**Smooks Developer Tools Reference Guide** 



Version: 3.2.0.Beta

1. Introduction	1
1.1. Key Features of Smooks Tools	. 1
1.2. What is Smooks?	1
1.3. What is Smooks Tools?	2
1.4. Adding Smooks jars	2
2. Tasks	4
2.1. New Smooks Configuration File Creation	4
2.2. Input Task Configuring	5
2.3. "Java Mapping" or "Apply Template"?	7
2.4. Java Mapping Task	7
2.5. Apply Template Task	10
2.6. Smooks Configuration testing using Smooks Run Configuration	13
3. Reference	15
3.1. Process tab	15
3.1.1. Processing Task section	16
3.1.2. Selected Task Details Section	17
3.2. Options Tab	27
3.2.1. Smooks Configuration section	28
3.2.2. Filter Settings Filter section	28
3.3. Source Tab	29
3.3.1. XML Source Editor	29
3.3.2. Error underlining in Graphical Editor	30
3.3.3. Smooks Configuration File Validator	31
3.4. Properties View	33
3.4.1. Decode Configuration	34
3.4.2. Apply Template Wizard	36
4. Summary	43
4.1. Other relevant resources on the topic	43

# Introduction

This chapter gives you a short introduction to Smooks, Smooks tools and its installation.

First, have a look at the key features of Smooks tools:

# **1.1. Key Features of Smooks Tools**

Here, we provide you with a key functionality which is integrated in Smooks tools.

Table 1.1. Key Functionality for Smooks Tools

Feature	Benefit	Chapter
Smooks Configuration File Wizard	Smooks tools allows to create/edit the Smooks configuration file for Java2Java data transformation.	<u>Smooks</u> Configuration File Wizard
Smooks Editor	Smooks Editor helps configure the created Smooks configuration file.	Smooks Editor

# 1.2. What is Smooks?

Smooks is a Java Framework/Engine for processing XML and non XML data (CSV, EDI, Java, JSON etc). It provides:

- I. **Transformation**: Perform a wide range of Data Transforms. Supports many different Source and Result types -XML/CSV/EDI/Java/JSON to XML/CSV/EDI/Java/JSON.
- II. Java Binding: Bind into a Java Object Model from any data source (CSV, EDI, XML, Java, JSON etc).
- III. Huge Message Processing: Process huge messages (GBs) Split, Transform and Route message fragments to JMS, File, Database etc destinations. Route multiple message formats to multiple destinations in a single pass over a message.
- IV.Message Enrichment: Enrich a message with data from a Database, or other Datasources.
- V. **Combine**: Combine the above features in different ways e.g. add Message Enrichment as part of a Splitting and Routing process.



#### Figure 1.1. Smooks

For more informations about Smooks, please visit Smooks official site [http://www.smooks.org].

# 1.3. What is Smooks Tools?

Smooks tools is a set of graphical tools for editing Smooks configuration file based on Eclipse.

😵 smooks-config.xml 🛙	° 6
Smooks Processir	ng
Processing Tasks	
📦 🚯 Input Task	
Selected Task Details	3
Select the task node	
Process Options Source	

#### Figure 1.2. Smooks Form editor

# 1.4. Adding Smooks jars

During your development, you probably will be faced with the necessity to include some additional Smooks jars to your project. This problem can be solved in such a way:

- Create the folder named "lib" inside your project, if it doesn't exist yet. Copy all the Smooks jars you need to include into the *lib* directory.
- Right-click on the project and select Properties.
- Select the "Java Build Path" item in the Properties list, then the Libraries tab, and click "Add JARs".
- In the Jar Selection dialog, select all the jars in the "lib" directory you want to include and click OK.
- Now you should see all the jars included to your project hierarchy.

smooks-config.xml 🛛	- 8
Smooks Processing	
Processing Tasks	
Input Task	
Selected Task Details	
Select the task node	
Process Options Source	

Figure 1.3. Smooks Form editor

# Tasks

This chapter describes the main tasks a user can be faced during Smooks tools usage. In this chapter we use the example that can be downloaded from <u>here</u> [http://anonsvn.jboss.org/repos/jbosstools/trunk/smooks/docs/reference/xml-to-java.zip].

# 2.1. New Smooks Configuration File Creation

Select the project where you want to create new Smooks Configuration File and right-click on it, select in the menu *New > Other*, then find *Smooks > Smooks Configuration File*. Click the *Next* button.



## Figure 2.1. Selecting Smooks Configuration File Wizard

The wizard page is a file path creation page. Select the *src* folder to be the files container, and input the name *smooks-config.xml*. Click *Next*.

6				×
Smooks Configurat	ion File Wizard Pa	age		
Create a new Smook	s configuration file.			
				-
Enter or select the pa	erent folder:			
smooks_java_java/sr	rc			
⊽ 🥵 smooks_java_j	ava			
🗁 .settings				
👂 🗁 bin				
🗁 libs				
🕨 🚑 src				
File na <u>m</u> e: smooks-c	onfig.×ml			
Advanced >>				
?	< Back	<u>N</u> ext >	Cancel	Finish

## Figure 2.2. Choosing the configuration file container and the file name

# 2.2. Input Task Configuring

Input task configuring is an obligatory step for your smooks project creation. You can configure it on the Process page of the editor: look for the "Input Task" in the Process Map at the top of the page.

📽 smooks-config.xml 🕴	- 8
Smooks Processing	
Processing Tasks	
▲ Input Task	
Selected Task Details	
Process Options Source	

#### Figure 2.3. Input Task Configuring

Select it and you will see all the properties to set for the Input reader of your Smooks configuration. "Input type" corresponds to the type of data that you will be working with. For example, to work with incoming CSV (Comma-separated Values) data, you would specify "CSV" in the drop-down list. Each reader type has slightly different configuration details that must be set in the "Input configuration" area. For instance, the CSV reader requires you to specify details such as the encoding, quote character, separator character, and the list of incoming fields. The EDI reader requires the encoding and the path to the Mapping Model describing the incoming data. In the *Input data* section, you specify some sample data that conforms to your reader configuration.

Once you've specified your reader configuration and sample data, you can see the input model rendered in a tree form in the *Input model* section. On the picture below you can see the correct configuration of some XML input task where *input-message.xml* is set as an input data file.

▼ Input Type	✓ Input E	)ata		
Input Type: XMI	Select a s	ample	data file	
input typer	Туре		Path	Add
	🗹 🖹 inp	ut.xml	Workspace://xml-to-java/src/input-message.xml	
Input Configuration				Delete
	▼ Input N	1odel	View	
	Refresh			
	V e ord	ler		
	▽ 0	header		
		🖲 date	9	
	$\bigtriangledown$	🖲 cust	omer	
		@ n	umber	
	▽ 0	order-it	tems	
	$\bigtriangledown$	🖲 orde	er-item	
		e p	roduct	
		e q	uantity	
		e n	rice	

#### Figure 2.4. Input Task Configuring

# 2.3. "Java Mapping" or "Apply Template"?

Though there are many options in Smooks as far as what you can do with input data such as transformation, routing, and persistence, this version of the Smooks Configuration Editor focuses only on these areas: mapping to java and applying templates to create different output formats. If you have a set of Java classes you want to use the incoming data for, you can use the "Java Mapping" task to specify those classes and use drag and drop to map between the input model generated by the reader and elements in the output model. Or if you simply want to transform your output to one or more formats, you can use the "Apply Template" task to map it to a CSV file, XML or XSD file (and other formats in the future).



#### Note

Now you can't transform your output directly, using only Input and Template tasks. You should use Mapping as an interagent between these tasks.

## 2.4. Java Mapping Task

If you decide to do Java Mapping, you need to make sure that your Input reader has been set up and you have some sample data specified. Then you should select *Input Task* in the Process tab and click the plus (+) sign to the right of the icon. Select *Java Mapping* from the popup menu and it will appear to the right, connected to *Input Task*. Then select *Java Mapping* task.

📽 *smooks-config.xml 🕴	- 8
Smooks Processing	
Processing Tasks	
Input Task Java Mapping	
Selected Task Details	

## Figure 2.5. Java Mapping configuration

Another method of adding *Java Mapping* element to the canvas in the Processing Tasks section is to right click Input Task element and select *Java Mapping* in the popup menu.

🐮 *smooks	s-config.xml 🛙	
Smooks	Processing	
Processi	ng Tasks Add Task Delete Input Methods	Input       Java Mapping       Apply Template

## Figure 2.6. Java Mapping configuration

Right-click on the canvas in an empty space and select "Add ->Java Class".



## Figure 2.7. Java Mapping configuration

Java Bean Creation wizard appears. Specify a unique identifier for the new class, the class path. If the Java class is specified, you'll see a list of the properties in the box below. Click *Finish* when you're done. Now with the input and output models on the canvas, you can click and drag from the various input elements to corresponding output elements. Make sure to connect collection elements to corresponding collection elements. Finally your mapping should look nearly like the one on the picture below.



## Figure 2.8. Final Mapping schema

For details, also see the movie, <u>"XML to Java"</u> [http://www.screencast.com/users/tfennelly/folders/ Camtasia/media/a6648ba3-953f-40bf-8241-570306fba776].

If you are interested in transforming EDI to Java, please, follow the <u>link</u> [http://www.screencast.com/users/tfennelly/folders/Camtasia/media/a72704fb-ff74-4d5d-9869-9092611f52c2].

# 2.5. Apply Template Task

The "Apply Template" task works very similarly to the <u>"Java Mapping" task</u>, where you map between an input model and an output model. Select the Java Mapping task you want to use as the input model in the Process Map pane and click the plus (+) sign to the right of the icon.

¥ smooks-config.xml 🖾		- 8
Smooks Processing		
Processing Tasks	Java Mapping	

## Figure 2.9. Apply Template configuration

The Message Type Selection wizard will appear. In our example we will transfer our data to csv output format, so you should select *CSV* and click Next.

							×
ge Type	Selecti	on					
"Messa	ge Type"						
е Туре							
							٥
_		_					
	< <u>B</u> ack		<u>N</u> ext >		Cancel		Einish
	<b>je Type</b> "Messa e Type	ge Type Selection "Message Type" e Type e Type < Back	ge Type Selection "Message Type" . e Type < Back	ge Type Selection "Message Type" . e Type < Back <u>N</u> ext >	ge Type Selection "Message Type" . e Type < Back <u>N</u> ext >	ge Type Selection "Message Type" . e Type < Back <u>N</u> ext > Cancel	ge Type Selection "Message Type" . e Type < Back Next > Cancel

## Figure 2.10. Message Type Selection

On the next wizard page put the following string into the Fields, select Output Fields Names and click Finish.

		×
CSV Template Conf	iguration	
Configure CSV Templ string.	ate: Use ',' to separate fields in the CSV field	
Fields	productId,quantity,price	
Separator Character	: .	
Quote Character :	u .	
Output Field Names :		
(?) < <u>B</u> a	ck Next > Cancel Einish	
		_

#### Figure 2.11. CSV output message configuration

After these steps "Apply Template" task will appear to the right, connected to the task you created it from. To continue the process of configuration you should click on it and find *CSV Template* item with entered fields on the canvas.Now you can click and drag from various input elements to corresponding output elements in the template. Make sure to connect collection elements to the c orresponding *csv-record*. In our example we transfer into CSV output only the elements of orderItems collection (see the picture below):



Figure 2.12. Relations between input and output models

# 2.6. Smooks Configuration testing using Smooks Run Configuration

This option is intended to view the results of Smooks transforming procedure. To do the testing you should select your *Smooks Configuration file* you want to transfer in the Project Navigator or open it in the Smooks Configuration Editor and then select "Run As..." from the *Run* toolbar button or *Run->Smooks Run Configuration* option in the top menu bar. And Smooks Configuration file will be run automatically.



## Figure 2.13. Smooks Configuration testing

If any errors or warnings appear, they will pop up in a dialog. The output of the test will appear in the Console view. In our case the following streaming output will appear:

```
[Stream Templating Result ...]
|--
|"productId","quantity","price"
|"111","2","8.9"
|"222","7","5.2"
```

```
|--
[Java Mapping Results...]
  |--
  |> order (beanId = "order")
  > header (beanId = "header")
       > date = "2006-11-15 20:45:28.0 EET"
       > customerNumber = 123123L
       > customerName = "Joe"
   > orderItems (beanId = "orderItems")
        > example.model.OrderItem (beanId = "orderItem")
          > productId = 111L
          > quantity = 2I
          > price = 8.9D
       > example.model.OrderItem (beanId = "orderItem")
  I
          > productId = 222L
          > quantity = 7I
  I
          > price = 5.2D
  Ι
  |---
```

If the test runs but doesn't generate any streaming output the Console output will be the following:

Nothing to Display:

- No Java Mappings.
- No Templates Applied.

# Reference

This chapter includes detailed reference information about all tabs of the Smooks Configuration Editor.

# 3.1. Process tab

The Process tab of the Smooks Configuration Editor helps to configure different types of transformations. By default smooks configuration file is opened in this editor. If you have another default settings for editor opening you should left click smooks configuration file and select: *Open With->Smooks Configuration Editor*.

The Process tab has two sections:

- Processing Task section
- Selected Task Details section

You can see them on the picture below.

📽 smooks-config.xml 😫		- 8
Smooks Processing		
Processing Tasks		
	P	<u> </u>
Input Task	Java Mapping	~
Selected Task Details		
		$\widehat{}$
▼ Input Type	✓ Input Data Select a comple data file	
Input Type: XML	Contraction Select a sample data file	_
	V input.xm Workspa	ce:/
- Input Configuration		
Specify sample data in the Input Data section.	▼ Input Model View	=
	Refresh	
	🗢 🖻 order	
	🗢 🖻 header	
	e date	
	🗢 🖻 customer	
	(a) number	
	⊽ 🖻 order-items	
	⊽ 🖲 order-item	
	e product	
	e quantity	~
Process Options Source		

## Figure 3.1. Two Sections of the Process tab.

## 3.1.1. Processing Task section

Using the popup menu in the Processing Task section you can select which types of technologies(templating or mapping ones) you will use for transofmation:

The descriptions of the popup menu options are in the following table.

Option	Description	Default
Add Task	Select one of the following tasks according to the necessary type of Source and Result types of the files:	
	• <i>Input</i> - this task is required and appears automatically when Smooks config file is created. You should just configure it properly.	

Option	Description	Default
	Java Mapping	
	Apply Template	
Delete	Click this option if you want to delete some task from the section. Note:you can't delete input task because it's required.	
Input Methods	<ul><li>Choose one of the following methods:</li><li>System</li></ul>	System
	Simple	
	• Amharic(EZ+)	
	• Cedilla	
	• Cyrillic	
	Inuktitut	
	• IPA	
	Multipress	
	SCIM Bridge Input Method	
	SCIM Input Method	
	• Thai-Lio	
	<ul> <li>Tigrigna-Eritrean(EZ+)</li> </ul>	
	<ul> <li>Tigrigna-Ethiopian(EZ+)</li> </ul>	
	Vietnamese	
	X input Method	

## 3.1.2. Selected Task Details Section

The options of this section depends on the selected task in the Processing Task section. Because there are 3 types of tasks there are 3 different sets of its options in the Selected Task Details Section. They will be described one by one.

## 3.1.2.1. Selected Task Details Section for Input Task.

On the picture below you can find an example of Selected Task Details Section view if XML is selected as input type.

Selected Task Details	F
▼ Input Type	✓ Input Data Select a sample data file
Input Type: XML ♀	Type Path       Type     Path       Image: Constraint of the second seco
Specify sample data in the Input Data section.	✓ Input Model View <u>Refresh</u>
	⊽ e order ⊽ e header

## Figure 3.2. Selected Task Details Section for Input XML Task.

As you can see on the picture above Input Configuration section is empty for XML input file. But this section has special configuration options for CSV,EDI,JSON,Custom input files.

Here are the screens of these configuration options:

• CSV:

✓ Input Type	
Input Type: CSV	0
<ul> <li>Input Configuration</li> </ul>	
Fields :	3
Separator Character :	
Quote Character :	П
Record Name :	csv-record
Root Name :	csv-set
Indent :	
Skip Lines :	0

## Figure 3.3. Selected Task Details Section for Input CSV Task.

• EDI:

✓ Input Type Input Type: EDI	
▼ Input Configur	ation
Target Profile :	
Encoding :	UTF-8
Mapping Model :	Browse
Validate :	

## Figure 3.4. Selected Task Details Section for Input EDI Task.

• JSON:

• Input Type		
Input Type: JSON	:	
N		
<ul> <li>Input Configuration</li> </ul>		_
Target Profile :		^
Array Element Name :	element	
Encoding :	UTF-8	
Illegal Element Name Char Replacement :		
Indent :		
Key Prefix On Numeric :		=
Key Whitspace Replacement :		
Null Value Replacement :		
Root Name :	json	
Key Maps		
	New	
	Remove	
	Up	~

## Figure 3.5. Selected Task Details Section for Input JSON Task.

• Custom:

put Type: Custom	
Input Configuration	
rget Profile :	
rget Profile :	
h	
ass :	Browse
landlers	
	New
	Remove
	Up
	Down
	Properties
eatures	
	New
	Remove
	Up
	Down
	Properties
arameters	

## Figure 3.6. Selected Task Details Section for Input Custom Task.

All the input task configuration positions can be found in the table below:

Table 3.2. Selected Task Details Section. Optic	ons for Input Task.
---	---------------------

Option	Description	Default
Input type	<ul><li>Select your type of input file. If don't find your type in the list, you should use Custom type:</li><li>No Input</li></ul>	XML
	• XML	
	• Java	
	• XSD/WSDL	

Option	Description	Default
	• CSV	
	• EDI	
	• JSON	
	Custom	
Input	No Input - no info required	• CSV
configuration	• <i>XML</i> - no info required	<ul> <li>not defined</li> </ul>
	• Java - no info required	• ','
	XSD/WSDL - no info required	• '"'
	• CSV	csv-record
	• Fields - Comma separated list of CSV record field	csv-set
	names	• true
	• Separator Character - Field separator character.	• 0
	Quote Character - Quote character.	• EDI
	• Record Name - Name of csv record element.	<ul> <li>not defined</li> </ul>
	• Root Name - Name of csv root element.	• UTF-8
	<ul> <li>indent - Add indentation character data to the generated event stream. This simply makes</li> </ul>	<ul> <li>not defined</li> </ul>
	the generated event stream easier to read in its	• true
	serialized form. Useful for testing etc.	• JSON
	<ul> <li>Skip Lines - Number of lines to skip before processing starts</li> </ul>	<ul> <li>not defined</li> </ul>
	• FDI	element
	Target Profile - Defines the output transformation	
	profile	• not defined
	• Encoding - The character encoding.	
	Mapping Model - Defines the EDI Mapping Model	• Taise
	configuration for processing the EDI message	<ul> <li>not defined</li> </ul>
	processed by Smooks.	<ul> <li>not defined</li> </ul>
		<ul> <li>""(an empty string)</li> </ul>

Option	Description	Default
	<ul> <li>Validate - This attribute turns on/off datatype validation in the EDI Parser. Validation is on by default. It makes sense to turn datatype validation off on the EDI Reader if the EDI data is being bound into a Java Object model.</li> <li>JSON</li> </ul>	<ul> <li>'json'</li> <li>not defined</li> <li><i>Custom</i></li> <li>no defaults</li> </ul>
	Target Profile - Defines the output transformation profile	
	<ul> <li>Array Element Name - The element name of an array element.</li> </ul>	
	<ul> <li>Encoding - encoding: The default encoding of any JSON message InputStream processed by this Reader.</li> </ul>	
	<ul> <li>Illegal Element Name Char Replacement - If illegal characters are encountered in a JSON element name then they are replaced with this value. By default this is not defined, so that the reader doesn't doesn't search for illegal characters.</li> </ul>	
	• <i>Indent</i> - Add indentation character data to the generated event stream. This simply makes the generated event stream easier to read in its serialized form. Useful for testing etc.	
	• <i>Key Prefix on Numeric</i> - The prefix character to add if the JSON node name starts with a number. By default this is not defined, so that the reader doesn't search for element names that start with a number.	
	• <i>Key Whitespace Replacement</i> - The replacement character for whitespaces in a JSON map key. By default this not defined, so that the reader doesn't search for whitespaces.	
	<ul> <li>Null Value Replacement - The replacement string for JSON NULL values.</li> </ul>	
	<ul> <li>Root Name - The element name of the document root.</li> </ul>	

Option	Description	Default
	<ul> <li><i>Key Maps</i> - Defines a JSON element name mapping The "from" key will be replaced with the "to" key or the contents of the element.</li> <li><i>Custom</i></li> <li><i>Target Profile</i> - Defines the output transformation profile</li> <li><i>Class</i> - Custom reader class.</li> <li><i>Handlers</i> - Set a handler on the reader instance e.g. an EntityResolver, ErrorHandler etc.</li> <li><i>Features</i> - Reader Features List</li> </ul>	
	Parametres - Resource Parameters	
Input Data	You should select a data file using <i>Add</i> and <i>Delete</i> buttons	
Input Model View	Using this view you can see the structure of your input file.If the file has been changed, to see the changes click <i>Refresh</i> link.	

## 3.1.2.2. Selected Task Details section for Java Mapping Task.

Selected Task Details section for this task is presented by the graf, that lighten the process of java mapping.



## Figure 3.7. Selected Task Details Section for Mapping Task.

This graphical editor allows you to perform drag&drop operation with the nodes of transformed data to map the source data to target data. When you save the changes in the graphical editor the correct Smooks configuration file content will be generated.

Using the popup menu in the *Selected Task Details* section you can manage the diagram elements on the canvas.

The descriptions of the popup menu options are in the following table.

Option	Description	Default
Add	Select one of the following tasks:	
	• Java Class - adds bean context item to the config file. This option is available when no elements are selected and a user right click the canvas.	
	• <i>Expession Binding</i> - adds expression based binding to selected java binding element.	
	<ul> <li>Value Binding - adds Value binding (<jb:value>) to the selected java binding element.</jb:value></li> </ul>	

 Table 3.3. Popup menu. Selected Task Details section.

Option	Description	Default
	<ul> <li>Bean Binding - adds Wiring binding (<jb:wiring>) to the selected java binding element.</jb:wiring></li> </ul>	
Undo	By this option you can revert the changes made at the previous step.	
Redo	By this option you can redo the changes made at the previous step.	
Delete	This option is available only if you select some element on the canvas. Click this option if you want to delete the element from it.	
Auto Layout	Sets the default layout of the elements on the canvas.	
Properties	Click this option if you want to add <i>Properties view</i> to the current perspective. The just opened <i>Properties view</i> will automatically reflect the properties of the selected diagram element.	
Input Methods	<ul><li>Choose one of the following methods:</li><li>System</li><li>Simple</li></ul>	System
	Amharic(EZ+)	
	• Cedilla	
	• Cyrillic	
	• Inuktitut	
	• IPA	
	Multipress	
	SCIM Bridge Input Method	
	SCIM Input Method	
	• Thai-Lio	
	<ul> <li>Tigrigna-Eritrean(EZ+)</li> </ul>	
	<ul> <li>Tigrigna-Ethiopian(EZ+)</li> </ul>	
	Vietnamese	
	X input Method	

## **3.1.2.3. Selected Task Details section for Template Task.**

Selected Task Details section for this task is presented by the graf, that is similar to the one in the *previous section*.



## Figure 3.8. Selected Task Details Section for Template Task.

Popup menu similar to the one in <u>Selected Task Details section for Java Mapping Task</u> is also available here.

# 3.2. Options Tab

This section describes Options tab of the Smooks Configuration File editor, gives short recommendations how this tab can be used during the project configuring.

¥ smooks-config.xml 🛙	- 8
Options	
Smooks configuration	- Filter Settings
Smooks Platform Version : 1.2 🗘	Stream Filter Type: SAX ᅌ
Process Options Source	

#### Figure 3.9. Options tab of the Smooks Configuration File editor

## 3.2.1. Smooks Configuration section

In the Smooks Configuration section of Options Tab only one element is available: Smooks Platform Version

Smooks configuration	
Smooks Platform Version :	1.2 0

## Figure 3.10. Smooks Configuration section of Options tab

This parameter is not rechangable, and is set according to the vesion of the Smooks libraries that are added to the project.

## 3.2.2. Filter Settings Filter section

In Filter Settings section you can set the following global options responsible for Smooks filtering configuring:

- Filter Settings		
Stream Filter Type:	SAX	0
Default Serialization is On:		

Figure 3.11. Filter Settings section of Options tab

This behavior can be turned off using this global configuration parameter and can be overriden on a per fragment basis by targetting a Visitor implementation at that fragment that takes ownership of the Result writer (in the case of SAX filtering), or simply modifies the DOM (in the case of DOM filtering). As an example of this, see the FreeMarkerTemplateProcessor.

Option	Description	Default
Stream Filter	Determines the type of processing model that will	SAX
Гуре	be used. Please refer to <u><i>Filtering Process Selection</i></u> section [http://www.smooks.org/mediawiki/index.php?	
	title=V1.2:Smooks_v1.2_User_Guide#Filtering_Process_	Selection28DOM
	of the official Smooks User Guide for more information	
	about these models:	
	• SAX	
	• DOM	
Default	Defines whether default serialization should be	false
Serialization is	switched on. Default serialization being turned on leads	
	objects provided to the Smooks.filterSource method	
	and to serialization all the events to that Result.	

#### Table 3.4. Options Tab. Filter Settings section.

## 3.3. Source Tab

This section provides information about Smooks Source Editor Page.

## 3.3.1. XML Source Editor

You can use this editor to edit the Smooks Configuration file directly.



## Figure 3.12. Graphical Editor

## 3.3.2. Error underlining in Graphical Editor

If the Smooks tools can't understand the configuration file or the configuration file is illegal (XML structure isn't right for Smooks Configuration file, etc.), the error is underlined.



## Figure 3.13. Graphical Editor

## 3.3.3. Smooks Configuration File Validator

Smooks configuration file validator will validate your Smooks configuration file. Just right-click on the file and then click on the Validate button. The validator can be enabled/disabled in *Window* -> *Preferences* -> *Validation*:

9		Preferences				X
typ	pe filter text	Validation			<b>⇔</b> - ⇔~	-
Þ	Help					
0 0 0	HQL editor Install/Update Java Java EE JBoss jBPM	<ul> <li>Allow projects to override these pr</li> <li>Suspend all validators</li> <li>Save all modified resources autom</li> <li>Show a confirmation dialog when provide the second se</li></ul>	eference natically p performin	setting rior to g mani	ys validating ual validations	
Þ	JBoss jBPM	The selected validators will run when v	alidation	is perf	ormed:	
~	JBoss Tools	Validator	Manual	Build	Settings	<u> </u>
	JBoss ESB Runtin	JSF Application Configuration Validate				
	JBoss Portlet	JSF View Validator				
	Project Example:	ISP Syntax Validator				
	▷ Web	ModuleCore Validator				
P	Plug-in Developmer	Seam Ear Project Validator				
	Project Archives	Seam Project Property Validator	N			
P	Run/Debug	Seam Validator				
Ľ	Server	Smooks File Validator		<b>V</b>		
Ь	Team	Tag Library Descriptor Validator			-	
ľ	TestNG	War Validator				
	Validation	WSDL Validator				Ц.
Þ	Web	WS-I Message Validator				
Þ	Web Services	XHTML Syntax Validator				~
Þ	XDoclet	Enable All Disable All				
Þ	XML		-			_
<			F	lestore	Defaults Apply	
(	?			Car	ocel OK	

## Figure 3.14. Validation: Smooks Configuration File Validator

You can set up your Smooks validator to include, exclude groups to validate and specify rules for validation. Just click on the Settings button and use the options provided:

Validation Filters for Smooks	File Validator 🛛 🗙
Filter rules are grouped into groups. Inside of any one g using the logical OR operator. There are two types of gr the rules in the Exclude group match, the resource is n groups, at least one rule from each group must match	group the rules are combined by roups Include and Exclude. If any of not validated. If there are Include before the resource is validated.
▼ Include Group	Add Include Group
Content Type: org.jboss.tools.smooks.ui.smooks	Add <u>E</u> xclude Group
	Add Rule
	Bemove
	Restore <u>D</u> efaults
	>
	Cancel OK

#### Figure 3.15. Smooks Configuration File Validator Settings

For more details about Smooks editor, also see the movie, "Overview of the Smooks Editor" [].

## **3.4. Properties View**

*Properties View* is available for some elements on the canvas of Java Mapping and Apply Template Tasks,like:

- Java mapping: java class members, its fields, links between input values and the class members;
- Apply Template:output template.

To add *Properties View* to the opened perspective the user can either open Window->Show View->Preferences in the toolbar or right click the element which properties he wants to inspect and select Properties in the popup menu.On the picture below you can see how this view looks like when some csv template is selected.

ers 🗖 Properties 🛛		2		- 8
CSV Properties				
Separator Character :	,			
Quote Character :	n			
	ers ☐ Properties ⊠ CSV Properties Separator Character : Quote Character :	ers Properties 23 CSV Properties Separator Character : , Quote Character : "	ers Properties 22	ers Properties 22 CSV Properties Separator Character : , Quote Character : "

#### Figure 3.16. Properties View

This view is fully syncronized with the canvas of *Smooks Configuration Editor*. This means that when you change selected element by click, the properties of a new element are immediatly displayed in it. Using *Properties View* you can edit all the properties of the selected item.

Selected Ta	isk Details		
₹ X ▼ © ▼ Process Optic	ML Input Model order e header e date e customer ⊚ number e order-items e order-item	order © header © orderitems ↓ ⊆ Collection Entry ↓ @ header	
Ju II Init 🔲 Co			_
	onsole 🕫 Servers 🛄 I		
Properties	Properties	Properties &	
Properties	Properties Bean Id* :	orderitems	
Properties	Properties Bean Id* : <u>Class* :</u>	orderitems java.util.ArrayList Browse	
Properties	Properties Bean Id* : <u>Class* :</u> Target Profile :	orderitems java.util.ArrayList Browse	
Properties	Properties Bean Id* : Class* : Target Profile : Extend Lifecycle :	orderitems java.util.ArrayList Browse	

#### Figure 3.17. Synchronization between Properties View and the canvas

## 3.4.1. Decode Configuration

Smooks tools support decode parameter configuration through the Decode tab in *Properties View* activated by clicking the connection between input model and bean items.

On the picture below you can see an example of decode configurations for mapping some Input Model Item to Date format:

Ju JUnit 📮 Console	🗑 🖗 Servers 🗖 Propertie	es 🛛 📑	~ - 8
Decode	Decode Configuratio	n	
Mapping Path	Decoder : Date		~
	Decoder Parameters :		
	Name	Value	
	format	yyyy-MM-dd'T'HH:mm:ss	
	locale-country	IE	
	locale-language	en	

## Figure 3.18. Decode Configuration tab in Properties View

The descriptions of the Decode Configuration tab options are listed in the following table:

Table 3.5. Decode	Configuration tab i	n Properties View
-------------------	---------------------	-------------------

Option	Description	
Decoder	Select from the dropdown list the type of decoder you need.	
Decoder Parameters	For most of decoders <i>Decoder Parametres table</i> is empty. But some of the decoders require additional configuration (like Date decoder on the picture above),so you should configure them by editing the corresponding line in the Value row. For example for Date Decoder:	
	<ul> <li>format - Date format string.</li> <li>locale country - ISO Country Code. Upper case two-letter code defined by ISO-3166.</li> <li>locale language - ISO Language Code. Lower case two-letter code defined by ISO-639.</li> </ul>	

The Decoder Parameters section for EnumDecoder quite differs from other types of decoders. See the picture below:

Ju JUnit 🖳 Console	🖓 Servers 🔲 Properti	es X	2 - 0
Decode	Decode Configuratio	n	
Mapping Path	Decoder : Enum		~
	Decoder Parameters : From Data Value	To Enum Value	
	low	LOW	
	medium	MEDIUM	
	big	BIG	

#### Figure 3.19. EnumDecoder in Properties View

The Decoder Parameters section for EnumDecoder in Properties View consists of 2 rows:

- From Data Value The lines in this row are editable. You can change them according to the names of enum types you used in input file.
- To Enum Value The lines in this row are not editable. Here a set of all constants declared in mapped Enum type is listed. The user is responsible for correspondence between the values in these two rows.

For more information about different decoder parametres read <u>Smooks Technology</u> <u>Documentation</u> [http://www.smooks.org/mediawiki/index.php?title=Main\_Page].

## 3.4.2. Apply Template Wizard

The Apply Template Wizard helps you to add a new Apply Template Task to Smooks configuration file. You can call it from the popup menu when Java Mapping item in Processing Task section is selected by following *Add Task > Apply Template* (see <u>Apply Template configuration</u> picture).

The wizard consists of several pages:

1. The first one includes only one option to adjust. The user should select in which of the two formats(XML or CSV) he prefers to create an output message:

	×
Message Type Selection	
Choose "Message Type" .	
XML	
CSV	
< Back	jinish

#### Figure 3.20. The first page of Apply Template Wizard

- 2. The second page is specific for each of the output message formats:
  - If the CSV output message type was selected at the previous step the second wizard page will be the following :

	×
CSV Template Configuration	
Configure CSV Template: Use ',' to separate fields in the CSV field string.	
Fields	
Separator Character : 🕠	
Quote Character :	
Output Field Names :	
(?) < Back Next > Cancel Enish	

#### Figure 3.21. CSV: The second page of Apply Template Wizard

The wizard page includes the next options to adjust:

Table 3.6. Apply Template Wizard. Second Page Options if CSV output format is selected.

Option	Description	Default
Fields	Comma separated list of CSV record field names.	Empty
Separator Character	Field separator character in the output message.	,
Quote Character	Quote character in the output message.	H
Output Field name	Click the checkbox if you want the output csv message also include field names.	

• The following second wizard page will appear if XML output message type was selected at the previous step:

5	×
Apply XML Template	
Select template generation source	
Create Templete Fremi	
Sample XML	
File :	Load
	Browse File System Browse WorkSpace
Colort root element	
Select root element	
< <u>B</u> ack Ne	xt > Cancel Einish

Figure 3.22. Apply Template Wizard. Second Page Options if XML output format is selected.

Here you should firstly select XSD or Sample XML format of output template and then click Browse File System or Browse Workspace button depending on what browse type you want to use.For example, if you click Browse Workspace the following view will appear:

<b>E</b>		×
Apply XML Te	emplate	
Select templa	te generation source	
Create Templa	Select Files	×
File :	<ul> <li>✓ ⊯ xml-to-java</li> <li>▷ ⇔ .settings</li> <li>▷ ⇔ lib</li> <li>▷ ⇔ src</li> </ul>	
Select root e	Earget schema.xsd	
?		
	(?) Cancel	ок

## Figure 3.23. Browse Workspace

In the workspace you should select the template you want to use and click Ok.

If you selected XSD format after adjusting the template path you should click Load button:

Apply XML T	emplate	
🙆 Press 'Load	' button to load XSD root element names	
Create Templa	ate From:	
XSD		•
File : Works	pace://xml-to-java/schema.xsd Browse File :	Load System Browse WorkSpace
Select root e	lement	
?	< Back	Cancel

## Figure 3.24. Load Button

After that it is necessary to select in the Select Root Element the root node for the template and click Finish.

		×
Apply XML Template		
Select template generation source		
Create Template From:		
XSD		0
File : Workspace://xml-to-java/schema.>	ksd	Load
	Browse File System	Browse WorkSpace
Select root element		
✓ order		
< Back Nex	t > Cancel	Einish

## Figure 3.25. Load Button

If you have chosen Sample XML option after selecting the template xml file destination you should only click Finish:

	×
Apply XML Template	
Select template generation source	
Create Template From:	_
Sample XML	0
File : Workspace://xml-to-java/simple.xml	
Browso File System	3
Blowse rile System	1
(?) < Back Next > Cancel Finish	$\mathcal{D}$

Figure 3.26. Load Button

# Summary

In conclusion, with this document you know all the capabilities of Smooks Tools and could easily start with them. The chapters above walked you through the steps on how to create and configure some XML to JAVA mapping project. If you have questions or suggestions concerned both the documentation and tools behavior, you are welcome to JBoss Tools Users forum. Please, use Jira to report bugs and requests on documentation.

## 4.1. Other relevant resources on the topic

All JBoss Developer Studio/JBoss Tools release documentation you can find at <u>http://</u> <u>docs.jboss.org/tools</u> in the corresponding release directory.

The latest documentation builds are available at <u>http://download.jboss.org/jbosstools/nightly-docs</u>.

For more information about Smooks technology please visit <u>Smooks Technology Home Page</u> [http://www.smooks.org/mediawiki/index.php?title=Main\_Page]

You can find a set of screencasts on Smooks tools technology <u>here</u> [http://community.jboss.org/ wiki/JBossTools-SmooksEditor].