



# **VDPMill User Guide**

# RenderX VDPMill User Guide

Copyright © 2009 RenderX Inc. All rights reserved.

This documentation contains proprietary information belonging to RenderX, and is provided under a license agreement containing restrictions on use and disclosure. It is also protected by international copyright law.

Because of continued product development, the information contained in this document may change without notice. The information and intellectual property contained herein are confidential and remain the exclusive intellectual property of RenderX. If you find any problems in the documentation, please report them to us in writing. RenderX does not warrant that this document is error-free.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means - electronic, mechanical, photocopying, recording or otherwise - without the prior written permission of RenderX.

## RenderX

Telephone: 1 (650) 328-8000

Fax: 1 (650) 328-8008

Website: <http://www.renderx.com>

Email: [support@renderx.com](mailto:support@renderx.com)

This product includes L2FProd.com Common Components licensed under the Apache License, Version 2.0 Copyright © 2005-2007 L2FProd.com. All rights reserved.

URL: <http://common.L2FProd.com>

See doc/APACHE-LICENSE-2.0.txt or <http://www.apache.org/licenses/LICENSE-2.0> for the specific language governing permissions and limitations under the License.

# Table of Contents

|   |    |
|---|----|
| Preface .....   | v  |
| 1. What's in this Document? .....   | v  |
| 2. Prerequisites .....  | v  |
| 3. Acronyms .....   | vi |
| 4. Technical Support .....  | vi |
| 1. Overview .....   | 7  |
| 1.1. Introduction .....   | 7  |
| 1.1.1. VDPMill Package Content .....  | 7  |
| 1.2. Uninstalling VDPMill .....   | 8  |
| 2. VDPMill Configuration .....  | 9  |
| 2.1. VDPMill Configuration Files .....  | 9  |
| 2.2. <code>factory.conf</code> Description .....                              | 9  |
| 2.3. <code>splitter.conf</code> Description .....                             | 10 |
| 3. VDPMill Document Processing .....  | 13 |
| 3.1. Processing Basics .....  | 13 |
| 3.2. Splitting .....  | 14 |
| 3.2.1. When Splitting is Useful .....   | 14 |
| 3.2.2. Splitting Basics .....   | 14 |
| 3.2.3. How to Set Splitting Start Point and Split Points in an XML File ..... | 16 |
| Defining a Splitting Start Point .....  | 16 |
| Defining Split Point .....  | 16 |
| 3.2.4. Splitter 2g - Double Pass Splitter .....                               | 16 |
| 3.2.5. Problems Caused by Splitting .....                                     | 19 |
| 3.2.6. Examples .....   | 19 |
| 3.3. Formatting and XSLT .....  | 20 |
| 3.4. Joining XEPOUT Files .....   | 20 |
| 3.4.1. Problems Caused by Joining .....                                       | 22 |
| 3.5. Custom Postprocessing .....  | 22 |
| 3.6. Generation .....   | 22 |
| 4. VDPMill GUI .....  | 23 |
| 4.1. What Is VDPMill GUI? .....   | 23 |
| 4.2. Launching VDPMill GUI .....  | 23 |
| 4.3. Processing Files Using VDPMill GUI .....                                 | 23 |
| 4.3.1. Adding Files .....   | 23 |
| 4.3.2. Processing Files .....   | 24 |
| 4.4. Configuring VDPMill Using GUI .....                                      | 25 |
| 5. VDPMill Command Line Interface .....                                       | 29 |
| 5.1. Using VDPMill from the Command Line .....                                | 29 |
| Index .....   | 31 |



# Preface

## 1. What's in this Document?

The RenderX User Guide provides background information about what VDPMill does and explains how to use the product. The manual consists of the following sections:

1. Overview
2. VDPMill Configuration
3. VDPMill Document Processing
4. VDPMill GUI
5. VDPMill Command Line Interface

## 2. Prerequisites

VDPMill runs on most systems where Java Virtual Machine 1.6 or newer is available. This includes:

- Unixes;
- Microsoft Windows;
- Linux;
- Mac OS X;
- Other platforms and operating systems.

VDPMill requires Sun Java VM 1.6 or higher to run properly.

VDPMill supports XSL-FO formatting via RenderX XEP engine, the required version is 4.16 or higher.

RenderX EnMasse is needed to provide high-speed formatting, the required version is 2.3 or higher.

A PDF viewer is required in order to view PDF output. Adobe provides a free one which can be downloaded from [Adobe](#).

To view PostScript files, one option is to use [GhostView](#) which may be used for viewing PDF as well. Versions are available for most operating systems.

---

### 3. Acronyms

The following table lists acronyms used in this manual:

**Table 1. Acronyms**

| Acronym | Full Term   |
|---------|---|
| CLI     | Command Line Interface                            |
| GUI     | Graphical User Interface                          |
| PI      | XML Processing Instruction                        |
| URL     | Uniform Resource Locator (World Wide Web address) |
| W3C     | World Wide Web Consortium                         |
| XML     | eXtensible Markup Language                        |
| XSL     | eXtensible Stylesheet Language                    |
| XSL-FO  | eXtensible Stylesheet Language Formatting Objects |
| XSLT    | XSL Transformation                                |
| XEPOUT  | XEP intermediate OUTput format                    |

### 4. Technical Support

You can contact RenderX technical support by:

- Using the RenderX support portal at <http://www.renderx.com/support/index.html>
- Sending an email to [support@renderx.com](mailto:support@renderx.com)
- Calling 1 (650) 328-8000

# Chapter 1. Overview

## 1.1. Introduction

This section contains introductory information about VDPMill.

VDPMill is a library of Java classes for splitting, collecting XML data and converting it to printable formats, such as PDF, PostScript, etc.

The VDPMill concept is to build a complete solution for small to large businesses for collecting, composing and creating print packages of content to address batch print for mailing and distribution, as well as creating of scheduled volumes of information for archive or other forms of distribution such as email.

The document processing steps are described in [Chapter 3, VDPMill Document Processing](#)

The VDPMill distribution includes a GUI-based tool suitable for users that prefer graphical user interface. For a detailed description, see [Chapter 4, VDPMill GUI](#). VDPMill can be run from the command line as described in [Chapter 5, VDPMill Command Line Interface](#). VDPMill settings can be customized according to a user's preferences. For a detailed reference, please refer to [Chapter 2, VDPMill Configuration](#).

There is a command-line interface (CLI) for VDPMill which provides full functionality and is suitable for users who prefer Unix-like systems. It also can be used to call VDPMill from shell/batch scripts and Makefiles.

### 1.1.1. VDPMill Package Content

VDPMill package includes the following:

`vdpmill(.bat)`, `vdpmill-gui(.bat)`<sup>1</sup>

VDPMill CLI and GUI running scripts respectively (see [Chapter 5, VDPMill Command Line Interface](#) and [Chapter 4, VDPMill GUI](#) for more details).

`conf/`

VDPMill configuration files (see [Chapter 2, VDPMill Configuration](#) for details).

`doc/`

VDPMill User Guide and VDPMill API reference (in the `api/` subdirectory).

`lib/`

VDPMill JAR files.

---

<sup>1</sup> Everywhere throughout this guide, extensions are shown in brackets for batch files for Windows-based computer systems; the names without these extensions refer to shell files for Unix- and Linux-based systems and Mac OS X.

`samples/`

VDPMill API use samples.

## 1.2. Uninstalling VDPMill

To uninstall VDPMill, remove the VDPMill installation directory and all its content.

# Chapter 2. VDPMill Configuration

## 2.1. VDPMill Configuration Files

VDPMill includes the following configuration files:

- `factory.conf` - Common configuration parameters. Includes common options and links to component configuration files.
- `splitter.conf` - Input XML file splitting rules. Contains information about splitting start point and split point settings.

## 2.2. `factory.conf` Description

**Table 2.1.** `factory.conf` file options

| Name                          | Possible Values                              | Default Value    | Description  |
|-------------------------------|--|------------------|--|
| <code>log-level</code>        | <code>error, warning, info, log, none</code> | <code>log</code> | Sets the log-level. <code>log</code> means that all messages (there are four types of messages: <code>log</code> , <code>information</code> , <code>error</code> , and <code>warning</code> ) are displayed or written to the log, <code>info</code> means everything except log messages, <code>warning</code> means errors and warnings, <code>error</code> means errors only, and <code>none</code> means no logging. |
| <code>max-threads</code>      | positive integer                             | 20               | The maximum number of simultaneous formatting threads in VDPMill.  |
| <code>data-buffer-size</code> | positive integer                             | 1048576          | The buffer size for I/O operations on files and sockets.   |
| <code>tmpdir</code>           | existing folder name                         | no default value | A folder to store temporary files. <sup>2</sup>  |

<sup>2</sup> Relative URLs are resolved relative to the `factory.conf` file location. For example, if the configuration file, `C:\VDPMill\conf\factory.conf`, contains a `../../../../splitter.conf` URL, the path to `splitter.conf` will be resolved relative to the `factory.conf` URL. `factory.conf`'s URL is `file:/C:/VDPMill/conf/factory.conf`. After the resolving, the absolute URL of `splitter.conf` will be `file:/C:/splitter.conf`.

| Name                          | Possible Values  | Default Value | Description   |
|-------------------------------|------------------|---------------|---|
| <code>toaster-attempts</code> | positive integer | 5             | The number of attempts to format document if network error occurred to format document via EnMasse. |

Table 2.2. `factory.conf` elements

| Element Name                | Description   |
|-----------------------------|---|
| <code>local-enmasse</code>  | Local EnMasse configuration. <sup>3</sup> The <code>configuration</code> attribute is the URL of the <code>toaster.conf</code> file. <sup>2</sup>   |
| <code>remote-enmasse</code> | Remote EnMasse configuration. The <code>host</code> attribute is the hostname EnMasse runs on. The <code>port</code> attribute is a TCP port EnMasse runs on. <code>format</code> is the EnMasse output format (the possible value is <code>xep</code> ). |
| <code>xep</code>            | XEP configuration. The <code>configuration</code> attribute is the URL of the <code>xep.xml</code> file. <sup>2</sup>   |
| <code>formatter</code>      | VDPMill formatter options. The <code>type</code> attribute is the formatter type, the possible values are <code>xep</code> <sup>4</sup> and <code>enmasse</code> .  |
| <code>generator</code>      | VDPMill generator options. The <code>type</code> attribute is the generator type. The possible value is <code>xep</code> . <sup>5</sup>   |
| <code>splitter</code>       | Splitter configuration. The <code>rules</code> attribute is an URL of the <code>splitter.conf</code> rules file. <sup>2</sup>   |

## 2.3. `splitter.conf` Description

`splitter.conf` defines splitter type and input XML document splitting rules, split start type and split point type to be used.

The splitter type can be the following:

- `splitter1g` - Single pass splitter. For a detailed description, see [Section 3.2.2, “Splitting Basics”](#).
- `splitter2g` - Double pass splitter. For a detailed description, see [Section 3.2.4, “Splitter 2g - Double Pass Splitter”](#).

<sup>3</sup> For EnMasse configuration see [EnMasse User Guide](#).

<sup>4</sup> `xep` options can be found in [XEP Configuration](#).

<sup>5</sup> Options must be written as `format:com.renderx.xep:OPTION_NAME_FROM_XEP`, e.g. `pdf:com.renderx.xep.DROP_UNUSED_DESTINATIONS`. For XEP options, see [XEP Output Formats](#).

The split start type can be the following:

- *pi* - Split start is set as PI.
- *element* - Split start is set as XML element definition.

The split point type can be the following:

- *pi* - Split point is set as PI.
- *element-count* - Split point is set as element count.

The splitting rules are explained in the table below

**Table 2.3.** `splitter.conf`

| Element Name                           | Description  | Example  |
|--|--|--|
| <code>splitter</code>                  | Defines the splitter type by the <code>type</code> attribute, the split start point type by the <code>split-start-type</code> attribute and the split point type by the <code>split-point-type</code> attribute. | <code>&lt;splitter xmlns="http://www.renderx.com/DF/splitter/config" type="splitter1g" split-start-type="pi" split-point-type="element-count"&gt;</code> |
| <code>start-point-pi</code>            | Start point PI should be specified by two attributes: <code>target</code> and <code>data</code> .  | <code>&lt;start-point-pi target="xepx-df-split" data="start"/&gt;</code>   |
| <code>start-point-element</code>       | Start point element should be specified by two attributes: <code>name</code> and <code>namespace</code> .  | <code>&lt;start-point-element name="page" namespace="http://www.renderx.com/XEP/xep"/&gt;</code>   |
| <code>split-point-pi</code>            | Split point PI should be specified by two attributes: <code>target</code> and <code>data</code> .  | <code>&lt;split-point-pi target="xepx-df-split" data="here"/&gt;</code>  |
| <code>split-point-element-count</code> | Split point element count should be specified by the <code>element-count</code> attribute.   | <code>&lt;split-point-element-count element-count="10"/&gt;</code>   |



# Chapter 3. VDPMill Document Processing

## 3.1. Processing Basics

The logical flow of document processing can be divided into four phases:

- **Splitting** – VDPMill Splitter splits an XML file into multiple XML files according to the splitting configuration.
- **XSLT Transformation** - XSLT transformation is done, if the document being processed contains a reference to an XSLT stylesheet or an external stylesheet is specified.
- **Formatting** - XSL-FO is fed into the formatter, which creates and fills pages according to the specification defined in the XSL-FO document. The results of the formatting stage is XEPOUT files to be processed later.
- **Joining** - VDPMill Joiner concatenates XEPOUT files into one large file in the specified order.
- **Custom Postprocessing** - User-defined postprocessing of XEPOUT intermediate document before the Generation phase.
- **Generation** - XEPOUT file is converted to the requested output format supported by RenderX XEP. For more detailed information, see [XEP User Guide](#).

Splitting divides a file into smaller chunks. The content and size of the chunks depend on the source file and the splitting rules. The split chunks can be formatted and then delivered in a printable format. They can be joined after the formatting into one large file. Splitting large files helps to avoid out of memory errors on large file formatting processes.

XSLT transforms a user-defined XML document to the XSL-FO format. Several tasks can be done in this stage of processing (adding barcodes to a document, etc).

Formatting converts XSL-FO files to the XEPOUT format. If the source file contains a reference to a stylesheet, or an external stylesheet specified (in the GUI, from the command line, or via VDPMill API), XSLT is done before formatting. An external stylesheet takes precedence over an embedded or associated stylesheet.

The Joining functionality can be used to get one large output file after Splitting and Formatting. It also can be used to concatenate XEPOUT files from other sources.

Custom Postprocessing can be used to customize a formatted document before generation (adding watermarks to a document, etc).

Generation makes printable documents (PDF, PostScript) from XEPOUT files. It can be done on small XEPOUT chunks as well as on a large file output from the Joining process.

## 3.2. Splitting

### 3.2.1. When Splitting is Useful

Large files can cause out of memory errors during formatting. Splitting allows dividing a large file into small ones and formatting them simultaneously, which makes formatting process faster.

An XML file may contain collected information about similar but different things. It can be split to be delivered as a collection of a separate printable files.

### 3.2.2. Splitting Basics

An XML file contains two parts:

- Common parts - content that should appear in each split chunk.
- Parts to be split - content that should be split into chunks.

For example, if there is an XSL-FO file like this:

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>
  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA1 </fo:block>
      <fo:block> DATA2 </fo:block>
      <fo:block> DATA3 </fo:block>
      <fo:block> DATA4 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

The common parts in that file are:

- Header - the content from the beginning of the file to the `<fo:flow>` start tag included;
- Footer - the content from the `</fo:flow>` end tag to the end of the file;

and the part to be split (split area) contains the `fo:block` elements with their content.

Let's introduce two splitting terms:

1. Splitting start point - a place in an XML file which separates the head common part from the split area.
2. Split points - places where data in split area can be separated.

In the example above, the splitting start point lies after the `<fo:flow>` start tag. Split points can be placed between `<fo:block>` elements. For example, if four chunks are needed, it means that each chunk will contain one `fo:block` element, and the split files will be the following:

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>
  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA1 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

...

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>
  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA4 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

**!** Currently, no document common footer part is supported. Only end tags can be repeated in split files.

**!** The splitting start point and split points should be on the same level in an XML file, if the single pass splitter (Splitter 1g) is used.

### 3.2.3. How to Set Splitting Start Point and Split Points in an XML File

VDPMill provides two ways to define a splitting start point, as well as two ways to define split points. VDPMill Splitter should be configured properly. See [Section 2.3, “splitter.conf Description”](#) for details.

#### Defining a Splitting Start Point

A splitting start point can be defined in the XML files as:

- PI
- Start tag

#### Defining Split Point

A split point can be defined in an XML files as:

- PI
- Number of elements (element count) that need to be included in each chunk

### 3.2.4. Splitter 2g - Double Pass Splitter

Splitter 2g can be used to split an XML file which contains split points on mixed levels in the document. It works similarly to the single pass splitter - Splitter 1g (see [Section 3.2.2, “Splitting Basics”](#)) - except that the document can be split into chunks constructed from different level data. Here is an example:

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>

  <?xepx-df-split start?>

  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA1 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

```

    </fo:flow>
  </fo:page-sequence>

  <?xepx-df-split here?>

  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA2 </fo:block>
    </fo:flow>
  </fo:page-sequence>

  <?xepx-df-split here?>

  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA3 </fo:block>
      <fo:block> DATA4 </fo:block>
      <fo:block> DATA5 </fo:block>

      <?xepx-df-split here?>

      <fo:block> DATA6 </fo:block>
      <fo:block> DATA7 </fo:block>

    </fo:flow>
  </fo:page-sequence>
</fo:root>

```

The chunks generated from the example above will be the following:

```

<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>

  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA1 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>

```

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>

  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA2 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>

  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA2 </fo:block>
      <fo:block> DATA4 </fo:block>
      <fo:block> DATA5 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

```
<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>

  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <fo:block> DATA6 </fo:block>
    </fo:flow>
  </fo:page-sequence>
</fo:root>
```

```

    <fo:block> DATA7 </fo:block>
  </fo:flow>
</fo:page-sequence>
</fo:root>

```

### 3.2.5. Problems Caused by Splitting

Cross-references in the source file can cause problems, if the referenced destination does not lie in the same chunk as the reference itself.

### 3.2.6. Examples

Consider the XSL-FO example file from [Section 3.2.2, “Splitting Basics”](#). The file should be split into chunks containing three `<fo:block>` elements each. In this case, splitting start point can be defined in that file as a processing instruction and a split point should be defined after the third `<fo:block>`.

```

<?xml version="1.0"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master master-name="all-pages">
      <fo:region-body region-name="xsl-region-body" margin="0.7in"
        column-gap="0.25in" border="0.25pt solid gray" padding="6pt"/>
    </fo:simple-page-master>
  </fo:layout-master-set>
  <fo:page-sequence master-reference="all-pages">
    <fo:flow flow-name="xsl-region-body">
      <?xepx-df-split start?> ❶
      <fo:block> DATA1 </fo:block>
      <fo:block> DATA2 </fo:block>
      <fo:block> DATA3 </fo:block> ❷
      <fo:block> DATA4 </fo:block>
      ❸
    </fo:flow>
  </fo:page-sequence>
</fo:root>

```

- ❶ Splitting start point
- ❷ Split point
- ❸ Implicit end of split area

The appropriate splitter configuration file should be the following:

```

<?xml version="1.0" encoding="utf-8"?>
<splitter xmlns="http://www.renderx.com/DF/splitter/config" type="splitter1g"
  split-start-type="pi" split-point-type="element-count">
  <start-point-pi target="xepx-df-split" data="start" /> ❶

```

```

<start-point-element name="page" namespace=""/>
<split-point-pi target="xepx-df-split" data="here"/>
<split-point-element-count element-count="3"/> ❷
</splitter>

```

- ❶ Defined the splitting start point as a processing instruction.
- ❷ Defined the split point as three elements in each split chunk.

### 3.3. Formatting and XSLT

VDPMill uses RenderX XEP and EnMasse to format XSL-FO files.

VDPMill supports XSLT via the `<?xml-stylesheet ... ?>` processing instruction or via explicit stylesheet specification. If an XML file contains reference to a stylesheet or a stylesheet is specified explicitly, the XML file will be transformed before formatting. `<?xml-stylesheet ... ?>` will be ignored, if an external stylesheet is specified.

The use of EnMasse turns on parallel formatting implicitly. Each file passed to VDPMill Formatter will be processed simultaneously via EnMasse.

**!** The EnMasse instance used by VDPMill Formatter should be configured to run as Toaster and to produce XEPOUT files. See the EnMasse documentation for details. Running the EnMasse instance used by VDPMill on the same physical machine is the best option, because this ensures correct processing of document external resources (stylesheets, images) which appear in the documents as relative URIs.

### 3.4. Joining XEPOUT Files

Some printers accept large files (PostScript, AFP) and process them fast. Large printable files can be generated from XEPOUT chunks by VDPMill Joiner. It simply concatenates XEPOUT files by adding the pages sequence from the second file to the first and so on. Joining of the following files:

```

<?xml version="1.0" encoding="UTF-8"?>

<xep:document xmlns:xep="http://www.renderx.com/XEP/xep" producer="XEP 4.13"
  creator="Unknown" author="Unknown" title="Untitled">

  <xep:page width="576000" height="792000" page-number="1" page-id="1">
    <xep:word-spacing value="0"/>
    <xep:letter-spacing value="0"/>
    <xep:font-stretch value="1.0"/>
    <xep:font family="Helvetica" weight="400" style="normal" variant="normal"
      size="12000"/>
    <xep:gray-color gray="0.0"/>
    <xep:text value="D" x="56400" y="725334" width="8184"/>
  </xep:page>
</xep:document>

```

```

...
</xep:page>
</xep:document>

```

```

<?xml version="1.0" encoding="UTF-8"?>

<xep:document xmlns:xep="http://www.renderx.com/XEP/xep" producer="XEP 4.13"
  creator="Unknown" author="Unknown" title="Untitled">

  <xep:page width="576000" height="792000" page-number="1" page-id="1">
    ...
  </xep:page>
</xep:document>

```

```

<?xml version="1.0" encoding="UTF-8"?>

<xep:document xmlns:xep="http://www.renderx.com/XEP/xep" producer="XEP 4.13"
  creator="Unknown" author="Unknown" title="Untitled">

  <xep:page width="576000" height="792000" page-number="1" page-id="1">
    ...
  </xep:page>
</xep:document>

```

will result in:

```

<?xml version="1.0" encoding="UTF-8"?>

<xep:document xmlns:xep="http://www.renderx.com/XEP/xep" producer="XEP 4.13"
  creator="Unknown" author="Unknown" title="Untitled">

  <xep:page width="576000" height="792000" page-number="1" page-id="1">
    <xep:word-spacing value="0"/>
    <xep:letter-spacing value="0"/>
    <xep:font-stretch value="1.0"/>
    <xep:font family="Helvetica" weight="400" style="normal" variant="normal"
      size="12000"/>
    <xep:gray-color gray="0.0"/>
    <xep:text value="D" x="56400" y="725334" width="8184"/>
    ...
  </xep:page>

  <xep:page width="576000" height="792000" page-number="1" page-id="1">
    ...
  </xep:page>

```

```
<xep:page width="576000" height="792000" page-number="1" page-id="1">
...
</xep:page>
</xep:document>
```

### 3.4.1. Problems Caused by Joining

The main problem of joining is page numbering. VDPMill Joiner doesn't adjust page numbers in joined chunks to allow cross-references working.

## 3.5. Custom Postprocessing

XEPOUT document postprocessing can be done using VDPMill API. The user-defined post-processor has a reference to the final format generator's SAX content handler and receives SAX events of the XEPOUT document which is being processed. See VDPMill API samples and VDPMill API reference for more details.

## 3.6. Generation

The Generation process makes printable files from XEPOUT format. Supported output formats depend on the generator used. Currently the XEP generator is supported. See XEP User Guide for more details.

# Chapter 4. VDPMill GUI

## 4.1. What Is VDPMill GUI?

VDPMill contains a user-friendly GUI tool called the VDPMill GUI. Use of VDPMill GUI simplifies whole functionality of VDPMill.

VDPMill GUI look and feel varies depending on the operating system the application is running on.

## 4.2. Launching VDPMill GUI

To open VDPMill GUI, launch the `vdpmill-gui` script (the `vdpmill-gui.bat` batch file for MS Windows) located in the VDPMill installation directory.

## 4.3. Processing Files Using VDPMill GUI

### 4.3.1. Adding Files

To process a file, first of all you must add it to the list of input files.

**To add an existing XML or XSL-FO file:**

1. From the main menu, click **File**.

The **File** menu is displayed.


2. From the **File** menu, click **Add ...** or press **Ctrl+O**.

The file open dialog box is displayed.

3. Browse for the files you wish to add.

The files are added to the **Input Files** list within VDPMill GUI.

If you need to exclude files, select the appropriate checkboxes in the list of input files

and click the  **Remove** button on toolbar or select **Remove Selected** from the **File** menu.

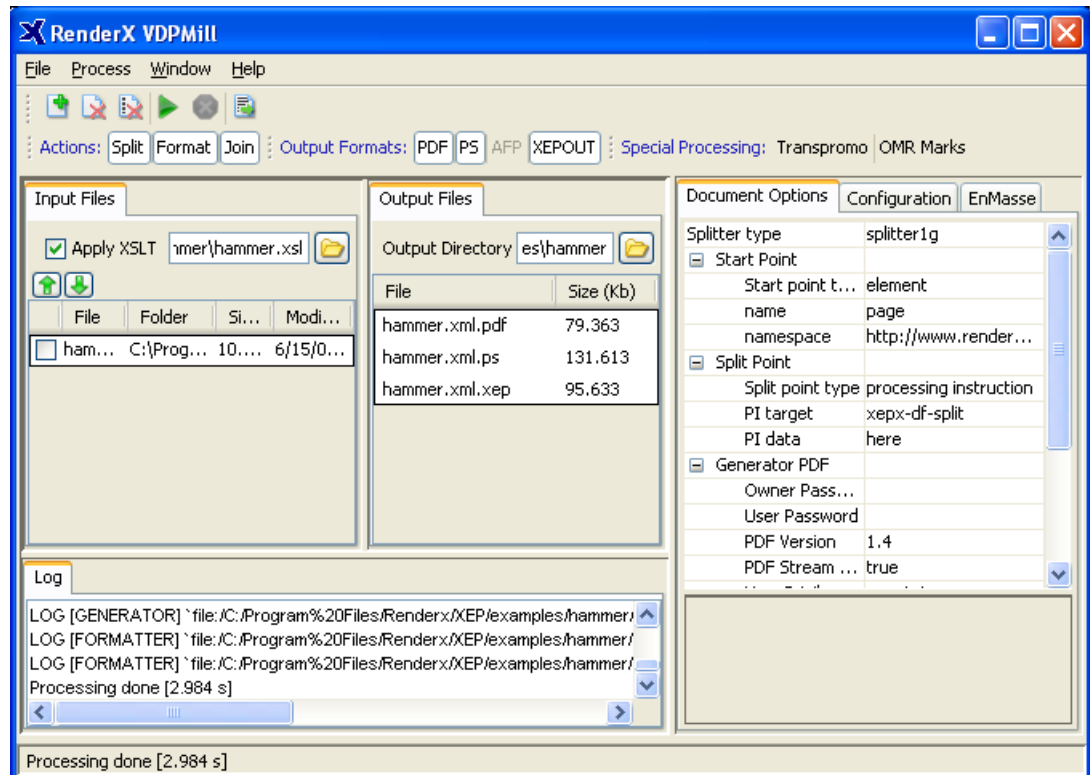






Figure 4.1. XML file displayed in VDPMill GUI

### 4.3.2. Processing Files

Once the files are added, they can be processed with VDPMill and output files will be generated in the chosen output format(s).

#### To process XML files:

1. Select **Apply XSLT** and specify an XSLT stylesheet (use the  **Browse...** button next to the text field), if the input files should be transformed before formatting. If any file contains a reference to a stylesheet (via an XML PI), that reference will be ignored and the file will be transformed using the stylesheet specified.
2. Select the desired actions on the main toolbar above the **Input Files** list.
3. Select output format(s) for result files.
4. Choose the output directory by clicking the  **Browse...** button or by typing the full directory path in the **Output Directory** text box.

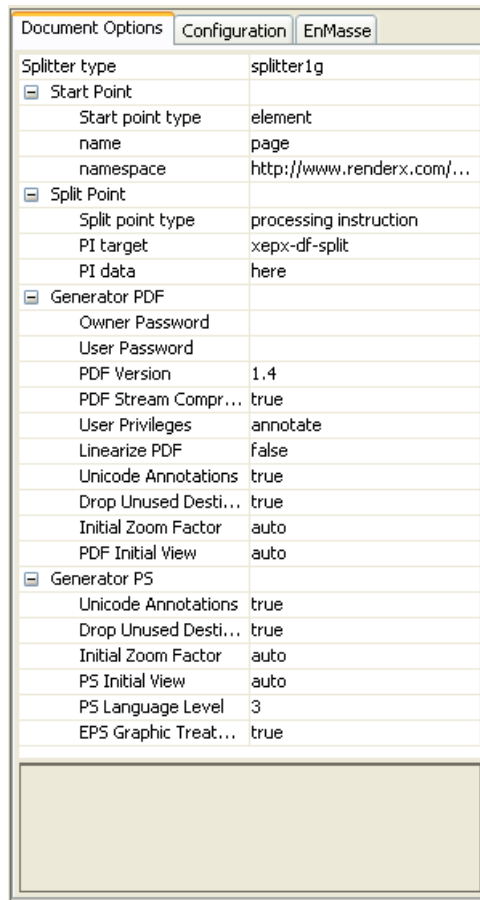
- Press the  **Process** button. The logging information will be displayed in the tab below the **Input Files** tab. When in progress, processing can be interrupted by clicking the  **Cancel** button.

## 4.4. Configuring VDPMill Using GUI

VDPMill can be configured using the GUI.

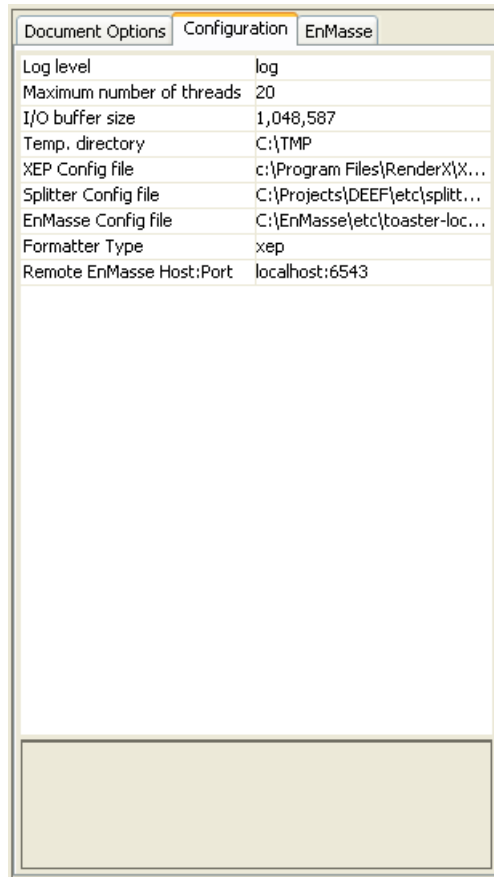
### To configure VDPMill:

- From the main menu select **Window**. The **Window** menu is displayed.
- Make sure that the **Configuration** menu item is checked.
- To change VDPMill document processing options, select the **Document Options** tab.



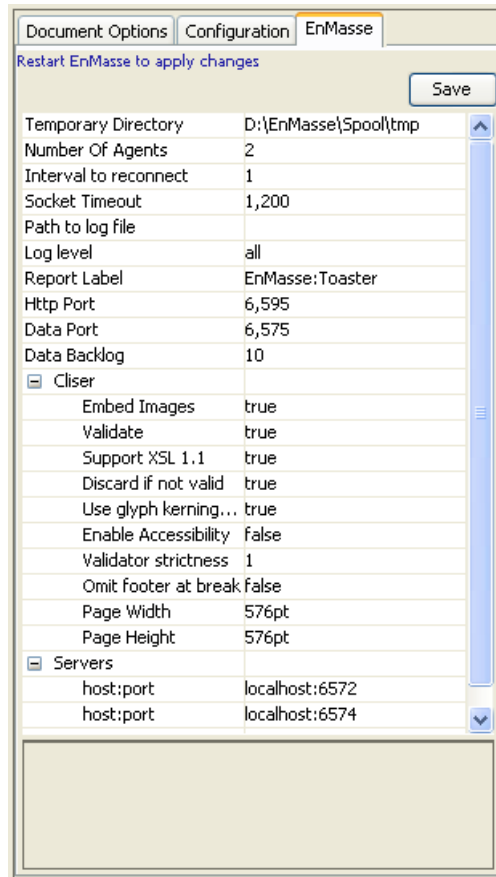
**Figure 4.2. Document Options**

- To change VDPMill core options, select the **Configuration** tab.



**Figure 4.3. Configuration**

5. To change EnMasse settings, select the **EnMasse** tab.



**Figure 4.4. EnMasse settings**

For a detailed description of VDPMill options and component types see [Chapter 2, VDPMill Configuration](#).



# Chapter 5. VDPMill Command Line Interface

## 5.1. Using VDPMill from the Command Line

The VDPMill command line syntax is the following:

```
Usage: com.renderx.df.Main -h|--help| [-d ]-c configfile -o output actions
      [-f format ] [ -s|--xsl stylesheet ] files
```

where:

- *-h, --help* - Print help and exit.
- *-d* - Turn the debug mode on. Prints a lot of information.
- *-c configfile* - Specifies the VDPMill configuration file. A relative path is resolved relative to the current directory.
- *-o output* - Output directory if many files are produced, or a single output file.
- *actions* - Actions to perform on the input file(s):
  - *split* - Split input file. This action (and all actions which contain *split*) don't accept multiple input files.
  - *split format* - Split input file and then format split chunks, generate output file(s) for selected output format(s), write output files in the output directory.
  - *split format join* - Split input file, format split chunks, concatenate formatted chunks and generate output file(s) for selected output format(s)
  - *format* - Format input file(s), generate output file(s) for selected output format(s)
  - *format join* - Format input files, join them and generate output file(s) for selected output format(s)
  - *join* - Join input XEPOUT files and generate output file(s) for selected output format(s)
- *-f format* - Output format. One of the formats supported by the generator selected in the generator configuration file. Can be specified multiple times to produce many output files from each formatted file. A PDF file will be produced if this parameter is omitted.
- *-s, --xsl stylesheet* - XSLT stylesheet to apply to input document(s) being formatted.
- *files* - Input files (split actions accept only one input file).



# Index

- Formatting, 13
- Generation, 13
- Joining, 13
- Postprocessing, 13
- Splitting, 13
- XSLT Transformation, 13

## A

- Acronyms, vi
- Adding an Existing XML or XSL-FO File, 23
- Adding Files, 23

## C

- Configuration, 9

## D

- Document Contents, v
- Document Processing, 13

## F

- Factory Configuration, 9
- Formatting, 20

## G

- Generation, 22
- GUI Tool, 23

## J

- Joining, 20

## P

- Package Content, 7
- Postprocessing, 22
- Prerequisites, v
- Processing files, 24
- Processing files with VDPMill GUI, 23
- Processing XML Files, 24

## S

- Splitter 2g, 16
- Splitter Configuration, 10
- Splitting, 14

## T

- Technical Support, vi

## U

- Uninstall, 8

## V

- VDPMill Command Line Interface, 29
- VDPMill GUI, 23
  - Access, 23
  - Add an XML or XSL-FO file, 23
  - Adding Files, 23
  - Configuration, 25
  - Open, 23
  - Processing files, 24
  - Processing files with VDPMill, 23
  - Processing XML Files, 24

## X

- XSLT, 20

