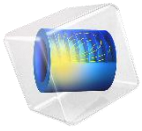


INTRODUCTION TO

Application Builder

日本語訳版

5.3a

COMSOL

Introduction to Application Builder 日本語訳版

バージョン: COMSOL 5.3a

日本語訳:計測エンジニアリングシステム株式会社

※もし日本語訳と原文(英語版)に差違がある場合は、原文を正とします。

お問い合わせ

一般的なお問い合わせ、技術サポートのお問い合わせ、住所および電話番号の検索については、www.comsol.com/contact にアクセスしてください。
住所および連絡先情報については、www.comsol.com/contact/offices でも確認できます。

サポート窓口へのお問い合わせについては、www.comsol.com/support/case からリクエストフォームを送信してください。

その他の有用なリンク:

- サポートセンター: www.comsol.com/support
- 製品のダウンロード: www.comsol.com/product-download
- 製品のアップデート: www.comsol.com/support/updates
- COMSOL ブログ: www.comsol.com/blogs
- フォーラム: www.comsol.com/community
- イベント: www.comsol.com/events
- COMSOL ビデオギャラリー: www.comsol.com/video
- サポートナレッジベース: www.comsol.com/support/knowledgebase

部品番号: CM020011

目次

まえがき

シミュレーションパッケージの典型的なユーザは、博士号や修士課程を持ち、モデリングとシミュレーションを数年経験し、特定のパッケージを使用するために徹底した訓練を受けた人達です。そのような人は、一般的に大きな組織の R&D 部門の科学者として、または学術研究者として働いています。シミュレーションの理論は複雑であり、典型的なシミュレーションパッケージには多くのオプションが提示されているので、モデルとシミュレーションを検証するためにその人の専門知識が採用できるかどうかはユーザ次第です。

これは、シミュレーションの専門家の小グループが、製品開発で、生産で、あるいは物理効果を学ぶ学生として働く人達の非常に大きなグループにサービスを提供していることを意味します。多くの場合、シミュレーションモデルは複雑であるため、安全に有用な出力を得るために入力データを提供することができるのはモデルを実装する人だけです。したがって、コンピュータモデリングとシミュレーションの利用では、製品開発、生産、および教育におけるボトルネックが生まれます。

アプリケーションビルダーは、この小さなグループが非常に大きいグループにサービスを提供することができるようにするためのソリューションを提供しています。これによって、シミュレーションの専門家が、その人達のための、あるいは一般的なコンピュータモデル – すぐに使用できるアプリケーション – のための直感的で非常に特定的なユーザインタフェースを作成することができるようになります。一般的な場合、目前の特定の作業にだけ関連する入力と出力のオプションをユーザに提示するアプリケーションを、複数のさまざまなアプリケーションの出発点として役立てることができます。アプリケーションには、ユーザマニュアル、「入力値が設定範囲内か」のチェック、ボタンクリックによる既定のレポートを含めるといったことができます。

多くの場合、アプリケーションを作成するためには、物理学、数値解析、プログラミング、ユーザインタフェース設計、およびグラフィックデザインの領域の専門家による共同の努力が必要です。

COMSOL のテクニカルサポートチームは、合理的な範囲で、アプリケーションのための物理学や数値解析の設定について助言することができます。また、COMSOL のマニュアルとオンラインリソースは大きな手助けとなります。プログラミングとデザインのための非常に限定されたヘルプがテクニカルサポートチームによって提供されており、利用することができます。その領域では、独自の開発努力が重要です。

アプリケーションビルダーによって、関係するアウトプットの詳細への焦点を保ちながら偶発的なユーザの入力エラーを避けるように、良く練られたアプリケーションをチームが容易に作成することができるようになります。

我々は、これが COMSOL でこの世界におけるシミュレーション利用の成功を広めるための方法であると確信しており、これを可能にする手助けとなることをお約束します。

はじめに

COMSOL® アプリケーションは、非常に特異なユーザインタフェースを通じて COMSOL Multiphysics® モデルと直観的で効率的なやりとりができる方法です。本書では、フォームエディターとメソッドエディターの使い方の具体例を掲載し、アプリケーションビルダーのデスクトップ環境における概要がすぐに把握できるように示されています。本書には、利用可能な組み込みメソッドと機能のリストを含む参考資料が掲載されています。モデルビルダーの使い方に関する詳細については、*Introduction to COMSOL Multiphysics* を参照してください。

本書を読む前にアプリケーション例を確認したい場合には、アプリケーションライブラリにある Applications という名前のフォルダの中から一つのアプリケーションを開いて詳しく調べてみてください。開いたままの状態で、本書を読みながらいろいろなことを試してみてください。Applications という名前のフォルダは、ユーザインタフェースを伴う（アプリケーションビルダーが作成されている）アプリケーションだけを含んでいます。アプリケーションライブラリにあるその他のフォルダは、ユーザインタフェースを持たないチュートリアルモデルです。

アプリケーションビルダーは、COMSOL マルチフィジックスの Windows® 版に搭載されており、COMSOL Desktop® 環境でアクセス可能です。
アプリケーションの作成には、COMSOL マルチフィジックスに加えてアドオン製品も使用されます。COMSOL マルチフィジックス、または COMSOL Server™ でアプリケーションを実行する際にも、これと同じアドオン製品のライセンスが必要です。
動画のチュートリアルなどの追加情報は、www.comsol.com からオンラインで入手することができます。

COMSOL マルチフィジックスでのアプリケーションの実行

COMSOL マルチフィジックスのライセンスによって、Windows®、OS X 、および Linux® での COMSOL デスクトップからアプリケーションを実行することができます。

COMSOL サーバーでのアプリケーションの実行

COMSOL サーバーのライセンスによって、Windows®、OS X、iOS、Linux®、Android™ などのプラットフォームの主要なウェブブラウザでアプリケーションをウェブ実行することができます。また、Windows® において、COMSOL クライアントから COMSOL サーバーへ接続して、COMSOL アプリケーションを実行することもできます。この COMSOL クライアントは、容易に www.comsol.com からダウンロードしてインストールすることができます。COMSOL サーバーでは、COMSOL デスクトップ環境に搭載されているアプリケーションビルダー、フィジックスビルダー、およびモデルビルダーといったツールを利用することはできません。

アプリケーション構築のためのガイドライン

グラフィカルユーザインタフェースの構築やプログラミングに慣れていない場合は、319 ページの「付録 F—アプリケーション構築のためのガイドライン」をお読みください。

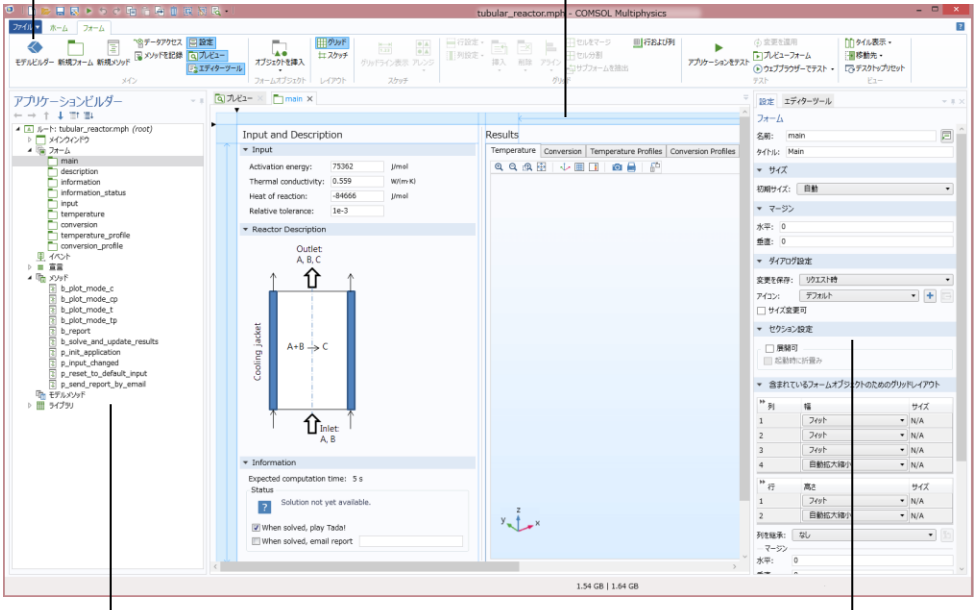
追加ドキュメント

アプリケーション構築に関する情報を含む追加のドキュメントは、*Application Programming Guide*、*Application Builder Reference Manual*、および *Programming Reference Manual* のドキュメント内の掲載をご覧ください。

アプリケーションビルダーのデスクトップ環境

モデルビルダーとアプリケーションビルダー—
このボタンをクリックすることによって、モデルビルダーとアプリケーションビルダーを切り換えることができます。

COMSOL デスクトップ環境—
COMSOL デスクトップ環境は、モデルビルダーと同様に、フォームとメソッドエディターを含むアプリケーションビルダーへのアクセスを提供してくれます。



アプリケーションビルダーウィンドウ—
アプリケーションビルダーウィンドウには、アプリケーションツリーが表示されています。

設定ウィンドウ—
アプリケーションツリーにおいて、フォームオブジェクトやメソッドのノードをクリックすると、その設定ウィンドウが表示されます。

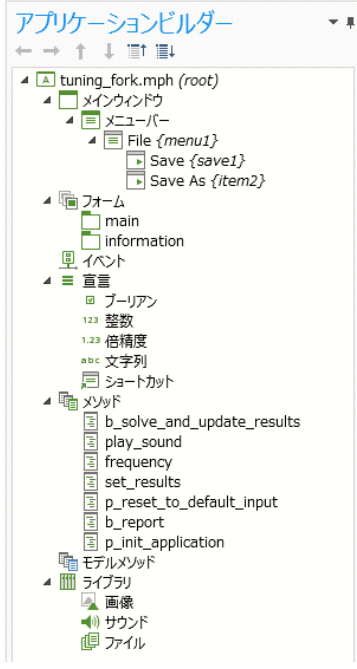
上図のスクリーンショットは、アプリケーションビルダーの作業中の典型的な画面構成です。以下に、アプリケーションビルダーのデスクトップ環境における主要な構成を示します。

- アプリケーションビルダーウィンドウとリボンタブ
- COMSOL デスクトップ環境
- フォームエディター（40 ページ参照）
- メソッドエディター（149 ページ参照）

アプリケーションツリー

アプリケーションツリーは、以下のノードで構成されています。

- **メインウィンドウ**
- **フォーム**
- **イベント**
- **宣言**
- **メソッド**
- **モデルメソッド**
- **ライブラリ**



メインウィンドウノードはアプリケーションの主要なウィンドウであり、ユーザインタフェースのためのトップレベルのノードでもあります。そこには、ウィンドウの配置、メインメニューの設定、およびオプションのリボンの設定が含まれています。

作成されたフォームは、**フォーム**ノードの下にサブノードとして展開されています。その各々のフォームは、そこに配置された入力フィールド、グラフィックス、ボタンなどの多くのフォームオブジェクトを含んでいることになります。

グローバルイベントは、**イベント**ノードの下にサブノードとして展開されます。これらには、グローバルパラメータや文字列変数など、さまざまなデータ入力の変化によって起動されるイベントを全て含んでいます。また、グローバルイベントは、アプリケーションのスタートアップとシャットダウンに関連付けることもできます。

宣言ノードでは、グローバル変数を宣言します。モデルの中で定義されているグローバルパラメータや変数に加えて使われます。

アプリケーションメソッドは、**メソッド**ノードの下にサブノードとして展開されています。このメソッドには、モデルビルダーのモデルツリーノードの標準実行コマンドだけでは実行できない動作が、コードとして記述されます。このメソッドの例としては、ループの実行、入出力の処理、アプリケーションのユーザへのメッセージや警告の送信などがあります。アプリケーションメソッドは、実行中のアプリケーションのモデルオブジェクトを変更しますが、現行のセッションでモデルビルダーによって表示されるモデルオブジェクトを変更することはできません。

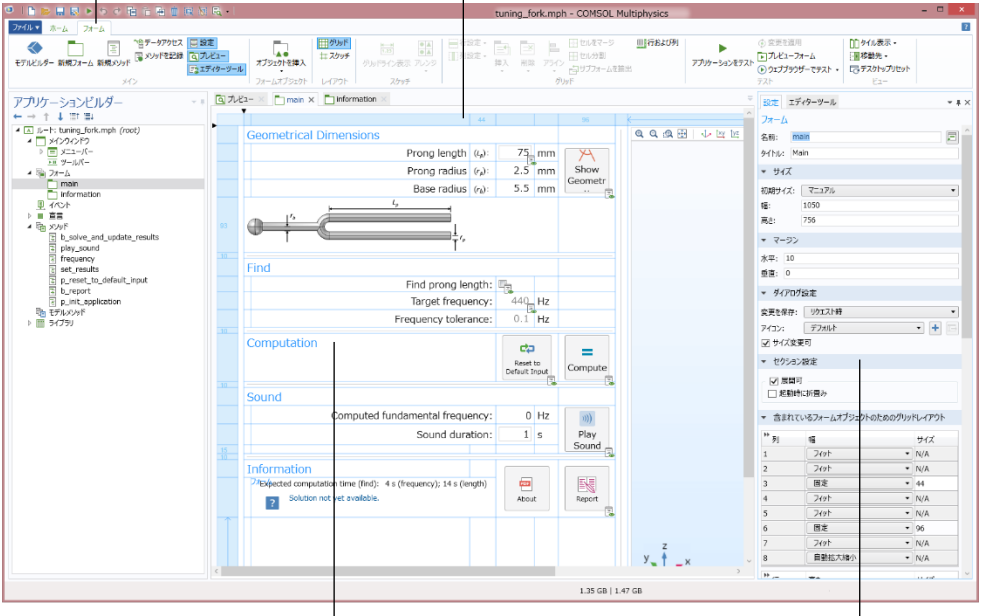
モデルメソッドノードは、メソッドノードに似ています。メソッドノードには、実行中のアプリケーションのモデルオブジェクトを変更するアプリケーションメソッドが含まれますが、モデルメソッドノードには、現行のセッションでモデルビルダーによって表示されるモデルオブジェクトを直接変更するモデルメソッドが含まれます。モデルメソッドを使用して、いくつかのマニュアルステップからなるモデリングタスクの自動化が可能です。

ライブラリノードには、MPH ファイルに組み込まれた画像、サウンド、ファイルが展開されています。これにより、それらをアプリケーションの MPH ファイル自体と別々に配布する必要がありません。さらに、**ライブラリ**ノードには、外部 Java® ライブラリと C ライブラリのノードや Java® ユーティリティクラスのノードも含めることができます。

フォームエディター

フォームタブ—
リボンのフォームタブにより、フォームエディターを容易にアクセスできます。

フォームエディターウィンドウ—
タブ化されたフォームエディターウィンドウでは、オブジェクトをドラッグして移動させることができます。そのオブジェクトの設定を編集するには、そのオブジェクトをクリックします。



フォームオブジェクト—
各々のフォームは、入力フィールド、チェックボックス、グラフィックス、画像、ボタンなどのフォームオブジェクトを含んでいます。

設定とエディターツールウィンドウ—
アプリケーションツリーノード、またはフォームオブジェクトをクリックして、その設定ウィンドウを開くことができます。エディターツールウィンドウは、フォームオブジェクトを迅速に作成するために使用されます。

以下に、フォームエディターは、入力フィールド、グラフィックス、ボタンといったフォームオブジェクトによってフォームを作成するユーザインタフェースレイアウトのために使います。フォームエディターの主要なコンポーネントを以下に示します。

- フォームリボンタブ
- アプリケーションツリーが表示されるアプリケーションビルダーウィンドウ
- フォームウィンドウ
- エディターツールウィンドウ
- 設定ウィンドウ

新規フォームの作成

新規フォームを作成するには、アプリケーションツリーの**フォーム**ノードを右クリックし、**新規フォーム**を選択します。または、リボンの**新規フォーム**をクリックすることもできます。新規フォームの作成を実行すると、自動的に**新規フォーム**ウィザードが開かれます。

すでに作成されているフォーム、例えば **form1** を編集する場合には、次のいずれかの方法でフォームエディターを開きます。

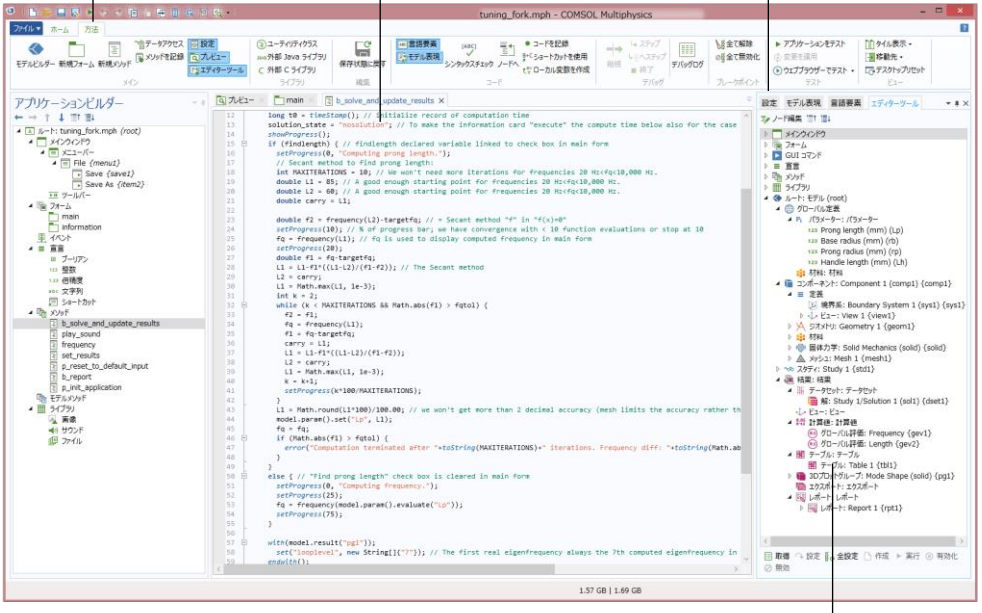
- アプリケーションツリーの中で、**form1** ノードをダブルクリックする。
- アプリケーションツリーの中で、**form1** ノードを右クリックし、**編集**を選択する。

メソッドエディター

メソッドタブ—
リボンのメソッドタブによって、コードの記述とデバッグを行うツールを容易にアクセスできます。

メソッドウィンドウ—
メソッドウィンドウにタブが付けられているため、あるメソッドウィンドウでの編集中に他のメソッドウィンドウに切り換えることができます。

設定ウィンドウ—
アプリケーションツリーノードをクリックして、その設定ウィンドウを開くことができます。



モデル表現、言語要素、およびエディターツールウィンドウ—
これらのウィンドウは、コードを記述するためのツール表示です。モデル表現のウィンドウは、モデルの中から入手可能な全ての定数、パラメータ、変数、および関数を表示してくれます。言語表現のウィンドウは、組み込みメソッド用のテンプレートコードを挿入するために使われます。エディターツールウィンドウは、モデルツリーノードを編集し、動作させるためのコードを入手するために使われます。

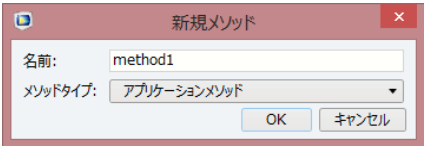
メソッドエディターは、モデルツリーノードの標準的な使用方法ではカバーできない動作のメソッドを記述するために利用します。メソッドとは、他のプログラミング言語におけるサブルーチン、機能、または手続として知られているものの別名です。

メソッドエディターの主要な構成を以下に示します。

- メソッドリボンタブ
- アプリケーションツリーが表示されるアプリケーションビルダーウィンドウ
- メソッドウィンドウ
- モデル表現、言語要素、エディターツール、および設定ウィンドウ（これらは重ねて表示され、タブで切り替えられます。）

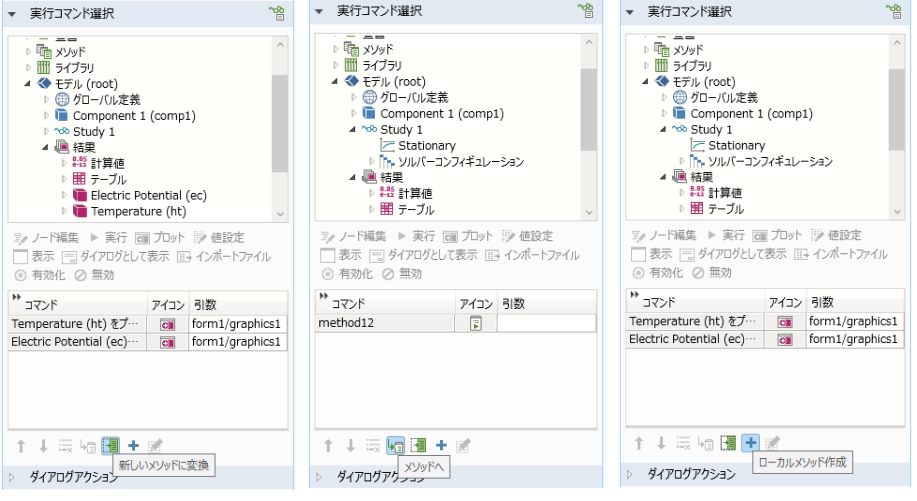
新規メソッドの作成

新規メソッドを作成するには、アプリケーションツリーの中の**メソッド**ノードを右クリックし、**新規メソッド**を選択します。リボンにある**新規メソッド**をクリックする方法もあります。**新規メソッド**ダイアログボックスの中で、**アプリケーションメソッド**か**モデルメソッド**かの**メソッドタイプ**を設定します。



新規メソッドが作られると同時に、メソッドエディターが開かれます。この方法で作られるメソッドはグローバルメソッドであり、全てのメソッドやフォームオブジェクトからアクセスすることが可能です。モデルメソッドはグローバルメソッドであり、全ての他のメソッドやモデルビルダーのリボンにあるデベロッパータブからアクセスすることが可能です。

新しいメソッドに変換をクリックすると、実行コマンド選択セクションのツリーから設定されたコマンドシーケンスが自動的に一つのメソッドに変換されます。**メソッドへ**をクリックすると、新規に(グローバルな)メソッドが開かれます。**ローカルメソッド作成**をクリックすると、そのフォームオブジェクトに固有のローカルなメソッドが作成されます。これらの選択肢を下図に示しています。



すでに method1 という名前のメソッドが生成されている場合は、例えば、以下の方法でメソッドエディターを開きます。

- アプリケーションツリーにおける **method1** ノードをダブルクリックする。

- アプリケーションツリーにおける **method1** ノードを右クリックし、**編集**を選択する。
- フォームオブジェクトまたはイベントの**設定**ウィンドウのコマンドシーケンスの下側にある**メソッドへ**をクリックします。

アプリケーションの環境設定

アプリケーションビルダーの**環境設定**にアクセスするには、**ファイル**メニューから**環境設定**を選択し、**アプリケーションビルダー**のページを選択します。

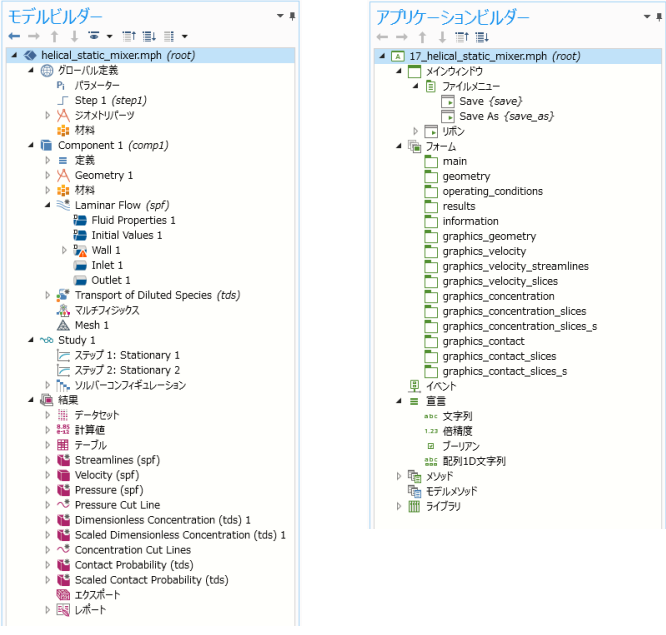
アプリケーションビルダーが別のデスクトップウィンドウで表示されるように、COMSOL デスクトップ環境を設定することができます。**アプリケーションビルダー用に別のデスクトップウィンドウを使用**のチェックボックスを選択します。

モデルビルダーとアプリケーションビルダーを切り換えるには、それぞれ、Ctrl + Alt+ M と Ctrl+ Alt + A のキーボードショートカット操作を使用することができます。

フォームエディターやメソッドエディターウィンドウを開くタブの数に上限を設定することができます。**終了前のエディターの最大数**のチェックボックスを選択し、その数値（デフォルト 15）を編集します。これを小さな数にすることによって、多くのフォームを含んだアプリケーションを読み込む場合に高速化することができます。

アプリケーションビルダーとモデルビルダー

アプリケーションビルダーは、モデルビルダーで作成されたモデルに基づいたアプリケーションを作成するためのものです。アプリケーションビルダーには、アプリケーションを作成するために重要なツールが二つあります。その一つであるフォームエディターには、入力フィールド、グラフィックスオブジェクト、ボタンといったインタフェースコンポーネントを使用するためのドラッグ＆ドロップ機能があります。もう一つのメソッドエディターは、モデルのいろいろな箇所のデータ構造を変更することができるプログラミング環境です。下図に、モデルビルダーウインドウとアプリケーションビルダーウインドウを示します。



一般に、アプリケーションを作成する時には既存のモデルから始めます。しかし、アプリケーションユーザインタフェースと元となるモデルを同時に構築することができれば、それに越したことはありません。モデルビルダーとアプリケーションビルダーは、いつでも容易に切り替えることができます。アプリケーションにおいてモデルツリーで表されているモデルの部分を、埋め込みモデルと呼ぶことにします。

アプリケーションビルダーのツールによって、いくつかの方法で埋め込みモデルの設定をアクセスし、処理することができます。その例を以下に示します。

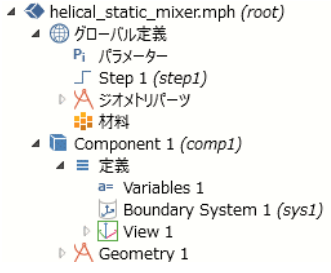
- モデルがパラメータと変数を利用している場合、新規フォームウィザードやエディターツールを使って、アプリケーションの中の入力フィールドに直接これらをリンクさせることができます。このように、アプリケーションのユーザは、モデルに影響するパラメータと変数の値を直接編集することができます。詳細については、49 および 79 ページを参照してください。
- 新規フォームウィザードやエディターツールを使って、スタディ（Study）ノードを実行してソルバーを開始するためのボタンを実装することができます。さらには、このウィザードを利用してグラフィック

ス、数値出力、チェックボックス、およびコンボボックスを挿入することもできます。詳細については、35 および 49 ページを参照してください。

- フォームオブジェクトやメソッドにおいてモデル内の低レベルの設定に直接アクセスするには、データアクセスツールとエディターツールウィンドウを利用することができます。詳細については、49、89、および 154 ページを参照してください。
- コード記録ツールを使って、モデルツリーとそのノード内で操作して実行した内容をコマンドとして記録することができます。その後、このコードをさらに編集するという方法をとることができます。詳細については、159 ページを参照してください。

パラメータ、変数、スコープ

モデルツリーには、モデルの設定をコントロールするためのパラメータと変数の両方が含まれている場合があります。下図は、**パラメータ**と**変数**の両方のノードが含まれるアプリケーションのモデルツリーを示しています。



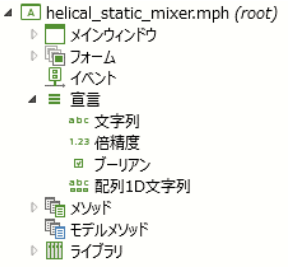
パラメータはモデルツリーの**グローバル定義**ノードの下で定義され、モデルビルダー全体で使用可能なユーザ定義のスカラー定数です。すなわち、それらは本質的に「グローバル」です。重要な用途としては、以下があります。

- ジオメトリの寸法のパラメータ化
- メッシュ要素サイズの指定
- パラメータスイープの定義

変数は、**グローバル定義**ノード、またはそのモデルの**コンポーネント**(Component)ノードの下の**定義**サブノードにおいて定義することができます。モデルのコンポーネントに定義された変数はそのコンポーネントの中でしか使うことができないのに対し、グローバルに定義された変数はモデル全体で使うことができます。変数は、解析のための従属フィールド変数を含む、空間的または時間的変化の式に使われます。

モデルビルダーでは、パラメータまたは変数は、その値が有効なモデル表現であるという付加的な制約条件を伴った文字列です。モデルのパラメータと変数の使用方法の詳細については、*Introduction to COMSOL Multiphysics* を参照してください。

アプリケーションでは、フォームエディターとメソッドエディターで使用する変数を追加する必要が生じる場合があります。そのような変数は、アプリケーションツリーの**宣言**ノードの下で宣言されます。下図は、複数の宣言を伴ったアプリケーションツリーを示します。



アプリケーションビルダーで宣言される変数は、スカラー、配列、ブーリアン、文字列、整数、および倍精度実数といった型の変数です。変数を使う前に、その型を宣言する必要があります。

変数の型が指定されているため、最初に組み込みメソッドを使って型変換するというような必要がなく、これらの変数を直接メソッドの中で使うことができます。モデルの式に使われているパラメータと変数によるよりも、この型宣言された変数を利用した方がメソッドのコードの記述が容易です。一方で、アプリケーションビルダーには、変数の種類を変換するためのいくつかのツールがあります。詳細については、125 および 302 ページを参照してください。各型の変数に関する詳細については、*Application Programming Guide* を参照してください。

アプリケーションの実行

COMSOL マルチフィジックスのライセンスによって、アプリケーションは COMSOL デスクトップ環境から実行することが可能です。COMSOL サーバーライセンスによって、アプリケーションは各種のオペレーティングシステムとハードウェアプラットフォームの主要なウェブブラウザにおいて実行することができます。さらに、Windows® のためのインストールが容易な COMSOL クライアントによって COMSOL サーバーに接続することによっても、アプリケーションを実行することができます。

以下の二つのセクションでは、どのように COMSOL マルチフィジックスと COMSOL サーバーの環境からアプリケーションを実行するかを説明しています。三つ目のセクションとして、33 ページの「COMSOL アプリケーションの公開」では、アプリケーションを公開するための権限について説明しています。

COMSOL マルチフィジックスでのアプリケーションの実行

COMSOL マルチフィジックスでは、以下のいずれかの方法によってアプリケーションを実行することができます。

- リボンまたはクイックアクセスツールバーで、**アプリケーションをテスト**をクリックする。
- **ファイル**メニューまたはクイックアクセスツールバーで、**アプリケーションを実行**を選択する。
- デスクトップ上の MPH ファイルアイコンをダブルクリックする。
- リボンの**ウェブブラウザーでテスト**を選択する。

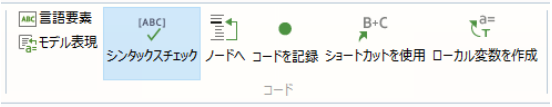
アプリケーションのテスト

アプリケーションをテストによって迅速な確認をすることができます。アプリケーションビルダーのデスクトップ環境を継続しながら、アプリケーションのユーザインタフェースの別のウィンドウが開かれます。

アプリケーションをテストしている最中、**変更を適用**ボタンをクリックすることによって、実行中にフォーム、メソッド、および埋め込みモデルの変更を適用することができます。全ての変更を実行中に適用できるわけではありません。そのような場合は、アプリケーションをいったん閉じてから**アプリケーションをテスト**を再度クリックするように促されます。

アプリケーションを実行せずにフォームのレイアウトをプレビューするには、リボンにある**プレビューフォーム**をクリックします。

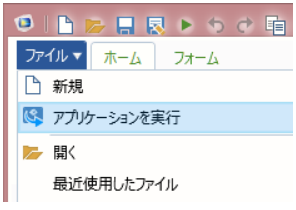
アプリケーションをテストを実行する時には、全てのメソッドは組み込まれている Java® コンパイラによって自動的にコンパイルされます。どのようなシンタックスエラーでもエラーメッセージが生成され、アプリケーションのテストプロセスは止められます。アプリケーションをテストする前にシンタックスエラーをチェックするには、リボンの**メソッド**タブの**シンタックスチェック**ボタンをクリックします。



シンタックスチェックは、組み込まれている Java® コンパイラを使ってメソッドをコンパイルすることによってシンタックスエラーを検索します。この場合、どのようなシンタックスエラーでもメソッドエディター内と**エラーと警告**ウィンドウ内に表示されます。詳細については、149 ページの「メソッドエディター」を参照してください。

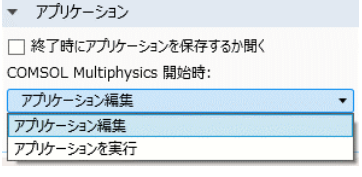
アプリケーションを実行

アプリケーションを実行によって、COMSOL デスクトップ環境においてアプリケーションが開始されます。製品として実行利用させる目的のアプリケーションを利用するには、**アプリケーションを実行**を選択します。例えば、他の誰かが作成したアプリケーションにおいて、編集はパスワードによって保護されているけれども実行は保護されていないような場合、あなたはそれを実行することができます。



MPH ファイルのダブルクリック

Windows® デスクトップで MPH ファイルのアイコンをダブルクリックすると、COMSOL マルチフィジックスに関連付けられた MPH ファイル拡張子を持ったそのアプリケーションが COMSOL マルチフィジックスで開かれます。アプリケーションは、編集か実行のどちらの用途としてでも開くことができます。この選択は、アプリケーションツリーの root ノードで設定することができます。このノードの**設定**ウィンドウには、**アプリケーション**というタイトルのセクションがあり、そこで**アプリケーション編集**か**アプリケーションを実行**の選択を設定します。この選択を変更した場合の設定は、MPH ファイルを保存する際に適用されます。



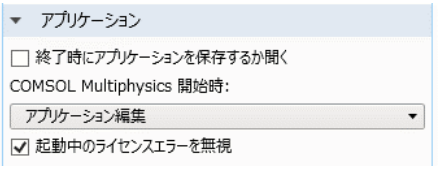
アプリケーション編集を選択した場合は、アプリケーションビルダーでアプリケーションが開かれます。

アプリケーションを実行を選択した場合は、製品として実行利用させる目的のアプリケーションをランタイムモードで開きます。これを選択した場合は、MPH ファイルをダブルクリックすることによって新しい COMSOL マルチフィジックスのセッションが始められるという点を除けば、ファイルメニューで**アプリケーションを実行**を選んだ場合と同じです。

COMSOL Client for Windows® がインストールされていれば、MPH ファイル拡張子は代わりに COMSOL クライアントに関連付けられ、MPH ファイルをダブルクリックした時に COMSOL サーバー端末へのログイン画面が表示されます。

ライセンスエラーの無視

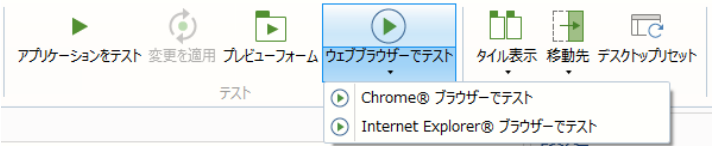
アプリケーションツリーのrootノードの**設定**ウィンドウで、**起動時のライセンスエラーを無視**のチェックボックスを使用して、アプリケーション実行時にライセンスされた製品に関する動作を制御することができます。



これを選択すると、必要なライセンスが全て使用できなくてもアプリケーションを起動できます。ライセンスが利用できない製品の機能は、依然として使用できません。ただし、所望の機能が使用可能なライセンスのタイプに動的に適合するように、メソッドを記述してアプリケーションを作成することは可能です。

ウェブブラウザーでのアプリケーションのテスト

ウェブブラウザーでテストは、ウェブブラウザーでアプリケーションをテストするために使われます。この機能によって、COMSOL サーバー端末に接続されたウェブブラウザー上でアプリケーションが実行された場合のルック&フィール(外観および操作感)を容易にテストすることができます。



アプリケーションを実行させたいウェブブラウザーを選びます。**ウェブブラウザーでテスト**を使うと、アプリケーションビルダーのデスクトップ環境を継続して動作させながら、アプリケーションユーザインタフェースとは分離されたブラウザーウィンドウが開かれます。

アプリケーションをテスト VS ウェブブラウザーでテスト

アプリケーションをテストでは、Microsoft®. NET Framework に基づいたユーザインタフェースによってアプリケーションが実行されます。それに対して、**ウェブブラウザーでテスト**では、HTML5 コンポーネントに基づいたユーザインタフェースによってアプリケーションが実行されます。**アプリケーションをテスト**では、そのアプリケーションが COMSOL マルチフィジックスや COMSOL サーバーで実行された場合のユーザインタフェースとして表示されます。ここで、COMSOL サーバーで実行する場合は、COMSOL Client for Windows® が COMSOL サーバー端末への接続に使われます。
ウェブブラウザーでテストでは、ウェブブラウザーが COMSOL サーバー端末への接続に使われ、そのアプリケーションが COMSOL サーバーで実行された場合のウェブブラウザー上のユーザインタフェースとして表示されます。

OS X、iOS、Linux®、および Android™ といったウェブブラウザー上でのアプリケーションユーザインタフェースの外観と機能をテストするためには、COMSOL サーバー端末が必要です。
次の表は、アプリケーションを実行するための選択肢を簡単に示しています。

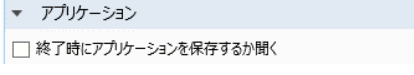
サーバーソフトウェア	クライアントソフトウェアツール または コンポーネント
COMSOL Multiphysics	アプリケーションをテスト
COMSOL Multiphysics	ウェブブラウザーでテスト
COMSOL Multiphysics	アプリケーションを実行
COMSOL Server	COMSOL Client for Windows®
COMSOL Server	ウェブブラウザー

サーバーの欄は、CPU で重い計算を実行するソフトウェアコンポーネントを示しています。クライアントの欄は、アプリケーションユーザインタフェースの表示に使用されるソフトウェアツールまたはコンポーネントを示しています。

実行中のアプリケーションの保存

アプリケーションをテストしている時には、オリジナルの MPH ファイルのコピーとして Untitled.mph という名前が割り当てられています。アプリケーションを実行している場合は、これには当てはまりません。

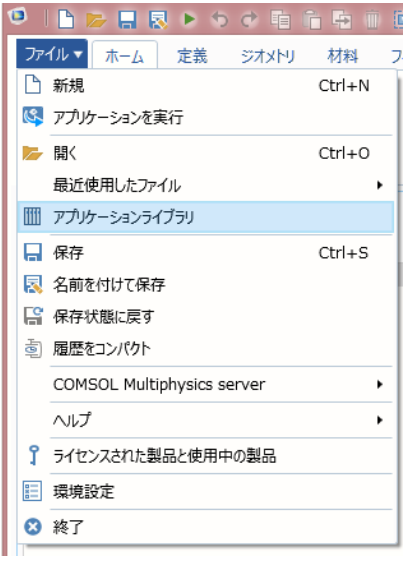
デフォルトでは、アプリケーションを終了する時に、ユーザに対して変更を保存するように表示されます。アプリケーションツリーの root ノードによって、この動作をコントロールすることができます。下図に示すように、このノードの設定ウィンドウには**アプリケーション**というセクションがあり、そこで**終了時にアプリケーションを保存するか聞く**のチェックボックスを設定します。



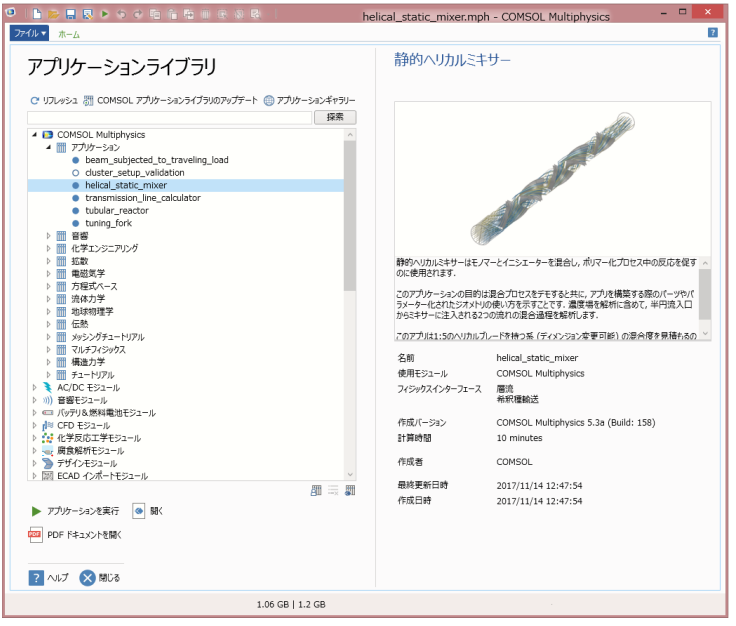
この代わりに、アプリケーションを保存するためのコマンドを設定したボタンやメニュー項目を追加する方法もあります。詳細については、116 ページを参照してください。

アプリケーションライブラリ

ファイルメニューから**アプリケーションライブラリ**を選択し、COMSOL のインストールに含まれているアプリケーションのサンプルを実行して調べることができます。本書に掲載しているスクリーンショットの多くは、それらのサンプルから抜粋されています。



アプリケーションライブラリのツリーの下側にあるボタンをクリックして、アプリケーションを実行するか、または編集のために開きます。



モデルが作成されていてもフォームやメソッドの作成は追加されていないアプリケーションでは、実行することができず、編集のためだけに開かれます。フォームとメソッドを含んでいるアプリケーションは、アプリケーションライブラリの **Applications** という名前のフォルダに集められています。

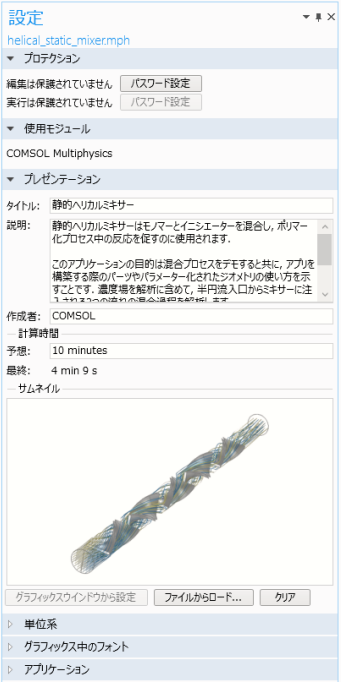
アプリケーションライブラリのアプリケーションは、継続的に改良されてアップデートされています。ツリーの上側にある **COMSOL アプリケーションライブラリのアップデート**をクリックして、アプリケーションライブラリを更新することができます。

アプリケーションライブラリには入っていなが追加されているアプリケーションは、アプリケーションギャラリーの COMSOL ウェブサイトから入手可能です。これらのアプリケーションを探すには、ツリーの上側にある**アプリケーションギャラリー**のボタンをクリックします。これによって、アプリケーションギャラリーのウェブページがブラウザーで開かれます。

アプリケーションライブラリでアプリケーションを選択すると、各々に関するサムネイル画像が表示されます。COMSOL サーバーのウェブインタフェースでは、そこにあるアプリケーションライブラリのページにこのサムネイル画像が表示されます。

サムネイル画像を設定するには、アプリケーションツリーの root ノードをクリックします。その**設定**ウィンドウには、**グラフィックスウィンドウから設定**と**ファイルからロード**の画像選択のための二つのオプションがあります。画像を**クリア**することもできます。

ファイルからロードを選択した場合は、PNG または JPG ファイル形式の画像をロードすることができます。画像が COMSOL マルチフィジックスと COMSOL サーバーのサムネイルとして適切に表示されることを保証するため、280×210 ピクセルから 1024×768 ピクセルまでの画像サイズを選択します。



グラフィックスウィンドウから設定の選択では、以下の二つのサムネイル画像が自動的に作成されます。

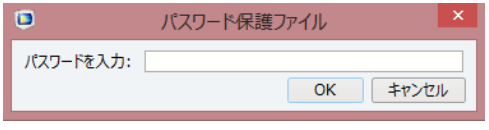
- アプリケーションツリーの root ノードの設定ウィンドウ、およびアプリケーションライブラリにおける表示画像として使われる 280×210 ピクセルサイズの画像
- レポート、および COMSOL サーバーのアプリケーションライブラリにおけるデフォルトのタイトルページの表示画像として使われる 1024×768 ピクセルサイズの画像

パスワード保護

アプリケーションをパスワードで保護して、アクセス権限を管理することが可能です。アプリケーションビルダーウィンドウのアプリケーションツリーの root ノードをクリックし、その**設定**ウィンドウの中で編集のためと実行のための個別のパスワードを割り当てます。

このパスワードを作成するためには、アプリケーションを編集するアクセス権限を持っていなければなりません。

パスワードで保護された MPH ファイルを編集や実行のために開く時に、そのパスワードの入力を要求するダイアログボックスが表示されます。



パスワード保護を解除するには、パスワードの設定を空白にします。

パスワード保護は、メソッドを含む全てのモデルとアプリケーション設定を暗号化するために用いられます。しかし、CAD ファイル、メッシュデータ、ソリューションデータを組み込んで完成されたジオメトリのようなバイナリデータは、暗号化されません。

セキュリティ設定

アプリケーションビルダーでアプリケーションを作成する時には、アプリケーションをホスティングしているコンピュータのセキュリティを考慮することが重要です。COMSOL マルチフィジックスと COMSOL サーバーでは、アプリケーションが C ライブラリへのリンク、MATLAB 関数の実行、外部プロセスのアクセスなどの外部機能の呼び出しを許可するかどうかをコントロールするために、通常のホストコンピュータの場合と非常に類似したセキュリティ設定のセットを提供しています。

COMSOL マルチフィジックスのセキュリティ設定は、**ファイル**メニューからアクセスできる**環境設定**ウィンドウの**セキュリティ**ページにあります。COMSOL サーバーでは、管理者権限でログインできる場合、COMSOL サーバーウェブインタフェースの環境設定のページでセキュリティ設定をすることができます。もし、どんなセキュリティ設定にするべきかわからない場合は、システム管理者に連絡してください。

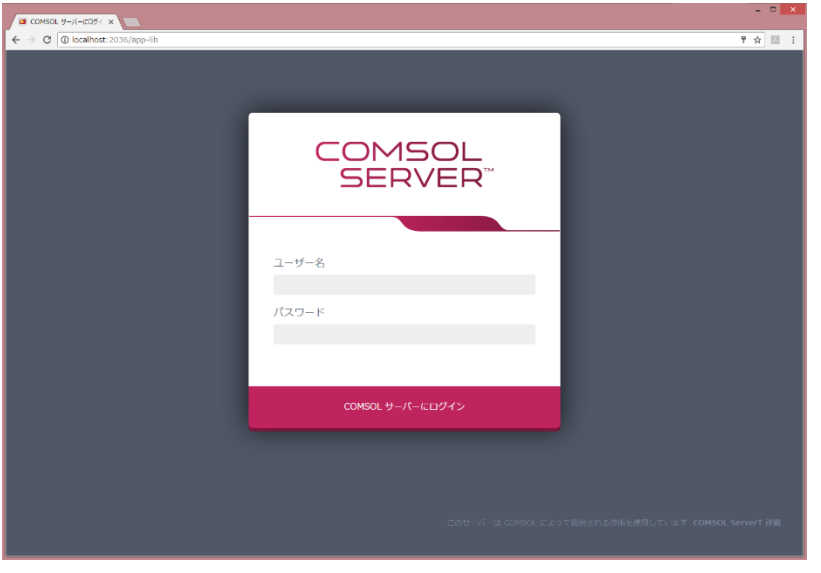
COMSOL サーバーでのアプリケーションの実行

COMSOL アプリケーションは、ウェブブラウザー、または COMSOL Client for Windows® から COMSOL サーバーに接続することによって実行可能です。COMSOL Client for Windows® では、30 ページの COMSOL クライアントでのアプリケーションの実行 で説明されているように、CAD のための LiveLink™ 製品を必要とするアプリケーションの実行をユーザに許可しています。

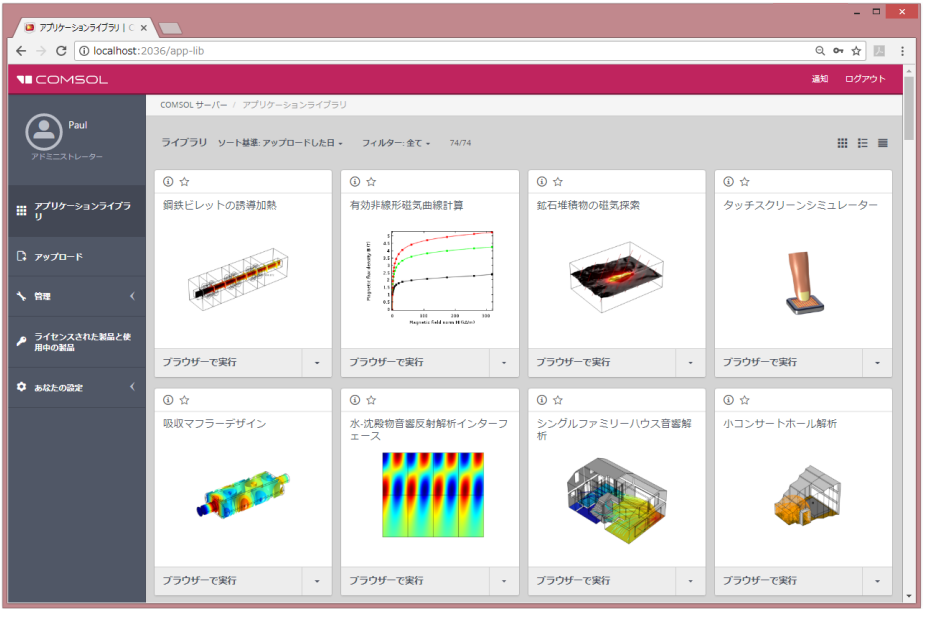
ウェブブラウザーでアプリケーションを実行するには、インストールの必要は全くなく、ウェブブラウザープラグインも全く必要ありません。ウェブブラウザーでのアプリケーションの実行では、1D、2D、および 3D の対話型グラフィックスがサポートされています。ウェブブラウザーでの 3D のグラフィックス描画は、全ての主要なウェブブラウザーに備わっている WebGL™ の技術に基づいています。

ウェブブラウザーでのアプリケーションの実行

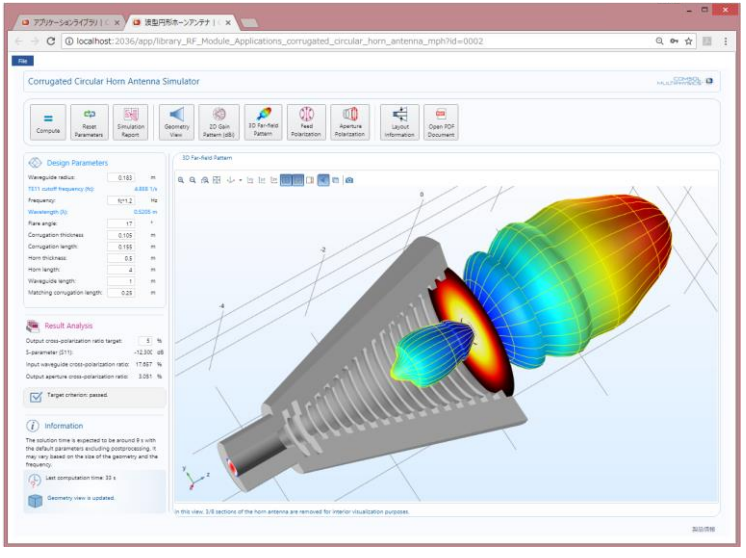
ウェブブラウザーを使うことによって、COMSOL サーバーウェブインタフェースのコンピュータ名とポート番号を直接指定することができます。例えば、ポート番号 2036 が COMSOL サーバー端末で使われると仮定すると、http://comsol-server-machine-url.com:2036 の如くです。ログインには、ユーザ名とパスワードの入力が必要です。



ログイン後は、実行するアプリケーションのリストが示されているアプリケーションライブラリのページが表示されます。



アプリケーションを実行するには、**ブラウザーで実行**をクリックします。アプリケーションは、ブラウザー表示の中に個々のタブが作られて実行されます。



ウェブブラウザーでのアプリケーション実行時の制限

ウェブブラウザーで動作させるアプリケーションを作成する場合は、95 ページの「スケッチレイアウトとグリッドレイアウト」を参照し、アプリケーションビルダーのグリッドレイアウトモードを使うようにしてください。これにより、ブラウザーウィンドウのサイズと縦横比に適応したユーザインタフェースレイアウトとなります。低解像度のディスプレイの場合には、目的とするプラットフォームのユーザインタフェースレイアウトをテストして、その全てのフォームオブジェクトが表示されることを確かめてください。サイズ変更可能なグラフィックスフォームを含んでいるアプリケーションは、低解像度のディスプレイには適応しないかもしれません。そのような場合には、グラフィックスの幅と高さを固定とし、全てのフォームオブジェクトが目的のブラウザウィンドウに適応することを確かめてください。ウェブブラウザーとグラフィックスカードのタイプによっては、アプリケーションで使うことができるグラフィックスオブジェクトの数に制限がある場合があります。そのような制限を回避するために、複数のグラフィックスオブジェクトを使う代わりに、ソースを切り換えて同じグラフィックスオブジェクトを再利用する方法をとることができます。

ウェブブラウザーでの実行では、CAD ソフトウェアパッケージのための LiveLink™ 製品はサポートされていません。

スマートフォンやタブレットを使ってウェブブラウザーで COMSOL アプリケーションを実行する場合は、全ての機能がサポートされてはいません。この典型的な制限としては、音を鳴らしたりドキュメントを開くといった機能です。さらに、ファイルのアップロードとダウンロードはサポートされていません。

境界条件を設定するために境界をクリックするといったように、ユーザに対する選択を可能にしているアプリケーションでは、ウェブブラウザーで実行する場合と COMSOL マルチフィジックスまたは COMSOL Client for Windows® で実行する場合とは異なります。ウェブブラウザーでは、境界がホバリング時に自動的にハイライト表示されません。その代わり、境界をハイライト表示にするには、1 回クリックする必要があります。2 回目のクリックによって選択されます。3 回目のクリックによって選択解除のためにハイライト表示され、4 回目のクリックによって選択が解除されます。このプロセスは、ドメイン、エッジ、およびポイントの場合も同様です。

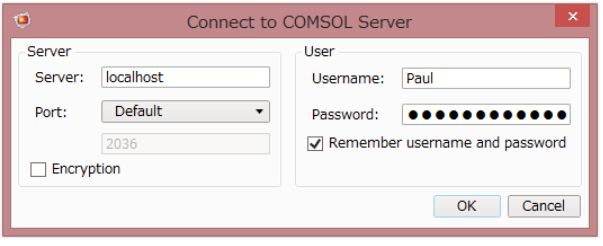
ファイルのブラウズ機能は、ウェブブラウザーやウェブブラウザーのバージョンによって多少異なります。このため、ファイルをクライアントコンピュータに保存する機能を持つアプリケーションを実行する際などには、そのユーザの経験に影響するかもしれません。例えば、ダウンロードフォルダの場所は、多くのウェブブラウザーの設定で変更することができます。また、ウェブブラウザーでは、各ファイルのダウンロード場所をユーザが手動で指定することも許されます。詳細については、対象のウェブブラウザーのドキュメントを参照してください。

COMSOL クライアントでのアプリケーションの実行

アプリケーションを実行するためにウェブブラウザーを使う代わりに、COMSOL Client for Windows® 使った COMSOL サーバーとの接続によって、Windows® オペレーティングシステムでの自然な（作成時の形に近い）アプリケーションの実行が可能です。一般に、これによって、よりよいグラフィックスの性能を得ることができ、1D、2D、および 3D での洗練されたグラフィックス描画をサポートしてくれます。さらに、COMSOL Client for Windows® は、接続されている COMSOL サーバーが必要なライセンスを持っているならば、CAD のための LiveLink™ 製品を必要とするアプリケーションの実行が許可されます。COMSOL Client for Windows® でアプリケーションを開くには、次に示す二つの方法があります。

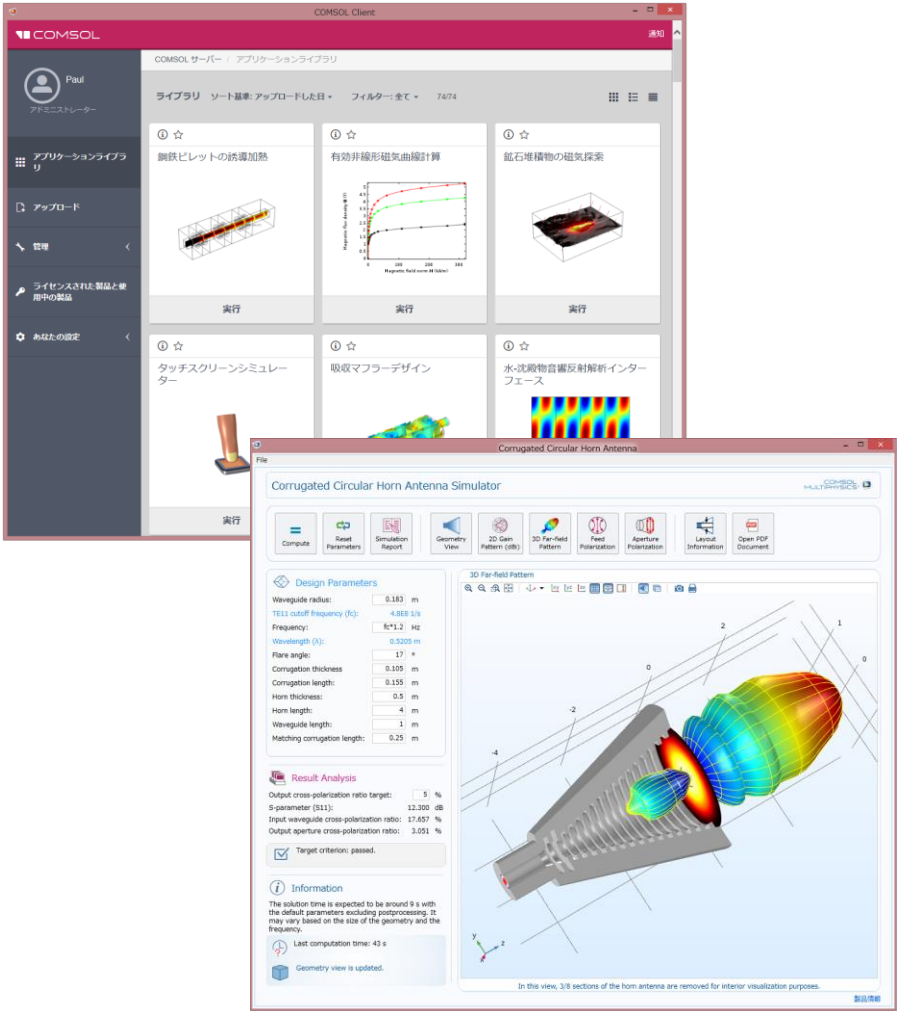
- COMSOL サーバーウェブインタフェースでは、アプリケーションをウェブブラウザーで実行するか、COMSOL Client for Windows® で実行するかを選択することができます。ここで、アプリケーションを COMSOL クライアントで実行する選択をした場合に、まだインストールされていなければ、それをダウンロードしてインストールすることを促す表示がされます。

- すでに COMSOL Client for Windows® がインストールされていれば、デスクトップショートカットが利用可能です。そのデスクトップのアイコンをダブルクリックして、アプリケーションを実行する COMSOL クライアントを使用開始にする前に、有効なユーザ名とパスワードを入力して COMSOL サーバーにログインする必要があユーザ名とパスワードを入力して COMSOL サーバーにログインする必要があります。ログイン後、COMSOL クライアントは、ウェブブラウザーからログインした場合と同様の COMSOL サーバーウェブインタフェースを表示します。



COMSOL クライアントを使うと、アプリケーションは別ウィンドウによって自然な（作成時の形に近い）Windows® アプリケーションとして動作します。例えば、COMSOL クライアントで実行されるアプリケーションは、Windows® のタブ付きのリボンを持ちます。ウェブブラウザーで実行される時には、そのリボンはツールバーとして表示されます。

下図の上側には COMSOL サーバーウェブインタフェースが示されており、下側には COMSOL Client for Windows® で実行されたアプリケーションが示されています。



複数のコンピュータやクラスタでの COMSOL サーバーの実行

COMSOL アプリケーションを複数のコンピュータまたはクラスタで実行するには、主に二つの方法があります。

- Primary および Secondary インスタンスとして COMSOL サーバーをインストールする方法。
- 特定のクラスタのために、モデルビルダーのスタディ(Study)ノードの一つを設定する方法。

Primary および Secondary インスタンス

Primary と Secondary インスタンスを使用して複数のコンピュータで COMSOL サーバーを実行すると、単一のコンピュータインスタンス（またはインストール）によるよりも、より多くの同時ユーザやアプリケーションが可能になります。メイン COMSOL サーバーのインスタンスは Primary と呼ばれ、その他は Secondary と呼ばれます。Primary サーバーは、全ての着信接続 — 例えば、ウェブインタフェースを表示したり、ウェブブラウザーや COMSOL クライアントでアプリケーションを実行するために使用されます。実際の計算は、Secondary サーバーコンピュータに任されます。このタイプのインストールを行うと大きな利点があります。それは、特別なクラスタのためにアプリケーションをカスタムビルドする必要がないこと、また、Secondary サーバー間で作業負荷が分散され、Primary サーバーによってロードバランスが自動で管理されることです。COMSOL サーバーのインストールは、追加のライセンス要件なしに、複数の Primary および Secondary サーバーのインストールからなります。アプリケーション実行中のユーザのライセンスキーをチェックアウトせずに、COMSOL サーバーのウェブインタフェースを使用して管理タスクを実行できます。ライセンスキーは、アプリケーションの実行時にのみチェックアウトされます。

クラスタスイープまたはクラスタコンピューティングのためのスタディノード設定

大規模なパラメトリックスイープや高性能な計算パワーを必要とするアプリケーションのためにクラスタを利用する場合には、クラスタスイープとクラスタコンピューティングのオプションを使用して、アプリケーションのモデルビルダーのスタディ（Study）ノード（複数可）を設定します。このようなアプリケーションを構築するには、フローティングネットワークライセンスが必要になることに注意してください。*Introduction to COMSOL Multiphysics* と *COMSOL Multiphysics Reference Manual* の中で、クラスタのためのスタディ（Study）ノードの設定に関する詳細情報を参照することができます。このようなクラスタ対応のアプリケーションを実行するためには、COMSOL サーバー、または COMSOL マルチフィジックスのフローティングネットワークライセンスのいずれかを使用します。クラスタシステム構成は、COMSOL サーバーのウェブインタフェースから利用できます。

COMSOL サーバーについての詳細情報は、COMSOL サーバーのインストール、または http://www. comsol.com/documentation/COMSOL_ServerManual.pdf から入手可能な *COMSOL Server Manual* を参照してください。

COMSOL アプリケーションの公開

COMSOL ソフトウェアライセンス契約書（SLA）では、使用する他の人のために COMSOL のアプリケーションを公開する権限を与えており、商業的にも許しています。SLA は、www.comsol.com/sla から利用可能で、その中でいくつかの制限について記述されています。この権限によって、他のユーザとアプリケーションを共有し、あなたのアプリケーションを使用するために彼らに課金することが可能です。

あなたが利用できるようにするアプリケーションのユーザは、COMSOL マルチフィジックス、あるいは必要なアドオン製品を持つ COMSOL サーバーのいずれかへのアクセスが必要となります。ユーザが COMSOL マルチフィジックスでアプリケーションを使用するには、COMSOL マルチフィジックスのライセンスを購入した同じ組織に属している必要があります。より柔軟性を高めるために、COMSOL サーバーのインストールをセットアップすることによって、世界中のユーザにアプリケーションをアクセスさせることができます。あなたは自分の COMSOL サーバーのインストールでユーザ名とパスワードを提供する必要があるだけです。代わりに、ユーザが自分の COMSOL サーバーのライセンスを購入することもできます。

また、www.comsol.com/sla でも入手できる COMSOL のアプリケーションライセンスでは、あなたがアプリケーションライブラリで利用可能なアプリケーションをさらに変更し、アプリケーションライセンスで記載されている一定の制約のもとで、商業目的も含めてそれらの変更されたアプリケーションを公開することができます。例えば、これによって、アプリケーションライブラリにあるアプリケーションのいずれかを使用し、そこに独自の機能を追加したり削除したりして、独自のアプリケーション作成の出発点とすることができます。

あなたが作成したアプリケーションにアプリケーションライセンスを適用したい場合、その方法はアプリケーションライセンスに説明されています。また、アプリケーションライブラリで利用可能なアプリケーションを変更するための選択条件の使い方が記載されています。アプリケーションライブラリのオリジナル部分はアプリケーションライセンスの下で常に利用可能です。

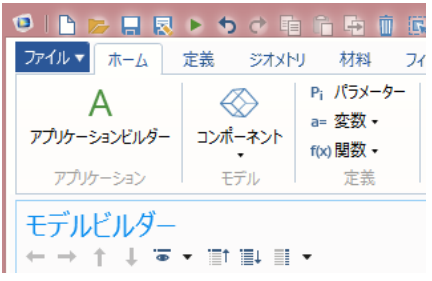
また、あなたがホストしてアプリケーションを実行するために COMSOL サーバーを使用している場合、SLA では、あなたがホストして一定の制限に基づいて他の人に公開しているアプリケーションを実行するために、組織外の人が利用可能な許可が COMSOL サーバーライセンス（CSL）によって与えられています。

アプリケーションビルダーを始める

COMSOL マルチフィジックスモデルから始める

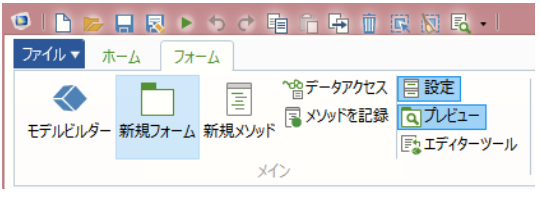
まだ COMSOL デスクトップ環境にモデルをロードしていないなら、**ファイル** > **開く** を選択し、ファイルシステムから MPH ファイルを選択するか、またはアプリケーションライブラリからファイルを選択します。**Applications** のフォルダにあるファイルがすぐ使えるアプリケーションであることを確認してください。アプリケーションライブラリにある全てのファイルには、モデルとドキュメントが含まれていますが、アプリケーションユーザインタフェースが全てに含まれているわけではありません。

いったんモデルがロードされたら、リボンの**ホーム**タブにある**アプリケーションビルダー**のボタンをクリックします。これにより、アプリケーションビルダーのデスクトップ環境に切り換わります。



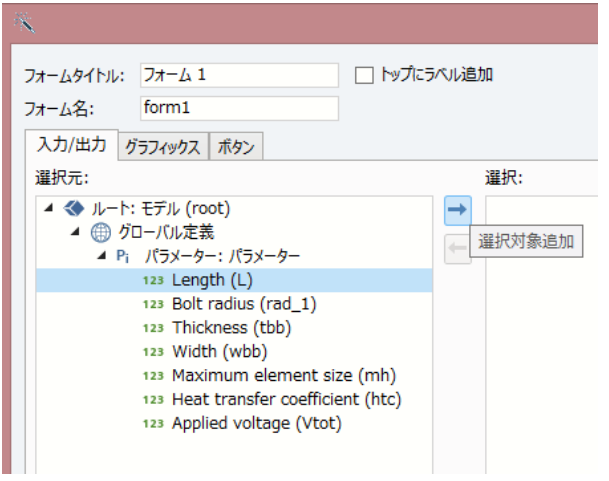
新規フォームの作成

ユーザインタフェースレイアウトの作業を始めるには、**ホーム**タブの**新規フォーム**のボタンをクリックします。これにより、**新規フォーム**ウィザードが開始されます。



新規フォームウィザードは、最も一般的なユーザインタフェースの構成要素、いわゆるフォームオブジェクトをアプリケーションの最初のフォームに追加するための補助をしてくれます。そのウィザードには、以下の三つのタブがあります。

- **入力/出力**
- **グラフィックス**
- **ボタン**

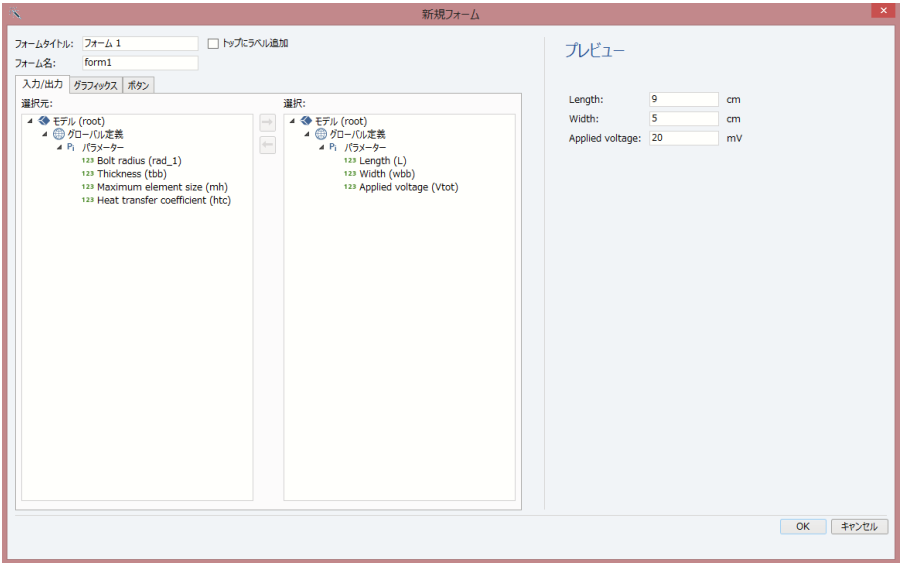


ノードをダブルクリックするか、**選択対象追加**のボタンをクリックすると、そのノードが**選択元**エリアから**選択**エリアに移されます。選択されたノードは、アプリケーションのフォームオブジェクトとして生成され、右側の**プレビュー**エリアにフォームのプレビューが表示されます。サイズは、フォームオブジェクトのその他の設定と同様、ウィザードを終了した後に編集して変更することができます。ウィザードウィンドウの一番上側の部分では、フォームタイトルとフォーム名を変更することができます。詳細は、41 ページの「個々のフォーム設定ウィンドウ」を参照してください。
この段階で**新規フォーム**ウィンドウを終了するには、**OK** をクリックします。その後にフォームオブジェクトを手動で追加することもできます。

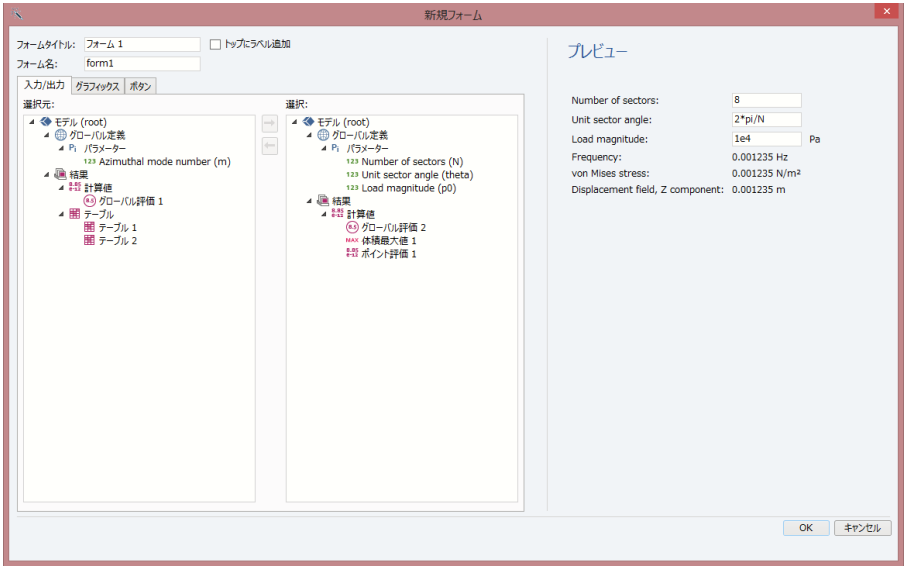
入力/出力タブ

入力/出力タブにはモデルツリーノードが表示されており、入力フィールド、データ表示オブジェクト、チェックボックス、またはコンボボックスの利用に役立ちます。このウィザードによって入力フィールドが追加される場合、適用可能なテキストラベルと単位もいっしょに追加されます。**データアクセス**(89 ページ参照)を使えば、そのモデルの他の部分も入力と出力に利用可能にすることができます。チェックボックスとコンボボックスオブジェクトは、この方法によってのみ利用可能にすることができます。例えば、**メッシュ**ノードの下にある**要素サイズ**を**データアクセス**によって有効にすれば、ウィザードで利用可能な既定のコンボボックスを作ることができます。

下図では、長さ、幅、および適用電圧の三つのパラメータが、入力フィールド用に選択されています。



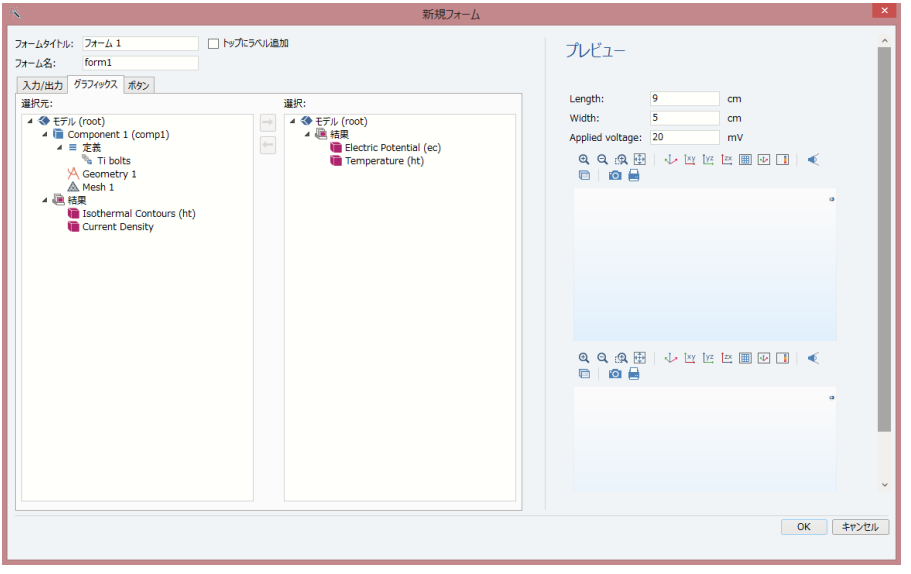
別のモデルにおける下図では、三つの**計算値**（Derived Values）ノードが、データ表示オブジェクト用に選択されています。



ウィザードを終了した後に、入力フィールドとデータ表示オブジェクトのサイズ、フォント色、およびその他の設定も編集することができます。

グラフィックスタブ

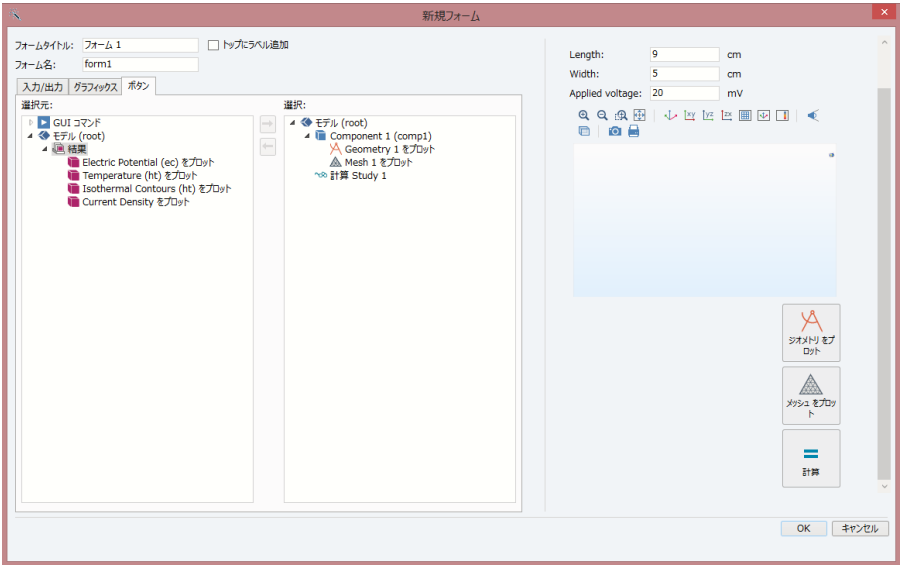
グラフィックスタブでは、グラフィックスオブジェクトとして利用される可能性の高いモデルツリーノードの候補が表示されています。その候補としては、**ジオメトリ**(Geometry)、**選択**(Selection)、**メッシュ**(Mesh)、および**結果**(Results)があります。下図では、そのような二つのノードが選択されています。



ボタンタブ

ボタンタブでは、アプリケーションユーザインタフェースであるボタンをクリックすることによって実行できるモデルとアプリケーションツリーノードが表示されます。そのようなツリーノードの例としては、**プロット ジオメトリ**(Geometry)、**プロット メッシュ**(Mesh)、**計算 スタディ**(Study)、および**結果**(Results)ノードの下に展開されるさまざまなプロットグループがあります。さらに、**GUI コマンド**、**フォーム**、および**メソッド**のボタンを追加することができます。

下図では、**プロット ジオメトリ**、**プロット メッシュ**、**計算**の三つのボタンが追加されています。



ウィザード終了後、フォームエディターを使うことによって、自分自身でカスタマズしたコマンドシーケンスやメソッドを実行するためのボタンを追加することができます。

ウィザードの終了

ウィザードを終了するには、**OK** をクリックします。これにより、自動的に開かれたフォームエディターが表示されます。

アプリケーションの保存

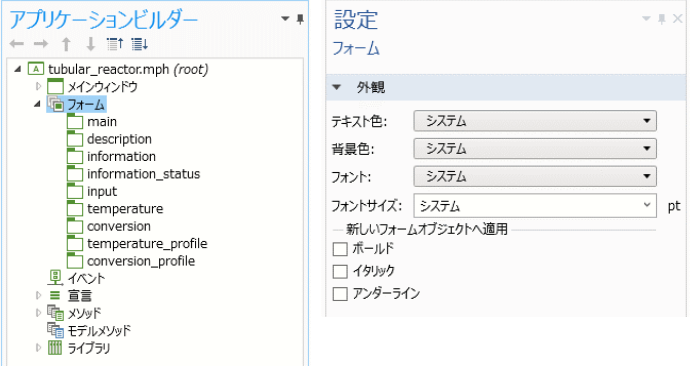
アプリケーションを保存するには、**ファイル**メニューから**ファイル > 保存** を選択します。書き込みが許可されたフォルダにファイルをブラウズし、MPH ファイル形式で保存します。MPH ファイルにはアプリケーションに関する情報の全てが含まれており、モデルビルダーで作成された埋め込みモデル情報も含まれています。

フォームエディター

フォームエディターは、入力フィールド、グラフィックス、ボタンなどのフォームオブジェクトを用いてフォームを作成するためのユーザインタフェースレイアウトとして使われます。

フォーム設定ウィンドウ

フォームの**設定**ウィンドウは、アプリケーションツリーにある**フォーム**ノードをクリックすると表示されます。**テキスト色**、**背景色**、**フォント**、**フォントサイズ**、**ボールド**、**イタリック**、および**アンダーライン**の設定によって、フォーム全体の外観を変更することができます。

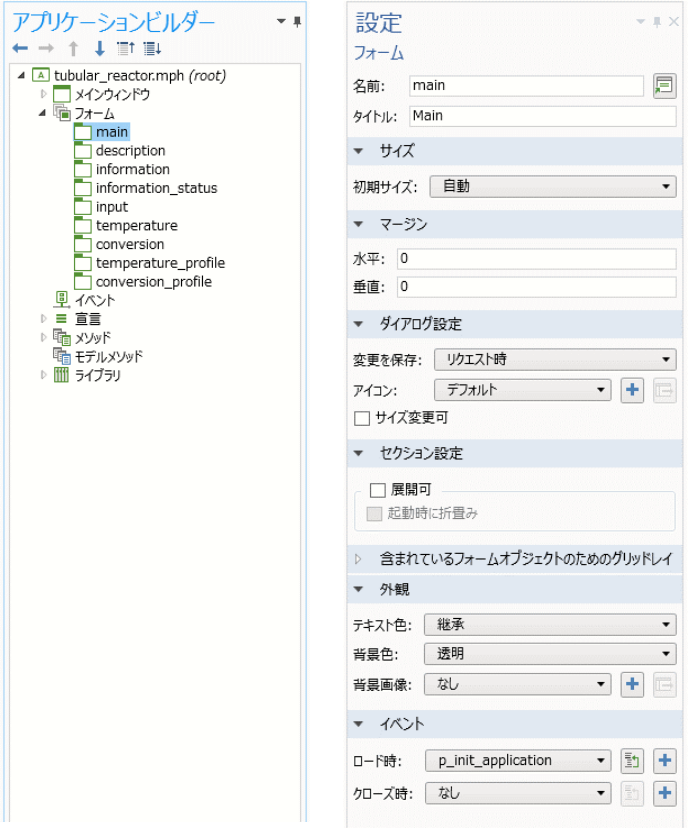


全ての新規フォームと新たに追加したオブジェクトに、該当するこれらの設定がデフォルトとして引き継がれます。

上図や、以降のいくつかの図で示している**設定**ウィンドウは、**アプリケーションビルダー**ウィンドウの右側に配置されています。デフォルトとして、**設定**ウィンドウはアプリケーションビルダーのデスクトップ環境の中で最も右側に配置されます。

個々のフォーム設定ウィンドウ

下図は、あるフォームにおける**設定**ウィンドウを示しています。

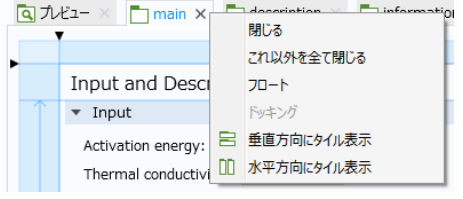


個々のフォームにはそれ自身の**設定**ウィンドウがあり、以下のような設定項目があります。

- 他のフォームのフォームオブジェクトやメソッドから、そのフォームを参照するために用いられる**名前**
- 複数のフォームを持つアプリケーションで使われるフォームの**タイトル**
- フォームがダイアログボックスとして使われる場合、または**メインウィンドウ**がフォームで決められたサイズに設定される場合におけるフォームの**初期サイズ**
- フォームの左上角（水平と垂直）の余白を設定する**マージン**のセクション
- ダイアログボックスで変更を格納するための設定（**変更を保存**）
  これに関しては、59 ページの「ダイアログボックスとしてフォームを表示」を参照してください。
- ダイアログボックスの左上角に表示させる**アイコン**

- フォームがダイアログボックスとして使われる場合、**サイズ変更可**か否かの設定
- セクションを**展開可**として見せるか否か、**起動時に折畳み**とするか否かの設定（**セクション設定**）
- フォームの全ての行と列のサイズ形式を設定するテーブル（**含まれているフォームオブジェクトのためのグリッドレイアウト**のセクション）――――（注）グリッドモードの場合
- **テキスト色**、**背景色**、および**背景画像**の設定をする**外観**のセクション
- フォームがロードされる時（**ロード時**）、または閉じられる時（**クローズ時**）に起動される**イベント**

フォームエディターのウィンドウを開くためには、そのフォームノードをダブルクリックします。あるいはその代わりに、フォームノードを右クリックして**編集**を選択します。フォームウィンドウのタブを右クリックすると、閉じる、フロート、およびタイル表示といったフォームウィンドウの選択肢に関するコンテキストメニューが表示されます。



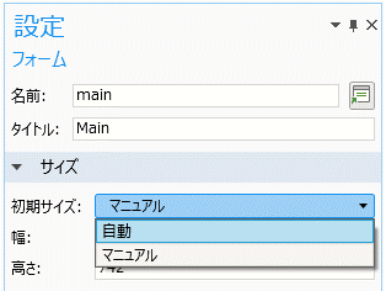
スケッチレイアウトモードとグリッドレイアウトモード

アプリケーションビルダーは、スケッチレイアウトモードがデフォルト設定となっており、それによってオブジェクトの位置とサイズを固定して使うことができます。「フォームエディター」の章における紹介では、指定されない限り、フォームエディターはスケッチレイアウトモードであると仮定しています。グリッドレイアウトモードについての情報としては、95 ページの「スケッチレイアウトとグリッドレイアウト」を参照してください。

フォームの初期サイズ

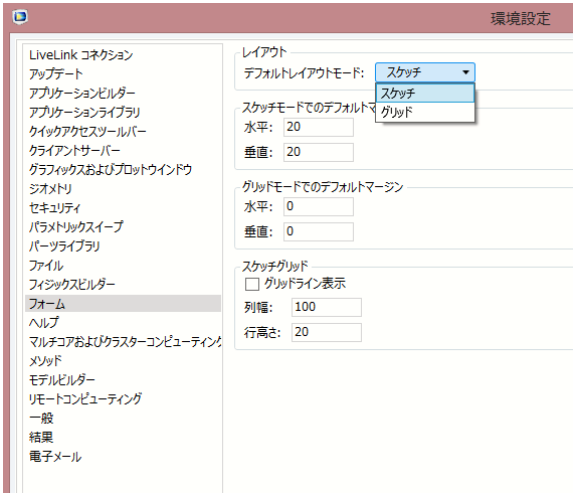
フォームの初期サイズとして、以下の二つの選択肢があります。

- **マニュアル**は、幅と高さのピクセルサイズを入力指定します。
- **自動**は、フォームに配置されているフォームオブジェクトに基づいてサイズが決められます。グリッドレイアウトモードを使っている場合に、適合するようにセットされた行または列がある場合、そのフォームサイズはフォームオブジェクトによっては決められません。この場合には、サイズはフォームエディターのグリッドサイズを基点として使って見積もられます。（それは通常は少し大き目に。）グリッドの右側または下側の境界線をドラッグしてグリッドサイズを変更することができます。グリッドレイアウトモードの詳細については、98 ページの「グリッドレイアウト」を参照してください。



フォームエディターの環境設定

フォームエディターの**環境設定**にアクセスするためには、**ファイル**メニューから**環境設定**を選択し、その中の**フォーム**のページを選びます。

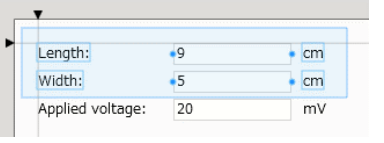


その**フォーム**のページでは、**レイアウトモード**、**マージン**、および**スケッチグリッド**のデフォルトを変更するための設定が可能です。

フォームオブジェクトの配置

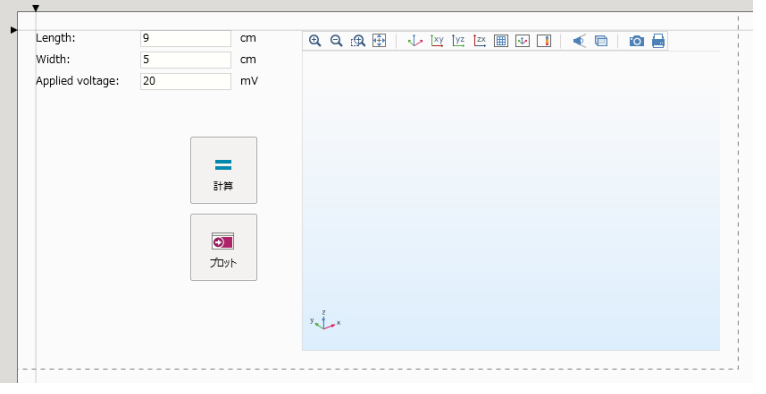
入力フィールド、グラフィックス、ボタンなどのフォームオブジェクトの配置は、以下のいずれかの方法で容易に変更することができます。

- オブジェクトをクリックして選択します。選ばれたオブジェクトは青色の枠でハイライト表示されます。
- 複数のオブジェクトの選択は、Ctrl+click の操作によります。また、クリック&ドラッグの操作によってフォームウィンドウに選択領域を指定することによって、その範囲内の全てのオブジェクトを選択することができます。



- 選択領域にあるオブジェクトの位置の移動は、ホールド&ドラッグの操作によります。この選択範囲が青色で示されているため、他のオブジェクトとの位置関係を把握しながら移動させられます。
- スケッチレイアウトモードでは、キーボードの矢印キーを使ってオブジェクトを移動させることもできます。配置を微調整するためには、Ctrl+arrow キーの操作によります。

下図では、**プロット**ボタンを元の配置から移動しています。青色のガイドラインは、そのボタンと**計算**ボタンの相対位置の一致を表しています（下図ではボタン右辺の縦線）。



フォームオブジェクトのサイズ変更

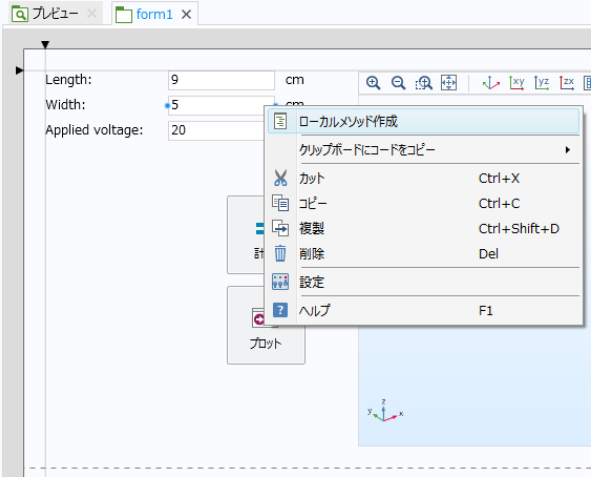
オブジェクトのサイズを変更するには、以下のように行います。

- まず、オブジェクトをクリックします。
- ハイライト表示された青色の枠に付いているドット部分（ハンドル）のうちの 1 つをホールド&ドラッグ操作することによって、サイズを変更することができます。このドット部分（ハンドル）が一つも表示されないタイプのフォームオブジェクトは、サイズ変更ができません。

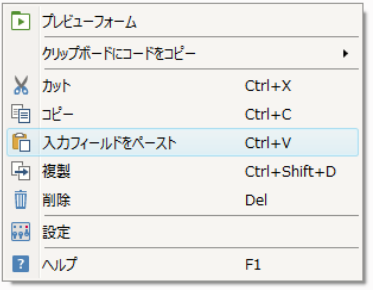
オブジェクトのコピー、ペースト、複製、削除

オブジェクトを削除するには、それをクリック選択してからキーボードの Delete を押します。また、クイックアクセスツールバーにある削除ボタンをクリックすることによっても削除が可能です。

オブジェクトのコピーペーストは、Ctrl+C と Ctrl+V のキー操作でも可能です。その代わりに、オブジェクトを右クリックして、**コピー**、**複製**、**削除**などの選択メニューを表示させる方法もあります。



すでにコピーしたオブジェクトを貼り付けるには、フォーム上の空き領域で右クリックします。オブジェクトがコピーされていれば、選択メニューにペーストが表示されています。下図では、以前に入力フィールドがコピーされた結果として、選択メニューに**入力フィールドをペースト**が表示されています。



ピクセル数による位置およびサイズの調整

スケッチレイアウトモードの場合、**設定**ウィンドウの**位置およびサイズ**のセクションでピクセル数を入力することによって、オブジェクトの位置およびサイズを調整することができます。

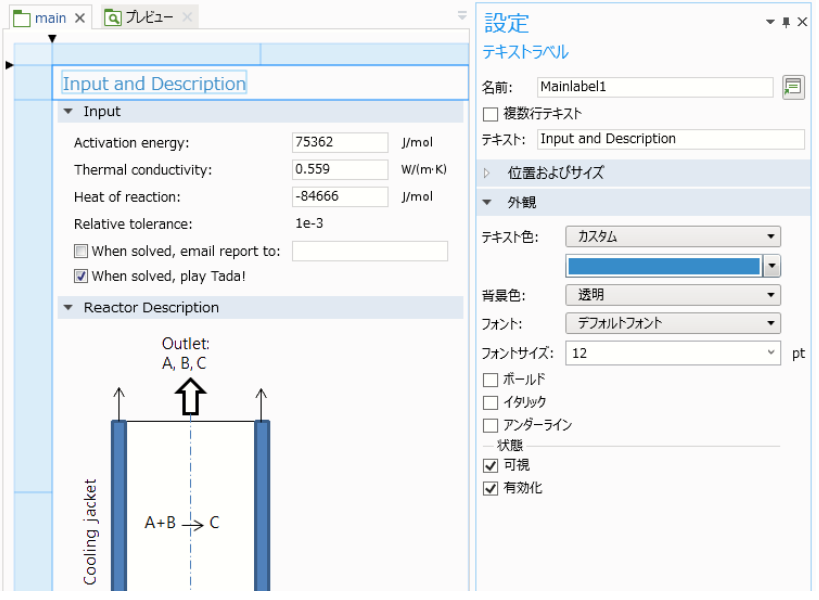
- その設定をするには、まずはオブジェクトをクリックし、その**設定**ウィンドウが表示されたことを確認します。表示されない場合は、オブジェクトをダブルクリックするか、**フォーム**タブにある**設定**ボタンをクリックしてください。
- **位置およびサイズ**のセクションで、ピクセル数を編集します。（下図は、スケッチレイアウトモード時の場合です。）

位置およびサイズのセクションの選択項目は、フォームオブジェクトのタイプによって異なります。グリッドレイアウトモードの場合には、オブジェクトの列と行の位置に関する追加設定項目があります。詳細は、95 ページの「スケッチレイアウトとグリッドレイアウト」を参照してください。

表示されたテキストの外観変更

テキストが表示されるフォームオブジェクトの場合、**設定**ウィンドウの**外観**セクションでは、テキスト表示、そのフォント、フォント色、フォントサイズなどの設定内容を変更することができます。ボタンなどのいくつかのフォームオブジェクトの場合、そのオブジェクトに表示するテキスト文字列の長さに応じてサイズも変化します。

下図に示すテキストラベルオブジェクトの**設定**ウィンドウでは、フォントサイズと色が変更されています。



グリッドレイアウトモードを使用することにより、ボタンに任意のサイズを設定するなど、フォームオブジェクトのサイズ以上のさらなる制御を得ることができます（95 ページの「スケッチレイアウトとグリッドレイアウト」を参照）。

複数のフォームオブジェクトの選択

複数のフォームオブジェクトを選択する場合は、例えば Ctrl+click の操作によります。この場合の**設定**ウィンドウは、選択されている複数のオブジェクトに共通の設定項目となっています。共通の設定項目は、多くの場合、**外観**セクション、**位置およびサイズ**セクション、または**イベント**セクションです。

フォームオブジェクトの名前

フォームオブジェクトの**名前**は、半角スペースを含まないテキスト文字列です。この文字列には、文字、数字、および下線が含まれます。さらに、root と parent は予約語となっているので、使うことができません。**名前**の文字列は、そのオブジェクトを参照する他のフォームオブジェクトやメソッドでも使われます。**設定**ウィンドウの**名前**フィールドへのマウスカーソルのホバリングによって、オブジェクトへのパスがツールチップ表示されます。

フォームオブジェクトの挿入

新規フォームウィザードにより作成されたフォームオブジェクトに加えて、後からフォームオブジェクトを挿入することができます。**フォーム**リボンタブの**オブジェクトを挿入**をクリックしてメニューを表示し、そこにある全ての選択可能なオブジェクトのリストから選びます。

フォームエディター の章で説明されていない内容としては、**新規フォーム**ウィザードによって追加されるフォームオブジェクトのタイプに関してだけです。ウィザードを使って追加されたフォームオブジェクトには、以下が含まれています。

- **ボタン**
- **グラフィックス**
- **入力フィールド**
- **テキストラベル**（入力フィールドに関連）
- **単位**（入力フィールドに関連）
- **データ表示**

一方、**データアクセス**（ 89 ページ参照）を使う時には、以下のフォームオブジェクトも追加することが可能です。

- **チェックボックス**
- **コンボボックス**

チェックボックス、コンボボックス、およびその他のフォームの詳細については、195 ページの「付録 A—フォームオブジェクト」を参照してください。

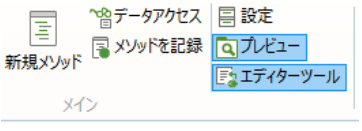
フォームオブジェクトに関連するイベントとアクション

ボタン、メニュー項目、リボンボタン、フォーム、フォームオブジェクトなどのオブジェクトを、イベントによって起動されるアクションに関連付けることができます。このアクションは、グローバルメソッドまたはローカルメソッドを含む一連のコマンドです。ローカルメソッドは、それらが定義されているオブジェクト以外からはアクセスできず、アプリケーションビルダーツリーに表示もされていません。オブジェクトと関連付けることができるイベントは、オブジェクトのタイプに依存し、次のものを含んでいます： ボタンクリック、キーボードショートカット、フォームのロード（**ロード時**）、フォームを閉じる（**クローズ時**）、変数の値の変更（**データ変更時**）。

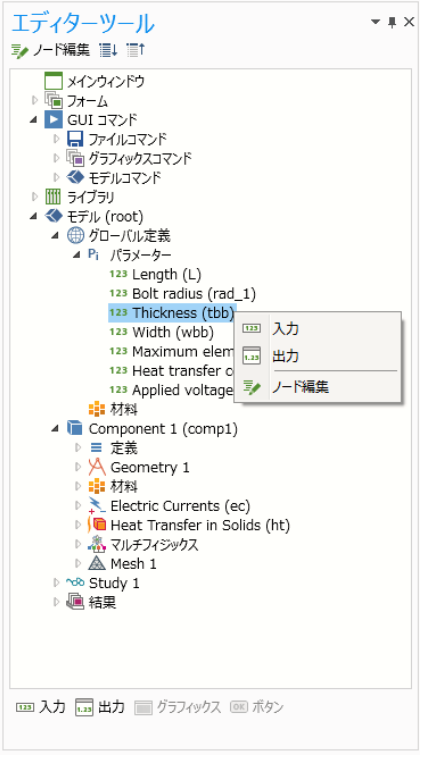
フォームオブジェクト上での Ctrl+Alt+click により、それに関連したメソッドを**メソッド**エディターで開くことができます。もし、フォームオブジェクトと関連したメソッドが全くない場合には、そのフォームオブジェクトに関連付けられた新規のローカルメソッドが作成されて、**メソッド**エディターで表示されます。

フォームエディターでのエディターツール

エディターツールウィンドウは、複合のフォームオブジェクトを迅速に作成することができ、**新規フォーム**ウィザードと**オブジェクトを挿入**のメニューに対して重要な補足ツールです。**エディターツール**ウィンドウを表示するためには、**フォーム**タブの**メイン**グループの**エディターツール**ボタンをクリックします。



新規フォームウィザードでは、フォームオブジェクトのセットが自動的に一括で作成されました。これと同じように追加で作成したい場合には、エディターツリーの該当するノードを右クリックします。



エディターツリーの下にあるツールバーには、ノードが選択された時に挿入可能なオブジェクトの選択肢が表示されます。また、ノードを右クリックして、これらの選択肢のリストを表示させることもできます。

ノードによって、次の選択肢が利用可能です。

- **入力**

  入力フィールド、**チェックボックス**、**コンボボックス**、または**ファイルインポート**オブジェクトが、次のように挿入されます。

  - エディターツールで選択されたノードを**ソース**として使う**入力フィールド**が挿入されます。挿入可能な場合、それと一緒に**テキストラベル**と**単位**オブジェクトも挿入されます。
  - 選択されたノードを**ソース**として使う**チェックボックス**が挿入されます。
  - エディターツールで選択されたノードを**ソース**として使う**コンボボックス**が挿入されます。このとき、そのノードにおけるリストと一致した選択リストが自動的に作成されます。**データアクセス**（ 89 ページ参照）を使用して、対応するノードをエディターツリーで挿入可能にした場合にのみ、この選択が可能となります。
  - 選択されたノードを**ファイルの出力先**として使う**ファイルインポート**オブジェクトが挿入されます。

- **出力**
  - 挿入可能な場合、**データ表示**オブジェクトが**テキストラベル**と一緒に挿入されます。
  - 選択されたノードが**テーブル**である場合、**結果テーブル**オブジェクトが挿入されます。
- **ボタン**
  - 選択されたノードを実行するコマンドシーケンスが設定された**ボタン**オブジェクトが挿入されます。
- **グラフィックス**
  - 選択されたノードを**初期グラフィックスコンテンツのソース**として使う**グラフィックス**オブジェクトが挿入されます。
- **ノード編集**
  - 対応するモデルツリーノードの**設定**ウィンドウに切り換わります。

エディターツールウィンドウは、メソッドエディターで作業する場合にも重要なツールです。つまり、メソッドエディターにおいて、エディターツリーのノードと関連したコードを生成するために使われます。詳細については、154 ページの「メソッドエディターでのエディターツール」を参照してください。

ボタン

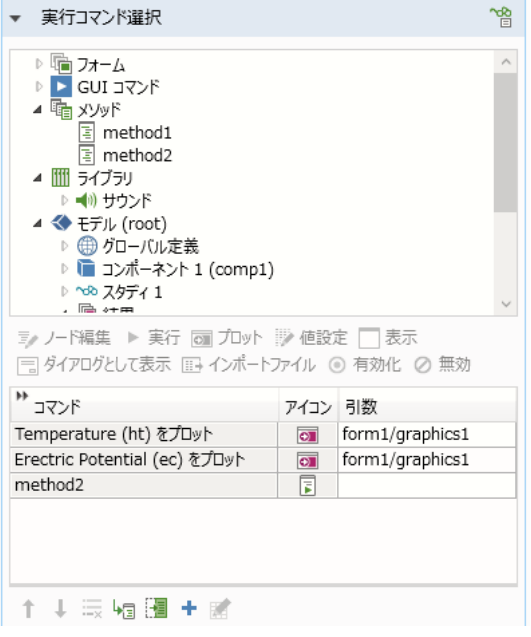
ボタンをクリックすることによって、そのボタンのコマンドシーケンスに設定されているアクションのイベントが起動されます。ボタンのための**設定**ウィンドウの主要なセクションでは、以下のことが可能です。

- ボタンのフォームオブジェクトの**名前**を編集する。
- ボタンに表示する**テキスト**を編集する。
- ボタンのデフォルトの画像の代わりに**アイコン**を設定する。
- ボタンの**サイズ**を**大**から**小**に変更する。

- ボタンの上にマウスが乗ったときに表示されるテキストを**ツールチップ**として追加設定する。
- **キーボードショートカット**を追加するには、その入力フィールドをクリックし、Shift、Ctrl、Alt といった修飾キーを押しながら他のキーを押します。Alt は少なくとも一つの追加修飾キーを伴わなければなりません。

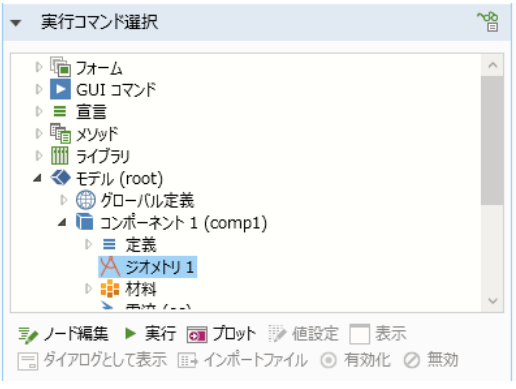
実行コマンドの選択

実行コマンド選択のセクションでは、ボタンクリックのイベントと関連するアクションを制御します。下図には、四つのコマンドの一連のシーケンスを起動させるボタンの**設定**ウィンドウを示しています。



メニュー、リボン、またはツールバー項目の**設定**ウィンドウにも、**実行コマンド選択**セクションがあり、この章で説明される機能が当てはまります。メニュー、リボン、およびツールバー項目を利用するためのさらに詳細な情報は、68 ページの「グラフィックスツールバー」、111 ページの「メインウィンドウ」、265 ページの「テーブル」、および 274 ページの「ツールバー」を参照してください。

実行コマンド選択セクションにおけるシーケンスコマンドを定義するには、エディターツリーのノードを選択します。そして、ツリーの下にある有効なボタンの一つをクリックするか、またはノードを右クリックしてコマンドを選択します。次の図では、**ジオメトリ**（Geometry ）のノードが選択され、選択可能なコマンドである**実行**と**プロット**のボタンがハイライト表示になっています。ジオメトリを作成するためのコマンドをコマンドシーケンスに追加するには、**実行**ボタンをクリックします。ジオメトリの作成を実行してからプロットするコマンドを追加するには、**プロット**ボタンをクリックします。選択肢の**ノード編集**をクリックすると、対応したモデルツリーまたはアプリケーションツリーのノード選択画面に切り換わります。



ジオメトリ（Geometry）**をプロット**コマンドの前に、**実行**をクリックして**ビルド ジオメトリ**（Geometry）を置く必要はありません。**ジオメトリ**（Geometry）**をプロット**コマンドは、最初にジオメトリの作成を実行してからそのジオメトリをプロットします。同様に、**メッシュ**（Mesh）**をプロット**コマンドは、最初にメッシュの作成を実行してからそのメッシュをプロットします。

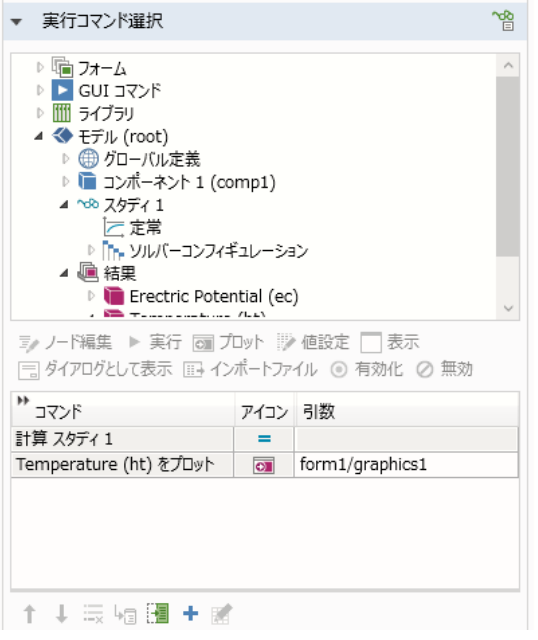
ツリーノードに対して適用可能な特定のコマンドアイコンのみが、選択のためにハイライト表示されます。以下は、ノードによって選択可能となるコマンドアイコンのリストです。

- **実行**
- **プロット**
- **値設定**
- **表示**
- **ダイアログとして表示**
- **インポートファイル**

- **有効**
- **無効**

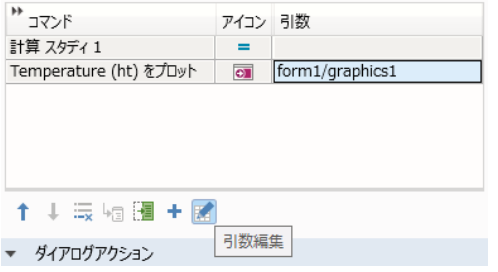
さまざまなプロットコマンドのようなコマンドには、引数の設定が必要です。例えば、プロットコマンドの引数には、そのプロット画像を表示させるグラフィックスオブジェクトを定義します。

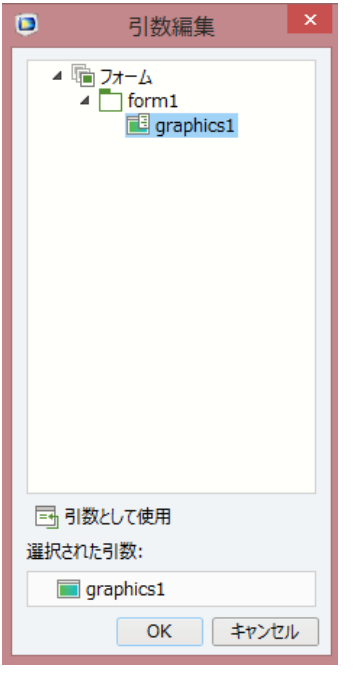
下図の例では、**新規フォーム**ウィザードによって作られた**計算**ボタンの**設定**ウィンドウのコマンドシーケンスを示しています。下図のボタンでは、コマンドシーケンスとして二つのコマンドがあります：**計算 スタディ1**（Study 1）と**Temperature をプロット**です。



Temperature をプロットのコマンドには、一つの引数 graphics1 が定義されています。

入力する引数の追加や編集をするには、下図に示すように、コマンドシーケンスの下側にある**引数編集**のボタンをクリックします。





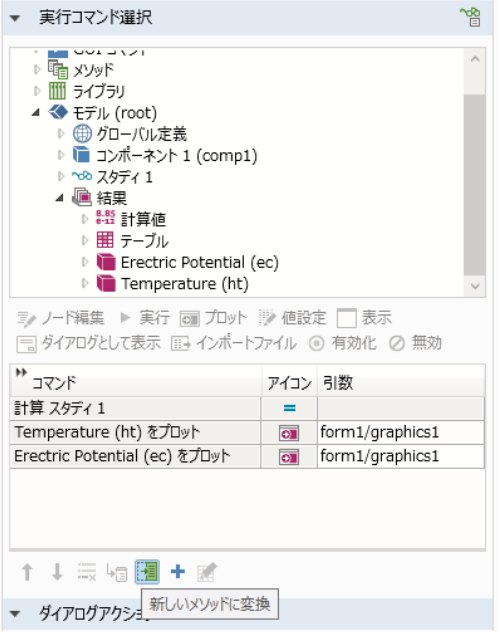
フォームを特定してグラフィックスオブジェクトを参照するには、/form1/graphics2、/form3/graphics1 などのような記述形式が使われます。

例えば graphics1 のように特定のフォームを指定しない場合には、そのボタンが配置されているフォームという定義になります。

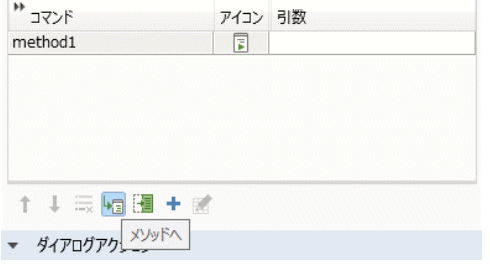
コマンドのシーケンスの順番や内容を変更する場合には、コマンドシーケンステーブルの下側にある**上へ移動**、**下へ移動**、および**削除**のボタンを利用します。

コマンドシーケンスのメソッドへの変換

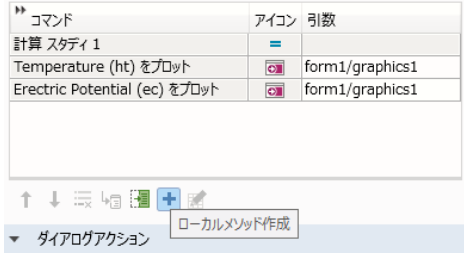
新しいメソッドに変換をクリックすることによって、一連のシーケンスコマンドの設定は自動的に新しいメソッドに変換され、さらにメソッドエディターの中で編集できるようになります。



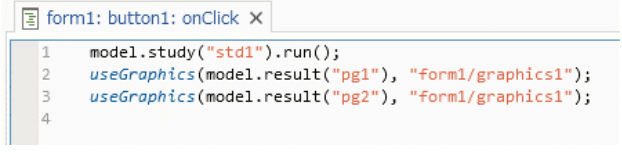
メソッドへをクリックすると、新規メソッドが開かれます。



ローカルメソッド作成をクリックすると、そのフォームオブジェクトに対するローカルなメソッドが作成されます。下図に、これらのオプションを示しています。



メソッドには、下図に示しているように、コマンドシーケンスに設定されたコマンドに対応する組み込みメソッドが呼び出されて表示されます。



この例の最初の行:

model . study (“std1”) . run ( )

は、最初のスタディ std1 に対応したモデルツリーノードが実行されます（最初のスタディノードは、ユーザによって変更されない限りは**スタディ1**（Study 1）という名前です）。2 行目と 3 行目:

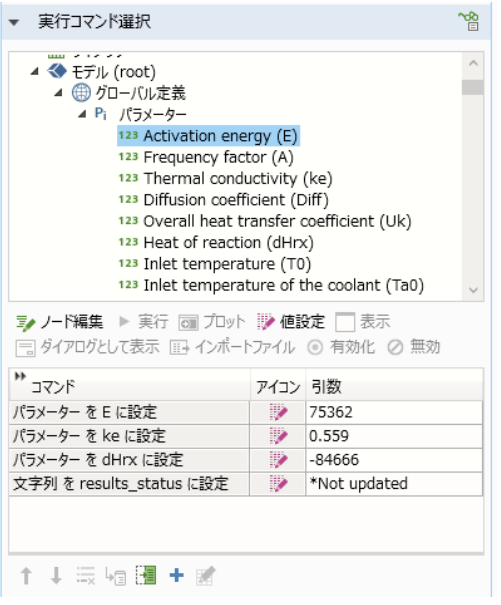
useGraphics (model . result ("pg2") ,"form1 / graphics1") ;
useGraphics (model . result ("pg1") , "form1 / graphics2") ;

では、グループ pg1 と pg2 の各プロットに対応したプロットを表示するために、組み込みメソッド useGraphics が使われています。この例では、プロットは、二つの異なったグラフィックスオブジェクト graphics1 と graphics2 の各々にプロット表示されます。メソッドに関する詳細については、149 ページの「メソッドエディター」を参照してください。

パラメータと変数の設定値

値設定コマンドでは、**パラメータ**、**変数**、および**宣言**ノードで選択可能なパラメータと変数の値を設定することができます。さらに、**値設定**では、**データアクセス**（89 ページ参照）によって利用できるプロパティの値を設定することができます。

下図には、パラメータのセットと一つの文字列変数を初期設定している場合を示しています。

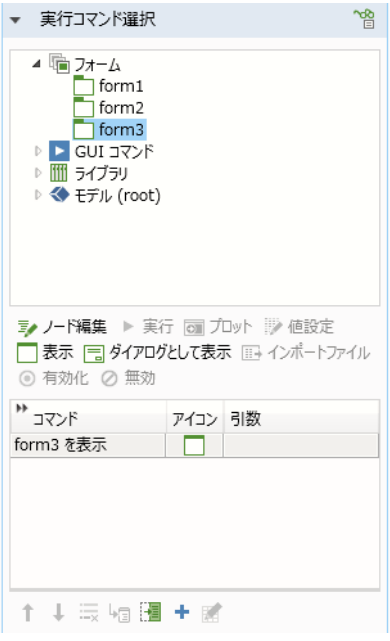


メソッドでどのように同じ操作のシーケンスを実現するかを学ぶためには、コマンドテーブルの下にある**新しいメソッドに変換**ボタンをクリックしてみてください。

フォーム表示の変更

フォームのボタンを、新しいフォームを表示するために使用することもできます。この方法には二つあります。一つ目は、新しいフォームをオリジナルのフォームに置き換える**表示**コマンドを使う方法です。二つ目は**ダイアログとして表示**コマンドを使う方法です。この場合には、現在のフォーム上でダイアログボックスとして新しいフォームが表示され、通常はそこにユーザへの入力要求があります。

表示コマンドは、**実行コマンド選択**のセクションで選択します。下図には、ボタンのコマンドシーケンスに**form3 を表示**のコマンドが設定されている場合を示しています。



このコマンドは、フォーム(form3)がボタンに関連付けられており、ボタンがクリックされたときにその指定フォーム(form3)がユーザに対して表示されます。

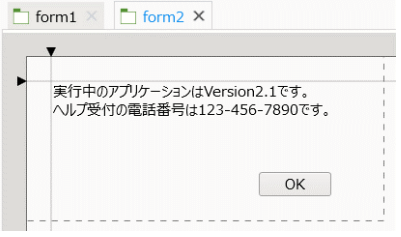
ダイアログボックスとしてフォームを表示

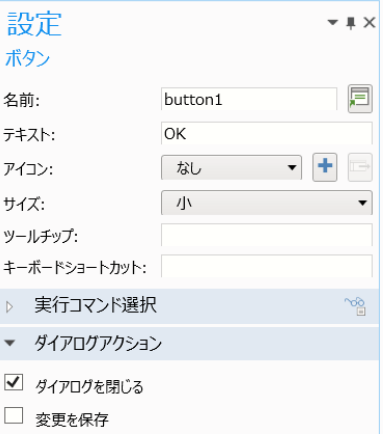
ダイアログとして表示コマンドを使うためには、**実行コマンド選択**セクションの設定の中で表示したいフォームを選択します。

下図には、**ダイアログとして form2 を表示**のコマンドが設定されているボタンの設定例を示しています。

この設定でアプリケーションを実行してボタンをクリックすると、以下のように **form2** として作成したダイアログボックスが表示されます。

この例では、下図に示すように、**form2** ウィンドウにテキストラベルオブジェクトと **OK** ボタンが配置されています。





設定ウィンドウの**ダイアログアクション**のセクションには、以下の二つのチェックボックスがあります。

- **ダイアログを閉じる**
- **変更を保存**

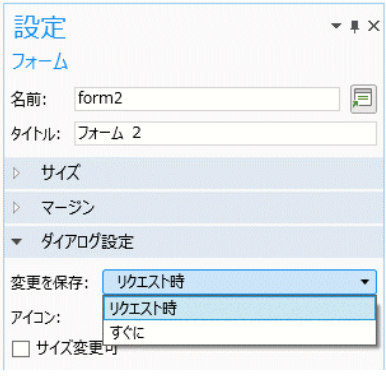
上図の例では、**ダイアログを閉じる**のチェックボックスが選択されています。この設定により、**OK** ボタンがクリックされると **form2** ウィンドウが閉じられます。**form2** にはユーザ入力が実装されていないので、**変更を保存**のチェックボックスを選択する必要はありません。

典型的なダイアログボックスボタン、およびそれらに関するダイアログアクションを下表に示します。

ボタン	ダイアログアクション
OK	ダイアログを閉じて変更内容を保存する
キャンセル	ダイアログを閉じる
適用	変更内容を保存する

アプリケーション実行中にダイアログボックスが表示されている間は、それが閉じられるまで、それ以外の全てのユーザ対話はブロックされます。

ダイアログボックスへの入力データをいつ保存するかを制御するために、フォームの**設定**ウィンドウの**ダイアログ設定**セクションにある**変更を保存**のリストを選択します。データを**リクエスト時**に保存するのか、または**すぐに**保存するのかを選択することができます。



外観

ボタンの**設定**ウィンドウの**外観**セクションには、ボタンオブジェクトの状態を制御する設定だけでなく、フォントの設定も含まれています。

フォームオブジェクトの有効化と可視状態の変更

ボタンオブジェクトを**可視**または**有効化**にするかどうかは、サブセクションの**状態**にあるチェックボックスで制御します。同様の設定がほとんどのフォームオブジェクトの**外観**セクションにありますが、追加オプションのあるものもあります。例えば、入力フィールドオブジェクトが該当します。

可視のチェックボックスがクリアされたボタンや他のフォームオブジェクトでは、実行中のアプリケーションのユーザインタフェースで表示されません。**有効化**のチェックボックスがクリアされたフォームオブジェクトでは、無効にされ、表示はされますが「グレーアウト」表示となります。また、フォームオブジェクトの状態は、組み込みメソッドを使用して制御することができます。例えば、ブーリアン変数 enabled_or_disabled は、**名前**が button3 のボタンの有効/無効状態を決定するために使用することを前提としています。この場合、以下のようにそのボタンの状態を制御することができます。

```
setFormObjectEnabled ( "button3" , enabled_or_disabled ) ;
```

同様に、以下の呼び出しでは、そのボタンがユーザに表示されているか否かをブーリアン変数 visible_or_not で制御しています。

```
setFormObjectVisible ( "button3" , visible_or_not ) ;
```

詳細については、308 ページの「GUI 関連のメソッド」と *Application Programming Guide* を参照してください。

グラフィックス

各**グラフィックス**オブジェクトは、それが作成された段階で graphics1、graphics2 などのデフォルト名が設定されています。これらの名前は、ボタンやメニュー項目のコマンドシーケンスとメソッドの中で、グラフィックスオブジェクトを参照するために使われます。特定のフォームのグラフィックスオブジェクトを参照する場合に使われる記述形式は: /form1/graphics2、/form3/graphics1 などです。

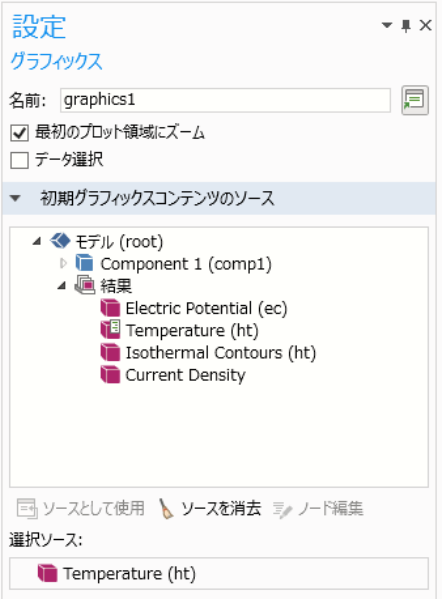
初期グラフィックスコンテンツのソース選択

グラフィックスオブジェクトの**設定**ウィンドウの**初期グラフィックスコンテンツのソース**セクションで、デフォルト表示されるプロットグループまたはアニメーションを設定します。

その設定には、**ソースとして使用**をクリックするか、ツリーのノードをダブルクリックします。

設定したプロットグループに表示可能なモデルのソリューション結果が存在していれば、アプリケーションが実行開始された段階でそれがグラフィックスに表示されます。

下図では、グラフィックスオブジェクトの**設定**ウィンドウで、**Temperature** のプロットがソースとして選択されています。



結果（Results）のプロットノードに加えて、**選択**（Selection）、**ジオメトリ**（Geometry）、**メッシュ**（Mesh）を**選択ソース**として設定することもできます。

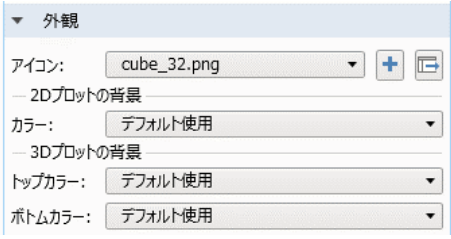
最初のプロット領域にズームのチェックボックスを選択すると、グラフィックスキャンバスに表示される最初のプロットがモデル全体（ズーム範囲）となるように表示されます。このアクションは、グラフィックスコンテンツがグラフィックオブジェクトに送信される最初の一度だけトリガされます。

データ選択のチェックボックスを選択すると、グラフィックスオブジェクトがインタラクティブに利用可能となります。例えば、特定のポイントでプロットをクリックした際に、その座標での温度の数値を取得することができるようになります。詳細は、77 ページの「データピッキング」を参照してください。

外観

グラフィックスオブジェクトの**設定**ウィンドウの**外観**セクションには、以下の設定項目があります。

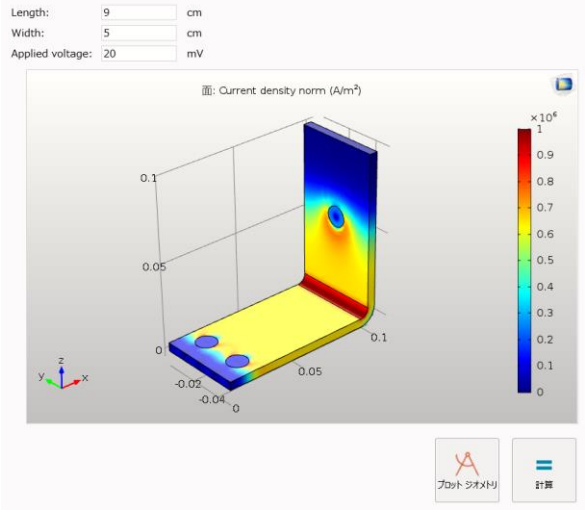
- グラフィックスオブジェクトの右上角に表示するロゴ画像などの**アイコン**を設定
- 2D プロットの背景の**カラー**を設定
- 3D プロットの背景のカラーを均一またはグラデーションにするため、**トップカラー**と**ボトムカラー**を設定



また、**状態**というサブセクション（上図に示されていません）では、グラフィックスオブジェクトの可視および有効化の状態の設定ができます。詳細については、62 ページの「フォームオブジェクトの有効化と可視状態の変更」を参照してください。

下図は、背景の**トップカラー**が白、**ボトムカラー**がグレーに設定されたアプリケーションです。

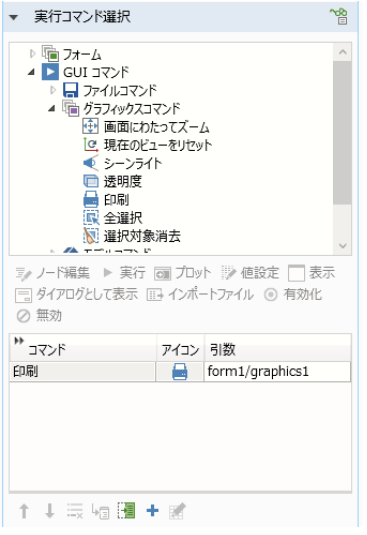
さらに、標準のプロットツールバーも表示しない設定になっています。



グラフィックスコマンド

例えば、ボタンのコマンドシーケンスに使われるエディターツリーの**グラフィックスコマンド**フォルダには、グラフィックスオブジェクトの処理や変更をするコマンドが含まれています。

下図では、コマンドシーケンスにグラフィックスオブジェクトを印刷するコマンドが設定されています。



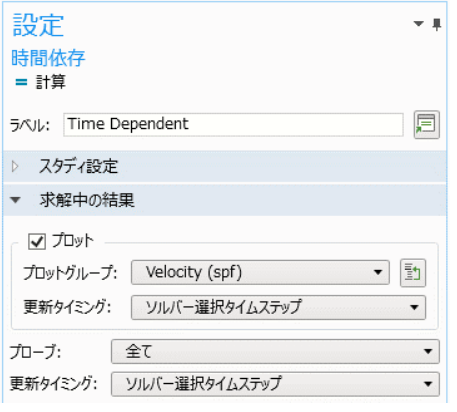
利用可能な**グラフィックスコマンド**を以下に示します:

- **画面にわたってズーム**
  - そのモデル全体を画面内で最大となるように表示します。
- **現在のビューをリセット**
  - 現在アクティブなビューを、アプリケーションが実行開始されたときの状態にリセットします。71 ページの「ビュー」を参照してください 。
- **シーンライト**
  - シーンライトのオンとオフを切り換えます。
- **透明度**
  - 透明度の設定のオンとオフを切り換えます。
- **印刷**
  - グラフィックスオブジェクトの表示内容を印刷します。
- **全選択**
  - 全てのオブジェクトを選択します。
- **選択対象消去**
  - 全てのオブジェクトの選択をクリアします。

標準のグラフィックスツールバーには、**画面にわたってズーム**、**現在のビューをリセット**、**シーンライト**、**透明度**、および**印刷**のコマンドに対応したツールバーボタンがあります。次のページの「グラフィックスツールバー」を参照してください。

求解中のプロット

収束状況をモニターするために、求解中に結果をプロットすることができます。この例では、**求解中の結果**の中の**プロット**の選択が有効となっていることが前提です。この選択は、下図に示すように、モデルツリーの**スタディ**(Study)ノードの**設定**ウィンドウの中で設定します。



組み込みの sleep メソッドを呼び出すメソッドを含めることによって、別のタイプのグラフィックスの表示に切り換える前に簡単にグラフィックス情報を表示することができます。下図のように、コマンドシーケンスでプロットコマンドの次にそれを挿入します。

この例では、sleep_a_bit メソッドに 1 行のコードが含まれています。

```
sleep ( 1000 ) ;  //  sleep for 1000 ms
```

sleep メソッドの詳細については、313 ページの「スリープ」を参照してください。

上記のコマンドシーケンスでは、**Velocity をプロット**のコマンドは**計算 Study** コマンドの前となっています。これによって、求解中にグラフィックスオブジェクトは Velocity のプロットを表示します。

複数のグラフィックスオブジェクトの利用

アプリケーションが実行されるプラットフォームにおいて、グラフィックスのハードウェア制限の可能性があるならば、使用するグラフィックスオブジェクトの数をできるだけ少なくするべきです。これが、アプリケーションの移植性を極力保証することになります。さらに、ウェブブラウザーでアプリケーションを実行する場合には、使うことができるグラフィックスオブジェクトの数に対する制限が追加されます。ハードウェア、オペレーティングシステム、およびウェブブラウザーの組み合わせによって、さまざまな制限があります。

同じ名前のグラフィックスが各々別のフォームに存在しているというような状況では、二つの異なるグラフィックスオブジェクトとしてカウントされます。例えば、form1/graphics1 と form2/graphics2 は、二つの異なるグラフィックスオブジェクトを表しています。さらに、一つのグラフィックスオブジェクトが一つのサブフォーム（ 235 ページの「フォーム」参照）に使われている場合、そのサブフォームは一つのグラフィックスオブジェクトとして別にカウントされます。

アプリケーションで多くのさまざまなプロットを表示するためには、例えば、ボタン、トグルボタン、ラジオボタンなどを作成し、サブフォームを使わないでフォーム上の同じ一つのグラフィックスオブジェクトに単純にプロットします。

もし、プロットを変更するメソッドを作る必要がある場合には、useGraphics メソッドを利用します。メソッドの記述についての詳細は、149 ページの「メソッドエディター」を参照してください。以下のコード例に示すように、ブーリアン変数の値に基づき、同じグラフィックスオブジェクトを再利用することによってプロットグループを切り換えます。

```
if ( my_boolean ) {
   useGraphics ( model . result ( "pg1" ) , "form1 / graphics1" ) ;
   my_boolean = ! my_boolean ;  //  logical NOT to change between true and false
} else {
   useGraphics ( model . result ( "pg2" ) ,"form1 / graphics1" ) ;
   my_boolean = ! my_boolean ;
}
```

グラフィックスオブジェクトの内容のクリア

次のような useGraphics メソッドの呼び出しによって、グラフィックスオブジェクトの内容をクリアすることができます。

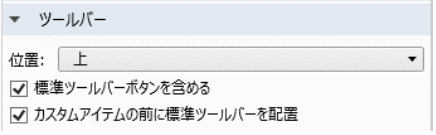
```
useGraphics ( null ,  "/form1/graphics1" )
```

ここでは、フォーム 1 のグラフィックスオブジェクト graphics1 の内容をクリアしています。

グラフィックスツールバー

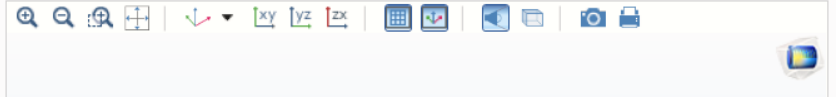
初期グラフィックスコンテンツのソースで選択されるツリーノードのタイプによって、表示されるツールバーのタイプが決められます。このツールバーは、空間次元によって異なり、参照されるソースが**ジオメトリ**、**メッシュ**、**選択**、または**プロットグループ**のいずれのノードかによっても異なります。例えば、**プロットグループ**ノードでは**レジェンド表示**のボタンが追加で表示されます。

グラフィックスオブジェクトの**設定**ウィンドウの**ツールバー**セクションでは、その位置（**上、下、左、右**）だけでなくグラフィックスツールバーを含めるかどうかを選択することができます。



ジオメトリとメッシュのためのグラフィックスツールバー

下図は、3D モデルにおいて、**ジオメトリ**または**メッシュ**ノードが**初期グラフィックスコンテンツのソース**として使用される場合に表示されるような、標準のグラフィックスツールバーを示しています。



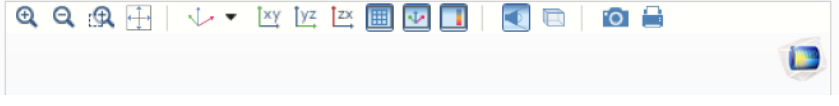
選択のためのグラフィックスツールバー

初期グラフィックスコンテンツのソースが**明示的**選択に設定されている場合、グラフィックスツールバーには三つの追加項目が含まれています： **選択対象をズーム**、**ボックス選択**、**ボックス選択解除**。これを下図に示しています（**ボックス選択解除**は**ボックス選択**の右隣にあります）。

選択についての詳細は、74 ページの「選択」を参照してください。

プロットグループのためのグラフィックスツールバー

下図は、**3D プロットグループ**ノードが**初期グラフィックスコンテンツのソース**として使用される場合に表示されるような、標準のグラフィックスツールバーを示しています。

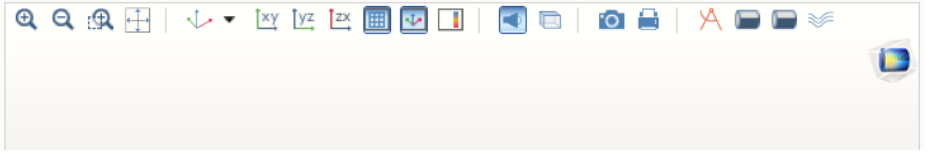


カスタムのグラフィックスツールバーボタン

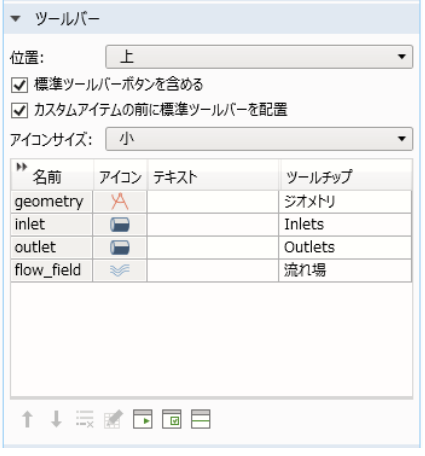
ツールバーのセクションでは、グラフィックスツールバーにカスタムボタンも追加できます。

カスタムツールバーボタン（アイテム）の追加や削除には、テーブルの下のボタンを使います。ツールバーボタンを上または下に移動させることができ、**セパレーター**を追加し、ボタンの**編集**も可能です。下図

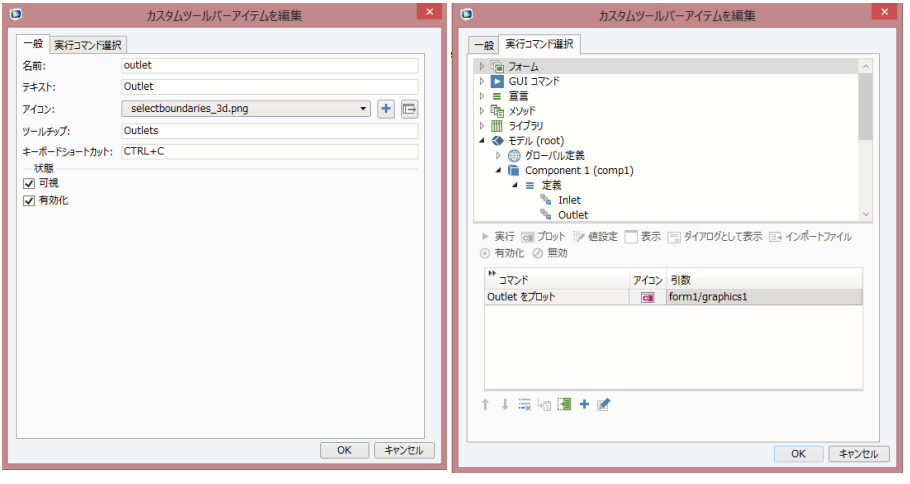
は、結果のための標準のグラフィックスツールバーの右側に四つのカスタムボタンを追加した場合を示しています。



下図の設定には、グラフィックスツールバー項目に対応したテーブルが示されています。



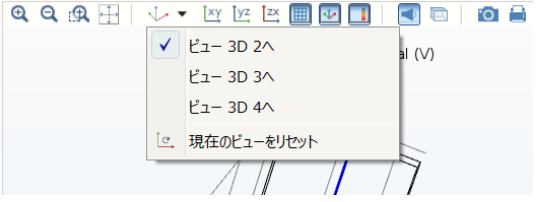
ツールバーアイテムのコマンドシーケンスを編集するには、テーブルの下側にある**編集**ボタンをクリックして**カスタムツールバーアイテムを編集**のダイアログボックスを開きます。



このダイアログボックスは、アイテムがトグルアイテムであるかどうかによって内容が二つまたは三つのタブに分割され、ボタンまたはツールバーアイテムの設定と類似した設定を備えています。詳細は、51 ページの「ボタン」および 274 ページの「ツールバー」を参照してください。

ビュー

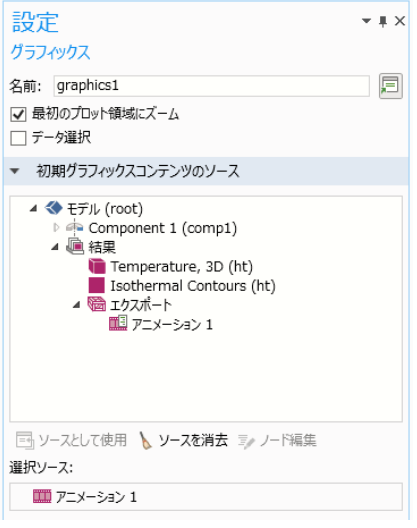
グラフィックスツールバーの**デフォルト 3D ビューへ**（3D グラフィックスの場合）のボタンをクリックすると、全ての表示可能なビューのメニューが表示されます。現在アクティブなビューは、チェックマークで示されます。



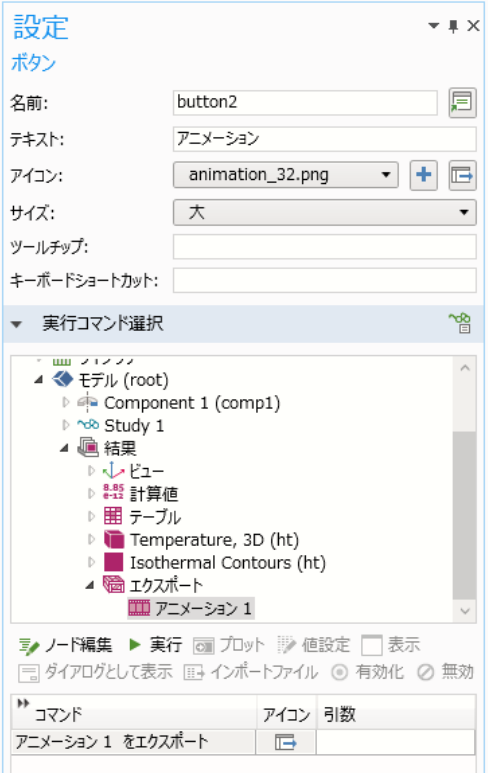
リストには、全てのビューの他に**現在のビューをリセット**という項目があります。それをクリックすると、現在のアクティブな表示からアプリケーションが実行された時の表示状態にリセットされます。

アニメーション

初期グラフィックスコンテンツのソースのツリーに表示される**結果** > **アニメーション**ノードを使って、アプリケーションの中でアニメーションを実行させることができます。



アニメーションを実行するには、新規フォームウィザードまたはエディターツールウィンドウを使用して、たとえば、**結果 > アニメーション**ノードを実行するボタンからコマンドを作成します。



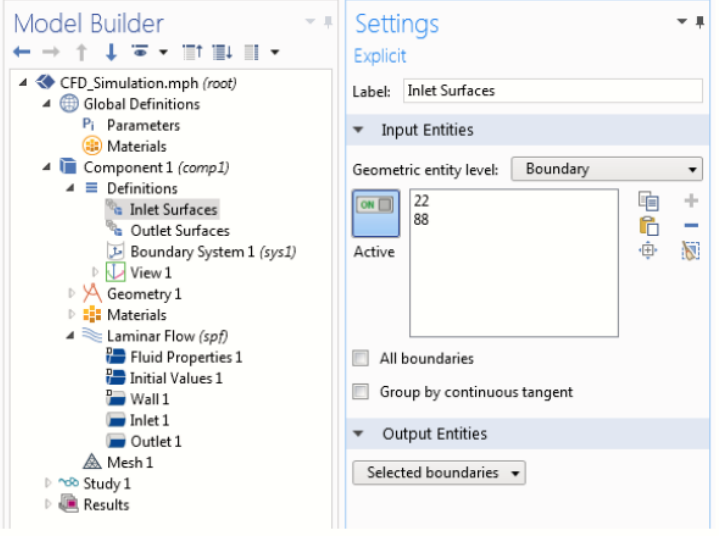
新規フォームウィザードまたはエディターツールを使用する場合、アニメーションボタンのデフォルトの外観は次のようになります。

選択

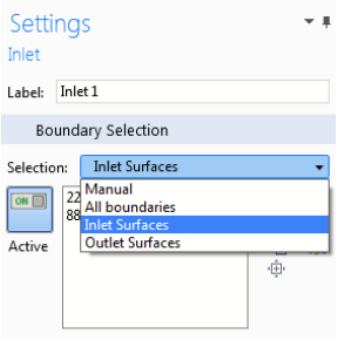
モデルビルダーでの選択

モデルビルダーにおいて、材料特性、境界条件、またはその他のモデル設定を定義する場合に、**選択**という呼び方でドメイン、境界、エッジ、またはポイントがグループ化されます。**コンポーネント**（Component）> **定義**（Definitions）ノードの下にサブノードを追加して、**選択**の種々のタイプを作成することができます。これらは、モデルのコンポーネント（Component）の中のどこででも利用することができます。

選択の使用方法の一例として、境界条件を選択することを考えてみましょう。ある境界条件に関連付ける境界を選択する場合、COMSOL デスクトップ環境のグラフィックスウィンドウでそれらの境界を直接クリックすることができます。これは、**マニュアル**選択と呼ばれるデフォルトのオプションです（下記参照）。それによって、それらの境界がその境界条件に対してローカルな選択が追加されます。その代わりに、名前付きの選択によってグローバルな選択を定義することができ、ドロップダウンリストから選択するだけでいくつかの異なる種類の境界条件を再利用することができます。下図は、二つの関係する境界（22 および 88）を持つ **Inlet Surfaces** という名前の**明示的**選択を示しています。



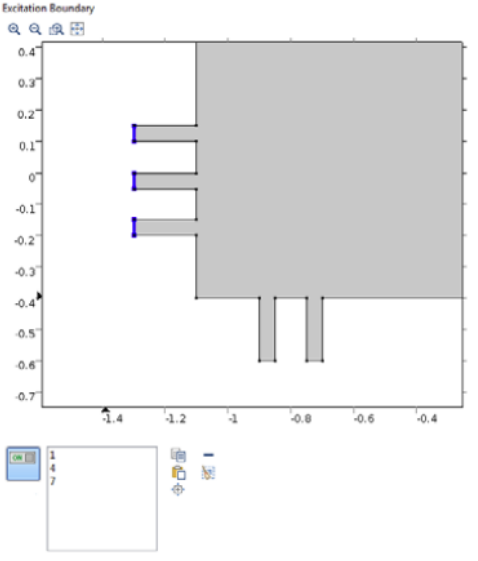
下図は、**Inlet Surfaces** 選択が使用されている**流入口**（**Inlet**）の境界条件の**設定**ウィンドウを示しています。また、この例では Outlet Surfaces の選択もあります。



また、便宜上、**マニュアル**（Manual）オプションに加えて、**全境界**（All boundaries）のショートカットもあります。

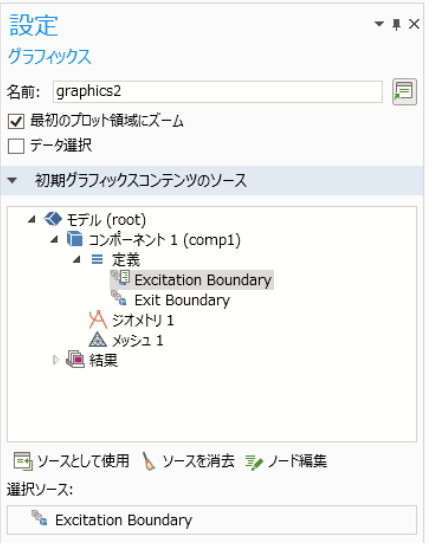
アプリケーションビルダーでの選択

明示的選択のタイプは、エンティティの番号によってドメイン、境界、エッジ、またはポイントをグループ化することができ、アプリケーションビルダーで使われる最も容易に利用可能な選択のタイプです。**選択入力**オブジェクトと**グラフィックス**オブジェクトを持つ**明示的**選択に属するエンティティを、アプリケーションのユーザに対話的に変更させることができます。次の図の例では、埋め込まれているモデルが**明示的**選択によって定義された境界条件を持っています。この**選択入力**オブジェクトと**グラフィックス**オブジェクトによって、入力波で励起される境界をユーザに選択させています。



ここで、ユーザは、その**グラフィックス**オブジェクトのグラフィックスウィンドウで直接クリックするか、あるいは**選択入力**オブジェクトの境界番号リストでジオメトリのエンティティ番号を追加することによって、所望の境界を選ぶことができます。

境界をクリックして直接選択することができるようにするためには、次の図に示すように、グラフィックスオブジェクトを境界のグループ化に使われている**明示的**選択にリンクさせます。**明示的**選択を選び、**ソースとして使用**をクリックします。次の図では、二つの**明示的**選択 **Excitation Boundary** と **Exit Boundary** が存在し、グラフィックスオブジェクト graphics2 は選択ソースとして **Excitation Boundary** にリンクしています。



このように、グラフィックスオブジェクトが選択と直接リンクされている時には、グラフィックスオブジェクトはそのジオメトリを表示し、ユーザはそこで対話的に境界をクリックすることができます。それによって、対応する選択に境界が追加（または削除）されます。

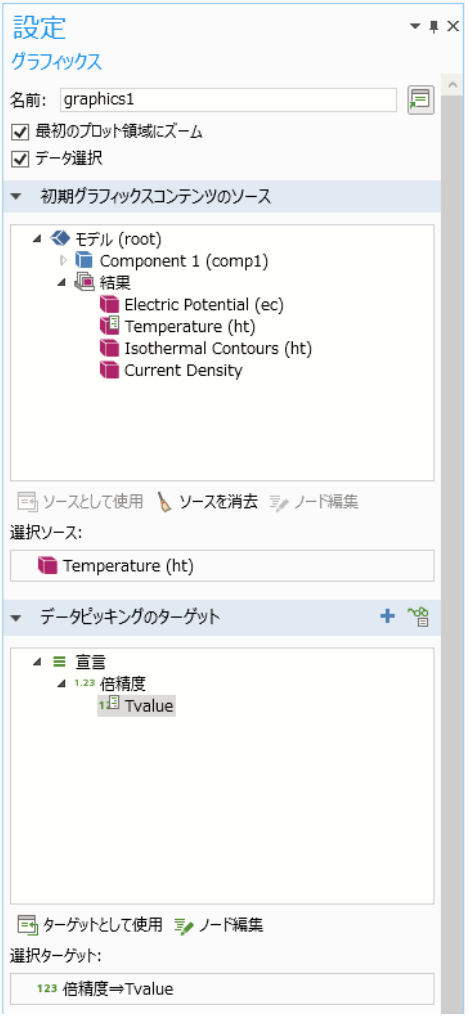
選択入力オブジェクトを明示的選択にリンクさせることによって、番号による選択が可能となります。詳細については、256 ページの「選択入力」を参照してください。

エディターツールウィンドウでは、**明示的**な選択にリンクされている**グラフィックス**オブジェクトまたは**選択入力**オブジェクトを簡単に追加できます。これらのオプションを取得するには、エディターツリーの**明示的**（Explicit）な選択ノードを右クリックします。

明示的選択によってグローバル**イベント**を起動させるようにすることができます。これによって、ユーザがジオメトリオブジェクト、ドメイン、サーフェス、エッジ、またはポイントをクリックした際に、コマンドシーケンスまたはメソッドを実行させることができます。グローバルイベントの使用方法の詳細については、118 ページの「イベント」と 121 ページの「データ変更イベントのソース」を参照してください。

データピッキング

グラフィックオブジェクトの**設定**ウィンドウで、**データ選択**のチェックボックスを選択すると、グラフィックスオブジェクトがインタラクティブに利用可能となります。例えば、特定のポイントでプロットをクリックした際に、その座標での温度の数値を取得することができるようになります。次の図の**データピッキングのターゲット**のセクションでは、スカラ倍精度変数 Tvalue が選択されています。この変数は**宣言**ノードの下で宣言されています。実行中のアプリケーションでは、ポインタ位置の温度値がこの変数 Tvalue に格納されます。



データピッキングのターゲットが1Dの倍精度配列の場合、格納された値は代わりに、クリックされた位置のx、y（2D）またはx、y、z座標に対応します。

データピッキングのターゲットは、次のいずれかです。

- スカラー倍精度変数
- 1D 倍精度配列
- ドメインポイントプローブ

- 境界点ポイントプローブ
- グラフィックデータ宣言

グラフィックスデータ宣言の詳細は、145ページの「グラフィックスデータ」を参照してください。

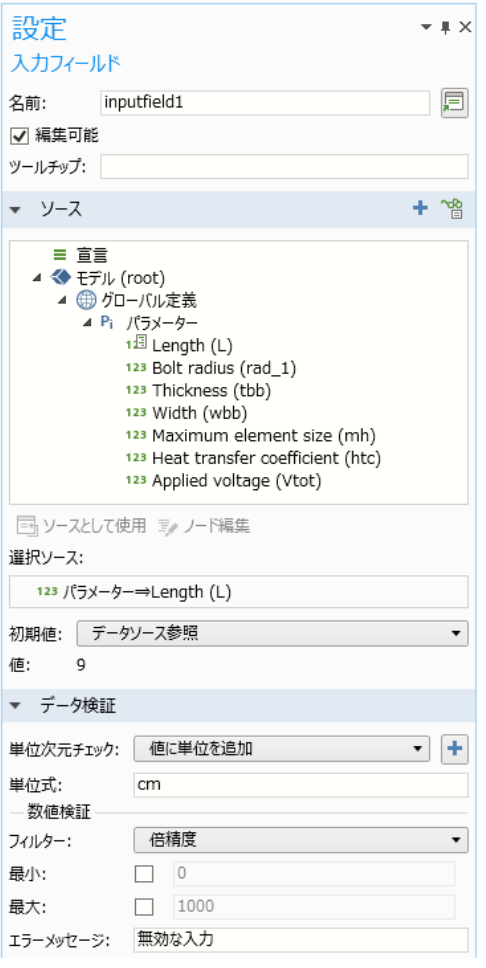
入力フィールド

入力フィールドは、パラメータや変数の値を変更させるために利用できます。新規フォームウィザードでパラメータまたは変数を一つ選択すると、同時に三つのフォームオブジェクトが作成されます。

- パラメータまたは変数の説明用のテキストラベルオブジェクト
- その値を入力するための入力フィールドオブジェクト
- その値に適用される測定単位がある場合は、単位オブジェクト

エディターツールウィンドウを使用してパラメータや変数を選択することによって、同時に三つのフォームオブジェクトが作成されます。

エディターツールウィンドウを使用しないと仮定した場合には、追加の入力フィールドを挿入するには、リボンにある**オブジェクトを挿入**のメニューを使用して**入力フィールド**を選択します。フォームエディターで、**ソース**セクションのツリーから特定のパラメータまたは変数を選択して入力フィールドにリンクさせ、**ソースとして使用**をクリックします。**設定**ウィンドウの**ソース**セクションでは、**初期値**を設定することもできます。下図に、入力フィールドの**設定**ウィンドウを示します。



入力フィールドでは、パラメータと変数に加えて、**ソース**として**情報**（Information）ノードを使用することができます。

初期値のデフォルト設定は、**データソース参照**となっています。この場合に入力フィールドに表示される初期値は、もし選択ソースがパラメータであるなら、モデルビルダーの**パラメータ**ノードの設定で定義されているパラメータ値と同じになります。**初期値**として**カスタム値**を選択した場合には、選択ソースとは異なる初期値を設定することができます。入力フィールドの設定ウィンドウの上部にある**編集可能**のチェックボックスのチェックがクリアされている場合、入力フィールドには**初期値**がアプリケーションによって表示されますが、ユーザがその値を直接変更することはできません。

カーソルを入力フィールドの上にホバリングした時に表示されるテキストは、**ツールチップ**に入力して設定します。

ソースセクションのヘッダ右側にある二つのボタンをアクセスすることによって、入力フィールドのソースとしてプロパティや変数を容易に追加することができます。

新規変数を作成しソースとして使用のボタンによって、**宣言**ノードの下に新しい変数を追加することができます。詳細については、125 ページでの「宣言」を参照してください。**モデルビルダへスイッチしデータアクセスをアクティベート**のボタンによって、次の章で説明しているように、低レベルのモデルプロパティにアクセスすることができます。**データアクセス**についての詳細は、89 ページの「フォームエディターでのデータアクセス」を参照してください。

データ検証

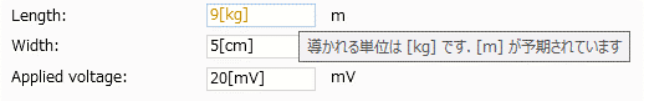
入力フィールドの**設定**ウィンドウの**データ検証**セクションでは、ユーザ入力の単位と入力値を検証させる設定を行います。

新規フォームウィザードで選択して作成された入力フィールドでは、適用可能な場合、**値に単位を追加**の設定が使われています。この設定は、ユーザがその入力フィールドに数値を入力することを前提としていますが、その後に続けて COMSOL の角括弧[ ] の単位表記を使った単位を入力すればその数が処理されるようになります。例えば、もし**単位式**が mm のときには、1[mm]を 0.1[cm]のように入力することができ、他の長さの単位でも同様です。互換性のないタイプの単位が入力された場合には、エラーメッセージが表示されます。1.23[mm]という式が定義されてソースとして使われているパラメータでは、付加される単位は mm 、編集フィールドに表示される初期値は 1.23 です。

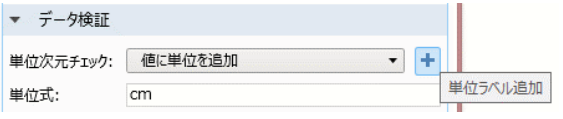
単位次元チェックのリストには、以下の選択肢があります。

- **なし**
- **物理量と互換**
- **単位式と互換**
- **値に単位を追加**（デフォルト）
- **単位セットから単位を追加**

アプリケーションのユーザが互換性のない単位を入力した場合、その値または式は警告のためにオレンジ色でハイライト表示されます。そのような互換性のない単位とは、**データ**設定で指定された単位に変換できないと判断される単位です。
物理量と互換、または**単位式と互換**を選択し、この機能を有効にして確かめてみてください。さらに、下図に示すように、ツールチップによって単位が不適当であることの説明が表示されることも確認してみましょう。



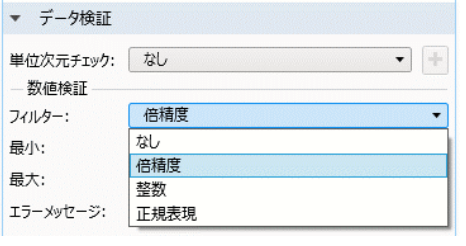
もし単位が不適当であり、これ以上のアプリケーションによるエラー制御が実行不能な場合には、入力された式による数値はデフォルトの単位に変換されます。上図の数値の場合は、9[kg] は 9[m]に変換されます。
単位次元チェックのリストの右側にある**単位ラベル追加**のボタンを利用することができます。



その入力フィールドのところに単位ラベルが置かれていない状態で、このボタンをクリックすることによって、入力フィールドの右側に単位ラベルが追加されます。
単位次元チェックの**なし**を選択した場合は、単位の適合性の検証はされません。

数値検証

値に単位を追加、**単位セットから単位を追加**、および**なし**を選択した場合には、入力数値の適合性を検証するためのフィルタを使うことができます。



単位次元チェックの選択を**なし**とした場合、**フィルタ**のリストには以下の選択肢があります。

- **なし**
- **倍精度**
- **整数**

- **正規表現**

単位次元チェックの選択を**値に単位を追加**と**単位セットから単位を追加**とした場合には、**フィルタ**のリストは**倍精度**と**整数**のみとなります。

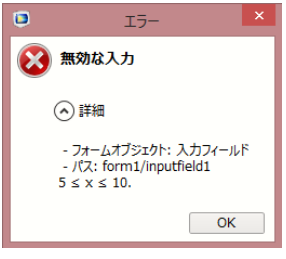
倍精度と**整数**を選択した場合は、**最小**と**最大**の設定値に基づいて入力値にフィルタがかけられます。入力値がこれらの設定範囲を超えている場合に、**エラーメッセージ**が表示されます。これらのフィールドにグローバルパラメータを使うことができます。グローバルパラメータを使用する場合、単位付き、あるいは単位のないパラメータを定義することができます。もし単位なしでグローバルパラメータを使用する場合、それらが**最小**値と**最大**値として使用される際に、これらのパラメータの数値だけが考慮されます。例えば、単位が cm の長さのパラメータ L の入力フィールドのデータ検証を検討してみます。ここで、グローバルパラメータ Lmax は**最大**値として使用されているとします。L の最大値を 15 cm としたい場合、パラメータ Lmax として適した値は 15（単位なし）、15[cm]、0.15[m]、150[mm]、などです。

単位セットから単位を追加のオプションでは、**最小**値と**最大**値は、単位セットで設定された単位の**初期値**に常に対応しています。単位セットの詳細については、138 ページの「単位セット」を参照してください。

単位次元チェックが**なし**に設定されている場合に利用可能となる**正規表現**のオプションでは、入力文字列にマッチする正規表現を利用することができるようになります。正規表現の詳細については、ダイナミックヘルプを参照してください。ウィンドウの右上隅にあるヘルプアイコンをクリックして、「正規表現」を検索してください。より高度な要件として、事実上、何種類かの入力フィールドの内容の検証は、入力フィールドの**設定**ウィンドウの**イベント**セクションを使用してメソッドを呼び出すことによって可能である点に留意してください。

エラーメッセージ

エラーメッセージに表示されるテキストは、カスタマイズすることができます。アプリケーションの開発やデバッグのときに、時々発生したエラーがどこなのかを推察するのが難しい場合があります。そのような時には、**アプリケーションをテスト**を実行中にデバッグ情報を下図のように追加表示させます。

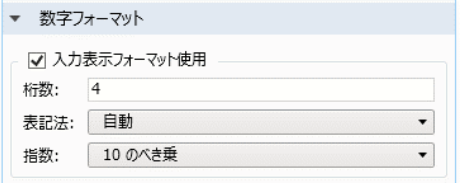


デバッグ情報は、一般に、フォームオブジェクトのタイプ、フォームオブジェクトへのパス、およびエラーの理由（例えば 5<=x<=10）から成ります。

アプリケーションを実行によって起動、または COMSOL サーバーによって起動したアプリケーションでは、このような特別なデバッグ情報は一切追加表示されません。

数字フォーマット

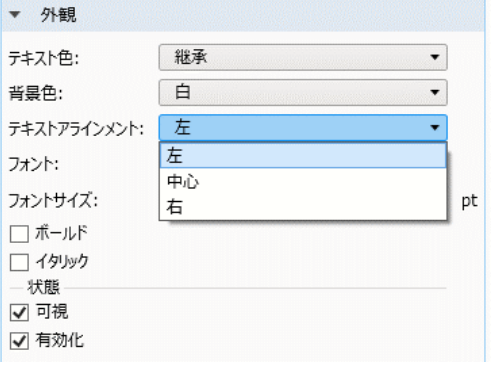
数字フォーマットのセクションには、**入力表示フォーマット使用**のチェックボックスがあります。これにチェックを入れると、**データ表示**オブジェクトの場合と同じように、このセクションで設定された表示形式が入力フィールドに適用されます。



詳細については、86 ページの「データ表示」を参照してください。

外観

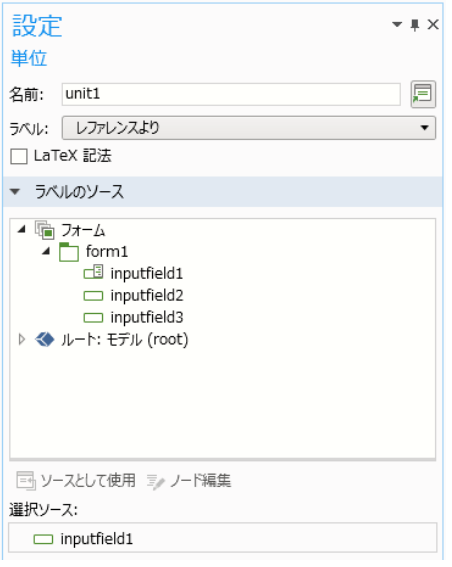
入力フィールドの**外観**セクションでは、色とフォントの設定の他に**テキストアラインメント**の設定があります。これによって、フィールド内のテキストの配置が**左**、**中心**、または**右**に調整可能です。



入力フィールドを**可視**または**有効化**にするかどうかは、**状態**のサブセクションにあるチェックボックスで指定できます。詳細については、62 ページの「フォームオブジェクトの有効化と可視状態の変更」を参照してください。

単位

単位オブジェクトの**設定**ウィンドウでは、単位を固定の文字列に設定するか、または入力フィールドなどのソースにリンクさせることができます。下図は、単位オブジェクトの**設定**ウィンドウを示しています。



新規フォームウィザードで入力フィールドを追加した時には、適用可能な単位があれば、単位オブジェクトが自動的に追加されます。デフォルトで、単位は Unicode 表示されます。代わりに、**LaTeX 記法**のチェックボックスにチェックを入れて、LaTeX 形式の表示を使うことができます。その場合、単位の表示は選択されたフォントに依存しません。

テキストラベル

テキストラベルオブジェクトは、単にフォームにテキストを表示するためのものです。新規フォームウィザードで入力フィールドを追加する時には、関連したパラメータや変数の説明用テキストとして、**テキストラベル**オブジェクトが自動的に追加されます。設定には、**複数行テキスト**のチェックボックスがあります。これがチェックされた場合には、**テキストを折畳み**のチェックボックスが追加設定として表示されて利用可能になります。

下図は、**テキストラベル**オブジェクトの設定ウィンドウです。（下図は、スケッチレイアウトモード時の場合です。）

テキストラベルを追加作成するには、リボンにある**オブジェクトを挿入**のメニューから**テキストラベル**を選択します。

データ表示

データ表示オブジェクトは、スカラーと配列の数値を表示するために使われます。関連する単位がある場合には、それも**データ表示**オブジェクトの一部として表示されます。

ソース

データ表示オブジェクトの**設定**ウィンドウで、**ソース**セクションのモデルツリーで所望のノードを選択します。そして、その下側にある**ソースとして使用**のボタンをクリックします。有効なパラメータ、変数、およびプロパティとしては、以下があります。

- グローバル評価や体積最大値ノードなどの**計算値**（Derived Values）ノードからの出力

- **宣言 > スカラー**、**配列 1D**、および**配列 2D** のノードの下で宣言された変数
- **データアクセス**ツールを使って利用可能となるプロパティ。89 ページの「フォームエディターでのデータアクセス」を参照してください。
- root ノードの下、および各スタディ（Study）ノードの下にある**情報**（Information）ノードの以下に示す変数のうちの 1 つ
  - **予想計算時間**
    これは、root ノードの**設定**ウィンドウの中で**予想**のフィールドに入力設定されている値です。
  - **前回の計算時間**（root ノードの下）
    これは、最後に計算されたスタディの計測時間です。
  - **前回の計算時間**（スタディ（Study）ノードの下）
    これは、そのスタディの最後の（前回の）計測時間です。

  最初にアプリケーションを実行開始した時点では、最後の計測時間はリセットされています。

データ表示オブジェクト生成のための新規フォームウィザードの利用

新規フォームウィザードの**入力/出力**タブにおいて、**計算値**（Derived Values）ノードだけが**データ表示**オブジェクトを生成します。一方、**宣言**ノードの下の変数や**モデルデータアクセス**によって利用可能となる定数は、**入力フィールド**オブジェクトを生成します。

計算値（Derived Values）ノードが選択される場合は、対応した**計算値**（Derived Values）ノードの変数に基づいた二つのフォームオブジェクトが作成されます:

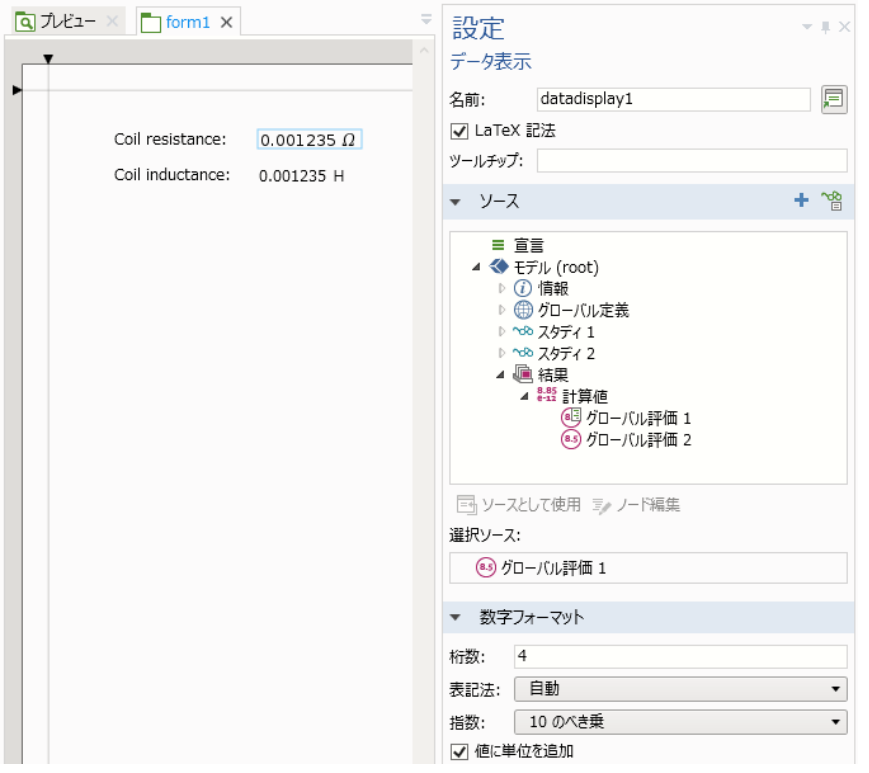
- その変数の**説明**のための**テキストラベル**オブジェクト
- その変数の値のための**データ表示**オブジェクト

これらのフォームオブジェクトの設定は、新規フォームウィザードの後に編集が可能です。データ表示オブジェクトを追加作成するには、リボンにある**オブジェクトを挿入**のメニューから**データ表示**を選択します。

数字フォーマット

数字フォーマットセクションでは、**桁数**、**表記法**、および**指数**を設定することができます。

下図は、変数 Coil resistance と Coil inductance のデータ表示オブジェクトの例です。単位が適用可能な場合、決められた表記形式の単位ラベルが、自動的にデータ表示オブジェクトの一部として表示されます。



メソッドの表記

データ表示オブジェクトの単位は、デフォルトでは Unicode 表記を使って表示されます。その代わりに、**LaTeX 記法**のチェックボックスにチェックを入れることによって、LaTeX を使った表記が可能です。その場合、単位の表示は選択されたフォントには依存しません。

配列と行列の表示フォーマットは、LaTeX 表記によってのみサポートされます。下図に示した 2D の倍精度配列（133 ページを参照）は、**LaTeX 記法**を選択した**データ表示**オブジェクトを使って表示されています。

$$\begin{bmatrix} 0 & 0 & 0.9 & 0.8 & 0.7 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0.5 & 0.7 & 1.5 & 0.8 & 0.6 & 0.3 & 0.2 & 0.1 & 0.1 & 0.1 \\ 0 & 0 & 0.9 & 0.8 & 0.7 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

ツールチップの設定にテキストを入力しておけば、データ表示オブジェクトをホバリングした時にそれを表示させることができます。

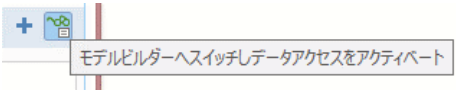
フォームエディターでのデータアクセス

フォームオブジェクトの多くのタイプの**設定**ウィンドウには、モデルツリーやモデルツリーの一部やアプリケーションツリーの一部を含んだツリー構造のノードを選択することが可能なセクションがあります。例えば、入力フィールドの**ソース**セクション、またはボタンの**実行コマンド選択**セクションです。モデルやアプリケーションのツリーには、デフォルトで利用可能な多くのプロパティが存在しています。というのも、モデルには数百またはさらに数千ものプロパティが含まれているかもしれないからです。これらを全て完全にリスト化するのは、かえって扱いにくいことになります。しかし、これらの「隠されている」プロパティは、**データアクセス**と呼ばれるテクニックによって、アプリケーションに利用することができるようになります。

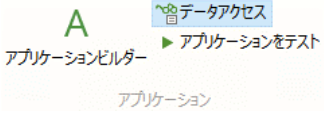
この章の以降では、入力フィールドとボタンに関する例を取り上げ、**データアクセス**の使い方を紹介しています。

入力フィールドのためのデータアクセス

パラメータや**変数**ノードの下のモデルツリーで定義されたパラメータや変数、および**宣言**ノードの下のアプリケーションツリーで定義された変数に、デフォルトで、入力フィールドをリンクさせることができます。モデルツリーノードのプロパティを追加してアクセスするためには、下図に示すように、入力フィールドの**設定**ウィンドウの**ソース**セクションのヘッダ部分にある**モデルビルダーへスイッチしデータアクセスをアクティベート**のボタンをクリックします。

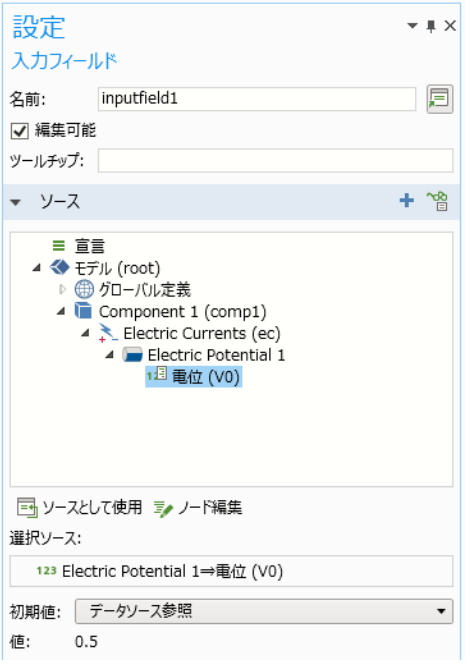


モデルビルダーの**ホーム**タブの**アプリケーション**グループからでも、それをアクセスすることができます。



その後、モデルツリーノードをクリックすると、個々の設定の隣にチェックボックスが出現します。下図では、**電位**の境界条件のためのチェックボックスが選択されています。

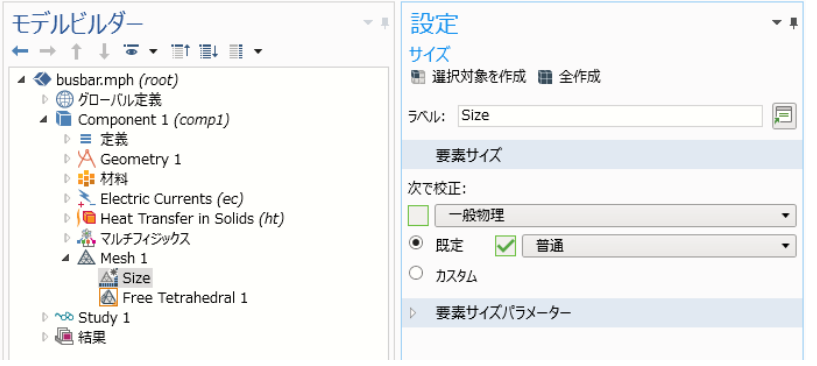
下図は、入力フィールドのための**設定**ウィンドウを示しています。先ほどの**電位**が、このフィールドに可能なソースのリストに含まれています。



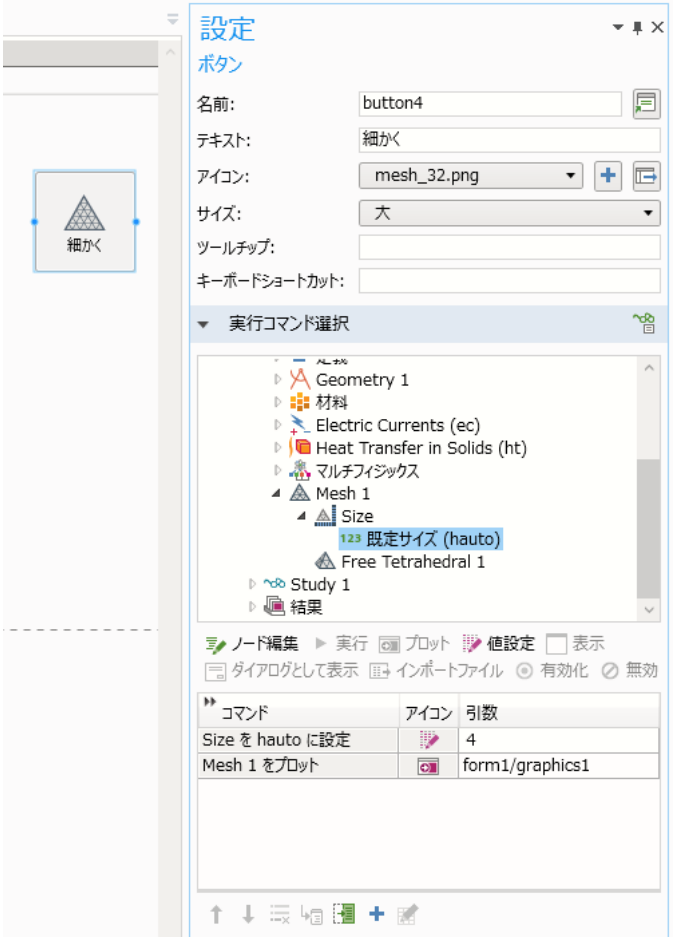
データアクセスのボタンへの利用

パラメータ、変数、またはモデルのプロパティの値を設定する用途のボタンに、データアクセスを利用することができます。例えば、既定のメッシュ要素サイズを設定する用途でボタンを作成する場合です。**メッシュ**ノードの**設定**ウィンドウで、**シーケンスタイプ**が**ユーザー制御メッシュ**に設定されている場合、次の図に示した設定が利用可能です。

この例では、**要素サイズ**の**既定**のプロパティが選択され、アプリケーションビルダー側で利用可能になっています。



下図では、ボタンの**設定**ウィンドウが示されています。このボタンは、**要素サイズ**の**既定**で**細かい**が選択されている設定を利用してメッシュを作成するために使われています。



この上の例において、**メッシュ**の**既定サイズ(hauto)** のプロパティの値を設定するために、**値設定**のコマンドが用いられています。**メッシュ**の**既定サイズ(hauto)**のプロパティは、先に示した**サイズ**ノードに以下の設定が対応しています。

既定のメッシュサイズ	値
極めて細かい	1
さらに細かい－さらに粗い	2-8
極めて粗い	9

hauto プロパティの値は倍精度であり、任意の正の値をとることができます。非整数値については、線形補間がカスタムメッシュパラメータのために使用されます。例えば、スライダーオブジェクトに既定のメッシュサイズを調整させることができます。スライダーオブジェクトの詳細については、270 ページの「スライダー」を参照してください。

通常、個々のモデルツリープロパティの値を変更している最中にコードを記録し、自動的に生成されたコードを調べることによって、それらに許される値を迅速に知ることができます。詳細については、159 ページの「コードの記録」を参照してください。

また、**極めて細かい**から**極めて粗い**までの全ての選択肢を直接アクセスできるようにするために、コンボボックスオブジェクトを利用することができます。詳細については、203 ページの「コンボボックス」を参照してください。

データアクセスの概要

下表は、フォームオブジェクトとイベント、およびメニュー、ツールバー、リボンの各項目に関して、**データアクセス**で入手可能な場所を簡単にまとめたものです。

フォームオブジェクト、イベント、または項目	設定ウィンドウのセクション
入力フィールド	ソース
ボタン	実行コマンド選択
トグルボタン、メニュー項目切替え、 リボン項目切替え	ソース、および実行コマンド選択
チェックボックス	ソース
コンボボックス	ソース
データ表示	ソース
グラフィックス（グラフィックスツールバーアイテム）	実行コマンド選択
フォームコレクション	アクティブペーン選択 タイルまたはタブ
カードスタック	アクティブカード選択
情報カードスタック	アクティブ情報カード選択
ラジオボタン	ソース
テキスト	ソース
リストボックス	ソース
スライダ	ソース
ツールバー（ツールバーアイテム）	実行コマンド選択
メニュー項目	実行コマンド選択

フォームオブジェクト、イベント、または項目	設定ウィンドウのセクション
リボン項目	実行コマンド選択
イベント(グローバル)	実行コマンド選択 データ変更イベントのソース

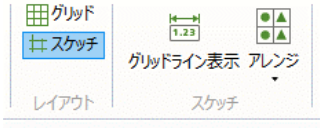
グローバルイベント、メニュー、リボン、またはツールバー項目では、その**設定**ウィンドウに**実行コマンド選択**セクションがあり、その機能に関してはボタンの章で説明されている内容が適用できます。グローバルイベントと多くのフォームオブジェクトでは、その**設定**ウィンドウに**ソース**セクションがあり、その機能に関しては入力フィールドの章で説明されています。グローバルイベント、メニュー、リボン、およびツールバー項目についての情報は、68 ページの「グラフィックスツールバー」、111 ページの「メインウィンドウ」、118 ページの「イベント」、265 ページの「テーブル」、および 274 ページの「ツールバー」を参照してください。

スケッチレイアウトとグリッドレイアウト

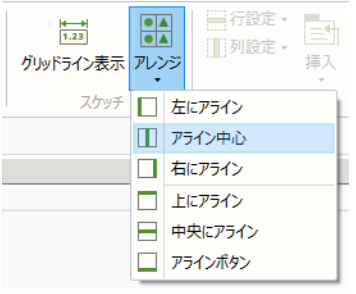
フォームエディターでは、フォームオブジェクトを配置するためのレイアウトモードとして、スケッチレイアウトモードとグリッドレイアウトモードの二つの方法が提供されています。デフォルトではスケッチレイアウトモードとなっており、ピクセルによるオブジェクトの位置とサイズが固定にされて使われます。セルが作られるグリッド背景に基づいて位置とサイズを調整するには、グリッドレイアウトモードを使います。グリッドレイアウトモードでは、一つのフォームは行と列の交差するいくつかのセルにまたがって配置されます。各交差では、最大一つのフォームオブジェクトが配置されます。複数のプラットフォームのウェブブラウザーで実行されるアプリケーションをデザインするといったように、サイズ変更可能なユーザインタフェースをデザインする場合には、グリッドレイアウトモードの利用をお勧めします。

スケッチレイアウト

リボンの**レイアウト**のグループにある**スケッチ**または**グリッド**をクリックして、スケッチレイアウトモードとグリッドレイアウトモードを切り換えます。



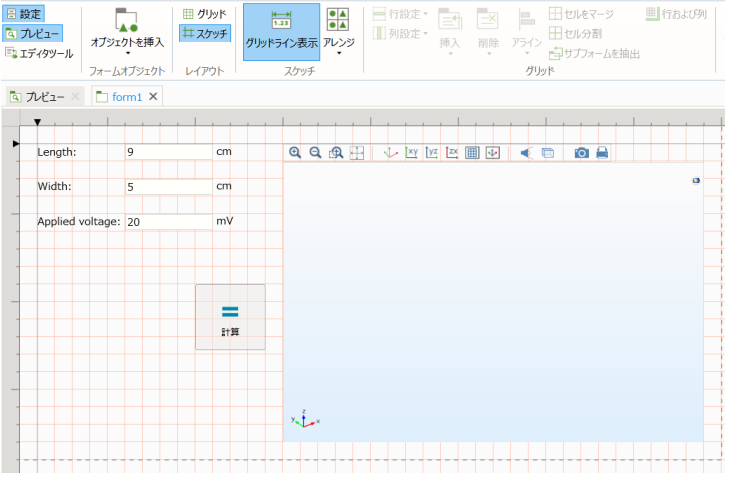
フォームタブの**スケッチ**のグループには、**グリッドライン表示**と**アレンジ**の二つの項目があります。**アレンジ**のメニューによって、互いに関連するフォームオブジェクトのグループを整列させることができます。



スケッチグリッド

グリッドライン表示の項目によって、オブジェクトがスナップされるスケッチグリッドを表示させることができます。スケッチレイアウトモードで使われるグリッドは、グリッドレイアウトモードで使われるグリッドとは異なることに注意してください。スケッチレイアウトモードでは、デフォルト設定としてグリッドラインが表示されません。グリッドラインを表示しない状態では、ドラッグされているフォームオブジェクトは他のフォームオブジェクトの位置と関連してスナップされます。

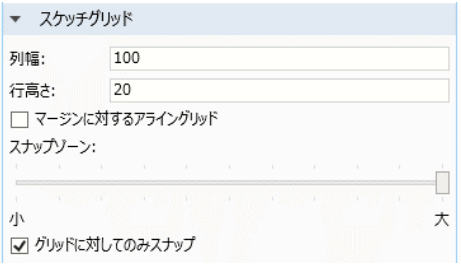
グリッドライン表示の項目を選択すると、ドラッグされているフォームオブジェクトの左上角がグリッドライン交差ポイントにスナップされます。



そのフォームの**設定**ウィンドウで、スケッチグリッドの以下の設定を変更することができます。

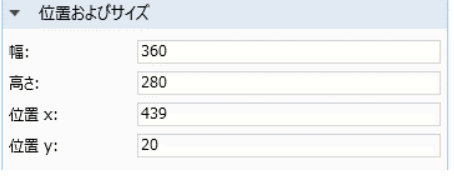
- **列幅**
- **行高さ**

- **マージンに対するアライングリッド**
- **スナップゾーン**
  - スライダーによって、スナップゾーンのサイズを**小**から**大**まで変更することができます。
- **グリッドに対してのみスナップ**
  - このチェックボックスをクリアすると、グリッドと他のフォームオブジェクトの配置の両方に関してスナップさせることができます。



位置およびサイズ

スケッチレイアウトモードでの位置はピクセルで示されます。フォームオブジェクトの位置は、スクリーンの左上角から測定されたフォームオブジェクトの左上角の座標として示されます。オブジェクトをスクリーンの右に動かすと x 座標が増加し、オブジェクトをスクリーンの上から下に動かすと y 座標が増加します。このフォームオブジェクトの絶対座標の位置は、その**設定**ウィンドウの**位置およびサイズ**のセクションで設定することができます。（下図は、スケッチレイアウトモード時の場合です。）

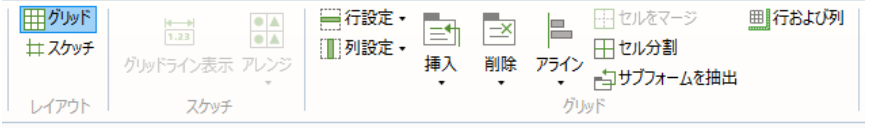


フォームオブジェクトには、必要とするスペース、または設定された**幅**と**高さ**によるスペースが割り当てられます。フォームオブジェクト同士を重ねることは可能です。

ボタンとトグルボタンのフォームオブジェクトには、**幅**と**高さ**の値の**自動**オプションと**マニュアル**オプションがあります。**マニュアル**オプションはピクセルベースの入力を可能にし、**自動**オプションはボタンのサイズを**テキスト**文字列のサイズに適合させます。

グリッドレイアウト

リボンの**レイアウト**のグループにあるの**グリッド**をクリックすると、グリッドレイアウトモードに切り換わります。

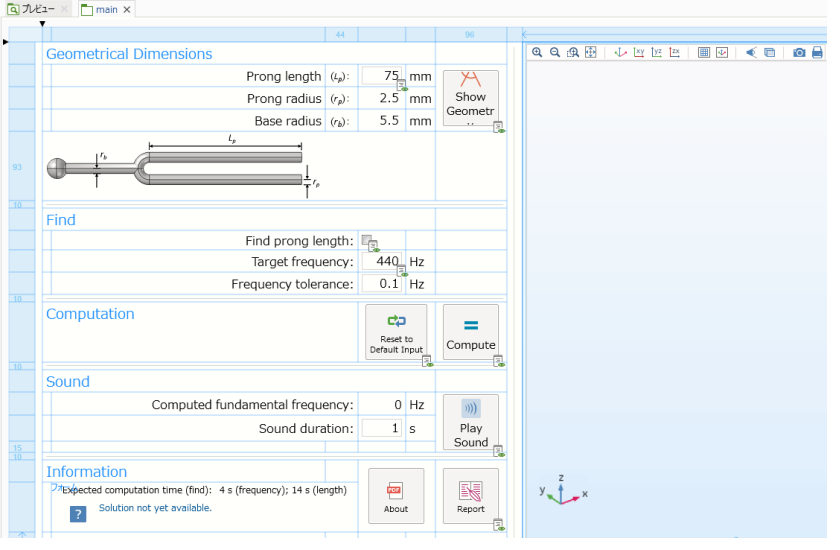


リボンの**グリッド**のグループにあるボタンとメニューによって、以下のようにコマンドへの容易なアクセスが可能です。

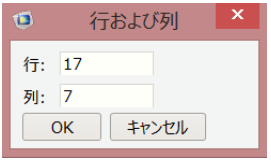
- ユーザインタフェースがサイズ変更される時にレイアウトを決定する、**フィット**、**自動拡大縮小**、**固定**の選択による行と列の適応ルールの変更（**行設定**と**列設定**）
- 行と列の挿入と削除（**挿入**と**削除**）
- グリッドセル内のフォームオブジェクトの整列（**アライン**）
- セルのマージと分割（**セルをマージ**と**セル分割**）
- サブフォームとしてセルの矩形配列を抽出し、新規フォームに挿入（**サブフォームを抽出**）
- 行数と列数の指定（**行および列**）

フォーム設定ウィンドウとグリッド

グリッドレイアウトモードに切り換えると、フォームウィンドウには青色のグリッド線が表示されます。

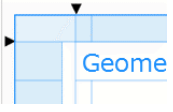


行と列の数を指定するには、リボンにある**行および列**のボタンをクリックします。



設定ウィンドウの**含まれているフォームオブジェクトのためのグリッドレイアウト**のセクションには、列の幅と行の高さが表示されます。

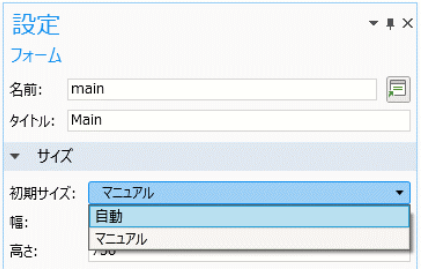
フォームエディターに表示されているフォームをインタラクティブに選択するには、フォームの左上隅をクリックします。

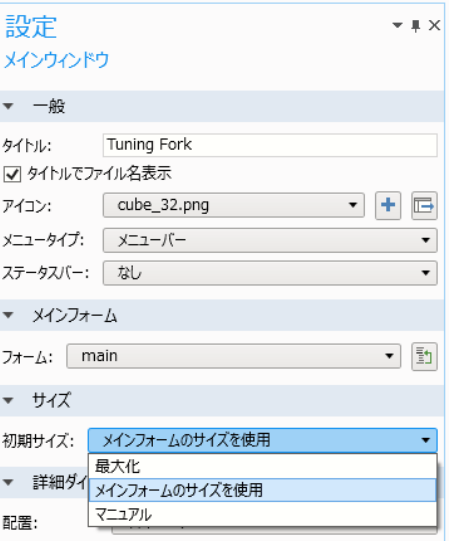


現在、青枠が表示されています。フォーム全体のサイズをインタラクティブに変更するには、その右と下の境界線をドラッグします。これを行うために、必ずしもフォーム全体を選択している必要はありません。

スケッチレイアウトモードからグリッドレイアウトモードに切り換えた場合、全ての行と列が**フィット**に設定されており、フレーム用のハンドルが表示されないことに注意してください。行と列の**高さ**または**幅**の設定が**自動拡大縮小**に設定された場合には、そのフレームを垂直または水平のそれぞれの方向にサイズ変更するためのハンドルがフレームに表示されます。

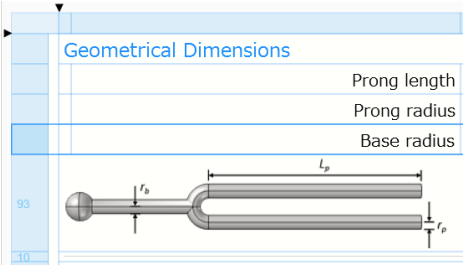
インタラクティブにサイズ変更されるフレームのサイズは、**初期サイズ**の設定が**自動**に設定されている場合のみ、そのフォームの初期サイズに影響します。また、**メインウィンドウ**の**初期サイズ**の設定が**メインフォームのサイズを使用**に設定されている場合は、そのフレームのサイズはメインウィンドウの初期サイズに影響します。



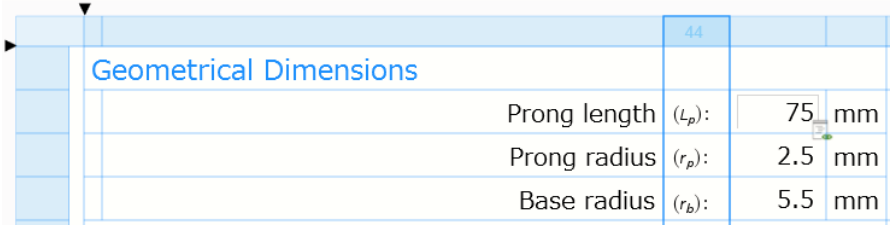


行および列

行を選択するには、その行の左端のセルをクリックします。左端のセルは行のみを選択するために使用され、そこにフォームオブジェクトを配置することはできません。行が選択されている場合、**行設定**メニューだけでなく**挿入**や**削除**コマンドもリボンタブで有効になっています。下図では、4 行目が選択されてハイライト表示されています。

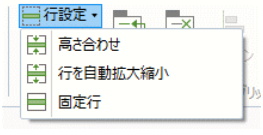


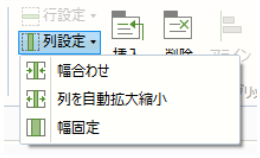
同様に、列を選択するには、その列の最上部のセルをクリックします。このセルには、フォームオブジェクトを配置することはできません。下図では、3 列目が選択されてハイライト表示されています。この場合、**列設定**メニューがリボンタブで有効になっています。



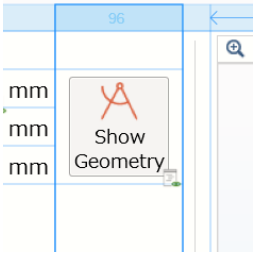
行設定と**列設定**には、どちらも同じ三つの選択肢があります。

- **合わせ**は、その行や列にあるフォームオブジェクトのサイズに対して、行の高さや列の幅ができるだけ最小値となるようにする設定です。
- **自動拡大縮小**は、フォーム全体のサイズに比例して、行の高さや列の幅を適応させるようにする設定です。
- **固定**は、行の高さや列の幅をピクセル数による固定値とする設定です。



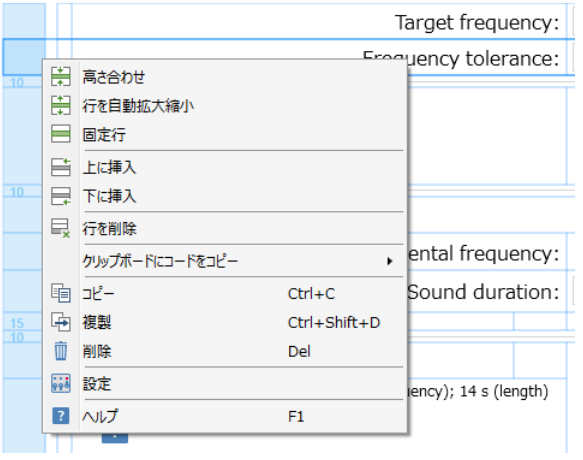


グリッド線をドラッグして、行の高さと列の幅を対話的に変更することができます。



この場合、ピクセル数が表示され、**行設定**や**列設定**の適応方針が自動的に**固定**に変更されます。

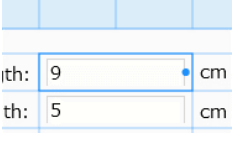
リボンの**行設定**や**列設定**を変更する代わりに、行や列を右クリックして、その時に表示されるメニューから選択することもできます。



また、行や列を右クリックした時に表示されるメニューでは、行や列に対する挿入、削除、コピー、ペースト、複製を利用することができます。

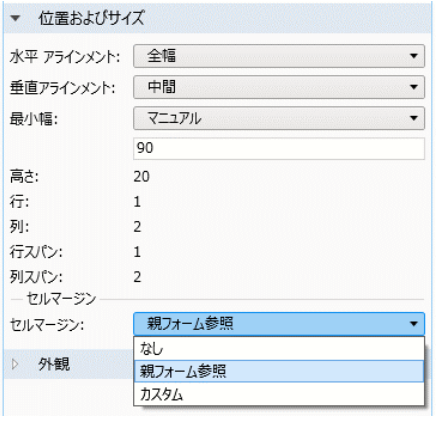
セル

セルの選択には、個々のセルをクリックします。選択されたセルは、濃い青色のグリッド線で表示されます。



セルサイズとレイアウトをフォームオブジェクトに合わせるには、リボンの**セルをマージ**と**セル分割**を選択します。

グリッドレイアウトモードでは、フォームオブジェクトとそれが置かれているセルの境界線との間に確保されるマージン（余白）を指定することができます。



フォームオブジェクトの**設定**ウィンドウの**位置およびサイズ**のセクションにおいて、**セルマージン**の設定の選択肢として以下があります。

- **なし**
  - セルマージンなし
- **親フォーム参照**（デフォルト）
  - フォームの**設定**ウィンドウで指定されたマージンは、108 ページの「列とセルマージンを継承」を参照してください。
- **カスタム**
  - カスタマイズしたマージンは、このフォームオブジェクトにのみ適用されます。

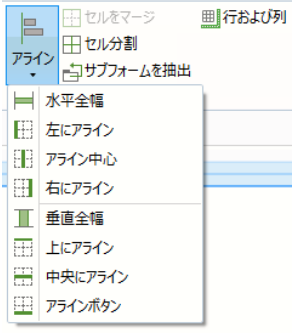
水平アラインメントまたは**垂直アラインメント**が**全幅**に設定され、行や列の適応方針がフォームオブジェクトのサイズ変更可能とされている場合には、最小の幅や高さをそれぞれ指定することができます。最小サイズは、**マニュアル**または**自動**に設定することができます。**マニュアル**を選択した場合は、最小サイズのピクセル値を指定します。**自動**を選択した場合は、もしフォームオブジェクトがより大きな値を必要としないなら、0 ピクセルの最小サイズも可能です。最小サイズを設定することによって、実行時にフォームオブジェクトがその最小サイズより小さくなると、スクロールバーが確実に表示されるようになります。

セルに含まれるフォームオブジェクトのタイプに応じて、**幅**と**高さ**の値は、97 ページの「位置およびサイズ」で説明されているように、**自動**または**マニュアル**に設定できます。

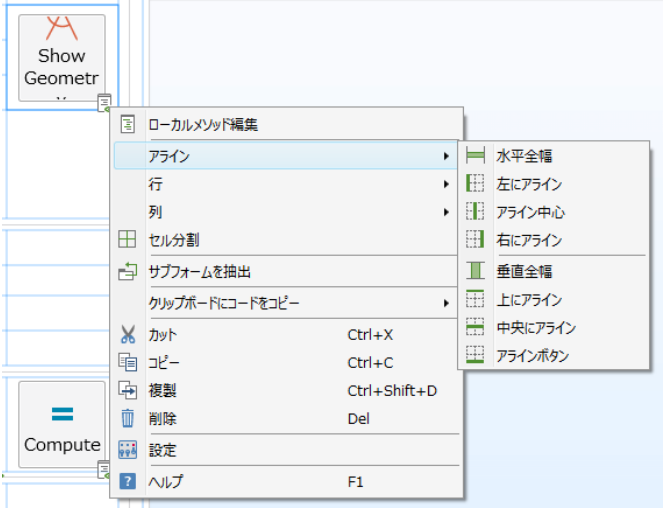
ラバーボックスをクリックしてドラッグすると、複数のセルを選択できます。

フォームオブジェクトのアライン

アラインメニューには、セル内のフォームオブジェクトを整列させるための選択肢が用意されています。また、フォームオブジェクトをセルの水平全幅または垂直全幅にするダイナミックな設定も可能です。

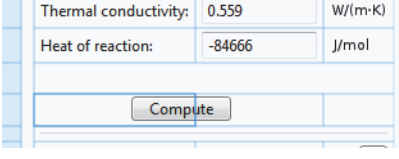


代わりに、フォームオブジェクトを右クリックし、コンテキストメニューから選択することもできます。



フォームオブジェクトのドラッグ＆ドロップ

ドラッグ＆ドロップでフォームオブジェクトを移動することができます。フォームオブジェクトをクリックして選択し、すでに他のフォームオブジェクトによって占有されていない別のセルにそれをドラッグします。



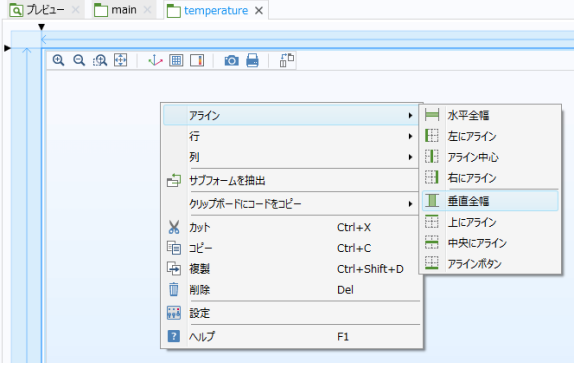
すでに占有されているセル内のオブジェクトにドロップした場合、オブジェクト同士が入れ替わります。

グラフィックスオブジェクトの自動サイズ変更

サイズ変更可能なアプリケーションのグラフィックスオブジェクトを作成する手順を、以下に示します。

- グラフィックスオブジェクトを含むフォームのレイアウトモードを、スケッチレイアウトモードからグリッドレイアウトモードに変更します。
- グラフィックオブジェクトが置かれている領域全体の行について、行の**高さ**の設定を**自動拡大縮小**に変更します。この変更を行うには、左端の列の中でアクセスしたい行をクリックします。その後、フォームの**設定**ウィンドウで、その行の**高さ**の設定を変更します。別の方法として、右クリックして**行を自動拡大縮小**を選択することもできます。

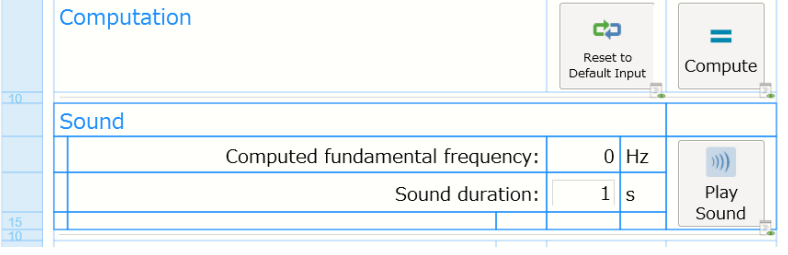
- グラフィックオブジェクトが置かれている領域全体の列について、列の**幅**の設定を**自動拡大縮小**に変更します。この変更を行うには、最上部の行の中でアクセスしたい列をクリックし、先と同様の方法で**列を自動拡大縮小**を選択します。
- グラフィックスオブジェクトを選択し、**水平アラインメント**と**垂直アラインメント**の両方を**全幅**に変更します。この設定は、**設定**ウィンドウによって行うか、またはグラフィックスオブジェクトを右クリックして、**アライン > 水平全幅**、および**アライン > 垂直全幅**を選択することによって行います。



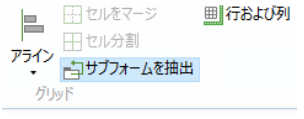
上記の手順に従いながら、セルをマージしたり、空の行と列を追加するなどのグリッドレイアウトモード操作を行ってみれば、簡単にサイズ変更可能なグラフィックスを作れることがわかるはずです。事前にグリッドレイアウトモードになっていれば、そこに挿入されたグラフィックスオブジェクトは水平と垂直の両方向とも**全幅**にデフォルト設定されています。

サブフォームの抽出

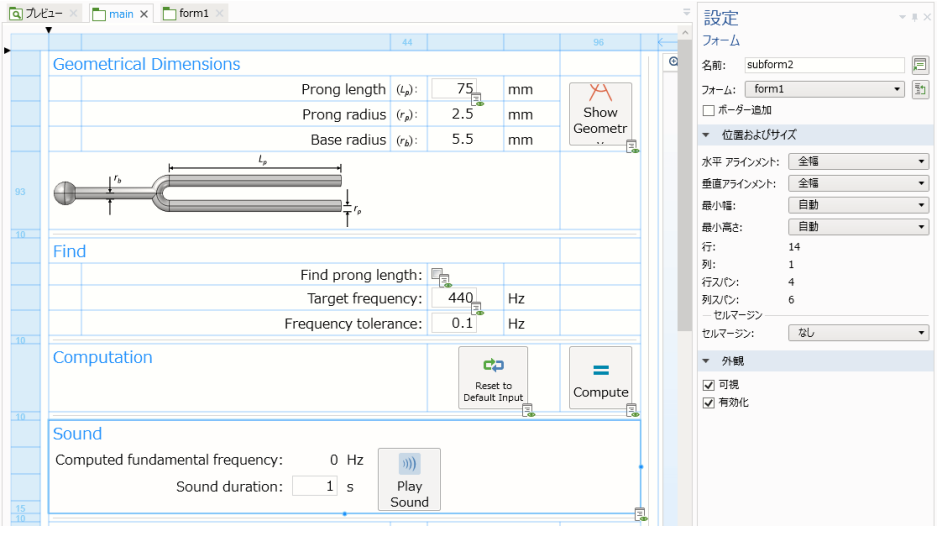
フォーム内のセルの矩形配列を選択して、それを新しいフォームに移動させることができます。まず最初に、Ctrl+click または Shift+click の操作でセルを選択します。

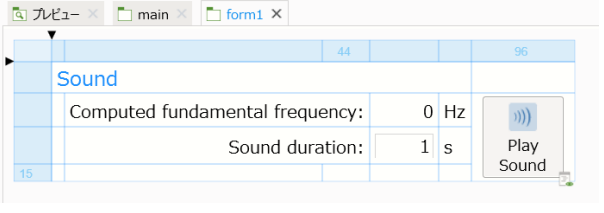


そして、リボンで**サブフォームを抽出**のボタンをクリックします。



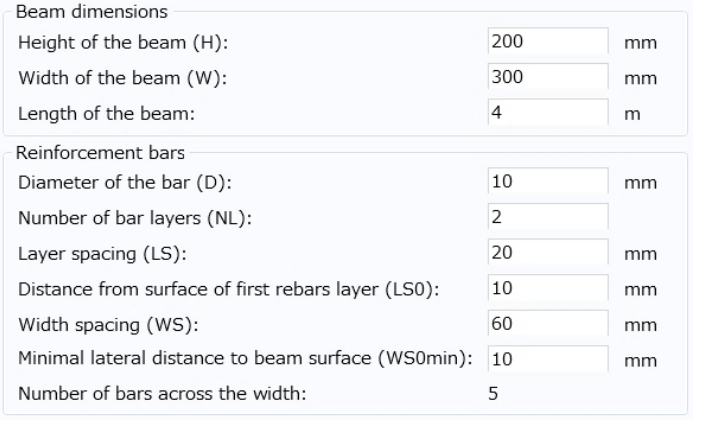
この操作によって、選択したセルで新しいフォームが作成され、元のセルは**フォーム**タイプのサブフォームオブジェクトに置き換えられます。サブフォームの**設定**ウィンドウの**フォーム**設定の参照先には、元となるセルを含んだ新しいフォームが示されています。





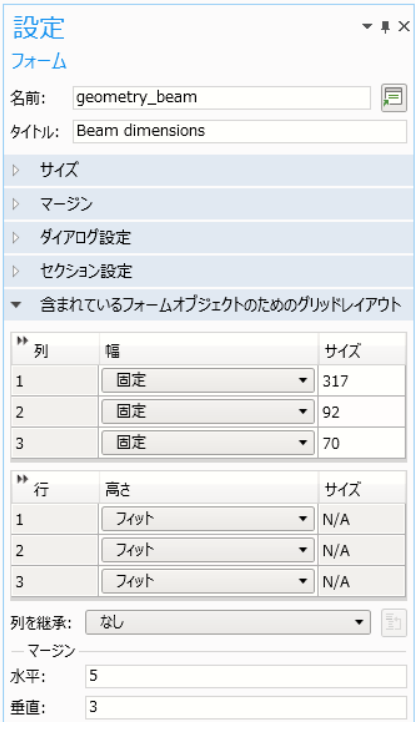
列とセルマージンを継承

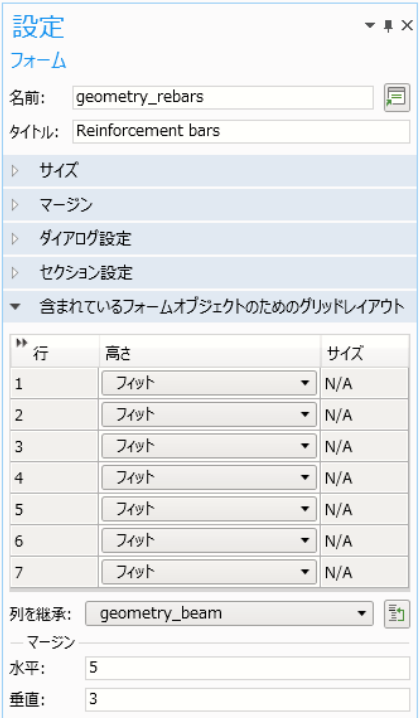
例えば、入力フォームのセットをグループ化する場合など、サブフォームを使用することでユーザインタフェースを整理することができます。下図には、**Beam dimensions** と **Reinforcement bars** のための二つのサブフォームを持つ実行中のアプリケーションの一部を示しています。



フォームにサブフォームを追加する方法の詳細については、前章と 235 ページの「フォーム」を参照してください。

上図の例のように、垂直方向にサブフォームを整列させる時に、全ての列が同じ幅になるようにすることができます。このためには、サブフォームの**設定**ウィンドウで**列を継承**の選択を使用します。次の図は、geometry_beam という**名前**の **Beam dimensions** サブフォーム（左）と geometry_rebars という**名前**の **Reinforcement bars** サブフォーム（右）の**設定**ウィンドウの一部を示しています。geometry_rebars のサブフォームの**列を継承**の設定で、geometry_beam が指定されています。





サブセクションの**マージン**では、フォームオブジェクトが置かれているセルの境界線とフォームオブジェクトの間隔を、**水平**と**垂直**のマージンとして設定することができます。この設定は、フォームに含まれているフォームオブジェクトの各**セルマージン**が**親フォーム参照**に設定されることによって、それら全てに対して影響を与えます。102 ページの「セル」を参照してください。

アプリケーション間のコピー

同時に起動させている複数の COMSOL マルチフィジックスのセッション間で、フォームやフォームオブジェクトをコピーペーストすることができます。また、一つのセッションの中で、現在のアプリケーションから新たにロードにたアプリケーションにコピーペーストすることができます。グリッドレイアウトモードのセッション間で、一つのセル、複数のセル、列全体、あるいは行全体をコピーすることができます。

アプリケーション間でフォームやフォームオブジェクトをコピーペーストする場合、コピーされたオブジェクトは他のフォームやフォームオブジェクトへの参照を含んでいます。そのような参照は、コピーされたアプリケーションでは適合しないかもしれません。オブジェクトを貼り付ける場合に適用される設定ルールの詳細については、277 ページの「付録 B—アプリケーション間のコピー」を参照してください。

アプリケーション間でコピーペーストする際に潜在的な互換性の問題が検出された場合には、メッセージダイアログボックスが表示されます。このとき、ペースト操作のキャンセルを選択することもできます。

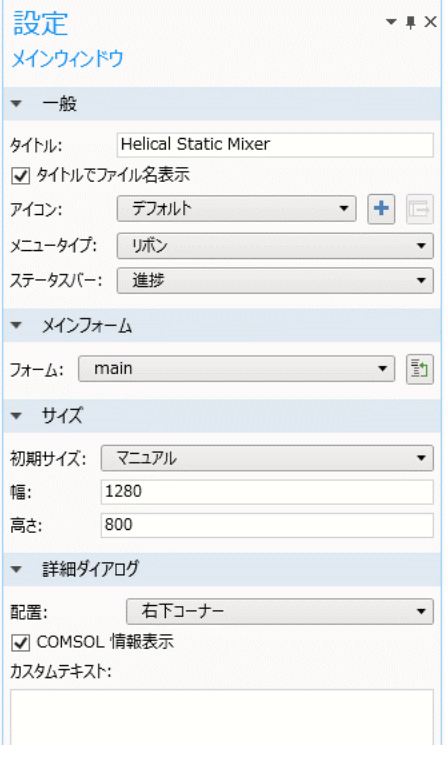
メインウィンドウ

アプリケーションツリーの**メインウィンドウ**ノードはアプリケーションのメインのウィンドウを表しており、ユーザインタフェースのための最上位のノードです。そこには、ウィンドウのレイアウト、メインメニューの設定、およびリボン項目の設定を含んでいます。

一般

設定ウィンドウでは、**一般**セクションに以下の設定を含んでいます。

- **タイトル**
- **タイトルでファイル名表示**
- **アイコン**
- **メニュータイプ**
- **ステータスバー**



タイトルは、アプリケーションを実行したときのメインウィンドウのトップに表示されるテキストで、**アイコン**は、そのタイトルが表示されている行の最も左端に表示されます。タイトルの初期値は、アプリケーショ

ンの作成に使用しているモデルのタイトル名と同じです。アプリケーションのファイル名を**タイトル**の左側に続けて表示した場合には、**タイトルでファイル名表示**のチェックボックにチェックを入れます。

アイコンの設定は、リストにあるライブラリから画像を選択します。リストのライブラリにローカルファイルシステムから画像（*.png）を追加して、アイコンとして使うこともできます。新しい画像を追加した場合、それは画像ライブラリに追加されて、アプリケーションに組み込まれます。

この**ライブラリに対する画像を追加してここで使用**のボタン（＋印）の右隣にある**エクスポート**ボタンをクリックすることによって、アイコンをエクスポートすることもできます。
アプリケーションツリーの**メインウィンドウ**ノードは、**メニューバー**という名前のサブノードを持つことができます。メインウィンドウノードの設定ウィンドウの**メニュー**タイプ設定により、このサブノードを**メニューバー**から**リボン**に変更することができます。
ステータスバーの設定リストは、ステータスバーによる表示を選択します。利用する場合には進捗バーを表示するための**進捗**（デフォルト）を選択し、そうでない場合は**なし**を選択します。メソッドを利用すれば、カスタマイズした進歩バーを作成することができます。

メインフォーム

メインフォームセクションは、メインウィンドウが表示するフォームへの参照設定です。これによって、アプリケーション実行時にどのフォームがメインウィンドウとして最初に表示されるかを定義します。フォームコレクションを使う時には、この設定が重要です。

サイズ

サイズセクションにおける**初期サイズ**の設定は、アプリケーションの実行がスタートする時のメインウィンドウのサイズです。この設定には三つの選択肢があります。

- アプリケーションが実行された際にウィンドウが最大化される**最大化**
- **メインフォームのサイズを使用**を選択した場合は、メインフォームのサイズが適用されます。これに関しては、41 ページの「個々のフォーム設定ウィンドウ」を参照してください。メインフォームは**メインフォーム**セクションによって定義されています。これを選択すると、メインウィンドウ自体が必要なサイズとして、ウィンドウフレームとタイトルバー、メインメニュー、メインツールバー、およびリボンのサイズが追加されます。このサイズは、メニュータイプが**メニューバー**か**リボン**かによって自動的に計算されます。
- **マニュアル**を選択すると、幅と高さのピクセルサイズを入力する設定が表示されます。この場合は、ここで設定する幅と高さ以外に追加されるサイズはありません。このため、これを使う時には、ウィンドウのタイトル、リボン、およびメニューバーのために十分な余地があることを確認する必要があります。

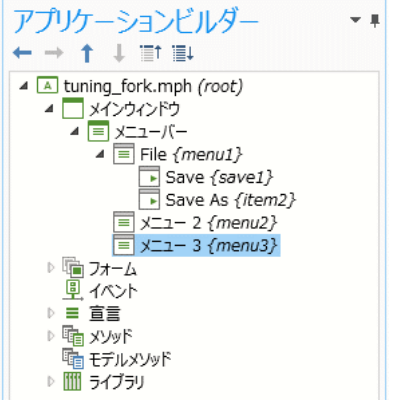
ここでの選択肢である**メインフォームのサイズを使用**の詳細については、98 ページの「フォーム設定ウィンドウとグリッド」を参照してください。

詳細ダイアログ

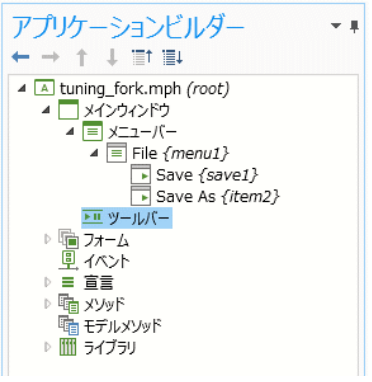
詳細ダイアログセクションでは、法的な情報が含まれた**製品情報**ダイアログボックスのカスタマイズ部分のための設定が含まれています。メニュータイプがリボンに設定されている場合、**配置**オプションを使用して、**自動**、**ファイルメニュー**、**リボン**、または**右下コーナー**から選択することができます。メニュータイプがメニューバーに設定されている場合、**配置**オプションを使用して、**自動**、**メニューバー**、**ツールバー**、または**右下コーナー**から選択することができます。**右下コーナー**を選択した場合は、アプリケーションのユーザインタフェースの右下隅に、**製品情報**ダイアログボックスへのハイパーリンクが配置されます。**COMSOL 情報表示**のチェックボックスにチェックを入れている場合は、COMSOL のソフトウェアバージョンと製品情報が表示されます。**カスタムテキスト**フィールドに入力したテキストは、ダイアログボックスで法的な情報に関して表示されます。**カスタムテキスト**フィールドでは、可能なら、http または www を含む言葉はハイパーリンクとして解釈されます。例えば、http://www.comsol.com or www.comsol.com はハイパーリンクに置き換えられます。

メニューバーとツールバー

メニューバーノードでは、メインウィンドウのトップでメニュー表示される**メニュー**サブノードを設定できます。

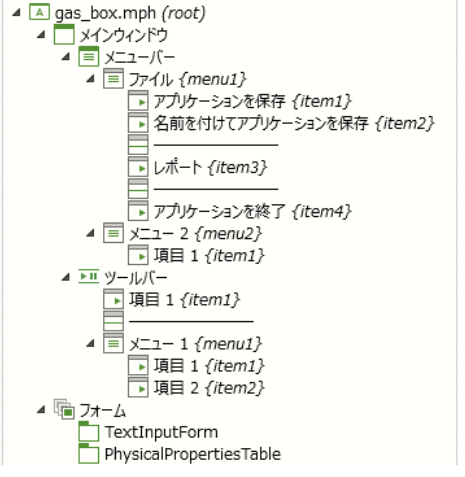


メニューバーのサブノードとして、**ツールバー**も追加することができます。**ツールバー**ノードと**メニュー**ノードは、同じタイプのサブノードを持ちます。

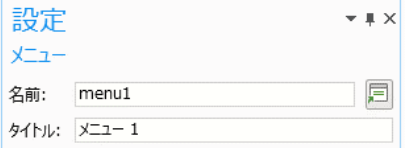


メニュー、項目、セパレータ

メニューと**ツールバー**ノードのサブノードには、下図に例示したように、**メニュー**、**項目**、**項目切替え**、および**セパレーター**のタイプがあります。



メニューノードには、**名前**と**タイトル**の設定があります。

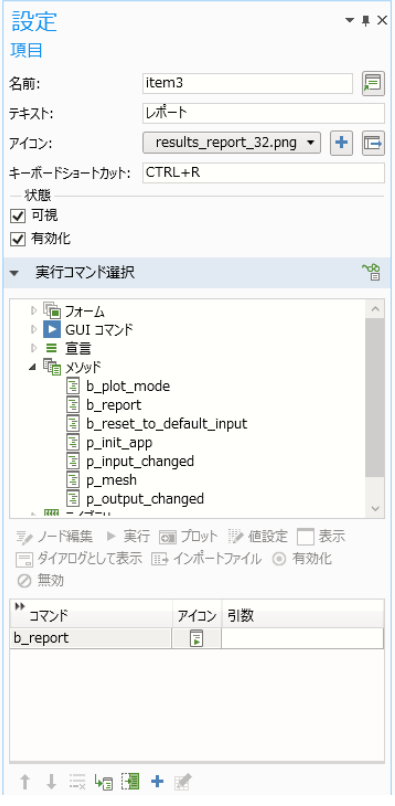


メニューノードは、その分割メニューとしてサブ**メニュー**ノードを追加することができます。

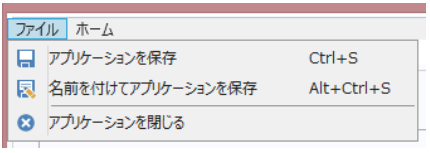
セパレーターは、メニューと項目のグループの間に水平線を挿入表示するためのもので、設定はありません。

項目ノードの設定ウィンドウには、ボタンの場合と同様に、実行コマンド選択セクションがあります。項目のテキストとアイコンとキーボードショートカットの設定に関しても、ボタンの場合と全く同じです。詳細については、51 ページの「ボタン」を参照してください。同様に、**項目切替え**ノードの**設定**ウィンドウは、トグルボタンの場合と同様です。

下図には、レポートの作成方法に関する**項目**の**設定**ウィンドウを示しています。

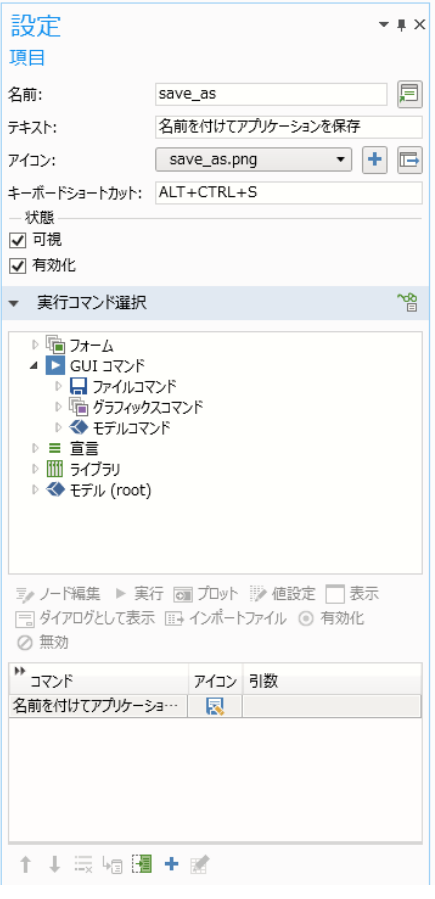


下図には、**ファイル**メニューに関するアプリケーションの実行例を示しています。



COMSOL デスクトップ環境でアプリケーションを実行する際には、**アプリケーションを閉じる**のメニュー項目は常にデフォルト表示されます。

下図には、**名前を付けてアプリケーションを保存**の項目の**設定**ウィンドウを示しています。



リボン、メニュー、およびツールバーの各項目は、メソッドによる有効/無効の制御が可能です。詳細については、297 ページの「付録 E—組み込みメソッドライブラリ」を参照してください。

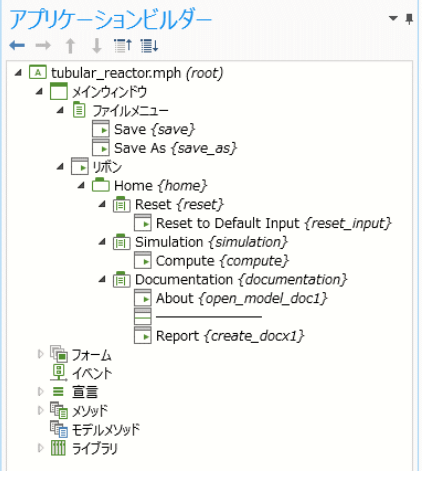
リボン

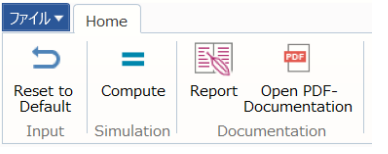
メニューバーの代わりに、**リボン**を**メインウィンドウ**に追加することができます。**リボン**ノードは、一つ以上のリボンタブの上にツールバーが配置されるという構成の仕様になっています。この**リボン**を選択することによって、**ファイル**メニューを**メインウィンドウ**ノードのすぐ下に作ることができます。

リボンタブとリボンセクション

リボンノードの下に作られるサブノードは、**リボンタブ**というタイプです。**リボンタブ**の下に作られるサブノードは、**リボンセクション**というタイプです。**リボンセクション**の下に作ることができるサブノードは、**項目**、**項目切替え**、**メニュー**、および**セパレーター**です。

この**項目**と**メニュー**に関しては、前章の**メニューバー**と**ツールバー**のところで説明した内容と機能的に同じです。**リボンセクション**の下に追加される**セパレーター**は、アプリケーション実行時に**項目**と**メニュー**のグループを分離表示するための垂直線です。この**セパレーター**は、アプリケーションツリーの中では水平線として表示されています。下図に一例を示します。





イベント

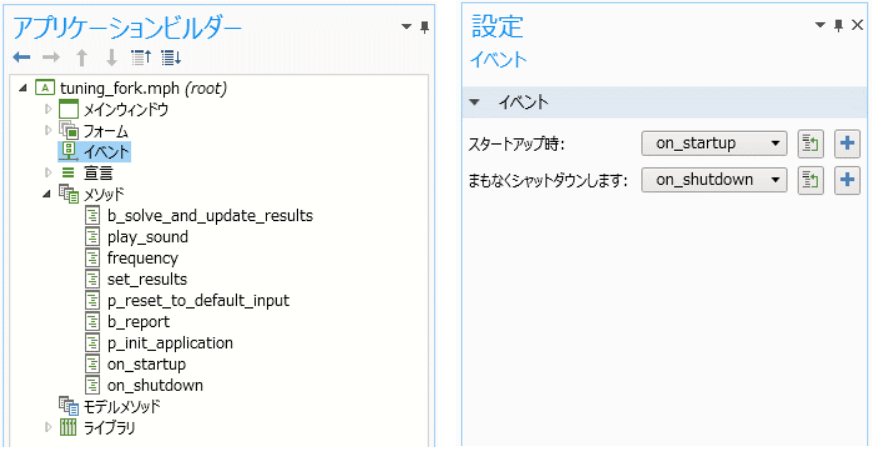
イベントは、アプリケーションで 1 回以上のアクションを実行させる必要が場合に、その指令を発信する操作です。例えば、ボタンをクリックする、キーボードショートカットをタイプする、フォームをロードする、または変数の値を変更するといった操作です。個々の動作は、先に説明したような一連のコマンドによるタイプ、または、メソッドによって実行されます。メソッド自体は、アプリケーションにおいてどこからでも開始できる特定のフォームオブジェクトのローカルメソッド、またはグローバルメソッドです。グローバルメソッドは、アプリケーションツリーの**メソッド**ノードの下に表示されます。ローカルメソッドは、そのメソッドを関連付けるフォームやフォームオブジェクトの**設定**ウィンドウで定義されます。フォームオブジェクトに関連付けられたメソッドが作成されている場合、そのオブジェクトを選択して Ctrl+Alt+click の操作をすることで編集用に開くことができます。メソッドが作成されていないフォームオブジェクトに対して Ctrl+Alt+click を操作した場合には、そのオブジェクトに関連付けるローカルメソッドが新規に作成され、編集用に開かれます。

また、これらのアクションを発動するイベントにも、グローバルとローカルのタイプがあります。グローバルイベントは、アプリケーションツリーの**イベント**ノードの下に表示されます。グローバルパラメータや文字列変数といったさまざまなデータ入力の変化による起動を発動するイベントには、全て、このグローバルイベントを使うことができます。また、グローバルイベントは、アプリケーションのスタートアップやシャットダウンと関連付けることができます。ローカルイベントは、ローカルなオブジェクトのように、それらが関連付けられるフォームやフォームオブジェクトの**設定**ウィンドウにおいて定義されます。

イベントノードは、ソースデータが変化する時にはいつでも起動され、メソッドやフォームオブジェクト、あるいは他の方法のいずれによる変化なのかは関係しません。フォームオブジェクトに関連付けられたイベントでは、ユーザがそのフォームオブジェクトの値を変化させた時にだけ起動されます。

スタートアップとシャットダウン時のイベント

グローバルメソッドまたはローカルメソッドは、アプリケーションのスタートアップ（**スタートアップ時**）やシャットダウン（**まもなくシャットダウンします**）の時のイベントに関連付けることができます。これらのイベントにアクセスするには、アプリケーションツリーの**イベント**ノードをクリックします。



次の場合にシャットダウンイベントが起動されます。

- アプリケーションウィンドウの右上角にあるアプリケーションを**閉じる**のアイコンをクリックして、アプリケーションウィンドウが閉じられるとき
- フォームオブジェクトによって、**アプリケーションを終了**のコマンドが実行されるとき
- メソッドの中で、コマンド exit( ) を使ったコードが実行されるとき

例えば、シャットダウンイベントで実行されるメソッドによって、自動的に重要なデータを保存したり、データの保存を促すメッセージを出すことができます。また、シャットダウンイベントで実行されるメソッドで、ブーリアンの true 値を返すことによってシャットダウンをキャンセルすることができます。

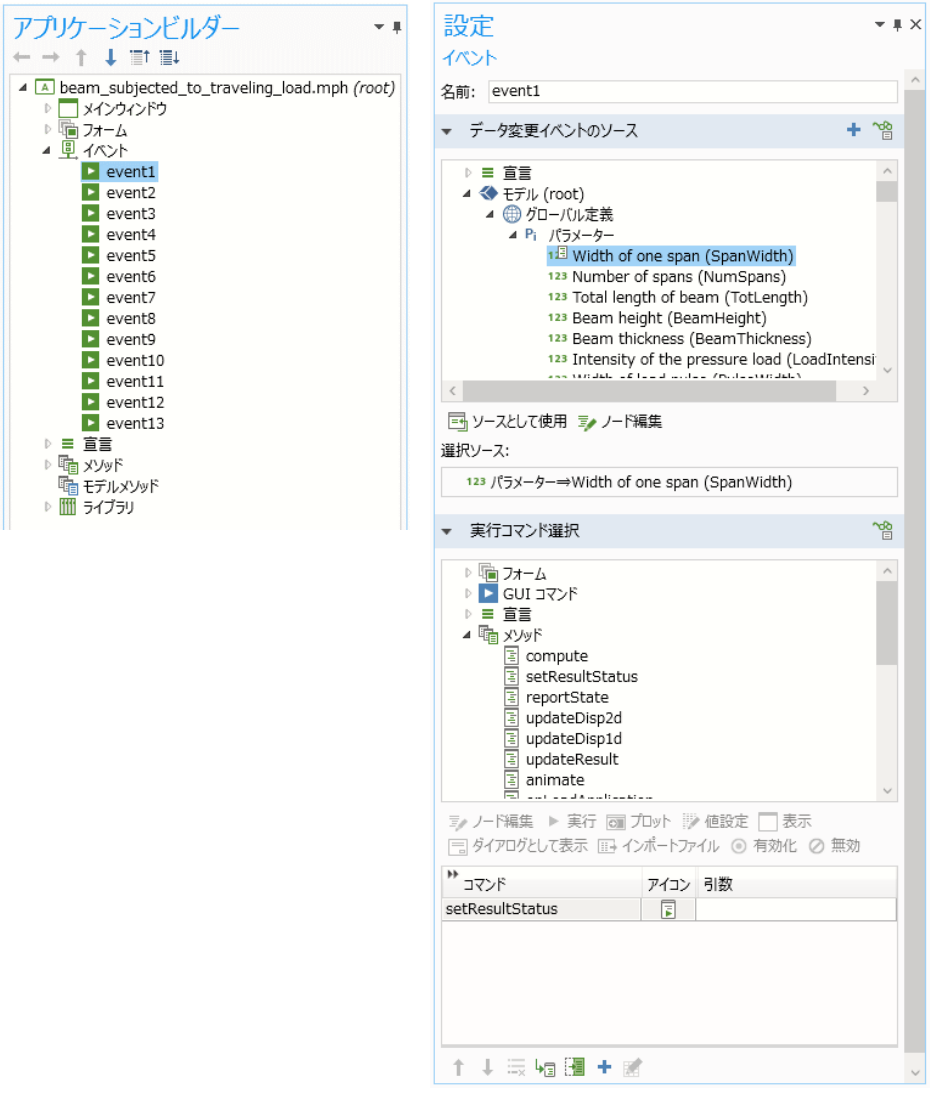
画面にわたってズームなどのグラフィックスを初期化するために使われるメソッドは、グローバルな**スタートアップ時**のイベントとしてではなく、そのフォームの**ロード時**のイベントとして実行される必要があることに注意してください。

スタートアップ時の制限

スタートアップ時のイベントに使用されるメソッドは、グラフィックスやユーザインタフェースに関連するアプリケーションビルダーの機能を利用できません。これは、アプリケーションユーザインタフェースが完全に読み込まれる前に**スタートアップ時**のイベントが実行されるためです。例えば、**画面にわたってズーム**などのグラフィックスを初期化するために使用されるメソッドは、グローバルな**スタートアップ時**のイベントとしてではなく、フォームの**ロード時**のイベントとして実行する必要があります。別の例としては、confirm などの組み込みメソッドを使用してダイアログボックスを表示するような場合です。この場合、ダイアログボックスは表示されず、操作は単に無視されます。

グローバルイベント

イベントノードを右クリックして**イベント**を選択すると、アプリケーションにイベントが追加されます。イベントは、アプリケーションの実行中の変化を待っています。もし変化が起こったら、一連のコマンドが実行されます。下図では、文字列変数 SpanWidth の値が変更されるとメソッド setResultsStatus が実行されます。



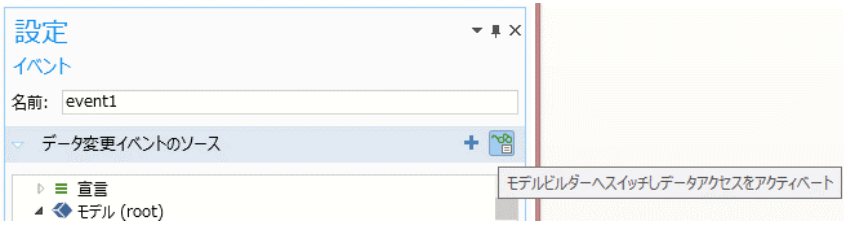
このタイプのイベントはグローバルスコープ（有効範囲）を持ち、特定のフォームに限定されないことに留意してください。グラフィックスオブジェクトを参照する時には、フルパス：/form1/graphics1 を使う必要があります。

以下に説明する二つのセクションは、イベントの**設定**ウィンドウで利用可能な設定です。

データ変更イベントのソース

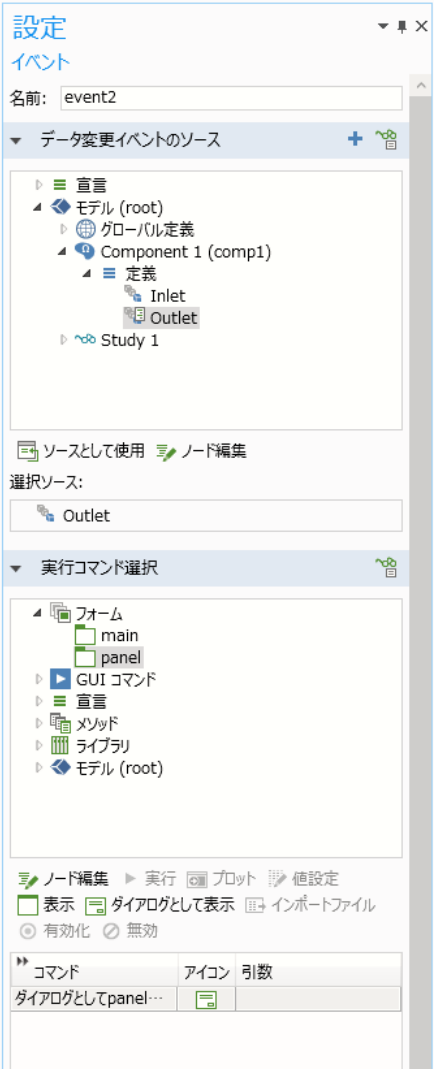
このセクションのツリーには、アプリケーションビルダーウィンドウから検索された候補が表示されています。そのノードには、そのサブノードも含めて、ある種類のデータが表示されます。

データ変更イベントのソースセクションのヘッダにある**モデルビルダへスイッチしデータアクセスをアクティベート**のボタンをクリックすると、利用可能なデータノードを追加して増やすことができます。



詳細については、157 ページの「メソッドエディターでのデータアクセス」を参照してください。

また、**明示的**選択が**データ変更イベントのソース**として設定可能であることに注意してください。これにより、ユーザがジオメトリオブジェクト、ドメイン、サーフェス、エッジ、またはポイントをクリックした際に、コマンドシーケンスまたはメソッドを実行することができます。次の図の設定では、ユーザが **Outlet** という名前の**明示的**選択の内容を変更した時に、ダイアログボックスとしてフォームパネルを開くグローバルイベント用のダイアログボックスが表示されます。



実行コマンド選択

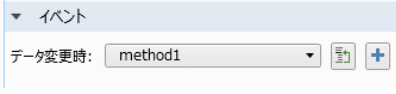
イベントの**設定**ウィンドウにおける**実行コマンド選択**セクションでは、ボタンの場合と同様に一連のコマンドを設定することができます。詳細については、51 ページの「ボタン」を参照してください。

フォームとフォームオブジェクトのイベント

フォームとフォームオブジェクトのイベントは、フォームや個々のフォームオブジェクトのために定義されている点でグローバルイベントと異なります。これらのイベントでは実行コマンドのリストを選択するような設定はなく、それとは無関係に直接一つのグローバルメソッドまたはローカルメソッドを参照します。

データ変更によって起動されるイベント

あるタイプのフォームオブジェクトにおいて、データが変更される際に実行するメソッドを指定することができます。この設定は、下図に示すように、フォームオブジェクトの**設定**ウィンドウの**イベント**セクションで利用することができます。



データ変更時のドロップダウンリストには、**なし**（デフォルト）、アプリケーションツリーの**メソッド**ノードの下に作成されている利用可能なメソッド、およびローカルメソッド（選択による）があります。

このタイプのイベントをサポートしているフォームオブジェクトには、以下があります。

- **入力フィールド**
- **チェックボックス**
- **コンボボックス**
- **グラフィックス**
- **ファイルインポート**
- **配列入力**
- **ラジオボタン**
- **テキスト**
- **リストボックス**
- **テーブル**
- **スライダ**

ボタンは、クリックによって起動されるイベントを関連付けます。メニュー、リボン、およびツールバーの項目は、それらを選択することによって起動されるイベントを関連付けます。このボタンオブジェクトや項目の**設定**ウィンドウにおいて定義されたコマンドシーケンスが、それらのイベントに対応したアクションとなります。コマンドシーケンスの詳細については、51 ページの「ボタン」を参照してください。

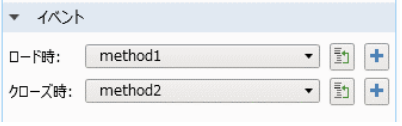
複数のフォームオブジェクトの選択

Ctrl+click の操作で、同時に複数のフォームオブジェクトの**データ変更時**のイベントを指定し、実行するメソッドを選択することができます。例えば、プロットや出力が無効であることをユーザに知らせるための

メソッドをいくつかのフォームオブジェクトで実行すべきような場合には、データ変更時のイベントを使ったこの方法で簡単に設定することができます。

ロード時やクローズ時に起動されるイベント

フォームでは、それらがロードされる(**ロード時**)、または閉じられる(**クローズ時**)場合のメソッドを実行することができます。



このタイプのイベントは、フォームの**設定**ウィンドウで利用可能です。通常は、フォームをダイアログボックスとして表示するため、または、フォームコレクションのペインとして使われるフォームをアクティブにするために利用されます。**画面にわたってズーム**などのグラフィックスを初期化するために使われるメソッドは、グローバルな**スタートアップ時**のイベントとしてではなく、そのフォームの**ロード時**のイベントとして実行される必要があることに注意してください。

ローカルメソッドの利用

イベントは、アプリケーションツリーには表示されないローカルメソッドを呼び出すことができます。ローカルメソッドの詳細については、172 ページの「ローカルメソッド」を参照してください

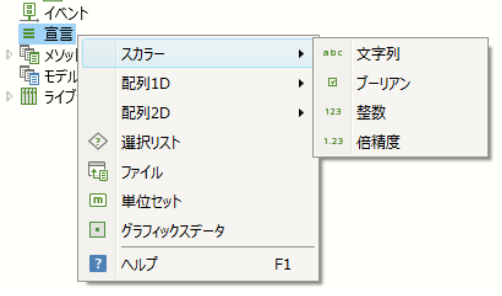
宣言

アプリケーションツリーにある**宣言**ノードでは、モデルですでに定義されているグローバルパラメータと変数に加えて使用するグローバル変数とオブジェクトを宣言することができます。**宣言**ノードの下で定義された変数は、フォームオブジェクトとメソッドで使うことができます。フォームオブジェクトの設定で宣言された変数を使う場合、他のフォームオブジェクトまたはメソッドで使われる値を引数として格納します。フォームオブジェクトとメソッドの間で引き渡されることがなくメソッド内でのみ使われる変数は、**宣言**ノードに宣言する必要はありません。メソッドでは、**宣言**ノードの下で定義された変数はグローバルスコープを持ち、自分の名前を直接使用することができます。モデルツリーで定義されたグローバルパラメータにアクセスする方法についての情報は、187 ページの「グローバルパラメータへのアクセス」を参照してください。

宣言には、以下のタイプがあります。

- **スカラー**
- **配列 1D**
- **配列 2D**
- **選択リスト**
- **ファイル**
- **単位セット**
- **ショートカット**
- **グラフィックスデータ**

宣言タイプにアクセスして定義するには、**宣言**ノードを右クリックするか、またはリボンを使います。



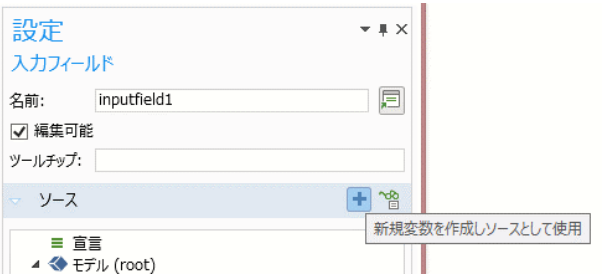
ショートカットは、このメニューからではなく、フォームオブジェクトの**設定**ウィンドウの**名前**の横にある**ショートカット作成**ボタンをクリックするか、あるいはフォームオブジェクトを選択した状態で Ctrl+ K の操作を行うことによって作成されることに注意してください。

さらに、宣言の最初の三つのタイプには、以下のデータ型をもっています。

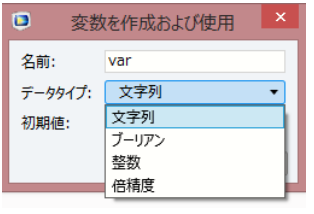
- **文字列**
- **ブーリアン**

- **整数**
- **倍精度**

宣言ノードを右クリックして定義する以外に、フォームオブジェクトの多くのタイプの**ソース**セクションにある**新規変数を作成しソースとして使用**のボタンをクリックして定義することもできます。



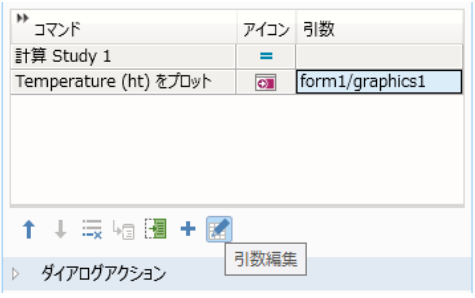
この方法では、スカラー変数を迅速に宣言することができるダイアログボックスが開かれます。

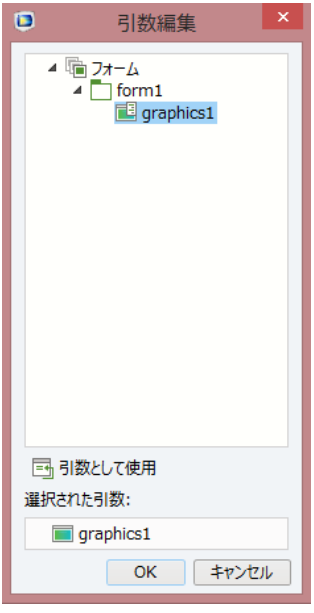


コマンドに入力する引数への宣言の利用

例えば、ボタンのコマンドシーケンスでは、引数の入力に使われるコマンドがあります。詳細については、51 ページの「ボタン」を参照してください。

下図は、コマンドシーケンスのコマンド欄の **Temperature (ht) をプロット**のコマンドに form1/graphics1 の引数を入力している場合を示しています。





コマンドに入力する引数として宣言を使うことができます。

入力する引数としてスカラー変数、配列 1D、または配列 2D を使うために、それに対応する変数名を付けます。配列 1D の単一要素、または配列 2D の列や行の要素にアクセスするには、インデックスを使います。例えば、配列 1D my_variable の最初のコンポーネントにアクセスするには、my_variable(1) とします。配列 2D の要素では、二つのインデックス、例えば my_matrix(2,3) のように一つのスカラーとして取り扱います。例えば、my_variable(n) のように、インデックス n が異なれば、それ毎に異なった宣言変数として扱われます。

引数としてグラフィックスオブジェクトを入力するコマンドでは、文字列型の宣言だけが許されます。その際に必要なら適切なインデックスの配列とします。graphics1 という名前のグラフィックスオブジェクトと graphics1 と名付けた文字列宣言がある場合には、文字列宣言の方の内容が使われます。例外として、グラフィックスオブジェクト graphics1 が使われる場合、'graphics1'のようにシングルクオーテーションが使われます。この規則は、その他のコマンドと引数の組み合わせにも適用されます。

変数の名前

変数の**名前**はスペースのないテキスト文字列とします。文字列には、文字、数字、および下線を含みます。root と parent の予約語を使うことは許されず、また、Java® プログラミング言語のキーワードを使うことはできません。

スカラー

スカラーの宣言によって、文字列、ブーリアン、整数、または倍精度の変数を定義します。

文字列

スカラーの文字列変数は、モデルにおけるグローバルパラメータや変数に似ていますが、異なる点もあります。モデルのパラメータや変数は、その値が有効なモデル表現であるように定義されなければならないという制約がありますが、スカラー文字列変数にはそのような制約がありません。メソッドに関する変換機能を使って、倍精度、整数、またはブーリアンを表す文字列変数を使うことができます。詳細については、315 ページの「変換メソッド」を参照してください。また、入力フィールド、コンボボックス、カードスタック、リストボックスなどの多くのフォームオブジェクトのソースとして文字列変数を使うことができます。

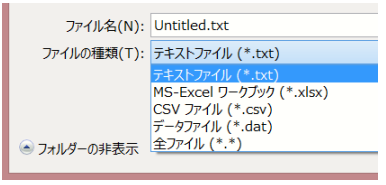
下図には、**設定**ウィンドウに文字列変数 graphics_pane、email_to、および solution_state を定義している場合を示しています。

他の宣言と同様、文字列宣言では、**変数リスト**の下にある**ファイルからロード**と**ファイルに保存**のボタンを使って、ファイルに対するロードと保存をすることができます。

ファイルからロードと**ファイルに保存**のボタンは、以下に示すファイル形式に対するロードと保存に使われます。

- テキストファイル (.txt)
- Microsoft® Excel® Workbook (.xlsx)
  - LiveLink™ *for* Excel® が必要です。
- CSV ファイル (.csv)
- データファイル (.dat)

下図には、これらのファイル形式を選択するドロップダウンリストを示しています。



宣言された文字列の使用法の説明のため、下図に、カードスタックオブジェクトの**設定**ウィンドウで文字列変数 viewCard がソース(**アクティブカード選択**)として設定されている場合を示しています。

カードスタックの使い方の詳細については、239 ページの「カードスタック」を参照してください。

ブーリアン

ブーリアン変数は、チェックボックスなどのフォームオブジェクトやメソッドのソースとして使われます。ブーリアン変数は、真理値の true（真）と false（偽）という 2 値をとるデータ型です。デフォルト値は false となっています。下図は、二つのブーリアン変数の宣言を示しています。

コード例

以下に示したコード例では、ブーリアン変数 bvar の値がチェックボックスによって制御されます。もし bvar が true（真）ならば、プロットグループ 4（pg4）が graphics1 にプロットされます。そうでない場合には、プロットグループ 1（pg1）がプロットされます。

```
if ( bvar ) {
    useGraphics ( model . result ( "pg4" ) , "graphics1" ) ;
} else {
    useGraphics ( model . result ( "pg1" ) , "graphics1" ) ;
}
```

整数と倍精度

整数と倍精度の変数は、その値がそれぞれ整数か倍精度であるということ以外は、文字列と同様です。

配列 1D

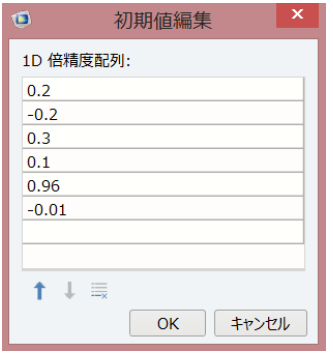
配列 1D ノードでは、フォームオブジェクトとメソッドで使用する文字列、ブーリアン、整数、または倍精度の配列（1 つ以上の要素数）を宣言します。1D の配列の要素数には、制限が全くありません。例えば、行ごとに一つの変数を持つテーブルにおいて、その一列を格納するために 1D の配列を用いることができます。この**設定**ウィンドウには一つのテーブル設定だけがあり、その行ごとに配列変数を指定します。

下図では、xcoords と ycoords の二つの倍精度配列が宣言されています。

新規要素値の欄には、テーブルフォームオブジェクトに行が追加された時に、配列の新しい要素が割り当てられます。文字列、ブーリアン、および整数の配列においても、この倍精度の場合と同様です。

初期値

1D の配列の初期値は、任意の長さで構いません。その初期値を編集するには、**変数リスト**の下にある**初期値編集**のボタンをクリックします。これによって、個々のコンポーネントの値を入力することができるダイアログボックスが開かれます。下図には、倍精度の 1D の配列の例を示しています。



配列の記述方法

配列の定義は、波括弧で始まり、波括弧で終わります({ })。個々の要素はコンマによって区切ります。配列要素の中に特殊な文字(例えばスペースやコンマ)が必要な場合は、その文字をシングルクオーテーション(')で囲みます。下表は、配列 1D のいくつかの例を示しています。

配列の記述例	説明
{ 1 , 2 , 3 }	要素が 1, 2, 3 の 3 要素の配列
{ }	空の配列
{ 'one , two' , 'three by four' }	特殊な文字を含んだ 2 要素の配列
{ { 1 , 2 , 3 } , { 'one , two' , 'three by four' } }	3 要素の配列と 2 要素の配列で構成される、全体として 2 要素の配列

配列 2D

配列 2D ノードでは、一つ以上の配列 2D を宣言します。フォームオブジェクトとメソッドを使って、その宣言をアクセスすることができます。下図は、倍精度の配列 2D として宣言された xycoords の設定を示しています。

初期値

2D 配列の初期値(デフォルト値)は、任意のサイズです。その初期値を編集するには、**変数リスト**の下に

ある**初期値編集**のボタンをクリックします。これによって、配列の各々のコンポーネントの初期値が入力できるダイアログボックスが開かれます。下図には、倍精度の配列 2D の例を示しています。

配列の記述方法

下表には、配列 2D のいくつかの例を示しています。

配列の記述例	説明
{{}}	空の配列 3D
{{"5","6"},{"7","8"}}	文字列の 2×2 行列
{{1,2,3},{4,5,6}}	倍精度の 2×3 行列

配列 2D において、最初のインデックスが列と一致しているので、{{1、2、3}、{4、5、6}} は、以下の行列に相当します。

1 2 3

4 5 6

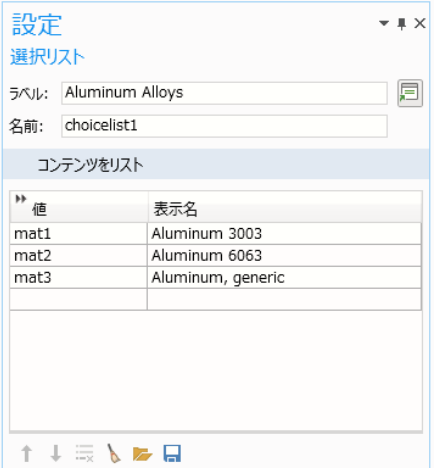
そして、もし上記の 2×3 行列が配列 2D の変数 arr に格納されると仮定すると、その要素 arr[1][0] は 4 です。

2D 配列の**初期値**をインタラクティブに定義するには、**列数**の**未定義**オプションを選択します。**初期値編集**ボタンをクリックすると、下図に示すように、行数と列数をインタラクティブに変更できるダイアログボックスが開きます。

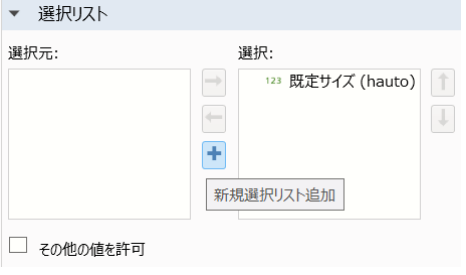
選択リスト

選択リストノードで設定されるリストは、コンボボックス、ラジオボタン、またはリストボックスで使われます。選択リストの**設定**ウィンドウには、**ラベル**、**名前**、および**値**と**表示名**の欄があるテーブルの設定があります。最初の列にはプロパティ値(**値**)を入力し、それに対応したユーザへの表示用テキスト(例えばコンボボックスのリスト項目)を 2 番目の列(**表示名**)に入力します。**値**は常に文字列として認識されます。

下図に示した設定例の場合、mat1 がコンボボックスから返される時には、“mat1”という文字列となります。



宣言ノードを右クリックして選択リストを作成する代わりに、下図に示すように、選択リストを使うフォームオブジェクトの設定ウィンドウにある**新規選択リスト追加**のボタンをクリックする方法もあります。

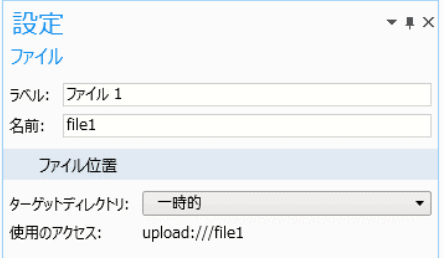


アクティベーション条件

選択リストノードを右クリックして、サブノードとして**アクティベーション条件**を追加することができます。**アクティベーション条件**は、変数の値を条件として二つ以上の選択リストを切り換えるために使われます。**アクティベーション条件**を利用した選択リストの使用例は、211 ページの「材料変更へのコンボボックスの利用」を参照してください。

ファイル

ファイルの宣言は、主に、メソッドのコードとして組み込まれたメソッド importFile を使うことによってファイルをインポートする場合に使われます。メソッド importFile やファイル処理の他のメソッドの詳細については、303 ページの「ファイルメソッド」を参照してください。実はさらに、**ファイル**宣言ノードは、**ファイルインポート**オブジェクトからも参照して使われます。下図には、ファイル宣言の**設定**ウィンドウを示しています。



ユーザによって選ばれたファイルは、フォームオブジェクトや upload:///file1、upload:///file2 などの記述方法によってメソッドの中で参照することができます。その時に、ファイルハンドル名(file1、file2 など)が、ユーザによって実行時に指定される実際のファイル名を参照するのに使われます。

ファイル宣言とファイル処理の詳細については、279 ページの「付録 C—ファイル処理とファイルスキーム表記法」を参照してください。

単位セット

単位セットノードは、コンボボックス、ラジオボタン、またはリストボックスで単位を変更するために使用することができるリストを含んでいます。単位セットの**設定**ウィンドウでは、**単位グループ**と**単位リスト**の二つのセクションがあります。

単位グループのテーブルの各行では、アプリケーションのユーザインタフェースのコンテキストで特定の意味を持つ単位のコレクションが単位グループとして表わされています。各列では、単位の一つのグループが**値**と**表示名**に分類されて表わされています。

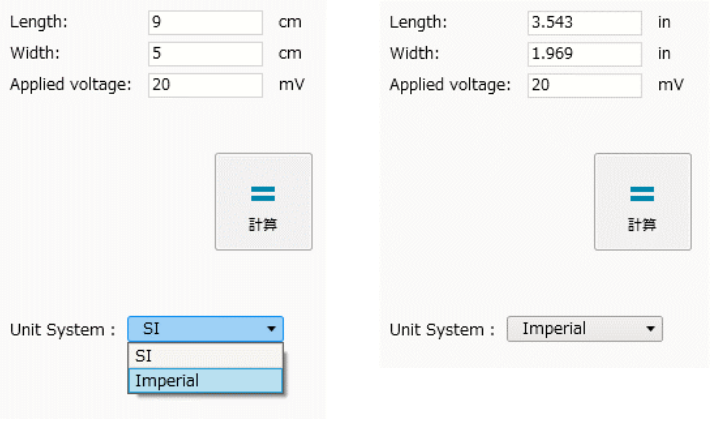
単位リストのテーブルの各行は、列ごとに同じ次元の単位が含まれる単位のリストです。例えば、mm、cm、dm、m、km といった列に分類されます。**単位リスト**テーブルの見出しは**名前**であり、**表示名**は**単位グループ**のセクションで定義されているものになります。単位リストでは、**単位セット**を参照するフォームオブジェクトがアプリケーション実行時に切り換えることができる単位を指定します。

上図は、メートル法とヤードポンド法の単位間の切り換えを可能にするアプリケーションのための**単位セット**の利用を示しています。この例では、SI と Imperial の二つの単位グループが定義されています。**単位セット**の**ラベル**は、Unit System に変更されています。

値の列には、単位グループの現在の選択を表す文字列値が含まれています。これらの文字列の値はメソッドから操作することができます。**表示名**の列は、ユーザインタフェースに表示される文字列です。**初期値**のリストには、デフォルトの単位グループが含まれています（上記の例では SI）。

上記の例では、**単位リスト**のテーブルには、**名前**、**SI**、および **Imperial** の三つの列があります。**SI** と **Imperial** の列は、**単位グループ**セクション内のグループに基づいて同時に作成されます。テーブルの各行は、この例では長さと電位といったように、物理量に対応しています。テーブルの各列は、それぞれ可能な、長さの単位と電位の単位に対応しています。

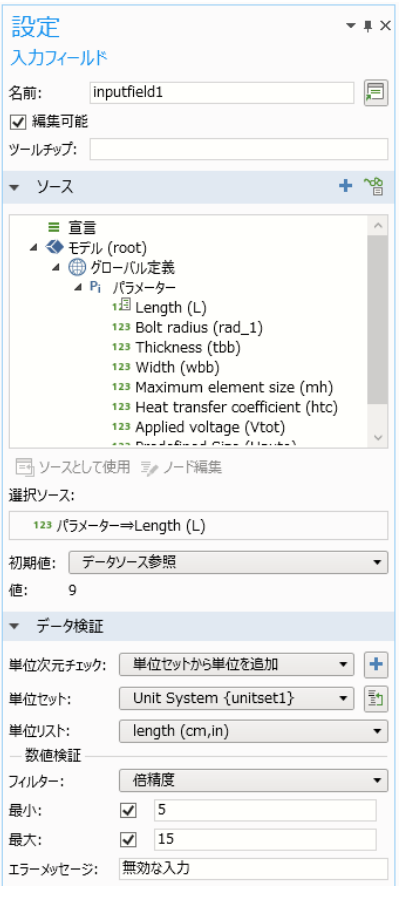
下図は、コンボボックスフォームオブジェクトによって **SI** と **Imperial** の単位グループを選択するアプリケーション例を示しています。

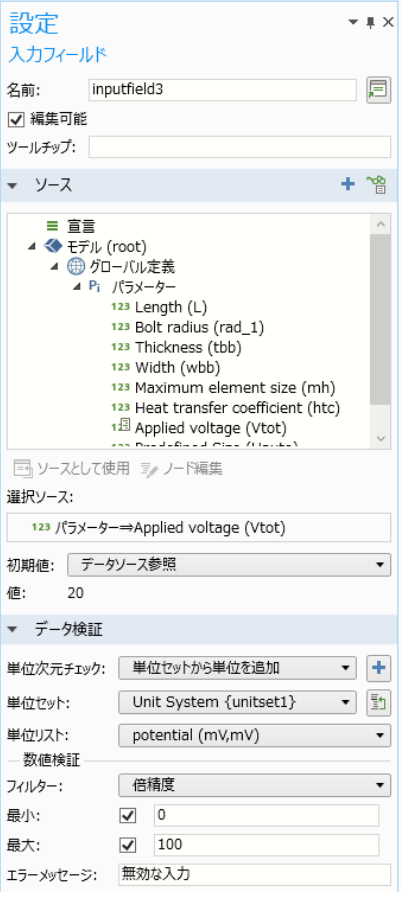


下図は、**ソース**としての上記の例の**単位セット**を使用しているコンボボックスの**設定**ウィンドウを示しています。

このようにして、単位選択用のコンボボックスを作成するために、**単位セット**が**選択リスト**の代わりに使われます。コンボボックスの代わりに、同様の方法でリストボックスまたはラジオボタンオブジェクトを使用することができます。

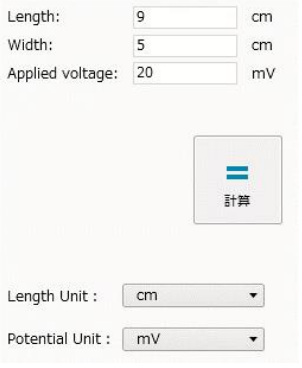
下の二つの図は、**Length** と **Applied voltage** の二つの入力フィールドに対応する**設定**ウィンドウを示しています。

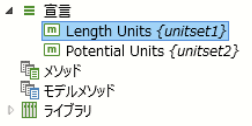




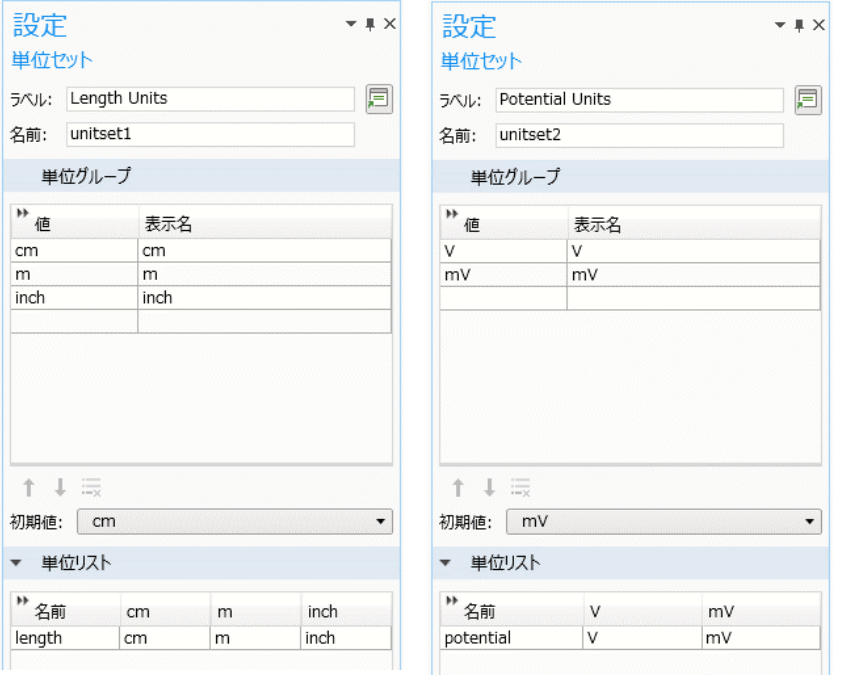
単位次元チェックは、**単位セットから単位を追加**に設定されています。**単位セット**は、**Unit System {unitset1}**（この例で使用されている**単位セット**宣言のユーザ定義ラベル）に設定されています。**単位リスト**は、それぞれ、**length** と **potential** に設定されています。**単位セットから単位を追加**を使用する場合は、（**データ検証**セクションにある）**数値検証**では、**単位セット**の**初期値**を参照しています。この場合はそれぞれ、**cm** と **mV** です。**最小**値と**最大**値はアプリケーションが実行された際に自動的にスケーリングされ、単位はアプリケーションのユーザによって変更されます。入力フィールドオブジェクトの設定についての詳細は、79 ページの「入力フィールド」を参照してください。

下図は、長さと電位の単位をそれぞれ設定する二つの**単位セット**宣言の使用例を示します。

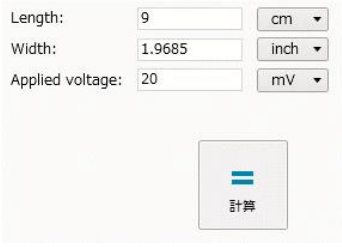




下図は、その**単位セット**宣言に対応する**設定**ウィンドウを示しています。



この例では、三つの**単位セット**宣言を使用して、**Length** と **Width** の入力フィールドの個々の長さの単位の設定を備えることができます。次の図は、三つのコンボボックスが単位ラベルを交換するために使用されている例を示しており、それぞれのコンボボックスは別個の**単位セット**宣言をソースとして使用しています。

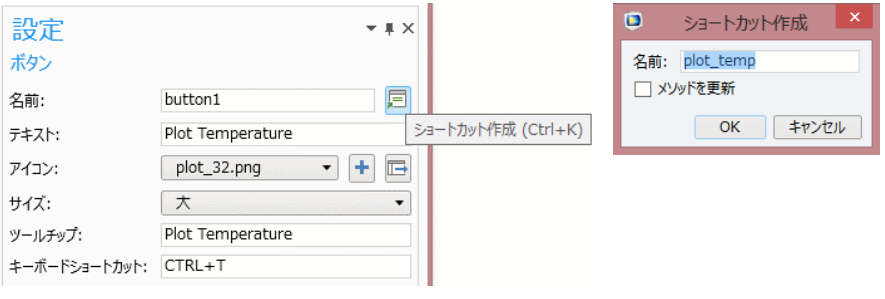


より多くの柔軟性が必要とされる場合には、**選択リスト**と**単位セット**の使用を組み合わせることができます。例えば、コンボボックスにおいて、**選択ソース**（文字列）として**単位セット**を使用し、**単位セット**ではない**選択リスト**を選択することができます。

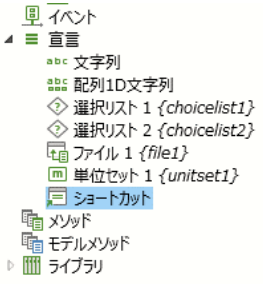
ショートカット

フォームオブジェクトと他のユーザインタフェースのコンポーネントは、特定の構文を使用してメソッドで参照されます。例えば、デフォルトの名前付けスキームを使って、form3 / button5 は form3 にある button5 という名前のボタンを参照し、form2 / graphics3 は form2 にある graphics3 という名前のグラフィックスオブジェクトを参照します。また、フォームやフォームオブジェクトのデフォルト名を変更することもできます。例えば、form1 があなたのメインフォームである場合、その名前を main に変更することができます。

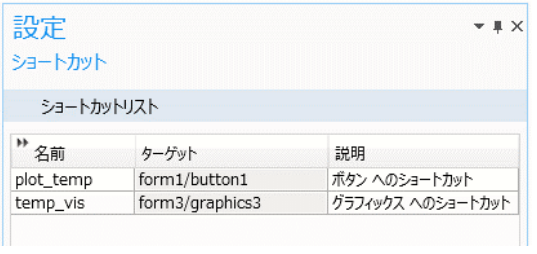
メニュー、リボン、ツールバー項目だけでなく、フォームオブジェクトの名前による参照を簡素化するには、名前をカスタマイズしてショートカットを作成することができます。オブジェクトまたは項目の**設定**ウィンドウで、**名前**フィールドの右にあるボタンをクリックしてお好みの名前を入力します。



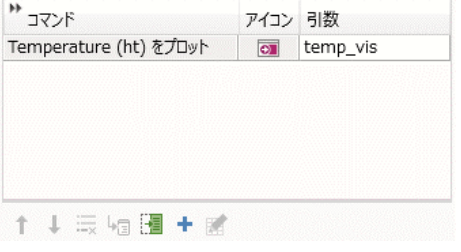
ショートカットを作成や編集するには、キーボードショートカットの Ctrl+ K を使用することもできます。作成した全てのショートカットは、アプリケーションツリーの**宣言**ノードの下の**ショートカット**ノードで利用できるようになります。



以下の**ショートカット**の**設定**ウィンドウでは、plot_temp と temp_vis の二つのショートカットが、それぞれボタンとグラフィックスオブジェクト用に作成されています。



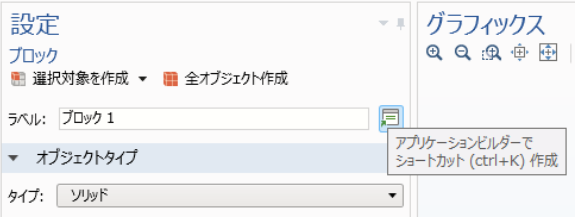
ショートカットは、他のフォームオブジェクトで、あるいはメソッドエディターのコードで参照することができます。下図は、ショートカット temp_vis が温度プロットへの入力引数として使用されている例を示しています。



オブジェクトが名前変更、移動、コピー、および複製された時、そのショートカットは自動的に更新されます。それらは、ちょうど文字列、整数、倍精度、およびブーリアンの宣言のように、読み取り専用の Java®変数としてアプリケーションのメソッドで利用することができます。

ショートカットを使用することによって、アプリケーションのユーザインタフェースの作りを変更する際に、メソッドエディターコードの調整が必要なくなります。この点で、ショートカットの利用をお勧めします。

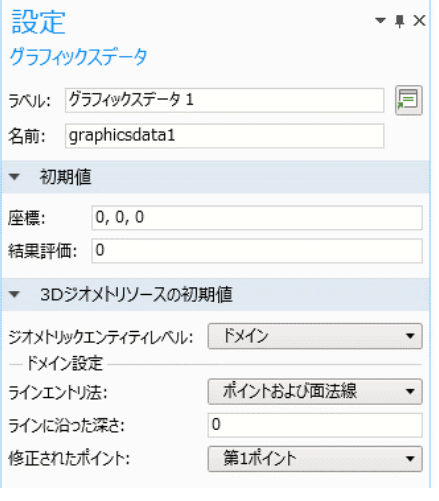
また、ショートカットは、アプリケーションビルダでの利用に対して、モデルビルダでも利用できます。モデルツリーノードの**設定**ウィンドウで、**ラベル**フィールドの右側にあるボタンをクリックして、選択した名前を入力します。



ショートカットのカスタム名は、メソッドのグローバル変数として使用可能になり、コードや新規メソッドを記録するときなどに使用されます。

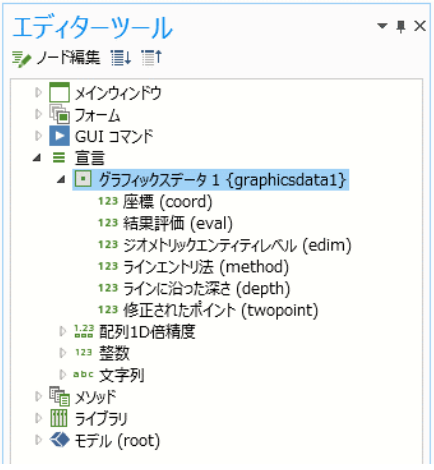
グラフィックスデータ

グラフィックスデータの宣言ノードは、ユーザによるマウスクリックに基づいてグラフィックスオブジェクトから特定の座標のデータを選択するために使用されます。下図は、対応する**設定**ウィンドウを示しています。

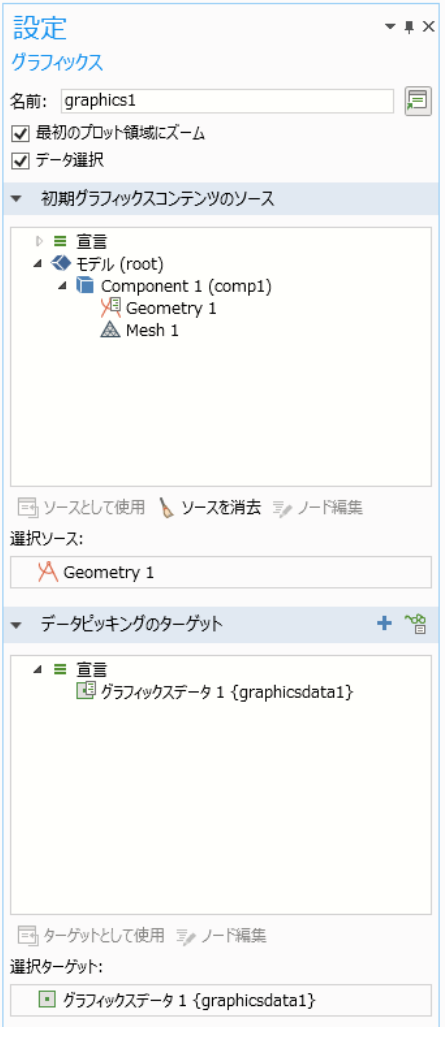


初期値セクションには、抽出されたデータプロパティの**座標**と**結果評価**のデフォルト値が含まれています。**3D ジオメトリソースの初期値**セクションには、グラフィックスオブジェクトの**初期グラフィックスコンテンツのソース**がジオメトリノードに設定されている場合、使用可能な選択方法の設定が含まれています。

グラフィックスデータ宣言のさまざまなプロパティは、下図に示すようにエディターツールウィンドウから使用できます。



グラフィックスデータ宣言ノードを使用してデータを選択するには、下図に示すように、グラフィックスオブジェクトの**設定**ウィンドウで**データ選択**のチェックボックスをオンにし、**データピッキングのターゲット**として**グラフィックスデータ**ノードを選択します。

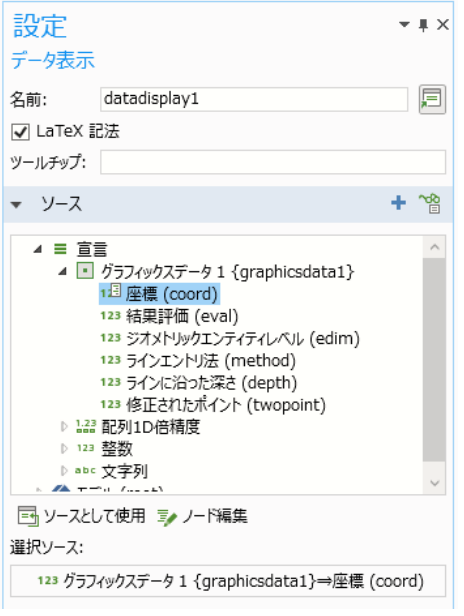


結果からのグラフィックスデータ

グラフィックスオブジェクトの**初期グラフィックスコンテンツのソース**がプロットグループのノードに設定されている場合、**結果評価**値はマウスポインタによって決定された位置のフィールド値に対応していま

す。**座標**値はその位置の座標に対応しています。モデルビルダーでは、これは**評価 2D** または**評価 3D** のテーブルに表示されるデータに対応しています。

下図は、**座標**プロパティが**ソース**として使用されているデータ表示オブジェクトを示しています。



また、**座標**プロパティを配列入力オブジェクトの**ソース**として使用することもできます。**結果評価**プロパティは、データ表示および入力フィールドオブジェクトを含むいくつかのフォームオブジェクトの**ソース**として使用することができます。

ジオメトリからのグラフィックスデータ

ジオメトリックエンティティレベル、**ラインエントリ法**、**ラインに沿った深さ**、および**修正されたポイント**は、ジオメトリオブジェクトの**初期グラフィックスコンテンツのソース**が 3D ジオメトリノードに設定されている場合にのみ適用されます。**ジオメトリックエンティティレベル**が**ドメイン**に設定されている場合、これらの設定は**ドメインポイントプローブ**と同じポイント選択方法を提供します。**ジオメトリックエンティティレベル**が**境界**に設定されている場合は、**境界ポイントプローブ**と同じです。**ラインエントリ法**、**ラインに沿った深さ**、**修正されたポイント**の設定は、**ジオメトリックエンティティレベル**が**ドメイン**に設定されている場合にのみ適用されます。

メソッドエディター

メソッドエディターは、モデルビルダーのモデルツリーノードにある標準実行コマンドだけでは不可能な動作をコードで記述するために使われます。例えば、ループの実行、入出力の処理、メッセージや警告のアプリケーションユーザへの発信などのメソッドです。

COMSOL のメソッドの記述には Java® プログラミング言語が使われており、全ての Java® 記述方法と Java® ライブラリを使用することができます。Java® ライブラリに加え、アプリケーションビルダー自体にもアプリケーションを構築してモデルオブジェクトを変更するためのライブラリが組み込まれています。モデルオブジェクトは、アプリケーションに組み込まれて基礎を成している COMSOL マルチフィジックスモデルの状態を格納するデータ構造です。これらの組み込みメソッドについての詳細情報は、302 ページの「付録 E—組み込みメソッドライブラリ」と *Application Programming Guide* を参照してください。

アプリケーションビルダーのアプリケーションツリーの内容は、モデルオブジェクトの重要な部分であるアプリケーションオブジェクトを介してアクセスされます。メソッドエディターを使ってコードを記録して記述することができ、ボタンのテキスト、アイコン、色、フォントといったような実行中のアプリケーションのユーザインタフェースの外観を直接アクセスして変更することもできます。

メソッドにはアプリケーションメソッドとモデルメソッドの二種類があります。アプリケーションメソッドは、実行中のアプリケーションのモデルオブジェクトを変更しますが、現行のセッションでモデルビルダーによって表されるモデルオブジェクトは変更できません。

モデルメソッドはアプリケーションメソッドと似ていますが、重要な違いは、現行のセッションでモデルビルダーが表すモデルオブジェクトを直接変更するという点です。

アプリケーションメソッドには、グローバルとローカルがあります。グローバルメソッドはアプリケーションツリーに表示され、全てのアプリケーションメソッドとフォームオブジェクトからアクセスすることができます。モデルメソッドはアプリケーションメソッドから呼び出すことができますが、逆の方法で呼び出すことはできません。ローカルメソッドはフォームオブジェクトまたはイベントに関連付けられ、それぞれに対応した設定ウィンドウから開くことができます。ローカルメソッドの詳細については、172 ページの「ローカルメソッド」を参照してください。

この章の内容は、アプリケーションメソッドとモデルメソッドの両方に適用されますが、例外はあります。モデルメソッドの詳細については、174 ページの「モデルメソッド」を参照してください。

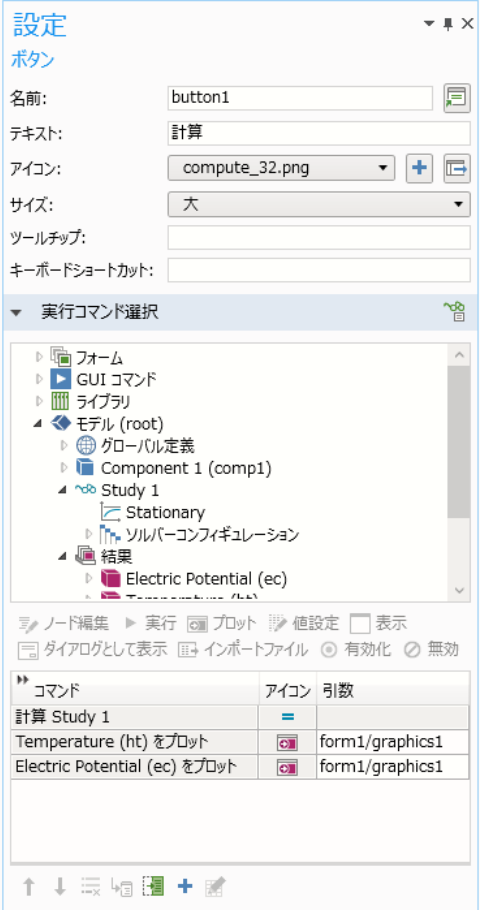
メソッドのコードを作成する上で、利用可能な多くのツールやリソースがあります。これらは以降の章で説明されており、例えば、これによってコードのブロックをコピーペーストしたり自動生成することができるようになれば作成効率が上がります。

コマンドシーケンスのメソッドへの変換

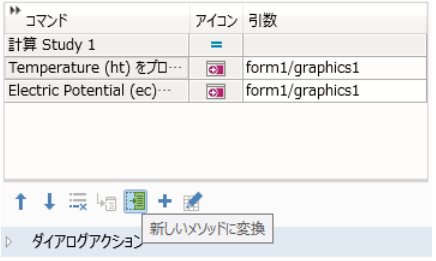
フォームエディターにおいて、設定ウィンドウにあるコマンドシーケンス表示の下側にある**新しいメソッドに変換**のボタンをクリックします。コマンドシーケンスは、自動的に同等のアプリケーションメソッドに変換されます。

計算ボタンを作成し、計算が終わったときに音で警報を出したい場合に、これをメソッドエディターを使ってどのように実現するかをみていきましょう。

この章の後半では、これをメソッドエディターを使わずにどのように実現するかについても学びます。下図には、**計算**ボタンの**設定**ウィンドウを示しています。

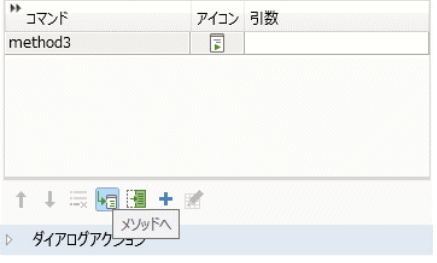


コマンドシーケンスの下側の**新しいメソッドに変換**のボタンをクリックします。

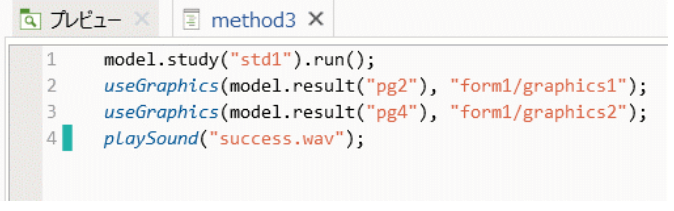


この例では、コマンドシーケンスがメソッド method3 に変換されています。

メソッドへのボタンをクリックすると、アクティブな method3 のタブとしてメソッドエディターが開きます。

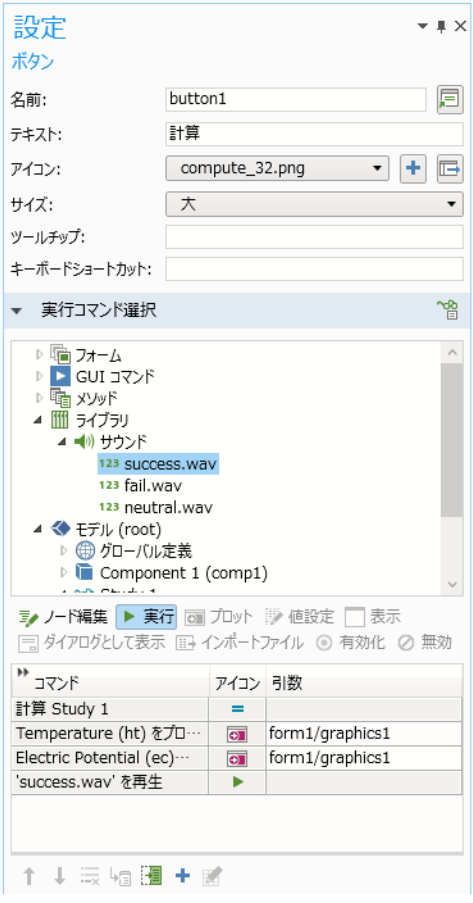


メソッドエディターで、下図に示した記述方法を使って、COMSOL のサウンドライブラリから入手したサウンドファイル success.wav を再生するため、組み込みメソッド playSound を呼び出します。



新たに追加されたコードのラインは、左側に表示された緑色のバーによって示されています。

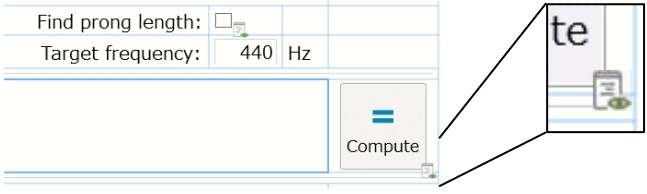
上記の例では、必ずしもメソッドエディターを使う必要がないことに注意してください。下図に示したように、実行コマンド選択のツリーで**ライブラリ > サウンド**の下にあるファイル success.wav を選択し、ツリーの下の**実行**のコマンドボタンをクリックしてコマンドシーケンスに追加します。



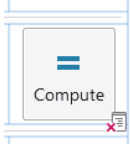
しかし、このような対応が可能なコマンドシーケンスノードが存在しない場合があり、そのための組み込みメソッドが多く用意されています。詳細については、302 ページの「付録 E—組み込みメソッドライブラリ」を参照してください。

関連付けられたメソッドを持つフォームオブジェクト

メソッドが関連付けられたフォームオブジェクトには、下図に示すようにフォームエディタ—に特別なアイコンが表示されます。この例では、**Find prong length** のチェックボックスと **Compute** ボタンにメソッドが関連付けられています。

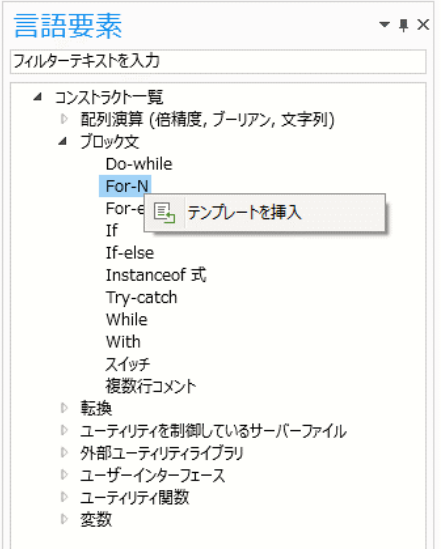


そのフォームオブジェクトの上で Ctrl+Alt+click の操作を行うと、メソッドエディターのメソッドが開かれます。もしフォームオブジェクトに関連付けられたメソッドが存在しない場合には、そのフォームオブジェクトに関連付けたローカルメソッドが新規に作成され、そのメソッドエディターが開かれます。関連するメソッドにコンパイルエラーがある場合は、下図に示すように、異なるアイコンで表示されます。



言語要素ウィンドウ

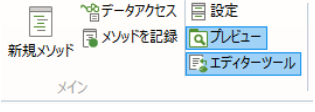
メソッドエディターの**言語要素**ウィンドウでは、いくつかの言語構成要素のリストが表示されます。そのリストにある項目の一つをダブルクリック、または右クリックすると、選択中のメソッドにテンプレートのコードが挿入されます。



184 ページの「よく使うオペレータ検索の例」も参照してください。

メソッドエディターでのエディターツール

エディターツールウィンドウを表示するには、**方法**タブの**メイン**のグループにある**エディターツール**ボタンをクリックします。



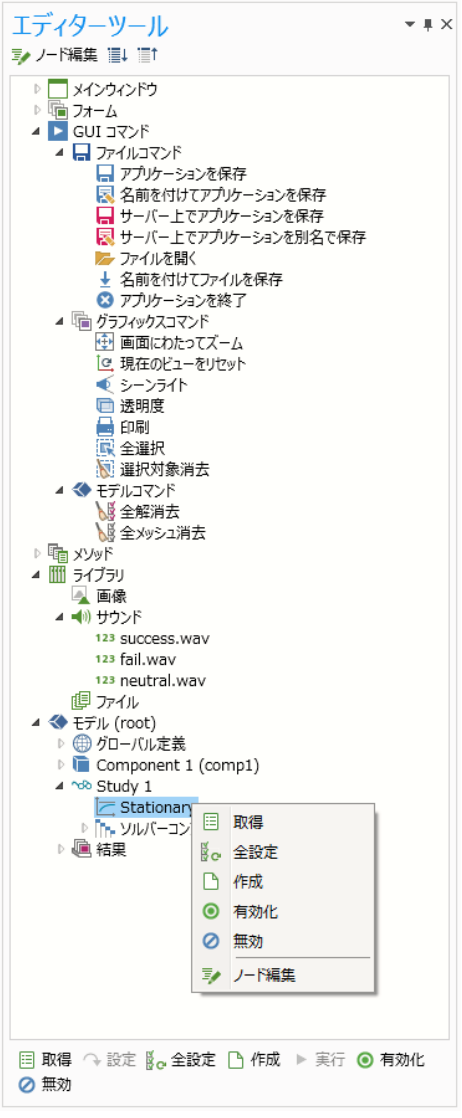
メソッドエディターで**エディターツール**ウィンドウを使う場合、エディターツリーのノードを右クリックすると、そのノードに関連したコードを生成することができます。ノードによっては最大八つの項目が選択できます。

- **取得**
- **設定**

- **全設定**
- **作成**
- **実行**
- **有効**
- **無効**
- **ノード編集**

最初の七つの項目のうちのいずれかを選択すると、そのノードに対応したコードを現在選ばれているメソッドに追加することができます。**ノード編集**の項目を選択した場合は、対応するモデルツリーノードの**設定**ウィンドウに切り換わります。

下図は、ノードを右クリックした時に六つの項目が表示されている場合です。



ノードが選択された際に、エディターツールの下側のツールバーには、そのコードに対して生成可能な項目が表示されます。

また、**エディターツール**ウィンドウは、フォームエディターで作業するときにも重要なツールです。この詳細については、49 ページの「フォームエディターでのエディターツール」を参照してください。

キーボードショートカット

モデルオブジェクトを参照する次の 1 行のコードのメソッドについて考えてみましょう。

```
model . result ( "pg3" ) . feature ( "surf1" ) . create ( "hght1" , "Height" ) ;
```

“surf1”にマウスポインタを置き、キーボードの F11 を押すか、右クリックして**ノードへ**を選択するか、またはリボンにある**ノードへ**をクリックすると、該当する**サーフェス**（Surface）プロットのノードが**エディターツール**ウィンドウにおいてハイライト表示されます。**ノード編集**をクリックすると、その**設定**ウィンドウが開かれます。

キーボードショートカットに関する詳細について、299 ページの「付録 D—キーボードショートカット」を参照してください。

メソッドエディターでのデータアクセス

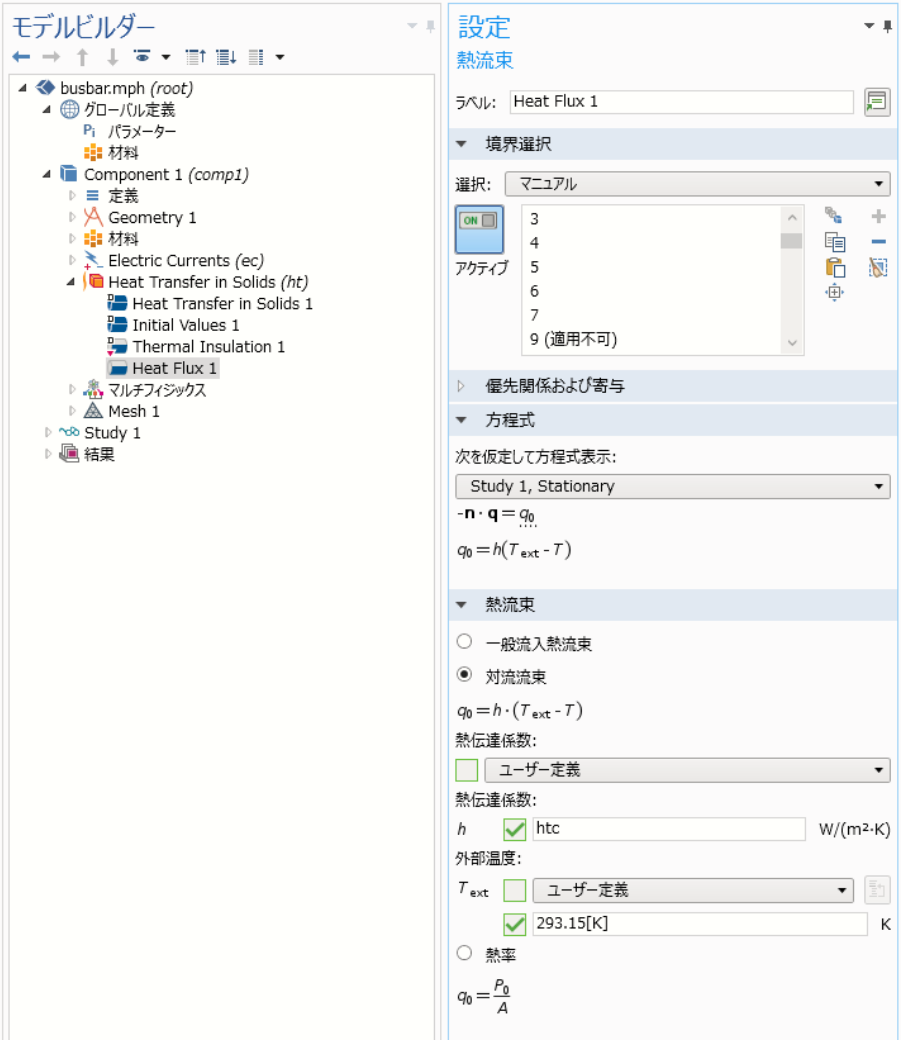
モデルツリーノードの個々のプロパティにアクセスするには、モデルビルダーのリボンの**デベロッパー**タブの**アプリケーション**セクションにある**データアクセス**ボタンをクリックします。

その代わりに、フォームオブジェクトの**設定**ウィンドウの**ソース**セクションのヘッダにある**データアクセス**ボタンをクリックすることによっても可能です。89 ページの、「フォームエディターでのデータアクセス」も参照してください。

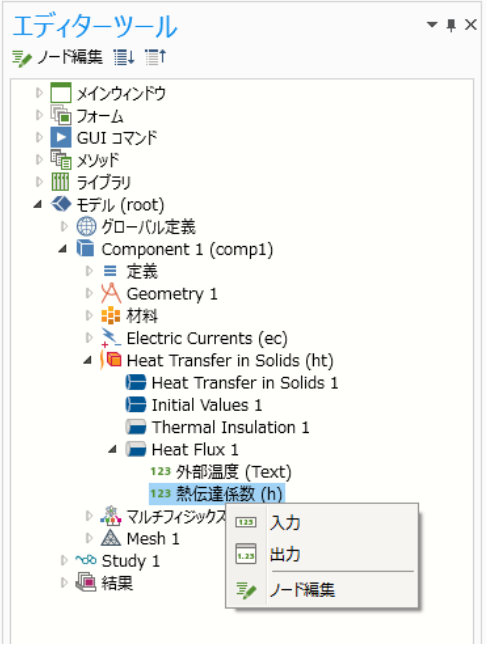
通常、モデルには、アクセス可能な数百あるいは数千ものプロパティが存在し、それを全てリスト表示するには多過ぎるので実用的ではありません。このため、**データアクセス**という方法がとられています。

モデルツリーノードをクリックした際、次の図の **Heat Flux** ノードの場合のように、モデルデータアクセスによるチェックボックスが設定ウィンドウの個々のプロパティの隣に表示されます。この例は、*Introduction to COMSOL Multiphysics* の中の説明で使われているブスバー（busbar）のチュートリアルモデルに基づいています。

下図では、**熱伝達係数**と**外部温度**のためのチェックボックスが選択されています。



その後、**エディターツール**ウィンドウに切り換えると、**Heat Flux** ノードの下にノードが追加作成されているのを確認することができます。下図に示すように、右クリックして**取得**または**設定**を選択することによって、アクティブなメソッドウィンドウでコードが生成されます。



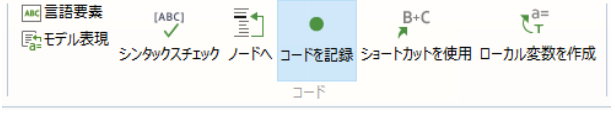
この例において、**熱伝達係数**と**外部温度**のプロパティに対する**取得**と**設定**によって、以下のコードが生成されます。

```
model . physics ( "ht" ) . feature ( "hf1" ) . getString ( "h" ) ;
model . physics ( "ht" ) . feature ( "hf1" ) . getString ( "Text" ) ;

model . physics ( "ht" ) . feature ( "hf1" ) . set ( "h" , "htc" ) ;
model . physics ( "ht" ) . feature ( "hf1" ) . set ( "Text" , "293.15[K]" ) ;
```

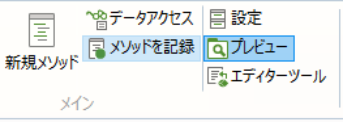
コードの記録

下図に示すように、メソッドエディター用のリボンの**コード**のグループにある**コードを記録**のボタンをクリックすることによって、モデルツリーを使って実行する一連の操作をコードとして記録することができます。



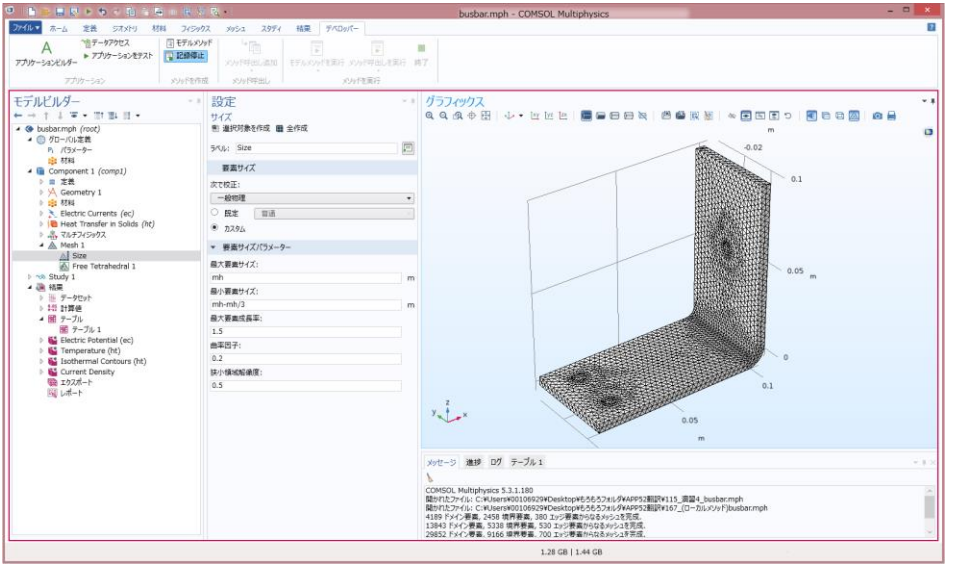
アプリケーションツリーでの特定の操作も記録することが可能です。それには、アプリケーション実行中にテキストラベルの色を変えるなどのユーザインタフェースを変更するために使われるメソッドが含まれます。

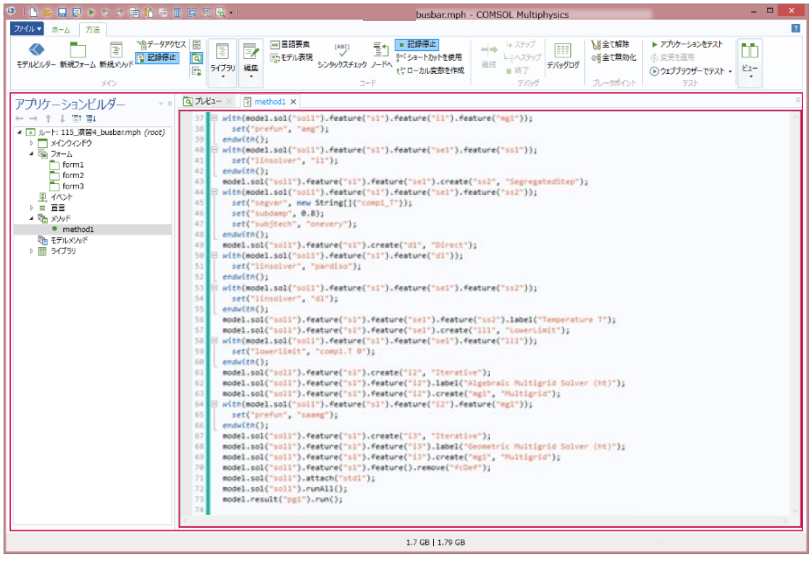
新規メソッドを記録するには、フォームエディターまたはメソッドエディター用のリボンの**メイン**セクションにある**メソッドを記録**のボタンをクリックします。



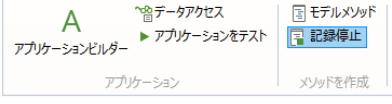
モデルビルダーのリボンの**デベロッパー**タブの**メソッドを記録**ボタンをクリックすることもできます。

コードの記録中、COMSOL デスクトップウィンドウは赤い枠で囲われた状態となります。





コードの記録を停止するには、モデルビルダーかアプリケーションビルダーのいずれかの**記録停止**ボタンをクリックします。



前の**データアクセス**の章では、ブスバーチュートリアルにおける**熱伝達係数**と**外部温度**プロパティの値をどのように設定するかについて説明しました。これと同様のコードを**コードを記録**を使って生成する手順を以下に示します。

- ブスバーモデル（MPH ファイル）で簡単なアプリケーションを作成します。
- モデルビルダーウィンドウ側で**新規メソッドをレコード**をクリックするか、またはメソッドエディターを開いた状態で**コードを記録**をクリックします。
- **熱伝達係数**の値を 5 に変更します。
- **外部温度**の値を 300[K]に変更します。
- **記録停止**をクリックします。
- このとき、あらかじめメソッドが開かれていなかった場合には、コードが記録された新規メソッドが開かれます。

上記ステップの結果、以下のコードが記録されます。

```
with ( model . physics ( "ht" ) . feature ( "hf1" ) ) ;
    set ( "h", "5" ) ;
    set ( "Text", "300[K]" ) ;
endwith ( ) ;
```

この場合、自動的に記録されるコードをよりコンパクトにするために、with( )ステートメントが使われています。with( )の使用方法についての詳細は、186 ページの「With ステートメント」を参照してください。

アプリケーションオブジェクトへの変更に対応するコードを生成するには、**コードを記録**または**メソッドを記録**を使用します。その後、フォームエディターに切り換え、例えば、フォームオブジェクトの外観を変更します。次のコードは、テキストラベルの色をデフォルトの**継承**から**青**に変更した場合です。

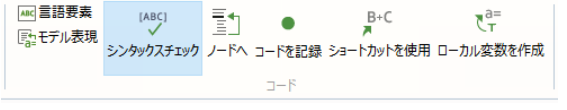
```
with  ( app . form ( "form1" ) . formObject ( "textlabel1" ) ) ;
    set ( "foreground" , "blue" ) ;
endwith  ( ) ;
```

モデルオブジェクトとアプリケーションオブジェクトの変更に関する詳細情報については、*Application Programming Guide* を参照してください。

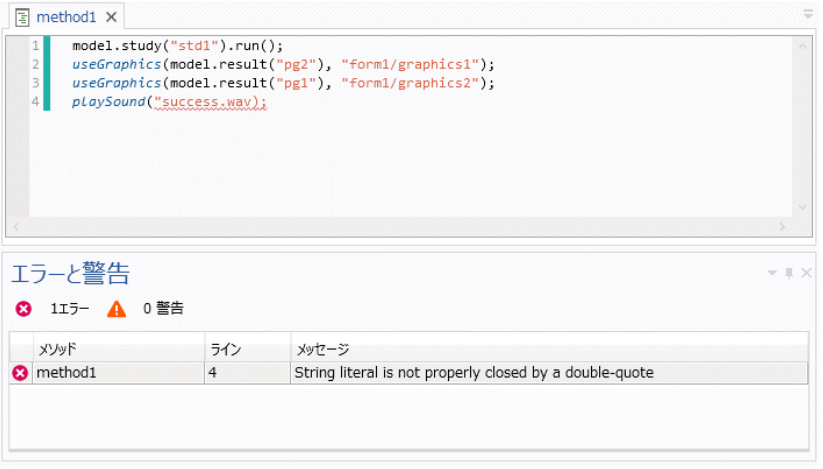
モデルオブジェクトをどのようにコードに記載するかを効率的に学習するには、コードを記録するツールを使うことが有効です。自動生成されたコードには、プロパティ、パラメータ、および変数の名前が示されています。モデルプロパティに新たなパラメータ値を割り当てるには、文字列や文字列に変換された数字を使います。記録中に**データアクセス**を使うことによって、例えば、get を使ってパラメータ値を取り出し、メソッドの中でその値を処理し、set を使ってモデルオブジェクトにそれを戻して設定するといったようなことが可能です。**データアクセス**の詳細情報については、157 ページの「メソッドエディターでのデータアクセス」を参照してください。

シンタックスチェック

リボンの**シンタックスチェック**をクリックして、**エラーと警告**ウィンドウのメッセージを表示させ、シンタックスエラーや未使用変数を確認することができます。

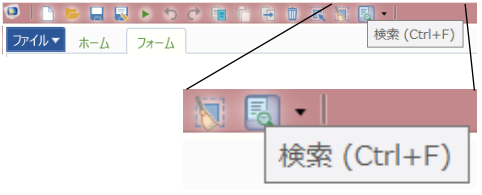


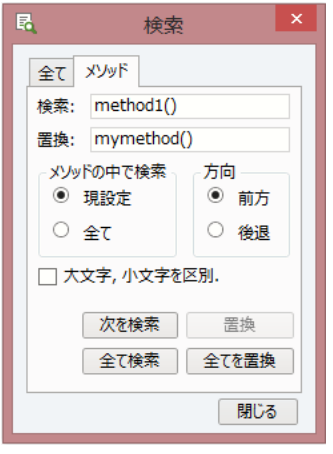
エラーと警告ウィンドウのメッセージに加え、下図に示すように、コード中でシンタックスエラー箇所が赤い波線で下線表示されます。



検索と置換

下図に示すように、クイックアクセスツールバーにある**検索**をクリックすると、メソッドの中の文字列の検索と置換のためのダイアログボックスが開かれます。



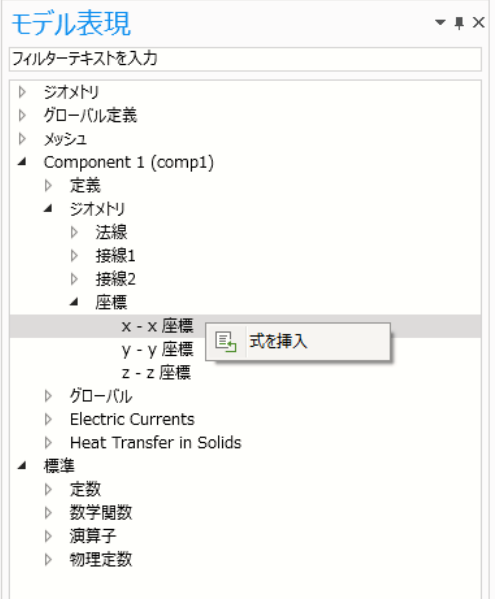


クイックアクセスツールバーは、COMSOL デスクトップユーザインタフェースにおいてリボンの上側の左寄りに配置されています。

このダイアログボックスの**全て**のタブでは、モデルビルダーとアプリケーションビルダーの両方に対して文字列と変数を検索するために利用されます。

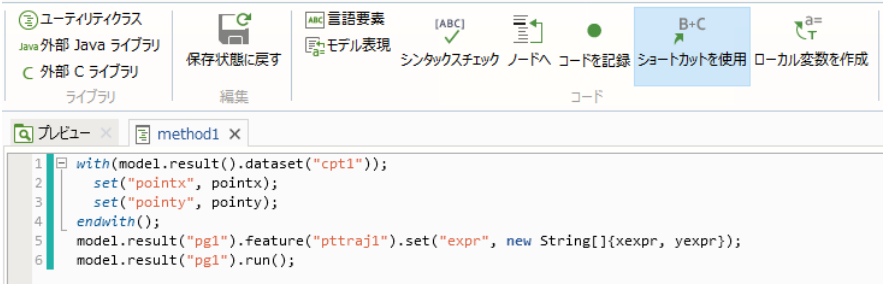
モデル表現ウィンドウ

メソッドエディターの**モデル表現**ウィンドウは、入出力の引数としてあらかじめ定義されているモデル側の表記リストを示しています。そのリストの項目の一つをダブルクリックまたは右クリックして、モデル表現を挿入することができます。



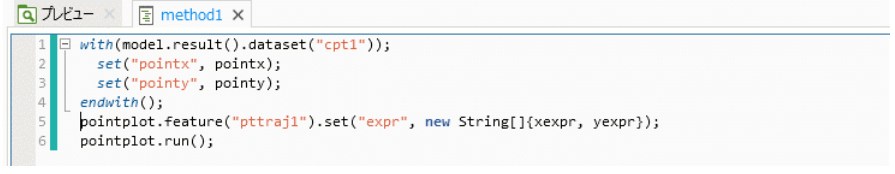
ショートカットの使用

以下の例を見ると、コードの最後の二行が model.result("pg1") で始まっていることに気付くはずです。

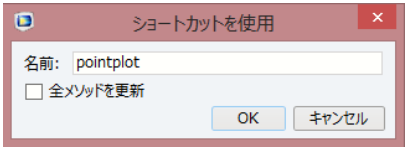


ショートカットを使用のボタンは、これらのインスタンスを変数名に置き換えてコードを簡素化してくれます。

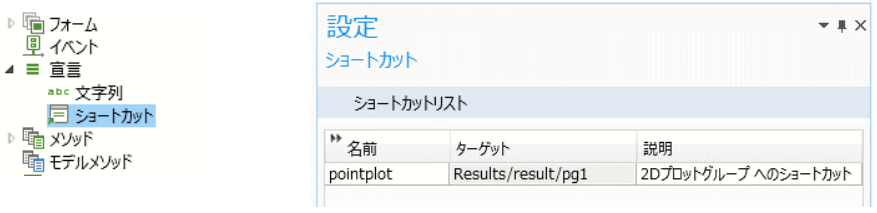
上記の例では、マウスポインタを model.result("pg1") の最初の記載位置に置きます。その状態で、**ショートカットを使用**のボタンをクリックすると、ソースコードが下図のように変換されます。



頭が model.result("pg1") で始まるコードは、変数 pointplot に置き換えられています。**ショートカットの使用**のボタンをクリックすると、**ショートカットの使用**ダイアログボックスが開き、**名前**のフィールドに適した変数名（この場合は pointplot）を入力することができます。



この変数は、**宣言**ノードにショートカットとして保存されます。下図には、対応する**設定**ウィンドウとともに示しています。



シンタックスのハイライト表示、コードの折り畳み、および字下げ

コードに記述する言語要素の表示には、いろいろなスタイルが使われています。下図にその例を示します。

```
64  with(model.result("pg1"));
65    set("looplevel", new String[]{"7"}); // 7th frequency
66  endwith();
67  useGraphics(model.result("pg1"), "graphics1");
68  zoomExtents("graphics1");
69
70  if (customProgress) {
71    setProgressBar("/progressform/progress1", 100);
72  }
73  else {
74    setProgress(100);
75  }
76  play_sound();
77
78  if (customProgress) {
79    closeDialog("progressform");
80  }
81  else {
82    closeProgress();
83  }
```

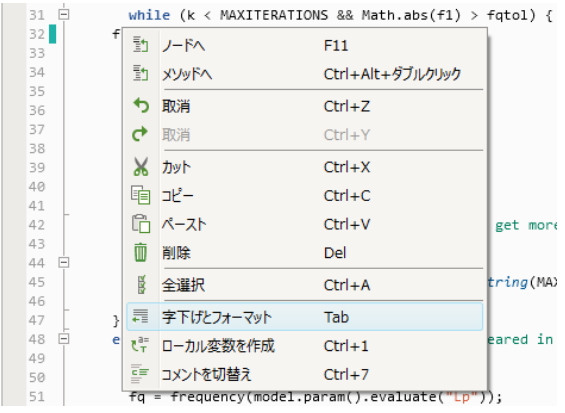
この例では、五つのスタイルが使われています:

- if、else、for、while、double および int といったキーワードは、太字（ボールド）、青色フォントで表示されています。
- 組み込みメソッドは、イタリックの青色フォントで表示されています。
- 文字列は、赤色フォントで表示されています。
- コメントは、緑色フォントで表示されています。
- その他のコードは、黒色フォントで表示されています。

環境設定のダイアログボックスで、シンタックスハイライトのテーマをカスタマイズ設定することができます。次の章 メソッドエディターの環境設定 を参照してください。

for、while、if、および else ステートメントがあるコードブロックに対して、そのコードを広げて全て表示したり折り畳んで縮めて表示したりすることができます。次の章「メソッドエディターの環境設定」で説明されているように、この機能を無効にすることもできます。

コードの記述の際にキーボードのタブキーを使うと、自動的にそのコードの行頭に必要な空白が挿入され、コードの先頭が右に字下げされます。代わりに、メソッドエディターで右クリックし、下図に示すように、**字下げとフォーマット**を選択することもできます。



また、キーボードのフォーカスがメソッドエディターからいったん離れた段階でも、この字下げと空白の挿入が自動的に行われます。**環境設定**の**メソッド**セクションにある**自動で字下げとフォーマット**のチェックボックスをクリアすることによって、この動作を無効にできます。

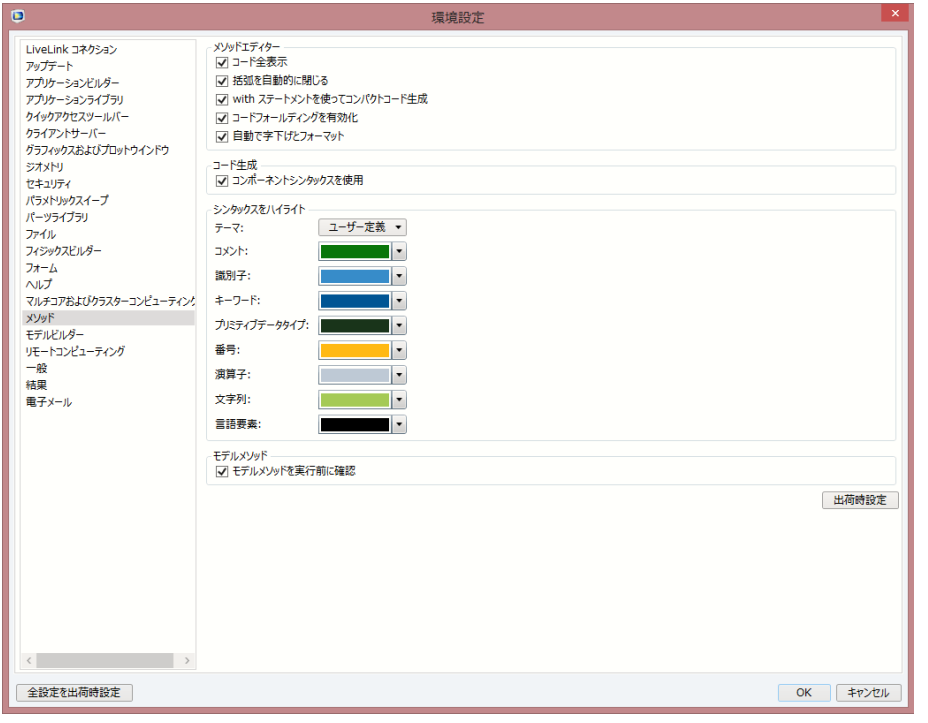
選択されているコードブロック全体に対して、右クリックして上記のコンテキストメニューを使用してコメントのオン/オフを切り替えることができます。これは、メニューから**コメントを切替え**を選択するか、あるいは Ctrl + 7 のキーボードショートカットによります。

メソッドの名前

メソッドの**名前**は、スペースを含まないテキスト文字列とします。この文字列には、文字、数字、および下線を含むことができます。予約語である root と parent の使用は許されません。また、Java® プログラミング言語のキーワードも使うことができません。

メソッドエディターの環境設定

メソッドの**環境設定**にアクセスするには、**ファイル > 環境設定** を選択し、表示されるダイアログボックスのメニューから**メソッド**セクションを選択します。



デフォルトでは、メソッドエディターは最低限必要なコードだけを表示します。メソッドの全てのコードを見るには、**コード全表示**のチェックボックスをチェックします。

括弧を自動的に閉じるのチェックボックスは、メソッドエディターが波括弧{ }、角括弧[ ]、丸括弧( )といった括弧閉じを自動的に追加するかどうかの設定です。

with ステートメントを使ってコンパクトコード生成のチェックボックスは、自動的にコードを生成する際に with ステートメントを活用するか否かの設定です。詳細については、186 ページの「With ステートメント」を参照してください。

コードフォールディングを有効化のチェックボックスをチェックすると、for、while、if、および else ステートメントが記述されているコードブロックに対して、そのコードを広げて全て表示したり折り畳んで縮めて表示したりすることができます。

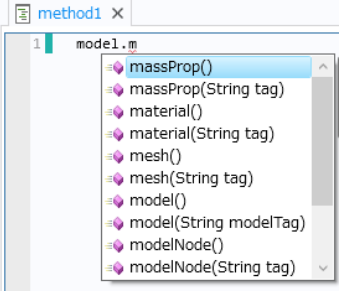
コンポーネントシンタックスを使用のオプションは、モデルコンポーネントのスコープも含むメソッドシンタックスを生成します。

シンタックスをハイライトのすぐ下の**テーマ**のリストには、あらかじめ**モダン**と**クラシック**の 2 つの選択項目があります。**ユーザ定義**を選ぶと、シンタックスハイライトモードにおける個々の言語要素に割り当てる色を定義することができます。

モデルメソッドを実行前に確認のオプションをオフにすると、モデルメソッドの実行時に確認しないようになります。

コード補完のための Ctrl+Space とタブ

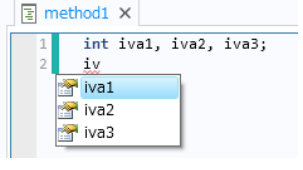
メソッドエディターでコードを記述する際、アプリケーションビルダーはそのコードの完成形を提案してくれます。入力の最中に、可能性のある完成形のリストが個々に開かれて表示されます。状況によっては、リストで入力候補を選択した際に、その詳細情報が別のウィンドウで表示されます。コードの完成形の表示要求は、キーボードショートカット Ctrl+Space の操作によります。例えば、モデルオブジェクトの部分にアクセスした場合、可能性のある完成形のリストが下図に示すように表示されます。



矢印キーを用いてリストの入力候補を選び、タブまたは Enter キーを押してその選択を確定します。

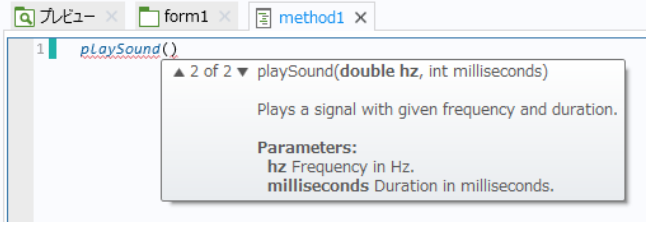
もしリストの選択肢が多い場合には、探している完成形の先頭の数文字をメソッドエディターに入力することによって、リストの選択肢を減らして見つけ易くします。

例えば、変数やメソッド名の先頭の数文字を入力して Ctrl+Space を押せば、可能性のある完成形が下図のように表示されます。



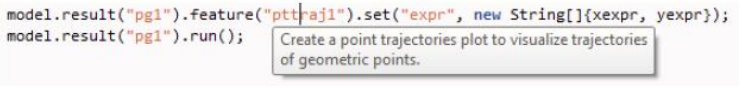
上記の例では、文字列 iv に一致する変数だけがリスト表示されています。この例は、そのメソッドの中で定義されているローカルな変数でも、完成形の提案としてリストに表示されることを示しています。

また、Ctrl + Space キー操作を使用することによって、直接モデルオブジェクトに関連していない組み込みメソッドの構文について学ぶことができます。コマンドの名前を入力し、Ctrl + Space キーを使用することによって、利用可能なさまざまな呼び出しシグネチャー情報のウィンドウを開くことができます。

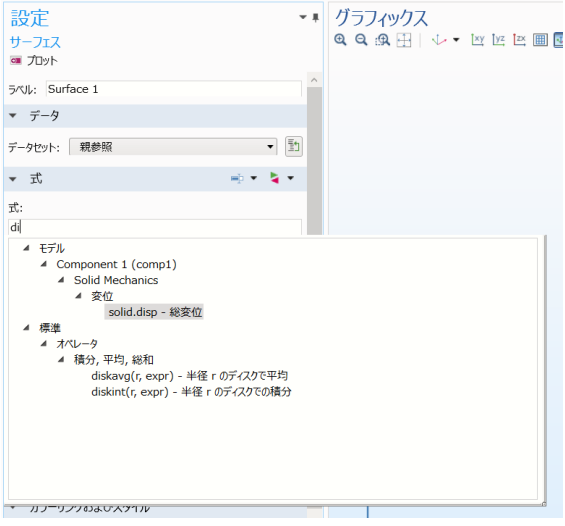


利用可能な組み込みメソッドのリストについては、154 ページに記述されている**言語要素**ウィンドウ、または 302 ページの「付録 E―組み込みメソッドライブラリ」を参照してください。

メソッドの呼び出し、プロパティ名、宣言、またはショートカットといったさまざまな部分をホバリングすると、類似情報がツールチップに表示されます。



また、キーボードショートカット Ctrl+Space は、モデルビルダーにおいても使われます。例えば、**結果**（Results）の中の**式**フィールドに入力する際、下図に示すように、変数に一致した候補を探すために Ctrl+Space を使用することができます。

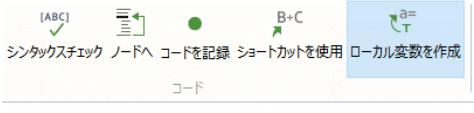


ローカル変数の作成

ローカル変数のタイプを自動的に設定することができます。例えば、

x = model.geom()

というコードを入力し、リボンのメソッドタブのコードグループにある**ローカル変数を作成**のボタンをクリックします。



その後、コードは次のように変更されます。

GeomList x = model.geom()

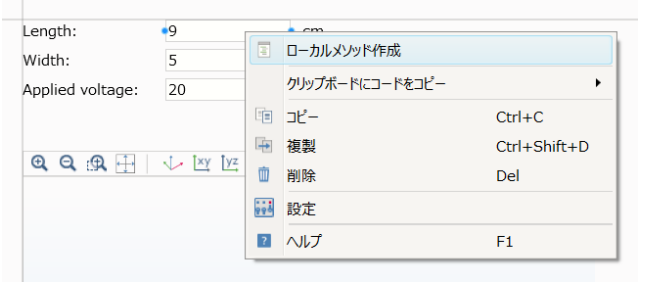
ここで、GeomList は model.geom()のデータ型です。

ローカルメソッド

ボタン、メニュー項目、およびイベントに対して、ローカルなメソッドを追加することができます。このローカルメソッドは、アプリケーションツリーにある**メソッド**ノードの下には表示されません。ローカルメソッドのメソッドウィンドウのタブ部分には、そのユーザインタフェースのコンポーネントへのパスが記載されています。
チェックボックスオブジェクトの場合の例を、下図に示します。

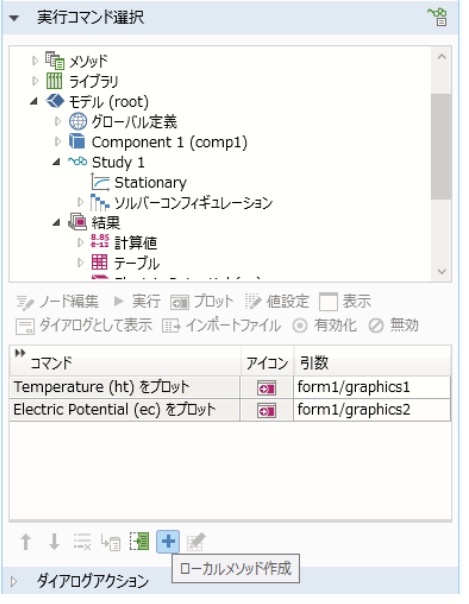
```
main: checkbox1: onDataChange
1   setFormObjectEditable("main/inputfield1", !findlength);
2   setFormObjectEditable("main/inputfield5", findlength);
3   setFormObjectEnabled("main/inputfield5", findlength);
```

フォームエディターでフォームオブジェクトを右クリックし、下図に示すように、表示されるメニューから**ローカルメソッド作成**を選択します。



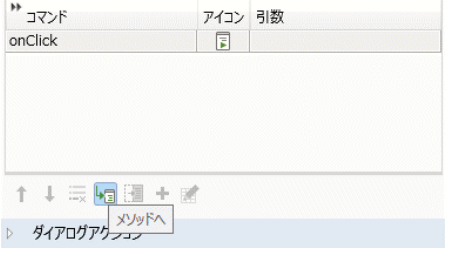
ボタン、メニュー項目、およびグローバルイベントのローカルメソッド

ボタン、リボン、メニュー、ツールバー項目、およびグローバルイベントにおいて、下図に示すように、コマンドシーケンスの下側にある**ローカルメソッド作成**のツールバーボタンをクリックすることにより、ローカルメソッドを追加作成することができます。



そのボタンの機能は、15 ページの「新規メソッドの作成」の章で説明されている内容と同様です。唯一の違いは、アプリケーションツリーにあるグローバルメソッドのリストには表示されないなローカルメソッドが作成されるという点です。それが作成されると、メソッドエディターに新規メソッドが開かれます。Ctrl+Alt+click のショートカットの操作によっても、ローカルメソッドを作成することができます。

メソッドへのボタンをクリックすると、ローカルメソッドが開かれます。下図に示すように、ローカルメソッドの呼び出しはボタンに関連付けられています。



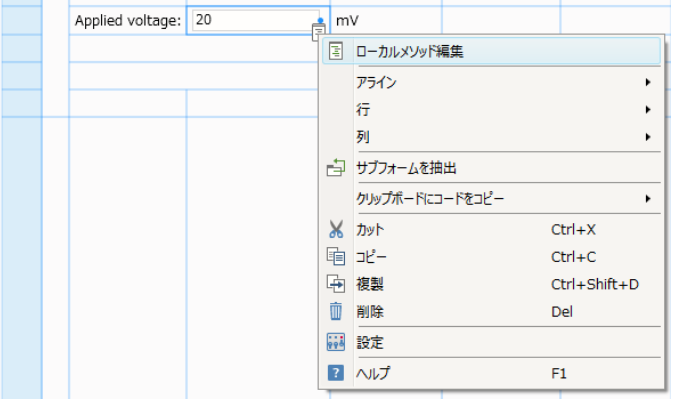
コマンドシーケンスでローカルメソッドがすでに定義されている場合は、**新しいメソッドに変換**を使うことができません。これは、ローカルメソッドの中に間違ったコードがある場合のリスクを避けるためです。

フォームとフォームオブジェクトイベントのローカルメソッド

フォームまたはフォームオブジェクトイベントに対してローカルメソッドを追加するには、**設定**ウィンドウの**イベント**セクションで**ローカルメソッド作成**のボタンをクリックします。下図に示すように、**データ変更時**のメソッド選択が**なし**から**ローカルメソッド**に変更され、メソッドエディターが開かれます。

メソッドエディターで既存のローカルメソッドを開くためには、**ソースへ**のボタンをクリックします。ローカルメソッドを削除するには、**ローカルメソッドを削除**のボタンをクリックします。

Ctrl+Alt+click の操作の代わりの方法として、フォームオブジェクトを右クリックし、表示されるコンテキストメニューから**ローカルメソッド編集**を選択することもできます。



詳細については、118 ページの「イベント」を参照してください。

モデルメソッド

モデルメソッドはアプリケーションメソッドと似ていますが、最も重要な違いは、現行のセッションでモデルビルダーが表すモデルオブジェクトを直接変更することです。アプリケーションメソッドと同様に、メソッドエディターを使用して編集されます。モデルメソッドを使用して、いくつかの手動の手順によるモデリング作業を自動化することができます。例えば、複数のスタディがあるモデルでは、最初に計算するスタディ

1 のプロセスのコードを記録します。その後、スタディ 1 からの解（Solution）に基づくスタディ 2 を計算するといった具合です。

アプリケーションメソッドの場合とは異なり、モデルメソッドには次のような追加の制限があります。

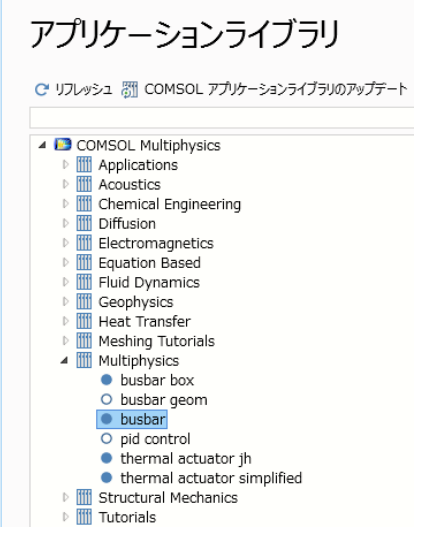
- モデルメソッドを使用してアプリケーションオブジェクトを変更することはできません。
- モデルメソッドは、アプリケーションメソッドまたは Java®ユーティリティクラスを呼び出すことはできません。
- **言語要素**ウィンドウの**ユーザインタフェース**のブランチ（分岐）に対応する組み込みアプリケーションメソッドの多くは、モデルメソッドからは使用できません。

モデルメソッドには、入力引数と出力引数があります。モデルメソッドへの入力引数は、**グローバル定義**の下に**メソッド呼出し**ノードを追加することで得られます。178 ページの「入出力の引数を持つメソッド」を参照してください。

計算後のレポート自動生成

モデルメソッドを使用する例として、解（Solution）を最初に計算してからレポートを生成するプロセスを考えてみましょう。これを自動化するには、モデルビルダーでこれに対応する操作を最初に記録した後にモデルメソッドを実行します。

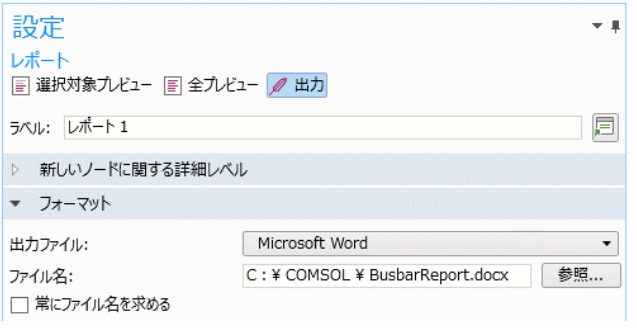
Introduction to COMSOL Multiphysics の中で掲載されているブスバーの例から始めましょう。下図に示すように、このサンプル MPH ファイルはアプリケーションライブラリからロードできます。



- モデルを**開く**。
- モデルツリーで、**結果**の下にある**レポート**ノードを右クリックします。

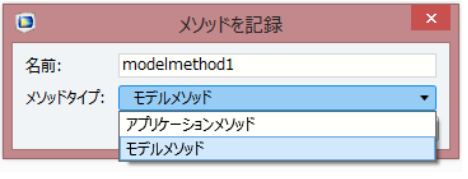
- **簡略レポート**を選択します。
- **出力フォーマット**を Microsoft® Word に変更します（この例はデフォルトの HTML フォーマットでも機能します）。
- **参照**ボタンをクリックし、システム上の書き込み権限がある場所でファイル名を選択します。例えば、次のようになります。

  C:\ COMSOL \ BusbarReport.docx
- **出力**をクリックしてレポートを生成し、ファイルに保存します。



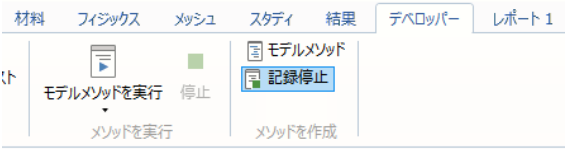
- Microsoft® Word ドキュメントを閉じます。
- （**モデルビルダー**の）リボンの**デベロッパー**タブをクリックし、**メソッドを記録**ボタンをクリックします。

- **メソッドを記録**のダイアログボックスで**モデルメソッド**を選択します。

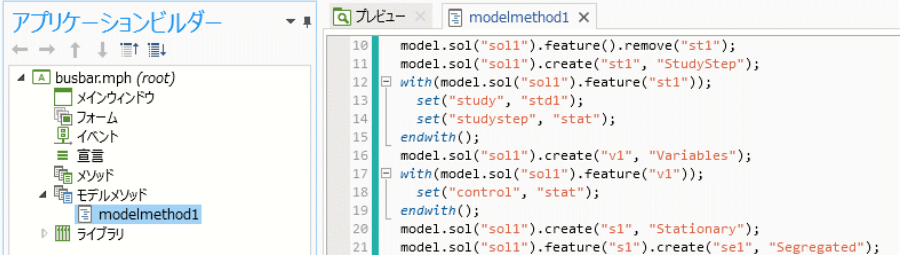


- モデルツリーで、**スタディ 1** ノードを右クリックし、**計算**を選択します（またはリボンオプションの**計算**を使用します）。
- モデルツリーで、**レポート 1** ノードを右クリックし、**出力**を選択します。

- リボンの**デベロッパー**タブをクリックし、**停止**ボタンをクリックします。



- Microsoft® Word ドキュメントを閉じます。
- ここで、リボンの**アプリケーションビルダー**ボタンをクリックしてアプリケーションビルダーに切り換え、アプリケーションツリーとメソッドエディターで記録されたメソッドを確認することができます。

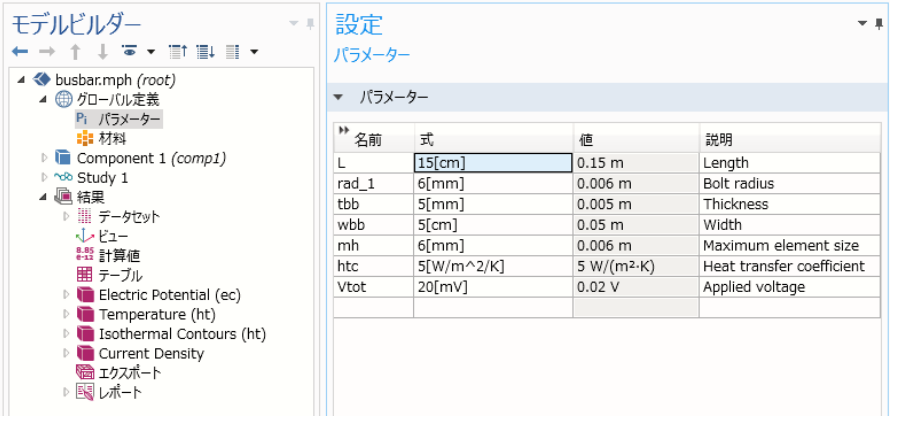


スタディノードから生成されるコードの詳細については、*Application Programming Guide* を参照してください。

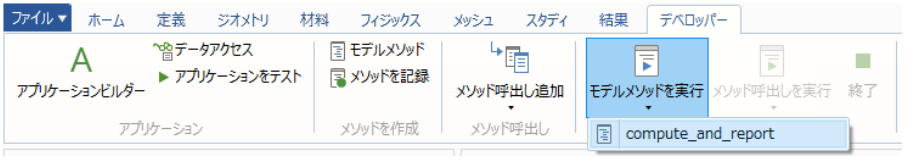
- ここでは、生成されたコードを検査や編集したり、メソッド名を **compute_and_report** などに変更することができます。

- リボンの**モデルビルダー**ボタンをクリックしてモデルビルダーに戻ります。

- **グローバル定義 > パラメータ** で、**Length** を 15 [cm]に変更します。



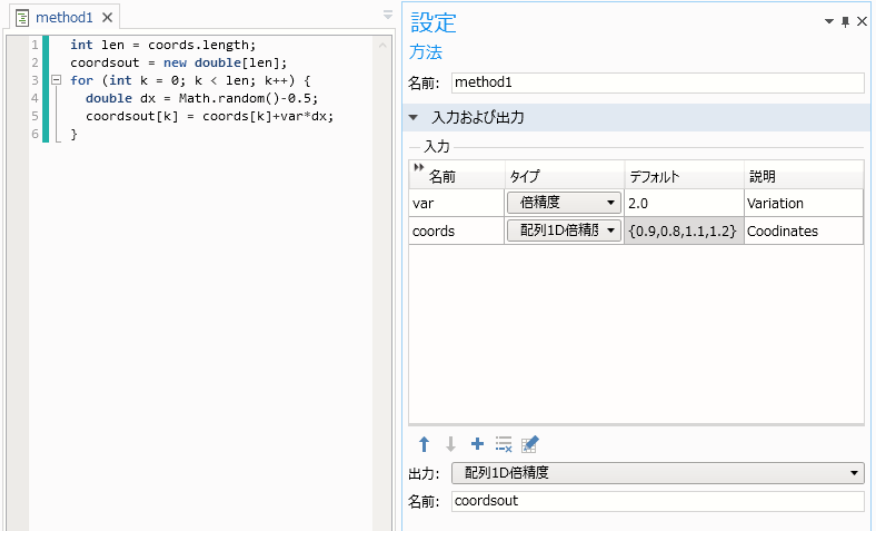
- リボンで、**デベロッパー**タブをクリックし、**モデルメソッドを実行**メニューから **compute_and_report** を選択します。



モデルには複数のモデルメソッドを含めることができます。

入出力の引数を持つメソッド

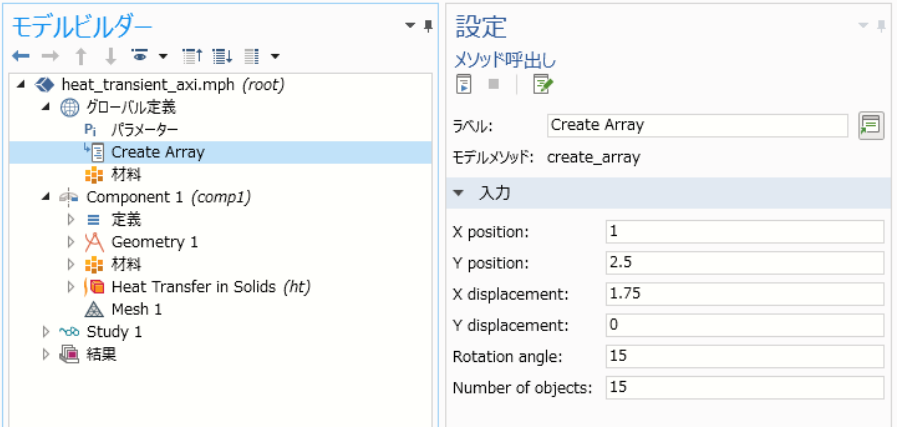
メソッドには、いくつかの入力引数、および一つの出力引数を持つことが許されています。アクティブなメソッドウィンドウの**設定**ウィンドウにある入力と出力の引数を定義します。もし**設定**ウィンドウが表示されていない場合は、リボンの**メソッド**タブの**設定**をクリックします。次の図には、二つの入力引数 var と coords、および一つの出力 coordsout を持つメソッドの場合を示しています。このメソッドでは、配列 coords にランダムな値を加算しています。このランダム性の程度は、入力の変数 var で制御されています。新しいく作られた値は、配列 coordsout に格納されています。



メソッドから別のメソッドを呼び出す記述がある場合、Ctrl+Alt+double-click の操作でその参照先のメソッドのウィンドウを開くことができます。メソッドは、再帰呼び出しの目的で自身を呼び出すことが許されています。

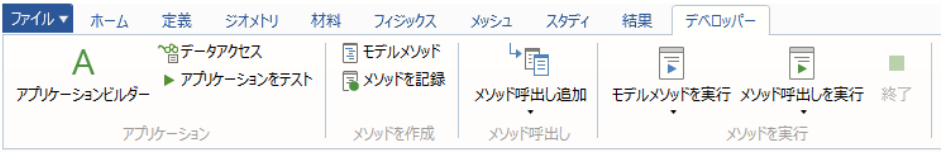
モデルメソッドのためのメソッド呼び出し

特定の入力引数値のセットのモデルメソッドを呼び出すには、**グローバル定義**の下に**メソッド呼出し**ノードを追加することによって行うことができます。**メソッド呼出し**ノードを追加するには、**グローバル定義**を右クリックし、既に作成してあるモデルメソッドのうちの一つを選択します。次の図は、幾何学的配列を作成するメソッドへの**メソッド呼出し**を示しています。



下図に示すように、**設定**ウィンドウの対応するツールバーボタンをクリックすると、**メソッド呼出し**を実行、停止、編集できます。

この機能は、下図に示すように、モデルビルダーのリボンの**デベロッパー**タブからも利用できます。



下図は、アプリケーションビルダーの対応するモデルメソッドの**設定**ウィンドウと、入力引数の定義を示しています。

同じモデルメソッドに対して複数の**メソッド呼出し**ノードを追加することができます。各呼び出しには、異なる入力引数値のセットを含めることができます。

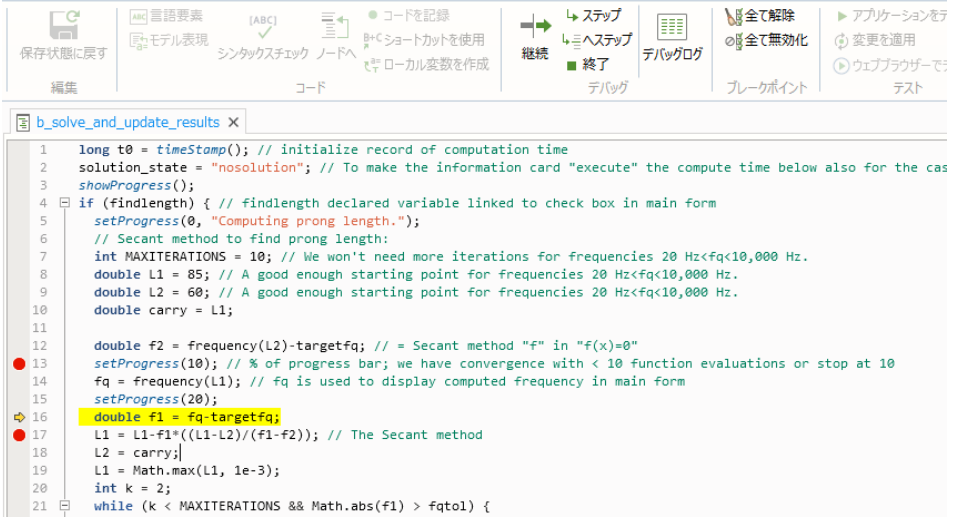
モデルビルダーのモデルメソッドから出力引数を直接使用する方法はありません。但し、組み込みメソッドメッセージの呼び出しを使用して、COMSOL デスクトップ環境の**メッセージ**ウィンドウでモデルメソッドで使用される変数を表示することができます。次の例は、二つの倍精度変数の値を**メッセージ**ウィンドウに表示する方法を示しています。

```
message ( "Width:   " + toString ( width ) ) ;
message ( "Depth:   " + toString ( depth ) ) ;
```

デバッグの目的のために、代わりに 183 ページの「デバッグ」セクションで説明している debugLog メソッドを使用することができます。

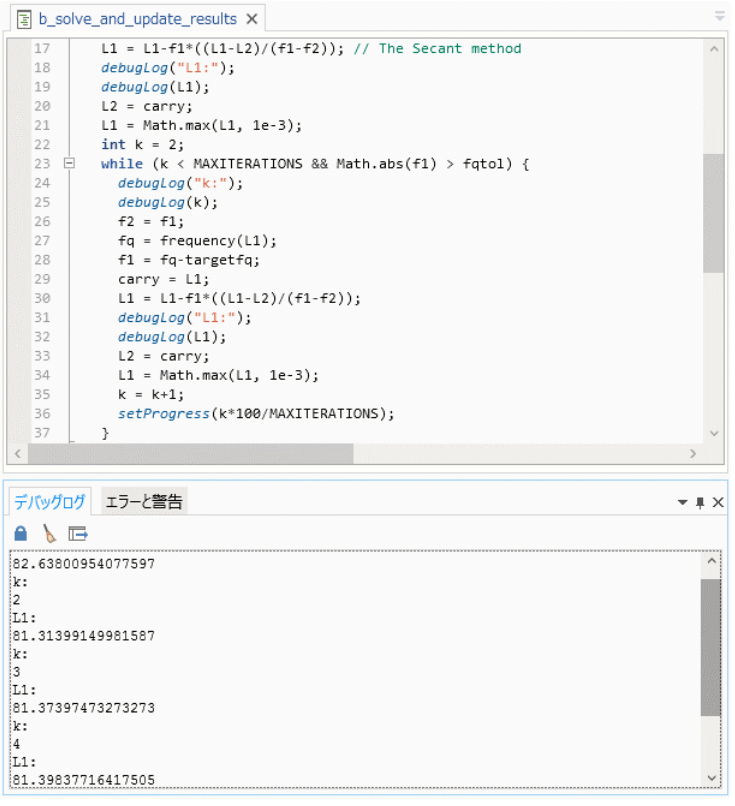
デバッグ

デバッグするために、下図に示すように、コード行番号の左側にある灰色の列部分をクリックして、ブレークポイントを設定することができます。



リボンの**デバッグ**グループには、メソッドをデバッグするために利用するツールが用意されています。アプリケーションを実行した際に、そのメソッドは設定したブレークポイントで停止します。その後、**ステップ**のボタンをクリックすると、メソッドの次の行に進みます。上図は、メソッドが黄色でハイライト表示された行にステップして停止している状態です。

メソッドを次のブレークポイントまで実行させるには、**継続**のボタンをクリックします。メソッドの実行を中止する場合は、**停止**のボタンをクリックします。もし、別のメソッドを呼び出しているような場合、**ヘステップ**のボタンをクリックすることによって、そのメソッド表示にステップしてデバッグすることができます。**全て解除**のボタンを使って、全てのブレークポイントを取り消すことができます。また、**全て無効化**のボタンをクリックして、全てのブレークポイントを無効にすることができます。**デバッグログ**のボタンをクリックすると、次の図に示すように、メソッドとは別の**デバッグログ**ウィンドウが開かれ、そこにデバッグメッセージが表示されます。



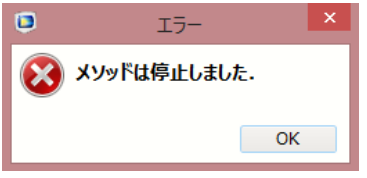
debugLog コマンドを使って、**デバッグログ**ウィンドウに変数の値を表示することができます。以下に示したコードは、倍精度の配列 1D の文字列と配列コンポーネントの値を表示する使用例を示したものです。

```
int len = xcoords . length ;
if ( selected == 0 ) {
   for ( int i = 0 ; i < len ; i ++ ) {
        double divid = double ( i ) / len ;
        xcoords[ i ] = Math . cos ( 2.0 * Math . PI * divid ) ;
        ycoords[ i ] = Math . sin ( 2.0 * Math . PI * divid ) ;
        debugLog ( "x :" ) ;
        debugLog ( xcoords[ i ] ) ;
        debugLog ( "y :" ) ;
        debugLog ( ycoords[ i ] ) ;
        debugLog ( "selected is 0" ) ;
   }
}
```

デバッグのための組み込みメソッドの詳細については、311 ページの「デバッグメソッド」、および *Application Programming Guide* を参照してください。

メソッドの停止

キーボードショートカット Ctrl+Pause の操作によって、アプリケーションのテスト中にメソッドの実行を停止させることができます。その場合、以下のダイアログボックスが表示されます。



モデルオブジェクト

モデルオブジェクトは、操作シーケンスの設定や実行のためのメソッドを含んだ非常に多くのメソッドを提供してくれます。**メソッドに変換**、**コードを記録**、**エディターツール**、および**言語要素**といったメソッドエディターのユーティリティによって、そのようなモデルオブジェクトメソッドを使ったステートメントを作ることができます。モデルオブジェクトとそのメソッドに関するコード例の詳細情報については、*COMSOL Programming Reference Manual* の他に、*Introduction to COMSOL Multiphysics* の“Appendix C—Language Elements and Reserved Names”、*Application Programming Guide* を参照してください。

言語要素の例

COMSOL のメソッド記述には、Java® プログラミング言語が使われており、一般の Java® ステートメントと記述方法を使うことができます。詳細については、*Application Programming Guide* と *Programming Reference Manual* を参照してください。

モデルオブジェクトの単項および二項演算子

下表は、材料特性と境界条件を定義するといったような場合や、結果の中でポスト処理や視覚化に使われる式において、モデルオブジェクトへのアクセスで使われる単項および二項演算子を示しています。

優先順位	記号	説明
1	( ) { } .	グルーピング、リスト、範囲
2	^	べき乗
3	! - +	単項: 論理否定、マイナス、プラス
4	[ ]	単位
5	* /	二項: 乗算、除算
6	+ -	二項: 加算、減算
7	< <= > >=	比較: より小さい、以下、より大きい、以上
8	== !=	比較: 等しい、等しくない
9	&&	論理積
10	||	論理和
11	,	リスト要素の分離

メソッド(JAVA® 表記)における単項および二項演算子

下表は、メソッドの JAVA® コードで使われる最も重要な単項および二項演算子を示しています。

優先順位	記号	説明
1	++ --	単項: 後置加算と後置減算
2	++ -- + - !	単項: 加算、減算、正符号、負符号、論理否定 not
3	* / %	二項: 乗算、除算、剰余
4	+ -	二項: 加算、減算
5	!	論理否定
6	< <= > >=	比較: より小さい、以下、より大きい、以上
7	== !=	比較: 等しい、等しくない(不一致)
8	&&	二項: 論理積
9	||	二項: 論理和
10	?:	条件付き三項

優先順位	記号	説明
11	= += -= *= /= %= >>= <<=	代入
12	,	リスト内の要素の分離

宣言ノードの変数へのアクセス

宣言ノードで定義された変数はグローバル変数であるため、どのメソッドでも利用することができ、メソッド毎に変数を定義する必要はありません。

組み込まれている基本的な数学関数

メソッドで使われる基本的な数学関数は、Java® 数学ライブラリに依存しています。下記はその一例です。

```
Math . sin ( double )
Math . cos ( double )
Math . random ( )
Math . PI
```

IF ステートメント

```
if ( a < b ) {
    alert ( toString( a ) ) ;
} else {
    alert ( toString( b ) ) ;
}
```

FOR ステートメント

```
// Iterate i from 1 to N:
int N=10 ;
for ( int i = 1 ; i <= N ; i ++ ) {
   // Do something
}
```

WHILE ステートメント

```
double t = 0 , h = 0.1 , tend = 10 ;
while ( t < tend ) {
   // do something with t
   t = t + h ;
}
```

WITH ステートメント

```
// Set the global parameter L to a fixed value
with ( model . param ( ) ) ;
    set ( "L" , "10[cm]" ) ;
endwith ( ) ;
```

以下のコードは、上記のコードの内容と同じになります。

```
model . param ( ) . set ( "L" , "10[cm]" ) ;
```

グローバルパラメータへのアクセス

通常、グローバルパラメータの値を設定するためのコードを生成するには、**エディターツール**ウィンドウを使用します。メソッドエディターの中にカーソルを置いた状態で、そのパラメータを右クリックして**設定**を選択します。

グローバルパラメータ L の値を 10cm に設定する記述は、次のようになります。

```
model . param ( ) . set ( "L" , "10[cm]" ) ;
```

グローバルパラメータ L を取得して、それを倍精度変数 Length に格納する記述は、次のようになります。

```
double Length = model . param ( ) . evaluate ( "L" ) ;
```

この場合の評価（evaluation）は、モデルツリーの root ノードで定義された基準**単位系**に対してです。

もし単位がある場合、そのパラメータ L の単位を返すには、次のようになります。

```
String Lunit = model . param ( ) . evaluateUnit ( “L” ) ;
```

グローバルパラメータに倍精度の値を書き込むには、それを文字列に変換する必要があります。それは、グローバルパラメータはモデル表現であり、単位を含んでいるかもしれないからです。

以下のコードでは、変数 Length の値を 2 倍し、その結果に単位 cm を含めてパラメータ L に書き込んでいます。

```
Length = 2 * Length ;
model . param ( ) . set ( "L" , toString ( Length ) + "[cm]" ) ;
```

基本単位系とは異なる単位でパラメータの値を返すには、次のようになります。

```
double Length_real = model . param ( ) . evaluate ( "L" , "cm" ) ;
```

パラメータが複素数化されている場合、実部と虚部は配列数 2 の倍精度ベクトルとして返すことができます。

```
double[ ] realImag = model . param ( ) . evaluateComplex ( "Ex" , "V/m" ) ;
```

文字列の比較

Java® の文字列の値の比較は、= = ではなく、equals( ) を使わなければなりません。これは、= = の演算子はそれらの文字列が同じオブジェクトであるかどうかを比較しており、それらの値を比較しているわけではないからです。次のコードは、文字列の比較の例です。

```
Boolean streq = false ; String a = "string A" ; String b = "string B" ; streq = a . equals( b ) ;
// In this case streq == false

streq = ( a == b ) ;
// In this case streq == false

b = "string A" ;
streq = a . equals ( b ) ;
// In this case streq == true
```

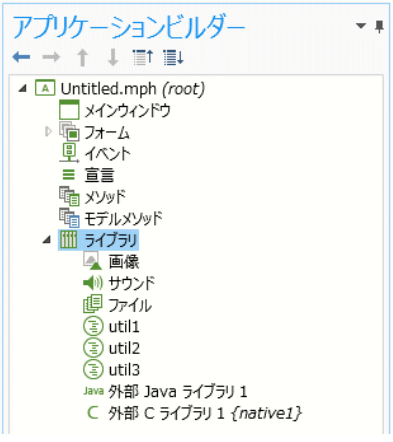
警告とメッセージ

メソッドの alert 、confirm、および request によって、テキストの文字列とユーザの選択入力を備えたダイアログボックスを表示させることができます。以下の例では、confirm を使って、アプリケーションで直接ソルバーを使うか反復ソルバーを使うかをユーザに問い合わせています。そして、そこでユーザが選択したソルバーのタイプに基づいて見積もられるメモリーの要求メッセージを、alert 機能を使ってダイアログボックスに表示しています。

```
String answer = confirm ( "Which solver do you want to use?", "Solver Selection" ,
"Direct" , "Iterative" ) ;
if ( answer . equals ( "Direct" ) ) {
   alert ( "Using the direct solver will require about 4GB of memory when solving." ) ;
} else {
   alert ( "Using the iterative solver will require about 2GB of memory when solving." ) ;
}
```

ライブラリ

アプリケーションツリーにある**ライブラリ**ノードには、画像、サウンド、およびファイルが含まれています。これらは、MPHファイルに組み込まれているので、アプリケーションを配布する際にそれらも別々に配布するといった必要はないわけです。さらに、**ライブラリ**ノードには、Java® ユーティリティクラスのノードと、外部Java® ライブラリと外部Cライブラリのためのノードを含めることができます。ユーティリティークラスと外部ライブラリー利用の詳細については、*Application Builder Reference Manual* を参照してください。



例えば、組み込まれたファイルは、embedded : / / / file1、embedded : / / / file2 などの記述方法によって、フォームオブジェクトやメソッドの中で参照することができます。例えば、画像ファイル compute.png を参照するためには、embedded : / / / compute.png という表記となります。

ファイル名には拡張子を付ける必要はないことに注意してください。その上、任意の名前を付けることができます。MPH ファイルのサイズを最小化するためには、未使用の画像、サウンド、またはその他のファイルを削除します。

アプリケーションの実行中にユーザによってロードされるファイルを管理するためには、**ファイル**の宣言やフォームオブジェクトの**ファイルインポート**の利用などの方法があります。実行中にロードされるファイルについての詳細情報は、137 ページの「ファイル」、243 ページの「ファイルインポート」、および 279 ページの「付録 C—ファイル処理とファイルスキーム表記法」を参照してください。

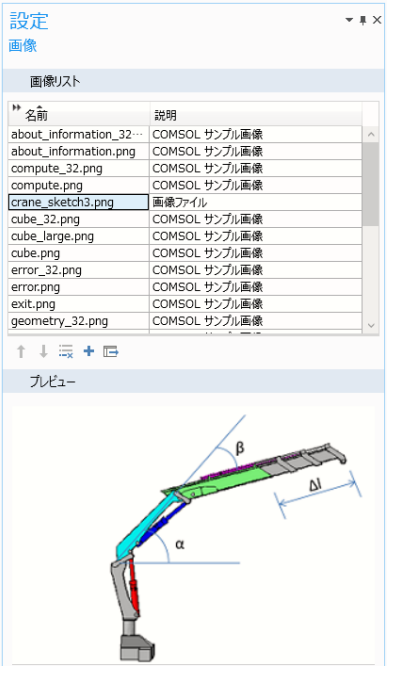
画像

画像ライブラリには、多くのプリロードされた PNG ファイル形式のサンプル画像が含まれています。そこにない他の画像ファイルを組み込みたい場合には、**画像リスト**の下側にある**ファイルをライブラリに追加**のボタンをクリックします。COMSOL インストールフォルダの data/images には豊富な画像ファイルが格納されており、そこから選択して取り込むことができます。画像はアイコンとして使われて、画像フォ

ームオブジェクトまたはメソッドで参照することができます。画像が使われるアイコンのサイズには 2 種類サポートされています：16×16 ピクセル（小）と 32×32 ピクセル（大）があります。

サポートされている画像フォーマットは、JPG、GIF、BMP、および PNG です。

画像をプレビューするには、**画像リスト**の中でその画像の名前をクリックします。下図に示すように、その画像が**プレビュー**セクションに表示されます。



選択した画像をエクスポートするには、画像リストの下にある**選択画像ファイルをエクスポート**のアイコンボタンをクリックします。

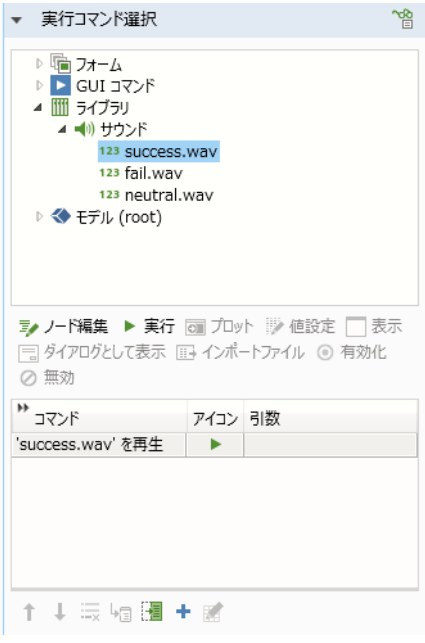
サウンド

サウンドライブラリには、いくつかのプリロードされた WAV ファイル形式のサウンドが含まれています。そこにない他のサウンドファイルを組み込みたい場合には、**サウンドリスト**の下側にある**ファイルをライブラリに追加**のボタンをクリックします。COMSOL インストールフォルダの data/sounds には多くのサウンドファイルが格納されており、そこから選択して取り込むことができます。

サウンドを再生するには、そのサウンドの名前をクリックして**サウンドリスト**の下側にある**プレビュー**ボタンをクリックします。

選択したサウンドをエクスポートするには、プレビューボタンの右隣にある**選択音声ファイルをエクスポート**のボタンをクリックします。

アプリケーションでサウンドを再生するには、ボタン、リボン、メニュー、またはツールバー項目の**設定**ウィンドウのコマンドを追加します。**実行コマンド選択**のセクションのツリーからサウンドを選択し、ツリーの下側にある**実行**ボタンをクリックします。これによって、次の図に示すように、コマンドシーケンスに**再生**コマンドが設定されます。

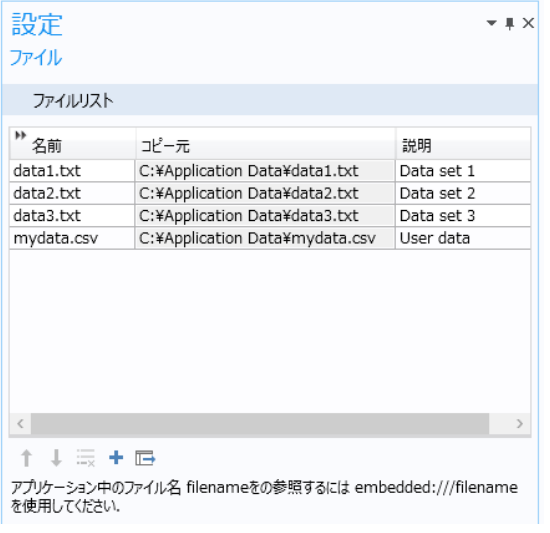


メソッドでは、以下のように組み込みメソッド playSound を使ってサウンドを再生することができます。

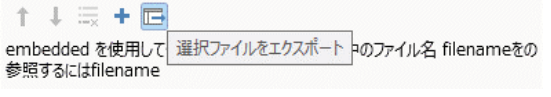
```
playSound ( "success . wav" ) ;
```

ファイル

ファイルライブラリは、デフォルトでは空です。**ファイルをライブラリに追加**のボタンをクリックして、どのようなタイプのファイルでもアプリケーションに組み込むことができます。



選択したファイルをエクスポートするには、**ファイルをライブラリに追加**のボタンの右隣にある**選択ファイルをエクスポート**のボタンをクリックします。



組み込まれたファイルは、embedded:///data1.txt、embedded:///data2.txt などの記述方法によって、メソッドの中で参照することができます。詳細については、137 ページの「ファイル」、279 ページの「付録 C―ファイル処理とファイルスキーム表記付法」、および 303 ページの「ファイルメソッド」を参照してください。

付録 A－フォームオブジェクト

この付録では、フォームとフォームオブジェクトについての情報を提供しています。40 ページの「フォームエディター」の章を拡張しています。以下のリストで * の印が添付されている項目は、その該当する章ですでに詳細説明されていることを意味しています。それ以外の残りの項目は、本付録の中で掲載説明されています。

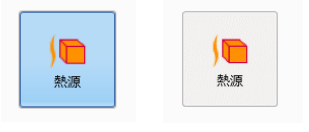
全てのフォームオブジェクトのリスト

- 入力
  - 入力フィールド*
  - ボタン*
  - トグルボタン
  - チェックボックス
  - コンボボックス
- ラベル
  - テキストラベル*
  - 単位*
  - 方程式
  - ライン
- 表示
  - データ表示*
  - グラフィックス*
  - ウェブページ
  - 画像
  - ビデオ
  - 進捗バー
  - ログ
  - メッセージログ
  - 結果テーブル

- サブフォーム
  - フォーム
  - フォームコレクション
  - カードスタック
- コンポジット
  - ファイルインポート
  - 情報カードスタック
  - 配列入力
  - ラジオボタン
  - 選択入力
- その他
  - テキスト
  - リストボックス
  - テーブル
  - スライダー
  - ハイパーリンク
  - ツールバー
  - スペーサー

トグルボタン

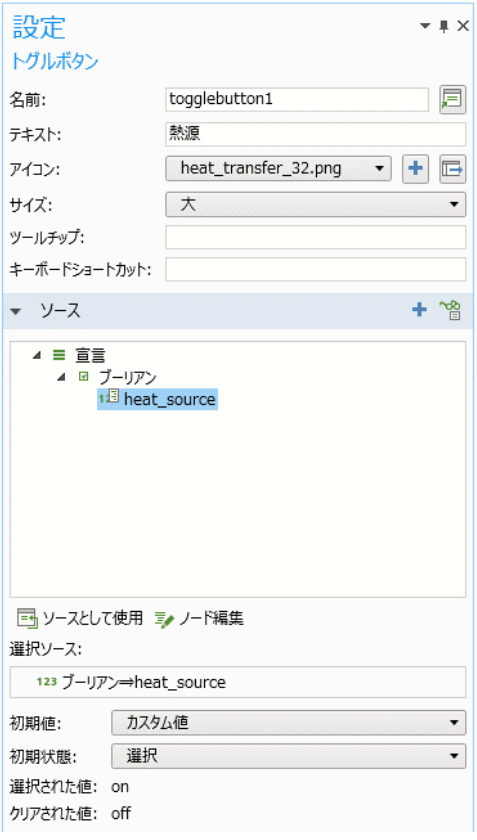
トグルボタンオブジェクトは、下図に示すように、選択と選択解除の二つの状態を持つボタンです。



このセクションの情報は、**メニュー**の**項目切替え**と**リボン**の**項目切替え**にも適用可能です。

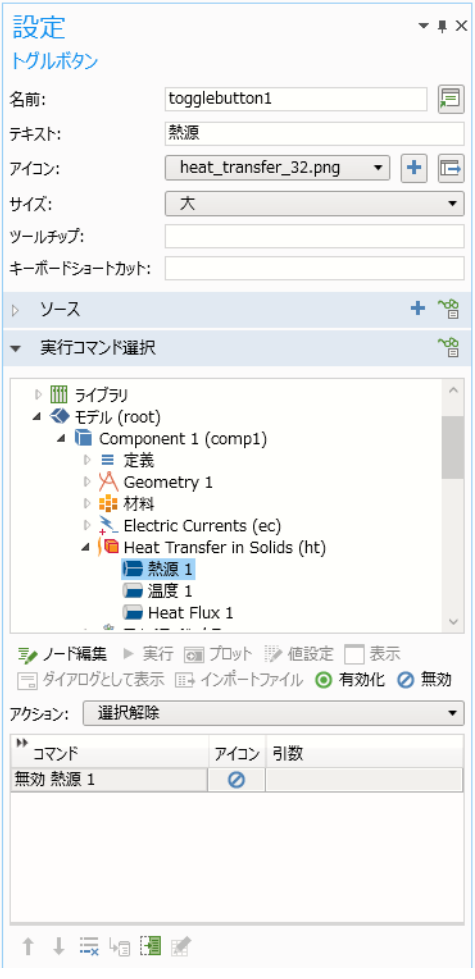
熱源の有効/無効のためのトグルボタンの利用

トグルボタンの二つの状態の設定は、ブーリアン変数に関連付けられています。次の図に示したボタンの**設定**ウィンドウでは、そのボタンの状態によって熱源が有効または無効となるように設定されています。その**ソース**セクションでは、ブーリアン変数 heat_source が選択されています。

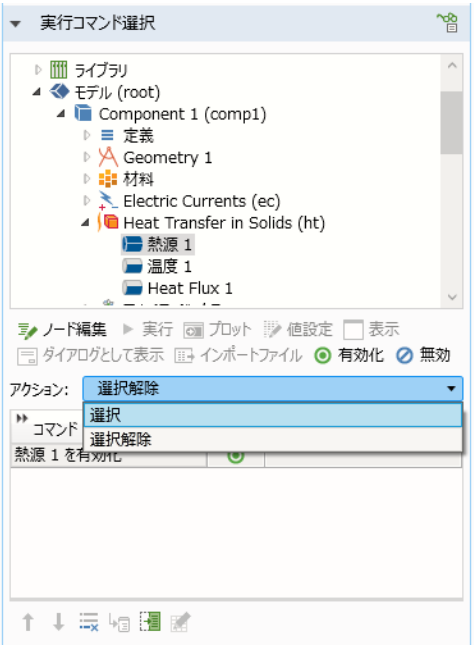


トグルボタンが選択状態に反転したときには、ブーリアン変数 heat_source は true に設定され、熱源は有効となります。トグルボタンが選択解除の状態に反転すると、ブーリアン変数 heat_source は false に設定されて、熱源は無効となります。

ソースセクションの下には、**実行コマンド選択**セクションがあります。そこにある**アクション**には、**選択**と**選択解除**の二つの異なったコマンドを表す選択項目があります。下図では、**無効 熱源**のコマンドに対するアクションを**選択解除**としている**設定**ウィンドウを示しています。



下図では、**熱源を有効化**のコマンドが設定されたコマンドシーケンスに対するアクションを**選択**としています。



ブーリアン変数と関連付けられている点では、トグルボタンはチェックボックスと同じです。チェックボックスはイベントを使ってアクションを定義するのに対して、トグルボタンはコマンドシーケンスを使ってアクションを定義します。これについては、次の章で説明されています。

チェックボックス

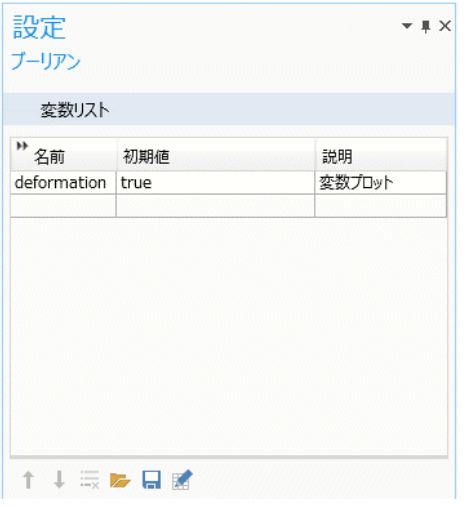
チェックボックスは、選択に対応する on とクリアに対応する off の二つの設定値を持ちます。チェックボックスの状態は、**宣言**ノードに定義したブーリアン変数に格納されます。

視覚化制御へのチェックボックスの利用

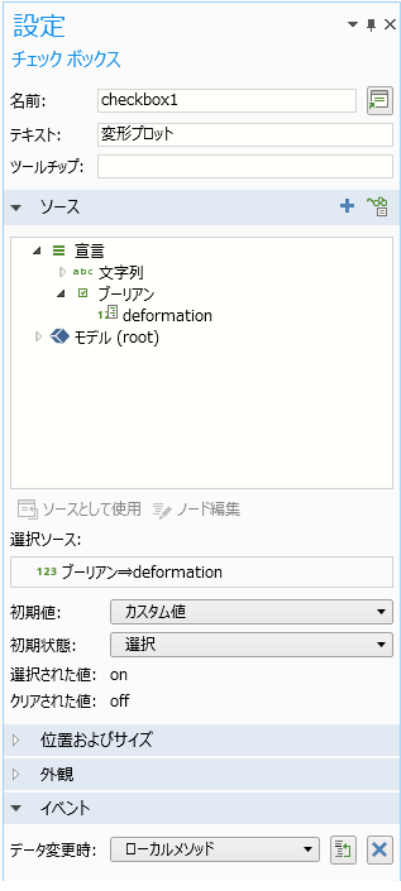
下図は、チェックボックスのチェックの有無によって、変形プロットを有効か無効にするアプリケーションの例です。

この左側の図は、アプリケーションの実行中におけるスクリーンショットです。右側のスクリーンショットは、グリッドレイアウトモードのフォームエディターにおける該当箇所を示しています。

下図に示した**設定**ウィンドウに定義されたブーリアン変数 deformation は、チェックボックスの状態を格納するための例です。



下図は、チェックボックスの**設定**ウィンドウです。



宣言したブーリアン変数を**ソース**セクションのツリーから選択し、**ソースとして使用**をクリックすることにより、チェックボックスに関連付けることができます。

チェックボックスのテキストラベルには、関連付けたブーリアン変数の設定ウィンドウの**説明**欄に記載した内容がデフォルトで設定されます。

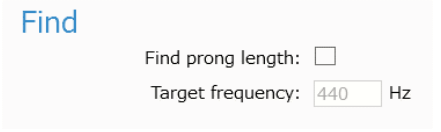
変数 deformation の**初期値**は、**選択された値**(on)、または**クリアされた値**(off)が上書きされ、編集する必要はありません。メソッドで使われる場合、on と off の値はそれぞれ true と false の別名です。例えば、ステートメントにおいて、これらの値はブーリアンとして扱われます。

以下は、ブーリアン変数 deformation の値が変更された時に、**データ変更時**のイベント設定によって実行されているローカルメソッドのコードステートメントです。

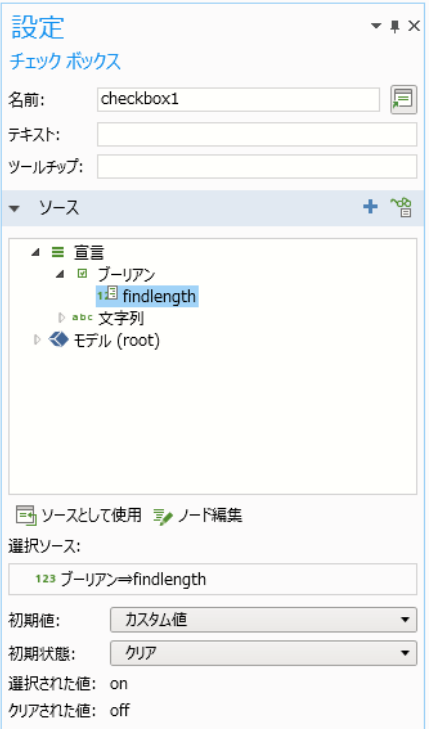
```
model . result ( "pg1" ) . feature ( "surf1" ) . feature ( "def" ) . active ( deformation ) ;
useGraphics ( model . result ( "pg1" ) , "graphics1" ) ;
```

フォームオブジェクトの有効/無効のためのチェックボックスの利用

下図には、チェックボックスのチェックの有無によって、入力フィールドの有効か無効かを制御するアプリケーションの部分を示しています。



下図は、そのチェックボックスの**設定**ウィンドウを示しており、チェックボックスの状態を格納するためのブーリアン変数 findlength が関連付けられています。



以下は、ブーリアン変数 findlength の値が変更された時に、**データ変更時**のイベント設定によって実行されているローカルメソッドのコードステートメントです。

```
setFormObjectEditable ( "main/inputfield1" , ! findlength ) ;
setFormObjectEditable ( "main/inputfield5" , findlength ) ;
setFormObjectEnabled ( "main/inputfield5" , findlength ) ;
setFormObjectEditable ( "main/inputfield6" , findlength ) ;
setFormObjectEnabled ( "main/inputfield6" , findlength ) ;
solution_state = "inputchanged" ;
```

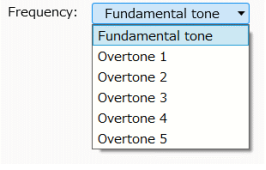
コンボボックス

コンボボックスは、編集可能なテキストフィールドの機能を併せ持ったドロップダウンリストボックス、または編集機能のないドロップダウンリストボックスとして利用することができます。

結果のパラメータ変更へのコンボボックスの利用

コンボボックスの利用方法を解説するために、構造の振動分析において、六つの異なるモード形状のうちの一つを選んで視覚化されるアプリケーションを考えてみましょう。この例では、固体力学のフィジックスインタフェースの固有周波数のスタディを使っており、類似の分析にも適用することができます。

ユーザがコンボボックスから選択する六つの異なった固有周波数をこれらの六つのモード形状に対応させます。



この例において、コンボボックスは、文字列変数 mode の値を制御するために使われています。下図は、この変数の**設定**ウィンドウを示しています。

ソースの選択

下図は、このコンボボックスの**設定**ウィンドウです。

ソースセクションでは、コンボボックスによって制御する値を持たせる役目のスカラー変数を選択し、**ソースとして使用**をクリックします。コンボボックスの**設定**ウィンドウの**初期値**のリストでは、コンボボックスのデフォルト値を定義する方法を選択します。その選択肢には、**最初の許される値**（デフォルト）と**カスタム値**があります。**カスタム値**を選択した場合は、関連して表示されるリストにデフォルトの値を設定します。そのデフォルト値は、リストにある許される値の中から選択します。

選択リスト

振動のモード 1－6 は通常の剛体モードに相当し、このアプリケーションには重要ではありません。そのため、最初に重要となるモードは 7 です。選択リストの**表示名**の欄に入力した文字列だけが実際に表示されるので、ユーザにはモデルのモード値が表示されません。非剛体モードの初めは、Fundamental tone、Overtone 1、Overtone 2、などと名付けられています。

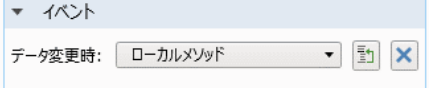
選択リストのセクションでは、許される値をコンボボックスに与える役目の選択リストを追加します。下図には、この例に関する**選択リスト**の宣言を示しています。

文字列変数 mode は、7、8、9、10、11、12 の六つの値のうちの一つを持つことになります。コンボボックスに表示されるのは、**表示名**の欄のテキスト文字列です。

コンボボックスの**設定**ウィンドウで、**その他の値を許可**のチェックボックスにチェックを入れることによって、コンボボックスで値を自由に入力することができるようになります。そのように設定したコンボボックスでは、選択リストによって定義された値に限らず、どのような値にでも適応します。しかし、この例では、ほんの六つのあらかじめ定義された値だけが許されています。選択リストの詳細について、135 ページの「選択リスト」を参照してください。

イベント

イベントセクションでは、コンボボックスの値（すなわちソースとして使われている文字列変数の値）がユーザによって変更された時に実行されるメソッドを指定します。今の場合は、変数 mode の値が変更され、下図に示すようにローカルメソッドが実行されます。



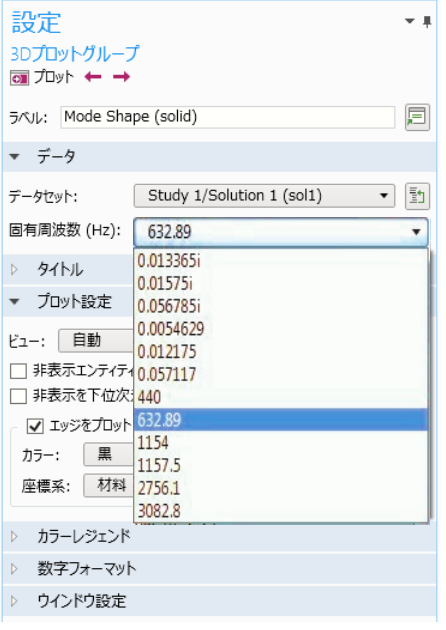
以下に、ローカルメソッドのコードを掲載します。

```
with  ( model . result ( "pg1" ) ) ;
      set ( "looplevel" , new String[ ] { mode } ) ;
endwith ( ) ;
model . result ( "pg1" ) . run ( ) ;
```

このコードは、文字列 mode の値を、プロットグループ pg1 で設定している固有周波数に結び付けています。このケースでは、文字列 svar は値 “7”、”8”、”9”、”10”、”11”、”12” をとっています。

上記のコードは、以下に示すように、メソッドエディターのコードの記録機能を使って自動的に生成することができます。

- モデルビルダーに行き、**メソッドを記録**をクリックします。
- 構造力学分析のための固有周波数のスタディを使う時には、デフォルトで**モード形状**（Mode Shape）のプロットグループが作成されます。このプロットグループで、**固有周波数**をモード 7 からモード 8 に変更します。下図に示した**モード形状**（Mode Shape）のプロットグループの**設定**ウィンドウでは、これが 440Hz から 632.89Hz への変更に相当しています。



- **レコード停止**をクリックします。

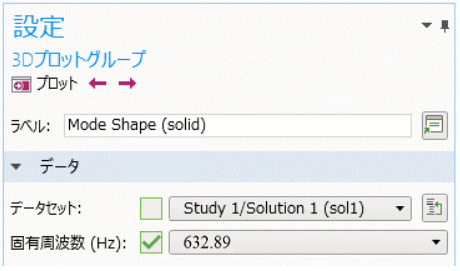
結果として生成されるコードを、以下に示します。

```
with ( model . result ( "pg1" ) ) ;
    set ( "looplevel" , newString[ ] { "8" } ) ;
endwith ( ) ;
model . result ( "pg1" ) . run ( ) ;
```

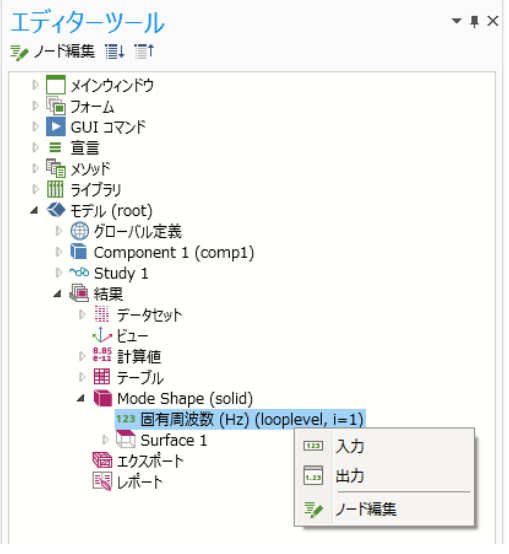
ここで、上に掲載しているコードを完成させるため、コード中の文字列 ”8” を変数 mode に書き換えます。例えば、これが method1 というメソッドに格納されている場合、コンボボックスに関係付けたローカルメソッドを生成し、そこに method1 からコードをコピーします。その後、method1 を削除します。

データアクセスの利用

コンボボックスを使うための迅速、かつ一般的な方法は、**データアクセス**を**エディターツール**と組み合わせて使うことです。下図に示すように、この章で使われる例では、**データアクセス**を有効にするところから始め、**モード形状**（Mode Shape）のプロットグループの**設定**ウィンドウで**固有周波数**を選択します。

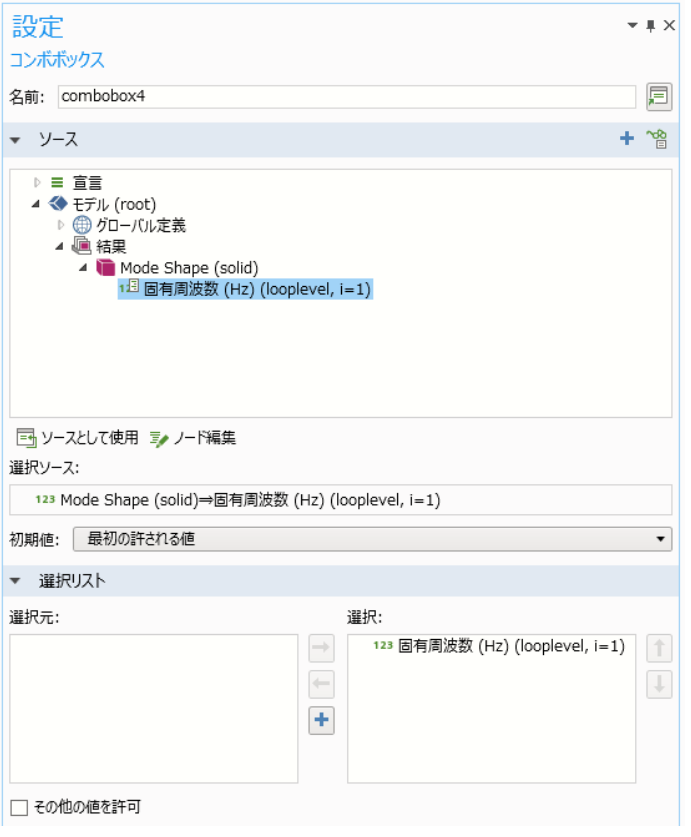


エディターツールウィンドウでは、**固有周波数**のパラメータは**固有周波数**（looplevel）として表示されます。コンボボックスを作成するためには、この**固有周波数**（looplevel）を右クリックし、**入力**を選択します。



一般的に、**固有周波数**（looplevel）という名称は、求解のパラメータに使われます。もし一つの求解が二つ以上のパラメータを持っているならば、選択可能な二つ以上の**固有周波数**（looplevel）があります。

下図は、この例に対応したコンボボックスの**設定**ウィンドウです。

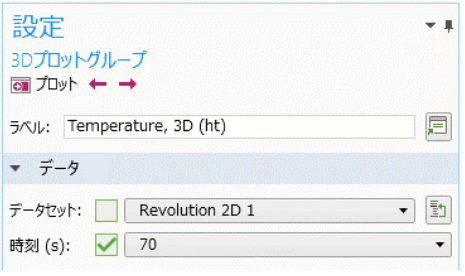


選択リストの**固有周波数**(looplevel)は、**エディターツール**を使ってコンボボックスを挿入する時に自動的に生成されます。この方法で生成された選択リストは、**宣言**ノードの下には表示されず、ユーザにより修正できないことに注意してください。個々のパラメータや固有周波数の値に名前を付けるといったような高い柔軟性を求めるならば、前の章で説明されているように、選択リストを自分で宣言定義する必要があります。

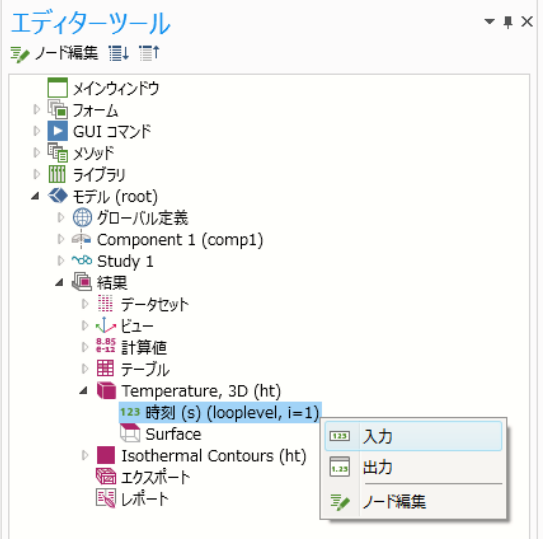
時間変更へのコンボボックスの利用

時間依存のスタディ(Study)ステップで指定された時刻のパラメータリストは、**結果**(**Results**)ノードの下の多くの場所で使用することができます。アプリケーションの個々の時刻パラメータは、**データアクセス**を使用して**エディターツール**と組み合わせて、パラメータの最後のセクションで説明したものと同様の方法でアクセスすることができます。

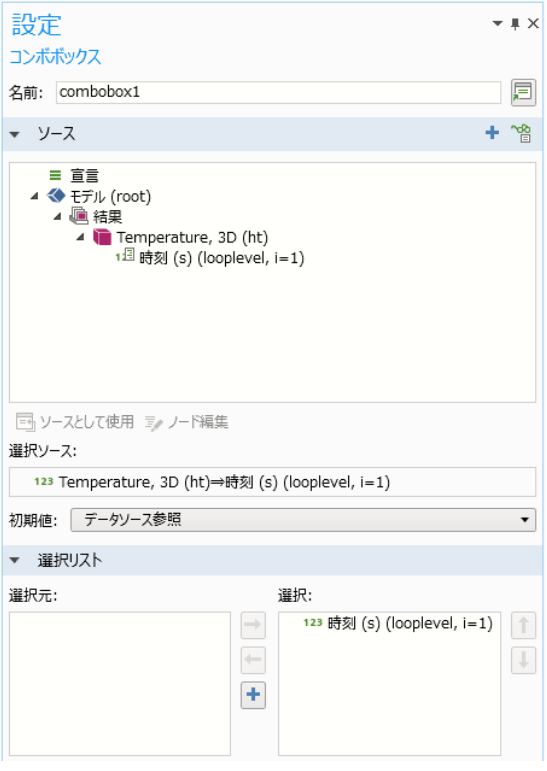
下図の**設定**ウィンドウでは、**データアクセス**が温度プロットでの**時刻**のパラメータリストにアクセスするために利用されています。



下図では、**時刻**リストへのハンドルが、**エディターツール**でまさに利用可能な状態を示しています。



そこで**入力**を選択することにより、下図に示すように、**ソース**としてそれを使用してコンボボックスを作成することができます。

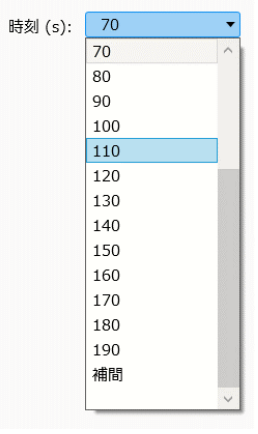


例えば、異なる時刻パラメータに対応するプロットを更新するといったように、コンボボックスでは複数の目的に使用することができます。ユーザが新しい時刻パラメータを選択するためにコンボボックスを使用した場合、それが自動的に更新するプロット動作となるようにするには、そのコンボボックスの設定ウィンドウの最下部でイベントを追加します。下図では、温度プロットを更新するために plot_T というメソッドが呼ばれています。

以下のコード行は、plot_T メソッドの内容を示しています。

```
model . result ( "pg1" ) . run ( ) ;
```

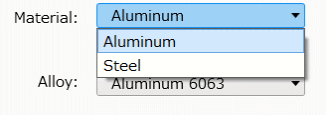
下図に示したアプリケーションユーザインタフェースのコンボボックスが最終結果で、ユーザが**時刻**リストの新しい値を選択した時に自動的に温度プロットを更新します。

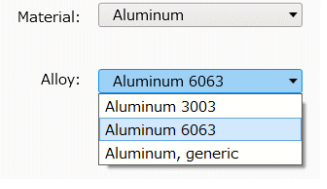


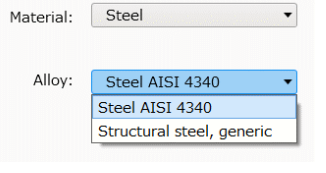
材料変更へのコンボボックスの利用

材料を選ぶためにコンボボックスを利用するアプリケーションを考えてみましょう。この場合には、ユーザインタフェースデザインの高い柔軟性のために、アクティベーション条件（ 136 ページの「アクティベーション条件」を参照 ）を使います。

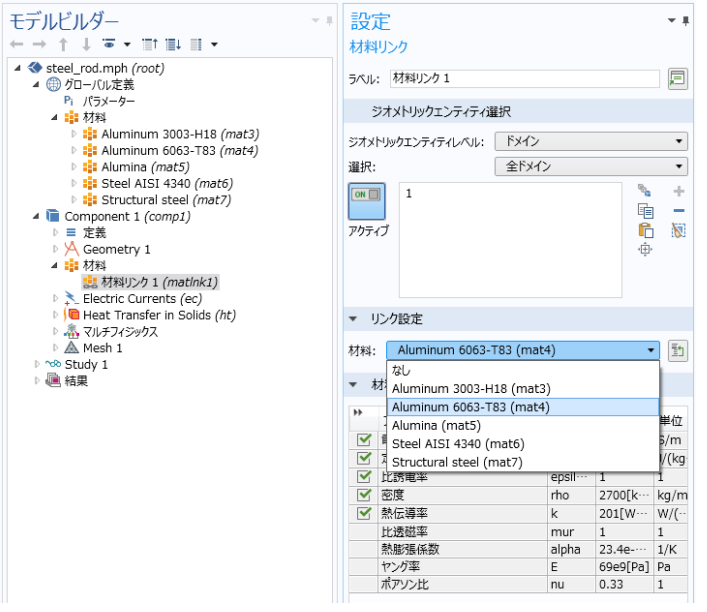
次の図は、ユーザが **Material** と名付けられたコンボボックスを使って、**Aluminum** と **Steel** の二つの材料から選択することができるアプリケーションのスクリーンショットです。**Alloy** と名付けられた 2 番目のコンボボックスは、1 番目の **Materia**l のリストの選択結果によって、**Aluminum** 合金、または **Steel** 合金のリストを表示します。







材料の選択は、下図に示されるように、埋め込みモデルでグローバル材料と材料リンクを使って実行されます。



個々の材料には、mat1、mat2、...、mat5 という文字列のインデックスが付けられています。コンボボックスによって制御されているグローバル変数 alloy の値が変更されることを、イベントが監視しています。この値が変更された時には、以下に掲載したメソッドが実行されます。

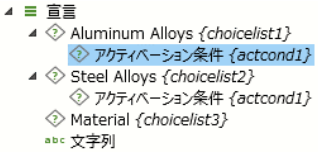
```
with  ( model . material ( "matlnk1" ) ) ;
      set ( "link" , alloy ) ;
endwith ( ) ;
```

下図には、**Material** と **Alloy** のコンボボックスによってそれぞれ制御される二つの文字列変数 material と alloy の宣言を示しています。

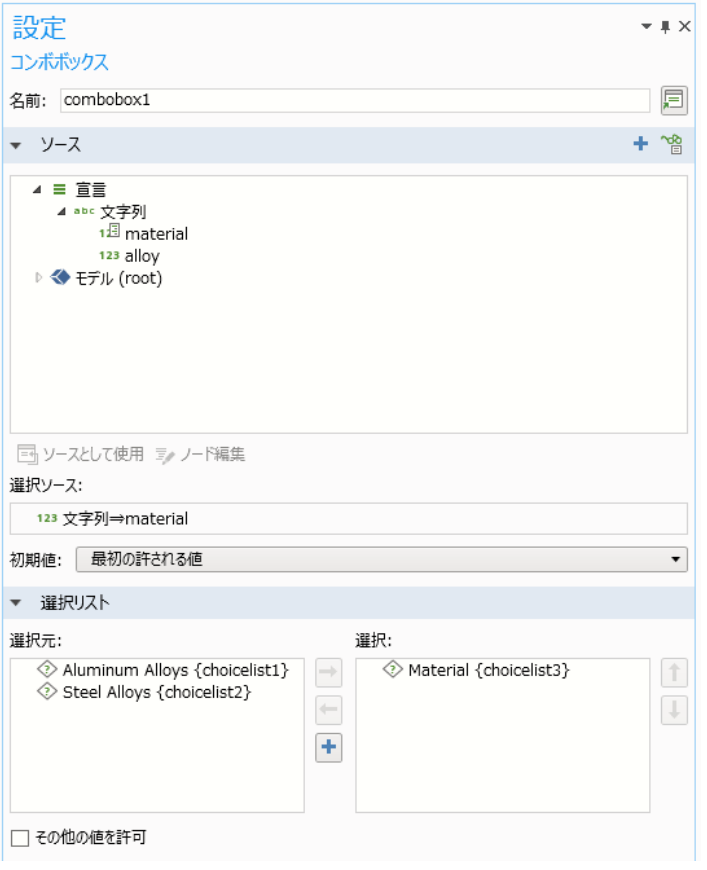
このアプリケーションでは、**Aluminum Alloys**、**Steel Alloys**、および **Material** の三つの選択リストを宣言して利用しています。

アクティベーション条件

アクティベーション条件は、下図に示すように、**Aluminum Alloys** と **Steel Alloys** の選択リストのために使われています。



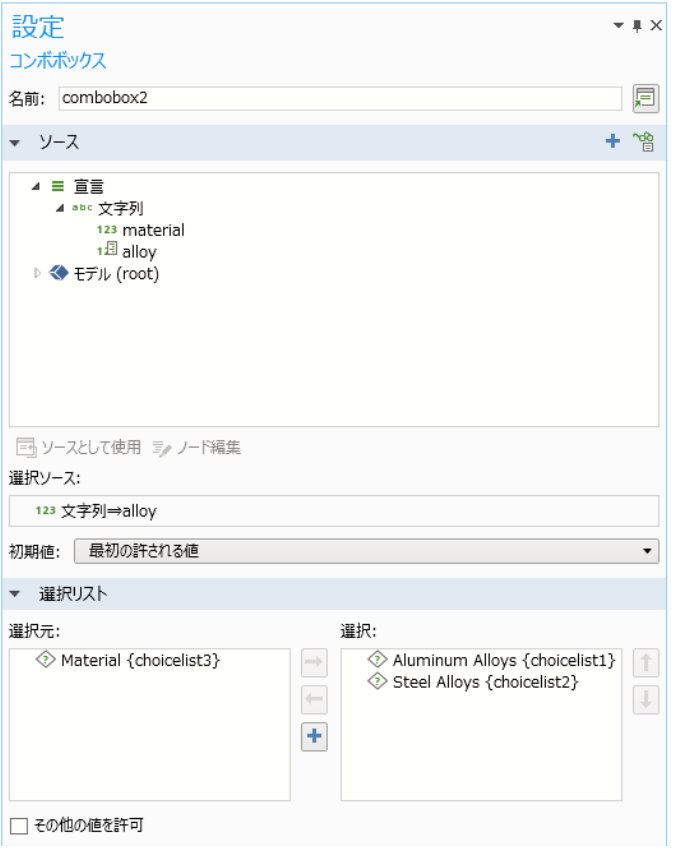
Material のコンボボックスの**設定**ウィンドウを、下図に示しています。



Material のコンボボックスは、そのソースとして文字列変数 material を使っていることに注意してください。**Material** の選択リストは、その文字列変数 material に許される値の個々の設定を定義するために使われています。

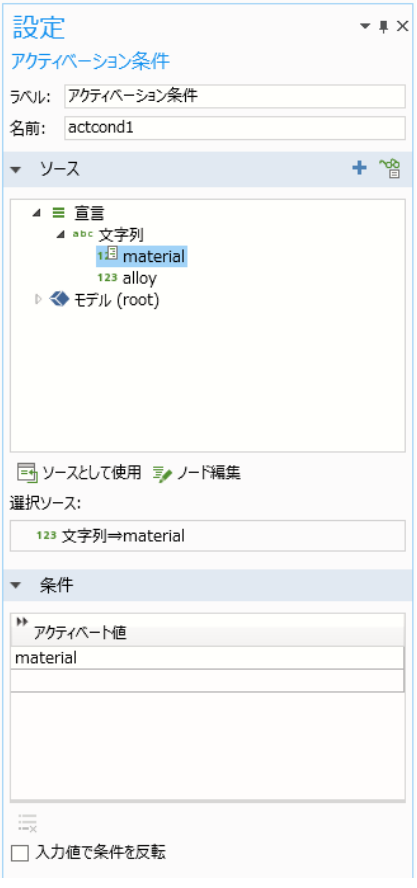
Material の選択リストの**設定**ウィンドウを、下図に示しています。

Alloy のコンボボックスの**設定**ウィンドウを、下図に示しています。



Alloy のコンボボックスが **Aluminum Alloys** と **Steel Alloys** の選択リストの両方を使っていることに注意してください。**Aluminum Alloys** のための選択リストを、下図に示しています。

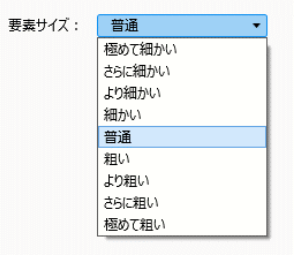
Aluminum Alloys の選択リストのためのアクティベーション条件を、下図に示しています。



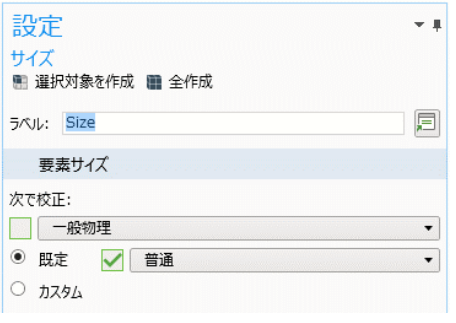
要素サイズ変更へのコンボボックスの利用

コンボボックスを作成する時に、**データアクセス**機能を利用することによって、モデルビルダー側に存在しているコンボボックスの機能を再現することができます。

例えば、下図のように、メッシュの要素サイズを変更するためにコンボボックスが利用されるアプリケーションを考えてみましょう。



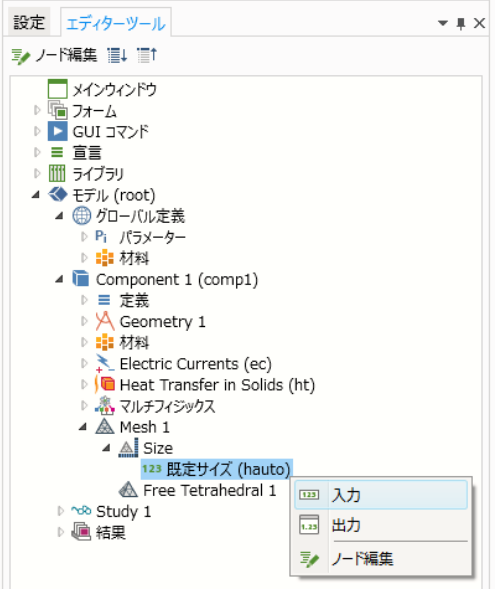
モデルビルダーに切り換えて、**メッシュ**（Mesh）ノードを選択します（ここで、モデルが単にただ一つのメッシュを持っていると仮定します）。**メッシュ**（Mesh）ノードの**設定**ウィンドウにおいて、**ユーザ制御メッシュ**が選択された状態にします。**メッシュ**（Mesh）ノードのすぐ下の**サイズ**（Size）ノードで、**既定**にチェックを入れます。リボンで**データアクセス**をクリックします。これによって、下図に示すように、コンボボックスで既定の要素サイズを利用できるようになります。



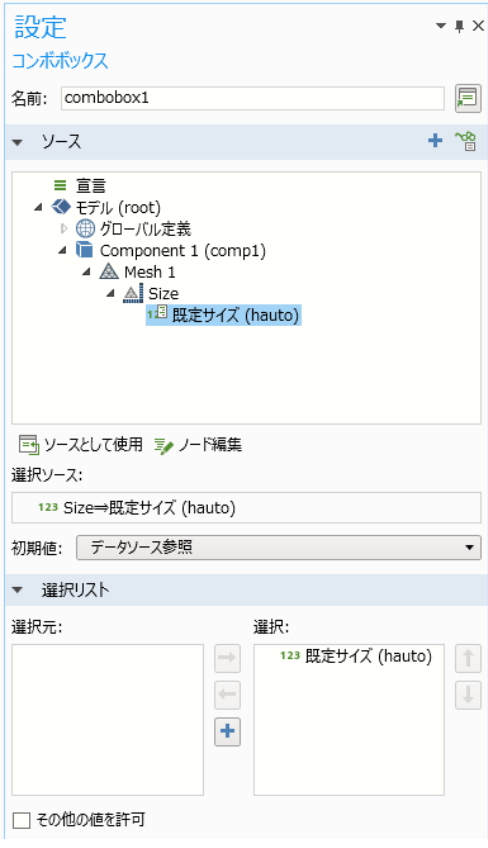
それをアプリケーションビルダーのコンボボックスのソースとして使用可能にするためには、**既定**のリストの左に表示された緑色のチェックボックスにチェックを入れます。その後、アプリケーションビルダーに戻れば、新しいコンボボックスの**設定**において、そのメッシュサイズの選択リストが選択可能な**ソース**になっていることが確認できます。

コンボボックスオブジェクトを挿入するには、以下の二つの方法があります。

- リボンの**オブジェクトを挿入**のメニューから、**コンボボックス**を選択します。コンボボックスの**設定**ウィンドウの**ソース**セクションで**既定サイズ（hauto）**ノードを選択してから、**ソースとして使用**のボタンをクリックします。
- **エディターツール**ウィンドウにおいて、**メッシュ**（Mesh）> **サイズ**（Size）ノードの下の**既定サイズ（hauto）**ノードを選択します。そして、下図に示すように、それを右クリックして**入力**を選択します。



対応するコンボボックスの**設定**ウィンドウを、下図に示しています。



初期値を**データソース参照**に変更することで、そのモデルで設定している要素サイズ（ このケースでは**普通**（Normal））がアプリケーションのデフォルトの要素サイズに使用されるようになります。現在、モデルビルダーから引き継がれた選択リストである**既定サイズ（hauto）**は、アプリケーションビルダーのコンボボックスのための選択リストに設定されています。この選択リストはモデルビルダーから参照されているため、アプリケーションツリーの**宣言**ノードの下に定義された選択リストとしては表示されていません。このため、選択リストにもっと制限を加えたいような場合でも、そのような編集をすることができません。これを改善するためには、代わりの手段として、コンボボックスの**ソース**として使用されている既定のリストを削除し、**宣言**ノードの下に自分自身の新しい選択リストを宣言して作成し直す必要があります。

下図は、三つの入力値を持った選択リストを作成した場合の設定例を示しています。

モデルの**要素サイズ**リストで使われている値を知るためには、**メソッドを記録**を使い、値を**普通**（Normal）から**細かい**（Fine）に変更し、それから**粗い**（Coarse）に変更し、その後に**普通**（Normal）に戻します。**記録停止**をクリックし、自動生成されたコードで値を調べます。以下に示した自動生成されたコードから、**要素サイズ**のプロパティ名は hauto で、**細かい**（Fine）、**普通**（Normal）、および**粗い**（Coarse）のための値はそれぞれ 4、5、および 6 であることがわかります。

```
with  ( model . mesh ( "mesh1" ) . feature ( "size" ) ) ;
    set ( "hauto" , "4" ) ;
    set ( "hauto" , "6" ) ;
    set ( "hauto" , "5" ) ;
endwith ( ) ;
```

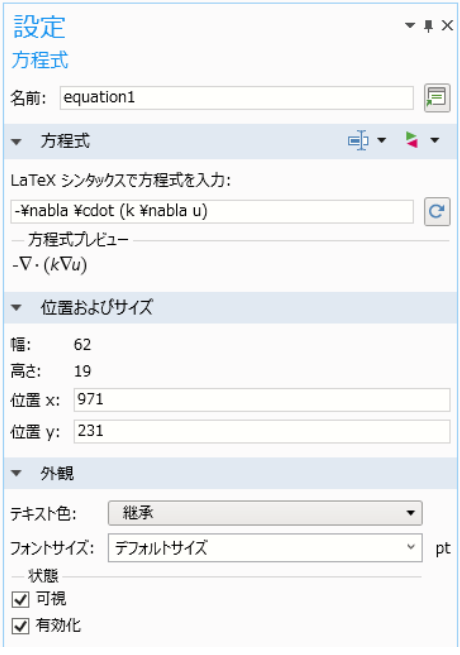
hauto のプロパティは非整数値をとることもできます。**要素サイズ**の詳細については、91 ページの「データアクセスのボタンへの利用」を参照してください。

選択リストの代わりに単位セットの利用

コンボボックスが単位を変更する目的で使用される場合には、**選択リスト**の代わりに**単位セット**を使用することができます（その場合も、コンボボックスの**設定**ウィンドウの**選択リスト**のセクションで単位セットを選択する点は同じです）。

方程式

方程式オブジェクトでは、**LaTeX シンタックスで方程式を入力**のフィールドに式を入力することによって、LaTeX 方程式を表示させることができます。（下図は、スケッチレイアウトモード時の場合です。）



テキストフィールドから離れると、その入力設定によって描画される LaTeX シンタックスがプレビューに表示されます。

ライン

ラインフォームオブジェクトは、例えば、フォームオブジェクトのグループの区分けを示す目的で、フォーム上に水平線または垂直線を追加するために使われます。水平線を選択した場合は、そのライン内にテキストの表示を追加することができます。（下図は、スケッチレイアウトモード時の場合です。）

ウェブページ

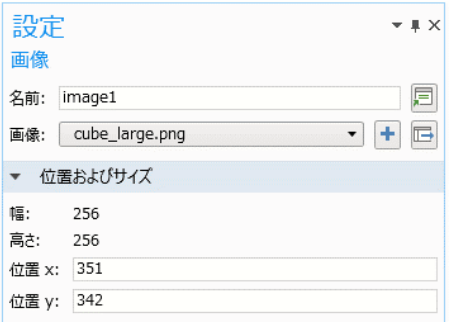
ウェブページオブジェクトでは、ユーザインタフェースの一部としてウェブページのコンテンツを表示することができます。（下図は、スケッチレイアウトモード時の場合です。）

ページのソースを指定するには、**ソース**セクションのリストにある四つの方法があります。

- そのリストの下側にあるテキストエリアに HTML コードを入力するためには、デフォルトの**ページ**を選択します。そのコードは、開始タグ<html>と終了タグ</html>で囲まれた範囲に入力します。
- インターネットのウェブページにリンクする **URL** を選択します。
- HTML コードを含むローカルなファイルリソースを指定するためには、**ファイル**を選択します。**ファイル**フィールドにファイル名を入力するか、または**参照**をクリックしてローカルファイルシステム上のファイルの位置を指定します。
- HTML レポートを埋め込むためには、**レポート**を選択します。これを選択した場合、ブラウザープレビューは利用できません。

画像

画像フォームオブジェクトを使って、画像をフォームに追加することができます。画像オブジェクトは、それがインタラクティブではないという点でグラフィックスオブジェクトと異なります。画像ファイルは、ドロップダウンリストにある利用可能な画像ライブラリの中から選択するか、または**ライブラリに対する画像を追加してここで使用**のボタンをクリックしてローカルファイルシステムからファイルを選択します。下図は、**ライブラリ**ノードに定義された画像オブジェクトの**設定**ウィンドウにおいて、画像 cube_large.png を参照している場合を示しています。（下図はスケッチレイアウトモード時の場合です。）



自分のファイルシステムにある画像ファイルを選択した場合には、そのファイルはアプリケーションに組み込まれ、**ライブラリ**ノードの下の**画像**リストに追加されます。

画像の位置 x および位置 y を変更することはできますが、画像ファイルの幅と高さのサイズは固定です。

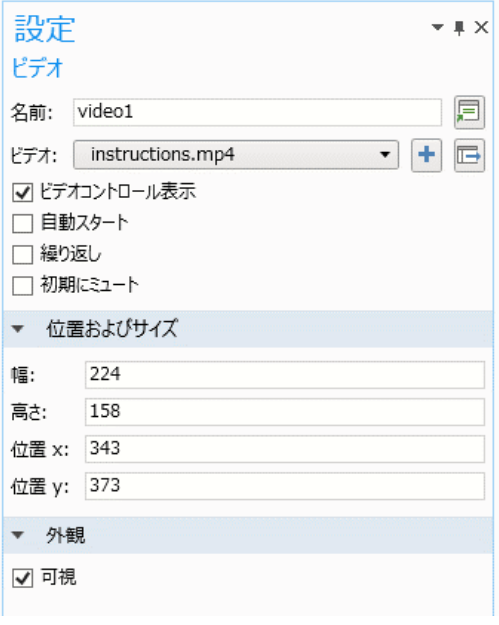
Ctrl+V の操作によって、クリップボードからフォームウィンドウに画像を貼り付けることができます。例えば 、スライド形式のプレゼンテーションソフト PowerPoint® から画像をコピーペーストすることができます。その画像は、自動的に**画像**ライブラリに追加され、アプリケーションに組み込まれます。貼り付けられた画像の名前は、pasted_image_1.png、pasted_image_2.png などのように、自動的に設定されます。

ビデオ

ビデオオブジェクトは、フォームにビデオファイルを組み込みます。サポートされているビデオファイル形式は MP4（.mp4）、OGV（.ogv）、および WebM の（.webm）です。

フォームに追加すると、**ビデオ**オブジェクトは、次の図に示したような画像によってフォームエディターに表示されます。

下図は、**ビデオ**オブジェクトの**設定**ウィンドウを示しています。



使用可能な設定は以下の通りです。

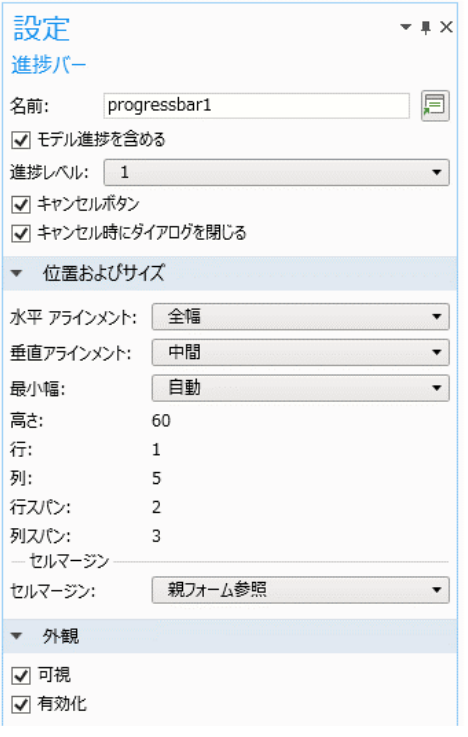
- **ビデオコントロール表示**
- **自動スタート**
- **繰り返し**
- **初期にミュート**

ビデオコントロール表示の選択では、再生や停止などのビデオ制御を可能にします。**初期にミュート**の選択では、最初に音をオフとして映像を再生したい場合を対象としています。例えば、ビデオが自動的に起動するように設定されている場合に、ユーザに音をオンにする必要があるかどうかを選択させることができるようにします。ユーザは、**ビデオコントロール表示**のチェックボックスを選択するか、あるいはビデオプレーヤーで右クリックして選択するかのいずれかのビデオコントロールからの音を有効にすることができます。

進捗バー

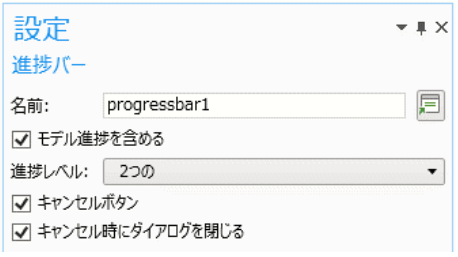
進捗バーオブジェクトは、メソッドによって更新される値に基づいて、一つのカスタマイズされた進捗バー、または二つの進捗バーのペアを表示します。進捗バーは、アプリケーションの実行時間の残量を知らせるために使われます。

下図は、一つの進捗レベルを持つ進捗バーオブジェクトの**設定**ウィンドウを示しています。(下図は、グリッドレイアウトモード時の場合です。)



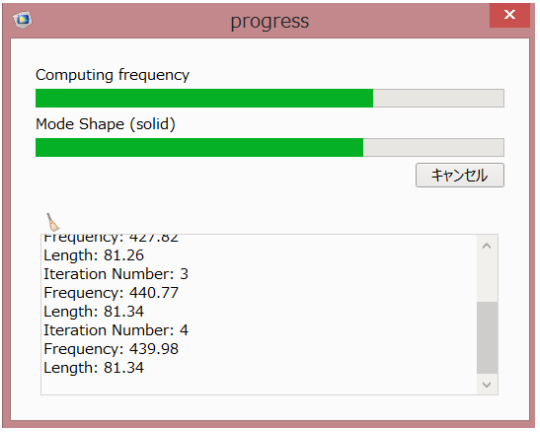
アプリケーションのステータスバー表示は、**メインウィンドウ**ノードの**設定**ウィンドウによって設定される組み込みの進捗バーです。デフォルトでは、この組み込みの進捗バーは、ジオメトリ作成、メッシュ作成、ソルバー球解などの埋め込まれている COMSOL マルチフィジックスのコアアルゴリズムの進捗を示しています。setProgress メソッドを使うことによって、組み込みの進捗バーで示される情報をカスタマイズすることができます。詳細については、313 ページの「進捗メソッド」と *Application Programming Guide* を参照してください。

下図には、二つの進捗レベルを持つ進捗バーオブジェクトの**設定**ウィンドウを示しています。

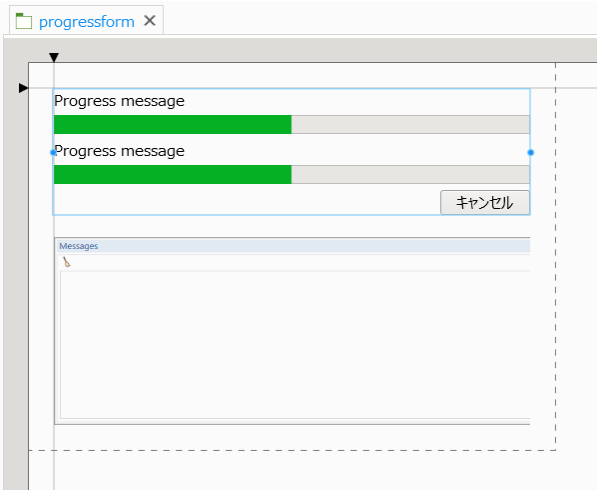


この例では、進捗バーオブジェクトは、二つのレベルを持つ進捗バーとメッセージログを表示するためのフォーム progressform に利用されています。

下図は、上記設定の場合にアプリケーション実行時に表示される進捗ダイアログボックスです。



その progressform フォームを下図に示します。



以下のコードセグメントは、進捗バーとメッセージログを更新するために使われる典型的な組み込みメソッドです。

```
// show progress dialog box :
dialog ( "progressform" ) ;
setProgressBar ( "/ progressform / progress1" , 0 , "Computing prong length." ) ;

// code for iterations goes here :
lastProgress = 20 ;
// ...

// update message log :
message ( "Iteration Number : " + k ) ;
message ( "Frequency : " + Math . round ( fq * 100 ) / 100.00 ) ;
message ( "Length : " + Math . round ( L1 * 100 ) / 100.00 ) ;

// update progress bar :
setProgressInterval ( "Computing frequency" , lastProgress , k * 100 /
MAXITERATIONS ) ;
// more code goes here :
// ...

// finished iterating :
setProgressBar ( "/progressform/progress1" , 100 ) ;
closeDialog ( "progressform" ) ;
```

前記の例において、進捗バーの二つのレベルを更新するための中心的な機能を担っている呼び出しは、以下の部分です。

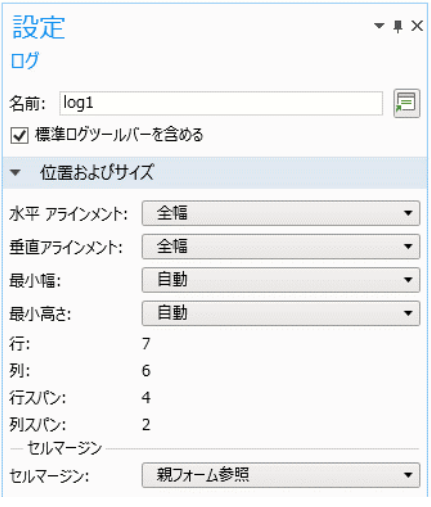
```
setProgressInterval ( "Computing frequency" , lastProgress , k * 100 / MAXITERATIONS ).
```

組み込みメソッドとそれらの記述方法についての詳細は、313 ページの「進捗メソッド」と *Application Programming Guide* を参照してください。

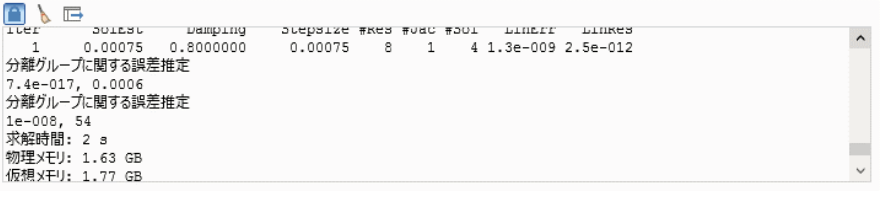
ログ

ログフォームオブジェクトは、ジオメトリ作成、メッシュ作成、ソルバー球解などの、埋め込まれている COMSOL マルチフィジックスのコアアルゴリズムからのメッセージを表示するログウィンドウです。

デフォルトでは、**標準ログツールバーを含める**のチェックボックスがチェックされています。このチェック設定がされている場合には、デスクトップ上の**ログ**ウィンドウにツールバーが表示され、実際にアプリケーションを実行した際にツールバーが含まれるようになります。（下図は、グリッドレイアウトモード時の場合です。）

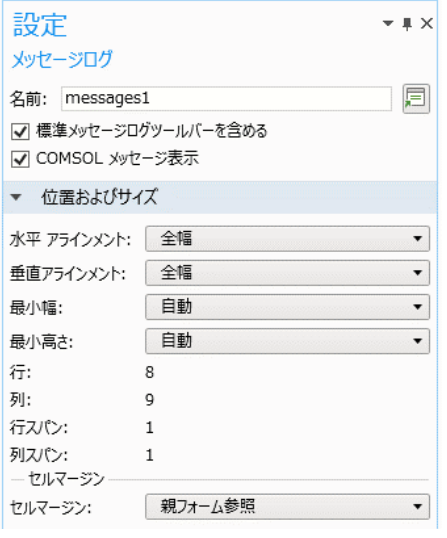


下図は、アプリケーションユーザインタフェースのログウィンドウの部分を示しています。



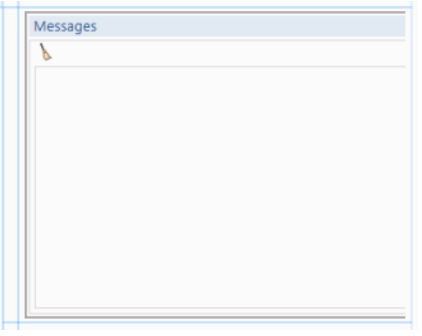
メッセージログ

メッセージログオブジェクトは、アプリケーションが実行するオペレーションをユーザに知らせるためのメッセージを表示してくれるウィンドウです。この機能は、組み込みメソッド message を使って実装することができ、message (String message) という記載方法によります。308 ページの「GUI 関連のメソッド」も参照してください。



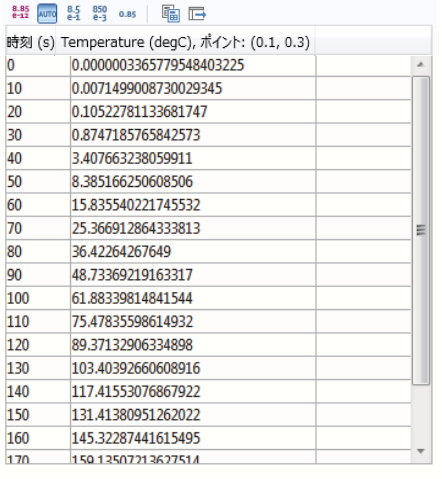
デフォルトでは、**標準メッセージログツールバーを含める**のチェックボックスがチェックされています。このチェックがされている場合には、デスクトップ上の**メッセージ**ウィンドウにツールバーが表示され、実際にアプリケーションを実行した際にツールバーが含まれるようになります。また、**COMSOL メッセージ表示**のチェックボックスもデフォルトでチェックされており、ジオメトリ作成、メッシュ作成、ソルバー球解といった、埋め込まれている COMSOL マルチフィジックスのコアアルゴリズムからのメッセージが有効となっています。そのチェックボックスがクリアされた設定では、アプリケーション自体からのメッセージ表示のみとなります。

下図には、メソッドからの収束情報が表示されるようにカスタマイズされたメッセージウィンドウ（左側）と、それに対応した**メッセージログ**フォームオブジェクト（右側）を示しています。



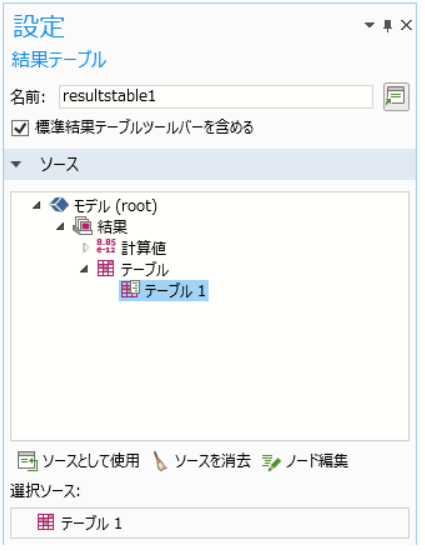
結果テーブル

結果テーブルオブジェクトは、数値結果をテーブル表示するために利用されます。



時刻 (s)	Temperature (degC), ポイント: (0.1, 0.3)
0	0.0000003365779548403225
10	0.0071499008730029345
20	0.10522781133681747
30	0.8747185765842573
40	3.407663238059911
50	8.385166250608506
60	15.835540221745532
70	25.366912864333813
80	36.42264267649
90	48.73369219163317
100	61.88339814841544
110	75.47835598614932
120	89.37132906334898
130	103.40392660608916
140	117.41553076867922
150	131.41380951262022
160	145.32287441615495
170	159.13507213627514

結果テーブルデータのソースには、**結果**(Results)ノードの下にある**計算値**(Derived Values)または**テーブル**(Tables)のサブノードのうちの一つが設定されます。下図では、**テーブル**ノードがソースとして使われています(ツリーのその項目を選択し、その下側の**ソースとして使用**ボタンをクリックすることによります)。



結果テーブルのツールバー

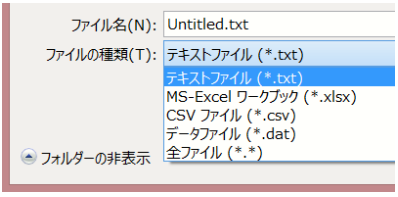
デフォルトでは、**標準結果テーブルツールバーを含める**のチェックボックスがチェックされています。これがチェックされている場合、結果テーブルに以下に示すツールバーが含まれます。

- **フル精度**
- **自動表記**
- **サイエンティフィック表記**
- **エンジニアリング表記**
- **小数表記**
- **テーブルとヘッダをクリップボードにコピー**
- **エクスポート**

エクスポートのボタンは、以下のファイル形式に対応したエクスポートに使用可能です。

- テキストファイル (.txt)
- Microsoft® Excel® Workbook (.xlsx)
  - LiveLink™ *for* Excel® が必要です。
- CSV ファイル (.csv)
- データファイル (.dat)

以下に、エクスポート時にファイル形式を選択する画面を示します。

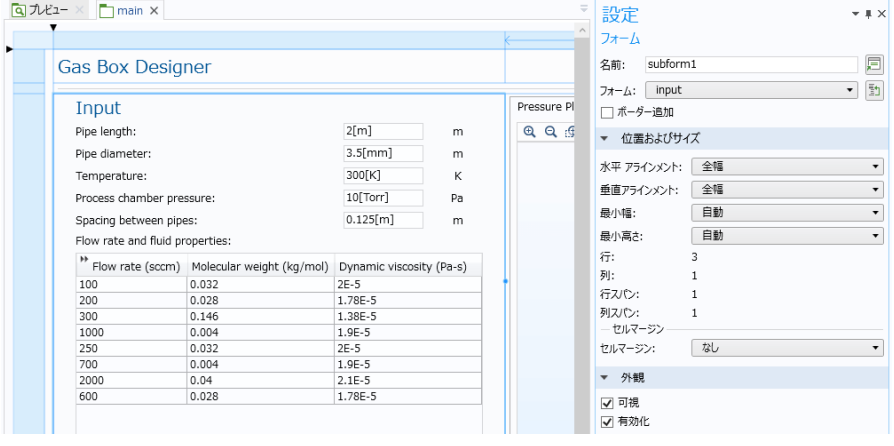


メソッドによる結果テーブルの制御

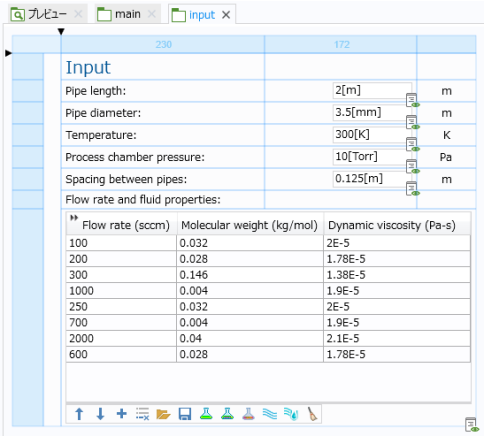
組み込みメソッド useResultsTable( ) によって、特定の結果テーブルオブジェクトにどのテーブルを表示するかを変更することができます。この組み込みメソッドの詳細については、308 ページの「GUI 関連のメソッド」を参照してください。

フォーム

フォームというタイプのフォームオブジェクトは、一つのメインフォームの中に一つ以上のサブフォームを構成するために使われます。追加配置するサブフォームにフォームを埋め込むには、サブフォームの**設定**ウィンドウの**フォーム**の参照設定でリンクしたいフォームを選択します。下図の例では、フォーム main の中のセルの一つに置かれたサブフォームがフォーム input にリンクされている場合を示しています。



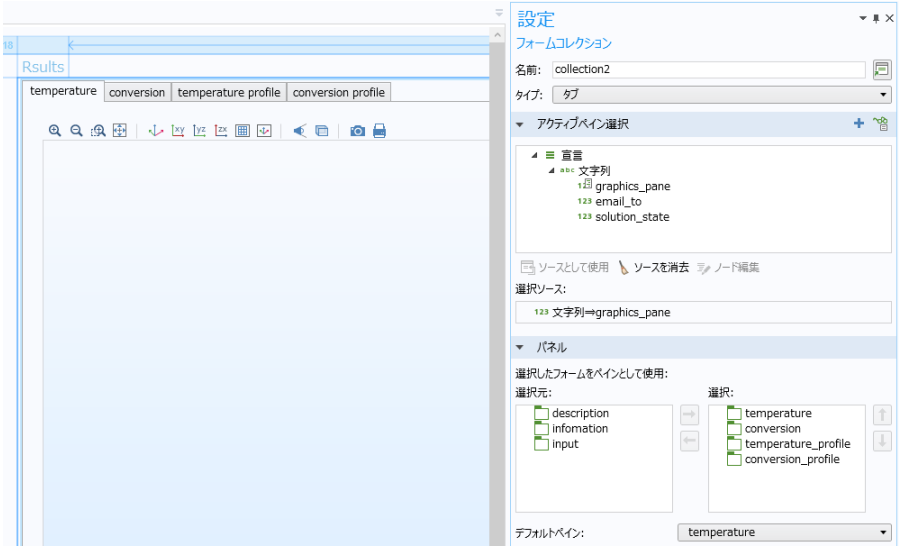
下図は、参照されているフォーム input そのものを示しています。



グリッドレイアウトモードを使用している場合、リボンにある**サブフォームを抽出**のボタンを使用して、すぐにサブフォームを作成することができます。106 ページの「サブフォームの抽出」を参照してください。

フォームコレクション

フォームコレクションオブジェクトは、メインフォームに表示されるいくつかのフォーム、あるいはペインから成ります。この例では、一つのメインウィンドウの中に、タブ形式で表示可能な四つのフォームが作られています。



四つの種類のレイアウト選択肢があり、設定ウィンドウの**タイプ**のリストから選択します。

- **タブ**は、タブ付きのペインを使ってフォームが表示されます。デフォルトでは、これが設定されています。
- **リスト**は、フォームを表示するペイン領域の左側にリストが表示され、表示したいフォームをそのリストから選択します。
- **セクション**では、フォーム毎に個別のセクションとして表示されます。
- **タイルまたはタブ**では、ブーリアン変数の真理値によって、タイルとタブの二つの方法のうちのどちらか一方が選ばれて表示されます。詳細については、本章のこれ以降の説明を参照してください。

設定ウィンドウの**パネル**のセクションで、**選択したフォームをペインとして使用**のリストにある各フォームがペインを表します。そこで選択されたリストの順番が、これらのフォームがアプリケーションで表示される順番になります。その順番は、パネルセクションの右側にある**上へ移動**と**下へ移動**のボタンをクリックして変更することができます。

アクティブペーン選択のセクションで文字列変数にリンクさせることによって、どのタブ（または、リスト項目）をアクティブにするかを制御することができます。

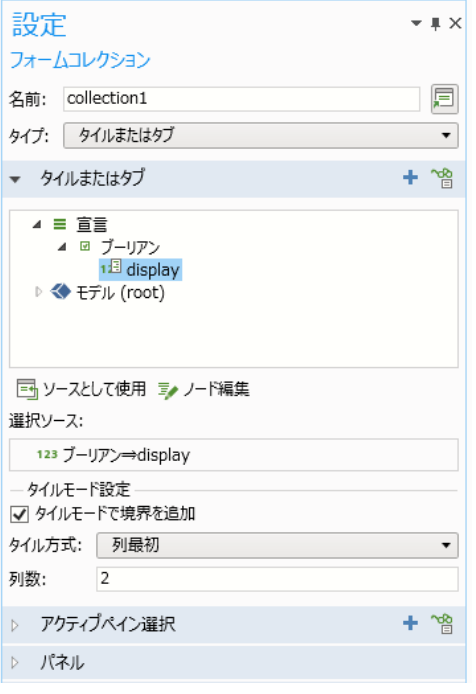
その文字列変数には、例えば上記の場合であれば、フォームコレクションで選択した temperature または conversion などのフォーム名のいずれかが設定される必要があります。そうでない場合、それは無視されます。

ある時点（例えばボタンメソッド）で実行されるメソッドの中で、アクティブペイン選択に設定した文字列変数（例えば pane）の値が変更されると、その新しい値に対応したペインがアクティブになります。以下にその例を示しています。

```
pane = ”conversion” ;  /* Activate the conversion pane on completion of this method  */
```

フォームコレクションの**タイプ**に**セクション**を設定した場合には、**アクティブペイン選択**は無効となります。**アクティブペイン選択**の利用は任意であって、タブをクリックする以外に、メソッドの中で有効なタブを制御するときにのみ必要です。**アクティブペイン選択**で設定した文字列変数を削除するには、そのツリーの下側にある**ソースを消去**のツールボタンをクリックします。

タイルまたはタブが選択された場合、設定ウィンドウの一番上にある**タイルまたはタブ**のセクションでソースとして選択されたブーリアン変数の真理値によって、タイルとタブの二つの方法のどちらか一方が選ばれてフォームが表示されます。



ここでのタブモードは、**タイプ**に**タブ**を設定した場合のフォームコレクションと同じです。タイルモードでは、全てのフォームは一つのグリッドの中に同時に表示されます。タイルモードのレイアウトは、サブセクションの**タイルモード設定**によって制御することができます。

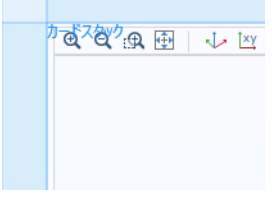
カードスタック

カードスタックは、カードを含むことができるフォームオブジェクトです。**カード**は、カードスタックのコンテキストの中でのみ使われる別のタイプのフォームオブジェクトです。毎回、カードをめくって、そのカードのうちの一つをカードスタックに表示します。カードスタックをデータソースに関連付け、そのデータソースによってどのカードが表示されるかを制御します。個々のカードに指定された値が、カードスタックのデータソースと比較されます。カードスタックは、最初に一致する値のカードを表示します。どのカードも一致しない場合は、何も表示しません。

グラフィックスオブジェクトの切り換えへのカードスタックの利用

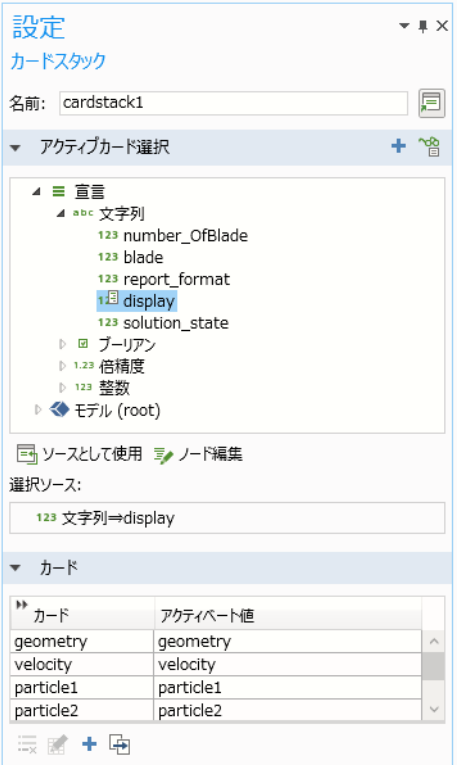
グラフィックスがスカラー変数の値によって表示されるアプリケーションを考えてみましょう。 例えば、ユーザがラジオボタンをクリックした時に、その変数が変更されるとします。また、例えば、モデルツリーの**グローバル評価**ノードの値といったような計算値によって、その変数を変更することができるとします。

下図は、フォームエディターのカードスタックオブジェクトを示しています。



この例では、カードスタックがグラフィックスオブジェクトを含んだカードを持っています。

下図は、カードスタックの**設定**ウィンドウを示しており、そこで五つのカードが選択され、**アクティブカード選択**で文字列変数 display がソースとして設定されています。

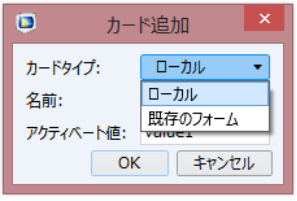


カードセクションのカードテーブルの中から一つのカードをクリックし、次にそのテーブルの下側にあるツールボタンの一つをクリックすることによって、以下のカード操作を実行することができます。

- **削除**
- **編集**
- **カード追加**
- **複製**

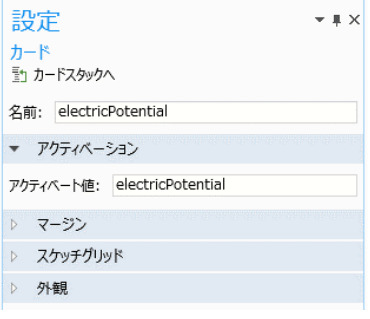
テーブルの各行には、カード名が**カード**の列に、それらに関連付けられたアクティベート値が**アクティベート値**の列に指定されます。スタックは、それらのアクティベート値によって表示するカードを判断します。この例では、アクティベート値は geometry、velocity、particle1 などの文字列です。

カード追加のボタンをクリックすると、以下のダイアログボックスが表示されます。

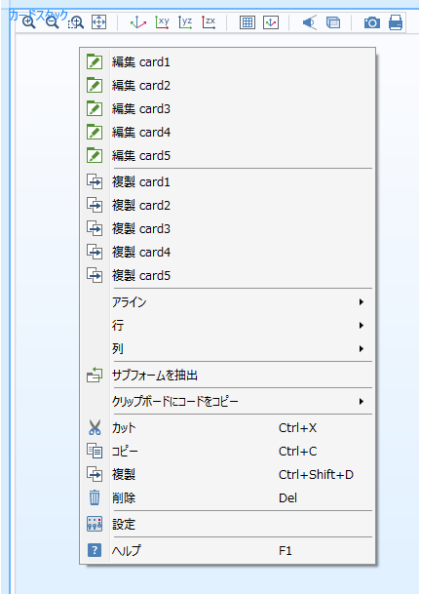


カードタイプは、デフォルトで**ローカル**に設定されています。これは、そのカードがカードスタックオブジェクトの中でローカルに定義されることを意味しています。その代わりに、**カード**タイプが**既存のフォーム**に設定される場合は、既存のフォームのうちの一つを選択することができます。**既存のフォーム**の設定を行うには、そのノードをクリックするか、または対応するカードスタックの**設定**ウィンドウの**カード**セクションで**編集**ボタンをクリックすることによって、そのフォームエディターから直接アクセスします。

カードセクションの表の下にある**編集**アイコンボタンをクリックすると、**カード**の**設定**ウィンドウの表示が下図に示すように変更されます。

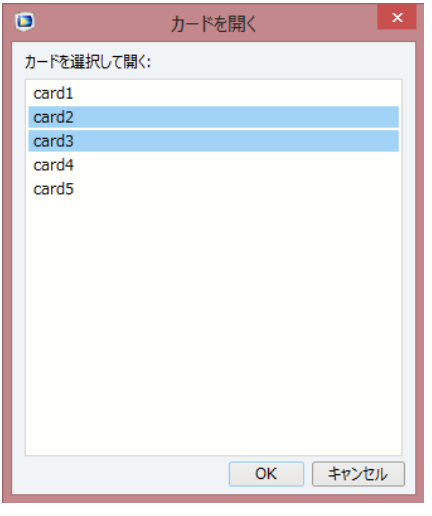


ローカルに定義されたカードにアクセスするには、下図に示すように、フォームウィンドウのカードスタックを右クリックし、カードスタックの中にあるカードから選択します。

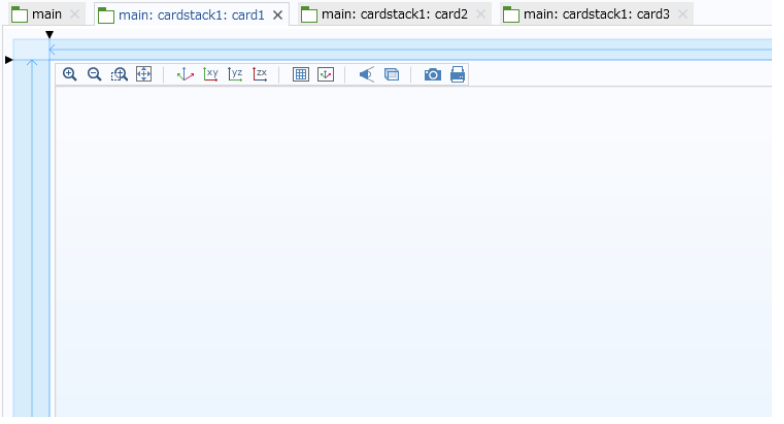


このメニューから、カードを複製することもできます。

カードを編集するために、Alt+click の操作を利用できます。これによって、ダイアログボックスが表示され、一度に複数のカードを選択することができます。



下図は、グラフィックスフォームオブジェクトを持つカードスタック card1 を示しています。

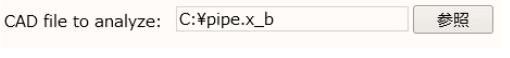


ファイルインポート

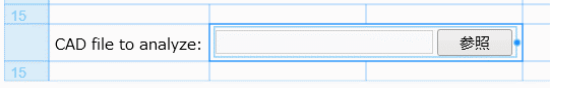
ファイルインポートオブジェクトは、ファイルブラウザーを表示するために利用します。そのファイルブラウザーは、ファイルを参照するため、またはそのパスと名前を入力するための入力フィールドを持っています。

アプリケーションの中で事前にファイルを入手しておくことができないような場合に、ユーザに対してアプリケーションを実行中にファイルインポートを有効にするために使用されます。

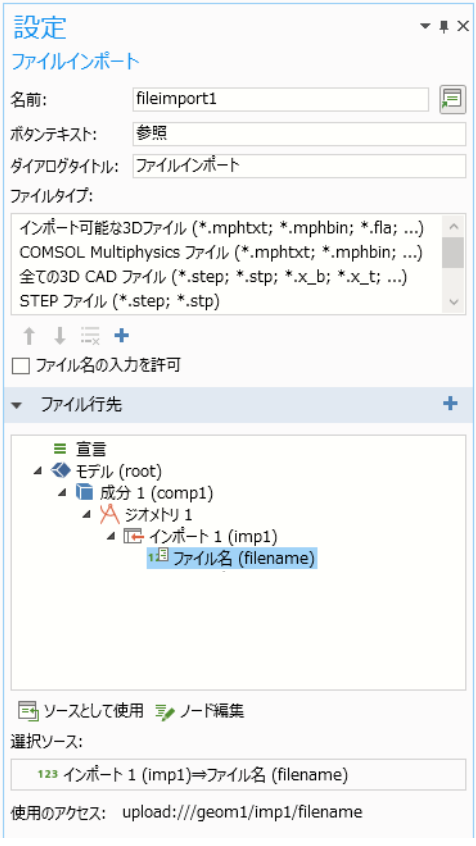
下図に示すように、実行中に CAD ファイルを選択してインポートできるアプリケーションを考えてみましょう。



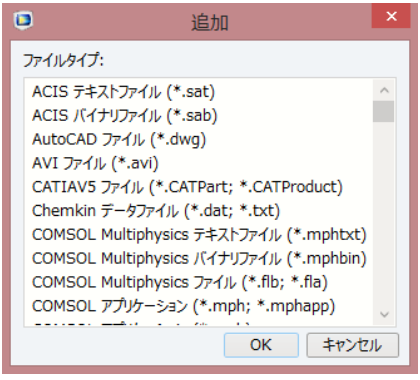
下図は、それに対応した**ファイルインポート**オブジェクトを示しています。



ファイルインポートオブジェクトの**設定**ウィンドウには、**ファイル行先**のセクションがあります。このセクションでは、ファイル名の入力が可能なツリーノードを選択することができます。下図に示した設定では、ジオメトリ（Geometry）の**インポート**ノードの**ファイル名**が選択されています。



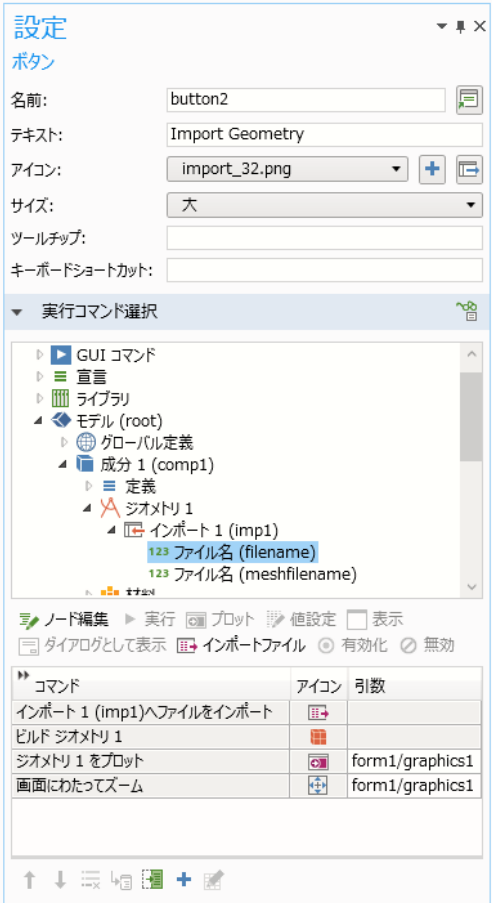
このアプリケーションでは、**ファイルタイプ**のテーブルの中で、CAD ファイルだけが許される指定になっています。**ファイルタイプ**のテーブルの下側にある**追加**と**削除**のボタンをクリックすることによって、許される**ファイルタイプ**をさらに制御することができます。**追加**のボタンをクリックすると、以下に示すダイアログボックスが表示されます。



ファイルインポートオブジェクト利用への代案

ファイルパスと名前のための入力フィールドが必要ではないならば、ファイルインポートの別の方法によってユーザがファイルブラウザーのファイルを取り出すことができます。例えば、メニュー、リボン、ツールバー項目、またはボタンを使うことができます。その場合は、そのボタンや項目のコマンドシーケンスに**ファイルを開く**コマンドを設定します。

下図に示したボタンの**設定**ウィンドウでは、CAD ファイルをインポートするように設定されています。

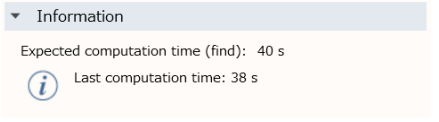


ファイルインポートオブジェクトは、**ファイル**宣言を参照することもできます。詳細については、137 ページの「ファイル」を参照してください。一般的なファイル処理に関する詳細情報は、279 ページの「付録 C—ファイル処理とファイルスキーム表記法」を参照してください。

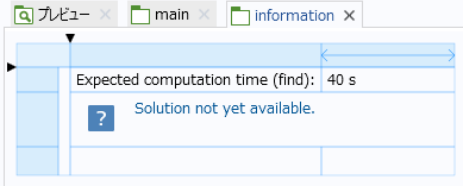
情報カードスタック

情報カードスタックオブジェクトは、ユーザからアプリケーションに与えられる入力とソリューションの関係についての情報を表示するために使われる、特殊なタイプの**カードスタック**オブジェクトです。

下図は、情報カードスタックが予想計算時間の情報と共に使われているアプリケーションにおいて、それが実行中の該当部分を示しています。



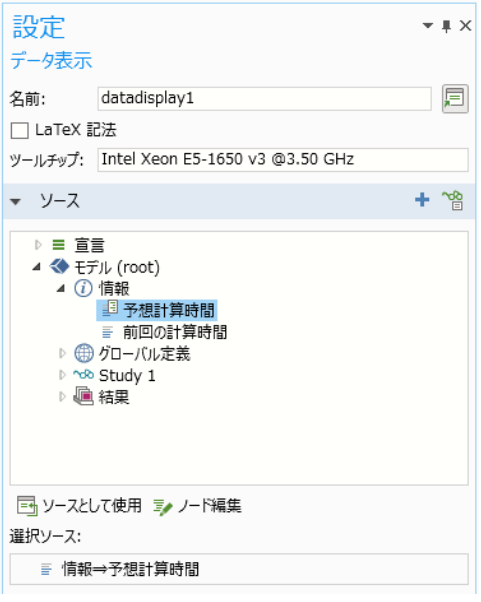
下図は、それに対応したフォームオブジェクトを示しています。



下図に示した**設定**ウィンドウでは、文字列変数 solution_state がソースとして設定されています。

カードスタックオブジェクトと類似していますが、個々のカードは**情報カード**のためのアイコンとテキストを持っています。上記の例では、どの**情報カード**が表示されるかは、文字列変数の値 nosolution、inputchanged、および solutionexists でコントロールされます。

この例では、情報カードスタックはデータ表示オブジェクトと共に利用されています。そのデータ表示オブジェクトでは、モデルツリーの情報(Information)ノードにある**予想計算時間**がソースとして設定されています。下図は、その**設定**ウィンドウを示しています。

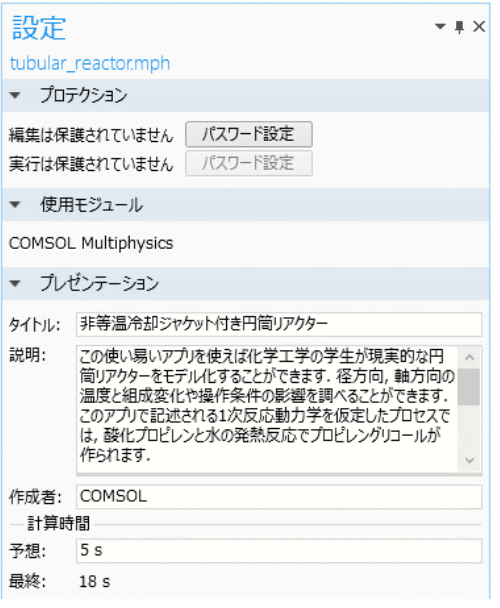


モデルツリーの情報(Information)ノードはアプリケーションビルダーでの作業中にだけ表示されることに注意してください。それらは、フォームオブジェクトの**設定**ウィンドウの**ソース**セクションで、適用可能な場合に選択することができるようになっています。

また、各スタディ(Study)の下の情報(Information)ノードにも**前回の計算時間**があることに注意してください。モデルノードのすぐ下の情報(Information)ノードにある**前回の計算時間**は、この前回計算されたスタディ(Study)の計算時間と一致します。

情報ノードは、入力フィールドオブジェクト、テキストオブジェクト、およびデータ表示オブジェクトのソースとして使うことができます。入力フィールドオブジェクトとテキストオブジェクトの設定で情報ノードを利用するためには、設定ウィンドウにある**編集可能**のチェックボックスをクリアして、情報ノードをアクセス可能とする必要があります。

予想計算時間には、下図に示すように、アプリケーションツリーの root ノードに設定されているデータが参照されます。



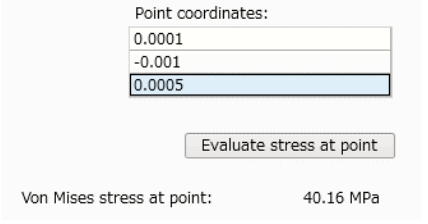
例えば、同じスタディが繰り返し呼び出されるといったように、計算時間の大部分がメソッドで使われる場合には、組み込みメソッド timeStamp と setLastComputationTime を使って計算時間を直接マニュアル測定することができます。詳細については、313 ページの「日付と時刻のメソッド」を参照してください。

配列入力

配列入力オブジェクトは、配列やベクトル値入力データを入力するために使われる入力テーブルを持っています。配列入力オブジェクトでは、データソースとしての文字列配列がサポートされています。ラベル、シンボル、および単位を設定によって追加することができます。

3D ポイント座標入力への配列入力オブジェクトの利用

ユーザがストレスを評価する点の 3D 座標を入力するアプリケーションを考えてみましょう。下図は、配列入力、ボタン、テキストラベル、およびデータ表示オブジェクトを持つアプリケーションのスクリーンショットです。



下図は、その配列入力オブジェクトの**設定**ウィンドウを示しています 。

配列入力フォームオブジェクトには、**倍精度**形式の 1D 配列である samplecoords という名前の**ソース**が設定されています。この**配列 1D 倍精度**の配列は、**配列入力**オブジェクトの作成に先がけて宣言されており、以下に示す**設定**となっています。

配列入力オブジェクトの**設定**ウィンドウにおいて、

- **長さ**フィールドには、配列の長さを正の整数で設定します。デフォルト値は 3 です。
- **ベクトル表示**のリストから**テーブル**（デフォルト）を選択すると、配列成分がテーブルとして表示されます。あるいは、リストから**成分**を選択すると、配列成分がラベル付きの個別の入力フィールドとして表示されます。
- **値**のテーブルには、配列成分のための初期値を設定します。
- **レイアウトオプション**セクションでは、配列入力へのコンポーネントラベルと単位の追加設定をします。

この例では、**Evaluate stress at point** というラベル名のボタンをクリックすると、次のメソッドが実行されます。

```
with  ( model . result ( ) . dataset ( "cpt1" ) ) ;
    set ( "pointx" , samplecoords[ 0 ] ) ;
    set ( "pointy" , samplecoords[ 1 ] ) ;
    set ( "pointz" , samplecoords[ 2 ] ) ;
endwith ( ) ;
```

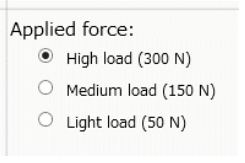
この Pointx、pointy、および pointz の設定値が、その後のストレス評価の座標として使われます。

ラジオボタン

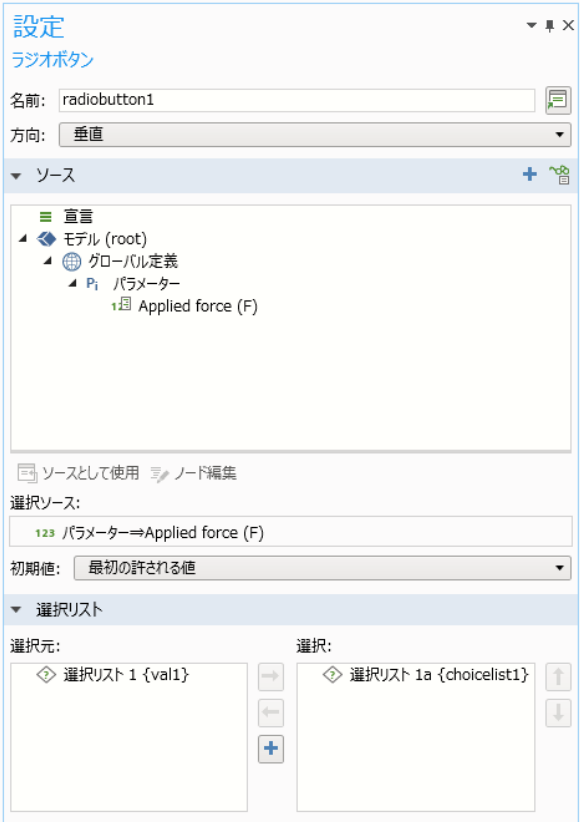
ラジオボタンオブジェクトには、割り当てた数の選択肢を持たせて、そこから一つを選択することができます。選択肢から一つだけを選ばせる用途では、最も適しています。

負荷の選択へのラジオボタンの利用

下図に示すように、ユーザが既定の 3 種類の負荷から一つを選択することができるアプリケーションを考えてみましょう。



それに対応した**設定**ウィンドウを下図に示しています。そこでは、グローバルパラメータ F がソースとして使われています。



方向は**垂直**（デフォルト）または**水平**を設定できます。

初期値のリストから、ラジオボタンの初期選択方法を選びます。その選択項目としては、**データソース参照**、**最初の許される値**（デフォルト）、および**カスタム値**があります。**カスタム値**を選択した場合は、加えて、選択リストで選択設定されているリストに定義された選択肢の中から一つを選びます。

選択リストのセクションでは、ラジオボタンにその選択肢を関連付ける選択リストを設定します。選択リストに定義された値の各々が、ラジオボタンの一つ一つを表します。

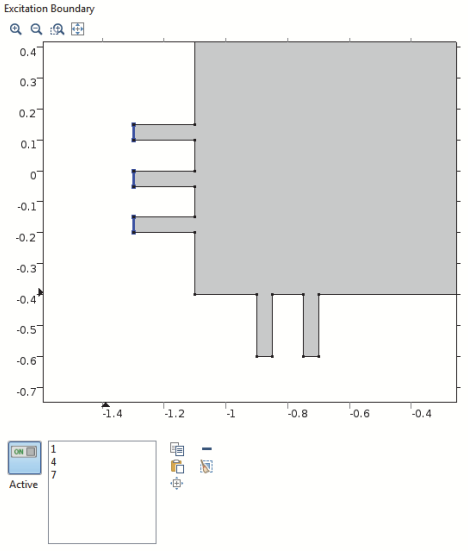
ラジオボタンの選択肢の表示名は、関連付けられている選択リストの**表示名**の欄の記載から参照されます。この例で使われている選択リストの例を、下図に示しています。

選択リストの代わりに単位セットの利用

ラジオボタンが単位を変更する目的で使用される場合には、**選択リスト**の代わりに**単位セット**を使用することができます（その場合も、ラジオボタンオブジェクトの**設定**ウィンドウの**選択リスト**のセクションで単位セットを選択する点は同じです）。

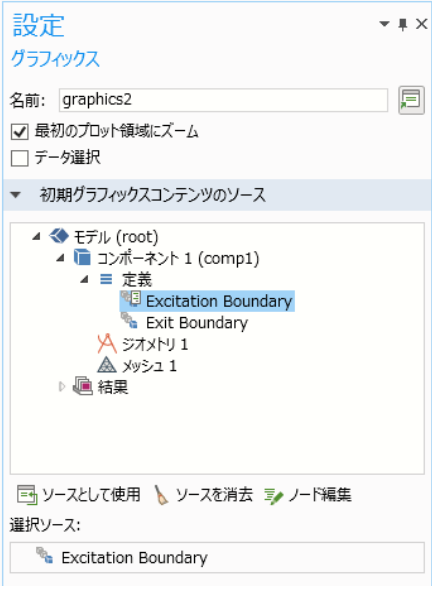
選択入力

アプリケーションビルダーでは、**選択入力**オブジェクトと**グラフィックス**オブジェクトを持つ**明示的**選択に属するエンティティを、アプリケーションのユーザにインタラクティブに変更させることができます。選択の詳細については、74 ページの「選択」を参照してください。以下の例では、埋め込まれているモデルが**明示的**選択によって定義された境界条件を持っています。この**選択入力**オブジェクトと**グラフィックス**オブジェクトによって、入力波で励起される境界をユーザに選択させています。



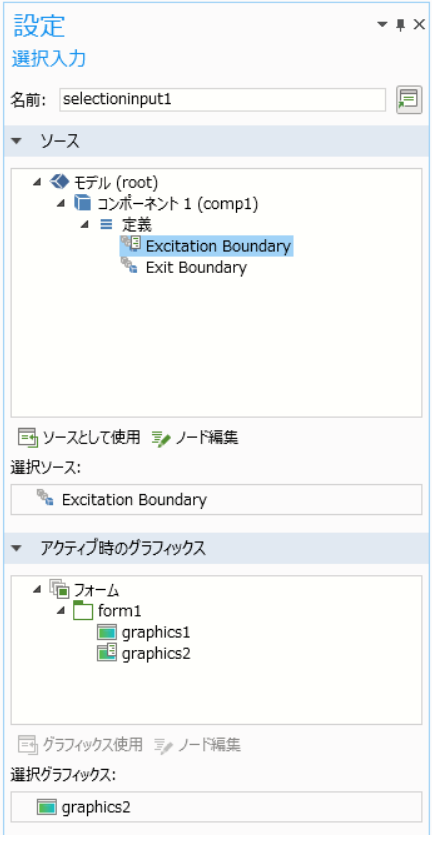
ここで、ユーザは、その**グラフィックス**オブジェクトのグラフィックスウィンドウで直接クリックするか、あるいは**選択入力**オブジェクトの境界番号リストでジオメトリのエンティティ番号を追加することによって、所望の境界を選ぶことができます。

境界をクリックして直接選択することができるようにするためには、下図に示すように、グラフィックスオブジェクトを境界のグループ化に使われている明示的**選択**にリンクさせます。明示的選択を選び、**ソースとして使用**をクリックします。次の図では、二つの明示的選択 **Excitation Boundary** と **Exit Boundary** が存在し、グラフィックスオブジェクト graphics2 は選択ソースとして **Excitation Boundary** にリンクしています。



このように、グラフィックスオブジェクトが明示的選択と直接リンクされている時には、グラフィックスオブジェクトはそのジオメトリを表示し、ユーザはそこでインタラクティブに境界をクリックすることができます。それによって、対応する明示的選択に境界が追加（または削除）されます。

下図に示すように、選択入力オブジェクトを明示的選択にリンクすることによって、番号による選択が可能となります。

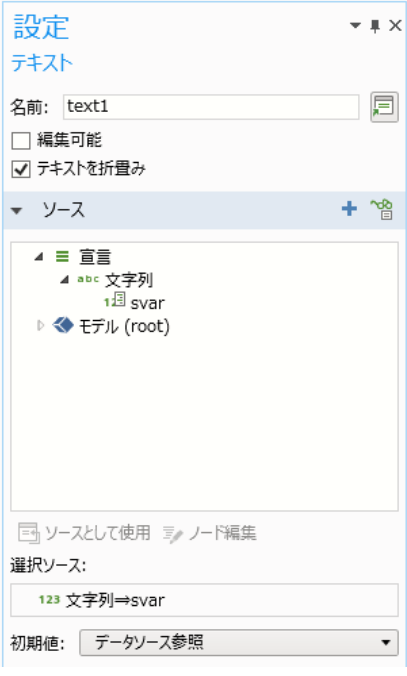


選択入力オブジェクトでは、コピー、ペースト、削除、クリア、および選択箇所の拡大が可能です。

選択入力オブジェクトなしでも、グラフィックスオブジェクトだけを選択ソースとして使って選択することもできます。また、グラフィックスオブジェクトと選択入力オブジェクトの両方を同じ明示的な選択にリンクさせることもできます。

テキスト

テキストオブジェクトは、文字列変数 または**情報**(Information)ノードの内容を参照して掲載するデフォルトテキストを持ったテキストフィールドです。テキストオブジェクトの**設定**ウィンドウを下図に示します。



ソースセクションのツリーから文字列変数 または**情報**(Information)ノードを選び、それから、**ソースとして使用**をクリックします。**初期値**として**カスタム値**を選択した場合には、その下の**値**フィールドに初期のテキストを入力します。デフォルトでは、**初期値**の設定は**データソース参照**になっています。

編集可能のチェックボックスは、デフォルトではクリアされています。これにチェックを入れた場合には、例えば、実行中のアプリケーションでテキストオブジェクトにコメントを書き込むといった用途に利用することができます。このように、テキストがユーザによって変更される場合、**初期値**の設定に関わらず、それはデータソースとして使われている文字列変数に格納されます。

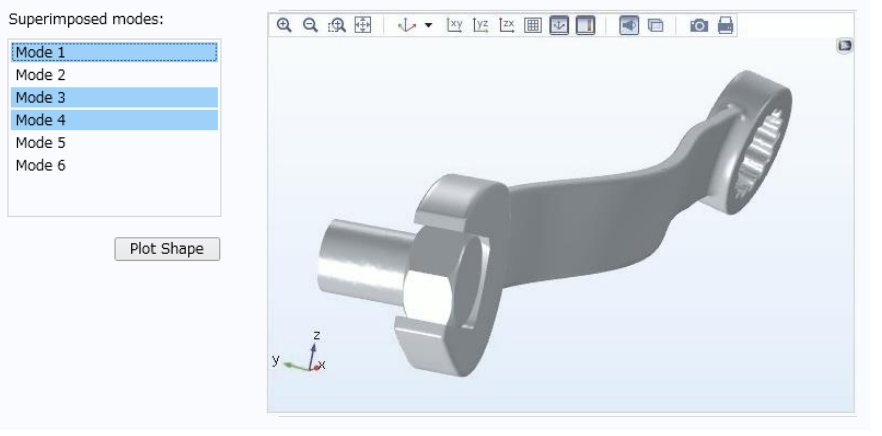
テキストを折畳みのチェックボックスは、デフォルトでチェックされています。これをクリアした場合は、テキストは折畳まれず、テキストがいっぱいになった時にはスクロールバーが表示されます。
情報(Information)ノードの詳細については、86 ページの「データ表示」を参照してください。

リストボックス

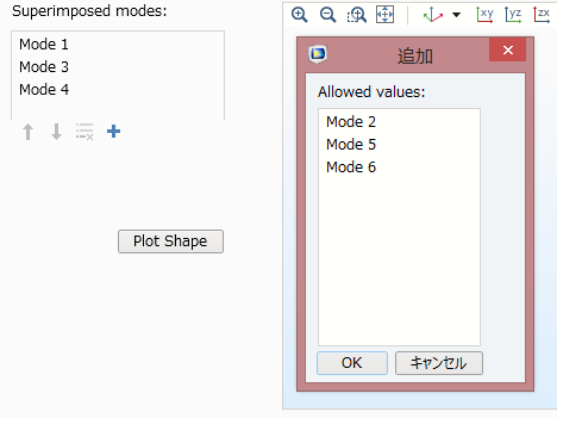
リストボックスオブジェクトは、それが複数の選択肢を同時に選択することが可能であること以外は、ラジオボタンオブジェクトと同様です。

重畳振動モードへのリストボックスの利用

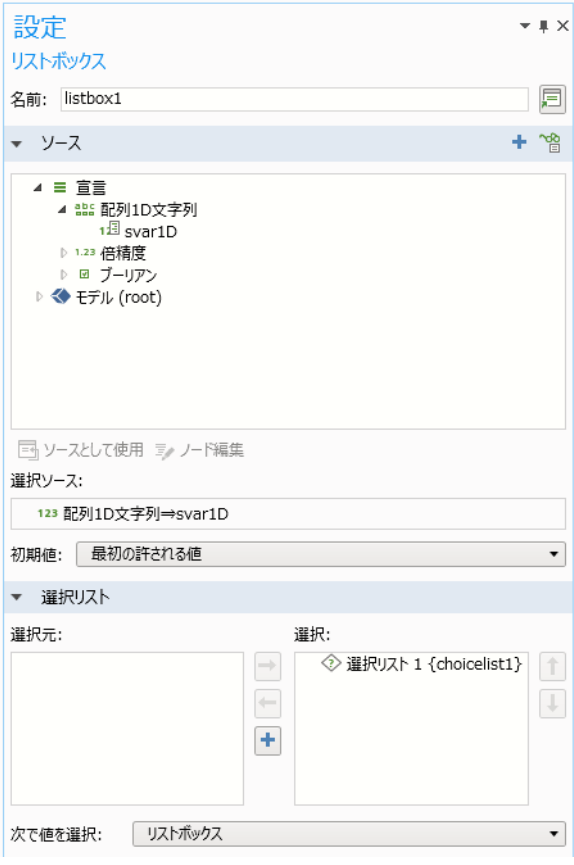
下図に示すように、機械的な部品の最初の六つの振動モードをリストボックスから選ぶことによって、重畳され視覚化できるアプリケーションを考えてみましょう。



一つの表示方法として、下図は、リストがダイアログボックス表示されている場合を示しています。



下図は、この例におけるリストボックスの**設定**ウィンドウです。

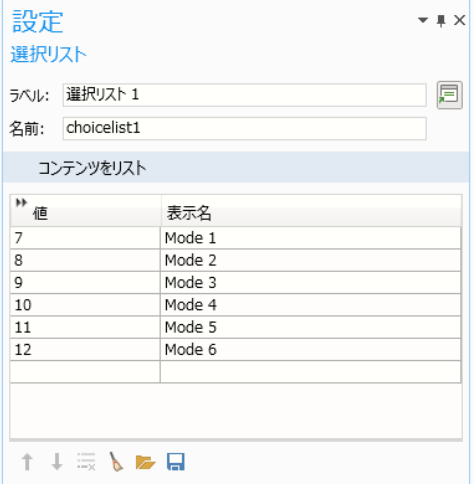


次で値を選択のリストにある**リストボックス**と**ダイアログ**の二つの選択肢から、リストの表示方法を選択することができます。

ソースとしては、スカラー、または配列の宣言が使われます。ツリーから選択し、**ソースとして使用**をクリックします。ソースとして文字列配列を使っている場合には、実行中のアプリケーションで Shift+click または Ctrl+click の操作を使ってリストの複数の項目を選択することができます。

その他のソースの場合には、リストから単に一つの項目しか選択できません。この例では、文字列配列 1D である svar1D を使っています。その**設定**ウィンドウを下図に示しています。

選択リストセクションでは、リストボックスに掲載する選択肢を提供する選択リストを追加することができます。下図は、この例で使われている選択リストを示しています。



振動モード 1－6 は通常の剛体モードに相当し、このアプリケーションでは重要ではありません。このため、**値**の欄は 7 から始まっています。選択リストでは、**表示名**の欄の文字列が表示されるだけであり、ユーザに対してそのモデルの実際のモード値は隠されます。最初の非剛体モードから順に Mode 1、Mode 2、などと名付けられています。

以下のメソッドでは、重畳モードを視覚化するために、COMSOL マルチフィジックスのステートメント with( ) が使われています。この例は、多少簡略化されており、モードの振幅と位相の効果が無視されています。

```
String withstru = "0" ;
String withstrv = "0" ;
String withstrw = "0" ;
for ( int i = 0 ; i < svar1D . length ; i ++ ) {
   withstru = withstru + "+" + "with (" + svar1D[ i ] + ", u )" ;
   withstrv = withstrv + "+" + "with (" + svar1D[ i ] + ", v )" ;
   withstrw = withstrw + "+" + "with (" + svar1D[ i ] + ", w )" ;
}

with ( model . result ( "pg7" ) . feature ( "surf1" ) . feature ( "def" ) ) ;
   setIndex ( "expr" , withstru , 0 ) ;
   setIndex ( "expr" , withstrv , 1 ) ;
   setIndex ( "expr" , withstrw , 2 ) ;
endwith ( ) ;
useGraphics ( model . result ( "pg7" ) , "/ form1 / graphics8" ) ;
zoomExtents ( "/ form1 / graphics8" ) ;
```

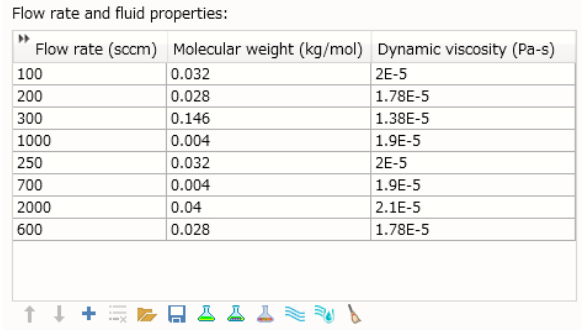
ユーザがリストボックスを使ってモード 1、3、および 5 を選んだと仮定すると、メソッドで作られる表記としては、with (1 , u) + with (3 , u) + with (5 , u) となります。この表記は、置換プロットの x 置換（従属変数 u）のために使われています。同様に、y と z 置換それぞれに関する変数 v と w のためのメソッド表記が自動的に作成されます。上の例の結果で使われているコマンド with( ) は、317 ページの「With、Get、および Set メソッド」で説明されている短縮表記で用いられている組み込みの with( ) コマンドとは異なる点に注意してください。

選択リストの代わりに単位セットの利用

リストボックスが単位を変更する目的で使用される場合には、**選択リスト**の代わりに**単位セット**を使用することができます（その場合も、リストボックスの**設定**ウィンドウの**選択リスト**のセクションで単位セットを選択する点は同じです）。

テーブル

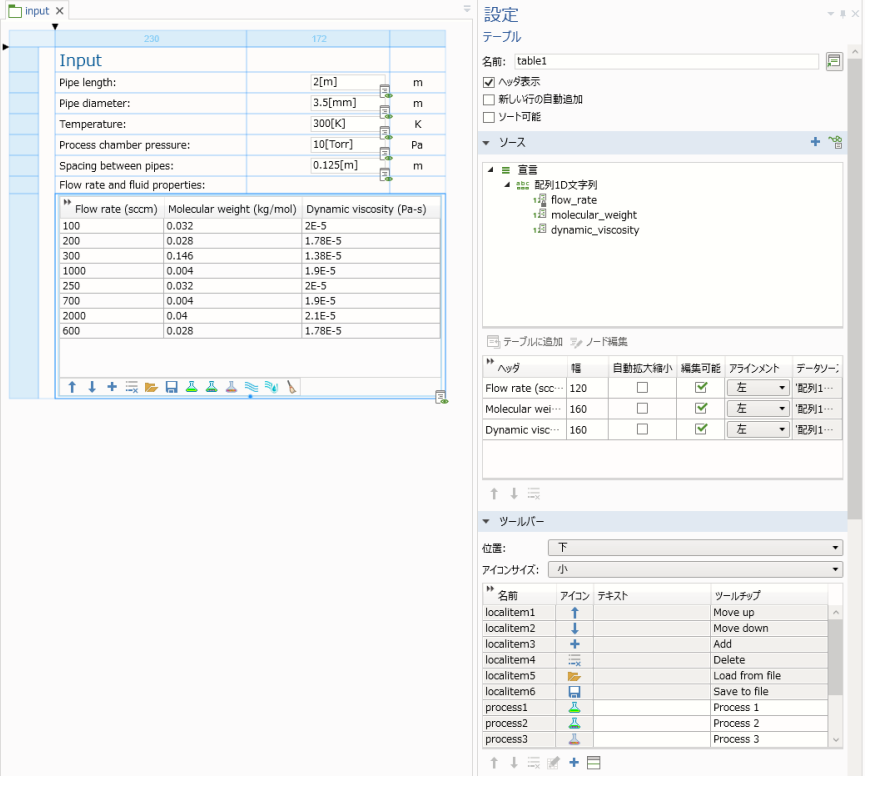
テーブルオブジェクトは、入力と出力を定義するために使うことができる行と列を持ったテーブルを表示します。下図は、実行中のアプリケーションにおいて、テーブルオブジェクトの三つの列に入力が対応付けられている例を示しています。



Flow rate and fluid properties:

Flow rate (sccm)	Molecular weight (kg/mol)	Dynamic viscosity (Pa-s)
100	0.032	2E-5
200	0.028	1.78E-5
300	0.146	1.38E-5
1000	0.004	1.9E-5
250	0.032	2E-5
700	0.004	1.9E-5
2000	0.04	2.1E-5
600	0.028	1.78E-5

下図は、それに対応したフォームオブジェクトと、その**設定**ウィンドウを示しています。



この例では、データソースとして三つの文字列配列 1D を参照しています。ソースとしてどのタイプの配列も選ぶことができ、その後に**ソースとして使用**をクリックします。

以下の三つのチェックボックスによって、テーブル全体の外観が制御されます。

- **ヘッダ表示**
- **新しい行の自動追加**
- **ソート可能**

新しい行の自動追加のチェックボックスをチェックした場合には、テーブルへの書き込みがいっぱいになった際、常に追加の空白行が利用可能となります。そのテーブルのソースとして使われている全ての文字列配列 1D において、各宣言の**設定**ウィンドウの**新規要素値**の欄に空白でない値が設定されている場合には、この機能は無効となります。この場合に新たな行を追加するには、単にテーブルのツールバーに関連付けた**追加**ボタンをクリックします。

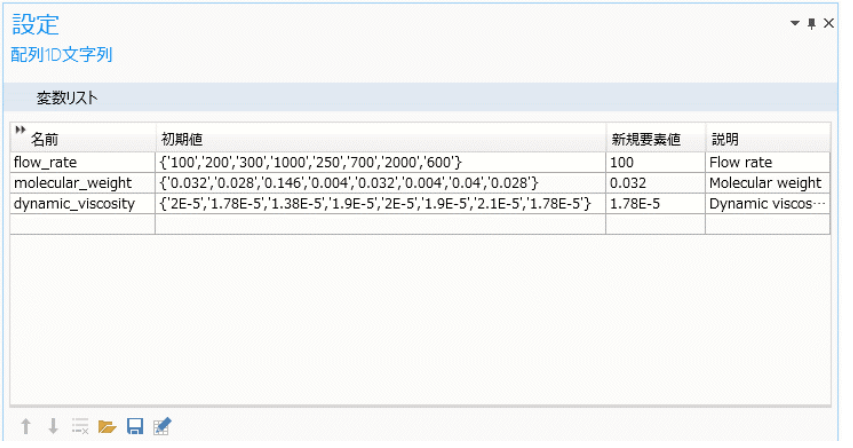
ソート可能のチェックボックスをチェックした場合には、特定の列のヘッダをクリックすることによって、その列に対してテーブルをソートすることが可能です。

ソースセクションのテーブルには、以下の六つの列があります。

- **ヘッダ**
- **幅**
- **自動拡大縮小**
- **編集可能**
- **アラインメント**
- **データソース**

このテーブルの行の各々が、テーブルオブジェクトの列を定義しています。**自動拡大縮小**のオプションは、フォームのサイズが変更された際に、個々の列が拡大縮小することができます。このオプションは、グリッドモードでテーブルの**水平アラインメント**が**全幅**に設定されている場合にのみ適用されます。

例では、下図に示すように、文字列配列の設定テーブルの各行に、それに対応するテーブルオブジェクトの三つの列の初期値が定義されています。



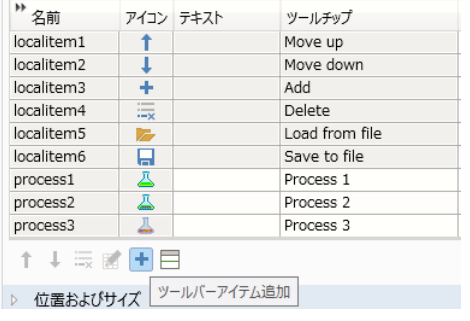
ツールバー

このセクションでは、テーブルの内容を制御するために使うツールバーアイテム（ボタン）を選択して設定します。**位置**のリストは、以下の選択肢からテーブルに対するツールバーの配置を定義します。

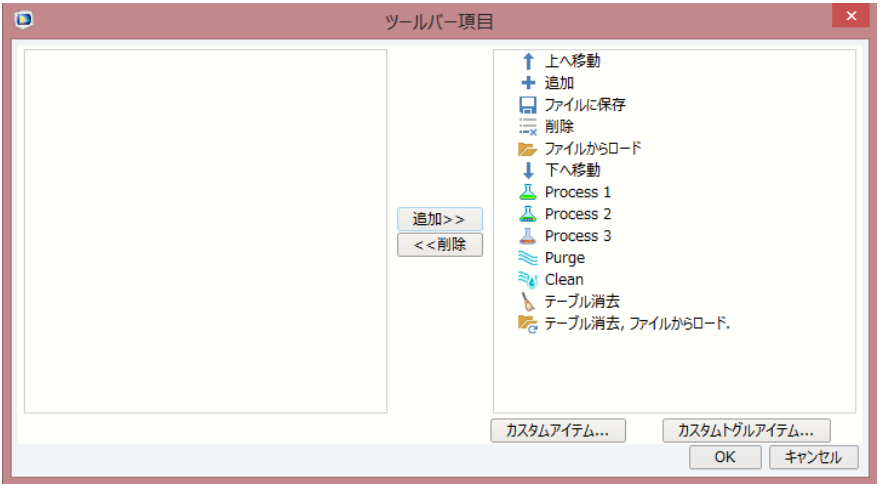
- **下**
- **上**
- **左**
- **右**

アイコンサイズの設定では、**小**さいアイコンまたは**大**きいアイコンを選択できます。

ツールバーにアイテムを追加するには、テーブルの下側にある**ツールバーアイテム追加**ボタンをクリックします。



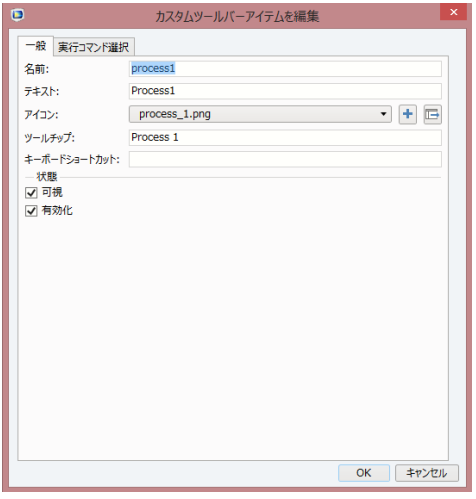
その際、下図に示すダイアログボックスが表示されます。

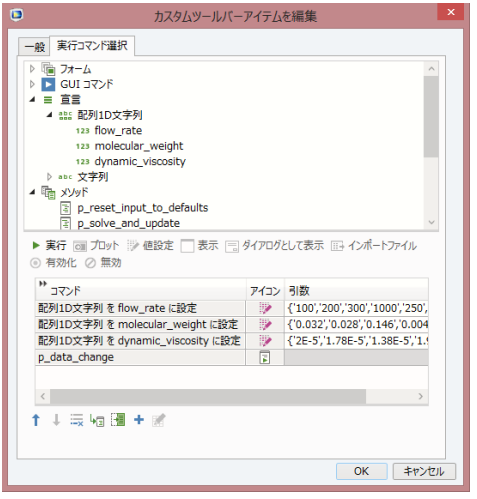


次の項目を追加することができます。

- **ファイルに保存**
- **上へ移動**
- **下へ移動**
- **追加**
- **削除**
- **テーブル消去**
- **テーブル消去、ファイルからロード**
- **ファイルからロード**

さらに、**ツールバー項目**のダイアログボックスにある**カスタムアイテム**または**カスタムトグルアイテム**をクリックして、カスタマイズされたアイテムを追加することができます。下図は、カスタマイズされたボタンを定義するために使用される**カスタムツールバーアイテム編集**のダイアログボックスを示しています。ダイアログボックスには、通常のアイテム用の二つのタブとトグルアイテム用の三つのタブがあります。この場合、**Process 1** ボタンを使用して、特定の処理にデフォルト値を設定します。



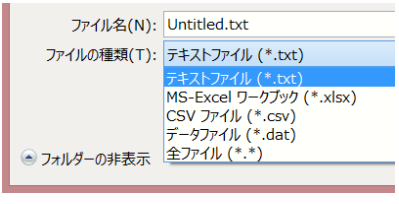


ここにある**実行コマンド選択**のタブは、ボタンの他にメニュー、リボン、およびツールバー項目における設定の場合に類似しています。

ファイルからロードと**ファイルに保存**のボタンは、以下のファイル形式に対するロードと保存に対応しています。

- テキストファイル (.txt)
- Microsoft® Excel® Workbook (.xlsx)
  - LiveLink™ *for* Excel® が必要です。
- CSV ファイル (.csv)
- データファイル (.dat)

以下に、ファイル形式を選択する画面を示します。



許可されるセパレータは、コンマ、CSV ファイル用のセミコロンとタブ、および DAT や TXT ファイル用のスペースとタブです。

スライダー

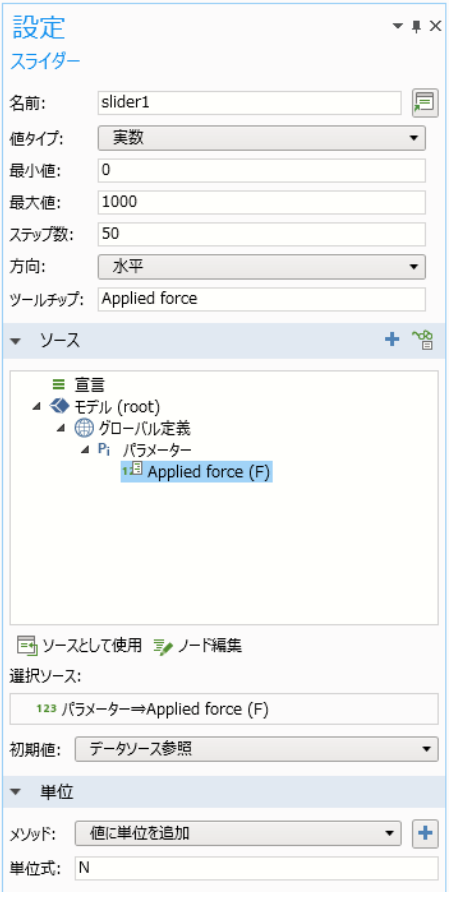
スライダーは、スライダーコントロールを使って数値入力を選ぶためのフォームオブジェクトです。

構造負荷の変更へのスライダーの利用

下図に示したような、負荷の大きさがスライダーコントロールで変更することができるアプリケーションを考えてみましょう。

この例では、スライダーで選ばれた値を表示するための入力フィールドが、スライダーといっしょに配置されています。

下図に、スライダーの**設定**ウィンドウを示しています。



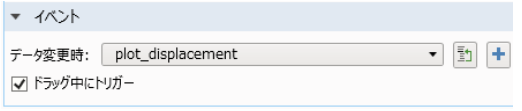
この例では、スライダーのソースとしてグローバルパラメータ F が使われています。ソースとしては、パラメータ、変数、または宣言されたスカラー変数のいずれが選択されても構いません。アプリケーションツリーから選択し、**ソースとして使用**をクリックします。

値タイプのリストから、スライダーのデータソースのデータタイプに合わせて、**整数**または**実数**（デフォルト）を選択します。

スライダーの**最大値**、**最小値**、および**ステップ数**を定義して、データソースの値の範囲を決めます。**方向**では、**水平**か**垂直**かを設定できます。また、スライダーの上でカーソルをホバリングした時に表示される**ツールチップ**を設定することもできます。**値に単位を追加**を選択すると、単位をスライダーに関連付けることができます。この単位は、ソース変数に値が渡される前に、[N] のように標準の角括弧を使って数値に付加されます。上記の例では、入力フィールドとスライダーの両方とも、**値に単位を追加**の設定が有効にされています。**値に単位を追加**の代わりに、**単位セットから単位を追加**を選択することができます。 詳細については、138ページの「単位セット」を参照してください。

初期値のリストからは、スライダーの初期値として、**データソース参照**または**カスタム値**を選択します。

イベントセクションでは、**データ変更時**のイベントを呼び出すメソッドを指定するほか、**ドラッグ中にトリガー**チェックボックスをオンにすることもできます。この設定は、スライダーがドラッグされている間、またはそのリリース時にのみ、イベントメソッドを継続的に呼び出すかどうかを決定します。

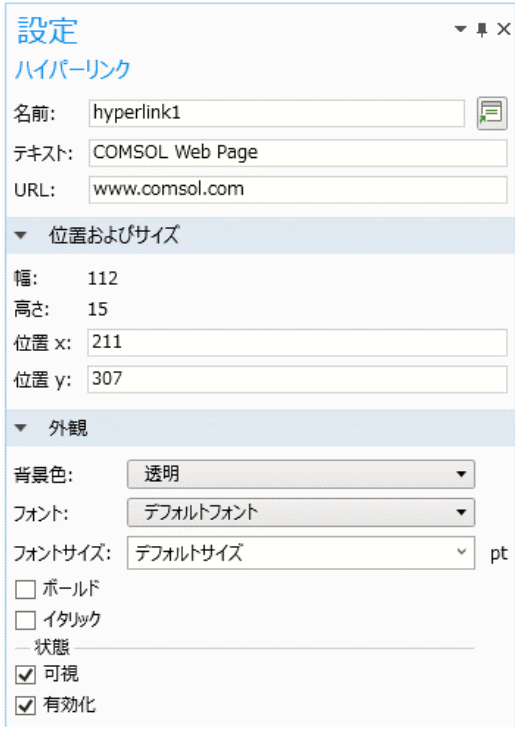


この設定は、**データ変更時**のイベントによって呼び出されるメソッドが計算上重く、スライダーをドラッグするときに遅れがある場合に便利です。

ハイパーリンク

ハイパーリンクオブジェクトは、フォームにハイパーリンクを組み込みます。下図は、ハイパーリンクの一例を示しています。

下図は、対応する**設定**ウィンドウを示しています。



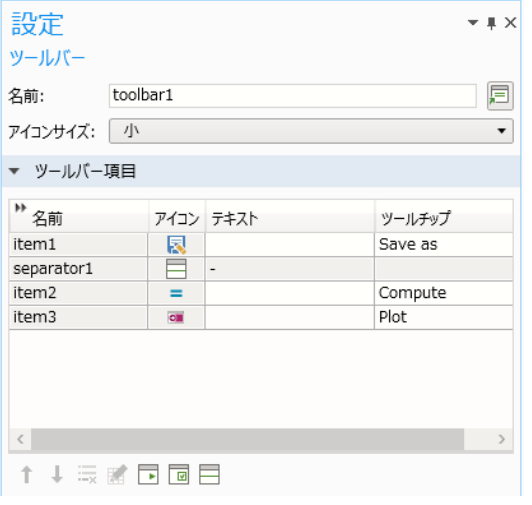
ハイパーリンクオブジェクトは、ウェブブラウザで使用することができる URL のタイプをサポートしており、以下を含みます。

- ウェブページ： ウェブページのハイパーリンクをクリックすると、ユーザのデフォルトブラウザーで開かれます。URL 文字列は、protocol://address の形式である必要があり、プロトコルは伝送プロトコルです。例えば、HTTP または HTTPS です。ウェブアドレスは部分的または完全な記述が可能ですが、完全なウェブアドレスを使用することをお勧めします。
- 電子メール： 電子メールアドレスは、mailto:emailaddress の形式で指定されます。これは、ユーザのデフォルトの電子メールアプリケーションプログラムを起動し、To フィールドに指定アドレスが設定された新規メッセージを用意します。対話形式で COMSOL アプリケーションから電子メールを送信するこの方法は、組み込みメソッドを使用した場合とは異なります。電子メールのための組み込みメソッドの詳細については、305 ページの「電子メールメソッド」を参照してください。

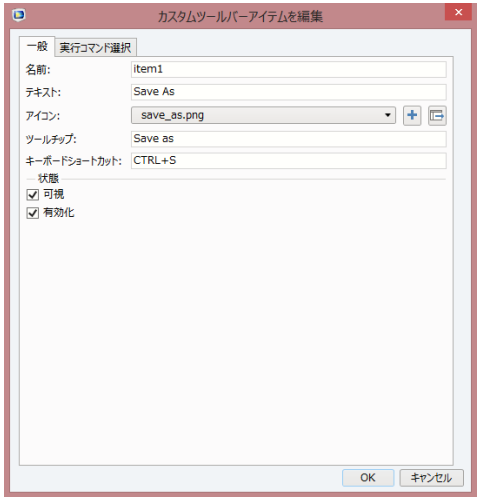
ツールバー

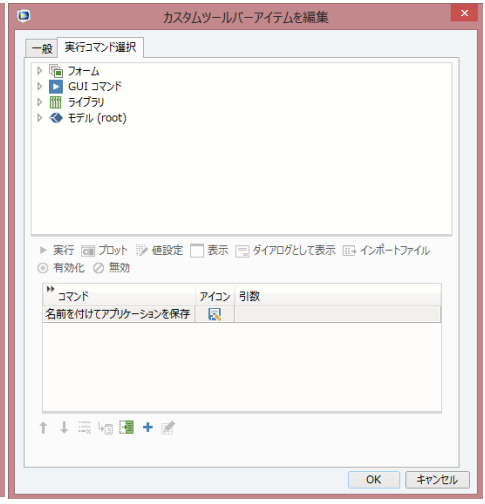
ツールバーオブジェクトは、ツールバーに設定される個々のツールバーボタンを含んでいます。下図は、**保存**、**計算**、および**プロット**のボタンを備えたツールバーを示しています。

このツールバーの**設定**ウィンドウを、下図に示します。



ツールバーアイテムのテーブルの各行には、ツールバーボタンに対応した**アイテム**か**セパレーター**が含まれています。 テーブルの下側にあるボタンを使って、**アイテム**または**セパレーター**の追加、行の順番の入れ換え、または行の削除が可能です。 **編集**ボタンをクリックすると、各行に関連付けられた**設定**ウィンドウが表示されます。 次の図は、**item1**、**名前を付けて保存**のアイテムの**設定**ウィンドウを示しています。



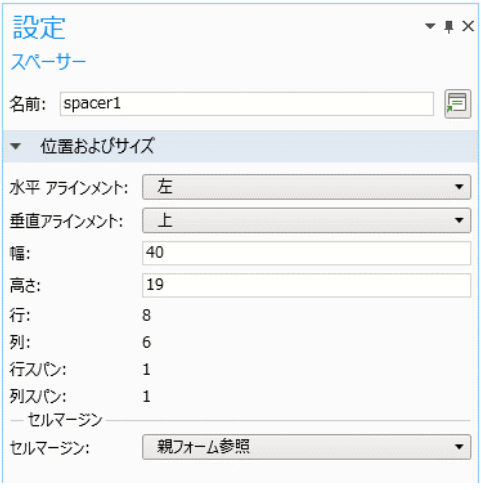


ツールチップフィールドに入力されたテキストは、ツールバーボタンにカーソルをホバリングした時のツールチップとして表示されます。**テキスト**フィールドのテキストは、アイコンが存在する場合 はその横に表示されます。アイコンがない場合には、テキストのみが表示されます。**アイコン**のリスト、**キーボードショートカット**のフィールド、および**実行コマンド選択**のツリーに関しては、ボタンオブジェクトの設定と同じ機能が表示されています。詳細については、51 ページの「ボタン」を参照してください。

スペーサー

スペーサーオブジェクトは、ユーザインタフェースで表示されるものではなく、単にグリッドレイアウトモードでの作業中に利用されるものです。それによって、隣接するフォームオブジェクト同志が十分な表示スペースを確保できるように、固定サイズのスペースを定義することができます。一般に、テーブルまたはグラフィックスオブジェクトの隣にスペーサーを使用し、確実にそれらが適切に表示されるようにします。もしユーザがウィンドウのサイズをリサイズしてスペーサーのサイズより小さくしてしまった場合は、スクロールバーが表示されることによってウィンドウの実際に使われるサイズが維持されます。

下図は、スペーサーオブジェクトの**設定**ウィンドウを示しています。



付録 B—アプリケーション間のコピー

アプリケーションツリーの多くのノードは、以下を含むアプリケーション間でのコピーペーストが可能です:

フォーム、フォームオブジェクト、メニュー項目、メソッド、Java® ユーティリティメソッド、外部ライブラリ、ファイル宣言、選択リスト宣言、メニュー、メニュー項目、リボンセクション、リボンタブ、リボン項目

アプリケーション間でフォーム、フォームオブジェクト、および項目をコピーペーストする時、コピーされるオブジェクトが他のオブジェクトや項目への参照を含んでいるかもしれません。そのような場合、その参照がコピー先のアプリケーション側では有効でないこともあります。オブジェクトがクリップボードから貼り付けられる時には、以下の規則が適用されます。

- フォームオブジェクトまたはメニュー項目で参照している宣言は、オブジェクトがコピーされた時には含まれていますが、必ずしも貼り付けられるとは限りません。コピー先にそれと互換の宣言が全くない場合には、単純に貼り付けられます。コピー先に互換の宣言が存在する場合には、それが代わりに使われます。ここで、互換の宣言とは、同じ名前とタイプを持っているものと定義しています。例えば、文字列宣言は整数宣言と互換ではありません。既存の宣言に矛盾するデフォルト値が設定されているかもしれませんが、貼り付けられる際にそのようなチェックは全くされません。
- 参照されているグローバルパラメータは異なる単位を持っているかもしれませんが、この時点では常に互換であるとされます。
- 別のフォームオブジェクトから直接参照されているフォームまたはフォームオブジェクトは、オブジェクトをコピーする時に自動的には含まれません。直接参照しているオブジェクトが既存のオブジェクトと同じ名前である場合には、その参照は既存のオブジェクトを指していることとされます。コピーされたオブジェクトの中に元々参照していたオブジェクトがあるような場合には、そのオブジェクトと同じ名前のオブジェクトが存在していたとしても、そのオブジェクトがそのまま参照として使われます。コピーされた参照の名前は、名前の重複を避けるために変更されます。
- モデルツリーのオブジェクトは、自動的にはコピーされません。例えば、ジオメトリを参照しているグラフィックスオブジェクト、またはデータアクセスによる低レベルの設定を参照している入力フィールドなどです。目的とするアプリケーションのモデルツリーに、その参照するオブジェクトが存在している場合には、その参照が使われます。
- 存在していないオブジェクトの参照は、貼り付けられる時に積極的に削除されます。例外として、ボタンのコマンドシーケンスに存在しない参照が設定されている場合、全てのコマンドは保持されていますが無効のマークが付けられます。
- コピーペースト操作に、ローカルメソッドも含まれています。しかし、メソッドのコードがアップデートされることはありません。これは、グローバルメソッドをコピーする場合にも当てはまります。
- ボタンまたはメニュー項目のコマンドシーケンスのコマンドの引数は、そのまま残されます。

- 全ての画像の参照は、適用可能な場合には、自動的にコピーされて画像ライブラリに追加されます。もし、同じ名前の画像がすでにあった場合には、コピーされた画像の代わりにそれが使われます。
- 参照されているファイル、サウンド、またはメソッドは、自動的にはコピーされません。しかし、メソッドはマニュアルでコピーペーストすることが可能です。
- 貼り付けるオブジェクトの名前が既存のオブジェクトの名前と重複してしまっている場合には、新しい名前が付けられます。他の貼り付けられたオブジェクトから新しく名前を付けられたオブジェクトへの参照もアップデートされます。

付録 C—ファイル処理とファイルスキーム表記法

ファイルの処理はアプリケーションの重要な機能です。例えば、実験データを入力した表計算ファイル、インポートされる CAD ファイル、または生成されエクスポートされるレポートが、アプリケーションに必要であるかもしれません。アプリケーションビルダーでは、ファイル全体またはファイルの一部を読み書きするツールを提供しています。これを行う方法は、アプリケーションが実行されているシステムによって異なります。そのファイルシステムは、アプリケーションを開発する COMSOL マルチフィジックスを動作させているコンピュータ、および COMSOL サーバーをインストールして起動後にアプリケーションが実行されるコンピュータに依存しています。

COMSOL サーバーでのファイル処理

一般に、ウェブブラウザーや COMSOL Client for Windows® によってアプリケーションを実行している最中に、ローカルディレクトリに対するファイルの読み書きをすることはできません。アプリケーションとそのメソッドはサーバー上で実行され、（ウェブブラウザーや COMSOL クライアントが走る）クライアントファイルシステムの知識は全く必要ではありません。

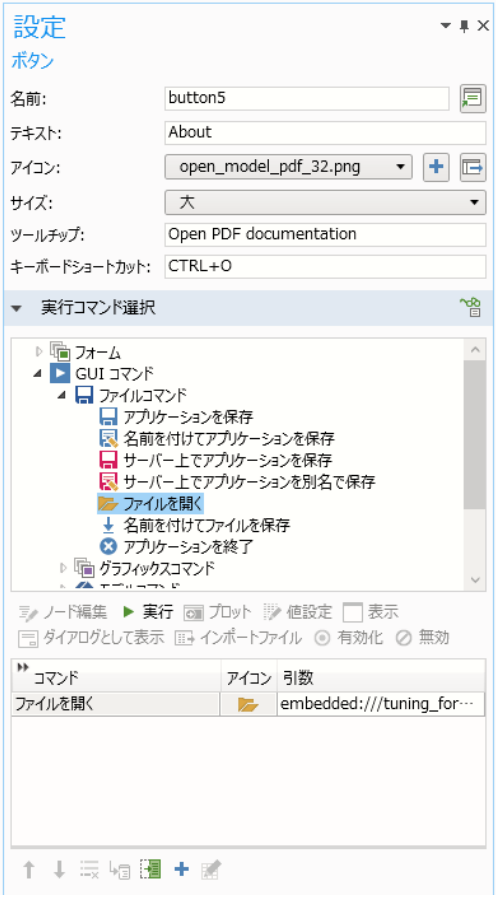
しかし、ウェブブラウザーと COMSOL クライアントによってアプリケーションが実行されている時に、クライアントファイルシステムとの間でファイルを移動させるためにはテクニックが必要です。

ユーザにファイルを要求するには、**ファイルインポート**オブジェクトを利用することができます。それによって、ユーザはクライアントファイルシステムのファイルをブラウズすることができます。この場合、クライアントファイルシステムは、COMSOL サーバーファイルシステムにアップロードされており、アプリケーションとそのメソッドへの利用が可能になっているとします。ユーザがアプリケーションの実行中に、例えば CAD ファイルまたは実験データファイルを提供して利用可能とするために使われます。これについては、284 ページの「ファイルインポート」の章で詳細説明されています。

例えば、ボタンのコマンドシーケンスによって、埋め込みモデルの**エクスポート**（Export）または**レポート**（Report）ノードのサブノードを実行させて生成されたデータをエクスポートすることができます。これについては、292 ページの「ファイルエクスポート」の章で詳細説明されています。

ファイルコマンドを使ったファイルの保存と開く

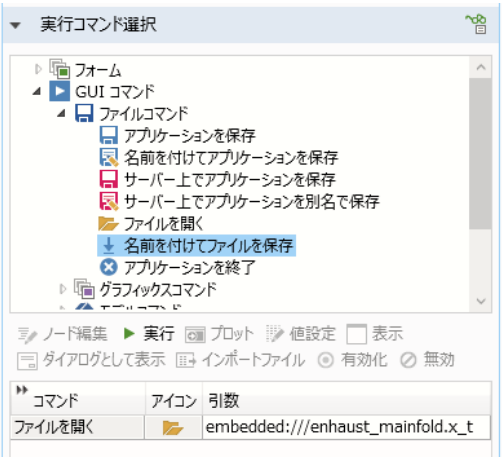
コマンドシーケンスで使われるエディターツリーにある**ファイルコマンド**フォルダには、アプリケーションの終了だけでなく、アプリケーションとファイルを保存したりロードするためのコマンドが含まれています。



ファイルを開くのコマンドでは、メソッドやモデルによって作られたりアプリケーションに組み込まれたファイルをサーバーから選定し、クライアント上の関連アプリケーションを使ってそのファイルを開きます。これは、例えば、クライアントファイルシステムにある PDF ファイルを開く、または、クライアント側のモデルからエクスポートされたテキストファイルまたは画像を表示するために使われます。上図では、アプリケーションで PDF ドキュメントを開くために、**ファイルを開く**のコマンドが使われています。対応する PDF ファイルは、**ライブラリ > ファイル**ノードに格納されることによってアプリケーションに埋め込まれています。そこに置かれたファイルは、次のセクションと 282 ページの「ファイルスキーム表記法」で説明しているファイルスキームシンタックスの embedded:/// を使って参照されます。

メソッドからファイルを開くには、組み込みメソッド fileOpen を使います。305 ページの「オペレーティングシステムメソッド」を参照してください。

ファイルを保存するには、**ファイルを開く**の場合と同じように、**名前を付けてファイルを保存**のコマンドを使います。これによって、サーバーファイルシステムからファイルを取得し、ユーザに**保存**ダイアログボックスを表示します。そこで、ユーザはファイルを保存するクライアントのロケーション位置にブラウズすることができます。ウェブブラウザー内のリンクからファイルをダウンロードする場合も、これと同様です。下図では、**名前を付けてファイルを保存**のコマンドが、**ライブラリ > ファイル**ノードに格納されている CAD モデルを保存するために使われています。



メソッドからファイルを保存する場合は、組み込みメソッド fileSaveAs を使います。311 ページの「GUI コマンドメソッド」を参照してください。ファイルの保存とエクスポートについての詳細は、292 ページの「ファイルエクスポート」を参照してください。

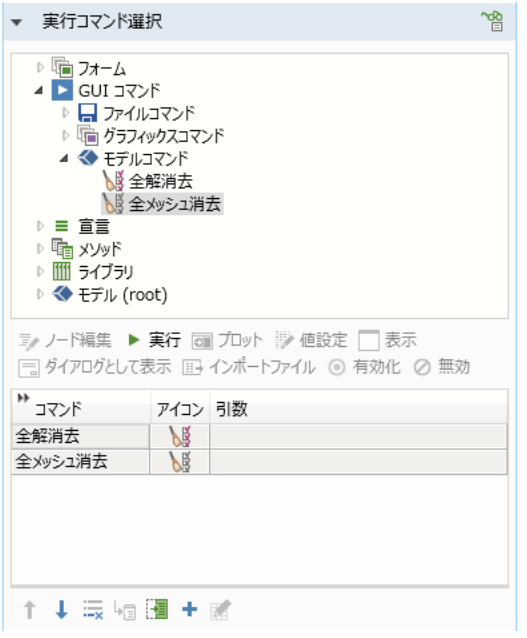
アプリケーションを保存と**名前を付けてアプリケーションを保存**のコマンドは、フォームオブジェクトのコマンドシーケンスで使うことが可能です。**名前を付けてアプリケーションを保存**のコマンドでは、**保存**ダイアログボックスが表示されます。そこで、ユーザはアプリケーション全体が保存されるクライアントパスを指定することができます。

同様に、**サーバー上でアプリケーションを保存**と**サーバー上でアプリケーションを別名で保存**のコマンドを使うことによって、サーバーファイルシステムにアプリケーション全体を保存することができます。これに関する組み込みメソッドについての情報は、311 ページの「GUI コマンドメソッド」を参照してください。

簡単に言えば、クライアントファイルシステムからのファイルのアップロードとダウンロードは、両方ともサポートされています。一方で、ユーザがファイルのソースや目的場所をブラウズすることなしに、アプリケーションの内部でそれが勝手に実行されることは決してありません。

モデルコマンド

コマンドシーケンスで使われているエディターツリーにある**モデルコマンド**フォルダには、**全解消去**と、**全メッシュ消去**の二つのコマンドが含まれています。これらのコマンドは、アプリケーションを保存する前に、解とメッシュのデータを消去して MPH ファイルサイズを小さくするために使われます。



ファイルスキーム表記法

アプリケーションをいつでも移動できるようにするために、アプリケーションビルダーでは、ファイルスキームによる仮想ファイルロケーションを使うことができるようになっています。ファイルスキームは、ファイルシステムのファイルのポインタとして考えることができますが、アプリケーションにおいて実際にファイルが格納されているのがどこなのかを知る必要はありません。

さまざまな目的のための種々のファイルスキームがあります。

- user ファイルスキームは、同じユーザによってアプリケーションを実行するたびに繰り返し使われるファイルに用います。
- common ファイルスキームは、基本的な動作は user と同じですが、全てのユーザ間で共有されるファイルに使います。
- temp ファイルスキームは、アプリケーションが閉じられた時にファイルが削除される用途で使われます。
- embedded ファイルスキームは、ファイルをアプリケーション自体に格納するために使われます。これは、アプリケーションを自己完結型にして、他の誰かに送りたい場合に有効です。

- 最後に、upload ファイルスキームは、ユーザが CAD ファイルをブラウズする場合のように、ユーザがアプリケーションを実行中にファイルをアップロードする用途で使われます。

下表では、全ての利用可能なファイルスキームの概要を示しています。

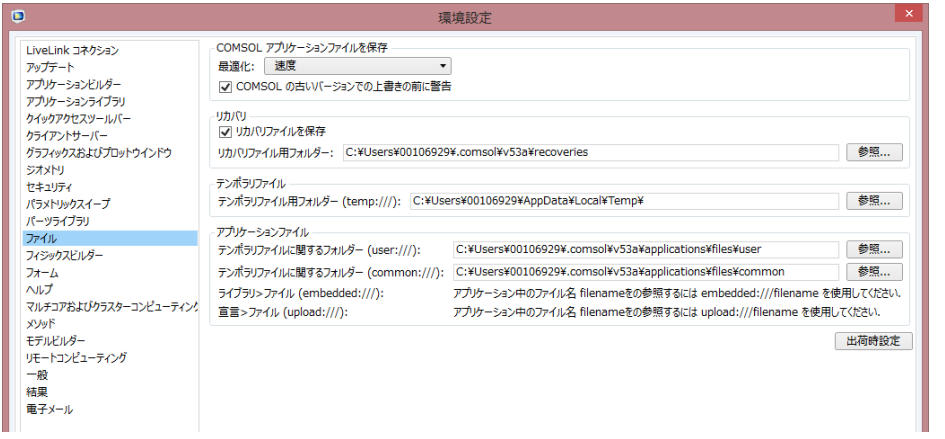
スキーム	参照先	デフォルトのパス	一般的な用途
embedded : / / /	ライブラリ > ファイルによってアプリケーションに組み込まれたファイル。	該当なし	実験データ、CAD ファイル、メッシュファイル、補間データ
upload : / / /	実行中にユーザがアップロードするファイル。	ファイル宣言の設定ウィンドウにある目的のディレクトリを指定	実験データ、CAD ファイル、メッシュファイル、補間データ
temp : / / /	アプリケーションを実行開始する度に固有である、テンポラリディレクトリにあるファイル。 これらのファイルは、アプリケーションが閉じられる時に消	環境設定 > ファイルの設定で指定される、テンポラリファイル用のフォルダへの任意のサブディレクトリ	コマンドシーケンスやメソッドによって生成されるテンポラリファイル、または（COMSOL サーバー利用者の）クライアントに保存される出力
user : / / /	現在のユーザの全てのアプリケーションに共有のディレクトリにあるファイル。	環境設定 > ファイルの設定で指定	セッション間に保存されるメソッドからの出力
common : / / /	全てのユーザに共有のディレクトリにあるファイル。	環境設定 > ファイルの設定で指定	多くのユーザやアプリケーション間で共有されるファイル

ライブラリノードにあるファイルは、embedded : / / / という表記法によってアクセス可能です。この詳細については、189 ページの「ライブラリ」を参照してください。

下表は、各ファイルスキームの用途を示しています。表中のチェックマークは、そのスキームが利用可能であることを意味しています。また、(r) は、それが推奨スキームであることを意味しています。

用途	EMBEDDED	UPLOAD	TEMP	USER	COMMON
入力用途のファイル	√ (r)	√			√
出力ファイル			√ (r)	√	
ファイルを読み込むメソッド	√ (r)	√	√	√	√
ファイルを書き込むメソッド			√ (r)	√	
クライアント側のファイル	√	√	√ (r)	√	√

ファイルメニューをアクセスして、下図に示すように、**環境設定**ダイアログボックスの**ファイル**ページで、テンポラリ(temp)、ユーザ(user)、コモン(common)ファイルのパスを設定することができます。

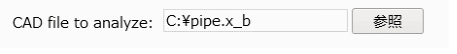


注） ユーザ(user) と コモン(common)ファイルに関するフォルダー設定箇所が、
いずれも「テンポラリファイルに関するフォルダー」と間違って記載されています。

ファイルインポート

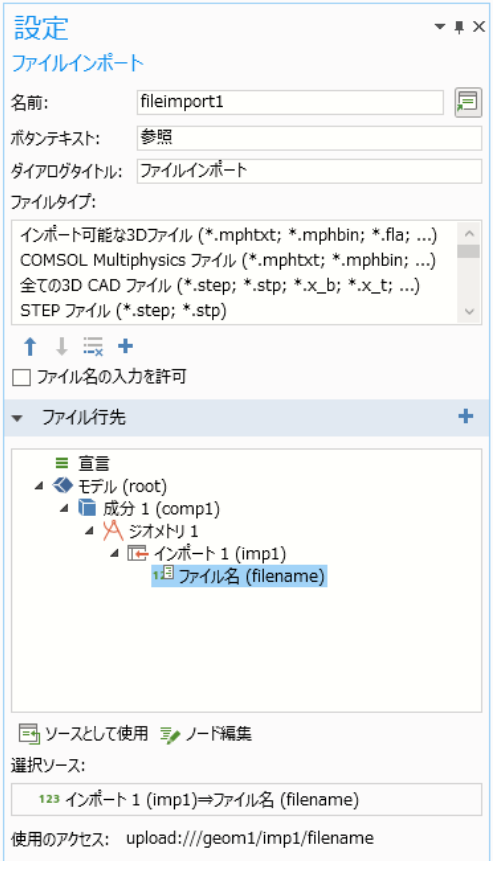
CAD インポートへのモデルツリーとファイルインポートオブジェクトの利用

ファイルインポートオブジェクトはファイルブラウザーを表示するために使われ、ファイルをブラウズしてパスと名前を指定するための入力フィールドを持っています。これを利用すれば、アプリケーションの実行前には入手できないファイルでも、ユーザによってアプリケーションの実行中にファイルインポートを利用可能にすることができます。モデルツリーにあるファイル**インポート**ノード、例えば CAD **インポート**ノードに、直接**ファイルインポート**オブジェクトをリンクさせることができます。下図に示すように、実行中に CAD ファイルを指定してインポートすることができるアプリケーションを考えてみましょう。



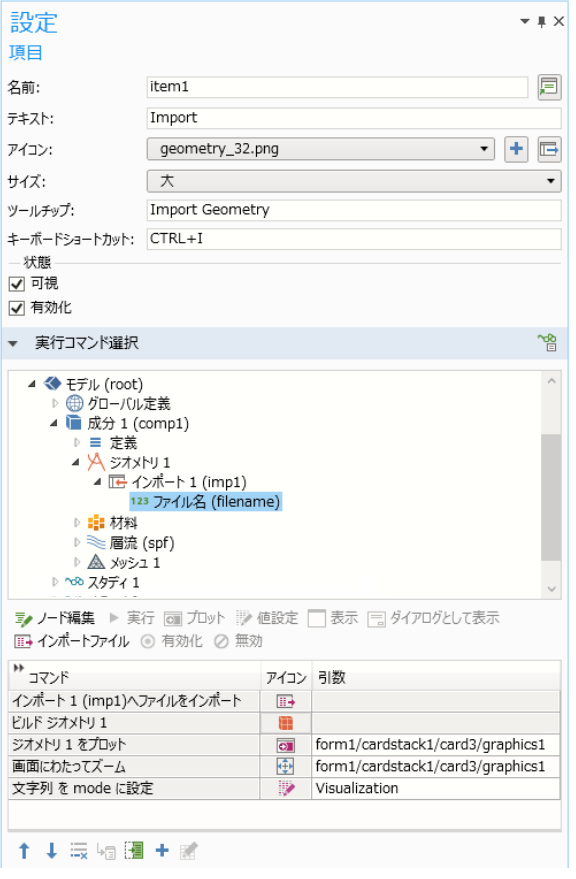
下図は、これに対応した**ファイルインポート**オブジェクトです。

ファイルインポートオブジェクトの**設定**ウィンドウには、**ファイル行先**セクションがあります。このセクションで、ファイル名の入力を許可するツリーノードを選択します。これが下図に示されており、**インポート**ノードにある**ファイル名**（filename）が選択されています。

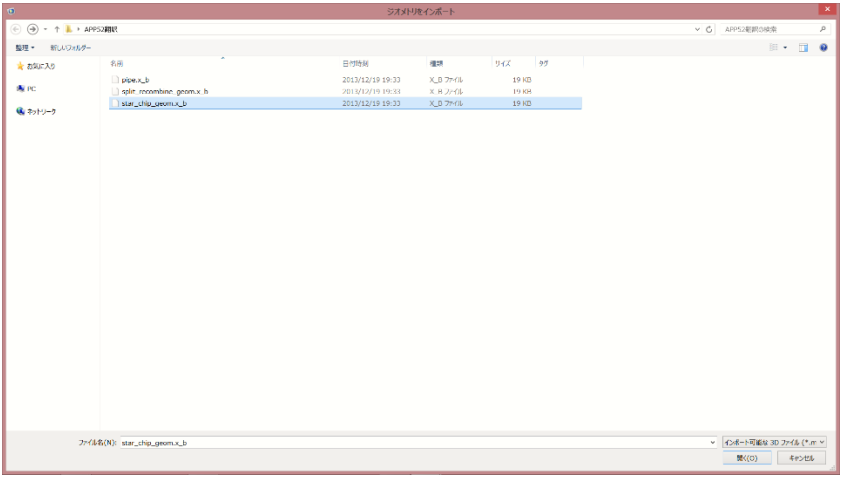


もし、**ファイルインポート**オブジェクトを使いたくないような場合には、メニュー、リボン、ツールバーの項目やボタンから**ファイル名**を直接参照することができます。

以下には、ジオメトリインポートのために使われるリボン項目の表示、およびその**設定**ウィンドウを示しています。



前図の**設定**ウィンドウに設定されている**インポート1（imp1）へファイルをインポート**のコマンドによって、ユーザがファイルを指定するためのファイルブラウザが開かれます。下図に、そのファイルブラウザーの例を示します。



それ以降のコマンドは、ジオメトリの作成とプロット、ズーム範囲を使ったズーム、最終的な文字列変数値の設定です（この例では、カードスタックのコントロールに使われています）。

ファイルインポートオブジェクトの詳細は、243 ページの「ファイルインポート」を参照してください。

メソッドでのファイルインポート

前章の例の続きとして、今、**設定**ウィンドウで**新規メソッドに変換**をクリックすると仮定してみます。これに対応した以下のコードは、CAD インポートがどのようにメソッドで記述されるかを示しています。

```
importFile ( model . geom ( "geom1" ) . feature ( "imp1" ) , "filename" ) ;
model . geom ( "geom1" ) . run ( ) ;
useGraphics ( model . geom ( "geom1" ) , "form1 / cardstack1 / card3 / graphics1" ) ;
zoomExtents ( "form1 / cardstack1 / card3 / graphics1" ) ;
mode = "Visualization" ;
```

最初のラインは、組み込みメソッド importFile の使われ方を表しています。メソッド importFile の詳細、およびファイル処理のためのこれ以外のメソッドについては、303 ページの「ファイルメソッド」を参照してください。

ファイルアクセスとファイル宣言

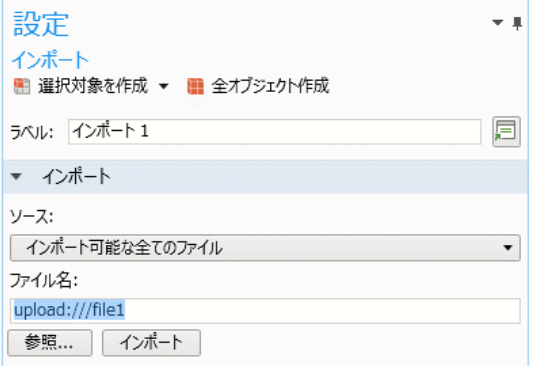
ファイルインポートオブジェクトの**設定**ウィンドウの下側で、インポートするファイルをメソッドでアクセスするために、どのようなファイルスキーム表記が使われるかを見ることができます（**使用のアクセス**：に続けて表示されています）。

下図は、**ファイル行先とファイル名**が使われている例を示しています。

ここでのファイルスキーム表記 upload : / / / geom1 / imp1 / filename は、このファイルにアクセスする時にはいつも使う必要があります。

その代わりの方法として、**宣言**ノードの下の**ファイル**宣言を使うことができます。
(しかし、**ファイル**宣言は、主に、メソッドのコードによるファイルインポートに使われます。)この場合には、ユーザが選択するファイルは、upload : / / / file1、upload : / / / file2 などの表記法によって、フォームオブジェクトまたはメソッドの中で参照されます。ファイル名ハンドル(file1、file2 など)は、実行中にユーザによって指定される実際のファイル名を参照するために使われます。137 ページの「ファイル」も参照してください。

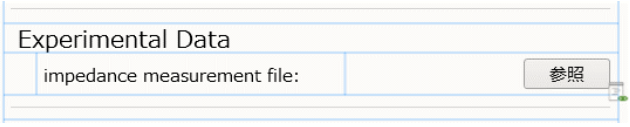
この表記は、モデルビルダーノードにおけるファイルブラウザーのテキストフィールドでも使われています。下図には、ジオメトリをインポートするモデルの**インポート**のモデルツリーノードにおいて、その設定ウィンドウの**ファイル名**フィールドで使われているファイル参照を示しています。



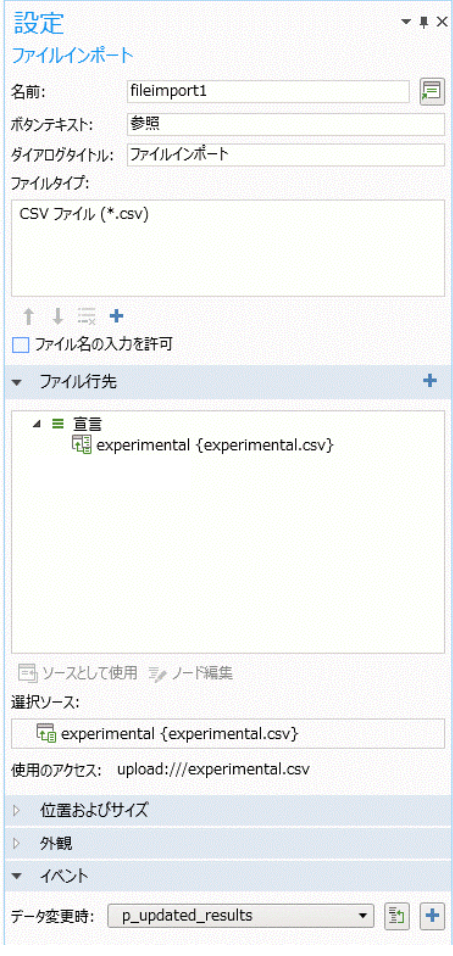
しかし、より迅速な方法は、以前に 284 ページの「CAD インポートへのモデルツリーとファイルインポートオブジェクトの利用」の章で説明されているように、ファイルインポートオブジェクトを**ファイル名**のフィールドに直接リンクさせることです。

実験データのインポート

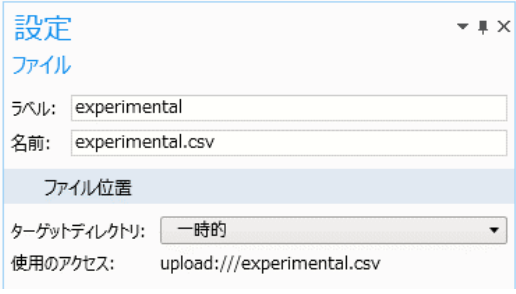
ユーザに実験データのファイルをアプリケーションの実行中に提供することを考えてみましょう。下図では、そのようなアプリケーションのファイルインポートオブジェクトがグリッドレイアウトモードで表示されています。



下図は、対応したファイルインポートオブジェクトの**設定**ウィンドウとファイル宣言へのリンクを示しています。



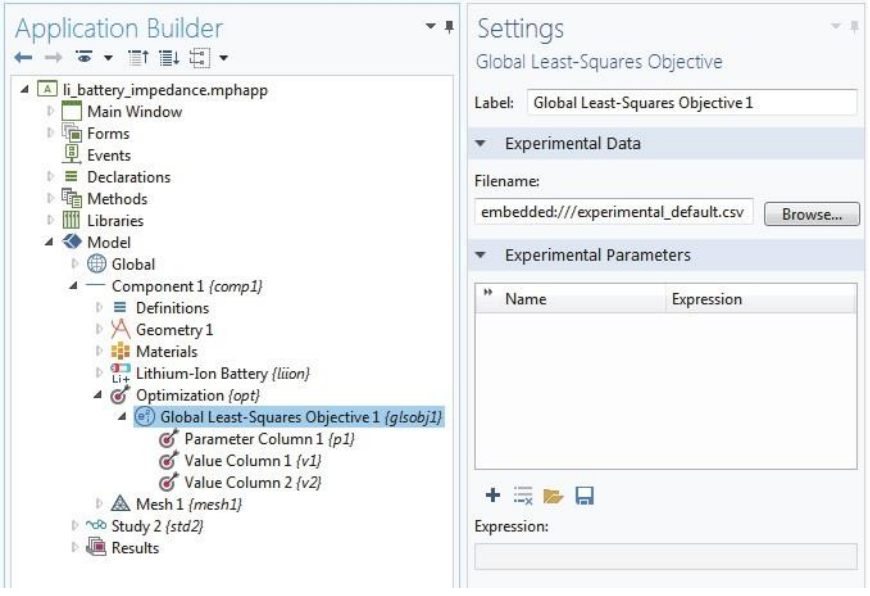
このアプリケーションでは、**ファイルタイプ**のテーブルは、CSV ファイルだけが許されるように指定されています。下図には、**ファイル**宣言の**設定**ウィンドウを示しています。



ファイル宣言では、ファイル upload : / / / experimental.csv に書かれているというように、インポートデータの「宛先」の役目を果たしています。

宣言で使われるファイル拡張子.csv は任意であり、また、アプリケーションの実行中にユーザによって指定されるファイルはどのような名前でも構わないことに注意してください。例えば、アプリケーションの実行中に指定されたファイル名が my_data.csv であったとしても、メソッドコードで参照される時には抽象的なファイルハンドル名 experimental.csv がいつも使われます。

最初に実験データを読み込む必要がなくてもアプリケーションを実行することができるようにするには、デフォルトの実験データを含んだファイルをアプリケーションに組み込んでおきます。このデフォルトのデータファイルは、次の図に示すように、embedded : / / / のファイルスキーム表記でのアクセスによってアプリケーションで使用されます。最適化モジュールを使ったこの例では、アプリケーションは実験データに適した最小二乗法を実行しています。



注）上図は、バージョン 5.0 での設定画面です。
バージョン 5.1 以降では、アプリケーションビルダーツリーにモデルツリーは展開されません。

以下のメソッドは、ユーザ提供された実験データファイルが存在するか、またはデフォルトデータ設定が使われるかを決定するためのロジックを処理しています。

```
if ( exists ( "upload : / / / experimental.csv" ) ) {
    with ( model . physics ( "opt" ) . feature ( "glsobj1" ) ) ;
        set ( "fileName" , "upload : / / / experimental.csv" ) ;
    endwith ( ) ;
}
else {
    String s_data = confirm ( "No experimental data file was uploaded. Do you
    want to use the embedded data? " , "Experimental Data" , "Yes", "Cancel
    Parameter Estimation" ) ;
    if ( s_data.equals ( "Cancel ParameterEstimation" ) ) {
        return ;
    }
}
```

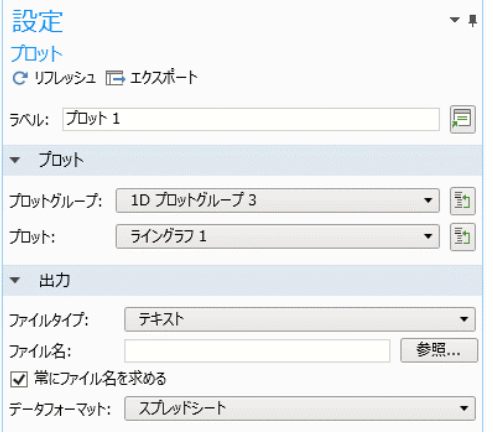
ここでは、ユーザ提供のファイルが存在する場合、embedded : / / / experimental_default.csv をフィジックスインタフェース glsobj1 の upload : / / / experimental.csv と置き換えています。

モデルツリーを使ったファイルエクスポート

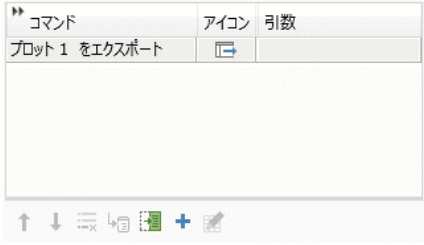
例えば、ボタンのコマンドシーケンスにおいて、埋め込みモデルの**エクスポート**または**レポート**ノードのサブノードを実行することによって、そこで生成されたデータをエクスポートすることができます。モデルツリーの**エクスポート**ノードでは、ファイルをエクスポートするためのサブノードとして、以下のタイプを選択することができます。

- **データ**
- **プロット**
- **メッシュ**
- **テーブル**
- **3D 画像**
- **2D 画像**
- **1D 画像**
- **アニメーション**

これらのノードの各**設定**ウィンドウには、その**出力**セクションに**ファイル名**を指定するフィールドがあります。下図は、**エクスポート** > **プロット**ノードを選択した場合のプロットの**設定**ウィンドウを示しています。

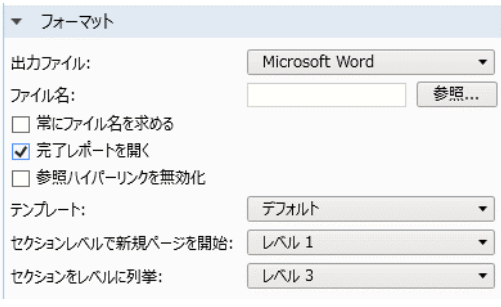


上図に示したように、**ファイル名**フィールドはブランクのままにしておくことができます。例えば、次の図に示すように、ボタンのコマンドシーケンスにおいて、そのモデルの**エクスポート** > **プロット**ノードをランタイムに実行することができ、その時にファイルの格納場所とファイル名を指定するためのファイルブラウザーウィンドウが開かれます。



アプリケーションの開発中は、繰り返しモデルビルダーを使用してエクスポートデータをチェックする必要があるかもしれません。このような場合には、**ファイル名**フィールドを使ってテストファイルを指定すると同時に、**常にファイル名を求める**のチェックボックスにチェックを入れておきます。そうすれば、ファイルブラウザーをランタイムに開かせることができます。

エクスポートのサブノードと同じように、各**レポート**のサブノードには、下図に示すように、その**フォーマット**セクションに**ファイル名**を指定するフィールドがあります。



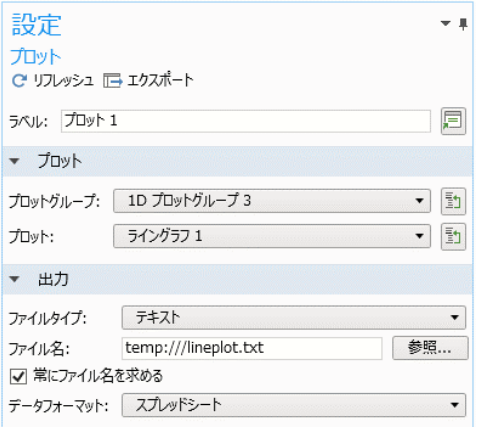
レポートのサブノードを実行することによって、レポートの格納場所とファイル名を指定するためのファイルブラウザーウィンドウが開かれます。

代わりに、ファイルスキームを使うことによって、ファイルのインポートとエクスポートのさらにきめ細かい制御が可能です。

テンポラリファイルへのファイルエクスポート

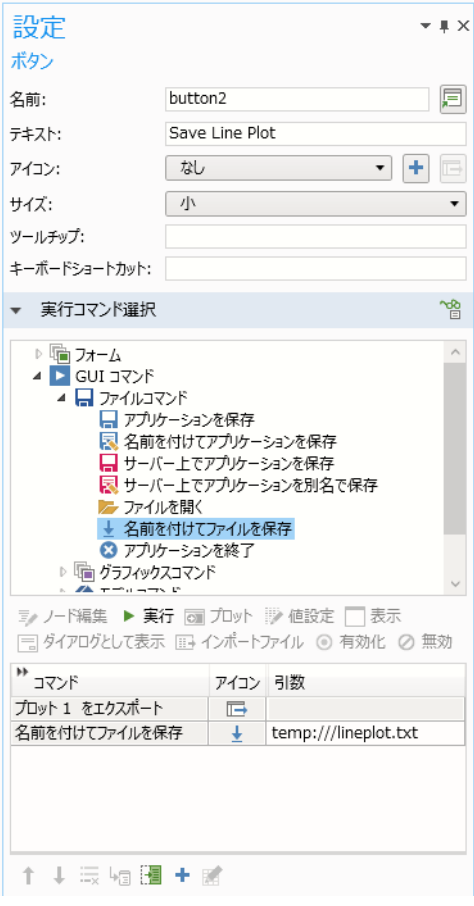
あるアプリケーションでは、テンポラリファイルを作る必要があるかもしれません。そのような場合は、ファイルスキーム temp : / / / を使って行うことができます。テンポラリファイルは任意の一時的なディレクトリに格納され、その格納場所はアプリケーションを実行開始する度に変わります。これらのファイルは、アプリケーションが閉じられる時には削除されます。テンポラリファイルは、コマンドシーケンスやメソッドによって生成することができ、また、COMSOL サーバーでクライアントに保存する際にも出力することができます。

下図には、数値としてのプロットデータをエクスポートする場合の**エクスポート > プロット**ノードの**設定**ウィンドウの例を示しています。

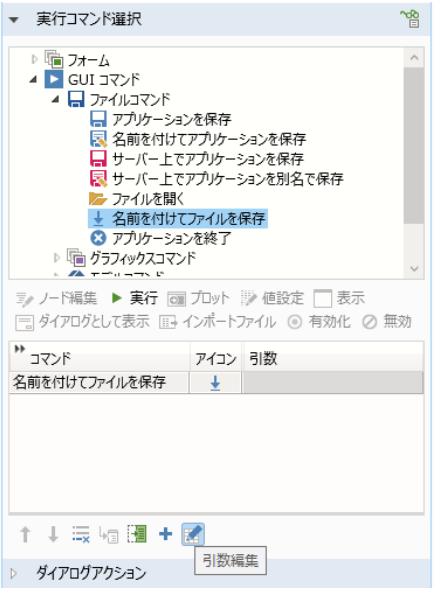


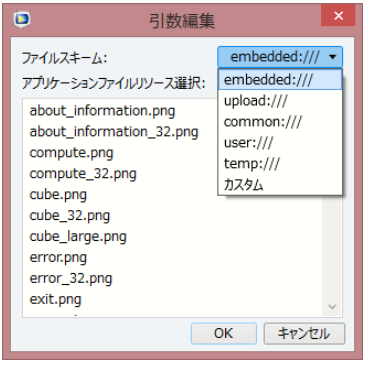
この**出力**セクションの**ファイル名**の設定は、temp : / / / lineplot.txt となっています。

この例では、プロットをディスクに保存するためのボタンが作成されています。そのボタンの**設定**ウィンドウの**実行コマンド選択**では、最初に、あらかじめ作られた**エクスポート > プロット**ノードを選んで**実行**をクリックすることによって、出力するグラフファイルを作成するように設定しています。次に、**GUI コマンド > ファイルコマンド > 名前を付けてファイルを保存**を選んで再び**実行**をクリックしています。**プロットをエクスポート**のコマンドによって作られるテンポラリファイルの名前が、このボタン**設定**の**出力**セクションで設定しているファイル名です。この例では、temp : / / / lineplot.txt が設定されています。


設定
ボタン
名前:
button2
テキスト:
Save Line Plot
アイコン:
なし
サイズ:
小
ツールチップ:
キーボードショートカット:
実行コマンド選択
フォーム
GUI コマンド
ファイルコマンド
アプリケーションを保存
名前を付けてアプリケーションを保存
サーバー上でアプリケーションを保存
サーバー上でアプリケーションを別名で保存
ファイルを開く
名前を付けてファイルを保存
アプリケーションを終了
グラフィックスコマンド
ノード編集
実行
プロット
値設定
表示
ダイアログとして表示
インポートファイル
有効化
無効
コマンド
アイコン
引数
プロット 1 をエクスポート
名前を付けてファイルを保存
temp:///lineplot.txt

名前を付けてファイルを保存のコマンドには引数があり、**引数編集**のボタンによって専用のダイアログボックスが表示されます。これによって、全てのファイルスキームのショートカットだけでなく、全ての組み込みファイルに容易にアクセスすることができます。

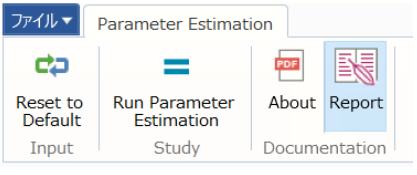




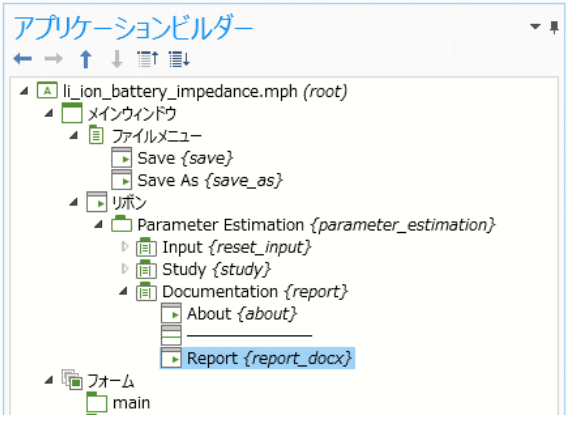
低レベルの機能によるレポートの作成

この章では、低レベルの機能のみによって簡単にレポートを作成する方法について説明しています。メソッドで直接作成する場合については、292 ページの「ファイルエクスポート」を参照してください。

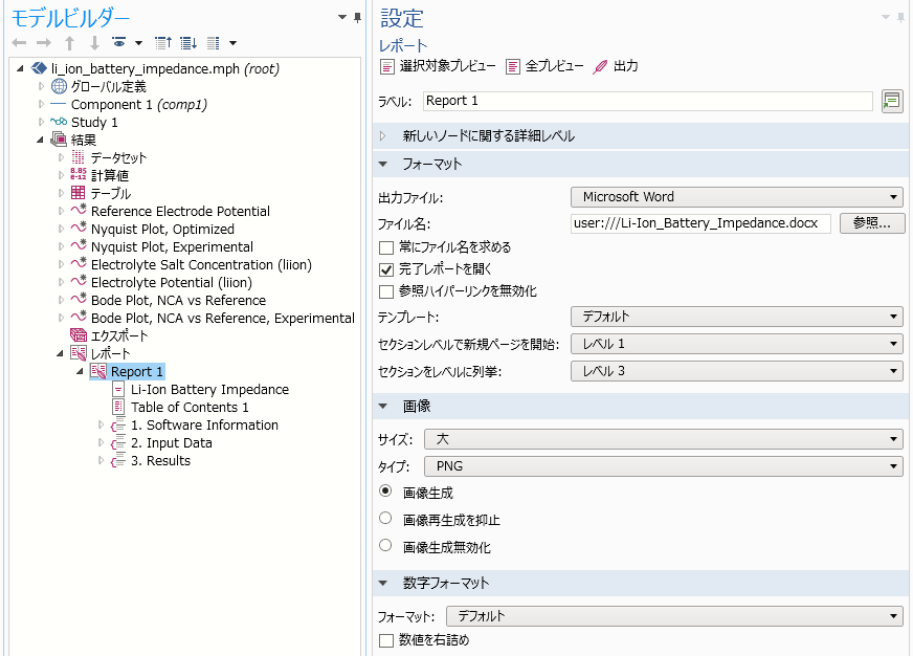
以下の例は、Microsoft® Word® 形式(.docx) のレポートをユーザが保存できるアプリケーションです。下図は、そのアプリケーションのリボンのタブを示しています。このタブの **Documentation** のセクションに **Report** のボタンが作られています。



それに関するアプリケーションツリーのノードを、下図に示しています。



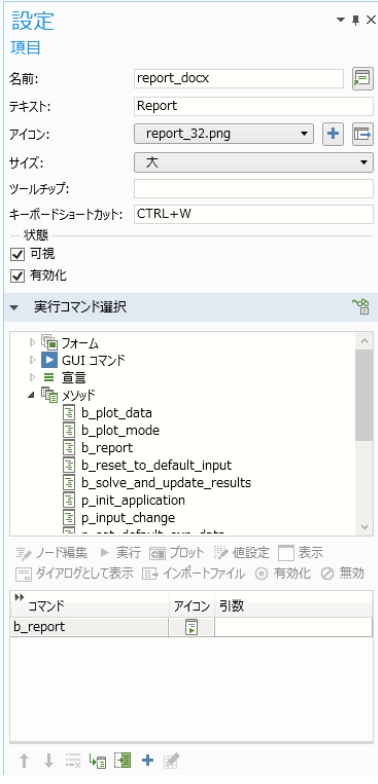
下図では、モデルビルダーの**レポート**(Report)ノードの**設定**ウィンドウにおける**ファイル名**のフィールドで、user : / / file の表記法が使われていることを示しています。



このアプリケーションでは、**完了レポートを開く**のチェックボックスがチェックされているため、レポートが作成された後に Word® 文書が開かれます。その後、アプリケーションのユーザは、そのレポートを Word® ファイルのメニューから保存することができます。

この例では、同様に、ファイルスキーム common : / / / も使うことができます。同じファイルを繰り返しアプリケーションで使うとき、user と common ファイルスキームを優先的に利用します。

下図は、**レポート**のリボン項目の**設定**ウィンドウを示しています。



メソッド b_report は、以下のコードを含んでいます。

```
if  ( length ( information_card ) > 0 ) {
     alert ( "New input data. Compute to update resultsfirst." ) ;
}
else  {
     model . result ( ) . report ( "rpt1" ) . run ( ) ;
}
```

ファイルスキーム表記法を、メソッドで直接使うこともできます。以下のコードは、HTML レポートをエクスポートするためのメソッドの部分です。

```
String  answerh =  request ( "Enter file name" , "File Name" , "Untitled . html" ) ;
if  ( answerh != null ) {
   model . result ( ) . report ( "rpt1" ) . set ( "format" , "html" ) ;
   model . result ( ) . report ( "rpt1" ) . set ( "filename" , "user : / / /" + answerh ) ;
   model . result ( ) . report ( "rpt1" ) . run ( ) ;   }
```

付録 Dーキーボードショートカット

以下の表は、アプリケーションビルダーで利用可能なキーボードショートカットのリストです。

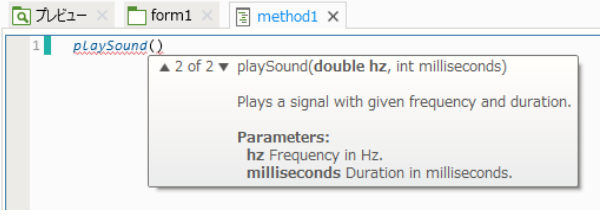
ショートカット	動作	アプリケーションビルダ	フォームエディター	メソッドエディター
Ctrl+A	全て選択	√	√	√
Ctrl+D	全て選択解除		√	
Ctrl+C	コピー	√	√	√
Ctrl+V	ペースト貼り付け		√	√
Del	削除	√	√	√
Ctrl+N	新規アプリケーションの作成	√	√	√
Ctrl+S	アプリケーションの保存	√	√	√
Ctrl+F8	アプリケーションのテスト	√	√	√
Alt+click	フォームオブジェクトの編集		√	
Ctrl+Pause	メソッドの停止	√		
Ctrl+Shift+F8	変更の適用	√	√	√
Ctrl+R	コードをレコード			√
F11	ノードへ			√
Ctrl+K	ショートカットの作成			√
F1	ヘルプの表示	√	√	√
F2	アプリケーションノードの名前の変更	√		
F3	アプリケーションノードを無効化	√		
F4	アプリケーションノードを有効化	√		
Ctrl+up arrow	アプリケーションノードを上に移動	√		
Ctrl+down arrow	アプリケーションノードを下に移動	√		
Ctrl+Z	取り消し	√	√	√
Ctrl+Y	やり直し（Mac では Control+Shift+Z）	√	√	√
F5	継続（デバッグ時）			√
F6	ステップ（デバッグ時）			√
F7	ステップへ（デバッグ時）			√

ショートカット	動作	アプリケーションビルダ	フォームエディター	メソッドエディター
F8	文法のチェック			√
Ctrl+F	メソッドのテキストを検索と置換			√
Ctrl+Space, Ctrl+/, or Ctrl+OEM2	メソッドのコードの自動コンプリート			√
Ctrl+U	選択されたコードを小文字に変換			√
Ctrl+Shift+U	選択されたコードを大文字に変換			√
Ctrl+B	選択行のブレークポイントを切り替え			√
Ctrl+M	一対の丸括弧、角括弧、または中括弧を切り替え			√
Ctrl+Shift+M	一対の丸括弧、角括弧、または中括弧の間の全ての文字を選択			√
Ctrl+scroll wheel up	メソッドウィンドウのコードをズームイン			√
Ctrl+scroll wheel down	メソッドウィンドウのコードをズームアウト			√
Ctrl+all arrow keys	選択されたフォームオブジェクトの配置を微調整		√	
All arrow keys	選択されたフォームオブジェクトの配置を微調整		√	
Ctrl+Alt+A	アプリケーションビルダーウィンドウへ		√	√
Ctrl+Alt+M	モデルビルダーウィンドウへ	√	√	
Ctrl+Alt+left-Click	ローカルメソッドを作成するか、フォームオブジェクトに関連付けられたメソッドを開く		√	
Ctrl+Alt+ Double-Click	メソッドエディターコードからメソッドを開く			√
Alt+F4	ウィンドウを閉じる	√	√	√
Ctrl+F4	ドキュメントを閉じる		√	√
Ctrl+Shift+F4	全てのドキュメントを閉じる		√	√

ショートカット	動作	アプリケーションビルダ	フォームエディター	メソッドエディター
Ctrl+7	コメントのオン/オフを切り替え			√
Press Ctrl and left-click. キーとボタンを押しながらマウスをドラッグ。	フォームオブジェクトをコピー		√	

付録 E—組み込みメソッドライブラリ

この付録は、モデルオブジェクトとアプリケーションオブジェクトで操作するメソッド以外の、メソッドエディターから利用可能な全ての組み込みメソッドをリストにしています。組み込みメソッドの利用と使用する全ての構文についての詳細情報は、*Application Programming Guide* と *Programming Reference Manual* を参照してください。
これらのメソッドの構文を学習する別の方法として、メソッドの名前を入力してから Ctrl + Space キーを操作することによってコード補完を利用することができます。ウィンドウには、構文とメソッドシグネチャに関する情報が表示されます。



モデルユーティリティメソッド

モデルユーティリティメソッドによって、メソッドに MPH ファイルのモデルオブジェクトの部分をロードして処理を追加することができます。

名前	説明
clearModel	モデルオブジェクトのコンテンツをクリアする。
createModel	指定されたタグを持つ新規モデルを作成する。
removeModel	モデルを削除する。組み込みモデルを削除することはできない。
modelTags	組み込みモデルを含む、全てのロードされたモデルのモデルタグの配列を返す。
uniqueModeltag	使用されていないモデルタグを返す。
getModel	指定されたタグのモデルを返す。
loadModel	ファイルから、指定されたタグを持つモデルをロードする。
loadProtectedModel	ファイルから、指定されたタグを持ちパスワードで保護されたモデルをロードする。
loadRecoveryModel	リカバリディレクトリ/フォルダ構造からモデルをロードする。
saveModel	モデルをファイルに保存する。ファイル名は、ファイルスキームパスを指定可能。また、セキュリティ設定で許可されている場合には、サーバーのファイルパスも指定可能。
getComsolVersion	現在のソフトウェアのバージョンを文字列で返す。

ファイルメソッド

名前	説明
readFile	指定されたファイルのコンテンツを文字列として返す。
openFileStreamReader	指定されたファイル name から、行単位、または文字単位で読み取るために使用可能な CsReader オブジェクトを返す。
openBinaryFileStreamReader	指定されたファイルのバイトから、バイト単位で読み取るために使用可能な CsBinaryReader オブジェクトを返す。
readMatrixFromFile	指定されたファイルの内容を倍精度の行列に読み込む。そのファイルは、モデルツリーのエクスポートノードで利用可能なようなスプレッドシートタイプの形式。
readStringMatrixFromFile	指定されたファイルの内容を文字列の行列に読み込む。そのファイルは、モデルツリーのエクスポートノードで利用可能なようなスプレッドシートタイプの形式。
readCSVFile	カンマで区切られた値を持つファイル（CSV ファイル）を、文字列の行列に読み込む。CSV 用の RFC4180 形式を使用するファイルを想定。
writeFile	指定されたファイルに配列 data を書き込む。スプレッドシート形式が使われており、そのデータは readMatrixFromFile または readStringMatrixFromFile によって読み取り可能。
openFileStreamWriter	指定されたファイルに書き込み可能な CsWriter オブジェクトを返す。
openBinaryFileStreamWriter	バイト単位で指定されたファイルのバイトへの書き込みに使用可能な CsBinaryWriter オブジェクトを返す。
writeCSVFile	指定された倍精度または文字列の配列 data を CSV ファイルに書き込む。CSV には RFC4180 形式が使用される。
exists	指定した名前のファイルが存在するかどうかをテストする。
deleteFile	指定された名前のファイルが存在する場合、それを削除する。ファイルはサーバー上で削除される。
copyFile	サーバー上のファイルをコピーする。ソースとターゲットの両方の名前は、ファイルスキームのパスを使用可能。
importFile	ファイルブラウザーのダイアログボックスを表示し、指定された name のファイル宣言に選択したファイルをアップロードする。 代わの方法としては、指定されたモデルオブジェクトのエンティティにあるファイル名のテキストフィールドに選択したファイルをアップロードする。

名前	説明
writeExcelFile	指定された文字列の配列 data を、Excel ファイルの指定のシートの指定のセルから書き込む。
readExcelFile	Excel ファイルの指定のシートを、指定のセルから 2D 文字列配列に読み込む。
getFilePath	あるファイルスキームのパスに対応するサーバーのプロキシファイルの絶対サーバーファイルパスを返す。指定されたパスのサーバープロキシファイルがない場合は、null を返す。 このメソッドは、外部コードまたはアプリケーションにファイルへのスキーム（ 例えば temp : / / / ）を使用してパスを渡すために利用可能。
getClientFileName	クライアントファイルシステム上でアップロードされたファイルの元の名前を返す。（指定したファイルスキームのパスと一致するアップロードファイルが存在しない場合は null を返す。） このメソッドは、ユーザインタフェースからのフィードバックのためにのみ有用。例えば、使用されているアップロードファイルに関する情報を取得する。元のファイルがクライアント上にまだ存在する、あるいは現在のクライアントが元のクライアントと同じであるといった保証はない。
getClientFilePath	クライアントファイルシステム上でアップロードされたファイルの元のパスを返す。（指定したファイルスキームのパスと一致するアップロードファイルが存在しない場合は null を返す。） このメソッドは、ユーザインタフェースからのフィードバックのためにのみ有用。例えば、使用されているアップロードファイルに関する情報を取得する。元のファイルがクライアント上にまだ存在する、あるいは現在のクライアントが元のクライアントと同じであるといった保証はない。

オペレーティングシステムメソッド

名前	説明
executeOSCommand	指定されたコマンド（フルパス）とパラメータで OS コマンドを実行する。該当する場合、コマンドはサーバー側で実行される。
fileOpen	クライアント上の関連するプログラムでファイルを開く。「ファイルメソッド」の章も参照のこと。
getUser	アプリケーションを実行しているユーザのユーザ名を返す。アプリケーションが COMSOL サーバーから実行されていない場合、環境設定 一般 > ユーザ名 > 名前の設定値が返される。
openURL	クライアントのデフォルトブラウザーで URL を開く。
playSound	クライアントでサウンドを再生する。

電子メールメソッド

名前	説明
emailFromAddress	COMSOL サーバーまたは環境設定から、アドレスからの電子メールを返す。（送信元アドレス）
sendEmail	指定された件名、本文テキスト、および、ファイルがあれば添付して、デフォルトの受信者に電子メールを送信する。添付ファイルは、埋め込みモデルのレポート、エクスポート、およびテーブルノードから作成。
userEmailAddress	ユーザが電子メールアドレスを設定されていない場合、現在ログインしているユーザに設定された電子メールアドレス、または空の文字列を返す。

電子メールのクラスメソッド

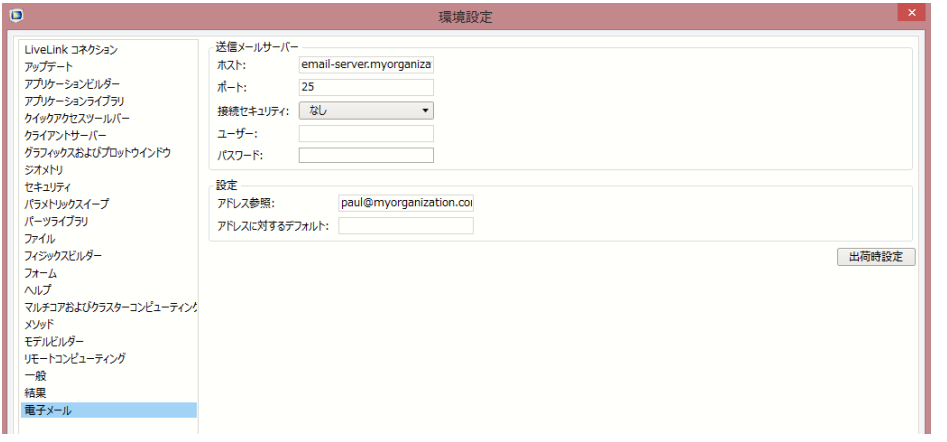
クラス EmailMessage は、カスタマイズした電子メールメッセージを作成するために利用できます。

名前	説明
EmailMessage	新しい EmailMessage オブジェクトを作成する。
EmailMessage.set Server	この電子メールメッセージに使用する電子メール（SMTP）サーバーのホストとポートを設定する。
EmailMessage.set User	ユーザ名と電子メール（SMTP）サーバーの認証に使用するパスワードを設定する。このメソッドは、setServer メソッドの後に呼び出さなければならない。
EmailMessage.set Security	電子メール（SMTP）サーバー通信の接続セキュリティのタイプを設定する。
EmailMessage. setFrom	送信元アドレスを設定する。

名前	説明
EmailMessage. setTo	宛先アドレスを設定する。
EmailMessage. setCc	cc アドレスを設定する。
EmailMessage. setBcc	bcc アドレスを設定する。
EmailMessage. setSubject	電子メールの件名を設定する。改行文字は許可されていないことに注意。
EmailMessage. setBodyText	プレーンテキスト（文字列形式）で電子メールの本文を設定する。電子メールは、テキストと HTML の本文の両方を含めることができる。
EmailMessage. setBodyHtml	HTML テキストで電子メールの本文を設定する。電子メールは、テキストと HTML の本文の両方を含めることができる。
EmailMessage. attachFile	添付ファイルを追加する。添付の MIME タイプは、ファイル名の拡張子によって決定される。
EmailMessage. attachFile	指定された MIME タイプの添付ファイルを追加する。
EmailMessage. attachFromModel	モデルのレポート、エクスポート、またはテーブルの機能で作成された添付ファイルを追加する。
EmailMessage. attachText	プレーンテキスト（文字列形式）または HTML のように、指定されたサブ MIME タイプの添付テキストを追加する。
EmailMessage. attachBinary	指定された MIME タイプのバイト配列の添付ファイルを追加する。
EmailMessage. send	電子メール（SMTP）サーバーに電子メールを送信する。電子メールオブジェクトは一度だけ送信することができる。

電子メールの環境設定

電子メール（SMTP）サーバーに対する環境設定を行うには、下図に示すように、**環境設定**ダイアログボックスの**電子メール**ページを開きます。



COMSOL サーバーでは、この電子メールの環境設定と同様の設定が可能になっています。

GUI 関連のメソッド

名前	説明
Call a method directly	例えば、method1( )、method2( )といったその名前を使って、メソッドリストからメソッドを呼び出す。
callMethod	メソッドリストからメソッドを呼び出すための代替方法。内部的に使われたり、名前が重なる場合に使われる。
useGraphics	第 2 引数に名前または名前のパスで指定されるグラフィックスフォームオブジェクトで、指定されたエンティティ（プロットグループ、ジオメトリ、メッシュまたは明示的選択）をプロットする。
useForm	現在のメインウィンドウ内の指定された名前のフォームを表示する。フォームオブジェクトの use メソッドと同じ。下記参照。
closeDialog	ダイアログボックスとして表示される、指定された名前のフォームを閉じる。
dialog	指定された名前のフォームを、ダイアログボックスとして表示する。フォームオブジェクトの dialog メソッドと同じ。下記参照。
alert	実行を停止し、指定されたテキストの警告メッセージを表示する。
alert	実行を停止し、指定されたテキストとタイトルの警告メッセージを表示する。
confirm	実行を停止し、指定されたテキストとタイトルの確認ダイアログボックスを表示する。そこには、「Yes」、「No」、または「Cancel」といった二つまたは三つのボタンも表示される。
error	実行を停止し、指定されたメッセージを持つエラーダイアログボックスを開く。
request	実行を停止し、ユーザからの入力を要求するテキストフィールドを持つダイアログボックスを表示する。
message	メッセージログにメッセージを送信する。
evaluateToResultsTable	第 2 引数に名前または名前のパスによって指定されたテーブルオブジェクトにおいて、指定されたエンティティ、計算値（Derived Value）を評価する。この第 2 引数は、そこで計算値（Derived Value）を評価するためのデフォルト対象である。第 3 引数が true の場合、そのテーブルは新しいデータを追加する前にクリアされる。そうでない場合には、データが追加される。

名前	説明
evaluateToDoubleArray2D	指定されたエンティティと計算値（Derived Value）を評価し、倍精度行列として作成された実数型テーブルのパラメータではない列部分を返す。その全ての数値設定が評価されるが、現在のテーブルに数値の紐付け設定が成り立っている場合は無視される。
evaluateToIntegerArray2D	指定されたエンティティと計算値（Derived Value）を評価し、整数行列として作成された実数型テーブルのパラメータではない列部分を返す。その全ての数値設定が評価されるが、現在のテーブルに数値の紐付け設定が成り立っている場合は無視される。
evaluateToStringArray2D	指定されたエンティティと計算値（Derived Value）を評価し、文字列行列として作成された実数型テーブルのパラメータではない列部分を返す。その全ての数値設定が評価されるが、現在のテーブルに数値の紐付け設定が成り立っている場合は無視される。
useResultsTable	**結果テーブル**のフォームオブジェクトの tableFeature の値を表示する。
getChoiceList	宣言ノードの下の選択リストノードである ChoiceList 型オブジェクトを返す。ChoiceList 型には、その値や表示名を容易に変更することができる関連メソッドがある。 *Application Programming Guide* 参照。
setFormObjectEnabled	name または名前のパスで指定されたフォームオブジェクトを有効状態に設定する。
setFormObjectVisible	name または名前のパスで指定されたフォームオブジェクトを可視状態に設定する。
setFormObjectText	第 2 引数に name または名前のパスで指定されたフォームオブジェクトのテキストを設定する。指定されたフォームオブジェクトのテキストを設定することができない場合、このメソッドはエラーを throw する。
setFormObjectEditable	name または名前のパスで指定されたフォームオブジェクトを編集可能な状態に設定する。この機能は、テキストフィールドオブジェクトにのみ使用可能です。
setMenuBarItemEnabled	最初の引数で、name または（メニューバーからの）名前のパスで指定されたメニューバー項目を有効状態に設定する。
setMainToolbarItemEnabled	最初の引数で、name または（メインツールバーからの）名前のパスで指定されたメインツールバー項目を有効状態に設定する。

名前	説明
setFileMenuItemEnabled	最初の引数で、name または（ファイルメニューからの）名前のパスで指定されたファイルメニュー項目を有効状態に設定する。
setRibbonItemEnabled	最初の引数で、name または（メインウィンドウからの）名前のパスで指定されたリボン項目を有効状態に設定する。
setToolbarItemEnabled	最初の引数で、name または名前のパスで指定されたツールバーフォームオブジェクト項目を有効状態に設定する。
useView	第 2 引数に name または名前のパスで指定されたグラフィックスコンテンツにビュー（view）を適用する。
resetView	第 2 引数に name または名前のパスで指定されたグラフィックスコンテンツのビュー（view）をリセットして初期状態にする。
getView	第 2 引数に name または名前のパスで指定されたグラフィックスコンテンツで、現在使用されているビュー（view）を返す。
setWebPageSource	最初の引数で、name または名前のパスで指定されたフォームオブジェクトのソースを設定する。その名前がウェブページのフォームオブジェクトを参照していない場合、このメソッドはエラーを throw する。
getScreenHeight	クライアントシステムの主要な画面、またはウェブクライアントが使用されている場合はブラウザーウィンドウの高さをピクセル単位で返す。
getScreenWidth	クライアントシステムの主要な画面、またはウェブクライアントが使用されている場合はブラウザーウィンドウの幅をピクセル単位で返す。

GUI コマンドメソッド

名前	説明
clearAllMeshes	全てのメッシュをクリアする。
clearAllSolutions	全ての解をクリアする。
clearSelection	指定されたグラフィックスオブジェクトの選択をクリアする。
exit	アプリケーションを終了する。
fileOpen	クライアントに関連付けられたプログラムでファイルを開く。
fileSaveAs	クライアントにファイルをダウンロードする。「ファイルメソッド」の章も参照。
printGraphics	指定されたグラフィックスオブジェクトをプリントする。
saveApplication	アプリケーションを保存する。
saveApplicationAs	別の名前で(または MPH ファイルとして)アプリケーションを保存する。
scenelight	指定されたグラフィックスオブジェクトのシーンライトを切り替える。
selectAll	指定されたグラフィックスオブジェクトの全てのオブジェクトを選択する。
transparency	指定されたグラフィックスオブジェクトの透過度を切り替える。
zoomExtents	指定されたグラフィックオブジェクトを画面にわたってズームする。

デバッグメソッド

名前	説明
debugLog	デバッグログウィンドウに入力引数の値を表示する。入力引数には、文字列、倍精度、整数、ブーリアンのタイプのスカラー、1D 配列、2D 配列をとることができる。

外部 C ライブラリメソッド

外部メソッド

名前	説明
external	ライブラリ機能の名前によって指定される外部 C(ネイティブ)ライブラリへのインタフェースを返す。外部クラスは、Java Native Interface (JNI)のフレームワークを使う。 詳細は *Application Programming Guide* を参照。

外部メソッドによって返されるメソッド

外部メソッドは、以下のメソッドによって外部タイプのオブジェクトを返します。

名前	説明
invoke	指定された引数を持つライブラリ内の一つの名前付きネイティブメソッドを呼び出す。
invokeWideString	指定された引数を持つライブラリ内の名前付きネイティブメソッドを呼び出す。
close	ライブラリとフリーのリソースを解放する。このメソッドを呼び出さないと、外部ライブラリが不要になったときに自動的には呼び出されない。

進捗メソッド

名前	説明
setProgressInterval	トップの進捗レベルの進捗表示間隔を設定し、そのレベルの message を表示する。 このメソッドを呼び出すと、setProgress( )の呼び出しによって事前に設定されたいくつかのマニュアル進捗が暗黙のうちにリセットされる。
setProgress	ユーザ制御の進捗レベルの値を設定する。
resetProgress	全ての進捗レベルを削除し、進捗を 0 にリセットし、メッセージを空の文字列にする。
showIndeterminate Progress	message とキャンセルボタンが与えられた不確定進捗バーを持つ進捗ダイアログボックスを表示する。
showProgress	キャンセルボタンと選択可能なモデルの進捗状況、および一つまたは二つの進捗情報を持つ進捗ダイアログボックスを表示する。
closeProgress	現在表示中の進行ダイアログボックスを閉じる。
startProgress	指定された進捗バーフォームオブジェクトの name の値を 0 にリセットする。
setProgressBar	指定された進捗バーフォームオブジェクトの name の値を 0～100 の範囲で設定し、関連する進捗メッセージも設定する。

日付と時刻のメソッド

名前	説明
currentDate	(サーバーのデフォルト設定に応じてフォーマットされた)文字列としての現在の日付を返す。
currentTime	(日付は含まず、サーバーのデフォルト設定に応じてフォーマットされた)文字列として現在の時刻を返す。
formattedTime	指定された形式を使ってフォーマットされた時刻を返す。フォーマットとしては、時間単位、または、より長いフォーマットを記述したテキストが可能。
sleep	ミリ秒単位で指定された時間のスリープ。
timeStamp	世界協定時(UTC)1970 年 1 月 1 日、午前 0 時 0 分 0 秒からのミリ秒単位の現在時刻。
getExpectedCom putationTime	アプリケーションのおおよその計算時間を記述した文字列を返す。文字列は、メソッド setExpectedComputationTime によって変更することができる。

名前	説明
setLastComputationTime	自動的に生成された時間を上書きし、前回の計算時間を設定する。 時間差を記録するためにメソッド timeStamp を使うことができ、その後にミリ秒で測定された時間（長整数型）を設定する。
getLastComputationTime	指定されたフォーマットで、前回の計算時間を設定する。指定された形式を使ってフォーマットされた時刻を返す。フォーマットとしては、時間単位、または、より長いフォーマットを記述したテキストが可能。このフォーマットは現地語化され、現在の言語設定に変換されて出力される。

ライセンスメソッド

名前	説明
checkoutLicense	指定された各製品のライセンスを一つずつチェックアウトする。
checkoutLicenseForFile	MPH ファイルを開くために必要な各製品のライセンスを一つずつチェックアウトする。
checkoutLicenseForFileOnServer	MPH ファイルを開くために必要な各製品のライセンスを一つずつチェックアウトする。
getLicenseNumber	現在のセッションのライセンス番号を持つ文字列を返す。 例：licensenumber=getLicenseNumber()
hasProduct	指定された製品を実行するために必要なソフトウェアコンポーネントが COMSOL インストールに含まれている場合、true を返す。
hasProductForFile	指定された MPH ファイルの実行に必要なソフトウェアコンポーネントが COMSOL インストールに含まれている場合、true を返す。
hasProductForFileOnServer	指定された MPH ファイルの実行に必要なソフトウェアコンポーネントが COMSOL インストールに含まれている場合、true を返す。

変換メソッド

名前	説明
toBoolean	文字列と文字列配列をブーリアンに変換する（'true'には true を戻し、全ての他の文字列は false を返す）。
toDouble	文字列と文字列配列を倍精度に変換する。
toInt	文字列と文字列配列を整数に変換する。
toString	ブーリアン、整数、および倍精度を、配列の場合も含めて文字列に変換する。

配列メソッド

名前	説明
getColumn	2D 配列（行列）における指定された列の文字列、倍精度、整数、またはブーリアン配列を返す。例えば、値がファイルから読まれて、ある列だけをテーブルに表示するような場合に利用する。
getSubMatrix	入力行列の矩形部分行列を返す。文字列、倍精度、整数、またはブーリアン 2D 配列での利用可能。

名前	説明
insert	配列に一つ以上の要素を挿入し、その拡張された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
append	配列の最後に一つ以上の要素を追加し、その拡張された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
remove	配列から一つ以上の要素を削除し、その短縮された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
insertRow	2D の矩形配列に一つ以上の行を挿入し、その拡張された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
appendRow	2D の矩形配列の最後に一つ以上の行を追加し、その拡張された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
removeRow	2D の配列から一つ以上の行を削除し、その縮小された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
replaceRow	2D の矩形配列の一つ以上の行を置き換え、配列を返す。 文字列、倍精度、整数、またはブーリアン配列で利用可能。
insertColumn	2D の矩形配列に一つ以上の列を追加し、その拡張された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
appendColumn	2D の矩形配列の最後に一つ以上の列を追加し、その拡張された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
removeColumn	2D の矩形配列から一つ以上の列を削除し、その縮小された配列を返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。
replaceColumn	2D の矩形配列の一つ以上の列を置き換え、配列を返す。 文字列、倍精度、整数、またはブーリアン配列で利用可能。
matrixSize	行列の行数と列数を長さ 2 の整数配列として返す。文字列、倍精度、整数、またはブーリアン配列での利用可能。

文字列メソッド

名前	説明
concat	セパレーター(separator)で指定された文字を使って、単一文字列に、指定された文字列の配列または行列を連結する。
contains	指定された文字列配列に指定された文字列が含まれている場合に、true を返す。
find	文字列配列内からある文字列が出現する全てのインデックスを持つ配列を返す。
findIn	文字列配列内からある文字列が最初に出現するインデックス、または文字列内の従属文字列の最初に出現したインデックスを返す。
length	文字列の文字長さを返す。
replace	文字列が別の文字列に置き換えられた文字列を返す。
split	指定された文字列を指定されたセパレーター(separator)で分割した文字列の配列を返す。
substring	指定された位置から始まる指定された文字長さの従属文字列を返す。
unique	指定された文字列の配列内の重複値のない文字列配列を返す。

コレクションメソッド

名前	説明
copy	指定された配列、または行列のコピーを返す。文字列、倍精度、整数、またはブーリアン配列が利用可能。
equals	指定された配列内の全ての要素が等しく、同じ要素数である場合は true を返す。文字列、倍精度、整数、またはブーリアン配列が利用可能。倍精度の場合、相対誤差が許容範囲内であるかどうかによって比較される。
sort	指定された配列をソートする。 注: 配列は所定の位置にソートされる。文字列、倍精度、整数配列が利用可能。配列が 2 次元(行列)の場合、列の値は上から下にソートされる。
merge	指定された配列がマージされ、その全ての要素を持った配列を返す。文字列、倍精度、整数配列が利用可能。

With、Get、および Set メソッド

名前	説明
with	コードをよりコンパクトにするために使用する。
endwith	with ステートメントの終わり。

名前	説明
set	ブーリアン、整数、倍精度、または文字列のプロパティ値を設定する。スカラー、配列、または行列プロパティが許される。
setIndex	指定された index での行列またはベクトルの文字列、倍精度、または整数のプロパティ値を設定する。
getIntArray	整数ベクトルプロパティを取得する。
getIntMatrix	整数行列プロパティを取得する。
getBoolean	ブーリアンプロパティを取得する。
getBooleanArray	ブーリアンベクトルプロパティを取得する。
getBooleanMatrix	ブーリアン行列プロパティを取得する。
getDouble	倍精度プロパティを取得する。
getString	文字列のスカラー、ベクトル、または行列のプロパティを取得する。
getDoubleArray	倍精度ベクトルプロパティ、またはパラメータを取得する。
getDoubleMatrix	倍精度行列プロパティ、またはパラメータを取得する。
getStringArray	文字列ベクトルプロパティ、またはパラメータを取得する。
getStringMatrix	文字列行列プロパティ、またはパラメータを取得する。
getDblStringArray	文字列の行列としての値を返す。
getInt	整数プロパティを取得する。
get	変数の式を返す。
descr	変数の記述を返す。

付録 F—アプリケーション構築のためのガイドライン

一般的なヒント

- 入力データとそれに対応した出力データを持つファイルに、レポート機能を追加しましょう。
- 直感的に作りましょう。ヘルプ、ヒント、およびドキュメントを必要に応じて提供してください。
- フールプルーフ設計をしましょう:「安全な I/ O」、「データをデフォルトにリセット」、など。
- モデルと共にサムネイル画像を保存してください。
- 説明テキストを含めることができます(それは COMSOL サーバーライブラリで見ることができます)。
- 意図しているコンピュータプラットフォーム上でアプリケーションをテストしましょう。
- 最小化を心掛けましょう。開発者の視点からは、論理的作業、構造化、デバッグ、メンテナンス、さらにはアプリケーションの開発の確認が非常に容易です。ユーザの視点からは、アプリケーションを簡単に使用することができます。この最小化の取り組みには、後のメンテナンスをさらに少なくし、多くのユーザに採用してもらえるようにするといった以外にも、開発中にはより多くの注意が必要です。
- 扱いやすいサイズのライブラリなら、モデルに組み込んで利用することができます。
- 予想計算時間と、計算後には実際の計算時間を表示しましょう。
- 計算がキャンセルされると、前の計算による出力データは消去されるように作りましょう。
- 必要に応じてパスワード保護を施せます。(注意:パスワードを忘れてしまった場合の手段はありません。)

メソッド

- 必要以上のメソッドを作らないようにしましょう。

  少ないメソッドなら短いリストなので、何かを探している時にザッと目を通すことができます。少ないメソッドとは、通常、コードの行数が心配するほど少ないということです。

  - もし、記述したいくつかのメソッドが本質的に同じことをする場合は、それらを一つのメソッドにマージし、入力引数によってさまざまなケースを扱うことを検討してみましょう。
  - もし、メソッドを単にある場所から呼び出すことができるなら、そこではそのメソッドを作成する必要はありません。代わりに、その場所に呼び出しのコードを正しく挿入しておきます。

- フォームにおいて、フォームのイベントまたはフォームオブジェクトのイベントによって起動するだけのために使われるメソッドの場合には、ローカルメソッドを作成しましょう。
- メソッドに説明的な名前を付けて、アルファベット順にソートした時に同様のメソッドが一緒にグループ化されるようにします。これによって、名前を覚えておく必要が少なくて済み、探しているメソッドを容易に見つけることができます。短い名前は理解しづらいので、長い名前とする方がよいでしょう。
- メソッドに説明的な名前を付けて、アルファベット順にソートした時に同様のメソッドが一緒にグループ化されるようにします。これによって、名前を覚えておく必要が少なくて済み、探しているメソッドを容易に見つけることができます。短い名前は理解しづらいので、長い名前とする方がよいでしょう。

メソッドの名前付けの例:

- 出力処理のないメソッドは、全て“p”(procedure の頭文字)で始める。
- 出力処理をするメソッドは、全て“f”(function 機能の頭文字)で始める。
- メニュー項目のメソッド出力は、全て“m”(menu メニューの頭文字)で始める。
- 頻繁に利用するメソッドは、リストの中で最初に表示されるように“a”で始める。
- プロットのメソッドは、全て“Plot”で始める。(例えば、メニュー項目のメソッドなら、mPlotMesh、mPlotResults。)

- 上記に示した要点は、メソッドのコードにも同様に適用すべきです。できるだけ少ないコード行と変数で最小化し、変数には説明的な名前を付け、短い名前は理解しづらいので長い名前とし、効率的に実行されるようにコードを最適化します。
- 上記に示した要点は、宣言にも同様に適用すべきです。考慮した名前を付け、必要以上にならないように(フォームやメソッド、またはモデルで)使われる変数を宣言します。

フォーム

- 必要以上のフォームを作らないようにしましょう。
- フォームには、説明的な名前を付けます。これは、メソッドの場合と同じ理由からです。
- 多くのさまざまなタイプのフォームオブジェクトを有効利用しましょう。フォームオブジェクトごとに、その最適な利用の仕方があります。
- 必要以上のフォームオブジェクトを挿入しないようにしましょう。入力データのためにあまりにも多くの選択肢を設けると、アプリケーションが使いづらくなります。あまりにも多くの出力データは、重要な情報を見つけづらくします。
- アプリケーションを保存する時に、入出力データの利用者の設定で保存するコメントを残すために、ユーザ用のテキストフィールドを挿入しましょう。
- データをデフォルトにリセットするメソッドを持ったボタンを挿入することを検討しましょう。
- 「安全な I/ O」を適用しましょう。
  - 入力フィールドが範囲外の入力データの場合には、ユーザに警告しましょう。入力フィールドの**データ変更時**のイベントで起動された警告によって、またはフォームオブジェクトの設定ウィンドウで制限を設定(その後、ハードリミットを設定)することによって、その警告をすることができます。警告時にユーザが先に進める選択が可能なように、警告を生成しているメソッドで入力データを設定することができます。
  - 出力フィールドでは、適切な精度で出力を表示させることができます。現在の結果が現在の入力データに基づいていない場合は、それを表示しましょう。計算が失敗した場合は、それを表示しましょう。

- ツールチップ、ヘルプ、ドキュメント、ヒント、および総合的なレポートを含めるようにしましょう。
- 一般的なコンピュータ上で、シミュレーションを実行するのにかかる時間についてのユーザ情報を、デフォルトの入力データを使用して提供することができます。それは、アプリケーションによっては何秒、何時間、何日かもしれず、ユーザが計算ボタンを押す前に知っておきたい情報です。計算が終了した時にユーザに警告するように、サウンドを再生することも考えてみましょう。ユーザは、結果を待っている間、別のことをしていても構いません。(計算が本当に長い場合、計算の最中にユーザや他の場所にレポートを添付した電子メールを送るのは、待っている間の作業として良いかもしれません。)
- フォームの配置にいくらかでも時間をかけましょう。見栄えの良いフォームによって、使い易く楽しいアプリケーションになります。
- ボタンとメニュー項目のキーボードショートカットの設定を考慮しましょう。

付録 G—アプリケーションライブラリ例

アプリケーションライブラリでは、アプリケーションビルダーの機能を紹介するアプリケーションのサンプルを見つけることができます。それらは、Applications という名前のフォルダに集められており、多くのアドオン製品に利用することができます。これらのアプリケーションを編集し、独自のアプリケーションを設計する場合の出発点、あるいはアイデアの創造に利用することができます。各アプリケーションには、そのアプリケーションを説明しているドキュメント（PDF）とレポートを作成するための選択手段が含まれています。
以下は、アプリケーションライブラリツリーに表示されている利用可能なアプリケーション例の一部のリストです。

名前	アプリケーションライブラリ
Beam Subjected to Traveling Load	COMSOL マルチフィジックス
Helical Static Mixer	COMSOL マルチフィジックス
Transmission LineCalculator	COMSOL マルチフィジックス
Tubular Reactor	COMSOL マルチフィジックス
Tuning Fork	COMSOL マルチフィジックス
Effective Nonlinear Magnetic Curves	AC/DC モジュール
Induction Heating of a Billet	AC/DC モジュール
Magnetic Prospecting	AC/DC モジュール
Touch Screen Simulator	AC/DC モジュール
Absorptive Muffler Designer	音響モジュール
Acoustic Reflection Analyzer	音響モジュール
One-Family House Analyzer	音響モジュール
Organ Pipe Design	音響モジュール, パイプ流れモジュール [12]
Small Concert Hall Analyzer	音響モジュール
Cyclic Voltammetry	電気化学モジュール, 電気めっきモジュール, バッテリ &燃料電池モジュール, 腐食解析モジュール
Electrical Impedance Spectroscopy	電気化学モジュール, 電気めっきモジュール, バッテリ &燃料電池モジュール, 腐食解析モジュール
Li-Ion Battery Impedance	バッテリ &燃料電池モジュール [1]
Inkjet	CFD モジュール, マイクロフルイディクスモジュール
NACA Airfoil Optimization	CFD モジュール [8]
Water Treatment Basin	CFD モジュール

名前	アプリケーションライブラリ
Biosensor Design	化学反応工学モジュール
Liquid Chromatography	化学反応工学モジュール
Membrane Dialysis	化学反応工学モジュール
Ship Hull ICCP	腐食解析モジュール
Frame Fatigue Life	疲労解析モジュール [11]
Parameterized Concrete Beam	ジオメカニクスモジュール [11]
Concentric Tube Heat Exchanger	伝熱モジュール [2]
Heat Sink with Fins	伝熱モジュール
Equivalent Properties of Periodic	伝熱モジュール
Flash Method	伝熱モジュール
Forced Air Cooling with Heat Sink	伝熱モジュール
Inline Induction Heater	伝熱モジュール [9]
Parasol and Solar Irradiation	伝熱モジュール
Thermoelectric Cooler	伝熱モジュール
MEMS Pressure Sensor Swelling	MEMS モジュール, 構造力学モジュール
Microresistor Beam	MEMS モジュール
Red Blood Cell Separation	マイクロフルイディクスモジュール, 粒子追跡モジュール [3]
Mixer	ミキサーモジュール [4]
Charge Exchange Cell Simulator	分子流モジュール, 粒子追跡モジュール [10]
Ion Implanter Evaluator	分子流モジュール
Centrifugal Governor	マルチボディダイナミクスモジュール [11]
Truck Mounted Crane Analyzer	マルチボディダイナミクスモジュール [5, 11]
Stress Analysis of a Pressure Vessel	非線形構造材料モジュール [11]
Laminar Static Particle Mixer Designer	粒子追跡モジュール
Geothermal Heat Pump	パイプ流れモジュール
CCP Rector	プラズマモジュール
Distributed Bragg Reflector Filter	光線光学モジュール
Corrugated Circular Horn Antenna	RF モジュール
Frequency Selective Surface Simulator	RF モジュール
Microstrip Patch Antenna Array	RF モジュール
Plasmonic Wire Grating	RF モジュール, 波動光学モジュール
Wavelength Tunable LED	半導体モジュール

名前	アプリケーションライブラリ
Beam Section Calculator	構造力学モジュール [6]
Bike Frame Analyzer	構造力学モジュール [13]
Interference Fit	構造力学モジュール
Truss Bridge Designer	構造力学モジュール
Truss Tower Buckling	構造力学モジュール
Viscoelastic Structural Damper	波動光学モジュール
Fiber Simulator	波動光学モジュール
Plasmonic Wire Grating	RF モジュール, 波動光学モジュール [7]

1 バッテリー＆燃料電池モジュールと最適化モジュールが必要です。

2 CFD モジュール、伝熱モジュール、マイクロフルイディクスモジュール、またはプラズマモジュールのいずれかで実行されます。

3 このアプリケーションのマイクロフルイディクスモジュール版では、マイクロフルイディクスモジュールと粒子追跡モジュールが必要です。粒子追跡モジュール版では、粒子追跡モジュールと CFD モジュール、マイクロフルイディクスモジュール、または地下水流モジュールのいずれかが必要です。

4 CFD モジュールとミキサーモジュールが必要です。

5 構造力学モジュールとマルチボディダイナミクスモジュールが必要です。

6 LiveLink™ *for* Excel® 製品を使ったこのアプリケーションの拡張版も利用可能です。

7 このアプリケーションの RF モジュール版では、RF モジュールが必要です。波動光学モジュール版では、波動光学モジュールが必要です。

8 CFD モジュールと最適化モジュールが必要です。

9 伝熱モジュールと AC/DC モジュールが必要です。

10 分子流モジュールと粒子追跡モジュールが必要です。

11 構造力学モジュールも必要です。

12 音響モジュールとパイプ流れモジュールが必要です。

13 このアプリケーションでは、フルパラメトリック機能のために LiveLink™ *for* SOLIDWORKS®が必要です。

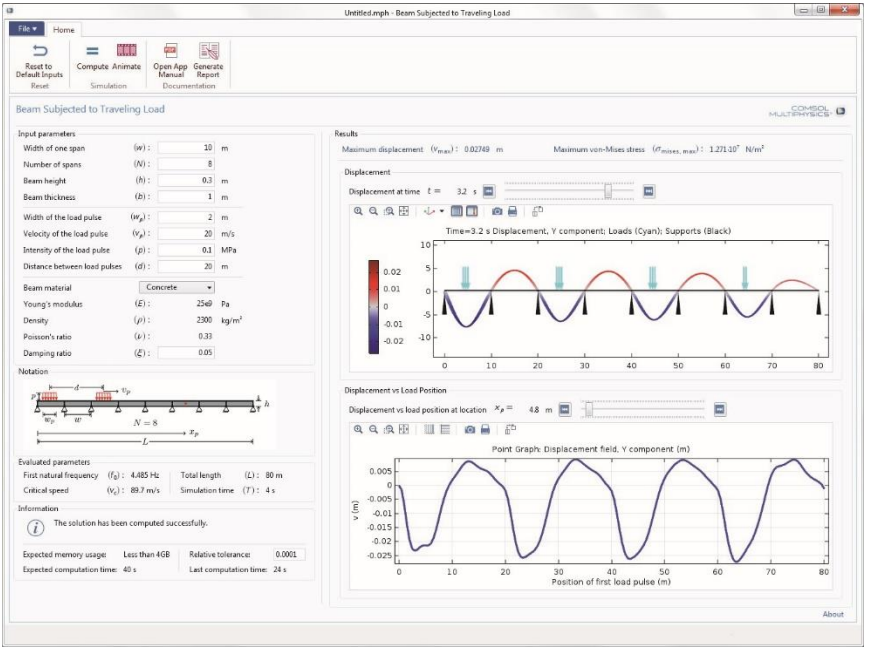
以下の章では、上表に掲載されたアプリケーション例の概要を示しています。ここで掲載しているアプリケーションは重要なアプリケーションビルダー機能のさまざまな例を示しており、そこにはアニメーション、電子メール、最適化、パラメータ評価、テーブル、および実験データのインポートの使用方法が含まれています。

ビーム走行荷重 （ Beam Subjected to Traveling Load ）

このアプリケーションは、いくつかの等間隔の支持体上に配置され、走行荷重を受けるビームの過渡応答を解析しています。アプリケーションの目的は、車両が通過するときの橋の応答を分析することです。与えられたジオメトリと材料特性を持つブリッジによって、特定の車両速度がブリッジ共振を引き起こし、高振幅振動を受けることを観察することができます。アプリケーションでは、ビームのモデル化において

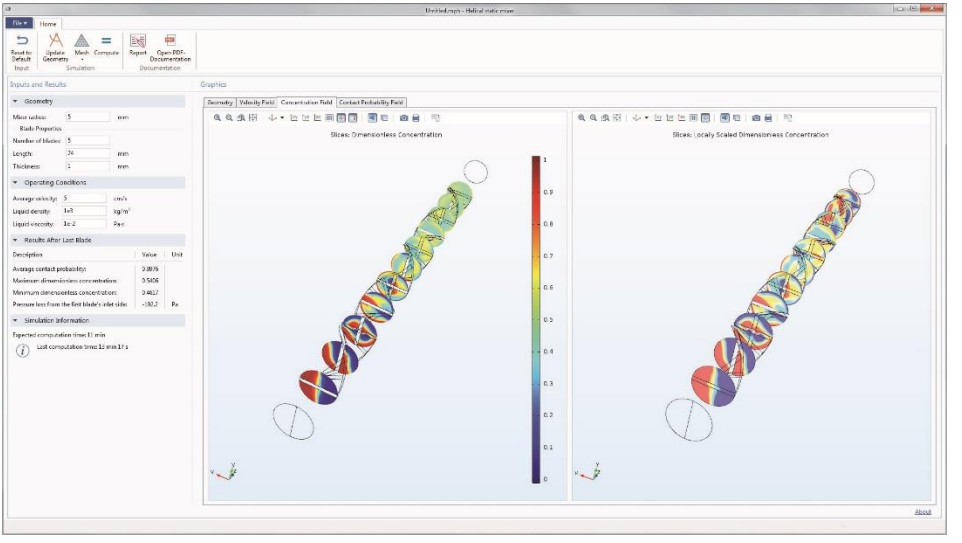
2 次元平面応力近似が仮定されています。ビームは、コンクリートで作られています。

このアプリケーションでは、アニメーションやスライダーの使用方法が示されています。第 1 のスライダーは負荷の位置に対する変形を示し、第 2 のスライダーは変位の時間変化を示しています。このアプリケーションは、アドオン製品を必要としません。



ヘリカルスタティックミキサー（*Helical Static Mixer*）

このアプリケーションの目的は、ジオメトリの部品やパラメータ化ジオメトリの使用を実証することです。さらに、アプリケーションは、一般には、重合反応中にモノマー（単量体）とイニシエータ（開始剤）を混合するために、1～5 個のヘリカルブレードを含むシステムにおける混合の度合いを推定することができます。 アプリケーションは、ニュートン流体に制限されている。それは、ミキサー自体の重合がごくわずかである場合に良く近似しています。このアプリケーションでは、**タイル**や**タブ**のタイプのフォームコレクションの利用方法を示してくれています。このアプリケーションは、アドオン製品を必要としません。



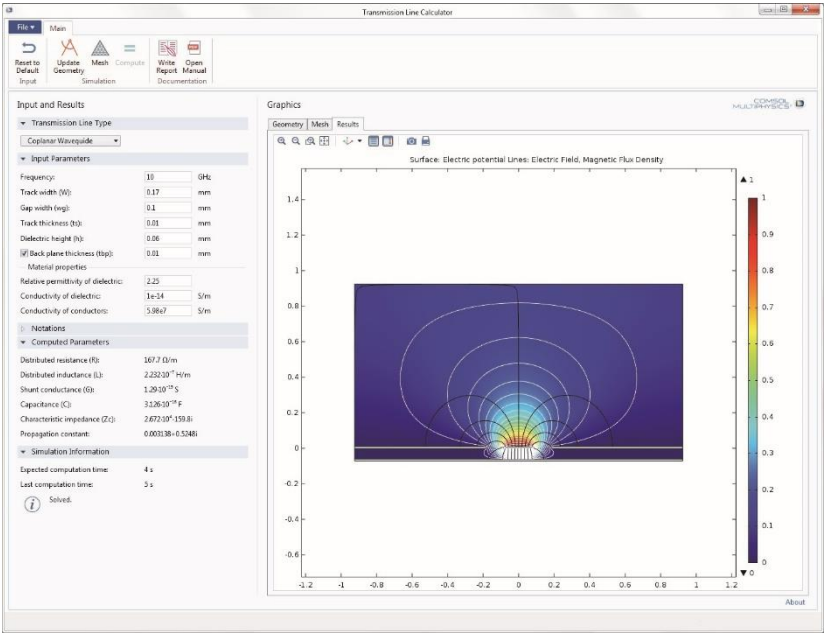
送電線計算機 *（Transmission Line Calculator）*

伝送線路は、無線周波数での電流と電圧の交流波を誘導するために使用されます。伝送線は様々な形で存在し、その多くは、プリント回路基板設計での製造と作業の容易性に適しています。それらは最新の電子機器における重要な要素であり、デバイス内やデバイス間のある場所から別の場所へ、最小の損失と歪みで情報を運ぶために使用されます。

このアプリケーションは、伝送線パラメータ R、L、G、C、並びに、以下に示すような一般的な伝送線路タイプのパラメータ化された断面の γ および Z_0 を計算するために、事前定義されたユーザインタフェースを提供しています。

- 同軸線
- ツインリード
- マイクロストリップライン
- コプレーナ導波路（CPW）

また、ジオメトリ、メッシュ、電位、電界線と磁束線のプロットも提供されます。このアプリケーションは、アドオン製品を必要としません。



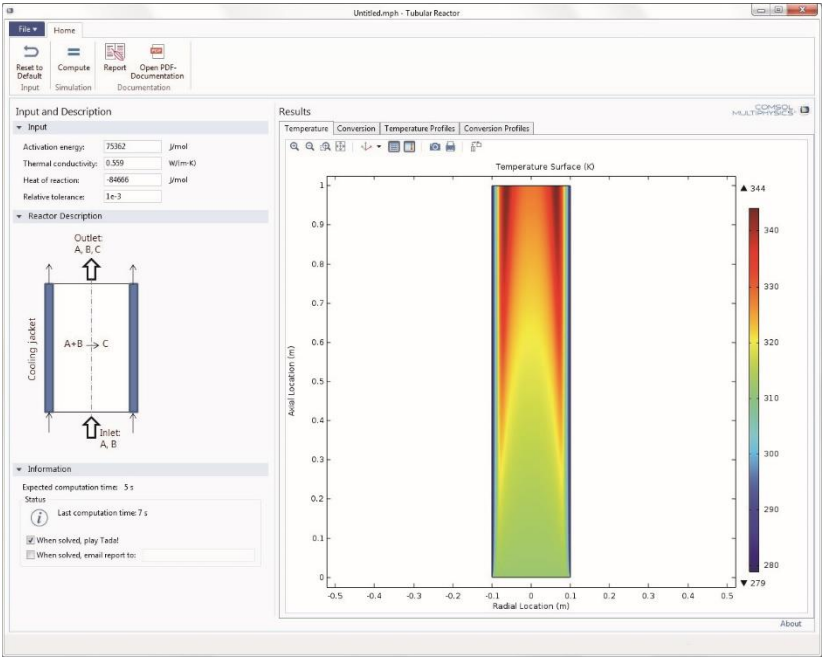
管状反応炉（*Tubular Reactor*）

このアプリケーションによって、化学工学の学生は、温度および構成の半径方向と軸方向の変形を含む非理想的な管状反応炉をモデル化し、異なる動作条件の影響を調査することができます。アプリケーションには、プロピレングリコールを形成する水を持つプロピレン酸化物の発熱反応のプロセスが記載されています。

また、このアプリケーションでは、教師が想像力の問題に挑戦する生徒に合わせたインタフェースを構築する方法が例示されています。このモデルや演習は、元々、スコットフォグラー氏の *Elements of Chemical Reaction Engineering* に記述されています。

数学的モデルは、軸対称座標系で説明されるエネルギーバランスと物質的バランスから成ります。学生は、反応の活性化エネルギー、熱伝導率、および反応炉の反応熱を変更することができます。結果として得られる解は、反応炉内の軸方向および半径方向の変換だけでなく、温度プロファイルを提供します。一部のデータについては、解析による結果が明らかではありません。これは、モデルの結果の解釈が問題解決の演習になるということも意味しています。

また、このアプリケーションでは、電子メールチェックボックスを選択し、電子メールアドレスを入力することによって計算の準備ができた時に、電子メールを送ることができます。これによって、設定に関するレポートと計算結果が送信されます。学生がこの機能を使用して、スーパーバイザに結果を送信することができます。計算に時間がかかるような場合には、この機能は非常に有用です。例えば、シミュレーションを開始し、オフィスや研究室を離れ、その後、道路やメールにアクセス可能などこかからアクセスし、計算が行われているアプリケーションからのフルレポートを得ることができます。このアプリケーションは、アドオン製品を必要としません。



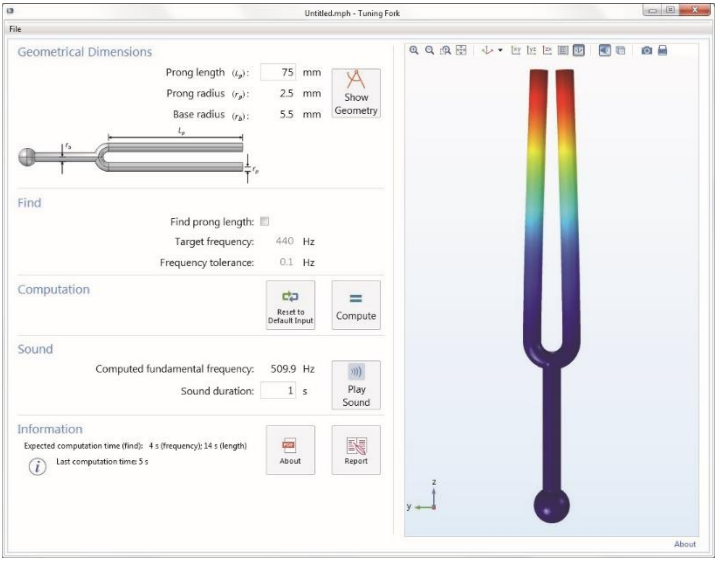
音叉（*Tuning Fork*）

このアプリケーションは、ユーザ定義の先端部長さによって、音叉の共振周波数を計算しています。代わりに、このアプリケーションでユーザ定義の目標周波数を指定することができ、対応する先端部長さを見つけることができます。先端部とハンドル半径は、市販の入手可能な音叉から得ます。

アプリケーションの埋め込みモデルは、COMSOL マルチフィジックスに含められている固体力学インタフェースを使って定義されて、アドオンの製品は必要としません。先端部長さの探索アルゴリズムは、割線法です。

計算の終わりに、組み込みメソッド playSound が、計算された周波数で正弦波音を再生するために使われています。

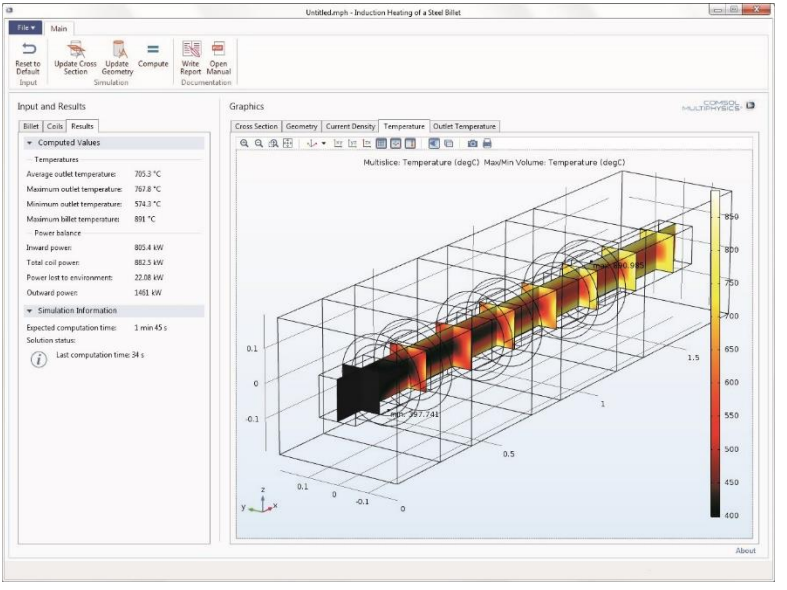
多くのバックグランドとなる理論を得るためには、モデル tuning_fork.mph のアプリケーションライブラリのドキュメントを参照してください。



スチール製ビレットの誘導加熱（*Induction Heating of a Billet*）

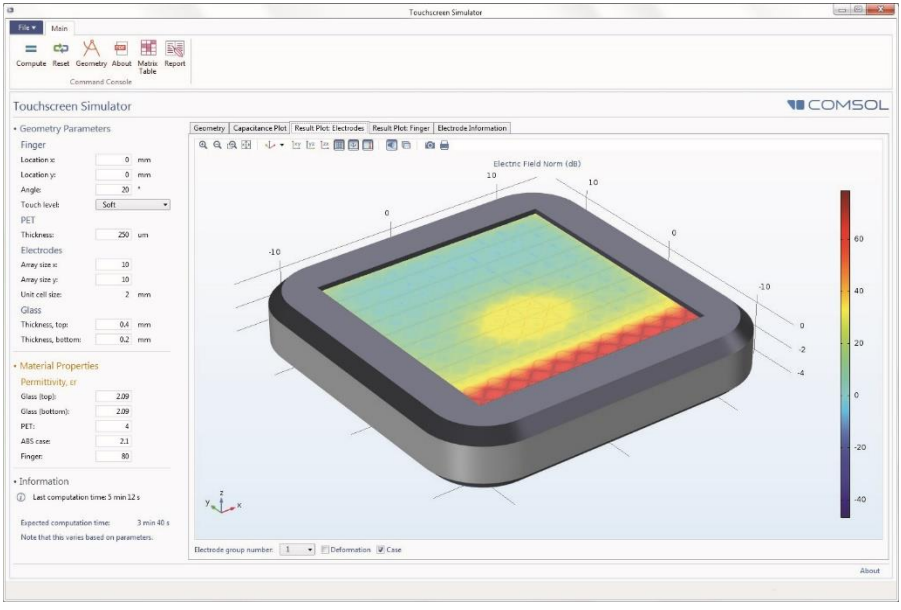
このアプリケーションは、スチール製ビレットの簡単な誘導加熱システムをデザインするために使用でき、一つ以上の電磁コイルから成ります。ビレットはそれを通して一定速度で動かされます。コイルは交流によってエネルギーを与えられて、金属ビレットの渦電流を引き起こし、ジュール加熱によって熱を発生します。

ビレット断面、コイル数、配置とサイズ、および初期だけでない周囲温度と個々のコイル電流の全てが、入力として指定されます。ソリューションが計算された後、アプリケーションは、処理中のビレット温度の 3D プロット、引き起こされた電流密度、出力断面温度の 2D プロットを表示します。最後に、アプリケーションは、ビレットの予測温度範囲とシステムのパワーバランスの数値データを計算します。



タッチスクリーンシミュレーター（*Touch Screen Simulator*）

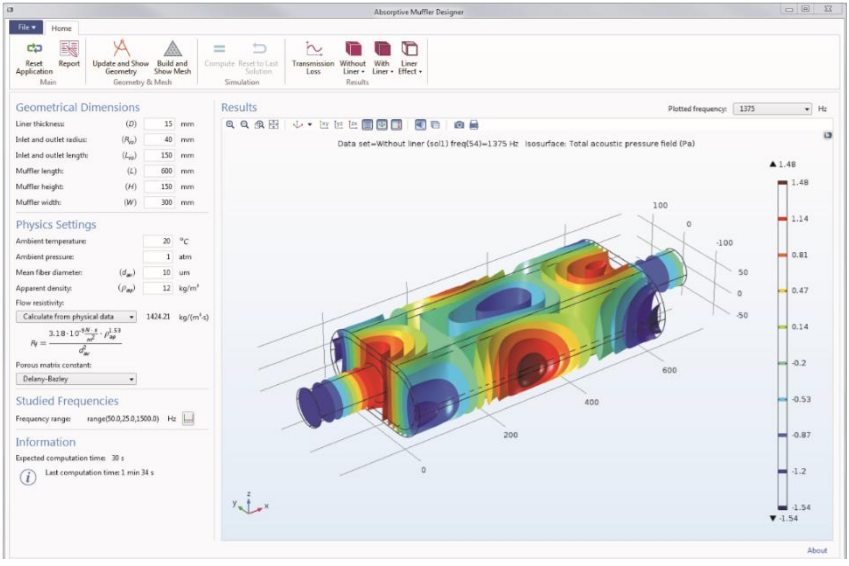
このアプリケーションは、人の指が接触した際のタッチスクリーンのささいな静電容量の反応を計算しています。この情報は、指の位置を検出する電子回路に利用されます。アプリケーションにおいて、指の位置と方向が入力パラメータを使って制御され、その結果として生成される静電容量行列が出力として計算されます。



吸収マフラーデザイナー（*Absorptive Muffler Designer*）

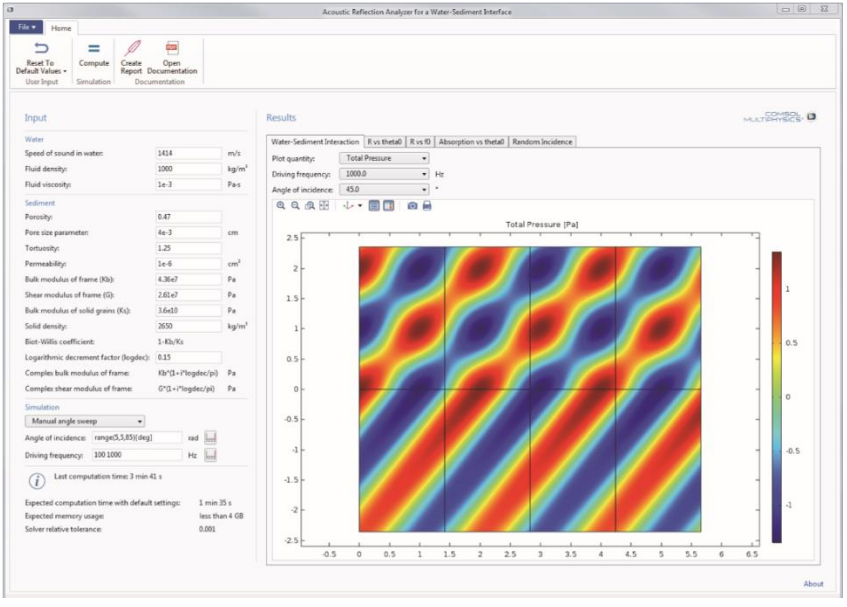
このアプリケーションは、多孔性のライニングによって、シンプルな共振マフラーを調べてデザインする目的に向いています。マフラーは、例えば内燃機関または HVAC システムによる放射ノイズを減衰させるのに使われ、一般に特定の周波数範囲でよく機能するはずです。減衰の度合いは透過損失（TL）と呼ばれて、周波数の関数として dB 単位で減衰が示されます。透過損失は、マフラーのジオメトリ、およびシステムに置かれる多孔性の繊維状材料の特徴に依存します。このアプリケーションは、多孔性ライナーの材料プロパティだけでなく、マフラーの寸法、周囲の作業条件を変更した結果を調べるのに利用されます。すなわち、これらの変更がどのようにシステムの透過損失に影響するかを調べることができます。

このアプリケーションは、与えられたマフラーモデルの“dynamic specification sheet”の一つの例です。セールスエンジニアはこのタイプのアプリケーションを顧客に持っていき、特別に設計されたカスタマイズされたマフラーの性能を示すことができます。例えば、マフラーが空間の制約を伴う車両に取り付けられるように設計できます。この場合の性能とは迅速な視覚化であるかもしれません。また、さまざまなオプションを顧客と検討することができます。



音響の反射分析器 （*Acoustic Reflection Analyzer*）

このアプリケーションは、水堆積物界面からの平面波の反射を分析しています。反射と吸収係数は、投射角と周波数の関数として決められます。さらに、ランダムな発生吸収係数、または放散するフィールド吸収係数は、解析されたデータに基づいて計算されます。流体の材料プロパティ、この場合は水、および多孔質媒体のプロパティ、ここでは半無限の堆積層、は変更可能です。



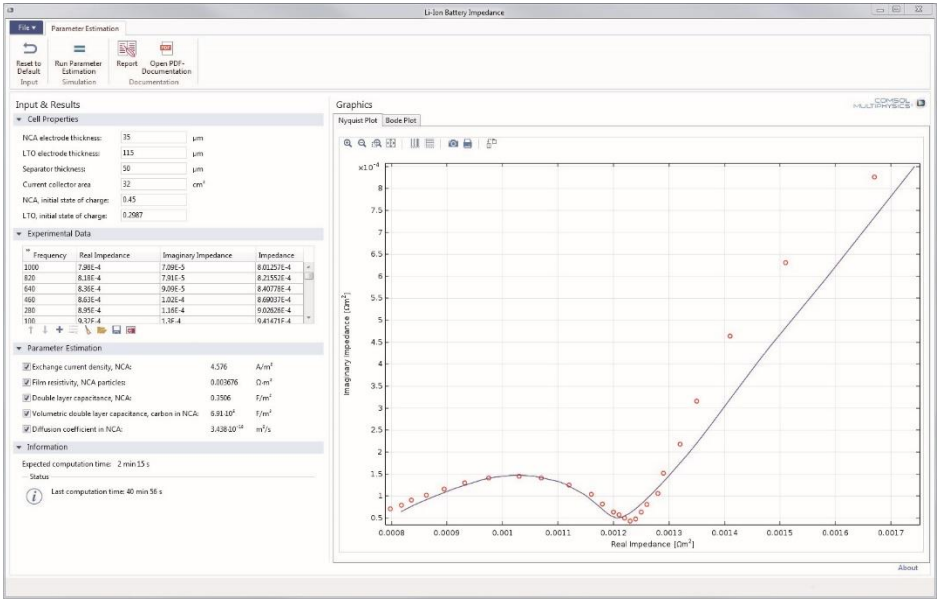
リチウムイオン電池インピーダンス（*Li-Ion Battery Impedance*）

このアプリケーションのゴールは、実験的な電気化学インピーダンス分光測定（EIS）を説明すること、およびリチウムイオン電池の特性を評価するためにどのようにモデルと測定を使用するかを示すことである。

アプリケーションは、EIS 測定による実験データを入力として取得し、その測定値を解析し、それから実験データに基づいてパラメータ評価を行います。

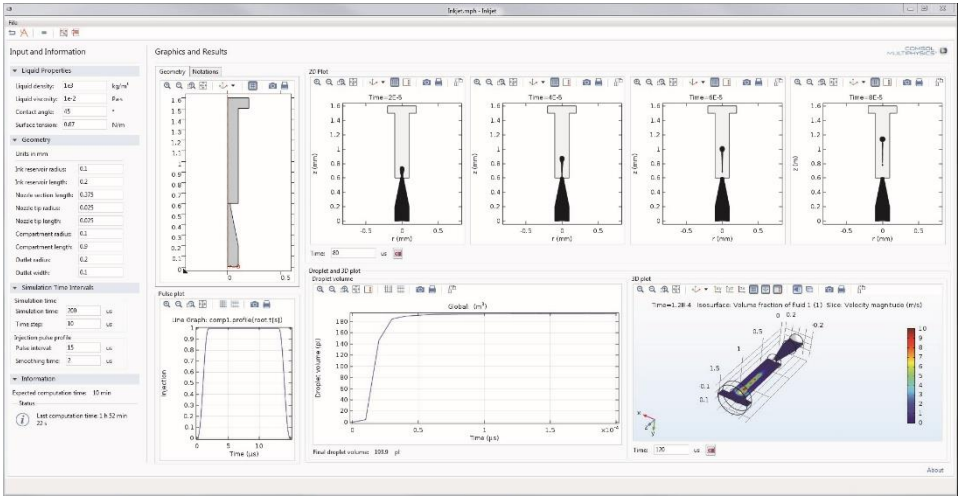
コントロールパラメータは、交換電流密度、粒子の抵抗層の抵抗率、NCA の二重層容量、および正極での炭素担体の二重層容量です。正極の測定インピーダンスが 10 mHz から 1kHz までの周波数範囲でグラフ化されます。

アプリケーションでは、コンマで区切られた CSV ファイル形式の実験データをロードするといった実演も可能です。パラメータ評価のために、最適化モジュールを利用しています。



インクジェット *（Inkjet）*

このアプリケーションの目的は、必要な粒子サイズにインクジェットノズルの形状と動作を合わせることですが、これは注入される液体の接触角、表面張力、粘度、密度次第です。解析の結果から、注入したインクが基質で最終粒子に融合する前に、いくつかの粒子に分散するかどうかもわかります。

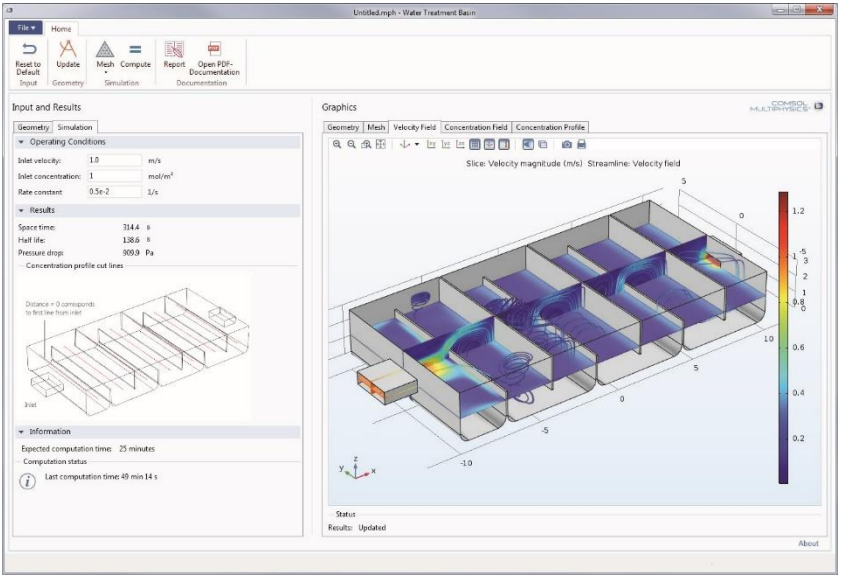


水処理池 *（Water Treatment Basin ）*

水処理池のアプリケーションの目的は、液体の 3D 乱流モデリングのためのアプリケーションの使用を実証することです。また、アプリケーションの興味深い面として、そのソリューションの中で溶質の材料バランスを説明しています。この溶液は、一次反応、すなわち良く希釈された化学種の減衰を説明するための反応の共通のタイプで反応します。このアプリケーションは、乱流モデリングのための完全にパラメータ化されたジオメトリ、および累積する選択肢をどのように使うかについても示してくれています。

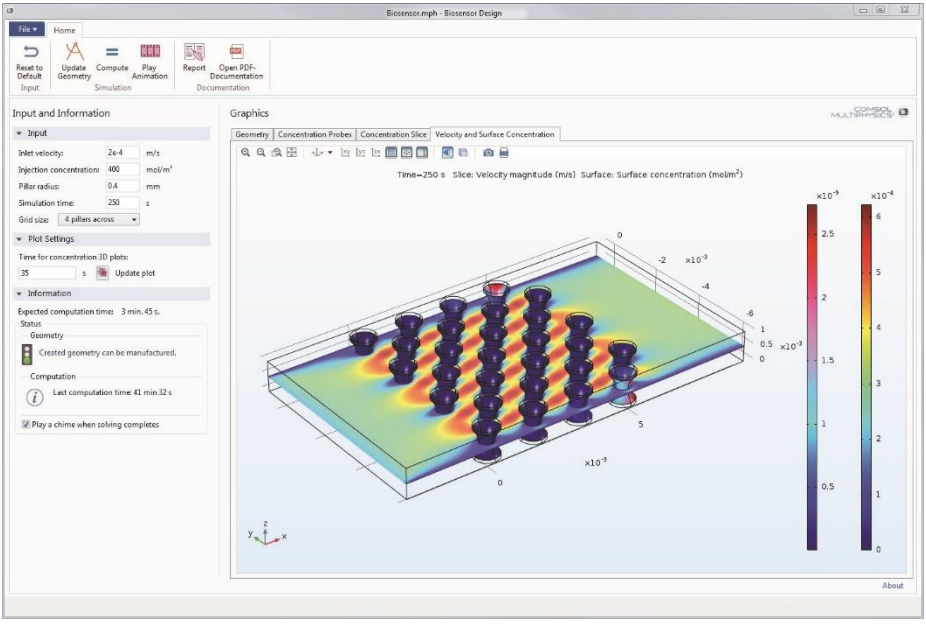
アプリケーションは、良く希釈された溶質の反応による液体の乱定常流モデリングを、あなた自身のアプリケーションの出発点として利用することができます。

例示されたシステムは、水処理プロセスにおける塩素処理池です。



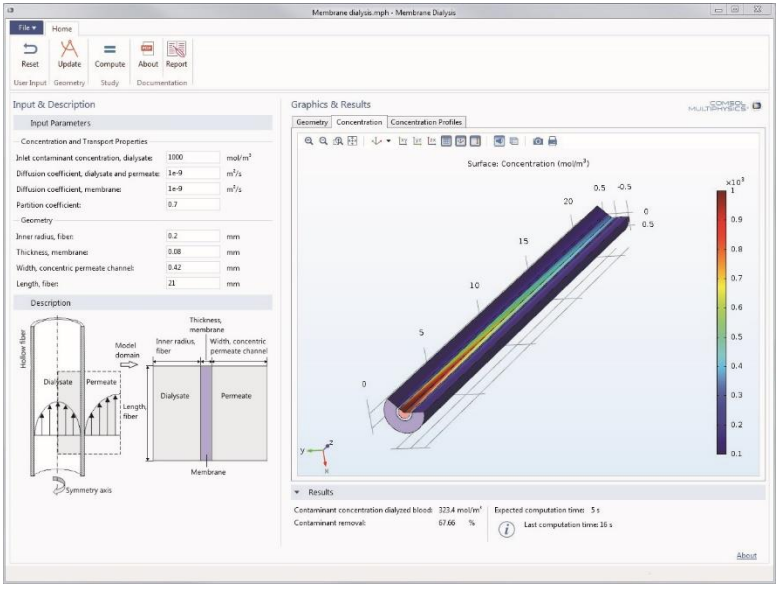
バイオセンサーの設計 （*Biosensor Design*）

バイオセンサーでのフローセルは、生体分子を検出するために使用されるマイクロピラーの配列が含まれています。柱は、選択サンプルストリーム中の生体分子を吸収する活物質でコーティングされています。これらの生体分子は、その後、表面に反応します。このアプリケーションは、設計変更が検出結果にどのように影響するか色々に試してみるために、柱の直径、柱の間隔、入力側の流速などのパラメータを変更してユーザがセンサの設計を変更することを可能にしています。ジオメトリおよび動作条件は、信号強度、および拡散性に大きな影響を与えます。なお、柱の距離の最小値として設定される「製造上の制約」はアプリケーション内で通知されます。



膜透析（*Membrane Dialysis*）

このアプリケーションは、膜透析装置内に精製される血流内の汚染物質の濃度を解析しています。モデル化された透析装置は、中空繊維の壁が汚染物質を取り除く膜として機能する中空繊維モジュールで構成されています。繊維の内部で透析液が流れ、その外側に透過液が通過します。このアプリケーションでは、入力パラメータを変化させることによって、デバイス内の汚染物質除去を最大化する方法についてアプローチすることができます。

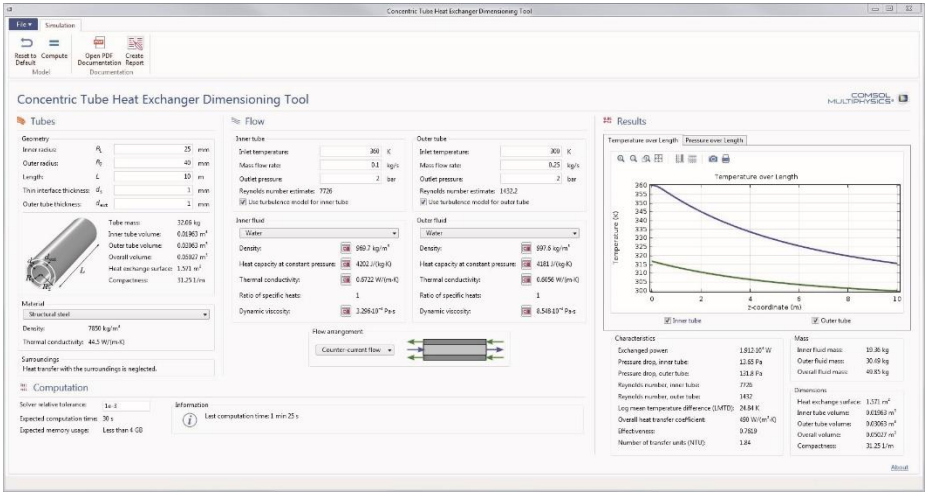


同心円状チューブ熱交換器（*Concentric Tube Heat Exchanger*）

寸法の数値は、熱交換器の動作の最初の指標です。このアプリケーションは、指定された構成におけるこれらの値を計算することを目指しています。

このアプリケーション例は、二つの別個の流体を分離する二つの同心円状チューブの場合をスタディしています。流体は向流または平行流のいずれかで実行することができます。ユーザインタフェースを使用して、チューブと流体の両方をカスタマイズすることができます。

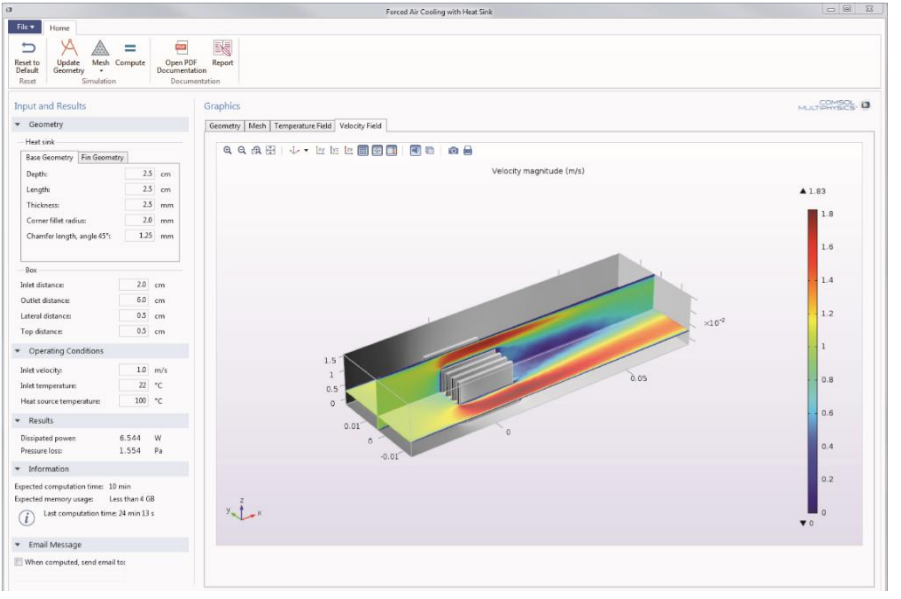
計算した後、温度プロフィール、およびいくつかの数値が表示されます。利用可能な容積、質量、コンパクト性（熱交換器の容積に対する交換表面の比率）、および材料特性などといった流体特性を追加で入力します。



フィン付きヒートシンク（*Heat Sink with Fins*）

ヒートシンクは通常、指定されるファン曲線における熱の放散能力がベンチマークされます。この種の実験を実施する上で可能な一つ方法として、絶縁壁を有する長方形のチャネルにヒートシンクを配置します。このチャネルの入口と出口での温度と圧力、ならびにヒートシンクベースを指定温度に保持するために必要な電力を測定します。これらの条件下で、チャネル上の圧力損失とヒートシンクによる消費熱量を推定することができます。

このアプリケーションの目的は、モデリングと解析を用いてベンチマーク実験の調査を行うことです。例えば、フィンの数が多いと放熱量も増えますが、それに従って、フィンが流れに対して大きな障害となって流れを減少させ、放熱量が逆に低下します。これは、チャネルを介して与えられた全圧力損失のために、最適な寸法と最高の冷却力を与えるフィンの数が存在する可能性があることを意味しています。このアプリケーションによって、そのような調査を行うことができます。



周期的な微細構造の同等特性

（Equivalent Properties of Periodic Microstructures）

周期的な微細構造は、しばしば、このような炭素繊維とハニカム構造体のような複合材料に見出されます。これらは、伝搬する 3 方向に沿って繰り返される単位セル（細胞）で表すことができます。解析で計算コストを削減するために、同等の特性を有する均質なドメインを有する複合材料の全ての詳細を同等の特性を有する均質なドメインに置き換えることができます。

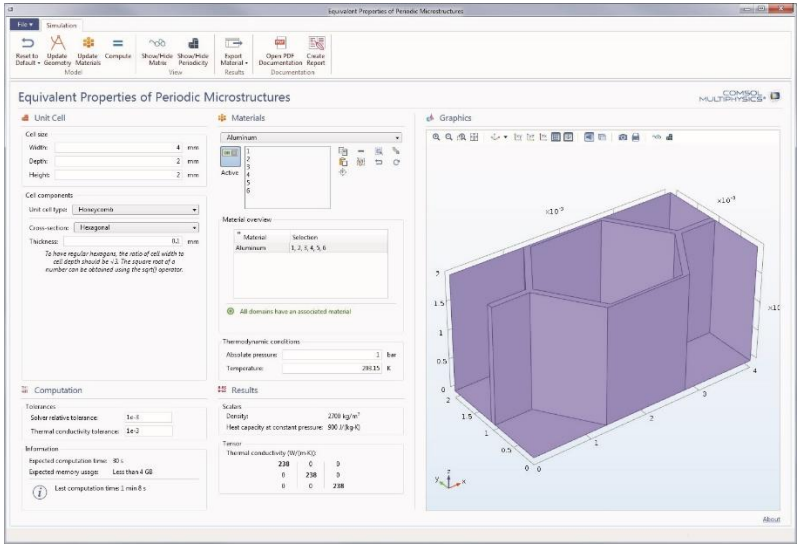
このアプリケーションは、ジオメトリ構成からの同等の特性と単位セルの材料特性を計算します。 これは、9 パラメータ設定可能な単位セルと 13 の事前に定義された材料リストの選択肢を提供しています。このアプリケーションを追加の平行六面体ユニットセルに拡張するか、他の材料を追加することは容易に行うことができます。

このような熱伝導方程式のような拡散に類似した方程式では、同等の拡散係数はテンソルの一般形を取ります。

このアプリケーションにおいて、以下の材料特性は、単位セルの種々の領域に与えられた材料と選択された単位セルの形状から計算されます。

- 密度
- 定圧熱容量
- 熱伝導率

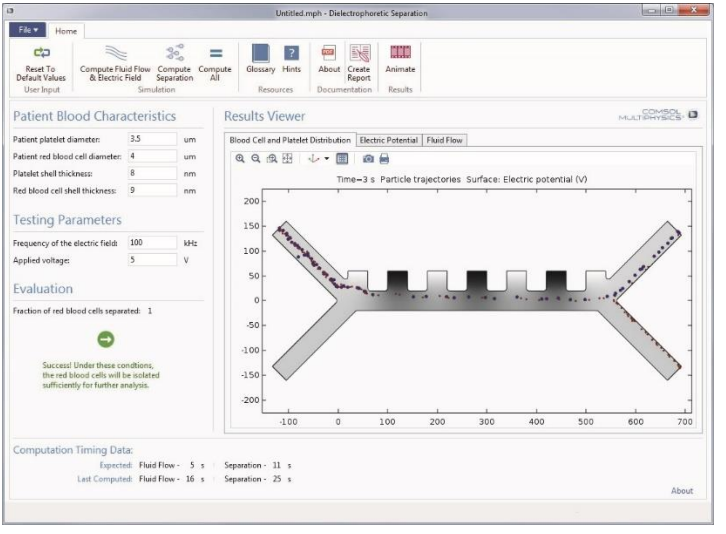
このアプリケーションに組み込まれている単位セルライブラリは、並列積層、繊維強化複合材料、またはハニカム構造体などのような、いくつかの広く使用されているセルタイプを含んでいます。いったんジオメトリが設定されると、フィジックスは、セルの反対の境界での周期的な熱条件で構成されます。



赤血球分離　（*Red Blood Cell Separation*）

誘電泳動（DEP）は、それが不均一な電場を受けたときの力が誘電体粒子に作用する現象です。電界はその後、電位の勾配に比例した DEP 力の対象となる粒子の分離を引き起こします。

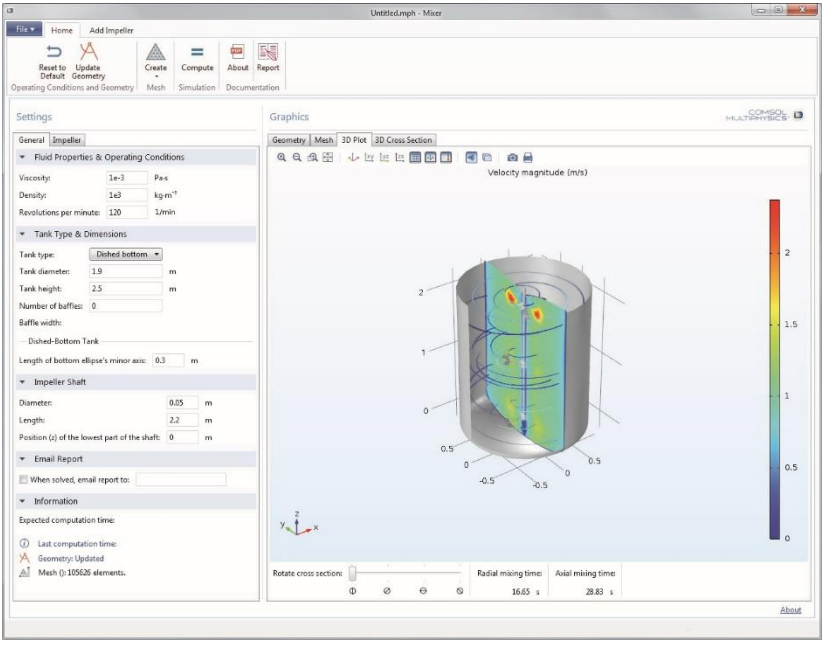
DEP 力は、大きさ、形状、および粒子の誘電特性に敏感です。このため、DEP が粒子の種類を分離するために利用されています。このプロセスの一つの用途は生物工学の分野にあり、DEP を混合物から異なる種類の細胞を分離するために使用することができます。このアプリケーション例では、血小板から赤血球を分離するために、血液試料を濾過して選択することができる方法を示しています。これは、血小板が凝血を引き起こす場合に有用です。いったん血液の凝固塊が形成されたら、その血小板で汚染されたサンプルが以降の試験には適さないということを知ることができます。



ミキサー *（Mixer）*

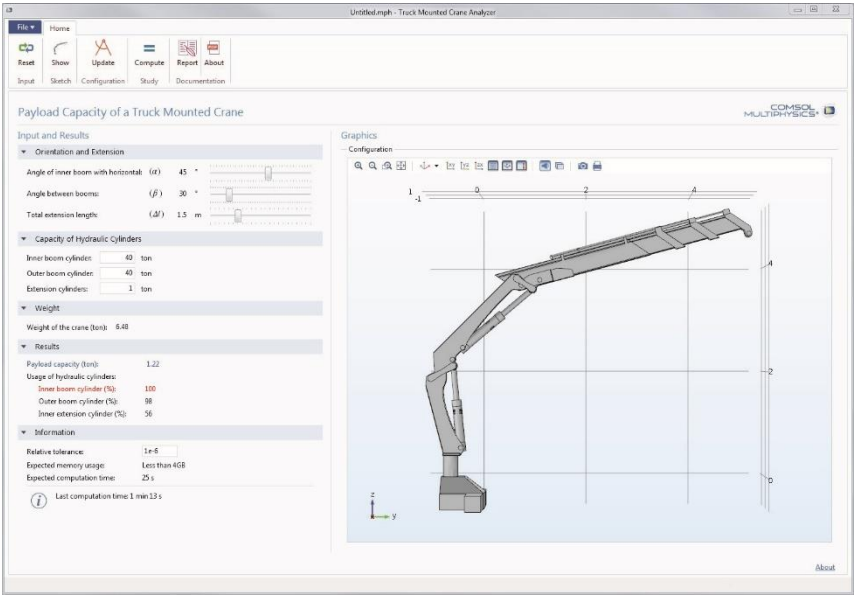
ミキサーアプリケーションの目的は、科学者、プロセス設計者、プロセス技術者が、容器、インペラ、および動作条件における混合効率やインペラの駆動に必要な電源への影響を調べることができ、ユーザフレンドリなインタフェースを提供することです。このアプリケーションは、与えられた流体でのミキサーの設計や動作を理解し最適化するために使用することができます。しかし、おそらく最も重要なのは、独自にミキサーと反応炉のモデリングと解析のアプリケーションを作る上で、出発点として利用することができることです。

アプリケーションは、埋め込みモデルで設定されているドメインとの境界を自動的に設定するための部品や度重なる追加選択に関する使用方法を示してくれています。アプリケーションのユーザの選択によって非常に多様なジオメトリを作成する場合でも、これらの設定を自動的に作成することができます。



トラック積載型クレーンアナライザー（*Truck Mounted Crane Analyzer*）

多くのトラックは、荷役用クレーンが装備されています。このようなクレーンは、クレーンの動き、およびいくつかのメカニズムを制御するいくつかの油圧シリンダを持っています。

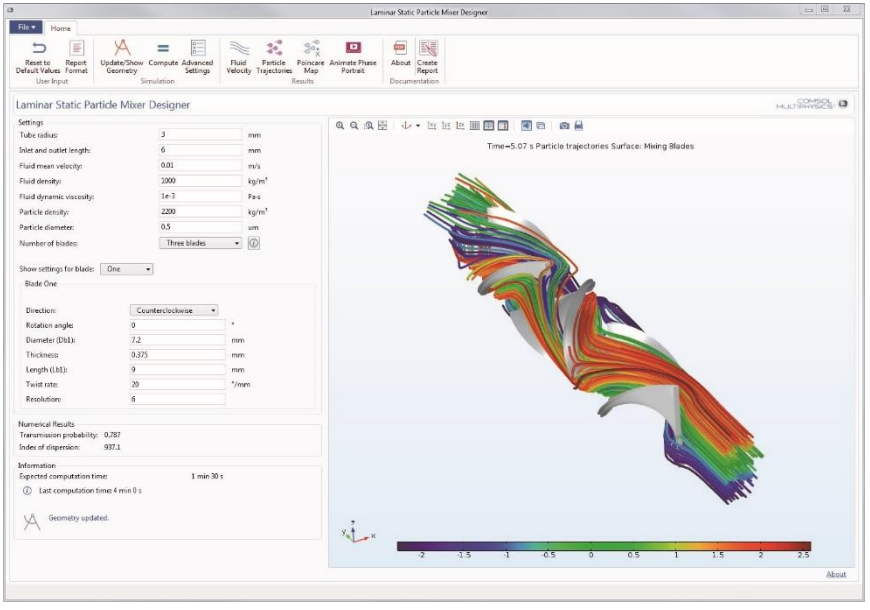


このアプリケーションにおいて、クレーンの剛体分析は、クレーンの指定された方向と伸長のためのペイロード能力を見つけるために実行されます。また、このアプリケーションは、油圧シリンダの用法も提供し、制限シリンダを強調しています。油圧シリンダの容量は、ペイロード容量とシリンダの使用を改善するために変更することができます。

静的層流粒子ミキサーデザイナー

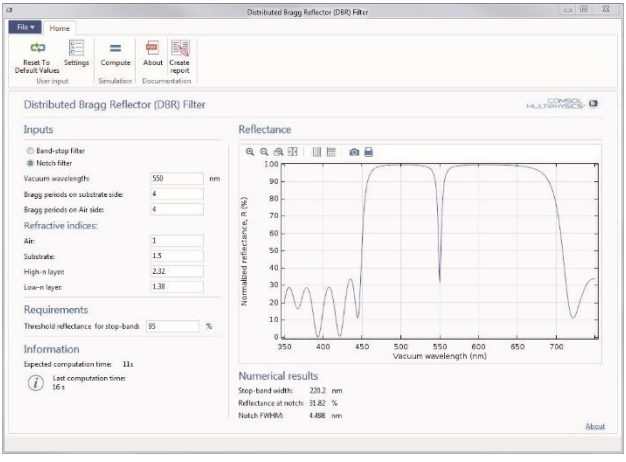
（Laminar Static Particle Mixer Designer）

静止ミキサーでは、固定混合ブレード付きのパイプから流体を汲み上げます。この混合技法は、主に、層流混合に適していますが、これはこの流動の枠組みで生じる圧力損失が小さいためです。このアプリケーションは、ツイストブレード静止ミキサー内の流れをスタディしています。これは、ミキサーを通して浮遊粒子の軌道を計算し、混合性能を評価しています。このアプリケーションは、室温で溶媒中に溶かされた 1 種類の静的混合を計算しています。粒子混合上の流体と粒子の特性の影響だけでなく、静止ブレードの構成を学ぶことができます。



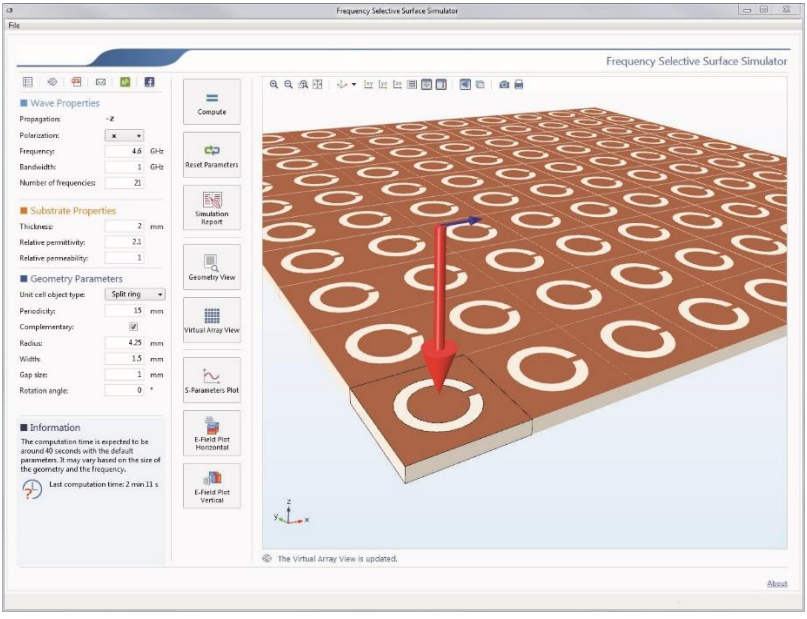
分布型ブラッグ反射器フィルター（*Distributed Bragg Reflector Filter*）

分布型ブラッグ反射器 (DBR) フィルターは、二つ 2 の材料を交互に複数重ねた層からなります。材料はそれぞれ屈折率が異なり、DBR 層に垂直方向に高い屈折率と低い屈折率の繰り返しパターンが発生します。光がこの構造を伝播すると、層間の各インタフェースで反射が発生します。複数の反射波の間での干渉効果は、DBR の反射率が波長の依存性が高い原因となります。通常の金属製鏡の上の DBR の主な利点は、DBR が選択された波長でのカスタムな反射率となるように操作することができるということです。



周波数選択面シュミレーター（*Frequency Selective Surface Simulator*）

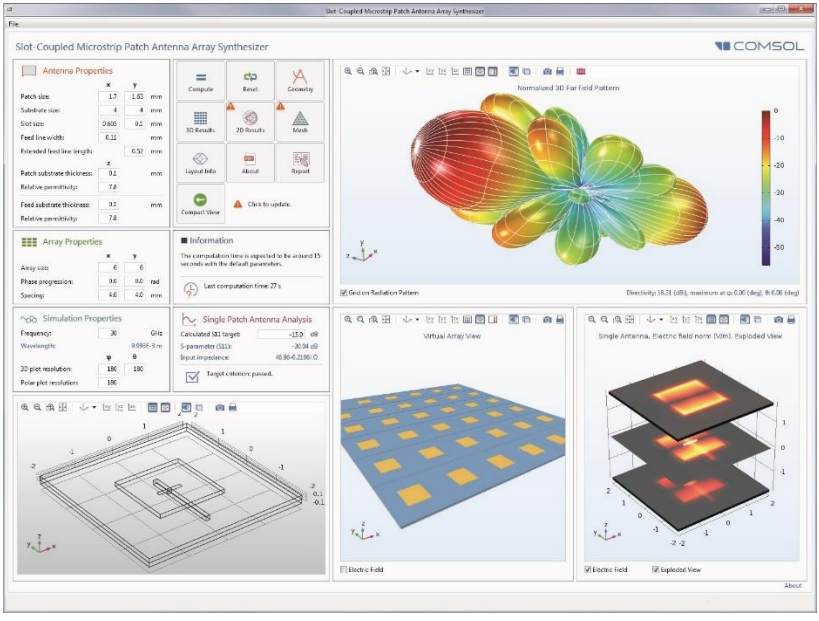
周波数選択面（FSS）は、バンドパス周波数応答またはバンドストップ周波数応答を生成する周期的構造です。このアプリケーションは、組み込み単位セルタイプから選んだユーザ指定の周期的構造を解析しています。このアプリケーションでは、FSS 解析で一般に使用されている 五つの単位セルタイプのほか、FSS に対して法線入射になる 1 方向に固定の伝播方向の二つの定義済み偏光を用意しています。本解析には、反射スペクトルと透過スペクトル、単位セルの上面の電場ノルム、単位セルドメインの垂直切断面に表示される dB スケールの電場ノルムが組み込まれています。



マイクロストリップパッチアンテナアレイシンセサイザー

（Microstrip Patch Antenna Array Synthesizer）

このアプリケーションは、重層化低温共焼成セラミック（LTCC）基板上に組み立てたシングルスロット結合マイクロストリップパッチアンテナをシミュレートしています。その結果には、アンテナアレイとその指向性の遠視野放射パターンを含みます。遠視野放射パターンは、複雑なフルアレイモデルをシミュレートしなくてもアレイ要素とシングルアンテナ放射パターンを掛け合わせて効率的な遠視野解析を実行して近似します。また、5G モバイルネットワークのフェーズドアンテナアレイプロトタイプは、デフォルトのデフォルト入力周波数 30 GHz で評価できます。また、アプリケーションでは、カメラがアンテナの周りを移動するアニメーションを実演しています。

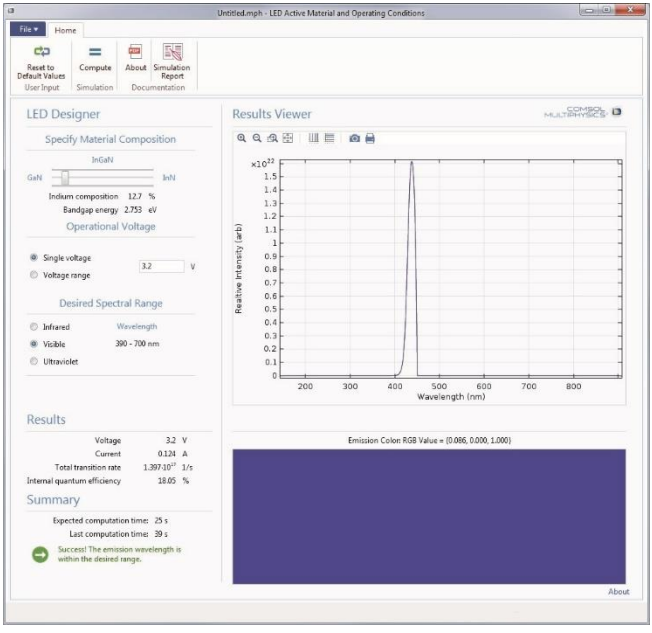


波長可変 LED　（*Wavelength Tunable LED*）

青色 LED は、現代の高効率照明としての利用が興味深いものです。その大きなバンドギャップエネルギーの故に、ガリウム窒化物が青色光を生成するために広く利用されています。このアプリケーションは、窒化ガリウム系発光ダイオードの発光特性を解析しています。

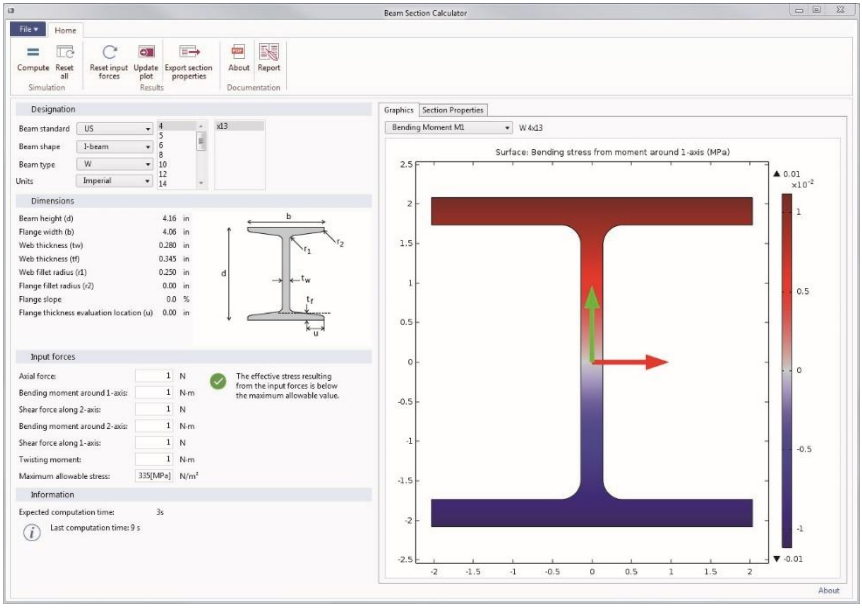
デバイスの活性領域に用いられる材料は、InxGa1•xN です。それには、ガリウムとインジウムの両方の混合物が含まれており、インジウムの割合は x で与えられています。この光学活性領域のバンドギャップは、インジウムの割合を変化させて材料の組成を変えることによって制御することができます。純粋な InN と GaN は、それぞれ、スペクトル範囲の赤外線および紫外線領域で放射するので、この技術を使って全可視スペクトルにわたって InxGa1•xN の発光エネルギーをチューニングすることが可能です。

このアプリケーションは、デバイスのインジウム割合および動作電圧を制御することができます。その後、デバイスの電流、発光強度、エレクトロルミネッセンススペクトル、および内部量子効率が計算されます。単一の動作電圧または電圧範囲のいずれかを入力することができます。電圧範囲が入力された場合、電流 – 電圧曲線も計算され、デバイスのターンオン電圧の決定が可能になります。このアプリケーションは、非常に多くのメソッドが使われています。



梁断面の計算機 （*Beam Section Calculator*）

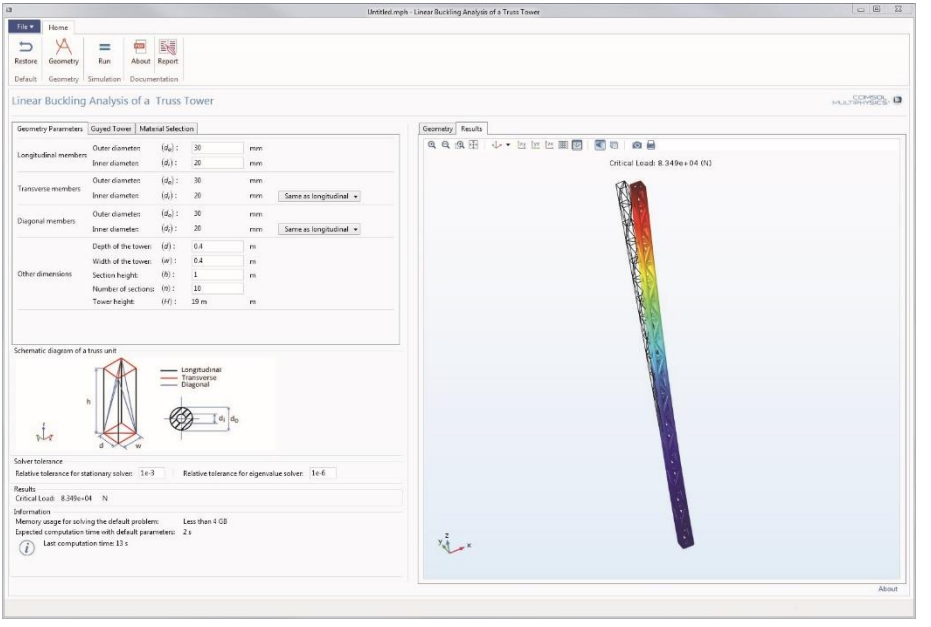
このアプリケーションは、設計されるスチール製の梁断面の梁断面特性を計算しています。また、それに作用する力とモーメントが加わる断面全体にわたる詳細な応力分布を計算することができます。アメリカやヨーロッパの標準的な梁に広範囲に利用可能です。LiveLink™ for Excel® 製品のライセンスでは、全ての入力および結果のデータは Excel® のファイルにエクスポートされたてテーブル表示されます。この梁の寸法データが含まれる Excel® のワークブックを編集し、そのデータをアプリケーションに戻す再インポートをすることができます。



トラスタワー座屈 （Truss Tower Buckling ）

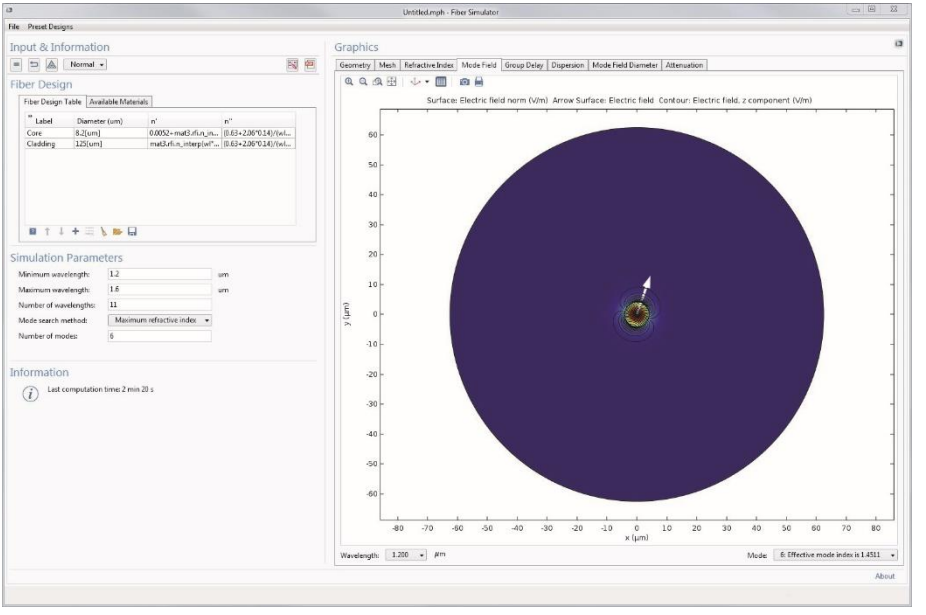
座屈解析では、それを超えと構造が不安定になる圧縮荷重の臨界を探します。このアプリケーションは、垂直方向の圧縮荷重下のトラスタワーの座屈を解析することができます。タワーは、必要に応じて支線ワイヤによって支持することができます。アプリケーションの目的は、ジオメトリの様々な条件下、すなわち、タワーの高さ、断面積、ならびに異なる材料でのタワーの座屈負荷を計算して分析することです。

このアプリケーションでは、計算を実行しながら、死荷重（トラスの自重と支持支線ワイヤとそのプリテンション（引張力））の効果を考慮します。



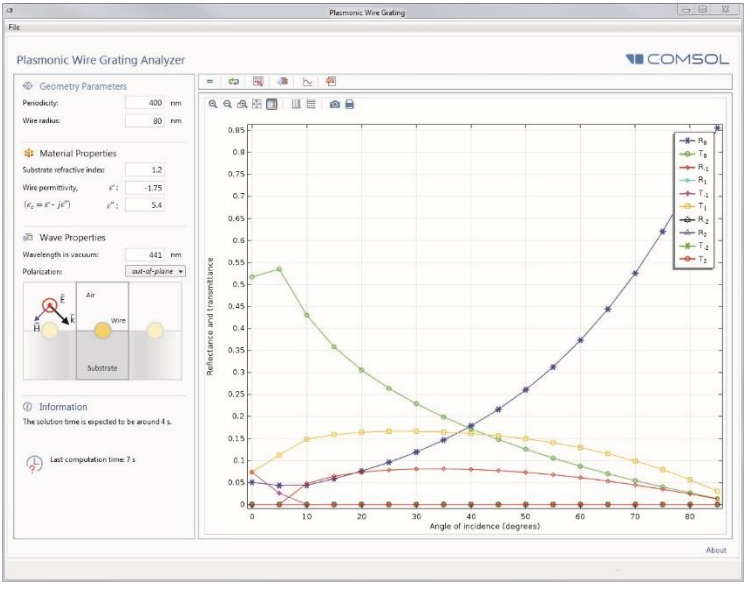
ファイバシミュレーター（*Fiber Simulator*）

このアプリケーションは、同心円状の誘電体層構造のモード解析を行っています。それぞれの層は、外径、および屈折率の実部と虚部によって描かれます。屈折率の式は、波長と半径方向の距離の両方に依存します。従って、このシミュレーターは、ステップインデックスファイバとグレーデッドインデックスファイバの両方を分析するために使用することができます。これらのファイバは、同心円状の任意の層数を有することができます。計算結果には、群遅延と分散係数を含んでいます。



プラズモニックワイヤーグレーティング（*Plasmonic Wire Grating*）

このアプリケーションは、誘電性の基板ときしんでいるワイヤの入射角度の関数として、送信波と反射波（m = 0）と、第 1 および第 2 の回折次数（m = ±1 と±2）の回折効率を計算しています。平面波の入射角は、垂直入射から斜入射に掃引されます。また、このアプリケーションは、選択した入射角度の複数の微小角期間の電界ノルムプロットを表示します。


Plasmonic Wire Grating
File
Plasmonic Wire Grating Analyzer
COMSOL
Geometry Parameters
Periodicity: 400 nm
Wire radius: 80 nm
Material Properties
Substrate refractive index: 1.2
Wire permittivity, ε': -1.75
(ε_z = ε' - jε'') ε'': 5.4
Wave Properties
Wavelength in vacuum: 441 nm
Polarization: out-of-plane
Air
Wire
Substrate
Information
The solution time is expected to be around 4 s.
Last computation time: 7 s
Reflectance and transmittance
Angle of incidence (degrees)
R0
T0
R-1
R1
T-1
T1
R2
R-2
T2
T-2
About

Introduction to Application Builder 日本語訳版（バージョン 5.3a）

2018 年 02 月 22 日　初版発行（ バージョン 5.3a ）

著者：　COMSOL AB.　/　COMSOL, Inc.
編集：　計測エンジニアリングシステム株式会社

Printed in Japan

COMSOL
URL: https://www.comsol.jp

KESCO KEISOKU ENGINEERING SYSTEM
計測エンジニアリングシステム株式会社
〒101-0047 東京都千代田区内神田 1-9-5 井門内神田ビル 4F
TEL: 03-5282-7040/FAX: 03-5282-0808
URL: http://www.kesco.co.jp
E-mail: support@kesco.co.jp