OPTIMIZE

MERCURY QUICKTEST PROFESSIONAL™

VERSION 9.0

上 級 機 能 ユ ー ザ ー ズ ・ ガ イ ド

MERCURY™

BUSINESS TECHNOLOGY OPTIMIZATION

Mercury QuickTest Professional

上級機能ユーザーズ・ガイド

Version 9.0

MERCURY™

Mercury QuickTest Professional 上級機能ユーザーズ・ガイド，Version 9.0

Mercury Interactive Corporation
379 North Whisman Road
Mountain View, CA 94043
Tel: (650) 603-5200
Toll Free: (800) TEST-911
Customer Support: (877) TEST-HLP
Fax: (650) 603-5300

本書に関するご意見，ご要望は documentation@mercury.com まで電子メールにてお送りください。

QTPAFUG9.0JP/01

マルチ・ボリューム版の各章の概要

QuickTest Professional のユーザ・マニュアルは 2 冊に分かれています。

- 『**QuickTest Professional 基本機能ユーザーズ・ガイド**』では，QuickTest を紹介するとともに，日常的なテスト実行で使用される基本機能について説明します。
- 『**QuickTest Professional 上級機能ユーザーズ・ガイド**』では，アプリケーションのテスト時に使用できる上級機能について説明します。また，その他の Mercury 製品を使った作業の方法についても説明します。

個々のガイドに含まれる各章の概要は，次のとおりです。

QuickTest Professional 基本機能ユーザーズ・ガイド

QuickTest Professional 上級機能ユーザーズ・ガイド

目次

第 2 部：オブジェクト・リポジトリの管理と結合

第 3 部： 高度な設定

QuickTest へようこそ

QuickTest Professional へようこそ。QuickTest は Mercury のキーワード駆動型テスト・ソリューションです。QuickTest には，テストを素早く作成し，実行するために必要な機能がすべて含まれています。

注 :『QuickTest Professional 基本機能ユーザーズ・ガイド』および**『QuickTest Professional 上級機能ユーザーズ・ガイド』**は，PDF 形式でのみ個別の文書として提供されています。コンテキスト・センシティブ・ヘルプでは，これら 2 つの文書の情報が 1 つにまとめられています。

本書の使い方

本書では，QuickTest を使ってアプリケーションをテストする方法について説明します。テストの作成，デバッグ，実行の方法と，テスト・プロセス中に検出された不具合の報告の方法を順を追って説明します。本書は，次の部で構成されています。

第 1 部　高度なテスト機能を使用した作業

アクション，仮想オブジェクト，回復シナリオの使用方法，オブジェクト識別の設定方法，およびスマート認識定義の作成方法について説明します。QuickTest の中でユーザ定義関数および関数ライブラリを使用する方法についても説明します。また，より強力なスクリプトを作成するためのプログラミング技術や，エキスパート・ビューを使用してテストを強化する方法について説明します。QuickTest 操作を自動化する方法についても説明します。

第 2 部　オブジェクト・リポジトリの管理と結合

QuickTest がアプリケーションのオブジェクトをどのように識別するかについて，またオブジェクト・リポジトリの管理と結合の方法について説明します。

第 3 部　高度な設定

Web イベント記録の設定方法，エキスパート・ビューのカスタマイズ方法，および実行セッション中のテスト・オプションの設定方法について説明します。

第 4 部　その他の Mercury 製品を使用した作業

テストを実行し，Mercury の企業向け Microsoft Windows アプリケーション機能テスト・ツールである WinRunner でコンパイルされたモジュールの関数を呼び出す方法について説明します。また，この部では，QuickTest を Business Process Testing と組み合わせて使用する方法や，QuickTest を Mercury の集中品質管理ソリューションである Mercury Quality Center（以前の TestDirector）と連携させる方法についても説明します。さらに，この部では，Mercury のパフォーマンス・テスト製品やアプリケーション管理製品で使用できるように QuickTest テストを設計するための注意事項について詳しく説明します。

第 5 部　付録

よく尋ねられる質問に関する情報を提供します。

製品マニュアル

QuickTest Professional には，この『上級機能ユーザーズ・ガイド』のほかに，次のマニュアルが付属しています。

『**QuickTest Professional インストール・ガイド**』では，QuickTest Professional のインストール方法について説明します。

『**QuickTest Professional の新情報**』（［**ヘルプ**］＞［**新情報**］メニューから利用できます）では，QuickTest Professional の最新の機能，機能強化，および最新バージョンでサポートされる環境について説明します。

『**QuickTest Professional 基本機能ユーザーズ・ガイド**』では，QuickTest Professional を使って Web サイトまたはアプリケーションをテストする方法を手順ごとに説明します。

『**QuickTest Professional for Business Process Testing ユーザーズ・ガイド**』では，QuickTest Professional を使ってビジネス・プロセス・テストで使用する資産を作成および管理するための手順を順を追って説明します。

『**QuickTest Professional チュートリアル**』では，QuickTest の基本的なスキルと，アプリケーションのテストを設計する方法を習得できます。

『**Readme**』（［**スタート**］メニューの QuickTest Professional プログラム・グループから利用できます）では，QuickTest Professional についての最新のニュースと情報を参照できます。

『**Printer-Friendly Documentation**』（［**ヘルプ**］＞［**印刷用ドキュメント**］メニューから利用できます）では，すべてのマニュアルが Adobe PDF で表示されます。オンライン文書は，Acrobat Reader を使って読んだり印刷したりできます。Acrobat Reader は，Adobe の Web サイト（http://www.adobe.co.jp）からダウンロードできます。

『**QuickTest Professional コンテキスト・センシティブ・ヘルプ**』（ダイアログ・ボックスとウィンドウから利用できます）では，QuickTest のダイアログ・ボックスとウィンドウについて説明します。

『**QuickTest Professional オブジェクト・モデル・リファレンス**』（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］メニューから利用できます）では，QuickTest Professional テスト・オブジェクトについての説明，各オブジェクトに関連するメソッドおよびプロパティの一覧，構文情報とメソッドの例を参照できます。

『QuickTest Professional オートメーション・オブジェクト・モデル・リファレンス』（［スタート］メニューの QuickTest Professional プログラム・グループの［**Documentation**］からか，［**ヘルプ**］＞［**QuickTest オートメーション オブジェクト モデル リファレンス**］から利用できます）では，構文，解説情報，オートメーション・オブジェクト，メソッドおよびプロパティの例を参照できます。また，QuickTest の自動スクリプトを記述する際の詳しい概要も含まれます。オートメーション・オブジェクト・モデルは，QuickTest のほぼすべての機能を制御することを可能にするオブジェクト，メソッド，プロパティを提供することによって，テスト管理の自動化を支援します。

『VBScript リファレンス』（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］メニューから利用できます）には，VBScript，Script ランタイム，および Windows Script Host などを含む Microsoft VBScript 文書が含まれます。

追加のオンライン・リソース

QuickTest Professional には，次の追加のオンライン・リソースがあります。

本書に含まれる多くの例は，サンプルの **Mercury Tours** Web サイト（[スタート] メニューの QuickTest Professional プログラム・グループか，QuickTest Professional の［記録と実行環境設定］ダイアログ・ボックスから利用できます）および，サンプルの **Mercury Tours** Windows アプリケーション（[スタート] メニューの QuickTest Professional プログラム・グループから利用できます）に基づいています。Mercury Tours Web サイトの URL は，http://newtours.mercury.com です。

「**Knowledge Base**」（［**ヘルプ**］＞［**ナレッジ ベース**］から利用できます）では，普段お使いの Web ブラウザで Mercury のカスタマー・サポートのナレッジ・ベースを開くことができます。ナレッジ・ベースでは，Mercury およびユーザの方々が投稿したナレッジ・ベース記事を参照したり，自分の記事を追加したりできます。この Web サイトの URL は，http://support.mercury.com/cgi-bin/portal/CSO/kbBrowse.jsp です。

「**カスタマー・サポート Web サイト**」（［**ヘルプ**］＞［**カスタマ サポート Web サイト**］メニューから利用できます）では，普段お使いの Web ブラウザで Mercury のカスタマー・サポート Web サイトが開きます。このサイトでは，Mercury の最新情報や製品に関する情報をご覧になれます。この Web サイトの URL は http://www.mercury.com/jp/services/support/ です。

「**製品に関するご意見・ご要望**」（［**ヘルプ**］＞［**フィードバックの送信**］で利用できます）を使用すると，QuickTest Professional に関するオンライン・フィードバックが製品チームに送信されます。

「**Mercury ホーム・ページ**」（［**ヘルプ**］＞［**Mercury ホーム ページ**］メニューから使用できます）では，普段お使いの Web ブラウザで Mercury のホーム・ページが開きます。このサイトでは，Mercury の最新情報や製品に関する情報をご覧になれます。新しいソフトウェアのリリース，セミナー，展示会，カスタマー・サポート，教育サービスなどに関する情報をご覧いただけます。Mercury の Web サイトの URL は，http://www.mercury.com/jp/ です。

「**Mercury Best Practices**」には，ワールドクラスの IT 環境を計画，構築，配備および管理するためのガイドラインが含まれています。Mercury は，Process Best Practices，Product Best Practices および People Best Practices の 3 種類のベスト・プラクティスを提供しています。Mercury ソフトウェアのライセンスをお持ちのお客様は，カスタマー・サポート・サイト http://support.mercury.com/ から入手可能な Mercury Best Practices を利用できます。

文書の更新

Mercury では，製品マニュアルを新しい情報で絶えず更新しています。このマニュアルの最新版は Mercury のカスタマー・サポート Web サイト（http://support.mercury.com/）からダウンロードできます。

更新された文書をダウンロードするには，次の手順を実行します。

1 カスタマー・サポート Web サイトで，［**Documentation**］リンクをクリックします。

2 ［**Please Select Product**］で［**QuickTest Professional**］を選択します。

［**QuickTest Professional**］がリストに表示されていない場合は，顧客プロファイルに追加する必要があります。［**My Account**］をクリックしてプロファイルを更新します。

3 ［**Retrieve**］をクリックします。文書のページが開き，現在のリリースと以前のリリースに関する使用可能な文書がリストされます。文書が最近更新された場合，文書名の隣に「**Updated**」のマークが表示されます。

4 文書のリンクをクリックして，文書をダウンロードします。

本書の表記規則

本書では，次の表記規則に従います。

1, 2, 3	太字の数字は，操作手順を示します。
>	大なり記号はメニュー・レベルを区切ります（例：［**ファイル**］＞［**開く**］）。
［**太字**］	インタフェース要素の名前は，全角の大括弧に**太字**で示します（例：［**実行**］ボタンをクリックします）。
太字	**太字**のテキストは，メソッド名または関数名を示します。また，メソッドまたは関数の引数およびドキュメント名を示します。新しく使用される用語もこの字体で示します。
＜＞	山括弧は，ユーザの環境しだいで変わる可能性のあるファイル・パスや URL アドレスの一部を囲みます（例：**＜製品のインストール・パス＞¥bin**）。
Arial	使用例やユーザがそのまま入力する必要がある文字列は，Arial という書体で示します。
Arial bold	**Arial bold** のフォントはそのまま入力する必要のある構文記述のテキストに使用します。
SMALL CAPS	SMALL CAPS のフォントは，キーボードのキーを示します。
...	構文内の省略記号は，同じ形式で項目をさらに組み入れることができることを意味しますプログラミング例に含まれる場合は，何行かが意図的に省略されていることを示します。
[ ]	省略可能な引数は，半角の大括弧で囲んで示します。
|	垂直バー（パイプ記号）は，バーで区切られているオプションのいずれかを指定しなければならないことを示します。

第 1 部

高度なテスト機能を使用した作業

第 1 章

高度なアクション機能を使用した作業

テストを複数のアクションに分割して，アプリケーションまたは Web サイトのテストのプロセスを合理化できます。本章では，テストでのアクションの高度な操作について説明します。アクション関連の基本的な機能の使用方法については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 17 章「アクションを使った作業」で説明します。

本章では，次の項目について説明します。

- 高度なアクション機能を使用した作業について
- 既存のアクションへの呼び出しの挿入
- アクション・パラメータの設定
- アクション・パラメータの使用
- アクションの呼び出しのプロパティの設定
- アクション情報の共有
- エキスパート・ビューのアクションの構文について
- アクションの終了

高度なアクション機能を使用した作業について

アクションを使用すると，Web サイトのメイン・セクションや，アプリケーション内でユーザが実行する特定の操作などの論理単位にテストを分割できます。

テストは，アクションへの呼び出しで構成されています。新しく作成したテストには，1 つのアクションへの呼び出しが含まれています。複数のアクションを呼び出すテストを作成することによって，モジュール化された，効率の良いテストを設計できます。

アクション間で情報を受け渡すには，いくつかの方法があります。アクション内のステップが，テストの別の場所から指定された値を使用できるように，アクションに入力パラメータを指定することもできます。また，アクションから値を出力して，テストの後半のステップで使用したり，テストを実行したアプリケーションに返したりすることもできます。詳細については，15 ページ「アクション・パラメータの使用」を参照してください。

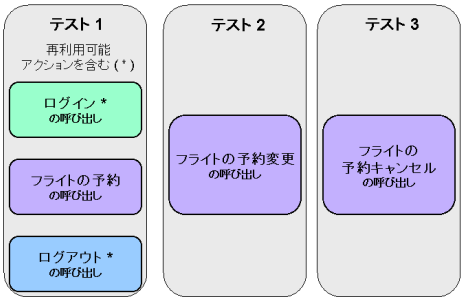
既存のアクションへの呼び出しの挿入

一連のテストを計画する場合，各テストで「ログイン」のような同一の動作がいくつか必要であることがあります。3 つの異なるテストでログイン・プロセスを 3 回記録し，スクリプトの当該部分をテストごとに（チェックポイント，パラメータ化，およびプログラミング・ステートメントによって）別々に拡張するのではなく，フライト予約システムにログインする 1 つのアクションを作成し，それを 1 つのテストに格納します。アクションの作成が済んだら，作成した既存のアクションへの呼び出しを他のテストに挿入できます。

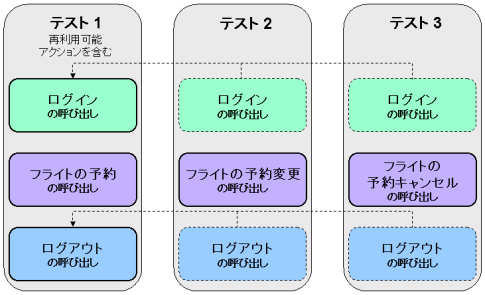
既存のアクションへの呼び出しを挿入するには，アクションのコピーへの呼び出しを挿入するか，元のアクションへの呼び出しを挿入します。

たとえば，フライトの予約，予約の変更，予約の削除の 3 つのテストを Mercury Tours サイトで記録するとします。テストを計画するときに，各テストについて，サイトへのログインとサイトからのログアウトが必要であるため，これら 3 つすべてのテストには合計 5 つのアクションが必要であることが分かったとします。

まず 5 つのアクションを使用して 3 つのテストを作成します。テスト 1 には 2 つの再利用可能なアクション（ログインとログアウト）が含まれています。これらのアクションは，後でテスト 2 およびテスト 3 から呼び出すことができます。



その後，テスト 2 およびテスト 3 にテスト 1 で作成した再利用可能なアクションへの呼び出しを挿入して作成が完了します。



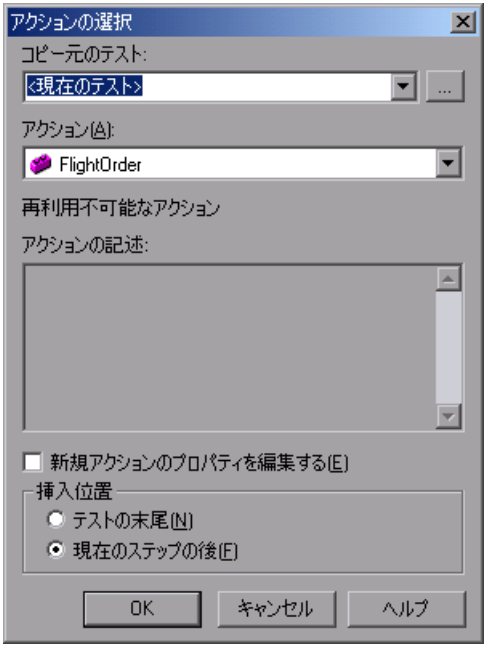
アクションのコピーへの呼び出しの挿入

アクションのコピーへの呼び出しをテストに挿入すると，チェックポイント，パラメータ化の内容，データ・テーブルの対応する［アクション］タブ，およびすべての定義済みアクション・パラメータを含む，元のアクション全体がコピーされます。コピーするテストのオブジェクトがローカル・オブジェクト・リポジトリにある場合，コピーしたアクションのローカル・オブジェクト・リポジトリもアクションと一緒にコピーされます。

アクションは，独立した，再利用不可能なアクションとして（元のアクションが再利用可能であっても）テストに挿入されます。アクションをテストにコピーしたら，他の再利用不可能なアクションと同様に，アクションを対象とした追加，削除，変更ができます。挿入したアクションに加えた変更はそのアクションにだけ影響し，元のアクションに対する変更は，コピーしたアクションに影響を及ぼしません。

アクションのコピーを作成し，テストでそのコピーを呼び出すには，次の手順を実行します。

1 テストの記録または編集中に，［**挿入**］＞［**アクションのコピーの呼び出し**］を選択するか，アクションのアイコンを右クリックして［**アクションのコピーへの呼び出しを挿入**］を選択するか，あるいは，任意のステップを右クリックして［**アクション**］＞［**コピーへの呼び出しを挿入**］を選択します。［アクションの選択］ダイアログ・ボックスが開きます。



2 コピーするアクションが含まれるテストを見つけるには，［**コピー元のテスト**］の参照ボタンを使用します。［**アクション**］ボックスには，すべてのローカル・アクション（選択したテストとともに格納されるアクション）が表示されます。

注 :［**コピー元のテスト**］ボックスには，Quality Center フォルダまたは相対パスを入力できます。相対パスを入力すると，QuickTest によって［オプション］ダイアログ・ボックスの［フォルダ］タブで指定されているフォルダのテストが検索されます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 702 ページ「テストのフォルダ・オプションの設定」を参照してください。

---

3 ［**アクション**］リストでは，挿入するアクションを選択します。アクションを選択すると，そのタイプ（**再利用不可能なアクション**または**再利用可能なアクション**）および存在する場合は説明が表示されます。これによって，コピーするアクションを識別できます。アクションの説明の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 473 ページ「一般的なアクション・プロパティの設定」を参照してください。

4 コピーしたアクションのプロパティを変更する場合は，［**新規アクションのプロパティを編集する**］チェック・ボックスを選択します。このオプションを選択した場合は，［**OK**］をクリックすると［アクションのプロパティ］ダイアログ・ボックスが表示されます。20 ページ「アクションの呼び出しのプロパティの設定」の説明に従って，アクション・プロパティを修正できます。

注 : このオプションを選択していない場合は，キーワード・ビューでアクションのアイコンを右クリックし，［**アクションのプロパティ**］を選択して，後でアクションのプロパティを変更できます。

---

5 アクションのコピーへの呼び出しを挿入する場所を決定し，［**テストの末尾**］または［**現在のステップの後**］を選択します。

アクションへのアクションの挿入の詳細については，15 ページ「アクション・パラメータの使用」を参照してください。

注：現在選択されているステップが他のテストからの再利用可能なアクションである場合は，アクションのコピーへの呼び出しは自動的にテストの末尾に追加されます（[**現在のステップの後**］のオプションは無効になります)。

---

6 ［**OK**］をクリックします。アクションへの呼び出しは，独立した，再利用不可能なアクションとしてテストに挿入されます。アクション呼び出しをテスト内の任意の場所にドラッグして移動できます。アクションの移動の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 135 ページ「階層でのアクションとステップの移動」を参照してください。

既存のアクションの呼び出しの挿入

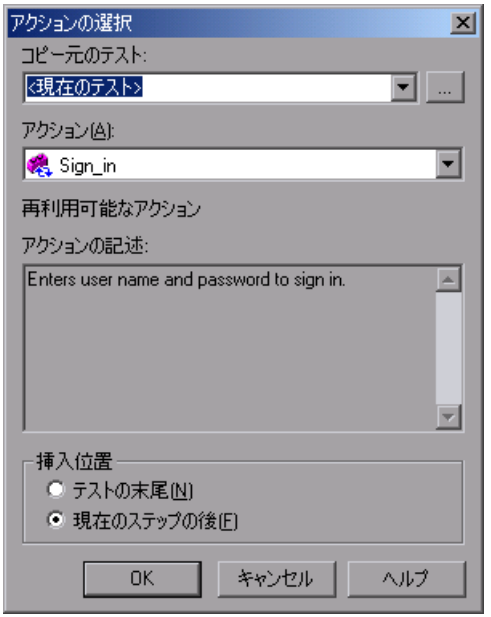
現在のテスト（ローカル・アクション)，または別のテスト（外部アクション）に格納されている再利用可能なアクションの呼び出しを挿入することができます。既存のアクションの呼び出しの挿入は，呼び出しにリンクを設定するのに似ています。アクションのステップはアクション・ビューに表示できますが，変更はできません。呼び出し先アクションのローカル・オブジェクト・リポジトリ（存在する場合）もまた読み取り専用です。ただし，外部アクションを呼び出す場合，アクションのデータ・シートからのデータをローカルの編集可能なコピーとしてインポートするか，元のアクションから（読み取り専用の）データを使用するかを選択できます。

呼び出した外部アクションを変更するには，アクションとともに保存されているテストを開いて，そのテストで変更を行う必要があります。この変更は，そのアクションを呼び出すすべてのテストに適用されます。元のアクションのデータを使用する場合，外部アクションを呼び出すと，元のアクションのデータに対する変更も適用されます。

ヒント：元のアクションの場所は，［アクションのプロパティ］ダイアログ・ボックスの［一般］タブに表示されます。

既存のアクションへの呼び出しを挿入するには，次の手順を実行します。

1 ［**挿入**］＞［**既存アクションの呼び出し**］を選択するか，アクションのアイコンを右クリックして［**既存アクションへの呼び出しを挿入**］を選択するか，任意のステップを右クリックして［**アクション**］＞［**既存への呼び出しを挿入**］を選択します。［アクションの選択］ダイアログ・ボックスが開きます。



2 呼び出すアクションが含まれるテストを見つけるには，［**コピー元のテスト**］参照ボタンを使用します。［**アクション**］ボックスには，選択したテストのすべての再利用可能なアクションが表示されます。

注 : ［**コピー元のテスト**］ボックスには，Quality Center フォルダまたは相対パスを入力できます。相対パスを入力すると，QuickTest によって［オプション］ダイアログ・ボックスの［フォルダ］タブで指定されているフォルダのテストが検索されます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 702 ページ「テストのフォルダ・オプションの設定」を参照してください。

---

3 ［**アクション**］リストで，呼び出すアクションを選択します。アクションを選択すると，そのタイプ（**再利用可能なアクション**）および存在する場合は説明が表示されます。これによって，呼び出すアクションを識別できます。アクションの説明の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 473 ページ「一般的なアクション・プロパティの設定」を参照してください。

ヒント：テストが呼び出す外部アクションは，リスト内にも表示されます。呼び出すアクションが選択したテスト内からすでに呼び出されている場合は，アクションのリストからそのアクションを選択できます。これにより，元のアクションへの別の呼び出しが作成されます。

注：選択したテストに再利用可能なアクションまたは外部アクションがない場合は，QuickTest によって［**アクション**］リストが無効になります。

---

4 アクションへの呼び出しを挿入する場所を決定し，［**テストの末尾**］または［**現在のステップの後**］を選択します。

注：現在選択されているステップが他のテストからの再利用可能なアクションである場合は，アクションへの呼び出しは自動的にテストの末尾に追加されます（**現在のステップの後** は無効になります）。

アクションへのアクションの挿入の詳細については，15 ページ「アクション・パラメータの使用」を参照してください。

5 ［**OK**］をクリックします。アクションへの呼び出し  がテスト・フローに挿入されます。アクション呼び出しをテスト内の任意の場所にドラッグして移動できます。アクションの移動の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 135 ページ「階層でのアクションとステップの移動」を参照してください。

ヒント：CTRL キーを押しながら，アクションをテスト内の並列（兄弟）レベルの別の場所にドラッグ・アンド・ドロップすると，テスト内の再利用可能なアクションや外部アクションへの追加呼び出しを作成できます。

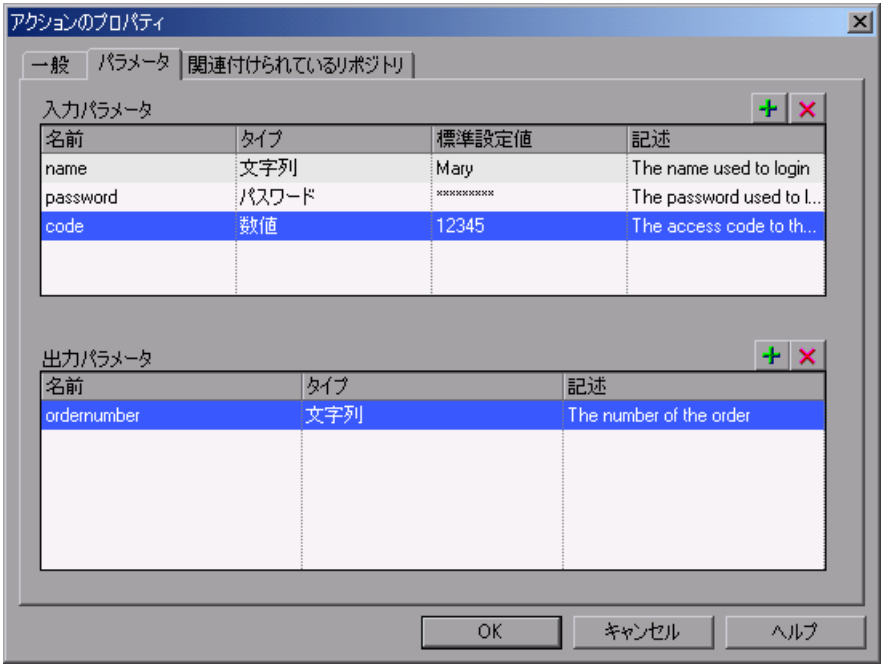
アクション・パラメータの設定

アクション内のステップが，テストの別の場所から指定された値を使用できるように，アクションに入力パラメータを指定できます。アクション・パラメータの入力値は，最上位レベルのアクション場合はテストから，ネストされたアクションの場合はそれを呼び出す親アクションのパラメータから，兄弟アクションの場合は直前のアクション呼び出しの出力値から取得できます。

アクションがテストの後の場所で使用する値を返せるように，アクションに出力パラメータを指定できます。たとえば，後でネストされたアクションが値を使用できるように，パラメータの値を親アクションに出力できます。

入力または出力アクション・パラメータごとに，名前とタイプ，および任意で説明を定義します。各アクション入力パラメータに標準設定の値を指定したり，ユーザが選択したパラメータ値のタイプに対して QuickTest が指定する標準設定の値を使用したりすることもできます。アクション呼び出しのパラメータに対して値が定義されていない場合は，標準設定の値はアクションとともに保存され，アクションにより使用されます。［アクションのプロパティ］ダイアログ・ボックスの［パラメータ］タブでは，入力および出力パラメータの定義，変更，および削除が行えます（［**編集**］＞［**アクション**］＞［**アクションのプロパティ**］を選択するか，アクションを右クリックして［**アクションのプロパティ**］を選択します）。

アクション・パラメータの使用方法の詳細については，15 ページ「アクション・パラメータの使用」および 18 ページ「アクション・パラメータを使った作業についてのガイドライン」を参照してください。



新しい入力または出力アクション・パラメータを追加するには，次の手順を実行します。

1 ［**入力パラメータ**］または［**出力パラメータ**］リストの上にある［**パラメータの追加**］ボタンをクリックして，新しいパラメータを適切なリストに追加します。関連するリストに新しいパラメータ用の行が追加されます。

2 ［**名前**］ボックスをクリックし，パラメータの名前を入力します。

3 [**タイプ**] ボックスでパラメータの値のタイプを選択します。次のタイプのいずれかを選択することができます。

- **文字列**：「"New York"」など，引用符のペアで囲まれた文字列。値を入力するときに引用符を含めなかった場合は，テスト実行時に値がスクリプトに挿入される際に QuickTest によって引用符が自動的に追加されます。標準設定の値は空の文字列です。
- **ブール値**：true または false の値。「**ブール値**」のタイプを選択した場合，[**標準設定値**] カラムをクリックして矢印をクリックすると，[**True**] または [**False**] の値を選択できます。標準設定の値は **True** です。
- **日付**：「2005/03/02」などの日付の文字列。「**日付**」値のタイプを選択した場合，[**標準設定値**] カラムをクリックして矢印をクリックすると，日付を選択できるカレンダーを開くことができます。標準設定の値は当日の日付です。
- **数値**：任意の数値。標準設定の値は 0 です。
- **パスワード**：暗号化されたパスワードの値。「**パスワード**」値のタイプを選択した場合は，[**標準設定値**] フィールドにパスワードを入力する際にはパスワードの文字はマスクで隠されます。ただし，アクション内では，値は暗号化されて表示されます。標準設定の値は空の文字列です。これも実際のアクション内では暗号化された値として表示されます。
- **任意**：バリアント型の値のタイプで，上記の値のタイプの任意を使用できます。「**任意**」値のタイプを選択した場合は，値を使用する予定の場所で必要な形式で値を指定する必要があります。たとえば，後に値を文字列として使用する予定である場合，文字列を引用符で囲む必要があります。「**任意**」の値タイプを指定した場合，QuickTest によってそれが数値であるかどうかがチェックされます。値が数値でない場合は，QuickTest によってその値は自動的に引用符で囲まれます。既存の値を編集している場合は，以前の値に引用符が付いていれば，その値は自動的に引用符で囲まれます。標準設定の値は空の文字列です。

4 入力アクション・パラメータを定義する場合は，[**標準設定値**] ボックスをクリックしてパラメータの標準設定値を入力します。あるいは，QuickTest によってそのパラメータ値のタイプに対して指定される標準設定値のままにしておくことができます。テスト内の他の場所からパラメータ値を受け取ることなくアクションを実行できるように，標準設定の値が必要になります。

5 （任意）［**記述**］ボックスをクリックし，アクション内でのパラメータの目的など，パラメータの説明を入力します。QuickTest はこれらの説明を，［出力オプション］，［パラメータ オプション］，［値設定オプション］ダイアログ・ボックスなど，アクション・パラメータを選択できる任意のダイアログ・ボックス内にパラメータの名前と一緒に表示します。

既存のアクション・パラメータを変更するには，次の手順を実行します。

1 変更するパラメータを［**入力パラメータ**］リストまたは［**出力パラメータ**］リストから選択します。

2 必要に応じて，パラメータ行のエディット・ボックスで値を変更します。

既存のアクション・パラメータを削除するには，次の手順を実行します。

1 削除するパラメータを［**入力パラメータ**］リストまたは［**出力パラメータ**］リストから選択します。

2 ［**パラメータの削除**］ボタンをクリックします。パラメータがリストから削除されます。

注：アクション・パラメータを削除する場合，そのアクション・パラメータを使用するステップもすべて必ず削除してください。

アクション・パラメータの使用

アクション・パラメータを使用すれば，テストから最上位レベルのアクションへ，親アクションからネストされたアクションへ，またはアクションからテスト内の後続の兄弟アクションへ，値を受け渡しできます。また，アクション・パラメータを使用して，アクション内のステップからその親アクションへ，または最上位レベルのアクションから，テストを実行（または呼び出した）スクリプトまたはアプリケーションへ，出力値を受け渡しできます。たとえば，ネストされたアクションのステップから値を出力し，その値を出力アクション・パラメータに格納してから，呼び出し元の親アクションの後のステップで入力としてその値を使用することができます。

アクション・パラメータはアクション内の任意のステップ（関数呼び出しを含む）で使用できます。アクションが受け取ることのできるパラメータ，およびアクションが返すことができる出力値は，［アクションのプロパティ］ダイアログ・ボックスの［パラメータ］タブで定義します（［**編集**］＞［**アクション**］＞［**アクションのプロパティ**］を選択するか，アクションを右クリックして［**アクションのプロパティ**］を選択します）。［アクション呼び出しプロパティ］ダイアログ・ボックス（アクションを右クリックして［**アクション呼び出しプロパティ**］を選択すると開きます）の［パラメータの値］タブを使用して，これらのパラメータに供給される実際の値と，出力値が格納される場所を指定します。

アクションがテストの別の場所から入力値を受け取ることができるように，アクションに入力パラメータを指定できます。アクション・パラメータの入力値は，最上位レベルのアクション場合はテストから，ネストされたアクションの場合はそれを呼び出す親アクションのパラメータから，兄弟アクションの場合は直前のアクション呼び出しの出力値から取得できます。また，アクションの出力パラメータを指定して，テストの後の部分で使用したり，テストを実行したアプリケーションに返したりすることもできます

たとえば，テストを実行する（呼び出す）外部アプリケーションから値を取得し，その値をテスト内のアクションで使用するとします。次のテストでは，Action2 および Action3 を介して，外部アプリケーションから Action4 の必要なステップに，入力テスト・パラメータを渡す必要があります。

この操作は，次のように行います。

1 テストの後半で使用したい値を使用して，入力テスト・パラメータを定義します（［**ファイル**］＞［**設定**］＞［**パラメータ**］タブ）。

2 入力テスト・パラメータと同じ値のタイプを使用して，Action2 に対して入力アクション・パラメータを定義します（［**編集**］＞［**アクション**］＞［**アクションのプロパティ**］＞［**パラメータ**］タブ）。

3 前述の手順で指定した入力テスト・パラメータ値を使用して，入力アクション・パラメータ値をパラメータ化します（［**編集**］＞［**アクション**］＞［**アクション呼び出しプロパティ**］＞［**パラメータの値**］タブ）。

4 入力テスト・パラメータと同じ値のタイプを使用して，Action3 に対して入力アクション・パラメータを定義します（［**編集**］＞［**アクション**］＞［**アクションのプロパティ**］＞［**パラメータ**］タブ）。

5 入力アクション・パラメータ値をパラメータ化します。

- ［**編集**］＞［**アクション**］＞［**アクション呼び出しプロパティ**］＞［**パラメータの値**］タブを選択し，Action2 に対して指定した入力アクション・パラメータ値を選択します。
- **Parameter** ユーティリティ・オブジェクトを使用して，アクション・パラメータを［エキスパート ビュー］の **RunAction** ステートメントの **parameter** 引数として指定します。詳細については，28 ページ「パラメータを使用したアクションの呼び出し」を参照してください。

6 入力テスト・パラメータと同じ値のタイプを使用して，Action4 に対して入力アクション・パラメータを定義します（［**編集**］＞［**アクション**］＞［**アクションのプロパティ**］＞［**パラメータ**］タブ）。

7 入力アクション・パラメータ値をパラメータ化します。

- ［**編集**］＞［**アクション**］＞［**アクション呼び出しプロパティ**］＞［**パラメータの値**］タブを選択し，Action3 に対して指定した入力アクション・パラメータ値を選択します。
- **Parameter** ユーティリティ・オブジェクトを使用して，アクション・パラメータを［エキスパート ビュー］の **RunAction** ステートメントの **parameter** 引数として指定します。詳細については，28 ページ「パラメータを使用したアクションの呼び出し」を参照してください。

8 Action4 で必要なステップの値をパラメータ化します。

- パラメータ化アイコン  をクリックし，Action4 に対して指定した入力アクション・パラメータを使用して［値設定オプション］ダイアログ・ボックスでパラメータを指定します。
- ［エキスパート ビュー］で **Parameter** ユーティリティ・オブジェクトを使用して，ステップに使用する値を指定します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 368 ページ「［エキスパート ビュー］のステップでのアクション・パラメータの使用」を参照してください。

アクションのパラメータはそのアクションとともに保存され，そのアクションに対するすべての呼び出しに対して同じになります。アクション・パラメータの名前，タイプ，または説明を変更し，テストの別の部分にある，同じアクションへの呼び出しのアクション・プロパティを表示すると，アクション・パラメータが変更されたことを確認できます。

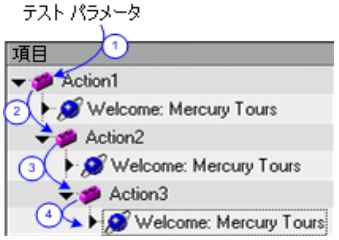
入力アクション・パラメータに対して定義されている実際の値と，アクション出力パラメータに対して指定されている場所は，アクションに対する呼び出しごとに異なることが可能です。アクションのコピーへの呼び出しを挿入する場合，アクションのコピーは，ユーザがコピーしたアクションに対して定義されていたアクション・パラメータおよびアクション呼び出しパラメータとともに挿入されます。アクションを分割すると，アクション・パラメータは両方のアクションにコピーされます。第 2 のアクションのアクション呼び出し値は，そのアクションのパラメータの標準設定値から取得されます。

アクション・パラメータと，アクション呼び出しで使用される値の定義の詳細については，12 ページ「アクション・パラメータの設定」および 22 ページ「アクションの呼び出しのパラメータ値の設定」を参照してください。

アクション・パラメータを使った作業についてのガイドライン

アクション・パラメータを使って作業をする場合，次のガイドラインを考慮します。

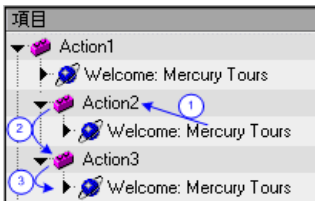
- 入力アクション・パラメータ値は，現在のアクションのステップ内でのみ使用できます。別のアクション（またはテスト）からのアクション入力値は，値を使用するアクションまで値をテスト階層の下方に向かってアクションからアクションへ受け渡した場合にのみ使用できます。例を次に示します。
  テスト → アクション 1 → アクション 2 → アクション 3 →（アクション 3）ステップ 1



- 出力アクション・パラメータ値は，同じ階層の直前のアクション，親アクション，または現在のアクションから取得できます。次の場合は，別のアクションのステップ内にある 1 つのアクションからのアクション出力値を使用できます。
  - アクションからアクションへテスト階層の上方向へ，値を使用するアクションまで値を渡す場合。
    例：**（アクション 3）ステップ 1 → アクション 3 → アクション 2 → アクション 1 → テスト → アクション 4**

この例では，出力値はアクション 4 のステップで使用されていますが，アクション 1，アクション 2，またはアクション 3 のどのステップでも，（アクション 3）ステップ 1 の出力値を使用できます。

  - 直前のアクションから，値を使用する兄弟アクションへ値を渡す場合。
    例：**（アクション 2）ステップ 1 → アクション 2 → アクション 3 → （アクション 3）ステップ 1**



この例では，出力値は（アクション 3）ステップ 1 で使用されていますが，アクション 2 またはアクション 3 のステップでも，（アクション 2）ステップ 1 の出力値を使用できます。

➤ 呼び出し元アクションの後続のステップでは，呼び出し先アクションから取得した値であれば，任意のタイプのアクション出力値を変数として使用できます。たとえば，ActionA が ActionB を呼び出し，ActionB の出力パラメータを格納する変数として MyBVar を指定した場合，ActionB への呼び出しの後の ActionA のステップでは，他の変数を使用するのと全く同じように MyBVar を値として使用できます。

アクションの呼び出しのプロパティの設定

［アクション呼び出しプロパティ］ダイアログ・ボックスは，アクションへの特定の呼び出しにおけるアクションの動作を制御します。呼び出し先のアクションを QuickTest が（データ・テーブルの列の数に従って）何回実行するかを指定できるだけでなく，入力アクション・パラメータの初期値や，出力アクション・パラメータの値を格納する場所も指定できます。

注：次の項では，［アクション呼び出しプロパティ］ダイアログ・ボックスを使用してアクションの呼び出しのプロパティを定義する方法を説明します。また，［エキスパート ビュー］でアクションの呼び出しとアクションの呼び出しのパラメータを定義することもできます。詳細については，28 ページ「エキスパート・ビューのアクションの構文について」を参照してください。

テストの記録または編集中に［アクション呼び出しプロパティ］ダイアログ・ボックスを開くには，次のいずれかを実行します。

➤ アクション・ノードが強調表示されている状態で，キーワード・ビューから［**編集**］＞［**アクション**］＞［**アクション呼び出しプロパティ**］を選択する。

➤ キーワード・ビュー内のアクション・ノードを右クリックし，［**アクション呼び出しのプロパティ**］を選択する。

［アクション呼び出しプロパティ］ダイアログ・ボックスでは，特定のアクション呼び出しにのみ適用されるオプションを設定できます。このダイアログ・ボックスには，［実行］タブと［パラメータの値］タブがあります。

アクションの実行プロパティの設定

［アクション呼び出しプロパティ］ダイアログ・ボックスの［実行］タブを使って，QuickTest に対し呼び出したアクションで反復を 1 回だけ実行したり，データ・テーブルのすべての行を対象に反復を実行したり，データ・テーブルの特定の行だけを対象に反復を実行したりする指定が行えます。

［実行］タブには，次のオプションがあります。

オプション	詳細
［反復なしで実行する］	アクションのデータ・テーブルにある最初の行を使って，呼び出したアクションを 1 回だけ実行します。
［すべての行で実行する］	アクションのデータ・テーブルの行数に従った反復の回数，呼び出したアクションを実行します。
［実行開始行 X 終了行］	指定した行の範囲に従った反復の回数，呼び出したアクションを実行します。

注：

あるアクションで複数の反復を実行する場合，アクションがアプリケーションの同一の場所で開始および終了するようにして，アプリケーションがアクションの次の反復を実行するときに，正しい場所と状態にあるようにする必要があります。

［アクション呼び出しプロパティ］ダイアログ・ボックスの［実行］タブは，アクションの呼び出しごとに適用され，アクションのデータ・シートの行を参照します。［テストの設定］ダイアログ・ボックスの［実行］タブで，テスト全体の実行プロパティを設定できます（グローバル・データ・シートの行を対象とした反復の設定）。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 25 章「個別のテストのオプションの設定」を参照してください。

アクションの呼び出しのパラメータ値の設定

呼び出したアクションにより使用される入力アクション・パラメータの値を指定したり，出力アクション・パラメータ値を格納する場所を指定したりする場合は，［アクション呼び出しプロパティ］ダイアログ・ボックスの［パラメータの値］タブを使用します。使用可能な任意のパラメータ・タイプを使用して，特定の入力アクション・パラメータに使用される値をパラメータ化することもできます。

アクションが受け取ったり返したりできる実際の入力および出力アクション・パラメータと，それらのタイプは，［アクションのプロパティ］ダイアログ・ボックスで定義します。

注：アクションの呼び出しでの入力および出力パラメータ値の指定は任意です。

入力アクション・パラメータの値を設定しない場合は，［アクションのプロパティ］ダイアログ・ボックスで指定されている標準設定の値が使用されます。

出力パラメータ値の格納場所を定義しない場合でも，呼び出し元アクションは，呼び出し先アクションによって生成された出力パラメータ・データにアクセスできます。ただし，格納場所を指定したほうが，アクション呼び出しステートメントは読みやすくなります。

入力および出力アクション・パラメータの定義の詳細については，20 ページ「アクションの呼び出しのプロパティの設定」を参照してください。アクション・パラメータの使用法の詳細については，15 ページ「アクション・パラメータの使用」を参照してください。

入力アクション・パラメータの値を指定するには，次の手順を実行します。

1 ［**入力パラメータ**］領域でパラメータの［**値**］ボックスをクリックし，値を入力します。各値のタイプに使用可能なさまざまなオプションの詳細については，12 ページ「アクション・パラメータの設定」の定義を参照してください。

または，［**値**］ボックスのパラメータ化のボタン  をクリックして，値をパラメータ化できる［値設定オプション］ダイアログ・ボックスを開きます。値のパラメータ化には，テスト・パラメータまたはアクション・パラメータ（最上位レベルのアクションの場合はテスト・パラメータ，ネストされたアクションまたは兄弟アクションの場合はアクション・パラメータ），データ・テーブル・パラメータ，環境パラメータ，または乱数パラメータを使用できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 15 章「値のパラメータ化」を参照してください。

2 設定する必要がある追加の入力アクション・パラメータに対してこの手順を繰り返します。

出力アクション・パラメータ値を格納する場所を指定するには，次の手順を実行します。

1 ［**出力パラメータ**］領域でパラメータの［**保管先**］ボックスをクリックし，変数名を入力します。

または，［**保管先**］ボックスの出力の格納のボタン  をクリックして，出力値を格納する場所を指定できる［保管場所オプション］ダイアログ・ボックスを開きます。値の格納先として，テスト・パラメータ，呼び出し元アクション・パラメータ，データ・テーブル・パラメータ，または環境パラメータを選択できます。詳細については，25 ページ「アクション情報の共有」および『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 548 ページ「戻り値およびアクション出力パラメータ値の格納」を参照してください。

2 リスト内の各出力アクション・パラメータに対してこの手順を繰り返します。

アクション情報の共有

アクションどうしで値を共有したり，互いに受け渡ししたりする方法は複数あります。

- 呼び出し先アクションの出力アクション・パラメータに値を格納し，呼び出し元アクション内のアクションの呼び出しの後に実行されるステップ，または兄弟アクション内のステップでこれらの値を使用する。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 404 ページ「テスト・パラメータおよびアクション・パラメータへの値の格納」を参照してください。
- あるアクションで生成された値をグローバル・データ・テーブルに格納し，それらの値を別のアクションでデータ・テーブル・パラメータとして使用する。詳細については，「グローバル・データ・テーブルを使用した値の共有」を参照してください。
- あるアクションで値をユーザ定義の環境変数として設定し，別のアクションでその環境変数を使用する。詳細については，26 ページ「環境変数を使用した値の共有」を参照してください。
- あるアクションで VBScript Dictionary オブジェクトに値を追加し，別のアクションでその値を読み込む。詳細については，27 ページ「Dictionary オブジェクトを使用した値の共有」を参照してください。

グローバル・データ・テーブルを使用した値の共有

グローバル・データ・テーブルに値を格納することによって，あるアクションで生成された値をテスト内で他のアクションと共有できます。これにより，他のアクションはデータ・テーブルの値を入力パラメータとして使用できます。データ・テーブルに値を格納するには，グローバル・データ・テーブルに値を出力するか，［エキスパート ビュー］で **Data Table**，**Sheet**，**Parameter** オブジェクトおよびメソッドを使用して値を追加または変更します。

たとえば，フライト予約アプリケーションをテストするとします。ユーザがアプリケーションにログインすると，ページの最上部にユーザの氏名が表示されます。操作を進め，チケットを購入することにした場合，ユーザはクレジット・カードに表示されている名前を入力する必要があります。テストに Login，SelectFlight，PurchaseTickets の 3 つのアクションがあり，反復ごとに異なるログイン名を使って複数の反復を実行するように設定されているとします。Login アクションで，表示されたユーザ名を格納するテキスト出力値を作成できます。PurchaseTickets アクションでは，ユーザの氏名を含むデータ・テーブル・カラムを使用して，［Credit Card Owner］エディット・ボックスで設定する値をパラメータ化できます。

出力値の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 16 章「値の出力」を参照してください。パラメータ化の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 15 章「値のパラメータ化」を参照してください。データ・テーブル・オブジェクトおよびメソッドの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 19 章「データ・テーブルを使った作業」，および『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

環境変数を使用した値の共有

テストの反復を何度も実行する必要がない場合，または共有している値をすべての反復で一定に保つ場合は，テストのすべてのローカル・アクションがアクセスできる，内部のユーザ定義環境変数を使用できます。

たとえば，ユーザが入力するクレジット・カードの失効日を，フライト予約アプリケーションが正確に検査するかどうかをテストするとします。入力された失効日が，予定されたフライト出発日以前の場合，アプリケーションは別のクレジット・カードを要求するはずです。SelectFlight アクションで，出発日エディット・ボックスに入力された値を環境変数に格納できます。そして PurchaseTickets アクションで，失効日エディット・ボックスの値と環境変数に格納された値を比較できます。

環境変数の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 15 章「値のパラメータ化」を参照してください。**Environment** オブジェクトの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

Dictionary オブジェクトを使用した値の共有

すでに説明したように，アクション間で値を共有するために環境変数を使用する代わりに，VBScript Dictionary オブジェクトを使用できます。Dictionary オブジェクトを使用すると，Dictionary オブジェクトが作成されるテストで呼び出されるすべてのアクション（ローカルおよび外部）からアクセス可能な変数に，値を割り当てることができます。

Dictionary オブジェクトを使用するには，最初に ProgID = "Scripting.Dictionary" である予約オブジェクトをレジストリ（**HKEY_CURRENT_USER¥Software¥Mercury Interactive¥QuickTest Professional¥MicTest¥ReservedObjects¥**）に追加する必要があります。例を次に示します。

```
HKEY_CURRENT_USER¥Software¥Mercury Interactive¥QuickTest
Professional¥MicTest¥ReservedObjects¥GlobalDictionary
```

予約 Dictionary オブジェクトをレジストリに追加し，QuickTest を再起動すると，あるアクションでディクショナリを対象に値の追加と削除を行い，同じテストの別のアクションでその値を取得できるようになります。

たとえば，SelectFlight アクションで設定された出発日を PurchaseTickets アクションで利用したい場合，次のように，DepartDate WebEdit オブジェクトの値を SelectFlight アクションのディクショナリに追加できます。

```
GlobalDictionary.RemoveAll
GlobalDictionary.Add "DateCheck", DepartDate
```

これによって，次のように PurchaseTickets アクションから日付を取得できます。

```
Dim CompareDate
CompareDate=GlobalDictionary("DateCheck")
```

Dictionary オブジェクトの詳細については，VBScript リファレンスのドキュメント（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］＞［**Script ランタイム**］を選択）を参照してください。

エキスパート・ビューのアクションの構文について

エキスパート・ビューでのアクションへの呼び出しでは，アクションの反復，入力パラメータ値，出力パラメータの格納場所，およびアクションの戻り値を定義できます。

基本構文を使用したアクションの呼び出し

エキスパート・ビューでは，パラメータを持たないアクションへの呼び出しは，次の基本構文を使用して呼び出し元のアクション内に表示されます。

RunAction ActionName, IterationQuantity

たとえば，**Select Flight** アクションを呼び出し，その反復を 1 回のみ実行するには，次の構文を使用します。

RunAction "Select Flight", oneIteration

たとえば，**Select Flight** アクションを呼び出し，データ・テーブルの行の数と同じ回数反復を実行するには，次の構文を使用します。

RunAction "Select Flight", allIterations

たとえば，**Select Flight** アクションを呼び出し，その反復を 4 回実行する（データ・テーブルの最初の 4 行）には，次の構文を使用します。

RunAction "Select Flight", "1 - 4"

パラメータを使用したアクションの呼び出し

呼び出しているアクションに入力パラメータや出力パラメータがある場合は，入力パラメータの値や，出力パラメータの格納場所を **RunAction** ステートメントの引数として指定できます。入力パラメータは出力パラメータの上に表示されます。

入力パラメータには，固定値を指定するか，引数が値を取得する別の定義済みパラメータ（呼び出し元のアクションのデータ・テーブル・パラメータ，環境パラメータ，アクション入力パラメータ）の名前を指定します。

出力パラメータには，値を格納する変数，または定義済みのパラメータ（呼び出し元のアクションのデータ・テーブル・パラメータ，環境パラメータ，アクション入力パラメータ）の名前のいずれかを指定できます。

パラメータを使用したアクションの呼び出しには，次の構文があります。

RunAction ActionName, IterationQuantity, Parameters

たとえば，Action1 から Action2 を呼び出し，Action2 に入力パラメータと出力パラメータが 1 つずつ設定されているとします。

次のステートメントは，入力パラメータに MyValue の文字列値を指定し，MyVariable という変数に出力パラメータの結果の値を格納します。

```
RunAction "Action2", oneIteration, "MyValue", MyVariable
```

次のステートメントは，Action1 の Axn1_In 入力アクション・パラメータに定義されている値を入力パラメータの値として使用して，Action1 のデータ・テーブル・シートの Column1_out というカラムに出力パラメータの結果の値を格納します。

```
RunAction "Action2", oneIteration, Parameter("Axn1_In"),
DataTable("Column1_out", dtLocalSheet)
```

次の例では，最初のステートメントが標準設定の入力パラメータ値を使用して Action2 を呼び出します。2 番目のステートメントが，Action2 の Axn2_out 出力アクション・パラメータに定義されている値を Action3 の入力パラメータの呼び出しの値として使用し，Action1 の Axn1_out に出力パラメータの結果の値を格納することで，出力値を親アクション・レベルで利用できるようにします。

```
RunAction "Action2", oneIteration
RunAction "Action3", oneIteration, Parameter("Action2","Axn2_out"),
    Parameter("Axn1_out")
```

Action2 への呼び出しで格納場所が指定されていなくても，Action2 の出力パラメータを Action3 への呼び出しに使用できる点に注目してください。

アクションの戻り値の格納

RunAction ステートメントによって呼び出されたアクションに **ExitAction** ステートメントが含まれている場合，**RunAction** ステートメントは **ExitAction** の **RetVal** 引数の値を返せます。この戻り値は，アクション呼び出しそのものの戻り値であり，アクション呼び出しの特定の出力パラメータによって返される任意の値とは独立したものです。

アクション呼び出しの戻り値を格納する構文は次のとおりです。

MyRetVal=RunAction (ActionName, IterationQuantity, Parameters)

エキスパート・ビューの詳細については，第 5 章「エキスパート・ビューおよび関数ライブラリ・ウィンドウを使用した作業」を参照してください。**RunAction** ステートメントの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

アクションの終了

アクション全体が完了する前にアクションを終了するには，［エキスパートビュー］でスクリプトに行を追加します。このオプションは，アクションの現在の値を，実行のある時点の値に戻す場合，または条件ステートメントの結果に基づいて使用します。終了アクションのステートメントには，次の 4 種類があります。

- **ExitAction**：反復の属性に関係なく，現在のアクションを終了します。
- **ExitActionIteration**：アクションの現在の反復を終了します。
- **ExitRun**：反復の属性に関係なく，テストを終了します。
- **ExitGlobalIteration**：現在のグローバルの反復を終了します。

終了アクションのノードは，［テスト結果］ツリーに表示されます。終了アクションのステートメントによって値が戻されると，この値はアクション，反復，またはテストのサマリのいずれかに表示されます。

これらの機能の詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。テスト結果の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 23 章「テスト結果の分析」を参照してください。

第 2 章

仮想オブジェクトの学習

アプリケーションの任意の領域を「**仮想オブジェクト**」として定義することにより，QuickTest でこの領域がオブジェクトとして認識されるように設定できます。仮想オブジェクトを利用すると，QuickTest が通常ならば認識しないオブジェクトを対象とするテストの記録と実行が行えます。

本章では，次の項目について説明します。

- 仮想オブジェクトの学習について
- 仮想オブジェクトについて
- 仮想オブジェクト・マネージャについて
- 仮想オブジェクトの定義
- 無効な仮想オブジェクト定義の削除

仮想オブジェクトの学習について

アプリケーションには，標準のオブジェクトと同様に動作するにもかかわらず，QuickTest が認識できないオブジェクトが含まれることがあります。これらのオブジェクトを仮想オブジェクトとして定義し，ボタンやチェック・ボックスなどの標準クラスに割り当てることができます。これにより，実行セッション中，QuickTest によって，この仮想オブジェクトに対するユーザのアクションがエミュレートされます。テスト結果には，仮想オブジェクトは標準クラスのオブジェクトとして表示されます。

たとえば，ユーザがクリックするビットマップが含まれる Web ページを対象とするテストを記録するとします。このビットマップには，それぞれ異なるリンク・ページを表示するいくつかのハイパーリンク領域があります。テストを記録するときに，Web サイトでは，クリックしたビットマップ上の座標が検出され，リンク・ページが表示されます。

実行セッション中に QuickTest で必要な座標をクリックできるようにするため，その座標が含まれるビットマップ領域の仮想オブジェクトを定義し，これをボタン・クラスに割り当てます。テストを実行すると，QuickTest によって，仮想オブジェクトとして定義した領域のビットマップがクリックされ，対応するリンク・ページが Web サイトに表示されます。

仮想オブジェクトの定義は，仮想オブジェクト・ウィザードで行います（［**ツール**］＞［**仮想オブジェクト**］＞［**新規仮想オブジェクト**］）。このウィザードでは，仮想オブジェクトの割り当ての対象にする標準のオブジェクト・クラスを選択するよう指示されます。その後，十字形のポインタで，仮想オブジェクトの境界を指定します。次に，この仮想オブジェクトの親とする，テスト・オブジェクトを選択します。最後に，仮想オブジェクトの名前とコレクションを指定します。仮想オブジェクトの**コレクション**とは，仮想オブジェクト・マネージャに格納されている仮想オブジェクトの集合に名前を付けたものです。

注：QuickTest では，アナログや低レベル記録用の仮想オブジェクトはサポートされていません。低レベルの記録の詳細については，422 ページ「テストの記録と実行」を参照してください。

仮想オブジェクトについて

QuickTest では，境界線に基づいて仮想オブジェクトが識別されます。オブジェクトの境界を指定することで，Web ページまたはアプリケーション・ウィンドウにおけるオブジェクトのサイズと位置を指定します。仮想オブジェクトの親としてテスト・オブジェクトを割り当てると，仮想オブジェクトの境界座標はその親オブジェクトを基準とする相対座標と認識されます。テストを記録するときに，QuickTest によって，親オブジェクトに含まれる仮想オブジェクトが認識され，実行セッション中にこのオブジェクトを識別できるようオブジェクト・リポジトリにテスト・オブジェクトとして追加されます。仮想オブジェクトを手作業でオブジェクト・リポジトリに追加した場合も，QuickTest によって仮想オブジェクトがテスト・オブジェクトとして認識されます。

注：実行セッション中は，アプリケーション・ウィンドウは記録時と同じサイズで同じ位置になければなりません。それ以外の場合，親オブジェクトに対する仮想オブジェクトとの相対座標が記録時と異なり，それが実行セッションの正常な実行に影響を与える可能性があります。

仮想オブジェクト・マネージャから削除しなくても，仮想オブジェクトが認識されないようにすることも可能です。詳細については，40 ページ「無効な仮想オブジェクト定義の削除」を参照してください。

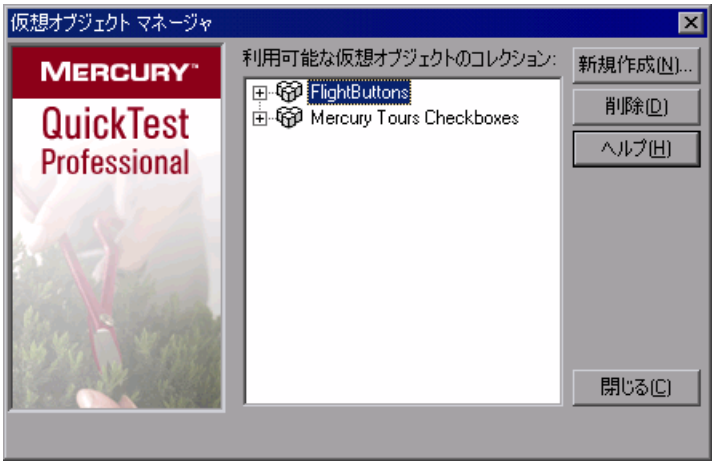
注：

仮想オブジェクトは，テストの記録時と実行時にのみ使用できます。仮想オブジェクトへのチェックポイントの挿入や，オブジェクト・スパイによる仮想オブジェクトのプロパティの表示を行うことはできません。

マークを付けた仮想オブジェクトを対象に ActiveScreen で操作を実行するには，そのオブジェクトをあらかじめ記録しておく必要があります。それにより，仮想オブジェクトのプロパティが，オブジェクト・リポジトリのテスト・オブジェクト記述に保存されます。まだ記録していない仮想オブジェクトを対象に ActiveScreen で操作を実行すると，仮想オブジェクトは標準オブジェクトとして扱われます。

仮想オブジェクト・マネージャについて

仮想オブジェクト・マネージャには，お使いのコンピュータで定義されているすべての仮想オブジェクト・コレクションが含まれます。仮想オブジェクト・マネージャで，仮想オブジェクトとコレクションの定義と削除が行えます。



［**利用可能な仮想オブジェクトのコレクション**］：お使いのコンピュータで定義されている仮想オブジェクト・コレクションと，それぞれのコレクションに含まれる仮想オブジェクトが表示されます。コレクションの横の［**+**］と［**-**］記号を使用して，コレクションに定義されている仮想オブジェクトを表示したり非表示にしたりできます。

［**新規作成**］：［仮想ユーザ オブジェクト ウィザード］を開きます。このウィザードは，新規あるいは既存のコレクションに新規仮想オブジェクトを定義する手順を示します。詳細については，40 ページ「無効な仮想オブジェクト定義の削除」を参照してください。

［**削除**］：選択した仮想オブジェクトまたは仮想オブジェクト・コレクションを削除します。詳細については，40 ページ「無効な仮想オブジェクト定義の削除」を参照してください。

注：［仮想オブジェクト マネージャ］に表示される仮想オブジェクト・コレクションは，お使いのコンピュータに格納されますが，仮想オブジェクト・ステップを含むテストと一緒には保存されません。つまり，テスト・ステップの中で仮想オブジェクトを使用した場合，オブジェクトが実行セッション中に認識されるのは，適切な仮想オブジェクト定義を含むコンピュータで実行された場合のみとなります。仮想オブジェクト・コレクション定義を別のコンピュータにコピーするには，
＜ QuickTest インストール・フォルダ＞ ¥dat¥VoTemplate フォルダの内容（またはこのフォルダ内の個々の **.vot** コレクション・ファイル）を，コピー先コンピュータの同じフォルダにコピーします。

仮想オブジェクトの定義

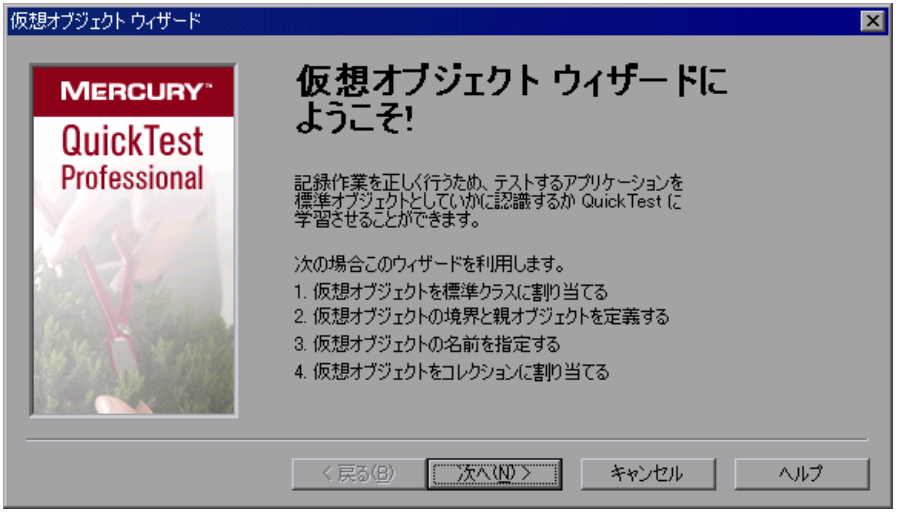
仮想オブジェクト・ウィザードを使って，仮想オブジェクトを標準のオブジェクト・クラスに割り当てたり，仮想オブジェクトの境界と親を指定したり，名前を割り当てたりできます。また，コレクションに割り当てることによって，仮想オブジェクトを論理的なグループにまとめることができます。

注：仮想オブジェクトとして定義できるのは，クリックまたはダブルクリックできるオブジェクトで，**Click** または **DblClick** ステップとして記録されるオブジェクトだけです。それ以外の仮想オブジェクトは無視されます。たとえば，WinList オブジェクトを仮想オブジェクトとして定義しようとすると，**Select** 操作は記録されますが，仮想オブジェクトは無視されます。

仮想オブジェクトを定義するには，次の手順を実行します。

1 QuickTest を記録モード以外のモードで起動している状態で，Web サイトまたはアプリケーションを開いて，仮想オブジェクトを定義する領域を含むオブジェクトを表示します。

2 QuickTest で，［**ツール**］＞［**仮想オブジェクト**］＞［**新規仮想オブジェクト**］を選択します。あるいは，［仮想ユーザ オブジェクト マネージャ］で［**新規作成**］をクリックします。仮想オブジェクト・ウィザードが開きます。

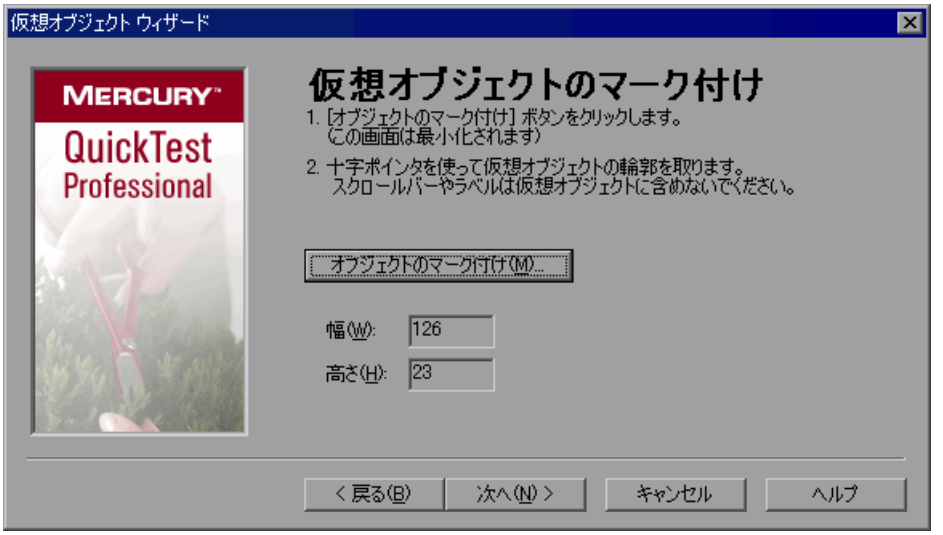


［**次へ**］をクリックします。

3 仮想オブジェクトの割り当て先の標準クラスを選択します。

list クラスを選択した場合，仮想オブジェクトの行数を指定します。table クラスの場合，行数とカラム数を選択します。［**次へ**］をクリックします。

4 ［**オブジェクトのマーク付け**］をクリックします。



QuickTest ウィンドウと仮想オブジェクト・ウィザードが最小化されます。十字形のポインタで，仮想オブジェクトの領域を指定します。マウスの左ボタンを押しながら矢印キーを使用すると，十字形ポインタで定義した領域を微調整できます。

［**次へ**］をクリックします。

注：仮想オブジェクトをマーク付けするときは，アプリケーションまたは Web ページの他の仮想オブジェクトと重ならないようにします。仮想オブジェクトが別の仮想オブジェクトと重なっていると，その仮想オブジェクトに対するテストが正しく記録または実行されないことがあります。

---

5 仮想オブジェクトの親として割り当てるオブジェクトをオブジェクト・ツリーの中でクリックします。

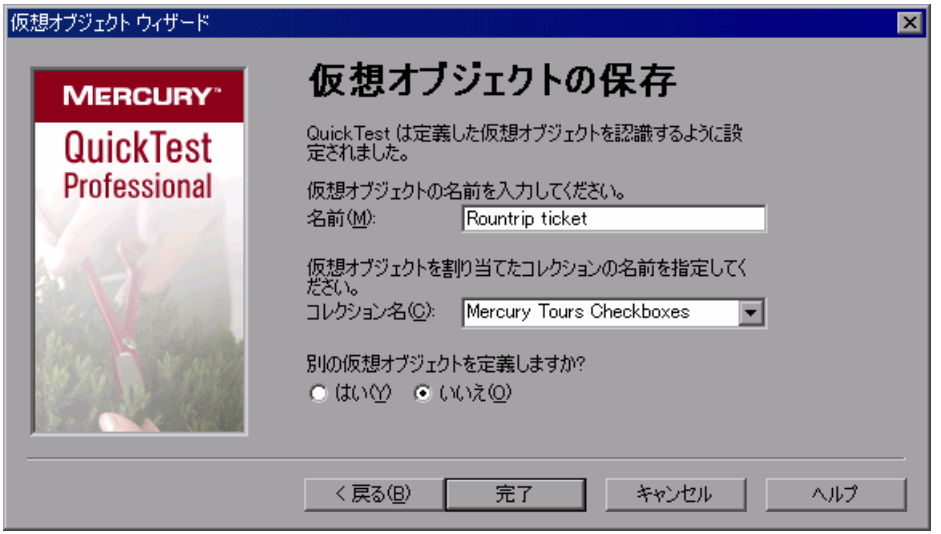
仮想オブジェクトの輪郭の座標は，選択した親オブジェクトを基準とする相対座標です。

6 ［**オブジェクト認識のレベル**］ボックスで，仮想オブジェクトの識別方法と割り当て方法を選択します。

- QuickTest で，仮想オブジェクトのすべての出現を識別させる場合，［**親のみ**］を選択します。親階層全体ではなく，直接の親だけを基準にして仮想オブジェクトが識別されます。たとえば，Browser("A").Page("B").Image("C") を使用して仮想オブジェクトが定義されている場合，階層が，Browser("X").Page("Y").Image("C") であっても仮想オブジェクトが識別されます。
- QuickTest で，仮想オブジェクトの特定 1 回の出現を識別させる場合，［**親階層の全体**］を選択します。QuickTest では，親階層が完全に一致している場合のみ仮想オブジェクトが識別されます。たとえば，Browser("A").Page("B").Image("C") を使用して仮想オブジェクトが定義されている場合，階層が Browser("X").Page("B").Image("C") に変わると，この仮想オブジェクトは識別されません。

［**次へ**］をクリックします。

7 仮想オブジェクトの名前とコレクションを指定します。コレクションのリストから選択するか，［**コレクション名**］ボックスに新しい名前を入力して新しいコレクションを作成します。



8 次の手順のいずれかを実行します。

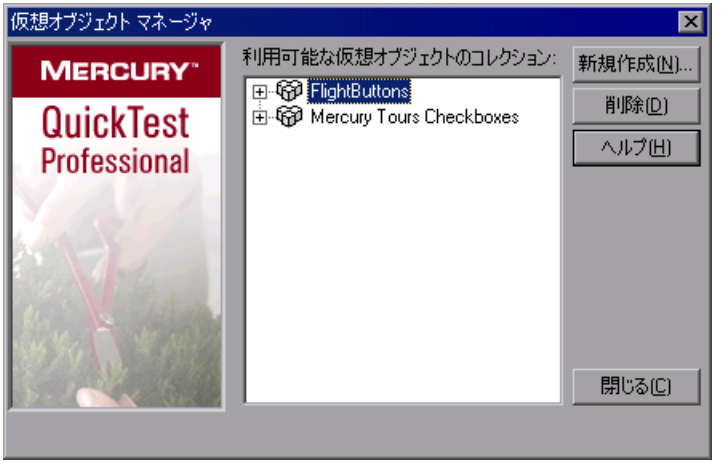
- 仮想オブジェクトを仮想オブジェクト・マネージャに追加してウィザードを閉じるには，［**いいえ**］を選択し，［**完了**］をクリックします。
- 仮想オブジェクトを仮想オブジェクト・マネージャに追加し，さらに別の仮想オブジェクトを定義するには，［**はい**］を選択し，［**次へ**］をクリックします。ウィザードが［標準クラスへの割り当て］画面に戻り，次の仮想オブジェクトを定義できます。

無効な仮想オブジェクト定義の削除

仮想オブジェクトをテストに含めないようにするには，記録された仮想オブジェクトをテストから削除するか，記録時に認識されないように設定します。

仮想オブジェクトを削除するには，次の手順を実行します。

1 ［**ツール**］＞［**仮想オブジェクト**］＞［**仮想ユーザ オブジェクト マネージャ**］を選択します。仮想オブジェクト・マネージャが起動します。



2 利用可能な仮想オブジェクトのコレクションのリストで，コレクションの横にあるプラス記号をクリックし，削除する仮想オブジェクトを表示します。仮想オブジェクトを選択し，［**削除**］をクリックします。

コレクション全体を削除するには，コレクションを選択し，［**削除**］をクリックします。

3 ［**閉じる**］をクリックします。

ヒント：新しい仮想オブジェクトを定義するには，仮想オブジェクト・マネージャで［**新規作成**］をクリックし，仮想オブジェクト・ウィザードを起動します。

記録中に仮想オブジェクトが認識されないようにするには，次の手順を実行します。

1 ［**ツール**］＞［**オプション**］を選択するか，［**オプション**］ツールバー・ボタンをクリックします。［オプション］ダイアログ・ボックスが開きます。

2 ［一般］タブで，［**記録時の仮想オブジェクト認識を無効にする**］チェック・ボックスをオンにします。

3 ［**OK**］をクリックします。

注：記録中に QuickTest で仮想オブジェクトが認識されるようにするには，［オプション］ダイアログ・ボックスの［一般］タブの［**記録時の仮想オブジェクト認識を無効にする**］チェック・ボックスがクリアされていることを確認します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 700 ページ「テストの一般オプションの設定」を参照してください。

第 3 章

回復シナリオの定義と使用

実行セッション中，テスト環境で生じる予期しないイベントやエラーから回復するよう QuickTest に指示できます。

本章では，次の内容について説明します。

- 回復シナリオの定義と使用について
- 回復シナリオを使用するタイミングの決定
- 回復シナリオの定義
- 回復シナリオ・ウィザードについて
- 回復シナリオの管理
- テスト用の回復シナリオ・リストの設定
- プログラムによる回復メカニズムの制御

回復シナリオの定義と使用について

実行セッション中に，予期しないイベント，エラー，およびアプリケーション・クラッシュが発生すると，実行セッションが妨げられ，正しいテスト結果が得られない可能性があります。これは，テストを自動で実行する場合に特に問題になります。回復に必要な操作を実行するまで，テストが一時停止状態になるからです。どのようなときに回復シナリオを使用するかについては，45 ページ「回復シナリオを使用するタイミングの決定」を参照してください。

回復シナリオ・マネージャでは，「**回復シナリオ**」を定義するプロセスを案内するウィザードが使用できます。回復シナリオは，予期しないイベントと，テストの実行を回復するために必要な操作の定義です。たとえば，「**Printer out of paper**」というメッセージを検出し，［**OK**］ボタンをクリックしてメッセージを閉じることによって実行セッションを回復し，テストを続行するよう QuickTest に指示できます。

回復シナリオは，次の要素で構成されています。

- **トリガ・イベント**：実行セッションを中断するイベントです。たとえば，画面上にポップアップ表示されるウィンドウや，QuickTest の実行エラーなどです。
- **回復操作**：テストの実行を続行するために実行しなければならない操作です。たとえば，ポップアップ・ウィンドウの［**OK**］ボタンをクリックすることや，Microsoft Windows の再起動などです。
- **回復後のテスト実行のオプション**：回復操作を実行してからの QuickTest の継続方法の指示，および回復操作を実行した場合はテストのどのポイントから QuickTest を継続するかの指示です。たとえば，テストを初めからやり直したり，完全に 1 つのステップをスキップしたり，テストの次のステップから続けたりすることができます。

回復シナリオは回復シナリオ・ファイルに保存されます。回復シナリオ・ファイルは回復シナリオの論理的な集合で，特定の独自の要件に従ってグループ化されています。

実行セッション中に回復シナリオを実行するよう QuickTest に指示するには，まずそのテストに回復シナリオを関連付ける必要があります。テストに関連付けることができる回復シナリオの数に制限はありません。テストに関連付けるシナリオに優先順位を付けて，必要な順序でトリガ・イベントを認識，処理させることができます。詳細については，77 ページ「テストへの回復シナリオの追加」を参照してください。

回復シナリオを定義した対象のテストを実行して，エラーが発生した場合は，QuickTest によって，エラーの原因である定義済みのトリガ・イベントが検索されます。トリガ・イベントが発生した場合，QuickTest によって，対応する回復操作および回復後操作が実行されます。

テストに Recovery ステートメントを挿入することで，実行セッション中に回復シナリオを制御し，呼び出すこともできます。詳細については，82 ページ「プログラムによる回復メカニズムの制御」を参照してください。

注 : ［テストの設定］ダイアログ・ボックスの［回復］タブにある［**回復シナリオのアクティブ化**］ボックスで「**エラー発生時**」を選択した場合，回復メカニズムでは，テストの最後のステップで発生するトリガは処理されません。このオプションを選択し，かつテストの最後のステップで発生する可能性のある予期しないイベントやエラーから回復する必要がある場合は，テストの最後にさらにステップを追加することで，予期しないイベントやエラーから回復できます。

回復シナリオを使用するタイミングの決定

イベントがテストの特定のポイントで発生することが予測できる場合は，回復シナリオを使用するのではなく，**If** ステートメントオプション・ステップなどを追加して，テストから直接イベントを処理することをお勧めします。たとえば，実行セッション中に［**保存**］ボタンがクリックされるとファイルの上書きメッセージ・ボックスが表示される場合は，メッセージ・ボックスが開くと［**OK**］をクリックする **If** ステートメントを追加するか，メッセージ・ボックスで［**OK**］をクリックするオプション・ステップを追加することでこのイベントを処理できます。テストから直接イベントを処理すると，回復シナリオよりもより明確にエラー処理が行えます。これは回復シナリオが一般的な予測しないイベントのセットを処理するよう設計されているためです。またテストから直接イベントを処理すれば，調整処理のタイミングを制御でき，最小の労力で最大の効果を挙げることができます。標準設定では，回復処理はステップがエラーを返した後にのみ有効になります。つまり，実際にエラーの原因となったステップよりもいくつか後のステップで行われる可能性があります。また，ステップごとにトリガ・イベントを確認すれば，パフォーマンスが低下します。

回復シナリオは，予期しないイベントや，テスト内の特定のステップと同期化できないイベントにのみ使用します。たとえば，回復シナリオは［プリンタ エラー］メッセージ・ボックスの標準設定のボタンをクリックすることによってプリンタ・エラーを処理できます。ネットワークがプリンタ・エラーを返すのがどのポイントか知ることは不可能なので，このエラーはテスト内から直接処理することはできません。テストでこのイベントを処理するには，プリンタにファイルを送るステップの直後に **If** ステートメントを追加します。しかし，ネットワークがプリンタ・エラーを返すまでに時間がかかると，テストはエラーが表示されるまでにステップがいくつか進んでしまう場合があります。したがって，この種のイベントの場合は，回復シナリオだけが処理できることになります。

オプション・ステップの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 616 ページ「オプション・ステップの使用」参照してください。**If** ステートメントなど，プログラミング・ステートメントの挿入の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 20 章「プログラミング・ロジックを含むステップの追加」を参照してください。

回復シナリオの定義

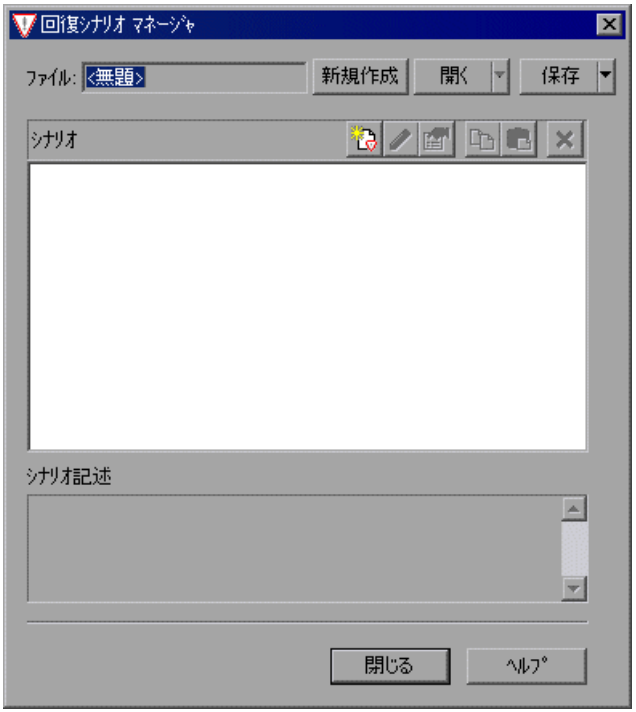
［回復シナリオ マネージャ］ダイアログ・ボックスでは，回復シナリオを作成し，回復ファイルにシナリオを保存できます。回復シナリオ・ウィザードを使用して回復シナリオを作成します。このウィザードは，回復シナリオの各段階の定義プロセスを案内します。回復ファイルに回復シナリオを保存し，特定のテストと関連付けます。

回復ファイルの作成

回復シナリオは，回復ファイルに保存します。回復ファイルは，複数の回復シナリオをまとめて整理，格納するのに便利です。新規の回復ファイルを作成することも，既存のファイルを編集することもできます。

回復ファイルを作成するには，次の手順を実行します。

1 ［**リソース**］ > ［**回復シナリオ マネージャ**］を選択します。［回復シナリオ マネージャ］ダイアログ・ボックスが開きます。



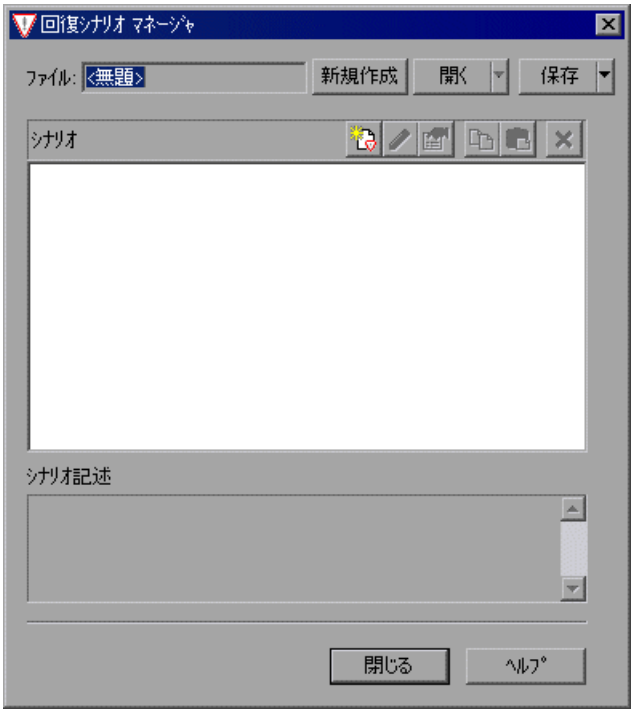
2 標準設定では，［回復シナリオ マネージャ］ダイアログ・ボックスには新規の回復ファイルが表示されます。この新規ファイルを使用するか，［**開く**］ボタンをクリックし，既存の回復ファイルを選択します。あるいは，［**開く**］ボタンの横にある矢印をクリックして，最近使用した回復ファイルをリストから選択します。

次の項で説明するように，回復シナリオ・ウィザードを使用して回復シナリオを作成し，回復ファイルに保存することができます。

[回復シナリオ マネージャ] ダイアログ・ボックスについて

[回復シナリオ マネージャ] ダイアログ・ボックスでは，回復ファイルの作成と編集，および回復シナリオの作成と管理ができます。

[回復シナリオ マネージャ] ダイアログ・ボックスには，現在開いている回復ファイルの名前，回復ファイルに保存したシナリオのリスト，および各シナリオの説明が表示されます。



［回復シナリオ マネージャ］ダイアログ・ボックスには，次のツールバー・ボタンがあります。

オプション	詳細
新規作成	新規の回復ファイルを作成します。詳細については，46 ページ「回復ファイルの作成」を参照してください。
開く	既存の回復ファイルを開きます。矢印をクリックして，最近使用した回復ファイルのリストから回復ファイルを選択することもできます。
保存	現在の回復ファイルを保存します。詳細については，72 ページ「回復ファイルへの回復シナリオの保存」を参照してください。
	回復シナリオ・ウィザードを開き，新規の回復シナリオを定義します。詳細については，50 ページ「回復シナリオ・ウィザードについて」を参照してください。
	選択した回復シナリオの回復シナリオ・ウィザードを開き，回復シナリオの設定を変更できます。詳細については，75 ページ「回復シナリオの変更」を参照してください。
	選択した回復シナリオのサマリ・プロパティを読み取り専用形式で表示します。詳細については，74 ページ「回復シナリオのプロパティの表示」を参照してください。
	回復シナリオを，開いている回復ファイルからクリップボードにコピーします。これによって，回復シナリオを別の回復ファイルに貼り付けることができます。詳細については，76 ページ「回復シナリオ・ファイル間での回復シナリオのコピー」を参照してください。
	回復シナリオを，クリップボードから開いている回復ファイルに貼り付けます。詳細については，76 ページ「回復シナリオ・ファイル間での回復シナリオのコピー」を参照してください。
	回復シナリオを削除します。詳細については，76 ページ「回復シナリオの削除」を参照してください。

注：各回復シナリオには，その種類ごとに異なるアイコンが関連付けられています。詳細については，73 ページ「回復シナリオの管理」を参照してください。

回復シナリオ・ウィザードについて

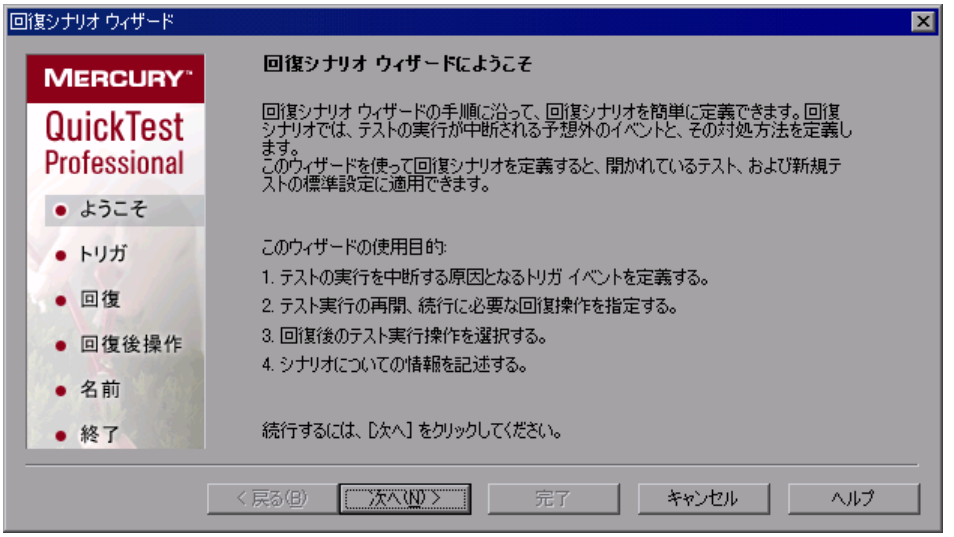
回復シナリオ・ウィザードでは，回復シナリオを作成するプロセスを段階的に案内します。回復シナリオ・ウィザードには，次の 5 つの主要なステップが含まれています。

- 実行セッションを中断するトリガ・イベントの定義
- 継続に必要な回復操作の指定
- 回復後のテスト実行の操作の選択
- 回復シナリオの名前と説明の指定
- 回復シナリオを現在のテストに関連付けるか，すべての新規テストに関連付けるかどうかの指定

回復シナリオ・ウィザードを開くには，［回復シナリオ マネージャ］ダイアログ・ボックス（［**リソース**］＞［**回復シナリオ マネージャ**］）の［**新規シナリオ**］ボタンをクリックします。

［回復シナリオ ウィザードにようこそ］画面

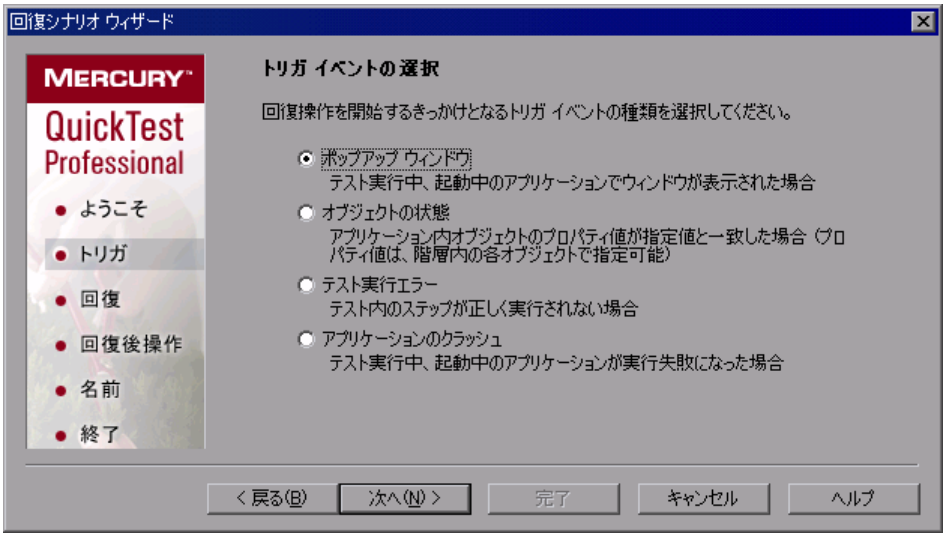
［回復シナリオ ウィザードにようこそ］画面には，回復シナリオ・ウィザードのさまざまなオプションについての一般的な情報と，回復シナリオの定義に関連する段階の概要が表示されます。



［**次へ**］をクリックし，［**トリガ イベントの選択**］画面に進みます。

［トリガ イベントの選択］画面

［トリガ イベントの選択］画面では，回復シナリオをトリガするイベントのタイプと，QuickTest によるイベントの認識方法を定義できます。



トリガのタイプを選択し，［**次へ**］をクリックします。ウィザードに表示される次の画面は，次のどのトリガのタイプを選択したかによって異なります。

- ［**ポップアップ ウィンドウ**］：QuickTest によって，ポップアップ・ウィンドウが検出され，ウィンドウのタイトルとテキストの内容に従ってそのウィンドウが識別されます。たとえば，実行セッション中にプリンタの用紙切れを示すメッセージ・ボックスが表示される場合があります。QuickTest では，実行セッションを続行するために，このウィンドウを検出し，定義済みの回復シナリオを呼び出すことができます。

  このオプションを選択し，［**次へ**］をクリックして，［**ポップアップ ウィンドウの条件を指定**］画面に進みます。

- ［**オブジェクトの状態**］：QuickTest によって，特定のテストのオブジェクト状態が検出され，そのプロパティ値とすべての祖先のプロパティ値に従ってオブジェクト状態が識別されます。オブジェクトはクラスではなく，プロパティ値によってのみ識別されます。

たとえば，特定のプロセスが開いている場合にダイアログ・ボックスの特定のボタンが無効になることがあります。QuickTest では，この問題のプロセスを開いているときに発生するボタンのオブジェクト・プロパティ状態を検出し，定義済みの回復シナリオを呼び出して，そのプロセスを閉じて実行セッションを続行できます。

このオプションを選択し，［**次へ**］をクリックして，［**オブジェクトの選択**］画面に進みます。

- ［**テスト実行エラー**］：QuickTest では，テストの実行エラーが検出され，メソッドからの失敗した戻り値によってエラーが識別されます。たとえば，QuickTest では，メニュー項目は実行セッション中の特定の場所では利用できないため，メソッド引数で指定されたメニュー項目を識別できない場合があります。QuickTest では，実行セッションを続行するために，この実行エラーを検出し，定義済みの回復シナリオを呼び出すことができます。

  このオプションを選択し，［**次へ**］をクリックして，［**テスト実行エラーの選択**］画面に進みます。

- ［**アプリケーションのクラッシュ**］：QuickTest によって，アプリケーション・クラッシュが検出され，定義済みのアプリケーションのリストに従って識別されます。たとえば，実行セッション中にステップを実行したときに，2 次的なアプリケーションがクラッシュすることが考えられます。使用中のアプリケーションの問題ではない可能性がある，このクラッシュによって実行セッションを失敗させないように，QuickTest では，このアプリケーションのクラッシュを検出し，定義済みの回復シナリオを呼び出して実行セッションを続行することができます。

  このオプションを選択し，［**次へ**］をクリックして［**プロセスの選択**］画面に進みます。

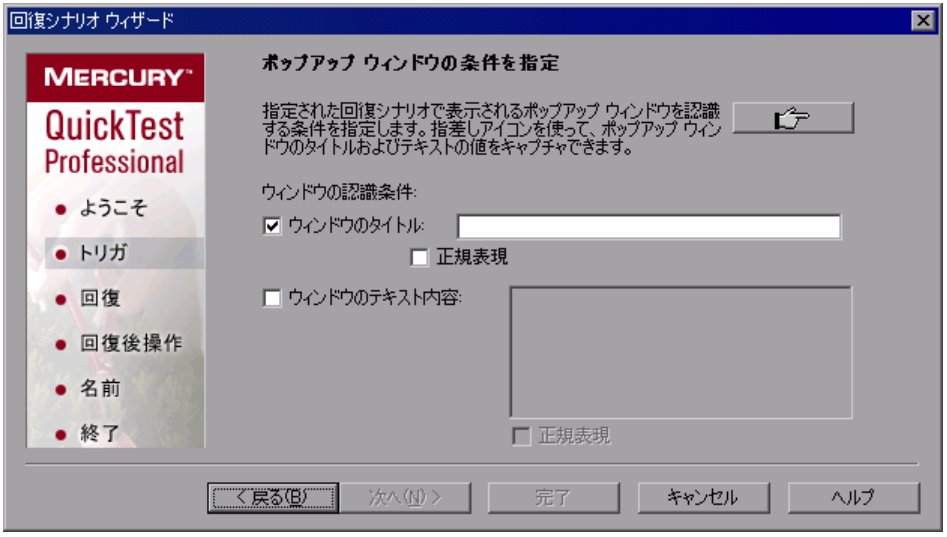
注：

一連の回復操作は，トリガ・イベントの条件に合致する状態が発生するたびに実行されます。たとえば，特定のオブジェクト状態を定義し，2 つのオブジェクトがこの状態と一致した場合，指定した状態に各オブジェクトが一致するたびに 1 回ずつ，合計 2 回の回復操作が実行されます。

回復メカニズムでは，テストの最後のステップで発生するトリガは処理されません。テストの最後のステップで発生する可能性のある予期しないイベントやエラーから回復する必要がある場合は，テストの最後にさらにステップを追加することで，予期しないイベントやエラーから回復できます。

［ポップアップ ウィンドウの条件を指定］画面

［トリガ イベントの選択］画面にある［**ポップアップ ウィンドウ**］トリガを選択した場合，［ポップアップ ウィンドウの条件を指定］画面が開きます。



次のいずれかを実行してポップアップ・ウィンドウの識別方法を指定します。

- ポップアップ・ウィンドウの［**ウィンドウのタイトル**］または［**ウィンドウのテキスト内容**］，あるいはその両方に基づいてポップアップ・ウィンドウを識別するかどうかを選択し，ポップアップ・ウィンドウの識別に使用するテキストを入力します。ウィンドウ・タイトルまたはテキスト内容に対して正規表現を使用するには，該当する［**正規表現**］チェック・ボックスを選択し，該当する場所に正規表現を入力します。正規表現の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 345 ページ「正規表現の使用について」を参照してください。
- 指差しマークをクリックした後ポップアップ・ウィンドウをクリックし，ウィンドウ・タイトルとウィンドウのテキスト内容をキャプチャします。

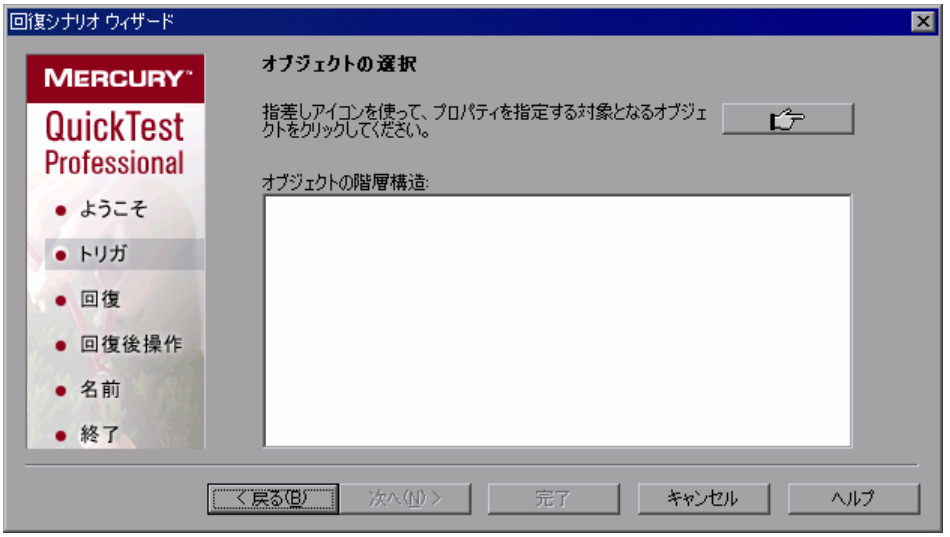
注：前者（［**ウィンドウのタイトル**］または［**ウィンドウのテキスト内容**］，あるいはその両方）を実行した場合，QuickTest は指定したタイトルまたはテキスト，あるいはその両方を含んでいる任意のポップアップ・ウィンドウを識別します。後者（指差しマーク）を実行した場合，QuickTest は選択したウィンドウのオブジェクト・プロパティ値に一致するポップアップ・ウィンドウのみを識別します。

ヒント：ウィンドウのフォーカスを変更したり，ショートカット・メニューを表示するために右クリックやマウスオーバーなどの操作を実行したりするには，CTRL キーを押しながら操作を行います。選択対象オブジェクトを含んでいるウィンドウが最小化されている場合は，左の CTRL キーを押したまま，Windows タスク・バー内のアプリケーションを右クリックして，ショートカット・メニューから［**元のサイズに戻す**］を選択することで，ウィンドウを表示できます。

［**次へ**］をクリックして，［**回復操作**］画面に進みます。

［オブジェクトの選択］画面

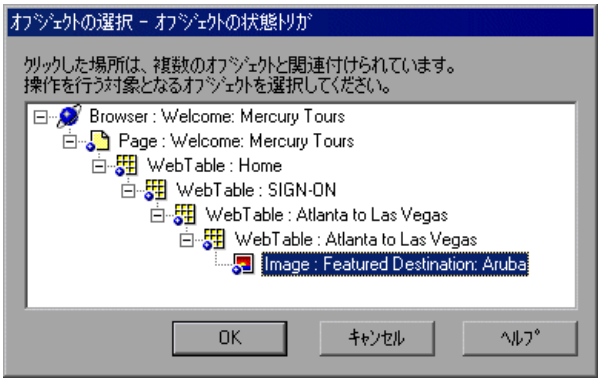
［トリガ イベントの選択］画面にある［**オブジェクトの状態**］トリガを選択すると，［オブジェクトの選択］画面が開きます。



指差しマークをクリックしてから，プロパティを指定するオブジェクトをクリックします。

ヒント：ウィンドウのフォーカスを変更したり，ショートカット・メニューを表示するために右クリックやマウスオーバーなどの操作を実行したりするには，CTRL キーを押しながら操作を行います。選択対象オブジェクトを含んでいるウィンドウが最小化されている場合は，左の CTRL キーを押したまま，Windows タスク・バー内のアプリケーションを右クリックして，ショートカット・メニューから［**元のサイズに戻す**］を選択することで，ウィンドウを表示できます。

クリックした場所が複数のオブジェクトに関連付けられている場合は，［オブジェクトの選択－オブジェクトの状態トリガ］ダイアログ・ボックスが開きます。



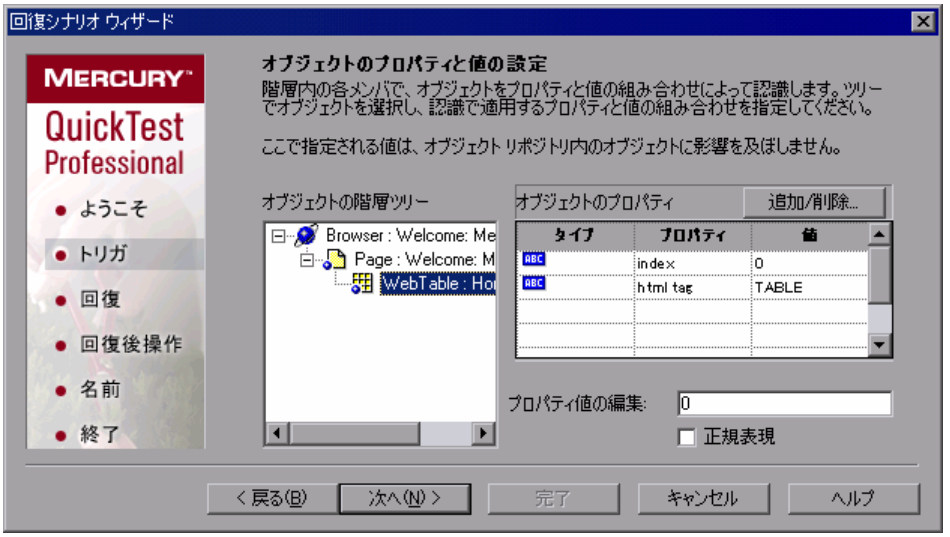
指定するプロパティを含んでいるオブジェクトを選択し，［**OK**］をクリックします。選択したオブジェクトとその親が［オブジェクトの選択］画面に表示されます。

注：階層オブジェクトの選択ツリーでは，Web テーブルなど，QuickTest が通常は記録しないオブジェクト（親以外のオブジェクト）を選択することもできます。

［**次へ**］をクリックして，［**オブジェクトのプロパティと値の設定**］画面に進みます。

［オブジェクトのプロパティと値の設定］画面

［オブジェクトの選択］画面でプロパティを指定するオブジェクトを選択すると，［オブジェクトのプロパティと値の設定］画面が開きます。



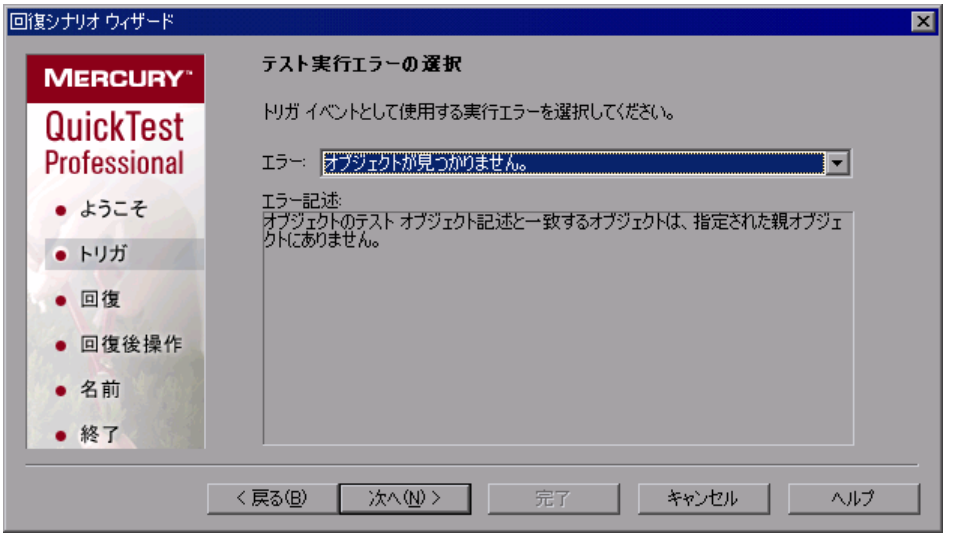
階層の各オブジェクトに対して，［**プロパティ値の編集**］ボックスで，オブジェクトの識別に使用するプロパティ値を変更できます。［**追加 / 削除**］ボタンをクリックし，検査対象のプロパティ値のリストからオブジェクト・プロパティを追加または削除することもできます。オブジェクトはクラスではなく，プロパティ値によってのみ識別されます。

プロパティ値で正規表現を使用する場合は，［**正規表現**］チェック・ボックスを選択します。正規表現の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 345 ページ「正規表現の使用について」を参照してください。

［**次へ**］をクリックして，［**回復操作**］画面に進みます。

［テスト実行エラー］画面

［トリガ イベントの選択］画面にある［**テスト実行エラー**］トリガを選択すると，［テスト実行エラーの選択］画面が開きます。



［**エラー**］リストでは，次のように，トリガ・イベントとして使用するテストの実行エラーを選択します。

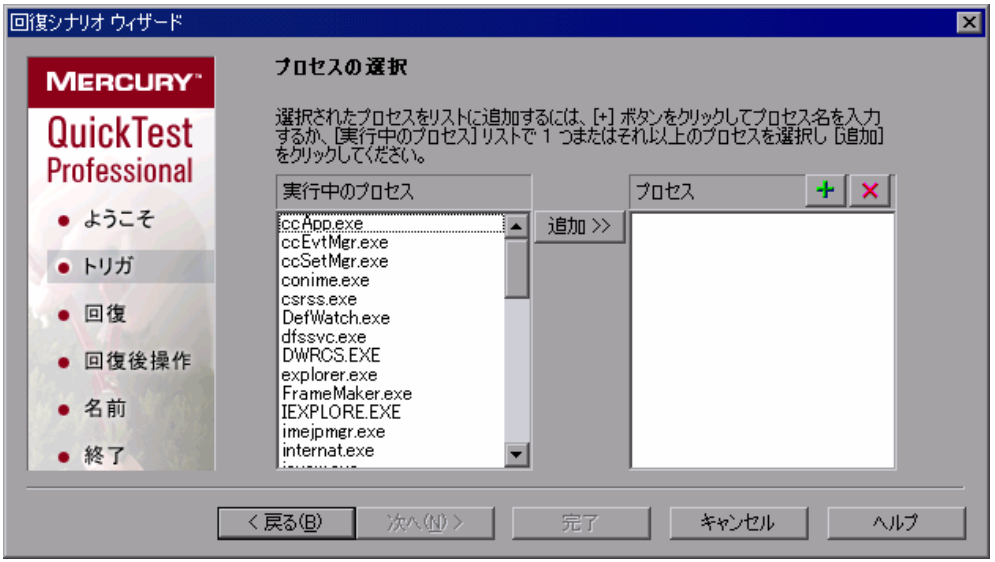
- ［**任意のエラー**］：テスト・オブジェクト・メソッドによって返されるすべてのエラー・コード。
- ［**リストまたはメニュー内の項目が一意ではありません**］：リスト，メニュー，またはツリーにある複数の項目に，メソッド引数で指定されている名前がある場合に発生します。
- ［**リストまたはメニュー内の項目が見つかりません**］：メソッド引数で指定されているリスト，メニュー，またはツリー項目が QuickTest によって識別できない場合に発生します。この原因としては，その項目を現在利用できない，または名前が変更されていることが考えられます。
- ［**物理記述に対して複数のオブジェクトが該当します**］：アプリケーションの複数のオブジェクトが，ステップで指定されているオブジェクトのテスト・オブジェクト記述で指定されている値と同一のプロパティ値を有する場合に発生します。

- ［**オブジェクトが無効になっています**］：ステップで指定されているオブジェクトが現在無効となっているため，QuickTest がステップを実行できない場合に発生します。
- ［**オブジェクトが見つかりません**］：指定された親オブジェクト内に，オブジェクトのテスト・オブジェクト記述と一致するオブジェクトがない場合に発生します。
- ［**オブジェクトが非表示になっています**］：ステップで指定されたオブジェクトが現在画面上に表示されていないため，QuickTest がステップを実行できない場合に発生します。

［**次へ**］をクリックして，［**回復操作**］画面に進みます。

［プロセスの選択］画面

［トリガ イベントの選択］画面にある［**アプリケーションのクラッシュ**］トリガを選択した場合，［プロセスの選択］画面が開きます。



［**実行中のプロセス**］リストには，現在実行中のすべてのアプリケーション・プロセスが表示されます。［**プロセス**］リストには，クラッシュした場合に回復シナリオをトリガするアプリケーション・プロセスが表示されます。

アプリケーション・プロセスを［**プロセス**］リストに入力するか，［**実行中のプロセス**］リストからアプリケーション・プロセスを選択することによって，［**プロセス**］リストにアプリケーション・プロセスを追加できます。

［**実行中のプロセス**］リストからプロセスを追加するには，［**実行中のプロセス**］リストのプロセスをダブルクリックするか，プロセスを選択して［**追加**］ボタンをクリックします。Windows で複数選択する際の標準的な方法（CTRL キーや SHIFT キー）を使用して，複数のプロセスを選択できます。

［**プロセス**］リストにプロセスを直接追加するには，［**新規プロセスの追加**］ボタンをクリックし，リストに追加するプロセスの名前を入力します。

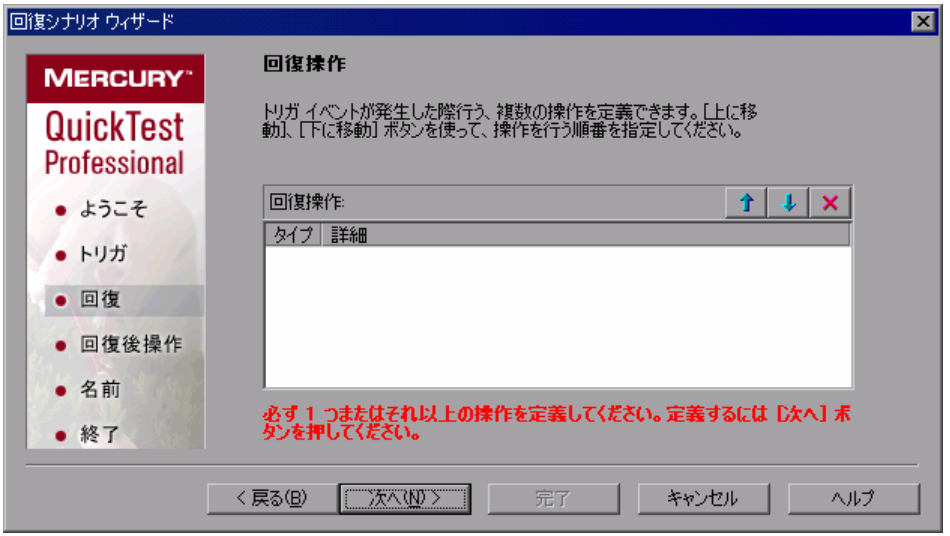
［**プロセス**］リストからプロセスを削除するには，プロセスを選択して［**プロセスの削除**］ボタンをクリックします。

ヒント：プロセスの名前を変更するには，［**プロセス**］リストでプロセスを選択し，プロセス名をクリックして編集します。

［**次へ**］をクリックして，［**回復操作**］画面に進みます。

［回復操作］画面

［回復操作］画面では，回復シナリオにおける一連の回復操作を管理できます。回復操作とは，QuickTest によってトリガ・イベントが認識されたときに順次実行される操作です。



少なくとも 1 つの回復操作を定義する必要があります。回復操作を定義し，それを［**回復操作**］リストに追加するには，[**次へ**] をクリックして，［**回復操作**］画面に進みます。

2 つ以上の回復操作を定義する場合，回復操作を選択し，［**上に移動**］または［**下に移動**］ボタンを使用して，QuickTest による回復操作の実行順序を変更できます。回復操作を選択し，［**削除**］ボタンをクリックして，回復シナリオから回復操作を削除することもできます。

注：［**Microsoft Windows の再起動**］回復操作を定義する場合，常に最後の回復操作としてこの操作が挿入されるため，リストで位置は変更できません。

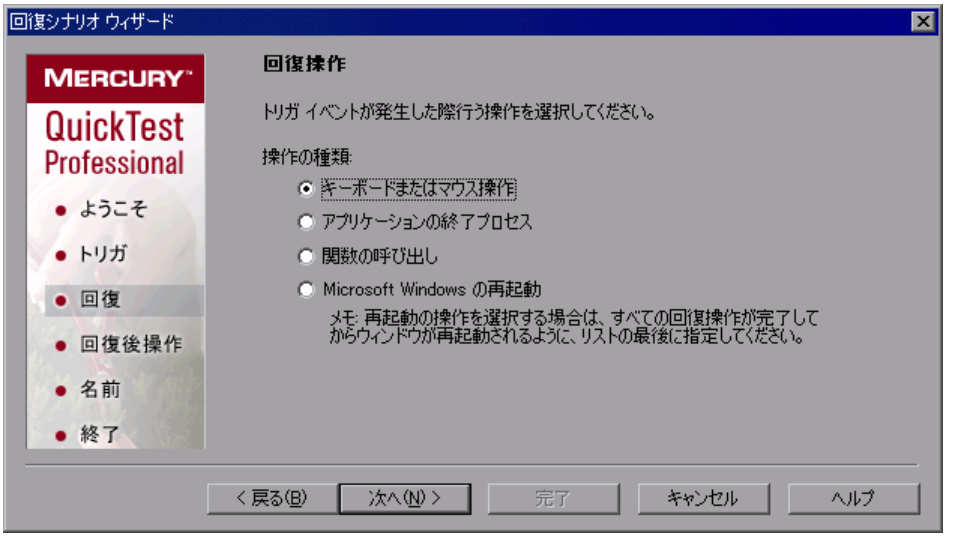
回復操作を 1 つでも定義すると，［**他の回復操作を追加する**］チェック・ボックスが表示されます。

➤ 他の回復操作を定義するには，このチェック・ボックスを選択し，［**次へ**］をクリックします。

➤ ［**他の回復操作を追加する**］チェック・ボックスをクリアし，［**次へ**］をクリックして，［**回復後操作**］画面に進みます。

［回復操作］画面

［回復操作］画面では，QuickTest によるトリガ・イベント検出後に実行される操作を指定できます。



回復操作のタイプを選択し，［**次へ**］をクリックします。ウィザードに表示される次の画面は，選択する回復操作のタイプに応じて異なります。

次のタイプの回復操作を定義できます。

➤ ［**キーボードまたはマウス操作**］：QuickTest によって，ウィンドウのボタンをクリックする操作，またはキーボードのキーを押す操作がシミュレートされます。このオプションを選択し，［**次へ**］をクリックして［**回復操作ーボタン、またはキーを押す**］画面に進みます。

➤ ［**アプリケーションの終了プロセス**］：QuickTest によって，指定のプロセスが閉じます。このオプションを選択し，［**次へ**］をクリックして［**回復操作ープロセスの終了**］画面に進みます。

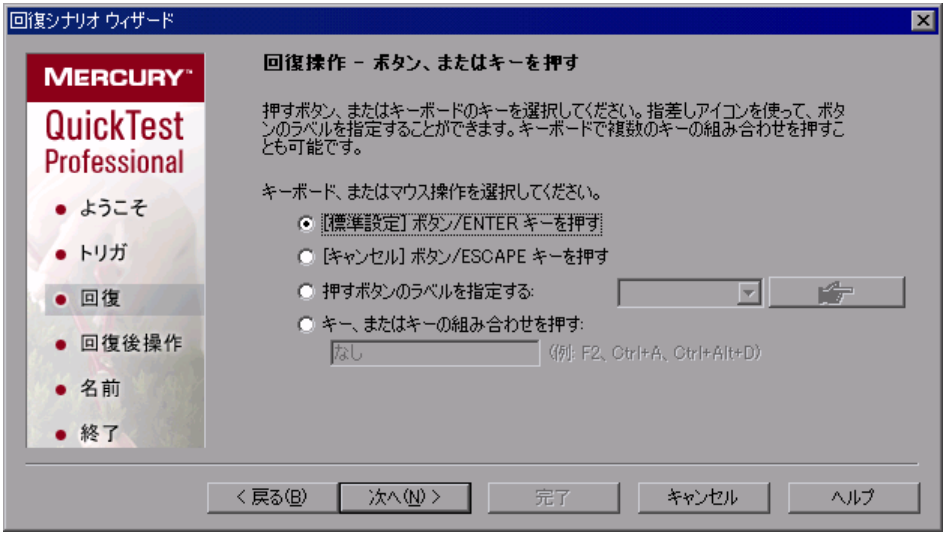
➤ ［**関数の呼び出し**］: QuickTest によって，VBScript 関数が呼び出されます。このオプションを選択し，［**次へ**］をクリックして［**回復操作ー関数の呼び出し**］画面に進みます。

➤ ［**Microsoft Windows の再起動**］: QuickTest によって Microsoft Windows が再起動されます。このオプションを選択し，［**次へ**］をクリックして［**回復操作**］画面に進みます。

注：［**Microsoft Windows の再起動**］回復操作を使用する場合，操作を実行する前に，この回復シナリオと関連付けられているテストをすべて保存する必要があります。また，再起動時に自動ログインするように，テストを実行するコンピュータを設定する必要があります。

［回復操作ーボタン、またはキーを押す］画面

［回復操作］画面で［**キーボードまたはマウス操作**］回復操作を選択した場合，［回復操作ーボタン、またはキーを押す］画面が開きます。



QuickTest によってトリガ・イベントが検出された場合に実行させる，キーボードまたはマウスの操作を指定します。

- [**[標準設定] ボタン /ENTER キーを押す**]：トリガが発生した場合に，表示されているウィンドウで，標準のボタンをクリックする，または ENTER キーを押すよう QuickTest に指示します。
- [**[キャンセル] ボタン /ESCAPE キーを押す**]：トリガが発生した場合に，表示されているウィンドウで，[**キャンセル**] ボタンをクリックする，または ESCAPE キーを押すよう QuickTest に指示します。
- [**押すボタンのラベル指定する**]：トリガが発生した場合に，表示されているウィンドウで，指定したラベルの付いたボタンをクリックするよう QuickTest に指示します。このオプションを選択した場合は，指差しマークをクリックした後，トリガ・ウィンドウの中の任意の場所をクリックします。

ヒント：ウィンドウのフォーカスを変更したり，ショートカット・メニューを表示するために右クリックやマウスオーバーなどの操作を実行したりするには，CTRL キーを押しながら操作を行います。選択対象オブジェクトを含んでいるウィンドウが最小化されている場合は，左の CTRL キーを押したまま，Windows タスク・バー内のアプリケーションを右クリックして，ショートカット・メニューから [**元のサイズに戻す**] を選択することで，ウィンドウを表示できます。

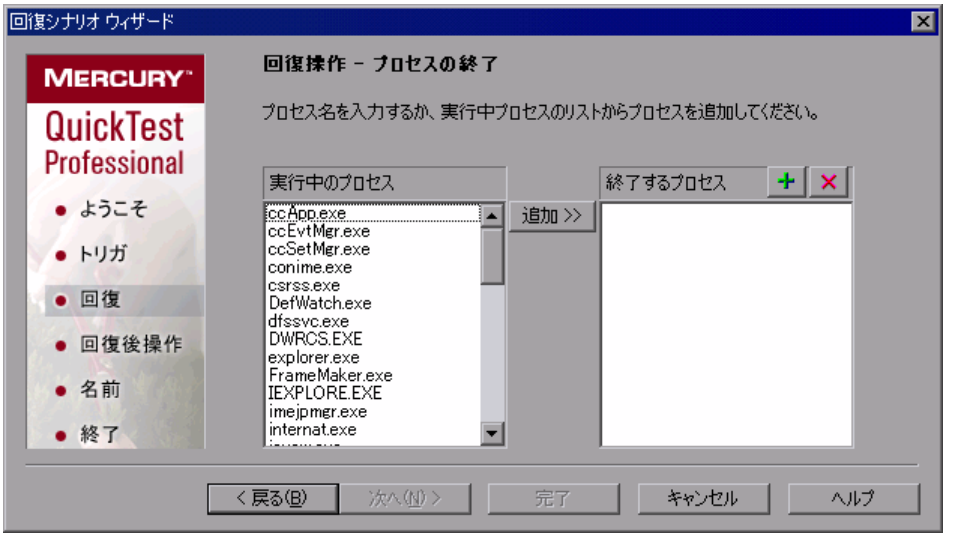
選択されたウィンドウのボタン・ラベルはすべて，リスト・ボックスに表示されます。リストから必要なボタンを選択します。

- [**キー、またはキーの組み合わせを押す**]：トリガが発生した場合に，表示されているウィンドウで，指定したキーボードのキーまたはキーの組み合わせを押すよう QuickTest に指示します。このオプションを選択した場合は，エディット・ボックスをクリックした後，指定する単独のキーまたはキーの組み合わせを押します。

  [**次へ**] をクリックします。[回復操作] 画面が再び開き，定義したキーボードまたはマウスの回復操作が表示されます。

[回復操作－プロセスの終了] 画面

[回復操作] 画面の [**アプリケーションの終了プロセス**] 回復操作を選択すると，[回復操作－プロセスの終了] 画面が開きます。



[**実行中のプロセス**] リストには，現在実行中のすべてのアプリケーション・プロセスが表示されます。[**終了するプロセス**] リストには，トリガが呼び出されたときに閉じるアプリケーション・プロセスが表示されます。

[**実行中のプロセス**] リストからプロセスを追加するには，[**実行中のプロセス**] リストのプロセスをダブルクリックするか，プロセスを選択して [**追加**] ボタンをクリックします。Windows で複数選択する際の標準的な方法（CTRL キーや SHIFT キー）を使用して，複数のプロセスを選択できます。

[**終了するプロセス**] リストにプロセスを直接追加するには，[**新規プロセスの追加**] ボタンをクリックし，リストに追加するプロセスの名前を入力します。

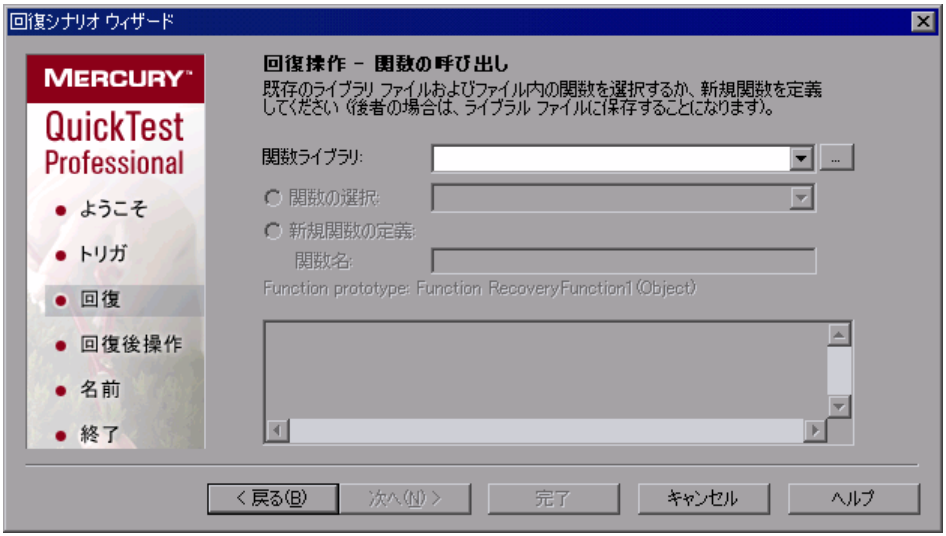
[**終了するプロセス**] リストからプロセスを削除するには，プロセスを選択して [**プロセスの削除**] ボタンをクリックします。

ヒント：プロセスの名前を変更するには，［**終了するプロセス**］リストでプロセスを選択し，プロセス名をクリックして編集します。

［**次へ**］をクリックします。［回復操作］画面が再び開き，定義した「プロセスを閉じる」回復操作が表示されます。

［回復操作－関数の呼び出し］画面

［回復操作］画面の［**関数の呼び出し**］回復操作を選択すると，［回復操作－関数の呼び出し］画面が開きます。



［**関数ライブラリ**］ボックスで，最近指定した関数ライブラリを選択します。あるいは，参照ボタンをクリックして，既存の関数ライブラリに移動します。

注：QuickTest では，選択した関数ライブラリが，テストと自動的に関連付けられます。したがって，［テストの設定］ダイアログ・ボックスの［リソース］タブで，それぞれ関数ライブラリとテストを関連付ける必要はありません。

関数ライブラリを選択した後，次のオプションのいずれか 1 つを選択します。

➤ ［**関数の選択**］：選択した関数ライブラリから既存の関数を選択します。

注 : ［トリガ イベントの選択］画面で選択したトリガ・タイプのプロトタイプ構文と一致する関数のみが表示されます。各トリガ・タイプのプロトタイプを次に示します。

テスト実行エラーのトリガ
OnRunStep
(
[in] Object as Object: 現在のステップのオブジェクト。
[in] Method as String: 現在のステップのメソッド。
[in] Arguments as Array: 実際のメソッドの引数。
[in] Result as Integer: 実際のメソッドの結果。
)

ポップアップ・ウィンドウとオブジェクトの状態のトリガ
OnObject
(
[in] Object as Object: 検出されたオブジェクト。
)

アプリケーションのクラッシュのトリガ
OnProcess
(
[in] ProcessName as String: 検出されたプロセスの名前。
[in] ProcessId as Integer: 検出されたプロセスの ID。
)

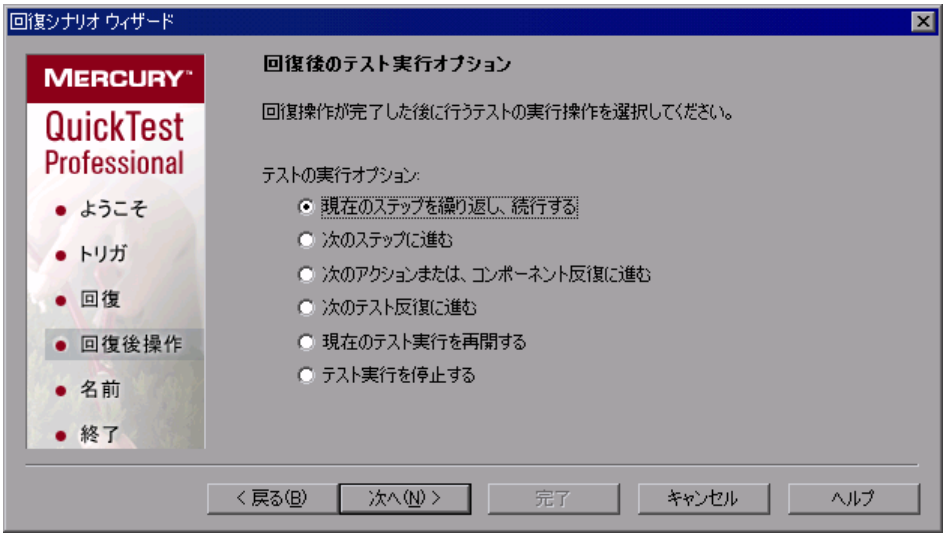
➤ ［**新規関数の定義**］：一意の名前を指定し，表示される関数のプロトタイプに従って［**関数名**］ボックスで関数を定義することによって，新規の関数を作成します。新しい関数が，選択した関数ライブラリに追加されます。

注：複数のシナリオで異なる関数ライブラリから同一の名前を持つ関数を使用すると，回復プロセスは失敗することがあります。この場合，実行セッション中に回復の失敗に関する情報が表示されます。

［**次へ**］をクリックします。［回復操作］画面が再び開き，定義した関数の操作が表示されます。

［回復後のテスト実行オプション］画面

［回復操作］画面の［**他の回復操作を追加する**］チェック・ボックスをクリアし，［**次へ**］をクリックすると，［回復後のテスト実行オプション］画面が開きます。回復後のテスト実行のオプションでは，QuickTest によってイベントが識別され，指定の回復オプションがすべて実行された後の，実行セッションの継続方法を指定します。



定義した回復操作が実行された後，QuickTest では，次の実行セッションのオプションのいずれか 1 つを実行できます。

➤ **［現在のステップを繰り返し、続行する］**

この現在のステップとは，回復シナリオがトリガされた際に QuickTest が実行中であったステップです。回復シナリオに対して「**エラー発生時**」呼び出しオプションを使用している場合，通常，エラーを返すステップは，トリガ・イベントの発生原因となったステップより 1 つ以上後のステップです。

したがってほとんどの場合，現在のステップを繰り返しても，トリガ・イベントは再現されません。詳細については，81 ページ「回復シナリオの有効化と無効化」を参照してください。

➤ **［次のステップに進む］**

回復シナリオがトリガされた際に QuickTest が実行中であったステップをスキップします。アプリケーションに対して操作を実行するステップをスキップすると，それ以降のステップが失敗する場合があることに注意してください。

➤ **［次のアクションまたはコンポーネント反復に進む］**

現在のアクション反復内のステップの実行を停止し，次のアクション反復を先頭から開始します（または現在のアクション内でほかに反復を実行する必要がなければ，次のアクションを開始します)。

➤ **［次のテスト反復に進む］**

現在のアクション内のステップの実行を停止し，次のテスト反復を先頭から開始します（または現在のアクション内でほかに反復を実行する必要がなければ，テストの実行を停止します)。

➤ **［現在のテスト実行を再開する］**

ステップの実行を停止し，テストを始めから再実行します。

➤ **［テストの実行を停止する］**

テストの実行を停止します。

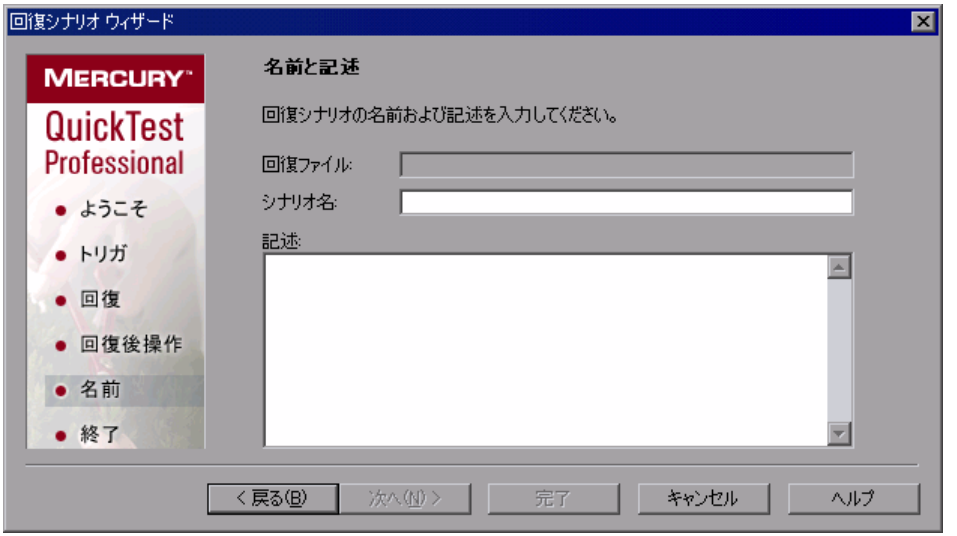
注：回復操作として［**Microsoft Windows の再起動**］を選択した場合，上記の最後 2 つのテスト実行のオプションのみから選択できます。

テスト実行のオプションを選択し，［**次へ**］をクリックして，［**名前と記述**］画面に進みます。

［名前と記述］画面

［回復後のテスト実行オプション］画面でテスト実行のオプションを指定し，［**次へ**］をクリックすると，［名前と記述］画面が開きます。

［名前と記述］画面では，回復シナリオを識別するための名前を指定します。シナリオに関する説明情報を追加することもできます。

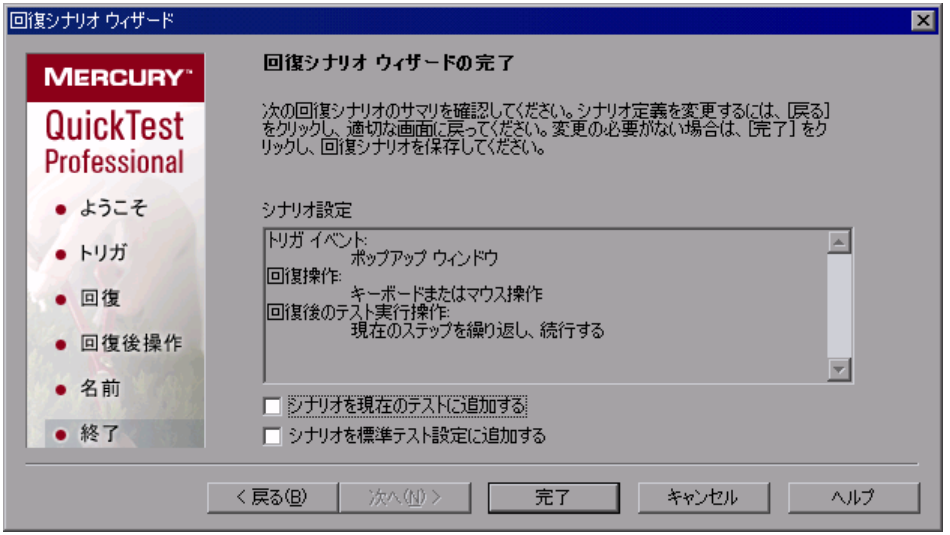


回復シナリオの名前とテキスト形式の説明を入力し，［**次へ**］をクリックして，［**回復シナリオ ウィザードの完了**］画面に進みます。

[回復シナリオ ウィザードの完了] 画面

[名前と記述] 画面で回復シナリオの名前と説明を指定し，[**次へ**] をクリックすると，[回復シナリオ ウィザードの完了] 画面が開きます。

[回復シナリオ ウィザードの完了] 画面では，定義したシナリオ設定の概要を確認できます。回復シナリオを自動的に現在のテストに関連付けるかどうか，または新しいすべてのテストの標準設定に回復シナリオを追加するかどうかを指定できます。



[**シナリオを現在のテストに追加する**] チェック・ボックスを選択すると，この回復シナリオを現在のテストに関連付けることができます。[**完了**] をクリックすると，QuickTest によって，[テストの設定] ダイアログ・ボックスの [回復] タブにある [**シナリオ**] リストに回復シナリオが追加されます。

この回復シナリオを，新しく作るすべてのテストの標準設定のシナリオにするには，[**シナリオを標準テスト設定に追加する**] チェック・ボックスを選択します。次回テストを作成する際には，[テストの設定] ダイアログ・ボックスの [回復] タブの [**シナリオ**] リストに，このシナリオが表示されます。

注：シナリオは標準シナリオのリストから削除できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 772 ページ「テストのための回復シナリオ設定の定義」を参照してください。

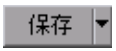
［**完了**］をクリックすると，回復シナリオの定義は完了します。

回復ファイルへの回復シナリオの保存

回復シナリオ・ウィザードを使用して，回復ファイル内の回復シナリオの作成または変更を行ったら，その回復ファイルを保存する必要があります。

新規の回復ファイルまたは変更した回復ファイルを保存するには，次の手順を実行します。

1 ［**保存**］ボタンをクリックします。既存の回復ファイル内のシナリオの追加または変更を行うと，回復ファイルとそのシナリオが保存されます。新規の回復ファイルを使用している場合は，［添付の保存］ダイアログ・ボックスが表示されます。



ヒント：［**保存**］ボタンの右側の矢印をクリックし，［**名前を付けて保存**］を選択して，別の名前で回復ファイルを保存することもできます。

---

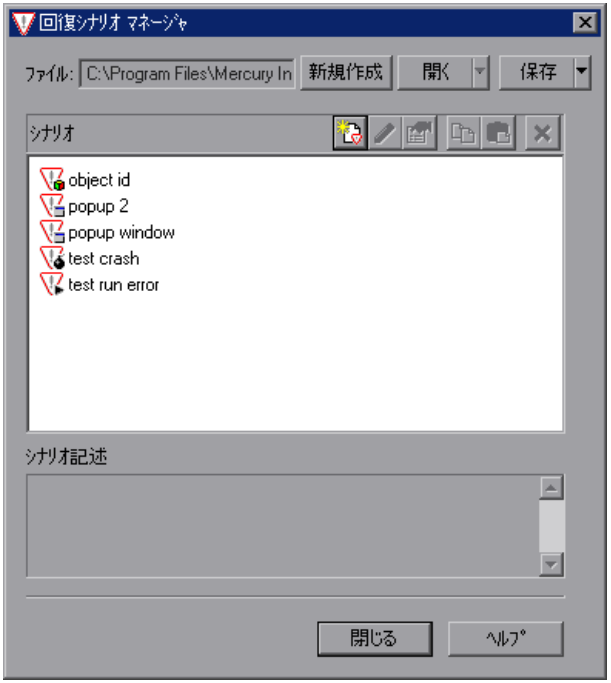
2 ファイルの保存先フォルダを選択します。

3 ［**ファイル名**］ボックスにファイルの名前を入力します。回復ファイルは，ファイル拡張子 **.qrs** 付きで，指定の場所に保存されます。

ヒント：回復ファイルを保存せずに［回復シナリオ マネージャ］ダイアログ・ボックスの［**閉じる**］ボタンをクリックすると，回復ファイルを保存するよう求められます。［**はい**］をクリックし，上記のステップ 2 に進みます。既存の回復ファイル内のシナリオの追加または変更を行い，メッセージに対して［**はい**］をクリックすると，回復ファイルとそのシナリオは保存されます。

回復シナリオの管理

回復シナリオを作成したら，回復シナリオ・マネージャを使用してシナリオを管理できます。



回復シナリオ・マネージャには，次の回復シナリオ・アイコンがあります。

アイコン	詳細
	実行セッション中，開いているアプリケーションでウィンドウがポップアップしたときに回復シナリオがトリガされることを示します。
	対象回復シナリオが，アプリケーション内のオブジェクトのプロパティ値が特定の値に一致したときに起動されることを示します。
	テストのステップが正しく実行されないときに回復シナリオがトリガされることを示します。
	実行セッション中に，開いているアプリケーションが失敗したときに回復シナリオがトリガされることを示します。

回復シナリオ・マネージャでは，次の方法で既存のシナリオを管理できます。

- 回復シナリオのプロパティの表示
- 回復シナリオの変更
- 回復シナリオの削除
- 回復シナリオ・ファイル間での回復シナリオのコピー

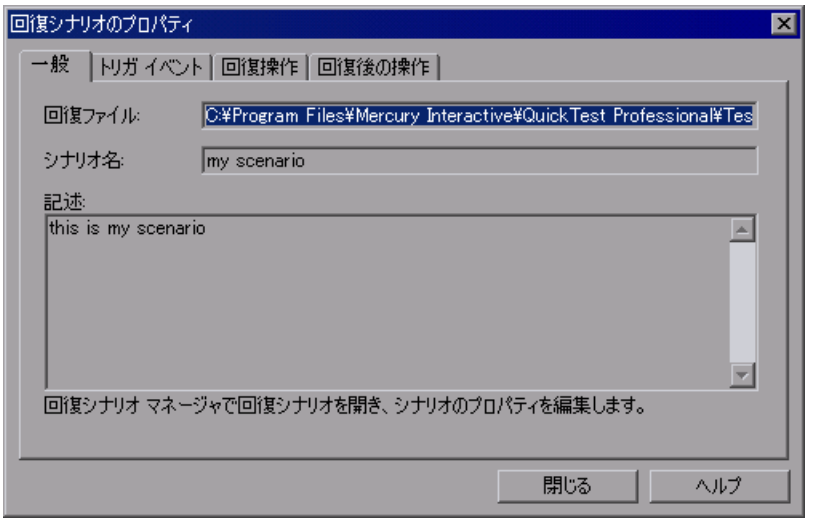
回復シナリオのプロパティの表示

定義済みのすべての回復シナリオのプロパティを表示できます。

回復シナリオのプロパティを表示するには，次の手順を実行します。

1 ［**シナリオ**］ボックスで，プロパティを表示する回復シナリオを選択します。

2 ［**プロパティ**］ボタンをクリックします。あるいは，［**シナリオ**］ボックスのシナリオをダブルクリックする方法もあります。［回復シナリオのプロパティ］ダイアログ・ボックスが開きます。



［回復シナリオのプロパティ］ダイアログ・ボックスには，選択したシナリオに関する，次の読み取り専用の情報が表示されます。

- ［**一般**］タブ：回復シナリオに対して定義されている名前と説明に加えて，シナリオが保存されている回復ファイルの名前とパスが表示されます。
- ［**トリガ イベント**］タブ：回復シナリオに対して定義されているトリガ・イベントの設定が表示されます。
- ［**回復操作**］タブ：回復シナリオに対して定義されている回復操作が表示されます。
- ［**回復後の操作**］タブ：回復シナリオに対して定義されている回復後の操作が表示されます。

回復シナリオの変更

既存の回復シナリオの設定を変更できます。

回復シナリオを変更するには，次の手順を実行します。

1 ［**シナリオ**］ボックスで，変更するシナリオを選択します。

2 ［**編集**］ボタンをクリックします。回復シナリオ・ウィザードが開き，選択した回復シナリオに対して定義した設定が表示されます。

3 回復シナリオ・ウィザードを操作して，必要に応じて詳細を変更します。回復シナリオ・ウィザードのオプションの詳細については，46 ページ「回復シナリオの定義」を参照してください。

注：行った変更は，［回復シナリオ マネージャ］ダイアログ・ボックスで［**保存**］をクリックするまで保存されません。変更を保存せずに［回復シナリオ マネージャ］ダイアログ・ボックスの［**閉じる**］ボタンをクリックすると，回復ファイルを保存するよう求められます。［**はい**］をクリックし，変更を保存します。

回復シナリオの削除

必要のない既存の回復シナリオは削除できます。回復シナリオ・マネージャから回復シナリオを削除すると，回復シナリオ・ファイルからは対応する情報も削除されます。

注：削除した回復シナリオがテストと関連付けられている場合，QuickTest では，実行セッション中はその回復シナリオが無視されます。

回復シナリオを削除するには，次の手順を実行します。

1 ［**シナリオ**］ボックスで，削除するシナリオを選択します。

2 ［**削除**］ボタンをクリックします。［回復シナリオ マネージャ］ダイアログ・ボックスから，回復シナリオが削除されます。

注：シナリオは，［回復シナリオ マネージャ］ダイアログ・ボックスで［**保存**］をクリックするまで実際には削除されません。削除を保存せずに［回復シナリオ マネージャ］ダイアログ・ボックスの［**閉じる**］ボタンをクリックすると，回復ファイルを保存するよう求められます。［**はい**］をクリックして回復シナリオ・ファイルを保存し，シナリオを削除します。

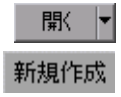
回復シナリオ・ファイル間での回復シナリオのコピー

ある回復シナリオ・ファイルから別の回復シナリオ・ファイルに，回復シナリオをコピーできます。

ある回復シナリオ・ファイルから別の回復シナリオ・ファイルに回復シナリオをコピーするには，次の手順を実行します。

1 ［**シナリオ**］ボックスで，コピーする回復シナリオを選択します。

2 ［**コピー**］ボタンをクリックします。シナリオがクリップボードにコピーされます。



3 [**開く**] ボタンをクリックして，シナリオのコピー先となる回復シナリオ・ファイルを選択するか，[**新規作成**] ボタンをクリックして，シナリオのコピー先となる回復シナリオ・ファイルを新規作成します。

4 [**貼り付け**] ボタンをクリックします。シナリオが新規の回復シナリオ・ファイルにコピーされます。

注：

回復シナリオ・ファイルに同じ名前のシナリオがすでに存在する場合，それをコピーした新しいシナリオで置き換えるかどうかを選択できます。

変更は，[回復シナリオ マネージャ] ダイアログ・ボックスで [**保存**] をクリックするまで保存されません。変更を保存せずに [回復シナリオ マネージャ] ダイアログ・ボックスの [**閉じる**] ボタンをクリックすると，回復ファイルを保存するよう求められます。[**はい**] をクリックし，変更を保存します。

テスト用の回復シナリオ・リストの設定

回復シナリオを作成した後は，トリガ・イベントが発生した場合，実行セッション中に QuickTest によって適切なシナリオが実行されるように，作成した回復シナリオと選択したテストまたはコンポーネントを関連付けます。シナリオに優先順位を付け，実行セッション中のシナリオの適用順序を設定できます。テストに関連付けられている特定のシナリオ，またはすべてのシナリオを無効にすることもできます。すべての新しいテスト用の標準シナリオとして，どの回復シナリオを使用するかも定義できます。

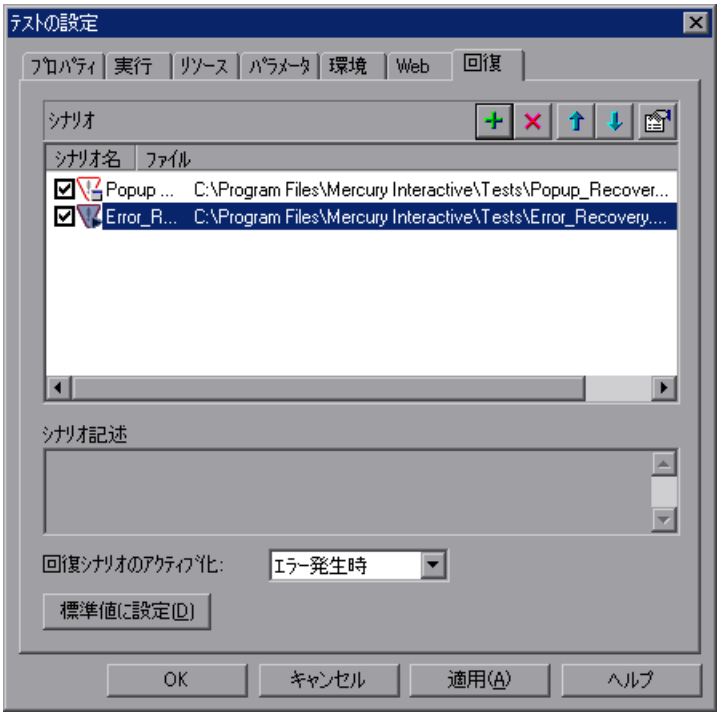
テストへの回復シナリオの追加

回復シナリオを作成した後は，トリガ・イベントが発生すると，実行セッション中に回復シナリオを実行するよう QuickTest に指示するために，1 つ以上のシナリオをテストと関連付けることができます。[テストの設定] ダイアログ・ボックスの [回復] タブには，現在のテストに関連付けられているすべての回復シナリオが表示されます。

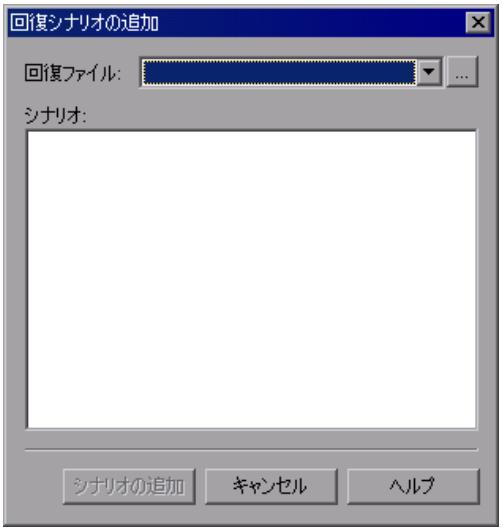
ヒント：トリガ・イベントが発生すると，QuickTest によって，［回復］タブに表示されている順序に従って該当する回復シナリオが確認されます。80 ページ「回復シナリオの優先順位の設定」で説明されている手順で，この順序を変更できます。

回復シナリオをテストに追加するには，次の手順を実行します。

1 ［**ファイル**］＞［**設定**］を選択します。［テストの設定］ダイアログ・ボックスが開きます。［**回復**］タブを選択します。



2 [**追加**] ボタンをクリックします。[回復シナリオの追加] ダイアログ・ボックスが開きます。



3 [**回復ファイル**] ボックスで，テストと関連付ける回復シナリオが含まれている回復ファイルを選択します。あるいは，参照ボタンをクリックして，選択する回復ファイルに移動します。[**シナリオ**] ボックスには，選択したファイルに保存されているシナリオの名前が表示されます。

4 [**シナリオ**] ボックスで，テストと関連付けるシナリオを選択し，[**シナリオの追加**] をクリックします。[回復シナリオの追加] ダイアログ・ボックスが閉じ，選択したシナリオが [回復] タブの [**シナリオ**] リストに追加されます。

ヒント：パスを 1 回クリックして強調表示した後，再度パスをクリックして編集モードに入ることで，回復シナリオ・ファイルのパスを編集できます。たとえば，絶対ファイル・パスを相対ファイル・パスに変更する場合などにこの操作を行います。回復シナリオ・ファイルのパスを変更する場合，テストを実行する前に，回復シナリオが新しいパスの位置に定義されていることを確認する必要があります。

回復シナリオのプロパティの表示

テストと関連付けられているすべての回復シナリオのプロパティを表示できます。

注：［回復シナリオ マネージャ］ダイアログ・ボックスから，回復シナリオの設定を変更します。詳細については，75 ページ「回復シナリオの変更」を参照してください。

回復シナリオのプロパティを表示するには，次の手順を実行します。

1 ［**シナリオ**］ボックスで，プロパティを表示する回復シナリオを選択します。

2 ［**プロパティ**］ボタンをクリックします。あるいは，［**シナリオ**］ボックスのシナリオをダブルクリックする方法もあります。［回復シナリオのプロパティ］ダイアログ・ボックスが開き，選択したシナリオの設定に関する読み取り専用の情報が表示されます。詳細については，74 ページ「回復シナリオのプロパティの表示」を参照してください。

回復シナリオの優先順位の設定

実行セッション中に，関連付けられたシナリオが QuickTest によって実行される順序を指定できます。トリガ・イベントが発生した場合，QuickTest によって，［テストの設定］ダイアログ・ボックスの［回復］タブに表示されている順序に従って該当する回復シナリオが確認されます。

回復シナリオの優先順位を設定するには，次の手順を実行します。

1 ［**シナリオ**］ボックスで，優先順位を変更するシナリオを選択します。

2 ［**上に移動**］ボタンまたは［**下に移動**］ボタンをクリックします。選択に従って，選択したシナリオの優先順位が変更されます。

3 優先順位を変更するシナリオごとに 1 ～ 2 の手順を繰り返します。

テストからの回復シナリオの削除

［テストの設定］ダイアログ・ボックスの［回復］タブを使用して，特定のシナリオとテストの間の関連付けを削除できます。テストからシナリオを削除した後も，まだシナリオ自体は存在していますが，QuickTest によって実行セッション中にシナリオが実行されることはありません。

回復シナリオをテストから削除するには，次の手順を実行します。

1 ［**シナリオ**］ボックスで，削除するシナリオを選択します。

2 ［**削除**］ボタンをクリックします。選択したシナリオは，テストとの関連付けが解除されます。

回復シナリオの有効化と無効化

［テストの設定］ダイアログ・ボックスの［回復］タブでは，特定のシナリオを有効または無効にしたり，QuickTest による回復シナリオのメカニズムの呼び出し条件を指定したりできます。特定のシナリオを無効にしても，テストとの関連付けは残りますが，実行セッション中にそのシナリオが QuickTest によって実行されることはありません。そのシナリオは後で有効にできます。

また，回復シナリオを呼び出す条件を指定することもできます。

特定の回復シナリオを有効 / 無効にするには，次の手順を実行します。

- シナリオを有効にするには，各シナリオの左側にあるチェック・ボックスを選択します（複数選択も可）。
- シナリオを無効にするには，各シナリオの左側にあるチェック・ボックスをクリアします。

回復メカニズムを呼び出すか条件を定義するには，次の手順を実行します。

- ［**回復シナリオのアクティブ化**］ボックスにある次のオプションのいずれかを選択します。
  - **各ステップごと**：ステップを実行するたびに回復メカニズムが呼び出されます。
  - **エラー発生時**：エラーの戻り値を返すステップの後のみ，回復メカニズムが呼び出されます。

    エラーを返すステップは，多くの場合，例外イベントの発生原因となるステップとは同じではありません。

    たとえば，チェック・ボックスを選択するステップによって，ポップアップ・ダイアログ・ボックスが開いたとしましょう。ポップアップ・ダイアログ・ボックスがトリガ・イベントとして定義されてはいるものの，チェック・ボックスを選択するステップの実行は成功しているため，QuickTest の処理は次のステップに進みます。続くいくつかのステップでは，アプリケーションに対する操作の実行を必要としないチェックポイント，関数，その他

の条件ステートメントまたはループ・ステートメントが実行される可能性があります。そして，ポップアップ・ダイアログ・ボックスによって実行が妨げられる操作をアプリケーションに対して実行するように QuickTest に指示するステップが登場するには，10 個のステートメントを経てようやく，ということが考えられます。この場合，エラーを返し，回復メカニズムをトリガしてダイアログ・ボックスを閉じるのは，この 10 番目のステップです。回復操作が完了した後は，現在のステップはこの 10 番目のステップであり，トリガ・イベントの原因となったステップではありません。

- **なし**：回復メカニズムが無効になります。

注：［**各ステップごと**］を選択すると，テストの実行中にパフォーマンスが低下する場合があります。

ヒント：実行セッション中，テストに関連付けられている特定のシナリオ，またはすべてのシナリオをプログラムの中から有効 / 無効にすることもできます。詳細については，82 ページ「プログラムによる回復メカニズムの制御」を参照してください。

すべての新しいテストの標準回復シナリオの設定

［テストの設定］ダイアログ・ボックスの［回復］タブで［**標準値に設定**］ボタンをクリックすると，回復シナリオの現在のリストを，すべての新しいテスト用の標準シナリオに設定できます。設定以降，現在の回復シナリオのリストに加える変更は現在のテストにのみ影響し，定義した標準設定のリストは変更されません。

プログラムによる回復メカニズムの制御

Recovery オブジェクトを使用すれば，実行セッション中にプログラムの中で回復メカニズムを制御できます。たとえば，回復メカニズム全体を有効または無効にしたり，実行セッションの一部分で特定の回復シナリオを有効または無効にしたりできます。また，特定の回復シナリオに関するステータス情報を取得

したり，実行セッションの特定の時点で回復メカニズムを明示的に呼び出したりできます。

標準設定では，実行セッション中にエラーが返された場合，QuickTest によって回復トリガが調べられます。Recovery オブジェクトの **Activate** メソッドを使用すると，QuickTest に，実行セッションの特定のステップの後でトリガを調べさせることができます。たとえば，オブジェクト・プロパティ・チェックポイントの実行時に，あるプロセスが開いていると，そのチェックポイントが失敗すると分かっているとします。アプリケーションにおける別の問題である可能性があるため，こうした開いているプロセスがチェックポイントの成功または失敗に影響を及ぼさないようにする必要があります。

しかし，チェックポイントの失敗は，実行エラーにはなりません。そのため，標準設定では，回復メカニズムがオブジェクトの状態によって呼び出されることはありません。オブジェクトのプロパティが特定の状態にあるときに指定の開いているプロセスを探して閉じる回復シナリオを定義できます。オブジェクトのプロパティの状態は，問題のあるプロセスが開いている場合の値を示します。QuickTest に対して，チェックポイントが失敗したときに回復メカニズムを呼び出させ，問題のプロセスが開いていないか調べさせて，あれば閉じさせ，失敗したチェックポイントを実行しなおすように指示できます。これにより，チェックポイントが 2 回目に実行されるときには，チェックポイントは開かれているプロセスによる影響を受けなくなります。

Recovery オブジェクトとそのメソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

第 4 章

オブジェクトの認識の設定

オブジェクトに対する操作を記録するとき，またはオブジェクト・リポジトリにオブジェクトを追加するとき，QuickTest は当該オブジェクトのプロパティと値のセットを学習します。このプロパティと値のセットは，当該オブジェクトをオブジェクト階層の中で一意に識別する記述です。多くの場合，この記述で，QuickTest が実行セッション中にオブジェクトを十分に識別できます。

特定のオブジェクト・クラスを表す記述が，アプリケーションのオブジェクトを最も論理的に記述するものでないことが判明した場合や，オブジェクト記述の中のプロパティの値が頻繁に変わることが予想される場合には，QuickTest によるオブジェクトの学習方法，識別方法を設定できます。また，ユーザ定義オブジェクトを標準のテスト・オブジェクト・クラスに割り当て，QuickTest によるユーザ定義オブジェクト・クラスのオブジェクトを学習する方法も設定できます。

本章では，次の内容について説明します。

- オブジェクトの認識の設定について
- ［オブジェクトの認識］ダイアログ・ボックス
- スマート認識の設定
- ユーザ定義のテスト・オブジェクト・クラスの割り当て

オブジェクトの認識の設定について

QuickTest には，各テスト・オブジェクトに対して学習するプロパティのセットがあらかじめ用意されています。記録または追加対象オブジェクトを一意に識別するのにこれらの必須プロパティ値では十分でない場合，QuickTest は何らかの補足プロパティまたは序数識別子を追加して，一意の記述を作成します。

「必須プロパティ」とは，QuickTest が特定のテスト・オブジェクト・クラスについて必ず学習するプロパティです。

「補足プロパティ」とは，QuickTest がアプリケーションの特定のオブジェクトについて学習する必須プロパティでは一意の記述を作成するのに不十分である場合にだけ QuickTest が学習するプロパティです。1 つのオブジェクト・クラスに複数の補足プロパティが定義されている場合，QuickTest は補足プロパティを 1 つずつ順番に学習し，オブジェクトの一意の記述ができたところで学習を止めます。QuickTest が学習した補足プロパティはテスト・オブジェクトの記述に追加されます。

注：すべての定義済みの必須および補足プロパティを組み合わせても一意のテスト・オブジェクト記述の作成に十分でない場合，QuickTest は選択された序数識別子の値も学習します。詳細については，93 ページ「序数識別子の選択」を参照してください。

テストの実行の際，QuickTest は，学習した（序数識別子のない）記述に一致するオブジェクトを検索します。記述に一致するオブジェクトが見つからない場合や，記述に一致するオブジェクトが複数ある場合，QuickTest はスマート認識メカニズムを使って（ただし，有効になっている場合），オブジェクトを識別します。多くの場合，スマート認識定義が存在すれば，学習した記述が 1 つ以上のプロパティ値が変更されたことで識別に失敗する場合に，QuickTest によるオブジェクトの識別に役立ちます。テスト・オブジェクト記述は，スマート認識メカニズムでもオブジェクトの候補を 1 つに絞り込めない場合に限り，序数識別子と組み合わせて使われます。

［オブジェクトの認識］ダイアログ・ボックス（［**ツール**］＞［**オブジェクトの認識**］）では，QuickTest がアプリケーションの中のオブジェクトの記述を学習するのに使う，必須プロパティ，補足プロパティ，および序数識別子プロパティを設定できるほか，スマート認識の有効化と設定が可能です。

また，新しいユーザ定義クラスを設定し，それらを既存のテスト・オブジェクト・クラスにマップして，テスト実行時にユーザ定義クラスからオブジェクトを QuickTest が認識するようにもできます。

[オブジェクトの認識] ダイアログ・ボックス

[オブジェクトの認識] ダイアログ・ボックスのメイン画面を使って，必須および補足プロパティの設定，序数識別子の選択，および各テスト・オブジェクトに対するスマート認識メカニズムを有効にするかどうかを指定します。

[オブジェクトの識別] ダイアログ・ボックスから，ユーザ定義オブジェクト・クラスの定義とそれらの標準ウィンドウ・オブジェクト・クラスへの割り当ても行えます。また，[**テスト オブジェクト クラス**] リストに表示される任意のオブジェクトにスマート識別メカニズムを設定することもできます。

注：

[オブジェクトの認識] ダイアログ・ボックスで行った変更は，すでにオブジェクト・リポジトリに追加されたオブジェクトには影響しません。

WinMenu，VbLabel，VbObject，VbToolbar オブジェクトなど，特定のテスト・オブジェクトの学習したスマート認識プロパティは，設定できません。したがって，これらのオブジェクトには選択した環境の [**テスト オブジェクト クラス**] リストが含まれません。

詳細については，次を参照してください。

- 88 ページ「必須および補足記録プロパティの設定」
- 93 ページ「序数識別子の選択」
- 98 ページ「スマート認識の有効化と無効化」
- 99 ページ「テスト・オブジェクトの標準設定のオブジェクト認識設定の復元」
- 99 ページ「オブジェクト認識設定用の自動スクリプトの生成」
- 100 ページ「スマート認識の設定」
- 109 ページ「ユーザ定義のテスト・オブジェクト・クラスの割り当て」

必須および補足記録プロパティの設定

QuickTest が特定のオブジェクト・クラスのために使う記述が，アプリケーションのオブジェクトを記述するために十分には論理的でないことが判明した場合や，オブジェクト記述の中で現在使われているプロパティの値が変わることが予想される場合には，そのクラスのオブジェクトを学習するときに QuickTest が学習する必須および補足プロパティを変更できます。

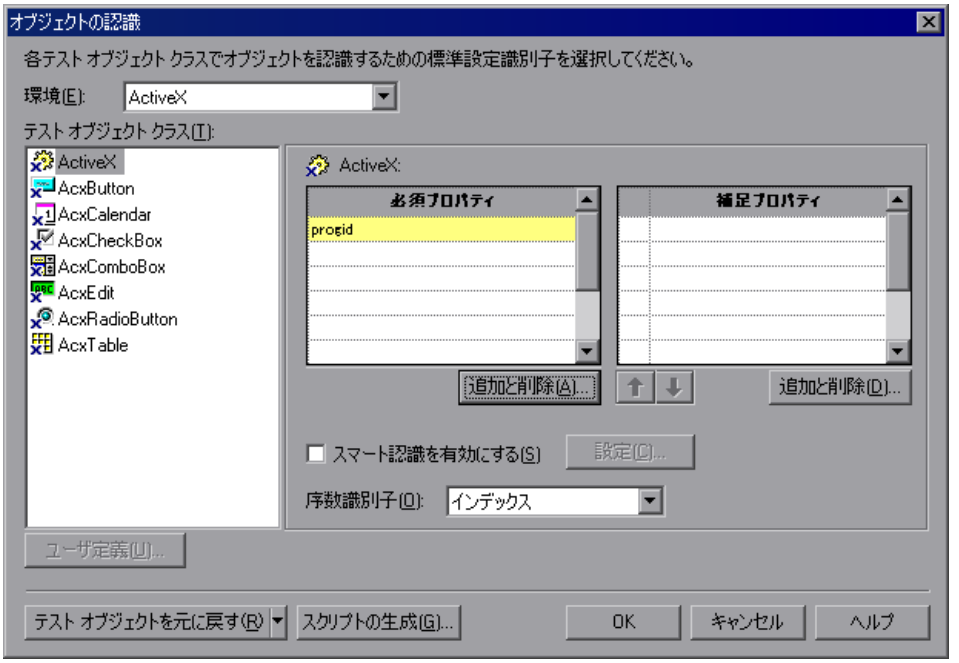
実行セッション中，QuickTest はテスト・オブジェクト記述中のすべてのプロパティに一致するオブジェクトを探します。このとき，必須プロパティとして学習したものと補足プロパティとして学習したものを区別しません。

たとえば，Web Image オブジェクトの標準の必須プロパティは，**alt**，**html tag**，および **image type** プロパティです。標準の補足プロパティは定義されていません。Web サイトに複数の広告を循環して表示するいくつかの広告枠があるものとします。これらの広告枠のそれぞれのイメージをクリックするテストを記録したいものとします。しかし，それぞれの広告イメージの **alt** 値は異なるため，テストを作成すると 1 つの alt 値が記録され，そしてテストを実行するとほとんどの場合，別の **alt** 値がキャプチャされるため，テスト実行が失敗することになります。この場合，Web Image 必須プロパティ・リストから **alt** プロパティを削除できます。その代わり，サイトの特定の広告枠に表示される各広告イメージは，イメージの **name** プロパティの値が同じなので，必須プロパティにその **name** プロパティを追加して，QuickTest が一意にオブジェクトを識別できるようにします。

また，ページの複数の場所に表示される Web イメージに（たとえば，ロゴがページの一番上と下に表示されるなど），Web デザイナによって Image タグに特別な **ID** プロパティが追加されたとします。ページに一度だけ表示されるイメージであれば，一意の記述を作成するには必須プロパティで十分ですが，同じページにイメージが複数回表示される場合には，QuickTest に **ID** プロパティも学習させたいところです。このためには，**ID** プロパティを補足プロパティとして追加して，一意のテスト・オブジェクト記述の作成に必要な場合にだけ QuickTest に **ID** プロパティを学習させるようにします。

テスト・オブジェクト・クラスのための必須および補足プロパティを作成するには，次の手順を実行します。

1 ［**ツール**］＞［**オブジェクトの認識**］を選択します。［オブジェクトの認識］ダイアログ・ボックスが表示されます。



2 ［**環境**］リストで適切な環境を選択します。選択した環境に関連付けられているテスト・オブジェクト・クラスが［**テスト オブジェクト クラス**］リストにアルファベット順に表示されます（標準の Windows では，ユーザ定義オブジェクトがリストの一番下に表示されます）。

注：［環境］リストに含まれている環境は，読み込まれたアドイン環境に対応するものです。アドインの読み込みの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 27 章「QuickTest アドインの使用法」を参照してください。

---

3 ［**テスト オブジェクト クラス**］リストで，設定するテスト・オブジェクト・クラスを選択します。

4 ［**必須 プロパティ**］リストで，［**追加と削除**］をクリックします。必須プロパティのための［プロパティの追加と削除］ダイアログ・ボックスが表示されます。

5 必須プロパティ・リストに含めるプロパティを選択します。また，リストから削除するプロパティを消去します。

注：同じプロパティを必須と補足の両方のプロパティ・リストに含めることはできません。

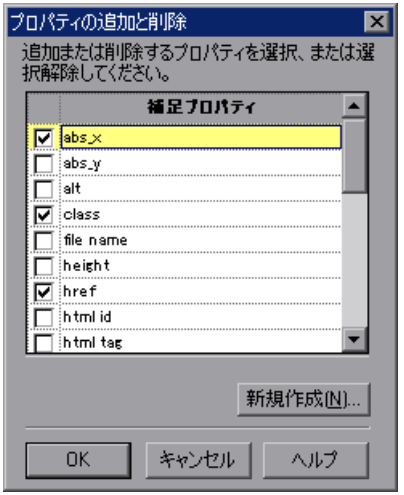
［**新規作成**］をクリックして，表示されるダイアログ・ボックスに有効なプロパティ名を指定することにより，新規プロパティを指定できます。

ヒント：attribute/ ＜プロパティ名＞の形式を使用して，Web オブジェクトに使用可能なプロパティの集合にプロパティ名を追加することもできます。これを行うためには，［**新規作成**］をクリックします。［新規プロパティ］ダイアログ・ボックスが開きます。有効なプロパティを，attribute/ ＜プロパティ名＞の形式で入力して，［**OK**］をクリックします。新プロパティが［**プロパティの追加と削除**］リストに追加されます。たとえば，MyColor というプロパティを追加するには，attribute/MyColor と入力します。

---

6 ［**OK**］をクリックし，［プロパティの追加と削除］ダイアログ・ボックスを閉じます。更新された必須プロパティの集合が［**必須プロパティ**］リストに表示されます。

7 ［**補足 プロパティ**］リストで，［**追加と削除**］をクリックします。必須プロパティのための［プロパティの追加と削除］ダイアログ・ボックスが表示されます。



8 補足プロパティ・リストに含めるプロパティを選択します。また，リストから削除するプロパティを消去します。

注：同じプロパティを必須と補足の両方のプロパティ・リストに含めることはできません。

［**新規作成**］をクリックして，表示されるダイアログ・ボックスに有効なプロパティ名を指定することにより，新規プロパティを指定できます。

ヒント：attribute/ **＜プロパティ名＞**の形式を使用して，Web オブジェクトに使用可能なプロパティの集合にプロパティ名を追加することもできます。これを行うためには，［**新規作成**］をクリックします。［新規プロパティ］ダイアログ・ボックスが開きます。有効なプロパティを，attribute/ **＜プロパティ名＞**の形式で入力して，［**OK**］をクリックします。新規プロパティが［**補足プロパティ**］リストに追加されます。たとえば，MyColor というプロパティを追加するには，attribute/MyColor と入力します。

---

9 ［**OK**］をクリックし，［プロパティの追加と削除］ダイアログ・ボックスを閉じます。プロパティは［補足プロパティ］リストに表示されます。

10 上向き矢印と下向き矢印を使って，補足プロパティの順序を指定します。オブジェクトを学習するときに一意のオブジェクト記述を作成するために補足プロパティが必要な場合，QuickTest は，一意の記述を作成するのに十分な情報が得られるまで，［補足プロパティ］リストでの順序に従って，記述に補足プロパティを 1 つずつ追加していきます。

序数識別子の選択

［オブジェクトの認識］ダイアログ・ボックスで指定した必須プロパティと補足プロパティを学習するのに加え，QuickTest は予備的に各テスト・オブジェクトの序数識別子も学習できます。「序数識別子」は，同じ記述を持つオブジェクト（必須および補足プロパティ・リストで指定されているすべてのプロパティの値が同じオブジェクト）を区別するために，他のオブジェクトとの相対的な順番を表す数値を割り当てます。この順位の値によって QuickTest は，必須および補足プロパティでは一意の記述を作成するのに不十分な場合でも，一意の記述を作成できます。

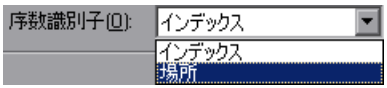
割り当てられた序数プロパティの値は，相対的な値であり，QuickTest がオブジェクトを学習するときに表示されていた他のオブジェクトとの相対関係に基づいています。アプリケーションのページやスクリーンのレイアウトや構成が変われば，オブジェクト自体には一切変化がなくても，この値が変わることがあります。そのため，QuickTest はすべての利用可能な必須および補足プロパティを使っても一意の記述を作成できない場合にだけ，この予備的な序数識別子の値を学習します。

さらに，QuickTest は序数識別子を学習しても，実行セッション中に序数識別子を使うのは，学習した記述およびスマート認識メカニズムを使ってアプリケーションのオブジェクトを十分に識別できない場合だけです。QuickTest が他のテスト・オブジェクト・プロパティを使用して実行セッション中にオブジェクトを識別できれば，序数識別子は無視されます。

QuickTest がオブジェクトの識別に使用できる序数識別子のタイプは次のとおりです。

- ［**インデックス**］：オブジェクトがアプリケーション・コードの中に出現する順序を，それ以外は同じ記述を持つ他のオブジェクトとの相対関係で表します。詳細については，94 ページ「インデックス・プロパティを使用したオブジェクトの識別」を参照してください。
- ［**場所**］：親ウィンドウ，フレーム，またはダイアログ・ボックス内においてオブジェクトが出現する順序を，その他の記述が同じであるほかのオブジェクトとの相対位置で表します。詳細については，95 ページ「場所プロパティを使用したオブジェクトの識別」を参照してください。
- ［**CreationTime**］：（Browser オブジェクトのみ）同じ記述を持つブラウザが開いた相対的な順番を表します。詳細については，96 ページ「CreationTime プロパティを使用したオブジェクトの識別」を参照してください。

標準設定では，テスト・オブジェクト・クラスごとに序数識別子があります。標準設定の序数識別子を変更するには，［**序数識別子**］ボックスから，目的のタイプを選択します。



ヒント：QuickTest は，記録に必須および補足プロパティを使って一意のテスト・オブジェクト記述を作成できた場合には，序数識別子は学習しません。［オブジェクトのプロパティ］または［オブジェクト リポジトリ］ダイアログ・ボックスの序数識別子ダイアログを使用して，後からオブジェクトのテスト・オブジェクト・プロパティに序数識別子を追加できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 6 章「テスト・オブジェクトを使用した作業」を参照してください。

インデックス・プロパティを使用したオブジェクトの識別

オブジェクトの学習中，QuickTest はオブジェクトを一意に識別できるように，オブジェクトの **インデックス**・プロパティに値を割り当てることができます。この値は，ソース・コード内のオブジェクトの順番に基づいています。最初の番号は 0 です。

インデックス・プロパティ値は，各オブジェクトに固有の値です。したがって，ある WebEdit テスト・オブジェクトを記述するのに Index:=3 を使用すると，QuickTest はページ内の 4 番目の WebEdit を探します。一方，WebElement オブジェクトを記述するのに Index:=3 を使用すると，WebElement オブジェクトはすべての Web オブジェクトに適用されるため，QuickTest はタイプに関係なくページ内の 4 番目の Web オブジェクトを探します。

たとえば，次のオブジェクトを含んだページがあるとします。

- Apple という名前の画像
- UserName という名前の画像
- UserName という名前の WebEdit オブジェクト
- Password という名前の画像
- Password という名前の WebEdit オブジェクト

次のステートメントは，リストの 3 番目の項目を表します。ページ内で UserName という名前を持つ最初の WebEdit オブジェクトだからです。

```
WebEdit("Name:=UserName", "Index:=0")
```

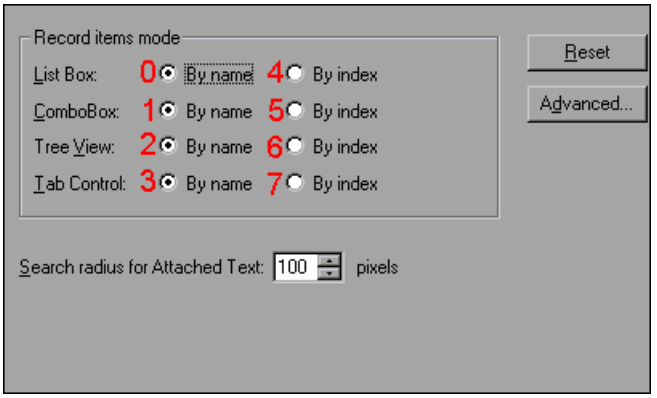
一方，次のステートメントは，リストの 2 番目の項目を表します。ページ内で UserName という名前を持つ最初の任意のタイプ（WebElement）のオブジェクトだからです。

```
WebElement("Name:=UserName", "Index:=0")
```

場所プロパティを使用したオブジェクトの識別

オブジェクトの学習中，QuickTest はオブジェクトを一意に識別できるように，オブジェクトの **場所** プロパティに値を割り当てることができます。この値は，ウィンドウ，フレーム，ダイアログ・ボックス内に現れる同一のプロパティを持つ他のオブジェクトとの相対的な順番に基づいて決まります。最初のオブジェクトの場合，値は 0 です。値はカラム内で上から下に，そして左から右への順序で割り当てられます。

次の例では，ダイアログ・ボックス内のラジオ・ボタンは，位置のプロパティに従って番号が付けられています。



場所プロパティ値はオブジェクト固有の値です。したがって，ある WinButton テスト・オブジェクトを記述するのに Location:=3 を使用すると，QuickTest は 4 番目の WinButton をページ内の上から下，左から右に探します。一方，WinObject オブジェクトを記述するのに Location:=3 を使用すると，WinObject オブジェクトはすべての標準オブジェクトに適用されるため，QuickTest はタイ

プに関係なくページ内の 4 番目の標準オブジェクトをページ内の上から下，左から右に探します。

たとえば，次のオブジェクトを含んだダイアログ・ボックスがあるとします。

- OK という名前のボタン・オブジェクト
- Add/Remove という名前のボタン・オブジェクト
- Add/Remove という名前のチェック・ボックス・オブジェクト
- Help という名前のボタン・オブジェクト
- Check spelling という名前のチェック・ボックス・オブジェクト

次のステートメントは，リストの 3 番目の項目を表します。ページ内で Add/Remove という名前を持つ最初のチェック・ボックス・オブジェクトだからです。

```
WinCheckBox("Name:=Add/Remove", "Location:=0")
```

一方，次のステートメントは，リストの 2 番目の項目を表します。ページ内で Add/Remove という名前を持つ最初の任意のタイプ（WinObject）のオブジェクトだからです。

```
WinObject("Name:=Add/Remove", "Location:=0")
```

CreationTime プロパティを使用したオブジェクトの識別

Browser オブジェクトの学習中，テスト・オブジェクトの記述によってオブジェクトを一意に特定できない場合，QuickTest は **CreationTime** テスト・オブジェクト・プロパティに値を割り当てます。この値は，同じ記述を持つブラウザが開いた相対的な順番を示します。最初に開いたブラウザは，CreationTime = 0 の割り当てとなります。

実行セッション中，テスト・オブジェクトの記述だけに基づいて Browser オブジェクトを一意に識別できないとき，QuickTest はブラウザが開いた順番を確認し，**CreationTime** プロパティを使用して正しい Browser オブジェクトを識別します。

たとえば，それぞれ 9:01 pm，9:03 pm，9:05 pm と，開いた時間だけが違う同一の 3 つのブラウザを対象にテストを記録する場合，QuickTest は 9:01 pm のブラウザに CreationTime = 0 を，9:03 pm のブラウザに CreationTime = 1 を，9:05 pm のブラウザに CreationTime = 2 を割り当てます。

10:30 pm にテストを実行したときに，ブラウザが 10:31 pm，10:33 pm，10:34 pm に開いたとします。QuickTest は 10:31 pm のブラウザが CreationTime = 0 のブラウザ・テスト・オブジェクト，10:33 pm のブラウザが CreationTime = 1 のブラウザ・テスト・オブジェクト，10:34 pm のブラウザが CreationTime = 2 のブラウザ・テスト・オブジェクトであると識別します。

開いているブラウザがいくつかある場合，CreationTime の最も低いものが最初に開いたものであり，最も高いものが最後に開いたものになります。たとえば，3 つ以上のブラウザが開いている場合，CreationTime = 2 のブラウザは 3 番目に開いたブラウザです。セッションの記録中に 7 つのブラウザが開いている場合，CreationTime = 6 のブラウザが最後に開いたブラウザです。

特定の CreationTime 値のブラウザを対象にステップが記録され，実行セッション中にこの CreationTime 値を持つブラウザが開いていない場合，ステップは CreationTime 値の最も高いブラウザで実行されます。たとえば，ステップが CreationTime = 6 のブラウザを対象に記録され，CreationTime = 0 と CreationTime = 1 という 2 つのブラウザだけが実行セッション中に開いているとすると，ステップは最後に開いたブラウザ（この例では CreationTime = 1）で実行されます。

注：セッション中の特定の時間に使用できる CreationTime 値は連番になっていない可能性があります。たとえば，記録または実行セッション中に 6 つのブラウザを開いたとして，セッション中にそのうち 2 番目と 4 番目に開いたブラウザ（CreationTime 値 1 と 3）を閉じたとすると，セッションの最後で開いているブラウザは CreationTime 値が 0，2，4，5 のブラウザになります。

スマート認識の有効化と無効化

特定のテスト・オブジェクト・クラスの［**スマート認識を有効にする**］チェック・ボックスを選択すると，QuickTest によってオブジェクトの基本フィルタ・プロパティとオプション・フィルタ・プロパティの一方または両方で指定されているすべてのプロパティ値が学習されます。

標準では，一部のテスト・オブジェクトはすでにスマート認識が設定されており，他は設定されていません。標準で設定されているものは，［**スマート認識を有効にする**］チェック・ボックスも標準で選択されています。

スマート認識設定が定義されているテスト・オブジェクト・クラスのみスマート認識メカニズムを有効にします。しかし，特定のテスト・オブジェクト・クラスを対象としたスマート認識設定を定義した場合でも，スマート認識プロパティ値を学習したくないこともあります。スマート認識プロパティを学習しない場合は，［**スマート認識を有効にする**］チェック・ボックスをクリアします。

注：特定のオブジェクトのスマート認識プロパティを学習するように設定してある場合でも，［オブジェクトのプロパティ］または［オブジェクト リポジトリ］ダイアログ・ボックスで，特定のオブジェクトに対するスマート認識機能の使用を無効にできます。また，［テストの設定］ダイアログ・ボックスの［実行］タブでテスト全体に対するスマート認識メカニズムの使用を無効にできます。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の第 6 章「テスト・オブジェクトを使用した作業」および 751 ページ「テストのための実行設定の定義」を参照してください。

ただし，スマート認識プロパティを学習しなければ，オブジェクトに対して後でスマート認識メカニズムを有効にすることはできません。

スマート認識メカニズムの詳細については，100 ページ「スマート認識の設定」を参照してください。

テスト・オブジェクトの標準設定のオブジェクト認識設定の復元

ロードされているすべての環境，現在の環境のみ，または選択したテスト・オブジェクトの，オブジェクトの認識設定およびスマート認識プロパティ設定を，標準の設定に戻すことができます。

リセットできるのは組み込みオブジェクトのプロパティのみです。標準の Windows 環境をリセットすると，ユーザ定義のオブジェクトは削除されます。

注 : [オブジェクトの認識] ダイアログ・ボックスの [環境] ボックスには，現在ロードされている環境のみが一覧表示されます。

標準では [**テスト オブジェクトを元に戻す**] ボタンが表示されますが，下矢印をクリックして，次のいずれかのオプションを選択することができます。

- [**テスト オブジェクトを元に戻す**]：選択したテスト・オブジェクトの設定をリセットして，システム標準の値に戻します。
- [**環境を元に戻す**]：現在の環境のすべてのテスト・オブジェクトの設定をリセットして，システム標準の値に戻します。
- [**すべて元に戻す**]：現在ロードされている環境のすべての設定をリセットして，システム標準の値に戻します。

オブジェクト認識設定用の自動スクリプトの生成

[**スクリプトの生成**] ボタンをクリックして，現在のオブジェクトの認識設定を含む自動スクリプトを生成できます。詳細については 215 ページ「QuickTest 操作のオートメーション」を参照するか，『**QuickTest オートメーション・オブジェクト・モデル・リファレンス**』（[**ヘルプ**] > [**QuickTest オートメーション オブジェクト モデル リファレンス**]）を参照してください。

スマート認識の設定

スマート認識プロパティを設定すれば，学習したオブジェクト記述に含まれるプロパティの一部が変更されている場合に，QuickTest がアプリケーションのオブジェクトを識別するのに役立ちます。

QuickTest は，学習した記述を使ってオブジェクトを識別するとき，記述中のすべてのプロパティ値と一致するオブジェクトを検索します。ほとんどの場合，この記述はオブジェクトを識別する最も簡単な方法です。そしてオブジェクトの主要なプロパティが変更されない限り，この方法は有効です。

QuickTest が学習したオブジェクト記述に一致するオブジェクトを見つけられない場合や，複数のオブジェクトが記述に適合する場合，QuickTest は学習した記述を無視し，スマート認識メカニズムを使ってオブジェクトの識別を試みます。

スマート認識メカニズムはもう少し複雑ですが，柔軟性が高いので，スマート認識定義を適格に設定すれば，学習した記述では識別できないときに，QuickTest がオブジェクト（存在していれば）を識別するのに役立ちます。

スマート認識メカニズムは次の 2 タイプのプロパティを使います。

- ［**基本フィルタのプロパティ**］は，特定のテスト・オブジェクト・クラスの最も基本的なプロパティです。その値は，元のオブジェクトの根本的な部分を変えなければ変わりません。たとえば，Web のリンクのタグが <A> から何か別の値に変わった場合には，もはやそれを同じオブジェクトとは呼べません。
- ［**オプション フィルタのプロパティ**］: 特定のクラスのオブジェクトを識別するのに役立つ別のプロパティです。これらのプロパティは，通常は変わらないとみなされますが，適用できなくなった場合には無視できます。

スマート認識の処理過程について

QuickTest が実行セッション中にスマート認識メカニズムに切り替わると（学習した記述ではオブジェクトを識別できなかったため），スマート認識は次のプロセスでオブジェクトを識別します。

1 QuickTest は学習したテスト・オブジェクト記述を「忘れ」，［基本フィルタのプロパティ］リストに含まれているすべてのプロパティに適合するオブジェクト（親オブジェクト内のオブジェクト）を含んだ新しいオブジェクト候補リストを作成します。

2 QuickTest は，［オプション フィルタのプロパティ］リストの最初のプロパティに適合しないオブジェクトをオブジェクト候補リストからすべて除外します。残りのオブジェクトが新しいオブジェクト候補リストになります。

3 QuickTest が新しいオブジェクト候補リストを評価します。

- 新しいオブジェクト候補リストに，まだ複数のオブジェクトがある場合，QuickTest はこの新しい（より小さい）オブジェクト候補リストを使って，リスト中の次のオプション・フィルタ・プロパティを使って手順 2 を繰り返します。
- 新しくできたオブジェクト候補リストが空の場合，QuickTest はこのオプション・フィルタ・プロパティを無視し，前のオブジェクト候補リストに戻って手順 2 をリストの次のオプション・フィルタ・プロパティを使って繰り返します。
- オブジェクト候補リストにオブジェクトが 1 つだけ含まれている場合，QuickTest はそれが識別されたオブジェクトであると判断し，そのオブジェクトを含んでいるステートメントを実行します。

4 QuickTest は手順 2 と 3 で説明した処理を，1 つのオブジェクトを識別するか，オプション・フィルタ・プロパティを使い果たすまで実行し続けます。

スマート認識の除外処理完了後も，QuickTest がまだオブジェクトを識別できない場合は，QuickTest は学習した記述に加え，序数識別子を使ってオブジェクトを識別します。

学習したスクリプトと序数識別子の組み合わせでもオブジェクトを識別するのに不十分な場合は，QuickTest は実行セッションを中止し，実行エラー・メッセージを表示します。

テスト結果に含まれるスマート認識情報の参照

学習した記述では QuickTest が指定されたオブジェクトを一度で識別できず，しかもスマート認識定義が定義されている（そして有効である）場合には，QuickTest はスマート認識メカニズムを使ってオブジェクトの識別を試みます。

QuickTest が，学習した記述では一致するオブジェクトを見つけられず，スマート認識でオブジェクトを見つけるのに成功した場合，テスト結果は警告ステータスを受け取り，スマート認識メカニズムが使用されたことを示します。

スマート認識メカニズムでオブジェクトを識別できない場合，QuickTest は学習した記述に加え，序数識別子を使ってオブジェクトを識別します。それでもオブジェクトを識別できない場合は，テストは失敗し，結果に通常の失敗ステップが表示されます。

詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 673 ページ「テスト結果に含まれるスマート認識情報の分析」を参照してください。

スマート認識の処理過程の例

以下では，あるオブジェクトの認識の課程を見ていきます。

テスト中に次のステートメントがあるものとします。

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Click 22.17
```

テストを作成したとき，QuickTest は Login 画像について次のオブジェクト記述を学習しました。

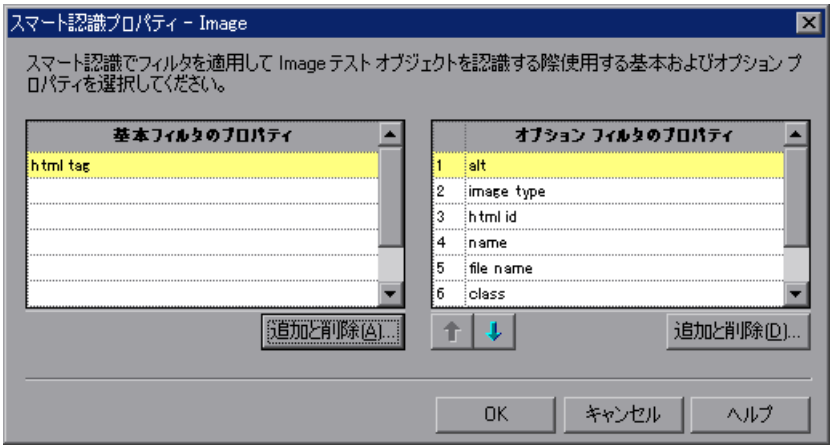
名前	値
記述プロパティ	
image type	Image Button
html tag	INPUT
alt	Sign-In

しかし，テストを作成した後で，ページに 2 つ目のログイン・ボタン（Web サイトの VIP セクションにログインするためのもの）が追加されたため，Web デザイナは元のログイン・ボタンの **alt** タグを basic login に変えました。

Web Image オブジェクトの標準の記述（**alt**，**html tag**，**image type**）はサイト内のほとんどの画像に使えますが，もはやログインの画像には使えません。その画像の **alt** プロパティが学習した記述とは一致しないからです。したがって，テストを実行すると，QuickTest は学習した記述に基づいてログイン・ボタンを識別することができません。しかし，QuickTest はスマート認識定義を使って，ログイン・ボタンをうまく識別できました。

下の例では，QuickTest がスマート認識を使って Login オブジェクトを見つける課程を示します。

1 Web 画像オブジェクトに対するスマート認識定義によれば，QuickTest は Login 画像に対するクリックを記録したとき，次のプロパティの値を学習しています。



学習した値は次のとおりです。

基本フィルタのプロパティ

プロパティ	値
html tag	INPUT

オプション・フィルタのプロパティ

プロパティ	値
alt	Login
image type	Image Button
name	login
file name	login.gif
class	<null>
visible	1

2 QuickTest は Mercury Tours ページの基本フィルタ・プロパティ定義（**html tag** = INPUT および **image type** = Image Button）に適合する 5 つのオブジェクトを識別することによってスマート認識処理を開始します。QuickTest はこれらをオブジェクト候補と考え，［**オプション フィルタのプロパティ**］リストを使った確認を開始します。

3 QuickTest は各オブジェクト候補の **alt** プロパティを確認しますが，どれにも **alt** の値が Login ではありません。したがって，QuickTest はこのプロパティを無視し，次に移ります。

4 QuickTest は各オブジェクト候補の **name** プロパティを確認し，2 つのオブジェクト（基本および VIP ログイン・ボタン）に name:login があることを見つけます。QuickTest は他の 3 つのオブジェクトをリストから除外し，これら 2 つのログイン・ボタンが新しいオブジェクト候補になります。

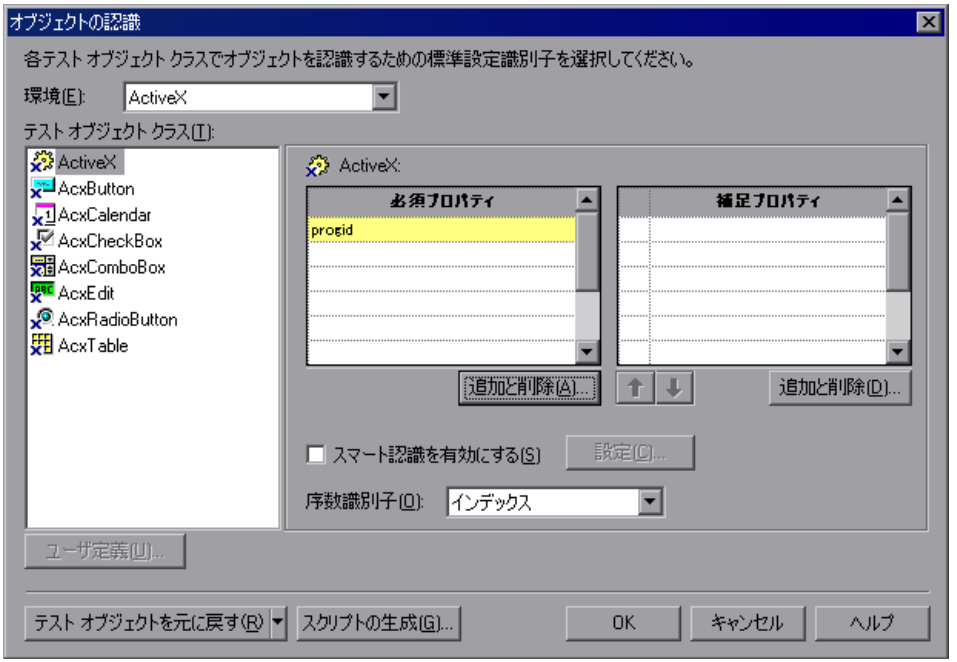
5 QuickTest は残った 2 つのオブジェクト候補の **file name** プロパティを確認します。そのうちの 1 つだけにファイル名 login.gif があるので，QuickTest はログイン・ボタンを見つけたと正しく結論を出し，それをクリックします。

スマート認識定義のステップごとの設定

［オブジェクトの認識］ダイアログ・ボックスからアクセスできる［スマート認識プロパティ］ダイアログ・ボックスを使って，テスト・オブジェクト・クラスのスマート認識定義を設定できます。

スマート認識のプロパティを設定するには，次の手順を実行します。

1 ［**ツール**］＞［**オブジェクトの認識**］を選択します。［オブジェクトの認識］ダイアログ・ボックスが表示されます。



2 ［**環境**］リストで適切な環境を選択します。選択した環境に関連付けられているテスト・オブジェクト・クラスが［**テスト オブジェクト クラス**］リストに表示されます。

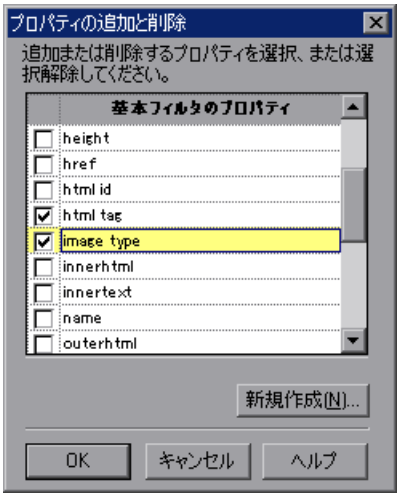
注：［環境］リストに含まれている環境は，読み込まれたアドイン環境に対応するものです。アドインの読み込みの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 27 章「QuickTest アドインの使用法」を参照してください。

---

3 設定するテスト・オブジェクト・クラスを選択します。

4 ［**スマート認識を有効にする**］チェック・ボックスの横にある［**設定**］ボタンをクリックします。［**設定**］ボタンは，［**スマート認識を有効にする**］オプションが選択されている場合のみ有効になります。［スマート認識プロパティ - Image］ダイアログ・ボックスが表示されます。

5 ［**基本フィルタのプロパティ**］リストで，［**追加と削除**］をクリックします。基本フィルタ・プロパティのための［プロパティの追加と削除］ダイアログ・ボックスが表示されます。



6 ［**基本フィルタのプロパティ**］リストに含めるプロパティを選択します。また，リストから削除するプロパティを消去します。

注：同じプロパティを基本とオプションの両方のプロパティ・リストに含めることはできません。

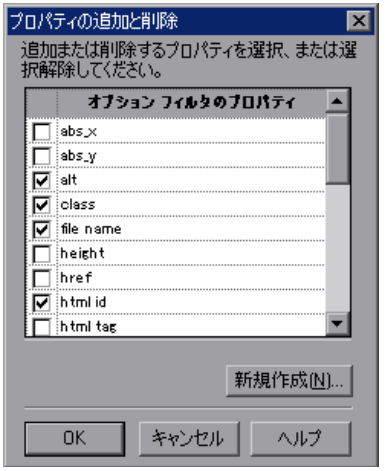
［**新規作成**］をクリックして，表示されるダイアログ・ボックスに有効なプロパティ名を指定することにより，新規プロパティを指定できます。

ヒント：attribute/ **＜プロパティ名＞**の形式を使用して，Web オブジェクトに使用可能なプロパティの集合にプロパティ名を追加することもできます。これを行うためには，［**新規作成**］をクリックします。［新規プロパティ］ダイアログ・ボックスが開きます。有効なプロパティを，attribute/ **＜プロパティ名＞**の形式で入力して，［**OK**］をクリックします。新規プロパティが［**基本フィルタのプロパティ**］リストに追加されます。たとえば，MyColor というプロパティを追加するには，attribute/MyColor と入力します。

---

7 ［**OK**］をクリックし，［プロパティの追加と削除］ダイアログ・ボックスを閉じます。更新された基本フィルタ・プロパティの集合が［**基本フィルタのプロパティ**］リストに表示されます。

8 [**オプション フィルタのプロパティ**] リストで，[**追加と削除**] をクリックします。オプション・フィルタ・プロパティのための [プロパティの追加と削除] ダイアログ・ボックスが表示されます。



9 [**オプション フィルタのプロパティ**] リストに含めるプロパティを選択します。また，リストから削除するプロパティを消去します。

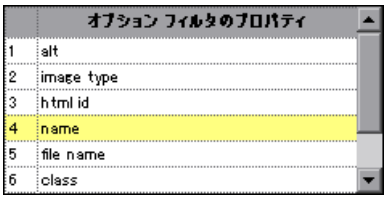
注 : 同じプロパティを基本とオプションの両方のプロパティ・リストに含めることはできません。

[**新規作成**] をクリックして，表示されるダイアログ・ボックスに有効なプロパティ名を指定することにより，新規プロパティを指定できます。

ヒント : attribute/ **＜プロパティ名＞**の形式を使用して，Web オブジェクトに使用可能なプロパティの集合にプロパティ名を追加することもできます。これを行うためには，[**新規作成**] をクリックします。[新規プロパティ] ダイアログ・ボックスが開きます。有効なプロパティを，attribute/ **＜プロパティ名＞**の形式で入力して，[**OK**] をクリックします。新規プロパティが [**オプション フィルタのプロパティ**] リストに追加されます。たとえば，MyColor というプロパティを追加するには，attribute/MyColor と入力します。

---

10 ［**OK**］をクリックし，［プロパティの追加と削除］ダイアログ・ボックスを閉じます。プロパティは［**オプション フィルタのプロパティ**］リストに表示されます。



11 上向き矢印と下向き矢印を使って，オプション・プロパティの順序を指定します。QuickTest はスマート認識メカニズムを使うとき，オプション・プロパティに対する残りのオブジェクト候補を［**オプション フィルタのプロパティ**］で設定した順序に従って，オブジェクト候補が 1 つになるまで 1 つずつチェックします。

ユーザ定義のテスト・オブジェクト・クラスの割り当て

［オブジェクトの割り当て］ダイアログ・ボックスを使って未定義クラスまたはユーザ定義クラスを標準の Windows クラスに割り当てることができます。たとえば，アプリケーションに識別できないボタンがある場合，そのボタンは汎用の WinObject として学習されます。QuickTest に対して，そのオブジェクトが標準の Windows button クラスに属しているものとして識別するように指示できます。そのようにしておくと，記録中にそのボタンをクリックすると，QuickTest はその操作を標準の Windows ボタンをクリックしたのと同じように記録します。未定義オブジェクトまたはユーザ定義オブジェクトを標準オブジェクトに割り当てると，そのオブジェクトは標準の Windows テスト・オブジェクト・クラスのリストに，ユーザ定義オブジェクトとして追加されます。オブジェクトの認識設定を他のあらゆるオブジェクト・クラスと同様に，ユーザ定義オブジェクト・クラスに設定できます。

認識されないオブジェクトの割り当てを行うときは，同等の動作をする標準の Windows クラスにのみ割り当てるようにします。たとえば，ボタンと同等の動作をするオブジェクトを edit クラスに割り当ててはなりません。

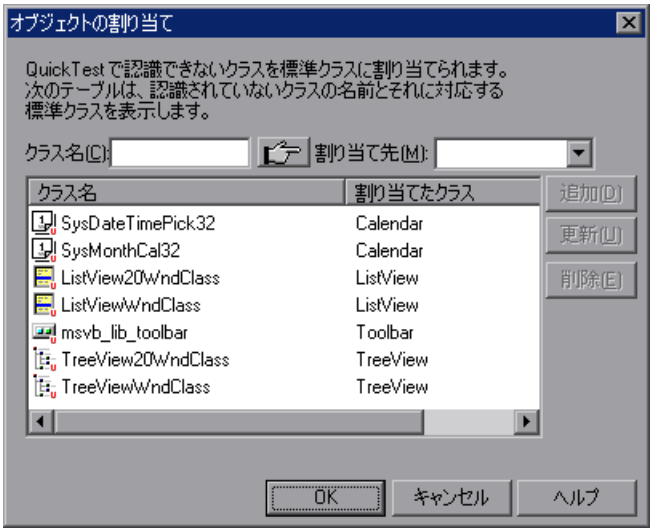
注：ユーザ定義クラスを定義できるのは，［**環境**］ボックスで［**Standard Windows**］が選択されている場合だけです。

未定義クラスまたはユーザ定義クラスを標準の Windows クラスに割り当てるには，次の手順を実行します。

1 ［**ツール**］＞［**オブジェクトの認識**］を選択します。［オブジェクトの認識］ダイアログ・ボックスが表示されます。

2 ［**環境**］ボックスで［**Standard Windows**］を選択します。［**ユーザ定義**］ボタンが有効になります。

3 ［**ユーザ定義**］ボタンをクリックします。［オブジェクトの割り当て］ダイアログ・ボックスが表示されます。



4 指差しボタンをクリックしてから，ユーザ定義クラスに追加するクラスのオブジェクトをクリックします。ユーザ定義オブジェクトの名前が［**クラス名**］ボックスに表示されます。

ヒント：ウィンドウのフォーカスを変更したり，ショートカット・メニューを表示するために右クリックやマウスオーバーなどの操作を実行したりするには，CTRL キーを押しながら操作を行います。選択対象オブジェクトを含んでいるウィンドウが最小化されている場合は，左の CTRL キーを押したまま，Windows タスク・バー内のアプリケーションを右クリックして，ショートカット・メニューから［**元のサイズに戻す**］を選択することで，ウィンドウを表示できます。

---

5 ［**割り当て先**］ボックスで，ユーザ定義オブジェクト・クラスを割り当てる対象となる標準オブジェクト・クラスを選択して［**追加**］をクリックします。クラス名と割り当てがオブジェクト割り当てリストに追加されます。

6 標準クラスにさらにオブジェクトを追加するにはオブジェクトごとに手順 4 ～ 5 を繰り返します。

7 ［**OK**］をクリックします。［オブジェクトの割り当て］ダイアログ・ボックスが閉じ，オブジェクトが標準の Windows テスト・オブジェクト・クラスのリストにユーザ定義テスト・オブジェクトとして追加されます。追加したオブジェクトのアイコンの右下角には，ユーザ定義クラスであることを表す，赤い U の文字が入ります。

8 ユーザ定義オブジェクト・クラスのオブジェクトの認識設定は，他のあらゆるオブジェクト・クラスと同様に設定できます。詳細については，88 ページ「必須および補足記録プロパティの設定」および 100 ページ「スマート認識の設定」を参照してください。

既存の割り当てを変更するには，次の手順を実行します。

1 ［オブジェクトの割り当て］ダイアログ・ボックスのオブジェクト割り当てリストで，変更するクラスを選択します。そのクラス名と現在の割り当てが，［クラス名］および［割り当て先］ボックスに表示されます。

2 選択したユーザ定義オブジェクト・クラスを割り当てる対象となる標準オブジェクト・クラスを選択して［**更新**］をクリックします。オブジェクト割り当てリストのクラス名と割り当てが更新されます。

3 ［**OK**］をクリックし，［オブジェクトの割り当て］ダイアログ・ボックスを閉じます。

既存の割り当てを削除するには，次の手順を実行します。

1 ［オブジェクトの割り当て］ダイアログ・ボックスのオブジェクト割り当てリストで，削除するクラスを選択します。

2 ［**削除**］をクリックします。そのクラス名と割り当てが，［オブジェクトの割り当て］ダイアログ・ボックスのオブジェクト割り当てリストから削除されます。

3 ［**OK**］をクリックします。［オブジェクトの割り当て］ダイアログ・ボックスが閉じ，［オブジェクトの認識］ダイアログ・ボックスの標準の Windows テスト・オブジェクト・クラスからクラス名が削除されます。

第 5 章

エキスパート・ビューおよび関数ライブラリ・ウィンドウを使用した作業

QuickTest のテストは，Microsoft のプログラミング言語である VBScript で記述されたステートメントによって構成されています。エキスパート・ビューは，VBScript に慣れているテスト担当者がキーワード・ビューの代わりに使える機能です。QuickTest では VBScript を使用して関数ライブラリも作成できます。

本章では，エキスパート・ビューでの作業方法を説明し，VBScript について簡単に紹介し，いくつかの簡単なプログラミング・テクニックを使ってテストおよび関数ライブラリを拡張する方法を示します。

本章では，次の内容について説明します。

- エキスパート・ビューおよび［関数ライブラリ］ウィンドウを使った作業について
- エキスパート・ビューの理解と使用
- エキスパート・ビューおよび関数ライブラリ内での操作
- VBScript の基本的な構文の理解
- プログラム的記述の使用
- プログラムによるアプリケーションの実行と終了
- コメント，フロー制御，その他の VBScript ステートメントの使用
- テスト・オブジェクトのプロパティ値の取得と設定
- 実行環境オブジェクトのプロパティおよびメソッドへのアクセス
- DOS コマンドの実行
- Windows API を使用したテストおよび関数ライブラリの拡張
- 実行セッション中に報告するステップの選択

エキスパート・ビューおよび［関数ライブラリ］ウィンドウを使った作業について

エキスパート・ビューでは，アクションを VBScript として表示できます。VBScript に慣れていれば，プログラミングを通じてステートメントの追加と更新を行い，テストおよび関数ライブラリを拡張できます。また，［関数ライブラリ］ウィンドウを使用して，関数ライブラリの作成や関数ライブラリを使った作業ができます。

ステップを記録すると，ステップはエキスパート・ビューに VBScript ステートメントとして表示されます。テストを記録した後に，記録可能な，あるいは記録不可能な VBScript ステートメントを追加することで，テストの機能と柔軟性を高めることができます。

VBScript での作業の詳細については，QuickTest の［ヘルプ］メニューから VBScript に関するマニュアルを参照してください（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］）。

オブジェクトの操作や，アプリケーションからの情報を取得するステートメントを追加できます。たとえば，オブジェクトが存在するかどうかを検査するステップの追加や，メソッドの戻り値の取得ができます。

手作業でまたはステップ・ジェネレータを使用して，テストまたは関数ライブラリにステップを追加することができます。ステップ・ジェネレータの使用法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 531 ページ「ステップ・ジェネレータを使用したステップの挿入」を参照してください。

エキスパート・ビューに表示されたテストまたは関数ライブラリはいつでも印刷できます。印刷出力には追加の情報を含めることもできます。エキスパート・ビューでの印刷の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 109 ページ「テストの印刷」を参照してください。関数ライブラリの印刷の詳細については，185 ページ「関数ライブラリの印刷」を参照してください。

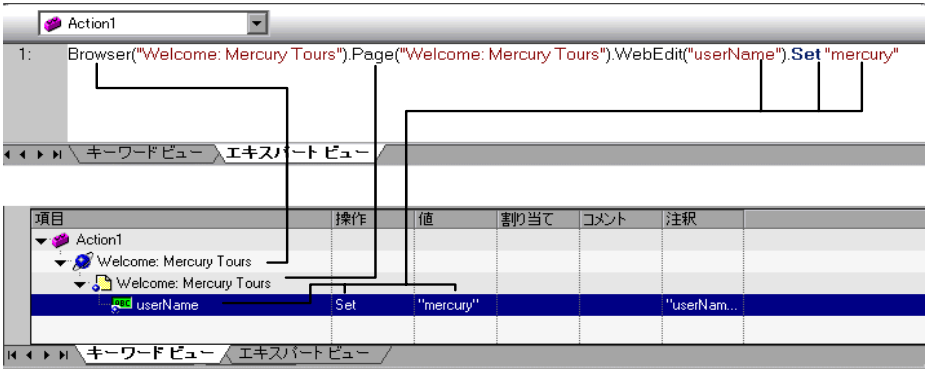
エキスパート・ビューの理解と使用

VBScript ステートメントの作業を行うには，キーワード・ビューの代わりに，エキスパート・ビューでテストの作業を行うことができます。それぞれのビューを切り替えることが可能です。ビューを切り替えるには，QuickTest ウィンドウのテスト表示枠の最下部にある［エキスパート ビュー］タブまたは［キーワード ビュー］タブを選択します。

エキスパート・ビューには，キーワード・ビューと同じステップおよびオブジェクトが表示されますが，表示される形式が異なります。

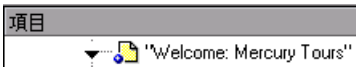
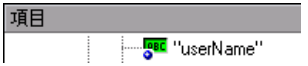
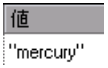
- キーワード・ビューでは，各ステップの情報とともに，オブジェクトの階層がアイコン形式のテーブルとして表示されます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 5 章「キーワード・ビューを使った作業」を参照してください。
- エキスパート・ビューでは，各ステップが VBScrip の行として表示されます。オブジェクト・ベースのステップでは，VBScript のステートメントがオブジェクト階層を定義します。

次の図は，同じオブジェクト階層をエキスパート・ビューで表示した場合とキーワード・ビューで表示した場合を示します。



エキスパート・ビューの VBScript の各行は，テストの各ステップに相当します。上の例は，ユーザがエディット・ボックスに「mercury」という名前を挿入するテストのステップを表しています。ステップの階層では，サイト名，ページ名，ページ内のオブジェクトのタイプと名前，オブジェクトに対して実行されたメソッド名を確認できます。

次の表は，同じステップの各部がキーワード・ビューとエキスパート・ビューのそれぞれでどのように表現されるかを示します。

キーワード・ビュー	エキスパート・ビュー	説明
	Browser("Welcome:Mercury Tours")	ブラウザ・テスト・オブジェクトの名前は Welcome:Mercury Tours。
	Page ("Welcome:Mercury Tours")	現在のページの名前は Welcome:Mercury Tours。
	WebEdit ("userName")	オブジェクトのタイプは WebEdit。操作の対象となるエディット・ボックスの名前は userName。
	Set	対象エディット・ボックスで実行されるメソッドは **Set**。
	"mercury"	［**username**］エディット・ボックスに挿入される値は mercury。

エキスパート・ビューでは，オブジェクトの記述は，オブジェクトのタイプに続いて，括弧内に表示されます。オブジェクトのリポジトリに格納されているすべてのオブジェクトにとって，オブジェクトの記述としては，オブジェクトの名前で十分です。次の例では，オブジェクトのタイプは Browser で，オブジェクトの名前は「Welcome: Mercury Tours」です。

```
Browser("Welcome:Mercury Tours")
```

ヒント：テスト・オブジェクトおよびメソッドの名前は大文字小文字が区別されません。

オブジェクトの階層では，オブジェクトはピリオドで区切られます。次の例では，Browser と Page は同じ階層構造内の 2 つの別々のオブジェクトです。

```
Browser("Welcome:Mercury Tours").Page("Welcome:Mercury Tours")
```

オブジェクトに対して実行される操作（メソッド）は，常にステートメントの末尾に，操作に関係する値が後ろに続く形式で表示されます。次の例では，**Set** メソッドを使って，「userName」エディット・ボックスに「mercury」という文字列を挿入しています。

```
Browser("Welcome:Mercury Tours").Page("Welcome:Mercury Tours").
WebEdit("userName").Set "mercury"
```

テストを記録すると，QuickTest では，アプリケーション内のオブジェクトを対象として実行した操作が記録されます。

QuickTest 内のオブジェクトは，環境によって分けられます。QuickTest 環境は標準の Windows オブジェクト，Visual Basic オブジェクト，ActiveX オブジェクト，Web オブジェクトに加え，他の環境からのオブジェクトを外部アドインとして含みます。

ほとんどのオブジェクトには，対応するメソッドがあります。たとえば，**Back** メソッドは **Browser** オブジェクトに関連付けられています。

オブジェクトと，それに関連付けられているメソッドとプロパティの完全なリストを参照するには，［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］を選択し，［**目次**］タブで［**Object Model Reference**］を開きます。

メソッドを使用して操作を行うステップを追加する方法の詳細については，120 ページ「エキスパート・ビューまたは関数ライブラリでのステートメントの生成」を参照してください。

VBScript を使用する方法の詳細については，136 ページ「VBScript の基本的な構文の理解」を参照してください。

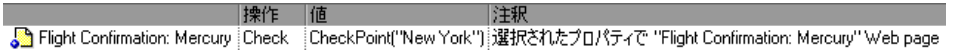
チェックポイント・ステートメントおよび出力ステートメントについて

QuickTest で，ページ，テキスト文字列，テーブル，およびその他のオブジェクトを対象としたチェックポイントおよび出力値を作成できます。キーワード・ビューでチェックポイントまたは出力値を作成すると，QuickTest によって，対応する VBScript 行がエキスパート・ビューに作成されます。チェックポイントの実行には **Check** メソッドが使用され，出力値ステップの実行には **Output** メソッドが使用されます。

たとえば，次のステートメントでは，QuickTest によって「New York」という文字列が検査されます。

```
Browser("Mercury Tours").Page("Flight Confirmation"). Check Checkpoint("New York")
```

キーワード・ビューでは，対応するステップが次のように表示されます。



注：

チェックポイントの詳細は，対応する［チェックポイントのプロパティ］ダイアログ・ボックスで設定され，検査対象のオブジェクトとともに保存されます。出力値の詳細は，対応する［出力値のプロパティ］ダイアログ・ボックスで設定され，値を出力する元となるオブジェクトとともに保存されます。エキスパート・ビューに表示されるステートメントは，格納されている情報への参照です。したがって，エキスパート・ビューに手作業でチェックポイント・ステートメントおよび出力値ステートメントを挿入することはできません。また，エキスパート・ビューから別のテストに **Checkpoint** ステートメントおよび **Output** ステートメントをコピーすることもできません。

チェックポイントを挿入または変更する方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 7 章「チェックポイントについて」を参照してください。出力値を挿入または変更する方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 16 章「値の出力」を参照してください。

パラメータ指定について

QuickTest を使って，テストの値をパラメータ化することにより，テストを拡張できます。「パラメータ」とは，外部のデータ・ソースまたはジェネレータから値が代入される変数です。

キーワード・ビューでパラメータを作成すると，対応する VBScript 行がエキスパート・ビューに作成されます。

たとえば，メソッド引数の値としてデータ・テーブル・パラメータとして定義した場合，次の構文を使用してデータ・テーブルから値が取得されます。

Object_Hierarchy.Method DataTable (parameterID, sheetID)

項目	詳細
Object_Hierarchy	テスト・オブジェクトの階層定義。ピリオドで区切られた 1 つ以上のオブジェクトの並びから成ります。
Method	パラメータ化されたオブジェクトを対象に QuickTest によって実行されるメソッドの名前。
DataTable	データテーブルを表す予約済みオブジェクト。
parameterID	値の取得先となるデータ・テーブル内のカラムの名前。
sheetID	値が格納されているシートの名前。パラメータがグローバル・パラメータである場合は，dtGlobalSheet が sheetID となります。

たとえば，Mercury Tours サイトを対象にテストを記録しているときに，目的地として「San Fransisco」を選択するとします。エキスパート・ビューで，次のステートメントがテストに挿入されます。

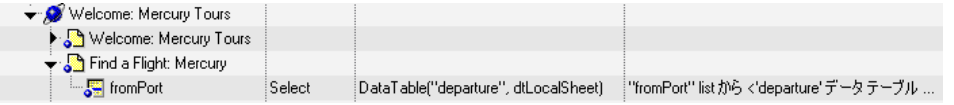
```
Browser("Welcome:Mercury").Page("Find a Flight:").WebList("toPort").Select
"San Francisco"
```

ここで，目的地の値をパラメータ化し，データ・テーブル内に「**Destination**」カラムを作成したとします。前のステートメントが次のように変更されます。

```
Browser("Welcome:Mercury").Page("Find a Flight:").WebList("toPort").Select
DataTable("Destination",dtGlobalSheet)
```

ここで，**Select** はメソッド名，DataTable はデータ・テーブルを表すオブジェクト，Destination はデータ・テーブルのカラムの名前，dtGlobalSheet はデータ・テーブルのシートの名前です。

キーワード・ビューでは，このステップが次のように表示されます。



パラメータ値の使用方法と定義方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 15 章「値のパラメータ化」を参照してください。

エキスパート・ビューまたは関数ライブラリでのステートメントの生成

ステートメントを生成するには，次のいずれかの方法を使用します。

- ステップ・ジェネレータを使用して，メソッドおよび関数を使用するステップを追加できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 531 ページ「ステップ・ジェネレータを使用したステップの挿入」を参照してください。
- メソッドを使用して操作を行う VBScript ステートメントを手作業で挿入できます。QuickTest は，ステートメントで使用するテスト・オブジェクト，メソッド，プロパティ，またはコレクションを選択し，エキスパート・ビューまたは関数ライブラリでの入力に合わせて対応する構文が表示されるステートメント補完（IntelliSense）機能を備えています。詳細については，次の「オブジェクトを対象としたステートメントの生成」を参照してください。
- エキスパート・ビューまたは関数ライブラリで VBScript のキーワードを入力し始めると，［**VBScript 構文を自動的に拡張する**］オプションが有効になっていれば，対応する構文またはブロックがスクリプトに追加されます。詳細については，125 ページ「VBScript 構文の自動補完」を参照してください。

オブジェクトを対象としたステートメントの生成

エキスパート・ビューまたは関数ライブラリで入力すると，IntelliSense（QuickTest のステートメント補完機能）により，ステートメントで使用するテスト・オブジェクト，メソッド，プロパティ，またはコレクションをドロップダウン・リストからの選択し，対応する構文を表示できます。

［**ステートメントの自動補完を行う**］オプションは標準で有効になっています。このオプションは，［エディタ オプション］ダイアログ・ボックスで設定と解除ができます。詳細については，第 11 章「［エキスパート ビュー］および関数ライブラリ・ウィンドウのカスタマイズ」を参照してください。

［**ステートメントの自動補完を行う**］オプションが有効になっていると，次のようになります。

- Page(というように，オブジェクトに続いて開き括弧「**(**」を入力すると，そのオブジェクトのタイプに一致する，オブジェクト・リポジトリ内の全テスト・オブジェクトのリストが 表示されます。オブジェクト・リポジトリに，当該タイプに一致するオブジェクトが 1 つだけある場合には，そのオブジェクトの名前が開き括弧の後ろに，引用符で囲まれた状態で 自動的に挿入されます。
- ステートメントの中でテスト・オブジェクトの後にピリオドを入力すると，入力したオブジェクトの後に追加できるテスト・オブジェクト，メソッド，プロパティ，コレクションおよび登録済みの関数のリストが 表示されます。
- メソッドまたはプロパティの名前を入力すると，使用可能なメソッドおよびプロパティのリストが 表示されます。CTRL + SPACE キーを押すと，選択肢が 1 つしかなければキーワードが自動的に補完されます。選択肢が複数ある場合は，入力したテキストに一致する（アルファベット順で）最初のメソッドまたはプロパティが強調表示されます。
- メソッドまたはプロパティを入力すると，メソッドまたはプロパティの構文が，その必須引数および任意引数とともに 表示されます。メソッドまたはプロパティを使用するステップを追加するとき，メソッドまたはプロパティの必須引数に対して値を定義する必要があります。
- CTRL+SPACE を押すと，追加が可能な対応するテスト・オブジェクト，メソッド，プロパティ，コレクション，VBScript 関数，ユーザ定義関数，VBScript 定数，およびユーティリティ・オブジェクトのリストが 表示されます。このリストは，オブジェクト・リポジトリにまだ追加されていないオブジェクトを入力した場合にも表示されます。テストに関数が含まれている場合，またはテストが関数ライブラリと関連付けられている場合は，その関数もリストに表示されます。

➤ **Object** プロパティをステートメントの中で使用すると，オブジェクトのデータが ActiveScreen，または開いているアプリケーションから現在利用できる場合には，アプリケーションの任意の実行環境オブジェクトのネイティブ・メソッドが 表示されます。**Object** プロパティの詳細については，165 ページ「実行環境オブジェクトのプロパティおよびメソッドへのアクセス」を参照してください。

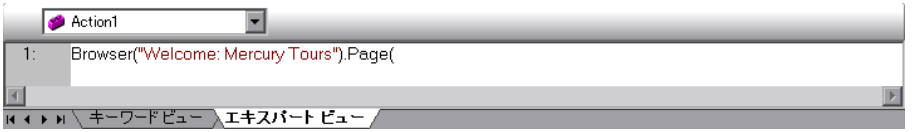
エキスパート・ビューまたは関数ライブラリでステートメントの自動補完機能を使用してステートメントを生成するには，次の手順を実行します。

1 ［**ステートメントの自動補完を行う**］オプションが選択されていることを確認します（［**ツール**］＞［**表示オプション**］＞［**一般**］タブ）。

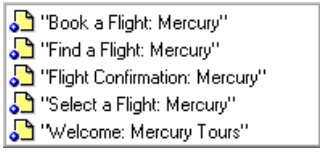
2 次の手順のいずれかを実行します。

➤ 関数ライブラリで作業をしている場合は，手順 4 に進みます。

➤ エキスパート・ビューで作業をしている場合は，オブジェクトに続いて開き括弧「**(**」を入力します。



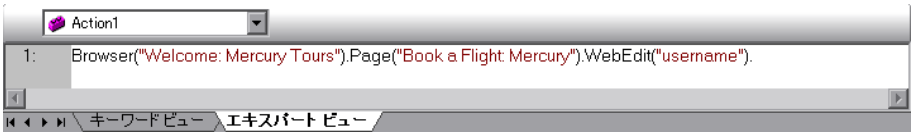
オブジェクト・リポジトリに，当該タイプに一致するオブジェクトが 1 つだけある場合には，そのオブジェクトの名前が開き括弧の後ろに，引用符で囲まれた状態で QuickTest によって自動的に挿入されます。当該タイプのオブジェクトがオブジェクト・リポジトリに複数存在する場合には，QuickTest によってそれらがリストに表示されます。



3 リスト内のオブジェクトをダブルクリックするか，矢印キーを使ってオブジェクトを選択し，ENTER キーを押します。QuickTest によってオブジェクトがステートメントに挿入されます。

4 次の手順のいずれかを実行します。

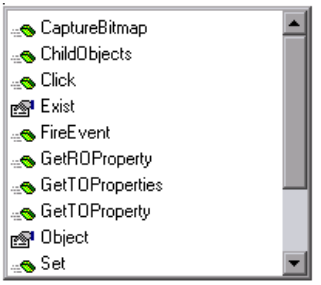
➤ エキスパート・ビューで作業をしている場合は，メソッドを実行する対象となるオブジェクトの後に，ピリオド（.）を入力します。



➤ 関数ライブラリで作業している場合は，次の例のようにオブジェクトの階層全体を入力します。

```
Browser("Welcome:Mercury Tours").Page("Book a Flight:
    Mercury).WebEdit("username").
```

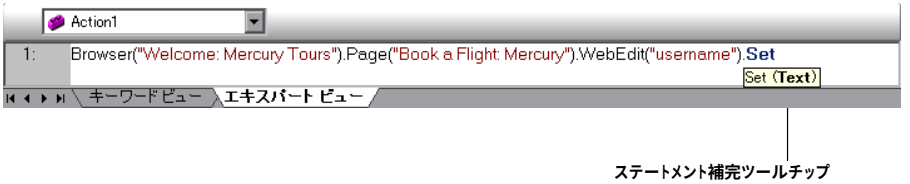
1 ("username"). のように，オブジェクト記述の後ろにピリオド（.）を入力すると，オブジェクトに対して利用できるメソッドおよびプロパティのリストが 表示されます。



ヒント：ピリオドの後，またはメソッド名またはプロパティ名の入力を始めてから CTRL + SPACE キーを押すか，［**編集**］＞［**詳細設定**］＞［**単語入力候補**］を選択します。入力したテキストに一致するメソッドまたはプロパティが 1 つのみの場合は，メソッド名またはプロパティ名が自動的に補完されます。入力したテキストに一致するメソッドまたはプロパティが複数ある場合には，入力したテキストに一致する（アルファベット順で）最初のメソッドまたはプロパティが強調表示されます。

---

2 リスト内のメソッド，またはプロパティをダブルクリックするか，矢印キーを使ってメソッドまたはプロパティを選択し，ENTER キーを押します。QuickTest によって，メソッドまたはプロパティがステートメントに挿入されます。次に示すエキスパート・ビューの例のように，メソッドまたはプロパティに引数が含まれている場合は，ツールチップにそのメソッドまたはプロパティの構文が表示されます。

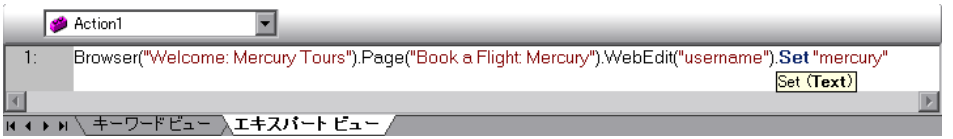


上の例では，**Set** メソッドには **Text** という引数が 1 つあります。この引数名は，ボックスの中に挿入するテキストを表します。

ヒント：また，引数を含む任意のメソッドまたは関数にカーソルを置き，CTRL + SHIFT + SPACE キーを押すか，［**編集**］＞［**詳細**］＞［**引数詳細**］を選択すると，その項目に対応するステートメント補完（引数の構文の）ツールチップが表示されます。

---

3 表示される構文に従って，メソッドの後ろにメソッド引数を入力します。



注：エキスパート・ビューでステップを追加した後，追加した新しいステップをキーワード・ビューに表示することができます。エキスパート・ビューに追加したステートメントに構文エラーがある場合，［キーワード ビュー］を選択したときに それらのエラーが情報表示枠に表示されます。詳細については，142 ページ「VBScript 構文エラーの処理方法」を参照してください。

QuickTest メソッドの詳細と用例については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

VBScript の構文の詳細については，136 ページ「VBScript の基本的な構文の理解」を参照してください。

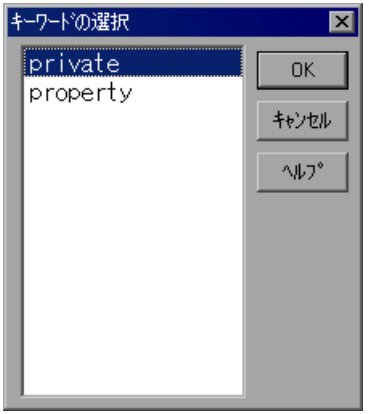
VBScript 構文の自動補完

エキスパート・ビューまたは関数ライブラリで VBScript のキーワードを入力し始めたとき，［**VBScript 構文を自動的に拡張する**］オプションが有効になっていれば，キーワードの最初の 2 文字が QuickTest によって自動的に認識され，対応する VBScript 構文またはブロックがスクリプトに追加されます。たとえば，行頭で「if」という文字に続いてスペースを入力すると，自動的に次の構文が入力されます。

```
If Then
End If
```

［**VBScript 構文を自動的に拡張する**］オプションは標準で有効になっています。このオプションは，［エディタ オプション］ダイアログ・ボックスで設定と解除ができます。詳細については，318 ページ「エディタの動作のカスタマイズ」を参照してください。

入力した 2 文字が複数のキーワードの最初の 2 文字に一致する場合，［キーワードの選択］ダイアログ・ボックスが表示されるので，使用したいキーワードを選択できます。たとえば，「pr」という文字に続いてスペースを入力すると，private および property というキーワードを含んだ［キーワードの選択］ダイアログ・ボックスが表示されます。



そこでリストからキーワードを選択して［**OK**］をクリックします。対応する VBScript 構文またはブロックが スクリプトに自動的に挿入されます。

VBScript の構文の詳細については，136 ページ「VBScript の基本的な構文の理解」を参照してください。

エキスパート・ビューおよび関数ライブラリ内での操作

［移動］ダイアログ・ボックスまたはブックマークを使って，エキスパート・ビューまたは関数ライブラリ内の特定の行に移動できます。また，エキスパート・ビューまたは関数ライブラリ内の特定の文字列を検索し，必要があれば，その文字列を別の文字列で置換することが可能です。これらのオプションを使えば，長いアクションや関数のいくつものセクションの間を移動するのが容易になります。

注：テストが対象の場合，エキスパート・ビューにはアクションが 1 つのみ表示されます。この項で説明する操作機能は，テスト全体ではなく，現在選択されているアクションを対象とするものです。

［移動］ダイアログ・ボックスの使用

［移動］ダイアログ・ボックスを使用して，アクション内または関数ライブラリ内の特定の行に移動できます。

ヒント：標準設定では，エキスパート・ビューおよび関数ライブラリに行番号が表示されます。行番号が表示されない場合は，［**ツール**］＞［**表示オプション**］＞［**一般**］タブの［**行番号を表示する**］オプションを選択します。エディタ・オプションの詳細については，第 11 章「［エキスパート ビュー］および関数ライブラリ・ウィンドウのカスタマイズ」を参照してください。

［移動］ダイアログ・ボックスを使ってエキスパート・ビューまたは関数ライブラリ内の特定の行に移動するには，次の手順を実行します。

1 ［エキスパート ビュー］タブをクリックするか，関数ライブラリを有効にします。

2 ［**移動**］ボタンをクリックするか，［**編集**］＞［**移動**］を選択します。［移動］ダイアログ・ボックスが表示されます。

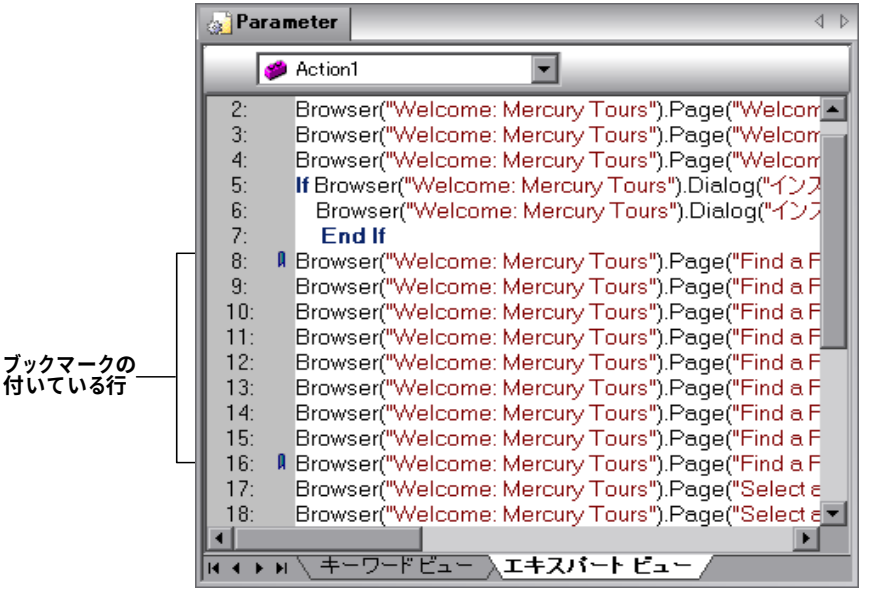
3 ［**行番号**］ボックスに移動先の行番号を入力し，［**OK**］をクリックします。指定した行の先頭にカーソルが移動します。

ブックマークの使用

ブックマークを使って，アクションまたは関数ライブラリの中の重要なセクションにマークを付けることで，さまざまな部分の間を簡単に移動できます。テストの中では，ブックマークは特定のアクション内でのみ使用できます。アクション間を移動するときには維持されず，テストまたは関数ライブラリとともに保存されません。

ブックマークを割り当てると，エキスパート・ビューまたは関数ライブラリ内で選択した行の左に，アイコンが付加されます。［ブックマーク］ダイアログ・ボックスの［**移動**］ボタンを使用して，ブックマークの付いている行に移動できます。

ブックマークの見え方はテストでも関数ライブラリでも同じです。次の例ではテスト内のアクションに 2 つのブックマークを追加しています。

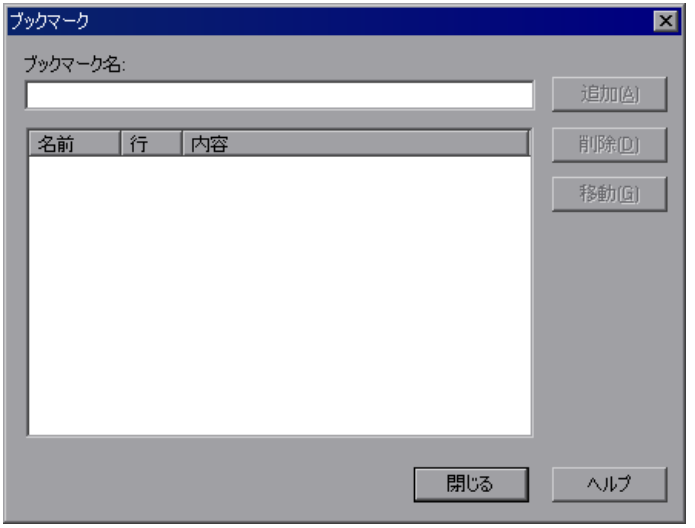


ブックマークを設定するには，次の手順を実行します。

1 ［**エキスパート ビュー**］タブをクリックするか，関数ライブラリを有効にします。

2 ブックマークを設定する対象となる行をクリックします。

3 ［**編集**］＞［**ブックマーク**］を選びます。［ブックマーク］ダイアログ・ボックスが表示されます。



4 ［**ブックマーク名**］フィールドに，一意の名前を入力し，［**追加**］をクリックします。ブックマークが，対象となる行の行番号とそのテキスト内容とともに［ブックマーク］ダイアログ・ボックスに追加されます。また，エキスパート・ビューまたは関数ライブラリ内で選択した行の左に，ブックマーク・アイコン が付加されます。

5 ブックマークを削除するには，対象ブックマークを選択し，［**削除**］をクリックします。

特定のブックマークに移動するには，次の手順を実行します。

1 ［**エキスパート ビュー**］タブをクリックするか，関数ライブラリを有効にします。

2 ［**編集**］＞［**ブックマーク**］を選択します。［ブックマーク］ダイアログ・ボックスが表示されます。

3 リストからブックマークを選択して［**移動**］ボタンをクリックします。QuickTest が現在のアクションまたは関数ライブラリの該当する行に移動します。

ヒント：標準設定では，エキスパート・ビューおよび関数ライブラリに行番号が表示されます。行番号が表示されない場合は，［**ツール**］＞［**表示オプション**］＞［**一般**］タブの［**行番号を表示する**］オプションを選択します。エディタ・オプションの詳細については，第 11 章「[エキスパート ビュー］および関数ライブラリ・ウィンドウのカスタマイズ」を参照してください。

テキスト文字列の検索

エキスパート・ビュー内の現在のアクションの中または関数ライブラリ内で，検索する文字列を指定できます。また，［HTML ソースの編集］ダイアログ・ボックスおよび［HTML タグの編集］ダイアログ・ボックスのほか，［スクリプトに "With" を追加］ダイアログ・ボックスの中の文字列も検索できます。リテラル・テキストを検索することも，正規表現を使用した高度な検索を行うこともできます。また，他のオプションを使用して検索結果の絞込みを行うことも可能です。

文字列を検索するには，次の手順を実行します。

1 エキスパート・ビューまたは関数ライブラリで，次のいずれかを実行します。

➤ ［**検索**］ボタンをクリックします。

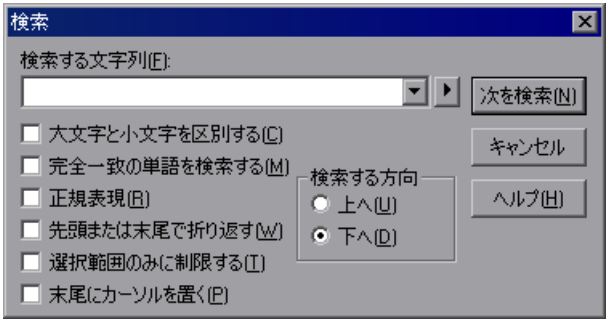
➤ ［**編集**］＞［**検索**］を選択します。

ヒント：エキスパート・ビューでは，次のいずれかも実行できます。

［**編集**］＞［**詳細設定**］＞［**スクリプトに "With" を追加**］を選択してから CTRL + F キーを押します。

［ページ チェックポイントのプロパティ］ダイアログ・ボックスで，［**HTML ソースの編集**］または［ **HTML タグの編集**］を選択してから右クリックし，表示されたダイアログ・ボックスで［**検索**］を選択します。

［検索］ダイアログ・ボックスが開きます。



2 ［**検索する文字列**］ボックスに，検索する文字列を入力します。

3 指定する文字列の中で正規表現を使用したい場合には，矢印ボタン をクリックして正規表現を選択します。リストから正規表現を選択すると，その表現が［**検索する文字列**］ボックス内のカーソルの位置に自動的に挿入されます。詳細については，135 ページ「[検索］および［置換］ダイアログ・ボックスにおける正規表現の使用」を参照してください。

4 次の任意のオプションを選択して検索結果の絞込みを行うことも可能です。

- ［**大文字と小文字を区別する**］：検索の際に大文字と小文字を区別します。［**大文字と小文字を区別する**］を選択した場合，大文字小文字が，［**検索する文字列**］ボックスに入力した文字列と正確に一致する対象のみが QuickTest によって検索されます。
- ［**完全一致の単語を検索する**］：単語の一部ではなく単語全体が一致する文字列を検索します。

- ［**正規表現**］：指定した文字列が正規表現として処理されます。リストから正規表現を選択した場合には，このオプションが自動的に選択されます。
- ［**先頭または末尾で折り返す**］：検索の方向に応じて，検索がアクション，ダイアログ・ボックス，または関数ライブラリの先頭または末尾に達したときに，それらの先頭または末尾から検索を続けます。
- ［**選択範囲のみに制限する**］：アクション，ダイアログ・ボックス，または関数ライブラリの中で選択されているテキストの範囲内に限定して検索を行います。
- ［**末尾にカーソルを置く**］：検索対象文字列が見つかったときに，その文字列を強調表示して，文字列の末尾にカーソルを移動します。

5 アクション，ダイアログ・ボックス，または関数ライブラリ内の現在のカーソルの位置からどちらの方向に向かって検索を行うか指定します。［**上へ**］または［**下へ**］のいずれかを選択できます。

6 現在のアクションまたはダイアログ・ボックスの中，あるいはアクティブな関数ライブラリの中で，検索文字列の次の出現を強調表示するには，［**次を検索**］をクリックします。

テキスト文字列の置換

エキスパート・ビュー内の現在のアクションまたは現在の関数ライブラリの中で検索する文字列，およびそれらの文字列を置き換えるための文字列を指定できます。［HTML ソースの編集］ダイアログ・ボックスおよび［HTML タグの編集］ダイアログ・ボックス内の文字列を検索して置換することも可能です。リテラル・テキストを検索して置換することも，正規表現を使用した高度な処理を行うこともできます。また，他のオプションを使用して検索と置換の処理を詳細に設定することも可能です。

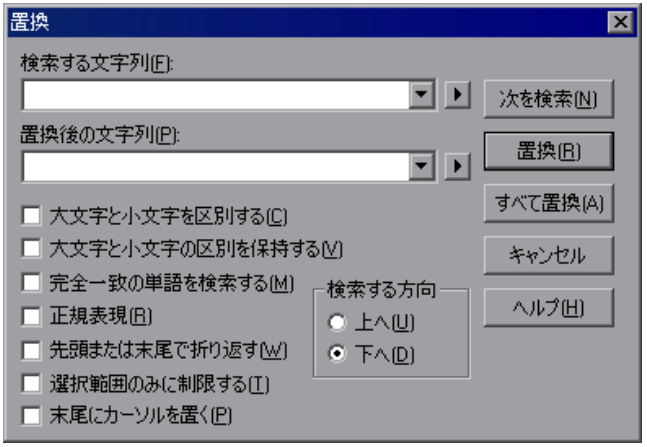
文字列を置換するには，次の手順を実行します。

1 エキスパート・ビューまたは関数ライブラリで，次のいずれかを実行します。

- ［**置換**］ボタンをクリックします。
- ［**編集**］＞［**置換**］を選択します。

ヒント：（テストの場合のみ）［ページ チェックポイントのプロパティ］ダイアログ・ボックスで，［**HTML ソースの編集**］または［ **HTML タグの編集**］を選択してから右クリックし，表示されたダイアログ・ボックスで［**置換**］を選択します。

［置換］ダイアログ・ボックスが表示されます。



2 ［**検索する文字列**］ボックスに，検索する文字列を入力します。

3 ［**置換後の文字列** ］ボックスに，見つかったテキストを置き換える文字列を入力します。

4 ［**検索する文字列**］または［**置換後の文字列**］に指定する文字列の中で正規表現を使用したい場合には，矢印ボタン ▶ をクリックして正規表現を選択します。リストから正規表現を選択すると，その表現が［**検索する文字列**］または［**置換後の文字列**］ボックス内のカーソルの位置に自動的に挿入されます。詳細については，135 ページ「[検索］および［置換］ダイアログ・ボックスにおける正規表現の使用」を参照してください。

5 次の任意のオプションを選択して検索結果の絞込みを行うことも可能です。

➤ ［**大文字と小文字を区別する**］：検索の際に大文字と小文字を区別します。［**大文字と小文字を区別する**］を選択した場合，大文字小文字が，［**検索する文字列**］ボックスに入力した文字列と正確に一致する対象のみが QuickTest によって検索されます。

- ［**大文字と小文字の区別を保持する**］：［**検索する文字列**］に指定した文字列について，全部小文字，全部大文字，先頭のみ大文字，大文字小文字入り混じりのそれぞれを調べます。［**置換後の文字列**］に指定した文字列は，大文字小文字が入り混じっている場合を除き，見つかった文字列と同じ大文字小文字に変換されます。大文字小文字が入り混じっている場合，［**置換後の文字列**］の文字列が変換されずにそのまま使用されます。
- ［**完全一致の単語を検索する**］：単語の一部ではなく単語全体が一致する文字列を検索します。
- ［**正規表現**］：指定した文字列が正規表現として処理されます。リストから正規表現を選択した場合には，このオプションが自動的に選択されます。
- ［**先頭または末尾で折り返す**］：検索の方向に応じて，検索がアクション，ダイアログ・ボックス，または関数ライブラリの先頭または末尾に達したときに，それらの先頭または末尾から検索を続けます。
- ［**選択範囲のみに制限する**］：アクション，ダイアログ・ボックス，または関数ライブラリの中で選択されているテキストの範囲内に限定して検索を行います。
- ［**末尾にカーソルを置く**］：検索対象文字列が見つかったときに，その文字列を強調表示して，文字列の末尾にカーソルを移動します。

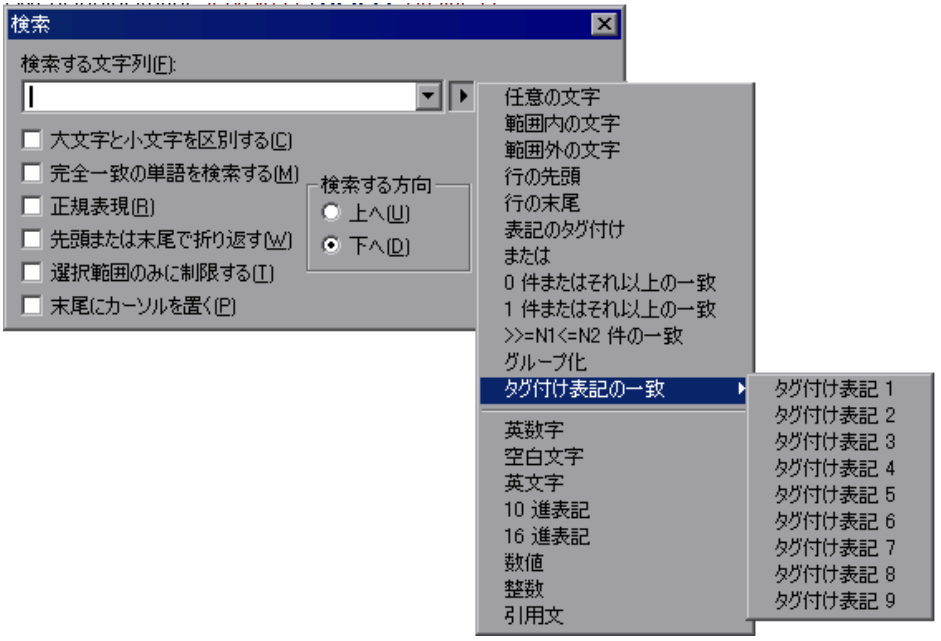
6 現在のアクションまたはダイアログ・ボックスの中，あるいはアクティブな関数ライブラリの中で，検索テキスト文字列の次の出現を強調表示するには，［**次を検索**］をクリックします。

7 強調表示されている文字列を［**置換後の文字列**］ボックスの文字列で置き換えるには［**置換**］をクリックします。現在のアクションまたはダイアログ・ボックス，あるいはアクティブな関数ライブラリにおいて，［**検索する文字列**］ボックスに指定した文字列のすべての出現を［**置換後の文字列**］ボックスの文字列で置き換えるには，［**すべて置換**］をクリックします。

［検索］および［置換］ダイアログ・ボックスにおける正規表現の使用

［**検索する文字列**］および［**置換後の文字列**］の文字列の中で正規表現を使用して検索を拡張できます。正規表現の概要ついては，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 345 ページ「正規表現の使用について」を参照してください。［検索］および［置換］ダイアログ・ボックスで使用できる表現と QuickTest の他の場所で使用できる表現に違いがあることに注意してください。

選択できる正規表現を表示するには，［検索］または［置換］ダイアログ・ボックスの中で矢印ボタン▶をクリックします。



定義済みの正規表現のリストから選択できます。また，タグ付きの表現も使用できます。文字列の検索に正規表現を使用するとき，見つかった文字列に応じて表現を変えたい場合があります。

たとえば，**(save¥:n)¥1** という正規表現を指定して検索をすると，**save** という文字列の後に任意の数字があり，その直後に **save** と，先に見つかった任意の数字があるという文字列が見つかります（つまり，たとえば **save6save6** という文字列が一致し，**save6save7** という文字列は一致しません）。

また，タグ付き表現を使用して，見つかった文字列の一部を置換文字列に取り込むことができます。たとえば，**save(¥:n)** を検索して，**open¥1** で置換することができます。この場合，**save** の後に任の数字がある文字列が検索され，それが **open** と，見つかった数字で置き換えられます。

検索文字列の中でタグ付き表現を示すための括弧「**()**」を挿入するには，正規表現のリストから［**表記のタグ付け**］を選択します。

使用するタグ表現を，「**¥**」に 1 から 9 のいずれかのタグ・グループ番号が続いた形式で指定するには，［**タグ付け表記の一致**］を選択してから対応するタグ・グループ番号を選択します（タグ付き表現の番号を調べるには，検索文字列の開き括弧「**(**」を数えます。最初の（最も左側にある）タグ付き表現は「**¥1**」で最後のは「**¥9**」です）。

VBScript の基本的な構文の理解

VBScript は簡単に覚えられる強力なスクリプト言語です。VBScript を使用すれば，これまでにプログラミングの経験がなくても，オブジェクトを対象とした簡単な作業から複雑な作業まで実行できます。

この項では，VBScript ステートメントを使用して QuickTest テストまたは関数ライブラリを拡張するための基本的なガイドラインをいくつか示します。VBScript での作業の詳細については，QuickTest の［ヘルプ］メニューから VBScript に関するマニュアルを参照してください（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］）。

VBScript の各ステートメントには個別の構文規則があります。これらの規則に従わないと，問題のあるステップを実行したときにエラーが生成されます。加えて，エキスパート・ビューからキーワード・ビューに切り替えるときに，情報表示枠のドキュメント内に構文エラーがあれば，QuickTest によってそれらがリスト表示されます。構文エラーを修正してなくさない限り，キーワード・ビューに切り替えることはできません。詳細については，142 ページ「VBScript 構文エラーの処理方法」を参照してください。

ヒント：現在のドキュメントの構文は，［**構文チェック**］ボタンをクリックするか，［**ツール**］＞［**構文チェック**］を選択することにより，いつでも確認できます。テストが開いている場合は，すべてのアクションの構文が確認されます。関数ライブラリが開いている場合は，ライブラリ・スクリプトの構文が確認されます。

エキスパート・ビューまたは関数ライブラリで作業をするときは，次に示す VBScript の一般構文規則とガイドラインに留意してください。

- **大文字と小文字の区別**：標準では VBScript は変数，オブジェクト，メソッドの名前，および定数などの大文字と小文字を区別しません。

  たとえば，次の 2 つのステートメントは VBScript においては同じです。

  ```
  Browser("Mercury").Page("Find a Flight:").WebList("toDay").Select "31"
  browser("mercury").page("find a flight:").weblist("today").select "31"
  ```

- **テキスト文字列**：値をテキスト文字列として入力するとき，文字列の前後に引用符を追加しなければなりません。たとえば，先に示したスクリプト行では，Web サイト，Web ページ，およびエディット・ボックスの名前はすべて引用符で囲まれたテキスト文字列です。

  31 の値が引用符で囲まれているのは，それが数値ではなく，数字を表すテキスト文字列だからです。

  次の例では，プロパティ名（最初の引数）のみがテキスト文字列で，引用符で囲まれています。2 番目の引数（プロパティの値）は変数なので引用符で囲まれていません。3 番目の引数（タイムアウト値を指定）は数値なので，これも引用符が不要です。

  ```
  Browser("Mercury").Page("Find a Flight:").WaitProperty("items count",
      Total_Items, 2000)
  ```

- **変数**：文字列，整数，配列，オブジェクトを格納するための変数を指定できます。変数を使用することで，スクリプトが読みやすくなり柔軟性が高くなります。詳細については，次の「変数の使用」を参照してください。

- **括弧**：望む結果を得てエラーを避けるには，ステートメントの中で括弧を正しく使用することが重要です。詳細については，139 ページ「括弧の使用」を参照してください。

- **インデント**：スクリプトをステートメントの論理構造およびネストに合わせてインデントをしたりインデントを解除したりできます。詳細については，141 ページ「VBScript テキストの書式設定」を参照してください。
- **コメント**：ステートメントにコメントを追加するには，単一引用符（'）を独立の行の先頭で使用するか，ステートメントの末尾で使用します。スクリプトを分かりやすくして保守しやすいように，可能限りのコメントを追加することをお勧めします。詳細については，141 ページ「VBScript テキストの書式設定」および 156 ページ「コメントの挿入」を参照してください。
- **スペース**：スペースを追加することでスクリプトを分かりやすくすることができます。これらのスペースは VBScript によって無視されます。

個々の VBScript を使用してテストまたは関数ライブラリを拡張する方法の詳細については，155 ページ「コメント，フロー制御，その他の VBScript ステートメントの使用」を参照してください。

変数の使用

テストまたは関数ライブラリの中でテスト・オブジェクトや単純な値を格納するための変数を指定することができます。テスト・オブジェクトを変数に格納する場合，他のステートメントの中でオブジェクト階層全体を指定することの代わりに変数を使用できます。変数をこのように使用すれば，ステートメントが読みやすくなり，保守もしやすくなります。

オブジェクトを格納する変数を指定するには，**Set** ステートメントを次の構文で使用します。

Set ObjectVar = ObjectHierarchy

次の例では，**Set** ステートメントを使用して UserEditBox 変数に，**username** エディット・ボックスの Browser ＞ Page ＞ WebEdit オブジェクト階層全体を格納するよう指定しています。その後，UserEditBox 変数を対象に **Set** メソッドを使用して，**username** エディット・ボックスに John という値を入力しています。

```
Set UserEditBox = Browser("Mercury Tours").Page("Mercury Tours").
    WebEdit("username")
UserEditBox.Set "John"
```

注：単純な値（文字列や数字など）を格納する変数の指定には **Set** ステートメントを使用しないでください。次の例は，単純な値のための変数を定義する方法を示します。

```
MyVar = Browser("Mercury Tours").Page("Mercury Tours").
    WebEdit("username").GetTOProperty("type")
```

Dim ステートメントを使用して，文字列，整数，配列など，他の型の変数を宣言できます。このステートメントは必須ではありませんが，テストまたは関数ライブラリの構造を強化するために使用できます。次の例では，**Dim** ステートメントを使用して passengers 変数を宣言し，現在のアクションまたは関数ライブラリの他のステートメントの中で使用できるようにしています。

```
Dim passengers
passengers = Browser("Mercury Tours").Page("Find Flights").
   WebEdit("numpassengers").GetROProperty("value")
```

括弧の使用

VBScript でプログラミングをするとき，ステートメント内での括弧「()」の使用・不使用に関する規則に従うことが重要です。

値を返すメソッドを呼び出し，返された値を使用する場合には，メソッドの引数を括弧で囲む必要があります。

たとえば，値を変数に返す場合，メソッドを **If** ステートメントの中で使用する場合，あるいは，**Call** キーワードを使用してアクションまたは関数を呼び出す場合などにメソッド引数を括弧で囲みます。チェックポイントの戻り値を取得したい場合にも，チェックポイント名を括弧で囲む必要があります。

ヒント：テストまたは関数ライブラリの中でステップを実行しているときに **Expected end of statement** エラー・メッセージを受け取った場合には，ステップのメソッドの引数を括弧で囲む必要があるかもしれません。

次に，括弧を使用する場合の例および使用しない場合の例をいくつか示します。

次の例では，**ChildItem** メソッドが値を変数に返すため，メソッドを括弧で囲む必要があります。

```
Set WebEditObj = Browser("Mercury Tours").Page("Method of Payment").
    WebTable("FirstName").ChildItem (8, 2, "WebEdit", 0)
WebEditObj.Set "Example"
```

次の例では，**Call** を使用しているため，メソッドの引数を括弧で囲む必要があります。

```
Call RunAction("BookFlight", oneIteration)
```

または

```
Call MyFunction("Hello World")
...
...
```

次の例では，**If** ステートメントの中でメソッドを使用しているため，**WaitProperty** メソッドの引数を括弧で囲む必要があります。

```
If Browser("index").Page("index").Link("All kind of").
    WaitProperty("attribute/readyState", "complete", 4) Then
    Browser("index").Page("index").Link("All kind of").Click
End If
```

次の例では，**Check** メソッドがチェックポイントの値を返すため，メソッドの引数を括弧で囲む必要があります。

```
a = Browser("MyBrowser").Page("MyPage").Check (CheckPoint("MyProperty"))
```

次の例では，**Click** メソッドが値を返さないため，メソッドの引数を括弧で囲む必要はありません。

```
Browser("Mercury Tours").Page("Method of Payment").WebTable("FirstName").
    Click 3,4
```

VBScript テキストの書式設定

エキスパート・ビューまたは関数ライブラリで作業をするときは，コメントやインデントについて VBScript の慣例に従うことが重要です。

コメントを使用してスクリプトのセクションを説明するようにします。これにより読みやすさが向上し，テストおよび関数ライブラリの保守や更新が容易になります。詳細については，156 ページ「コメントの挿入」を参照してください。

インデントはステートメントの論理構造およびネストを反映するために使用します。

- **コメントの追加**：ステートメントにコメントを追加するには，単一引用符（'）を独立の行の先頭に追加するか，ステートメントの末尾に追加します。

ヒント：

ステートメントをコメントにするには，ステートメントの任意の場所をクリックして，［**コメント ブロック**］ボタンをクリックします。

選択したテキストのブロックをコメントにするには，［**コメント ブロック**］ボタンをクリックするか，［**編集**］＞［**詳細設定**］＞［**コメント ブロック**］を選択します。ブロック内の各行の先頭には単一引用符が付きます。

- **コメントの削除**：ステートメントからコメントを削除するには，独立の行の先頭またはステートメントの末尾の単一引用符（'）を削除します。

ヒント：選択したテキストのブロックまたは行のコメントを解除するには，［**ブロックのコメント解除**］ボタンをクリックするか，［**編集**］＞［**詳細設定**］＞［**ブロックのコメント解除**］を選択します。

- **ステートメントのインデント**：ステートメントをインデントするには，ステートメントを選択して，［**インデント**］ボタンをクリックします。あるいは，テキストを選択して，［**編集**］＞［**詳細設定**］＞［**インデント**］を選択するか，TAB キーを押します。318 ページ「エディタの動作のカスタマイズ」で説明されているように，［エディタ オプション］ダイアログ・ボックスで選択されているタブ間隔に従ってテキストがインデントされます。

注 : [エディタ オプション] ダイアログ・ボックスの [**タブ キーを押して選択されたテキストでインデントを行う**] チェック・ボックスが選択されている必要があります。選択されていない場合，TAB キーを押すと選択したテキストが削除されます。

➤ **ステートメントのインデント解除** : ステートメントのインデントを解除するには，ステートメントを選択して，[**インデント解除**] ボタンをクリックします。あるいは，[**編集**] > [**詳細設定**] > [**インデント解除**] を選択するか，ステートメントの先頭のスペースを削除します。

VBScript での書式設定の詳細については，QuickTest の [ヘルプ] メニューから VBScript に関するマニュアルを参照してください（[**ヘルプ**] > [**QuickTest Professional ヘルプ**] > [**VBScript リファレンス**]）。

VBScript 構文エラーの処理方法

エキスパート・ビューで [キーワード ビュー] タブを選択すると，更新された情報が QuickTest によってキーワード・ビューに表示されます。新規または更新された VBScript ステートメントに構文エラーがある場合，ステータス・バーの右側に「**エラー**」というテキストが赤で点滅し，ステータス・バーに，スクリプトの構文エラーの情報について情報表示枠を確認する必要があることを示すエラー・メッセージが表示されます。QuickTest は，すべての構文エラーが修正されるまで，キーワード・ビューにドキュメントを表示できません。

[VBScript リファレンス] には個々の VBScript エラーの説明が表示されます。詳細については，[**ヘルプ**] > [**QuickTest Professional ヘルプ**] > [**VBScript リファレンス**] > [**VBScript**] > [**リファレンス**] > [**エラー**] > [**VBScript 構文エラー**] を選択してください。

ヒント：

現在のドキュメントの構文は，［**構文チェック**］ボタンをクリックするか，［**ツール**］＞［**構文チェック**］を選択することで，いつでも確認できます。テストが開いている場合は，すべてのアクションの構文が確認されます。関数ライブラリが開いている場合は，ライブラリ・スクリプトの構文が確認されます。

『Microsoft VBScript 言語リファレンス』では，VBScript の構文エラーを次のように定義しています。「ある VBScript ステートメントの構造が VBScript スクリプト言語の 1 つ以上の文法規則に違反した結果として生じるエラー」。VBScript での作業の詳細については，QuickTest の［ヘルプ］メニューから VBScript のリファレンスを参照してください（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］）。

情報表示枠に，ドキュメントの中で見つかった構文エラーの一覧が表示されるので，各構文エラーの場所を調べて修正することができます。

情報

詳細	項目	アクション	行
Expected end of statement	RegExpression	Action1	2
Expected ')'	RegExpression	Action1	6
Expected end of statement	RegExpression	Action1	8
Expected ')'	RegExpression	Action1	10
Expected end of statement	RegExpression	Action1	12
Expected 'End If'	RegExpression	Action1	26

情報表示枠には，各構文エラーについて次の情報が表示されます。

表示枠内の要素	詳細
［詳細］	構文エラーの詳細。たとえば，条件ブロックを **If** ステートメントで開始したけれども **End If** ステートメントで終了しなかった場合，［詳細］には Expected 'End If' と表示されます。 **注：** 状況によっては，QuickTest がエラーを正確に特定できずに，次のようにいくつかの候補を表示することがあります。 Expected 'End Sub', or 'End Function', or 'End Property' この場合，示された行のステートメントを調べてどちらが該当するかを明らかにします。
［項目］	問題のステートメントが含まれているテストまたは関数ライブラリの名前。
［アクション］	問題のステートメントが含まれているアクションの名前。このカラムは，アプリケーション領域を通じてビジネス・コンポーネントに関連付けられている関数ライブラリには適用されません。
［行］	構文エラーが含まれている行。行の番号は各アクションまたは関数ライブラリの先頭から数えられます。

情報表示枠の使用

- 構文エラーの詳細の上にマウスのポインタを合せたままにすると，現在不正である構文が表示されます。
- 特定の構文エラーを含む行に移動するには，情報表示枠内で構文エラーをダブルクリックします。
- 情報表示枠内のカラムのカラム・ヘッダをドラッグしてサイズを変更し，情報を読みやすくすることができます。
- 情報表示枠内の詳細情報を昇順または降順でソートするには，カラム・ヘッダをクリックします。
- 情報表示枠内のエラーで F1 キーを押すと，VBScript 構文エラーに関する情報が表示されます。

プログラム的記述の使用

オブジェクトに対する操作を記録すると，QuickTest により，適切なテスト・オブジェクトがオブジェクト・リポジトリに追加されます。オブジェクトがオブジェクト・リポジトリに追加されたら，エキスパート・ビューでステートメントを追加することで，そのオブジェクトに対して追加のメソッドを実行できます。ステートメントを追加するには，通常，各オブジェクトの名前（大文字小文字は区別されない）をそのオブジェクトの階層にオブジェクト記述として入力した後で，適切なメソッドを追加します。

たとえば，次に示すステートメントでは，「username」はエディット・ボックスの名前です。このエディット・ボックスは「Mercury Tours」という名前を持つページ上にあり，このページは「Mercury Tours」という名前を持つブラウザで記録されたものです。

```
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username")
```

オブジェクト・リポジトリ内の各オブジェクトは一意の名前を持っているので，指定する必要があるのはこの名前だけです。実行セッションの実行中，QuickTest はオブジェクト・リポジトリの中で名前と親オブジェクトに基づいてオブジェクトを検索し，格納されているテスト・オブジェクトの記述を使って，Web サイトまたはアプリケーション内のオブジェクトを識別します。

QuickTest に対して，オブジェクト・リポジトリまたはオブジェクト名を参照せずに，オブジェクトに対するメソッドを実行するように指示できます。これを行うためには，QuickTest に，メソッドの実行対象としたいオブジェクトを識別するために使えるプロパティと値のリストを提供します。

そのようなプログラム的記述は，オブジェクト・リポジトリに格納されていないオブジェクトに対する操作を行う場合に，非常に便利なことがあります。プログラム的記述は，何らかの共通するプロパティを持つ複数のオブジェクトを対象に同じ操作を行う場合や，実行セッション中に動的に決まる記述に適合するプロパティを持つ 1 つのオブジェクトに対する操作を行う場合にも使えます。

たとえば，入力した人名情報に基づいて，雇用主のリストを表示し，リストから選択した雇用主に履歴書を送れるようにする Web サイトのテストをするものとしましょう。テストでは，リストに表示されたすべての雇用主を選択したいのに，テストを設計するときには，ページにいくつのチェック・ボックスが表示されるか分からず，もちろん各チェック・ボックスの正確なオブジェクト記述も知ることができません。こうした状況で，プログラム的記述を使うことで

Set "ON" メソッドを，「HTML TAG = input，TYPE = check box」という記述に適合するすべてのオブジェクトを対象に実行することができます。

プログラム的記述には 2 つのタイプがあります。テスト・ステートメントの中に，オブジェクトを記述するプロパティと値のセットを直接列挙することも，プロパティと値のコレクションを Description オブジェクトに追加してから，ステートメントにその Description オブジェクトの名前を入力することもできます。

オブジェクト記述に対する要求が基本的なものであれば，ステートメントにプログラム的記述を直接入力するほうが簡単でしょう。しかし，ほとんどの場合，Description オブジェクト方式のほうが強力で効率的です。

ステートメントへのプログラム的記述の直接入力

テスト・ステートメントにオブジェクトを直接記述するには，オブジェクトの論理名を指定する代わりに，オブジェクトを記述する **property:=value** のペアを指定します。

一般的な構文は次のとおりです。

```
TestObject("PropertyName1:=PropertyValue1", "..." ,
    "PropertyNameX:=PropertyValueX")
```

TestObject：テスト・オブジェクト・クラスです。

PropertyName:=PropertyValue：テスト・オブジェクトのプロパティとその値です。**property:=value** の各ペアは，カンマと二重引用符で区切る必要があります。

実行セッション中に取得するプロパティ値に基づいてオブジェクトを検索する場合には，プロパティ値として変数名を入力できます。

注：QuickTest はプログラム的記述のプロパティ値をすべて正規表現として評価します。したがって，正規表現において特別な意味を持つ文字（*，?，+ など）を含んだ値を入力するには，¥（円記号）を使用して，その特殊文字をリテラルな文字として扱うように指示します。正規表現の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 345 ページ「正規表現の使用について」を参照してください。

次に示すステートメントでは，Mercury Tours ページに，author という名前と，3 というインデックスを持つ WebEdit テスト・オブジェクトを指定します。実行セッション中，QuickTest は一致するプロパティ値を持つ WebEdit オブジェクトを検索し，「Mark Twain」というテキストを入力します。

```
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("Name:=Author",
    "Index:=3").Set "Mark Twain"
```

注：テスト・オブジェクト階層の特定のポイントからプログラム的記述を使用する場合には，同じステートメント内では，そのポイント以降は必ずプログラム的記述を使用する必要があります。プログラム的記述を使用して階層内のオブジェクトを指定した後に，オブジェクト・リポジトリでの名前を使用してテスト・オブジェクトを指定すると，当該オブジェクトは QuickTest によって識別されません。

たとえば，次の例ではテスト・オブジェクト階層全体を通してプログラム的記述を使用しているので，このステートメントは使用できます。

```
Browser("Title:=Mercury Tours").Page("Title:=Mercury Tours").
    WebEdit("Name:=Author", "Index:=3").Set "Mark Twain"
```

次の例も，特定のポイントから（Page オブジェクト記述以降）プログラム的記述を使用しているので，ステートメントを使用できます。

```
Browser("Mercury Tours").Page("Title:=Mercury Tours").
    WebEdit("Name:=Author", "Index:=3").Set "Mark Twain"
```

しかし，次の例では，Browser および Page オブジェクトについてはプログラム的記述を使用しているものの，WebEdit テスト・オブジェクトについてはオブジェクト・リポジトリでの名前を使おうとしているので，このステートメントは使用できません。

```
Browser("Title:=Mercury Tours").Page("Title:=Mercury Tours").
    WebEdit("Author").Set "Mark Twain"
```

QuickTest は，WebEdit オブジェクトをその名前を使用して特定しようとしますが，親オブジェクトがプログラム的記述を使用して指定されているため，リポジトリの中で当該オブジェクトを見つけることができません。

テスト・オブジェクトを使った作業の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 6 章「テスト・オブジェクトを使用した作業」を参照してください。

1 回のテストまたは 1 つの関数ライブラリで同じプログラム的記述を複数回使用するために，作成したオブジェクトを変数に割り当てることができます。

たとえば，次のように入力する代わりに，

```
Window("Text:=Myfile.txt - Notepad").Move 50, 50
Window("Text:=Myfile.txt - Notepad").WinEdit("AttachedText:=Find what:").
    Set "hello"
Window("Text:=Myfile.txt - Notepad").WinButton("Caption:=Find next").Click
```

次のように入力できます。

```
Set MyWin = Window("Text:=Myfile.txt - Notepad")
MyWin.Move 50, 50
MyWin.WinEdit("AttachedText:=Find what:").Set "hello"
MyWin.WinButton("Caption:=Find next").Click
```

さらに別の方法として，**With** ステートメントを使うこともできます。

```
With Window("Text:=Myfile.txt - Notepad")
    .Move 50, 50
    .WinEdit("AttachedText:=Find what:").Set "hello"
    .WinButton("Caption:=Find next").Click
End With
```

With ステートメントの詳細については，162 ページ「With ステートメント」を参照してください。

プログラム的記述のための Description オブジェクトの使用

Description オブジェクトを使用して，**Property** オブジェクトのセットを格納した **Properties** コレクションを返すことができます。**Property** オブジェクトは，プロパティ名とプロパティ値で構成されます。返された **Properties** コレクションを，ステートメント内でオブジェクト名の代わりに使用できます（各プロパティ・オブジェクトには，プロパティ名とプロパティ値のペアが含まれています）。

注：標準設定では，**Properties** コレクションに追加された **Property** オブジェクトの値はすべて正規表現として処理されます。したがって，正規表現において特別な意味を持つ文字（*，?，+ など）を含んだ値を入力するには，¥（円記号）を使用して，その特殊文字をリテラルな文字として扱うように指示します。正規表現の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 345 ページ「正規表現の使用について」を参照してください。

コレクション内の特定の **Property** オブジェクトの値をリテラル値として指定するには，**RegularExpression** プロパティを False に設定します。詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』の「ユーティリティ」の節を参照してください。

Properties コレクションを作成するには，次の構文を使って **Description.Create** ステートメントを入力します。

```
Set MyDescription = Description.Create()
```

Properties オブジェクト（たとえば上の例の **MyDescription**）を作成すれば，実行セッション中に **Properties** オブジェクトのプロパティと値の追加，編集，削除，取得を行うステートメントを入力できます。これにより，オブジェクト記述にどのプロパティをいくつ含めるかを，実行セッション中に動的に決めることができます。

Properties コレクションに Property オブジェクト（プロパティと値）のセットを設定したら，テスト・ステートメントの中で，この **Properties** オブジェクトをオブジェクト名の代わりに指定できます。

たとえば，次のように入力する代わりに，

```
Window("Error").WinButton("text:=OK", "width:=50").Click
```

次のように入力できます。

```
Set MyDescription = Description.Create()
MyDescription("text").Value = "OK"
MyDescription("width").Value = 50
Window("Error").WinButton(MyDescription).Click
```

ヒント：ActiveX テスト・オブジェクトにプログラム的記述を作成し，対応する実行環境オブジェクトはウィンドウレス（ウィンドウ・ハンドルのないウィンドウ）になった場合は，記述に **windowless** プロパティを追加して，その値を **True** に設定する必要があります。

例を次に示します。

```
Set ButDesc = Description.Create
ButDesc("ProgId").Value = "Forms.CommandButton.1"
ButDesc("Caption").Value = "OK"
ButDesc("Windowless").Value = True
Window("Form1").AcxButton(ButDesc).Click
```

注：テスト・オブジェクト階層の特定のポイントからプログラム的記述を使用する場合には，同じステートメント内では，そのポイント以降は必ずプログラム的記述を使用する必要があります。プログラム的記述を使用して階層内のオブジェクトを記述した後に，オブジェクト・リポジトリでの名前を使用してテスト・オブジェクトを指定すると，当該オブジェクトは QuickTest によって識別されません。

たとえば，Browser(Desc1).Page(Desc1).Link(desc3) ではテスト・オブジェクト階層全体を通してプログラム的記述を使用しているので，このステートメントは使用できます。

Browser("Index").Page(Desc1).Link(desc3) も，特定のポイントから（Page オブジェクト記述以降）プログラム的記述を使用しているので，このステートメントは使用できます。

しかし，Browser(Desc1).Page(Desc1).Link("Example1") の場合，Browser および Page オブジェクトにプログラム的記述を使用する一方で，Link テスト・オブジェクトにオブジェクト・リポジトリでの名前を使用しているので，このステートメントは使用できません（QuickTest によって，オブジェクト名に基づいて Link オブジェクトが検索されますが，プログラム的記述を使用して親オブジェクトを指定しているため，リポジトリの中でオブジェクトが見つかりません）。

Properties オブジェクトを使っているときには，プロパティや値の代わりに変数名を使うことで，実行セッション中に取得したプロパティや値に基づくオブジェクト記述を生成できます。

複数のオブジェクトでプログラム的記述を使いたい場合には，テスト内に複数の **Properties** オブジェクトを作成することもできます。

Description および **Properties** オブジェクト，および関連するメソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

子オブジェクトの取得

ChildObjects メソッドを使って，指定された親オブジェクトの中にあるすべてのオブジェクト，あるいは特定のプログラム的記述に適合する子オブジェクトのみを取得できます。この子オブジェクトのサブセットを取得するには，**Description** オブジェクトを使って，まず記述オブジェクトを作成してから，子オブジェクト・コレクションに適合するプロパティと値の集合を追加します。

注：ChildObjects 記述引数のためのプログラム的記述を作成するには，**Description** オブジェクトを使用します。**property:=value** 構文を使ってプログラム的記述を引数に直接入力することはできません。

記述オブジェクトの中に記述を「構築」したら，次の構文を使ってその記述に適合する子オブジェクトを取得します。

Set MySubSet=TestObject.ChildObjects(MyDescription)

たとえば，次のステートメントは QuickTest に対して，Itinerary Web ページ上のすべてのチェックボックスを選択するよう指示します。

```
Set MyDescription = Description.Create()
MyDescription("html tag").Value = "INPUT"
MyDescription("type").Value = "checkbox"

Set Checkboxes =
Browser("Itinerary").Page("Itinerary").ChildObjects(MyDescription)
NoOfChildObjs = Checkboxes.Count
For Counter=0 to NoOfChildObjs-1
    Checkboxes(Counter).Set "ON"
Next
```

ChildObjects メソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

WebElement オブジェクトに対するプログラム的記述の使用

WebElement オブジェクトを使用すれば，ほかの Mercury テスト・オブジェクト・クラスには適合しない Web オブジェクトを対象にメソッドを実行できます。**WebElement** テスト・オブジェクトが記録されることは決してありませんが，プログラム的記述と **WebElement** オブジェクトを使って，Web サイト内の任意の Web オブジェクトを対象にメソッドを実行できます。

たとえば，次のステートメントを実行するとします。

```
Browser("Mercury Tours").Page("Mercury Tours").
    WebElement("Name:=UserName", "Index:=0").Click
```

または

```
set WebObjDesc = Description.Create()
WebObjDesc("Name").Value = "UserName"
WebObjDesc("Index").Value = "0"
Browser("Mercury Tours").Page("Mercury Tours").WebElement(WebObjDesc).
   Click
```

QuickTest は，Mercury Tours ページにある UserName という名前の最初の Web オブジェクトをクリックする操作を実行します。

WebElement オブジェクトの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

プログラム的記述での Index プロパティの使用

Index プロパティは，オブジェクトを一意に識別するためのテスト・オブジェクト・プロパティとして役立つ場合があります。Index テスト・オブジェクト・プロパティを使うと，オブジェクトはソース・コード内に出現する順序（最初の出現は 0）に基づいて識別されます。

Index プロパティ値は，各オブジェクトに固有の値です。つまり，インデックス値 3 を使って WebEdit テスト・オブジェクトを記述すると，QuickTest によってページ内の 4 番目の WebEdit オブジェクトが検索されます。

これに対し，インデックス値 3 を使って WebElement オブジェクトを記述すると，QuickTest は，タイプには関係なく，そのページの 4 番目の Web オブジェクトが検索されます。これは，WebElement オブジェクトがすべての Web オブジェクトに適用されるためです。

たとえば，次のオブジェクトが含まれるページがあるとします。

- Apple という名前の画像
- UserName という名前の画像
- UserName という名前の WebEdit オブジェクト
- Password という名前の画像
- Password という名前の WebEdit オブジェクト

次の記述は，前述のリストの 3 番目の項目を表します。ページ内で UserName という名前を持つ最初の WebEdit オブジェクトだからです。

```
WebEdit("Name:=UserName", "Index:=0")
```

一方，次の記述は，前述のリストの 2 番目の項目を表します。ページ内で UserName という名前を持つ最初の任意のタイプ（WebElement）のオブジェクトだからです。

注：オブジェクトが 1 つのみの場合，**index=0** を指定しても取得されません。この場合，**Index** プロパティをオブジェクト記述に含めるべきではありません。

プログラムによるアプリケーションの実行と終了

QuickTest では，［記録と実行環境設定］ダイアログ・ボックスを使って，テストの実行開始時にブラウザまたはアプリケーションを起動するように指定したり，テスト対象アプリケーションを手作業で起動したりできます。また，テスト対象アプリケーションを起動または終了するステートメントをテストに挿入することもできます。

指定した場所から任意のアプリケーションを実行するには，**SystemUtil.Run** ステートメントを使います。これはテストに複数のアプリケーションが含まれていて，［記録と実行環境設定］ダイアログ・ボックスで［**開かれている Window ベースのアプリケーションすべてでテストを記録して実行する**］チェック・ボックスを選択している場合に特に便利です。アプリケーションを指定して，サポートされている任意のパラメータを渡したり，ファイル名を指定して，関連付けられているアプリケーションが起動しそのファイルを開くようにできます。

ほとんどのアプリケーションは，**Close** メソッドを使って閉じることができます。

たとえば，次に示すステートメントでは，**type.txt** というファイルを標準のテキスト編集アプリケーション（「メモ帳」など）で開き，happy days と入力してから，ショートカット・キーを使ってファイルを保存し，アプリケーションを終了しています。

```
SystemUtil.Run "C:¥type.txt", """","",""
Window("Text:=type.txt - Notepad").Type "happy days"
Window("Text:=type.txt - Notepad").Type micAltDwn & "F" & micAltUp
Window("Text:=type.txt - Notepad").Type micLShiftDwn & "S" & micLShiftUp
Window("Text:=type.txt - Notepad").Close
```

注：

実行するアプリケーションを［記録と実行環境設定］ダイアログ・ボックスを使用して指定した場合，QuickTest はテストに **SystemUtil.Run** ステートメントを追加しません。

InvokeApplication メソッドは，実行可能ファイルのみを開くことができ，主に下位互換性を維持するために使用します。

詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

コメント，フロー制御，その他の VBScript ステートメントの使用

QuickTest では，テストまたは関数ライブラリの論理フローを制御する条件文を追加することで，テストまたは関数ライブラリに意思決定機能を組み込むことができます。さらに，QuickTest からテスト結果に送信するメッセージをテスト内に定義できます。テストおよび関数ライブラリの読みやすさを向上させるために，それらにコメントを追加することもできます。

キーワード・ビューでこれらのプログラミングの概念を使う方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 20 章「プログラミング・ロジックを含むステップの追加」を参照してください。

注：「**VBScript リファレンス**」（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］メニューから利用できます）には，VBScript，Script Runtime，および Windows Script Host などを含む Microsoft VBScript に関するマニュアルが含まれます。

コメントの挿入

コメントとは，アポストロフィ（'）の後に書かれる，テスト・スクリプトの行または行の一部です。テストを実行しても，QuickTest ではコメントは処理されません。読みやすさを向上させ，テストおよび関数ライブラリの更新を容易にするため，テスト・スクリプトのセクションを説明するコメントを使用することをお勧めします。

次の例では，コメントを使ってその下のステートメントの目的を説明しています。

```
' 文字列 "mercury" を "username" エディット・ボックスに設定する
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").
    Set "mercury"
```

標準設定では，コメントはエキスパート・ビューおよび関数ライブラリに緑色で表示されます。コメントの表示は［エディタ オプション］ダイアログ・ボックスでカスタマイズできます。詳細については，322 ページ「エレメントの見映えのカスタマイズ」を参照してください。

ヒント：

テキストのブロックをコメントにするには，［**編集**］＞［**詳細設定**］＞［**コメント ブロック**］を選択するか，［**コメント ブロック**］ボタンをクリックします。

コメントを解除するには，［**編集**］＞［**詳細設定**］＞［**ブロックのコメント解除**］を選択するか，［**ブロックのコメント解除**］ボタンをクリックします。

注：コメント行を追加するには，VBScript の **Rem** ステートメントを使う方法もあります。詳細については，「Microsoft VBScript Language Reference」（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］＞［**VBScript**］を選択）を参照してください。

計算の実行

数値演算子を使って，簡単な計算を実行するステートメントを作成できます。たとえば，サイトで 2 つのテキスト・ボックスに表示された値を掛け算するには，乗法演算子を使用します。VBScript では，次の数値演算子が使用できます。

演算子	詳細
+	加法
−	減法
−	否定（負の数値－単項）
*	乗法
/	除法
^	指数

次の例では，乗客 1 人あたりの荷物の重さの上限が 100 ポンドの場合の荷物の総重量を計算するために，乗法演算子を使用しています。

```
' GetROProperty メソッドを使って，エディット・ボックスから乗客数を取得する
passenger = Browser ("Mercury_Tours").Page ("Find_Flights").
    WebEdit("numPassengers").GetROProperty("value")
' 乗客数に 100 を乗じる
weight = passenger * 100
' メッセージ・ボックスに重量の上限を挿入する
msgbox(" この団体の荷物重量の上限は "& weight &" ポンドです。")
```

For...Next ステートメント

For...Next ループは，1 つまたは複数のステートメントを，指定した回数だけ実行するように QuickTest に指示します。構文は次のとおりです。

```
For counter = start to end [Step step]
    statement
Next
```

項目	詳細
counter	反復の回数を表すカウンタとして使用する変数
start	カウンタの開始値
end	カウンタの終了値
step	各ループの終わりに増分する値。 **標準設定値 = 1**。 **省略可能**
statement	ループ中に実行する 1 つ以上のステートメント

次の例では，QuickTest で **For** ステートメントを使って乗客数の階乗値を計算しています。

```
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numPassengers").GetROProperty("value")
total = 1
For i=1 To passengers
    total = total * i
Next
MsgBox "!" & passengers & "=" & total
```

For...Each ステートメント

For...Each ループは，配列またはオブジェクト・コレクションの各要素に対して 1 つまたは複数のステートメントを実行するように QuickTest に指示します。構文は次のとおりです。

```
For Each item In array
    statement
Next
```

項目	詳細
item	配列の要素を表す変数
array	配列の名前
statement	ループ中に実行する 1 つ以上のステートメント

次の例では，**For...Each** ループを使用して配列の各要素の値を表示しています。

```
MyArray = Array("one","two","three","four","five")
For Each element In MyArray
    msgbox element
Next
```

Do...Loop ステートメント

Do...Loop ステートメントは，条件が真である間，または条件が真になるまで，1 つまたは複数のステートメントを実行するように QuickTest に指示します。構文は次のとおりです。

```
Do [{while} {until} condition]
    statement
Loop
```

項目	詳細
condition	満たされるべき条件
statement	ループ中に実行する 1 つ以上のステートメント

次の例では，QuickTest で **Do...Loop** ステートメントを使って乗客数の階乗値を計算しています。

```
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numPassengers").GetROProperty("value")
total = 1
i = 1
Do while i <= passengers
     total = total * i
     i = i + 1
Loop
MsgBox "!" & passengers & "=" & total
```

While ステートメント

While...Wend ステートメントは，条件が真である間，1 つまたは複数のステートメントを実行するように QuickTest に指示します。構文は次のとおりです。

```
While condition
    statement
Wend
```

項目	詳細
condition	満たされるべき条件
statement	ループ中に実行する 1 つ以上のステートメント

次の例では，QuickTest で **While** ステートメントを使って，乗客数が 10 人未満である間，ループを実行します。QuickTest によって，ループが 1 回実行されるたびに，乗客数が 1 ずつ増えます。

```
passengers = Browser("Mercury Tours").Page("Find Flights").
    WebEdit("numpassengers").GetROProperty("value")
While passengers < 10
    passengers = passengers + 1
Wend

msgbox(" この団体の人数は "& passengers & " 人です。")
```

If...Then...Else ステートメント

If...Then...Else ステートメントは，特定の条件に基づいて 1 つまたは複数のステートメントを実行するように QuickTest に指示します。条件が満たされない場合は，次の **Elseif** 条件または **Else** ステートメントが試されることになります。構文は次のとおりです。

```
If condition Then
    statement
ElseIf condition2 Then
    statement
Else
    statement
End If
```

項目	詳細
condition	満たされるべき条件
statement	実行されるステートメント

次の例では，乗客数が 4 名未満の場合に，QuickTest によってブラウザが閉じられます。

```
passengers = Browser("Mercury Tours").Page("Find Flights").
WebEdit("numpassengers").GetROProperty("value")
If (passengers < 4) Then
    Browser("Mercury Tours").Close
Else
    Browser("Mercury Tours").Page("Find Flights").Image("continue").Click 69,5
End If
```

次の例では，**If**，**ElseIf**，および **Else** ステートメントを使用して値が 1，2，またはその他の値に等しいかどうかを調べています。

```
value = 2
If value = 1 Then
  msgbox "one"
Elseif value = 2 Then
  msgbox "two"
Else
  msgbox "three"
End If
```

With ステートメント

With ステートメントで，同じ親階層を持つ連続するステートメントをグループ化することによって，スクリプトが短くなり，読み書きと編集がしやすくなります。

注 : With ステートメントをスクリプトに適用しても実行セッション自体には影響しません。エキスパート・ビューにどのように表示されるかにのみ影響します。

With ステートメントの構文は，次のとおりです。

```
With object
    statements
End With
```

項目	詳細
object	オブジェクトを返すオブジェクトまたは関数
statements	オブジェクトに対して実行する 1 つ以上のステートメント

たとえば，次のようなスクリプトがあったとします。

```
Window("Flight Reservation").WinComboBox("Fly From:").Select "London"
Window("Flight Reservation").WinComboBox("Fly To:").Select "Los Angeles"
Window("Flight Reservation").WinButton("FLIGHT").Click
Window("Flight Reservation").Dialog("Flights Table").WinList("From").
    Select "19097   LON "
Window("Flight Reservation").Dialog("Flights Table").WinButton("OK").Click
```

これは，次のスクリプトで置き換えることができます。

```
With Window("Flight Reservation")
    .WinComboBox("Fly From:").Select "London"
    .WinComboBox("Fly To:").Select "Los Angeles"
    .WinButton("FLIGHT").Click
    With .Dialog("Flights Table")
        .WinList("From").Select "19097   LON "
        .WinButton("OK").Click
    End With 'Dialog("Flights Table")
End With 'Window("Flight Reservation")
```

注：エキスパート・ビューで **With** ステートメントを入力しても，キーワード・ビューにはまったく影響しません。

注：With ステートメントは手作業で入力することもできますが，**With** ステートメントを自動生成させたり，既存のテストに基づいて **With** ステートメントを生成させたりもできます。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 559 ページ「テストに対する「With」ステートメントの生成」を参照してください。

テスト・オブジェクトのプロパティ値の取得と設定

テスト・オブジェクトのプロパティは，各オブジェクトについて QuickTest によって定義されている一連のプロパティです。テスト・オブジェクトのプロパティ値の設定と取得ができます。また，テスト・オブジェクト・プロパティの値を実行環境オブジェクトから取得することもできます。

テストを実行すると，QuickTest では，テスト・オブジェクト・リポジトリに格納されているテスト・オブジェクトの一時的なインスタンスが生成されます。テストまたは関数ライブラリ内の **GetTOProperty**，**GetTOProperties**，および **SetTOProperty** メソッドを使って，テスト・オブジェクトのテスト・オブジェクト・プロパティ値の設定と取得ができます。

GetTOProperty および **GetTOProperties** メソッドを使って，QuickTest がオブジェクトの識別に使う，特定のプロパティ値またはすべてのプロパティと値を取得できます。

SetTOProperty メソッドを使って，QuickTest がオブジェクトを識別するために使うプロパティ値を変更できます。

注：QuickTest は実行セッション中にテスト・オブジェクトの一時的なインスタンスを参照するため，**SetTOProperty** メソッドを使用して行ったすべての変更は実行セッション中にのみ有効で，テスト・オブジェクト・リポジトリに格納されている値には影響を与えません。

たとえば，次に示すステートメントは，［**Submit**］ボタンの名前値を「my button」に設定し，次に値「my button」を取得して **ButtonName** 変数に代入しています。

```
Browser("QA Home Page").Page("QA Home Page").
    WebButton("Submit").SetTOProperty "Name", "my button"

ButtonName=Browser("QA Home Page").Page("QA Home Page").
    WebButton("Submit").GetTOProperty("Name")
```

テスト・オブジェクト・プロパティの現在の値をアプリケーションの実行環境オブジェクトから取得するには，**GetROProperty** メソッドを使います。

たとえば，次のようにして，実行セッション時にリンクのターゲット値を取得できます。

```
link_href = Browser("Mercury Technologies").Page("Mercury Technologies").
    Link("Jobs").GetROProperty("href")
```

ヒント：Web サイトやアプリケーション内にあるオブジェクトのテスト・オブジェクト・プロパティが不明の場合は，オブジェクト・スパイを使うことでそれらを表示できます。オブジェクト・スパイの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 3 章「テスト・オブジェクト・モデルについて」を参照してください。

各オブジェクトでサポートされるテスト・オブジェクト・プロパティのリストと説明，および **GetROProperty**，**GetTOProperty**，**GetTOProperties**，および **SetTOProperty** メソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

実行環境オブジェクトのプロパティおよびメソッドへのアクセス

特定のテスト・オブジェクトで利用可能なテスト・オブジェクト・メソッドまたはプロパティによって必要な機能が提供されない場合，**Object** プロパティを使用することで，アプリケーションの任意の実行環境オブジェクトのネイティブ・メソッドおよびプロパティにアクセスできます。

QuickTest ステートメント完了機能をオブジェクトのプロパティに対して使用すれば，オブジェクトに対して利用可能なネイティブ・メソッドとプロパティのリストを表示できます。ステートメント補完機能の詳細については，120 ページ「エキスパート・ビューまたは関数ライブラリでのステートメントの生成」を参照してください。

ヒント：オブジェクトが Web オブジェクトである場合は，プログラム的記述の中で「attribute/ ＜プロパティ＞」の形式を使用してネイティブのプロパティにもアクセスできます。詳細については，166 ページ「Web オブジェクトのユーザ定義プロパティへのアクセス」を参照してください。

実行環境オブジェクト・プロパティの取得

Object プロパティを使用することで，任意の実行環境オブジェクトのネイティブ・プロパティにアクセスできます。たとえば，ActiveX カレンダーの内部 **Day** プロパティについて現在の値を取得するには，次のようにします。

```
Dim MyDay
Set MyDay=
Browser("index").Page("Untitled").ActiveX("MSCAL.Calendar.7").Object.Day
```

Object プロパティの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

実行環境オブジェクト・メソッドのアクティブ化

Object プロパティを使用することで，任意の実行環境オブジェクトの内部メソッドをアクティブ化できます。たとえば次のようにすることで，エディット・ボックスのネイティブ **focus** メソッドを呼び出せます。

```
Dim MyWebEdit
Set MyWebEdit=Browser("Mercury Tours").Page("Mercury Tours").
    WebEdit("username").Object
MyWebEdit.focus
```

Object プロパティの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

Web オブジェクトのユーザ定義プロパティへのアクセス

attribute/ ＜プロパティ名＞表記を使って，Web オブジェクトのネイティブ・プロパティにアクセスし，プログラム的記述を使ってこれらのプロパティに基づいてオブジェクトを識別することができます。

たとえば，ページ内の 2 か所に同じ企業ロゴ画像を持つ Web ページがあるとします。

```
<IMG src="logo.gif" LogoID="122">
<IMG src="logo.gif" LogoID="123">
```

ユーザ定義プロパティの LogoID を次のように記述に含めることで，クリックする画像をプログラム的記述を使って識別できます。

```
Browser("Mercury Tours").Page("Find Flights").Image("src:=logo.gif",
    "attribute/LogoID:=123").Click 68, 12
```

プログラム的記述の詳細については，145 ページ「プログラム的記述の使用」を参照してください。

DOS コマンドの実行

QuickTest テストまたは関数ライブラリの中で，VBScript Windows Scripting Host Shell オブジェクト（WSCript.shell）を使って，標準の DOS コマンドを実行できます。たとえば，次のステートメントを使うことにより，DOS コマンド・ウィンドウを開いてパスを C:¥ に変更し，**DIR** コマンドを実行できます。

```
Dim oShell
Set oShell = CreateObject ("WSCript.shell")
oShell.run "cmd /K CD C:¥ & Dir"
Set oShell = Nothing
```

詳細については，『Microsoft VBScript 言語リファレンス』（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］＞［**VBScript**］を選択）を参照してください。

Windows API を使用したテストおよび関数ライブラリの拡張

Windows API を使用してテスト機能を拡張し，テストおよび関数ライブラリの使い勝手と柔軟性を高めることができます。Windows オペレーティング・システムは，Windows での操作を制御，管理するために使用できる多数の関数を備えています。これらの関数を使用することで追加機能を利用できます。

Windows API は，次の URL から参照できる Microsoft MSDN Web サイトに文書資料があります。
http://msdn.microsoft.com/library/en-us/winprog/winprog/windows_api_start_page.asp?frame=true

個々の API 関数のリファレンスについては，次を参照してください。
http://msdn.microsoft.com/library/en-us/winprog/winprog/windows_api_reference.asp?frame=true

Windows API 関数を使用するには，次の手順を実行します。

1 MSDN において，テストまたは関数ライブラリの中で使用する関数を探します。

2 資料を読んで必要なパラメータと戻り値を把握します。

3 API 関数の場所を書き留めます。API 関数は Windows の DLL に含まれています。要求する関数が含まれている DLL の名前は通常，関数の説明の「Import Library」のセクションに記載してある名前と同じです。たとえば，説明文書の中で **User32.lib** と記載してあれば，関数は **User32.dll** という名前の DLL に含まれています。この DLL は通常，System32 ライブラリに含まれています。

4 QuickTest **Extern** オブジェクトを使用して外部関数を宣言します。詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

次の例では，**user32.dll** にある **GetForegroundWindow** という関数への呼び出しを宣言しています。

```
extern.declare micHwnd, "GetForegroundWindow", "user32.dll",
"GetForegroundWindow"
```

5 hwnd = extern.GetForegroundWindows() のように，宣言した関数を呼び出し，必要な引数を渡します。

この例では，前面にあるウィンドウのハンドラが取得されます。このようにすることで，前面のウィンドウがオブジェクト・リポジトリにない場合や，あらかじめ知ることができない場合（たとえば，動的なタイトルを持つウィンドウなど）にテストまたは関数ライブラリを拡張できます。このハンドルを，次のように，ウィンドウのプログラム的記述の一部として使用することも可能です。

```
Window("HWND:="&hWnd).Close
```

状況によっては，あらかじめ定義されている定数値を関数の引数として使用しなければならない場合があります。そうした定数は，テストまたは関数内で定義されていないため，呼び出し先の関数に渡すためにはそれらの値を調べる必要があります。こうした定数の値は通常，使用する関数に対応するヘッダー・ファイルに宣言されています。ヘッダー・ファイルに関する記述も，各関数の説明文書の「Header」セクションに記載されています。使用しているコンピュータに Microsoft Visual Studio がインストールされていれば，ヘッダー・ファイルは通常，X:¥Program Files¥Microsoft Visual Studio¥VC98¥Include の下に格納されています。

たとえば，**GetWindow API** 関数では，指定されたウィンドウとハンドルを取得するウィンドウとの関係を表す数値を受け取ることが想定されています。MSDN の説明文書には次の定数が記載されています。GW_CHILD，GW_ENABLEDPOPUP，GW_HWNDFIRST，GW_HWNDLAST，GW_HWNDNEXT，GW_HWNDPREV および GW_HWNDPREV。**GetWindow** の説明文書に記載されている **WINUSER.H** ファイルを開くと，次のフラグ値が設定されているのが分かります。

```
/*
 * GetWindow() Constants
 */
#define GW_HWNDFIRST0
#define GW_HWNDLAST 1
#define GW_HWNDNEXT2
#define GW_HWNDPREV 3
#define GW_OWNER 4
#define GW_CHILD 5
#define GW_ENABLEDPOPUP 6
#define GW_MAX 6
```

例

次の例では,「メモ帳」アプリケーションの特定のメニュー項目を取得しています。

```
' 定数値
const MF_BYPOSITION = 1024
' API 関数の宣言
Extern.Declare micHwnd,"GetMenu","user32.dll","GetMenu",micHwnd
Extern.Declare
micInteger,"GetMenuItemCount","user32.dll","GetMenuItemCount",micHwnd
Extern.Declare
micHwnd,"GetSubMenu","user32.dll","GetSubMenu",micHwnd,micInteger
Extern.Declare
micInteger,"GetMenuString","user32.dll","GetMenuString",micHwnd,micInteger,
    micString+micByRef,micInteger,micInteger
' Notepad.exe
hwin = Window("Notepad").GetROProperty ("hwnd")' ウィンドウのハンドルを
取得
MsgBox hwin
men_hwnd = Extern.GetMenu(hwin)' ウィンドウのメイン・メニューのハンドル
を取得
MsgBox men_hwnd
' API 関数を使用する
item_cnt = Extern.GetMenuItemCount(men_hwnd)
MsgBox item_cnt
hSubm = Extern.GetSubMenu(men_hwnd,0)
MsgBox hSubm
rc = Extern.GetMenuString(hSubm,0,value,64 ,MF_BYPOSITION)
MsgBox value
```

実行セッション中に報告するステップの選択

Report.Filter メソッドを使って，テスト結果にどのステップあるいはどのステップのタイプを含めるかを決めることができます。ステートメントの後のステップの報告の有効化と完全な無効化を行うことができます。あるいは，以降の失敗したステップ，もしくは失敗および警告のステップだけをレポートに含めるように指定することができます。また，**Report.Filter** メソッドを使って，現在のレポート・モードを取得することもできます。

次のレポート・モードが使用できます。

モード	詳細
0 または **rfEnableAll**	すべてのイベントがテスト結果に表示されます。**標準設定値**。
1 または **rfEnableErrorsAndWarnings**	ステータスが警告または失敗のイベントだけがテスト結果に表示されます。
2 または **rfEnableErrorsOnly**	ステータスが失敗のイベントだけがテスト結果に表示されます。
3 または **rfDisableAll**	テスト結果にはイベントは表示されません。

以降のステップの報告を行わないようにするには，次のステートメントを入力します。

```
Reporter.Filter = rfDisableAll
```

以降のステップの報告を再び行うようにするには，次のように入力します。

```
Reporter.Filter = rfEnableAll
```

以降の失敗したステップだけをテスト結果に含めるには，次のように入力します。

```
Reporter.Filter = rfEnableErrorsOnly
```

以降の失敗または警告のステップだけをテスト結果に含めるには，次のように入力します。

```
Reporter.Filter = rfEnableErrorsAndWarnings
```

現在のレポート・モードを取得するには，次のように入力します。

```
MyVar=Reporter.Filter
```

詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

第 6 章

ユーザ定義関数および関数ライブラリを使用した作業

QuickTest テスト・オブジェクト・モデルでサポートされているテスト・オブジェクト，メソッド，および組み込み関数に加え，VBScript 関数，サブルーチン，モジュールなどが含まれる独自の関数ライブラリを定義して，その関数をテストから呼び出せます。

本章では，次の内容について説明します。

- ユーザ定義関数および関数ライブラリの使い方について
- 関数ライブラリの管理
- 関連付けられている関数ライブラリを使用した作業
- 関数定義ジェネレータの使用方法
- ユーザ定義関数のテスト・オブジェクト・メソッドとしての登録
- ユーザ定義関数の使い方のヒント
- テストからの外部定義された関数の実行

ユーザ定義関数および関数ライブラリの使い方について

1 つまたは複数のテストで何度も使う必要のあるコード・セグメントがあるときには，ユーザ定義関数を作成するとよい場合があります。ユーザ定義関数とは，何らかの処理（またはプログラミングが必要な一連のステップ）を 1 つのキーワード（または操作）にカプセル化したものです。ユーザ定義関数を使用することによって，テストの簡潔になり，設計，理解，保守が容易になります。QuickTest エンジニアが関連するキーワード（または操作）をアクションに挿入することで，そのアクションからユーザ定義関数を呼び出すことができます。

ユーザ定義関数を，QuickTest テスト・オブジェクトのメソッドとして登録できます。登録したメソッドは，実行セッションの間だけ既存のテスト・オブジェクト・メソッドの機能をオーバーライドしたり，テスト・オブジェクト・クラスの新しいメソッドとして登録したりできます。ユーザ定義関数の登録の詳細については，191 ページ「関数定義ジェネレータの使用方法」および 205 ページ「ユーザ定義関数のテスト・オブジェクト・メソッドとしての登録」を参照してください。

注：ユーザ定義関数を作成するときは，組み込みの関数と同じ名前（たとえば，**GetLastError**，**MsgBox**，**Print** など）を指定しないようにします。組み込み関数は，ユーザ定義関数に優先します。したがって，組み込み関数と同じ名前のユーザ定義関数を呼び出しても，組み込み関数が代わりに呼び出されます。組み込み関数のリストについては，ステップ・ジェネレータ（[**挿入**] > [**ステップ ジェネレータ**]）の「**組み込み関数**」リストを参照してください。

QuickTest では，ユーザ定義関数を定義し，関数ライブラリ（標準設定では **.qfl** ファイルとして保存）または直接テスト内のアクションに格納できます。関数ライブラリとは，VBscript 関数，サブルーチン，モジュールなどが含まれる Visual Basic スクリプトのことです。また，QuickTest では，既存の関数ライブラリ（**.vbs** ファイルまたは **.txt** ファイルなど）を変更，デバッグすることもできます。VBScript の使用の詳細については，142 ページ「VBScript 構文エラーの処理方法」および 136 ページ「VBScript の基本的な構文の理解」を参照してください。

関数を関数ライブラリに格納し，その関数ライブラリをテストに関連付ければ，その関数ライブラリのパブリック関数をテストで呼び出すことができます。詳細については，187 ページ「関連付けられている関数ライブラリを使用した作業」を参照してください。関連付けられている関数ライブラリに格納されている関数は，ステップ・ジェネレータ，およびキーワード・ビューの［**操作**］カラムからアクセスできます。また，［エキスパート ビュー］では手動で入力できます。

関数をテスト・アクションに格納した場合，その関数はそのアクション内からのみ呼び出すことができます。ほかのアクションまたはテストから呼び出すことはできません。これは，特定のアクションの外側から関数を使用できないようにする場合に役に立ちます。

また，プライベート関数を定義して関数ライブラリに格納することができます。プライベート関数は，同じ関数ライブラリ内の他の関数からのみ呼び出せる関数です。これは，パブリック関数の中でコード・セグメントを再利用する場合に便利です。

関数は，手作業で定義することも，関数定義ジェネレータを使って定義することもできます。関数定義ジェネレータは，関数の基本的な定義を自動的に作成します。関数を手作業で定義する場合でも，関数定義ジェネレータを使用すれば，ヘッダ情報の追加，テスト・オブジェクトへの関数の登録，テスト・オブジェクトの標準メソッドとしての関数の設定を行うために必要な構文を表示できます。詳細については，191 ページ「関数定義ジェネレータの使用方法」を参照してください。

関数ライブラリの管理

QuickTest で関数ライブラリを作成し，その関数をテストのアクションから呼び出すことができます。関数ライブラリとは，VBscript 関数，サブルーチン，モジュールなどが含まれる独立した QuickTest 文書のことです。各関数ライブラリは別々のウィンドウで開くため，同時に 1 つまたは複数の関数ライブラリを開いて作業できます。関数ライブラリの編集が終了したら，関数ライブラリは閉じて，QuickTest セッションは開いたままにしておけます。また，開いているすべての関数ライブラリを同時に閉じることもできます。

ユーザ定義関数を関数ライブラリで実装し，その関数ライブラリをテストに関連付けることによって，他のユーザでも，テストに直接コードを追加せずに，複雑な操作（テスト・ステップへの if/then ステートメントとループの追加や，

ユーティリティ・オブジェクトを使った作業など）を実行する関数を選択できるようになります。さらに，再利用可能な関数の実装および使用により，時間とリソースを節約できます。

QuickTest には，あらゆる関数ライブラリを（外部エディタで作成された関数ライブラリでも）編集およびデバッグできるツールがあります。たとえば，QuickTest では関数の構文をチェックできます。関数ライブラリ・ウィンドウには，［エキスパート ビュー］が備えているものと同様の編集機能があります。［エキスパート ビュー］で使用できるオプションの詳細については，第 5 章「エキスパート・ビューおよび関数ライブラリ・ウィンドウを使用した作業」を参照してください。

注： QuickTest では，テストを開くと，Quality Center プロジェクトに保存されている外部リソースのローカル・コピーが作成されます。したがって，別のユーザが関数ライブラリなど Quality Center プロジェクトに保存されている外部リソースを変更した場合，または，外部エディタ（QuickTest ではないもの）を使用してリソースを変更した場合，その変更はテストを閉じて再度開くまでテストコンポーネントに実装されません。これに対して，関数ライブラリなどファイル・システムに保存されている外部リソースに適用した変更は，ただちに実装されます。これらのファイルは直接アクセスされ，テストを開いたときにローカル・コピーとして保存されないためです。

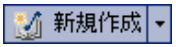
関数ライブラリの作成

新しい関数ライブラリは，いつでも作成できます。

QuickTest で新しい関数ライブラリを作成するには，次の手順を実行します。

次の手順のいずれかを実行します。

➤ ［**ファイル**］＞［**新規作成**］＞［**関数ライブラリ**］を選択します。



➤ ［**新規作成**］ボタンの下向き矢印をクリックし，［**関数ライブラリ**］を選択します。

新しい関数ライブラリが開きます。

これで，関数ライブラリに内容を追加し，保存できます。関数ライブラリに内容を追加すると，［エキスパート ビュー］の内容に適用されるのと同じ書式設定が適用されます。書式設定は，必要に応じて変更できます。詳細について

は，317 ページ「[エキスパート ビュー］および関数ライブラリ・ウィンドウのカスタマイズ」を参照してください。

関数ライブラリの保存

QuickTest で関数ライブラリを作成または編集したら，Quality Center プロジェクトまたはファイル・システムに関数ライブラリを保存できます。

ヒント：

- 関数ライブラリに変更を加えると，関数ライブラリが保存されるまで，タイトル・バーにアスタリスク（*）が表示されます。
- 開いている文書をすべて保存するには，［**ファイル**］＞［**すべて保存**］を選択します。まだ保存されていない新規ファイルについては，保存先を指定するように求められます。
- 複数の文書を保存するには，［**ウィンドウ**］＞［**ウィンドウ**］を選択します。［ウィンドウ］ダイアログ・ボックスで，保存する文書を選択し，［**保存**］ボタンをクリックします。まだ保存されていない新規ファイルについては，保存先を指定するように求められます。
- アクティブな関数ライブラリを別の名前または別のパスで保存するには，［**ファイル**］＞［**名前を付けて保存**］を選択します。

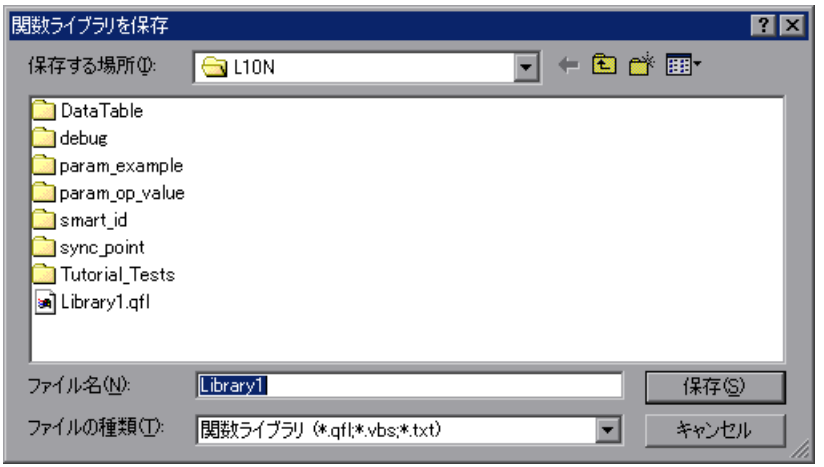
関数ライブラリを保存するには，次の手順を実行します。

1 保存する対象となる関数ライブラリがアクティブな文書であることを確認します（フォーカスを対象に移動するには，対象関数ライブラリのタブをクリックします）。

2 次の手順のいずれかを実行します。

- ［**保存**］ボタンをクリックします。
- ［**ファイル**］＞［**保存**］を選択します。
- 関数ライブラリ文書のタブを右クリックし，［**保存**］を選択します。

この関数ライブラリを以前に保存したことがあれば，変更された関数ライブラリが保存されます。この関数ライブラリを初めて保存する場合は，［関数ライブラリを保存］ダイアログ・ボックスが開きます。



3 Quality Center プロジェクトまたはファイル・システムに関数ライブラリを保存します（関数ライブラリがビジネス・プロセス・テストで使用される場合は，その関数ライブラリを Quality Center プロジェクトに保存する必要があります）。

注：Quality Center に接続している場合，表示されるダイアログ・ボックスは，ファイル・システムの場合の標準ダイアログ・ボックスと異なります。［**保存**］ダイアログ・ボックスで［**ファイル システム**］ボタンおよび［**Quality Center**］ボタンをクリックすることで，ダイアログ・ボックスの 2 つのバージョンを切り替えることができます。

- 関数ライブラリを Quality Center プロジェクトに保存するには，［テスト計画ツリー］ボックスで関数ライブラリの保存先フォルダを選択します。［**添付名**］ボックスに関数ライブラリ名を入力し，［**OK**］をクリックします。
- 関数ライブラリをファイル・システムに保存するには，［関数ライブラリを保存］ダイアログ・ボックスの中で，［**ファイル名**］ボックスに関数ライブラリ名を入力し，［**保存**］をクリックします。

関数ライブラリは，拡張子 **.qfl** を付けて保存されます（ただし，**.vbs** や **.txt** などの別の拡張子を指定した場合，または，拡張子を完全に削除した場合を除く）。

関数ライブラリを開く

QuickTest では，すでに別の文書が開いていても，ファイル・システムまたは Quality Center プロジェクトに保存されている関数ライブラリを開くことができます。関数ライブラリは，当該ファイルに対する読み取り許可または読み書きの許可がある場合にのみ開けます。

関数ライブラリを編集モードで開くか読み取り専用モードで開くかを選択できます。

- **編集モード**：関数ライブラリを表示，変更できます。あるコンピュータで関数ライブラリが開いている間は，ほかのユーザは，そのファイルを読み取り専用モードで表示できますが，変更はできません。
- **読み取り専用モード**：関数ライブラリを表示することはできますが，変更はできません。標準設定では，現在ほかのコンピュータで開いている関数ライブラリを開くと，読み取り専用モードになります。関数ライブラリを表示する一方で，ほかのユーザが変更できるようにもしたい場合は，関数ライブラリを読み取り専用モードで開けます。

ヒント：自分の文書の関数から別の関数ライブラリの関数定義へ直接移動することもできます。詳細については，182 ページ「関数ライブラリの特定関数への移動」を参照してください。

既存の関数ライブラリを開くには，次の手順を実行します。

次の手順のいずれかを実行します。

➤［**ファイル**］＞［**開く**］＞［**関数ライブラリ**］を選択します。

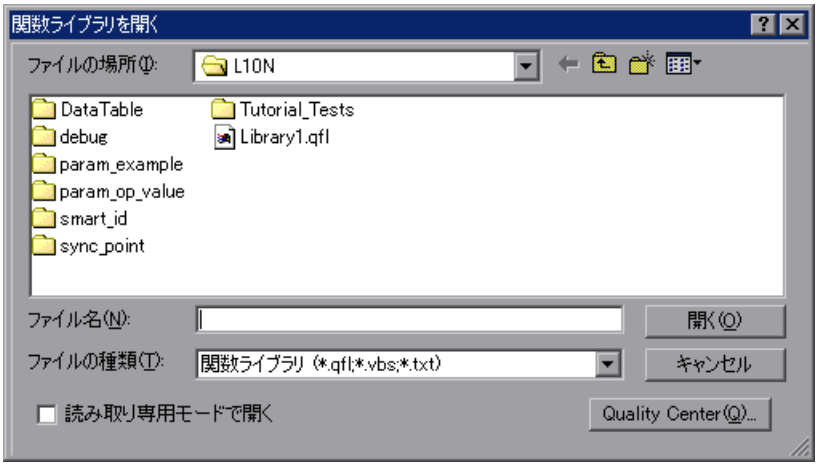
➤［**開く**］ボタンの下向き矢印をクリックし，［**関数ライブラリ**］を選択します。

ヒント：

最近作成した，または開いた関数ライブラリは，［**ファイル**］メニューに表示される最近使用したファイルから選択できます。

開いているテストに関数ライブラリが関連付けられている場合，［**リソース**］＞［**関連付けのあるライブラリ**］からその関数ライブラリを選択できます（Quality Center プロジェクトに格納されている関数ライブラリを選択すると，関連付けられている関数ライブラリを開くために QuickTest が当該プロジェクトに接続されている必要があります）。

［関数ライブラリを開く］ダイアログ・ボックスが開きます。



注：Quality Center に接続している場合，表示されるダイアログ・ボックスは，ファイル・システムの場合の標準ダイアログ・ボックスと異なります。［**開く**］ダイアログ・ボックスで［**ファイル システム**］ボタンおよび［**Quality Center**］ボタンをクリックすることで，ダイアログ・ボックスの 2 つのバージョンを切り替えることができます。

ヒント：関数ライブラリを読み取り専用モードで開くには，［**読み取り専用モードで開く**］チェック・ボックスを選択します。

関数ライブラリを探して選択し，［**開く**］をクリックします。指定した関数ライブラリが新しいウィンドウで開きます。これで，関数ライブラリの内容を表示，変更できるようになります。詳細については，182 ページ「関数ライブラリの編集」および 184 ページ「関数ライブラリのデバッグ」を参照してください。

開いている QuickTest 文書間でのフォーカスの移動

テストが開いている間，複数の関数ライブラリを開き，開いているすべての文書間でフォーカスを移動することができます。

開いている QuickTest 文書間でフォーカスを移動するには，次の手順を実行します。

次の手順のいずれかを実行します。

➤ 文書表示枠で，必要な文書のタブをクリックします。

ヒント：スペースが足りず，すべてのタブが表示されない場合は，文書表示枠の左右のスクロール矢印を使用して，必要な文書のタブを表示します。

➤ キーボードの CTRL キーを押しながら TAB キーを押し，開いている文書間を切り替えます。

➤ ［ウィンドウ］メニューから必要な文書を選択します。

➤ [**ウィンドウ**] > [**ウィンドウ**] を選択し，[ウィンドウ] ダイアログ・ボックスで必要な文書を選択して，[**切り替え**] ボタンをクリックします。

注 : [**リソース**] > [**関連付けのある関数ライブラリ**] を選択して，リストから必要な関数ライブラリを選択することもできます。この操作では，テストに関連付けられている，閉じられている関数ライブラリも開きます。

関数ライブラリの特定関数への移動

関数の呼び出しを挿入した後，ソース文書内の当該関数の定義へ直接移動できます。関数の定義は，同じ文書（テストまたは関数ライブラリ），またはテストに関連付けられている別の関数ライブラリに置くことができます。関数の定義が含まれる文書がすでに開いている場合は，そのウィンドウがアクティブになります（そこにフォーカスが移動します）。文書が閉じている場合は，その文書が開きます。

関数の定義に移動するには，次の手順を実行します。

1 [エキスパート ビュー] または関数ライブラリで，該当する関数が含まれているステップをクリックします。

2 次の手順のいずれかを実行します。

➤ [**編集**] > [**詳細設定**] > [**関数定義に移動**] を選択します。

➤ ステップを右クリックして，ショートカット・メニューの [**関数定義に移動**] を選択します。

該当する文書がアクティブになり（関数の定義が別の関数ライブラリにある場合），関数の定義の先頭にカーソルが置かれます。

関数ライブラリの編集

[エキスパート ビュー] の QuickTest 編集機能を使用して，関数ライブラリをいつでも編集できます。

文書間で関数（またはその一部分）のドラッグ・アンド・ドロップが可能です（それには，[**最小化**] ボタン（QuickTest ウィンドウの [**最小化 / 最大化**] ボタンの下にあります）をクリックすることで，タブ付きの文書を別々の文書表示枠に分ける必要があります）。

手動で，またはステップ・ジェネレータを使用して，関数ライブラリにステップを追加できます。ステップ・ジェネレータを使用すれば，予約オブジェクト(ユーティリティ・オブジェクトなど，機能拡張のために提供されるオブジェクト)，VBScript 関数（**MsgBox** など)，ユーティリティ・ステートメント(**Wait** など)，同じ関数ライブラリに定義されているユーザ定義関数を含むステップを追加できます。IntelliSense は，アクションに定義されているすべの関数，または関連付けられている関数ライブラリに定義されているパブリック関数で使用できます。

注：関数ライブラリでは，IntelliSense を使用してテスト・オブジェクトの名前またはコレクションを表示できません。これは，関数ライブラリがオブジェクト・リポジトリに接続されていないためです。

構文をチェックするように QuickTest に指定するには，［**構文チェック**］ボタンをクリックするか，［**ツール**］＞［**構文チェック**］を選択します。

ヒント：

VBScript の使用の詳細については，136 ページ「VBScript の基本的な構文の理解」を参照してください。

テストに関連付けられているすべての関数ライブラリの構文をチェックするには，［テストの設定］ダイアログ・ボックス（[**ファイル**］＞［**設定**]）の［リソース］タブにある［**構文チェック**］ボタンをクリックします。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 755 ページ「テストのためのリソース設定の定義」を参照してください。

読み取り専用関数ライブラリの編集

関数ライブラリを読み取り専用モードで開き，その後でその関数ライブラリに変更を加えることにした場合，その関数ライブラリを編集可能ファイルに変換できます。ただし，ほかのユーザがその関数ライブラリをロックしていない場合に限ります。関数ライブラリを開くときに使用できるオプションの詳細については，179 ページ「関数ライブラリを開く」を参照してください。

注：デバッグ・セッション中は，すべての文書（テストおよび関数ライブラリなど）が読み取り専用になります。デバッグ・セッション中に文書を編集するには，まず，デバッグ・セッションを停止する必要があります。

読み取り専用の関数ライブラリを編集するには，次の手順を実行します。

［**ファイル**］＞［**編集可能にする**］を選択するか，［**編集可能にする**］ボタンをクリックします。これで，関数ライブラリを編集できます。

関数ライブラリのデバッグ

関数ライブラリをデバッグする前に，関数ライブラリをテストに関連付けて，ライブラリ内の関数への呼び出しを少なくとも 1 つ挿入しておく必要があります。たとえば，［デバッグ ビューア］を使用して，関数ライブラリのオブジェクトまたは変数の現在の値を表示，設定，変更できます。関数（ユーザ定義関数を含む）のステップ・イントゥ，ブレークポイントの設定，ブレークポイントでの停止，式の表示などが可能です。デバッグは特定のステップから開始したり，特定のステップで一時停止するように指定したりできます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 577 ページ「テストと関数ライブラリのデバッグ」を参照してください。

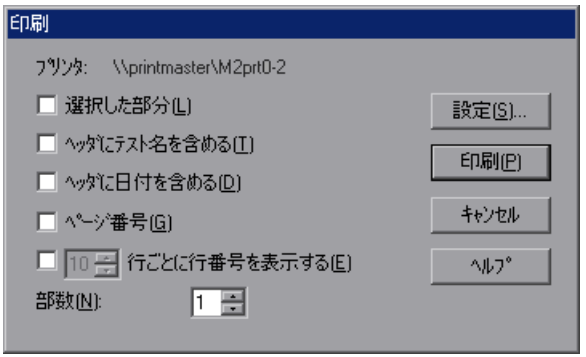
注：デバッグ・セッション中は，すべての文書が読み取り専用になり，編集できません。デバッグ・セッション中に文書を編集するには，まず，デバッグ・セッションを停止する必要があります。

関数ライブラリの印刷

関数ライブラリは，いつでも印刷できます。印刷出力には追加の情報を含めることもできます。

関数ライブラリを印刷するには，次の手順を実行します。

1 ［**印刷**］ボタンをクリックするか，［**ファイル**］＞［**印刷**］を選択します。［印刷］ダイアログ・ボックスが開きます。



2 印刷オプションを指定します。

- ［**プリンタ**］: 印刷ジョブの送信先となるプリンタが表示されます。プリンタを変更するには，［**設定**］ボタンをクリックします。
- ［**選択した部分**］: 関数ライブラリ内の現在選択されている（強調表示されている）テキストのみ印刷されます。
- ［**ヘッダに文書名を含める**］: 印刷出力の上部に関数ライブラリの名前が挿入されます。
- ［**ヘッダに日付を含める**］: 印刷出力の上部にその日の日付が挿入されます。日付書式は Windows の地域のオプションに基づきます。
- ［**ページ番号**］: 印刷出力の下部にページ番号が挿入されます（例：page 1 of 3）。
- ［**X 行ごとに行番号を表示する**］: 指定どおりに，スクリプト行の左側に行番号が表示されます。
- ［**部数**］: ドキュメントを印刷する回数を指定します。

3 別のプリンタに印刷する場合やプリンタの設定を変更する場合は，［**設定**］をクリックして［プリンタの設定］ダイアログ・ボックスを表示します。

4 ［**印刷**］をクリックすると，選択内容に従って印刷されます。

関数ライブラリを閉じる

個々の関数ライブラリを閉じたり，複数の関数ライブラリが開いている場合は，その一部または全部を同時に閉じたりすることができます。いずれかの関数ライブラリが保存されていないと，保存するよう QuickTest に求められます。

個々の関数ライブラリを閉じるには，次の手順を実行します。

次の手順のいずれかを実行します。

- 保存する関数ライブラリがアクティブな文書であることを確認し（当該関数ライブラリにフォーカスを移動するには，関数ライブラリのタブをクリックします），［**ファイル**］＞［**閉じる**］を選択します。
- 関数ライブラリ文書のタブを右クリックし，［**閉じる**］を選択します。
- 
関数ライブラリ・ウィンドウの右上角にある［**閉じる**］ボタンをクリックします。
- ［**ウィンドウ**］＞［**ウィンドウ**］を選択します。［ウィンドウ］ダイアログ・ボックスで，閉じる関数ライブラリが選択されていない場合は選択し，［**ウィンドウを閉じる**］ボタンをクリックします。

複数の関数ライブラリを閉じるには，次の手順を実行します。

［**ウィンドウ**］＞［**ウィンドウ**］を選択します。［ウィンドウ］ダイアログ・ボックスで，閉じる関数ライブラリを選択し，［**ウィンドウを閉じる**］ボタンをクリックします。

開いているすべての関数ライブラリを閉じるには，次の手順を実行します。

［**ファイル**］＞［**全関数ライブラリを閉じる**］を選択するか，［**ウィンドウ**］＞［**全関数ライブラリを閉じる**］を選択します。

関連付けられている関数ライブラリを使用した作業

QuickTest では，関数，サブルーチン，モジュールなどが含まれる関数ライブラリを作成し，そのファイルをテストに関連付けることができます。これによって，QuickTest エンジニアは，関連付けられている関数ライブラリ内のパブリック関数やサブルーチンへの呼び出しを，当該テストに挿入することができます（関数ライブラリに格納されているパブリック関数は，関連付けられている任意のテストから呼び出せるのに対し，プライベート関数は同じ関数ライブラリ内からのみ呼び出せます）。

テストがステップ内で使用されている関数にアクセスできなくなると（たとえば，関連付けられている関数ライブラリからその関数が削除された場合など），キーワード・ビュー内でステップの横に  アイコンが表示されます。そのテストを実行すると，存在しない関数を使用しているステップに達したときにエラーが発生します。

注：標準の VBScript 構文で書かれたあらゆるテキスト・ファイルを関数ライブラリとして使えます。

［テストの設定］ダイアログ・ボックス（［**ファイル**］＞［**設定**］＞［**リソース**］タブ）で，新規のすべてのテストに関連付ける標準の関数ライブラリを指定できます。テストが作成されると，標準関数ライブラリのリストがテストに統合されます。したがって，［テストの設定］ダイアログ・ボックスの標準関数ライブラリ・リストに変更を加えても，既存のテストには影響しません。

既存のテストに関連付けられている関数ライブラリのリストは［テストの設定］ダイアログ・ボックスで編集できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 755 ページ「テストのためのリソース設定の定義」を参照してください。

注：

- 関連関数ライブラリの中にある関数のほかに，任意の関数ライブラリ（または VBScript ファイル）に含まれている関数を任意のアクションから **ExecuteFile** 関数を使って直接呼び出すことができます。関連関数ライブラリに **ExecuteFile** 関数を挿入することもできます。詳細については，213 ページ「テストからの外部定義された関数の実行」を参照してください。
- **ExecuteFile** ステートメントを使用して呼び出されるファイルおよび当該ファイルに含まれている関数をデバッグすることはできません。また，**ExecuteFile** ステートメントを含んでいるテストをデバッグする場合，実行マーカが正しく表示されないことがあります。

Quality Center での関連付けられている関数ライブラリを使用した作業

関数ライブラリがファイル・システムまたは Quality Center プロジェクトに格納されているかどうかに関係なく，テストに関数ライブラリを関連付けることができます。ただし，ビジネス・プロセス・テストで関数ライブラリを使用する計画がある場合は，関数ライブラリを Quality Center プロジェクトに保存する必要があります。

Quality Center および関連関数ライブラリを使用するときには，［テストの設定］ダイアログ・ボックスの［リソース］タブで関連ファイルを指定する**前に**，関連関数ライブラリを Quality Center プロジェクトに添付ファイルとして保存する必要があります。Quality Center プロジェクトには新規または既存の関数ライブラリを追加できます。

ファイル・システムから Quality Center プロジェクトに既存の関数ライブラリを追加する場合，実際にはそのファイルのコピーがプロジェクトに追加されます。したがって，これらの関数ライブラリのどちらか一方（ファイル・システム内または Quality Center プロジェクト内）を後で変更した場合，もう一方の関数ライブラリは影響を受けません。

テストとの関数ライブラリの関連付け

現在開いているテストに，開いている関数ライブラリを関連付けることができます。

また，関連付けられている関数リストを使用して，現在開いているテストに関数ライブラリを関連付けることもできます。詳細については，190 ページ「関数ライブラリの関連付けの変更」を参照してください。

テストに関数ライブラリを関連付けるには，次の手順を実行します。

1 関数ライブラリを関連付ける対象となるテストが QuickTest で開かれていることを確認します。

2 QuickTest で関数ライブラリを作成するか開きます （次の手順に進む前に，テストに関連付ける関数ライブラリがアクティブな文書であることを確認します)。対象関数ライブラリにフォーカスを移動するには，関数ライブラリのタブをクリックします)。詳細については，175 ページ「関数ライブラリの管理」を参照してください。

3 関数ライブラリを，添付ファイルとして Quality Center プロジェクト，またはファイル・システムに保存します。詳細については，177 ページ「関数ライブラリの保存」を参照してください。

4 QuickTest の中で，[**ファイル**]＞[**ライブラリ ' ＜ Function Library ＞ ' を 'Test' に関連付ける**]を選択するか，関数ライブラリの中で右クリックして［**ライブラリ ' ＜ Function Library ＞ ' を 'Test' に関連付ける**］を選択します。
QuickTest によって，開いているテストに関数ライブラリが関連付けられます。

関数ライブラリの関連付けの変更

テストに関連付けられている関数ライブラリのリストを変更できます。リストに関数ライブラリを追加したり，リストから削除したりできます。また，関数ライブラリの優先順位を変更することもできます。

テストへの関数ライブラリの関連付けを変更するには，次の手順を実行します。

1 ［テストの設定］ダイアログ・ボックスで，［**リソース**］タブをクリックします。

2 関連付けられている関数ライブラリ・リストで，［**関数ライブラリの追加**］ボタンをクリックします。QuickTest に参照ボタンが表示されます。このボタンを使用して，ファイル・システム内の関数ライブラリを参照できます。Quality Center プロジェクトに接続している場合は，ファイル・パスに [QualityCenter] が追加され，Quality Center プロジェクト内およびファイル・システム内の関数ライブラリを参照できることが示されます。

ヒント：Quality Center プロジェクトからファイルを追加する必要があり，なおかつ Quality Center に接続されていない場合は，SHIFT キーを押したまま［**追加**］ボタンをクリックします。QuickTest は [QualityCenter] を追加し，パスが手動で入力できるようになります。その場合には，[QualityCenter] の後にスペースを入れてください。例を次に示します。[QualityCenter] Subject¥Tests ただし，QuickTest が Quality Center のプロジェクト・フォルダを検索するのは，対応する Quality Center プロジェクトに接続しているときだけです。

---

3 テストに関連付ける関数ライブラリを選択し，［**開く**］または［**OK**］をクリックします（関数ライブラリをファイル・システムまたは Quality Center プロジェクトのどちらから選択するかに応じて異なります）。

ヒント：関連付けられている関数ライブラリをリストから削除するには，削除する関数ライブラリを選択して，［**削除**］ボタンをクリックします。関連付けられている関数ライブラリの優先順位付けを行うには，**上向き**矢印および**下向き**矢印を使用します。

詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 755 ページ「テストのためのリソース設定の定義」を参照してください。

関数定義ジェネレータの使用方法

QuickTest の関数定義ジェネレータを使用すれば，新規ユーザ定義関数のための定義を生成し，その定義にヘッダ情報を追加できます。その後，必要に応じて，その関数をテスト・オブジェクトに登録できます。必要な情報を入力すると，関数定義ジェネレータが基本的な関数定義を自動的に作成します。関数定義を定義した後，それを関数ライブラリに挿入してテストに関連付けることができます。また，定義はエキスパート・ビュー内のテスト・スクリプトに直接挿入することもできます。最後に，内容（コード）を追加して関数を完成させます。

注：関数をエキスパート・ビューに直接挿入した場合，テストは特定のアクションのどこからでもその関数にアクセスできるようになります。

テスト・オブジェクトに関数を登録すると，関数は，そのテスト・オブジェクトによって呼び出すことができ，そのテスト・オブジェクトの使用可能な操作のリストに表示されます。

テスト・オブジェクトに関数を登録しないと，この関数はグローバル操作となり，ステップ・ジェネレータの［**操作**］ボックスの操作一覧に，キーワード・ビューの［**操作**］カラムに，また IntelliSense の使用時に表示されます。関数を登録すると，その関数が登録されているテスト・オブジェクトが選択された場合にステップ・ジェネレータまたはキーワード・ビューに表示される標準の操作として定義できます。

最後に，ステップ・ジェネレータまたはキーワード・ビューで操作にカーソルを位置付けたときと，IntelliSense の使用時に表示されるツールチップを定義することによって，ユーザ定義関数に説明を付けることができます。また，ユーザ定義関数を含むステップが実際に何をするかについて説明した文章を追加することもできます。この文章は，ステップ・ジェネレータの［**ステップについてのコメント**］ボックスと，キーワード・ビューの［**注釈**］カラムに表示されます。

関数定義ジェネレータに情報を入力するにつれて，でき上がりつつある関数定義が［**プレビュー**］領域に表示されます。関数の定義が完了したら，アクティブな QuickTest 文書に定義を挿入します。定義を関数ライブラリに挿入した場合，関連付けられているテストは関数にアクセスできます。関数を［エキスパート

ビュー］内のテストに直接挿入した場合，関数は，特定のアクション内からのみ呼び出すことができます。最後に，関数の内容（コード）を追加します。

次の項では，関数定義ジェネレータで関数を作成するときに実行する手順の概要を説明します。

関数定義ジェネレータを使用するには，次の手順を実行します。

1 193 ページ「関数定義ジェネレータの起動」の説明に従って，関数定義ジェネレータを開きます。

2 194 ページ「関数の定義」の説明に従って，関数を定義します。

3 195 ページ「関数ジェネレータを使用した関数の登録」の説明に従って，必要に応じて関数をテスト・オブジェクトに登録します。

標準設定では，テスト・オブジェクトに登録されない関数は，ステップ・ジェネレータの［**関数**］カテゴリまたはキーワード・ビューの［**操作**］項目を選択することによって呼び出せる，あるいは IntelliSense の使用時に呼び出せるグローバル関数として自動的に定義されます。関数をテスト・オブジェクトに登録した場合は，その関数（操作）をそのテスト・オブジェクトの標準設定の操作として定義することもできます。

4 199 ページ「関数の引数の指定」の説明に従って，関数に引数を追加します。

5 ヘッダ情報を追加して関数を文書化します。詳細については，200 ページ「関数への説明の追加」を参照してください。

6 関数の仕上げの前に，202 ページ「関数のプレビュー」の説明に従って，その関数をプレビューします。

7 202 ページ「別のユーザ定義関数の生成」の説明に従って，必要に応じて別の関数定義を生成します。

8 203 ページ「ユーザ定義関数の仕上げ」の説明に従って，アクティブな文書に各関数を挿入し，それらの関数に内容を追加して，各関数を仕上げます。

注： この項で説明する手順では，前述の手順を実行しているものと想定しています。

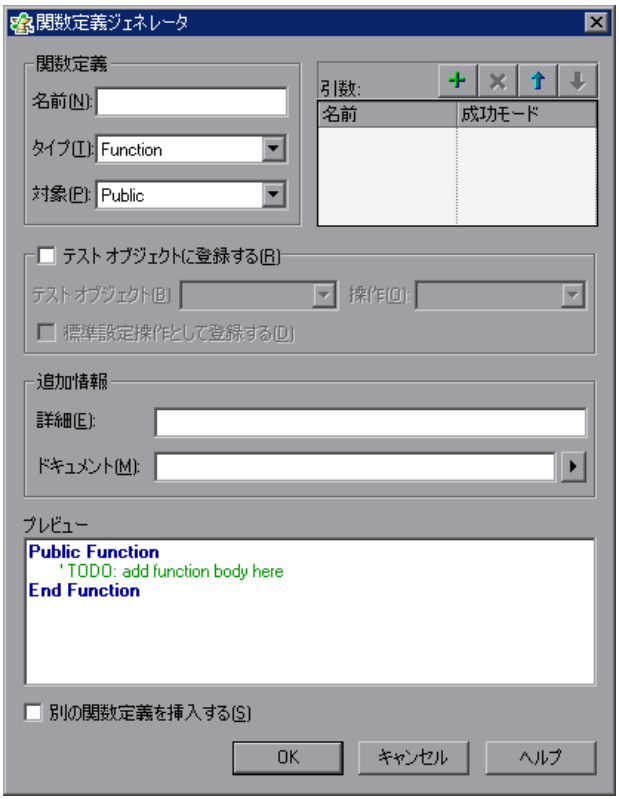
関数定義ジェネレータの起動

QuickTest から関数定義ジェネレータを開きます。

関数定義ジェネレータを開くには，次の手順を実行します。

1 関数定義を挿入する対象となる関数ライブラリまたはテストがアクティブな文書であることを確認します（対象文書にフォーカスを移動するには，文書のタブをクリックします）。これは，関数定義の完了後，関数定義ジェネレータが現在アクティブな文書に関数を挿入するためです。

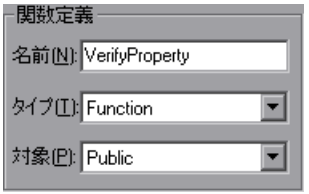
2 ［**挿入**］＞［**関数定義ジェネレータ**］を選択するか，［**関数定義ジェネレータ**］ボタンをクリックします。関数定義ジェネレータが開きます。



関数定義ジェネレータを開いた後，新しい関数の定義を開始できます。

関数の定義

関数定義ジェネレータを開いたら，関数の定義を開始できます。



たとえば，指定されたプロパティの値を確認する関数を定義する場合，関連付けられている任意のテストから呼び出せるように，その関数に VerifyProperty という名前を付けてパブリック関数として定義できます（関数ライブラリがそのアプリケーション領域に関連付けられている場合のみ）。（プライベートとして定義した関数は，同じ関数ライブラリ内にある別の場所からのみ呼び出せます。プライベート関数は，テスト・オブジェクトに登録できません）。

関数を定義するには，次の手順を実行します。

1 ［**名前**］ボックスに，新しい関数の名前を入力します。ステップ・ジェネレータまたはキーワード・ビューから簡単に選択できるように，何をする操作なのかがはっきりと分かる名前を付けてください。関数名には，英字以外の文字を含めることはできません。また，関数名は英字で始まらなければならず，スペースや次の文字を含めてはなりません。
! @ # $ % ^ & * ( ) + = [ ] \ { } | ; ' : " , / < > ?

注：ユーザ定義関数に，組み込みの関数と同じ名前（たとえば，**GetLastError**，**MsgBox**，**Print** など）を付けないようにします。組み込み関数は，ユーザ定義関数に優先します。したがって，組み込み関数と同じ名前のユーザ定義関数を呼び出しても，組み込み関数が代わりに呼び出されます。組み込み関数のリストについては，ステップ・ジェネレータ（［**挿入**］＞［**ステップ ジェネレータ**］）の「**組み込み関数**」リストを参照してください。

---

2 関数またはサブルーチンのどちらを定義するのかに応じて，［**タイプ**］リストから［**Function**］または［**Sub**］を選択します。

3 ［**対象**］リストから，関数の適用範囲として，［**Public**］（この関数ライブラリに関連付けられている任意のテストコンポーネントからこの関数を呼び出せるようにする場合）または［**Private**］（同じ関数ライブラリ内の別の場所からのみこの関数を呼び出せるようにする場合）を選択します。標準設定では，適用範囲は［**Public**］に設定されています （パブリック関数のみ，テスト・オブジェクトに登録することができます)。

注：ユーザ定義関数を手作業で作成し，範囲を［**Public**］とも［**Private**］とも定義しなかった場合，その関数は標準設定でパブリック関数として扱われます。

パブリック関数を定義した後，関数を登録できます。また，プライベート関数を定義した場合，あるいは関数を登録しない場合は，引き続き関数に引数を指定できます。詳細については，199 ページ「関数の引数の指定」を参照してください。

関数ジェネレータを使用した関数の登録

パブリック関数をテスト・オブジェクトに登録すると，その関数（操作）をテスト・オブジェクトに対して実行できるようになります。関数をテスト・オブジェクトに登録するときには，既存の操作の機能をオーバーライドすることも，そのテスト・オブジェクトに対する新しい操作として登録することもできます。

関数をテスト・オブジェクトに登録すると，ステップ・ジェネレータでそのテスト・オブジェクトを選択したときに，およびキーワード・ビューの中で［**項目**］リストからそのテスト・オブジェクトを選択したときに［**操作**］リストに，また IntelliSense に，およびステップ・ジェネレータの一般的な［**操作**］リストに，関数が操作として表示されます。関数をテスト・オブジェクトに登録すると，その関数はそのテスト・オブジェクトからしか呼び出せなくなります。

テスト・オブジェクトに関数を登録することを選択した場合，関数定義ジェネレータの右上角にある［引数］領域に，1 番目の引数として自動的に **test_object** 引数が追加されます。また，関数定義ジェネレータは，関数定義のすぐ後ろに，適切な引数値を持つ **RegisterUserFunc** ステートメントを自動的に追加します。

関数をテスト・オブジェクトに登録するときに，任意でその関数をテスト・オブジェクトの標準設定の操作として定義することもできます。定義した場合は，QuickTest エンジニアまたは各分野のエキスパートが［**項目**］リスト内の関連付けられているテスト・オブジェクトを選択したときに，標準設定で［**操作**］カラムにその関数が表示されます。また，IntelliSense でも関数を選択できるようになります。関数をテスト・オブジェクトの標準設定の関数として定義すると，**RegisterUserFunc** ステートメントの 4 番目の引数の値として **True** が指定されます。

関数を特定のテスト・オブジェクトに登録しなかった場合，その関数は自動的にグローバル関数として定義されます。グローバル関数は，ステップ・ジェネレータの［**関数**］カテゴリ，またはキーワード・ビューの［**操作**］項目を選択して呼び出します。グローバル関数のリストは，ステップ・ジェネレータで［**関数**］カテゴリを選択した場合は［**操作**］ボックスに，キーワード・ビューの［**項目**］リストから［**操作**］項目を選択した場合は［**操作**］リストに，また IntelliSense の使用時に，アルファベット順に表示されます。

実行時に，QuickTest によって，指定された関数がまずテストの中で検索され，次に［リソース］タブに表示されている順番で関数ライブラリの中で検索されます。指定したテストまたは関数ライブラリ内に関数名の一致する関数が複数見つかった場合は，テストまたは関数ライブラリ内で最後に検出された関数が使用されます。QuickTest によって 2 つの異なる関数ライブラリで同じ名前の 2 つの関数が見つかった場合，優先順位の高い方の関数ライブラリの関数が使用されます。混乱を避けるために，1 つのテストに関連付けられているリソースの中では，それぞれの関数に一意の名前を付けることを推奨します。

ヒント：この時点で関数を登録しなかった場合は，後ほど手作業で，その関数の後に次の例のように **RegisterUserFunc** ステートメントを付け加えて，登録することができます。

```
RegisterUserFunc "WebEdit", "MySet", "MySetFunc"
```

この例では，ユーザ定義関数 **MySetFunc** を使用して，**MySet** メソッド（操作）を WebEdit テスト・オブジェクトに追加しています。追加後，QuickTest エンジニアキーワード・ビューで［**項目**］リストから WebEdit テスト・オブジェクトを選択すると，［**操作**］リストに，WebEdit テスト・オブジェクトの登録されているほかの操作，およびあらかじめ用意されている操作とともに **MySet** 操作が表示されます。

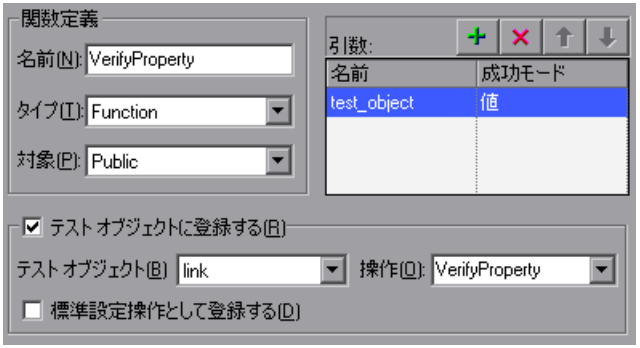
また，関数をほかのテスト・オブジェクトに登録することもできます。それには，関数コードを関数ライブラリに保存するときに，**RegisterUserFunc** ステートメントを複製し（コピーして貼り付けて），必要に応じて引数値を変更します。

この関数を標準設定の関数として定義するには，**RegisterUserFunc** ステートメントの 4 番目の引数の値を **True** に指定します。例を次に示します。
RegisterUserFunc "WebEdit", "MySet", "MySetFunc", True

注：登録した関数もグローバル関数も，テストに関連付けられているテスト・スクリプトまたは関数ライブラリに追加された後にのみ，テストから呼び出せます。

関数をテスト・オブジェクトに登録するには，次の手順を実行します。

1 ［**テスト オブジェクトに登録する**］チェック・ボックスを選択します。この領域内のオプションが使用可能になり，関数定義ジェネレータの右上角にある［**引数**］領域内の引数のリストに **test_object** という新しい引数が自動的に追加されます （**test_object** 引数は，関数を登録する対象となるテスト・オブジェクトを受け取ります)。



注 :［**テスト オブジェクトに登録する**］チェック・ボックスをクリアすると，標準の **test_object** 引数が［**引数**］領域から自動的に削除されます（名前を変更しなかった場合）。

---

2 使用可能なオブジェクトのリストから **Test object** を選択します。たとえば，サンプルの **VerifyProperty** 関数は，**Link** テスト・オブジェクトに登録することになるでしょう。

3 テスト・オブジェクトに追加またはオーバーライドする**操作**を指定します。

- 新しい操作を定義するには，［**操作**］ボックスに新しい操作の名前を入力します。たとえば，サンプルの **VerifyProperty** 関数の場合は，新しい VerifyProperty 操作を定義します。
- 既存の操作の標準機能をオーバーライドするには，［**操作**］ボックスで使用可能な操作のリストから操作を選択します。

4 QuickTest エンジニアまたは各分野のエキスパートが，関連付けられている項目を選択したときに関数を標準設定の操作として［**操作**］カラムに表示されるようにするには，［**標準設定操作として登録する**］チェック・ボックスを選択します。

たとえば，**VerifyProperty** 操作を Link テスト・オブジェクトの標準設定の操作として定義した場合は，**RegisterUserFunc** ステートメントの 4 番目の引数に値として **True** が定義されます。その構文は次のようになります。

```
RegisterUserFunc "Link", "VerifyProperty", "VerifyProperty", True
```

テスト・オブジェクト登録情報を指定した後，関数のその他の引数を指定します。

関数の引数の指定

基本的な関数定義を行い，テスト・オブジェクト登録情報を指定したら，必要に応じて関数の引数を指定できます。

たとえば，195 ページ「関数ジェネレータを使用した関数の登録」で説明した例のように，テスト・オブジェクトに関数を登録することにした場合は，1 番目の引数の **test_object** に加えて，prop_name（チェック対象プロパティの名前）および expected_value（プロパティの期待値）という引数を割り当てることができます。関数が正しく動作するためには，要求された引数を定義する必要があります。

引数は任意の順序でリスト表示できます。ただし，関数をテスト・オブジェクトに登録する場合は，常に先頭の引数がテスト・オブジェクトを受け取らなければなりません。

関数の引数を定義するには，次の手順を実行します。

［**引数**］領域で，関数の引数を指定します。必要に応じて，引数はいくつでも追加できます。分かりやすくするために，それぞれの引数には，どのような値を入力する必要があるかを示す名前を付けてください。

- 引数を追加するには，  をクリックして引数の名前を入力します。引数には，その引数にどのような値を入力する必要があるかをはっきりと示す名前を付けてください。引数名には，英字以外の文字を含めることはできません。また，引数名は英字で始まらなければならず，スペースや以下の文字を含めてはなりません。
  @ # $ % ^ & * ( ) + = [ ] \ { } | ; ’ : " , / < > ?

  標準設定では，［**成功モード**］は［**値**］に設定されます。この設定では，QuickTest が引数として値を関数に渡します。引数値を参照によって渡す場合は，［**成功モード**］ボックスで［**リファレンス**］を選択します。
- 引数を削除するには，その引数を選択して  をクリックします。これで，その引数が関数定義ジェネレータから削除されます。

➤ 引数の順序を設定するには，矢印  と  を使用します。引数の順序は関数コードの分かりやすさに影響するだけですが，パブリック関数を登録する場合は例外です。その場合は，先頭の引数がテスト・オブジェクトを受け取る必要があります。

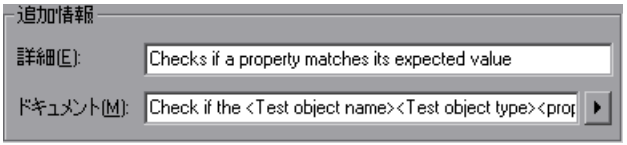
関数への説明の追加

関数定義ジェネレータでは，ユーザ定義関数にヘッダ情報を追加できます。カーソルが操作の上に置かれたときにツールチップとして表示される説明を追加できます。QuickTest エンジニアは，このツールチップの情報に基づいて，使用可能な操作のリストからどの操作を選択するべきかを判断できます（説明文は可能な限り短く簡潔にしておくことをお勧めします）。

さらに，当該の関数を使用するステップが正確に何をするかを指定する注釈を追加できます。テスト・オブジェクト名，テスト・オブジェクト・タイプ，およびテキスト内の任意の引数値を含めることができます。また，必要に応じてテキストを手作業で入力することもできます。ここで追加したテキストは，ステップ・ジェネレータの［**ステップについてのコメント**］ボックスと，キーワード・ビューの［**注釈**］カラムに表示されます。したがって，明確で分かりやすい文章にしなければなりません。

たとえば，検索エンジンから「Mercury」へのリンクをチェックする場合，関数定義ジェネレータを使用して次の注釈を定義します。

'@Documentation Check if the <Test object name> <Test object type> <prop_name> value matches the expected value: <expected_value>.



キーワード・ビューで引数の値を選択した後は，この注釈はたとえば次のように表示されます。

Check if the "Mercury Business Technology" link "text" value matches the expected value: "Mercury Business Technology Optimization (BTO) Software".

ヒント：キーワード・ビューで任意のカラム・ヘッダを右クリックして，［**ドキュメントのみ**］オプションを選択すると，ステップのリストを表示または印刷することができます。このオプションを選択すると，QuickTest に［**注釈**］カラムだけが表示されるようになります。［**編集**］＞［**ドキュメントをクリップボードにコピー**］を選択して，任意のアプリケーションに注釈を貼り付けることもできます。したがって，ステップに関してこのカラムに表示される文章は，手動テスト用の指示としても使用できる明確な文章でなければなりません。

関数に説明を付けるには，次の手順を実行します。

1 ［**詳細**］ボックスに，ツールチップとして表示されるテキストを入力します。ツールチップは，ステップ・ジェネレータの［**操作**］リスト，キーワード・ビューの［**操作**］カラムおよび IntelliSense でカーソルが関数名の上に置かれたときに表示されます。

たとえば，サンプルの **VerifyProperty** 関数の場合は，次のように入力できます。

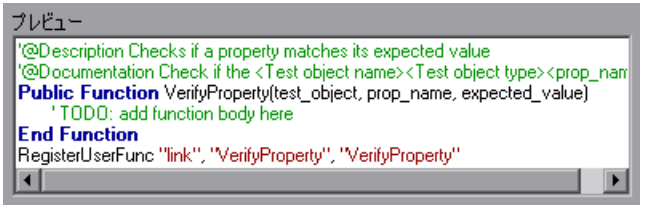
Checks whether a property value matches the actual value.

2 ［**ドキュメント**］ボックスに，ステップ・ジェネレータの［**ステップについてのコメント**］ボックスまたはキーワード・ビューの［**注釈**］カラムに表示させるテキストを入力します。［**注釈**］のテキストには引数を含めることができます。それには，  をクリックして使用する引数を選択します。［**テスト オブジェクトに登録する**］チェック・ボックスがオンになっている場合は，  をクリックすることで，表示されたリストから，［**Test object name**］項目または［**Test object type**］項目，あるいはその両方を［**注釈**］カラムに追加することもできます。これらのテスト・オブジェクト項目や引数項目を［**注釈**］テキストに含めると，それらの項目は，対応するテスト・オブジェクト名，テスト・オブジェクト・タイプ，または引数値と動的に置き換えられます。

関数のプレビュー

関数を定義するにつれて，その関数のコードが［**プレビュー**］領域に読み取り専用形式で表示されます。ここで関数の内容を確認して，必要なら関数定義ジェネレータのさまざまな領域で変更を加えることができます。

たとえば，サンプルの **VerifyProperty** 関数の場合は，［**プレビュー**］領域に次のようなコードが表示されます。



コードを確認したら（アクティブな文書に関数を挿入する前に），別の関数定義を生成するか，または定義した関数のコードを仕上げるかを選択できます。

別のユーザ定義関数の生成

コードをプレビューしたら（アクティブな文書に関数を挿入する前に），さらにほかの関数定義を生成するかどうかを選択できます。

注：ほかの関数を定義しない場合は，次の項に進みます。

ほかのユーザ定義関数を生成するには，次の手順を実行します。

1 ［**別の関数定義を挿入する**］チェック・ボックスを選択して，［**挿入**］をクリックします。アクティブな文書に関数定義が挿入され，関数定義ジェネレータからのデータが削除されます。関数定義ジェネレータは開いたままとなります。

2 194 ページ「関数の定義」で説明するように，新しい関数を定義します。

ユーザ定義関数の仕上げ

コードをプレビューしたら，関数をアクティブな文書に挿入します。関数を関数ライブラリに挿入した場合，関数ライブラリに関連付けられているテストは，関数にアクセスできます。関数をテスト（[エキスパート ビュー] 内）に直接挿入した場合，テストには，特定のアクション内の任意の場所からの関数呼び出しを含めることができます。

コードを必要な場所に挿入した後，関数を仕上げることができます。たとえば，**VerifyProperty** 関数の場合，関数ライブラリまたはテストに次のコードが挿入されます。

```
'@Description Checks whether a property matches its expected value
'@Documentation Check whether the <Test object name> <Test object type>
<prop_name> value matches the expected value: <expected_value>.Public
Function VerifyProperty (test_object, prop_name, expected_value)
    'TODO: add function body here
End Function
RegisterUserFunc "Link", "VerifyProperty", "VerifyProperty"
```

ヒント：RegisterUserFunc ステートメント（最後の行）が **VerifyProperty** 関数を Link テスト・オブジェクトに登録します。関数を複数のテスト・オブジェクトに登録するには，この行をコピーして各テスト・オブジェクト用に複製し，必要に応じて引数値を変更します。

関数を仕上げるには，その内容を追加します（TODO コメントを置き換えます）。たとえば，関数の中でプロパティの期待値が特定のテスト・オブジェクトの実際のプロパティ値と一致しているかどうかを検証するには，関数の本体に次のコードを付け加えます。

```
Dim actual_value
    ' Get the actual property value
    actual_value = obj.GetROProperty(prop_name)
    ' Compare the actual value to the expected value
    If actual_value = expected_value Then
        Reporter.ReportEvent micPass, "VerifyProperty Succeeded", "The " &
prop_name & " expected value:" & expected_value & " matches the actual value"
        VerifyProperty = True
    Else
```

```
        Reporter.ReportEvent micFail, "VerifyProperty Failed", "The " &
prop_name & " expected value:" & expected_value & " does not match the actual
value:" & actual_value
        VerifyProperty = False
    End If
```

ユーザ定義関数を仕上げるには，次の手順を実行します。

1 ［**OK**］をクリックします。アクティブな文書に関数定義が挿入され，関数定義ジェネレータが閉じます。

注：アクション内で関数を直接定義すると，その関数はそのアクションの中でのみ呼び出せます。

---

2 関数ライブラリまたはテストの中で，必要に応じて TODO 行を置き換えて，関数コードに内容を追加します。

ヒント：実行セッション後に関数をテスト結果ツリー（［テスト結果］ウィンドウ）に表示するには，**Reporter.ReportEvent** ステートメントを関数コードに追加します（前出の例のように）。ユーザ定義関数の中で標準設定のテスト・オブジェクト・メソッドを使用している場合，このステップは実行セッション後に［テスト結果］ウィンドウに表示されます。ただし，**Reporter.ReportEvent** ステートメントを関数コードに追加して追加情報を提供したり，必要に応じてテストのステータスを変更したりできます。

---

3 コードを関数ライブラリに挿入した場合は，その関数ライブラリをテストに関連付けて，そのユーザ定義関数にアクセスできるようにする必要があります。また，コードの構文を確認して，テストがその関数にアクセスできることと，QuickTest エンジニアがその関数を表示し，使用できることを確認する必要もあります。詳細については，187 ページ「関連付けられている関数ライブラリを使用した作業」を参照してください。

ユーザ定義関数のテスト・オブジェクト・メソッドとしての登録

195 ページ「関数ジェネレータを使用した関数の登録」で説明した，QuickTest 関数定義ジェネレータを使用した関数の登録に加え，**RegisterUserFunc** ステートメントを使って，テスト・オブジェクトに新規メソッドを追加したり，実行セッション中に既存のテスト・オブジェクト・メソッドの振る舞いを変更したりできます。

関数をテスト・オブジェクトに登録するときに，必要に応じてその関数をテスト・オブジェクトの標準設定の操作として定義することができます。標準設定の操作は，その関数が登録されているテスト・オブジェクトが選択されたときに標準でステップ・ジェネレータ，またはキーワード・ビューの［**操作**］カラムに表示されます。

関数をテスト・オブジェクトに登録しないと，この関数はグローバル関数になります。グローバル関数は，ステップ・ジェネレータの［**関数**］カテゴリ，またはキーワード・ビューの［**操作**］項目を選択して，または IntelliSense の使用時に呼び出します。**UnregisterUserFunc** ステートメントを使用すれば，新規メソッドを無効にしたり，既存のメソッドを QuickTest の元々の振る舞いに戻したりできます。

メソッドを登録するには，まずテストまたは関連関数ライブラリ内に関数を定義します。次に，関数の末尾に **RegisterUserFunc** ステートメントを挿入して，テスト・オブジェクト・クラス，使用する関数，および関数を呼び出すメソッド名を指定します。テスト・オブジェクト・クラスに新しいメソッドを追加することも，既存のメソッド名を使って，指定したメソッドの機能を（一時的に）オーバーライドすることもできます。

登録したメソッドは，メソッドを登録したテストまたは関数ライブラリにのみ適用されます。また，QuickTest は，各実行セッションの開始時にすべての登録関数を消去します。

ユーザ定義関数の準備

ユーザ定義関数の使用範囲をローカル・アクションに限定するには，関数をテストに直接書き込みます。ユーザ定義関数を多数のアクションやテストで使えるようにするは，関数を関連関数ライブラリに格納します（推奨）。ローカルのアクションと関連関数ライブラリの中に同じ名前の関数が存在する場合，QuickTest ではアクション内で定義されている関数が使用されます。

登録されているメソッドを含んだステートメントを実行すると，ステートメントによって対象テスト・オブジェクトが最初の引数としてメソッドに送られます。したがって，ユーザ定義関数には少なくとも 1 つの引数がなければなりません。ユーザ定義関数は任意の数の引数を取ることができます。また，テスト・オブジェクト引数のみを取ることもできます。関数が既存のメソッドをオーバーライドする場合，その構文はオーバーライド対象の関数と正確に同じでなければなりません。つまり，最初の引数はテスト・オブジェクトで，残りの引数はすべて元のメソッド引数と一致することになります。

ヒント：parent テスト・オブジェクト・プロパティを使用して，関数の最初の引数で表されるオブジェクトの親を取得できます。
例：ParentObj = obj.GetROProperty("parent")

自分で関数を書く場合には，標準の VBScript ステートメントに加え，QuickTest の任意の予約済みオブジェクト，メソッド，関数，および関数の最初の引数として渡されるテスト・オブジェクトに関連付けられている任意のメソッドが使えます。

たとえば，エディット・ボックスに新しい値を設定する前に現在の値を［テスト結果］に報告したいとします。その場合は，標準の QuickTest **Set** メソッドを，エディット・ボックスの現在の値を取得し，その値を［テスト結果］に報告してからエディット・ボックスに新しい値を設定する関数でオーバーライドします。

その関数は次のようになります。

```
Function MyFuncWithParam (obj, x)
    dim y
    y = obj.GetROProperty("value")
    Reporter.ReportEvent micDone, "previous value", y
    MyFuncWithParam=obj.Set (x)
End Function
```

注：この関数では戻り値を定義して，テストから呼び出されるたびに，**Set** メソッドの引数の値を返すようにしています。

ユーザ定義テスト・オブジェクト・メソッドの登録

RegisterUserFunc ステートメントを使って QuickTest に対して，テストの実行中，あるいはメソッドの登録を解除するまでの間，ユーザ定義関数を指定されたテスト・オブジェクト・クラスのメソッドとして使うように指示できます。

注：メソッドを登録する（そしてアクションが終っても登録を解除しない）外部アクションを呼び出した場合，登録したメソッドは，そのアクションを呼び出したテストでそれ以降も使用できます。

ユーザ定義関数をテスト・オブジェクト・メソッドとして登録するには，次の構文を使います。

RegisterUserFunc TOClass, MethodName, FunctionName, SetAsDefault

項目	詳細
TOClass	任意のテスト・オブジェクト・クラス。 **注：**QuickTest の予約済みオブジェクト（たとえば **DataTable**，**Environment**，**Reporter** など）のメソッドとしてユーザ定義メソッドを登録することはできません。
MethodName	登録するメソッドの名前（QuickTest 上で，たとえばキーワード・ビューや IntelliSense で表示されます）。特定のテスト・オブジェクト・クラスにすでに関連付けられているメソッドの名前を指定すると，ここで指定するユーザ定義関数が既存のメソッドをオーバーライドします。新しい名前を指定すると，オブジェクトがサポートするメソッドのリストにそのメソッドが追加されます。
FunctionName	テストから呼び出すユーザ定義関数の名前。この関数はテストまたは任意の関連付けられている関数ライブラリに置けます。
SetAsDefault	登録する関数を，テスト・オブジェクトの標準設定のメソッドとして使用するかどうかを示します。 キーワード・ビューまたはステップ・ジェネレータでテスト・オブジェクトを選択すると，［**操作**］カラム（キーワード・ビュー）または［**操作**］ボックス（ステップ・ジェネレータ）に標準設定のメソッドが自動的に表示されます。

ヒント：登録する関数が関数ライブラリ内で定義されている場合は，関数ライブラリに **RegisterUserFunc** ステートメントを含めることをお勧めします。これにより，当該関数ライブラリを使用する任意のテストでメソッドをすぐに利用できるようになります。

たとえば，Find Flights Web ページに［**Country**］エディット・ボックスが含まれており，標準設定でこのボックスに「USA」という値が含まれているとします。次の例では，エディット・ボックスの標準設定の値を新しい値が入力される前に取得するために，**MySet** 関数を使用するように **Set** メソッドを登録しています。

```
Function MySet (obj, x)
    dim y
    y = obj.GetROProperty("value")
    Reporter.ReportEvent micDone, "previous value", y
    MySet=obj.Set(x)
End Function

RegisterUserFunc "WebEdit", "Set", "MySet"
Browser("MercuryTours").Page("FindFlights").WebEdit("Country").Set "Canada"
```

詳細と例については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

ユーザ定義テスト・オブジェクト・メソッドの登録解除

RegisterUserFunc ステートメントを使ってメソッドを登録すると，そのメソッドはテストの終わりまで，または登録解除されるまで指定のテスト・オブジェクトのメソッドとして認識されます。このメソッドが QuickTest のメソッドをオーバーライドしている場合，このメソッドの登録を解除すると，メソッドは通常の動作に戻ります。他のメソッドの登録を解除すると，テスト・オブジェクトによってサポートされているメソッドのリストから，それらを削除することになります。

メソッドの登録解除は，再利用可能なアクションに QuickTest のメソッドをオーバーライドする登録メソッドが含まれている場合に特に重要です。たとえば，呼び出し先のアクションの中で直接定義されている関数を使うメソッドの

登録を解除しない場合，登録したメソッドが後のアクションで呼び出されると，呼び出し元のテストが失敗します。これは，関数の定義を見つけることができないことが原因です。

この場合，登録した関数が関数ライブラリ内で定義されていれば，呼び出し元のテストは正常に動作できます（関数ライブラリが呼び出し元のテストに関連付けられている場合）。しかし，呼び出し元のテストの作者は，呼び出し先のアクションに登録された関数が含まれていることに気付かずに，後のアクションで QuickTest の通常の動作を想定して，登録されたメソッドを使う可能性があります。そして，予期しない結果が生じることも考えられます。

ユーザ定義メソッドの登録を解除するには，次の構文を使用します。

UnRegisterUserFunc TOClass, MethodName

項目	詳細
TOClass	メソッドが登録されているテスト・オブジェクト・クラス。
MethodName	登録を解除するメソッド。

たとえば，Find Flights Web ページに［**Country**］エディット・ボックスが含まれており，標準設定でこのボックスに「USA」という値が含まれているとします。次の例では，エディット・ボックスの標準設定の値を新しい値が入力される前に取得するために，**MySet** 関数を使用するように **Set** メソッドを登録しています。［**Country**］エディット・ボックスのための **WebEdit.Set** ステートメントの中で，登録されたメソッドを使用した後，**Set** メソッドを標準の機能に戻すために **UnRegisterUserFunc** ステートメントを使用しています。

```
Function MySet (obj, x)
    dim y
    y = obj.GetROProperty("value")
    Reporter.ReportEvent micDone, "previous value", y
    MySet=obj.Set(x)
End Function

RegisterUserFunc "WebEdit", "Set", "MySet"
Browser("MercuryTours").Page("FindFlights").WebEdit("Country").Set "Canada"
UnRegisterUserFunc "WebEdit", "Set"
```

ユーザ定義関数の使い方のヒント

ユーザ定義関数を使用するときには，次のヒントとガイドラインを考慮してください。

- 関数定義ジェネレータを使って関数を定義し，さまざまなオプションを試してみれば，必要な構文について詳しい知識を得られます。
- 関数は，登録されると，テスト・オブジェクト・クラス全体に適用されます。特定のテスト・オブジェクトに限定してメソッドを登録することはできません。
- その他のテスト・オブジェクトから関数を呼び出す場合は，RegisterUserFunc 行をコピーして，別の関数の直後に貼り付け，適切な引数値を指定します。
- 登録する関数が関数ライブラリ内で定義されている場合は，関数ライブラリに **RegisterUserFunc** ステートメントを含めることをお勧めします。これにより，当該関数ライブラリを使用する任意のテストでメソッドをすぐに利用できるようになります。
- QuickTest は各実行セッションの開始時にすべての登録メソッドを消去します。
- ［**ステップから実行**］や［**ステップから開始**］などの部分的な実行またはデバッグ・オプションを使用して，（関数ライブラリの中でなく）テスト・ステップ内のメソッド登録の後の位置からテストの実行を開始した場合，QuickTest はメソッド登録を認識しません。これは，登録が現在の実行セッションの開始よりも前に行われるためです。
- テストに関連付けられている関数ライブラリ内で **Option Explicit** ステートメントを使用するには，ステートメントをテストに関連付けられているすべての関数ライブラリに含める必要があります。関連付けられている関数ライブラリの一部にのみ **Option Explicit** ステートメントを含めた場合，すべての関数ライブラリ内のすべての **Option Explicit** ステートメントが無視されます。**Option Explicit** ステートメントは，制限なしにアクション・スクリプトの中で直接使用することができます。
- 各関数ライブラリのグローバル・スコープにある変数は，一意でなければなりません。2 つの関連付けられている関数ライブラリにおいて，Dim ステートメントを使用してグローバル・スコープ内で同じ変数を定義している場合，または同じ名前を持つ 2 つの定数を定義している場合，2 番目の定義によって構文エラーが発生します。グローバル・スコープにおいて同じ名前を持つ 2 つ以上の変数を使用する必要がある場合，（関数ライブラリは逆順で読み込まれるため）最後の関数ライブラリにのみ Dim ステートメントを挿入します。

- 標準設定では，ユーザ定義関数を使用するステップは，実行セッション後に［テスト結果］ウィンドウのテスト結果ツリーに表示されません。関数がテスト結果ツリーに表示されるようにするには，**Reporter.ReportEvent** ステートメントを関数コードに付け加える必要があります。たとえば，必要に応じて追加情報を提供したり，テストのステータスを変更したりすることが考えられます。
- 使用されている関数を関連付けられている関数ライブラリから削除すると，その関数を使用しているテスト・ステップは  アイコンで示されます。その後，そのテストの実行セッションで，存在しない関数を使用しているステップに達すると，エラーが発生します。
- テストが参照する関数ライブラリをほかのユーザが変更した場合，または，QuickTest エンジニアが外部エディタ（QuickTest 以外）を使用して関数ライブラリを変更した場合，変更は，テストを再度開くまで反映されません。
- 同じ名前を持つ 2 つ以上の関数がテスト・スクリプトまたは関数ライブラリに存在する場合，必ず最後の関数が呼び出されます（QuickTest は，関数ライブラリを検索する前に，テスト・スクリプト内で関数を検索します）。混乱を避けるために，1 つのテストに関連付けられているリソースの中では，それぞれの関数に一意の名前を付けてください。
- 再利用可能なアクションの中でメソッドを登録した場合には，アクションの終わりにそのメソッドの登録を解除することを（そして必要があれば次のアクションの初めに再登録することを）強くお勧めします。そうすることで，そのアクションを呼び出すテストがメソッドの登録による影響を受けないようにします。
- 先にメソッドを登録解除しなくても，一度登録したメソッドが異なるユーザ定義関数を使うように登録しなおすことが可能です。ただし，このメソッドの登録を解除すると，QuickTest の本来の動作に戻り（あるいは，新規のメソッドだった場合には完全に消去され），それ以前に登録されていたものには戻りません。

  たとえば，次のステートメントを入力するとします。

  ```
  RegisterUserFunc "Link", "Click", "MyClick"
  RegisterUserFunc "Link", "Click", "MyClick2"
  UnRegisterUserFunc "Link", "Click"
  ```

  UnRegisterUserFunc ステートメントを実行した後，**Click** メソッドは **MyClick2** 関数で定義されている機能を使うのを止め，QuickTest の元の **Click** 機能に戻り，**MyClick** 関数で定義されている機能には戻りません。

➤ VBScript を使用して関数とサブルーチンを作成する方法については，QuickTest の［ヘルプ］メニューから VBScript に関するマニュアルを参照してください（［**ヘルプ**］＞［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］）。

テストからの外部定義された関数の実行

関数ライブラリ（VBScript ファイル）をテストに関連付けず，なおかつテスト内のアクションまたはほかの関数ライブラリから関数，サブルーチン，クラスなどを呼び出せるようにするには，アクションに **ExecuteFile** ステートメントを挿入します。

テストを実行すると，**ExecuteFile** ステートメントは関数ライブラリ内のすべてのグローバル・コードを実行して，そのファイル内のすべての定義がアクションのスクリプトのグローバル・スコープから利用できるようにします。

注：ExecuteFile ステートメントを使用して呼び出されるファイルおよび当該ファイルに含まれている関数をデバッグすることはできません。また，**ExecuteFile** ステートメントを含んでいるテストをデバッグする場合，実行マーカが正しく表示されないことがあります。

ヒント：作成するすべてのアクションに同じ **ExecuteFile** ステートメントを含めたければ，ステートメントをアクション・テンプレートに追加します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 492 ページ「アクション・テンプレートの作成」を参照してください。

外部定義関数を実行するには，次の手順を実行します。

1 標準の VBScript 構文を使って VBScript ファイルを作成します。詳細については，『Microsoft VBScript 言語リファレンス』（［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］＞［**VBScript**］）を参照してください。

2 作成したファイルを，テストを実行するコンピュータからアクセスできる任意のフォルダに保存します。

3 次の構文を使用して，テストのアクションに **ExecuteFile** ステートメントを追加します。

ExecuteFile FileName

ここで **FileName** は，作成した VBScript ファイルの絶対パスまたは相対パスです。

4 必要に応じて，アクション内で，指定した VBScript ファイルにある関数やサブルーチンを使用します。

注：

ExecuteFile ステートメントでは，VBScript の **ExecuteGlobal** ステートメントが利用されます。詳細については，『Microsoft VBScript 言語リファレンス』（［**QuickTest Professional ヘルプ**］＞［**VBScript リファレンス**］＞［**VBScript**］）を参照してください。

ExecuteFile ステートメントをアクション内で実行するとき，ファイル内の関数は現在のアクションからのみ呼び出せます。テスト全体で VBScript ファイル内の関数を利用できるようにするには，［テストの設定］ダイアログ・ボックスの［リソース］タブの関連関数ライブラリのリストにファイル名を追加します。詳細については，187 ページ「関連付けられている関数ライブラリを使用した作業」を参照してください。

第 7 章

QuickTest 操作のオートメーション

QuickTest を使用してアプリケーションのテストを自動化するのと同じように，QuickTest Professional のオートメーション・オブジェクト・モデルを使用して QuickTest 操作を自動化できます。QuickTest のオートメーション・オブジェクト・モデルによって公開されているオブジェクト，メソッド，およびプロパティを使用すれば，QuickTest のオプションを設定したりテストを実行したりする操作を QuickTest のインタフェースを使用して手作業で行う方法の代わりに，これらを行うプログラムを作成することができます。

オートメーション・プログラムは，同じ作業を複数回実行したい場合や，複数のテストを対象に実行したい場合，あるいは，特定の環境またはアプリケーションのニーズに合わせて QuickTest をすばやく設定したい場合に特に有用です。

本章では，次の内容について説明します。

- QuickTest 操作のオートメーションについて
- QuickTest オートメーション・プログラムを使用する条件
- オートメーション・プログラムの設計と実行に使用するプログラミング言語と開発環境の選択
- QuickTest オートメーション・プログラムの基本要素の学習
- オートメーション・スクリプトの生成
- QuickTest オートメーション・オブジェクト・モデル・リファレンスの使用

QuickTest 操作のオートメーションについて

QuickTest Professional オートメーション・オブジェクト・モデルを使用して QuickTest 操作を自動化するプログラムを作成できます。QuickTest オートメーション・オブジェクト・モデルは，別のアプリケーションから QuickTest を制御できるようにするオブジェクト，メソッド，およびプロパティを提供します。

オートメーションとは

「**オートメーション**」とは，あるアプリケーション内のソフトウェア・オブジェクトを別のアプリケーションからアクセスできるようにする Microsoft 社の技術です。これらのオブジェクトは，VBScript や VC++ などのスクリプティング言語またはプログラミング言語を使用して簡単に作成して操作できます。オートメーションを利用することで，アプリケーションの機能をプログラムの中から制御できるようになります。

「**オブジェクト・モデル**」とは，システムまたはアプリケーションの実装を構成するソフトウェア・オブジェクト（クラス）を構造化して表したものです。オブジェクト・モデルは，クラスとインタフェースのセットに加えて，プロパティ，メソッド，およびイベント，そしてそれらの関係を定義します。

QuickTest オートメーション・オブジェクト・モデルとは

QuickTest のインタフェースを通じて提供されるほとんどの設定および実行機能は，QuickTest オートメーション・オブジェクト・モデルにおいて，オブジェクト，メソッド，およびプロパティを通じて何らかの方法で表されます。必ずしも 1 対 1 の関係とはなりませんが，QuickTest の大半のダイアログ・ボックスには対応するオートメーション・オブジェクトがあり，ダイアログ・ボックスの大半のオプションは対応するオブジェクト・プロパティを使用して設定と取得が可能で，ほとんどのメニュー・コマンドおよびその他の操作は対応するオートメーション・メソッドがあります。

QuickTest オートメーション・オブジェクト・モデルによって公開されているオブジェクト，メソッド，およびプロパティを，ループや条件判断ステートメントなどの標準のプログラミング要素と組み合わせてプログラムを設計できます。

オートメーション・プログラムは，同じ作業を複数回実行したい場合や，複数のテストを対象に実行したい場合，あるいは，特定の環境またはアプリケーションのニーズに合わせて QuickTest をすばやく設定したい場合に特に有用です。

たとえば，Microsoft Visual Basic を使用して，テストに必要なアドインをロードし，QuickTest を可視モードで開始し，テストを開き，［オプション］，［テストの設定］および［記録と実行環境設定］の各ダイアログ・ボックスに対応する設定を行い，テストを実行し，テストを保存するオートメーション・プログラムを作成して実行できます。

以後，プログラムに簡単なループを追加し，1 つのプログラムで上記の操作を複数のテストに対して実行するようにできます。

また，QuickTest を特定の設定で起動する初期化プログラムを作成することも可能です。そうしておけば，テスト担当者全員にこのオートメーション・プログラムを使用して QuickTest を起動するように指示することで，テスト担当者の全員が必ず同じ設定で作業を行うことができます。

QuickTest オートメーション・プログラムを使用する条件

QuickTest を使用して設計するテストと同様に，有用な QuickTest オートメーション・プログラムを作成するには，計画，設計，テストの段階を踏む必要があります。常に，初期投資と，時間がかかったり煩雑だったりする作業を自動化することで実現される時間と人的資源の節約とを天秤にかけなければなりません。

何度も繰り返し実行する必要があったり，定期的に実行する必要があったりする QuickTest 操作は，QuickTest オートメーション・プログラムを使用する有力な候補となります。

次にいくつかの有用な QuickTest オートメーション・プログラムを示します。

- **初期化プログラム**：QuickTest を自動的に起動し，特定の環境での記録に必要なオプションおよび設定を指定するプログラムを作成できます。
- **テストの維持**：テストのコレクションを反復処理して特定の目的を達成するプログラムを作成できます。例を次に示します。
  - **値の更新**：適切なアドインを使用して各テストを開き，更新されたアプリケーションを対象に更新モードで実行した後に保存することで，アプリケーションの更新された値に合わせてすべてのテストの値を更新します。

- **既存のテストへの新規オプションの適用**：QuickTest の新しいバージョンにアップグレードしたときに，既存のテストに適用したいオプションが新しいバージョンに存在する場合があります。既存のテストをそれぞれ開いて，新しいオプションのための値を設定し，変更を保存して閉じるというプログラムを作成できます。
- **他のアプリケーションからの QuickTest の呼び出し**：QuickTest オートメーション・プログラムを実行するオプションやコントロールを持った独自のアプリケーションを設計できます。たとえば，QuickTest に精通していない製品マネージャでも QuickTest の実行予定を立てられるような Web フォームや簡単な Windows インタフェースを作成できます。

オートメーション・プログラムの設計と実行に使用するプログラミング言語と開発環境の選択

オートメーション・プログラムの作成に使用できるオブジェクト指向プログラミング言語がいくつかあります。それぞれの言語に対して，オートメーション・プログラムの設計と実行に使用できるいくつかの開発環境が提供されています。

オートメーション・プログラムの作成

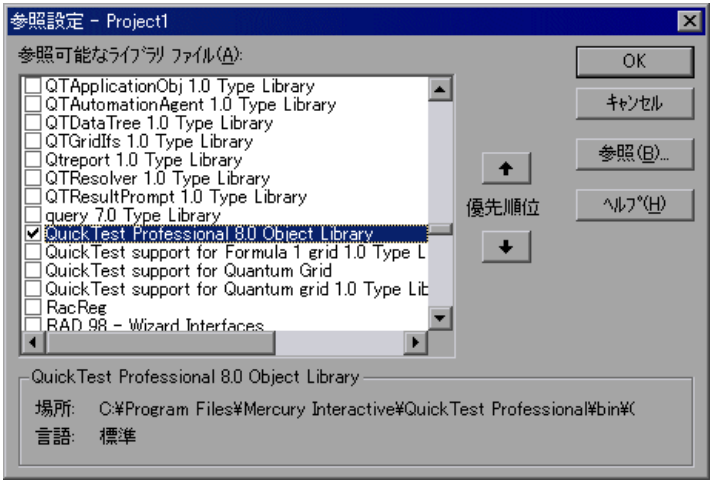
QuickTest オートメーション・プログラムはオートメーションをサポートする任意の言語および開発環境を使用して作成できます。たとえば，VBScript，JavaScript，Visual Basic，Visual C++，Visual Studio.NET を使用できます。

タイプ・ライブラリの参照がサポートされている開発環境もあります。**タイプ・ライブラリ**とは，オブジェクトの記述，インタフェース，その他，オブジェクト・モデルの定義を含んでいるバイナリ・ファイルです。

タイプ・ライブラリの参照をサポートする開発環境を選択した場合，Microsoft IntelliSense，自動ステートメント補完，ステータス・バーのツールチップなどの機能を，プログラムの作成時に利用することができます。QuickTest オートメーション・オブジェクト・モデルは，**QTObjectModel.dll** という名前のタイプ・ライブラリ・ファイルを提供します。このファイルは，**＜QuickTest のインストール・フォルダ＞¥bin** に格納されています。

タイプ・ライブラリをサポートする環境を選択する場合には，オートメーション・プログラムの作成および実行を開始する前に，必ず QuickTest タイプ・ライブラリを参照するようにします。たとえば，Microsoft Visual Basic を使用して

作業をする場合には，［**プロジェクト**］＞［**参照設定**］を選択してプロジェクトの［参照設定］ダイアログ・ボックスを開きます。続いて「**QuickTest Professional < バージョン > Object Library**」（ただし，**< バージョン >** は現在インストールされている QuickTest オートメーション・タイプ・ライブラリのバージョン）を選択します。



オートメーション・プログラムの実行

オートメーション・プログラムを実行できるアプリケーションがいくつあります。Microsoft Windows Script Host を使用すれば，オートメーション・プログラムをコマンド・ラインからも実行できます。

たとえば，次のコマンド・ラインを使用してオートメーション・プログラムを実行できます。

```
WScript.exe  /E:VBSCRIPT myScript.vbs
```

QuickTest オートメーション・プログラムの基本要素の学習

大半のオートメーション・オブジェクト・モデルと同様に，QuickTest オートメーション・オブジェクト・モデルのルート・オブジェクトは，**Application** オブジェクトです。**Application** オブジェクトは，QuickTest のアプリケーション・レベルを表します。このオブジェクトを使用して，**Test** オブジェクト（テスト・ドキュメントを表します），**Options** オブジェクト（［オプション］ダイア

ログ・ボックスを表します），**Addins** コレクション（[アドイン マネージャ] ダイアログ・ボックスのアドインのセットを表します）など，QuickTest の他の要素を返したり，アドインをロードする，QuickTest を起動する，テストを開いて保存する，QuickTest を終了するなどの操作を実行したりできます。

Application オブジェクトによって返されるオブジェクトはそれぞれ他のオブジェクトを返したり，オブジェクトに関係する操作を行ったり，オブジェクトに関連付けられているプロパティの取得と設定を行ったりできます。

オートメーション・プログラムは必ず QuickTest **Application** オブジェクトの作成から始まります。オブジェクトを作成しても QuickTest は起動しません。QuickTest オートメーション・オブジェクト・モデルの他のオブジェクト，メソッド，プロパティにアクセスするためのオブジェクトを用意するだけです。

注：必要があれば，オブジェクトを作成する対象となるリモート QuickTest コンピュータ（プログラムを実行するコンピュータ）を指定することも可能です。詳細については，オンラインの『**QuickTest オートメーション・オブジェクト・モデル・リファレンス**』の「リモート・コンピュータ上のでオートメーション・プログラムの実行」の節を参照してください。

プログラムの残りの部分の構造は，プログラムの目的に応じて異なります。QuickTest を開始する前に，テストに対応する関連アドインの取得，アドインのロード，可視モードでの QuickTest の開始など，いくつかの操作を行うことができます。これらの準備を行った後，QuickTest がコンピュータでまだ起動されていなければ，**Application.Launch** メソッドを使用して QuickTest を起動できます。オートメーション・プログラムの大半の操作は，**Launch** メソッドの後に実行します。

オートメーション・プログラムで実行できる操作の詳細については，『**QuickTest オートメーション ・オブジェクト・モデル・リファレンス**』を参照してください。このヘルプ・ファイルの詳細については，221 ページ「QuickTest オートメーション・オブジェクト・モデル・リファレンスの使用」を参照してください。

必要な操作を完了したら，あるいは，ロードされているアドインのセットを変更する場合など，QuickTest をいったん終了して再度起動する必要のある操作を実行したい場合には，**Application.Quit** メソッドを使用します。

オートメーション・スクリプトの生成

［テストの設定］ダイアログ・ボックスの［プロパティ］タブ，［オプション］ダイアログ・ボックスの［一般］タブ，および，［オブジェクトの認識］ダイアログ・ボックスのそれぞれに，［**スクリプトの生成**］ボタンがあります。このボタンをクリックすると，該当するダイアログ・ボックスの現在の設定を含んだオートメーション・スクリプト・ファイル（**.vbs**）が生成されます。

生成されたスクリプトをそのまま実行すれば，スクリプトの生成に使用した QuickTest アプリケーションとまったく同じ設定で QuickTest を起動できます。また，生成されたファイルから特定の行をコピーして，自分のオートメーション・スクリプトに貼り付けることもできます。

たとえば，［オプション］ダイアログ・ボックスから生成したスクリプトは，次のようになります。

```
Dim App 'As Application
Set App = CreateObject("QuickTest.Application")
App.Launch
App.Visible = True
App.Options.DisableVORecognition = False
App.Options.AutoGenerateWith = False
App.Options.WithGenerationLevel = 2
App.Options.TimeToActivateWinAfterPoint = 500
...
...
App.Options.WindowsApps.NonUniqueListItemRecordMode = "ByName"
App.Options.WindowsApps.RecordOwnerDrawnButtonAs = "PushButtons"
App.Folders.RemoveAll
```

［**スクリプトの生成**］ボタンの詳細，および，［オプション］，［オブジェクトの認識］，［テストの設定］の各ダイアログ・ボックスのオプションの詳細については，第 4 章「オブジェクトの認識の設定」，および『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 24 章「グローバル・テスト・オプションの設定」と第 25 章「個別のテストのオプションの設定」を参照してください。

QuickTest オートメーション・オブジェクト・モデル・リファレン

スの使用

『QuickTest オートメーション・オブジェクト・モデル・リファレンス』は，QuickTest オートメーション・オブジェクト・モデルのオブジェクト，メソッド，プロパティに関する詳細な説明，構文情報，使用例を提供するヘルプ・ファイルです。

『**QuickTest オートメーション・オブジェクト・モデル・リファレンス**』は，次の場所から表示できます。

- QuickTest プログラム・フォルダ（［**スタート**］メニューの［**QuickTest Professional**］プログラム・グループから［**Documentation**］＞［**QuickTest Automation Reference**］を選択します）
- QuickTest の［ヘルプ］メニュー（［**ヘルプ**］＞［**QuickTest オートメーション オブジェクト モデル リファレンス**］）

第 2 部

オブジェクト・リポジトリの管理と結合

第 8 章

オブジェクト・リポジトリの管理

オブジェクト・リポジトリ・マネージャでは，オブジェクトの追加および定義，オブジェクトおよびその記述の変更，リポジトリの汎用性を高めるためのパラメータ化，リポジトリの保守および組織化，リポジトリの結合，XML 形式でのリポジトリのインポートおよびエクスポートなど，組織で使用されているすべての共有オブジェクト・リポジトリを一元管理することができます。

本章では，次の内容について説明します。

- オブジェクト・リポジトリの管理について
- オブジェクト・リポジトリ・マネージャについて
- オブジェクト・リポジトリを使った作業
- オブジェクト・リポジトリの変更
- リポジトリ・パラメータを使用した作業
- テスト・オブジェクトの詳細の変更
- オブジェクトの検索
- 結合操作の実行
- インポートおよびエクスポート操作の実行

オブジェクト・リポジトリの管理について

オブジェクト・リポジトリ・マネージャでは，共有オブジェクト・リポジトリの作成および保守ができます。ファイル・システムおよび Quality Center プロジェクトに保存されているオブジェクト・リポジトリを使用できます。

各オブジェクト・リポジトリには，QuickTest によるアプリケーション内のオブジェクトの識別を可能にする情報が含まれています。QuickTest により，テスト・オブジェクトに関するすべての情報を共有オブジェクト・リポジトリに格納することで，テストの再利用性を維持できます。アプリケーションのオブジェクトに変更があった場合，オブジェクト・リポジトリ・マネージャは，複数のテストのテスト・オブジェクト情報を 1 か所で集中して更新できる場所となります。

注：共有オブジェクト・リポジトリの代わりに，または共有オブジェクト・リポジトリに加えて，一部または全部のオブジェクトをアクションごとにローカル・オブジェクト・リポジトリに保存するという選択肢もあります。ローカル・オブジェクト・リポジトリの詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の第 6 章「テスト・オブジェクトを使用した作業」を参照してください。

同じアクションに関連付けられているローカル・オブジェクト・リポジトリと共有オブジェクト・リポジトリの両方に，同じ名前と記述を持つオブジェクトが存在する場合は，そのアクションに対するローカルのオブジェクト定義が使用されます。同じアクションに関連付けられている複数の共有オブジェクト・リポジトリ内に同じ名前と記述を持つオブジェクトがある場合，共有オブジェクト・リポジトリがアクションに関連付けられている順序に従って，最初に出現したオブジェクトのオブジェクト定義が使用されます。共有オブジェクト・リポジトリの関連付けの詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 476 ページ「オブジェクト・リポジトリとアクションの関連付け」を参照してください。

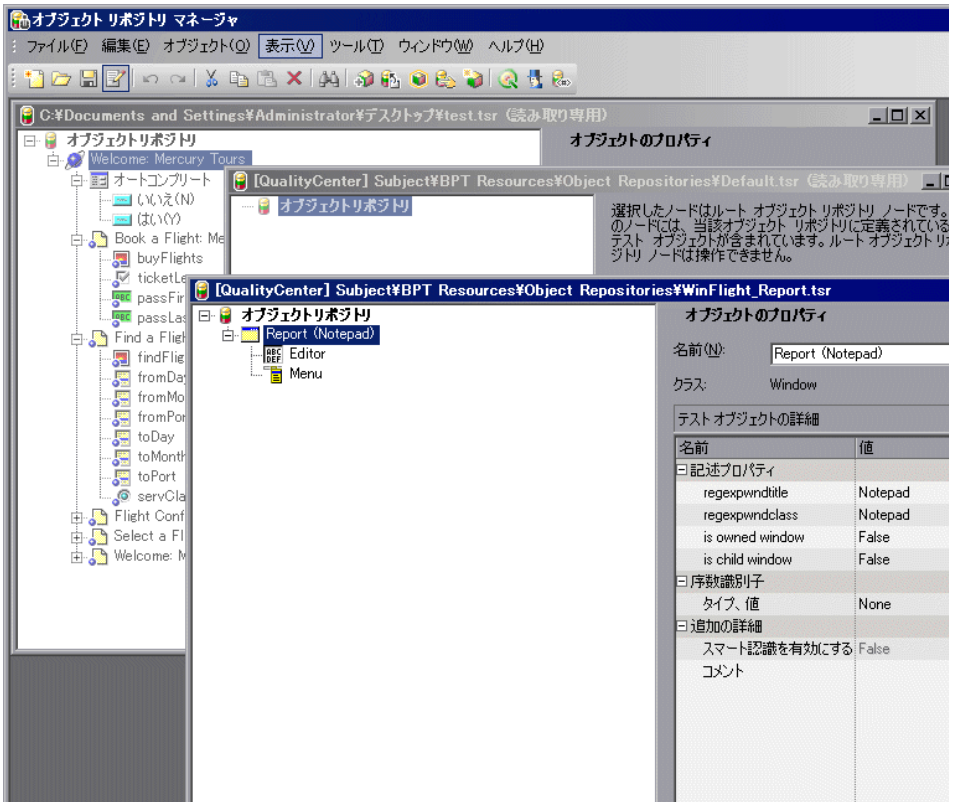
複数のアクションで同じ共有オブジェクト・リポジトリを使用できます。1 つのアクションで複数のオブジェクト・リポジトリを使用することもできます。また，アクションを使用してオブジェクトをローカル・オブジェクト・リポジトリに直接保存することもできます。これにより，オブジェクトはそのアクションからのみアクセスできます。

アプリケーション内のオブジェクトのプロパティ値の中に，QuickTest によってオブジェクトの識別に使用されるプロパティ値と異なるものがあると，テストは失敗することがあります。そのため，アプリケーション内にあるオブジェクトのプロパティ値に変更があった場合，既存のテストを継続して使用するには，該当するオブジェクト・リポジトリ内にある該当するテスト・オブジェクトのプロパティ値を修正する必要があります。

本章で説明するように，共有オブジェクト・リポジトリのオブジェクトを変更するには，オブジェクト・リポジトリ・マネージャを使用します。ローカル・オブジェクト・リポジトリに格納されているオブジェクトを変更するには，［オブジェクト リポジトリ］ウィンドウを使用します。［オブジェクト リポジトリ］ウィンドウの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 6 章「テスト・オブジェクトを使用した作業」を参照してください。

オブジェクト・リポジトリ・マネージャについて

オブジェクト・リポジトリ・マネージャを開くには，［**リソース**］＞［**オブジェクト リポジトリ マネージャ**］を選択します。オブジェクト・リポジトリ・マネージャでは，複数の共有オブジェクト・リポジトリを開き，必要に応じてそれらを変更できます。オブジェクト・リポジトリは，ファイル・システムまたは Quality Center プロジェクトから開くことができます。



ヒント：オブジェクト・リポジトリ・マネージャを開いている間も，引き続き QuickTest の他のウィンドウで作業が行えます。

共有オブジェクト・リポジトリは，必要な数だけ開けます。共有オブジェクト・リポジトリは，それぞれ別のドキュメント・ウィンドウに開きます。開いたウィンドウは，必要に応じてサイズ変更，最大化，最小化して配置することで，共有オブジェクト・リポジトリ間でオブジェクトのコピー，ドラッグしての移動などができるほか，個別のオブジェクト・リポジトリを対象に操作が行えます。共有オブジェクト・リポジトリ・ウィンドウに表示される情報の詳細については，233 ページ「共有オブジェクト・リポジトリ・ウィンドウについて」を参照してください。

共有オブジェクト・リポジトリは，［共有オブジェクト リポジトリを開く］ダイアログ・ボックスから開きます。このダイアログ・ボックスの［**読み取り専用モードで開く**］チェック・ボックスは標準設定で選択されています。このチェック・ボックスをクリアすると，共有オブジェクト・リポジトリは編集可能なモードで開きます。それ以外の場合は，共有オブジェクト・リポジトリは読み取り専用モードで開きます。変更するには［**編集を有効化**］ボタンをクリックする必要があります。詳細については，242 ページ「オブジェクト・リポジトリの編集」を参照してください。

オブジェクト・リポジトリ・マネージャの中でメニュー項目を選択するかツールバー・ボタンをクリックすると，ウィンドウが現在アクティブな（フォーカスがある）共有オブジェクト・リポジトリを対象に，選択した操作が実行されます。ウィンドウのタイトル・バーに，共有オブジェクト・リポジトリの名前およびファイル・パスが表示されます。オブジェクト・リポジトリ・マネージャのツールバー・ボタンの詳細については，230 ページ「オブジェクト・リポジトリ・マネージャ・ツールバーの使用について」を参照してください。

オブジェクト・リポジトリ・マネージャの中で実行できる共有オブジェクト・リポジトリ操作の多くは，（［オブジェクト リポジトリ］ウィンドウを使用して）ローカル・オブジェクト・リポジトリに格納されているオブジェクトに変更を加える場合と似ています。したがって，手順の多くは『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 6 章「テスト・オブジェクトを使用した作業」で説明しています。手順のほとんどは，オブジェクト・リポジトリ・マネージャと［オブジェクト リポジトリ］ウィンドウで同じですが，ウィンドウとオプションに若干の相違がある場合があります。

オブジェクト・リポジトリ・マネージャ・ツールバーの使用について

よく実行する操作は，オブジェクト・リポジトリ・マネージャ・ツールバーから実行できます。オブジェクト・リポジトリ・マネージャ・ツールバーには，次のボタンがあります。

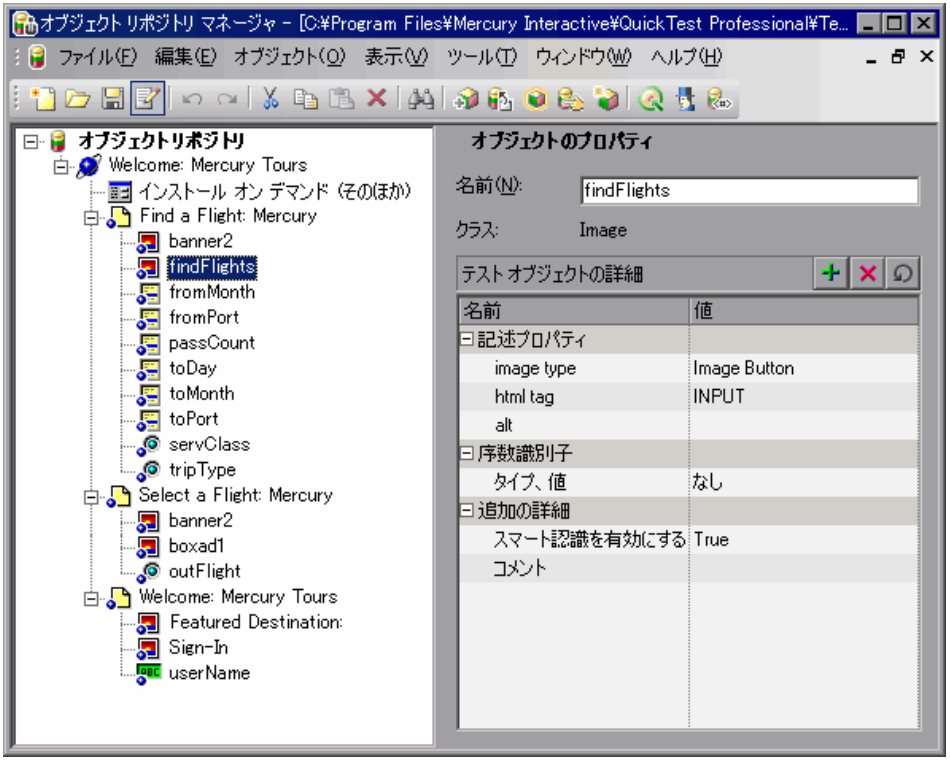
ボタン	詳細
	新規の共有オブジェクト・リポジトリを作成します。詳細については，235 ページ「新しいオブジェクト・リポジトリの作成」を参照してください。
	ファイル・システムまたは Quality Center から共有オブジェクト・リポジトリを開きます。詳細については，235 ページ「オブジェクト・リポジトリを開く」を参照してください。
	アクティブな共有オブジェクト・リポジトリをファイル・システムまたは Quality Center に保存します。詳細については，237 ページ「オブジェクト・リポジトリの保存」を参照してください。
	アクティブな共有オブジェクト・リポジトリを編集可能にすることで，共有オブジェクト・リポジトリを編集します。詳細については，242 ページ「オブジェクト・リポジトリの編集」を参照してください。
	アクティブな共有オブジェクト・リポジトリで行った 1 つ前の操作を元に戻します。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 203 ページ「オブジェクト・リポジトリ内のオブジェクトのコピー，貼り付け，および移動」を参照してください。
	アクティブな共有オブジェクト・リポジトリで元に戻した操作を再度実行します。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 203 ページ「オブジェクト・リポジトリ内のオブジェクトのコピー，貼り付け，および移動」を参照してください。
	アクティブな共有オブジェクト・リポジトリで選択されている項目またはオブジェクトを切り取ります。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 203 ページ「オブジェクト・リポジトリ内のオブジェクトのコピー，貼り付け，および移動」を参照してください。

ボタン	詳細
	アクティブな共有オブジェクト・リポジトリで選択されている項目またはオブジェクトをクリップボードにコピーします。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 203 ページ「オブジェクト・リポジトリ内のオブジェクトのコピー，貼り付け，および移動」を参照してください。
	クリップボードのデータをアクティブな共有オブジェクト・リポジトリに貼り付けます。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 203 ページ「オブジェクト・リポジトリ内のオブジェクトのコピー，貼り付け，および移動」を参照してください。
	アクティブな共有オブジェクト・リポジトリで選択されている項目またはオブジェクトを削除します。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 206 ページ「オブジェクト・リポジトリからのオブジェクトの削除」を参照してください。
	アクティブな共有オブジェクト・リポジトリ内のオブジェクト，プロパティ，またはプロパティ値を検索します。指定したプロパティ値を検索して置換することもできます。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 207 ページ「オブジェクト・リポジトリ内のオブジェクトの検索」を参照してください。
	アクティブな共有オブジェクト・リポジトリにオブジェクトを追加します。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 191 ページ「オブジェクト・リポジトリへのオブジェクトの追加」を参照してください。
	アプリケーション内のオブジェクトの実際のプロパティに従って，アクティブな共有オブジェクト・リポジトリ内のテスト・オブジェクト・プロパティを更新します。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 172 ページ「アプリケーション内のオブジェクトからのテスト・オブジェクト・プロパティの更新」を参照してください。

ボタン	詳細
	アクティブな共有オブジェクト・リポジトリ内でオブジェクトを選択すると，それがアプリケーション内で強調表示されます。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 210 ページ「アプリケーション内のオブジェクトの強調表示」を参照してください。
	アプリケーション内でオブジェクトを選択すると，アクティブな共有オブジェクト・リポジトリ内でそれが強調表示されます。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 211 ページ「オブジェクト・リポジトリ内のオブジェクトの場所の特定」を参照してください。
	アプリケーションに存在しないテスト・オブジェクトを定義し，アクティブな共有オブジェクト・リポジトリにそれを追加します。ローカル・オブジェクト・リポジトリでの場合と同じように行います。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 201 ページ「新規テスト・オブジェクトの定義」を参照してください。
	Quality Center に接続し，Quality Center プロジェクトに格納されているオブジェクト・リポジトリ・ファイルを使って作業を行います。Quality Center には，QuickTest のメイン・ウィンドウから，またはオブジェクト・リポジトリ・マネージャから接続します。詳細については，350 ページ「Quality Center への QuickTest の接続」を参照してください。
	オブジェクト・スパイを開き，アプリケーション内の実行時オブジェクトまたはテスト・オブジェクトのプロパティと値を表示します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 71 ページ「オブジェクト・スパイを使用したオブジェクトのプロパティの表示」を参照してください。
	アクティブな共有オブジェクト・リポジトリ内のリポジトリ・パラメータを追加，編集，および削除します。詳細については，244 ページ「リポジトリ・パラメータの管理」を参照してください。

共有オブジェクト・リポジトリ・ウィンドウについて

オブジェクト・リポジトリ・マネージャで開いた各共有オブジェクト・リポジトリは，個別のドキュメント・ウィンドウに表示されます。各共有オブジェクト・リポジトリのウィンドウには，オブジェクト・リポジトリ内のすべてのオブジェクトのツリーが，選択したオブジェクトのテスト・オブジェクト情報とともに表示されます。



［オブジェクト リポジトリ］ウィンドウには，ツリーの中で選択した各テスト・オブジェクトの情報が表示されます。共有オブジェクト・リポジトリ内の任意のテスト・オブジェクトのテスト・オブジェクト記述の表示，テスト・オブジェクトやそのプロパティの変更，共有オブジェクト・リポジトリへのオブジェクトの追加を行うことができます。詳細については，240 ページ「オブジェクト・リポジトリの変更」および 249 ページ「テスト・オブジェクトの詳細の変更」を参照してください。

各オブジェクト・リポジトリ・ウィンドウには，次の情報が表示されます。

情報	詳細
［オブジェクト リポジトリ］ツリー	共有オブジェクト・リポジトリのすべてのテスト・オブジェクトが含まれています。
［名前］	QuickTest によって選択したテスト・オブジェクトに割り当てられている名前です。テスト・オブジェクトの名前は変更できます。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 175 ページ「テスト・オブジェクトの名前の変更」を参照してください。
［クラス］	選択したオブジェクトのクラスです。
［テスト オブジェクトの詳細］	実行セッション中に，選択したオブジェクトの識別に使用されるプロパティおよびプロパティ値の表示や変更ができます。詳細については，249 ページ「テスト・オブジェクトの詳細の変更」を参照してください。

注 : テスト・オブジェクトが含まれるステップをアクションから削除しても，オブジェクトはオブジェクト・リポジトリから削除されません。共有オブジェクト・リポジトリのオブジェクトを削除するには，オブジェクト・リポジトリ・マネージャを使用して，ローカル・オブジェクト・リポジトリからオブジェクトを削除する場合とほぼ同じ方法で削除します。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 206 ページ「オブジェクト・リポジトリからのオブジェクトの削除」を参照してください。

オブジェクト・リポジトリを使った作業

オブジェクト・リポジトリ・マネージャを使用して，オブジェクト・リポジトリを新規作成し，既存のオブジェクト・リポジトリを開いて変更し，終了時にはファイルを保存して閉じることができます。

新しいオブジェクト・リポジトリの作成

新しいオブジェクト・リポジトリを作成してオブジェクトを追加し，保存することができます。その後，QuickTest 内から，1 つ以上のアクションをオブジェクト・リポジトリに関連付けることができます。共有オブジェクト・リポジトリの関連付けの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 476 ページ「オブジェクト・リポジトリとアクションの関連付け」を参照してください。

新しいオブジェクト・リポジトリを作成するには，次の手順を実行します。

オブジェクト・リポジトリ・マネージャの中で，［**ファイル**］＞［**新規作成**］を選択するか，［**新規作成**］ボタンをクリックします。新しいオブジェクト・リポジトリが開きます。これで，オブジェクトの追加や，オブジェクト・リポジトリの変更および保存ができます。詳細については，240 ページ「オブジェクト・リポジトリの変更」および 237 ページ「オブジェクト・リポジトリの保存」を参照してください。

オブジェクト・リポジトリを開く

既存のオブジェクト・リポジトリを開き，表示または変更できます。オブジェクト・リポジトリは，ファイル・システムまたは Quality Center プロジェクトから開くことができます。

Quality Center プロジェクトには，QuickTest またはオブジェクト・リポジトリ・マネージャから，［**ファイル**］＞［**Quality Center の接続**］を選択するか，［**Quality Center の接続**］ボタンをクリックして接続します。Quality Center への接続の詳細については，350 ページ「Quality Center への QuickTest の接続」を参照してください。

QuickTest の以前のバージョンをお使いのユーザの方へ：

以前のバージョンの QuickTest を使用して作成されたオブジェクト・リポジトリを開くとき，オブジェクト・リポジトリを編集する前に，QuickTest を使用して最新の形式への変換を行う必要があります。

オブジェクト・リポジトリに外部アドインのテスト・オブジェクトが含まれている場合，オブジェクト・リポジトリを現在の形式に変換するためには，該当するアドインがインストールされている必要があります。インストールされていない場合，オブジェクト・リポジトリは読み取り専用形式でのみ開けます。

オブジェクト・リポジトリを変換しない場合は，オブジェクト・リポジトリ・ファイルは読み取り専用形式で表示されます。いったんファイルを変換して保存すると，そのファイルは以前のバージョンの QuickTest で使用できなくなります。

オブジェクト・リポジトリを開くには，次の手順を実行します。

1 オブジェクト・リポジトリ・マネージャの中で，［**ファイル**］＞［**開く**］を選択するか，［**開く**］ボタンをクリックします。［共有オブジェクト リポジトリを開く］ダイアログ・ボックスが表示されます。

注：Quality Center に接続している場合と，標準ファイル・システムを使用する場合とでは，表示されるダイアログ・ボックスが異なります。［共有オブジェクト リポジトリを開く］ダイアログ・ボックスで［**ファイル システム**］ボタンおよび［**Quality Center**］ボタンをクリックすることで，ダイアログ・ボックスの切り替えができます。

2 開く対象となるオブジェクト・リポジトリを選択し，［**開く**］または［**OK**］をクリックします（ファイル・システムまたは Quality Center プロジェクトのどちらから開いているかによります）。オブジェクト・リポジトリが開きます。

標準設定では，オブジェクト・リポジトリは読み取り専用モードで開きます。編集可能な形式で開くには，［共有オブジェクト リポジトリを開く］ダイアログ・ボックスで［**読み取り専用モードで開く**］チェック・ボックスをクリアします。242 ページ「オブジェクト・リポジトリの編集」で説明している方法で，オブジェクト・リポジトリを編集することもできます。

オブジェクト・リポジトリが編集可能になっている場合は，オブジェクトの追加，オブジェクト・リポジトリの変更，および保存ができます。詳細については，240 ページ「オブジェクト・リポジトリの変更」および 237 ページ「オブジェクト・リポジトリの保存」を参照してください。

ヒント：オブジェクト・リポジトリは，［**ファイル**］メニューの［**最近使用したファイル**］のリストからも開けます。

オブジェクト・リポジトリの保存

オブジェクト・リポジトリを作成または変更したら，保存する必要があります。オブジェクト・リポジトリに変更を加えると，オブジェクト・リポジトリを保存するまで，タイトル・バーにアスタリスク（*）が表示されます。

オブジェクト・リポジトリは，ファイル・システムまたは Quality Center プロジェクト（Quality Center プロジェクトに接続している場合）に保存できます。Quality Center プロジェクトに接続するには，QuickTest またはオブジェクト・リポジトリ・マネージャから，［**ファイル**］＞［**Quality Center の接続**］を選択するか，［**Quality Center の接続**］ボタンをクリックします。Quality Center への接続の詳細については，350 ページ「Quality Center への QuickTest の接続」を参照してください。

注：オブジェクト・リポジトリに加えた変更はすべて，まだ変更内容を保存していなくても，変更した直後に，オブジェクト・リポジトリを使用している同じコンピュータ上で開いているすべてのテストにおいて自動的に更新されます。変更を保存せずにオブジェクト・リポジトリを閉じると，変更時に開いていたすべてのテストにおいて，反映されていた変更内容がロールバックされます。オブジェクト・リポジトリを変更したコンピュータ上でテストを開くと，テストは，関連付けられているオブジェクト・リポジトリに保存した変更内容で自動的に更新されます。保存した変更を，別のコンピュータ上で開いているテストまたはリポジトリに反映するには，そのテストまたはオブジェクト・リポジトリを開くかコンピュータ上で編集用にロックすることによって，変更を読み込む必要があります。

オブジェクト・リポジトリを保存するには，次の手順を実行します。

1 保存するオブジェクト・リポジトリがアクティブ・ウィンドウであることを確認します。

2 [**ファイル**] > [**保存**] を選択するか，[**保存**] ボタンをクリックします。ファイルがすでに保存されている場合は，変更が保存されます。ファイルがまだ保存されていない場合は，[共有オブジェクト リポジトリの保存] ダイアログ・ボックスが開きます。

注：Quality Center に接続している場合と，標準ファイル・システムを使用する場合とでは，表示されるダイアログ・ボックスが異なります。[共有オブジェクト リポジトリを開く] ダイアログ・ボックスで [**ファイル システム**] ボタンおよび [**Quality Center**] ボタンをクリックすることで，ダイアログ・ボックスの切り替えができます。

---

3 オブジェクト・リポジトリを保存するフォルダを選択します。

4 [**ファイル名**] または [**添付名**] ボックスに，オブジェクト・リポジトリの名前を入力します（ファイル・システムまたは Quality Center プロジェクトのどちらへ保存するかによります）。ファイルを識別しやすいように，分かりやすい名前を使用します。

注：オブジェクト・リポジトリ名には，「¥」，「/」，「:」，「*」，「"」，「?」，「<」，「>」，「|」の文字は使用できません。

---

5 ［**保存**］または［**OK**］をクリックします（ファイル・システムまたは Quality Center プロジェクトのどちらへ保存するかによります）。QuickTest はオブジェクト・リポジトリを拡張子 **.tsr** を付けて指定された場所に保存し，オブジェクト・リポジトリの名前とパスをリポジトリ・ウィンドウのタイトル・バーに表示します。

オブジェクト・リポジトリを閉じる

オブジェクト・リポジトリの変更または使用が終了したら，オブジェクト・リポジトリを閉じます。ファイルを閉じると，他の人が使用したり変更したりできるように，ファイルのロックが自動的に解除されます。開いているすべてのオブジェクト・リポジトリを閉じることもできます。

注：QuickTest を終了すると，オブジェクト・リポジトリ・マネージャも終了します。変更内容をまだ保存していない場合は，オブジェクト・リポジトリ・マネージャを終了する前に保存するよう求められます。

オブジェクト・リポジトリを閉じるには，次の手順を実行します。

1 閉じる対象となるオブジェクト・リポジトリがアクティブ・ウィンドウであることを確認します。

2 ［**ファイル**］＞［**閉じる**］を選択するか，オブジェクト・リポジトリ・ウィンドウのタイトル・バーで［**閉じる**］ボタンをクリックします。オブジェクト・リポジトリが閉じ，ロックが自動的に解除されます。変更内容をまだ保存していない場合は，ファイルを閉じる前に保存するよう求められます。

開いているすべてのオブジェクト・リポジトリを閉じるには，次の手順を実行します。

［**ファイル**］＞［**すべてのウィンドウを閉じる**］または，［**ウィンドウ**］＞［**Close All Windows**］を選択します。開いているすべてのオブジェクト・リポジトリが閉じ，ロックが自動的に解除されます。変更内容をまだ保存していない場合は，ファイルを閉じる前に保存するよう求められます。

オブジェクト・リポジトリの変更

オブジェクト・リポジトリを初めて使用するための準備を整えるために，または，テスト・プロセス全体を通じてオブジェクト・リポジトリを更新するために，さまざまな方法でオブジェクト・リポジトリを変更することができます。共有オブジェクト・リポジトリのオブジェクトやオブジェクト・プロパティの追加および変更，オブジェクト・リポジトリ間でのオブジェクトのコピーまたは移動，階層内の別の場所へのオブジェクトのドラッグ，オブジェクトの削除，オブジェクト名の変更ができます。オブジェクト・リポジトリに変更を加えると，オブジェクト・リポジトリを保存するまで，タイトル・バーにアスタリスク（*****）が表示されます。

ヒント：必要に応じて，［**編集**］＞［**元に戻す**］や［**編集**］＞［**やり直し**］オプションまたは［**元に戻す**］ボタンや［**やり直し**］ボタンを使用して，変更の取り消しや繰り返しを行うことができます。［**元に戻す**］および［**やり直し**］オプションは，アクティブなドキュメントを対象とします。オブジェクト・リポジトリを保存すると，保存前にファイルに対して行った操作の取り消しや，やり直しはできません。

オブジェクト・リポジトリを読み取り専用モードで開いた場合，変更を加えるためには，オブジェクト・リポジトリを編集可能にする必要があります。編集可能にすると、オブジェクト・リポジトリがロックされ，複数のユーザが同時に変更できないようになります。

注：オブジェクト・リポジトリに加えた変更はすべて，まだ変更内容を保存していなくても，変更した直後に，オブジェクト・リポジトリを使用している同じコンピュータ上で開いているすべてのテストにおいて自動的に更新されます。変更を保存せずにオブジェクト・リポジトリを閉じると，変更時に開いていたすべてのテストにおいて，反映されていた変更内容がロールバックされます。オブジェクト・リポジトリを変更したコンピュータ上でテストを開くと，テストは，関連付けられているオブジェクト・リポジトリに保存した変更内容で自動的に更新されます。保存した変更を，別のコンピュータ上で開いているテストまたはリポジトリに反映するには，そのテストまたはオブジェクト・リポジトリを開くかコンピュータ上で編集用にロックすることによって，変更を読み込む必要があります。

ヒント：共有オブジェクト・リポジトリは，別の共有オブジェクト・リポジトリと結合することでも変更できます。2 つの共有オブジェクト・リポジトリを結合すると，両方のオブジェクト・リポジトリの内容を合わせた新しい共有オブジェクト・リポジトリが作成されます。ローカル・オブジェクト・リポジトリを共有オブジェクト・リポジトリに結合すると，共有オブジェクト・リポジトリが，ローカル・オブジェクト・リポジトリの内容で更新されます。詳細については，第 9 章「共有オブジェクト・リポジトリの結合」を参照してください。

共有オブジェクト・リポジトリが編集可能であり，アクティブ・ウィンドウであることを確認したら，ローカル・オブジェクト・リポジトリを変更するときと同じ方法で共有オブジェクト・リポジトリを変更します。詳細については，次を参照してください。

- 次の「オブジェクト・リポジトリの編集」
- 『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 191 ページ「オブジェクト・リポジトリへのオブジェクトの追加」
- 『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 203 ページ「オブジェクト・リポジトリ内のオブジェクトのコピー，貼り付け，および移動」
- 『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 206 ページ「オブジェクト・リポジトリからのオブジェクトの削除」

オブジェクト・リポジトリの編集

標準設定では，オブジェクト・リポジトリは読み取り専用モードで開きます。編集可能な形式で開くには，開くときに［共有オブジェクト リポジトリを開く］ダイアログ・ボックスで［**読み取り専用モードで開く**］チェック・ボックスをクリアします。

オブジェクト・リポジトリを読み取り専用モードで開いた場合，変更を加えるためには，オブジェクト・リポジトリを編集可能にする必要があります。オブジェクト・リポジトリを表示するだけなら，あるいは，別のオブジェクト・リポジトリにオブジェクトをコピーするだけなら，編集可能にする必要はありません。

オブジェクト・リポジトリを編集可能にすると，他のユーザが変更できないようにオブジェクト・リポジトリがロックされます。オブジェクト・リポジトリを他のユーザから変更できるようにするには，ロックを解除する必要があります（編集モードを無効にするか，オブジェクト・リポジトリを閉じます）。オブジェクト・リポジトリがすでに別のユーザによってロックされている場合，読み取り専用形式で保存されている場合，またはオブジェクト・リポジトリを開くための権限がユーザにない場合は，編集可能にすることはできません。

QuickTest の以前のバージョンをお使いのユーザの方へ：以前のバージョンの QuickTest を使用して作成されたオブジェクト・リポジトリを編集する場合，オブジェクト・リポジトリを編集する前に，QuickTest を使用して最新の形式に変換する必要があります。変換しない場合は，オブジェクト・リポジトリ・ファイルは読み取り専用形式で表示されます。いったんファイルを変換して保存すると，ファイルは以前のバージョンの QuickTest で使用できなくなります。

オブジェクト・リポジトリを編集可能にするには，次の手順を実行します。

1 編集対象オブジェクト・リポジトリがアクティブ・ウィンドウであることを確認します。

2 ［**ファイル**］＞［**編集を有効化**］を選択するか，［**編集を有効化**］ボタンをクリックします。オブジェクト・リポジトリが編集可能になります。

リポジトリ・パラメータを使用した作業

リポジトリ・パラメータを使用すれば，特定のプロパティ値をパラメータ化するように指定しつつ，実際のパラメータ化の定義は，パラメータ化の対象となるテスト・オブジェクト・プロパティ値を含んでいるオブジェクト・リポジトリに関連付けられている各テストで行うようにできます。

リポジトリ・パラメータは，動的に変化するオブジェクトを対象にテストを作成し，実行する場合に便利です。オブジェクトがアプリケーション内で頻繁に更新される場合，またはデータベースなどの動的コンテンツによってプロパティ値が設定される場合，オブジェクトは動的に変化することがあります。

たとえば，ローカライズされたアプリケーションにおいて，テキストのプロパティ値がユーザ・インタフェースの言語に応じて変化するボタンがある場合があります。リポジトリ・パラメータを使用して名前のプロパティ値をパラメータ化した後，当該オブジェクト・リポジトリを使用する各テストにおいて，プロパティ値をどこから取得するかを指定できます。たとえば，このオブジェクト・リポジトリを使用するあるテストではプロパティ値を環境変数から取得し，別のテストではデータ・テーブルから取得し，さらに別のテストでは定数値を使用するように指定できます。

特定のオブジェクト・リポジトリに対するリポジトリ・パラメータはすべて，［リポジトリ パラメータの管理］ダイアログ・ボックスを使用して定義します。各リポジトリ・パラメータは，任意で標準設定値を指定し，分かりやすい説明とともに定義します。詳細については，244 ページ「リポジトリ・パラメータの管理」を参照してください。

標準設定値が定義されていないリポジトリ・パラメータを持つオブジェクト・リポジトリを使用するテストを開くと，欠落リソース表示枠に，割り当てが必要なリポジトリ・パラメータが存在することを示す指示が表示されます。その場合は，そのテスト内のリポジトリ・パラメータを必要に応じて割り当てることができます。また，標準設定値があるリポジトリ・パラメータの割り当てを行ったり，すでに割り当てのあるリポジトリ・パラメータの割り当てを変更したりもできます。リポジトリ・パラメータの割り当ての詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 504 ページ「未割り当ての共有オブジェクト・リポジトリ・パラメータ値の処理」を参照してください。

リポジトリ・パラメータの管理

［リポジトリ パラメータの管理］ダイアログ・ボックスでは，1 つの共有オブジェクト・リポジトリを対象に，リポジトリ・パラメータの追加，編集，および削除ができます。

リポジトリ・パラメータを管理するには，次の手順を実行します。

1 パラメータを管理する対象となるオブジェクト・リポジトリがアクティブ・ウィンドウであることを確認します。

2 オブジェクト・リポジトリが読み取り専用形式の場合は，［**ファイル**］＞［**編集を有効化**］を選択するか，［**編集を有効化**］ボタンをクリックします。オブジェクト・リポジトリが編集可能になります。

3 ［**ツール**］＞［**リポジトリ パラメータの管理**］を選択するか，［**リポジトリ パラメータの管理**］ボタンをクリックします。［リポジトリ パラメータの管理］ダイアログ・ボックスが表示されます。

［リポジトリ パラメータの管理］ダイアログ・ボックスには，次の情報およびオプションがあります。

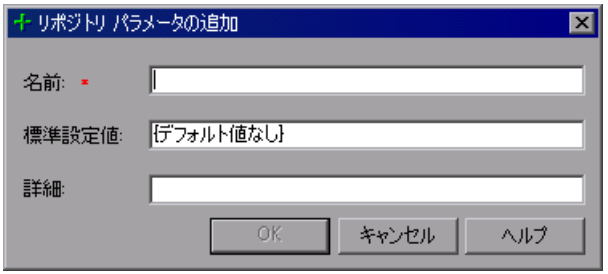
オプション	詳細
［名前］	リポジトリ・パラメータを管理する対象となるオブジェクト・リポジトリの名前およびパスが表示されます。
	新しいリポジトリ・パラメータを追加します。詳細については，246 ページ「リポジトリ・パラメータの追加」を参照してください。
	現在選択されているリポジトリ・パラメータを削除します。詳細については，248 ページ「リポジトリ・パラメータの削除」を参照してください。
パラメータ・リスト	オブジェクト・リポジトリに現在定義されているリポジトリ・パラメータのリストが表示されます。パラメータ・リストでは，パラメータの標準設定の値および説明を直接変更できます。詳細については，248 ページ「リポジトリ・パラメータの変更」を参照してください。
［リポジトリの中で検索］	オブジェクト・リポジトリ・ツリー内で，選択されているリポジトリ・パラメータを使用している最初のテスト・オブジェクトを検索して強調表示します。このボタンを再度クリックすると，選択したパラメータの次の出現箇所を検索できます。

リポジトリ・パラメータの追加

［リポジトリ パラメータの追加］ダイアログ・ボックスでは，新しいリポジトリ・パラメータを定義できます。パラメータの標準設定値を指定したり，パラメータがテストのステップで使用されている場合に識別に役立つ分かりやすい説明を指定することもできます。

リポジトリ・パラメータを追加するには，次の手順を実行します。

1 ［リポジトリ パラメータの管理］ダイアログ・ボックスの中で，［**リポジトリ パラメータの追加**］ボタンをクリックします。［リポジトリ パラメータの追加］ダイアログ・ボックスが表示されます。



2 ［**名前**］ボックスに，パラメータに対して分かりやすい名前を指定します。パラメータ名は英字で始める必要があり，英数字およびアンダスコアのみを使用できます。

3 ［**標準設定値**］ボックスに，リポジトリ・パラメータに使用する標準設定の値を指定できます。この値は，このオブジェクト・リポジトリを使用するテストの値またはパラメータ・タイプに，リポジトリ・パラメータを割り当てなかった場合に使用されます。標準設定の値を指定しなかった場合，リポジトリ・パラメータは，この共有オブジェクト・リポジトリを使用するテストにおいて未割り当てとして表示されます。

ヒント：標準設定の値を指定した場合，それを後で削除するには，［リポジトリ パラメータの管理］ダイアログ・ボックスの中で該当するパラメータの［**標準設定値**］セルをクリックした後，［**標準設定値をクリア**］ボタンをクリックします。セル内に**{ デフォルト値なし }**というテキストが表示されます。

---

4 ［**詳細**］ボックスに，リポジトリ・パラメータの説明を入力できます。この説明は，テストの中でリポジトリ・パラメータを割り当てるときに，パラメータの識別に役立ちます。

5 ［**OK**］をクリックして，［リポジトリ パラメータの管理］ダイアログ・ボックスのパラメータのリストにパラメータを追加します。

リポジトリ・パラメータの変更

［リポジトリ パラメータの管理］ダイアログ・ボックスでは，リポジトリ・パラメータの標準設定値または説明を直接変更できます。ただし，リポジトリ・パラメータの名前は変更できません。

リポジトリ・パラメータを変更するには，次の手順を実行します。

1 ［リポジトリ パラメータの管理］ダイアログ・ボックスの中で対象パラメータを選択します。

2 標準設定の値を変更するには，対象パラメータの［**標準設定値**］セルをクリックします。標準設定の値は，新しい値を入力して変更することも，［**標準設定値をクリア**］ボタンをクリックして削除することもできます。標準設定の値を削除すると，セル内に **{ デフォルト値なし }** というテキストが表示されます。標準設定の値を指定しない場合，リポジトリ・パラメータは，この共有オブジェクト・リポジトリを使用するテストにおいて未割り当てとして表示されます。

注 : テキストを手作業で削除しても，標準設定の値は削除されません。標準設定の値は空の文字列になります。標準設定の値を削除する場合は，［**標準設定値をクリア**］ボタンをクリックする必要があります。

---

3 パラメータの説明を変更するには，対象パラメータの［**詳細**］セルをクリックして，必要な説明を入力します。

リポジトリ・パラメータの削除

リポジトリ・パラメータの定義は，不要になった場合は削除できます。テスト・オブジェクトの定義で使用されているリポジトリ・パラメータを削除すると，パラメータがなくなったにもかかわらず，テスト・オブジェクトのプロパティ値は当該パラメータに割り当てられたままです。したがって，リポジトリ・パラメータを削除する前に，パラメータがどのテスト・オブジェクト記述にも使用されていないことを確認する必要があります。使用されていると，これらのテスト・オブジェクトを使用するステップがあるテストを実行すると失敗します。

ヒント：［リポジトリ パラメータの管理］ダイアログ・ボックスの［**リポジトリの中で検索**］ボタンを使用すれば，リポジトリ・パラメータが使用されている場所を確認できます。

リポジトリ・パラメータを削除するには，次の手順を実行します。

1 ［リポジトリ パラメータの管理］ダイアログ・ボックスの中で，パラメータ名の左側にある選択領域をクリックすることで，削除対象のリポジトリ・パラメータを選択します。

2 ［**リポジトリ パラメータを削除**］ボタンをクリックします。選択したリポジトリ・パラメータが削除されます。

テスト・オブジェクトの詳細の変更

オブジェクト・リポジトリ・マネージャで開いている共有オブジェクト・リポジトリの［**テスト オブジェクトの詳細**］領域では，実行セッション中にオブジェクトの識別に使用するプロパティおよびプロパティ値の表示や変更ができます。

共有オブジェクト・リポジトリが編集可能であり，アクティブ・ウィンドウであることを確認したら，ローカル・オブジェクトを変更するときと同じ方法で，共有オブジェクト・リポジトリ内のオブジェクトのテスト・オブジェクトの詳細を変更します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』を参照してください。

注：必要に応じて，［**編集**］＞［**元に戻す**］や［**編集**］＞［**やり直し**］オプションまたは［**元に戻す**］ボタンや［**やり直し**］ボタンを使用して，変更の取り消しや繰り返しを行うことができます。［**元に戻す**］および［**やり直し**］オプションは，アクティブなドキュメントを対象とします。リポジトリを保存すると，保存前にファイルに対して行った操作の取り消しや，やり直しはできません。

オブジェクト・リポジトリ・マネージャを使用して，共有オブジェクト・リポジトリ内のテスト・オブジェクト記述のプロパティ値を指定します。共有オブジェクト・リポジトリにあるオブジェクトのプロパティ値の指定時に使用可能なオプションと，ローカル・リポジトリにあるオブジェクトのプロパティの指定時に使用可能なオプションは異なります。共有オブジェクト・リポジトリにあるオブジェクトのプロパティ値の指定の詳細については，250 ページ「プロパティ値の指定」を参照してください。

プロパティ値の指定

テスト・オブジェクト記述に含まれるプロパティの値を指定または変更できます。定数値を使用して値を指定することも（単純な値または正規表現を含んだ定数値のいずれか），リポジトリ・パラメータを使用して値をパラメータ化することもできます。リポジトリ・パラメータの詳細については，243 ページ「リポジトリ・パラメータを使用した作業」を参照してください。

プロパティ値を指定するには，次の手順を実行します。

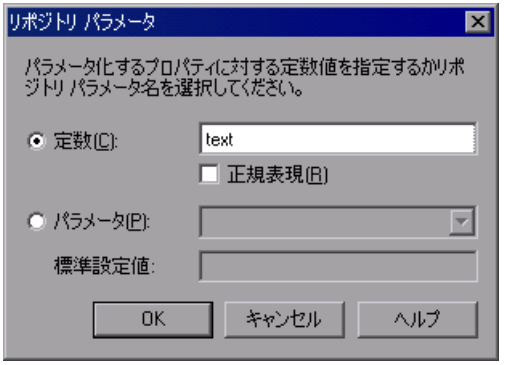
1 プロパティ値を指定する対象となるテスト・オブジェクトを選択します。

2 ［**テスト オブジェクトの詳細**］領域の中で，対象プロパティの［値］セルをクリックします。

3 次のいずれかの方法でプロパティ値を指定します。

➤ 単純な定数値を指定する場合は，［値］セルに値を入力します。［値］セルに定数値を指定した場合は，残りの手順は必要ありません。次の説明のように，［リポジトリ パラメータ］ダイアログ・ボックスの中で正規表現を使用して定数値を指定することもできます。

➤ リポジトリ・パラメータを使用して値をパラメータ化するには，［値］セルの中でパラメータ化ボタンをクリックします。［リポジトリ パラメータ］ダイアログ・ボックスが表示されます。



4 次のいずれかのオプションを選択して，プロパティの値を指定します。

➤ ［**定数**］ラジオ・ボタンを選択して，定数値を指定します。［**テスト オブジェクトの詳細**］領域の［値］セルに，定数値を直接入力することもできます。定数値の中で正規表現を使用した場合は，［**正規表現**］チェック・ボックスを選択します。

➤ ［**パラメータ**］ラジオ・ボタンを選択して，定義済みのリポジトリ・パラメータのリストからパラメータを選択します。パラメータに対して標準設定の値が定義されていれば，それも表示されます。

注：リポジトリ・パラメータを定義するには，［リポジトリ パラメータの管理］ダイアログ・ボックスを使用します。詳細については，244 ページ「リポジトリ・パラメータの管理」を参照してください。

---

5 ［**OK**］をクリックして［リポジトリ パラメータ］ダイアログ・ボックスを閉じます。値をパラメータ化した場合は，次に示すように，［**テスト オブジェクトの詳細**］領域の［**値**］カラムにパラメータ名がアイコンとともに表示されます。それ以外の場合には，指定した定数値が［**値**］カラムに表示されます。

オブジェクトの検索

オブジェクト・リポジトリ内にある特定のオブジェクトを検索するには，複数の方法があります。オブジェクトは種類に基づいて検索できます。たとえば，特定のエディット・ボックスを検索したり，アプリケーション内のオブジェクトをポイントすることで，リポジトリ内の該当するオブジェクトを自動的に強調表示したりできます。特定のプロパティ値をほかのプロパティ値に置換できます。たとえば，プロパティ値 userName を user name という値に置換できます。また，オブジェクト・リポジトリ内のオブジェクトを選択して，アプリケーション内で強調表示させることで，それがどのオブジェクトであるかを確認できます。

共有オブジェクト・リポジトリがアクティブ・ウィンドウであることを確認したら，ローカル・オブジェクト・リポジトリで行うのと同じ方法で，共有オブジェクト・リポジトリ内のオブジェクトを検索します。プロパティ値を置き換える場合は，オブジェクト・リポジトリが編集可能であることも確認する必要があります。

詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』を参照してください。

結合操作の実行

オブジェクト・リポジトリ結合ツールでは，オブジェクト・リポジトリ・マネージャの［**ローカル リポジトリから更新**］オプションを使用して（［**ツール**］＞［**ローカル リポジトリから更新**］），1 つ以上のアクションを持つローカル・オブジェクト・リポジトリのオブジェクトを共有オブジェクト・リポジトリに結合できます。たとえば，テストの特定のアクションでオブジェクトをローカルに学習していて，オブジェクト・リポジトリを使用するさまざまなテストのすべてのアクションでオブジェクトを使用できるように，それらを共有オブジェクト・リポジトリに追加することが考えられます。また，オブジェクト・リポジトリ結合ツールを使用して，2 つの共有オブジェクト・リポジトリを 1 つにすることもできます。

オブジェクト・リポジトリ結合ツールを開くには，オブジェクト・リポジトリ・マネージャで［**ツール**］＞［**オブジェクト リポジトリ結合ツール**］を選択します。結合操作の実行と，ローカル・オブジェクトを使用したオブジェクト・リポジトリの更新の詳細については，第 8 章「オブジェクト・リポジトリの管理」を参照してください。

注 : オブジェクト・リポジトリ結合ツールを開いている間は，オブジェクト・リポジトリ・マネージャを使用した作業は行えません。

インポートおよびエクスポート操作の実行

オブジェクト・リポジトリは，XML ファイルに対してインポートおよびエクスポートができます。XML は構造化された利用しやすい形式あるため，任意の XML エディタを使用してオブジェクト・リポジトリに変更を加えた後，それらを QuickTest にインポートし直すことができます。オブジェクト・リポジトリに必要な形式を知るには，保存したオブジェクト・リポジトリをエクスポートします。

ファイルのインポートおよびエクスポートは，ファイル・システムまたは Quality Center プロジェクト（QuickTest が Quality Center に接続されている場合）のどちらに対しても行うことができます。

Quality Center プロジェクトに接続するには，QuickTest またはオブジェクト・リポジトリ・マネージャから，［**ファイル**］＞［**Quality Center への接続**］を選択するか，［**Quality Center への接続**］ボタンをクリックします。Quality Center への接続の詳細については，350 ページ「Quality Center への QuickTest の接続」を参照してください。

XML からのインポート

必要な形式を使用して作成された XML ファイルをオブジェクト・リポジトリとしてインポートできます。XML ファイルは，オブジェクト・リポジトリ・マネージャを使用して XML 形式にエクスポートしたオブジェクト・リポジトリ，または QuickTest Siebel Test Express や独自作成のユーティリティなどのツールを使用して作成した XML ファイルです。XML の構造および形式に準拠する必要があります。

ヒント：必要な XML の構造および形式を確認するには，既存の共有オブジェクト・リポジトリを XML ファイルにエクスポートし，参考にします。詳細については，255 ページ「XML へのエクスポート」を参照してください。

XML ファイルからインポートするには，次の手順を実行します。

1 ［**ファイル**］ > ［**XML からインポート**］を選択します。［XML からインポート］ダイアログ・ボックスが開きます。

注：Quality Center に接続している場合，表示されるダイアログ・ボックスは，ファイル・システムの場合の標準ダイアログ・ボックスと異なります。［XML からインポート］ダイアログ・ボックスで［**ファイル システム**］ボタンおよび［**Quality Center**］ボタンをクリックすることで，ダイアログ・ボックスの切り替えができます。

---

2 インポートする XML ファイルを選択し，［**開く**］または［**OK**］をクリックします（ファイル・システムまたは Quality Center プロジェクトのどちらから開くかによります）。

3 XML ファイルがインポートされると，サマリ・メッセージ・ボックスが開き，指定のファイルから正常にインポートされたオブジェクト数，パラメータ数，およびメタデータの数に関する情報が表示されます。

4 ［**OK**］をクリックし，メッセージ・ボックスを閉じます。インポートされた XML ファイルが新しいオブジェクト・リポジトリとして開きます。これで，必要に応じて変更を加えたり，オブジェクト・リポジトリとして保存したりできます。

XML へのエクスポート

オブジェクト・リポジトリの内容を XML ファイルにエクスポートできます。これにより，オブジェクト・リポジトリの内容を任意の XML エディタを使用して簡単に編集したり，アクセス可能な柔軟な形式で保存したりできます。

XML ファイルにエクスポートするには，次の手順を実行します。

1 エクスポートするオブジェクト・リポジトリがアクティブ・ウィンドウであることを確認します。

2 ［**ファイル**］＞［**XML へエクスポート**］を選択します。［XML へエクスポート］ダイアログ・ボックスが開きます。

注：Quality Center に接続している場合，表示されるダイアログ・ボックスは，ファイル・システムの場合の標準ダイアログ・ボックスと異なります。［XML へエクスポート］ダイアログ・ボックスで［**ファイル システム**］ボタンおよび［**Quality Center**］ボタンをクリックすることで，ダイアログ・ボックスの切り替えができます。

---

3 ファイルを保存する場所を選択してファイル名または添付名を指定し，［**保存**］または［**OK**］をクリックします（ファイル・システムまたは Quality Center プロジェクトのどちらに保存するかによります）。

4 オブジェクト・リポジトリが指定の XML ファイルにエクスポートされると，サマリ・メッセージ・ボックスが開き，指定のファイルへ正常にエクスポートされたオブジェクト，パラメータ，およびメタデータの数に関する情報が表示されます。

5 ［**OK**］をクリックし，メッセージ・ボックスを閉じます。これで XML ファイルを開き，任意の XML エディタを使用して表示または変更できます。

第 9 章

共有オブジェクト・リポジトリの結合

QuickTest Professional では，オブジェクト・リポジトリ結合ツールを使用して，2 つの共有オブジェクト・リポジトリを結合して単独の共有オブジェクト・リポジトリにすることができます。また，このツールを使用して，1 つ以上のアクションのローカル・オブジェクト・リポジトリのオブジェクトを，共有オブジェクト・リポジトリに結合することもできます。

本章では，次の内容について説明します。

- 共有オブジェクト・リポジトリの結合について
- オブジェクト・リポジトリ結合ツールについて
- オブジェクト・リポジトリ結合ツールのコマンドの使用方法
- 標準設定の定義
- 2 つのオブジェクト・リポジトリの結合
- ローカル・オブジェクト・リポジトリからの共有オブジェクト・リポジトリの更新
- 結合の統計情報の表示
- オブジェクトの矛盾について
- オブジェクトの矛盾の解決
- ターゲット・リポジトリ表示枠に対するフィルタの設定
- オブジェクト・リポジトリ・ビューの同期
- 特定のオブジェクトの検索
- ターゲット・リポジトリの保存

共有オブジェクト・リポジトリの結合について

QuickTest Professional では，オブジェクト・リポジトリ結合ツールを使用して，2 つの共有オブジェクト・リポジトリの既存の資産を結合して単独の共有オブジェクト・リポジトリにすることができます。このツールを使用することで，2 つの共有オブジェクト・リポジトリ・ファイル（それぞれ「**一次**」リポジトリおよび「**二次**」リポジトリと呼びます）を結合して 3 つ目の新しいリポジトリ（「**ターゲット**」リポジトリと呼びます）を作成できます。一次リポジトリと二次リポジトリにあるオブジェクトは自動的に比較され，オブジェクト間の矛盾の解決方法を定義した事前設定可能なルールに従って，ターゲット・リポジトリに追加されます。

結合処理の後，一次リポジトリと二次リポジトリにある元のオブジェクト（これらは変更されずに残っています）のほか，結合後のターゲット・オブジェクト・リポジトリにあるオブジェクトが，オブジェクト・リポジトリ結合ツールに視覚的に表示されます。矛盾のあったオブジェクトは強調表示されます。ターゲット・オブジェクト・リポジトリでオブジェクトを選択すると，その矛盾の詳しい説明が表示されます。オブジェクト・リポジトリ結合ツールには，矛盾ごとに，提示されている解決方法を維持するか，矛盾の解決方法を個別に変更するかを，必要に応じて決めることができる専用のオプションが用意されています。

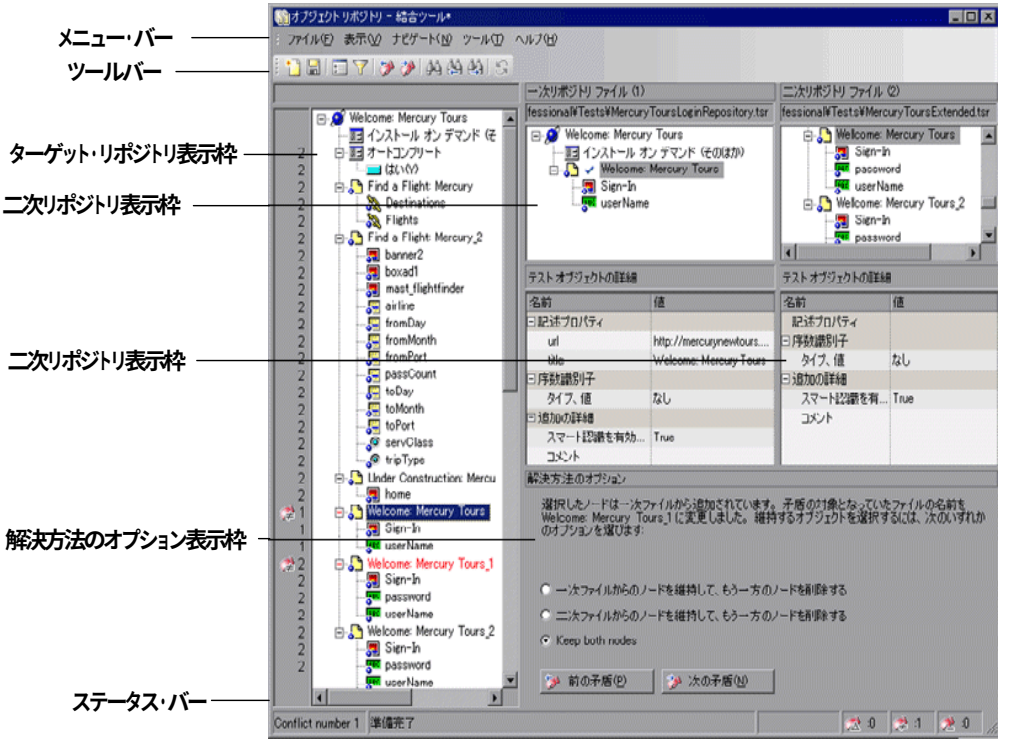
オブジェクト・リポジトリ結合ツールではまた，1 つ以上のアクションのローカル・オブジェクト・リポジトリのオブジェクトを，共有オブジェクト・リポジトリに結合することもできます。たとえば，テスト内の特定のアクションの中でオブジェクトをローカルに学習した場合に，それらを共有オブジェクト・リポジトリに追加することで，そのオブジェクト・リポジトリを使用する他のテストのすべてのアクションでそれらのオブジェクトを使用できるようになります。

注：オブジェクト・リポジトリ結合ツールが開いている間は，オブジェクト・リポジトリ・マネージャを操作することはできません。オブジェクト・リポジトリ・マネージャの詳細については，第 8 章「オブジェクト・リポジトリの管理」を参照してください。

オブジェクト・リポジトリ結合ツールについて

オブジェクト・リポジトリ結合ツールを開くには，オブジェクト・リポジトリ・マネージャで［**ツール**］＞［**オブジェクト リポジトリ結合ツール**］を選択します。

［オブジェクト リポジトリ - 結合ツール］ウィンドウの例を次に示します。



［結合ツール］ウィンドウには，次の主要な要素があります。

- **メニュー・バー**：オブジェクト・リポジトリ結合ツールのコマンドのメニューが表示されます。これらのコマンドについては本章の各所で説明します。メニュー・コマンドのショートカット・キーの詳細については，265 ページ「ショートカット・キーを使用したコマンドの実行」を参照してください。

- **ツールバー**：よく使用するメニュー・コマンドのボタンがあります。オブジェクト・リポジトリの結合，管理，および保存を行うことができます。ツールバー・ボタンの詳細については，265 ページ「ツールバー・コマンドの使用方法」を参照してください。
- **ターゲット・リポジトリ表示枠**：一次リポジトリおよび二次リポジトリから結合されたテスト・オブジェクトが表示されます。ターゲット・リポジトリ表示枠で選択したテスト・オブジェクトのプロパティが表示されるターゲット リポジトリ オブジェクトのプロパティ表示枠は，表示と非表示を切り替えることもできます。詳細については，261 ページ「ターゲット・リポジトリ表示枠」を参照してください。
- **一次リポジトリ表示枠**：一次オブジェクト・リポジトリにあるテスト・オブジェクトが表示されます。詳細については，262 ページ「一次リポジトリ表示枠および二次リポジトリ表示枠」を参照してください。
- **二次リポジトリ表示枠**：二次オブジェクト・リポジトリにあるテスト・オブジェクトが表示されます。詳細については，262 ページ「一次リポジトリ表示枠および二次リポジトリ表示枠」を参照してください。
- **解決方法のオプション表示枠**：ターゲット・リポジトリ表示枠にあるオブジェクトについて，それらのソース，矛盾，および解決方法の詳細が表示され，矛盾があった場合に適用される解決方法を変更できます。詳細については，263 ページ「解決方法のオプション表示枠」を参照してください。
- **ステータス・バー**：ターゲット・リポジトリ表示枠の中で選択したオブジェクトのソース，矛盾，および解決方法の詳細のほか，アイコンの凡例が表示されます。詳細については，263 ページ「ステータス・バー」を参照してください。

ビューの変更

オブジェクト・リポジトリ結合ツールに表示されるビューを自分が作業しやすいように変更することができます。

- オブジェクト・リポジトリ結合ツール・ウィンドウ内で表示枠のサイズを変更するには，表示枠の縁をドラッグします。
- 結合ツール内でこれらの表示枠の表示と非表示を切り替えるには，［**表示**］メニューから［**一次リポジトリ**］，［**二次リポジトリ**］，［**ターゲット リポジトリ オブジェクトのプロパティ**］，または［**解決方法のオプション**］を選択します。

➤ 現在のビューを，オブジェクト・リポジトリ結合ツールを開くたびに表示される標準設定のビューとして設定するには，［**表示**］＞［**標準のレイアウトとして設定**］を選択します。変更を加えた後に画面を標準設定に戻すには，［**表示**］＞［**標準のレイアウトを復元**］を選択します。

ターゲット・リポジトリ表示枠

ターゲット・リポジトリ表示枠には，一次および二次リポジトリから結合されたテスト・オブジェクトの階層と，それらのオブジェクトのプロパティと値が表示されます。オブジェクト階層の左側のカラムには，各オブジェクトのソース・ファイルが表示され（一次ファイルの場合は **1** と表示され，二次ファイルの場合は **2** と表示されます），矛盾がある場合は矛盾のタイプを表すアイコンが表示されます。

ターゲット・オブジェクト・リポジトリを保存すると，そのファイル・パスがオブジェクト階層の上に表示されます。

注：オブジェクトのステータスをひと目で確認できるように，ターゲット・オブジェクト・リポジトリ内のオブジェクト名のテキストの色を，各オブジェクトのソースと，矛盾の原因となったかどうかに基づいて，設定することができます。詳細については，267 ページ「色の設定の指定」を参照してください。

ターゲット・リポジトリ表示枠には次の機能があります。

➤ ターゲット・オブジェクト・リポジトリ内のオブジェクトを選択すると，一次ソース・ファイル階層または二次ソース・ファイル階層，あるいはその両方にある対応するオブジェクトが探し出され，チェック・マークによって示されます。

➤ ターゲット・リポジトリ内のオブジェクトを選択すると，そのプロパティと値が，ターゲット・リポジトリ表示枠（［**表示**］＞［**ターゲット リポジトリ オブジェクトのプロパティ**］）の一番下にある［**オブジェクト プロパティ ‐ ターゲット ファイル**］領域に表示されます。

- 結合の結果として矛盾が生じた場合は，ターゲット・オブジェクト・リポジトリ内で，矛盾のあるオブジェクトの左側にアイコンが表示されます。アイコンの上にポインタを置くと，矛盾のタイプを説明するツールチップが表示されます。
- オブジェクトを右クリックすると，ショートカット・メニューが開きます。ここから選択できるオプションでは，ターゲット・オブジェクト・リポジトリの階層全体を展開または折りたたんだり，該当する場合には矛盾の解決方法とその結果を変更したりできます。
- ノードをダブルクリックすると，ノードの階層を展開または折りたたむことができます。また，［**表示**］メニューの［**すべて折りたたみ**］または［**すべて開く**］を選択しても，ターゲット・オブジェクト・リポジトリの階層全体を展開または折りたたむことができます。

- ［**ナビゲート**］メニューの［**次の矛盾**］または［**前の矛盾**］を選択するか，ツールバーあるいは解決方法のオプション表示枠の［**次の矛盾**］または［**前の矛盾**］ボタンをクリックすると，ターゲット・オブジェクト・リポジトリ階層内の次の矛盾または前の矛盾に直接移動できます。
- ［検索］ダイアログ・ボックスを使用して，ターゲット・オブジェクト・リポジトリ内で 1 つ以上のオブジェクトを検索できます。詳細については，287 ページ「特定のオブジェクトの検索」を参照してください。
- ［**表示**］＞［**ターゲット リポジトリ オブジェクトのプロパティ**］を選択すると，ターゲット・リポジトリにあるオブジェクトのプロパティの表示と非表示を切り替えることができます。

一次リポジトリ表示枠および二次リポジトリ表示枠

一次リポジトリ表示枠および二次リポジトリ表示枠には，結合を行う元のソース・リポジトリにあるテスト・オブジェクトと，それらのプロパティおよび値が，階層表示されます。各オブジェクト階層の上にはファイル・パスが表示されます。

この表示枠には次の機能があります。

- 選択した項目をダブルクリックすると，その項目の階層を展開または折りたたむことができます。
- テスト・オブジェクトを該当する表示枠の中で選択すると，そのオブジェクトのプロパティと値が［**テスト オブジェクトの詳細**］領域に表示されます。

- ［**表示**］メニューの［**一次リポジトリ**］または［**二次リポジトリ**］を選択すると，表示枠の表示と非表示を切り替えることができます。

解決方法のオプション表示枠

解決方法のオプション表示枠には，ターゲット・オブジェクト・リポジトリで選択されているオブジェクトについて，結合中に発生した矛盾に関する情報が表示されます。また，標準設定の解決方法のオプションを使用して適用された矛盾の解決方法について，それを維持するか変更するかを決めるオプションもあります。

解決方法のオプション表示枠には次の機能があります。

- ターゲット・オブジェクト・リポジトリの中で矛盾のあるオブジェクトを選択すると，矛盾を説明するテキストと，オブジェクト・リポジトリ結合ツールによって採用される解決方法が，表示枠に表示されます。採用されている解決方法の代わりとなる他の解決方法の選択肢が用意されています。
- ラジオ・ボタンを選択することで，代わりとなる矛盾の解決方法を選択できます。変更を加えるたびに，ターゲット・オブジェクト・リポジトリが自動的に更新され，再表示されます。
- ［**次の矛盾**］ボタンまたは［**前の矛盾**］ボタンをクリックすることで，ターゲット・リポジトリ階層内の次の矛盾または前の矛盾に直接移動できます。
- ［**表示**］メニューの［**解決方法のオプション**］を選択またはクリアすることで，表示枠の表示と非表示を切り替えることができます。

ステータス・バー

ステータス・バーには，ターゲット・リポジトリ表示枠の中で選択されているオブジェクトについて，矛盾（存在する場合）の数と，ターゲット・リポジトリ表示枠で使用されているアイコンの凡例が表示されます。

次のアイコンがステータス・バー（およびターゲット・リポジトリ表示枠）に表示される場合があります。


- 類似記述の矛盾


- 同じ名前で記述が異なる矛盾


- 同じ記述で名前が異なる矛盾

アイコンの上にポインタを置くと，矛盾のタイプを説明するツールチップが表示されます。

矛盾のタイプの詳細については，280 ページ「オブジェクトの矛盾について」を参照してください。

ヒント：

ステータス・バーの矛盾アイコンをクリックすると，［統計情報］ダイアログ・ボックスが表示されます。詳細については，279 ページ「結合の統計情報の表示」を参照してください。

アイコンの左側にあるボックスをクリックすると，［フィルタ］ダイアログ・ボックスが表示されます。この領域には，フィルタが現在使用されている場合にフィルタ・アイコンが表示されます。詳細については，285 ページ「ターゲット・リポジトリ表示枠に対するフィルタの設定」を参照してください。

オブジェクト・リポジトリ結合ツールのコマンドの使用方法

オブジェクト・リポジトリ結合ツールのコマンドは，メニュー・バーまたはツールバーから選択できます。コマンドによってはショートカット・キーを押して実行できるものもあります。詳細については，265 ページ「ショートカット・キーを使用したコマンドの実行」を参照してください。ターゲット リポジトリ表示枠の中でオブジェクトを選択し，ショートカット（右クリック）メニューからコマンドを選択することもできます。

ツールバー・コマンドの使用方法

よく使用するコマンドは，ツールバーのボタンをクリックして実行できます。

ショートカット・キーを使用したコマンドの実行

ショートカット・キーを押すことで，オブジェクト・リポジトリ結合ツールのいくつかのコマンドを実行できます。次のショートカット・キーは，対応するメニュー・コマンドの横に表示されます。

対応するショートカット・キーを押して，次の［**ファイル**］メニュー・コマンドを実行できます。

コマンド	ショートカット・キー	機能
［新規結合］	CTRL+N	新しい結合操作を実行する対象となる 2 つのオブジェクト・リポジトリを指定できます。
［保存］	CTRL+S	結合された共有オブジェクト・リポジトリを保存します。

対応するショートカット・キーを押して，次の［**ナビゲート**］メニュー・コマンドを実行できます。

コマンド	ショートカット・キー	機能
［次の矛盾］	F4 キー	結合後のオブジェクト・リポジトリの中で次の矛盾オブジェクトを検索します。
［前の矛盾］	SHIFT+F4	結合後のオブジェクト・リポジトリの中で前の矛盾オブジェクトを検索します。
［検索］	CTRL+F	［検索］ダイアログ・ボックスを開きます。
［次を検索］	F3 キー	［検索］ダイアログ・ボックスの検索条件に従って，結合後のオブジェクト・リポジトリの中で次のオブジェクトを検索します。
［前を検索］	SHIFT+F3	［検索］ダイアログ・ボックスの検索条件に従って，結合後のオブジェクト・リポジトリの中で前のオブジェクトを検索します。

標準設定の定義

オブジェクト・リポジトリ結合ツールでは，オブジェクト・リポジトリの結合時に使用される設定があらかじめ定義されています。標準設定は次のとおりです。

- ターゲット・オブジェクト・リポジトリに表示されるオブジェクト名のテキストの色を指定します。
- オブジェクト・リポジトリ結合ツールが一次および二次リポジトリ内のオブジェクトの矛盾をどのように処理するのかを設定します。あるいは，ローカル・オブジェクト・リポジトリからの共有オブジェクト・リポジトリを更新するときに，ローカルおよび共有リポジトリ内のオブジェクトの矛盾をどのように処理するのかを設定します。

これらの設定をいつでも変更して，新しい標準設定を作成できます。設定を変更すると，以降のすべての新しい結合が新しい標準設定に従って実行されます。

ヒント：2 つのリポジトリを結合する前に設定を変更するには，［**キャンセル**］をクリックして［新規結合］ダイアログ・ボックスを閉じ，以降の各項の説明に従って設定を変更した後，結合を実行する必要があります。

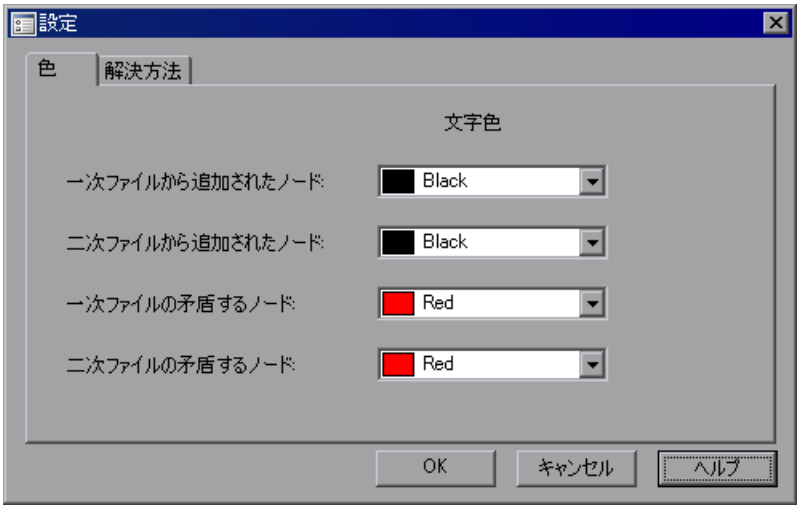
色の設定の指定

ターゲット・オブジェクト・リポジトリに表示されるオブジェクト名の色を，オブジェクトのソースと，矛盾の原因となったかどうかに基づいて指定できます。これを利用することで，各オブジェクトのステータスを容易に判断できるようになります。

注：［設定］ダイアログ・ボックスの［色］タブにあるオプションは，［**ローカル リポジトリから更新**］操作を実行するときに，ローカル（一次）オブジェクト・リポジトリおよび共有（二次）オブジェクト・リポジトリから追加されたオブジェクトにも適用されます。

色の設定を指定するには，次の手順を実行します。

1 ［**ツール**］＞［**設定**］を選択するか，［**設定**］ボタンをクリックします。［設定］ダイアログ・ボックスが開きます。



2 ［色］タブにある各項目について，テキスト・ボックスの横にある下矢印 をクリックし，識別用の色を選択します。

3 ［**OK**］をクリックします。ターゲット・オブジェクト・リポジトリ内のオブジェクト名が選択した色で表示されます。

標準の解決方法の設定

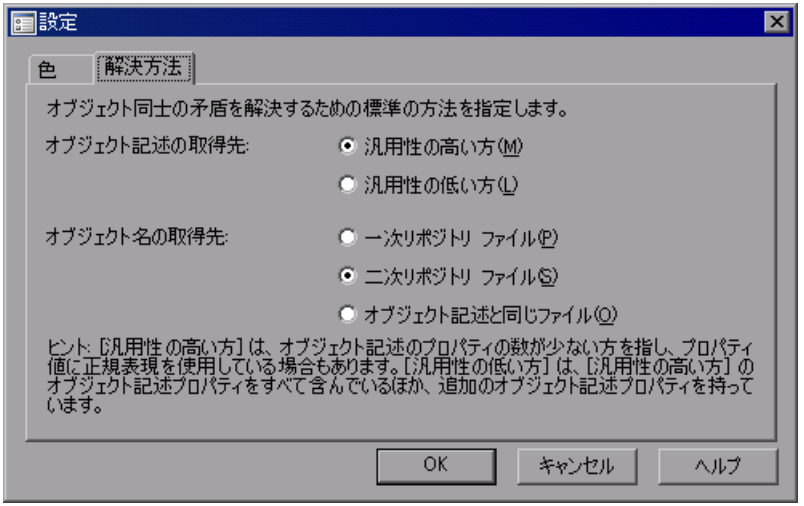
結合処理中のオブジェクトの矛盾をオブジェクト・リポジトリ結合ツールでどのように自動的に処理するかを設定できます。

注 : ［設定］ダイアログ・ボックスの［解決方法］タブにあるオプションは，［**ローカル リポジトリから更新**］操作を実行するときに，ローカル・オブジェクト・リポジトリおよび共有オブジェクト・リポジトリから追加されたオブジェクトにも適用されます。

標準の解決方法を設定するには，次の手順を実行します。

1 ［**ツール**］＞［**設定**］を選択するか，［**設定**］ボタンをクリックします。［設定］ダイアログ・ボックスが開きます。

2 ［**解決方法**］タブをクリックします。



3 矛盾のあるオブジェクトを処理するときにオブジェクト・リポジトリ結合ツールに適用させる標準の解決方法を指定する適切なラジオ・ボタンを選択します。

➤ ［**オブジェクト記述の取得先**］: 2 つのテスト・オブジェクトの名前が同じで記述が異なる場合の，矛盾の解決方法を指定します。ターゲット・オブジェクト・リポジトリにおいて，汎用性の高い方のオブジェクト記述を採用するか，汎用性の低い方のオブジェクト記述を採用するかを指定できます。

- ［**汎用性の高い方**］: 矛盾相手のオブジェクトよりも識別プロパティが少ないか，プロパティ値の中で正規表現を使用しているオブジェクトを採用するよう，オブジェクト・リポジトリ結合ツールを設定します。これが標準設定です。
- ［**汎用性の低い方**］: 矛盾相手のオブジェクトのすべての識別プロパティに加えて他の識別プロパティも持っているオブジェクトを採用するよう，オブジェクト・リポジトリ結合ツールを設定します。

➤［**オブジェクト名の取得先**］: 2 つのテスト・オブジェクトの記述が同一または類似しながらも名前が異なる場合の矛盾を解決する方法を指定します。ターゲット・オブジェクト・リポジトリにおいて採用するオブジェクト名の取得先となるソースを選択できます。

- ［**一次リポジトリ ファイル**］: 一次オブジェクト・リポジトリ内のオブジェクトのオブジェクト名を，ターゲット・オブジェクト・リポジトリで採用します。これが標準設定です。
- ［**二次リポジトリ ファイル**］: 二次オブジェクト・リポジトリ内のオブジェクトのオブジェクト名を，ターゲット・オブジェクト・リポジトリで採用します。
- ［**オブジェクト記述と同じファイル**］: オブジェクト記述の取得先と同じオブジェクト・リポジトリ内のオブジェクトのオブジェクト名を，ターゲット・オブジェクト・リポジトリで採用します。

4 ［**OK**］をクリックします。以降，オブジェクト・リポジトリ結合ツールで実行するリポジトリの結合においてオブジェクト間の矛盾を解決する際に，ここで選択した方法が適用されます。

注 : 結合後のオブジェクト・リポジトリを開いたままの状態で解決方法の設定に変更を加えた場合は，開いているファイルについて新しい設定でもう一度結合するかどうか尋ねられます。新しい設定でもう一度ファイルを結合する場合は，［**はい**］をクリックします。以前の設定で作成した既存の結合を維持する場合は，［**いいえ**］をクリックします。［**いいえ**］をクリックした場合，新しい設定は以降の結合にのみ適用されます。

---

2 つのオブジェクト・リポジトリの結合

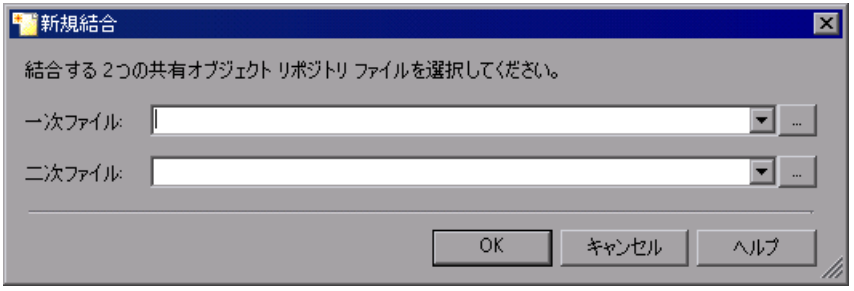
オブジェクト・リポジトリ結合ツールを使用すると，2 つのソース・オブジェクト・リポジトリを結合して新しい共有オブジェクト・リポジトリを作成できます。リポジトリにあるオブジェクトは自動的に比較され，オブジェクト間の矛盾の解決方法を定義する設定可能なルールに従って，新しいリポジトリに追加されます。元のソース・ファイルは変更されません。

注：別のユーザが開いているオブジェクト・リポジトリはロックされます。ロックされたファイルを統合しようとした場合，警告メッセージが表示されますが，結合プロセスはソース・ファイルを変更しないため，結合を実行することは可能です。ただし，ロックされたファイルに対して他のユーザによって加えられた変更が，結合後のオブジェクト・リポジトリに含まれない可能性があります。

---

2 つのオブジェクト・リポジトリを結合するには，次の手順を実行します。

1 オブジェクト・リポジトリ・マネージャの中で，［**ツール**］＞［**オブジェクト リポジトリ結合ツール**］を選択します。［オブジェクト リポジトリ - 結合ツール］ウィンドウの手前に，［新規結合］ダイアログ・ボックスが開きます。



ヒント：

［オブジェクト リポジトリ - 結合ツール］ウィンドウがすでに開いている場合は，［**ファイル**］＞［**新規結合**］を選択するか，［**新規結合**］ボタンをクリックすることで，［新規結合］ダイアログ・ボックスを開けます。

リポジトリを結合する前に設定済みの設定を変更するには，［**キャンセル**］をクリックして［新規結合］ダイアログ・ボックスを閉じ，266 ページ「標準設定の定義」の説明に従って設定を変更した後，結合を実行する必要があります。

---

2 ［**一次ファイル**］および［**二次ファイル**］ボックスで，結合後の単独のリポジトリとなる **.tsr** オブジェクト・リポジトリの名前を入力するか，または参照して選択します。各ボックスの横にある下矢印 をクリックすると，最近使用したファイルを表示および選択できます。

注：

一次リポジトリには，最も作業内容の多いオブジェクト・リポジトリ，つまり，より多くのオブジェクト，オブジェクト・プロパティ，およびオブジェクト値を持つリポジトリを選択することをお勧めします。

拡張子が **.tsr** でないファイル，パスが正しくない **.tsr** ファイル，または存在しないファイルを入力した場合は，対応するテキスト・ボックスの横に警告アイコンが表示されます。アイコンの上にポインタを置くと，エラーを説明するツールチップが表示されます。正しいパスを持つ既存の **.tsr** ファイルを入力または選択してください。

以前のバージョンの QuickTest を使用して作成されたオブジェクト・リポジトリを結合する場合は，はじめにオブジェクト・リポジトリ・マネージャでそのオブジェクト・リポジトリを開いてから保存して新しい形式に更新する必要があります。

---

3 ［**OK**］をクリックします。オブジェクト・リポジトリ結合ツールによって，設定されている解決方法の設定に従い，選択したオブジェクト・リポジトリが新しいターゲット・オブジェクト・リポジトリに自動的に結合されます。結合の結果は，［オブジェクト リポジトリ - 結合ツール］ウィンドウの手前に表示される［統計情報］ダイアログ・ボックスに表示されます。

4 279 ページ「結合の統計情報の表示」の説明を参考にして，結合に関する統計情報を確認し，［**閉じる**］をクリックします。

［オブジェクト リポジトリ - 結合ツール］ウィンドウでは，次を実行できます。

- ソース・リポジトリのオブジェクト間の矛盾を解決する方法を必要に応じて変更できます。詳細については，283 ページ「オブジェクトの矛盾の解決」を参照してください。
- ターゲット・オブジェクト・リポジトリ内のオブジェクトにフィルタを適用できます。詳細については，285 ページ「ターゲット・リポジトリ表示枠に対するフィルタの設定」を参照してください。
- ターゲット・オブジェクト・リポジトリ内で特定のオブジェクトを検索できます。詳細については，286 ページ「オブジェクト・リポジトリ・ビューの同期」を参照してください。
- ターゲット・オブジェクト・リポジトリをファイル・システムまたは Quality Center プロジェクトに保存できます。詳細については，289 ページ「ターゲット・リポジトリの保存」を参照してください。

ローカル・オブジェクト・リポジトリからの共有オブジェクト・リポジトリの更新

1 つ以上のテストにあるアクションに関連付けられているローカル・オブジェクト・リポジトリを，共有オブジェクト・リポジトリに結合することによって，共有オブジェクト・リポジトリを更新することができます。更新後，ローカル・オブジェクト・リポジトリから結合されたオブジェクトは，任意のテストの中で当該共有オブジェクト・リポジトリを使用する任意のアクションから利用できるようになります。

結合処理では，選択したアクションに対応するローカル・オブジェクト・リポジトリ内のオブジェクトが，ターゲットの共有オブジェクト・リポジトリに移動します。そして，当該アクションで，更新後の共有オブジェクト・リポジトリのオブジェクトが使用されるようになります。

複数のアクション用のローカル・オブジェクト・リポジトリを追加することを選択した場合は，QuickTest によって複数の結合が実行され，各アクションのローカル・オブジェクト・リポジトリがリスト内のすべてのアクション に対して一度に 1 つずつターゲット・オブジェクト・リポジトリに結合されます。必要ならば，各結合の結果を表示および変更できます。

注：結合できるのは，更新対象の共有オブジェクト・リポジトリに関連付けられているアクションのローカル・オブジェクト・リポジトリのみです。

ローカル・オブジェクト・リポジトリから共有オブジェクト・リポジトリを更新するには，次の手順を実行します。

1 ［**リソース**］＞［**オブジェクト リポジトリ マネージャ**］を選択します。オブジェクト・リポジトリ・マネージャが開きます。

注：オブジェクト・リポジトリ・マネージャの詳細については，第 8 章「オブジェクト・リポジトリの管理」を参照してください。

2 オブジェクト・リポジトリ・マネージャの中で，［**ファイル**］＞［**開く**］を選択するか，［**開く**］ボタンをクリックします。［共有オブジェクト リポジトリを開く］ダイアログ・ボックスが表示されます。

現在 Quality Center プロジェクトに接続している場合は，［共有オブジェクト リポジトリを開く］ダイアログ・ボックスにプロジェクトのテスト計画ツリーが表示されます。テストを選択すると，テストに添付されている共有オブジェクト・リポジトリが表示されます。

3 更新対象の共有オブジェクト・リポジトリが格納されている **.tsr** ファイルを参照し，［**読み取り専用モードで開く**］チェック・ボックスの選択を解除して，［**開く**］をクリックします。または，Quality Center の添付ファイルの場合は［**OK**］をクリックします。ファイルが開き，オブジェクトとプロパティが編集可能な形式で表示されます。

ヒント：オブジェクト・リポジトリを読み取り専用モードで開いた場合は，［**ファイル**］＞［**編集を有効化**］を選択するか，オブジェクト・リポジトリ・マネージャのツールバーの［**編集を有効化**］ボタンをクリックします。オブジェクト・リポジトリ・ファイルが編集可能になります。

---

4 ［**ツール**］＞［**ローカル リポジトリから更新**］を選択します。［ローカル リポジトリから更新］ダイアログ・ボックスが開きます。

5 ［**テストを追加します**］ボタンの横にある下矢印 をクリックして，［**テストを参照**］を選択します。［テストを開く］ダイアログ・ボックスが開きます。現在 Quality Center プロジェクトに接続している場合は，［Quality Center プロジェクトからテストを開く］ダイアログ・ボックスが開きます。

対象アクションが含まれているテストを参照します。このアクションのローカル・オブジェクト・リポジトリが共有オブジェクト・リポジトリに結合されます。

注：追加可能なテストは，更新対象の共有オブジェクト・リポジトリに関連付けられているアクションが含まれていて，そのアクションのローカル・オブジェクト・リポジトリにオブジェクトが含まれているテストのみです。

---

6 必要ならば，手順 5 を繰り返してテストをさらに追加します。

注：一覧表示されているテストに含まれているすべてのアクションに関連付けられているローカル・オブジェクト・リポジトリが結合に含まれます。特定のアクションを結合から除外するには，一覧の中でそれを選択して［**削除**］をクリックします。

---

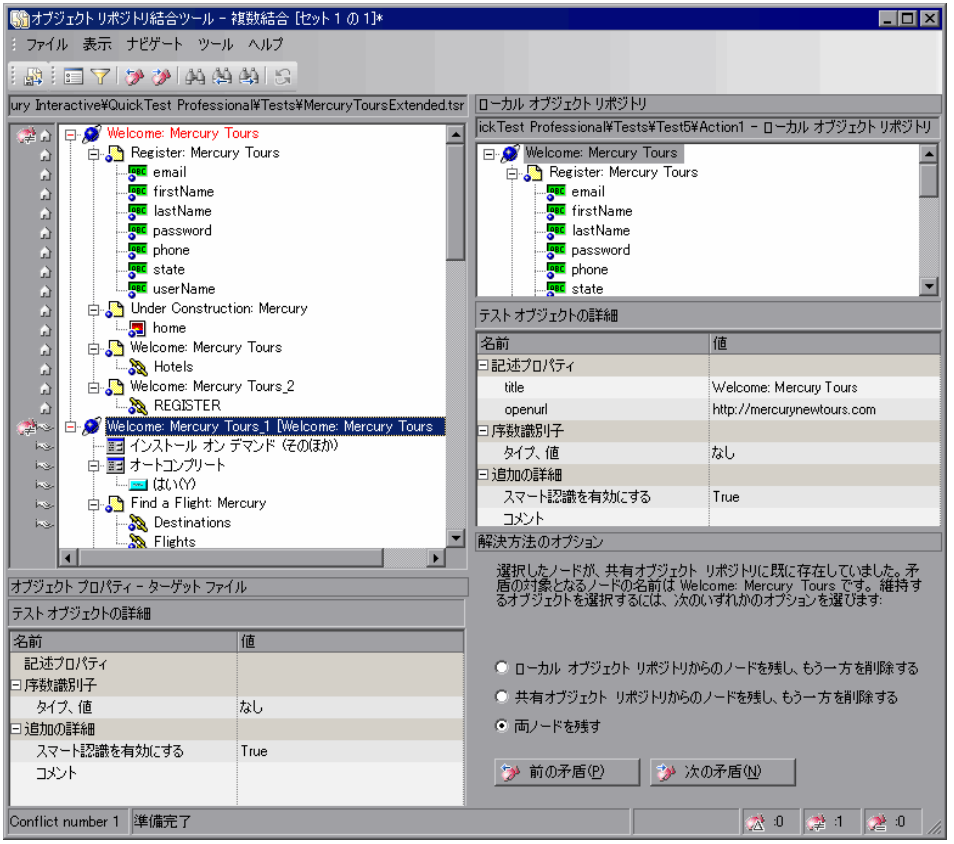
7 ［**すべて更新**］をクリックします。QuickTest によって，設定済みの設定に従い，最初のアクションのローカル・オブジェクト・リポジトリが共有オブジェクト・リポジトリに自動的に結合されます。結合の結果は，［オブジェクト リポジトリ - 結合ツール］ウィンドウの手前に表示される［統計情報］ダイアログ・ボックスに表示されます。

注：各結合の前に，別のユーザがローカル・オブジェクト・リポジトリを使用していないかどうかが QuickTest によって確認されます。別のユーザが使用している場合，そのローカル・オブジェクト・リポジトリはロックされているので，選択したアクションに対応したオブジェクトをターゲットの共有オブジェクト・リポジトリに移動することはできません。警告メッセージが表示されます。他のユーザがローカル・オブジェクト・リポジトリの使用を止めれば，結合を実行できます。

---

8 279 ページ「結合の統計情報の表示」の説明を参考にして，結合に関する統計情報を確認し，［**閉じる**］をクリックします。

ローカル・オブジェクト・リポジトリの結合の場合，［オブジェクト リポジトリ - 結合ツール］ウィンドウには，ローカル・オブジェクト・リポジトリが一次オブジェクト・リポジトリとして表示され，共有オブジェクト・リポジトリがターゲット・オブジェクト・リポジトリとして表示されます。



ターゲット・オブジェクト階層内の各オブジェクトの左側には，オブジェクトのソースを示すアイコンが表示されます。

は，ノードがローカル・オブジェクト・リポジトリから追加されたことを示します。

は，ノードがすでに共有オブジェクト・リポジトリに存在することを示します。

注 : [ローカル リポジトリから更新] ダイアログ・ボックスで複数のアクションを指定した場合は，QuickTest によって複数の結合が実行され，各アクションのローカル・オブジェクト・リポジトリが一度に 1 つずつターゲット・オブジェクト・リポジトリに結合されます。この手順の後に表示される [統計情報] ダイアログ・ボックスと [オブジェクト リポジトリ結合ツール - 複数結合] ウィンドウには，最初の結合の結果が表示されます（最初のアクションのローカル・オブジェクト・リポジトリが共有オブジェクト・リポジトリに結合されます）。必要ならば，QuickTest で各結合の結果を順に表示および変更できます。複数結合では，各結合セットの番号がタイトル・バーに表示されます。たとえば，「**[セット 3 の 2]**」などと表示されます。

---

9 共有オブジェクト・リポジトリに結合された各オブジェクトに対しては，自動結合を選択するか，解決方法のオプション表示枠を使用して次のことを行うことができます。

- 特定のオブジェクトを共有オブジェクト・リポジトリに追加し，ローカル・オブジェクト・リポジトリから削除する。
- 特定のオブジェクトをローカル・オブジェクト・リポジトリに維持し，共有オブジェクト・リポジトリには追加しない。

詳細については，283 ページ「オブジェクトの矛盾の解決」を参照してください。

10 複数の結合を実行している場合は，[オブジェクト リポジトリ - 結合ツール] のツールバーの [**保存して次を結合**] ボタンをクリックして，次の結合を実行します（次のアクションのローカル・オブジェクト・リポジトリが共有オブジェクト・リポジトリに結合されます）。

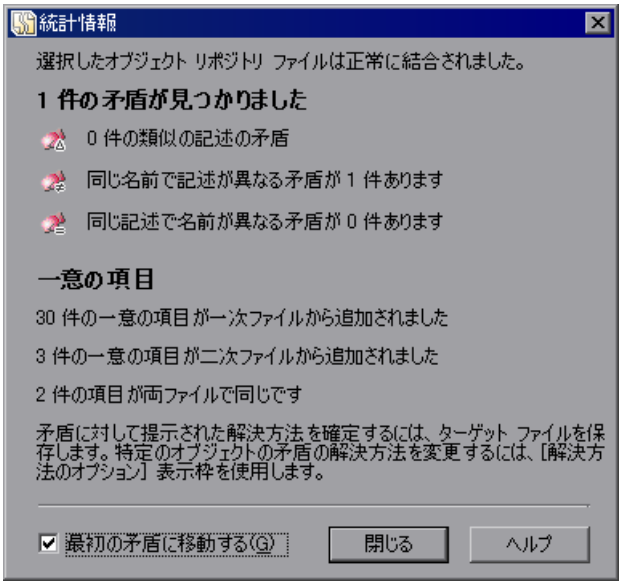
11 結合ごとに [**はい**] をクリックして変更を保存します。[**いいえ**] をクリックした場合は，現在の結合（最後のアクションから結合されたオブジェクト）は保存されません。

12 手順 8 から 11 を繰り返して複数結合を最後まで実行します。

13 [**ファイル**] > [**終了**] をクリックし，[**はい**] をクリックすると，更新されたオブジェクト・リポジトリが保存されます。

結合の統計情報の表示

2 つのオブジェクト・リポジトリを結合した後，オブジェクト・リポジトリ結合ツールには［統計情報］ダイアログ・ボックスが表示されます。このダイアログ・ボックスには，ファイルが結合された方法と，結合中に解決されたすべての矛盾の数とタイプが表示されます。



注：［**ローカル リポジトリから更新**］操作を行った後に表示される統計情報は，前述のオプションとは若干異なります。

ヒント：［統計情報］ダイアログ・ボックスの結合に関する統計情報は，［オブジェクト リポジトリ - 結合ツール］ウィンドウで［**表示**］＞［**統計情報**］を選択するか，ステータス・バーの矛盾アイコンをクリックすることで，いつでも表示できます。

［統計情報］ダイアログ・ボックスには次の情報が表示されます。

- ターゲット・オブジェクト・リポジトリに追加されたオブジェクト間で発生した，すべての矛盾の数とタイプ。矛盾タイプの詳細については，283 ページ「オブジェクトの矛盾の解決」を参照してください。
- ターゲット・オブジェクト・リポジトリに追加された，一次ファイルまたは二次ファイル（またはローカル・ファイル）の中の一意の項目または両方のファイルでまったく同じ項目の数。

ヒント：［**最初の矛盾に移動する**］チェック・ボックスを選択すると，［統計情報］ダイアログ・ボックスを閉じた直後に，ターゲット・オブジェクト・リポジトリ内の最初の矛盾に移動します。

オブジェクトの矛盾について

2 つのオブジェクト・リポジトリを結合する際，それらに含まれているテスト・オブジェクトどうしが類似しているために，矛盾が発生することがあります。オブジェクト・リポジトリ統合ツールでは，発生する可能性のある 3 つの矛盾タイプが識別されます。

- ［**類似の記述の矛盾**］：名前とオブジェクト階層が同じで，記述がわずかに異なる，2 つのテスト・オブジェクト。この矛盾タイプでは，一方のオブジェクトが常に他方のプロパティ・セットのサブセットを持っています。これらの矛盾の詳細については，281 ページを参照してください。

  標準設定では，このタイプの矛盾に対する矛盾の解決方法の設定は，矛盾相手のオブジェクトよりも識別プロパティの少ない方のオブジェクトがターゲット・オブジェクト・リポジトリで採用されるように設定されます。標準設定の変更方法の詳細については，266 ページ「標準設定の定義」を参照してください。

- ［**同じ名前で記述が異なる矛盾**］：名前とオブジェクト階層が同じであるものの，それらの記述がどこか異なる（たとえば，プロパティが異なる，プロパティが同じでも値が異なるなど），2 つのテスト・オブジェクト。これらの矛盾の詳細については，282 ページを参照してください。

標準設定では，このタイプの矛盾に対する矛盾の解決方法の設定は，両方のファイルのオブジェクトがターゲット・オブジェクト・リポジトリで採用されるように設定されます。二次ファイルから追加されるオブジェクトの名前は変更され，1 つずつ値が大きくなる数字の接尾辞が名前に付けられます。たとえば，Edit_1 などという名前になります。標準設定の変更方法の詳細については，266 ページ「標準設定の定義」を参照してください。

➤ [**同じ記述で名前が異なる矛盾**]：記述がまったく同じで，オブジェクト階層が同じものの，オブジェクト名が異なる，2 つのテスト・オブジェクト。これらの矛盾の詳細については，282 ページを参照してください。

標準設定では，このタイプの矛盾に対する矛盾の解決方法の設定は，一次ソース・ファイルのオブジェクト名がターゲット・オブジェクト・リポジトリで採用されるように設定されます。標準設定の変更方法の詳細については，266 ページ「標準設定の定義」を参照してください。

注：記述を持たない Page オブジェクトや Browser オブジェクトなどのオブジェクトは名前のみ比較されます。両方のソース・リポジトリに同じオブジェクトが含まれていて，それらの名前が異なる場合は，2 つの別々のオブジェクトとしてターゲット・オブジェクト・リポジトリに結合されます。

類似記述の矛盾

一次オブジェクト・リポジトリ内のオブジェクトと二次オブジェクト・リポジトリ内のオブジェクトが，同じ名前を持ち，まったく同じではないけれども類似する記述プロパティおよび値を持っている場合です。一方のオブジェクトが常に他方のプロパティ・セットのサブセットを持っています。たとえば，二次オブジェクト・リポジトリにある Button_1 という名前のオブジェクトが，一次オブジェクト・リポジトリにある Button_1 という名前のオブジェクトと同じ記述プロパティおよび値を持っているものの，さらに追加のプロパティと値を持っているとします。

この矛盾タイプは次のようにして解決できます。

- 一次リポジトリから追加されるオブジェクトのオブジェクト記述を採用する。
- 二次リポジトリから追加されるオブジェクトのオブジェクト記述を採用する。
- 両方のオブジェクトをターゲット・オブジェクト・リポジトリに取り込む。この場合，ターゲット・オブジェクト結合ツールでは，二次ファイルから追加されるオブジェクトの名前が自動的に変更され，1 つずつ値が大きくなる数字の接尾辞が名前に付けられます。たとえば，Edit_1 などという名前になります。

同じ名前で記述が異なる矛盾

一次オブジェクト・リポジトリ内のオブジェクトと二次オブジェクト・リポジトリ内のオブジェクトが，同じ名前を持つものの，完全に異なる記述プロパティおよび値を持っている場合です。

この矛盾タイプは次のようにして解決できます。

- 一次リポジトリから追加されるオブジェクトのみを維持する。
- 二次リポジトリから追加されるオブジェクトのみを維持する。
- 両方のリポジトリからのオブジェクトを維持する。この場合，ターゲット・オブジェクト結合ツールでは，二次ファイルから追加されるオブジェクトの名前が自動的に変更され，1 つずつ値が大きくなる数字の接尾辞が名前に付けられます。たとえば，Edit_1 などという名前になります。

同じ記述で名前が異なる矛盾

一次オブジェクト・リポジトリ内のオブジェクトと二次オブジェクト・リポジトリ内のオブジェクトが，異なる名前を持つものの，同じ記述プロパティおよび値を持っている場合です。

この矛盾タイプは次のようにして解決できます。

- 一次リポジトリにある該当オブジェクトのオブジェクト名を採用する。
- 二次リポジトリにある該当オブジェクトのオブジェクト名を採用する。

オブジェクトの矛盾の解決

一次オブジェクト・リポジトリと二次オブジェクト・リポジトリにあるオブジェクトどうしの矛盾は，オブジェクト・リポジトリ結合ツールによって，標準として設定されている解決方法に従って自動的に解決されます。標準の解決方法は，結合を実行する前に設定できます。詳細については，266 ページ「標準設定の定義」を参照してください。

ただし，オブジェクト・リポジトリ結合ツールでは，矛盾の原因となった個々のオブジェクトごとに，結合の実行方法を変更することも可能です。

たとえば，一次リポジトリ内のオブジェクトが，二次リポジトリ内のオブジェクトと同じ名前だったものの，記述が異なったとします。このとき，標準の設定として，汎用性の高い方のオブジェクト記述を持つオブジェクト，つまり，プロパティの数の少ない方のオブジェクトをターゲット・オブジェクト・リポジトリに追加すると定義していたとしましょう。しかし，自動結合の後に矛盾を確認した結果，その特定の矛盾を別の方法（たとえば，両方のオブジェクトを維持するなど）で扱うように判断することが考えられます。

注：標準の矛盾の解決方法に変更を加えること自体が新しい矛盾の原因となり，ターゲット・オブジェクト・リポジトリに影響を与えることがあります。前述の例では，両方のオブジェクトを維持することが名前の矛盾の原因となります。したがって，矛盾の解決方法を変更するたびにターゲット・オブジェクト・リポジトリが更新され，再表示されます。

オブジェクト・リポジトリ結合ツールのターゲット・オブジェクト・リポジトリ表示枠で，オブジェクト名の左側に表示されるアイコンと，テキストの色によって，矛盾の原因となったオブジェクトと，矛盾のタイプを，識別することができます。矛盾のあるオブジェクトを選択すると，矛盾の詳細な説明が，オブジェクト・リポジトリ結合ツールによる矛盾の自動解決方法とともに，解決方法のオプション表示枠に表示されます。

解決方法のオプション表示枠には，代わりの解決方法のオプションが提示されます。標準の解決方法がニーズに合っていればそれを維持することも，代わりのオプションを使用して矛盾を別の方法で解決することも選択できます。

ヒント：また，標準の解決方法の設定を変更して，ファイルを再び結合することもできます。詳細については，266 ページ「標準設定の定義」を参照してください。

オブジェクトの矛盾を解決するには，次の手順を実行します。

1 ターゲット・オブジェクト・リポジトリで，矛盾のあるオブジェクトを選択します。矛盾のあるオブジェクトはオブジェクト名の左側のアイコンで示されます。矛盾しているオブジェクトはソース・リポジトリで強調表示されています。

矛盾の説明と，オブジェクト・リポジトリ結合ツールが使用する解決方法が，解決方法のオプション表示枠に表示されます。使用できる代替の解決方法ごとに，対応するラジオ・ボタンが表示されます。各矛盾タイプの詳細については，280 ページ「オブジェクトの矛盾について」を参照してください。

2 解決方法のオプション表示枠で，ラジオ・ボタンを選択し，代わりとなる矛盾の解決方法を選択します。選択した解決方法に従ってターゲット・オブジェクト・リポジトリが更新され，再表示されます。

3 解決方法のオプション表示枠で，［**次の矛盾**］または［**前の矛盾**］ボタンをクリックすると，ターゲット・リポジトリ階層内の次または前の矛盾に直接移動できます。

4 矛盾の解決方法をほかにも変更する必要があれば，手順 1 から 3 を繰り返します。

5 ターゲット・オブジェクト・リポジトリを保存します。詳細については，289 ページ「ターゲット・リポジトリの保存」を参照してください。

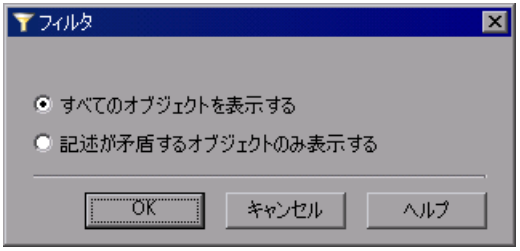
ターゲット・リポジトリ表示枠に対するフィルタの設定

2 つのオブジェクト・リポジトリを結合した結果，ターゲット・オブジェクト・リポジトリに含まれるオブジェクトの数が非常に多くなることがあります。ターゲット・リポジトリ表示枠でのナビゲーションや特定のオブジェクトの検索が容易になるように，オブジェクト・リポジトリ結合ツールでは，表示枠内のオブジェクトを絞り込み（フィルタ処理），結合中に解決された矛盾のあるオブジェクトだけを表示することができます。

注：フィルタは，ターゲット・リポジトリ表示枠にどのオブジェクトを表示するかのみを決めるものです。どのオブジェクトをターゲット・オブジェクト・リポジトリに含めるのかを決めるものではありません。

ターゲット・リポジトリ表示枠のオブジェクトを絞り込むには，次の手順を実行します。

1 ［**ツール**］＞［**フィルタ**］を選択するか，［**フィルタ**］ボタンをクリックします。［フィルタ］ダイアログ・ボックスが表示されます。



ヒント：ステータス・バー内のアイコンの左側にあるボックスをクリックして［フィルタ］ダイアログ・ボックスを表示することもできます。この領域には，フィルタが現在使用されている場合にフィルタ・アイコンが表示されます。

---

2 ターゲット・オブジェクト・リポジトリに表示するオブジェクトに対応したラジオ・ボタンを選択します。

- ［**すべてのオブジェクトを表示する**］：ターゲット・オブジェクト・リポジトリ内のすべてのオブジェクトが表示されます。
- ［**記述が矛盾するオブジェクトのみ表示する**］：ターゲット・オブジェクト・リポジトリ内のオブジェクトのうち，記述の矛盾が生じたもののみが表示されます。

3 ［**OK**］をクリックします。表示枠内のオブジェクトが絞り込まれ，設定したオブジェクト・タイプのみがターゲット・オブジェクト・リポジトリに表示されます。

オブジェクト・リポジトリ・ビューの同期

ターゲット・オブジェクト結合ツールでは，ターゲット，一次，二次の各オブジェクト・リポジトリを独立してナビゲートできます。各種の表示枠のサイズを変更して，リポジトリに格納されているオブジェクトの一部だけを表示することもできます。このため，大きなオブジェクト・リポジトリを扱っている場合に，各種の表示枠にリポジトリ階層の異なる領域が表示され，結合プロセスの影響を受ける特定のオブジェクトを探して追跡することが難しくなることがあります。

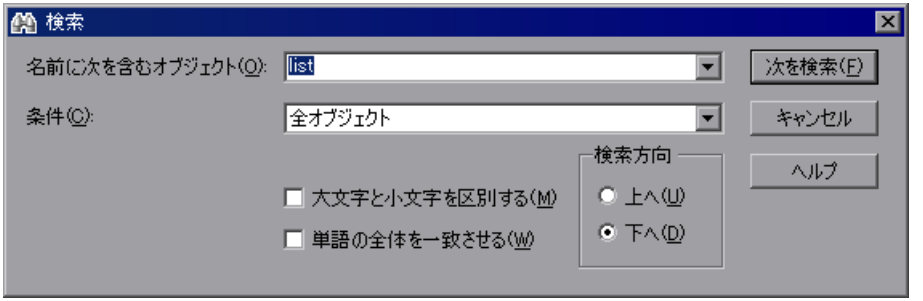
リポジトリの同期をとり，両方のビューに同じオブジェクトが表示されるようにするには，当該オブジェクトが現在表示されている一次または二次オブジェクト・リポジトリ内でそのオブジェクトを選択し，［**ビューを同期化**］をクリックします。

特定のオブジェクトの検索

オブジェクト・リポジトリ結合ツールの検索機能を使用して，名前に指定の文字列が含まれている 1 つ以上のオブジェクトをターゲット・オブジェクト・リポジトリの中で検索できます。見つかったオブジェクトは，対応する一次リポジトリまたは二次リポジトリの一方または両方でも強調表示されます。

オブジェクトを検索するには，次の手順を実行します。

1 ［**ナビゲート**］＞［**検索**］を選択するか，［**検索**］ボタンをクリックします。［検索］ダイアログ・ボックスが開きます。



2 ［**名前に次を含むオブジェクト**］ボックスに，検索するオブジェクトの名前の全体または一部を入力します。

3 ［**条件**］ボックスで，検索するオブジェクトを選択して検索対象を絞込みます。次の条件を使用できます。

- ［**全オブジェクト**］
- ［**一方のソースにのみ存在するオブジェクト**］
- ［**矛盾するオブジェクト**］
- ［**矛盾するオブジェクトまたは一方にのみ存在するオブジェクト**］

4 次のオプションの一方または両方を選択して検索の絞り込みに役立てることができます。

- ［**大文字と小文字を区別する**］：検索の際に大文字と小文字を区別します。［**大文字と小文字を区別する**］を選択した場合，大文字小文字が，［**名前に次を含むオブジェクト**］ボックスに入力した文字列と正確に一致する対象のみが QuickTest によって検索されます。
- ［**単語の全体を一致させる**］：単語の一部ではなく単語全体が一致する文字列を検索します。

5 現在のカーソルの位置からどちらの方向に向かって検索を行うかを指定します。［**上へ**］または［**下へ**］のいずれかを選択できます。

ヒント：［**上へ**］または［**下へ**］を選択すると，現在のカーソルの位置から先頭または末尾に向かって，ターゲット・オブジェクト・リポジトリが検索されます。リポジトリ全体を検索するには，階層内の最初（または最後）のオブジェクトを選択し，［**下へ**］（または［**上へ**］）を選択します。

---

6 ［**次を検索**］をクリックすると，ターゲット・オブジェクト・リポジトリ内で指定の条件に一致する次のオブジェクトが強調表示されます。

［検索］ダイアログ・ボックスを閉じて次のコマンドを使用することもできます。

- ［**次を検索**］ボタンをクリックするか，［**ナビゲート**］＞［**次を検索**］を選択すると，指定の条件に一致する次のオブジェクトが強調表示されます。

- ［**前を検索**］ボタンをクリックするか，［**ナビゲート**］＞［**前を検索**］を選択すると，指定の条件に一致する前のオブジェクトが強調表示されます。

ターゲット・リポジトリの保存

オブジェクトの矛盾が意図どおりに解決されたことを確認したら，ターゲット・リポジトリをファイル・システムまたは Quality Center プロジェクト（QuickTest が現在 Quality Center プロジェクトに接続している場合）に保存できます。

保存できるファイルは，結合したオブジェクト・リポジトリの種類に応じて異なります。2 つの共有オブジェクト・リポジトリを結合した場合は，作成された新しいターゲット・オブジェクト・リポジトリを保存することができます。1 つ以上のローカル・オブジェクト・リポジトリを共有オブジェクト・リポジトリと結合した場合は，ローカル・オブジェクト・リポジトリからのオブジェクトおよびデータを含んでいる既存の共有オブジェクト・リポジトリ・ファイルを保存することができます。

ファイル・システムへのオブジェクト・リポジトリの保存

新しく結合した共有オブジェクト・リポジトリを，いつでもファイル・システムに保存できます。

オブジェクト・リポジトリをファイル・システムに保存するには，次の手順を実行します。

1 ［**ファイル**］＞［**保存**］を選択するか，［**保存**］ボタンをクリックします。ファイルを以前に保存したことがあれば，現在の変更内容が保存されます。ファイルを保存したことがなければ，［共有オブジェクト リポジトリの保存］ダイアログ・ボックスが開きます。

注：Quality Center に接続している場合の［共有オブジェクト リポジトリの保存］ダイアログ・ボックスは，標準のファイル選択ダイアログ・ボックスとは異なります。このダイアログ・ボックスにある［**ファイル システム**］ボタンをクリックすることで，ファイルをファイル・システムに保存するよう切り替えることができます。

---

2 オブジェクト・リポジトリを保存するフォルダに移動し，そのフォルダを選択します。オブジェクト・リポジトリの名前を［**ファイル名**］ボックスに入力します。

ファイルが識別しやすいように，分かりやすい名前を使用します。オブジェクト・リポジトリ・ファイルの名前では，次の文字は使用できません。
: ¥ / : " ? < > | *

3 ［**保存**］をクリックします。QuickTest によって，ファイル名に **.tsr** 拡張子が付けられ，指定された場所にオブジェクト・リポジトリが保存されます。そして［オブジェクト リポジトリ - 結合ツール］ウィンドウ内のターゲット・オブジェクト・リポジトリの上に，ファイル名とパスが表示されます。

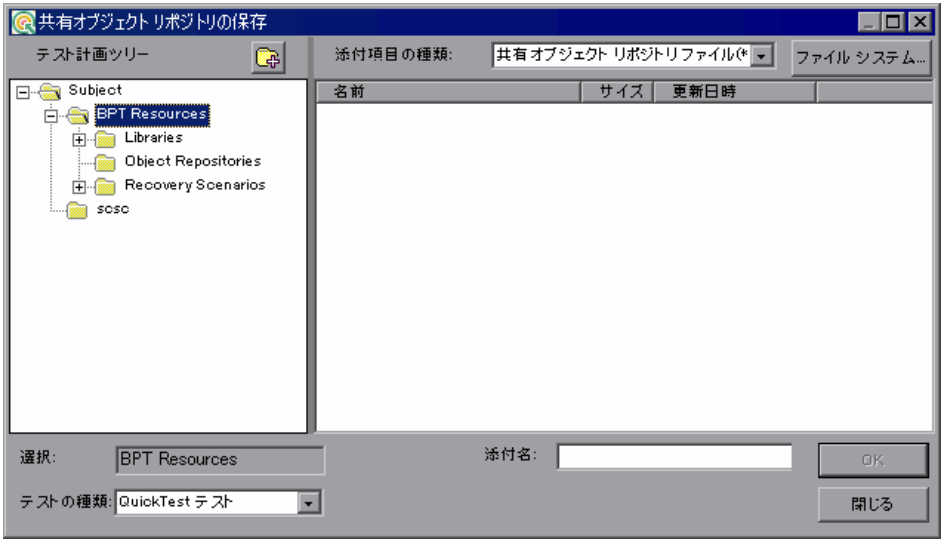
Quality Center プロジェクトへのオブジェクト・リポジトリの保存

Quality Center に接続している場合は，結合後の共有オブジェクト・リポジトリを，プロジェクトのテスト計画ツリー内のテストへの添付ファイルとして保存できます。

注：Quality Center 内の既存のオブジェクト・リポジトリを上書きすることはできません。

オブジェクト・リポジトリを Quality Center プロジェクトに保存するには，次の手順を実行します。

1 ［**ファイル**］＞［**保存**］を選択するか，［**保存**］ボタンをクリックします。ファイルを以前に Quality Center に保存したことがあれば，現在の変更内容がオブジェクト・リポジトリに保存されます。ファイルを保存したことがなければ，［**共有オブジェクト リポジトリの保存**］ダイアログ・ボックスが開きます。



2 テスト計画ツリーの中で，オブジェクト・リポジトリの保存先となるテストまたはフォルダを選択します。

［**フォルダの新規作成**］ボタンをクリックして，Quality Center のテスト計画ツリーに新しいテスト・フォルダを作成することもできます。

注：［共有オブジェクト リポジトリの保存］ダイアログ・ボックスにある［**ファイル システム**］ボタンをクリックすると，ファイルをファイル・システムに保存するように切り替えることができます。［**Quality Center**］ボタンをクリックすれば，Quality Center 用の［共有オブジェクト リポジトリの保存］ダイアログ・ボックスに戻ることができます。

---

3 オブジェクト・リポジトリの名前を［**添付名**］ボックスに入力します。

オブジェクト・リポジトリが識別しやすいように，分かりやすい名前を使用します。オブジェクト・リポジトリ・ファイルの名前では，次の文字は使用できません。
¥ / : " ? < > | *

注：既存のオブジェクト・リポジトリを上書きすることはできません。

---

4 ［**OK**］をクリックします。QuickTest によって，オブジェクト・リポジトリが Quality Center に保存され，［オブジェクト リポジトリ - 結合ツール］ウィンドウ内のターゲット・オブジェクト・リポジトリの上にファイル名とパスが表示されます。Quality Center では，ファイルは対応するテストまたはフォルダの［添付ファイル］タブに表示されます。

第 3 部

高度な設定

第 10 章

Web イベント記録の設定

QuickTest で要求に合うイベントが記録されない場合，Web オブジェクトの種類ごとに記録するイベントを設定できます。

本章では，次の項目について説明します。

- Web イベント記録の設定について
- 標準で使用するイベント記録設定の選択
- イベント記録設定のカスタマイズ
- マウスの右ボタン・クリックの記録
- ユーザ定義イベント設定ファイルの保存と読み込み
- イベント記録設定のリセット

Web イベント記録の設定について

QuickTest では，Web ベースのアプリケーションで実行したイベントを記録することで，テストを作成できます。イベントとは，状態の変更などの操作に応えて行われる通知や，ユーザがドキュメントを表示しているときにマウスをクリックしたり，キーを押したりした結果行われる通知のことです。記録する必要のあるイベントの数が，QuickTest の標準設定に応じて自動的に記録される数よりも多い場合や少ない場合があります。そのような場合，［Web イベント記録の設定］ダイアログ・ボックスで，3 つの定義済みの設定から，いずれかを選択することで標準で使用するイベント記録設定を変更できます。また，特定の条件に合わせて，イベント記録の設定を個別にカスタマイズすることもできます。

たとえば QuickTest では，通常はリンク・オブジェクト上の mouseover イベントは記録されません。しかし，マウスを対象の上に移動したときに生じる**動作**（マウスオーバ動作）がリンクに関連付けられている場合は，mouseover イベントを記録することが重要になるかもしれません。この場合，リンク・オブジェクトが操作に関連付けられているときに，リンク・オブジェクト上の mouseover イベントが必ず記録されるように，設定をカスタマイズできます。

注：

イベント設定はグローバルな設定のため，設定を変更した後で記録されるすべてのテストに影響します。

イベント設定の変更は，すでに記録されたテストには影響しません。必要なイベントが QuickTest で記録されなかった場合や，不要なイベントが記録された場合は，イベント記録設定を変更し，テストの中でその変更の影響を受ける部分を再度記録します。

［ユーザ定義 Web イベント記録の設定］の設定に対する変更は，すでに開いているブラウザには直接反映されません。［Web イベント記録の設定］ダイアログ・ボックスで必要な変更を行い，その変更を既存のテストに適用するには，開いているブラウザを更新し，新しい記録セッションを開始します。

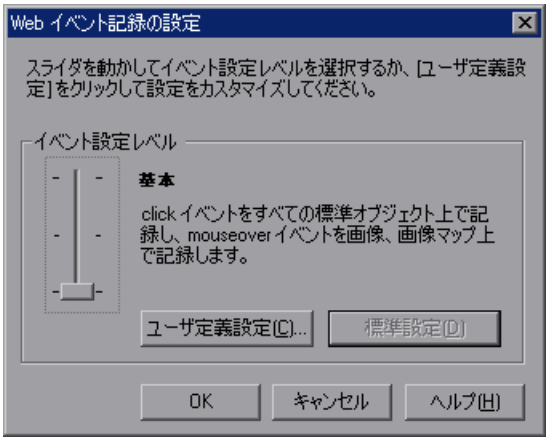
標準で使用するイベント記録設定の選択

［Web イベント記録の設定］ダイアログ・ボックスでは，3 つの定義済みのイベント設定レベルを選択できます。特に指定がなければ，QuickTest では**基本**記録設定レベルが使用されます。必要なイベントの一部が QuickTest で記録されない場合は，イベント設定のレベルを高くする必要があります。

レベル	詳細
［基本］	**標準設定。** • 画像，ボタン，ラジオ・ボタンなどの標準的な Web オブジェクトに対するクリック・イベントを必ず記録します。 • フォーム内での送信イベントを必ず記録します。 • **ハンドラ**または**動作**が関連付けられているその他のオブジェクトでのクリック・イベントを記録します。ハンドラおよび動作の詳細については，305 ページ「応答条件」を参照してください。 • イメージおよびイメージ・マップに対する mouseover イベントに続くイベントが当該オブジェクトを対象とするものである場合に限り，当該 mouseover イベントを記録します。
［中］	基本レベルで記録されるオブジェクト以外に，HTML タグ・オブジェクトの <DIV>，<SPAN>，<TD> に対するクリック・イベントも記録します。
［高］	基本レベルで記録されるオブジェクト以外に，**ハンドラ**または**動作**が関連付けられているオブジェクトに対する mouseover イベント，mousedown イベント，および double-click イベントを記録します。 ハンドラおよび動作の詳細については，305 ページ「応答条件」を参照してください。

標準で使用するイベント記録の設定を設定するには，次の手順を実行します。

1 ［**ツール**］ > ［**Web イベント記録の設定**］を選択します。［Web イベント記録の設定］ダイアログ・ボックスが表示されます。



2 スライダを使って，標準で使用するイベント記録設定を選択します。

ヒント：［**ユーザ定義設定**］ボタンをクリックして［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスを開けます。ここでイベント記録の設定をカスタマイズできます。詳細については，イベント記録設定のカスタマイズを参照してください。

［**標準設定**］ボタンをクリックして，スライダを［**基本**］レベルの位置に戻すことができます。

---

3 ［**OK**］をクリックします。

イベント記録設定のカスタマイズ

標準のイベント設定レベルで必要な記録が行われない場合は，［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスを使って，イベント記録設定をカスタマイズできます。

［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスでは，いくつかの方法でイベント記録をカスタマイズできます。次のことができます。

- QuickTest で特別な応答設定または記録設定を適用する対象となるオブジェクトの追加または削除。
- QuickTest が応答するべきイベントの追加または削除。
- イベントの応答と記録の設定変更。

イベント記録の設定をカスタマイズするには，次の手順を実行します。

1 ［**ツール**］＞［**Web イベント記録の設定**］を選択します。［**Web イベント記録の設定**］ダイアログ・ボックスが表示されます。

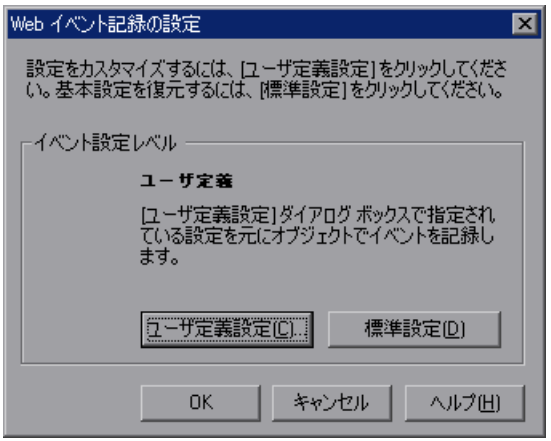
2 ［**ユーザ定義設定**］ボタンをクリックします。［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスが開きます。

3 次のオプションを使って，イベント記録設定をカスタマイズします。

オプション	詳細
オブジェクト表示枠	Web テスト・オブジェクト・クラスと HTML タグ・オブジェクトのリストが表示されます。 • オブジェクトを追加するには，［**オブジェクト**］＞［**追加**］を選択します。 • HTML タグ・オブジェクトのみ削除できます。リストから HTML オブジェクトを削除するには，［**オブジェクト**］＞［**削除**］を選択します。 詳細については，302 ページ「ユーザ定義のオブジェクト・リストに対するオブジェクトの追加と削除」を参照してください。
イベント表示枠	オブジェクトに関連付けられているイベントのリストが表示されます。 • イベント表示枠にイベントを追加するには，［**イベント**］＞［**追加**］を選択します。 • イベントを削除するには，［**イベント**］＞［**削除**］を選択します。 詳細については，304 ページ「オブジェクトの応答イベントの追加と削除」を参照してください。
［イベント名］	イベントの名前。

オプション	詳細
［応答］	QuickTest がイベントに応答する際の基準。 • **Always**：常にイベントに応答します。 • **If Handler**：ハンドラが結び付けられているイベントに応答します。ハンドラは，Web ページに含まれているコードであり，通常はスクリプト言語で書かれている関数またはルーチンです。対応するイベントが発生したときに制御が渡されます。 • **If Behavior**：DHTML 動作が結び付けられているイベントに応答します。DHTML 動作は，ページ上の特定の機能または動作をカプセル化します。ページ上の標準的な HTML 要素に適用されている場合，その要素の標準設定の動作が拡張されます。 • **If Handler or Behavior**：ハンドラまたは動作が結び付けられているイベントに応答します。 • **Never**：イベントに一切応答しません。 詳細については，305 ページ「イベントの応答設定と記録設定の変更」を参照してください。
［記録］	選択されたオブジェクトのイベントの記録を有効 / 無効にする，あるいは同じオブジェクトに対してその後イベントが発生した場合にのみイベントの記録を有効 / 無効にします。
［リセット］	あらかじめ設定されていたレベルに設定を戻します。

4 [**OK**] をクリックします。[ユーザ定義 Web イベント記録の設定] ダイアログ・ボックスが閉じます。[Web イベント記録の設定] ダイアログ・ボックスのスライダが消え，設定の内容として「**ユーザ定義**」と表示されます。



ユーザ定義のオブジェクト・リストに対するオブジェクトの追加と削除

[ユーザ定義 Web イベント記録の定義] ダイアログ・ボックスには，オブジェクト階層内のオブジェクトのリストが表示されます。階層の一番上には，[**任意の Web オブジェクト**] があります。[任意の Web オブジェクト] の設定は，テスト対象となる Web ページ上のすべてのオブジェクトにのうち，特にイベント記録設定が設定されていないすべてのオブジェクトに適用されます。その下には [**Web オブジェクト**] と [**HTML タグ オブジェクト**] カテゴリがあり，どちらにもオブジェクトのリストが含まれています。

[ユーザ定義 Web イベント記録の設定] ダイアログ・ボックス内のオブジェクトを使って作業するときは，次の原則に従います。

- オブジェクトが [ユーザ定義 Web イベント記録の設定] ダイアログ・ボックス内のリストに表示されている場合，そのオブジェクトの設定は [任意の Web オブジェクト] の設定に優先します。
- [**Web オブジェクト**] カテゴリのオブジェクト・リストに対する追加と削除はできませんが，任意のオブジェクトの設定を変更できます。
- Web ページの任意の HTML タグ・オブジェクトを，[HTML タグ オブジェクト] カテゴリに追加できます。

イベント設定オブジェクト・リストにオブジェクトを追加するには，次の手順を実行します。

1 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスで，［**オブジェクト**］＞［**追加**］を選択します。［HTML タグ オブジェクト］リストに，「**新しいオブジェクト**」が表示されます。

2 名前を変更するには，［**新しいオブジェクト**］をクリックします。HTML タグをそのまま名前として入力します。

標準設定では，新しいオブジェクトは，ハンドラが関連付けられている **onclick** イベントの応答と記録を行うように設定されています。

イベントの追加および削除の詳細については，オブジェクトの応答イベントの追加と削除を参照してください。応答設定および記録設定の詳細については，305 ページ「イベントの応答設定と記録設定の変更」を参照してください。

［HTML タグ オブジェクト］リストからオブジェクトを削除するには，次の手順を実行します。

1 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスの［HTML タグ オブジェクト］カテゴリで，削除するオブジェクトを選択します。

2 ［**オブジェクト**］＞［**削除**］を選択します。選択されていたオブジェクトがリストから削除されます。

注：［**Web オブジェクト**］カテゴリからオブジェクトを削除することはできません。

オブジェクトの応答イベントの追加と削除

QuickTest がオブジェクトへ応答するトリガとなるイベントのリストに対して，イベントの追加や削除を行うことができます。

オブジェクトに応答イベントを追加するには，次の手順を実行します。

1 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスの中で，イベントを追加する対象となるオブジェクト，または［**任意の Web オブジェクト**］を選択します。

2 ［**イベント**］＞［**追加**］を選択します。使用できるイベントのリストが開きます。

3 追加するイベントを選択します。イベントは［イベント名］列にアルファベット順で表示されます。標準設定では，QuickTest はハンドラが結び付けられているイベントに応答し，そのイベントを（それが何らかのレベルで応答されている限り）必ず記録します。

応答設定および記録設定の詳細については，イベントの応答設定と記録設定の変更を参照してください。

オブジェクトから応答イベントを削除するには，次の手順を実行します。

1 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスの中で，イベントを削除する対象となるオブジェクト，または［**任意の Web オブジェクト**］を選択します。

2 削除するイベントを［**イベント名**］カラムから選択します。

3 ［**イベント**］＞［**削除**］を選択します。［**イベント名**］カラムからイベントが削除されます。

イベントの応答設定と記録設定の変更

オブジェクトごとにリスト表示される各イベントについて，応答条件を選択し，記録するかどうかを設定できます。

注：応答と記録の設定は相互に独立しています。つまり，オブジェクトに対するイベントに応答しても，それを記録しないことや，オブジェクトに対するイベントに応答せずに，そのイベントを記録することができます。詳細については，307 ページ「イベントの応答と記録を行うためのヒント」を参照してください。

応答条件

イベントごとに，イベント・ハンドラがイベントに関連付けられている場合，DHTML 動作がイベントに関連付けられている場合，イベント・ハンドラまたは DHTML 動作がイベントに関連付けられている場合に，オブジェクトでイベントが発生するたびに応答するように，あるいはイベントに一切応答しないように QuickTest に指示できます。

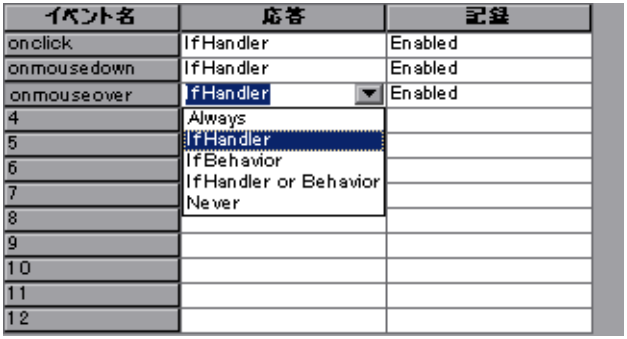
イベント・**ハンドラ**は，Web ページに含まれているコードであり，通常はスクリプト言語で書かれている関数またはルーチンです。対応するイベントが発生したときに制御が渡されます。

DHTML 動作は，ページ上の特定の機能または動作をカプセル化します。ページ上の標準的な HTML 要素に適用されている場合，その要素の標準設定の動作が拡張されます。

イベントに対する応答条件を指定するには，次の手順を実行します。

1 ［ユーザ定義 Web イベント記録設定］ダイアログ・ボックスの中で，応答条件を変更する対象となるオブジェクト，または［**任意の Web オブジェクト**］を選択します。

2 変更するイベントの行で，［**応答**］カラムから必要な応答条件を選択します。



［**Always**］，［**If Handler**］，［**If Behavior**］，［**If Handler or Behavior**］，または［**Never**］のいずれかを選択できます。

記録ステータス

イベントごとに，対象イベントを記録するようにも，記録しないようにも，あるいは次のイベントが選択されたイベントに依存している場合だけ記録するようにも設定できます。

➤ ［**Enabled**］：QuickTest が対象オブジェクトあるいはイベントの「バブリング先」である別のオブジェクトを応答している場合，イベントが生じるたびにそれを記録します。

バブリングとは，子オブジェクトで発生したイベントが，イベントを処理するイベント・ハンドラに遭遇するまで，HTML コード内の階層をさかのぼる処理です。

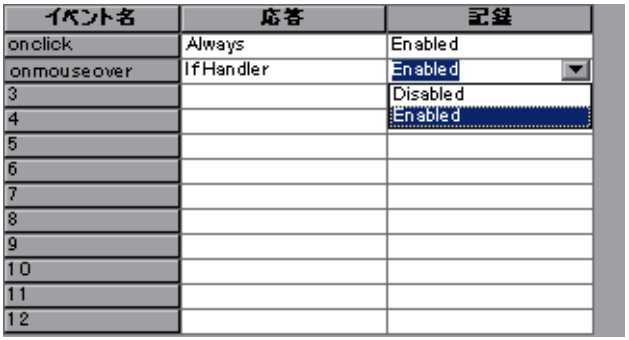
➤ ［**Disabled**］：指定されたイベントを記録せず，イベント・バブリングがある場合はそれを無視します。

➤ ［**Enabled on next event**］：［**Enabled**］との唯一の違いは，以降のイベントが同じオブジェクトで発生した場合にのみイベントを記録することです。たとえば，マウスオーバ動作によって画像リンクが変わるとします。この画像の上をマウスが通過するたびに，mouseover イベントを記録する必要はないかもしれません。ただし，リンクが mouseover イベント後に表示される画像によってのみ有効になるので，mouseover イベントを，同じオブジェクトに対するクリック・イベントの前に記録することが重要となります。このオプションは，Image および WebArea オブジェクトに対してのみ適用されます。

イベントの記録ステータスを設定するには，次の手順を実行します。

1 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスで，記録ステータスを変更する対象となるオブジェクト，または［**任意の Web オブジェクト**］を選択します。

2 変更対象のイベントの行で，記録ステータスを［記録］カラムから選択します。



イベントの応答と記録を行うためのヒント

理想的な応答と記録の設定を見つけるのが困難な場合があります。これらの設定を行うときには，次のガイドラインに留意します。

➤ オブジェクト表示枠の異なるオブジェクトの設定が矛盾する場合，QuickTest は，特定の［**HTML タグ オブジェクト**］の設定を最優先し，次に［**Web オブジェクト**］の設定を優先します。QuickTest が［**任意の Web オブジェクト**］の設定を Web オブジェクトに適用する対象は，［**HTML タグ オブジェクト**］と「**標準オブジェクト**」の範囲のどちらにも定義されていない Web オブジェクトに限ります。

- オブジェクトのイベントを記録するには，QuickTest がイベントに応答し，イベントが生じたときにそれを記録するように指定します。子プロジェクトのイベントは，そのイベントに対するハンドラまたは動作が親オブジェクトに含まれている場合にも応答できます。また，親オブジェクトのイベントは，そのイベントに対するハンドラまたは動作が子オブジェクトに含まれていても，応答できます。

  ただし，ソース・オブジェクト（どの親オブジェクトがハンドラまたは動作を含んでいるかによらずイベントが実際に発生したオブジェクト）に対するイベントの記録は有効にしておかなければなりません。

  たとえば，onmouseover イベント・ハンドラのあるテーブル・セルに 2 つのイメージが含まれているとします。マウスがどちらかのイメージの上を移動すると，バブリングによってイベントがそのセルに送られます。このバブリングにはマウスがどちらのイメージの上で移動したのかを示す情報が含まれています。この mouseover イベントは，次のようにして記録できます。

  - <TD> タグの mouseover イベントに対する応答を［**If Handler**］に設定する一方で（イベントが生じたときに QuickTest にイベントが「応答する」ように），イベントは記録しないようにした上で，<IMG> タグの mouseover イベントに対する応答は［**Never**］に設定し，<IMG> タグに対する記録は［Enabled］（<TD> レベルで応答した後の画像に対する mouseover イベントを記録します）に設定します。
  - <IMG> タグの mouseover イベントに対する応答を［**Always**］（イメージ・タグが動作またはハンドラを含んでいなくても mouseover イベントを応答する）に設定し，<IMG> タグに対する記録を［**Enabled**］（イメージに対する mouseover イベントを記録する）に設定します。

- 多数のオブジェクト上の多数のイベントを応答するように設定すると，QuickTest のパフォーマンスが低下することがあるので，応答の設定は，必要なオブジェクトに限定することをお勧めします。
- まれに，イベントが発生するオブジェクト（ソース・オブジェクト）に応答していると，イベントが妨害されることがあります。

  QuickTest を使ってアプリケーションの記録を始めるまでは，そのアプリケーションが正常に動作していたのであれば，**応答**の設定によって妨害が生じている可能性があります。

この問題がマウス・イベントのときに発生した場合，［詳細 Web オプション］ダイアログ・ボックスで，適切な［**標準 Windows マウスイベントの使用**］オプションを選択してみます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 737 ページ「詳細 Web オプション」を参照してください。

この問題がキーボード・イベントまたは内部イベントのときに発生した場合，あるいは［**標準 Windows マウスイベントの使用**］オプションを指定しても問題が解決しない場合，そのイベントに対するソース・オブジェクトでの応答の設定を［**Never**］に設定して（ただし，ソース・オブジェクトに対する記録の設定は Enabled のまま），親オブジェクトに対する応答の設定を［**Always**］にします。

マウスの右ボタン・クリックの記録

QuickTest では，マウスの左ボタン，中央ボタン，右ボタンを使用したクリックを記録できます。標準設定では，左クリックのみ記録されますが，右ボタンや中央ボタンのクリックも記録するように設定を変更できます。

OnClick イベントが発行されると **Click** ステートメントが記録されます。QuickTest は，マウスの各ボタンに対して設定されたイベントを応答することによって，マウス・ボタンを区別します。標準設定では，OnMouseUp イベントに応答しますが，［Web イベント記録の設定］ダイアログ・ボックスを使用して，OnMouseDown イベントに応答するように設定することもできます。

注：

複数のマウス・ボタンの同時クリックの記録はサポートされていません。

QuickTest では，ブラウザのショートカット・メニューを開く右クリック，およびショートカット・メニューからの項目の選択は記録されません。スクリプトを手作業で変更してこれらのオプションを有効にする方法の詳細については，次のナレッジ・ベース項目を参照してください。

- **問題 ID 31270**：How to replay right-clicking on an object to open a pop-up menu（オブジェクト上で右クリックしてポップアップ・メニューを開く動作の再生方法）

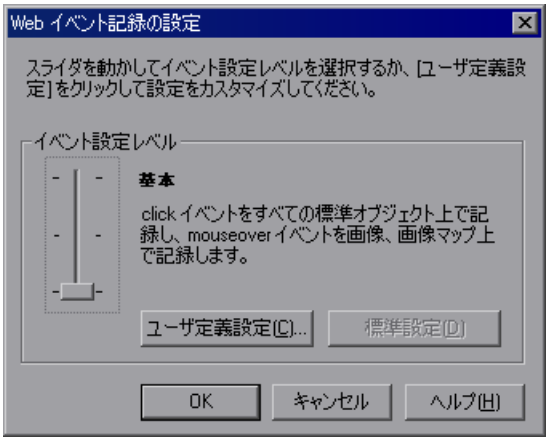
➤ **問題 ID 27184**：How to select an item from a right-click menu（右クリック・メニューからの項目の選択方法）

マウスの右クリックを記録するための QuickTest の設定

設定ファイルを手作業で変更してから読み込むことによって，マウスの右クリックを記録するよう QuickTest に指示します。

マウスの右クリックを記録するよう QuickTest に指示するには，次の手順を実行します。

1 ［**ツール**］＞［**Web イベント記録の設定**］を選択します。［Web イベント記録の設定］ダイアログ・ボックスが表示されます。



2 ［**ユーザ定義設定**］ボタンをクリックします。［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスが開きます。

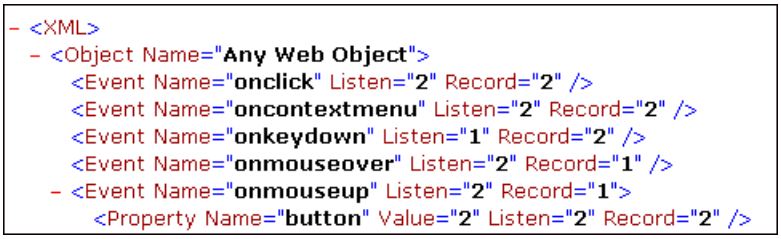
3 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスで，［**ファイル**］＞［**設定に名前を付けて保存**］を選択します。［名前を付けて保存］ダイアログ・ボックスが表示されます。

4 Web イベント記録設定ファイルを保存するフォルダに移動し，設定ファイル名を入力します。設定ファイルの拡張子は **.xml** です。

5 ［**保存**］をクリックしてファイルを保存し，ダイアログ・ボックスを閉じます。

6 編集のために，保存した設定ファイルをテキスト・エディタで開きます。設定ファイルでは定義済みの構造が使用されています。この XML ファイルの構造の詳細については，314 ページ「Web イベント記録の設定の XML 構造について」を参照してください。

Web オブジェクトに関連するファイルの冒頭部分を以下に示します。



Property Name 引数が，マウス・ボタンの記録を制御します。マウス・ボタンの値は次のように定義されています。

- **1**：左
- **2**：右
- **4**：中央

7 ファイルを次のように編集します。

- **onmouseup** イベントに対するマウスの左クリックを記録するには，次の行を追加します。

  <Property Name="button" Value="1" Listen="2" Record="2"/>

- **onmousedown** イベントに対するマウスの右クリックおよび左クリックを記録するには，次の行を追加します。

  <Event Name="onmousedown" Listen="2" Record="1">

  <Property Name="button" Value="2" Listen="2" Record="2"/>

  <Property Name="button" Value="1" Listen="2" Record="2"/>

  </Event>

注：onmouseup または **onmousedown** のどちらか 1 つのイベントのみを使用してマウス・クリックを処理してください。両方のイベントを使用した場合は，1 つではなく 2 つのクリックが記録されます。標準設定では，QuickTest は **onmouseup** イベントに応答します。

---

8 ファイルを保存します。

9 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスで，［**ファイル**］＞［**設定の読み込み**］を選択します。［ファイルを開く］ダイアログ・ボックスが開きます。

10 編集した設定ファイルを保存したフォルダに移動し，ファイルを選択して，［**開く**］をクリックします。［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスが再び開きます。

11 ［**OK**］をクリックします。新しい設定が読み込まれ，すべての設定が XML 設定ファイルで定義したものに対応するようになります。以降，記録する Web オブジェクトはこれらの新しい設定に従って記録されます。

ユーザ定義イベント設定ファイルの保存と読み込み

［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスで行った変更は保存して，必要なときに読み込むことができます。

XML ファイルを読み込む前に変更することもできます。この XML ファイルの構造の詳細については，314 ページ「Web イベント記録の設定の XML 構造について」を参照してください。

ユーザ定義設定を保存するには，次の手順を実行します。

1 イベント記録設定を，必要に合わせてカスタマイズします。設定をカスタマイズする方法の詳細については，299 ページ「イベント記録設定のカスタマイズ」を参照してください。

2 ［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスで，［**ファイル**］＞［**設定に名前を付けて保存**］を選択します。［名前を付けて保存］ダイアログ・ボックスが表示されます。

3 イベント設定ファイルを保存するフォルダに移動し，設定ファイル名を入力します。設定ファイルの拡張子は **.xml** です。

4 ［**保存**］をクリックしてファイルを保存し，ダイアログ・ボックスを閉じます。

ユーザ定義設定を読み込むには，次の手順を実行します。

1 ［**ツール**］ > ［**Web イベント記録の設定**］を選択し，［**ユーザ定義設定**］をクリックします。［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスが表示されます。

2 ［**ファイル**］ > ［**設定の読み込み**］を選択します。［ファイルを開く］ダイアログ・ボックスが開きます。

3 読み込むイベント設定ファイル（**.xml**）を見つけて，［**開く**］をクリックします。ダイアログ・ボックスが閉じ，選択した設定が読み込まれます。

Web イベント記録の設定の XML 構造について

Web イベント記録の設定の XML ファイルは，特定の構造形式になっています。ファイルに変更を加える場合や，独自のファイルを作成する場合，設定が有効であるためにはこの形式に従う必要があります。

XML ファイルの例を以下に示します。

```
<XML>
    <Object Name="Any Web Object">
        <Event Name="onclick" Listen="2" Record="2"/>
        <Event Name="onmouseup" Listen="2" Record="1">
            <Property Name="button" Value="2" Listen="2" Record="2"/>
        </Event>
    </Object>
        . . .
        . . .
        . . .
    <Object Name="WebList">
        <Event Name="onblur" Listen="1" Record="2"/>
        <Event Name="onchange" Listen="1" Record="2"/>
        <Event Name="onfocus" Listen="1" Record="2"/>
    </Object>
</XML>
```

次の値を使用して応答条件と記録ステータスのオプションを XML 形式で定義します。

設定	指定可能な値
［応答］	**1**：Always **2**：If Handler **4**：If Behavior **6**：If Handler or Behavior **0**：Never
［記録］	**1**：Disabled **2**：Enabled **6**：Enabled on Next Event

イベント記録設定のリセット

ユーザ定義の設定を行った後で標準の設定に戻すには，［Web イベント記録の定義］ダイアログ・ボックスで，イベント記録設定を基本レベルにリセットします。

注：標準設定をリセットすると，ユーザ定義の設定は完全になくなります。変更内容を失わないようにするには，イベント設定ファイルに設定を保存しておく必要があります。詳細については，313 ページ「ユーザ定義イベント設定ファイルの保存と読み込み」を参照してください。

［Web イベント記録の設定］ダイアログ・ボックスを使用して設定を基本レベルにリセットするには，次の手順を実行します。

1 ［**ツール**］＞［**Web イベント記録の設定**］を選択します。［Web イベント記録の設定］ダイアログ・ボックスが表示されます。

2 ［**標準設定**］をクリックします。標準設定スライダが再び表示され，すべてのイベント設定が**基本**イベント記録設定レベルに戻されます。

3 別の定義済みレベルを選択するには，297 ページ「標準で使用するイベント記録設定の選択」を参照してください。

［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスを使用して，設定を特定の（基本）ユーザ定義設定に戻すこともできます。これにより，その状態からカスタマイズを開始できます。

［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスを使用して，設定を特定のユーザ定義レベルにリセットするには，次の手順を実行します。

1 ［**ツール**］＞［**Web イベント記録の設定**］を選択します。［Web イベント記録の設定］ダイアログ・ボックスが表示されます。

2 ［**ユーザ定義設定**］ボタンをクリックします。［ユーザ定義 Web イベント記録の設定］ダイアログ・ボックスが表示されます。

3 ［**戻した後の値**］ボックスで，使用する定義済みイベント記録レベルを選択します。

4 ［**リセット**］をクリックします。すべてのイベント設定が，選択したレベルの標準設定に戻されます。

第 11 章

［エキスパート ビュー］および関数ライブラリ・ウィンドウのカスタマイズ

［エキスパート ビュー］でのテストの表示方法および，関数ライブラリ・ウィンドウでの関数の表示方法をカスタマイズできます。変更したすべての内容は，［エキスパート ビュー］およびすべての関数ライブラリ・ウィンドウにグローバルに適用されます。

本章では，次の項目について説明します。

- ［エキスパート ビュー］および関数ライブラリ・ウィンドウのカスタマイズについて
- エディタの動作のカスタマイズ
- エレメントの見映えのカスタマイズ
- 編集コマンドのカスタマイズ

［エキスパート ビュー］および関数ライブラリ・ウィンドウのカスタマイズについて

QuickTest には，強力でカスタマイズ可能なエディタが用意されています。これを使用して，［エキスパート ビュー］および関数ライブラリ・ウィンドウの各部を変更できます。

［エディタ オプション］ダイアログ・ボックスでは，［エキスパート ビュー］および関数ライブラリ・ウィンドウでの，スクリプトや関数ライブラリの表示方法を変更できます。また，スクリプトや関数ライブラリに表示されるテキストのフォントや文字の大きさを変更したり，コメント，文字列，QuickTest の予約語，演算子，数字などスクリプトのエレメントごとに色を変えたりできます。たとえば，すべての文字列を赤で表示することもできます。

QuickTest には，カーソルの移動，文字の削除，クリップボードを使った情報の切り取り，コピー，貼り付けを行うことができる，標準設定のキーボード・ショートカットのリストがあります。これらのショートカットは，任意のショートカットに置き換えることができます。たとえば，**Line start** コマンドを標準設定の HOME から ALT + HOME へ変更できます。

［印刷］ダイアログ・ボックスのオプションを使用して，スクリプトまたは関数ライブラリの印刷方法を変更することもできます。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 109 ページ「テストの印刷」および 185 ページ「関数ライブラリの印刷」を参照してください。

［エキスパート ビュー］の使用方法の詳細については，第 5 章「エキスパート・ビューおよび関数ライブラリ・ウィンドウを使用した作業」を参照してください。関数ライブラリを使った作業の詳細については，第 6 章「ユーザ定義関数および関数ライブラリを使用した作業」を参照してください。

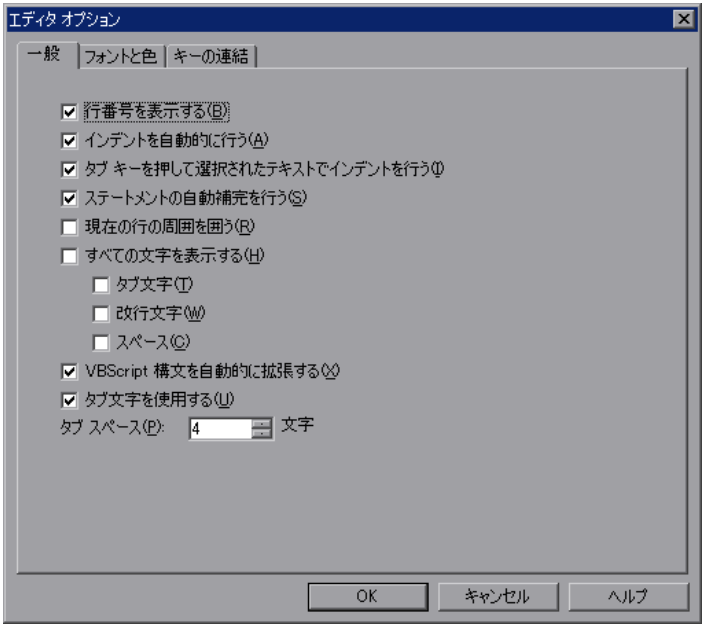
エディタの動作のカスタマイズ

［エキスパート ビュー］および関数ライブラリ・ウィンドウにおけるスクリプトや関数ライブラリの表示方法をカスタマイズできます。たとえば，文字記号を表示または非表示にしたり，行番号を表示するよう選択したりできます。［エキスパート ビュー］の使用方法の詳細については，第 5 章「エキスパート・ビューおよび関数ライブラリ・ウィンドウを使用した作業」を参照してください。関数ライブラリを使った作業の詳細については，第 6 章「ユーザ定義関数および関数ライブラリを使用した作業」を参照してください。

エディタの動作をカスタマイズするには，次の手順を実行します。

1 ［エキスパート ビュー］または関数ライブラリ・ウィンドウがアクティブなときに，［**ツール**］＞［**表示オプション**］を選択します。［エディタ オプション］ダイアログ・ボックスが開きます。

2 ［**一般**］タブをクリックします。



3 次のオプションから選択します。

オプション	詳細
［行番号を表示する］	スクリプトまたは関数の各行の左に行番号が表示されます。
［インデントを自動的に行う］	インデントを設定した行の後に続く行が，自動的に前の行と同じ位置から開始されます。キーボードの HOME キーを押すと，カーソルが左マージンに戻ります。

オプション	詳細
[タブ キーを押して選択されたテキストでインデントを行う]	TAB キーを押すと，選択されたテキストがインデントされます。このオプションが有効でない場合は，Tab キーを押すと，選択されたテキストがタブ文字 1 つで置換されます。
[ステートメントの自動補完を行う]	このオプションが選択されている場合は，次を入力すると，それぞれ下記の内容が表示されます。 • **テスト・オブジェクトの後にドットを入力**：入力したオブジェクトの後ろに追加できる使用可能なテスト・オブジェクトとメソッドのリストが表示されます。 • **オブジェクトの後で開き括弧 (を入力**：QuickTest によって，オブジェクト・リポジトリにこの種類のすべてのテスト・オブジェクトのリストが表示されます。オブジェクト・リポジトリに，当該タイプに一致するオブジェクトが 1 つだけある場合には，そのオブジェクトの名前が開き括弧の後ろに，引用符で囲まれた状態で QuickTest によって自動的に挿入されます。 • **メソッドを入力**：特定の必須および任意の引数を含むメソッドの構文が表示されます。 • **Object プロパティ**：ActiveScreen または開いているアプリケーションでオブジェクト・データが現在使用可能な場合は，アプリケーション内の任意の実行時オブジェクトのネイティブ・メソッドとプロパティが表示されます。 ステートメント補完（IntelliSense）機能の使用の詳細については，120 ページ「エキスパート・ビューまたは関数ライブラリでのステートメントの生成」を参照してください。
[現在の行の周囲を囲う]	テスト内で現在カーソルがある行の周りに枠が表示されます。
[すべての文字を表示する]	すべてのタブ，改行，空白文字などの記号が表示されます。また，対応するチェックボックスを選択 / クリアすれば，これらの文字の一部だけを表示させることもできます。

オプション	詳細
［VBScript 構文を自動的に拡張する］	キーワードの最初の 2 文字を自動的に認識し，該当するキーワードを入力するとそれに対応する VBScript 構文またはブロックをスクリプトに追加します。 たとえば，エキスパート・ビューの行頭で「if」という文字に続いてスペースを入力すると，次の構文が自動的に入力されます。 If ThenEnd If
［タブ文字を使用する］	キーボードの TAB キーを使用したときに，タブ文字が挿入されます。このオプションを選択していなければ，TAB キーを使用したときに代わりに指定された数のスペース文字が挿入されます。

4 ［**OK**］をクリックし，変更を保存してダイアログ・ボックスを閉じます。

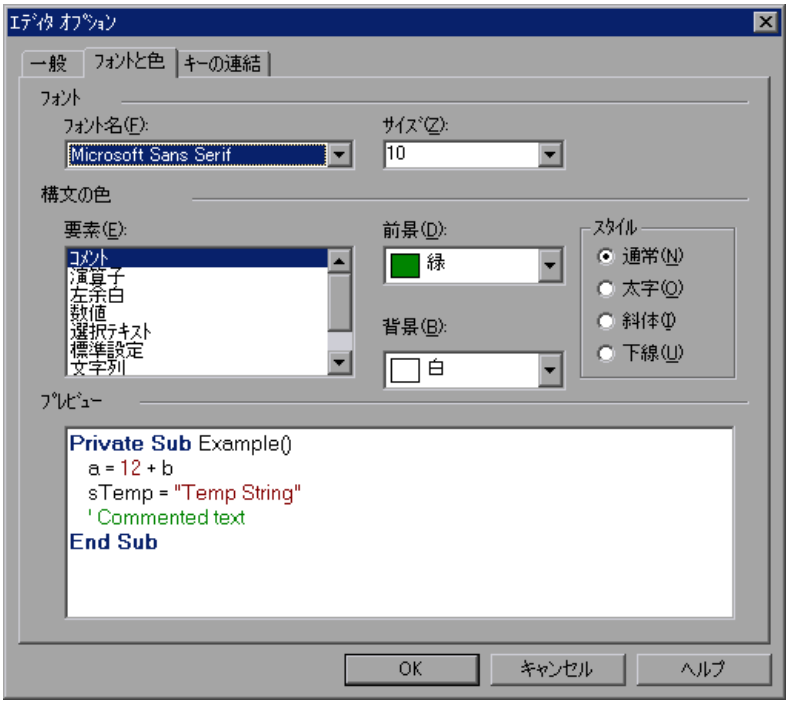
エレメントの見映えのカスタマイズ

QuickTest スクリプトおよび関数ライブラリには，コメント，文字列，QuickTest や VBScript の予約語，演算子，数字などの，さまざまなエレメントが含まれています。QuickTest スクリプトのそれぞれのエレメントは，異なる色で表示できます。エキスパート・ビューに表示されるすべてのエレメントのフォントや文字の大きさを指定することもできます。スクリプトの各エレメントに対して，独自にカスタマイズした配色を作成できます。たとえば，スクリプト内のすべてのコメントを，青の文字と黄色の背景で表示することもできます。

エレメントに対してフォントと色を設定するには，次の手順を実行します。

1 ［エキスパート ビュー］または関数ライブラリ・ウィンドウがアクティブなときに，［**ツール**］＞［**表示オプション**］を選択します。［エディタ オプション］ダイアログ・ボックスが開きます。

2 ［**フォントと色**］タブをクリックします。



3 ［**フォント**］領域で，すべてのエレメントの表示に使用する［**フォント名**］と［**サイズ**］を選択します。このエディタでは Unicode フォントである Microsoft Sans Serif フォントが標準で使用されます。

注：Unicode 環境でテストを行う場合は，Unicode 対応のフォントを選択しなければなりません。Unicode 対応のフォントを選択しないと，テストまたは関数ライブラリ内のエレメントが，［エキスパート ビュー］または関数ライブラリ・ウィンドウに正しく表示されない場合があります。ただし，テストまたは関数ライブラリは選択したフォントに関係なく，指定前と変わりなく実行されます。Unicode 対応でない環境で作業をしている場合は，Courier などの固定幅フォントを使用して，文字の揃えがよくなるようにします。

---

4 ［**要素**］リストから，エレメントを選択します。

5 文字の色と背景の色を選択します。

6 エレメントのフォント・スタイルを選択します（［**通常**］，［**太字**］，［**斜体**］，［**下線**］）

変更を適用した場合の例が，ダイアログ・ボックス下部の**プレビュー**表示枠に表示されます。

7 変更するエレメントごとに，手順 4 から 6 を繰り返します。

8 ［**OK**］をクリックし，変更を適用してダイアログ・ボックスを閉じます。

編集コマンドのカスタマイズ

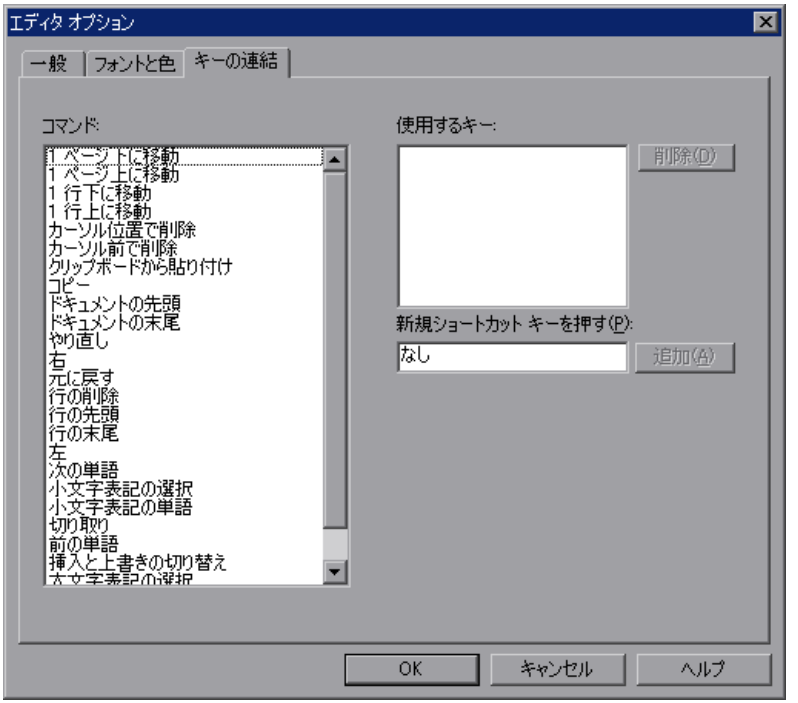
編集に使用する標準のキーボード・ショートカットをカスタマイズできます。QuickTest には，カーソルの移動，文字の削除，クリップボードを使った情報の切り取り，コピー，貼り付けを行うためのキーボード・ショートカットがあります。これらのショートカットは，任意のショートカットに置き換えることができます。たとえば，**Line end** コマンドを標準設定の END から ALT + END へ変更できます。

注：標準の QuickTest メニュー・ショートカット・キーは，ユーザが定義するキーの割り当てに優先します。たとえば，貼り付けコマンドのキーの組み合わせを CTRL + P と設定した場合でも，［印刷］ダイアログ・ボックスを開くための標準のショートカット・キー（［**ファイル**］＞［**印刷**］オプションに対応）が優先されます。QuickTest メニュー・ショートカット・キーの一覧は，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の 45 ページ「ショートカット・キーを使用したコマンドの実行」を参照してください。

編集コマンドをカスタマイズするには，次の手順を実行します。

1 ［エキスパート ビュー］または関数ライブラリ・ウィンドウがアクティブなときに，［**ツール**］＞［**表示オプション**］を選択します。［エディタ オプション］ダイアログ・ボックスが開きます。

2 ［**キーの連結**］タブをクリックします。



3 ［**コマンド**］リストからコマンドを選択します。

4 ［**新規ショートカット キーを押す**］ボックスをクリックして，選択されているコマンドに使用するキーを押します。たとえば，CTRL+4 を入力するには，CTRL キーを押しながら数字の 4 を押します。

5 ［**追加**］をクリックします。

注 : 指定したキーの組み合わせがサポートされていない場合，あるいは別のコマンドに対してすでに定義されている場合は，ショートカット・キー・ボックスの下にその旨を示すメッセージが表示されます。

---

6 ほかに追加するコマンドがあれば手順 3 ～ 5 を繰り返します。

7 リストからキー指定を削除するには，［**コマンド**］リスト内のコマンドを選択して［**使用するキー**］リスト内のキー（またはキーの組み合わせ）を強調表示し，［**削除**］をクリックします。

8 ［**OK**］をクリックし，変更を適用してダイアログ・ボックスを閉じます。

第 12 章

実行セッション中のテスト・オプションの設定

実行セッション中にテスト・オプションを設定したり，取得したりすることによって，QuickTest でのテストの記録と実行の方法を制御できます。

本章では，次の項目について説明します。

- 実行セッション中のテスト・オプションの設定について
- テスト・オプションの設定
- テスト・オプションの取得
- テスト実行の制御
- テスト実行設定の追加と削除

実行セッション中のテスト・オプションの設定について

QuickTest のテスト・オプションは，テストの記録と実行の方法に影響します。たとえば，QuickTest においてページ内でオブジェクトが見つかるまで検索する時間の上限を設定できます。

テスト・オプションの値を実行セッション中に設定および取得するには，エキスパート・ビューで **Setting** オブジェクトを使います。エキスパート・ビューでの作業の詳細については，第 5 章「エキスパート・ビューおよび関数ライブラリ・ウィンドウを使用した作業」を参照してください。

Setting オブジェクトを使ってテスト・オプションを取得および設定することで，QuickTest でのテストの実行方法を制御できます。

多くのテスト・オプションは，［オプション］ダイアログ・ボックス（グローバルなテスト・オプション）および［テストの設定］ダイアログ・ボックス（テスト固有の設定）を使っても設定できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 24 章「グローバル・テスト・オプションの設定」および第 25 章「個別のテストのオプションの設定」を参照してください。

本章では，テスト・スクリプト内から **Setting** オブジェクトを使って設定できる QuickTest のテスト・オプションの一部について説明します。**Setting** オブジェクトに関して使用可能なすべてのメソッドとプロパティの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』の「**ユーティリティ オブジェクト**」を参照してください。

注：QuickTest オプションは，他の操作と同様，自動プログラムを使って外部アプリケーションから制御することもできます。詳細については，215 ページ「QuickTest 操作のオートメーション」を参照するか，『**QuickTest オートメーション・オブジェクト・モデル・リファレンス**』（［**ヘルプ**］＞［**QuickTest オートメーション オブジェクト モデル リファレンス**］）を参照してください。

テスト・オプションの設定

Setting オブジェクトを使って，テスト・オプションの値をテスト・スクリプト内から設定できます。オプションを設定するには，次の構文を使います。

Setting (testing_option) = new_value

オプションの中には，グローバルに設定するものと，テストごとに設定するものがあります。

Setting オブジェクトでグローバルなテスト・オプションを設定すると，テスト・オプションがグローバルに変更され，変更内容は［オプション］ダイアログ・ボックスにも反映されます。

たとえば，次のステートメントを実行するとします。

```
Setting("AutomaticLinkRun")=1
```

QuickTest では，テスト内で自動的に作成されたチェックポイントが無効化されます。現在の QuickTest セッションでは，この設定は，別の **Setting** ステートメントを使うか，または［詳細 Web オプション］ダイアログ・ボックス（［**ツール**］＞［**オプション**］＞［**Web**］タブを選択して［**詳細設定**］をクリック）で［**テストまたはコンポーネントの実行時に自動チェックポイントを無視する**］チェックボックスをクリアして，設定を変更するまで有効です。

Setting オブジェクトでテストごとのオプションを設定した場合にも，変更は［テストの設定］ダイアログ・ボックスに反映されます。**Setting** オブジェクトを使い，特定のテストの特定の部分の設定だけを変更することもできます。詳細については，331 ページ「テスト実行の制御」を参照してください。

たとえば，次のステートメントを実行するとします。

```
Setting("WebTimeOut")=50000
```

QuickTest によって，テストのステップを実行する前に Web ページの読み込みが終わるまで待機する時間の上限が，自動的に 50000 ミリ秒に変更されます。この設定は，現在の QuickTest セッション中に別の **Setting** ステートメントを使うか，［テストの設定］ダイアログ・ボックスの［Web］タブで［**ブラウザ ナビゲーションのタイムアウト**］オプションを使って，設定を変更するまで有効です。

注：Setting オブジェクトを使って行った変更が［オプション］ダイアログ・ボックスと［テストの設定］ダイアログ・ボックスに反映されている場合でも，同じダイアログ・ボックスで他の変更を手作業で行って［**適用**］または［**OK**］をクリックしなければ，これらの変更は QuickTest を閉じたときに保存されません。

テスト・オプションの取得

Setting オブジェクトを使って，テスト・オプションの現在の値を取得することもできます。

変数に値を格納するには，次の構文を使います。

新変数 = Setting (テスト・オプション)

メッセージ・ボックスに値を表示するには，次の構文を使います。

MsgBox (Setting (testing_option))

例を次に示します。

```
LinkCheckSet = Setting("AutomaticLinkRun")
```

これは，AutomaticLinkRun 設定の現在の値を，ユーザ定義変数の LinkCheckSet に代入します。

設定を取得する対象にできるその他のテスト・オプションの例については，329 ページ「テスト・オプションの設定」を参照してください。

テスト実行の制御

Setting オブジェクトの取得機能と設定機能を組み合わせることで，グローバル設定を変更せずに，実行セッションを制御できます。たとえば，**DefaultTimeOut** テスト・オプションを，1 つの Web ページ上のオブジェクトについてだけ 5 秒に変更するには，テスト・スクリプトの Web ページを開く処理の後に，次のステートメントを挿入します。

```
' DefaultTimeOut テスト・オプションの元の値を保存
old_delay = Setting ("DefaultTimeOut")

' DefaultTimeOut テスト・オプションの一時値を設定
Setting("DefaultTimeOut")= 5000
```

Web ページの最後で **DefaultTimeOut** テスト・オプションを元の値に戻すには，スクリプトで次のページにリンクする処理の直前に，次のステートメントを挿入します。

```
' DefaultTimeOut テスト・オプションを元の値に戻す
Setting("DefaultTimeOut")=old_delay
```

テスト実行設定の追加と削除

グローバルな設定や固有の設定だけでなく，実行環境の設定も追加，変更，削除できます。これらの設定は，実行セッション中にだけ適用されます。

新しい実行環境の設定を追加するには，次の構文を使います。

Setting.Add "testing_option", "value"

たとえば，現在のテスト実施者の名前を表示し，その名前をメッセージ・ボックスに表示する設定を作成できます。

```
Setting.Add "Tester Name", "Mark Train"
MsgBox Setting("Tester Name")
```

注：Setting.Add ステートメントを使って，既存のキー値を追加しようとするとエラーが起きます。このエラーを防ぐには，先に **Setting.Exists** ステートメントを使う必要があります。**Setting** メソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

すでに初期化されている実行環境設定を変更するには，標準の設定オプションを設定する場合と同じ構文を使います。

Setting (testing_option) = new_value

例を次に示します。

```
Setting("Tester Name")="Alice Wonderlin"
```

ユーザ定義の実行環境の設定を削除するには，次の構文を使います。

Setting.Remove (testing_option)

例を次に示します。

```
Setting.Remove ("Tester Name")
```

第 4 部

その他の Mercury 製品を使用した作業

第 13 章

WinRunner を使用した作業

QuickTest では，WinRunner のテストを実行したり，コンパイル済みモジュール内の TSL 関数またはユーザ定義関数を呼び出したりすることもできます。

本章では，次の項目について説明します。

- WinRunner を使用した作業について
- WinRunner テストの呼び出し
- WinRunner 関数の呼び出し

WinRunner を使用した作業について

コンピュータに WinRunner 7.5 以上がインストールされている場合には，QuickTest のテストに WinRunner のテストまたは関数への呼び出しを含めることができます。

注：バージョン 7.6 以前の WinRunner の場合には，QuickTest の Web アドインがロードされていると，QuickTest から Web ページを対象に WinRunner テストを（WinRunner の WebTest アドインを使用して）実行することはできません。WinRunner 7.6 でこの機能を有効にするには，Mercury Customer Support サイト（http://support.mercury.com）のパッチ・データベースから **WR76P10 - Support WR/QTP integration** というパッチを入手してインストールします。WinRunner の以降のバージョンでは，この機能は組み込まれて提供されます。

WinRunner テストまたは関数への呼び出しを作成したら，呼び出しステートメントをエキスパート・ビューまたはキーワード・ビューで編集して，そのステートメントの引数値を変更できます。

QuickTest を，WinRunner のテストまたはコンパイル済みモジュールが含まれている Quality Center プロジェクトに接続すると，Quality Center プロジェクトに格納されている WinRunner テストまたは関数を呼び出すことができます。

WinRunner テストの呼び出し

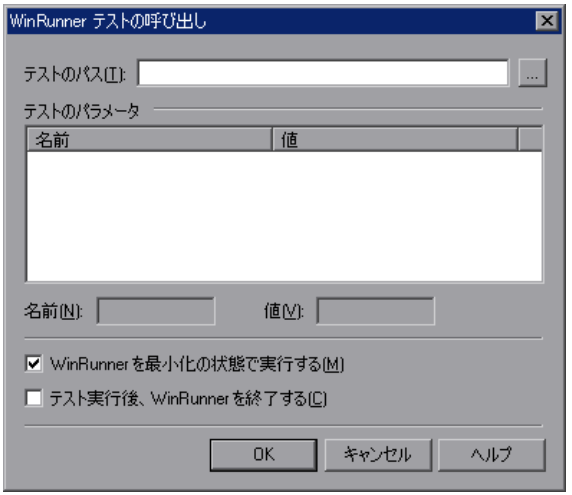
テストを実行するために，QuickTest から WinRunner に接続すると，WinRunner が起動されテストが開かれ，実行されます。WinRunner テスト実行に関する情報は，QuickTest の［テスト結果］ウィンドウに表示されます。

WinRunner テストへの呼び出しを挿入するには，［WinRunner テストの呼び出し］ダイアログ・ボックスを使用するか，エキスパート・ビューの中で **TSLTest.RunTestEx** ステートメントを入力します。

注：QuickTest テストへの呼び出しを含む WinRunner テストを呼び出すことはできません。

［WinRunner テストの呼び出し］ダイアログ・ボックスを使用して WinRunner テストへの呼び出しを挿入するには，次の手順を実行します。

1 ［**挿入**］＞［**WinRunner の呼び出し**］＞［**テスト**］を選択します。［WinRunner テストの呼び出し］ダイアログ・ボックスが表示されます。



2 ［**テストのパス**］ボックスに WinRunner テストのパスを入力するか，参照ボタンを使って WinRunner テストを指定します。

Quality Center に接続していると，参照ボタンをクリックしたときに，Quality Center プロジェクトのモジュールを選択するための，［Quality Center プロジェクトから WinRunner テストを開く］ダイアログ・ボックスが開きます。このダイアログ・ボックスの詳細については，362 ページ「Quality Center プロジェクトのテストを開く」を参照してください。

3 ［テストのパラメータ］ボックスには，WinRunner テストに必要なすべてのテスト・パラメータのリストが表示されます。パラメータの値を入力するには，次の手順を実行します。

➤ ［**テストのパラメータ**］リスト内のパラメータを選択して強調表示にします。選択した値がリストの下の［**名前**］ボックスに表示されます。

➤ ［**値**］ボックスに新しい値を入力します。

注：QuickTest ランダム・パラメータまたは環境パラメータのパラメータ値や，QuickTest データ・テーブルのパラメータ値を，WinRunner テストのパラメータとして使用することもできます。それにはパラメータの情報を **TSLTest.RunTestEx** ステートメントに手作業で入力することもできます。詳細については，340 ページ「QuickTestWinRunner テストへの QuickTest でパラメータ化された値の引き渡し」を参照してください。

---

4 WinRunner ウィンドウがテストの実行中に表示されないようにするには，［**WinRunner を最小化の状態で実行する**］を選択します（このオプションのサポート対象は，WinRunner バージョン 7.6 以降です）。

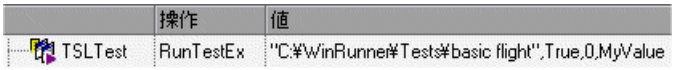
5 WinRunner テストを呼び出したステップが完了したときに WinRunner アプリケーションを終了させるには，［**テスト終了後、WinRunner を終了する**］を選択します（このオプションのサポート対象は，WinRunner バージョン 7.6 以降です）。

6 ［**OK**］をクリックして，ダイアログ・ボックスを閉じます。

WinRunner のテスト・パラメータの詳細については，『**WinRunner ユーザーズ・ガイド**』を参照してください。

QuickTest では，WinRunner テストへの呼び出しが次のように表示されます。

- キーワード・ビューには，WinRunner の **RunTestEx** ステップが表示されます。例を次に示します。



- エキスパート・ビューには，VBScript で書かれた **TSLTest.RunTestEx** ステートメントが表示されます。例を次に示します。

```
TSLTest.RunTestEx "C:¥WinRunner¥Tests¥basic_flight",TRUE, 0, "MyValue"
```

RunTestEx メソッドの構文は，次のとおりです。

TSLTest.RunTestEx テストのパス , 最小化実行 , アプリケーション終了 [, パラメータ]

注：QuickTest 6.0 で作成したテストは，**RunTest** メソッドを使用して WinRunner テストを呼び出している場合があります。このメソッドは，構文が少し異なります。このメソッドでもテストは正常に実行されます。ただし，WinRunner 7.6 以降を使用している場合には，**RunTestEx** メソッド（および対応する引数の構文）を使用するようにテストを更新することをお勧めします。これらのメソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

RunTestEx メソッドと利用方法のサンプルの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

QuickTestWinRunner テストへの QuickTest でパラメータ化された値の引き渡し

WinRunner テストに要求されるパラメータに固定の値を設定する代わりに，QuickTest データ・テーブル，ランダム・パラメータまたは環境パラメータに定義されているパラメータ値を WinRunner に引き渡せます。これらのパラメータ値を指定するには，**TSLTest.RunTestEx** ステートメントの**パラメータ**引数として，適切なステートメントを入力します。

たとえば，Windows ベースのフライト予約アプリケーションを対象に WinRunner テストを実行するとき，そのテストにフライトの乗客数と座席のクラスをパラメータ化するステートメントが組み込まれていたとします。その場合，最初のパラメータの値を QuickTest ランダム・パラメータ（1 から 100 までの数をランダムに生成）から，また座席のクラスの値を QuickTest データ・テーブルの Class というカラムから WinRunner テストに渡します。その場合，QuickTest の **TSLTest.RunTestEx** ステートメントは，たとえば次のようになります。

```
TSLTest.RunTestEx "D:¥test1", TRUE, FALSE, RandomNumber(1, 100) ,
DataTable("Class", dtGlobalSheet)
```

RandomNumber メソッド，**Environment** メソッド，および **DataTable** メソッドの構文と使用法の詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』の「ユーティリティ オブジェクト」を参照してください。

結果の表示

WinRunner テストへの呼び出しを実行するとき，コンピュータに WinRunner 7.6 以降がインストールされていれば，通常なら WinRunner の結果に含まれるイベントごとに，QuickTest の結果にノードが含まれます。WinRunner ステップに対応するノードを選択すると，右の表示枠に WinRunner テストのサマリと，選択したステップの詳細が表示されます。

詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 687 ページ「Quality Center プロジェクトへの不具合の自動送信」を参照してください。

WinRunner テストの設計と実行の詳細については，WinRunner のマニュアルを参照してください。

WinRunner 関数の呼び出し

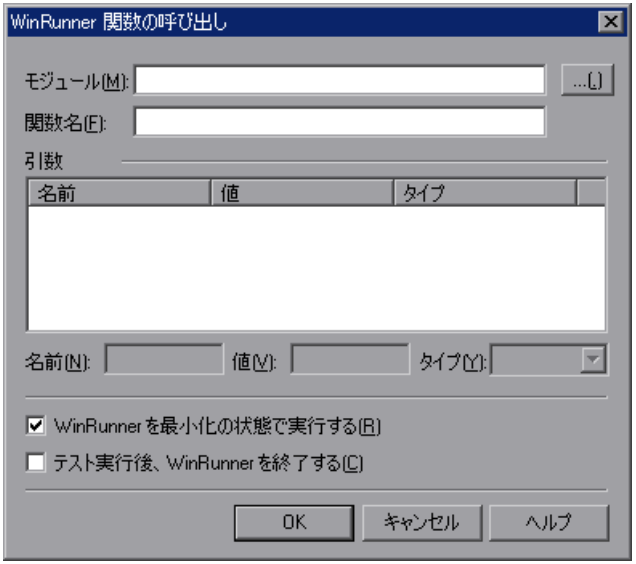
関数を呼び出すために，QuickTest から WinRunner に接続すると，WinRunner が起動され，コンパイル済みモジュールが読み込まれてから，関数が呼び出されます。これは，QuickTest で WinRunner のユーザ定義関数を使用する場合に便利です。

QuickTest から WinRunner 関数を呼び出すには，その関数と，その関数が含まれているコンパイル済みモジュールを指定します。

注：WinRunner 関数から返される値を QuickTest テストの中で取得することはできません。ただし，返された値は，結果の中で見ることができます。

WinRunner のコンパイル済みモジュールのユーザ定義関数を呼び出すには，次の手順を実行します。

1 ［**挿入**］＞［**WinRunner の呼び出し**］＞［**関数**］を選択します。［WinRunner 関数の呼び出し］ダイアログ・ボックスが表示されます。



2 ［**モジュール**］ボックスに，その関数を含んでいるコンパイル済みモジュールのパスを入力するか，参照ボタンを使って指定します。

Quality Center に接続していると，参照ボタンをクリックしたときに，Quality Center プロジェクトのコンパイル済みモジュールを選択するための，［Quality Center プロジェクトから WinRunner テストを開く］ダイアログ・ボックスが開きます。

WinRunner の TSL 関数を呼び出すには，任意のコンパイル済みモジュールのパスを入力します。

3 ［**関数名**］ボックスに，指定したコンパイル済みモジュールに定義されている関数の名前を入力するか，任意の WinRunner TSL 関数を入力します。

4 ［**引数**］ボックスの中をクリックします。コンピュータ上で WinRunner が開いている場合，［**引数**］ボックスには，選択した関数に対して定義されている引数の名前が表示されます。WinRunner が開いていなければ，［**引数**］ボックスには関数に対する最大 15 個までの引数を表す **p1** から **p15** までが表示されます。

5 次のように，**in** または **inout** 引数の値を入力します。

- ［**引数**］ボックスで引数を選択して強調表示にします。引数の名前が［**名前**］ボックスに表示されます。
- ［**タイプ**］ボックスで，正しい引数のタイプ（**in**/**out**/**inout**）を選択します。
- ［in］または［inout］の引数タイプを選択した場合は，［**値**］ボックスに新しい値を入力します。

注：QuickTest ランダム・パラメータまたは環境パラメータのパラメータ値や，QuickTest データ・テーブルのパラメータ値を，関数の **in** 引数または **inout** 引数として使用することもできます。それには引数の情報を **TSLTest.CallFuncEx** ステートメントに手作業で入力します。詳細については，「WinRunner 関数への QuickTest パラメータの引き渡し」を参照してください。

関数パラメータの詳細については，『**WinRunner ユーザーズ・ガイド**』を参照してください。

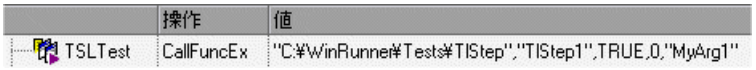
6 関数が実行されている間，WinRunner ウィンドウが表示されないようにするには，［**WinRunner を最小化の状態で実行する**］を選択します（このオプションのサポート対象は，WinRunner バージョン 7.6 以降です）。

7 WinRunner 関数を呼び出したステップが完了したときに WinRunner アプリケーションを終了させるには，［**テスト実行後、WinRunner を終了する**］を選択します（このオプションのサポート対象は，WinRunner バージョン 7.6 以降です）。

8 ［**OK**］をクリックして，ダイアログ・ボックスを閉じます。

QuickTest では，TSL 関数への呼び出しが次のように表示されます。

➤ キーワード・ビューには，WinRunner の **CallFuncEx** ステップが表示されます。以下に例を示します。



➤ エキスパート・ビューには，VBScript で書かれた **TSLTest.CallFuncEx** ステートメントが表示されます。例を次に示します。

```
CallFuncEx "C:¥WinRunner¥Tests¥TIStep","TIStep1",TRUE, 0, "MyArg1"
```

CallFuncEx メソッドの構文は，次のとおりです。

TSLTest.CallFuncEx モジュールのパス , 関数 , 最小化実行 , アプリケーション終了 [, 引数]

注：QuickTest 6.0 で作成したテストは，**CallFunc** メソッドを使用して WinRunner テストを呼び出している場合があります。このメソッドは，構文が少し異なります。このメソッドでもテストは正常に実行されます。ただし，WinRunner 7.6 以降を使用している場合には，**CallFuncEx** メソッド（および対応する引数の構文）を使用するようにテストを更新することをお勧めします。これらのメソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

CallFuncEx メソッドとサンプルの利用方法の詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

WinRunner 関数，関数引数，および WinRunner のコンパイル済みモジュールの詳細については，『**WinRunner ユーザーズ・ガイド**』および『**WinRunner TSL リファレンス・ガイド**』を参照してください。

WinRunner 関数への QuickTest パラメータの引き渡し

WinRunner 関数の in 引数や inout 引数に固定した値を設定する代わりに，QuickTest ランダム・パラメータや環境パラメータで定義されたパラメータ値，または QuickTest データ・テーブルで定義されたパラメータ値を WinRunner に引き渡すように，QuickTest に指示できます。これらのパラメータを指定するには，**TSLTest.CallFuncEx** ステートメントの **TSLTest.CallFunc** 引数として，適切なステートメントを入力します。

たとえば WinRunner を使って，アプリケーションを起動し，そのアプリケーションにユーザ名とパスワードを入力するユーザ定義関数を作成するとします。

そのためには，ユーザ名とパスワードの値を QuickTest データ・テーブルの FlightUserName 列と FlightPwd 列から WinRunner に引き渡すよう，QuickTest に指示します。その場合，QuickTest の **TSLTest.CallFuncEx** ステートメントは，たとえば次のようになります。

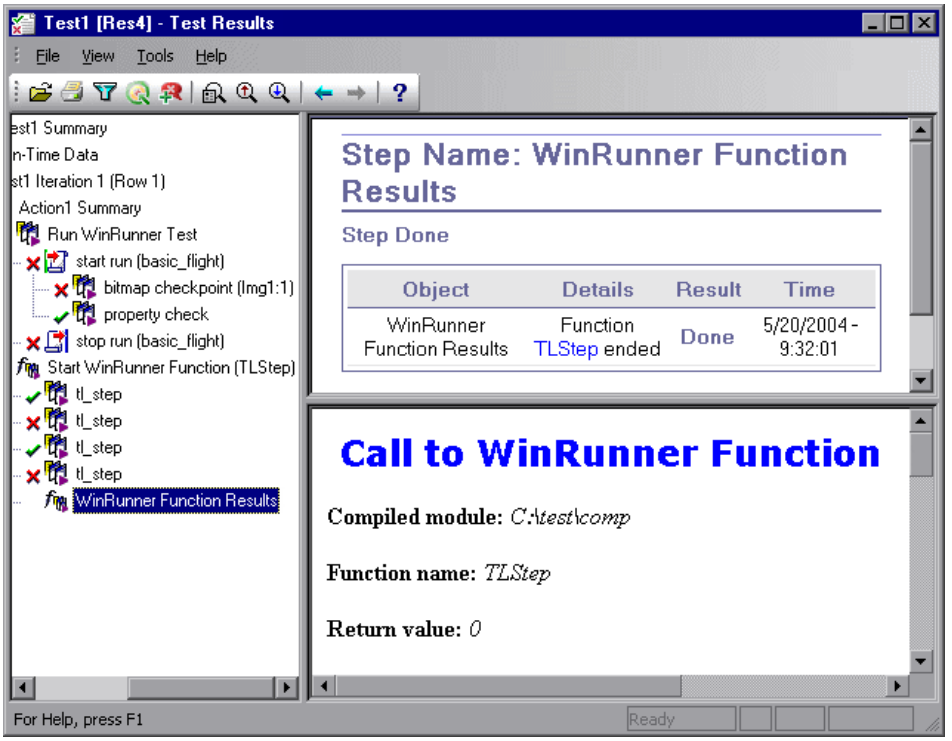
```
TSLTest.CallFuncEx "D:¥flightfuncs", "run_flight", TRUE, FALSE,
DataTable("FlightUserName", dtGlobalSheet), DataTable("FlightPwd",
dtGlobalSheet)
```

RandomNumber メソッド，**Environment** メソッド，および **DataTable** メソッドの構文および使用方法の詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』の「ユーティリティ オブジェクト」を参照してください。

結果の表示

QuickTest から WinRunner 7.6 以降を使用して WinRunner 関数を実行した後，関数呼び出しの結果を表示することができます。QuickTest の［テスト結果］ウィンドウに，WinRunner 関数の開始と WinRunner 関数の結果が表示されます。呼び出した関数に **report_msg** または **tl_step** などのイベントが含まれている場合，それらのイベントの結果に関する情報も含まれます。

結果ツリーの［**WinRunner Function Results**］を選択して強調表示にすると，関数の戻り値と関数への呼び出しに関する追加情報が表示されます。



WinRunner 関数およびコンパイル済みモジュールを使用した作業の詳細については，WinRunner のマニュアルを参照してください。

第 14 章

Quality Center を使用した作業

アプリケーションの包括的なテストを確実に行うためには，通常，多くのテストを作成して実行する必要があります。Mercury Quality Center は，品質の集中管理ソリューション（以前の TestDirector）であり，テスト・プロセスの整理と管理に役立ちます。

注：本章に含まれる Quality Center の説明は，Quality Center と TestDirector の現在サポートされているすべてのバージョンに適用されます。Quality Center と TestDirector のサポートされているバージョンについては，「**QuickTest Professional 最初にお読みください**」の一覧を参照してください。

本章では，次の項目について説明します。

- Quality Center を使用した作業について
- Quality Center との接続と接続解除
- Quality Center プロジェクトへのテストの保存
- Quality Center プロジェクトのテストを開く
- テンプレート・テストを使用した作業
- Quality Center プロジェクトに格納されているテストの QuickTest からの実行
- QuickTest でのテストのバージョン管理
- Quality Center テストの実行に関する設定

Quality Center を使用した作業について

QuickTest は Mercury の品質集中管理ソリューションである Quality Center と統合できます。Quality Center は，アプリケーションの機能のあらゆる側面を網羅するあらゆる種類のテスト（QuickTest テスト，ビジネス・プロセス・テスト，手動テスト，他の Mercury 製品を使用して作成したテストなど）で構成されるプロジェクトの保守を支援します。プロジェクトの各テストは，アプリケーションの特定のテスト要件を満たすように設計されています。プロジェクトの目標を達成するためには，プロジェクトのテストを個別のグループにまとめます。

Quality Center には，テストのスケジュール設定と実行，結果の収集，結果の分析，テストのバージョン管理などを行うための直感的かつ効率的な手段が用意されています。不具合を追跡するためのシステムも備わっているため，不具合の検出から解決に至るまで，不具合を厳密に監視することができます。

Quality Center プロジェクトとは，テスト・プロセスに関係するデータを収集し，格納するためのデータベースです。QuickTest から Quality Center プロジェクトにアクセスするには，Quality Center がインストールされているローカルまたはリモートの Web サーバに接続する必要があります。QuickTest が Quality Center に接続されている場合は，テストを作成し，このテストを Quality Center プロジェクトに保存できます。テストの実行後には，Quality Center で結果を確認できます。

Quality Center で QuickTest を使用するには，次のアクセス許可が必要です。

- Quality Center キャッシュ・フォルダへの完全な読み取りおよび書き込み許可
- Quality Center インストール・フォルダの QuickTest アドインへの完全な読み取りおよび書き込み許可

Quality Center を使って作業をしている場合，Quality Center プロジェクトに添付されている外部ファイルをテストに関連付けることができます。すべてのテストまたは 1 つのテストに対して，外部ファイルを関連付けることができます。たとえば，新規テストのための標準リポジトリ・モードとして，共有オブジェクト・リポジトリ・モードを設定するとします。この場合，Quality Center に格納されている特定のオブジェクト・リポジトリを使用するように QuickTest に指定できます。

すべてのテストで外部ファイルを使用するように指定する方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 24 章「グローバル・テスト・オプションの設定」を参照してください。単独のテストで外部ファイルを使用するように指定する方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 25 章「個別のテストのオプションの設定」を参照してください。

不具合は，発生するたび自動的に，または手作業で QuickTest の［テスト結果］ウィンドウから Quality Center プロジェクトに直接報告できます。Quality Center プロジェクトへの自動または手作業による不具合の報告については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 686 ページ「実行セッション中に検出された不具合の送信」を参照してください。

Quality Center での作業の詳細については，『**Mercury Quality Center ユーザーズ・ガイド**』を参照してください。QuickTest と Quality Center の統合に関する最新情報およびヒントについては，「**QuickTest Professional 最初にお読みください**」（［**スタート**］メニューから［**QuickTest Professional**］プログラム・メニューを開き，［**Readme**］を選択します）を参照してください。

Quality Center との接続と接続解除

QuickTest および Quality Center の両方で作業する場合，QuickTest を Quality Center プロジェクトに接続できます。

テスト・プロセスのどの時点でも，Quality Center プロジェクトと QuickTest とを接続したり，接続を解除したりできます。ただし，Quality Center から QuickTest テストを開いているときや，QuickTest で Quality Center の共有リソース（共有オブジェクト・リポジトリやデータ・テーブル・ファイルなど）を使用しているときは，Quality Center と QuickTest との接続は解除しないでください。

注：Quality Center または TestDirector の現在サポートされている任意のバージョンに接続できます。Quality Center と TestDirector のサポートされているバージョンについては，「**QuickTest Professional 最初にお読みください**」の一覧を参照してください。詳細については，359 ページ「Quality Center 接続アドインを使用した作業」を参照してください。

Quality Center への QuickTest の接続

接続プロセスには 2 つの段階があります。まず，QuickTest をローカルまたはリモートの Quality Center サーバに接続します。このサーバによって，QuickTest と Quality Center プロジェクトの間の接続が処理されます。

次に，ログインし，QuickTest のアクセス対象となるプロジェクトを選択します。このプロジェクトには，テスト対象の Web サイトまたはアプリケーションに関するテスト・セッションと実行セッションの情報が保存されます。Quality Center プロジェクトは，パスワードで保護されるため，ユーザ名とパスワードを指定する必要があります。

QuickTest を Quality Center サーバに接続するには，次の手順を実行します。

1 ［**ファイル**］＞［**Quality Center への接続**］を選択するか，［**Quality Center への接続**］ツールバー・ボタンをクリックします。［**Quality Center への接続 - サーバへの接続**］ダイアログ・ボックスが表示されます。

2 ［**サーバ URL**］ボックスに，Quality Center がインストールされている Web サーバの URL アドレスを入力します。

注：ローカル・エリア・ネットワーク（LAN）または広域ネットワーク（WAN）を介してアクセス可能な Quality Center サーバを選択できます。

---

3 次回の QuickTest の起動時に，Quality Center サーバに自動的に再接続するには，［**起動時にサーバに再接続する**］チェック・ボックスを選択します。

4 ［**接続**］をクリックします。

接続手順の第 2 段階は，接続先のサーバのバージョンによって異なります。該当する項を参照してください。

➤ 351 ページ「8.x サーバを使用したプロジェクトへの接続」

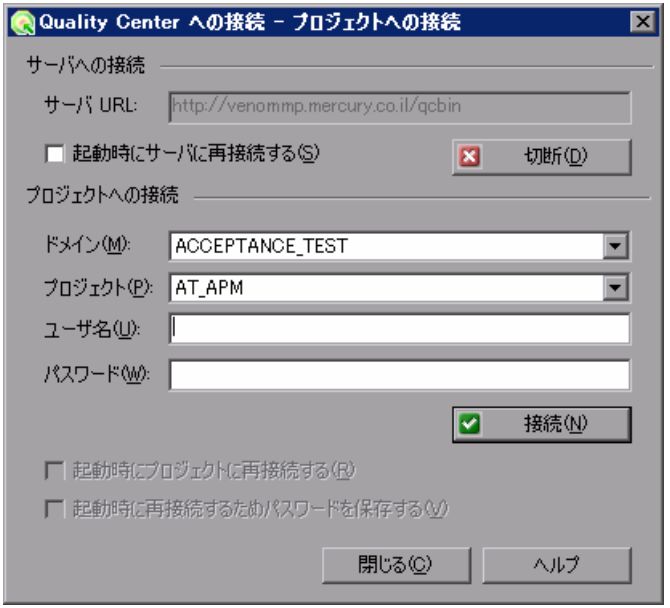
➤ 354 ページ「9.0 サーバを使用したプロジェクトへの接続」

8.x サーバを使用したプロジェクトへの接続

TestDirector 8.0 Service Pack 2 サーバまたは Mercury Quality Center 8.2 Service Pack 1 サーバに接続した場合は，接続先のドメインとプロジェクトを指定し，そのプロジェクトにログインします。

8.x サーバを使用したプロジェクトに接続するには，次の手順を実行します。

1 TestDirector 8.0 Service Pack 2 または Mercury Quality Center 8.2 Service Pack 1 にあるプロジェクトに接続した場合は，［Quality Center **への接続** - プロジェクトへの接続］ダイアログ・ボックスが開きます。



サーバの名前が読み取り専用形式で［サーバ URL］ボックスに表示されます。

2 ［**ドメイン**］ボックスで，Quality Center プロジェクトが保存されているドメインを選択します。

3 ［**プロジェクト**］ボックスで，作業対象のプロジェクトを選択します。

注：アクセス権を持っていないプロジェクトを選択した場合は，［**接続**］をクリックしたときに通知が表示されます。

---

4 ［**ユーザ名**］ボックスに，選択したプロジェクトを開くためのユーザ名を入力します。

5 ［**パスワード**］ボックスに，選択したプロジェクトのパスワードを入力します。

6 ［**接続**］をクリックして，選択したプロジェクトに QuickTest を接続します。

選択したプロジェクトへの接続が確立されると，［**プロジェクトへの接続**］領域内のフィールドが読み取り専用形式で表示されます。

7 次回の QuickTest の起動時に，Quality Center サーバに自動的に再接続するには，［**起動時にサーバに再接続する**］チェック・ボックスを選択します。

8 ［**起動時にサーバに再接続する**］チェック・ボックスをオンにすると，［**起動時にプロジェクトに再接続する**］チェック・ボックスが有効になります。起動時に，選択したプロジェクトに自動的に接続するには，［**起動時にプロジェクトに再接続する**］チェック・ボックスを選択します。

9 ［**起動時にプロジェクトに再接続する**］チェック・ボックスをオンにすると，［**起動時に再接続するためパスワードを保存する**］チェック・ボックスが有効になります。起動時に再接続するためのパスワードを保存するには，この［**起動時に再接続するためパスワードを保存する**］チェック・ボックスを選択します。

パスワードを保存しないと，起動時に QuickTest が Quality Center に接続するときに，パスワードを入力するよう求められます。

10 ［**閉じる**］をクリックし，［Quality Center への接続 - プロジェクトへの接続］ダイアログ・ボックスを閉じます。ステータス・バーに，QuickTest が Quality Center プロジェクトに接続されていることを示す Quality Center アイコンが表示されます。

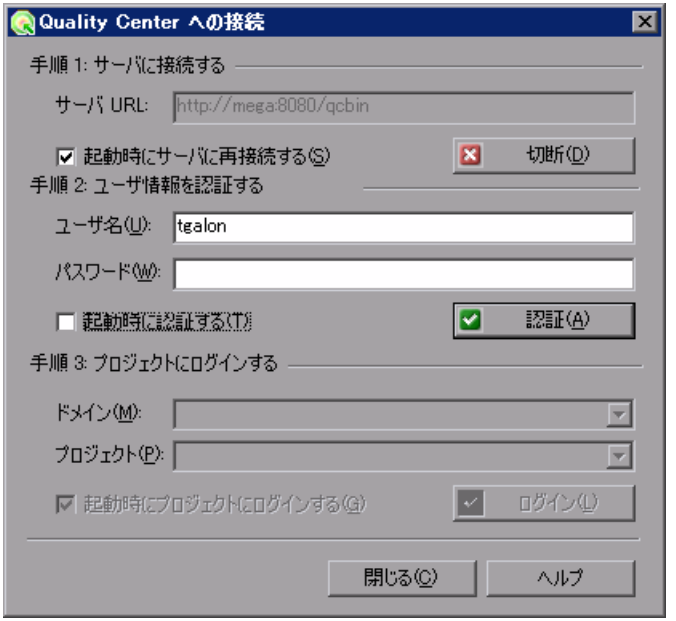
ヒント：現在の Quality Center への接続を表示するには，ステータス・バーの［**Quality Center**］アイコンをポイントします。QuickTest が接続されている Quality Center サーバ名とプロジェクトがツールチップに表示されます。［Quality Center への接続］ダイアログ・ボックスを開くには，**Quality Center** アイコンをダブルクリックします。

9.0 サーバを使用したプロジェクトへの接続

Mercury Quality Center 9.0 サーバに接続した場合は，接続先のドメインとプロジェクトを指定し，そのプロジェクトにログインします。

9.0 サーバを使用したプロジェクトに接続するには，次の手順を実行します。

1 Quality Center 9.0 にあるプロジェクトに接続した場合は，［Quality Center への接続］ダイアログ・ボックスが開きます。



Quality Center サーバの名前が読み取り専用形式で［サーバ URL］ボックスに表示されます。

2 ［**ユーザ名**］ボックスに，Quality Center ユーザ名を入力します。

3 ［**パスワード**］ボックスに，Quality Center のパスワードを入力します。

4 ［**認証**］をクリックし，Quality Center サーバに対してユーザ情報を認証させます。

ユーザ情報の認証が完了すると，［**ユーザ情報を認証する**］領域の各フィールドが読み取り専用形式で表示されます。［**認証**］ボタンが［**ユーザを変更**］ボタンに変わります。

ヒント：同じ Quality Center サーバに別のユーザ名でログインするには，［**ユーザの変更**］をクリックし，新しいユーザ名とパスワードを入力して，再び［**認証**］をクリックします。

---

5 ［**ドメイン**］ボックスで，Quality Center プロジェクトが保存されているドメインを選択します。ユーザが接続の権限を持っているドメインのみ表示されます。

6 ［**プロジェクト**］ボックスで，作業対象のプロジェクトを選択します。ユーザが接続の権限を持っているプロジェクトのみ表示されます。

7 ［**ログイン**］をクリックします。

8 次回の QuickTest の起動時に，Quality Center サーバに自動的に再接続するには，［**起動時にサーバに再接続する**］チェック・ボックスを選択します。

9 ［**起動時にサーバに再接続する**］チェック・ボックスをオンにすると，［**起動時に認証する**］チェック・ボックスが有効になります。次回の QuickTest の起動時に，ユーザ情報を自動的に認証させるには，［**起動時に認証する**］チェック・ボックスを選択します。

10 ［**起動時に認証する**］チェック・ボックスをオンにすると，［**起動時にプロジェクトにログインする**］チェック・ボックスが有効になります。起動時に，選択したプロジェクトにログインするには，［**起動時にプロジェクトにログインする**］チェック・ボックスを選択します。

11 ［**閉じる**］をクリックし，［Quality Center への接続］ダイアログ・ボックスを閉じます。ステータス・バーに，QuickTest が Quality Center プロジェクトに接続されていることを示す Quality Center アイコンが表示されます。

ヒント：現在の Quality Center への接続を表示するには，ステータス・バーの［**Quality Center**］アイコンをポイントします。QuickTest が接続されている Quality Center サーバ名とプロジェクトがツールチップに表示されます。［Quality Center への接続］ダイアログ・ボックスを開くには，**Quality Center** アイコンをダブルクリックします。

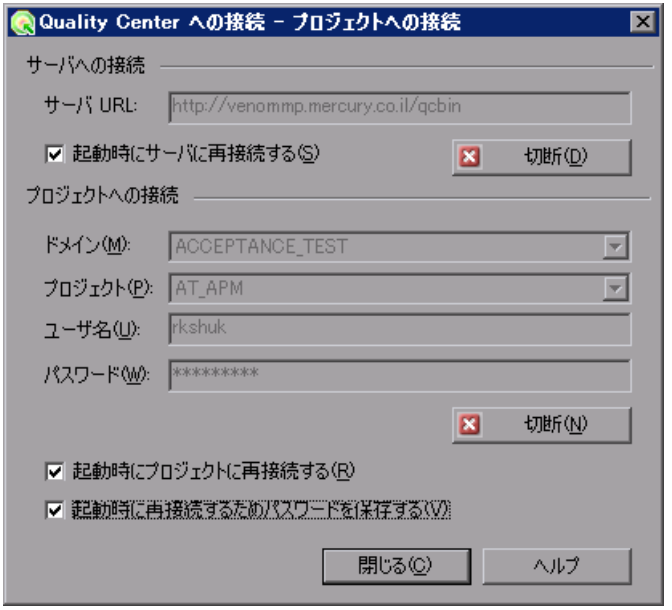
Quality Center への QuickTest の接続の解除

Quality Center プロジェクトまたはサーバへの接続を解除することができます。最初にプロジェクトへの接続を解除せずに，Quality Center サーバへの QuickTest の接続を解除すると，QuickTest から当該プロジェクト・データベースへの接続は自動的に解除されます。

注：Quality Center への接続を解除するときに Quality Center テスト，または共有ファイル（共有オブジェクト・リポジトリやデータ・テーブル・ファイルなど）が開いていると，それらは QuickTest によって閉じられます。

8.x サーバへの QuickTest の接続を解除するには，次の手順を実行します。

1 ［**ファイル**］＞［**Quality Center への接続**］を選択するか，［**Quality Center への接続**］ツールバー・ボタンをクリックします。［Quality Center への接続 - プロジェクトへの接続］ダイアログ・ボックスが表示されます。



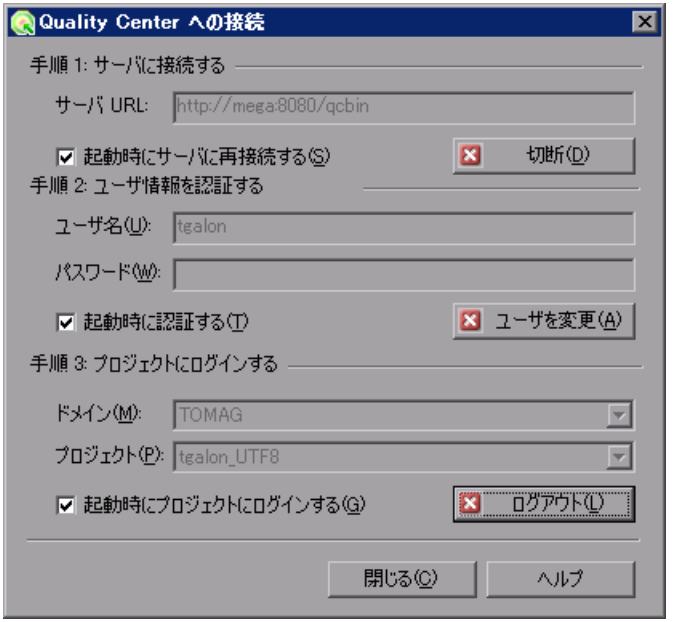
2 選択したプロジェクトへの QuickTest の接続を解除するには，［**プロジェクトへの接続**］領域で［**切断**］をクリックします。

3 選択した Quality Center サーバへの QuickTest の接続を解除するには，［**サーバへの接続**］領域の中で［**切断**］をクリックします。

4 ［**閉じる**］をクリックし，［Quality Center への接続 - プロジェクトへの接続］ダイアログ・ボックスを閉じます。

9.0 Web サーバへの QuickTest の接続を解除するには，次の手順を実行します。

1 ［**ファイル**］＞［**Quality Center への接続**］を選択するか，［**Quality Center への接続**］ツールバー・ボタンをクリックします。［Quality Center への接続］ダイアログ・ボックスが表示されます。



2 選択したプロジェクトへの QuickTest の接続を解除するには，［**手順 3：プロジェクトにログインする**］領域で［**ログアウト**］をクリックします。

3 選択した Quality Center サーバへの QuickTest の接続を解除するには，［**手順 1：サーバに接続する**］領域で［**切断**］をクリックします。

ヒント：同じ Quality Center サーバに別のユーザ名でログインするには，［**ユーザを変更**］をクリックし，新しいユーザ名とパスワードを入力して，再び［**認証**］をクリックします。

---

4 ［**閉じる**］をクリックし，［Quality Center への接続］ダイアログ・ボックスを閉じます。

Quality Center 接続アドインを使用した作業

Quality Center に接続するには，Quality Center 接続アドインをインストールする必要があります。このアドインは，［Quality Center への接続］ダイアログ・ボックスを使用して Quality Center に接続するときに自動的にインストールされます。

現在コンピュータにインストールされている Quality Center 接続アドインのバージョンを表示するには，［**ヘルプ**］＞［**QuickTest Professional のバージョン情報**］を選択し，［**製品情報**］ボタンをクリックします。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 57 ページ「製品情報の表示」を参照してください。

Quality Center 接続アドインを手作業でインストールするには，Quality Center アドインのページ（Quality Center のメイン画面から利用できます）から［**Mercury Quality Center 接続**］を選択します。

注：また Quality Center 接続アドインを使えば，Quality Center がコンピュータにインストールされていなくても（オートメーション・プログラム経由で）TDOTA 機能にアクセスできます。オートメーション・プログラムを使った TDOTA へのアクセスの詳細については，『**QuickTest オートメーション・オブジェクト・モデル・リファレンス**』を参照してください。

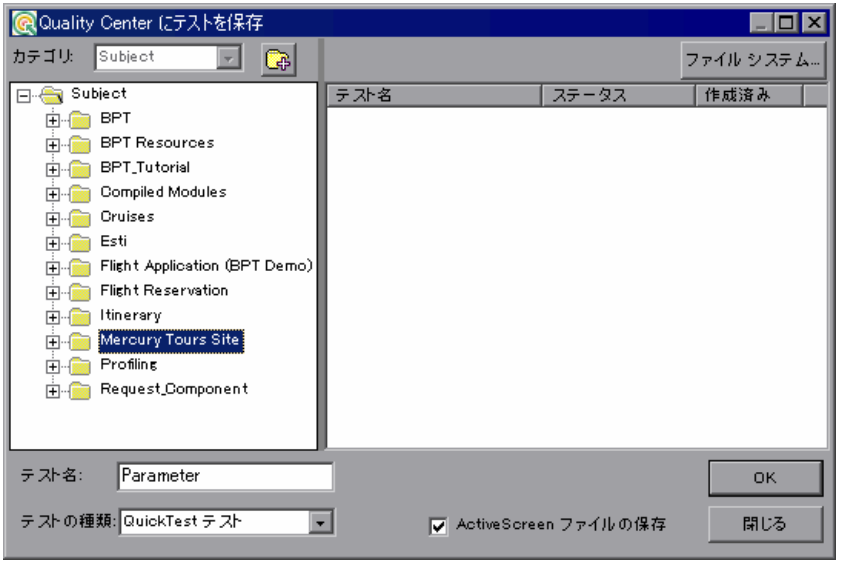
Quality Center プロジェクトへのテストの保存

QuickTest を Quality Center プロジェクトに接続すると，QuickTest で新しいテストを作成し，そのテストをプロジェクトに直接保存することができます。テストを保存するには，テストに分かりやすい名前を付け，テスト計画ツリーの対応するサブジェクトに関連付けます。これにより，各サブジェクトに作成したテストを容易に追跡し，テストの計画と作成の進行状況を素早く確認することができます。

テストを Quality Center プロジェクトに保存するには，次の手順を実行します。

1 Quality Center サーバおよびプロジェクトに接続します。詳細については，350 ページ「Quality Center への QuickTest の接続」を参照してください。

2 QuickTest で［**保存**］をクリックするか，［**ファイル**］＞［**保存**］を選択して，テストを保存します。［Quality Center にテストを保存］ダイアログ・ボックスが開き，テスト計画ツリーが表示されます。



［Quality Center にテストを保存］ダイアログ・ボックスは，QuickTest が Quality Center プロジェクトに接続されている場合にだけ開きます。

テストをファイル・システムに直接保存するには，［**ファイル システム**］ボタンをクリックし，［テストの保存］ダイアログ・ボックスを開きます。［テストの保存］ダイアログ・ボックスで［**Quality Center**］ボタンをクリックすると，［Quality Center にテストを保存］ダイアログ・ボックスに戻ることができます。

3 テスト計画ツリーの中で，該当するサブジェクト・フォルダを選択します。ツリーを展開して下位レベルを表示するには，閉じているフォルダをダブルクリックします。下位レベルを折りたたむには，開いているフォルダをダブルクリックします。

4 ［**テスト名**］ボックスにテストの名前を入力します。テストを識別しやすい，分かりやすい名前を付けます。テスト名はパスを含めて 220 文字を超えることはできません。スペースで始めたり，スペースで終わったりすることはできず，次の文字を含めることはできません。
¥ / : * ? " < > | % ' ;

5 テストと一緒に ActiveScreen ファイルを保存する場合，［**ActiveScreen ファイルを保存する**］が選択されていることを確認します。このチェック・ボックスをクリアすると，ActiveScreen ファイルが削除されるため，ActiveScreen オプションを使ったテストの編集ができなくなります。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 106 ページ「テストの保存」を参照してください。

6 ［**OK**］をクリックし，テストを保存してダイアログ・ボックスを閉じます。QuickTest がテストを保存している間は，ステータス・バーのテキストが変わります。

次回 Quality Center 起動時に，新しいテストが Quality Center のテスト計画ツリーに表示されます。詳細については，『**Mercury Quality Center ユーザーズ・ガイド**』を参照してください。

Quality Center プロジェクトのテストを開く

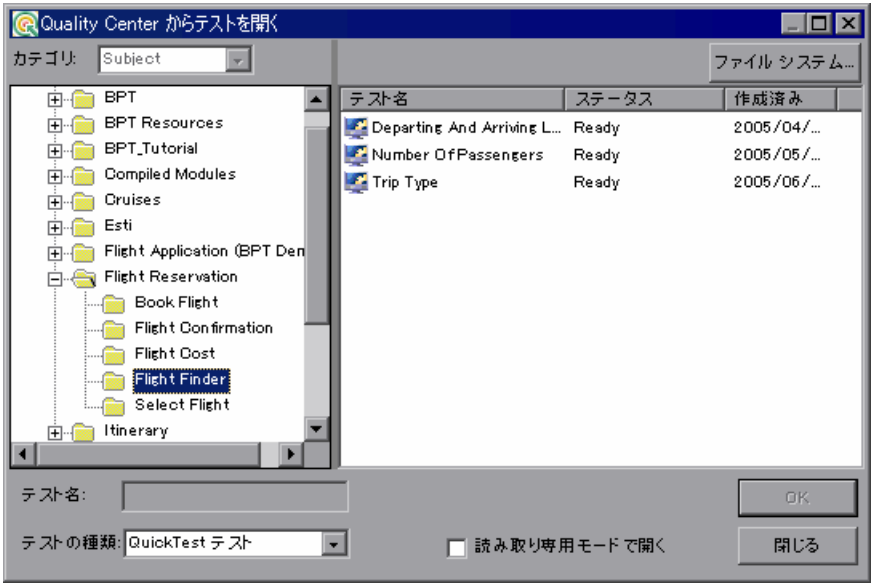
QuickTest を Quality Center プロジェクトに接続すると，Quality Center プロジェクトに含まれている QuickTest テストを開くことができます。テストは，ファイル・システムの実際の位置からではなく，テスト計画ツリーでの位置から見つけます。［**ファイル**］メニューの最近使用したテストのリストからテストを開くこともできます。

バージョン・コントロール・サポートの Quality Center プロジェクトのテストを開くと，アイコンはテストのバージョン・コントロール・ステータスを示します。

Quality Center プロジェクトからテストを開くには，次の手順を実行します。

1 Quality Center サーバおよびプロジェクトに接続します。詳細については，350 ページ「Quality Center への QuickTest の接続」を参照してください。

2 QuickTest で，［**開く**］をクリックするか，［**ファイル**］＞［**開く**］を選択し，テストを開きます。［Quality Center からテストを開く］ダイアログ・ボックスが開き，テスト計画ツリーが表示されます。



［Quality Center からテストを開く］ダイアログ・ボックスは，QuickTest が Quality Center プロジェクトに接続されている場合にだけ表示されます。

注：Quality Center に接続している状態でテストをファイル・システムから直接開くには，［**ファイル システム**］ボタンをクリックし，［テストを開く］ダイアログ・ボックスを表示します（［テストを開く］ダイアログ・ボックスで［**Quality Center**］ボタンをクリックすると，［Quality Center からテストを開く］ダイアログ・ボックスに戻ることができます）。

---

3 テスト計画ツリーで，関連するサブジェクトをクリックします。ツリーを展開して下位レベルを表示するには，閉じているフォルダをダブルクリックします。ツリーを折りたたむには，開いているフォルダをダブルクリックします。

サブジェクトを選択すると，そのサブジェクトに属するテストが［Quality Center からテストを開く］ダイアログ・ボックスの右表示枠に表示されます。

- テストがバージョン・コントロール・サポート付きの Quality Center プロジェクトに格納されると，［**テスト名**］の横に表示されるアイコンは，テストのバージョン・コントロール・ステータスを示します。詳細については，365 ページ「バージョン・コントロールがサポートされている Quality Center プロジェクトからテストを開く」を参照してください。
- ［**テスト名**］カラムには，選択したサブジェクトに属するテストの名前が一覧表示されます。
- ［**ステータス**］カラムは，各テストが **Design**（設計中）段階かテスト実行のための **Ready**（準備完了）段階かを示します。標準設定では，QuickTest から Quality Center プロジェクトに保存されたテストは **Design** と表示されます。ステータスを変更できるのは Quality Center クライアントだけです。
- ［**作成済み**］カラムは，各テストが作成された日付を示します。

4 ［**テスト名**］リストでテストを選択します。テストが読み取り専用の［**テスト名**］ボックスに表示されます。

5 読み取り専用モードでテストを開くには，［**読み取り専用モードで開く**］チェック・ボックスを選択します。

6 ［**OK**］をクリックし，テストを開きます。

QuickTest がテストをダウンロードして開くのに応じて，実行される操作がステータス・バーに表示されます。

テストを開くと，QuickTest タイトル・バーに，「[Quality Center]」という文字列と，サブジェクトのフルパスおよびテスト名が表示されます。例を次に示します。

[Quality Center] Subject¥System¥qa_test1

次の場合，テストは読み取り専用モードで開きます。

- ［**読み取り専用モードで開く**］を選択した場合
- 現在（バージョン・コントロールがサポートされているプロジェクトの）Quality Center バージョン・コントロール・データベースにチェック・インされているテストを開いた場合
- 現在（バージョン・コントロールがサポートされているプロジェクトの）ほかのユーザがチェック・アウトしているテストを開いた場合

詳細については，365 ページ「バージョン・コントロールがサポートされている Quality Center プロジェクトからテストを開く」を参照してください。

最近使用したテストのリストからテストを開く

［ファイル］メニューの最近使用したファイルのリストから Quality Center テストを開くことができます。Quality Center プロジェクトのテストを選択したときに，現在 QuickTest が Quality Center に接続されていないか，テストに対応する正しいプロジェクトに接続されていないと，［Quality Center プロジェクトへの接続］ダイアログ・ボックスが表示され，正しいサーバ，プロジェクト，および前回当該コンピュータでテストを開いたユーザの名前が表示されます。

プロジェクトにログインし，［**OK**］をクリックします。

また，前回異なる Quality Center ユーザ名を使用して編集したテストを開くことを選択した場合も，［Quality Center プロジェクトへの接続］ダイアログ・ボックスが開きます。表示されているユーザ名を使ってログインすることも，［**キャンセル**］をクリックして現在のユーザ名でログインしたままでいることもできます。

バージョン・コントロールがサポートされている Quality Center プロジェクトからテストを開く

［**開く**］ツールバー・ボタンをクリックするか，［**ファイル**］＞［**開く**］＞［**テスト**］を選択してバージョン・コントロールがサポートされている Quality Center プロジェクトからテストを開くと，［Quality Center プロジェクトから QuickTest テストを開く］ダイアログ・ボックスには，選択したサブジェクトの各テストのバージョン・コントロール・ステータスを示すアイコンが表示されます。

バージョン・コントロールがサポートされている Quality Center プロジェクトからテストを開くと，テストは現在のバージョン・コントロール・ステータスに応じて，読み書きモードまたは読み取り専用モードで開きます。

次の表に，バージョン・コントロール・ステータスのアイコンと各ステータスのオープン・モードを示します。

アイコン	詳細	オープン・モード
＜なし＞	対象テストは，現在バージョン・コントロール・データベースにチェック・インされています。	読み取り専用
	対象テストは，現在ユーザ自身によってチェック・アウトされています。	読み取り / 書き込み
	対象テストは，現在ほかのユーザによってチェック・アウトされています。	読み取り専用
	現在古いバージョンのテストをユーザのコンピュータで開いています。	現状のまま

バージョン・コントロールをサポートする Quality Center プロジェクトに格納されているテストを使った作業の詳細については，376 ページ「QuickTest でのテストのバージョン管理」を参照してください。

テンプレート・テストを使用した作業

「**テンプレート・テスト**」は，Quality Center で作成されるすべての QuickTest テストの土台となります。テンプレート・テストは，標準のテスト設定を含んだ QuickTest テストです。たとえば，テンプレート・テストでは，QuickTest アドイン，関連付けられる関数ライブラリ，テストに関連付けられる回復シナリオなどを指定します。これらのテスト設定は QuickTest の［テストの設定］ダイアログ・ボックス（［**ファイル**］＞［**設定**］）で変更できます。

標準のテスト設定以外に，テンプレート・テストには Quality Center で作成されるすべての新規の QuickTest テストに追加したい，任意のコメントやステップを含めることもできます。たとえば，テンプレート・テストに関連付けられているアドインの種類をユーザに伝えるコメントを追加したり，すべてのテストの開始時に特定の Web ページまたはアプリケーションを開くステップを追加したりできます。テンプレート・テストに追加したステップやコメントは，そのテンプレート・テストを土台として Quality Center で作成されるすべての新規のテストに追加されます。

Quality Center 用の QuickTest Professional アドインをインストールするとき，Quality Center クライアントごとに標準設定のテンプレート・テストがインストールされます。この標準設定のテンプレート・テストに変更を加えたり，さまざまなテスト設定が関連付けられた他のテンプレート・テストを作成したりできます。

標準設定のテンプレート・テストに変更を加える場合は，変更を加えたテンプレート・テストを，Quality Center ユーザがテストを作成する可能性のあるすべてのコンピュータ上の対応する **Templates** フォルダにコピーすることをお勧めします。こうすることでローカルのテンプレート・テストが上書きされ，すべての Quality Center ユーザが（それぞれの標準設定のローカル・コピーではなく）共通のテンプレート・テストを土台として QuickTest テストを作成するようになります。詳細については，次の「標準設定のテンプレート・テストを使用した作業」を参照してください。

テンプレート・テストはすべて（Quality Center クライアントに置かれる標準設定のテンプレート・テストを除き）Quality Center プロジェクトに保存されます。これらのテンプレート・テストについては各ユーザのローカル・コンピュータにコピーする必要がありません。これにより，ユーザは必要な場合に各自のローカルのテンプレート・テストをカスタマイズできる一方で，グローバルに管理されたテンプレート・テストにもアクセスできます。詳細については，368 ページ「新しいテンプレート・テストを使用した作業」を参照してください。

特定のテンプレート・テストを土台とするテストが Quality Center から実行されると，テスト内の定義に従って，関連付けられているアドインが QuickTest によって自動的にロードされ，必要な設定が適用されます。

標準設定のテンプレート・テストを使用した作業

Quality Center 用の QuickTest アドインをインストールすると，サポートされているすべての QuickTest バージョンに対応した標準設定のテンプレート・テストが，コンピュータの **＜Quality Center 用 QuickTest アドイン・フォルダ＞¥bin¥Templates** フォルダ（たとえば C:¥Program Files¥Mercury Interactive¥QuickTest Add-in for Quality Center¥bin¥Templates¥Template90 など）にインストールされます。

Quality Center ユーザが Quality Center で QuickTest テストを新規に作成すると，ユーザが別のテンプレート・テストを選択しない限り，インストールされている QuickTest のバージョンに対応した標準設定のテンプレート・テストが自動的にテストに関連付けられます。詳細については，370 ページ「Quality Center での QuickTest テストの作成」を参照してください。

Quality Center 用の QuickTest アドインとともに標準設定でインストールされるテンプレート・テストには，変更を加えることができます。標準設定のテンプレート・テストはローカルにインストールされるため，テンプレート・テストに加えた変更は自分のコンピュータで（Quality Center クライアントを使用して）作成したテストにのみ適用されます。したがって，ユーザのグループで使用するテンプレート・テストに変更を加えたい場合は，変更を加えたテンプレート・テストをすべての Quality Center クライアント・コンピュータにコピーする必要があります。そのようにすることで，Quality Center で標準設定のテンプレート・テストを土台として作成されるすべての新規のテストで，同じ基本テスト設定が定義されるようになります。

あるいは，以降の各項の説明に従って，テンプレート・テストを新規に作成することもできます。

標準設定のテンプレート・テストを Quality Center 内の新しい QuickTest に適用する方法の詳細については，370 ページ「Quality Center での QuickTest テストの作成」を参照してください。

新しいテンプレート・テストを使用した作業

新しいテンプレート・テストを作成すると，それらは Quality Center プロジェクトに格納され，その Quality Center プロジェクトで作成される新しい QuickTest テストの土台として使用できるようになります。

テンプレート・テストを複数作成し，それぞれを特定のテスト目的に使用することができます。たとえば，ActiveX コントロールを使用する Web アプリケーションをテストする QuickTest テスト用のテンプレート・テストを作成し，それとは別に，標準の Windows アプリケーションをテストする QuickTest テスト用のテンプレート・テストを作成できます。この場合，前者のテンプレート・テストには ActiveX と Web のアドインを関連付けます。後者のテンプレート・テストには，アドインを一切関連付けずに，記録と実行の対象となる Windows アプリケーションを指定します。このほか，必要に応じてそれぞれのテンプレート・テストのテスト設定に変更を加えることもできます。

作成した各テンプレート・テストは，ActiveX_Web_Addins_Template や Std_Windows_Template_Test のように，用途を明確に示す分かりやすい名前を付けて保存できます。そのようにすることで，ユーザは Quality Center で QuickTest テストを作成するときに適切なテンプレート・テストを選択できます。

注：特定の QuickTest アドインの関連付けを行うテンプレート・テストを定義するときは，テストを最終的に実行する QuickTest コンピュータに，そのアドインが実際にインストールされていることを確かめておきます。アドインがインストールされていないと，テストを実行したときに，QuickTest が必要なアドインをロードすることができず，テストが失敗することがあります。Quality Center からの QuickTest テストの実行の詳細については，Mercury Quality Center のマニュアルを参照してください。

テンプレート・テストの新規作成

テンプレート・テストを作成するには，まず，必要なテスト設定を持つ空のテストを QuickTest で作成します。次に，Quality Center プロジェクトのテスト計画モジュールで，QuickTest テストを参照し，それを「**テンプレート・テスト**」として保存します。

注：QuickTest でテストを保存するときは，用途を明確に示す分かりやすい名前を付けます。たとえば，テンプレート・テストを，ActiveX と Web のアドインを新しいテストに関連付ける目的で使用する場合，ActiveX_Web_Addins_Template という名前を付けることができます。

ヒント：Quality Center のテスト計画ツリー（テスト計画モジュール）で，テンプレート・テスト用の特別なフォルダを作成することもできます。このようにすれば，他のユーザが Quality Center で QuickTest テストを新規に作成するときに，関連するテンプレート・テストをすぐに見つけることができます。

テンプレート・テストを作成するには，次の手順を実行します。

QuickTest での作業：

1 必要なアドインをロードした状態で，QuickTest を開きます。QuickTest アドインのロードの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 799 ページ「QuickTest アドインのロード」を参照してください。

2 必要な設定を［テストの設定］ダイアログ・ボックス（［**ファイル**］＞［**設定**］）で定義します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 745 ページ「［テストの設定］ダイアログ・ボックスの使用」を参照してください。

3 このテンプレート・テストを土台とするすべてのテストにコメントやステップを追加したい場合は，それらを追加します。

4 ［**保存**］ボタンをクリックするか，［**ファイル**］＞［**保存**］を選択してテストを保存します。［Quality Center にテストを保存］ダイアログ・ボックスが表示されます。用途を明確に示す分かりやすい名前を付けて，テストを Quality Center プロジェクトに保存します。詳細については，360 ページ「Quality Center プロジェクトへのテストの保存」を参照してください。

Quality Center での作業：

5 Quality Center でプロジェクトを開き，サイドバーの［**テストの計画**］ボタンをクリックしてテスト計画モジュールを開き，手順 4 で保存したテストを参照します。

6 テストを右クリックし，［**テンプレート テストとしてマーク**］を選択します。テストがテンプレート・テストに変換されます。

7 必要に応じて，手順 1 から 6 を繰り返して追加のテンプレート・テストを作成します。

Quality Center での QuickTest テストの作成

Quality Center では，テスト計画モジュールで QuickTest テストを作成します。QuickTest テストを作成するときは，テンプレート・テストをそのテストに適用します。QuickTest クライアントに格納されている標準設定のテンプレート・テストを選択することも，Quality Center プロジェクトに保存されているテンプレート・テストを選択することもできます。

Quality Center プロジェクトにテンプレート・テストをまったく保存していない場合や，［テンプレート］ボックスで［**<なし>**］を選択した場合（361 ページに示した［テストの新規作成］ダイアログ・ボックスにあります），Quality Center では，Quality Center クライアントに「**Quality Center 用 QuickTest アドイン**」と一緒にインストールされたテンプレート・テストに定義されている設定が使用されます。詳細については，367 ページ「標準設定のテンプレート・テストを使用した作業」を参照してください。そうではなく，少なくとも 1 つのテンプレート・テストが Quality Center プロジェクトに保存されている場合は，QuickTest テストの新規作成時にはそのテンプレート・テストを選択できます。詳細については，368 ページ「新しいテンプレート・テストを使用した作業」を参照してください。

注：Quality Center で QuickTest テストを作成するときは，テストに関連付ける QuickTest アドインを指定するテンプレート・テストを選択する必要があります。アドインを指定するテンプレートを選択しないと，必要な QuickTest アドインが実行セッション中にロードされません。

新しい QuickTest テストでは，選択したテンプレート・テストに定義されているすべての設定が使用されます。テストが Quality Center から実行されたときは，［テストの実行］ダイアログ・ボックスに指定されている設定が QuickTest によって使用され，必要な QuickTest アドインが自動的にロードされます。

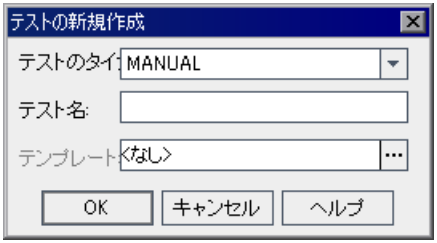
注：次の手順は，Quality Center でテンプレート・テストを使用してテストを作成する方法を示します。この手順は，使用する Quality Center のバージョンによって異なる場合があります。Quality Center でテストを新規に作成するための最新の手順については，『**Mercury Quality Center ユーザーズ・ガイド**』を参照してください。

Quality Center でテンプレート・テストを使用してテストを作成するには，次の手順を実行します。

1 Quality Center で，サイドバーの［**テストの計画**］ボタンをクリックして，テスト計画モジュールを表示します。

2 テスト計画ツリーで，フォルダを選択します。

3 ［**テストの新規作成**］ボタンをクリックするか，［**テスト**］＞［**テストの新規作成**］を選択します。［テストの新規作成］ダイアログ・ボックスが表示されます。



注：Quality Center アドインまたは **QuickTest Professional アドイン**がコンピュータにインストールされている場合にのみ，［**テンプレート**］ボックスが表示されます。［**テンプレート**］ボックスが表示されない場合は，**Quality Center アドイン**を QuickTest Professional CD-ROM からインストールするか，または QuickTest Professional アドインを［Mercury Quality Center アドインの追加］ページ（［Mercury Quality Center オプション］から，または［ログイン ウィンドウ］＞［**アドイン ページ**］リンクから開きます）からインストールする必要があります。

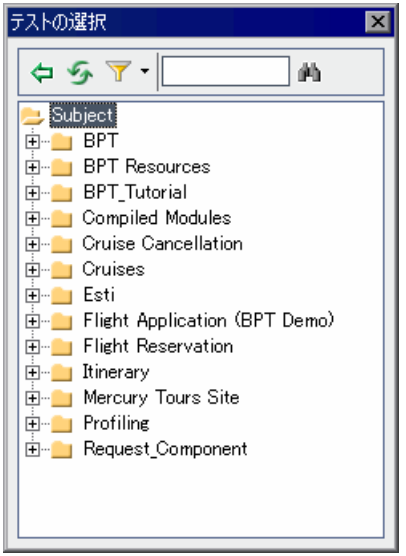
---

4 ［**テストのタイプ**］リストから，［**QUICKTEST_TEST**］を選択します。

5 ［**テスト名**］ボックスに，新しいテストの名前を英数字（および必要ならばアンダスコア）を使用して入力します。テスト名はパスを含めて 220 文字を超えることはできません。スペースで始めたり，スペースで終わったりすることはできず，次の文字を含めることはできません。
¥ / : * ? " < > | % ' ;

6 ［**テンプレート**］ボックスの参照ボタンをクリックします。［テストの選択］ダイアログ・ボックスが開きます。

7 テンプレート・テストが格納されているフォルダを展開します。



8 新しいテストの土台とするテンプレート・テストを選択し，［**追加**］ボタンをクリックします。［テストの選択］ダイアログ・ボックスが閉じ，選択したテンプレート・テストが［**テンプレート**］ボックス（［テストの新規作成］ダイアログ・ボックス内）に表示されます。

9 ［テストの新規作成］ダイアログ・ボックスで，［**OK**］をクリックします。テンプレート・テストに定義されているテスト設定を持つ，新しいテストが作成されます。

10 ［**OK**］をクリックして，［テストの新規作成］ダイアログ・ボックスを閉じます。新しいテストが，テスト計画ツリー内の選択したサブジェクト・フォルダの下に表示されます。

注 : ［必須フィールド］ダイアログ・ボックスが表示された場合は，必要な値を設定して［**OK**］をクリックします。詳細については，『**Mercury Quality Center 管理者ガイド**』を参照してください。

---

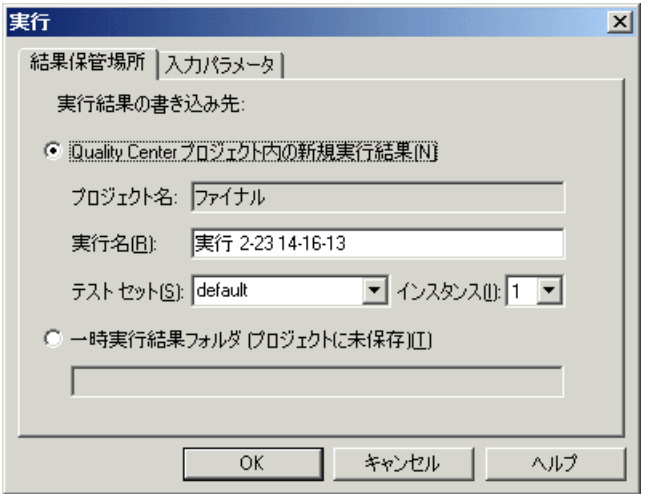
11 テストの作成を続けます。Quality Center でのテストの作成については，『**Mercury Quality Center ユーザーズ・ガイド**』を参照してください。

Quality Center プロジェクトに格納されているテストの QuickTest からの実行

QuickTest では，Quality Center プロジェクトからテストを実行し，実行結果をプロジェクトに保存できます。実行結果を保存するには，結果を格納する実行セッションとテスト・セットの名前を指定します。

テストの実行結果を Quality Center プロジェクトに保存するには，次の手順を実行します。

1 QuickTest で，［**実行**］ボタンをクリックするか，［**オートメーション**］＞［**実行**］を選択します。［実行］ダイアログ・ボックスが表示されます。



2 ［**プロジェクト名**］ボックスには現在接続している Quality Center プロジェクトが表示されます。

テストの実行結果を Quality Center プロジェクトに保存するには，標準設定の**実行名**を受け入れるか，別の名前をボックスに入力します。

3 標準設定の**テスト・セット**を受け入れるか，別のテスト・セットを参照して，選択します。

4 テスト・セットにテストのインスタンスが複数ある場合には，結果を保存するテストのインスタンスを［**インスタンス**］ボックスに指定します。

注：「**テスト・セット**」とは，特定のテスト目標を達成するために選択されたテストで構成されたグループです。たとえば，アプリケーションのユーザ・インタフェースや，負荷をかけた状態のアプリケーションのパフォーマンスをテストするテスト・セットを作成できます。テスト・セットの定義は，Quality Center のテスト実行モードで行います。詳細については，Quality Center のマニュアルを参照してください。

テストを実行し，前のテスト実行の結果を上書きするには，［**一時実行結果フォルダ（プロジェクトに未保存）**］オプションを選択します。

注：QuickTest では，すべてのテストの一時的なテスト実行結果が**＜システム・ドライブ＞:¥Temp¥TempResults** に格納されます。［**一時実行結果フォルダ（プロジェクトに未保存）**］オプションのテキスト・ボックスに表示されるパスは読み取り専用のため，変更できません。

---

5 ［**OK**］をクリックします。［実行］ダイアログ・ボックスが閉じ，QuickTest によってテストの実行が開始されます。QuickTest によるテストの実行が進行すると同時に，キーワード・ビューの各ステップが強調表示されます。

テストの実行が停止すると，［オプション］ダイアログ・ボックスの［実行］タブで［**テスト実行終了時に結果を表示する**］チェック・ボックスをクリアしていない限り，［テスト結果］ウィンドウが表示されます。［オプション］ダイアログ・ボックスの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 24 章「グローバル・テスト・オプションの設定」を参照してください。

テストの実行が停止すると，ステータス・バーに「**アップロードしています**」と表示されます。アップロード処理が完了すると，［テスト結果］ウィンドウが表示されます。

注：不具合は，発生するたび自動的に，または手作業で QuickTest の［テスト結果］ウィンドウから Quality Center プロジェクトに直接報告できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 686 ページ「実行セッション中に検出された不具合の送信」を参照してください。

QuickTest でのテストのバージョン管理

バージョン・コントロールがサポートされている Quality Center プロジェクトに QuickTest を接続すると，古いバージョンを維持したまま，自動化されたテスト・スクリプトの更新や変更ができます。これにより，各テスト・スクリプトに加えた変更の追跡，テスト・スクリプトのバージョン間の変更の確認，テスト・スクリプトの旧バージョンの復元が容易になります。

バージョン・コントロールがサポートされているプロジェクトにテストを保存すると，バージョン・コントロール・データベースにテストが追加されます。バージョン・コントロール・データベースを対象にテストをチェック・イン，チェック・アウトすることで，テストのバージョンを管理します。

最新バージョンのテストは，Quality Center のテスト・リポジトリにあり，Quality Center によってすべてのテストの実行に使用されます。

注：

バージョン・コントロールがサポートされている Quality Center プロジェクトを利用するは，バージョン・コントロールのソフトウェアと Quality Center のバージョン・コントロール・アドインをインストールする必要があります。詳細については，Quality Center のマニュアルを参照してください。

［**バージョン**］メニュー・オプション群は，バージョン・コントロールがサポートされている Quality Center プロジェクトのデータベースに接続して Quality Center テストを開いている場合にだけ使用できます。

バージョン・コントロール・データベースへのテストの追加

バージョン・コントロールがサポートされている Quality Center プロジェクトに［**名前を付けて保存**］を使って新規のテストを保存すると，QuickTest によって自動的にテストがプロジェクトに保存され，バージョン・コントロール・データベースにバージョン番号 1.1.1 としてチェック・インされ，その後作業を続けられるようにチェック・アウトされます。

QuickTest ステータス・バーにはそのつど各操作が表示されます。ただし，既存のテストに変更を保存しても，そのテストをチェック・インすることにはなりません。テストを保存して閉じても，チェック・インを選択するまではテストはチェック・アウトしたままです。詳細については，380 ページ「バージョン・コントロール・データベースへのテストのチェック・イン」を参照してください。

バージョン・コントロール・データベースからのテストのチェック・アウト

［**ファイル**］＞［**開く**］＞［**テスト**］を選択して，現在バージョン・コントロール・データベースにチェック・インされているテストを開くと，テストは読み取り専用モードで開きます。

注：［Quality Center からテストを開く］ダイアログ・ボックスには，プロジェクトの各テストのバージョン・コントロール・ステータスを示すアイコンが表示されます。詳細については，362 ページ「Quality Center プロジェクトのテストを開く」を参照してください。

チェック・インしたテストを確認できます。また，テストを実行して結果を表示することもできます。

テストに変更を加えるには，テストをチェック・アウトする必要があります。テストをチェック・アウトすると，テストは Quality Center によってユーザの固有のチェック・アウト・ディレクトリ（初めてテストをチェック・アウトするときに自動的に作成されます）にコピーされ，プロジェクト・データベース内のテストがロックされます。これにより，テストに加えた変更を Quality Center プロジェクトのほかのユーザに上書きされないようにします。ただし，この場合でも，ほかのユーザはデータベースにチェック・インされているテストの最新バージョンを実行できます。

テストを保存して閉じることはできますが，Quality Center データベースにテストを戻すまでテストはロックされています。テストを解放するには，テストをチェック・インするかチェック・アウト操作を取り消します。テストのチェック・インの詳細については，380 ページ「バージョン・コントロール・データベースへのテストのチェック・イン」を参照してください。チェック・アウトの取り消しの詳細については，386 ページ「チェック・アウト操作の取り消し」を参照してください。

標準設定では，チェック・アウト・オプションはテストの最新バージョンにアクセスします。古いバージョンのテストもチェック・アウトできます。詳細については，382 ページ「［バージョンの履歴］ダイアログ・ボックスの使用方法」を参照してください。

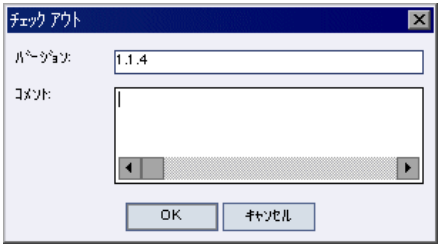
最新バージョンのテストをチェック・アウトするには，次の手順を実行します。

1 チェック・アウトするテストを開きます。詳細については，362 ページ「Quality Center プロジェクトのテストを開く」を参照してください。

注：開こうとしているテストが現在チェック・インされていることを確認します。自分がすでにチェック・アウトしているテストを開くと，［**チェック アウト**］オプションは無効になります。ほかのユーザによってチェック・アウトされているテストを開くと，［**バージョンの履歴**］オプションを除くすべての［**Quality Center バージョン コントロール**］オプション群が無効になります。

---

2 ［**バージョン**］＞［**チェック アウト**］を選択します。［チェック アウト］ダイアログ・ボックスが開き，チェック・アウトされるテスト・バージョンが表示されます。



3 ［**コメント**］ボックスに変更の詳細を入力します。

4 ［**OK**］をクリックします。読み取り専用のテストが閉じて，書き込み可能テストとして再び自動的に開きます。

5 必要に応じてテストを表示または編集します。

注：テストをチェック・インせずに変更を保存して閉じることができますが，テストをチェック・インするまで，ほかの Quality Center ユーザは加えられた変更を利用できません。変更を加えたテストをチェック・インしない場合は，チェック・アウトを取り消すことができます。テストのチェック・インの詳細については，380 ページ「バージョン・コントロール・データベースへのテストのチェック・イン」を参照してください。チェック・アウトの取り消しの詳細については，386 ページ「チェック・アウト操作の取り消し」を参照してください。

バージョン・コントロール・データベースへのテストのチェック・イン

あるテストがチェック・アウトされていても，Quality Center ユーザは以前にチェック・インされたバージョンを実行できます。たとえば，テストのバージョン 1.2.3 をチェック・アウトし，いくつか変更を加えてテストを保存するとします。バージョン 1.2.4（または指定した別の番号）としてテストをバージョン・コントロール・データベースに再びチェック・インするまで，Quality Center ユーザはバージョン 1.2.3 を引き続き実行できます。

テストの変更が終了し，Quality Center ユーザに新しいバージョンを使用してもらう準備が整ったら，バージョン・コントロール・データベースにテストをチェック・インします。

注：Quality Center データベースにテストをチェック・インしたくない場合は，チェック・アウト操作を取り消すことができます。詳細については，386 ページ「チェック・アウト操作の取り消し」を参照してください。

テストをバージョン・コントロール・データベースにチェック・インして戻すと，Quality Center はチェック・アウト・ディレクトリからテストのコピーを削除し，Quality Center プロジェクトのほかのユーザがそのバージョンのテストを利用できるように，データベース内のテストのロックを解除します。

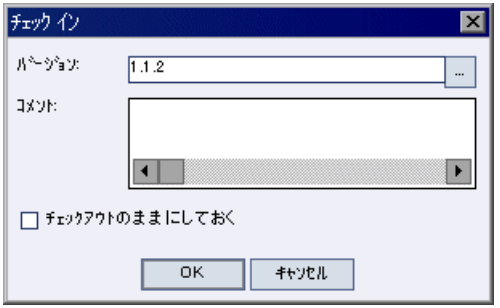
現在開いているテストをチェック・インするには，次の手順を実行します。

1 現在開いているテストを自分がチェック・アウトしていることを確認します。詳細については，382 ページ「テストのバージョン情報の表示」を参照してください。

注：開いているテストが現在チェック・インされていると，［**チェック イン**］オプションは無効になります。ほかのユーザによってチェック・アウトされているテストを開くと，［**バージョンの履歴**］オプションを除くすべての［**Quality Center バージョン コントロール**］オプション群が無効になります。

---

2 ［**ファイル**］＞［**Quality Center バージョン コントロール**］＞［**チェック イン**］を選択します。［チェック イン］ダイアログ・ボックスが表示されます。



3 標準設定の新しいバージョン番号を受け入れて手順 7 に進むか，参照ボタンをクリックしてユーザ定義のバージョン番号を指定します。参照ボタンをクリックすると，［チェック イン］ダイアログ・ボックスが開きます。

4 バージョン番号を，手作業またはバージョン番号の各要素の横にある上向き矢印と下向き矢印を使って変更します。最初の要素に入力できる数字は 1 ～ 900 です。第 2，第 3 の要素に入力できる数字は 1 ～ 999 です。バージョン・コントロール・データベースにある対象のテストの最新バージョン番号よりも低いバージョン番号は入力できません。

5 ［**OK**］をクリックしてバージョン番号を保存し，［チェック イン］ダイアログ・ボックスを閉じます。

6 テストのチェック・アウト時に変更の詳細を入力した場合，その詳細は［**コメント**］ボックスに表示されます。コメントの入力や修正はこのボックスで行えます。

7 ［**OK**］をクリックし，テストをチェック・インします。テストが閉じて，読み取り専用のテストとして再び自動的に開きます。

［バージョンの履歴］ダイアログ・ボックスの使用方法

［バージョンの履歴］ダイアログ・ボックスを使って，現在開いているテストに関するバージョン情報の表示や，古いバージョンのテストの表示および取得が行えます。

テストのバージョン情報の表示

Quality Center バージョン・コントロール・データベースに格納されている任意の開いているテストのバージョン情報を，現在のステータスに関係なく表示できます。

テストの［バージョンの履歴］ダイアログ・ボックスを開くには，テストを開き［**バージョン**］＞［**バージョンの履歴**］を選択します。

［バージョンの履歴］ダイアログ・ボックスには次の情報が表示されます。

［**テスト名**］：現在開いているテストの名前です。

［**テストのステータス**］：テストのステータスです。次のいずれかになります。

- ［**チェック イン**］：このテストは，現在バージョン・コントロール・データベースにチェック・インされています。テストは現在読み取り専用形式で開いています。テストをチェック・アウトして編集できます。
- ［**チェック アウト**］：このテストは，現在ユーザ自身がチェック・アウトしています。テストは現在読み書き可能な形式で開いています。
- ［**<他のユーザ>によるチェック アウト**］：このテストは，現在他のユーザによってチェック・アウトされています。テストは現在読み取り専用形式で開いています。示されているユーザがテストをチェック・インするまで，テストのチェック・アウトや編集はできません。

［**開いているテストのバージョン**］：現在 QuickTest コンピュータで開いているテストのバージョンです。

［**バージョン詳細**］：テストのバージョンの詳細です。

- ［**バージョン**］：テストのバージョンの一覧です。
- ［**ユーザ**］：各バージョンのチェック・インをしたユーザです。
- ［**日付**］：各バージョンがチェック・インされた日です。［**時刻**］：各バージョンがチェック・インされた時刻です。

［**コメント**］：選択したテスト・バージョンがチェック・インされたときに入力されたコメントです。

以前のバージョンのテストを対象とした作業

古いバージョンのテストは，読み取り専用モードで表示するか，チェック・アウトして最新バージョンとしてチェック・インできます。

古いバージョンのテストを表示するには，次の手順を実行します。

1 Quality Center テストを開きます。最新バージョンのテストが開きます。詳細については，362 ページ「Quality Center プロジェクトのテストを開く」を参照してください。

2 ［**ファイル**］＞［**Quality Center バージョン コントロール**］＞［**バージョンの履歴**］を選択します。［バージョンの履歴］ダイアログ・ボックスが表示されます。

3 表示するテストのバージョンを［**バージョン詳細**］リストから選択します。

4 ［**バージョンの取得**］ボタンをクリックします。テストがチェック・アウトされていないため読み取り専用モードで開く旨を知らせるメッセージが表示されます。

5 ［**OK**］をクリックし，QuickTest メッセージを閉じます。選択したバージョンが読み取り専用モードで開きます。

ヒント：

現在 QuickTest で開いているバージョン番号を確認するには，［バージョンの履歴］ダイアログ・ボックスの［**開いているテストのバージョン**］の値を調べます。

［**バージョンの取得**］オプションを使用して読み取り専用モードで古いバージョンを開いたら，［**バージョン**］＞［**チェック アウト**］を選択して，開いているテストをチェック・アウトできます。これは，［バージョンの履歴］ダイアログ・ボックスで［**チェック アウト**］ボタンを使用するのと同等の機能です。

古いバージョンのテストをチェック・アウトするには，次の手順を実行します。

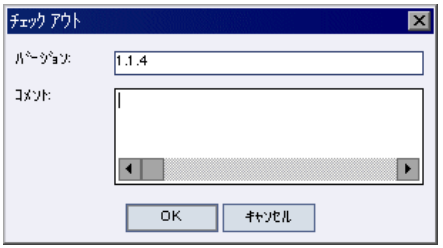
1 Quality Center テストを開きます。最新バージョンのテストが開きます。詳細については，362 ページ「Quality Center プロジェクトのテストを開く」を参照してください。

2 ［**ファイル**］＞［**Quality Center バージョン コントロール**］＞［**バージョンの履歴**］を選択します。［バージョンの履歴］ダイアログ・ボックスが表示されます。

3 表示するテストのバージョンを［**バージョン詳細**］リストから選択します。

4 ［**チェック アウト**］ボタンをクリックします。確認メッセージが開きます。

5 古いバージョンのテストをチェック・アウトすることを確認します。［チェック アウト］ダイアログ・ボックスが開き，チェック・アウトされるテスト・バージョンが表示されます。



6 ［**コメント**］ボックスに変更の詳細を入力します。

7 ［**OK**］をクリックします。開いていたテストが閉じ，選択したバージョンが書き込み可能テストとして開きます。

8 必要に応じてテストを表示または編集します。

9 新規の最新バージョンとしてテストを Quality Center データベースにチェック・インするには，［**ファイル**］＞［**バージョン コントロール**］＞［**チェック イン**］を選択します。変更したテストを Quality Center にアップロードしない場合は，［**ファイル**］＞［**Quality Center バージョン コントロール**］＞［**チェック アウト作業を元に戻す**］を選択します。

テストのチェック・インの詳細については，380 ページ「バージョン・コントロール・データベースへのテストのチェック・イン」を参照してください。
チェック・アウトの取り消しの詳細については，386 ページ「チェック・アウト操作の取り消し」を参照してください。

チェック・アウト操作の取り消し

テストをチェック・アウトした後に，変更を加えたテストを Quality Center にアップロードしないと決定した場合は，ほかの Quality Center ユーザがチェック・アウトできるように，チェック・アウト操作を取り消す必要があります。

チェック・アウト操作を取り消すには，次の手順を実行します。

1 チェック・アウトしたテストをまだ開いていない場合は開きます。

2 ［**ファイル**］＞［**Quality Center バージョン コントロール**］＞［**チェック アウト作業を元に戻す**］を選択します。

3 ［**はい**］をクリックし，チェック・アウト操作の取り消しを確定します。チェック・アウト操作が取り消されます。チェック・アウトしたテストが閉じ，前回チェック・インしたバージョンが読み取り専用モードで再び開きます。

Quality Center テストの実行に関する設定

Quality Center データベースに格納されている QuickTest テストの実行は，QuickTest，またはコンピュータにインストールされている Quality Center クライアント，あるいはリモートの Quality Center クライアントまたはサーバを介して行います。Quality Center は，QuickTest テストを実行するときに関連アドイン一覧を使用して，テストに適したアドインを QuickTest コンピュータにロードします。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 749 ページ「関連アドインの変更」および 366 ページ「テンプレート・テストを使用した作業」を参照してください。

注：QuickTest コンピュータがログオフまたはロックされている場合，Quality Center から QuickTest テストを実行することはできません。

Quality Center テストを QuickTest コンピュータで実行するとき，失敗したステップに関する不具合を報告するよう指示できます。また，［QuickTest テスト結果］ウィンドウから Quality Center に手作業で不具合を送信できます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 686 ページ「実行セッション中に検出された不具合の送信」を参照してください。

QuickTest テストを実行するようにリモートの Quality Center クライアントに指示する前に，QuickTest アプリケーションを使用する権限を Quality Center に与える必要があります。QuickTest リモート・エージェントの設定の表示や修正もできます。

Quality Center に対する QuickTest コンピュータでのテストの実行許可

セキュリティ上の理由により，QuickTest アプリケーションへのリモート・アクセスは許可されていません。Quality Center（またはほかのリモート・アクセスのクライアント）に対して QuickTest テストを開いて実行する許可を与えるには，［**他の Mercury 製品でテストまたはコンポーネントを実行可能にする**］オプションを選択する必要があります。

また，QuickTest テストを Quality Center からリモート実行する必要があり，QuickTest が Windows XP Service Pack 2 または Windows 2003 Server にインストールされている場合は，先に DCOM のアクセス許可を変更してファイアウォール・ポートを開く必要があります。詳細については，『**QuickTest Professional インストール・ガイド**』を参照するか，またはナレッジ・ベース（http://support.mercury.com/cgi-bin/portal/CSO/kbBrowse.jsp）にアクセスして記事番号 43245 を検索してください。

リモートの Quality Center クライアントが QuickTest コンピュータでテストを実行できるようにするには，次の手順を実行します。

1 QuickTest を開きます。

2 ［**ツール**］＞［**オプション**］を選択するか，［**オプション**］ツールバー・ボタンをクリックします。［オプション］ダイアログ・ボックスが開きます。

3 ［**実行**］タブをクリックします。

4 ［**他の Mercury 製品でテストまたはコンポーネントを実行可能にする**］チェック・ボックスを選択します。

詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 713 ページ「テストの実行オプションの設定」を参照してください。

ヒント：Quality Center から QuickTest テストにアクセスするには，Quality Center 用の QuickTest アドインが Quality Center コンピュータにインストールされている必要があります。このアドインの詳細については，QuickTest Professional アドイン画面 （Quality Center のメイン画面からアクセスできます）を参照してください。

QuickTest リモート・エージェントの設定

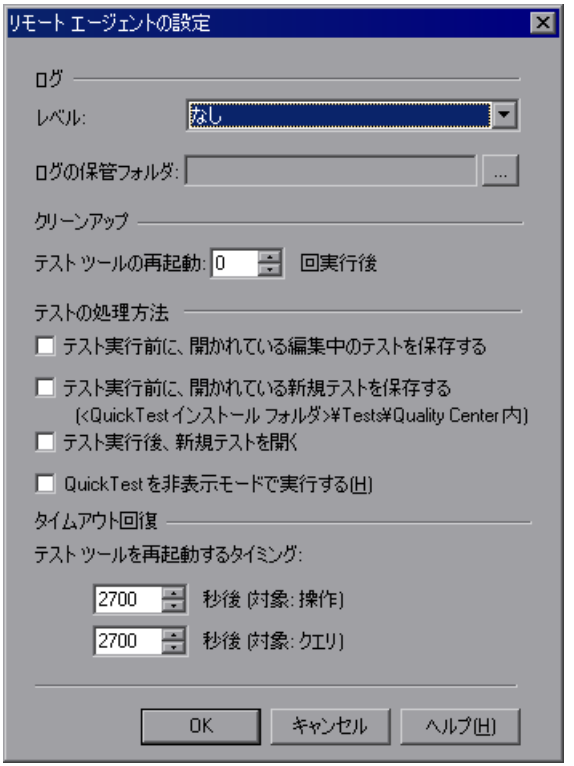
Quality Center から QuickTest テストを実行すると，QuickTest コンピュータで QuickTest リモート・エージェントが開きます。テストが Quality Center などのリモートのアプリケーションによって実行された場合の QuickTest の動作は，QuickTest リモート・エージェントによって決まります。

いつでも［リモート エージェントの設定］ダイアログ・ボックスを開いて，Quality Center がコンピュータで QuickTest テストを実行するときに QuickTest アプリケーションが使用する設定の表示や変更ができます。

［リモート エージェントの設定］ダイアログ・ボックスを開くには，次の手順を実行します。

1 ［**スタート**］メニューの［**QuickTest Professional**］プログラム・グループから［**Tools**］＞［**Remote Agent**］を選択します。リモート・エージェントが開き，リモート・エージェントのアイコンがタスクバー・トレイに表示されます。

2 リモート・エージェント・アイコンを右クリックし，［**設定**］を選択します。［リモート エージェントの設定］ダイアログ・ボックスが開きます。



3 ダイアログ・ボックスで設定の表示や変更ができます。詳細については次ページの「[リモート エージェントの設定］ダイアログ・ボックスについて」を参照してください。

4 ［**OK**］をクリックし，設定を保存してダイアログ・ボックスを閉じます。

5 リモート・エージェント・アイコンを右クリックして［**終了**］を選択し，リモート・エージェント・セッションを終了します。

［リモート エージェントの設定］ダイアログ・ボックスについて

［リモート エージェントの設定］ダイアログ・ボックスでは，Quality Center がコンピュータで QuickTest テストを実行するときに QuickTest アプリケーションが使用する設定の表示や変更ができます。

［リモート エージェントの設定］ダイアログ・ボックスには，次のオプションがあります。

オプション	詳細
［レベル］	Quality Center が QuickTest テストを実行するときに作成されるログに含まれる内容のレベルです。 **［なし］**（標準設定）：ログは作成されません。 **［低］**：ログには Quality Center と QuickTest の間の通信エラーがすべて表示されます。 **［中］**：ログには Quality Center と QuickTest の間の通信エラーと，Quality Center と QuickTest の間の通信に関するほかの重要な操作の情報が含まれます。 **［高］**：ログには Quality Center と QuickTest の通信に関する，すべての利用可能な情報が含まれています。
［ログの保管フォルダ］	ログ・ファイルが格納されているフォルダのパスです。［**レベル**］オプションでログの種類を指定した場合に必要です。
［テスト ツールの再起動 X 回実行後］	Quality Center がテスト実行を指定されている回数完了させたら，QuickTest アプリケーションを再起動します。QuickTest は再起動すると，テスト・セットの中の次のテストを実行します。 このオプションを使用して利用可能なメモリを最大限にできます。 テスト・セット中に QuickTest を再起動しない場合は，0（標準設定）を入力してください。
［テスト実行前に、開かれている編集中のテストを保存する］	リモート・エージェントによってテストの実行が開始されるときに，QuickTest で既存の（命名済みの）テストが開かれている場合，このオプションにより，テストに加えられたすべての変更が保存されます。

オプション	詳細
[テスト実行前に、開かれている新規テストを保存する]	リモート・エージェントによってテストの実行が開始されるときに，QuickTest で新規の（命名されていない）テストが開かれている場合，このオプションはテストに連番が付いたテスト名を指定して **＜ QuickTest のインストール・フォルダ＞¥Tests¥Quality Center** に保存します。
[テスト実行後、新規テストを開く]	標準設定では，すべてのテストの実行が終わったとき，リモート・エージェントによって最後に実行されたテストは QuickTest で開いたままです。ただし，開いているテストに共有リソース（共有オブジェクト・リポジトリやデータ・テーブル・ファイルなど）が関連付けられている場合，これらのリソースは，テストが閉じられるまでほかのユーザに対してロックされています。このオプションを選択すれば，Quality Center が実行する最後のテストを閉じ，代わりに空のテストを開くことができます。
[QuickTest を非表示モードで実行する]	非表示（サイレント）モードで QuickTest を実行するかどうかを指定します。
[テスト ツールを再起動するタイミング]	次に示す項目に指定されている秒数が経過しても応答がない場合，QuickTest を再起動します。 **操作**：開く，実行などの QuickTest 操作です。 **クエリ**：リモートのアプリケーションが実行してアプリケーションの応答を確認する，標準のステータス・クエリです（Quality Center の **get_status** クエリなど）。 両方のオプションの標準設定の値は 2700 秒（45 分）です。ただし，QuickTest 操作の応答時間が長くなることはあっても，クエリの応答時間は通常数秒しかかかりません。したがって，それぞれのオプションに異なる値を設定するとよいでしょう。

第 15 章

Business Process Testing を使用した作業

Business Process Testing のサポートが組み込まれた Quality Center プロジェクトに接続すると，QuickTest で Quality Center ビジネス・プロセス・テストで使用されるコンポーネントのステップの作成や実装を行えます。

本章では，次の内容について説明します。

- Business Process Testing での作業
- Business Process Testing での役割について
- Business Process Testing のテスト手法について

Business Process Testing での作業

Business Process Testing では，各分野のエキスパートがテストの実施および自動テスト環境の向上を目的として，キーワード駆動方式のテストを作成できます。

Business Process Testing は QuickTest を Quality Center と組み合わせて使用し，専用の Business Process Testing ライセンスを購入することによって有効にできます。QuickTest 内から Business Process Testing を使って作業するためには，Business Process Testing のサポートが組み込まれた Quality Center プロジェクトに接続する必要があります。

本項では Business Process Testing モデルの概要を説明します。詳細については，『**Business Process Testing ユーザーズ・ガイド**』（Quality Center 付属マニュアルに含まれています）および『**QuickTest Professional for Business Process Testing ユーザーズ・ガイド**』を参照してください。

Business Process Testing での役割について

Business Process Testing モデルはロール（役割）・ベースであり，（Quality Center で作業を行う）技術者でない各分野のエキスパートが，（QuickTest Professional で作業を行う）オートメーション・エンジニアと効率よく連携をとれるようになっています。各分野のエキスパートは，ビジネス・プロセス，ビジネス・コンポーネントおよびビジネス・プロセス・テストを定義し文書化します。一方，オートメーション・エンジニアは，共有オブジェクト・リポジトリ，関数ライブラリ，回復シナリオといった必要なリソースおよび設定を定義します。両者が連携することで，各分野のエキスパートにプログラミングの知識がなくても，ビジネス・プロセス・テストを作成し，データ駆動し，文書化し，実行できます。

注：役割の構造と，組織における役割ごとに実行されるタスクは，組織で採用されている方法論に応じて，ここに記述されているものとは異なる場合があります。これらの役割は柔軟性があり，Business Process Testing を使用する人の能力と時間に応じても役割は異なります。たとえば，各分野のエキスパートのタスクおよびオートメーション・エンジニアのタスクを同じ人が実行する場合もあります。組織に応じて定義すべき役割，またはユーザ・タイプに応じて実行できる Business Process Testing タスクを規定する製品固有の役割や制限はありません（ユーザが正しい権限を持っている限り）。

Business Process Testing モデルには，次のユーザの役割の分類があります。

各分野のエキスパート：各分野のエキスパートは，アプリケーション・ロジックに関する知識，システム全体の深い理解，テスト対象アプリケーションの基本となる個々の要素とタスクに関する深い理解を有しています。これにより，各分野のエキスパートはテストが必要な運用シナリオやビジネス・プロセスを決定したり，複数のビジネス・プロセスに共通の重要なビジネス・アクティビティを特定したりできます。

Quality Center でビジネス・コンポーネント・モジュールを使用して，各分野のエキスパートは，アプリケーションを対象に実行される個別のタスクと，こうしたタスクの前後のアプリケーションの条件または状態を記述するビジネス・コンポーネントを作成します。その後，各分野のエキスパートは，ビジネス・プロセスを構成する各ビジネス・コンポーネントの個々のステップを手動形式，すなわち自動化されていないステップ形式で定義します。

デザイン段階で，各分野のエキスパートは，オートメーション・エンジニアと作業を行い，コンポーネントの自動化に必要なリソースおよび設定を特定し，オートメーション・エンジニアがそれらを準備できるようにします。リソースと設定が準備できたら，各分野のエキスパートがそれらをキーワード駆動型のコンポーネントに変換して，手動のステップを自動化します。このプロセスでは，各コンポーネントのアプリケーション領域を選択する必要があります。アプリケーション領域には，必要なすべてのリソース・ファイルおよび設定が含まれます。これらは，テスト対象アプリケーションの特定の領域に固有のリソース・ファイルおよび設定です。各コンポーネントをアプリケーション領域に関連付けることで，コンポーネントはこれらのリソースおよび設定にアクセスできるようになります。

各分野のエキスパートは Quality Center のテスト計画モジュールを使用してビジネス・コンポーネントを組み合わせて，連続するコンポーネントのフローで構成されるビジネス・プロセス・テストを作成します。たとえば，ほとんどのアプリケーションでは，ユーザはアプリケーションの何らかの機能にアクセスする前にログインする必要があります。各分野のエキスパートは，このログインの手順を行う 1 つのビジネス・コンポーネントを作成できます。このコンポーネント・プロシージャは，さまざまなビジネス・プロセス・テストで使用できます。その結果，保守，更新，テスト管理がより容易かつコスト効率の高いものになります。

各分野のエキスパートは，ビジネス・プロセス・テストに使用される値を設定し，それらをテスト・セットの中で実行して，結果を検討します。また，各ビジネス・コンポーネントのテスト・ステップの保守も担当します。

コンポーネントを定義する間，各分野のエキスパートはオートメーション・エンジニアと協力を続けます。たとえば，コンポーネントに新しい操作（機能）を要求したり，コンポーネントに対して計画されている今後の変更を検討したりすることが考えられます。

オートメーション・エンジニア：オートメーション・エンジニアは QuickTest Professional のような自動テスト・ツールの使用に関するエキスパートです。オートメーション・エンジニアは各分野のエキスパートと連携して，さまざまなビジネス・プロセス・テストで必要なリソースを特定します。

その後，オートメーション・エンジニアは，個別のコンポーネントに関連付けられている機能をテストするために必要なリソースおよび設定を準備し，それらを Quality Center プロジェクト内のアプリケーション領域に格納します。Quality Center プロジェクトは，特定のアプリケーションを対象にしたビジネス・プロセス・テストの作成および実行を担当する各分野のエキスパートが使用するプロジェクトと同じものです。

各アプリケーション領域は，コンポーネントに必要なすべてのリソースおよび設定を保存する 1 つのエンティティとしての役割を果たします。これは，アプリケーションの特定部分のテストに関連付けられているすべての要素の保守を集中して行う場所となります。アプリケーション領域には一般に，1 つ以上の共有オブジェクト・リポジトリ，コンポーネントに対して使用できるキーワードのリスト，自動化関数（操作）を含む関数ライブラリ，失敗したステップのための回復シナリオ，そのほかコンポーネントが正しく動作するために必要な他のリソースおよび設定が含まれています。コンポーネントは，アプリケーション領域内のリソースおよび設定に結び付いています。したがって，アプリケーション領域に変更が加えられると，関連するすべてのコンポーネントが自動的に更新されます。

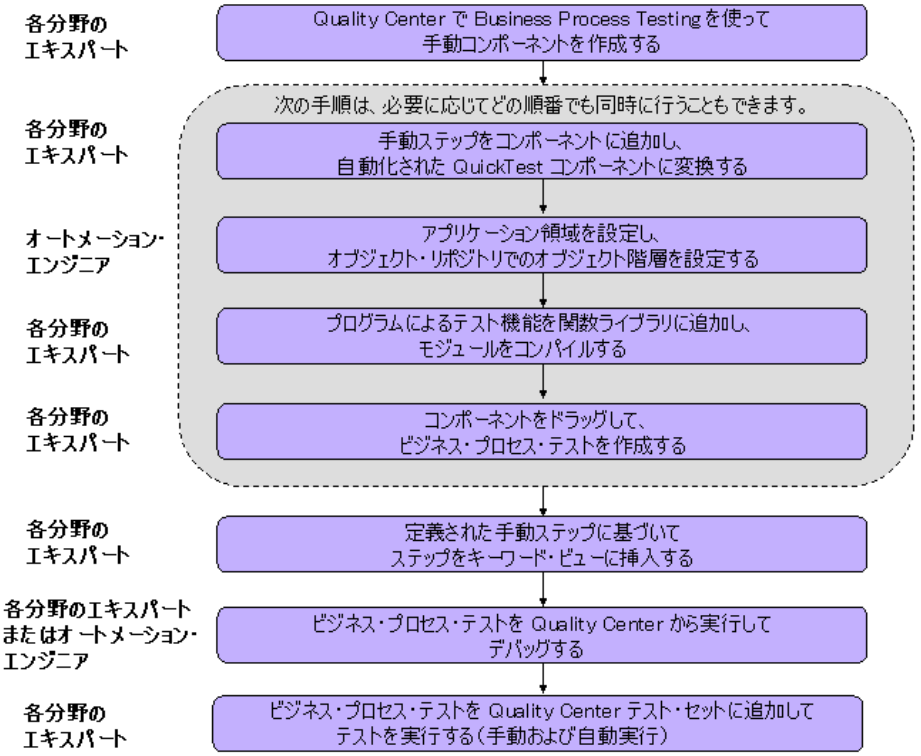
オートメーション・エンジニアは QuickTest の仕組みおよび機能を使用して，QuickTest からこれらのリソースを作成します。たとえば，QuickTest では，オートメーション・エンジニアは，アプリケーションの開発が完了する前でも，各種の共有オブジェクト・リポジトリを作成し，テスト対象アプリケーション内のさまざまなオブジェクトを表すテスト・オブジェクトを含めることができます。その後，オートメーション・エンジニアは必要に応じてリポジトリ・パラメータなどを追加できます。オートメーション・エンジニアは，オブジェクト・リポジトリ・マネージャを使用して各種のオブジェクト・リポジトリを管理でき，オブジェクト・リポジトリ結合ツールを使用してリポジトリを結合できます。オートメーション・エンジニアはまた，QuickTest の関数ライブラリ・エディタを使用して，プログラミング・ロジックの使用を通じて特定のタスクの実行に必要なステップをカプセル化する関数を含む関数ライブラリを，作成およびデバッグすることもできます。

オートメーション・エンジニアによって作成されたリソースを使用して，各分野のエキスパートはコンポーネント・ステップを自動化し，コンポーネントおよびビジネス・プロセス・テストの作成と保守ができます。

オートメーション・エンジニアは，必要に応じて QuickTest の中でコンポーネントを作成，デバッグ，および変更することもできます。

Business Process Testing のワークフローについて

Business Process Testing のワークフローはテストのニーズに応じて異なります。次は一般的なワークフローの例です。



Business Process Testing のテスト手法について

各分野のエキスパートが作成する各シナリオは，**ビジネス・プロセス・テスト**です。1 つのビジネス・プロセス・テストは，連続する**コンポーネント**のフローで構成されています。各コンポーネントによって特定のタスクが実行されます。また，コンポーネントから後続のコンポーネントにデータを引き渡せます。

コンポーネントについて

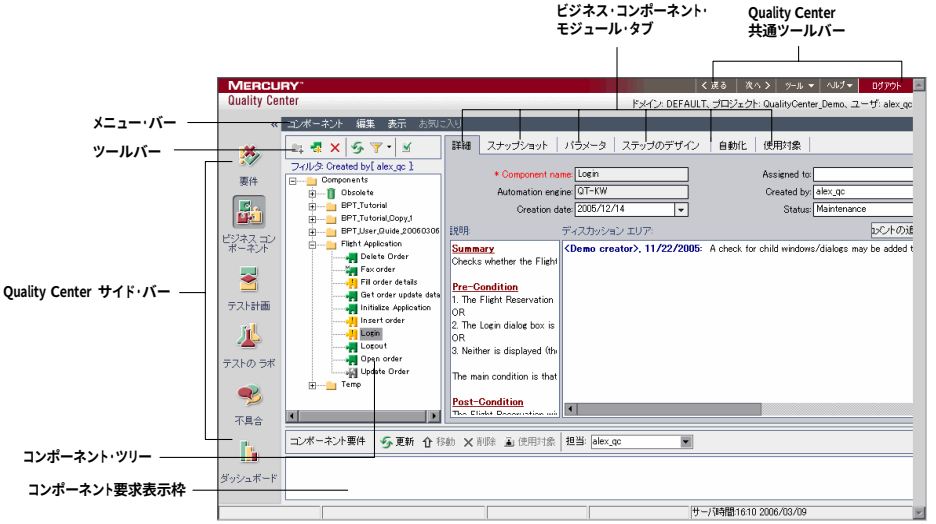
コンポーネントは，保守が簡単で，特定のタスクを実行する再利用可能なスクリプトであり，効果的なビジネス・プロセス・テスト構造を作成できる基礎的な要素です。コンポーネントは細分化されたビジネス・プロセスの一部です。たとえば，ほとんどのアプリケーションのユーザは，何かを行う前にログインする必要があります。各分野のエキスパートは，アプリケーションへログインする手続きを行う 1 つのコンポーネントを作成できます。各コンポーネントは別のビジネス・プロセスのテストで再利用できます。その結果，保守，更新，テスト管理がより容易になります。

コンポーネントはステップで構成されています。たとえば，ログイン・コンポーネントの最初のステップではアプリケーションを開きます。第 2 ステップはユーザ名の入力です。第 3 ステップはパスワードの入力，そして第 4 ステップは［**ログイン**］ボタンのクリックです。

サポートされている任意の環境を対象としたステップの追加，特定の項目のパラメータ化，および特定のタスクの実行に必要なステップをカプセル化する関数（操作）の組み込みによるコンポーネントの拡張など，QuickTest の中でコンポーネントの作成および編集ができます。Quality Center では，各分野のエキスパートはコンポーネントを作成し，それらを組み合わせて，アプリケーションが想定どおりに動作することをチェックするために使用するビジネス・プロセス・テストを組み立てます。

Quality Center ビジネス・コンポーネント・モジュールでのコンポーネントの作成

各分野のエキスパートは，新規のコンポーネントを作成し，Quality Center のビジネス・コンポーネント・モジュールの中でそれを定義できます。



ビジネス・コンポーネント・モジュールには，次の要素が含まれています。

- **説明**：コンポーネントの目的またはゴールのサマリおよびコンポーネントの実行前後のアプリケーション状態（前提条件および後堤条件）を指定します。
- **スナップショット**：コンポーネントの目的や操作を視覚的に示すイメージを表示します。
- **パラメータ**：ビジネス・コンポーネントの入力コンポーネント・パラメータ値および出力コンポーネント・パラメータ値を指定します。パラメータを実装および使用することで，コンポーネントは，外部ソースからデータを受け取り，ビジネス・プロセス・テスト・フローのほかのコンポーネントにデータを渡すことができるようになります。
- **ステップのデザイン**：ビジネス・コンポーネントの手動ステップを作成または表示できます。必要に応じて手動ステップを自動化できます。

- **オートメーション**：自動化されたコンポーネントを表示し，アクセスできるようにします。キーワード駆動型のコンポーネント用に，キーワード方式のテーブル形式で自動化されたビジネス・コンポーネントのステップを作成および変更できます。また実装されているコンポーネントの各ステップについて分かりやすく書かれた説明が表示されます。
- **使用対象**：現在選択されているビジネス・コンポーネントが含まれるビジネス・プロセス・テストの詳細が示されます。また，テスト計画モジュールの関連ビジネス・プロセス・テストへのリンクも示されます。
- **コンポーネント要件表示枠**：テスト計画モジュールで生成された新しいコンポーネント要件を処理します。コンポーネント要件とは，プロジェクトに新規ビジネス・コンポーネントを追加する要件です。

QuickTest Professional でのコンポーネントの実装

一般に，コンポーネントは各分野のエキスパートが Quality Center の中で作成しますが，QuickTest の中でも作成およびデバックができます。

QuickTest 内では，サポートされている任意の環境を対象としたステップを記録するか，ステップを手動で追加することで，コンポーネントを作成します（オブジェクト・リポジトリにデータが設定されており，必要な操作が利用可能な場合）。特定の項目のパラメータ化が可能です。また，コンポーネント固有のオプションの表示や設定も行えます。

QuickTest では，2 種類のコンポーネントの作成および変更が可能です。**ビジネス・コンポーネント**および**スクリプト・コンポーネント**の 2 つです。ビジネス・コンポーネントは，特定のタスクを実行する 1 つ以上のステップで構成されている，保守が容易で再利用可能な単位です。スクリプト・コンポーネントは，プログラム・ロジックを含むことのできる自動化されたコンポーネントです。スクリプト・コンポーネントは，テスト・アクションおよびビジネス・コンポーネントの両方の機能を併せ持ちます。たとえば，キーワード・ビュー，エキスパート・ビュー，その他の QuickTest ツールやオプションを使用して，QuickTest の中でスクリプト・コンポーネントの作成，表示，変更，デバッグが行えます。スクリプト・コンポーネントは複雑なため，QuickTest 内でのみ編集可能です。

Quality Center では，各分野のエキスパートは QuickTest 内で作成されたコンポーネントを開けます。その後，各分野のエキスパートはビジネス・コンポーネントを表示および編集できますが，スクリプト・コンポーネントについては詳細のみを表示できます。

Quality Center テスト計画モジュールでのビジネス・プロセスのテストの作成

ビジネス・プロセス・テストを作成するには，各分野のエキスパートがビジネス・プロセス・テストに適したコンポーネントを選択（ドラッグ・アンド・ドロップ）し，それらの実行設定を設定します。

各コンポーネントは，ビジネス・プロセス・テストごとに異なる使い方ができます。たとえば，コンポーネントに対してテストごとに異なる入力パラメータ値を使用したり，異なる反復回数で実行されるよう設定したりできます。

ビジネス・プロセス・テストを作成しているときに，各分野のエキスパートが，ビジネス・プロセス・テストに必要な要素に対してコンポーネントがまだ定義されていないことに気付いたとします。その場合は，テスト計画モジュールからコンポーネント要求を送信できます。

ビジネス・プロセスのテストの実行と結果の分析

QuickTest で［実行］オプションや［デバッグ］オプションを使用して，個々のコンポーネントの実行やデバッグができます。

Quality Center のテスト計画モジュールからテストを実行して，ビジネス・プロセス・テストのデバッグが行えます。このモジュールから実行する場合，デバッグ・モードで実行するコンポーネントを選択できます（その場合，コンポーネントの開始時に実行が一時停止します）。

ビジネス・プロセスのテストがデバッグされ，通常のテスト実行の準備が整ったら，各分野のエキスパートは，他のテストを Quality Center で実行する場合と同様の方法で，テストのラボ・モジュールからテストを実行します。テストを実行する前に，各分野のエキスパートは，テストのラボ・モジュールのグリッドの［**反復**］カラムを使用して，実行時のパラメータ値と反復を定義できます。

テストのラボ・モジュールから，ビジネス・プロセス・テスト全体の実行結果を確認できます。結果には，各パラメータの値や，QuickTest によって報告された個々のステップの結果が含まれています。

［**レポートの表示**］のリンクをクリックして，QuickTest レポート全体を開くことができます。この階層化されたレポートには，ビジネス・プロセス・テスト実行での，それぞれの反復やコンポーネントがすべて含まれています。

コンポーネントとテストの違いについて

すでに QuickTest を使用したアクション・ベースのテストの作成に慣れている場合は，その手順とコンポーネントの作成や編集の手順が非常に似ていることに気付くでしょう。ただし，コンポーネント・モデルの設計や目的により，コンポーネントの作成，編集，実行の点で多少の違いがあります。次のガイドラインでは，これらの違いの概要を説明します。

- コンポーネントは単独のエンティティです。複数のアクションを含めたり，別のアクションや別のコンポーネントを呼び出したりすることはできません。
- コンポーネントを使って作業をすると，外部ファイルはすべて現在接続している Quality Center プロジェクトに格納されます。
- キーワード・ビュー内のコンポーネント・ノードの名前は，保存されているコンポーネントと同じです。ノード名は変更できません。
- ビジネス・コンポーネントは，エキスパート・ビューではなくキーワード・ビューで作成します。
- コンポーネントに直接ではなく，コンポーネントのアプリケーション領域を介してリソースを追加します。
- コンポーネントは，関数ライブラリで作成されたユーザ定義のキーワードを使用して，プロパティ値の確認およびテスト対象アプリケーションの起動などの操作を実行します。

第 16 章

Mercury のパフォーマンス・テスト製品および Business Availability Center 製品を使用した作業

QuickTest を使用して，アプリケーションの機能をテストする一連のテストを作成して実行できたら，今度はアプリケーションがどれくらいの負荷を処理できるかを検証したり，アプリケーションが実行する様子を監視したりできます。

Mercury LoadRunner は，一定の負荷または過負荷状態でのアプリケーションのパフォーマンスをテストします。負荷を生成するために，LoadRunner によって無数の仮想ユーザが実行されます。これらの仮想ユーザは，一貫性のある再現可能かつ測定可能な負荷を提供し，現実のユーザとまったく同じようにアプリケーションを操作します。

Mercury Business Availability Center は，エンド・ユーザ体験をリアルタイムで監視できます。Business Process Monitor は，監視対象アプリケーションを対象に仮想ユーザを実行して典型的な操作を実行します。

すでに QuickTest でテストを作成して，それがユーザのアクションをうまく表現することが分かっている場合は，その QuickTest テストをパフォーマンス・テストおよびアプリケーション管理の基盤として使うことができます。サイレント・テスト・ランナーを使用して，QuickTest テストが LoadRunner およびビジネス・プロセス・モニタから正常に実行されることを事前に確認できます。

本章では，次の項目について説明します。

- Mercury のパフォーマンス・テストおよび Business Availability Center 製品を使用した作業について
- QuickTest のパフォーマンス・テストおよび Business Availability Center の使用
- LoadRunner または Business Process Monitor で使用する QuickTest テストの設計
- LoadRunner または Business Process Monitor でのテストの挿入と実行
- トランザクションの測定
- サイレント・テスト・ランナーの使用

Mercury のパフォーマンス・テストおよび Business Availability Center 製品を使用した作業について

QuickTest では，アプリケーション全般の機能を検査し，アプリケーションのすべての要素があらゆる状況で期待どおりに動作することを確認する複雑なテストを作成できます。

Mercury のパフォーマンス・テスト製品および Mercury Business Availability Center 製品で使用されている実行メカニズムは同じです。つまり，LoadRunner と Business Process Monitor の両方に互換性のあるテストを作成することができ，QuickTest で設計され，デバッグされたテストやテストのセグメントを利用できるということです。

たとえば，QuickTest テストを LoadRunner シナリオの特定のポイントに追加し，それらのポイントでの追加の負荷によってアプリケーションの機能が影響を受けていないことを確認できます。また，Business Process Monitor を対象として QuickTest テストを実行し，エンドユーザの体験をシミュレートし，アプリケーションが正常かつタイムリーに実行されることを確かめることもできます。

また QuickTest は，LoadRunner および Business Process Monitor との統合専用に設計されているいくつかの機能を提供します。しかし，LoadRunner および Business Process Monitor は，標準的なユーザ操作を同時に行う多数のユーザを表す仮想ユーザを使用して，テストを実行するように設計されているので，QuickTest とこれらの製品を統合すると，QuickTest のいくつかの機能が利用できない場合があります。

LoadRunner か Business Process Monitor またはその両方と，QuickTest とで，1 つのテストを使う場合は，テストを設計するときに，各製品がサポートするオプションの違いを考慮しなければなりません。詳細については，406 ページ「LoadRunner または Business Process Monitor で使用する QuickTest テストの設計」および 408 ページ「LoadRunner または Business Process Monitor でのテストの挿入と実行」を参照してください。

QuickTest のパフォーマンス・テストおよび Business Availability Center の使用

Services オブジェクトと関連メソッドを使用して，パフォーマンス・テストおよび Business Availability Center に特に関連のあるステートメントを挿入できます。これらには，**AddWastedTime**，**EndDistributedTransaction**，**EndTransaction**，**GetEnvironmentAttribute**，**LogMessage**，**Rendezvous**，**SetTransaction**，**SetTransactionStatus**，**StartDistributedTransaction**，**StartTransaction**，および **ThinkTime** があります。これらのメソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』の「**Services**」の節，および，LoadRunner または Business Availability Center のマニュアルを参照してください。

また，［**挿入**］＞［**トランザクションの開始**］および［**挿入**］＞［**トランザクションの終了**］メニュー・オプションまたはツールバー・ボタン を使用して **StartTransaction** および **EndTransaction** ステートメントを挿入できます。これらのオプションの詳細については，410 ページ「トランザクションの測定」を参照してください。

注：LoadRunner と Business Process Monitor は，トランザクションに含まれたデータだけを使用し，トランザクション外のテストのデータは無視します。

LoadRunner または Business Process Monitor で使用する QuickTest テストの設計

LoadRunner または Business Process Monitor で使用する QuickTest テストは，シンプルに保ち，対象の操作を限定し，外部アクションの使用や外部ファイルへの参照は避けるべきです。また，アクションの反復を操作するときは，対応関係にある **StartTransaction** ステートメントおよび **EndTransaction** ステートメントが同じアクション内に含まれている必要があります。

LoadRunner 向けのテストの設計

LoadRunner で使用するテストを設計するときは，次のガイドラインに従ってください。

- 外部アクションや，外部データ・テーブル・ファイル，環境変数ファイル，共有オブジェクト・リポジトリ，関数ライブラリなどの外部リソース（Quality Center に保存されているリソースを含む）への参照を含めないようにします。これは，LoadRunner が外部のアクションやリソースにアクセスできない場合があるためです（ただし，リソースがネットワーク上で見つかる場合，QuickTest はそのリソースを使用します）。
- LoadRunner で有益な情報を提供するためには，すべての QuickTest テストに少なくとも 1 つのトランザクションを含める必要があります。
- テスト内の最後の（1 つまたは複数の）ステップで，テスト対象のアプリケーションを，実行中のすべての子プロセスも含めて必ず終了するようにします。これにより，テストの次の反復で再びアプリケーションを開くことができるようになります。

Business Process Monitor 向けのテストの設計

Business Process Monitor で使用するテストを設計するときは，次のガイドラインに従ってください。

- Business Process Monitor で有益な情報を提供するためには，すべての QuickTest テストに少なくとも 1 つのトランザクションを含める必要があります。
- 2 つの異なるビジネス・プロセス・モニタ・プロファイルについて分散トランザクションを測定するときは，**StartDistributedTransaction** ステートメントのあるプロファイルを，対応する **EndDistributedTransaction** ステートメントのあるプロファイルよりも先に実行する必要があります。

- 分散トランザクションを測定するときは，必ずテストを単一のビジネス・プロセス・モニタ・インスタンスに関連付けるようにします。ビジネス・プロセス・モニタはすべてのインスタンスの中でトランザクション終了名を探します。そのため，トランザクション終了名が複数のインスタンスに含まれていると，誤った分散トランザクションが終了される可能性があります。
- 2 つのビジネス・プロセス・モニタ・プロファイルについて分散トランザクションを測定するときは，**StartDistributedTransaction** ステップが含まれているプロファイルと，**EndDistributedTransaction** ステップが含まれているプロファイルの前に実行されるすべてのプロファイルが，指定されたタイムアウトの値よりも短い時間で実行を完了するように，十分大きなタイムアウト値を指定するようにします。
- Business Process Monitor では，Quality Center に保存されているリソース（共有オブジェクト・リポジトリ，関数ライブラリ，外部データ・テーブル，外部アクションなど）を含む外部リソースへのアクセスを必要とする QuickTest Professional テストの実行はサポートされていません。外部リソースを必要とするテストは，Business Process Monitor 上での実行に失敗する場合があります（ただし，リソースがネットワーク上で見つかる場合，QuickTest はそのリソースを使用します）。
- テスト内の最後の（1 つまたは複数の）ステップで，テスト対象のアプリケーションを，実行中のすべての子プロセスも含めて必ず終了するようにします。このクリーンアップ・ステップにより，次のテスト実行で再びアプリケーションを開くことができるようになります。

LoadRunner または Business Process Monitor でのテストの挿入と実行

LoadRunner または Business Process Monitor で QuickTest テストを挿入し実行するときは，次のガイドラインに従ってください。

注：サイレント・テスト・ランナーを使用すれば，LoadRunner または Business Process Monitor からのテストの実行方法をシミュレートできます。詳細については，414 ページ「サイレント・テスト・ランナーの使用」を参照してください。

LoadRunner シナリオでのテストの挿入と実行

- 1 台のコンピュータで同時に実行できる GUI 仮想ユーザは 1 つまでです（GUI 仮想ユーザとは QuickTest テストを実行する仮想ユーザのことです）。
- QuickTest テストを LoadRunner シナリオに挿入するには，コントローラの［テストを開く］ダイアログ・ボックスでテスト・フォルダまで移動し，［**ファイルの種類**］ボックスで［**Astra テスト**］を選択します。これで，フォルダ内の QuickTest テストが表示されます。
- LoadRunner から QuickTest テストを実行する前に，QuickTest コンピュータ上の QuickTest が終了していることを確認します。
- QuickTest を使用して記録したテスト（スクリプト）では，トランザクション・ブレークダウンはサポートされていません。
- 次のコンピュータ上では QuickTest を実行できません。
  - ログオフまたはロックされたコンピュータ。これらの場合は，ターミナル・サーバでの QuickTest の実行を検討してください。
  - すでに QuickTest テストが実行されているコンピュータ。テストの実行が完了していることを確かめてから，別の QuickTest テストを開始してください。
- LoadRunner の［実行環境設定］ダイアログ・ボックスの設定は，QuickTest テストには関係しません。
- LoadRunner でテストを実行しているときは，**ResultDir** QuickTest 環境変数を使用することはできません。

LoadRunner を使用した作業の詳細については，LoadRunner のマニュアルを参照してください。

Business Process Monitor でのテストの挿入と実行

- Business Process Monitor で QuickTest テストを実行する前に，使用しているバージョンの Business Process Monitor でどのバージョンの QuickTest がサポートされているのかを確認してください。詳細については，Business Process Monitor のマニュアルを参照してください。
- Business Process Monitor で一度に実行できる QuickTest テストは 1 つだけです。前の QuickTest テストの実行が完了していることを確かめてから，別の QuickTest テストを開始してください。
- Business Process Monitor から QuickTest テストを実行する前に，QuickTest コンピュータ上の QuickTest が終了していることを確認します。
- QuickTest を使用して記録したテストでは，トランザクション・ブレークダウンはサポートされていません。
- QuickTest テストを Business Availability Center にアップロードした後に，テストのローカル・コピーに変更を加えた場合は，変更を加えたテストを Business Process Monitor で実行できるようにするために，ZIP 圧縮したテストを再度アップロードする必要があります。
- ログオフされたコンピュータ，ロックされたコンピュータ，または QuickTest を非対話型サービスとして実行しているコンピュータでは，QuickTest はテストを実行できません。
- Business Process Monitor でテストを実行しているときは，**ResultDir** QuickTest 環境変数を使用することはできません。

Business Availability Center を使用した作業の詳細については，Mercury Business Availability Center のマニュアルを参照してください。

トランザクションの測定

トランザクションを定義することで，テストの特定セクションの実行にかかる時間を測定できます。トランザクションは，アプリケーション内の測定対象プロセスを表します。テストには，LoadRunner またはビジネス・プロセス・モニタで使用できるトランザクションを含める必要があります。LoadRunner とビジネス・プロセス・モニタは，トランザクションに含まれているデータだけを使用し，トランザクション外のテストのデータは無視します。

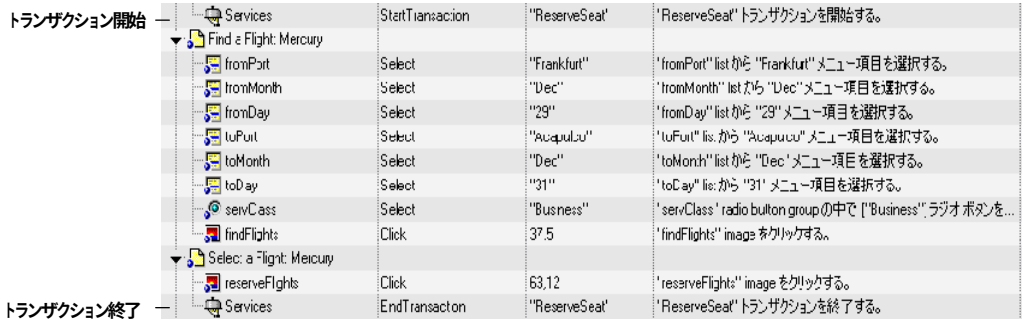
トランザクション**開始**ステートメントとトランザクション**終了**ステートメントでテストの該当セクションを囲むことで，テスト内のトランザクションを定義できます。たとえば，飛行機の座席を予約するのにかかる時間や，クライアントの端末に確認メッセージが表示されるまでにかかる時間を測定するトランザクションを定義できます。

テストの実行中，**StartTransaction** ステップは，時間測定の開始を示します。時間測定は，**EndTransaction** ステップに到達するまで継続されます。テスト・レポートには，トランザクションの実行にかかった時間が表示されます。

トランザクション内で使用するステートメントの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

テストに追加できるトランザクションの数に制限はありません。トランザクション内にトランザクションを挿入することも可能です。

次に，トランザクションを含むサンプル・テストの一部を，キーワード・ビューに表示されるとおりに示します。



エキスパート・ビューではテストのサンプル部分は次のように表示されます。

```
Services.StartTransaction "ReserveSeat"
Browser("Welcome:Mercury Tours").Page("Find a Flight:Mercury").
WebList("fromPort").Select "London"
Browser("Welcome:Mercury Tours").Page("Find a Flight:Mercury").
WebList("toPort").Select "Frankfurt"
Browser("Welcome:Mercury Tours").Page("Find a Flight:Mercury").
WebList("toDay").Select "12"
Browser("Welcome:Mercury Tours").Page("Find a Flight:Mercury").
WebRadioGroup("servClass").Select "Business"
Browser("Welcome:Mercury Tours").Page("Find a Flight:Mercury").
WebList("airline").Select "Blue Skies Airlines"
Browser("Welcome:Mercury Tours").Page("Find a Flight:Mercury").
Image("findFlights").Click 65,12
Browser("Welcome:Mercury Tours").Page("Select a Flight:Mercury").
WebRadioGroup("outFlight").Select "Blue Skies Airlines"
Browser("Welcome:Mercury Tours").Page("Select a Flight:Mercury").
WebRadioGroup("inFlight").Select "Blue Skies Airlines"
Browser("Welcome:Mercury Tours").Page("Select a Flight:Mercury").
Image("reserveFlights").Click 46,8
Services.EndTransaction "ReserveSeat"
```

ステップ・ジェネレータまたはエキスパート・ビューを使用すると，さまざまなトランザクション関連ステートメントを挿入できます。詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』の「**Services**」の節を参照してください。「トランザクション開始」ステップおよび「トランザクション終了」ステップは，QuickTest ウィンドウ内のオプションを使用して入力することもできます。

詳細については，次を参照してください。

- 「トランザクションの挿入」
- 413 ページ「トランザクションの終了」

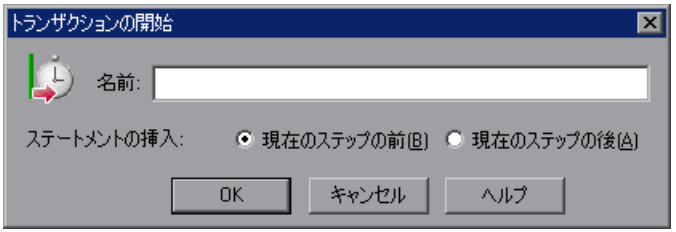
トランザクションの挿入

テストの実行中，「トランザクションの開始」は，時間測定の開始を示します。［トランザクションの開始］ダイアログ・ボックスでトランザクションの開始を定義します。

トランザクションを挿入するには，次の手順を実行します。

1 トランザクションの時間測定を開始するステップをクリックします。［ActiveScreen］タブに，対応するページが表示されます。

2 ［**トランザクションの開始**］ボタンをクリックするか，［**挿入**］＞［**トランザクションの開始**］を選択します。［**トランザクションの開始**］ダイアログ・ボックスが開きます。



3 ［**名前**］ボックスに，分かりやすい名前を入力します。

注：トランザクション名にスペースを含めることはできません。

---

4 トランザクションの時間測定を開始する場所を決定します。

- 現在のステップの前でトランザクションを終了するには，［**現在のステップの前**］を選択します。
- 現在のステップの後でトランザクションを終了するには，［**現在のステップの後**］を選択します。

5 ［**OK**］をクリックします。**StartTransaction** ステップがキーワード・ビューに追加されます。

トランザクションの終了

テストの実行中，「トランザクションの終了」で時間測定の終了を示します。［トランザクションの終了］ダイアログ・ボックスでトランザクションの終了を定義します。

注：実行セッション中にエラーが発生した場合でもトランザクション内のすべてのステップを実行するよう QuickTest を設定したい場合があります。これには，［テストの設定］ダイアログ・ボックス（［**ファイル**］＞［**設定**］）の［実行］タブで，［**実行セッション中にエラーが発生した場合**］リストから［**次のステップに進む**］を選択します。

トランザクションを終了するには，次の手順を実行します。

1 トランザクションの時間測定を終了するステップをクリックします。［ActiveScreen］タブに対応するページが表示されます。

2 ［**トランザクションの終了**］ボタンをクリックするか，［**挿入**］＞［**トランザクションの終了**］を選択します。［**トランザクションの終了**］ダイアログ・ボックスが開きます。

3 ［名前］ボックスには，現在のテストでユーザが定義したトランザクション名のリストが含まれています。終了するトランザクションの名前を選択します。

4 トランザクションを終了する場所を決めます。

- 現在のステップの前でトランザクションを終了するには，［**現在のステップの前**］を選択します。
- 現在のステップの後でトランザクションを終了するには，［**現在のステップの後**］を選択します。

5 ［**OK**］をクリックします。**EndTransaction** ステップがキーワード・ビューに追加されます。

サイレント・テスト・ランナーの使用

サイレント・テスト・ランナーを使用して，LoadRunner および Business Availability Center からの QuickTest テストの実行方法をシミュレートできます。サイレント・テスト・ランナーを使用してテストを実行すると，QuickTest のユーザ・インタフェースを開かずにサイレント・テスト・ランナーが起動し，LoadRunner または Business Availability Center から実行したときと同じ速度でテストが実行されます。テスト実行の最後では，テスト実行とトランザクション回数に関する情報を表示できます。

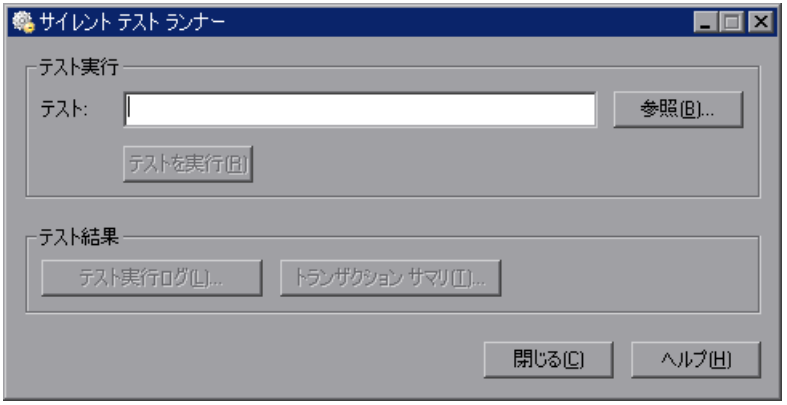
また，サイレント・テスト・ランナーを使用して，QuickTest テストが LoadRunner および Business Availability Center と互換性があることを確かめることもできます。サイレント・テスト・ランナーを使用してテストを実行するときに，LoadRunner または Business Availability Center でサポートされていない機能を使用している場合，テストは失敗します。サポートされていない機能の詳細については，406 ページ「LoadRunner または Business Process Monitor で使用する QuickTest テストの設計」および 408 ページ「LoadRunner または Business Process Monitor でのテストの挿入と実行」を参照してください。

注：QuickTest がすでに開いている場合や，別のテストが現在実行中の場合は，サイレント・テスト・ランナーを実行できません。QuickTest を閉じ，そのプロセスが終了するまで待ってから，サイレント・テスト・ランナーを使用してテストを実行する必要があります。

Silent Test Runner でテストを実行しているときは，**ResultDir** QuickTest 環境変数を使用することはできません。

サイレント・テスト・ランナーを使用して QuickTest テストを実行するには，次の手順を実行します。

1 サイレント・テスト・ランナーを開くには，［**スタート**］ボタンから［**QuickTest Professional**］プログラム・グループを開き，［**Tools**］＞［**Silent Test Runner**］を選択します。［サイレント テスト ランナー］ダイアログ・ボックスが開きます。



2 テストの場所まで移動するために［**参照**］ボタンをクリックします。［テストを開く］ダイアログ・ボックスが開き，**＜ QuickTest Professional ＞ ¥Tests** フォルダにあるテストが表示されます。

3 実行するテストを選択し，［**開く**］をクリックします。［テストを開く］ダイアログ・ボックスが閉じ，テスト名が［サイレント テスト ランナー］ダイアログ・ボックスの［**テスト**］ボックスに表示され，［**テストを実行**］ボタンが有効になります。

注：以前に実行したテストを選択した場合は，［**テスト実行ログ**］ボタンと［**トランザクション サマリ**］ボタンが有効になり，選択したテストの前回の実行に関する情報を表示できます。テストを初めて実行する場合は，［**テスト実行ログ**］ボタンと［**トランザクション サマリ**］ボタンは無効になっています。

---

4 ［**テストを実行**］をクリックしてテストを実行します。QuickTest ユーザ・インタフェースを開かずにテストが実行されます。テストの実行中は，「**テストを実行しています ...**」というテキストが［**テストを実行**］ボタンの横に表示されます。

注 : テストの実行を開始した後に，サイレント・テスト・ランナーからテストの実行を停止することはできません。サイレント・テスト・ランナーを閉じても，テストの実行は継続されます。テストを終了するには，**mdrv.exe** プロセスを終了します。

---

5 テストの実行が完了すると，テキスト「**テストを実行しています ...**」が「**テスト実行が完了しました**」に変わります。サイレント・テスト・ランナーがテストを実行できなかった場合は，「**テストを実行できませんでした**」というテキストが表示されます。［**テスト実行ログ**］ボタンが無効であった場合は，有効になります。トランザクションのあるテストを実行した場合，［**トランザクション サマリ**］ボタンが無効であった場合は，そのボタンも有効になります。ログ・ファイルの表示の詳細については，「テスト結果情報の表示」を参照してください。

テスト結果情報の表示

サイレント・テスト・ランナーでは，テストの実行情報がログ・ファイルに記録されます。テストごとにテスト実行ログが生成され，トランザクションのあるテストの場合は追加のトランザクション・サマリが生成されます。

テスト実行ログの表示

テスト実行ログは，**＜ QuickTest Professional ＞ ¥Tests¥ ＜テスト名＞**フォルダに **output.txt** として保存されます。ログ・ファイルはサイレント・テスト・ランナーによるテスト実行のたびに保存され，テストを再実行すると上書きされます。ログ・ファイルを開くには，［**テスト実行ログ**］をクリックします。

ログ・ファイルにはテストの実行に関する情報が表示されます。たとえば，個々の反復，アクション呼び出し，ステップ・トランザクション，失敗したステップなどに関する情報が表示されます。各行にはメッセージまたはエラー ID が表示されます。ログ・ファイル内のメッセージとエラー・コードの詳細については，Performance Center または Business Availability Center のマニュアルを参照してください。

トランザクション・サマリの表示

トランザクション・サマリは，**＜QuickTest Professional＞¥Tests¥＜テスト名＞**フォルダに **transactions.txt** として保存されます。トランザクション・サマリはトランザクションを含むテストごとに保存され，テストを再実行すると上書きされます。トランザクション・サマリを開くには，［**トランザクション サマリ**］をクリックします。トランザクション・サマリには，テスト内の各トランザクションに対応した行が表示されます。各トランザクションについて，ステータスと，総継続時間および浪費時間（秒単位）が表示されます。サイレント・テスト・ランナーでのトランザクション測定は，テストを LoadRunner または Business Availability Center から実行した場合とまったく同じになります。

注：

トランザクション・サマリは，**EndTransaction** ステートメントで終わるトランザクションが含まれているテストについてのみ生成されます。トランザクションが開始されたものの，テストの失敗のために終了しなかった場合は，そのトランザクションはトランザクション・サマリに記録されません。

分散トランザクション（あるテストで開始し，別のテストで終了するトランザクション）は，トランザクション・サマリには報告されず，テスト実行ログに記録されます。

トランザクション・サマリに記録されているトランザクション情報はすべて，テスト実行ログにも記録されます。

第 5 部

付録

付録 A

QuickTest を使用した作業ーよくある質問

本章では，QuickTest の**上級ユーザ**から寄せられることの多いいくつかの質問についてお答えします。質問と回答は次の項に分類されています。

- テストの記録と実行
- エキスパート・ビューでのプログラミング
- 動的なコンテンツを使用した作業
- Web に関する高度な問題
- テストの保守
- ローカライズされたアプリケーションのテスト
- QuickTest のパフォーマンスの向上

テストの記録と実行

➤ **QuickTest は，どのようにして Web ページのユーザ・プロセスをキャプチャするのでしょうか。**

QuickTest では，Microsoft Internet Explorer ブラウザにフックをかけます。ユーザが Web ベースのアプリケーションを操作すると，QuickTest によって，ユーザのアクションが記録されます（記録されるユーザ・アクションを変更する方法の詳細については，第 10 章「Web イベント記録の設定」を参照してください）。記録したテストは，QuickTest を使用して元の順序のままステップを実行できます。

➤ **QuickTest でサポートされていないオブジェクトまたは環境で記録を実行するには，どのようにすればよいですか。**

さまざまな方法があります。

- ➤ QuickTest では標準設定で，いくつかの開発環境がサポートされます。QuickTest Professional で使用可能な任意の外部アドインをインストールしてロードすることによって，Java，Oracle，.NET，SAP Solutions，Siebel，Web サービスなどの環境をさらにサポートできます。
- ➤ 識別されなかったクラスやユーザ定義のクラスのオブジェクトは，標準の Windows クラスにマップできます。オブジェクトのマッピングの詳細については，109 ページ「ユーザ定義のテスト・オブジェクト・クラスの割り当て」を参照してください。
- ➤ テスト・オブジェクトと同じように振る舞うオブジェクトに**仮想オブジェクト**を定義して，通常の記録モードで記録できます。仮想オブジェクトの詳細については，第 2 章「仮想オブジェクトの学習」を参照してください。
- ➤ **低レベル記録**あるいは**アナログ・**モードで，座標に基づいてクリックとキーボード入力を記録できます。低レベル記録とアナログ記録の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 96 ページ「記録モードの選択」を参照してください。

エキスパート・ビューでのプログラミング

➤ **関数やサブルーチンを関数ライブラリに保存できますか。**

関数は個々のテスト内で定義できます。または，関数が含まれる 1 つまたは複数の VBScript ライブラリ・ファイルを作成できます。そして，任意のテストからそれらを呼び出すことができます。

関数を QuickTest テスト・オブジェクトのメソッドとして登録することもできます。登録したメソッドは，実行セッションの間だけ既存のテスト・オブジェクト・メソッドの機能をオーバーライドしたり，テスト・オブジェクト・クラスの新しいメソッドとして登録したりできます。

詳細については，第 6 章「ユーザ定義関数および関数ライブラリを使用した作業」を参照してください。

動的なコンテンツを使用した作業

➤ **表示するたびに動的に変化するオブジェクトを対象としたテストを記録し，実行するにはどうすればよいでしょうか。**

Web ページまたはアプリケーション内のオブジェクトで動的コンテンツを持つものは内容が変化することがあります。これらのオブジェクトを示す動的な記述を作成することで，QuickTest がテストを実行する際にそれらのオブジェクトを認識するようにできます。詳細については，**『QuickTest Professional 基本機能ユーザーズ・ガイド』**の第 6 章「テスト・オブジェクトを使用した作業」を参照してください。

➤ **子ウィンドウの有無を検査するには，どうすればよいでしょうか。**

あるウィンドウ内のリンクが別のウィンドウを作成することがあります。

Exist プロパティを使用して，ウィンドウが存在するかどうかをチェックすることができます。次に例を示します。

```
Browser("Window_name").Exist
```

また，**ChildObjects** メソッドを使用して，テスクトップ上あるいはほかの親オブジェクト内のすべての子オブジェクト（または，特定の記述に一致する子オブジェクトのサブセット）を取得することもできます。

Exist プロパティと **ChildObjects** メソッドの詳細については，『**QuickTest Professional オブジェクト・モデル・リファレンス**』を参照してください。

➤ **QuickTest は，動的に生成される URL や Web ページをどのようにして記録するのでしょうか。**

QuickTest は，リンクがページに表示されると，実際にそのリンクをクリックします。そのため，QuickTest はオブジェクト自体ではなく，ページ上のリンクなど特定のオブジェクトを検索する方法を記録します。たとえば，動的に生成された URL へのリンクが画像である場合，QuickTest は「IMG」HTML タグと，その画像の名前を記録します。これにより，それ以後 QuickTest はこの画像を検索し，その画像をクリックできるようになります。

Web に関する高度な問題

➤ **QuickTest はクッキーをどのように処理するのでしょうか。**

CGI スクリプトなど接続のサーバ側では，クッキーを利用することで，接続のクライアント側に情報を格納したり，そこから情報を取得したりできます。

QuickTest ではユーザごとにメモリにクッキーを格納し，ブラウザは通常どおりにそれらを処理します。

➤ **QuickTest は，セッション ID をどのように処理するのでしょうか。**

ブラウザでなくサーバが，通常はクッキーによって，またはすべてのリンクにセッション ID を埋め込むことによって，セッション ID を処理します。これは，QuickTest には影響を与えません。

➤ **QuickTest は，サーバのリダイレクトをどのように処理するのでしょうか。**

サーバがクライアントをリダイレクトした場合，クライアントは一般にそれに気付くことはなく，リダイレクトの間違いが起こることはありません。ほとんどの場合，クライアントはサーバ上の別のスクリプトにリダイレクトされます。この追加のスクリプトが，以降に表示されるページの HTML コードを生成します。これは，QuickTest にもブラウザにも影響を与えません。

➤ **QuickTest は，META タグをどのように処理するのでしょうか。**

META タグは，ページの表示に影響を与えません。META タグには通常，ページの作成者，更新頻度，ページの内容説明，およびページの内容を表すキー

ワードの情報だけが含まれています。したがって，QuickTest は問題なく META タグを処理できます。

➤ **QuickTest は .asp に対応していますか。**

Active Server Page テクノロジを使用して動的に生成される Web ページには，.asp という拡張子が割り当てられています。これは完全にサーバ側の技術であるため，QuickTest には影響しません。

➤ **QuickTest は，COM に対応していますか。**

QuickTest は，COM 標準に準拠しています。

QuickTest は，Web ページに埋め込まれた COM オブジェクトをサポートしており（現在，COM オブジェクトは Microsoft Internet Explorer を使用している場合にだけアクセス可能です），VBScript 内で COM オブジェクトを操作できます。

➤ **QuickTest は XML に対応していますか。**

XML（eXtensible Markup Language）は，Web ドキュメント用に SGML を簡略化したものです。XML を使えば，Web デザイナーはカスタマイズした独自のタグを作成できます。QuickTest は XML に対応しており，XML タグをオブジェクトとして認識します。

また，Web ページ，Web フレーム，Web ファイルの XML ドキュメントの内容を検査する XML チェックポイントを作成できます。QuickTest は XML 出力とスキーマ検証もサポートしています。

詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 13 章「XML の検査」を参照してください。

テストの保守

➤ **アプリケーションに変更が加えられた場合，テストをどのように保守すればよいですか。**

アプリケーションに変更が加えられた場合のテストの保守方法は，アプリケーションに加えられた変更の量に応じて異なります。アプリケーション全体を対象に 1 つの大きなテストを作成するのではなく，テストを小さなグループに分けて作成するべき主な理由の 1 つがこれです。アプリケーションに変更が加えられた場合に，テストの一部だけを再記録できます。大きな変更でない場合は，テストを手作業で編集して更新できます。

また，QuickTest のアクションの機能を使用して，よりモジュール化され，より効果的なテストを設計できます。記録中，テストを機能ごとにいくつかのアクションに分割します。アプリケーションに変更が加えられたら，特定のアクションだけを再記録するだけでよく，テストのほかの部分は変更せずに済みます。可能なかぎり，複数のテストにまったく同じスクリプトを作成するのではなく，再利用可能なアクションの呼び出しを挿入するようにします。こうすることで，元の再利用可能なアクションに変更を加えるだけで，そのアクションを呼び出すすべてのテストに変更が自動的に適用されます。詳細については，第 1 章「高度なアクション機能を使用した作業」を参照してください。

同じテスト・オブジェクトが含まれるテストやアクションが数多くある場合は，1 か所で集中的にオブジェクト情報を更新できるように，共有オブジェクト・リポジトリを使用することをお勧めします。

オブジェクトのプロパティが変更されたときに，チェックポイント，ActiveScreen，テスト・オブジェクト・プロパティの情報を更新するには，あるいは，ステップを再記録せずに ActiveScreen 画像に新規のオブジェクトやステップを追加するには，［**更新モード**］オプションを使用します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 606 ページ「テストの更新」を参照してください。

➤ **テストの記録終了後，ActiveScreen 情報を増やしたり減らしたりできますか。**

記録後に，ActiveScreen に保存された情報が，テストを編集するには不十分な場合や，ActiveScreen 情報が不要になり，テストのサイズを小さくする場合に，テストに保存されている ActiveScreen 情報の量を変更する方法はいくつかあります。

- テストによって使用されるディスク容量を減らすには，［**名前を付けて保存**］を選択して［**ActiveScreen ファイルを保存する**］チェックボックスをクリアにすることで ActiveScreen 情報を削除します。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 106 ページ「テストの保存」を参照してください。
- Windows または Java アプリケーションの記録時に ActiveScreen の情報をすべて保存しないことを選択した場合，次のいずれかの方法を使用して，ActiveScreen に格納される情報を増やせます。

  ［オプション］ダイアログ・ボックス（［ツール］ > ［オプション］）の［ActiveScreen］タブで，ActiveScreen のキャプチャ設定が必要な量の情報をキャプチャするように設定されているか確認します。次に以下のことを行います。

- ［**更新モード**］を実行し，既存のすべてのステップについて必要な量の情報を ActiveScreen に保存します。［**更新モード**］オプションの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 606 ページ「テストの更新」を参照してください。
- ActiveScreen に追加するオブジェクトが含まれるステップを再記録します。

  ステップを再記録するには，記録するステップの「**前の**」ステップを選択し，テスト内の選択した位置と一致するようにアプリケーションを配置してから，記録を開始します。あるいは，追加するステップの「**前の**」ステップでテストにブレークポイントを設定し，そのブレークポイントまでテストを実行します。これにより，ステップを記録するのに適切な場所へ移動できます。ブレークポイントの設定の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 588 ページ「ブレークポイントの設定」を参照してください。

Windows アプリケーションについて ActiveScreen に保存される情報量の変更方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 697 ページ「グローバル・テスト・オプションの設定について」を参照してください。

ローカライズされたアプリケーションのテスト

➤ **あるアプリケーションについて，ローカライズされたいくつかのバージョンをテストしています。それぞれのバージョンには，ローカライズされたユーザ・インタフェース文字列が含まれています。QuickTest で効率的なテストを作成するには，どのようにすればよいですか。**

これらのユーザ・インタフェース文字列は，グローバル環境変数リストにあるパラメータを使ってパラメータ化できます。グローバル環境変数リストは，変数と，それに対応する値のリストで，任意のテストからアクセスできます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 15 章「値のパラメータ化」を参照してください。

➤ **あるアプリケーションについて，ローカライズされたいくつかのバージョンをテストしています。テストの際に，アプリケーションの言語に応じて異なるデータを効率的に入力するにはどうすればよいですか。**

テストの反復を 1 回だけ実行する場合，あるいはアクションまたはテストのすべての反復で変数の値を変えない場合は，環境変数を使用し，テストの実行ごとにアクティブな環境変数ファイルを切り替えます。

テストまたはアクションの反復を複数回実行し，入力データを反復ごとに変更する場合は，アプリケーションのローカライズ・バージョンごとに外部データ・テーブルを作成します。テスト対象を別のローカライズされたバージョンに変更するときには，［テストの設定］ダイアログ・ボックスの［リソース］タブでテスト用のデータ・テーブル・ファイルを別のデータ・テーブル・ファイルに切り替えます。

データ・テーブルの操作の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 19 章「データ・テーブルを使った作業」を参照してください。テストを対象とするデータ・テーブル・ファイルの選択方法の詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 755 ページ「テストのためのリソース設定の定義」を参照してください。

QuickTest のパフォーマンスの向上

➤ **QuickTest の動作速度を向上させるには，どうすればよいですか。**

QuickTest の動作速度を向上させるには，次のいずれも実行できます。

- ➤ QuickTest が起動する際に，アドイン・マネージャの中で，不必要なアドインが読み込まれないようにします。これにより，記録時間と実行セッションのパフォーマンスの両方を向上させることができます。アドインの読み込みの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 799 ページ「QuickTest アドインのロード」を参照してください。
- ➤ テストを「高速」モードで実行します。それには，［オプション］ダイアログ・ボックスの［実行］タブで，［**高速**］オプションを選択します。これにより，QuickTest は各ステップで実行矢印を表示せずにテストを実行するため，テストの実行を高速化できます。［オプション］ダイアログ・ボックスの［実行］タブの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 713 ページ「テストの実行オプションの設定」を参照してください。

➤ テストを編集する際に ActiveScreen を使用していない場合は，テストの編集中に ActiveScreen を非表示にしておくことで，編集時の応答時間を改善します。これを行うには，［**表示**］＞［**ActiveScreen**］を選択するか，または［ActiveScreen］ツールバー・ボタンをクリックして，ActiveScreen を非表示に切り替えます。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 2 章「QuickTest の概要」を参照してください。

➤ 情報をキャプチャして ActiveScreen に保存する条件とその量を指定します。キャプチャする情報が多いほど，さまざまな ActiveScreen オプションを使用してテストにステップを追加するのが容易になります。しかし，キャプチャする情報が多いと，記録や編集を行うのに時間がかかるようになります。パフォーマンスを向上させるために選択できる ActiveScreen オプションは，次のとおりです。

- Windows アプリケーションをテストしている場合は，あらゆるステップの ActiveScreen 情報をすべて保存する，特定のステップの ActiveScreen 情報だけを保存する，ActiveScreen のキャプチャを完全に無効化する，といった選択ができます。この設定は，［オプション］ダイアログ・ボックスの［ActiveScreen］タブで行います。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 705 ページ「ActiveScreen オプションの設定」を参照してください。

- Web アプリケーションをテストしている場合は，ActiveScreen でのすべてのステップの画面キャプチャを無効にします。［オプション］ダイアログ・ボックスの［ActiveScreen］タブで，［**ユーザ定義レベル**］をクリックして［Active Screen キャプチャのユーザ定義設定］ダイアログ・ボックスを開きます。

  次に，［**ActiveScreen のキャプチャを無効にする**］オプションを選択します。これにより，記録時間を短縮できます。［オプション］ダイアログ・ボックスの［ActiveScreen］タブの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 705 ページ「ActiveScreen オプションの設定」を参照してください。

- 新規テストを保存する場合，または［**名前を付けて保存**］を使用してテストを新しい名前で保存する場合は，［テストの保存］ダイアログ・ボックスの［**ActiveScreen ファイルを保存する**］オプションをクリアにすることで，テストでキャプチャされた ActiveScreen ファイルを保存しないように選択します。これは，テストの設計が完了し，テスト実行のためだけにテストを使用する場合に特に便利です。ActiveScreen ファイルのないテストは，速く開き，占有するディスク容量が極めて少なくなります。

➤ テスト結果としてアプリケーションの画像をキャプチャし，保存するタイミングを指定します。それには，［オプション］ダイアログ・ボックスの［実行］タブにある［**ステップ画面キャプチャの保存先テスト結果**］ボックスでオプションを選択します。特定の条件を満たした場合にだけ画面キャプチャを保存したり，画像を保存しないようにしたりすることで，テストの実行時間を短縮し，ディスク領域を節約できます。［オプション］ダイアログ・ボックスの［ActiveScreen］タブの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 705 ページ「ActiveScreen オプションの設定」を参照してください。

ヒント：ActiveScreen ファイルなしでテストを保存した後で ActiveScreen ファイルを回復する必要が生じた場合は，必要なステップを再び記録するか，［**更新モード**］オプションを使用してテストのすべてのステップの画面を再キャプチャします。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 606 ページ「テストの更新」を参照してください。

➤ **QuickTest が使用するディスク容量を減らすには，どうすればよいですか。**

QuickTest が使用するディスク容量を減らすには，次のいずれかを実行します。

➤ テスト結果としてアプリケーションの画像をキャプチャし，保存するタイミングを指定する。それには，［オプション］ダイアログ・ボックスの［実行］タブにある［**ステップ画面キャプチャの保存先テスト結果**］ボックスでオプションを選択します。特定の条件を満たした場合にだけ画面キャプチャを保存したり，画像を保存しないようにしたりすることで，ディスク容量を節約し，テストの実行時間を短縮できます。［オプション］ダイアログ・ボックスの［ActiveScreen］タブの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 705 ページ「ActiveScreen オプションの設定」を参照してください。

➤ 情報をキャプチャして ActiveScreen に保存する条件とその量を指定します。キャプチャする情報が多いほど，さまざまな ActiveScreen オプションを使用してテストにステップを追加するのが容易になります。しかし，キャプチャする情報が多いと，記録や編集を行うのに時間がかかるようになります。パフォーマンスを向上させるために選択できる ActiveScreen オプションは，次のとおりです。

- Windows アプリケーションをテストしている場合は，あらゆるステップの ActiveScreen 情報をすべて保存する，特定のステップの ActiveScreen

情報だけを保存する，ActiveScreen のキャプチャを完全に無効化する，といった選択ができます。この設定は，［オプション］ダイアログ・ボックスの［ActiveScreen］タブで行います。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 705 ページ「ActiveScreen オプションの設定」を参照してください。

- Web アプリケーションをテストしている場合は，ActiveScreen でのすべてのステップの画面キャプチャを無効にします。［ActiveScreen］タブで，［**ユーザ定義レベル**］をクリックして［Active Screen キャプチャのユーザ定義設定］ダイアログ・ボックスを開きます。次に，［**ActiveScreen のキャプチャを無効にする**］オプションを選択します。これにより，記録時間を短縮できます。［オプション］ダイアログ・ボックスの［ActiveScreen］タブの詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 705 ページ「ActiveScreen オプションの設定」を参照してください。
- 新規テストを保存する場合，または［名前を付けて保存］を使用してテストを新しい名前で保存する場合は，［テストの保存］ダイアログ・ボックスの［**ActiveScreen ファイルを保存する**］オプションをクリアにすることで，テストでキャプチャされた ActiveScreen ファイルを保存しないように選択します。これは，テストの設計が完了し，テスト実行のためだけにテストを使用する場合に特に便利です。ActiveScreen ファイルのないテストは，使用するディスク容量が極めて少なくなります。

ヒント：ActiveScreen ファイルなしでテストを保存した後で ActiveScreen ファイルを回復する必要が生じた場合は，必要なステップを再び記録するか，［**更新モード**］オプションを使用してテストのすべてのステップの画面を再キャプチャします。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の 606 ページ「テストの更新」を参照してください。

➤ 推奨されるテストの長さはありますか。

テストの長さについて公式の制限はありませんが，テストをアクションに分割し，可能な限り，再利用可能なアクションを使用することを推奨します。アクションに含むステップ数は，200 ～ 300 までにしてください。数十までが理想的です。詳細については，『**QuickTest Professional 基本機能ユーザーズ・ガイド**』の第 17 章「アクションを使った作業」を参照してください。

索引

え

お

か

き

く

け

こ

さ

し

す

せ

た

ち

つ

て

と

な

に

は

ま

み

む

め

や

ゆ

よ

り

れ

ろ

わ