■6-2-21 QuickTimeの圧縮設定ダイアログ

さらに、ムービーの画像を変形しながら再生することも、コンピューターに高い処理能力を要求する。一定の変形だけでよければ、あらかじめ変形した映像をムービー・ファイルとして保存しておき、それを単純に再生することが考えられる。

この方法ではリアルタイムの変形ができないが、実現したい処理内容に応じた工夫ができるかもしれない。例えば、10段階にムービーを拡大縮小したい場合は、拡大縮小した10種類のムービー・ファイルを作っておく。そして、状況に応じて10種類のムービーを切り替えて再生すればよいだろう。

6-3 ビットマップの描画

参照、P.119

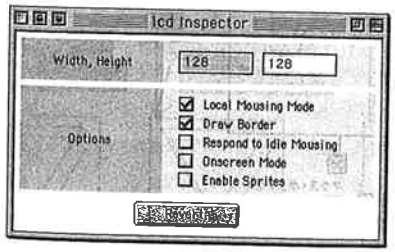
すでに『Chapter 3 プログラミングの手法』の題材として説明したlcd《LCD》オブジェクトだが、ここではあらためてその主な機能をまとめておこう。lcdオブジェクトはビットマップ(またはピクセルマップ)によって画像を表現する。ビットマップは小さな点の集合として画像を構成する。Photoshopなどのペイント系のグラフィック・ソフトで作成する画像や、デジタル・カメラで撮った画像などがビットマップである。

lcdオブジェクトの設定とマウス描画

lcdオブジェクトはオブジェクト・パレットの29番目にあり、鉛筆のアイコンで示される。lcdオブジェクトをパッチ・ウィンドウに作成すると、矩形の描画領域が作られる。インスペクターを開けば、lcdオブジェクトに関して次のような設定を行うことができる。

まず、Width, Height欄では、描画領域の大きさをピクセル単位で設定する。Local Mousing Modeはlcdオブジェクト上でのマウス・ドラッグによる描画を行うか否かを指定する。Draw Borderをチェックすれば、lcdオブジェクトの描画領域を枠線で描く。Response to Idle Mousingでは、マウス・ボタンを押していないアイドル時にマウス・ポインターの位置を出力することを指定する。Onscreen Modeを有効にすると、表示画像をメモリーに蓄えるオフ・スクリーンを使わずに描画を行う。そのため、メモリーを節約できるが、描き直すことができない。Enable Spritesは、後述するスプライトを用いる場合にチェックする。

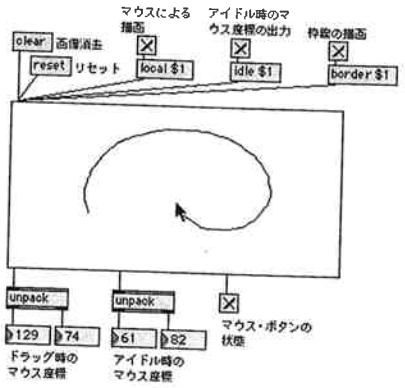
■6-3-1 lcdオブジェクトのインスペクター



lcdオブジェクトの上でマウスをドラッグすれば、線画を描くことができる。ドラッグ時のマウス・ポインターの位置はlcdオブジェクトの第1アウトレットからリストとして出力される。これは、lcdオブジェクトの左上隅を原点とするx座標とy座標から成るリストだ。また、第3アウトレットからはマウス・ボタンを押したときに1が、マウス・ボタンを離したときに0が出力される。マウス・ドラッグによる描画はlocalメッセージで有効か無効かを設定できる。localのアーギュメントが0なら、マウス・ドラッグしても何も描かれない。アーギュメントを0以外の整数にすれば、マウス・ドラッグで描画できるようになる。localメッセージはインスペクターのLocal Mousing Modeの設定に相当する。マウス・ボタンを押していない状態でも、マウス・ポインターの位置を知りたい場合は、0以外の整数をアーギュメントとするidleメッセージを送ればよい。これでlcdの第2アウトレットから、マウス・ポインターの位置がリストとして出力される。アーギュメントが0のidleメッセージを送れば、lcdオブジェクトの第2アウトレットからの出力が行われなくなる。idleメッセージによる指定は、インスペクターでのResponse to Idle Mousingの設定と同じだ。

lcdオブジェクトに描画した画像はclearメッセージで消去することができる。resetメッセージでも画像を消去することができるが、この場合はのちに説明する描画モードやペンの状態も初期化することになる。また、borderメッセージで、lcdオブジェクトの描画領域の枠線を描くかどうかを指定できる。これはインスペクターのDraw Borderに相当する。

■6-3-2 lcdオブジェクトでのマウスによる描画



ペンの移動と設定

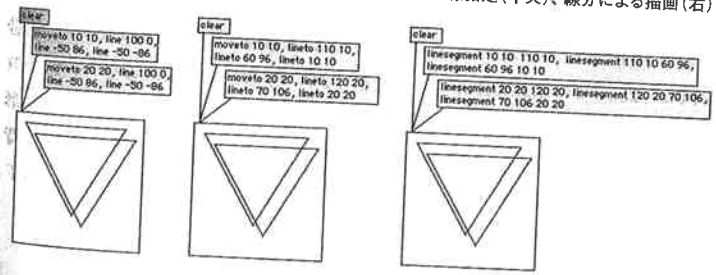
lcdオブジェクト上を実際にマウスでドラッグすることで線画を描画できるが、同じようにlcdオブジェクトは目に見えないペンも持っている。この仮想のペンをメッセージによって動かすことで、描画を行うことができる。ペンを使って線を描くにはlineメッセージやlinetoメッセージを用いる。どちらもアーギュメントとして2つの整数を与え、それぞれペンを移動する先のx座標とy座標をピクセル単位で指定する。ただし、lineメッセージの場合は現在のペンの位置からの相対座標で表し、linetoメッセージではlcdオブジェクトの左上隅を原点とする絶対座標で表す。

lineとlinetoメッセージは、紙(lcdオブジェクト)にペンを押しつけた状態で動かすので線が描かれる。これに対してペンを紙から離した状態で動かすには、moveメッセージとmovetoメッセージを用いる。この場合は、ペンは移動するが線は描かれない。moveメッセージはペンを移動する先を相対座標で指定し、movetoメッセージは絶対座標で指定する。

6-3-3のパッチはlineとlinetoメッセージを使って三角形を描いている。左側のパッチは、movetoメッセージで始点まで移動したあと、lineメッセージによって相対的に移動しながら描画しているのに対して、右側ではlinetoメッセージによって絶対座標を指定して描画している。同じ形の図形を描くときは、相対的な描画は始点を変えるだけで済むので便利だろう。

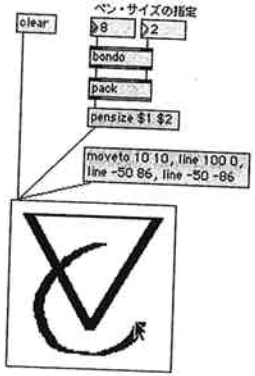
また線分を描くためにはlinesegmentメッセージが利用できる。linesegmentメッセージは4つの整数から成るアーギュメントを持ち、始点のx座標、y座標、終点のx座標、y座標を指定する。いずれもlcdオブジェクトの左上隅を原点とする絶対座標で表す。

■6-3-3 ペン移動による描画の相対座標指定(左)と絶対座標指定(中央)、線分による描画(右)



ペンの大きさはpensizeメッセージで設定することができる。pensizeメッセージは2つの整数のアーギュメントを持ち、ペンの横幅と縦幅をピクセル数で指定する。ペンの幅を大きくすれば、太い線が描かれることになる。ただし、ペンの形状は円形ではなく、矩形になっている。デフォルトのペンの大きさは縦横ともに1ピクセルだ。

■6-3-4　ペン・サイズの指定



ここで用いているbondoオブジェクトは、いずれのインレットにメッセージを受け取った場合でも、内部に蓄えている値を用いて、すべてのアウトレットからメッセージを出力するオブジェクトだ。このパッチでは、第2インレットへのナンバー・ボックスを操作した場合でも、pensizeメッセージが送られることになる。

ペンの色はcolorメッセージで指定できる。アーギュメントには0から255までの整数を与え、Maxが持つカラー・パレットのインデックス番号で指定する。

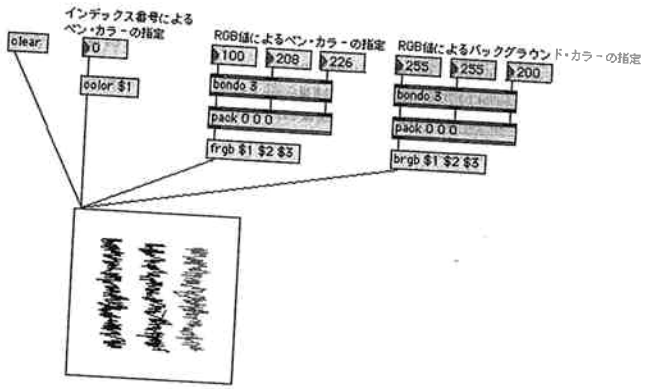
また、RGB値でペンの色を指定したい場合は、frgbメッセージを用いる。この場合は、0から255までの整数でRGBの各値を表す3つの整数から成るリストをアーギュメントに指定する。背景色はbrgbメッセージで指定し、アーギュメントとしてfrgbと同じ形式でRGB値を指定する。brgbメッセージを送ったあと、clearメッセージを送ると、背景色でlcdオブジェクトの描画領域が塗りつぶされる。

04/12/9

画像解析

アナログ

■6-3-5　ペンの色と背景色の指定



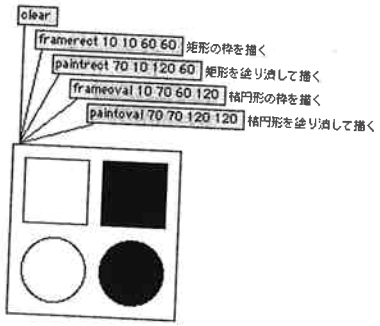
● 幾何学図形の描画

lcdpオブジェクトには矩形や楕円形など幾何学図形を描くメッセージが用意されている。同じ図形でも枠線だけを描くメッセージと内部を塗りつぶすメッセージとがある。以下、図形描画メッセージを見ていこう。

framerect	矩形の枠を描く
paintrect	矩形を塗りつぶして描く
frameoval	楕円形の枠を描く
paintoval	楕円形を塗りつぶして描く

いずれもアーギュメントに4つの整数を与え、図形の左端座標、上端座標、右端座標、下端座標を指定する。

■6-3-6 矩形と楕円形の描画

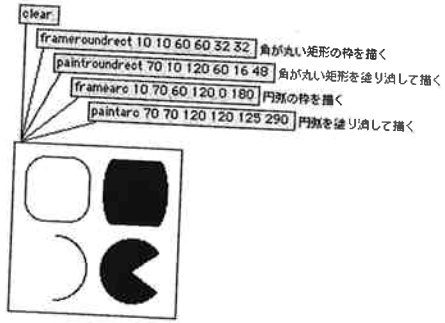


frameroundrect　角が丸い矩形の枠を描く
paintroundrect　角が丸い矩形を塗りつぶして描く

アーギュメントには図形の左端座標、上端座標、右端座標、下端座標に加えて、角を丸める横幅と縦幅を指定する。

framearc　円弧の枠を描く
paintarc　円弧を塗りつぶして描く

■6-3-7 角が丸い矩形と円弧の描画

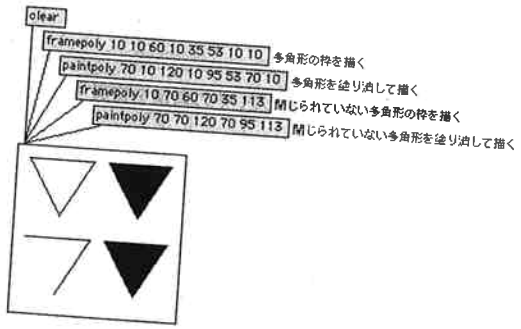


アーギュメントには図形の左端座標、上端座標、右端座標、下端座標、そして円弧の開始角度と終了角度を度数で指定する。

framepoly　多角形の枠を描く
paintpoly　多角形を塗りつぶして描く

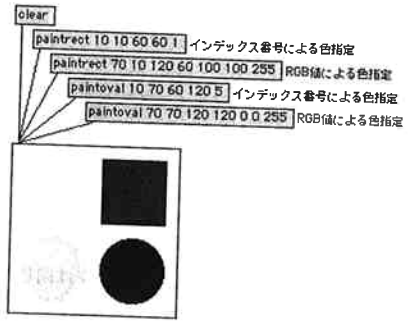
アーギュメントとして、多角形の頂点のx座標とy座標を一組として、頂点の数だけ繰り返して指定する。したがって、アーギュメントの数は三角形なら8個、五角形なら12個になる。最大254個のアーギュメントを指定できる。なお、framepolyメッセージは閉じていない多角形(線分)を描くことができるが、paintpolyメッセージは閉じていない部分を補って塗りつぶすことになる。

■6-3-8 多角形の描画



なお、いずれの描画メッセージも、同時に描画色を指定することができる。これは座標指定のアーギュメントに続けて、1つの整数を与えるとカラー・パレットのインデックス番号で色を指定したことになる。また3つの整数を続けた場合は、RGB値での色指定になる。このようにして描画色を指定すると、それ以降指定し直さない限り、その色によって描画が行われる。

■6-3-9 アーギュメントで描画色を指定した描画



画像と文字の描画

lcdオブジェクトには画像ファイルを読み込んで、その画像を描画することができる。画像ファイルのフォーマットはPICTが基本だが、QuickTimeがインストールされているなら、JPEGやGIFなどのQuickTimeが扱える画像フォーマットも利用できる。いずれも、Maxのサーチ・パス内に画像ファイルが存在している必要がある。

lcdオブジェクトではreadpictメッセージによって画像ファイルを読み込む。このとき、最初のアーギュメントで画像の名前を付け、2番目のアーギュメントで画像ファイル名を指定する。これ以降、画像を第1アーギュメントで指定した画像名で参照する。

画像を描画するにはdrawpictメッセージを用いる。このアーギュメントは次のようになる。

drawpict <画像名> <描画の左端座標> <描画の上端座標> <描画の横幅> <描画の縦幅> <元画像のx座標> <元画像のy座標> <元画像の横幅> <元画像の縦幅>

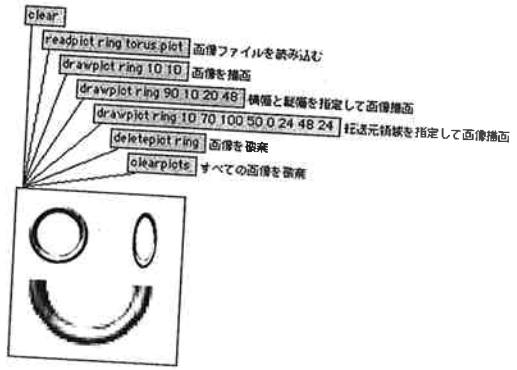
これらのアーギュメントは省略可能なので、必要とするアーギュメントだけを与えて、それ以降を省略しても構わない。例えば、描画の上端座標までのアーギュメントを与

えれば、指定した位置に原寸大で描画される。描画の縦幅までを指定すると、拡大または縮小して描画することになる。そして、すべてのアーギュメントを用いれば、元画像の一部を拡大または縮小して描画することができる。

なおマスクを指定して描画することはできないが後述する描画モードをtransparentにすれば、背景を抜いて描画することができる。

読み込んだ画像が不要になった際には、画像名をアーギュメントとしてdeletepictメッセージを送れば、その画像のために使用していたメモリーが解放される。すべての画像を破棄するには、clearpictsメッセージを用いる。

■6-3-10 画像の描画

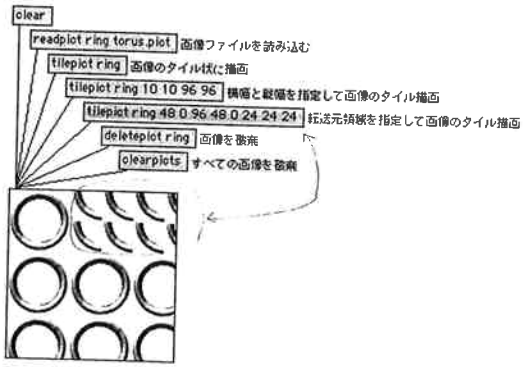


画像を1つだけ描くのではなく、タイルのように画像を敷き詰めて描くためにtilepictメッセージが用意されている。tilepictのアーギュメントはdrawpictと同じだ。

tilepict <画像名> <描画の左端座標> <描画の上端座標> <描画の横幅> <描画の縦幅> <元画像のx座標> <元画像のy座標> <元画像の横幅> <元画像の縦幅>

ただし、tilepictメッセージはアーギュメントで指定した領域内に画像をタイル状に描くので、drawpictメッセージのような画像の拡大や縮小は行わない。

■6-3-11 画像をタイル状に描画



lcdオブジェクトに文字を描画するにはasciiメッセージまたはwriteメッセージを用いる。asciiメッセージはアーギュメントの整数をASCIIコードと見なして文字を描画する。文字をASCIIコードに変換するにはspellオブジェクトが利用でき、writeメッセージはアーギュメントをそのまま文字列として描画する。いずれもアーギュメントを複数持たせることができる。文字はペンの位置に描画されるので、必要に応じてmoveやmovetoメッセージでペンを移動してから文字を描画するとよい。

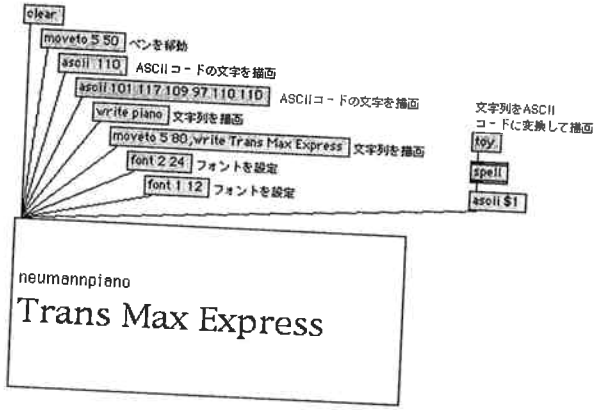
描画する文字のフォントを設定するには、fontメッセージを用いる。アーギュメントにはフォントのID番号とフォント・サイズを指定する。

ただし、システムへのインストール状態によって利用できるフォントが異なるので注意が必要だ。代表的なフォントとそのID番号を以下に示す。システム・フォントとアプリケーション・フォントは日本語システムの場合はOsakaだが、"アピアランス"コントロールパネルの設定によって異なる。

システム・フォント	0
アプリケーション・フォント	1
New York	2

Geneva	3
Monaco	4
Times	20
Helvetica	21
Courier	22
Symbol	24

■6-3-12 文字の描画



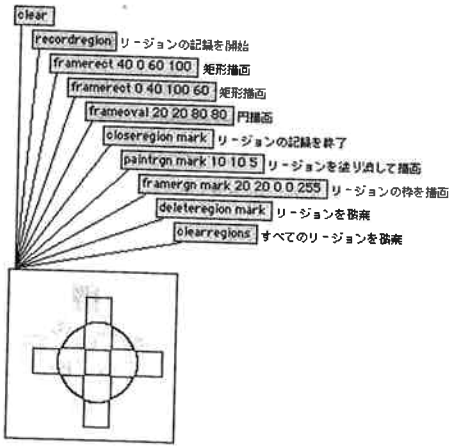
● リージョンとクリッピング

lcdオブジェクトではリージョンというテクニックを用いて、複雑な図形を描画することができる。リージョンは図形の記憶装置と考えればよいだろう。リージョンを利用するためには、まず、recordregionメッセージをlcdオブジェクトに送ってリージョンの記録を開始する。その後に、何らかの図形描画を行うと、それらはすべてリージョンに記憶される。ただし、この間の描画は実際には表示されない。closeregionメッセージを送ると記憶が終了する。closeregionメッセージはアーギュメントにシンボルを与えて、その

シンボルが記憶したリージョンの名前となる。記憶したリージョンは、paintrgnメッセージでリージョンを塗りつぶして描画することができる。framergnメッセージならリージョンの枠線を描く。いずれも、アーギュメントとしてリージョンの名前と描画位置を指定する。カラー・パレットのインデックス番号またはRGB値での描画色を指定することもできる。リージョンが不要になれば、リージョンの名前をアーギュメントとしてdeleteregionメッセージをlcdオブジェクトに送れば、そのリージョンは破棄され、メモリー領域が解放される。複数のリージョンを記憶しているときは、clearreginメッセージですべてのリージョンを破棄することができる。

6-3-13のパッチでは、上から順にメッセージ・ボックス(message《Message Box》)をクリックしていけば、リージョンの記録からリージョンによる描画、そしてリージョンの破棄までができるようになっている。リージョンとして記憶した図形は、2つの長方形と1つの円形で、いずれも枠線として描いている。塗りつぶした描画はリージョンとしては扱えず、線画としての描画が有効になる。つまり、framerectメッセージなどは使えるが、paintrectメッセージなどは使えない。

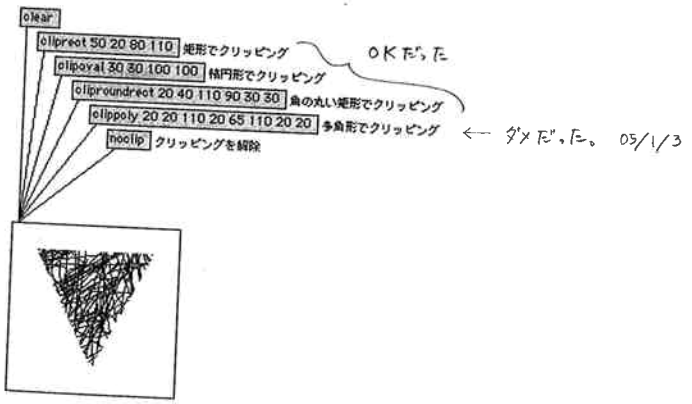
■6-3-13 リージョンによる描画



ちなみに、リージョンを記録中に、マウスをドラッグして線画を描いてもリージョンとして記憶される。lineやlinetoなどの線画を描くメッセージも有効だ。そして、リージョンを塗りつぶした描画から分かるように、図形が重なった部分はリージョンが反転するようになっている。

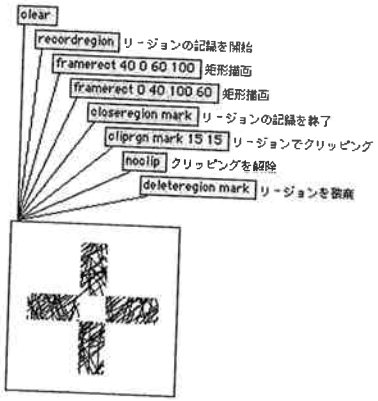
さらに、lcdではオブジェクトの領域全体に描画ができるが、特定の部分だけに描画するように設定することもできる。これをクリッピングと呼ぶ。クリッピングを行うには、cliprect、clipoval、cliproundrect、clippolyといったメッセージを用いる。それぞれ、矩形、楕円形、角の丸い矩形、多角形の形状でクリッピングし、framerectメッセージなどと同じようにアーギュメントで図形の座標を指定する。クリッピングを解除するにはnoclipメッセージをlcdオブジェクトに送ればよい。次のパッチでcliprectなどのメッセージを送った上で、lcdオブジェクト上でマウスをドラッグして描画してみよう。クリッピングした領域内では描画ができるが、それ以外では何も描かれないことが分かるだろう。

■6-3-14 クリッピングの設定



複雑な形状でクリッピングしたいときは、リージョンを用いてクリッピング領域を指定すればよい。このためには、先に説明したようにリージョンを記録し、cliprgnメッセージでクリッピングを指定することになる。

■6-3-15 リージョンによるクリッピングの設定



スプライトの描画

これまでに説明したlcdオブジェクトの描画とはやや異なる便利な機能として、スプライトがある。スプライトは、透明シートの上に描画した画像を、lcdオブジェクトの描画領域に重ねたものと考えるとよいだろう。したがって、通常のlcdオブジェクトの画像に影響を与えずにスプライトを移動することができる。これはPhotoshopでのレイヤー上の描画や、Directorのスプライト・アニメーションに似ている。

スプライトを利用するには、まずenablespritesメッセージをlcdオブジェクトに送って、lcdオブジェクトのスプライト機能を有効にする必要がある。enablespritesメッセージのアーギュメントが0以外の整数であれば、スプライトが利用できる。これはインスペクターのEnable Spritesをチェックすることと同じだ。

次にrecordspriteメッセージによってスプライトの記録を開始する。その後、lcdオブジェクトへの描画メッセージを使って、スプライトとして用いる画像を作成する。lcdオブジェクト上でマウス・ドラッグして線画を描いてもよい。ただしスプライト記録中の描画は表示されない。必要な描画が終わったらclosespriteメッセージを送ってスプライトの

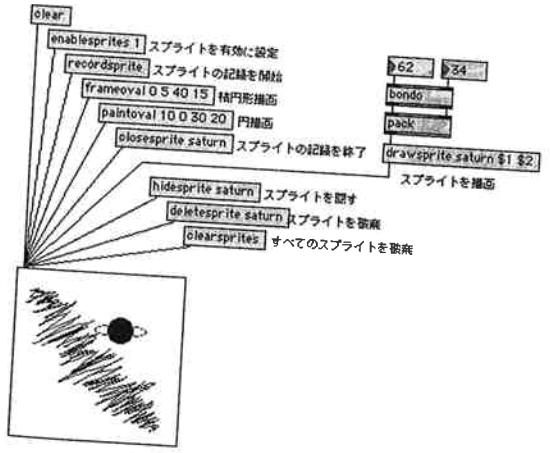
記録を終了する。closespriteメッセージはアーギュメントにシンボルを与え、そのシンボルでスプライトの名前を指定する。このスプライト名を使って描画を行うことになる。

スプライトを描画するにはdrawspriteメッセージを利用する。drawspriteメッセージは、アーギュメントにスプライトの名前と描画する位置を指定するものだ。スプライトは通常のlcdオブジェクトの画像に重ね合わせて表示される。スプライトを描画する位置を変えれば、スプライトが移動するように見えるだろう。

スプライトが必要でなければ、そのスプライトの名前をアーギュメントとしてdeletespriteメッセージをlcdオブジェクトに送ろう。これで、そのスプライトは破棄され、スプライトのために使われていたメモリーが解放される。またclearspritesメッセージを用いれば、すべてのスプライトが破棄される。

では具体的に、次のパッチでスプライトの機能を確認しよう。enablesprites 1からclosesprite saturnまでのメッセージ・ボックスを順にクリックすれば、スプライトが作成される。次いで、ナンバー・ボックスを操作すれば、スプライトを上下左右に移動することができる。lcdをマウス・ドラッグして画像を描けば、その画像に影響を与えずに、スプライトが移動することが分かるだろう。

■6-3-16 スプライトの描画

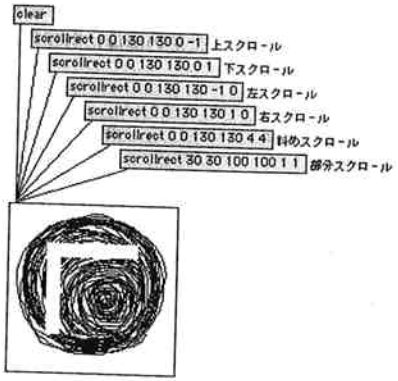


05/1/3

その他の描画処理

lcdオブジェクトの画像をスクロールすることによって、任意の方向に画像を移動することができる。このためには、scrollrectメッセージを用いる。scrollrectメッセージは6つの整数をアーギュメントとして持ち、順にスクロールする範囲の左端座標、上端座標、右端座標、下端座標、そして横方向のスクロール量と縦方向のスクロール量を指定する。いずれもピクセル単位だ。スクロール量が正の値であれば、右または下へスクロールし、負の値であれば、左または上へスクロールする。スクロールした結果空白が生じる部分は、背景色で塗りつぶされる。なお、画像のスクロールはスプライトに影響を与えず、スプライトはスクロールされない。

■6-3-17　画像のスクロール



05/1/4

lcdオブジェクトに描画を行うときに、penmodeメッセージによって描画モードを設定することができる。penmodeメッセージはアーギュメントに以下のような整数で描画モードを指定する。

0	Copy
1	Or

画像解析

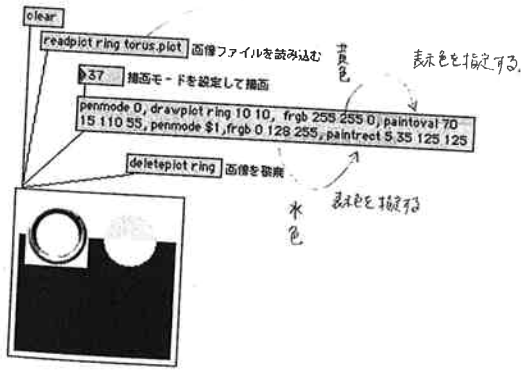
アナログ

2	Xor
3	Bic
4	NotCopy
5	NotOr
6	NotXor
7	NotBic
32	blend
33	addPin
34	addOver
35	subPin
36	transparent
37	adMax
38	subOver
39	adMin

描画モードの違いを言葉で説明するのは難しいが、6-3-18のパッチで描画モードによる描画の違いが分かるだろう。このパッチでは画像と黄色い円を描き、ナンバー・ボックスで指定した描画モードを用いて水色の矩形を半ば重なるように描いている。

■6-3-18　描画モードを指定した描画

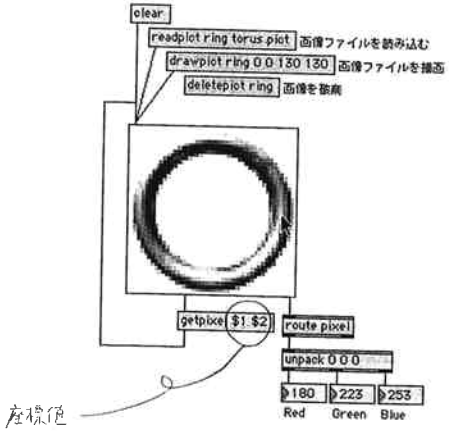
./patches/picts/wfknob.pct



6-3-18のパッチではlcdオブジェクトに描かれた画像のピクセルのRGB値を知るためにgetpixelメッセージが用意されている。getpixelメッセージはアーギュメントとして調べるピクセルの座標を指定する。このメッセージが送られると、lcdオブジェクトは第4アウトレットから、pixelに続けて3つの整数から成るメッセージを出力する。これらの整数はピクセルのRGB値を0から255までの範囲で表している。

また6-3-19のパッチでは、lcdオブジェクトのResponse to Idle Mousingを有効にしているので、マウス・ポインターをlcdオブジェクト上で動かすと、その座標値が第2アウトレットから出力される。この座標値を使ってgetpixelメッセージをlcdオブジェクトに送り、マウス・ポインターが示すピクセルのRGB値を得ている。第4アウトレットから出力されたpixelメッセージは、routeオブジェクトとunpackオブジェクトを使って、RGBのそれぞれの値をナンバー・ボックスに表示している。

■6-3-19　画像のピクセルのRGB値を取得



なおlcdオブジェクト上に作成した画像はwritepictメッセージを使って、ファイルに保存することができる。lcdオブジェクトにwritepictメッセージを送ると、ファイル保存ダイアログが開くので、保存する場所とファイル名を設定して保存を行えばよい。

画像はPICTフォーマットとしてファイルに保存される。writepictメッセージのアーギュメントにシンボルを与えると、そのシンボルをファイル名として保存が行われる。この場合、ファイル保存ダイアログは開かない。なお、スプライトを用いている場合は、スプライトが重ね合わさった画像としてファイルに保存される。

■6-3-20　画像のファイル保存