Oxygen PDF Chemistry 26.0
User Guide

XML + CSS = PDF
Contents

Chapter 1. Getting Started......................................................................................................................... 6
  Installing.................................................................................................................................................. 7
  Configuring Chemistry as an Oxygen External Tool.............................................................................. 8
  Command Line Interface.......................................................................................................................... 8
  Generating PDF Output from DITA Content Using a Command Line..................................................... 15
  Environment Variables............................................................................................................................ 15
  Your First Document............................................................................................................................... 16
  Debugging the CSS................................................................................................................................... 16
  Security Policy Configuration.................................................................................................................. 19

Chapter 2. Styling for Print......................................................................................................................... 21
  Associating a CSS to a Document........................................................................................................... 21
  Page Formatting...................................................................................................................................... 21
    The @Page Rule.................................................................................................................................... 21
    Columns................................................................................................................................................ 27
  Controlling Page Breaks.......................................................................................................................... 28
    Forcing Page Breaks............................................................................................................................... 28
    Avoiding Page Breaks............................................................................................................................. 28
    Page Breaks Between Named Pages..................................................................................................... 29
    Page Breaks Between Lines: Widows and Orphans............................................................................. 30
  Chapter Page Placement and Styling....................................................................................................... 30
  Headers and Footers................................................................................................................................. 32
    Extracting Text from Document Using String Sets............................................................................. 32
    Extracting Text from Document Using XPath..................................................................................... 33
    Multiple Lines in Headers and Footers.................................................................................................. 35
    The Page Counter................................................................................................................................ 35
    How to Use Flexible Layout in Headers and Footers.......................................................................... 36
    How to Style a Part of the Text from the Header.................................................................................. 36
    How to Use Dynamic Images in Headers and Footers......................................................................... 38
    How to Add a Link in Headers and Footers.......................................................................................... 38
  Footnotes.................................................................................................................................................. 39
    How to Add a Separator Above the Footnotes.................................................................................... 41
Cross References.................................................................................................. 41
  Internal links................................................................................................. 41
  External links............................................................................................... 42
Creating a Table of Contents (TOC)..................................................................... 43
Annotations........................................................................................................ 45
  Comments and Tracked Changes - HTML Fragment.................................... 45
PDF Output......................................................................................................... 47
  Bookmarks.................................................................................................... 47
  Metadata....................................................................................................... 49
  Named Destinations (Anchors)...................................................................... 52
  Accessibility (508 Compliance)..................................................................... 52
  Archiving...................................................................................................... 55
Hyphenation......................................................................................................... 55
  Hyphenation Dictionaries............................................................................ 56
  Installing New Hyphenation Dictionaries.................................................... 57
  How to Alter a Hyphenation Dictionary....................................................... 58
  How to Disable Hyphenation for a Word..................................................... 59
  How to Hide Hyphens.................................................................................. 60
  How to Force or Avoid Line Breaks at Hyphens.......................................... 60
Using XPath in CSS............................................................................................ 61
Using the :before(n) and :after(n) CSS Pseudo-Elements............................... 61
Change Bars........................................................................................................ 62

Chapter 3. Layout.................................................................................................. 64
  Tables............................................................................................................ 64
  Lists.............................................................................................................. 71
  Floats........................................................................................................... 74
  Aligning Blocks Horizontally...................................................................... 74
  Rotating Blocks........................................................................................... 75

Chapter 4. Graphics............................................................................................. 77
  Images.......................................................................................................... 77
    Supported Image Types ............................................................................. 77
    Setting Image Width and Height............................................................... 78
  Image Resolution......................................................................................... 79
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Fonts</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>Supported Fonts</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>Basic Fonts</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>Fallback Fonts</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>Using Web Fonts</td>
<td>93</td>
</tr>
<tr>
<td></td>
<td>Using Installed Fonts</td>
<td>93</td>
</tr>
<tr>
<td></td>
<td>Using Local Font Files</td>
<td>94</td>
</tr>
<tr>
<td></td>
<td>Font Embedding</td>
<td>95</td>
</tr>
<tr>
<td></td>
<td>Font Ligatures</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>Font Alternates</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>Using Simulated (Synthetic) Styles</td>
<td>100</td>
</tr>
<tr>
<td>6</td>
<td>Localization</td>
<td>103</td>
</tr>
<tr>
<td>7</td>
<td>Tutorials</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>How To Insert the Current Date in the Cover Page</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>How To Apply Chemistry on Multiple Files</td>
<td>106</td>
</tr>
<tr>
<td></td>
<td>Full Example: How to Format a Book</td>
<td>107</td>
</tr>
<tr>
<td>8</td>
<td>Appendix</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>CSS Media Rules</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>CSS Selectors</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>CSS Functions</td>
<td>117</td>
</tr>
<tr>
<td></td>
<td>CSS Units</td>
<td>118</td>
</tr>
<tr>
<td></td>
<td>CSS Properties</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td>Standard W3C CSS Properties</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td>Extension CSS Properties</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>XML Support</td>
<td>138</td>
</tr>
</tbody>
</table>
Contents

Configuration File......................................................................................................................... 139
Technical Issues.......................................................................................................................... 139
Index............................................................................................................................................. a
Copyright...................................................................................................................................... d
1. Getting Started

What is Oxygen PDF Chemistry?

Oxygen PDF Chemistry allows you to obtain PDF output from HTML or XML documents simply by styling them with CSS. It is a CSS Paged Media processor based on the open-source Apache FOP XSL-FO engine. Its main purpose is to provide you with a simple tool that allows you to leverage your CSS knowledge to create printable deliverables. While the support for fancy formatting is limited, it is a great processor for generating technical documentation.

Why Use CSS Instead of XSL-FO?

There is not one simple answer. It really depends on the needed output, but both methods will generate PDF documents without any problem. So why use CSS?

- CSS is a much easier language to learn and master than XSLT.
- More people know CSS rather than XSLT (and there are more CSS tutorials available than XSLT tutorials).
- The CSS debugging can be done directly from any browser or from Author mode/
- In most cases, the CSS customization will cover all of the user's needs.
- The CSS customization can be reused in any HTML output.
- The cascading priority scheme makes CSS easier to extend (no need to know previous rules to create new ones).
- PDF processors usually proposes both solutions (Oxygen PDF Chemistry, Antenna House, Prince XML).

Using CSS for Print

While CSS was designed as a language to style web pages, it turned out to be a good candidate to also format printed materials. The so-called CSS paged media module was thus added to the existing standard by W3C.

New concepts added by this module include:

- Page size
- Page breaks
- Headers and footers
- Footnotes
- Links containing the target page number
- Leaders that are used to fill spaces

For more information, see the following paged media specifications:
Installing

Prerequisite:
Oxygen PDF Chemistry is a Java application and requires the following Oracle Java version, depending on your operating system:

- **Windows** or **Linux** - Java JRE version 17.
- **macOS** - Java JDK version 17.

To install Oxygen PDF Chemistry on your platform, follow this procedure:

1. Download the distribution for your particular platform from the Oxygen website: [https://oxygenxml.com/pdf_chemistry/download.html](https://oxygenxml.com/pdf_chemistry/download.html).
2. Follow the instructions provided on that web page.
3. Add the full path for that folder to your PATH environment variable (for information about how to do this, see the related links below, depending on your operating system).
4. If you purchased the product, place the license key in the installation directory. The name of this file should be `licensekey.txt`. Note that unlicensed Oxygen PDF Chemistry produces watermarked output.
5. Test the installation by typing the following command in a console:

    **Windows**:
    ```
    chemistry -v
    ```

    **macOS or Linux**:
    ```
    sh chemistry.sh -v
    ```

**Result:** It should display the version.
Configuring Chemistry as an Oxygen External Tool

Oxygen PDF Chemistry can be run as an external tool, directly from the Oxygen XML Editor/Author environment.

To configure Oxygen XML Editor/Author to run Oxygen PDF Chemistry over the currently open XML file, follow this procedure:

1. In Oxygen XML Editor/Author, select Tools > External Tools > Configure.
2. Click the New button to add a new external tool.
3. In the Name field, enter "chemistry".
4. In the Command Line field, enter the following command, depending on your operating system (make sure you replace my_stylesheet.css with the path to your CSS):
   - Windows: cmd /C chemistry.bat -in "${cf}" -css "my_stylesheet.css" -out "${cfd}/${cfn}.pdf" -show-pdf
   - macOS or Linux: sh chemistry.sh -in "${cf}" -css "my_stylesheet.css" -out "${cfd}/${cfn}.pdf" -show-pdf

   Note: If you have not configured the PATH environment variable to point to the installation directory, use the full path to the chemistry executable script. If the installation directory contains spaces, use quotes around the full path of the script.

If you are using Oxygen PDF Chemistry distributed within Oxygen:

   - Windows: cmd /C "$\{oxygenInstallDir\}/oxygenChemistry.bat" -in "$\{cf\}" -css "my_stylesheet.css" -out "$\{cfd\}/$\{cfn\}.pdf" -show-pdf
   - macOS or Linux: sh "$\{oxygenInstallDir\}/oxygenChemistry.sh" -in "$\{cf\}" -css "my_stylesheet.css" -out "$\{cfd\}/$\{cfn\}.pdf" -show-pdf

Command Line Interface

You can process an XML or HTML document from a command-line interface like this:

```bash
chemistry -in my_file.xml -out my_file.pdf

chemistry -in my_file.html -out my_file.pdf
```
You can specify one or more CSS files to be used. If the document is an HTML document, it may include references to other stylesheets or styles can be embedded using the `<style>` element. The command-line CSS files take precedence over files referenced from the document:

```
chemistry -in my_file.html -out my_file.pdf -css style1.css style2.css
```

**Required Parameters**

- **-in**
  
  The input XML or HTML file in URI or File syntax.
  
  -in http://my.example.com/my_file.html
  -in C:\my.folder\my_file.xml

- **-out**
  
  The output PDF file in File syntax.
  
  -out C:\publishing\my_file.pdf

**Optional Parameters**

- **-catalog-prefer**
  
  Catalog preference mode. Possible values are: ‘system’ or ‘public’. Default is ‘public’.

- **-catalogs**
  
  The path to one or more XML catalogs, in URI or File syntax. You can separate the paths by using the “;” (semi-colon) character. You can use a catalog to solve DTD or Schema references to local files. If the list of catalog files is big, you may run into command-line size limitations. In this case, consider passing it as the XML_CATALOG_FILES (on page 15) environment variable.

- **-css**
  
  A list of CSS files, in URI or File syntax, separated by spaces.
  
  -css http://my.example.com/css/pages.css
  http://my.example.com/css/fonts-and-colors.css

If you want to process an HTML document, this parameter is optional. If you specify a CSS, it is considered to be more important than the ones referenced from the document (for example, using the `<link>` element or the `xml-stylesheet` processing instruction).

- **-disable-xinclude**

  A security setting that disables XML inclusions (XInclude). This is recommended when using Chemistry in a Web context. In addition, you should use a Java Security Manager to control the exact permissions granted to the processor.

- **-disable-xxe**
A security setting that disables XML external entities. This is recommended when using Chemistry in a Web context. In addition, you should use a Java Security Manager to control the exact permissions granted to the processor.

-dump-fo

Dumps the FO file (before converting it to PDF) in the same location as the output file. This can be used for debugging purposes.

-dump-styled-content

Dumps the intermediate, annotated XML file in the same location as the output file. This can be used for debugging purposes.

-drop-block-margins-at-page-boundary

Specifies that the top and bottom margins associated with a block element should be discarded when the block is at the top or bottom of the page. Allowed values:

  • yes (default)
  • no

-enable-latin-glyph-substitutions

When set to yes (default), glyph substitution is enabled (if the particular font supports it). This applies to Latin-based scripts only (the substitutions are always enabled in other types of scripts). If you encounter problems rendering or copying accented glyphs (i.e. umlauts or other diacritics), it might be helpful to set this parameter to no to disable the font glyph substitutions. Another example of a case when you might need to disable the substitutions is a situation where an accented character cannot be mapped to a compound glyph, resulting in the glyph not being rendered in the PDF output.

Warnings:

  • Disabling substitutions also disables Latin ligatures.
  • Disabling substitutions is not recommended unless absolutely necessary. It is better practice to use another font if you can find one that does not have the rendering issues.

-enable-latin-ligatures

Used to enable ligatures between some of the characters from the Latin glyph range. The character sequences that might be combined are highly dependent on the font, but they are usually: "ff", "ff", "ff".
Note:
This parameter is deprecated. Use the `font-variant-ligatures` CSS property instead. See *Font Ligatures (on page 99)* for more information.

- **expand-xpath-in-svg-templates**
  When set to yes (default), each XPath expression that has a format of `${expression}` inside SVG templates files (with a `.template.svg` file extension) will be expanded.

- **fonts-dir**
  The directory with additional fonts. The TTF files should be placed directly in it (no subdirectory).

- **help**
  Prints the usage information.

- **http-proxy-host**
  The HTTP proxy host.

- **http-proxy-port**
  The HTTP proxy port.
  ```
  -http-proxy-host my.proxy.server -http-proxy-port 3128
  ```

- **hyph-dir**
  The directory that contains additional hyphenation dictionaries. The XML files should be placed directly in it (not a subdirectory) and they should be named using the language code (e.g. `en.xml`). For more information about adding or altering dictionaries, see: *Hyphenation (on page 55)*.

- **image-resolution**
  The resolution (in DPI) of the raster images (an integer), for images that do not provide this meta information. For changing the resolution using CSS, see: *Image Resolution (on page 79)*.
  ```
  -image-resolution 72
  ```

- **licensekey-dir**
  Specifies the directory where the license key file is located. The license key file should have the name `licensekey.txt`.

- **no-aggressive-hyphenation**
  Oxygen PDF Chemistry normally uses an aggressive technique to create hyphenation points at underscores, dots, and case changes. To disable this functionality, use this `-no-aggressive-hyphenation` parameter.

- **no-network**
Denies access to all your network connections. If your XML document or CSS files reference remote resources, the transformation will fail. This parameter is ignored when the `-security-policy` file is specified.

-no-rtl-mirroring

Disables switching of left and right margins, padding, and borders for right-to-left content. To make stylesheet development easier, the left margin automatically becomes the right margin when the paragraph has an RTL orientation.

-pdf-a

Produces output that conforms to the PDF/A archiving standards. The conversion will fail if fonts are not embedded. See: Archiving (on page 55).

-pdf-security-encrypt-metadata

Encrypts the metadata. By default, it is active when other security parameters are set.

-pdf-security-encrypt-structure-element

Encrypts the structure elements. By default, inactive when other security parameters are set.

-pdf-security-restrict-accesscontent

Restricts extracting text and graphics. Used for protecting the PDF Document. The restriction is off by default.

-pdf-security-restrict-annotations

Restricts filling in existing interactive forms. Used for protecting the PDF Document. The restriction is off by default.

-pdf-security-restrict-assembledoc

Restricts assembling document (e.g. adding pages). Used for protecting the PDF Document. The restriction is off by default.

-pdf-security-restrict-copy

Restricts copying content. Used for protecting the PDF Document. The restriction is off by default.

-pdf-security-restrict-edit

Restricts copying content. Used for protecting the PDF Document. The restriction is off by default.

-pdf-security-restrict-fillinforms

Restricts filling in existing interactive forms. Used for protecting the PDF Document. The restriction is off by default.

-pdf-security-restrict-print

Restricts printing. Used for protecting the PDF Document. The restriction is off by default.

-pdf-security-restrict-printthq
Restricts high quality printing. Used for protecting the PDF Document. The restriction is off by default.

**-pdf-security-user-password**

The user password. The document can be opened using either this password or the owner password. When the owner password is specified, the document can be opened by using the user password but with possible other restrictions.

**-pdf-security-owner-password**

The owner password (if this is provided when the document is opened in the reader application, with no restrictions).

**-pdf-ua**

Produces output that conforms to the PDF/UA-1 accessibility standards. The conversion will fail if fonts are not embedded. See: [Accessibility (508 Compliance) on page 52](#).

**-pdf-version**

Use this parameter to specify the version of the produced PDF. It has no impact on the set of PDF features used by the engine, but may be used to signal a compatibility level to the PDF readers. The default is 1.5.

**-security-policy**

Specifies the Java security policy file (in URL or file path syntax). This policy file can be used to restrict the processor access to certain resources (e.g. files, network). For more details, see: [Security Policy Configuration on page 19](#).

**-security-resources-dir1**

**-security-resources-dir2**

**-security-resources-dir3**

**-security-resources-dir4**

**-security-resources-dir5**

Specifies additional directories (to the input folder) where resources (e.g. images, CSS stylesheets, etc.) are stored. These parameters are automatically expanded in the Java security policy file, with read access, when the -security-policy is specified. For more details, see: [Security Policy Configuration on page 19](#).

**-security-resources-host**

Specifies the host where resources (e.g. images, CSS stylesheets, etc.) are stored. This parameter is automatically expanded in the Java security policy file, with connect access, when the -security-policy was specified. For more details, see: [Security Policy Configuration on page 19](#).

**-security-workspace**
Specifies the directory where the temporary files and font cache is stored. This parameter is automatically expanded in the Java security policy file, with read and write access, when the `-security-policy` is specified. For more details, see: Security Policy Configuration (on page 19).

- `show-changes-and-comments-as-pdf-sticky-notes`
  When set to `yes` (default), the review elements are shown as PDF annotations. When set to `no`, the review elements are left in the document object model and can be styled using CSS rules.

- `show-changed-text-in-pdf-sticky-notes-content`
  When set to `yes` (default), the inserted and deleted text is shown in the PDF annotations. When set to `no`, only the `inserted` and `deleted` labels are shown in the PDF annotations.

- `show-image-map-area-numbers`
  When set to `yes`, a counter for each area from the image map will be displayed over the image, near the defined shape. The default is `no`.

- `show-image-map-area-shapes`
  When set to `yes`, each of the image map area shapes will be displayed with a translucent fill over the image. You can use this to debug your image maps. The default is `no`. For customizing the aspect of the shapes, see Image Maps (HTML) (on page 88).

- `show-pdf`
  Opens the created PDF file in the default application (Windows only).

- `use-css-for-embedded-svg`
  When set to `yes` (default), the CSS files specified by the `-css` parameter are also applied on embedded SVG elements. Allowed values are `yes` and `no`.

- `-v`
  Shows the version of the processor.

- `Xmx<NNN>m`
  Used to specify the maximum amount of memory that is available to the Oxygen PDF Chemistry process. For example, to allow the process to use 1GB of memory:

  ```
  -Xmx1024m
  ```

**Notes:**

- The default is 512MB.
- Larger memory settings (beyond 1GB) are permitted only when the Java Virtual Machine that runs Oxygen PDF Chemistry is 64 bits and there is enough physical memory accessible to the operating system.
Important:
If the JAVA_ARG_LINE environment variable is set, this parameter is ignored.

Generating PDF Output from DITA Content Using a Command Line

To process and publish DITA to PDF using CSS and Oxygen PDF Chemistry, follow this procedure:

1. Download and install the Oxygen Publishing Engine. This bundles a DITA Open Toolkit (http://www.dita-ot.org/download) with all the required plugins to generate PDF output, including the Oxygen PDF Chemistry processor.
2. Place the license key for the Oxygen Publishing Engine in a licensekey.txt file in the installation folder.
3. Run the publishing engine from a command line and make sure you specify the format to be pdf-css-html5 and the path to your main DITA map. Specify additional parameters (such as args.css) if you want to customize the output. See the documentation for more information about customizing DITA output and accepted parameters.
   For example:
   oxygen-publishing-engine-dir/bin/dita --input=path/to/my.ditamap --format=pdf-css-html5 --output=path/to/output/folder -Dargs.css=/path/to/my.css

Environment Variables

JAVA_HOME

Normally it is used by the java executable that is found in the PATH environment variable. By using this variable, you can specify a different Java Virtual Machine installation directory.

JAVA_ARG_LINE

You can specify a set of arguments for the Java Virtual Machine. The following example specifies the maximum and initial memory setting to be 300MB:

SET JAVA_ARG_LINE=-Xmx300m -Xms300m

Note:
Setting this environment variable disables the usage of the -Xmx (on page 14) parameter.

XML_CATALOG_FILES

The path to one or more XML catalogs, in URI or File syntax. You can separate the catalogs by using the ":" (colon) or ";" (semi-colon) symbols.

SET XML_CATALOG_FILES="file:/D:/catalogs/docbook.xml;/file:/D:/catalogs/other.xml;"
Your First Document

Prerequisite:
Make sure you followed the procedure in the Installing (on page 7) section.

The following procedure provides a simple example for using Oxygen PDF Chemistry to style an XML document:

1. Start with a simple XML document to test the installation. Save it as my_doc.xml.

```xml
<article>
  <title>Hello World!</title>
  <p>My first document.</p>
</article>
```

2. Create a CSS stylesheet in the same folder. Save it as my_style.css.

```css
article, title, p {
  display: block
}

title {
  font-size: larger;
  border-bottom: 1pt solid blue;
}
```

3. Open a command-line console, change directory to the folder that contains your samples, and invoke:

`chemistry -in my_doc.xml -css my_style.css -out my_doc.pdf`

4. Open the result in your PDF reader.

Debugging the CSS

If you notice that some of the CSS properties were not applied as expected, some of the tips offered in this topic might help you with the debugging process.

CAUTION:
Do not modify the built-in rules directly in the CSS files from the Oxygen XML Editor/Author installation. Instead, copy the rules to your own customization CSS.
Comparing the CSS Against the Input File

The first thing that you should try is to check the file structure of the input XML or HTML file and compare it to the CSS selectors to make sure they are written correctly against the document structure. If you still cannot identify the problem, then inspect how the styles are applied (you can try any of the methods listed below).

Inspecting the Applied Styles Using a Browser

The following procedure explains how to inspect the applied CSS styles using Chrome, but any modern browser can be used and the procedure for each of them is similar:

1. Open the XML or HTML file.
2. Open the Chrome Developer Tools by using \textbf{More Tools} > \textbf{Developer Tools} (or press \texttt{CTRL+SHIFT+I}).
3. Activate the \textbf{Rendering} pane by using \textbf{More Tools} > \textbf{Rendering} then select \texttt{print} from the Emulate CSS media section. This will activate the CSS selectors enclosed in @media print {...}:

\textbf{Note:}

This allows you to debug the styling of elements, the table of contents, and the index, but not the styles of the page margin boxes (headers, footers) or page breaks.

4. Right-click on the element you want to inspect and select the \textbf{Inspect} action. You will see the element (in the Elements pane) and the list of styles that are applied on it (in the Styles pane):
Tip:
Clicking any of the stylesheet links from the Styles pane opens the original CSS files in the Sources pane. Editing the rules in that pane results in a live preview of how the change will affect the output (these modifications will be lost on reload).

Inspecting the Applied Styles Using Oxygen XML Editor/Author
To inspect the applied CSS styles using Oxygen:

1. In Oxygen XML Editor/Author, open the XML or HTML file.
2. From the Styles toolbar, choose the + Print Ready entry. This will activate certain CSS selectors enclosed in @media print {...}.

Note:
This allows you to debug the styling of elements, the table of contents, and the index, but not the styles of the page margin boxes (headers, footers) or page breaks.
3. Right-click on the element you want to inspect and select the **Inspect Styles** action. The dedicated **CSS Inspector** view will be opened and it will show the applied CSS rules.

**Tip:**
With this file open in **Author** mode, it might be helpful to switch the **Tags Display Mode** to **Full Tags with Attributes**. You might be able to identify the selector you need to style without using the **CSS Inspector** view.

### Other Debugging Techniques

Here are some other debugging techniques you may find useful:

- Add background and border properties to the specific CSS rule. If they do not appear in the output, then there is a problem with the rule selector.
- Add the `!important` keyword to a property that is not applied, or make the selector more specific (by adding more parent selectors).
- Add the following fragment in your customization CSS to show how the elements are mapped to PDF:

```css
* {
    border: 1pt solid blue !important;
}

*:before(1000) {
    content: oxy_name() !important;
    color: orange;
}

*:before(900) {
    content: "[ class= '*' attr(class) ''] " !important;
    color: orange;
}
```

This will show the element name, its class attribute, and will paint a blue border around each of the elements in the output. It will not show the page margin boxes or some content elements that are hidden.

### Security Policy Configuration

When running the product on a system where you do not control the input (XML files, CSS), you must take some steps to ensure that the transformation process does not access files outside the allowed locations, and does not connect to other hosts. Follow this procedure:
1. Create a Java policy file. A sample Java policy file can be found in `config/chemistry.policy`. You can use this as it is or as a starting point to grant or revoke permissions. Follow the instructions from this file.

2. Specify the Java policy file location (in URL or file path syntax) using the `-security-policy` command-line parameter:

   `chemistry.bat -security-policy file:/some/path/to/chemistry.policy`

3. By default, the font cache file is stored in the home directory, while the temporary files are stored in the system temp folder. It is recommended to specify a workspace directory where these files are to be stored. The sample policy file automatically sets read and write permissions on this folder.

   `chemistry.bat \n    -security-policy file:/some/path/to/chemistry.policy \n    -security-workspace /path/to/dir`

4. If your CSS files, images, fonts, or other resources are stored in a different folder than the one that contains the input file, you need to indicate those folders.

   `chemistry.bat ... -security-resources-dir1 /path/to/resources \n    -security-resources-dir2 /other/path/to/resources`

5. If you access resources, from another server, you have to give access to connections to it (note that Google fonts servers are already added to the policy file).

   `chemistry.bat ... -security-resources-host my.font.and.css.server:80`

Related Information:

[https://docs.oracle.com/javase/7/docs/technotes/guides/security/PolicyFiles.html](https://docs.oracle.com/javase/7/docs/technotes/guides/security/PolicyFiles.html)
2. Styling for Print

This section contains information about styling various components in the printed output.

Associating a CSS to a Document

There are several ways to associate the CSS to your document:

- Specify it as a PI at the beginning of the XML document:
  ```xml-stylesheet type="text/css" href="my-style.css" ?>
  ```

- Specify it in the meta section of the HTML document:
  ```HTML
  <link rel="stylesheet" type="text/css" href="my-style.css" />
  ```

- Specify it from the command line using the `-css` argument:
  ```
  chemistry -in my-doc.xml -css my-style.css -out my-doc.pdf
  ```

Note:
The rules from the CSS file indicated using the `-css` command-line argument take precedence over the ones from CSS files referred from the input document.

Related Information:
- Command Line Interface (on page 8)
- Mozilla MDN Web Docs: HTML Element Reference

Page Formatting

This section contains information about how you can use the new concepts from the CSS media paged module, such as @page rules, page margin boxes, and more.

The @Page Rule

Setting the Page Size

If you do not set the size, Oxygen PDF Chemistry uses portrait US-LETTER with margins of 1in.

This is a basic choice you will have to make for your output. You can use the @page rule like this:
The page sizes can be selected by name, for example, A4, A3, US-LEGAL (these are case insensitive), or if you are planning to print on a non-standard sheet, you can specify the width and height:

```css
@page {
    size: 5in 7in;
}
```

You can also specify the page orientation using the `portrait` or `landscape` keywords.

```css
@page {
    size: A3 landscape;
}
```

You can use the `@page` rule multiple times and the properties will merge as in ordinary CSS element styling rules.

### Using Page Selectors to Style the Blank, Left, Right, First, and Last Pages

There are cases where you need to have different page settings depending on the position of the page in the printed material.

```css
@page :left {
    border-right: 0.5in solid yellow;
}
```

```css
@page :right {
    border-left: 0.5in solid yellow;
}
```

The `:first` and `:last` selectors are used to style the first and last page from a sequence of pages.

```css
@page :first,:last {
    border-top: 5pt solid yellow;
}
```

The `:blank` selector is used to style the blank pages that appear as a result of forced page breaks.

### Using Named Pages

In the examples above, the default page settings were changed.

Suppose that you need to put a particular `<table>` element on a landscape page. The example below defines a named page `table-landscape-page` that switches the orientation to landscape. Then, to maximize the space available to the table, the margin is reduced for this page. Also, the A4 size defined by the default `@page` rule is inherited by the named page.
Also, Oxygen PDF Chemistry allows you to break a sequence of elements that have the same page in different page groups. A simple example is a set of chapter elements where you need to apply specific formatting for the first page from each chapter.

```
<chapter/>...<chapter/>...<chapter/>...<chapter/>
```

The CSS would be:

```css
@chapter {  
  ...
}

@chapter:first {  
  background-color: yellow;
}

chapter {  
  page: chapter;
}
```

But this is not enough. According to the W3C specification, all chapters that have the same page will be merged in a single page group (sequence), and the first selector will apply only on the first page from the first chapter. Therefore, you need to use the `-oxy-page-group` property on each of the chapter elements.

```css
@chapter {  
  ...
}

@chapter:first {  
  background-color: yellow;
}
```
The accepted values for the `-oxy-page-group` property are:

- **start** - Forces the creation of a new page group (sequence) even if the page before the element has the same name. The W3C specification algorithm ([https://www.w3.org/TR/css3-page/#using-named-pages](https://www.w3.org/TR/css3-page/#using-named-pages)) would merge the current element with the open page sequence.
- **auto** - Uses the W3C algorithm normally.

## Setting Page Margins

The same margin model applies to the page as for any element. The content is surrounded by padding, then a border, then margins. The page margins are between the physical end of the page and the text (if no border or padding is defined).

For example, to create a default page with margins of two inches:

```css
@page {
    margin: 2in;
}
```

Or you can specify them for all the sides:

```css
@page {
    margin-left: 2in;
    margin-right: 1in;
    margin-top: 0.5in;
}
```

Or you may want the inner margins of the pages of a book to be larger:

```css
@page {
    margin: 0.5in;
}
```

```css
@page :left {
    margin-right: 1in;
}
```

```css
@page :right {
    margin-left: 1in;
}
```
If no margins are specified in the CSS, the page margins are set at 1in.

**Page Margin Boxes**

The CSS specification defines numerous rectangular areas placed in the margins that surround the content of the page. These are called *margin boxes* and may be used to display static CSS-generated content (such as page numbers, publication titles, or other artwork).

```css
@top-left {
  content: url('company-logo.png');
  background-color:red;
}
```

The following table shows each of the 16 margin boxes (taken from the CSS specification: [https://www.w3.org/TR/css3-page/#margin-boxes](https://www.w3.org/TR/css3-page/#margin-boxes)):

<table>
<thead>
<tr>
<th>Box</th>
<th>Description</th>
<th>Placement</th>
</tr>
</thead>
<tbody>
<tr>
<td>top-left-corner</td>
<td>A fixed-size box defined by the intersection of the top and left margins of the page box.</td>
<td><img src="image1" alt="Diagram" /></td>
</tr>
<tr>
<td>top-left</td>
<td>A variable or fixed-width box that fills the top page margin between the top-left-corner and top-center page margin boxes.</td>
<td><img src="image2" alt="Diagram" /></td>
</tr>
<tr>
<td>top-center</td>
<td>A variable or fixed-width box centered horizontally between the page's left and right border edges and fills the page top margin between the top-left and top-right page margin boxes.</td>
<td><img src="image3" alt="Diagram" /></td>
</tr>
<tr>
<td>top-right</td>
<td>A variable or fixed-width box that fills the top page margin between the top-center and top-right-corner page margin boxes.</td>
<td><img src="image4" alt="Diagram" /></td>
</tr>
<tr>
<td>top-right-corner</td>
<td>A fixed-size box defined by the intersection of the top and right margins of the page box.</td>
<td><img src="image5" alt="Diagram" /></td>
</tr>
</tbody>
</table>
Table 1. Page-Margin Box Definitions (continued)

<table>
<thead>
<tr>
<th>Box</th>
<th>Description</th>
<th>Placement</th>
</tr>
</thead>
<tbody>
<tr>
<td>left-top</td>
<td>A variable-height box that fills the left page margin between the top-left-corner and left-middle page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>left-middle</td>
<td>A variable-height box centered vertically between the page's top and bottom border edges and fills the left page margin between the left-top and left-bottom page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>left-bottom</td>
<td>A variable-height box that fills the left page margin between the left-middle and bottom-left-corner page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>right-top</td>
<td>A variable-height box that fills the right page margin between the top-right-corner and right-middle page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>right-middle</td>
<td>A variable-height box centered vertically between the page's top and bottom border edges and fills the right page margin between the right-top and right-bottom page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>right-bottom</td>
<td>A variable-height box that fills the right page margin between the right-middle and bottom-right-corner page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>bottom-left-corner</td>
<td>A fixed-size box defined by the intersection of the bottom and left margins of the page box.</td>
<td></td>
</tr>
<tr>
<td>bottom-left</td>
<td>A variable or fixed-width box that fills the bottom page margin between the bottom-left-corner and bottom-center page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>bottom-center</td>
<td>A variable or fixed-width box centered horizontally between the page's left and right border edges and fills the bottom page margin between the bottom-left and bottom-right page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>bottom-right</td>
<td>A variable or fixed-width box that fills the bottom page margin between the bottom-center and bottom-right-corner page margin boxes.</td>
<td></td>
</tr>
<tr>
<td>bottom-right-corner</td>
<td>A fixed-size box defined by the intersection of the bottom and right margins of the page box.</td>
<td></td>
</tr>
</tbody>
</table>
Notes:

- The rules for variable-width at-rules (top or bottom) are the following:
  - When one box is defined, it takes the full available width.
  - When left and right boxes are defined, they take half of the available width.
  - When all 3 boxes are defined, each of them is one-third of the available width.
- The width property values must be set for all 3 at-rules when using fixed values (regardless if those values are relative or absolute).
- The initial and unset property values are not supported when applied to page margin boxes.

Columns

If you need to spread content in multiple columns, use the two CSS properties: `column-count` and `column-gap`.

Suppose that you have an HTML section that will be shown in a two-column layout, and as a special constraint, you want the title to span on both columns.

```html
<section class='two-columns'>
  <h2>The section on two columns</h2>
  <p>The section content...</p>
</section>
```

You can define a page with two columns:

```css
@page two-columns-page {
  column-count: 2;
  column-gap: 1in;
}
```

Then associate the section element with this page:

```css
section.two-columns {
  page: two-columns-page;
}
```

To make the title span the entire page width, use the `column-span` property:

```css
section.two-columns h2 {
  column-span: all;
}
```

Limitation:

You cannot use multiple column configurations on the same page. Oxygen PDF Chemistry only takes the `column-count` and `column-gap` properties into account if they are set on `@page` rules, not on elements from the content.
To control exactly the column breaking algorithm you can use the following extension properties: `-oxy-column-break-inside` *(on page 130)*, `-oxy-column-break-before` *(on page 130)*, `-oxy-column-break-after` *(on page 130)*.

For example, to eliminate the possibility where a heading `<h3>` element remains at the end of a column and the text that follows it moves to the next (a column break just after the heading) you can use:

```css
section.two-columns h3 {
    -oxy-column-break-after: avoid;
}
```

**Controlling Page Breaks**

If you want to start a new page with each chapter title, or if you want to avoid breaking a table onto two pages or avoid a break after a section title, you can use CSS properties to control the breaks.

CSS offers the following properties to control the page breaking process:

- `page-break-before`
- `page-break-after`
- `page-break-inside`

**Forcing Page Breaks**

To always move the content to the beginning of the page (for example, before the title of a section), you can use:

```css
section h1 {
    page-break-before: always;
}
```

Or if you only want a specific element to appear on a single page:

```css
div.notice {
    page-break-before: always;
    page-break-after: always;
}
```

**Avoiding Page Breaks**

Sometimes page breaks should be avoided. For instance, to avoid breaking between a title and the subsequent content:

```css
h1, h2, h3, h4, h5, h6 {
    page-break-after: avoid;
}
```

Or if you want to avoid breaks inside tables and lists:
Page Breaks Between Named Pages

A page break will be created each time there is a change in the `page` property associated to the elements in a sequence. Suppose that you have a sequence of `<div>` elements, one of them associated with a "cover" page, and others with a "chapter" page:

```xml
<div class="cover"> Welcome to the User Guide... </div>
<div class="chapter"> Here are the details... </div>
<div class="chapter"> Here are some more details... </div>
```

In this example, there will be a forced page break between the first `<div>` (associated to the "cover" page) and the second because of the page change.

The next two `<div>` elements are not separated by page breaks because they have the same `page` name ("chapter") and they are grouped in the same page sequence. If you want to style the first page from that sequence in a different way, the selector:

```css
@page content:first{
    background-color: yellow;
}
```

will apply to the first page from the first "chapter" `<div>`.

Suppose you want each of the chapters to start a new page sequence, with the first page colored in yellow. To do so, you must declare a new sequence start on the `<div>` element. This can be done using the extension property `-oxy-page-group`:

```css
div.chapter {
    page: chapter;
    -oxy-page-group:start;
}
```
Page Breaks Between Lines: Widows and Orphans

There are cases where a page break is placed between paragraph lines. Typically, for aesthetic purposes, the first paragraph should have more than one line. Another constraint might be the number of lines that are moved to the next page and again, it should avoid leaving a single line. CSS defines two properties for this type of control:

- `orphans` - This property specifies the minimum number of line boxes that should be left in a paragraph before the page break occurs.
- `widows` - This property specifies the minimum number of line boxes of a block container that must be left in a paragraph created on the next page after a break.

The following example shows how to keep the paragraphs at least four lines on the page before the break, and two lines on the page following the break:

```css
p {
  orphans: 4;
  widows: 2;
}
```

Note:
As a difference from the W3C standard, the `widows` and `orphans` CSS properties are applied to lists as well (the default is 2). This means that a list that spans consecutive pages will have either zero or at least 2 lines on each of the pages.

Chapter Page Placement and Styling

Oxygen PDF Chemistry provides specialized support to help you customize how chapters are handled in the output.

How to Start Chapters on an Odd or Even Page

A common use case is to arrange the chapters of the publication to start on an odd page number so that in the printed output, the chapter starts on the right side of the book.

By default, pages that have the same name are merged together in a single page sequence. If you want each of the chapters to have its own sequence, you can use the `oxy-page-group:start` CSS property to create separate page sequences. This allows you to control the start page of each of the chapters.

```css
.chapter {
  page: chapter-page;
}
```
To specify which page the chapter should start on, you can use another CSS property, the `-oxy-initial-page-number` property (on page 132):

```css
@page chapter-page {
  -oxy-initial-page-number: auto-odd;
}
```

Supported values for `-oxy-initial-page-number` are: `auto`, `auto-even`, `auto-odd`, `<number>`.

Moving the chapter on a specific page number may create blank padding pages at the end of the previous page sequence. If you want to style those blank pages, use the `:blank` page selector:

```css
@page :blank{
  @top-center{content:"Intentionally left blank"};
}
```

**How to End Chapters on an Odd or Even Page**

Another use case is to specify how a page sequence should end. If the sequence should have a total number of odd or even pages (or if it should end on an even or odd page).

Suppose you have a table of contents that follows the cover page and you need to want to end up with an even number of pages.

```html
<div class='toc'>Table of Contents..

.toc {
  page: toc-page;
}
```

Now you can use the `-oxy-force-page-count` property (on page 130) with an `even` value:

```css
@page toc-page {
  -oxy-force-page-count: even;
}
```

Supported values for `-oxy-force-page-count` are: `even`, `odd`, `end-on-even`, `end-on-odd`, `auto`, `no-force`.

**How to Style the First Page of a Chapter**

You can use the `:first` page rule selector to control how the first page of a chapter will look. For example, suppose you do not want the header to appear on the chapter first page. Your CSS might look like this:

```css
@page {
  @top-right-corner{content: counter(page)}
  @top-left{content: 'My Publication'}
}
```
Headers and Footers

Most printed publications use page headers and footers for almost all of their pages. To define them, you will use the page margin boxes.

You can set content and style to multiple page margin boxes. You should place them in the same parent `@page` rule. In the example above, the text is displayed only in the `top-center` of the first page, due to the `:first` page selector.

Extracting Text from Document Using String Sets

To display the title of the publication or the title of the current chapter, you will need to extract some content from the document and use it in one or more page margin boxes. This is possible by using a `string-set` property. It is similar to a variable that is initialized to content each time a specific element is matched.

In the following example, the text content of the `<H1>` element is extracted and used as a publication title and the `<H2>` element defines the chapter title:
Important:
To define multiple string sets for an element, use a single string-set property with a list of comma-separated definitions:

```css
h1 {
  string-set: publication_title content(), publication_author attr(data-author);
}
```

The following example uses the collected strings in the top margins of the pages. It joins the publication title and the chapter title by a "/" character, then places them in the outer side of the pages (to the left for the left-side pages, to the right for the right-side pages).

```css
@page :left {
  @top-left {
    content: string(publication_title) " / " string(chapter_title);
  }
}

@page :right {
  @top-right {
    content: string(publication_title) " / " string(chapter_title);
  }
}
```

A string set may contain static text, content from the document, attributes from the element, or counters. This is a more complex example, where a chapter number is added to the chapter_title string set:

```css
h1 {
  counter-reset: chapter;
}

h2 {
  string-set: chapter_title "Chapter (" counter(chapter) ")" content();
  counter-increment: chapter;
}
```

Related Information:
How to Use Dynamic Images in Headers and Footers (on page 38)

Extracting Text from Document Using XPath

You can use the oxy_xpath CSS extension function to execute an XPath expression over the input document. The main advantage is that you can execute any XPath processing, including:
- Document data retrieval
- Mathematical calculations
- If/then/else conditions

You can use `oxy_xpath` in values of all properties defined in a page rule.

**Important:**
This technique is not standard and is guaranteed to work only with this processor.

**Note:**
The XPath expression from the page rules is evaluated in the context of the document root element, so you will need to use absolute expressions starting with `/` or `//`. This is different from the case when the `oxy_xpath` is used in CSS rules that match an element. In this case, the XPath expressions are evaluated in the context of the matched element and you can use relative paths.

**Tip:**
XPath 2.0 is supported (not schema aware).

Suppose your document defines a creation date in a metadata section. This section may be anywhere in your document. To place the creation date in the center of the publication header:

```css
@page {
  @top-center {
    content: "Created: " oxy_xpath("//div[@class='created']/text()");
  }
}
```

Another example is to use an image from the document in the publication header:

```css
@page {
  @top-center {
    content: url(oxy_xpath('//@img[@class='product-img']/@href'));
  }
}
```

If the URL returned by `oxy_xpath` is not absolute, it is considered to be relative to the CSS file. To obtain a URL relative to the XML document, you can use in the XPath expression functions, such as `resolve-uri` and `document-uri`:

```css
@page {
  @top-center {
    content: url(oxy_xpath(resolve-uri//@img[@class='product-img']/@href, document-uri(/))));
  }
}
```
Multiple Lines in Headers and Footers

Sometimes you need to format the text from the header (or any page margin box) on two or more lines.

For example, suppose you want to have the following notice in the footer:

Confidential Document.
Do not distribute it without written consent!

The solution is to use \a in the static content from your CSS. This is an escape representing the line feed character in ISO-10646 (U+000A). This character represents the generic notion of "newline" in CSS.

```css
@page {
  @bottom-center {
    content: "Confidential Document. \a Do not distribute it without written consent!";
  }
}
```

The Page Counter

Besides the CSS counters that can be set on elements (for numbering sections, lists, tables, etc.), the CSS paged media module defines two more counters:

- `page`
  This counter returns the number of the current page.

- `pages`
  The number of total pages from the publication.

These counters are automatically updated by the publishing processor and can be used from the page margin boxes.

```css
@page {
  @bottom-center {
    content: "Page: " counter(page);
  }
}
```

Or if you need to obtain "Page 4 of 100", you simply use:

```css
content: "Page: " counter(page) " of " counter(pages);
```

You can format the page counter with styles such as `decimal`, `roman`, `lower-roman`:

```css
@page table-of-contents {
  @top-right {
```
How to Use Flexible Layout in Headers and Footers

In typical usage, the sub-regions of the header (the page margin boxes \@top-left, \@top-center, \@top-right) and footer (\@bottom-left, \@bottom-center, \@bottom-right) are distinct regions. If you are specifying content for all of them, the content set in one does not use space from the others. Instead, it wraps lines in its own region.

This creates a sort of table with fixed equal-sized columns, with the text wrapping inside of them.

You may need to 'merge' the center and right sub-regions for the header so that the layout engine has more room for topic titles before it wraps the title to a new line.

The solution is to eliminate the center part from the header and footer (\@top-center and \@bottom-center), and move the content to one of the sides:

How to Style a Part of the Text from the Header

If you need to style a fragment of text (for example, a company slogan) with certain colors or font styles, you have several options:

• Use an SVG image as the background for a page margin box or for the entire page.
• Use the oxy_label constructor. This is a function that creates a text label with a set of styles.
You can combine the `oxy_label` with `oxy_xpath`, to extract and style a piece of text from the document:

```javascript
content: oxy_label(text, oxy_xpath("/some/xpath"), styles, "color:blue; ");
```

**Note:**
These functions work only with the Chemistry CSS processor.

**Note:**
You cannot use `string()` inside an `oxy_label()`. As a workaround, to apply styling on the dynamic text retrieved by `string()` function you can define some overall styles for the entire page margin box and then use the `oxy_label` to style differently the static text.

```javascript
@page {
  @top-right {
    color: red;
    content: oxy_label(text, "My Company", styles, "color:black")
    string(chaptertitle); /* This inherits the styling from @top-right*/
  }
}
```

- Use two adjacent page margin boxes, and style them differently:

```javascript
@page {
  @top-center {
    content: "First part";
    color: red;
    text-align:right;
  }
  @top-left {
    content: "- Second part";
    color: blue;
    text-align:left;
  }
}
```
How to Use Dynamic Images in Headers and Footers

It is possible to dynamically change the images from the publication header or footer depending on the section.

For example, you want to place an image that describes the current section. Assuming that the titles of the sections define some metadata attribute pointing to some image file and the image references should be absolute URIs or relative to the input document.

```
<h2 -data-header-image="installation.png">
...
</h2>
```

The CSS should define a `string-set` that extracts the attribute value and builds an image for it using the `attr(.., url)` or `url` function.

```
@page {
  margin: 1in;
  @top-right-corner {
    content: string(str);
    font-size: 0; /* Get rid of ascent and descent to avoid spaces around the image */
  }
}

h2 {
  string-set: str attr(-data-header-image, url); /* references relative to the input document */
}
```

Besides the `attr` function, you can add text:

```
...

  string-set: str "Section... " attr(-data-header-image, url);

...
```

How to Add a Link in Headers and Footers

**Method 1: Using an SVG Link Attribute**

It is possible to add a link inside the document header (or footer) by using the `<a>` element inside an SVG document. For example, suppose you have the following SVG document named `custom.svg`:
This creates an SVG link with *PDF Chemistry* displayed as its text (the content of the `<text>` element).

**Note:**
If you just want to add a link without text, you can define a rectangle that contains the link instead of text.

To display the link, you just need to set your SVG file as the content of one of the page margin boxes:

```css
@page {
  @top-left {
    content: url("custom.svg");
  }
}
```

**Method 2: Using the CSS -oxy-link Property**

It is also possible to add a link inside the document header (or footer) by using the `-oxy-link` property on the `@page` margin box declaration. The entire page margin box will behave as a link and will be clickable.

```css
@page {
  @top-left {
    content: "Link";
    -oxy-link: "https://www.oxygenxml.com/";
    color: blue;
  }
}
```

**Footnotes**

Footnotes are especially important in books with a lot of references and annotations. To mark an element as being a footnote, you should use the `float: footnote` property.

Suppose that you have the following document fragment:

```html
<p>Changing the oil <span class="fnote">This should be done by a specialist in a controlled environment</span> in your car.</p>
```

To mark the `span` with the class `footnote` as an actual one:
There is a counter named `footnote` that is incremented automatically by the formatting processor each time it encounters an element with `float: footnote` set on it. Sometimes it makes sense to reset this counter at each of the chapters or sections of the document.

```css
section {
  counter-reset: footnote 1;
}
```

### Styling the Footnote Call

The number that remains in the content is called a footnote call. To style it, use the `footnote-call` pseudo-element:

```css
span.fnote:footnote-call { 
  color: red;
}
```

**Note:**

By default, Oxygen PDF Chemistry considers all the `:footnote-call` pseudo-elements to have the content to be the value of the footnote counter, a smaller font size, and to be aligned at top of the line.

You can change these properties if you need something different:

```css
:footnote-call { 
  content: counter(footnote);
  vertical-align: super;
  font-size: 0.8em;
}
```

### Styling the Footnote Marker

The footnote marker is the number that is shown at the left side of the footnote text, in the lower part of the page. It has the same text as the footnote call. To style it, you can use the `footnote-marker` pseudo-element:

```css
span.fnote:footnote-marker { 
  font-weight: bold;
}
```

**Note:**

Oxygen PDF Chemistry considers all the `:footnote-marker` pseudo-elements to have the content to be the value of the footnote counter, followed by a dot, and to be aligned to the right, towards the footnote text. You can change these properties if you need something different:
How to Add a Separator Above the Footnotes

The \@footnote part of a \@page declaration controls the style of the separator between the page content and the footnotes. For the content, you should set a leader. The leader uses a letter or a line style to fill the entire width of the page.

\@page {
  margin: 0.5in;
  ...
  \@footnote {
    content: leader(solid);
    color: silver;
  }
}

To create a dotted line, you can use the dot character: leader(\'\'.\'). Other commonly used characters are: "." (dash) and ":" (underscore).

Cross References

Technical documentation excels in cross references between sections and links to external resources. The end-user must be able to follow these links both in printed form, and in on-screen PDF rendering software.

Internal links

For printed material, a cross reference cannot just be a simple link (although PDF renderers support them for on-screen display). It should also display the page number of the target. In CSS you can do this by using the target-counter function.

For example, to get:

For details see [Installation on page 34].

from:

&lt;p&gt; For details see &lt;a href="#installation">Installation</a&gt;.&lt;/p&gt;

you can use a static content that is shown after the text from the link, consisting of a fixed string " on page " and the number of the page of the element referred by the @href attribute:
The `target-counter` function may be used together with the `leader` function to create table of contents. See: "Creating a Table of Contents (TOC) (on page 43)")

The processor supports both `target-counter` and `target-counters` functions, on page or other counters associated to your document elements. For example, you can use the `target-counter` to fetch the number of the chapter that contains the target:

```html
<div class="chapter" id="intro">
  ... For details see the chapter: <a href="#install" class="number">2</a>.
</div>

<div class="chapter" id="install">
  ...
</div>
```

The text should render like:

For details see the chapter 2.

You can use the CSS:

```css
:root { 
  counter-reset: chapter;
}
div.chapter{
  counter-increment: chapter;
}
a.number {
  content: target-counter(attr(href), chapter, decimal);
  oxy-link: attr(href);
}
```

**Related Information:**

W3 CSS Generated Content for Paged Media Module: The 'target-counter' function

**External links**

Usually, when linking to resources outside the documentation, normal web links are used.

There are two aspects to take in consideration when styling them:

- When printed on paper, show the entire URL so that the user can see it and type it in a browser.
- When displayed in a PDF reader, mark it as a link so that the user can click on it.
For example:

```html
<p>This is a link to the <a href='http://www.w3.org/'>W3C</a> website. </p>
```

To fulfill both conditions, you can add text after the "W3C" text, with the entire value of the `@href` attribute, and use the `link` or `--oxy-link` property to mark the generated content as being a link:

```html
a:after {
  content: "(" attr(href) ")";
  link: attr(href);
}
```

### Creating a Table of Contents (TOC)

A TOC (table of contents) is a special page that contains links to the chapters and sections of your book. Each line contains:

- The title of the chapter or section.
- A line of dots or other decoration, called a *leader*.
- The page number of the target chapter/section.

It may look like this:

```
Installing the Software ..........................10
  On Windows ...................................12
  On Mac........................................17
```

There should be some structure in your document that reflects the tree of the TOC, with ID links.

```
<ul class="toc">
  <li><a href="#introduction">Introduction</a></li>
  <li><a href="#installing">Installing the Software</a></li>
  <li><ul>
      <li><a href="#installing_win">On Windows</a></li>
    </ul>
  </li>
  <li><ul>
      <li><a href="#installing_mac">On Mac</a></li>
    </ul>
  </li>
</ul>
```

You can use the same `target-counter` function as for Cross References (on page 41), but suppose that you want to create a special page for the TOC:

```
@page toc {
  @top-center {
    content: "Table of Contents";
  }
}
```
This page places the “Table of Contents” text in the header of the page hosting the TOC and puts the number of the TOC page in the footer, with lower roman digits.

The following example associates the defined page to the `<ul>` element that gives the structure:

```css
ul.toc {
    page: toc;
}
```

To style the TOC entries, this next snippet removes the bullet decoration from the `<li>` elements, then marks the `<a>` element as being a link (the name for each TOC entry is defined inside an `<a>` element).

```css
ul.toc li {
    list-style-type: none;
}
ul.toc a {
    display: block; /* Only necessary when using a leader */
    link: attr(href);
    text-decoration: none;
}
```

**Note:**

When using a *leader*, the alignment for each TOC entry is normally *justified*. The `display: block` property is used to treat the contents of the `<a>` element as a separate block with a different alignment (i.e. *Align Left*).

After the name for each TOC entry (the content defined inside the `<a>` elements), a *leader* is used to expand to the available width. You can specify a character pattern for the leader:

- **dotted** - Creates an area filled with dots.
- **solid** - Creates an area filled with a dash.
- **space** - Creates an area filled with spaces.

Also, it uses the page number of the target element (after the *leader*):

```css
ul.toc a:after{
    color:blue;
    content: leader(".") target-counter(attr(href), page);
    link: attr(href);
}
```
Annotations

You may create generic comments or mark a particular document fragment as being inserted, deleted, or highlighted. The rendering in each of these cases will be different.

Figure 1. Chemistry Annotations in Acrobat Reader

In the above image, commented sections are shown in yellow and deleted content is in red.

To create annotations in your PDF output, create a specific structure in your document. The child topics in this section contain information about how this structure could look like for XML documents and HTML documents.

If you want to use CSS rules to style the change elements as plain elements in the content, you can disable the annotation processing using the command-line parameter: `--annotations-for-change-tracking-and-comments`.

Comments and Tracked Changes - HTML Fragment

This section contains information about how each type of tracked change is structured in an HTML file.

Insertions

For an insertion type of tracked change, the structure that defines the insertion details is inside a `range` (<span class="oxy-range-start">oxy-range-start</span> to <span class="oxy-range-end">oxy-range-end</span>), the inserted text is highlighted by a `<span>` element with the class `oxy-insert-hl`, and the details are stored in a `<span>` element with the `oxy-insert` class.

```html
<span class="oxy-range-start" id="sc_1" hr_id="1"></span>

  <span class="oxy-insert" href="#sc_1" hr_id="1">This is an insert!!</span>
</span>
```

```html
  <span class="oxy-insert" href="#sc_1" hr_id="1">This is an insert!!</span>
</span>
```

```html
  <span class="oxy-insert-hl">This is an insert!!</span>
</span>
```
Comments

Similar to insertions, comments are defined in a range (oxy-range-start to oxy-range-end), the comment details in an element with the class oxy-comment, and the highlighted content is wrapped in the oxy-comment-hl element.

Note:
Comments that are marked as done have a flag="done" attribute:

Attribute changes

The attribute changes are more complex. The range is empty, and is directly above the affected element (the one that has modified attributes). The element with the class oxy-attributes contains details about multiple attribute changes, each stored in an element with the class oxy-attribute-change.
Deletions

For a deletion, there are some elements that define the start and end of the deletion, and the highlighted text is wrapped in an element with the class `oxy-delete-hl`.

```html
<span class="oxy-range-start" id="sc_2" hr_id="2"/>
<span class="oxy-delete-hl">This is a deleted text.</span>
<span class="oxy-range-end" hr_id="2"/>
```

There is a structure that offers details about the deletion change, using the element with the class `oxy-delete`. This is linked to the above deletion range by the same ID value:

```html
<span class="oxy-delete" href="#sc_2" hr_id="2">
  <span class="oxy-author">dan</span>
  <span class="oxy-content"><img href="../img/ex.gif"></span>
  <span class="oxy-date">2018/03/14</span>
  <span class="oxy-hour">11:38:06</span>
  <span class="oxy-tz">+02:00</span>
</span>
```

Colored Highlights

To show some text as highlighted with a background color:

```html
<span class="oxy-color-hl color="rgba(140,255,140,50)">Some colored text.</span>
```

PDF Output

You may have specific requirements for the PDF files you need to produce (such as the set of metadata, bookmarks, the level of accessibility, or the PDF format).

Bookmarks

PDF bookmarks provide an additional way of navigating, similar to a table of contents. The tree bookmark structure is intended to be used by the PDF readers, usually displayed in a side view. More often, the
bookmarks show the logical hierarchy of the book, with pointers to the chapters and section, similar to a TOC. Creating bookmarks has no effect on the printed material.

Oxygen PDF Chemistry can create PDF bookmarks by using the standard CSS properties: `bookmark-level`, `bookmark-label`, and `bookmark-state`.

For an HTML document, you can collect the titles from the heading elements text.

```html
h1, h2, h3, h4, h5, h6 {
  bookmark-label: content(text);
}
```

In the following example, the `:before` pseudo-element is concatenated. That prefixes each of the `h1` with the value of the chapter number, with the text from the element:

```html
body {
  counter-reset: chapter;
}
...

h1 {
  bookmark-label: content(before) " / " content(text);
}

h1:before{
  content: counter(chapter);
  counter-increment: chapter;
}
```

You can define the level (depth in the hierarchy) of the bookmarks. The deeper the section, the higher the level:

```html
h1 { bookmark-level: 1; }

h2 { bookmark-level: 2; }

h3 { bookmark-level: 3; }

h4 { bookmark-level: 4; }

h5 { bookmark-level: 5; }

h6 { bookmark-level: 6; }
```

Also, you can control if the bookmarks are shown expanded or collapsed in the bookmark view. By default, all bookmarks are open. To close all the nodes from the level 2, you can use:

```html
h2 {
  bookmark-state:closed;
}
```
Note:
In the built-in CSS that Oxygen PDF Chemistry uses for processing HTML, the bookmarks are already configured using the `bookmark-level` and `bookmark-label` properties. If you need to set the closed/open state, you should use the `bookmark-state` property in your custom CSS file.

Related Information:
W3C Working Draft: Generated Content for Paged Media Module: Bookmarks

Metadata

PDF files may contain metadata. Metadata provides additional information about a certain document, such as its title, author, organization, creation date, format, or copyright.

HTML defines the `meta` element for keeping track of information that describes your content. Most of this information should migrate to the PDF document properties. The property values may be either static (specified directly from the CSS) or dynamic (collected from the document) using the following functions:

- `content(text)`
- `attr()`
- `oxy_xpath()`

Predefined Meta Fields

Examples of common metadata:

- Publication title
- Author
- Keywords
- Short description
- Copyright information

Oxygen PDF Chemistry automatically extracts this information from HTML documents.

Suppose that you have the following arbitrary XML document:

```xml
<doc>
  <title>Publication title</title>
  <meta name='keywords' content='software, network'>
  <meta name='description' content='This is a publication about software products...'>
  <meta name='author' content='John, jo@mysite.example.com'>
  <meta name='copyright' content='Copyright My Company 2021'>
  ...
</doc>
```

You could use any of the following CSS selectors to extract the metadata:
-oxy-pdf-meta-title

It is used to extract the publication title. You can use it by matching the `<title>` element:

```
title {
  -oxy-pdf-meta-title: content(text);
}
```

If this CSS selector matches multiple elements, only the first element in the document order will be used to extract the title.

-oxy-pdf-meta-author

It is used to extract the publication author. You can use it by matching the `<meta>` element with the `name='author'` attribute:

```
meta[name='author'] {
  -oxy-pdf-meta-author: attr(content);
}
```

If this CSS selector matches multiple elements, only the first element in the document order will be used to extract the author.

-oxy-pdf-meta-description

It is used to extract the publication description. You can use it by matching the `<meta>` element with the `name='description'` or `name='subject'` attribute:

```
meta[name='description'],
meta[name='subject'] {
  -oxy-pdf-meta-description: attr(content);
}
```

If this CSS selector matches multiple elements, only the first element in the document order will be used to extract the description.

-xy-pdf-meta-keywords

It is used to extract the publication keywords. For example, you can use it by matching the `<meta>` element with the `name='keywords'` attribute. Its value should be a list of tokens, separated by commas:

```
meta[name='keywords'] {
  -oxy-pdf-meta-keywords: attr(content);
}
```

If this CSS selector matches multiple elements, only the first element in the document order will be used to extract the keywords.

-xy-pdf-meta-keyword
It is used to extract a single keyword. Individual keywords are accumulated from elements that match the CSS rule that uses this property and then concatenated into a single string. This single string is then set in the PDF 'keywords' section. For example, if you mark keywords in your HTML document with a span with a "kw" class, you can collect them all by using:

```css
span.kw {
    -oxy-pdf-meta-keyword: content(text);
}
```

- **oxy-pdf-meta-copyright**
- **oxy-pdf-meta-copyrighted**
- **oxy-pdf-meta-copyright-url**

These properties define the copyright metadata. Acrobat Reader Pro, for example, displays this in the Details tab of the File/Document Properties dialog box.

```css
meta[name='copyright'] {
    -oxy-pdf-meta-copyright: attr(content);
    -oxy-pdf-meta-copyrighted: copyrighted;
    -oxy-pdf-meta-copyright-url: "https://my.company/copyright-notice.html";
}
```

The `oxy-pdf-meta-copyright` property specifies the copyright text for its value, the `oxy-pdf-meta-copyrighted` property specifies whether or not the publication is copyrighted (accepts only copyrighted or public-domain for the value), and the `oxy-pdf-meta-copyright-url` property can be used to specify the location of an external copyright notice.

**Custom Meta Fields**

Metadata is not restricted to the above cases. You may have custom metadata fields. It is usually displayed in a tabular format (for example, in Acrobat Reader ™, it is in the Custom tab in the Properties dialog box).

- **oxy-pdf-meta-custom**

This property defines a list of pairs. Each pair contains the name and the value for the meta information field. The pairs must be separated by a comma: name1 value1, name2 value2

In the following example, all the HTML meta tags are dumped as custom meta fields in the PDF:

```css
meta {
    -oxy-pdf-meta-custom: attr(name) attr(content);
}
```

If you have a span that defines the document creation date somewhere in the document content, you can use:

```css
span.created {
    -oxy-pdf-meta-custom: "CreationDate" content(text);
}
```
In case of conflicts, when two or more elements trigger the setting of a meta field with the same name, only the first definition of a meta field will be used in the PDF output.

Named Destinations (Anchors)

The named destinations FO extension provides a way to link to a particular anchor within a PDF document.

Suppose your PDF output is published on a website and accessible at the URL http://my_site.com/files/my_document.pdf, and the original XML document has a `<section>` element with an `@id` attribute set to `installation`.

```xml
...  
<section id="installation">  
...  
</section>  
...  
```

To open it in the PDF reader exactly at that particular section (with the id value of `installation`), you can use the `#installation` anchor in the URL: http://my_site.com/files/my_document.pdf#installation.

Oxygen PDF Chemistry declares named destinations for any `@id` or `@xml:id` attributes from your input XML document. As an alternative, if you do not want to alter the ids in the document, the `@nd:nd-id` attribute can be used. In this case, make sure the `nd` prefix is bound to the `xmlns:nd="http://www.oxygenxml.com/css2fo/named-destinations"` namespace.

Accessibility (508 Compliance)

It is recommended that you make your PDF output accessible for people who are blind or visually impaired. Many government organizations require documents to be accessible.

PDF Accessibility Tagging

By default, Oxygen PDF Chemistry partially creates accessible PDF documents in the sense that most of the paragraphs, tables, lists, headers, and footers are tagged automatically for any XML vocabulary, and PDF readers use this information to present the content.

In addition, the default CSS files used by Oxygen PDF Chemistry to generate PDF based on HTML defines accessibility tags for headings (H1..H6), quotations (Q), sections (SECTION), and pre-formatted text (PRE).

However, this tagging just takes the element name into account. If your element has a different semantic, you can impose a different PDF accessibility tag by using the `-oxy-pdf-tag-type` extension. In the following example, a paragraph with the `note` class will be marked:

```css
p.note {  
  -oxy-pdf-tag-type: "Note";
}
```
Hints for Making Documents More Accessible

Hint 1: The title of the document must be marked using the metadata.

This is important for accessibility since it will allow the screen reader to identify the publication title. This is an example using the -oxy-pdf-meta-title extension:

```css
title {
    -oxy-pdf-meta-title: content(text);
}
```

Note:
The default CSS files for generating PDF based on HTML already contains this rule.

Hint 2: Specify the language on the root of your document.

For XML documents, use `xml:lang` on the root of your document. For HTML documents, use the `@lang` attribute.

Hint 3: Set alternate text on all images.

Oxygen PDF Chemistry supports the -oxy-alt-text extension that can be used to associate the alternate text.

The following is an example from the Oxygen PDF Chemistry default CSS for generating PDF based on HTML, where it maps the property to the value of the `@alt` attribute of the `<img>` tag:

```css
img {
    -oxy-alt-text: attr(alt);
}
```

For embedded SVG, Oxygen PDF Chemistry automatically uses the `<title>` element as the alternate text of the image.

For embedded MathML, Oxygen PDF Chemistry automatically uses the `@alttext` attribute as the alternate text of the equation.

Fully Accessible PDF (PDF/UA1)

To make the PDF fully accessible, you have to activate the PDF/UA-1 mode. PDF/UA documents meet the regulations set in Section 508. This mode has special requirements:
1. Activate the PDF UA-1 mode from the command line using the `-pdf-ua` parameter.

2. All the fonts must be embedded. If you are using one of the basic fonts (such as "Times", "Helvetica", etc.), make sure you explicitly define CSS font faces for them. For details, see: Font Embedding (on page 95).

```
Troubleshooting:
If you are using fonts other than the basic ones and still have problems embedding the basic default fonts, make sure all elements are styled using one of your fonts of choice. A catch all CSS rule might be helpful:

:root{
    font-family: Arial;
}

@page {
    @top-left {font-family: Arial }
    @top-right {font-family: Arial }
    @top-center {font-family: Arial }
    @top-left-corner {font-family: Arial }
    @top-right-corner {font-family: Arial }
    @bottom-left {font-family: Arial }
    @bottom-right {font-family: Arial }
    @bottom-center {font-family: Arial }
    @bottom-left-corner {font-family: Arial }
    @bottom-right-corner {font-family: Arial }
}
```

3. The title of the document must be marked using the metadata. This is important for accessibility since it will allow the screen reader to identify the publication title. This is important for accessibility since it will allow the screen reader to identify the publication title. This is an example using the `-oxy-pdf-meta-title` extension:

```
title {
    -oxy-pdf-meta-title: content(text);
}
```

Note: The default CSS files for generating PDF based on HTML already contains this rule.
Tools for Checking the Document Accessibility

- For smaller documents, this site might be helpful: http://www.access-for-all.ch/ch/pdf-werkstatt/pdf-accessibility-checker-pac.html

Related Information:

Metadata (on page 49)

Archiving

PDF/A is the ISO standard for PDF specialized in the archiving and long-term preservation of electronic documents. To use this mode, you must:

1. Set a PDF/A mode from the command line using the -pdf-a parameter with one of the following values:
   - PDF/A-1a
   - PDF/A-1b
   - PDF/A-2a
   - PDF/A-2b
   - PDF/A-2u
   - PDF/A-3a
   - PDF/A-3b
   - PDF/A-3u

2. Embed all of the fonts. If you use one of the basic fonts (such as "Times", "Helvetica", etc.), make sure you explicitly define CSS font faces for them. For details, see: Font Embedding (on page 95).

Hyphenation

The CSS hyphens property specifies how the words should be hyphenated when the paragraph text wraps on multiple lines.

The accepted values are:

  manual

Words are only hyphenated when there are characters inside the word that explicitly suggest hyphenation opportunities. Those characters are:

U+2010 (HYPHEN)

The "hard" hyphen character indicates a visible line break opportunity. The hyphen is always shown in the output.

U+00AD (SHY)
An invisible "soft" hyphen. This character is not rendered visibly. It marks a breaking place for the word if hyphenation is necessary. You can use \&lt;shy\> in XML or \&shy; in HTML.

**auto**

Words are hyphenated automatically according to an algorithm that is driven by a hyphenation dictionary. Also, Oxygen PDF Chemistry uses an aggressive technique to create hyphenation points at underscores, dots, and case changes. This is useful when your publication contains snippets of code (Java, JS). To disable this functionality, you can use the `-no-aggressive-hyphenation` parameter (on page 11).

**Note:**
The element, or one of its parents, must have a `lang` or `xml:lang` attribute present for the processor to identify the hyphenation dictionary. If this is missing, the manual hyphenation is performed (on page 55).

```html
<table lang="en">
  ...
</table>
```

**none**

Currently not supported by Oxygen PDF Chemistry and it falls back to manual (on page 55). If your document does not use the `HYPHEN` or `SHY` characters, no hyphenation is done.

**Example: Hyphenation**

To perform hyphenation on all paragraphs from an HTML document, you can use:

```css
p {
  hyphens: auto;
}
```

Usually, it is best to activate hyphenation for elements that are known to have a limited width (for example, on tables) where long words could bleed off the page:

```css
table {
  hyphens: auto;
}
```

**Hyphenation Dictionaries**

The Oxygen PDF Chemistry provides built-in hyphenation patterns for the following languages:

<table>
<thead>
<tr>
<th>Code</th>
<th>Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>da</td>
<td>Danish</td>
</tr>
</tbody>
</table>
The built-in hyphenation pattern license terms are listed in the XML files in the 
[CHEMISTRY_INSTALL_DIR]/config/hyph folder. Most of them comply with the \LaTeX\ distribution policy.

Installing New Hyphenation Dictionaries

Oxygen PDF Chemistry uses the \TeX\ hyphenation dictionaries converted to XML by the \OFFO\ project: https://sourceforge.net/projects/offo/.

The .xml files allow you to access the licensing terms and you can use them as a starting point to create customized dictionaries (see How to Alter a Hyphenation Dictionary (on page 58)).

The .hyp files are the compiled dictionaries that the Oxygen PDF Chemistry actually uses.

The hyphenation dictionaries are located in: [CHEMISTRY_INSTALL_DIR]/config/hyph.

One simple way to add more dictionaries:
1. Download and extract the `offo-hyphenation-compiled.zip` file. This file is a bundle of many dictionary files.
2. Copy the `fop-hyph.jar` file to the `[CHEMISTRY_INSTALL_DIR]/lib` directory.
3. If you just need a single dictionary, place the `.hyp` or `.xml` file extracted from the above jar in the `[CHEMISTRY_INSTALL_DIR]/config/hyph` directory, or in another directory and use the `-hyph-dir` parameter.

**How to Alter a Hyphenation Dictionary**

The hyphenation dictionaries are stored as XML files in the `[CHEMISTRY_INSTALL_DIR]/config/hyph` directory.

You can copy the dictionaries you need to change in another directory, then use the `-hyph-dir` parameter to refer them inside your transformation.

Each file is named with the language code and has the following structure:

```xml
<hyphenation-info>
  <hyphen-min before="2" after="3"/>

  <exceptions>
    o-mni-bus
    ...
  </exceptions>

  <patterns>
    préémi3nent.
    pröémi3nent.
    surémi3nent.
    ....
  </patterns>

</hyphenation-info>
```

To change the behavior of the hyphenation, you can modify either the patterns or the exceptions sections:

**exceptions**

Contains the list of words that are not processed using the patterns, each on a single line. Each of the words should indicate the hyphenation points using the hyphen ("-" character. If a word does not contain this character, it will not be hyphenated.

For example, `o-mni-bus` will match the `omnibus` word and will indicate two possible hyphenation points.
Note:
Compound words (i.e. e-mail) cannot be controlled by exception words.

patterns
Contains the list of patterns, each on a single line. A pattern is a word fragment, not a word. The numbers from the patterns indicate how desirable a hyphen is at that position.

For example, tran3s2act indicates that the possible hyphenation points are "tran-s-act" and the preferable point is the first one, having the higher score of "3".

How to Disable Hyphenation for a Word
To disable hyphenation for a specific word, there are several possible approaches:

• If the word is a compound (like "T-shirt") and you want to keep it on the same line, you have two options:

  Manual Approach
  Wrap the word in an inline element with the @outputclass attribute set. In the CSS, change its style to white-space:nowrap; For example:

  ...
  <ph outputclass="no-hyphenation">T-shirt</ph>...
  *[outputclass =~ "no-hyphenation"] {
  white-space: nowrap;
  }

  Automatic Approach
  A better alternative to this is to write an XSLT extension that matches the text nodes and performs automatic markup (to see an example, go to How to Wrap Words in Markup in the XSLT Extensions for PDF Transformations section). Then match the compound-word class the same as in the previous example:

  *[outputclass =~ "compound-word"] {
  white-space: nowrap;
  }

  Another Alternative
  In all the compound words from your documentation, replace the hyphen ("-"), with a non-breaking hyphen character U+2011 (or XML notation &#2011;).

  Then change the autocorrect settings to automatically replace the compound word with its equivalent. For example: "T-shirt" would be replaced with "T[\u2011]shirt".

• If the word is not a compound, you have two options:
  ◦ Use one of the approaches listed above.
  ◦ Alter the hyphenation dictionaries as explained in: How to Alter a Hyphenation Dictionary (on page 58).
How to Hide Hyphens

It is possible to hide hyphens for cases where they are not needed (for example, when hyphenation occurs in a section of code). To hide the hyphens, use the space character in the `-oxy-hyphenation-character` property:

```html
pre {
  -oxy-hyphenation-character: " ";
}
```

How to Force or Avoid Line Breaks at Hyphens

It is possible to force or avoid line breaks inside words with hyphens (U+2010). This can be useful, for example, inside tables that have product references if you want the display to remain on a single line (or to split it on multiple lines). To achieve this, you can use the `-oxy-break-line-at-hyphens` property:

The accepted values are:

- **auto**
  
  Words are hyphenated automatically according to an algorithm that is driven by a hyphenation dictionary. This can lead to line breaks at hyphens.

- **avoid**
  
  Words are still hyphenated automatically except no line break will occur on hyphens.

- **always**
  
  Words are still hyphenated automatically except line breaks will be forced on hyphens.

**Example:**

Suppose you have a products table like this:

```html
<table>
  <row>
    <cell>Product-1233-55-88</cell>
    <cell>120</cell>
  </row>
  <row>
    <cell>Product-1244-66-99</cell>
    <cell>112</cell>
  </row>
</table>
```

and the following rule in a CSS stylesheet:

```css
table {
  -oxy-break-line-at-hyphens: avoid;
}
```
In the output, the list of product references will be displayed in a single line. On the contrary, setting the property value to `always`, will force a break after each hyphen.

**Using XPath in CSS**

You can collect and process the document contents using XPath directly from CSS. The following example counts the words from a section and shows it in a static text after the section:

```css
section:after {  
  content: "Number of words: "  
  oxy_xpath("count(tokenize(normalize-space(string-join(text(), ' ')), ' '))");  
}
```

The following is an example of using the `oxy_xpath` function in a page rule property:

```css
@page {  
  @top-center{  
    content: oxy_xpath('/book/title');  
  }  
}
```

All of the standard XPath 2.0 functions are supported.

**Related Information:**

Oxygen XML Editor User Guide: `oxy_xpath()` Function

**Using the :before(n) and :after(n) CSS Pseudo-Elements**

Although not standard, this extension may be useful if you want to style sections by adding multiple levels of static content. To add static content to an element, you would normally use a `:before` or `:after` pseudo-element.

This example adds static text before the title ("Chapter 1", "Chapter 2", etc.):

```css
h1:before {  
  content: "Chapter " counter(chapter) ".";  
  color: blue;  
}
```

All of this is styled with the same color (blue in this example). Using standard CSS, it is impossible to style specific aspects of it (for example, just the chapter number with a larger font and with red). However, you can do it using multiple `before(n)` or `after(n)` pseudo-elements:

```css
h1:before(3) {  
  content: "Chapter ";  
  color: blue;  
}
```
Notes:

- The bigger the level, the more distant the pseudo-element is.
- Level 1 corresponds to normal :before or :after pseudo-elements.

Change Bars

*Change bars* are used to mark revised elements in the content. For example, they are useful for technical documentation to flag content that was added in a new version of the file.

Display Change Bars Using the ::changebar Pseudo-Element

Suppose you have the following document:

```html
<p>Once every 6000 kilometers or three months, you need to change the oil in your car. It will extend your car lifetime.<span class="cb">This should be done by a specialist in a controlled environment.</span></p>
```

To mark the `span` element as being a *change bar*, you can use the ::changebar pseudo-element:

```css
.cb::changebar {
  -oxy-changebar-offset: 1mm;
  -oxy-changebar-placement: start;
  -oxy-changebar-style: solid;
  -oxy-changebar-color: black;
  -oxy-changebar-width: 1pt;
}
```

Tip:

The *change bars* can be customized using the following properties:

- -oxy-changebar-offset *(on page 129)*
- -oxy-changebar-placement *(on page 129)*
- -oxy-changebar-style *(on page 129)*
Display Change Bars Using Start and End Markers

Suppose you have the following document:

```html
<p>Once every 6000 kilometers or three months, you need to change the oil in your car. It will extend your car lifetime. <change-bar-start color="blue">This should be done by a specialist in a controlled environment. <change-bar-end></p>
```

To display the change bar inside the paragraph, you can use the `display: -oxy-changebar-start` and `display: -oxy-changebar-end` properties on the `<change-bar-start>` and `<change-bar-end>` custom elements:

```css
change-bar-start {
    display: -oxy-changebar-start;
    -oxy-changebar-color: attr(color);
}
change-bar-end {
    display: -oxy-changebar-end;
}
```

Notes:

- All of the `-oxy-changebar` CSS properties support the `attr()`, `oxy_xpath()`, and `calc()` functions.
- If you do not want to create new elements to mark the starting and ending point of the change bars, you can use the `display` property on both the `::before` and `::after` pseudo-elements (you can also use `::changebar`).
3. Layout

This section includes topics that describe how to style elements as tables or lists, how to rotate content, and how to use inline blocks.

Tables

This section is of special interest if you are creating a CSS stylesheet for a custom XML document. For HTML, Oxygen PDF Chemistry already defines the needed selectors for the `<table>` element.

This is an example of a very simple table in an XML document:

```xml
<tbl>
  <caption>A table.</caption>
  <rw>
    <c rspan="2">Spans vertically</c>
    <c>Other cell</c>
    <c>Other cell</c>
  </rw>
  <rw>
    <c cspan="2">Other cell, spanning to the right</c>
  </rw>
</tbl>
```

Defining Rows and Cells

First, mark the `<tbl>` element as being a table:

```css
tbl {
  display: table;
}
```

Next, the rows and cells:

```css
rw {
  display: table-row;
}
c {
  display: table-cell;
}
```
Defining the Column and Row Span

The processor needs to know how the cells span multiple rows or columns. For this, there are two properties available: `table-column-span` and `table-row-span`.

```css
c[cspan] {
  table-column-span: attr(cspan, integer);
}
c[rspan] {
  table-row-span: attr(rspan, integer);
}
```

Defining the Table Caption

You can define table captions by using the `display:table-caption`. To change the position of the caption relative to the table grid, you need to use the `caption-side` property:

```css
caption {
  display: table-caption;
  caption-side: bottom;
}
```

Repeating Headers and Footers

Any element marked with either the `table-header-group` or `table-footer-group` property is automatically repeated when a table is split over a sequence of pages. It is required that they contain only elements with the `display` property set to `table-row`.

```html
<tbl>
  <head>
    <rw>
      <c> Name </c>
    </rw>
  </head>
  <body>
    <rw> ... </rw>
    <rw> ... </rw>
    <rw> ... </rw>
  </body>
</tbl>
```

The CSS:

```css
head {
  display: table-header-group;
}
rw {
```
Repeating Captions

By default, the captions are not repeated on all pages a table spans. To enable this, you should mark the elements with a table caption display as with the 
\texttt{-oxy-caption-repeat-on-next-pages} property:

\begin{verbatim}
caption {
  -oxy-caption-repeat-on-next-pages: yes;
}
\end{verbatim}

The allowed values are \texttt{yes} or \texttt{no} and it is inheritable.

If you want to indicate that the page is a continuation, you can associate a static text to the caption that appears only on subsequent pages. For example, you can use an \texttt{::after} element as in the following example:

\begin{verbatim}
caption::after(2) {
  display:inline;
  content: *(continued)*;
  -oxy-show-only-when-caption-repeated-on-next-pages: yes;
}
\end{verbatim}

This example uses an \texttt{::after} element with the value of 2 to avoid conflicts with normal \texttt{::after} static elements that you may already use in your CSS.

Column Width and Styles

Oxygen PDF Chemistry supports an automatic layout for tables. This means the allocation of column width is done automatically based on the content size. In the following example, the HTML table has an automatic layout (this is also the default):

\begin{verbatim}
table {
  table-layout:auto;
}
\end{verbatim}

\textbf{Tip:}

For large tables with long words that bleed out of the page, you can choose to hyphenate the words from the cells. If the table uses an automatic layout, you should specify a width (such as 100%). Otherwise, the table columns will just be wide enough to accommodate the smaller hyphenated words:

\begin{verbatim}
table {
  table-layout:auto;
  width: 100%;
}
\end{verbatim}
To switch to a fixed layout:

```html
table {
  table-layout: fixed;
}
```

If you use the fixed layout and you are not satisfied with equal column widths, for HTML, you should use the `@style` attribute on the `<col>` element:

```html
<table style="width:2in;">
  <colgroup>
    <col style="width: 40%; background-color:cyan"/>
    <col style="width: 60%"/>
  </colgroup>
  ...
</table>
```

For arbitrary XML, you should use the `table-column` value in a `display` property for the element that represents the column. Suppose you have the following XML:

```xml
<tbl>
  <colgr>
    <column wd="30%"/>
    <column wd="70%"/>
  </colgr>
  ...
</tbl>
```

The following CSS links the `@wd` attribute to the `width` property and defines a different background for the first column.

```css
tbl {
  display:table;
}
colgr {
  display: table-column-group;
}
column {
  display: table-column;
  width: attr(wd);
}
column:first-of-type {
  background-color: yellow;
}
```
Proportional column widths (such as the ones used in the CALS tables from DITA or DocBook) are supported, but only when they are set in a `width` attribute on the `<column>` element (the one with the `display` property set to `table-column`):

```html
...<column width="3*/">
<column width="7*/">
...```

### Rotating Tables

There are cases where you have large tables and you need to rotate them to make them fit on your page. For instance, the default page orientation is *portrait*, but if you have a wide table with lots of columns, it might bleed to the right of the page.

There are two ways of solving this:

- Associate a wider page (i.e. landscape) to the table that needs more space. The disadvantage is that the table will force a page break before and after it.

  ```css
  @page landscape-page-for-large-tables {
    size: A4 landscape;
  }

  tbl {
    display: table;
    page: landscape-page-for-large-tables;
  }
  ```

- Rotate the table using the `transform` CSS property. The table will not create page breaks, but is susceptible to bleeding if its height exceeds the page width.

  ```css
  tbl {
    display: table;
    width: 200pt;
    transform: rotate(90deg);
  }
  ```

  **Note:**
  The table needs to have `table-layout: fixed` and a `width`.

The page is now landscape and you probably also need to change the headers and footers to match this new orientation. One way of doing this is to move the header content from the `@top-left`, `@top-center`, and `@top-right` rules into page margin boxes from the right, and apply a `transform` property on them.
In the following example, there is static text in the top (header) for the normal pages that is either placed in the left or right side of the page, depending on the page position:

```latex
@page :left {
    @top-left {
        content: string(maptitle) string(parttitle) string(chaptertitle) " | " counter(page);
        font-size:8pt;
    }
    @bottom-left {
        /*
        */
    }
    @bottom-right {
        /*
        */
    }
}

@page :right {
    @top-right {
        content: string(maptitle) string(parttitle) string(chaptertitle) " | " counter(page);
        font-size:8pt;
    }
    @bottom-right {
        /*
        */
    }
    @bottom-left {
        /*
        */
    }
}
```

For the landscape page, you can move content to either the `right-bottom` or `right-top`, depending on the page position:

```latex
/*
 * Table orientation.
 */
@page table-landscape:right {
    size:landscape;

    @top-left{
        content: none;
    }
    @top-center{
        content: none;
    }
```
Rotating Cell Content

To rotate table cell content, you can use the `transform` CSS property. Remember that only the content of the cell is rotated, not the cell itself. The rotation angle is clockwise. In the following example, the cells marked with the class `rotate` on their side are rotated:

```css
.rotate {
  transform: rotate(270deg);
  font-style: bold;
  text-align: left;
}
```
The rotated cells are in the header of the table.

```html
<html xmlns="http://www.w3.org/1999/xhtml">
  <table>
    <tr>
      <th>Type</th>
      <th colspan="3">Features</th>
    </tr>
    <tr>
      <th>Sport</th>
      <th>All Terrain</th>
      <th>Family</th>
    </tr>
    <tr>
      <td>Dacia Duster</td>
      <td>X</td>
      <td>X</td>
    </tr>
    <tr>
      <td>Nissan Leaf</td>
      <td></td>
      <td>X</td>
    </tr>
    <tr>
      <td>Volvo XC90</td>
      <td>X</td>
      <td>X</td>
    </tr>
    <tr>
      <td>Toyota Celica</td>
      <td>X</td>
      <td>X</td>
    </tr>
  </table>
</html>
```

The result is:

<table>
<thead>
<tr>
<th>Type</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sport</td>
</tr>
<tr>
<td></td>
<td>All Terrain</td>
</tr>
<tr>
<td></td>
<td>Family</td>
</tr>
<tr>
<td>Dacia Duster</td>
<td>X</td>
</tr>
<tr>
<td>Nissan Leaf</td>
<td>X</td>
</tr>
<tr>
<td>Volvo XC90</td>
<td>X</td>
</tr>
<tr>
<td>Toyota Celica</td>
<td>X</td>
</tr>
</tbody>
</table>

**Lists**

For the HTML lists (<ol>, <ul>), Oxygen PDF Chemistry already defines the needed selectors, but sometimes you need greater control over the spacing or style of the marker.
The `list` element needs to have the `display` property set to `block`, and the children elements need to have the `display` property set to `list-item`.

```css
ul {
  display: block;
}

li {
  display: list-item;
  list-style-type: disc;
  margin-left: 0.5in;
}
```

Make sure you have a `margin-left` so that the bullet will have enough space to be painted inside the list item box.

**List Marker Position**

You can select whether the marker should be considered a decorator outside the box of the `list` element (this is the default) or if it should be inline, on the first line of the content of the element.

```css
li.inside {
  list-style-position: inside;
}
li.outside {
  list-style-position: outside;
}
```

**The `list-style-type` and `list-style-image` CSS Properties**

Oxygen PDF Chemistry supports the following values for the `list-style-type` property:

- box
- check
- circle
- diamond
- disc
- hyphen
- square
- none
- decimal
- lower-roman/lower-latin
- upper-roman/upper-latin
- decimal-leading-zero
- <string>
To use an image as a marker instead of a standard bullet or number, you can use the `list-style-image` property. You have to use the `url` function to point to an image resource:

```css
li {
  list-style-image: url("images/my_list_bullet.svg");
}
```

### Using a :marker Pseudo Selector

There is a CSS pseudo-element that allows you to associate styles with the `list` marker. The following example changes the background color, font, width, and even the content of a `list` marker:

```css
ol {
  ...
  counter-reset: cnt;
}
li:marker {
  width: 3em;
  background-color: silver;
  color: red;
  font-weight: bold;
  text-align: left;
  counter-increment: cnt;
  content: counter(cnt) " - \0430";
}
```

To change the marker symbol and its size:

```css
li:marker {
  /* Club Symbol */
  content: "\2663";
  font-size: 0.8em;
}
```

To use an image instead of a number as a marker:

```css
li:marker{
  content: url("images/my_list_bullet.svg");
}
```

You can even implement a custom list numbering using this selector. Such a technique may be useful for other list numbering schemes that are unique or currently not supported (such as lists lots of elements). You can use the `nth-of-type()` selector to choose the labels of each item, individually:

```css
li:nth-of-type(1):marker{
  content: "alpha";
}
```
Floats

Floats are not supported by Oxygen PDF Chemistry.

Aligning Blocks Horizontally

If you want to align just the text paragraphs from the block, you might find the text-align property useful.

To align a block, it must have a specified width (absolute or as a percentage).

Align Center

This can be done by setting both the margin-left and margin-right properties to auto:

```html
div {
    width: 200pt;
    margin-left: auto;
    margin-right: auto;
}
```

Align Left

This can be done by setting the margin-left to 0 and margin-right properties to auto:

```html
div {
    width: 200pt;
    margin-left: 0;
    margin-right: auto;
}
```

Align Right

This can be done by setting the margin-left to auto and the margin-right properties to 0:

```html
div {
    width: 200pt;
    margin-left: auto;
}```
Rotating Blocks

There are cases where you need to rotate some block elements for visual effects. For this, you should use the `transform` CSS property.

```css
code {
    transform: rotate(90deg); /*Clockwise*/
}
```

The block needs to have a specified width. If this is not specified, the built-in algorithm will try to compute a width and then rotate the block. For example, for rotating the text:

```
the quick brownish fox
```

The algorithm will try to break the line at each space (to wrap it as much as possible, so it gets a minimum width):

```
the
quick
brownish
fox
```

It will then use the maximum word size (the width of the `brownish` word). The result will be:

```
the quick brownish fox
```

If you want the entire phrase to be rotated (without wrapping), then you should specify that all the whitespaces must be preserved:

```css
code {
    transform: rotate(90deg); /*Clockwise*/
    white-space: pre;
}
```

The result will be:
4. Graphics

This section includes topics that describe how to style graphics in your PDF output.

Images

This topic contains information about how you can reference images from your HTML or XML documents.

For HTML, the `<img>` tag is recognized as an image without any other styling in your CSS files:

```html
...<p>And this is the picture of a happy face: <img src="happy.png" />.</p>...
```

For XML, you must add CSS rules that pick up the content of an attribute and use it as a source for the image:

```xml
...<para>And this is the picture of a happy face: <imagedata fileref="happy.png"/>.</para>...
```

The following example uses static content on the `imagedata :before` pseudo-element:

```css
imagedata[fileref]:before {
    content: attr(fileref, url);
}
```

Note:

It is important to use the `url` keyword when retrieving the attribute value. It signals that the value is a pointer to an external image.

Supported Image Types

Oxygen PDF Chemistry supports the following types of raster images:

- BMP (Microsoft Windows Bitmap)
- GIF (Graphics Interchange Format)
- JPEG (Joint Photographic Experts Group)
- PNG (Portable Network Graphic)
- TIFF (Tag Image Format File)

And the following types of vector images:
• SVG (Scalable Vector Graphics)
• PDF (PDF Documents)

**PDF Images**

You can reference PDF images the same way other image files are referenced:

```html
<img src="my_doc.pdf"/>
```

To point to a single page from your PDF document, use the following syntax (this example points to page 5):

```html
<img src="my_doc.pdf#page=5"/>
```

**Setting Image Width and Height**

The image size can be determined from the number of pixels of the image, taking the image resolution (on page 79) into account (if available). There are cases where this computed size is not what you need, and you want to specify the size explicitly.

For HTML, it is enough to use the image attributes directly in your document.

```html
<img src="my_image.png" width="300" height="250"/>
```

For an arbitrary XML, you can indicate the image width and height through a rule that matches the element (or its :before or :after pseudo-elements) and sets the width and height CSS properties.

```css
imagedata {
  display: inline;
  content: attr(src, url);
  width: attr(width, length);
  height: attr(height, length);
}
```

Or, if you use an image as a decorator, you can specify fixed dimensions in the CSS:

```css
chapter title:before {
  content: url("my_artwork.png");
  width: 300px;
}
```

**Note:**

For static content as in the example above, Oxygen PDF Chemistry tries to use the width and height set on the pseudo-element, then the ones that are set on the parent element, but only if the static content is composed of a single image. Mixing text and images in the content property disables the width and height specification.

If you want to limit the width of the images to a maximum size, you can use the max-width property. The image will be scaled down to fit the maximum size (if it is larger).
Some raster images (pixels, not vector) may have a default resolution, set by the designer, using an image-editing software. Usually, the image size and resolution are set to look best on the screen. The advantage of a resolution set in the image itself is that it will have the same effective size on the screen and on paper. For example, if the image has 144 dots in width, and an embedded resolution of 72dpi, it will be two inches on screen and on paper.

The problems start to arise when the resolution is not set on the image, and the PDF processor has to decide what resolution to use to determine the size of the graphic. To solve this, the processor extracts the DPI from:

- The `image-resolution` CSS property associated to the element that contains the image.
- The `-image-resolution` command-line parameter.
- The built-in fallback resolution of 96 DPI.

The recommended way to change this is by using the CSS Level 4 `image-resolution` property:

```css
img {
  image-resolution: 300dpi;
}
```

**Note:**

The `image-resolution` is inheritable, so you can associate it to the root element. It does not apply on the page generated content (margin boxes).

```css
:root {
  image-resolution: 300dpi;
}
```

To reset the image resolution to the one set in the image itself, you can use the constant `from-image` instead of a DPI value:

```css
title:before {
  content: url("chapter-decorator.png");
  image-resolution: from-image;
}
```
Tip:
To change the resolution for images that appear in a page margin box, set this property on that margin box, or directly on the @page rule, to apply it to all page margin boxes:

```css
@page front-page {
    image-resolution: 600dpi;
}

@page {
    @top-center {
        image-resolution: 600dpi;
        content: url("company-logo.png");
    }
}
```

Related Information:
Command Line Interface (on page 8)
https://drafts.csswg.org/css-images-4/#the-image-resolution

Background Images

You can use background images to impose a texture. You can use them to decorate an entire page, or a specific element from your document.

Supported properties:

- background-image
- background-repeat
- background-position

Page Background Images

You can set a background for a page. Usually, you do this for the cover page to impose a full-page artwork, or to add graphics to the header and footer of the page. Here is an example of how can you do it for the page:

```css
@page cover {
    size:A4;
    margin:1in;
    background-image:url("images/my_book_cover_artwork.png");
    background-repeat:no-repeat;
}

div.cover {
```
Note:
It is better to use SVG for the page artworks. It does not suffer from pixelation. If you are limited to using raster images, you can fine-tune their resolution by using the `image-resolution` property (on page 79).

Note:
To separate the header and footer from the main content using visual effects (lines, shadows, etc.), you can use a full page size artwork in SVG and set it to the default page:

```css
@page {
    background-image: url("images/my_header_footer_artwork.svg");
    background-repeat: no-repeat;
}
```

If your artwork is smaller, consider a "DRAFT" watermark (for instance). You should use the `background-position` to place it where you need:

```css
@page {
    ... 
    background-image: url("images/draft.svg");
    background-position: bottom center;
    ...
}
```

Element Background Images

You can style the background of your elements the same as for web pages:

```css
section {
    background-image: url("my_repeating_pattern.svg");
    background-repeat: repeat-y;
}
```

Foreground Images

You can use foreground images to impose a texture above entire pages (in the foreground).

Supported property: `-oxy-foreground-image`.

Usually, the foreground images contain watermarks (for example, "Draft", "Copyright", or "Do Not Copy"). This is an example of how can you do it for all the pages:
Notes:

- Make sure you use an image type that supports transparency.
- It is better to use SVG for the page artwork because they do not suffer from pixelation and because foreground images are stretched to the full page size.

Supported Colors

Along with the usual color values, Oxygen PDF Chemistry supports the following special values for all the color properties:

- **rgb** - Used to specify red, green, and blue components (for example, `color="rgb(255, 0, 153)"`).
- **rgba** - Used for transparent colors by specifying each color channel and the transparency. For example, the following would result in the background color being magenta since the red color from the parent `div` element will be visible through the blue color of the `p` element:

```html
div{
  background-color:rgb(255,0,0,0.3)
}
p{
  background-color:rgba(0,0,255,0.3)
}
```

For more information about color properties, see MDN Web Docs: Color.

SVG

Oxygen PDF Chemistry supports SVG images. The main advantage of using SVG is that the image looks good on paper no matter its size.

**SVG Referenced or Embedded in the Document**

These can either be referenced as external resources:

```html
<p> This is an SVG showing a happy face: <img src="happy.svg"/></p>
```

or embedded in the document as SVG fragments:

```html
<p> This is a red circle:
<svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 50 50" width="50" height="50">
```

```html```
The document styles are also applied to the SVG fragments. For instance, if the `<circle>` element has the `sun` class, you could change its appearance by using `.sun {fill:orange}` in your main CSS. As a general rule of thumb, keep distinct names for the SVG fragment class attributes from the ones used for general content styling.

Note:
For HTML5, the namespace declaration is not required.

Using SVG for Styling

To use SVG to decorate an element:

```html
div.note:before {
  content:url("images/note.svg");
}
```

To set an SVG image as the background of a page, or a page margin box:

```html
@page coverpage{
  background-image: url("images/clipart.svg");
  background-repeat:no-repeat;
  background-position:center center;

  @top-left {
    background-image: url("images/company.svg");
    background-repeat:no-repeat;
  }
}
```

Note:
The `image-resolution` CSS property does not apply for SVG vectors.

Linking from SVG to Parts of the Host Document

If you need to use the graphics as a type of table of contents, you can place links over parts of the image (used as callouts) that point to some descriptive sections in your document by using the `<a xlink:href=".."/>` markup. This is similar to what the `<imagemap>` HTML element does, but this is encoded directly in the graphics:

```html
<p>This image has a link that points to a paragraph:
<svg xmlns="http://www.w3.org/2000/svg"
```
SVG Templates

Oxygen PDF Chemistry supports SVG template images (with a .template.svg file extension). These files can contain XPath expressions that will be expanded by the processor.

Using SVG Template as a Cover Page

A good use-case for SVG templates is when you want to create a custom cover page. For example:

1. Suppose the following HTML file should be printed with a cover page:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>How to Dynamically add Content to SVG</title>
    <link rel="stylesheet" href="styles.css"/>
    <meta name="author" content="John Doe"/>
    <meta name="version" content="1.1"/>
    <meta name="copyright" content="© Oxygen PDF Chemistry. For demonstration purpose."/>
  </head>
  <body>
    ...
  </body>
</html>
```

2. The cover image (for example, named `cover.template.svg`) should display `<meta>` nodes information and the main title. Both the `author` and `version` will be displayed in rectangular shapes.
Oxygen PDF Chemistry 26.0 | 4 - Graphics | 85

prefixed with the displayed information ("Author:" and "Version:"), while the title is positioned in the middle of the page, at some absolute coordinates.

```xml
<svg width="8.5in" height="11in" xmlns="http://www.w3.org/2000/svg">
  <g>
    <rect height="39" width="180" y="40" x="579" stroke="#000" fill="#fff"/>
    <text font-size="8" stroke-width="0" y="52" x="582" stroke="#000" fill="#000000">
      Author:
    </text>
    <text font-size="16" stroke-width="0" y="66" x="643" stroke="#000" fill="#000000">
      ${//meta[@name='author']/@content}
    </text>
    <text font-size="8" stroke-width="0" y="91" x="582" stroke="#000" fill="#000000">
      Version:
    </text>
    <text font-size="16" stroke-width="0" y="104" x="659" stroke="#000" fill="#000000">
      ${//meta[@name='version']/@content}
    </text>
    <text font-size="30" stroke-width="0" y="252" x="123" stroke="#000" fill="#000000">
      ${//head/title}
    </text>
    <text font-size="12" stroke-width="0" y="993.40002" x="266" stroke="#000" fill="#000000">
      ${//meta[@name='copyright']/@content}
    </text>
  </g>
</svg>
```

**Notes:**
- XPath expressions are not expanded if the SVG template is open in **Author** mode.
- XPath expressions can be tested (without `${}`) using the XPath/XQuery Builder view.

**Important:**
- If you received the SVG image from someone else (e.g. a graphics designer), make sure that the text from the image was not converted to glyph shapes and that it is rendered using the `<text>` element.
- The SVG `<text>` element does not wrap the text if it overflows the image. If you have longer text that needs to be rendered, you might consider using multiple `<text>` elements and more evolved XPath expressions (for example, using the `substring()` function) to place the text on multiple lines.

**Tip:**
You can ask a designer to fill the image with some placeholders that you can later find and replace with your XPath expressions. In the above SVG, the designer could place the text Here comes the version, that you replace with `${//meta[@name='version']/@content}:`
3. The CSS stylesheet should declare a synthetic page to be printed before the HTML document and display the SVG cover image on this new page:

```css
@page cover-page {
  background-image: url('cover.template.svg');
  background-repeat: no-repeat;
}

html::before {
  content: " ";
  page: cover-page;
}
```

You can even create a sequence of synthetic pages using `before(1), before(2), ..., before(n)` pseudo-elements (on page 61) and multiple `@page` definitions for each of the `before` elements.
4. After the transformation, the final document cover will look like this:

How to Dynamically add Content to SVG

© Oxygen PDF Chemistry. For demonstration purpose.

MathML

MathML equations can either be referenced from the document as external resources (the file should end with the .mml extension):

<p>The quadratic formula: <img src="quadratic.mml"/></p>
The quadratic formula:

\[
\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
\]

Notes:

- No styling is required to show the embedded MathML in the output. It works automatically for HTML or XML document types.
- For HTML5, the namespace declaration is not required.

Image Maps (HTML)

The HTML `<map>` element is supported by the processor. It allows you to define a set of shapes over your original image and each shape behaves like a link pointing to a part of your publication or to an external resource.

1. Start by specifying the width and height of your image using attributes. The size and coordinates are in pixels. The size you define here is very important when you specify the coordinates of the shapes. You can use any CSS unit, including percents. The percents are solved relative to the image size and represent a way of creating "responsive" image maps (reusing the map on the same image with different sizes depending on the position in the document). If you are using the same customization CSS for HTML web output as well, make sure you use only pixels as some of the browsers do not support other units.
2. Create a map element and link the image to the map:

```html
<img src="engine-picture.png" width="400" height="300" usemap="mapID">
<map name="mapID">
  ...
</map>
```

3. In the map element, add areas, each with a shape and a set of coordinates:

```html
<area shape="rect" coords="100,70,120,90"
      href="#radiator" alt="Radiator" />
```

4. Verify how the shapes look in the output. You can make the shapes visible using one of these methods:
   - Using the `show-image-map-area-shapes` and `show-image-map-area-numbers` command-line arguments.
   - Adding a CSS snippet to your customization. The shapes have the `image-map-shape` class, the bullet around the image map number (image-map-number), and the text inside the bullet (image-map-number-text). To make them translucent yellow:

   ```css
   .image-map-shape{
     fill: yellow;
     fill-opacity: 0.5;
     stroke-opacity: 0.5;
   }
   .image-map-number-text {
     visibility: visible;
   }
   .image-map-number {
     fill: yellow;
     fill-opacity: 0.4;
     stroke-opacity: 0.7;
   }
   ```

Videos

This topic contains information about how you can reference videos from your HTML or XML documents.

⚠️ Warning:

- Only Acrobat Reader supports video embedding.
- Only the following extensions are supported: `.mov`, `.mp4` and `.m4v.`
Tip:
To process large video files, you need to increase the available memory (e.g. 3.5Gb of memory for a 2Gb video).

For HTML, the `<video>` tag can be used without any other styling in your CSS files:

```html
...  
<video data="path/to/video.mp4" format="mp4" width="480" height="270">  
  <source src="path/to/video.mp4"/>
</video>
...  
```

Note:
It is recommended to use absolute paths for `@data` and `@src` values like this: `src="file:/C:/absolute/path/to/video.mp4"`.

For XML, you must add CSS rules that pick up the content of an attribute and use it as a source for the video:

```html
...  
<para>Like shown in this video: <video src="files/earth.mp4"/></para>  
...  
```

For example, the CSS rules may look like this:

```css
video {
  display: inline;
}
video[src] {
  content: attr(src, url);
}
```

Note:
It is important to use the `url` keyword when retrieving the attribute value. It signals that the value is a pointer to an external image.

Change the Video Size

If your XML contains additional attributes such as `@width` or `@height`, you can complete the CSS:

```css
video[width] {
  width: attr(width, length);
}
video[height] {
  height: attr(height, length);
}
```
Notes:

• It is also possible to set both @width and @height attributes directly on the video `<object>`.
• If no values are set for width/height (neither from the CSS nor attributes), the extracted video size will be used.

Customize Video Cover Placeholder

By default, a placeholder is displayed in place of the video. When clicked, this placeholder will launch the video.

It is possible to change this placeholder with a custom one by using the `-oxy-video-cover (on page 128)` property:

```css
video {
  -oxy-video-cover: url("files/cover.png");
}
```
5. Fonts

This section contains information about using fonts to style the output, including the supported fonts, how to use Web fonts and other special fonts, and font embedding.

Supported Fonts

Oxygen PDF Chemistry supports TrueType fonts, either as local files or as web fonts, with the following limitations:

- Some fonts may use features that are unsupported, such as the `MarkGlyphSet` in the GDEF table.
- The advanced features of OTF are not supported, but the Compact Font Format (CFF) extracted from the OTF is embedded as a Type1C font. Some features such as ligatures and style sets are supported.
- Using a Tamil font might result in content that is not searchable or copyable.

Basic Fonts

By default, all fonts are automatically embedded in PDF documents by Oxygen PDF Chemistry. The exception to this is some fonts that are considered to be available in all PDF rendering applications. These are called basic fonts and include:

- `Times` (v3) (in regular, italic, bold, and bold italic)
- `Courier` (in regular, oblique, bold and bold oblique)
- `Helvetica` (v3) (in regular, oblique, bold and bold oblique)
- `Symbol`.
- `Zapf Dingbats`.

Fallback Fonts

Whenever a character is not covered by the font specified in the CSS, it can still be rendered by Oxygen PDF Chemistry fallback fonts. To use these fallback fonts, you need to add one of the three generic font families indicated below in your CSS.

Here is the list of fallback fonts that are automatically added for these generic font families:

- **Sans-Serif** - Segoe UI, Symbol, Segoe UI Symbol, Microsoft YaHei, Yu Gothic, Malgun Gothic, Latha, Arial Unicode MS
- **Serif** - Times, Times New Roman, Symbol, Segoe UI Symbol, SimSun, MingLiU, MS Mincho, Batang, Vijaya, Arial Unicode MS
- **Monospace** - Segoe UI, Symbol, Segoe UI Symbol, Microsoft YaHei, Yu Gothic, Malgun Gothic, Latha, Arial Unicode MS
If you are using a Linux machine, the list of fallback fonts is the following:


If you use the processor on multiple operating systems/machines, make sure you have a consistent list of fonts installed on each of them. For instance, if you are using both a Linux and a Windows machine to run transformations in parallel and the PDF produced on Linux has some missing characters (they are replaced with the # symbol), then you can fix this by inspecting the fonts listed in the Document Properties dialog box in the PDF reader and install them on the Linux machine (make sure you respect the licensing terms for the font).

**Note:**

It is not necessary to have all the fallback fonts installed on the system.

### Using Web Fonts

If the font is available from a website (such as Google Fonts), simply select the font, along with its font weights and insert the generated `@import` declaration in your CSS:

```css
@import url('https://fonts.googleapis.com/css?family=Montserrat:300,300i,400,500,700i');

section h2 {
  font-family: Montserrat, Serif;
  font-weight:500;
}
```

**Tip:**

It is possible to add a fallback to the remote font (Serif in the above example).

**Important:**

Make sure the website hosting the font is accessible. If you get errors in the console (for example, `Unknown host` or `I/O Exception`) regarding one of the font resources, check your networking proxy settings or your firewall settings. For the parameters that control the HTTP proxy, see Command Line Interface (on page 8).

### Using Installed Fonts

Suppose you want to style certain elements using a font that is available on your system. On Windows, it means it was installed in the Windows/Fonts directory. In this case, you can reference it directly like this:
Note:
When installing a font on Windows, make sure you select the **Install for all users** option.

### Using Local Font Files

#### Font Files Next to the CSS

If the font file is not installed in the system, you can place it next to your custom CSS file. You will have to declare one or more `@font-face` structures, with the same `font-family`, but with possible different `font-weight` and `font-style` properties (as in the example below where the `TitilliumWeb` font is a bundle of multiple `TTF` files, each one for a specific `font-weight` and style). The `TTF` files were placed in a folder next to the CSS named `fonts/titillium`.

```css
@font-face {
  font-family: titillium;
  font-style: normal;
  font-weight: 400;
  src: url(fonts/titillium/TitilliumWeb-Regular.ttf) ;
}
@font-face {
  font-family: titillium;
  font-style: normal;
  font-weight: 300;
  src: url(fonts/titillium/TitilliumWeb-Light.ttf) ;
}
@font-face {
  font-family: titillium;
  font-style: normal;
  font-weight: 200;
  src: url(fonts/titillium/TitilliumWeb-ExtraLight.ttf) ;
}
@font-face {
  font-family: titillium;
  font-style: normal;
  font-weight: 600;
  src: url(fonts/titillium/TitilliumWeb-SemiBold.ttf) ;
}
```
Font Files from a Directory

If you have your font files located in a particular directory, you can instruct Oxygen PDF Chemistry to load them. To do this, use the `-fonts-dir` command-line argument and just specify the name of the font directly in the `font-family` property, as you would for the built-in fonts. Using this approach there is no need to define a `@font-face` structure in the CSS.

**Tip:**

Using `@font-face` declarations in your CSS and keeping the font files next to the CSS is the recommended way to customize your output. In this way you can create a self-contained customization, with no need to deploy fonts in other directories.

Font Files from the Oxygen PDF Chemistry Installation

Another way is to copy your font files in the following subfolder: `[CHEMISTRY_INSTALL_DIR]/config/fonts`. Again, there is no need to define a `@font-face` structure in the CSS, just specify the name of the font directly in the `font-family` property.

**Note:**

When using the Oxygen Publishing Engine, the Chemistry processor is installed in the `[OPE_INSTALL_DIR]/plugins/com.oxygenxml.pdf.css/lib/oxygen-pdf-chemistry` folder. You will need to add the fonts in the `config/fonts` sub-folder. However, the best approach is to use `@font-face` definitions in your CSS, avoid altering the Chemistry installation.

In more simple cases, you might create a single `@font-face` Structure.

### Related Information:

- Command Line Interface (on page 8)

## Font Embedding

### CSS Font Embedding

All the font families that are referenced from the CSS are embedded automatically in the PDF by Oxygen PDF Chemistry.

### Basic Fonts Embedding

Although the basic fonts are guaranteed to be available in all PDF readers, there are some situations where you will have to embed them explicitly when using PDF/Universal Accessibility or Archiving. Because there are
some copyright restrictions on these fonts, Oxygen PDF Chemistry cannot redistribute them. You will have to locate the files on your system and declare a set of font faces with the same name as the default ones.

**Note:**
The following example assumes the fonts are placed in a `fonts` directory next to the stylesheet. The names of the fonts may differ on your platform.

```css
@font-face{
    font-family: Times;
    font-style: normal;
    font-weight: 400;
    src: url("fonts/TIMES.TTF");
}
@font-face{
    font-family: Times;
    font-style: italic;
    font-weight: 400;
    src: url("fonts/TIMESI.TTF");
}
@font-face{
    font-family: Times;
    font-style: italic;
    font-weight: 700;
    src: url("fonts/TIMESBI.TTF");
}
@font-face{
    font-family: Times;
    font-style: normal;
    font-weight: 700;
    src: url("fonts/TIMESBD.TTF");
}
/* -------------------------- */
@font-face{
    font-family: Helvetica;
    font-style: normal;
    font-weight: 400;
    src: url("fonts/ARIAL.TTF");
}
@font-face{
    font-family: Helvetica;
    font-style: italic;
    font-weight: 400;
    src: url("fonts/ARIALI.TTF");
}
@font-face{
    font-family: Helvetica;
    font-style: italic;
    font-weight: 700;
    src: url("fonts/ARIALBI.TTF");
}
```
src: url("fonts/COURBD.TTF");
}

/* ---------------------- */

@font-face{
  font-family: Symbol;
  font-style: normal;
  font-weight: 400;
  src: url("fonts/SYMBOL.TTF");
}

/* ---------------------- */

@font-face{
  font-family: "Zapf Dingbats";
  font-style: normal;
  font-weight: 400;
  src: url("fonts/WINGDING.TTF");
}

/* ---------------------- */

@font-face{
  font-family: Any;
  font-style: normal;
  font-weight: 400;
  src: url("fonts/TIMES.TTF");
}

@font-face{
  font-family: Any;
  font-style: italic;
  font-weight: 400;
  src: url("fonts/TIMESI.TTF");
}

@font-face{
  font-family: Any;
  font-style: italic;
  font-weight: 700;
  src: url("fonts/TIMESBI.TTF");
}

@font-face{
  font-family: Any;
Font Ligatures

Ligatures for non-Latin scripts (such as Arabic) are enabled by default.

For Latin scripts, you have to use the `font-variant-ligatures` CSS property:

```css
p {
    font-family: Calibri;
    font-variant-ligatures: common-ligatures;
}
```

This example results in the text being displayed with `font-variant-ligatures: none`:

**Attention**

Font Alternates

The `font-variant-alternates` CSS property controls the use of alternate glyphs, rather than the default ones.

⚠️ **Warning:**
Currently, Oxygen PDF Chemistry supports only the `styleset()` function for `font-variant-Alternates`.

Style Sets

The `styleset()` function defines a stylistic alternative for sets of characters. The parameter is a font-specific name mapped to a number. It corresponds to the OpenType value `ssXY` (e.g. `ss02`).

For example, if you have the following paragraph:

```html
<p>Lore ipsum dolor.</p>
```

and use the following CSS customization:

```css
p {
    font-family: Gabriola;
    font-variant-Alternates: styleset(ss06);
}
```

it will result in the following:
Using Simulated (Synthetic) Styles

Some fonts do not have a bold or italic variant, and Oxygen PDF Chemistry automatically falls back to the regular font. This topic demonstrates how to instruct Oxygen PDF Chemistry to synthesize font variants. Do not use this technique if you have all the font components.

⚠️ Important:
Before using simulated font-face variations, make sure you comply with the font licensing terms.

Using a Simulated Styles Font from a True Type File (ttf)

Suppose you plan to style the output with the fictitious Syncro-Oxygen font and you just have the Syncro-Oxygen-Regular.ttf font available. The regular font face definition will be:

```css
@font-face {
  font-family: "Syncro-Oxygen";
  font-style: normal;
  font-weight: 400; /* normal */
  src: url("fonts/raleway/Syncro-Oxygen-Regular.ttf");
}
```

For the missing style combinations, this example uses the -oxy-simulate-style: yes CSS extension property.

```css
@font-face {
  font-family: "Syncro-Oxygen";
  font-style: normal;
  font-weight: 700; /* bold */
  -oxy-simulate-style: yes;
  src: url("fonts/raleway/Syncro-Oxygen-Regular.ttf");
}
@font-face {
  font-family: "Raleway";
  font-style: italic;
  font-weight: 400; /* normal */
  -oxy-simulate-style: yes;
  src: url("fonts/raleway/Syncro-Oxygen-Regular.ttf");
}
@font-face {
  font-family: "Raleway";
```
Using a Simulated Font Style from a True Type Font Collection (ttc)

Chemistry Limitation:

The @font-face rules in your CSS work as expected when they are pointing to a .ttf file. If you intend to use a .ttc collection, the value of the font-family property should match exactly one of the font names from the collection (a collection contains multiple fonts).

The following example uses the fictitious Syncro-Fusion font (the sf.ttc file). Suppose this font provides regular, light, bold, but no italic or bold-italic variants. In this case, you may want Oxygen PDF Chemistry to generate the missing variants for you.

To use the regular variant, use a simple @font-face definition, making sure the font family is found in the collection:

```css
@font-face {
    font-family: "Syncro-Fusion";
    font-style: normal;
    font-weight: 400;
    src: url("fonts/syncro/sf.ttc");
}
```

For the missing style combinations, the example uses the -oxy-simulate-style: yes CSS extension property.

```css
@font-face {
    font-family: "Syncro-Fusion";
    font-style: normal;
    font-weight: 700;
    -oxy-simulate-style: yes;
    src: url("fonts/syncro/sf.ttc");
}
@font-face {
    font-family: "Syncro-Fusion";
    font-style: italic;
    font-weight: 400;
    -oxy-simulate-style: yes;
    src: url("fonts/syncro/sf.ttc");
}
@font-face {
    font-family: "Syncro-Fusion";
    font-style: normal;
    font-weight: 400;
    -oxy-simulate-style: yes;
    src: url("fonts/syncro/sf.ttc");
}
```
```
font-family: "Syncro-Fusion";
font-style: italic;
font-weight: 700;
-oxy-simulate-style: yes;
src: url("fonts/syncro/sf.ttc");
}

@font-face {
  font-family: "Syncro-Fusion";
  font-style: normal;
  font-weight: 100;
  -oxy-simulate-style: yes;
  src: url("fonts/syncro/sf.ttc");
}
```
Choosing the Fonts  

An important step in making sure your document is published properly in multiple scripts (languages) is choosing a font that covers the entire set of characters from that script (language).

However, you can specify a series of font families (using the CSS `font-family` property) that are used as fallbacks. If a word cannot be rendered using the first specified font, the processor tries the next font family, and so on. The following example uses some common fonts available in Windows to make a CSS stylesheet capable of properly displaying a large number of languages, from all European to Asian languages:

```css
* {
  font-family: Calibri, SimSun, "Malgun Gothic";
}
```

Assigning a font for all the elements, and not relying on basic fonts (on page 92), makes the document accessible (on page 52).

**Note:**
If your CSS does not assign a font to a part of your document, then the following sequence of fonts is used: Serif, Times, Times New Roman, DejaVu Serif, Symbol, SimSun, MingLiU, MS Mincho, Batang, Vijaya, Noto Serif CJK SC, Noto Serif CJK JP, Noto Serif CJK KR, Arial Unicode MS. This combination covers a wide character range, but the final result depends on the fonts from this list available on your OS. If a word contains a character that is not found in the current font, the fallback font list is iterated until one that supports all the word characters is found.

Support for Right-to-Left Languages

The Unicode BIDI algorithm is applied automatically. For the best results in HTML, make sure you mark the right-to-left content, or the left-to-right content embedded in right-to-left, with proper direction attributes:

```html
<p dir='rtl'>SOME ARABIC TEXT</p><p dir='ltr'>Some latin words.</p>
```

For arbitrary XML, use the `unicode-bidi` and `direction` CSS properties:

```xml
<para dir='right-to-left'>SOME ARABIC TEXT.</para>
```

CSS:

```css
para[dir='right-to-left'] {
  direction: rtl;
}
```
Note:
There are built-in rules in Oxygen PDF Chemistry that automatically match the @dir attribute in any XML vocabulary with the same semantic as for HTML. The accepted values are: ltr, rtl, auto. If you use other attribute names or other values for this attribute, you should add CSS rules similar to the one above.

For elements in a right-to-left context, Oxygen PDF Chemistry automatically switches the left borders, paddings, and margins with the ones from the right. This keeps your CSS as simple as possible.

In the following example, the p element has a left border.

```css
p {
  border-left: 1pt solid orange;
}
```

Suppose it is placed in a <div> with the default ltr direction. The orange line is painted in its left border. But if it is placed in a <div> with the rtl direction, the orange line will be painted in its right border because that is where the text begins.

Note:
To disable this behavior, you may use the -no-rtl-mirroring (on page 12) command-line parameter.

Changing Labels Depending on Language

When developing a CSS that will apply to output localized for multiple languages, you should consider changing the static text (CSS generated) depending on the language.

Note:
Your document must use the xml:lang or lang attributes to specify the content language, ideally on the root element. The value must be specified using a language identifier (such as "en", "en-US", "en-CA", "fr", "fr-CA").

Consider a case where all the chapter titles are prefixes with the word "Chapter", followed by the figure counter. Depending on the language of the XML/HTML document, you need this word to change to: "Kapitel" for German, or to "Chapitre" for French.

```html
<div class='chp'>
  <h2>Introduction</h2>
  ...
</div>
```
The CSS may be written starting with a default rule that will be used when the content has languages other than the ones that are expected (for example, in English):

```css
div.chp > h2:before{
    content: "Chapter " counter(chp);
}
```

Next, write rules for each of the specific languages:

```css
div.chp > h2:lang(de):before{
    content: "Kapitel " counter(chp);
}
div.chp > h2:lang(fr):before{
    content: "Chapitre " counter(chp);
}
```

**Tip:**

A good practice is to keep all the static text for a specific language in a separate CSS.

To make the maintenance easier, you can separate the strings from the counter value by using one of the advanced features of Oxygen PDF Chemistry (the :before and :after pseudo-elements with multiple levels (on page 61)). So you could write the default rule as:

```css
div.chp > h2:before(2){
    content: "Chapter ";
}
div.chp > h2:before(1){
    content: counter(chp);
}
```

Now, the more specific rules are more simple:

```css
div.chp > h2:lang(de):before(2){
    content: "Kapitel ";
}
div.chp > h2:lang(fr):before(2){
    content: "Chapitre ";
}
```

**Warning:**

The multiple-level pseudo-elements are non-standard features, and might not work if you switch to another processor.
7.

Tutorials

This section contains a simple tutorial that shows you how to format various aspects of an HTML version of a book to publish it in PDF output. This is also where you can find some how to topics that offer guidance for achieving certain results.

How To Insert the Current Date in the Cover Page

Oxygen PDF Chemistry provides a CSS extension, oxy_xpath, that can be used to execute XPath functions, including the fn:current-date function that simply returns the current date.

For example, if the text of the title is set in a `<booktitle>` element, you can use a synthetic block with the date content like this:

```
booktitle:after {
    content:oxy_xpath("current-date()"),
    color:gray,
    display:block;
}
```

How To Apply Chemistry on Multiple Files

It is possible to use Oxygen PDF Chemistry to transform multiple files in a directory at once. This can be done by using a particular script, depending on your operating system.

The following examples assume that you installed Oxygen PDF Chemistry and modified the PATH environment variable to include the installation directory (on page 7). Also, you can change the INPUT_DIR, CSS, and *xml file pattern according to your needs.

- **Windows**
  1. Create a `.bat` file with the following content:

```
@echo off

rem This is the input directory
set "INPUT_DIR=C:\some\input\directory"

rem This is the CSS used to style the xml documents.
set "CSS=C:\some\css\file"

rem Iterate all the files (%%f) that match the "*.xml" pattern in the input directory.
```
2. Run the .bat file from a command prompt.

   - **Linux/macOS**
     1. Create a script with the following content:

        ```
        #!/bin/sh
        
        # This is the input directory
        INPUT_DIR="/some/input/directory"
        
        # This is the CSS used to style the xml documents.
        CSS="/some/css/file"
        
        for f in $INPUT_DIR/*.xml; do chemistry.sh -in $f -out $f.pdf -css $CSS; done
        ```

2. Run the script from the Terminal application.

Full Example: How to Format a Book

This tutorial will show you how to format a book. For simplicity, the tutorial will use an HTML version of *The Adventures of Tom Sawyer* by *Mark Twain*, without many structures. It mainly consists of titles, paragraphs, and some meta-information. You can find the free ebook that is used for this tutorial at: [https://www.gutenberg.org](https://www.gutenberg.org) (search for *The Adventures of Tom Sawyer*).

Before getting started, save this file with the name "book.html" then create a CSS file named "book.css" in the same directory. Note that for the purposes of this tutorial, it is assumed that you are using *Oxygen XML Editor/Author* for your XML IDE.

1. Clean up the existing styles

   To make things easier, remove the `<style>` element from the header of the book. This will prevent mixing CSS rules coming from your CSS with the ones that were created for the browser display.

   Also, remove the `style="width:100%;"` attribute from all the images in the document.

2. Set up Chemistry in Oxygen

   To transform this book to PDF, configure Oxygen PDF Chemistry as an external tool in *Oxygen XML Editor/Author*. Go to **Tools > External Tools > Configure** menu, click the **New** button, and configure it as a new external tool. Set the **Command line** to:

   ```
   cmd /c "${oxygenInstallDir}\oxygenChemistry.bat" -catalogs ${xmlCatalogFilesList} -in "${cf}" -css "${cfd}/${cfn}.css" -out "${cfd}/${cfn}.pdf" -show-pdf
   ```

   **Result:** Every time you select the HTML book file in *Oxygen XML Editor/Author*, you can use this external tool from the toolbar.
3. Define the page size

To accommodate printing this book in a format similar to the original edition of the book, add the following to your CSS:

```css
@page {
  size: 6in 7.5in;
  margin: 0.5in;
}
```

4. Select fonts

Good novel books usually have clean serif fonts. You can choose one from Google Fonts by adding the following import at the beginning of the CSS file:

```css
@import url('https://fonts.googleapis.com/css?family=Crimson+Text');
```

Then, set it on the root of the document:

```css
:root {
  font-family: 'Crimson Text', serif;
  font-size: 11pt;
}
```

Besides the main content, the HTML book contains some descriptions and licensing terms, located in `<pre>` elements, directly under the `<body>` element. For these, to wrap the content (the default is not to wrap), use a smaller font:

```css
body > pre {
  font-family: sans-serif;
  font-size: 7pt;
  white-space: pre-wrap;
}
```

5. Transform the book with Chemistry

Try to transform the HTML book with Chemistry (on page 8). You will get something similar to this:
Notice that besides the formatting, Chemistry already helped with the following:

- Detected the publication title and set it as PDF. See the window title-bar.
- Created a tree of bookmarks by taking the `<H2>` elements into account.

For the purposes of this tutorial, the following still needs to be addressed:

- The preface, and each of the chapters should start on a new page.
- The chapter titles need to be formatted.
- The publication needs page numbers, page headers, and other styling.

6. Justify text

To improve the alignment of the right side of the book, justify the text by adding the following in the CSS:

```css
p {
  text-align: justify;
}
```

7. Make chapters start on a new page

Currently, the document is very flat and the chapters are just marked by the `<H2>` title elements between `<p>` elements. There are also `<pre>` elements used for the copyright and licensing:
<h2>CHAPTER III</h2>

TOM presented himself before Aunt Polly, who was sitting by an open window in a pleasant rearward apartment, which was bedroom, breakfast-room,

To make each of the chapters start on a new page, the CSS paged media module defines a way to forcibly break the page before an element:

```
h2 {
    page-break-before: always;
}
body > pre {
    page-break-before: always;
}
```

**Result:** Now all the chapters start at the beginning of a new page, and the book is starting to look like a real publication. If you want the chapters to always start on a page from the right side, use the `right` value for the property.

### 8. Format the chapter titles

Currently, the titles look rather dull, aligned to the left. Center them and give them styling:

```
h2 {
    text-align: center;
    font-size: larger;
}
```

### 9. Add page numbers

Most novels have the page numbers shown in the bottom center of the page. To achieve this, you can use the `@page` CSS counter set in a page margin box:

```
@page {
    @bottom-center {
        content: "-" counter(page) "-";
        font-size: 8pt;
    }
}
```
Result: What the document looks like so far

You have solved the text justification, page breaks, and page numbers. This is what the document looks like now:

Remaining things to be addressed:

- It needs a cover page.
- The page numbering should restart on the first chapter, after the preface, and should end before the licensing terms.

10. Add a cover page

You can find the original cover of the book on the same website as the Gutenberg project. For this tutorial, use this as artwork for the first page.

Start by defining a named page in your CSS file, with no page counter in the bottom-center region:

```css
@page cover-page {
    background-image: url('https://www.gutenberg.org/files/74/74-h/images/bookcover.jpg');
    background-size: 6in 7.5in;
    background-repeat: no-repeat;
}

@bottom-center {
    content: none;
}
```
When using images for your cover pages, make sure they respect the same aspect ratio as your page (width/height ratio), then use the `background-size` property to stretch it exactly to the page size.

Next, link this page to a synthetic element placed before the root. You can use a `:before` pseudo-element in the `<html>` root element:

```html
html:before{
  content: " ";
  page:cover-page;
}
```

You could place text over the cover image, but for the moment, just leave the content with blank text (a whitespace). It is necessary to have a `content` property that is not empty because Oxygen PDF Chemistry discards all the pseudo-elements without one.

### 11. Reset and style page numbers

To make the page numbers be restarted at the beginning of the first chapter, you can use the first title that follows the metadata at the beginning of the document:

```html
pre + h2{
  counter-reset: page 1;
}
```

The licensing terms at the end of the book can be numbered independently, and styled differently:

```html
body > pre {
  counter-reset:page;
  page:copyright-license-page;
}
```

For the copyright page, use a lower-roman numbering style:

```html
@page copyright-license-page{
  background-color:yellow;
  @bottom-center {
    content:counter(page, lower-roman);
  }
}
```

### Final Result

Now you have a nice looking book that can be distributed electronically, or printed:
8.

Appendix

CSS Media Rules

Oxygen PDF Chemistry applies all the element CSS selectors that are:

- not enclosed in a @media rule.
- enclosed in a @media rule with type print or oxygen-chemistry.

It is recommended to enclose the rules that use Oxygen extensions in @media oxygen-chemistry, and the ones that apply only to print into a @media print.

```css
chapter {
    margin-left: 2em;
}

@media oxygen-chemistry {
    chapter:before(2) {
        content: counter(chapter);
    }
}

@media print {
    * {
        color: black;
    }
}
```

Tip:
When debugging the results of the applied CSS styling, CSS selectors enclosed in @media print {..} are often not activated by default. There are several ways to activate them. For details, see Debugging the CSS (on page 16).

CSS Selectors

The following table summarizes the selectors supported by Oxygen PDF Chemistry:
<table>
<thead>
<tr>
<th>Pattern</th>
<th>Meaning</th>
<th>Described in section</th>
<th>CSS level</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>Any element</td>
<td>Universal selector</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>An element of type E</td>
<td>Type element selector</td>
<td>1</td>
</tr>
<tr>
<td>E[foo]</td>
<td>An E element with the &quot;foo&quot; attribute set</td>
<td>Attribute selectors</td>
<td>2</td>
</tr>
<tr>
<td>E[foo=&quot;bar&quot;]</td>
<td>An E element whose &quot;foo&quot; attribute value is exactly equal to &quot;bar&quot;</td>
<td>Attribute selectors</td>
<td>2</td>
</tr>
<tr>
<td>E[foo~=&quot;bar&quot;]</td>
<td>An E element whose &quot;foo&quot; attribute value is a list of space-separated values, one of which is exactly equal to &quot;bar&quot;</td>
<td>Attribute selectors</td>
<td>2</td>
</tr>
<tr>
<td>E[foo^=&quot;bar&quot;]</td>
<td>An E element whose &quot;foo&quot; attribute value begins exactly with the string &quot;bar&quot;</td>
<td>Attribute selectors</td>
<td>3</td>
</tr>
<tr>
<td>E[foo$=&quot;bar&quot;]</td>
<td>An E element whose &quot;foo&quot; attribute value ends exactly with the string &quot;bar&quot;</td>
<td>Attribute selectors</td>
<td>3</td>
</tr>
<tr>
<td>E[foo*=&quot;bar&quot;]</td>
<td>An E element whose &quot;foo&quot; attribute value contains the substring &quot;bar&quot;</td>
<td>Attribute selectors</td>
<td>3</td>
</tr>
<tr>
<td>E[lang=&quot;en&quot;]</td>
<td>An E element whose &quot;lang&quot; attribute has a hyphen-separated list of values beginning (from the left) with &quot;en&quot;</td>
<td>Attribute selectors</td>
<td>2</td>
</tr>
<tr>
<td>E:root</td>
<td>An E element, root of the document</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:first-child</td>
<td>An E element, first child of its parent</td>
<td>Structural pseudo-classes</td>
<td>2</td>
</tr>
<tr>
<td>E:last-child</td>
<td>An E element, last child of its parent</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:first-of-type</td>
<td>An E element, first sibling of its type</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:last-of-type</td>
<td>An E element, last sibling of its type</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:only-of-type</td>
<td>An E element, only sibling of its type</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:only-child</td>
<td>An E element, only child of its parent</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:nth-child(n)</td>
<td>An E element, the n-th child of its parent</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>Pattern</td>
<td>Meaning</td>
<td>Described in section</td>
<td>CSS level</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>-------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>E:nth-of-type(n)</td>
<td>An E element, the n-th sibling of its type</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:empty</td>
<td>An E element that has no children (including text nodes)</td>
<td>Structural pseudo-classes</td>
<td>3</td>
</tr>
<tr>
<td>E:lang(c)</td>
<td>An element of type E in (human) language c (the document language specifies how language is determined)</td>
<td>The :lang() pseudo-class</td>
<td>2</td>
</tr>
<tr>
<td>E:has(F)</td>
<td>An E element with the condition F applying to one of its children. Similar to the subject selector described in: The subject element pseudo-class.</td>
<td>Extension.</td>
<td>3</td>
</tr>
<tr>
<td>E:first-letter</td>
<td>The first formatted letter of an E element</td>
<td>The :first-letter pseudo-element</td>
<td>1</td>
</tr>
<tr>
<td>E:before</td>
<td>Generated content before an E element</td>
<td>The :before pseudo-element</td>
<td>2</td>
</tr>
<tr>
<td>E:after(n)</td>
<td>Generated content after an E element. Setting a value of 1 creates a normal :before. A higher value means the generated content is further from the element.</td>
<td>Extension</td>
<td></td>
</tr>
<tr>
<td>E:after</td>
<td>Generated content after an E element</td>
<td>The :after pseudo-element</td>
<td>2</td>
</tr>
<tr>
<td>E:before(n)</td>
<td>Generated content before an E element. Setting a value of 1 creates a normal :before. A higher value means the generated content is further from the element.</td>
<td>Extension</td>
<td></td>
</tr>
<tr>
<td>E:marker</td>
<td>Generated by list items to represent the item's marker (the bullet or number identifying each item). E is supposed to have display list-item</td>
<td>Marker pseudo-element</td>
<td>3</td>
</tr>
<tr>
<td>E#myid</td>
<td>An E element, its ID being equal to &quot;myid&quot;</td>
<td>ID selectors</td>
<td>1</td>
</tr>
<tr>
<td>E F</td>
<td>An F element descendant of an E element</td>
<td>Descendant combinator</td>
<td>1</td>
</tr>
<tr>
<td>E &gt; F</td>
<td>An F element child of an element E</td>
<td>Child combinator</td>
<td>2</td>
</tr>
<tr>
<td>E + F</td>
<td>An F element immediately preceded by an element E</td>
<td>Direct adjacent combinator</td>
<td>2</td>
</tr>
<tr>
<td>E ~ F</td>
<td>An F element preceded by an element E</td>
<td>Indirect adjacent combinator</td>
<td>3</td>
</tr>
</tbody>
</table>
CSS Functions

Supported CSS functions include:

*[attr()]*

It is used to retrieve the value of an attribute of the selected element and use it in the style sheet. The `attr()` function can be used with any CSS property. For more details, see MDN Web Docs: [attr()](https://developer.mozilla.org/en-US/docs/Web/CSS/attr).

*url()*

The `<url>` CSS data type denotes a pointer to a resource, such as an image or a font. It has no proper syntax and can only be expressed through the `url()` functional notation. URLs are used in numerous CSS properties, such as `background-image`, `cursor`, `@font-face`, and `list-style-image`.

*leader()*

Used to fill a space with a pattern.

```css
elem::before {
  content: "A" leader(".") "B";
  display: block;
}
```

The text "A" and "B" should be on the left and right sides of the page, with a line full of dots between them. See Creating a Table of Contents (TOC) *(on page 43)* for more examples.

*string()*

Retrieves the value of a string-set. See Headers and Footers *(on page 32)* for use-cases.

*calc()*

The `calc()` function allows mathematical expressions with addition (+), subtraction (-), multiplication (*), division (/) to be used as component values. Percentages are solved relative to the dimensions of the containing parent block. It can be used when length values are accepted:

```css
elem {
  width: calc(100% - 1em);
}
```

For more information, see: [https://drafts.csswg.org/css-values-3/#calc-notation](https://drafts.csswg.org/css-values-3/#calc-notation).

*content()*

Retrieves the text content from the current element. Used in a string-set definition. See Headers and Footers *(on page 32)* for use-cases.

*counter()*

Retrieves the value of the innermost counter of that name on the element. [https://www.w3.org/TR/css-lists-3/#funcdef-counter](https://www.w3.org/TR/css-lists-3/#funcdef-counter)
counters()

Retrieves all the values of a counter of that name on the element, separated by a delimiter.  
https://www.w3.org/TR/css-lists-3/#funcdef-counters

target-counter()

The target-counter() function retrieves the value of the innermost counter with a given name. The 
required arguments are the url of the target and the name of the counter. An optional counter-
style argument can be used to format the result.  
https://www.w3.org/TR/css-gcpm-3/#target-counter

target-counters()

The target-counters() function fetches the value of all counters of a given name from the end of 
a link, and formats them by inserting a given string between the value of each nested counter.  
https://www.w3.org/TR/css-gcpm-3/#funcdef-target-counters

oxy_ ... ()

This is a collection of extension functions that are only recognized by Oxygen products. They 
can be used to process strings, do mathematical calculations, execute XPath queries over 
the document, and retrieve text from the document. For the complete list, see: Custom CSS 
Functions.

The following example counts the number of words from a paragraph (including tracked 
changes) and displays the result in front of it:

```css
p:before{
  content: 
    concat("|Number of words:",
        oxy_xpath("count(tokenize(normalize-space(string-join(text(), '')), ' '))", 
            processChangeMarkers, 
            true),
        "| ");
}
```

CSS Units

Oxygen PDF Chemistry supports the following absolute CSS Units:

- **in** (inches) - 1in is equal to 2.54cm.
- **cm** (centimeters)
- **mm** (millimeters)
- **pt** (points) - One CSS point is equal to 1/72 of 1in.
- **pc** (picas) - 1pc is equal to 12pt.
- **px** (pixel units) - 1px is equal to 0.75pt.
The supported relative units are:

- **em** - Equal to the computed font-size property of the element on which it is used.
- **rem** - Equal to the computed font-size property of the document root element.

## CSS Properties

### Standard W3C CSS Properties

Table 2. The following standard properties are supported by Oxygen PDF Chemistry

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>alignment-baseline</td>
<td>middle</td>
<td>-oxy-first-line</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>background</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>background-color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>background-image</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>background-position</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>background-repeat</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>background-size</td>
<td>length</td>
<td>percent</td>
<td>initial</td>
</tr>
<tr>
<td>bookmark-label</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bookmark-level</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bookmark-state</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-bottom</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 2. The following standard properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>border-bottom-color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-bottom-left-radius</td>
<td>length</td>
<td>percent</td>
<td>initial</td>
</tr>
<tr>
<td>border-bottom-right-radius</td>
<td>length</td>
<td>percent</td>
<td>initial</td>
</tr>
<tr>
<td>border-bottom-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-bottom-width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-collapse</td>
<td>separate</td>
<td>collapse</td>
<td>initial</td>
</tr>
<tr>
<td>border-color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-left</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-left-color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-left-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-left-width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-right</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-right-color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-right-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-radius</td>
<td>length</td>
<td>percent</td>
<td>initial</td>
</tr>
<tr>
<td>border-right-style</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 2. The following standard properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>border-right-width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-spacing</td>
<td>&lt;length&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>border-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-top</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-top-color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-top-left-radius</td>
<td>length</td>
<td>percent</td>
<td>initial</td>
</tr>
<tr>
<td>border-top-right-radius</td>
<td>length</td>
<td>percent</td>
<td>initial</td>
</tr>
<tr>
<td>border-top-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-top-width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>border-width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bottom-property</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>break-after</td>
<td>auto</td>
<td>always</td>
<td>all</td>
</tr>
<tr>
<td>break-before</td>
<td>auto</td>
<td>always</td>
<td>all</td>
</tr>
<tr>
<td>break-inside</td>
<td>auto</td>
<td>avoid</td>
<td>initial</td>
</tr>
</tbody>
</table>
Table 2. The following standard properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
</table>
| caption-side        | top | bottom | initial | left | right |                    | • Counters do not add up: defining new counters on an element already defining a counter will reset the others.  
• Counters are "self-nesting": a counter defined in a parent can be reused in a child of this parent, read more at Nested Counters and Scope. |
| color               |           |        |        |          |          |                    |                                                     |
| column-span         |           |        |        |          |          |                    |                                                     |
| content             |           |        |        |          |          |                    |                                                     |
| counter-increment   |           |        |        |          |          |                    |                                                     |
| counter-reset       |           |        |        |          |          |                    |                                                     |
| direction           |           |        |        |          |          |                    |                                                     |
| display             | inline | block | inline-block | table | table-row | table-cell | table-caption | table-row-group | table-header-group | table-footer-group | table-column | -oxy-morph | initial | unset | For `display:none`, the elements are not eliminated from the DOM. They are just collapsed. The text may remain in the document, but is not visible. This is needed for keeping the links working.  
For `-oxy-morph`, read more here. |
<p>| empty-cells         |           |        |        |          |          |                    |                                                     |
| float               |           |        |        |          |          |                    |                                                     |
| font                |           |        |        |          |          |                    |                                                     |
| font-family         |           |        |        |          |          |                    |                                                     |
| font-size           |           |        |        |          |          |                    |                                                     |
| font-src            |           |        |        |          |          |                    |                                                     |
| font-style          |           |        |        |          |          |                    |                                                     |</p>
<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>font-variant</td>
<td>small-caps</td>
<td>normal</td>
<td>initial</td>
</tr>
<tr>
<td>font-variant-alternates</td>
<td>styleset</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>font-variant-ligatures</td>
<td>normal</td>
<td>none</td>
<td>common-ligatures</td>
</tr>
<tr>
<td>font-weight</td>
<td>normal</td>
<td>none</td>
<td>lining-nums</td>
</tr>
<tr>
<td>height</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 2. The following standard properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>hyphens</td>
<td>auto</td>
<td>manual</td>
<td>initial</td>
</tr>
<tr>
<td>image-resolution</td>
<td>&lt;dpi&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>left-property</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>letter-spacing</td>
<td></td>
<td></td>
<td>Discards the font ligatures.</td>
</tr>
<tr>
<td>line-height</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>list-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>list-style-image</td>
<td>none</td>
<td>url()</td>
<td>initial</td>
</tr>
<tr>
<td>list-style-position</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>list-style-type</td>
<td>&lt;string&gt;</td>
<td>box</td>
<td>check</td>
</tr>
<tr>
<td>margin</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>margin-bottom</td>
<td>&lt;length&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>margin-left</td>
<td>auto</td>
<td>&lt;length&gt;</td>
<td>initial</td>
</tr>
<tr>
<td>margin-right</td>
<td>auto</td>
<td>&lt;length&gt;</td>
<td>initial</td>
</tr>
<tr>
<td>margin-top</td>
<td>&lt;length&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
</tbody>
</table>
Table 2. The following standard properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>max-height</td>
<td>&lt;length&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>max-width</td>
<td>&lt;length&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>min-height</td>
<td>&lt;length&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>min-width</td>
<td>&lt;length&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>orphans</td>
<td>&lt;number&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>outline</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outline-color</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outline-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>outline-width</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>overflow-wrap</td>
<td>break-word</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>padding</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>padding-bottom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>padding-left</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>padding-right</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>padding-top</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>page</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>page-break-after</td>
<td>auto</td>
<td>always</td>
<td>avoid</td>
</tr>
</tbody>
</table>
Table 2. The following standard properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>page-break-before</td>
<td>auto</td>
<td>always</td>
<td>avoid</td>
</tr>
<tr>
<td>page-break-inside</td>
<td>auto</td>
<td>avoid</td>
<td>initial</td>
</tr>
<tr>
<td>position</td>
<td>fixed</td>
<td>relative</td>
<td>absolute</td>
</tr>
<tr>
<td>right-property</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>size</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>string-set</td>
<td>content()</td>
<td>counter()</td>
<td></td>
</tr>
<tr>
<td>table-column-span</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>table-layout</td>
<td>fixed</td>
<td>auto</td>
<td>initial</td>
</tr>
<tr>
<td>table-row-span</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>text-align</td>
<td>start</td>
<td>end</td>
<td>left</td>
</tr>
<tr>
<td>text-decoration</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>text-decoration-line</td>
<td>solid</td>
<td></td>
<td>double</td>
</tr>
<tr>
<td>text-decoration-style</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>text-indent</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>text-transform</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>top-property</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>transform</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 2. The following standard properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Supported Values</th>
<th>Unsupported Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>transform-rotation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>unicode-bidi</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vertical-align</td>
<td>baseline</td>
<td>sub</td>
<td>super</td>
</tr>
<tr>
<td>visibility</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>white-space</td>
<td>normal</td>
<td>nowrap</td>
<td>pre</td>
</tr>
<tr>
<td>widows</td>
<td>&lt;number&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>width</td>
<td>&lt;length&gt;</td>
<td>&lt;percentage&gt;</td>
<td>calc</td>
</tr>
</tbody>
</table>

Note:
The initial and unset values are not supported when applied to Page Margin Boxes (on page 25).
## Extension CSS Properties

### Table 3. The following extension properties are supported by Oxygen PDF Chemistry

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>-oxy-alt-text</td>
<td>Used to specify an alternative description for the element that is used by the PDF readers.</td>
<td><code>&lt;string&gt;</code></td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>image { -oxy-alt-text: &quot;Image about: &quot; attr(href); }</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-avoid-breaking-line-at-hyphens</td>
<td>This can be used to avoid a line break inside a word that contains a hard hyphen. This property is deprecated and has been replaced by -oxy-break-line-at-hyphens (on page 128).</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>-oxy-borders-conditionality</td>
<td>This can be used to configure whether or not bottom and top borders appear on a split element when a page break occurs. Supported values are:</td>
<td>discard</td>
<td>discard</td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td>• discard - Makes the borders disappear.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• retain - Forces the bottom and top border to be displayed.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• inherit - Inherits the value specified in the parent element.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>td { -oxy-borders-conditionality: retain; }</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-break-line-at-hyphens</td>
<td>This can be used to set the line break inside a word that contains a hard hyphen. Supported values are:</td>
<td>auto</td>
<td>avoid</td>
<td>always</td>
</tr>
<tr>
<td></td>
<td>• auto - Let the default hyphenation algorithm process the words</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• avoid - Forces the algorithm not to break inside a word at hard hyphen</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• always - Forces the algorithm to break inside a word at hard hyphen</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
</table>
| -oxy-caption-repeat-on-next-pages | This can be used to enable table caption repetition on subsequent pages (when the table is long and it spans multiple pages). It only affects elements with the `table-caption` display.  
Also see the `-oxy-show-only-when-caption-repeated-on-next-pages (on page 137)` property that can be applied on optional caption static content. | yes | no | yes |
| -oxy-changebar-offset | Defines the distance between the text and the change bar. A positive distance is directed away from the column region and into the margin regardless of the `-oxy-changebar-placement`. A negative distance is directed towards the content area. | `<length>` | initial | unset |
| -oxy-changebar-placement | This property determines where, relative to the column areas, the change bars will occur. | `start` | `end` | `left` | `right` | `inside` | `outside` | `alternate` | `initial` | unset | left | yes |
| -oxy-changebar-color | Specifies the color of the change bar. | `<color>` | initial | unset | black | yes |
| -oxy-changebar-style | Specifies the style of the change bar. | `<border-style>` | initial | unset | solid | yes |
| -oxy-changebar-width | Specifies the thickness of the change bar. | `<border-width>` | initial | unset | medium | yes |

**Note:**  
Breaking the line at a hard hyphen can cause the PDF readers to join the lines but discard the hyphen character when copying and pasting content.
### Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>-oxy-column-break-after</td>
<td>This can be used to either force or eliminate a column break between the matching element and the next sibling. Supported values are: always - to force the column break, or avoid - to keep the end of the matching element together with the next sibling beginning in the same column.</td>
<td>always</td>
<td>auto</td>
<td>avoid</td>
</tr>
<tr>
<td>-oxy-column-break-before</td>
<td>This can be used to either force or eliminate a column break between the matching element and the previous sibling. Supported values are: always - to force the column break, or avoid - to keep the end of the previous sibling together with the beginning of the matching element in the same column.</td>
<td>always</td>
<td>auto</td>
<td>avoid</td>
</tr>
<tr>
<td>-oxy-column-break-inside</td>
<td>This can be used to avoid column breaks inside the matching element. Use with care, if the element has large content, it may bleed out of the page.</td>
<td>always</td>
<td>auto</td>
<td>initial</td>
</tr>
<tr>
<td>-oxy-force-page-count</td>
<td>Used to impose a constraint on the number of pages in a page sequence. It may increase the total number of pages from the page sequence by adding blank pages.</td>
<td>auto</td>
<td>even</td>
<td>odd</td>
</tr>
</tbody>
</table>
Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>-oxy-foreground-image</td>
<td>This can be used to define a foreground image that will be displayed above the content, like a watermark.</td>
<td>url()</td>
<td>none</td>
<td>inherit</td>
</tr>
<tr>
<td>-oxy-hyphenation-character</td>
<td>Used to specify the Unicode character to be presented when a hyphenation break occurs. To hide hyphens, use the space character.</td>
<td>&lt;character&gt;</td>
<td>U+00AD</td>
<td>yes</td>
</tr>
<tr>
<td>-oxy-hyphenation-push-character-count</td>
<td>Used to specify the minimum number of characters in a hyphenated word after the hyphenation character (the minimum number of characters in the word pushed to the next line).</td>
<td>&lt;number&gt;</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>-oxy-hyphenation-remain-character-count</td>
<td>Used to specify the minimum number of characters in a hyphenated word before the hyphenation character (the minimum number of characters in the word left on the line ending with the hyphenation character).</td>
<td>&lt;number&gt;</td>
<td>2</td>
<td>yes</td>
</tr>
</tbody>
</table>
### Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>p</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>-oxy-hyphenation-remain-character-count: &quot;2&quot;;</code></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>-oxy-initial-page-number</strong></td>
<td>Used to set the initial page number to be used on this particular page sequence.</td>
<td>auto</td>
<td>auto-odd</td>
<td>no</td>
</tr>
</tbody>
</table>
| ``` @page chapter-page {  
    -oxy-initial-page-number: auto-odd;  
} ``` |                                                                                                  |                  |               |             |
| **-oxy-link** | Used to create links in the PDF output.                                                          | none | url() | no |
| ``` link {  
    -oxy-link: attr(href);  
} ``` |                                                                                                  |                  |               |             |
| **-oxy-page-group** | Using the *start* value forces the creation of a new page group (sequence) even if the page before the element has the same name. The *auto* value uses the W3C algorithm normally, which would merge the current element with the open page sequence. | start | auto | no |
| ``` chapter {  
    -oxy-page-group:start;  
    page: chapter;  
} ``` |                                                                                                  |                  |               |             |
| **-oxy-pdf-meta-author** | Used to forward meta-information to the PDF. Represents the author of the publication.         | `<string>` | initial | no |
| ``` meta[name='author'] {  
    -oxy-pdf-meta-author: attr(value);  
} ``` |                                                                                                  |                  |               |             |
| **-oxy-pdf-meta-custom** | Used to forward meta-information to the PDF. Represents a generic custom document property. Should have two strings, the name and the value. | `<string>` | initial | no |
| ``` ``` |                                                                                                  |                  |               |             |
Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadata</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-meta-custom</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>metadata</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-meta-copyright</td>
<td>Specifies the copyright text.</td>
<td>&lt;string&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>-oxy-pdf-meta-copyrighted</td>
<td>Specifies whether or not the publication is copyrighted or from public domain.</td>
<td>copyrighted</td>
<td></td>
<td>public-domain</td>
</tr>
<tr>
<td>-oxy-pdf-meta-copyright-url</td>
<td>Specifies the location of the external copyright notice.</td>
<td>&lt;string&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>-oxy-pdf-meta-description</td>
<td>Used to forward meta-information to the PDF. Represents the description of the publication.</td>
<td>&lt;string&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>-oxy-pdf-meta-keyword</td>
<td>Used to forward meta-information to the PDF. Represents a single keyword from the publication. The processor should aggregate all keyword definitions and separate them by comma.</td>
<td>&lt;string&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>-oxy-pdf-meta-keywords</td>
<td>Used to forward meta-information to the PDF. Represents the keywords of the publication. The value should contain the keywords separated by commas.</td>
<td>&lt;string&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
</tbody>
</table>
Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>meta[name='keywords'] {</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-meta-keywords: attr(value);</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-meta-title</td>
<td>Used to forward meta-information to the PDF. Represents the title of the publication.</td>
<td>&lt;string&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>meta[name='title'] {</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-meta-title: attr(value);</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-table-omit-footer-at-break</td>
<td>Flag that indicates if the table footer should be omitted after a page break.</td>
<td>true</td>
<td>false</td>
<td>inherit</td>
</tr>
<tr>
<td>-oxy-pdf-table-omit-header-at-break</td>
<td>Flag that indicates if the table header should be omitted after a page break.</td>
<td>true</td>
<td>false</td>
<td>inherit</td>
</tr>
<tr>
<td>-oxy-pdf-viewer-display-filename</td>
<td>Flag that indicates whether the PDF viewer should display the filename (true) or the document title (false) for the generated PDF.</td>
<td>true</td>
<td>false</td>
<td>initial</td>
</tr>
<tr>
<td>:root {</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-viewer-display-filename: true;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>:root {</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-viewer-fit-window: true;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>:root {</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-pdf-viewer-hide-menubar: true;</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>-oxy-pdf-viewer-hide-toolbar</td>
<td>Flag that indicates whether or not the PDF viewer should hide its toolbar when opening the PDF document.</td>
<td>true</td>
<td>false</td>
<td>initial</td>
</tr>
<tr>
<td></td>
<td>:root {</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-oxy-pdf-viewer-hide-toolbar: true;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>single-page</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Displays one page at a time.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>one-column</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Displays the pages in one column.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>two-column-left</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Displays the pages in two columns, with odd-numbered pages on the left.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>two-column-right</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Displays the pages in two columns, with odd-numbered pages on the right.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>two-page-left</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Displays the pages two at a time, with odd-numbered pages on the left.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>two-page-right</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Displays the pages two at a time, with odd-numbered pages on the right.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:**

These values are similar to those from the PDF specification (the "Document Catalog/Page Layout" section).
Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>-oxy-pdf-viewer-page-mode</td>
<td>Specifies which left-side view should be visible when the PDF document is opened, or if the document should be full screen. The use-outlines value refers to the bookmark tree.</td>
<td>use-none</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>:root {</td>
<td>use-thumbs</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>-oxy-pdf-viewer-page-mode: use-outlines;</td>
<td>full-screen</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>initial</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td>unset</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td><strong>use-none</strong></td>
<td>use-thumbs</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>Neither document outlines nor thumbnail images are visible.</td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td><strong>use-outlines</strong></td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>Document outlines (bookmarks) are visible.</td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td><strong>use-thumbs</strong></td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>Thumbnail images are visible.</td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td><strong>full-screen</strong></td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>Full-screen mode, with no menu bar, window controls, or any other window visible.</td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong></td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>These values are similar to those from the PDF specification (the &quot;Document Catalog/Page Mode&quot; section).</td>
<td>use-outlines</td>
<td>use-outlines</td>
<td>no</td>
</tr>
<tr>
<td>-oxy-pdf-viewer-zoom</td>
<td>A percentage or a fit value that indicates the zoom factor or the fitting factor of the PDF viewer.</td>
<td>&lt;percent&gt;</td>
<td>fit-page</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>:root {</td>
<td>fit-width</td>
<td>fit-page</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>-oxy-pdf-viewer-zoom: 125%;</td>
<td>fit-height</td>
<td>fit-page</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>}</td>
<td>initial</td>
<td>fit-page</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>:root {</td>
<td>unset</td>
<td>fit-page</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td>-oxy-pdf-viewer-zoom: fit-height;</td>
<td></td>
<td>fit-page</td>
<td>no</td>
</tr>
</tbody>
</table>
Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>-oxy-pdf-tag-type</td>
<td>Maps an element to a PDF accessibility tag.</td>
<td>&lt;string&gt;</td>
<td>initial</td>
<td>unset</td>
</tr>
<tr>
<td>-oxy-simulate-style</td>
<td>Set this flag to <strong>yes</strong> when the styles (bold, italic, or both) need to be simulated from a regular font. See: Using Simulated (Synthetic) Styles (on page 100).</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>-oxy-space-after-conditionality</td>
<td>Specifies whether the bottom margin (<code>margin-bottom</code> property) of an element should be kept or discarded.</td>
<td>discard</td>
<td>retain</td>
<td>inherit</td>
</tr>
<tr>
<td>-oxy-space-before-conditionality</td>
<td>Specifies whether the top margin (<code>margin-top</code> property) of an element should be kept or discarded.</td>
<td>discard</td>
<td>retain</td>
<td>inherit</td>
</tr>
<tr>
<td>-oxy-style</td>
<td>An Oxygen extension used to define additional styles for an element (for example, collected from an attribute).</td>
<td>CSS fragment</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>-oxy-show-only-when-caption-repeated-on-next-pages</td>
<td>This property only affects a <strong>:before</strong> or <strong>:after</strong> pseudo element that is associated with a table caption (an element with the table caption display marked with <strong>-oxy-</strong></td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>
Table 3. The following extension properties are supported by Oxygen PDF Chemistry (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Supported Values</th>
<th>Default Value</th>
<th>Inheritable</th>
</tr>
</thead>
<tbody>
<tr>
<td>caption-repeat-on-next-pages:yes</td>
<td>It signals that the static content is to be used only when the caption is displayed the second time (on the subsequent pages), when the table is long and it spans multiple pages. Also see: -oxy-caption-repeat-on-next-pages (on page 129).</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-oxy-video-cover</td>
<td>This property can be used to set a cover image on videos. This image will appear as a placeholder in the PDF output. If it is not set, a default cover is used.</td>
<td>none</td>
<td>initial</td>
<td>unset</td>
</tr>
</tbody>
</table>

Note:
The initial and unset values are not supported when applied to Page Margin Boxes (on page 25).

XML Support

Supported Schema Languages

Although it does not validate the input, Oxygen PDF Chemistry uses information from the schema associated with the document. The supported schema languages are:

- XML Schema
- DTD
- Relax NG

Entities

All XML entities are expanded prior to processing.

xi:include

All the xi:included fragments are expanded prior to processing.

Default Attributes

The default and fixed attributes defined in the associated schema are taken into account when the CSS rules are applied.

For instance, the DITA @class attributes are defined in the DTD rather than the XML instance (the CSS rules usually match this attribute, not the element names).
**Configuration File**

Oxygen PDF Chemistry uses the XSL-FO FOP processor as a last processing stage. You can control FOP by changing the `config/chemistry-fop.xconf` configuration file.

**Note:**

This file is a template that Oxygen PDF Chemistry uses to fill it with information about fonts. If you change it, make sure you preserve the font-related variables that are automatically expanded, or at least make sure you define the set of fonts that are used from the CSS.

**Related Information:**

- Apache FOP: Configuration

**Technical Issues**

Information about error messages and known issues for Chemistry.

This topic contains information about error messages, possible known issues, and other things that you might want to be aware of in regards to Oxygen PDF Chemistry.

**Out of Memory Error**

If you receive an error message indicating an *Out of Memory Error*, you can try the following:

- Make sure you are using Java 17.
- Use the `-Xmx<NNN>m` command-line parameter (on page 14) to specify a bigger value for the maximum allocated memory. If you are running Java on 64-bit, you can specify a larger memory size (for example, `-Xmx2048m`).

**Copying and Pasting Text from the PDF Reader does not Preserve Line Breaks**

The generated PDF is accessible. This causes some of the readers (such as Acrobat Reader) to change the way the lines are saved in the clipboard. The effect is that codeblock lines may be joined when pasting them into a text editor.

One solution is to disable the accessibility feature. To do this, edit the `config/chemistry-fop.xconf` file and change the `<accessibility>` element to `false`:

```xml
<accessibility>false</accessibility>
```
Index

Special Characters
:after pseudo-element
61
:before pseudo-element
61

A
Accessibility (508 compliance
52
Aligning blocks
74
Anchors
52
Archiving
55
Associating a CSS to a document
21
Avoid line breaks at hyphens
60

B
Bookmarks
47

C
Change Bars
62
Command-line arguments
8
Configuration file
139
Configuring PDF Chemistry as an external tool
8
Cross references
41
CSS functions
117
CSS media rules
114
CSS properties
119
CSS selectors
114

CSS units
118

D
Debugging
16
DITA to PDF from a command line
15

E
Element layout
64
Embedding fonts
95
Ending chapters on odd/even page
30
Error messages
139

F
Floats
74
Font Variant
99
Fonts
92
Footer
Dynamic images
38
Footers
32
Footnotes
39
Force line breaks at hyphens
60

G
Getting started
6
Graphics
77
Background images
80
Foreground images