CONTENTS

ビジュアルヘルプ – Igor を使う（Using Igor） 5

言語の選択 5
ヘルプの入手 5
ガイドツアー 5
Igor 内のヘルプ 6
Igor ヘルプブラウザー 6
Igor ヘルプファイル 6
ヘルプウィンドウの非表示と削除 6
ヘルプウィンドウからコマンドを実行する 7
ツールチップ 7
Igor ファイルの検索 7
サンプルのエクスペリメント 8
Igor メーリングリスト 8
IgorExchange 8
Igor のアップデート 8
テクニカルサポート 9
コマンドウィンドウ 9
コマンドウィンドウの例 10
コマンドバッファ 12
コマンドウィンドウのタイトル 12
履歴エリア 13
コマンド履歴の制限 13
履歴アーカイブ 13
履歴のカーボンコピー 14
コマンドウィンドウ内の検索 15
コマンドウィンドウの拡大縮小 16
コマンドウィンドウのフォーマット 16
コマンドラインからヘルプを取得 17
ウィンドウ 17
コマンドウィンドウ 17
その他のウィンドウ 17

ターゲットウィンドウ....18
ウィンドウの名前とタイトル....18
許可されるウィンドウ名....19
「ファイルを開く」サブメニュー....19
「ウィンドウ」メニュー....19
新しいウィンドウの作成....20
ウィンドウのアクティブ化....20
ウィンドウの表示と非表示....20
ウィンドウを閉じる....21
削除と非表示....21
ウィンドウの内容を保存....22
ウィンドウダイアログを閉じる....22
ウィンドウを再作成マクロとして保存....23
「ウィンドウ」メニューのマクロサブメニュー....24
再作成されたウィンドウの名前....24
マクロからウィンドウのスタイルを変更....24
ウィンドウコントロールダイアログ....25
ウィンドウブラウザー....25
ウィンドウの配置....26
ウィンドウの位置とサイズの管理....27
初期設定の位置へ移動....27
最大サイズの位置へ移動....27
ウィンドウを画面に戻す....28
すべてのウィンドウを画面に戻す....28
最背面に送る – 最前面に送る....28
テキストウィンドウ....28
コマンドの実行....29
テキストウィンドウのナビゲーション....29
テキストウィンドウの戻る／進む....29
アクティブなウィンドウ内のテキストの検索....30
検索と置換....30
複数のウィンドウ内のテキストの検索....31
テキストの拡大縮小....31
検索バー....32

検索と置換 32
「複数ウィンドウのテキスト検索」ダイアログ 33
複数ウィンドウ内の検索と置換 33
ホームテキストファイル vs 共有テキストファイル 34
外部のテキストエディターの使用 34
プレーンテキストファイルが外部で変更された 35
プレーンテキストファイルが外部および Igor 内で変更された 36
外部での変更のあるファイルを編集する 36
プレーンテキストファイルが見つからない 37
「テキストファイルの欠落または修正（Missing or Modified Text Files）」ダイアログ 37
オブジェクト名 37
標準のオブジェクト名 38
自由なオブジェクト名（リベラル名） 38
名前空間 39
オブジェクト名に関する関数 39
長いオブジェクト名 40
古い Igor バージョンでの長いオブジェクト名 41
長いオブジェクト名でのプログラミング 42
長いオブジェクト名と XOP 42
オブジェクト名の変更 43
オブジェクト名の衝突 43
Igor が許可するオブジェクト名の衝突 43
オブジェクト名の衝突と HDF5 ファイル 44
メモリの管理 45
Igor のエクステンション 45
WaveMetrics の XOP 45
サードパーティの XOP 45
64 ビットエクステンションの有効化 46
32 ビットエクステンションの有効化 46
「その他の設定」ダイアログ 46
プレファレンス 47
プレファレンスのディレクトリ 47
プレファレンスの使い方 47
環境設定を保存 48

ビジュアルヘルプ – Igor を使う（Using Igor）

※本セクション（Igor を使う）は日本語の GUI を使って説明しています。他のビジュアルヘルプは英語 GUI を使って解説しています。

言語の選択

Igor Pro 10 以降では、ユーザーインターフェイス（ダイアログ、メニューなど）およびドキュメント（Igor ヘルプファイル）において、複数言語に対応しています。
現時点（Igor Pro 10 リリース時）では、利用可能な言語は英語と日本語です。
Igor は、利用可能な場合はオペレーティングシステムの言語を使用します。
それ以外の場合は英語が使われます。
翻訳がない場合、英語の文字列とドキュメントが表示されます。
現状、ユーザー定義メニューや Igor プロシージャコードの翻訳に対する組み込みサポートは存在しないため、一部の Igor 機能は翻訳されていません。
一例として、「分析（Analysis）」→「Filter」ダイアログがあります。

言語選択の動作は、「その他の設定（Miscellaneous Settings）」 ダイアログから設定できます。
ここで言語を選択した場合、オペレーティングシステムの言語に関係なく、その言語が使われます。
言語を変更した後は、変更を有効にするために Igor を再起動する必要があることに注意してください。

言語は、/LANG フラグを指定して Igor を起動することでコントロールすることもできます。
詳細はヘルプ Calling Igor From Scripts を参照してください。

ヘルプの入手

Igor Pro について学ぶための情報源はいくつかあります。
最も重要なものは以下のものです：

- ガイドツアー（Guided Tour）
- Igor ヘルプシステム（Igor Help System）
- Examples フォルダー内のサンプルエクスペリメント
- Igor メーリングリスト
- IgorExchange ユーザー同士のサポート Web ページ

ガイドツアー

注記：Igor ガイドツアーは Igor を学ぶ上で不可欠です。

ガイドツアーに費やす時間は、生産性として何倍にも返ってきます。
ガイドツアーを行うには、「ヘルプ（Help）」→「はじめに（Getting Started）」を選択してください。
また、Web の日本語サポートのセクションに日本語によるビジュアルな解説を用意しています。

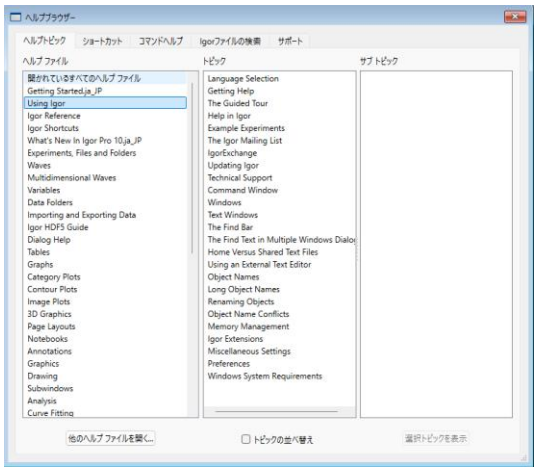
Igor 内のヘルプ

Igor のヘルプシステムは次のコンポーネントで構成されています：

- Igor ヘルプブラウザー
- Igor ヘルプファイル
- ツールチップ

Igor ヘルプブラウザー

Igor ヘルプブラウザーは、最も頻繁に使われる参考資料への迅速なアクセスを提供するとともに、他の種類の情報を検索する時の出発点となるよう設計されています。
Igor ヘルプブラウザーは次の方法で表示できます：



- ヘルプ（Help）→Igor ヘルプブラウザー（Igor Help Browser）を選択する
- F1 キーを押す
- コマンドウィンドウの右下隅にある「？」アイコンをクリックする

Igor ヘルプファイル

Igor のインストーラーは、ヘルプファイルを主に「Igor Pro Folder/Igor Help Files」および「Igor Pro Folder/More Help Files」に配置します。

Igor が起動すると、「Igor Pro Folder/Igor Help Files」および「Igor Pro Folder/More Help Files」に保存されている Igor ヘルプファイルを開くことで、自動的にヘルプウィンドウを作成します。
ヘルプブラウザーの「ヘルプファイル（Help Files）」タブを使うか、ヘルプ（Help）メニューの「ヘルプウィンドウ（Help Windows）」サブメニューから選択することで、ヘルプウィンドウを表示できます。

開いているヘルプファイルのトピック一覧を表示するには、Igor ヘルプブラウザーの「ヘルプトピック（Help Topics）」タブを使ってください。

開いているかどうかにかかわらず、インストーラーによってインストールされたすべてのヘルプファイルを検索するには、Igor ヘルプブラウザーの「Igor ファイルを検索（Search Igor Files）」タブを使ってください。

ヘルプウィンドウの非表示と削除

Igor ヘルプウィンドウの「閉じる」ボタンをクリックすると、Igor はそのウィンドウを非表示にします。

通常ヘルプファイルを強制終了させる必要はありませんが、どうしても終了させたい場合は、「閉じる」ボタンをクリックする時に Alt キーを押したままにする必要があります。

ヘルプウィンドウからコマンドを実行する

ヘルプウィンドウには、例として Igor コマンドが多数示されています。
ヘルプウィンドウからコマンドまたはコマンドのセクションを実行するには、コマンドテキストを選択し、Control-Enter または Control-Return を押します。
選択したテキストがコマンドラインに転送され、実行を開始します。

ツールチップ

Igor は、様々なアイコン、ダイアログ項目、その他の視覚的機能に対してツールチップを提供します。
これらのヒントは、「その他の設定（Miscellaneous Settings）」ダイアログの「ツールチップを表示（Show Tooltips）」チェックボックスを使ってオンまたはオフにできます。

グラフ内のトレースやテーブルの列に関する情報を取得するには、ツールチップも使用できます。
「グラフ（Graph）」→「トレース情報タグの表示（Show Trace Info Tags）」と「テーブル（Table）」→「列情報タグの表示（Show Column Info Tags）」を使って、これらのヒントの表示/非表示を切り替えます。

Igor ファイルの検索

Igor ヘルプブラウザーの「Igor ファイルを検索（Search Igor Files）」タブを使って、Igor ヘルプファイルやプロシージャファイルを検索できます。

検索式は１つ以上（最大８つまで）の用語で構成できます。
用語は「and」という単語で区切ります。

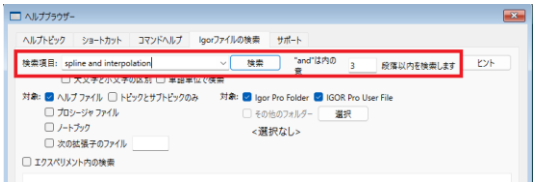
以下に例を示します：

```
interpolation                          // 用語１つ
spline interpolation                   // 用語１つ
spline and interpolation               // 用語２つ
spline and interpolation and smoothing // 用語３つ
```

２番目の例は「spline interpolation」という完全一致のフレーズを検索します。
一方、３番目の例は「spline」と「interpolation」という単語を含むセクションを検索しますが、必ずしも連続している必要はありません。

検索でサポートされているキーワードは「and」のみです。
検索内の引用符（" など）は特別な意味を持たないため、使わないでください。

「段落以内を検索します」の欄に 10 と入力した場合、これは 2 番目の用語が最初の用語から 10 段落以内に現れる必要があることを意味し、それによってヒットが成立します。
値が 0 の場合、用語は同じ段落内に現れなければなりません。



プロシージャファイルなどのプレーンテキストファイルにおいて、段落とはテキスト 1 行を指します。
空白行も 1 段落として扱われます。

複数の検索語を入力する時、検索を高速化するには、最も出現頻度の低い語を最初に入力してください。
例えば、Igor のヘルプファイルで「hidden and axis and graph」を検索する方が、「graph and axis and hidden」を検索するよりも速くなります。
というのも「hidden」は「graph」よりも使用頻度が低いためです。

サンプルのエクスペリメント

Igor Pro には、Igor の機能やテクニックを実演する豊富なサンプルが同梱されています。
「ファイル（File）」→「サンプルエクスペリメント（Example Experiments）」のサブメニューからアクセスできます。

Igor メーリングリスト

Igor メーリングリストは、Igor Pro ユーザー同士が互いに助け合い、解決策やアイデアを共有するためのインターネット上のディスカッションリストです。
WaveMetrics はまた、このリストを利用して最新の Igor 開発に関する情報を投稿しています。
メーリングリストの購読方法やその他の詳細については、次のウェブページをご覧ください：

http://www.wavemetrics.com/users/mailinglist.htm

IgorExchange

IgorExchange は、WaveMetrics が後援するユーザー間サポートおよびコラボレーションウェブサイトですが、Igor ユーザーによって運営され、Igor ユーザーのために存在します。
IgorExchange に関する情報は、次のウェブページをご覧ください：

http://www.IgorExchange.com

Igor のアップデート

WaveMetrics は定期的に Igor の無料アップデートをリリースしています。
アップデートではバグ修正が行われ、場合によっては新機能が追加されます。

Igor Pro は起動時に、当社のウェブサイトのいずれかに接続して利用可能な更新を確認します。
この更新チェックは、Igor の起動時のパフォーマンスへの影響を最小限に抑えるように実装されています。

更新が見つかった場合、Igor はダイアログを表示し、更新のダウンロード、現在の更新に関する通知のスキップ、または後で更新についてリマインドするかの選択が可能です。
また、アップデートのリリースノートを表示することもできます。

自動更新チェックは無効にできます。
これを行うには、「その他の設定（Miscellaneous Settings）」ダイアログの「更新（Updates）」セクションを使います。
ベータ版のチェックを有効にすることもできます。

Igor Pro の更新は、「ヘルプ（Help）」→「Igor Pro の更新（Updates for Igor Pro）」を選択することで、いつでも手動で確認できます。
これは、自動更新チェックが有効かどうかにかかわらず機能します。

インターネット接続にプロキシサーバーやその他の特殊な設定が必要な場合、Igor の更新チェック機能が正常に動作しない可能性があります。
そのような場合は、http://www.wavemetrics.net にアクセスして更新プログラムが利用可能かどうかを確認できます。

テクニカルサポート

WaveMetrics はメールによるテクニカルサポートを提供しています。

WaveMetics サポートにメールを送信するには、まず「ヘルプ（Help）」→「サポートに連絡（Contact Support）」を選択してください。
これにより、サポート提供に必要な Igor のシリアル番号やバージョンなど、Igor のインストールに関する情報が記載されたメールが生成されます。

ほとんどの場合、問題を解決するには、その問題を再現する必要があります。
問題を示す簡略化された例を提供していただけると助かります。

アップグレードやその他の技術以外の情報については、お問い合わせを下記までお送りください：

sales@wavemetrics.com

弊社のウェブサイトは https://www.wavemetrics.com/ で見ることができます。
また、「ヘルプ（Help）」→「WaveMetrics ホームページ（WaveMetrics Home Page）」を選択することもできます。

弊社のウェブサイトのサンプルギャラリーセクションには、数々のクールなグラフが掲載されています。
新しいクールなグラフを常に歓迎しております。
共有したいクールなグラフをお持ちの方は、sales@wavemetrics.com までご連絡ください。

コマンドウィンドウ

メニューやダイアログ、またはコマンドウィンドウから実行するコマンドを使って、Igor をコントロールできます。
一部の操作（例えばウェーブフォーム代入など）は、コマンドを入力する方がはるかに簡単に行えます。
コマンドはテーマのバリエーションを試すのにも便利です。
コマンドを修正して再実行するのが非常に早いです。
Igor を日常的に使っている場合、頻繁に行う操作に対してコマンドをますます多用するようになるかもしれません。

コマンドウィンドウでコマンドを実行するほか、ノートブック、プロシージャ、またはヘルプウィンドウでもコマンドを実行できます。

これらの方法はコマンドウィンドウほど頻繁には使われません。
詳細についてはヘルプ Notebooks as Worksheets を参照してください。

このセクションでは、コマンドウィンドウと一般的なテクニックおよびショートカットについて説明します。
コマンドの使用方法と構文の詳細については、ヘルプ Working with Commands を参照してください。

コマンドウィンドウは、コマンドラインと履歴エリアで構成されています。
コマンドラインでコマンドを入力し、Enter キーを押すと、コマンドが実行されます。
その後、それらは履歴エリアに保存され、確認できるようになります。
コマンドがテキスト出力を生成した場合、その出力も履歴エリアに保存されます。
履歴内のコマンド行には先頭に点記号が付加されるため、コマンド行と出力行を容易に区別できます。

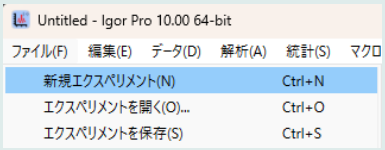
コマンドウィンドウには、履歴エリアのスクロールバーの直下にヘルプボタンがあります。
このボタンをクリックすると、ヘルプブラウザウィンドウが表示されます.

コマンドライン上のコマンドの最大長は 2500 バイトに制限されています。
コマンドラインで実行されるコマンドは 1 行に収まる必要があります。

コマンドウィンドウの例

コマンドの力を示す簡単な例と、コマンド操作を容易にするショートカットの一部を紹介します。

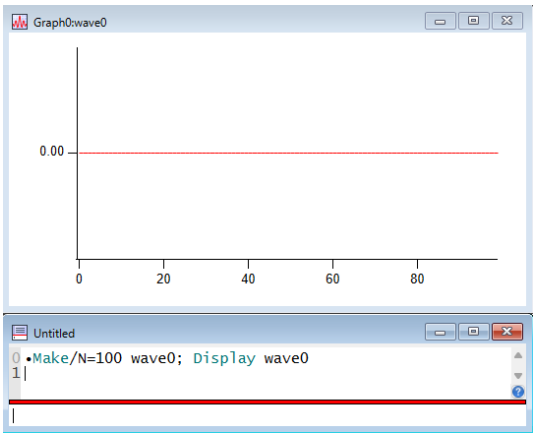
1. **「ファイル（File）」→「新規エクスペリメント（New Experiment）」を選択します。**



2. **コマンドラインに次のコマンドを入力し、Enter キーを押して実行してください。**

```
Make/N=100 wave0; Display wave0
```

これはグラフを表示します。



3. **Ctrl+J キーを押します。**

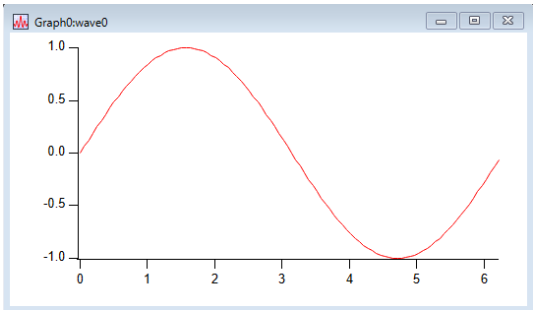
これでコマンドウィンドウがアクティブになります。

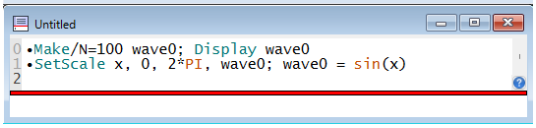
4. 次を実行します。

```
SetScale x, 0, 2*PI, wave0; wave0 = sin(x)
```

このグラフは、0 から 2π までの x の正弦値を示しています。

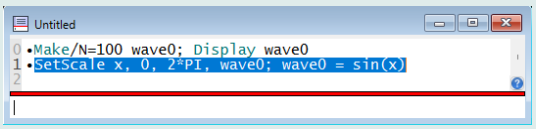
次に、コマンドを素早く取得、変更、再実行する方法を見ていきます。





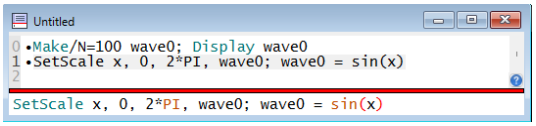
5. 上矢印キーを押します。

これで、先ほど実行したコマンドが選択されます。



6. Enter キーを押します。

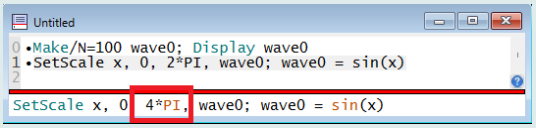
これにより、選択内容がコマンドラインに戻されます。



7. 「2」を「4」に変更します。

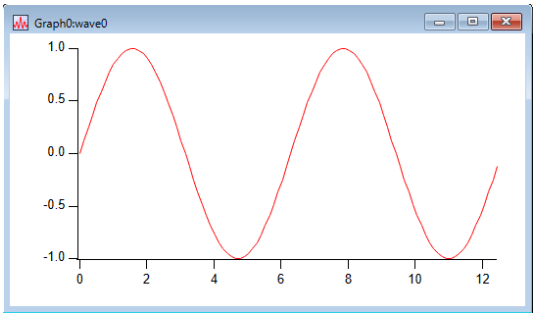
コマンドラインには現在、以下が含まれているはずです：

```
SetScale x, 0, 4*PI, wave0; wave0 = sin(x)
```



8. 変更したコマンドを実行するために、Enter キーを押します。

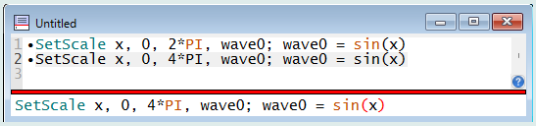
これは 0 から 4πまでの x の正弦を表示します。



9. Alt キーを押しながら、履歴の最後のコマンドをクリックしてください。

これは履歴からコマンドラインへコマンドを転送する別の方法です。
コマンドラインには現在以下が含まれているはずです：

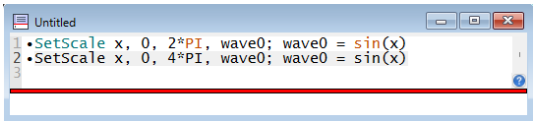
```
SetScale x, 0, 4*PI, wave0; wave0 = sin(x)
```



10. Ctrl+K を押します。

これによりコマンドラインの内容が「消去（Kill）」されます。

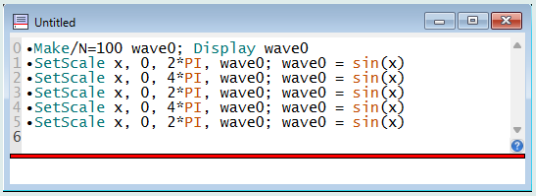
では、以前に実行したコマンドを素早く再実行する方法を見てみます。



11. Ctrl+Alt キーを押しながら、履歴の最後から 2 番目のコマンドをクリックします。

これはクリックされたコマンド（2*PI コマンド）を再実行します。

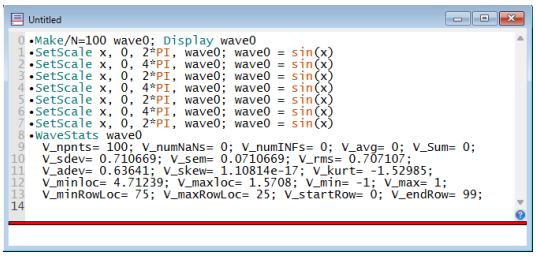
最後から 2 番目のコマンドをクリックする手順を数回繰り返してください。
これは 2PI コマンドと 4PI コマンドを交互に実行します。



12. 次を実行します：

```
WaveStats wave0
```

`WaveStats` コマンドの結果は履歴に記録されており、そこで確認できます。
履歴から数値をコピーして、ノートブックや注釈に貼り付けることもできます。



コマンドウィンドウのショートカットはすべてヘルプ Command Window Shortcuts にまとめられています。

コマンドバッファ

通常、コマンドバッファは空か、あるいはテキストが 1 行だけ含まれています。
ただし、任意のウィンドウから複数行のテキストをコピーし、コマンドバッファに貼り付けることができ、Shift キーを押しながら Enter キーを押すことで改行を挿入し、複数行を入力できます。
赤い区切り線を上にドラッグすると表示する行数を増やせます。下にドラッグすると行数を減らせます。

コマンドバッファの内容を消去するには、「編集（Edit）」メニューの「コマンドバッファを消去（Clear Command Buffer）」を選択するか、Ctrl+K を押します。

Igor ダイアログからコマンドを呼び出すと、ダイアログはコマンドをコマンドバッファに配置し、それを実行します。
コマンドは、手動で入力した場合と同様に履歴に転送されます。

コマンドの実行中にエラーが発生した場合、Igor はそれをコマンドバッファに残すため、編集して再実行できます。
コマンドを修正しない場合は、Ctrl+K を押してコマンドバッファから削除してください。

コマンドバッファには通常、何も含まれていないか、 1 つのコマンドしか含まれていないため、通常、それを 1 行として考え、「コマンドライン（英語において複数形ではない）」という用語を使います。

コマンドウィンドウのタイトル

コマンドウィンドウのタイトルは、現在読み込まれているエクスペリメントの名前です。
Igor を初めて起動したとき、または「ファイル（File）」メニューから「新規エクスペリメント（New Experiment）」を選択した場合、エクスペリメントのタイトル、そしてコマンドウィンドウのタイトルは「Untitled」となります。

エクスペリメントをファイルに保存すると、Igor はエクスペリメントの名前をファイル名から拡張子を除いたものに設定します。

ファイル名が「An Experiment.pxp」の場合、エクスペリメント名は「An Experiment」です。
Igor はコマンドウィンドウのタイトルとして「An Experiment」を表示します。

プロシージャ内で使う場合、IgorInfo(1) 関数は現在のエクスペリメントの名前を返します。

履歴エリア

履歴エリアはコマンドと結果の保存場所です。

履歴エリアのテキストは編集できませんが、クリップボードまたはコマンドラインにコピーできます。
テキストをクリップボードにコピーする操作は通常通り行います。
履歴からコマンドをコマンドバッファにコピーするには、履歴で該当コマンドを選択し、Enter キーを押します。
別の方法として、Alt キーを押しながら履歴エリアをクリックする方法があります。

履歴からコマンドをコマンドラインに簡単にコピーできるように、履歴エリアの行をクリックするとその行全体が選択されます。
クリックしてドラッグすることで、行の一部だけを選択することもできます。

上矢印キーと下矢印キーは、履歴内の選択範囲を１行ずつ上下に移動し、１行単位で選択します。
通常、コマンドをコマンドラインにコピーするために履歴から行を選択したい場合、上矢印キーと下矢印キーはコマンド行以外の行をスキップします。
左矢印キーと右矢印キーはコマンドライン上の挿入位置を移動します。

エクスペリメントを保存すると、履歴エリアの内容が保存されます。
次回、エクスペリメントを読み込む時、履歴はそのまま保持されています。
一部の人々は、Igor が履歴を再実行することでエクスペリメントを再現しているという印象を持っています。
しかし、これは正しくありません。
詳細はヘルプ How Experiments Are Loaded を参照してください。

コマンド履歴の制限

履歴エリアの内容は、時間の経過とともに非常に大きくなる可能性があります。
「その他（Misc）」メニューからアクセス可能な「その他の設定（Miscellaneous Settings）」ダイアログの「コマンドウィンドウ（Command Window）」セクションにある「コマンド履歴テキストの最大行数（Limit Command History Text）」機能を使うと、履歴に保持されるテキスト行数を制限できます。

コマンド履歴を制限した場合、エクスペリメントを保存する時に履歴行数を確認します。
制限を超えた場合、最も古い行から削除されます。

履歴アーカイブ

「コマンド履歴テキストの最大行数（Limit Command History Text）」機能で履歴の行が削除された場合、履歴アーカイブ機能により、削除された行をエクスペリメントのホームフォルダー内のテキストファイルに書き込むように設定できます。

特定のエクスペリメントに対して履歴アーカイブ機能を有効にするには、そのエクスペリメントのホームフォルダー内にプレーンテキストファイルを作成してください。
テキストファイルは次の名前でなければなりません：

```
<Experiment Name> History Archive UTF-8.txt
```

ここで <Experiment Name> は現在のエクスペリメントの名前です。
エクスペリメントを保存すると、Igor は削除された履歴の行をすべて履歴アーカイブファイルに書き込みます。

Igor Pro 7.0 以前のバージョンでは、履歴アーカイブファイルはシステムテキストエンコーディングで記述され、次の名前が付けられていました：

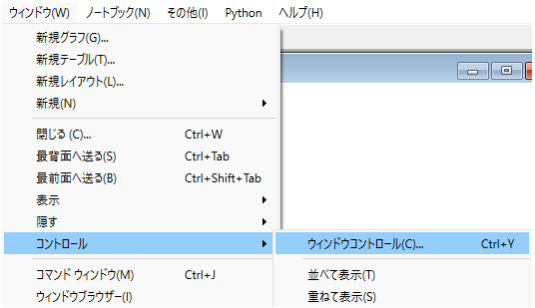
```
<Experiment Name> History Archive.txt
```

Igor Pro 6 で履歴アーカイブを使う場合、名前の中に「UTF-8」を含む新しい履歴アーカイブファイルを作成する必要があります。

履歴アーカイブファイルが Igor を含む他のプログラムで開かれている場合、履歴アーカイブ機能が正常に動作せず、履歴行が書き込まれない可能性があります。

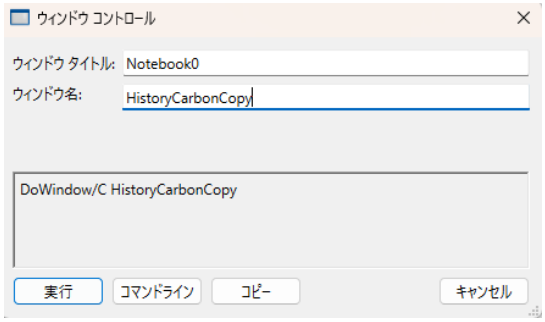
履歴のカーボンコピー

ノートブックを「履歴エリアのカーボンコピー」として指定するには、プレーンテキストまたは書式付きノートブックを作成し、「ウィンドウ（Windows）」→「コントロール（Control）」→「ウィンドウコントロール（Window Control）」経由でそのウィンドウ名を「HistoryCarbonCopy」に設定します。
HistoryCarbonCopy ノートブックが存在する場合、履歴テキストをノートブック内と履歴の両方に挿入します。



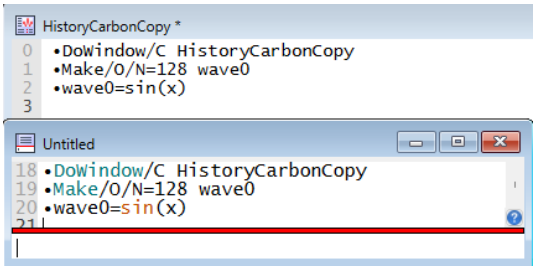
ただし、HistoryCarbonCopy ノートブックからコマンドが開始された場合（ヘルプ Notebooks as Worksheets を参照）、Igor はそのコマンドの実行中に、そのノートブックへの履歴テキストの送信を一時停止します。

ノートブックの名前を HistoryCarbonCopy 以外の名前に変更すると、そのノートブックへの履歴テキストの送信を停止します。



後で名前を HistoryCarbonCopy に戻すと、履歴テキストの送信を再開します。

「その他の設定」ダイアログで設定できる履歴の切り詰め機能は、HistoryCarbonCopy ノートブックには適用されません。
その場合、自分で切り詰める必要があります。
ノートブックは 1600 万段落に制限されています。



フォーマット済みノートブックを履歴のカーボンコピーとして使う場合、Command と Result という名前のノートブックルーラーを作ることで、コマンドと結果の書式設定をコントロールできます。
履歴カーボンコピーノートブックにテキストが送信されると、常に Command ルーラーがコマンドに適用されます。
現在のルーラーが通常、コマンド、または結果の場合、結果に Result ルーラーを適用します。
履歴のカーボンコピーにテキストを送信する時に、それらを使いたい場合は、Command ルーラーと Result ルーラーを作成する必要があります。

次の関数は、Igor が自動的に使う Command ルーラーと Result ルーラーに加え、カスタムエラーメッセージ用に使うエラールーラーを備えた、フォーマット済みの履歴カーボンコピーノートブックを作成します：

```
Function CreateHistoryCarbonCopy()
	NewNotebook /F=1 /N=HistoryCarbonCopy /W=(50,50,715,590)

	Notebook HistoryCarbonCopy backRGB=(0,0,0)		// 背景を黒に設定

	Notebook HistoryCarbonCopy showRuler=0

	// コマンドをコントロールするルーラーを定義
	// 履歴のカーボンコピーに送信されたコマンドに対してこれを自動的に適用
	Notebook HistoryCarbonCopy newRuler=Command,
rulerDefaults={"Geneva",10,0,(65535,65535,0)}

	// 結果をコントロールするルーラーを定義
	// 履歴のカーボンコピーに送信された結果に対してこれを自動的に適用
	Notebook HistoryCarbonCopy newRuler=Result,
rulerDefaults={"Geneva",10,0,(0,65535,0)}

	// ユーザー生成のエラーメッセージをコントロールするルーラーを定義
	// このルーラーを、印刷コマンド経由で履歴カーボンコピーに送信するエラーメッセージに適用
	Notebook HistoryCarbonCopy newRuler=Error,
rulerDefaults={"Geneva",10,0,(65535,0,0)}
End
```

現在のルーラーが Normal、Command、Result ではない場合、それは Print コマンドを使って履歴に送信される特別なメッセージに使いたいカスタムルーラーであると見なされます。
この場合、Igor は Result ルーラーを適用せず、代わりにカスタムルーラーを有効なままにします。

次の関数は、履歴カーボンコピーノートブック内のカスタムエラールーラーを使って、エラーメッセージを履歴に送信します：

```
Function PrintErrorMessage(message)
	String message

	Notebook HistoryCarbonCopy, ruler=Error
	Print message

	// ルーラーを Result に戻すことで、Command ルーラーと Result ルーラーの自動的な使用が、
	// 以降のコマンドで有効になる
	Notebook HistoryCarbonCopy, ruler=Result
End
```

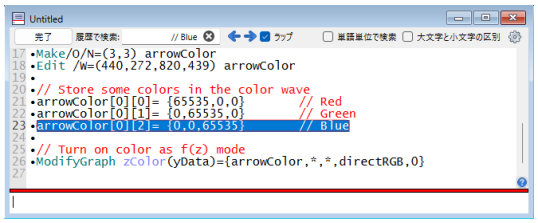
XOP プログラマーは、XOPNotice3 XOPSupport ルーチンを使って、履歴カーボンコピーノートブックに送信されるテキストの色をコントロールできます。

コマンドウィンドウ内の検索

コマンドラインまたは履歴を検索するには、「編集（Edit）」メニューから「検索（Find）」を選択するか、「編集」メニューに表示されているキーボードショートカットを使います。
Ctrl+F を押すと検索バーが表示され、カーソルの下にあるテキストが自動的に入力されます。
Ctrl+R を押すと検索と置換の両方のバーが表示され、新しいセッションでは自動的に検索バーにカーソルの下の現在のテキスト、またはセッション内の前回の検索時のテキストが入力されます。

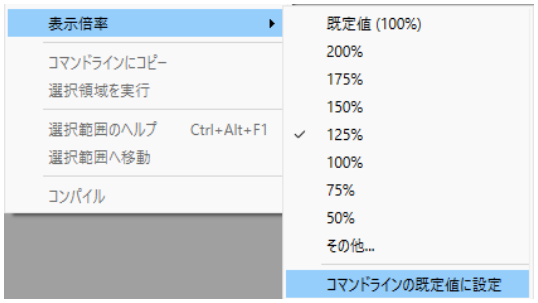
コマンドラインの検索は、主に以前に実行したコマンドを修正してから再実行するために使われます。
例えば、特定のウェーブの名前を別のウェーブの名前に置き換えたい場合などです。

履歴エリアがキーボードフォーカスを持っている場合（通常は履歴内のアクティブな選択で示されます）、検索バーは履歴エリアを検索します。
コマンドラインがキーボードフォーカスを持っている場合、検索バーはコマンドラインを検索します。
検索対象の領域を確実に指定するには、検索を開始する前にその領域をクリックしてください。



コマンドウィンドウの拡大縮小

コマンドラインと履歴エリアのテキストサイズを拡大縮小できます。

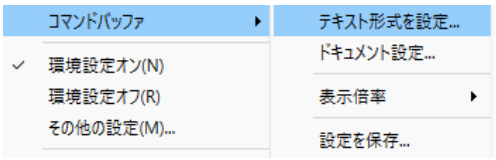


コマンドラインのテキストを拡大縮小するには、コマンドライン上で右クリックし、「表示倍率（Magnification）」サブメニューから項目を選択します。
新しい表示倍率を新規エクスペリメントのデフォルト倍率にするには、再度右クリックし、同じサブメニューから「コマンドラインの既定値に設定（Set as Default for Command Line）」を選択します。

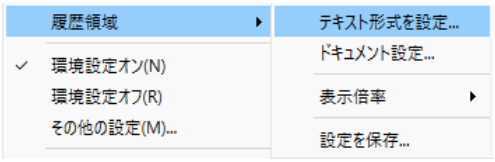
履歴エリアのテキストを拡大するには、履歴エリアを右クリックし、「表示倍率」サブメニューから項目を選択します。
新しい倍率を新規エクスペリメントのデフォルトの倍率にするには、再度右クリックし、同じサブメニューから「履歴領域の既定値デフォルトに設定（Set as Default for History Area）」を選択します。

コマンドウィンドウのフォーマット

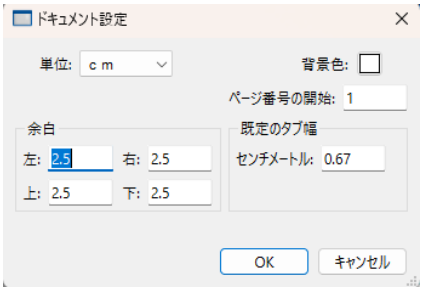
コマンドラインで使われるテキスト形式を変更できます。
例えば、別の色を使いたい場合などです。
これを行うには、コマンドラインをクリックし、「その他（Misc）」メニューの「コマンドバッファ（Command Buffer）」サブメニューから「テキスト形式を設定（Set Text Format）」を選択します。



履歴エリアのテキスト形式を設定するには、履歴エリアをクリックし、「その他」メニューの「履歴領域（History Area）」サブメニューから「テキスト形式を設定」を選択します。
この操作を行うには、履歴エリアにテキストが入力されている必要があります。

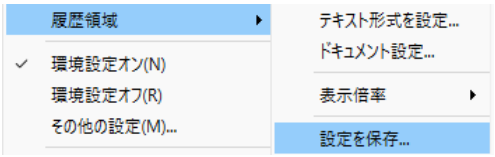


コマンドバッファまたは履歴エリアのサブメニューから「ドキュメント設定（Document Settings）」を選択することで、背景色などの他のプロパティを設定できます。
「ドキュメント設定」ダイアログでは、履歴を印刷する時に使うヘッダーとフッターも設定できます。



テキスト形式や文書設定を変更すると、現在のエクスペリメントのみが変更されます。

新しいエクスペリメントに対しては、新しいフォーマットと設定をプレファレンスとして保存することを推奨します。
これを行うには、コマンドバッファと履歴エリアのサブメニューから「設定を保存（Capture Prefs）」を選択します。

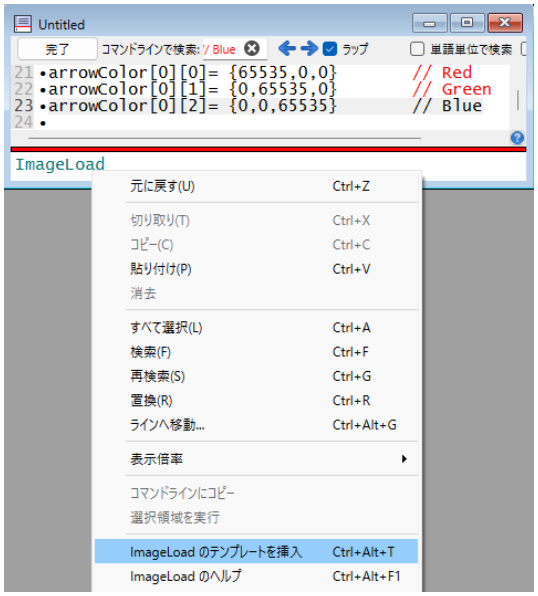


コマンドラインからヘルプを取得

コマンドラインを使う時、コマンドの作成に支援が必要になる場合があります。
テンプレートの挿入、ヘルプの表示、ユーザー関数の定義の検索を可能にするショートカットがあります。
コマンドウィンドウはコマンド補完もサポートしています。

テンプレートを挿入するには、コマンドまたは関数の名前を入力し、右クリックして「テンプレートを挿入（Insert Template）」を選択します。

ヘルプを表示したり、ユーザー関数の定義を表示するには、コマンドまたは関数の名前を入力し、右クリックして、そのコマンド名のヘルプを選択します。



ウィンドウ

このセクションでは、Igor のウィンドウ全般、「ウィンドウ（Windows）」メニュー、およびウィンドウ再作成マクロについて説明します。

コマンドウィンドウ

Igor を起動すると、画面の下部にコマンドウィンドウが表示されます。

マウスでダイアログを「ポイント＆クリック」操作すると、コマンドがコマンドウィンドウのコマンドラインに自動的に入力され、実行されます。
必要に応じてコマンドを直接入力し、Enter キーを押すこともできます。
Igor は実行されたコマンドの履歴を履歴エリアに保存します。

コマンドウィンドウについての詳細は、以降の「コマンドウィンドウ」のセクション、またはヘルプ Working with Commands を参照してください。

コマンドを扱う時に非常に便利なショートカットがいくつかあります。
詳細はヘルプ Command Window Shortcuts を参照してください。

その他のウィンドウ

また、初期状態では非表示になっている追加のウィンドウがいくつかあります：

- メインプロシージャウィンドウ
- Igor ヘルプブラウザー
- 「Igor Pro Folder/Igor Help Files」および「Igor Pro User Files/Igor Help Files」内のヘルプウィンドウ

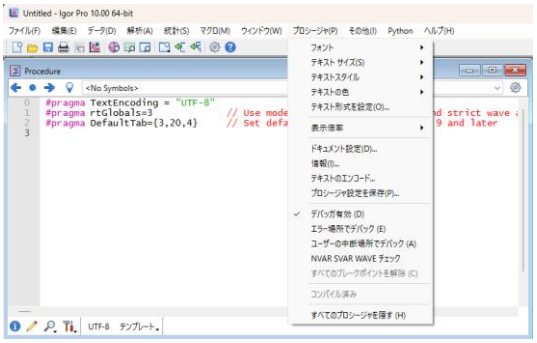
グラフ、テーブル、ページレイアウト、ノートブック、コントロールパネル、Gizmo（3D プロット）、補助 Procedure ウィンドウ、およびヘルプウィンドウを追加で作成できます。

ターゲットウィンドウ

Igor のコマンドとメニューはターゲットウィンドウ上で動作します。
ターゲットウィンドウとは、最前面のグラフ、テーブル、ページレイアウト、ノートブック、コントロールパネル、Gizmo、または XOP ターゲットウィンドウを指します。
「ターゲット」という用語は、これらのウィンドウが ModifyGraph や ModifyTable などのコマンドライン操作の対象となり得るということに基づいています。
コマンドウィンドウ、プロシージャウィンドウ、ヘルプウィンドウ、およびダイアログは、コマンドライン操作の対象となり得ないため、ターゲットウィンドウではありません。

メニューバーは、最前面のウィンドウとターゲットウィンドウに応じて変化します。
例えば、グラフがターゲットウィンドウの場合、メニューバーには「グラフ（Graph）」メニューが表示されます。

ただし、コマンドラインには任意のコマンドを入力できます。
ターゲットウィンドウに適用されないコマンドも含まれます。
Igor は正しいタイプの最前面ウィンドウにコマンドを適用します。



最前面のウィンドウがターゲットウィンドウでない場合でも、メニューバーが変更されることがあります。
例えば、プロシージャウィンドウをアクティブにすると、メニューバーに「プロシージャ（Procedure）」メニューが表示されます。

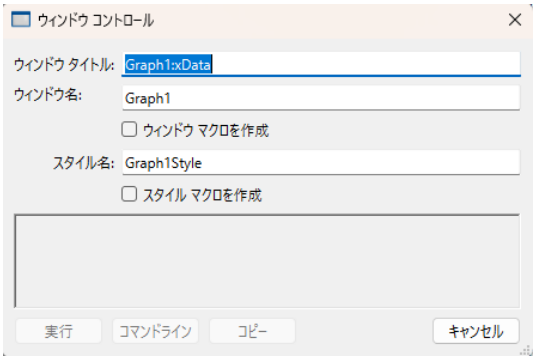
ウィンドウの名前とタイトル

各グラフ、テーブル、ページレイアウト、コントロールパネル、ノートブック、および Gizmo には、タイトルと名前が付けられています。

タイトルは、ウィンドウ枠の上部と「ウィンドウ（Windows）」メニューに表示されるものです。
その目的は、視覚的にウィンドウを識別しやすくすることであり、通常はその内容や目的を説明しています。

ウィンドウ名はタイトルとは異なります。
この名前の目的は、DoWindow や AppendToGraph コマンドなどのコマンドからウィンドウを参照できるようにすることです。

これらのウィンドウを初めて作成すると、Graph0、Table0、Layout0、Panel0 といった名前と、その名前とウィンドウの内容に基づいたタイトルを自動的に付与します。
ウィンドウのタイトルと名前は、よりわかりやすいものに変更できます。
これには「ウィンドウコントロール（Window Control）」ダイアログ（ウィンドウ→コントロールサブメニュー）を使います。



ウィンドウコントロールダイアログは、最前面のウィンドウ名を確認する手段としても有用です。
なぜなら、そのウィンドウにはウィンドウタイトルのみが表示されるためです。

コマンドウィンドウ、プロシージャウィンドウ、ヘルプウィンドウにはタイトルのみが表示されます。
タイトルは、それらが格納されているファイルの名前です。
これらのウィンドウには名前がありません。
コマンドライン操作の影響を受けないためです。

許可されるウィンドウ名

ウィンドウ名はコマンドに使われるため、Igor オブジェクトの命名に関する標準ルールに従う必要があります：

- 名前は文字で始めなければなりません。
- 英数字またはアンダースコア文字を使うことができます。
- 標準的なオブジェクト名には、スペースを含む他の文字は一切使用できません。
- 255 バイトを超えることはできません。
- 名前は他のオブジェクト名と衝突してはいけません（衝突する場合は Igor が通知します）。

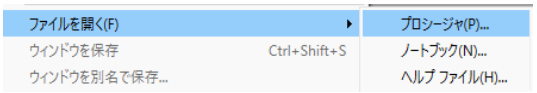
Igor Pro 8.0 以前のバージョンでは、ウィンドウ名は 31 バイトに制限されていました。
長いウィンドウ名を使う場合、エクスペリメントには Igor Pro 8.0 以降が必要です。

詳細については、「オブジェクト名」のセクションを参照してください。

「ファイルを開く」サブメニュー

「ファイル（File）」メニューには、既存のファイルをノートブック、Igor ヘルプウィンドウ、またはプロシージャウィンドウとして開くための「ファイルを開く（Open File）」サブメニューがあります。

サブメニューから項目を選択すると、ファイルを選択するための「ファイルを選択」ダイアログが表示されます。

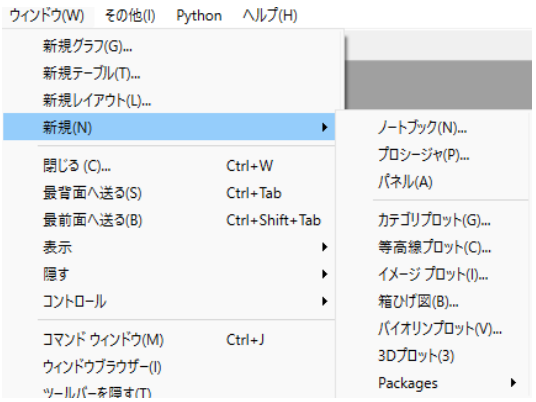


「ウィンドウ」メニュー

新しいウィンドウの作成、表示、配置、および閉じる（非表示または「強制終了」）には、「ウィンドウ（Windows）」メニューを使用できます。

また、「ウィンドウ再作成マクロ」を実行して終了したウィンドウを再作成したり、「スタイルマクロ」を実行して既存のウィンドウの外観を変更したりすることもできます。

新しいウィンドウの作成



「ウィンドウ」メニューおよび「ウィンドウ」→「新規（New）」サブメニューの各種項目を使って、新しいウィンドウを作成できます。
これらの項目のほとんどはダイアログを呼び出し、Igor がウィンドウを作成するために実行するコマンドを生成します。

コマンドラインに直接これらのコマンドを入力してウィンドウを作成することもできます。
例えば、

```
Display yData vs xData
```

これにより、Y 軸に yData という名前のウェーブ、X 軸に xData をプロットしたグラフが生成されます。

「ウィンドウ」メニューからウィンドウ再作成マクロの名前を選択することで、新しいウィンドウを作成できます。「ウィンドウのマクロサブメニュー」のセクションを参照してください。

「ファイル」→「ファイルを開く」サブメニューを使ってウィンドウを作成することもできます。

ウィンドウのアクティブ化

ウィンドウをアクティブにするには、ウィンドウをクリックするか、「ウィンドウ」メニューまたはそのサブメニューから項目を選択してください。

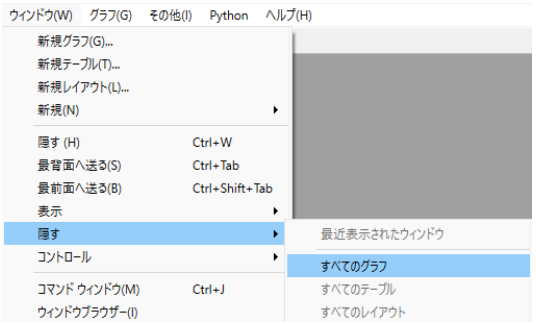
「最近使用したウィンドウ（Recent Windows）」サブメニューには、最近アクティブ化されたウィンドウが表示されます。
この情報はエクスペリメントをディスクに保存する時に保存され、後でエクスペリメントを再開する時に復元されます。

デフォルトでは、ウィンドウのタイトルのみが「ウィンドウ」メニューに表示されます。
「その他の設定（Miscellaneous Settings）」ダイアログの「その他（Miscellaneous）」セクションにある「ウィンドウメニューの表示（Windows Menu Shows）」ポップアップメニューを使って、対象ウィンドウにタイトルまたは名前を表示するかを選択できます。

ウィンドウの表示と非表示

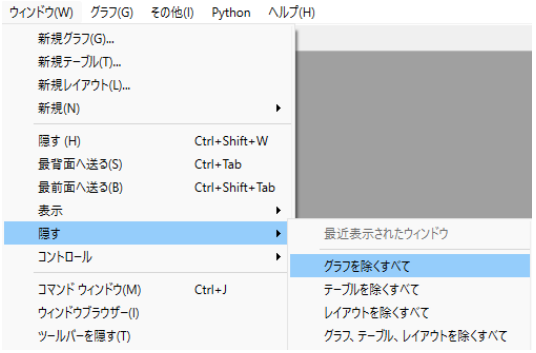
あらゆる種類のウィンドウは非表示にできます。

ウィンドウを非表示にするには、Shift キーを押しながら「ウィンドウ（Windows）」→「隠す（Hide）」を選択するか、キーボードショートカットの Ctrl+Shift+W を使います。



複数のウィンドウを一度に非表示にするには、「ウィンドウ」→「隠す」サブメニューを使います。
例えば、すべてのグラフを非表示にするには、「ウィンドウ」→「隠す」→「すべてのグラフ（All Graphs）」を選択します。

ウィンドウメニューをクリックする時に Shift キーを押すと、メニュー項目の意味が変わります。
例えば、「隠す」→「すべてのグラフ」が「隠す」→「グラフを除くすべて」に変わります。



コマンドウィンドウは、いかなる種類の非表示処理にも含まれません。
非表示にする場合は手動で行う必要があります。

同様に、「ウィンドウ」→「表示」サブメニューを使って複数のウィンドウを同時に表示できます。

例えば、すべてのグラフを表示するには、「ウィンドウ」→「表示」→「すべてのグラフ」を選択します。

「～を除くすべて（Show All Except）」メニュー項目は、プロシージャウィンドウとヘルプファイルを表示しません。
これらは非常に多いため、逆効果となるからです。

「ウィンドウ」→「表示」→「最近隠されたウィンドウ」は、一括の非表示操作（例：「隠す」→「すべてのグラフ」）によって最近非表示にされたウィンドウ、または手動で最近非表示にしたウィンドウ（閉じるボタンまたは Ctrl+Shift+W を使って 1 つずつ非表示にしたもの）を表示します。
手動で非表示にしたウィンドウの場合、「最近隠された」とは過去 30 秒以内を意味します。

XOP ウィンドウは、XOP プログラマーがこれらの機能を特にサポートしている場合にのみ、「すべての XOP ウィンドウを非表示」および「すべての XOP ウィンドウを表示」が追加されます。

ウィンドウを閉じる

ウィンドウを閉じるには、メニューバーの「ウィンドウ」→「閉じる（Close）」を選択するか、ウィンドウの閉じるボタンをクリックします。
最前面のウィンドウの種類に応じて、この操作はウィンドウを終了させるか非表示にします。場合によっては、確認を求めるダイアログが表示されることがあります。

削除と非表示

ウィンドウを「削除する（Kill）」とは、そのウィンドウをエクスペリメントから除外することを意味します。
ウィンドウが使っていたメモリは解放され、他の目的に利用可能になります。
ウィンドウのタイトルは「ウィンドウ」メニューから削除されます。
ディスク上のファイルを表すウィンドウを終了しても、ファイルは削除されません。
KillWindow コマンドでウィンドウを終了させることもできます。

ウィンドウを「非表示／隠す（Hide）」にするとは、単にウィンドウが見えなくなるだけであり、エクスペリメントの一部であり、同じ量のメモリを使っていることを意味します。
「ウィンドウ」メニューからそのタイトルを選択することで、再び表示させることができます。

コマンドウィンドウと組み込みプロシージャウィンドウは非表示にできますが、終了させることはできません。
その他の組み込みウィンドウはすべて非表示にしたり削除したりできます。

プロシージャからウィンドウを作成する場合、ウィンドウ作成コマンド内の /K=<num> フラグを使って、ユーザーが閉じるボタンをクリックしたときの動作をコントロールできます。

DoWindow/HIDE=1 コマンドを使って、プログラムでウィンドウを非表示にできます。

ウィンドウの内容を保存

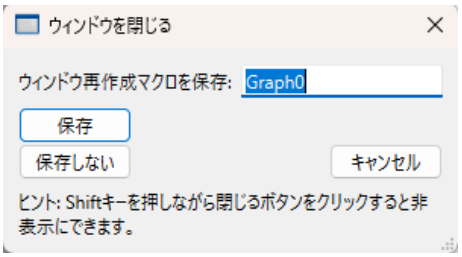
ノートブックとプロシージャウィンドウは、個別のファイルに保存することも、他のすべてとまとめて圧縮されたエクスペリメントファイルに保存することもできます。
「ノートブック」→「情報（Info）」または「プロシージャ」→「情報（Info）」を選択することで、どちらの場合か判別できます。

ノートブックまたはプロシージャウィンドウを終了させる時に、未保存の情報が含まれている場合、Igor はウィンドウを終了する前に保存するかどうかを確認します。

グラフ、テーブル、コントロールパネル、ページレイアウト、および Gizmo ウィンドウは個別のファイルとして保存されず、ウィンドウ再作成マクロを保存しない限り、それらを終了すると失われます。
このマクロを実行することで後からウィンドウを再作成できます。
これらのウィンドウを削除し、ウィンドウ再作成マクロ（組み込みプロシージャウィンドウに保存）として保存することで、情報を失うことなくメモリを解放し、ウィンドウの煩雑さを軽減します。

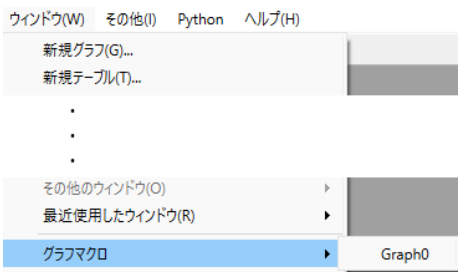
ウィンドウ再作成マクロは「フリーズドライされたウィンドウ」と考えることができます。

ウィンドウダイアログを閉じる



グラフ、テーブル、レイアウト、コントロールパネル、または Gizmo ウィンドウを閉じる時、Igor は「ウィンドウを閉じる」ダイアログを表示します。

「保存」ボタンをクリックすると、メインプロシージャウィンドウにウィンドウ再作成マクロを作成します。



これにより、マクロの名前がウィンドウメニューの適切な「マクロ（Macros）」サブメニューに表示されます。
このメニューを使ってウィンドウを再作成できます。

そのウィンドウを再度使わない場合は、「保存しない」ボタンをクリックしてください。
そうすると、ウィンドウ再作成マクロは作成されません。

以前にそのウィンドウ用の再作成マクロを作成している場合、ダイアログには「保存」ボタンの代わりに「置換」ボタンが表示されます。
「置換（Replace）」をクリックすると、古いウィンドウ再作成マクロが新しいものに置き換えられます。
ウィンドウを再作成する必要がないと分かっている場合は、マクロを削除できます（「ウィンドウを再作成マクロとして保存」のセクションを参照）。

ノートブックまたはプロシージャウィンドウ（組み込みプロシージャウィンドウを除く）を閉じると、Igor は「非表示または終了ダイアログ」を表示します。

ウィンドウを非表示にするには、閉じるボタンをクリックする時に Shift キーを押してください。

グラフ、テーブル、レイアウト、コントロールパネル、または Gizmo ウィンドウを閉じるダイアログを経由せずに終了するには、閉じるボタンをクリックする時に Alt キーを押してください。

Display、Edit、NewLayout、NewPanel、NewNotebook、または NewGizmo コマンドを使ってプログラムでウィンドウを作成する場合、/K フラグを使って閉じるボタンの動作を変更できます。

ウィンドウを再作成マクロとして保存

再作成マクロとして保存可能なウィンドウを閉じる時、「ウィンドウを閉じる」ダイアログを表示してマクロの作成を提案します。
Igor はウィンドウ再作成マクロを現在のエクスペリメントのメインプロシージャウィンドウに保存します。
このマクロはウィンドウよりもはるかに少ないメモリを使い、ウィンドウの煩雑さを軽減します。
ウィンドウ再作成マクロを後で呼び出して、ウィンドウを再作成できます。
また、「ウィンドウコントロール」ダイアログを使って、ウィンドウ再作成マクロを作成または更新することもできます。

ウィンドウ再作成マクロは、基盤となるデータがまだ存在している場合に、ウィンドウを再作成するために必要なすべてのコマンドを含んでいます。
例えば、グラフ再作成マクロには、ウェーブをグラフに追加するコマンドが含まれていますが、ウェーブデータ自体は一切含まれていません。
同様に、ページレイアウト再作成マクロには、グラフやテーブル、あるいはそれらを作成するコマンドは含まれていません。
マクロは、現在のエクスペリメントにおけるウェーブ、グラフ、テーブルを名前で参照します。

ウィンドウ再作成マクロは、ルートデータフォルダーのコンテキストで評価されます。
この詳細はプログラマーのみに関係します。
詳細については、ヘルプ Data Folders and Commands を参照してください。

ウィンドウコントロールダイアログを使うと、ウィンドウを終了させることなく、ウィンドウマクロを作成または置換できます。
ウィンドウ再作成マクロを置き換える最も一般的な理由は、そのマクロが作成するウィンドウとの一貫性を保つためです。

ウィンドウを閉じるダイアログを表示する時、ウィンドウ再作成マクロの提案名はウィンドウ名と同じになります。
必要に応じて、ダイアログに新しい名前を入力することで、ウィンドウ再作成マクロを別の名前で保存できます。
そうすると、Igor は新しいマクロを作成し、元のマクロはそのまま残ります。
新しいマクロを実行してウィンドウの新しいバージョンを作成することも、古いマクロを実行して以前のバージョンを再作成することもできます。
この方法であれば、複数のバージョンのウィンドウを保存しつつ、最新のバージョンのみを表示できます。

ウィンドウ再作成マクロはエクスペリメントのプロシージャウィンドウに無期限に残ります。
ウィンドウ再作成マクロが存在するウィンドウを再作成する必要がないと分かっている場合は、そのマクロを削除できます。

ウィンドウ再作成マクロを素早く見つけるには：

- 任意のプロシージャウィンドウをアクティブにし、Alt キーを押して、「ウィンドウ」メニューの該当する「マクロ」サブメニューからウィンドウ再作成マクロ名を選択します。

マクロを削除するには、今後使わないことを確認した上で、マクロの宣言行から End 行までのテキスト全体を選択してください。
Delete キーを押すと、選択したテキストが削除されます。

グラフに関する詳細は、ヘルプ Saving and Recreating Graphs を参照してください。

「ウィンドウ」メニューのマクロサブメニュー

「ウィンドウ（Windows）」メニューには、グラフ、テーブル、ページレイアウト、コントロールパネル、Gizmo 再作成マクロを含むサブメニューがあります。
これらのメニューには、グラフ、テーブル、ページレイアウトスタイルのマクロも含まれています。

ウィンドウ再作成マクロは、「ウィンドウを閉じる」、「ウィンドウコントロール」ダイアログ、DoWindow/R コマンドによって作成されます。
スタイルマクロは、「ウィンドウコントロール」ダイアログ、DoWindow/R/S コマンドによって作成されます。

Igor はマクロのサブタイプを検査することで、マクロを適切なマクロサブメニューに配置します。
サブタイプは、Graph、Table、Layout、Panel、Gizmo、GraphStyle、TableStyle、LayoutStyle です。
詳細はヘルプ Procedure Subtypes を参照してください。

マクロサブメニューから再作成マクロの名前を選択すると、マクロが実行され、ウィンドウが再作成されます。
スタイルマクロを選択すると、そのマクロが実行され、対象ウィンドウの外観（その「スタイル」）が変更されます。

ただし、プロシージャウィンドウが最前面にある状態で Alt キーを押した後、任意のマクロ名を選択すると、そのマクロが表示されますが実行はしません。

再作成されたウィンドウの名前

ウィンドウ再作成マクロを実行すると、そのマクロが作成したのと同じ名前のウィンドウを再作成します。
ただし、既に同じ名前のウィンドウが存在する場合を除きます。
その場合、Igor は新しく作成されたウィンドウの名前にアンダースコアと数字（例：_1）を追加し、既存のウィンドウと区別します。

マクロからウィンドウのスタイルを変更

「ウィンドウ（Windows）」メニュー、コマンドライン、または別のマクロからスタイルマクロを呼び出して実行すると、そのマクロ内のコマンドを最前面のウィンドウに適用します。
通常、これらのコマンドはウィンドウの外観を変更します。
例えば、グラフスタイルマクロは、グラフのトレースや軸の目盛りの色を変更する場合があります。

スタイルマクロはグラフウィンドウで最も効果的に使われます。
詳細については、ヘルプ Saving and Recreating Graphs と Graph Style Macros を参照してください。

ウィンドウコントロールダイアログ

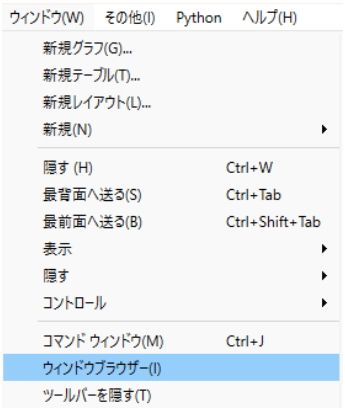
「コントロール（Control）」→「ウィンドウコントロール（Window Control）」を選択すると、ウィンドウコントロールダイアログが表示されます。
このダイアログでは、最前面ウィンドウのタイトルと名前を変更、再作成、そしてスタイルマクロを作成または更新できます。
このダイアログは Ctrl+Y を押すことで素早くアクセスできます。

ウィンドウの名前も変更できます。
ウィンドウ名は、MoveWindow などのコマンドライン操作からウィンドウを指定するために使われ、また Windows メニューのマクロサブメニューにも表示されます。

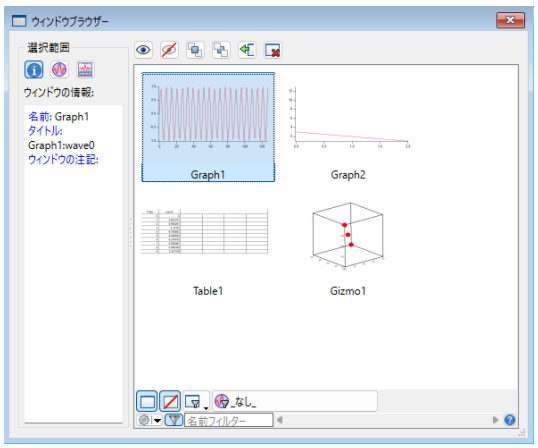
名前とタイトルの詳細については、「ウィンドウ名とタイトル」のセクションを参照してください。
ウィンドウ再作成マクロについては「ウィンドウを再作成マクロとして保存」のセクションも参照してください。
スタイルマクロの詳細については、ヘルプ Graph Style Macros を参照してください。

ウィンドウブラウザー

ウィンドウブラウザーは、目的のウィンドウを検索、管理できます。
「ウィンドウ」→「ウィンドウブラウザー」を選択して起動します。

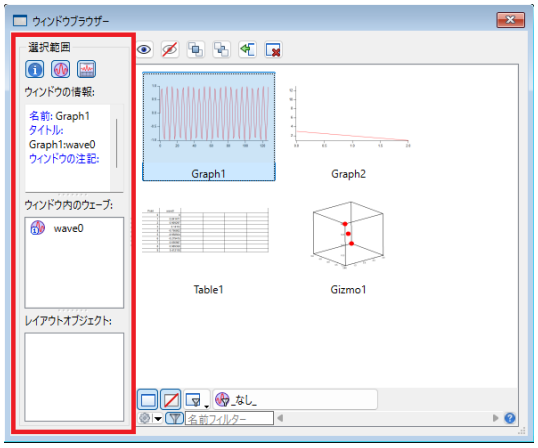


右側のウィンドウリストには、現在のフィルタリング条件に合致するウィンドウが表示されます。
ウィンドウは、名前、タイプ、表示状態（表示中または非表示）、および特定のウェーブを表示しているかどうかに基づいてフィルターできます。
リスト下の漏斗アイコンをクリックすると、詳細フィルタリングコントロールの表示が切り替わります。
歯車アイコンをクリックするとオプションメニューが表示され、並べ替えや表示オプションを設定できます。



ウィンドウリストで１つのウィンドウが選択されている場合、左側のコントロールにはそのウィンドウに関する情報が表示されます。
「選択範囲」の上部にあるボタンは、表示する情報の種類をコントロールします。

・ウィンドウの情報



ウィンドウ名、ウィンドウタイトル、ウィンドウノートを表示します。
名前とタイトルの区別については「ウィンドウ名とタイトル」のセクションを参照してください。

・ウィンドウ内のウェーブ

ウィンドウで使われているウェーブの一覧を表示します。
ウェーブアイコンを右クリックすると、テーブルでウェーブを編集したり、データブラウザで表示したり、現在のデータフォルダーをそのウェーブのデータフォルダーに設定したりできます。

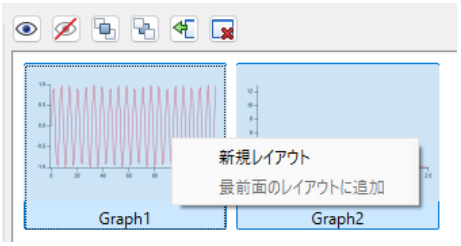
リスト内の任意の場所で右クリックすると、フルパスの表示を切り替えることもできます。

リストから１つ以上のウェーブを選択し、既存のグラフまたはテーブルウィンドウにドラッグすると、対象ウィンドウに追加できます。
詳細はヘルプ Appending Traces by Drag and Drop を参照してください。

・レイアウトオブジェクト

選択したページレイアウトウィンドウ内のページおよび各ページ上のオブジェクトに関する情報を表示します。

複数のウィンドウが選択されている場合、「ウィンドウの情報」コントロールはウィンドウ名のリストを表示します。



ウィンドウリストの上にあるボタンを使うと、選択したウィンドウの表示、非表示、閉じる操作が可能で、前面に表示したり、背面に送ったりできます。

選択したグラフ、テーブル、Gizmo ウィンドウをページレイアウトに追加するには、ウィンドウリストからレイアウトへドラッグするか、ウィンドウリストを右クリックして適切な項目を選択します。

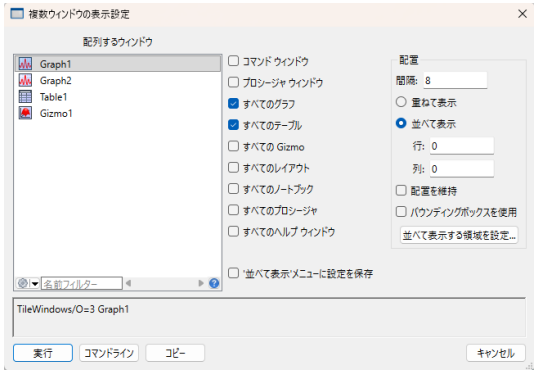
ウィンドウリストのコンテキストメニューにメニュー項目を追加できます。
詳細はヘルプ WindowBrowserWindowsPopup Menu を参照してください。

ウィンドウの配置

「ウィンドウ」メニューの「コントロール」サブメニューから適切な項目を選択することで、ウィンドウをタイル表示または積み重ね表示できます。

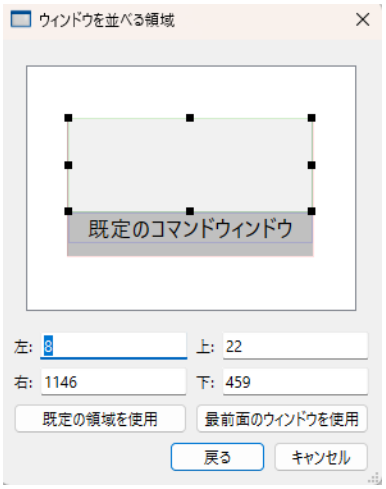
「複数ウィンドウの表示設定（Tile or Stack Windows）」ダイアログを使って、タイル、スタック項目の動作をカスタマイズできます。

タイル（またはスタック）メニュー項目に続く選択で、同じ行・列・間隔・タイル領域などを共有する同じ種類のウィンドウを積み重ねたい場合は、「配置を維持（Capture as pref）」チェックボックスをオンにしてください。
「配列するウィンドウ」メニューで選択されたウィンドウはプレファレンスに記憶されません。
記憶されるのはウィンドウタイプのチェックボックスのみです。
スタックとタイルにはそれぞれ個別の設定とプレファレンスがあります。



TileWindows および StackWindows コマンドはパネルをタイル状に配置したり積み重ねたりできますが、パネルはサイズ変更がうまくいかないため、ここでは表示されません。

「並べて表示する領域を設定（Window Tiling Area）」サブダイアログは、タイリングとスタッキングが行われる領域を指定します。
タイリング領域は次の４つの方法から指定できます：



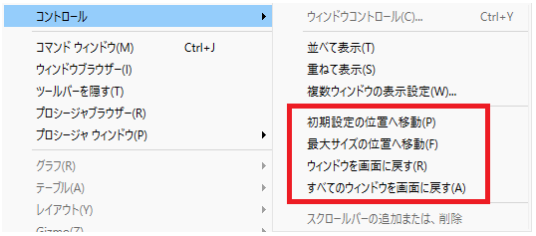
- ポイント単位で画面の位置を入力する
- タイリング領域の図形表示をドラッグする
- 「既定の領域を使用（Use default area）」ボタンをクリックして、既定のタイリング領域を使う
- ダイアログに入る前に任意のダイアログではないウィンドウを前面に配置し、「最前面のウィンドウを使用（Use top window）」ボタンをクリックする

ウィンドウの位置とサイズの管理

「ウィンドウ（Windows）」メニューの「コントロール（Control）」サブメニューには、ウィンドウの位置やサイズを管理するのに役立つ４つの項目があります。

初期設定の位置へ移動

アクティブなウィンドウを、プレファレンスで指定された位置とサイズに移動します。
各ウィンドウタイプごとに、「環境設定を保存（Capture Prefs）」ダイアログ（例：グラフの場合は「グラフ環境設定を保存」）を使って、優先位置とサイズを設定できます。



最大サイズの位置へ移動

アクティブなウィンドウを移動およびサイズ変更し、可能な限り多くのコンテンツを表示します。
サイズはフレームウィンドウのサイズに制限されます。

ウィンドウを画面に戻す

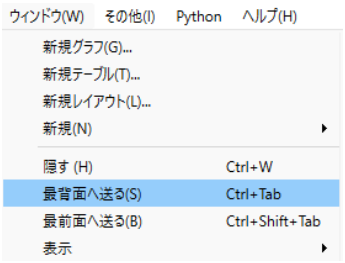
アクティブなウィンドウを移動し、必要に応じてサイズを変更して、ウィンドウ全体が表示されるようにします。

すべてのウィンドウを画面に戻す

すべてのウィンドウを移動し、必要に応じてサイズを変更して、各ウィンドウの全体がフレーム内に収まるようにします。
これは、自分のディスプレイよりも大きな画面や Windows のフレームで作成されたエクスペリメントを開く時に役立ちます。

最背面に送る – 最前面に送る

「ウィンドウ（Windows）」メニューの「最背面に送る（Send to Back）」は、最前面のウィンドウをデスクトップの最背面、他のすべてのウィンドウの後ろに移動します。
この機能は Ctrl+Tab を押すことでもアクセスできます。
ウィンドウを背面に送った後、「最前面に送る（Send to Back）」を選択するか Ctrl+Shift+Tab を押すことで前面に戻せます。
これらのキーを繰り返し押すと、すべてのウィンドウを切り替えることもできます。



DoWindow/B コマンドでウィンドウを背面に移動させ、DoWindow/F コマンドで前面に移動させることができます。

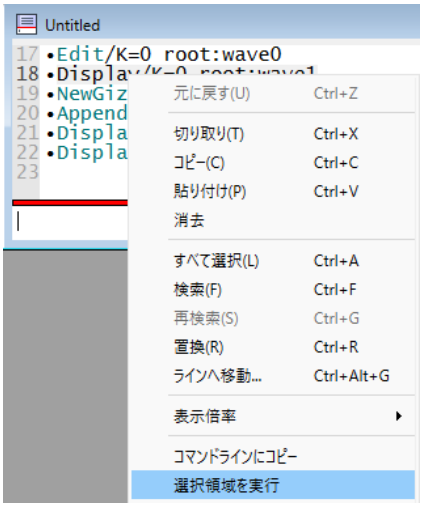
テキストウィンドウ

Igor Pro は、プロシージャ、ノートブック、Igor ヘルプウィンドウ、コマンドウィンドウのコマンドエリアと履歴エリアにテキストを表示します。
このセクションでは、これらすべてのウィンドウに共通する動作について説明します。

コマンドの実行

ノートブック、プロシージャ、またはヘルプウィンドウで選択したコマンドは、Control-Enter または Control-Return を押すことで実行できます。
選択したコマンドは、右クリックして「選択領域を実行（Execute Selection）」を選択することで実行することもできます。

詳細については、ヘルプ Notebooks as Worksheets を参照してください。



テキストウィンドウのナビゲーション

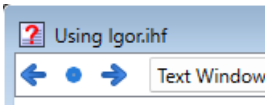
「キーボードナビゲーション」とは、矢印キーおよび Home、End、Page Up、Page Down キーへの応答として Igor が実行する選択やスクロール操作を指します。

キー	**他のキー操作なし**	**Ctrl キー**
左矢印	選択を 1 文字左に移動	選択を 1 単語左に移動
右矢印	選択を 1 文字右に移動	選択を 1 単語右に移動
上矢印	選択を 1 行上に移動	選択を 1 段落上に移動
下矢印	選択を 1 行下に移動	選択を 1 段落下に移動
Home	選択を行の先頭に移動	選択をドキュメントの先頭に移動
End	選択を行の最後に移動	選択をドキュメントの最後に移動
Page Up	1 画面スクロールアップ	
Page Down	1 画面スクロールダウン	

テキストウィンドウの戻る／進む

特定の操作は、メモリ内に保持する戻り先スタックにエントリを追加します。
これらの操作には、テキストウィンドウを開く、検索を実行する、ヘルプリンクをクリックする、右クリックして「ヘルプ（Help For）」メニューと「移動（Go To）」メニュー項目を選択する、ナビゲーションバーを使う、などが含まれます。

テキストウィンドウ内で最近訪れた場所へは、「戻る」ボタンと「進む」ボタンを使って戻ることができます。



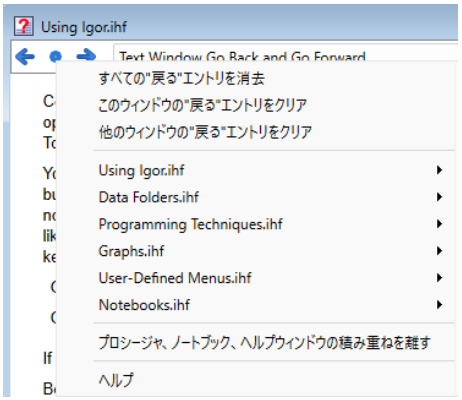
これらはプロシージャウィンドウおよびヘルプウィンドウのナビゲーションバーと、ノートブックウィンドウのステータス領域に表示されます。
「編集（Edit）」メニューの「戻る（Go Back）」と「次へ進む（Go Forward）」、または同等のキーボード操作も使用できます：

戻る　　Alt+左矢印

進む　　Alt+右矢印

マウスに「戻る」ボタンと「進む」ボタンがある場合は、それらも使用できます。

「戻る」ボタンと「進む」ボタンの間には、クリックすると「戻る」ポップアップメニューを表示するボタンがあります。
「戻る」ポップアップメニューのサブメニューから項目を選択することで、最近訪れた特定の場所に戻ることができます。



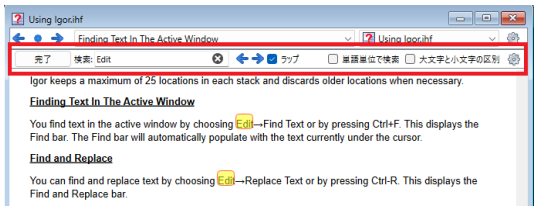
Igor は、すべてのテキストウィンドウの戻る位置を 1 つのスタックに保持することも、プロシージャウィンドウ、ノートブックウィンドウ、ヘルプウィンドウごとに個別のスタックを保持することもできます。
この操作は、「戻る」ポップアップメニュー内の「プロシージャ、ノートブック、ヘルプウィンドウの積み重ねを離す（Separate stacks for procedure, notebook, help windows）」メニュー項目を使ってコントロールします。

その項目がチェックされている場合、Igor は個別のスタックを持ちます。
チェックされていない場合、 3 種類のウィンドウすべてに対して 1 つのスタックを持ちます。
どちらを選ぶかは好みの問題です。
複数のスタックを維持する利点は、大抵の場合、同じタイプのウィンドウに関心がある点にあります。
スタックを 1 つだけ保持する利点は、コマンドや関数のヘルプを右クリックで表示した後、そのプロシージャファイルに戻れる点です。

Igor は各スタックに最大 25 個の場所を保持し、必要に応じて古い場所を破棄します。

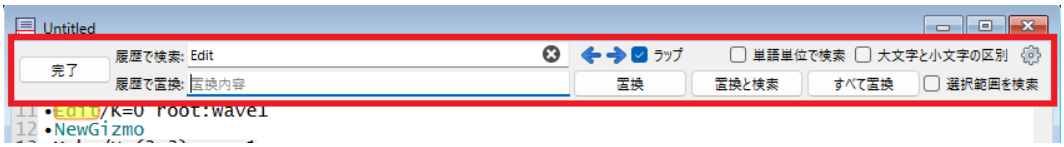
アクティブなウィンドウ内のテキストの検索

アクティブなウィンドウ内のテキストは、「編集（Edit）」→「検索（Find）」を選択するか、Ctrl+F を押すことで検索できます。
これにより、検索バーが表示されます。
検索バーには、カーソルの位置にあるテキストが自動的に入力されます。



検索と置換

テキストの検索と置換は、「編集（Edit）」→「置換（Replace）」を選択するか、Ctrl+R を押すことで実行できます。
これにより「検索と置換」バーが表示されます。



テキストを検索して置換する別の方法は次のとおりです：

1. 選択範囲をアクティブなウィンドウの最上部に移動します。

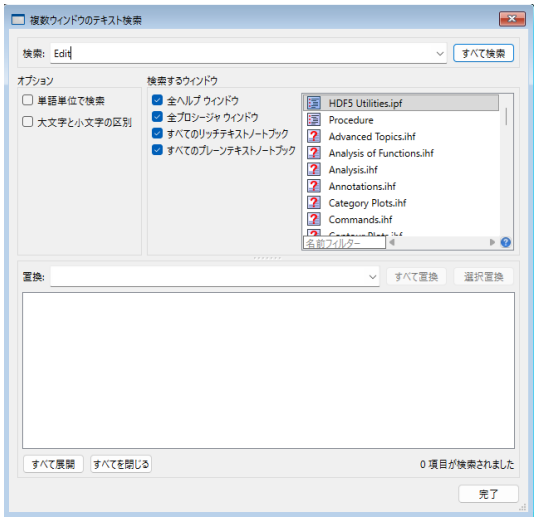
2. 「編集（Edit）」→「検索（Find）」を選択し、対象文字列の最初の出現箇所を検索します。

3. 最初のインスタンスを手動で変更し、新しいテキストをクリップボードにコピーします。
4. Ctrl+G キーを押すと、次の出現箇所を検索します。
5. Ctrl＋V キーを押して、ペーストします。
6. 完了するまでステップ 4 と 5 を繰り返します。

複数のウィンドウ内のテキストの検索

「編集（Edit）」→「複数ウィンドウの検索（Find in Multiple Windows）」を選択すると、複数のウィンドウ内でテキストを検索できます。
「複数ウィンドウの検索（Find in Multiple Windows）」ウィンドウが表示されます。
これにより、すべてのヘルプウィンドウ、すべての手順ウィンドウ、およびすべてのノートブックを検索できます。

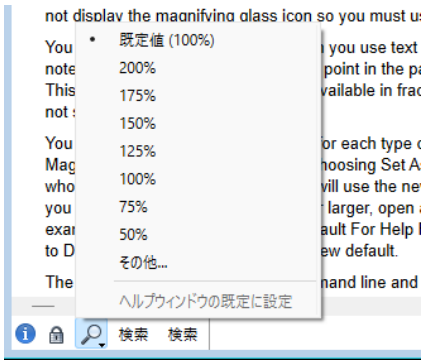
ヘルプブラウザーを使うと、現在のエクスペリメントで開いていないファイルを含む複数のファイルを検索できます。
詳細は「Igor ヘルプブラウザー」のセクションを参照してください。



テキストの拡大縮小

どのウィンドウでもテキストを拡大縮小して、好みに合わせて大きくしたり小さくしたりできます。

ヘルプウィンドウ、プロシージャウィンドウ、プレーンテキストノートブック、および書式付きテキストノートブックでは、ウィンドウの左下隅にある虫眼鏡アイコンを使用できます。
ウィンドウのコンテキストメニューにある「表示倍率（Magnification）」サブメニューも使用できます。
コンテキストメニューを表示するには、ウィンドウ本体を右クリックしてください。



コマンドライン、履歴エリア、デバッガーの拡大率も設定できます。
これらのエリアには虫眼鏡アイコンが表示されないため、コンテキストメニューを使う必要があります。

テキスト拡大縮小機能を使うと、いくつかの異常が発生する場合があります。
例えば、書式設定されたテキストノートブックでは、段落内の改行位置が異なる場合があり、タブストップとの関係で変更されることがあります。
これは、フォントが小数点以下のサイズで利用できないことと、テキストの実際の幅がフォントサイズに比例して線形に拡大縮小しないためです。

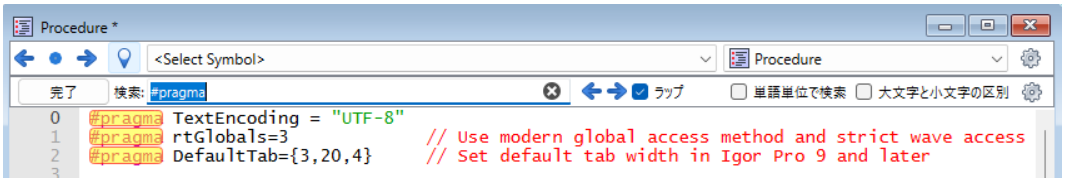
各テキストエリアのデフォルトの倍率を設定するには、倍率ポップアップメニューから倍率を選択し、同じポップアップメニューから「既定に設定」を選択します。
倍率が「既定値」に設定されているテキストエリアは、新たに指定された既定の倍率を使います。
例えば、すべてのヘルプファイルのテキストを大きく表示したい場合は、任意のヘルプファイルを開き、倍率を大きく設定します（例：125%）。
その後、「ヘルプファイルの既定値に設定（Set As Default For Help Files）」を選択します。
現在の倍率が「既定」に設定されているすべてのヘルプファイルは、新しい既定値を使うように更新されます。

コマンドラインおよび履歴エリアのデフォルトの倍率は、次回 Igor Pro を起動した時に使われる倍率をコントロールします。

倍率の設定は、書式設定済みのノートブックとヘルプファイルにのみ保存されます。
これらのファイルのいずれかの倍率の設定を変更し、ファイルを保存して閉じると、ファイルを再度開いた時に倍率の設定が復元されます。
その他のすべてのテキストエリア（プロシージャウィンドウやプレーンテキストノートブックを含む）については、倍率の設定はファイルに保存されません。
そのようなファイルを閉じて再度開くと、その種類のテキストエリアのデフォルトの倍率で再開されます。

検索バー

検索バーはヘルプ、プロシージャ、ノートブックウィンドウ、およびテキスト検索が必要なその他の場所で利用できます。
表示するには、「編集（Edit）」→「検索（Find）」を選択するか、Ctrl+F を押します。
検索バーはホストウィンドウの上部に表示され、カーソルの下の現在のテキストが自動的に入力されます：

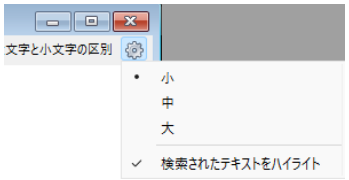


必要に応じて「検索」編集ボックス内のテキストを編集し、右矢印アイコンを押すと次の出現箇所を検索します。
左矢印アイコンを押すと前の出現箇所を検索します。

Ctrl+G を押すと、次の出現箇所を検索できます。
Ctrl+Shift+G を押すと、前の出現箇所を検索できます。

デフォルトでは、これらのキーボードショートカットを使う時、検索バーが表示されていない場合、それを表示します。

歯車アイコンを使って、検索バーのサイズを設定できます。



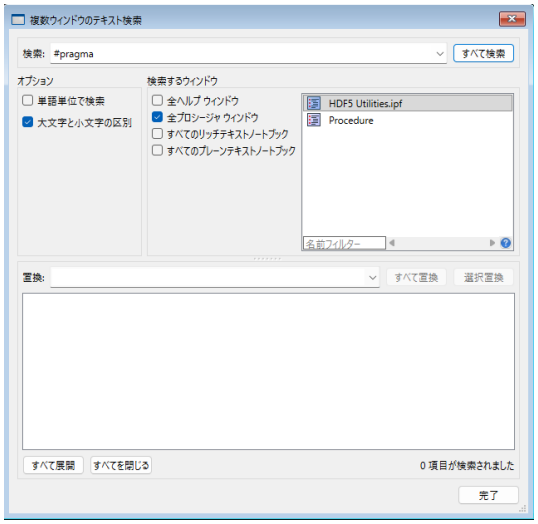
検索バーを非表示にするには、「完了（Done）」ボタンをクリックするか、Esc キーを押します。

検索と置換

編集中のウィンドウに編集可能なテキストが含まれている場合、「編集（Edit）」→「置換（Replace）」を選択するか、Ctrl+R キーを押すと、検索バーの置換テキスト部分が表示されます：

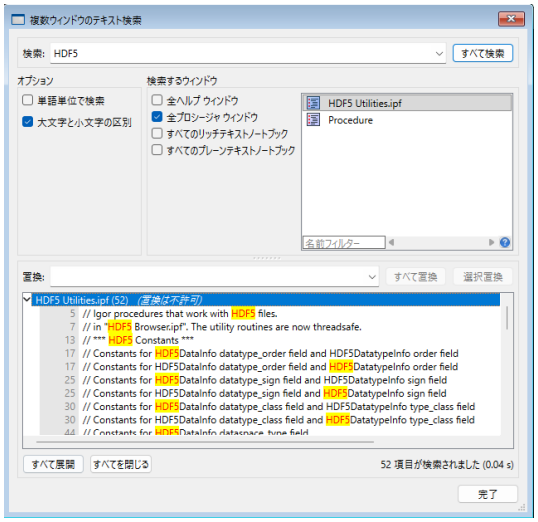
テキストの置換は元に戻せますが、意図せず広範囲に影響を及ぼす可能性があるため、一括置換（全置換）を実行する前にファイルを保存することを推奨します。
これにより、必要に応じて保存状態に戻すことが可能です。

「複数ウィンドウのテキスト検索」ダイアログ



複数のヘルプ、プロシージャ、ノートブックウィンドウで同時に検索を実行するには、「編集（Edit）」→「複数ウィンドウの検索（Find in Multiple Windows）」を選択するか、Ctrl+Shift+F を押します。
これにより「複数ウィンドウのテキスト検索」ダイアログが表示されます。

「検索（Find）」ボックスに検索するテキストを入力してください。
チェックボックスを使って、検索対象のウィンドウの種類を選択します。
検索対象のファイル一覧を絞り込むには、ウィンドウ一覧の下部にあるフィルターボックスにフィルター文字列を入力できます。リストには、名前がフィルター文字列を含むリスト項目のみが表示されます。



「すべて検索（Find All）」ボタンをクリックすると、リストされたウィンドウ内のすべてのテキストの検索を開始します。
検索中、検索がウィンドウリストのどこまで進んだかを示す小さな進行状況ダイアログを表示します。
キャンセルボタンをクリックすると検索を停止できます。

テキストが見つかった各ウィンドウごとに、ダイアログの下部にあるパネルにエントリが表示されます。
括弧内の数字は、そのウィンドウ内で検索テキストが何回見つかったかを示します。

これらのウィンドウエントリはそれぞれ、項目の左端にある「>」を使って開くことができます。
検索文字列の周囲のテキストの一部が表示され、検索文字列が強調表示されます。

検索されたテキストスニペットの左端にある数字は、検索対象ウィンドウ内の行番号です。

テキストスニペット項目をダブルクリックすると、該当するテキストを含むウィンドウ内のそのテキストに移動します。

複数ウィンドウ内の検索と置換

検索が完了した後、見つかったテキスト項目を使って、編集可能なウィンドウ内で見つかったテキストのすべてのインスタンスを置換できます。
置換するテキストを「置換（Replace）」編集ボックスに入力してください。

「すべて置換（Replace All）」ボタンをクリックすると、見つかったすべての箇所が置換テキストに置き換えられます。

警告：複数ウィンドウでの「すべて置換」は取り消せません。十分注意して使用してください。

誤った操作を行った場合は、「ファイル（File）」→「エクスペリメントを元に戻す（Revert Experiment）」、「ファイル」→「ノートブックを元に戻す（Revert Notebook）」、または「ファイル」→「プロシージャ元に戻す

（Revert Procedure）」を選択し、影響を受けたすべてのウィンドウが元に戻っていることを確認してください。これが復元する唯一の方法です。

検索されたテキスト項目のサブセットを選択し、「選択範囲を置換（Replace Selected）」ボタンをクリックすることで、置換範囲を制限できます。
見つかったテキスト項目をクリックして選択します。
Shift キーを押しながら別の項目をクリックすると、その項目と前にクリックした項目の間のすべての項目が選択されます。
非連続の項目を選択するには、Ctrl キーを押しながらクリックします。
複数の項目をクリックしてドラッグすることもできます。
置換前に保存しておくことをお勧めします。

ホームテキストファイル vs 共有テキストファイル

パックされたエクスペリメントファイル内に保存されるテキストファイル、またはパックされていないエクスペリメントに対してはエクスペリメントフォルダー内に保存されるテキストファイルが、「ホーム」テキストファイルです。
それ以外の場合は「共有」テキストファイルです。

ホームテキストファイルは通常、それが属するエクスペリメントでのみ使うことを意図しています。
共有テキストファイルは通常、複数のエクスペリメントでの使用を意図しています。

パックされたエクスペリメントでテキストファイルを作成すると、デフォルトではパックされたエクスペリメントファイル内に保存され、ホームテキストファイルとなります。
明示的にスタンドアロンファイルとして保存した場合のみ、共有テキストファイルになります。

未圧縮のエクスペリメントでテキストファイルを作成すると、デフォルトではエクスペリメントフォルダー内に保存され、ホームテキストファイルとなります。
エクスペリメントフォルダー外に独立したファイルとして明示的に保存した場合のみ、共有テキストファイルとなります。

パックされたエクスペリメントをパック解除状態で保存すると、ホームテキストファイルがエクスペリメントフォルダーに保存されます。

未圧縮のエクスペリメントを圧縮形式で保存すると、ホームテキストファイルが圧縮エクスペリメントファイルに保存されます。

「プロシージャ（Procedure）」→「情報（Info）」または「ノートブック（Notebook）」→「情報（Info）」を選択してアクセスする「ファイル情報（File Information）」ダイアログを使って、テキストファイルが共有されているかどうかを確認できます。
共有テキストファイルの場合、ダイアログには「このファイルはエクスペリメントファイルとは別に保存されています」と表示されます（圧縮済みおよび非圧縮の実験の両方において）。
ホームテキストの場合、ダイアログには「このファイルはエクスペリメントファイル内に圧縮形式で保存されています」と表示され（圧縮エクスペリメントの場合）、「このファイルはエクスペリメントフォルダー内にあります」と表示されます（非圧縮エクスペリメントの場合）。

共有テキストファイルをホームテキストファイルに適用するには、それを変換してください。
詳細はヘルプ Adopting Notebook and Procedure Files を参照してください。

外部のテキストエディターの使用

Igor は、プロシージャファイルやプレーンテキストノートブックの編集に外部エディターの使用をサポートしています。

これにより、プレーンテキストファイルの編集に Igor ではなく外部のテキストエディタを使うことが可能です。
これは主に、他のプログラミング環境で外部エディタの使用に慣れている上級プログラマーを対象としています。

Igor 7 以前では、対応するプロシージャまたはノートブックウィンドウが Igor で開かれている限り、Igor はプレーンテキストファイルを開いたままにしていました。
これにより、外部エディタの使用に支障が生じていました。

現在は、プロシージャまたはプレーンテキストのノートブックウィンドウは、テキストをメモリに読み込むのに必要な時間だけファイルを開き、その後すぐに閉じます。
Igor でテキストを編集して保存すると、ファイルを再度開き、変更されたテキストを書き込んでからファイルを閉じます。

Igor 以外の外部テキストエディタでファイルを変更した場合、Igor はその変更を検知し、ファイルを再度開いて変更されたテキストをメモリに再読み込みした後、ファイルを閉じます。

外部エディタのサポートは、Igor が対処しなければならない問題を引き起こします：

- 外部エディタでファイルを変更すると、テキストは Igor のウィンドウ内のテキストと同期が取れなくなります。この場合、Igor はファイルが変更されたことを検知し、外部エディタの各種設定に応じて、テキストをメモリに再読み込みするか、変更を通知します。
- 外部エディタでファイルの変更を保存した後、Igor でも文書を編集すると、Igor 内のテキストと外部ファイルが競合します。この場合、Igor は競合を通知し、どちらのバージョンを保持するか選択できます。
- ディスクファイルを新しい場所に移動したり、ファイルを削除または名前変更すると、Igor のファイル情報は無効になります。この場合、Igor はファイルが消失したことを検知し、対応方法の選択肢を提供します。

以下のセクションでは、Igor がこれらの問題をどのように処理するかをより詳細に説明します。

プレーンテキストファイルが外部で変更された

Igor は 1 秒に 1 回、すべてのプロシージャとプレーンテキストのノートブックウィンドウをチェックし、そのファイルが外部から変更されたかどうかを確認します。
その場合、ファイルが自動的にメモリに再読み込みされるか、外部での変更を通知し、ユーザーが任意のタイミングでファイルを再読み込みできるようにします。
製品出荷時のデフォルト動作は、外部で変更されたファイルを自動的に再読み込みし、変更されたファイルのチェックを Igor がアクティブなアプリケーションである場合のみ行うようになっています。

Igor の動作は、「その他の設定（Miscellaneous Settings）」ダイアログでコントロールできます。
「その他（Misc）」→「その他の設定（Miscellaneous Settings）」を選択し、左ペインで「テキスト編集（Text Editing）」をクリックし、「外部エディタ（External Editor）」タブをクリックします。

「再ロードする前に確認する（Ask Before Reloading）」を選択した場合、ファイルが外部で変更されると次の 2 つの処理が行われます：

- Igor はドキュメントウィンドウ下部のステータス領域に「再ロード（Reload）」ボタンを表示します。このボタンはウィンドウが表示されている場合にのみ表示されます。
- Igor はフローティング通知ウィンドウを表示します。

「再ロード」ボタンをクリックすると、Igor がファイルのテキストを再読み込みします。

通知ウィンドウの「レビュー（Review）」ボタンをクリックすると、現在外部で変更されているすべてのファイルを確認できるダイアログが表示されます。

「選択した項目を解決（Resolve Checked Items）」をクリックすると、ファイルがメモリに再読み込みされます。

プレーンテキストファイルが外部および Igor 内で変更された

ファイルが外部エディタと Igor ウィンドウの両方で編集された場合、そのファイルは「競合状態」にあると言います。
Igor は競合状態のファイルを自動的に再読み込みすることはありません。

ファイルに競合が発生した場合、Igor はドキュメントウィンドウのステータス領域に「競合を解決（Resolve Conflict）」ボタンを表示します。
このボタンをクリックすると、競合に対処するための 4 つの選択肢を示すダイアログが表示されます：

- 外部の変更を再ロード（Reload External Changes）
  ファイルの内容をメモリに読み込み、Igor で行われた変更を破棄します。
- ドキュメントを採用（Adopt Document）
  文書とファイルの関連付けを解除し、Igor での変更は保持し、ファイルへの変更は破棄します。
  ファイルの採用について詳しくは、ヘルプ Adopting Notebook and Procedure Files を参照してください。
- ウィンドウを閉じる（Close the Window）
  Igor 内のプロシージャまたはノートブックウィンドウを閉じ、Igor で行った変更を破棄します。
- Igor の変更を保存（Save Igor Changes）
  Igor で行った変更をファイルに書き込みます。これにより Igor 内の競合が解決され、外部エディタに競合が生じます。外部エディタによって、このような競合への対応方法は異なります。

競合しているファイルも通知ウィンドウを表示させます。通知ウィンドウの「レビュー（Review）」ボタンをクリックすると、競合を解決するための同じオプションを提供するダイアログが表示されます。

外部での変更のあるファイルを編集する

ファイルが外部で変更され、Igor で再読み込みされていない場合、Igor のウィンドウで入力すると、以前に競合がなかった場合でも競合が発生します。
これは望ましくない可能性があるため、この状況を検出すると、競合解決ダイアログと同様のダイアログを表示し、どうすべきかを尋ねます。
選択肢は：

- 入力を許可（Allow Typing）
  この選択は、Igor で文書を編集することを優先します。
  衝突の解決は後回しにします。
  いずれかの時点で、上記のいずれかの方法で対立を解決する必要があります。
- 外部の変更を再ロード（Reload External Changes）
  ファイルの内容をメモリに読み込み、Igor で行われた変更を破棄します。
- ドキュメントを採用（Adopt Document）
  文書とファイルの関連付けを解除し、Igor での変更は保持し、ファイルへの変更は破棄します。
  ファイルの採用について詳しくは、ヘルプ Adopting Notebook and Procedure Files を参照してください。

「キャンセル（Cancel）」ボタンをクリックする選択肢もあります。
その場合、状況は解決されず、ウィンドウ内でさらに文字を入力しようとすると、ダイアログが再度表示されます。

プレーンテキストファイルが見つからない

Igor のプレーンテキスト文書ウィンドウに関連付けられたファイルが移動、名前変更、または削除された場合、そのファイルが欠落していることを認識します。
この場合、Igor ドキュメントウィンドウのステータスエリアに「ファイルが欠落（File Missing）」ボタンを表示します。
そのボタンをクリックすると、「ファイルが欠落（File Missing）」ダイアログが表示され、競合に対処するための４つの選択肢が提示されます：

- ドキュメントを採用（Adopt Document）
  文書とファイルの関連付けを解除し、Igor での変更は保持し、ファイルへの変更は破棄します。
  ファイルの採用について詳しくは、ヘルプ Adopting Notebook and Procedure Files を参照してください。
- ウィンドウを閉じる（Close the Window）
  Igor 内のプロシージャウィンドウまたはノートブックウィンドウを閉じます。
- 欠落したファイルを検索（Find Missing File）
  移動または名前変更されたファイルを選択できる「ファイルを開く（Open File）」ダイアログを表示します。
- 名前を付けて保存（Save As）
  「名前を付けて保存（Save As）」ダイアログを表示し、文書を新しいファイルとして保存できるようにします。

「テキストファイルの欠落または修正（Missing or Modified Text Files）」ダイアログ

プレーンテキストファイル（プロシージャファイルまたはプレーンテキストノートブック）が欠落している可能性があります。
これは、ユーザーが削除、名前変更、移動したか、プログラムによって削除、名前変更、移動されたためです。

例えば、外部エディターで編集した場合など、プレーンテキストファイルは他のプログラムによって変更されている可能性があります。

プレーンテキストファイルが欠落または変更されている状態で保存を試みると、Igor は「欠落または変更されたファイル（Missing or Modified File）」ダイアログを表示し、保存をキャンセルするか続行するかを尋ねます。
通常は保存をキャンセルし、画面右上に表示される「ファイルが外部で変更されました（Files Were Modified Externally）」ウィンドウを使って状況を確認し対処する必要があります。

保存を続行することを選択した場合、エラーが発生する可能性があります。

自動処理がハングアップするのを防ぐため、プロシージャから保存が呼び出された場合、欠落または変更されたファイルのチェックは行われません。

オブジェクト名

すべての Igor オブジェクトには名前があり、これはオブジェクト作成時に指定するか、Igor が自動的に割り当てます。
オブジェクトの名前は、ダイアログ内、コマンドから、プロシージャからそのオブジェクトを参照するために使いま

す。
名前が付けられるオブジェクトは以下の通りです：

データフォルダー	ウェーブ	変数（数値・文字列）
ウィンドウ	軸	注釈
コントロール	ルーラー	特殊文字（ノートブック内）
シンボリックパス	画像	
FIFO	FIFO チャンネル	XOP

Igor Pro では、ウェーブおよびデータフォルダーの命名規則は、文字列変数や数値変数を含む他のすべてのオブジェクトの命名規則ほど厳格ではありません。
後者には標準の名前が要求されます。
以降のセクションでは、標準の命名規則と自由な命名規則について説明します。

標準のオブジェクト名

標準の有効なオブジェクト名の規則は以下の通りです：

- 長さは 1 バイトから 255 バイトまでです。
  Igor Pro 8.0 以前では、名前は 31 バイトに制限されていました。
  31 バイトを超える名前に関連する問題については、「長いオブジェクト名」のセクションを参照してください。
- アルファベット文字（A-Z または a-z）で始めなければなりません。
- ASCII アルファベット文字、数字、またはアンダースコア文字を含むことができます。
- 他の名前（コマンド、関数など）と競合できません。

Igor 内のすべての名前は大文字小文字を区別しません。
「wave0」と「WAVE0」は同じウェーブを指します。

英数字以外の文字（スペースやピリオドを含む）は使用できません。
この制限を設けることで、コマンド（ウェーブフォーム算術式を含む）内で名前を明確に識別できるようにしています。

自由なオブジェクト名（リベラル名）

リベラル名の命名規則は、ウェーブデータフォルダーとデータフォルダーにのみ適用されます。
コマンドやウェーブフォーム算術式でリベラル名を使う時、追加の手間をかけることを厭わなければ、ほぼあらゆる文字を含むウェーブおよびデータフォルダー名を使用できます。
リベラル名を作成する場合、コマンドやウェーブフォーム演算式で使う時には、必ず名前をシングルクォート（カーリークォートではない）で囲む必要があります。
これは、Igor が名前の終わりを判別できるようにするために必要です。
リベラル名は標準的な名前と同じ規則に従いますが、制御文字および以下の文字を除き、任意の文字を使用できます：

```
"  '  :  ;
```

リベラル名の作成と使用の例を次に示します：

```
Make 'wave 0';                // 'wave 0' はリベラル名
'wave 0' = p
Display 'wave 0'
```

注記：
リベラル名を指定するには、ユーザー自身による追加の努力とテストが必要です（ヘルプ Programming with Liberal Names を参照）。
ユーザー定義プロシージャでリベラル名を使う時、時々問題が発生する可能性があります。

名前空間

オブジェクトを名前で参照する場合（例えばユーザー関数内など）、参照対象のオブジェクトを明確に特定できなければなりません。
一般的に、オブジェクトは一意の名前を持つ必要があり、そうすることで、どのオブジェクトを指しているのかを判別できます。
Igor は文脈からオブジェクトの型を判別できる場合があり、その場合は他の型のオブジェクトと同じ名前を使用できます。
名前が同じにすることができるオブジェクトは、異なる名前空間にあるとも言えます。

データフォルダーは独自の名前空間にあります。
したがって、データフォルダーの名前は、階層構造の同じレベルにある別のデータフォルダーを除き、他のオブジェクトの名前と同じにすることができます。

ウェーブと変数（数値型および文字列型）は同じ名前空間にあるため、同じデータフォルダー内に同じ名前のウェーブと変数を同時に作成することはできません。

注釈は、それを含むウィンドウにローカルです。
その名前は、同じウィンドウ内の注釈間で一意である必要があります。
コントロールやルーラーについても同様です。
データフォルダー、ウェーブ、変数、ウィンドウ、シンボリックパス、および画像はグローバルオブジェクトであり、特定のウィンドウに関連付けられていません。

グローバルオブジェクトの名前は、データフォルダーを除き、マクロ、関数（組み込み、外部、またはユーザー定義）、およびコマンド（組み込みまたは外部）の名前と区別される必要があります。

以下は４つのグローバル名前空間の概要です：

名前空間	**要件**
データフォルダー	名前は、階層構造の同じレベルにある他のデータフォルダーと区別する必要があります。
ウェーブ、変数、ウィンドウ	名前は他のウェーブ、変数（数値および文字列）、ウィンドウと区別する必要があります。
画像	名前は他の画像と区別する必要があります。
シンボリックパス	名前は他のシンボリックパスと区別する必要があります。

オブジェクト名に関する関数

これらの関数は、プログラムでオブジェクト名を生成するのに便利です：CreateDataObjectName、CheckName、CleanupName、UniqueName。

長いオブジェクト名

Igor Pro 8.0 より前では、名前は 31 バイトに制限されていました。
現在は、名前は最大 255 バイトの長さまで指定できます。

Igor Pro 8.0 以降では、以下の種類のオブジェクトが長い名前をサポートします：

- データフォルダー
- ウェーブ
- 変数
- ウィンドウ
- 軸
- 注釈
- コントロール
- フォーマットされたドキュメントの特殊文字
- シンボリックパス
- XOP、外部コマンド、外部関数

さらに、Igor 8 以降では、ウェーブの次元ラベルとプロシージャ名は最大 255 バイトの長さまで指定可能です。

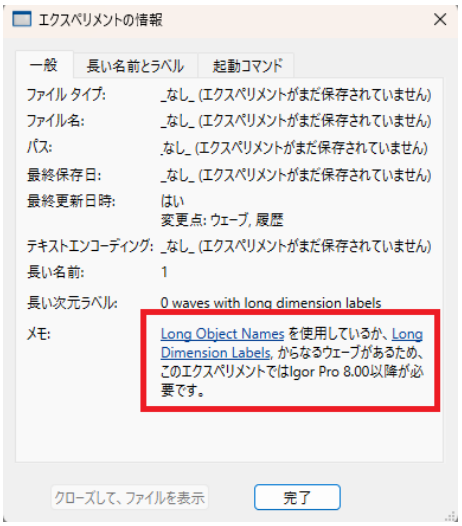
以下の種類のオブジェクトは、依然として 31 バイト以下の名前に制限されています：

- フォーマットされたノートブック内のルーラー
- グローバルな画像名
- ページ設定の名前
- FIFO
- FIFO チャンネル

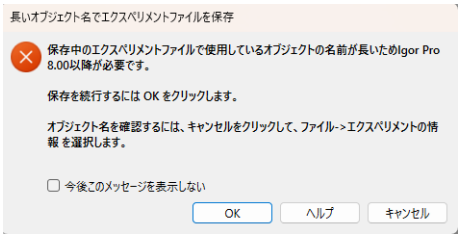
オブジェクト名を 31 バイトより長く設定しない場合は、作成したウェーブファイルおよびエクスペリメントファイルは、以前のバージョンの Igor と互換性があります。
ただし、長い名前を持つオブジェクトを作成した場合、古いバージョンの Igor では、長い名前を含むウェーブファイルやエクスペリメントファイルを開こうとするとエラーが発生します。

注記：
長い名前を使う場合、ウェーブファイルとエクスペリメントファイルは Igor Pro 8.0 以降が必要となり、それ以前のバージョンの Igor で開くとエラーが発生します。

「ファイル（File）」→「エクスペリメントの情報（Experiment Info）」を選択すると、現在のエクスペリメントが長いオブジェクト名を使っているか、長い次元ラベルを持つウェーブを含んでいるかを確認できます。
ExperimentInfo コマンドをプログラムから使うこともできます。
これらは、ウェーブ、変数、データフォルダー、ターゲットウィンドウ、シンボリックパス名、およびウェーブ次元ラベルのみをチェックします。
軸、注釈、コントロール、プロシージャ、その他の名前は確認しません。



長いウェーブ名、変数名、データフォルダー名、ターゲットウィンドウ名、シンボリックパス名を使っているエクスペリメントファイル、または長い次元ラベルを持つウェーブを含むエクスペリメントファイルを保存しようとすると、Igor はエクスペリメントダイアログを表示し、その実験には Igor Pro 8.0 以降が必要であることを通知します。



警告ダイアログは、エクスペリメントを対話的に保存する場合にのみ表示され、SaveExperiment をってプログラム的に保存する場合は表示されません。
「今後このメッセージを表示しない（Do not show this message again）」チェックボックスをクリックすると、ダイアログを非表示にできます。

グローバルな画像名（ヘルプ The Picture Gallery を参照）は 31 バイトに制限されますが、プロシージャ画像（ヘルプ Proc Pictures を参照）の名前には制限がありません。

ページ設定の名前は、各ページレイアウトウィンドウのページ設定レコードを保存するために、内部で使われます。
エクスペリメントファイル形式では、ページ設定レコードの名前は 31 バイトに制限されます。
レイアウトウィンドウ名が 31 バイトを超える場合、エクスペリメントを保存すると、そのウィンドウのページ設定レコードはエクスペリメントファイルに書き込まれません。
エクスペリメントを再開すると、レイアウトウィンドウにはデフォルトのページ設定が適用されます。
長いページレイアウト名は稀であり、ページ設定は印刷には影響しますがページの大きさには影響しないため（ヘルプ Page Layout Page Sizes を参照）、この問題はほとんど影響を与えません。

XOP 名は、拡張子「.xop」を除いた XOP ファイルの名前です。
Igor 8 以降では、XOP 名は最大 255 バイトまで指定可能です。
ただし、XOP 名が 31 バイトを超える場合、Igor は XOP に SAVESETTINGS メッセージを送信しません。
ほとんどの XOP 名は 31 バイト未満であり、またほとんどの XOP はエクスペリメント設定を保存しないため、このことが問題を引き起こす可能性は低いと考えられます。

古い Igor バージョンでの長いオブジェクト名

長い名前を使う場合、ウェーブファイルとエクスペリメントファイルは Igor Pro 8.0 以降が必要となり、それ以前のバージョンの Igor で開くとエラーが発生します。

現在のエクスペリメントが長いオブジェクト名を使っているか、長い次元ラベルを持つウェーブを含むかを確認するには、「ファイル（File）」→「エクスペリメントの情報（Experiment Info）」を選択します。
また、プログラムから ExperimentInfo コマンドを使うことも可能です。

Igor Pro 7.xx（xx は 01 以降）を実行中に、長いウェーブ、変数、データフォルダー、ウィンドウ、またはシンボリックパス名を使うエクスペリメントファイルを開くと、古いバージョンの Igor はエラーダイアログを表示し、そのエクスペリメントには Igor Pro 8.0 以降が必要であると表示します。
より新しいバージョンの Igor が必要であることを通知するこの仕組みは、長いウェーブ、変数、データフォルダー、ウィンドウ、およびシンボリックパス名に対してのみ機能します。
軸、注釈、コントロール、特殊文字、手順、または XOP 名には機能しません。
これらのオブジェクトタイプについては、古いバージョンの Igor で操作中に長い名前に初めて遭遇した時点でエラーが発生します。

Igor Pro 7.0 以前のバージョンで、長い名前を使っているエクスペリメントファイルを開こうとすると、「名前が長すぎる（name too long）」「互換性のない Igor バイナリバージョン（incompatible Igor binary version）」などのエラーが発生します。

長いオブジェクト名でのプログラミング

コードを Igor 7 以前で実行する必要がある場合、最善の対策は長いオブジェクト名の使用を避けることです。
長いオブジェクト名を条件付きでサポートしようとすると、コードが複雑で脆弱になります。
より良い方法は、Igor 7 のコードを凍結し、新機能は Igor 8 以降のブランチに追加することです。

条件付きプログラミングを使う必要がある場合、実行中の Igor のバージョンが長いオブジェクト名をサポートしているかどうかを次のようにテストできます：

```
Variable maxObjectNameLength = 31
if (IgorVersion() >= 8.00)
	maxObjectNameLength = 255
endif
```

関数、定数、変数、その他のプログラミング要素の名前は最大 255 バイトまで指定できますが、31 バイトを超える名前を使う場合、そのプロシージャは Igor Pro 8.0 以降が必要です。

パッケージ名は最大 255 バイトまで指定できますが、31 バイトを超える名前を使う場合、そのパッケージは Igor Pro 8.0 以降が必要です。

長いオブジェクト名と XOP

パッケージ名は最大 255 バイトまで指定できますが、31 バイトを超える名前を使う場合、そのパッケージは Igor Pro 8.0 以降が必要です。
ただし、既存の XOP に対して長いオブジェクト名を使用しようとすると、エラーが発生します。
例えば：

```
NewPath SymbolicPathWithANameThatExceeds31Bytes, <path>
OldFileLoader /P=SymbolicPathWithANameThatExceeds31Bytes <path>
```

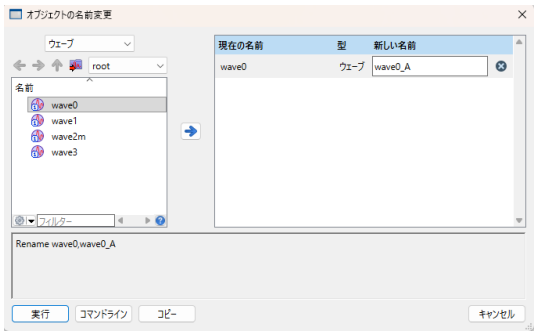
この処理はエラーを返します。
なぜなら、仮想的な OldFileLoader XOP は長いオブジェクト名に対応するよう更新されていないためです。
短いオブジェクト名での動作は引き続き可能です。

既存の XOP が長い名前のウェーブまたはデータフォルダの名前を取得しようとすると、Igor は XOP にエラーを返します。
XOP が適切に記述されている場合、「名前が長すぎます（name too long）」エラーを生成します。

XOP Toolkit 8 以降、XOP Toolkit は長いオブジェクト名に対応した XOP の作成をサポートしています。
長いオブジェクト名をサポートするには、XOP プログラマーが XOP を修正し再コンパイルする必要があります。
長いオブジェクト名をサポートするようにコンパイルされた XOP は、Igor Pro 8.0 以降が必要です。
したがって、多くの XOP はこれをサポートしていません。

現時点では、XOP 自体の名前（拡張子「.xop」を除く）が 31 バイトを超える場合、その XOP は SAVESETTINGS メッセージによる設定の保存や LOADSETTINGS メッセージによる設定の読み込みが行えません。
したがって、XOP の名前が 31 バイトを超えることは避けるのが最善です。

オブジェクト名の変更



ウェーブ、変数、文字列、シンボリックパス、画像の名前を変更するには、「その他（Misc）」→「オブジェクトの名前変更（Rename Objects）」または「データ（Data）」→「名前変更（Data→Rename）」を使用できます。
どちらの方法でも「オブジェクト名変更（Rename Objects）」ダイアログが表示されます。

グラフ、テーブル、ページレイアウト、ノートブック、コントロールパネル、Gizmo ウィンドウ、XOP ターゲットウィンドウは、ウィンドウコントロールダイアログ（「ウィンドウコントロールダイアログ」のセクションを参照）を使って作成できる DoWindow コマンドによって名前が変更されます。

データブラウザーを使って、データフォルダー、ウェーブ、変数の名前を変更できます。

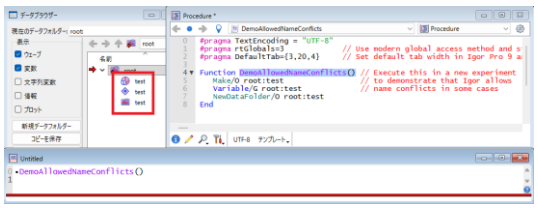
オブジェクト名の衝突

一般的に、Igor では同じ名前空間内の 2 つのオブジェクトが同じ名前を持つことを許可しません（Igor の名前空間一覧については「名前空間」のセクションを参照してください）。
このセクションでは、この規則の例外と関連する問題について説明します。
ほとんどのユーザーはこの内容を理解する必要はありません。

Igor が許可するオブジェクト名の衝突

Igor では、ウェーブ、変数、データフォルダー間で名前の衝突が発生します。
ただし、下記の特定の例外を除きます。
例えば：

```
Function DemoAllowedNameConflicts()          // これを新しいエクスペリメントで実行し、
   Make/O root:test                          // 場合によっては名前衝突を許容することを
   Variable/G root:test                      // 確かめてください。
   NewDataFolder/O root:test
End
```



これは、Igor が同じ名前のウェーブとグローバル変数を作成できること、またウェーブやグローバル変数と同じ名前のデータフォルダを作成できることを示しています。

ウェーブとグローバル変数が同じ名前を持つことが許可されてしまうのは、ウェーブと変数が同じ名前空間（いわゆる「メイン名前空間」）にあるため、本来は許されるべきではなかった歴史的な偶然です。
これは将来バージョンの Igor では許可されなくなる可能性があります。

データフォルダーがウェーブや変数と同じ名前を持つことを許可するのは、データフォルダーが独自の名前空間にあるための意図的な仕様です。

以下の例外は、メインの名前空間において許可されない名前衝突を示します：

1. マクロ、プロシージャ、またはコマンドラインから既存のウェーブと同じ名前の変数を作成することはエラーとなります（ただし、上記のようにユーザー定義関数から作成する場合はエラーになりません）。

2. 既存の変数（関数、マクロ、プロシージャ、またはコマンドラインからのもの）と同じ名前でウェーブを作成することは誤りです。

以下の表はそれらの規則を示しています。
左の列は最初に作成されるオブジェクトのタイプを示し、上段は次に作成されるオブジェクトのタイプを示します。

F は、関数内でオブジェクトを作成しようとすると、エラーを返すことを意味します。

M は、マクロ、プロシージャ、またはコマンドラインからオブジェクトを作成しようとすると、エラーを返すことを意味します。

--- は、関数、マクロ、プロシージャ、またはコマンドラインから、エラーを返さないことを意味します。

	ウェーブ	**変数**	**データフォルダー**
ウェーブ	N/A	M	---
変数	F, M	N/A	---
データフォルダー	---	---	N/A

オブジェクト名の衝突と HDF5 ファイル

エクスペリメントをパック形式またはアンパック形式で保存する場合、これらの名前衝突は問題を引き起こしません。
エクスペリメントを保存して再読み込みすれば、元の状態に戻ります。

ただし、名前衝突は HDF5 ファイルで問題を引き起こします。
これは「エクスペリメントを保存（Save Experiment）」または HDF5SaveGroup コマンドで保存されたファイルにおいて発生します。
HDF5 ではグループとデータセットが同じ名前を持つことを許可していないためです。

- ウェーブまたは変数とデータフォルダーとの間に衝突がある場合、HDF5 形式で保存しようとするとエラーが発生します。

- ウェーブと変数の間で衝突が発生した場合、上書きがオフであればエラーが発生します。
  上書きがオンであれば、変数のデータセットがウェーブのデータセットを上書きするため、ウェーブは結果の HDF5 ファイルに保存されません。
  Igor Pro 9.00 では、HDF5 パック形式のエクスペリメントファイルを書き込む時、Igor は上書きを有効にしていました。
  Igor Pro 9.01 では、この状況でエラーが発生するよう、上書きを無効に変更しました。

概念的には、ファイル保存時に衝突するオブジェクトのいずれかの名前を変更し、この変更が行われた事実を何らかの方法で記録し、ファイル読み込み時にこのプロセスを逆転させることで、これらの衝突を処理することは可能です。
このスキームを実装しようとしたところ、既に複雑なコードにさらなる複雑さが加わることが判明しました。
複雑さを加えることは、新たなバグを生み出す可能性や動作を遅くする可能性をもたらします。
これらの名前の衝突は稀であるため、このような回避策を実装するデメリットがメリットを上回ったことから、
HDF5 ファイルにおける名前衝突のサポートは見送ることにしました。

メモリの管理

理論上、Igor は約 10 億ギガバイトをアドレス指定できます。
ただし、実際のオペレーティングシステムでははるかに低い制限が課されます。

物理メモリに収まる以上のデータをロードすると、システムは「仮想メモリ」の使用を開始します。
これは、必要に応じて物理メモリとディスク間でデータをスワップすることを意味します。
これは非常に遅くなります。
したがって、物理メモリに収まる以上のデータをメモリにロードすることは避けるべきです。

たとえデータが物理メモリに収まる場合でも、1000 万、1 億、あるいは 10 億ポイントといった非常に大規模なウェーブのグラフ化や操作は遅くなります。

これらすべては、以下のルールに集約されます：

1. 一度にギガバイト単位のデータをメモリに読み込む必要がなければ、メモリ管理について心配する必要はありません。

2. 仮想メモリのスワッピングを回避するために、十分な物理メモリをインストールしてください。

非常に大きなウェーブに関する詳細情報は、ヘルプ IGOR64 Experiment Files を参照してください。

Igor のエクステンション

Igor には、C または C++ プログラマーがその機能を拡張できる機能が含まれています。
Igor の拡張機能は「XOP」（external operation/外部コマンドの略称）と呼ばれます。
XOP という用語は、当初、拡張機能が実行できるのはコマンドライン操作の追加だけだったという事実から由来しています。
現在では、拡張機能はコマンド、関数、メニュー、ダイアログ、ウィンドウを追加できます。

WaveMetrics の XOP

「Igor Pro Folder/Igor Extensions (64-bit)」および「Igor Pro Folder/More Extensions (64-bit)」フォルダには、WaveMetrics で開発した XOP が含まれています。
これらは Igor にファイル読み込みや機器コントロールなどの機能を追加するだけでなく、XOP が実現可能な機能の例としても役立ちます。
これらの XOP は非常に単純なものからかなり複雑なものまで様々です。
ほとんどの XOP には、その動作を説明するヘルプファイルが付属しています。

WaveMetrics XOP は、「ヘルプ（Help）」→「ヘルプウィンドウ（Help Windows）」サブメニューからアクセス可能な「XOP Index」に記載されています。

サードパーティの XOP

多くの Igor ユーザーが、それぞれの専門分野に合わせて Igor をカスタマイズする XOP を作成しています。
これらのうち、フリーウェアのものもあれば、シェアウェアのもの、商用プログラムのものもあります。
WaveMetrics は、当社の Web ページを通じてサードパーティ製 XOP を公開しています。
ユーザー開発の XOP は http://www.igorexchange.com から入手可能です。

独自の XOP を作成したいプログラマーの方は、ヘルプ Creating Igor Extensions を参照してください。

64 ビットエクステンションの有効化

64 ビット拡張機能を有効にするには、その拡張機能自体、またはその拡張機能へのショートカットを、「Igor Pro User Files」フォルダー内の「Igor Extensions (64-bit)」フォルダーに配置します。

説明のため、データベースへのアクセスを提供する Igor の SQL XOP を有効化するために必要な手順を以下に示します。

1. 「ヘルプ（Help）」→「Igor Pro User File を表示（Show Igor Pro User Files）」を選択すると、デスクトップ上の「Igor Pro User Files」フォルダーが開きます。
2. 「ヘルプ」→「Igor Pro Folder を表示（Show Igor Pro Folder）」を選択すると、デスクトップ上の「Igor Pro Folder」が開きます。
3. デスクトップ上の「Igor Pro Folder\More Extensions (64-bit) \Utilities」フォルダーを開きます。
4. 「SQL64.xop」のショートカットを作成し、それを「Igor Pro User Files」フォルダー内の「Igor Extensions (64-bit)」フォルダーに配置します。
5. Igor64 を再起動します。

特定のマシン上の全ユーザー向けに XOP を有効化したい場合、Igor Pro Folder 内の「Igor Extensions (64-bit)」フォルダーにショートカットを配置してください。
この操作には管理者権限で実行する必要がある場合があります。

「Igor Extensions (64-bit)」フォルダーへの変更は、次回 Igor を起動した時に反映されます。

32 ビットエクステンションの有効化

32 ビット版の Igor を実行している場合、前のセクションの手順に従い、若干の変更を加えることで、32 ビット版の XOP を有効化できます。

「Igor Extensions (64-bit)」の代わりに「Igor Extensions」を、「More Extensions (64-bit)」の代わりに「More Extensions」を使ってください。
SQL64.xop の代わりに、SQL.xop などの 32 ビット版の XOP をアクティブにします。

「その他の設定」ダイアログ

「その他設定（Miscellaneous Settings）」ダイアログを使って、Igor の動作に関する多くの側面をカスタマイズできます。ダイアログを表示するには、「その他（Misc）」→「その他の設定（Miscellaneous Settings）」を選択してください。

ほとんどの設定は一目瞭然です。
多くの設定には、その機能を説明するツールチップが付いています。

プレファレンス

プレファレンスは、新しいグラフ、パネル、テーブル、レイアウト、ノートブック、および手順ウィンドウの作成、ならびにグラフへのトレースの追加やテーブルへの列の追加に影響します。
さらに、プレファレンスはコマンドウィンドウとグラフのデフォルトフォントにも影響します。

プレファレンスは「その他（Misc）」メニューからオン/オフを切り替えられます。
通常は設定をオンにして実行します。

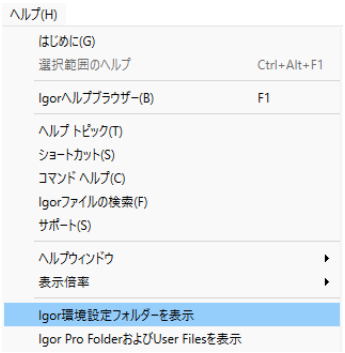
手順の実行中はプレファレンスが自動的に無効化され、手順の影響が全ユーザーで均一になるようにします。
詳細はヘルプ Preferences コマンドを参照してください。

プレファレンスがオフの場合、グラフのサイズ、位置、線のスタイル、サイズ、色などの設定には製品出荷時のデフォルト値が使われます。
プレファレンスがオンの場合、Igor はこれらの設定に対してユーザーが指定した値を適用します。

プレファレンスと組み込み設定の違いは、組み込み設定は通常即時有効となるのに対し、プレファレンスは何かが作成される際に使われる点にあります。

プレファレンスのディレクトリ

Igor のプレファレンスはユーザーごとのディレクトリに保存されます。
このディレクトリの場所は、オペレーティングシステムと設定によって異なります。
通常、プレファレンスのディレクトリにアクセスする必要はありませんが、Shift キーを押しながら操作すると、「ヘルプ（Help）」→「Igor 環境設定フォルダーを表示（Show Igor Preferences Folder）」でデスクトップ上で開くことができます。



Igor のプレファレンスディレクトリの場所を確認するには、次のコマンドを実行することもできます：

```
Print SpecialDirPath("Preferences", 0, 0, 0)
```

技術的な理由により、組み込み設定の大部分はプレファレンスディレクトリではなく、その一つ上の階層にある「Igor Pro 10.ini」というファイルに保存されます。

プレファレンスディレクトリと「Igor Pro 10.ini」ファイルを削除すると、すべての設定が製品出荷時のデフォルト状態に復元されます。
プレファレンスをより選択的に復元するには、「キャプチャされたプレファレンス」セクションで説明している「キャプチャ設定（Capture Prefs）」ダイアログを使ってください。

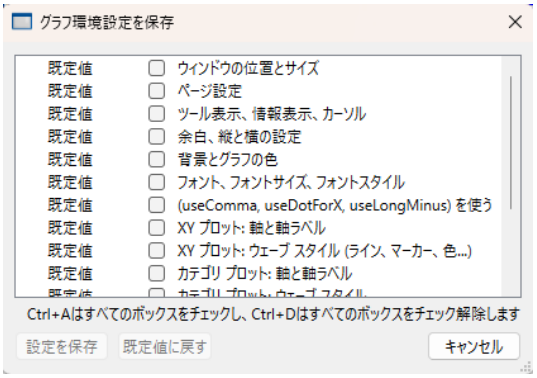
その他の情報は設定に保存されます。
これにはダイアログの画面位置、いくつかのダイアログ設定、カラーパレットで最近選択した色、ウィンドウのスタック順とタイル配置情報、ページ設定、フォント置換の設定、破線の設定などが含まれます。

プレファレンスの使い方

Igor 起動時には常にプレファレンスが有効です。
プレファレンスを無効にするには、「その他」メニューから「環境設定オフ（Preferences Off）」を選択してください。

Preferences コマンドで設定を有効化または無効化することもできます。

プレファレンスは、「環境設定を保存（Capture Preferences）」ダイアログ、タイルまたはスタックウィンドウダイアログ、およびフォント置換や破線ダイアログなどの一部のダイアログによって行われます。



一般的に、プレファレンスは新しいグラフ、グラフ内の新しいトレース、新しいノートブック、テーブル内の新しい列など、何か新しいものが作成される場合にのみ適用され、しかもプレファレンスが有効になっている場合に限られます。

プレファレンスは通常、手動（ポイント・アンド・クリック）操作でのみ有効であり、プロシージャ内のユーザープログラム操作では有効ではありません。
ヘルプ Procedures and Preferences を参照してください。

環境設定を保存

ほとんどのプレファレンスの値は、アクティブなウィンドウのメニューにある「環境設定を保存（Capture Prefs）」で、そのウィンドウの現在の設定をキャプチャすることで設定します。
（コマンドウィンドウの環境設定をキャプチャするダイアログは、「その他」メニューの「コマンドバッファ/設定を保存」サブメニューにあります。）
各ダイアログの詳細については、各ウィンドウの種類を説明するヘルプファイルに記載されています。
例えば、ヘルプ Graph Preferences を参照してください。