DevPartnerStudio クイック リファレンス

このクイック リファレンスは、全ページまたは一部を印刷（できればカラー印刷）して、適宜参照できるように、手近なところに備えておいてください。

DevPartnerの機能

DevPartnerの機能に関する参照情報には、以下の表の右欄からリンクされています。

用途	DevPartner 機能
プログラミング上の問題とネーミングの不一致を検出する	コード レビュー
ソース コードのランタイム エラーを診断する	エラー検出
アプリケーション内のパフォーマンス ボトルネックを特定する	カバレッジ分析、メモリ分析、パフォーマンス分析
開発とテストのフェーズを通してコード ベース安定性を確保する	カバレッジ分析セッション データ
アプリケーション内のメモリ割り当てを調べ、その情報によってメモリの消費量を減らす	メモリ分析

詳細情報

詳細については、DevPartnerオンライン ヘルプまたは『DevPartnerユーザー ガイド』を参照してください。

共通要素

DevPartnerのすべての機能で、以下の要素が提供されています。

- DevPartnerツールバー
- DevPartnerメニュー
- DevPartnerファイル拡張子
- コマンド ライン インストゥルメンテーション オプション

DevPartnerメニューおよびツールバー

DevPartnerのメニューまたはVisual Studioのツールバーからアクセスします。

メニュー項目またはツールバー ボタン	機能
 エラー検出	BoundsCheckerテクノロジを使用した、ランタイム エラーの検出
 カバレッジ分析	ランタイム コード カバレッジの分析
 エラー検出とカバレッジ分析	ランタイム エラーの検出とコード カバレッジの分析
 パフォーマンス分析	ランタイム パフォーマンスの分析
 メモリ分析	ランタイム メモリの分析
 パフォーマンス エキスパート	パフォーマンス エキスパートを使用したランタイムの分析
 コード レビューの実行	静的なコード分析
 コード レビューのルールの管理	コード レビュー ルール管理へのアクセス
エラー検出ルール	検出されたエラーのフィルタまたは抑制に使用されるエラー検出ルール管理へのアクセス
 ネイティブC/C++インストゥルメンテーション	エラー検出、エラー検出とカバレッジ分析、パフォーマンス分析、またはカバレッジ分析のコンパイル時インストゥルメンテーション
ネイティブC/C++インストゥルメンテーション マネージャ	インストゥルメンテーション マネージャへのアクセス
関連付け	パフォーマンス ファイルまたはカバレッジ ファイルの関連付け
カバレッジ ファイルのマージ	カバレッジ分析セッションのマージ
 オプション	DevPartnerオプションへのアクセスオプションの内容：分析、コード レビュー、エラー検出

DevPartnerファイル拡張子

セッション ファイルのファイル拡張子です。

DevPartner 機能	作成されるセッション ファイル（拡張子）
コード レビュー	.dpmdb
コード カバレッジ	.dpcov
コード カバレッジ マージ ファイル	.dpmrg
エラー検出	.dpbcl
メモリ分析	.dpmem
パフォーマンス分析	.dpprf
パフォーマンス エキスパート	.dppxp

コマンド ライン インストゥルメンテーション オプション

NMCLオプション

以下の表に、コマンド ラインからアンマネージ（ネイティブ）C++コードをインストゥルメントするために使用できるNMCLオプションを示します。NMCL.EXEは、DevPartnerのパフォーマンス／カバレッジ分析、またはエラー検出がインストゥルメントされているアンマネージC++コードのコンパイルだけに使用してください。マネージ コードではNMCLは使用されず、実行時に共通言語ランタイムに渡されるときにDevPartnerによってインストゥルメントされます。

NMCLオプションはすべて、以下の表に示されているように、スラッシュ（/）またはハイフン（-）に続くNMで始めてください。例：/NMoptionまたは-NMoption。

オプション	機能
/NMbcpath:bc-path	パス上にNMCLを含むディレクトリがない場合、bcinterf.libのディレクトリ場所を指定します。
/NMclpath:cl-path	CL.EXEのディレクトリの場所を指定します。DEVENVがインストールされている場所を無視する場合や、DEVENVがインストールされていない場合に、このオプションを使用できます。
/NMhelpまたは/?	ヘルプ テキストを表示します。
/NMignore:source-fileまたは /NMignore:source-file:method source-file	インストゥルメントしないソース ファイルまたはソース ファイル内のメソッドを指定します。
/NMlog:log-file	NMCLメッセージのログ ファイルを指定します（デフォルト：stdout）。
/NMnogm	CL /Gm（最小リビルド）オプションが指定されている場合、これを無視します。このオプションは、すでに判明しているNMAKE /AとCL /Gmオプション間の競合を避けるために使用できます。
/NMonly:source-file	インストゥルメントするソース ファイルを1つだけ指定します。
/NMopt:option-fileまたは /NM@option-file	オプション ファイル（各コマンドライン オプションが別々の行に書かれたASCIIファイル）を指定します。
/NMpass	パススルー モードを指定します。パススルー モードでは、NMCLがユーザーの介入なしにCLを呼び出します。この場合、インストゥルメンテーションは行われません。
/NMstoponerror	インストゥルメンテーション中にエラーが発生した場合、NMCLを中止します。このオプションを指定しないと、デフォルトで標準CLコンパイルにフォールバックします。
/NMbcOn	DevPartnerのエラー検出インストゥルメンテーションを使用します。これはデフォルトの設定です。
/NMtxOn	パフォーマンス分析とカバレッジ分析のインストゥルメンテーションを指定します。
/NMtxInlines	/O1、/O2、/Ob1、または/Ob2オプションでインライン最適化が有効になっていると、インライン メソッドがインストゥルメントされます。
/NMtxNoLines	DevPartnerに、ソース コードの行の情報を収集しないように命令します。このオプションを使用すると、[ソース]タブのコード行のデータは表示されません。また、アプリケーションのインストゥルメンテーションと実行にかかる時間を短縮することもできます。
/NMtxpath:tx-path	パスにNMCLを含むディレクトリがない場合、パフォーマンス分析とカバレッジ分析のライブラリ ファイルのディレクトリ場所を指定します。

NMCLを使用する場合、これらのユーティリティを含むディレクトリをパスに追加します。たとえば、製品をデフォルト ディレクトリにインストールした場合、以下のディレクトリをパスに追加します。

C:¥Program Files¥Common Files¥Micro Focus¥NMShared

メモ：64ビット バージョンのWindowsにインストールする場合は、以下のパスに追加します。

C:¥Program Files (x86)¥Common Files¥Micro Focus¥NMShared

NMLINK オプション

以下の表に、コマンド ラインからアンマネージ（ネイティブ コード）C++ アプリケーションを DevPartner にリンクするために使用できる NMLINK オプションを示します。

メモ：NMLINK オプションはすべて、以下の表に示されているように、スラッシュ（/）またはハイフン（-）に続く NM で始めてください。以下に例を示します。/NMoption または -NMoption。

オプション	機能
/NMbcOn	DevPartner のエラー検出インストゥルメンテーションを使用します。これはデフォルトの設定です。
/NMbcpath:bc-path	パス上に NMCL を含むディレクトリがない場合、bcinterf.lib のディレクトリ場所を指定します。
/NMhelp または /?	ヘルプ テキストを表示します。
/NMlinkpath:link-path	LINK.EXE のディレクトリの場所を指定します。DEVENV がインストールされている場所を無視する場合や、DEVENV がインストールされていない場合に、このオプションを使用できます。
/NMpass	パススルー モードを指定します。パススルー モードでは、NMLINK がユーザーの介入なしに LINK を呼び出します。
/NMtxOn	カバレッジ分析とパフォーマンス分析のインストゥルメンテーションを指定します。
/NMtxpath:tx-path	パスに NMCL を含むディレクトリがない場合、パフォーマンス分析とカバレッジ分析のライブラリ ファイルのディレクトリ場所を指定します。

NMCL と NMLINK を使用する場合、これらのユーティリティを含むディレクトリをパスに追加します。たとえば、製品をデフォルト ディレクトリにインストールした場合、以下のディレクトリをパスに追加します。

C:¥Program Files¥Common Files¥Micro Focus¥NMShared

メモ：64 ビット バージョンの Windows にインストールする場合は、以下のパスに追加します。

C:¥Program Files (x86)¥Common Files¥Micro Focus¥NMShared

コード レビュー

ルール マネージャのコマンド ショートカット

以下のショートカット キーを使用して、ルール マネージャのコマンドを入力できます:

コマンド	動作
Ctrl+A	[ルール]>[すべてのルールを選択]
Ctrl+C	[ルール]>[選択したルールをコピー]
Ctrl+N	[ルール]>[新規ルール]
Ctrl+O	[ファイル]>[ルール セットを開く]
Ctrl+P	[ファイル]>[印刷]
Ctrl+V	[ルール]>[ルールの貼り付け]
F5	[表示]>[リフレッシュ]

CRBatch で使用されるコマンド ライン スイッチ

CRBatch.exe /<switch>

スイッチ	説明
/f configuration file/file name	ソリューションまたはプロジェクトをレビューする際に使用する構成ファイルを CRBatch に知らせます。 このスイッチは必須です。
/v または /verbose	エラーをメッセージ ボックスに表示し、バッチ プロシージャで使用する終了コードを設定するように、CRBatch に指示します。 このスイッチはオプションですが、構成ファイルを物理的にデバッグする際に使用すると便利です。
/vs "10.0"、/vs "9.0"、または /vs "8.0"	Visual Studio環境にバッチ レビューの場所を知らせます。10.0 (Visual Studio 2010)、9.0 (Visual Studio 2008) または 8.0 (Visual Studio 2005) を指定します。 このスイッチを使用することをお勧めします。システムに Visual Studio の複数のバージョンをインストールしているときに特に重要です。このスイッチを指定しないと、DevPartner はデフォルトで最新バージョンを使用します。

CRExport で使用されるコマンド ライン スイッチ

CRExport.exe /<switch>

スイッチ	説明
/?	**ヘルプ**−使用可能なコマンド ライン インターフェイスの引数のリストを表示します。
/f *sessionfile*	**完全修飾されたセッション ファイルのパスと名前**−このエクスポートに使用するセッション データベースを指定します。(必須)。
/e *xml exportfile*	**完全修飾されたエクスポート ファイルのパスと名前**−エクスポートされたデータを受け取る XML ファイルを指定します。(必須)。
/a	**すべてのセッション データのエクスポート**−コール グラフ データ用のアウトバウンド メソッドを含む、指定されたセッションのすべてのデータをエクスポートします。インバウンド メソッドはエクスポートされません。
/a i	**インバウンド メソッドを含むすべてのセッション データのエクスポート**−コール グラフ データ用のインバウンド メソッドとアウトバウンド メソッドを含む、指定されたセッションのすべてのデータをエクスポートします。
/p	**問題データのエクスポート**−指定されたセッションの問題データをエクスポートします。
/m	**メトリクス データのエクスポート**−指定されたセッションのメトリクス データをエクスポートします。
/n	**ネーミング分析データのエクスポート**−指定されたセッションのネーミング分析データをエクスポートします。
/s	**コード サイズ データのエクスポート**−指定されたセッションのコード サイズ データをエクスポートします。
/c	**コール グラフ データのエクスポート**−指定されたセッションのコール グラフ データに含まれるアウトバウンド、つまり、呼び出されたメソッドをエクスポートします。
/c i	**インバウンド メソッドを含むコール グラフ データのエクスポート**−指定されたセッションのインバウンド メソッドとアウトバウンド メソッドを含むコール グラフ データをエクスポートします。

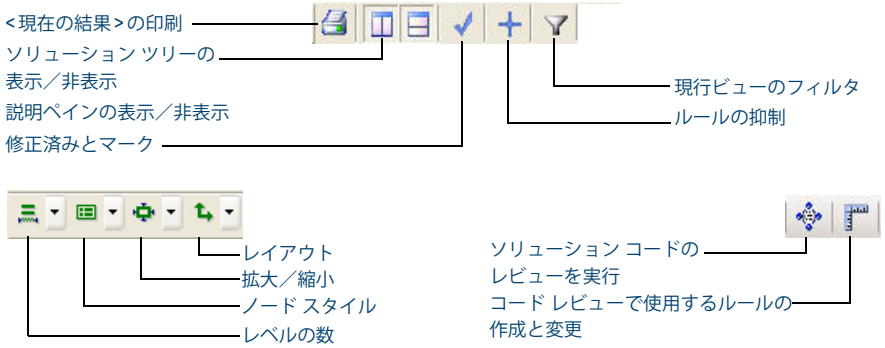
コード レビューのデフォルト オプション （全般ノード）

カテゴリ	設定
レビューするプロジェクト	選択されたすべてのプロジェクト（C# および VB..NET プロジェクトのみ）
ルール セット	すべてのルール
使用するネーミング分析	ネーミング ガイドライン（下記を参照）
メトリクスの収集	オン
コール グラフ データの収集	オン
常にバッチ ファイルを生成	オン
常にレビュー結果を保存	オン
セッション ファイル名入力のプロンプト	オフ

ネーミング ガイドライン

説明	デフォルト
分析の対象	すべてのパブリック識別子またはプロテクト識別子
ディクショナリの選択	アメリカ英語
ネーミング分析	選択されているすべての識別子
会社名	
テクノロジ名	

コード レビュー ツールバー



コード レビュー サマリ

問題サマリ*

タイプ	問題		重要度			
名前	合計	修正済み	高	中	低	警告
COM相互運用性	1	0	0	0	0	1
Windows API	0	0	0	0	0	0
エラー/例外処理	21	0	0	1	20	0
ガベージ コレクション	0	0	0	0	0	0
システム	0	0	0	0	0	0
セキュリティ	3	0	3	0	0	0
データベース	0	0	0	0	0	0
デザイン タイム プロパティ	0	0	0	0	0	0
バージョン管理	0	0	0	0	0	0
パフォーマンス	1	0	1	0	0	0
プロジェクトとソリューションのプロパティ	0	0	0	0	0	0
ユーザー定義のルール	0	0	0	0	0	0
ユーザビリティ	0	0	0	0	0	0
ロジック	2	0	0	0	0	2
保守性	13	0	0	2	3	8
信頼性	0	0	0	0	0	0
国際化	12	0	12	0	0	0
日付	0	0	0	0	0	0
標準	0	0	0	0	0	0
移植性	0	0	0	0	0	0
言語	0	0	0	0	0	0
合計	**53**	**0**	**16**	**3**	**23**	**11**

* サマリにはすべてのルール違反が含まれます。フィルタ設定が適用されません。

カウント サマリ

サマリ タイプ	カウント
レビュー時間(分)	0.733
合計行数(空白行を含む)	578
コードだけの行	386
コメントだけの行	90
コメント付きのコード	4
ルールとの比較の回数	127,716
チェックされた合計行数	533

レビュー設定

レビュー設定	設定値
ソリューション	SpeedBump.Net2003
ソリューション パス	C:¥SpeedBump.Net¥SpeedBump.Net2003.sln
セッション ファイル	C:¥SpeedBump.Net¥SpeedBump.Net2003.DPMDB
バッチ コマンド実行ファイル	C:¥SpeedBump.Net¥CR_SpeedBump.Net2003.BAT

プロジェクト リスト

プロジェクト名	コンパイル エラー	レビュー済み	プロジェクト パス
Driver2003	False	True	C:¥SpeedBump.Net¥Driver¥Driver2003.csproj
CSharp2003	False	True	C:¥SpeedBump.Net¥CSharp¥CSharp2003.csproj
VB2003	False	True	C:¥SpeedBump.Net¥VB¥VB2003.vbproj

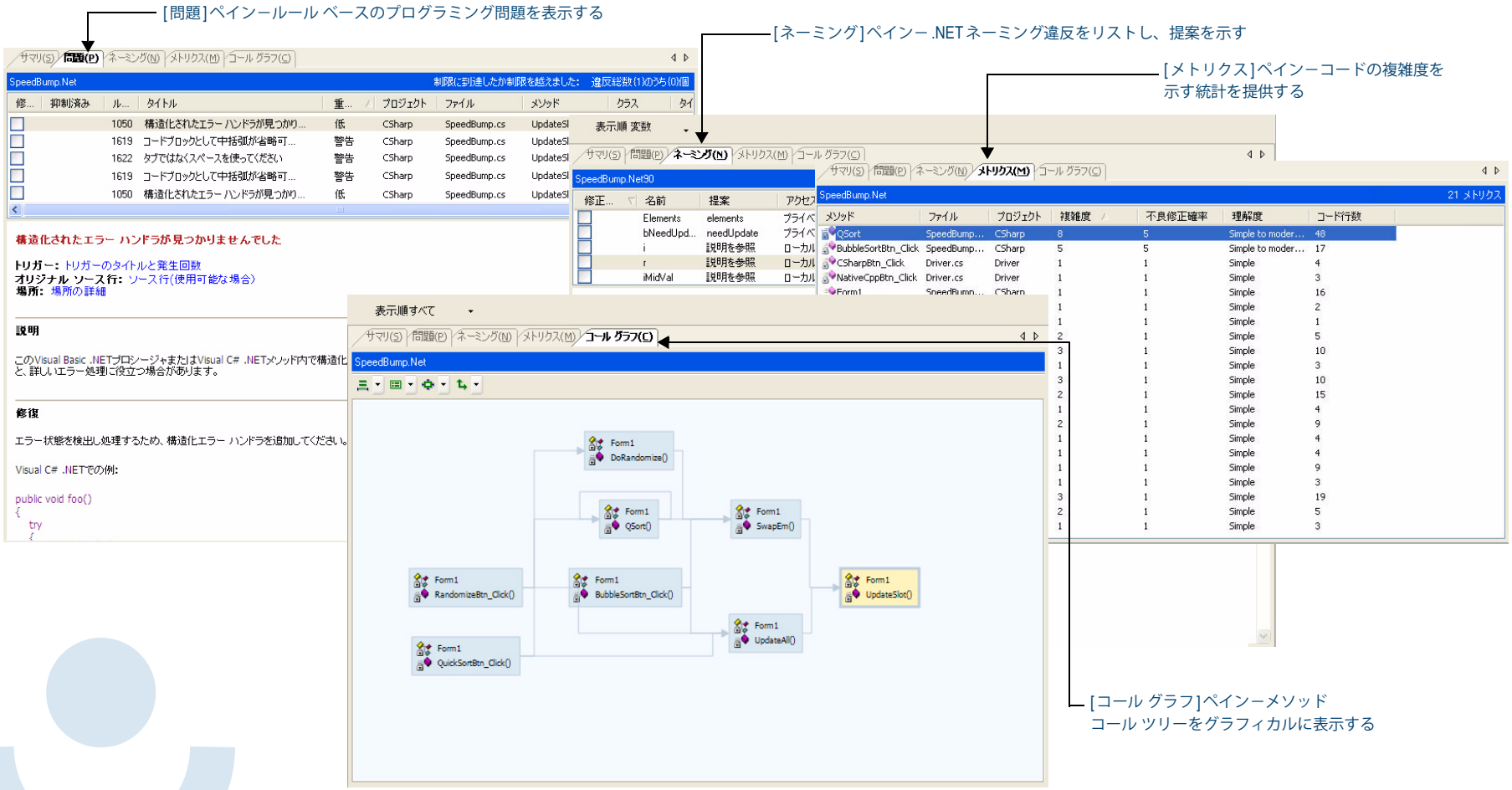
メトリクス分析	True
ネーミング分析	ネーミング ガイドライン
ディクショナリ名	アメリカ英語

コール グラフ データのサマリ

サマリ タイプ	カウント
グラフにした総メソッド数	41
未コールの総メソッド数	5

コール グラフ分析	True
コンパイル エラーの無視	False
ビルドを必要とするルールの除外	False
常にバッチ ファイルを生成	True

コード レビュー検証結果ペイン



カバレッジ分析、メモリ分析、パフォーマンス分析

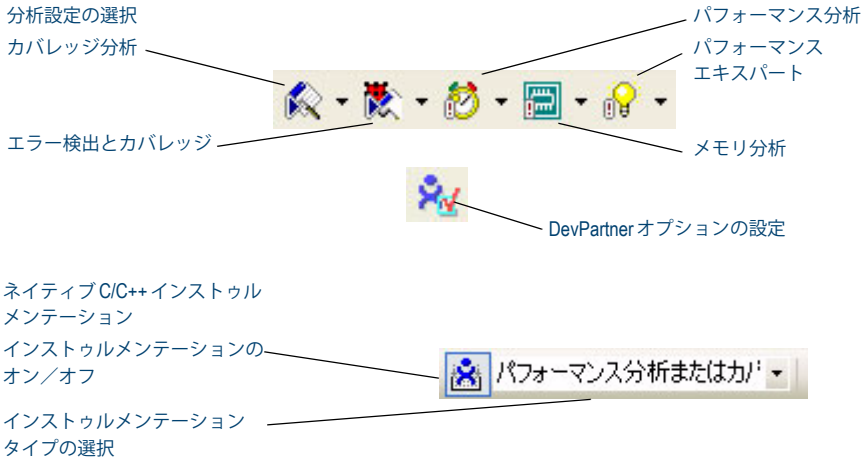
アプリケーションのテスト カバレッジの確認、アプリケーションのメモリ使用率の分析、アプリケーション パフォーマンスのプロファイルを行います。

全般およびデータ収集のプロパティ

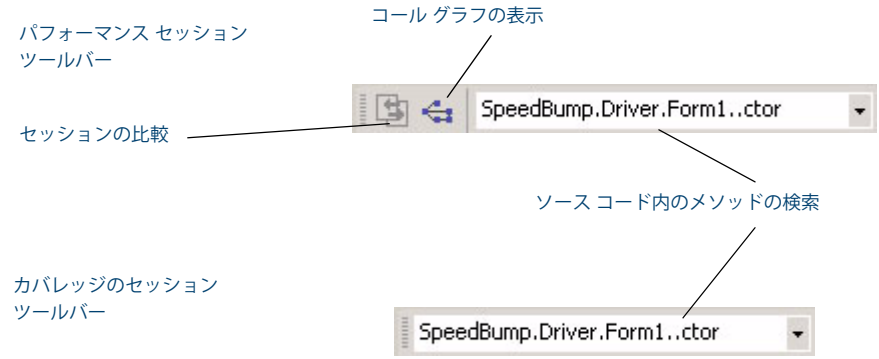
パフォーマンス分析、カバレッジ分析、メモリ分析では、以下のデータ収集プロパティを使用できます。

プロパティ	デフォルト設定
セッション ファイルを自動的にマージ	マージするかどうかを確認する
.NET アセンブリに関する情報を集める	True
COM 情報の収集	True
その他を除外	True
インライン関数をインストゥルメントする	True
インストゥルメンテーション レベル	行
システム オブジェクトの追跡	True

カバレッジ、メモリ、およびパフォーマンス用の DevPartner ツールバー ボタン



パフォーマンス分析とカバレッジ分析のセッション ツールバー



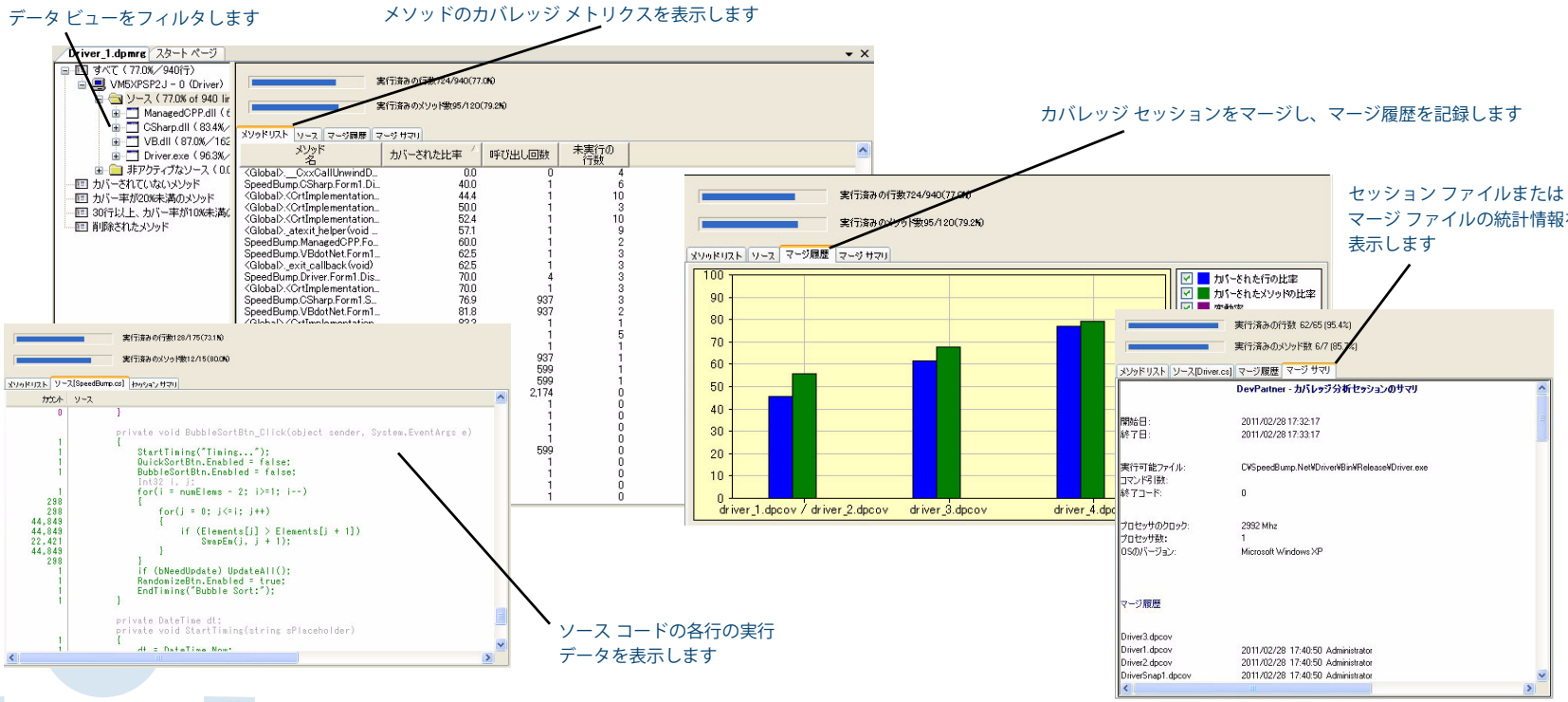
カバレッジ分析

カバレッジ分析セッション データ

結果のサマリ

DevPartnerはVisual Studioまたはカバレッジ分析ビューアにカバレッジ分析結果を表示します。セッション ファイルのデータは、以下のタブに表示されます。

- メソッド リスト
- ソース
- マージ履歴
- セッション サマリまたはマージ サマリ



メモリ分析

メモリ分析のセッション コントロール

メモリ リーク セッション コントロール

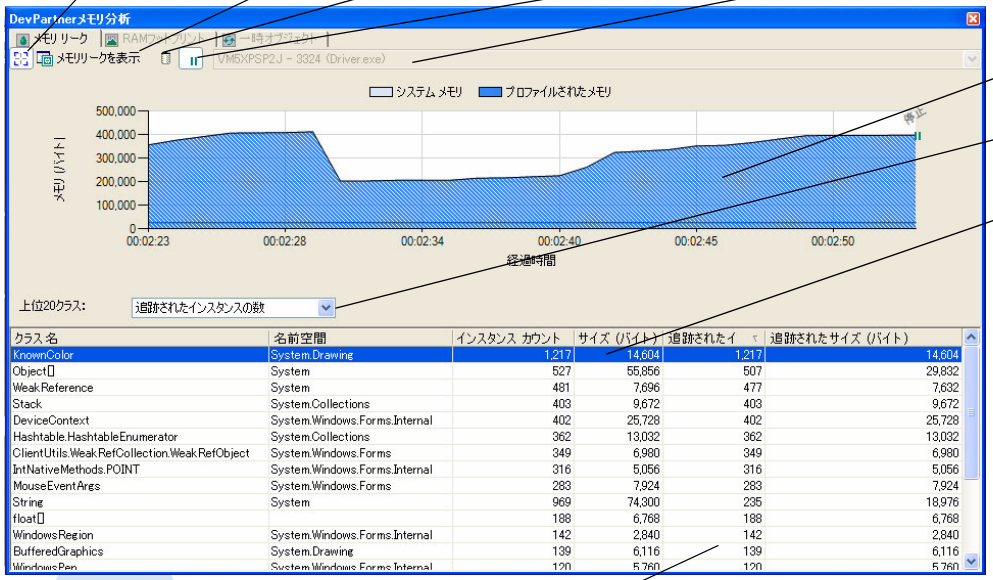
潜在的なメモリ リークの追跡を開始／中止します

メモリ リークのスナップショットを取ります

ガベージ コレクションを強制的に行います

リアルタイム グラフを一時停止します（データの収集は続行する）

プロファイルするプロセスを選択します



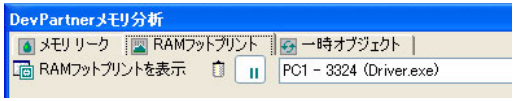
グラフはリアルタイムのマネージ ヒープの状態を示します

クラス リストのソート順を動的に変更します

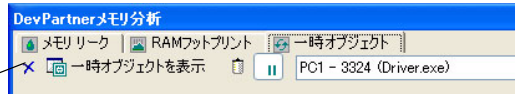
クラス リストは動的に更新されて、メモリの内容を示します

メモリ データ収集の種類に合わせてカスタマイズされたセッション コントロール

RAM フットプリント セッション コントロール：



一時オブジェクト セッション コントロール：



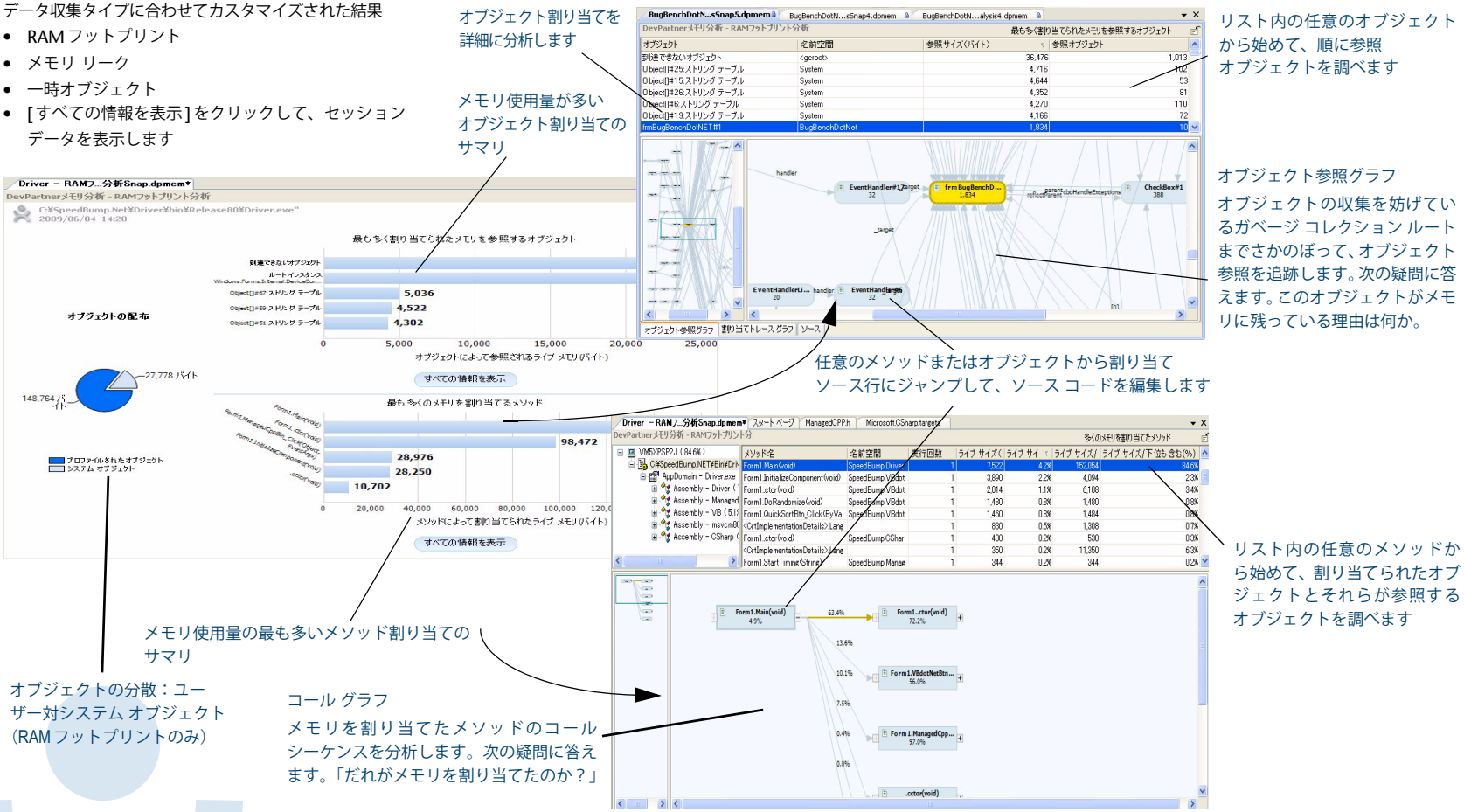
リーク分析で、期待どおりに収集されなかったオブジェクトの**追跡されたインスタンスの数**を監視します

この時点まで追跡された一時オブジェクトの割り当てをクリアします

メモリ分析セッション データ

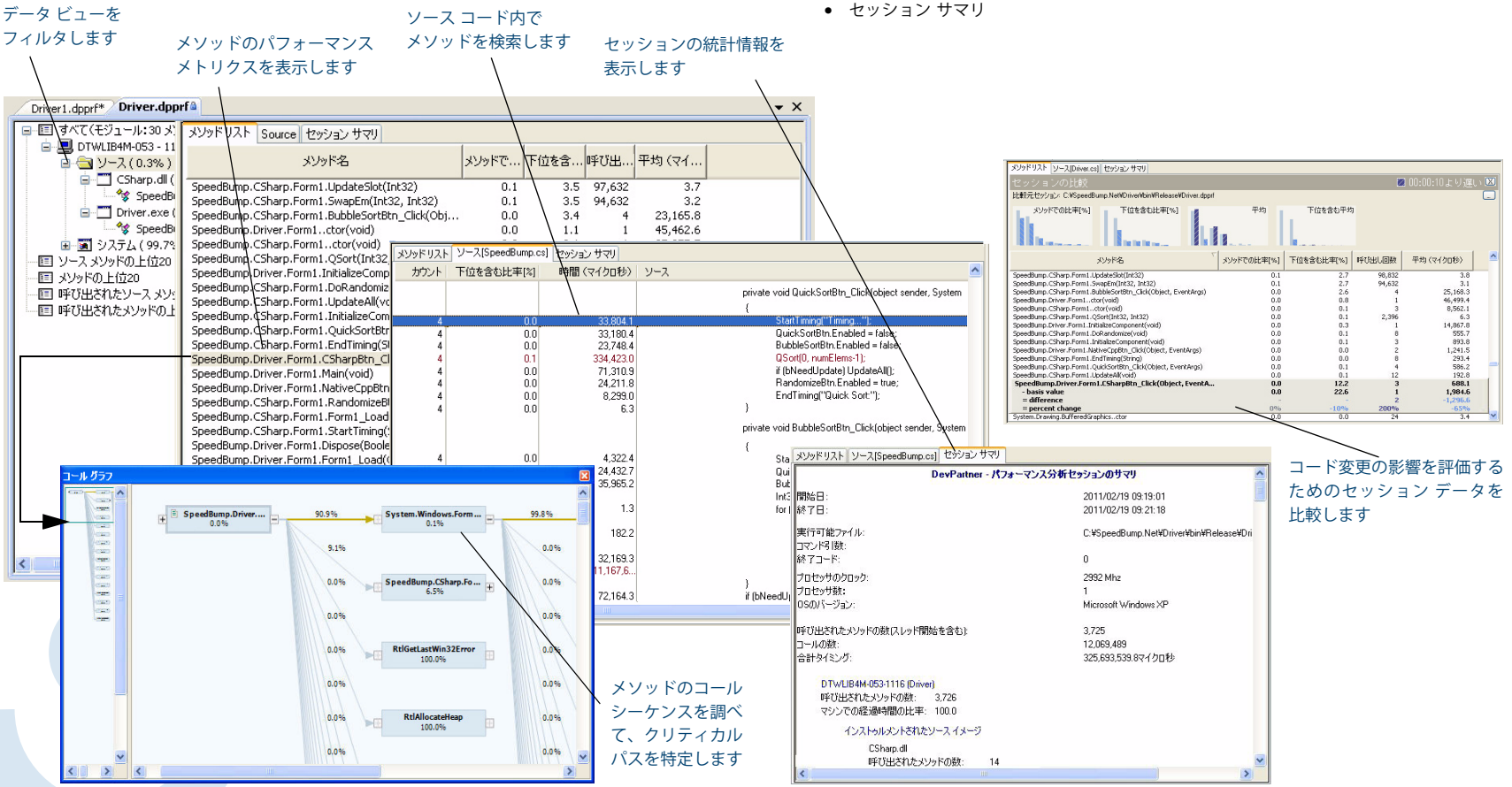
データ収集タイプに合わせてカスタマイズされた結果

- RAM フットプリント
- メモリ リーク
- 一時オブジェクト
- [すべての情報を表示] をクリックして、セッション データを表示します



パフォーマンス分析

パフォーマンス分析セッション データ



結果のサマリ

DevPartner は Visual Studio またはパフォーマンス分析ビューアにパフォーマンス分析結果を表示します。セッション ファイルのデータは、以下のタブに表示されます。

- メソッド リスト
- ソース
- セッション サマリ

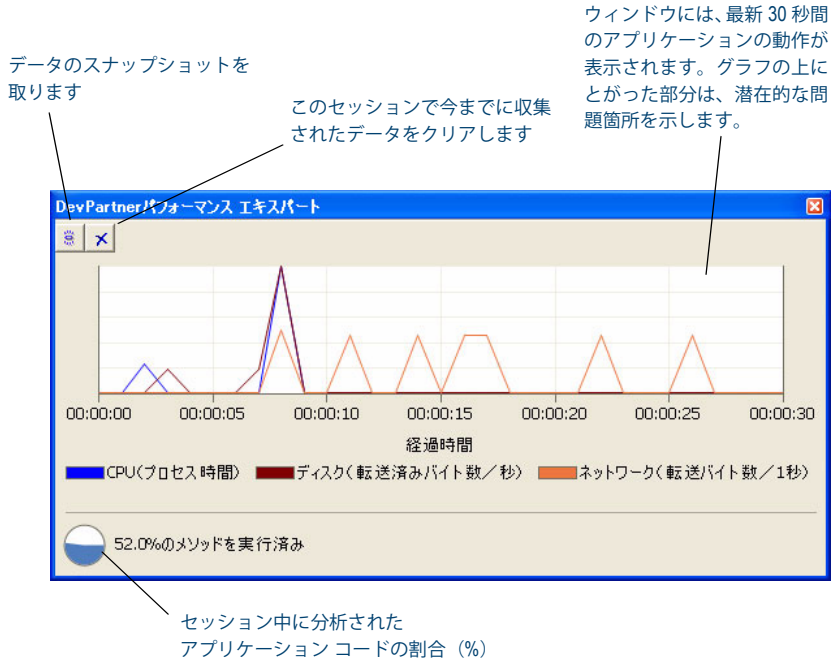
パフォーマンス エキスパート

結果のサマリ

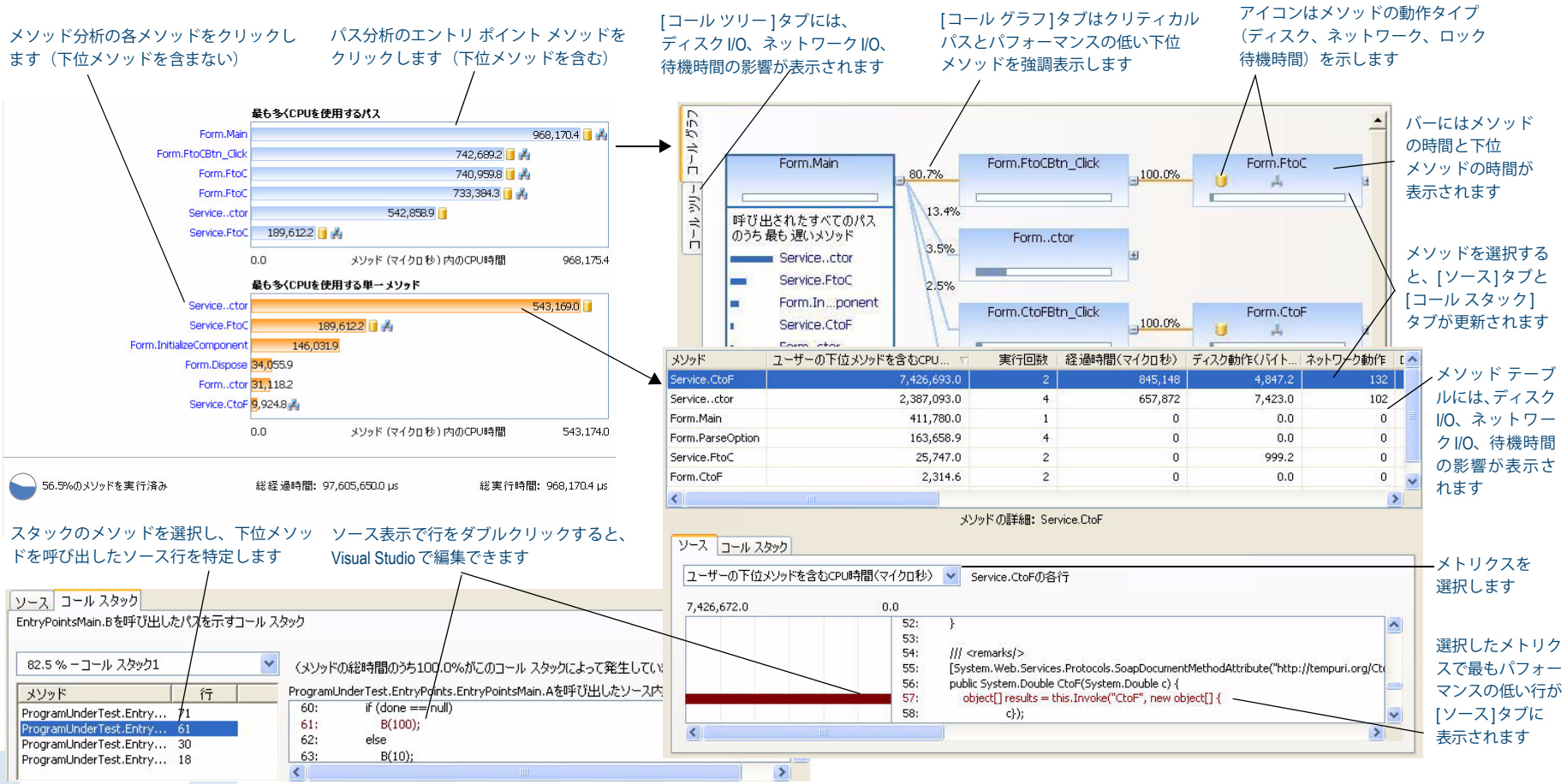
DevPartnerはセッション ファイルにパフォーマンス エキスパートの結果を表示します。セッション ファイルのデータは、以下のタブに表示されます。

- コール グラフ
- コール ツリー
- メソッド テーブル
- ソース
- コール スタック

パフォーマンス エキスパート セッション コントロール



パフォーマンス エキスパート セッション データ



DPAnalysis.exe の使用

DPAnalysis.exe を使用して、コマンド ラインからカバレッジ分析、メモリ分析、パフォーマンス分析、またはパフォーマンス エキスパートの各セッションを実行します。DPAnalysis.exe にはコマンド ライン スイッチまたは XML 構成ファイルを指定できます。

コマンド ライン操作

コマンド ラインからカバレッジ、メモリ、パフォーマンス、パフォーマンス エキスパートの各セッションを実行するには、以下の構文を使用します。

```
DPAnalysis.exe [a] {b} {c} {d} [e] target {target args}
```

DPAnalysis.exe では、分析とターゲットのタイプを指示するスイッチは必須です。その他のスイッチはオプションです。

以下の表に、DPAnalysis.exe で使用するスイッチをリストします。

カテゴリ	スイッチ
[a] **分析タイプ**	/Cov[erage] – DevPartner カバレッジ分析に分析のタイプを設定します
	/Mem[ory] – DevPartner メモリ分析に分析のタイプを設定します
	/Perf[ormance] – DevPartner パフォーマンス分析に分析のタイプを設定します
	/Exp[ert] – DevPartner パフォーマンス エキスパートに分析のタイプを設定します
{b} **データ収集**	/E[nable] –特定のプロセスまたはサービスのデータ収集を有効にします
	/D[isable] –特定のプロセスまたはサービスのデータ収集を無効にします
	/R[epeat] – /D スイッチを使用してプロファイリングを無効にしないかぎり、指定プロセスを実行するたびにプロファイリングが実行されます
{c} **その他のオプション**	/O[utput] –セッション ファイルの出力ディレクトリとファイル名のいずれかまたは両方を指定します
	/W[orkingDir] –プロセスまたはサービスの作業ディレクトリを指定します
	/H[ost] –ターゲットのホスト マシンを指定します
	/NOWAIT –プロセスの終了は待機せず、起動のみ待機します
	/NO_UI_MSG – UI エラー メッセージを抑制するにはこのスイッチを **true** に設定します。デフォルト値は **false** です。
	/N[ewconsole] –新しいコマンド ウィンドウでプロセスを実行します
	/F[orce] –マネージ コードまたは CTI を使用せずに記述したアプリケーションのカバレッジまたはパフォーマンスのプロファイリングを強制します
{d} **分析オプション**	/NO_QUANTUM –他のスレッドで費やされた時間の除外を無効にします
	/NM_METHOD_GRANULARITY –データ収集の精度をメソッド レベルに設定します（デフォルトは行レベル）
	/EXCLUDE_SYSTEM_DLLS –システム DLL に対するデータ収集を除外します（パフォーマンス分析のみ）
	/NM_ALLOW_INLINING –インライン メソッドの実行時インストゥルメンテーションを有効にします（カバレッジ分析とパフォーマンス分析のみ）
	/NO_OLEHOOKS – COM の収集を無効にします
	/NM_TRACK_SYSTEM_OBJECTS –追跡システムオブジェクトの割り当ての収集を無効にします（メモリ分析のみ）
[e] **ターゲットのタイプ**	プロセスまたはサービスとして、ターゲットを指定します。1 つだけ選択します。ターゲットの名前／パスのあとに指定するすべてのステートメントは、引数としてターゲットに渡されます。
	/P[ocess] –ターゲット プロセスを指定します（プロセスに渡される引数が続きます）
	/S[ervice] –ターゲット サービスを指定します（サービスに渡される引数が続きます）
	/C[onfig] –構成ファイルへのパスを指定します

構成ファイル

構成ファイルからカバレッジ、メモリ、パフォーマンス、パフォーマンス エキスパートの各分析セッションを実行するには、以下の構文を使用します。

DPAnalysis.exe /config c:¥temp¥config.xml

以下の表で、XML要素について簡単に説明します。詳細については、DevPartnerオンライン ヘルプまたは『DevPartnerユーザー ガイド』を参照してください。

要素	説明
AnalysisOptions	（オプション）プロセスまたはサービスごとに、0または1を指定します。特定のターゲット プロセスまたはターゲット サービスにランタイム属性を定義します。DevPartnerプロパティに対応する属性には、Visual Studioのプロパティ ウィンドウからアクセスできます。 **属性：**SESSION_DIR、SESSION_FILENAME、NM_METHOD_GRANULARITY、EXCLUDE_SYSTEM_DLLS、NM_ALLOW_INLINING、NO_OLEHOOKS、NM_TRACK_SYSTEM_OBJECTS、NO_QUANTUM
Arguments	（オプション）プロセスまたはサービスごとに、0または1を指定します。特定のターゲット プロセスまたはターゲット サービスにランタイム属性を定義します。DevPartnerのカバレッジ分析、メモリ分析、パフォーマンス分析の各プロパティに対応する属性には、Visual Studioのプロパティ ウィンドウからアクセスできます。 **属性：**SESSION_DIR、SESSION_FILENAME、NM_METHOD_GRANULARITY、EXCLUDE_SYSTEM_DLLS、NM_ALLOW_INLINING、NO_OLEHOOKS、NM_TRACK_SYSTEM_OBJECTS、NO_QUANTUM
ExcludeImages	（オプション）プロセスまたはサービスごとに、0または1を指定します。省略した場合のデフォルトはありません。ターゲット プロセスまたはターゲット サービスでロードされ、プロファイルされない場合に、イメージ（1つ以上、上限なし）を定義します。属性はありません。
Host	（オプション）プロセスまたはサービスごとに、0または1を指定します。省略した場合のデフォルトはありません。ターゲット プロセスまたはターゲット サービスのホスト マシンを設定します。属性はありません。
Name	サービスごとに1つ指定します。サービス コントロール マネージャに登録されているサービスの名前を指定します。これは、システムのNET STARTコマンドを使用するときと同じ名前です。属性はありません。
Path	プロセスごとに1つ指定します。実行可能ファイルの完全修飾パスまたは相対パスを指定します。実行可能ファイルが現在のディレクトリにある場合は、パスを指定せずに実行可能ファイル名を指定できます。属性はありません。
Process	構成ファイルには、少なくとも1つのProcess要素またはService要素を指定する必要があります。ターゲットの実行可能ファイルを指定します。 属性：CollectData、Spawn、NoWaitForCompletion、NewConsole
RuntimeAnalysis	必須の要素です。1つだけ指定します。分析のタイプと最長のセッション時間を定義します。
Service	構成ファイルには、少なくとも1つのProcess要素またはService要素を指定する必要があります。ターゲット サービスを指定します。 属性：CollectData、Start、RestartIfRunning、RestartAtEndOfRun
Targets	必須の要素です。1つだけ指定します。1つ以上のProcessエントリまたはServiceエントリのブロックを開始します。ターゲットのプロセスとサービスは、構成ファイルに指定されている順に開始されます。 属性：RunInParallel

エラー検出

エラー検出で使用されるファイル拡張子

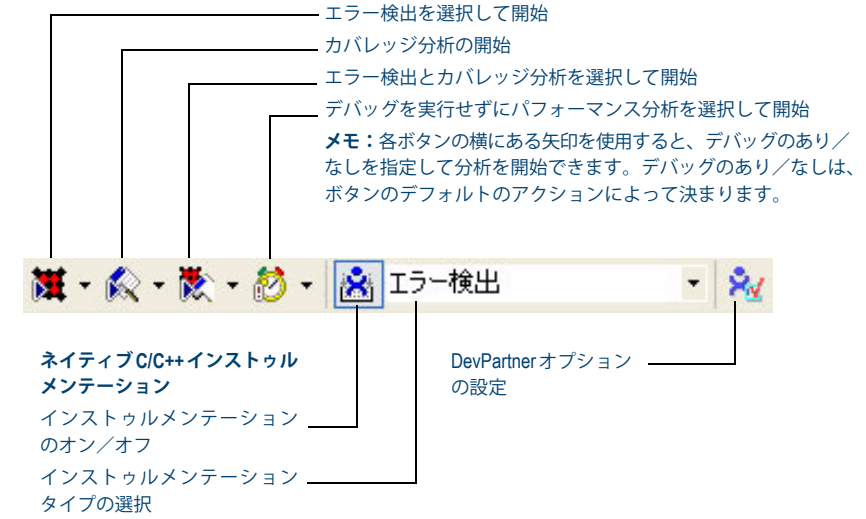
拡張子	ファイルの種類	説明
.dpbcl	エラー検出セッション ファイル	ユーザーのプログラム実行に関するエラー検出ログです。
.dpbcc .dpbcd	エラー検出設定ファイル	エラー検出に関するさまざまな設定を格納するファイルです。.dpbcd拡張子のファイルは、作成されたデフォルト設定ファイルを参照します。.dpbcc拡張子のファイルは、別に保存されているカスタム設定ファイルを参照します。
.dpsup	エラー検出抑制ファイル	ユーザーのプログラムに関するさまざまな抑制情報を格納するファイルです。
.dpflt	エラー検出フィルタ ファイル	ユーザーのプログラムに関するさまざまなフィルタ情報を格納するファイルです。
.dprul	エラー検出ルール ファイル	ユーザーの抑制とフィルタに関するデータベースです。

デフォルトのオプション（DevPartner Studio Professional Edition および Enterprise Edition）または設定（Visual C++ BoundsChecker Suite）

カテゴリ		設定
全般	オン	イベントをログに記録
	オン	エラーを表示して一時停止
	オフ	プログラム検証結果の保存を確認する
	オフ	アプリケーションを終了したときに、メモリおよびリソース ビューアを表示する
	オン	ソース ファイルの検索パス－.EXE（スタンドアロン）、.DSW（C++）、または .SLN（Visual Studio）の場所に応じる
	-	シンボル パスの上書き－**デフォルト：空白**
	-	作業ディレクトリ（スタンドアロンのみ）－.EXEの場所に応じる
	-	コマンド ライン引数（スタンドアロンのみ）－**デフォルト：空白**
データ収集	オン	コール パラメータのデータ表示の深さ = 1
	オン	メモリ割り当ての最大コール スタック数 = 5
	オン	エラーの最大コール スタックの深さ = 20
	オン	NLBファイル ディレクトリー.EXE（スタンドアロン）、.DSW（C++）、または .SLN（Visual Studio）の場所に応じる
APIコール レポーティング	オフ	APIコール レポーティングを有効にする。**この項目を選択しないと、その他の項目は選択できません。**
	-	ウィンドウ メッセージを収集する－**アクティブなときのデフォルト：オフ**
	-	APIメソッドのコールとリターンを収集－**-アクティブなときのデフォルト：オン**
	-	このアプリケーションに必要なモジュールだけを表示－**アクティブなときのデフォルト：オン**
	-	すべてのモジュール（ツリー ビュー）－**-アクティブなときのデフォルト：選択したものすべて**
コール バリデーション	オフ	コール バリデーションを有効にする。**この項目を選択しないと、その他の項目は選択できません。**
	-	メモリ ブロック チェックを有効にする－**アクティブなときのデフォルト：オフ**
	-	コール前に出力情報を入力する－**アクティブなときのデフォルト：オフ**
	-	COM失敗コード－**アクティブなときのデフォルト：オン**
	-	COMの"実装されていません"リターン コード**をチェックする－アクティブなときのデフォルト：オン**
	-	API失敗コード－**アクティブなときのデフォルト：オン**
	-	無効なパラメータ エラーのチェック：API、COM－**アクティブなときのデフォルト：どちらもオン**
	-	カテゴリ：ハンドルとポインタの引数－**アクティブなときのデフォルト：オン**
	-	カテゴリ：フラグ、範囲、および列挙の引数－**アクティブなときのデフォルト：オン**
	-	Cランタイムの静的ライブラリAPIをチェックする－**アクティブなときのデフォルト：オン**
		APIエラーをチェックするDLL（失敗または無効な引数）－**アクティブなときのデフォルト：選択したものすべて**
COMコール レポーティング	オフ	選択したモジュールに実装されたオブジェクト上でのCOMメソッド コールのレポートを有効にする
	-	リストされたモジュール外で実装されたオブジェクトのCOMメソッド コー**ルをレポートする－アクティブなときのデフォルト：オン**
	-	すべてのコンポーネント ツリー ビュー－**アクティブなときのデフォルト：選択したものすべて**
COMオブジェクトの追跡	オフ	COMオブジェクトの追跡を有効にする
	-	すべてのCOMクラス（ツリー ビュー）－**アクティブなときのデフォルト：選択したものすべて**
デッドロック分析	オフ	デッドロック分析を有効にする
	-	シングル プロセスと仮定する－**アクティブなときのデフォルト：オン**
	-	ウォッチャー スレッドを有効にする－**アクティブなときのデフォルト：オフ**
	-	エラーを生成するとき：クリティカル セクションが再入力されたとき－**アクティブなときのデフォルト：オフ**

カテゴリ		設定
	-	エラーを生成するとき：所有するミューテックスに待機が要求されたとき－**アクティブなときのデフォルト：オフ**
	-	リソースごとの過去のイベント数－**アクティブなときのデフォルト：10**
	-	同期APIタイムアウトをレポート－**アクティブなときのデフォルト：オフ**
	-	待機制限または実際の超過時間（秒）をレポート－**アクティブなときのデフォルト：60**
	-	同期ネーミング ルール－**アクティブなときのデフォルト：リソース ネーミングについて警告しない**
メモリの追跡	オン	メモリの追跡を有効にする
	オフ	リーク分析のみを有効にする
	オフ	リークしたアロケータブロックを表示する
	オン	厳密な再割り当てセマンティクスを適用する
	オン	FinalCheckを有効にする
	オン	保護バイトを有効にする；パターン = FC; カウント = 4バイト
	-	実行時のヒープ ブロックをチェックする：解放時
	オン	確保時にフィルする；パターン = FB
	オン	初期化されていないメモリをチェックする；サイズ = 2バイト
	オン	解放時に無効データでフィルする；パターン = FD
.NET分析	オフ	.NET分析を有効にする
	-	例外の監視－**アクティブなときのデフォルト：オン**
	-	ファイナライザの監視－**アクティブなときのデフォルト：オン**
	-	COM相互運用性の監視－**アクティブなときのデフォルト：オン**
	-	PInvoke相互運用性の監視－**アクティブなときのデフォルト：オン**
	-	相互運用性レポートのしきい値－**アクティブなときのデフォルト：1**
.NETコール レポーティング	オフ	.NETメソッドコール レポーティングを有効にする
	-	すべてのタイプ（ツリー ビュー）－**アクティブなときのデフォルト：選択されている**
	-	.NETユーザー アセンブリ（ツリー ビュー ノード）－**アクティブなときのデフォルト：選択されている**
	-	.NETシステム アセンブリ（ツリー ビュー ノード）－**アクティブなときのデフォルト：選択されていない**
リソースの追跡	オン	リソースの追跡を有効にする
	オン	リソース（ツリー ビュー）リストにあるすべてのリソースがデフォルトで選択される

Visual Studioのエラー検出ツールバー



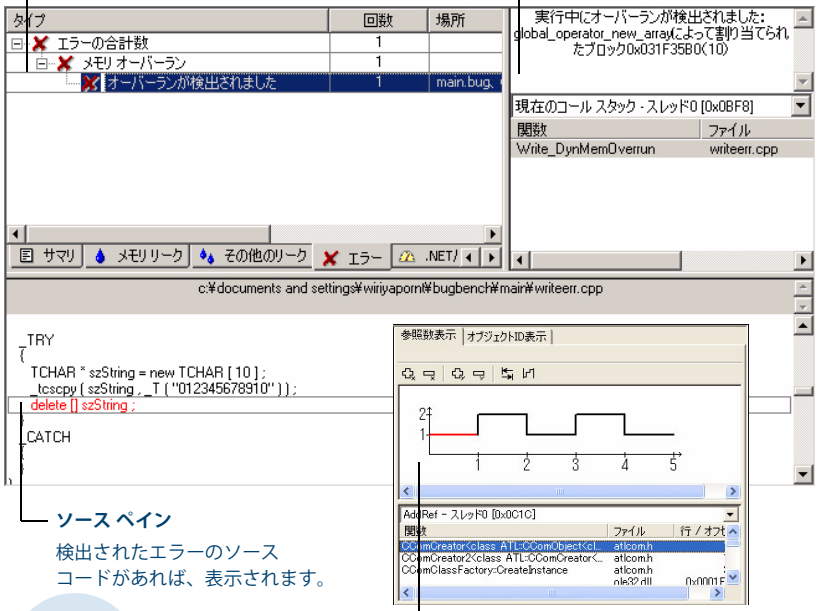
エラー検出ウィンドウ

検証結果ペイン

[サマリ]、[メモリ リーク]、[その他のリーク]、[エラー]、[.NETパフォーマンス]、[モジュール]、[通知情報]の各タブに、検出されたエラーに関する情報が表示されます。

詳細ペイン

検出されたエラー、コール スタック、参照回数グラフなどについて、詳細な説明が表示されます（下の別図を参照）。



ソース ペイン

検出されたエラーのソース コードがあれば、表示されます。

詳細ペイン－参照回数グラフ

検証結果ペインで[インターフェイス リーク]を選択すると、[参照カウントビュー]タブと[オブジェクト IDビュー]タブが表示されます。

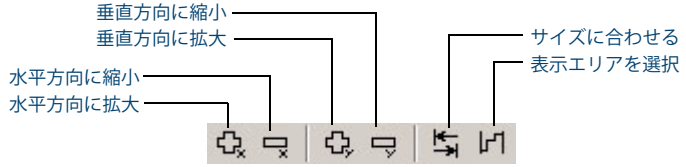
検証結果ペインで使用されるアイコン

アイコン	説明	アイコンが表示されるタブ
	メモリ リーク	サマリ、メモリ リーク、通知情報
	その他のリーク	サマリ、その他のリーク、通知情報
	エラー	サマリ、エラー、通知情報
	.NETパフォーマンス	サマリ、NETパフォーマンス
	モジュールのロード イベント	サマリ、モジュール、通知情報
	サブルーチン コール	通知情報
	ガベージ コレクション イベント	通知情報
	イベントの開始	通知情報
	イベントの再開	通知情報
	イベントの終了	通知情報

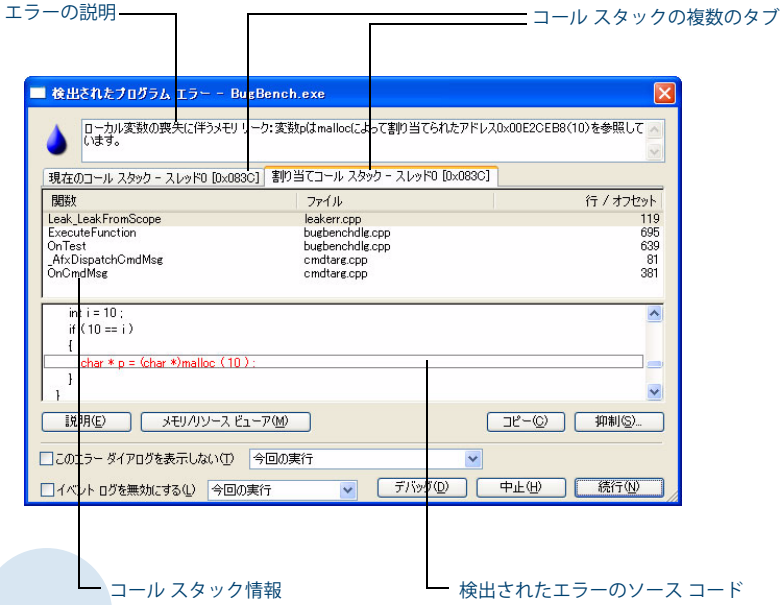
詳細ペインで使用されるアイコン

アイコン	説明
	サブルーチン コール
	開始パラメータ
	終了パラメータ
	戻り値
	データ型のプロパティ（デフォルト）
	データ型のプロパティ

参照回数グラフのツールバー



[検出されたプログラム エラー] ダイアログ ボックス



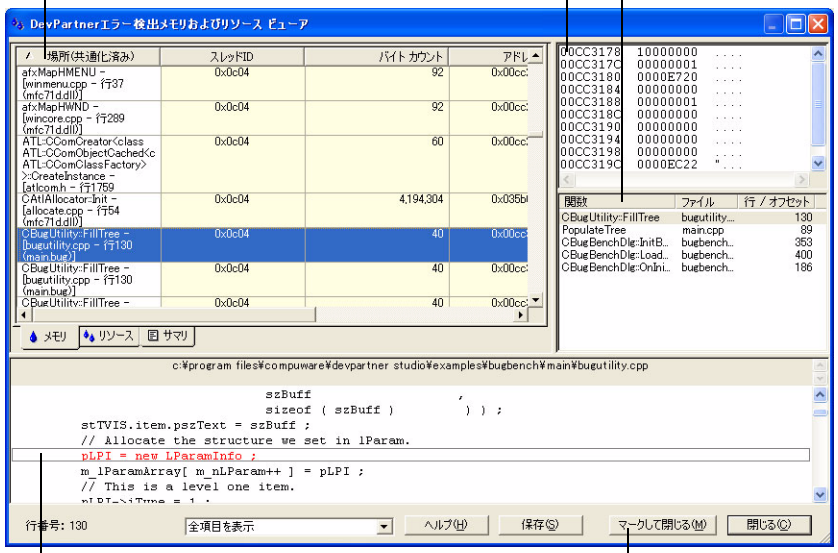
[メモリおよびリソース ビューア] ダイアログ ボックス

検証結果ペイン

[メモリ] タブ、[リソース] タブ、および [サマリ] タブで表示されます。

メモリ コンテンツ ペイン

スタック ペイン



ソース ペイン

検出されたエラーのソース コードがあれば、表示されます。

マークして閉じる

既存の割り当てをマークしたあとダイアログ ボックスを閉じる場合に、クリックします。[メモリおよびリソース ビューア] を次に開いたとき、マークされた項目は表示されません。

ActiveCheck と FinalCheck によるエラー検出

ActiveCheck

ActiveCheck™ はプログラムを分析し、プログラム実行ファイル、およびプログラムで使用されているダイナミック リンク ライブラリ（DLL）、他社製モジュール、COM コンポーネント内のエラーを検索します。以下の表に、ActiveCheck エラー検出機能によって検出されるエラーの種類を示します。

デッドロック関連エラー	API エラーと COM エラー
デッドロック	COM インターフェイス メソッドの失敗
潜在的なデッドロック	不正な引数
スレッドのデッドロック	パラメータ範囲エラー
クリティカル セクションのエラー	スレッドの不正な使用
セマフォ エラー	Windows 関数が失敗した場合
リソースの使用とネーミング エラー	Windows 関数が実装されていない場合
問題のある可能性が高いリソース使用状況	不正な COM インターフェイス メソッド引数
ハンドル エラー	
イベント エラー	
ミューテックス エラー	
Windows イベント エラー	

.NET エラー	ポインタ エラーとリーク エラー
ファイナライザ エラー	インターフェイス リーク
GC.Suppress finalize が呼び出されていない場合	メモリ リーク
Dispose 属性エラー	リソース リーク
処理されていないネイティブの例外がマネージ コードに渡された場合	

メモリ エラー
ダイナミック メモリ オーバーラン
ロックされているハンドルを解放しようとしている場合
ハンドルがすでにアンロックされている場合
メモリ割り当ての競合
アンロックされたメモリ ブロックをポインタが参照
スタック メモリ オーバーラン
スタティック メモリ オーバーラン

FinalCheck のコンパイル時インストゥルメンテーション－徹底したエラー検出

FinalCheck™ コンパイル時インストゥルメンテーション（CTI）を使用すると、メモリ リーク、ポインタ エラー、データ破壊エラーなどのエラーも、発生するたびにリアルタイムで検出されます。FinalCheck では、ActiveCheck で検出されるすべてのエラーのほか、以下のエラーが検出されます。

メモリ エラー	ポインタ エラーとリーク エラー
バッファ読み込みオーバーフロー	範囲を超えた配列の読み込み
未初期化メモリからの読み込み	有効範囲外を示すポインタのコピー
バッファ書き込みオーバーフロー	ダングリング ポインタの演算
	非関連ポインタの演算
	関数を示していない関数ポインタ
	リークによるリーク
	モジュール アンロードによるリーク
	アンワインドによるリーク
	メモリ領域の解放に伴うメモリ リーク
	メモリの再割り当てに伴うメモリ リーク
	ローカル変数の喪失に伴うメモリ リーク
	ローカル変数を指すポインタを返している場合

使用可能なコマンド キーのリスト－Visual Studio

コマンド	動作
Ctrl+Shift+O	[ファイル]>[開く]>[プロジェクト]
Ctrl+Shift+N	[ファイル]>[新規作成]>[プロジェクト]
Ctrl+S	[ファイル]>[プロジェクトの保存]
Ctrl+Shift+S	[ファイル]>[すべて保存]
Ctrl+Shift+F	[編集]>[ファイル内の検索]
Ctrl+Shift+H	[編集]>[ファイル内の置換]
Alt+F12	[編集]>[シンボルの検索]
Ctrl+Alt+L	[表示]>[ソリューション エクスプローラ]
Ctrl+Shift+C	[表示]>[クラス ビュー]
Ctrl+Alt+S	[表示]>[サーバー エクスプローラ]
Ctrl+Shift+E	[表示]>[リソース ビュー]
F4	[表示]>[プロパティ ウィンドウ]
Ctrl+Alt+X	[表示]>[ツールボックス]
Shift+Alt+Enter	[表示]>[全画面表示]
Shift+F4	[表示]>[プロパティ ページ]
Ctrl+Shift+B	[ビルド]>[ソリューションのビルド]
F5	[デバッグ]>[開始]
Ctrl+F5	[デバッグ]>[デバッグなしで開始]
Ctrl+Alt+E	[デバッグ]>[例外]
F11	[デバッグ]>[ステップイン]
F10	[デバッグ]>[ステップ オーバー]
Ctrl+B	[デバッグ]>[ブレークポイントの作成]
Ctrl+F1	[ヘルプ]>[ダイナミック ヘルプ]*
Ctrl+Alt+F1	[ヘルプ]>[目次]*
Ctrl+Alt+F2	[ヘルプ]>[キーワード]*
Ctrl+Alt+F3	[ヘルプ]>[検索]*
Shift+Alt+F2	[ヘルプ]>[キーワード検索の結果]*
Shift+Alt+F3	[ヘルプ]>[検索結果]*
	* Visual Studio 2005、2008

DevPartner データのエクスポート：コマンド ラインの使用

コマンド ラインから DevPartner.Analysis.DataExport.exe を使用して、DevPartner カバレッジ分析（.dpcov）、カバレッジ分析マージ（.dpmrg）、パフォーマンス分析（.dpprf）、およびパフォーマンス エキスパート（.dppxp）のセッション ファイル データを XML ファイルに変換することができます。

セッション データを XML にエクスポートするには、以下の構文を使用します。

```
DevPartner.Analysis.DataExport.exe ［セッション ファイル名 | ディレクトリへのパス］｛オプション｝
```

オプション

以下の表に、DevPartner.Analysis.DataExport.exe のコマンド ライン オプションの一覧を示します。

指定するオプションとオプション値を区切るには、等号、コロン、スペースのいずれかを使用します。

スイッチ	説明
/out[put]=<String>	エクスポートする XML ファイルのローカルまたはリモートの出力ディレクトリを指定します。ディレクトリが存在しない場合は作成します。
/r[ecurse]	DevPartner セッション ファイルのサブディレクトリを検索します。
/f[ilename]=<String>	XML 出力ファイルの名前を指定します。指定した名前に .xml が付加されます。
/showAll	パフォーマンス分析またはカバレッジ分析のセッション ファイルで利用可能な、すべてのパフォーマンス分析およびカバレッジ分析のセッション ファイル データが表示されます。 たとえば、このオプションを指定してパフォーマンス セッション ファイルをエクスポートすると、結果の XML ファイルにはパフォーマンスとカバレッジの両方のデータ フィールドが含まれます。 このオプションは、パフォーマンス エキスパート ファイルには利用できません。
/w[ait]	ユーザーの入力を待機して、コンソール ウィンドウを閉じます。
/nologo	ロゴや著作権情報を表示しません。
/help または /?	コンソール ウィンドウにヘルプを表示します。
/summary	パフォーマンス エキスパートのサマリ データをエクスポートします。CPU リソースを最も多く使用したコール パスとメソッドをエクスポートします。デフォルトでは、最大で上から 10 番めまでのコール パスとメソッドがエクスポートされます。/maxpaths オプションと /maxmethods オプションを使用すると、最大数を上書きできます。
/method	パフォーマンス エキスパートのメソッド データをエクスポートします。
/calltree	パフォーマンス エキスパートのコール ツリー データをエクスポートします。
/maxpaths=<integer>	パフォーマンス エキスパートの /summary オプションと共に使用します。CPU リソースを最も多く使用した上位のコール パスを、指定の数だけエクスポートします。
/maxmethods=<integer>	パフォーマンス エキスパートの /summary オプションと共に使用します。CPU リソースを最も多く使用した上位のメソッドを、指定の数だけエクスポートします。