実行再生エラーの対応ガイド

TestComplete 8.10

Rev. 2 - 2011 年 2 月 17 日

Rev. 1 - 2011 年 2 月 10 日

エクセルソフト株式会社

目 次

実行再生エラーの対応

このガイドブックは、テスト実行中に発生する一般的な問題の解決方法について説明します。

ヘルプ目次から [Using TestComplete] - [Handling Playback Errors] を選択して表示される [Handling Playback Errors] 章の翻訳です。

オリジナルの英語版 オンラインヘルプは、[スタート]ボタンから[プログラム] - [AutomatedQA] - [TestComplete] - [Documentation] - [TestComplete 8 Help]を選択して表示できます。

注意: 以下に記載されているトピックは、そのテスト中のエラーについてのみ説明します。テストしているアプリケーションで発生するエラーをどのように追跡するかは説明していません。アプリケーション エラーの追跡に関する詳細情報は、" Tracing Exceptions, Crashes and Freezes in Tested Applications (テストするアプリケーションでの例外、クラッシュ、フリーズの追跡) "セクションを参照してください。

エラーの原因を見つける
問題を診断するための一般的な情報を提供します。

予期しないウィンドウの対応
テストしているアプリケーションがテストで予期していないウィンドウまたはダイアログを表示するときの状態に対応する方法を説明します。

“Object Not Found”(オブジェクトが見つかりません) エラーの対応
テスト エンジンがテストが参照しているオブジェクトを見つけられないときに発生するエラーの対応する方法を説明します。

“Incomplete Keyboard Input”(キーボード入力が完了しない) 問題の解決
TestComplete がテストしているアプリケーションのキーの押下を送信できないときの問題を解決する方法を説明します。

エラーの原因を見つける

テストが正常に実行しないことがあります。この理由はいろいろあります。たとえば、アプリケーションの振る舞いが予期しているものと異なる、予期しないウィンドウがテスト実行中に表示される、テストスクリプトに誤った命令が含まれている、などです。以下の方法でエラーの原因を見つけることができます:

- テストログの最初のエラーを見てください。実行されたテストコマンドに依存しますが、TestComplete はログにいくつかのエラーメッセージを出力します。一般的に最初のエラーが問題の原因に関連し、そのあとのエラーは最初のエラーにより引き起こされています。
- テストログの Additional Information (補足情報) パネルを良く見てください。エラーに関する広範囲な情報を提供しています。
- エラーが発生したときに実行していたテストコマンドを表示するには、ログのエラーメッセージをダブルクリックします。TestComplete がテスト エディタを開き、そのテストコマンドをハイライト表示します。エラーが発生したときに何が実行されたか理解するためにそのテストを注意深く見てください。
- テスト実行で Test Visualizer が有効になっている場合、Test Visualizer がキャプチャーし、テストログに出力した画像を見てください。実行中にアプリケーションが何をしていたか、予期していた振る舞いまたは状態と異なるかどうか理解する助けになります。
- エラーが発生するときのアプリケーションの状態を分析するために、もう一度テストを実行します。この分析を実行するために、次のようにします--
    - 問題のあるテスト行にブレークポイントを設定します。これをするためには、その行の左端の空間をクリックします。テストエンジンがブレークポイントでテスト実行を自動的に一時停止します。一時停止中に、テスト オブジェクトと変数の値を見るために、Watch List (ウォッチリスト) または Locals (ローカル) パネル、Evaluate (評価) ダイアログを使用できます。(Debugging Tests - Overview を参照)。
    - 実行のために Test Visualizer を有効にします。この場合、TestComplete は、ユーザー操作のスクリーンショットをキャプチャーし、テストログに保存します。これらのスクリーンショットはアプリケーションの状態と振る舞いを理解することを簡単にします。Test Visualizer を有効にする方法は、Test Visualizer Overview を参照してください。
    - 異なる診断メッセージをログに出力するテストコマンドを追加します。たとえば:
        - 子オブジェクト(ウィンドウ)のリスト
        - ひとつまたは複数のウィンドウの画像
        - オブジェクト(ウィンドウ)の状態: 存在、可視性、など。たとえば、オブジェクトまたはウィンドウの存在を見るために、Exists プロパティを使用します; ウィ

ンドウまたはコントロールの可視性を見るには、Visible および VisibleOnScreen プロパティを使用できます; オブジェクト(ウィンドウ)が有効かフォーカスされているか見るには、Enabled および Focused プロパティを使用します。

分析の目的は、以下の質問に答えることです:

- 現在のアプリケーションの状態と期待する状態との違いは何か?
- エラーの原因は何か? スクリプト内のエラーまたは記述ミス、アプリケーションの不具合、あるいはその他?

エラーの原因を判断でいない場合、メッセージをテクニカルサポートに送付してください。テクニカルサポートがメッセージに回答するには、スクリプトと TestComplete のプロジェクトの結果フォルダを ZIP または LZH圧縮して添付してください。(プロジェクトの結果フォルダは、通常、<Your_Project_Folder>¥Log ディレクトリにあります)。エラー時のアプリケーションの状態の詳細な情報が分かるように、スクリプトを修正してください: アプリケーション ウィンドウの画像、テスト プロセスの子オブジェクトのリスト、など。問題を再現できる小さなサンプルアプリケーションがあれば、さらに助かります。

また、問題についてニュースグループや FAQ を検索することも有効です。詳細については、Technical Support and Resources を参照してください。

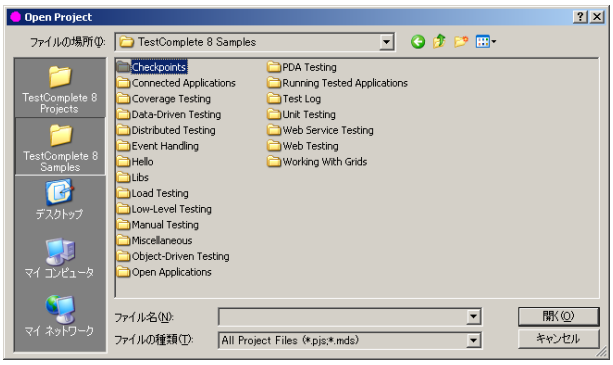
開発元（SmartBear Software 社）のサポートページ
http://www.automatedqa.com/support

エクセルソフトの サポートページ
http://www.xlsoft.com/jp/products/smartbear/support.html

簡単なサンプル プロジェクトでのエラー修正

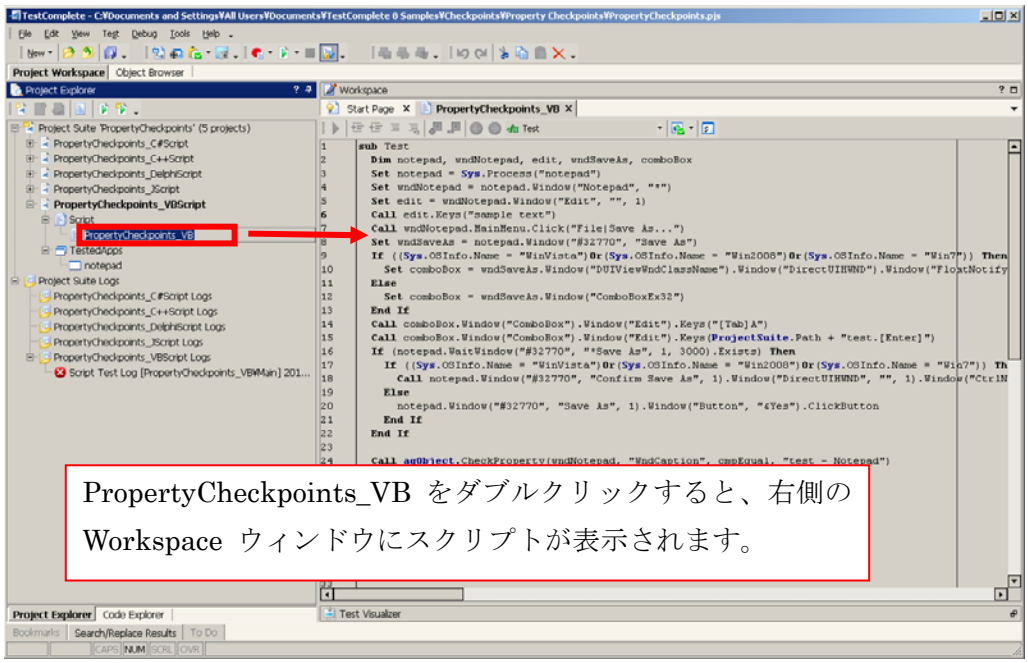
TestComplete 8 に含まれるサンプル プロジェクトを使用して、簡単なエラーの修正をしてみます。以下の操作手順に沿って、サンプルプロジェクトを再生実行します。

1. TestComplete を起動します。
2. **[File] - [Open]** メニューを選択します。**[Open Project]** ダイアログが開きます。

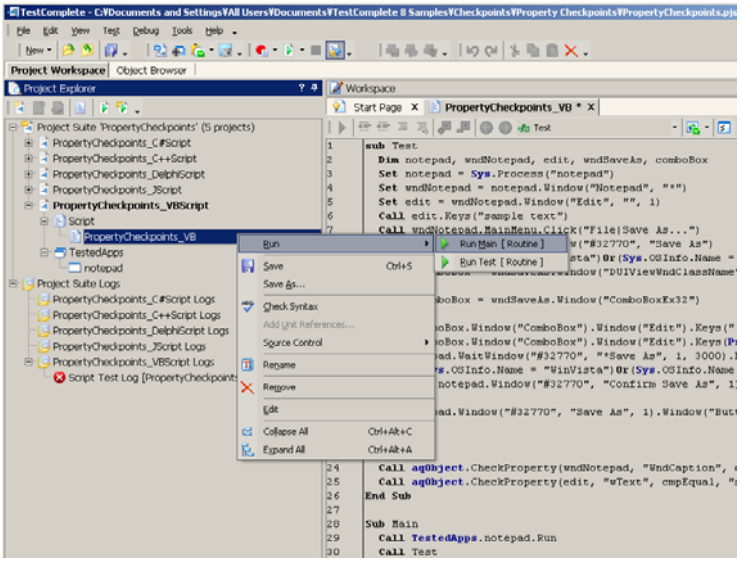


このダイアログの左側の TestComplete 8 Sample をクリックします。一覧から "Checkpoints" フォルダ、次に "Property Checkpoints" フォルダの順にクリックし、**"PropertyCheclpoints.pjs"** プロジェクト ファイルを選択します。PropertyCheckpoints プロジェクトが開かれます。

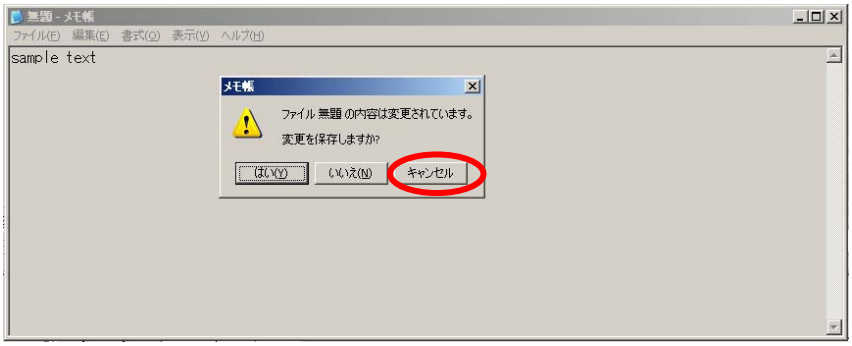
3. 左側の Project Explorer ウィンドウには、スクリプト言語ごとに 5 つのプロジェクトがあります。ここでは、例として **VBScript** を実行します。



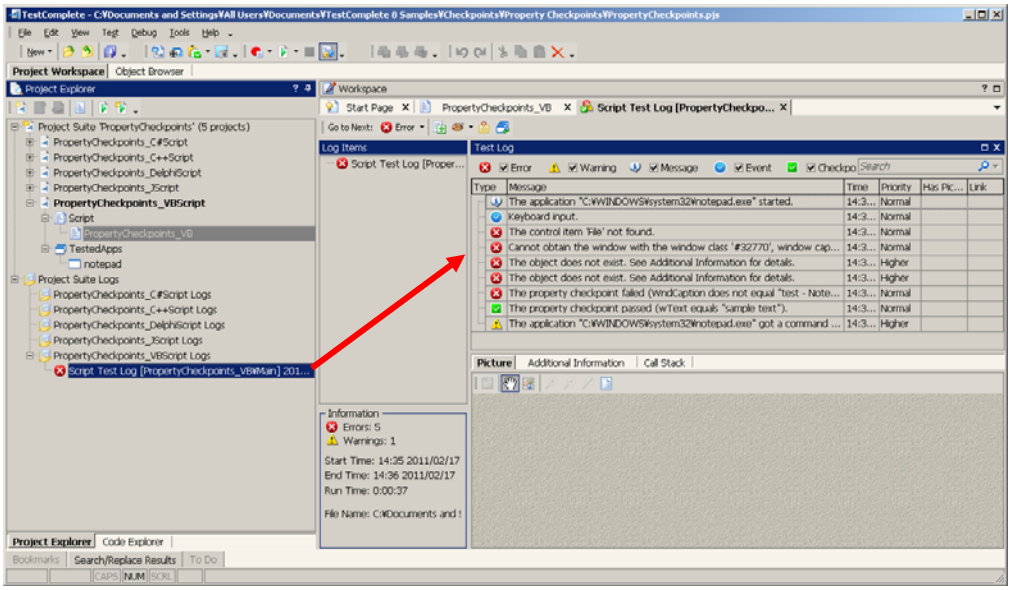
4. Project Explorer ウィンドウの VBScript を実行します。**ProjectCheckpoints_VB** を右クリックして、ポップアップ メニューから **[Run] -[Run Main [Routine]]** を選択します。



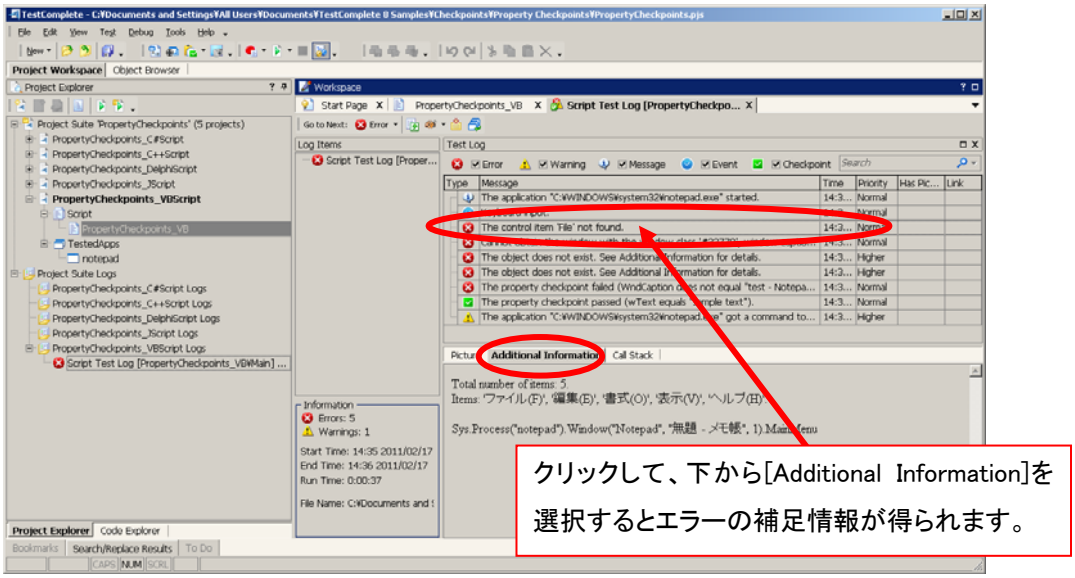
5. メモ帳が実行します。暫くすると"変更を保存しますか？" のメッセージが表示されたら、"キャンセル" ボタンをクリックし、そのままにしておきます。TestComplete がエラー終了するまで、待ちます。



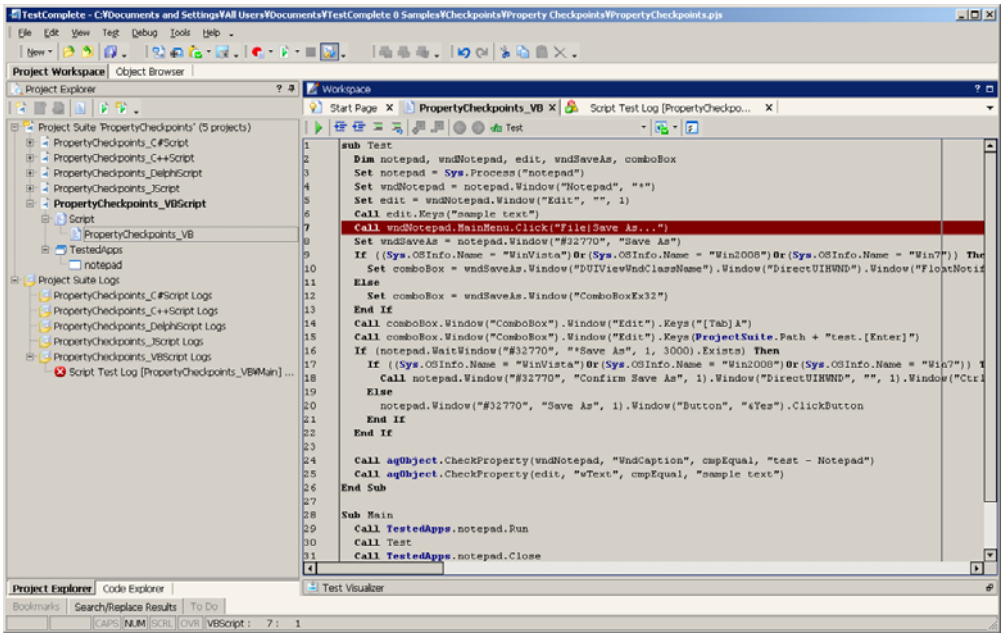
6. エラーログが表示されます。エラーログが右側の Workspace ウィンドウに表示されていない場合は、左側の Project Explorer の **Project Suite Logs** ノードを展開し、PropertyCheckpoints_VBScripts Logs の下にある**赤いX印**が付いた Script Test Log [PropertyCheckpoints_VB¥Main]yyyy/mm/dd hh:mm:ss の行をダブルクリックして実行したばかりの再生エラーログを開きます。



7. エラーログの最初のエラーメッセージ行をクリックし、画面下のペインから [Additional Information]タブを選択します。



8. エラーに関する詳細な情報が表示されます。
   日本語環境で実行しているため、メモ帳のメニュー オブジェクトはすべて日本語です。
9. ステップ 7 で選択した最初のエラーメッセージ行をダブルクリックすると、エラーが発生したスクリプト行がハイライトして表示されます。



10．VBScript では、メニューが英語 GUI のため、再生実行した日本語 GUI と異なるためメニュー項目が見つけられないエラーとなっています。

```
Call wndNotepad.MainMenu.Click("File|Save As...")
```

を以下のように修正します。

```
Call wndNotepad.MainMenu.Click("ファイル(F)|名前を付けて保存(A)...")
```

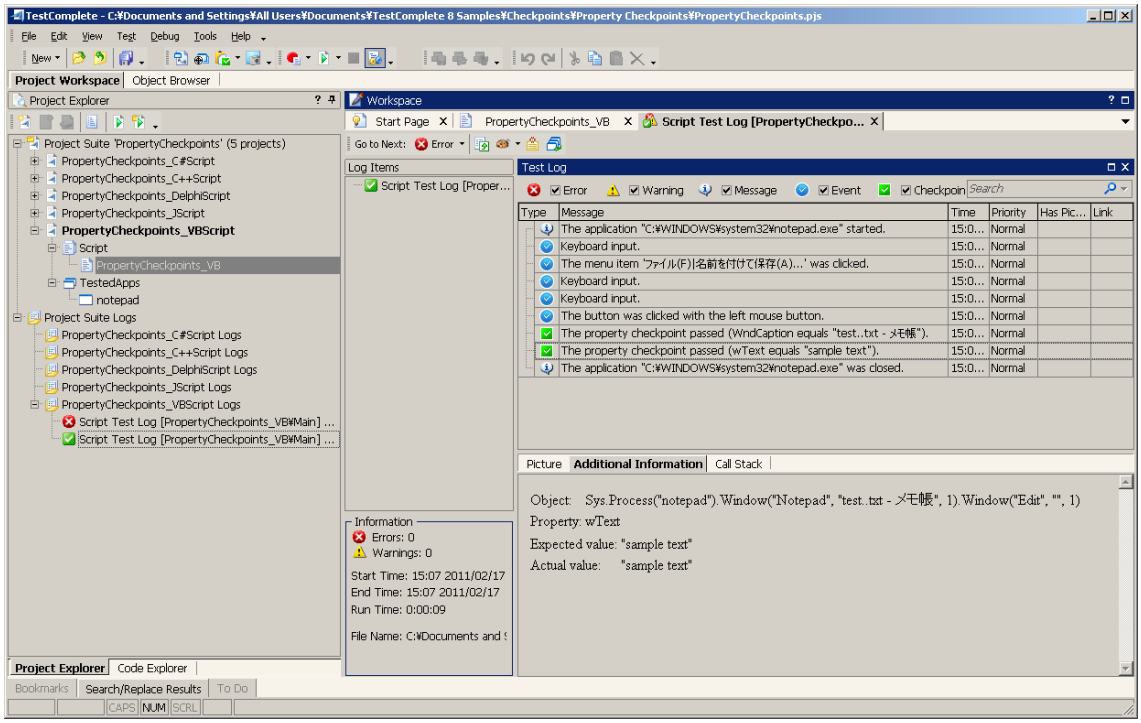
11. 同様にスクリプト内の英語 GUI 項目を日本語 GUI 項目に修正します。
以下に、修正したサンプルを示します。(赤い文字部分)

```
sub Test
  Dim notepad, wndNotepad, edit, wndSaveAs, comboBox
  Set notepad = Sys.Process("notepad")
  Set wndNotepad = notepad.Window("Notepad", "*")
  Set edit = wndNotepad.Window("Edit", "", 1)
  Call edit.Keys("sample text")
  Call wndNotepad.MainMenu.Click("ファイル(F)|名前を付けて保存(A)...")
  Set wndSaveAs = notepad.Window("#32770", "名前を付けて保存")
  If ((Sys.OSInfo.Name = "WinVista")Or(Sys.OSInfo.Name = "Win2008")Or(Sys.OSInfo.Name = "Win7")) Then
    Set                                    comboBox                                    =
wndSaveAs.Window("DUIViewWndClassName").Window("DirectUIHWND").Window("FloatNotifySink", "", 1)
  Else
    Set comboBox = wndSaveAs.Window("ComboBoxEx32")
  End If
  Call comboBox.Window("ComboBox").Window("Edit").Keys("[Tab]A")
  Call comboBox.Window("ComboBox").Window("Edit").Keys(ProjectSuite.Path + "test.[Enter]")
  If (notepad.WaitWindow("#32770", "*名前を付けて保存", 1, 3000).Exists) Then
    If ((Sys.OSInfo.Name = "WinVista")Or(Sys.OSInfo.Name = "Win2008")Or(Sys.OSInfo.Name = "Win7")) Then
      Call  notepad.Window("#32770",  "名前を付けて保存の確認",  1).Window("DirectUIHWND",  "",
1).Window("CtrlNotifySink", "", 7).Window("Button", "はい(&Y)").ClickButton
```

```
    Else
        notepad.Window("#32770", "名前を付けて保存", 1).Window("Button", "はい(&Y)").ClickButton
    End If
  End If

  Call aqObject.CheckProperty(wndNotepad, "WndCaption", cmpEqual, "test..txt - メモ帳")
  Call aqObject.CheckProperty(edit, "wText", cmpEqual, "sample text")
End Sub
```

12. 修正後に再度、テストを実行すると、エラーなしでテストが終了します。



予期しないウィンドウの対応

スクリプトを記録するときに、テストにおけるアプリケーションの予期した振る舞いを定義します。実行再生時に、OS からのアサーションやエラーのメッセージを表示するモーダルウィンドウが表示される、あるいは他のアプリケーションのウィンドウがマウス クリックのシミュレート位置に重なるなどの予期しない状況が発生することがあります。

TestComplete は、スクリプトがアクション(クリック、ダブルクリック、アクティベート)を実行しているときに、予期しないウィンドウのためにチェックのみします。このチェックは、スクリプトがウィンドウ プロパティの特定の操作を実行しているとき、あるいは TestComplete の WaitWindow などの関数を実行しているときにはしません。

スクリプトの実行を妨げるウィンドウはモーダルまたは非モーダルです。オペレーティングシステムは非モーダル ウィンドウに入力フォーカスを持たせたり、他のウィンドウで作業する前に閉じられることを要求しないので、非モーダル ウィンドウは問題にはなりません。実行再生中にこれらのウィンドウをバイパスするには、プロジェクトのプロパティで、On overlapping window | Ignore overlapping window オプションをオンにしてください。TestComplete は、非モーダル オーバーラップ ウィンドウ用に特定のアクションを実行することができる OnOverlappingWindow イベントを生成します。

ウィンドウがモーダルの場合、オペレーティング システムがこのウィンドウが閉じられるまで他のウィンドウに切り替えができません。TestComplete は、プリセットのシーケンスでこれらのウィンドウを自動的に処理します。このシーケンスで重要なステップは、"スクリプトは、今何をするか?"で、プロジェクトプロパティの Playback セクションで定義されます。シーケンスは次のようになります:

1. スクリプト実行は、プロジェクトの Playback(再生実行) オプションで設定される Auto-wait timeout インターバルの間隔が満了するまで遅れます。
2. 予期しないウィンドウがまだ開いている場合、TestComplete は、OnUnexpectedWindow イベントを生成します。
3. ウィンドウが OnUnexpectedWindow イベント ハンドラで閉じられない場合、TestComplete は、予期しないウィンドウの画像といっしょにエラー メッセージをログに出力し、Playback 設定に従って動作します:
4. Stop on error (エラーで停止) がチェックされている場合、 その時点で実行が終了します。
5. Unexpected Window (予期しないウィンドウ) で Stop execution (実行停止) がチェックされている場合も同様。 (違いは、Stop on error は任意のエラーで停止し、Stop execution は予期しないウィンドウで停止します。)

6. それ以外は、予期しないウィンドウの設定により、次の 4 つのアクションのいずれかになります:
    - フォーカスされたボタンをクリックし、ウィンドウのデフォルト ボタンのクリックをシミュレート。
    - Press ESC が ESC キー押下をシミュレートする。
    - Press Enter が ENTER キー押下をシミュレートする。
    - Send WM_CLOSE がウィンドウに通常のウィンドウを閉じるメッセージを送信する。
7. ウィンドウがまだ閉じられない場合(閉じるアクションをチェックしていない、または閉じるアクションが失敗した場合)、実行は終了します。

予期しないウィンドウが表示されるとき、TestComplete は、その画像を保存し、エラーメッセージをログに出力します。画像として画面全体を保存やキーワード テストを起動したりする特定のアクションを実行する必要がある場合、OnUnexpectedWindow イベントのイベント ハンドラを使用できます。詳細については、Handling Events(イベントのハンドリング) と Creating an Event Handler for the OnUnexpectedWindow Event (OnUnexpectedWindow イベントのイベント ハンドラの作成)を参照してください。OnUnexpectedWindow または OnOverlappingWindow イベントハンドラ内でマウスクリックをシミュレートする場合、TestComplete は再帰的に予期しないウィンドウをチェックすることに注意してください。

CLX ユーザーへの注意: 現在、Delphi と C++Builder CLX アプリケーションで予期しないウィンドウをハンドルすることはできません。これは、CLX ライブラリ実装の仕様のためです:アプリケーションのウィンドウはモーダル ダイアログが表示されていてもアクティベートできます。この振る舞いにより、TestComplete は予期しないウィンドウとしてモーダル CLX ウィンドウを扱いません。

CLX アプリケーションで予期しないウィンドウをハンドルするには、アプリケーションが実行したアクションのあとでモーダルウィンドウが表示されたかチェックするため、WaitWindow または FindChild メソッドを使用する必要があります。次のようなコード スニペットが例となります:

例:

VBScript
```
Sub Test
  Dim p, unexpWnd
  Set p = Sys.Process("MyApplication")
  ' Do something
  ...
  ' Handle possible unexpected window
  Set unexpWnd = p.WaitWindow("QWidget", "Error", -1, 1000)
  If unexpWnd.Exists Then
    ' Post the window image to the log
```

```
    Call Log.Picture(unexpWnd.Picture, "Unexpected window detected.")
    ' Close unexpected window
    unexpWnd.Close
  End If
  ' Continue testing
  ...
End Sub
```

JScript

```
function Test()
{
  var p, unexpWnd;
  p = Sys.Process("MyApplication");
  // Do something
  ...
  // Handle possible unexpected window
  unexpWnd = p.WaitWindow("QWidget", "Error", -1, 1000);
  if (unexpWnd.Exists)
  {
    // Post the window image to the log
    Log.Picture(unexpWnd.Picture(), "Unexpected window detected.");
    // Close unexpected window
    unexpWnd.Close();
  }
  // Continue testing
  ...
}
```

DelphiScript

```
procedure Test;
var p, unexpWnd;
begin
  p := Sys.Process('MyApplication');
  // Do something
  ...
  // Handle possible unexpected window
  unexpWnd := p.WaitWindow('QWidget', 'Error', -1, 1000);
  if unexpWnd.Exists then
  begin
    // Post the window image to the log
    Log.Picture(unexpWnd.Picture, 'Unexpected window detected.');
    // Close unexpected window
    unexpWnd.Close;
  end;
  // Continue testing
  ...
end;
```

C++Script, C#Script

```
function Test()
{
  var p, unexpWnd;
  p = Sys["Process"]("MyApplication");
  // Do something
  ...
```

```
  // Handle possible unexpected window
  unexpWnd = p["WaitWindow"]("QWidget", "Error", -1, 1000);
  if (unexpWnd["Exists"])
  {
    // Post the window image to the log
    Log["Picture"](unexpWnd["Picture"](), "Unexpected window detected.");
    // Close unexpected window
    unexpWnd["Close"]();
  }
  // Continue testing
  ...
}
```

'Object Not Found' (オブジェクトが見つかりません) エラーの対応

このセクションは、テスト エンジンがテスト実行中にテストするプロセス、ウィンドウ、またはコントロールを見つけられない一般的なエラーの対応方法について説明します。これらのエラーは、テスト実行中と作成中のアプリケーションの状態が異なるなど、テスト時にアプリケーションで何かが変更されたことを示しています。エラーをどのように対処するかの説明に従ってください:

- オブジェクトが存在しない
- オブジェクトのオブジェクト名を見つけられない
- *class name* のウィンドウ クラス、*window caption* と index *n* のウィンドウ キャプションを持つウィンドウを取得できない
- 'Ambiguous Recognition of the Tested Object(テストするオブジェクトのあいまいな認識)'メッセージについて
- 'Mapped item has the Extended Find attribute enabled (マップされた項目の拡張検索の属性が有効になっている)' メッセージについて

'Object Does Not Exist (オブジェクトが存在しない)' エラーの対処

テストエンジンがテスト実行中にテスト対象のプロセス、ウィンドウ、コントロールが見つけられないときに発生する “Object does not exist(オブジェクトが存在しない)”問題を診断し、解決する方法を説明します。

エラー メッセージについて

TestComplete は、プロセス、ウィンドウ、コントロールをオブジェクトとしてみなします。それらにテストコマンドを実行するとき、最初にシステム内にオブジェクトが存在するかチェックしてから、必要なこのオブジェクトのメソッドまたはプロパティを実行します。テスト エンジンがオブジェクトの検出に失敗した場合、テストログにエラー メッセージを出力します。このメッセージは、テストされるアプリケーションで何かが変更されたことを示します。テスト記録中または作成中の状態から変化しています。この問題の原因を見つけて、取り除く方法を見てみましょう。

問題の診断

1. テストログを見て、最初のエラーを見つけます。実行されたコマンドに依存しますが、TestComplete はログに複数のエラーメッセージを出力することがあります。これは、テストとスクリプト エンジン機能の特定の仕様のためです(スクリプト エンジンはスクリプトでないキーワードテストでも使用されます)。一般的に、最初のエラー メッセージは、問題のコマンドに対応していて、更なるエラーメッセージは、最初のエラーにより引き起こされます。これらの操作手順は、最初のエラーメッセージが “The object does not exist”(オブジェクトが存在しない)ことを仮定にしています。
2. ログ内のエラーメッセージを選択し、ログの **Additional Information** (補足情報) を見ます。このパネルにはエラーの説明が含まれています。**Tested Object** セクションは、呼ばれたメソッドまたはプロパティを所有するオブジェクト名を表示します。**Missing Object** セクションは、見つからなかったオブジェクトの名前を表示します。言い換えると、このセクションが問題のオブジェクトの名前を表示します。
3. エラーが発生したときに実行されたテストコマンドが何か知るには、テストログのエラーメッセージをダブルクリックします。TestComplete はエラーが発生したときに実行した行を自動的にハイライト表示して、テストを編集用に開きます。

アプリケーションを見て、目的のオブジェクトが存在するか確認します。たとえば、アプリケーションが変更され、ボタンなどがない場合は、再度レコーディングします。

オブジェクトが存在する場合、エラーの原因を見つける必要があります。オブジェクトのプロパティを調べてください。詳細は以下のステップに従ってください。

4. オブジェクトを調べるためには、テストするアプリケーションを実行している必要があります。アプリケーションが終了している場合、再度起動し、問題のテスト行で一時停止します。これをするには:
   - ブレークポイント をエラーが発生したときに実行していた行に設定します。

     編集するためにテストをまだ開いていない場合、テストログのエラー行をダブルクリックします。TestComplete が、テストを開いてエラーが発生したときに実行していた行をハイライト表示します。

     ブレークポイントを設定するには、Editor 画面のその行の左端の空間をクリックするか、 F9 を押します。ブレークポイントが設定されると、その行が赤くハイライト表示します。

   - テストを実行します。ブレークポイントに達すると自動的にテストエンジンがテストの実行を一時停止します。

ブレークポイントとデバッグに関する詳細は、Debugging Tests を参照してください。

これで、テストを解析することができます:

- **デスクトップ アプリケーションの場合**
  - TestComplete のツールバーから  "Display Object Spy" を選択します。[Object Spy] ウィンドウが表示されます。

    [Object Spy] ウィンドウを使って、画面上にある目的のオブジェクトを選択します。これをするには、**Finder ツール**  をオブジェクトにドラッグします。ドラッグしているとき、TestComplete はマウスカーソルの下にあるオブジェクトを赤い枠でハイライトします。カーソルが目的のウィンドウまたはコントロールの上にあり、赤い枠でハイライトされたときにマウス ボタンを放します。[Object Spy] ウィンドウがオブジェクトのプロパティを表示します。
    ヒントやメニュー項目のようなポップアップ オブジェクトを選択する必要があ

る場合、もうひとつの選択モードを使用します。[Object Spy] ウィンドウの [**Select object with cursor**] をクリックし、画面上に必要なオブジェクトが表示されるように必要な操作を実行します。オブジェクトが表示されたら、マウス オーバーして SHIFT+CTRL+A を押してオブジェクトを指定します (このショートカットキーの組み合わせは、TestComplete のオプションで変更できます)。[Object Spy] がオブジェクトのプロパティを表示します。

- [Object Spy] ウィンドウの **Highlight Object in Object Tree** アイコンをクリックします。これは、TestComplete の Object Browser パネルでオブジェクトを選択します。

- **PDA アプリケーションの場合**
  - TestComplete のツールバーから **Display Object Spy** を選択します。Object Spy ウィンドウが表示されます。
  - 画面上の目的のオブジェクトを選択するために Object Spy ウィンドウを使用します。これをするには、**Select object with cursor** を選択し、PDA Controller パネルをアクティベートします。その後で、カーソルをパネル上の目的のオブジェクトに移動して、 SHIFT+CTRL+A を押して、オブジェクトを指定します (このショートカットは、TestComplete のオプションで変更できます)。Object Spy ウィンドウがオブジェクトのプロパティを表示します。
  - **Highlight Object in Object Tree** アイコンをクリックします。これは、TestComplete の Object Browser パネルでオブジェクトを選択します。

可能性のある原因

以下に一般的な問題の原因とその対応方法を示します。

テストするプロセスが見つからない

テストしているプロセスがシステム上で実行しているかチェックします。プロセスを識別するために、TestComplete は、それらの名前とインデックスを使用します。テストを作成または記録したときに使用されたプロセス インデックスとテストしているプロセスのインデックスが一致するか照合してください。

オブジェクトの確認用プロパティが変更されている

ウィンドウとコントロールを認識するには、TestComplete は、その兄弟の中のオブジェクトを識別するためにいくつかのプロパティを使用します。

最も可能性の高いエラーの原因は、テストの記録中または作成中のプロパティの値が認識するプロパティの値と異なることです。

認識プロパティの値を判断するには、Object Browser パネルでそのオブジェクトの MappedName プロパティを選択し、 ボタンをクリックします。マッピング設定を表示する Name Mapping Item Information ダイアログが表示されます。

認識プロパティを調べて、期待する値と一致するか見てください。プロパティがオブジェクトを一意に識別していて、あいまいな識別の原因になっていないかどうかをチェックしてください。

プロパティが期待している値と一致しない場合、またはオブジェクトの一意の識別がされていない場合、マッピング設定条件を変更する必要があります。これをするには:

- Project Explorer パネルに切り替えます (これは、TestComplete のメインメニューから [View] - [Project Explorer] を選択します)。
- Project Explorer で、NameMapping 項目をダブルクリックします。Name Mapping Editor が呼び出されます。
- エディタで、Mapped Objects セクションを展開し (デフォルトでは折りたたまれています)、目的のオブジェクトを見つけます。
- Mapped Object ツリーでそのオブジェクトを選択します。このオブジェクトで使用されるマッピング設定のリストは、右側に表示されます。このリストで直接、設定を編集できます。マッピング設定でさらにプロパティを追加する必要がある場合、オブジェクトを再度マッピングしなければなりません。詳細は、Mapping Application Objects を参照してください。

  オブジェクトをマッピングする前に、既存のマッピング項目とエイリアスを削除する必要があります。テスト動作を維持するために、新しくマップされたオブジェクトに同じエイリアスを割り当てることができます。詳細は、Name Mapping - Aliases を参照してください。

ヒント:

- 認識条件が Index プロパティを含んでいて、そのプロパティの値が変更された場合、テストするオブジェクトの一意の識別にならないのでマッピングにこのプロパティを使用しないでください。見つからないオブジェクトの他のプロパティを調べてオブジェクトの一意の識別で利用できる他のプロパティを見つけてください。

  識別のためにオブジェクトのプロパティを使用するほかに、その子オブジェクトのプロパティを

使用することもできることに注意してください。たとえば、そのボタン、テキストボックスでフォームをマップすることができます。

- マッピング設定は、テストするウィンドウやコントロールのキャプション(またはテキスト)を含めることができます。ワイルドカード (* と ?) をキャプションの変化する部分に指定できます。"?"は、任意の 1 文字、"*" は任意の文字列を表します。 たとえば、ウィンドウのキャプションが *MyFile - Notepad* のような場合、* - *Notepad* のように * を名前と置き換えることができます。

  これらのワイルドカードを WndClass または wText のような任意のプロパティの文字列に指定できます。

  オブジェクトの識別に Name および FullName プロパティを使用しないでください。

- Open アプリケーションのオブジェクトをマップするために、TestComplete は edit11、button1 などのアプリケーション コードで定義されたオブジェクト名を使用します。Open アプリケーションは、その内部のオブジェクト、メソッド、プロパティにテストエンジンがアクセスできるアプリケーションです。 TestComplete は、テスト オブジェクトの一意の識別属性として、アプリケーションで定義した名前を使用します。テストするオブジェクトのアプリケーション定義の名前が変更された場合、マッピング設定を変更し、新しいアプリケーション定義の名前を使用する必要があります。

オブジェクト ツリー内のオブジェクトの位置が変更されている

識別設定が正しい場合、オブジェクトがテストの作成または記録中にあったオブジェクトツリーとは別のレベルにあるかもしれません。これは、開発者がテストするアプリケーションを変更したり、フォームまたはコントロールを追加、削除した場合に起こります。この状態の典型的な例は、フォームの間にパネルを追加し、ボタンが見つからない、あるいはフォームとテストするコントロールの間のパネルを削除したときです。

オブジェクト階層が変更された場合、一般的な解決策は、新しい階層に一致するようにテストを変更することです。しかし、これは時間を要する作業です。オブジェクトがエイリアスで特定できる場合、簡単に名前マッピングツリーを変更することですばやく問題を解決できます:

- Project Explorer パネルに切り替えます (TestComplete のメインメニューで[View] - [Project Explorer] を選択します)。
- Project Explorer で、NameMapping 項目をダブルクリックします。Name Mapping Editor が呼び出されます。

- エディタで、Mapped Objects セクションを展開し (デフォルトでは、折りたたまれています)、目的のオブジェクトを見つけます。
- Mapped Objects ツリーで、オブジェクトを必要なレベルにドラッグします。

オブジェクトの階層が 1 つのアプリケーション実行から別のアプリケーションに変更する場合、このレベルからさらに深くオブジェクトを探すため、テストエンジンを実行する一番上位のレベルにオブジェクトを移動することを推奨します:

- オブジェクトを最上位のレベルにドラッグします。
- Name Mapping エディタの Mapped Objects セクション内の任意の場所を右クリックし、コンテキストメニューから **Field Chooser** を選択します。利用可能な見出しカラムのリストが表示されます。
- リストから **Extended Find** 見出しを Mapped Objects テーブルの見出し行にドラッグします。
- 見出しリストを閉じます。
- オブジェクトの **Extended Find** チェックボックスを選択します。これは、テストエンジンがこのレベルからさらに深くオブジェクトを検索するコマンドを実行します。

オブジェクト名のマッピングに関する詳細については、Name Mapping を参照してください。

オブジェクトがまだ作成されていない

識別設定が正しく、オブジェクトがオブジェクト階層の期待しているレベルにある場合、テストエンジンがそれを探すときまでにオブジェクトが作成されていない可能性があります。たとえば、テストするダイアログボックスが長い時間かけて開くとき、テストエンジンは、ダイアログは存在しないと"決めます"。この問題を解決するには、テストを修正してオブジェクトを "待つ"必要があります。この方法はテストの種類に依存します。詳細は、Waiting for Process or Window Activation(プロセスまたはウィンドウのアクティベートを待つ)を参照してください。

オブジェクトの親オブジェクトの 1 つがあいまいで特定できない

すべての設定が正しく、TestComplete がオブジェクトを探すときに存在している場合、テストするアプリケーションであいまいなオブジェクト認識がある可能性があります。たとえば、オブジェクトの親オブジェクトのひとつのマッピングがあいまいで、TestComplete が探しているオブジェクトを識別できないことがあります。この問題を解決するには、対応するマップされたオブジェクトであいまいな認識を除去

することを推奨します。詳細については、About the 'Ambiguous Recognition of the Tested Object' Message (テストオブジェクトのメッセージのあいまいな認識)を参照してください。

テストしているアプリケーションが応答しない

テストするアプリケーションがユーザーアクションやオペレーティングシステムのメッセージに応答しない場合、TestComplete は目的のウィンドウまたはコントロールを見つけることができません。アプリケーションがフリーズしたかどうかオペレーティング システムのタスク マネージャでチェックできます。アプリケーションの**状態**が実行中か*応答なし*か表示されます。

テスト プロジェクトの Freeze Diagnostics プロパティに応じて、テスト エンジンはアプリケーションが応答を停止した場合には自動的にアプリケーションを終了させます。テスト エンジンがアプリケーションを終了させる場合、エラーメッセージ *"The process does not respond. It will be terminated."*がテストログに出力されます。Freeze Diagnostics プロパティの表示と変更に関する詳細は、Diagnosing Application Freezes を参照してください。

フリーズしたアプリケーションが AQtrace Reporter を含んでいる場合、TestComplete は、問題の原因をより早く見つけるための詳細なエラーレポートを自動的に生成することに注意してください。詳細は、Diagnosing Application Freezes を参照してください。

テストを作成したときとオブジェクトのツリーモデルが変わっている

- **デスクトップ アプリケーションの場合**

  TestComplete は、*Tree(ツリー)* と *Flat(フラット)* の 2 つのオブジェクト ツリー モデルをサポートします。ひとつのモデル用に作成されたテストは他方のモデルとは互換性がありません。Flat モデルは、TestComplete 3 およびそれ以前で使用されていました。TestComplete 4 からは、デフォルトが Tree モデルになっています。古いバージョンで作成したテストを実行する場合は、アクティブなモデルを確認する必要があります。これを確認するには:

  - Project Explorer パネルに切り替えます (メイン メニューで View | Project Explorer を選択します)。
  - Project Explorer パネルでプロジェクトを右クリックします。コンテキスト メニューから Edit | Properties を選択します。project editor が開き、プロパティ ページがアクティブになります。

- ページの左にあるプロパティ グループのツリーから General を選択します。右の Object ツリー モデル設定の値をチェックします。

- **Web ページの場合**

Web ページのテストをしている場合、Web オブジェクト用に TestComplete が使用する tree モデルになっていることを確認する必要があります。(Web Tree Models を参照してください)。これを確認するには:

- Project Explorer に切り替えますメインメニューで View | Project Explorer を選択します)。
- Project Explorer パネルでプロジェクトを右クリックします。コンテキスト メニューから Edit | Properties を選択し、ます。project editor が開き、プロパティ ページがアクティブになります。
- ページの左にあるプロパティ グループから Open Applications | Web Testing を選択します。テストを作成したときに使用されたモデルと同じモデルが設定されていなければなりません。

'Unable to Find the Object'(オブジェクトを見つけられない) エラーの対処

このトピックは、"オブジェクトを見つけられない" エラーの診断方法とその解決法を説明します。

エラーメッセージについて

TestComplete は、プロセス、ウィンドウ、コントロールをオブジェクトとして考えます。それらの上でテストコマンドを実行するとき、システム上にオブジェクトが存在するかどうかを最初にチェックし、次にオブジェクトの必要なメソッドまたはプロパティを実行します。テストに指定されたオブジェクトの参照を取得することに失敗したとき、テスト エンジンは、“Unable to find the object...”メッセージをテストログに出力します。以下にこのエラーを引き起こす典型的なスクリプト コードの例を記述します(*Button1* は存在しないオブジェクトの名前です) :

サンプル

VBScript

```
Set someVariable = form.Button1
form.Button1.ClickButton
```

JScript

```
var someVariable = form.Button1;
form.Button1.ClickButton();
```

DelphiScript

```
var
  someVariable : OleVariant;
begin
  someVariable := form.Button1;
```

```
form.Button1.ClickButton();
end;
```

C++Script, C#Script

```
var someVariable = form["Button1"];
form["Button1"]["ClickButton"]();
```

メッセージは、テストするアプリケーションで変更された項目が示され、その状態がテストを記録したときまたは作成したときの状態と異なっています。その問題の原因を見つけ出し、それを取り除く方法をみてみましょう。

問題の診断

1. テスト ログの最初のエラー メッセージを見つけてください。実行されたコマンドに依存しますが、TestComplete は複数のエラーメッセージをログに出力することがあります。これは、特定のテスト仕様、スクリプトエンジンの機能のためです(スクリプト エンジンは、スクリプトではなく、キーワードテストを実行しているときでも使用されます)。一般的には、最初のエラー メッセージは、問題のあるコマンドに関連していて、続くエラーは最初のエラーによって引き起こされます。ここでは、最初のエラー メッセージが“Unable to find the object *object_name*”(オブジェクト *object_name* が見つかりません)であること仮定にしています。
2. エラーが発生したときに、どのコマンドを実行していたか理解するために、テスト ログでそのエラー メッセージをダブルクリックします。TestComplete がテストを開いて、エラーが発生したときに実行していた行を自動的にハイライト表示します。

   テストを調査することで、見つからないオブジェクトのフルネームを決定できます。

3. テストしているアプリケーションで目的のオブジェクトが存在するかチェックします。たとえば、開発者がテストするアプリケーションを変更して、テストするウィンドウまたはコントロールを削除していることがあります。この変更は、テストでエラーを引き起こします。
4. アプリケーションにテストするオブジェクトが存在する場合、そのプロパティを調べ、問題の原因を見つけて修正します。可能性のある原因と解決策に関する情報は、以下を参照してください。

オブジェクトを調べるためには、テストするアプリケーションが実行されている必要があります。テストが終わった後でアプリケーションが終了した場合、再度実行し、問題のあるテスト行で実行を一時停止します。これをするには：

- エラーが発生したとき実行していた行にブレークポイントを設定します。

  テストをまだ開いていない場合、テストログのエラーメッセージをダブルクリックします。TestComplete はエラーが発生したときに実行していた行をハイライト表示します。

  ブレークポイントを設定するには、エディタでその行の横の左端の空間をクリックするだけです。または F9 を押します。ブレークポイントが設定されると、その行は赤色でハイライトされます。

- テストを実行します。テストエンジンが自動的にブレークポイントに到達したときにテストの実行を一時停止します。

ブレークポイントとデバッグに関する情報は、Debugging Tests を参照してください。

これで、テスト中のアプリケーションを調査することができます：

- **For desktop applications**
  - TestComplete で、 Tools ツールバーから**[Display Object Spy]** アイコンを選択します。
  - 画面上で目的のオブジェクトを選択するために [Object Spy]ウィンドウを使用します。これをするには、[Finder ツール ( )]をオブジェクトにドラッグします。ドラッグしている間、TestComplete は、マウス カーソルの下にあるオブジェクトを赤い枠でハイライトします。マウスが目的のオブジェクトの上にあり、TestComplete が赤い枠でそのウィンドウまたはコントロールをハイライトしたときにマウス ボタンを放します。[Object Spy] ウィンドウがオブジェクトのプロパティを表示します。

    ヒントやメニューなどのポップアップ オブジェクトを選択する必要がある場合、[Object Spy] ウィンドウの別の選択モードである**[Select object with cursor]** を使用し、画面に目的のオブジェクトが表示されるアクションを実行します。オブジェクトが表示されたときに、マウスをその上に持って行き、SHIFT+CTRL+A を押して、オブジェクトを指定します(TestComplete のオ

プションでこのショートカットキーは変更できます)。[Object Spy] ウィンドウにオブジェクトのプロパティが表示されます。

- **Highlight Object in Object Tree** アイコンをクリックします。これで、TestComplete の Object Browser パネルでオブジェクトが選択されます。

- **PDA アプリケーションの場合**
  - TestComplete で、Tools ツールバーから[Display Object Spy] アイコンを選択します。
  - 画面上の目的のオブジェクトを選択するために [Object Spy] ウィンドウを使用します。これをするには、[Select object with cursor] を選択し、PDA Controller パネルをアクティベートします。その後で、このパネルで必要なオブジェクトにカーソルを移動して、SHIFT+CTRL+A を押して、オブジェクトを指定します (このショートカットは、TestComplete のオプションで変更できます)。Object Spy ウィンドウがオブジェクトのプロパティを表示します。
  - **Highlight Object in Object Tree** アイコンをクリックします。 これは、TestComplete の Object Browser パネルを選択します。

可能性のある原因

以下に一般的な問題の原因とその対応方法を示します。

オブジェクト名が誤っている

指定しているオブジェクト名が正しいか確認します。スクリプトコードを使用している場合、JScript、C++Script、および C#Script は大文字/小文字を区別することに注意して、名前の中の大文字をチェックしてください。

オブジェクト名が変更された

Open アプリケーションのオブジェクトを参照するには、TestComplete はアプリケーション コード内で定義された button1、edit1、MainForm などのオブジェクト名を使用します。開発者がオブジェクト名を変更した場合、TestComplete は古い名前でオブジェクトを見つけることができません。この問題を

解決するには、テストを再度記録しなおすか、テストコマンドを更新して新しいオブジェクト名を使うようにします。

オブジェクトがマップされていない場合、それをマップしアクセスするためのエイリアスを使用することを推奨します。Name mapping (名前マッピング) は、TestComplete がアプリケーションで定義された名前によるオブジェクトの識別を可能にするだけでなく、プロパティ値によっても識別を可能にします。この識別アプローチは、テストするアプリケーションで発生する変更に対して、より耐性が増します。マッピングに関する詳細は、Name Mapping を参照してください。

オブジェクトツリー内のオブジェクトの位置が変更されている

このエラーの可能性のある原因の1つは、テストするオブジェクトが、テストを記録または作成中と異なるオブジェクト ツリーの別のレベルにある場合です。これは、開発者がテストするアプリケーションを変更してフォームまたはコントロールを追加したり削除した場合に起こります。この状態の典型的な例は、開発者がフォームと見つからないボタンの間にパネルを追加したり、フォームとテストするコントロールの間の重要なパネルを削除したときです。

オブジェクト階層が変更された場合、一般的な解決策は、テストを変更するか、新しい階層に一致するようにテストすることです。

もうひとつの方法は、オブジェクト プロパティによって目的のテスト オブジェクトを検索する特定のスクリプト関数を使用することです。検索が成功する場合、関数は見つけたオブジェクトを返します。テストの中にこれらの関数のコールを挿入でき、これらの関数から返されたテスト オブジェクトで作業ができます。詳細については、Searching for Objects in Tests を参照してください。

オブジェクトが TestComplete がそれを取得するときまでに作成されていない

識別設定が正しく、オブジェクトがオブジェクト階層の期待しているレベルにある場合、テストエンジンがそれを探すときまでにオブジェクトが作成されていない可能性があります。たとえば、テストするダイアログボックスが長い時間かけて開くとき、テストエンジンは、ダイアログは存在しないと "決めます"。この問題を解決するには、テストを修正してオブジェクトを "待つ" 必要があります。この方法はテストの種類に依存します。詳細は、Waiting for Process or Window Activation を参照してください。

テストしているアプリケーションが応答しない

テストするアプリケーションがユーザーアクションやオペレーティングシステムのメッセージに応答しない場合、TestComplete は目的のウィンドウまたはコントロールを見つけることができません。アプリケーションがフリーズしたかどうかオペレーティング システムのタスク マネージャでチェックできます。アプリケーションの**状態**が実行中か**応答なし**か表示されます。

テスト プロジェクトの Freeze Diagnostics プロパティに応じて、テスト エンジンはアプリケーションが応答を停止した場合には自動的にアプリケーションを終了させます。テスト エンジンがアプリケーションを終了させる場合、エラーメッセージ “The process does not respond. It will be terminated.”がテストログに出力されます。Freeze Diagnostics プロパティの表示と変更に関する詳細は、Diagnosing Application Freezes を参照してください。

フリーズしたアプリケーションが AQtrace Reporter を含んでいる場合、TestComplete は、問題の原因をより早く見つけるための詳細なエラーレポートを自動的に生成することに注意してください。詳細は、Diagnosing Application Freezes を参照してください。

テストを作成したときとオブジェクトのツリー モデルが異なっている

- **デスクトップ アプリケーションの場合**

  TestComplete は、*Tree(ツリー)* と *Flat(フラット)* の 2 つのオブジェクト ツリー モデルをサポートします。ひとつのモデル用に作成されたテストは他方のモデルとは互換性がありません。Flat モデルは、TestComplete 3 およびそれ以前で使用されていました。TestComplete 4 からは、デフォルトが Tree モデルになっています。古いバージョンで作成したテストを実行する場合は、アクティブなモデルを確認する必要があります。これを確認するには:

  - Project Explorer パネルに切り替えます (メイン メニューで View | Project Explorer を選択します)。
  - Project Explorer パネルでプロジェクトを右クリックします。コンテキスト メニューから Edit | Properties を選択します。project editor が開き、プロパティ ページがアクティブになります。
  - ページの左にあるプロパティ グループのツリーから General を選択します。右の Object ツリー モデル設定の値をチェックします。
- **Web ページの場合**

Web ページのテストをしている場合、Web オブジェクト用に TestComplete が使用する tree モデルになっていることを確認する必要があります。(Web Tree Models を参照してください)。これを確認するには:

- Project Explorer に切り替えますメインメニューで View | Project Explorer を選択します)。
- Project Explorer パネルでプロジェクトを右クリックします。コンテキスト メニューから Edit | Properties を選択し、ます。project editor が開き、プロパティ ページがアクティブになります。
- ページの左にあるプロパティ グループから Open Applications | Web Testing を選択します。テストを作成したときに使用されたモデルと同じモデルが設定されていなければなりません。

'Cannot Obtain the Window...' (ウィンドウを取得できない) エラーの対処

ここでは、テスト実行中に発生する “Cannot obtain the window with the window class *class_name*, window caption *window_caption* and index *n*”(ウィンドウ クラス *class_name*、ウィンドウ キャプション *window_caption*、および idex *n* を持つウィンドウを取得できません) エラーの診断方法とその解決方法について説明します。

エラー メッセージについて

TestComplete は、プロセス、ウィンドウ、コントロールをオブジェクトとしてみなします。それらにテストコマンドを実行するとき、最初にシステム内にオブジェクトが存在するかチェックしてから、必要なこのオブジェクトのメソッドまたはプロパティを実行します。テスト エンジンがオブジェクトの検出に失敗した場合、テストログにエラー メッセージを出力します。このメッセージは、テストされるアプリケーションで何かが変更されたことを示します。テスト記録中または作成中の状態から変化しています。この問題の原因を見つけて、取り除く方法を見てみましょう。

問題の診断

1. テストログを見て、最初のエラーを見つけます。実行されたコマンドに依存しますが、TestComplete はログに複数のエラーメッセージを出力することがあります。これは、テストとスクリプト エンジン機能の特定の仕様のためです(スクリプト エンジンはスクリプトでないキーワードテストでも使用されます)。一般的に、最初のエラー メッセージは、問題のコマンドに対応していて、更なるエラーメッセージは、最初のエラーにより引き起こされます。これらの操作手順は、最初のエラーメッセージが “Cannot obtain the window...”(ウィンドウを取得できない) ことを仮定にしています。
2. エラーが発生したときに実行されたテストコマンドが何か知るには、テストログのエラーメッセージをダブルクリックします。TestComplete はエラーが発生したときに実行した行を自動的にハイライト表示して、テストを編集用に開きます。
3. アプリケーションを見て、目的のオブジェクトが存在するかチェックします。たとえば、アプリケーションが変更され、テストするウィンドウやコントロールが削除されていることがあります。この変更は、エラーの原因になります。

4. オブジェクトが存在する場合、エラーの原因を見つける必要があります。オブジェクトのプロパティを調べてください。詳細は以下のステップに従ってください。

   オブジェクトを調べるためには、テストするアプリケーションを実行している必要があります。アプリケーションが終了している場合、再度起動し、問題のテスト行で一時停止します。これをするには:

   - ブレークポイント をエラーが発生したときに実行していた行に設定します。
     編集するためにテストをまだ開いていない場合、テストログのエラー行をダブルクリックします。TestComplete が、テストを開いてエラーが発生したときに実行していた行をハイライト表示します。
     ブレークポイントを設定するには、Editor 画面のその行の左端の空間をクリックするか、 F9 を押します。ブレークポイントが設定されると、その行が赤くハイライト表示します。
   - テストを実行します。ブレークポイントに達すると自動的にテストエンジンがテストの実行を一時停止します。

   ブレークポイントとデバッグに関する詳細は、Debugging Tests を参照してください。

   これで、テストを解析することができます:

   - **デスクトップ アプリケーションの場合**
     - TestComplete のツールバーから  "Display Object Spy" を選択します。
       [Object Spy] ウィンドウが表示されます。

       [Object Spy] ウィンドウを使って、画面上にある目的のオブジェクトを選択します。これをするには、**Finder ツール**  をオブジェクトにドラッグします。ドラッグしているとき、TestComplete はマウスカーソルの下にあるオブジェクトを赤い枠でハイライトします。カーソルが目的のウィンドウまたはコントロールの上にあり、赤い枠でハイライトされたときにマウス ボタンを放します。
       [Object Spy] ウィンドウがオブジェクトのプロパティを表示します。
       ヒントやメニュー項目のようなポップアップ オブジェクトを選択する必要がある場合、もうひとつの選択モードを使用します。[Object Spy] ウィンドウの [**Select object with cursor**] をクリックし、画面上に必要なオブジェクトが表示されるように必要な操作を実行します。オブジェクトが表示されたら、マウス オーバーして SHIFT+CTRL+A を押してオブジェクトを指定します

(このショートカットキーの組み合わせは、TestComplete のオプションで変更できます)。[Object Spy] がオブジェクトのプロパティを表示します。

- [Object Spy] ウィンドウの **Highlight Object in Object Tree** アイコンをクリックします。これは、TestComplete の Object Browser パネルでオブジェクトを選択します。

- **PDA アプリケーションの場合**

  - TestComplete のツールバーから **Display Object Spy** を選択します。Object Spy ウィンドウが表示されます。
  - 画面上の目的のオブジェクトを選択するために Object Spy ウィンドウを使用します。これをするには、**Select object with cursor** を選択し、PDA Controller パネルをアクティベートします。その後で、カーソルをパネル上の目的のオブジェクトに移動して、 SHIFT+CTRL+A を押して、オブジェクトを指定します (このショートカットは、TestComplete のオプションで変更できます)。Object Spy ウィンドウがオブジェクトのプロパティを表示します。

  - **Highlight Object in Object Tree** アイコンをクリックします。これは、TestComplete の Object Browser パネルでオブジェクトを選択します。

可能性のある原因

以下に一般的な問題の原因とその対応方法を示します。

ウィンドウ クラス名、キャプション、インデックスが誤っている

入力したクラス名、キャプション、インデックスが正しいか確認してください。

ウィンドウ キャプション(テキスト)、クラス名、またはインデックスが変更されている

ウィンドウを認識するには、通常、TestComplete は ウィンドウ クラス名、キャプション、インデックスの 3 つの属性を使用します(Naming Windows を参照)。これらのいずれかが期待している値と一致しない場合、TestComplete はテスト実行中にウィンドウの検出に失敗します。

この問題を解決するには、テストを変更するか、テスト コマンドを更新するかして、ウィンドウ認識のための適切な値を使用します。以下にいくつかのヒントがあります:

- ウィンドウ クラス名またはキャプションがテスト中に変わる、または 1 つのテストから別のテストで変わる場合、変化する部分に ？ と * ワイルドカードを使用します。"?"は、任意の 1 文字、"*" は任意の文字列を表します。たとえば、開発者がクラス名を *TComboBox* から *TComboBoxEx* に変えた場合、両方の値に一致するように *TComboBox** を使用できます。これは、テストするアプリケーションの変更に対して、さらに耐えるテストにします。
- クラス名とキャプションがウィンドウを一意に識別する場合、ウィンドウ名のインデックスを使用しないでください。これは、テストするアプリケーションの変更に対して、さらに耐えるテストにします。
- ウィンドウまたはコントロールをマップし、そのウィンドウまたはコントロールをアクセスするためにエイリアスを使用することを推奨します。Name Mapping は、テストするアプリケーションの変更に対して、さらに耐えるテストにします。詳細は、Name Mapping を参照してください。

オブジェクト ツリー内のオブジェクトの位置が変更されている

可能性のある原因のひとつは、テスト作成中またはテスト記録中と異なり、テストするオブジェクトがオブジェクト ツリーの別のレベルにあることです。これは、開発者がテストするアプリケーションを変更したり、フォームまたはコントロールを追加、削除した場合に起こります。この状態の典型的な例は、フォームの間にパネルを追加し、ボタンが見つからない、あるいはフォームとテストするコントロールの間のパネルを削除したときです。

オブジェクト階層が変更された場合、一般的な解決策は、新しい階層に一致するようにテストを変更することです。

テストするオブジェクトの名前をマップし、テストからオブジェクトを見つけるためにエイリアスを使用することを推奨します。オブジェクトがマップされている場合、TestComplete は、オブジェクト プロパティのセットを使用してそれを探します。また、それはオブジェクト階層の異なるレベル上で実行できます。すべてのこれらの機能は、テストするアプリケーションでの変更に対してさらに信頼性のあるものにします。マッピングの詳細は、Name Mapping を参照してください。

もうひとつの方法は、オブジェクト プロパティによって目的のテスト オブジェクトを検索する特定のスクリプト関数を使用することです。検索が成功する場合、関数は見つけたオブジェクトを返します。テストの中にこれらの関数のコールを挿入でき、これらの関数から返されたテスト オブジェクトで作業ができます。詳細については、Searching for Objects in Tests を参照してください。

TestComplete がオブジェクトを取得するときまでに作成されていない

識別設定が正しく、オブジェクトがオブジェクト階層の期待しているレベルにある場合、テストエンジンがそれを探すときまでにオブジェクトが作成されていない可能性があります。たとえば、テストするダイアログボックスが長い時間かけて開くとき、テストエンジンは、ダイアログは存在しないと"決めます"。この問題を解決するには、テストを修正してオブジェクトを "待つ"必要があります。この方法はテストの種類に依存します。詳細は、Waiting for Process or Window Activation を参照してください。

テストしているアプリケーションが応答しない

テストするアプリケーションがユーザーアクションやオペレーティングシステムのメッセージに応答しない場合、TestComplete は目的のウィンドウまたはコントロールを見つけることができません。アプリケーションがフリーズしたかどうかオペレーティング システムのタスク マネージャでチェックできます。アプリケーションの**状態**が実行中か**応答なし**か表示されます。

テスト プロジェクトの Freeze Diagnostics プロパティに応じて、テスト エンジンはアプリケーションが応答を停止した場合には自動的にアプリケーションを終了させます。テスト エンジンがアプリケーションを終了させる場合、エラーメッセージ “The process does not respond. It will be terminated.”がテストログに出力されます。Freeze Diagnostics プロパティの表示と変更に関する詳細は、Diagnosing Application Freezes を参照してください。

フリーズしたアプリケーションが AQtrace Reporter を含んでいる場合、TestComplete は、問題の原因をより早く見つけるための詳細なエラーレポートを自動的に生成することに注意してください。詳細は、Diagnosing Application Freezes を参照してください。

テストを作成したときとオブジェクトのツリー モデルが異なっている

- **デスクトップ アプリケーションの場合**

  TestComplete は、*Tree(ツリー)* と *Flat(フラット)* の 2 つのオブジェクト ツリー モデルをサポートします。ひとつのモデル用に作成されたテストは他方のモデルとは互換性がありません。Flat モデルは、TestComplete 3 およびそれ以前で使用されていました。TestComplete 4 からは、デフォルトが Tree モデルになっています。古いバージョンで作成したテストを実行する場合は、アクティブなモデルを確認する必要があります。これを確認するには:

- Project Explorer パネルに切り替えます (メイン メニューで View | Project Explorer を選択します)。
- Project Explorer パネルでプロジェクトを右クリックします。コンテキスト メニューから Edit | Properties を選択します。project editor が開き、プロパティ ページがアクティブになります。
- ページの左にあるプロパティ グループのツリーから General を選択します。右の Object ツリー モデル設定の値をチェックします。

- **Web ページの場合**

  Web ページのテストをしている場合、Web オブジェクト用に TestComplete が使用する tree モデルになっていることを確認する必要があります。(Web Tree Models を参照してください)。これを確認するには:

  - Project Explorer に切り替えます。メインメニューで View | Project Explorer を選択します)。
  - Project Explorer パネルでプロジェクトを右クリックします。コンテキスト メニューから Edit | Properties を選択し、ます。project editor が開き、プロパティ ページがアクティブになります。

  - ページの左にあるプロパティ グループから Open Applications | Web Testing を選択します。テストを作成したときに使用されたモデルと同じモデルが設定されていなければなりません。

'Ambiguous Recognition of the Tested Object (テストするオブジェクトのあいまいな認識)' メッセージについて

ここでは、アプリケーション オブジェクトのあいまいな認識による問題の見つけ方と修正方法を説明します。

エラーメッセージについて

毎回 TestComplete がオブジェクトの検出に失敗するとき、テスト エンジンはオブジェクトの親オブジェクトのツリーを通して、親オブジェクトのマッピング条件であいまいさを見つけようと繰り返します。何かあいまいなところを検出した場合、TestComplete は‘Ambiguous Recognition of the Tested Object’ 警告メッセージを表示し、エラーが発生します。このメッセージは、オブジェクト マッピングであいまいさがあることを示しています。この問題の原因を見つけ、それを除去する方法を見てみましょう。

問題の診断

あいまいな認識問題の原因となるオブジェクトを見つけるには:

1. プロジェクトのテスト ログを表示します。
2. Test Log パネルでこの警告メッセージの行を選択します。
3. Additional Information パネルを表示します。
   このパネルには、警告の原因に関する補足情報が含まれています。たとえば、マッピング条件に一致する複数のオブジェクトの名前が含まれることがあります。
4. Additional Information パネルでマップされたオブジェクトの名前を含むリンクをクリックします。
   TestComplete が Name Mapping editor を開き、Mapped Objects ツリーで必要案オブジェクトをハイライト表示します。

問題の解決

上記の手順であいまいな認識を引き起こすオブジェクトを見つけます。Name Mapping editor の Mapped Objects ツリーでオブジェクトを選択します。TestComplete が右側のペインでこのオブジェクトのマッピング条件を表示します。上記で説明したように、警告メッセージはテストするアプリケーションには、指定した条件に一致する複数のオブジェクトがあることを示しています。

そのため、推奨する解決方法は、アプリケーションでひとつのオブジェクトだけに合うマッピング条件を指定することです。

'Mapped Item Has the Extended Find Attribute Enabled (マップされた項目は拡張検索属性が有効)' メッセージについて

ここでは、マップされたオブジェクト用に Extend Find を使用することに関するトラブルシュートとオブジェクトの識別問題の修正方法を説明します。

警告メッセージについて

Name Mapping プロジェクト項目は、テストするアプリケーションのオブジェクトのためにテストでこれらのオブジェクトを参照するために使用される名前を定義することを可能にします。オブジェクトを Name Mapping スキームに追加し、カスタムの名前を定義するとき、アプリケーションでこのオブジェクトを一意に識別するために使用される条件を指定します。

識別プロパティと値の組み合わせに加えて、さらに柔軟にマッピングを設定することが可能な別の設定もあります。これらの設定のひとつに Extended Find 属性があります。これは、TestComplete がアプリケーションのオブジェクト階層の特定のレベルでオブジェクトを探すだけでなく、すべてのネストされたレベルでも検索を指示します。この属性は、アプリケーションのオブジェクト階層が動的なときに役に立ちます。しかし、これはテストするアプリケーションでマッピング設定に一致するオブジェクトが複数ある場合にあいまいなオブジェクト認識を引き起こすことがあります。

テスト実行中にマップされたオブジェクトが見つけられず、オブジェクトの親オブジェクトが Extended Find 属性が有効になっている場合、親オブジェクトがあいまいな識別のために選択が正しくないことを意味することがあります。この問題の可能性のある原因を通知するため、TestComplete はテストログに警告メッセージを出力します。問題の説明は、マップ設定に一致するテストするアプリケーションのオブジェクトのマップされていない完全な名前といっしょに Extended Find 属性が有効になっているマップされた名前を含みます。検出されたオブジェクトが正しいかどうか確認できます。または、マッピング設定を変更することができます。

警告を引き起こしたオブジェクトの識別

警告を引き起こしたオブジェクトを見つける方法はいくつかあります。可能性のあるアプローチをいくつか紹介します:

- **フルネームにより Object Browser でオブジェクトを見つける**

  テスト ログの Additional Information パネルに表示される問題の説明には、マッピング設定に一致するテストするアプリケーションのオブジェクトのフルネームが含まれています。テストするアプリケーションがそのとき実行している場合、このオブジェクトを見つけて、正しいオブジェクトかどうか調べるために Object Browser でそのオブジェクト階層を見ることができます。

  1. テスト ログで警告メッセージを選択し、次に Additional Information パネルを選択します。
  2. 問題の説明に含まれているオブジェクト名のフルネームを書き留めます。たとえば、それをメモ帳にコピー＆ペーストします。
  3. Object Browser に切り替えます。
  4. オブジェクトツリーでテストするアプリケーションのプロセスを見つけて、記載したオブジェクトが見つかるまでそのオブジェクト階層を展開します。
  5. オブジェクトを右クリックし、[Highlight on Screen] を選択します。TestComplete は、テストするアプリケーションでこのオブジェクトを点滅する赤い枠で示します。

  テストするアプリケーションが問題のオブジェクトを含んでいない場合、以下のアプローチを試してください:

- **デバッグ モードでテストを実行中にそのオブジェクトを見つける**

  デバッグモードでテストを実行し、問題の操作の前にブレークポイントを設定し一時停止するのは良い案です。これは、テストするアプリケーションを警告が発生する状態にし、オブジェクトの識別問題を診断する簡単な方法です。

  最初に、Name Mapping エディタでそのオブジェクトを見つける必要があります:

  1. テスト ログで警告メッセージを選択し、次に **Additional Information** パネルを選択します。

2. マップされたオブジェクト名のリンクをクリックします。TestComplete が Name Mapping エディタを開いて、指定したマップされたオブジェクトを選択します。

これで、デバッグモードでテストを実行し、テストするアプリケーションを調査するためテストを一時停止できます:

1. テストログで警告をダブルクリックします。TestComplete がテストを開いて警告の引き金になった操作をハイライトします。
2. この行にブレークポイントを設定します。これをするには、その行の横をエディタでクリックするか F9 を押します。(このショートカットは、Cutomize Keyboard ダイアログで変更できます)
3. **Test Engine** ツールバーの **[Enable Debugging]**  ボタンが押されていることを確認します。
4. テストを実行します。TestComplete がブレークポイントに達したとき自動的にテスト実行を一時停止します。
5. Name Mapping エディタに切り替えて、問題のオブジェクトを右クリックしコンテキストメニューから **[Show in Object Browser]** を選択します。TestComplete が、Object Browser を開き、マッピング条件に一致するオブジェクトを選択します。
6. Object Browser でこのオブジェクトを右クリックし、**[Highlight on Screen]** を選択します。TestComplete がアプリケーションでそのオブジェクトを点滅する赤い枠で示します。

- **テストログにオブジェクトのスクリーンショットを出力する**

テストログを分析するときにテストするアプリケーションを実行できない、またはトリガーとなる警告の状態にアプリケーションを進めるのに長い時間を要する場合があります。このような場合、テストログに問題のオブジェクトのスクリーンショットを出力する別の操作をテストに追加することができます。警告の結果になる直前の操作の前にこの出力操作を追加します。この方法でテストを変更した後で、次にテストを実行し、同じ警告がログに出力されたときに、警告の原因となるオブジェクトを明確に知ることができます。

テストにスクリーンショットのログ操作を追加するには、次のようにします:

1. テストログで警告を選択し、次に Additional Information パネルを選択します。

2. マップされたオブジェクト名を含むリンクをクリックします。TestComplete は、Name Mapping エディタを開き、指定したマップされたオブジェクトを選択します。
3. エディタの右上のコーナーに表示されるマップされたオブジェクトのフルネームを書き留めます。または、クリップボードにコピーします。
4. テスト ログに切り替えして戻り、警告をダブルクリックします。TestComplete が、警告のトリガーとなった操作を含むテストを開き、この操作をハイライトします。
5. 問題の操作の前に Post Screenshot 操作を挿入します (テストがキーワード テストの場合)、または Log.Picture メソッドをコールします (テストがスクリプトの場合)。スクリーンショットを撮るオブジェクトを指定するパラメータで、オブジェクトのフルネームを使用します。スクリーンショットのログ操作の設定に関する詳細は、Posting Images to the Log を参照してください。
6. テストを保存して実行します。
7. テストの実行が終了した後で、テストログを開きます。
8. ログのスクリーンショットのエントリを探して、選択します。次に、画像を表示するため Picture パネルを開きます。

問題の解決

警告メッセージの原因となるオブジェクトを識別した後で、更なるアクションはそれが正しいオブジェクトかどうかに依存します:

- 検出したオブジェクトが必要とするオブジェクトではなかった場合、これは現在のマッピング設定が正確ではなく、必要なオブジェクトの一意の識別を保証しないことを意味します。ほとんどの場合、マッピング設定が正しくない識別プロパティと値を含みます、またはこれらの設定に一致するオブジェクトがアプリケーションで複数存在します。この問題を解決するには、Name Mapping エディタで問題のオブジェクトのマッピング設定を見直し、そのオブジェクトを一意に識別するため修正します。
- 検出したオブジェクトが必要とするオブジェクトであった場合も、マッピング設定を正しくし、さらに正確にする必要があるかもしれません。この方法は、あいまいなオブジェクト認識の可能性を除去できます。たとえば、関連する識別プロパティを追加したり、必要な子オブジェクトを指定したりできます。また、Extended Find 属性がそのオブジェクト識別に実際に必要な場合に有効にし、そうでない場合はそれを無効にします。

様々なマッピング設定を変更する方法については、Modifying Mapping Criteria を参照してください。

' Incomplete Keyboard Input'(不完全なキーボード入力) 問題の解決

警告について

TestComplete は、オブジェクトへのキーボード入力をシミュレートする Key メソッドを提供します。TestComplete が指定されたストリングで定義されたキー押下をシミュレートできない場合、“Incomplete keyboard input” 警告がテスト ログに出力されます。ここでは、問題の可能性のある原因とその解決方法を説明します。

問題の診断

1. テストログを開き、“Incomplete keyboard input” 警告を見つけます。
2. Additional Information パネルをアクティブにし、エラーの詳細な説明を見ます。キーボード入力を受け取るオブジェクト名といっしょに、問題のキーストローク ストリングが含まれています。
3. テストログの “Incomplete keyboard input” 警告をダブルクリックします。TestComplete が、問題の Keys コマンドを含むテストを開き、それをハイライトします。

可能性のある原因

- キーストローク ストリングの構文が不正である

  Keys メソッドで受付られるストリングは、[Del]、[F5]、[Left]、などのように特別なキー定数を含むことができます(完全なリファレンスは、Method 説明を参照してください)。問題のストリングが不正なキー定数を含む場合、“Incomplete keyboard input” 警告を受け取ります。この問題を解決するには、ストリングの構文が正しいか確認してください:

  - キーが正しいスペルである (大文字/小文字は重要です)。
  - 使用されている修飾文字 ([, ^, ~ および !) が 2 重である。
  - [ReleaseLast] 定数は適切な [Hold] 定数に続きます。

- コンピュータに接続されているキーボード レイアウトを使用して入力できない文字がストリングに含まれている。

  キーボード レイアウトは、キーボードから入力できる文字を指定します。Keys メソッドは、ユーザーが入力したように実際のキーボードからの入力をシミュレートするため、コンピュータに接続されたキーボードレイアウトはシミュレートする文字を含む必要があります。異なるキーボード レイアウトで利用可能な文字についての情報は、Microsoft の下記の Web サイトを参照してください。

  Windows Keyboard Layouts
  http://msdn.microsoft.com/en-us/goglobal/bb964651.aspx

  この問題を解決するには、シミュレートするストリングの文字を調査し、必要な入力言語、キーボード レイアウトをインストールしてください。コントロール パネルの [地域と言語のオプション] を使用してキーボード レイアウトをインストールできます。詳細については、Windows のドキュメントを参照してください。

- ストリングに東アジア言語の文字が含まれているが、適切な IME がインストールされていない。

  東アジア言語の文字 (日本語、中国語、韓国語など)を入力するには、適切な入力メソッドエディタ(IME)が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。IME をインストールするには、コントロール パネルで [地域と言語オプション]を使用します。詳細は、Windows ドキュメントを参照してください。

- 高度なテキスト サービスがテストするアプリケーションに適用されていない。

  Windows XP および Windows Server 2003 で、不完全なキーボード入力は、高度なテキスト サービスの拡張サポートにより解決されます。これをするには、以下のステップに従ってください:

  - コントロール パネルを開きます。

- 地域と言語オプションを開きます。
- [言語] タブに切り替えて、[詳細] ボタンをクリックします。[テキスト サービスと入力言語] ダイアログが表示されます。
- [詳細設定] タブに切り替えて、[詳細なテキスト サービスのサポートをプログラムのすべてに拡張する] オプションを選択します。
- OK をクリックし、変更を保存し、[テキスト サービスと入力言語] ダイアログを閉じます。

注意: 設定を有効にするためコンピュータを再起動する必要があります。

[お問い合わせ]

TestComplete に関するサポート情報が、下記の Web ページに掲載されていますので、お問い合わせする前に、ご確認いただきますようお願いいたします。

http://www.xlsoft.com/jp/products/smartbear/support.html

上記ページで解決できない場合、下記のお問い合わせページからご質問ください。

http://www.xlsoft.com/jp/qa

TestComplete のバージョン情報、テスト環境(OS、テストしているアプリケーションのタイプなど)、問題の詳細情報を記入してください。 また、エラー画面、エラーを再現できる簡単なテストプロジェクトなどが用意できる場合は、ご提供ください。