第 1 回 UNIXゼミ

指導：長谷，チーフ：上川，サブチーフ：吉田，迫田

平成 13 年 5 月 10 日

1 UNIX とは

1.1 歴史的背景

UNIX とは，昔 AT&T で開発されたマルチユーザ，マルチタスクの OS です．現在 UNIX としては，The Open Group[1]が出している “The Single Unix Specification” に準拠していると認定された OS の事が一般的です．これとは別に UNIX の世界では，POSIX[2]という国際的な標準規格もあり，それに準拠すると，POSIX[3]準拠であると宣伝する権利が与えられます．

知的システムデザイン研究室で主として利用されている OS は，Debian[4] GNU[5]/Linux[6] です．Debian としては，GNU プロジェクトの成果物である UNIX 風[7]のアプリケーションプログラムを POSIX 準拠を一つの目標として開発されている Linux 上で動かしています．

1.2 Windows や，MacOS ではない

UNIX は，Windows とも MacOS[8]とも違います．これらの OS は根本的に，Windows も MacOS も，シングルユーザのシングルタスクの OS です．シングルユーザ，シングルタスクとは，ある一瞬に利用している人は，一人であり，さらに，この設計による物にはコンピュータの利用者を，そのコンピュータの生涯で，唯一一人であると前提にしているものもあります．UNIX は，これに対して，マルチユーザ，マルチタスクのシステムで，同一システム上で同時に複数の人が作業する事を考慮した設計になっています．

マルチユーザ，マルチタスクのシステムにはセキュリティーの概念が必要になって来ますが，セキュリティーも過度に複雑にならないよう，シンプルに設計されており，そのシンプルさゆえに，十分なセキュリティ管理のしやすさを Linux では実現しています．

UNIX では GUI には X(図 1) を利用しています．しかし X は，UNIX にとって必要不可欠な物ではないので，取り外したり，違うものを使ったりできます．例えば，二次元の概念が嫌な人は，例えば Berlin[9](図 2) を使うという選択肢があります．また，GUI によるシステムが無い場合には CUI (図 3) によるオペレーションを行います．

---

[1] http://www.opengroup.org
[2] Portable Operating System Interface
[3] SGI の IRIX や，Compaq の Tru64 UNIX，SCO UnixWare，LynxOS 等が POSIX 準拠であるものの例である．
[4] http://www.debian.org
[5] http://www.gnu.org
[6] http://www.kernel.org
[7] 厳密には POSIX 準拠では無い
[8] MacOS X は UNIX
[9] http://www.berlin-consortium.org

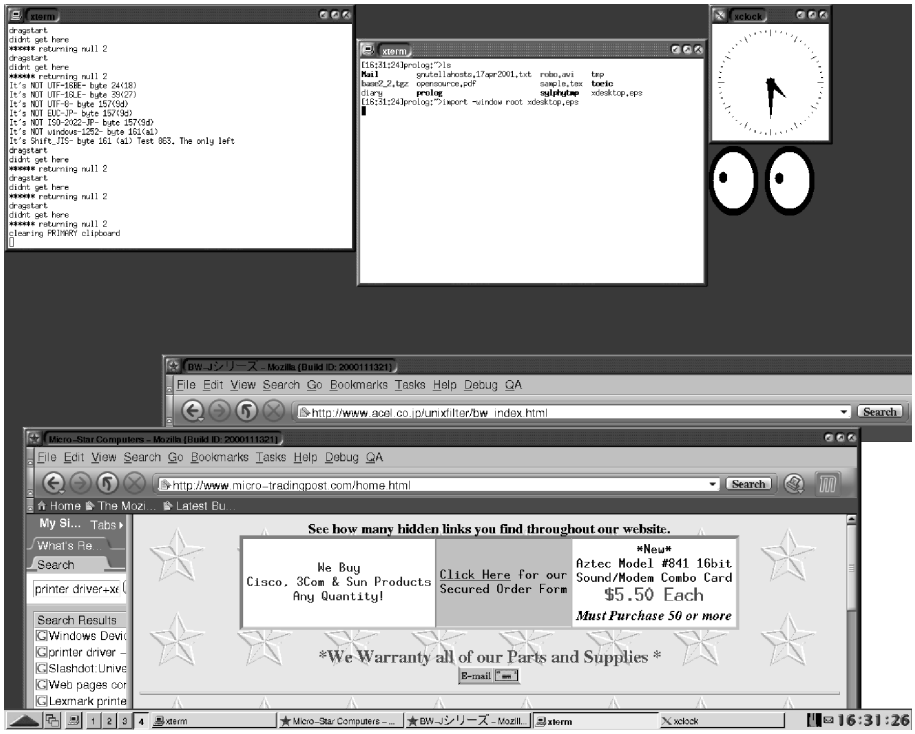

図 1: X のデスクトップ

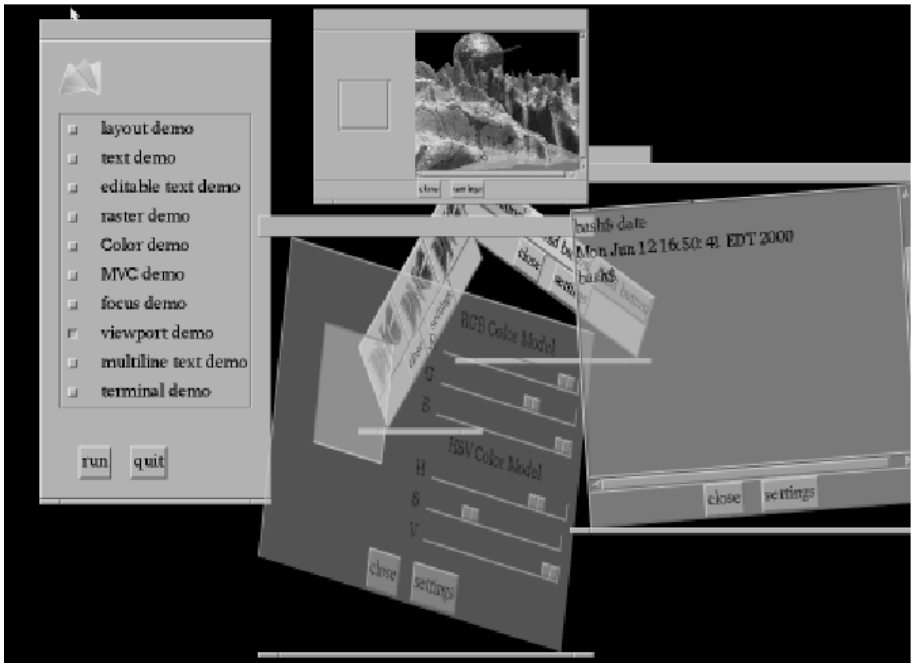

図 2: Berlin のデスクトップ

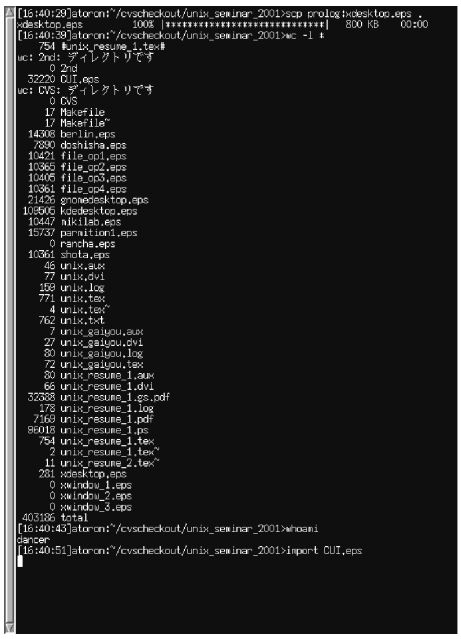

図 3: CUI の画面の例

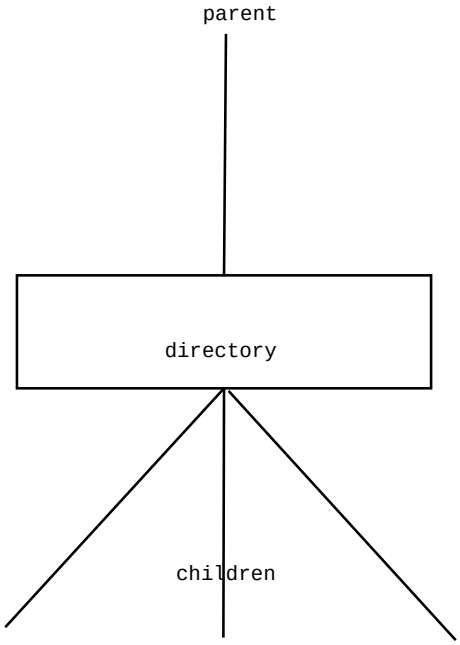

図 4: 再帰的ディレクトリ構造

1.3 なぜ UNIX？

UNIX は，再帰的なディレクトリ構造を基本とした 4，簡素な基礎概念に基づいて発展した OS で，全体の構造の見渡しが良く，Windows や MacOS よりもわかりやすい構造になっています．また，柔軟性が高く，OS 自体をコンパクトにおさえる事もでき，数値計算に専念するようなシステム構成にすることもできます．Windows をいれようとした場合には，GUI を入れないという選択肢は不可能ですが，UNIX では，入れないという選択肢も可能です．

コマンドラインで大抵の事はできるようなシステムなので，遠隔操作をするためには GUI の画像を転送するという負荷の高い方法はとらなくても済みます．

1.4 なぜオープンなアーキテクチャを利用するか

何がどうなっているかという仕様が分かる事は重要です．実際に理系の者として，実験機材に何がどうなっているのか全く分からないようなブラックボックスを利用する事は避けるべきです．

ブラックボックスでないがゆえに，システム自体に問題があっても解決することができます．研究目的で実験する場合，高負荷となる計算など，普通の状況では利用しないような環境で利用する事があるので，OS の実装にバグがありシステムとして修正が必要な場合がおこりえます．それが些細なものであっても，ブラックボックス的であると修正する事が不可能となってしまいます．オープンソースなシステムでは，そのソースを修正する事も可能な場合が多く，迅速な問題解決をはかることができます．[10] ．

[10]近年とみに利用者が増えて来ている Linux は，頻繁にバグが修正されています．自分でバグを修正する機会と言うのはもはや無いかもしれないです．一社の社内で，金でやとわれて，そのソフトを実際には自分で使っていないような人々だけでバグを修正していこうという事業形態では，難しい時代になったのでしょうか．

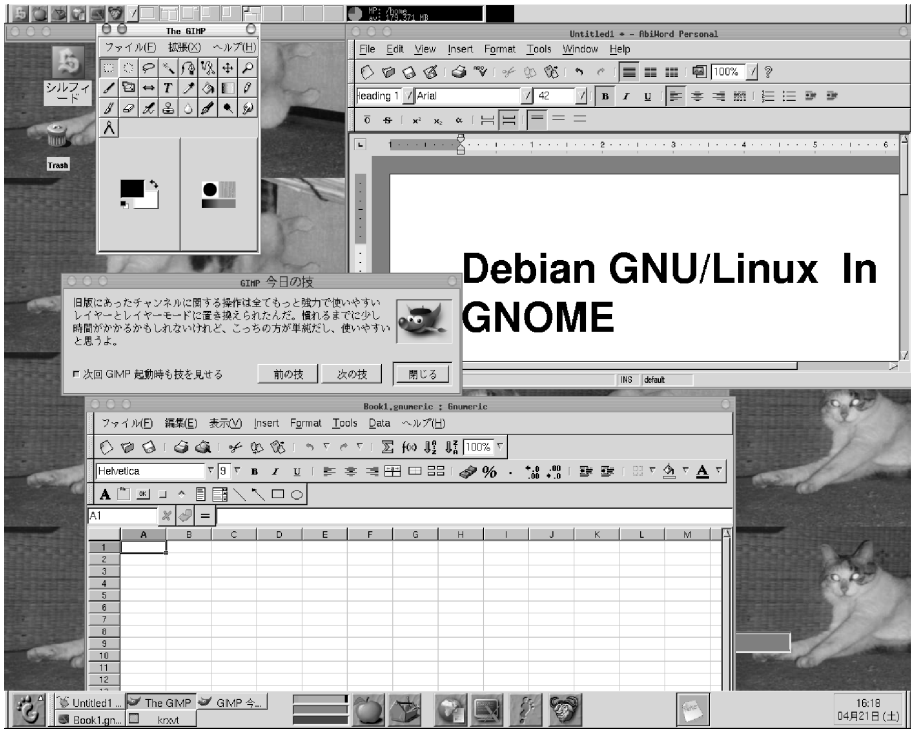

図 5: GNOME のデスクトップの例

1.5 なぜ Linux？

Linux のように無料で十分な開発環境まで手に入るシステムは，他にはあまり類を見ません[11]．そしてオープンなアーキテクチャにもとづいており，使えば使いこむ程よくなるシステムになっています．

Linux に対して，Linux がテキストモードの CUI でしか使えないというふうに誤解している人もいるようですが，Linux では GUI の操作が可能です．日常的に Linux を使っている人で，コマンドラインしか使っていないというような人は少ないです．一般に X というシステムで画像を表示し，GNOME[12](図 5) や KDE[13](図 6) 等の統合環境で作業を行えるようになっています．

もう一つの誤解として，Linux では，日本語が使えないのでは無いのだろうかという誤解もあります．それは，全くの嘘です[14]．最近の Linux 上での日本語の扱い方に関しては，Debian JP プロジェクトの成果などにより，飛躍的に向上しました．

1.6 文字コードについて

日常的に UNIX を利用する時に引っかかりやすい問題として，UNIX と Windows と MacOS では，文字列の表現形式が微妙に違うという事があります．

[11] ここでは，Linux を Debian GNU/Linux を例として考えています
[12] http://www.gnome.org
[13] http://www.kde.org
[14] データの表現形式の問題や，文字コードの表現形式の差異のため，Windows 等とのデータ交換には支障がある場合はあります．

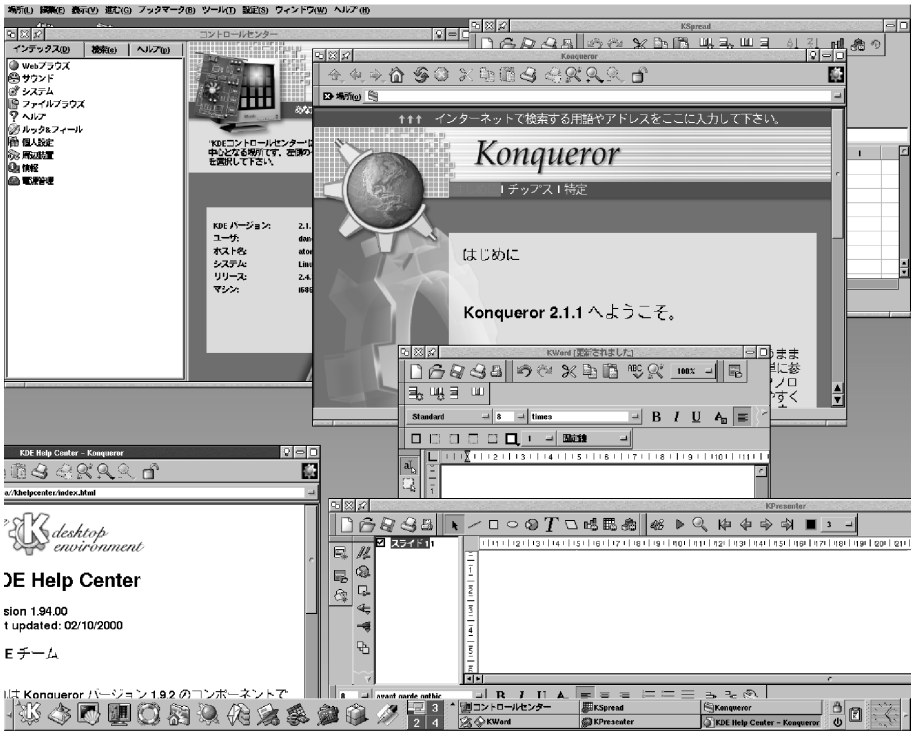

図 6: KDE のデスクトップの例

まず，改行に利用されるコードが UNIX では LF[15]であるのに対して，Mac では CR[16]で，Windows では CR + LF が連続になっています．

改行コードに加え，漢字コードは，UNIX では EUC-JP によるエンコードが一般的であり，Windows や Macintosh では，SJIS によるエンコードが一般的に使われています．何も考えずにそれぞれの形式を扱ってくれるエディタ[17]を使う場合は良いのですが，気を付けましょう[18]．

2 UNIX で行く unixmachine 一周の旅

2.1 login

ssh にて unixmachine に login しましょう．

```
yourname@unixmachine:~$
```

と表示されるはずです．また~$は指定のログイン名を持つ一般ユーザのホームディレクトリの絶対パスを示します．

2.2 自分を知る

まず，自分が何者で何処にどんなファイルがあるのかを把握しなければなりません．

[15]文字コード 10
[16]文字コード 13
[17]emacs は自動でだいたいのファイルは認識してくれます
[18]例えば秀丸は，自動認識はしてくれないようです

2.2.1 whoami

whoami と入力してください．自分が誰なのかがわかります．

```
sakota@unixmachine:~$ whoami
sakota
```

2.2.2 pwd

pwd と入力してください．自分が今どの場所にいるのかがわかります．現在は各ユーザ自身が自由にファイルやディレクトリを作ったり，削除したりできるホームディレクトリと呼ばれる所にいまるはずです．

```
sakota@unixmachine:~$ pwd
/home/sakota
```

2.2.3 ls

ls と入力してください．あなたのホームディレクトリにどんなファイルがあるのかを知ることのできるコマンドですが，現在はまだファイルやディレクトリは存在していません．

```
sakota@unixmachine:~$ ls
sakota@unixmachine:~$
```

2.3 ファイルの操作

2.3.1 mkdir 名前

ここでディレクトリを作成してみましょう**mkdir text** と入力してください．あなたのホームディレクトリに test という名前のディレクトリを作成します

```
sakota@unixmachine:~$ mkdir text
sakota@unixmachine:~$ ls
test
```

2.3.2 cd

別のディレクトリへ移動してみます．**cd test** と入力すると別のディレクトリ（ここでは test）へ移動できます．pwd　で場所を確認してみるとよいでしょう．

```
sakota@unixmachine:~$ cd test
sakota@unixmachine:~/test$ pwd
/home/sakota/test
```

上のディレクトリへ行きたい場合には**cd ..** と入力すると一つ上のディレクトリへ移動できます．上に移動してください．

```
sakota@unixmachine:~/test$ cd ..
sakota@unixmachine:~$
sakota@unixmachine:~/test$ pwd
/home/sakota
```

また**cd /**と入力すると”/”(root) まで移動できます.

```
sakota@unixmachine:~$ cd /
sakota@unixmachine:/$ pwd
/
```

ホームディレクトリに行くには**cd** と入力する.

```
sakota@unixmachine:/$ cd
sakota@unixmachine:~$ pwd
/home/sakota
```

2.3.3 ディレクトリの削除

rm -r test と入力してください. test ディレクトリが削除されます, ls コマンドで確認してみてください.

```
sakota@unixmachine:~$ rm -r text
sakota@unixmachine:~$ ls
sakota@unixmachine:~$
```

2.3.4 ファイルを読む

/usr/share/common-licenses/GPLにある GPLというファイルを読みます. **more /usr/share/common-licenses/GPL** と入力してください.

```
sakota@unixmachine:~$ more /usr/share/common-licenses/GPL
                    GNU GENERAL PUBLIC LICENSE
                       Version 2, June 1991

 Copyright (C) 1989, 1991 Free Software Foundation, Inc.
     59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                            Preamble

  The licenses for most software are designed to take away your
freedom to share and change it.  By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
```

```
software--to make sure the software is free for all its users.  This
General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to
using it.  (Some other Free Software Foundation software is covered by
the GNU Library General Public License instead.)  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
--more--(5%)
```

以上の様にファイルの一部ずつが表示されます．スペースや↓でその他の部分を参照できます．また，”q”を押すと中断されます．

2.3.5 ファイルのコピー

先程の GPL のファイルをコピーしたいと思います**cp /usr/share/common-licenses/GPL ./**と入力してください．GPL という名前のファイルがホームディレクトリにコピーされています．

```
sakota@unixmachine:~$ cp /usr/share/common-licenses/GPL ./
sakota@unixmachine:~$ ls
GPL
```

2.3.6 ファイルの詳細を見る

GPL のファイルの詳細をみます．**ls -al** と入力してください．現在のディレクトリにあるファイルやディレクトリの詳細を見ることができます．

```
sakota@unixmachine:~$ ls -al
drwxr-xr-x    2 sakota   usergrp      1024 Apr 18 03:58 .
drwxr-xr-x   66 root     usergrp      1024 Apr 11 16:22 ..
-rw-------    1 sakota   usergrp       958 Apr 18 03:19 .bash_history
-rw-r--r--    1 sakota   usergrp     17992 Apr 18 03:58 GPL
```

ここで GPL のパーミッションは-rw-r–r–（644）で，作成者は sakota だとわかります．

2.3.7 パーミッションの変更

chmod 000 GPL で変更します．

```
sakota@unixmachine:~$ chmod 000 GPL
sakota@unixmachine:~$ ls -al

drwxr-xr-x    2 sakota   usergrp      1024 Apr 18 03:58 .
drwxr-xr-x   66 root     usergrp      1024 Apr 11 16:22 ..
```

```
-rw-------    1 sakota   usergrp       958 Apr 18 03:19 .bash_history
----------    1 sakota   usergrp     17992 Apr 18 03:58 GPL
```

GPL のパーミッションは----------（000）へ変更されました．これで誰も読み取りや書き込みができないので**more GPL** と入力しても拒否されます．

```
sakota@unixmachine:~$ more GPL
GPL: Permission denied
```

パーミッション（000）のままではよくないので**chmod 644 GPL** と入力しパーミッションを戻しておきましょう．

2.3.8 次はファイル名の変更

mv [**ファイル名**] [**新しいファイル名**] にて変更できます．下記の例では GPL から test.txt に名前を変更しています．

```
sakota@unixmachine:~$ mv GPL test.txt
sakota@unixmachine:~$ ls -al

drwxr-xr-x    2 sakota   usergrp      1024 Apr 18 04:09 .
drwxr-xr-x   66 root     usergrp      1024 Apr 11 16:22 ..
-rw-------    1 sakota   usergrp       958 Apr 18 03:19 .bash_history
-rw-r--r--    1 sakota   usergrp     17992 Apr 18 03:58 test.txt
```

2.3.9 logout

最後に test.txt を削除し logout します．ファイル削除をおぼえてますか？

```
sakota@unixmachine:~$ rm -r GPL.txt
```

ですね？
ログアウトは**logout** と入力してください．お疲れさまでした．

3 基本的コマンド の詳細

3.1 この章の概要

この章では，ターミナル（Telnet,ssh,X-Window のコンソールなど）上で，実際に使用するコマンドを解説します．また，それに先立ちファイルを扱う上で非常に重要な概念である，ディレクトリ構造について第 3.2.1 節で詳しく説明しています．続く第 3.3 節では，ディレクトリの実際の操作方法について解説します．また，第 3.4 節でパーミッションについて，第 3.5 節でファイル操作について紹介しています．

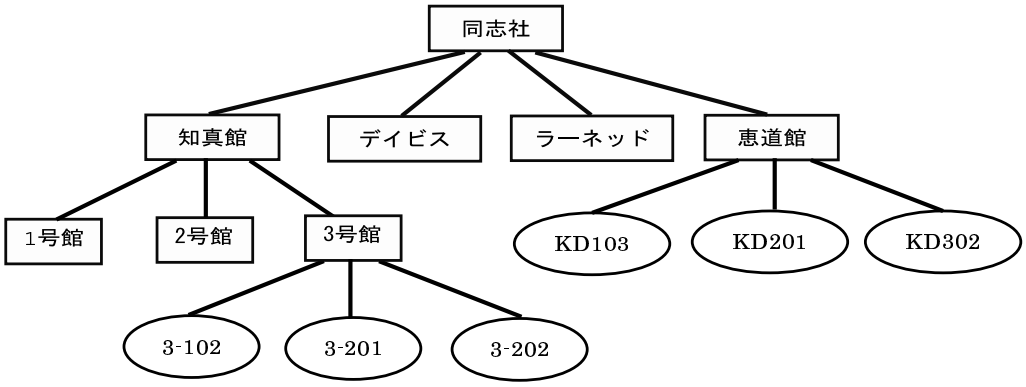

図 7: 同志社ディレクトリ

3.2 ディレクトリ構造

3.2.1 ディレクトリ構造とは

UNIX は，複数のユーザーが利用します．これらの人がごちゃごちゃにファイルを作っては，どのファイルが誰のファイルかわからなくなってしまいます．また，自分のファイルでもファイルが増えてくると，どのファイルが何のファイルなのか区別がつきにくくなってしまいます．そこで，UNIX では[19]，ディレクトリ構造という仕組みを利用して，ファイルを管理しています．ひとことで言えば，ファイルを種類ごとにまとめて，そのまとまりに名前を付けるようなものです．また，第 3.4 節で詳しくふれますが，UNIX ではファイルごとに読み込み/書き込み/実行の権限を設定できるのですが，これはディレクトリにも設定する事ができるのです．これにより，自分のディレクトリに他人が書き込めないようにしたりする事ができます．また，他人に見られたくない画像をまとめて保管することなどもできます．このように，ファイルを管理する上において，ディレクトリ構造は非常に有効なのです．ディレクトリ構造は，図 7 のように木構造になっています．

この図では各教室をファイル，建物をディレクトリと見立てて，話をしています．同志社というもっとも大きな建物の中に，知真館, デイビス，ラーネッド，恵道館という建物があります．さらに，知真館という建物の中には，１号館，２号館，３号館という建物があり，恵道館という建物の中には，KD103，KD201，KD302 という教室があります．そして，３号館という建物の中には，3-102，3-201，3-202 という教室があります．

同じ事を，ディレクトリとファイルという言葉に置き換えて考えてみます．同志社というもっとも大きなディレクトリの中に，知真館，デイビス，ラーネッド，恵道館というディレクトリがあります．さらに，知真館というディレクトリの中には，１号館，２号館，３号館というディレクトリがあり，恵道館というディレクトリの中には，KD103，KD201，KD302 というファイルがあります．

実際のコンピュータの中も，このように管理されています．また，我々は必ずどこかのディレクトリを見ている（どこかのディレクトリにいる）ことになります．ここで少し用語を解説しておきます．同志社というもっとも大きなディレクトリがありますが，このすべてを含むディレクトリを，**ルートディレクトリ**といいます．また，自分のファイル（またはディレクトリ）より１つ上のディレクトリをそのディレクトリの**親ディレクトリ**，１つ下のディレクトリをそのディレクトリの**子**

[19]先にも述べたように，MS-Windows をはじめとした，ほぼすべてのＯＳで，この仕組みは利用されています．

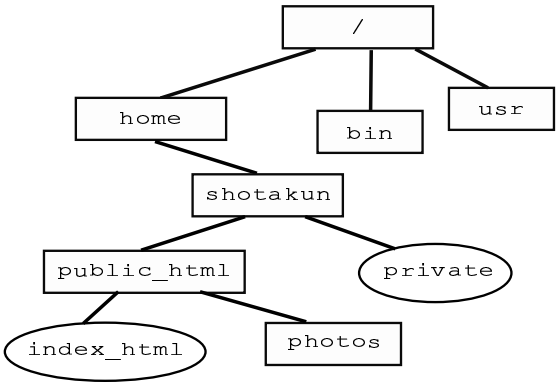

図 8: mikilab マシンのディレクトリ

ディレクトリまたは**サブディレクトリ**と言います．たとえば，知真館ディレクトリの親ディレクトリは同志社ディレクトリ（ルートディレクトリ），サブディレクトリは１号館，２号館，３号館ディレクトリになります．また，現在自分のいるディレクトリを，**カレントディレクトリ**と言います．

3.2.2 mikilab マシンでの実際

では，mikilab マシンでは，実際にはどのような構造になっているのでしょうか．それを表したのが図 8 です[20]．

図中１番上の部分にある記号”/”は，ディレクトリの区切りに利用する文字ですが，ルートディレクトリを表すときにも用います．mikilab マシンのルートディレクトリには，home，bin，usr の 3 つ[20]のディレクトリがあることがわかります．また，shotakun ディレクトリの親ディレクトリはhome，サブディレクトリには public_htmlがあり，またファイルprivate があります．ディレクトリ public_html には，ファイル index_htmlがあることがわかります．

3.2.3 絶対パスと相対パス

UNIX 上のファイルやディレクトリを指すには，絶対パスと相対パスという２つの方法があります．絶対パスは，常にルートから考えて，指定したいファイルやディレクトリがどこにあるか指定する方法です．相対パスは，カレントディレクトリから考えて，指定したいファイルやディレクトリを指定する方法です．たとえば，カレントディレクトリがshotakun だとすると，usr ディレクトリを指定するには，絶対パスでは/usr となり，相対パスでは../../usr となります．相対パス中に出てきた../は，親ディレクトリを示します．つまり，カレントディレクトリがshotakun の場合，../はhome ディレクトリを指します．public_html ディレクトリを指すには絶対パスでは，/home/shotakun/public_html となりますが，相対パスではカレントディレクトリがshotakun なら public_html となり，カレントディレクトリがusr なら../home/shotakun/public_html となります．

[20]各ディレクトリには，実際はもっと多くのファイルやディレクトリが存在しますが，説明の便宜上，大幅に省略しています

3.3 ディレクトリ操作

3.3.1 ホームディレクトリ

第 3.2.1 節でディレクトリの構造を説明しましたが，実際にディレクトリを移動・作成・削除する方法について，第 3.3.2 以降で説明します．簡単にまとめたものが表 1 です．

表 1: ディレクトリ操作コマンド

コマンド	意味	語源
pwd	カレントディレクトリを表示	print working directory
cd [ディレクトリ名]	指定したディレクトリに移動	change directory
ls [オプション] [ターゲット]	指定されたディレクトリ内のファイルと一覧を表示．-a オプションで，ファイル名の先頭にドットの付いたファイル を表示．-l オプションで詳細な一覧を表示.	list?
mkdir [ディレクトリ名]	指定したディレクトリ名でディレクトリを作成	make directory
rmdir [ディレクトリ名]	指定したディレクトリを削除	remove directory

その前に，ホームディレクトリについて説明します．ホームディレクトリとは，UNIX にログインした時最初にいる場所のことで，みなさんの場合ユーザー名と同じになっています．特殊な場合を除いて，ホームディレクトリは UNIX 上に与えられた，みなさん専用の場所です．みなさんは，ここにホームページや，メールのセーブなどをおくことになります．これは mikilab マシンでも同じで，みなさんはログインすると自動的に`home` などの下の自分のユーザー名のディレクトリに移動します．たとえば，吉田昌太の場合ホームディレクトリは，`shotakun` になります．従って，通常はホームディレクトリより上のディレクトリ構造をあまり意識する必要はありません．また，`shotakun` ディレクトリの下のファイルやディレクトリはすべて吉田昌太が作成したものです．

3.3.2 カレントディレクトリの取得と移動

現在のディレクトリを知るには，`pwd` コマンドを用います．プロンプトから

```
pwd
```

と入力します．現在，`shotakun` にいるとすると，

```
/home/shotakun
```

と結果が表示されます．

ディレクトリを移動するには，`cd` コマンドを用います．たとえば，ユーザー吉田昌太がログインし，`public_html` ディレクトリに移動するには，絶対パス指定では

```
cd /home/shotakun/public_html
```

相対パス指定では

```
cd public_html
```

と入力します.

3.3.3 ディレクトリの内容を表示

指定されたディレクトリには，どんなファイルやディレクトリがあるのかを知るには，ls コマンドを使用します．ls コマンドには，様々なオプションがありますが，ここではよく使われるls（オプション無し）とls -la について説明します．ls コマンドでは，ファイルとディレクトリの一覧が，区別無く羅列されます．たとえば，/home/shotakunディレクトリの内容を見るには，

```
ls /home/shotakun
```

と入力します．もちろん，相対パス指定も可能です．また，ディレクトリを指定しないとカレントディレクトリの内容を表示します．実行すると，

```
public_html       private
```

のように表示されます．もっと詳しい情報を知りたい場合は，ls -la を使用します．同じく，/home/shotakunディレクトリの内容を見るには，

```
ls -la /home/shotakun
```

と入力します．実行結果は，

```
drwxr-xr-x   5 shotakun  mikilab      1024 Apr 19 03:54 .
drwxr-xr-x  60 root      root         2048 Apr 13 13:24 ..
drwxr-xr-x   7 shotakun  mikilab      1024 Apr 22 01:36 public_html
-rw-r--r--   1 shotakun  mikilab     18944 Apr 27 15:03 private
```

のようになります．右端がファイル名，左端がファイル・ディレクトリの区別で，ディレクトリには”d”がつきます．d の後に続く”rwx”等の文字は，第 3.4 節で詳しく述べます.

3.3.4 ディレクトリの作成と削除

ディレクトリを新たに作成するには，mkdir コマンドを用います．カレントディレクトリがshotakunの状態で，shotakun の下にunix ディレクトリを新たに作るには，絶対パス指定では

```
mkdir /home/shotakun/unix
```

とします．すると，shotakun 以下のディレクトリ構造は図 9 のようになります.

すでにあるディレクトリを削除するには，rmdir コマンドを用います．カレントディレクトリがshotakun の状態で，shotakun の下にunix ディレクトリを削除するには，相対パス指定では，

```
rmdir unix
```

とします．すると，shotakun のディレクトリ構造は図 8 のように戻ります.

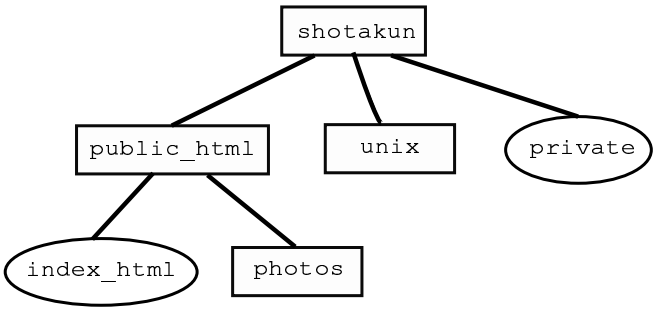

図 9: mkdir unix

3.4 パーミッション

3.4.1 パーミッションとは

UNIXでは，複数ユーザーが同時に同じマシンを使うため，各ファイルやディレクトリごとにパーミッションと呼ばれる，操作に対する制限が設定できます．ここで言う操作とは，Read/Write/eXecuteの3種類です．これらの設定を各ファイル,各ディレクトリごとに設定できます．ただし,ファイルに対しての設定とディレクトリに対する設定では意味が異なります．ディレクトリ,ファイルに対してのパーミッションの設定は，どちらも読み/書き/実行が行えます．ファイルに対しての読み/書き/実行はそのままですが,ディレクトリに対する読みは，そのディレクトリにどんなファイルがあるかを見ることで，書き込みとはそのディレクトリに新たにファイルを作ることを意味します．また，実行とはそのディレクトリをカレントディレクトリにすることを意味します．
では実際にファイル，ディレクトリがどのような設定なのかを見てみます．あるフォルダに対して，先ほど説明した`ls -l`コマンドを実行した結果を図10に示します．

```
syoshida@mikilab:~$ ls -l
total 5 ― 合計ブロック数
-rwxr-xr-x  1 syoshida mikilab     0 Apr 10 21:40 b
-rw-r--r--  1 syoshida mikilab    12 Apr  8 21:06 d
drwxr-sr-x  2 syoshida mikilab  1024 Apr 10 23:33 shota
-rw-r--r--  1 syoshida mikilab    28 Apr  8 21:17 test1
drwxr-sr-x  2 syoshida mikilab  1024 Apr  8 19:52 vazeana
drwxr-sr-x  2 syoshida mikilab  1024 Apr  8 19:52 syoshida
```

ファイルの種類 / パーミッション / ハードリンク数 / 所有者 / 所有グループ / ファイルサイズ / 更新日時 / ファイル名

図 10: ls -l の結果

図10の各項目について説明します．

- **合計ブロック数**

一覧に表示されたファイルやディレクトリが占めるディスクの容量がブロック数[21]で表示さ

[21]ブロックのサイズはシステムによって異なります．Linuxの標準的なファイルシステムでは，1024〜4096バイトが1ブロックになります．

れます.

- **ファイルの種類**

  一文字でファイルの種類を示します．ディレクトリであれば”d”，通常のファイルであれば”-”になります

- **パーミッション**

  ファイルやディレクトリに対してどの程度のアクセスが許可されているのかを示します．”r”は読み込み，”w”は書き込み，”x”は実行ができることを示します．最初の 3 文字はそのファイルの所有者，次の 3 文字はグループのメンバー，最後の 3 文字はその他のユーザーのアクセス権を示しています．たとえば図 10 に示すファイル`test1`は，所有者である`syoshida`にだけ読み込みと書き込みが許されており，グループユーザーである`mikilab`とその他のユーザーには読み込みだけが許されていることを示しています．

  なお，上記の 4 種類の文字 (d,r,w,x) の他に，図 10 には”s”という文字が用いられていますが，これについては高度な内容のため，ここでの説明は割愛します．

- **ハード リンク数**

  ファイルやディレクトリにリンクされているハード リンクの数を示します．ハード リンクとは Windows でいうショートカットみたいなものです．

- **所有者**

  ファイルやディレクトリの所有者のユーザ名を示します．通常はファイルを作成したユーザーになります．

- **グループ**

  ファイルやディレクトリを所有するグループ名．通常は，ファイルを作成したユーザーが現在所属しているグループになります．

- **ファイルサイズ**

  ファイルやディレクトリのサイズをバイト単位で示します．

- **更新日時**

  ファイルやディレクトリの更新日時を示します．

- **ファイル名**

  ファイルやディレクトリの名前を示します．

また r（読み込み），w（書き込み），x（実行）は，ファイルに適用される場合とディレクトリに適用される場合とで，多少意味が異なっています．違いを表 2 に示します．ちなみに，皆さんは，mikilab グループですので，これらのファイルにはグループユーザーとしてアクセスできます．しかし，もし次のようなリストだと，

```
-rw-r-----   1 ktanabe  student      1041 Apr  7 17:21 profile.htm
drwxr-xr--   2 ktanabe  student      1024 Dec  1 23:48 lecture
```

表 2: ファイルとディレクトリのパーミッションの違い

パーミッション	ファイル	ディレクトリ
r	内容を読み込むことができる (read)	ディレクトリの内容を`ls`コマンドのなどで表示できる
w	内容を変更したり，書き換えることができる (write)	ディレクトリ内にファイルを作成したり，削除したりできる．
x	ファイルを実行できる（実行可能ファイル）(executable)	ディレクトリにアクセスできる

みなさんは，個人でもグループユーザーでもない他人になりますので，`profile.htm`ファイルには何もできません．また，そのディレクトリに移動する（中に入る，そのディレクトリをカレントディレクトリにする）には，eXecute 権限が必要ですので，`ktanabe`本人と`student`グループユーザーしか，`lecture`ディレクトリには，入れないことになります．当然，中に何があるかも調べることができません．したがって，みなさんがもしホームページを作ったときに，`public_html`のアクセス権限が

```
drwxr-xr--   2 shota     mikilab      1041 Apr 22 01:36 public_html
```

とかになっていると自分と`mikilab`ユーザー以外は，ページを見れないと言うことになります．かならず，

```
drwxr-xr-x   2 shota     mikilab      1041 Apr 22 01:36 public_html
```

となるようにしてください．ちなみにだからといって，

```
drwxrwxrwx   2 shota     mikilab      1041 Apr 22 01:36 public_html
```

などとすると，だれでも何でも書き込みができるようになり非常に危険ですので，絶対にしないでください．

3.4.2 パーミッションの設定

では次にパーミッションを設定してみます．パーミッションを変更するのが，`chmod`[22]というコマンドです．このコマンドを利用する形式は

```
chmod [モード][ファイル名]
```

となります．ただしモードを指定する方法には，次の 2 通りがあります．

シンボリックモード　それぞれ表 3 に示す記号が割り当てられています．たとえばファイル`test1`を所有者以外読み出し禁止にするには，

```
chmod go-r test1
```

と入力します．

[22]change mode

表 3: 記号の割り当て

対象ユーザー	u（所有者），g(グループ)，o(その他)，a(すべて)
操作	+(追加)，-(削除)，=(設定)
保護モード	r(読み込み)，w(書き込み)，x(実行)

オクタルモード　オクタルモードは 8 進数によるモードの指定方法です．

```
chmod 777 filename
```

という書式を持っており，filename にはパーミッションを変更したいディレクトリやファイル名を指定します．また，777 の数字の部分は順に個人/グループ/他人のパーミッションを設定します．これは 2 進数で考えるとわかりやすいです．例えばrw-というパーミッションが設定されていれば，2 進数で110 となり，これを 10 進数に直すと 6 となります．オクタルモードにおける数字を表 4 にまとめます．

表 4: chmod の数字の意味

0	すべての権限なし	4	Read
1	eXecute	5	Read + eXecute
2	Write	6	Read + Write
3	Write + eXecute	7	Read + Write + eXecute

つまり，chmod 777 index.html とすればindex.html には，すべての人にrwx の権限が設定され，

```
chmod 755 index.html
```

とすればindex.html には，自分はrwx，自分以外はr-x の権限が設定されます．また，chmod 640 index.html とすればindex.html には，自分はrw-，グループユーザーはr--，他人は---となります．したがって，たとえばホームページの為のファイルはchmod 644，ディレクトリはchmod 755 とすれば，誰でも読めるが自分以外書き込めない設定ができます．

3.5　ファイル操作

3.5.1　ファイルについて

この節では，ファイルの操作について説明します．ファイルの操作には，内容の表示，コピー，移動，消去があります．

3.5.2　ファイルの内容の表示

ファイルの内容を表示するには，cat コマンドかmore コマンドを用います．書式は

```
cat filename
more filename
```

です．両者の違いは，cat[23]は全部一度に，more[24]は 1 画面ずつ区切って表示すると言うことです．ちなみに，両者とも表示できるのはテキストファイルのみですので，バイナリファイル[25]を表示しないように気をつけて下さい．

3.5.3 ファイルの消去

ファイルを消去するには，rm コマンドを利用します．書式は，

```
rm filename
```

です．

3.5.4 ファイルのコピー/移動

ファイルをコピーするにはcp[26]コマンドを，移動するにはmv[27]コマンドを利用します．書式は，

```
cp filename1 filename2
mv filename1 filename2
```

です．基本的にはfilename1 にコピー/移動元ファイル名を，filename2 にコピー/移動先ディレクトリ名を指定します．図 11 を例に説明します．

カレントディレクトリを shota として，ディレクトリ javajava の中の star.class を，ディレクトリ public_html にコピーするには，

```
cp javajava/star.class public_html
```

とします．コピー後は，図 12 のようになります．

また，コピー先にディレクトリ名ではなく，ファイル名を指定する事もできます．その場合，同じ内容で，ファイル名の異なるファイルができます．

```
cp cdlist cdlist2
```

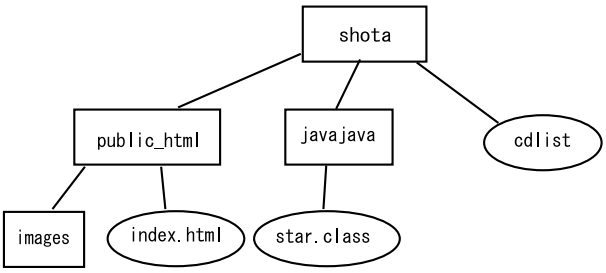

図 11: cp example

[23]catenate
[24]more?と聞いてくるから more らしい
[25]画像ファイルや実行ファイルなど，メモ帳で読めない形式すべて
[26]copy
[27]move

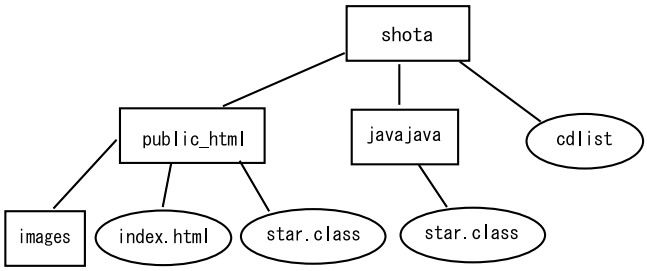

図 12: cp file dir

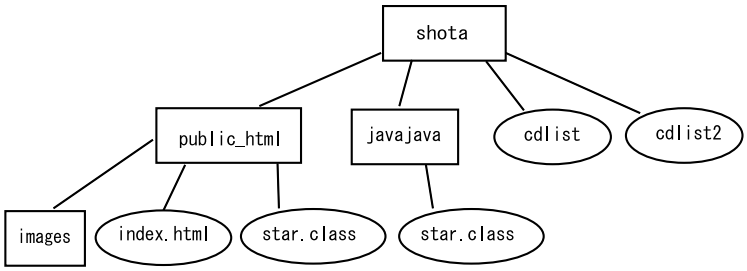

図 13: cp file file

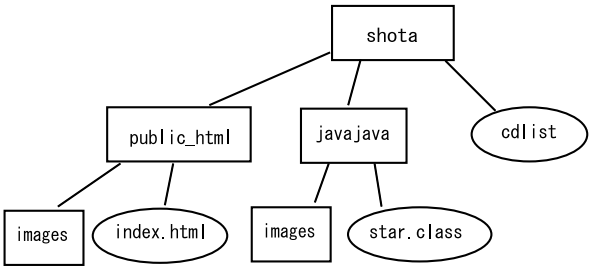

図 14: cp -r dir dir

実行後は，図 13 のようになります．
　さらに，コピー元にディレクトリ名を指定し，オプション-rを指定すると，ディレクトリ丸ごとコピーができます．

```
cp -r public_html/images javajava
```

結果は，図 14 のようになります．
　移動はコピー元が残らないだけで，あとはコピーと同じです．

3.5.5　ファイルの名前変更

　ファイル名を変更したいときにも，mvコマンドを使用します．コマンドの形式としては，次のようになります．

```
mv [変更前のファイル名] [変更後のファイル名]
```

この方法で，ディレクトリ名を変更することもできます．入力形式も同じです．