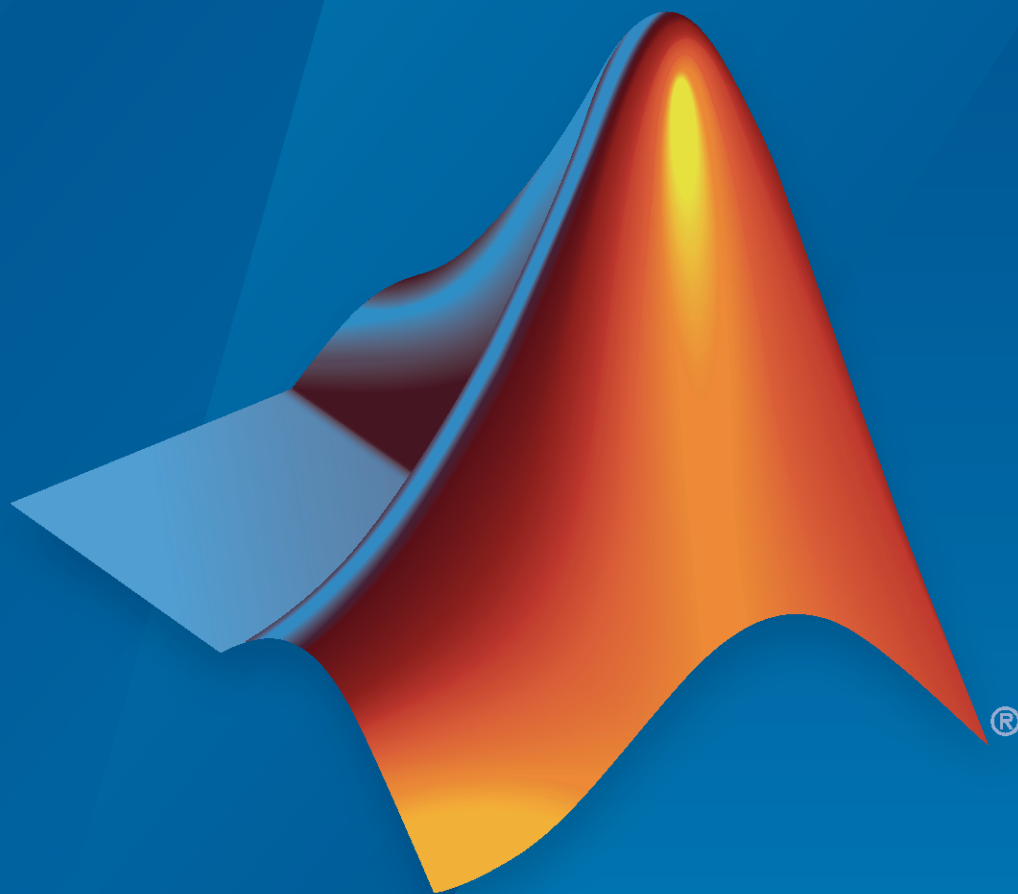


MATLAB® Report Generator™ Release Notes



MATLAB®



How to Contact MathWorks



Latest news: www.mathworks.com
Sales and services: www.mathworks.com/sales_and_services
User community: www.mathworks.com/matlabcentral
Technical support: www.mathworks.com/support/contact_us



Phone: 508-647-7000



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

MATLAB® Report Generator™ Release Notes

© COPYRIGHT 2002–2021 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

R2021a

Control format of numbers in reports	1-2
mlreportgen.report.MATLABCode: Report syntax-highlighted MATLAB code	1-2
rotate method: Change orientation of DOM API report pages	1-2
Specify page dimensions programmatically in Report API reports	1-3
Better equation rendering	1-4
HTML to DOM Conversion: Support for more HTML elements	1-4
mlreportgen.report.MATLABVariable: Improved reporting of enumeration classes	1-5
Scale Word document contents to paper size	1-5
mlreportgen.report.TableOfContents: Use new properties for number of levels and leader pattern	1-6
mlreportgen.ppt.InternalLink class: Link from one slide to another	1-6
mlreportgen.ppt.Presentation.createTemplate: Create a copy of the PPT API default presentation template	1-6

R2020b

Lists of figures, tables, and captions in reports	2-2
Changes to default border widths in PDF reports	2-2
Changes to hyphenation and line break behavior for PDF reports	2-3
Changes to an embedded file in a PDF report	2-3
append: Use append to add content to Report API objects	2-4

New mlreportgen.dom.EmbeddedObject creation syntax for specifying the style of link text	2-4
mlreportgen.report.MATLABVariable reporter options for specifying reported name and value	2-4
New PowerPoint table creation syntax for specifying the table style	2-4
Features being removed or changed	2-5
add method is not recommended	2-5

R2020a

PowerPoint API table formatting	3-2
mlreportgen.ppt.AutoFit: Scale text to fit placeholder or text box in slide	3-2
mlreportgen.dom.EmbeddedObject class: Embed files in a report	3-3
mlreportgen.report.HTMLModuleTabs class: Create tabbed panels in HTML reports	3-3
New functions for preparing HTML for conversion to DOM objects	3-3
mlreportgen.dom.Preformatted class: Preserve white-space formatting of text	3-3
mlreportgen.utils.normalizeLinkID: Generate a valid link target ID for all report types	3-3
HTML text no longer wrapped in a paragraph	3-4

R2019b

Inline display of equations	4-2
More options for vertically aligning inline elements	4-2
SVG images in Microsoft Word reports	4-2
Partial support for em units when converting HTML to DOM objects ...	4-3
Indentation of ordered lists for Microsoft Word documents	4-3

PropertyFilterFcn and NumericFormat properties in MATLAB variable reporter	4-5
Manipulation of tables and pictures that come from a Microsoft PowerPoint presentation template slide	4-5

R2019a

Report Explorer-based reporter	5-2
Create a custom reporter	5-2
Report API examples	5-2

R2018b

MATLAB Variable Reporter: Include information on MATLAB variables in a report	6-2
Table Slicer: Include tables sliced by column to fit on report pages	6-2
Enhanced table of contents customization	6-2
Collapsible table of contents for HTML reports	6-2
Utility functions	6-3

R2018a

Custom Finders: Search data containers and return results in reportable form	7-2
---	-----

R2017b

MATLAB Reporters: Use MATLAB objects to generate title pages, tables of contents, chapters, figures, and other report elements	8-2
PDF Image Format: Generate PDF Reports containing PDF images	8-2

Rebuild Template and Style Sheet Cache	8-2
---	------------

R2017a

Single-File HTML Output: Generate a report as a single HTML file	9-2
Embedded Style Sheet: Convert HTML document containing style sheets to Microsoft Word and PDF	9-2
Page Break Component: Insert page breaks in template-based Microsoft Word and PDF reports	9-2
White-Space Options: Use spaces and line feeds to format code and other text in reports	9-2
PDF Table Automatic Layout	9-2

R2016b

Report Forms: Use the Report Explorer and Microsoft Word, PDF, or HTML templates to create form-based reports	10-2
PDF Image Maps: Add hyperlinks to images in PDF reports	10-2
PDF Watermarks: Add watermarks to PDF reports	10-2
Collapsible Table of Contents: Hide table of contents in HTML reports	10-2
Simplified Table Creation: Create tables using MATLAB tables and categorical arrays	10-2
Report Components: Add a line break	10-3
Conversion Templates: Create nested numbered sections	10-3
PDF Fonts: System fonts detected for PDF output	10-3
Noto Fonts: New fonts supported for non-English PDF reports	10-3
Subdocuments in DOM Reports: Use docview function to unlink subdocuments	10-4
Diff and Merge: Reuse open MATLAB sessions for diff and merge from external source control tools	10-4

R2016a

Scalable Report Generation: Generate PDF reports as big as 10,000 pages	11-2
Table of Contents: Add TOCs programmatically	11-2
Page Numbers: Create page number placeholders programmatically ..	11-2
HTML Text Component: Convert HTML to Word or PDF	11-3
pptview Function: Open PowerPoint presentation or convert it to PDF	11-3
Cross-Platform PDF Viewer: View PDF with built-in viewer	11-3
Functionality Being Removed or Changed	11-3

R2015b

Programmatic interface for adding content to PowerPoint presentations	12-2
DOM API object display options	12-2
DOM API horizontal rule	12-2

R2015a

Support for appending HTML string or file to a Word or PDF report generated by the Document Object Model (DOM) API	13-2
Single-file output option for HTML reports generated by DOM API	13-2
Simplified table formatting with DOM API	13-2
Container for generating high-level HTML elements	13-3
Images and text for DOM report links	13-3

R2014b

Report formatting based on Word and HTML templates	14-2
MATLAB report objects for creating report scripts	14-2
Fast file converter with reduced memory requirements	14-2
Zippered package option for HTML reports	14-2
Fill-in-the-blanks Word and HTML forms for generating reports	14-2
Color settings preferences in MATLAB Comparison Tool	14-3

R2014a

OpenOffice support	15-2
---------------------------------	-------------

R2013b

Bug Fixes

R2013a

Bug Fixes

R2012b

Non-English character set report formats	18-2
---	-------------

R2012a

Enhanced Table Components	19-2
Title Page Formatting Enhancements	19-2
Text Component Supports Subscripts and Superscripts	19-2

R2011b

Full Page Image Option for PDF Reports	20-2
Include Legal Notice, Report Creation Date, and Copyright on the Title Page	20-2

R2011a

Improved PDF Pagination	21-2
Export XML Comparison Results to the Workspace	21-2

R2010b

Complete Text Included in Reports	22-2
Linking to MATLAB Commands	22-2
Insert Variable Component Enhancements	22-2
Nest Setup File Component Supports Relative Links	22-2
Enhanced Error Handling Code for Evaluate MATLAB Expression Component	22-2
Transposable Columns for Handle Graphics Summary Table Component	22-2
Additional Date Format for Title Page Component	22-2
Improved XML Comparison Report	22-2

R2010a

Bug Fixes

R2009b

Style Sheets and Style Sheet Data Items Now Alphabetized	24-2
Improved Images in Word and RTF Reports on Windows Platforms	24-2
Required Products Information for MATLAB/Toolbox Version Number Component	24-2
Adobe Illustrator Image File Format No Longer Supported	24-2
Navigation Controls for XML Comparison Report	24-2

R2009a

No New Features or Changes

R2008b+

Comparison of XML Files	26-2
--	-------------

R2008b

Bug Fixes

R2008a

Bug Fixes

R2007b

Text Formatting Options for Title Page, Text, and Paragraph Components	29-2
--	-------------

R2007a

Bug Fixes

R2006b

Bug Fixes

R2006a+

Bug Fixes

R2006a

User Interface and Performance Enhanced	33-2
XML File Format Changed	33-2

R14SP3

Style Sheets Modify PDF Headers and Footers	34-2
--	-------------

Stylesheet Editor	35-2
Table Cell Spanning	35-2
Generating Microsoft Word Documents as .doc Files	35-2
Improved Graphical User Interface	35-2

R2021a

Version: 5.10

New Features

Bug Fixes

Compatibility Considerations

Control format of numbers in reports

In previous releases, to include a number in a report, you represented the number as an `mlreportgen.dom.Text` object. To control the format of the number, you converted the number to text in MATLAB before you created the `Text` object.

In R2021a, you can use objects of the `mlreportgen.dom.NumberFormat` class to specify the format of numbers in a report. To specify the format of one number, represent the number as an object of the new `mlreportgen.dom.Number` class, instead of as an object of the `mlreportgen.dom.Text` class. Assign the `NumberFormat` object to the `Style` property of the `Number` object. For example:

```
import mlreportgen.dom.*

rpt = Document("Report with NumberFormat", "pdf");

n = Number(pi);
n.Style = [n.Style {NumberFormat("%0.4f")}];
append(rpt,n);

close(rpt);
rptview(rpt);
```

To specify the format of all numbers in a report element, such as a paragraph, list, or table, assign a `NumberFormat` object to the `Style` property of the element object.

You can specify the default number format that the DOM API uses to generate numbers by using the new function `mlreportgen.dom.setDefaultNumberFormat`. The default format applies for the duration of the MATLAB session. You can override the default format by using a `NumberFormat` object. To get the default format, use the new function `mlreportgen.dom.getDefaultNumberFormat`.

See “Format Numbers”.

mlreportgen.report.MATLABCode: Report syntax-highlighted MATLAB code

Use an object of the `mlreportgen.report.MATLABCode` class to include syntax-highlighted MATLAB code in a report. Optionally, you can apply smart indenting to the reported code and report the McCabe cyclomatic complexity of the functions in the code.

rotate method: Change orientation of DOM API report pages

Use the `rotate` method to change the orientation of pages in a Microsoft® Word or PDF report that is based on the DOM API. Use the `rotate` method with an `mlreportgen.dom.DOCXPageLayout` or `mlreportgen.dom.PDFPageLayout` layout object. You can access the default layout object by using the `CurrentPageLayout` property of an `mlreportgen.dom.Document` or `mlreportgen.dom.DocumentPart` object and use `rotate` with that object. For example:

```
import mlreportgen.dom.*;
d = Document("myreport", "docx");
open(d);

plObj = d.CurrentPageLayout;
rotate(plObj);
```

```
append(d, "This page has landscape orientation.");
close(d);
rptview(d);
```

You can also use `rotate` with a layout object that you create yourself and append to the document or document part. For example:

```
import mlreportgen.dom.*;
d = Document("myreport", "docx");
open(d);

plObj = DOCXPageLayout();
rotate(plObj);
append(d, plObj);

append(d, "This page has landscape orientation.");
close(d);
rptview(d);
```

The `rotate` method switches the `Height` and `Width` property values of the `mlreportgen.dom.Pagesize` object that is used by the `DOCXPageLayout` or `PDFPageLayout` object. The method also changes the value of the `Orientation` property from `portrait` to `landscape` or from `landscape` to `portrait`.

In previous releases, to change the page orientation, you had to switch the values of the `Height` and `Width` properties of the `PageSize` object yourself and update the `Orientation` property to be consistent with the page size.

Specify page dimensions programmatically in Report API reports

You can now programmatically customize the page sizes, margins, headers, footers, and gutters in a Report API report or report section. In previous releases, to customize the page dimensions, you had to modify the templates used by the report or report section. Use the template-based approach if you plan to make other modifications to the templates. Otherwise, use the programmatic approach.

Using the programmatic approach:

- To specify the page size, create an `mlreportgen.dom.PageSize` object and assign it to the `PageSize` property of the layout object used by the report or report section.
- To specify the page margins, header, footer, and gutter, create an `mlreportgen.dom.PageMargins` object and assign the object to the `PageMargins` property of the layout object used by the report or report section.

`PageSize` and `PageMargins` are new properties of the `mlreportgen.report.ReportLayout` and `mlreportgen.report.ReporterLayout` classes, which represent the layout in a report or report section, respectively. To access a layout object, use the `Layout` property of an `mlreportgen.report.Report` object or an object of one of these reporter classes:

- `mlreportgen.report.TitlePage`
- `mlreportgen.report.TableOfContents`
- `mlreportgen.report.ListOfFigures`
- `mlreportgen.report.ListOfTables`

- `mlreportgen.report.ListOfCaptions`
- `mlreportgen.report.Chapter`

For an example, see “Customize the Page Size and Margins of a Report Programmatically”.

Better equation rendering

In R2021a, an `mlreportgen.report.Equation` reporter can render an equation directly by using the rendering used by the Live Editor and Simulink® Editor. In previous releases, an `Equation` reporter used a figure window to render equations. Direct rendering has these advantages:

- The support for equation markup is better than the support provided by a figure window.
- Equations in a report have the same appearance as equations in the Live Editor and Simulink annotations.

In R2021a, direct rendering is disabled by default. To enable direct rendering, set the new `UseDirectRenderer` property of the `Equation` reporter to `true`.

Compatibility Considerations

If you set the `UseDirectRenderer` property to `true`:

- The image format must be PNG.
- Setting the background color has no effect.
- You must specify the color of an equation as a string scalar or character vector that contains a valid CSS name, RGB triplet, or hexadecimal color code. You cannot specify a color name using a short name, such as "r". For example, specify red as "red", "rgb(255,0,0)", or "#FF0000". For more information, see the `Color` property of the `mlreportgen.report.Equation` class.
- The `getSnapshotImage` method returns a data URL that contains the equation image instead of the path of the file that contains the image.

If your application requires formats other than PNG or a different background color, set the `UseDirectRenderer` property to `false`.

HTML to DOM Conversion: Support for more HTML elements

The classes that you use to convert HTML to DOM objects, `mlreportgen.dom.HTML` and `mlreportgen.dom.HTMLFile`, support these additional HTML elements:

- `address`
- `big`, `small`
- `blockquote`
- `center`
- `cite`, `dfn`, `em`, `var`
- `dd`, `dl`, `dt`
- `kbd`
- `mark`

- nobr
- samp
- strong
- th

For the complete list of supported HTML elements, see `mlreportgen.dom.HTML` and `mlreportgen.dom.HTMLFile`.

mlreportgen.report.MATLABVariable: Improved reporting of enumeration classes

An `mlreportgen.report.MATLABVariable` object now reports the class member name for a variable that has an enumeration type. For example:

x. Monday

If the enumeration class is a subclass of a MATLAB built-in class, the underlying value of the member is also reported. For example:

y. ON (true)

If the enumeration class defines properties, the member name is reported in the property table. For example:

Table 1. z

Enumerated Value: Error	
Property	Value
R	1
G	0
B	0

Scale Word document contents to paper size

You can scale the contents of a Microsoft Word document to fit A4 or US Letter paper sizes using one of these approaches:

- When you print a document with `docview`, use the `"printdocscaled"` option. For example:


```
docview("myDoc.docx", "printdocscaled");
```
- When you print the document using the `print` method of the `mlreportgen.utils.WordDoc` class, use the `ScaleToFitPaper` option. For example:


```
docobj = mlreportgen.utils.WordDoc("MyDoc.docx");
print(docobj, "ScaleToFitPaper", true);
```

mlreportgen.report.TableOfContents: Use new properties for number of levels and leader pattern

To specify the number of header levels and the leader type for a Report API table of contents, use the `NumberOfLevels` and `LeaderPattern` properties of the `mlreportgen.report.TableOfContents` object, instead of the `TOCObj` property.

Compatibility Considerations

Report generation programs that use the `TOCObj` property will continue to run.

mlreportgen.ppt.InternalLink class: Link from one slide to another

In R2021a, the API for PowerPoint® (PPT API) provides a new class, `mlreportgen.ppt.InternalLink`, that you can use to link from one slide in a presentation to another slide in the presentation. You can identify the target slide by its index or name. See “Create and Format Links”.

mlreportgen.ppt.Presentation.createTemplate: Create a copy of the PPT API default presentation template

Use the new `mlreportgen.ppt.Presentation.createTemplate` method to create a copy of the default PowerPoint API presentation template. For example:

```
import mlreportgen.ppt.*  
  
templateCopy = mlreportgen.ppt.Presentation.createTemplate("myTemplate.pptx");
```

You can customize the copy and use it as the template for another presentation. For example:

```
ppt = Presentation("myPresentation.pptx", templateCopy);
```

In previous releases, to create a copy of the default template, you had to create an empty presentation without specifying a template and save the presentation.

R2020b

Version: 5.9

New Features

Bug Fixes

Compatibility Considerations

Lists of figures, tables, and captions in reports

To create a list of figures section in a Report API report, use an `mlreportgen.report.ListOfFigures` object. To create a list of tables section, use an `mlreportgen.report.ListOfTables` object. For a section that contains a list of other types of report elements, such as equations, use an `mlreportgen.report.ListOfCaptions` object. See [Create a List of Figures or Tables in a Report API Report](#) and [Create a List of the Captions or Titles of Related Report Elements in a Report API Report](#).

To add a list of figures, tables, or captions to a DOM API report, use an `mlreportgen.dom.LOF`, `mlreportgen.dom.LOT`, or `mlreportgen.dom.LOC` object.

Changes to default border widths in PDF reports

In R2020b, the default border width for PDF reports is 3px. In previous releases, the default border width was 1px. To override the default border width, the default PDF templates now include markup to specify 1px borders for tables. The templates do not specify the border widths for report elements other than tables.

As a result of the new default border width, the border width in a PDF report generated in R2020b is 3px instead of 1px in these cases:

- You use an existing custom PDF template, that does not specify table border widths, with a report generation program that generates tables with borders.
- You create a border for a report element other than tables.

Compatibility Considerations

In R2020b, if you generate table borders using the default PDF templates, the borders are 1px. You do not have to make any changes.

In R2020b, if you use a custom PDF template that you created in a previous release and the template does not specify the border widths, the table borders are 3px. To update the custom templates to specify 1px borders:

- 1 Unzip the template. For example:

```
unzipTemplate('my_custom_template.pdf');
```

- 2 Open the `root.css` file, which is in the `stylesheets` folder of the unzipped template. For example, if the template is `my_custom_template.pdf`, open `my_custom_template/stylesheets/root.css`.

- 3 Add this markup to the `root.css` file:

```
th,  
td {  
    -fo-border-top-width: 1px;  
    -fo-border-bottom-width: 1px;  
    -fo-border-start-width: 1px;  
    -fo-border-end-width: 1px;  
}
```

- 4 Package the template. For example:

```
zipTemplate('my_custom_template.pdf');
```

See `Modify Styles` in `PDF Templates`.

In R2020b, if you generate borders for elements other than tables, the borders are 3px by default. To generate the borders with a different width, use an `mlreportgen.dom.Border` style object. When you specify a border style or color, it is a good practice to also specify the width.

Changes to hyphenation and line break behavior for PDF reports

R2020b includes these improvements to line break and hyphenation behavior in PDF reports:


- Where possible, line breaks occur between words. When necessary to prevent overflow, line breaks may occur anywhere in a word.
- When hyphenation is on, hyphenation characters are used only when a line break occurs at the end of a syllable.
- When hyphenation is off, table cell contents do not overflow. In previous releases, to prevent overflow, hyphenation was on by default for tables. Now that table cells do not overflow when hyphenation is off, hyphenation is off by default for tables.

Compatibility Considerations

A PDF report generated in R2020b might look different than the same report generated in previous releases due to the changes in hyphenation and word break behavior.

Changes to an embedded file in a PDF report

In R2020b, if you use `mlreportgen.dom.EmbeddedObject` to embed a file in a PDF report, the link to the embedded file is inserted into the PDF as an annotation that has a paper clip icon. For example:

About XYZ, Inc.

[CompanyInfo.pdf](#)

To open the embedded file, double-click the paper clip icon. If you do not see the icon, try opening the PDF with a different PDF viewer, such as Adobe® Acrobat®.

In previous releases, a hyperlink to the embedded file was inserted into the PDF. For example:

About XYZ, Inc.
[CompanyInfo.pdf](#)

Compatibility Considerations

If a report generation program embeds a file in a PDF report, the appearance of the link to the embedded file differs between R2020b and previous releases. In R2020b, you see a paper clip icon

that you click to open the embedded file. In previous releases, you see a hyperlink. No changes are required to your report generation program.

append: Use append to add content to Report API objects

To add content to `mlreportgen.report.Report`, `mlreportgen.report.Chapter`, and `mlreportgen.report.Section` objects, you can now use the `append` method instead of the `add` method. For compatibility considerations, see “add method is not recommended” on page 2-5.

New `mlreportgen.dom.EmbeddedObject` creation syntax for specifying the style of link text

The `mlreportgen.dom.EmbeddedObject` class was introduced in R2020a. In R2020b, you can create an object of this class by using a new syntax that allows you to specify the template-defined style name to use for the link text.

```
embObj = mlreportgen.dom.EmbeddedObject(path,linkText,styleName)
```

The new creation syntax creates an `mlreportgen.dom.Text` object to hold the link text and sets the `StyleName` property of the `Text` object to the value of the `styleName` argument.

`mlreportgen.report.MATLABVariable` reporter options for specifying reported name and value

Specify the name or value reported by an `mlreportgen.report.MATLABVariable` reporter using these new options:

- `Title` property — If you set this property, the reporter uses it as the title of the variable in a report. Otherwise, it uses the value of the `Variable` property.
- `setVariableValue` method — Use this method to specify the value to report for a variable. You can use this method to report values without creating a variable for the value. For example, you can report an individual entry in a `containers.Map` object.

```
myMap = containers.Map(["key1", "key2"], [100, 200]);  
rptr = mlreportgen.report.MATLABVariable("key1");  
setVariableValue(rptr, myMap("key1"));
```

The `setVariableValue` method sets the `Location` property to "User-Defined", which is a new value for this property in R2020b.

New PowerPoint table creation syntax for specifying the table style

You can create an object of the `mlreportgen.ppt.Table` class by using a new syntax that allows you to specify the template-defined style name to use for the table.

```
tableObj = mlreportgen.ppt.Table(tableValues, styleName)
```

The new creation syntax creates the `Table` object with the specified content and sets the `StyleName` property to the value of the `styleName` argument.

To view a list of valid style names, use the `getTableStyleNames` method of the `mreportgen.ppt.Presentation` object to which the table is added.

Features being removed or changed

add method is not recommended

Still runs

Starting in R2020b, use the `append` method instead of the `add` method to add content to objects of these Report API classes:

- `mreportgen.report.Report`
- `mreportgen.report.Chapter`
- `mreportgen.report.Section`

To add content to a DOM API object, such as an `mreportgen.dom.Paragraph` object, continue to use the `append` method of the DOM object. The advantage of using `append` to add content to Report API objects is that you use the same method name as you use to add content to DOM API objects.

There are no plans to remove the `add` methods of the `Report`, `Chapter`, or `Section` classes. Report API programs that use the `add` methods will continue to run.

To update existing code, replace the method name `add` with `append` as shown by the examples in the table.

Not Recommended	Recommended
<pre>import mreportgen.report.* import mreportgen.dom.* rpt = Report("myrpt","pdf"); ch = Chapter("My Chapter"); sect = Section("My Section"); para = Paragraph("My Content "); add(para,"more Content"); add(sect,para); add(ch,sect); add(rpt,ch); close(rpt); rptview(rpt);</pre>	<pre>import mreportgen.report.* import mreportgen.dom.* rpt = Report("myrpt","pdf"); ch = Chapter("My Chapter"); sect = Section("My Section"); para = Paragraph("My Content "); append(para,"more Content"); append(sect,para); append(ch,sect); append(rpt,ch); close(rpt); rptview(rpt);</pre>

R2020a

Version: 5.8

New Features

Bug Fixes

Compatibility Considerations

PowerPoint API table formatting

In R2020a, the API for PowerPoint (PPT API) provides new classes and properties for formatting tables.

The new format classes are:

- `mlreportgen.ppt.Border`
- `mlreportgen.ppt.ColSep`
- `mlreportgen.ppt.FlowDirection`
- `mlreportgen.ppt.RowHeight`
- `mlreportgen.ppt.RowSep`
- `mlreportgen.ppt.TextOrientation`

The new format properties of the `mlreportgen.ppt.Table` class are:

- `BackgroundColor`
- `FlowDirection`
- `Border`
- `BorderColor`
- `BorderWidth`
- `RowSep`
- `RowSepColor`
- `RowSepWidth`
- `ColSep`
- `ColSepColor`
- `ColSepWidth`

The new format properties of the `mlreportgen.ppt.TableEntry` class are:

- `Border`
- `BorderColor`
- `BorderWidth`
- `ColSpan`
- `RowSpan`
- `TextOrientation`

The `mlreportgen.ppt.TableRow` class has a new property, `Height`.

mlreportgen.ppt.AutoFit: Scale text to fit placeholder or text box in slide

Use an `mlreportgen.ppt.AutoFit` format object to scale text to fit a placeholder or text box in a PPT API slide.

mlreportgen.dom.EmbeddedObject class: Embed files in a report

Use objects of the `mlreportgen.dom.EmbeddedObject` class to embed files in a document rather than linking to the external files. You can then move the report without having to move the files.

In some cases, a file cannot be embedded in a document. An `mlreportgen.dom.EmbeddedObject` causes a document to link to the external file instead of embedding the file in these cases:

- The document type is single-file HTML
- The document type is Microsoft Word and the external file type is *not* one of these types:
 - `xlsx`
 - `pptx`
 - `docx`

mlreportgen.report.HTMLModuleTabs class: Create tabbed panels in HTML reports

An object of the new `mlreportgen.report.HTMLModuleTabs` class adds a widget that consists of a stack of tabbed panels (module tabs) to an HTML report. Selecting a tab displays the contents of the panel. Use an `HTMLModuleTabs` object to display related information in compact form in an HTML or single-file HTML report.

New functions for preparing HTML for conversion to DOM objects

Use the new `mlreportgen.utils.html2dom.prepHTMLString` and `mlreportgen.utils.html2dom.prepHTMLFile` functions to prepare HTML for use with `mlreportgen.dom.HTML` or `mlreportgen.dom.HTMLFile` objects. You use `HTML` or `HTMLFile` objects to convert HTML to DOM objects. The `HTML` and `HTMLFile` objects typically cannot accept the raw HTML output of third-party applications, such as Microsoft Word, that export native documents as HTML markup. See [Convert HTML Content to a DOM Object](#).

mlreportgen.dom.Preformatted class: Preserve white-space formatting of text

Use objects of the `mlreportgen.dom.Preformatted` class to preserve spaces and line feeds in a block of report text and render the text in monospace font. For example, use an `mlreportgen.dom.Preformatted` object to add program code to a report.

mlreportgen.utils.normalizeLinkID: Generate a valid link target ID for all report types

Use the new function `mlreportgen.utils.normalizeLinkID` to generate a link target ID that is valid for Microsoft Word, PDF, and HTML reports. The ID generated by `mlreportgen.utils.normalizeLinkID` consists of the prefix `mw_` and an MD5 hash of the input ID. The generated ID conforms to the Word limitation on ID length and the PDF requirement that an ID begin with an alphabetic character.

Compatibility Considerations

In a future release, a link target ID generated by `mlreportgen.utils.hash` might not be valid for PDF reports. To ensure that a link target ID is valid for all report types, use `mlreportgen.utils.normalizeLinkID` instead of `mlreportgen.utils.hash`.

To generate a link target ID for a Simulink or Stateflow® object, continue to use `slreportgen.utils.getObjectID`. This function generates a link target ID that is valid for all report types.

HTML text no longer wrapped in a paragraph

In R2020a, for HTML reports, text that is appended directly to a document, table, or list is no longer wrapped in a paragraph by default. For example, consider this code:

```
import mlreportgen.dom.*
d = Document('myReport', 'html');

append(d, Text('abc'));
append(d, Text('def'));
close(d);
rptview(d);
```

In previous releases, the text in the generated report looked like:

```
abc
def
```

In R2020a, the text looks like:

```
abcdef
```

For Microsoft Word and PDF reports, there is no behavior change for text appended directly to a document, table, or list. The DOM API wraps the text in a paragraph.

Compatibility Considerations

In R2020a, to generate HTML with text wrapped in a paragraph, put the text in an `mlreportgen.dom.Paragraph` object and append the object to the document. For example:

```
import mlreportgen.dom.*
d = Document('myReport', 'html');

append(d, Paragraph('abc'));
append(d, Paragraph('def'));
close(d);
rptview(d);
```

R2019b

Version: 5.7

New Features

Bug Fixes

Compatibility Considerations

Inline display of equations

You can display an equation in line with the text of a paragraph by using the `DisplayInline` property and the `getImpl` method of an `mlreportgen.report.Equation` reporter object. Set the `DisplayInline` property to `true` and call the `getImpl` method to get a DOM image of the equation that you can append to the paragraph. For example:

```
...
eq = Equation('\int_{0}^{2} x^2\sin(x) dx');
eq.DisplayInline = true;
impl = getImpl(eq,rpt);
append(p,impl);
...
```

If `DisplayInline` is `false`, `getImpl` returns a document part.

More options for vertically aligning inline elements

You have more options for vertically aligning inline elements, such as text or images. You can set the `Value` property of an `mlreportgen.dom.VerticalAlign` object to one of these new values:

- `'text-top'` — Aligns an element with the top of the content area of the parent element.
- `'text-bottom'` — Aligns an element with the bottom of the content area of the parent element.
- `length` — Offsets an element from the baseline by a specified length.

The values `'text-top'` and `'text-bottom'` are not supported for Microsoft Word documents.

SVG images in Microsoft Word reports

You can now include Scalable Vector Graphics (SVG) images in Microsoft Word reports. In previous releases, you could include SVG images in only PDF or HTML reports. The support for SVG images in Word reports affects the report elements in this table.

Report Element	Behavior Change
<code>mlreportgen.report.Figure</code>	<ul style="list-style-type: none"> • The figure reporter can export snapshot images in SVG format for all report types. • The default value of the <code>SnapshotFormat</code> property is now <code>'svg'</code>. In previous releases, it was <code>'auto'</code>. • The value <code>'auto'</code> now indicates <code>'svg'</code> for all report types. In previous releases, <code>'auto'</code> indicated <code>'svg'</code> for HTML and PDF reports and <code>'emf'</code> or <code>'png'</code> for Word reports, depending on the platform.
<code>mlreportgen.report.Equation</code>	The default format used for the snapshot image of an equation is now SVG for all report types. In previous releases, for Word reports, the format was EMF or PNG, depending on the platform. You can use the new <code>SnapshotFormat</code> property to change the format.

Report Element	Behavior Change
<code>mreportgen.dom.Image</code>	You can create an <code>mreportgen.dom.Image</code> object from an SVG image and include it in a Word report. In previous releases, to include an image in a Word report, you had to create the <code>mreportgen.dom.Image</code> object from an image with a format other than SVG.

Compatibility Considerations

Word reports that contain SVG images require Word 2016 or later. In MATLAB R2019b, to generate reports with images that are compatible with earlier versions of Word, you can:

- Set the `SnapshotFormat` property of an `mreportgen.report.Figure` or `mreportgen.report.Equation` object to a value other than `'svg'`. To specify the image format used by default in previous releases of MATLAB, set `SnapshotFormat` to:
 - `'emf'` for a Windows® platform
 - `'png'` for a UNIX® or Mac platform
- Create `mreportgen.dom.Image` objects from images that have a format other than SVG.

Partial support for em units when converting HTML to DOM objects

In previous releases, when you created a DOM document from HTML markup, `em` units in the HTML were ignored. In R2019b, `em` units are converted to points. By default, the font size of one `em` unit is 12 points.

If you create the document by using an `mreportgen.dom.HTML` object, you can specify the font size, in points, of one `em` unit by setting the `EMBaseFontSize` property. For example, if `EMBaseFontSize` is 14, 2 `em` units become 28 points.

Set the `EMBaseFontSize` property in an empty `mreportgen.dom.HTML` object and then add the HTML to the object. For example:

```
import mreportgen.dom.*;
rpt = Document('MyReport', 'pdf');
htmlobj = HTML();
htmlobj.EMBaseFontSize = 14;
htmltext = fileread('MyHTML.html');
append(htmlobj,htmltext);
append(rpt,htmlobj);
close(rpt);
rptview('MyReport.pdf');
```

Setting the `EMBaseFontSize` property in an `mreportgen.dom.HTML` object that already contains the HTML has no effect.

Indentation of ordered lists for Microsoft Word documents

In R2019b, the default indentation for an ordered list in Microsoft Word output is 0.25 inches. The default indentation for an ordered list is now the same as the default indentation for an unordered list.

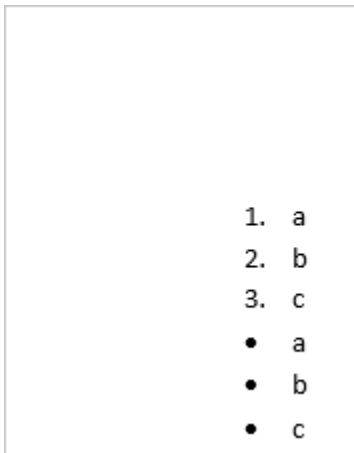
Compatibility Considerations

In R2019b, by default, an ordered list is indented in Word output. In previous releases, by default, an ordered list was unindented in Word output.

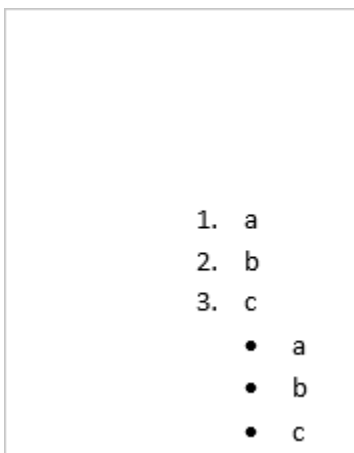
For example, consider the ordered list `o1` and the unordered list `o2`:

```
o1 = OrderedList({'a' 'b' 'c'});  
o2 = UnorderedList({'a' 'b' 'c'});
```

In R2019b, both lists are indented in the Word output. The lists look like:



In previous releases, the unordered list is indented, but the ordered list is not indented. The lists looked like:



In R2019b, to generate an unindented ordered list, set the outer margin of the list to 0.25 inches.

```
o1 = OrderedList({'a' 'b', 'c'});  
o1.Style = {mlreportgen.dom.OuterMargin('0.25in', '0in')};
```

Specify 0.25 inches, not 0 inches, to allow for the 0.25 inches required for the list item numbers.

PropertyFilterFcn and NumericFormat properties in MATLAB variable reporter

The MATLAB variable reporter, `mlreportgen.report.MATLABVariable`, has two new properties:

- `PropertyFilterFcn` — For a variable that contains an object, prevent the display of one or more of the object properties.
- `NumericFormat` — For a variable that contains a noninteger number, format the display of the number or specify the number of significant digits to display.

By default, these properties are empty.

Manipulation of tables and pictures that come from a Microsoft PowerPoint presentation template slide

In R2019b, when you use the PowerPoint API to create and update a presentation that comes from a template, you can modify properties of the tables and pictures that come from the template presentation slides.

For a table that comes from a template slide, you can modify the properties of the `mlreportgen.ppt.TemplateTable` object that represents the table. For a picture, you can modify the properties of the `mlreportgen.ppt.TemplatePicture` object that represents the picture. You can change the position (X and Y properties), Height, Width, and Name. You can also modify the XML markup for the table or picture. In previous releases, to make those kinds of changes, you had to replace the table or picture.

For example, to change the position of a picture, use the following code to find the `mlreportgen.ppt.TemplatePicture` object that represents the picture and modify its X and Y properties.

```
import mlreportgen.ppt.*
ppt = Presentation('newppt', 'oldppt');
open(ppt);
templatePicObj = find(ppt, 'MyPicture');
templatePicObj.X = '100px';
templatePicObj.Y = '200px';
close(ppt);
```


R2019a

Version: 5.6

New Features

Bug Fixes

Report Explorer-based reporter

`mlreportgen.report.RptFile` creates a reporter from a Report Explorer `.rpt` setup file. Adding this reporter to a Report API report setup file generates the setup file contents and adds the contents to the report.

Create a custom reporter

A new method, `mlreportgen.report.Reporter.customizeReporter`, has been added to the Report API Reporter class. Use this method to create a skeleton reporter that you can use as a starting point for creating a custom reporter.

Report API examples

Examples have been added to the documentation under MATLAB Report Generator Task Examples. These examples illustrate using the Report API to generate and format common types of report content, such as wide tables fitting on a page and side-by-side images, figures, and tables.

R2018b

Version: 5.5

New Features

Bug Fixes

MATLAB Variable Reporter: Include information on MATLAB variables in a report

You can use the `MATLABVariable` reporter to format and include the values of MATLAB variables in a report. The reporter formats a variable as a table, paragraph, or string, using the format most suitable for the variable value type.

The `MATLABVariable` reporter reports variables defined:

- In the MATLAB workspace
- In a MAT-file
- As global variables
- In the scope of the reporter

These types of variable values are supported:

- Characters, character arrays, and strings
- Cell vectors and cell arrays
- Logical scalars, logical vectors, and logical arrays
- Numeric scalars, numeric vectors, and numeric arrays
- MATLAB tables
- MATLAB objects, MATLAB object vectors, and MATLAB object arrays
- Simulink objects
- Stateflow objects
- Graphics objects
- MATLAB structures, vectors of structures, and arrays of structures

Table Slicer: Include tables sliced by column to fit on report pages

A `TableSlicer` utility splits a table vertically into multiple slices, each small enough to fit legibly on a page. The Report API `BaseTable`, `LookupTable`, and `Simulink TruthTable` reporters use the `TableSlicer` to generate legible tables.

Enhanced table of contents customization

The `TableOfContents` reporter allows programmatic customization of the table of contents (TOC) that it generates. For example, you can now programmatically specify the number of TOC levels and the TOC leader pattern.

Collapsible table of contents for HTML reports

In HTML reports, you can collapse the table of contents to provide additional width for viewing the report content.

Utility functions

These utility functions enable advanced users to manipulate documents and applications directly, and to obtain, manipulate, and test files, characters, and strings.

<code>mlreportgen.utils.tidy</code>	Correct and clean XML/HTML content
<code>mlreportgen.utils.rptviewer</code>	Display a report
<code>mlreportgen.utils.powerpoint</code>	MATLAB interface to PowerPoint
<code>mlreportgen.utils.word</code>	MATLAB interface to Microsoft Word
<code>mlreportgen.utils.capitalizeFirstChar</code>	Capitalize first character of a string
<code>mlreportgen.utils.normalizeString</code>	Remove extra spaces and line feeds from a string
<code>mlreportgen.utils.hash</code>	Hash a given string
<code>mlreportgen.utils.makeSingleLineText</code>	Force a variable to be a 1-by-N string
<code>mlreportgen.utils.toString</code>	Create a string representation of a MATLAB variable
<code>mlreportgen.utils.fileToURI</code>	Convert file system path to a Universal Resource Identifier (URI)
<code>mlreportgen.utils.findFile</code>	Find a file path
<code>mlreportgen.utils.isFileLocked</code>	Test whether a file is locked
<code>mlreportgen.utils.units</code>	Set of methods to convert from one unit to another

R2018a

Version: 5.4

New Features

Bug Fixes

Custom Finders: Search data containers and return results in reportable form

You can create custom finders to search data containers for specified objects and return the results in reportable form. By using finders, you can separate search logic from report logic in your report generators.

To create a finder, define a MATLAB class that is a subclass of the `mlreportgen.finder.Finder` class. For more information, see [Create and Use a Custom Finder](#).

R2017b

Version: 5.3

New Features

Bug Fixes

MATLAB Reporters: Use MATLAB objects to generate title pages, tables of contents, chapters, figures, and other report elements

You can use the Report API to create MATLAB programs to generate reports. This API simplifies creating reports by providing objects for common report elements, such as title pages and chapters. You can use both Report API and DOM API objects in the same program. Using DOM API objects, you can customize the formatting of the provided Report API elements and insert additional elements in your report.

When you create a report program, you can use the following Report API objects to generate report elements, images, and MATLAB figure snapshots.

- `mlreportgen.report.Report` contains the report.
- `mlreportgen.report.Reporter` base class for reporter objects.
- `mlreportgen.report.ReportLayout` specifies layout options for the report.
- `mlreportgen.report.ReporterLayout` specifies layout options for a reporter.
- `mlreportgen.report.TitlePage` generates the title page of the report.
- `mlreportgen.report.TableOfContents` generates the table of contents of the report.
- `mlreportgen.report.Chapter` generates a chapter of the report.
- `mlreportgen.report.Section` generates a section or subsection of the report.
- `mlreportgen.report.Equation` generates an equation for the report.
- `mlreportgen.report.Figure` generates a snapshot of a MATLAB figure, including an optional caption.
- `mlreportgen.report.FormalImage` generates an image from an image file, including an optional caption.
- `mlreportgen.report.BaseTable` generates a table for the report.
- `mlreportgen.report.InlineContent` fills holes in templates.

PDF Image Format: Generate PDF Reports containing PDF images

The `mlreportgen.dom.Image` object and Report Explorer GUI support adding PDF images (.pdf) to PDF reports. In Report Explorer, PDF images are supported only for `Direct PDF (from Template)` output. You can create PDF images using the `print` command.

Rebuild Template and Style Sheet Cache

Two functions, `rptrebuildcache` and `rptrebuildregistry`, rebuild the Report Explorer template cache and style sheet registry, respectively. Use these functions if you add a template or style sheet to the MATLAB path after opening the Report Explorer or before running a setup file for the first time.

R2017a

Version: 5.2

New Features

Bug Fixes

Compatibility Considerations

Single-File HTML Output: Generate a report as a single HTML file

In Report Explorer, you now can generate an HTML report as a single file by selecting **Single-file HTML (from template)** or **Single-file HTML** as the file output format. The single file contains all the document content, including text, images, and style sheet. You can continue to obtain an HTML report as a zipped or unzipped folder of separate HTML documents. For the folder option, use **HTML (from template)** or **HTML** as the file output format.

Embedded Style Sheet: Convert HTML document containing style sheets to Microsoft Word and PDF

Word and PDF reports preserve the HTML styles defined in the head element of an HTML document. Preserving the HTML styles ensures that both the original HTML document and the generated report use the same formatting.

Page Break Component: Insert page breaks in template-based Microsoft Word and PDF reports

You now can add a page break in your report setup file using the Report Explorer. The component library includes a **Page break** component under **Formatting**. When you insert a **Page break**, the sections or subsections start on new pages in **.docx (Word (from template))**, **PDF (from Word template)**, and **Direct PDF (from Template)** reports.

White-Space Options: Use spaces and line feeds to format code and other text in reports

In the **Paragraph** and **Text** components dialog boxes, a **Preserve white space** format option replaces the **Retain spaces and carriage returns** and **Show text as syntax-highlighted MATLAB code** options. You can now preserve white space, such as spaces and line feeds, in proportionally spaced text. Two new components, **Code** and **Preformatted**, replace the formatting options previously provided in the **Paragraph** component. Both of these new components preserve white space and default to monospace font. The **Code** component differentiates inline code from other paragraph text. The **Preformatted** component preserves white-space formatting in a program listing.

PDF Table Automatic Layout

The default table layout method for PDF has changed from fixed to auto. The fixed method uses the table width and column widths that you specify to lay out the table. If you do not specify a table width and column widths, the fixed method lays out the table so that it occupies the width of the page between the page margins. The table columns, in this case, have equal widths.

Auto layout also uses the table width and column widths that you specify. If you do not specify a table width and column widths, the auto method sizes the table columns to fit the widest content in each column. The table width is the sum of the width of its columns. Auto layout collapses empty columns.

Compatibility Considerations

The layout of a table that does not specify its width and column widths differs in this release from the layout of a table from a previous release. To preserve the layout that the table had in previous releases, add the table format `ResizeToFitContents(false)` to the table style. For example,

```
table.Style=[table.Style, {ResizeToFitContents(false)}];
```


R2016b

Version: 5.1

New Features

Bug Fixes

Compatibility Considerations

Report Forms: Use the Report Explorer and Microsoft Word, PDF, or HTML templates to create form-based reports

You can now use the Report Explorer to create form-based reports. Forms help you to build a report based on a template, simplifying the creation of your report setup. Your template can contain holes that you can fill with content (such as text and tables) or with subforms. You can fill a hole in a subform with another subform. For more information, see [Form-Based Reports](#).

For PDF and Word output types, each subform can contain a page layout, which can include a header, a footer, and page layout information. Each of these parts of the layout can contain holes that you can fill. For example, you can fill a hole in a header with the name of a chapter.

You can also use a subform in a loop to fill the form with content from each iteration.

Report Explorer comes with default templates that include holes, such as a title page, chapter titles, and header and footers. To create a form-based report, in the Report Explorer, select **File > New Form**. The components associated with creating form-based reports appear in the component library under **Form-Based Reporting**.

PDF Image Maps: Add hyperlinks to images in PDF reports

PDF output now supports image maps. In previous releases, only HTML output supported image maps. To see the approach for creating image maps, see [Create Image Maps](#).

PDF Watermarks: Add watermarks to PDF reports

You can add a watermark to a PDF report. A watermark is an image that appears in the background of a page. For example, a watermark can contain the word *Draft* or *Confidential*. It runs behind the text on each page you apply it to. Add the watermark in a PDF page layout using the Report Explorer or the DOM API.

In the API, use a `mlreportgen.dom.Watermark` object to create the watermark based on a `.bmp`, `.jpg`, `.png`, `.svg`, or `.tiff` image. Assign it to the `Watermark` property of an `mlreportgen.dom.PDFPageLayout` object.

In Report Explorer, use the `PDF Page Layout` component to specify the watermark image file name.

Collapsible Table of Contents: Hide table of contents in HTML reports

In HTML reports that contain a table of contents, you can now expand and collapse the TOC structure. For the file output format, select `HTML (from template)` before you generate the report. Expand or collapse each node by clicking the plus or minus sign on the node. Press **Ctrl**+click to expand or collapse the entire structure.

Simplified Table Creation: Create tables using MATLAB tables and categorical arrays

You can now use MATLAB categorical arrays as values for creating tables. You can use this data type when creating `mlreportgen.dom.Table` and `mlreportgen.dom.FormatTable` objects and with the `mlreportgen.dom.Document.append` method.

The `mlreportgen.dom.MATLABTable` class has also been added. Use it to create a DOM object that represents a MATLAB table in a report using MATLAB formatting of tables.

Report Components: Add a line break

You can now add a line break in your report setup file using the Report Explorer. The component library includes a `Line Break` component for this purpose, under the Formatting components. You can add a `Line Break` component as a child of a `Paragraph` component or a `Table Entry` component.

Conversion Templates: Create nested numbered sections

Numbered sections can now include nested numbered sections. For each output type, you can select a template that supports this formatting. Nested numbered sections render with a number for each subsection. Previously, only the number for the top level of a section appeared in the report. For the file format for the report you are generating, select one of the (`from template`) options. Select `Default Numbered` from the template list, or create your own template based on this template.

PDF Fonts: System fonts detected for PDF output

You can now specify any font installed on your system for use in PDF reports without additional configuration for MATLAB Report Generator. In previous releases, you had to configure your Report Generator setup to detect these fonts. PDF font detection applies to reports created using the Report Explorer and the DOM API.

Noto Fonts: New fonts supported for non-English PDF reports

In 2016b, you can use Google® Noto fonts to produce PDF reports in languages that require non-English characters, such as Japanese, Korean, or Cyrillic. Noto font support applies to reports generated by the Report Explorer and the DOM API.

- 1 Download the Noto Sans font that supports your language.
- 2 Install the font to one of these folders. On Linux®:
 - `$user/.fonts`, where `$user` is your home folder
 - `/usr/local/fonts`
 - `/usr/share/fonts`
 - `/usr/X11R6/lib/X11/fonts`

On Windows:

- `$windir\Fonts`, where `$windir` is the Windows folder, for example, `C:\Windows`

On a Mac:

- `$usr/Library/Fonts/`, where `$usr` is your home folder
- `/Library/Fonts/`
- `/System/Library/Fonts/`
- `/Network/Library/Fonts/`

- 3 Update the language font map to set the new font as the default for Report Explorer, or specify the font in your template for DOM output. To update the language font map, see [Configure PDF Fonts](#).

Subdocuments in DOM Reports: Use docview function to unlink subdocuments

On Windows systems, you can remove links to subdocuments from the report document and instead include that content in the master document. With the `docview` function, use the `'unlinkdocxsubdoc'` option.

Diff and Merge: Reuse open MATLAB sessions for diff and merge from external source control tools

Save time by using open MATLAB sessions for XML comparison and merge of Simulink models from your external source control client.

Compatibility Considerations

In previous releases, you could use the `fileComparisonDriver` function to customize external source control tools, but you could not reuse open MATLAB sessions. The `fileComparisonDriver` will be removed in a future release. Instead, use the instructions here: [Customize External Source Control to Use MATLAB for Diff and Merge](#).

R2016a

Version: 5.0

New Features

Bug Fixes

Scalable Report Generation: Generate PDF reports as big as 10,000 pages

In previous releases, the DOM API used Microsoft Word to generate PDF documents from Word documents. In R2016a, the DOM API generates PDF documents directly. The direct-to-PDF capability allows you to programmatically generate large PDF documents on all platforms supported by MATLAB. This release also adds a new output type to Report Explorer, `Direct PDF (from template)`, based on the direct-to-PDF capability. The new output type speeds up PDF generation and expands and simplifies PDF formatting.

Direct-to-PDF allows you to use an HTML template to specify the fixed content and format of a report and holes for inserting generated content. The template can use a predefined set of HTML tags to specify a report's page layout, table of contents location, and the content of page headers and footers. You can also programmatically specify page layout, table of contents location, and page headers and footers. The DOM API automatically generates table of contents and page numbers. See [Create an HTML or PDF Template](#).

See these new classes associated with the new PDF capability in the DOM API:

- `mlreportgen.dom.StyleRef`

Use this class to create running page headers and footers in Word and PDF documents. See [Create Running Page Headers and Footers](#)

- `mlreportgen.dom.PDFPageLayout`
- `mlreportgen.dom.PDFPageHeader`
- `mlreportgen.dom.PDFPageFooter`
- `mlreportgen.dom.Page`
- `mlreportgen.dom.NumPages`
- `mlreportgen.dom.CSSProperty`
- `mlreportgen.dom.CSSProperties`
- `mlreportgen.dom.FOProperty`
- `mlreportgen.dom.FOProperties`
- `mlreportgen.dom.PageRef`
- `mlreportgen.dom.Heading1`
- `mlreportgen.dom.PageBreak`
- `mlreportgen.dom.Leader`
- `mlreportgen.dom.ListStyleType`

Table of Contents: Add TOCs programmatically

In R2016a, you can create a placeholder for a table of contents in your DOM API report program. Use the `mlreportgen.dom.TOC` class to create the placeholder. For more information on creating a table of contents, see [Create a Table of Contents](#).

Page Numbers: Create page number placeholders programmatically

In R2016a, you can create page number placeholders programmatically in Word and PDF reports. Previously you used only a template to create the placeholders. Updating a Word document replaces

the placeholders with automatically generated page numbers. Closing a PDF DOM document similarly replaces the placeholders with automatically generated page numbers. For more information, see the `mlreportgen.dom.Page` class.

HTML Text Component: Convert HTML to Word or PDF

The new HTML Text component allows you to include HTML markup in a Report Explorer report. If you specify Word or PDF as the report output type, the included HTML text is converted to Word or PDF. The conversion preserves the text's HTML-specified format. You can enter the HTML text in the component or specify a file or workspace variable that contains the text.

pptview Function: Open PowerPoint presentation or convert it to PDF

The `pptview` function opens a PPT presentation in PowerPoint on Windows or Apache OpenOffice™ on Linux. If you have Microsoft Office installed, you can also use this function to convert a PowerPoint presentation to PDF format.

Cross-Platform PDF Viewer: View PDF with built-in viewer

Previously, the Report Generator displayed PDF reports in Adobe Acrobat on Windows and in the PDF viewer you specified on other platforms. In R2016a, the Report Generator displays PDF reports on the same built-in viewer on all platforms. This change avoids the need for you to interact with a new Acrobat security dialog box on Windows every time you display a report. You also do not need to set up a viewer on Linux and Macintosh systems.

Note The new PDF viewer does not display a report's bookmarks.

Functionality Being Removed or Changed

Functionality	Result	Use Instead	Compatibility Considerations
<code>mlreportgen.dom.DOCXPageMargins</code> class	Still works for Microsoft Word output	<code>mlreportgen.dom.PageMargins</code>	To prevent your code from being output dependent, replace instances of <code>DOCXPageMargins</code> with <code>PageMargins</code> .
<code>mlreportgen.dom.DOCXPageSize</code> class	Still works for Word output	<code>mlreportgen.dom.PageSize</code>	Replace instances of <code>DOCXPageSize</code> with <code>PageSize</code> .
<code>mlreportgen.dom.DOCXRawFormat</code> class	Still works for Word output	<code>mlreportgen.dom.PageRawFormat</code>	Replace instances of <code>DOCXRawFormat</code> with <code>PageRawFormat</code> .

Functionality	Result	Use Instead	Compatibility Considerations
<code>mreportgen.dom.DOCXSection</code> class	Still works for Word output	<code>mreportgen.dom.DOCXPageLayout</code> or <code>mreportgen.dom.PDFPageLayout</code>	Replace instances of <code>DOCXSection</code> with <code>DOCXPageLayout</code> . If appropriate, add a <code>PDFPageLayout</code> object.
<code>CurrentDOCXSection</code> property	Still works for Word output	<code>CurrentPageLayout</code>	Replace instances of <code>CurrentDOCXSection</code> property with <code>CurrentPageLayout</code> property.

R2015b

Version: 4.2

New Features

Bug Fixes

Programmatic interface for adding content to PowerPoint presentations

You can use the PPT (PowerPoint) API to add generated content to an existing PowerPoint presentation or to create a complete PowerPoint presentation programmatically. Your presentation can capture dynamic information from a MATLAB program without you making manual updates to the presentation. To format the layout and general look of your presentation, use template slide masters, style layouts, and styles. To format specific presentation content, you can use the PPT API format objects and properties.

For examples of PPT API code, see:

- Update Presentation Content Programmatically
- Create a Presentation Programmatically

DOM API object display options

For Microsoft Word reports, you can use an `mlreportgen.dom.Display` object to specify whether to display an `mlreportgen.dom.Text` object. For HTML reports, you can use a `Display` object to specify the how to display DOM objects such as text, paragraphs, images, list items. For HTML reports, some display options are:

- Display as an inline or block element.
- Display the element similar to a table column, table caption, list element, or a few other elements.
- Suppress the display of the object.

For details, see `mlreportgen.dom.Display`.

DOM API horizontal rule

You can use an `mlreportgen.dom.HorizontalRule` object to specify a horizontal line to visually divide report content.

For details, see `mlreportgen.dom.HorizontalRule`.

R2015a

Version: 4.1

New Features

Bug Fixes

Support for appending HTML string or file to a Word or PDF report generated by the Document Object Model (DOM) API

You can append HTML markup or the contents of a whole HTML file to a programmatic report written with the Document Object Model (DOM) API to:

- Convert an existing HTML report to a Microsoft Word or PDF report.

You can append the HTML markup for a report to a DOM report, which you can then generate in Word or PDF format. For example, you can use a MATLAB `printf` statement to generate an HTML report file, and then append that HTML file to a DOM report to generate a PDF or Word version of the report. Another example is to convert a report generated by Model Advisor from HTML to Word or PDF.

- Add content based on HTML markup.

You can append the HTML code for a report to a DOM report, using the HTML code as a building block in a DOM report that includes other report elements.

Perform these steps to append HTML to a DOM report.

- 1 Ensure that the HTML code complies with HTML standards, such as including beginning and end tags (for example, `<p>` and `</p>`). Consider using a utility such as HTML Tidy.
- 2 In a DOM report, use `addHTML` to append HTML code or `addHTMLFile` to append an HTML file to a document or document part.

For example, this DOM code creates a Word report that displays Hello World, based on the HTML code that you append to the report.

```
import mlreportgen.dom.*;
d = Document('MyReport', 'docx');
addHTML(d, '<p style="color:blue"><b>Hello World</b></p>');
```

For information about using the `addHTML` or `addHTMLFile` methods, see `mlreportgen.dom.Document.addHTML` or `mlreportgen.dom.Document.addHTMLFile`.

Single-file output option for HTML reports generated by DOM API

To simplify display of an HTML report, you can specify that the DOM API generate the report as a single HTML file that includes the report images.

Specify `'html-file'` for the `type` input argument for a `Document` or `DocumentPart` constructor. For details, see the `mlreportgen.dom.Document` reference page.

Simplified table formatting with DOM API

In R2015a, the DOM API includes several enhancements to simplify creating and formatting tables.

- You can use an `mlreportgen.dom.ResizeToFitContents` format object in the `Style` property for `mlreportgen.dom.Table` or `mlreportgen.dom.FormatTable` objects. The `ResizeToFitContents` specifies whether to resize columns to fit the widest content in a column.
- You can specify an `mlreportgen.dom.Height` format object in the `Style` property of an `mlreportgen.dom.TableEntry` object. The height of a row is the tallest height specified for a

table entry in that row. If no entries in the row specify a height, then the row height is specified by the `Height` property of the `TableRow` object.

- Table entries can now inherit the border formatting specified in a `TableEntriesStyle` property of `mlreportgen.dom.Table` or `mlreportgen.dom.Table` objects.

Container for generating high-level HTML elements

You can use the `mlreportgen.dom.Container` class to create high-level HTML elements not otherwise supported by the DOM API, such as `div`, `section`, or `article`.

You can also use a `Container` object to simulate format inheritance in Word reports.

For details, see `mlreportgen.dom.Container`.

Images and text for DOM report links

You can append text and images to `mlreportgen.dom.ExternalLink` and `mlreportgen.dom.InternalLink` objects.

For details, see `mlreportgen.dom.ExternalLink.append`.

R2014b

Version: 4.0

New Features

Bug Fixes

Report formatting based on Word and HTML templates

You can use Microsoft Word or HTML templates for reports that you create using:

- The Report Explorer. For details, see [Generate a Report Using a Template](#).
- MATLAB report objects. See [Document Object Model](#).

MATLAB report objects for creating report scripts

You can use the DOM (Document Object Model) API to create MATLAB programs to generate reports. The DOM API provides a set of objects for creating text, paragraphs, tables, images, lists, and other kinds of report content. When you create a report program, you can:

- Produce Microsoft Word, HTML, and PDF output using the same code.
- Specify formatting for report objects, such as color and bold for text objects.
- Use format objects (for example, `Bold` and `FontFamily`) to format report objects.
- Add MATLAB data directly to report objects (for example, a char array to a `Text` object).
- Use formatting and fixed content that is defined in a Word or HTML template.
- Create form-based reports that contain fixed and generated content.
- Modularize a report into parts for generation of reports that have repetitive elements of the same format.

For information about how to create report programs, see [Document Object Model](#).

Fast file converter with reduced memory requirements

Generate template-driven reports much faster than in previous releases, without using Java memory.

To take advantage of these performance advantages, use one of these **File format** report options:

- HTML (from template)
- PDF (from template)
- Word (from template)

For details, see [Report Generation Using Templates](#).

Zippered package option for HTML reports

You can use a zip file to package reports that use an HTML template. Zip file packaging compresses and consolidates the files. You can also package the HTML files as unzipped files or as both zipped and unzipped files.

Fill-in-the-blanks Word and HTML forms for generating reports

You can use the DOM (Document Object Model) report objects in a MATLAB program along with Word and HTML fill-in-the-blanks templates that contain fixed and holes to include generated content. The report program uses report objects to generate the variable information. See [Form-Based Reporting](#).

Color settings preferences in MATLAB Comparison Tool

You can now change and save your diff color preferences for the MATLAB Comparison Tool. You can apply your color preferences when comparing text files, MAT-files, variables, zip files or folders.

R2014a

Version: 3.16

New Features

Bug Fixes

OpenOffice support

On Linux and Macintosh platforms, the report output appears in Apache OpenOffice if you:

- Set **File format** to either Word Document (RTF) or Rich Text Format
- Enable **View report generation**

In the MATLAB Report Generator Preferences, the **View command** preference default now enables viewing the report in OpenOffice for Linux and Macintosh platforms.

For details, see Report Output Format and Stylesheet.

R2013b

Version: 3.15

Bug Fixes

R2013a

Version: 3.14

Bug Fixes

R2012b

Version: 3.13

Bug Fixes

Non-English character set report formats

The MATLAB Report Generator provides basic PDF font support for:

- English
- 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 style sheet.

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. For details, see [PDF Fonts for Non-English Platforms](#).

R2012a

Version: 3.12

New Features

Bug Fixes

Compatibility Considerations

Enhanced Table Components

You can use the following new components to create and format a table:

- Table
- Table Column Specification
- Table Header
- Table Body
- Table Footer
- Table Row
- Table Entry

Formatting options that the Table component and its child components support include:

- Text color and alignment
- Row background colors
- Inclusion of images
- Inclusion of hyperlinks
- Table cells spanning columns and rows

You can conditionally control the content or format of a table.

For more information, see [Table Formatting Components](#).

Compatibility Considerations

Releases earlier than R2012a included a different Table component. Starting in R2012a, that component is now called the `Array-Based Table` component.

The `Array-Based Table` component produces the same output, and has the same properties, as it did when it was called the `Table` component. The `Array-Based Table` component converts a rectangular array into a table and inserts the table into the report.

If a report that was created in a previous release already includes a `Table` component, the output is the same as in previous releases. To change property values for that component in R2012a or later, use the `Array-Based Table` dialog box.

Title Page Formatting Enhancements

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](#).

Text Component Supports Subscripts and Superscripts

You can include a subscript or superscript in a paragraph, using a `Text` component.

R2011b

Version: 3.11

New Features

Bug Fixes

Full Page Image Option for PDF Reports

To display full-page images in PDF reports, set the **Paper orientation** property to the new **Full page image (PDF only)** option. The following components provide this new option:

- Axes Snapshot
- Figure Snapshot

The Image component has a **Full page image (PDF only)** check box, which provides the same functionality as the **Full page image (PDF only)** property option.

Include Legal Notice, Report Creation Date, and Copyright on the Title Page

The Title Page component includes a new **Display legal notice on title page** check box. Use this option to include the legal notice, report creation date, and copyright information on the title page of reports that use PDF or Microsoft Word format.

R2011a

Version: 3.10

New Features

Bug Fixes

Improved PDF Pagination

Reports generated using PDF output format produce better pagination, including:

- Section titles are on the same page as the associated text (now section titles cannot appear as orphans at the bottom of a page).
- Captions are on the same page as their associated images.

Export XML Comparison Results to the Workspace

You can now export XML comparison results to the MATLAB base workspace. You can use the results data for tasks such as further analysis or incorporating into different reports.

For details, see [Export Results to the Workspace](#) in the MATLAB Report Generator User's Guide.

R2010b

Version: 3.9

New Features

Bug Fixes

Complete Text Included in Reports

Any component that displays text in reports now displays the whole text, without any truncation.

Linking to MATLAB Commands

Using the `Link` component, in a report you can now include a link that executes a MATLAB command.

Insert Variable Component Enhancements

The `Insert Variable` component includes several new options for specifying the content that the component displays and the table layout that the component uses.

Nest Setup File Component Supports Relative Links

The `Nest Setup File` component includes a new **Link to external report is relative** option, which makes the link to the nested report a relative link. This feature facilitates including a report on a Web site.

Enhanced Error Handling Code for Evaluate MATLAB Expression Component

The `Evaluate MATLAB Expression` component includes enhanced error handling code that you can easily customize (for example, the error handling code can stop report generation, with or without displaying an exception). The default error handling code now supports MATLAB class exception handling, using the `evalException` variable instead of `lastError`.

Transposable Columns for Handle Graphics Summary Table Component

For the `Handle Graphics Summary Table` component, use the new **Transpose table** check box to change the summary table rows into columns in the generated report. Specifying this option puts the property names in the first column and the values in the other columns.

Additional Date Format for Title Page Component

The `Title Page` component includes an additional date format for the **Include report creation date** field: `yyyy/mm/dd`.

Improved XML Comparison Report

Reports for comparisons of XML files have a new faster user interface, with changed parameters displayed in a separate panel for easier review. The enhanced report now has color highlighting for new and changed items.

For details, see *Explore the XML Comparison Report* in the MATLAB Report Generator User's Guide.

R2010a

Version: 3.8

Bug Fixes

R2009b

Version: 3.7

New Features

Bug Fixes

Compatibility Considerations

Style Sheets and Style Sheet Data Items Now Alphabetized

When you create or modify a style sheet, the Report Explorer **Options** pane now alphabetically sorts the style sheets and the data items in each style sheet category.

Compatibility Considerations

The order in which a style sheet or data item appears in a list may have changed from the order used in previous releases.

Improved Images in Word and RTF Reports on Windows Platforms

To provide better graphics quality in reports using the Word document format or RTF (Rich Text format) on Windows platforms, the default format for Handle Graphics® images is now Windows metafile format. The default had been black and white TIFF.

You can control the image format using the MATLAB preferences or the MATLAB Report Generator preferences, or from snapshot components such as the Axes Snapshot component.

Compatibility Considerations

To change the image format back to the previous default of black and white TIFF, change the preferences.

Required Products Information for MATLAB/Toolbox Version Number Component

The MATLAB/Toolbox Version Number component now supports options to display information based on whether a Simulink model or Stateflow chart requires a MathWorks product. This feature does not apply to reports on MATLAB or toolbox code.

Adobe Illustrator Image File Format No Longer Supported

Compatibility Considerations

The **Image file format** options for the Axes Snapshot and the Figure Snapshot components no longer include Adobe Illustrator. If you wish to integrate a MATLAB Report Generator image into Adobe Illustrator, specify a image file format supported by Adobe Illustrator, such as PDF, and then open the image file in Adobe Illustrator.

Navigation Controls for XML Comparison Report

The XML comparison report has new navigation controls to step through differences in the report. You can use the toolbar buttons or the **XML** menu to move to the next or previous group of changes.

For details, see Explore the XML Comparison Report in the MATLAB Report Generator User's Guide.

R2009a

Version: 3.6

No New Features or Changes

R2008b+

Version: 3.5

New Features

Bug Fixes

Comparison of XML Files

New comparison report for pairs of XML files.

For details, see [Comparing XML Files](#) in the MATLAB Report Generator documentation.

See also the following new example: [Comparing XML Files](#).

R2008b

Version: 3.4

Bug Fixes

R2008a

Version: 3.3

Bug Fixes

R2007b

Version: 3.2.1

New Features

Bug Fixes

Text Formatting Options for Title Page, Text, and Paragraph Components

This release includes new text formatting options for the Title Page, Text, and Paragraph components.

For more information, see the MATLAB Report Generator documentation.

R2007a

Version: 3.2

Bug Fixes

R2006b

Version: 3.1

Bug Fixes

R2006a+

Version: 3.0.1

Bug Fixes

R2006a

Version: 3.0

New Features

Bug Fixes

Compatibility Considerations

User Interface and Performance Enhanced

The Report Explorer interface has a number of enhancements. The **View** menu has commands to increase and decrease font size and to show the message window if it is not displayed.

MATLAB Report Generator performance has improved. Reports are often generated more rapidly and with less chance of running out of memory.

XML File Format Changed

The format of XML files has changed in V3.0 (R2006a).

Compatibility Considerations

In V3.0 (R2006a), Report Explorer and the `rptconvert` command can convert XML files produced by the report generation process in V2.3.1 (R14SP3) and previous versions. However, in V2.3.1 (R14SP3) and previous versions, Report Explorer and the `rptconvert` command cannot convert XML files produced by the report generation process in V3.0 (R2006a).

R14SP3

Version: 2.3.1

New Features

Bug Fixes

Style Sheets Modify PDF Headers and Footers

This release adds support for modifying the content of headers and footers in PDF reports. This feature is implemented using style sheet cells and cell groups. A cell group contains one or more style sheet cells. A style sheet cell is a set of values that determine the content of a particular portion of the header or footer for a page.

Two cell groups, **Header Content** and **Footer Content**, are available for PDF reports. You can use templates to specify XML code that defines the content for a header or footer.

R14SP2+

Version: 2.3

New Features

Bug Fixes

Stylesheet Editor

You can use the Stylesheet Editor to customize the formatting of your HTML, PDF, and Word documents. For example, you can specify fonts, paper layout, table presentation, as well as other characteristics of your final report.

Customizing reports by editing style sheets gives you the flexibility to create reports that conform to corporate style guidelines and requirements.

To open the Stylesheet Editor, right-click **Report Generator** in the Outline pane of the Report Explorer and then click **Edit Stylesheet** on the shortcut menu.

For more information about using the Stylesheet Editor, see Create Custom Stylesheets in the MATLAB Report Generator documentation.

Table Cell Spanning

You can create advanced table layouts for property tables. The advanced layouts include cells that span multiple rows or columns.

Generating Microsoft Word Documents as .doc Files

You can generate reports that are saved as Microsoft Word documents with the .doc extension. You can also convert XML source files that are generated by the report generation process to .doc files.

The reports you create are easier to distribute via e-mail, to share with others, and to manage.

You must have Microsoft Word software installed to use this feature, and you must be running the MATLAB Report Generator software on a Microsoft Windows system.

To use this feature select **Word Document** from the **Stylesheet** drop-down list in the Convert Source File dialog box.

Improved Graphical User Interface

This version of the MATLAB Report Generator software has the following changes and improvements to the graphical user interface:

- The File Converter has a **Source File** drop-down list. Click **Browse** to navigate to a folder to select a source file. The default folder is the current folder.
- In the File Converter, you can select a style sheet from the drop-down list and click **Edit** to open the Stylesheet Editor.
- An improved mechanism detects and displays errors that occur when you attempt to add a component that makes the style hierarchy invalid. Users of earlier versions may notice that warnings are now issued for components that had no problems in earlier versions. In this release, warning messages contain instructions about how to fix the problem.
- The **Figure Screen Capture** component has been improved to capture window decorations such as menus, title bars, and toolbars.