CONTENTS

コマンド

Cursor [*flags*] *cursorName traceName x_value*

Cursor [*flags*] *cursorName traceName x_value*, *y_value*

Cursor /K [/W=*graphName*] *cursorName*

Cursor /I [/F] [*flags*] *cursorName image_name x_value*, *y_value*

Cursor /M [*flags*] *cursorName*

Cursor コマンドは、*cursorName* で指定されたカーソルを、X 値が x_*value* である指定されたトレース上の位置、あるいは画像ピクセルの座標、または x_*value* と y_*value* の位置にある自由なカーソル位置に移動させます。

パラメーター

cursorName は、A から J までの 10 個のカーソルのいずれかです。

フラグ

/A=*a*　　*a*=0：　カーソルを無効にします。

a=1：　カーソルを有効にします。

アクティブなカーソルは、矢印キーまたはカーソルパネルを使って移動できます。

/C=(*r*, *g*, *b*, [*a*])

カーソルの色を設定します。*r*、*g*、*b*、*a* は、RGBA 値として色と（オプションの）不透明度を指定します。デフォルトは不透明な黒です。

/DF=*format*　　Graph Info パネルで日付・時刻データを表示する時に使う形式を設定します（ヘルプ Info Panel and Cursors を参照）。

/DF フラグは、Igor Pro 9.0 で追加されました。

format の値は以下の通りです：

0：　コンパクト形式：YYMMDD HHMM で表示されます。

1：　秒を含むコンパクトな形式：YYMMDD HHMMSS で表示されます。秒の部分には、必要に応じて秒の端数を表示することも可能です。詳細は、以下の /SDGT フラグを参照してください。

2：　日付と時刻は、より読みやすい形式で表示されます。これは、グラフの軸で「短い日付」形式を選択した時に表示される形式と同じです。時刻は HH:MM の形式で表示されます。

3：　日付と時刻（秒を含む）。時刻は HH:MM:SS の形式で表示されます。秒の部分には、必要に応じて秒の端数を表示することも可能です。詳細は、以下の /SDGT フラグを参照してください。

4：　日付を含まない時刻です。時刻は HH:MM:SS の形式で表示されます。必要に応じて秒の端数を表示することも可能です。詳細は、以下の /SDGT フラグを参照してください。

/DGTS=*nd*　グラフ情報パネルにカーソルの値を表示する時に使う有効桁数を設定します（ヘルプ Info Panel and Cursors を参照）。有効桁数は *nd* で設定され、1 から 15 までの値でなければなりません。

/DGTS フラグは、Igor Pro 9.0 で追加されました。

/F　カーソルを自由に移動させることができます。トレースまたは画像は、設定と読み取りのための x 座標と y 座標を定義する軸のペアを提供します。/P を使うと、相対座標で設定できます。この場合、0,0 は軸によって定義される矩形の左上隅、1,1 は右下隅となります。

フリーカーソルの基本座標はプロット相対座標です。トレースや画像を使用できるのは、読み出し座標の軸を見つけるための便利な手段にすぎません。

/H=*h*　カーソルに十字線を表示するように指定します。

h=0：　十字線を完全に非表示にする

h=1：　十字線を完全に表示する

h=2：　縦のヘアラインを表示する

h=3：　横のヘアラインを表示する

/I　指定された画像にカーソルを合わせます。

/K　指定されたカーソルを上のグラフから削除します。

/L=*lStyle*　十字線の線種（太線または細線）です。

lStyle=0：　実線

lStyle=1：　色が交互に変わる点線

/M　トレース座標や画像座標を指定することなく、プロパティを変更します。/F または /I フラグを指定した場合は機能しません。

/N=*noKill*　ユーザーがカーソルをプロット領域の外へドラッグした場合、カーソルが削除（Kill）されるかどうかを決定します。

noKill=0：　カーソルを削除する（デフォルト）

noKill=1：　カーソルを削除しない

/NUML=*n*　/H フラグの *h* が 0 以外の場合に組み合わせて使います。描画する十字線の本数を設定します。*n* は 1 から 3 の間の値でなければなりません。
n が 1 より大きい場合、線の間隔は /T=*t* フラグによって設定されます。
n=2 または 3 で、*t* が 3 未満の場合、線は *n*=1 の場合と同じように表示されます。
n=3 で、*t* が 5 未満の場合、表示は *n*=2 の場合に戻ります。線はカーソル位置を中心に左右対称に配置されます。*n*=3 の場合、t は外側の 2 本の線の間隔を設定します。

/NUML は Igor Pro 7.0 で追加されました。

/P　*x_value* を X 値ではなくポイント番号として解釈します。カーソルがウェーブのサブレンジを表すトレース上にある場合、ポイント番号は「トレース」のポイント番号となります。詳細は以下の「詳細」を参照してください。

/I フラグと組み合わせて使う場合、*x_value* と *y_value* は行番号と列番号を表します。

/F フラグと組み合わせて使う場合、*x_value* と *y_value* はグラフ上の相対座標（0～1）となります。

/S=*s*　　カーソルのスタイルを設定します。

s=0：　元の正方形または円

s=1：　文字が入った小さな十字線

s=2：　文字のない小さな十字線

/SDGT=*nd*　　秒を含む日付/時刻モードのいずれかで表示されている場合、または対応する軸に経過時間が表示されている場合に、グラフ情報パネル（ヘルプ Info Panel and Cursors を参照）に表示される小数点以下の桁数を設定します。*nd* は 0 から 6 までの値です。

/SDGT フラグは、Igor Pro 9.0 で追加されました。

/T=*t*　　/H オプションの *h* が 0 以外の場合、十字線の太さを設定します。/NUML オプションで線の数を 1 より大きい値に設定した場合、/T オプションは外側の 2 本の線の間隔を設定します。

t は、ピクセル単位での線の太さまたは間隔を表します。デフォルトは /T=1 です。

/T={*mode* , *t1* , *t2*} という形式を使うと、よりきめ細かなコントロールが可能になります。

/T は Igor Pro 7.0 で追加されました。

/T={*mode* , *t1* , *t2*}
/H オプションの *h* が 0 以外の場合、十字線の太さを設定します。/NUML オプションで線の数を 1 より大きい値に設定した場合、/T オプションで外側の 2 本の線の間隔を設定します。

mode=1 の場合、*t1* と *t2* は画面ピクセル単位となります。*t1* は垂直方向の線の太さまたは間隔であり、*t2* は水平方向の線の太さまたは間隔です。

デフォルトの十字線の外観は、/T={1,1,1} と同等です。

mode=0 の場合、*t1* と *t2* は軸座標の単位で表され、その結果、軸の範囲やグラフのサイズの変化に追従します。通常、*t1* は垂直線の太さまたは間隔、*t2* は水平線の太さまたは間隔を表しますが、トレースまたはグラフが XY 入れ替えモードになっている場合は、これらが入れ替わります。

/T は Igor Pro 7.0 で追加されました。

/W=*graphName*
特定のグラフウィンドウまたはサブウィンドウの名前を指定します。省略した場合は、その処理はアクティブなウィンドウまたはサブウィンドウに適用されます。

graphName を使ってサブウィンドウを特定する時は、ウィンドウ階層の構成に関する詳細について、ヘルプ Subwindow Syntax を参照してください。

詳細

通常、*traceName* はそのトレースによって表示されるウェーブ名と同じですが、インスタンス表記による名前である場合もあります。
トレース名とインスタンス表記に関する説明については、ModifyGraph コマンドを参照してください。

traceName を含む文字列を $ 演算子と組み合わせて使うことで、*traceName* を指定することができます。

x_value は、*traceName* によって表示されるウェーブの X スケールに基づく X 値です。
traceName が XY ペアとしてグラフ化される場合、x_*value* は X 軸の座標とは異なります。
グラフで XY ペアを表示する際には X スケールが無視されるため、/P フラグを使い、x_*value* にはポイント番号を指定することをお勧めします。

cursorName は名前であり、文字列ではありません。

カーソルの表示を表示するには、Graph メニューから Show Info を選択してください。

ウェーブのサブレンジを表すトレースにカーソルが設定されている場合、/P フラグを指定すると、x_*value* はウェーブのポイント番号ではなく、トレースのポイント番号として解釈されます。

例えば、トレースが次のコマンドによって作成された場合：

```
Display yWave[4,25;3]
```

したがって、x_value=0 はウェーブ点 4 に対応するトレースポイント 0 を参照し、x_*value*=1 はウェーブ点 7 に対応するトレースポイント 1 を参照します。

マクロや関数内でカーソルを移動しても、以前のカーソル位置はすぐには消去されません。
DoUpdate を明示的に呼び出す必要があります。

例

```
AppendToGraph myWave                    // myWave の X スケーリングによる X 座標
Cursor A, myWave, leftx(myWave)         // myWave の最初の点にあるカーソル A

AppendToGraph yWave vs xWave            // xWave からの X 座標であり、X のスケーリングではない
Cursor/P B, yWave,numpnts(yWave)-1      // yWave の最終ポイント上のカーソル B
DoUpdate
```

参照

DoUpdate コマンド
ヘルプ Programming With Cursors

CurveFit [*flags*] *fitType*, [*kwCWave=coefWaveName* ,] *waveName* [*flag parameters*]

WaveMetrics Web サイトの日本語のコマンドのヘルプセクションにある別ファイルを参照してください。

CustomControl [/Z] *ctrlName* [*keyword=value* [, *keyword=value* ...]]

CustomControl コマンドは、対象ウィンドウ内にカスタムコントロールを作成または変更します。
CustomControl は最初は一般的なボタンとして表示されますが、その外観と動作の両方をカスタマイズすることができます。

パラメーター

ctrlName は、作成または変更するカスタムコントロールの名前です。
標準のデフォルトパラメーターについては、Button コマンドを参照してください。

以下の keyword=value のパラメータがサポートされています。

align=*alignment*　コントロールの配置モードを設定します。配置モードは、pos キーワードに対する *leftOrRight* パラメーターの解釈をコントロールします。
align キーワードは Igor Pro 8.0 で追加されました。

alignment=0（デフォルト）の場合、*leftOrRight* はコントロールの左端の位置を指定し、コントロールのサイズが変更されても左端の位置は固定されたままになります。

alignment=1 の場合、*leftOrRight* はコントロールの右端の位置を指定し、コントロールのサイズが変更されても右端の位置は固定されたままになります。

fColor=(*r*, *g*, *b*[, *a*])

画像が使われておらず、かつ frame=1 の場合にのみ、ボタンの色を設定します。

r、*g*、*b*、*a* は、RGBA 値として色と（オプションの）不透明度を指定します。

focusRing=*fr*　キーボードフォーカスを示す矩形の表示を有効または無効にします。

fr=0：　フォーカス矩形は描画されません

fr=1：　フォーカス矩形が表示されます（デフォルト）

frame=*f*　画像が使われていない場合にのみ適用されるフレームスタイルを設定します：

f=0：　枠なし（タイトルのみが描画される）

f=1：　デフォルトでは、タイトルが中央揃えになったボタンが表示されます

ボタンに色を付けるには、fColor を黒以外の色に設定してください

f=2：　シンプルなボックス

f=3：　3D のくぼみフレーム

f=4：　3D 浮き彫りフレーム

f=5：　テキストハイライト

guides={*left*, *hcenter*, *right*, *top*, *vcenter*, *bottom*}

レイアウトガイドへのコントロールアンカーポイントの配置をコントロールします。詳細や例は、ヘルプ Laying Out Controls in Guides Mode を参照してください。

left、*hcenter*、*right*、*top*、*vcenter*、*bottom* のそれぞれは、適切な方向のレイアウトガイドの名前に置き換えられます。つまり、*left*、*hcenter*、*right* は垂直ガイドへの配置を示し、*top*、*vcenter*、*bottom* は水平ガイドへの配置を示します。配置されていないアンカーポイントを表すには、特別な名前 kwNone を使ってください。

コントロールのレイアウトに過度な制約を課すと、エラーが発生します。３つの水平または垂直のアンカーポイントのうち、レイアウトガイドに固定できるのは最大２つまでです。３つのうち１つを固定すると、レイアウトガイドが移動する時にコントロールも一緒に移動します。２つを固定すると、コントロールは移動するとともにサイズも変更されます。したがって、kwNone という名前は少なくとも２回出現することになります。

すべてのアンカーポイントに kwNone を指定すると、コントロールがレイアウトガイドから完全に切り離されます。

カスタムコントロールのユーザー定義アクションプロシージャでは、コントロールのサイズ変更や、レイアウトガイドの配置と競合する可能性のあるその他の操作が行われる場合があることに注意してください。コントロールに画像が使われている場合、レイアウトガイドによってコントロールのサイズが変更されると、画像が歪んでしまうことがあります。

help={*helpStr*}　コントロールのヘルプを設定します。

helpStr の最大長は 1970 バイトです（Igor Pro 8 とそれ以前のバージョンでは 255 バイト）。

引用符で囲まれた文字列の中に “\r” を入れると、改行を挿入できます。

labelBack=(*r*, *g*, *b*[, *a*]) または 0

画像が使われておらず、かつフレームが 1 ではない場合にのみ、コントロールの背景色を設定します。

r、*g*、*b*、*a* は、RGBA 値として色と（オプションの）不透明度を指定します。設定されていない場合（または labelBack=0 の場合）、背景は透明（消去されない）になります。

mode=*m*

コントロールに対して、何らかの事象が発生したことを通知します。あらゆる目的に使用できます。kCCE_mode イベントに関する詳細については、「詳細」の項を参照してください。

noproc

カスタムコントロールをクリックしても、プロシージャが実行されないように指定します。

picture=*pict*

指定されたプロシージャ画像を使って、コントロールを描画します。picture キーワードをこの形式で使うと、画像は 3 つのフレームが並んだものとみなされ、それぞれが通常の状態、マウスボタンが押されている状態、および無効状態におけるコントロールの外観を表します。

コントロールアクション関数では、picture={*pict*,*n*} という構文を使って、フレーム数を変更することができます。

コントロールの作成時に size キーワードが指定されていない場合、画像のサイズがコントロールの初期サイズとなり、幅は画像の幅の 3 分の 1 になります。コントロールのサイズを変更すると、画像が歪んでしまいます。

picture={*pict*,*n*}

指定されたプロシージャ画像を使ってコントロールを描画します。この画像は、デフォルトの 3 フレームではなく、n 個のフレームが並列に配置された構成となっています。

コントロールの作成時に size キーワードが指定されていない場合、画像のサイズがコントロールの初期サイズとなります。幅は、画像の幅を *n* で割った値になります。コントロールのサイズを変更すると、画像が歪んでしまいます。

pos={*leftOrRight*, *top*}

コントロールの配置モードが 0 の場合は、コントロールの左上隅の位置を、配置モードが 1 の場合は、コントロールの右上隅の位置を、それぞれコントロールパネル単位で設定します。詳細は、前述の align キーワードを参照してください。

pos+={*dx*, *dy*}

コントロールの位置を、コントロールパネル単位でオフセットします。

proc=*procName*

コントロールのアクション関数の名前を指定します。この関数は、コントロールやウィンドウを終了させてはなりません。

rename=*newName*

コントロールに新しい名前を付けます。

size={*width*, *height*}

コントロールのサイズをコントロールパネル単位で設定します。画像を使っている場合、画像が歪むことがあります。picture キーワードを参照してください。

title=*titleStr*

コントロールに表示されるテキストを指定します。

エスケープコードを使うと、タイトルのフォント、サイズ、書式、色を変更できます。詳細は、ヘルプ Annotation Escape Codes を参照してください。

userdata(*UDName*)=*UDStr*
名前なしのユーザーデータを *UDStr* に設定します。オプションの（*UDName*）を使って、作成する名前付きユーザーデータを指定します。

userdata(*UDName*)+=*UDStr*
名前なしのユーザーデータに *UDStr* を追加します。オプションの（*UDName*）を使うと、名前付きユーザーデータに追加することができます。

value=*varName*　コントロールに関連付けられた数値変数、文字列変数、またはウェーブを設定します。ウェーブの場合は、標準の角括弧表記を使って、ポイント番号（value=awave[4]）または行ラベル（value=awave[%alabel]）のいずれかでポイントを指定します。

valueColor=(*r*, *g*, *b*[, *a*])
画像が使われず、かつ frame=1 の場合にのみ描画されるボタンのタイトルの初期色を設定します。

r、*g*、*b*、*a* は、RGBA 値として色と（オプションの）不透明度を指定します。デフォルトは不透明な黒です。

タイトルテキストの色をさらに変更するには、title=*titleStr* で説明しているようにエスケープシーケンスを使ってください。

フラグ

/Z　エラーの報告はありません。

詳細

カスタムコントロールを作成する場合、アクションプロシージャは WMCustomControlAction 構造体を使って、そのコントロールの状態に関する情報を取得します。WMCustomControlAction 構造体の詳細については、WMCustomControlAction コマンドを参照してください。

現在、戻り値は使われていませんが、アクションプロシージャは常にゼロを返すようにしてください。

kCCE_draw イベントのアクションプロシージャを呼び出す時点で、基本ボタン画像（カスタムまたはデフォルト）はすでに描画済みとなっています。
DrawLine などの標準的な描画コマンドを使って、基本画像の上に描画を行うことができます。
通常、描画コマンドは描画操作リストに追加されるだけで、実際に描画されるのはそのあとになりますが、kCCE_draw イベントの描画コマンドは直接実行されます。
座標系は変更できませんが、コントロールパネルの単位系であり、(0,0) はコントロールの左上隅となります。
ほとんどの描画コマンドは使用可能ですが、描画が即座に行われる性質上、DrawPoly の /A (追加) フラグは使用できません。

mode キーワードを指定して CustomControl を実行した時に Igor Pro から送信される kCCE_mode イベントは、どのような目的にも使用できますが、主にコントロールに対して何らかのイベントが発生したことを通知する役割を果たします。
例えば、コントロールに情報を送信するには、名前付き（または名前のない）ユーザーデータを設定し、コントロールがそのユーザーデータを検査するように指示するモードを設定します。
このシグナリングの目的では、モード値として 0 を使う必要があります。
この値は、再作成マクロの一部には含まれないためです。

picture パラメーターで *pict* を指定すると、その *pict* がオフスクリーンビットマップに描画される時に kCCE_drawOSBM イベントが発生します。
一度作成されると、新しい picture パラメーターを指定するまで、すべての更新ではそのオフスクリーンビットマップが使われます。
従って、このイベントで行われるカスタム描画は静的であり、コントロールが描画されるたびに異なる可能性がある kCCE_draw イベント中に行われる描画とは異なります。

pict には複数のフレームが並列して含まれる可能性があるため、オフスクリーンビットマップの幅は、ctrlRect フィールドから導出された幅にフレーム数を乗じたものになります。

アクション関数は、さまざまなコントロールイベントの最中に呼び出されるため、コントロールやウィンドウを破棄してはなりません。
そうすると、ほぼ確実にクラッシュが発生します。

CustomControl アクションプロシージャ

CustomControl コントロールのアクションプロシージャでは、関数のパラメーターとして、あらかじめ定義された構造体 WMCustomControlAction を受け取ります。

```
Function ActionProcName(s)
    STRUCT WMCustomControlAction& s
    ...
    return 0
End
```

WMCustomControlAction 構造体の詳細は、WMCustomControlAction コマンドを参照してください。

現在、戻り値は使われていませんが、アクションプロシージャは常にゼロを返すようにしてください。

アクションプロシージャには、SetDrawEnv や DrawRect などの描画コマンドが含まれている場合があります。
これらのコマンドには、特定のウィンドウまたはサブウィンドウに対して操作を行うよう指定する /W フラグがあります。
ただし、CustomControl のアクションプロシージャでは、/W フラグは無視され、すべての描画コマンドは、そのコントロールを含むウィンドウまたはサブウィンドウに対して実行されます。

アクションプロシージャは、Igor Pro がクラッシュすることなく中断できない描画操作中に実行されるため、その実行中はデバッガを呼び出すことができません。
そのため、この関数に設定されたブレークポイントは無視されます。
代わりに、「Print ステートメントを使ったデバッグ（ヘルプ Debugging With Print Statements）」を使ってください。

例

カスタムコントロールの例は、ヘルプ Creating Custom Controls を参照してください。

参照

コントロールパネルとコントロールに関する詳細は、ヘルプ Controls and Control Panels を参照してください。

コントロールで使われる単位については、ヘルプ Control Panel Units の項を参照してください。

コントロールの情報については ControlInfo コマンドのセクションを参照してください。

指定されたユーザーデータの取得については GetUserData コマンドのセクションを参照してください。

ヘルプ Proc Pictures

TextBox、DrawPoly、DefaultGUIControls コマンド

デモ

メニュー File→Example Experiments→Feature Demos 2→Custom Control Demo.pxp

CWT [*flags*] *srcWave*

CWT コマンドは、1D の実数入力ウェーブ (*srcWave*) に対して連続ウェーブレット変換 (CWT) を計算します。
入力は任意の数値型で構いません。
計算された CWT は、現在のデータフォルダ内のウェーブ M_CWT に格納されます。
M_CWT は倍精度の 2D ウェーブであり、マザーウェーブレットや出力形式の設定によっては、複素数となる場合もあります。
M_CWT の次元は、オフセットおよびスケールの仕様によって決定されます。
この演算は、成功した場合は変数 V_flag を 0 に設定し、何らかの理由で失敗した場合は 0 以外の数値に設定します。

フラグ

/DEST=*destWave* コマンドによって生成される主な出力ウェーブを指定します（M_CWT に代わるものです）。

/DSTS=*destSWave* コマンドによって生成されるスケーリング出力ウェーブを指定します（W_CWTScaling に代わるものです）。

srcWave と宛先ウェーブの両方に同じウェーブを指定することはエラーとなります。

関数内で使われた場合、CWT 演算は *destWave* と *destSWave* に対して実数ウェーブ参照を作成します。詳細は、ヘルプ Automatic Creation of WAVE References を参照してください。

/ENDM=*method* 直接積分（/M=1）において、データ配列の両端を処理するために使う方法を選択します。

method=0： 配列の両端をゼロで埋めます。

method=1： 配列の先頭と末尾の両方で反射されます。

method=2： 循環的に繰り返される配列です。

/FREE *destWave* と *destSWave* をフリーウェーブとして作成します。

/FREE は関数内でのみ使用可能であり、かつ /DEST および /DSTS で指定された *destWave* と *destSWave* が単純な名前、またはウェーブ参照構造体のフィールドである場合に限り使用できます。

詳細は、ヘルプ Free Waves を参照してください。

/FREE フラグは、Igor Pro 10.0 で追加されました。

/FSCL 出力ウェーブの２次元ウェーブスケーリングに対して補正係数を適用し、数値がフーリエウェーブにより密接に関連するようにします。これらの補正係数の計算に関する詳細は、以下の参考文献を参照してください。このフラグは、Haar ウェーブレットからの出力には影響しません。

/M=*method* CWT の計算方法を指定します。

method=0： 高速なメソッドでは FFT を使います（デフォルト）。

method=1： 直接積分を用いた、より時間のかかる手法です。

基本的には、より効率的な FFT 法を使うべきです。直接法は、FFT 法では最適な結果が得られない場合にのみ使うべきです。理論的には、FFT 法が機能しない場合、例えばアンダーサンプリングされた信号の場合などでは、直接法もかなり不正確になるはずです。直接法の主な利点は、エッジ効果の調査に利用できることです。

/OSCW　　CWT は、画像プロットにおける Y ウェーブとして使用できる W_CWTScaling という名前の出力スケーリングウェーブを生成します。
例えば：

```
Display; AppendImage M_CWT vs {*,W_CWTScaling}
```

/OSCW を省略した場合、CWT は /SMP2=4 を指定した場合にのみ W_CWTScaling を作成します。

/OSCW は Igor Pro 9.0 で追加されました。

/OUT=*format*　　出力ウェーブ M_CWT の形式を決定します。

フォーマット	**M_CWT フォーマット**
1	複素数
2	実数値
3	実数であり、その絶対値を含む

計算方法やマザーウェーブレットの選択によっては、変換の「ネイティブ」な出力は実数または複素数となる場合があります。このフラグを使うことで、出力を希望の形式に強制的に設定することができます。

/Q　　静音モード：履歴に結果は出力されません。

/R1={*startOffset*, *delta1*, *numOffsets*}
CWT のオフセットを指定します。オフセットは CWT の第 1 次元です。通常は、*srcWave* が示すオフセットの全範囲に対して CWT を計算するため、このフラグを使用する必要はありません。ただし、低速なメソッドを使う場合、このフラグによってオフセットの出力範囲を制限し、計算時間を短縮することができます。*startOffset* (整数) は、*srcWave* 内の最初のオフセットのポイント番号です。*delta1* は、2 つの連続する CWT オフセット間の間隔です。これは *srcWave* のポイント数で表されます。*numOffsets* は、CWT が計算されるオフセットの数です。

デフォルトでは、*startOffset*=0、*delta1*=1 であり、*numOffsets* は *srcWave* のポイント数です。*startOffset* と *delta1* のみを指定したい場合は、*numOffsets*=0 に設定することで、ソースウェーブと同じポイント数を使用できます。

/R2={*startScale*, *scaleStepSize*, *numScales*}
CWT 計算におけるスケールの範囲を指定します。スケールは、出力ウェーブの第 2 次元です。ただし、データのサンプリングに関連して、最小スケールおよび最大スケールには制限があることに注意してください。フーリエ空間周波数と CWT スケールにはおおまかな対応関係があるため、理論上の最大スケールも存在することを理解しておく必要があります。これは、FFT を使って CWT を計算する場合に明らかですが、スロー法にも同様に当てはまります。許容範囲外の範囲を指定した場合、対応する CWT 値は NaN に設定されます。

このフラグのパラメーターにデフォルト値を使いたい場合は、NaN を指定してください。

startScale のデフォルト値は、ソースウェーブのサンプリングとウェーブレットパラメータまたは次数によって決定されます。

少なくとも、*scaleStepSize* または *numScales* のいずれかを指定する必要があります。

/SMP1=*offsetMode*　連続するオフセットの計算方法を決定します。現在、線形なユーザー指定の部分オフセット制限（/R1 フラグを参照）については、*offsetMode* = 1 のみをサポートしています：*val1 = startOffset + numOffsets * delta1*

/SMP2=*scaleMode*　連続するスケールの計算方法を決定します。/R2 フラグを指定した場合、*scaleMode* のデフォルト値は 1 になります。

scaleMode=1：　線形

```
theScale = startScale + index *scaleStepSize
```

scaleMode=2：　自動

```
theScale = numScales * startScale/(index+1)
```

scaleMode=4：　2 の累乗によるスケーリング間隔

```
theScale = startScale * 2.^(index *scaleStepSize)
```

scaleMode=4 を使う場合、このコマンドにより連続するスケール値がウェーブ W_CWTScaling に保存されます。また、対応する /R2 フラグを指定せずに *scaleMode*=4 を使うと、デフォルトの *scaleStepSize* が 1 でスケール値が 64 であるため、スケール値が許容範囲をすぐに超えてしまう点にも注意してください。

（上記の方程式で使われているさまざまなパラメータの詳細は、/R2 フラグを参照してください。）

/SUBM　変換を計算する前に、入力の平均値を引き算します。
/SUBM は Igor Pro 9.0 で追加されました。

/SW2=*sWave*　変換が評価される具体的なスケール値を指定します。/R2 フラグの代わりに使います。ウェーブ内のエントリが *srcWave* のサンプリング密度に適していることを確認する必要があります。

/WBI1={*Wavelet* [, *order*]}
組み込みのウェーブレット（母関数）を指定します。ウェーブレットは、マザーウェーブレットの名前です：Morlet（既定値）、MorletC（複素数）、Haar、MexHat、DOG、Paul。

Morlet: :: $\Psi_0(x) = \frac{1}{\pi^{1/4}} \cos(\omega x) \exp\left(-\frac{x^2}{2}\right)$

デフォルトでは、ω=5 です。ω に他の値を指定するには、/WPR1 フラグを使ってください。

MorletC: :: $\Psi_0(x) = \frac{1}{\pi^{1/4}} \exp(i\omega x) \exp\left(-\frac{x^2}{2}\right)$

デフォルトでは、ω=5 です。ω に他の値を指定するには、/WPR1 フラグを使ってください。

Haar: :: $\Psi_0(x) = \begin{cases} 1 & 0 \le x < 0.5 \\ -1 & 0.5 \le x < 1 \end{cases}$

Dog: :: $$\Psi_0(x) = \frac{(-1)^{m+1}}{\sqrt{\Gamma\left(m+\frac{1}{2}\right)}} \frac{d^m}{dx^m}\left(\exp\left(-\frac{x^2}{2}\right)\right)$$

MexHat: *m*=2 の場合の DOG の特別なケース。

Paul: :: Paul:: $$\Psi_0(m,x) = \frac{2^m i^m m!}{\sqrt{\pi(2m)!}}(1-ix)^{-(m+1)}$$

この順序は、DOG ウェーブレットと Paul ウェーブレットにのみ適用され、ウェーブレットファミリーの特定の要素である *m* を指定します。

デフォルトのウェーブレットは Morlet です。

/WPR1={*param1*}

param1 は、/WBI1 で選択されたウェーブレット関数に固有のパラメーターです。例えば、/WPR1={6} を使うと、Morlet の周波数をデフォルト値（5）から変更できます。

/Z エラーは報告されません。エラーが発生した場合、V_flag を -1 に設定しますが、関数の実行は中断されません。

詳細

CWT は、その定義積分から直接計算することも、また、その積分が畳み込みを表しており、高速フーリエ変換（FFT）を使って効率的に計算できるという事実を利用することによっても計算することができます。

FFT 法を使う場合、典型的なサンプリング問題やエッジ効果が生じます。エッジ効果はスロー法を使う場合にも見られますが、高スケールでのみ顕著になります。

サンプリングに関する考察から、離散入力信号の最大周波数は 1/2dt であることが示されます。
ここで、dt は２つのサンプリング値間の時間間隔です。
従って、CWT の最小スケールは 2dt で、最大スケールは Ndt となります。
ここで、N は入力ウェーブの総サンプリング数です。

M_CWT の変換は、ウェーブスケーリングとともに保存されます。
startOffset と *delta1* は、X 軸のスケーリングに使われます。
startOffset と *delta1* は、いずれも /R1 フラグで指定されるか、*srcWave* からコピーされます。
M_CWT の Y 軸のスケーリングは、/SMP2 の設定によって異なります。
CWT のスケーリングが線形の場合、ウェーブスケーリングは *startScale* と *scaleStepSize* に基づいて行われます。
2 の累乗のスケーリング間隔を使っている場合、M_CWT の Y ウェーブスケーリングは *start*=0 と *delta*=1 となり、ウェーブ W_CWTScaling には M_CWT の各列に対する実際のスケーリング値が含まれます。
なお、W_CWTScaling には、次のような操作による表示に適するように、１つの追加データポイントが含まれていることに注意してください：

```
AppendImage M_CWT vs {*, W_CWTScaling}
```

文献では、Morlet ウェーブレットについて２つの異なる定義が見受けられます。
１つ目は複素関数であり、２つ目は実関数です。
これらの定義のいずれかを選択する代わりに、適切なウェーブレットを選べるよう、両方を実装しました。

参考文献

Torrence, C., and G.P. Compo, A Practical Guide to Wavelet Analysis, *Bulletin of the American Meteorological Society,79*, 61-78, 1998.

Torrence と Compo によるこの論文は、以下の URL でもオンラインで閲覧できます：
<http://paos.colorado.edu/research/wavelets/>。

参照

FFT コマンド
離散ウェーブレット変換を行うには、DWT コマンドを使います。

デモ

メニュー File→Example Experiments→Analysis→CWT Demo

Debugger

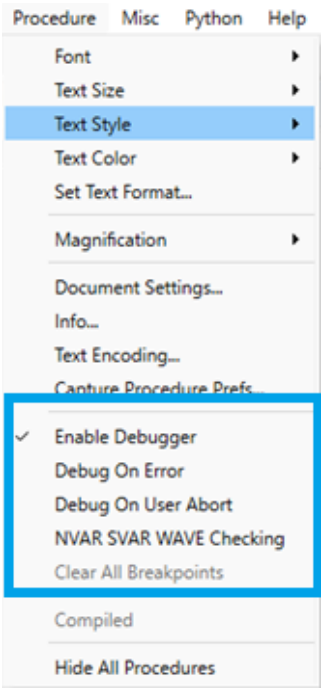
Debugger コマンドは、有効になっている場合、デバッガーに切り替えます。

参照

DebuggerOptions コマンド
ヘルプ The Debugger

DebuggerOptions [*enable=en*, *debugOnAbort=doa*, *debugOnError=doe*, *NVAR_SVAR_WAVE_Checking=nvwc*]

DebuggerOptions コマンドは、ユーザーレベルのデバッガーの設定をプログラムで変更します。
これらは、Procedure メニュー（およびデバッガーのソースペインのコンテキストメニュー）で利用可能な 3 つの設定と同じものです。



パラメーター

すべてのパラメーターはオプションです。何も指定しない場合、何の処理も行われませんが、出力変数は設定されます。

enable=*en*

デバッガーを有効または無効にします。

en=0： デバッガーを無効にします。

en=1： デバッガーを有効にします。

デバッガーが無効になっている場合、他の設定が有効になっていても、それらの設定はすべてクリアされます。

debugOnAbort=*doa*

「ユーザーによる中断時のデバッグ」を有効または無効にします。

doa=0：Debug On Abort を有効にします。

doa=1：Debug On Abort を有効にし、デバッガーも有効にします（enable=1 と同等）。

Debug on User Abort 機能は、Igor Pro 9.0 で追加されました。詳細は、ヘルプ Debugging on User Abort を参照してください。

debugOnError=*doe*

「エラー時のデバッグ」を有効または無効にします。

doe=0：Debugging On Error を無効にします。

doe=1：Debugging On Error を有効にし、デバッガーも有効にします（enable=1 を意味します）。

NVAR_SVAR_WAVE_Checking=*nvwc*

NVAR、SVAR、ウェーブのチェックを有効または無効にします。

nvwc=0： NVAR SVAR WAVE Checking を無効にします。詳細は、ヘルプ Accessing Global Variables And Waves を参照してください。

nvwc=1： このチェックを有効にし、デバッガーも有効にします（enable=1 を意味します）。

出力変数

DebuggerOptions は、コマンドの実行後に有効となるデバッガーの設定を示すために、以下の変数を設定します。
値が 0 の場合は設定がオフ、0 以外の場合は設定がオンであることを意味します。

```
V_enable, V_debugOnError, V_debugOnAbort, V_NVAR_SVAR_WAVE_Checking
```

参照

Debugger コマンド
ヘルプ The Debugger

DefaultFont [/U] "*fontName*"

DefaultFont コマンドは、グラフ内の軸ラベル、目盛ラベル、注釈、ページレイアウト内の注釈に使われるデフォルトのフォントを設定します。

パラメーター

"*fontName*" にはフォント名を指定します。
必要に応じて引用符で囲むこともできます。
フォント名が 1 語の場合は、引用符は不要です。

フラグ

/U 既存のグラフやページレイアウトを直ちに更新し、新しいデフォルトのフォントを使うようにします。

DefaultGUIControls [/W=*winName* /Mac/Win] [*appearance*]

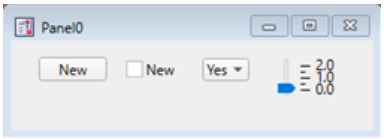
DefaultGUIControls コマンドは、ユーザー定義コントロールの外観を変更します。

注記
ユーザー定義コントロールの外観を変更する推奨方法は、Miscellaneous Settings ダイアログの Compatibility タブにある "Native GUI Appearance for Controls" チェックボックスを使うことです。このチェックボックスをオンにすると DefaultGUIControls native と同等になり、オフにすると DefaultGUIControls os9 と同等になります。

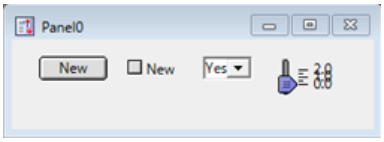
パラメーター

appearance は、以下のいずれかにすることができます。

native　現在のコンピュータープラットフォームに合わせて、標準的な外観のコントロールを作成します。これがデフォルト値です。



os9　Igor Pro 5 の外観（Macintosh OS 9 風の操作画面）。



default　親ウィンドウまたはエクスペリメント全体のデフォルト設定のいずれかからウィンドウの外観を継承します（/W オプション指定時のみ有効）。

フラグ

/Mac　Igor Pro 9 またはそれ以前のバージョンを実行している Macintosh コンピューターでのみ、コントロールの外観を変更します。

/W=*winName*　指定されたウィンドウまたはサブウィンドウに適用されます。省略した場合は、エクスペリメント全体に適用されるデフォルト値が設定されます。

winName を使ってサブウィンドウを特定する時は、ウィンドウ階層の構成に関する詳細について、ヘルプ Subwindow Syntax を参照してください。

/Win　Windows でのみ、コントロールの外観を変更します。

詳細

appearance が指定されていない場合、何も変更されません。
appearance の現在の値が S_value に返されます。

appearance が指定された場合、ウィンドウ全体またはエクスペリメント全体のデフォルト値として設定されていた以前の *appearance* 値が S_value に返されます。

/W オプションを指定すると、コントロールの表示設定は指定されたウィンドウ（Graph または Panel）にのみ適用されます。
このオプションを指定しない場合、設定は現在のコンピューター上で実行されるすべてのエクスペリメントに共通して適用されます。

> **注記**
> /W=# を使うと、現在アクティブなサブウィンドウを指定できます。

/Mac と /Win フラグは、対象となるコンピュータープラットフォームを指定します。
指定されたプラットフォームと現在のプラットフォームが異なる場合、その設定は使われませんが、（ネイティブまたは OS9 の場合）、ウィンドウ再作成マクロやエクスペリメントの再作成で使用できるよう記憶されます。
つまり、現在のプラットフォームに応じて外観が異なるエクスペリメントを作成することができます。

/Mac も /Win も指定しない場合は、現在のプラットフォームが暗黙的に使われます。
両方のプラットフォームでネイティブな外観を設定するには、次の 2 つのコマンドを使います。

```
DefaultGUIControls/W=Panel0/Mac native
DefaultGUIControls/W=Panel0/Win native
```

注記
/W オプションを指定しない場合の DefaultGUIControls の設定は、エクスペリメントファイルには保存されません。
これは、Miscellaneous Settings ダイアログの Compatibility タブにある "Native GUI Appearance for Controls" チェックボックスで設定されるプレファレンスです。
DefaultGUIControls native または DefaultGUIControls os9 コマンドを使う場合、このチェックボックスにはエクスペリメント全体の設定の現在の状態は表示されません。
Miscellaneous Settings ダイアログで Save Settings をクリックすると、DefaultGUIControls の設定が上書きされます（ただし、ウィンドウごとの設定は上書きされません）。

DefaultGUIControls は、コントロールが作成される前に使うように設計されていますが、これを呼び出すと、開いているすべてのウィンドウが更新されます。

DefaultGUIControls はコントロールのフォントやフォントサイズを変更しないため、位置のずれや重なりを防ぐために位置を再調整することなく、「ネイティブ風」な外観のコントロールを作成することができます。

ネイティブの外観は、TabControl と GroupBox コントロールにおけるコントロールの描画方法に影響を与えます。

TabControl の背景の詳細

os9 のタブ表示では、タブ領域を定義するために輪郭線のみが描画され（中央部分はそのまま残される）、これとは異なり、ネイティブのタブ表示ではタブ領域全体が塗りつぶされます。
幸いなことに、TabControl は他のあらゆる種類のコントロールよりも先に描画されるため、ウィンドウ再作成マクロ内でボタンが定義されている順序に関係なく、囲まれたコントロールを TabControl の上に重ねて描画することができます。

ただし、ネイティブの TabControl の描画順序は重要です。
最上層の TabControl は、他の TabControl の上に描画されます。
（最上層の TabControl は、ウィンドウ再作成マクロ内で最後にリストされています。）
os9 の外観では、タブは通常塗りつぶされていないため、より小さい（ネストされた）TabControl を、それより後に描画される（外側の）TabControl の下に配置することができます。
これらのタブをネイティブの外観に変換すると、ネストされたタブが非表示になります。

既存のパネルにおける描画順序の問題を修正するには、描画ツールを有効にし、矢印ツールを選択して、そのパネルを囲んでいる TabControl を右クリックし、Send to Back を選択して修正してください。
TabControl 自体が別の TabControl 内に含まれている場合は、その外側の TabControl を選択し、同様に Send to Back を選択してください。

パネルを作成したウィンドウ再作成マクロまたは関数を修正するには、囲んでいる TabControl コマンドを、囲まれた TabControl を作成するコマンドよりも先に実行されるように配置してください。

ネイティブで描画される TabControl は、タブ領域に完全に囲まれた描画オブジェクトを描画するため、内部に描画オブジェクトを含む os9 の非塗りつぶし TabControl と同じ動作をします。

GroupBox コントロールの背景の詳細

GroupBox コントロールは、TabControl とは異なり、他のすべてのコントロールよりも先に描画されるわけではないため、GroupBox に背景色（塗りつぶし色）が指定されており、かつ他のコントロールが含まれている場合、描画順序は常に重要になります。

ネイティブ表示を有効にすると、GroupBox 内のいくつかのコントロールが表示されなくなる場合があります。
これらは、描画順序において GroupBox の下（手前）にあるためと考えられます。

既存のパネルでこの問題を修正するには、描画ツールを有効にし、GroupBox を右クリックして Send to Back を選択します。

パネルを作成したウィンドウ再作成マクロまたは関数を修正するには、GroupBox のコマンドを、その中に含まれるコントロールを作成するコマンドよりも先に実行されるように配置してください。

ネイティブで描画された GroupBox は、そのボックスに完全に囲まれている描画オブジェクトをすべて描画しますが、os9 の塗りつぶし機能を使った GroupBox ではそうはなりません。

参照

DefaultGUIFont、ModifyControl、Button、GroupBox、TabControl コマンド

コントロールパネルとコントロールに関する詳細は、ヘルプ Controls and Control Panels を参照してください。

コントロールで使われる単位については、ヘルプ Control Panel Units の項を参照してください。

デモ

メニュー File→Example Experiments→Feature Demos 2→All Controls Demo

DefaultGUIFont [/W=*winName* /Mac/Win] *group*={*fNameStr*, *fSize*, *fStyle*}
[, ...]

DefaultGUIFont コマンドは、ユーザー定義コントロールやその他のグラフィカルインターフェイス要素の既定のフォントを変更します。

パラメーター

fNameStr はフォント名、*fSize* はサイズ、*fStyle* はビット単位のパラメーターであり、各ビットがフォントスタイルの１つの側面をコントロールします。
これらのパラメーターの詳細については、Button コマンドを参照してください。

group は、以下のいずれかになります：

all　すべてのコントロール

button　ボタンとデフォルト CustomControl

checkbox　CheckBox コントロール

tabcontrol　TabControl コントロール

popup　PopupMenu コントロールのアイコン（タイトルではなく）に使われるテキストのフォントに影響します。ポップアップ状態でのテキストはシステムによって設定されるため、変更することはできません。PopupMenu のタイトルは「all」グループの影響を受けますが、アイコンのテキストは影響を受けません。

panel　パネル内のテキスト描画に使うデフォルトのフォントを設定します。

graph　オーバーレイグラフのデフォルトのフォントを設定します。
サイズは、ModifyGraph gfSize= -1; の場合にのみ使われ、スタイルは使われません。

table　オーバーレイテーブルのデフォルトのフォントを設定します。

フラグ

/Mac　この変更は、Igor Pro 9 以前のバージョンが動作する Macintosh コンピューターでのみ、フォントに影響します。

/W=*winName*　指定されたウィンドウまたはサブウィンドウに適用されます。省略した場合は、エクスペリメント全体に適用されるデフォルト値が設定されます。

winName を使ってサブウィンドウを特定する時は、ウィンドウ階層の構成に関する詳細について、ヘルプ Subwindow Syntax を参照してください。

/Win　この変更は Windows でのみフォントに影響し、Windows で使われるたびにエクスペリメントの結果に影響を及ぼします。

詳細

コントロールが作成される前に使うように設計されていますが、DefaultGUIFont を呼び出すと、コントロールが含まれるすべてのウィンドウが更新されます。
これにより、フォントによるエクスペリメントが容易になります。
ただし、マシンやプラットフォーム間を移動する際、フォントによって互換性の問題が生じる可能性があることに留意してください。

/Win フラグは、フォントを使うプラットフォームを指定します。
現在のプラットフォームが指定されたプラットフォームと異なる場合、その設定は適用されませんが、ウィンドウ再作成マクロやエクスペリメントの再実行時に使用できるよう記憶されます。
これにより、ユーザーは現在のプラットフォームに応じて異なるフォントを使うエクスペリメントを作成することができます。

/W フラグを使って場合、フォント設定は指定されたウィンドウ（グラフまたはパネル）にのみ適用されます。
/W フラグを使わない場合、設定はエクスペリメント全体に適用されます。

注記

/W=# を使うと、現在アクティブなサブウィンドウを指定できます。

fNameStr に空文字列（「」）を指定すると、グループがクリアされます。
フォント名を “_IgorSmall”、”_IgorMedium”、または ”_IgorLarge” に設定すると、Igor Pro 独自のデフォルト設定が使われます。

デフォルトのフォントとサイズ

Igor Pro のコントロールの標準デフォルト設定は、すべてを “_IgorSmall”、タブコントロールを ”_IgorMedium”、ボタンを “_IgorLarge” に設定したものと同等です。
サイズについても標準のデフォルト設定にするには、*fSize* に 0 を指定してください。
３つのデフォルトフォントとサイズはすべて同じです。

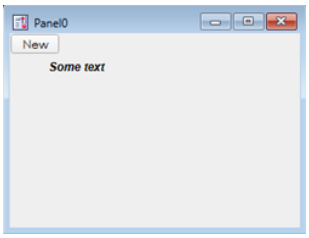
コントロール	フォント	フォントサイズ
Button	MS Shell Dlg+	12
Checkbox	MS Shell Dlg	12
GroupBox	MS Shell Dlg	12
ListBox	MS Shell Dlg	12
PopupMenu*	MS Shell Dlg	12
SetVariable	MS Shell Dlg	12
Slider	MS Shell Dlg	12
TabControl	MS Shell Dlg	12
TitleBox	MS Shell Dlg	12
ValDisplay	MS Shell Dlg	12

* ポップアップメニューのフォント、あるいはポップアップメニュー自体については、どちらもフォントは MS Shell Dlg 12 です。

+ MS Shell Dlg は「仮想フォント名」で、Windows XP では Tahoma、Windows 7 では MS Sans Serif、Windows 8 および Windows 10 では Segoe UI に割り当てられています。

例

```
DefaultGUIFont/Mac all={"Zapf Chancery",12,0},panel={"geneva",12,3}
DefaultGUIFont/Win all={"Century Gothic",12,0},panel={"arial",12,3}
NewPanel
Button b0
DrawText 40,43,"Some text"
```



参照

DefaultGUIControls コマンド

コントロールパネルとコントロールに関する詳細は、ヘルプ Controls and Control Panels を参照してください。

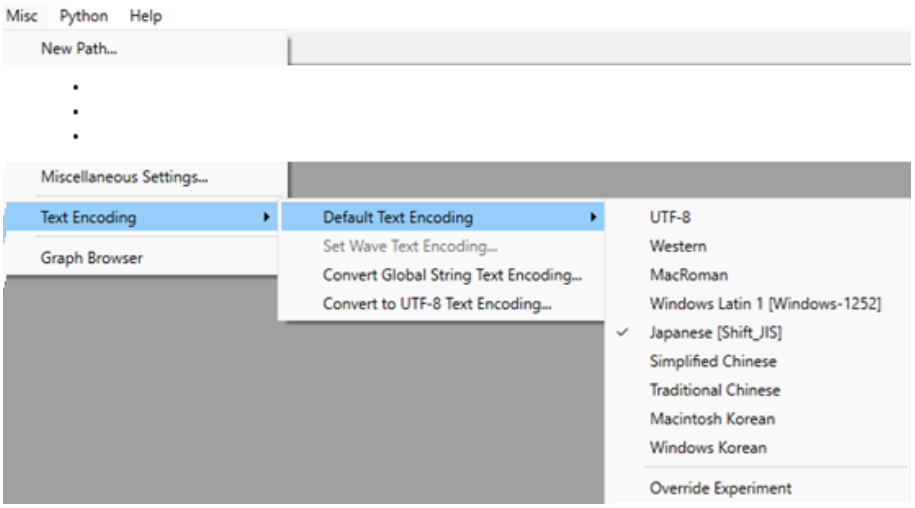
コントロールで使われる単位については、ヘルプ Control Panel Units の項を参照してください。

デモ

メニュー File→Example Experiments→Feature Demos 2→All Controls Demo

DefaultTextEncoding [*encoding=textEncoding, overrideDefault=override*]

DefaultTextEncoding コマンドは、プログラムによってデフォルトのテキストエンコーディングとエクスペリメント用テキストエンコーディングのオーバーライド設定を変更します。
ヘルプ The Default Text Encoding の項で説明しているこれらの設定は、Misc→Text Encoding→Default Text Encoding メニューからもアクセスできます。



DefaultTextEncoding が必要になることはほとんどありません。
通常、デフォルトのテキストエンコーディングを変更する必要がある場合でも、メニューから手動で変更することになるためです。

DefaultTextEncoding コマンドは、Igor Pro 7.0 で追加されました。

パラメーター

すべてのパラメーターはオプションです。
何も指定しない場合、何の処理も行われませんが、出力変数は設定されます。

encoding=*textEncoding*　*textEncoding* は、新しいデフォルトのテキストエンコーディングを指定します。コードの一覧については、ヘルプ Text Encoding Names and Codes を参照してください。

0 を指定すると、デフォルトのテキストエンコーディングが、Default Text Encoding サブメニューから “Western” を選択した場合と同等になります。

バイナリテキストエンコーディングタイプに対応する値「255」は、textEncoding パラメーターに対して無効な値として扱われます。

overrideDefault=*override*　エクスペリメントのテキストエンコーディングのオーバーライドを有効または無効にします。

0：　オーバーライドを無効にする

1：　オーバーライドを有効にする

詳細

デフォルトのテキストエンコーディングは、テキストエンコーディングが不明なファイルを開く時の動作に影響します。
詳細は、ヘルプ The Default Text Encoding を参照してください。

エクスペリメントのテキストエンコーディングは、エクスペリメントの読み込み時にファイルがどのように扱われるかにのみ影響します。
オーバーライド設定を使うと、エクスペリメントに保存されているテキストエンコーディングを上書きすることができます。
通常、この設定を行う必要はありません。
詳細は、ヘルプ The Default Text Encoding を参照してください。

Igor Pro のコマンドの中には、テキストエンコーディングを指定するための /ENCG フラグがなく、代わりに現在のデフォルトのテキストエンコーディングが使われるものがあるため、デフォルトのテキストエンコーディングを変更する必要がある場合があります。
そのような場合は、元のデフォルトのテキストエンコーディングを保存し、必要に応じて変更した後、元のテキストエンコーディングに戻すことを推奨します。
以下の例では、この手法について説明しています。

出力変数

DefaultTextEncoding は、コマンドの実行後に有効となる設定を示すために、以下の出力変数を設定します。

V_defaultTextEncoding　テキストエンコーディングのコード

V_overrideExperiment　値が 0 の場合は設定がオフ、0 以外の場合は設定がオンであることを意味します。

例

```
Function DemoDefaultTextEncoding()
    // 元のデフォルトのテキストエンコーディングを保存する
    DefaultTextEncoding
    Variable originalTextEncoding = V_defaultTextEncoding

    // 新しいデフォルトのテキストエンコーディングを設定する
    DefaultTextEncoding encoding = 3       // 3 = Windows-1252

    // [ デフォルトのテキストエンコーディングに依存する処理を行う ]
    // 元のデフォルトのテキストエンコーディングに戻す
    DefaultTextEncoding encoding = originalTextEncoding
End
```

参照

ヘルプ The Default Text Encoding、Text Encoding Names and Codes

DefineGuide [/W=*winName*] *newGuideName*={[*guideName1*, *val* [, *guideName2*]]} [, ...]

DefineGuide コマンドは、対象のウィンドウ、指定されたウィンドウ、またはサブウィンドウ内に、ユーザー定義のガイドラインを作成または上書きします。
ガイドラインは、ホストウィンドウ内でのサブウィンドウの位置決めに役立ちます。

パラメーター

newGuideName は、新しく作成されるガイドの名前です。
既存のユーザー定義ガイドの名前である場合、そのガイドは新しい位置に移動されます。

guideName1、*guideName2* などには、既存のガイドの名前を指定する必要があります。

val の意味は、コマンド構文の形式によって異なります。
ガイド名を１つだけ指定する場合、*val* はそのガイドを基準とした相対的なオフセット値となります。
val の正の値は、ガイドの右側または下側を指します。
単位はポイントですが、パネル内ではコントロールパネルの単位が使われます。
２つのガイド名を指定する場合、*val* は２つのガイド間の距離の比率を表します。

フラグ

/W=*winName*　指定されたウィンドウまたはサブウィンドウ内のガイドに作用します。省略された場合、アクションはアクティブなウィンドウまたはサブウィンドウに作用します。

winName を使ってサブウィンドウを特定する時は、ウィンドウ階層の構成に関する詳細について、ヘルプ Subwindow Syntax を参照してください。

詳細

組み込みガイドの名称は、次の表に定義されているとおりです。

ガイド名	左	右	上	下
サブウィンドウフレーム	FL	FR	FT	FB
外のグラフの矩形範囲	GL	GR	GT	GB
内側のプロット矩形範囲	PL	PR	PT	PB
レイアウトの余白範囲	ML	MR	MT	MB

フレームガイドは、すべてのウィンドウとサブウィンドウの種類に適用されます。
グラフ矩形ガイドとプロット矩形ガイドは、グラフウィンドウとサブウィンドウにのみ適用されます。
レイアウト余白矩形ガイドは、レイアウトウィンドウにのみ適用されます。

ガイドを削除するには、GuideName={} を使ってください。

参照

Display、Edit、NewPanel、NewImage、NewWaterfall、GuideInfo コマンド

DelayUpdate

DelayUpdate コマンドは、マクロの実行中にグラフや表の更新を遅延させます。

詳細

マクロ内の次の行を、グラフやテーブルが更新される前に実行させたい場合は、その行の末尾に 'DelayUpdate' を指定してください。

これはユーザー定義関数では効果がありません。
ユーザー定義関数では実行中、DoUpdate コマンドを明示的に呼び出した場合にのみウィンドウを更新します。

参照

DoUpdate、PauseUpdate、ResumeUpdate コマンド

DeleteAnnotations [*flags*] [*tagOffscreen*, *tagTraceHidden*, *invisible*, *offsetOffscreen*, *tooSmall*[=*size*]]

DeleteAnnotations コマンドは、出力変数 S_name に、フラグとキーワードで指定された理由により非表示となっている注釈を一覧表示し、必要に応じてそれらを削除します。

このコマンドは、/W フラグで指定されたウィンドウまたはサブウィンドウに対して行われます。
/W が省略された場合は、アクティブなウィンドウまたはサブウィンドウに対して行われます。

DeleteAnnotations を使って、特定の 1 つの注釈をプログラムで削除しないでください。
代わりに、以下を使ってください：

```
TextBox/W=winName/K/N=annotationName
```

/LIST フラグを指定すると、注釈を削除するのではなく、一覧表示のみを行うように制限されます。

DeleteAnnotations コマンドは、Igor Pro 7.0 で追加されました。

キーワード

キーワードは、非表示にされた理由に基づいて注釈を識別します。

invisible　/V=0 で非表示にされた注釈を削除または一覧表示します。

offsetOffscreen　通常、/X および /Y オフセットが大きすぎるために画面外にある注釈を削除または一覧表示します。

tagOffscreen　画面外にあるトレースポイントに紐付けられているために非表示になっているタグを削除または一覧表示します。これは、Modify Annotation ダイアログの Position タブで設定された "if offscreen" の設定が "hide the tag" に設定されている場合、トレースタグ、軸タグ、および画像タグに影響します。

tagTraceHidden　タグが付けられたトレースが非表示になっているために非表示になっているタグを削除または一覧表示します。

tooSmall [=*size*]
高さまたは幅が *size* ポイント以下の注釈を削除または一覧表示します。*size* はポイント単位で指定され、デフォルト値は 8 です。これは、小さすぎて見づらかったり、ダブルクリックできなかったりする注釈を削除するのに便利です。

フラグ

/A　非表示かどうかにかかわらず、すべての注釈が一覧表示されるか、削除されます。すべてのキーワードは無視されます。

/LIST　他のパラメーターで指定された注釈を、出力変数 S_name に一覧表示するが、削除は行わないことを指定します。

/W=*winName*　*winName* を使ってサブウィンドウを特定する時は、ウィンドウ階層の構成に関する詳細について、ヘルプ Subwindow Syntax を参照してください。

出力変数

S_name　キーワードとフラグによって設定された条件に一致する注釈の、セミコロン区切りのリストです。

V_flag　削除または一覧表示された注釈の数に設定されます。

例

```
Function DeleteAnnotationsInWin(win)
    String win              // 最上位ウィンドウまたはサブウィンドウを指定する

    // 指定された最上位ウィンドウまたはサブウィンドウを処理する
    DeleteAnnotations/W=$win/A
    Variable numDeleted = V_Flag

    // 次に、サブウィンドウがある場合は、それらを処理する
    String children = ChildWindowList(win)
    Variable n = ItemsInList(children)
    Variable i
    for(i=0; i<n; i+=1)
        String child = StringFromList(i, children)
        numDeleted += DeleteAnnotationsInWin(child)     // 再帰
    endfor

    return numDeleted
End
```

参照

TextBox、StringFromList、AnnotationList コマンド

DeleteFile [/I/M=*messageStr* /P=*pathName* /Z] [*fileNameStr*]

DeleteFile コマンドは、ディスク上のファイルを削除します。

パラメーター

fileNameStr には、削除対象のファイルへのフルパス（この場合は /P は不要）、*pathName* に関連付けられたフォルダーを基準とした部分パス、または *pathName* に関連付けられたフォルダー内のファイル名を指定できます。

fileNameStr と *pathName* からファイルの場所を特定できない場合、削除するファイルを指定できるダイアログが表示されます。

いずれかのファイルに対してフルパスまたは部分パスを使う場合は、パスの構成方法の詳細については、ヘルプ Path Separators を参照してください。

フラグ

/I　対話モード。*fileNameStr* が指定され、ファイルが存在する場合でも、Open File ダイアログが表示されます。

/M=*messageStr*　Open File ダイアログに表示されるプロンプトメッセージを指定します。

/P=*pathName*　ファイルを検索するフォルダーを指定します。*pathName* は、既存のシンボリックパスの名前です。

/Z[=*z*]　存在しないファイルを削除しようとした場合でも、プロシージャの実行が中断されないようにします。実行を中断させるのではなく、プロシージャ内でこのケースを処理したい場合は、/Z オプションを使ってください。

- /Z=0：　/Z を指定しない場合と同じです。
- /Z=1：　ファイルが存在する場合にのみ、そのファイルを削除します。/Z のみを指定した場合、/Z=1 を指定した場合と同じ効果があります。
- /Z=2：　ファイルが存在する場合はそれを削除し、存在しない場合はダイアログを表示します。

変数

Delete コマンドでは、以下の変数に情報が返されます。

V_flag　ファイルが削除された場合は 0、ユーザーが Open File ダイアログをキャンセルした場合は -1、指定されたファイルが存在しないなどのエラーが発生した場合は 0 以外の値に設定されます。

S_path　削除されたファイルへのフルパスを格納します。エラーが発生した場合、またはユーザーがコマンドをキャンセルした場合は、空の文字列に設定されます。

参照

DeleteFolder、MoveFile、CopyFile、NewPath, コマンド
ヘルプ Symbolic Paths

DeleteFolder [/I/M=*messageStr* /P=*pathName* /Z] [*folderNameStr*]

DeleteFolder コマンドは、ディスク上のフォルダーとその中身をすべて削除します。

注意

DeleteFolder コマンドはデータを完全に削除します！
削除されたフォルダーとその内容は、ゴミ箱やリサイクルビンには移動されません！

DeleteFolder は、ユーザーから許可が与えられた場合にのみフォルダーを削除します。
デフォルトでは、許可を求めるダイアログが表示されます。

ユーザーは、Miscellaneous Settings ダイアログの Misc カテゴリから、この動作を変更することができます。

権限が拒否された場合、フォルダーは削除されず、V_Flag は 1088（コマンドが無効です）または 1276（フォルダーの削除権限が拒否されました）を返します。
/Z フラグが指定されていない限り、コマンドの実行は中止されます。

folderNameStr には、削除対象のフォルダーのフルパスを指定できます。
その場合は /P は不要です。
また、*pathName* で指定されたフォルダーを基準とした部分パス、あるいは *pathName* で指定されたフォルダー内のフォルダー名を指定することもできます。

folderNameStr と *pathName* からフォルダーの場所を特定できない場合、削除するフォルダーを指定できる Select Folder ダイアログが表示されます。

/P=*pathName* が指定され、*folderNameStr* が指定されていない場合、pathName に関連付けられたフォルダーが削除されます。

いずれかのフォルダーに対してフルパスまたは部分パスを使う場合は、パスの構成方法の詳細についてヘルプ Path Separators を参照してください。

フォルダーパスは、1 つのパス区切り文字で終わってはいけません。
MoveFolder コマンドの「詳細」セクションを参照してください。

フラグ

/I　対話モードでは、*folderNameStr* が指定され、そのフォルダーが存在する場合でも、Select Folder ダイアログが表示されます。

/M=*messageStr*　Select Folder ダイアログに表示されるプロンプトメッセージを指定します。

/P=*pathName*　フォルダーを検索する対象のフォルダーを指定します。*pathName* は、既存のシンボリックパスの名前です。

/Z[=*z*]　存在しないフォルダーを削除しようとした場合でも、プロシージャの実行が中断されないようにします。実行を中断させるのではなく、プロシージャ内でこのケースを処理したい場合は、/Z オプションを使ってください。

- /Z=0：　/Z を指定しない場合と同じです。
- /Z=1：　フォルダーが存在する場合にのみ、そのフォルダーを削除します。/Z のみを指定した場合、/Z=1 を指定した場合と同じ効果があります。
- /Z=2：　フォルダーが存在する場合はそれを削除し、存在しない場合はダイアログを表示します。

変数

DeleteFolder コマンドでは、以下の変数に情報が返されます。

V_flag　フォルダーが削除された場合は 0、ユーザーが Select Folder ダイアログをキャンセルした場合は -1、指定されたフォルダーが存在しないなどのエラーが発生した場合は 0 以外の値に設定されます。

S_path　削除されたフォルダーへのフルパスを格納します。エラーが発生した場合、またはユーザーがコマンドをキャンセルした場合は、空の文字列に設定されます。

詳細

削除するソースフォルダーを指定するには、/P=*pathName*（*folderNameStr* を省略）のみを使用できます。

フォルダーパスは、1 つのパス区切り文字で終わってはいけません。
MoveFolder コマンドの「詳細」セクションを参照してください。

参照

DeleteFile、MoveFolder、CopyFolder、NewPath、IndexedDir コマンド
ヘルプ Symbolic Paths

DeletePoints [/M=*dim*] *startElement*, *numElements*, *waveName* [, *waveName*]...

DeletePoints コマンドは、指定されたウェーブから、*startElement* を起点として *numElements* 個の要素を削除します。

フラグ

/M=*dim* *dim* は、要素を削除する次元を指定します。値は、行の場合は 0、列の場合は 1、レイヤーの場合は 2、チャンクの場合は 3 です。/M を省略した場合、DeletePoints は行次元から削除を行います。

詳細

ウェーブには、0 を含む任意の数のポイントが存在することができます。
いずれかの次元からすべての要素を取り除くと、ウェーブからすべてのポイントが取り除かれ、ポイントが 0 個の 1D ウェーブが残ります。

すべての要素を削除する場合を除き、DeletePoints はウェーブの次元数を変更しません。
次元数を変更するには、Redimension を使ってください。

参照

Redimension コマンド

デモ

メニュー File→Example Experiments→Techniques→Delete Points From Wave

メニュー File→Example Experiments→Techniques→Delete XY Points Demo

Differentiate [/DIM=*d* /EP=*e* /METH=*m* /P][*typeFlags*] *yWaveA* [/X=*xWaveA*] [/D=*destWaveA*][, *yWaveB* [/X=*xWaveB*][/D=*destWaveB*][, ...]]

WaveMetrics Web サイトの日本語のコマンドのヘルプセクションにある別ファイルを参照してください。

Dir [*dataFolderSpec*]

Dir コマンドは、指定されたデータフォルダー内のすべてのオブジェクトの一覧を返します。

パラメーター

dataFolderSpec を省略した場合は、現在のデータフォルダーが使われます。

dataFolderSpec が存在する場合、その値は、現在のデータフォルダー内の子データフォルダーの名前のみ、または（現在のデータフォルダーを基準とした）部分パスと名前、あるいは（ルートから始まる）絶対パスと名前のいずれかになります。

詳細

出力される情報の形式は、文字列関数 DataFolderDir で使われる形式と同じです。
Igor Pro プログラマーにとっては、CountObjects や GetIndexedObjName を使ったほうが便利かもしれません。

通常は、Igor ブラウザー（Data メニュー）を使うほうが簡単です。
ただし、名前をコマンドラインにコピーしたい場合や、フォルダーの現在の状態を履歴として記録したい場合には、Dir が便利です。

参照

ヘルプ Data Folders

Display [/B[=*axisName* /HIDE=*h* /FG=(*gLeft*, *gTop*, *gRight*, *gBottom*) /HOST=*hcSpec* /I /K=*k* /L[=*axisName*]/M /N=*name* /PG=(*gLeft*, *gTop*, *gRight*, *gBottom*) /R[=*axisName*] /T[=*axisName*] /VERT/W=(*left*, *top*, *right*, *bottom*)] [*waveName* [, *waveName*] ... [*vs xwaveName*]] [*as titleStr*]

Display コマンドは、新しいグラフウィンドウまたはサブウィンドウを作成し、指定されたウェーブがある場合はそれらを追加します。
ウェーブは 1D トレースとして表示されます。

デフォルトでは、ウェーブは左軸および下軸に対してプロットされます。
他の軸に対してウェーブをプロットするには、/L、/B、/R、/T フラグを使ってください。

パラメーター

コマンドラインの長さが 2500 バイト以内である限り、最大 100 個のウェーブ名を指定できます。
ウェーブ名が指定されていない場合は、空白のグラフが作成され、軸のフラグは無視されます。

vs xwaveName を指定すると、指定されたウェーブの Y 値が、*xwaveName* の Y 値に対してプロットされます。
vs xwaveName を指定しない場合、各 *waveName* の Y 値は、それ自体の X 値に対してプロットされます。

xwaveName がテキストウェーブまたは特別なキーワード「_labels_」である場合、結果として得られるプロットはカテゴリプロットになります（カテゴリプロットを参照）。
waveName の各要素は、デフォルトでバーモード（ModifyGraph mode=5）でプロットされ、*xwaveName* の対応する要素のテキスト、または最初の Y ウェーブの次元ラベルのテキストでラベル付けされたカテゴリに対してプロットされます。

カテゴリプロットのウェーブフォームには、ポイントスケーリングが適用されるべきです（ウェーブフォームのスケーリングに関する詳細は、ヘルプ The Waveform Model of Data を参照）。
これが本来の意図であるため、その方がより適切に動作します。

titleStr は、グラフのタイトルを含む文字列式です。
指定しない場合、グラフに表示されているウェーブを識別できるタイトルを自動的に付与します。

行列の個々の行や列を含むデータのサブセットは、サブレンジ（一部）表示構文を使って指定することができます。

waveName の指定に /TN=*traceName* を追加することで、トレースに独自の名前を付けることができます。
これは、同じ名前を持ちますが異なるデータフォルダーからのウェーブを表示する場合に便利です。
詳細については、ヘルプ Trace Name Parameters を参照してください。

フラグ

/B[=*axisName*]　　X 座標を、標準の、または名前が指定された下側の軸に対してプロットします。

/FG=(*gLeft*, *gTop*, *gRight*, *gBottom*)
ホストウィンドウ内で、サブウィンドウの外枠がどのフレームガイドに固定されるかを指定します。

フレームガイドの標準名称は、左、右、上、下のフレームガイドについてそれぞれ FL、FR、FT、FB ですが、ホストで定義されたユーザー定義のガイド名を使うこともできます。

ガイドは、/W で設定された数値による配置を上書きすることができます。

/HIDE=*h*　　ウィンドウを非表示（*h*=1）または表示（*h*=0、デフォルト）にします。

/HOST=*hcSpec*　　*hcSpec* で指定されたホストウィンドウまたはサブウィンドウに、新しいグラフを埋め込みます。

hcSpec を使ってサブウィンドウを特定する時は、ウィンドウ階層の構築に関する詳細について、ヘルプ Subwindow Syntax を参照してください。

/I　　/W の座標がインチ単位であることを指定します。

/K=*k*　　ユーザーがウィンドウを閉じようとした時の動作を指定します。

k=0：　ダイアログ付きの標準の動作です（デフォルト）。

k=1：　ダイアログなしで Kill します。

k=2：　Kill を無効にします。

k=3：　ウィンドウを非表示にします。

k=4：　ダイアログなしで Kill し、エクスペリメントの結果にも保存されません。

/K=2 または /K=3 を指定した場合でも、KillWindow コマンドを使ってウィンドウを終了させることができます。

/L[=*axisName*]　　Y 座標を、標準軸または名前が付けられた左軸に対してプロットします。

/M　　/W の座標がセンチメートル単位であることを指定します。

/N=*name*　　作成されるグラフに、その名前がまだ使われていない場合は、この名前を付けるよう要求します。
すでに使われている場合は、*name0*、*name1* といった順に試行し、未使用のウィンドウ名が見つかるまで続けます。

/NCAT　　Igor Pro 6.37 以降では、数値プロットにカテゴリトレースを後から追加できるようになりました。詳細は、ヘルプ Combining Numeric and Category Traces を参照してください。

/PG=(*gLeft*, *gTop*, *gRight*, *gBottom*)
ホストウィンドウ内のグラフサブウィンドウの内側のプロット矩形を指定します。

プロット矩形ガイドの標準的な名前は、それぞれ左、右、上、下のプロット矩形ガイドに対して PL、PR、PT、PB であり、あるいはホストで定義されたユーザー定義のガイド名も使用できます。* を使うと、デフォルトのガイド名を指定できます。

ガイドは、/W で設定された数値による位置指定を上書きすることができます。

/R=[*axisName*]　　Y 座標を、標準軸または指定された右軸に対してプロットします。

/T=[*axisName*]　　Y 座標を、標準軸または指定された上軸に対してプロットします。

/TN=*traceName*　トレースに独自の名前を付けることができます。これは、同じ名前を持ちますが異なるデータフォルダーからのウェーブを表示する時に便利です。詳細は、ヘルプ User-defined Trace Names を参照してください。

/VERT　データを縦軸でプロットします。SwapXY（ModifyGraph）と似ていますが、トレースごとに処理が行われます。

/W=(*left*, *top*, *right*, *bottom*)
グラフを画面上の特定の位置とサイズに配置します。/W の座標は、/W の前に /I または /M が指定されていない限り、ポイント単位で指定されます。

/HOST フラグと併用する場合、指定された位置座標には、次の 2 つの意味のいずれかが適用されます。

1. すべての値が 1 未満の場合、座標はホストフレームのサイズに対する相対的な小数値であるとみなされます。
2. いずれかの値が 1 より大きい場合、座標は、ホストフレームの左上隅を基準として、ポイント単位、またはコントロールパネルホストの場合はコントロールパネル単位で測定された固定位置とみなされます。

ガイドを使ってサブウィンドウの位置を完全に指定した場合（/HOST、/FG、または /PG フラグを使用）、/W フラグは必須ではありませんが、引き続き使うことができます。

詳細

/N オプションを使わない場合、Display は自動的に「Graph*n*」という形式の名前（*n* は整数）をグラフに割り当てます。
関数やマクロ内では、割り当てられた名前が S_name 文字列に格納されます。
これは、プロシージャからグラフを参照する時に使用できる名前です。
グラフの名前を変更するには、RenameWindow コマンドを使ってください。

例

コンタープロットを作成するには、

```
Display; AppendMatrixContour waveName
```

または、

```
Display; AppendXYZContour waveName
```

とします。

画像を表示するには、

```
Display; AppendImage waveName
```

または、

```
NewImage waveName
```

とします。

参照

AppendToGraph コマンド

コンタープロットを作成するには、Display; AppendMatrixContour または Display;　AppendXYZContour を使います。

画像を表示するには、NewImage または Display; AppendImage を使います。

ModifyGraph を使って、トレースや軸などの表示を変更します。

DisplayHelpTopic [/K=*k* /Z] *TopicString*

DisplayHelpTopic コマンドは、ヘルプファイル内のヘルプリンクがクリックされた場合と同様に、ヘルプトピックを表示します。

パラメーター

TopicString は、トピックを含む文字列式です。
その形式は、<トピック名>、<サブトピック名>、<トピック名>[<サブトピック名>] の 3 つのいずれかになります。
これらの形式については、以下の例で説明します。

Igor Pro が意図したヘルプファイル以外のヘルプファイルからトピックを見つけてしまう可能性を最小限に抑えるため、トピック文字列は具体的に指定してください。
この問題を回避するには、可能であれば <トピック名>[<サブトピック名>] という形式を使うのが最善です。

フラグ

/K=*k*　　k=0：　ヘルプファイルを無期限に開いたままにします（デフォルト）。どのエクスペリメントにおいてもそのヘルプトピックが参考になる可能性がある場合に、この設定を使ってください。

k=1：　見つかったトピックが閉じられたヘルプファイル内にある場合、そのヘルプファイルは現在のエクスペリメント終了時に閉じられます。ヘルプトピックが現在のエクスペリメントと密接に関連している場合に、この機能を使ってください。

/Z　　エラーを無視します。/Z オプションが指定された場合、DisplayHelpTopic は、ヘルプトピックが見つかったときは V_Flag を 0 に、見つからなかったときは 0 以外のエラーコードに設定します。V_Flag は、/Z オプションが指定された場合にのみ設定されます。

詳細

DisplayHelpTopic は、まず開いているヘルプファイルの中から指定されたトピックを検索します。
トピックが見つからない場合は、Igor Pro フォルダーとそのサブフォルダー内のすべてのヘルプファイルを検索します。

それでもトピックが見つからない場合、現在のエクスペリメントのホームフォルダー内にあるすべてのヘルプファイルを検索しますが、サブフォルダー内は検索対象外となります。
これにより、特定のエクスペリメント専用のヘルプファイルが、そのエクスペリメントのホームフォルダーに配置されることになります。

それでもトピックが見つからず、かつ DisplayHelpTopic が Igor プロシージャから呼び出され、そのプロシージャがディスク上の独立したファイル（つまり、組み込みプロシージャウィンドウやパックされたプロシージャファイル内ではない）にある場合、そのプロシージャファイルのフォルダー内にあるすべてのヘルプファイルを検索しますが、サブフォルダー内は検索しません。
これにより、特定のプロシージャ群専用のヘルプファイルが、そのプロシージャファイルと同じフォルダーに配置されます。

Igor Pro がそのトピックを見つけた場合は、それを表示します。
そのトピックを見つけられなかった場合は、/Z が指定されていない限り、エラーメッセージが表示されます。

例

```
// この例では、トピックのみを使います。
DisplayHelpTopic "Modifying Traces"

// この例では、サブトピックのみを使います。
DisplayHelpTopic "Markers"

// この例では、トピック[サブトピック]という形式を使います。
DisplayHelpTopic "Modifying Traces[Markers]"
```

参照

ヘルプ Igor Help Files

DisplayProcedure [/B=*winTitleOrName* /L=*lineNum* /W=*procWinTitle*] [*functionOrMacroNameStr*]

DisplayProcedure コマンドは、指定された関数、マクロ、または行が表示されているプロシージャウィンドウを最前面に表示し、その関数、マクロ、または行を強調表示します。

パラメーター

functionOrMacroNameStr は、表示する関数またはマクロの名前を含む文字列式です。
functionOrMacroNameStr を省略する場合は、/W または /L を使う必要があります。

functionOrMacroNameStr は単純な名前である場合もあれば、静的関数を表示するために独立したモジュールやモジュール名のプレフィックスを含む場合もあります。

/L オプションを使って特定の行を表示する場合は、functionOrMacroNameStr を省略する必要があります。

スクロールや選択状態を変更せずにプロシージャウィンドウを表示するには、/W を使い、*functionOrMacroNameStr* を省略します。

フラグ

/B=*winTitleOrName*
この名前またはタイトルを持つウィンドウのすぐ後ろに、プロシージャウィンドウを表示します。

/L=*lineNum*
/W が指定された場合、*lineNum* は指定されたウィンドウ内の 0 を起点とする行番号となります。

/W が指定されていない場合、*lineNum* は「グローバル」な行番号となります。各プロシージャウィンドウの行には、開いているすべてのプロシージャファイルが 1 つの大きなファイルに連結されたかのように、一意のグローバル行番号が割り当てられます。プロシージャが再コンパイルされると、ファイルの連結順序が変わる場合があります。

/L を使う場合は、*functionOrMacroNameStr* を省略する必要があります。

/W=*procWinTitle*
このタイトルを含むプロシージャウィンドウ内の検索です。

procWinTitle は名前であり、文字列ではないため、/W は次のように指定します：

```
/W=$"New Polar Graph.ipf"
```

/W を省略した場合、DisplayProcedure は開いているすべての（独立モジュールではない）プロシージャウィンドウを検索します。

詳細

プロシージャウィンドウに構文エラーがあり、関数やマクロの開始位置と終了位置を特定できない場合、DisplayProcedure はそのプロシージャを検出できない可能性があります。

winTitleOrName は文字列ではなく、名前です。
タイトルにスペースが含まれるウィンドウの背後に、検索されたプロシージャウィンドウを配置するには、以下の 2 番目の例のように $ 演算子を使ってください。

winTitleOrName に一致するウィンドウがない場合、見つかったプロシージャウィンドウは、最上位のターゲットウィンドウの背後に配置されます。

lineNum は 0 を起点とする行番号です。0 はウィンドウの 1 行目です。
プロシージャウィンドウの各行は段落であるため、行番号と段落番号は一致します。
Procedure→Info メニューを使うと、選択範囲の開始および終了の段落番号／行番号を表示できます。

procWinTitle も名前の一つです。
/W= $"New Polar Graph.ipf" を使うと、そのプロシージャファイル内のみから関数やマクロを検索できます。

functionOrMacroNameStr と /L=*lineNum* の両方を指定しないでください。
これは曖昧であり、許可されていません。

高度な詳細

SetIgorOption IndependentModuleDev=1 に設定されている場合、procWinTitle には、タイトルにスペースを 1 つ空けて、括弧内に独立モジュール名を指定することもできます。
その場合、関数やマクロの検索は、指定されたプロシージャウィンドウと独立モジュール内で行われます。

例えば、あるプロシージャファイルに次のような文が含まれている場合：

```
#pragma IndependentModule=myIM
#include <Axis Utilities>
```

コマンド

```
DisplayProcedure/W=$"Axis Utilities.ipf [myIM]" "HVAxisList"
```

は Axis Utilities.ipf ファイルと独立モジュール myIM に含まれる HVAxisList 関数が定義されているプロシージャウィンドウを開きます。
このコマンドでは、スペースや角括弧がコマンドの解析に支障をきたすため、$"" という構文が使われています。

同様に、SetIgorOption IndependentModuleDev=1 に設定した場合、*functionOrMacroNameStr* には、独立モジュールのプリフィックスに続いて # 文字が含まれることもあります。
前述のコマンドは次のように書き換えることができます。

```
DisplayProcedure/W=$"Axis Utilities.ipf" "myIM#HVAxisList"
```

または、より簡単に

```
DisplayProcedure/W=$"[myIM]" "HVAxisList"
```

例

```
DisplayProcedure "Graph0"
DisplayProcedure/B=Panel0 "MyOwnUserFunction"
DisplayProcedure/W=Procedure                // メインプロシージャウィンドウを表示
DisplayProcedure/W=Procedure/L=5            // 行 5 を表示（6 番目の行）
DisplayProcedure/W=$"Wave Lists.ipf"
```

```
DisplayProcedure "moduleName#myStaticFunctionName"
SetIgorOption IndependentModuleDev=1
DisplayProcedure "WMGP#GizmoBoxAxes#DrawAxis"
```

参照

HideProcedures、DoWindow コマンド
ProcedureText、ProcedureVersion、ModifyProcedure コマンド
MacroList、FunctionList コマンド

ヘルプ Independent Modules

DoAlert [/T=*titleStr*] *alertType*, *promptStr*

DoAlert コマンドでは、アラートダイアログが表示され、ユーザーがボタンをクリックするまで待機します。

パラメーター

alertType = 0 : OK ボタン付きのダイアログ
1 : Yes と No のボタンがあるダイアログ
2 : Yes、No、Cancel ボタンが表示されたダイアログ

promptStr アラートダイアログに表示されるテキスト

フラグ

/T=*titleStr* ダイアログウィンドウのタイトルを、デフォルトのタイトルから変更します。

詳細

DoAlert は特別な変数を設定します

V_flag = 1: Yes がクリックされた
2 : No がクリックされた
3 : Cancel がクリックされた

Igor Pro 10 以降、*promptStr* には HTML 4 のサブセットを含めることができます。
サポートされている HTML 4 サブセットの詳細については、<https://doc.qt.io/qt-6/richtext-html-subset.html> を参照してください。

例

```
Macro example()
    String msg= "<p style=\"color:red;\">This is red text.</p>"
    msg += "<p>Click <a href=\"https://www.wavemetrics.com/\">here to buy Igor</a>.</p>"
    DoAlert/T="Buy Igor!" 0, msg
End
```