


HTML Table Checkpoint Extension

HTML Table Checkpoint Extension

The HTML Table Checkpoint script extension lets you add new functionality to TestComplete via a custom plug-in. This extension allows you to retrieve and compare HTML tables. It is provided by the HTMLTableCheckpoint.tcx package. This topic contains information about the HTML Table Checkpoint extension and describes how you can install it into TestComplete.

The HTML Table Checkpoint extension provides the functionality needed to create [HTML Table checkpoints](#). The extension adds the  HTML Table Checkpoint buttons to the Recording and Tools toolbars, a number of dialog boxes to TestComplete's UI and the [HTMLTableCheckpoint](#) program object.

How to Install the Extension

To install the HTML Table Checkpoint script extension, run the HTMLTableCheckpoint.tcx file. The Installing Extension - HTMLTableCheckpoint.tcx dialog will appear. In the dialog select the folder for the installation and click OK. This will install the plug-in into TestComplete. To check whether the extension is installed and enabled, use the Install Script Extensions dialog:

- Select File | Install Script Extensions from TestComplete's main menu to invoke the Install Script Extensions dialog.
- Check that the HTML Table Checkpoint extension is listed in the dialog and that its check box is selected. If the extension is unselected, its functionality is disabled.

HTML Table Checkpoint Extension Files

The HTMLTableCheckpoint.tcx package contains the following extension files:

- htcTableSelectorForm.aqfrm - The [Create HTML Table Checkpoint](#) dialog.
- ctcForm.aqfrm - The [Copy Text to Clipboard](#) dialog (displayed at design time).

- htmlTableCode.js - Script unit that contains functions executed during script recording or at design time.
 - htcIcon.bmp - The icon of the items added to the Recording and Tools toolbars.
 - Description.xml - Used to install the extension in TestComplete. Contains information about the extension, the items to be added to the Recording and Tools toolbar and about the HTMLTableCheckpoint program object.
-

© 2009 AutomatedQA Corp.

[Send feedback on this topic](#)



HTML Table Checkpoints Extension

HTML Table Checkpoints

When testing web pages, you may need to check whether the tested HTML table contains specific data. To do this, you typically compare the table's contents with a baseline copy. We call these comparison actions HTML Table checkpoints. The following topics provide detailed information about these checkpoints:

[HTML Table Checkpoints - Overview](#)

Explains how HTML Table checkpoints function, provide information about requirements and tells you where to find the script code that implements the checkpoint's functionality.

[Creating HTML Table Checkpoints in Scripts](#)

Describes how to create and use HTML Table checkpoints in scripts.

[HTML Table Checkpoints in Keyword Tests](#)

Explains how to insert HTML Table checkpoints into keyword tests.

See also

[Creating HTML Table Checkpoints in Scripts](#) | [HTML Table Checkpoints in Keyword Tests](#) | [HTMLTableCheckpoint Object](#)

© 2009 AutomatedQA Corp.

[Send feedback on this topic](#)



HTML Table Checkpoints

HTML Table Checkpoints - Overview

This topic provides an overview of HTML Table checkpoints. It explains how these checkpoints work, describes the requirements and explains where you can find the code that implements the checkpoint's functionality.

[About the HTML Table Checkpoint](#)

[Requirements](#)

[Creating the Checkpoint](#)

[Checkpoint's Source Code](#)

About the HTML Table Checkpoint

HTML Table checkpoints are used to compare HTML tables' data with a baseline copy of data. The baseline copy is stored as an element of the Stores | XML collection of your test projects. These elements are created when you create these checkpoints. The current contents of the HTML table is stored in the stores project item and then used as a baseline copy.

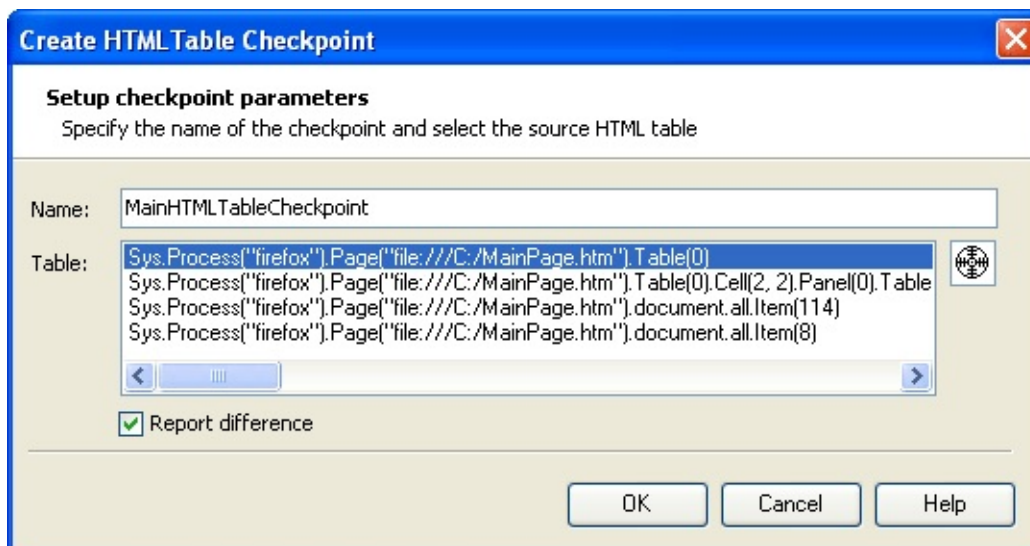
The HTML Table Checkpoint script extension appends the [HTMLTableCheckpoint](#) object to TestComplete's object model. This object contains the [HTMLTableCheckpoint.Compare](#) method that performs the comparison. It compare data of an HTML table with the stored baseline copy.

Requirements

HTML Table checkpoints are created with the script extensions technology. The toolbar items, dialogs and scripting objects that are used to create and work with HTML Table checkpoints are only available if the [HTML Table Checkpoint extension](#) is installed in TestComplete.

Creating the Checkpoint

You create the checkpoints with the [Create HTML Table Checkpoint](#) dialog. You may call it during test recording or at design time:



In the dialog you specify the HTML table whose contents will be verified and the name of the project element, to which the table's contents will be saved.

Upon closing the dialog, TestComplete displays the [Copy Text to Clipboard](#) dialog that displays the generated script code that performs the comparison. Here is an example of this code:

[Show Example](#)

For detailed information on inserting the checkpoint into script code, see [Creating HTML Table Checkpoints in Scripts](#).

The HTML Table Checkpoint extension does not provide a keyword test operation that allows you to use the checkpoint from keyword tests. So, to use the checkpoint functionality from keyword tests, use the operations that will call the `HTMLTableCheckpoint.Compare` method from keyword tests. For instance, you can use the Run Code Snippet or Call Object Method operations. For more information about this, see [HTML Table Checkpoints in Keyword Tests](#).

Checkpoint's Source Code

The `HTMLTableCheckpoint.tcx` file includes the checkpoint's source code. This file is a ZIP archive that has the `.tcx` extension. So, you can use any zip archive

utility to unpack the files from it. For information about the extension's files, see [HTML Table Checkpoint Extension Files](#).

See also

[Creating HTML Table Checkpoints in Scripts](#) | [HTML Table Checkpoints in Keyword Tests](#) | [HTMLTableCheckpoint Object](#)

© 2009 AutomatedQA Corp.

[Send feedback on this topic](#)




HTML Table Checkpoints

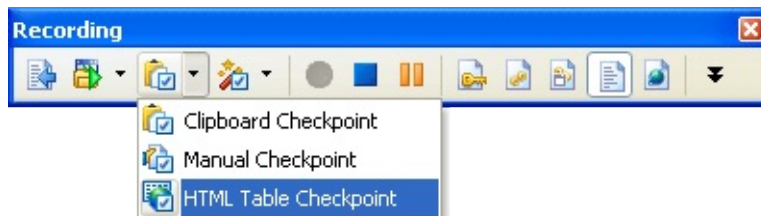
Creating HTML Table Checkpoints in Scripts


This topic describes how to create an HTML Table checkpoint and insert it into script code. For information on how to insert HTML Table checkpoints into keyword tests, see [HTML Table Checkpoints in Keyword Tests](#).

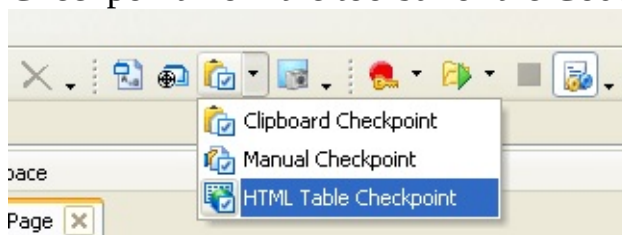
You can only create HTML Table checkpoints for web pages that are **Note:** displayed in web browsers supported by TestComplete, because the test scripts have access to elements in these web pages.

To create an HTML Table checkpoint, follow these steps:


- You can create HTML Table checkpoints both during script recording and at design time:
 - To create the checkpoint during script recording, select  HTML Table Checkpoint from the Recording toolbar:



- To create the checkpoint at design time, select the  HTML Table Checkpoint from the toolbar of the Code Editor.



- After selecting the HTML Table Checkpoint toolbar item, the [Create HTML Table Checkpoint](#) dialog will appear.
- In the dialog:

- Specify the name of the checkpoint.
- Specify the source HTML table by dragging the  Finder Tool icon to it.
- Set the Report difference option. If this check box is selected, then the created checkpoint will post messages on the differences between the compared tables to the test log. If the check box is clear, no messages will be posted to the log.
- Press OK to close the dialog.

Upon closing the dialog, TestComplete will check whether your project contains the Stores project item, the XML collection and the specified element. If the project does not have any of these items, TestComplete will display dialog boxes asking to create them.

After the table is added to the XML collection, TestComplete generates the comparison code. If you create the checkpoint during script recording, the generated code will be inserted into the recorded script automatically. So, you can continue recording after the Create HTML Table Checkpoint dialog is hidden.

If you create the checkpoint at design time, TestComplete will invoke the [Copy Text to Clipboard](#) dialog with the generated code in it. You can then copy this code in the dialog and paste it in your script code. Below is an example of the generated code:

[Show Example](#)

The generated code includes a call to the [HTMLTableCheckpoint.Compare](#) method of the [HTMLTableCheckpoint](#) object. This method returns True or False according to the comparison results. The method has a required reportDifference parameter that specifies whether the method should post notifications about the differences to the test log. If this parameter is True, the method posts warning messages about each difference that was found (the warning message type is used because some tests may expect that the stored and actual data differs, that is, in general, a difference is not an error). If the reportDifference parameter is False, the method does not post any message to the test log.

See also

[HTML Table Checkpoints](#) | [HTML Table Checkpoints in Keyword Tests](#)

© 2009 AutomatedQA Corp.

[Send feedback on this topic](#)




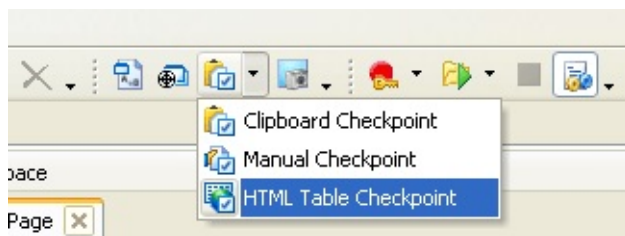
HTML Table Checkpoints

HTML Table Checkpoints in Keyword Tests

The HTML Table Checkpoint extension does not provide special operations for creating HTML Table checkpoints. So, to use the checkpoint functionality in your keyword test, use the keyword test operations that will call the [HTMLTableCheckpoint.Compare](#) scripting method from the test. For instance, you can use the Run Code Snippet or Call Object Method operation to call this method. You cannot insert these operations during test recording, so you can insert HTML Table checkpoints into your keyword tests only at design time. Here is a possible scenario.

First, we need to generate the comparison code. To do this:

- Select  HTML Table Checkpoint from the Tools toolbar (if this toolbar is hidden, right-click somewhere in the toolbar area and select Tools from the ensuing context menu):



This will invoke the [Create HTML Table Checkpoint](#) dialog.

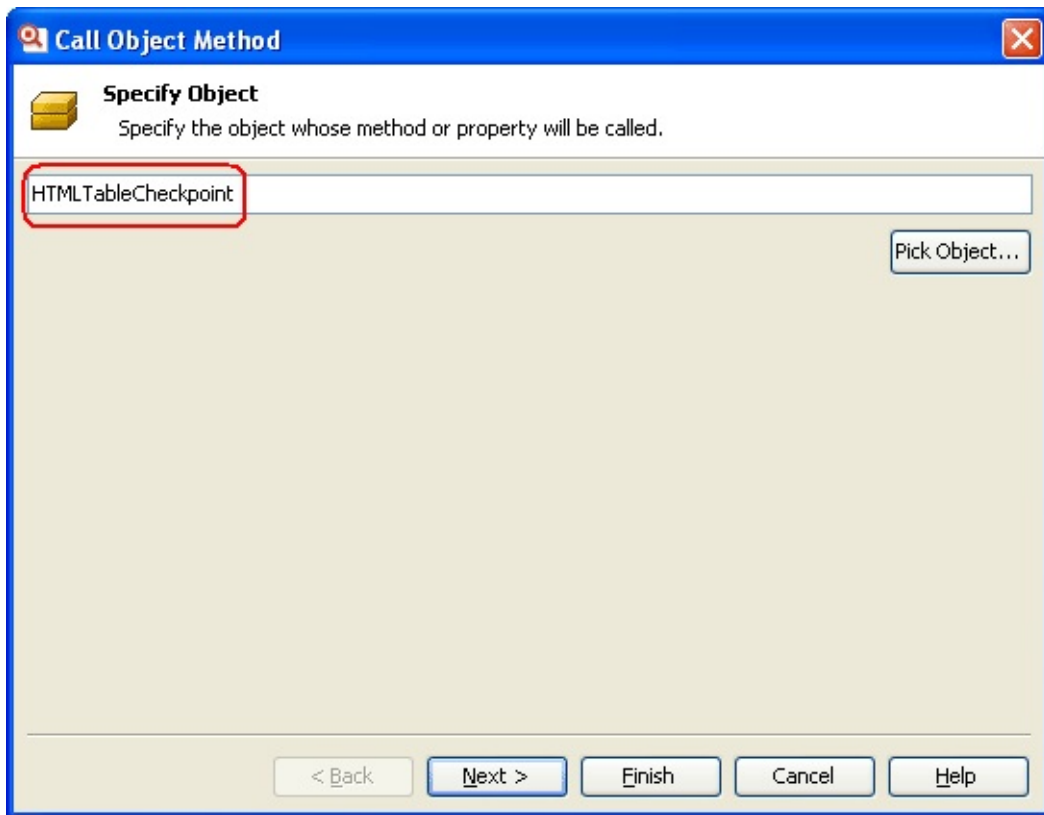
- In the dialog, specify the tested HTML table and the element of the Stores | XML collection that will store the baseline copy of the table's data. Press OK to close the dialog and to create the checkpoint.
- After you close the dialog, the [Copy Text to Clipboard](#) dialog with generated script code in it will appear. Close this dialog.

Now we may add keyword test operations that will use this code to perform the comparison:

- Open your keyword test for editing.
- Add the Call Object Method operation to your test.

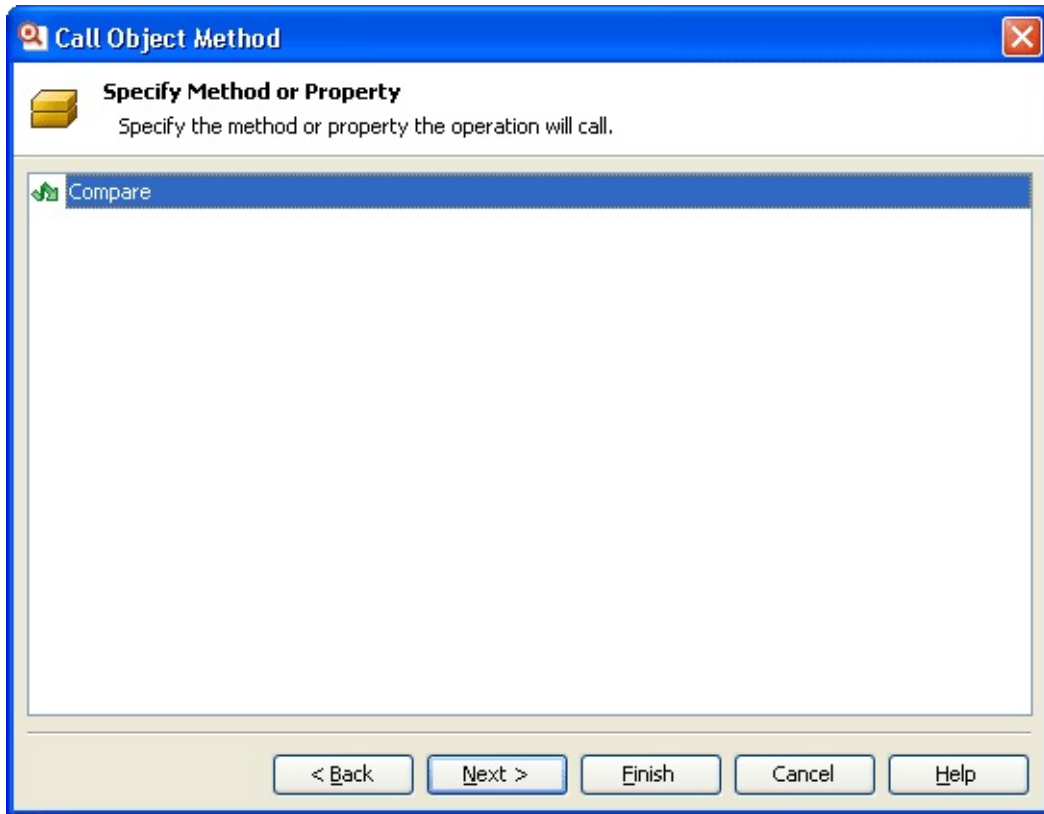
After you dragged the operation from the Operations list to your test in the KeywordTest Editor, TestComplete will display the Operation Parameters wizard.

- On the first page of the wizard, you specify the object whose method will be called. Enter **HTML Table Checkpoint** into the edit box:



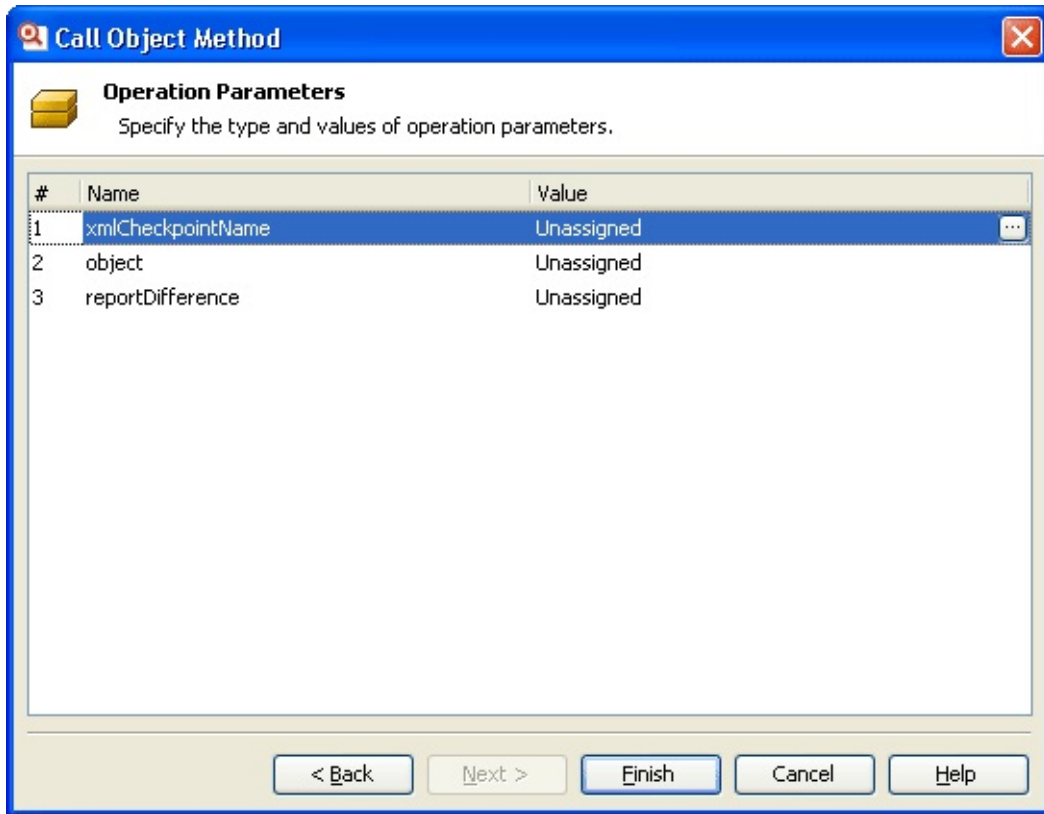
Click Next to continue.

- On the next page of the Operation Parameters wizard you can choose the method to be called. In our case, the list contains only one item - Compare.

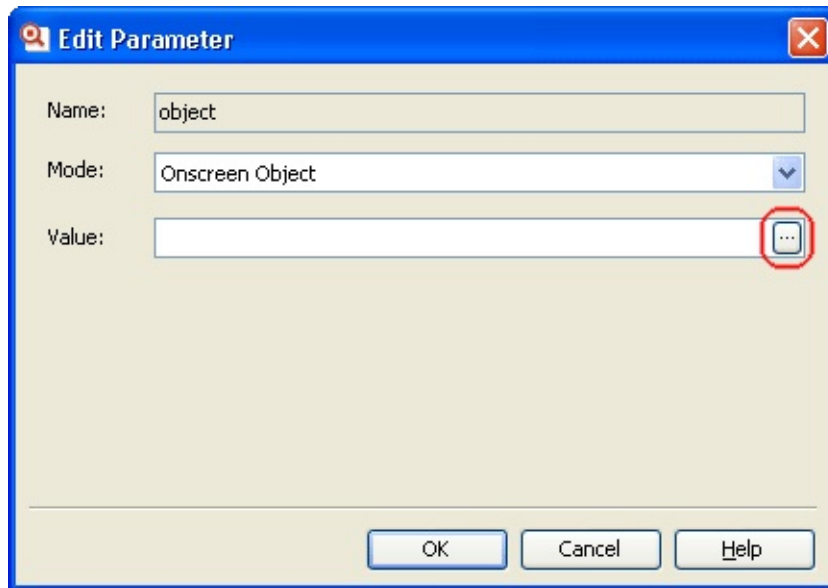


Select it and click Next to continue.

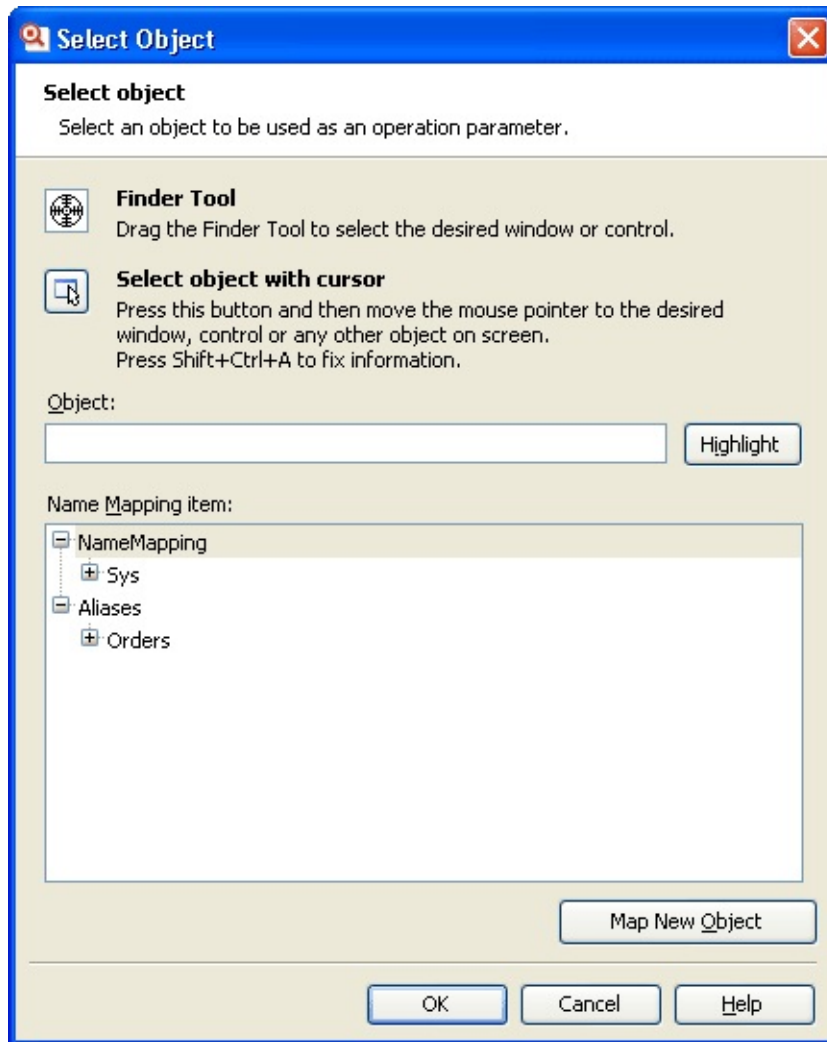
- On the next page of the Operation Parameters wizard you can specify the parameters of the method.



- The xmlCheckpointName parameter specifies the name of the Stores | XML collection's element that contains the baseline copy. Type the same name that you specified in the Create HTML Table Checkpoint dialog when creating the checkpoint.
- The object parameter specifies the name of the HTML table to be verified. To enter it:
 - Click the ellipsis button of the Value cell. This will invoke the Edit Parameter dialog.
 - In the dialog, choose **Onscreen Object** from the Mode drop-down list and then click the ellipsis button of the Value edit box:



This will invoke the Select Object dialog:



- In this Select Object dialog, drag the Finder tool (🌐) to the table to choose it.

When dragging, TestComplete displays the red frame around the objects that are under the mouse cursor. If you move the cursor over a table's cell, TestComplete will only highlight this cell, not the table. To choose the table, place the cursor on the table's border until TestComplete draws the red frame around the table.

- Close the Select Object dialog with the OK button. You will return to the Edit Parameter dialog.
- Close the Edit Parameter dialog with the OK button. You will

return back to the Operation Parameters wizard.

The name of the select HTML table will be shown as the value of the object parameter

- The reportDifference parameter specifies whether the checkpoint will log messages about the differences to the test log. If this parameter is `true`, the checkpoint will post messages to the log. If the parameter is `false`, the differences will not be reported automatically.
- Click Finish to close the wizard and to add the operation to the test.

See also

[HTML Table Checkpoints](#) | [Creating HTML Table Checkpoints in Scripts](#)

© 2009 AutomatedQA Corp.

[Send feedback on this topic](#)



HTML Table Checkpoint Extension

HTMLTableCheckpoint Object

The HTMLTableCheckpoint object provides a scripting interface to the [HTML Table Checkpoint](#) elements of the **Stores | XML** collection. Using this object you can access a stored table and compare it with a tested HTML table. The object contains only one method.

[HTMLTableCheckpoint Object Methods](#)

Note that the HTMLTableCheckpoint object was created with the script extensions technology. The object is only available if the [HTML Table Checkpoint script extension](#) is installed in TestComplete.

See also

[HTML Table Checkpoint Extension](#) | [HTMLTableCheckpoint.Compare](#)

© 2009 AutomatedQA Corp.
[Send feedback on this topic](#)



HTMLTableCheckpoint Object

HTMLTableCheckpoint.Compare

The method compares an HTML Table Checkpoint element with a tested HTML table.

Declaration

HTMLTableCheckpointObj.Compare(xmlCheckpointName, object, reportDifference)

Parameters

xmlCheckpointName	[in] Required String
object	[in] Required String
reportDifference	[in] Required Boolean
Result	Boolean

Description

Use the Compare method to compare the values stored in project element, to which the [HTMLTableCheckpoint](#) object corresponds, with actual data of the HTML table that is specified by the object parameter.

Parameters

The method has the following parameters:

xmlCheckpointName

Specifies the project element that contains the baseline data with which the tested table's data will be compared. This element belongs to the Stores | XML collection of your project.

object

Specifies the full name of the tested HTML table

reportDifference

Specifies whether the method posts a warning message about each difference to the test log.

Return Value

True, if the comparison is successful and False otherwise.

Remarks

The Compare method returns True or False indicating the comparison result. It also posts a warning message for each difference found during the comparison. If there are several differences, the method will post several warning messages. To disable this functionality, pass False to the reportDifference parameter when calling the method.

For sample code, see [Creating HTML Table Checkpoints in Scripts](#).

See also

[HTMLTableCheckpoint Object](#) | [Creating HTML Table Checkpoints in Scripts](#) | [HTML Table Checkpoints in Keyword Tests](#)

© 2009 AutomatedQA Corp.

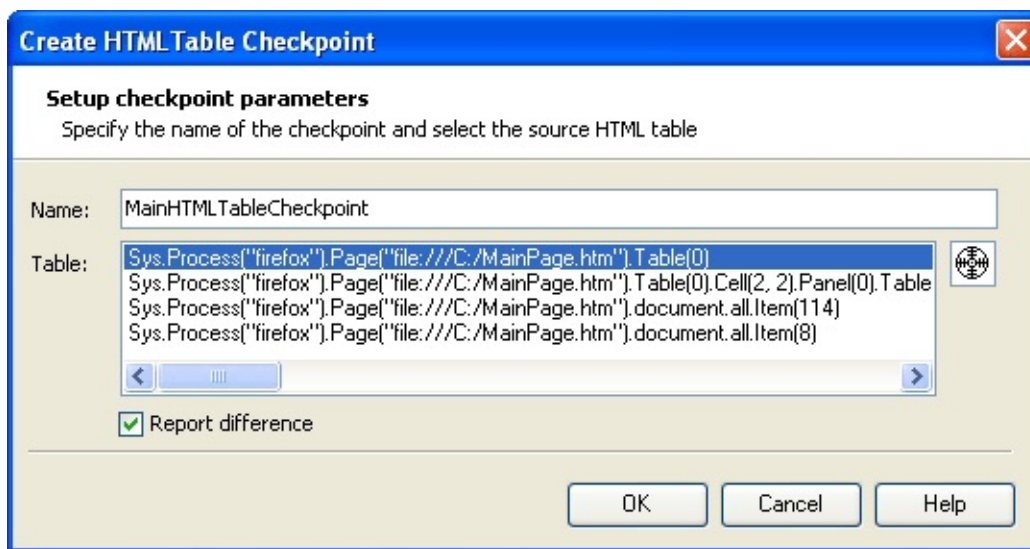
[Send feedback on this topic](#)



HTML Table Checkpoint Extension

Create HTML Table Checkpoint Dialog


Use the Create HTML Table Checkpoint dialog to retrieve data that will be used to create a new [HTML Table checkpoint](#). The dialog is called when you add an HTML Table checkpoint to your keyword test or script routine during test recording or at design time.



The HTML Table Checkpoint compares data of a tested HTML table with a baseline copy. The baseline data is stored as an element of your project's Stores | XML collection. In the dialog, you can select the table to be compared and the name of the project element that will store the baseline data. This element will be created after you close the dialog with the OK button.

To specify the stores element, enter the desired name of the element into the Name edit box. This name will be used to address the element from scripts, so, it must be a valid identifier. Scripting languages supported by TestComplete use different naming rules for identifiers. To specify a name that will match the rules of any language, enter a string that starts with a letter and that contains only letters, digits and underscore symbols.

To specify the HTML table, whose data will be retrieved and stored in the

HTML Table Checkpoint element, click the  Finder Tool icon and then drag it to the HTML table, whose contents will be verified. While you are dragging, TestComplete displays a red frame around the table that is under the mouse cursor. The full names of the table and its child tables are displayed in the Table box of the dialog. After selecting the desired table from this list, a red frame will highlight it on screen.

Note: If the project's Tree model option is set to Hybrid, the Table box displays the table names in both DOM and Tree formats.

If the Report difference check box is selected, the created checkpoint will post messages about the differences between the compared tables to the test log. If the check box is clear, no messages will be posted. This check box is a visual analogue to the reportDifference parameter of the [HTMLTableCheckpoint.Compare](#) method.

To create a checkpoint, press OK. To close the dialog and abort the checkpoint creation, press Cancel.

See also

[HTML Table Checkpoints - Overview](#) | [Creating HTML Table Checkpoints in Scripts](#) | [HTML Table Checkpoints in Keyword Tests](#)

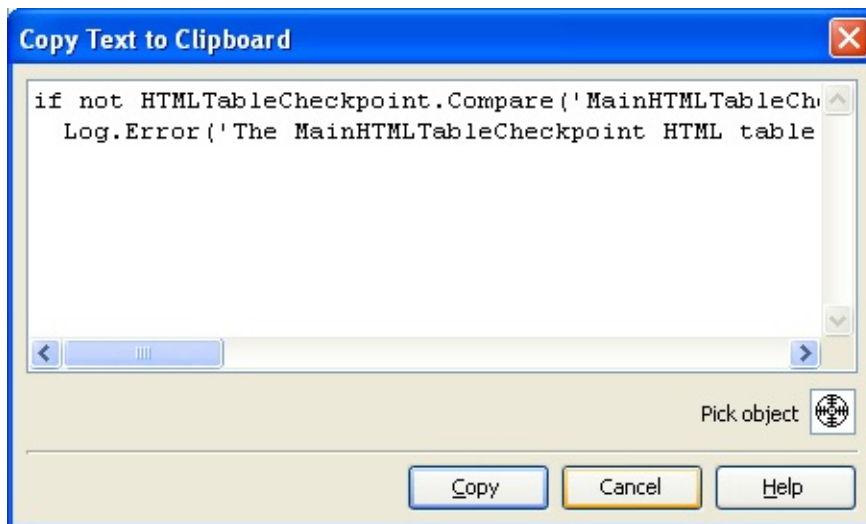




HTML Table Checkpoint Extension

Copy Text to Clipboard Dialog

This dialog appears when you insert an [HTML Table checkpoint](#) into script code at design time. The dialog contains the script code generated to perform the comparison actions.

Here is a sample image of the dialog:



You can insert an additional statement into your code in the dialog. Note, that to add the name of an onscreen object you can pick the desired object from the screen with the  Finder tool. To do this, drag the  icon to the desired object and release the mouse button over this object (while dragging, a red frame will indicate the object that is currently under the mouse pointer). TestComplete will insert the full name of the selected object into the code box.

Press Copy to store the text to the clipboard. You can then switch to the Code Editor and paste this text into the script code. Cancel will close the dialog without copying the text.

See also

[HTML Table Checkpoint Extension](#)