7.Xウインドウを使う

ワークステーションを利用する場合、以前のように文字端末からログインすることもあるでしょうが基本的にはXウインドウシステムと呼ばれるグラフィカルユーザインターフェース(GUI)を使うのが一般的になりました。ここではXウインドウとは一体どんなものかを説明します。ワークステーション毎にウィンドウの見栄えはずいぶん違いますし、マニュアルなど読まなくても画面を見るだけで操作方法がわかるのがGUIですから、個別のシステムの説明はしません。

7.1.Xウインドウシステムって何

Xウインドウシステムはネットワークベースのグラフィックシステムです。ユーザから見るとスクリーンとキーボード、そしてマウスがXウインドウのハードウエアです。これはX端末かも知れないし、ワークステーションかも知れないし、場合によってはパソコンかも知れません。このハードウエアを直接動かしているのはXのサーバです。Xサーバは別のところにあるクライアントプロセスから命令を受け取ってそれに従って画面に絵を書いたり押されたキーの情報を送ったり、マウスの位置を報告したりします。例えば画面に時計が表示されているとしましょう。Xサーバが自分の時計を勝手に表示しているわけではありません。何処かの計算機の`clock`というプログラムが現在の時刻の位置に針を描くようXサーバに命令を送っています。1秒に1回、以前に描いた秒針(線分)を背景色で塗りつぶして次の位置にあらたに秒針(線分)を描けという命令がネットワークを経由して送られてくるのです。このようなモデルをクライアント・サーバモデルといいます。`clock`というプログラムがクライアントです。送られる命令体形はXプロトコルといわれています。

Xウインドウシステムを使うということは使いたいXサーバを指定してXクライアントプログラムを動かすということになります。

7.2.X端末からのログイン

ではX端末から計算機にログインしてXのアプリケーションを動かしてみましょう。身近にある端末はすでに管理者の手で設定がされています。通常はログインするユーザ名とパスワードを入力すればウインドウが表示されるようになっていると思います。ロ

グインの後ウインドウを立ち上げるために必要な手続きをX端末がやってくれました。もしくは何処かの計算機が面倒を見てくれたのかも知れません。いずれにせよこの端末がXサーバになってホストと対話が出来るようになりました。ここで次をやってみましょう。(ホストはsakanayaです。)

```
% printenv DISPLAY
komeya:0.0
%
```

DISPLAYという環境変数はXクライアントプログラムが参照するXサーバがどこにあるかを指定します。komeyaというホスト名を持つXサーバが選ばれています。後に続く:0.0はその機械の上の最初の表示装置の最初のスクリーンを使う意味になりますが、普通は1つしかありませんから両方とも0です。この環境変数の設定はウインドウ立ち上げ手続きの中でなされますが、手で設定することもあります。例えばkomeyaからsakanayaにログインした状態でさらにyaoyaにtelnetでログインします。yaoyaの上で

```
% setenv DISPLAY komeya:0.0
%
```

とやるとyaoyaの上のプロセスもkomeyaのクライアントに出来ます。sakanayaとyaoyaが両方ともkomeyaをXサーバとして使えるわけです。この時yaoyaがkomeyaをアクセスできるようにsakanayaの上でxhostコマンドを実行する必要があるかも知れません。

```
% xhost yaoya
yaoya being added to access control list
%
```

ネットワーク上の誰からでもアクセスを許す場合にはホスト名の替わりに+とだけ書きます。こうしておくと別人のウインドウが突然現われたりもしますが。(DISPLAYを設定せずにクライアントプログラムを起動するとその計算機の上のXサーバがデフォルトで使われます。)

　このようにしてクライアント・サーバモデルによりネットワーク上に分散した様々なプロセスを一つのスクリーンの上にすべて引き寄せることが出来ます。このことは最近のネットワーク分散型データ収集システムでは非常に重要なことです。

7.3.ウインドウマネージャ

Xウインドウサーバはクライアントの命令にしたがって動作するだけです。ですから立ち上げたばかりのXサーバは全くそっけないものです。ただ灰色の画面の真ん中に×の形をしたカーソルが表示されるだけです。マウスを動かすとカーソルはついてきますが、ボタンを押しても何も起こりません。ボタンを押した動作を受け取るクライアントがいないからです。例えばこの状態でいきなりXのアプリケーションを起動してもウインドウの移動や寸法の変更すら出来ません。こういったことはアプリケーションに固有なことではなくてウインドウを使うクライアント全体に共通することですから誰かに面倒を見てもらいたいものです。ウインドウ全体を管理するクライアントをまず走らせて、その上で様々なアプリケーションを走らせます。そうすることにより個々のアプリケーションウインドウを共通の操作方法で制御できるようになります。ウインドウ全体の管理を行なうクライアントのことをウインドウマネージャといいます。ログインの次にやらなければならないことはウインドウマネージャの立ち上げです。多くの場合これも自動的になされます。

ウインドウマネージャには様々な種類があります。まず計算機ベンダーから提供されるウインドウマネージャにはSUNのOpenlookウインドウマネージャ(`olwm`)やDECのデックウインドウ(`dxwm`)やMotif(`mwm`)、HPのVUE(`vuewm`)などがあります。()内はウィンドウマネージャの名前(コマンド名)です。またそれ以外に`twm`など`ftp`で手に入るウインドウマネージャも色々あります。ウインドウマネージャ毎にウィンドウの形や操作方法が異なります。どれを選ぶかは好みの問題です。

それぞれのウインドウマネージャは通常ユーザが何もしなくても自動的に立ち上がるようになっている場合も多いですが、その手順を知っておくことも理解するうえで有効です。まず使うXサーバ(環境変数`DISPLAY`に設定されている)に、デフォルトで使う色や文字など(リソースと呼ばれる)を教えます。`xrdb`というコマンドが自分のホームディレクトリの`.Xdefaults`というファイルを読み込んで設定します。自分の`.Xdefaults`ファイルを読んでみてください。次にウインドウマネージャをバックグラウンドで立ち上げます。この段階で画面はそれぞれの雰囲気に変わります。続いて自動的に立ち上げるよう設定したアプリケーションが起動され、ウインドウの立ち上げ完了となります。

7.4.お好みのウインドウ

ウインドウシステムはいわば自分の作業場みたいなものですから、自分の好みに合わせて使いやすいようにあつらえたいものです。背景やウインドウの色や柄、マウスの応答、文字フォント、キーボードのクリック音、自動的に立ち上げられるアプリケーション、起動メニューに載っているアプリケーションの種類など色々なものを設定したくなります。先に挙げた.Xdefaultsを見てみましょう。

```
OpenWindows.SelectDisplaysMenu:     False
OpenWindows.WindowColor:            #e3e5b7
OpenWindows.DragRightDistance:      100
OpenWindows.WorkspaceColor:         #72bf86
OpenWindows.IconLocation:           right-bt
OpenWindows.MultiClickTimeout:      4
OpenWindows.Beep:       always
OpenWindows.SetInput:   followmouse
OpenWindows.ScrollbarPlacement:     right
KTerm*VT100*background: #f0f8ff
KTerm*VT100*kanjiMode:  euc
KTerm*VT100*saveLines:  512
KTerm*VT100*foreground: #001020
KTerm*VT100*scrollBar:  true
KTerm*pointerColor:     black
KTerm*loginShell:       true
```

これはSUNの.Xdefaultsから取ってきたものですが、Xリソースの名前が左に、その値が右に書かれています。リソース名はアプリケーションの名前と.もしくは*で区切られたリソースで表されます。この例ではOpenwindowsやKTermがアプリケーションです。値のうち#で始まるのはRGBの色の度合いを8ビットで表したものです。blackのように色の名前も使えます。色の名前は/usr/lib/X11/rgb.txtというファイルで定義されています。

7.5.X上のアプリケーション

Xウインドウ上のアプリケーションには様々なものがあります。時計やカレンダー、電卓などのほか、ファイル管理ツールや端末エミュレータ、メイル読み書きツール、印刷機能、画面ハードコピーなどいわゆるデスクトップと呼ばれる標準のアプリケーションがウインドウマネージャ毎に提供されています。ウインドウマネージャによらないXのアプリケーションもたくさんあります。例えば画面を上下に動かすスクロールバーと呼ばれるものも現在のウインドウマネージャの提供するものでないものが使われる場合

があります。これは何による違いなのでしょう。Xのアプリケーションがどのように作られているかを知っておくのはいいことです。

Xサーバとクライアントの間でやりとりされる命令をXプロトコルと呼ぶことは説明しましたが、クライアントプログラムがXプロトコルを送るために呼び出す関数を用意する必要があります。Xlibと呼ばれるライブラリーがそれですが、線を引いたり字を書いたりと言った基本的な動作を提供しているだけなので、ウインドウのメニューやボタン、スクロールバーなども沢山の単純な構成要素に分解して呼び出す必要があります。これは大変なのでメニューやボタンと言ったウインドウの部品のレベルでプログラミングが出来るライブラリーが用意されています。Xtoolkitと呼ばれます。Xtoolkit自身は特定の窓枠の形状などを知っているわけではありません。窓枠やメニュー、ボタン、スクロールバー、ポップアップ等の特定の形状や操作方法(ポリシーと呼ばれる)はウィジェットと呼ばれウインドウマネージャごとに用意されています。ですから通常のウインドウアプリケーションはあるウインドウマネージャの提供するウイジェットセットを用いてXtoolkitを呼出し、さらにXlibを呼び出します。

7.6.Xシステムコマンド

自分が今使っているXウインドウシステムの設定を確認したり変更するためのコマンドがいくつか用意されています。通常使うことはないかも知れませんがXがどのようなものかを理解するためにも少し見てみましょう。

7.6.1.Xサーバの設定

基本的なX端末(Xサーバ)の機能は`xset`コマンドで設定します。現在の設定がどうなっているか`xset -q`で問い合わせてみましょう。

```
%xset -q
Keyboard Control:
 auto repeat:  on    key click percent:  60    LED mask:  00000000
 auto repeating keys:  0000000000000000
                       0000000000000000
                       0000000000000000
                        0000000000000000
  bell percent:  60    bell pitch:  440    bell duration:  200
Pointer Control:
```

```
  acceleration:  2 = 2 / 1    threshold:  4
Screen Saver:
  prefer blanking:  yes    allow exposures:  yes
  timeout:  600    cycle:  600
Colors:
  default colormap:  0x27    BlackPixel:  1    WhitePixel:  0
Font Path:
  /usr/lib/X11/fonts/rom/,/home/cit/r5/100dpi,/home/cit/r5/75dpi,
  /home/cit/r5/misc,/home/cit/r5/xesnfOPW
```

キーボード・マウス・スクリーンセーバ・色・フォント検索パスがこのように設定されていることがわかります。各々の項目はxsetコマンドを使って設定できます。manで見てください。

X端末によっては例えばデリートキーがうまく動作しないなど、キー動作がアプリケーションに正しく伝わらないことがあります。様々な種類の計算機や端末がそれぞれ固有のキーボード配列を持っていますから、物理的にキーボードが送り出すコードと論理的にXが使うコードの間の対応を教えてやらなければならない場合があります。文字端末の設定ではsttyコマンドを使いましたが、Xの場合はxmodmapというコマンドを使います。

```
% xmodmap -e 'keycode 50 = Delete'
% xmodmap -e 'keysym BackSpace = Delete'
```

上の例ではキーコード50番のキーに論理キー(キーシムと呼ばれる) Deleteを割り当てたものです。下の例では論理キーBackSpaceを論理キーDeleteで置き換えました。-eは次の引数をコマンドとして実行しなさいという意味です。詳しくはmanで見てください。キーコードや論理キー(キーシム)が実際のキーボードに割り付けられているかを調べるにはxevコマンドを使います。

```
% xev
```

とやると、小さなウインドウが表示されます。フォーカスをそのウインドウに持っていきます。(フォーカスとはキーボードフォーカスのことです。端末には1つしかキーボードがありませんからどれか一つのアプリケーションだけがキーボードを占有します。占有することをフォーカスといいます。通常あるウインドウにマウスカーソルを持っていくかそのウインドウをクリックすることによってフォーカスは移動します。)その間にもxevを起動したウインドウには色々なメッセージが表示されます。xevはXのイベントを通知するテストツールです。小さいウインドウにフォーカスが移ったところでキー

を何か押してみましょう。

```
KeyPress event, serial 17, synthetic NO, window 0xc00001,
    root 0x8006c, subw 0x0, time 4169328, (104,97), root:(199,214),
    state 0x0, keycode 50 (keysym 0xff08, BackSpace), same_screen YES,
    XLookupString gives 1 characters:  "

KeyRelease event, serial 17, synthetic NO, window 0xc00001,
    root 0x8006c, subw 0x0, time 4169448, (104,97), root:(199,214),
    state 0x0, keycode 50 (keysym 0xff08, BackSpace), same_screen YES,
    XLookupString gives 1 characters:  "
```

バックスペースキーを押してみました。押したイベントと離したイベントが通知されました。キーコードが50でキーシムがBackSpaceであることがわかります。

Xウインドウは通常灰色の愛想のない背景を表示します。何もウインドウの表示されていない地面のスクリーンのことをルートウインドウと呼びます。様々なアプリケーションがルートウインドウの上に色々なウインドウを表示していくわけです。ルートウインドウの属性変更をするコマンドがxsetrootです。ルートウインドウの色や柄を変えたりマウスの形状を変えたりできます。PCのウインドウズの壁紙の様にカラーイメージをルートウインドウに表示させて気分をなごませたいこともあるでしょう。xloadimageコマンドがそれをやってくれます。

7.6.2.フォント関係

X端末では非常に多くの種類のフォントが使われます。アプリケーション毎に使われるフォントも違っています。X端末ではフォントサーバに設定されたホストの上のフォントパスに設定されたディレクトリから必要なフォントを探してきます。フォントパスの追加や削除はxsetコマンドで行なえます。フォントパスに指定されたディレクトリには.pcfや.snfといった拡張子をもったフォントデータファイルとfonts.dir、fonts.aliasという名前のファイルがあります。fonts.dirはリソースとしてのフォントの名前とフォントデータファイルの対応を記述します。fonts.aliasは該当する名前のフォントがないとき代用するフォントの名前を記述します。

xlsfontsというコマンドはそのXサーバで使えるフォントの名前の一覧を表示します。Xのアプリケーションが"フォントが見つからない"と言ってくる場合があります。この時はxlsfontsの中にあるフォントで代用するようにします。例えばxtermなどで自分の好みのフォントを使って起動することが出来ます。

```
% xterm -fn helvb12 &
```

こうやると例えばヘルベチカボールド12ポイントのフォントでxtermが起動されます。xlsfontsで表示されるフォントが実際どんなものなのかを表示させるにはxfdというコマンドを使います。

```
% xfd rk24
```

などとフォント名を後に付けて呼び出します。
　Xウインドウ関係の解説書はプログラマー向けのものが多いのですが、ユーザ向けのものとして「入門XWindow」松田晃一・暦本純一著、アスキー出版局はお勧めです。