6-5 ユーザー・インターフェースの作成

最近のアプリケーションには現実のボタンやスライダーなどと見間違えるような写実的なユーザー・インターフェースが使用されていることがある。Maxも画像さえ用意すれば同じようなユーザー・インターフェースを作ることができる。もちろん写実的な表現だけに限らないが、目的に応じた独自のユーザー・インターフェースを自作できることに価値がある。ここではそのようなユーザー・インターフェースの作成方法を見ていこう。

ubuttonオブジェクトによる透明ボタン

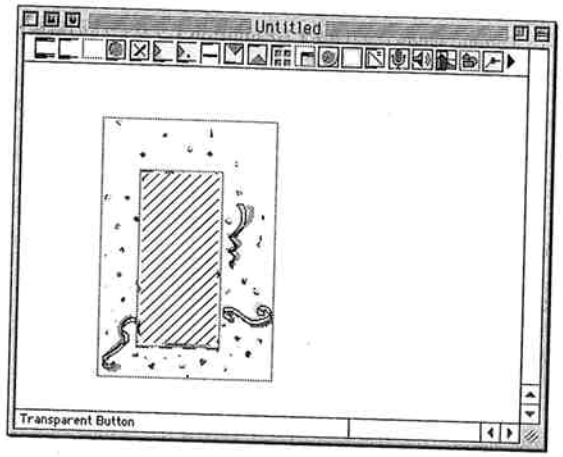
vpictureやfpicオブジェクトを用いて、パッチ・ウィンドウに画像を表示することができることはすでに述べたが、この画像は単に表示されているだけで、ユーザーがクリックしても何らかの動作をするわけではない。しかし、ubutton《Transparent Button》オブジェクトを併用することで、パッチ・ウィンドウの画像をボタンとして用いることができる。ここでは、クリップボード経由で配置した画像をボタンとして用いる方法を見ていこう。

ubuttonはオブジェクト・パレットの後ろから4番目にある斜線の入ったアイコンだ。アシスタンス・エリアにはTransparent Buttonと表示される。このubuttonオブジェクトをパッチ・ウィンドウに作成し、画像全体または適当な部分を覆うように位置と大きさを調整しよう(6-5-1)。

ubuttonオブジェクトをクリックすると、その領域の中で白い部分が強調色でハイライト表示される。これは"アピアランス"コントロールパネルで設定した強調表示色であり、画像に白い部分が少なければ目立たない。そのため、使用する画像に白い部分を多く持たせるか、画像よりも一回り大きくubuttonで覆うといった工夫をした方がよい。

マウス・ボタンを押したときにubuttonオブジェクトの第2アウトレットからbangメッセージが出力され、マウス・ボタンを離すと第1アウトレットからbangメッセージが出力される。第3アウトレットからは、クリックした位置がリストとして出力される。これはubuttonオブジェクトの左上隅を原点とする座標として表される。また、マウス・ボタンを押したときと離したときに、マウス・ポインターがubuttonオブジェクトの領域内にあれば、第4アウトレットから1が、そうでなければ0が出力される(6-5-2)。

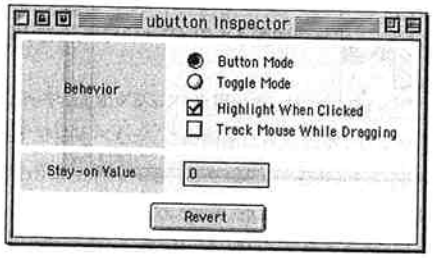
■6-5-1 画像の上にubuttonオブジェクトを配置



■6-5-2 ubuttonオブジェクトの動作

ubuttonオブジェクトはインスペクターによって設定を変更することができる。Button Modeでは先に説明した動作を行うが、Toggle Modeを選べばtoggleオブジェクトに似た動作になる。Toggle Modeでクリックするとubuttonオブジェクトの領域がハイライト表示され、第2アウトレットからbangメッセージが出力される。次にクリックしたときにはハイライト表示が解除され、第1アウトレットからbangメッセージが出力されるようになる。いずれのモードでもHighlight When Clickedのチェックを外すと、ハイライト表示が行われない。

■6-5-3 ubuttonオブジェクトのインスペクター

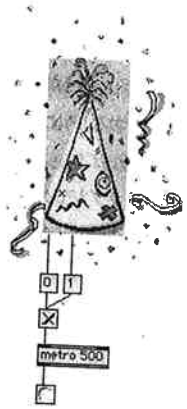


ubuttonの機能を使えば、いろいろと応用が考えられるが、単純には第1アウトレットまたは第2アウトレットから出力されるbangメッセージを使って、ubuttonオブジェクトがクリックされたときの処理を行えばよいだろう。

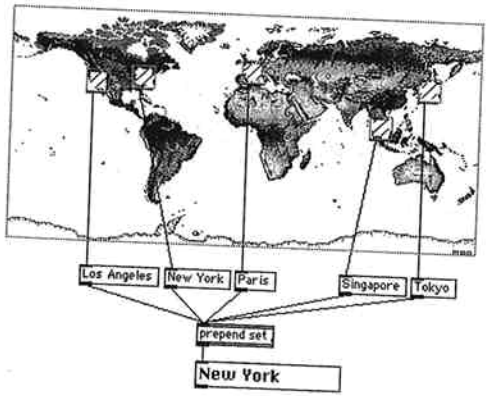
6-5-4の例では、ubuttonオブジェクトをToggle Modeで用い、第2アウトレットと第1アウトレットからの出力によってmetroオブジェクトの動作を切り替えている。ubuttonオブジェクトは画像の三角帽子部分に重ねているので、ユーザーから見れば三角帽子をクリックする度に、ハイライト表示が切り替わり、ボタンの点滅が始まったり、停止したりすることになるわけだ。

また6-5-5の例では、世界地図の画像の上に数個のubuttonオブジェクトを配置している。これらのubuttonオブジェクトをクリックすれば、その位置にある都市の名前を一番下のメッセージ・ボックスに表示するようになっている。

■6-5-4 ubuttonオブジェクトのクリックに応じた処理



■6-5-5 画像に複数のubuttonオブジェクトを配置した例



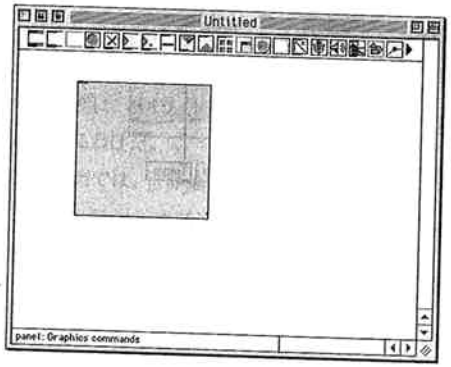
複数のオブジェクトを重ねて配置する場合、その重なり具合に気を付ける必要がある。画像の背後にubuttonオブジェクトがある場合は、編集中に見えなくなってしまう。ubuttonオブジェクトは透明ボタンであるので、実用上問題はないが、他のオブジェクトでは表示や動作に支障が出るかもしれない。そのような場合には、オブジェクトを選択し、ObjectメニューのBring to Frontを選ぶと、そのオブジェクトが手前に引き出される。Send to Backを選べば、そのオブジェクトは奥に位置するようになる。これらのメニュー・コマンドを使って、オブジェクトの前後関係を設定すればよい。

このように、パッチ・ウィンドウに張り付けた画像とubuttonオブジェクトを使って、簡単なユーザー・インターフェースを作ることができる。しかし、ubuttonオブジェクトでの表示方法は限られるので、画像が変化するようなユーザー・インターフェースを作成するには、後述するpictctrlなどのオブジェクトを利用するのがよいだろう。

panelオブジェクトによるパネル表示

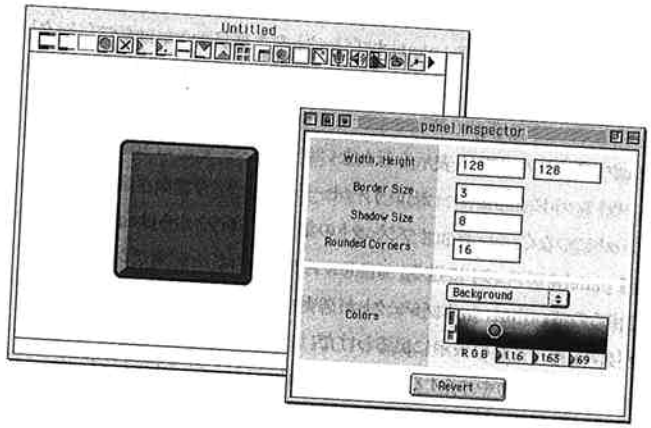
panelオブジェクトは、単純に矩形を表示するオブジェクトだ。しかし、任意の色を表示する以外にも、境界線の色や太さ、陰影の幅、角の丸みの設定が可能である。panelはオブジェクト・パレットの後半にあり、オレンジ色と水色の正方形が重なったアイコンで示される。パッチ・ウィンドウに作成したpanelオブジェクトは、灰色の平坦な矩形として表示される。

■6-5-6 パッチ・ウィンドウに作成したpanelオブジェクト



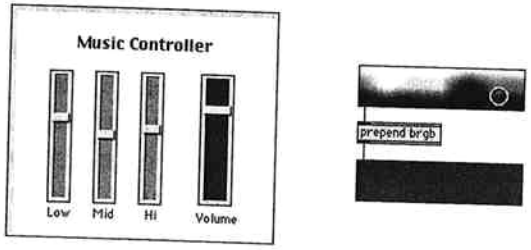
panelオブジェクトはインスペクターを使って表示を設定することができる。WidthとHeightはpanelの幅と高さをピクセル数で指定するパラメーターだ。他のオブジェクトと同じように、panelオブジェクトの右下隅をドラッグすることで大きさを変えてもよい。Border Sizeは境界線の太さをピクセル数で指定する。これを0とすると境界線は描かれない。Shadow Sizeには陰影の幅をピクセル数で指定する。この数値を大きくすると手前に迫り出した立体的な表示になる。負の値であれば、奥にくぼんだような表示になる。Rounded Cornersは角の丸みの指定で、0の場合は直角だが、大きな値にすれば次第に角が丸くなる。panelオブジェクトの幅や高さと同じ数値にすれば、円形になる。一番下のカラー・スウォッチでは、panelオブジェクトの色を指定する。ポップアップ・メニューでBackgroundを選んでいるときはpanelオブジェクトの領域の色を、Frameを選んでいるときは境界線の色を指定することになる。

■6-5-7 panelオブジェクトとインスペクター



panelオブジェクトの簡単な利用方法としては、いくつかのオブジェクトのまとまりを示すために、グループ・ボックスとして用いることが考えられる。また、panelオブジェクトはいくつかのメッセージを受け取り、例えばbrgbメッセージで表示色を設定することが可能だ。そこで、swatchオブジェクトから出力されるメッセージにbrgbを付加すれば、swatchで選択した色を確認するためにpanelオブジェクトを用いることができる。

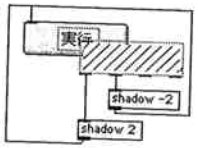
■6-5-8 panelオブジェクトの利用例



panelとともにcommentオブジェクトとubuttonオブジェクトを使えば、MacOS標準のプッシュ・ボタンに似た動作を実現できる。このためには、ubuttonオブジェクトのBorder Sizeを1、Shadow Sizeを2、Rounded Cornersを8とするのが適当だろう。色は薄いグレーにする。次に、panelオブジェクトの手前にcommentオブジェクトを置き、ボタンの表示文字として用いる。また、panelオブジェクトと同じ大きさのubutonオブジェクトを作成し、Highlight When Clickedのチェックを外しておく。これで、ubuttonオブジェクト自体はハイライト表示を行わないようになる。そして、ubuttonオブジェクトの第2アウトレットにshadow −2というメッセージ・ボックスをつなぎ、panelオブジェクトのインレットにつなぐ。ubuttonオブジェクトの第1アウトレットからはshadow 2というメッセージをpanelオブジェクトに送る。shadowメッセージはpanelオブジェクトの陰影の幅を設定するので、ubuttonオブジェクトがクリックされたときはくぼんだ表示になり、マウス・ボタンを離すと元の表示に戻るわけだ。

6-5-9では、分かりやすくするためにubuttonオブジェクトの位置をずらしているが、実際にはubuttonオブジェクトはpanelオブジェクトの大きさに一致するように配置する。

■6-5-9 Mac OS風のプッシュ・ボタンのパッチ



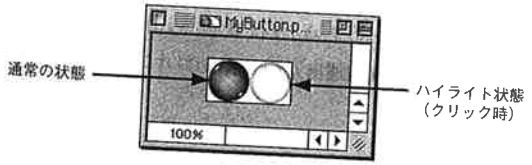
■6-5-10 Mac OS風のプッシュ・ボタンの表示

❷ pictctrlオブジェクトによるボタン、トグル、ダイアル

Maxに備わったユーザー・インターフェース・オブジェクトではできない特別な表示を行うために、pictctrl《Picture-based Control》オブジェクトとpictslider《Picture-based slider》オブジェクトが用意されている。これらを使用することで、ピクチャ・ファイルを使った独自の外観を持つオブジェクトが作成できる。このうち、pictctrlでは、ボタン、トグル、ダイアルの機能を持つオブジェクトを作成することができ、画像全体を入れ替えることで表示を変化させるようになっている。

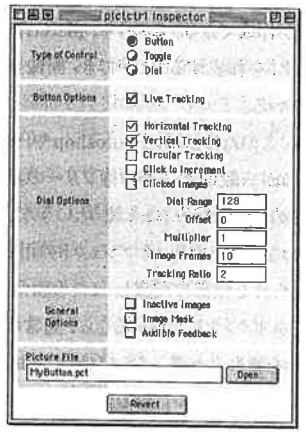
まずpictctrlで使用する画像を作成しよう。このためには、PhotoshopやPainterなどのグラフィック・アプリケーションを用いるのが一般的だ。著作権フリーの素材集などを利用するのもよいだろう。さらには、Maxのlcdオブジェクトを使うという方法も考えられる。作成する画像は任意の大きさで構わない。ただし、オブジェクトの用途によって、適切な数の画像を用意し、それらを定められた位置に配置し、1つの画像としてまとめなければならない。pictctrlオブジェクトをボタンとして用いるなら、通常の状態とハイライト状態の2つの画像を作成し、それらを横に並べる。6-5-11の例では、縦横とも32ピクセルの画像を並べているので、横幅は64ピクセルになる。作成した画像はファイルに保存する。ファイル・フォーマットはPICTが基本だが、QuickTimeが扱える数多くのフォーマットも利用できる。ファイルは、パッチと同じフォルダかMaxのサーチ・パス内に置くのがよいだろう。

■6-5-11 ボタン用の画像の例



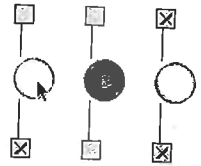
次に、Maxでpictctrlオブジェクトを作成する。pictctrlオブジェクトは、オブジェクト・パレットの後半にあるモノクロの人の顔が描かれたアイコンだ。アシスタンス・エリアにはPicture-based controlと表示される。pictctrlオブジェクトをパッチ・ウィンドウに作成すると、単に点線で矩形が描かれる。この状態では、表示すべき画像を指定していないためだ。そこで、pictctrlオブジェクトのインスペクターを開き、Picture Fileにファイル名を入力する。Open...ボタンをクリックして、ダイアログでファイルを選択するか、Finderから画像ファイルをドラッグして、テキスト・フィールドにドロップするのがよいだろう。ボタンの場合は、他の設定はそのままでよい。

■6-5-12　ボタン型pictctrlオブジェクトのインスペクター



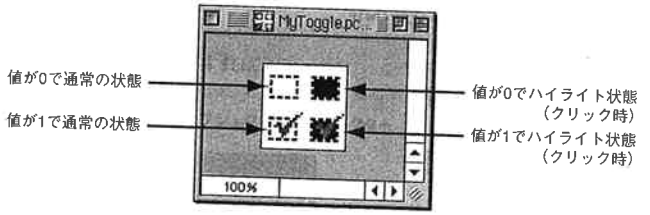
このようにして作成したボタン型のpictctrlオブジェクトをクリックするとハイライト画像が表示されるとともに1が出力され、マウス・ボタンを離すと0が出力されて通常の画像に戻る。bangメッセージを出力するbuttonオブジェクトとは動作が異なるので注意が必要だ。また、インレットに1を受け取るとハイライト状態になり、0を受け取るとノーマル状態になる。受け取った数値はそのまま出力される。

■6-5-13　ボタン型pictctrlオブジェクトの動作



次に、トグル型のpictctrlオブジェクトを作ろう。これを作るためには、値が0の場合の通常状態とハイライト状態、そして値が1の場合の通常状態とハイライト状態の4種類の画像が必要になる。

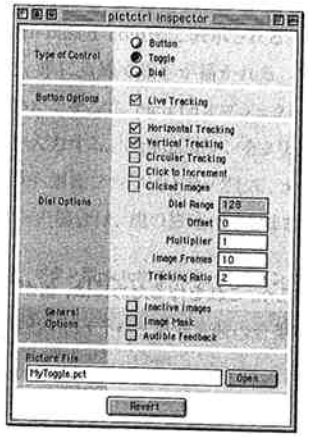
■6-5-14　トグル用の画像の例



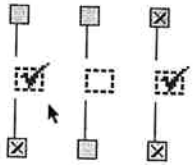
インスペクターを使ったpictctrlオブジェクトの設定では、Type of ControlをToggleにし、Picture Fileに画像ファイルの名前を設定する。

こうすることでトグル型のpictctrlオブジェクトをクリックする度に、値が1と0とに交互に変わり、その値を出力するようになる。インレットに値を受け取れば表示が変化し、その値が出力される。これはtoggleオブジェクトと同じ動作と考えてよいだろう。

■6-5-15 トグル型pictctrlオブジェクトのインスペクター



■6-5-16 トグル型pictctrlオブジェクトの動作



ダイアル型のpictctrlオブジェクトを作る際は、ダイアルがとる値の数だけの画像を横に並べればよい。もしも0から127までの値をとるとすれば128個の画像が必要になる。もっとも、表示の段階を粗くしても構わないなら、その半分や4分の1の数でもよいが、視覚的には細かな変化が確認できない。6-5-17では、ラジオの向きを変えた5種類の画像を用いている。

pictctrlオブジェクトのインスペクターでは、まず、Type of ControlとしてDialを選び、Picture Fileに画像ファイルの名前を設定する。そして、使用する画像の数である

■6-5-17 ダイアル用の画像の例

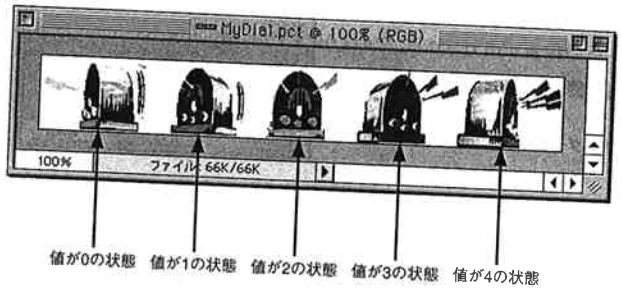
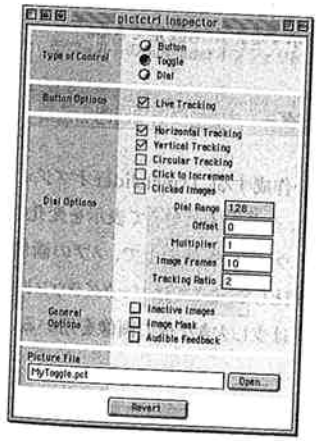


Image Framesに5を指定し、ダイアルの値の範囲のDial Rangeも5とする。これでダイアルは5段階の表示となり、0から4までの整数を出力する。

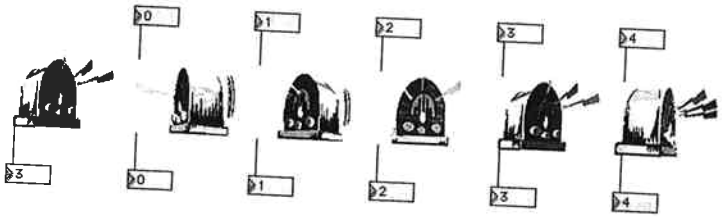
また、このダイアルは左右方向のマウス・ドラッグに対応すればよいので、Horizontal Trackingをチックし、Vertical Trackingはチェックしない。Tracking Ratioはマウスを何ピクセル動かせば値が変わるかという設定だ。ここでは10としたが、これは操作感に応じて適切な値を設定すればよい。

■6-5-18 ダイアル型pictctrlオブジェクトのインスペクター



作成したダイアル型pictctrlは、オブジェクトを左右にドラッグするとラジオの向きが変わり、それに応じた数値が出力されるだろう。また、受け取った数値に応じて表示が変わり、その数値が出力される。このようにダイアル型pictctrlは、dial《Dial》オブジェクトと同じ動作を行う。

■6-5-18 ダイアル型pictctrlオブジェクトの動作

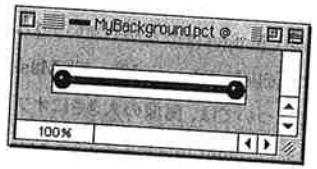


以上のように、独自に作成した画像とpictctrlオブジェクトを用いて、特別な表示を行うオブジェクトを作成することができる。ここでは最低限必要になる画像だけを用いたが、これら以外にもオブジェクトが操作できない状態の画像や、オブジェクトを重ね合わせたときの透過部分を指定するマスク画像も使用可能だ。また、ダイアル型pictctrlオブジェクトではドラッグ中に異なる画像を表示するようにも設定できる。

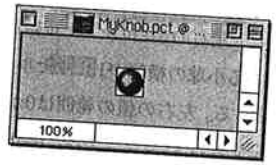
pictsliderオブジェクトによるスライダー

スライダー的なユーザー・インターフェースを作成するにはpictsliderオブジェクトを利用する。先に説明したpictctrlオブジェクトが、全体を入れ替えて表示を変化させるのに対して、pictsliderオブジェクトはバックグラウンドの画像の上で、ノブの画像を移動させることによってスライダーの表示を行う。例として、ここではバックグラウンドとして両端に小さな球が付いた横棒の画像と、ノブには少し大きな球の画像を用いる。これらは別々のピクチャ・ファイルとして保存しておく。

■6-5-20 バックグラウンドの画像の例

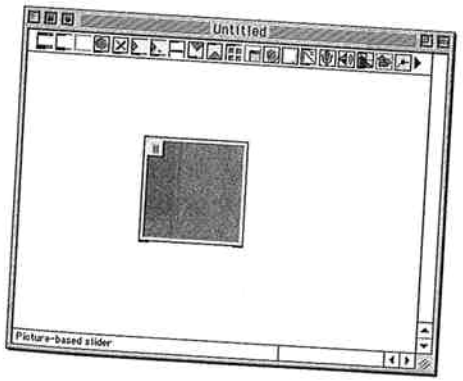


■6-5-21 ノブの画像の例



オブジェクト・パレットではpictctrlアイコンの右隣にpictsliderオブジェクトのアイコンがある。アシスタンス・エリアにはPicture-based sliderと表示される。pictsliderオブジェクトをパッチ・ウィンドウに作成すると、6-5-22のような平面上の2次元スライダーが現れる。もちろんこのスライダーをそのまま使っても構わないが、ここでは用意した画像を使って、横方向のスライダーにする。

■6-5-22 デフォルトのpictsliderオブジェクト

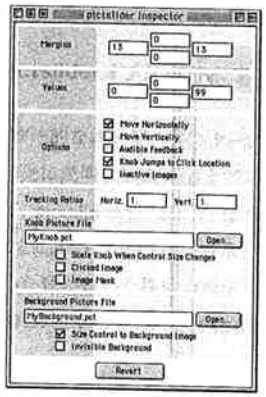


pictsliderオブジェクトのインスペクターでは、まずノブとバックグラウンドのピクチャ・ファイルをそれぞれ設定する。ノブの画像に関する設定では、ノブの大きさは変化させないので、Scale Knob When Control Size Changesはチェックしない。クリックしたときの画像やマスク画像は用いていないので、Clicked ImageとImage Maskはいずれもチェックを外す。バックグラウンドの画像については、画像の大きさにオブジェクトの大きさを合わせるので、Size Control to Background Imageをチェックする。つまり、画像を拡大や縮小をせずに用いることになる。Invisible Backgroundはチェックしないでおく。これをチェックすれば、バックグラウンドの画像が表示されなくなる。

次に、バックグラウンド画像の上でノブが移動可能な範囲を設定する。これはMarginsの4つの項目なので、ノブが移動しない上下左右の余白のピクセル数として指定する。ここでは、バックグラウンド画像の左右にある小球の横幅が13ピクセルなので、左右とも余白を13とし、上下の余白はないので0とする。左右の値の範囲は0から99までとする。これはオブジェクトの用途に応じて設定すればよい。ここでは、100ピクセルの移動量になるようにノブとバックグラウンドの画像を作っている。

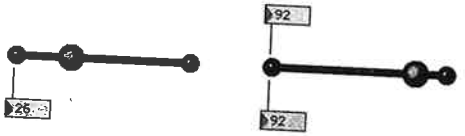
このノブは左右に動くが上下には動かないようにしたいので、Move Horizontallyをチェックし、Move Verticallyはチェックを外す。また、クリックした位置にノブが移動してもよいのでKnob Jumps to Click Locationにチェックを入れる。なお、オブジェクトが使用できないときの画像は作成していないので、Inactive Imagesはチェックしない。

■6-5-23　pictsliderオブジェクトのインスペクター



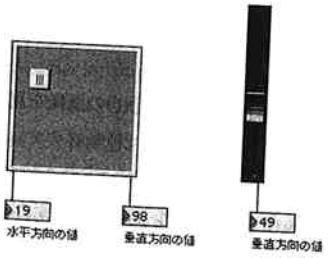
このようにして作成したpictsliderオブジェクトのノブを左右にドラッグすると、ノブの位置に応じた数値が第1アウトレットから出力される。また、第1インレットに受け取った数値に応じてノブの位置が変わり、その数値が出力される。つまり、このpictsliderオブジェクトはhslider《Horizontal Slider》オブジェクトと同じ動作をするわけだ。

■6-5-24　hsliderオブジェクトに似せた設定のpictsliderオブジェクト



ただし、pictsliderは2つずつアウトレットとインレットを持っている。左右水平方向のノブの値は第1アウトレットと第1インレットで扱い、上下垂直方向のノブの値は第2アウトレットと第2インレットで扱うようになっている。そのため、6-5-25のように、上下左右にノブが動く2次元スライダーにしたいなら、両方のアウトレットとインレットを利用すればよい。また垂直に動くスライダーとしてpictsliderオブジェクトを作成したときは、第2アウトレットと第2インレットだけを使うことになる。

■6-5-25　2次元スライダーと垂直スライダーとしてのpictsliderオブジェクト

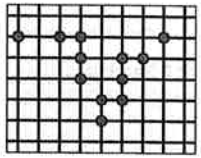


なお、pictsliderオブジェクトは通常状態の画像だけでなく、ノブにはクリックしたときの画像や使用不能時の画像を持たせることができ、バックグラウンドにも使用不能時の画像を設定することができる。また、ノブにはマスク画像の指定も可能だ。バックグラウンドの画像やノブの形状によっては、重ね合わさる部分が適切に表示されるようにマスク画像を作成しなければならない。

matrixctrlオブジェクトによるボタン配列

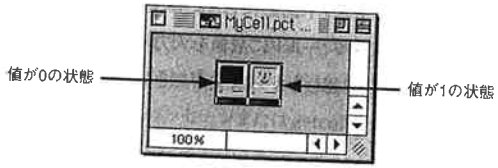
数多くのボタンが並んでいるユーザー・インターフェースを作るためにmatrixctrl《Matrix Control》オブジェクトが用意されている。matrixctrlオブジェクトはオブジェクト・パレットの後半にあり、黒い碁盤目と赤い点を持つアイコンで表されている。matrixctrlオブジェクトをパッチ・ウィンドウ上に作ると、黒い碁盤目のオブジェクトが現れる。matrixctrlオブジェクトの右下付近をドラッグすれば、オブジェクトの大きさを変えることができ、碁盤目の交差点をクリックすれば、赤い丸印が付く。もう1度クリックすれば消えてなくなる。matrixctrlオブジェクトは、ステップ・シーケンサーやマルチチャンネルのスイッチなどに活用できるだろう。

■6-5-26　パッチ・ウィンドウに作成したmatrixctrlオブジェクト



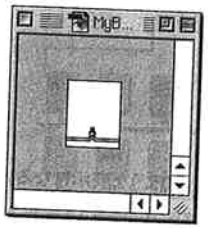
matrixctrlオブジェクトをそのまま用いても構わないが、独自の画像を用意して異なる表示や動作を設定することもできる。このためには画像は2種類必要で、1つはボタンに当たるセル画像だ。セル画像は、最低でも値が0の状態と値が1の状態の2つを作成する。そして、これらを横に並べて、1つのファイルとして保存する。ここでは、Macintoshのアイコンの電源が入っていない状態と、電源が入った状態の2つの画像を作成している。

■6-5-27　セルの画像の例



ここでもう1つ、matrixctrlオブジェクトの背景として用いるバックグラウンド画像を作らなければならない。バックグラウンド画像はパターンとしてmatrixctrlオブジェクトの背景に繰り返して表示される。ここでは、Macintoshアイコンをつなぐネットワーク・ケーブルの画像をバックグラウンドに用いることにする。

■6-5-28　バックグラウンドの画像の例

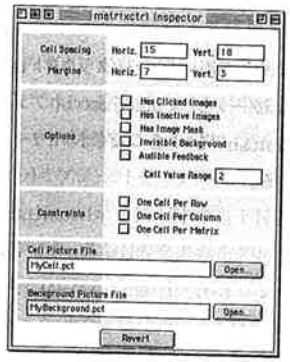


セル用とバックグラウンド用の画像ファイルを作成したなら、matrixctrlオブジェクトのインスペクターを開いて設定を行おう。最初にCell Picture Fileにセル画像のファイル名を設定し、Background Picture Fileにバックグラウンド画像のファイル名を設定する。ここではセルは2つの状態を用意したので、Cell Value Rangeは2とする。より多くの状態の画像を作成していれば、それに応じた数値を設定すればよい。

また、作例ではクリック時の画像や使用不可能時の画像は作っていないので、Has Clicked ImagesとHas Inactive Imagesのチェックを外しておこう。マスク画像もないので、Has Image Maskのチェックも不要だ。このような画像も含めてファイルを作成しているなら、それに応じてチェックを付けることになる。

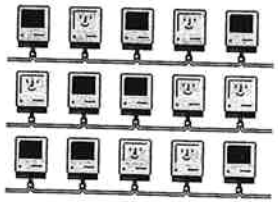
次にバックグラウンド画像をセル画像が適切に重なるように調整する。まず、左上のセル画像に注目して、これがバックグラウンド画像に合うようにMarginsの値を設定し、その後、全体のセル画像がバックグラウンド画像に合うようにCell Spacingの値を設定すればよいだろう。いずれもテキスト・フィールドに数値を入力してenterキーを押せば、画像の配置が更新されるので、目で確認しながら調整ができる。

■6-5-29 matrixctrlオブジェクトのインスペクター



以上のようにして作成したmatrixctrlオブジェクトは、次のような表示になる。セルをクリックする度に、Macintoshのアイコンが起動状態と終了状態に切り替わるだろう。もちろんこれと同じことを複数のpictctrlオブジェクトを使って実現することもできなくはないが、その作成は面倒であり、パッチの処理も複雑になる。

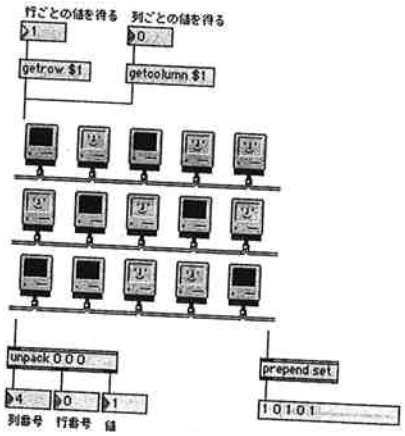
■6-5-30 matrixctrlオブジェクトの表示



セルをクリックすれば、matrixctrlオブジェクトの第1アウトレットから、3つの整数からなるリストが出力される。これは、クリックされたセルの列番号、行番号、そしてセルの値を示している。同じ形式で、3つの整数のリストをmatrixctrlオブジェクトに送れば、該当するセルの値を変更することもできる。ちなみに、列番号も行番号も0から始まる。6-5-31では、右下のセルの列番号は4、行番号は2になる。

セルの状態はgetrowメッセージまたはgetcolumnメッセージによって知ることができる。getrowメッセージは行番号のアーギュメントを伴い、指定した行のセルの値がリストとして第2アウトレットから出力される。同じように、getcolumnメッセージでは列ごとのセルの値が得られる。これらのメッセージを利用して、matrixctrlオブジェクトを使ったパッチを作ることができる。

■6-5-31 matrixctrlオブジェクトの動作



なお、インスペクターのConstrainsの各項目は、行や列、あるいはマトリクスで、有効なセルを1つに限定する設定だ。例えば、One Cell Per Matrixをチェックすると、どれか1つのセルだけが値が1の状態になる。したがって、あるセルをクリックすると、それまで値が1であったセルは自動的に値が0になる。これらは、matrixctrlオブジェクトを利用する目的に合わせて設定すればよい。