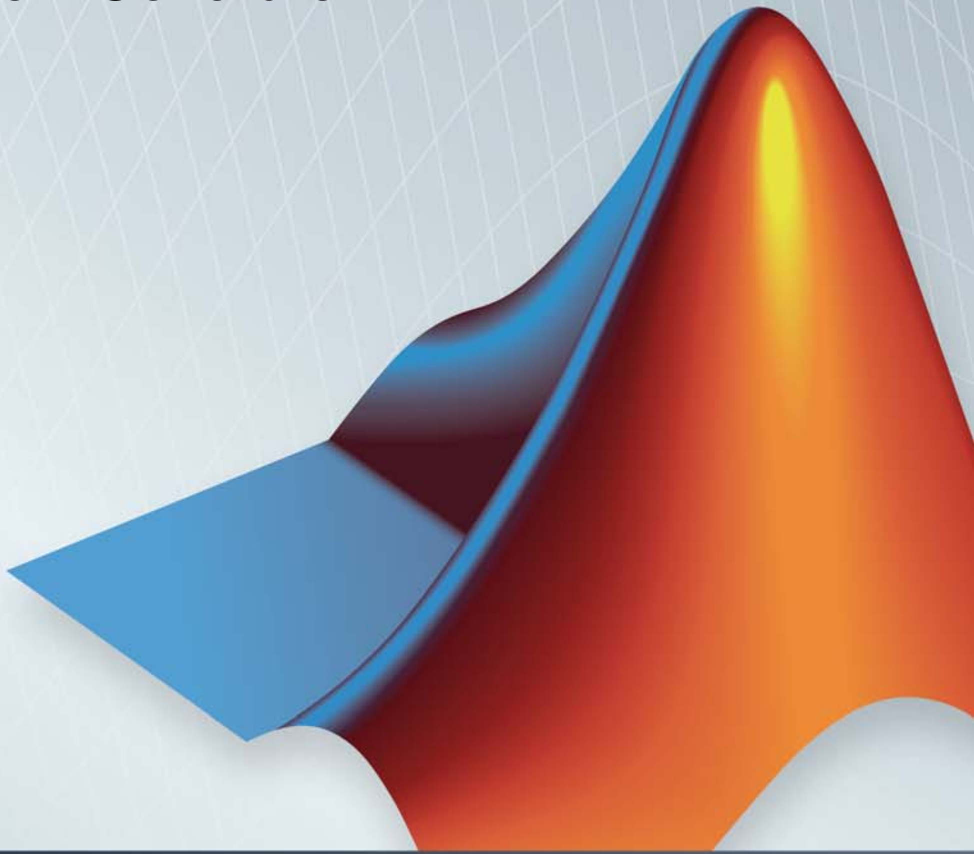
# MATLAB® Report Generator™

## User's Guide

**R**2013**b**



MATLAB®

MathWorks®

## How to Contact MathWorks

| | |
|---|---|
| www.mathworks.com | Web |
| comp.soft-sys.matlab | Newsgroup |
| www.mathworks.com/contact_TS.html | Technical Support |
| suggest@mathworks.com | Product enhancement suggestions |
| bugs@mathworks.com | Bug reports |
| doc@mathworks.com | Documentation error reports |
| service@mathworks.com | Order status, license renewals, passcodes |
| info@mathworks.com | Sales, pricing, and general information |

☎ 508-647-7000 (Phone)

508-647-7001 (Fax)

✉ The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

*MATLAB® Report Generator™ User's Guide*

### Trademarks

### Patents

## Revision History

| | | |
|---|---|---|
| January 1999 | First printing | New (Release 11) |
| December 2000 | Second printing | Revised (Release 12) |
| June 2004 | Third printing | Revised for Version 2.02 (Release 14) |
| August 2004 | Online only | Revised for Version 2.1 |
| October 2004 | Online only | Revised for Version 2.1.1 (Release 14SP1) |
| December 2004 | Online only | Revised for Version 2.2 (Release 14SP1+) |
| April 2005 | Online only | Revised for Version 2.2.1 (Release 14SP2+) |
| September 2005 | Online only | Revised for Version 2.3.1 (Release 14SP3) |
| March 2006 | Online only | Revised for Version 3.0 (Release 2006a) |
| September 2006 | Online only | Revised for Version 3.1 (Release 2006b) |
| March 2007 | Fourth printing | Revised for Version 3.2 (Release 2007a) |
| September 2007 | Fifth printing | Revised for Version 3.2.1 (Release 2007b) This publication was previously for MATLAB® and Simulink®. It is now for MATLAB® only. |
| March 2008 | Online only | Revised for Version 3.3 (Release 2008a) |
| October 2008 | Online only | Revised for Version 3.4 (Release 2008b) |
| October 2008 | Online only | Revised for Version 3.5 (Release 2008b+) |
| March 2009 | Online only | Revised for Version 3.6 (Release 2009a) |
| September 2009 | Online only | Revised for Version 3.7 (Release 2009b) |
| March 2010 | Online only | Revised for Version 3.8 (Release 2010a) |
| September 2010 | Online only | Revised for Version 3.9 (Release 2010b) |
| April 2011 | Online only | Revised for Version 3.10 (Release 2011a) |
| September 2011 | Online only | Revised for Version 3.11 (Release 2011b) |
| March 2012 | Online only | Revised for Version 3.12 (Release 2012a) |
| September 2012 | Online only | Revised for Version 3.13 (Release 2012b) |
| March 2013 | Online only | Revised for Version 3.14 (Release 2013a) |
| September 2013 | Online only | Revised for Version 3.15 (Release 2013b) |

# Contents

## Getting Started

## 1

## Create Your First Report

## 2

## 4 Generate a Report

## 5 Add Content with Components

## 6 Create Custom Components

## 7 Create Custom Stylesheets

## 8 Comparing XML Files

## 9 Components — Alphabetical List

## 10 Functions – Alphabetical List

## Index

# Getting Started

# MATLAB Report Generator Product Description

### Generate documentation for MATLAB® applications and data

The MATLAB Report Generator™ product automatically documents tasks that you perform in MATLAB, such as analyzing and visualizing data and developing algorithms. It enables you to run MATLAB code and capture the graphics and data as they are produced. You can use the prebuilt templates or create a template that incorporates your own styles and standards.

## Key Features

- Enables push-button documentation from MATLAB
- Documents MATLAB functions and scripts and their outputs as they are executed
- Lets you create and distribute documentation templates
- Provides extensible components and style sheets
- Creates reports in multiple documentation formats, including HTML, PDF, RTF, Microsoft® Word, and XML
- Enables conditional documentation generation via logical template components, such as IF, THEN, ELSE, and WHILE

# Present MATLAB Code and Results

You can use the MATLAB Report Generator to create reports for sharing your MATLAB code and presenting the results of the code.

In addition, MATLAB provides several methods for presenting MATLAB code and results, including:

- MATLAB `publish` command
- MATLAB `notebook` command

MATLAB enables you to publish your MATLAB code quickly, so that you can describe and share your code with others, even if they do not have MATLAB software. You can publish in various formats, including HTML, XML, and LaTeX. If Microsoft Word or Microsoft PowerPoint® applications are on your Microsoft Windows® system, you can publish to their formats as well.

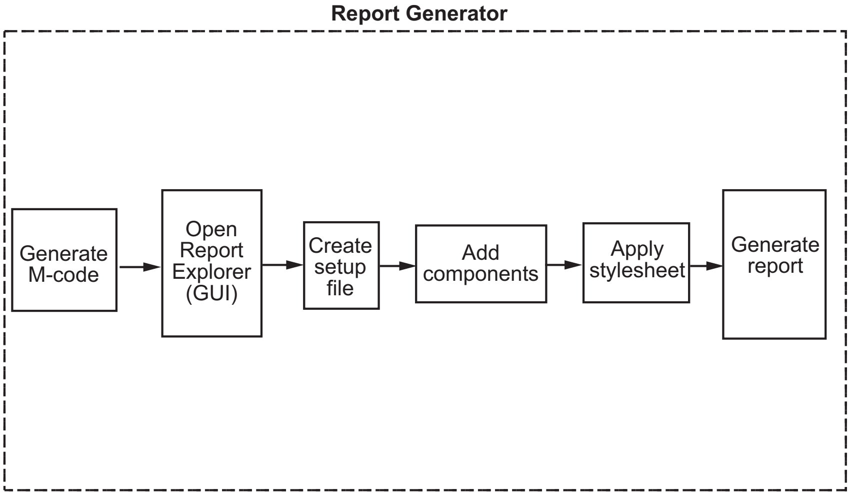
On Windows platform, you can use the `notebook` command to create a Microsoft Word document that contains text, MATLAB commands, and the output from MATLAB commands. The document is a record of an interactive MATLAB session annotated with text or a document embedded with live MATLAB commands and output.

To compare the MATLAB tools for presenting MATLAB code and results and MATLAB Report Generator, see “Options for Presenting Your Code”.

# Report Creation Workflow

1. Open the Report Explorer.
2. Create a report setup file. For details about report setups, see "Report Setup".
3. Add content by adding to the report setup file existing components or custom components that you create. For details about using components, see "Addition of Components".
4. Apply styles and standards to the report by choosing an existing stylesheet or a custom stylesheet. For details on stylesheets and attributes, see "Layout Stylesheets".
5. Generate the report. See "Report Generation".

The following figure illustrates a typical MATLAB Report Generator workflow.



To practice using this report creation workflow, see "Addition of Components".

# How MATLAB Report Generator and MATLAB Software Interact

The MATLAB Report Generator and MATLAB software interact to create reports. You can access the Report Explorer from the MATLAB command line.

The following table summarizes these interactions.

| User Interface | MATLAB Report Generator Interaction | Description |
|---|---|---|
| Report Explorer | The Report Explorer provides a graphical interface. For more information, see "Report Explorer" on page 1-8. | Use the Report Explorer to edit existing report templates, components, stylesheets, and attributes, or to customize your own. |
| MATLAB command line | Enter commands at the MATLAB command line to:<br>• Start the Report Explorer<br>• Create and modify report template files<br>• Apply stylesheets<br>• Specify output formats for reports<br>• Generate reports | The following MATLAB commands work with the MATLAB Report Generator software:<br>• `report` — Start the Report Explorer.<br>• `setedit` — Edit a report template with the Report Explorer.<br>• `rptconvert` — Convert a source file created by the report generation process to the desired output format.<br>• `rptlist` — List `.rpt` files in the current path. |

# Report Components

## Types of Report Components

Components are MATLAB objects that specify the content of a report. The MATLAB Report Generator provides a set of components for specifying the types of content that commonly occur in MATLAB-based reports. The Simulink® Report Generator provides additional components to facilitate generation of reports from Simulink models. You can also create custom components to handle content specific to your application.

Using the Report Explorer, you can interactively combine components to create a report setup (see "Report Setup") that specifies the content of a particular report or type of report. You can then run the setup from the Report Explorer or the MATLAB command line to create instances of the report.

Use a combination of the following types of components in your report setup file, based on the goals for the report.

| Type of Component | Description |
|---|---|
| "Report Structure Components" on page 5-4 | Include a title page, chapters, sections, paragraphs, lists, tables, and other standard document structure elements. |
| "Table Formatting Components" on page 5-5 | Organize generated content into tables. |
| "Property Table Components" on page 5-6 | Display tables with property name/property value pairs for objects. |
| "Summary Table Components" on page 5-17 | Display tables with specified properties for objects. |
| "Logical and Looping Components" on page 5-22 | Run child components a specified number of times. There are several looping components, including logical loops and Handle Graphics® loops. |

Use the Report Explorer to add components to a report setup file and to specify component properties.

# Report Explorer

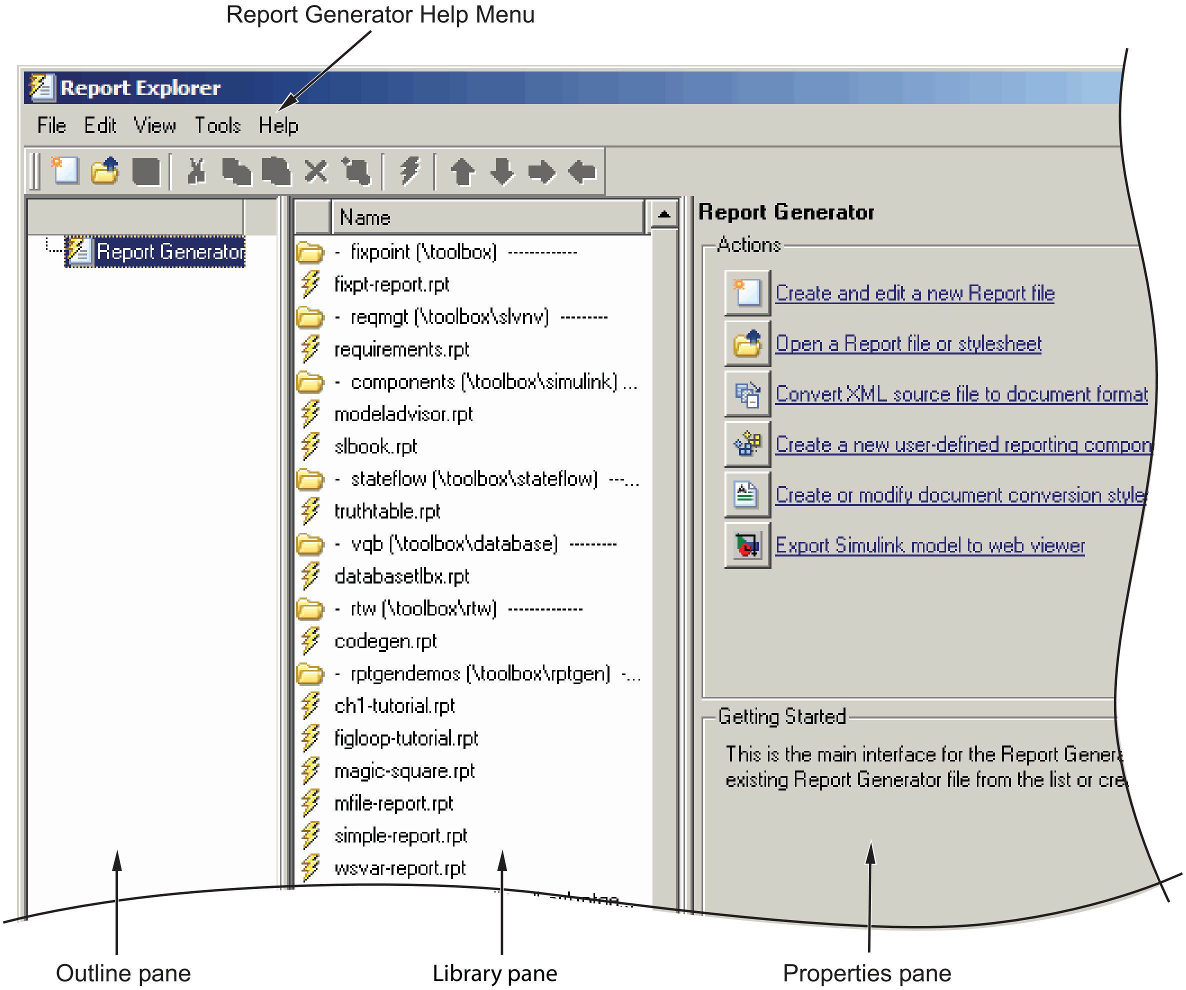
## About the Report Explorer

Use the *Report Explorer* to:

- Create and modify report setup files.
- Apply stylesheets to format the generated report.
- Specify the report file format.
- Generate reports.

Open the Report Explorer using one of these approaches:

- From the MATLAB Toolstrip, in the **Apps** tab, in the **Database Connectivity and Reporting** section, click **Report Generator**.
- In the MATLAB Command Window, enter `report`.



- fixpoint (\toolbox) ------------
fixpt-report.rpt
- reqmgt (\toolbox\slvnv) --------
requirements.rpt
- components (\toolbox\simulink) ...
modeladvisor.rpt
slbook.rpt
- stateflow (\toolbox\stateflow) ---...
truthtable.rpt
- vqb (\toolbox\database) --------
databasetlbx.rpt
- rtw (\toolbox\rtw) -------------
codegen.rpt
- rptgendemos (\toolbox\rptgen) -...
ch1-tutorial.rpt
figloop-tutorial.rpt
magic-square.rpt
mfile-report.rpt
simple-report.rpt
wsvar-report.rpt

Create and edit a new Report file
Open a Report file or stylesheet
Convert XML source file to document format
Create a new user-defined reporting compon
Create or modify document conversion style
Export Simulink model to web viewer

Getting Started

This is the main interface for the Report Gener
existing Report Generator file from the list or cre

The Report Explorer has three panes:

- The *Outline pane* on the left shows the hierarchy of components in currently opened report setup files. Report components can reside within other report components, creating parent, child, and sibling relationships.
- The *Library pane* in the middle lists the objects available in the context of the Outline pane.

| Outline Pane Context | Library Pane Contents |
|---|---|
| No report setup file is open. | Reports |
| Report setup file is open. | Components |
| Stylesheet is open. | Stylesheet attributes |

- The *Properties pane* contents depend on the Outline pane context. If no report setup file is open, on the right displays tasks the Report Explorer can perform. If a report setup file is open, the Properties pane displays the properties for the item that is currently selected in the Options pane.

| Outline Pane Context | Properties Pane Contents |
|---|---|
| No report setup file is open. | Tasks that the Report Explorer can perform |
| Report setup file is open. | Properties for the item that is currently selected<br>After you create a report setup file, the Properties pane initially displays properties for the report setup file as a whole. |

**Tip** If the Report Explorer window opens with only two panes, one of the panes is hidden. You can move the vertical boundaries between the panes to reveal any hidden pane, or to make visible panes wider or narrower.

# Supported Report Formats

When the report-generation process first creates a report, it generates a DocBook XML source file. You can customize this XML as needed. For more information on how to customize DocBook XML, see the OASIS™ DocBook TC Web page at `http://www.oasis-open.org/committees/docbook` and the online version of DocBook: The Definitive Guide by Norman Walsh and Leonard Muellner, with contributions from Bob Stayton at http://www.docbook.org/tdg/en/html/docbook.html.

Next, the report-generation process converts the XML source to one of these user-specified report formats:

- Adobe® Acrobat® PDF
- Hypertext Markup Language (HTML)
- Microsoft Word (`.doc`)
- Rich Text Format (RTF)

**Note** RTF reports use placeholders (field codes) for dynamically generated content, such as page numbers or images.

On Windows platforms, to display that content, press **Ctrl+A**, and then press **F9**.

On Linux® and Mac platforms, use the field code update interface for the program that you are using to view the RTF document.

## Limitations for Report Formats

PDF reports only support bitmap (`.bmp`), jpeg (`.jpg`), and Scalable Vector Graphics (`.svg`). The SVG format is only supported for Simulink models and Stateflow® charts.

For reports that use the `Word Document` format, you must have Microsoft Word software installed on the machine that you use to generate the report.

2

# Create Your First Report

# Create a MATLAB Report

This example shows how to create a report that explains and illustrates magic squares – matrices whose columns, rows, and diagonals each add up to the same number (see the `magic` function reference in the MATLAB documentation).

To create this report, you perform these main tasks:

- "Create a Report Setup File" on page 2-3
- "Add Components to the Setup" on page 2-5

---

**Note** You do not need to know the MATLAB software to use this example. However, knowledge of MATLAB is helpful for understanding the MATLAB code that executes during report generation.

---

This example includes separate sections for different kinds of report creation and generation tasks. Each section builds on the previous sections. However, if you want to work through a later section without having done the previous sections, you can view the completed report setup file: `Magic Squares Report`.

# Create a Report Setup File

To set up the magic squares report, first create a setup file to store the setup. Then add MATLAB objects, called components, to the setup to specify the report content.

To create the report setup file:

1. Start a MATLAB software session.
2. Open the Report Explorer. From the MATLAB Toolstrip, in the **Apps** tab, in the **Database Connectivity and Reporting** section, click **Report Generator**.
3. Select **File > New** to create a report setup file. The new report setup has the default name `Unnamed.rpt`.
4. In the Properties pane on the right:
   - a. To save the report in the current working folder, select `Present working directory` from the **Directory** list.
   - b. Set **File format** to `web (HTML)` to create the report as an HTML file.
   - c. In the **Report description** text box, replace the existing text with the following text.

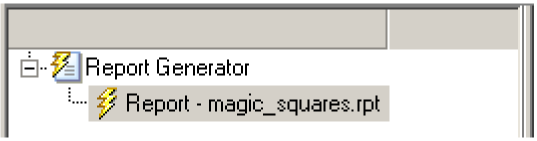
     **Tip** Copy and paste this text from the HTML documentation into the Report Explorer.

     ```
     This report creates a series of magic squares
     and displays them as images.

     A magic square is a matrix in which the
     columns, rows, and diagonal all add up to the
     same number.
     ```

**Note** When you change a Properties pane field, its background color changes. This indicates that there are unapplied changes to that field. As soon as you perform any action with another component, MATLAB Report Generator applies the changes, and the background color becomes white again.

5 Save your report. Select **File > Save As** and name your report setup file `magic_squares.rpt`.

The new file name appears in the Outline pane.



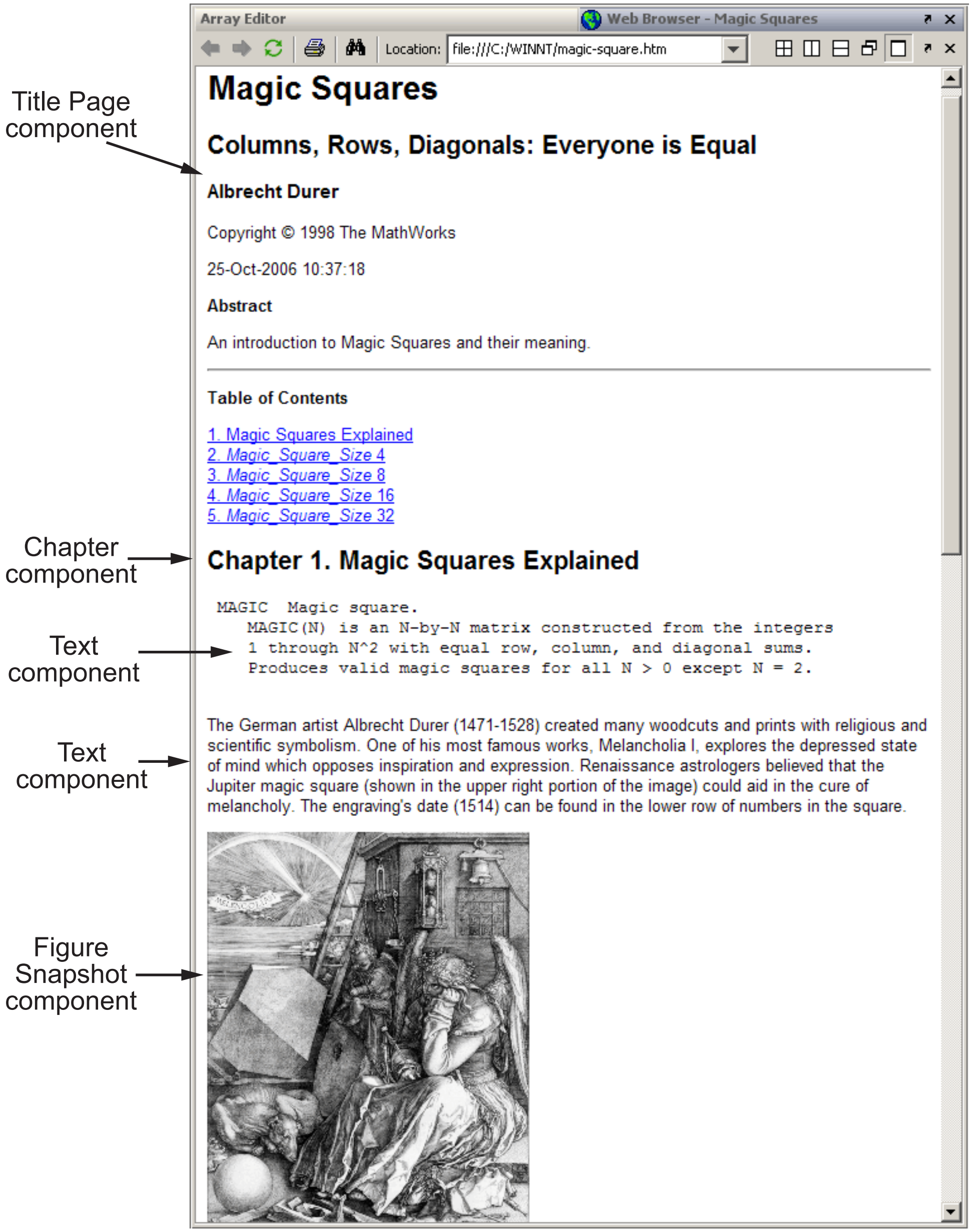
To create the content for the report, see .

# Add Components to the Setup

**In this section...**

## Report Components

Report components specify the information to include in the report. The following figure shows a sample page from the report that you create in this example, highlighting components that you use to produce the report.


Array Editor
Web Browser - Magic Squares
Location: file:///C:/WINNT/magic-square.htm
Title Page component
Chapter component
Text component
Text component
Figure Snapshot component

# Magic Squares

## Columns, Rows, Diagonals: Everyone is Equal

**Albrecht Durer**

Copyright © 1998 The MathWorks

25-Oct-2006 10:37:18

**Abstract**

An introduction to Magic Squares and their meaning.

**Table of Contents**

## Chapter 1. Magic Squares Explained

```
 MAGIC  Magic square.
    MAGIC(N) is an N-by-N matrix constructed from the integers
    1 through N^2 with equal row, column, and diagonal sums.
    Produces valid magic squares for all N > 0 except N = 2.
```

The German artist Albrecht Durer (1471-1528) created many woodcuts and prints with religious and scientific symbolism. One of his most famous works, Melancholia I, explores the depressed state of mind which opposes inspiration and expression. Renaissance astrologers believed that the Jupiter magic square (shown in the upper right portion of the image) could aid in the cure of melancholy. The engraving's date (1514) can be found in the lower row of numbers in the square.

## Specify Report Variables

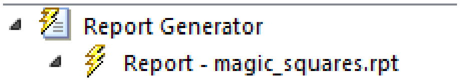
The magic squares report uses variables defined in the MATLAB workspace to specify the number and sizes of squares to display and whether to display the variables as tables of numbers or images of color-coded squares:

- The *magicSizeVector* variable specifies an array of magic square sizes
- *largestDisplayedArray* variable specifies the size of the largest magic square to be displayed as an array of numbers

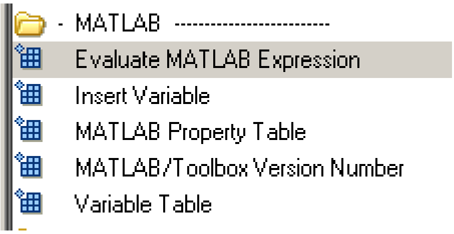
You could require that a user create these variables in the MATLAB workspace before running the report. However, a better solution is to let the report itself create the variables, using the `Evaluate MATLAB Expression` component.

To use the Evaluate MATLAB Expression component to define the report variables.

1 In the Outline pane on the left, select the root component of the report setup.



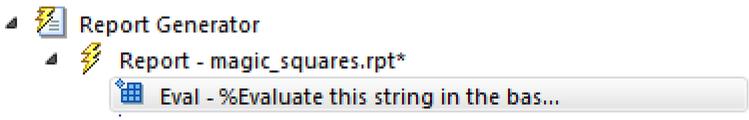
2 In the Library pane in the middle, under the `MATLAB` category, select `Evaluate MATLAB Expression`.


- MATLAB -------------------------
Evaluate MATLAB Expression
Insert Variable
MATLAB Property Table
MATLAB/Toolbox Version Number
Variable Table

3 In the Properties pane on the right, click the icon next to **Add component to current report** to insert the Evaluate MATLAB Expression component into the report.

You cannot edit the component information in the Properties pane until you have added the component to the report.

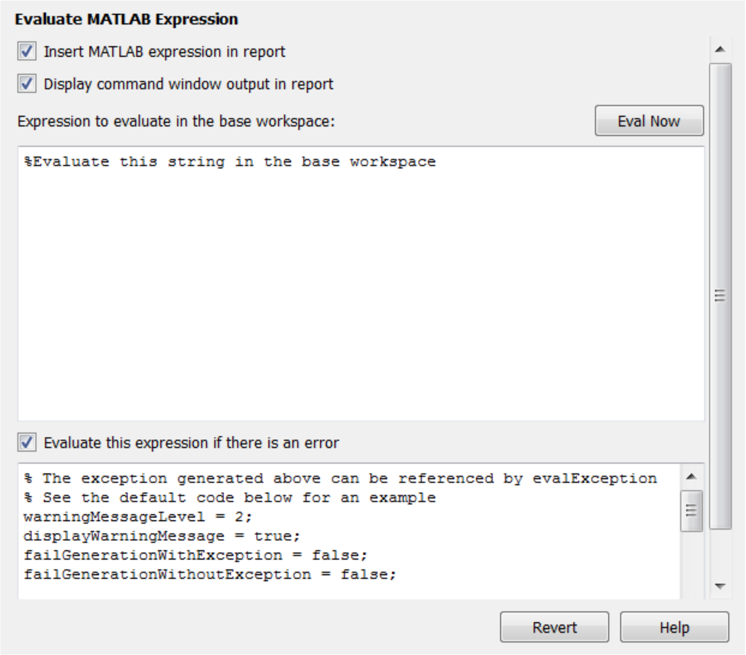
In the Outline pane, the `Eval` component appears under the `magic_squares` report.



Report Generator
Report - magic_squares.rpt*
Eval - %Evaluate this string in the bas...

The icon in the upper left corner of the `Eval` component indicates that this component cannot have child components. By default, any components you add with the `Eval` component selected are siblings to this component.

The options for the Evaluate MATLAB Expression component appear in the Properties pane.



**Evaluate MATLAB Expression**

☑ Insert MATLAB expression in report

☑ Display command window output in report

Expression to evaluate in the base workspace:

```
%Evaluate this string in the base workspace
```

☑ Evaluate this expression if there is an error

```
% The exception generated above can be referenced by evalException
% See the default code below for an example
warningMessageLevel = 2;
displayWarningMessage = true;
failGenerationWithException = false;
failGenerationWithoutException = false;
```

4 To exclude the MATLAB code details and its output in this report, clear the **Insert MATLAB expression in report** and **Display command window output in report** check boxes.

5 In the **Expression to evaluate in the base workspace** text box, replace the existing text with the following MATLAB code.

---

**Tip** Copy and paste this text from the HTML documentation into the Report Explorer.

---

```
%This MATLAB code sets up two variables
%that define how the report runs.
%magicSizeVector is a list of MxM
%Magic Square sizes to insert into
%the report. Note that magic
%squares cannot be 2x2.

magicSizeVector=[4 8 16 32];

%largestDisplayedArray sets the
%limit of array size that will be
%inserted into the report with the
%Insert Variable component.

largestDisplayedArray=15;
```

6 In the **Evaluate this expression if there is an error** text box, replace the existing text with the following text.

```
disp(['Error during eval: ', evalException.message])
```

This causes an error to display if the MATLAB code fails.

---

**Tip** To execute these commands immediately, in the top right corner of the Report Explorer, click the **Eval Now** button. This confirms that your commands are correct, to reduce the chances of report generation problems.

---

7 Save the report. Select **File > Save**.

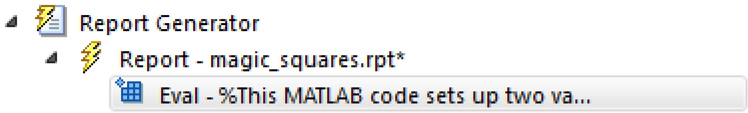
## Create a Title Page

**Note** This section builds on the previous tasks described in the step-by-step example summarized in “Create a MATLAB Report” on page 2-2.

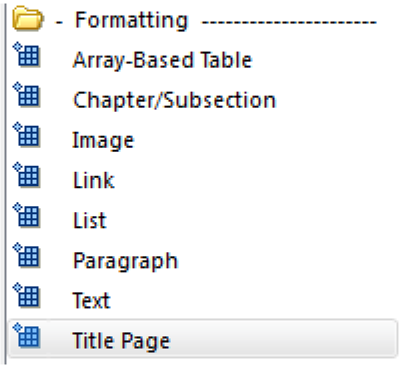
If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

To create a title page for the report, use the Title Page component.

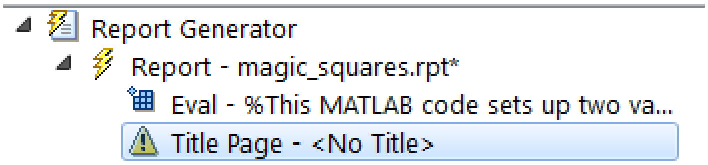
1 In the Outline pane on the left, select the `Eval` component.


Report Generator
Report - magic_squares.rpt*
Eval - %This MATLAB code sets up two va...

2 In the Options pane in the middle, under the `Formatting` category, double-click `Title Page` to add the component to the report.



Because the `Eval` component icon indicates that this component cannot have children, the `Title Page` component is a sibling of the `Eval` component. Likewise, the `Title Page` component also cannot have child components.



Report Generator

Report - magic_squares.rpt*

Eval - %This MATLAB code sets up two va...

Title Page - <No Title>

---

**Note** To use a Title Page component, you need to have a Chapter component in your report. You have not yet added a Chapter component, so the Properties pane displays a message indicating that a chapters is required for the Title Page component to appear correctly. Because later in this example you add Chapter components to this report, you can ignore that message.

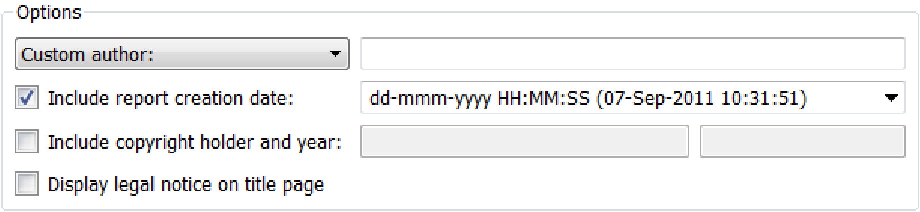
---

**3** In the Properties pane on the right, use the **Main** tab to enter the following title page information.

**a** In the **Title** text box, enter `Magic Squares`.

**b** In the **Subtitle** text box, enter `Columns, Rows, Diagonals:  Everyone is Equal`.

**c** Under **Options**, choose `Custom author` from the selection list.



Options

Custom author:

☑ Include report creation date: dd-mmm-yyyy HH:MM:SS (07-Sep-2011 10:31:51)

□ Include copyright holder and year:

□ Display legal notice on title page

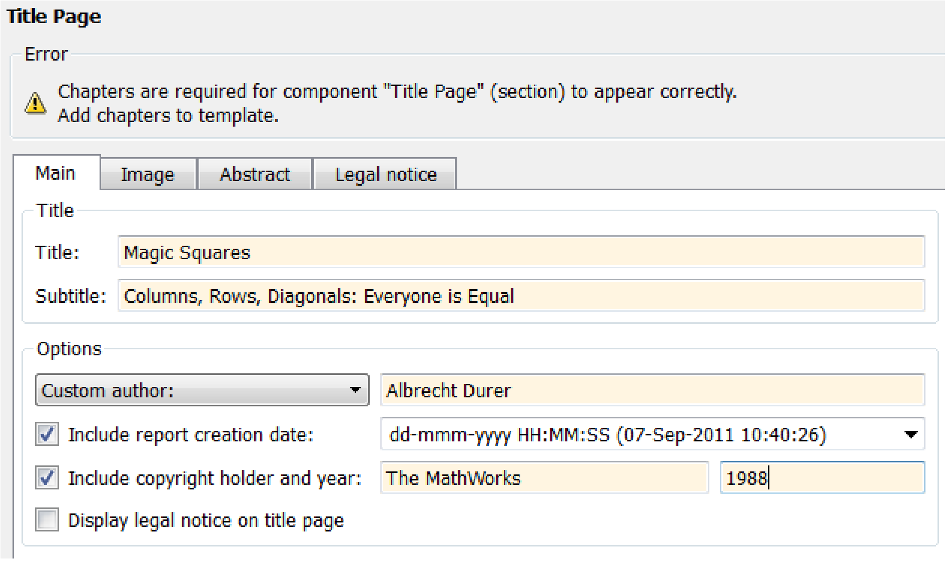
**d** In the field to the right of the **Custom author** field, enter `Albrecht Durer`.

Albrecht Dürer created an etching that contains a magic square. Your final report includes an image of that etching.

**e** Select the **Include copyright holder and year** check box.

**f** In the next text box, enter `The MathWorks`.

**g** In the second text box, enter 1988.



**Title Page**

Error

⚠ Chapters are required for component "Title Page" (section) to appear correctly.
Add chapters to template.

Title

Title: Magic Squares

Subtitle: Columns, Rows, Diagonals: Everyone is Equal

Options

Custom author: Albrecht Durer

☑ Include report creation date: dd-mmm-yyyy HH:MM:SS (07-Sep-2011 10:40:26)

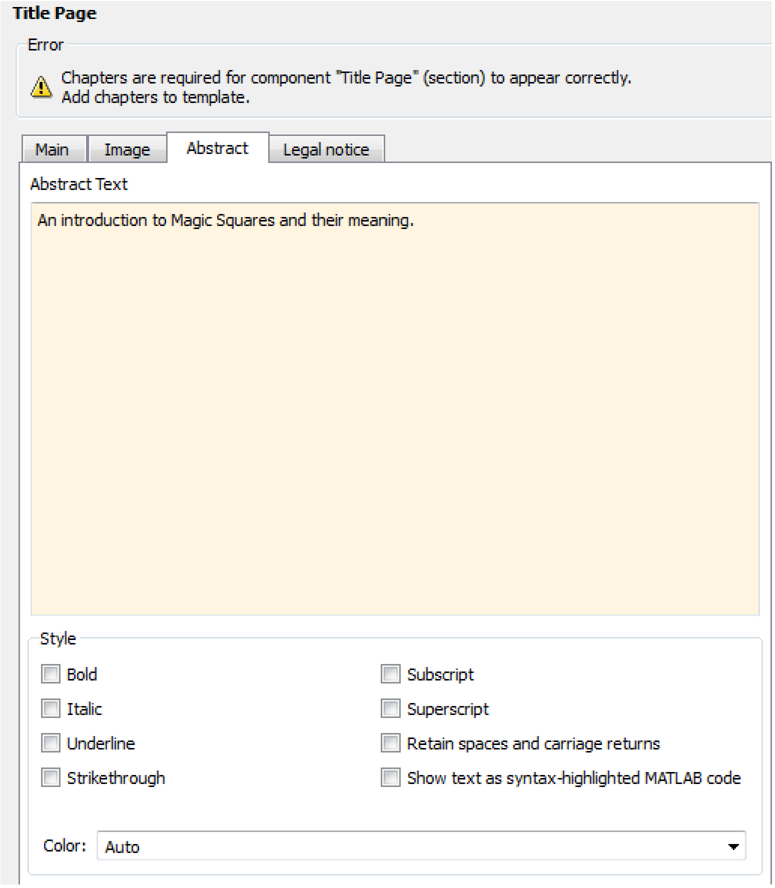
☑ Include copyright holder and year: The MathWorks 1988

□ Display legal notice on title page

**4** In the Properties pane, click the **Abstract** tab and then enter the following text:

```
An introduction to Magic Squares and their meaning.
```

The pane should look as follows:



**Title Page**

Error

⚠ Chapters are required for component "Title Page" (section) to appear correctly.
Add chapters to template.

Abstract Text

An introduction to Magic Squares and their meaning.

Style

| | |
|---|---|
| □ Bold | □ Subscript |
| □ Italic | □ Superscript |
| □ Underline | □ Retain spaces and carriage returns |
| □ Strikethrough | □ Show text as syntax-highlighted MATLAB code |

Color: Auto

**5** Save the report.

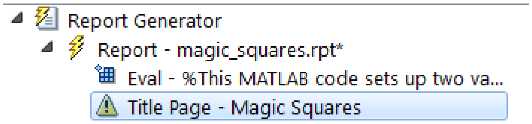
## Add a Chapter

**Note** This section builds on the previous tasks described in the step-by-step example summarized in “Create a MATLAB Report” on page 2-2.

If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

Add a chapter to the report by using the Chapter/Subsection component.

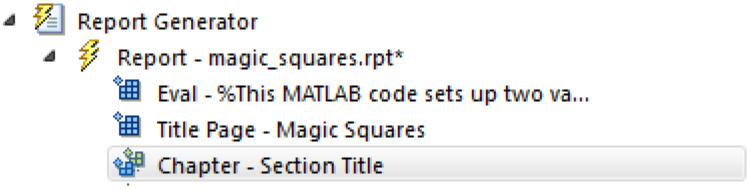
1 In the Outline pane on the left, select the `Title Page` component.



Report Generator
Report - magic_squares.rpt*
Eval - %This MATLAB code sets up two va...
Title Page - Magic Squares

2 In the Library pane in the middle, under the `Formatting` category, double-click `Chapter/Subsection`.

The Outline pane looks as follows.



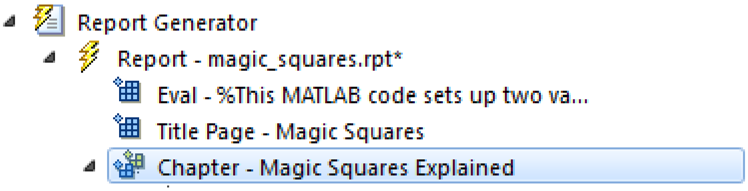
Report Generator
Report - magic_squares.rpt*
Eval - %This MATLAB code sets up two va...
Title Page - Magic Squares
Chapter - Section Title

The `Eval`, `Title Page`, and `Chapter` components are all child components of the report's top level, and are siblings of one another.

The `Chapter` component can have child components. The next section explains how to add child components to this `Chapter` component.

3 For the custom chapter title, in the Properties pane on the right, enter `Magic Squares Explained`.

The Outline pane changes to reflect the chapter title.



Report Generator
Report - magic_squares.rpt*
Eval - %This MATLAB code sets up two va...
Title Page - Magic Squares
Chapter - Magic Squares Explained

4 Save the report.

## Add Introductory Text to the First Chapter

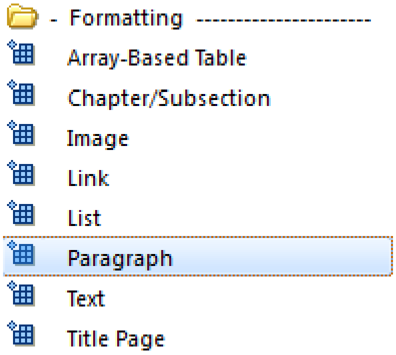
**Note** This section builds on the previous tasks described in the step-by-step example summarized in "Create a MATLAB Report" on page 2-2.

If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

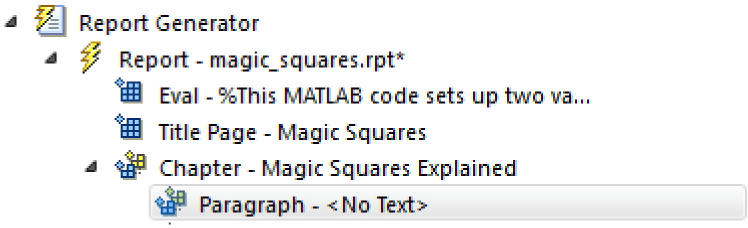
Include introductory text in the first chapter by adding the Paragraph and Text components.

1 In the Outline pane on the left, select the `Chapter` component.

2 In the Library pane in the middle, under the `Formatting` category, double-click `Paragraph`.



In the Outline pane, the new component appears as a child of the `Chapter` component.


Report Generator
Report - magic_squares.rpt*
Eval - %This MATLAB code sets up two va...
Title Page - Magic Squares
Chapter - Magic Squares Explained
Paragraph - <No Text>

3 By default, the Paragraph component inherits its text from its child components. Add two `Text` components.

---

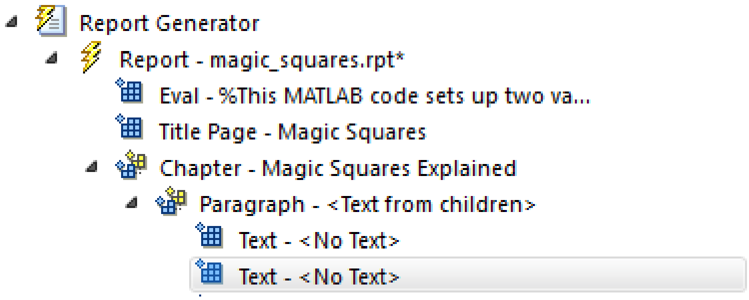
**Note** The `Text` component must have the `Paragraph` component as its parent.

---

4 In the Library pane, under the `Formatting` category, double-click `Text`.

5 Double-click `Text` again to add a second component.

The Outline pane looks as follows.



Report Generator
Report - magic_squares.rpt*
Eval - %This MATLAB code sets up two va...
Title Page - Magic Squares
Chapter - Magic Squares Explained
Paragraph - <Text from children>
Text - <No Text>
Text - <No Text>

6 In the Outline pane, select the first `Text` component.

7 In the **Text to include in report** text box, enter `%<help('magic')>`.

The `%` sign and angle brackets `<>` indicate to the MATLAB Report Generator software that this is MATLAB code to evaluate. The command `help('magic')` displays information about the MATLAB `magic` function.

8 In the Outline pane, select the second `Text` component.

9 In the **Text to include in report** text box, enter the following text.

---

**Tip** Copy and paste this text from the HTML documentation into the Report Explorer.

---

```
The German artist Albrecht Durer (1471-1528)
created many woodcuts and prints with religious
and scientific symbolism.  One of his most famous
works, Melancholia I, explores the depressed state
of mind that opposes inspiration and expression.
Renaissance astrologers believed that the Jupiter magic
square (shown in the upper right portion of the image)
could aid in the cure of melancholy. The engraving's
date (1514) can be found in the lower row of numbers
in the square.
```

10. Save the report.

The contents of the first chapter are now complete.

## Add an Image

**Note** This section builds on the previous tasks described in the step-by-step example summarized in “Create a MATLAB Report” on page 2-2.

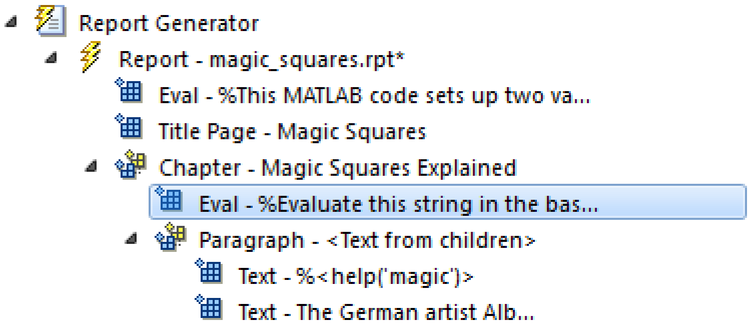
If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

The next steps create an image of Albrecht Dürer and include it in the report.

1. In the Outline pane on the left, select the `Chapter` component.

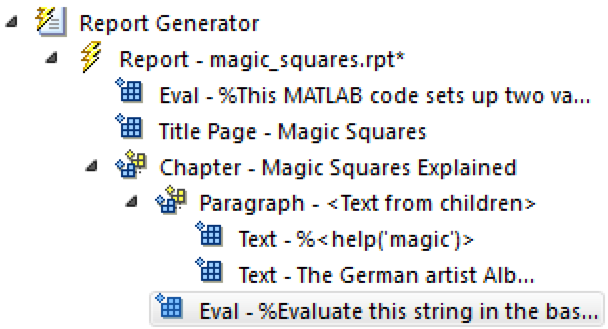
2. In the Library pane in the middle, under the `MATLAB` category, double-click `Evaluate MATLAB Expression`.

   The new component becomes a child of the `Chapter` component.



- Report Generator
  - Report - magic_squares.rpt*
    - Eval - %This MATLAB code sets up two va...
    - Title Page - Magic Squares
    - Chapter - Magic Squares Explained
      - Eval - %Evaluate this string in the bas...
      - Paragraph - <Text from children>
        - Text - %<help('magic')>
        - Text - The German artist Alb...

3 Move the `Eval` component under the `Paragraph` component so that the image follows the introductory text by clicking the **down** arrow on the toolbar.



- Report Generator
  - Report - magic_squares.rpt*
    - Eval - %This MATLAB code sets up two va...
    - Title Page - Magic Squares
    - Chapter - Magic Squares Explained
      - Paragraph - <Text from children>
        - Text - %<help('magic')>
        - Text - The German artist Alb...
      - Eval - %Evaluate this string in the bas...

4 With the `Eval` component still selected, do the following in the Properties pane on the right:

a Clear the **Insert MATLAB expression in report** and **Display command window output in report** check boxes. You do not want to include the code or its output in the report.

b In the **Expression to evaluate in the base workspace** text box, replace the existing text with the following MATLAB code.

**Tip** Copy and paste this text from the HTML documentation into the Report Explorer.

```
%This loads a self-portrait of Albrecht
%Durer, a German artist. There is a
%magic square in the upper right corner
%of the image.

durerData=load('durer.mat','-mat');
figure('Units','Pixels',...
'Position',[200 200 size(durerData.X,2)*.5 size(durerData.X,1)*.5 ]);

image(durerData.X);
colormap(durerData.map);
axis('image');
set(gca,...
    'Xtick',[],...
    'Ytick',[],...
    'Units','normal',...
    'Position',[0 0 1 1]);

clear durerData
```

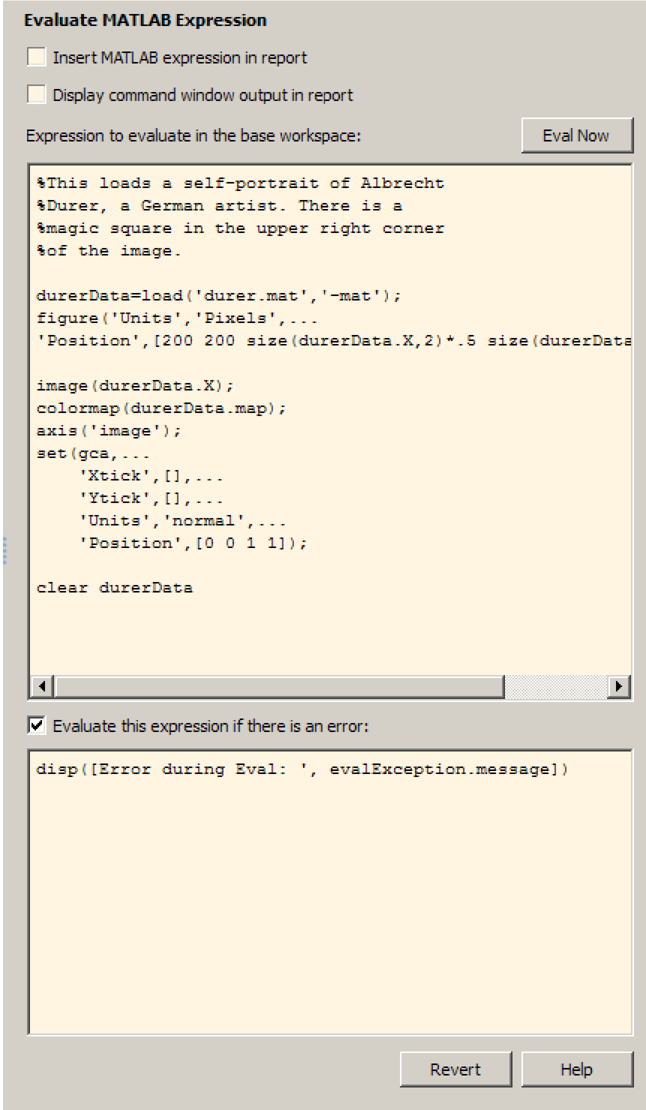
This MATLAB code displays the Dürer etching in a MATLAB figure window.

**c** In the **Evaluate expression if there is an error** text box, replace the existing text with the following text:

```
disp(['Error during eval: ', evalException.message])
```

This code executes if an error occurs while loading the Dürer etching.

The Properties pane on the right looks as follows.



Evaluate MATLAB Expression

□ Insert MATLAB expression in report

□ Display command window output in report

Expression to evaluate in the base workspace:

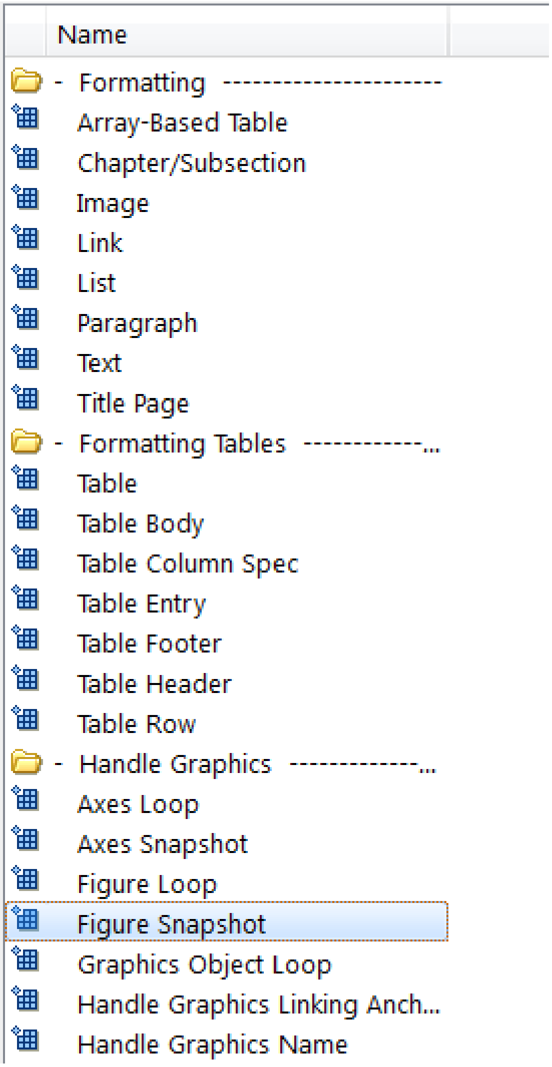
```
%This loads a self-portrait of Albrecht
%Durer, a German artist. There is a
%magic square in the upper right corner
%of the image.

durerData=load('durer.mat','-mat');
figure('Units','Pixels',...
'Position',[200 200 size(durerData.X,2)*.5 size(durerData

image(durerData.X);
colormap(durerData.map);
axis('image');
set(gca,...
    'Xtick',[],...
    'Ytick',[],...
    'Units','normal',...
    'Position',[0 0 1 1]);

clear durerData
```

☑ Evaluate this expression if there is an error:

```
disp(['Error during Eval: ', evalException.message])
```

**5** In the Outline pane on the left, select the `Eval` component.

**6** In the Library pane in the middle, under the `Handle Graphics` category, double-click `Figure Snapshot`.



Name

- Formatting ----------------------
  - Array-Based Table
  - Chapter/Subsection
  - Image
  - Link
  - List
  - Paragraph
  - Text
  - Title Page
- Formatting Tables ------------...
  - Table
  - Table Body
  - Table Column Spec
  - Table Entry
  - Table Footer
  - Table Header
  - Table Row
- Handle Graphics ------------...
  - Axes Loop
  - Axes Snapshot
  - Figure Loop
  - Figure Snapshot
  - Graphics Object Loop
  - Handle Graphics Linking Anch...
  - Handle Graphics Name

To inline an image component (such as Image or Figure Snapshot), include it within a Paragraph component.

7. In the Properties pane:

   a. In the **Paper** orientation list, select `Portrait`.

   b. In the **Invert hardcopy** list, select `Don't invert`.

      Selecting this option specifies not to change the image's on-screen colors for printing.

   The next three steps set up the report to delete the image from the MATLAB workspace after the image has been added to the report.

8. In the Outline pane, select the `Figure Snapshot` component.

9. In the Library pane, under the `MATLAB` category, double-click `Evaluate MATLAB Expression`.

10. In the Properties pane:

    a. Clear the **Insert MATLAB expression in report** and **Display command window output in report** check boxes. You do not want to include the code or its output in the report.

    b. In the **Expression to evaluate in the base workspace** text box, replace the existing text with the following text:

    ```
    %This command deletes the Durer image
    delete(gcf);
    ```

    The `delete(gcf)` command deletes the current image in the MATLAB workspace, in this case, the Dürer etching.

    c. In the **Evaluate expression if there is an error** text box, replace the existing text with the following text:

    ```
    disp(['Error during eval: ', evalException.message])
    ```

    This code executes if an error occurs while deleting the Dürer etching.

11. Save the report.

## Create the Magic Squares and Their Images

In the next steps, you add a chapter to the report for each magic square specified by the *magicSizeVector* report variable. You use a `For Loop` component to perform this essentially repetitive task. To create the magic squares and their images, you perform these tasks:

- "Create a For Loop" on page 2-23
- "Add a Chapter for Each Square" on page 2-26
- "Determine the Matrix Size" on page 2-27
- "Insert the Magic Square Size into the Report" on page 2-29
- "Create the Magic Square" on page 2-31
- "Add Display Logic" on page 2-34
- "Display the Magic Square" on page 2-37

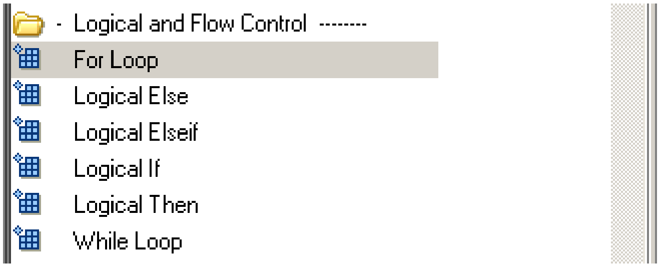
## Create a For Loop

**Note** This section builds on the previous tasks described in the step-by-step example summarized in "Create a MATLAB Report" on page 2-2.

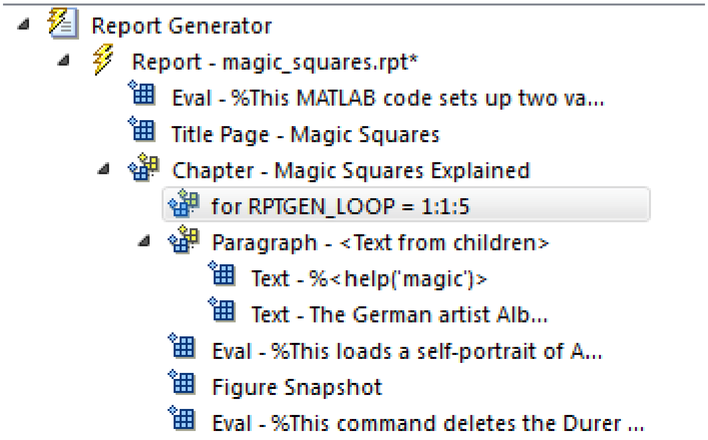
If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

1 In the Outline pane on the left, select the `Chapter` component.

2 In the Library pane in the middle, under the `Logical and Flow Control` category, double-click `For Loop`.



The Outline pane looks as follows.

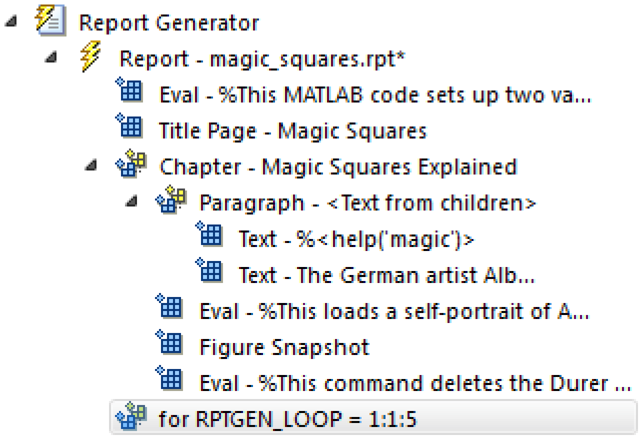


```
Report Generator
    Report - magic_squares.rpt*
        Eval - %This MATLAB code sets up two va...
        Title Page - Magic Squares
        Chapter - Magic Squares Explained
            for RPTGEN_LOOP = 1:1:5
            Paragraph - <Text from children>
                Text - %<help('magic')>
                Text - The German artist Alb...
            Eval - %This loads a self-portrait of A...
            Figure Snapshot
            Eval - %This command deletes the Durer ...
```

This For Loop component appears inside the `Chapter` component. However, the magic squares should be processed *after* the first chapter, so the `for` component should be a sibling of the `Chapter` component, not a child.

**3** In the Outline pane, select the `for` component.

**4** Click the **left** arrow to make the `for` component a sibling, not a child, of the `Chapter` component.



```
Report Generator
    Report - magic_squares.rpt*
        Eval - %This MATLAB code sets up two va...
        Title Page - Magic Squares
        Chapter - Magic Squares Explained
            Paragraph - <Text from children>
                Text - %<help('magic')>
                Text - The German artist Alb...
            Eval - %This loads a self-portrait of A...
            Figure Snapshot
            Eval - %This command deletes the Durer ...
        for RPTGEN_LOOP = 1:1:5
```

**5** In the Properties pane on the right:

**a** In the **End** text box, replace the existing text with the following text:

```
length(magicSizeVector)
```

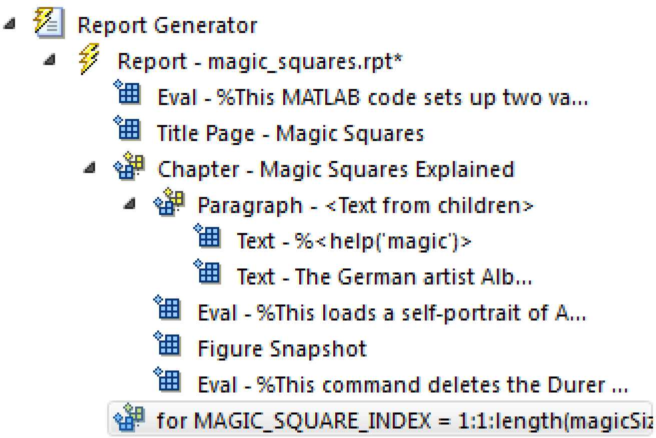
This is the length of the vector that contains the various sizes for the magic square matrices.

**b** In the **Variable name** text box, replace the existing text with the following text:

```
MAGIC_SQUARE_INDEX
```

This variable acts as a loop index.

The Outline pane looks as follows.



```
Report Generator
  Report - magic_squares.rpt*
    Eval - %This MATLAB code sets up two va...
    Title Page - Magic Squares
    Chapter - Magic Squares Explained
      Paragraph - <Text from children>
        Text - %<help('magic')>
        Text - The German artist Alb...
      Eval - %This loads a self-portrait of A...
      Figure Snapshot
      Eval - %This command deletes the Durer ...
    for MAGIC_SQUARE_INDEX = 1:1:length(magicSiz
```

**6** Save the report.

## Add a Chapter for Each Square

**Note** This section builds on the previous tasks described in the step-by-step example summarized in “Create a MATLAB Report” on page 2-2.

If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

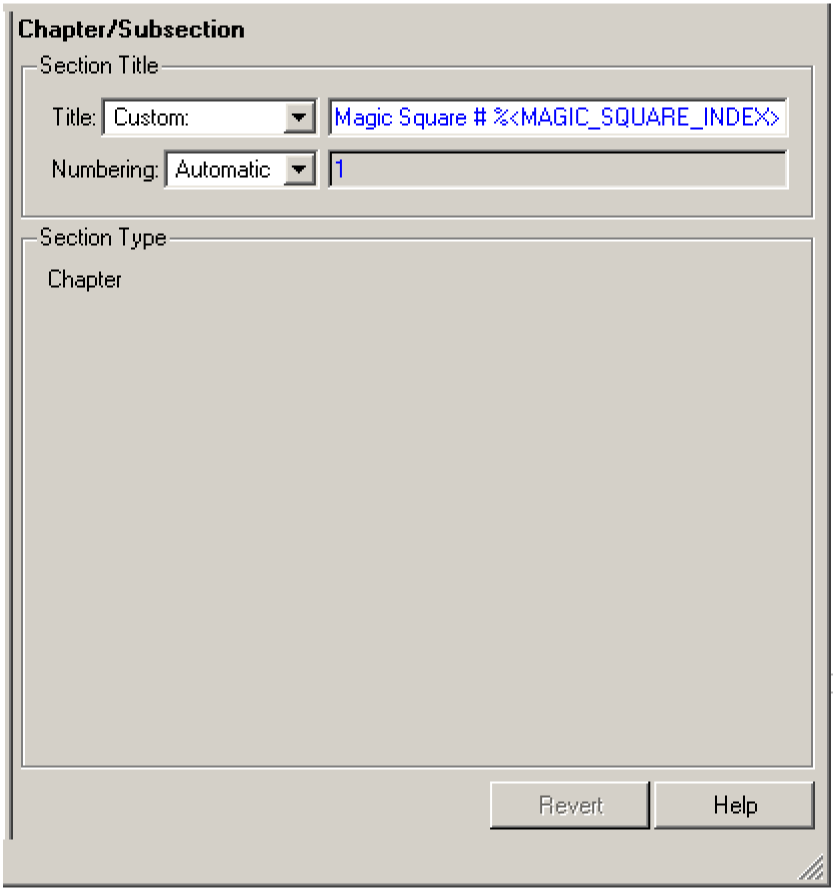
Next create a chapter for each square by adding a Chapter component to the report as a child of the For Loop component. This causes the Report Generator to create a chapter on each iteration of the For Loop during report generation.

1. In the Outline pane on the left, select the `for` component.
2. In the Library pane in the middle, under the `Formatting` category, double-click `Chapter/Subsection`.

   It becomes a child of the `for` component.
3. In the Properties pane on the right, select `Custom` from the **Title** list and enter the following for the chapter title:

   ```
   Magic Square # %<MAGIC_SQUARE_INDEX>
   ```

   The Properties pane looks as follows.



Title: Custom: Magic Square # %<MAGIC_SQUARE_INDEX>

Numbering: Automatic 1

Chapter

**4** Save the report.

## Determine the Matrix Size

**Note** This section builds on the previous tasks described in the step-by-step example summarized in “Create a MATLAB Report” on page 2-2.

If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

Extract the size of each magic square matrix from `magicSizeVector` using an Evaluate MATLAB Expression component.

1 In the Outline pane on the left, select the bottom `Chapter` component.

2 In the Library pane in the middle, under the `MATLAB` category, double-click `Evaluate MATLAB Expression`.

3 In the Properties pane on the right:

   a Clear the **Insert MATLAB expression in report** and **Display command window output in report** check boxes.

   b In the **Expression to evaluate in the base workspace** text box, replace the existing text with the following text:

   ```
   magic_Square_Size=magicSizeVector(MAGIC_SQUARE_INDEX);
   ```

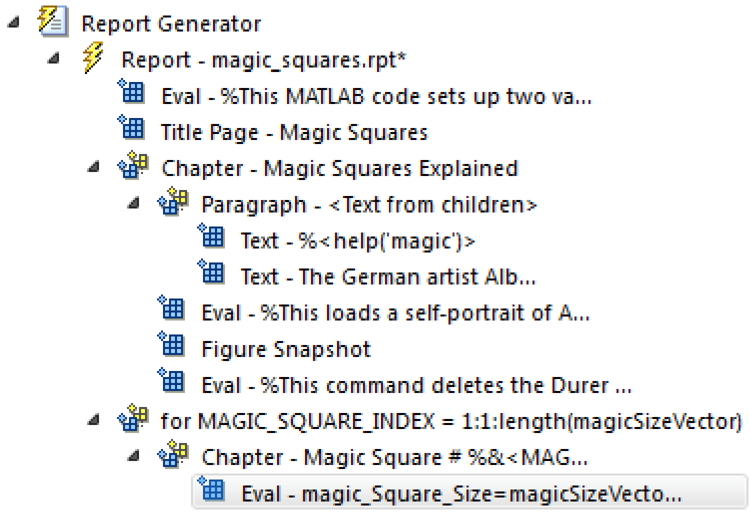
   This command extracts the next size for the magic square from the vector of sizes initialized in the first `Eval` component of the report. The variable `magic_Square_Size` represents the size of the current magic square being processed.

   c In the **Evaluate expression if there is an error** text box, replace the existing text with the following:

   ```
   disp(['Error during eval: ', evalException.message])
   ```

   This code executes if an error occurs while attempting to extract a value from `magicSizeVector`.

The Outline pane looks as follows.



- Report Generator
  - Report - magic_squares.rpt*
    - Eval - %This MATLAB code sets up two va...
    - Title Page - Magic Squares
    - Chapter - Magic Squares Explained
      - Paragraph - <Text from children>
        - Text - %<help('magic')>
        - Text - The German artist Alb...
      - Eval - %This loads a self-portrait of A...
      - Figure Snapshot
      - Eval - %This command deletes the Durer ...
    - for MAGIC_SQUARE_INDEX = 1:1:length(magicSizeVector)
      - Chapter - Magic Square # %&<MAG...
        - Eval - magic_Square_Size=magicSizeVecto...

**4** Save the report.

## Insert the Magic Square Size into the Report

---

**Note** This section builds on the previous tasks described in the step-by-step example summarized in "Create a MATLAB Report" on page 2-2.

If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

---

Insert the size of the magic square into the report using the Paragraph and Insert Variable components.

**1** In the Outline pane on the left, select the bottom `Eval` component.

**2** In the Library pane in the middle, under the `Formatting` category, double-click `Paragraph`.

Do not change the properties. The variable that contains the size of the magic square goes in this paragraph.

3 In the Outline pane, select the `Paragraph` component (below the `for` component).

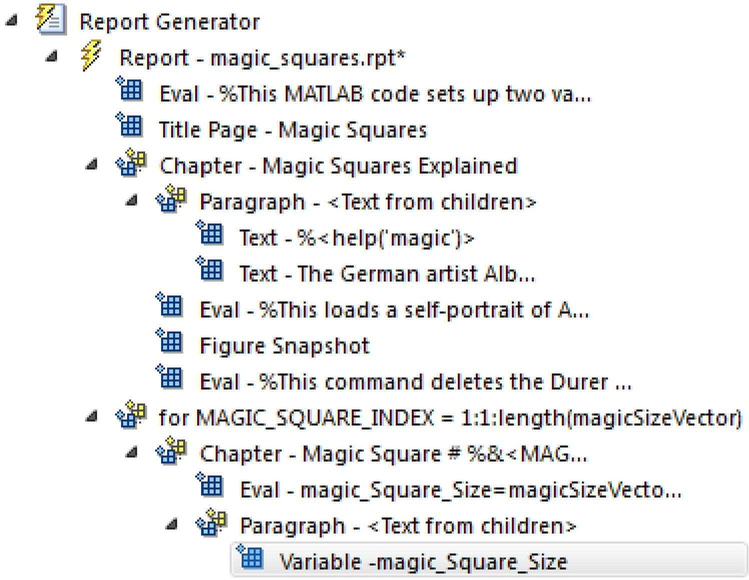
4 In the Library pane, under the `MATLAB` category, double-click `Insert Variable`.

5 In the Properties pane on the right:

a In the **Variable name** text box, enter `magic_Square_Size`.

b In the **Display as** list, select `Inline text`.

The Outline pane looks as follows.



```
Report Generator
  Report - magic_squares.rpt*
    Eval - %This MATLAB code sets up two va...
    Title Page - Magic Squares
    Chapter - Magic Squares Explained
      Paragraph - <Text from children>
        Text - %<help('magic')>
        Text - The German artist Alb...
      Eval - %This loads a self-portrait of A...
      Figure Snapshot
      Eval - %This command deletes the Durer ...
    for MAGIC_SQUARE_INDEX = 1:1:length(magicSizeVector)
      Chapter - Magic Square # %&<MAG...
        Eval - magic_Square_Size=magicSizeVecto...
        Paragraph - <Text from children>
          Variable -magic_Square_Size
```

6 Save the report.

## Create the Magic Square

**Note** This section builds on the previous tasks described in the step-by-step example summarized in "Create a MATLAB Report" on page 2-2.

If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

To create the magic square and display the associated matrix or image, use the Evaluate MATLAB Expression component.

1. In the Outline pane on the left, select the bottom `Paragraph` component.
2. In the Library pane in the middle, under the `MATLAB` category, double-click `Evaluate MATLAB Expression`.

   Make this component a sibling of the `Paragraph` component, not a child, as described in the next two steps.
3. In the Outline pane, select the `Eval` component.
4. Click the left arrow on the toolbar to make the `Eval` component a sibling of the previous `Paragraph` component.
5. In the Properties pane on the right:
   - a Clear the **Insert MATLAB expression in report** and **Display command window output in report** check boxes.
   - b In the **Expression to evaluate in the base workspace** text box, replace the existing text with the following MATLAB code.

     **Tip** Copy and paste this text from the HTML documentation into the Report Explorer.

     ```
     %This MATLAB script produces a magic
     %square of size magic_Square_Size
     %and creates an image of that square.
     ```

```
mySquare=magic(magic_Square_Size);
clf
imagesc(mySquare);
title(sprintf('Magic Square N=%i',magic_Square_Size))
set(gca,'Ydir','normal');
axis equal;
axis tight;
```

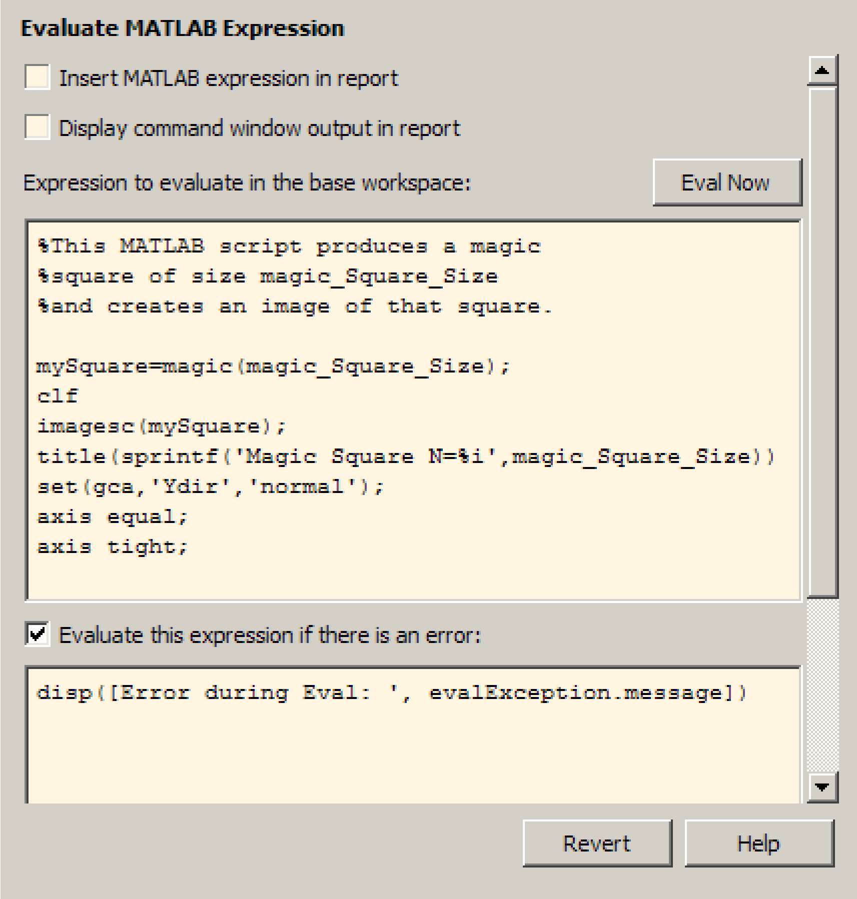
This code creates a magic square matrix `mySquare` of size `magic_Square_Size`, and opens an image of that matrix in the MATLAB figure window.

c In the **Evaluate expression if there is an error** text box, replace the existing text with the following:

```
disp(['Error during eval: ', evalException.message])
```

This code executes if an error occurs while creating and displaying the magic square.

The Properties pane looks as follows.



☐ Insert MATLAB expression in report

☐ Display command window output in report

Expression to evaluate in the base workspace:

```
%This MATLAB script produces a magic
%square of size magic_Square_Size
%and creates an image of that square.

mySquare=magic(magic_Square_Size);
clf
imagesc(mySquare);
title(sprintf('Magic Square N=%i',magic_Square_Size))
set(gca,'Ydir','normal');
axis equal;
axis tight;
```

☑ Evaluate this expression if there is an error:

```
disp([Error during Eval: ', evalException.message])
```

**6** Save the report.

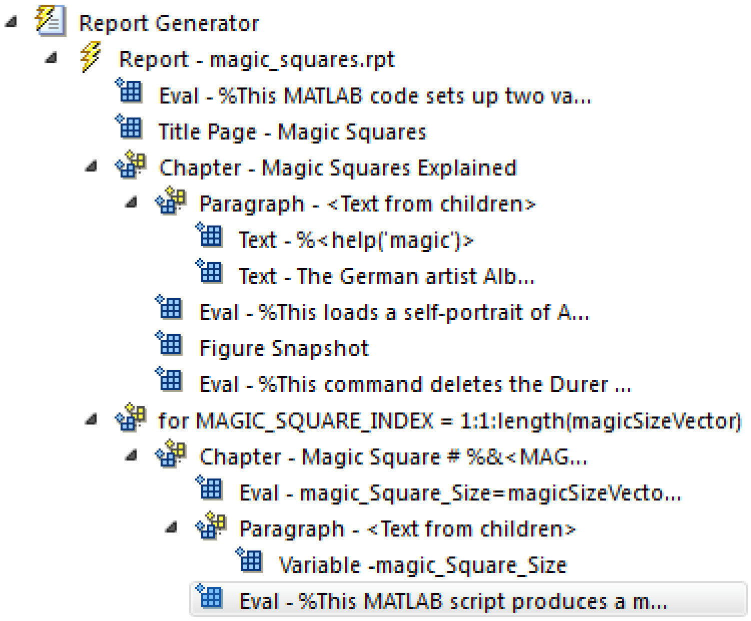
## Add Display Logic

**Note** This section builds on the previous tasks described in the step-by-step example summarized in “Create a MATLAB Report” on page 2-2.

If you have not completed the previous sections of this example, see open the completed report setup file: `Magic Squares Report`.

Use `Logical If`, `Logical Then`, and `Logical Else` components to determine whether to display the magic square as an array of numbers or as an image.

**1** In the Outline pane on the left, select the `Eval` component.



```
Report Generator
  Report - magic_squares.rpt
    Eval - %This MATLAB code sets up two va...
    Title Page - Magic Squares
    Chapter - Magic Squares Explained
      Paragraph - <Text from children>
        Text - %<help('magic')>
        Text - The German artist Alb...
      Eval - %This loads a self-portrait of A...
      Figure Snapshot
      Eval - %This command deletes the Durer ...
    for MAGIC_SQUARE_INDEX = 1:1:length(magicSizeVector)
      Chapter - Magic Square # %&<MAG...
        Eval - magic_Square_Size=magicSizeVecto...
        Paragraph - <Text from children>
          Variable -magic_Square_Size
        Eval - %This MATLAB script produces a m...
```

**2** On the Library pane in the middle, under the `Logical and Flow Control` category, double-click `Logical If`.

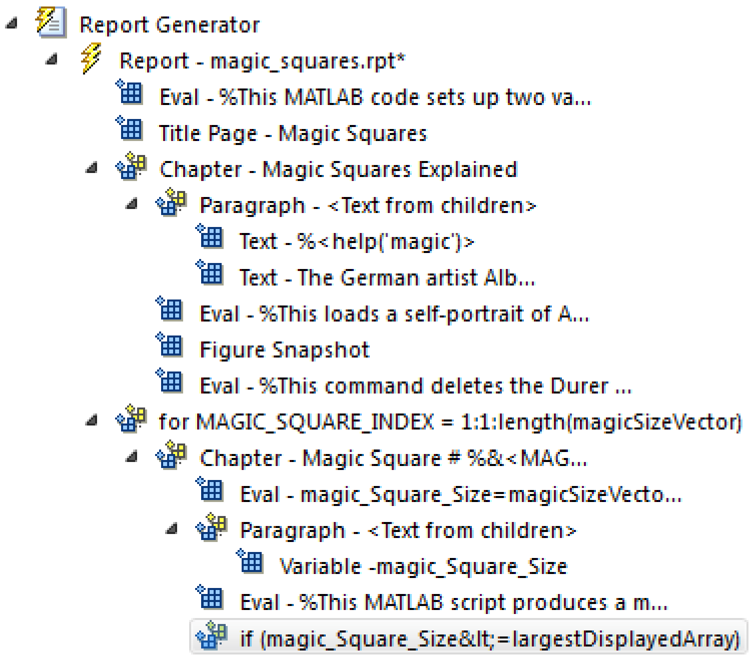
**3** On the Properties pane on the right, in the **Test Expression** text box, replace the existing text with the following text:

```
magic_Square_Size<=largestDisplayedArray
```

This command tests whether the current matrix size (`magic_Square_Size`) is less than or equal to the value assigned in the first `Eval` component of the report (`largestDisplayedArray=15`).

To process the result of this `Logical If` component, create two child components—`Logical Then` and `Logical Else`. If `magic_Square_Size` is less than or equal to 15, the matrix variable appears in the report. If `magic_Square_Size` is greater than 15, the matrix image appears in the report.

**4** On the Outline pane, select the `if` component.



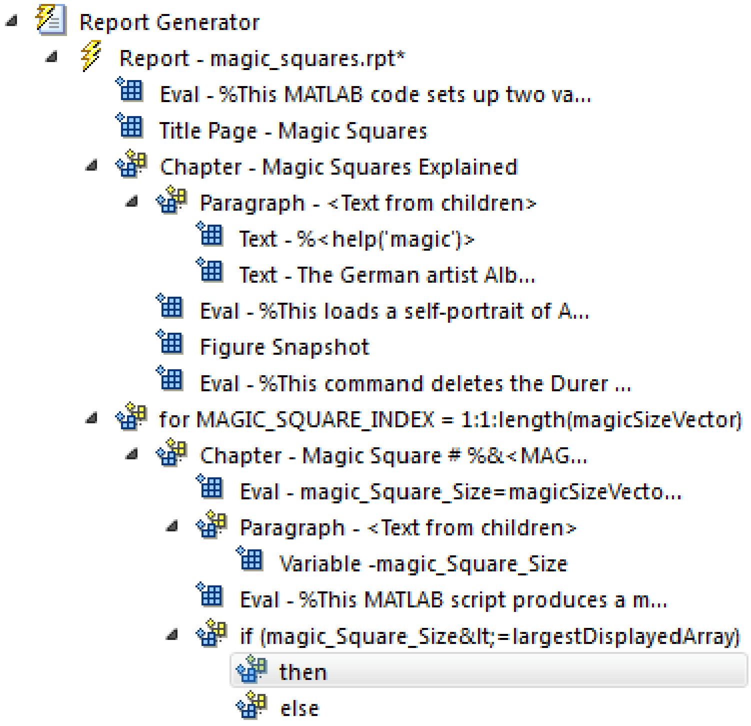
```
Report Generator
  Report - magic_squares.rpt*
    Eval - %This MATLAB code sets up two va...
    Title Page - Magic Squares
    Chapter - Magic Squares Explained
      Paragraph - <Text from children>
        Text - %<help('magic')>
        Text - The German artist Alb...
      Eval - %This loads a self-portrait of A...
      Figure Snapshot
      Eval - %This command deletes the Durer ...
    for MAGIC_SQUARE_INDEX = 1:1:length(magicSizeVector)
      Chapter - Magic Square # %&<MAG...
        Eval - magic_Square_Size=magicSizeVecto...
        Paragraph - <Text from children>
          Variable -magic_Square_Size
        Eval - %This MATLAB script produces a m...
        if (magic_Square_Size&lt;=largestDisplayedArray)
```

**5** On the Library pane, under `Logical and Flow Control`, double-click `Logical Else`.

6 On the Outline pane, select the `if` component again.

7 On the Library pane, under `Logical and Flow Control`, double-click `Logical Then`.

The `then` component appears above the `else` component.



```
Report Generator
  Report - magic_squares.rpt*
    Eval - %This MATLAB code sets up two va...
    Title Page - Magic Squares
    Chapter - Magic Squares Explained
      Paragraph - <Text from children>
        Text - %<help('magic')>
        Text - The German artist Alb...
      Eval - %This loads a self-portrait of A...
      Figure Snapshot
      Eval - %This command deletes the Durer ...
    for MAGIC_SQUARE_INDEX = 1:1:length(magicSizeVector)
      Chapter - Magic Square # %&<MAG...
        Eval - magic_Square_Size=magicSizeVecto...
        Paragraph - <Text from children>
          Variable -magic_Square_Size
        Eval - %This MATLAB script produces a m...
        if (magic_Square_Size&lt;=largestDisplayedArray)
          then
          else
```

8 Save the report.

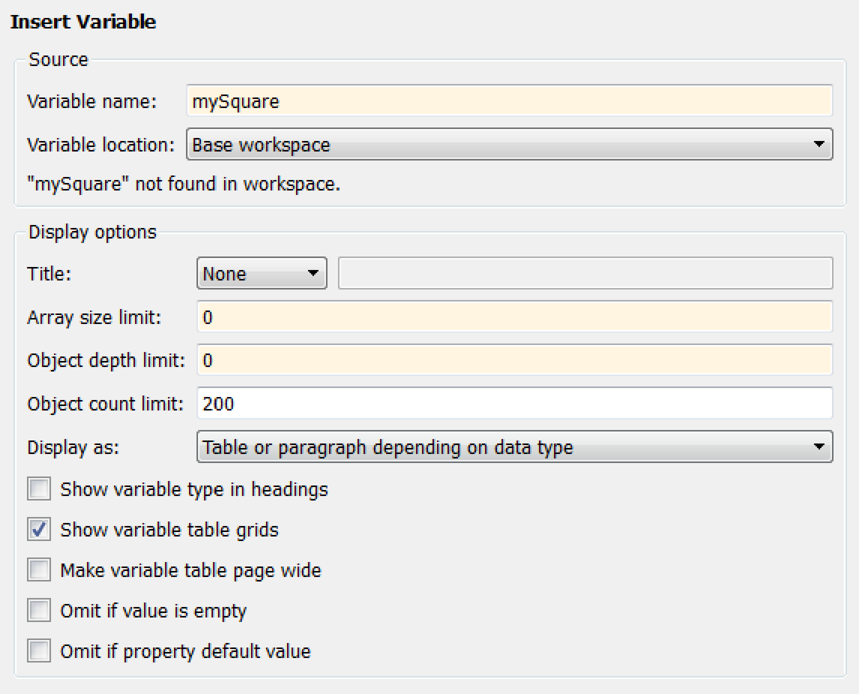
## Display the Magic Square

**Note** This section builds on the step-by-step example presented in "Create a MATLAB Report" on page 2-2.

To see the completed report setup file, open `Magic Squares Report`.

1. In the Outline pane on the left, select the `then` component.
2. In the Library pane in the middle, under the `MATLAB` category, double-click `Insert Variable`.
3. In the Properties pane on the right:
   - a In the **Variable name** text box, enter `mySquare`, which is the variable that contains the magic square of the specified size.
   - b In the **Title** list, select `None`.
   - c In the **Array size limit** text box, enter `0`.

     The Properties pane on the right looks as follows.



Insert Variable

Source

Variable name: mySquare

Variable location: Base workspace

"mySquare" not found in workspace.

Display options

Title: None

Array size limit: 0

Object depth limit: 0

Object count limit: 200

Display as: Table or paragraph depending on data type

☐ Show variable type in headings

☑ Show variable table grids

☐ Make variable table page wide

☐ Omit if value is empty

☐ Omit if property default value

This Variable component displays the magic square matrix, stored in the `mySquare` variable.

4 In the Outline pane, select the `else` component.

5 In the Library pane, under the `Handle Graphics` category, double-click `Figure Loop`.

Do not change its properties.

6 In the Outline pane, select the `Figure Loop` component.

7 In the Library pane, under the `Handle Graphics` category, double-click `Figure Snapshot`.

8 In the Properties pane:

a In the **Paper orientation** list, select `Portrait`.

**b** In the **Image size** list, select `Custom`.

**c** Under the **Image size** list, enter `[5 4]` for the custom image size.

**d** In the **Invert hardcopy** list, select `Invert`.

This option changes dark axes colors to light axes colors, and vice versa.

The Properties pane on the right looks as follows.

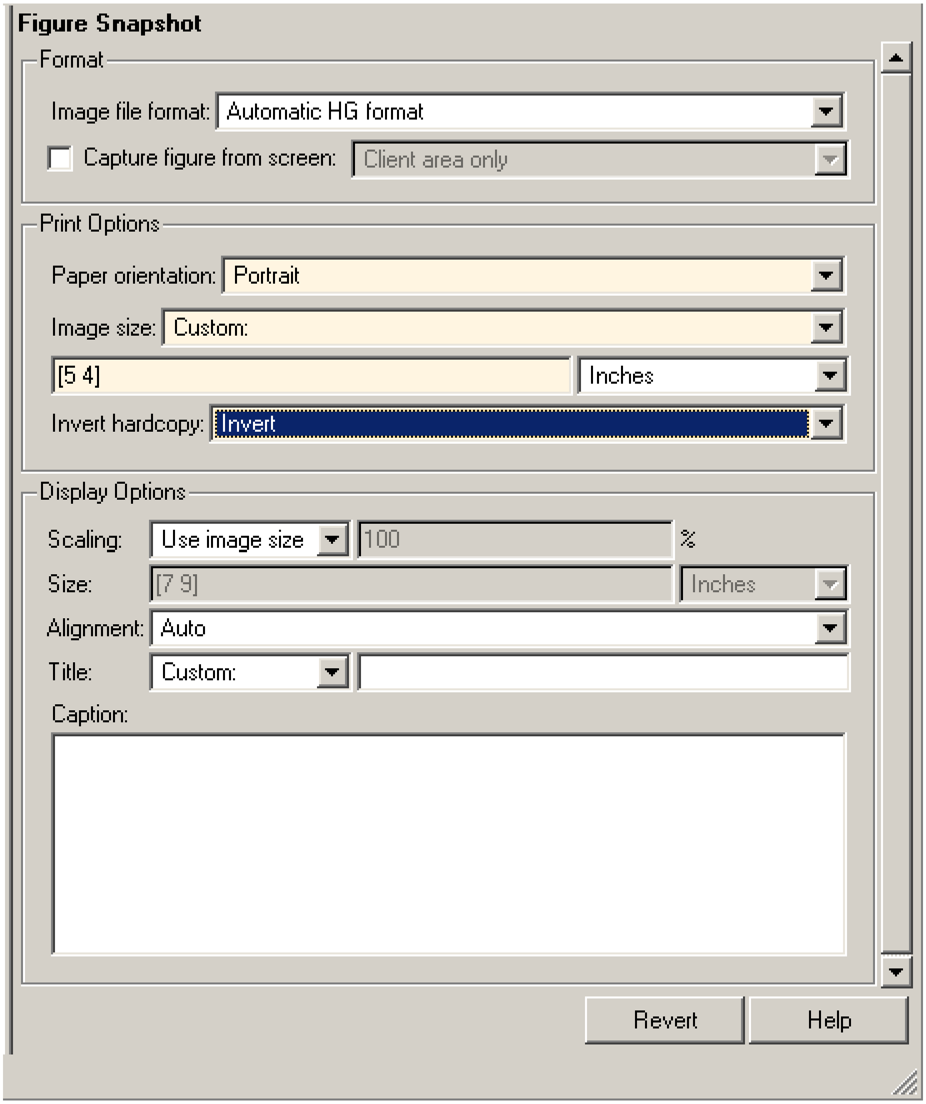


Figure Snapshot

Format

Image file format: Automatic HG format

□ Capture figure from screen: Client area only

Print Options

Paper orientation: Portrait

Image size: Custom:

[5 4] Inches

Invert hardcopy: Invert

Display Options

Scaling: Use image size 100 %

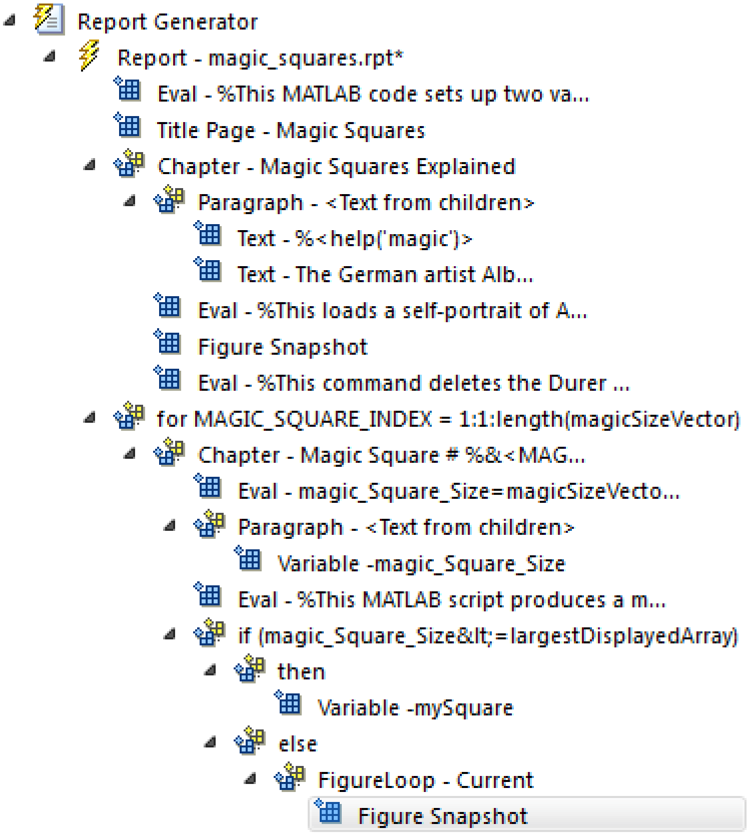
Size: [7 9] Inches

Alignment: Auto

Title: Custom:

Caption:

The Outline pane looks like the following.



```
Report Generator
  Report - magic_squares.rpt*
    Eval - %This MATLAB code sets up two va...
    Title Page - Magic Squares
    Chapter - Magic Squares Explained
      Paragraph - <Text from children>
        Text - %<help('magic')>
        Text - The German artist Alb...
      Eval - %This loads a self-portrait of A...
      Figure Snapshot
      Eval - %This command deletes the Durer ...
    for MAGIC_SQUARE_INDEX = 1:1:length(magicSizeVector)
      Chapter - Magic Square # %&<MAG...
        Eval - magic_Square_Size=magicSizeVecto...
        Paragraph - <Text from children>
          Variable -magic_Square_Size
        Eval - %This MATLAB script produces a m...
        if (magic_Square_Size&lt;=largestDisplayedArray)
          then
            Variable -mySquare
          else
            FigureLoop - Current
              Figure Snapshot
```

**9** Save the report.

# Generate the Report

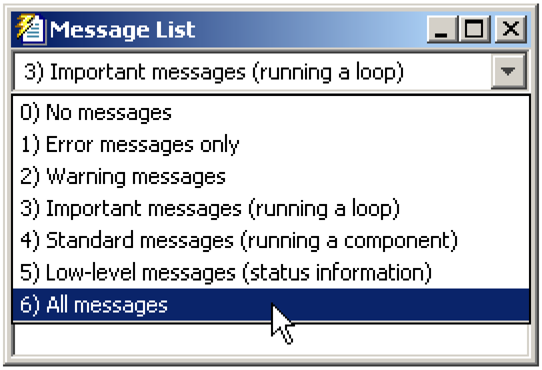
**Note** This section builds on the step-by-step example presented in "Create a MATLAB Report" on page 2-2.

To see the completed report setup file, open `Magic Squares Report`.

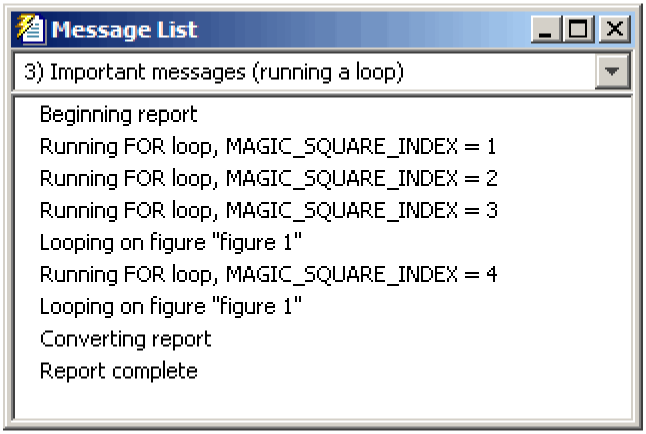
Now the report contains all the components it needs.

On the toolbar, click the **Report** icon to generate the report. The following dialog box appears.

1 A Message List window appears, displaying informational and error messages as the report processes. While the report generates, specify the level of detail you would like the Message List window to display. Options range from 0 (least detail) to 6 (most detail). Click the list located under the title bar of the Message List window to choose an option, as shown here.


Message List
3) Important messages (running a loop)
0) No messages
1) Error messages only
2) Warning messages
3) Important messages (running a loop)
4) Standard messages (running a component)
5) Low-level messages (status information)
6) All messages

Message level 3 (Important messages) is used for the remainder of this example.



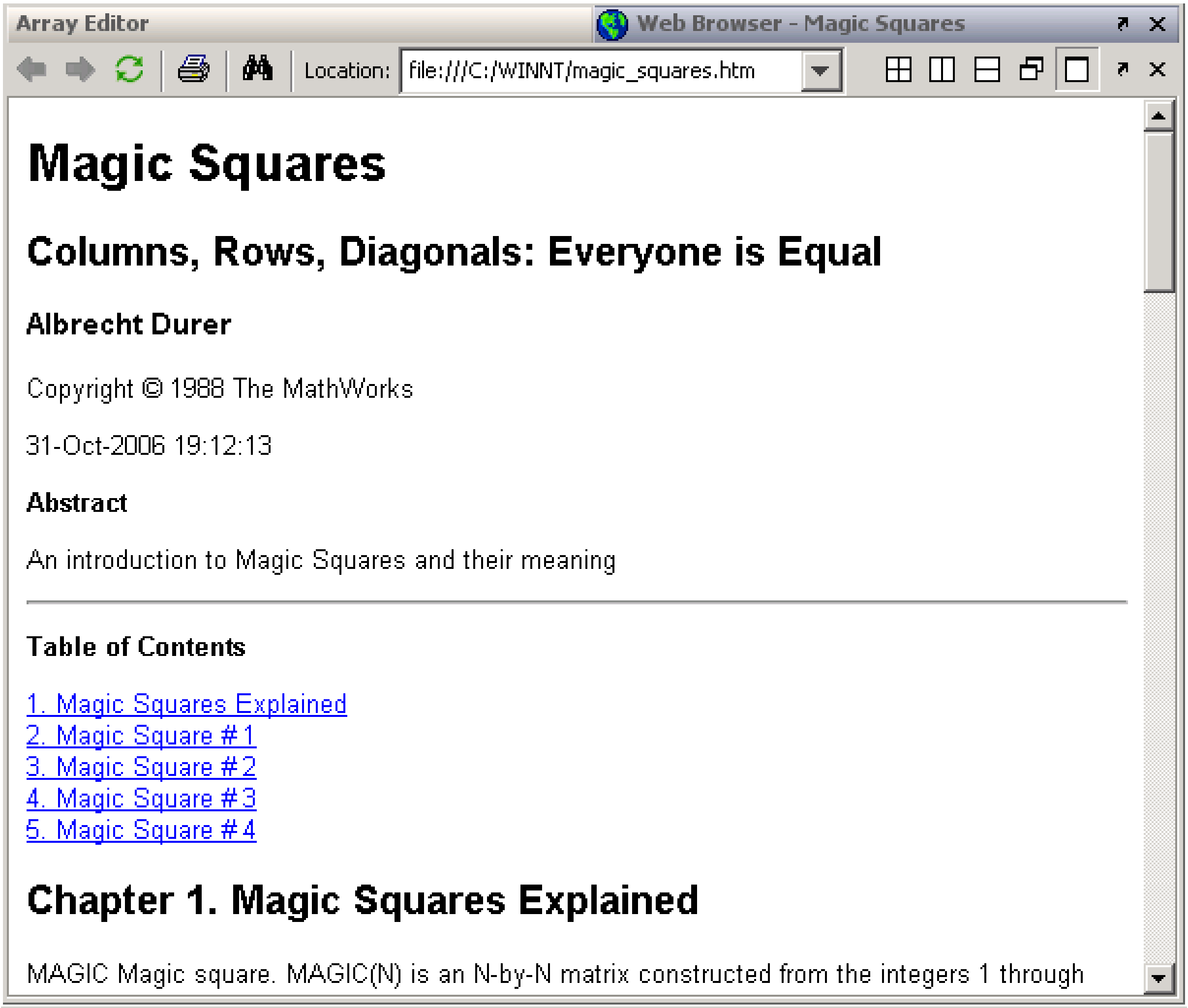
```
Beginning report
Running FOR loop, MAGIC_SQUARE_INDEX = 1
Running FOR loop, MAGIC_SQUARE_INDEX = 2
Running FOR loop, MAGIC_SQUARE_INDEX = 3
Looping on figure "figure 1"
Running FOR loop, MAGIC_SQUARE_INDEX = 4
Looping on figure "figure 1"
Converting report
Report complete
```

2 An image of the etching appears briefly.

3 Images of two magic square images of sizes 16 and 32 appear briefly.

4 In the Outline pane on the left of your Report Explorer window, each component of the report setup file is highlighted as it is executed.

At the beginning of this example you specified HTML as the output format of this report. When processing finishes, the MATLAB Web browser opens and displays the report's HTML file.



Array Editor
Web Browser - Magic Squares
Location: file:///C:/WINNT/magic_squares.htm

Magic Squares

Columns, Rows, Diagonals: Everyone is Equal

Albrecht Durer

Copyright © 1988 The MathWorks

31-Oct-2006 19:12:13

Abstract

An introduction to Magic Squares and their meaning

Table of Contents

Chapter 1. Magic Squares Explained

MAGIC Magic square. MAGIC(N) is an N-by-N matrix constructed from the integers 1 through

# Set Up a Report

# Report Setups

**In this section...**

A *report setup* is a set of MATLAB objects, called *components*, that specifies the content and form of a report.

The MATLAB Report Generator provides a setup editor, called the Report Explorer, that you use to create and edit report setups.

Once you create a setup, you can generate a report from it, using the Report Explorer or MATLAB commands.

## Setup Hierarchy

A report setup has a hierarchical structure that generally mirrors the structure of the type of report that it defines.

For example, a report typically has a title page and one or more chapters. Each chapter contains one or more sections, each of which contains one or more paragraphs, figures, tables, lists, etc. A report setup typically comprises components that correspond to these structural elements of a report.

In a report setup, child-parent relationships among the components correspond to the containment relationships among the structural elements of the report. In particular, all setups contain a *root component* that serves as the ancestor for all other components in the setup. The root component also specifies the setup name and report generation options, such as the document type of the generated report (for example, HTML or PDF) and the path for the generated report. The root component typically parents a title page component and one or more chapter components that in turn parent one or more section components that parent one or more paragraph, figure, table, and list components.

## Setup Files

The report generator stores setups in files called *setup files*. The name of a setup file consists of the name of the setup that it stores followed by the file extension `.rpt`. For example, the name of the setup file for a setup named `myreport` would be `myreport.rpt`.

## Create a Report Setup

To create a report setup:

- Create a new setup file.
- Insert components to define the content and format of the report.
- Set component properties.
- Save the setup.

Once you create a template, you can execute it to generate an instance of the type of report that it defines.

# Create a New Setup File

**In this section...**

Create a new setup file either interactively from the Report Explorer or programmatically.

## Create Setup File Using the Report Explorer

1 If the Report Explorer is not already open, from the MATLAB Toolstrip, in the **Apps** tab, in the **Database Connectivity and Reporting** section, click **Report Generator**.

2 In the Report Explorer, use *one* of these approaches:

- Select**File > New**.
- On the Report Explorer toolbar, click the new template button

The Outline pane displays a new setup named `Unnamed`, as a child of the `Report Generator` node.

## Create Setup File Programmatically

To create a setup file programmatically (from the MATLAB command line), use the `setedit` command. For example, assuming a setup named `myreport` does not already exist in the current directory, use the following command:

```
setedit myreport.rpt
```

## Working with Setup Files

For details about performing operations on report setup files, see:

- "Open a Report Setup" on page 3-6
- "Close a Report Setup" on page 3-9

- "Save a Report Setup" on page 3-10

# Open a Report Setup

**In this section...**

To make changes to a saved report setup, you must open its setup file. Open a report setup either interactively from the Report Explorer or programmatically.

## Opening a Setup on the MATLAB Path

**Tip** Use the `setedit` command to obtain a list of all the report setups on the MATLAB path.

To open a setup that resides on the MATLAB path:

1. If the Report Explorer is not already open, from the MATLAB Toolstrip, in the **Apps** tab, in the **Database Connectivity and Reporting** section, click **Report Generator**.

2. In the Report Explorer, in the Outline pane on the left, select the `Report Generator` node.

   The Library pane in the middle displays a list of all the setup files that exist on the MATLAB path.

3. In the Library pane, select the setup file that you want to open.

   The setup properties dialog box appears in the Properties pane on the right.

4. To open the setup, in the Report Explorer use *one* of these approaches:

   - On the Properties pane, click the **Open report** button.
   - On the Library pane, double-click the entry for the setup.

- On the Library pane, from the context menu for the setup, select **Open report**.

The setup appears in the Outline pane as a child of the `Report Generator` node.

## Opening a Setup Not on the MATLAB Path

**Tip** Use the `setedit` command to obtain a list of all the report setups on the MATLAB path.

To open a setup that resides off the MATLAB path:

1. If the Report Explorer is not already open, from the MATLAB Toolstrip, in the **Apps** tab, in the **Database Connectivity and Reporting** section, click **Report Generator**..

2. In the Report Explorer, select **File > Open** or select the file open button on the Report Explorer toolbar.

   A file browser opens.

3. Use the file browser to find the report setup in your file system and enter the setup name in the file browser **File name** field.

4. Select the file browser **Open** button.

The setup appears in the Outline pane as a child of the `Report Generator` node.

## Opening a Setup Programmatically

To open a report programmatically, use the `setedit` command. For example, the following command opens the `simple-report.rpt` example that comes with the MATLAB Report Generator.

```
setedit simple-report
```

This command opens the Report Explorer, if it is not already open, and opens the `simple-rpt` setup in the Report Explorer.

**Tip** If a setup exists on the MATLAB path, you do not need to specify its full path when using the `setedit` command. Use the `setedit` command to obtain a list of all the report setups on the MATLAB path.

The newly opened report appears in the Outline pane as a child of the `Report Generator` node.

# Close a Report Setup

**In this section...**

Closing a setup removes the setup from the Report Explorer and from memory.

## Close a Setup Using the Report Explorer

1 In the Report Explorer, in the Outline pane, select the setup root node.

2 In Report Explorer, use *one* of these approaches:

- Click the **Delete** button.
- Select **File >  Close**.
- From the context menu of the root node of the setup file, select **Close**.

## Close a Setup Programmatically

You can close a report that you have previously opened. For example, the following code opens a setup and then closes it.

```
setup('simple-report.rpt');
root = RptgenML.Root;
root.closeReport('simple-report');
```

# Save a Report Setup

**In this section...**

## Save a Setup Under Its Existing Name

1. In the Report Explorer, in the Outline pane, select the setup root node.
2. In Report Explorer, use *one* of these approaches:
   - Click the **Save** button.
   - Select **File > Save**.
   - From the context menu of the root node of the setup file, select **Save**.

## Save a Setup Under a New Name

1. In the Report Explorer, in the Outline pane, select the setup root node.
2. Select **File > Save As**.

   A file browser opens.
3. Use the file browser to select a new path for the setup.
4. In the file browser, click **Save**.

# Load Report Setup into MATLAB Workspace

To load a setup into the MATLAB workspace without loading it into the Report Explorer, use the `rptgen.loadRpt` function.

You can then modify the setup programmatically. For example, the following code loads a setup into memory, sets its output type to PDF, and generates a report.

```
setupRoot = rptgen.loadRpt('simple-report');
setupRoot.Format = 'pdf-fop';
setupRoot.execute;
```

# Insert Components

**In this section...**

## Point-and-Click Method

1. In the Report Explorer, in the Outline pane, select the parent node of the component to be inserted. For example, if you are inserting a paragraph into a section, select the section that will contain the paragraph.

2. In the Library pane, select the type of component that you want to insert in the report setup.

3. In the Properties pane, select the **Add component to current report** button.

## Drag-and-Drop Method

1. In the Report Explorer, in the Library pane, select the type of component that you want to insert in the setup.

2. Drag the component from the Library pane into the Outline pane and drop it onto the parent of the component to be created.

## Fix Context Violations

The Report Explorer allows you to insert components into invalid contexts.

For example, a Chapter/Subsection component is a valid parent for a Paragraph component, but not vice-versa. Nevertheless, the Report Explorer allows you to insert a Chapter/Subsection as a child of a Paragraph. If you insert a component in an invalid context, the Report Explorer displays a warning.

Although you can create an invalid setup hierarchy, you cannot generate a report from an invalid hierarchy. You must fix the context violations first. For example, move components from invalid contexts to valid contexts (see "Move Components" on page 3-15).

# Set Component Properties

## Edit Component Property Values

Most components have properties that you can set to select optional features. For example, the Text component lets you specify the color of the text that it generates among other properties.

To set component properties:

1. In the Report Explorer, in the Outline pane, select the component.

   The Properties dialog box for the component appears in the Properties pane.

2. Use the Properties dialog box to set component properties.

## Computed Property Values

During report generation, the Report Generator can compute the values of component properties, using MATLAB expressions that you specify. This enables dynamic creation of report content. For example, you can use MATLAB expressions to compute the content of Paragraph components and the value of looping components that generate repeated content.

You can use MATLAB expressions to compute the value of any string property of a component. A string property is a property whose value is a string of text. To specify a MATLAB expression as a string property value, in the Properties dialog box, in the property edit box, enter `%<expr>`, where `expr` is a MATLAB expression that evaluates to a string.

# Move Components

| In this section... |
|---|
| "Point-and-Click Method" on page 3-15 |
| "Drag-and-Drop Method" on page 3-16 |

## Point-and-Click Method

1. In the Report Explorer, in the Outline pane, select the component that you want to move.

2. Reposition the component in the setup hierarchy, using *one* of these approaches:

   - On the Report Explorer toolbar, use the move buttons.
   - From the **Edit** menu, use the move commands.
   - From the context menu of the component, use the move commands.

**Note** The move buttons and commands are enabled only if they are valid in the context of the component to be moved. For example, if a component cannot move further to the right in the hierarchy, the **Move Right** button is disabled.

The following table summarizes the available move buttons and commands.

| Move Command or Button | Effect |
|---|---|
| **Move Up** | Moves a component ahead of the sibling that formerly preceded it in the hierarchy. If the component is the first child of its parent, the component becomes a sibling of its former parent. |
| **Move Down** | Moves a component after the sibling that formerly followed it in the hierarchy. If a component is the last sibling of its parent, it moves up one level in the hierarchy to become a sibling of its former parent. |
| **Move Left** | Moves a component up one level in the hierarchy. The component becomes a sibling of its former parent. |
| **Move Right** | Moves a component down one level in the hierarchy. The component becomes the child of the sibling that formerly preceded it in the hierarchy. |

## Drag-and-Drop Method

1. In the Report Explorer, in the Outline pane, select the component that you want to move.

2. Drag the component and drop it on the component that you want to be its parent.

# Delete Components

1. In the Report Explorer, in the Outline pane, select the component that you want to delete.

2. Delete the component, using *one* of these approaches:

   - On the Report Explorer toolbar, click the **Delete** button.
   - Select **Edit >  Delete**.
   - From the context menu of the component, select **Delete**.

# Deactivate Components

You can deactivate any component in a report setup. Deactivating a component causes it to be skipped during generation of a report.

Deactivating components can be useful for debugging setups. For example, you can deactivate a component that you suspect is causing an error or you can activate only the components you want to debug, thereby cutting the time required to verify a fix.

To deactivate (or reactivate) a component:

1. In the Report Explorer, in the Outline pane, select the component that you want to deactivate (or reactivate).

2. Select the appropriate **Activate/Deactivate Component** option from either the **Edit** menu or from the context menu of the component.

# Send Components to the MATLAB Workspace

You can send the components of a setup from the Report Explorer to the MATLAB workspace. This allows you to inspect and set their properties at the MATLAB command line.

Sending components to the workspace can be useful for creating or debugging MATLAB programs that create report setups and generate reports from them.

To send a component to the MATLAB workspace:

1. In the Report Explorer, in the Outline pane, select the component that you want to send to the workspace.
2. From the context menu of the component, select **Send to Workspace**.

# Generate a Report

# Generate a Report

**In this section...**

## Run a Report

You can generate a MATLAB Report Generator report using one of these methods:

You can generate a Simulink Report Generator report using one of these methods:

- In the Report Explorer Outline pane, select a report and do one of the following actions:
  - In the Report Explorer toolbar, click the Report button (⚡).
  - Press **CTL+R**.
  - Select **File > Report**.
- From the MATLAB command line, use the `report` command. For example, to print the `system1_description` report in PDF format, use:

  ```
  report system1_description -fpdf
  ```

## Report Output Options

Before you generate a report, you can set options to control aspects of report generation processing such as:

- Output file format (PDF, HTML, or Microsoft Word)
- Stylesheet for the selected output file format, to control the layout of the report (for example, whether to display a title page, font, and section numbering)
- Output file location
- Whether to view the generated report automatically

For details, see:

- “Report Output Format and Stylesheet” on page 4-5
- “Output File Location” on page 4-11
- “Create a Log File” on page 4-22
- “Report Description” on page 4-13
- “Change Report Locale” on page 4-18

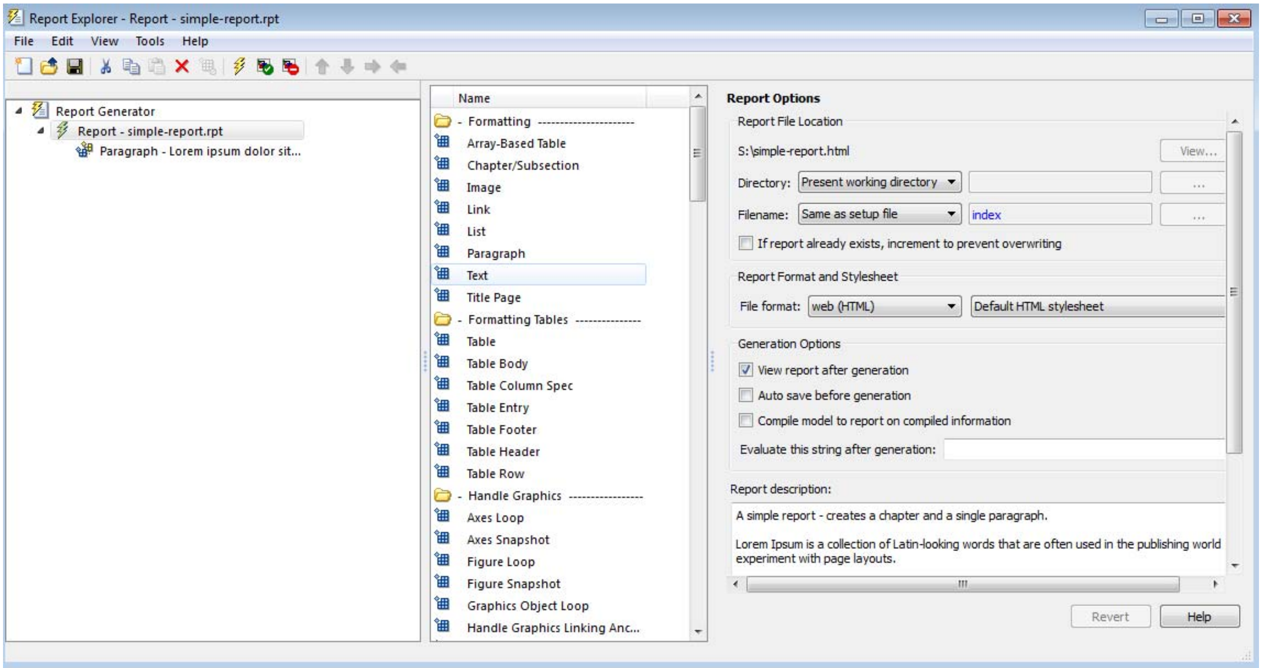
# Report Generation Options

**In this section...**

## Report Options Dialog Box

To specify report generation options for a specific report, in the Report Explorer, use the Report Options dialog box.

The Report Options dialog box in the Properties (right hand) pane of the Report Explorer.



Report Options

Report File Location

S:\simple-report.html

Directory: Present working directory

Filename: Same as setup file index

□ If report already exists, increment to prevent overwriting

Report Format and Stylesheet

File format: web (HTML) Default HTML stylesheet

Generation Options

☑ View report after generation

□ Auto save before generation

□ Compile model to report on compiled information

Evaluate this string after generation:

Report description:

A simple report - creates a chapter and a single paragraph.

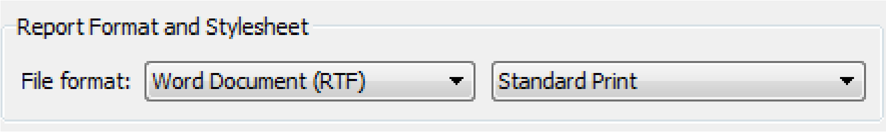
Lorem Ipsum is a collection of Latin-looking words that are often used in the publishing world experiment with page layouts.

To set defaults for report generation options that you can override with the Report Options dialog box or with individual components, use the Report Generator Preferences pane. For details, see “Report Generation Preferences” on page 4-14.

## Report Output Format and Stylesheet

In the Report Explorer, in the **File format** text box, choose the report output format . For example, if you want to use Microsoft Word, choose `Word Document (RTF)`.

Each output format has a default stylesheet associated with it. Specify the stylesheet in the text box next to the **File format** text box.


Report Format and Stylesheet

File format: Word Document (RTF) Standard Print

The following table shows which output format to use with different software, and which stylesheets you can use with each output format.

| Viewer | Format | Description | Stylesheets |
|---|---|---|---|
| Adobe Acrobat Reader | Adobe Acrobat (PDF) | Produce a PDF that you can view using Adobe Acrobat Reader software.<br>See "PDF: Image Formats" on page 4-7. | PDF (see "PDF Stylesheets" on page 4-8) |
| Web browser | `Web (HTML)` (default) | Use for publishing on the World Wide Web. | Web (see "Web Stylesheets" on page 4-9) |
| Word processor | `Word Document (RTF)` or `Rich Text Format (RTF)` | Produce output that is compatible with most word-processing packages, including Microsoft Word software<br>See "RTF: Display of Hidden Content" on page 4-7. | Print (see "RTF (DSSSL Print) and Word Stylesheets" on page 4-10) |
| DocBook | `DocBook (XML)` | Produce a report in DocBook format | N/A |

**Note** For reports that use `Word Document (RTF)` format, you must have Microsoft Word installed on the machine that you use to generate the report.

**Tip** To create and use customized styles, see “Create a New Stylesheet” on page 7-5.

### PDF: Image Formats

PDF reports only support bitmap (`.bmp`), jpeg (`.jpg`), and Scalable Vector Graphics (`.svg`).

The SVG format is only supported for Simulink models and Stateflow charts. For example, MATLAB figures do not display in SVG when you select the SVG format for PDF reports.

### RTF: Display of Hidden Content

RTF reports use placeholders (field codes) for dynamically generated content, such as page numbers or images.

On Windows platforms, to display that content, press **Ctrl-A**, and then press **F9**.

On Linux and Mac platforms, use the field code update interface for the program that you are using to view the RTF document.

### Change the Default Output Format

In the Report Generator Preferences pane, use the **Format ID** preference to specify the default output format for reports.

### Stylesheets

For each output format, you can choose from several stylesheets for each report output format. For details, see:

- “PDF Stylesheets” on page 4-8
- “Web Stylesheets” on page 4-9
- “RTF (DSSSL Print) and Word Stylesheets” on page 4-10

**Note** Some Web and Print stylesheets include an automatically generated list of titles, which includes table titles and figures with titles.

## PDF Stylesheets

| PDF Stylesheet | Description |
|---|---|
| `Default print stylesheet` | Displays title page, table of contents, list of titles |
| `Standard Print` | Displays title page, table of contents, list of titles |
| `Simple Print` | Suppresses title page, table of contents, list of titles |
| `Compact Simple Print` | Minimizes page count, suppresses title, table of contents, list of titles |
| `Large Type Print` | Uses 12-point font (slightly larger than Standard Print) |
| `Very Large Type Print` | Uses 24-point font and landscape paper orientation |
| `Compact Print` | Minimizes white space to reduce page count |
| `Unnumbered Chapters & Sections` | Uses unnumbered chapters and sections |
| `Numbered Chapters & Sections` | Numbers chapters and sections |
| `Paginated Sections` | Prints sections with page breaks |
| `Custom Header` | Lets you specify custom headers and footers |
| `Custom Titlepage` | Lets you specify custom title page content and presentation |
| `Verbose Print` | Lets you specify advanced print options |

## Web Stylesheets

| Web Stylesheet | Description |
|---|---|
| `Default HTML stylesheet` | HTML on a single page |
| `Simulink book HTML stylesheet` | HTML on multiple pages; suppresses chapter headings and table of contents |
| `Truth Table HTML stylesheet` | HTML on multiple pages; suppresses chapter headings and table of contents |
| `Multi-page Web` | HTML, with each chapter on a separate page |
| `Single-page Web` | HTML on a single page |
| `Single-page Unnumbered Chapters & Sections` | HTML on a single page; chapters and sections are not numbered |
| `Single-page Numbered Chapters & Sections` | HTML on a single page; chapters and sections are numbered |
| `Single-page Simple` | HTML on a single page; suppresses title page and table of contents |
| `Multi-page Simple` | HTML on multiple pages; suppresses title page and table of contents |
| `Multi-page Unnumbered Chapters & Sections` | HTML on multiple pages; chapters and sections are not numbered |
| `Multi-page Numbered Chapters & Sections` | HTML on multiple pages; chapters and sections are numbered |

## RTF (DSSSL Print) and Word Stylesheets

| RTF or Word Stylesheet | Description |
|---|---|
| `Standard Print` | Displays title page, table of contents, list of titles |
| `Simple Print` | Suppresses title page, table of contents, list of titles |
| `Compact Simple Print` | Minimizes page count, suppresses title, table of contents, list of titles |
| `Large Type Print` | Uses 12-point font (slightly larger than Standard Print) |
| `Very Large Type Print` | Uses 24-point font and landscape paper orientation |
| `Compact Print` | Minimizes white space to reduce page count |
| `Unnumbered Chapters & Sections` | Uses unnumbered chapters and sections |
| `Numbered Chapters & Sections` | Numbers chapters and sections |

## Report Generation Processing

The Report Options dialog box includes several options for controlling report processing.

| Option | Purpose |
|---|---|
| **View report after generation** | View the report automatically. When report generation finishes, the viewer associated with the report output format displays the report.<br>To view the report manually, browse to the location specified in the **Report File Location** section in the Properties pane on the right, and open the file. |
| **Auto save before generation** | Automatically save the report setup file before you generate a report. |

| Option | Purpose |
|---|---|
| **Compile model to report on compiled information** | Ensure that a report reflects compiled values.<br>By default, the Simulink Report Generator reports uncompiled values of Simulink parameters. The uncompiled values of some parameters, such as signal data types, can differ from the compiled values used during simulation.<br>This option causes the report generator to compile a model before reporting on model parameters. After generating the report, the report generator returns the model to its uncompiled state.<br>**Note** When you select this option, whenever report generation requires simulating the model (for example, the report includes a Model Simulation component), the report generator uncompiles the model and then recompiles the model, if necessary, to report on model contents. If a report requires multiple compilations, the processing can be quite time-consuming.<br>To minimize compilations, consider using separate reports to report on the contents of a model and on the results of simulating that model. |
| **Evaluate this string after generation** | Specify MATLAB code for processing to occur after the report is generated. For example, you could specify to close a model. |

## Output File Location

Choose a folder to store the report file. You must have write privileges for that folder.

## Folder

In the Report Explorer, in the Report Options dialog box, use the **Directory** field to specify the name of the folder in which to store the generated report file. Specify a folder to which you have write privileges.

The following table summarizes the report file location options.

| Folder | Option |
|---|---|
| The same folder as the report setup file | `Same as setup file` |
| The current working folder | `Present working directory` |
| Temporary folder | `Temporary directory` |
| Another folder | `Custom`.<br>Use the **Browse** button (...) to select from a list of directories. |

You can use `%<VariableName>` notation to specify a folder in the **Custom** text box. For more information, see "%<VariableName> Notation" on page 9-90 on the `Text` component reference page.

## Report File Name

In the Report Explorer, in the Report Options dialog box, use the **Filename** field to specify a file name for the report file. Select one of the following options.

| File Name | Option |
|---|---|
| The same file name as the report setup file | `Same as setup file` (default) |
| A file name different from the report setup file name | `Custom`.<br>Enter the name of the report. |

You can use `%<VariableName>` notation to specify a file name in the **Custom** text box. For more information, see “%<VariableName> Notation” on page 9-90 on the `Text` component reference page.

### Increment to Prevent Overwriting

To maintain the previous version of the setup file when you save updates to the setup file, select **If report already exists, increment to prevent overwriting**.

### Image Output File Location

Images are placed in a folder with the same name as the report file. For example, `testreport.html` images are placed in a folder named `testreport_files`.

## Report Description

To record notes and comments about your report setup, use the **Report Description** field. This text that you enter appears in the Properties pane when you select a report setup file in the Outline pane.

# Report Generation Preferences

**In this section...**

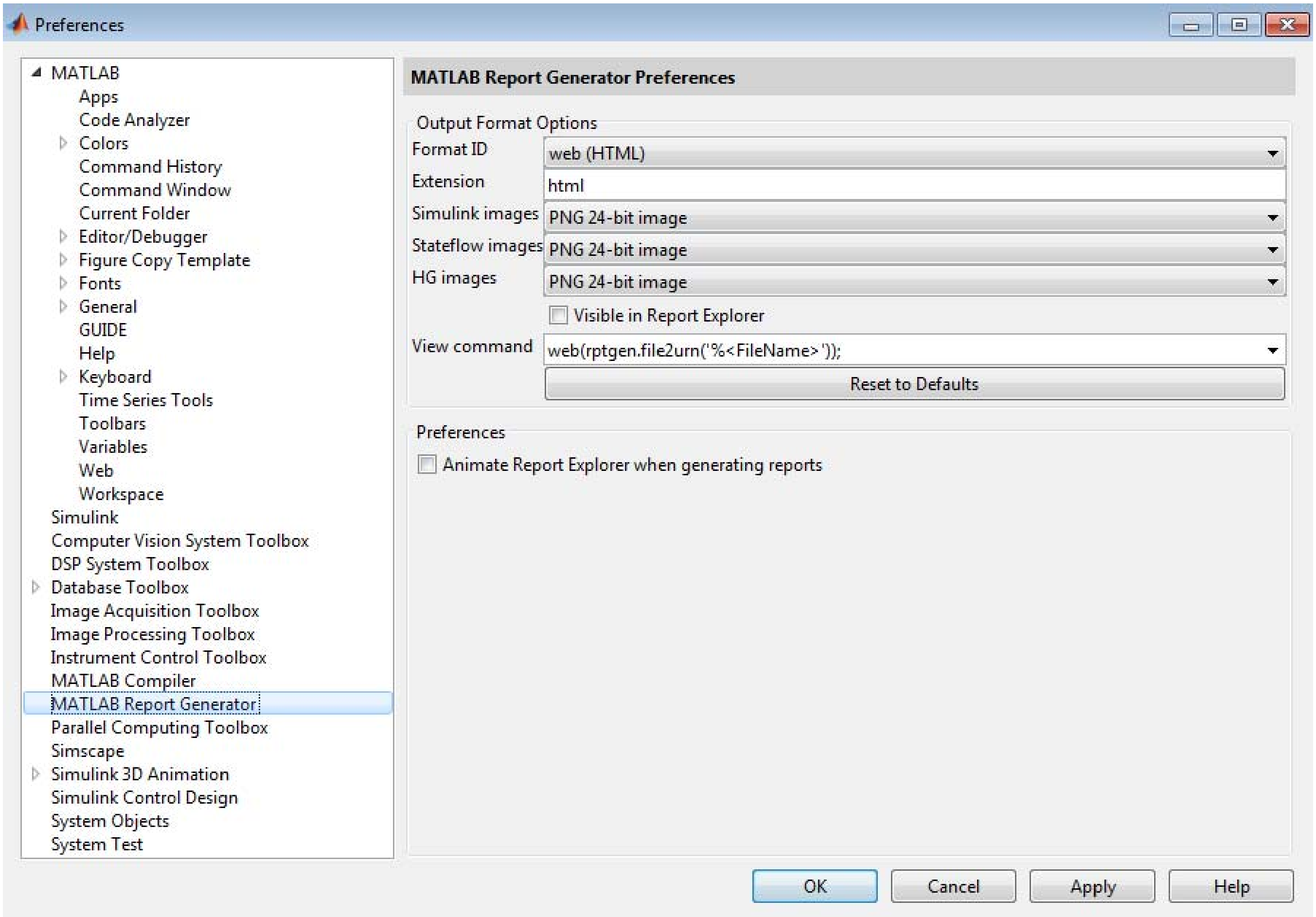
## Report Generator Preferences Pane

To set defaults for report generation options, use the Report Generator Preferences pane. You can override these preferences with the Report Options dialog box or with individual components.

To specify report generation options for a specific report, in the Report Explorer, use the Report Options dialog box. For details, see "Report Generation Options" on page 4-4.

To open the Report Generator Preferences pane, use *one* of these approaches:

- In the Report Explorer, select **File > Preferences**.
- From the MATLAB Toolstrip, in the **Home** tab, in the **Environment** section, select **Preferences > Report Generator**.



MATLAB Report Generator Preferences

Output Format Options

Format ID: web (HTML)

Extension: html

Simulink images: PNG 24-bit image

Stateflow images: PNG 24-bit image

HG images: PNG 24-bit image

☐ Visible in Report Explorer

View command: web(rptgen.file2urn('%<FileName>'));

Preferences

☐ Animate Report Explorer when generating reports

## File Format and Extension

To specify the default file format for reports, use the **Format ID** preference. The default preference is `web (HTML)`. You can select from a range of file formats, such as PDF, Microsoft Word, or LaTex.

**Note** For reports that use the Word Document format, you must have Microsoft Word installed on the machine that you use to generate the report.

The **Extension** preference reflects the standard file extension for the file format specified with the **Format ID** preference. You can change the extension.

## Image Formats

To set the default image formats associated with the output format for a report, use the following preferences.

| Preference | Purpose |
|---|---|
| `Simulink Images` | Specify the format for Simulink images to include in the report. |
| `Stateflow Images` | Specify the format for Stateflow charts to include in the report. |
| `HG Images` | Specify the format for Handle Graphics images to include in the report. |

**Note** The default preferences for image formats should work in most viewing environments. However, some image formats do not display in some viewing environments.

Several components, such as the Figure Snapshot component, include an option for specifying the image format. The component setting overrides the image format preference.

## Report Viewing

To control how you view a generated report, you can set the following preferences.

| Preference | Purpose |
|---|---|
| `View command` | Specify the MATLAB command you want to use to view the report.<br>Each file format has an associated default view command preference. You can modify the view command (for example, to support the use of a system browser). |
| `Visible in Report Explorer` | Deselect this check box to make the current output format unavailable in the Report Explorer. For example, if your specified report format is `Word document` and you deselect this check box, then the Microsoft Word document format is no longer available for reports created using the Report Explorer. |
| `Animate Report Explorer when generating reports` | Select this check box if you want components in the Outline pane to be animated as the report generates. This box is selected by default.<br>To speed up the report generation processing, consider clearing this preference. |

## Reset to Defaults

To reset all of the preferences in the Output Format Options section of the Report Generator Preferences pane, click **Reset to Defaults**.

The **Reset to Defaults** button does not change the **Animate Report Explorer when generating reports** preference.

# Change Report Locale

Versions 2.0 and later of the MATLAB Report Generator and Simulink Report Generator software use the locale (system language settings) through the Oracle® Java® interface; therefore, they should use the language specified on your system.

Alternatively, you can change the language directly in Java from the MATLAB command line. The following example sets the language to Italian:

```
java.util.Locale.setDefault(java.util.Locale.ITALY)
```

Alternatively, you can set the preferred language directly in your `.rpt` file:

1. Right-click the **Report** component and select **Send to Workspace**.

   This displays the properties of the report, which are stored in the variable *ans*. Access the report's `Language` property from the command line through this variable. By default, `Language` is `auto`, which indicates that the system's default language is in use.

2. Override the default value of `Language` by setting this property to your desired language; for example, `en` for English or `it` for Italian.

# Convert XML Documents

**In this section...**

## Why Convert XML Documents?

You can generate a report in a different output file format without regenerating it by using either the Report Explorer File Converter or the `rptconvert` command. These utilities convert DocBook XML source files created by the report-generation process into formatted documents such as `HTML`, `RTF`, or `PDF`.

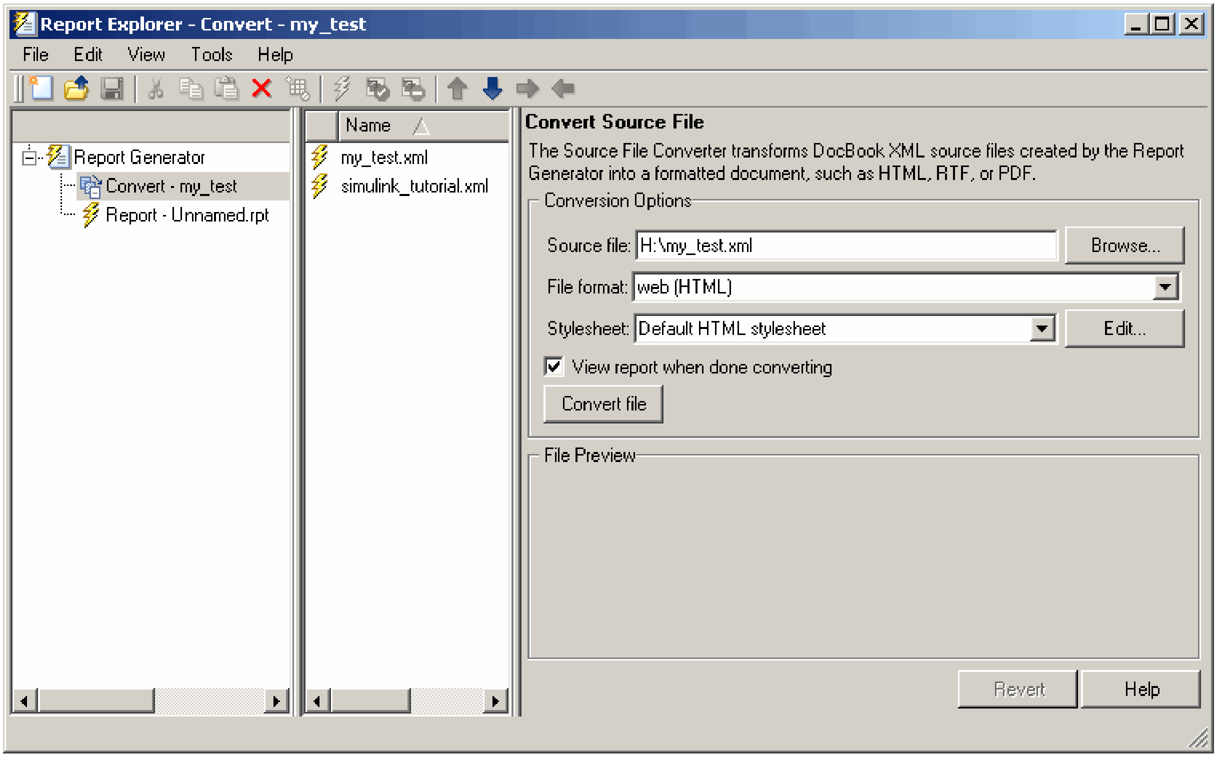
**Note** The report-generation process can only convert XML source files created by the latest version of the software.

## Convert XML Documents Using the Report Explorer

To open the **Convert** Properties pane:

1. In the Report Explorer, select **Tools > Convert source file**.

   The Convert Source File Properties pane appears. All XML files in your current folder appear in the Options pane in the middle.



Convert Source File

The Source File Converter transforms DocBook XML source files created by the Report Generator into a formatted document, such as HTML, RTF, or PDF.

Conversion Options

Source file: H:\my_test.xml

File format: web (HTML)

Stylesheet: Default HTML stylesheet

☑ View report when done converting

File Preview

**2** Select your XML source file using one of the following options:

- Click **Browse** in the Properties pane on the right to browse to the location of your XML source.
- Double-click a file name in the Options pane in the middle to automatically enter it into the **Source file** field in the Properties pane.

**3** Select your output format and stylesheet:

**a** In the **File format** text box, select an output format.

**b** In the **Stylesheet** text box, select a stylesheet. The stylesheet choice depends on the specified output format. You can use a predefined or customized stylesheet.

For more information about available formats and predefined stylesheets, see “Report Output Format and Stylesheet” on page 4-5.

For more information about customizing stylesheets, see “Create a New Stylesheet” on page 7-5.

4 Use the **View Report when done converting** check box to indicate whether you want to view the report after it has conversion.

5 To begin the conversion, click **Convert file**.

## Convert XML Documents Using the Command Line

To convert files using the command line, use the `rptconvert`function.

## Edit XML Source Files

Before you send a source file to the converter, edit it as text in the Report Explorer:

1 In the Outline pane on the left, open the File Converter.

2 Right-click **MATLAB Report Generator** and select **Convert source file**.

3 In the Options pane in the middle, select the source file to edit.

4 In the Properties pane on the right, click **Edit as text**.

5 Use the MATLAB Editor to edit and save the text.

# Create a Log File

A log file describes the report setup file report-generation settings and components. A log file can be used for many purposes, including:

- As a debugger
- As a reference to a report setup file
- To share information about a report setup file through email

A log file includes the following information:

- Report setup file outline
- Components and their attributes
- Generation status messages currently displayed in the **Generation Status** tab

To generate a log file, click **File > Log File**. An HTML version of the log file with the name `<report_template_file_name_log>.html` is saved in the same folder as the report setup file.

# Generate MATLAB Code from Report Setup File

You can generate MATLAB code versions of report setup files in the form of a MATLAB file (`*.m`). A MATLAB file of a report setup file is useful for various purposes, including generating reports and modifying report setup files programmatically.

To generate a MATLAB file, load a report setup file into the Report Explorer and click **File > Generate MATLAB File**. After the MATLAB file generates, it opens in the MATLAB Editor. The filename for the generated file is the file name of the report setup file , preceded by “build.”

## Generate Reports from MATLAB Files

This example generates a MATLAB file from the `figloop_tutorial.rpt` report setup file, which is part of the MATLAB Report Generator software. The example then uses the `report` function to generate a report from the MATLAB file. For more information about this function, see the `report` reference page.

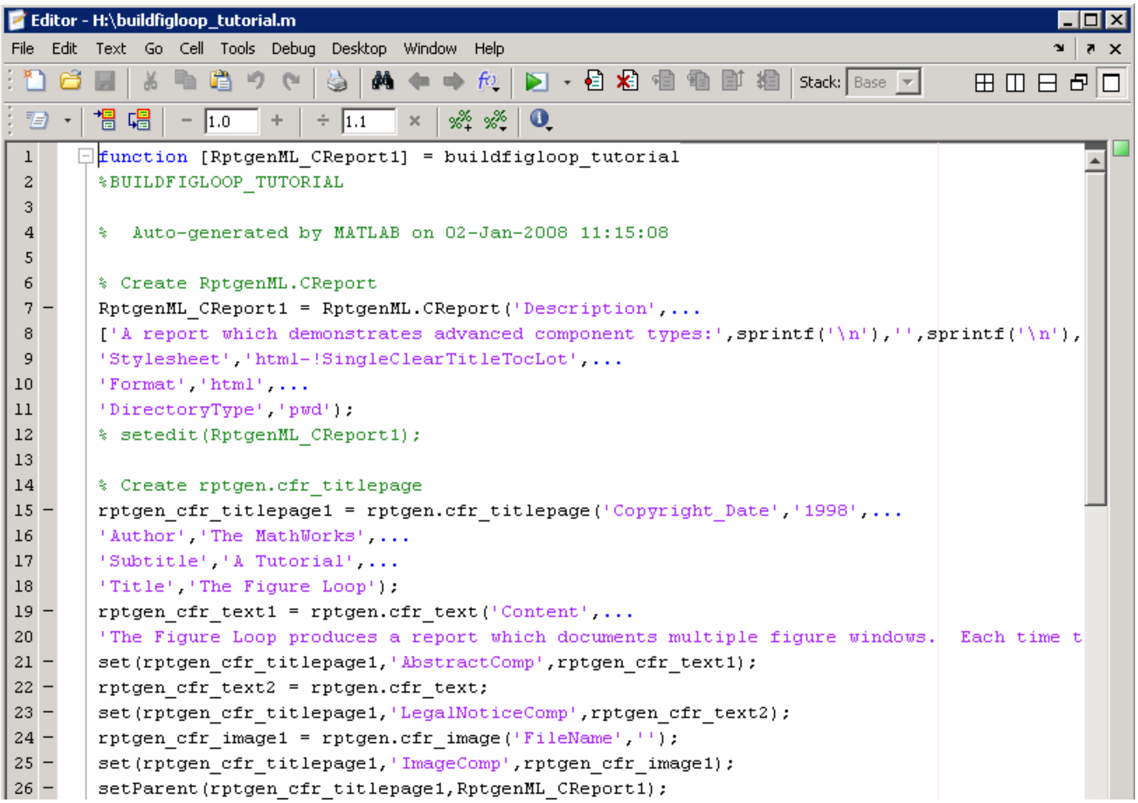
1. Start the Report Explorer by entering `report` in the MATLAB Command Window.

2. In the Options pane in the middle, double-click `figloop_tutorial.rpt` to open its report setup file.

3. In the Outline pane on the left, click `Report - figloop_tutorial.rpt` to select it.

4. In the Report Explorer menu bar, click **File > Generate MATLAB File**.

   The MATLAB Report Generator software generates MATLAB code for the `figloop_tutorial.rpt` report setup file. It saves this code in the `buildfigloop_tutorial.m` file in the folder you specify. Part of this file appears in the following figure.

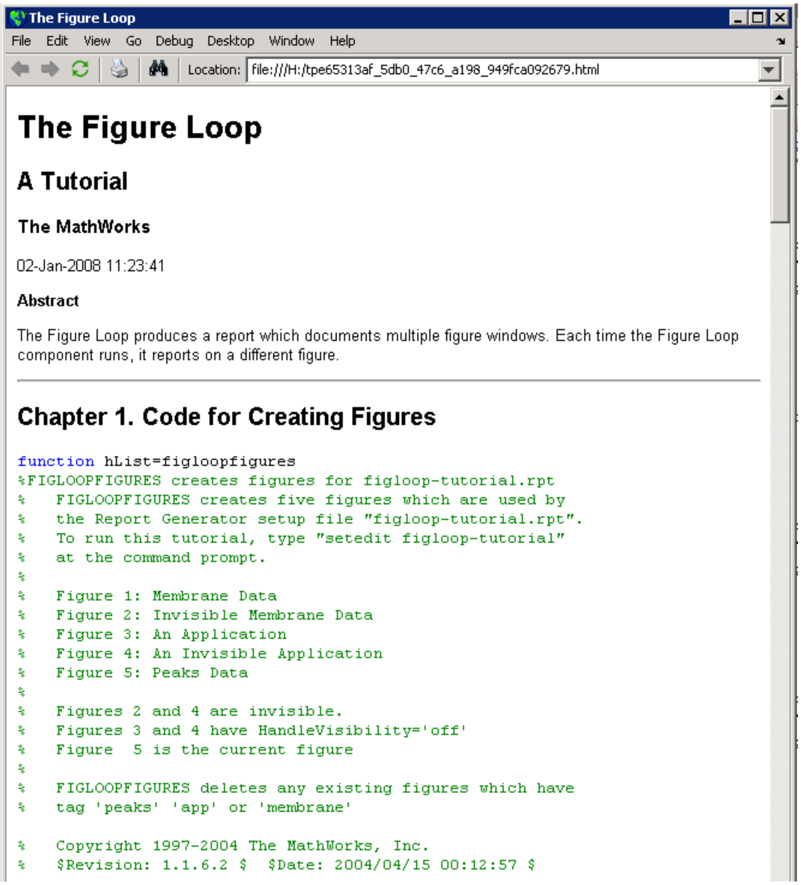

Editor - H:\buildfigloop_tutorial.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1

```
function [RptgenML_CReport1] = buildfigloop_tutorial
%BUILDFIGLOOP_TUTORIAL

%  Auto-generated by MATLAB on 02-Jan-2008 11:15:08

% Create RptgenML.CReport
RptgenML_CReport1 = RptgenML.CReport('Description',...
['A report which demonstrates advanced component types:',sprintf('\n'),'',sprintf('\n'),
'Stylesheet','html-!SingleClearTitleTocLot',...
'Format','html',...
'DirectoryType','pwd');
% setedit(RptgenML_CReport1);

% Create rptgen.cfr_titlepage
rptgen_cfr_titlepage1 = rptgen.cfr_titlepage('Copyright_Date','1998',...
'Author','The MathWorks',...
'Subtitle','A Tutorial',...
'Title','The Figure Loop');
rptgen_cfr_text1 = rptgen.cfr_text('Content',...
'The Figure Loop produces a report which documents multiple figure windows.  Each time t
set(rptgen_cfr_titlepage1,'AbstractComp',rptgen_cfr_text1);
rptgen_cfr_text2 = rptgen.cfr_text;
set(rptgen_cfr_titlepage1,'LegalNoticeComp',rptgen_cfr_text2);
rptgen_cfr_image1 = rptgen.cfr_image('FileName','');
set(rptgen_cfr_titlepage1,'ImageComp',rptgen_cfr_image1);
setParent(rptgen_cfr_titlepage1,RptgenML_CReport1);
```

5 To generate the `figloop_tutorial` report from this MATLAB file, run the following command in the MATLAB Command Window:

```
report(buildfigloop_tutorial);
```

The MATLAB Report Generator software runs and displays the report.



Location: file:///H:/tpe65313af_5db0_47c6_a198_949fca092679.html

# The Figure Loop

## A Tutorial

### The MathWorks

02-Jan-2008 11:23:41

**Abstract**

The Figure Loop produces a report which documents multiple figure windows. Each time the Figure Loop component runs, it reports on a different figure.

## Chapter 1. Code for Creating Figures

```
function hList=figloopfigures
%FIGLOOPFIGURES creates figures for figloop-tutorial.rpt
%   FIGLOOPFIGURES creates five figures which are used by
%   the Report Generator setup file "figloop-tutorial.rpt".
%   To run this tutorial, type "setedit figloop-tutorial"
%   at the command prompt.
%
%   Figure 1: Membrane Data
%   Figure 2: Invisible Membrane Data
%   Figure 3: An Application
%   Figure 4: An Invisible Application
%   Figure 5: Peaks Data
%
%   Figures 2 and 4 are invisible.
%   Figures 3 and 4 have HandleVisibility='off'
%   Figure  5 is the current figure
%
%   FIGLOOPFIGURES deletes any existing figures which have
%   tag 'peaks' 'app' or 'membrane'

%   Copyright 1997-2004 The MathWorks, Inc.
%   $Revision: 1.1.6.2 $  $Date: 2004/04/15 00:12:57 $
```

# Troubleshooting Report Generation Issues

| In this section... |
|---|
| "Memory Usage" on page 4-26 |
| "HTML Report Display on UNIX Systems" on page 4-26 |

## Memory Usage

By default, the MATLAB software sets a limit of 100 MB on the amount of memory the Oracle Java Virtual Machine (JVM™) software can allocate. The memory that the report generation process uses to build a document must fit within this limit. If you are having trouble processing large reports, it might be helpful to increase the amount of memory that MATLAB Report Generator and Simulink Report Generator software can allocate. See the following sections for more information.

### Run the MATLAB Software Without a Desktop

One way to increase the amount of JVM memory available to the MATLAB Report Generator and Simulink Report Generator software is to run the MATLAB software with `-nodesktop` mode enabled.

### Increase the MATLAB JVM Memory Allocation Limit

To increase the amount of JVM memory available by increasing the MATLAB JVM memory allocation limit, from the MATLAB Toolstrip, in the **Home** tab, in the **Environment** section, select **Preferences**. Use the **General > Java Heap Memory** dialog box.

## HTML Report Display on UNIX Systems

HTML reports may not automatically display on some UNIX® platforms. To work around this issue, configure the MATLAB Report Generator software to launch an external browser:

1 In the Report Explorer, click **File > Preferences**.

2 Enter the following text in the **View command** field:

```
web(rptgen.file2urn('%file name'), '-browser')
```

Where *file name* is the name of your report setup file.

# Add Content with Components

# Components

Components are MATLAB objects that specify the content of a report. Add components to specify the types of content that commonly occur in reports. The MATLAB Report Generator provides a set of components for specifying the types of content that commonly occur in MATLAB-based reports. The Simulink Report Generator provides additional components to facilitate generation of reports from Simulink models. You can also create custom components to handle content specific to your application.

Using the Report Explorer, you can interactively combine components to create a report setup that specifies the content of a particular report or type of report. For general information about working with components, see:

- "Insert Components" on page 3-12
- "Set Component Properties" on page 3-14

Use a combination of the following types of components in your report setup file, based on your goals for the report.

| Type of Component | Description |
|---|---|
| "Report Structure Components" on page 5-4 | Include a title page, chapters, sections, paragraphs, lists, tables, and other standard document structure elements. |
| "Table Formatting Components" on page 5-5 | Organize generated content into tables. |
| "Property Table Components" on page 5-6 | Display tables with property name/property value pairs for objects. |
| "Summary Table Components" on page 5-17 | Display tables with specified properties for objects. |
| "Logical and Looping Components" on page 5-22 | Run child components a specified number of times. There are several looping components, including logical loops and Handle Graphics loops. |

## Control Format with Stylesheets

When you generate a report, a stylesheet formats the content specified by these components to reflect the document organization. For example, a report stylesheet typically uses different font sizes for chapter titles and section titles.

In the report setup, specify the stylesheet to use to format your report . You can change the format of your report by modifying the specified stylesheet or selecting another stylesheet. For details, see “Report Output Format and Stylesheet” on page 4-5.

## Control Format at the Component Level

The output format and the stylesheet that you select for a report determines most aspects of the formatting of the generated report. For details, see “Report Output Format and Stylesheet” on page 4-5.

Several components include properties that you can set to specify formatting details for that specific instance of a component. For example, for the `MATLAB Property Table`, you can specify formatting such as whether to display table borders or the alignment of text in table cells.

# Report Structure Components

Use report structure components to organize the content of your report into chapters, sections, paragraphs, lists, tables, and other standard document structure elements. The following table summarizes the report structure components.

| Component | Usage |
|---|---|
| `Title Page` | Generate a title page for a report. |
| `Chapter/Subsection` | Parent components that generate the content of a chapter or chapter subsection. |
| `Paragraph` | Specify the content and text format of a paragraph of text. Can serve as the parent of one or more text components, enabling use of multiple text formats (for example, bold, regular, or italic) in the same paragraph. |
| `Text` | Format strings of generated text. |
| `List` | Generate a list from a cell array of numbers or strings or parent components (for example, Paragraph components) that specify the items in a list. You can create multilevel lists by embedding list components within list components. |
| `Link` | Generate a hyperlink from one location in a report to another or to an external location on the user's file system or the Worldwide Web. |
| `Image` | Insert an image into a report. |
| `Array-Based Table` | Generate a table from a cell array of numbers or strings. |
| `Table` | Parent a table body component. See "Table Formatting Components" on page 5-5. |

# Table Formatting Components

Use table formatting components to organize generated content into tables. The following table summarizes the table formatting components.

| Component | Usage |
|---|---|
| `Table` | Parent a table body component. Can also parent column specification components and a table header and a table footer component. Specifies properties of the table as a whole (for example, its title, number of columns, and border). |
| `Table Body` | Parent the rows that make up the table body. Specifies the default vertical alignment of entries in a table body. |
| `Table Column Specification` | Specify attributes of a table column, such as its width and borders and the default horizontal alignment of column entries. |
| `Table Entry` | Parent a component that determines a table entry's content, such as a paragraph, image, list, or another table component. Specifies attributes of a table entry, such as the number of rows and columns that it spans. |
| `Table Footer` | Parent the row components that generate the content of a table footer. |
| `Table Header` | Parent the row components that generate the content of a table header. |
| `Table Row` | Parent the table entry components that generate the content of a table row. |

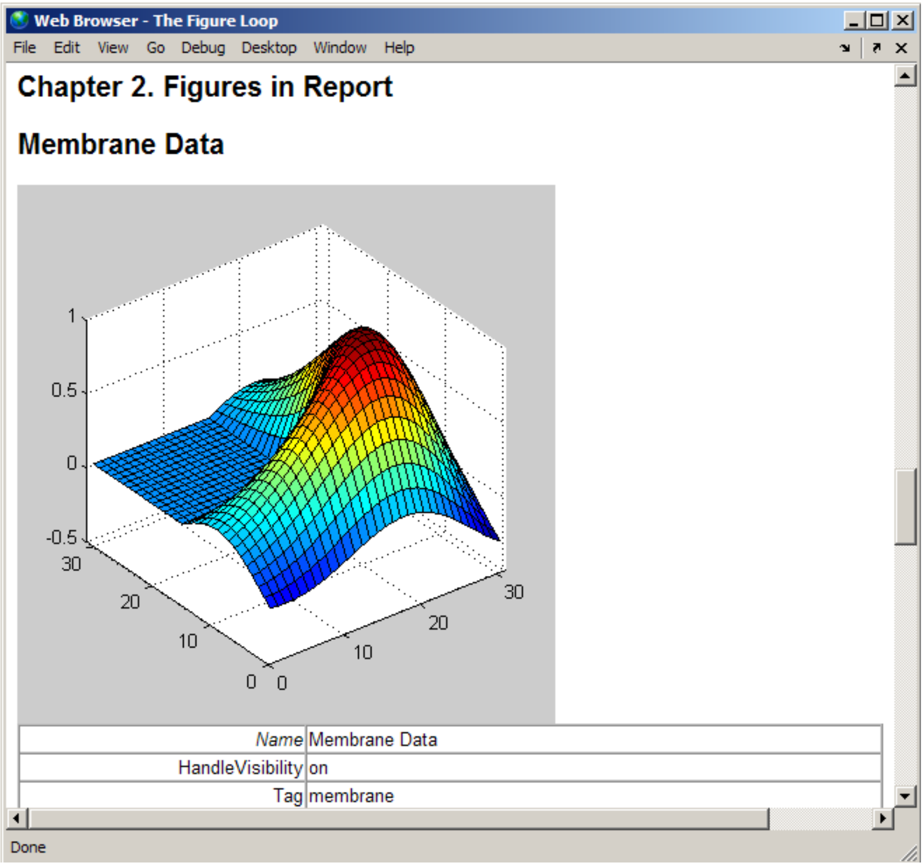
**Tip** Inserting a Table component into a setup also inserts all the descendant components required to generate a 2x2 table, creating a table template. Edit this template to create a table that suits your needs.

# Property Table Components

**In this section...**

## About Property Table Components

Property Table components display property name/property value pairs for objects in tables. The following example shows a property table from the `figloop-tutorial` report.


Web Browser - The Figure Loop
File Edit View Go Debug Desktop Window Help
1
0.5
0
-0.5
30
20
10
0
0
10
20
30
Done

Chapter 2. Figures in Report

Membrane Data

| | |
|---|---|
| *Name* | Membrane Data |
| HandleVisibility | on |
| Tag | membrane |

Many types of property table components are available, including:

- MATLAB Property Table
- Simulink Property Table (requires Simulink Report Generator)
- Stateflow Property Table (requires Simulink Report Generator)

The component used in this example represents MATLAB Report Generator property table components, all of which exhibit similar behavior.

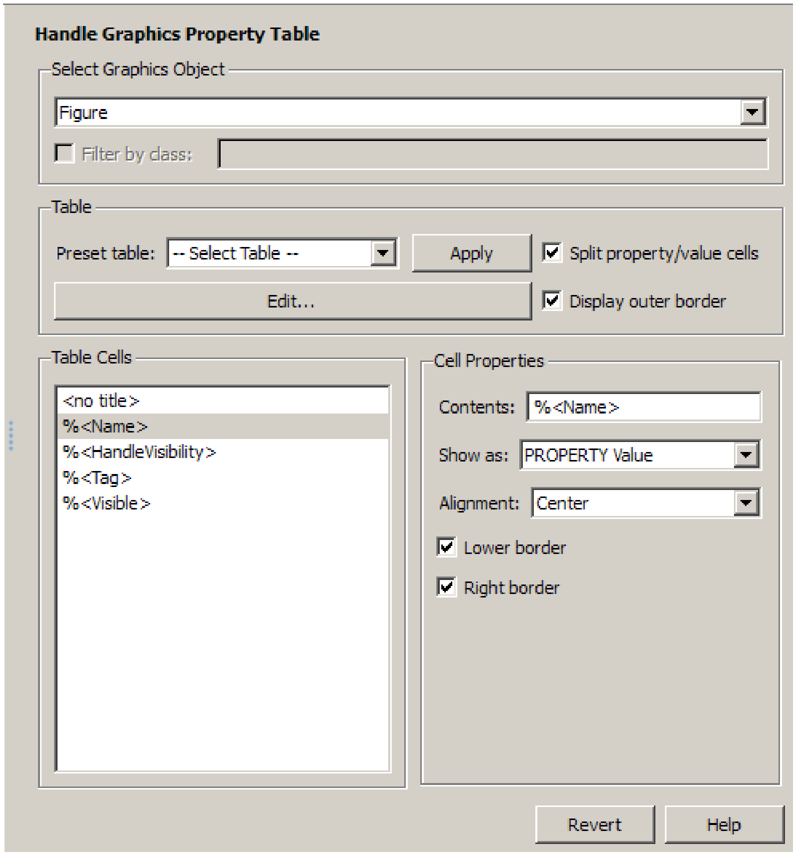
## Open the Example Report Template

This example uses the `figloop-tutorial` report template. To open the figure loop tutorial report template, at the MATLAB command line enter:

```
setedit figloop-tutorial
```

## Examine the Property Table Output

Property pages for all property table components are similar in form. In the Outline pane, select the `Figure Prop Table` component. To modify table settings, in the Handle Graphics Property Table dialog box, click the **Edit...** button.



Handle Graphics Property Table

Select Graphics Object

Figure

☐ Filter by class:

Table

Preset table: -- Select Table --

☑ Split property/value cells

☑ Display outer border

Table Cells

<no title>
%<Name>
%<HandleVisibility>
%<Tag>
%<Visible>

Cell Properties

Contents: %<Name>

Show as: PROPERTY Value

Alignment: Center

☑ Lower border

☑ Right border

## Select Object Types

Property table components offer multiple object types on which to report. For example, the Handle Graphics Property Table lets you report on a figure, an axes object, or a Handle Graphics object.

You can select a different object type on which to report in the **Object type** list in the Properties pane for the component.

## Display Property Name/Property Value Pairs

### Split Property/Value Cells

1 In the Properties pane for the Handle Graphics Property Table component, clear the **Split property/value cells** check box.

2 Click **Edit**. The table is now in *nonsplit mode*. Nonsplit mode supports more than one property name/property value pair per cell and text.

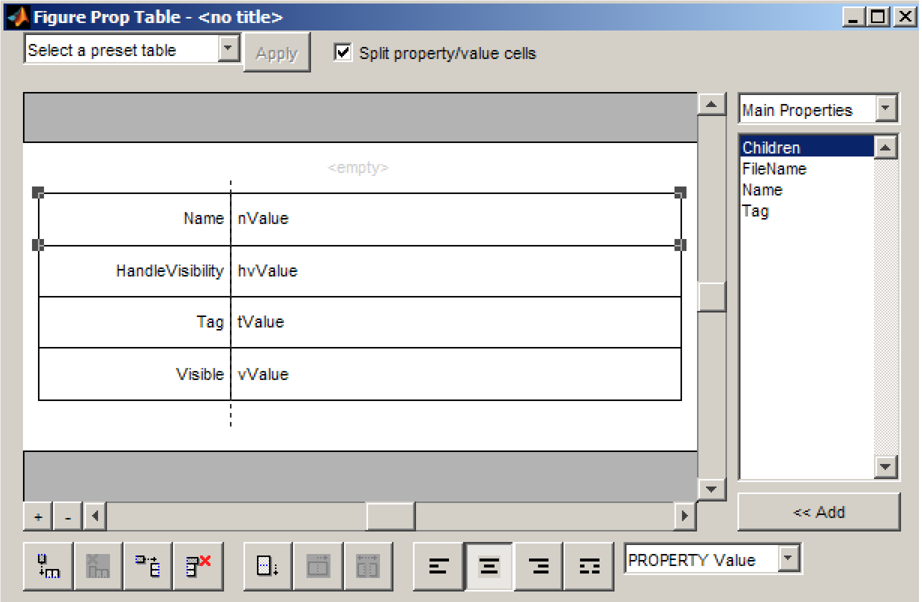


Figure Prop Table - <no title>

☑ Split property/value cells

| | |
|---|---|
| Name | nValue |
| HandleVisibility | hvValue |
| Tag | tValue |
| Visible | vValue |

3 For the property name and property value to appear in adjacent horizontal cells in the table, select the **Split property/value cells** check box. The table is now in *split mode*. Split mode supports only one property name/property value pair per cell. If more than one property pair appears in a cell, only the first pair appears in the report; all subsequent pairs are ignored.

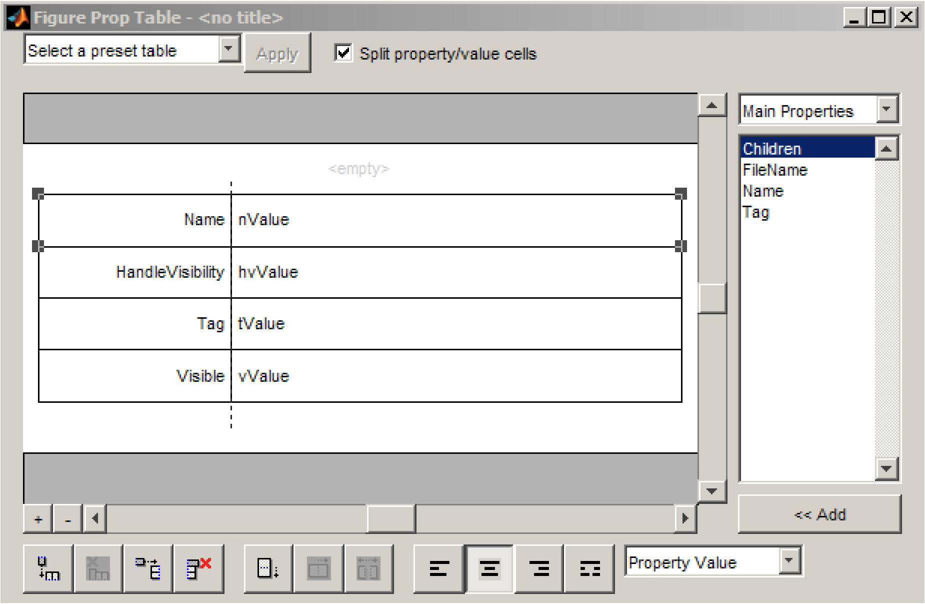


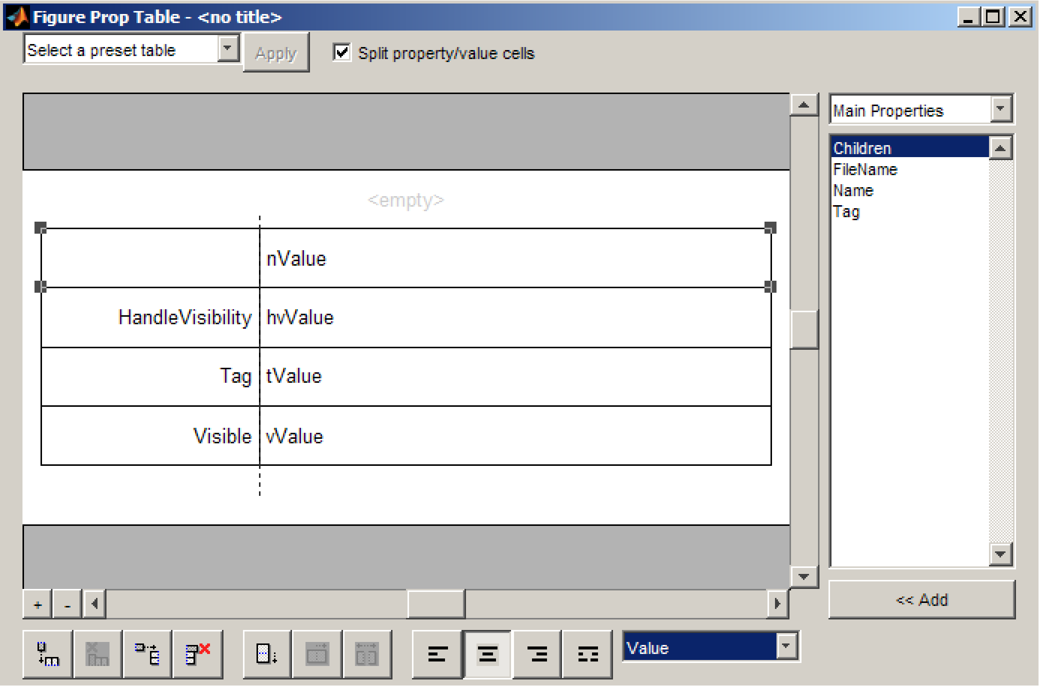
Figure Prop Table - <no title>

☑ Split property/value cells

| | |
|---|---|
| Name | nValue |
| HandleVisibility | hvValue |
| Tag | tValue |
| Visible | vValue |

## Display Options

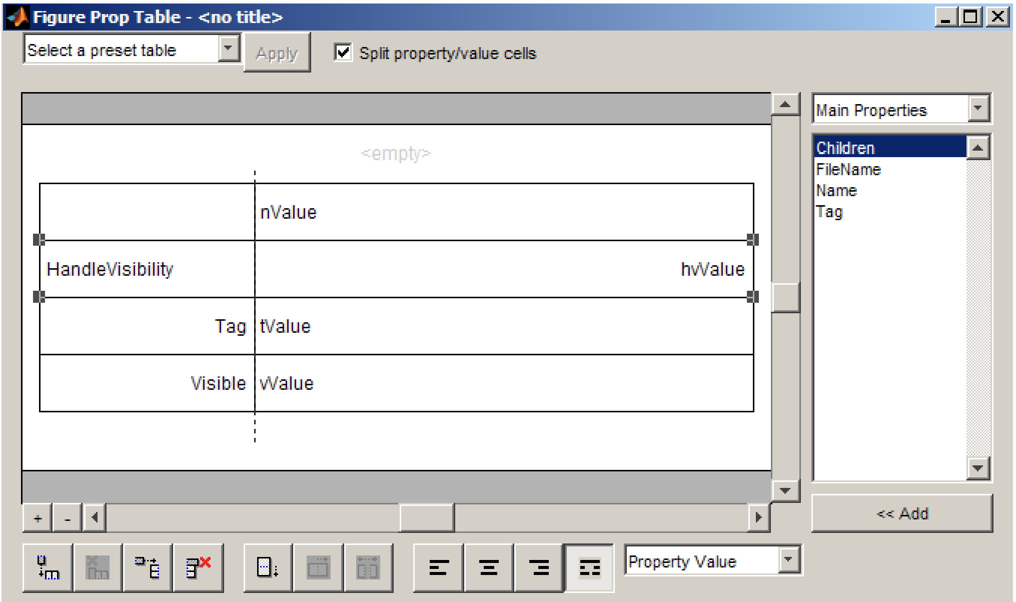
Property name/property value pairs can appear in cells in several ways. To specify how a given property name/property value pair appears in a cell, select that field in the table (for this tutorial, select the `Name` property). Choose `Value` from the display options drop-down list at the bottom of the dialog box. In the selected table row, only the value appears.



## Format Options

To specify alignment for text in a given cell, in the toolbar at the bottom of the dialog box use the four justification buttons.

Select the `HandleVisibility` table row. Then select the double-justify button (the last button to the right).



## Edit Table Titles

Table titles can contain properties and text. By default, the title of a table is the same as the value of the `%<Name>` property. You can modify this property to modify the table title.

**Note** Table titles are always in nonsplit mode.

## Enter Text into Table Cells

For the text to be visible, the table must be in nonsplit mode. Clear **Split property/value cells**.

To enter text into the `HandleVisibility` table cell, double-click the cell. A gray box appears with the label for the cell property.

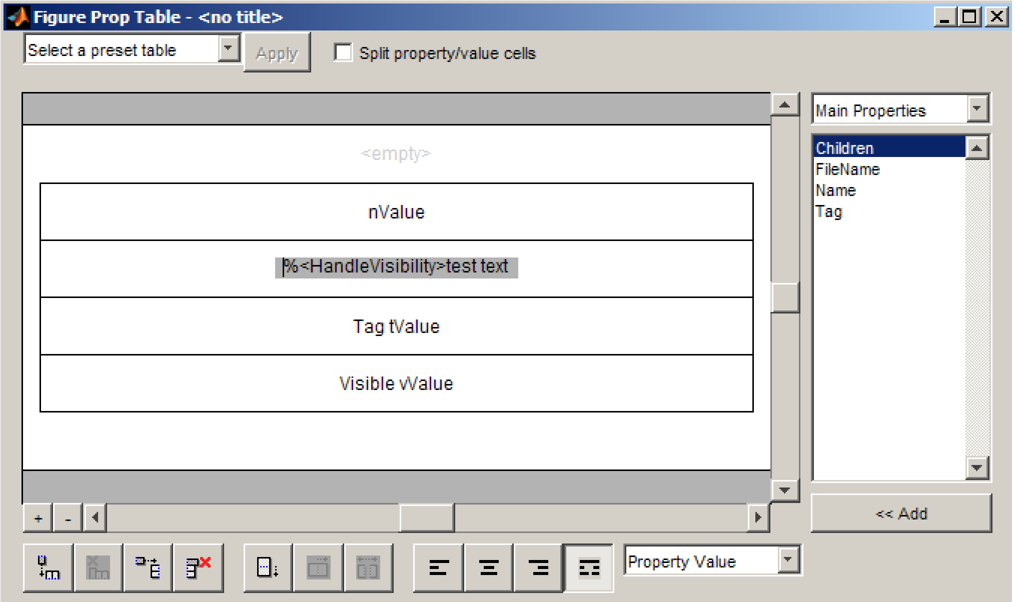


Figure Prop Table - <no title>

Select a preset table | Apply | □ Split property/value cells

<empty>

nValue

%<HandleVisibility>test text

Tag tValue

Visible vValue

Main Properties

Children
FileName
Name
Tag

<< Add

Property Value

If you type text outside the angle brackets, the text appears as is in the report. Text inside the table brackets must specify a valid property name. If you enter an invalid property name, the property name appears in the report without a property value.

# Add, Replace, and Delete Properties in Tables

## Adding Table Properties

To add a Handle Graphics property to a table, use the following steps.

1 In the Figure Property Table window, select a table row above which you want add a new property.

2 Click the Add Row Above Current Cell button

A new row appears above the current row.

3. Add the property to the new table row.

   a. Select the new table row.

   b. In the Properties Type drop-down list at the upper-right of the dialog box, select a property type.

   c. In the **Properties** list, select the property you want to add.

   d. Click the **<< Add** button, or double-click the property name. The property appears in the table row.

   Alternatively, if you know the name of the property you want to add, enter the property name directly into the cell as described in “Enter Text into Table Cells” on page 5-12. For information about adding new table rows, see “Add and Delete Columns and Rows” on page 5-14.

### Replace Table Properties

To replace a property in a cell of a table in split mode, follow the instructions in “Adding Table Properties” on page 5-13.

---

**Note** You cannot use these steps to delete a property in a cell when the table is in nonsplit mode.

---

### Delete Table Properties

Delete a property by backspacing over it or using the **Delete** key.

## Format Table Columns, Rows, and Cells

### Add and Delete Columns and Rows

To add or delete a column or row, select a cell and then click one of the buttons described in the following table.

---

**Note** You cannot delete a row or column when it is the only row or column in the table.

---

| Button | Action |
|---|---|
|  | Add column (added to the left of the selected column) |
|  | Delete selected column |
|  | Add row (added above the selected row) |
|  | Delete selected row |

### Resize Columns

To resize the width of a column, click and drag its vertical borders as needed.

### Merge and Split Cells

To merge or split table cells, select a row and then click one of the buttons described in the following table.

| Button | Action |
|---|---|
|  | Merge cells downward |
|  | Merge cells to the right |
|  | Split cells |

### Display or Hide Cell Borders

To toggle cell borders on and off:

1 Place your cursor in a cell and right-click to invoke its context menu.

2 Choose **Cell borders > Top**, **Bottom**, **Right**, or **Left** to toggle the specified border on or off.

## Zoom and Scroll

You can zoom in and out of the table with the zoom buttons, which are located to the left of the horizontal scroll bar.

| Button | Action |
|---|---|
| + | Zoom in |
| - | Zoom out |

You can scroll vertically and horizontally using the table scroll bars.

## Select a Table

To display property name/property value pairs, you can select a preset table or use a custom table.

- A preset table is built-in and formatted. You can select a preset table in the preset table selection list in the upper-left of the Figure Prop Table window. To apply a preset table, select the table and click **Apply**.
- To create a custom table, select a preset table and modify it to fit your needs by adding and/or deleting rows and properties. You may want to start with the **Blank 4x4** preset table.

**Note** You cannot save a custom table as a preset table. If you do so, you lose all changes to the custom table.

# Summary Table Components

**In this section...**

## About Summary Table Components

Summary table components insert tables that include specified properties for objects into generated reports. Summary tables contain one object per row, with each object property appearing in a column, as shown in the following summary table in the `figloop-tutorial` report.

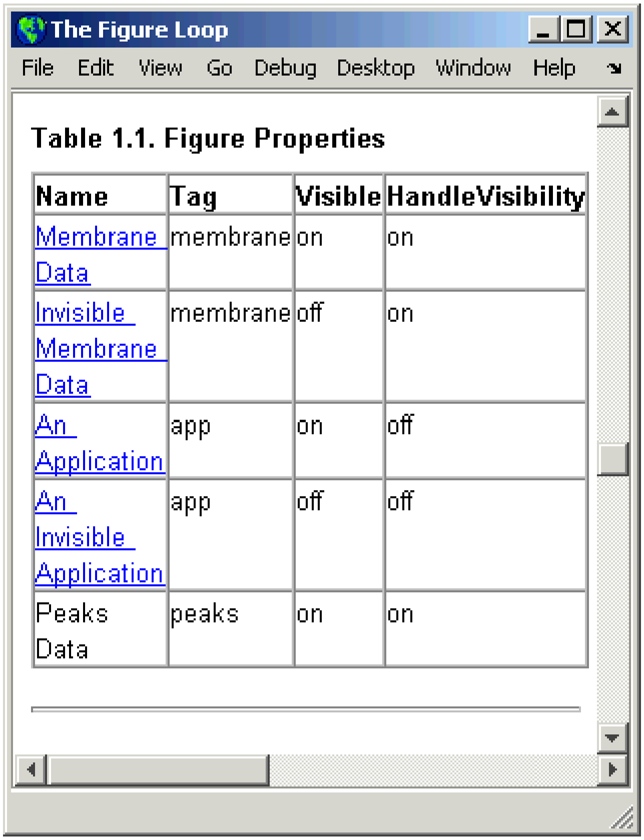


Table 1.1. Figure Properties

| Name | Tag | Visible | HandleVisibility |
|---|---|---|---|
| Membrane Data | membrane | on | on |
| Invisible Membrane Data | membrane | off | on |
| An Application | app | on | off |
| An Invisible Application | app | off | off |
| Peaks Data | peaks | on | on |

Many types of summary table components are available, including:

- Handle Graphics Summary Table
- Simulink Summary Table (requires Simulink Report Generator)
- Stateflow Summary Table (requires Simulink Report Generator)

The component used in this example represents MATLAB Report Generator summary table components, all of which exhibit similar behavior

## Open the Example Report Template

This example uses the `figloop-tutorial` report template. To open the figure loop tutorial report template, enter the following at the MATLAB command line:

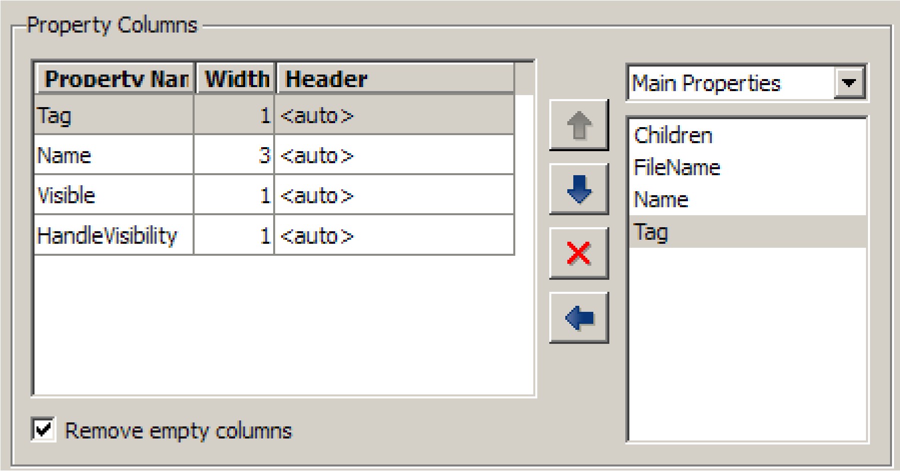
```
setedit figloop-tutorial
```

## Select Object Types

You can use the **Object type** selection list to choose Handle Graphics object types for the summary table, including blocks, signals, systems, and models. The `figloop-tutorial` reports on figure objects.

## Add and Remove Properties

You can select object properties to appear in the summary table from the Property Columns pane. To add a property to the summary table, select the property category from the property category drop-down box to the right of the **Property columns** table. Each property category has its own list of properties, which appear in the field under the box. The following figure shows `Main Properties` as the selected category.



| Property Nar | Width | Header |
|---|---|---|
| Tag | 1 | <auto> |
| Name | 3 | <auto> |
| Visible | 1 | <auto> |
| HandleVisibility | 1 | <auto> |

☑ Remove empty columns

Main Properties

Children
FileName
Name
Tag

To add a property:

1. Select the category from the property category drop-down box.
2. Select a property in the properties list.
3. Click the Add property ⬅ button.

   The property appears in the **Property columns** table.

To remove a property from the table:

1. Select the property in the **Property columns** table.
2. Click the Delete property ✕ button.

   The property name is removed from the **Property columns** table.

---

**Note** After making changes in the Report Explorer, click **Apply** to make the changes take effect.

---

You can define your own properties by entering their names into the **Property columns** table using valid variable notation. For more information, see "%<VariableName> Notation" on page 9-90 on the `Text` component reference documentation.

## Set Relative Column Widths

To apply a relative column width to the summary table columns in the generated report, double-click on the `Width` column of a row in the **Property columns** table . If you do not specify a value for this field, column widths automatically set.

## Set Object Row Options

You can use the Object Rows pane to set options for table rows, including anchor, filtering, and sorting options. Select **Insert anchor for each row** to place an anchor in each table row in the report. Use the **Include figures** list to specify what objects to include in the summary table.

Summary table components in `figloop-tutorial` report on figure objects. For more information on options for these figure objects, see the following sections:

- “Loop on the Current Figure” on page 5-29
- “Loop on Visible Figures” on page 5-29
- “Loop on Figures with Tags” on page 5-29

# Logical and Looping Components

Logical and looping components execute conditionally, determining when a child component executes or how many times a child component executes.

A *looping component* runs its child components a specified number of times. There are several looping components, such as logical loops and Handle Graphics loops.

A *looping component* runs its child components a specified number of times. There are several looping components, such as logical loops, Handle Graphics loops, and model and chart loops. For model and chart loops, you can control aspects such as the order in which the report sorts blocks.

For an example that uses loop components, see "Edit Figure Loop Components" on page 5-27.

You can use loop context functions with loop components. For details, see:

- "Filter with Loop Context Functions"
- "Loop Context Functions"

# Filter with Loop Context Functions

Use loop context functions to filter the modeling elements to report on and to perform special reporting on specific elements.

In the following example, in a Block Loop component, you use `RptgenSL.getReportedBlock` in a Logical If component to report on targeted blocks within a Block Loop component.

For a summary of loop context functions, see "Loop Context Functions" on page 5-26.

## Create and Save the Setup File

1. Open the f14 model.

2. At the MATLAB command prompt, enter:

   ```
   report
   ```

3. In the Report Explorer, select **File > New**.

4. In the Properties pane, set **Directory** to `Present working directory`.

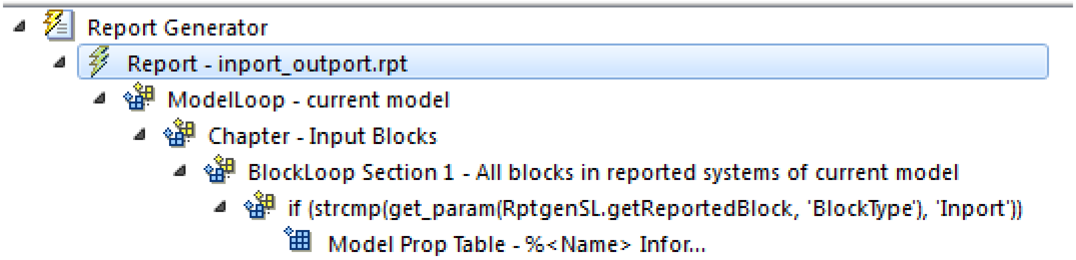
5. Save the setup file as `inport_outport.rpt`.

## Add Components

Add these components to the report, in order.

| From this Library Folder | Add this Component | Set this Property |
|---|---|---|
| `Simulink` | Model Loop | N/A |
| `Formatting` | Chapter | **Title** to `Inport Blocks` |
| `Simulink` | Block Loop | N/A |
| `Logical and Flow Control` | Logical If | **Test Expression** to<br>`strcmp(get_param...`<br>`(RptgenSL.getReportedBlock,'BlockType'),...`<br>`'Inport')` |
| `Simulink` | Simulink Property Table | N/A |

The report setup file looks like this:



```
Report Generator
  Report - inport_outport.rpt
    ModelLoop - current model
      Chapter - Input Blocks
        BlockLoop Section 1 - All blocks in reported systems of current model
          if (strcmp(get_param(RptgenSL.getReportedBlock, 'BlockType'), 'Inport'))
            Model Prop Table - %<Name> Infor...
```

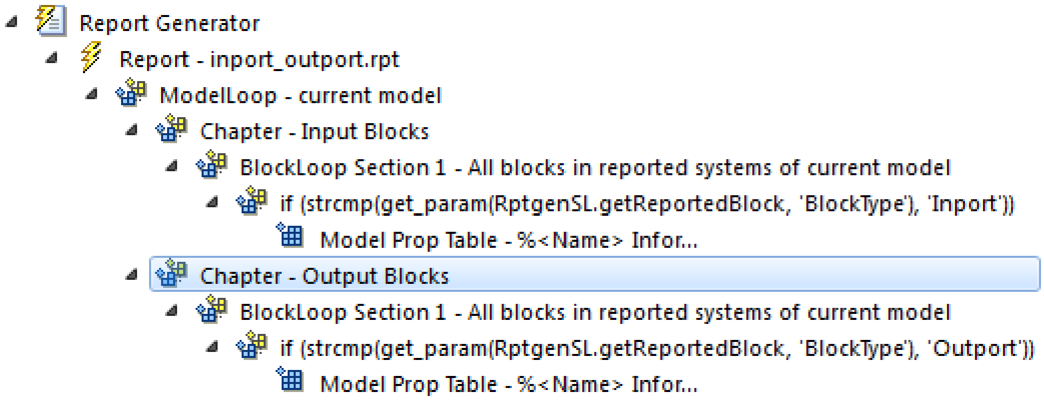
## Run the Report

1. Select `inport_outport.rpt`.

2. From the context menu, select **Report**.

The report includes a chapter with properties for the Inport blocks only.

If you wish, create a second chapter that reports on Outport blocks only, as shown below.



- Report Generator
  - Report - inport_outport.rpt
    - ModelLoop - current model
      - Chapter - Input Blocks
        - BlockLoop Section 1 - All blocks in reported systems of current model
          - if (strcmp(get_param(RptgenSL.getReportedBlock, 'BlockType'), 'Inport'))
            - Model Prop Table - %<Name> Infor...
      - Chapter - Output Blocks
        - BlockLoop Section 1 - All blocks in reported systems of current model
          - if (strcmp(get_param(RptgenSL.getReportedBlock, 'BlockType'), 'Outport'))
            - Model Prop Table - %<Name> Infor...

# Loop Context Functions

| In this section... |
|---|
| "For Simulink Modeling Elements" on page 5-26 |
| "For Stateflow Modeling Elements" on page 5-26 |

You can use these loop context functions in similar ways as shown in "Filter with Loop Context Functions" on page 5-23.

## For Simulink Modeling Elements

| Modeling Element | Looping Component | Function |
|---|---|---|
| Simulink modeling elements | | |
| Block | Block Loop | `RptgenSL.getReportedBlock` |
| Signal | Signal Loop | `RptgenSL.getReportedSignal` |
| System | System Loop | `RptgenSL.getReportedSystem` |
| Model | Model Loop | `RptgenSL.getReportedModel` |

## For Stateflow Modeling Elements

| Modeling Element | Looping Component | Function |
|---|---|---|
| Object | Object Loop | `RptgenSF.getReportedObject` |
| State | State Loop | `RptgenSF.getReportedState` |
| Chart | Chart Loop | `RptgenSF.getReportedChart` |

# Edit Figure Loop Components

**In this section...**

## Figure Loop in a Report

This example uses the Figure Loop, which is representative of many types of loops. The Figure Loop component runs its child components several times. In each iteration, the Figure Loop applies its child components to Handle Graphics figures. The `figloop-tutorial` report setup file creates a report that documents several Handle Graphics figures.

1. At the MATLAB command prompt, enter:

   ```
   setedit figloop-tutorial
   ```

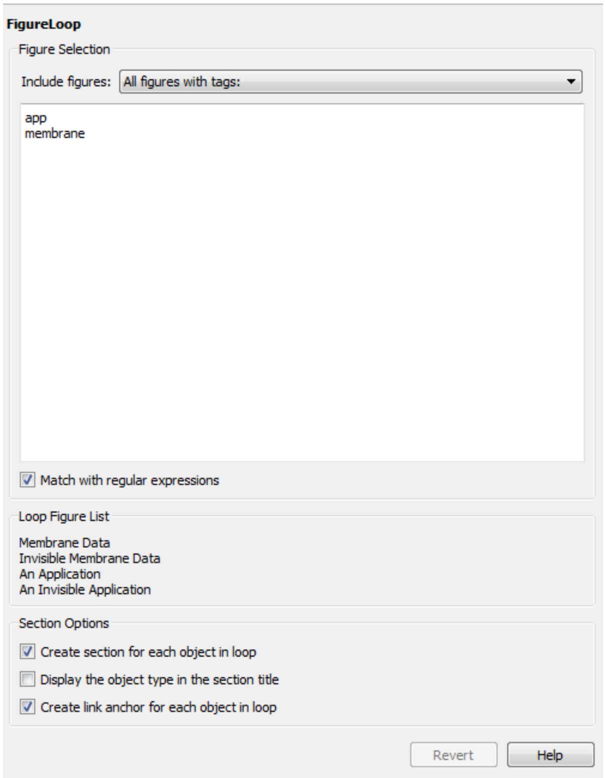
2. To display the Handle Graphics figures, enter:

   ```
   figloopfigures
   ```

   The figures `Membrane Data`, `An Application`, and `Peaks Data` appear on the screen because their `visible` property is `'on'`. The `Invisible Membrane Data` and `An Invisible Application` figures do not appear on screen because their `visible` property is `'off'`. These invisible figures exist, but they are hidden.

3. In the Report Explorer, in the Outline pane on the left, select the Figure Loop component called `Figure Loop Section 1`.

   The Properties pane for the Figure Loop component appears.



FigureLoop

Figure Selection

Include figures: All figures with tags:

app
membrane

☑ Match with regular expressions

Loop Figure List

Membrane Data
Invisible Membrane Data
An Application
An Invisible Application

Section Options

☑ Create section for each object in loop

□ Display the object type in the section title

☑ Create link anchor for each object in loop

## Figure Properties

Figure properties control which figures appear in the report. Table 1.1 of the `figloop-tutorial` report includes a summary of the properties of the figures used in this tutorial.

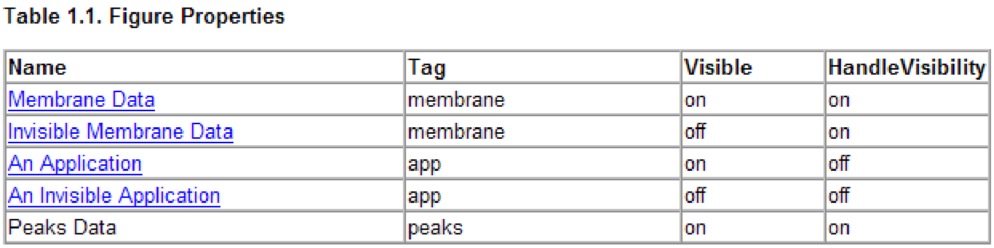


**Table 1.1. Figure Properties**

| Name | Tag | Visible | HandleVisibility |
|---|---|---|---|
| Membrane Data | membrane | on | on |
| Invisible Membrane Data | membrane | off | on |
| An Application | app | on | off |
| An Invisible Application | app | off | off |
| Peaks Data | peaks | on | on |

For this example, do not change these properties. For more information, see "Setting and Querying Property Values".

## Loop on the Current Figure

To include only the current figure in the report, select `Current figure only` from the **Include figures** selection list. The current figure is the figure that is current when the report generates. This figure may not be the same figure that you selected as the current figure in the Report Explorer before report generation. For example, if the report generation process creates figures in your report, the last figure created with `HandleVisibility` set to `'on'` is the current figure.

## Loop on Visible Figures

To include snapshots of all visible figures in your report, in the **Include figures** selection list, select `Visible figures`. This option inserts a snapshot and Property Table for all figures that are currently open and visible.

1. Select the **Data figures only (Exclude applications)** option to exclude figures from the loop whose `HandleVisibility` parameter is `'off'`.
2. To generate the report, in the Report Explorer toolbar click the **Report** button.

In the generated report, scroll down to "Chapter 2 Figures in Report." The `Membrane Data` and `Peaks Data` figures appear in the generated report.

## Loop on Figures with Tags

To include figures with specified tags in the report:

1. In the **Include figures** selection list, select the `All figures with tags` option.
2. In the list of tags, delete `membrane`.
3. Click **Report** to generate the report.

The `An Application` and `An Invisible Application` figures appear in the report. They both have an `app` tag.

## Modify Loop Section Options

In a loop, a *section* refers to a space in the generated report in which information, including text, images, and tables, appears. You can alter the appearance of sections in each loop appear in the report by using the options in the Figure Loop component's Section Options pane.

- **Create Section for Each Object in Loop** — Create an individual section for each object found in the loop, using the object title as the section title. This option is useful when a loop does not contain a Chapter/Subsection component that organizes the loop results.
- **Display the Object Type in the Section Title** — Precede section titles with object titles. Enable this option by selecting **Create section for each object in loop**. For example:

  1. Enter `membrane` back in the list of tags.
  2. Generate the `figloop-tutorial` report.

     The figures produced by the loop are:

     ```
     Membrane Data
     Invisible Membrane Data
     An Application
     An Invisible Application
     ```

  3. Enable the **Create section for each object in loop** option.
  4. Enable the **Display the Object Type in the Section Title** option.
  5. Generate the `figloop-tutorial` report.

     The figures produced are now:

     ```
     Figure - Membrane Data
     Figure - Invisible Membrane Data
     Figure - An Application
     Figure - An Invisible Application
     ```

The figures produced are now:

```
Figure - Membrane Data
Figure - Invisible Membrane Data
Figure - An Application
Figure - An Invisible Application
```

- **Create a Link Anchor for Each Object in Loop** — Create a hyperlink to the object in the generated report.

# Create Custom Components

## About Custom Components

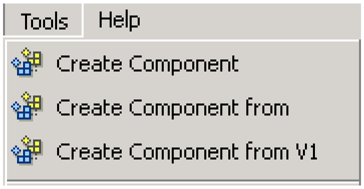
In most cases, the components provided with the MATLAB Report Generator software should be more than adequate for your needs. You can, however, create custom components if you want to generate a report via functionality that is not available in the standard MATLAB Report Generator components. For example, you can create a component that inserts a corporate logo into your report, or a component that plots data.

# Create Custom Components

To create a component:

1. Open the Report Explorer.

2. Select one of the component creation choices from the **Tools** menu:



- To create a custom component, select **Create Component**.
- To create a custom component from an existing component, select **Create Component from**.
- To create a component from an existing version 1 component, select **Create Component from V1**.

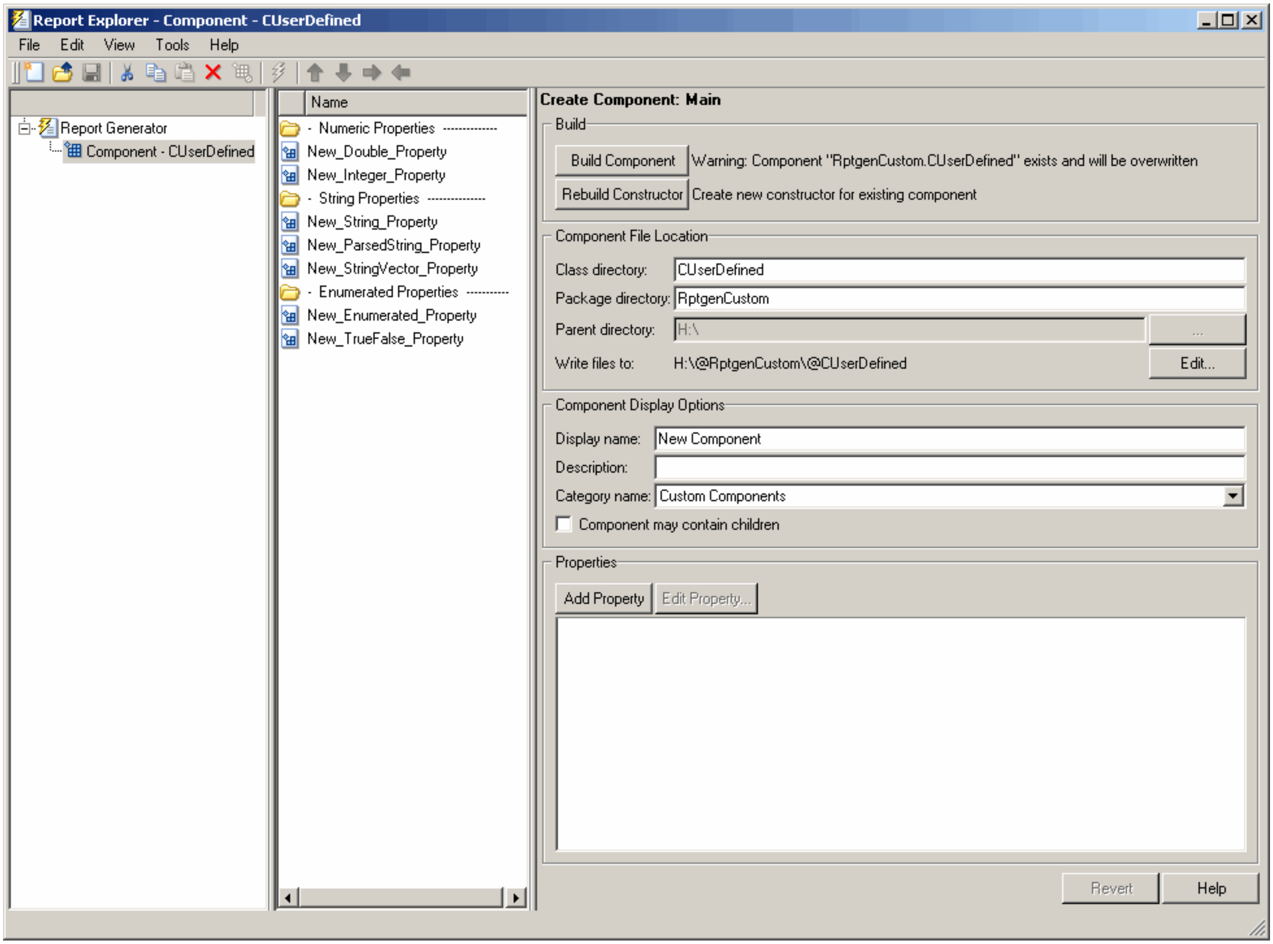
---

**Tip** You can also create a custom component by clicking the **Create a new user-defined reporting component** link in the Report Explorer Properties pane on the right.

---

The Report Explorer displays as follows:

- The Outline pane on the left displays the structure of components you create.
- The Options pane in the middle lists properties you add to components.
- The Properties pane on the right specifies the behavior of component properties.



Create Component: Main

Build

Build Component: Warning: Component "RptgenCustom.CUserDefined" exists and will be overwritten

Rebuild Constructor: Create new constructor for existing component

Component File Location

Class directory: CUserDefined

Package directory: RptgenCustom

Parent directory: H:\

Write files to: H:\@RptgenCustom\@CUserDefined

Component Display Options

Display name: New Component

Description:

Category name: Custom Components

□ Component may contain children

Properties

3 Specify properties of the component in the Properties pane of the Report Explorer. For more information, see "Define Components" on page 6-6.

4 Specify tasks you want the component to perform by editing the MATLAB files that comprise the framework of the component. For more information, see "Specify Tasks for a Component to Perform" on page 6-14.

5 Build the component. For more information, see "Build Components" on page 6-12.

After you build the custom component, you can use it to specify options for your generated report in the report setup file.

**Note** You must restart the MATLAB software session before using a newly created or rebuilt component.

# Define Components

**In this section...**

## Required Component Data

You must specify the following information when you create a component:

1. The path where you want to put the folder that contains all files for the component. For information on how to specify this folder, see "Specify the Location of Component Files" on page 6-6.

2. Properties of the component. For more information, see "Specify Component Properties" on page 6-9.

3. Display options for the component, including its display name, category, and description. For more information, see "Set Component Display Options" on page 6-7.

## Specify the Location of Component Files

You can create components that perform similar functions and group them in *Package Directories*. Each package folder must have a *Parent Directory* that is in the MATLAB path. When you build a new component, the MATLAB Report Generator software creates files that make up the component. These files are stored in the folder structure `<parent>/@package_name/@class_name`.

Specify these directories in the following fields in the **Component File Location** area of the Properties pane:

1. **Class Directory Field**. Specify a class name for your component. The build process creates a folder with the name you specify and places the component's files in it. The class folder name must be unique for each component in the package. By convention, component class names begin with an uppercase or lowercase letter `c`; for example, `cUserDefinedComponent`.

2. **Package Directory Field.** Specify the folder in which to store files for groups of components you create. Files for each component are stored in a subfolder with the name you entered into **Class Directory Field**.

3. **Parent Directory Field**. Specify this folder when you create a package for the first time. This folder is the parent folder of the Package Directory.

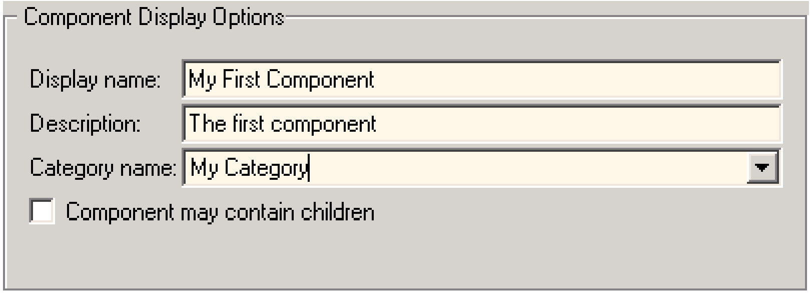
## Set Component Display Options

You can specify how you want your component to appear in the Report Explorer by entering data in the **Component Display Options** area of the Properties pane. Enter the following information:

1. **Display Name.** Specify a display name for the component to appear in the list of components for its associated category. Component categories and display names appear in the Options pane in the middle of the Report Explorer.

   For information on specifying component categories, see step 3, **Category Name**.

   The following example shows how to create a component called `My First Component` in a category called `My Category`.


Component Display Options

Display name: My First Component

Description: The first component

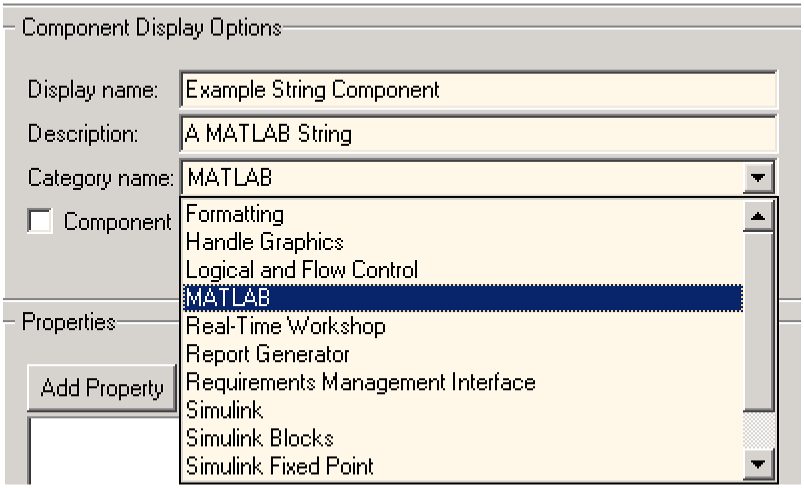
Category name: My Category

□ Component may contain children

2 **Description.** Enter a description for the component. This description appears when you click the component name or category name in the Options pane in the middle of the Report Explorer. Make the description informative, but brief.

3 **Category Name.** Specify the category of components to which the new component belongs. The component appears under this category in the Options pane in the middle of the Report Explorer.

Predefined choices appear in the **Category name** list. Select a component category from this list.


Component Display Options

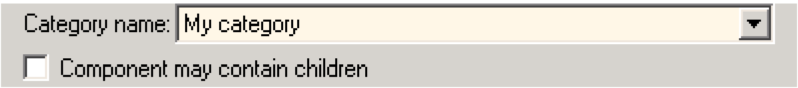
Display name: Example String Component

Description: A MATLAB String

Category name: MATLAB

□ Component

To create a custom component category, type the name for the category into the **Category name** field. This category name appears in the list of available categories in the Report Explorer.



Category name: My category

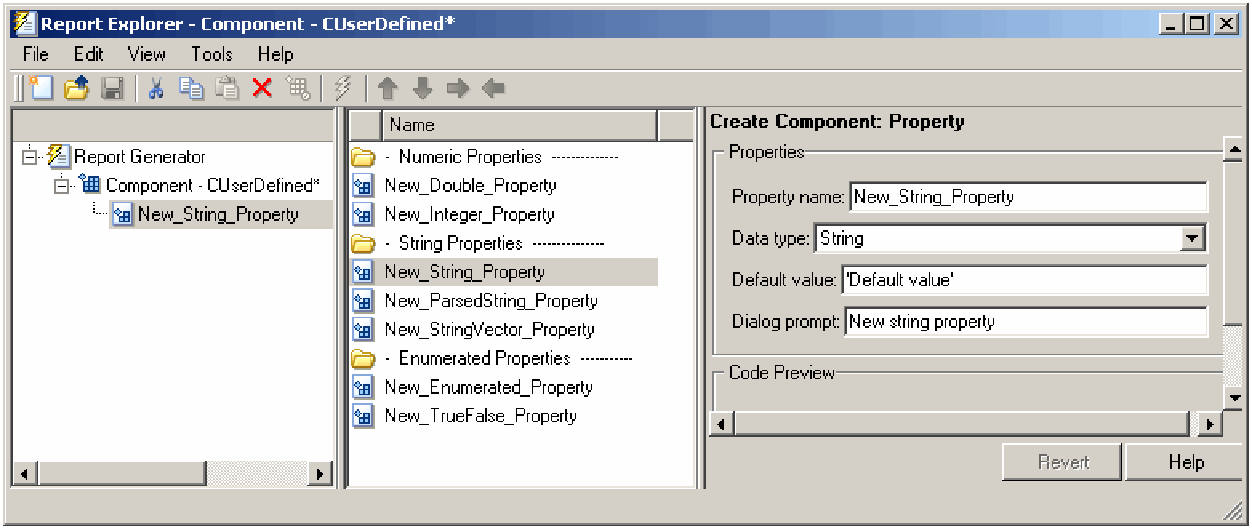
□ Component may contain children

**4 child components**.

Select the **Component may contain children** check box if you want the component to have child components. Child components appear under the component in the Report Explorer hierarchy. During report generation, the component runs all child components and includes their output in the report.

## Specify Component Properties

Component properties determine how a component behaves and what information it inserts into a report. To see the current value of a component's property, double-click it in the Outline pane on the left in the Report Explorer. For example, the following figure displays the property values for `New_String_Property`.



Property name: New_String_Property

Data type: String

Default value: 'Default value'

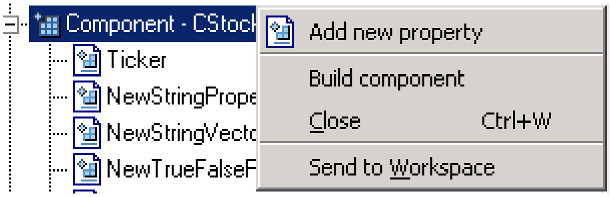
Dialog prompt: New string property

## Add Properties to Components

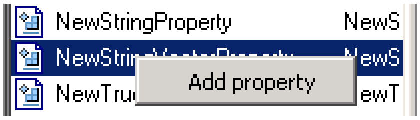
You add properties to a component from the properties list. Each property has a default value that you can modify as needed.

There are several ways to add properties to components:

1 Right-click the name of the component to which you want to add properties in the Outline pane on the left. Select **Add new property** from its context menu.

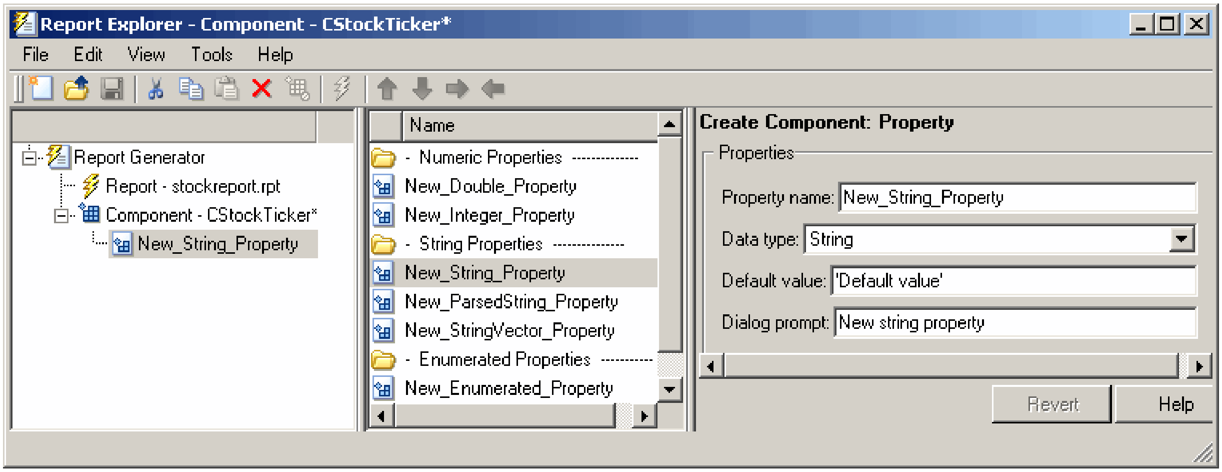


2 Right-click the name of a predefined property in the Options pane in the middle. From the context menu, select **Add property**.



3 Left-click the name of a property in the Options pane, and then drag it on top of a component in the Outline pane on the left.

4 Double-click the property name in the Options pane in the middle. The property is added to the component and property values appear in the Properties pane on the right.


Report Explorer - Component - CStockTicker*
File Edit View Tools Help
Report Generator
Report - stockreport.rpt
Component - CStockTicker*
New_String_Property
Name
- Numeric Properties -------------
New_Double_Property
New_Integer_Property
- String Properties --------------
New_String_Property
New_ParsedString_Property
New_StringVector_Property
- Enumerated Properties ----------
New_Enumerated_Property
Revert
Help

Create Component: Property

Properties

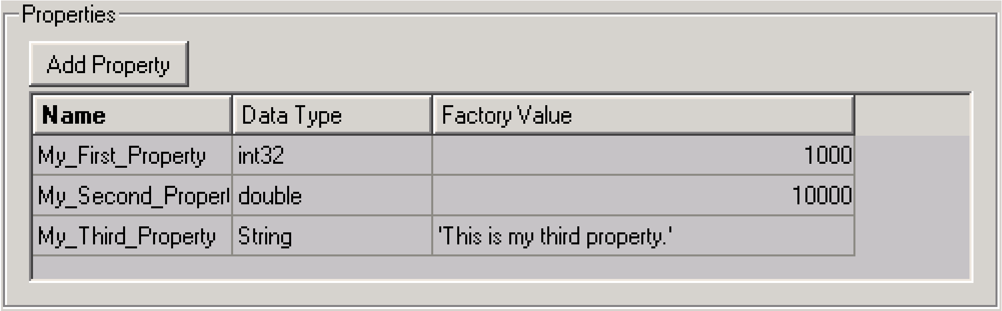
Property name: New_String_Property

Data type: String

Default value: 'Default value'

Dialog prompt: New string property

5 Click the **Add Property** button on the Properties pane on the right.



| Name | Data Type | Factory Value |
|---|---|---|
| My_First_Property | int32 | 1000 |
| My_Second_Properl | double | 10000 |
| My_Third_Property | String | 'This is my third property.' |

## Specify Component Properties

1 **Property Name.** Create a name for the new property. A property name must be a valid MATLAB variable name, and must be unique within a component.

2 **Data Type.** Specify the property's data type. Options are:

- `Double`
- `Enumeration`
- `Integer`

- `String`
- `String Vector`
- `%<Parsed String>`

  Use this data type to include the value of a variable in the MATLAB workspace in a component. For more information about this data type, see "%<VariableName> Notation" on page 9-90 on the `Text` reference page.
- `True/False`

3. **Default Value.** Set a default value for the property. The default value must be compatible with the data type. If incompatibilities exist between the default value and the data type, the component might not build.

4. **Dialog Prompt.** This text appears next to the widget on the component's dialog box. It indicates what the property does and how it affects report generation.

**Note** When the component builds, a colon is appended to your entry in the **Dialog prompt** field. Your entry appears in the Properties pane with the colon appended.

## Modify Existing Components

Report components are modifiable. You can derive a new component from an existing component by double-clicking the name of the component and modifying its values and properties.

## Build Components

After you have entered all data required for defining the component, you build it by clicking the **Build Component** button. The build process creates all files needed for the component and stores them in the specified folder. For more information about specifying where components are stored, see "Specify the Location of Component Files" on page 6-6.

**Note** Existing files in this location are overwritten.

## Rebuild Existing Components

To add, remove, or change properties of an existing component, use the **Rebuild Constructor** button. This button becomes active only after you have previously created a component using the **Build Component** button. To activate the **Rebuild Constructor** button, specify the **Package name** and **Class name** for an existing component. These fields are located in the **Component File Location** area of the Properties pane.

If you select a component using **Tools > Create component from**, the component's fields are filled in automatically and the button becomes active.

After you have finished modifying the component, click the **Rebuild Constructor** button to rebuild the component. Writable files in the component's folder location are not overwritten.

## Remove a Component

To remove a component:

1. Delete its class folder, `<root>/@package_name/@class_name`. If the component you want to remove is the only component in the package, delete the entire package.

2. Edit `<root>/@package_name/rptcomps2.xml` to remove the XML element that registers the component.

# Specify Tasks for a Component to Perform

**In this section...**

## About Component Customization

Building a component creates MATLAB files in the MATLAB workspace. Specify tasks that you want your component to perform by editing these MATLAB files.

---

**Note** You *must* specify the format and content of your report output by editing `execute.m`. This file is called during report generation to invoke your component's tasks. Optionally, you can specify additional component properties and behavior by editing other MATLAB files.

---

For more information, see the following sections:

- "Required Customization: Specify Format and Content of Report Output" on page 6-15
- "Change a Component's Outline String in the Report Explorer Hierarchy" on page 6-17
- "Modify the Appearance of Properties Dialog Boxes" on page 6-18
- "Specify Additional Component Properties" on page 6-18

## Required Customization: Specify Format and Content of Report Output

After you build the component, specify the format and content of your report output by editing the `execute.m` file.

The `execute` command has the following syntax:

```
out = execute(thisComp, parentDoc)
```

Where:

- `thisComp` is a handle to the component that you are running.
- `parentDoc` is a handle to the document that you are generating.
- `out` is a Document Object Model (DOM) node or string to add to the report.

  For information on manipulating DOM nodes, see `xmlwrite` in the MATLAB documentation.

One or more default lines of code within the `execute.m` file show each property for the component. Here is an example of a component property line within an `execute.m` file:

```
pstring = thisComp.NewStringProperty;  % New string property;
```

The following sections describe how to edit `execute.m` to create additional report elements.

### Create Tables

To create a table, replace the `Source` property value with the name of a cell array or structure:

```
out = execute(rptgen.cfr_table(...
'Source', tableSrc,...
'numHeaderRows',1,...
'TableTitle','Example Title'),...
 parentDoc);
```

For more information, enter `help(rptgen.cfr_table)` at the MATLAB command line.

### Create Lists

To create a list, replace the `Source` property value with the name of a cell vector:

```
out = execute(rptgen.cfr_list(...
'Source', listSrc,...
'ListStyle','orderedlist',...
'ListTitle','Example List'),...
 parentDoc);
```

For more information, enter `help(rptgen.cfr_list)` at the MATLAB command line.

### Create Text

To create text, replace the `ParaText` property value with a text string:

```
out = execute(rptgen.cfr_paragraph(...
'ParaText', paraSrc,...
 parentDoc);
```

For more information, enter `help(rptgen.cfr_paragraph)` at the command line.

### Create Figures

To create figures, specify a figure in the `FigureHandle` property value.

```
figSrc = gcf;
out = execute(rptgen_hg.chg_fig_snap(...
'FigureHandle', figSrc,...
'Title', '',...
'isResizeFigure', 'manual',...
'PrintSize', [6 4],...
'PrintUnits', 'inches'),...
 parentDoc);
```

For more information, enter `help(rptgen_hg.chg_fig_snap)` at the MATLAB command line.

### Run Child Components

The following code runs child components. The first line calls execute.m for child components. The second line appends the results of running the child components to the report:

```
childOut = thisComp.runChildren(parentDoc);
out = parentDoc.createDocumentFragment(out, childOut);
```

## Change a Component's Outline String in the Report Explorer Hierarchy

To change the string used to describe the component in the Report Explorer hierarchy, edit the getOutlineString MATLAB file. By default, getoutlinestring returns the display name of the component. The getOutlineString command has the following syntax:

```
olstring = getOutlineString(thisComp)
```

Where:

- thisComp is the component whose description you are specifying.
- olstring is a single-line string that displays information about the component. It can contain a maximum of 32 characters.

Customize the string to include additional information about the component, such as information about its properties. In the following example, the truncatestring function converts input data into a single-line string. If the data is empty, the second argument is the return value, The third argument is the maximum allowed size of the resulting string.

```
cInfo = '';
pstring = rptgen.truncateString(thisComp.string,'<empty>',16);
```

Use a dash (-) as a separator between the name and additional component information, as follows:

```
if ~isempty(cInfo)
   olstring = [olstring, '-', cInfo];
end
```

## Modify the Appearance of Properties Dialog Boxes

You can edit the `getdialogschema.m` file to control most aspects of dialog box layout, including:

- Creation and placement of widgets
- Organization of widgets into panes
- Creation of the top-level display within which panes reside

The syntax of the command is:

```
dlgstruct = getdialogschema(thisComp, name)
```

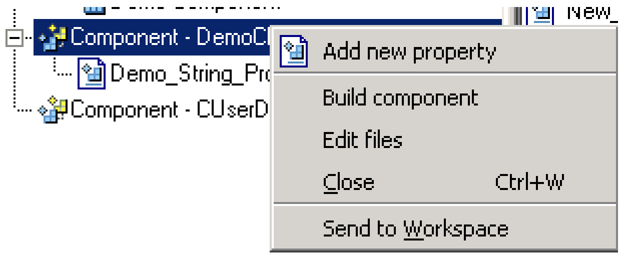
Where:

- `thisComp` is the instance of the component being edited.
- `name` is a string that is passed to `getdialogschema` to build a specific type of pane. Usually, `name` is empty in the Report Explorer.

**Note** Do not modify fields that are not explicitly included in this file. These fields are subject to change in future releases.

## Specify Additional Component Properties

You can edit additional MATLAB files to customize your component further. To access these files, right-click the component in the Outline pane on the left in the Report Explorer and select **Edit files** from its context menu.



For more information, see the following sections:

- “Specify Whether Components Can Have Children Components” on page 6-19
- “Modify a Component Description” on page 6-19
- “Change a Component Display Name” on page 6-19
- “Change a Component Category Name” on page 6-20
- “Register Components” on page 6-20
- “Display Component Help in the MATLAB Help Browser” on page 6-20

### Specify Whether Components Can Have Children Components

To specify whether a component can have children, edit `getParentable.m`. This command returns the value `true` or `false`. For example, if you no longer want your component to have child components, modify the value within the code as follows:

```
p = false;
```

### Modify a Component Description

The description n `getDescription.m` is the same value as the **Description** field in the Report Explorer. The following example shows how to edit the `compDesc` string in this file to change a component’s description to `An example component`:

```
compDesc = 'An example component';
```

### Change a Component Display Name

The display name in `getName.m` is the same value as the **Display name** field in the Report Explorer. The following example shows how to edit the `compName` string in this file to change a component’s display name to `Example Component`:

```
compName = 'Example Component';
```

## Change a Component Category Name

The category name in `getType.m` is the same value as the **Category name** field in the Report Explorer. The following example shows how to edit the `compCategory` string in this file to change a component's category name to `Custom Components`:

```
compCategory = 'Custom Components';
```

## Register Components

You can register components in the Report Explorer using `rptcomps2.xml`. This file also helps build the list of available components.

The content of this file must be consistent with the values in the `getName.m` and `getType.m` files. If you have changed values in either of these files, you must also change their values in `rptcomps2.xml`. You must restart the MATLAB software session for the Report Explorer to display new information.

## Display Component Help in the MATLAB Help Browser

The `viewHelp.m` file displays a help file for the component within the MATLAB Help browser. To display the help file, highlight the name of the component in the Report Explorer and click **Help**.

# Customized Components

**In this section...**

**Note** These examples require the Datafeed Toolbox™ software.

## Fetching Securities Data and Displaying It in a Table

This example shows how to create a component that fetches securities data and displays it in a report as a table.

1 Create a component named `Equity Values` in the class folder named `CStockTicker`.

2 Give the component one string property named `Ticker`, and specify its attributes.

   a In the Options pane of the Report Explorer, double-click **New_String_Property**.

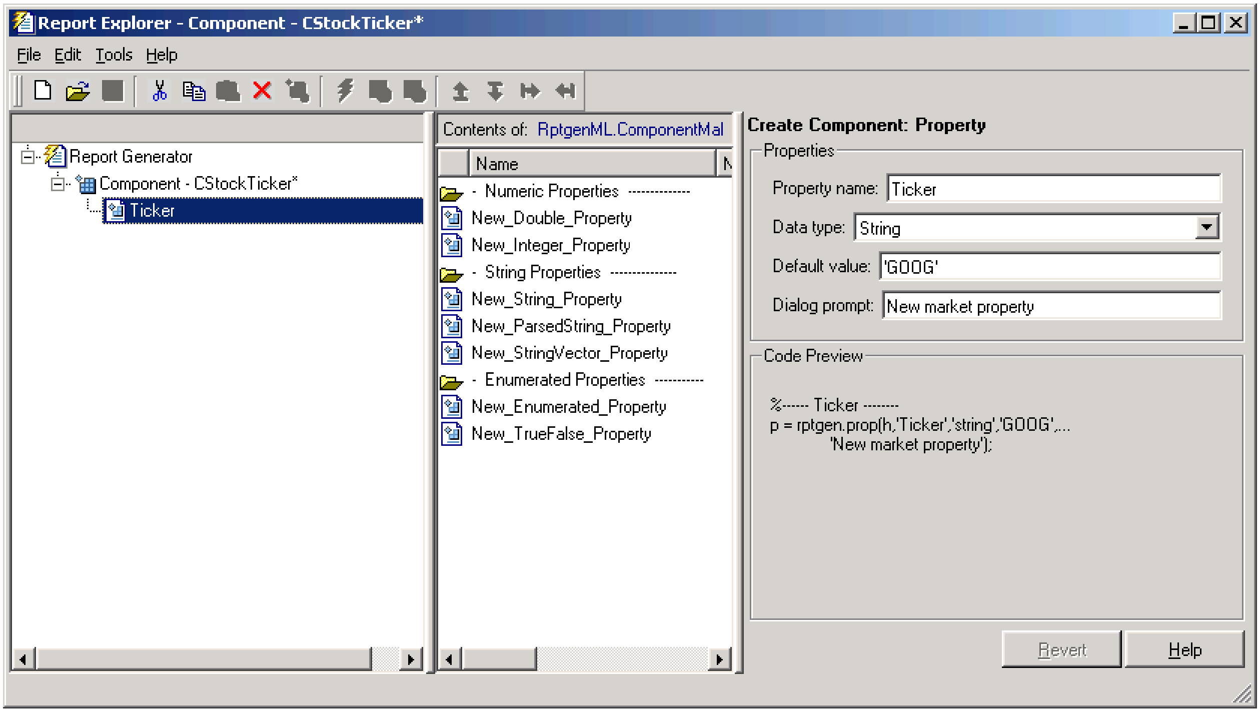
   b For **Property name**, specify a valid MATLAB variable name.

   c Specify the property’s data type. In this case, `Ticker` is a string value, which is the default data type.

   d Specify the property’s default value.

   Because this example fetches ticker values for the security `Google`, set the **Default value** to `'GOOG'`. (The single quotation marks are required to specify a string value for this field.)

   Your specified settings appear in the **Code Preview** pane.



Report Generator
Component - CStockTicker*
Ticker

Contents of: RptgenML.ComponentMal

| Name |
|---|
| - Numeric Properties ------------- |
| New_Double_Property |
| New_Integer_Property |
| - String Properties -------------- |
| New_String_Property |
| New_ParsedString_Property |
| New_StringVector_Property |
| - Enumerated Properties ---------- |
| New_Enumerated_Property |
| New_TrueFalse_Property |

Create Component: Property

Properties

Property name: Ticker

Data type: String

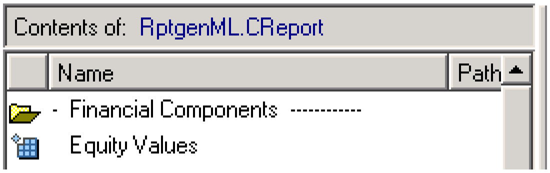
Default value: 'GOOG'

Dialog prompt: New market property

Code Preview

```
%------ Ticker -------
p = rptgen.prop(h,'Ticker','string','GOOG',...
        'New market property');
```

**3** To build the new component, click the **Build** button in the Report Explorer. The Equity Values component now appears in the Options pane in the middle of the Report Explorer.



Contents of: RptgenML.CReport

| Name | Path |
|---|---|
| - Financial Components ------------ | |
| Equity Values | |

**4** Edit the component's `execute.m` file to retrieve stock market data and display it in a table in the generated report.

**a** In the `@CStockTicker` folder, open `execute.m`.

**b** Enter the following text into `execute.m`.

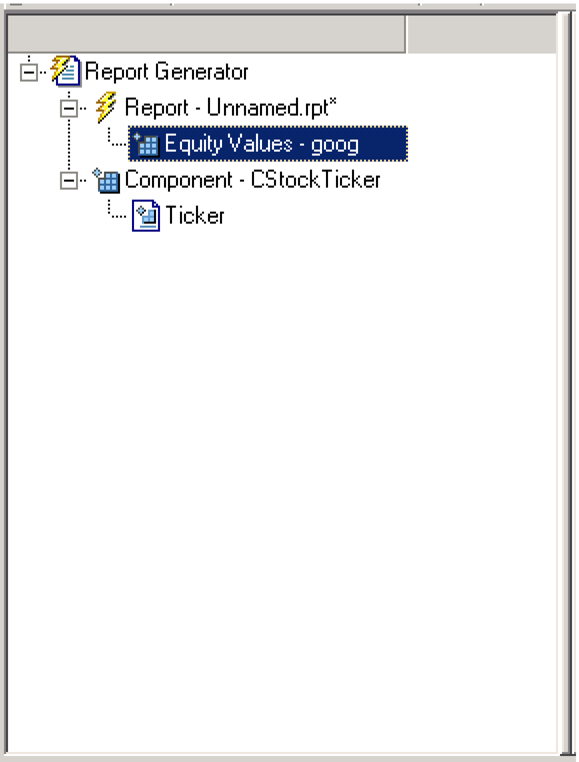
```
function out=execute(thisComp,parentDoc,varargin)
```

```
  stockQuote = fetch(GOOG, thisComp.Ticker);
  stockQuote.Date = datestr(stockQuote.Date,1);
  stockQuote.Time = datestr(stockQuote.Time,13);
  out = execute(rptgen.cfr_table(...
       'Source', stockQuote,...
       'numHeaderRows', 0,...
       'TableTitle', 'Stock Market Pricing Data'),...
      parentDoc);
```

**5** Append the security symbol, `goog`, to the component name. Enter the following text into `getOutlineString.m`.

```
function olstring=getOutlineString(thisComp)

olstring = [getName(thisComp),' - ',thisComp.Ticker];
```

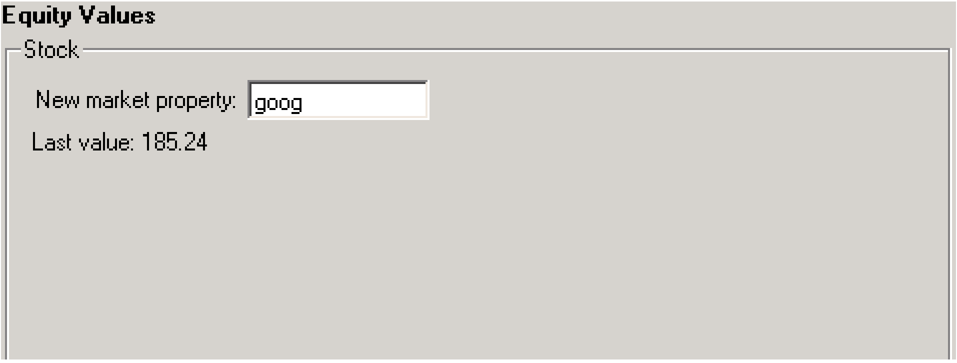
The component name now appears as `Equity Values   goog`.



6 Modify the `getdialogschema.m` file to change the appearance of the Properties pane. Enter the following text into this file to display the last quoted price for the security in the Properties pane.

```
function dlgStruct = getdialogschema(thisComp, name)

try
    currQuote = fetch(yahoo, thisComp.Ticker);
    quoteStr = sprintf('Last value: %g', currQuote.Last);
catch
    quoteStr = sprintf('Warning: ...
    "%s" is not a valid symbol.', thisComp.Ticker);
end

dlgStruct = thisComp.dlgMain(name,...
    thisComp.dlgContainer({
      thisComp.dlgWidget('Ticker',...
        'DialogRefresh',true,...
        'RowSpan',[1 1],'ColSpan',[1 1]);
      thisComp.dlgText(quoteStr,...
        'RowSpan',[2 2],'ColSpan',[1 1]);
    },'Stock',...
      'LayoutGrid',[3 2],...
      'RowStretch',[0 0 1],...
      'ColStretch',[0 1]));
```

The Properties pane for the component, `Equity Values`, now looks as follows.



Equity Values

Stock

New market property: goog

Last value: 185.24

7 Click **File > Report** to generate the report. The following output appears in the report.

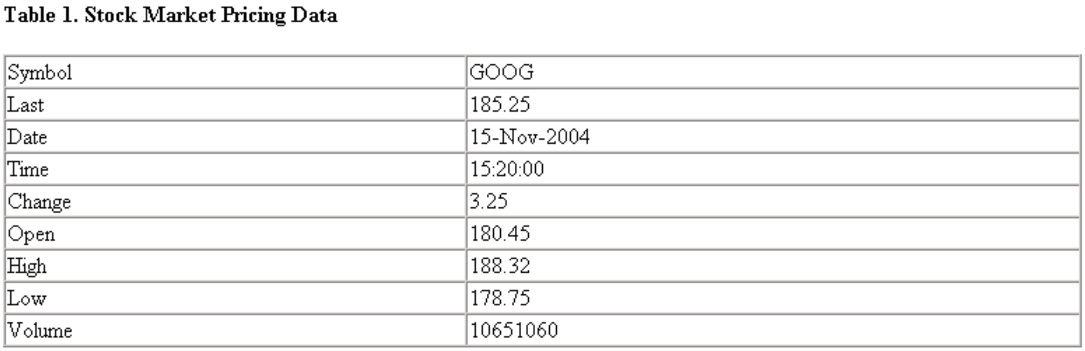


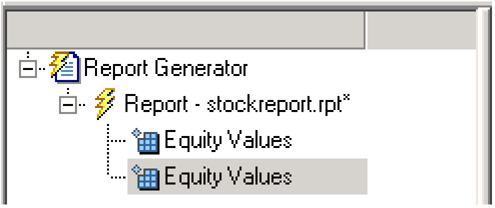
**Table 1. Stock Market Pricing Data**

| | |
|---|---|
| Symbol | GOOG |
| Last | 185.25 |
| Date | 15-Nov-2004 |
| Time | 15:20:00 |
| Change | 3.25 |
| Open | 180.45 |
| High | 188.32 |
| Low | 178.75 |
| Volume | 10651060 |

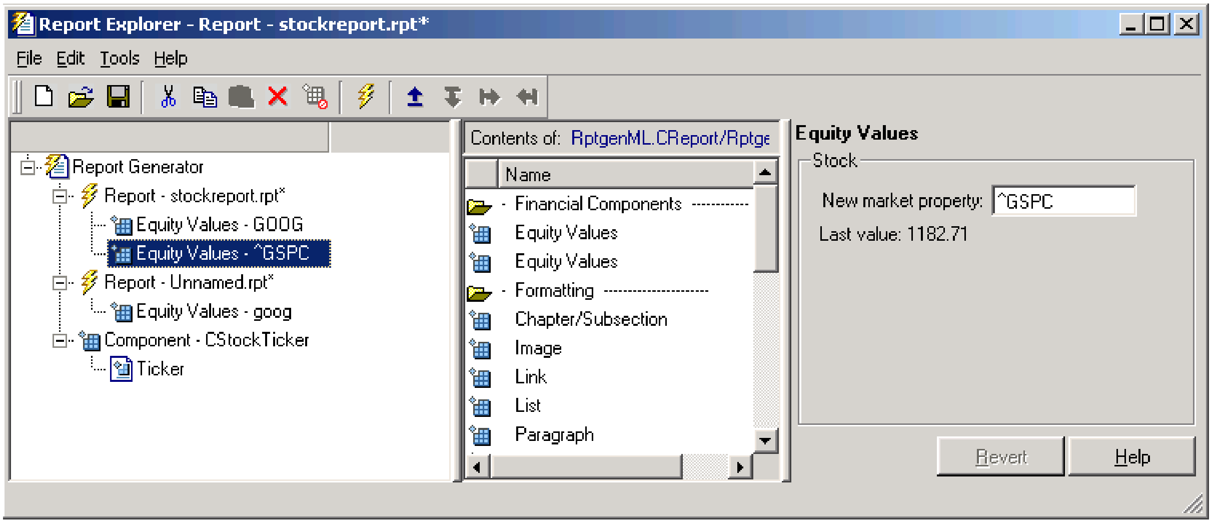
## Displaying Securities Data in Two Tables

This example, which shows how to use multiple properties within a component, expands upon “Fetching Securities Data and Displaying It in a Table” on page 6-21.

1 Create a report setup file and save it as `stockreport.rpt`. Add two `Equity Values` components to the setup file.



2 Edit the entry in the **New marker property** field to change the `ticker` property of the second component to `'^GSPC'` (S&P 500 index).



New market property: ^GSPC

Last value: 1182.71

**3** Run the report.

The report displays two tables of data, one for Google® and another for the S&P 500 index.

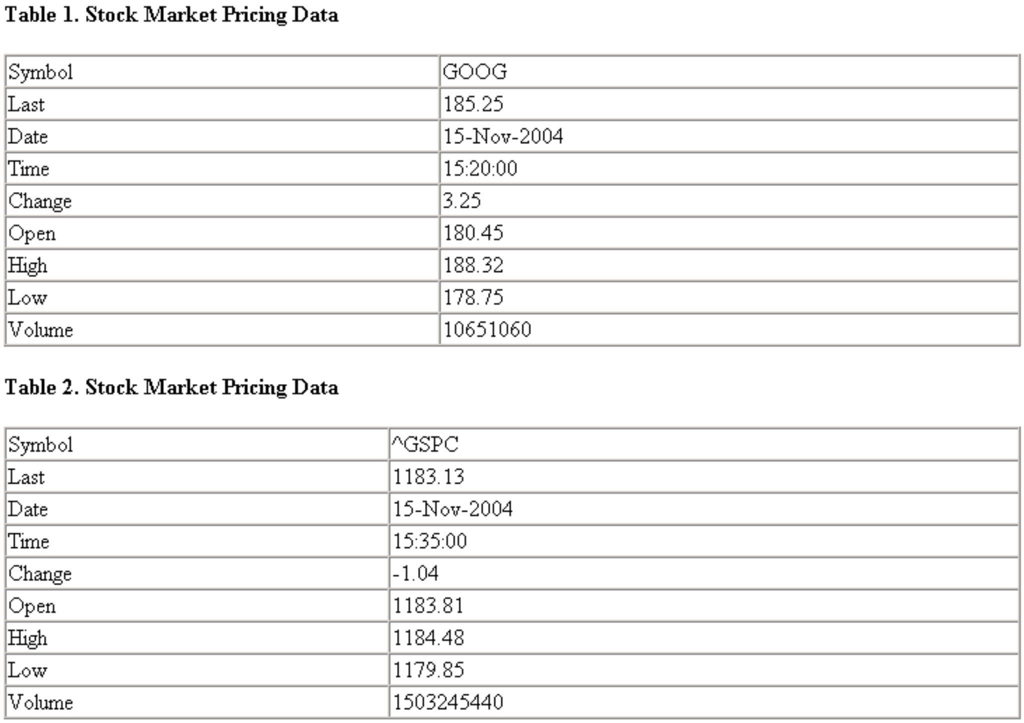


**Table 1. Stock Market Pricing Data**

| | |
|---|---|
| Symbol | GOOG |
| Last | 185.25 |
| Date | 15-Nov-2004 |
| Time | 15:20:00 |
| Change | 3.25 |
| Open | 180.45 |
| High | 188.32 |
| Low | 178.75 |
| Volume | 10651060 |

**Table 2. Stock Market Pricing Data**

| | |
|---|---|
| Symbol | ^GSPC |
| Last | 1183.13 |
| Date | 15-Nov-2004 |
| Time | 15:35:00 |
| Change | -1.04 |
| Open | 1183.81 |
| High | 1184.48 |
| Low | 1179.85 |
| Volume | 1503245440 |

# Create Custom Stylesheets

# Stylesheets

**In this section...**

## Built-In Versus Custom Stylesheets

*Stylesheets* specify formatting and display settings for reports. The report-generation process uses stylesheets to convert reports from DocBook XML format to a format that you specify. If you want to generate the given report in a different format than initially specified, you can convert the XML document using a different or modified stylesheet.

The following table lists report output formats and their default stylesheets.

| Report Format | Default Stylesheet |
|---|---|
| HTML | Uses stylesheets for either single- or multiple-page documents |
| PDF | Formatting Object (FO) stylesheet |
| RTF, Word | Document Style Semantics and Specification Language (DSSSL) stylesheet |

The following table shows a list of properties for the built-in stylesheets.

**Properties of Stylesheets**

| Name | Description |
|---|---|
| Description | A description of the stylesheet. |
| Display name | The stylesheet name that appears in the Options pane. |
| Transform type | The process used to generate reports that use a specified stylesheet. Supported types are:<br>• HTML<br>• FO (Formatting Object) for PDF reports<br>• DSSSL (Document Style Semantics and Specification Language) for RTF and Word reports<br>**Note** This field is not editable. |

In most cases, the stylesheets provided with the MATLAB Report Generator software should be more than adequate for your needs. However, you may want to modify the built-in stylesheets to meet special requirements. For example, suppose one of the built-in stylesheets meets your requirements, but you want to change the page orientation. You can create a custom stylesheet by editing the built-in stylesheet to your specifications.

## Customize Stylesheets Using Data Items

Each built-in stylesheet includes editable styles, also called *data items*, organized in categories. These data items specify styles that the file converter uses for a given report. You can edit these data items to customize stylesheets for your reports.

Data items can be of different types, some of which require different editing methods. For more information about editing data items, see “Edit Stylesheet Data Items” on page 7-10.

**Tip** See the **Help** area at the bottom of the Properties pane on the right for a description of a specific data item that you are editing.

# Create a New Stylesheet

To create a stylesheet:

1 Open the Report Explorer.

2 From the menu bar, click **Tools > Edit Stylesheet**.

3 In the Properties pane on the right, choose the built-in stylesheet for the format with which you want to work. Options are:

- **New HTML.** Creates a stylesheet for HTML reports.
- **New multi-page HTML.** Creates a stylesheet for HTML reports with more than one page.
- **New FO (PDF).** Creates a stylesheet for PDF reports.
- **New DSSSL (RTF).** Creates a stylesheet for RTF reports.

The new stylesheet appears in the Outline pane on the left.

4 In the Properties pane on the right, modify the properties for the stylesheet as needed. Add data items to the new stylesheet:

a Drag the data item you want to add from the Options pane in the middle to the stylesheet in the Outline pane on the left.

b In the Properties pane on the right, edit the data items for the selected style. For more information, see “Edit Stylesheet Data Items” on page 7-10

5 Save the stylesheet. For information about how to save a stylesheet, see “Save a Stylesheet” on page 7-9.

# Edit, Save, or Delete a Stylesheet

**In this section...**

## Edit a Stylesheet

To edit a stylesheet:

1. In Report Explorer, select a report setup file in the Outline pane on the left.

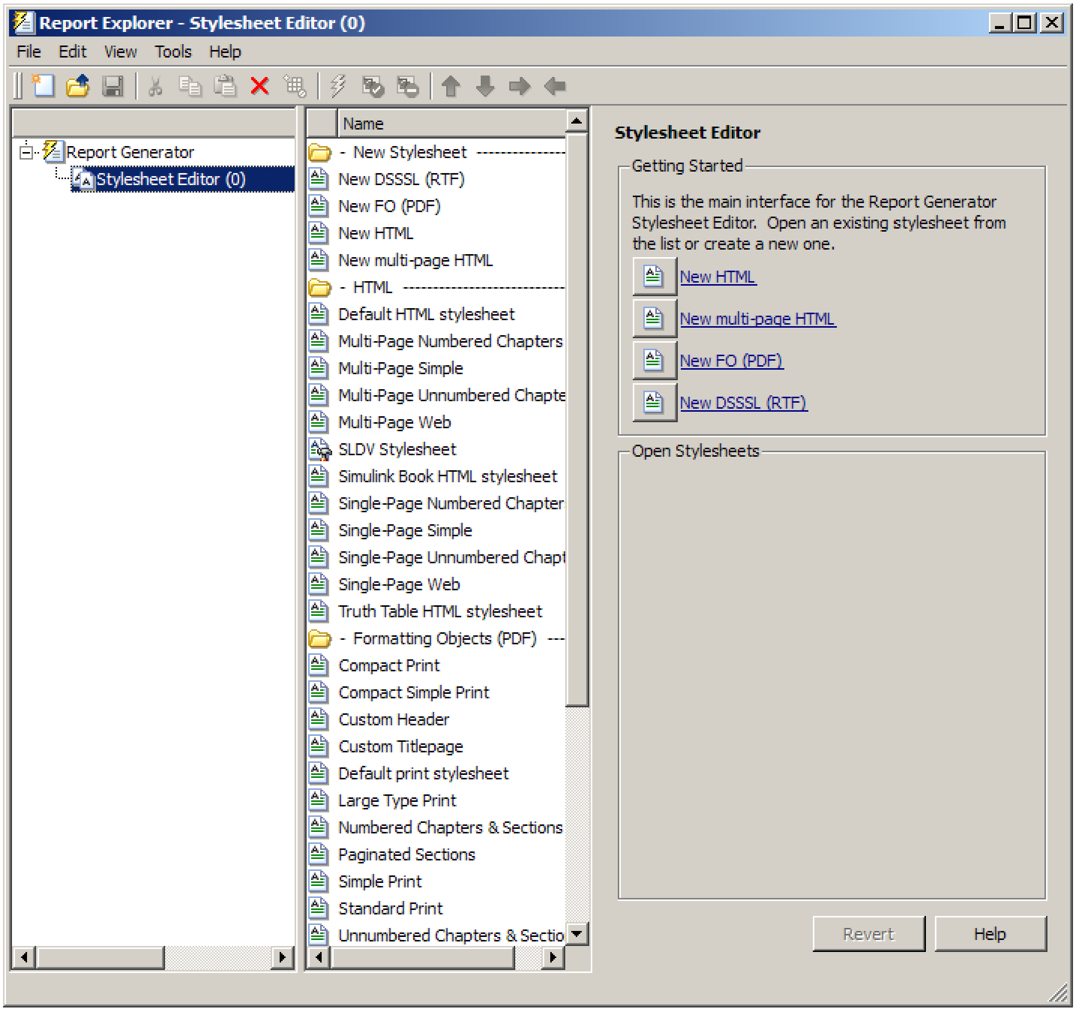
2. From the menu bar, click **Tools > Edit Stylesheet**.

The Report Explorer displays as follows.

- The Outline pane on the left displays the structure of stylesheets you create.
- The Options pane in the middle lists stylesheets available for customizing.

**Tip** Double-click a category to collapse it. Double-click it again to expand it.

- The Properties pane on the right shows properties of stylesheets, such as name and description.



- New Stylesheet ---------------
New DSSSL (RTF)
New FO (PDF)
New HTML
New multi-page HTML
- HTML ---------------------------
Default HTML stylesheet
Multi-Page Numbered Chapters
Multi-Page Simple
Multi-Page Unnumbered Chapte
Multi-Page Web
SLDV Stylesheet
Simulink Book HTML stylesheet
Single-Page Numbered Chapter
Single-Page Simple
Single-Page Unnumbered Chapt
Single-Page Web
Truth Table HTML stylesheet
- Formatting Objects (PDF) ---
Compact Print
Compact Simple Print
Custom Header
Custom Titlepage
Default print stylesheet
Large Type Print
Numbered Chapters & Sections
Paginated Sections
Simple Print
Standard Print
Unnumbered Chapters & Sectio

Stylesheet Editor

Getting Started

This is the main interface for the Report Generator Stylesheet Editor. Open an existing stylesheet from the list or create a new one.

New HTML
New multi-page HTML
New FO (PDF)
New DSSSL (RTF)

Open Stylesheets

You can use the Report Explorer to work with stylesheets as follows.

| Task | Pane to Use | Action |
|---|---|---|
| Create a stylesheet | Properties | Click the link that corresponds to the kind of stylesheet you want to create |
| Open an existing stylesheet | Properties | Click the name of the stylesheet, which appear in the **Open Stylesheets** area |
| Select a stylesheet to use for converting an XML source file | Options | Select a stylesheet by clicking on it |
| View a list of customized styles in a stylesheet | Outline | Expand any open stylesheet |
| View a list of styles in a stylesheet | Outline or Options | Double-click the stylesheet |
| View a list of stylesheets available for editing in a given category | Options | Double-click the folder that corresponds to the kind of output you want (that is, HTML, PDF, RTF, or Word) |
| View open stylesheets | Outline | Expand the Stylesheet Editor item in Report Explorer |
| Change the name or description of the current stylesheet | Properties | Edit the text in the **Display Name** or **Description** field. |
| Convert an XML source file using the current stylesheet | Properties | Click **Send to Source File Converter** in the Properties pane. |
| Edit customized style data | Properties | Click the style data item, which appears in the **Stylesheet Customizations** area |
| Open a style data item for editing or viewing | Options | Double-click the data item that you want to edit. |
| View a list of customized style data | Outline | Expand the stylesheet |

## Save a Stylesheet

You must save a stylesheet before you can use it to convert a source file or associate it with a report. To use the Report Explorer to save a stylesheet:

1. Select the stylesheet that you want to save in the Outline pane on the left.
2. Select **File > Save As** from the menu bar and specify a new name for the stylesheet (to avoid overwriting built-in stylesheets). You must save the file in a folder in your MATLAB path for the stylesheet to appear in the Report Explorer. The file name must be unique in the MATLAB path.

   By convention, MATLAB Report Generator stylesheets have `.rgs` as their file name extension.

## Delete a Stylesheet

To use the Report Explorer to delete a stylesheet that you created:

1. Select the stylesheet that you want to delete in the Outline pane on the left.
2. Click the stylesheet to delete from the Options pane in the middle.
3. Click **Delete stylesheet** in the stylesheet's Properties pane on the right.

You must restart the MATLAB software session for deleted stylesheets to disappear from the Options pane.

**Note** You cannot delete built-in stylesheets.

# Edit Stylesheet Data Items

**In this section...**

## Data Item Categories in Built-In Stylesheets

You can edit data items in built-in stylesheets to customize them. Data items appear in *categories*, according to their function. The following tables list the categories and data items for each type of stylesheet provided with the MATLAB Report Generator software.

### Categories of Styles in PDF (FO) Stylesheets

| Category | Description of Data Items in Category |
|---|---|
| Automatic labeling | Options for enumeration of parts of the report, such as chapters and sections |
| Callouts | Options and specifications related to callouts, such as defaults, use of graphics, size, path, fonts, characters, and extensions |
| Cross References | Option to control whether page numbers appear in report |
| Font Families | Specification of defaults for body text, copyright, quotes, symbols, dingbats, monospace, sans serif, and titles |
| Graphics | Specification of default width and options related to scaling attributes |
| Lists | Specification of spacing related to lists and list items |
| Meta/*Info | Options related to year ranges |

**Categories of Styles in PDF (FO) Stylesheets (Continued)**

| Category | Description of Data Items in Category |
|---|---|
| Miscellaneous | Options and specifications for placement of titles, comments, variable lists, block quotations, ulinks, hyphenations of URLs, verbatim environment display, use of SVG, table footnote numbers, superscript, and subscript |
| Pagination and General Styles | Specifications of page orientation, margins, double-sided, paper type, hyphenation, line height, columns, master font, draft mode, watermark, blank pages, rules for headers and footers, and content of headers and footers<br>**Note** You can specify parameters in this category, such as margin widths and header and footer height, in units of inches (`in`), millimeters (`mm`), or picas (`pi`), where 1 pica = 1/6 inch. |
| Property Sets | Specification and options related to figure titles, monospace properties, verbatim text, section titles, and levels of sections |
| Reference Pages | Option to control whether the class name is displayed |
| Stylesheet Extensions | Line numbering and table columns extensions |
| Table of Contents (TOC)/List of Tables (LOT)/Index Generation | Specifications for layout of TOC, depth of sections, indentation, and margins |
| Tables | Specifications for size of tables and their borders |
| Title Page | Specifications for positioning and transformation of title page elements and properties of title page text elements |

For information about DocBook XSL stylesheets, see http://docbook.sourceforge.net/release/xsl/current/doc/ .

You can set up font mappings for non-English PDF fonts. The PDF stylesheets override those mappings. For details, see

**Categories of Styles in HTML and Multi-Page HTML Stylesheets**

| Category of Style | Description of Data Items in Category |
|---|---|
| Automatic labeling | Options for enumeration of parts of the report, such as chapters and sections |
| Callouts | Options and specifications related to callouts, such as defaults, use of graphics, size, path, fonts, characters, and extensions |
| Chunking | Options related to using an explicit TOC for chunking, depth of section chunks, navigational graphics, and display of titles in headers and footers |
| Stylesheet Extensions | Line numbering, graphic size, and table columns extensions |
| Graphics | Specification of default width and depth, use of HTML embed for SVG, viewports, and options related to scaling attributes |
| HTML | Specifications related to dynamically served HTML, base and head elements, type of stylesheet, css, propagation of styles, longdesc, validation, cleanup, draft mode, watermark, and generation of abstract |
| Linking | Specification of Mailto URL and target for ulinks |
| Meta/*Info | Options related to year ranges |
| Miscellaneous | Options and specifications for comments, verbatim environment pixels, em space, use of SVG, and table footnote numbers |
| Reference Pages | Option control whether the class name is displayed |

### Categories of Styles in HTML and Multi-Page HTML Stylesheets (Continued)

| Category of Style | Description of Data Items in Category |
|---|---|
| Table of Contents (TOC)/List of Tables (LOT)/Index Generation | Specifications for layout of TOC, depth of sections, indentation, and margins |
| Tables | Specifications for size of tables, table cell spacing and padding, and borders |
| Title Page | Specifications for positioning and transformation of title page elements and properties of title page text elements |
| XSLT Processing | Options related to header and footer navigation and rules |

For information about:

- DocBook — see http://www.oasis-open.org/docbook/documentation/reference/html/docbook.html
- DocBook XSL stylesheets — see http://docbook.sourceforge.net/release/xsl/current/doc/
- DocBook print parameters, see http://docbook.sourceforge.net/release/dsssl/current/doc/print/

### Categories of Styles in RTF (DSSSL) Stylesheets

| Category of Style | Description of Data Items in Category |
|---|---|
| Admonitions | Options and path for admonition graphics |
| Backends | Options for Tex, MIF, and RTF back-end usage |
| Bibliographies | Options related to checking citations; suppressing, enumerating, and using titles of entries |
| Fonts | Specifications for font family and size to use for some elements |

**Categories of Styles in RTF (DSSSL) Stylesheets (Continued)**

| Category of Style | Description of Data Items in Category |
|---|---|
| Footnotes | Options for ulinks as footnotes and page location |
| Graphics | Specifications for file extensions, file names, and loading library database |
| Indents | Specifications for hanging indents, first paragraphs, and start of blocks |
| Labeling | Enumeration of sections and other elements |
| Miscellaneous | Options for floating formal objects, punctuation for run-in heads and honorifics, bold for first use of term, minimum leading between lines, and automatic hyphenation |
| OLinks | Using an extension for finding outline information |
| Object Rules | Specifications for placement and width of rules |
| Paper/Page Characteristics | Specifications for paper type, page numbers, width of pages, margins, and columns; heading-levels, sides; and writing mode (such as left-to-right) |
| Quadding | Specifications for justifying paragraphs |
| RefEntries and Functions | Options related to generation and display of reference entries and synopses for functions |
| Running heads | Options for generating and displaying running heads of chapters |
| Table of Contents (TOC)/List of Tables (LOT) | Options to produce or display TOC for sets, books, parts, references, articles. Options to display TOC on title page |
| Tables | Specification of width in simple list |
| VariableLists | Options and specifications for term length and formatting |

**Categories of Styles in RTF (DSSSL) Stylesheets (Continued)**

| Category of Style | Description of Data Items in Category |
|---|---|
| Verbatim Environments | Specifications for width, enumeration, size, indentation, line frequency, and callouts |
| Vertical Spacing | Specifications for space between lines and paragraphs |

## Edit Data Items in Simple or Advanced Edit Mode

- To edit a data item in *simple edit mode*, edit a simple string that corresponds to the data in the stylesheet. This string appears in the field to the right of the **Value** label. For some values, use a selection list to change the value instead of typing in text.
- To edit a data item in *advanced edit mode*, edit the XML code directly.

**Note** This section gives instructions for simple edit mode, except where explicitly specified otherwise.

The user interface is in simple edit mode when the data item appears in a pane labeled **Value**. It is in advanced edit mode when the data item appears in a pane labeled **Value (XML)**. To switch from simple to advanced edit mode, click **Edit as XML**.

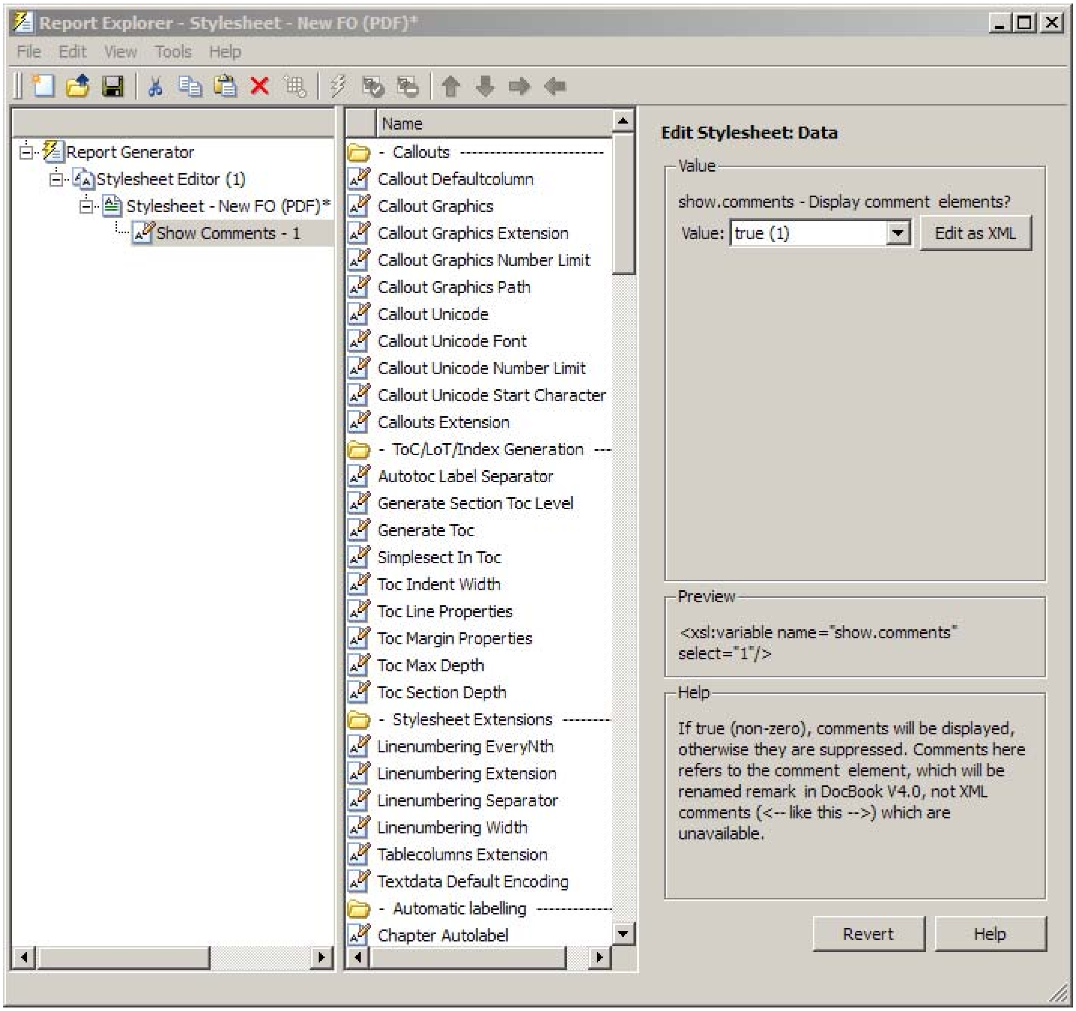
Edit values for most data items in PDF and HTML stylesheets in either simple edit mode or advanced edit mode. Edit values for RTF stylesheets in simple edit mode only. Data items in RTF stylesheets do not support advanced edit mode.

**Note** To modify content for headers and footers you edit *stylesheet cells*, which do not appear in either simple or advanced mode. For more information, see “Stylesheet Cells for Headers and Footers” on page 7-25.

## Data Items

Select a stylesheet from the Options pane in the middle of the Report Explorer. The Outline pane on the left shows the name of the current style data item inside its stylesheet. The Options pane in the middle shows a list of available stylesheet data items. The Properties pane on the right displays **Stylesheet Editor: Data**. It also includes the following information:

- The value of the data item is in a pane labeled **Value** in simple edit mode or **Value (XML)** in advanced edit mode.
- To the right of the value is the **Edit as XML** toggle button.
- The **Preview** pane includes a partial view of the stylesheet that specifies the data item. The data in this pane is not editable.
- The **Help** pane contains information about the data item. This information is not editable.



Edit Stylesheet: Data

Value

show.comments - Display comment elements?

Value: true (1)

Preview

<xsl:variable name="show.comments" select="1"/>

Help

If true (non-zero), comments will be displayed, otherwise they are suppressed. Comments here refers to the comment element, which will be renamed remark in DocBook V4.0, not XML comments (<-- like this -->) which are unavailable.

## Edit Boolean and Enumerated Values

In the previous figure, theShow Comments data item is of type Boolean. Its current value is true(1). Change this value using the menu list for the value field. In this case, the only other possible value is false(2).

### Edit Strings

For the values of some data items, the Report Explorer displays text in the editable **Value** field. You can specify an XML expression, though you are not required to do so.

### Edit XML Expressions

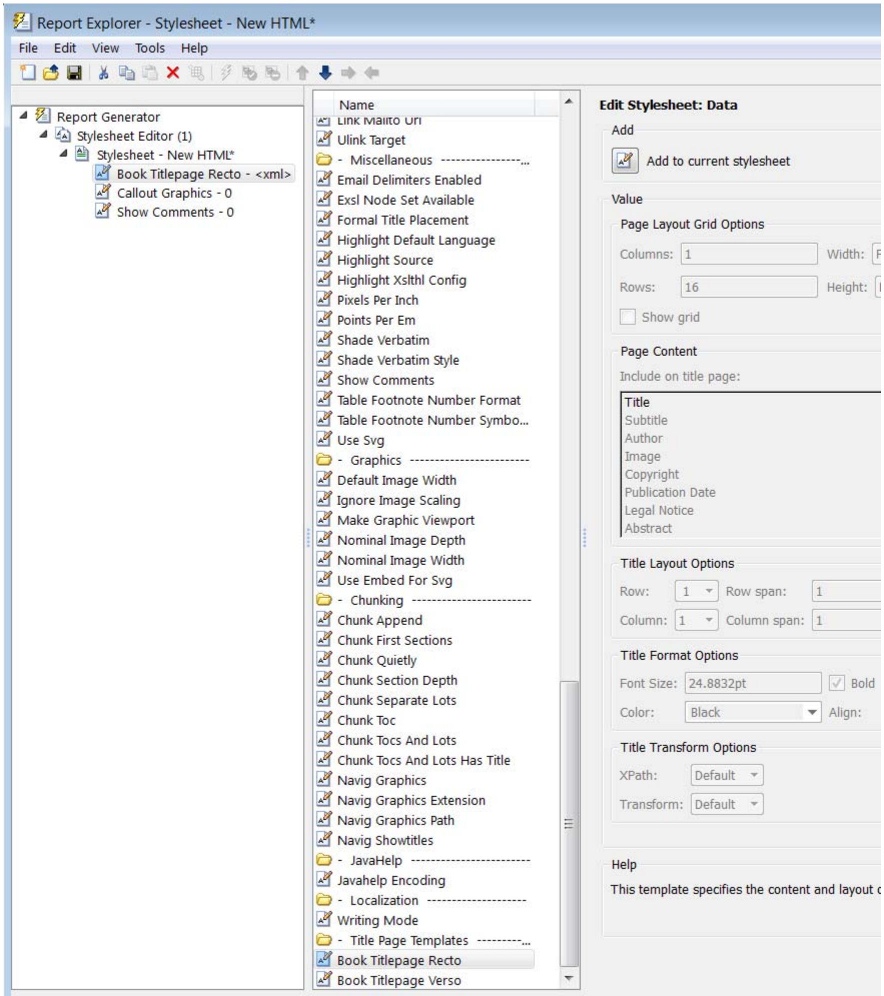
To make complex changes to a stylesheet, consider using Advanced edit mode. This enables you to edit XML expressions directly in the **Value (XML)** pane. If this pane does not appear, click **Edit as XML** to switch to advanced edit mode.

Make sure that you enter valid XML. Invalid XML values generate an error, which appears at the top of the Properties pane.

### Modify Title Page Properties

For PDF or HTML stylesheets, you can modify the layout, contents, and format of a title page by using the Stylesheet Editor.

1. In the Outline pane, select the stylesheet you want to edit.
2. In the Options pane, in the **Title Page Templates** section, select:
   - `Book Titlepage Recto` to specify properties for the front side of the title page
   - `Book Titlepage Verso` to specify properties for the back side of the title page
3. In the Properties pane, select **Add to current stylesheet** and edit the properties.



Columns: 1 Width: F

Rows: 16 Height: I

□ Show grid

Include on title page:

Title
Subtitle
Author
Image
Copyright
Publication Date
Legal Notice
Abstract

Row: 1 Row span: 1

Column: 1 Column span: 1

Font Size: 24.8832pt ☑ Bold

Color: Black Align:

XPath: Default

Transform: Default

This template specifies the content and layout c

To adjust the grid used to position the title page elements (such as the title and author) on the page, in the Properties pane specify:

- **Columns** — The number of columns in the page grid
- **Width** — The width of each column
- **Rows** — The number of rows in the page grid

- **Width** — The width of each row

To view the grid layout on the generated title page, select **Show grid**.

By default, all of the title page elements appear on the title page. To exclude display of a title page element:

1. In the Properties pane, in the **Include on title page** list, select an element to exclude.
2. Click the right arrow button. The element appears in the **Exclude from Title Page** list.

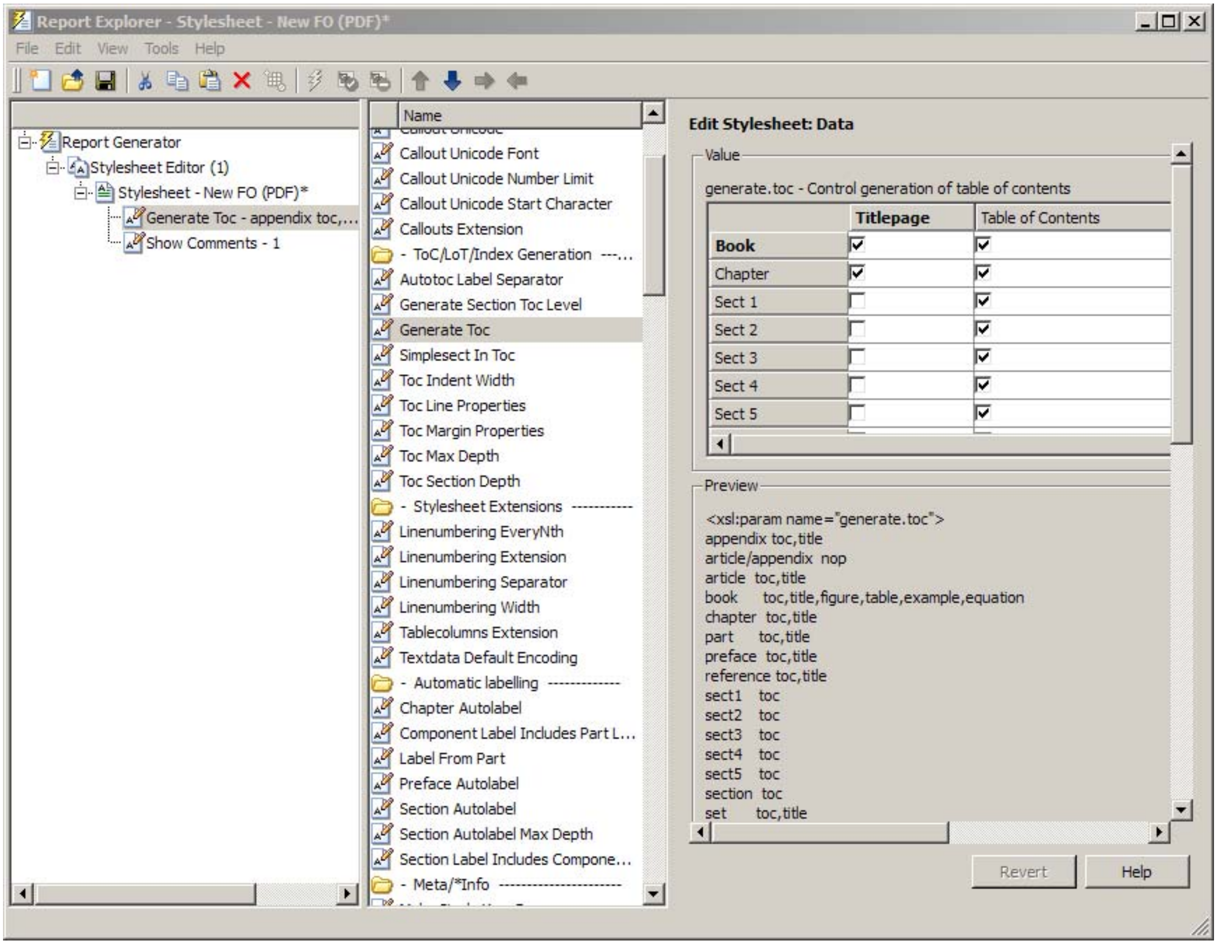
To specify properties for an individual title page element:

1. In Properties pane, in the **Include on title page** list, select the title page element.
2. Adjust the applicable properties:
   - Layout options — Specify which title page grid row and column in which you which you want the element to appear. To span multiple rows or columns, specify numbers for the **Span row** and **Span column** properties.
   - Format options — For text elements, specify the font size, whether to use bold or italics, text color, and text alignment.
   - Transform options — You can use these options to specify XSLT code to customize the contents and format of title page elements. Use the **XPath** property to specify the path to the XSLT object that you want to modify. Use the **Transform** property to specify the custom content and layout.

### Modify TOC Properties

To change values for generation of the report's table of contents (TOC), select the appropriate values from a matrix of check boxes.

The following figure shows the values for the **Generate Toc** data item on the **PDF** stylesheet. Select the check boxes to control the values that appear in the report's title page and table of contents.


Report Explorer - Stylesheet - New FO (PDF)*
File Edit View Tools Help
Report Generator
Stylesheet Editor (1)
Stylesheet - New FO (PDF)*
Generate Toc - appendix toc,...
Show Comments - 1
Name
Callout Unicode Font
Callout Unicode Number Limit
Callout Unicode Start Character
Callouts Extension
- ToC/LoT/Index Generation ---...
Autotoc Label Separator
Generate Section Toc Level
Generate Toc
Simplesect In Toc
Toc Indent Width
Toc Line Properties
Toc Margin Properties
Toc Max Depth
Toc Section Depth
- Stylesheet Extensions -----------
Linenumbering EveryNth
Linenumbering Extension
Linenumbering Separator
Linenumbering Width
Tablecolumns Extension
Textdata Default Encoding
- Automatic labelling ------------
Chapter Autolabel
Component Label Includes Part L...
Label From Part
Preface Autolabel
Section Autolabel
Section Autolabel Max Depth
Section Label Includes Compone...
- Meta/*Info --------------------
Edit Stylesheet: Data
Value
Preview
Revert
Help

generate.toc - Control generation of table of contents

| | Titlepage | Table of Contents |
|---|---|---|
| **Book** | ☑ | ☑ |
| Chapter | ☑ | ☑ |
| Sect 1 | □ | ☑ |
| Sect 2 | □ | ☑ |
| Sect 3 | □ | ☑ |
| Sect 4 | □ | ☑ |
| Sect 5 | □ | ☑ |

```
<xsl:param name="generate.toc">
appendix toc,title
article/appendix  nop
article  toc,title
book     toc,title,figure,table,example,equation
chapter  toc,title
part     toc,title
preface  toc,title
reference toc,title
sect1    toc
sect2    toc
sect3    toc
sect4    toc
sect5    toc
section  toc
set      toc,title
```

## Modify Title Placement Properties

The **Title Placement** data items, which are in the **Miscellaneous** category, control the position of titles for figures and tables.

Selecting one of these data items for editing causes the Properties pane on the right to display possible values in a menu list. Specify whether you want the title to appear before or after a given figure or table.

## Modify Attributes

An *attribute* is a data item that specifies information for an XML element. An attribute must be a child of an *attribute set*. For more information, see "Edit Attribute Sets" on page 7-22.

---

**Note** The information in the **Help** area of the Properties pane of an attribute describes the set to which the attribute belongs.

---

## Edit Attribute Sets

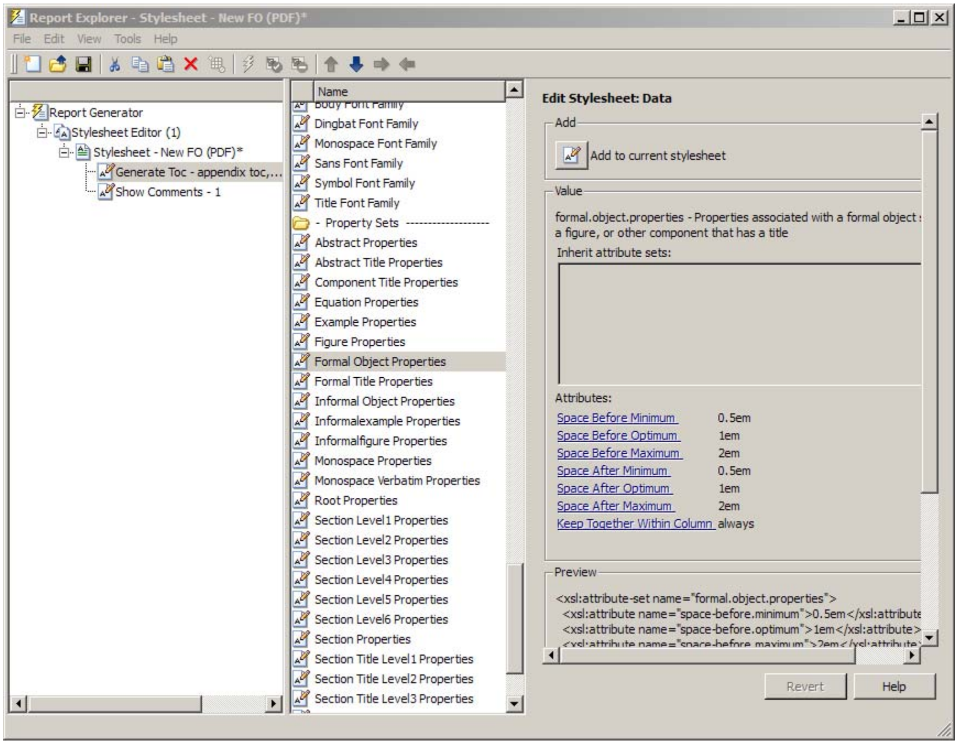
An *attribute set* consists of a group of attributes. Selecting an attribute set in the Outline pane on the left causes the Properties pane to list the attributes that belong to that set.

To edit a specific attribute, expand the attribute set in the Outline pane and select the attribute you want to edit.

To edit the attribute set, type `text` in the **Inherit attribute sets** area of the Properties pane.

An example of an attribute set is `Formal Object Properties`, a data item in the `Property Sets` category of the default print stylesheet for PDF documents.

Here is an example of the Report Explorer showing the `Formal Object Properties` attribute set in the Properties pane.


Report Explorer - Stylesheet - New FO (PDF)*
File Edit View Tools Help
Report Generator
Stylesheet Editor (1)
Stylesheet - New FO (PDF)*
Generate Toc - appendix toc,...
Show Comments - 1
Name
Body Font Family
Dingbat Font Family
Monospace Font Family
Sans Font Family
Symbol Font Family
Title Font Family
- Property Sets ------------------
Abstract Properties
Abstract Title Properties
Component Title Properties
Equation Properties
Example Properties
Figure Properties
Formal Object Properties
Formal Title Properties
Informal Object Properties
Informalexample Properties
Informalfigure Properties
Monospace Properties
Monospace Verbatim Properties
Root Properties
Section Level1 Properties
Section Level2 Properties
Section Level3 Properties
Section Level4 Properties
Section Level5 Properties
Section Level6 Properties
Section Properties
Section Title Level1 Properties
Section Title Level2 Properties
Section Title Level3 Properties
Edit Stylesheet: Data
Add
Add to current stylesheet
Value
Revert
Help

formal.object.properties - Properties associated with a formal object :
a figure, or other component that has a title

Inherit attribute sets:

Attributes:

| | |
|---|---|
| Space Before Minimum | 0.5em |
| Space Before Optimum | 1em |
| Space Before Maximum | 2em |
| Space After Minimum | 0.5em |
| Space After Optimum | 1em |
| Space After Maximum | 2em |
| Keep Together Within Column | always |

Preview

```
<xsl:attribute-set name="formal.object.properties">
  <xsl:attribute name="space-before.minimum">0.5em</xsl:attribute
  <xsl:attribute name="space-before.optimum">1em</xsl:attribute>
  <xsl:attribute name="space-before.maximum">2em</xsl:attribute>
```

## Edit Varpair Values

Data items in RTF stylesheets appear as `varpair` data items, which are name/value pairs of information. RTF stylesheets are the only type of stylesheet that includes `varpair` data items.

Edit `varpair` data items as strings or as Boolean values. Boolean values appear as `true (#t)` and `false (#f)`.

**Note** You cannot edit RTF stylesheet data items as XML.

**Note** Data of type `varpair` is sometimes represented in stylesheets as DSSSL rather than XML. As a result, the code that appears in the **Preview** pane of the Properties pane on the right looks different from code associated with other kinds of MATLAB Report Generator stylesheets.

### Delete Data Items

To delete a customized data item:

1. Right-click the data item in the Outline pane on the left.
2. Select **Delete**.

# Stylesheet Cells for Headers and Footers

**In this section...**

## About Stylesheet Cells and Cell Groups

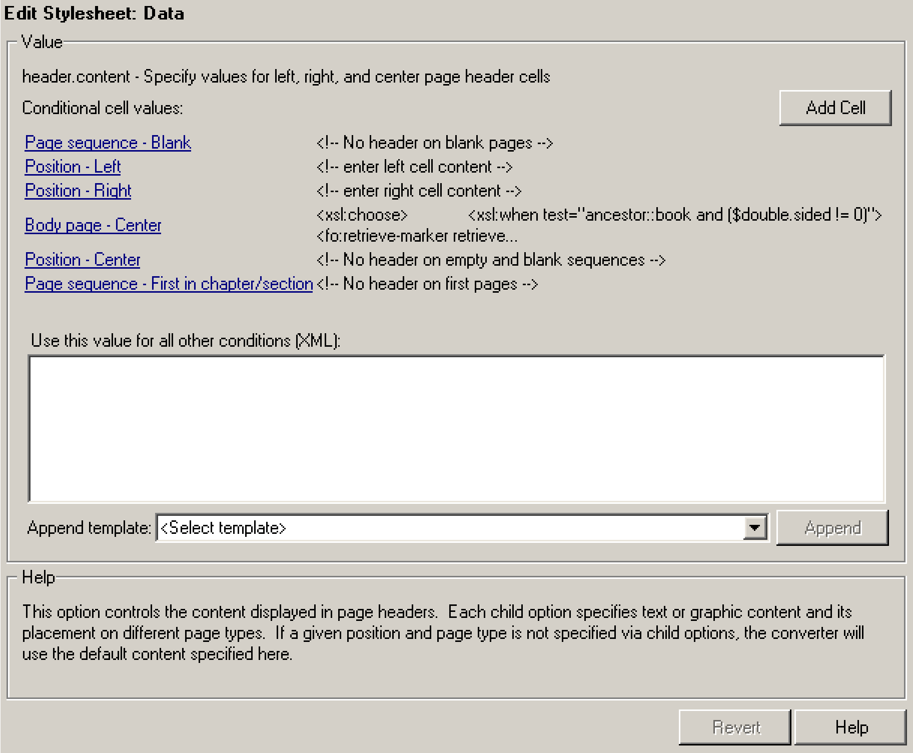
Use *stylesheet cells* to specify content of headers and footers in PDF reports.

The MATLAB Report Generator software defines a page as six *cells*. These cells correspond to the left, right, and center of the page's header and the left, right, and center of the page's footer.

A *cell group* consists of one or more stylesheet cells. Two cell groups are available for PDF reports: **Header Content** and **Footer Content**.

The Properties pane for each cell in a cell group lists the group's current stylesheet cell definitions. These definitions appear in a two-column list of **Conditional cell values**. The first column displays the name of a *condition*. The second column displays *content* that appears in the report if the specified condition is met.

For example, the stylesheet cell `Page sequence - Blank` specifies the content for a blank page; by default, the content is empty. Similarly, `Cell - Right Side` specifies the content for the right side of the header on every page.



Edit Stylesheet: Data

Value

header.content - Specify values for left, right, and center page header cells

Conditional cell values:

| | |
|---|---|
| Page sequence - Blank | <!-- No header on blank pages --> |
| Position - Left | <!-- enter left cell content --> |
| Position - Right | <!-- enter right cell content --> |
| Body page - Center | <xsl:choose> <xsl:when test="ancestor::book and ($double.sided != 0)"><br><fo:retrieve-marker retrieve... |
| Position - Center | <!-- No header on empty and blank sequences --> |
| Page sequence - First in chapter/section | <!-- No header on first pages --> |

Use this value for all other conditions (XML):

Append template: <Select template>

Help

This option controls the content displayed in page headers. Each child option specifies text or graphic content and its placement on different page types. If a given position and page type is not specified via child options, the converter will use the default content specified here.

You can use many combinations of conditions and values to customize content of headers and footers. The MATLAB Report Generator software provides several predefined conditions that are frequently used. These predefined cells appear in the Properties panes for the Header Content and for Footer Content cell groups.

## Headers and Footers

### Add Content That Satisfies Specified Conditions

You can use the Properties pane of a stylesheet cell to specify content that satisfies specified conditions. The Properties pane for a stylesheet cell includes the following.

| Label | Definition | Description |
|---|---|---|
| **Condition** | Condition that must be met for content to appear in the report | This is a selection list of frequently used and predefined conditions. Select a condition and click **Edit** to view or change a condition's XML code |
| **Value (XML)** | Content to appear in the report if the condition is met | Modify or create XML code for header or footer content |
| **Append Template** | Name of the template that you use to add content | Templates containing XML code that you can use to add content. For more information, see "Add Content to Headers and Footers Using Templates" on page 7-28. |

When the File Converter processes a page, it evaluates settings that are relevant to each of the six cells on the page and adds content accordingly. If there are no conditions in effect for a given cell, the File Converter uses the default values for the cell group.

Possible conditions and their values as coded in XML are shown in the following table.

| Name of Condition | Possible Values for the Condition | Sample XML Code |
|---|---|---|
| `$position` | `right`<br>`center`<br>`left` | `$position='right'`<br>`$position='center'`<br>`$position='left'` |
| `$sequence` | `odd`<br>`even`<br>`first`<br>`blank` | `$sequence=odd`<br>`$sequence=even`<br>`$sequence=first`<br>`$sequence=blank` |
| `$double-sided` | `0`<br>`1` | `$double-sided=0`<br>`$double-sided=1` |
| `$pageclass` | `$titlepage`<br>`$lot`<br>`$body` | `$pageclass=$titlepage`<br>`$pageclass=$lot`<br>`$pageclass=$body` |

Use standard logical operators (such as `=` , `!=` , `and`, `or`) and nested expressions (characters between parentheses are an expression within an expression) to specify *complex conditions*. You can use complex conditions to set the position of headers and footers on pages. You can also use them to specify other settings, such as where in the report the content appears.

## Add Content to Headers and Footers Using Templates

*Templates* are available for adding the following items to headers and footers:

- Text
- Author names
- Page numbers
- Titles for chapters and sections

- Chapter numbering
- Draft information
- Comments
- Graphics

Templates used by the File Converter are Extensible Style Language Transformations (XSLT), which is a language for transforming XML documents into other XML documents. For details about XSLT, see the Web site for the World Wide Web Consortium (W3C®) at `http://www.w3.org/TR/xslt`.

To use a template to specify content for a header or footer:

1. In the **Append template** list, select the type of content you want to add.
2. Click **Append**.

   The Properties pane on the right displays default content for the type you select. Edit the XML code to change the default content.

For example, to specify `text` as the content:

1. Select **Text** from the **Append template** list.
2. Click **Append**.
3. The default value for `xsl:text` is `Confidential`. Edit the value as needed.

## Insert Graphics Files

To add a graphics file to headers or footers in a report, you must:

1. Specify the name of the file in the **Header Content** or **Footer Content** stylesheet cell.
2. Edit the values of the `Region Before Extent` and `Region After Extent` data items. These are located in the **Pagination and General Styles** folder of the **Options** pane for PDF formatting.

For an example of adding a graphic file to a header, see “Add Graphics to Headers in PDF Reports” on page 7-32.

**Note** PDF reports only support bitmap (`.bmp`), jpeg (`.jpg`), and Scalable Vector Graphics (`.svg`) images in headers and footers.

## Modify Fonts and Other Properties

You cannot use stylesheet cells to modify the font family or other such properties of headers and footers. To specify the style of the content in headers and footers, use the **Header Content Properties** and **Footer Content Properties** attribute sets.

Each of these attribute sets is a pagination style data item for PDF stylesheets. You can edit a particular attribute in the set by selecting it in the Outline pane on the left.

For an example of modifying font size and other properties of a PDF report, see “Change Font Size, Page Orientation, and Paper Type of a Generated Report” on page 7-37.

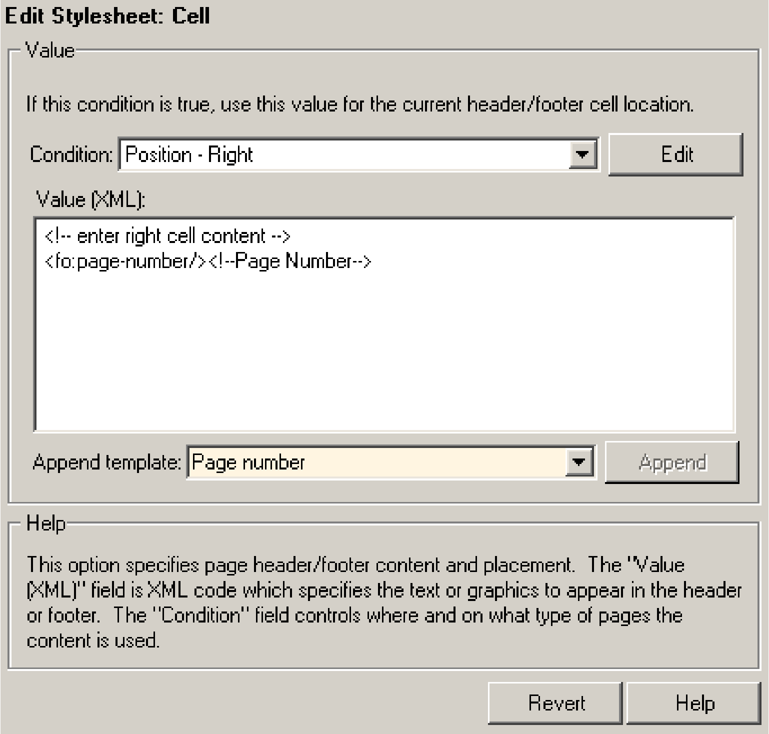
# Customized Stylesheets

**In this section...**

## Number Pages in a Report

This example shows how to edit a stylesheet cell to number the upper-right side of all pages in the generated report.

1. Define a basic stylesheet cell in the Header Content cell group with a condition of `right`.
   a. Open a PDF stylesheet in the Report Explorer.
   b. Double-click **Header Content** (under **Pagination and General Styles**) in the Options pane in the middle.
   c. Click **Position - Right** in the Properties pane on the right.
2. Set the header content to the current page number by selecting `Page number` from the **Append template** selection list.



Edit Stylesheet: Cell

Value

If this condition is true, use this value for the current header/footer cell location.

Condition: Position - Right

Value (XML):

```
<!-- enter right cell content -->
<fo:page-number/><!--Page Number-->
```

Append template: Page number

Help

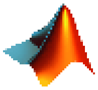
This option specifies page header/footer content and placement. The "Value (XML)" field is XML code which specifies the text or graphics to appear in the header or footer. The "Condition" field controls where and on what type of pages the content is used.

3 Click **Append**.

## Add Graphics to Headers in PDF Reports

This example shows how to include an image in the center of the header of each page in a PDF report, excluding the report's title page and the first page of each chapter. You do this by editing default header content for a PDF stylesheet. This example uses the report setup file `mfile-report.rpt`.

You can use any bitmap or jpeg file as image content. You must know the size of the image so that you can allow enough room for it in the header. This example uses the `sample_logo.bmp` image, which is shown here.



**Note** PDF reports only support bitmap (`.bmp`), jpeg (`.jpg`), and Scalable Vector Graphics (`.svg`) images.

To include this image file in the center of each header in the body of a PDF report:

1. Open `mfile-report.rpt` by entering the following at the MATLAB command prompt:

   ```
   setedit mfile-report
   ```

2. Create a custom stylesheet.

   a. Select **Tools > Edit Stylesheet** in the menu bar of the Report Explorer.

   b. Click **New FO (PDF)** in the Properties pane on the right.

   c. As the **Display name**, enter `Logo stylesheet for PDF`.

   d. As **Description**, enter `Company logo in center of header`.

   e. Save the stylesheet as `logo_stylesheet.rgs` in a folder on your MATLAB path.

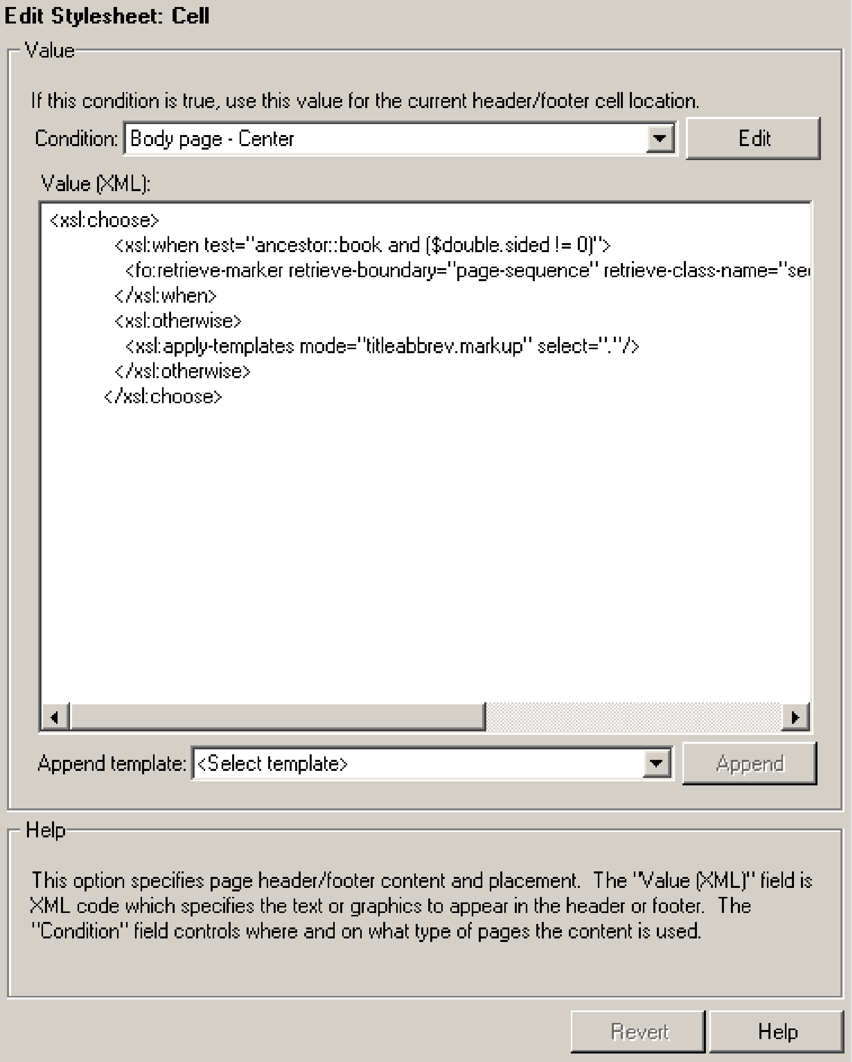
3. Open the cell group for editing.

   a. Scroll through the Options pane on the left to the **Pagination and General Styles** folder.

   b. Double-click **Header Content** in the Options pane.

   c. Click **Body page – Center** from the list of cells in the Properties pane on the right.

      The Properties pane appears as shown.



If this condition is true, use this value for the current header/footer cell location.

Condition: Body page - Center

Value (XML):

```
<xsl:choose>
          <xsl:when test="ancestor::book and ($double.sided != 0)">
            <fo:retrieve-marker retrieve-boundary="page-sequence" retrieve-class-name="se
          </xsl:when>
          <xsl:otherwise>
            <xsl:apply-templates mode="titleabbrev.markup" select="."/>
          </xsl:otherwise>
        </xsl:choose>
```

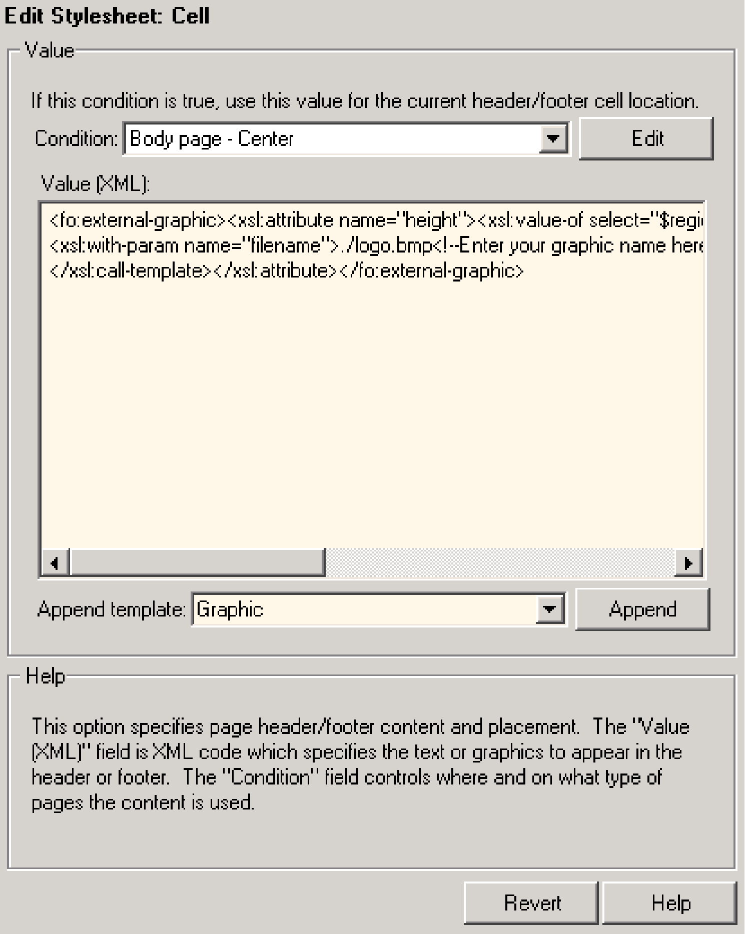
Append template: <Select template>

This option specifies page header/footer content and placement. The "Value (XML)" field is XML code which specifies the text or graphics to appear in the header or footer. The "Condition" field controls where and on what type of pages the content is used.

**d** Delete the text in the **Value (XML)** field.

**e** Select **Graphic** from the **Append template** selection list and click **Append**.

The Properties pane on the right shows the XML code that tells the File Converter to include the graphic.



**Edit Stylesheet: Cell**

Value

If this condition is true, use this value for the current header/footer cell location.

Condition: Body page - Center

Value (XML):

```
<fo:external-graphic><xsl:attribute name="height"><xsl:value-of select="$regi
<xsl:with-param name="filename">./logo.bmp<!--Enter your graphic name here
</xsl:call-template></xsl:attribute></fo:external-graphic>
```

Append template: Graphic

Help

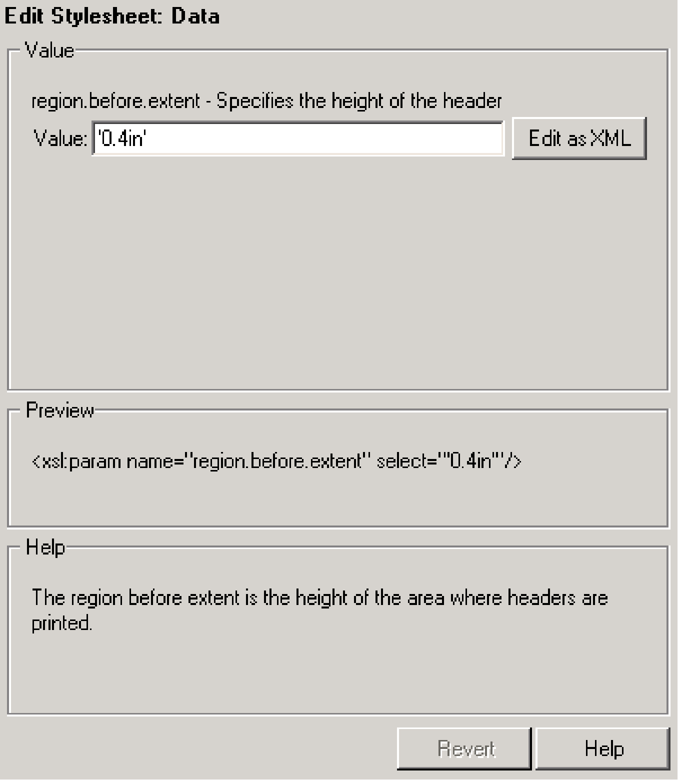
This option specifies page header/footer content and placement. The "Value (XML)" field is XML code which specifies the text or graphics to appear in the header or footer. The "Condition" field controls where and on what type of pages the content is used.

**4** By default, the name of the graphic is `logo.bmp`. Change all instances of this name to `sample_logo.bmp` in the **Value (XML)** field.

**5** Save the stylesheet.

**6** Make sure that the amount of room available in the header is large enough to accommodate the image file.

**a** In the Options pane in the middle, double-click **Region Before Extent**, which is in the **Pagination and General Styles** folder.



Edit Stylesheet: Data

Value

region.before.extent - Specifies the height of the header

Value: '0.4in'

Preview

<xsl:param name="region.before.extent" select="'0.4in'"/>

Help

The region before extent is the height of the area where headers are printed.

**b** By default the value for the height of the header is `0.4` inch. Replace this value with `1.0in`.

**c** Save the stylesheet.

**7** Generate the report with the new styles.

**a** Select **mfile-report.rpt** in the Outline pane on the left.

**b** In the selection lists under the **Report Format and Stylesheet** area of the Properties pane on the right:

- Specify `Acrobat (PDF)` for **File format**
- Specify `Logo stylesheet for PDF`.

**c** Click **Report** on the toolbar to generate the report.

## Change Font Size, Page Orientation, and Paper Type of a Generated Report

This example shows how to:

- Generate an XML source file without converting it to a supported report format
- Make section headers in a report larger
- Change the report page orientation to landscape
- Change the report paper type to A4

Create a custom stylesheet by editing an existing stylesheet to change the appearance of the `wsvar-report` report, which is provided with the MATLAB Report Generator software.

1 Generate a source file for the report.

   a Open the report by entering the following command in the MATLAB Command Window:

   ```
   setedit wsvar-report
   ```

   b In the **Report Format and Stylesheet** area of the Properties pane, change the format to **DocBook (no transform)**.

   c Check the **If report already exists, increment to prevent overwriting** check box.

   d Select **File > Report** to generate the report.

   The report-generation process creates an XML source file in the MATLAB Editor.

2 Convert the report to PDF format.

   a Select **Tools > Convert Source File** from the Report Explorer menu bar to open the File Converter.

   b From the **Source file** selection list, enter **wsvar-report0.xml**.

   c From the **File format** selection list, select **Acrobat (PDF)**.

d From the **Stylesheet** selection list, select **Unnumbered Chapters and Sections**.

e Click **Convert File**.

The MATLAB Report Generator software converts the XML source file for `wsvar-report` to PDF format, and then opens the PDF document.

3 Make the report headers more prominent.

a In the File Converter, click **Edit**.

The Report Explorer displays the **Unnumbered Chapters and Sections** stylesheet.

b In the Properties pane on the right, enter `Custom Large Section Headers` as the stylesheet name.

c Enter the description `No chapter and section numbering, larger section titles`.

d In the Outline pane on the left, select the **Custom Large Section Headers** stylesheet.

e In the Options pane in the middle, select **Section Title Level 1 Properties**.

f In the Properties pane on the right, click **Add to current stylesheet**.

The **Section Title Level 1 Properties** data item appears in the Outline pane on the left as a child of the `Custom Large Section Headers` stylesheet.

g In the Properties pane on the right, select the **Font Size** attribute.

The Properties pane on the right displays an XML expression specifying font size as a multiple of the Body Font Size attribute.

h Click **Edit as string**.

The MATLAB Report Generator software converts the XML expression to a simple string, which appears in a pane labeled **Value**.

i Enter the value `18pt`.

The size of the font is now fixed at 18 points, rather than being a multiple of the body font size attribute.

- **j** Select **File > Save** to save the stylesheet.
- **k** Save the stylesheet as `customheader.rgs`, in a folder in your MATLAB path.

  The `customheader.rgs` stylesheet appears as an available stylesheet in the Options pane in the middle of the Report Explorer. It also appears as an option in the File Converter.

4 Use the new stylesheet to convert the current XML source file.

- **a** In the **Stylesheet Editor: Main** Properties pane on the right, click **Send to File Converter**

  The File Converter appears, with the `customheader.rgs` stylesheet selected.
- **b** Click **Convert file**.

5 Change page orientation and paper type.

- **a** On the File Converter Properties pane, click **Edit**.
- **b** In the Options pane on the left, double-click the **Page Orientation** data item.
- **c** In the Properties pane on the right, use the selection list to change the value of the data item to `Landscape`.
- **d** In the Options pane in the middle, double-click **Paper Type** in the **Pagination and General Styles** folder.
- **e** In the Properties pane on the right, select **A4** from the selection list.
- **f** Save the stylesheet.

6 Generate the report `wsvar-report.xml` in PDF format using `customheader.rgs`..

The PDF report appears with horizontally oriented pages of slightly different dimensions.

## Edit Font Size as a Derived Value in XML

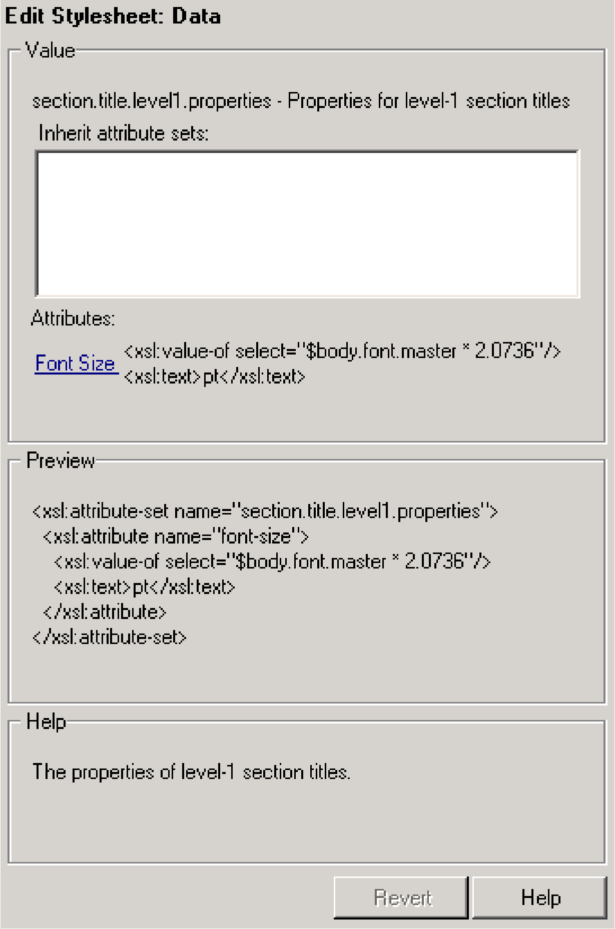
This example shows how to change the font size in a report to a value derived from other values. You do this by editing the PDF report's XML source directly.

1 Open the default print stylesheet for PDF documents.

2 In the Options pane in the middle, select and expand the **Property Sets** folder.

3 In the Options pane, double-click the **Section Title Level1 Properties** data item.

   The Properties pane on the right appears as follows.



**Edit Stylesheet: Data**

Value

section.title.level1.properties - Properties for level-1 section titles

Inherit attribute sets:

Attributes:

Font Size
```
<xsl:value-of select="$body.font.master * 2.0736"/>
<xsl:text>pt</xsl:text>
```

Preview

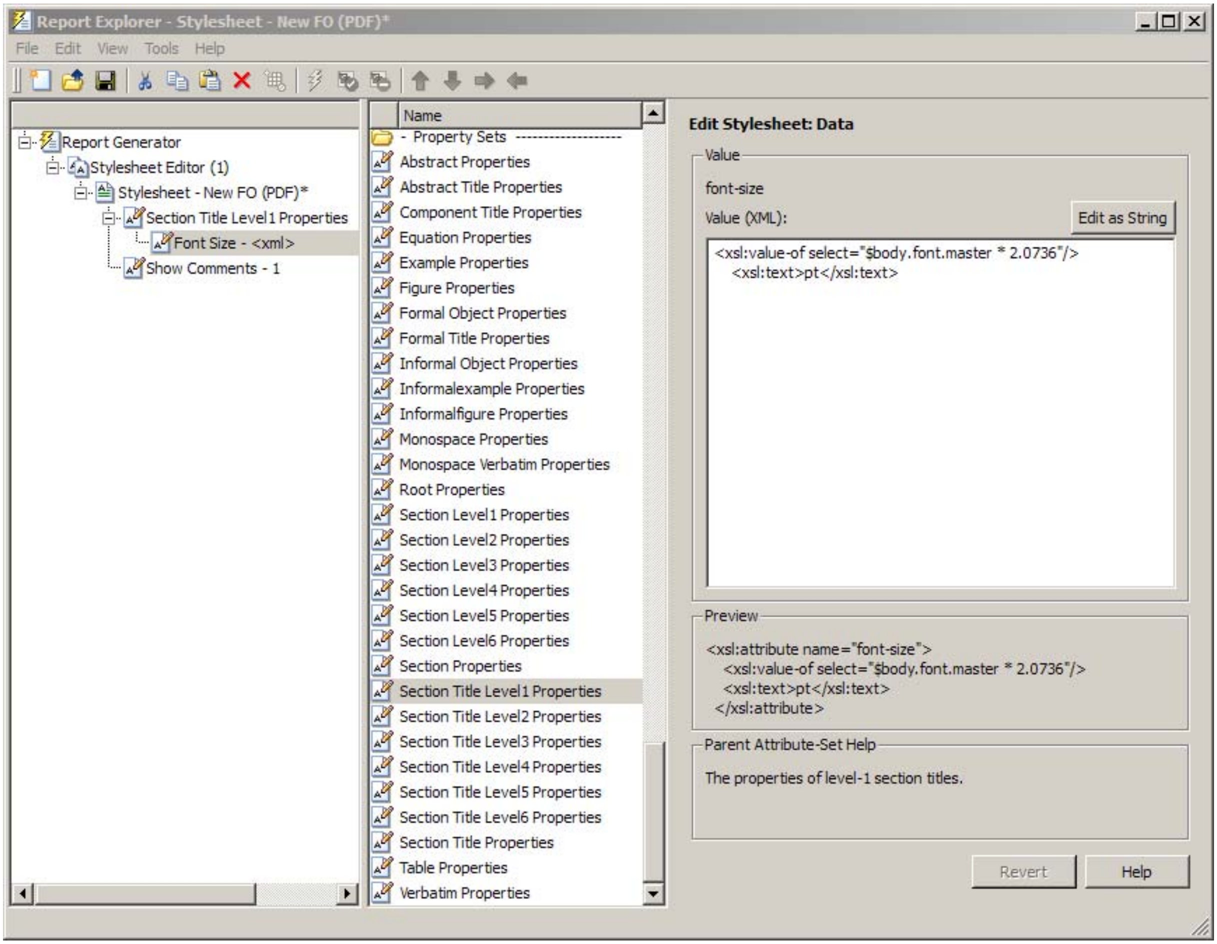
```
<xsl:attribute-set name="section.title.level1.properties">
  <xsl:attribute name="font-size">
    <xsl:value-of select="$body.font.master * 2.0736"/>
    <xsl:text>pt</xsl:text>
  </xsl:attribute>
</xsl:attribute-set>
```

Help

The properties of level-1 section titles.

4 In the **Attributes** area of the Properties pane on the right, click **Font Size - <xml>**.

The Report Explorer looks as follows.


Report Explorer - Stylesheet - New FO (PDF)*
File
Edit
View
Tools
Help
Report Generator
Stylesheet Editor (1)
Stylesheet - New FO (PDF)*
Section Title Level1 Properties
Font Size - <xml>
Show Comments - 1
Name
- Property Sets ------------------
Abstract Properties
Abstract Title Properties
Component Title Properties
Equation Properties
Example Properties
Figure Properties
Formal Object Properties
Formal Title Properties
Informal Object Properties
Informalexample Properties
Informalfigure Properties
Monospace Properties
Monospace Verbatim Properties
Root Properties
Section Level1 Properties
Section Level2 Properties
Section Level3 Properties
Section Level4 Properties
Section Level5 Properties
Section Level6 Properties
Section Properties
Section Title Level1 Properties
Section Title Level2 Properties
Section Title Level3 Properties
Section Title Level4 Properties
Section Title Level5 Properties
Section Title Level6 Properties
Section Title Properties
Table Properties
Verbatim Properties
Edit Stylesheet: Data
Value
Edit as String
Preview
Parent Attribute-Set Help
Revert
Help

font-size

Value (XML):

```
<xsl:value-of select="$body.font.master * 2.0736"/>
  <xsl:text>pt</xsl:text>
```

Preview

```
<xsl:attribute name="font-size">
  <xsl:value-of select="$body.font.master * 2.0736"/>
  <xsl:text>pt</xsl:text>
</xsl:attribute>
```

Parent Attribute-Set Help

The properties of level-1 section titles.

The `font size` value is a product of `$body.font.master` and `2.0736`. To change the font size to a larger size, change the multiplication factor to `3.0736`.

**Tip** You specify the value for the `$body.font.master` data item in the **Body Font Master** property. This property is in the **Pagination and General Styles** category in the Options pane in the middle. The default value of this data item is 10. Changing this value causes the derived values to change accordingly.

# PDF Fonts for Non-English Platforms

**In this section...**

## PDF Font Support for Languages

The MATLAB Report Generator supports a wide range of English language fonts for PDF reports.

The MATLAB Report Generator also provides basic PDF font support for some non-English languages, including:

- Japanese
- Korean
- Russian (Cyrillic)

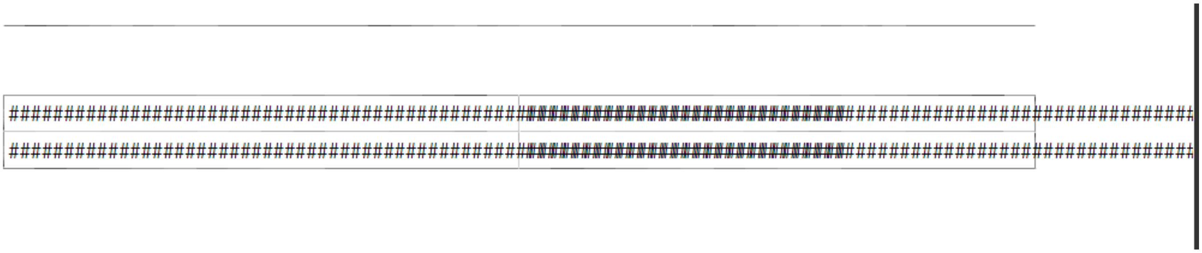
You can use the language font map to:

- Add or modify specifications for PDF font usage for supported non-English languages.
- Create PDF font support for a non-supported language.
- Change the default English fonts, if you do not specify a stylesheet.

The language font map specifications indicate what font to use on a specific platform (for example, Windows) for basic report elements such as body text.

## Identifying When to Specify a Font

If a required non-English font is missing for a report, the generated text includes pound sign characters (#). For example:



```
####################################################################################################
####################################################################################################
```

## Stylesheets Override PDF Font Mapping

PDF stylesheets for the MATLAB Report Generator specify fonts for body text, copyright, quotes, symbols, dingbats, monospace, sans serif, and titles.

The PDF stylesheet settings override the PDF font mapping entries.

If you do not specify a PDF stylesheet, then you can use PDF language font mapping entries to change the default fonts for English reports.

## Non-English PDF Font Mapping Tasks

To add or modify non-English PDF font mapping specifications:

- "Locate Non-English Fonts" on page 7-48
- "Add or Modify Language Font Mappings" on page 7-49
- "Specify the Location of Font Files" on page 7-50

## lang_font_map.xml File

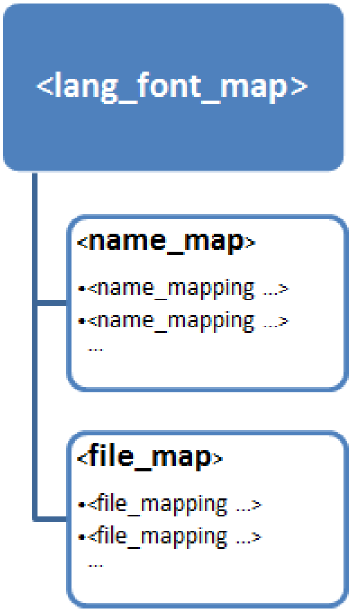
Use an XML editor with the `lang_font_map.xml` file to enter all the PDF font mappings for your reports.

Installing the MATLAB Report Generator software loads the `lang_font_map.xml` file in the following location:

```
<matlabroot>/toolbox/shared/rptgen/resources/fontmap
```

The lang_font_map.xml file includes two sections:

- name_map — Contains name_mapping elements that specify the name of the font, the language, and the font usage in the report (for example, body text).
- file_map— Contains entries for the location of the font files for the fonts specified in the name_map.



For example, the following lang_font_map.xml file includes name_map and file_map entries that provide basic PDF font support for Japanese (ja), Korean (ko), and Russian (ru).

### lang_font_map.xml example

```
<?xml version="1.0" encoding="UTF-8" ?>
<lang_font_map>
  <name_map>
    <name_mapping lang="ja" platform="win" usage="body">MS Gothic</name_mapping>
    <name_mapping lang="ja" platform="win" usage="monospace">MS Gothic</name_mapping>
    <name_mapping lang="ja" platform="win" usage="sans">MS Gothic</name_mapping>
    <name_mapping lang="ja" platform="win" usage="title">MS Gothic</name_mapping>

    <name_mapping lang="ko" platform="win" usage="body">Gulim</name_mapping>
    <name_mapping lang="ko" platform="win" usage="monospace">Gulim</name_mapping>
    <name_mapping lang="ko" platform="win" usage="sans">Gulim</name_mapping>
    <name_mapping lang="ko" platform="win" usage="title">Gulim</name_mapping>

    <name_mapping lang="ru" platform="win" usage="body">Arial Unicode MS</name_mapping>
    <name_mapping lang="ru" platform="win" usage="monospace">Arial Unicode MS</name_mapping>
    <name_mapping lang="ru" platform="win" usage="sans">Arial Unicode MS</name_mapping>
    <name_mapping lang="ru" platform="win" usage="title">Arial Unicode MS</name_mapping>

    <name_mapping lang="en" platform="glnx" usage="body">FreeSerif, Regular</name_mapping>
    <name_mapping lang="en" platform="glnx" usage="monospace">FreeMono, Regular</name_mapping>
    <name_mapping lang="en" platform="glnx" usage="sans">FreeSans, Regular</name_mapping>
    <name_mapping lang="en" platform="glnx" usage="title">FreeSerif, Bold</name_mapping>

    <name_mapping lang="ru" platform="mac" usage="body">Arial Unicode MS</name_mapping>
    <name_mapping lang="ru" platform="mac" usage="monospace">Arial Unicode MS</name_mapping>
    <name_mapping lang="ru" platform="mac" usage="sans">Arial Unicode MS</name_mapping>
    <name_mapping lang="ru" platform="mac" usage="title">Arial Unicode MS</name_mapping>
  </name_map>

  <file_map>
    <file_mapping lang="ja" platform="win" name="MS Gothic">msgothic.ttc</file_mapping>
    <file_mapping lang="ja" platform="win" name="MS PGothic">msgothic.ttc</file_mapping>

    <file_mapping lang="ko" platform="win" name="Gulim">gulim.ttc</file_mapping>

    <file_mapping lang="en" platform="glnx" name="FreeSerif, Regular">FreeSerif.ttf</file_mapping>
    <file_mapping lang="en" platform="glnx" name="FreeMono, Regular">FreeMono.ttf</file_mapping>
    <file_mapping lang="en" platform="glnx" name="FreeSans, Regular">FreeSans.ttf</file_mapping>
    <file_mapping lang="en" platform="glnx" name="FreeSerif, Bold">FreeSerifBold.ttf</file_mapping>

    <file_mapping lang="ru" platform="mac" name="Arial Unicode MS">Arial Unicode.ttf</file_mapping>
  </file_map>
</lang_font_map>
```

## Locate Non-English Fonts

The system from which you generate a report using the language font map must have access to the appropriate non-English fonts.

Use one of these font formats for non-English font support:

- Type 1 (PostScript®)
- TrueType
- OpenType®

Fonts in other formats, such as bitmap fonts for the X Window System (X11), produce poor MATLAB Report Generator report output.

Some TrueType fonts are grouped into packages called TrueType Collections. To specify a collection in the language font map file, specify the individual font within the `TrueType Collection`.

In addition to the font name, the weight (e.g., bold) and slant (e.g., italic, oblique) may distinguish one font from another in the same family.

The approach you use to identify font names depends on your computer platform.

### Font names on Windows

To identify a TrueType font name on Windows systems:

1. Navigate to the font folder (usually `C:\Windows\Fonts`).
2. If the font is a simple TrueType (not a collection), in the window, right-click the font and choose **Properties** to see the name of the file containing that font.
3. If the font is a TrueType Collection, right-click to open the collection, optionally in a new window. Each constituent font appears, with its name. Use the name of the constituent font, not the name of the whole collection.
   a. Right-click any of constituent font and select **Properties**. The properties box displays the name of the file containing that font.

### Font names on Mac OS X

Mac OS X provides an application called `Font Book` (in the `/Applications` folder) that provides information about available fonts on the system. The application shows all the fonts on your system. Hover over a specific font to see a datatip with the font name and the path to the font.

### Font names on Linux

Linux distributions use a variety of conventions for the location of fonts, or how those font folders can be found. By default, MATLAB Report Generator searches these folders, in this order:

1. `/.fonts/`
2. `/usr/local/share/fonts/`
3. `/usr/X11R6/lib/fonts/`
4. `/usr/share/fonts/`

You can specify alternative folders in the `fonts.conf` file (in the `/etc/fonts/` folder).

## Add or Modify Language Font Mappings

In the `name_map` section of the `lang_font_map.xml` file, add a separate `name_mapping` entry for each combination of language, font, and usage that you want in PDF reports.

Each `name_mapping` element has three attributes:

- `lang` specifies the two letter ISO 639-1 code corresponding to the language of the report.
- `platform` specifies the operating system platform:
  - `win` — Windows
  - `mac` — Mac OS X
  - `glnx`— Linux

- usage specifies the kind of report element or font:
  - body
  - title
  - monospaced
  - sans (sanserif)

The text of the name_mapping element is a font name, as specified in an XSL-FO stylesheet.

Here is an example name_mapping entry:

```
<name_mapping_lang="ja" platform="win" usage="body">MS Gothic</name_mapping>
```

## Specify the Location of Font Files

In the file_map section, add a file_mapping entry that identifies the location of the font file for each font that you include in the name_map section.

Each of the platforms (Windows, Mac, and Linux) has a different default search path for fonts. If the lang_font_map.xml file does not contain a full file path for a font, the MATLAB Report Generator uses a platform-specific approach to search for the font.

### Windows Font File Locations

On Windows platforms, the MATLAB Report Generator searches for fonts in <windir>/Fonts, where *winder* is an operating system environment variable. The typical location is C:\Windows or C:\Winnt.

### Mac Font File Locations

On Mac OS X platforms, fonts are generally in one of these folders:

- ~/Library/Fonts
- /Library/Fonts
- Network/Library/Fonts
- System/Library/Fonts

- `System/Folder/Fonts`

### Linux Font File Locations

On Linux platforms, the convention for locating fonts can differ, depending on the Linux distribution. The MATLAB Report Generator follows the Debian® convention of finding the list of font folders in the `/etc/fonts/fonts.conf` file.

If the MATLAB Report Generator does not find the `fonts.conf` file in `/etc/fonts/` folder, it searches the following folders, in the following order:

1. `/.fonts`
2. `/usr/local/share/fonts`
3. `/usr/X11R6/lib/fonts`
4. `/usr/share/fonts`

Because of the variety of conventions used in different Linux distributions, consider using full file paths in `file_mapping` elements.

# Comparing XML Files

# Compare XML Files

You can use MATLAB Report Generator software to compare a pair of XML text files.

The XML comparison tool processes the results into a report that you can use to explore the file differences.

You can access the XML comparison tool from:

- The MATLAB Current Folder browser context menu
- The MATLAB Comparison Tool
- The MATLAB command line.

The XML comparison tool compares the files using the “Chawathe” algorithm, as described in this paper:

> *Change Detection in Hierarchically Structured Information*, Sudarshan Chawathe, Anand Rajaraman, and Jennifer Widom; SIGMOD Conference, Montreal, Canada, June 1996, pp. 493-504.

This conference paper is based upon work published in 1995: see `http://dbpubs.stanford.edu:8090/pub/1995-45`.

XML comparison reports display in the Comparison Tool. For more information about the Comparison Tool, see “Comparing Files and Folders” in the MATLAB documentation.

The XML comparison report shows a hierarchical view of the portions of the two XML files that differ. The report does not show sections of the files that are identical.

If the files are identical you see a message reporting there are no differences.

**Note** It might not be possible for the analysis to detect matches between previously corresponding sections of files that have diverged too much.

Change detection in the Chawathe analysis is based on a scoring algorithm. Items match if their Chawathe score is above a threshold. The MATLAB Report Generator implementation of Chawathe's algorithm uses a comparison pattern that defines the thresholds. For more information, see "How the Matching Algorithm Works" on page 8-12.

For information about creating and using XML comparison reports, see:

1. "How to Compare XML Files" on page 8-4
2. "Explore the XML Comparison Report" on page 8-7

# How to Compare XML Files

**In this section...**

## Select Files to Compare

### From the Current Folder Browser

To compare two files from the Current Folder browser:

- For two files in the same folder, select the files, right-click and select **Compare Selected Files/Folders**.
- To compare files in different folders:
  1. Select a file, right-click and select **Compare Against**
  2. Select the second file to compare in the Select Files or Folders for Comparison dialog box.
  3. Leave the default **Comparison type**, `XML text comparison`.
  4. Click **Compare**.

If the selected files are XML files, the XML comparison tool performs a Chawathe analysis on the files and displays a report in the Comparison Tool.

The file you right-click to launch the XML comparison tool displays on the right side of the report.

For more information about comparisons of other file types with the Comparison Tool, such as text, MAT or binary, see "Comparing Files and Folders" in the MATLAB documentation.

### From the Comparison Tool

To compare files using the Comparison Tool, from the MATLAB Toolstrip, in the **File** section, select the **Compare** button. In the dialog box select files to compare.

If the files you select to compare are XML files and you select an `XML comparison`, the XML comparison tool performs a Chawathe analysis of the XML files, and generates an report.

### From the Command Line

To compare XML files from the command line, enter

```
visdiff(filename1, filename2)
```

where `filename1` and `filename1` are XML files. This XML comparison functionality is an extension to the MATLAB `visdiff` function.

## Change Comparison Type

If you specify two XML files to compare using either the Current Folder Browser or the `visdiff` function, then the Comparison Tool automatically performs the default comparison type, `XML text comparison`.

To change comparison type, either create a new comparison from the Comparison Tool, or use the **Compare Against** option from the Current Folder browser. You can change comparison type in the Select Files or Folders for Comparison dialog box. If you want the MATLAB text differences report for XML files, change the comparison type to `Text comparison` in the dialog before clicking **Compare**.

## XML Comparison Examples

For an example with instructions, see `mlxml_testplan`.

## See Also

For an overview, see “Compare XML Files” on page 8-2.

For explanations on how to use and understand the report and the XML comparison functionality, see “Explore the XML Comparison Report” on page 8-7

# Explore the XML Comparison Report

**In this section...**

## Navigate the XML Comparison Report

The XML comparison report shows changes only. The report is a hierarchical view of the differences between two XML text files, and is not a hierarchical view of the original XML data.

To *step through differences*, use the **Comparison** tab on the toolstrip. To move to the next or previous group of differences, on the **Comparison** tab, in the **Navigate** section, click the arrow buttons to go to the previous or next difference.

You can also click to select items in the hierarchical trees.

- Selected items appear highlighted in a box.
- If the selected item is part of a matched pair it is highlighted in a box in both left and right trees.

Report item highlighting indicates the nature of each difference as follows:

| Type of report item | Highlighting | Notes |
|---|---|---|
| Modified | Pink | Modified items are matched pairs that differ between the two files. When you select a modified item it is highlighted in a box in both trees.<br>Changed parameters for the selected pair are displayed in a separate **Parameters** panel for review. If strings are too long to display in the **Parameters** table, right-click |

| Type of report item | Highlighting | Notes |
|---|---|---|
| | | and select **Compare as Text** to open a new comparison of the parameters. |
| Unmatched | Green | When you select an unmatched item it is highlighted in a box in one tree only. |
| Container | None | Rows with no highlighting indicate a container item that contains other modified or unmatched items. |

Use the toolbar buttons or the **Comparison** menu for the following functions:

- **Refresh** — Run Chawathe analysis again to refresh the comparison report.
- **Swap Sides** — Swap sides and rerun comparison. Runs the Chawathe analysis again.
- **Save As > Save as HTML** — Opens the Save dialog box, where you can choose to save a printable version of the XML comparison report. The report is a noninteractive HTML document of the differences detected by the Chawathe algorithm for printing or archiving a record of the comparison.
- **Save As > Save to Workspace** — Export XML comparison results to workspace.
- In the **Navigate** section, click the arrow buttons (or press Up or Down keys) to go to the previous difference or go to the next difference.
- **Compare Selected Parameter** — Open a new report for the currently selected pair of parameters. Use this when the report cannot display all the details in the Parameters pane, e.g., long strings or a script.

Use the **View** tab controls on the toolstrip for the following functions:

- **Expand All** — Expands every item in the tree.

**Tip** Right-click to expand or collapse the hierarchy within the selected tree node.

- **Collapse All** — Collapses all items in the tree to the most compact view possible.

See also "XML Comparison Examples" on page 8-5.

### Unexpected Results

If you see unexpected results within an XML comparison report, try reading the documentation section on "How the Matching Algorithm Works" on page 8-12.

---

**Note** It may not be possible for the analysis to detect matches between previously corresponding sections of files that have diverged too much.

---

## Save Comparison Log Files in a Zip File

Temporary XML comparison files accumulate in `tempdir/MatlabComparisons/XMLComparisons/TempDirs/`. These temporary files are deleted when you close the related comparison report.

You can zip the temporary files (such as log files) created during XML comparisons, for sharing or archiving. While the comparison report is open, enter:

```
xmlcomp.zipTempFiles('c:\work\myexportfolder')
```

The destination folder must exist. The output reports the zip file name:

```
Created the zipfile "c:\work\myexportfolder\20080915T065514w.zip"
```

To view the log file for the last comparison in the MATLAB Editor, enter:

```
xmlcomp.showLogFile
```

## Export Results to the Workspace

To export the XML comparison results to the MATLAB base workspace,

1. On the **Comparison** tab, in the **Comparison** section, select **Save As > Save to Workspace**.

The Input Variable Name dialog box appears.

2 Specify a name for the export object in the dialog and click **OK**. This action exports the results of the XML comparison to an `xmlcomp.Edits` object in the workspace.

The `xmlcomp.Edits` object contains information about the XML comparison including file names, filters applied, and hierarchical nodes that differ between the two XML files.

To create an `xmlcomp.Edits` object at the command line without opening the Comparison Tool, enter:

```
Edits = xmlcomp.compare(a.xml,b.xml)
```

| **Property of `xmlcomp.Edits`** | **Description** |
|---|---|
| `Filters` | Array of filter structure arrays. Each structure has two fields, Name and Value. |
| `LeftFileName` | File name of left file exported to XML. |
| `LeftRoot` | `xmlcomp.Node` object that references the root of the left tree. |
| `RightFileName` | File name of right file exported to XML. |
| `RightRoot` | `xmlcomp.Node` object that references the root of the right tree. |
| `TimeSaved` | Time when results exported to the workspace. |
| `Version` | MathWorks® release-specific version number of `xmlcomp.Edits` object. |

| **Property of `xmlcomp.Node`** | **Description** |
|---|---|
| `Children` | Array of `xmlcomp.Node` references to child nodes, if any. |
| `Edited` | Boolean — If `Edited = true` then the node is either inserted (green) or part of a modified matched pair (pink). |
| `Name` | Name of node. |
| `Parameters` | Array of parameter structure arrays. Each structure has two fields, Name and Value. |
| `Parent` | `xmlcomp.Node` reference to parent node, if any. |
| `Partner` | If matched, `Partner` is an `xmlcomp.Node` reference to the matched partner node in the other tree. Otherwise empty `[]`. |

# How the Matching Algorithm Works

**In this section...**

## Why Do I See Unexpected Results?

The core of the XML file comparison engine is Chawathe's matching algorithm. This matching algorithm is a heuristic method based on a scoring system. This means that comparison results could be unexpected when many elements in each document are very similar.

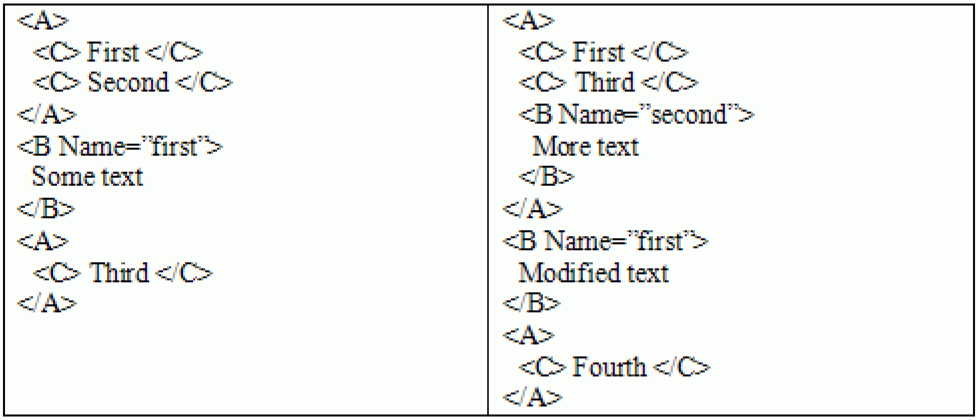
See the following sections for some examples.

## How the Chawathe Algorithm Works

XML text documents are hierarchical data structures. Users can insert, delete, or reorder elements, modify their contents, or move elements across different parts of the hierarchy. The Chawathe algorithm can detect these different types of changes within the hierarchy of the document. As with conventional text differencing utilities, the Chawathe algorithm detects local text that is added, deleted, or changed, and additionally can prepare an edit script that can be used to create a report of the hierarchical location of detected differences.

The Chawathe algorithm attempts to match elements that are of the same category. The Chawathe paper refers to these categories as *labels*. In the following XML example documents (with labels `A`, `B`, and `C`):

- The three `C` elements on the left are compared with the three `C` elements on the right
- The single `B` element on the left is compared with the two `B` elements on the right



```
<A>
  <C> First </C>
  <C> Second </C>
</A>
<B Name="first">
  Some text
</B>
<A>
  <C> Third </C>
</A>
```

```
<A>
  <C> First </C>
  <C> Third </C>
  <B Name="second">
    More text
  </B>
</A>
<B Name="first">
  Modified text
</B>
<A>
  <C> Fourth </C>
</A>
```

The Chawathe algorithm matches a particular label by extracting a flat sequence of elements from the hierarchical document and attempting to match the elements in the sequences. In the example above, elements of the sequence

```
(<C> First </C>, <C> Second </C>, <C> Third </C>)
```

are matched against elements of the sequence

```
(<C> First </C>, <C> Third </C>, <C> Fourth </C>)
```

Sequences are matched using a Longest Common Subsequence (LCS) algorithm. For example, if `C` elements are matched on their text content, the LCS of the above sequences is given by:

```
(<C> First </C>, <C> Third </C>)
```

You can define a *score* for matching elements of a particular label in different ways. For instance, in the above example, `C` elements can be matched on text content, `B` can be matched on text content and on `Name`, and `A` on the number of `B` and `C` elements they have in common. To determine whether elements match or not, the Chawathe algorithm compares the score to a threshold.

The implementation can specify scoring methods, thresholds, the definition of labels, and the order in which labels are processed. These can be defined separately for each problem domain or type of XML file. The XML comparison tool provides suitable definitions for a set of common XML file types, and uses a default definition for any type of XML document it does not recognize.

## Why Use a Heuristic Algorithm?

Chawathe’s algorithm is a heuristic. That is, it cannot guarantee to return the optimal matching between two sequences. It is the use of a threshold mechanism in combination with an LCS algorithm that makes the algorithm a heuristic. A heuristic algorithm is preferable to an optimal matching because the heuristic is much faster.

An algorithm can only guarantee a mathematically optimal matching by exhaustively computing the score between all pairs of elements in the two sequences and choosing those pairs that maximize an overall matching score between the two sequences. This exhaustive approach is computationally very expensive because its running time increases exponentially with the length of the sequences to be matched.

Also a user’s expectations can depend on context information that is not available to the matching algorithm (e.g., prior knowledge of the precise sequence of changes applied). This means even a mathematically optimal algorithm might match elements unexpectedly from a user’s perspective.

In contrast with the mathematically optimal approach, Chawathe’s algorithm guarantees linear running time for sequences that are the same or very similar. The worst-case scenario is quadratic running time for sequences that are entirely different.

The XML comparison tool performs best when the files to be compared are mostly similar. It becomes slower for files that contain more differences.

## Examples of Unexpected Results

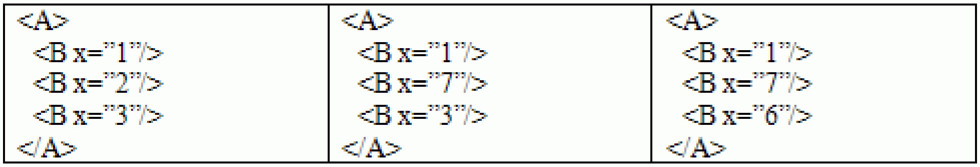
- “Elements Matched in Previous Comparisons Fail to Match” on page 8-14
- “Elements Matched Across Different Parts of the Hierarchy” on page 8-15
- “Two Sequences of Elements Are Cross-Matched” on page 8-17

### Elements Matched in Previous Comparisons Fail to Match

Elements could fail to match even if they were matched in comparisons of previous versions of the documents. A seemingly small change in one of the properties used for matching can cause this to happen if it tips the score under the threshold.

Consider the following example where

- B elements are scored on the value of x
- A elements are scored on the ratio of matching B elements
- For both A and B the score is compared with a threshold of 0.5.



| | | |
|---|---|---|
| <A><br>  <B x=”1”/><br>  <B x=”2”/><br>  <B x=”3”/><br></A> | <A><br>  <B x=”1”/><br>  <B x=”7”/><br>  <B x=”3”/><br></A> | <A><br>  <B x=”1”/><br>  <B x=”7”/><br>  <B x=”6”/><br></A> |

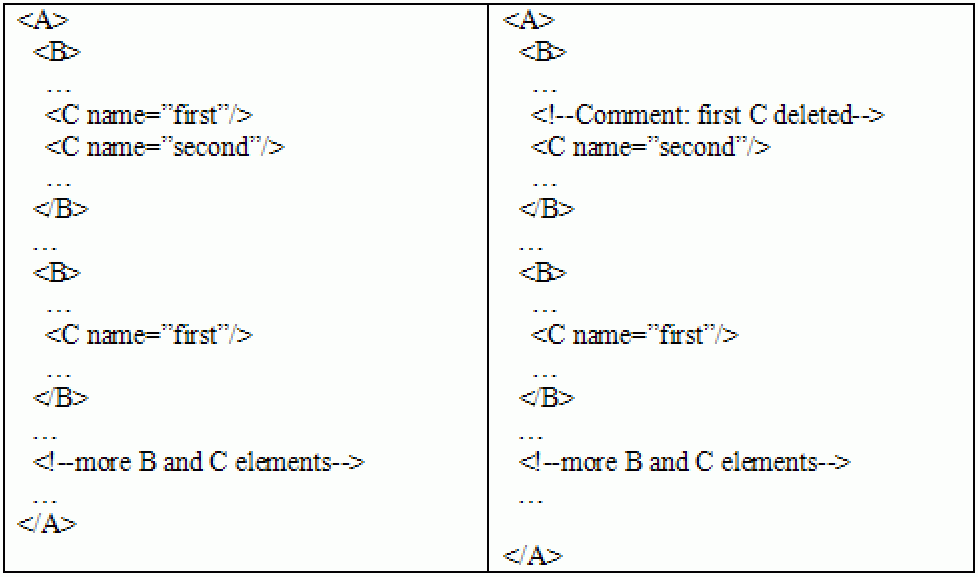
The left A and the middle A have two out of three B elements in common, resulting in a matching score of 2/3=0.66. The XML comparison tool marks the A elements as matched and the report shows that their contents have been *modified*.

When a user makes a further change to the middle document (resulting in the right document), and this new document is compared again to the left document, the matching score for A drops to 1/3=0.33. The algorithm considers the A elements unmatched this time. In this case, the difference between the two documents is marked as a *deletion* of A from the left document and an *insertion* of a new A into the right document.

This problem is likely to occur when there is little information available inside a single element to score a match. A seemingly small change in one of the properties used for matching could tip the score under the threshold, and therefore result in a large change in the outcome of the comparison.

### Elements Matched Across Different Parts of the Hierarchy

Sometimes unexpected matches of similar items occur across different parts of the hierarchy. In the following example, C elements are matched on name:



```
<A>
  <B>
    ...
    <C name=”first”/>
    <C name=”second”/>
    ...
  </B>
  ...
  <B>
    ...
    <C name=”first”/>
    ...
  </B>
  ...
  <!--more B and C elements-->
  ...
</A>
```

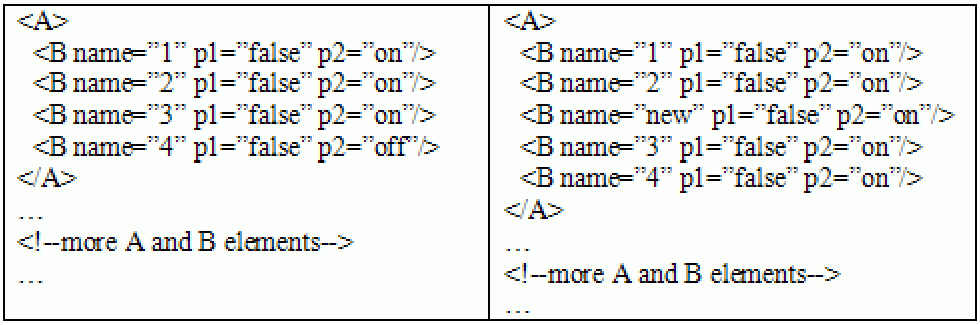
```
<A>
  <B>
    ...
    <!--Comment: first C deleted-->
    <C name=”second”/>
    ...
  </B>
  ...
  <B>
    ...
    <C name=”first”/>
    ...
  </B>
  ...
  <!--more B and C elements-->
  ...

</A>
```

In this case, the user might expect to see the very *first* `C` element on the left marked as deleted, with the second and third `C` elements matched to the corresponding `C` element on the right. However, this might not happen, if the first `C` on the left is matched to the second `C` on the right, even though these two `C` elements exist in very different parts of the document hierarchy. This mismatch would result in the third `C` element on the left being marked as deleted, which the user might find unexpected.

This case is likely to occur when there are several potential matching candidates for a particular element. In other words, when elements of a particular label tend to be very similar. Whether such a spurious cross-matching occurs or not depends on all of the other `C` elements within the two documents. The LCS algorithm used for matching the two sequences favors *local* matches over *distant* ones. In other words, sub-sequences of elements that are close together in the first sequence tend to be matched to sub-sequences of elements that are close together in the second sequence. However, this locality is not always guaranteed, and the outcome depends on how other elements in the sequence are matched.

### Two Sequences of Elements Are Cross-Matched

It is difficult to distinguish many similar potential matches and this could produce unexpected results. In the following example, B elements are scored on name, p1, and p2, and the score is compared to a threshold of 0.5.



```
<A>
  <B name=”1” p1=”false” p2=”on”/>
  <B name=”2” p1=”false” p2=”on”/>
  <B name=”3” p1=”false” p2=”on”/>
  <B name=”4” p1=”false” p2=”off”/>
</A>
...
<!--more A and B elements-->
...
```

```
<A>
  <B name=”1” p1=”false” p2=”on”/>
  <B name=”2” p1=”false” p2=”on”/>
  <B name=”new” p1=”false” p2=”on”/>
  <B name=”3” p1=”false” p2=”on”/>
  <B name=”4” p1=”false” p2=”on”/>
</A>
...
<!--more A and B elements-->
...
```

The right document contains one B element more than the left document, and therefore one of the B elements on the right must remain unmatched and the tool will mark one as inserted. However, since most B elements on the left potentially match most B elements on the right, it is impossible to predict exactly how the sequences will be matched. For instance, the comparison could generate the following result:

```
B name= 1      >  B name= 2
B name= 2      >  B name= new
B name= 3      >  B name= 3
B name= 4      >  B name= 4
```

In this case, “B name= “1” on the right remains unmatched. As in the previous example, this depends on how all of the other B elements in the two documents are matched. This situation is likely to occur when elements have several potential matching candidates.

# Components — Alphabetical List

Array-Based Table
Axes Loop
Axes Snapshot
Chapter/Subsection
Comment
Empty Component
Evaluate MATLAB Expression
Figure Loop
Figure Snapshot
For Loop
Graphics Object Loop
Handle Graphics Linking Anchor
Handle Graphics Name
Handle Graphics Parameter
Handle Graphics Property Table
Handle Graphics Summary Table
Image
Import File
Insert Variable
Link
List
Logical Else
Logical Elseif
Logical If
Logical Then
MATLAB Property Table

MATLAB/Toolbox Version Number
Nest Setup File
Paragraph
Stop Report Generation
Table
Table Body
Table Column Specification
Table Entry
Table Footer
Table Header
Table Row
Text
Time/Date Stamp
Title Page
Variable Table
While Loop

## Purpose

Convert rectangular array into table and insert it into report

## Description

This component converts a rectangular cell array into a table and inserts the table into the report.

## Table Content

- **Workspace variable name**: Specifies the workspace variable name with which to construct the table.
- **Collapse large cells to a single description**: Consolidates large cells into one description.

## Formatting Options

- **Table title**: Specifies the title of your table.
- **Cell alignment**:
  - `left`
  - `center`
  - `right`
  - `double justified`
- **Column widths**: Inputs a vector with *m* elements, where *m* equals the number of columns in the table. Column sizing is relative and normalized to page width. For example, say that you have a 2-by-3 cell array and input the following into the **Column widths** field:

  ```
  [1 2 3]
  ```

  The report output format for the cell array is such that the second column is twice the width of the first column, and the third column is three times the width of the first column. If the vector is greater than the number of columns in the table, the vector is truncated so that the number of elements equals the number of columns. If *m* is less than the number of columns in the table, the vector is padded with `1`s so that the number of elements equals the number of columns.

  If you use this field, it is recommended that you specify a width for each column. Any width not specified defaults to 1. MATLAB displays a warning when defaulting any unspecified column width to 1.

- **Table grid lines**: Displays grid lines, which create borders between fields, in the table.
- **Table spans page width (HTML only)**: Sets the table width to the width of the page on which it appears.

### Header/Footer Options

Designating a row as a header or footer row causes the contents of the row to appear in boldface.

- **Number of header rows**: Specifies the number of header rows.
- **Footer list**:
  - `No footer`: Specifies no footers for the report.
  - `Last N rows are footer`: Enables you to select a footer that is different from your header.

#### Example

Consider the following cell array in the MATLAB workspace:

```
{'foo','bar';[3],[5]}
```

Its cell table in the report appears as follows.

| | |
|---|---|
| foo | bar |
| 3 | 5 |

Note that the table has no headers or footers and no title.

### Insert Anything into Report?

Yes. Table.

### Class

`rptgen.cfr_table`

## See Also

Table, Table Body, Table Column Specification, Table Entry, Table Footer, Table Header, Table Row, Chapter/Subsection, Empty Component, Image, Link, List, Paragraph, Text, Title Page

# Axes Loop

## Purpose

Run child components for all axes objects in MATLAB workspace

## Description

The Axes Loop component runs its child components for all axes objects in the MATLAB workspace. For information about working with looping components, see “Logical and Looping Components” on page 5-22.

## Object Selection

- **Loop type**:
  - `All axes`: Loops on all axes objects.
  - `Current axes`: Loops on the currently selected axes object.
- **Exclude objects which subclass axes**: Excludes objects, such as legends and color bars, from the loop.
- **Loop Menu**:
  - `Loop on axes with handle visibility "on"`: Loops only on visible axes objects.
  - `Loop on all axes`: Loops on all axes objects.
- **Search terms**: Specifies search terms for the loop. For example, to search for `Tag` and `My Data`, enter `"Tag", "My Data"`.

## Section Options

- **Create section for each object in loop**: Inserts a section in the generated report for each object found in the loop.
- **Display the object type in the section title**: Automatically inserts the object type into the section title in the generated report.
- **Create link anchor for each object in loop**: Creates a hyperlink to the object in the generated report.

## Insert Anything into Report?

Yes, inserts a section if you select the **Create section for each object in loop** option.

## Class

`rptgen_hg.chg_ax_loop`

### See Also

`Axes Snapshot`, `Figure Loop`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics Name`, `Handle Graphics Parameter`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

# Axes Snapshot

## Purpose

Insert image of selected MATLAB axes objects into the generated report

## Description

Inserts an image of selected MATLAB axes objects into the generated report.

## Format

- **Image file format**: Specifies the image file format. Select `Automatic HG Format` to automatically choose the format best suited for the specified report output format. Otherwise, choose an image format that your output viewer can read. `Automatic HG Format` is the default option. Options include:
  - `Automatic HG Format` (uses the Handle Graphics file format selected in the Preferences dialog box)
  - `Bitmap (16m-color)`
  - `Bitmap (256-color)`
  - `Black and white encapsulated PostScript`
  - `Black and white encapsulated PostScript (TIFF)`
  - `Black and white encapsulated PostScript2`
  - `Black and white encapsulated PostScript2 (TIFF)`
  - `Black and white PostScript`
  - `Black and white PostScript2`
  - `Color encapsulated PostScript`
  - `Color encapsulated PostScript (TIFF)`
  - `Color encapsulated PostScript2`
  - `Color encapsulated PostScript2 (TIFF)`
  - `Color PostScript`
  - `Color PostScript2`
  - `JPEG high quality image`
  - `JPEG medium quality image`

  - `JPEG low quality image`
  - `PNG 24-bit image`
  - `TIFF - compressed`
  - `TIFF - uncompressed`
  - `Windows metafile`
- **Capture figure from screen**: Captures the figure for the generated report directly from the screen. Options include:
  - `Client area only`: Captures part of the figure.
  - `Entire figure window`: Captures the entire figure window.

## Print Options

- **Paper orientation**:
  - `Landscape`
  - `Portrait`
  - `Rotated`
  - `Use figure orientation`: Uses the orientation for the figure, which you set with the `orient` command.
  - `Full page image (PDF only)`: In PDF reports, scales images to fit the full page, minimizes page margins, and maximizes the size of the image by using either a portrait or landscape orientation.

  For more information about paper orientation, see the `orient` command in the MATLAB documentation.
- **Image size**:
  - `Use figure PaperPositionMode setting`: Sets the image size in the report to the `PaperPositionMode` property of the figure. For more information about paper position mode, see `orient` in the MATLAB documentation.
  - `Automatic (same size as onscreen)`: Sets the image in the report to the same size as it appears on the screen.

  - `Custom`: Specifies a custom image size. Specify the image size in the **Size** field and **Units** list.

- **Size**: Specifies the size of the figure snapshot in the format `[w h]` (width, height). This field is active only if you choose `Custom` in the **Image size** selection list.
- **Units**: Specifies the units for the size of the figure snapshot. This field is active only if you choose `Set image size` in the **Custom** selection list.
- **Invert hardcopy**: Sets the `InvertHardcopy` property of Handle Graphics figures. This property inverts colors for printing; that is, it changes dark colors to light colors and vice versa. Options include:
  - `Automatic`: Automatically changes dark axes colors to light axes colors. If the axes color is a light color, it is not inverted.
  - `Invert`: Changes dark axes colors to light axes colors and vice versa.
  - `Don't invert`: Does not change the colors in the image displayed on the screen for printing.
  - `Use figure's InvertHardcopy setting`: Uses the `InvertHardcopy` property set in the Handle Graphics image.
  - `Make figure background transparent`: Makes the image background transparent.

## Display Options

- **Scaling**: Controls size of the image, as displayed in a browser. Making an image larger using this option does not affect the storage size of the image, but the quality of the displayed image may decrease as you increase or decrease the size of the displayed image.

  Generally, to achieve the best and most predictable display results, use the default setting of `Use image size`.

  - `Use image size`: Causes the image to appear the same size in the report as on screen (default).
  - `Fixed size`: Specifies the number and type of units.

  - `Zoom`: Specifies the percentage, maximum size, and units of measure.

- **Size**: Specifies the size of the snapshot in the format `w h` (width, height). This field is active only if you choose `Fixed size` in the **Scaling** list.
- **Max size**: Specifies the maximum size of the snapshot in the format `w h` (width, height). This field is active only if you choose `Zoom` from the **Scaling** list.
- **Units**: Specifies the units for the size of the snapshot. This field is active only if you choose `Zoom` or `Fixed size` in the **Image size** selection list.
- **Alignment**: Only reports in `PDF` or `RTF` format support this property. Options are:
  - `Auto`
  - `Right`
  - `Left`
  - `Center`
- **Title**: Specifies text to appear above the snapshot.
- **Caption**: Specifies text to appear under the snapshot.

## Insert Anything into Report?

Yes. Image.

## Class

`rptgen_hg.chg_ax_snap`

## See Also

`Axes Loop`, `Figure Loop`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics Name`, `Handle Graphics Parameter`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

# Chapter/Subsection

## Purpose

Group portions of report into sections with titles

## Description

This component groups portions of the report into sections. Each section has a title and content.

The following rules apply to this component:

- Child components appear inside the section created by this component.
- Selecting the **Get title from first child component** check box prevents this component from accepting paragraph-level children. In this case, this component's first child must be a `Text` component.
- This component can have Chapter/Subsection components as its children.
- Sections can be nested. There are seven levels of nesting possible. The seventh nested section in the report is untitled, although the child components of this section include information into the report.

### Chapter Numbering

By default, chapters are numbered and sections are not numbered. Specify chapter and section numbering using a stylesheet. For more information about chapter and section numbering options in Web and print stylesheets, see "Report Output Format and Stylesheet" on page 4-5.

## Section Title

- **Title**: Specifies a title to display in the generated report:
  - `Automatic`: Automatically generates a title.
  - `Custom`: Specifies a custom title.

- **Numbering**: Specifies a numbering style for the report:
  - `Automatic`: Numbers by context.
  - `Custom`: Allows you to create your own numbering style.
- **Section Type**: Shows you in which level a selected section resides.

**Insert Anything into Report?**

Yes. Chapter or section.

**Class**

`rptgen.cfr_section`

**See Also**

`Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Table`, `Text`, `Title Page`

# Comment

## Purpose

Insert comment into XML source file created by report generation process

## Description

This component inserts a comment into the XML source file created by the report-generation process. This comment is not visible in the generated report.

This component can have children. Child components insert their output into the XML source file, but this does not appear in the generated report.

To make comment text appear in the report:

1. Edit the XML source file (which has the same name as your report file, but has a `.xml` extension).

2. Find the `comment` area in the XML source file by locating the comment tags `<--` and `-->`.

3. Remove the comment tags.

4. Convert the XML source file using the `rptconvert` command.

## Properties

- **Comment text**: Specifies comments to include in the report.
- **Show comment in Generation Status window**: Displays comments in the **Generation Status** tab while the report generates.
- **Status message priority level**: Specifies the priority level of the status messages that appear during report generation. Priority options range from `1) Error messages only` to `6) All messages`. The default is `3) Important messages`. This option is only available if you select the **Show comment in Generation Status window** option.

## Insert Anything into Report?

No. This component inserts comments, which can appear in the report, into the report's XML source file.

## Class

`rptgen.crg_comment`

## See Also

"Convert XML Documents" on page 4-19, `Import File`, `Nest Setup File`, `Stop Report Generation`, `Time/Date Stamp`

# Empty Component

## Purpose

Group components to move, activate, or deactivate them, or create blank space in list

## Description

This component does not insert anything into the generated report. It can have any component as a child. You can use it to group components together so that you can easily move, activate, or deactivate them, or create a blank space in a list.

If the MATLAB Report Generator software does not recognize a given component when loading a report setup file, it replaces the unrecognized component with this component.

## Insert Anything into Report?

No.

## Class

`rptgen.crg_empty`

## See Also

`Chapter/Subsection`, `Image`, `Link`, `List`, `Paragraph`, `Table`, `Text`, `Title Page`

# Evaluate MATLAB Expression

## Purpose

Evaluate specified MATLAB expression

## Description

This component evaluates a specified MATLAB expression. You can include code and/or command-line output in the report.

## Properties

- **Insert MATLAB expression in report**: Causes the MATLAB expression that this component evaluates to appear in the report.
- **Display command window output in report**: Includes the command window output that results from the evaluation of the specified MATLAB expression.
- **Expression to evaluate in the base workspace**: Specifies the expression to evaluate in the MATLAB workspace. .

  If you are using Simulink Report Generator, then you can use functions such as `Rptgen.getReportedBlock` to filter the modeling elements on which to report and to perform special reporting on specific elements. For more information, in the Simulink Report Generator documentation, see "Loop Context Functions".
- **Evaluate this expression if there is an error**: Evaluates another MATLAB expression if the specified expression produces an error. You must enter in this field the expression to evaluate in case of an error.

  If you do not change the default error handling code, then when you generate the report, and there is an error in the MATLAB code that you added:

  - If you clear **Evaluate this expression if there is an error** check box, then the complete report is generated, without displaying an error message at the MATLAB command line.
  - If you select **Evaluate this expression if there is an error** check box, then the complete report is generated and an error message appears at the MATLAB command line.

  To stop report generation when an error occurs in the MATLAB code that you added, change the second and third lines of the following default error handling code, as described below:

```
warningMessageLevel = 2;
displayWarningMessage = true;
failGenerationWithException = false;
failGenerationWithoutException = false;
```

To stop report generation and display an exception, change the default code to:

```
displayWarningMessage = false;
failGenerationWithException = true;
```

To stop report generation without displaying an exception, change the default code to:

```
displayWarningMessage = false;
failGenerationWithoutException = true;
```

If you want to completely replace the default error handling code, use the `evalException.message` variable in your code to return information for the exception.

## Insert Anything into Report?

Inserts text only if you select one of the following options:

- `Insert MATLAB expression string in report`
- `Display command window output in report`

## Class

`rptgen.cml_eval`

## See Also

`Insert Variable`, `MATLAB Property Table`, `MATLAB/Toolbox Version Number`, `Variable Table`

## Purpose

Apply child components to specified graphics figures

## Description

This component applies each child component to specified figures in the report. For more information about working with this component, see "Logical and Looping Components" on page 5-22.

## Figure Selection

- **Include figures**
  - `Current figure only`: Includes only the current figure in the report.
  - `Visible figures`: Loops on all visible figures. The `Data figures only` option is checked by default and excludes figures with `HandleVisibility = 'off'` from the loop.
  - `All figures with tags`: Loops on figures with specified tags, select When you select a given tag, all figures with that tag appear in the loop, regardless of whether each figure is visible or whether its `HandleVisibility` attribute is `'on'` or `'off'`.

    The tag field (located under `All figures with tags`) shows selected tags. To add tags to this field, type in the tag names, separating them with new lines.
- **Loop Figure List**: Shows all figures that are included in the loop. If the report setup file generates new figures or changes existing figures, figures in the **Loop Figure List** are not the figures that are reported on.

## Section Options

- **Create section for each object in loop**: Inserts a section in the generated report for each object found in the loop.
- **Display the object type in the section title**: Inserts the object type automatically into the section title in the generated report.
- **Create link anchor for each object in loop**: Creates a hyperlink to the object in the generated report.

### Insert Anything into Report?

Yes, inserts a section if you select the **Create section for each object in loop** option.

### Class

`rptgen_hg.chg_fig_loop`

### See Also

`Axes Loop`, `Axes Snapshot`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics Name`, `Handle Graphics Parameter`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

## Purpose

Insert snapshot of Handle Graphics figure into report

## Description

This component inserts a snapshot of a Handle Graphics figure into the report.

## Format

- **Image file format**: Specifies the image file format. Select `Automatic HG Format` to automatically choose the format best suited for the specified report output format. Otherwise, choose an image format that your output viewer can read. `Automatic HG Format` is the default option. Other options include:
  - `Automatic HG Format` (uses the Handle Graphics file format selected in the Preferences dialog box)
  - `Bitmap (16m-color)`
  - `Bitmap (256-color)`
  - `Black and white encapsulated PostScript`
  - `Black and white encapsulated PostScript (TIFF)`
  - `Black and white encapsulated PostScript2`
  - `Black and white encapsulated PostScript2 (TIFF)`
  - `Black and white PostScript`
  - `Black and white PostScript2`
  - `Color encapsulated PostScript`
  - `Color encapsulated PostScript (TIFF)`
  - `Color encapsulated PostScript2`
  - `Color encapsulated PostScript2 (TIFF)`
  - `Color PostScript`
  - `Color PostScript2`
  - `JPEG high quality image`
  - `JPEG medium quality image`

  - `JPEG low quality image`
  - `PNG 24-bit image`
  - `TIFF - compressed`
  - `TIFF - uncompressed`
  - `Windows metafile`
- **Capture picture from screen**:
  - `Client area only`: Captures a portion of the figure window.
  - `Entire figure window`: Captures the entire figure window.

## Print Options

- **Paper orientation**:
  - `Landscape`
  - `Portrait`
  - `Rotated`
  - `Use figure orientation`: Uses the orientation for the figure, which you set with the `orient` command.
  - `Full page image (PDF only)`: In PDF reports, scales images to fit the full page, minimizes page margins, and maximizes the size of the image by using either a portrait or landscape orientation.

  For more information about paper orientation, see the `orient` command in the MATLAB documentation.
- **Image size**:
  - `Use figure PaperPositionMode setting`: Uses the figure's `PaperPositionMode` property as the image size in the report. For more information about paper position mode, see the `orient` command in the MATLAB documentation.
  - `Automatic (same size as on screen)`: Sets the image in the report to the same size as it appears on the screen.

  - `Custom`: Specifies a custom image size. Set the image size in the **Size** field and **Units** list.

- **Size**: Specifies the size of the figure snapshot in the form `w h` (width times height). This field is active only if you choose `Custom` from the **Image size** selection list.
- **Units**: Specifies units for the size of the figure snapshot. This field is active only if you choose `Custom` in the **Image size** selection list.
- **Invert hardcopy**: Sets print colors using the figure's `InvertHardcopy` property, which inverts colors for printing. Options include:
  - `Automatic`: Automatically changes dark axes colors to light axes colors. If the axes color is a light color, it is not inverted.
  - `Invert`: Changes dark axes colors to light axes colors and vice versa.
  - `Don't invert`: Does not change the colors in the image.
  - `Use figure's InvertHardcopy setting`: Uses the value of the `InvertHardcopy` property set in the Handle Graphics image.
  - `Make figure background transparent`: Makes the image background transparent.

## Display Options

- **Scaling**:
  - `Fixed size`: Specifies the number and type of units.
  - `Zoom`: Specifies the percentage, maximum size, and units of measure.
  - `Use image size`: Causes the size of the image in the report to appear the same size as on screen.
- **Size**: Specifies the size of the snapshot in the format `w h` (width, height). This field is active only if you choose `Fixed size` in the **Scaling** list.

- **Max size**: Specifies the maximum size of the snapshot in the format `w h` (width, height). This field is active only if you choose `Zoom` from the **Scaling** list.
- **Units**: Specifies units for the size of the snapshot. This field is active only if you choose `Zoom` or `Fixed size` in the **Image size** selection list.
- **Alignment** Only reports in `PDF` or `RTF` format support this property. Options include:
  - `Auto`
  - `Right`
  - `Left`,
  - `Center`
- **Title**: Specifies a title for the figure:
  - `Custom`: Specifies a custom title.
  - `Name`: Specifies the figure name as the title.
- **Caption**: Specifies text to appear under the snapshot.

## Insert Anything into Report?

Yes. Image.

## Class

`rptgen_hg.chg_fig_snap`

## See Also

`Axes Loop`, `Axes Snapshot`, `Figure Loop`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics Name`, `Handle Graphics Parameter`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

## Purpose

Iteratively execute child components

## Description

This component functions like the MATLAB `for` loop, except that instead of executing a statement, it executes its child components. It must have at least one child component to execute.

## Loop Type

The loop type can have incremented indices or a vector of indices. For more information on for loops and indices, see `for` in the MATLAB documentation.

- **Incremented indices**: Executes a `for` loop of the form:

  ```
  for varname=x:y:z
  ```

  - **Start**: Corresponds to `x` in the previous expression.
  - **Increment**: Corresponds to `y` in the previous expression.
  - **End**: Corresponds to `z` in the previous expression.
- **Vector of Indices**: Executes a `for` loop of the form:

  ```
  for varname=[a b c ...]
  ```

  Specify appropriate values in the **Vector** field in the form `a b c ...`.

## Workspace Variable

- **Show index value in base workspace**: Displays the loop index in the MATLAB workspace while other components execute.
- **Variable name**: Allows you to specify the variable name. The default is `RPTGEN_LOOP`.
- **Remove variable from workspace when done**: Removes the loop index from the MATLAB workspace. This option is only available if you select the **Show index value in base workspace** option.

### Insert Anything into Report?

No.

### Class

`rptgen_lo.clo_for`

### See Also

`Logical Else`, `Logical Elseif`, `Logical If`, `Logical Then`, `While Loop`

# Graphics Object Loop

## Purpose

Run child components for each Handle Graphics object open in MATLAB workspace

## Description

This component runs its child components for each Handle Graphics object that is currently open in the MATLAB workspace. The component inserts a table into the generated report.

## Select Objects

- **Exclude GUI objects (uicontrol, uimenu, ...)**: Excludes graphical interface objects, such as `uicontrol` and `uimenu`, from the loop.
- **Loop list**: Specifies the loop level for Handle Graphics objects:
  - `Loop on objects with handle visibility "on"`
  - `Loop on all objects`
  - **Search for**: Allows you to enter space-delimited search terms.

## Section Options

- **Create section for each object in loop**: Inserts a section in the generated report for each object found in the loop.
- **Display the object type in the section title**: Inserts the object type automatically into the section title in the generated report.
- **Create link anchor for each object in loop**: Creates a hyperlink to the object in the generated report.

## Insert Anything into Report?

Yes, inserts a section if you select the **Create section for each object in loop** option.

## Class

`rptgen_hg.chg_obj_loop`

## See Also

`Axes Loop`, `Axes Snapshot`, `Figure Loop`, `Figure Snapshot`, `Handle Graphics Linking Anchor`, `Handle Graphics Name`, `Handle Graphics Parameter`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

# Handle Graphics Linking Anchor

## Purpose

Designate location to which links point

## Description

This component designates a location to which links point. It should have a looping component as its parent.

## Properties

- **Insert text**: Specifies text to appear after the linking anchor.
- **Link from current**: Sets the current model, system, block, or signal as the linking anchor:
  - `Automatic`: Automatically selects the appropriate figure, axes, or object as a linking anchor. If the `Figure Loop` component is this component's parent, the linking anchor points to the current figure. Similarly, if the `Graphics Object Loop` is this component's parent, the linking anchor points to the current object.
  - `Figure`: Sets the linking anchor to the current figure.
  - `Axes`: Sets the linking anchor to the current axes.
  - `Object`: Sets the linking anchor to the current object.

## Insert Anything into Report?

Yes. Anchor.

## Class

`rptgen_hg.chg_obj_anchor`

## See Also

`Axes Loop`, `Axes Snapshot`, `Figure Loop`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Name`, `Handle Graphics Parameter`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

# Handle Graphics Name

## Purpose

Insert name of Handle Graphics object into the report

## Description

This component inserts the name of a Handle Graphics object as text into the report. You can use this component to create a section title based on the current figure by making it the first child component of a `Chapter/Subsection` component, and then selecting the `Chapter/Subsection` component's **Get title from first child component** option.

## Properties

- **Display name as**:
  - `Type Name`
  - `Type  Name`
  - `Type:  Name`
- **Show name of current**:
  - `Figure`: Shows the name of the current figure. The first nonempty figure parameter determines the name.
  - `Axes`: Shows the name of the current axes. The first nonempty axes parameter determines the name.
  - `Other Object`: Sets the name of the current object to the figure's `CurrentObject` parameter and its first nonempty parameter.

## Insert Anything into Report?

Yes. Text.

## Class

`rptgen_hg.chg_obj_name`

## See Also

`Axes Loop`, `Axes Snapshot`, `Figure Loop`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics Parameter`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

# Handle Graphics Parameter

## Purpose

Insert property name/property value pair from Handle Graphics figure, axes, or other object

## Description

This component inserts a property name/property value pair from a Handle Graphics figure, axes, or other object.

## Property Selection

- **Get property from current**: Reports on a specified Handle Graphics object:
  - `Figure`: Inserts a figure's property name/property value pair.
  - `Axes`: Inserts an axes' property name/property value pair.
  - `Object`: Inserts an object's property name/property value pair.
- **Figure property**: Specifies the type of property to include. The `All` option shows every parameter for the current object.

## Display Options

- **Title**: Specifies a title for the generated report:
  - `None` (default): No title.
  - `Automatic`: Automatically generates the title from the parameter.
  - `Custom` Specifies a custom title.
- **Size limit**: Limits the width of the display in the generated report. Units are in pixels. The size limit of a given table is the hypotenuse of the table width and height [`sqrt(w^2+h^2)`]. The size limit of a text string equals its number of characters squared. If you exceed the size limit, the variable appears in condensed form, such as `[64x64 double]`. Setting a size limit of `0` displays the variable in full, no matter how large it is.
- **Display as**: Choose a display style:
  - `Auto table/paragraph`: Displays as a table or paragraph.
  - `Inline text`: Displays in line with the surrounding text.
  - `Paragraph`: Displays as a paragraph.
  - `Table`: Displays as a table.

- **Ignore if value is empty**: Excludes empty parameters from the generated report.

## Insert Anything into Report?

Yes. Text.

## Class

`rptgen_hg.chg_property`

## See Also

`Axes Loop`, `Axes Snapshot`, `Figure Loop`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics Name`, `Handle Graphics Property Table`, `Handle Graphics Summary Table`

# Handle Graphics Property Table

## Purpose

Insert table that reports on property name/property value pairs

## Description

This component inserts a table that reports on property name/property value pairs.

For more information on using this component, see "Property Table Components" on page 5-6.

## Select Graphics Object

- **Object type**: Specifies an object type for the generated report:
  - `Figure`
  - `Axes`
  - `Object`
- **Filter by class**: Specifies a class or classes for the table.

## Table

You can select a preset table, which is already formatted and set up, from the list in the upper-left corner of the attributes page.

To create a custom table, edit a preset table, such as `Blank 4x4`. Add and delete rows and add properties. To open the Edit Table dialog box, click **Edit**.

For details about creating custom property tables, see "Property Table Components" on page 5-6.

- **Preset table**: Specifies the type of table to display the object property table.
  - `Defaults`
  - `Callbacks`
  - `Graphics`
  - `Printing`
  - `Blank 4x4`

  To apply a preset table, select the table and click **Apply**.

- **Split property/value cells**: Splits property name/property value pairs into separate cells. For the property name and property value to appear in adjacent horizontal cells in the table, select the **Split property/value cells** check box. In this case, the table is in split mode and there can be only one property name/property value pair in a cell. If there is more than one name/property pair in a cell, only the first pair appears in the report. All subsequent pairs are ignored.

  For the property name and property value to appear together in one cell, clear the **Split property/value cells** check box. In this case, the table is in nonsplit mode, which supports more than one property name/property value pair per cell. It also supports text.

  Before switching from nonsplit mode to split mode, make sure that you have only one property name/property value pair per table cell. If you have more than one property name/property value pair or any text, only the first property name/property value pair appears in the report; subsequent pairs and text are omitted.

- **Display outer border**: Displays the outer border of the table in the generated report.

## Table Cells

Select table properties to modify. The selection in this pane affects the availability of fields in the **Title Properties** pane.

If you select `Figure Properties`, only the **Contents** and **Show** options appear. If you select any other object in the **Table Cells** pane, the **Lower border** and **Right border** options appear.

## Title Properties

- **Contents**: Enables you to modify the contents of the table cell selected in the **Table Cells** pane. Options include:
  - `Left`
  - `Center`
  - `Right`
  - `Double justified`

- **Show as**: Enables you to choose the format for the contents of the table cell. Options include:
  - `Value`
  - `Property Value`
  - `PROPERTY Value`
  - `Property:  Value`
  - `PROPERTY: Value`
  - `Property - Value`
  - `PROPERTY - Value`
- **Alignment**: Aligns text in the table cells. Options are:
  - `Left`
  - `Center`
  - `Right`
  - `Double-justified`
- **Lower border**: Displays the lower border of the table in the generated report.
- **Right border**: Displays the right border of the table in the generated report.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen_hg.chg_prop_table`

## See Also

`Axes Loop`, `Axes Snapshot`, `Figure Loop`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics Name`, `Handle Graphics Parameter`, `Handle Graphics Summary Table`

# Handle Graphics Summary Table

## Purpose

Insert table that summarizes Handle Graphics object properties

## Description

This component inserts a table that summarizes Handle Graphics object properties. Each row in the table represents an object. Each column in the table represents a property. You can specify object properties to include in the report.

## Properties

- **Object type**: Specifies the object type to display in the generated report. Options include:
  - `figure`
  - `axes`
  - `object`

  The available options in the **Select Objects** pane depend on your selection in the **Object type** menu.
- **Table title**: Specifies a title for the table in the generated report. Options include:
  - `Automatic`: Generates a title automatically.
  - `Custom`: Specifies a custom title.

## Property Columns

- **Property columns**: Specifies object properties to include in the table in the generated report.
  - To add a property:
    1. Select the appropriate property level in the menu.
    2. In the list under the menu, select the property to add and click **Add**.
  - To delete a property, select its name and click **Delete**.

  Some entries in the list of available properties (such as `Depth`) are “virtual” properties which you cannot access using the `get_param` command. The properties used for property/value filtering in the block and system loop components must be retrievable by the get_param. Therefore, you cannot configure your summary table to report on all blocks of Depth == 2.

- **Remove empty columns**: Removes empty columns from the table in the generated report.
- **Transpose table**: Changes the summary table rows into columns in the generated report, putting the property names in the first column and the values in the other columns.

## Object Rows

**Insert anchor for each row**: Inserts an anchor for each row in the summary table.

## Figure Selection

The options displayed in the **Figure Selection** pane depend on the object type selected in the **Object type** list:

- If **Object type** is `figure`, the following options appear:
  - **Include figures**
    - `Current figure only`: Includes only the current figure in the report.
    - `Visible figures`: Executes child components for figures that are currently open and visible. The `Data figures only` option is checked by default. This option excludes figures with `HandleVisibility = 'off'` from the loop.
    - `All figures with tags`: Includes all figures with a specified tag regardless of whether they are visible or their `HandleVisibility` parameter is `'on'` or `'off'`. The tag selection list, located under this option, shows available tags. You can add tag names to this list.
  - **Data figure only (Exclude applications)**: Shows only data figures.
  - **Loop Figure List**: Shows figures within the current set of figures to display.
- If **Object type** is `axes`, the following options appear:

- **Loop type**:
  - `All axes`: Loops on all axes objects.
  - `Current axes`: Loops on the selected axes object.
- **Exclude objects which subclass axes**: Excludes objects such as legends and color bars.
- **Loop Menu**:
  - `Loop on axes with handle visibility "on"`: Loops on visible axes objects.
  - `Loop on all axes`: Loops on all axes objects.
- **Search terms**: Specifies search terms for the loop. For example, to search for `Tag` and `My Data`, enter `"Tag", "My Data"`.

- If **Object type** is `object`, the following options appear:
  - **Exclude GUI objects (uicontrol, uimenu, ...)**: Excludes graphical interface objects, such as `uicontrol` and `uimenu`, from the loop.
  - **Loop menu**: Specifies the loop level:
    - `Loop on objects with handle visibility "on"`
    - `Loop on all objects`
  - **Search for**: Specifies space-delimited search terms.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen_hg.chg_summ_table`

## See Also

`Axes Loop`, `Axes Snapshot`, `Figure Loop`, `Figure Snapshot`, `Graphics Object Loop`, `Handle Graphics Linking Anchor`, `Handle Graphics`

# Handle Graphics Summary Table

Name, Handle Graphics Parameter, Handle Graphics Property Table

## Purpose

Insert image from external file into report

## Description

This component inserts an image from an external file into the report. It can have the `Chapter/Subsection` or `Paragraph` component as its parent. If the `Paragraph` component is its parent, you must select the **Insert as inline image** check box.

## Class

- **File name**: Specifies the image file name. You can enter this name manually, or use the **Browse** button **(...)** to find the image file.

  The image must be in a format that your viewer can read. An error like the following appears if you specify the name of an image file that does not exist:

  ```
  No file name.  Could not create graphic.
  ```

  This field supports `%<VariableName>` notation. For more information about this notation, see "%<VariableName> Notation" on page 9-90 on the `Text` component reference page.

- **Copy to local report files directory**: Saves a copy of the image to a local report files folder.

## Display Options

- **Scaling**: Controls size of the image, as displayed in a browser. Making an image larger using this option does not affect the storage size of the image, but the quality of the displayed image may decrease as you increase or decrease the size of the displayed image.

  Generally, to achieve the best and most predictable display results, use the default setting of `Use image size`.

  - `Use image size`: Causes the image to appear the same size in the report as on screen (default).
  - `Fixed size`: Specifies the number and type of units.
  - `Zoom`: Specifies the percentage, maximum size, and units of measure.

- **Size**: Specifies the size of the snapshot in the format `w h` (width, height). This field is active only if you choose `Fixed size` from the **Scaling** list.
- **Max size**: Specifies the maximum size of the snapshot in the format `w h` (width, height). This field is active only if you choose `Zoom` from the **Scaling** list;
- **Units**: Specifies units for the size of the snapshot. This field is active only if you choose `Zoom` or `Fixed size` in the **Image size** selection list.
- **Alignment**: Only reports in `PDF` or `RTF` formats support this format property. Options are:
  - `Auto`
  - `Right`
  - `Left`
  - `Center`
- **Title**: Specifies text to appear above the snapshot.
- **Caption**: Specifies text to appear under the snapshot.
- **Full page image (PDF only)**: In PDF reports, scales images to fit the full page, minimizes page margins, and maximizes the size of the image by using either a portrait or landscape orientation.

### Preview

The image that you specify in the **Image file name** field appears in this pane. You cannot preview Adobe PostScript images, or images with formats that the `imread` function does not support, such as `.gif`.

Clicking an image causes it to display in full size.

### Insert Anything into Report?

Yes. Image.

## Class

rptgen.cfr_image

## See Also

Chapter/Subsection, Empty Component, List, Paragraph, Table, Text, Title Page

# Import File

## Purpose

Import `ASCII` text file into report

## Description

This component imports an `ASCII` text file into the report.

## Properties

- **File name**: Specifies the name of the file to import into the text field. You can enter a name, or use the **Browse** button **(...)** to find the file.
- **Import file as**: Specifies formatting for the imported file. Options include:
  - `Plain text (ignore line breaks)`: Imports the file as plain text without any line breaks (no paragraphs). If you select this option, the `Import File` component acts like the `Text` component, so it should have the `Paragraph` component as its parent.

    The examples in this section use the following text as the input file:

    ```
    This is the first row of text from the imported file.
      The second row follows a line break in the first row.

    There is a blank line above the third row.
    ```

    `Plain text (ignore line breaks)` produces the following formatting for the example file:

    ```
    This is the first row of text from the imported file.
    The second row follows a line break in the first row.

    There is a blank line above the third row.
    ```

  - `Paragraphs defined by line breaks`: Imports the file as text, in paragraphs with line breaks (hard returns or carriage returns). This option produces the following formatting for the example file:

    ```
    This is the first row of text from the imported file.
    The second row follows a line break in the first row.

    There is a blank line above the third row.
    ```

- `Paragraphs defined by empty rows`: Imports the file as text, in paragraphs with empty rows (rows that include no text). This option produces the following formatting for the example file:

  ```
  This is the first row of text from the imported file.
  The second row follows a line break in the first row.

  There is a blank line above the third row.
  ```

- `Text (retain line breaks)` (default): Imports the file as plain text with line breaks. This option produces the following formatting for the example file:

  ```
  This is the first row of text from the imported file.
  The second row follows a line break in the first row.

  There is a blank line above the third row.
  ```

- `Fixed-width text (retain line breaks)`: Imports the file as fixed-width text (all letters have the same width or size), including line breaks. This option is useful for importing MATLAB files. This option produces the following formatting for the example file:

  ```
  This is the first row of text from the imported file.
  The second row follows a line break in the first row.

  There is a blank line above the third row.
  ```

- `DocBook XML`: Inserts an XML source file, and makes no changes to its format.
- `Formatted Text (RTF/HTML)`: Inserts an `RTF` or `HTML` source file, and makes no changes to its format.
- `Syntax highlighted MATLAB code`: Inserts a MATLAB file.

The **File Contents** field displays the first few lines of the file to be imported.

### Insert Anything into Report?

Yes.

- Inserts text if you select one of the following options:
  - `Plain text (ignore line breaks)`
  - `Text (retain line breaks)`
  - `Fixed-width text (retain line breaks)`
- Inserts paragraphs if you select one of the following options:
  - `Paragraphs defined by line breaks`
  - `Paragraphs defined by empty rows`
- Inserts the contents of an XML file if you select the `DocBook XML` option.
- Inserts the contents of the `RTF` or `HTML` file if you select the `Formatted text (RTF/HTML)` option.
- Inserts a link to a file if you import the file into an HTML report.

### Class

`rptgen.crg_import_file`

### See Also

`Comment`, `Nest Setup File`, `Stop Report Generation`, `Time/Date Stamp`

## Purpose

Insert variable values into report

## Description

This component inserts the value (and, optionally, the name) of each the following variables into the report:

- A variable from the MATLAB workspace
- A variable from a MAT-file
- A global variable
- A variable that you specify directly

## Source

- **Variable name**: Specifies the name of the variable:
  - `%<VariableName>`: Inserts the value of a variable from the MATLAB workspace into the report.
- **Variable location**:
  - `Base Workspace`: Gets a variable from the MATLAB workspace.
  - `MAT File`: Gets a variable from a binary file with a `.mat` extension.
  - `Global variable`: Gets a global variable.
  - `Direct`: Gets a variable that you specify directly.

## Display Options

- **Title**: Specify a title for the report:
  - `Automatic`: Generates a title automatically.
  - `Custom`: Specifies a custom title.
  - `None`: Specifies no title.
- **Array size limit**: Limits the width of the display in the generated report. Units are in pixels. The size limit for a given table is the hypotenuse of the table width and height `[sqrt(w^2+h^2)]`. The size limit of a given text string is the number of characters squared. If you exceed the size limit, the variable appears in condensed form, such as `[64x64 double]`. Setting a size limit of `0` displays the variable in full, regardless of its size.

- **Object depth limit**: Specifies the maximum number of nesting levels to report on for a variable value
- **Object count limit**: Specifies the maximum number of nested objects to report on for a variable value
- **Display as**: Choose a display style from the menu:
  - `Table`: Displays as a table.
  - `Paragraph`: Displays as a paragraph.
  - `Inline text`: Displays inline with the surrounding text.
  - `Table or paragraph depending on data type`: Displays as a table or paragraph.
- **Show variable type in headings**: Show data type of this variable in the title of its report.
- **Show variable table grids**: Show grid lines for the table used to report the value of this variable.
- **Make variable table page wide**: Make the variable table as wide as the page on which the table appears.
- **Omit if value is empty**: Exclude empty parameters from the generated report.
- **Omit if property default value**: Exclude object property from the report if that property uses the default value.

## Insert Anything into Report?

Yes. Text.

## Class

`rptgen.cml_variable`

## See Also

`Evaluate MATLAB Expression`, `MATLAB Property Table`, `MATLAB/Toolbox Version Number`, `Variable Table`

## Purpose

Insert linking anchors or pointers into report

## Description

This component inserts linking anchors or pointers into the report.

For a PDF report, if you open the report from MATLAB (for example, if you open the report right after generating it), the link does not work. However, if you open a PDF report outside of MATLAB (for example, from Adobe Acrobat), the link works properly.

## Properties

- **Link type**: Select the type of link to insert into the report. Options include:
  - `Linking anchor`: Specifies a link to an anchor.
  - `Internal document link`: Specifies a location in the report (as specified by an anchor).
  - `URL (external) link`: Specifies a link to a Web site or to a MATLAB command to execute from generated report.
- **Link identifier**: Indicates the location to which the link points. It can contain only `ASCII` characters, and it is not visible in the generated report.

  For a Web link, the link identifier options are context sensitive; their formats differ depending on the link type you select. For example, to link to an external file `foo.txt`, specify the link identifier as follows:

  - On UNIX systems:

    ```
    file:///home/janedoe/foo.txt
    ```

  - On Microsoft Windows systems:

    ```
    H:\foo.txt
    ```

  For a link to a MATLAB command, enter `matlab:` followed by a space and the MATLAB command that you want the link to execute.

- **Link text**: Specifies text to use in the link.
- **Emphasize link text**: Italicizes the link text.

## Examples

### Link to an External Web Site

1. Open Report Explorer with the `setedit` command.

2. In the Properties pane on the right, click **Create and edit a new report file**.

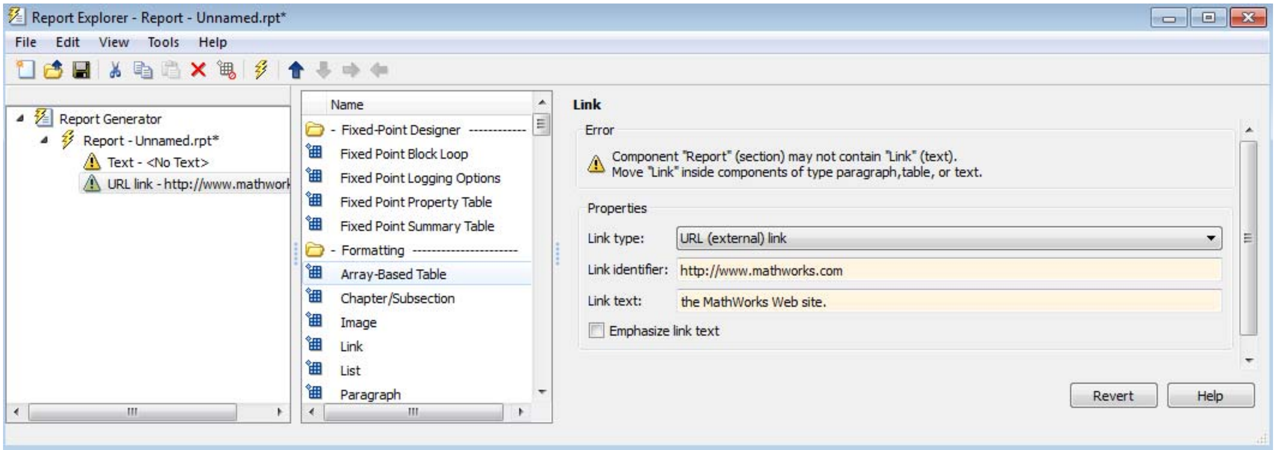
3. In the Library pane in the middle, under the Formatting category, select the Text component and click the **Add component** icon.

4. In the Properties pane, enter `Open the ` (add a blank space at the end of the string).

5. In the Library pane, under the Formatting category, select the Link component and click the **Add component** icon.

6. In the Properties pane:

   - Set **Link type** to `URL (external) link`.
   - In **Link Identifier**, enter `http://www.mathworks.com`.
   - In **Link text**, enter `MathWorks Web site.` (with a period).



Link

Error

Component "Report" (section) may not contain "Link" (text).
Move "Link" inside components of type paragraph, table, or text.

Properties

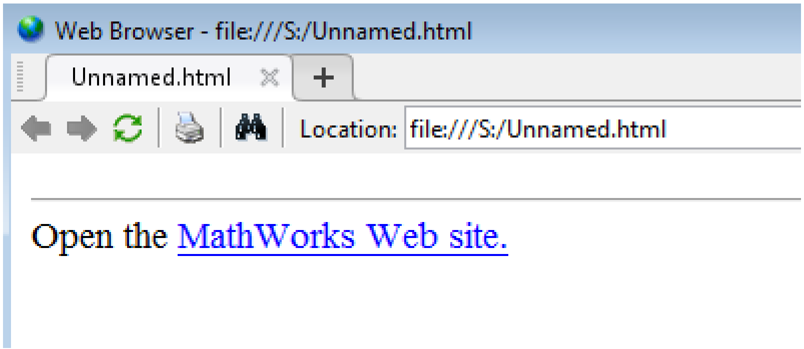
Link type: URL (external) link

Link identifier: http://www.mathworks.com

Link text: the MathWorks Web site.

□ Emphasize link text

7 Generate the report.



Location: file:///S:/Unnamed.html

Open the MathWorks Web site.

8 Click the link to open the MathWorks Web site.

## Link to Another Place in a Report

1 At the MATLAB command line, enter `setedit magic-square.rpt`.

2 In the Outline pane on the left, select the Title Page component.

3. In the Library pane in the middle, under the Formatting category, select the Link component and click the **Add component** icon.

4. In the Properties pane:

   - Set **Link type** to `Linking anchor`.
   - In **Link identifier**, enter `explanation`.

   In the Contents pane, the Link component appears as `Anchor - explanation`.

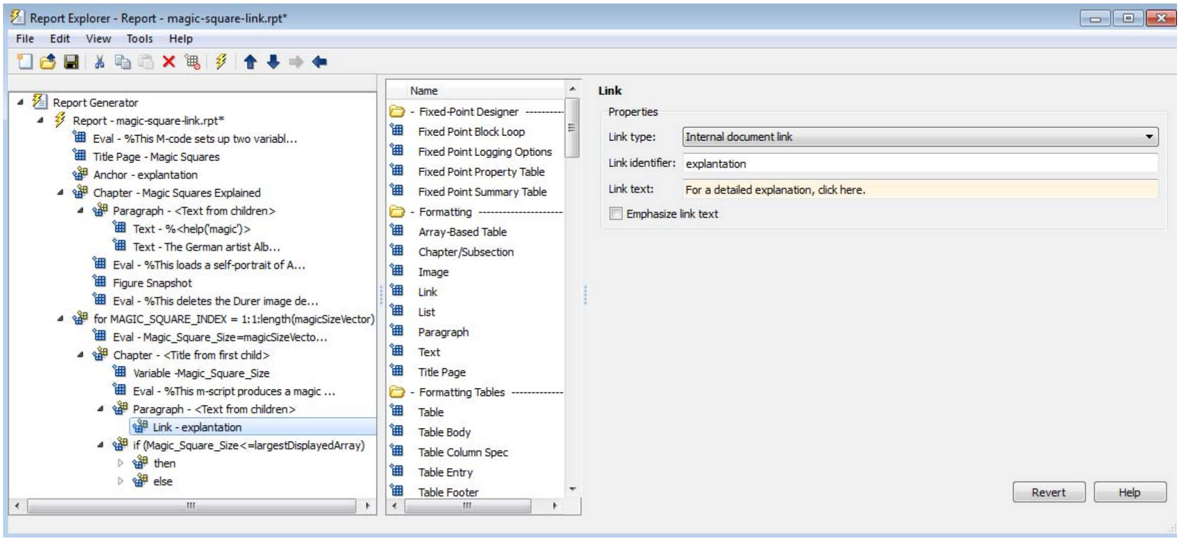
5. In the Outline pane, under the second Chapter component, select the Eval component.

6. In the Library pane, under the Formatting category, select the Paragraph component and click the **Add component** icon.

7. In the Library pane, under the Formatting category, select the Link component and click the **Add component** icon.

8. In the Properties pane:

   - Set **Link type** to `Internal document link`.
   - In **Link identifier**, enter `explanation`.
   - In **Link text**, enter `For a detailed explanation, click here.` (with the period).


Report Explorer - Report - magic-square-link.rpt*
File Edit View Tools Help
Report Generator
Report - magic-square-link.rpt*
Eval - %This M-code sets up two variabl...
Title Page - Magic Squares
Anchor - explantation
Chapter - Magic Squares Explained
Paragraph - <Text from children>
Text - %<help('magic')>
Text - The German artist Alb...
Eval - %This loads a self-portrait of A...
Figure Snapshot
Eval - %This deletes the Durer image de...
for MAGIC_SQUARE_INDEX = 1:1:length(magicSizeVector)
Eval - Magic_Square_Size=magicSizeVecto...
Chapter - <Title from first child>
Variable -Magic_Square_Size
Eval - %This m-script produces a magic ...
Paragraph - <Text from children>
Link - explantation
if (Magic_Square_Size<=largestDisplayedArray)
then
else
Name
- Fixed-Point Designer
Fixed Point Block Loop
Fixed Point Logging Options
Fixed Point Property Table
Fixed Point Summary Table
- Formatting
Array-Based Table
Chapter/Subsection
Image
Link
List
Paragraph
Text
Title Page
- Formatting Tables
Table
Table Body
Table Column Spec
Table Entry
Table Footer
Link
Properties
Revert
Help

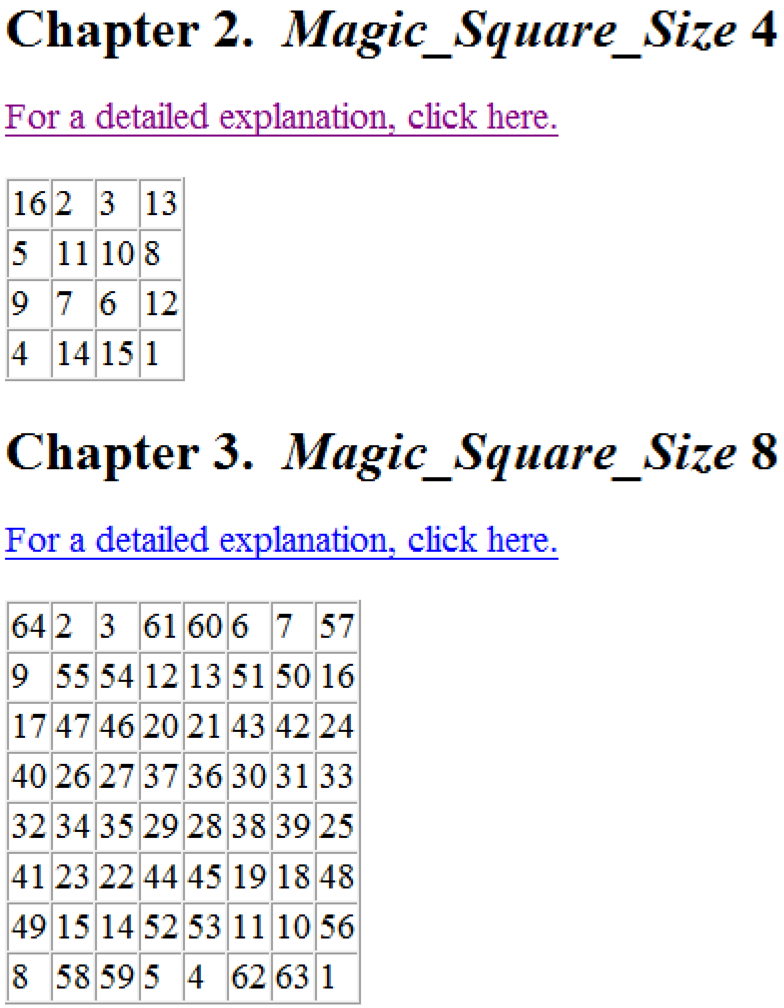
Link type: Internal document link

Link identifier: explantation

Link text: For a detailed explanation, click here.

□ Emphasize link text

**9** Generate the report.



## Chapter 2. *Magic_Square_Size* 4

For a detailed explanation, click here.

| | | | |
|---|---|---|---|
| 16 | 2 | 3 | 13 |
| 5 | 11 | 10 | 8 |
| 9 | 7 | 6 | 12 |
| 4 | 14 | 15 | 1 |

## Chapter 3. *Magic_Square_Size* 8

For a detailed explanation, click here.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 64 | 2 | 3 | 61 | 60 | 6 | 7 | 57 |
| 9 | 55 | 54 | 12 | 13 | 51 | 50 | 16 |
| 17 | 47 | 46 | 20 | 21 | 43 | 42 | 24 |
| 40 | 26 | 27 | 37 | 36 | 30 | 31 | 33 |
| 32 | 34 | 35 | 29 | 28 | 38 | 39 | 25 |
| 41 | 23 | 22 | 44 | 45 | 19 | 18 | 48 |
| 49 | 15 | 14 | 52 | 53 | 11 | 10 | 56 |
| 8 | 58 | 59 | 5 | 4 | 62 | 63 | 1 |

**10** Click the link to move to near the top of the report, to “Chapter 1. Magic Squares Explained.”.

### Link to a Model

This example shows how to add a link to a Simulink model. To view the model, you must have the Simulink software installed.

1. Open Report Explorer with the `setedit` command.

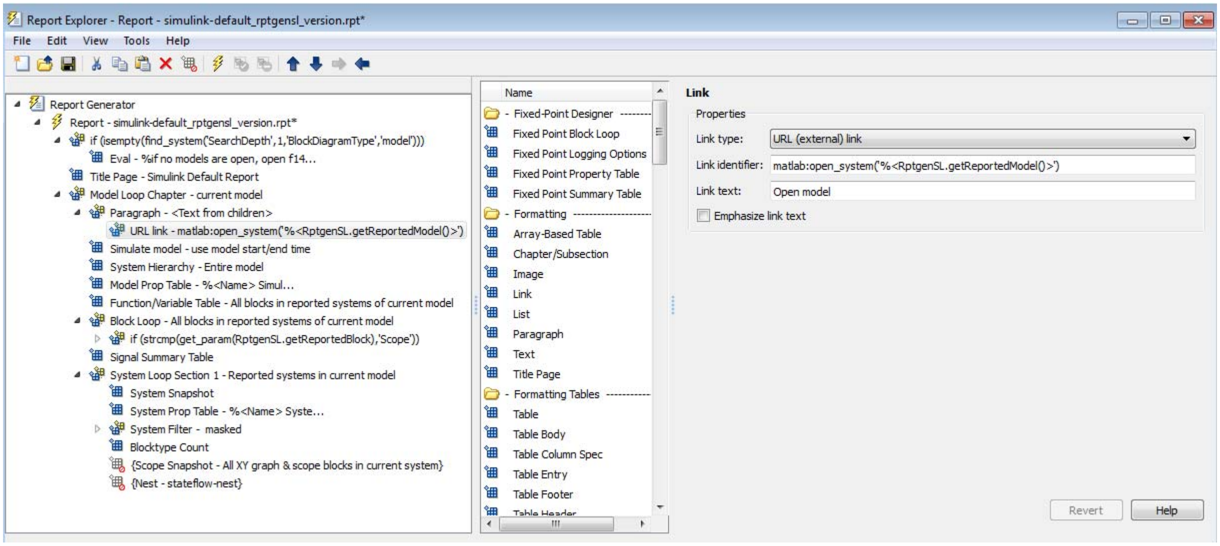
2. In the middle pane, click `simulink-default_rptgensl_version.rpt`.

3. In the Library pane in the middle, under the Formatting category, select the Paragraph component and click the **Add component** icon.

4. In the Library pane in the middle, under the Formatting category, select the Link component and click the **Add component** icon.

5. In the Properties pane:

   - Set **Link type** to `URL (external) link`.
   - In **Link Identifier**, enter `matlab:open_system('%<RptgenSL.getReportedModel()>')`.
   - In **Link text**, enter `Open model`.


Report Explorer - Report - simulink-default_rptgensl_version.rpt*
File
Edit
View
Tools
Help
Name
Revert
Help

```
Report Generator
  Report - simulink-default_rptgensl_version.rpt*
    if (isempty(find_system('SearchDepth',1,'BlockDiagramType','model')))
      Eval - %if no models are open, open f14...
    Title Page - Simulink Default Report
    Model Loop Chapter - current model
      Paragraph - <Text from children>
        URL link - matlab:open_system('%<RptgenSL.getReportedModel()>')
      Simulate model - use model start/end time
      System Hierarchy - Entire model
      Model Prop Table - %<Name> Simul...
      Function/Variable Table - All blocks in reported systems of current model
      Block Loop - All blocks in reported systems of current model
        if (strcmp(get_param(RptgenSL.getReportedBlock),'Scope'))
      Signal Summary Table
      System Loop Section 1 - Reported systems in current model
        System Snapshot
        System Prop Table - %<Name> Syste...
        System Filter - masked
        Blocktype Count
        {Scope Snapshot - All XY graph & scope blocks in current system}
        {Nest - stateflow-nest}
```

```
- Fixed-Point Designer --------
Fixed Point Block Loop
Fixed Point Logging Options
Fixed Point Property Table
Fixed Point Summary Table
- Formatting -------------------
Array-Based Table
Chapter/Subsection
Image
Link
List
Paragraph
Text
Title Page
- Formatting Tables -----------
Table
Table Body
Table Column Spec
Table Entry
Table Footer
Table Header
```

**Link**

Properties

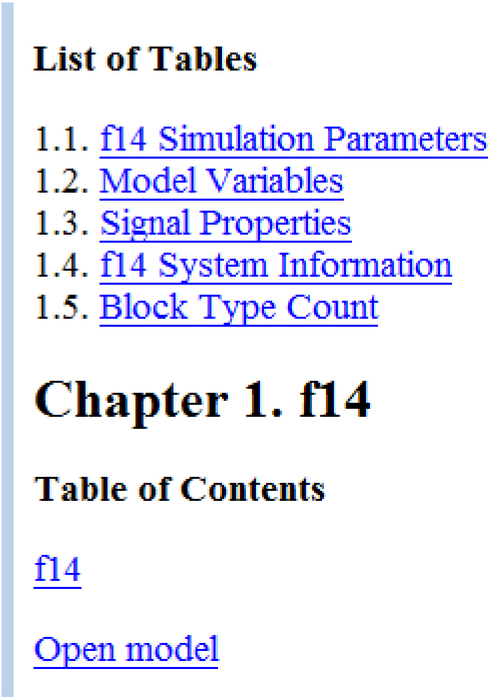
Link type: URL (external) link

Link identifier: matlab:open_system('%<RptgenSL.getReportedModel()>')

Link text: Open model

□ Emphasize link text

**6** Generate the report.



### List of Tables

1.1. f14 Simulation Parameters
1.2. Model Variables
1.3. Signal Properties
1.4. f14 System Information
1.5. Block Type Count

# Chapter 1. f14

### Table of Contents

f14

Open model

**7** Click the “Open model” link to open the model.

## Insert Anything into Report?

Yes. Text or anchor.

## Class

`rptgen.cfr_link`

## See Also

`Chapter/Subsection`, `Empty Component`, `List`, `Paragraph`, `Table`, `Text`, `Title Page`

## Purpose

Create bulleted or numbered list from cell array or child components

## Description

This component creates a bulleted or numbered list from a cell array or child components.

## List Content

- **Create list from workspace cell array**: Creates the list from of the 1-by-n or n-by-1 cell array. This option is not available when this component has child components — in this case, the list automatically generates from the child components.
- **List title**: Specifies the title of the list.

## List Formatting

- **List style**:
  - `Bulleted list`
  - `Numbered list`.
- **Numbering style**: Specifies a numbering style for numbered lists. This setting is supported only in the RTF/DOC report format. Options include:
  - `1,2,3,4,...`
  - `a,b,c,d,...`
  - `A,B,C,D,...`
  - `i,ii,iii,iv,...`
  - `I,II,III,IV,...`
- **Show parent number in nested list (1.1.a)**: Displays all level numbers in a nested list. You can create a nested list by putting one cell array inside another or by nesting one `List` component inside another. Following is an example of how a list appears when you select this option:

```
1. Example
2. Example
 2.1. Example
 2.2. Example
```

```
  2.2.a. Example
  2.2.b. Example
3. Example
```

This option is not available if you select `Show only current list value (a)`.

- **Show only current list value (a)**: Displays only the current list value. Following is an example of how a list appears when you select this option:

```
1. Example
2. Example
 1. Example
 2. Example
  1. Example
  2. Example
3. Example
```

This option is not available if you select `Show parent number in nested list (1.1.a)`.

### Example 1: Creating a Nested List

Consider the following report setup file, which includes a nested list created by putting a `List` component inside another `List` component:

```
[-] Report - Unnamed.rpt
   [-] Bulleted list from child components
      [ ] Text - sky
      [ ] Table - varname
      [ ] Image - test.jpg
      [ ] Text - grass
      [-] Bulleted list from child components
         [ ] Text - clouds
         [ ] Text - sun
      [-] Paragraph - information
```

This report setup file generates a report that includes the following bulleted lists:

- `sky`
- `varname`, the table from the variable
- `test.jpg`, a snapshot of the image
- `grass`
  - `clouds`
  - `sun`
- `information`

### Example 2: Creating a List Using Child Components

To generate a report that includes the following bulleted list:

- `red`
- `green`
- `blue`

Use the following report setup file:

```
[-] Report - Unnamed.rpt
   [-] Bulleted list from child components
      [ ] Text - red
      [ ] Text - green
      [ ] Text - blue
```

### Creating a List Using a Cell Array

To generate the same bulleted list as in the previous example, configure a report setup file to call a cell array, `colors`:

```
[-] Report - Unnamed.rpt
   [-] Bulleted list from cell array called colors
```

Wherecolors is:

```
colors={'red','green','blue'}
```

## Insert Anything into Report?

Yes. List.

## Class

`rptgen.cfr_list`

## See Also

`Chapter/Subsection`, `Empty Component`, `Link`, `Paragraph`, `Table`, `Text`, `Title Page`

# Logical Else

## Purpose

Specify an `else` condition for a `Logical If` component

## Description

This component acts as an `else` when it is the child of the `Logical If` component. You can specify this component in one of the following ways:

-
  ```
  if
   then
   else
  ```

-
  ```
  if
   then
   elseif
   elseif
       .
       .
       .
   else
  ```

## Properties

**If component has no children, insert text**: Inserts specified text into your report when the `Logical Else` component has no child components. In this case, this component acts like the `Text` component.

## Insert Anything into Report?

Yes, when `if` or `elseif` statement is false.

## Class

`rptgen_lo.clo_else`

## See Also

`For Loop`, `Logical Elseif`, `Logical If`, `Logical Then`, `While Loop`

## Purpose

Specify an elseif condition for a Logical If component

## Description

This component acts as an elseif when it is the child of the Logical If component. You must specify this component as follows:

```
if
 then
 elseif
 elseif
    .
    .
    .
 else
```

## Properties

- **Test expression**: Specifies a MATLAB expression to evaluate.
- **If component has no children, insert text**: Inserts the specified text into the report when the Logical Elseif component has no child components. In this case, this component acts like the Text component.

## Insert Anything into Report?

Yes, when parent if statement is false.

## Class

rptgen_lo.clo_else_if

## See Also

For Loop, Logical Else, Logical If, Logical Then, While Loop

# Logical If

## Purpose

Specify `logical if` condition

## Description

This component acts as a logical `if`; it can have the `Logical Then`, `Logical Elseif`, or `Logical Else` components as children components. This component executes its child components when the specified workspace expression is true. It displays a specified string when it has no child components. You can specify this component as follows:

-
  ```
  if
   then
  ```

-
  ```
  if
   then
   else
  ```

-
  ```
  if
   then
   elseif
   elseif
      .
      .
      .
   else
  ```

## Properties

- **Test expression**: Specifies a MATLAB expression to evaluate.
- **If component has no children, insert text**: Inserts specified text into the report when the `Logical If` component has no children.

## Insert Anything into Report?

Depends on specified attribute values.

## Class

`rptgen_lo.clo_if`

## See Also

`For Loop`, `Logical Else`, `Logical Elseif`, `Logical Then`, `While Loop`

# Logical Then

## Purpose

Specify a `then` condition for a `Logical If` component

## Description

This component acts as a `then` when it is the child of the `Logical If` component. You can specify this component as follows:

-
  ```
  if
   then
  ```

-
  ```
  if
   then
   else
  ```

-
  ```
  if
   then
   elseif
   elseif
       .
       .
       .
   else
  ```

## Attributes

**If component has no children, insert text**: Inserts specified text into the report when the `Logical Then` component has no children. In this case, this component acts like the `Text` component.

## Insert Anything into Report?

Yes, when parent `if` statement is true.

## Class

`rptgen_lo.clo_then`

## See Also

`For Loop`, `Logical Else`, `Logical Elseif`, `Logical If`, `While Loop`

# MATLAB Property Table

## Purpose

Insert table that includes MATLAB object property name/property value pairs

## Description

This component inserts a table that includes MATLAB object property name/property value pairs.

## Table

Select a preset table, which is already formatted and set up, in the preset table list in the upper-left corner of the attributes page.

- **Preset table**: Choose a type of table:
  - `Default`
  - `Blank 4x4`

  To apply the preset table, select the table and click **Apply**.
- **Split property/value cells**: Splits property name/property value pairs into separate cells. Select the **Split property/value cells** check box for the property name and property value to appear in adjacent cells. In this case, the table is in split mode; only one property name/property value pair per cell is allowed. If more than one name/property pair exists in a cell, only the first pair appears in the report; subsequent pairs are ignored.

  Clear the **Split property/value cells** check box for a given property name and property value to appear together in one cell. In this case, the table is in nonsplit mode, which supports more than one property name/property value pair. It also supports text.

  Before switching from nonsplit mode to split mode, make sure that you have only one property name/property value pair per table cell.
- **Display outer border**: Displays the outer border of the table in the generated report.
- **Table Cells**: Modifies table properties. The selection in this pane affects the available fields in the **Cell Properties** pane.

## Cell Properties

Available options in the **Cell Properties** pane depend what you select for **Table Cells**. If you select `Workspace Properties`, only the **Contents** and **Show** options appear. If you select any other option, the **Lower border** and **Right border** options appear.

- **Contents**: Modifies the contents of the table cell selected in the **Table Cells** pane.
- **Show as**: Specifies the format for the contents of the table cell. Options include:
  - `Value`
  - `Property Value`
  - `PROPERTY Value`
  - `Property:  Value`
  - `PROPERTY: Value`
  - `Property - Value`
  - `PROPERTY - Value`
- **Alignment**: Specifies how to align the contents of the selected table cell in the **Table Cells** field. Options include:
  - `Left`
  - `Center`
  - `Right`
  - `Double justified`
- **Lower border**: Displays the lower border of the table in the generated report.
- **Right border**: Displays the right border of the table in the generated report.

### Creating Custom Tables

To create a custom table, edit a preset table, such as `Blank 4x4`. You can add and delete rows and add properties. To open the Edit Table dialog box, click **Edit**.

For details about using this dialog box to create custom property tables, see “Property Table Components” on page 5-6.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cml_prop_table`

## See Also

`Evaluate MATLAB Expression`, `Insert Variable`, `MATLAB/Toolbox Version Number`, `Variable Table`

## Purpose

Insert table that shows version and release numbers and release date of MathWorks products

## Description

Using the Table Filter, specify whether this component reports version information for all installed MathWorks products or just those products required for a model.

For the specified set of products, this component inserts a table showing any of these columns that you specify:

- Version number
- Release number
- Release date
- Is required for model

You can list all your MathWorks products by typing `ver` at the MATLAB command line.

## Table Title

**Table title**: Specifies the table title. The default is `version number`.

## Table Filter

**Show only toolboxes required for model**: When you select this option, the report shows version information for only those products required for a model or chart. By default, the report shows version information for all installed MathWorks products.

---

**Note** This option uses the Simulink Manifest Tools analysis to determine what products appear in the version information table. See "Analysis Limitations" for Manifest Tools analysis limitations.

---

## Table Columns

- **Version number**: Includes the product version number (for example, 3.4) for all installed MathWorks products or for only those products required for a model or chart.

  or

- **Release number**: Includes the MathWorks release number (for example, R2009b) for all installed MathWorks products or for only those products required for a model or chart.
- **Release date**: Includes the release date of for all installed MathWorks products or for only those products required for a model.
- **Is required for model**: Indicates “Yes” for each MathWorks product required for a model or chart.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cml_ver`

## See Also

`Evaluate MATLAB Expression`, `Insert Variable`, `MATLAB Property Table`, `Variable Table`

# Nest Setup File

## Purpose

Allow one report setup file (`.rpt` file) to run inside another

## Description

This component runs another report setup file at the point where the `Nest Setup File` component is located in the current report setup file.

The components of the inserted report setup file are stored in the current report setup file at the same level as the `Nest Setup File` component. Thus, inserted components have the same parent component as the `Nest Setup File` component.

## Properties

- **Report filename**: Specifies the name of the report setup file to import and run. You can enter a path to the file or use the **browse** button **(...)** to find the file. You can enter an absolute path or a relative path, relative to the report into which you nest the report.
- **Nest all reports with specified file name**: Nests all reports with the same name as specified in the **Report filename** option.
- **Inline nested report in this report**: Inserts the nested report in the original report setup file where this component is located.
- **Recursion limit**: Allows you to nest a report setup file inside itself by setting a recursion limit in this field. The recursion limit sets a limit on the number of times the report setup file can run itself.
- **Insert link to external report**: Creates two separate reports, one using the original report setup file and one using the nested report setup file. The report that includes the nested report includes an absolute path link to the nested report.
- **Link to external report is relative**: If you select **Insert link to external report**, then you can use the **Link to external report is relative** option to ensure the link to the nested report is a relative link. This feature facilitates including the report on a Web site.
- **Increment file name to avoid overwriting**: Appends a number to the file name of report that includes the nested report, to preserve earlier versions of current report file.

The Nest Setup File dialog box displays the report description of the nested report, if the nested report has a report description.

### Example

In the following example, the report setup file `R2.rpt` is nested in `R1.rpt`:

```
[-] Report - R1.rpt                [-] Report - R2.rpt
 [ ] Chapter                              [ ] 1
      [-] B                               [ ] 2
         [ ] Nest Setfile - R2.rpt       [-] Chapter
         [ ] C                              [ ] 4
      [ ] D                                 [ ] 5
```

The generated report is identical to the one generated by the following report setup file:

```
[-] Report - R1.rpt
  [ ] Chapter
  [-] B
     [ ] 1
     [ ] 2
     [-] Section 1
       [ ] 4
       [ ] 5
     [ ] C
  [ ] D
```

Components that determine their behavior from their parents, such as `Chapter/Subsection`, are affected by components in the parent report setup file.

### Insert Anything into Report?

Yes, if the nested report setup file produces a report.

## Class

`rptgen.crg_nest_set`

## See Also

`Comment`, `Import File`, `Stop Report Generation`, `Time/Date Stamp`

# Paragraph

## Purpose

Insert paragraph text into report

## Description

This component inserts a paragraph into the report. The paragraph text is taken from a child text component, or from text that you enter in the **Paragraph Text** field.

## Title Options

- **No paragraph title**(default): Specifies no title for the paragraph.
- **Get title from first child** : Gets the title of the paragraph from its first child component, which should be a `Text` component.
- **Custom title**: Specifies a custom title for the paragraph.

## Paragraph Text

Enter paragraph text into this field. If the `Paragraph` component has child components, the paragraph content is taken from the child components; otherwise, the `Paragraph` component inserts text from this field. If the `Paragraph` component does not have any child components and you do not enter any text into this field, nothing appears in the report.

Use the `%<VariableName>` notation in this field if you want to insert the value of a variable from the MATLAB workspace. For more details about this notation, see "%<VariableName> Notation" on page 9-90 on the `Text` component reference page.

## Style

- **Bold**: Makes the text bold.
- **Italic**: Makes the text italic.
- **Underline**: Underlines the text.
- **Strikethrough**: Strikes through the text.
- **Retain spaces and carriage returns**: Formats text in the report in the same way as it is entered.
- **Show text as syntax-highlighted MATLAB code**: Displays the text as syntax-highlighted MATLAB code.
- **Color**: Specifies the color of the text.

- Select a color from a list of colors.
- Enter `%<expr>`.
- Enter an RGB (truecolor) value as #RRGGBB. For example, `#0000ff` is a shade of blue.

## Insert Anything into Report?

Yes, depending on child components.

## Class

`rptgen.cfr_paragraph`

## See Also

`Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Table`, `Text`, `Title Page`

# Stop Report Generation

## Purpose

Halt report generation

## Description

This component acts like **Stop** during report generation. You can use this component inside an `if/then` statement by using Logical and Flow Control components to halt the report-generation process when the specified condition is `true`. When report generation halts, an XML source file is created, but not converted.

## Confirmation Properties

- **Confirm before stopping generation**: Generates a confirmation dialog box before stopping report generation.
- **Confirmation question**: Specifies a confirmation question for the prompt. The default is `Stop generating the report?`
- **Halt button name**: Specifies a name for the button that stops report generation. The default is **“Halt Generation”**.
- **Continue button name**: Specifies a name for the button that continues report generation. The default is **“Continue Generation”**.

### Example

This example creates a simple report that takes a snapshot of the current figure. If there is no current figure, the report generation automatically halts:

```
[-] Report - figure-report.rpt
 [-] if (isempty(get(0,'CurrentFigure')))
  [ ] Stop Generation
 [-] Figure Loop - current
  [-] Chapter - <Title from SubComponent1>
   [ ] Figure Name
   [ ] Graphics Figure Snapshot
   [ ] Figure Prop Table - Figure Properties
```

## Insert Anything into Report?

No.

**Class** `rptgen.crg_halt_gen`

**See Also** `Comment`, `Import File`, `Nest Setup File`, `Time/Date Stamp`

## Purpose

Insert parent of table

## Description

This component is a parent of a component hierarchy that you specify to insert a table into a report. Adding this component creates a hierarchy that defines a 2x2 table that you modify to define your specific table.

## Properties

- **Table title**: Specifies a title for the table. Enter text or `%<expr>`. If you specify a table title, text such as `Table 1:` precedes the table title.
- **Number of columns**: Specifies the number of columns in the table. Enter a number or `%<expr>`. You must have at least one column in the table.
- **Table width options**: Determines the width of the table.
  - `Auto`: Automatically sets the table width based on the table contents.
  - `Specify`: Enter the table width as either a percentage of the page width (for example, 75%), or provide an absolute width for the table. You can specify a table width in inches (`in`), picas (`pi`), or points (`pt`).
- **Table spans page width**: Spreads the table across the width of the page. If you clear this property, the table uses the **Table width options** setting.
- **Border**: Specifies whether to draw border lines around the outside edges of the table. For example, to draw a border line only at the top of the table, select `Top`.
- **Between columns**: Draw a vertical line on the right side of each column (except for the last column) in the table.

  To override this setting for a specific column or table entry, use the **Column separator** property of the Table Column Specification or Table Entry components, respectively.
- **Between rows**: Draw a horizontal line at the bottom of each row (except for the last row) in the table.

To override this setting for a specific table column, row, or entry, use the **Row separator** property of the appropriate component: Table Column Specification, Table Row, or Table Entry.

- **Horizontal entry alignment**: Aligns the position of Table Entry component content relative to the left and right sides of a table column.
  - `Left`: Aligns content with the left side of the column
  - `Center`: Aligns content in the middle of the column
  - `Right`: Aligns content with the right side of the column
  - `Double justified`: Justifies the left and right sides of the entry content, to avoid ragged left and right alignment

  To override this setting:

  - For a specific table column, use the Table Column Specification **Entry horizontal alignment** property.
  - For a specific table entry, use the Table Entry **Horizontal alignment** property.
- **Rotate table 90 degrees**: For PDF and HTML output file formats, rotates the table 90 degrees counterclockwise to the direction of the text flow on the page.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cfr_ext_table`

## See Also

`Table Body`, `Table Column Specification`, `Table Entry`, `Table Footer`, `Table Header`, `Table Row`, `Array-Based Table`, `Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Text`, `Title Page`

# Table Body

## Purpose

Insert parent of table body

## Description

This component is a parent of the rows that define the body of a table.

This component must be a child of a Table component. Add Table Row components as children to define the content of the table body.

## Properties

- **Entry vertical alignment**: Positions table entry content relative to the top and bottom of the row in which the table entry appears.

  To override this setting for a table header or footer, or for a table row within one of those table elements, use the **Entry vertical alignment** property for the Table Header, Table Footer, or Table Row component.

  To override this setting for a specific table entry, use the Table Entry **Vertical alignment** property.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cfr_ext_table_body`

## See Also

`Table`, `Table Column Specification`, `Table Entry`, `Table Footer`, `Table Header`, `Table Row`, `Array-Based Table`, `Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Text`, `Title Page`

## Purpose

Specify table column properties

## Description

Specifies the format of a table column. Add a Table Column Specification component for only those columns that you do not want the default settings for the table.

## Properties

- **Column number**: Specifies a column number for the column to which this column specification applies. Enter a number or `%<expr>`. Avoid using the same column number for two column specifications in the same table.
- **Column name**: Specifies the name of this column. The name appears in the Outline pane of the Report Explorer. Enter text or a `%<expr>`.

  A Table Entry component can use this name to specify that it starts or ends on this column.
- **Column width**: Specifies the width of the column.

  To specify an absolute column width, specify a number or `%<expr>`. Use one of these units of measure: inches (`in`), picas (`pi`), or points (`pt`).

  You can use relative widths for columns. If you use relative widths for one column in a table, you must use relative widths for the other columns in the table. Specify `1*` for one column, as a baseline. For other columns, specify the width as a factor of the baseline column. The width of each column reflects its relative size. For example, suppose a two column table is 6 inches wide. The width of the first column is set to 1*, and the width of the second column is set to 2*. The width of the first column is 2 inches, and the width of the second column is 4 inches.
- **Entry horizontal alignment**: Justifies the position of table entries in the column, relative to the left and right sides of the column.

  Use the **Horizontal entry alignment** setting of the Table component, or explicitly set this property:

  - `Left`: Aligns content with the left side of the column.

  - `Center`: Aligns content in the middle of the column.
  - `Right`: Aligns content with the right side of the column.
  - `Double justified`: Justifies the left and right sides the entry content, to avoid ragged left and ragged right alignment.

  To override this setting for a specific table entry, use the Table Entry **Horizontal alignment property** for that table entry.

- **Column separator**: Use the **Between columns** setting of the Table component, or explicitly set the **Column separator** property.
  - `True`: Draws a vertical line at the right edge of the column (except for the last column).
  - `False`: Draws no vertical line at the right edge of the column.
- **Row separator**: Use the **Between rows** setting of the Table component, or explicitly set the **Row separator** property.
  - `True`: Draws a horizontal line at the bottom of each row in the column (except for the bottom row).
  - `False`: Does not draw a horizontal line at the bottom of each row in the column.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cfr_ext_table_colspec`

## See Also

`Table`, `Table Body`, `Table Entry`, `Table Footer`, `Table Header`, `Table Row`, `Array-Based Table`, `Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Text`, `Title Page`

## Purpose

Insert table entry

## Description

Specifies the format of a table entry.

This component must be a child of a descendant of a Table Row component. Add Paragraph, Image, List, and other components to define the content of the table entry.

## Properties

- **Horizontal alignment**: Use the **Entry horizontal alignment** setting of the Table Column Specification component for the column in which the table entry appears, or explicitly set the **Horizontal alignment** property.
  - `Left`: Aligns content with the left side of the column.
  - `Center`: Aligns content in the middle of the column.
  - `Right`: Aligns content with the right side of the column.
  - `Double justified`: Justifies the left and right sides the entry content, to avoid ragged left and right alignment.
- **Vertical alignment**: Positions the table entry content relative to the top and bottom of the row in which the table entry appears.

  Use this property to override the **Entry vertical alignment** setting of the Table Row component in which this table entry appears.
- **Column separator**: Use this property to override the **Column separator** setting of the Table Column Specification component for the column in which the table entry appears.
  - `True`: Draws a vertical line at the right edge of the column for this table entry.
  - `False`: Draws no vertical line at the right edge of the column for this table entry.
- **Row separator**: Use this property to override the **Row separator** setting of the Table Row component for the row in which the table entry appears.

  - `True`: Draws a horizontal line at the bottom of the row, below the table entry.
  - `False`: Does not draw a horizontal line at the bottom of the row, below the table entry.

- **Background color**: Specifies the background color of the table entry. You can:
  - Use `Auto` to apply the **Background Color** setting of the Table Row component in which the table entry appears.
  - Select a color from a list of colors.
  - Enter `%<expr>`.
  - Enter an RGB (truecolor) value as #RRGGBB. For example, `#0000ff` is a shade of blue.
- **Span start column name**: Specifies the name of the column (as defined by the Table Column Specification component) to use as the first (left side) of a set of spanned columns for displaying the table entry content.
- **Span end column name**: Specifies the name of the column (as defined by the Table Column Specification component) to use as the last (right side) of a set of spanned columns for displaying the table entry.
- **Rows spanned**: Specifies the number of rows to span for the table entry. The spanning starts with the table row in which you define the table entry and extends below that row for the number of rows that you specify.
- **Text orientation**: Rotates table entry text in 90 degree increments, relative to the page text flow.

  To use the text orientation of the table row in which this table entry appears, select `Auto`.

  To override the **Text orientation** setting for the Table Row component in which this table entry appears, select a rotation value.

- **Rotated text width**: Specifies the width of table entry text that you rotate (with the **Text orientation** property).

  Specify the text width in inches (`in`), picas (`pi`), or points (`pt`).

  To avoid truncating the rotated text, set the **Rotated text width** to a value that allows the display of the longest string in the table row.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cfr_ext_table_entry`

## See Also

`Table`, `Table Body`, `Table Column Specification`, `Table Footer`, `Table Header`, `Table Row`, `Array-Based Table`, `Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Text`, `Title Page`

# Table Footer

## Purpose

Insert parent of table footer

## Description

This component is a parent of the Table Row components that define a table footer.

## Properties

- **Entry vertical alignment**: Positions the table entry content relative to the top and bottom of the table footer rows in which the table entries appear.

  To override this setting for a specific row in the table footer, use the **Entry vertical alignment** property of the Table Row component for that row.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cfr_ext_table_foot`

## See Also

`Table`, `Table Body`, `Table Column Specification`, `Table Entry`, `Table Header`, `Table Row`, `Array-Based Table`, `Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Text`, `Title Page`

## Purpose

Insert parent of table header

## Description

This component is a parent of the Table Row components that define a table header.

## Properties

- **Entry vertical alignment**: Positions the table entry content relative to the top and bottom of the table header rows in which the table entries appear.

  To override this setting for a specific row in the table header, use the **Entry vertical alignment** property of the Table Row component for that row.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cfr_ext_table_head`

## See Also

`Table`, `Table Body`, `Table Column Specification`, `Table Entry`, `Table Footer`, `Table Row`, `Array-Based Table`, `Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Text`, `Title Page`

# Table Row

## Purpose

Insert parent of table row entries

## Description

This component is a parent of Table Entry components that define a table row.

## Properties

- **Entry vertical alignment**: Positions the table entry content relative to the top and bottom of the table row in which the table entries appear.

  Use this property to override the **Entry vertical alignment** setting of the Table Header, Table Footer, or Table Body component in which the table row appears.

- **Row separator**: Use this property to override the **Row separator** setting of the Table component.

  - `True`: Draws a horizontal line at the bottom of the row (except for the last row).
  - `False`: Does not draw a horizontal line at the bottom of the row.

- **Background color**: Specifies the background color of the table row. You can:

  - Use `Auto` for the background color that the report stylesheet specifies, which for stylesheets provided with MATLAB Report Generator is white by default.
  - Select a color from a list of colors.
  - Enter `%<expr>`.
  - Enter an RGB (truecolor) value as #RRGGBB. For example, `#0000ff` is a shade of blue.

- **Row height**: Specifies the height of the table row.

  To let the table contents automatically set the row height, use `Auto`.

  To specify an absolute height for this table row, select `Specify` and enter the height in inches (`in`), picas (`pi`), or points (`pt`).

- **Text orientation**: Rotates text in the table entries in this table row, relative to the page text flow.

  To override the text rotation for a specific table entry, use the Table Entry **Text orientation** property for that table entry.

- **Rotated text width**: Specifies the width of table entry text that you rotate with the **Text orientation** property.

  Specify the text width in inches (`in`), picas (`pi`), or points (`pt`).

  To avoid truncating the rotated text, set the **Rotated text width** to a value that allows the display of the longest string in the table row.

  To override the rotated text width for a specific table entry in the table row, use the Table Entry **Rotated text width** property.

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cfr_ext_table_row`

## See Also

`Table`, `Table Body`, `Table Column Specification`, `Table Entry`, `Table Footer`, `Table Header`, `Array-Based Table`, `Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Text`, `Title Page`

# Text

## Purpose

Format and insert text into report

## Description

This component formats and inserts text into the report. It must have the `Paragraph` component as its parent.

## Properties

**Text to include in report**: Specifies text to include in the report.

### %<VariableName> Notation

You can enter `%<VariableName>` in this field (and in any field where the text appears blue) to include the value of a variable from the base MATLAB workspace. You cannot enter more than one variable in `%<>`. If you enter an invalid variable name, the report includes the text `%<VariableName>` instead of the value of the variable.

### Example

1. Enter the following text:

```
I have a %<ObjName> and it has %<NumLeaves> leaves.
The word '%<ObjName>' has %<size(ObjName)> letters.
```

2. Set `ObjName` = `"plant"` and `NumLeaves` = 3.

3. Generate the report. It looks as follows:

```
I have a plant and it has 3 leaves.
The word 'plant' has 5 letters.
```

## Style

- **Bold**: Makes the text bold.
- **Italic**: Makes the text italic.
- **Underline**: Underlines the text.
- **Strikethrough**: Strikes through the text.
- **Subscript**: Formats text as a subscript, in a smaller font than the other text, set slightly below the other text.

- **Superscript**: Formats text as a superscript, in a smaller font than the other text, set slightly above the other text.
- **Retain spaces and carriage returns**: Formats the text in the report as you entered it.
- **Show text as syntax-highlighted MATLAB code**: Shows the text as syntax-highlighted MATLAB code.
- **Color**: Specifies the color of the text.
  - Select a color from a list of colors.
  - Enter `%<expr>`.
  - Enter an RGB (truecolor) value as #RRGGBB. For example, `#0000ff` is a shade of blue.

## Insert Anything into Report?

Yes. Text.

## Class

`rptgen.cfr_text`

## See Also

`Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Table`, `Title Page`

# Time/Date Stamp

## Purpose

Insert time and date of report generation into report

## Description

This component inserts the time and date of the report generation into your report as text. It must have the `Paragraph` or `Chapter/Subsection` component as its parent.

## Prefix

**Include text before stamp** : Includes text before the time/date stamp. Specify the text in the corresponding field.

## Time Stamp Properties

- **Include current time in stamp**: Inserts the current time into the time/date stamp.
- **Time display**: Specifies the appearance of the time display. Options include:
  - `12-hour`
  - `24-hour`
- **Time Separator**: Specifies a separation marker between hours, minutes, and seconds. Options include:
  - `Blank space ( )`: Formats time as `Hour Minute Second`
  - `Colon (:)`: Formats time as `Hour:Minute:Second`
  - `Period (.)`: Formats time as `Hour.Minute.Second`
  - `None ()` : Formats time as `HourMinuteSecond`
- **Include seconds in time stamp**: Displays seconds in the time/date stamp.

## Date Stamp Properties

- **Include current date in stamp**: Inserts the current date in the time/date stamp.
- **Date order**: Specifies the order in which the day, month, and year appear. Options include:
  - `Day Month Year`
  - `Month Day Year`

  - `Year Month Day`

- **Date separator**: Specifies a separation marker between day, month, and year. Options include:
  - `Blank space ( )`: Displays date as `Day Month Year`
  - `Colon (:)`: Displays date as `Day:Month:Year`
  - `Slash (/)`: Displays date as `Day/Month/Year`
  - `Period (.)`: Displays date as `Day.Month.Year`
  - `None ()`: Displays date as `DayMonthYear`
- **Month display**: Specifies how the month displays. Options include:
  - `Long (December)`
  - `Short (Dec)`
  - `Numeric (12)`
- **Year display**: Specifies how the month displays. Options include:
  - `Long (2007)`
  - `Short (07)`

## Preview

This pane displays the time/date stamp to appear in the report.

## Insert Anything into Report?

Yes. Text.

## Class

`rptgen.crg_tds`

## See Also

`Comment`, `Import File`, `Nest Setup File`, `Stop Report Generation`

# Title Page

## Purpose

Insert title page at beginning of report

## Description

This component inserts a title page at the beginning of the report. You can use it in a report setup file as a child of a `Chapter/Subsection` component or by itself.

To use the Title Page component, you need to have a Chapter component in your report.

For PDF and HTML reports, you can use the Stylesheet Editor to position title page elements (for example, title, copyright, and images) anywhere on the front or reverse side of the title page in any order. You can specify the size, color, weight, and slant of text elements. For details, see "Modify Title Page Properties" on page 7-18.

## Properties

The text fields on this property pane support the `%<VariableName>` notation. For more details, see "%<VariableName> Notation" on page 9-90 on the `Text` component reference page.

## Main Tab

### Title

- **Title**: Specifies the title of the report. The title is in a large font.
- **Subtitle**: Specifies the subtitle of the report. The subtitle is in a smaller font under the title.

### Options

- **Author**:
  - `Custom`(default): Specifies the author of the report.
  - `No author`: Does not specify an author name.
  - `Automatic author`: Automatically includes your user name as the author name.

  The author name appears under the subtitle, in a smaller font than the subtitle.

- **Include report creation date**: Includes the date that the report is created. Choose the date format in the corresponding list.
- **Include copyright holder and year**: Includes copyright holder and year information.
- **Display legal notice on title page**: Includes the legal notice, report creation date, and copyright information on the title page of PDF and Microsoft Word reports.

## Image Tab

### File

- **File name**: Specifies the file name of an image to appear under the subtitle, on the title page.
- **Copy to local report files directory**: Copies the image file into the folder in which the report file is located.

### Display Options

- **Scaling**: Controls size of the image, as displayed in a browser. Making an image larger using this option does not affect the storage size of the image, but the quality of the displayed image may decrease as you increase or decrease the size of the displayed image.

  Generally, to achieve the best and most predictable display results, use the default setting of `Use image size`.

  - `Use image size`: Causes the image to appear the same size in the report as on screen (default).
  - `Fixed size`: Specifies the number and type of units.
  - `Zoom`: Specifies the percentage, maximum size, and units of measure.
- **Size**: Specifies the size of the snapshot in the form `w h` (width, height). This field is active only if you choose `Fixed size` in the **Scaling** list
- **Alignment**: Only reports in `PDF` or `RTF` format support this property. Options include:

  - `Auto`
  - `Right`
  - `Left`
  - `Center`

### Abstract Tab

**Abstract Text**: Specifies an optional abstract for the report.

#### Style

- **Bold**: Makes the text bold.
- **Italic**: Makes the text italic.
- **Underline**: Underlines the text.
- **Strikethrough**: Strikes through the text.
- **Retain spaces and carriage returns**: Formats the text in the generated report as you entered it.
- **Show text as syntax-highlighted MATLAB code**: Shows the text as syntax-highlighted MATLAB code.
- **Color**: Specifies the color of the text.

### Legal Notice Tab

**Legal Notice Text**: Specifies an optional legal notice for the report.

#### Style

- **Bold**: Makes the text bold.
- **Italic**: Makes the text italic.
- **Underline**: Underlines the text.
- **Strikethrough**: Strikes through the text.
- **Retain spaces and carriage returns**: Formats the text in the generated report as you entered it.
- **Show text as syntax-highlighted MATLAB code**: Shows the text as syntax-highlighted MATLAB code.

- **Color**: Specifies the color of the text.

## Insert Anything into Report?

Yes. Title page.

## Class

`rptgen.cfr_titlepage`

## See Also

`Chapter/Subsection`, `Empty Component`, `Image`, `Link`, `List`, `Paragraph`, `Table`, `Text`

# Variable Table

## Purpose

Insert table that displays all the variables in the MATLAB workspace

## Description

This component inserts a table that displays all the variables in the MATLAB workspace.

**Tip** Find all the variables in the MATLAB workspace by typing `whos` at the command line.

## Source Workspace

**Read variables from**:

- `Base workspace`: Reads variables from the MATLAB workspace.
- `MAT-file`: Reads variables from a binary file with a `.mat` extension. Use the `%<VariableName>` notation. For more details about this notation, see "%<VariableName> Notation" on page 9-90 on the `Text` component reference page.

## Table Title

- **Table title**:
  - `Automatic (Variables from MATLAB workspace)`: Sets the table title to the name of a MATLAB variable.
  - `Custom`: Specifies a custom title.
- **Table Columns**:
  - `Variable dimensions (MxN)`: Includes the size of the variable.
  - `Variable memory bytes`: Includes the number of bytes of memory consumed by the variable.
  - `Variable class`: Includes the variable class.
  - `Variable value`: Includes the value of the variable.

**Note** Large variable arrays collapse to `[MxN CLASS]`. For example, if you have a 300-by-200 double array, it appears in the report as `[300x200 DOUBLE]`.

### Example

The following is an example of a variable table that includes size, memory bytes, and value information in the table columns.

| Name | Size | Bytes | Value |
|---|---|---|---|
| `aCell` | 1x2 | 238 | { [ 1 2 3 4 ] Speed (kph) } |
| `aNumber` | 1x1 | 8 | 1 |
| `aString` | 1x11 | 22 | Speed (kph) |
| `aStructure` | 1x1 | 302 | [struct w/ fields. Inputs, Outputs] |
| `aVector` | 1x4 | 32 | [ 1 2 3 4 ] |

## Insert Anything into Report?

Yes. Table.

## Class

`rptgen.cml_whos`

## See Also

`Evaluate MATLAB Expression`, `Insert Variable`, `MATLAB Property Table`, `MATLAB/Toolbox Version Number`

# While Loop

## Purpose

Iteratively execute child components while a specified condition is true

## Description

This component iteratively executes its child components while a specified condition is true. The `While Loop` component must have at least one child component; the purpose of this component is to run its children several times. If it does not have any children, this component does not add anything to the report.

**Tip** Limit the number of repetitions to prevent infinite loops.

## Logic Properties

- **Continue looping if this expression is true**: Specifies a string to evaluate. This string must be a valid MATLAB expression that evaluates to `1` or `0` (`true` or `false`).

  For example, if `a = 1`, `b = 2`, and `c = 3`, the following command:

  ```
  d=(a>b/c)
  ```

  returns:

  ```
  d = 1
  ```

  Because `1` is greater than `b/c` (`2/3`), this expression is `true` and evaluates to `1`.

- **Limit number of loops to**: Allows you to prevent infinite loops. Use the left and right arrows to increase or decrease the number of loops.
- **Initialize with this expression**: Initializes the loop with a valid MATLAB expression.

## Insert Anything into Report?

Yes, if it has a child component.

**Class** `rptgen_lo.clo_while`

**See Also** `For Loop`, `Logical Else`, `Logical Elseif`, `Logical If`, `Logical Then`

10

# Functions – Alphabetical List

# compwiz

## Purpose

Create custom MATLAB Report Generator components

## Syntax

```
compwiz
compwiz (' browse')
compwiz (' v1browse')
compwiz (rptgen.cfr_list)
```

## Description

The Create Component dialog box creates a framework for custom report components. For more information, see "Create Custom Components" on page 6-3.

- `compwiz` with no arguments displays the Component Editor in the Report Explorer.
- `compwiz (' browse')` displays a list of components from which to derive a new component.
- `compwiz (' v1browse')` displays a list of legacy (`v1.x`) components from which to derive a new component.
- `compwiz (rptgen.cfr_list)` initializes the Component Editor with the settings of the referenced components.

## See Also

`setedit` | `report` | `rptconvert` | `rptlist`

## How To

- "Create Custom Components" on page 6-3

## Purpose

Generate reports from report setup file

## Syntax

```
report
report (filename,...)
report (___,-oOPATH)
report (___,-fFORMAT)
report (___,-genOption1,...)
[report1, report2, ...] = report (rptfile1, rptfile2, ...)
```

## Description

- `report` with no arguments opens the Report Explorer. For more information on the Report Explorer, see "Report Explorer" on page 1-8
- `report (filename,...)` generates a report from the specified report setup files. You can specify one or more report setup files. When specifying the name of the report setup file, omit the `.rpt` file name extension.
- `report (___,-oOPATH)` sets the name of the generated report. You can specify a path or a single file name for the `OPATH` path argument.
- `report (___,-fFORMAT)` sets the output format and file name extension of the generated report. Supported formats include:
  - Adobe Acrobat PDF (`.pdf`)
  - HTML (`.html`)
  - Microsoft Word (`.doc`)
  - Rich Text format (`.rtf`)

  For example, `report('simple-report','-fPDF')` generates a PDF file.
- `report (___,-genOption1,...)` specifies one or more of the following report generation options:
  - `-noview` — Prevents launching the file viewer
  - `-graphical` — Shows hierarchy in Report Explorer
  - `-debug` — Enables debug mode

  - `-quiet` — Sets error echo level to `0`
  - `-sSTYLESHEETNAME` — Sets stylesheet name (not required when choosing format)
- `[report1, report2, ...]  = report (rptfile1, rptfile2, ...)` returns the names of the generated reports. If the MATLAB Report Generator software cannot generate a given report, its returned name is empty.

**Note** For reports that use the `Word Document` format, you must have Microsoft Word software installed on the machine that you use to generate the report.

## Examples

### Example 1: Setting the format of the generated report

- Generate the report `testrpt` in `PDF` format:

```
report testrpt -fpdf
```

- Generate the report `testrpt` in `RTF` format:

```
report testrpt -frtf
```

- Generate the report `testrpt` in Microsoft Word format:

```
report testrpt -fdoc
```

**Note** Only Microsoft Windows platforms support this option.

- Generate a multipage `HTML` report from the `figloop-tutorial` report setup file:

```
report figloop-tutorial -fhtml -shtml-!MultiPage
```

### Example 2: Specifying the file and path of the generated report

Generate a report named `simple-report` in the folder `/tmp/index.html`:

```
report ('simple-report','-o/tmp/index.html')
```

## See Also

`setedit` | `rptconvert` | `rptlist` | `compwiz`

## How To

- "Report Generation"

# rptconvert

## Purpose

Convert DocBook XML files into supported document formats

## Syntax

```
rptconvert
rptname = rptconvert (source)
rptname = rptconvert (source, format)
rptname = rptconvert (source, format, stylesheet)
...=rptconvert(...,'-view')
...=rptconvert(...,'-quiet')
...=rptconvert(...,'-verbose')
sheetlist = rptconvert('-stylesheetlist')
sheetlist = rptconvert('-stylesheetlist',format)
FORMATLIST = rptconvert('-formatlist')
```

## Description

This function converts a DocBook XML source file created by the report-generation process to a supported document format. For information about supported output formats, see “Supported Report Formats” on page 1-11.

`rptconvert` with no input arguments launches the converter. When input arguments are passed to this function, `rptconvert` converts the XML document to the specified format and displays status messages to the MATLAB Command Window.

In the following commands:

- `rptname = rptconvert (source)`
- `rptname = rptconvert (source, format)`
- `rptname = rptconvert (source, format, stylesheet)`

`source` is the name of the DocBook XML file created by the report-generation process. You can specify this file name with or without its file extension.

`format` is a unique identifier code for each output format type. If you omit this argument, the XML file is converted to HTML format by default.

`stylesheet` is a unique identifier for a given stylesheet. If you omit this argument, the default stylesheet for the selected `format` is used.

You can also pass the following flags to the input arguments:

- `...=rptconvert(...,'-view')` displays the converted document.
- `...=rptconvert(...,'-quiet')` suppresses status messages.
- `...=rptconvert(...,'-verbose')` shows detailed status messages.
- `sheetlist = rptconvert('-stylesheetlist')` returns a two-column cell array. The first column of this array includes valid stylesheet identifiers. The second column includes descriptions of each stylesheet.
- `sheetlist = rptconvert('-stylesheetlist',format)` returns an array like that returned by `sheetlist = rptconvert('-stylesheetlist')`. The first column of this array includes stylesheet identifiers for the specified `format`.
- `FORMATLIST = rptconvert('-formatlist')` returns a two-column cell array. The first column of this array includes valid `format` values, the second column includes descriptions of each format.

## Examples

Retrieve a list of available `HTML` stylesheets:

```
rptconvert('-stylesheetlist','html')
```

## See Also

`setedit` | `report` | `rptlist` | `compwiz`

## How To

- “Convert XML Documents” on page 4-19

# rptlist

## Purpose

Retrieve list of all report setup files in MATLAB path

## Syntax

```
rptlist
rptlist ('system_name')
list = rptlist
```

## Description

`rptlist` with no arguments opens the Report Explorer, which lists available report setup files in the MATLAB path.

`rptlist ('system_name')` opens the Report Explorer with the Simulink system’s `ReportName` property selected.

`list = rptlist` returns a list of report setup files in the MATLAB path.

## See Also

`setedit`

## Purpose

Start Report Explorer

## Syntax

```
setedit (filename)
```

## Description

`setedit (filename)` opens the Report Explorer and loads the report setup file named `filename`. If a file with the specified name does not exist, Report Explorer opens an empty report setup file with that name.

## See Also

`rptlist` | `report` | `rptconvert`

## How To

- "Report Explorer" on page 1-8

# Index