Natural for Windows

デバッガ

バージョン 6.3.3

October 2008

Natural

This document applies to Natural バージョン 6.3.3 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

目次

1 デバッガ

このドキュメントは、『*Natural スタジオの使用*』の補足ドキュメントです。Natural アプリケーションのデバッグ方法について説明します。 次の項目で構成されています。

全般的な情報	Natural スタジオに統合されているデバッガについて説明します。 リモートデバッグやそのための環境の設定手順について説明します。
デバッガの開始と終了	SYMGEN パラメータに関する情報です。 デバッガの開始、再スタート、および終了の方法について説明します。
デバッガの要素	デバッガを開始したときに Natural スタジオウィンドウで利用可能な追加要素について説明します。
コードの参照	コードのステップ実行方法、およびブレイクポイント、ウォッチポイント、イベント、またはカーソル位置への移動方法について説明します。
ブレイクポイントとウォッチポイントの設定	ブレイクポイントとウォッチポイントの追加方法、およびブレイクポイントとウォッチポイントウィンドウでの管理方法について説明します。
変数の変更と監視	変数の変更、ウォッチ変数の追加、および変数ウィンドウでの変数の管理の方法について説明します。
コールスタックの使用	コールスタックウィンドウでのオブジェクトの管理方法について説明します。

Natural スタジオに統合されている新しいデバッガに関する上記のトピックの他に、セクション「*以前のデバッガの使用*」では以前のデバッガに関するドキュメントが用意されています。 開発サーバーに Natural の旧バージョンがインストールされていると、以前のデバッガが表示されることがあります。「*全般的な情報*」も参照してください。

2 全般的な情報

このchapterでは、次のトピックについて説明します。

デバッガの概要

Natural for Windows バージョン 6.2 以降、デバッガは Natural スタジオに統合されています。これにより、すべての Natural スタジオ機能をデバッガと併用できます。 例えば、デバッガがアクティブのときに、ライブラリワークスペースにある別のオブジェクトに移動したり、［オブジェクト検索］コマンドを使用して特定のオブジェクトを検索したりできます。

デバッガは、次の環境で Natural アプリケーションをデバッグするために使用します。

- ローカル環境。
- リモート環境。具体的には、マップされた開発サーバー（SPoD）。次のいずれかのバージョンが開発サーバーにインストールされていることが前提条件です。
  - Natural for Mainframes バージョン 4.2 以上
  - Natural for UNIX バージョン 6.2 以上

デバッグに追加設定は不要です。 Natural スタジオがすべてのステップを内部的に処理します（対応サーバーとの通信の設定や切断など）。

『*Natural スタジオの使用*』ドキュメントの「*ライブラリワークスペースの環境とビュー*」および『*SPoD を使用したリモート開発*』の「*リモート開発環境へのアクセス*」も参照してください。

Note: アプリケーションをリモートメインフレーム環境でデバッグする場合に、いくつか相違点があります。 この相違点については、各プラットフォームの『Natural 開発サーバー』（NDV）ドキュメントに挙げられており、今回の Natural リリースも該当しています。NDV ドキュメントは個別に用意されており、この『Natural for Windows』ドキュメントの一部ではありません。

SPoD で以前のデバッガが使用される場合

Naturalスタジオでの作業中に、マップされた開発サーバー上のオブジェクトに対してデバッガを呼び出すと、統合された新しいデバッガではなく、以前のデバッガが呼び出されることがあります。これは、Naturalの以前のバージョンが開発サーバーにインストールされている場合の動作です。 以前のバージョンとは以下のとおりです。

- Natural for UNIX バージョン 6.1.1 以下

「*以前のデバッガの使用*」を参照してください。

リモートデバッグ

今後のバージョンのいずれかで、リモートデバッグがサポートされなくなる予定です。 そのために、Natural スタジオに統合されているデバッガを使用する必要性が生じます。

リモートデバッグが実行されるのは、ネイティブ Natural for UNIX アプリケーションを Windows コンピュータでデバッグする場合です。または Natural ダイアログアプリケーションを他の PC からリモートでデバッグする場合です。 この動作は SPoD のコンテキスト外で行われます。

リモートデバッグを有効にするには、以下の手順を実行します。

- デバッグフロントエンドを Windows コンピュータにインストールします。 これにより、リモートデバッグ用にアクティブにする必要があるリモートデバッグサービス `natdbgsv` がインストールされます。 「*リモートデバッガのインストール*」を参照してください。
- デバッグ対象アプリケーションが存在する環境のパラメータ `RDNODE`、`RDPORT`、および `RDACTIVE` を定義します。 詳細については、「*リモートデバッグのための環境の設定*」を参照してください。
- デバッグ対象アプリケーションが存在する環境でシステムコマンド `DEBUG object-name` を入力して、デバッガを呼び出します。

以下では次のトピックについて説明します。

Important: リモートデバッガを Microsoft Windows XP 環境で使用する場合は、パーソナルファイアウォールを無効にする必要があります。 Natural for Windows の『*オペレーション*』ドキュメントの「*Natural を実行するための Microsoft Windows XP パーソナルファイアウォールの設定*」を参照してください。

リモートデバッガのインストール

Natural for Windows がインストール済みの場合は、Natural for Windows に付属のリモートデバッガを使用する必要があります。リモートデバッガがまだインストールされていない場合は、Natural インストールパッケージで［変更］オプションを使用して、リモートデバッガを Natural for Windows インストールに追加します。 Natural for Windows の『*インストール*』ドキュメントで「*Natural または Natural ランタイム環境のメンテナンス*」を参照してください。

Natural for Windows がインストールされていない場合は、リモートデバッガをスタンドアロンでインストールします。 UNIX プラットフォーム上の Natural アプリケーションをデバッグする場合は、*$NATDIR/$NATVERS/dbrmt/I386/nrd.exe* を UNIX インストール媒体から Windows コンピュータ（の一時ディレクトリなど）へコピーし、解凍します。 *setup.exe* を実行して、リモートデバッガのインストールを開始します。

リモートデバッグのための環境の設定

以下では次のトピックについて説明します。

Windows側（端末サービスなし）

リモートデバッガをインストールするか（該当ファイルは UNIX インストール媒体に収録されています）、Natural for Windows をインストールします（リモートデバッガは Natural for Windows のカスタムインストールを使用してオプションでインストールできます。Natural for Windows の『*インストール*』ドキュメントを参照してください）。これにより、Natural リモートデバッグサービス `natdbgsv` がインストールされます。

リモートデバッグサービスをアンインストールするには、コマンド行で「`natdbgsv -u`」と入力します。現在のサービスのポート名およびバージョンを表示するには、「`natdbgsv -s`」と入力します。 サービスを別のポートに再インストールするには、いったんアンインストールしてから「`natdbgsv -i portnumber`」と入力します。*portnumber* は `RDPORT` プロファイルパラメータの値です。指定したポート番号がすでに使用されている場合は、ダイアログが表示されるので、別のポート番号を入力します。

Note: リモートデバッグサービスを 2600（デフォルト値）以外のポートにインストールする場合は、その前に `RDPORT` プロファイルパラメータの値を変更して、Natural アプリケーションのデバッグに使用するクライアントコンピュータのポート番号と一致させる必要があります。

Windows側（端末サービスあり）

リモートデバッガをインストールするか（該当ファイルは UNIX インストール媒体に収録されています）、Natural for Windows をインストールします（リモートデバッガは Natural for Windows のカスタムインストールを使用してオプションでインストールできます。Natural for Windows の『*インストール*』ドキュメントを参照してください）。これにより、［スタート］メニューの Natural へのショートカットがあるプログラムフォルダにデバッガへのショートカットが作成されます。これは、リスナプロセス `natdbgsv` です。 リモートデバッグを使用するには、`natdbgsv` が開始されている必要があります。 このリスナプロセスを特定のユーザーセッションで初めて開始したときに、空きポート番号が表示されます。この番号を `RDPORT` プロファイルパラメータの該当フィールドに入力する必要があります。

その後、`natdbgsv` をアクティブにすると、リスナはこのポート番号で開始されます。 この番号が他のアプリケーションですでに使用されている場合は、`natdbgsv` のポートダイアログで別のポート番号を指定し、`RDPORT` も併せて変更する必要があります。

Natural側

以下のプロファイルパラメータ設定によって Natural を起動します。

- RDACTIVE を "ON" に設定します。
- RDNODE を Windows サーバーのノード名に設定します。
- RDPORT を "2600" または他のポート番号に設定します。他のポート番号を指定する場合は、リモートデバッグサービスをインストールしたポートの番号（「*Windows 側（端末サービスなし）*」を参照）またはリスナプロセスが開始されたポートの番号（「*Windows 側（端末サービスあり）*」を参照）を指定します。

リモートデバッグのシナリオ

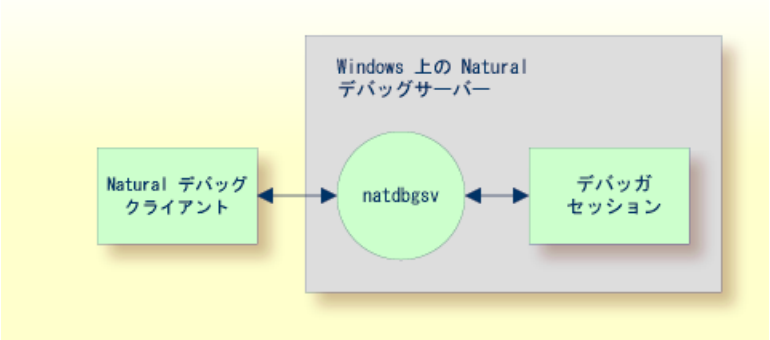
リモートデバッグを使用する方法にはいくつかのシナリオがあります。例えば、単一のNaturalクライアントを単一のリモートデバッグセッションの制御下で実行したり、分散Naturalアプリケーションを複数のリモートデバッグセッションの制御下で実行したりします。分散アプリケーションには、Natural RPC サーバーや Natural DCOM サーバー、さらには Natural で記述されていないコンポーネント、例えば Visual Basic クライアントが含まれていることがあります。

以下では次のトピックについて説明します。

- シナリオ 1：単一の Natural アプリケーションのデバッグ
- シナリオ 2：分散 Natural アプリケーションのデバッグ
- シナリオ 3：異機種アプリケーションの Natural 部分のデバッグ

シナリオ 1：単一の Natural アプリケーションのデバッグ

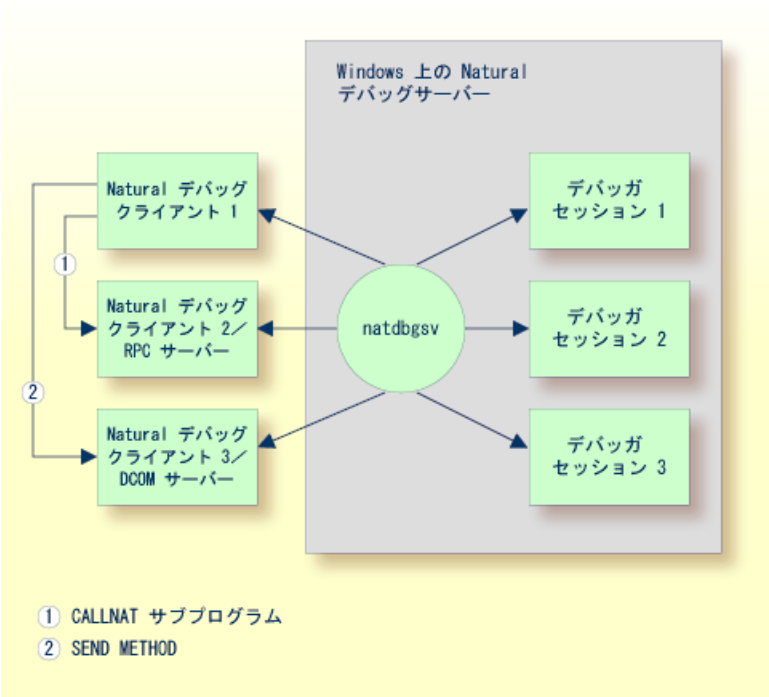
以下の図は、単一の Natural アプリケーションのデバッグを示しています。



シナリオ 2：分散 Natural アプリケーションのデバッグ

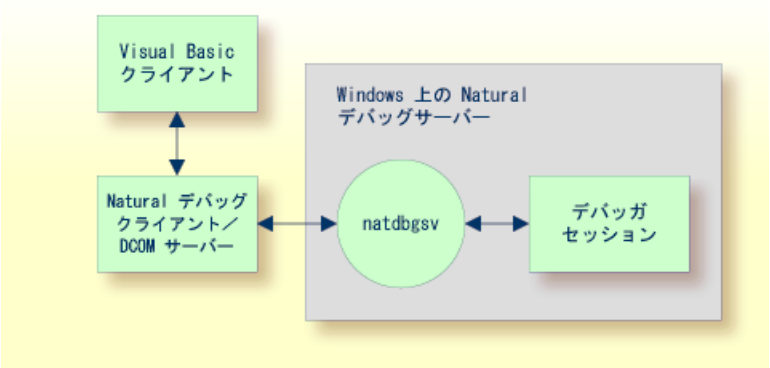
以下の分散 Natural アプリケーションの各コンポーネントをデバッグするには、Natural デバッグクライアント 1 のコマンド行に `DEBUG objectname` を入力します。 Natural デバッグクライアントが Natural RPC サーバー上のサブプログラムを初めて呼び出すと、新しいデバッグセッションが RPC サーバー用に開きます。 RPC サーバーの処理がデバッグされます。 デバッグセッションは、RPC サーバーが終了すると直ちに閉じられます。

Natural DCOM サーバーの場合も同様です。



シナリオ 3：異機種アプリケーションの Natural 部分のデバッグ

前述のシナリオと同様に、DCOM サーバー上のメソッドが初めて呼び出されると、新しいデバッグセッションが DCOM サーバー用に開きます。DCOM サーバーの処理がデバッグされ、DCOM サーバーが終了するとデバッガセッションも閉じます。


Visual Basic
クライアント
Natural デバッグ
クライアント／
DCOM サーバー
Windows 上の Natural
デバッグサーバー
natdbgsv
デバッガ
セッション

3 デバッガの開始と終了

このchapterでは、次のトピックについて説明します。

デバッガの使用準備

デバッガの全機能を使用できるようにするには、パラメータ SYMGEN を "ON" に設定する必要があります。 このパラメータは以下のいずれかの方法で設定できます。

- Natural の起動時にダイナミックに設定
- セッションパラメータを変更して現在のセッションにのみ設定
- コンフィグレーションユーティリティを使用してパラメータファイルに設定

オブジェクトをカタログ化または格納し、SYMGEN を "ON" に設定すると、生成プログラムの一部としてシンボルテーブルが生成されます。 このテーブルには、このオブジェクトに対して有効な変数に関連する情報が含まれているため、SYMGEN を指定せずに変数にアクセスすることはできません。ただし、オブジェクトをデバッグすることはできます。

Note: メインフレーム上の SPoD 環境でデバッグしている場合は、パラメータ SYMGEN の設定は不要です。

デバッガの開始

デバッガは、格納またはカタログされた Natural プログラムおよびダイアログで使用できます。ローカル環境でもリモート環境でも使用可能です。

システムコマンド DEBUG の説明も参照してください。

手順 3.1. デバッガを開始するには

1 デバッグ対象のオブジェクトのエディタを開きます。

Or:

目的のオブジェクトをライブラリワークスペースで選択します。

2 ［デバッグ］メニューで［開始］を選択します。

Or:

Ctrl キーを押したまま F7 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

Or:

ライブラリワークスペースでオブジェクトを選択したら、コンテキストメニューを表示し、［デバッグ］を選択します。

選択したオブジェクトのエディタがそれまで開いていなかった場合は、これで開きます。

ダイアログのダイアログソースは別ウィンドウに表示されます。

デバッガが開始されると、Naturalスタジオウィンドウで追加要素が使用可能になります。詳細については、「*デバッガの要素*」を参照してください。

デバッガの再スタート

デバッグセッションを再スタートすると、デバッガの位置はアプリケーションの先頭に戻りますが、ブレイクポイント、ウォッチポイント、およびウォッチ変数の現在の設定はすべて保持されます。このため、デバッグに関連する設定を指定せずにアプリケーションを再実行する場合は、デバッグセッションの再起動が役に立ちます。

手順 3.2. デバッガを再スタートするには

■ ［デバッグ］メニューで［再スタート］を選択します。

Or:

Ctrl + Shift + F7 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

デバッガの終了

アプリケーションがエラーなしに終了すると、デバッガは自動的に終了します。 デバッガは、自動終了する前に停止させることもできます。以下の説明を参照してください。

Note: エディタウィンドウを閉じてもデバッガは終了しません。

デバッガを終了または停止すると、ブレイクポイント、ウォッチポイント、およびウォッチ変数の設定が自動的に保存されます。 これらすべての設定は、デバッガを次回開始したときに復元されます。

エラーがあった場合は、該当するソースが表示され、トレース位置によってエラー発生行が示されます。 メッセージウィンドウに、適切なエラーメッセージと、デバッグセッションを継続するか終了するかの選択肢が表示されます。 アプリケーションにエラー処理（エラートランザクションを含む）が含まれている場合や、デバッグセッションを終了する前に変数の内容を表示する場合などには、デバッグセッションの継続が有効なことがあります。

手順 3.3. デバッガを停止するには

■ ［デバッグ］メニューで［停止］を選択します。

Or:

Shift キーを押したまま F7 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

デバッグセッションが終了し、制御が Natural に戻ります。

4 デバッガの要素

デバッガが開始されると、Natural スタジオウィンドウで追加要素が使用可能になります。

タイトルバーに表示されるデバッガ情報

Natural スタジオウィンドウのタイトルバーには、以下のいずれかが表示されます。

- **[中断]**
  タイトルバーに "[中断]" と表示されているとき、制御の主体はデバッガの側にあります。
- **[実行中]**
  "[実行中]" がタイトルバーに表示されているときは、現在デバッグ中の Natural アプリケーションに制御があります。

SPoD を使用するリモート環境のオブジェクトをデバッグしている場合は、タイトルバーにホストのポート番号も表示されます。

メニューコマンド

［デバッグ］メニューのコマンドはデバッガに適用されます。

デバッガが開始されていないときは、［デバッグ］メニューには［開始］コマンドのみ表示されます。デバッガが開始されると、［デバッグ］メニューの他のコマンドが有効になり、［開始］コマンドの代わりに［実行］コマンドが表示されます。

エディタウィンドウがアクティブになり、このウィンドウのオブジェクトに対してデバッガが開始されると、コンテキストメニューにデバッガに適用されるコマンドが表示されます。 デバッガが開始されていないときは、コンテキストメニューではデバッグコマンド［ブレイクポイント **ON/OFF**］のみ使用できます。

これらのコマンドの詳細については、このドキュメントで後述します。

ツールバー

デバッガには専用のツールバーがあり、［デバッグ］メニューのコマンドに簡単にアクセスできます。 デバッガが開始されていないときは、［デバッグ］ツールバーでは［開始］および［ブレイクポイント **ON/OFF**］コマンドに対応するツールバーボタンのみ有効になります。 デバッガが開始されると、他のすべてのツールバーボタンが有効になります。

［デバッグ］ツールバーのボタンは、以下のメニューコマンドを表します。

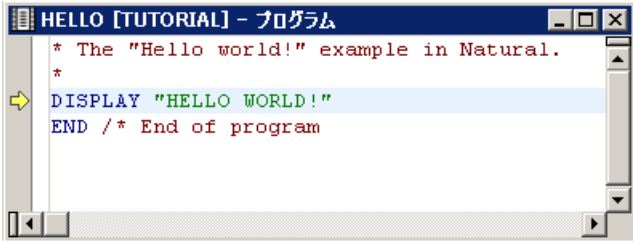
- 開始（デバッガが開始されていないときのみ表示）
- 実行（デバッガが開始されているときのみ表示）
- 再スタート
- 停止
- ステップオーバー
- ステップイン
- ステップアウト
- トレース位置の表示
- ブレイクポイント **ON/OFF**
- 変数の変更

［デバッグ］ツールバーは、表示と非表示を切り替えることができます。 詳細については、『*Natural スタジオの使用*』ドキュメントの「*Natural スタジオのカスタマイズ*」を参照してください。

エディタウィンドウでのトレース位置

エディタウィンドウでは、現在のトレース位置が左マージンに矢印で示されます。

デバッガが開始されると、トレース位置は最初に実行可能なソースコード行を示します。 例：



エディタウィンドウをスクロールしてトレース位置が見えなくなったときは、以下の手順でトレース位置に戻ることができます。

Note: 「*現在のトレース位置のオブジェクトに戻る*」も参照してください。

手順 4.1. トレース位置に戻るには

■ ［デバッグ］メニューで［トレース位置の表示］を選択します。

Or:

Alt キーを押しながら NUM* キーを選択します。

Note: NUM* は、数値キーパッド上の乗算キーです。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

デバッガのウィンドウ

デバッガが開始されると、以下のウィンドウが表示されます。

- 変数
- ブレイクポイントとウォッチポイント
- コールスタック

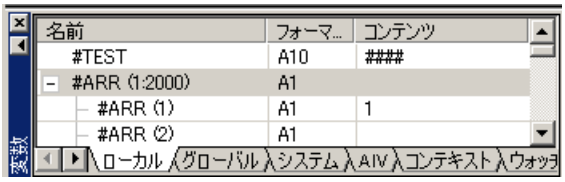
デバッガウィンドウの各タブにはコンテキストメニューがあり、タブ全体を対象とするコマンド（エントリが選択されていない場合）と選択されているエントリを対象とするコマンドのいずれかが表示されます。 これらのコマンドの詳細については、このドキュメントで後述します。

デバッガウィンドウは移動したりドッキングしたりすることができます。『*Natural スタジオの使用*』ドキュメントの「*ドッキング可能なウィンドウ*」を参照してください。

Note: 以前に［表示］メニューのコマンドを使用してデバッガウィンドウの表示を変更した場合は、このデバッガウィンドウ（アクティブな環境で定義されているブレイクポイントとウォッチポイントが表示されている）が、以下で説明するデバッガウィンドウに置き換えられます。 『*Natural スタジオの使用*』ドキュメントの「*デバッガウィンドウ*」も参照してください。

変数

このウィンドウには、プログラム実行の現在の状態で使用できるすべての変数が表示されます。



変数名の左側にある展開／圧縮トグルは、グループフィールド、配列フィールド、または再定義されたフィールドを示します。再定義されたフィールドのトグルには、"R"が表示されます（例：  ）。

変数はいくつかのカテゴリに分類されます。 カテゴリごとにタブがあります。

- ローカル
  生成されたアクティブなプログラムで使用されているローカル変数が表示されます。
- グローバル
  参照されているグローバルデータエリアのグローバル変数が表示されます。
- システム
  現在のプラットフォームのすべてのシステム変数が表示されます。 例えば、マップされたUNIX環境でアプリケーションのデバッグを現在実行している場合は、UNIXに有効なすべてのシステム変数が表示されます。
- **AIV**
  アプリケーションで現在使用可能なアプリケーション非依存変数（AIV）が表示されます。
- コンテキスト
  アプリケーションで現在使用可能なコンテキスト変数が表示されます。
- ウォッチ
  監視するためにユーザーが追加した変数が表示されます。 「*ウォッチ変数の追加*」を参照してください。

変数ウィンドウの下部で該当するタブを選択することで、さまざまな種類の変数の表示を切り替えることができます。

詳細については、「*変数の変更と監視*」を参照してください。

手順 4.2. 変数ウィンドウの表示／非表示を切り替えるには

- ［デバッグ］メニューで、［**Windows**］、［変数］の順に選択します。

  Or:

  Ctrl + Alt + 1 キーを押します。

変数ウィンドウがNaturalスタジオウィンドウに表示されると、このメニューコマンドの横にチェックマークが付きます。

手順 4.3. ショートカットキーを使用して変数ウィンドウをアクティブにするには

- 変数ウィンドウが Natural スタジオウィンドウに表示されたら、Ctrl + Shift + V キーを押します。これによりアクティブになります。

ブレイクポイントとウォッチポイント

このウィンドウには、現在定義されているすべてのブレイクポイントとウォッチポイントが表示されます。

ブレイクポイントとウォッチポイントウィンドウの下部で該当するタブを選択することで、ウォッチポイントとブレイクポイントの表示を切り替えることができます。

詳細については、「*ブレイクポイントとウォッチポイントの設定*」を参照してください。

手順 4.4. ブレイクポイントとウォッチポイントウィンドウの表示を切り替えるには

- ［デバッグ］メニューで、［**Windows**］、［ブレイクポイントとウォッチポイント］の順に選択します。

  Or:

  Ctrl + Alt + 2 キーを押します。

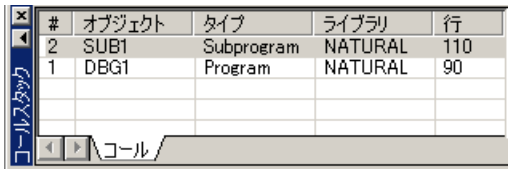
  ブレイクポイントとウォッチポイントウィンドウがNaturalスタジオウィンドウに表示されると、メニューコマンドの横にチェックマークが付きます。

手順 4.5. ショートカットキーを使用してブレイクポイントとウォッチポイントウィンドウをアクティブにするには

- ブレイクポイントとウォッチポイントウィンドウがNaturalスタジオウィンドウに表示されたら、Ctrl + Shift + B キーを押します。これによりアクティブになります。

コールスタック

このウィンドウには、現在のデバッグセッション中に呼び出されたオブジェクトが階層形式で表示されます。



詳細については、「*コールスタックの使用*」を参照してください。

手順 4.6. コールスタックウィンドウの表示／非表示を切り替えるには

■ ［デバッグ］メニューで、［**Windows**］、［コールスタック］の順に選択します。

Or:

Ctrl + Alt + 3 キーを押します。

コールスタックウィンドウがNaturalスタジオウィンドウに表示されると、このメニューコマンドの横にチェックマークが付きます。

手順 4.7. ショートカットキーを使用してコールスタックウィンドウをアクティブにするには

■ コールスタックウィンドウがNaturalスタジオウィンドウに表示されたら、Ctrl+Shift+C キーを押します。これによりアクティブになります。

5 コードの参照

このchapterでは、次のトピックについて説明します。

コードのステップ実行

デバッガに、次のプログラムステップを実行するよう指示できます。 この操作に使用できるコマンドはいくつかあります。

- 他のオブジェクトのステップオーバー
- 他のオブジェクトへのステップイン
- 他のオブジェクトからのステップアウト

他のオブジェクトのステップオーバー

他のオブジェクトをステップオーバーするようデバッガに指示すると、次のプログラムステップが実行され、該当するソースコード行でトレース位置が示されます。 このソースコード行が他のNaturalオブジェクトを呼び出すか、このソースコード行に他のNaturalオブジェクトが含まれている場合は、デバッガはこのオブジェクトをステップオーバーし、このオブジェクトのすべてのソースコードが一度に実行されます。 ただし、このオブジェクトにウォッチポイントまたはブレイクポイントが含まれているとデバッガは停止します。

手順 5.1. 他のオブジェクトをステップオーバーするには

■ ［デバッグ］メニューで［ステップオーバー］を選択します。

Or:

F10 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

他のオブジェクトへのステップイン

他のオブジェクトにステップインするようデバッガに指示すると、次のプログラムステップが実行され、該当するソースコード行でトレース位置が示されます。 このソースコード行が他のNatural オブジェクトを呼び出すか、このソースコード行に他の Natural オブジェクトが含まれている場合は、デバッガはこのオブジェクトにステップインし、最初の実行可能な行でトレース位置が示されます。

手順 5.2. 他のオブジェクトにステップインするには

■ ［デバッグ］メニューで［ステップイン］を選択します。

Or:

F11 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

他のオブジェクトからのステップアウト

他のオブジェクトからステップアウトするようデバッガに指示すると、デバッガは以前のプログラムレベルに戻ります。 ただし、以前のレベルに到達する前にウォッチポイントまたはブレイクポイントが検出されるとデバッガは停止します。

このコマンドは、サブプログラムのデバッグ中にサブプログラムの残りの部分の実行を続行するときに便利です。 実行は、中断されることなく継続され、呼び出し元プログラム内でサブプログラムが呼び出された位置の後ろで停止します。

手順 5.3. 他のオブジェクトからステップアウトするには

■ ［デバッグ］メニューで［ステップアウト］を選択します。

Or:

Ctrl キーを押したまま F11 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

次のブレイクポイントまたはウォッチポイントまでの実行

デバッガに、次の有効なブレイクポイントが検出されるかウォッチポイント条件が真になるまでオブジェクトを実行するよう指示できます。 この場合は、デバッガはウォッチポイントまたはブレイクポイントで停止し、該当するソースコード行でトレース位置が示されます。

手順 5.4. 次のウォッチポイントまたはブレイクポイントまで実行するには

■ ［デバッグ］メニューで［実行］を選択します。

Or:

F7 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

次のイベントまでの実行

デバッガに、イベントドリブンアプリケーションでアプリケーションに次のイベントが送信されるまでオブジェクトを実行するよう指示できます。 ただし、次のイベントが送信される前に有効なウォッチポイントまたはブレイクポイントが検出されるとデバッガは停止します。

手順 5.5. 次のイベントまで実行するには

■ ［デバッグ］メニューで［次のイベントまで実行］を選択します。

Note: 非イベントドリブンアプリケーションでは、このコマンドの機能は［実行］コマンドと同じです。

Or:

Alt キーを押したまま F7 キーを押します。

カーソル位置までの実行

デバッガに、現在カーソルがあるソースコード行までオブジェクトを実行するよう指示できます。

手順 5.6. カーソル位置まで実行するには

1 実行を一時停止させるソースコード行にカーソルを置きます。

2 エディタでコンテキストメニューを表示し、［カーソル行まで実行］を選択します。

Or:

Ctrl キーを押したまま F10 キーを押します。

次のステートメントまでの実行

デバッガに、コードをスキップし、カーソルを置いたソースコード行からオブジェクトの実行を再開するよう指示できます。 スキップされたコードは実行されません。

Caution: スキップするコードによっては、このコマンドによって予期しない結果が生じることがあります。 このコマンドの使用には十分な注意が必要です。

手順 5.7. 次のステートメントまで実行するには

1 実行を再開するソースコード行にカーソルを置きます。

2 エディタでコンテキストメニューを表示し、［次のステートメントを設定］を選択します。

Note: このコマンドは、現在処理中のオブジェクトに対してのみ使用可能です。

6 ブレイクポイントとウォッチポイントの設定

このchapterでは、次のトピックについて説明します。

ブレイクポイントとウォッチポイントの概要

プログラムには、デバッグ目的で以下の 2 種類のエントリを定義できます。

- ブレイクポイント
  ブレイクポイントとは、Naturalオブジェクトの実行中に制御がユーザーに返されるポイントです。

  1つのステートメントが複数行にわたる場合は、ブレイクポイントは先頭行以外には設定できません。

  ブレイクポイントを誤って非実行可能行（コメント行など）に設定しようとすると、ブレイクポイントは自動的に次の実行可能行に移動されます。
- ウォッチポイント
  ウォッチポイントを使用すると、エラーを含むオブジェクトによるNatural変数の予期しない変更を迅速に検出できます。

  デフォルトでは、ウォッチポイントを使用して、変数の内容が変更されたときにNaturalオブジェクトの実行を中断するようデバッガに指示します。 ただし、ウォッチポイントの設定時にウォッチポイント演算子を使用して変数に特定の値を指定すると、条件が設定され、その条件が真の場合のみウォッチポイントが有効になります。

  変数が変更されたと見なされるのは、変数の現在の値が、前回ウォッチポイントがトリガされたときに記録された値または初期値と異なる場合です。

各ブレイクポイントまたはウォッチポイントは、ブレイクポイントとウォッチポイントウィンドウの該当するタブに表示されます。 ブレイクポイントについては、各ブレイクポイントが定義されている行番号が表示されます。 ウォッチポイントについては、属している変数名に対応する名前が割り当てられ、ブレイク条件が表示されます。

タブの先頭列のチェックボックスを使用すると、ブレイクポイントまたはウォッチポイントの有効と無効をデバッグのセッション中にいつでも切り替えることができます。 「*ブレイクポイントとウォッチポイントの一時的な無効化*」を参照してください。

どのブレイクポイントまたはウォッチポイントにもヒットカウントがあり、デバッグエントリが通過するたびにカウントが増えます。 ただし、デバッグエントリの実行回数は、以下の方法で制限できます。

- ブレイクポイントまたはウォッチポイントが実行される前のスキップ回数を指定できます。 その後、イベントカウントが指定されているスキップの数より多くなるまで、デバッグエントリは無視されます。

- 実行回数の上限を指定して、イベントカウントが指定の実行回数を上回ったら直ちにブレイクポイントまたはウォッチポイントが無視されるようにすることができます。

ブレイクポイントまたはウォッチポイントがヒットした場所が、現在アクティブな別のオブジェクト内部だった場合は、新しいエディタウィンドウが開き、この新しいオブジェクトのソースが表示されます。

ブレイクポイントの追加と削除

ブレイクポイントは、ブレイクポイントとウォッチポイントウィンドウの［ブレイクポイント］タブで現在のカーソル位置に追加できます。 また、現在のカーソル位置にすでに設定されているブレイクポイントを、［ブレイクポイント］タブで削除できます。

この場合は、ダイアログボックスは表示されません。 ブレイクポイントの変更方法については（ブレイク回数の上限を定義するなど）、「*ブレイクポイントの変更*」を参照してください。

「*ブレイクポイントとウォッチポイントの削除*」も参照してください。

Note: ローカル環境でもリモート環境（SPoD）でも、デバッガがアクティブでないときに、以下の手順でブレイクポイントを設定または削除できます。 エディタで定義された各ブレイクポイントは、デバッガが開始されたときに検証されます。 許可されない位置に定義されたブレイクポイントは、次にブレイクポイントを定義できる行に移動されます。 『*Natural スタジオの使用*』ドキュメントの「*デバッガウィンドウ*」も参照してください。

手順 6.1. ブレイクポイントを切り替えるには

1 ブレイクポイントを設定または削除する行を選択します。

2 エディタでコンテキストメニューを表示し、［ブレイクポイント **ON/OFF**］を選択します。

Or:

F9 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

Or:

エディタウィンドウの左マージン（ブレイクポイントの記号が表示される場所）で、目的の行の横の位置をクリックします。

ブレイクポイントが設定されると、エディタウィンドウの左側マージンに記号が表示されます。「*エディタウィンドウで使用する記号*」を参照してください。ブレイクポイントのエントリは、ブレイクポイントとウォッチポイントウィンドウの［ブレイクポイント］タブにも表示されます。

ブレイクポイントが削除されると、対応する記号も表示されなくなります。ブレイクポイントのエントリが、ブレイクポイントとウォッチポイントウィンドウの［ブレイクポイント］タブから削除されます。

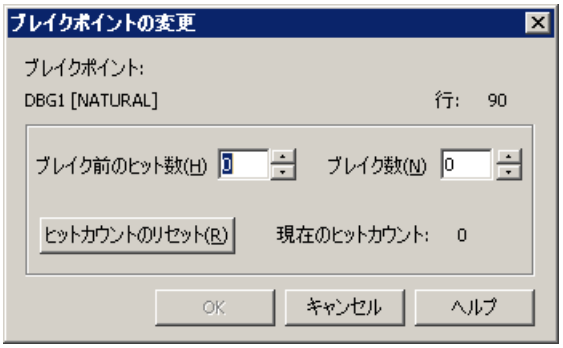
ブレイクポイントの変更

ブレイクポイントとウォッチポイントウィンドウに現在表示されている各ブレイクポイントを変更できます。

手順 6.2. ブレイクポイントを変更するには

1 目的のブレイクポイントを選択し、コンテキストメニューを表示し、［変更］を選択します。

次のダイアログボックスが表示されます。



2 必要なオプションを設定します。

ブレイク前のヒット数
: プログラムが所定の回数実行されるまでブレイクポイントが実行されないようにする場合の、ブレイクポイント実行前のスキップ回数です。 デフォルトは 0 です。

ブレイク数
: ブレイクポイントの実行回数の上限です。この回数に達すると、その後ブレイクポイントは無視されます。 デフォルトは 0 です。

ヒットカウントのリセット
: このコマンドボタンを選択すると、現在のヒットカウントは 0 にリセットされます。

3 ［**OK**］ボタンを選択します。

ウォッチポイントの追加

ウォッチポイントは、ブレイクポイントとウォッチポイントウィンドウの［ウォッチポイント］タブに表示されます。

ウォッチポイントの追加方法はいくつかあります。

- エディタウィンドウでのウォッチポイントの追加
- 変数ウィンドウでのウォッチポイントの追加
- ダイアログボックスを使用したウォッチポイントの追加

エディタウィンドウでのウォッチポイントの追加

エディタウィンドウの現在のカーソル位置にある変数をブレイクポイントとウォッチポイントウィンドウの［ウォッチポイント］タブに追加できます。 この場合はダイアログは表示されません。

手順 6.3. 変数をウォッチポイントとして定義するには

1 変数名内にカーソルを置いて、エディタで変数を選択します。

2 コンテキストメニューを表示し、［ウォッチポイントの追加］を選択します。

Or:

Ctrl + Shift + W キーを押します。

Or:

エディタで変数を選択します。 マウスを使用して、選択した変数をドラッグし、［ウォッチポイント］タブにドロップします。

変数ウィンドウでのウォッチポイントの追加

変数ウィンドウの変数をブレイクポイントとウォッチポイントウィンドウの［ウォッチポイント］タブに追加できます。 この場合はダイアログは表示されません。

手順 6.4. 変数をウォッチポイントとして定義するには

1 変数ウィンドウで必要な変数を選択します。

2 コンテキストメニューを表示し、［ウォッチポイントの追加］を選択します。

Or:

Ctrl + Shift + W キーを押します。

ダイアログボックスを使用したウォッチポイントの追加

ダイアログボックスを使用して、ブレイクポイントとウォッチポイントウィンドウの［ウォッチポイント］タブにウォッチポイントを追加できます。

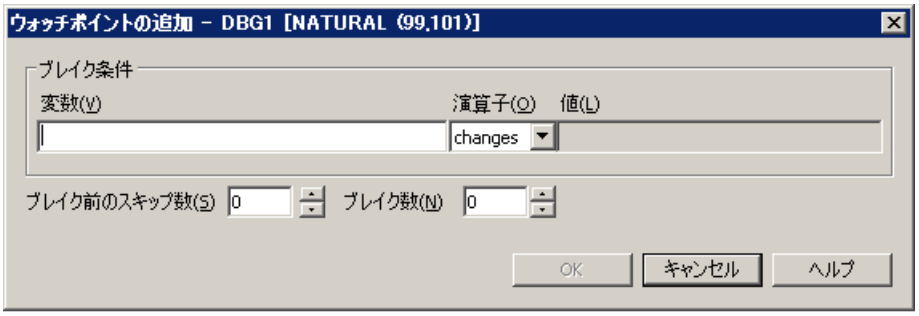
手順 6.5. ウォッチポイントを追加するには

1 ［デバッグ］メニューで［ウォッチポイントの追加］を選択します。

Or:

ブレイクポイントとウォッチポイントウィンドウの［ウォッチポイント］タブで、コンテキストメニューを表示し、［追加］を選択します。 他に選択されているエントリがないことを確認してください。 他にも選択されていると、コンテキストメニューにこのコマンドが表示されません。

［ウォッチポイントの追加］ダイアログボックスが開きます。 タイトルバーに、現在のプログラムやライブラリの名前、および現在の FUSER のデータベース ID とファイル番号が表示されます。



2 必要なオプションを設定します。

変数
デバッグ中のプログラムで監視する変数です。

演算子／値
ウォッチポイントの条件を定義するには、適切なウォッチポイント演算子を選択し、その演算子に対して値を指定します。条件を指定しない場合は、デフォルト設定（"変更"）が適用されます。

ウォッチポイント演算子は以下のとおりです。

演算子	ウォッチポイントが有効になる条件
変更	変数が変更されるたび。 デフォルトです。
EQ（=）	変数の現在値が指定値と等しい場合のみ。
NE（!=）	変数の現在値が指定値と等しくない場合のみ。
GT（>）	変数の現在値が指定値より大きい場合のみ。
LT（<）	変数の現在値が指定値より小さい場合のみ。
GE（>=）	変数の現在値が指定値以上の場合のみ。
LE（<=）	変数の現在値が指定値以下の場合のみ。

ブレイク前のスキップ数
プログラムが所定の回数実行されるまでウォッチポイントが実行されないようにする場合の、ウォッチポイント実行前のスキップ回数です。 デフォルトは 0 です。

ブレイク数
ウォッチポイントの実行回数の上限です。この回数に達すると、その後ウォッチポイントは無視されます。 デフォルトは 0 です。

3 ［**OK**］ボタンを選択します。

選択した変数の名前が、ブレイクポイントとウォッチポイントウィンドウの［ウォッチポイント］タブに表示されます。

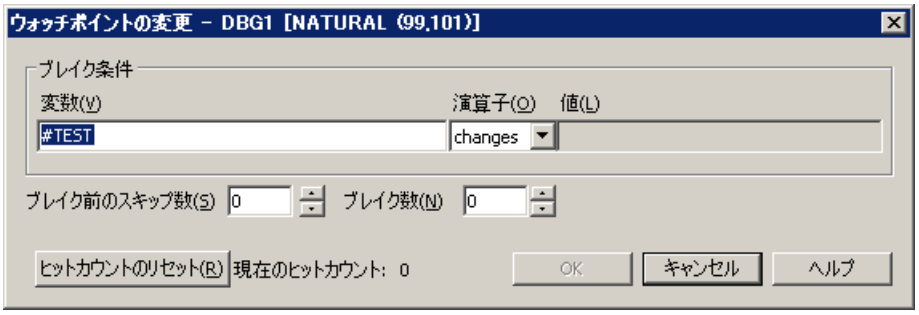
ウォッチポイントの変更

ブレイクポイントとウォッチポイントウィンドウに表示されている各ウォッチポイントを変更できます。

手順 6.6. ウォッチポイントを変更するには

1 目的のウォッチポイントを選択し、コンテキストメニューを表示し、［変更］を選択します。

［ウォッチポイントの変更］ダイアログボックスが開きます。



このダイアログボックスには、［ウォッチポイントの追加］ダイアログボックスと同じオプションがあります。 このダイアログボックスで指定できるオプションの詳細については、「*ダイアログボックスを使用したウォッチポイントの追加*」を参照してください。

このダイアログボックスには、［ヒットカウントのリセット］ボタンもあります。 このコマンドボタンを選択すると、現在のヒットカウントは 0 にリセットされます。

2 必要な変更を行い、［**OK**］ボタンをクリックします。

ブレイクポイントとウォッチポイントの一時的な無効化

定義されているブレイクポイントまたはウォッチポイントを個別に一時的に無効にできます。

手順 6.7. ブレイクポイントまたはウォッチポイントを無効にするには

1 ［ブレイクポイントとウォッチポイント］ウィンドウで必要なタブを選択します。

2 目的のエントリについて、タブの先頭列を選択してチェックマークをはずします。

Or:

コンテキストメニューを表示し、［有効／無効］を選択します。

ブレイクポイントを無効にすると、エディタウィンドウの左マージンに表示されている記号が変わります。 「*エディタウィンドウで使用する記号*」を参照してください。

手順 6.8. 無効になっているブレイクポイントまたはウォッチポイントを有効にするには

1 ［ブレイクポイントとウォッチポイント］ウィンドウで必要なタブを選択します。

2 目的のエントリについて、タブの先頭列を選択してチェックマークを再び付けます。

Or:

コンテキストメニューを表示し、［有効／無効］を選択します。

ブレイクポイントを有効にすると、エディタウィンドウの左マージンに表示されている記号が変わります。「*エディタウィンドウで使用する記号*」を参照してください。

手順 6.9. すべてのブレイクポイントまたはウォッチポイントを無効または有効にするには

1 ［ブレイクポイントとウォッチポイント］ウィンドウで必要なタブを選択します。

2 選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［すべて無効］または［すべて有効］を選択します。

すべてのブレイクポイントを無効または有効にすると、エディタウィンドウの左マージンに表示されている記号が変わります。「*エディタウィンドウで使用する記号*」を参照してください。

定義されているブレイクポイントまたはウォッチポイントのソースコードの表示

ブレイクポイントとウォッチポイントウィンドウに表示されている定義済みブレイクポイントまたはウォッチポイントそれぞれについて（有効かどうかに関係なく）、このブレイクポイントまたはウォッチポイントが定義されているソースに移動できます。

手順 6.10. ブレイクポイントまたはウォッチポイントが定義されているソースに移動するには

1 ［ブレイクポイントとウォッチポイント］ウィンドウで必要なタブを選択します。

2 目的のエントリを選択し、コンテキストメニューを表示し、［ソースコードに移動］を選択します。

Or:

目的のエントリをダブルクリックします。

ブレイクポイントの場合は、ブレイクポイントが定義されているソースコード行の横にトレース位置が示されます。

ウォッチポイントの場合は、ウォッチポイントが定義されているソースコード全体が表示されます。

ブレイクポイントとウォッチポイントの削除

選択したブレイクポイントまたはウォッチポイントを削除したり、すべてのブレイクポイントまたはウォッチポイントを削除したりすることができます。

「*ブレイクポイントの追加と削除*」も参照してください。

手順 6.11. ブレイクポイントまたはウォッチポイントを削除するには

1 ［ブレイクポイントとウォッチポイント］ウィンドウで必要なタブを選択します。

2 必要なエントリを選択します。

3 コンテキストメニューを表示し、［削除］を選択します。

Or:

Delete キーを押します。

手順 6.12. すべてのブレイクポイントまたはウォッチポイントを削除するには

1 ［ブレイクポイントとウォッチポイント］ウィンドウで必要なタブを選択します。

2 選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［すべて削除］を選択します。

エディタウィンドウで使用する記号

エディタウィンドウの左マージンには、以下の記号が表示されます。


この行には有効なブレイクポイントが含まれています。


この行には無効化されたブレイクポイントが含まれています。


この行には有効なブレイクポイントが含まれています。 トレース位置も含まれています。


この行には無効化されたブレイクポイントが含まれています。 トレース位置も含まれています。


この行には、まだ検証されていない（つまり、マークされた行にデバッガがまだ達していない）ブレイクポイントが含まれています。 状態は、有効（赤い背景）または無効（白い背景）として表示できます。


この行には無効なブレイクポイントが含まれています（ブレイクポイントが END ステートメントより後ろの行に設定されているなど）。 状態は、有効（赤い背景）または無効（白い背景）として表示できます。

7 変数の変更と監視

このchapterでは、次のトピックについて説明します。

変数の変更

変数の変更方法はいくつかあります。

エディタウィンドウでの変数の変更

エディタウィンドウの現在のカーソル位置に表示されている変数を変更できます。 この操作で表示されるダイアログボックスで、変更する別の変数の名前を入力することもできます。

手順 7.1. カーソル位置の変数を変更するには

1 変数名内にカーソルを置いて、エディタで変数を選択します。

2 コンテキストメニューを表示し、［変数の変更］を選択します。

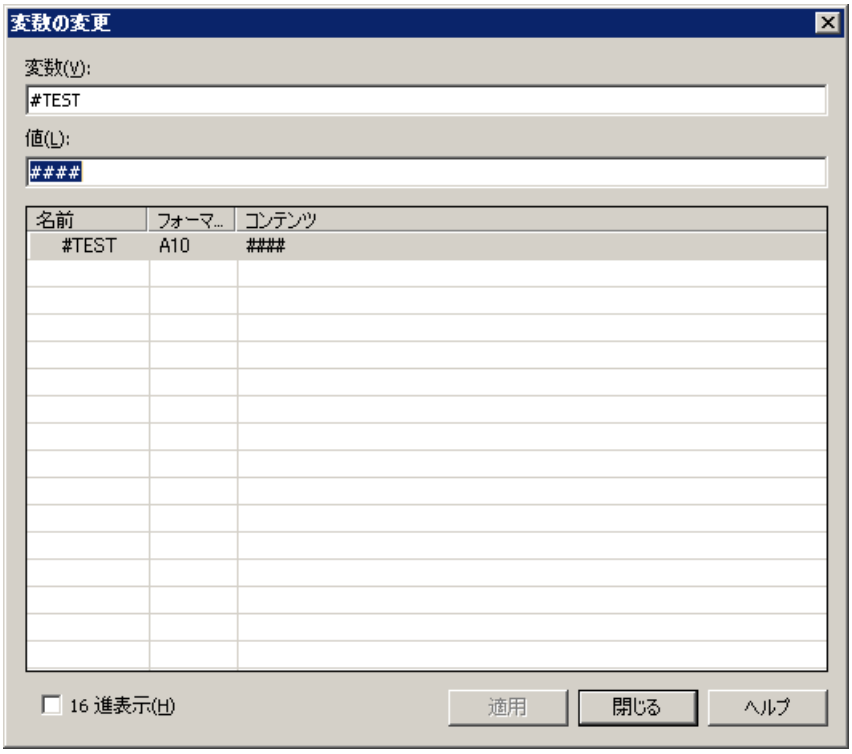
Or:

Shift キーを押したまま F9 キーを押します。

Or:

［デバッグ］ツールバーが表示されたら、次のツールバーボタンを選択します。

［変数の変更］ダイアログボックスに、選択した変数の内容が表示されます。 配列の場合は、ノードはデフォルトで展開されます。



3 変数の内容を変更するには、［値］テキストボックスに新しい値を入力します。

Note: 変更できない変数（変更が許可されないシステム変数など）の場合は、［値］テキストボックスはグレー表示されます。

［変数の変更］ダイアログボックスを以下のように使用することもできます。

- ［変数］テキストボックスに既存の別の変数名を入力すると、この変数の内容が直ちにダイアログボックスに表示され、変更できます。
- 配列の場合は、この配列のオカレンスを以下の手順で変更できます。
  - すべてのオカレンスについて同じ値を定義できます。この場合は、最上位ノードを選択し、新しい値を入力します。
  - 各オカレンスを個別に変更できます。この場合は、目的のオカレンスを選択し、新しい値を入力します。
  - 指定した複数のオカレンスを同時に変更できます。この場合は、［変数］テキストボックスに目的のオカレンスを指定します。例えば、`#MYVAR(1:6)`を`#MYVAR(2:3)`に変更すると、2番目と3番目のオカレンスのみダイアログボックスに表示されます。新しい値を入力すると、これらのオカレンスにのみ適用されます。

変更した変数（とオカレンス）は赤で示されます。

4 ［**16**進表示］チェックボックスをオンにすると、変数の内容が16進形式で表示されます。

5 ［適用］ボタンをクリックします。

［適用］ボタンをクリックすると、変更が直ちに保存されます。 これらはまだ変数ウィンドウには表示されません。

6 ［閉じる］ボタンをクリックして、ダイアログボックスを閉じます。

これで変更が変数ウィンドウに表示されます。

変数ウィンドウでの変数の変更

変数ウィンドウに表示されている変数を変更できます。

さらに展開できるエントリ（ビューなど）を変更することはできません。 そのエントリを展開した後の個々の変数のみ変更できます。

変数ウィンドウでは、エントリが以下のように色分けされます。

- グレー
  変更できない変数（変更が許可されないシステム変数など）はグレー表示されます。
- 赤
  変更した変数は赤で表示されます。

手順 7.2. 変数ウィンドウで変数を変更するには

1 変数ウィンドウで必要なタブを選択します。

2 必要なエントリを選択します。

3 コンテキストメニューを表示し、［変更］を選択します。

［変数の変更］ダイアログボックスに、選択した変数の内容が表示されます。

このダイアログボックスの詳細については、「*エディタウィンドウでの変数の変更*」を参照してください。

ウォッチ変数の追加

特定の変数を監視するには、変数ウィンドウの［ウォッチ］タブにその変数を追加します。

ウォッチ変数の追加方法はいくつかあります。

- **エディタウィンドウでのウォッチ変数の追加**
- **変数ウィンドウでのウォッチ変数の追加**

Note: ウォッチ変数の内容を変更することはできません。

エディタウィンドウでのウォッチ変数の追加

エディタウィンドウの現在のカーソル位置にある変数をウォッチ変数として定義できます。

手順 7.3. ウォッチ変数を変数ウィンドウに追加するには

1 変数名内にカーソルを置いて、エディタで変数を選択します。
2 エディタでコンテキストメニューを表示し、［ウォッチ変数の追加］を選択します。

Or:

Ctrl + Shift + T を押します。

Or:

エディタで変数を選択します。 マウスを使用して、選択した変数をドラッグし、変数ウィンドウの［ウォッチ］タブにドロップします。

変数ウィンドウでのウォッチ変数の追加

変数ウィンドウで先の方にあるタブの変数を、同じウィンドウの［ウォッチ］タブに追加できます。

手順 7.4. 変数をウォッチ変数として定義するには

1 変数ウィンドウで必要な変数を選択します。
2 コンテキストメニューを表示し、［ウォッチ変数の追加］を選択します。

Or:

Ctrl + Shift + T を押します。

変数ウィンドウでの変数の管理

以下では次のトピックについて説明します。

- 最後に変更された変数の表示
- 変数の検索
- 変数の内容の英数字形式表示と 16 進形式表示
- 表示内容の更新
- ウォッチ変数の削除

「*変数ウィンドウでのウォッチポイントの追加*」も参照してください。

最後に変更された変数の表示

デバッグ中に変更された変数が変数ウィンドウに常に表示されるように定義できます。これは、変数ウィンドウに一度に表示できる数より多く変数があるプログラムをデバッグするときに便利です。 変数ウィンドウに現在表示されていない変数がデバッグ中に変更されると、変数ウィンドウの表示がスクロールされ、変更された変数が表示されます。

手順 7.5. この機能のオン／オフを切り替えるには

1 変数ウィンドウのいずれかのタブを選択します。

2 そのタブで選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［最終更新日時の表示］を選択します。

この機能が有効の場合は、このメニューコマンドの横にチェックマークが付きます。

変数の検索

アクティブな変数ウィンドウで、現在選択しているタブ上の変数を検索できます。

Note: 変数ウィンドウ内のノードが展開されていない場合は、その内容は検索の対象になりません。

手順 7.6. 変数を検索するには

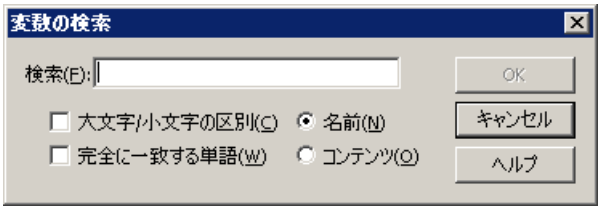
1 変数ウィンドウで必要なタブを選択します。

2 Ctrl キーを押したまま F キーを押します。

Or:

そのタブで選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［検索］を選択します。

［変数の検索］ダイアログボックスが表示されます。



3 検索条件を指定します。

オプション	説明
検索	検索対象文字列。
大文字／小文字の区別	このチェックボックスがオンの場合は、［検索］テキストボックスに入力したとおりに大文字と小文字が一致する文字列のみ検索されます。オフの場合は、大文字と小文字が区別されずに検索されます。
完全に一致する単語	このチェックボックスがオンの場合は、完全に一致する単語のみ検索されます。 オフの場合は、この文字列を含む単語が検索されます。
名前	このオプションを選択した場合は、［検索］テキストボックスの文字列は変数名に適用されます。
コンテンツ	このオプションを選択した場合は、［検索］テキストボックスの文字列は変数の内容に適用されます。 つまり、指定した内容を含む変数が検索されます。

4 ［**OK**］ボタンを選択します。

指定した条件を満たす変数が現在のタブで見つかると、その名前が強調表示されます。

Note: 指定したテキストが見つかったかどうかを通知するメッセージがしばらくの間表示されます。

手順 7.7. 指定した検索条件を満たす次の変数を検索するには

■ F3 キーを押します。

Or:

そのタブで選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［次を検索］を選択します。

変数の内容の英数字形式表示と 16 進形式表示

変数ウィンドウに変数の内容を英数字形式と 16 進形式のどちらで表示するかを定義できます。

手順 7.8. 形式を切り替えるには

1 変数ウィンドウで必要なタブを選択します。

2 そのタブで選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［**16** 進表示］を選択します。

［コンテンツ］列の値が以前に英数字形式で表示されていた場合は、表示が 16 進形式に変わります。逆の場合も同様です。

16 進形式が使用されている場合は、このメニューコマンドの横にチェックマークが付きます。

表示内容の更新

Natural スタジオに変更が加えられた場合は通常、表示内容が自動的に更新されます。デバッガでは、変数の内容が変更されると表示が更新されます。 このように自動的に更新するには、該当するオプションをワークスペースオプションで設定する必要があります。

自動更新がワークスペースオプションで無効になっており、現在選択しているタブで1つ以上の変数の内容を変更した場合、現在の値を見るには表示を手動で更新する必要があります。

ただし、例外があります。ワークスペースオプションの設定とは無関係に、ウォッチ変数は常に自動更新されます。

手順 7.9. 表示内容を手動で更新するには

1 変数ウィンドウで［ウォッチ］タブ以外の任意のタブを選択します。

2 F5 キーを押します。

Or:

そのタブで選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［更新］を選択します。

ウォッチ変数の削除

選択したウォッチ変数またはすべてのウォッチ変数を変数ウィンドウから削除できます。

手順 7.10. ウォッチ変数を削除するには

1 変数ウィンドウの［ウォッチ］タブで目的のウォッチ変数を選択します。

2 コンテキストメニューを表示し、［削除］を選択します。

Or:

Delete キーを押します。

手順 7.11. すべてのウォッチ変数を削除するには

- 変数ウィンドウの［ウォッチ］タブで選択されているエントリがないことを確認し（選択されていると、コンテキストメニューに該当のコマンドが表示されません）、コンテキストメニューを表示し、［すべて削除］を選択します。

8 コールスタックの使用

このchapterでは、次のトピックについて説明します。

コールスタックの概要

コールスタックウィンドウには、現在のデバッグセッション中に呼び出されたオブジェクトが階層形式でリストされます。

現在のオブジェクトは常にリストの先頭に表示されます。 変数ウィンドウには、そのオブジェクトにデフォルトで属するすべての変数が表示されます。 例えば、サブプログラムにステップインすると、そのサブプログラムがリストの先頭に表示され、変数ウィンドウにはそのサブプログラムの変数が自動的に表示されます。

コールスタックウィンドウで特定のオブジェクトに対応するエントリをダブルクリックすると、そのオブジェクト用のエディタウィンドウが前面に移動します。

Notes:

1. エディタウィンドウに表示されるグレーの矢印は、コールスタック階層で前にあるオブジェクトが呼び出された位置を示します。
2. デバッグ対象がコピーコードの場合は、コールスタックにそのコピーコードの追加エントリは含まれません。

他のオブジェクトのソースコードの表示

コールスタックにリストされる各オブジェクトについて、そのエディタウィンドウを前面に移動してソースコードを表示できます。 この操作に使用できるコマンドはいくつかあります。

- ソースコードに移動
  このコマンドを選択した場合は、アクティブなエディタウィンドウ内にあるオブジェクトの変数は変数ウィンドウに表示されません。 前に呼び出されたオブジェクトの変数は引き続き表示されます。
- コールレベルに移動
  このコマンドを選択した場合は、アクティブなエディタウィンドウ内にあるオブジェクトの変数は変数ウィンドウに表示されます。

手順 8.1. 他のオブジェクトのソースコードに移動するには

- コールスタックで、ソースコードを表示するオブジェクトを選択し、コンテキストメニューの［ソースコードに移動］を選択します。

  このオブジェクトのエディタウィンドウが有効になります。

手順 8.2. 別のオブジェクトのソースコードに移動し、そのオブジェクトの変数を表示するには

■ コールスタックで、ソースコードを表示するオブジェクトを選択し、コンテキストメニューの［コールレベルに移動］を選択します。

このオブジェクトのエディタウィンドウが有効になります。 変数ウィンドウの表示内容が変わり、選択したオブジェクトの変数が表示されます。

現在のトレース位置のオブジェクトに戻る

他のオブジェクトのソースコードを表示した後、現在のトレース位置（矢印で示されている）のオブジェクトに戻ることができます。戻ると、そのエディタウィンドウが前面に移動します。

手順 8.3. 現在のトレース位置のオブジェクトに戻るには

■ ［デバッグ］メニューで［トレース位置の表示］を選択します。

Note: 「*エディタウィンドウでのトレース位置*」も参照してください。

現在のトレース位置を含むエディタウィンドウがアクティブになります。 変数ウィンドウの表示内容が変わり、選択したオブジェクトの変数が表示されます。

9 以前のデバッガの使用

開発サーバーに旧バージョンのNaturalがインストールされていると、以前のデバッガが表示されることがあります。詳細については、「*全般的な情報*」を参照してください。

このchapterでは、次のトピックについて説明します。

Natural オブジェクトの準備

Natural デバッガの全機能を使用できるようにするには、以下の Natural プロファイルパラメータを、ダイナミックに指定するか Natural パラメータファイルで指定する必要があります。

`SYMGEN` を "ON" に設定

オブジェクトがカタログ化または格納され、`SYMGEN` が "ON" に設定されている場合は、生成プログラムの一部としてシンボルテーブルが生成されます。 このテーブルには、このオブジェクトに対して有効な変数に関連する情報が含まれているため、`SYMGEN` を指定せずに変数にアクセスすることはできません。ただし、オブジェクトをデバッグすることはできます。

デバッガの開始

デバッガは、格納またはカタログされたNaturalプログラムおよびダイアログについてのみ使用できます。

手順 9.1. デバッガを開始するには

- 以下の Natural コマンドを入力します。

  ```
  DEBUG objectname
  ```

  objectname は、デバッグする Natural オブジェクト名です。

タイトルバーに以下のいずれかが表示されます。

- **[break]**
  タイトルバーに "[break]" と表示されているとき、制御の主体はデバッガの側にあります。
- **[waiting]**
  "[waiting]" がタイトルバーに表示されているときは、現在デバッグ中の Natural アプリケーションに制御があります。

リモートデバッガがWindowsオペレーティングシステムでアクティブになると、タイトルバーに "Debugging remote Natural client (*nodename*::*username*::*process-id*)" と表示されます。*nodename* は Natural が実行されているコンピュータの名前、*username* は Natural ユーザー名、*process-id* は Natural プロセス ID です。

デバッガウィンドウには子ウィンドウがあり、デバッグ対象として指定されたオブジェクトのソースのリストが表示されます。

このオブジェクトソースの他に、制御バーを使用して以下の情報を表示できます。

制御バー	機能
ブレイクポイントとウォッチポイント	この制御バーには、2 つのタブエリアがあります。 1 つにブレイクポイント、もう 1 つにウォッチポイントが表示されます。
変数	この制御バーには、有効な変数とその実際の内容が表示されます。 変数は、カテゴリ［Locals］、［Globals］、［Systems］、［AIVs］、および［Contexts］にわかれて表示されます。
ウォッチ変数	この制御バーには、変数制御バーのカテゴリからユーザーが選択した変数が表示されます。

制御バーの詳細については、このセクションで後述します。

デバッガの終了

デバッガは、［**Exit**］（次を参照）または該当するツールバーボタンを使用して、アプリケーションのどこからでも終了できます。

デバッガは、アプリケーションがエラーなしで終了した場合も終了します。その場合は、トレースカーソルは最後に実行されたソースコード行に移動します。

エラーがあった場合は、該当するソースがソースウィンドウに表示され、トレースカーソルはエラーが発生した行に移動します。 メッセージウィンドウに、適切なエラーメッセージと、デバッグセッションを継続するか終了するかの選択肢が表示されます。 以下の場合は、デバッグセッションを継続すると便利です。

- アプリケーションにエラー処理（エラートランザクションなど）が存在する場合
- デバッグセッションを終了する前に変数の内容を確認したい場合

デバッガを終了すると、ブレイクポイント、ウォッチポイント、およびウォッチ変数の設定は、ウィンドウやツールバーの設定とともに自動的に保存されます。 これらすべての設定は、デバッガを次回呼び出したときに復元されます。

Note: リモート開発において、リモートシステム上のデバッガを終了した場合は、プログラムの実行は続行されますが、プログラム実行に対するデバッグ制御は停止します。

［Exit］コマンド

［**Exit**］は、デバッグセッションを終了し、制御を Natural に戻します。 ［**Exit**］コマンドは、デバッガの 5 つの主な機能それぞれでメインウィンドウの先頭のメニューにあります。

デバッガの操作

デバッガのソースウィンドウなどの主な機能の詳細を説明する前に、このセクションでデバッガの全般的な情報について説明します。

ウィンドウとメニュー

デバッガには、さまざまなウィンドウ、制御バー、ツールバー、およびメニューがあります。

頻繁に使用するメニューコマンドは、対応するツールバーのツールバーボタンからも使用できます。

メニューの代わりに、ツールバーボタンやショートカットキーを使用できます。

Natural 本体とは以下の点が異なります。

- デバッガにコマンド行はありません。
- デバッガの［**Tools**］メニューには以下のオプションがあります。
  - ［**Customize**］：メニューやツールバーの外観を変更したり、頻繁に使用するコマンドのショートカットを定義したりできます。
  - ［**Fonts**］：ソースウィンドウのフォントを変更できます。
  - ［**Warning messages**］：ソースコードが見つからない場合の警告メッセージや記号情報を表示するかどうかを指定できます。 現在表示されているソースコードが、対応する生成プログラムより新しいかどうかを示すメッセージも、この指定の影響を受けます。

ツールバーボタン

ツールバーを使用すると、頻繁に使用するコマンドに簡単にアクセスできます。 コマンドの簡単な説明を表示するには、目的のボタンにマウスポインタを合わせます。 説明が、デバッガのメインウィンドウ下部のステータスバーに表示されます。 現在使用できないコマンドに対応するボタンは無効になります。

ショートカットキー

デバッガコマンドを実行するもう1つの方法は、対応するショートカットをキーボードから入力することです。 デフォルトでは、以下のショートカットが定義されています。

メニュー	ショートカット	機能
File	Ctrl + O	Open
Edit	Ctrl + F	Find
	F3	Find Next
Debug	F4	Close
	F5	Go
	F6	Step Over
	F7	Step In
	Ctrl + F7	Step Out
	Ctrl + F6	Run To Cursor
	Alt + *	Show Trace Position
	F9	Toggle Breakpoint
Variables	Ctrl + M	Modify Variable
	Ctrl + D	Display Variable
	Ctrl + V	Add to Watchvariables
	Ctrl + W	Add to Watchpoints

ウォッチポイントとブレイクポイント

プログラムには、ウォッチポイントとブレイクポイントという2種類のエントリをデバッグ目的で定義できます。 各ウォッチポイントまたはブレイクポイントは、対応する制御バーに表示されます。各ウォッチポイントには、対象となる変数の名前に対応する名前が割り当てられます。

各ウォッチポイントまたはブレイクポイントは、対応するチェックボックスを使用して、デバッグセッション中にいつでも有効または無効にできます。

各ウォッチポイントまたはブレイクポイントにはイベントカウントがあり、デバッグエントリが通過するたびにカウントが増えます。 ただし、デバッグエントリの実行回数は 2 通りの方法で制限できます。

1. ウォッチポイントまたはブレイクポイントが実行される前のスキップ回数を指定できます。その後、イベントカウントが指定されているスキップの数より多くなるまで、デバッグエントリは無視されます。
2. 実行回数の上限を指定して、イベントカウントが指定の実行回数を上回ったら直ちにウォッチポイントまたはブレイクポイントが無視されるようにすることができます。

デバッグセッションの再スタート

デバッグセッションを再スタートすると、デバッガの位置がアプリケーションの先頭に戻り、現在のすべての設定（ウォッチポイントやブレイクポイントなど）は保持され、すべてのカウンタと呼び出し履歴が初期化されます。 このため、デバッグに関連する設定を指定せずにアプリケーションを再実行する場合は、デバッグセッションの再起動が役に立ちます。 デバッグセッションは、［Restart］コマンドまたは該当するツールバーアイコンをクリックして、アプリケーションのどこからでも再スタートできます。 ［**Restart**］コマンドは、［Debug］メニューにあります。

Note: デバッグセッションをリモート環境で実行している場合は、［**Restart**］コマンドを使用できません。DCOM または RPC サーバーをデバッグしている場合に［**Restart**］コマンドを使用すると、呼び出されているメソッドまたはサブプログラムが再スタートします。

デバッガソースウィンドウ

呼び出されたデバッガは、指定されたNaturalオブジェクトの制御を受け取り、対応するソースをソースウィンドウに表示します。 表示するソースがない場合は、ウィンドウには何も表示されません。 トレースカーソルは、最初に実行可能なソースコード行に移動します。

ユーザーが新しいオブジェクトを開いた場合、または現在アクティブな別のオブジェクトでウォッチポイントまたはブレイクポイントがヒットした場合は、新しいソースウィンドウが開き、新しいオブジェクトのソースが表示されます。

以下では次のトピックについて説明します。

- ［Debug］メニュー
- ［Variables］メニュー
- ダイアログボックス
- 変数の選択
- ソースウィンドウでのテキストのマーク
- 表示（Display）
- 変更（Modify）
- クイックウォッチ（Quick Watch）
- ウォッチ（Add Watch）
- ウォッチポイントの追加（Add Watchpoint）

［Debug］メニュー

以下の［**Debug**］メニューコマンドは、ソースウィンドウに関連して使用できます。

ステップイン（**Step Into**）
：［**Step Into**］コマンドを選択すると、次のプログラムステップが実行され、トレースカーソルは対応するソースコード行に移動します。

このソースコード行が呼び出された場合、または他の Natural オブジェクトが含まれていた場合は、デバッガはこのオブジェクトにステップインします。

ステップオーバー（**Step Over**）
：［**Step Over**］コマンドを選択すると、次のプログラムステップが実行され、トレースカーソルは対応するソースコード行に移動します。 デバッガは、呼び出されたか含まれている Natural オブジェクトをステップオーバーしますが、このオブジェクトにウォッチポイントまたはブレイクポイントが含まれていると停止します。

ステップアウト（**Step Out**）
：［**Step Out**］コマンドを選択すると、デバッガは前のプログラムレベルに戻りますが、前のレベルに達する前にウォッチポイントやブレイクポイントがあった場合は停止します。

アニメーションステップイン（**Animated Step Into**）
：［**Animated Step Into**］コマンドを選択すると、そのプログラムの終わりまで自動的にステップ実行されます。 Natural オブジェクトが呼び出されたか含まれていた場合は、デバッガはステップインします。

アニメーションステップオーバー（**Animated Step Over**）
：［**Animated Step Over**］コマンドを選択すると、そのプログラムの終わりまで自動的にステップ実行されます。 デバッガは、呼び出されたか含まれている Natural オブジェクトをステップオーバーします。ウォッチポイントまたはブレイクポイントが設定されている場合は、対応するステートメント行にジャンプし、アニメーションを続行します。

実行（**Go**）
：［**Go**］コマンドを選択すると、プログラムは次の有効なウォッチポイントまたはブレイクポイントまで実行され、トレースカーソルは対応するソースコード行に移動します。

次のイベントまで実行（**Go Until Next Event**）
：［**Go Until Next Event**］コマンドを選択すると、非イベントドリブンアプリケーションの［**Go**］コマンドと同じ操作が実行されます。 ただし、イベントドリブンアプリケーションでは、オブジェクトは次のイベントがアプリケーションに送信されるまで実行されます。次のイベントが送信される前に有効なウォッチポイントまたはブレイクポイントがあった場合は停止します。

カーソル行まで実行（**Run to Cursor**）
：［**Run to Cursor**］コマンドを選択すると、プログラムは現在のカーソル位置のソース行に達するまで実行されます。

トレース位置の表示（**Show Trace Position**）

［**Show Trace Position**］コマンドを選択すると、現在のトレースカーソルが表示されます。

ブレイクポイント **ON/OFF**（**Toggle Breakpoint**）

［**Toggle Breakpoint**］コマンドを選択すると、現在のトレース位置のブレイクポイントがブレイクポイント制御バーに追加されます。このカーソル位置にブレイクポイントがすでに存在している場合は、そのブレイクポイントがブレイクポイント制御バーから削除されます。

コール（**Calls**）

［**Calls**］サブメニューは、コピーコードやインラインサブルーチンを含む、最近呼び出された Natural オブジェクトのリスト（履歴）を表示します。 最高で 20 のオブジェクトが表示されます。直前に呼び出されたオブジェクトはリストの先頭に表示されます。

オブジェクトリストには以下の情報が表示されます。

- コピーコードとインラインサブルーチンを除く、呼び出されたオブジェクトのプログラムレベル。
- コピーコードとインラインサブルーチンを含む、呼び出されたオブジェクトのプログラムレベル。
- 呼び出されたオブジェクトの名前。
- 呼び出されたオブジェクトのタイプ。
- 処理されるイベントハンドラのイベントおよび制御ハンドル（イベントドリブンアプリケーションの場合のみ）。

デバッガのメインウィンドウ下部のステータスバーには、呼び出されたオブジェクトに関する以下の追加情報が表示されます。

- 呼び出し側オブジェクトの名前。

  呼び出されたオブジェクトがアプリケーション起動プログラムの場合、または Natural スタックから起動されたプログラム（エラートランザクションプログラムや、アプリケーション内で `RUN` ステートメントによって起動されたプログラムを含む）の場合は、"Natural" は呼び出し側オブジェクトとして表示されます。

- オブジェクトの呼び出しが行われたソースコード行。

  オブジェクトをリストから選択した場合は、"Natural" を除いて、呼び出し側プログラムのソースはソースウィンドウの中央に表示され、呼び出しが発生した行の先頭にカーソルが移動します。

［Variables］メニュー

［**Variables**］メニューは以下の操作に使用します。

- 選択した変数の内容を表示する。
- 選択した変数の内容を変更する。
- 現在のトレース位置で変数の内容を簡単に監視する。
- 変数をウォッチ変数制御バーに追加する。
- 変数をウォッチポイント制御バーに追加する。

ダイアログボックス

［**Display**］、［**Modify**］、［**Add Watch**］、または［**Add Watchpoint**］コマンドを選択すると、ダイアログボックスが開き、現在のデバッグコンテキストで有効なすべてのローカル変数、グローバル変数、AIV、またはシステム変数のリストが表示されます。 このダイアログボックスには以下のコントロールがあります。

- ［**Variable**］テキストボックス：現在選択されている変数が表示されます。
- ［**Line Reference**］または［**Context ID**］ボックス：［**Variable**］テキストボックスに現在含まれている変数またはコンテキスト変数のソースコード行番号が表示されます。

  ［**Line Reference**］ボックスは、変数選択を明確にするために行参照が必要な場合のみ表示されます。 以下の場合が該当します。

  - 変数がマップに属している。この場合は、マップエディタによって生成された対応する `RULEVAR` 構文要素のソースコード行番号が表示されます。
  - 変数がデータベース変数である（レポーティングモードのみ）か `*ISN`、`*COUNTER`、`*NUMBER` のいずれかの変数である。この場合は、対応するデータベースループ／アクセスステートメントのソースコード行番号が表示されます。
  - 変数がレポーティングモードで定義されているが、`DEFINE DATA` ステートメントが使用されていない。

  ［**Context ID**］ボックスは、変数がコンテキスト変数の場合のみ表示されます。この場合は、ボックスに「ctx-Id」（コンテキスト ID）と表示されます。
- ［**History List**］リストボックス（［**Display**］コマンドのみ）：最近選択された変数（最高 20 個）が先入れ先出し方式で表示されます。

  履歴リストを使用すると、選択済みの変数を簡単に見つけることができます。 履歴リストでは、変数リストの場合と同じように変数を選択できます。
- ［**Variable**］リストボックス：対応する変数リストが表示されます。

変数の選択

ダイアログボックスの変数リストで選択した変数は、対応する［**Variable**］テキストボックスに表示されます。

ダイアログボックスの変数リストで変数を選択すると、別のダイアログボックスが表示されます（［**Watch**］コマンドの場合を除く）。

配列または変数グループを選択したときの動作は以下のとおりです。

- 配列またはグループの各要素は別のダイアログボックスに表示されます（表示）。
- 別のダイアログボックスに表示された配列は変更できます。グループは変更できません（変更）。
- 対応するエラーメッセージが表示されます（ウォッチポイント）。

変数は、ソースウィンドウで直接マークし、［**Display**］、［**Modify**］、［**Add Watch**］、または［**Add Watchpoint**］コマンドを選択することでも、選択できます。これにより、対応するダイアログボックスの［**Variable**］テキストボックスに、マークしたソースコード部分がそのまま表示され、そこで変更できます。

ソースウィンドウでのテキストのマーク

ソースウィンドウでは、マウスまたはキーボードを使用して、変数や文字列を選択対象としてマークできます。

マウスを使用してテキストをマークする場合は、マウスポインタを選択対象の先頭の文字に置き、ポインタを対象の最後の文字までドラッグし、マウスボタンを放します。 選択をキャンセルするには、ドキュメント内の任意の場所をクリックします。

キーボードを使用してテキストをマークする場合は、カーソル移動キーを使用します。矢印キーを使用してカーソルを先頭の文字に置き、Shift キーを押しながら以下のキーを使用してテキストを選択します。

- カーソル位置より左をマークするには、左矢印キーを使用します。
- カーソル位置より右をマークするには、右矢印キーを使用します。
- ソースコード行の末尾までマークするには、End キーを使用します。
- ソースコード行の先頭までマークするには、Home キーを使用します。

表示（Display）

このコマンドを使用すると、変数をダイアログボックスのリストから選択し、現在の内容を別のダイアログボックスの［**Display Variable**］に表示できます。このダイアログボックスでは、変数値の英数字表現とバイナリ表現を切り替えることができます。

配列、ハンドル変数、または変数グループを選択した場合は、各要素とその値は別のダイアログボックスに表示されます。 配列の場合は、変数インデックス式はすべて評価されます。

要素リストは、［**Expand/Contract**］ボタンを使用して展開または収縮できます。 リストの配列、グループ、またはダイアログエレメントの数が表示の上限を超えた場合は、"More" 行が表示されます。これを使用して、隠れているオブジェクトを表示できます。また、［**Expand**］コマンドを使用することもできます。

変数、変数の配列、または変数のグループは、ソースウィンドウで左マウスボタンを使用して直接選択することでも、［**Display Variable**］ダイアログボックスに表示できます。

変更（Modify）

このコマンドを使用すると、変数をダイアログボックスのリストから選択し、その現在の値を別のダイアログボックスの［**Modify Variable**］に表示できます。このダイアログボックスでは、値を変更できます。

システム変数を変更する場合は、最初のダイアログボックスには変更可能なシステム変数のみ表示されます。

配列を変更する場合は、別のダイアログボックスには名前のみが表示され、値は表示されません。 入力した値は、すべての配列要素に対して有効になります。

変数のグループを選択して変更することはできません。

クイックウォッチ（Quick Watch）

このコマンドを選択すると、ダイアログボックスが開き、現在のカーソル位置にある変数の内容が表示されます。

ウォッチ（Add Watch）

このコマンドを使用すると、ダイアログボックスのリストから、変数、変数の配列、または変数のグループを選択して、ウォッチ変数制御バーに追加できます。

ウォッチポイントの追加（Add Watchpoint）

このコマンドを使用すると、ダイアログボックスのリストから単一の変数とグループまたは配列の各要素を選択し、ウォッチポイントを別のダイアログボックスの［**Set Watchpoint**］で定義できます。変数の配列やグループを選択することはできません。

この［**Set Watchpoint**］ダイアログボックスには、ウォッチポイント名（選択した変数の名前に対応）、行参照（該当する場合）、および対応するNaturalオブジェクトおよびライブラリの名前が表示されます。

Note: システム変数では、対応するウォッチポイントが特定のライブラリおよびオブジェクトに添付されません。このため、オブジェクトとライブラリの名前は常にSYSTEMです。

ウォッチポイントを定義するには、該当するダイアログボックスで以下の項目を指定します。

- ウォッチポイントの状態
- ウォッチポイントが有効になる条件（オプション）
- ウォッチポイントを実行するまでのスキップ回数
- ウォッチポイントの実行回数の上限

［File］メニュー

以下の［**File**］メニューコマンドは、ソースウィンドウに関連して使用できます。

開く（**Open**）
　［**Open**］コマンドを使用すると、ソースウィンドウに追加ロードするソースプログラムを指定できます。［**Open Source**］ダイアログボックスが開き、プログラム名を指定できます。また、プログラムが現在のライブラリに含まれていない場合（デフォルト）は適切なライブラリ名も指定できます。

　［**Open Source**］ダイアログボックスに指定する文字列は、ソースウィンドウでマークし、［**Open**］コマンドを選択することでも選択できます。

閉じる（**Close**）
　［**Close**］コマンドは、現在アクティブなソースウィンドウを閉じます。閉じようとしているソースウィンドウにトレースバーが存在する場合は、ウィンドウはアイコン化されます。

終了（**Exit**）
　［**Exit**］コマンドは、デバッガを終了し、現在のプログラム実行を終了します。

［Edit］メニュー

以下の［Edit］メニューコマンドは、ソースウィンドウに関連して使用できます。

検索（**Find**）
: ［**Find**］コマンドを使用すると、アクティブなウィンドウ内を上方向または下方向に検索して、指定した単語や文字列を見つけることができます。

  ［**Find**］ダイアログボックスが開くので、検索するテキストを［**Find**］テキストボックスに入力します。また、［**Match Upper/Lower Case**］オプションや［**Whole Words Only**］オプションをオンまたはオフにすることもできます。

  指定したテキストが見つかった場合は、最初の出現が強調表示（選択）されます。見つからなかった場合は、メッセージが表示されます。

  ［**Match Upper/Lower Case**］オプションでは、大文字と小文字を一致させて検索するか（オン）、または大文字と小文字を区別せずに検索するか（オフ）を指定します。

  ［**Whole Words Only**］オプションでは、完全に一致する単語のみ検索し、文字列の一部である場合を対象外にするか（オン）、または完全に一致するか文字列の一部かに関係なく、指定したテキストの出現をすべて検索するか（オフ）を指定します。

  検索方向は、［**Up**］ボタンを選択するとテキストの先頭に向かって上方向に検索され、［**Down**］ボタンを選択するとテキストの末尾に向かって下方向に検索されます。デフォルトは［**Down**］です。

  テキストの途中から検索を開始し、指定したテキストが見つからなかった場合は、ダイアログボックスが表示されます。［**Yes**］を選択すると、テキストの先頭（または末尾）から検索が続行されます。［**No**］を選択すると検索が終了します。

  ［**Find**］テキストボックスに指定する文字列をソースウィンドウで直接マークし、［**Find**］コマンドを選択することでも、文字列を選択できます。

次を検索（**Find Next**）
: このコマンドを使用すると、直前の検索操作を繰り返して、［**Find**］コマンドで指定したテキストの次の出現を検索できます。

ウォッチ変数制御バー

ウォッチ変数制御バーの主な目的は、前に選択した変数を表示し、その内容を注意して継続的に観察できるようにすることです。

コンテキストメニューがあり、制御バー全体を対象とするコマンドと各ウォッチ変数を対象とするコマンドのいずれかを使用できます。

コンテキストメニューを開くには、制御バーキャプションまたは特定のウォッチ変数を右クリックします。

変数制御バー

変数制御バーには、プログラム実行の現在の状態で使用できるすべての変数が表示されます。どの変数もいずれかのカテゴリに属します。 カテゴリは、［**Locals**］、［**Globals**］、［**Systems**］、［**AIVs**］、および［**Contexts**］です。 カテゴリを切り替えるには、制御バーの下部で該当するタブを選択します。 内容を変更するには、目的の変数の内容フィールドを選択します。 一部のシステム変数は読み取り専用で、変更できません。

変数制御バーにはコンテキストメニューがあり、制御バー全体を対象とするコマンドと各変数を対象とするコマンドのいずれかを使用できます。

コンテキストメニューを開くには、制御バーキャプションまたは特定の変数を右クリックします。

ウォッチポイントとブレイクポイント制御バー

ウォッチポイントとブレイクポイント制御バーは、ウォッチポイントやブレイクポイントの追加および管理に使用します。 ウォッチポイントとブレイクポイントを切り替えるには、制御バーの下部で該当するタブを選択します。

ウォッチポイント

ウォッチポイントを使用すると、エラーを含むオブジェクトによる Natural 変数の 「無効な」変更を迅速に検出できます。

デフォルトでは、ウォッチポイントを使用して、変数の内容が変更されたときに Natural オブジェクトの実行を中断するようデバッガに指示します。 ただし、ウォッチポイントの設定時にウォッチポイント演算子を使用して変数に特定の値を指定すると、条件が設定され、その条件が真の場合のみウォッチポイントが有効になります。

変数が変更されたと見なされるのは、変数の現在の値が、前回ウォッチポイントがトリガされたときに記録された値または初期値と異なる場合です。

ウォッチポイントを一時的に無効にするには、対応するウォッチポイントエントリのチェックボックスをオフにします。

制御バーのウォッチポイントタブにはコンテキストメニューがあり、タブ全体を対象とするコマンドと各ウォッチポイントを対象とするコマンドのいずれかを使用できます。

コンテキストメニューを開くには、タブのキャプションまたは特定のウォッチポイントを右クリックします。

ウォッチポイントの追加

新しいウォッチポイントを追加するには、［Variables］メニューの［**Add Watchpoint**］コマンドを選択するか、［Watchvariables］タブのコンテキストメニューの［**Add**］コマンドを選択します。

［**Add Watchpoint**］ダイアログボックスが開き、使用可能な変数のリストから、変数、配列、または個別のグループ要素を選択できます。［**OK**］ボタンをクリックしてこのダイアログボックスを閉じると、［**Set Watchpoint**］ダイアログボックスが開き、このウォッチポイントの条件を指定できます。

Note: システム変数では、対応するウォッチポイントが特定のライブラリおよびオブジェクトに添付されません。このため、オブジェクトとライブラリの名前は常に SYSTEM です。

［Set Watchpoint］ダイアログボックス

［**Set Watchpoint**］ダイアログボックスには、ウォッチポイント名（選択した変数の名前に対応）、行参照／コンテキスト ID（該当する場合）、および対応する Natural オブジェクトおよびライブラリの名前が表示されます。

Note: システム変数では、対応するウォッチポイントが特定のライブラリおよびオブジェクトに添付されません。このため、オブジェクトとライブラリの名前は常に SYSTEM です。

ウォッチポイントを定義するには、該当するダイアログボックスで以下の項目を指定します。

- 設定するウォッチポイントの状態。有効な状態は "active"（デフォルト）または "pending" です。
- ウォッチポイントが有効になる条件（オプション）。

適切な値とウォッチポイント演算子を指定できます。演算子と値（条件）の指定がない場合は、デフォルトの設定（MOD）が適用されます（ウォッチポイント演算子については、以下を参照）。

- プログラムが所定の回数実行されるまでウォッチポイントが実行されないようにする場合の、ウォッチポイント実行前のスキップ回数。デフォルトは 0 です。
- ウォッチポイントの実行回数の上限。デフォルトは 0 です。

ウォッチポイントは、［**OK**］ボタンをクリックするか Enter キーを押すと設定されます。
［**Cancel**］ボタンをクリックするか Esc キーを押すと、ウォッチポイントは設定されません。

指定したウォッチポイントは、明示的に削除するまで残ります。

ウォッチポイント演算子

ウォッチポイント演算子の設定にはオプションボタンを使用します。使用できるウォッチポイント演算子は以下のとおりです。

演算子	意味	説明
MOD	変更	ウォッチポイントは、変数の内容が変更されるたびに有効になります。 デフォルトです。
LT	より小さい	ウォッチポイントは、変数の現在値が指定値より小さい場合のみ有効になります。
LE	以下	ウォッチポイントは、変数の現在値が指定値以下の場合のみ有効になります。
GT	より大きい	ウォッチポイントは、変数の現在値が指定値より大きい場合のみ有効になります。
GE	以上	ウォッチポイントは、変数の現在値が指定値以上の場合のみ有効になります。
EQ	等しい	ウォッチポイントは、変数の現在値が指定値と等しい場合のみ有効になります。
NE	等しくない	ウォッチポイントは、変数の現在値が指定値と等しくない場合のみ有効になります。

ブレイクポイント

ブレイクポイントとは、Naturalオブジェクトの実行中に制御がユーザーに返されるポイントです。

ブレイクポイントを一時的に無効にするには、対応するブレイクポイントエントリのチェックボックスをオフにします。

制御バーのブレイクポイントタブにはコンテキストメニューがあり、タブ全体を対象とするコマンドと各ブレイクポイントを対象とするコマンドのいずれかを使用できます。

コンテキストメニューを開くには、タブのキャプションまたは特定のブレイクポイントを右クリックします。

ブレイクポイントの追加

新しいブレイクポイントを定義するには、［**Add**］コマンドを使用します。［**Add Breakpoint**］ダイアログボックスが開くので、以下の項目を該当するボックスに入力して、ブレイクポイントを定義します。

- 設定するブレイクポイントの状態。有効な状態は "active"（デフォルト）または "pending" です。
- ブレイクポイントのあるNaturalオブジェクトの名前。デフォルトのオブジェクト名は、ソースウィンドウに現在表示されているオブジェクトの名前です。
- ブレイクポイントが設定されているオブジェクトを含むNaturalライブラリの名前。デフォルトのライブラリ名は、ソースウィンドウに現在表示されているオブジェクトを含むライブラリの名前です。
- ブレイクポイントが実行される、オブジェクトのソースコードの行番号。

［**Begin**］では、指定したオブジェクトの最初の実行可能コード行にブレイクポイントを設定します。［**End**］では、指定したオブジェクトの最後の実行可能コード行にブレイクポイントを設定します。

- プログラムが所定の回数実行されるまでブレイクポイントが実行されないようにする場合の、ブレイクポイント実行前のスキップ回数。デフォルトは 0 です。
- ブレイクポイントの実行回数の上限。デフォルトは 0 です。

ブレイクポイントは、［**OK**］ボタンをクリックするか Enter キーを押すと設定されます。［**Cancel**］ボタンをクリックするか Esc キーを押すと、ブレイクポイントは設定されません。

ブレイクポイントは、適切なステートメント行を右ダブルクリックすることによって、ソースウィンドウに現在表示されているプログラムに直接設定することもできます。 この方法では、ブレイクポイントはすべてのデフォルト値および対応するソースコード行番号で定義されます。このブレイクポイントは、対応する機能を使用して表示または変更できます。

ブレイクポイントはコメント行に設定することはできません。また、1つのステートメントが複数行にわたる場合、先頭行以外にブレイクポイントを設定することはできません。

定義したブレイクポイントは、明示的に削除するまで残ります。

索引

さ

し

せ

そ

た

つ

て

と

ひ

ふ

へ

む

め

よ

り

ろ