MapleMBSE User's Guide

Copyright © Maplesoft, a division of Waterloo Maple Inc.
2019

MapleMBSE User's Guide

版権

Maplesoft、MapleMBSE、Maple は、すべて Waterloo Maple Inc. の商標です。

© Maplesoft, a division of Waterloo Maple Inc. 2019. All rights reserved.

本書のいかなる部分も、電子的、機械的、写真、記録など形式および手段を問わず、複製、検索システムへの保存、転写を行うことを禁止します。本書の情報は、予告なく変更することがあり、販売元がその内容を保証するものではありません。本書で説明しているソフトウェアは、ライセンス合意に基づいて提供されるもので、その合意に従う場合に限り使用および複製を許可します。合意で明示的に許可されている場合を除き、メディアを問わずソフトウェアをコピーすることは法律違反となります。

Microsoft、Excel、Windows は、Microsoft Corporation の登録商標です。

Oracle、Java、および JRE は、Oracle およびその関連企業の登録商標です。

IBM、Rational、および Rhapsody は世界のさまざまな地域で登録された International Business Machines Corp. の商標または登録商標です。

他のすべての商標は、その所有者が所有権を有します。

本書は、Mapleの特別バージョンおよびDocBookを使用して作成されています。

目次

はじめに vii
1 はじめての MapleMBSE 1
1.1 MapleMBSE コンポーネント 1
サポートされるモデル形式 4
1.2 MapleMBSE の起動 4
コマンドラインからの起動 7
1.3 MapleMBSE 固有の機能の使用 8
モデルスプレッドシートのリロード/更新 8
Excel ファイルのエクスポートとインポート 9
現在のモデルのコピーを保存 9
1.4 モデルの保存 9
1.5 MapleMBSE の終了 10
1.6 No Magic Teamwork Cloud Adapter の使用 11
MapleMBSE での TWCloud モデルの起動 11
モデルの編集と変更の保存 13
モデルの変更の自動保存 16
2 MapleMBSE におけるモデルスプレッドシートの編集 19
2.1 ワークブックとレイアウト 19
2.2 操作の概要 21
可能な操作 23
MapleMBSE エラーメッセージにつながる編集操作 24
MapleMBSE とモデル要素間のマッピングに否定的な変化をもたらす操作 25
2.3 モデル要素の追加 25
ネスト化された要素 27
代替グループ 30
2.4 レコードのソート 35
2.5 モデル要素の削除 37
参照つきレコードの削除 39
2.6 モデル要素の変更 39
フィールドのタイプ 40
レコードの状態 42

要素追加時に複数のフィールドを使用する例 42
紐付けされていないレコードの属性フィールドの変更 47
紐付けされていないレコードの参照フィールドの変更 49
紐付けされたレコードの属性フィールドの変更 51
データ型の変換 54
紐付けされたレコードの参照フィールドの変更 55
非マッピングフィールド 58
2.7 ワークシートでの要素のロック 61
編集中のロック取得 61
ロック取得の失敗 62
変更の確定およびコードの解除 62
ほかの依存関係に関する操作 63
2.8 Excel でのその他の編集操作 63
CutCopy モード 63
セルの書式設定 63
A ログファイル 65
A.1 ログファイルの説明 65
A.2 ログファイルを使用した問題の特定 65
B エラーメッセージとトラブルシューティング 67
B.1 MapleMBSE 起動時のエラーメッセージ 67
B.2 SyncView 表を変更しようとしたときに表示されるエラーメッセージ ... 67
B.3 モデルスプレッドシートの編集時に表示されるエラーメッセージ 68
B.4 編集時のエラーメッセージ (処理できない例外) 69
B.5 メモリ残量に関する警告メッセージ 69
B.6 トラブルシューティング 70
Java ヒープサイズの変更 70

図目次

図 1.1: MapleMBSE が機能するしくみの概要図 2
図 1.2: モデル、メモリ内モデル、およびスプレッドシートの関係 3

はじめに

MapleMBSE の概要

MapleMBSE™では、スプレッドシートに基づくユーザインターフェースにより、構造、挙動、要件、およびパラメータ制約を含む詳細な設計定義を直感的に入力することができます。

関連製品

MapleMBSE 2019 を使用するには、以下の製品のインストールが必要です。

- Microsoft® Excel® 2010 Service Pack 2、Excel 2013 (最新バージョン。詳細についてはリリースノートを参照) または Excel 2016
- Oracle® Java® SE Runtime Environment 8

注 : MapleMBSE では以下の順序で Java Runtime Environment を探します。

1) **OSGiBridge.ini** に指定された -vm オプションを使用する場合 (デフォルトでは指定なし)、それが使用されます。

2) 現在の環境にシステム JRE (つまり、環境変数 JRE_HOME および JAVA_HOME で (この順に) 指定された JRE、または Windows レジストリ (JRE インストーラで作成) で指定された JRE) がある場合は、システム JRE が使用されます。

3) MapleMBSE インストールディレクトリにインストールされている JRE。

MapleMBSE で IBM® Rational® Rhapsody® を使用する場合、サポートされているバージョンは以下のとおりです。

- Rational Rhapsody バージョン 8.1.5

Teamwork Cloud™ を使用する場合、サポートされているバージョンは以下のとおりです。

- Teamwork Cloud™ サーバ 18.5 またはバージョン 19.0

サーバ以外のサポート対象製品のアーキテクチャ (32 ビットまたは 64 ビット) は、MapleMBSE のアーキテクチャと一致している必要があります。

関連リソース

リソース	説明
MapleMBSE インストールガイド	MapleMBSE のシステム要件およびインストール手順は以下のとおりです。『**MapleMBSE インストールガイド**』は、MapleMBSE のインストール DVD または MapleMBSE のインストール先のフォルダに保存されている **Install.html** ファイルに用意されています。

追加情報については、**http://www.maplesoft.com/site_resources** を参照してください。

サポートについて

カスタマーサポートまたはテクニカルサポートを受けるには、**http://www.maplesoft.com/support** を参照してください。

カスタマーフィードバック

Maplesoft はユーザからのフィードバックを歓迎します。MapleMBSE 製品のドキュメントに関連するコメントは、doc@maplesoft.com までご連絡ください。

第1章 はじめての MapleMBSE

この章の内容は以下のとおりです。

- *MapleMBSE コンポーネント [1ページ]*
- *MapleMBSE の起動 [4ページ]*
- *MapleMBSE 固有の機能の使用 [8ページ]*
- *モデルの保存 [9ページ]*
- *MapleMBSE の終了 [10ページ]*
- *No Magic Teamwork Cloud Adapter の使用 [11ページ]*

1.1. MapleMBSE コンポーネント

MapleMBSE のプラットフォームでは、ダイアグラムベースのモデル情報を表形式にマップできます。ダイアグラムベースのモデルは、UML、SysML、またはモデリングツール固有のほかの形式で定義されたモデルです。サポートされるモデル形式のリストについては、*サポートされるモデル形式 [4ページ]*を参照してください。

以下の図は、MapleMBSE で使用するコンポーネントと、各コンポーネントの相互関係を示しています。

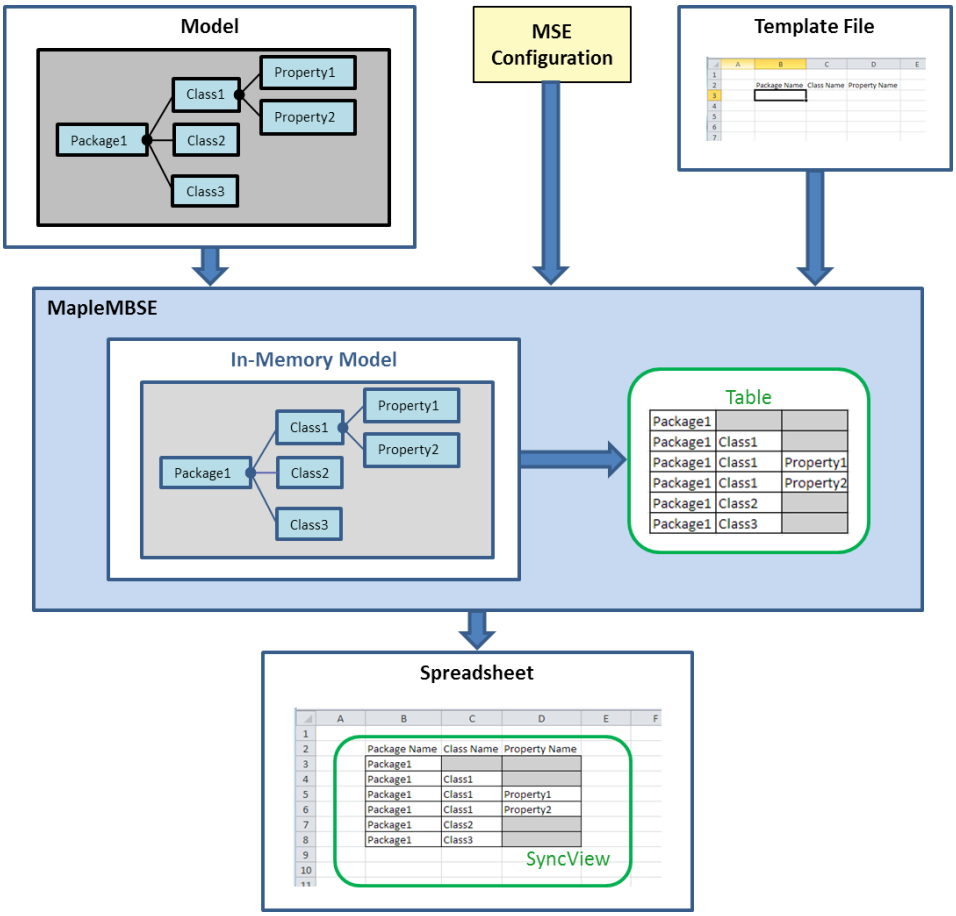


図1.1 MapleMBSE が機能するしくみの概要図

- **モデル**
  表示/編集するファイルシステム内またはサーバ上のモデル。*サポートされるモデル形式 [4ページ]*を参照してください。
- **スプレッドシート**
  モデルデータの表示/編集用のユーザインターフェースとして使用するExcelスプレッドシート。
- **メモリ内モデル**
  MapleMBSE の起動時に、特定のモデルを MapleMBSE の内部メモリにロード

します。スプレッドシート内のデータは、このメモリ内モデルと同期されます。

- **MSE 設定**
  ユーザが設定ファイル(.MSE)を作成します。このファイルでは、モデルを変換してスプレッドシートに表示するルールを指定します。以下の項目をルールで指定します。

1. モデルからデータを抽出する方法と表形式にマップする方法。上の図で、表は緑色で示されています。
2. 抽出された表をスプレッドシート上で表示する方法と位置。SyncView 表を表示するスプレッドシートの領域。*図1.1「MapleMBSEが機能するしくみの概要図」 [2ページ]* で、SyncView は緑色で示されています。

MSE 設定言語の詳細については、*『MapleMBSE Configuration Language Guide』* を参照してください。

- **テンプレート**
  MSE 設定ファイルと同じ名前の Excel ファイル。MSE 設定ファイルと同じディレクトリに保存されています。このファイルでは、特定の設定ファイルの書式(列の幅やヘッダなど) を定義します。詳細については、*『MapleMBSE Configuration Language Guide』* を参照してください。

モデルファイル、MapleMBSEのメモリ内にロードされたモデル、およびスプレッドシートの関係を以下の図に示します。

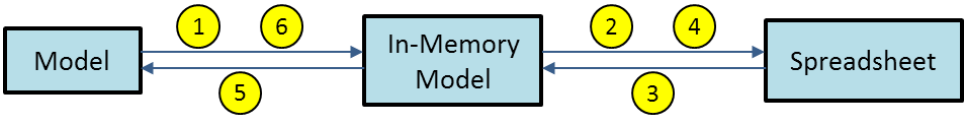


図1.2 モデル、メモリ内モデル、およびスプレッドシートの関係

1. MapleMBSEが特定のファイルから自身の内部メモリにモデルをロードします。
2. メモリ内モデルのデータがスプレッドシートの SyncView 領域にロードされます。
3. SyncView領域でのユーザ入力がメモリ内モデルへの変更として変換されます。

4. 同じ SyncView または更新された要素にリンクされているその他の SyncView の場所が更新されます。
5. メモリ内モデルへの変更がファイルシステム内またはサーバ上のモデルに適用されるのは、スプレッドシートが保存されるときのみです。
6. ファイルシステム内またはサーバ上のモデルに変更が加えられた場合は、[Reload/Update] 機能でその変更内容をメモリ内モデルにリロードできます (*MapleMBSE 固有の機能の使用 [8ページ]*を参照)。

注 :

- MapleMBSE とその他のツールで同じモデルを同時に操作することはサポートされていません。
- MapleMBSE で変更中のモデルを別のツールで変更することは推奨されません。

サポートされるモデル形式

MapleMSBE 2019 では以下のバージョンの UML モデル形式をサポートしています。

- UML® 2.0 (**http://www.uml.org**)

対応するアダプタがインストールされている場合は、以下のモデルがサポートされます。

- IBM® Rational® Rhapsody® (MapleMBSE Rhapsody Adapter を使用)
- Teamwork Cloud™ 上のモデル (MapleMBSE Teamwork Cloud Adapter を使用)

1.2. MapleMBSE の起動

注 : No Magic Teamwork Cloud Adapter を使用して起動する方法については、*No Magic Teamwork Cloud Adapter の使用 [11ページ]*を参照してください。

MapleMBSE を起動するには、以下の手順に従います。

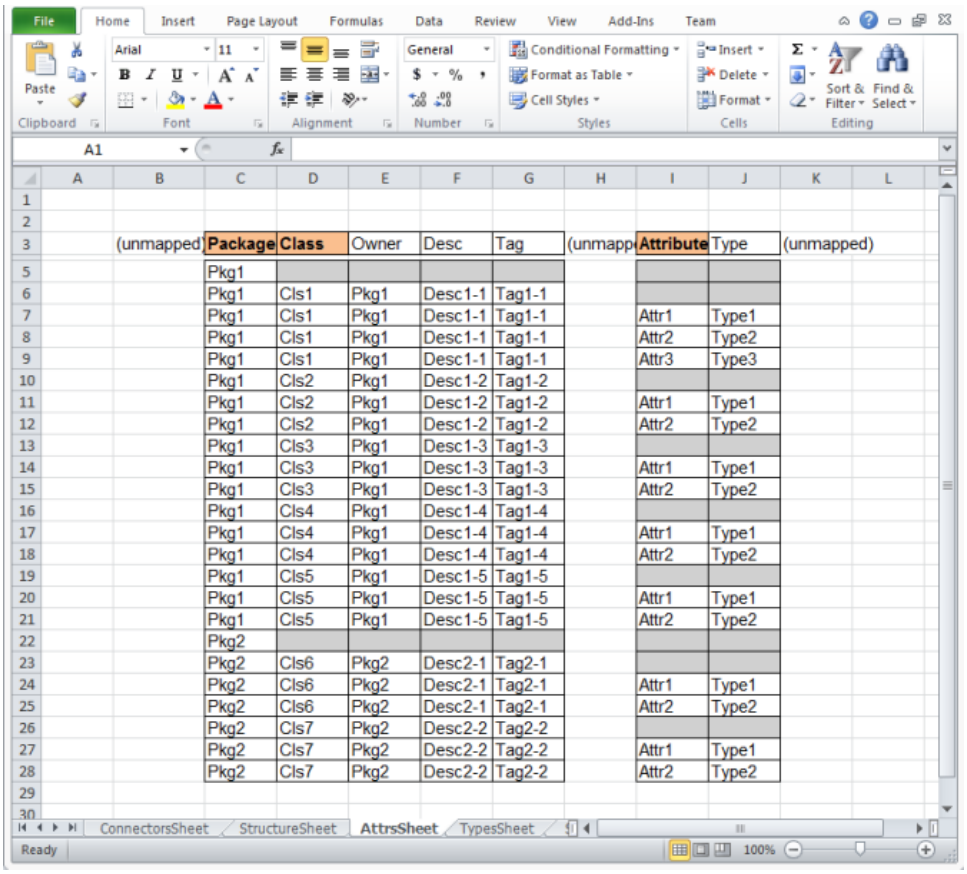
1. **[スタート]** メニューから **[すべてのプログラム] > [MapleMBSE 2019]** を選択します。
2. **[MapleMBSE 2019]** をクリックします。
3. MapleMBSE 設定ファイル (ファイル拡張子 *.MSE* を持つ設定ファイル) の格納場所を参照し、設定ファイルを選択して **[開く]** をクリックします。

   ヒント : 設定ファイルが見つからない場合、**<Install Directory>\Example\MechSheet\MechSheet.tritest.MSE** からサンプル設定ファイルを開くことができます。ここで、**<Install Directory>** は **MapleMBSE 2019** のインストール場所です。

4. モデルファイルの場所を参照してモデルファイルを選択し、**[開く]** をクリックします。

   ヒント : モデルファイルが見つからない場合、**<Install Directory>\Example\MechSheet\MechSheet-sample1.tritest** からサンプルモデルファイルを開くことができます。ここで、**<Install Directory>** は **MapleMBSE 2019** のインストール場所です。

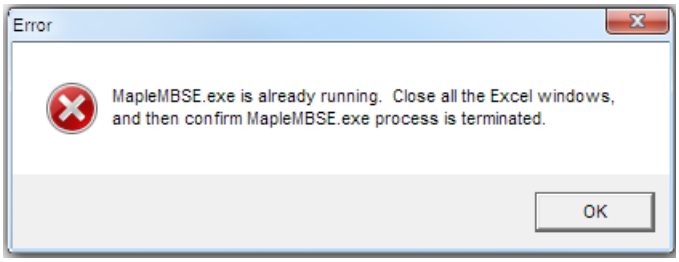
スプレッドシートにモデル情報が挿入された状態で Microsoft Excel が開きます (下図参照)。セルを修正したり行を挿入したりして、スプレッドシートのモデルを編集することができます。MapleMBSE でモデル情報をスプレッドシートに表示する方法と、さまざまな項目を編集する方法の詳細については、*MapleMBSE におけるモデルスプレッドシートの編集 [19ページ]*を参照してください。モデルを保存するには、この Excel ファイルを保存します。No Magic Teamwork Cloud Adapterを使用して開いたモデルを保存する方法については、*No Magic Teamwork Cloud Adapter の使用 [11ページ]*を参照してください。保存方法の詳細については、*モデルの保存 [9ページ]*を参照してください。



(unmapped)	Package	Class	Owner	Desc	Tag	(unmapp	Attribute	Type	(unmapped)
	Pkg1								
	Pkg1	Cls1	Pkg1	Desc1-1	Tag1-1				
	Pkg1	Cls1	Pkg1	Desc1-1	Tag1-1		Attr1	Type1	
	Pkg1	Cls1	Pkg1	Desc1-1	Tag1-1		Attr2	Type2	
	Pkg1	Cls1	Pkg1	Desc1-1	Tag1-1		Attr3	Type3	
	Pkg1	Cls2	Pkg1	Desc1-2	Tag1-2				
	Pkg1	Cls2	Pkg1	Desc1-2	Tag1-2		Attr1	Type1	
	Pkg1	Cls2	Pkg1	Desc1-2	Tag1-2		Attr2	Type2	
	Pkg1	Cls3	Pkg1	Desc1-3	Tag1-3				
	Pkg1	Cls3	Pkg1	Desc1-3	Tag1-3		Attr1	Type1	
	Pkg1	Cls3	Pkg1	Desc1-3	Tag1-3		Attr2	Type2	
	Pkg1	Cls4	Pkg1	Desc1-4	Tag1-4				
	Pkg1	Cls4	Pkg1	Desc1-4	Tag1-4		Attr1	Type1	
	Pkg1	Cls4	Pkg1	Desc1-4	Tag1-4		Attr2	Type2	
	Pkg1	Cls5	Pkg1	Desc1-5	Tag1-5				
	Pkg1	Cls5	Pkg1	Desc1-5	Tag1-5		Attr1	Type1	
	Pkg1	Cls5	Pkg1	Desc1-5	Tag1-5		Attr2	Type2	
	Pkg2								
	Pkg2	Cls6	Pkg2	Desc2-1	Tag2-1				
	Pkg2	Cls6	Pkg2	Desc2-1	Tag2-1		Attr1	Type1	
	Pkg2	Cls6	Pkg2	Desc2-1	Tag2-1		Attr2	Type2	
	Pkg2	Cls7	Pkg2	Desc2-2	Tag2-2				
	Pkg2	Cls7	Pkg2	Desc2-2	Tag2-2		Attr1	Type1	
	Pkg2	Cls7	Pkg2	Desc2-2	Tag2-2		Attr2	Type2	

注 : ファイルマネージャから設定ファイルをダブルクリックして MapleMBSE を開始することもできます。これにより設定ファイルの選択手順をスキップして、**[Open Model File]** ダイアログに直接アクセスできます。

注 : 同時に実行できる MapleMBSE のインスタンスは 1 つのみです。MapleMBSE でモデルをすでに開いているときに別のモデルを開こうとすると、以下のエラーが表示されます。開いているモデルを閉じてから、新しいモデルを開いてください。



コマンドラインからの起動

MapleMBSE をコマンドラインから起動するには、インストールディレクトリ内で MapleMBSE.exe コマンドを実行します。コマンドの文法は、以下のとおりです。

MapleMBSE.exe [/R <memsize>] [/L <logfile>] [/E <excel.exe>] [[/I | CONFIG(*.MSE)] [MODEL]] |

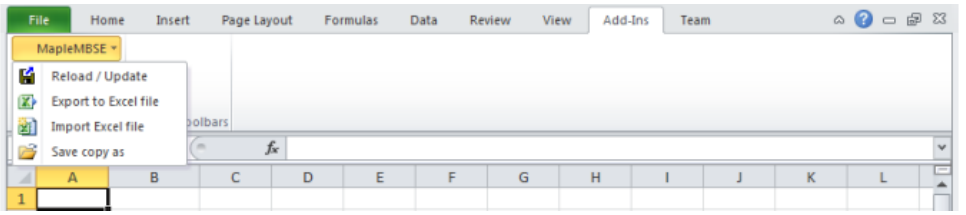
- **CONFIG** ファイル (*.MSE であること) と **MODEL** ファイルを指定すると、これらのファイルを使用して MapleMBSE が起動します。**MODEL** ファイルまたは **CONFIG** ファイルの指定を省略すると、入力を求めるダイアログが表示されます。
- **/I** を指定すると、MapleMBSE.exe では MSE ファイルを入力するように求めるダイアログを常に表示して、最初のスラッシュなしの引数をモデルファイルとみなします。
- **MODEL** には、ローカルファイルまたは URI を指定できます。現在、Teamwork Cloud の URI には以下の形式を使用できます。一部の項目を省略すると、それらの入力を求めるダイアログが表示されます。
  =
  twc://[SERVER:PORT][/PROJECT][?[username=USERNAME][&password=PASSWORD]]
- ファイル引数を指定しないと、MapleMBSE.exe が保存されているディレクトリ内の AddIns ディレクトリに MapleMBSE.xlsm が存在するかどうか確認されます。存在する場合は MapleMBSE が起動されます。
- **/R <memsize>** : (省略可能、デフォルト値は 905) Excel プロセスで Java VM 用に予約されるメモリアドレス空間 (MB 単位)。

- **/L<logfile>**: (省略可能、デフォルト値は"MapleMBSE_Launcher.log") ログファイル名。%USERPROFILE% を基準とする相対パスまたは絶対パス。
- **/E<excel.exe>** : (省略可能) EXCEL.EXE のパス。
  指定されていない場合は、レジストリエントリが使用されます。
  = HKEY_CLASSES_ROOT\CLSID\<"Excel.Application" の CLSID>\LocalServer32
  = HKEY_CLASSES_ROOT\CLSID\<"Excel.Application" の CLSID>\LocalServer
  = HKEY_CLASSES_ROOT\Wow6432Node\CLSID\<"Excel.Application" の CLSID>\LocalServer32
  = HKEY_CLASSES_ROOT\Wow6432Node\CLSID\<"Excel.Application" の CLSID>\LocalServer

1.3. MapleMBSE 固有の機能の使用

MapleMBSE 固有の機能を使用するには、**[アドイン]** メニューを選択し、**[MapleMBSE]** を展開します。以下の項目を使用できます。

- モデルスプレッドシートのリロード/更新
- Excel ファイルへの書き出し
- Excel ファイルのインポート
- モデルの複製を保存



モデルスプレッドシートのリロード/更新

MapleMBSE の外部でモデルに変更が加えられた場合、その変更を反映するには現在のモデルを更新します。

モデルをリロードするには、以下の手順に従います。

1. **[アドイン] > [MapleMBSE] > [Reload/Update]** を選択します。
2. **[Reload/Update]** ダイアログウィンドウで、**[OK]** をクリックします。**注：** モデルをリロードすると、保存されていない変更は破棄されます。
3. モデルがモデルスプレッドシートにリロードされます。

注：MapleMBSE とその他のツールで同じモデルを同時に操作することはサポートされていません。MapleMBSE で変更中のモデルを別のツールで変更することは推奨されません。

Excel ファイルのエクスポートとインポート

[アドイン] > [MapleMBSE] > [Excelファイルへの書き出し] を選択して、現在のワークシートを Excel ファイルに保存します。

[アドイン] > [MapleMBSE] > [Excelファイルのインポート] を選択して、現在開いているモデルに Excel ファイルを読み込みます。

現在のモデルのコピーを保存

現在のモデルを別のモデルファイルに保存し、現在のモデルファイルに変更を加えないようにすることができます。この機能により、モデルのスナップショットを作成することができます。メモリ内モデルは引き続き元のモデルファイルと関連付けられます。スナップショット作成後に加えられた変更はすべて元のモデルにのみ保存され、スナップショットには保存されません。

現在のモデルのコピーを保存するには、以下の手順に従います。

1. **[アドイン] > [MapleMBSE] > [モデルの複製を保存]** を選択します。
2. ファイル名と保存場所を指定します。
3. **[保存]** をクリックします。

1.4. モデルの保存

注：No Magic Teamwork Cloud Adapter を使用して開いたモデルを保存する方法については、*No Magic Teamwork Cloud Adapter の使用 [11ページ]*を参照してください。

モデルデータを保存する方法は複数あります。

- **メモリ内モデルをモデルファイルに保存**
  スプレッドシートで加えられた変更はメモリ内モデルに対して実行されます。メモリ内モデルの変更をモデルファイルに保存するには、Excel で通常使用する保存方法のいずれかを使用します。

- **[ファイル] > [上書き保存]** を選択
- **[Ctrl]** + **[S]** キーを押下
- ウィンドウの上部にある**保存**アイコンをクリック

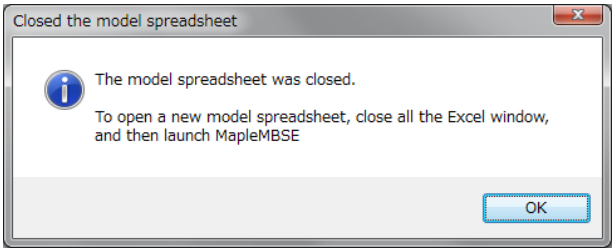
- **メモリ内モデルを別のモデルファイルとして保存**
  メモリ内モデルを別のモデルに保存するには、**[ファイル] > [名前を付けて保存]** を選択し、ファイル名と保存場所を指定して、**[保存]** をクリックします。メモリ内モデルがこの新しいモデルファイルにリンクされます。以降の保存操作はこの新しいファイルに対して実行されます。

- **モデルのコピーを保存**
  モデルのコピーを保存して、メモリ内モデルと元のファイルとのリンクを維持するには、*現在のモデルのコピーを保存 [9ページ]*を参照してください。

- **データを Excel ファイルとして保存**
  モデルデータを Excel ワークブックファイルに保存するには、**[アドイン] > [MapleMBSE] > [Excel ファイルへの書き出し]**を選択します。

1.5. MapleMBSE の終了

モデルを編集しているワークブックを閉じると、MapleMBSE も終了します。ほかに開いているワークブックがない場合は、Excel も終了します。以下のダイアログが表示され、MapleMBSE を再度使用する前に、実行中の Excel のインスタンスをすべて終了するよう求められます。



MapleMBSE のすべての Excel ウィンドウを閉じる必要があります。MapleMBSE と関係のない Excel ウィンドウを閉じる場合のルールを以下に示します。

- MapleMBSE を起動する前に開いた Excel ワークブックは開いたままにできます。
- MapleMBSE を起動した後に開いた Excel ワークブックは閉じる必要があります。

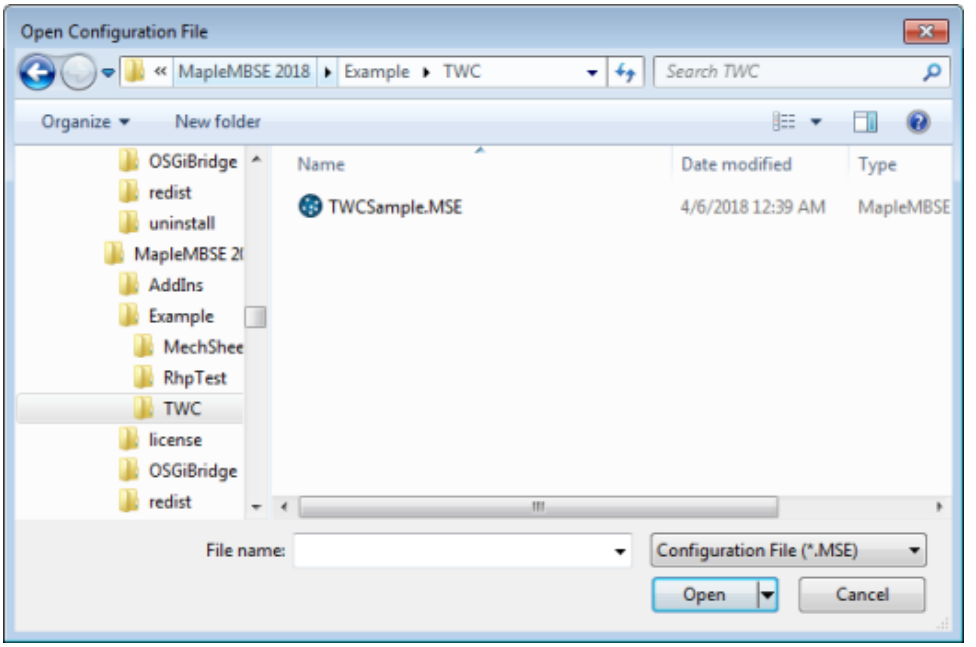
1.6. No Magic Teamwork Cloud Adapter の使用

MapleMBSE で Teamwork Cloud (TWCloud) Adapter を使用すると、モデルスプレッドシートでモデルに加えた変更を TWCloud リポジトリに保存できます。

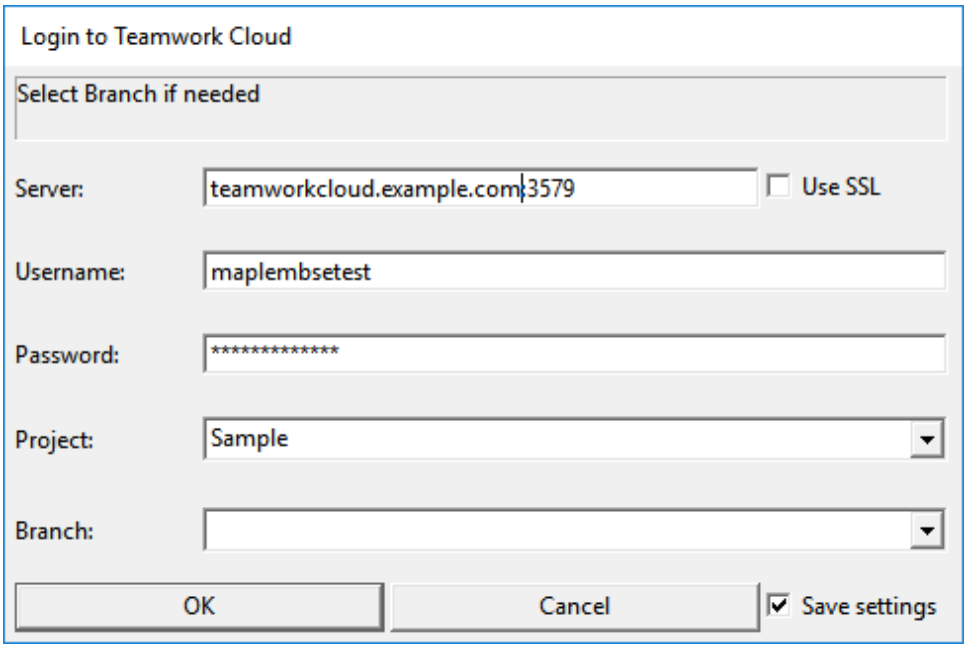
MapleMBSE での TWCloud モデルの起動

MapleMBSE で TWCloud モデルを開くには、以下の手順に従います。

1. **[スタート] > [すべてのプログラム] > [MapleMBSE 2019]** から、**[Open Teamwork Cloud Model]** を選択します。
2. TWCloud 設定ファイルを選択します。



3. **[Login to Teamwork Cloud]** ダイアログウィンドウで、TWCloud のプロジェクト名、ユーザ名、パスワード、サーバ、およびプロジェクトが格納されているブランチを入力します。TWCloud と MapleMBSE 間で SSL 接続を使用するには、**[Use SSL]** をオンにします。詳細については、*SSL 証明書の設定 [13ページ]*を参照してください。[Save settings] をオンにすると、成功したログイン情報 (パスワード以外) が保存され、次回ログイン時にその情報が自動的に入力されます。



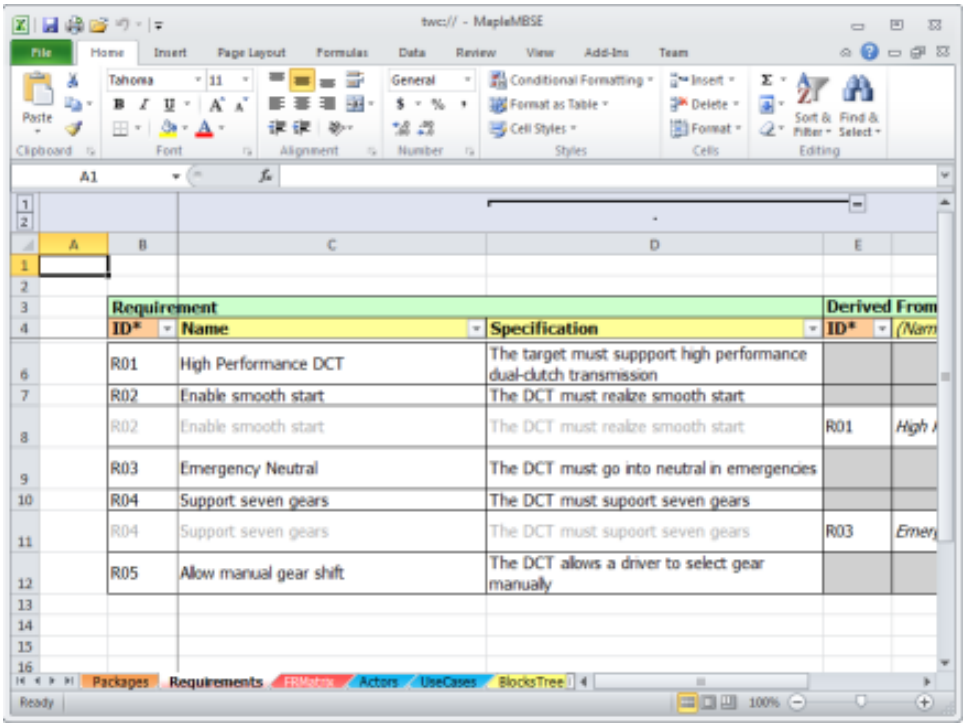
4. **[OK]** をクリックします。

SSL 証明書の設定

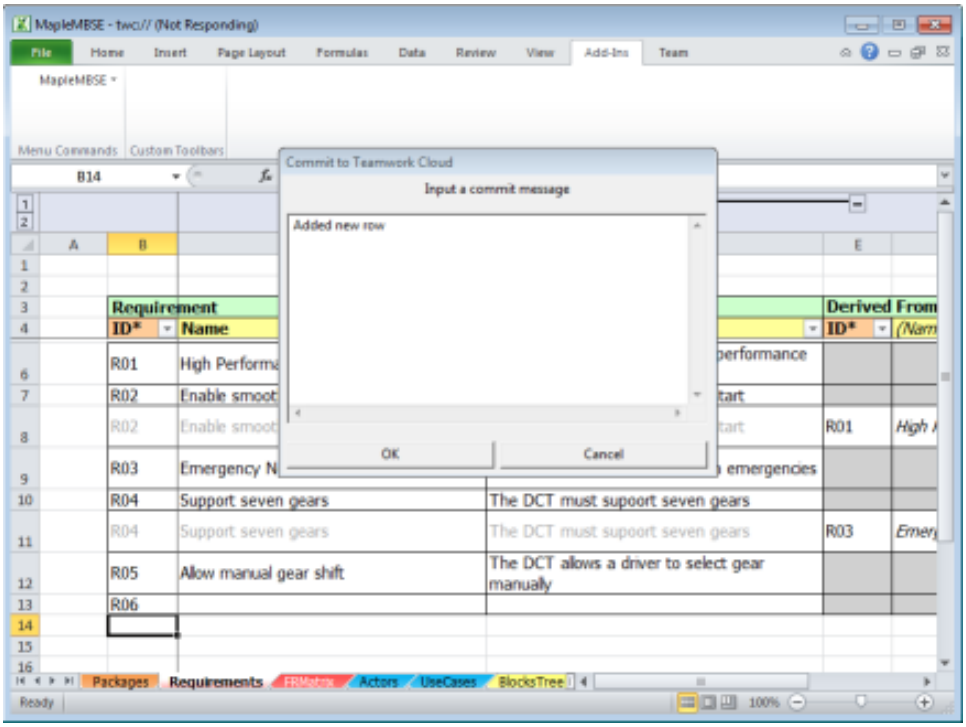
SSL 接続の場合は、JKS および PASS ファイルを **<MAPLEMBSE-INSTALLATION-DIRECOTRY>**\certs に指定する必要があります。デフォルトで、TWC アダプタでは cert.jsk と cert.pass が使用されます。これを設定するには、TWC URI に「*ssl*」パラメータを指定します。たとえば、*MapleMBSE TWCSample.MSE twc://?ssl=cert2* を起動すると、cert2.jks と cert2.pass が SSL 接続に使用されます。

モデルの編集と変更の保存

1. MapleMBSE でモデルを変更します。

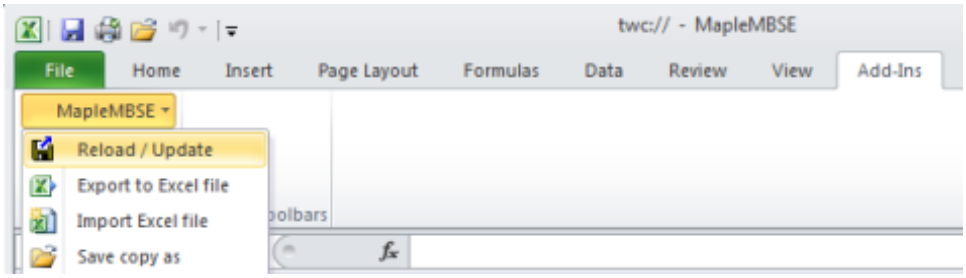


2. 変更を (ローカルの) スナップショットに保存するには、**保存** を 1 回クリックします。
3. TWCloud リポジトリに保存するには、**保存** をダブルクリックします。**[Commit to Teamwork Cloud]** ダイアログが表示されます。

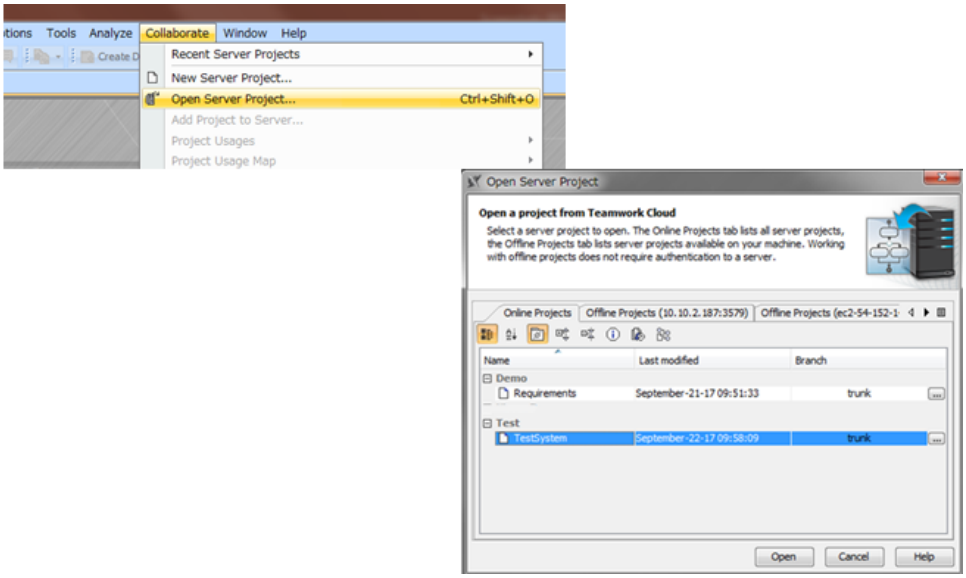


4. コミットメッセージを入力し、**[OK]** をクリックします。変更が TWCloud リポジトリにコミットされ、ほかのユーザと共有されます。

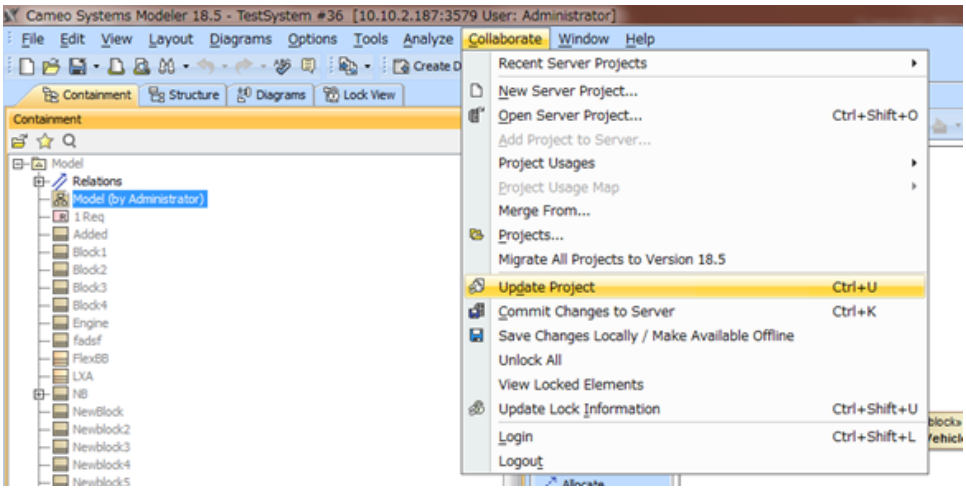
5. MapleMBSE でモデルをリロード/更新します。



6. MagicDraw™/Cameo Systems Modeler™ でサーバプロジェクトを開きます。



7. MagicDraw/Cameo Systems Modeler でプロジェクトを更新します。



モデルの変更の自動保存

MapleMBSE の TWCloud Adapter では、モデルの変更のスナップショットが自動的に作成されます。MapleMBSE がクラッシュした場合、このスナップショットを使用して回復します。

MapleMBSE は定期的 (10 分ごと) に 1 分間の無操作状態を確認します。このあいだにスナップショットが自動的に作成されます。

クラッシュ後のモデルのロード

クラッシュ後にロードされるモデルリソースは、クラッシュ直前に実行された直近の保存操作または同期操作によって異なります。

たとえば、クラッシュ直前の操作が Teamwork Cloud への変更のコミットであった場合は、Teamwork Cloud に保存されているスナップショットがロードされます。

一方、直前の操作がローカルドライブへの保存であった場合は、このローカルスナップショットがロードされます。

さらに、MapleMBSE がクラッシュする直前に自動保存が開始された場合、クラッシュから回復した後にダイアログウィンドウが表示され、ロードするスナップショットをユーザが選択できます。

第2章 MapleMBSE におけるモデルスプレッドシートの編集

この章の内容は以下のとおりです。

- *ワークブックとレイアウト [19ページ]*
- *モデル要素の追加 [25ページ]*
- *レコードのソート [35ページ]*
- *モデル要素の削除 [37ページ]*
- *モデル要素の変更 [39ページ]*

2.1. ワークブックとレイアウト

モデルのワークブックの内容は、MSE設定ファイルによって管理されます。設定ファイルは、スプレッドシートに表示するモデルデータおよびこのデータの表示方法について指定します。この章の以降の記述では、データの表示方法、およびこの表示方法がスプレッドシートに対するユーザインタラクションに及ぼす影響について説明しています。表示するモデルデータの管理ルールについての詳細は、『**MapleMBSE Configuration Language Guide**』を参照してください。

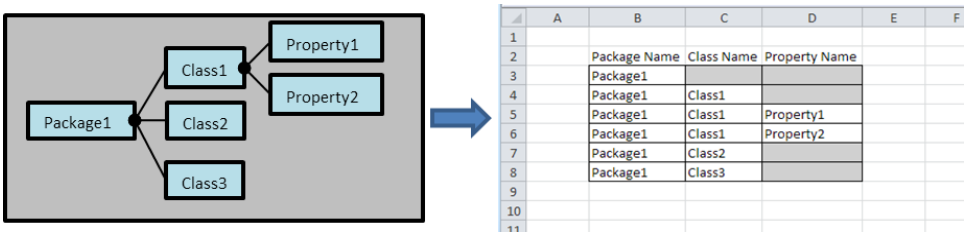
設定ファイルは、ワークブックに含まれるスプレッドシートの数、およびスプレッドシート上に配置される SyncView の位置と配置方法を指定します。それらの仕様に変更をもたらす可能性のあるユーザインタラクションはいずれもエラーとなり、操作は行われません。特に、以下の情報は、MSE設定がワークブックに対して指定しているものです。

- スプレッドシート数
- スプレッドシート名
- スプレッドシートの順序

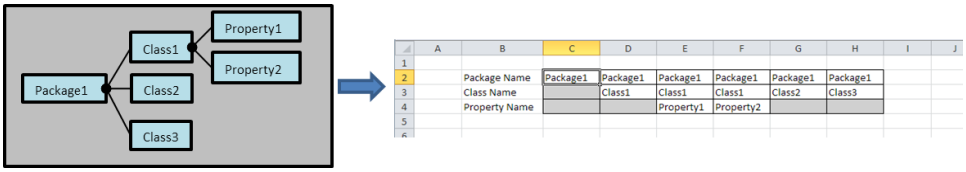
スプレッドシートを削除、名前を変更、または移動するとエラーになります。スプレッドシートの順序は、テンプレートファイルによっても管理されています。詳細については、『**MapleMBSE Configuration Language Guide**』を参照してください。

SyncViewについては、SyncView領域が始まるスプレッドシート内のセル、フィールド数 (*操作の概要 [21ページ]*を参照)、および SyncView の配置タイプを MSE 設定が指定します。スプレッドシート上で SyncView の位置を移動させる、またはフィールド数に干渉するなどの動作はすべて、MapleMBSE がブロックします。フィールド数に干渉する動作は、配置タイプによって異なります。SyncView の配置には、縦方向の表、横方向の表、および行列の 3 つのタイプがあります。

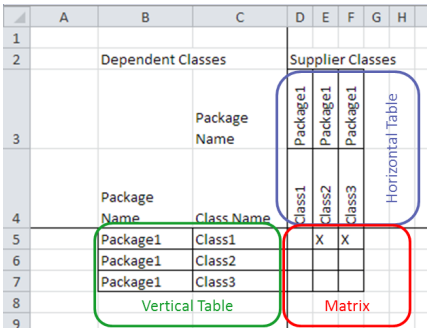
- **縦方向の表**
  縦方向の表は、要素が縦に並べられている表で、各行はモデル要素、各列はその要素の特定の属性を表します。例を下図に示します。MSE設定は、SyncViewが始まるセル (例の B3) を指定します。そのセルの前 (行 3 より上、または列 B より左) で列または行を挿入または削除すると、SyncViewが移動し、仕様との矛盾が生じるためエラーが発生します。MSE 設定は、縦方向の表の列数 (列 B から 3 列) も指定します。その領域 (B から D) で列を挿入または削除すると、エラーが発生します。行数は、要素数によって決まります。行を挿入または削除すると、要素が新規追加されるか、既存の要素が削除されます。



- **横方向の表**
  横方向の表は、要素が横に並べられている表で、各列はモデル要素、各行はその要素の特定の属性を表します。例を下図に示します。MSE設定は、SyncViewが始まるセル (例の C2) を指定します。縦方向の表と同様、そのセルの前 (行 2 より上、または列 C より左) で列または行を挿入または削除することはできません。横方向の表では、MSE 設定によって表の行数 (行 2 から 3 行) が指定されます。その領域 (行 2 から行 4) で行を挿入または削除すると、エラーが発生します。列数は、要素数に対応します。列を挿入または削除すると、要素が新規追加されるか、既存の要素が削除されます。



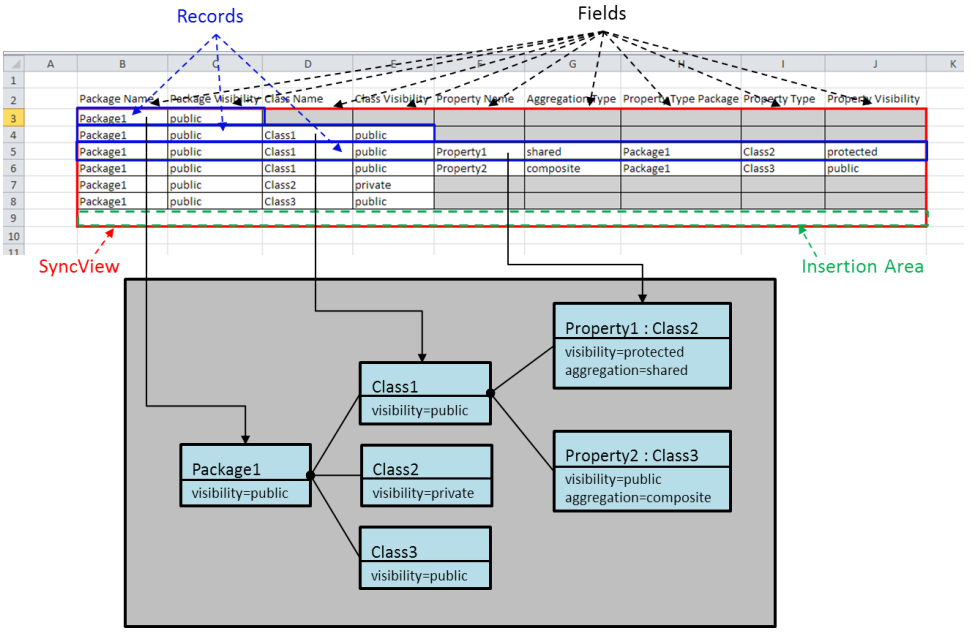
- **行列**
  行列は、縦方向の表および横方向の表の両方を組み合わせたもので、行列形式で 2 つの表の関係を示します。モデル要素間の依存関係を表すのに便利です。下図の例では、行列は、従属クラスのサプライヤクラスに対する依存を表しています。従属クラスは縦方向の表 (緑の枠線) に表され、サプライヤクラスは横方向の表 (青の枠線) に表されています。行列 (赤の枠) は、依存関係を示します。例では、Package1 の Class1 は、Package1 の Class2 および Package1 の Class3 に依存しています。MSE 設定が、行列部分が始まるセル (例の D5) を指定します。縦方向および横方向の表の位置は、そこから決まります。そのセルの前 (行 5 より上、または列 D より左) で行または列を挿入または削除することはできません。行列内の列数および行数は、縦方向および横方向の表でそれぞれ表される要素数によって決まります。行列領域 (行 5〜7、または列 D〜F) に影響を及ぼす部分で行または列を挿入または削除すると、縦方向または横方向の表でそれぞれ表される要素が追加または削除されます。



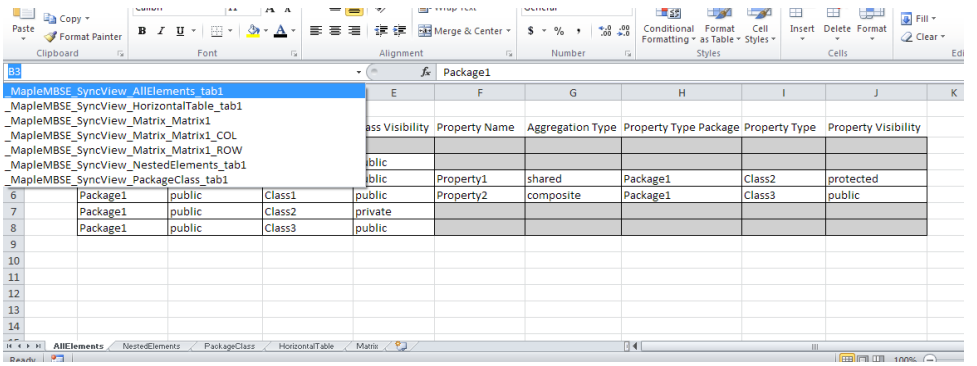
2.2. 操作の概要

MapleMBSE を使用して、SyncView 領域内に作成された表からモデルを編集できます。表内で行の挿入、削除、更新をすると、モデル要素がそれぞれ追加、削除、更新されます。

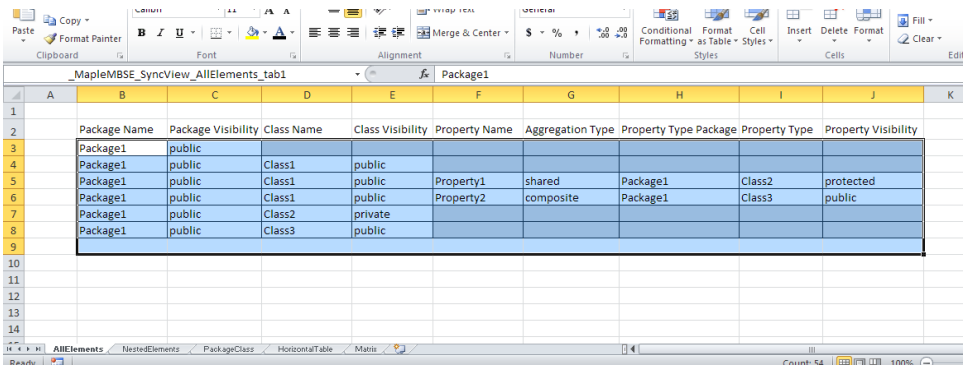
スプレッドシートでMapleMBSEが表示するデータの基本要素には、SyncView、レコード、およびフィールドがあります。スプレッドシート上でモデルデータを表す領域は、下図では赤の枠線で示されており、SyncView と呼ばれています。わかりやすくするため、この例ではSyncViewを縦方向の表とします(ワークブックとレイアウト *[19ページ]*の縦方向の表を参照)。スプレッドシートの SyncView 部分での操作のみがモデル上の操作に移動します。スプレッドシートのほかの部分での操作は、モデルに影響しません。SyncView の縦 (または横) 方向の表最後部の空の行 (または列) は、新しい要素を挿入するために使用され、挿入領域 (下図の緑の破線) と呼ばれます。各 SyncView は、モデル要素のセットに対応します。各要素は、レコードに対応し、縦方向の表の行 (下図の青の枠線) で表されます。各レコードは、フィールドの配列であり、行内のセルで表されます。各フィールドは、モデル要素のプロパティを表します。



ワークブックの SyncView 領域は、**名前ボックス** (数式バーの左側のボックス) で検索できます。下図を参照してください。**名前ボックス**でリストを展開すると、ワークブックのすべてのスプレッドシートのすべての SyncView 領域のリストが表示されます。名前には、MSE 設定で定義されるスプレッドシート名および SyncView 領域名が含まれます。



名前の 1 つを選択すると、対応するセルの範囲が表示されます (下図参照)。



以降、このセクションでは、スプレッドシートでの Excel 操作の概要を説明します。

可能な操作

この表では、SyncViewを縦方向の表として想定しています。横方向の表の場合、操作は行ではなく列に適用されます。

編集操作	説明
SyncView 表に新しい行を挿入	Excelのコンテキストメニューを使用して、表の任意の場所に新しい行を挿入します。
表の最下行直下のセルに入力することで新しい行を挿入	表のすぐ下 (かつ隣接) のセルに内容を入力し、[Enter] を押します。これにより、新しい行が表に追加され、SyncView領域が拡大します。

コピーしたセルを新しい行として挿入	コピーしたセルを挿入または貼り付けて(Excelのコンテキストメニューから、[コピーしたセルの挿入] 機能を使用)、表領域に行を挿入します。
表内の既存の行でセル内容を更新	SyncView 領域内で表のセル内容を更新します。
行の削除	SyncView 領域内で表の行を削除します。
SyncView 領域内で行をソート	表全体を選択してからソートすることで、表内で行を入れ替えます。

MapleMBSE エラーメッセージにつながる編集操作

下表に示す編集操作を行うと、MapleMBSE 設定ファイルで定義された SyncView 表の構造が破壊されるため、、エラーメッセージが生成され、操作が取り消されます。

編集操作	**説明**
スプレッドシート最上部に新しい行を挿入	スプレッドシート最上部、SyncView 表の上に新しい行を挿入します。
表の左側に列を挿入	スプレッドシートで、表の一番左の列の左側に列を挿入すると、エラーメッセージが生成されます。
複数の隣接しない矩形領域を更新	セルの内容をコピーし、表内の複数の領域を選択して貼り付けようとすると、エラーメッセージが生成されます。
挿入領域で行または列を削除	挿入領域で、行または列を削除します。
SyncView 領域の移動	表領域の場所を移動する編集操作を行います。

MapleMBSEとモデル要素間のマッピングに否定的な変化をもたらす操作

表内の行および列に編集を加える場合、以下の操作によってMapleMBSE設定ファイルで定義されたSyncView表のレイアウトが変更されることはありません。しかし、これらの操作を実行すると、モデルスプレッドシート内の表の行とモデル要素そのものとのあいだのマッピングが壊れるおそれがあります。これにより、後に予期せぬ挙動が起こる可能性があります。

編集操作	**説明**
SyncView領域内の表の一部のみをソート	SyncView内で選択された行、セルまたは列のみをソートします。
表に内容を挿入し、セルを上方向へシフト	表に内容を挿入し(表のセルで右クリックして[挿入]を選択など)、その後[上方向にシフト]または[下方向にシフト]のいずれかを選択します。
選択した表から内容を削除	表への挿入同様、内容削除後にセルを上または下方向へシフトします。

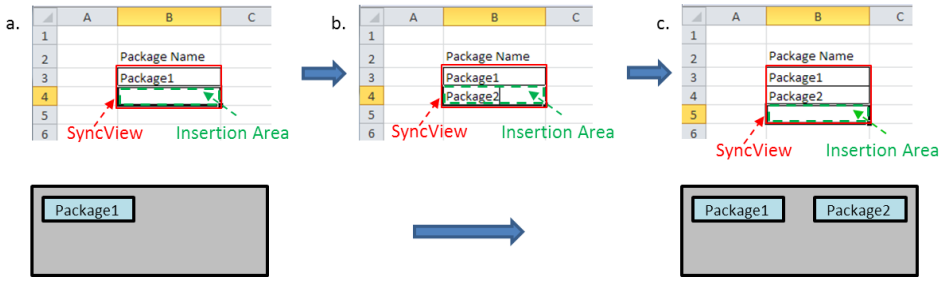
2.3. モデル要素の追加

要素の追加は、SyncView領域へレコードを追加することで実行されます。レコードの追加方法は、SyncViewのレイアウトにより異なります。レコードの追加は、縦方向の表では行の追加、横方向の表では列の追加に対応します。わかりやすくするため、以下の手順ではSyncViewを縦方向の表と想定します。縦方向の表へのレコードの追加は、以下の2つの方法で実行できます。

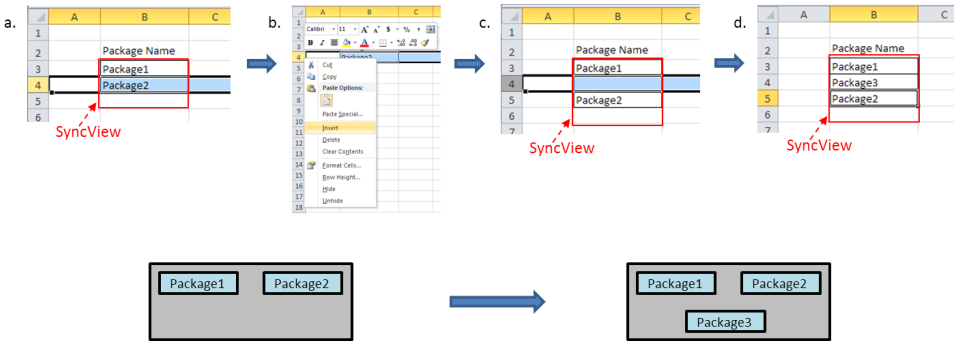
ヒント: この例の手順は、*UserGuide.MSE*設定ファイル、*AddingNewElements.uml*モデルの*Packages*スプレッドシートを使用して実行できます。これらのファイルは**<MapleMBSE>/Example/UserGuide**にあります。ここで、**<MapleMBSE>**はMapleMBSEのインストールディレクトリです。

1. 挿入領域に必要な情報を入力します。
   下図では、SyncViewは赤の枠線、挿入領域は緑の枠線で示されています(aのセルB4)。挿入領域(b)に「Package2」と入力し、[Enter]を押します。MapleMSBEがこの名前を持つ要素がほかに存在しないことを確認し、「Package2」という名前を持つ新しいパッケージ要素を追加します(c)。モデ

ルは、図の下部に表示されています。SyncView 領域が 1 行ずつ拡大します。挿入領域は、セル B5 です。



2. SyncView 領域に行を挿入し、必要な情報を入力します。
   下図では、SyncView 領域は赤の枠線で示されています (a)。たとえば、行 4 を選択し、コンテキストメニューで **[挿入]** を選択します (b)。行 4 の場所に 1 行挿入され、前の行は下方向にシフトしています。SyncView 領域が 1 行ずつ拡大します。新しいパッケージの名前「Package3」を入力します。MapleMBSE が、「Package3」という名前の新しい要素を作成します。



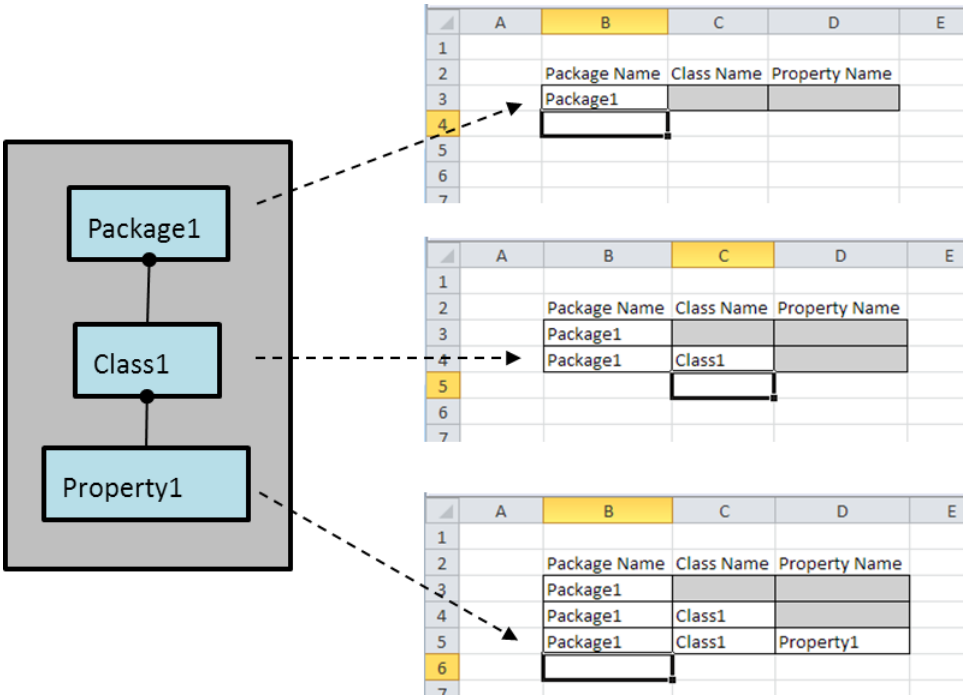
これらの変更は、MapleMBSE の内部メモリ内のモデルで実行されます。変更をモデルに反映し保存するには、ワークブックを保存します。詳細については、*モデルの保存 [9ページ]* を参照してください。

MapleMBSE では、新しい要素のソートが自動的に行われることはありません。ソートは手動で行われます。詳細については、*レコードのソート [35ページ]*を参照してください。レコードの追加は、同じモデル要素を参照するほかのスプレッドシートに反映されます。ほかのワークシートでは、追加されたレコードは常に最後の行に追加されます。

ネスト化された要素

モデルは通常、ツリー構造で表される多くの依存関係を持っています。表にこの構造を反映させるには、階段型の表を使用します。下図の例で考えます。図の左側のような構造を定義しようとしているとします。Package1 は、Property1 を含む Class1 を含んでいます。各要素は、表内の各レコードにより表されます。各要素を、Package1 から定義していきます。

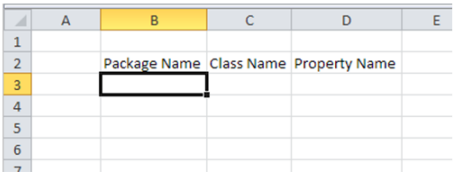
ヒント : この例は、UserGuide.MSE 設定ファイル、AddingNewElements.uml モデルの *NestedElements* スプレッドシートを使用して実行できます。これらのファイルは **<MapleMBSE>/Example/UserGuide** にあります。ここで、**<MapleMBSE>** は MapleMBSE のインストールディレクトリです。



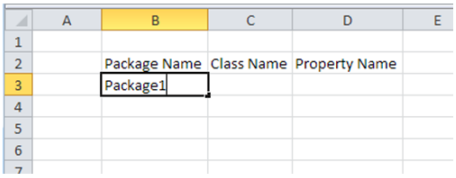
- *Package1 の定義*
  Package1 を定義するための手順を下図に示します。パッケージの定義に対応する列の挿入領域で (a)、「Package1」と入力し (b)、[Enter] を押します。[Enter] を押すと、メモリ内モデルにその名前を持つ要素がすでにあるかを MapleMBSE がチェックします。そのような要素がまだ存在しないことを確認

した場合は、その行が新しい要素を表します。1 行が 1 要素に対応するため、この行では要素内の追加要素を定義することはできません。MapleMBSE は、ネスト化された要素に対応するフィールドをグレーで表示することで、そのことを示します。

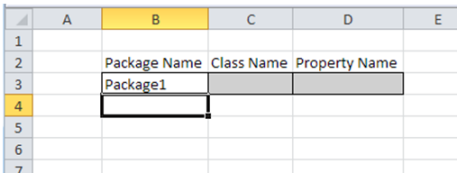
a.



b.



c.

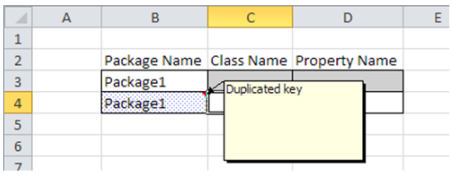


- *Class1 の定義*

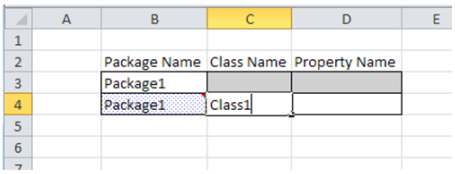
  Package1 の Class1 を定義するための手順を下図に示します。Class1 を定義するには、まず属するパッケージを指定する必要があります。パッケージに対応する列に「Package1」と入力し、右矢印キーを押します (a)。モデル内にこの名前を持つ要素がすでにあるか、MapleMBSE がチェックします。その要素が存在することを確認した場合は、このフィールドに対して「キーが重複しています」という警告を表示します。ほかのフィールドはまだ白く表示されているため、Package1 内部のネスト化された要素を定義できます。「Class1」と入力し (b)、[Enter] を押します。Package1 内部に Class1 がすでに存在するかを MapleMBSE が再検証し、存在しないことを確認すると、新しい要素を作成します。パッケージフィールドからの警告が解除されます (c)。1 つのレコードは 1 つの要素にのみ対応し、ここでこのレコードは Class1 を定義するため、同じ行で Class1 内部の要素を定義することはできません。MapleMBSE は、[Property

Name] の列をグレーで表示することで、そのことを示します。

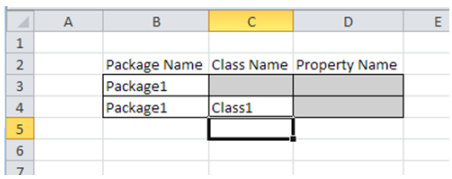
a.



b.

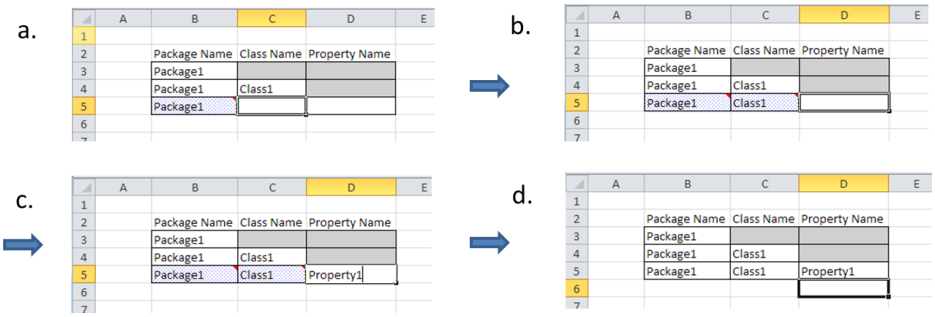


c.



- *Property1 の定義*
  Property1 を定義するための手順を下図に示します。Property1 を定義するには、まず属するパッケージおよびクラスを指定する必要があります。パッケージに対応する列に「Package1」と入力し、右矢印キーを押します (a)。Class1 を定義した際と同様に、MapleMBSE はその要素 Package1 がすでに存在することを確認し、そのセルに警告マークを追加します。「Class1」と入力し、右矢印キーを押します (b)。この時点で、Class1 は Package1 にすでに存在するため、MapleMBSE はそのセルに対しても「キーが重複しています]」の警告を表示します。プロパティフィールドは白く表示されており、Class1内部のプロパティを表す要素をここで定義できます。「Property1」と入力し (c)、[Enter] を押します。MapleMBSE は、Class1 内部に Property1 と呼ばれる既存のプロパティがないことを確認し、レコードは新しく固有な要素を表します。ほかの

フィールドからの警告は、すべて解除されます (d)。



代替グループ

モデルでは、異なるタイプの要素が同じ親に属することがあります。たとえば、パッケージには、クラスまたはほかのパッケージが含まれる場合があります。MSE 設定は、レコードに含まれるタイプ、つまりレコードによって定義される要素のタイプを指定します。*ネスト化された要素 [27ページ]* の例では、MSE 設定により、上位レベルで定義されたパッケージ、上位パッケージ内で定義されたクラス、クラス内で定義されたプロパティの 3 種類のレコードで定義することができました。上位パッケージ内のほかのパッケージ、またはほかのクラス内のクラスなどは、定義することができませんでした。ネスト化された同じレベルで異なる要素のタイプを定義可能にするには、代替グループがMSE設定で定義される必要があります。詳細については、『*MapleMBSE Configuration Language Guide*』を参照してください。ここでは、そのような設定を使用する場合について考えてみます。

ヒント : この例の手順は、UserGuide.MSE 設定ファイル、AddingNewElements.uml モデルの *AlternativeGroups* スプレッドシートを使用して実行できます。これらのファイルは **<MapleMBSE>/Example/UserGuide** にあります。ここで、**<MapleMBSE>** は MapleMBSE のインストールディレクトリです。

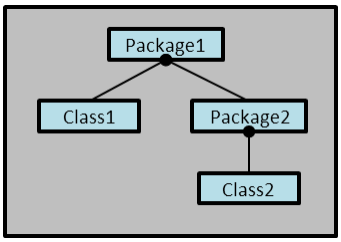
この例の MSE 設定により、SyncView の対応するフィールドを定義することで、以下の 4 つのタイプの要素を定義できます。

- 上位レベルパッケージ - **Package Name**
- 上位レベルパッケージ内のクラス - **Class Name**

- 上位レベルパッケージ内でネスト化されたパッケージ - **Nested Package Name**
- ネスト化されたパッケージ内のクラス - **Nested Class Name**

	A	B	C	D	E	F
1						
2		Package Name	Class Name	Nested Package Name	Nested Class Name	
3						
4						
5						
6						
7						
8						

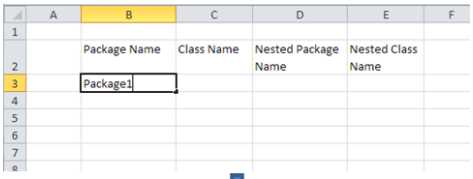
この例で定義しようとしているモデルを下図に示します。上位レベルパッケージ (Package1) には、クラスタイプ (Class1) 1 つ、およびパッケージタイプ (Package2) 1 つの、合わせて 2 つの要素が含まれています。ネスト化されたパッケージ (Package2) には、その中で定義された 1 つのクラス (Class2) があります。



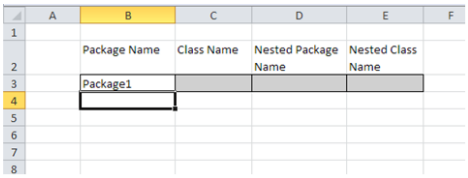
- *Package1 の定義*
  Package1 の定義プロセスは、*モデル要素の追加 [25ページ]* および *ネスト化された要素 [27ページ]* の例と同じです。挿入領域の [Package Name] 列に

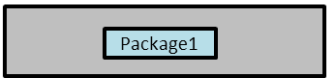
「Package1」と入力し、[Enter] を押します。

a.



b.





- *Class1 の定義*
  Class1 を定義するには、下図の手順に従います。新しい行 (行 4) の [Package Name] 列に「Package1」と入力し、[Enter] を押します (a)。*ネスト化された要素 [27ページ]* での *Class1 の定義*の例のように、Package1 を定義するセルに、重複キーであることを示す警告が追加されます。その例とは異なり、この行のほかのセルの背景は、濃いグレーに変わります。この色は、その行にはこれ以上定義できる要素がないことを示す薄いグレー (行 3) とは意味が異なります。濃いグレーは、そのフィールドが代替フィールドを表すことを示します。**[Class Name]** に「Class1」と入力し (b)、[Enter] を押します。Class1 という新しいクラスが Package1 に追加されます (c)。SyncView 領域のこの行の残りのセルの背景が薄いグレーに変わり、この行ではこれ以上定義できないことを

示します。

a.

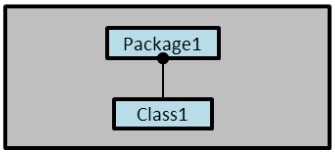
	A	B	C	D	E	F
1						
2		Package Name	Class Name	Nested Package Name	Nested Class Name	
3		Package1				
4		Package1				
5						
6						
7						
8						

b.

	A	B	C	D	E	F
1						
2		Package Name	Class Name	Nested Package Name	Nested Class Name	
3		Package1				
4		Package1	Class1			
5						
6						
7						
8						

c.

	A	B	C	D	E	F
1						
2		Package Name	Class Name	Nested Package Name	Nested Class Name	
3		Package1				
4		Package1	Class1			
5						
6						
7						
8						



- *Package2 の定義*
  Package1 内の Package2 を定義するには、下図の手順に従います。最初に、新しい行 (行 5) に「Package1」と入力し、[Enter] を押します (a)。重複キーの警告が Package1 のセルに追加されて、SyncView 領域内のこの行のほかのセルの背景が濃いグレーに変わり、ほかのセルが代替グループを表すことを示します。**[Class Name]** 列は空欄のままで、**[Nested Package Name]** 列に「Package2」と入力します (b)。[Enter] を押します (c)。新しいパッケージである Package2 がモデルに追加されます。Package2 内のネスト化された要素を表すセル (**[Nested Class Name]**) の背景が薄いグレーに変わり、このレコードが Package2 内のネスト化されたクラスを定義するために使用できないことを示します。代替 (**[Class Name]**) を表すセルの背景は、濃いグレーのままで

す。

a.

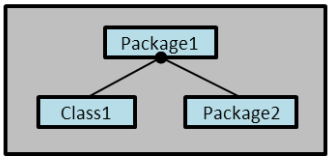
	Package Name	Class Name	Nested Package Name	Nested Class Name
3	Package1			
4	Package1	Class1		
5	Package1			

b.

	Package Name	Class Name	Nested Package Name	Nested Class Name
3	Package1			
4	Package1	Class1		
5	Package1		Package2	

c.

	Package Name	Class Name	Nested Package Name	Nested Class Name
3	Package1			
4	Package1	Class1		
5	Package1		Package2	



- *Class2 の定義*
  Package2 内の Class2 を定義するための手順を下図に示します。**[Package Name]** 列に「Package1」と入力し、[Enter] を押します (a)。**[Nested Package Name]** に「Package2」と入力し、[Enter] を押します (c)。Package1 内に Package2 というパッケージがすでに存在するため、重複キー警告が Package2 のセルに追加されます。**[Nested Class Name]** セルの背景は、濃いグレーのままです。**[Nested Class Name]** セルに「Class2」と入力し (d)、[Enter] を押します (e)。Class2 というクラスが、モデルの Package2 に追加されます。警告が解除されます。**[Class Name]** セルの背景は濃いグレーのままで、Package1 内の要素の代替タイプを表すことを示します。そこに入力した値はすべて無視

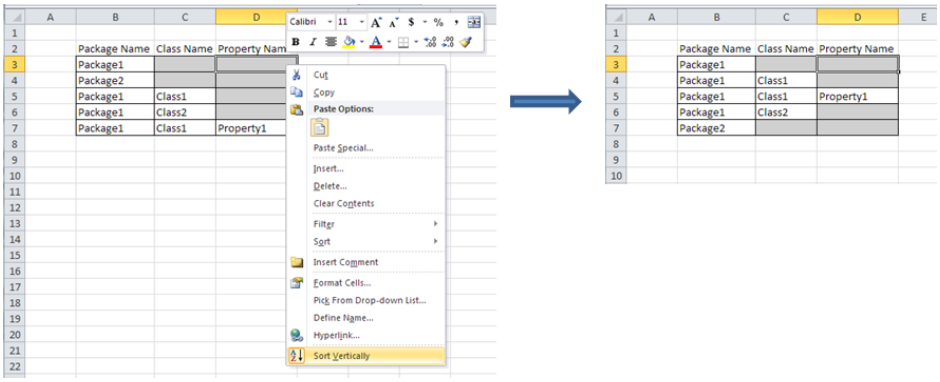
され、[Enter] を押すと削除されます。

2.4. レコードのソート

ソートの順序は、MSE設定ファイルで指定されています。モデルを開くと、設定された仕様に従いすべてのレコードがソートされます。新しいレコードが表に追加された場合、ソートが自動的に行われることはありません。レコードは手動でソートできます。レコードのソートは、モデルに影響することはありません。

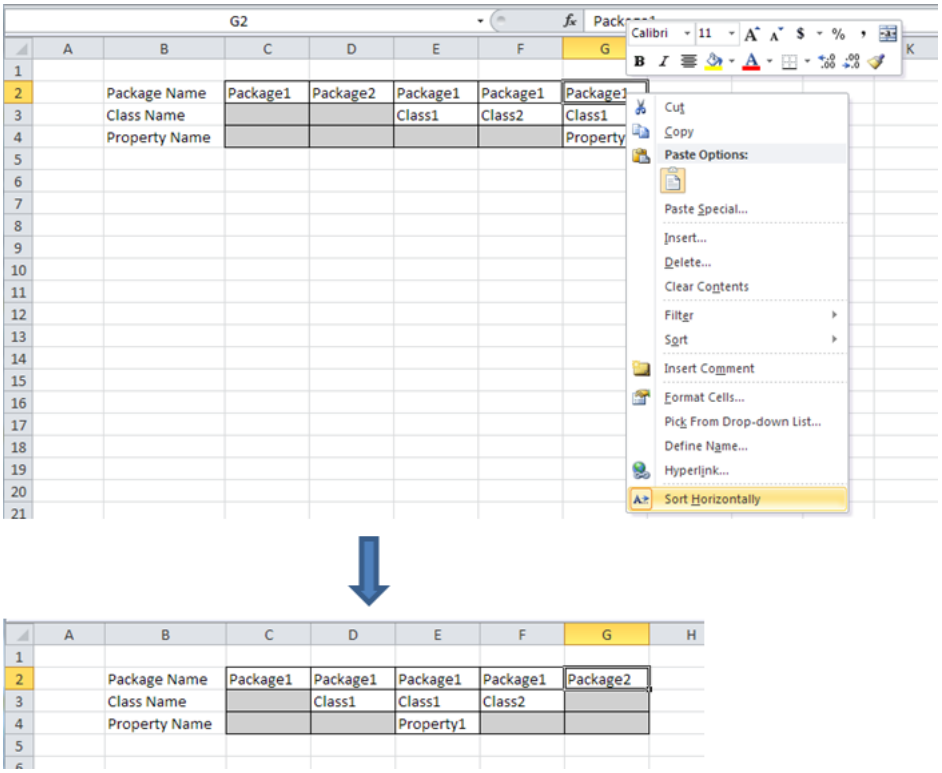
- **縦方向の表**
  縦方向の表でレコードを手動でソートするには、SyncView 領域の任意の場所

で右クリックし、**[Sort Vertically]** を選択します。

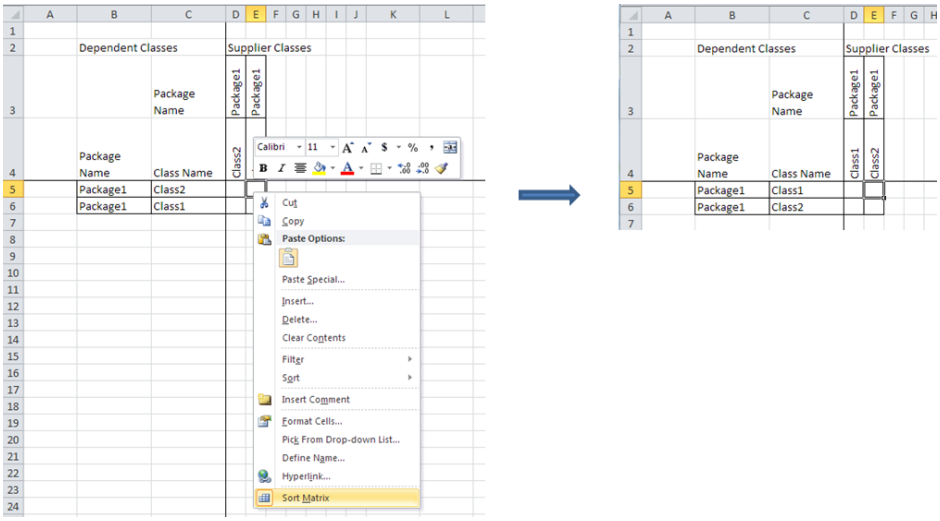


- **横方向の表**

  横方向の表でレコードをソートするには、SyncView 領域の任意の場所で右クリックし、**[Sort Horizontally]** を選択します。

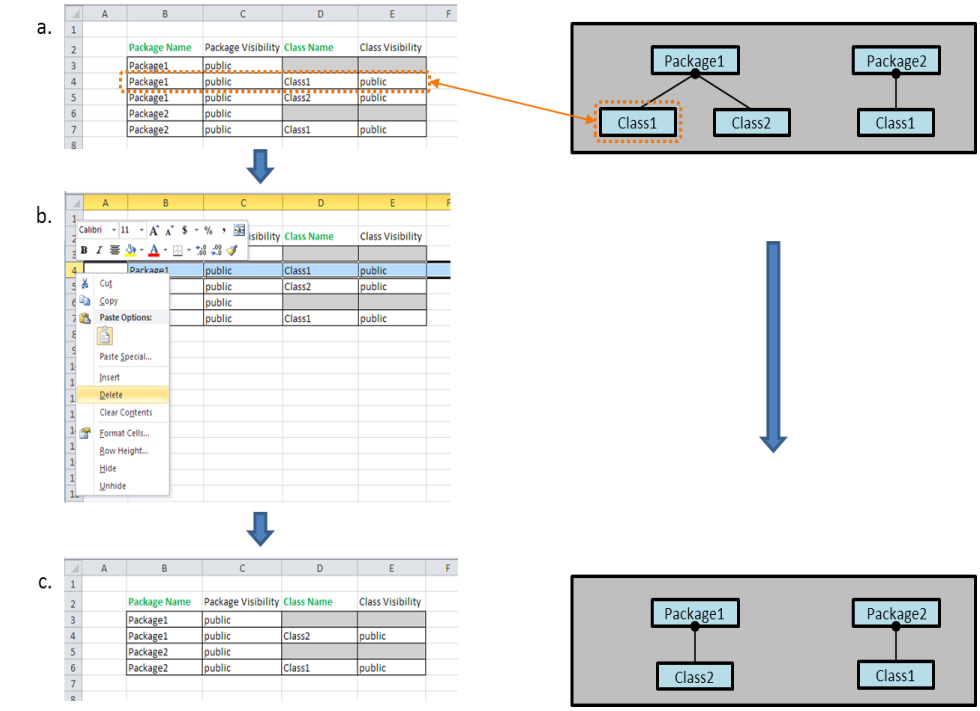


- **行列**
  行列でレコードをソートするには、SyncView 領域の任意の場所で右クリックし、**[Sort Matrix]** を選択します。

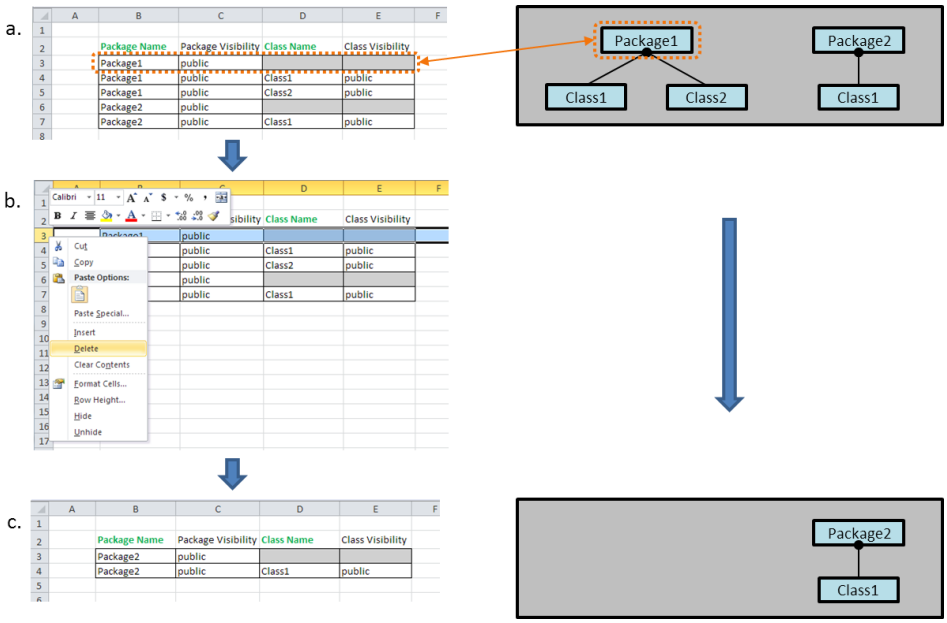


2.5. モデル要素の削除

SyncView 領域内の対応するレコードを削除することで、要素を削除できます。縦方向の表では行の削除、横方向の表では列の削除がこれに当てはまります。行列では、行または列を削除することで、対応する縦方向または横方向の表の行または列も削除され、よってその行または列により表される要素も削除されます。下図では、SyncView は縦方向の表です (a)。Package1 の Class1 を削除するには、この要素を表す行 (行 4) を削除します (b)。要素は、表およびモデルから削除されます (c)。



要素を削除すると、それに属するすべての要素も削除されます。たとえば、下図で Package1 を削除すると (a)、そこに属する Class1 および Class2 も削除されます。



参照つきレコードの削除

レコード内の参照フィールドは、モデルのほかの要素を参照しています。詳細については、*フィールドのタイプ[40ページ]*を参照してください。参照フィールドを含むレコードを削除しても、参照先の対応する要素は削除されません。

2.6. モデル要素の変更

モデル要素の追加 [25ページ] および *レコードのソート [35ページ]* で考えた例では、SyncViewsを使用して新しい要素を追加しましたが、要素のプロパティの編集や表示は行うことはできませんでした。このセクションでは、要素のさまざまなプロパティを操作できるようにする SyncViews のほかの特性を定義します。

操作の概要 [21ページ] で説明されているように、レコードは SyncViews でモデル要素を表し、各レコードは 1 つ以上のフィールドで構成されます。そのようなフィールドは、要素のさまざまなプロパティを表します。Excel スプレッドシートでフィールドを編集することで、モデル要素の対応するプロパティを変更できます。フィールドの編集がモデル要素に与える影響は、フィールドのタイプによって異なります。フィールドのタイプは、MSE設定によって設定されます。こ

こで、フィールドのさまざまなタイプと、それらが編集操作に及ぼす影響について説明します。

フィールドのタイプ

SyncViews のフィールドには、主に 3 つのタイプがあります。

- 属性フィールド
- 参照フィールド
- 非マッピングフィールド

属性フィールドは、文字列または数字などの、原始的なタイプのプロパティの値を指します。参照フィールドは、モデルにある別の要素などの、原始的ではないタイプのプロパティを表します。非マッピングフィールドは、要素のどのプロパティにもマッピングされていないフィールドです。これはモデルに影響を及ぼしませんが、コメントやほかのフィールドに関連する Excel 操作にとって有用な場合があります。詳細については、*非マッピングフィールド [58ページ]*を参照ししてください。

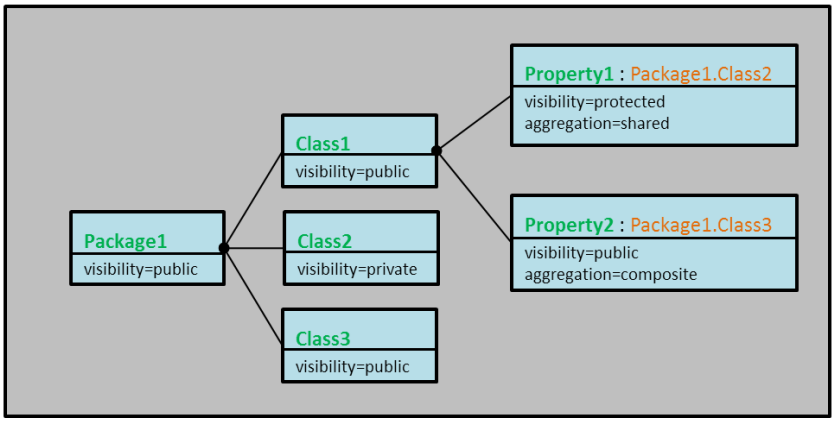
さらに、フィールドには、**キー**フィールドまたは**非キー**フィールドの 2 種類があります。**キー**フィールドは、モデル要素を固有に識別するために使用されます。したがって、同じキーフィールドを持つ複数のレコードは、同じ SyncView に存在することができません。1 つのレコードには、1 つ以上のキーフィールドが存在します。フィールドがキーであるかは、MSE設定で設定されます。**非キー**フィールドは、キーフィールドで定義される要素の特性を単に表すものです。

注: レコードキーフィールドと**参照キー**フィールドには、区別があります。レコードの中に複数の参照フィールドがある場合、そのうちのいくつかは、参照が指し示す要素を決定する参照キーフィールドである必要があります。レコードキーフィールドは、レコードを固有に決定するために使用されます。参照キーフィールドは、レコードキーフィールドでもある場合があります。

下図では、SyncView は縦方向の表です。各行はレコードであり、さまざまなフィールドは、さまざまな列で表されます。レコードキーフィールドは、緑の枠線で示され、[Package Name]、[Class Name]、および [Property Name] がこれに含まれます。これらは属性フィールドです。レコード非キー属性フィールドは、青の枠線で示され、[Package Visibility]、[Class Visibility]、[Property

Visibility]、および [Aggregation Type] がこれに含まれます。キー属性フィールド、および非キー属性フィールドはいずれも文字列です。クラスのプロパティには、ほかのクラスによって定義されたタイプが含まれている必要があります。言い換えると、プロパティのタイプは、モデルの別の要素です。そのため、プロパティのタイプを表すには、参照フィールドを使用する必要があります。モデル内でクラスを固有に定義するには、そのクラスが属するパッケージの名前とそれ自身の名前を指定する必要があります。オレンジ色の枠線で示される [Property Type Package] フィールド、および [Property Type] フィールドは、参照キーフィールドです。この例では、参照キーフィールドはレコードキーフィールドではありません (このようなフィールドは、レコードそのものではなく、レコードが参照する要素を決定するために必要なものです)。[Property Type Package]、および [Property Type] は、プロパティのタイプを定義するために十分な情報を提供します。しかしながら、同じレコード内でタイプに関する情報を表示したい (タイプの表示など) 場合があるかもしれません。そのため、非キー参照フィールドが使用されます。紫の枠線で示される [Type Visibility] のフィールドは、プロパティのタイプとして使用されるクラスの表示または非表示を表す非キー参照フィールドです。

	Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
3	Package1	public								
4	Package1	public	Class1	public						
5	Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	private
6	Package1	public	Class1	public	Property2	public	composite	Package1	Class3	public
7	Package1	public	Class2	private						
8	Package1	public	Class3	public						



モデル要素の追加 [25ページ] および *レコードのソート [35ページ]* で考えた例では、SyncViewsのレコードは、キー属性フィールドでのみ構成されていました。*モデル要素の削除 [37ページ]* の例では、SyncViewには、レコードキー属性フィールド ([Package Name] と [Class Name])、および非キー属性フィールド ([Package Visibility] と [Class Visibility]) がありました。

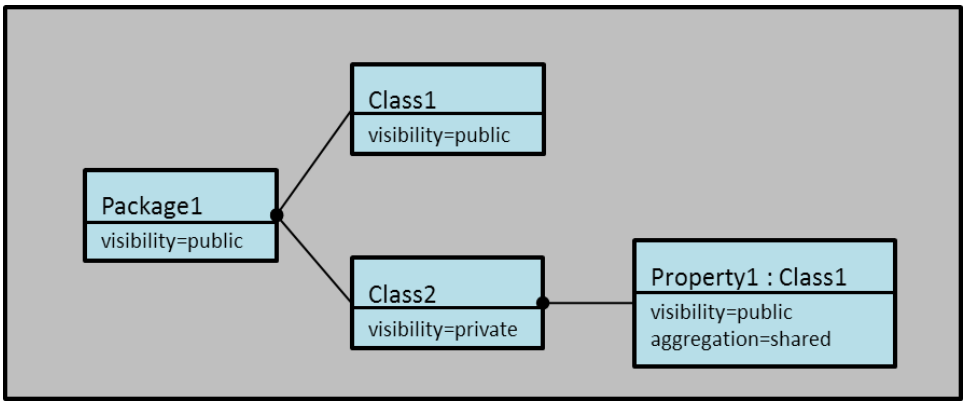
レコードの状態

SyncViewsでレコードを扱う際に重要となる別の定義に、レコードの状態があります。各レコードには、メモリにロードされたモデルへの関連付けの有無に応じて、**紐付けされている**状態、および**紐付けされていない**状態という 2 つの状態があります。新しいレコードが最初に作成されたときは、紐付けされていない状態であり、モデルには関連付けられていません。すべてのレコードキーフィールドが入力され、それらがまとめて固有な要素を表していることが確認されると、レコードは紐付けされている状態へと変わり、メモリ内のモデルに関連付けられます。これが既存の要素を表す場合、非キーフィールドにはメモリ内モデルからの情報が表示されます。要素が存在しない場合、モデル内で作成されます。モデルツールによっては、新しい要素にデフォルト値が設定される属性もあります。このような属性が非キーフィールドにマッピングされる場合、フィールドにはその値が表示されます。

要素追加時に複数のフィールドを使用する例

複数タイプのフィールドをともなって SyncView に要素を追加する例として、下図でモデルを作成してみます。SyncView の MSE 設定は、*フィールドのタイプ [40ページ]* の例のとおりフィールドのタイプを指定します。

ヒント : この例の手順は、UserGuide.MSE 設定ファイル、AddingNewElements.uml モデルの *AllElements* スプレッドシートを使用して実行できます。これらのファイルは **<MapleMBSE>/Example/UserGuide** にあります。ここで、**<MapleMBSE>** は MapleMBSE のインストールディレクトリです。



- *Package1 の定義*
  Package1 を定義するための手順を下図に示します。レコードキーフィールドに対応する列の名前は、緑色で表示されています。[Package Name] 列に「Package1」と入力します (a)。このレコードは紐付けられていません。[Enter] キーを押します。MapleMBSE は、レコードが固有の要素を定義していることを確認します。レコードが紐付けされている状態に変わります。MapleMBSE が、モデルに「Package1」という名前の新しいパッケージを作成します。デフォルトでは、UML の新しい要素は、表示設定が public に指定されます。ここで、レコードは紐付けられているため、[Package Visibility] フィールドの値は自動的に表示されます (b)。

a.

	A	B	C	D	E	F	G	H	I	J	K	L
1												
2		Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility	
3		Package1										
4												
5												
6												
7												
8												

b.

	A	B	C	D	E	F	G	H	I	J	K	L
1												
2		Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility	
3		Package1	public									
4												
5												
6												
7												
8												

- *Class1 の定義*
  Package1 内の Class1 を定義するための手順を下図に示します。Class1 を定義するには、[Package Name] および [Class Name] のキーフィールドに入力するのみです (a)。[Package Name] 列に「Package1」と入力すると、MapleMBSE

が、この名前を持つ要素がすでに存在することを確認するため、レコードは紐付けされていない状態のままとなります。[Package Visibility] は、表示されないままです。[Class Name] 列に「Class1」と入力し、[Enter] を押します。ここでMapleMBSEは、レコードが固有な要素を表していることを確認し、レコードをモデルに紐付けます。そして、Package1 に Class1 という新しいクラスを作成します。デフォルトでは、表示は public に設定されます。[Package Visibility] および [Class Visibility] の非キーフィールドが表示されます (b)。

a.

	Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
3	Package1	public								
4	Package1		Class1							

b.

	Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
3	Package1	public								
4	Package1	public	Class1	public						

- *Class2 の定義*
  Package1 内の Class2 を定義するための最初の手順は、Class1 を定義すすると きの手順と同じです (下図の a および b)。最後のステップは、Class2 の表示の

デフォルト値を private に変更することです (c)。

a.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1		Class2							

b.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	public						

c.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	private						

- *Property1 の定義*
  Class2 内の Property1 を定義するための手順を下の 2 つの図に示します。最初に、キーフィールドにのみ入力します。パッケージ名とクラス名を入力すると、これらの名前を持つパッケージとクラスがすでに存在するため、レコードは紐付けされないままになります。非キーフィールドは、表示されないままです。[Property Name] 列に「Property1」と入力し (a)、[Enter] を押します。ここで、レコードは新しい固有の要素を表すため、MapleMBSE はそのレコードをモデルに紐付け、新しい要素を作成します。非キーフィールドが表示されます (b)。既存の要素 (Package1 および Class2) の表示または非表示を表す値は、これらの要素に対して定義された値 (それぞれ public および private) から取得されます。Property1 の表示は、デフォルトでは public に指定されます。[Property Type Package] 列および [Property Type] 列にはデフォルト値がありません。これらは、既存のクラスに対して、(対応する設定ファイルで) 参照として定義されます。フィールドは空のため、参照先はなく、「参照先が見つかりません」という警告が [Property Type Package] (c) および [Property Type] フィールドに対して表示されます。

a.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	private						
Package1		Class2		Property1					

b.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	private						
Package1	public	Class2	private	Property1	public	none			

c.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	private						
Package1	public	Class2	private	Property1	public	none			

No reference target

タイプを定義するには、プロパティがインスタンスを作成するクラスの完全な名前を指定する必要があります。この完全な名前には、クラスが属するパッケージの名前、およびそのクラスそのものの名前が含まれます (d)。パッケージ名を指定するだけではクラス参照は処理されないため、警告が解除されないことに注意してください。クラス名を入力し、[Enter] を押した後でのみ、参照が処理され、警告が解除されます (e)。

注 : 参照先はモデル内にすでに存在する必要があります。この行は、参照される要素の作成用ではありません。

最後に、[Aggregation Type] を、そのデフォルト値 (none) から shared に変更

します (f)。

d.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	private						
Package1	public	Class2	private	Property1	public	none	Package1	Class1	

e.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	private						
Package1	public	Class2	private	Property1	public	none	Package1	Class1	public

f.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class2	private						
Package1	public	Class2	private	Property1	public	shared	Package1	Class1	public

紐付けされていないレコードの属性フィールドの変更

紐付けされている、および紐付けされていないレコードの定義については、*レコードの状態 [42ページ]* を参照してください。

- *キー属性フィールドの変更*
  モデルに紐付けされる前に新しいレコードが定義されると、キーフィールドで値が入力および変更されることで、そのレコードが表す新しいモデル要素が指定されます。これは、要素追加の例で行った操作です。
- *非キー属性フィールドの変更*
  レコードが紐付けされていない状態で値が非キーフィールドに入力された場合、フィールドが既存の要素を表すか、またはそのレコードが定義する新しい要素を表すかによって、その結果は異なります。

- *既存の要素を表す場合*
  入力された値は無視され、要素の既存の値で上書きされます。下図の例では、Package1 内で Class1 を追加していますが、Package1 はすでに存在しており、表示は public に設定されています。Class1 を定義する際、レコードが紐付け

されていない状態で Package1 の表示を private に設定します (a)。クラス名の入力が終わり [Enter] を押すと、レコードが紐付けされ、Package1 の表示はモデルで定義された値 (public) に設定されます。

a.

	A	B	C	D	E	F
1						
2		Package Name	Package Visibility	Class Name	Class Visibility	
3		Package1	public			
4		Package1	private	Class1		
5						
6						

b.

	A		C	D	E	F
1						
2		Package Name	Package Visibility	Class Name	Class Visibility	
3		Package1	public			
4		Package1	public	Class1	public	
5						
6						

- *新しい要素を表す場合*

  入力された値は、要素が作成されると、その新しい要素の対応するプロパティに割り当てられます。下図の例では、Class1 を追加する際、その表示設定はすべてのキーフィールドが要素を固有に定義する前に入力されており、レコードは紐付けされていません (a)。クラス名のキーフィールドが入力されます (b)。レコードがモデルに紐付けられます。Package1 の表示設定は、Package1 の定義に基づき決定されます。新しいクラス (Class1) が Package1 で作成され、そ

の表示は private に設定されます。

a.

	A	B	C	D	E	F
1						
2		Package Name	Package Visibility	Class Name	Class Visibility	
3		Package1	public			
4		Package1			private	
5						

b.

	A	B	C	D	E	F
1						
2		Package Name	Package Visibility	Class Name	Class Visibility	
3		Package1	public			
4		Package1		Class1	private	
5						

c.

	A	B	C	D	E	F
1						
2		Package Name	Package Visibility	Class Name	Class Visibility	
3		Package1	public			
4		Package1	public	Class1	private	
5						

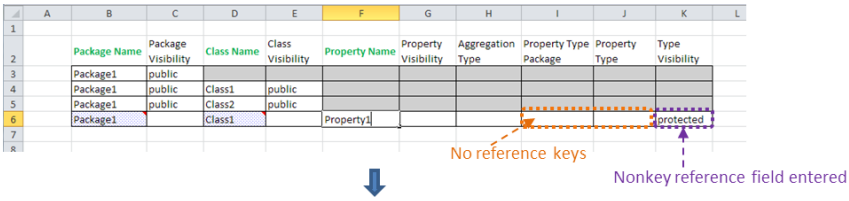
紐付けされていないレコードの参照フィールドの変更

紐付けされている、および紐付けされていないレコードの定義については、*レコードの状態 [42ページ]* を参照してください。

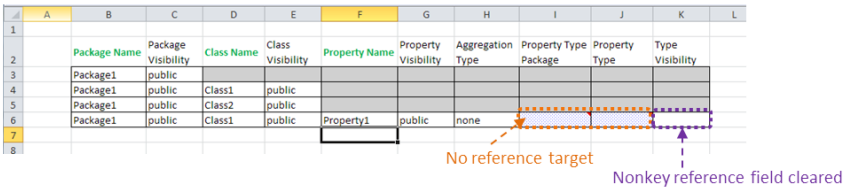
- *参照キーフィールドの変更*
  新しいレコードを追加するときは、参照キーフィールドを既存の要素に設定する必要があります。参照フィールドは、新しい要素の作成用ではありません。参照先を定義するキーが複数ある場合、それらのキーすべてを定義する必要があり、またそれらのキーは、モデル内にすでに存在する要素を指し示す必要があります。*要素追加時に複数のフィールドを使用する例 [42ページ]* の *Property1 の定義* の手順を参照してください。
- *参照非キーフィールドの変更*
  紐付けされていないレコードの非キー参照フィールドに値が入力された場合、参照キーが定義されているかによって、その結果は異なります。

- *参照キーが定義済みでない場合*
  キー参照フィールドが定義されていない場合、紐付けされていないレコードの非キー参照フィールドに入力された値 (下図 a の紫の枠線) は無視され、レコードが紐付けされると削除されます (b)。

a.

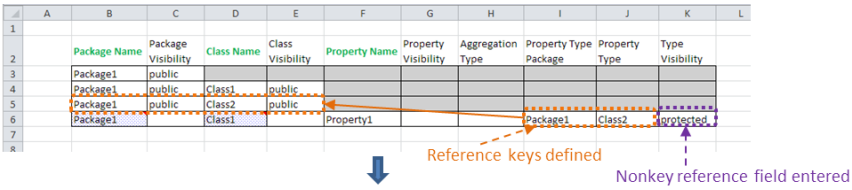


b.

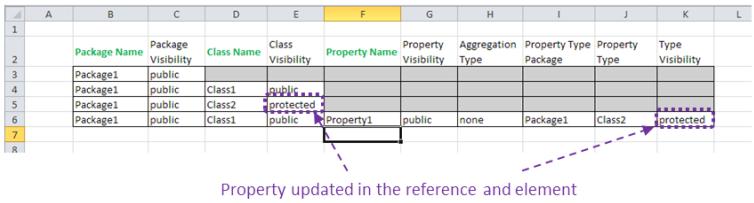


- *参照キーが定義済みの場合*
  参照キーが定義され、既存のレコードを参照すると (下図 a のオレンジ色の枠線)、紐付けされていないレコードの非キー参照フィールドに入力された値 (下図 a の紫の枠線) は、レコードが紐付けられると参照される要素に適用されます。同じプロパティに紐付けられるほかのフィールドや SyncViews も更新されます (b)。

a.



b.

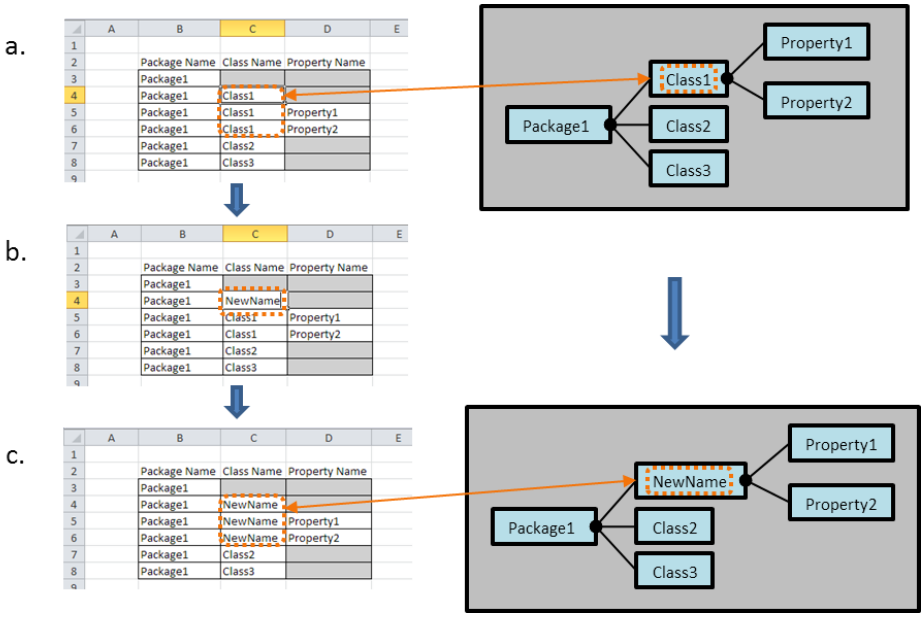


紐付けされたレコードの属性フィールドの変更

紐付けされている、および紐付けされていないレコードの定義については、*レコードの状態 [42ページ]* を参照してください。

紐付けられたレコードは、モデル内の要素にリンクされます。紐付けられたレコードの属性フィールドの値を変更すると、要素の対応するプロパティが変更されます。ほかのレコードやSyncView内にその要素と同じ属性を示すほかのフィールドがある場合、これらについても更新されます。

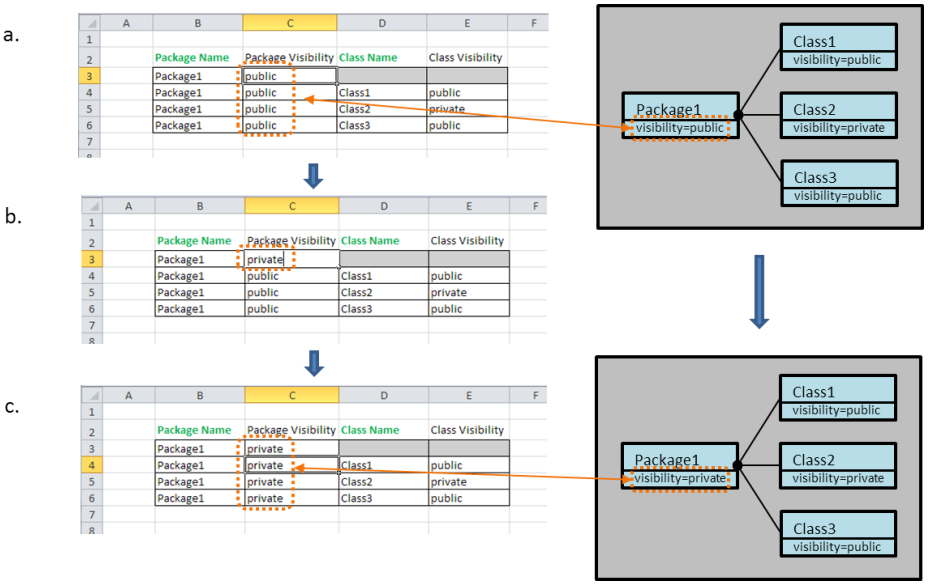
- *キー属性フィールドの変更*
  キー属性が変更されると、MapleMBSE は、レコードのキーフィールドの新しい組み合わせが固有であるかを最初に確認します。そうである場合、新しい値はモデル および SyncView のプロパティに割り当てられます。



注: MapleMBSE は、変更が実行されたレコードのキーの組み合わせが固有であるかについてのみ確認します。たとえば、上の例で、Property1 (行 5) を定義するレコードの Class1 を Class2 に変更した場合 (b)、レコードのキーフィールドは Package1-Class2-Property1 になります。そのようなキーを持つ要素はなく、よって変更は適用されます。その後、変更はモデルに伝搬し、Class1 は、Class2 がすでに存在する場所で、Class2 に名前が変更されます。MapleMBSE

では、この競合を検出できません。モデルを保存し、対応するツールで開いた際、このツールが検証を行い、競合を検出します。このため、キー属性の変更が必要な場合は、上の例のように、キーが最終キーフィールドとなっているレコードで行うことが推奨されます。

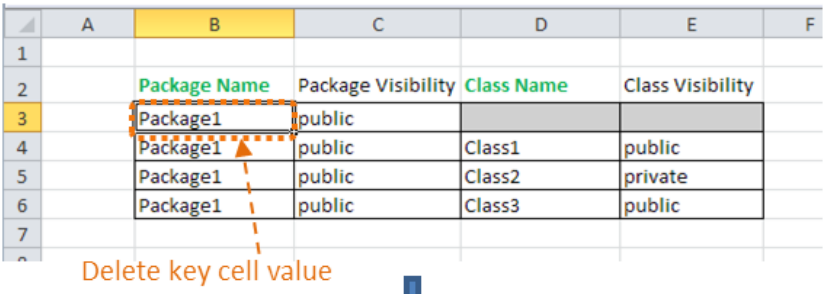
- *非キー属性フィールドの変更*
  非キー属性が変更されると、その変更はモデルに適用され、変更された要素の変更されたプロパティに紐付けされるすべてのフィールドおよび SyncView に反映されます。変更は、下図のように要素を定義するレコードで、またはこのプロパティにリンクされるほかのフィールドで行われます。たとえば、Package1 に対しては、下の例のとおり、列 C のオレンジ色の枠線で囲まれたセル (a および c) で同じ変更が行われます。



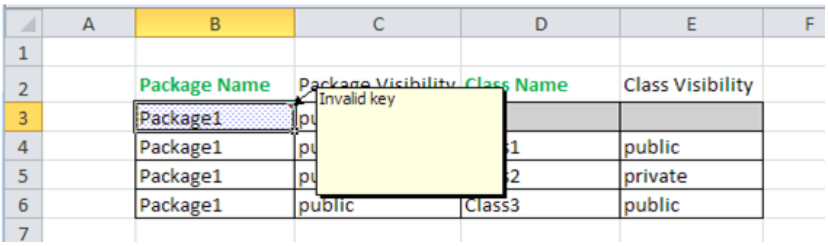
- *キー属性の値の削除*
  キーフィールド属性の値を削除すると、エラーになります。変更は、モデルに

反映されません。

a.



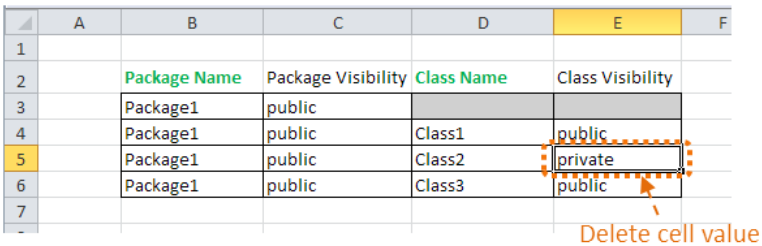
b.



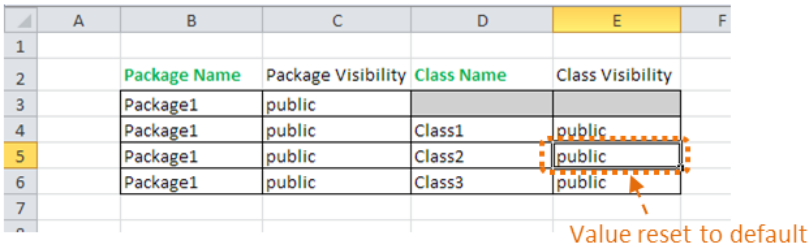
- *非キー属性の値の削除*

  非キー属性セルの値を削除した場合、値は削除されます。ただし、モデルによっては、値がデフォルト値にリセットされる場合があります。

a.



b.

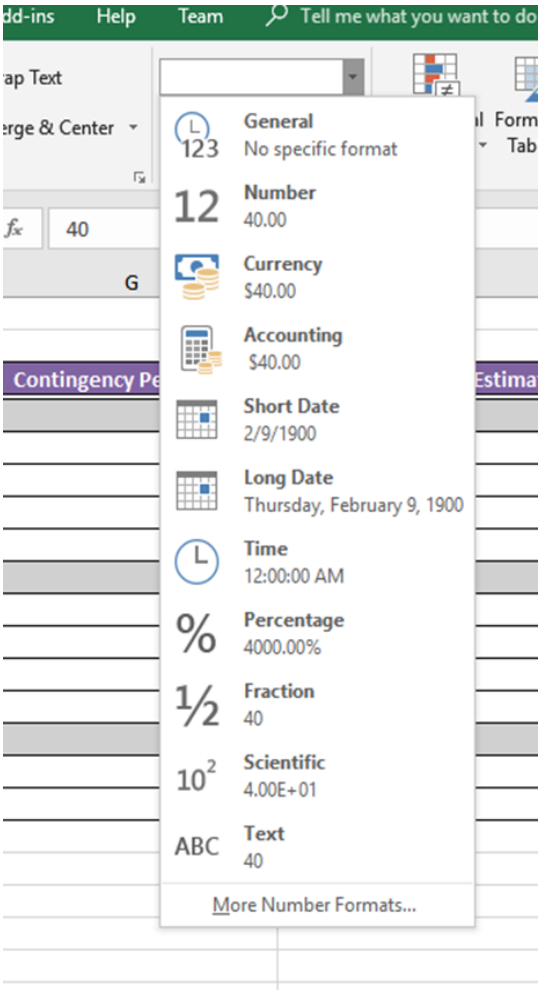


データ型の変換

Excel のセルの値およびモデル要素の値には、データ型があります。MapleMBSE では、セルの値のデータ型を暗黙的に変換して、割り当て時にモデル要素の値のタイプに合わせます。

- 文字列の変換
  - ターゲットモデル要素タイプが文字列の場合、セルの値は文字列に変換されます。
- Enum 変換
  - ターゲットモデル要素タイプが Enum の場合、文字列または整数は Enum に変換されます。失敗した場合、無効な値であることを示すエラーがセルに示されます。
- EDataType 変換
  - ターゲットモデル要素がほかの EDataType のものであり、セルの値がテキストの場合、その要素のターゲット EDataType への変換を試行します。失敗した場合、無効な値であることを示すエラーがセルに示されます。
- 数値の変換
  - ターゲットモデル要素が数値で、セルの値も数値の場合、セルの値を自動的に割り当てて、ターゲットモデル要素タイプに合わせます。たとえば、モデル要素が整数型で、セルの値が double 型の場合、セルの値は整数型になるよう切り捨てられ、モデル要素に対して設定されます。

セルの値のタイプは、セルの書式によって決まり、下図のように、数値の書式設定メニューを使用するか、セルを右クリックして [セルの書式設定] を選択することで設定できます。現在、MapleMBSE は、数値、テキスト、およびデータ型をサポートしています。



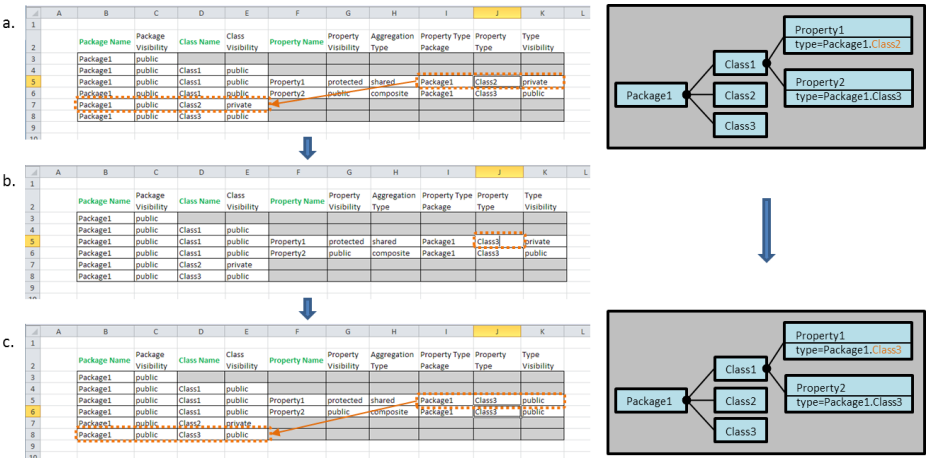
セルの値にアポストロフィー(')を付けると、書式設定値ではなくテキストとして値を指定できます。

紐付けされたレコードの参照フィールドの変更

紐付けされている、および紐付けされていないレコードの定義については、*レコードの状態 [42ページ]* を参照してください。

- *参照キーフィールドの変更*
  参照キーフィールドの値を、モデルですでに存在する別の要素を表す値に変更

する場合、参照先の要素が変更されます。モデルに存在しないキー値への変更の試みは、エラーになります。



- *参照非キーフィールドの変更*
  非キー参照フィールドの値を変更する場合、参照先の要素の対応するフィールドが更新されます。SyncView 内にこのプロパティに紐付けられる別のフィー

ルドがある場合は、それも更新されます。

a.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	private
Package1	public	Class1	public	Property2	public	composite	Package1	Class3	public
Package1	public	Class2	private						
Package1	public	Class3	public						

b.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	protected
Package1	public	Class1	public	Property2	public	composite	Package1	Class3	public
Package1	public	Class2	private						
Package1	public	Class3	public						

c.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	protected
Package1	public	Class1	public	Property2	public	composite	Package1	Class3	public
Package1	public	Class2	protected						
Package1	public	Class3	public						

- *参照キーフィールドの値の削除*

  参照キーフィールドの値を削除すると、エラーになります。

a.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	private
Package1	public	Class1	public	Property2	public	composite	Package1	Class3	public
Package1	public	Class2	private						
Package1	public	Class3	public						

Delete key reference value

b.

Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
Package1	public								
Package1	public	Class1	public						
Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	[illegible]
Package1	public	Class1	public	Property2	public	composite	Package1		[illegible]
Package1	public	Class2	private						
Package1	public	Class3	public						

No reference target

- *参照非キーフィールドの値の削除*
  非キー参照フィールドの値を削除した場合、値は削除されます。変更は、参照先の要素に反映されます。ただし、モデルによっては、値がデフォルト値にリセットされる場合があります。

a.

	Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
3	Package1	public								
4	Package1	public	Class1	public						
5	Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	private
6	Package1	public	Class1	public	Property2	public	composite	Package1	Class3	public
7	Package1	public	Class2	private						
8	Package1	public	Class3	public						

Delete cell value

b.

	Package Name	Package Visibility	Class Name	Class Visibility	Property Name	Property Visibility	Aggregation Type	Property Type Package	Property Type	Type Visibility
3	Package1	public								
4	Package1	public	Class1	public						
5	Package1	public	Class1	public	Property1	protected	shared	Package1	Class2	public
6	Package1	public	Class1	public	Property2	public	composite	Package1	Class3	public
7	Package1	public	Class2	public						
8	Package1	public	Class3	public						

Value reset to default

非マッピングフィールド

非マッピングフィールドは、要素のどのプロパティにもマッピングされていないレコードにあるフィールドです。そのため、非マッピングフィールドに入力された値は、モデルに影響を及ぼしません。ただし、非マッピングフィールドは、コメントやほかのフィールドに関連するExcel操作にとって有用な場合があります。非マッピングフィールドの数式は、テンプレートファイルに入力され(*MapleMBSE コンポーネント [1ページ]* を参照)、モデルが開いているときはいつでも使用できます。以下の例は、UserGuide.MSE 設定ファイルの AddingNewElements.uml モデルで使用される *UnmappedFields* スプレッドシートから取得されます。これらのファイルは **<MapleMBSE>/Example/UserGuide** にあります。ここで、**<MapleMBSE>** は MapleMBSE のインストールディレクトリです。

下図では、列 B、D、G、I が非マッピングフィールドです。これらには、アプリケーションへの入力によって更新される数式が含まれています。たとえば、セル B3 の数式は、下図の数式バーに表示されています。

B3 =IF(ISBLANK(C3),"Define Package in C","Done")

	A	B	C	D	E	F	G	H	I	J
1										
2		Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)	
3		Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing	
4										
5										
6										
7										
8										

Step 1 (列 C でパッケージ名を入力する) を完了すると、フィールドが更新されます。

	A	B	C	D	E	F	G	H	I	J
1										
2		Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)	
3		Done	Package1	Done			Done		Package	
4										
5										
6										
7										
8										

数式は、SyncView の新しい行に自動的に適用されることはありません。手動でこの操作を行うには、公式が入っているセルの右下角をクリックし、下方向にドラッグして任意の数の行にまで広げます。

	A	B	C	D	E	F	G	H	I	J
1										
2		Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)	
3		Done	Package1	Done			Done		Package	
4		Define Package in C								
5		Define Package in C								
6		Define Package in C								
7										
8										

SyncView領域が広がり、これらの行を取り込みます。列Cのセルの警告は、キーフィールドへの入力が必要であることを示しています。ほかの数式を SyncView 領域に広げることもできます。数式は、行に対して更新されます。たとえば、B4 の数式は C4 に対して更新されます。

B4 =IF(ISBLANK(C4),"Define Package in C","Done")

	A	B	C	D	E	F	G	H	I	J
1										
2		Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)	
3		Done	Package1	Done			Done		Package	
4		Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing	
5		Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing	
6		Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing	
7										
8										

ここで、列 B、D、G で指定された Step に従い、任意の要素を定義できます。

ヒント：この例は、*代替グループ [30ページ]* の例で示される手順に従ったものです。

Package1 の Class1 を定義します。

a.

	Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)
3	Done	Package1	Done			Done		Package
4	Done	Package1	Define Class in E or Package in F	Class1		Define Class in H		Package
5	Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing
6	Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing

b.

	Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)
3	Done	Package1	Done			Done		Package
4	Done	Package1	Done	Class1		Done		Class
5	Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing
6	Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing

Package1 の Package2 を定義します。

a.

	Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)
3	Done	Package1	Done			Done		Package
4	Done	Package1	Done	Class1		Done		Class
5	Done	Package1	Define Class in E or Package in F		Package2	Define Class in H		Package
6	Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing

b.

	Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)
3	Done	Package1	Done			Done		Package
4	Done	Package1	Done	Class1		Done		Class
5	Done	Package1	Done		Package2	Done		Nested Package
6	Define Package in C		Define Class in E or Package in F			Define Class in H		Nothing

Package2 の Class2 を定義します。

a.

	Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)
3	Done	Package1	Done			Done		Package
4	Done	Package1	Done	Class1		Done		Class
5	Done	Package1	Done		Package2	Done		Nested Package
6	Done	Package1	Done		Package2	Define Class in H	Class2	Nested Package

b.

	Step 1 (unmapped)	Package Name	Step 2 (unmapped)	Class Name	Nested Package Name	Step 3 (unmapped)	Nested Class Name	What is defined (unmapped)
3	Done	Package1	Done			Done		Package
4	Done	Package1	Done	Class1		Done		Class
5	Done	Package1	Done		Package2	Done		Nested Package
6	Done	Package1	Done		Package2	Done	Class2	Nested Class

2.7. ワークシートでの要素のロック

MapleMBSE のロック機能により、モデル要素の同時編集を制御し、編集中のモデル要素が別ユーザによって上書きされるのを防ぐことができます。

ユーザがモデルを編集すると、MapleMBSE は、変更中の特定のモデル要素をロックしようとします。これが成功した場合、ロックは MapleMBSE によって維持され、モデルは変更されます。成功しなかった場合、MapleMBSE はエラーを報告し、変更は取り消されます。

MapleMBSE から自動的に取得されるすべてのロックは、以下の場合に解除されます。

- ユーザが変更を確定した場合
- ユーザがモデルをリロードした場合
- MapleMBSE が終了した場合

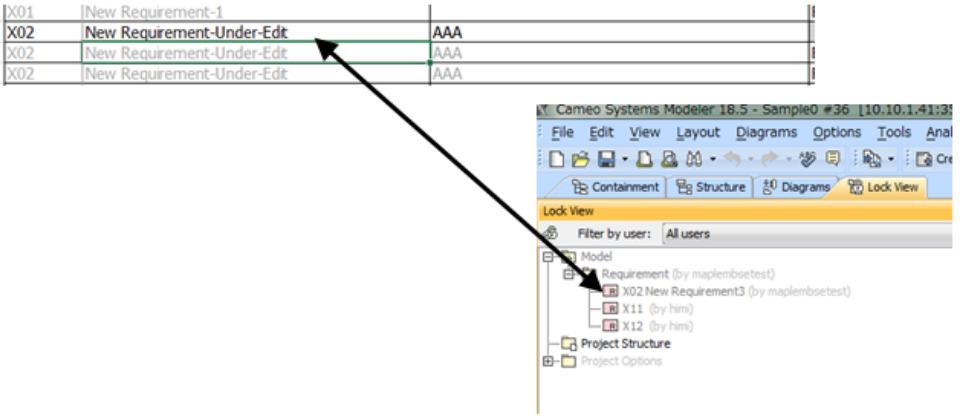
注 : MapleMBSE がクラッシュした場合、要素のロックは残ります。

また、ロックは必ずしも実行されないため、注意してください (要素のロックをサポートしない旧バージョンの MapleMBSE を使用する場合など)。

編集中のロック取得

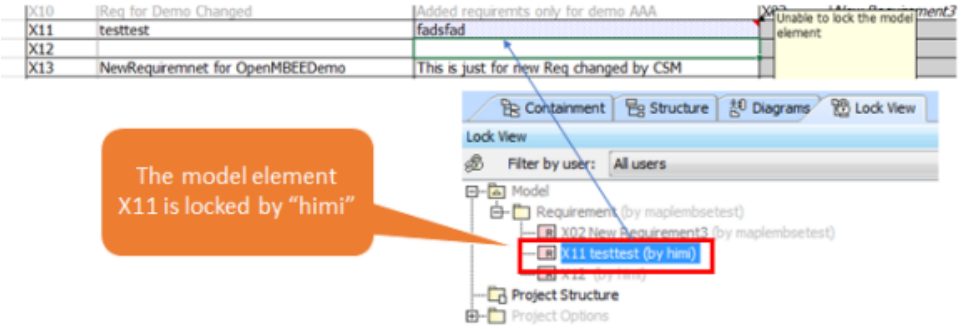
MapleMBSE でなんらかの変更を行う場合、MapleMBSE は変更するモデル要素のロックを取得しようとします。

下の例では、要件の名前 (X02) が「New Requirement-Under-Edit」に変わっています。



ロック取得の失敗

MapleMBSE は、モデル要素をロックできないという旨のエラーメッセージを Excel に表示します。MapleMBSE はユーザの変更を取り消します。



変更の確定およびコードの解除

MapleMBSE が変更を確定する、終了する、またはモデルをリロードする場合、ユーザがモデル編集中に自動的に取得されたすべてのロックを自動的に解除します。ほかのアプリケーションによって取得済みのほかのロックはそのまま維持されます。

注 : MapleMBSE が異常終了 (クラッシュ) した場合、MapleMBSE が設定したロックは残ります。ユーザは、Cameo Systems Modeler、MagicDraw、または

Teamwork Cloud Web UI から、これらのロックを手動で削除する必要があります。

ほかの依存関係に関する操作

その他の依存関係や要件がある場合、それらはロックされます。

2.8. Excel でのその他の編集操作

CutCopy モード

Excel の CutCopy モードは、下図に示すような点線で囲まれた、切り取り/コピー対象として選択された領域内のモードです。Excel では通常、CutCopy モードは、貼り付けが行われた後も維持されます。MapleMBSE では、CutCopy モードは、貼り付け処理が完了した後は解除されます。

	A	B	C	D	E	F
1						
2		Package Name	Class Name	Nested Package Name	Nested Class Name	
3		Package1				
4		Package1	Class1			
5		Package1		Package2		
6		Package1		Package2	Class2	
7						

セルの書式設定

- モデルでは、セルの値のみが保存されます。セルのスタイルおよび書式設定は、保存されず、モデルにも反映されません。

付録A ログファイル

A.1. ログファイルの説明

以下の表では、MapleMBSE で作成されるログファイルについて説明します。
この表で、<InstallDir> は MapleMBSE のインストールフォルダ、%USERPROFILE% は環境変数 USERPROFILE の値をそれぞれ表しています。

通常、USERPROFILE の値は以下のように設定されています。

- Windows 7 および 10 の場合 : C:\Users\<User Name>

ログファイル	説明
%USERPROFILE%\MapleMBSE_Launcher.log	MapleMBSE.exe のログデータ
%USERPROFILE%\MapleMBSE_OSGiBridge.log	• MapleMBSE のモジュールを Excel にロードするモジュール (OSGiBridge) のログデータ • **<InstallDir>\OSGiBridge.ini** でログレベルを指定できます。
%USERPROFILE%\Documents\MapleMBSE.log	• MapleMBSE.log ファイルは **My Documents** フォルダに保存されています。 • MapleMBSE 自体のログデータ • **<InstallDir>\OSGiBridge\log4j.properties** でログレベルを指定できます。 • MapleMBSE.log ファイルは MapleMBSE を起動するたびに上書きされます。

A.2. ログファイルを使用した問題の特定

MapleMBSE.log ファイルには、ワークシートで更新されたセルの値が記録されます。

大量のデータがログに記録されていると、コンピュータの実行速度が遅くなることがあります。MapleMBSE.logファイルのサイズは数十MBを超えることがあります。

1. **<InstallDir>\OSGiBridge\log4j.properties** の名前を変更します。**log4j-orig.properties** という名前を指定します。
2. **<InstallDir>\OSGiBridge\log4j-debug.properties** の名前を変更します。**log4j.properties** という名前を指定します。
3. MapleMBSE を再起動します。
4. MapleMBSE で問題を再現します。
5. 上記の手順を実行した後で、%USERPROFILE%\Documents に保存されている **MapleMBSE.log** を適切な場所にコピーします。
6. **<InstallDir>\OSGiBridge\log4j-orig.properties** の名前を **log4j.properties** 変更し、ログ記録のプロパティを元の設定に戻します。

付録Bエラーメッセージとトラブルシューティング

B.1. MapleMBSE 起動時のエラーメッセージ

エラーメッセージ	説明	解決方法
二重に MapleMBSE.exe を起動できません。開いている Excel のウィンドウをすべて閉じ、MapleMBSE.exe のプロセスが終了していることを確認してから再度起動してください。	別の MapleMBSE インスタンスを起動しようとして失敗しました。同時に実行できる MapleMBSE インスタンスは 1 つのみです。	すべての Excel ウィンドウを閉じ、Excel プロセス (MapleMBSE.exe) が終了していることを確認してから、もう一度 MapleMBSE を起動します。
ファイルオープンエラー	モデルファイルまたは SyncModel ファイルを開けません。	エラーメッセージダイアログの指示に従います。

B.2. SyncView 表を変更しようとしたときに表示されるエラーメッセージ

編集時にメッセージボックスに表示されるメッセージを以下に示します。
これらのエラーはMapleMBSEで処理されます。ユーザに起因するエラーの場合、操作は取り消されます。

エラーメッセージ	説明	解決方法
最左列の複数行にわたる上書きはできません。	左端の列を複数行にわたって上書きしようとしましたが、左端のセルのプロパティには MapleMBSE がデータ管理に使用する ID が指定されています。そのため、左端の列を複数行にわたって上書きすることはできません。	表示されたエラーメッセージの原因となった編集操作は取り消されます。モデルを保存せずに MapleMBSE を再起動します。

最上行への挿入は現在は禁止されています。	SyncView 表の先頭行の上に行を挿入しようとしましたが、この操作は実行できません。	同上
挿入行(列)の削除はできません。	挿入行または挿入列を削除しようとしましたが、この操作は実行できません。	同上
表の移動はできません。	SyncView 表を移動しようとしましたが、この操作は実行できません。	同上
複数の矩形領域からなる更新を一度に行うことはできません。	複数の矩形領域を更新しようとしました。たとえば、表内で隣接しない複数の領域を選択し、それらの領域にコンテンツをコピーするなどです。この操作は実行できません。	同上
編集領域全体を削除することはできません。	SyncView 表全体を削除しようとしましたが、この操作は実行できません。	同上
列の削除はできません。	SyncView 表の列を削除しようとしましたが、この操作は実行できません。	同上
列の挿入はできません。	SyncView に列を挿入しようとしましたが、この操作は実行できません。	同上

B.3. モデルスプレッドシートの編集時に表示されるエラーメッセージ

編集時にセルにコメントで表示されるメッセージを以下に示します。

エラーメッセージ	**説明**	**解決方法**
キーが重複しています。	既存のキー名が指定されました。	別のキー名を指定するか、キーが終了キーでない場合は終了キーを追加して指定します。

キーが未入力です。	キーが未入力です。	キーを入力します。
参照先が見つかりません。	リンクフィールドに表示されているリンクは存在しません。	リンクフィールドに正しい値を入力します。
値が無効です。	セルの値が無効です。	正しい値を入力します。
セルを変更できません。	変更できないセルを変更しようとしました。	行を挿入した後にこのエラーが発生する場合は、無効な機能がモデルに挿入された可能性があるので、挿入した行を削除します。また、セル値を変更した後にこのエラーが発生する場合は、モデルで変更できない機能を変更している可能性があるので、元の値に戻します。どちらの場合も、変更はモデルに反映されません。

B.4. 編集時のエラーメッセージ (処理できない例外)

処理できない例外のエラーが発生する場合は、モデルに名前を付けて保存してから、MapleMBSE を終了します。

B.5. メモリ残量に関する警告メッセージ

MapleMBSE では、操作のたびに残りのメモリ容量を確認します。使用可能なメモリ量が不足すると、警告メッセージが表示されます。メッセージの指示に従います。

メモリ残量に関する警告が頻繁に表示される場合は、現在のメモリ (Java のヒープサイズ) が編集中のモデルに対して十分でない可能性があります。「Java ヒープサイズの変更」の手順に従って、Javaのヒープサイズを拡張することを検討してください。

B.6. トラブルシューティング

• *Java ヒープサイズの変更 [70ページ]*

Java ヒープサイズの変更

MapleMBSE の起動時または編集時にメモリ不足に関するエラーダイアログが表示される場合があります。この問題が発生する場合は、以下の手順に従って、Java ヒープサイズを調整します。

1. デスクトップ上のショートカット (またはアプリケーションのフォルダ) で MapleMBSE を右クリックし、[プロパティ] を選択します。
2. フィールドでコマンドラインオプション /R<memsize> をショートカットリンクターゲット (たとえば、"C:\Program Files\MapleMBSE 2017\MapleMBSE.exe") の末尾に追加します。ここで <memsize> は、Java 仮想マシン用に予約された連続メモリの値 (800MB から 1100MB までの範囲) です。このメモリの一部は Java ヒープに、一部は JVM 自体に使用されます。指定可能なその他のオプションを含む詳細については、*コマンドラインからの起動 [7ページ]*を参照してください。
3. **[OK]** または **[適用]** をクリックします。
4. ショートカットをダブルクリックして MapleMBSE を起動します。
5. エラーメッセージが表示される場合は、上記の手順 2 で指定した <memsize> の値を増やします。MapleMBSE の起動時にエラーメッセージが表示されなくなるのに十分な値が見つかるまで、この手順を繰り返します。
6. MapleMBSE のインストールフォルダに保存されている OSGiBridge.ini を開き、テキストエディタで JVM の最大ヒープサイズを変更します。
7. このファイルで「-Xmx512m」というエントリを探します。「512」を手順 2 の <memsize> の値で置き換えます。
8. MapleMBSE の起動に失敗する場合は、手順 7 で設定した Java ヒープ用の -Xmx<memsize>m をそれよりも小さい (<memsize> よりも 150MB 〜 200MB 程度少ない) 値で置き換えます。これにより、JVM 自体に必要な一部のメモリ容量を割り当てることができます。