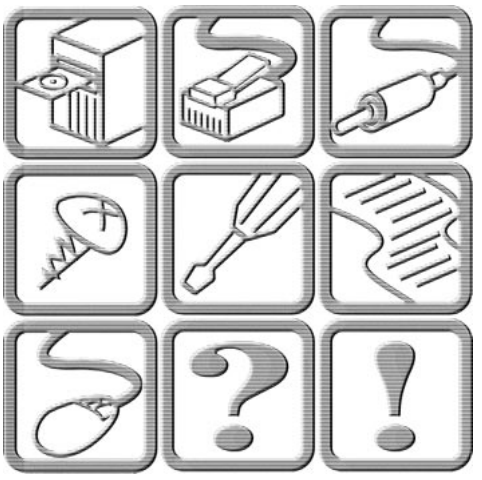
このパラメータは、client code pageパラメータが932（シフトJIS）の場合にのみ有効になる。指定できるのは以下のような文字列だ。

オプション値	内容
SJIS	シフトJISそのまま
JIS8	8ビットJIS
JIS7	7ビットJIS
JUNET	JUNETコード
EUC	EUCコード
HEX	16進のテキスト化コード
CAP	CAP（Columbia AppleTalk Program）互換

いろいろなオプション値が用意されているが、これはSambaがオープン、つまり異機種接続性を考慮した設計になっているからだ。つまり、日本語コードとしてJISやシフトJISを使っているシステムでもSambaが動作するように、このように多彩な設定が用意されているのだ。Linuxを始めとする、現在主流の日本語対応UNIXのほとんどは、日本語コードとしてEUCを使っているので、Sambaでも、これを使うのが基本となる。

ただし、英語しか扱えない端末やコンソールからも、該当ファイルを扱うことが多いのであれば、HEXを使うのもいいだろう。これらの変換モードでは、英語端末でも入力・表示できる文字にファイル名を変換するため、コマンドラインでの指定が多少は容易になる。

Try & [trái] Try [trái]



システムのお引越し

文：政久忠由
Text : Tadayoshi Masahisa

今回は、システムの引越しを行いたいと思う。さすがに通常の引越しを想像している人はいないだろうが、一応説明しておくと、ここではハードディスクのスワップが中心となる。今まで過ごしていた家が手狭になったので、大きな家に引っ越す、そんなイメージである。

僕のシステム環境では、Linuxマシンは、ハードウェアリソースを分配して割り当てる際のプライオリティ付けが一番低い。その理由はさておき、つまりはLinuxマシンには、“おさがり”部品が集まってくるのだ。今回、別のシステム用に新しいハードディスクドライブを購入したので、そのシステムで使用していたハードディスクが余ることになった。そこで、Linuxマシンにというわけだ。単に増設して使ってもよかったのだが、このハードディスクの性能のほうが今使っているものより“よい”ので、スワップしてしまうことにしたのである。

引越し準備

慧眼（けいがん、えげん）という言葉がある。ものごとの本質を見抜く、優れた眼力とか洞察力のことなんだけど、システムの引越しには、この慧眼がもっとも必要とされる。

想像してみよう。何をどのように移行すれば、問題なく、以前どおりに稼動させることができるのか。実際にやってみないと実感としてわかないかもしれないが、想像の段階でディテールまでキチンと把握できていないと、たいていの場合、トラブルに見舞われて引越しは失敗に終わる。ディテールまでキチンと把握しておくこと、つまり、本質を見抜いておくことで、問題を極力回避し、また、もしトラブルが発生したとしても容易に対処できるようになるのだ。

では、まずシステムの引越しについての手順を考えてみよう。今回の手順の大筋をまとめると次のようになる。

①新しいディスクを適当な場所に接続し、パーティション構成を設計する。
②既存のディスクから新しいディスクへ必要なデータすべてをコピーする。
③新しいディスクからブートできるようにする。

ざっとこんな感じだ。それでは、個々の部分のディテールを詰めていくことにしよう。なお、これ以降のすべての手順は、基本的にスーパー（ルート）ユーザーで行うものとする。

新しいディスクの構成

新しいディスクを接続して、パーティションを構成すること自体は、それほど難しくないと思うので、ポイントだけ説明しておこう。

■パーティションサイズとブロックサイズ

実際にLinuxを運用してみると、最初に設定したパーティション構成が良くなかったことに気づくことは多い。必要以上に細かく分けてしまって使い勝手が悪かったり、逆に分けなくて効率が低下したりと、後悔は尽きない。新しいディスクでは、それらの経験を踏まえてパーティションを構成しよう。

僕の場合、実験用で自分以外使わないので、/（ルート）用に4Gバイト、/var用に1Gバイト、ソースコードを展開する作業用に10Gバイト、そして各種ドキュメント、データ用に2Gバイトとした（実は以前と比べて、作業用を増やしただけなんだけどね）。スワップ領域は、僕の使い方では、今まで使われることがなかったので思い切ってなしとした。

またファイルシステムのブロックサイズについても多少考慮したほうがよい。頻繁にアクセスするファイルを格納する部分は4Kバイト、たまにしかアクセスしないデータやProxyキャッシュの領域は1Kバイトといった感じだ。

ファイルのコピー

新しいディスクのパーティション構成、そしてファイルシステム作成が終わったら、適当なところにマウントしてファイルの移行を始めることにしよう。

マウントする位置はどこでもよいのだが、ここでは、/mntの下に適当なディレクトリを作成してマウントすることにする。

```
# cd /mnt
# mkdir n1 n2 n3 n4
# mount /dev/hdb1 n1
# mount /dev/hdb2 n2
…
```

早速、ファイルのコピーといきたいところだが、その前に移行時の余計なディスクアクセスを避けるためにシステムをシングルユーザーモードにしておこう。もちろん最初からシングルユーザーモードでブートしておいてもいい。

```
# init S（もしくは1）
```

シングルユーザーモードへの移行が完了するとちょっとしたログデーモンとシェルが動作している状態になる。一応psコマンド（ps auxなど）で確認してほしい。

シングルユーザーモードになったことで、ほかのユーザーやさまざまなデーモンからのディスクアクセスを気にせず、作業できるようになった。いよいよファイルをコピーすることにしよう。

■パーミッション情報の保全

しかし、単純にファイルをコピーすればよいというものではない。ファイルのパーミッション情報やシンボリックリンクファイルの扱いを考えなくてはならないし、ディレクトリ以下、全部まとめてコピーもしたい。

ファイルのパーミッション情報やシンボリックリンクファイルなどをそのままの状態でコピーするには、いくつかの方法がある。ここでは、cpコマンドとtarコマンドの2種類の方法を紹介しておこう。

■cpコマンド

Linuxでファイルのコピーを行うための標準的なcpコマンドは、GNU　fileutilsに含まれているものが主流である。このcpコマンドにはいくつかのオプションが用意されていて、今回のような要件にも対処できるようになっている。ここでは、-aオプションを利用する（詳細はヘルプを参照）。-aオプションは、-dpRと同じ意味で、リンクの状態、属性を保持し、再帰的にコピーするためのものだ。なお、GNU　fileutilsの古いバージョンだとこれらのオプションはないので注意すること。

```
# cp -a /bin /dev /etc /sbin /lib /usr  …  /mnt/n1
```

コピー元に/（ルート）を指定したいところだが、いくつかファイルシステムをマウントした状態なので個別に指定している。また/varなど、マウントして利用するファイルシステムへは、ディレクトリからではなく、その内容からコピーするように気をつけてほしい。つまり、/varを/mnt/n2にコピーする際、/mnt/n2/varとならないようにする必要がある。

■tarコマンド

tarコマンドを用いる方法では、通常のアーカイブを作成するオプションに加えて、pオプションを指定する。これでリンクや属性の状態が保持されるようになる。tarコマンドは、次のように使用する。

```
# tar cfp - /bin | (cd /mnt/n1 ; tar xf -)
```

まず、cfpはそれぞれ、アーカイブを作成、ターゲットはファイル、リンクや属性の状態を保持というオプションだ。続くスペースにはさまれた－（ハイフン）は、標準出力を表す。つまり/binをアーカイブして標準出力に出力するという意味だ。その出力はパイプでつながれたコマンドに渡される。こちらでは、cdコマンドでカレントディレクトリを移動し、tarで標準入力を受け、展開する。なおtarには、-Cオプションがあり、ディレクトリを指定して変更できるのだが、オプションが複雑になるのを避けるためにあえて、cdコマンドを利用している。

cpコマンド、tarコマンド、どちらを用いても結果は同じなので、好みの方法を採用するといいだろう。まあ、cpコマンドのほうが楽だとは思うけど。

新しい構成のチェック

ファイルのコピーが完了したら、次は、新しいディスクをhdaに付け替える前のチェックだ。新しいディスクのルートファイルシステムになるパーティションのetcに移動し、fstabを確認しよう。

```
# cd /mnt/n1/etc
```

fstabでは、新しいディスクのパーティション構成に合わせてマウントポイントなどの修正を行う。ルートデバイスさえ、きちんと設定してあればとりあえずはどうにかなると思う。

ブートはどうする

新しいディスクをhdaに付け替えた時、ブートするにはどのようにしておけばよいのだろう。

これには、liloを用いることができる。単純に現在のディスク構成でディスクジオメトリ、つまりliloのセカンドローダやLinuxカーネルの位置を算出した場合、ディスク内のジオメトリ自体は問題ないが、ディスク装置の番号が問題となる。通常BIOSでは、1、2、もしくは1～4のディスク装置の指定ができ、liloでもそれを指定しているのだ。しかしliloには、指定したディスク装置を特定のディスク装置に見立てて設定するオプションが用意されているので、今回のようにhdbのディスク装置をhdaとして設定することが可能だ。これには、disk=オプションに加えて、bios=でディスク番号として0x80などを関連付けるのだが、少々面倒なので、ここでは手軽なブートフロッピーを使用することにする。

ブートフロッピーの作成

ここで作成するブートフロッピーは、カーネルを含むものだ。liloのブートイメージが収められたものではなく、ブートとカーネルをメモリへ展開するルーチンの付加された圧縮カーネルイメージである。

今回のようなシステムの引越しをやってみようと考えるユーザーなら、カーネルの再構築ぐらいは行ったことがあると思う。カーネルを再構築した際、/usr/src/linux/arch/i386/bootに作成されるbzImageもしくはzImageが上記の圧縮カーネルイメージだ。/boot/vmlinuzもそれをコピーしてリネームしているだけなのでそちらを使ってもよい。

この圧縮カーネルイメージには、実は、ルートデバイス、スワップデバイス、RAMディスクサイズ、ビデオモードなどの情報が含まれている（ファイルの最後に付加されている）。ここで重要なのはルートデバイスだけだが、最初にこの情報を確認しておこう。これらの情報の設定、確認には、rdevコマンドを使用する。

```
# rdev bzImage
Root device /dev/hda1
```

このbzImageファイルでは、ルートデバイスとして/dev/hda1が設定されていることがわかる。通常、liloを用いてブートする場合には、lilo.confの中でルートデバイスを指定していると思うが、これは、カーネルをロードする際に、引数としてそのルートデバイスを渡しているのだ。すなわちbzImageファイルのデフォルト値は無視されるわけだが、ここで作成するブートフロッピーでは引数を渡さない（渡すすべがない）ので、きちんとデフォルト値を設定しておく必要があるのだ。

もし、ルートデバイスが/dev/hda2となるのであれば、次のように設定する。

```
# rdev bzImage /dev/hda2
```

これでbzImageは、/dev/hda2をルートデバイスとしてマウントし、/sbin/initをキックして、/etcの初期化スクリプトを実行するようになる。

それでは、ルートデバイスをきちんと設定したbzImageをフロッピーディスクにコピーして、ブートフロッピーを作成することにしよう。イメージは直接/dev/fd0にcpコマンドやddコマンドでコピーすればよい。

```
# cp bzImage /dev/fd0
or
# cp /boot/vmlinuz /dev/fd0
```

これでブートフロッピーの完成だ。あとは、再度新しいディスクのetcディレクトリのfstabを確認して、システムのシャットダウンを行う。そしてディスクの物理的な設定変更のあと、ブートしてみよう。フロッピーディスクから起動し、新しいディスク装置のルートデバイスがマウントされ、初期化スクリプトの処理が始まれば成功だ。スクリプトの処理が終わり、ログインプロンプトが表示されたら、rootでログインして、liloの設定を行う必要がある。この時点では、余計なトリックなどの必要はないので、素直にlilo.confを設定し、liloを実行すればよい。そうすれば次回からはブートフロッピーでなく、ハードディスクから起動できるようになると思う。

カスタムカーネルではない場合のブートフロッピーの作成

カーネルを再構築して作成したカスタムカーネルでは、基本的にブートの際にそのほかの余計なモジュールの支援は必要ない（必要ないように設定するから）のだが、ディストリビューションが配布するカーネルを使用している場合は、もしかすると依存している可能性がある。これらの場合に対応できるようにliloを用いたハードディスクにあまり依存しないブートフロッピーの作成方法も紹介しておこう。

まず、フロッピーディスクをフォーマットし、ファイルシステムを作成する。ここでは一般的な1.44Mバイトのフォーマットを行っている。またファイルシステムは対応していれば何でもよいのだが、カーネルに必ず含まれているであろうext2を採用している（FATとかminixだと含まれているかどうか不安なので…）。

```
# fdformat /dev/fd0H1440
# mkfs /dev/fd0
```

作成したらフロッピーディスクを/mnt/floppyなどにマウントし、/bootのvmlinuz-2.xx、initrd-2.xx、module-info-2.xx、そしてスペースが許せばSystem.map-2.xxをコピーする。

次に/etc/のlilo.confをlilo.conf.fdにコピーし、次のような変更を行う。

```
boot=/dev/fd0
root=/dev/hda1
read-only
image=/mnt/floppy/vmlinuz-2.2.12-20
        label=linux
        initrd=/mnt/floppy/initrd-2.2.12-20.img
```

boot=には/dev/fd0、root=にはハードディスク上の/dev/hda1を指定している。またカーネルイメージ、モジュールイメージとして、/mnt/floppy/のファイルを指定している。そんな指定で大丈夫なの？と不安を覚える人もいるかもしれないが、これらのイメージはliloが実行される段階でBIOSコールのパラメータに置き換えられるので問題ないのだ。/mnt/floppy/という指定でもちゃんとフロッピーディスクのCHS番号として設定されるのである。

この設定を有効にするには、-Cオプションを指定して次のように行う。

```
# lilo -C lilo.conf.fd
```

最初に説明したブートフロッピーと比べて、こちらの場合は、liloを使用しているので、カーネルにブートパラメータを渡すことができるというメリットがある。

今回は、ハードディスクをスワップする目的で話を進めてきたが、これらのテクニックは、単にルートデバイスやディスク（パーティション）構成を変更したり、何かしらのトラブルシューティングをしたりするなど、さまざまな用途に応用できるので、ぜひとも使いこなせるようになっておこう。

Windowsと共存しているディスクの場合

ディスクにWindowsを共存させている場合は少し面倒だが、難しくはない。まぁ、面倒なだけあって説明すると

長くなるのでここでは割愛するが、ファイルのコピーに関しては、リペア用の環境を別途用意して行えばよいし、ブートに関しても、Windows 9xの場合は、それぞれの修復ディスクからsysコマンドでio.sysへのジオメトリを設定し直すだけ、Windows NTの場合は、ファイルシステムをフォーマットした段階で最下位のブートコードが埋め込まれるようになっているので、特別なことをしなくてもファイルシステム上のntldrを見つけられるようになっている。

またWindows NTのブートローダからWindows 9xをブートするようにしている場合は、まずWindows 9xの設定を行い、Windows NTの修復セットアップでブート環境を修復するか、直接bootsect.dosファイルのio.sysへのジオメトリを書き換えるなどする。Windows NTの修復セットアップは時間がかかるので、Windows系のファイルシステムのフォーマットはすべてNTで行い、NTがブートできる状態にしておき、ひとまず、その状態のパーティションブートセクタをファイルに保存する。次にWindows 9xのsysコマンドを実行したのち、そのパーティションブートセクタをbootsect.dosとして保存する。そして先ほど保存しておいたNTのブートセクタファイルを書き戻すのが手っ取り早いだろう。この時、適当なディスクエディタを持っていないのであれば、Linuxのddコマンドが便利だ。たとえば、/dev/hda1がWindowsのパーティションだとすると、次のように実行する。

Windows NTのブートセクタを保存する：

```
# dd if=/dev/hda1 of=bootsect.backup bs=512 count=1
```

Windows 9xでsysコマンドを実行したあと、ブートセクタを取り替える：

```
# dd if=/dev/hda1 of=bootsect.dos bs=512 count=1
# dd if=bootsect.backup of=/dev/hda1 bs=512 count=1
```

if=で入力ファイル、of=で出力ファイル、bsで読み書きするサイズ、count=でその回数を指定する。つまり上記の最初のコマンドの場合は、/dev/hda1から読み出し、bootsect.backupに書き出す、サイズは512バイト、回数は1回ということになる。

ループバックデバイスの活用

冒頭での今回のパーティション構成を見るとわかると思うけど、僕は細かくパーティションを分割するのは好きじゃない。でも、各種実験の時など、ちょっとだけ分けたいこともある。こんなときに利用するのが、ループバックデバイスだ。ループバックデバイスは、実際のデバイスとは異なり、すべてが内部的に処理される仮想デバイスと考えておけばよい。フロッピーディスクのシミュレーションなどにもよく利用されるので、見たことがあるかもしれない。そういえば、ループバックファイルにLinuxのシステムファイル全部入れて、Windowsと簡単に共存できるようにしたディストリビューションもいくつかあるようだし。

ループバックデバイスでは、通常のファイル（ブロックデバイスでもOK）の中にファイルシステムを作成して、任意のディレクトリにマウントして使用することができる。ここで、少し作り方を説明しておこう。

```
# dd if=/dev/zero of=test.dat bs=1k count=200000
# losetup /dev/loop0 test.dat
# mkfs /dev/loop0
# mount /dev/loop0 /mnt/8
# umount /dev/loop0
# losetup -d /dev/loop0
```

まず、ddコマンドを使用して、空のファイルを作成する。ここでは、入力デバイスとして/dev/zeroを指定して必要なサイズ、1Kバイトを20万回、つまり200Mバイトの0で埋め尽くしたファイルを生成している。

次にlosetupコマンドで、/dev/loop0と作成したファイルfile.loを関連付けている。次のようにデバイス名だけをオプションとして指定すると状態を見ることができる。

```
# losetup /dev/loop0
/dev/loop0: [0301]:106462 (test.dat) offset 0, no
encryption
```

losetupコマンドでは、暗号化の設定ができるようになっており、-eオプションで指定することができる。一種の暗号化ファイルシステムとしても利用できるわけだ。暗号化には、XORとDESを設定可能だが、DESルーチンはカーネルに含まれていない場合がほとんどなので、たぶん普通は利用できないと思う。

続いてmkfsコマンドで/dev/loop0にファイルシステムを作成し、mountコマンドで/mnt/8にマウントしている。ファイルシステムはシステムが対応しているものであれば

何でもよい。ここでは何も指定していないのでデフォルトのext2、ブロックサイズ1Kでフォーマットされる。/mnt/8にマウントしたあとは、通常のファイルシステムと同様に扱えるようになる。

取り外すときは、umountコマンドを用い、さらに/dev/loop0との関連付けを削除するために、losetupコマンドで-dオプションを指定して実行している。

なお、mountコマンドだけでも操作可能だ。

```
# mount -o loop test.dat /mnt/8
# mount
/root/file.lo on /mnt/8 type ext2 (rw,loop=/dev/loop0)
```

mountコマンドで-o loopオプションを指定することで、losetupで指定していた/dev/loop0などとの関連付けを自動的に行わせることができるのだ。またこの場合、umountコマンドでデバイスを取り外せば、losetup -d /dev/loop0を実行しなくても、関連付けが削除されるようになっている。

また次のような使い方もできる。

```
# dd if=/dev/fd0 of=fdimage
# mount -o loop fdimage /mnt/floppy
```

この例では、フロッピーディスクのイメージをfdimageファイルに書き出し、それを/mnt/floppyにマウントしている。

ループバックデバイスのパフォーマンスは、通常のディスク上のファイルシステムと比べると約半分程度に低下する（Rewriteやランダムシークはさらに劇的に遅くなる場合もある）が、それほどパフォーマンスを求める分野で利用するものでもないので問題にはならないだろう。でも、フロッピーと比べると断然速い（当たり前だけど）。あと、作成するループバックファイルをスパースファイルにしておくと、省スペースにもなる。

スワップファイル

今回スワップパーティションはなくしてしまったわけだが、どうしてもスワップ領域が必要になることがあるかもしれない。こんな時は、どうすればいいのだろう？

実はスワップ領域は、専用のパーティション以外、ズバリ、通常のファイルにも作成することができるのだ。まあ、ブロックデバイスとかの仕組みを考えれば、できて当然なのだが、問題点としては、先ほどのループバックファイル同様、パフォーマンスが犠牲になってしまうことだ。

このパフォーマンスの問題さえ納得できれば、スワップ領域を通常のファイルにしてしまうメリットは大きい。

```
# dd if=/dev/zero of=/pagefile bs=1k count=1000000
# mkswap /pagefile
Setting up swapspace version 1, size = 1023995904
bytes
```

まず、ループバックファイル同様、適当なサイズのファイルを生成する。ここでは、100Mバイトのサイズでルートにpagefileというファイルを作成した。ファイル名、その格納場所の制限はないが、一応、パーミッションは、600にしておくと安全だ。

次にmkswapコマンドでそのファイルをスワップ領域用にセットアップする。mkswapコマンドでは、スワップ領域としての印をつけるだけで、システムが使えるようにはならない。システムで使用できるようにするには、次のswaponコマンドでそのスワップ領域を有効にする必要がある。-sオプションを使用するとシステムのスワップ領域の状態を確認することができる。一応、freeコマンドでメモリ使用の状況も見ておこう。なお、スワップ領域を解除するには、swapoffコマンドを使用する。

```
# swapon /pagefile
# swapon -s
Filename        Type    Size    Used    Priority
/pagefile       file    999992  0       -1
```

このスワップ領域は、fstabに次のように記述しておくことができる。このようにしておけば、起動時に自動的にマウントされるようになる。ただし、ファイルが格納されているファイルシステムがマウントされている必要があるので、記述する順番には気をつけよう。

```
/swapfile  swap  swap  defaults  0  0
```

パーティション領域と違い、ファイルだと柔軟にサイズの変更などが行えるので、パフォーマンスさえ気にしなければ、結構便利だ。

Linux Today

米国LinuxToday提携
http://linuxtoday.com/
毎月、米国の人気Linuxサイト「Linux Today」に掲載された記事の要約をお届けします。記事の全文は日刊アスキーLinuxで読むことができます。
http://www.linux24.com/

訳：日下部圭子

フリーソフトウェアの15年間

Text : Richard Stallman
http://linuxtoday.com/stories/4145.html（原文）
http://www.ascii.co.jp/linux/linuxtoday/article/article323968-000.html（邦訳全文）

フリーソフトウェア運動およびGNUプロジェクトが開始されてから、ようやく15年を超えた。これまでの間に我々は大いに進歩をとげた。

1984年には、ライセンスによる制限のある商用OSをインストールせずには、最新のコンピュータを使うことはできなかった。人々は誰も、ソフトウェアを仲間のコンピュータユーザーと自由に共有することを許されず、また自分の必要に合わせてソフトウェアを変更することは困難だった。ソフトウェアの所有者たちは、私たちをばらばらに分断する壁を作り上げていた。

そうした状況を一新するために、GNUプロジェクトが設立された。その最初の目的は、UNIX互換の、移植性のある、100%フリーなOSを開発することだった。 95%ではなく、99.5%でもなく、100%フリーであることによって、ユーザーはシステム全体を再配布することができ、また、それを自由に変更してコントリビュートできるのだ。このシステムのGNUという名前は「GNU's Not Unix」の頭文字をとった再帰的なものだ。これはUNIXに敬意を表わし、同時に、それとは異なるものであることを示している。技術的にはGNUはUNIXに似ている。しかしUNIXとは異なり、GNUはユーザーに自由をもたらすのだ。

このOSの開発では、何百人ものプログラマーが何年も働く必要があった。中にはFree Software Foundationや、フリーソフトウェアをビジネスにする企業によって報酬が支払われた人々もいたが、ほとんどはボランティアだった。有名になった人もいなくはないが、ほとんどの人々は、そのコードを使ったり変更したりするほかのハッカー仲間から知られているだけだ。しかし全員が人類すべてのために、コンピュータネットワークの可能性を解放するための手助けをした。

1991年には、UNIXライクなシステムの、最後に残った重要な本質部分が開発された。これがLinuxであり、Linus Torvaldsが書いたフリーなカーネルである。今日では、GNUとLinuxの組み合わせは世界中の数百万の人々に使われていて、その人数は増えつつある。99年3月、我々はGNOMEのリリース1.0を発表した。これはGNUのグラフィカルなデスクトップで、我々はこれによってGNU/Linuxシステムを、他のOSと同様に使いやすくしたいと考えている。

しかし、我々の自由は永久に保証されているわけではない。世界は不動のものではなく、現在自由だからといって今後5年間自由でいられることを、あてにはできないのだ。フリーソフトウェアは、困難な問題や脅威に直面する。我々の自由を守るためには、初めに我々が自由を得たときと同じぐらいの、断固とした努力が必要だ。一方、このOSはまだ動き始めたばかりだ。今度はユーザーが望むすべての分野の仕事を扱えるように、フリーのアプリケーションを付け加えていく必要がある。

これからも、フリーソフトウェア社会が直面している固有の難問について、またGNU/Linux OSについての事情や、コンピュータユーザーの自由に影響を及ぼす問題点について書いていくつもりだ。

プロフィール
1984年にRichard StallmanはMITを退職し、フリーなUNIXライクなOSの開発を始めた。彼はその活動の中でFree Software Foundationを設立し、GNU General Public License（GPL）を作り上げ、gccやEmacsを作り上げた。

Corel社CEO Michael Cowpland博士インタビュー

Text : Dwight Johnson

http://linuxtoday.com/stories/4147.html（原文）

http://www.ascii.co.jp/linux/linuxtoday/article/article327232-000.html（邦訳全文）

なぜ、CorelはLinuxに力を入れるようになったのか。今日のCorelを築いたCowpland博士にLinuxTodayのDwight Johnsonが聞く。

Dwight：初めてLinuxに関心を持ったときのことを教えてください。

Michael：それは昨年の初めのことでした。私たちはNetWinder（http://www.rebel.com/netwinder/）のためのOSを探していて、QNXやいくつかの組み込みシステムを検討していたのですが、私たちの開発者の1人が、Linuxを提案したのです。調べれば調べるほどLinuxを気に入りました。私たちはソースコードの内容を調べ、それがあまりにもすっきりうまく書かれているので信じられないような気持ちでした。それが最初で、私たちはLinuxをNetWinderに採用し、それはすばらしい働きをしてくれました。それから私たちはOttawaでのユーザーグループミーティングに行き、NetWinderを起動したところ大きな反響を得ました。この時点で私たちはLinuxは将来有望なOSであることを十分に理解したので、自分たちのすべてのアプリケーションをLinuxに移植しているところなのです。

私たちがそれに取り組み始めたのが1997年の末ごろで、1998年初頭に自分たちのすべてのアプリケーションを移植することを明言し、WordPerfect 8を移植しましたが、最近そのダウンロード回数が60万回を超え、それ以来、次のアプリケーションを提供する最良の方法はそれらについて一度にひとつずつやるのではなく、Wineへの注力を大幅に増やすことだとわかりました。そうすることで自動的に各アプリケーションの90%の作業が済んだことになり、コンパイルすることで各アプリケーションの調節をある程度可能にしつつも、作業の大部分はもう済んでいるということになるわけです。

Dwight：ではそのときすでにNetWinderを計画していたのですね。昨年5月にNetWinderをアナウンスしましたよね。そのときにもうLinuxを使うと決めていたのですか?

Michael：そうです。私たちはそれよりずっと前にOSを決めていなければなりませんでした。1997年の末には決まっていたはずです。

Dwight：でも、thin-clientとして計画したことは初めから正しかったのですか?

Michael：ええ、これは基本的にネットワークコンピュータであると同時にthin-clientです。そして、Linuxコンピュータとなりました。

Dwight：Javaを検討しませんでしたか？

Michael：当初、私たちはあるOS上で動くJavaを想定していて、OSはある程度無関係なものになるだろうと考えていました。それから突然、LinuxがJavaより重要だと分かったのです。それが転換のときでした。実際、それは妙なことでしたが、私たちはだんだんわかってきたのです。私たちは、Javaは必要不可欠ではないようだ、枝葉末節の部分になるはずだと考え始めました。そして現状その通りです。私たちにはJVMがあります。これは持っているべき重要なツールです。Javaで書かれたMIS（Management Information System）ミドルウェアがたくさんあります。このため、これは確かにある役割を果たしています。しかし我々の重要視する部分はすぐにほかへ移ってしまったわけです。

Dwight：話は変わりますが、現在Corel内で使われているLinuxやオープンソースソフトウェアはいかがですか？

Michael：私たちはLinux上で数多くのサーバを稼働させています。

Dwight：Webサーバですか？

Michael：ええ、そして私は自分の机に2台のコンピュータを置いています。1つはLinuxでもう1つはWindows、それぞれ直接あちこちにネットワークを繋ぎ、うまくやっています。そして私たちはこれからLinuxを、急速に増加しつつあるユーザー層に対して、デュアルプラットフォームモードに、すなわちデュアルブートもしくは2台のコンピュータを持つといった形式で普及させていくことになるだろうと思います。

Dwight：LinuxはCorel従業員のうちのかなりの割合の人々に、メインのOSとして使われると思いますか？

Michael：それにはある程度の期間がかかると思います。おそらく18カ月かかるでしょう。私たちはすべてのアプリケーションを持っているので、それを使わない理由はあまりありません。ほとんどの人々が使うものはWebとオフィススイートとグラフィックスで、それらはみな私たちが扱っているものです。ですから私たちはきっと、Linuxを最大限に利用することになるでしょう。最初は並行して利用されるでしょうが、後に優先的に選ばれるシステムになると思います。私たちがLinuxに直接関わってみて得られた利点は、Corel独自のディストリビューションを作るべき理由がわかったことです。というのは、Linuxはたいへん良いシステムなのですが、セットアップが難しいのです。セットアップの専門家が必要です。私たちはそのすべての面倒を見ます。同様に、ネットワークをブラウズするのは現状ではWindowsのように簡単にはできません。私たちはそれについても面倒を見ます。

私たちは、何があって何が足りないのかがわかります。そして不足しているものはそれほど多くありません。私たちは不足している部分を埋めるつもりです。

プロフィール

Michael Cowpland氏はCorelの創設者であり、現在も社長兼CEO。CorelはCorel LINUX OSを発表、今後もLinuxに力をいれていくという。

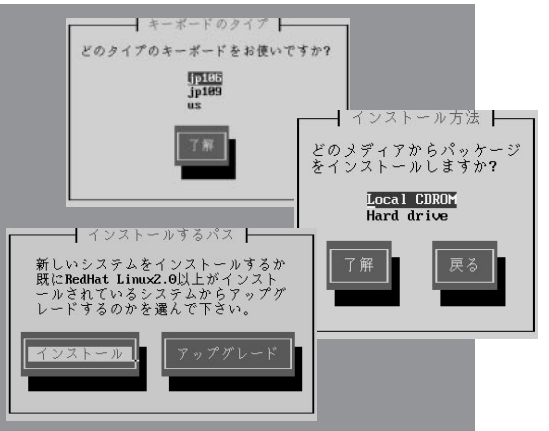
付録CD-ROM Disk2に収録した
LASER5 Linux 6.0フリー版のインストール

CD-ROMブートができる場合は、付録CD-ROM Disk2をCD-ROMドライブに入れ、CD-ROMから起動します。CD-ROMブートができない場合は、DOS、Windows環境でブートディスクを作成する必要があります。この場合、フォーマット済みのフロッピーディスクをAドライブにセットし、CD-ROMのimages\makediskディレクトリに移動して、ここにあるバッチファイルでブートディスクを作成します。通常はmkboot.batファイルで作成するブートディスクでよいでしょう。

通常のブートディスクではサポートされないCD-ROMドライブや、SCSIインターフェイスを使っている場合はmkboot_ext.batでブートディスクを作ります。

CD-ROM、またはブートディスクから起動し、“boot:”のプロンプトが表示されたらEnterキーを押します。プロンプトに対し、“english”と入力し、Enterキーを押すとメッセージが英語になります。

インストーラの操作は、［Tab］キーで項目の移動、矢印キーで選択の移動、スペースキーで選択、Enterキーで決定となります。

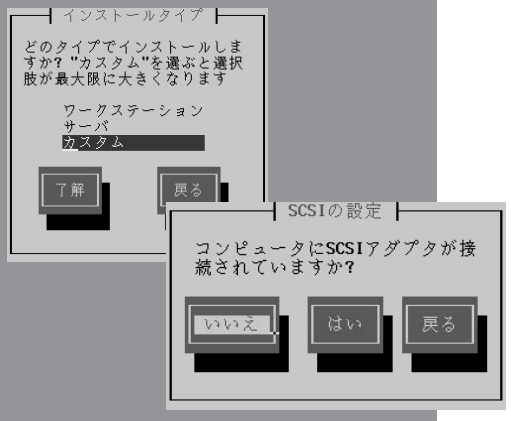


1 キーボード、インストール方法、インストールパスの設定

最初に、使用するキーボードの種類を選択します。日本語106キーボードでは「jp106」を、Windowsキーのある日本語109キーボードでは「jp109」を選びます。英語キーボードでは「us」を選びます。

「インストール方法」の画面では、利用するメディアを選びます。ここでは「Local CDROM」を選択します。

「インストールするパス」では、新規インストールをするか、アップグレードするかを選びます。Red Hat Linux 2.0以降がインストールされている場合は、「アップグレード」を選ぶことができます。ここでは「インストール」を選択することにします。



2 インストールタイプ、SCSIの設定

どのようなアプリケーションパッケージをインストールするか選択します。「ワークステーション」と「サーバ」では、規定のパッケージがインストールされます。「ワークステーション」を選ぶと既存のLinuxパーティションが、「サーバ」を選ぶとハードディスクの全領域が初期化され、パーティション設定からインストールまでが自動的に行われます。「サーバ」を選ぶと、Windowsなど、ほかのOSの領域もすべて消えてしまいますので注意してください。「カスタム」ではこれらの設定を手動で行います。ここではカスタムを選択したものとして説明を続けます。

「SCSIの設定」画面では、SCSIインターフェイスを接続している場合は「はい」を選びます。なければ「いいえ」を選択します。「はい」を選ぶとSCSIインターフェイスが自動検出されます。

3 ディスクのセットアップ（カスタムのみ）

ここでは、ディスクのパーティションとマウントポイントを設定します。Disk Druidとfdiskのどちらかのツールを利用できます。ここでは、Disk Druidを使って説明します。

パーティションは、最低限“/（ルート）”と“スワップ”の2つが必要です。カーネルの置かれる“/boot”を最初に10～20Mバイト作ってもよいでしょう。パーティションを追加するには、「追加」ボタンを押して、マウントポイント、サイズなどを設定します。スワップ以外は、タイプを「Linux native」にします。

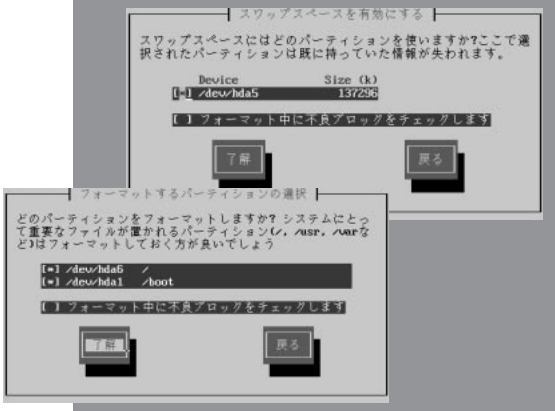
スワップパーティションを追加するときは、タイプで「Linux swap」を選びます。サイズは64～127Mバイトでよいでしょう。

必要なパーティションを作成したら、「了解」を選択します。

スワップスペースを有効にする、フォーマットするパーティションの選択

「スワップスペースを有効にする」画面では、スワップとして使用するパーティションを選択します。“＊”がついているのが利用されるパーティションです。「フォーマット中に不良ブロックをチェックします」に“＊”をつけると、そのパーティションのディスクメディアのチェックをします。

「フォーマットするパーティションの選択」画面では、フォーマットするパーティションを選びます。ここでも、利用するパーティションに“＊”をつけます。



インストールするコンポーネント（カスタムのみ）、インストールログ

インストールするアプリケーションパッケージを選びます。スペースキーでインストールしたいパッケージに“＊”をつけます。「個々のパッケージを選択する」をチェックすると、パッケージの詳細な選択ができます。

このあと「解決されていない依存関係」の画面になることがありますが、そのまま「了解」を選べば、自動的に必要なパッケージがインストールされます。

「インストールログ」の画面では「了解」を押します。インストールの過程は/tmp/install.logにログとして保存されます。

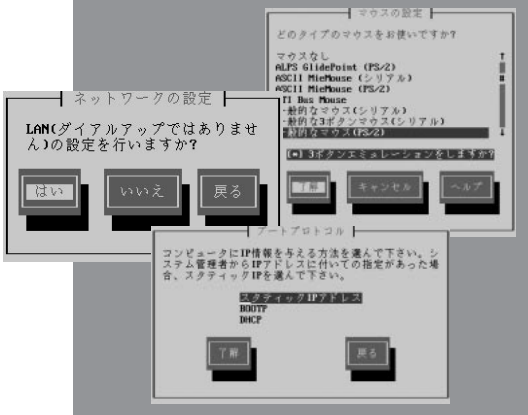
「インストール状況」画面では、選択したソフトウェアをハードディスクにコピーします。進捗状況がグラフ表示されますので、終わるまで待ちましょう。

マウスの設定、ネットワークの設定、ブートプロトコル

マウスの種類を設定します。2ボタンマウスを利用する場合は、「3ボタンエミュレーションをしますか？」にチェックを入れます。これをチェックすると、3ボタンマウスの中ボタンを、左右ボタン同時押しで代替します。

「ネットワークの設定」画面では、LANの設定を行うかどうかを選択します。ここでは「はい」を選択することにします。「はい」を選択すると、ネットワークカードの検出を行います。自動検出できない場合は手動で選択します。

「ブートプロトコル」画面では、IPアドレスをどのように設定するかを選択します。「スタティックIPアドレス」では、手動でIPアドレスなどを入力します。「BOOTP」、「DHCP」では、それぞれの該当するサーバから必要な情報を取得します。ここでは、「スタティックIPアドレス」を選択して進めることにします。

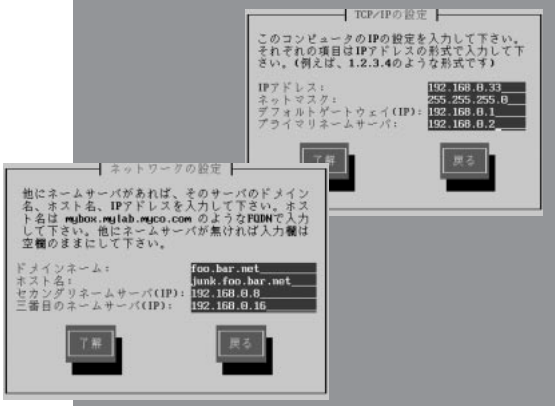


TCP/IPの設定、ネットワークの設定

「ブートプロトコル」画面で「スタティックIPアドレス」を選ぶと「TCP/IPの設定」画面になります。ここでは、マシンのIPアドレスなどを設定します。設定する内容がわからない場合は、ネットワーク管理者に問い合わせてください。

「ネットワークの設定」画面では、ドメイン名、ホスト名を設定します。また、必要に応じて、セカンダリネームサーバ、3番目のネームサーバも設定します。

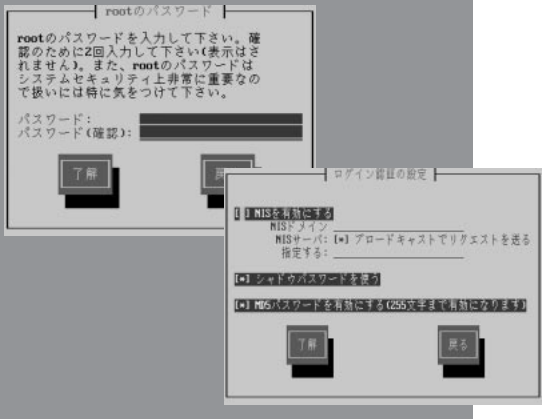
設定する内容がわからない場合は、ネットワーク管理者に問い合わせてください。



タイムゾーンの設定、サービス

「タイムゾーンの設定」画面では、世界標準時からの時差を設定します。日本国内で利用するなら「Japan」を選択します。同じマシンでほかのOSも利用する場合は「ハードウェアクロックをGMTに設定しますか？」にチェックをいれてはいけません。

「サービス」の画面では、Linuxのブート時にスタートさせるサービス（デーモン）を選択します。サービスを選んでF1キーを押すと、そのサービス（デーモン）の説明を英文で表示します。よくわからなければ、デフォルトのまま「了解」を押して次に進んでも大丈夫です。

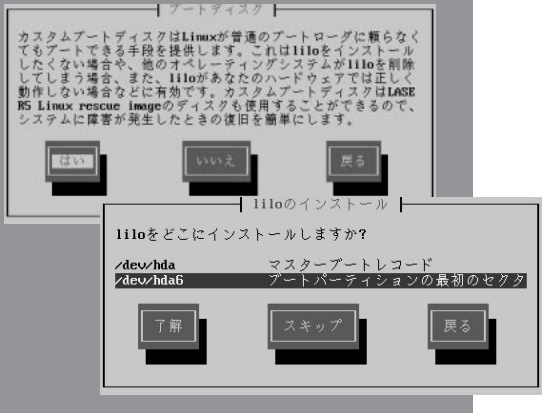


プリンタの設定、rootのパスワード、ログイン認証の設定

「プリンタの設定」画面で「はい」を選択するとプリンタの設定が行えます。

「rootのパスワード」画面では、システム管理を行うルートユーザーのパスワードを設定します。確認のため2回入力する必要があります。なお、セキュリティを確保するため、入力した文字は画面に表示されません。

「ログイン認証の設定」画面では、ユーザーがマシンにログインする際の認証方式を設定します。通常はそのまま「了解」を選べばいいでしょう。NISサーバが設置されている場合はネットワーク管理者の指示に従ってください。



ブートディスク、liloのインストール、ブート可のパーティション

ブート用のフロッピーディスクを作成するかどうか選択します。緊急時に必要になりますので、できるだけ作成しておきましょう。

「liloのインストール」画面では、LILO（LInux LOader）のインストール先を選択します。Linuxのみ、あるいはLinuxとWindows 9Xのデュアルブートをする場合は「マスターブートレコード」を選びます。System Commanderなどのブートセレクタを利用している場合は「ブートパーティションの最初のセクタ」を選びます。

「ブート可のパーティション」画面では、LILOで起動可能にするOSを設定します。Windows 9Xは、デフォルトで設定されている“dos”というラベルのままで問題ありません。

続いてX Window Systemの設定になりますので、画面の指示に従って設定すれば、インストールは終了します。

LASER5 Linux 6.0のアップデート

引き続き、LASER5 Linux 6.0の不具合修正のため、アップデートファイルをインストールします。

LASER5 Linux 6.0を起動し、rootでログインします。最初に、CD-ROMをマウントします。付録CD-ROM Disk2をドライブに入れ、次のようにコマンドを入力します。

```
# mount /mnt/cdrom
```

CD-ROMをマウントしたら次の手順でアップデート作業を行います。

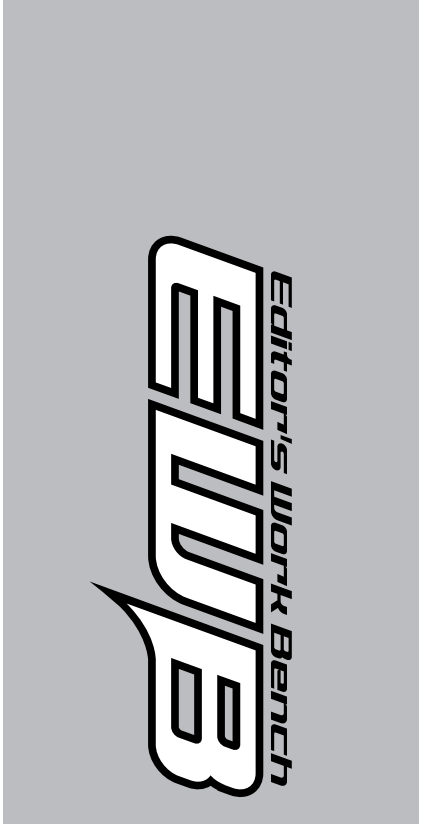
```
# rpm -Fvh /mnt/cdrom/updates/*.rpm
# cp -p /mnt/cdrom/untested/libnsfix-wcsmbs.so /usr/local/netscape
# cd /usr/local/netscape
# patch < /mnt/cdrom/untested/jcommunicator40x.sh.patch
```

作業が済んだらCD-ROMをアンマウントします。

```
# umount /mn/cdrom
```

プロフェッショナルユース組み版システム

EWB



1999年10月、EWB（Editor's Work Bench）システムが、オープンソース・ソフトウェアとして一般公開された。もともとはアスキーが、文書の電子化を推進する目的で社内用に開発したシステムである。開発当初の組版機能は文字原稿の棒打ち出力だけであったが、pTeXや周辺ツールの開発と改良を重ねるにつれ、完全ページアップも可能となり、さらに面付けもできるようになっている。アスキーが出版している書籍の多くがEWBを利用して作成され、雑誌でもスペック表のような定型ページに用いられたりもしている。本誌ではオープンソース化を機に、付録CD-ROMに収録した。

文：**中野ケン**
Text : Ken Nakano
株式会社アスキー 出版技術部

EWBの特徴

EWBは本を作るためのシステム、つまり組版システムである。EWBは、つぎのような特徴を持っている。

- **マークアップ方式**
- **簡単なトリガ**
- **文章とレイアウトの分離**
- **TeXによる組版**
- **PostScriptへの対応**

●マークアップ方式

一般に組版システムは、文章のレイアウト作業をどのように行うかで、DTP（WYSIWYG）方式とマークアップ（バッチ）方式とに大きく分類される。DTP方式は、ディスプレイ上で対話的にレイアウトしていく方式で、Quark Expressや FrameMakerなどに代表される。対して、マークアップ方式は、レイアウト処理用の記号を文章に埋め込み、それをプログラムで一括処理する。TeXやXMLなどがこの方式に属する。EWBもマークアップ方式だ。

DTP方式とEWBでの作業は**図1**のような違いがある。

DTPでは、通常、原稿を編集する編集者と、その文章をレイアウトする制作者とが共同で作業を行う。レイアウト後の文章の修正は、編集者が紙の上で行い、制作者がそれを反映をしていく。現在のように、市場への早急な対応が望まれてくると、このやりとりの手間や時間がボトルネックになってく

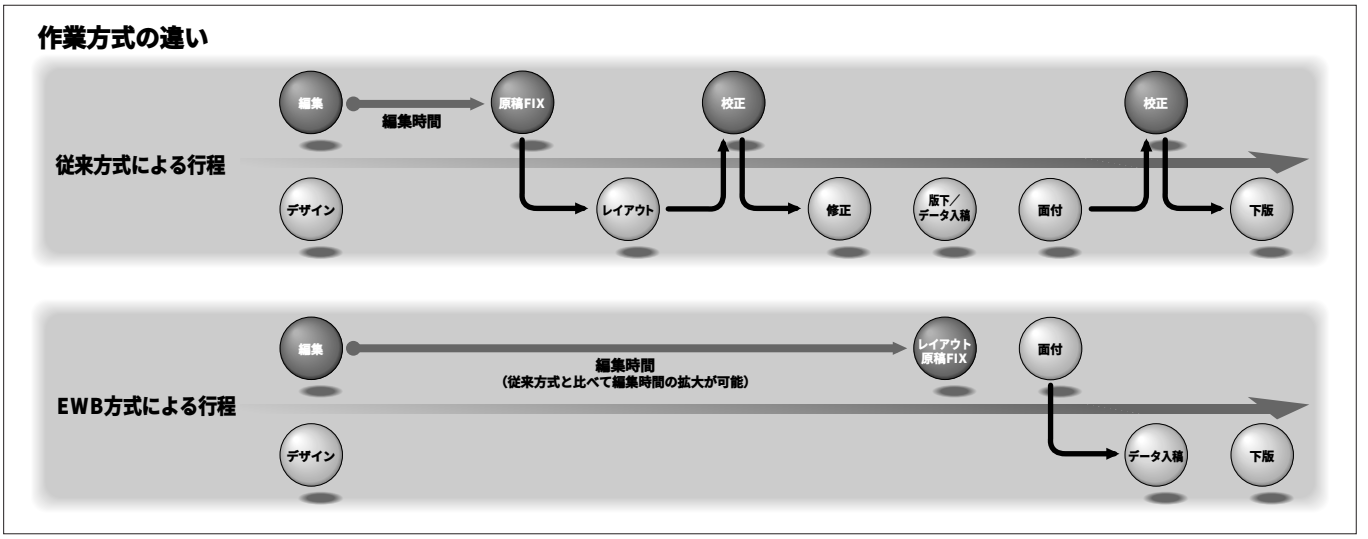


図1 DTPとEWBでの作業の比較

る。

一方、EWBでの作業は、編集者は手もとの原稿ファイルを自分で修正するだけで済んでしまう。やりとりが少ないぶんだけ、手間も時間もかからない。逆に、同じ時間を与えられたのならば、より丁寧に文章を校正する余裕が生まれる。

●簡単なトリガ

メールや文章を書くとき、●で印を付けたり、＊＊で語句を強調したり、マイナス記号を連続させたラインで区切ったりして、相手が読みやすいように工夫することがある。マークアップ方式で文章に記号を付けるのも、これと変わらない。しかし、マークアップ方式は、記号の入力が面倒だったり、数が膨大だったり、規則が細かかったりすることが難点とされる。

そこで、EWBでは誰もが容易に覚えることができ、すぐに利用することができるよう、記号（トリガ）をシンプルにし、規則もワープロで原稿を作成するときとほとんど変わらないようにしている（**表1**、**リスト1**）。

ここでとくに重要なのは、トリガがレイアウト用の指定をするための命令としてでなく、文章の骨格を示すように意味付けられている点である。文章を執筆したり、編集したりするときに、レイアウトのことを気にするのではなく、その文章における、内容上の位置付けを意識するようになっているのだ。メールでの工夫のように、文章の内容を読み手に伝える、いわばコミュニケーション言語としての役割を担っているのである。

そのため、最終的にはDTPソフトで組版されてしまうにしても、EWBのトリガを編集作業に利用し、制作者に意図を伝えるために、EWBのトリガを付けた原稿ファイルを渡す編集者もいる。渡された制作者も、トリガをマクロなどで自動処理するような工夫をして、手作業を軽減させることも可能だ。

なお、**表1**のトリガがすべてなのではなく、必要に応じてトリガを定義できるようにもなっている。

●文章とレイアウトの分離

表1のトリガは文章の意味上の役割

▼リスト1　原稿ファイルの例

```
//i EWB

//ii はじめに
   1999年10月、EWB（Editor's Work Bench）システムが、オープン・ソース・ソフトウェアとして一般公開された。もともとはアスキーが、文書の電子化を推進する目的で社内用に開発したシステムである。開発当初の組版機能は文字原稿の棒打ち出力だけであったが、pTeXや周辺ツールの開発と改良を重ねるにつれ、完全ページアップも可能となり、さらに面付けもできるようになっている。アスキーが出版している書籍の多くがEWBを利用して作成され、雑誌でもスペック表のような定型ページに用いられたりもしている。本を作るためのシステム、つまり組版システムである。

//ii EWBの特徴
   EWBは、つぎのような特徴を持っている。

//k1{
・マークアップ方式
・簡単なトリガ
・文章とレイアウトの分離
・TeXによる組版
・PostScriptへの対応
//}

//iii マークアップ方式
   一般に組版システムは、文章のレイアウト作業をどのように行うかで、DTP（WYSIWYG）方式とマークアップ（バッチ）方式とに大きく分類される。DTP方式は、ディスプレイ上で対話的にレイアウトしていく方式で、QuarkExpressやFrameMakerなどに代表される。対して、マークアップ方式は、レイアウト処理用の記号を文章に埋め込み、それをプログラムで一括処理する。TeXやXMLなどがこの方式に属する。EWBもマークアップ方式だ。
   DTP方式とEWBでの作業は//f100のような違いがある。
   DTPでは、通常、原稿を編集する編集者と、その文章をレイアウトする制作者とが共同で作業を行う。レイアウト後の文章の修正は、編集者が紙の上で行い、制作者がそれを反映をしていく。現在のように、市場への早急な対応が望まれてくると、このやりとりの手間や時間がボトルネックになってくる。
   一方、EWBでの作業は、編集者は手もとの原稿ファイルを自分で修正するだけで済んでしまう。やりとりが少ないぶんだけ、手間も時間もかからない。逆に、同じ時間を与えられたのならば、より丁寧に文章を校正する余裕が生まれる。

//f100 DTPとEWBでの作業の比較

//iii 簡単なトリガ
   メールや文章を書くとき、●で印を付けたり、**で語句を強調したり、マイナス記号を連続させたラインで区切ったりして、相手が読みやすいように工夫することがある。マークアップ方式で文章に記号を付けるのも、これと変わらない。しかし、マークアップ方式は、記号の入力が面倒だったり、数が膨大だったり、規則が細かかったりすることが難点とされる。
   そこで、EWBでは誰もが容易に覚えることができ、すぐに利用することができるよう、記号（トリガ）をシンプルにし、規則もワープロで原稿を作成するときとほとんど変わらないようにしている（//t100、//l100）。
   ここでとくに重要なのは、トリガがレイアウト用の指定をするための命令としてでなく、文章の骨格を示すように意味付けられている点である。文章を執筆したり、編集したりするときに、レイアウトのことを気にするのではなく、その文章における、内容上の位置付けを意識するようになっているのだ。メールでの工夫のように、文章の内容を読み手に伝える、いわばコミュニケーション言語としての役割を担っているのである。
   そのため、最終的にはDTPソフトで組版されてしまうにしても、EWBのトリガを編集作業に利用し、制作者に意図を伝えるために、EWBのトリガを付けた原稿ファイルを渡す編集者もいる。渡された制作者も、トリガをマクロなどで自動処理するような工夫をして、手作業を軽減させることも可能だ。
   なお、//t100のトリガがすべてなのではなく、必要に応じてトリガを定義できるようにもなっている。
```

//i	大見出し
//ii	中見出し
//iii	小見出し
//g{〜//g}	語句の強調
//k1{〜//k1}	箇条書
//c1{〜//c1}	小組
//list{〜//list}	プログラムリスト
//f番号	図の指定と参照
//t番号	表の指定と参照

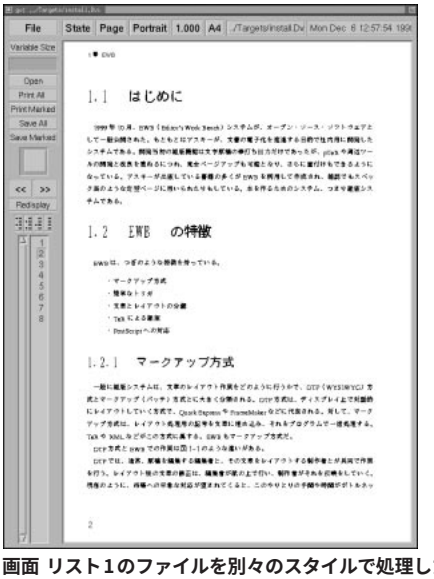
表1 EWBのトリガ

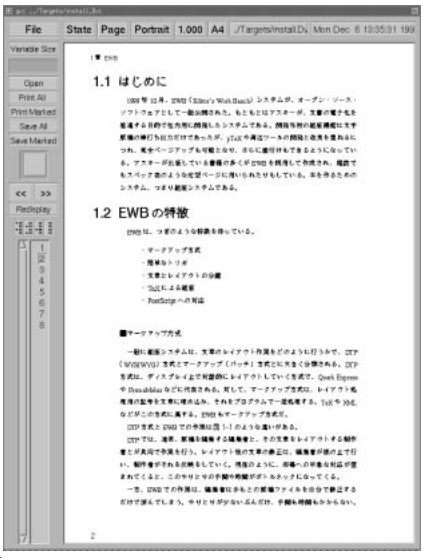
を示しているだけである。そのレイアウト的な意味は何もない。たとえば、「//g」は強調してほしいという以外のことは何も伝えない。それが新ゴシックであろうが、太ゴシックであるかは文章の内容からすれば関係ないから当然だ。トリガがレイアウト上でどのように表現されるのかは「スタイルファイル」で行う。

よく、コンピュータ言語の仕様書などで関数の引数を特別な書体で示すことがある。このとき、原稿ファイルでは「//arg{x//}」のように特別なトリガでその語句を示しておく。すると、最初はイタリックで表示しようと思ったけれど、後からスラント体にすることにした場合も、原稿ファイルはいっさい修正することなく、スタイルファイルの定義を変更するだけで済んでしまう（**画面**）。

DTPのように文章に具体的な指定を行っているのでは、こうはいかない。すべての変更部分を読んで確認し、手で修正する必要がある。DTPは、現実の作業をそのままコンピュータ上で行っているために、アプリケーション操作は簡単だが、効率的に行える部分も少ないのである。

また、マークアップ言語同士には、それぞれで同じ意味を持つ記号を提供していることが少なくない。たとえば、見出しを表す記号としてHTMLには「H1」タグ、LaTeXでは「\chapter」コマンドがある。EWBのトリガを、プログラムによって、それぞれのマークアップ言語の記号に置き換えるだけで、ひとつの文章をさまざまな形式で読者に提供できるわけだ。もちろん、すべての記号が対応するとは限らない。しかし、全部を手作業で変換するよりは、明らかに効率的である。





画面 リスト1のファイルを別々のスタイルで処理した例

●TeXによる組版

TeXはUNIX上で広く用いられているフリーの組版ソフトだ。多くのLinuxディストリビューションにも付属している。数式組版の能力や、強力なマクロ機能、柔軟性などで評価が高い。その一方で、記号付けの規則が細かくて大変、コマンド操作が面倒、スタイルファイルの作成が難しいともいわれる。

EWBでは、文章への指定はトリガによって簡略化している。それをプログラムでTeXの文書形式に変換し、組版をしている（**図2**）。

そして、原稿ファイルの変換から、PostScriptファイルの作成までの操作を対話的に行うための環境を用意している。このインターフェイスを用いると、UNIXのコマンドオペレーションに慣れていないユーザでもスムーズに作業を進めることができる。

スタイルファイルの作成については、基本組、見出し、小組などの書体や級数、字数といった基本的な体裁を対話的に設定できるようなツールを提供している。

なお、EWBではTeXのマクロ機能によって、自動的に分割される囲み罫マクロなどを用意している。たとえば、コラムやプログラムリストの周りを罫線で囲むようなレイアウトの場合、それらが1ページに収まるのならば対応できるソフトは多い。けれども、複数ページにまたがる囲み罫を自動的に分割し、レイアウトするソフトはほとんどない。このときは手作業で対処しなければならず、とくにマークアップ方式ではかなり面倒で負担の大きい作業である。EWBでは、囲み罫を自動的に分割する機能を備えることによって実現している。このように、TeXに強力なマクロ機能を用いて組版処理をカスタマイズして、文章に指定するトリガの数や面倒な作業を少なくすることが可能である。

PostScriptへの対応

EWBでは、最終的な組版データはPostScript形式としてファイルに格納する。WYSIWIG方式の組版ソフトでも最終的にはPostScript形式で出力される。以前は日本語書体などの問題もあったが、近年の出力環境の充実は著

しい。たとえば、以前は出力データは印画紙に出力され、フィルム、刷版の工程を経て、印刷されていたが、現在はフィルムに直接出力するのが一般的だ。そして、刷版に直接出力するCTP（Computer To Plate）の段階になってきている。工程が減るぶん、時間もコストも抑えることができる。もちろん、EWBで組版したデータもCTPで出力可能である。

また、PostScriptデータは、Adobe Distillerを使って、簡単にPDFファイルに変換することができる。PDFファイルは、組版された文章をネットワークやCD-ROMなどで配付するのに最適な形式だ。画像データもそのまま入っているため、JPGやGIFに変換したりの手間もいらない。

今後、少量部数の印刷は、データを直接出力し、製本をするオンデマンドが主流になるだろう。そのためのデータフォーマットはPostScriptかPDFが最有力として認識されている。

EWBのインストールの前に

EWBは、文書処理や組版、プレビューなどをするのに表2のようなプログラムを利用する。これらのうち、まだインストールしていないプログラムがあったら、まず、それらをインストールしておく。

表2のプログラムのうち、とくにsedは注意が必要だ。インストールされているsedがマルチバイト対応版であっても、日本語を正しく扱えるとは限らないからだ。

たとえば、

```
//list{
echo 'ありがとうございました。' | sed
'y/あいう/アイウ/'
//}
```

としたとき、

```
//list{
アリガトウゴザゥマシタ。
//}
```

のように変換されてしまったら、そのsedは日本語処理用には向かない。本来、期待する動作は、

```
//list{
アりがとウござイました。
//}
```

である。後者のようになっていなかったら、jsedをインストールする必要がある。

なお、macutilsをインストールしてある場合、/usr/bin/tomacプログラムがEWBのtomacに上書きされてしまう。macutilsのtomacコマンドも残しておきたければ、EWBをインストールする前に、

```
# cd /usr/bin
# mv tomac tomac.macutils
```

のようにしておく。

EWBのインストール

EWBのソースファイルは、本体、ライブラリ、GUIライブラリの3つに分けて提供されている。

●本体のインストール

本体には、EWB文書ファイルをLaTeX文書ファイルに変換するフィルタや、コマンドラインでの操作環境プログラムなどが含まれている。

インストールは、つぎの手順で行う。configure時、prefixに指定するディレクトリは、TeXをインストールしたの

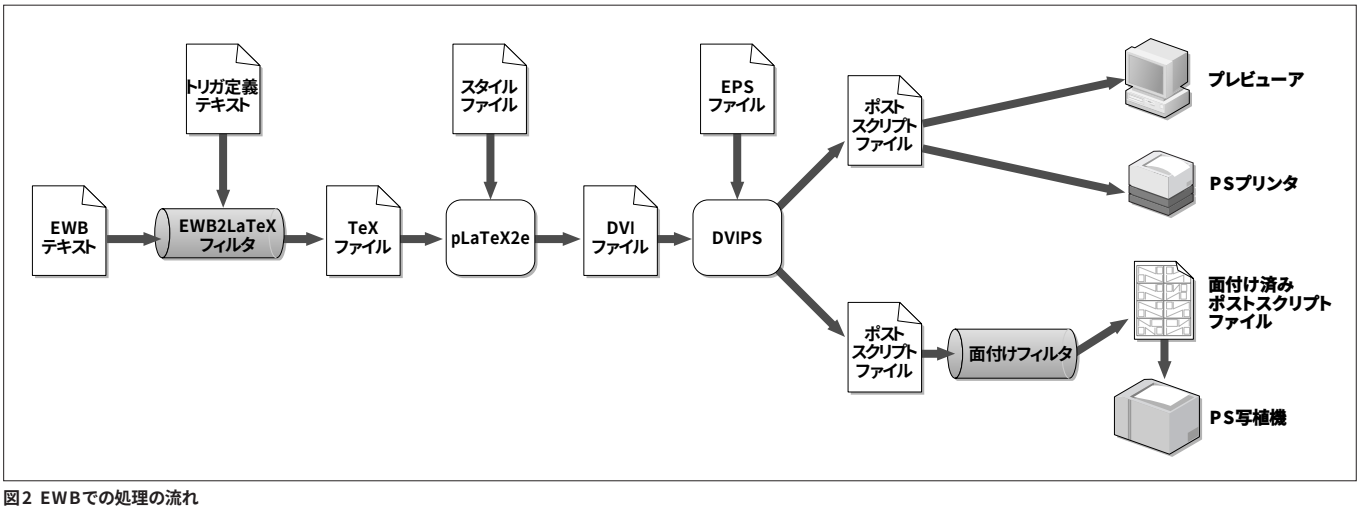


図2 EWBでの処理の流れ

と同じにする。わからなければ、

```
# kpsewhich texmf.cnf | sed -e
's!/share/texmf/web2c/texmf.cnf!!'
```

として表示された位置を指定すればよいだろう。インストールはこのディレクトリをconfigureで指定して次のように行う。CD-ROMは、/cdromにマウントされているものとする。

```
# cd /usr/src
# tar zxvf /cdrom/EWB/ewb-3.1-
R1.tar.gz
# cd ewb-3.1-R1
# ./configure --prefix=/usr
# make
# make install
```

●ライブラリのインストール

ライブラリのファイルには、EWBで用いるLaTeXマクロ、欧文のPost Scriptフォント、和文のTFMフォントやVFフォントなどが含まれている。

インストール手順は、本体とほとんど同じだ。和文のVFフォント（21書体）は最後の「make install」時に作成されるので時間がかかる。また、作成するVFファイルだけで、100Mバイト程度のディスク容量を必要とすることに注意する。

```
# cd /usr/src
# tar zxvf /cdrom/EWB/ewblib-
3.1.tar.gz
# cd ewblib-3.1
# ./configure --prefix=/usr
# make
# make install
```

●GUIのインストール

EWBのGUIは、レイアウト作業を対話的に行うためのものでなく、文書処理操作を対話的に行うための環境だ。Tcl/Tkを利用して実現している。メニューやボタンなどで日本語を用いているため、日本語化されたTcl/Tkがインストールされていなければならない。

X Window System	XFree86等（X11R6以降）
X漢字端末	kterm
TeX	pTeX p2.1.8（EUCバージョン）
LaTeX	pLaTeX（EUCバージョン）
索引整形	mendex 2.4c（EUCバージョン）
DVI->PS変換	dvipsk 5.78 p1.4c
PSインタプリタ	ghostscript（日本語対応版）
PSビューア	ghostview または gv
漢字コード変換	nkf
文字置換フィルタ	jsed（日本語対応sed）
perlインタプリタ	perl5（日本語対応版）
テキストエディタ	jvim（日本語対応 vi） xemacs-canna
Tcl/Tkインタプリタ	tcl/tk 8.0（日本語対応版）
かな漢字逆変換	kakasi

表2 EWBシステムが利用するプログラム

インストール方法は、本体やライブラリなどと同様である。

```
# cd /usr/src
# tar zxvf /cdrom/EWB/ewbgui-3.1-
R1.tar.gz
# cd ewbgui-3.1-R1
# ./configure --prefix=/usr
# make
# make install
```

EWBの動作確認

インストールが終了したら、動作確認も兼ね、EWBのマニュアルを処理してみよう。EWBのマニュアルは、ewb-handbook.tar.gzという名前で付録CD-ROMに入っている。

まずは、このファイルを展開する。

```
# tar zxvf /cdrom/EWB/ewb-
handbook.tar.gz
# cd ewb-handbook
```

文書を組版するには、GUIシェルを起動する。

```
# guishell
```

起動すると、EWBの文書ファイルが一覧になって表示される。つぎに、1-1.docの行を選択し、プレビューボタンを押してみよう。すると、1-1.docで使っているEPSの変換、LaTeX文書への変換、組版、PostScriptファイルへの変換が自動的に行われ、ディスプレイに文書が表示される。

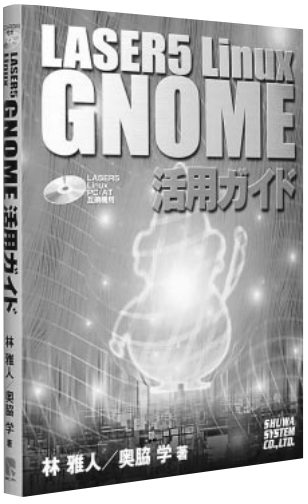
EWBコンソーシアム

EWBの公開と同時にEWBコンソーシアムが設立されている。コンソーシアムは、EWBの普及を通じて、電子出版のための技術の改良や標準化、情報の共有などをはかり、既存出版と電子出版の発展を目指している。詳細については、

http://www2.ascii.co.jp/EWB/

を参照してほしい。

Books



LASER5 Linux GNOME活用ガイド

林 雅人、奥脇 学 著
秀和システム
B5変形判／246ページ／CD-ROM付き　本体価格 2600円

GNOMEは、GNUプロジェクトによって開発されているフリーソフトウェアで、多くのLinuxディストリビューションのデスクトップ環境として、KDEとともに標準的に採用されている。本書は、ユーザーが一番多く接するであろうウィンドウ画面（GNOME）の使い方を中心に、LASER5 Linux 6.0の基本的な設定方法を解説している。

GNOMEは、Windowsに近い操作性を持っているためわかりやすいが、本書のメニューの設定やパネル上で動作するアプレット、GNOMEアプリケーションの使用方法などを参考にして、ユーザーがカスタマイズすることでもっと使いやすくなるだろう。

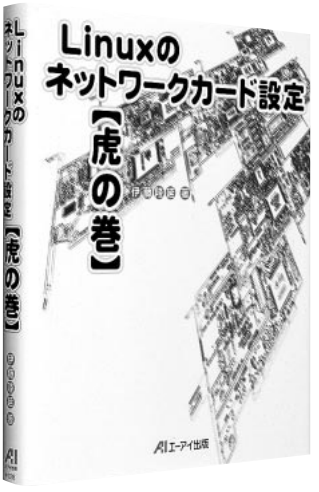
付属CD-ROMにLASER5 Linux 6.0を収録し、インストール手順も詳しく書かれている。GUIで操作できる設定ツールのLinuxconfや、コントロールパネルを使ってハードウェア、プリンタ、ダイヤルアップ、ネットワークなどを設定でき、Linux初心者にもお勧めである。

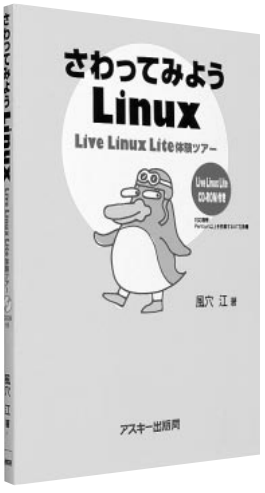
Linux ネットワークカード設定【虎の巻】

伊藤隆延 著
エーアイ出版
B5変形判／362ページ／CD-ROM付き　本体価格 3200円

ネットワークカードをWindowsで使う場合に問題が起こることは少ない。Windowsが標準で多くのドライバをサポートしているし、ほとんどのネットワークカードにはドライバが付属しているからだ。しかし、Linux用のドライバはハードウェアメーカーではなく、Linuxのコミュニティが動作を解析して開発しているため、いろいろとトラブルが発生することがある。

本書は、主要な12のディストリビューションを使って、約30種類のネットワークカードについて動作検証を行っている。Linuxのインストーラが自動で認識しない場合には、必要なドライバをFTPサイトなどから手に入れ、ドライバをコンパイルして動作させる方法も記述されている。ネットワークカードの知識だけではなく、ドライバのインストールを行うことで、Linuxのドライバの仕組みや、ディストリビューションごとの違いが理解できるだろう。なお、付属CD-ROMには、日本ではあまり馴染みのないLinux Mandrake 6.1を収録している。





さわってみようLinux—Live Linux Lite体験ツアー

風穴 江 著
アスキー
A5判／128ページ／CD-ROM付き　本体価格1280円

あなたのまわりにも、「Linuxをいじってみたいけれど、インストールがめんどくさそうで…」という人がいることだろう。そんなパソコンユーザーにぴったりの本が登場した。Linux入門書が星の数ほど出ている中で、Linux業界の気鋭ライター兼編集者として活躍中の著者があえて本書を執筆したのは、伊達や酔狂ではない。本書の付属CD-ROMに収録された「Live Linux Lite」（メディアラボ社製）は、インストールが一切不要で、CD-ROMから直接起動できるという特異なディストリビューション。本来は緊急用やテスト用などの目的で開発されたものだが、体験用としても最適である。本書の内容は「Linuxの世界へ体験ツアーに出かける」という趣向。実際にLinuxを操作しながら、LinuxあるいはUNIXの考え方の基本をやさしく解説している。巻末には主要ディストリビューションや入門書のガイドも掲載されており、「本書以後」への配慮も万全だ。

Linuxで構築するファイアウォール

サンモアテック
OSS研究会 著
セレンディップ
B5変形判／272ページ／CD-ROM付き
本体価格 3200円

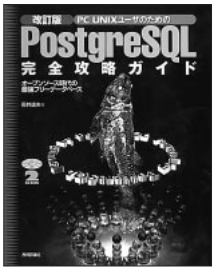
Macintoshで徹底活用 LinuxPPC

三浦一則 著
毎日コミュニケーションズ
B5変形判／320ページ／CD-ROM付き
本体価格 2700円

Linux論文作成術 ーGNU Toolを使いこなすー

臼田昭司、伊藤 敏、井上祥史 著
オーム社
B5変形判／280ページ
本体価格 2800円

改訂版PC UNIXユーザのための PostgreSQL完全攻略ガイド



石井達夫 著
技術評論社
B5変形判／388ページ／CD-ROM付き
本体価格 3200円

らぶらぶLinux2
Windowsと使いたい人のための混在環境ガイド

西村めぐみ 著
ソシム
B5変形判／344ページ／CD-ROM付き
本体価格 2460円

10分でできる！Linuxサーバ

伊藤幸夫 著
エーアイ出版
B5変形判／256ページ／CD-ROM付き
本体価格 2500円

最近売れてるLinux書籍は？

まず、この冬熱く（厚く？）盛り上がっている、オブジェクト指向のプログラム関係の2冊から紹介しましょう。1冊目は、本誌で赤松氏による記事が連載されているRubyのバイブル『オブジェクト指向スクリプト言語Ruby』（まつもとゆきひろ著、アスキー出版局刊、ISBN4-7561-3254-5）がブッチギリの絶好調です。私にはあまり詳しいことまではわかりませんが、プログラムに興味のある方には必読の1冊との評判です。さらに、「ガンマ本」の名で知られる『オブジェクト指向における再利用のためのデザインパターン 改訂版』（Gamme、Helm、Johnson、Visaides共著、本位田 真一、吉田和樹監訳、ソフトバンクパブリッシング刊、ISBN4-7973-1112-6）も好調な売れ行きです。不況といわれる昨今にもかかわらず、4000円台という値段をものともせず売れているこの2冊を見ていると、やはり良書は売れるのだな、と久々にうれしい気持ちにさせられます。もっと、このような硬派な書籍の登場に期待したいところです。

さて、肝心のLinuxの話題となると…、これといった動きがないというのが本音です。ただ、これまた本書で連載のあるフリーのデータベースPostgreSQLの使いこなしをあますところなく紹介している『改訂版 PC UNIXユーザのためのPostgreSQL完全攻略ガイド』（石井達夫著、技術評論社刊、ISBN4-7741-0890-1）も、初版と同じくらい売れ行きに勢いがあります。この書籍は、しばらくベスト10上位の座にいることは間違いないでしょう。

最後に、今年の秋にあった悲報をお伝えしておきましょう。Advanced Programming in the UNIX EnvironmentやTCP/IP Illustratedなどのベストセラーなどで知られる、W. Richard Stevens氏の突然の死です。

Stevens氏は、これまた名著として知られるUNIX Network ProgrammingシリーズのVol.3を執筆中にこの世を去りました。コンピュータ業界に偉大な業績を遺した彼の死は、世界中のコンピュータユーザーから惜しまれています。ただ、未完となったVol.3は、関係者により2001年には完成する予定で、トッパンより翻訳書籍も発行される予定とのことです。しかし本当に惜しい人材を亡くしました。ご冥福をお祈り申し上げます。

（書泉ブックドーム 古田島）

読者の声

俺にもいわせろ！

みなさん、お元気ですか。担当は、先日、ばっちりと風邪を引き2日間も寝込んでしまいました。その後、編集長にうつしたらすっかり元気になりました。今月も、元気にまいりましょう！

1月号「ディストリビューション最前線」へのご意見

1月号の特集は読みごたえがあった。ここ2カ月で3つのディストリビューションを購入して試したが、初心者にはまだまだ難しい面が多い。特に導入後の設定については、標準から少しでもはずれると「ドロ沼」にはまってしまう。

ローカルデバイス（プリンタ、PPP、ネットワーク）の基本的な設定の特集をお願いしたい。

（大阪府　永田 誠一郎さん）

1月号50ページにあるユーザー数調査結果を見て、Plamoユーザーの少なさに奮起して、ハガキを出してしまいました（笑）。

べつに、ほかのディストリビューションと争うつもりはないのに、自分の使っているディストリビューションのユーザーがほかに比べて少ないと、何かくやしく感じるのはなぜだろう？

次号の付録に、ぜひSlackware 7.0をお願いします。

（東京都　西村大助さん）

自分に合ったディストリビューションを選ぶというのは、とても難しいですね。好みや機能だけでなく、用途やマシンの構成も考えて、ベストなものを選んでください。36ページからの、新着ディストリビューション紹介もぜひご覧ください。

1月号「マルチブート完全制覇！」へのご意見

1月号の特集では、Windowsとの共存を中心に取り上げていますが、FreeBSD＋Linuxや、Solaris、BeOS、OS/2なども制覇しなければ完全でないような気がします。

（群馬県　山本秀之さん）

私のマルチブート法は、ハードディスクを交換することです。20Gバイトのハードディスクが2万円で、交換用のBOXが1000円で手に入るので、その方法をとっています。昔は20Mバイトのハードディスクを20万円で買ったのですが……。

（滋賀県　林 正之さん）

山本さんのおっしゃるとおり、そこまで網羅できればベストなんですが。ご意見、参考にさせていただきます。

編集部の共有マシンでも、リムーバブルハードディスクを利用しているものがあります。この方法だと、難しいことを考えないで、どんなOSでもインストールできるので便利ですね。

南極でもLinux!

昨日、たまたまLinux magazine 12月号を買ってぺらぺら見てましたら、Red Hat Linux 6.1（英語版）のインストール記事、「タイムゾーンの設定」の項目に、「余談ですが、南極大陸の昭和基地も選べるのには驚きました。Linuxユーザーはいるのでしょうか」と書かれているのに驚きました。

なんということをおっしゃるのでしょう！　これこれ、ばかにしてはいけません!!!　なんて。;-P

メールやDNSなどのサーバも、'98年からLinuxに代わりましたし、当方では、ある観測用に、数年前からQNXやLinuxを必要に応じて投入しています。先日も、カーネル 2.2.13にした最新LinuxのRAIDディスクを船に積み込んだばかりです。ほかのプロジェクトでも、使っているところはあるでしょうね（北欧の辺鄙な田舎での地磁気観測などでも、一部RT Linuxが走っていますよ）。間違ってもMSは使いません（笑）。

無償配布のSolarisに手を出したくなったりもしますが……やめときましょう。

じつは、観測にはUTC（協定世界時）しか使わないので、私には昭和基地のタイムゾーンは不要ですが、メールサーバなどはSyowa Local Timeの設定でしょうね。わざわざそのためにRed Hat6.1

に入れ替えないでしょうけど……。でも、選べるようにしているというのは、昭和のユーザーは喜ぶでしょうね。

でも、その設定は正しいのだろうか……（Syowa LT = UTC + 3hoursです）。また、South Poleも設定は合っているかな？一読者のつぶやきでした。

（国立極地研究所　行松さん）

昭和基地でもLinuxが使われていたのですね。学術研究の分野でも、Linuxが活躍しているとは聞いていたのですが、担当の勉強不足でした。国立極地研究所、南極観測のWebページ（http://jare.nipr.ac.jp/）には、南極大陸の紹介、第41次観測計画、南極観測船「しらせ」の現在位置や動向などが掲載されています。担当も、さっそく勉強させていただきました。昭和基地では、今の時期は日が沈まないのですね。行松さん、貴重な情報をどうもありがとうございました。

編集長に、昭和基地の取材はどうでしょうと打診したところ、「俺が行く」という答えが返ってきました。

他人にはわからぬこの悩み

「ニューテクノロジー from SGI」の筆者と同姓同名なので、メーリングリストとかには本名が出せません。プアーな知識しか持っていないので、間違えられたら大変です。

「viはじめました」は最高ですね。おかげさまで、以前できなかった置換などができるようになりました。

今度は、「正規表現はじめました」でも始めてもらうと助かります。

（千葉県　鈴木大輔さん）

偶然とは恐ろしいものです。鈴木さん（お葉書をくださった鈴木さんです）も災難（？）でしたね。

viをはじめとして、LinuxやUNIX系OSでは、正規表現が使えるツールがたくさんあります。正規表現って、とても便利なんですが、ちょっと理解しにくいのが難点です。213ページからの「Rubyで行こう」では正規表現を解説しています。もちろん、Rubyもよろしくお願いします。

rpmの互換性問題

Red Hat、Vine、LASER5、etc…、RPM系のディストリビューションといっても、rpmファイルに互換性がないので非常に使いづらいのは何とかならないのでしょうか……（Slackware？Debian？）。

（福岡県　福島大輔さん）

libc、glibcのバージョンなど、ディストリビューション間の差異によって、rpmファイルが違うのは不便ですね。少々面倒ですが、gccなどの開発ツールをインストールしていれば、src.rpmから、システムに合ったrpmファイルを作ることができます。src.rpmをインストールして、/usr/src/RedHat/SPECSディレクトリで、

```
# rpm -bb hoge.spec
```

とすると、/usr/src/redhat/RPMSの中のディレクトリに、rpmファイルが作られます。ディストリビューションによっては、ディレクトリ名のredhatがturboなどと異なることもあります。

My Home for Linux

現在、家を新築中！　全室にLANケーブルを引きました。

（東京都　吉村昌晃さん）

すばらしいですね。電力容量とコンセントの位置、数も重要です。よく使うコンセントって1カ所に集中するんですよね。

幸福量保存の法則

関係ないことだけど、この前、ビックカメラでプリンター買ったら“当たり”がレシートに出てタダになった。お金そのまま返されて、すっげーウレシイ反面、持って帰るのがなんか罪悪感……。その分、パチンコで負けたけど。

（埼玉県　Rin ♥ さん）

世の中って、そういう風にできてるんです。:）でも、ラッキーなのはよいことです。

98でもLinux

NECのPC-9800シリーズで使用できるLinuxはないでしょうか。

（東京都　守矢優二さん）

Slackwareベースの Plamo Linux（http://www.linet.gr.jp/~kojima/Plamo/）が動作します。PC-98シリーズに移植をしているKMC（京大マイコンクラブ）のLinux/98ページも参考になるでしょう（http://www.kuis.kyoto-u.ac.jp/~kmc/proj/linux98/）。

インストール道を極める

実験用の自作マシンを1台決めて、それにいろんな種類のディストリビューションをインストールしては楽しんでいます。Linuxはインストールが楽しいです。ノートには満足のいくインストールが出来たことがないです。

（山口県　花村義一さん）

「満足のいくインストール」という言葉に奥の深さを感じました。いくつものディストリビューションを試せるのもLinuxの面白さのひとつですね。