

Borland® Together® Edition for Microsoft® Visual Studio® .NET 2.0

Getting Started ガイド

Borland® Together®

統合 / アジャイル モデリング ソリューション

Borland®
Excellence Endures™

Borland Software Corporation の 100% 子会社である TogetherSoft Corporation は、本書に 記載されているアプリケーションに対する特許を取得または申請している場合があります。適用される特許については、製品 CD または [バージョン情報] ダイアログボックスを参照して下さい。本書の提供によって、これらの特許に関するいかなる権利も使用者に付与されるものではありません。

Copyright © 2003-2004 TogetherSoft Corporation, a wholly owned subsidiary of Borland Software Corporation. All rights reserved. Borland のブランド名および製品名はすべて、米国 Borland Software Corporation の米国およびその他の国における商標または登録商標です。Microsoft および Visual Studio は Microsoft Corp. の米国およびその他の国における登録商標です。その他の製品名は、各社の商標または登録商標です。

2004 年 5 月 20 日更新

目次

Borland Together Edition for Microsoft Visual Studio .NET 入門 5

Borland Together Edition for Microsoft Visual Studio .NET とは？	5
このガイドについて	6
Together VS .NET の技術資料	7
使用権について	8

モデリングから始めましょう 9

Together VS .NET を動かしてみよう	9
一般的なモデリング オプションの設定	10
プロジェクトの作成とモデル ビューやダイアグラム ビューのオープン	11
ユースケース図の作成	12
図の記述	13
システム境界の追加	14
ユースケース要素と通信リンクの追加	14
ステレオタイプの追加	15
シナリオの完成	16
クラス図の作成	17
区画の表示設定	17
メソッドやフィールドの追加	18
関係とリンクの作成	21
ハイパーリンクの作成	23
シーケンス図の作成	25
初期シーケンス図の作成	25
オブジェクトへクラスを関連付ける	26
新しいメソッドをメッセージリンクに追加する	26
既存のソース コードからシーケンス図を生成する	27
シーケンス図とコラボレーション図間の変換	28
モデル ビューの利用	29
参照の表示	30
展開可能な図ノードの表示	30
図の要素の検索	31
プロパティ ウィンドウの利用	32
概要図の利用	33
ダイアグラム ビューの管理	34
情報の表示 / 非表示	34
[非表示要素の表示] ダイアログの利用	35

Together VS .NET 機能の利用 37

パターンの利用と作成	37
品質保証の実行	40

検査の実行	40
検査結果ビューでの作業	41
ソース コードを開き、検査項目の説明を表示 する	41
検査結果の印刷	42
検査結果の保存	43
検査結果の並び替えとグループ化	43
検査結果の更新と再実行	44
特定の要素を選択して検査を実行する	44
リファクタリング	45
名前の変更	46
[リファクタリング] ウィンドウでの作業	46
ドキュメント生成	47
XMI プロジェクトのインポート / エクスポート	50
XMI プロジェクトのエクスポート	50
XMI プロジェクトのインポート	51

第 1 章

Borland Together Edition for Microsoft Visual Studio .NET 入門

この章では次のトピックについて説明します。

- 「[Borland Together Edition for Microsoft Visual Studio .NET とは？](#)」 (5 ページ)
- 「[このガイドについて](#)」 (6 ページ)
- 「[Together VS .NET の技術資料](#)」 (7 ページ)
- 「[使用権について](#)」 (8 ページ)

Borland Together Edition for Microsoft Visual Studio .NET とは？

Borland Together Edition for Microsoft Visual Studio .NET (Together VS .NET) は、Microsoft® Visual Studio® .NET 2003 (Visual Studio .NET) に対する、UML モデリング拡張パッケージです。Together VS .NET は Borland 製品ラインのエディションの 1 つであり、UML モデリングやパターン、コードリファクタリング、ドキュメント生成などの機能を持つ、数々の受賞歴を持つ設計主導の環境を、Visual Studio .NET 開発環境での C# および Visual Basic .NET プロジェクトに対してシームレスに統合します。

メモ QA 検査とコードリファクタリングは C# プロジェクトに対してのみ有効な機能です。

Together VS .NET の主要機能である LiveSource™ は、ユーザーの Together VS .NET 図と Visual Studio .NET エディタ上のソースコードとの同期を取ります。

このガイドでは、Together VS .NET および Visual Studio .NET が既にインストール済みであることを想定しています。インストール場所に関しては、Together VS .NET も Visual Studio .NET もどちらも任意の場所で構いません。インストールを調整したり、同じルートディレクトリにインストールすることによるメリットやデメリットは、特にありません。Together VS .NET および Visual Studio .NET のインストールに関する情報については、それぞれの製品ドキュメントを参照して下さい。

このガイドについて

本ガイドの目的は、Together VS .NET に関する基本概念と簡単な利用方法を提供することです。まずサンプルプロジェクトの作成方法を説明した後、そのプロジェクトを利用して、Together VS .NET の基本機能の使用方法を紹介していきます。

このため、本ガイド内のサンプルは、前から順番にご覧になっていただくことをお勧めします。各作業に特化した詳細な説明については、Together VS .NET のオンラインヘルプを参照して下さい。Visual Studio .NET の [ヘルプ] メニューから **[目次]** を選択し、その [目次] ペインで **Borland Together Edition for Microsoft Visual Studio .NET** を選択して下さい。

メモ 本ガイドで提供されている作業手順は、Borland Together Edition for Microsoft Visual Studio .NET の現バージョンに特化したものです。

本ガイドでは、次の項目に関する例を説明していきます。

- ・「[Together VS .NET を動かしてみよう](#)」
- ・「[プロジェクトの作成とモデル ビューやダイアグラム ビューのオープン](#)」
- ・「[ユースケース図の作成](#)」
- ・「[クラス図の作成](#)」
- ・「[シーケンス図の作成](#)」
- ・「[モデル ビューの利用](#)」
- ・「[プロパティ ウィンドウの利用](#)」
- ・「[概要図の利用](#)」
- ・「[ダイアグラム ビューの管理](#)」
- ・「[パターンの利用と作成](#)」
- ・「[品質保証の実行](#)」
- ・「[リファクタリング](#)」
- ・「[ドキュメント生成](#)」
- ・「[XMI プロジェクトのインポート / エクスポート](#)」

Together VS .NET の技術資料

Together VS .NET の技術資料は、表 1 のような構成になっています。

表 1 Together Edition for Microsoft Visual Studio .NET 技術資料

項目	説明	場所
README	最新情報。以下の項目を含みます。 <ul style="list-style-type: none">• ライセンス ノート• システム要件• Together VS .NET のインストールと起動• 既知の問題	README の場所： <ul style="list-style-type: none">• Together VS .NET インストール ディレクトリ以下の Docs ディレクトリ： Docs/readme.html• http://info.borland.com/techpubs/together
新機能	Together Edition for Microsoft Visual Studio .NET の新しい機能と改良された機能についての詳細情報	新機能の場所： <ul style="list-style-type: none">• Together VS .NET インストール ディレクトリ以下の Docs ディレクトリ： Docs/what's_new.html• http://info.borland.com/techpubs/together
Getting Started ガイド	初めて使用するユーザー向けの情報。以下の項目を含みます。 <ul style="list-style-type: none">• Together VS .NET と Visual Studio .NET で利用する C# プロジェクトの作成• ユースケースとクラス図での作業• Together VS .NET でのオプションの設定• Together VS .NET の使用方法に関するその他の情報	PDF の場所： <ul style="list-style-type: none">• Together VS .NET インストール ディレクトリ以下の Docs ディレクトリ： Docs/gettingStarted.pdf• http://info.borland.com/techpubs/together
オンライン ヘルプ	Together VS .NET に関するオンラインヘルプ。ユーザーに特に関係のある包括的な情報で、以下の項目を含みます。 <ul style="list-style-type: none">• Together VS .NET 入門• 個人のプリファレンスおよびオプションの設定• Together VS .NET の機能を使用するための詳細な説明• Together VS .NET における UML 図での作業	Visual Studio .NET のメイン メニューの [ヘルプ 目次]
Together VS .NET の登録と使用権の設定	Together VS .NET の使用権の設定	<ul style="list-style-type: none">• Together VS .NET インストール ディレクトリ以下の Docs ディレクトリ： Docs/setting_up_licensing.html• http://info.borland.com/techpubs/together

使用権について

Together VS .NET と Visual Studio .NET は、それぞれ別々に使用権が必要となります。Visual Studio .NET の使用権については、Visual Studio .NET のドキュメントを参照して下さい。

Together VS .NET の使用権についての情報は、『**Borland Together Edition for Microsoft Visual Studio .NET の登録と使用権の設定**』

([setting_up_licensing.html](#)) ドキュメントを参照して下さい。これは、お使いの Together VS .NET のインストール ディレクトリ以下、Docs ディレクトリにあります。

第 2 章

モデリングから始めましょう

この章では、Together VS .NET の基本的な使用方法を説明していきます。具体的には、Together VS .NET 環境内をナビゲートし、プロジェクトを作成、そして図を作成する方法を見ていきます。

この章では次のトピックについて説明します。

- 「Together VS .NET を動かしてみましょう」(9 ページ)
- 「プロジェクトの作成とモデル ビューやダイアグラム ビューのオープン」(11 ページ)
- 「ユースケース図の作成」(12 ページ)
- 「クラス図の作成」(17 ページ)
- 「シーケンス図の作成」(25 ページ)
- 「モデル ビューの利用」(29 ページ)
- 「プロパティウィンドウの利用」(32 ページ)
- 「概要図の利用」(33 ページ)
- 「ダイアグラム ビューの管理」(34 ページ)

Together VS .NET を動かしてみましょう

Together VS .NET は、Visual Studio .NET に対して、モデル ビューとそれに対応するダイアグラム ビューとプロパティウィンドウを提供します。統合された形でこれらのツールは、プロジェクトを UML で設計し、そのプロジェクトのためのソース コードを作成するための、中心的な役割を果たします。さらには、Together VS .NET によって、それらの図とソース コードの同期は、常に維持されます。

Together VS .NET が、お使いの Visual Studio .NET 開発環境にインストールされると、Together VS .NET のモデリング機能は、自動的に C# と Visual Basic .NET プロジェクトに対して使用可能な状態になります。

メモ このガイドでは、C# プロジェクトでのモデリングを行います。

Together VS .NET は、次の 3 つの主要なコンポーネントを Visual Studio .NET 開発環境に追加します。


- **モデル ビュー**：Together VS .NET の基本ナビゲーションビュー。このビューを利用して、図の要素の管理や、開閉、作成、削除などができます。新たな要素を見つけた際には、そのコンテキストメニューを確認していくと良いでしょう。
- **ダイアグラム ビュー**：UML 図がここに表示されます。このビューには、開いている図 1 つにつきタブが 1 つ表示されます。
- **プロパティ ウィンドウ**：Together VS .NET を使用すると、ダイアグラム ビューやモデル ビューで選択された要素のプロパティが、このビューに表示されます。各要素のプロパティは、通常 2 つのカテゴリに分類されています。
 - **[デザイン]** - 選択要素の前景色と背景色の定義
 - **[全般]** - 選択要素の UML プロパティ、およびソースコード要素のソースコードプロパティ

メモ 分類子（クラス、インターフェイス、デリゲート、列挙型、構造体（C# プロジェクトのみ）、モジュール（Visual Basic .NET プロジェクトのみ））については、プロパティ ウィンドウに **[コメント]** カテゴリが追加されており、ソースコードに **[作者]**、**[導入されたバージョン]**、**[バージョン]** のコメント タグを追加することができます。

モデル ビューやダイアグラム ビューを開く

既存または新規の C# プロジェクトに対して Together VS .NET を使用するには、そのプロジェクトのモデル ビューと、それに対応するダイアグラム ビューを開く必要があります。

C# プロジェクトに対して、モデル ビューを開き、それに対応するダイアグラム ビューを表示させるには：

- 1 **[表示]** メニューから、**[Together VS .NET モデル ビュー]** を選択します。モデル ビューが開かれます。
- 2 ルートプロジェクト ノードを展開します。
- 3 そのプロジェクトに対するダイアグラム ビューを開くには、**(デフォルト)** クラス図ノード  をダブル クリックします。

全般的なモデリング オプションの設定

Together VS .NET では、Together VS .NET のモデリング サポートを、**[オプション]** ダイアログを使用して、開いている C# および Visual Basic .NET プロジェクトすべて、または新しい C# および Visual Basic .NET プロジェクトすべてに対して、全体的に無効化することができます。

すべての新しいプロジェクトに対して、モデリング サポートを無効化するには：

- 1 **[ツール]** メニューから、**[オプション]** を選択します。**[オプション]** ダイアログが開かれます。

- 2 **[Together VS .NET]** フォルダをクリックして、Together VS .NET 固有の設定オプションを参照します。オプション [一般] が表示されています。
- 3 [オプション] ダイアログで、フィールド **[新しいプロジェクトで Together VS .NET サポートを自動的に有効化]** をクリックします。ドロップ ダウン矢印が現れます。
- 4 ドロップダウン矢印をクリックします。ドロップダウンリストから **[いいえ]** を選択し、**[OK]** をクリックします。これにより、Together VS .NET モデリング サポートが無効化されます。ただし、今回のチュートリアルを完成するには、値を **[はい]** に戻しておくよう気をつけて下さい。

ヒント また、フィールド **[開いたプロジェクトで Together VS .NET サポートを自動的に有効化]** をクリックして、現在開かれているプロジェクトに対するモデリング サポートを有効化 / 無効化することもできます。このオプションは Together VS .NET サポートを一度も有効化していないプロジェクトに対してのみ機能します。

プロジェクトの作成とモデル ビューやダイアグラム ビューのオープン

このガイドでは、Together VS .NET の様々な機能を、簡単なビデオ ストア (VideoStore) プロジェクトを設計しながら確認していきます。


新しいプロジェクトを作成するには：

- 1 [ファイル] メニューから、**[新規作成 | プロジェクト]** を選択します。[新しいプロジェクト] ダイアログが開かれます。
- 2 **[プロジェクトの種類]** ペインで、**[Visual C# プロジェクト]** を選択します。
- 3 **[テンプレート]** ペインで、**[空のプロジェクト]** を選択します。
- 4 プロジェクト名には、「VideoStore」と入力して下さい。
- 5 **[OK]** をクリックします。

プロジェクトが作成され、ソリューション エクスプローラ タブのページに表示されます。

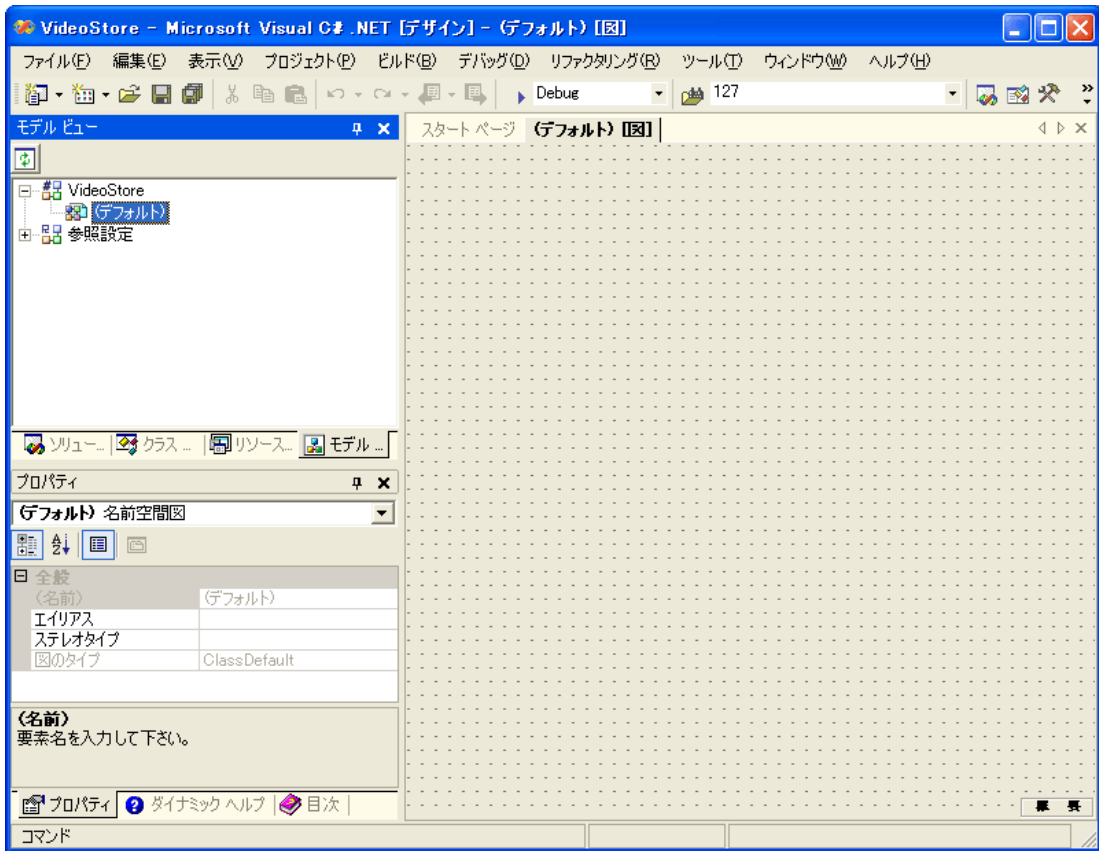
ヒント ソリューション エクスプローラを手早く開くには、**[CTRL] + [ALT] + [L]** を入力します。

モデル ビューを開くには：

- 1 [表示] メニューから、**[Together VS .NET モデル ビュー]** を選択します。モデル ビューが開かれます。
- 2 [VideoStore] プロジェクト ノードを展開します。
- 3 **(デフォルト)** クラス図ノード  をダブルクリックします。ダイアグラム ビューが表示されます (図 1 参照)。

モデル ビューが、ドッキング可能なウィンドウである点に注目して下さい。モデル ビューは最初、自由に移動可能なフローティング ウィンドウとして開かれます。しかし、フローティング形式のままでも、ドッキング領域へドッキングしても構いません。モデル ビューのドッキング領域は、Visual Studio .NET ウィンドウの 4 辺どれを使用しても構成することができます。つまり、モデル ビュー ウィンドウを好みに応じて好きな場所に配置することができますのです。

図 1 Together VS .NET を使用した状態の VideoStore プロジェクト



ユースケース図の作成

以降の手順では、前セクションで説明した VideoStore プロジェクトが作成されていることを前提とします。

ユースケース図を作成するには：

- 1 モデル ビューで **[VideoStore]** プロジェクト ノードを右クリックし、**[追加 | その他の図]** を選択します。
- 2 その結果開かれたダイアログ ボックスで、**[ユースケース図]** を選択します。名前には、「VideoStore Use Case」と入力して下さい。
- 3 **[OK]** をクリックします。ダイアグラム ビューに、ユースケース図が開かれます。

ヒント このダイアログ ボックスを使用すれば、Together VS .NET でサポートされている図なら、どれでも作成することができます。

図の記述

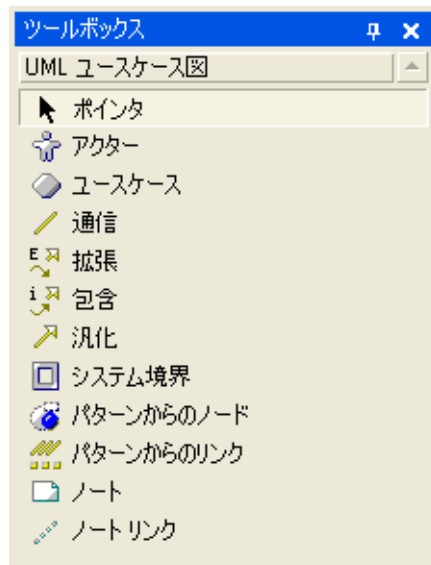
ユースケース図は、システムが**何**を行うのかを記述するシナリオです。この図には、3つの主要な要素として、アクター、ユースケース、通信リンクを含みます。ツールボックスや図のコンテキストメニューには、図を記述するための設計要素を含んでいます。


図を記述するには：

- 1 ツールボックスを使用して、ユースケース図に設計要素を追加します。ツールボックスを開くには、[表示]メニューから、**[ツールボックス]**を選択して下さい。
- 2 設計要素を参照するには、ツールボックス上の**[UML ユースケース図]**をクリックします。

UML ユースケース図のツールボックス要素を、図 2 に示します。

図 2 ユースケース図のツールボックス要素




- 3 [アクター] ボタン  をクリックし、続いて図の背景をクリックします。
- 4 要素の名前は自動的に、直接編集可能になります。そのアクターの名前には、「Clerk」と入力して下さい。[Enter] を押し、名前を適用して直接編集エディタを閉じます。

ヒント 図の要素に対して直接編集エディタを再度使用可能な状態にしたい場合には、その要素のテキスト キャプション部分をダブルクリックして下さい。

システム境界の追加

図にシステム境界を追加して、在庫システムと外部のアクターを分離します。

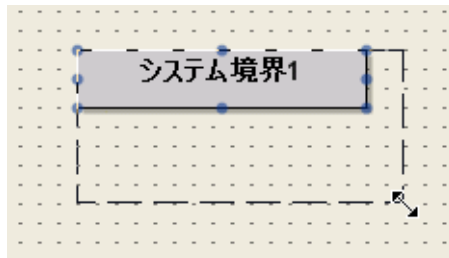
システム境界を追加するには：

- 1 [システム境界] ボタン  をクリックし、続いて図の背景の Clerk の右の部分をクリックします。
- 2 直接編集エディタを使用して、システム境界の名前を「Inventory System」として下さい。

ヒント 代わりに、図の背景をクリックし、左マウス ボタンを押したままマウスをドラッグして、特定のサイズになるようシステム境界を描画することもできます。

システム境界の（または、その他の図の要素でも）サイズを変更するには、図上で目的のシステム境界オブジェクトを選択し、目的のサイズになるまで、その要素の角をドラッグします（図 3 参照）。マウス ポインタの移動方向によって、要素は拡大 / 縮小されます。



図 3 システム境界のサイズ変更



ユースケース要素と通信リンクの追加

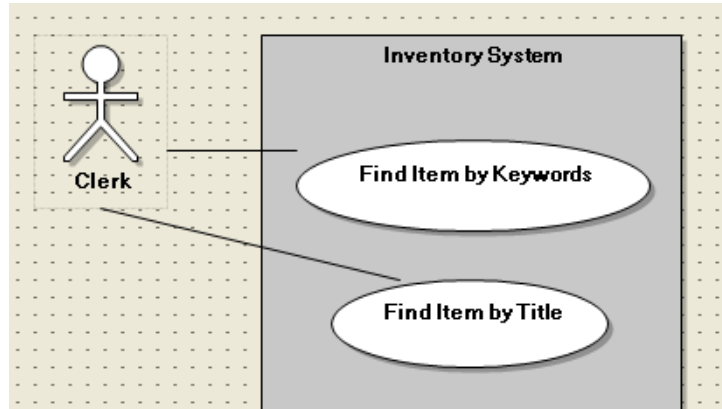
ユースケース要素は「タイトルから映画を検索する」などといった 1 つのタスクを表すシナリオの概要を表現します。通信リンクはアクターとユースケース要素間の関連を示します。

ユースケース要素と通信リンクを追加するには：

- 1 [ユースケース] ボタン  をクリックし、このシステム境界の中に、ユースケースを 2 つ作成して下さい。ユースケースの名前を、それぞれ「Find Item by Keywords」と「Find Item by Title」に変更します。
- 2 [通信] ボタン  をクリックし、アクターからユースケースへドラッグ&ドロップします。この作業を繰り返し、各ユースケースに対してそれぞれリンクを作成して下さい。

作成中のユースケース図は、以下ようになります（図 4 参照）。

図 4 サンプル ユースケース図



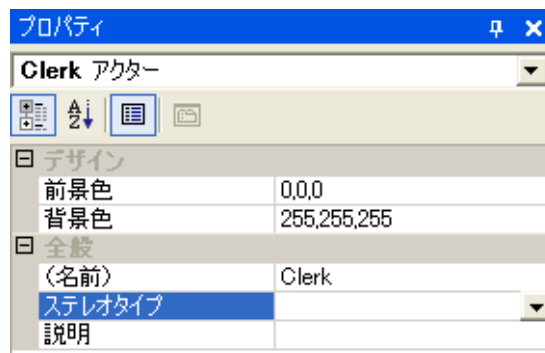
ステレオタイプの追加

Together VS.NET は、ステレオタイプの使用をサポートしています。UML で定義されたステレオタイプやユーザーの要件に基づいたカスタム ステレオタイプを指定することができます。


ステレオタイプを追加するには：

- 1 Clerk アクターを選択し、Clerk のプロパティをプロパティウィンドウで確認します (図 5 参照)。プロパティウィンドウを開くには、[表示]メニューから **プロパティウィンドウ** を選択します。または、キーボードから [F4] キーを使用することもできます。

図 5 Clerk のプロパティウィンドウ



- 2 プロパティウィンドウで、**ステレオタイプ** フィールドをクリックします。図 5 に示すような、ドロップダウン矢印が現れます。
- 3 ドロップダウン矢印をクリックし、一覧から **[manager]** を選択します。

メモ プロパティの中には、ドロップダウン矢印や情報ボタンを持つものがあります。その場合、ドロップダウンリストやダイアログの形で、選択肢やエントリを続いて指定することができます。

ヒント 一旦、図の要素へステレオタイプが追加されたら、以降は直接編集エディタを使用して、[ステレオタイプ] フィールドの値を変更することができます（図 6 参照）。

図 6 [ステレオタイプ] フィールドに対して直接編集エディタを使う



シナリオの完成

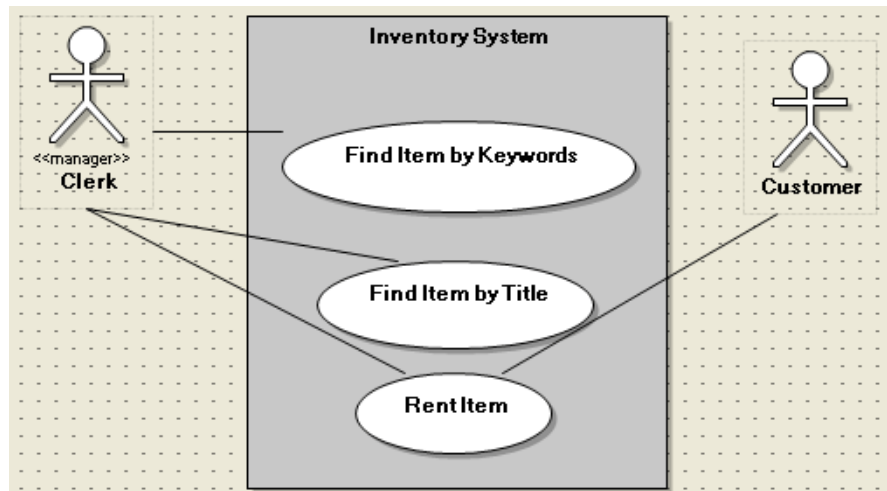
Video Store ユースケース図を完成するには、ユースケースとアクター要素、および適切な要素間に通信リンクを追加する必要があります。

図を完成するには：

- 1 もう 1 つユースケースを、**Inventory System** システム境界内に追加します。名前を「Rent Item」とします。
- 2 **Inventory System** システム境界の右側に、もう 1 つアクターを追加します。名前を「Customer」とします。
- 3 [通信] ボタンを使用して、このユースケースから、Customer と Clerk アクターの両方へリンクを張ります。

作成中の図は、以下のようになります（図 7 参照）。



図 7 VideoStore プロジェクトのユースケース図



クラス図の作成

Together VS .NET は、プロジェクト内の名前空間それぞれに対して、物理的なクラス図を作成します。VideoStore プロジェクトにも (デフォルト) クラス図があり、ルートレベルにてこのプロジェクトを表しています。

(デフォルト) クラス図を開いて使用するには :

- 1 モデル ビューで、VideoStore プロジェクト ノードを展開します。
- 2 VideoStore の (デフォルト) クラス図 ノード  をダブルクリックします。ダイアグラム ビューに、(デフォルト) クラス図が開かれます。
- 3 クラスを作成するには、ツールボックスを使用します。ツールボックスで **[UML クラス図]** をクリックし、クラス図要素を表示させます。[クラス] ボタン  をクリックし、続いて図の背景をクリックします。直接編集エディタを使用して、クラスの名前を「Store」に変更して下さい。
- 4 **Store** クラスを選択し、プロパティ ウィンドウにその詳細を表示させます。
- 5 **[ステレオタイプ]** フィールドを選択して、ドロップダウン矢印をクリックし、**[place]** を選択します。
- 6 ステップ 3-5 を繰り返し、クラス **Clerk** (ステレオタイプ = role) と **Item** (ステレオタイプ = description) を作成して下さい。

ここまで説明したステップ以外にも、Together VS .NET には、他に 2 つのクラス作成方法があります。

- ・ 図の背景を右クリックし、**[追加 | クラス]** を選択。
- ・ モデル ビューで、ルート プロジェクト ノードまたは (デフォルト) クラス図を右クリックし、**[追加 | クラス]** を選択。

区画の表示設定

ユーザーは、クラス要素やインターフェイス要素における、様々なメンバのための区画を、開いたり閉じたりすることができます。区画は、図においてデフォルトで表示されます。[オプション] ダイアログを使用して、区画制御の表示を、好みに応じて、直線や折りたたみ可能なリストに、設定することができます。区画制御は、常に表示する必要のないコンテンツを持つ大きなコンテナ要素を取り扱う場合、特に便利です。

区画に表示形式を設定するには :

- 1 [ツール] メニューから、**[オプション]** を選択して下さい。[オプション] ダイアログが開かれます。
- 2 **Together VS .NET** フォルダをクリックし、**[図]** ノードを選択します。
- 3 **[区画を線として表示]** フィールドをクリックします。
- 4 ドロップダウン矢印をクリックし、**[いいえ]** を選択します。
- 5 **[OK]** をクリックします。作成中の図要素は、以下のようになります (図 8 参照)。

区画を展開 / 折りたたむには :

- 1 図上で、クラス (またはインターフェイス) を選択します。

2 区画の左端にある、「+」または「-」をクリックします。

図 8 クラス Store、Clerk、Item が作成されたクラス図



メソッドやフィールドの追加

VideoStore Inventory システムのためにユースケース図で抽出される要件定義のうち、今作成したクラスに追加する必要があるメソッドとフィールドのリストが、表 2 です。

表 2 VideoStore サンプル プロジェクトの要件定義

クラス	フィールド	メソッド
Store	address	findByTitle
	storeNumber	findByKeywords
	hours	findByItemNumber
	phone	
Clerk	employeeId	
	name	
Item	itemNumber	getItemNumber
	title	setItemNumber
		getTitle
		setTitle

クラスのメンバを作成するには：

- 1 (デフォルト) クラス図を開き、Store クラスの右クリック メニューから **追加 | フィールド** を選択します。代わりに、ダイアグラム ビューまたはモデル ビューで Store クラスを選択し、**[Ctrl] + [W]** を押しても構いません。
- 2 フィールド名をクリックして、直接編集可能にします。直接編集エディタを使用して、フィールドの名前を「address:string」に変更して下さい。図上で、直接編集エディタを使用してクラスメンバの名前を変更する際に、その型も一緒に入力することができます。

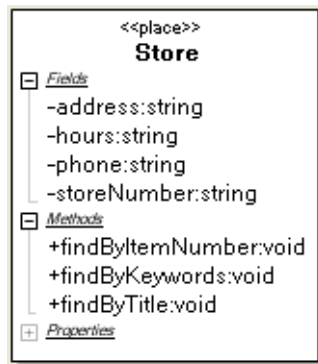
ヒント または、フィールドを選択して、キーボードから **[F2]** を押しても、直接編集可能な状態にすることができます。

- 3 プロパティウィンドウで **[可視性]** フィールドをクリックし、ドロップダウンリストから **[private]** を選択します。デフォルトで、Together VS .NET は、フィールドなら `public int` 型のものを、メソッドなら `public void` 型を返すものを生成します。

ヒント 一旦 **[可視性]** フィールドをクリックしたら、キーボードで **[Tab]** キーをクリックし、続いて下向き矢印を利用することにより、異なる可視性値をスクロールして選択していくことができます。

- 4 クラスにメソッドを追加するには、Store クラスの右クリックメニューから **[追加 | メソッド]** を選択します。代わりに、ダイアグラムビューまたはモデルビューで Store クラスを選択し、**[Ctrl] + [M]** を押しても構いません。
- 5 直接編集エディタを使用して、メソッドの名前を「findByTitle」に変更します。
- 6 ここまでのステップを繰り返し、表 2 に従い、Store クラスに残りのフィールドやメソッドを追加して下さい。作成中の Store クラスは、以下のようになります（図 9 参照）。

図 9 フィールドとメソッドを追加した Store クラス



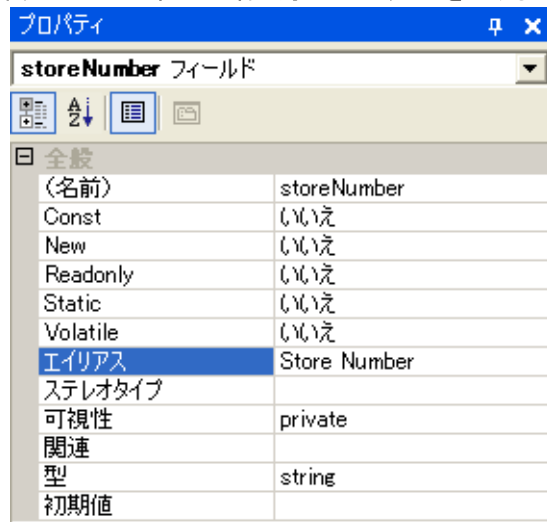
- 7 引き続き、表 2 にあるフィールドやメソッドを、Clerk クラスや Item クラスに対しても追加します。メソッド `getItemNumber` や `getTitle` に対しては、戻り値型として `string` を指定します。

ヒント ダイアグラムビューまたはモデルビュー上でフィールドやメソッドなどのクラスメンバを選択し、キーボードの **[Insert]** キーを使用すると、それと同じ型のメンバを、選択されているクラスに簡単に追加することができます。新しいメンバを追加した後、キーボードから **[F2]** を押して直接編集エディタを起動します。コンテキストメニューに表示されるショートカットキーも、メンバや図の要素を簡単に追加するのに使用することができます。

- 8 Store クラスの `storeNumber` フィールドを選択します。プロパティウィンドウで、**[エイリアス]** フィールドに「Store Number」と入力します（図 10 参照）。

ヒント **[エイリアス]** プロパティは、実際のコードで使用されている名前とは異なる名前を指定したい場合に、便利な機能です。

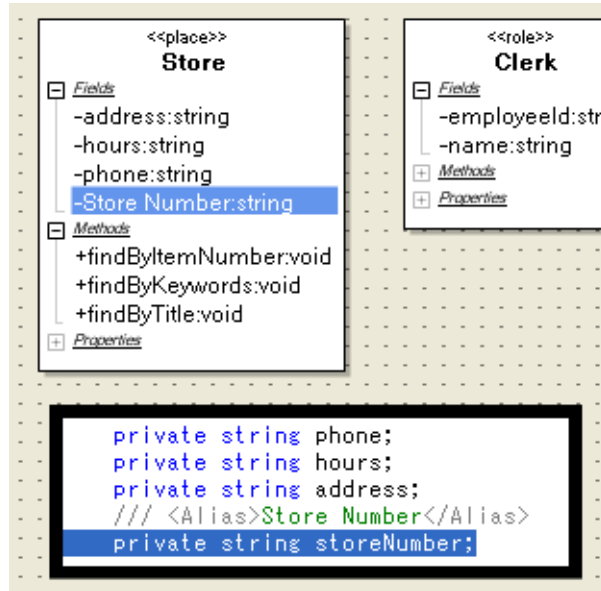
図 10 フィールド名に対してエイリアスをつける



エイリアスは、図上におけるフィールドの見た目のラベルのみを変更するだけのものです（図 11 参照）。フィールド自体は、ソースコードの中で、`storeNumber` として保持されます。

ヒント ソースコードを参照するには、そのクラスノードをダブルクリックするか、フィールドの右クリックメニューから **定義へ移動** を選択します。

図 11 storeNumber のエイリアスと、それに対応するコード



関係とリンクの作成

サンプルの VideoStore プロジェクトには、2 つの関係が含まれています。

- Clerk は、Store と関連付けられる
- Store は、複数の Items から構成される

関連リンクを作成することにより、Clerk と Store の間に、「クライアント」と「サプライヤ」の関係を設定することができます。2 つ目の関係としては、Store が複数の Items を持っているため、集約関係を作成します。[\[パターンからのリンク\]](#) ボタンを使って、コレクションとしての集約を簡単に実装することができます。

関係とリンクを作成するには：


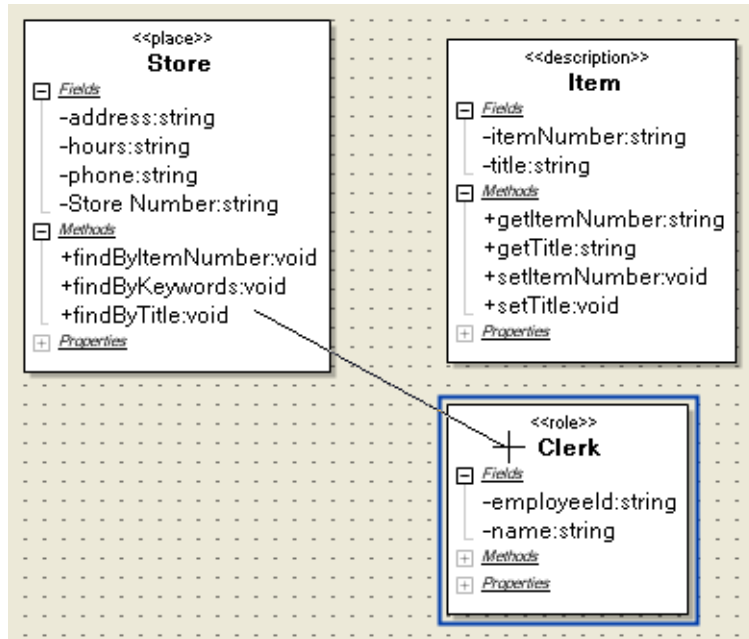
- 1 ツールボックスの **[関連]** ボタン  をクリックします。Store をクリックし、続いて Clerk へリンクをドラッグ&ドロップします。これにより、「クライアント」から「サプライヤ」への関係が設定されました。マウスをクラスの上へ持っていくと、Together VS .NET がリンクの末端を配置するのに適した要素として認識できたものが、青線の枠で示されることを確認しておきましょう（図 12 参照）。

図 12 ドラッグ&ドロップを利用した関連リンクの作成




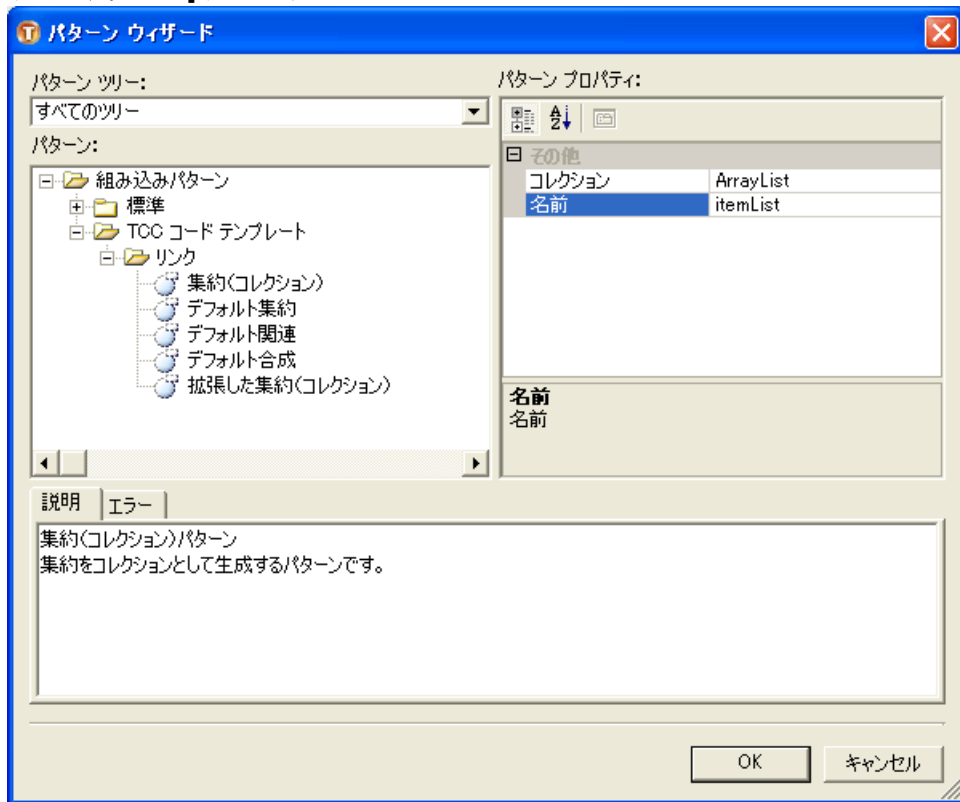
- 2 ツールボックスの **[パターンからのリンク]** ボタン  をクリックします。Store をクリックし、続いて Item へリンクをドラッグ&ドロップします。[パターンウィザード] ダイアログが開かれます。
- 3 [パターンウィザード] で、**[TCC コード テンプレート | リンク]** を展開し、**[集約 (コレクション)]** を選択します。
- 4 図 13 に示すように、フィールド名としてコレクションを示す「itemList」を入力し、**[OK]** をクリックします。

図 13 [パターン ウィザード] ダイアログ



Together VS .NET は、リンク情報をフィールドの上部にタグを配置する形で、ソースコードに自動的に追加します。このサンプルでは、作成したリンクは `Item` クラスへの集約リンクとなります。ダイアグラム ビューまたはモデル ビュー上で、クラスをダブルクリックすることにより、Visual Studio エディタにそのソースコードを開いて、これらのタグを確認してみてください。例えば、Store クラスの `itemList` フィールドは、次のようになります。

```

/// <link_link>aggregation</link_link>
/// <link_supplierCardinality>0..*</link_supplierCardinality>
/// <Associates>Item</Associates>

```

ハイパーリンクの作成

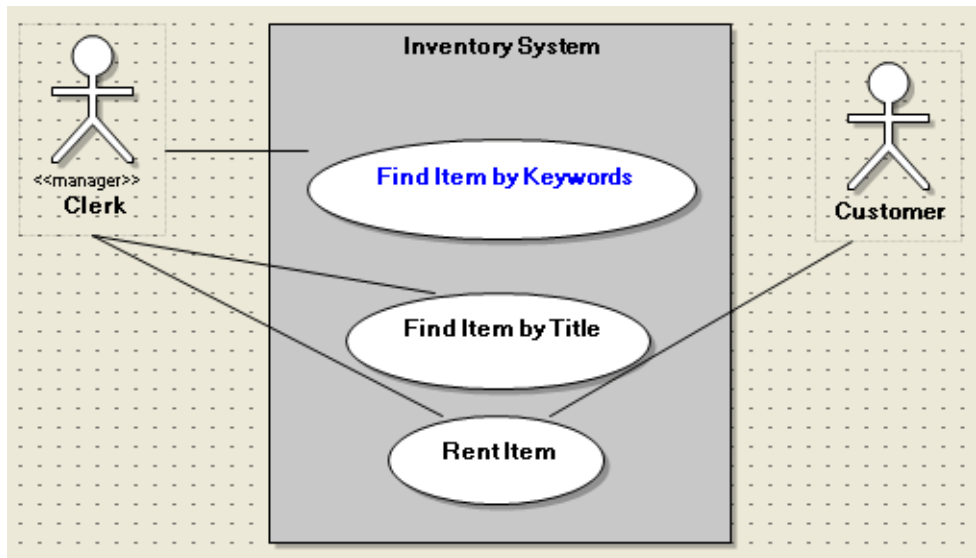
VideoStore には、`item` を検索するために使用するメソッド `findByKeywords` が用意されています。このメソッドのための要件が、先程作成したユースケース **Find Item by Keywords** として定義されています。ハイパーリンクを使用して図と要素をリンク付けることにより、図を利用する他のユーザーのために、モデル上でこれらの関係のタイプを表し、記録しておくことができます。

ユースケースとメソッド間にハイパーリンクを作成するには：

- 1 VideoStore User Case 図を開きます。
- 2 **Find Item by Keywords** ユースケースの右クリックメニューから **[ハイパーリンク | 編集]** を選択します。[ハイパーリンクの編集] ダイアログが開かれます。
- 3 **[モデル要素]** ペインで、**[VideoStore | Store | findByKeywords]** を選択します。
- 4 **[追加 >>]** をクリックして、その要素を右の **[選択済み]** ペインに追加し、**[OK]** をクリックします。

新たにハイパーリンクが作成されたこのユースケースは、**青色文字**でハイライトされています（図 14 参照）。

図 14 ハイパーリンクされた要素



- 5 ハイパーリンクをテストするには、ユースケースの右クリックメニューから **[ハイパーリンク | VideoStore.Store.findByKeywords]** を選択します。対応するメソッドがあるクラス図が開かれ、そのメソッドが選択されます。

メモ ハイパーリンクは双方向です。ハイパーリンクされたクラス図上のメソッドも**青色文字**でハイライトされます。ハイパーリンクされた要素の **[ハイパーリンク]** コマンドを使用すると、クラス図上のメソッドからユースケース図へ、またはその逆方向にも移動することが可能です。

シーケンス図の作成

シーケンス図は、オブジェクトモデルの動的な側面を設計するのに使用されます。
Together VS .NET は、シーケンス図を作成するための 3 つのモードを提供します。

- シンプルな「スケッチ帳」
- ソースコードとクラス双方向同期
- 既存のメソッドからシーケンス図を生成

初期シーケンス図の作成

ビデオの貸出に必要な処理をモデル化するために、VideoStore プロジェクト内にシーケンス図を作成することができます。

シーケンス図を作成するには：



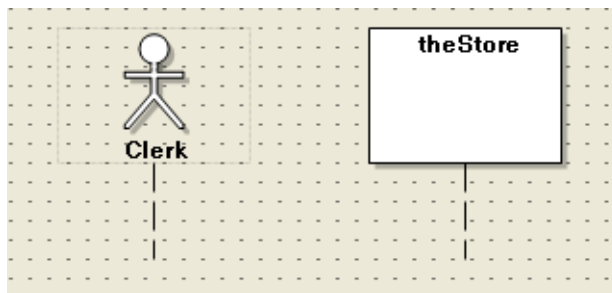

- 1 モデルビューで [VideoStore] プロジェクト ノードを右クリックし、[追加 | その他の図] を選択します。
- 2 その結果開かれたダイアログ ボックスで、[シーケンス図] を選択します。
- 3 その名前に「Rent Movie Sequence」と入力し、[OK] をクリックします。
- 4 ツールボックスから、[UML 相互作用図] を選択します。[アクター] ボタン  をクリックし、続いて図の背景をクリックします。
- 5 直接編集エディタを使用して、アクターの名前に「Clerk」と入力下さい。
- 6 ツールボックスから、[オブジェクト] ボタン  をクリックし、続いてアクターの線の右側をクリックして下さい。作成されたオブジェクトの名前を「theStore」とします（図 15 参照）。

図 15 アクター Clerk とオブジェクト Store



- 7 メッセージリンクを追加するために、[メッセージ] ボタン  をクリックし、続いて**アクターの生存線** から **theStore の生存線** までドラッグ & ドロップします。

オブジェクトへクラスを関連付ける

シーケンス図やコラボレーション図では、オブジェクトと分類子との間の関連を作成することができます。

メモ クラス、インターフェイス、構造体（C# プロジェクトのみ）、モジュール（Visual Basic .NET プロジェクトのみ）、列挙型、デリゲートを含むすべての分類子とオブジェクトを関連付けることができます。

オブジェクトをインスタンス化する分類子は、モデルから選択することができます。相互作用図では、オブジェクトの右クリックメニューから、次の 2 つのコマンドが利用できます。

- **[追加]** : モデルに分類子を新規作成します。
- **[クラスの選択]** : 利用可能な分類子のリストから選択します。

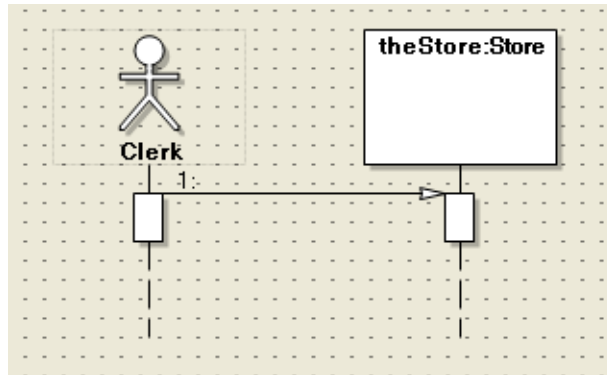
このサンプルでは、`theStore` オブジェクトを `Store` クラスと関連付けます。

オブジェクトへ分類子を関連付けるには :

- 1 `theStore` オブジェクトの右クリックメニューから、**[クラスの選択 | Store]** を選択します。その結果、シーケンス図は次のようになります（図 16 参照）。オブジェクト名の後ろに、関連付けられた分類子の名前が付けられます。

メモ 分類子をオブジェクトに関連付けた後、そのオブジェクトの右クリックメニューから、**[クラスの選択解除]** を選択すると、その関連を削除することができます。

図 16 Store クラスに関連付けられた `theStore` オブジェクト



新しいメソッドをメッセージリンクに追加する

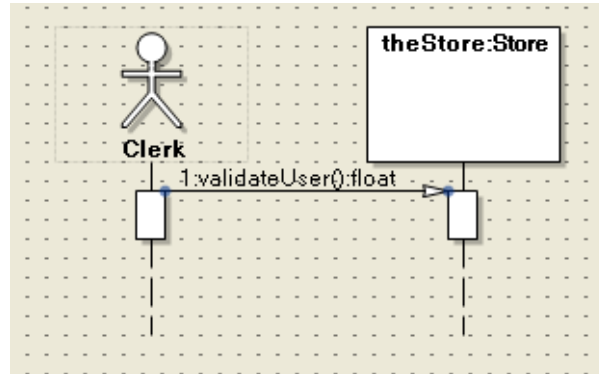
メッセージリンクを受信クラスのメソッドに関連付けることができます。この方法には、受信クラスの既存のメソッドからメソッドを選択するか、新しいメソッドを作成する方法があります。実際には、メッセージの右クリックメニューによって提供される 2 つのコマンド（**[追加]** および **[メソッドの選択]**）を使います。

このサンプルでは、`Store` クラスに新しいメソッド `validateUser` を作成します。

新しいメソッドをメッセージ リンクに追加するには：

- 1 メッセージリンクを右クリックし、[追加 | メソッド] を選択します。Method1():void という名前の新しいメソッドが Store クラスに作成されます。
- 2 メッセージリンクのプロパティ ウィンドウで、[オペレーション] フィールドを選択し、新しいメソッドの名前 validateUser():float を入力して変更します。
- 3 [オペレーション名の変更] ダイアログが開き、Store クラスに validateUser メソッドが存在しないことが示されます。このダイアログで、新しいメソッドを作成するか、既存のメソッド名を変更するか選択することができます。[名前の変更] を選択して、既存のメソッド名を変更します。その結果、図は次のようになります（図 17 参照）。

図 17 validateUser メソッド



メッセージリンクの右クリック メニューの [メソッドの選択] コマンドを使って、メッセージリンクをオブジェクトの受信クラスの既存のメソッドに関連付けることもできます。

メモ メソッドをメッセージリンクに関連付けた後、そのメッセージリンクの右クリック メニューから [メソッドの選択解除] を選択すると、関連を削除することができます。


既存のソース コードからシーケンス図を生成する

シーケンス図とコラボレーション図を作成したら、ツールボックスのボタンを使って図を記述することができます。また、クラス図のメソッドからシーケンス図を生成することもできます。

メモ この機能は C# プロジェクトに対してのみ利用可能です。

このサンプルでは、Video Store プロジェクトを使って簡単なシーケンス図を生成します。

シーケンス図を生成するには：

- 1 VideoStore の (デフォルト) クラス図 ノード  をダブルクリックします。ダイアグラム ビューに、(デフォルト) クラス図が開かれます。
- 2 Store クラスの findByKeywords メソッドを右クリックし、[シーケンス図の生成 ...] を選択します。
- 3 [シーケンス図の生成] ダイアログが開きます。[OK] をクリックします。

シーケンス図が生成され、ダイアグラムビューで開かれます。

シーケンス図への変更を保存せずに図を閉じることもできます（この場合、図は空になります）。

ヒント

メソッドからコラボレーション図を生成するには、まずシーケンス図を生成し、その後、図をコラボレーション図に変換します。詳細については、「[シーケンス図とコラボレーション図間の変換](#)」を参照して下さい。

シーケンス図とコラボレーション図間の変換

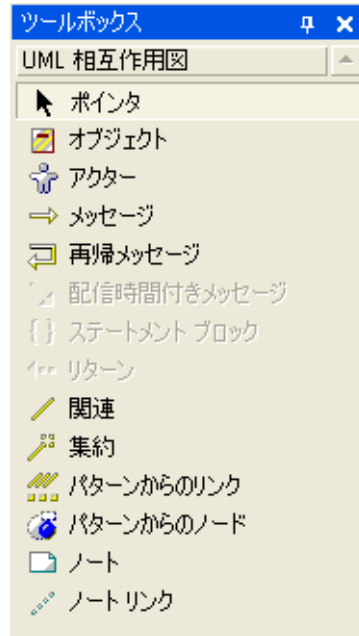
シーケンス図とコラボレーション図間で、相互に変換することができます。ただし、Together VS .NET で [新しい図の追加] ダイアログを使用して新しい図を作成する際には、それがシーケンス図かコラボレーション図のどちらであるかを、指定しなければなりません。

シーケンス図 Rent Movie Sequence をコラボレーション図へ変換するには：

- 1 Rent Movie Sequence シーケンス図を開きます。
- 2 図の背景の右クリック メニューから [**コラボレーションとして表示**] を選択します。コラボレーション図が開かれます。
- 3 ツールボックスを開きます。[表示] メニューから [**ツールボックス**] を選択して下さい。
- 4 ツールボックスでは、[**UML 相互作用図**] をクリックし、コラボレーション図の要素を表示させます（図 18 参照）。

シーケンス図とコラボレーション図は [UML 相互作用図] ツールボックスを共有していますが、それぞれの図のタイプに即した要素のみが有効になっている点に注目して下さい。

図 18 コラボレーション図の要素



- 5 図の背景を右クリックし、**[レイアウト | レイアウトの実行]**を選択します。
- 6 シーケンス図に再び戻すには、このコラボレーション図の背景を右クリックし、**[シーケンス図として表示]**を選択して下さい。図がシーケンス図として表示されます。

モデル ビューの利用

Together VS.NET では、図の要素を管理するために、モデル ビューを基本的にナビゲーション ビューとして使います。モデル ビューのコンテキスト メニューを使用することにより、図を開いたり、図の要素を作成 / 削除することができます。

モデル ビューは、数多くのアイコンやコンテキスト メニューから構成されており、ユーザーはこれに慣れていく必要があります。新たな要素や図を見つけた際には、そのコンテキスト メニューを確認していくと良いでしょう。

プロジェクトのルート ノードのコンテキストを利用することにより、Together VS.NET の次の機能にアクセスすることができます。

- プロジェクトへの図の追加
- 図の要素（名前空間、クラス、インターフェイス、など）の追加
- コードのリファクタリング
- C# プロジェクトに対する検査の実行

参照の表示

プロジェクトに含まれる参照に対してクラス図をモデルビューに表示することができます。参照は、ソリューション エクスプローラを使って追加することができます。

Video Store プロジェクトで MsCorLib.dll を表示するには：

- 1 モデル ビューで、**[参照設定]** ノードを展開し、さらに、**[MsCorLib.dll]** ノードを展開します。
- 2 (デフォルト) 図で右クリックし、**[図を開く]** を選択します。ダイアグラム ビューに、(デフォルト) 図が開かれます。

[Microsoft] や [System] フォルダを展開して、他のクラス図と同様に表示することもできます。

ファイル メニュー (**[ファイル | 印刷 ...]** または **[ファイル | 図を画像にエクスポート ...]**) から、図を印刷したり画像ファイルにエクスポートしたりすることができます。

展開可能な図ノードの表示

デフォルトでは、モデル ビューにおける図のノードは、展開可能な状態になっています。モデル ビューにおけるノードを展開してその中の要素を表示できるようにするかどうか、制御することができます。

図のノードを展開可能にするには：

- 1 [ツール] メニューから、**[オプション]** を選択します。[オプション] ダイアログが開かれます。
- 2 **[Together VS .NET]** フォルダをクリックして、Together VS .NET 固有の設定オプションを参照します。オプション [一般] が表示されています。
- 3 一覧からオプション カテゴリ **[モデル ビュー]** を選択します。
- 4 **[図のノードを展開可能にする]** フィールドをクリックします。ドロップ ダウン矢印が現れます。
- 5 ドロップダウン矢印をクリックし、一覧から **[はい]** を選択します。
- 6 **[OK]** をクリックすると、変更が適用されダイアログが閉じられます。

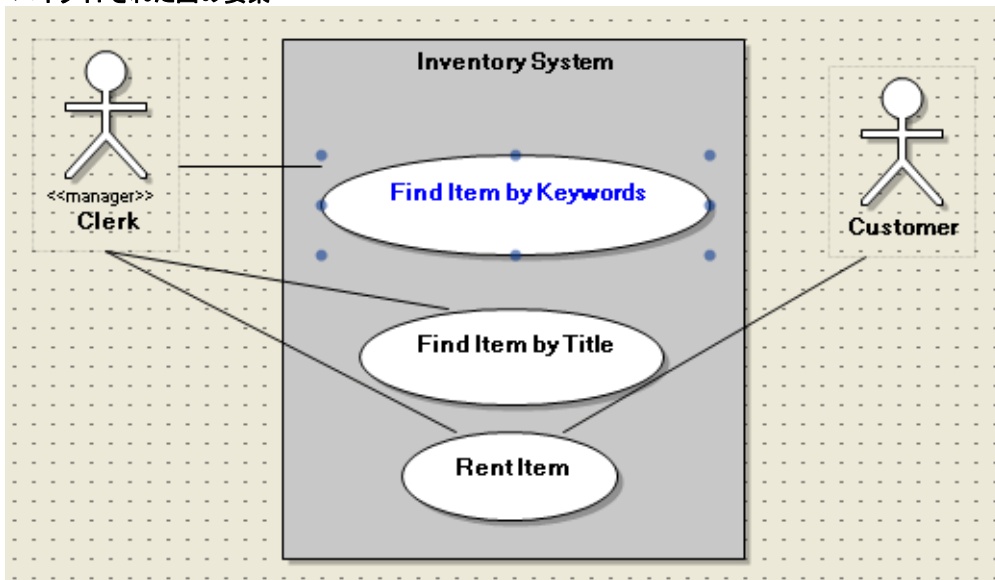
モデル ビューの各図のノードは、展開可能な形で表示されます。**VideoStore Use Case** 図のノードを展開し、その要素を参照してみてください。

図の要素の検索

ただし、モデルビューで図の要素を選択しても、その要素がダイアグラムビューで自動的に選択される訳ではありません。モデルビューで選択されている要素をダイアグラムビューに表示させるには、モデルビューでその要素の右クリックメニューから**【図で選択】**を選択して下さい。このコマンドを使用することにより、その要素が存在する図が開かれ、図上で要素がハイライトされます（図 19 参照）。

同様に、ダイアグラムビューで作業中に**【モデルビューの同期】**コマンドを使って、モデルビューの要素に直接ナビゲートすることができます。図の要素の右クリックして、コンテキストメニューから**【モデルビューの同期】**を選択します。

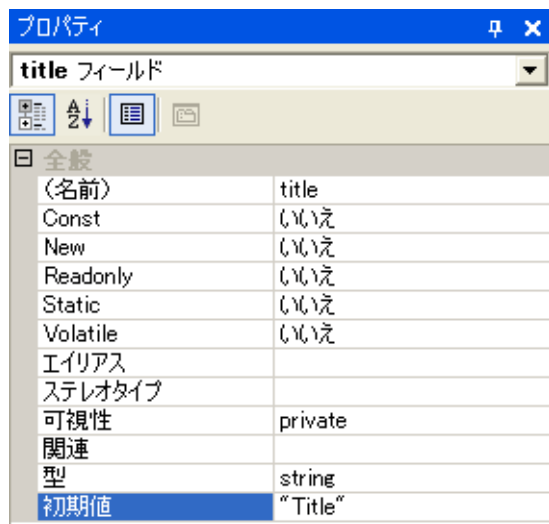
図 19 ハイライトされた図の要素



プロパティ ウィンドウの利用

プロパティ ウィンドウを使用すると、[エイリアス]、[名前]、[可視性]、その他 図 20 に見られるような値の、確認や編集を行うことができます。

図 20 プロパティ ウィンドウ



Item クラスの詳細情報を追加するには：

- 1 (デフォルト) クラス図を開き、Item クラスのフィールド title を選択します。
- 2 プロパティ ウィンドウを開くには、[表示] メニューから [プロパティ ウィンドウ] を選択します。または、[F4] を使用して下さい。
- 3 フィールド [初期値] には、「"Title"」(引用符「"」も含みます) と入力して下さい (図 20 参照)。

この変更を反映するように、ソース コードも自動的に更新されます。ダイアグラム ビューからソース コードへは、簡単に移動することができます。

ソース コード エディタを開いて、フィールド title の部分を参照するには：

- 1 図上で、クラス Item のフィールド title を右クリックします。
- 2 その右クリック メニューから [定義へ移動] を選択します。

ソース コード エディタが開かれ、該当するコード部分がハイライトされます。フィールド title の初期値は、プロパティ ウィンドウで指定されている点に注目して下さい。

概要図の利用

ダイアグラムビューの概要図には、現在開いている図のサムネイルビューが表示されます。[概要]ボタンは、すべての図の右下隅に配置されています（図 21 参照）。

図 21 [概要]ボタン

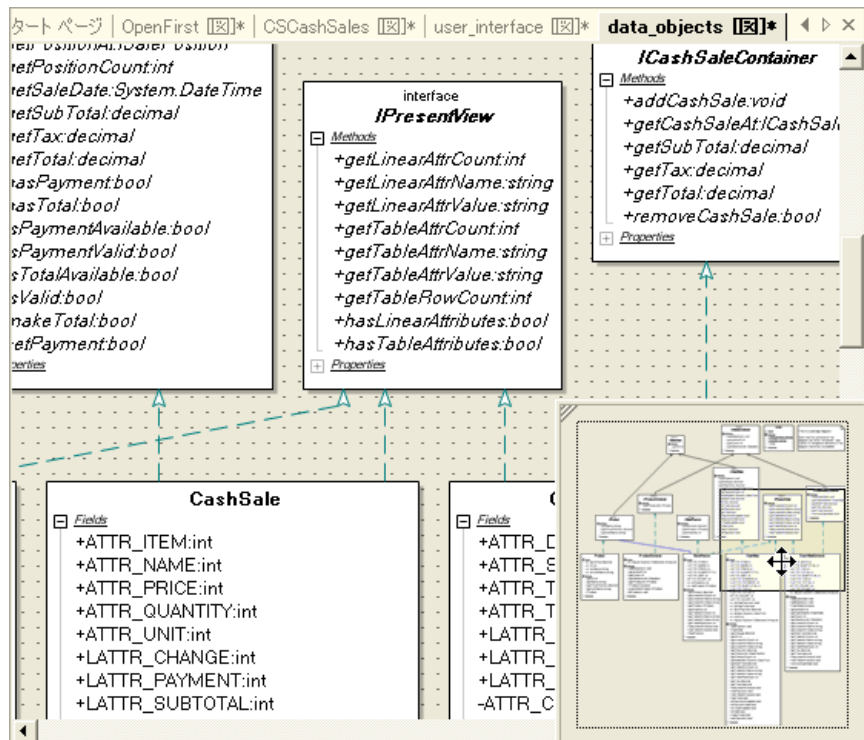


概要機能を利用するには：

- 1 (デフォルト) クラス図上で、[概要]ボタンをクリックします。ペインが広がり、そこに現在開いている図のサムネイル画像が表示されます。
- 2 マウスで、影の領域をクリックし、それを自由にドラッグして下さい。図の表示領域をスクロールする際に、便利な機能です。
- 3 [概要]ペインの左上隅をつかんでドラッグすると、ペインのサイズを変更することができます。

[概要]ペインは、図の他の要素を選択すると、自動的に閉じられます。図 22 は、大きなクラス図の[概要]ペインの様子です。

図 22 大きなクラス図のサムネイル



ダイアグラム ビューの管理

[表示管理] では、Together VS .NET において、どの図や要素が表示されるのかを制御します。

情報の表示 / 非表示

大規模なプロジェクトを扱う場合、図に表示される情報量は膨大なものとなってしまいます。このため、Together VS .NET では、表示 / 非表示する情報を選択できるようになっています。

ダイアグラム ビュー全体に対するグローバルな制御には、[オプション] ダイアログでのフィルタを使用します。[ツール] メニューから、[オプション] を選択して下さい。
[Together VS .NET] フォルダをクリックし、[表示管理] ノードを選択します (図 23 参照)。

図 23 表示管理のためのオプション ダイアログ

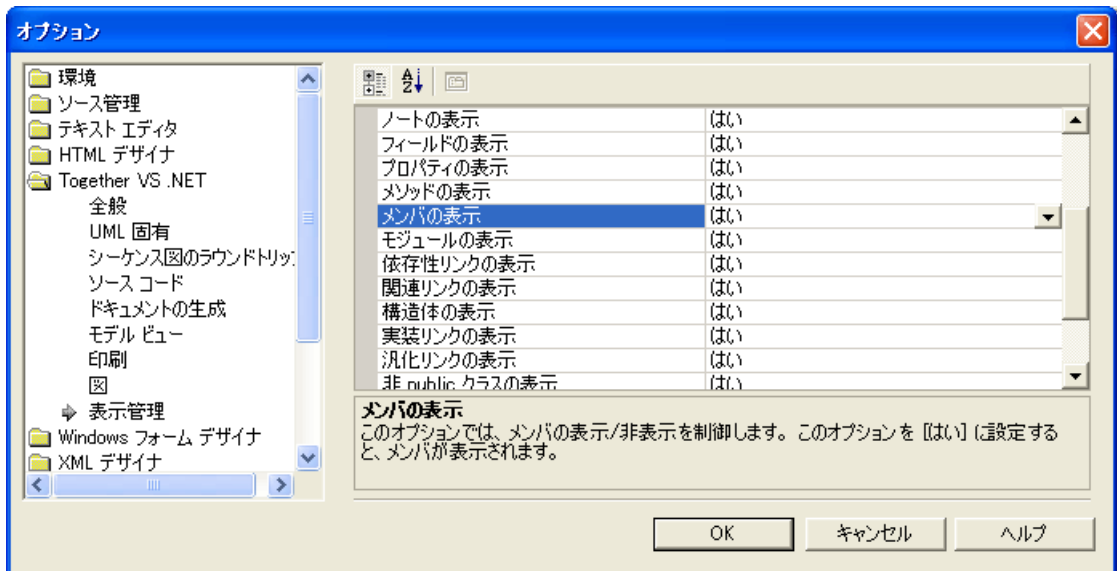


図 23 に表示されているフィルタは、グローバル フィルタです。メンバに対してのみフィルタリングしたい場合には、プロパティ [メンバの表示] を [いいえ] に設定します。

メンバをフィルタリングするには：

- 1 フィルタ [メンバの表示] をクリックします。
- 2 ドロップダウン矢印をクリックし、[いいえ] を選択します。
- 3 [OK] をクリックします。

これにより、フィールド、メソッド、非 public メンバ、および内部分類子に対するフィルタが無効になります。

内部分類子はコンテナ要素のメンバとして扱われるので、次のフィルタは内部分類子をフィルタリングしません。


- クラスの表示
- デリゲートの表示
- 列挙型の表示
- インターフェイスの表示
- 構造体の表示

メモ 現在、構造体要素は C# プロジェクトでのみ利用できます。

[非表示要素の表示] ダイアログの利用

[非表示要素の表示] ダイアログを使って、図の情報の表示 / 非表示を選択することもできます。次の例では、Video Store プロジェクトの (デフォルト) クラス図の情報を表示 / 非表示する方法を説明します。

要素を表示 / 非表示するには :

- 1 モデル ビューで VideoStore の (**デフォルト**) クラス図 ノード  をダブルクリックします。ダイアグラム ビューに、(デフォルト) クラス図が開かれます。
- 2 ダイアグラム ビューで、Store クラスを右クリックして、コンテキスト メニューから、[**非表示**] を選択します。Store クラスが図に表示されなくなります。
- 3 Store クラスを表示するには、図の背景を右クリックして、コンテキスト メニューから、[**表示 / 非表示 ...**] を選択します。[**非表示要素の表示**] ダイアログが開きます。
- 4 Store クラスは、ダイアログの右側の [**非表示要素**] リストに表示されます。Store クラスを選択し、[**<< 削除**] をクリックしてから、[**OK**] をクリックします。すると、Store クラスは再び図に表示されます。

第 3 章

Together VS .NET 機能の利用

この章では、Borland Together Edition for Microsoft Visual Studio .NET (Together VS .NET) 独自機能の利用方法について説明します。より総合的な情報については、Together VS .NET のオンライン ヘルプを参照して下さい。

この章では次のトピックについて説明します。

- 「パターンの利用と作成」(37 ページ)
- 「品質保証の実行」(40 ページ)
- 「ドキュメント生成」(47 ページ)
- 「XMI プロジェクトのインポート/エクスポート」(50 ページ)

パターンの利用と作成

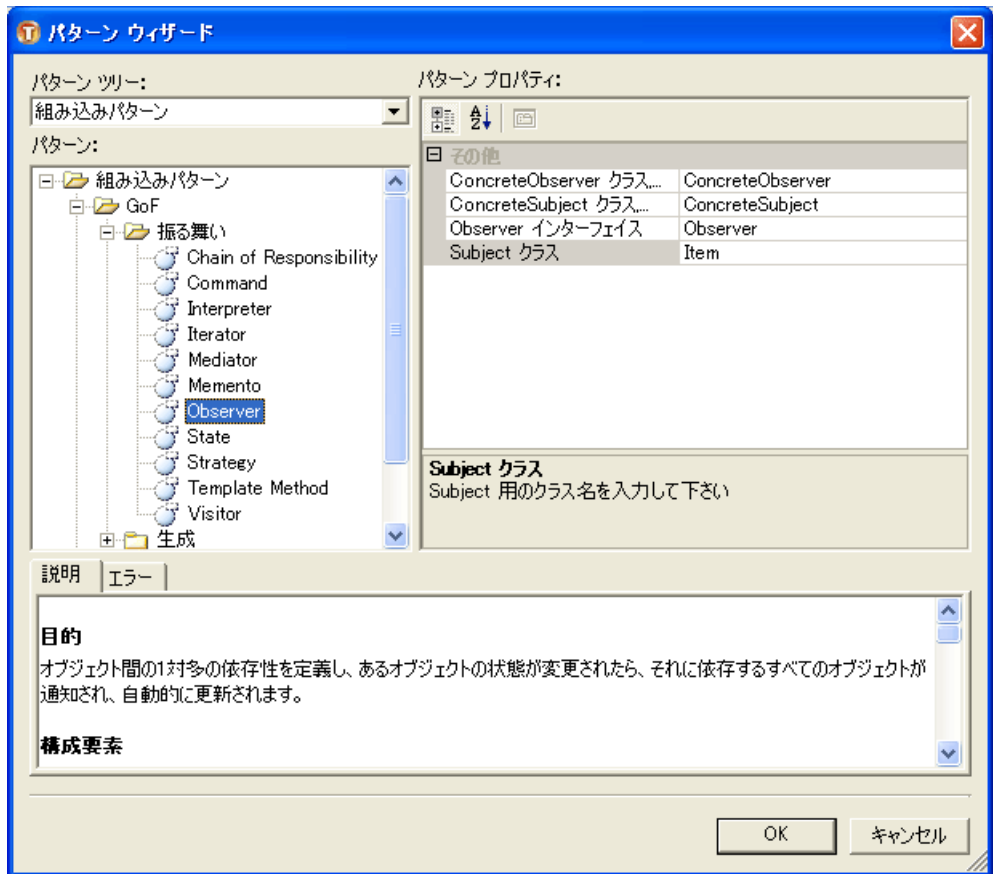
Together VS .NET は、GoF パターンのように、頻繁に使用されるパターンをサポートしています。これらのパターンを使用して、既成のリンクやクラスを作成、修正することができます。

以降の例では、クラス `Item` が動的なシステムであり、`Item` が返ってくると `Counter` が更新され、他のシステムに通知する必要がある、と想定します。このようなシステムの設計には、Observer パターンが便利です。Observer パターン情報を `Item` クラスへ適用するには、パターン ウィザードを使用します。

パターン ウィザードを使用するには：

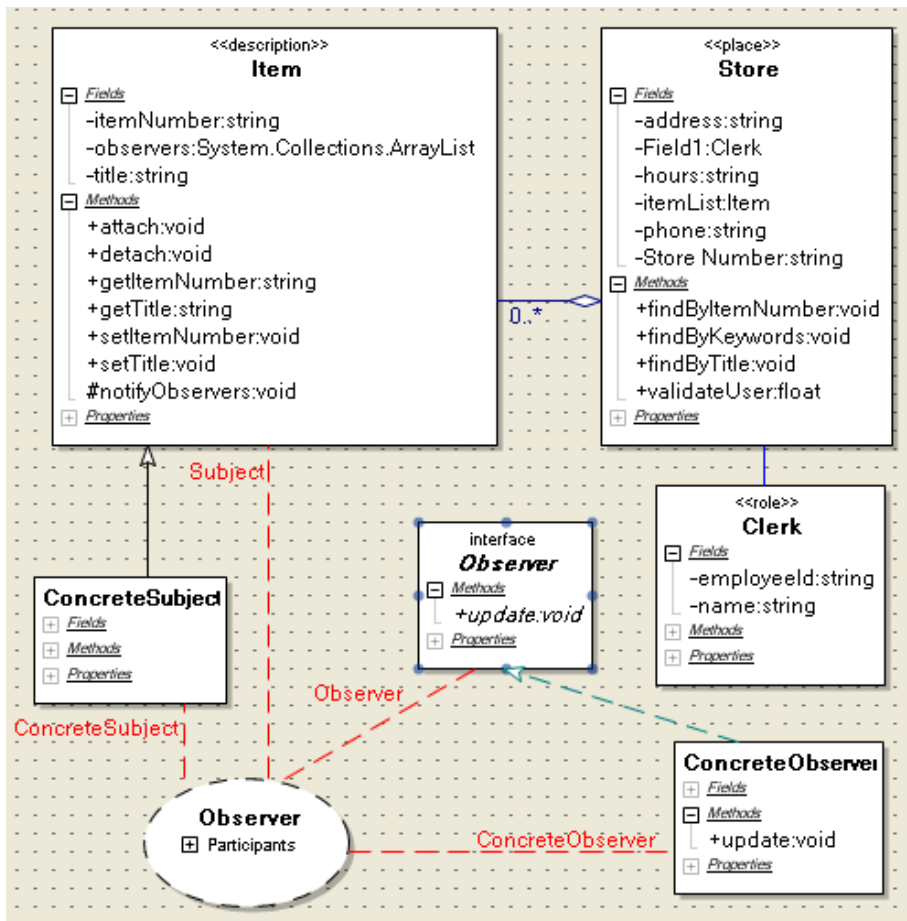
- 1 Video Store クラス図の背景を右クリックし、**[パターンの適用]**を選択します。[パターン ウィザード] ダイアログが開かれます (図 24 参照)。

図 24 パターン ウィザード



- 2 左の **[パターン]** ペインから、パターン **[GoF | 振る舞い | Observer]** を選択します。
- 3 右側の **[パターン プロパティ]** ペインで、**[Subject クラス]** の名前を、Item に変更します。
- 4 他のプロパティはデフォルト設定を利用するので、そのまま **[OK]** をクリックします。
図が次のように更新されます（図 25 参照）。

図 25 Observer パターンの適用



Item は、通知メソッドや監視メソッド（attach と detach）によって更新されています。r その他、クラスやインターフェイスも、すぐ使用できる状態に作成されています。Together VS .NET は、パターンを認識し、その要素を図に楕円形で表示する他、パターンの参加要素（participants）の一覧やリンクも、追加します。Observer 要素上にある **Participants** ノードを展開すると、パターン参加要素の情報を参照することができます。

図上のパターン要素は、それぞれ固有のアクションを持っており、ユーザーは、パターンに関連づけるアクションを選ぶことができます。楕円形の Observer 要素を右クリックメニューから **[追加]** を選択し、その要素で使用する事ができるパターンアクションを参照します。

品質保証の実行

C# プロジェクトに対して品質保証（QA 検査）を実行することによって、あらかじめ定義された尺度のセットを基準としてコードの品質を検査することができます。

メモ QA 検査機能は C# プロジェクトに対してのみ利用可能です。

検査の実行

検査機能は、企業標準や慣習を強要することなく自然に徹底させ、作業を改善させるのに役立ちます。検査を実行する際には、ソースコードが準拠すべき特定の規則を選択します。結果として選択された規則の違反箇所が表示されるので、それぞれの問題を検討し、ソースコードを修正するかどうかを決定します。Together VS .NET では、設計上の問題から命名規則に至るまで、幅広い様々な検査項目と、それらの項目についての説明、および違反の修正方法が提供されています。

Video Store プロジェクトに対して検査を実行するには：

- 1 Video Store クラス図の背景を右クリックし、**[QA 検査 ...]**を選択します。**[QA 検査]**ダイアログが開きます。**[対象範囲]**ドロップダウンリストから**[モデル]**を選択します。
- 2 ダイアログの左側で、検査カテゴリ（コーディングスタイル、宣言スタイル、など）のノードを展開して、利用可能な検査項目を表示し、実行したい検査項目をチェックします。検査項目を選択すると、ダイアログの下側のペインに説明が表示されます。
- 3 検査項目それぞれに対して、重要度のレベルやその他検査項目固有のオプションがダイアログの右側のパネルに表示されます。必要に応じて設定を変更して下さい。
- 4 このサンプルでは、デフォルトの設定を利用するので、**[開始]**をクリックします。**[操作を実行しています ...]**ダイアログが開き、処理が完了するまで進捗状況を示すステータスバーが表示されます。検査結果は、図 26 のようになります。

図 26 検査ビュー（結果は異なる場合があります）

検査					
VideoStore					
略語	説明	重...	リソース	ファイル	行
RCD	public クラス Clerk は public メンバあ...	警告	Clerk	C:\Documents ...	2
MNU	フィールド employeeId が使用されてい...	警告	Clerk	C:\Documents ...	3
MNU	フィールド name が使用されていません	警告	Clerk	C:\Documents ...	4
ULVFP	形式パラメータ subject は使用されて...	警告	Concret...	C:\Documents ...	2
RCD	public クラス ConcreteSubject は pu...	警告	Concret...	C:\Documents ...	1
MNU	フィールド itemNumber が使用されて...	警告	Item	C:\Documents ...	3
MNU	フィールド title が使用されていません	警告	Item	C:\Documents ...	4
MNU	フィールド phone が使用されていません	警告	Store	C:\Documents ...	3
MNU	フィールド hours が使用されていません	警告	Store	C:\Documents ...	4
MNU	フィールド address が使用されていま...	警告	Store	C:\Documents ...	5
MNU	フィールド storeNumber が使用されて...	警告	Store	C:\Documents ...	7
MNU	フィールド Field1 が使用されていません	警告	Store	C:\Documents ...	21
MNU	フィールド itemList が使用されていま...	警告	Store	C:\Documents ...	25

検査結果ビューでの作業

検査結果ビューを使って、次のような作業を実行することができます。

- 違反した検査項目に対応したソースコードを開く
- 違反した検査項目の説明を表示
- 検査結果の印刷と保存
- 検査結果の並び替えとグループ化
- 現在表示されている違反した検査項目の更新（再計算）
- 検査の再実行

検査結果ビューはドッキング可能なウィンドウです。検査結果は最初、自由に移動可能なフローティングウィンドウとして開かれます。しかし、フローティング形式のままでも、ドッキング領域へドッキングしても構いません。検査結果のドッキング領域は、Visual Studio .NET ウィンドウの 4 辺どれを使用しても構成することができます。好きな場所に検査結果ウィンドウを配置して下さい。

ソースコードを開き、検査項目の説明を表示する

検査結果は図の要素とソースコードと密接に関係しています。検査結果から、実際に違反しているソースコードの特定の場所にナビゲートすることができます。


検査項目の違反に対応するソース コードを開き、説明を表示するには：

- 1 検査ビューで 3 番目の「フィールド name が使用されていません」をダブル クリックします。Together VS .NET は、該当ソースを Visual Studio エディタで開き、その行をハイライトします。
- 2 検査ビューで、「フィールド name が使用されていません」を右クリックして、[説明の表示] を選択します。「Member is Not Used (MNU)」検査項目の説明が表示されます。

検査結果の印刷

違反した検査項目全体や選択した特定の行を印刷することができます。

検査結果を印刷するには：


- 1 印刷したい行を選択します。リスト全体を印刷したい場合は、場にも選択する必要はありません。
- 2 ツールバーの印刷ボタン  をクリックします。[検査結果の印刷] ダイアログが開きます。
- 3 [出力項目の選択] コンボボックスを使って印刷する結果の範囲を選択します([すべての結果]、[選択したグループ]、[選択した行])。検査結果は検査ビューでタブ付きのページに表示されます。検査結果のコンテキストメニューの [グループ化 ...] コマンドを使って結果をグループ化したり、解除したりできます。
メモ： [グループ化] コマンドを使って結果をグループしない限り、[選択したグループ] オプションを選択することはできません。[出力項目の選択] オプションは次の通りです。
 - [すべての結果]：結果がグループ化されている場合、現在のタブ付きページのすべてのグループに対するレポートが印刷されます。結果がグループ化されていない場合は、現在のタブ付きページのすべての結果が印刷されます。
 - [選択したグループ]：結果がグループ化されている場合、現在のタブ付きページで選択されているグループの結果が印刷されます。
 - [選択した行]：検査結果ビューで、1 つまたは複数の行を選択することができます。[選択した行] を選択すると、このような選択した行のみの結果が印刷されます。
- 4 必要に応じて、[印刷倍率] フィールドで、印刷倍率の因数を指定したり、[ページに合わせる] をチェックして 1 ページ内に結果を印刷したりできます。これをチェックすると、[印刷倍率] フィールドは無効になります。
- 5 ページやプリンタの設定を調整することもできます。
 - [印刷] コンボボックスをクリックし、[印刷設定] コマンドを選択してプリンタを選択します。
 - メインメニューから [ツール | オプション] を選択して [オプション] ダイアログを開き、[Together VS .NET | 印刷] を選択します。このオプションでは、[用紙サイズ]、[印刷方向]、[マージン]などを設定できます。
 - [プレビュー] オプションの右の下向きの矢印をクリックすると、プレビュー ペインが開きます。[プレビュー ズーム (自動)] スライダーや、[自動調整] チェックボックスを使って、表示倍率を調整できます。[プレビュー] オプションの右の上向きの矢印をクリックすると、プレビュー ペインが閉じます。

- 6 **[印刷]** をクリックすると、システムの印刷ダイアログが開き、プリンタに印刷できます。

検査結果の保存

検査結果を XML ファイルや HTML ファイルにエクスポートし、チームのメンバと結果を共有したり後でレビューしたりすることができます。

検査結果を保存するには：

- 1 保存したい行を選択します。リスト全体を保存したい場合は、場にも選択する必要はありません。
- 2 ツールバーの保存ボタン  をクリックします。
- 3 **[検査結果の保存]** ダイアログが開きます。[出力項目の選択] コンボボックスを使ってエクスポートする結果の範囲を選択します (**[すべての結果]**、**[選択したグループ]**、**[選択した行]**)。検査結果は検査ビューでタブ付きのペインに表示されます。検査結果のコンテキストメニューの **[グループ化 ...]** コマンドを使って結果をグループ化したり、解除したりできます。
メモ：**[グループ化]** コマンドを使って結果をグループしない限り、[選択したグループ] オプションを選択することはできません。[出力項目の選択] オプションは次の通りです。
 - **[すべての結果]**：結果がグループ化されている場合、現在のタブ付きペインのすべてのグループに対するレポートが生成されます。結果がグループ化されていない場合は、現在のタブ付きページのすべての結果が出力されます。
 - **[選択したグループ]**：結果がグループ化されている場合、現在のタブ付きページで選択されているグループの結果が出力されます。
 - **[選択した行]**：検査結果ビューで、1 つまたは複数の行を選択することができます。**[選択した行]** を選択すると、このような選択した行のみの結果が出力されます。
- 4 **[出力形式の選択]** コンボボックスでは、エクスポートするファイルの形式を選択します。
 - **[XML]**：XML 形式のレポートを生成します。
 - **[HTML]**：HTML 形式のレポートを生成します。
[HTML] 形式を選択すると、チェックボックス (**[説明を追加]**、および **[ブラウザを起動]**) が有効になります。
 - **[説明を追加]**：このオプションをチェックすると、別のフォルダにある検査項目の説明へのリンクが検査結果のファイルに保存され、ナビゲートできるようになります。
 - **[ブラウザを起動]**：このオプションをチェックすると、生成した HTML ファイルがデフォルトビューアで開かれます。
- 5 **[保存]** をクリックして、指定した場所に結果を保存します。



検査結果の並び替えとグループ化


検査結果を表示しているとき、結果のレポートで項目を比較したり整理したりしたい場合があるでしょう。次のようにして結果を並び替えたりグループ化したりできます。

- **行での並び替え**：特定の行の値に従ってすべての項目を並べ替えるには、行の見出しをクリックします。一度クリックすると昇順に並べ替えられます。2 度目のクリックでは、降順に並べ替えられます。
- **検査結果のグループ化**：現在の行に従って項目をグループ化するには、検査結果の表上で右クリックし、[**グループ化 ...**] を選択します。これによって、行と列の関係を変更して結果を構成することができます。結果のグループ化を解除するには、表を右クリックして [**グループ化解除**] を選択します。

検査結果の更新と再実行

ツールバーを利用して結果の表を更新あるいは再実行することができます。


- **[更新]**  をクリックすると、現在表示されている結果を再計算します。
- **[再実行]**  をクリックすると、[QA 検査] ダイアログが開き、必要に応じて設定を変更してから検査を再実行することができます。現在表示されている結果は新しい結果で置き換えられます。

ダイアログの右上隅にある閉じるボタン  をクリックすると、検査ビューが閉じます。

特定の要素を選択して検査を実行する

Together VS .NET の検査機能を使って、どのクラスに対して検査を実行するか選択することができます。

特定のクラス要素に対して検査を実行するには：

- 1 Video Store クラス図を開き、Item クラスを右クリックして、[**QA 検査 ...**] を選択します。これによって、検査の [**選択範囲**] を Item クラスに限定することができます。
- 2 他のすべてのプロパティはデフォルト設定のまま、[**開始**] をクリックします。検査結果ビューが開きます。プロジェクトに対して検査を実行した場合と同じく、タブ付きページ形式で検査結果ビューに結果が表示されます。最新の実行結果がフォーカスを持っています。
- 3 対応するタブを右クリックして、[**閉じる**] を選択することで、検査結果の個々のタブ付きページ毎に閉じることができます。ダイアログの右上隅にある閉じるボタン  をクリックすると、検査ビューが閉じます。

リファクタリング

リファクタリングとは、外部に対する振る舞いを変更することなく、その設計を改善することを目的として既存のソースコードを書き換えることを意味します。リファクタリングではソースコードの構造に焦点を当て、コードが理解、保守、修正しやすくなるように設計を変更していきます。

Together VS .NET は次のリファクタリング コマンドをサポートします。



- **名前の変更**：コード生成要素の名前を変更します。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **移動**：static メソッドを他のクラスに移動します。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **インターフェイスの抽出**：既存のクラス、構造体、メソッド、プロパティ、イベント、インデクサから新しいインターフェイスを作成します。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **スーパークラスの抽出**：既存のクラス、インターフェイス、メソッド、プロパティ、イベント、フィールド、インデクサから新しいスーパークラスを作成します。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **メンバのプル アップ**：クラス階層の上位に選択したメンバを移動します。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **メンバのプッシュ ダウン**：クラス階層の下位に選択したメンバを移動します。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **安全な削除**：要素を削除する前にその使用箇所を表示します。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **パラメータの変更**：選択したメソッドの既存のパラメータを変更したり、新しいパラメータを追加したりします。ダイアグラム ビュー、モデル ビュー、Visual Studio エディタで利用可能です。
- **メソッドの抽出**：Visual Studio エディタ上で完結した文の集合を抽出して新しいメソッドを作成します。Visual Studio エディタでのみ利用可能です。
- **変数のインライン化**：Visual Studio エディタ上でローカル変数を選択して変数をインライン化します。Visual Studio エディタでのみ利用可能です。
- **フィールドの導入**：単一の完結した式から新しいフィールドを作成します。Visual Studio エディタでのみ利用可能です。
- **変数の導入**：単一の完結した式から新しい変数を作成します。Visual Studio エディタでのみ利用可能です。

これらのコマンドは、メイン メニューの **[リファクタリング]** メニューから、あるいはクラス図のコード生成要素のコンテキスト メニューの **[リファクタリング | <メニュー コマンド>]** コマンドから利用することができます。これらのコマンドに関するより詳細な情報は、Together VS .NET オンライン ヘルプを参照して下さい。

名前の変更

クラス、インターフェイス、メソッド、フィールド、パラメータ、ローカル変数など、任意のコード要素の名前を変更することができます。Together VS .NET は、名前の変更をプロジェクト内で依存しているコードにも伝播させます。

Item クラスに対して [名前の変更] コマンドを使用するには：


- 1 ダイアグラム ビューで `Clerk` クラスを右クリックして、**[リファクタリング | 名前の変更 クラス "Clerk"]** を選択します。[クラス名の変更] ダイアログが開きます。
- 2 新しい名前として、ダイアログの**[新しい名前]** フィールドに `Employee` と入力します。
- 3 オプション **[リファクタリング前に参照を表示]** がチェックされていることを確認します。このオプションを有効にして **[名前の変更]** をクリックすると、[リファクタリング] ウィンドウが開き、変更を確認してから適用することができます。このオプション **[リファクタリング前に参照を表示]** がチェックせずに **[名前の変更]** をクリックすると、[リファクタリング] ウィンドウは開きますが、既に名前の変更は実行されています。
- 4 **[名前の変更]** をクリックします。[リファクタリング] ウィンドウが開きます。[リファクタリング] ウィンドウには、[名前の変更] コマンドを実行した結果が表示されます。[名前の変更] コマンドによって、`Clerk` クラスの名前と、見つかった `Clerk` クラスの使用箇所も変更します。このサンプルでは、`Clerk` と `Store` クラス間に関連が存在するため、1 つの使用箇所だけが表示されます。関連は `Store` クラスのソースコードで、`Field1` フィールドとして現われています。
- 5 ダイアログの左側にあるリファクタリングの実行ボタン  をクリックして名前の変更を実行します。
- 6 [リファクタリング] ウィンドウを閉じます。ウィンドウの右上隅にある閉じるボタン  をクリックして下さい。

[リファクタリング] ウィンドウでの作業

[リファクタリング] ウィンドウはドッキング可能なウィンドウです。最初、自由に移動可能なフローティング ウィンドウとして開かれます。しかし、フローティング形式のままでも、ドッキング領域へドッキングしても構いません。ウィンドウのドッキング領域は、Visual Studio .NET ウィンドウの 4 辺どれを使用しても構成することができます。好きな場所に [リファクタリング] ウィンドウを配置して下さい。

同じプロジェクトに対して [リファクタリング] コマンドを利用する毎に、結果はタブ付きページ形式で表示され、最新の結果にウィンドウのフォーカス移ります。対応するタブを右クリックして、次のコンテキスト メニューから結果を閉じることができます。

- ・ **[閉じる]**：フォーカスを持った現在のタブを閉じます。
- ・ **[すべて閉じる]**：すべてのタブと [リファクタリング] ウィンドウを閉じます。
- ・ **[このタブ以外すべて閉じる]**：フォーカスを持った現在のタブを除くすべてのタブを閉じます。

ウィンドウの右上隅にある閉じるボタン  をクリックすると、すべてのタブを開いたまま [リファクタリング] ウィンドウが閉じます。

ドキュメント生成

Together VS .NET は、UML ドキュメントウィザード機能を提供しており、ユーザーはこれを使用して VideoStore プロジェクトのための HTML ドキュメントを自動生成することができます。

プロジェクトの HTML ドキュメントを生成するには：

- 1 メイン メニューから、**[ツール | ドキュメントの生成 ...]** を選択します。[ドキュメントの生成] ダイアログが開きます（図 27 参照）。
- 2 好みの **[対象範囲]** と **[オプション]** 設定を選択し、**[OK]** をクリックして下さい。ドキュメントが生成されます。
- 3 Video Store プロジェクトにディレクトリ out/doc を作成するように促されます。**[はい]** をクリックします。

デフォルトでは、ドキュメント生成ウィザードは、プロジェクト全体のドキュメントを生成します。異なる **[対象範囲]** オプションを選択することにより、より小規模のドキュメントを生成するよう、ドキュメントの出力範囲を限定することができます。

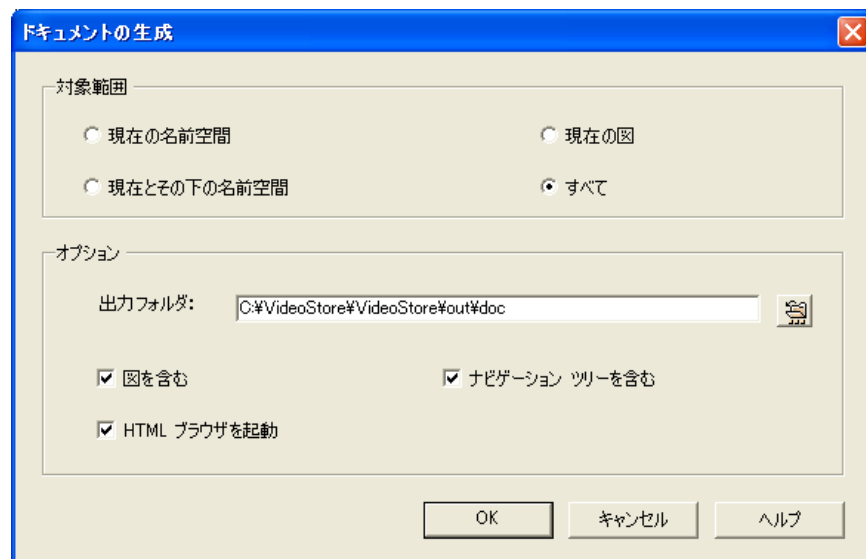
このダイアログの最上部にある **[対象範囲]** セクションには、解析し、生成されるドキュメントに含めるプロジェクトの内容を指定するラジオ ボタンがあります。

- **[現在の名前空間]**：生成される出力には、モデル ビューで選択した現在の名前空間のみが含まれます。
- **[現在とその下の名前空間]**：生成される出力には、モデル ビューで選択した現在の名前空間と、その下にある名前空間が含まれます。
- **[現在の図]**：生成される出力には、ダイアグラム ビューでフォーカスされている現在の図が含まれます。
- **[すべて]**：生成される出力には、プロジェクト全体が含まれます。

ダイアログの **[オプション]** セクションには、ドキュメントの出力先と、他の任意の動作を指定するためのオプションがあります。

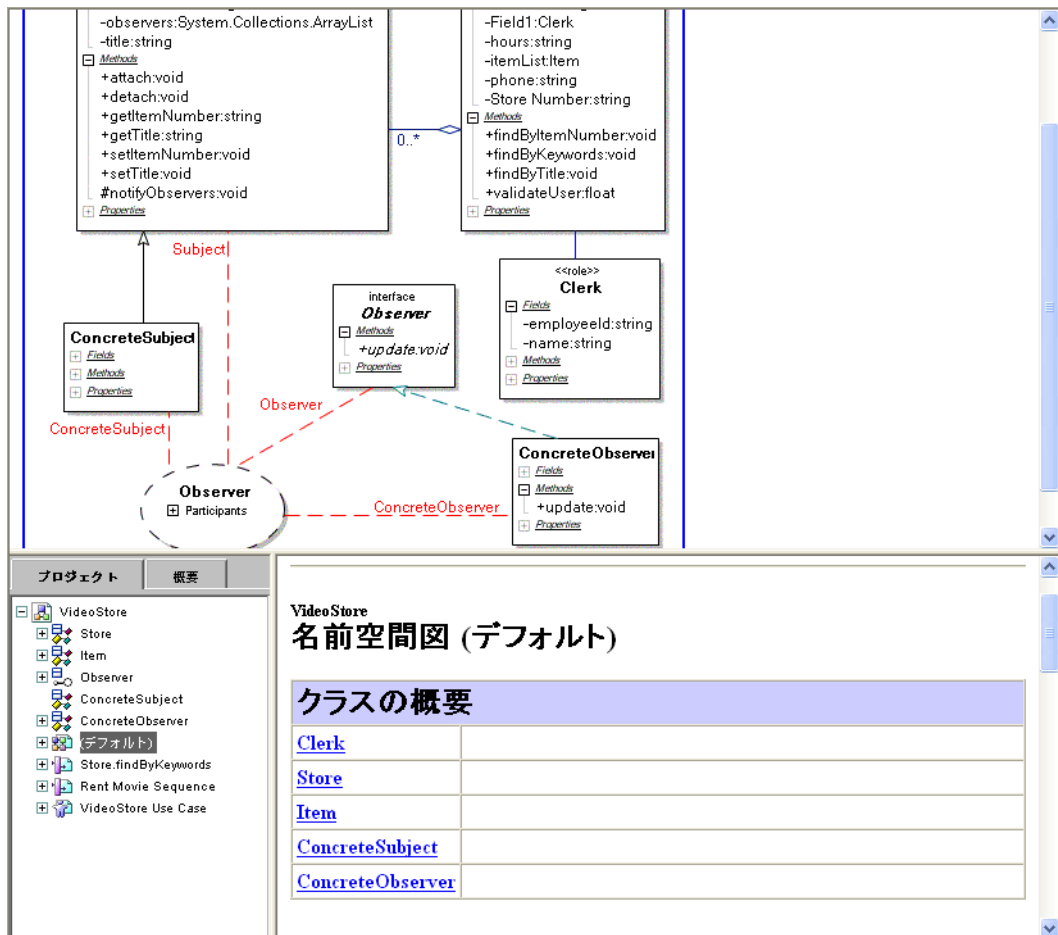
- **[出力フォルダ]**：生成される出力の格納場所を入力するか、ファイル セクタ ボタンで場所を選択します。
- チェックボックス：
 - **[図を含む]**：出力に図のイメージを含める場合にチェックします。
 - **[ナビゲーション ツリーを含む]**：出力にナビゲーション ツリーを含める場合にチェックします。
 - **[HTML ブラウザを起動]**：オペレーティング システムのデフォルト Web ブラウザでドキュメントを開く場合にチェックします。必要に応じてアプリケーションが起動されます。このオプションを選択しなくても、設定した出力フォルダへ行き、index.html ファイルを直接開くことにより、そのドキュメントを開くことができます。

図 27 [ドキュメントの生成] ダイアログ



Together VS .NET がドキュメントの生成を完了すると、システムのデフォルト Web ブラウザでドキュメントが開かれます。ブラウザは生成されたドキュメントを表示するため、フレームセットを開きます。左下フレームにあるツリーで、**[VideoStore]** ノードを展開して下さい。ブラウザには以下の図のようなページが表示されます (図 28 参照)。左下フレーム内のクラス名をクリックすると、右下ペインに該当するドキュメントが開かれることを確認して下さい。

図 28 出力されたドキュメント



生成されたドキュメントの機能をさらに確認するために、以下のステップを行ってみて下さい。

- ・ **[プロジェクト]** タブで、プロジェクト内をナビゲートします。
- ・ **Store** クラスをクリックして、関連するドキュメントを表示します。
- ・ **findByKeywords** をクリックして、ドキュメント内のそのセクションにジャンプします。
- ・ ハイパーテキストを使用して、ドキュメントをたどり、インデックスにもアクセスしてみます。

XMI プロジェクトのインポート / エクスポート

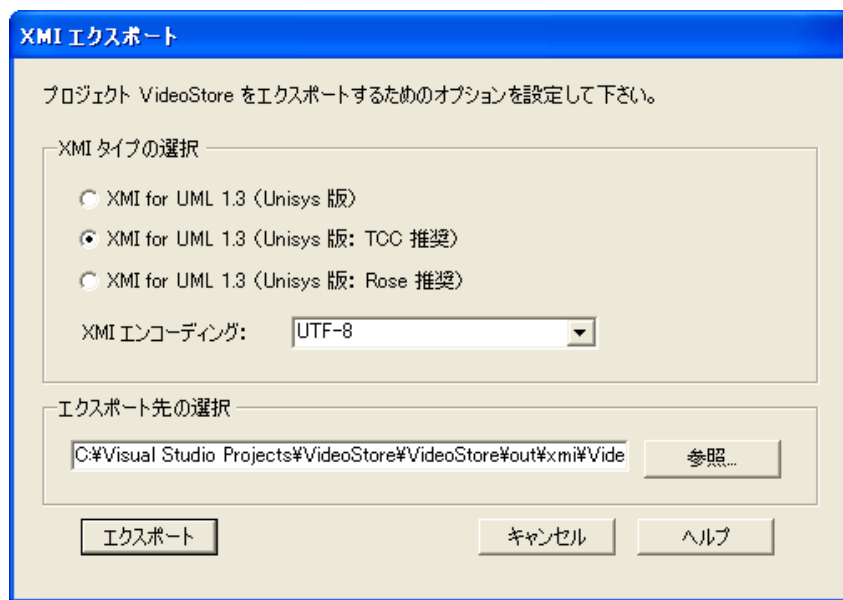
Together VS .NET は、XMI のインポート / エクスポート機能も提供しています。XMI ファイルによって記述されたモデルを、ソースコードを生成する Together VS .NET プロジェクトにインポートすることが可能です。また逆に、Together VS .NET プロジェクトを XMI ファイルにエクスポートし、そのモデル情報を Together VS .NET の外部で使用できるようにすることも可能です。

XMI プロジェクトのエクスポート

プロジェクトを XMI ファイルとしてエクスポートするには：

- 1 モデルビューで Video Store プロジェクトを右クリックして、**[プロジェクトを XMI にエクスポート ...]**、または**[ファイル | プロジェクトを XMI にエクスポート ...]**（図を開くか、モデルビューでプロジェクト ルートのノードを選択）を選択します。**[XMI エクスポート]** ダイアログが現れます（図 29 参照）。

図 29 [XMI エクスポート] ダイアログ



- 2 その結果開かれたダイアログ ボックスで、XMI タイプ情報と、XMI ファイルのエクスポート出力先を指定することができます。この例では、デフォルト値をそのまま使用します。**[エクスポート]** をクリックします。
- 3 新しいディレクトリを作成していいか、確認ダイアログが表示されます。**[はい]** をクリックします。
- 4 使用しているシステムのディレクトリ階層で、新しく生成された XML ファイルを <project_directory>\out\xmi\Video Store.xml を探します。

XMI プロジェクトのインポート

ここでは、既存の XMI ファイルのインポートは行いませんが、そのやり方はエクスポートの方法とほぼ同様です。

XMI ファイルをプロジェクトへインポートするには：

- 1 モデルビューで Video Store プロジェクトを右クリックして、**[プロジェクトを XMI からインポート ...]**、または **[ファイル | プロジェクトを XMI からインポート ...]**（図を開くか、モデルビューでプロジェクト ルートのノードを選択）を選択します。[XMI インポート] ダイアログが現れます（図 30 参照）。
- 2 **[参照 ...]** ボタンを使用して、ソース ファイルを指定します。
- 3 **[インポート]** をクリックします。

図 30 [XMI インポート] ダイアログ

