Maven PDF Plugin

v. 1.1

User Guide

Table of Contents

1 Introduction

1.1 Overview

This plug-in allows you to generate a PDF version of your project's documentation.

1.1.1 Goals Overview

The PDF Plugin only has one goal.

- pdf:pdf Generates a PDF document containing all project documentation.

1.1.2 Usage

General instructions on how to use the PDF Plugin can be found on the usage page. Some more specific use cases are described in the examples given below. Last but not least, users occasionally contribute additional examples, tips or errata to the plugin's wiki page.

In case you still have questions regarding the plugin's usage, please have a look at the FAQ and feel free to contact the user mailing list. The posts to the mailing list are archived and could already contain the answer to your question as part of an older thread. Hence, it is also worth browsing/ searching the mail archive.

If you feel like the plugin is missing a feature or has a defect, you can fill a feature request or bug report in our issue tracker. When creating a new issue, please provide a comprehensive description of your concern. Especially for fixing bugs it is crucial that the developers can reproduce your problem. For this reason, entire debug logs, POMs or most preferably little demo projects attached to the issue are very much appreciated. Of course, patches are welcome, too. Contributors can check out the project from our source repository and will find supplementary information in the guide to helping with Maven.

1.1.3 Examples

Have a look at the PDF version of this web site.

The following examples show how to use the PDF Plugin in more advanced usecases:

- Filtering Document Descriptor
- Site Phase Integration
- Configuring reports

2 Filtering Document Descriptor

2.1 Filtering Document Descriptor

The document descriptor (aka src/site/pdf.xml) could be filtered by System properties, Maven project properties and some date properties.

Expression Samples	Description
${JAVA_HOME}	The JAVA_HOME environment value.
${project.name}	The project name defined in <name/> tag in the pom.xml.
${project.developers[0].email}	The email of the first developed defined in <developers/> tag in the pom.xml.
${date}	The current date displayed in ISO-8601 format (i.e. `yyyy-MM-dd`), for instance `2009-06-22`.
${time}	The current time displayed in ISO-8601 format (i.e. `HH:mm:ss'Z`), for instance `12:26:48Z`.
${dateTime}	The current dateTime displayed in ISO-8601 format (i.e. `yyyy-MM-dd'T'HH:mm:ss'Z`), for instance `2009-06-22T12:24:17Z`.
${year} ${month} ${day}	The single date informations.
${hour} ${minutes} ${second}	The single time informations.

2.2 Example

For instance, if you have defined the following pom.xml and pdf.xml:

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <version>1.0-SNAPSHOT</version>
  <name>Your project</name>
  ...
  <developers>
    <developer>
      <email>your@email.com</email>
      ...
    </developer>
  </developers>
  ...
</project>
```

```
<document xmlns="http://maven.apache.org/DOCUMENT/1.0.1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/DOCUMENT/1.0.1 http://maven.apache.org/xsd/document-1.0.1.x
  outputName="maven-pdf-plugin-doc-${project.version}">
  <meta>
    <title>User guide of ${project.name} version ${project.version}</title>
    <author>${project.developers[0].email}</author>
  </meta>
  <toc name="Table of Contents">
    ...
  </toc>
  <cover>
    <coverdate>${date}</coverdate> <!-- current date in ISO 8601 format -->
    <!-- <coverdate>${day}/${month}/${year}</coverdate> current date in French format -->
    ...
  </cover>
</document>
```

The title will be `User guide of Your project version 1.0-SNAPSHOT` and the author will be `your@email.com`.

3 Limitations

3.1 Known Bugs and Limitations

Just a brief selection...

3.1.1 Current Limitations

- Current prerequisite is Maven >= 2.0.6. It's using Doxia-1.1.x via the shade-plugin (see MNG-3402).

3.1.2 Missing Features

- Menu sub-items are not supported in TOC (every source document starts a new chapter).

3.1.3 Implementation Specific Issues

These are not limitations of the plugin itself, but are listed here for completeness.

- Apache FOP issues
  - Table widths are always uniformly distributed.
  - Identical id attributes (eg anchors) within one document will lead to a build failure.
- iText issues

4 FAQ

4.1 Frequently Asked Questions

General

Specific problems

4.2 General

Is it possible to create a book?

The Doxia Book code currently only supports the iText module for generating a pdf book.

[top]

What graphics formats are supported?

You can use the same graphics formats as are supported by the chosen implementation, eg see Apache FOP Graphics Formats and iText Images. You should probably take care of image resolution, see bellow.

[top]

4.3 Specific problems

Why does my image not fit on the page?

This is most likely a resolution problem, for instance your image was saved with a 72 dpi resolution. Try to use an image with a higher resolution, like 96 dpi. You could resize your image whith this program: gimp. This is the only solution if you include the image from an apt source file (since in apt there is no possibility to specify the size of an image), if you are using xdoc, you may additionally indicate the size of the image using the width/height attributes of the `img` tag.

[top]

How can I center/in-line my image?

If you are using apt then your images will always be block-level elements, ie they will get centered in a separate paragraph. Apt does not support in-line images.

Using xdoc you are more flexible. By default a simple `<img>` tag can be used for an in-line image, eg:

```
<p>
  Here's a little icon: <img src="image.jpg"/> inside my text.
</p>
```

If you want your image centered you may put it explicitly inside a centered paragraph:

```
<p align="center">
  <img src="image.jpg"/>
</p>
```

or you may use the Doxia-specific class attribute in a surrounding `<div>` block:

```
<div class="figure">
  <img src="image.jpg"/>
</div>
```

[top]