Smooks Developer Tools Reference Guide

Version: 3.2.0.Beta

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.	*Smooks 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.

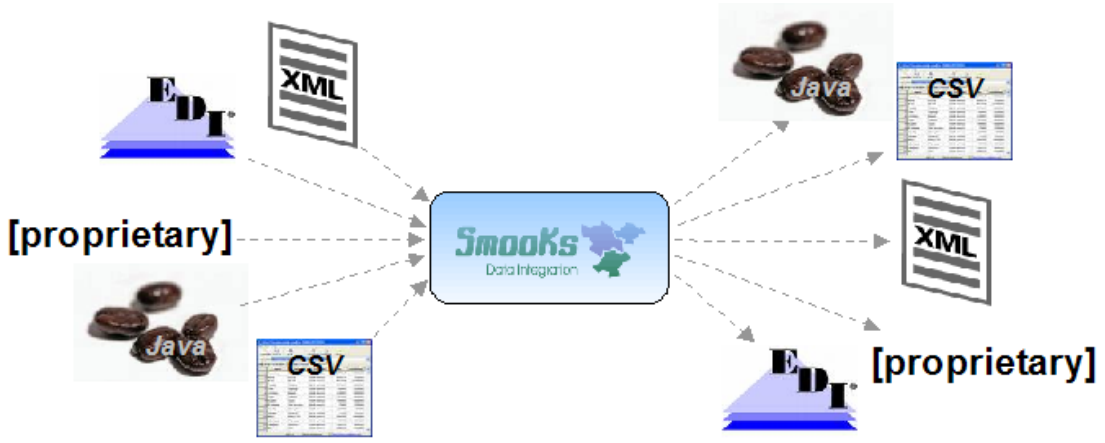

Figure 1.1. Smooks

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.

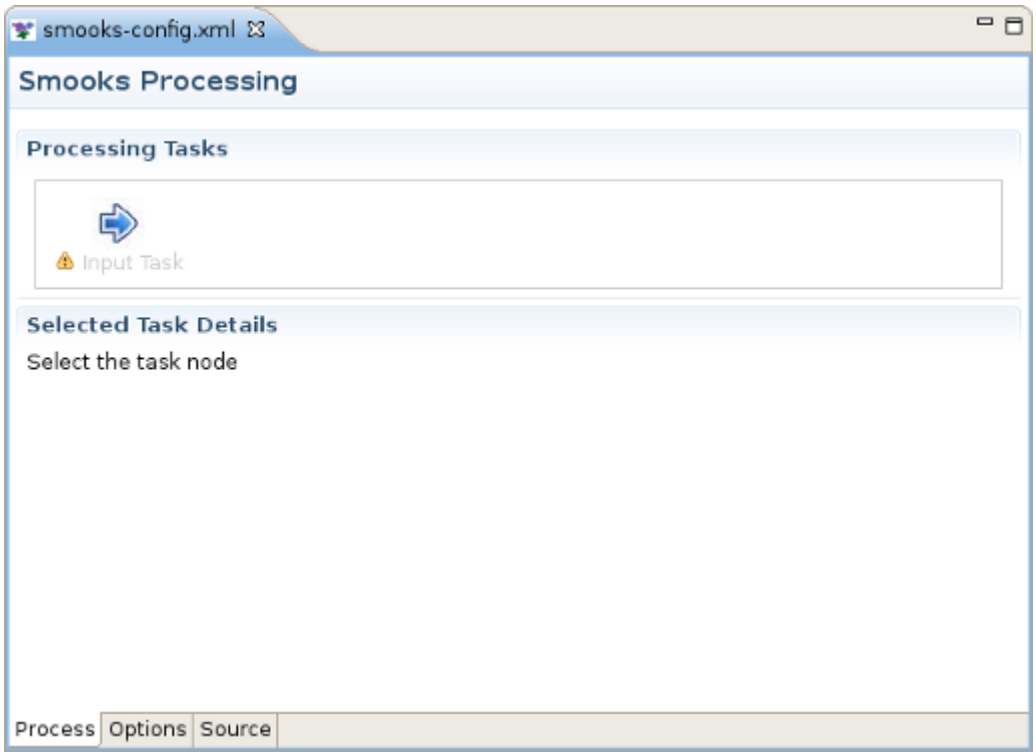

Figure 1.2. Smooks Form editor

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.

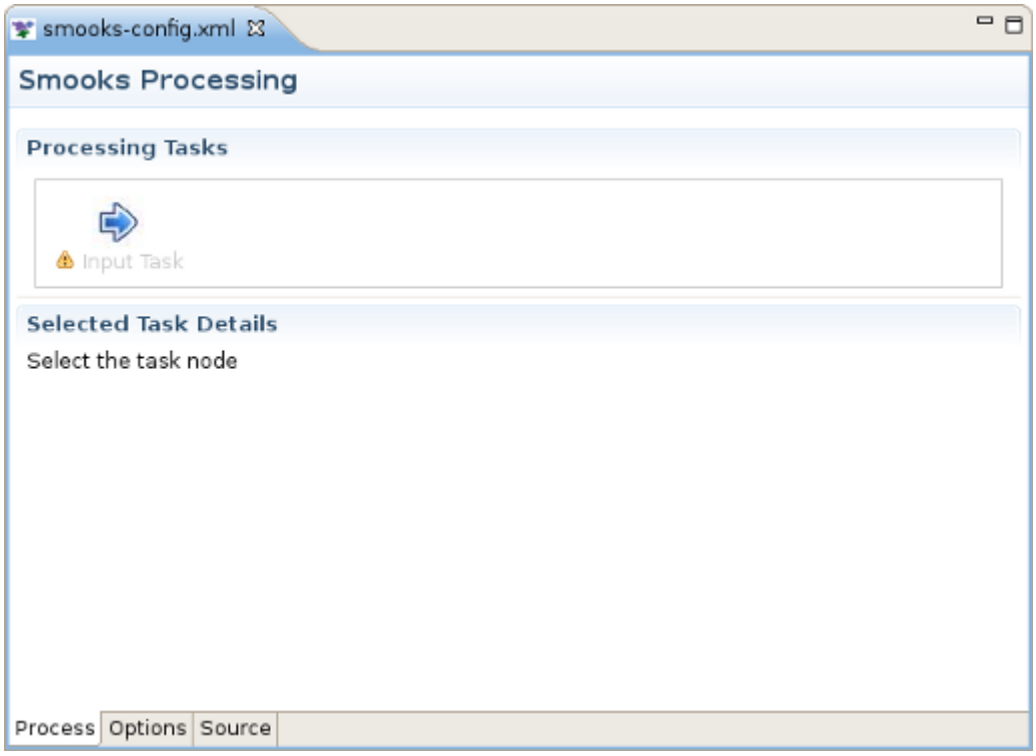

Figure 1.3. Smooks Form editor

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 *here* [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.

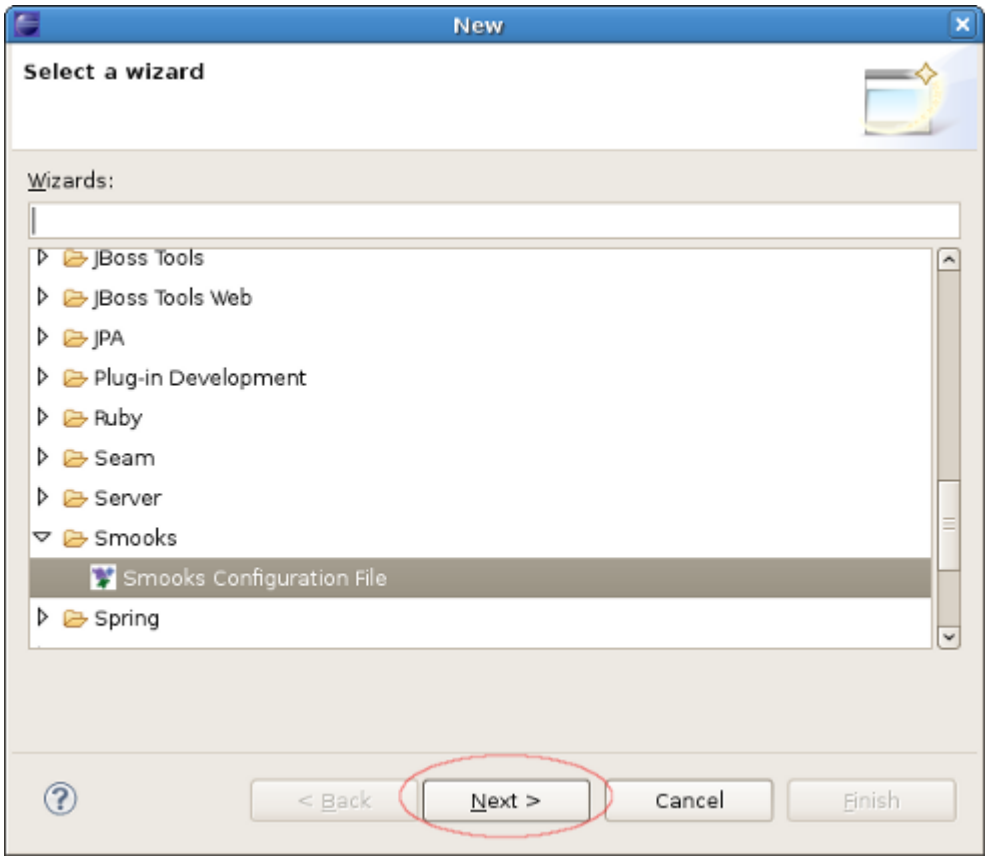

Figure 2.1. Selecting Smooks Configuration File Wizard

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*.

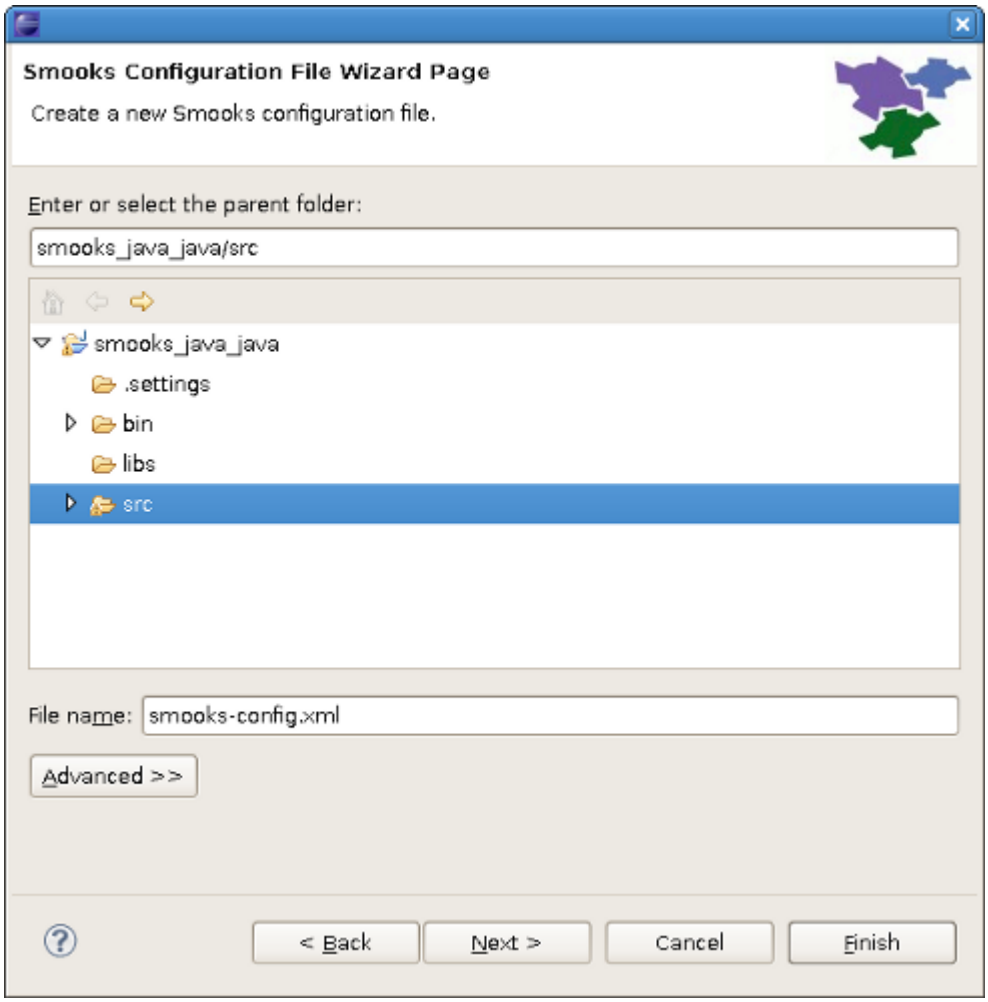

Figure 2.2. Choosing the configuration file container and the file name

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.

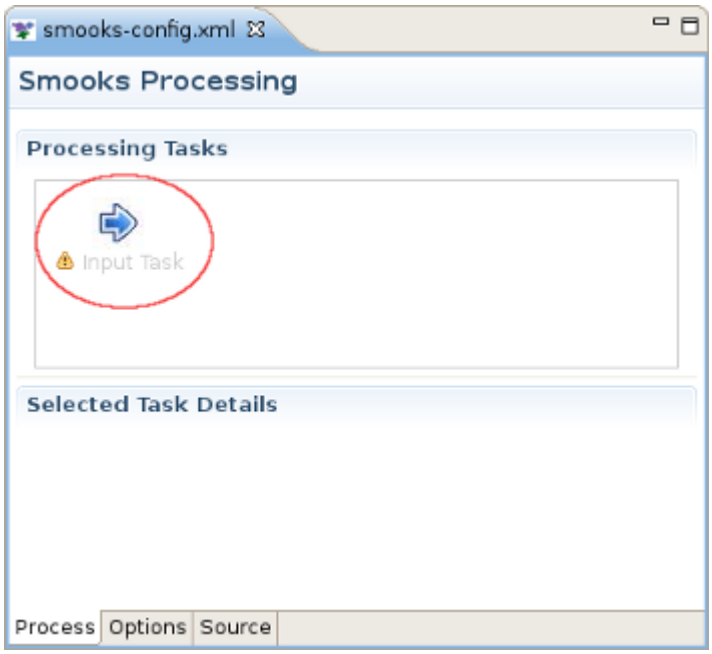

Figure 2.3. Input Task Configuring

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.

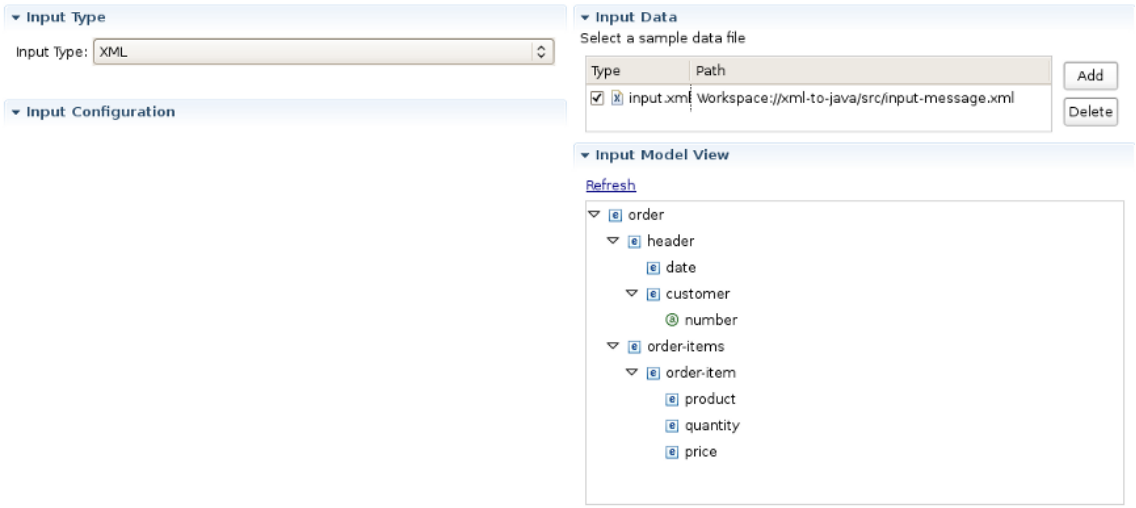

Figure 2.4. Input Task Configuring

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.

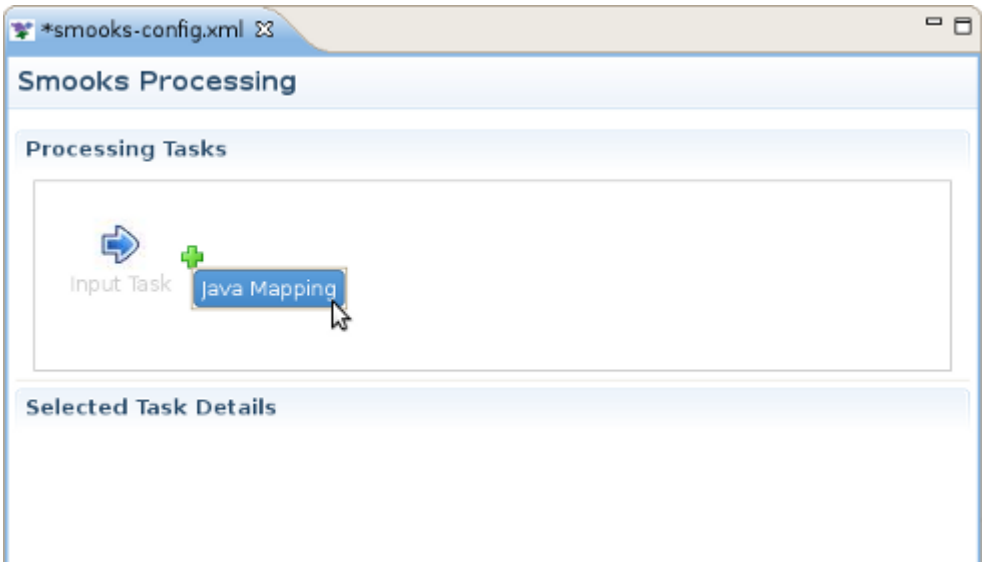

Figure 2.5. Java Mapping configuration

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.

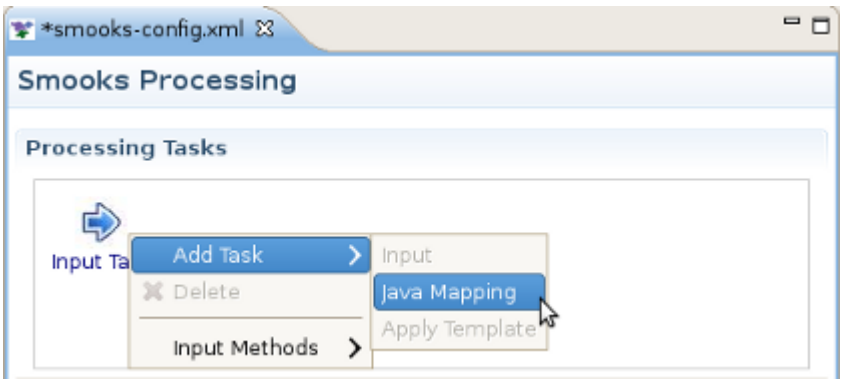

Figure 2.6. Java Mapping configuration

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

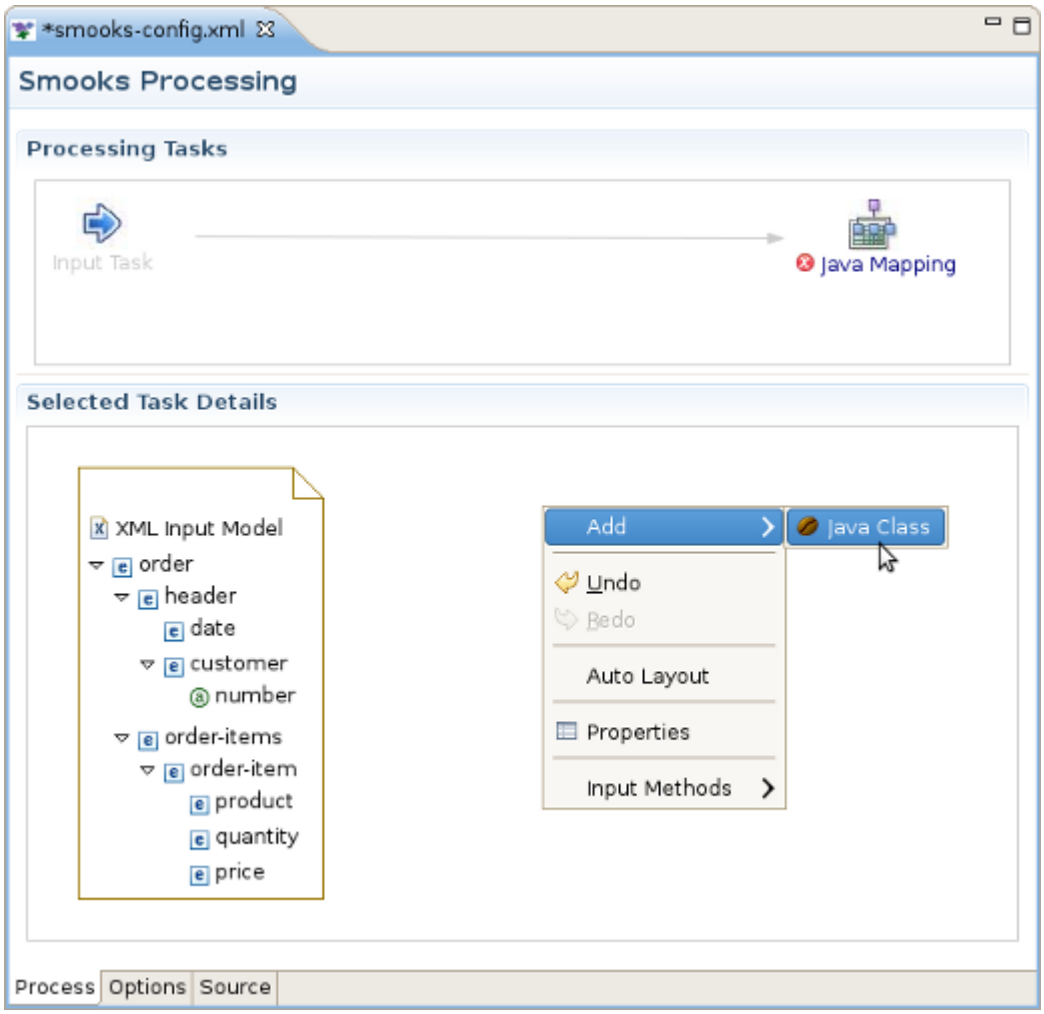

Figure 2.7. Java Mapping configuration

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.

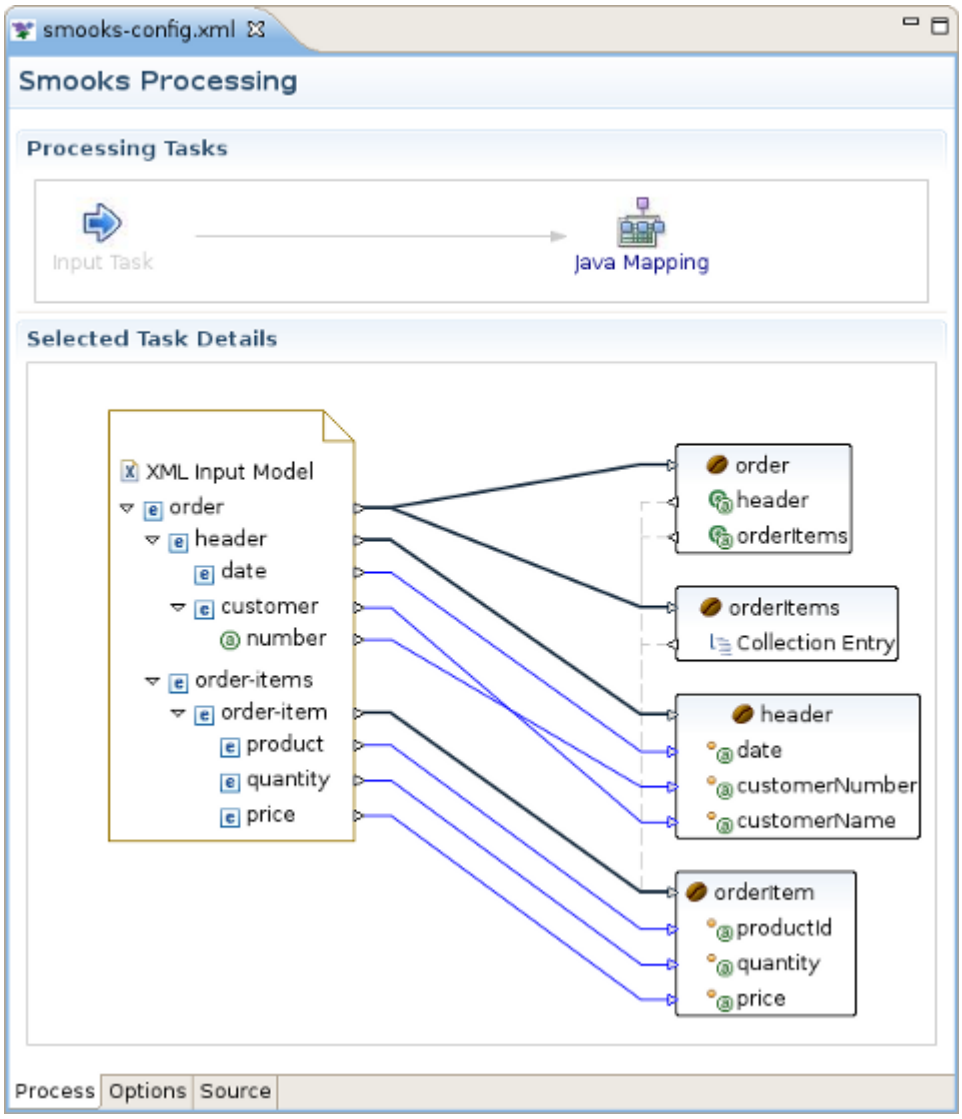

Figure 2.8. Final Mapping schema

For details, also see the movie, *"XML to Java"* [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 *link* [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 *"Java Mapping" task*, 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.

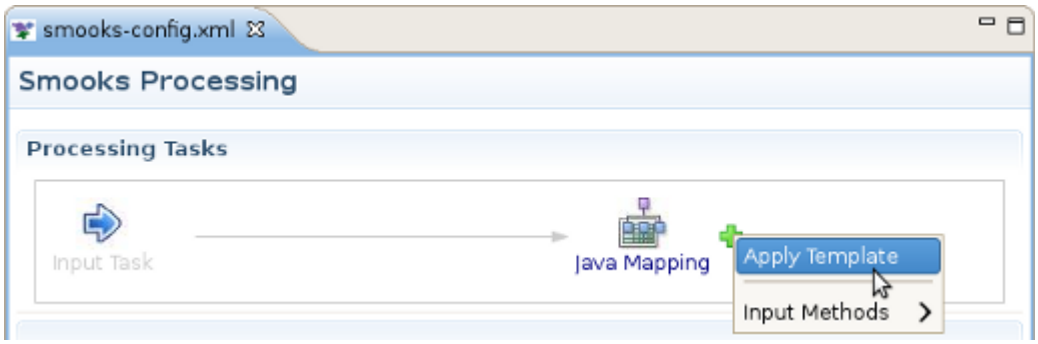

Figure 2.9. Apply Template configuration

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.

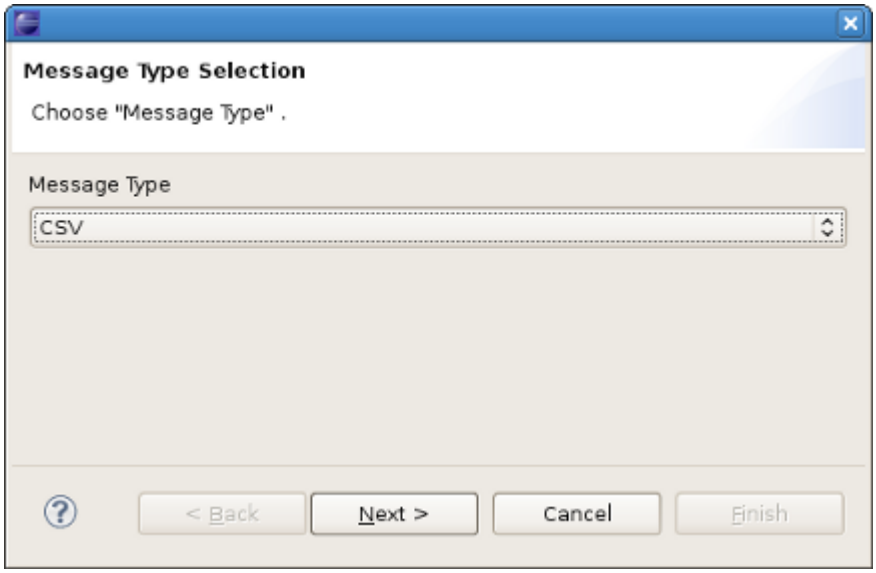

Figure 2.10. Message Type Selection

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

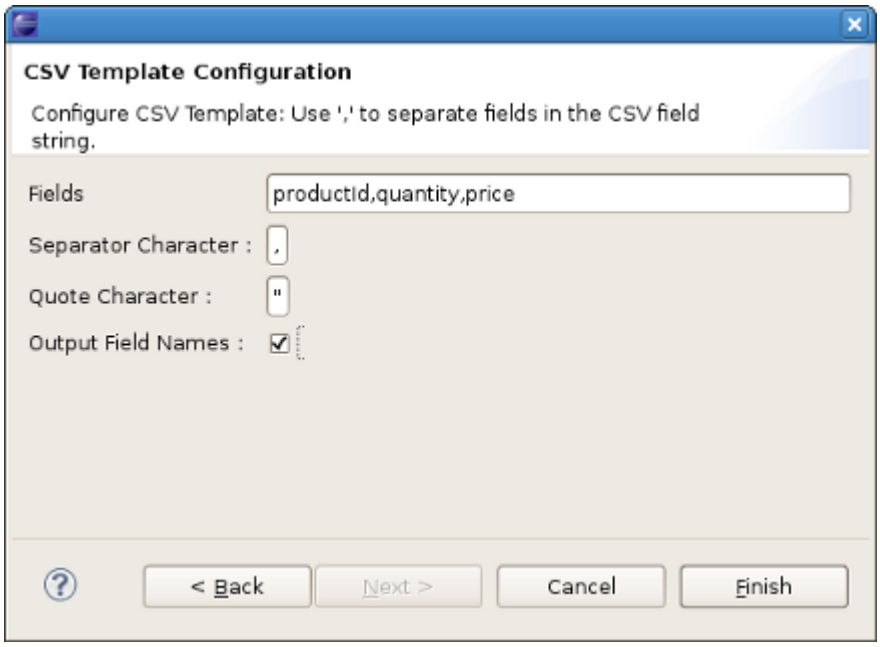

Figure 2.11. CSV output message configuration

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):

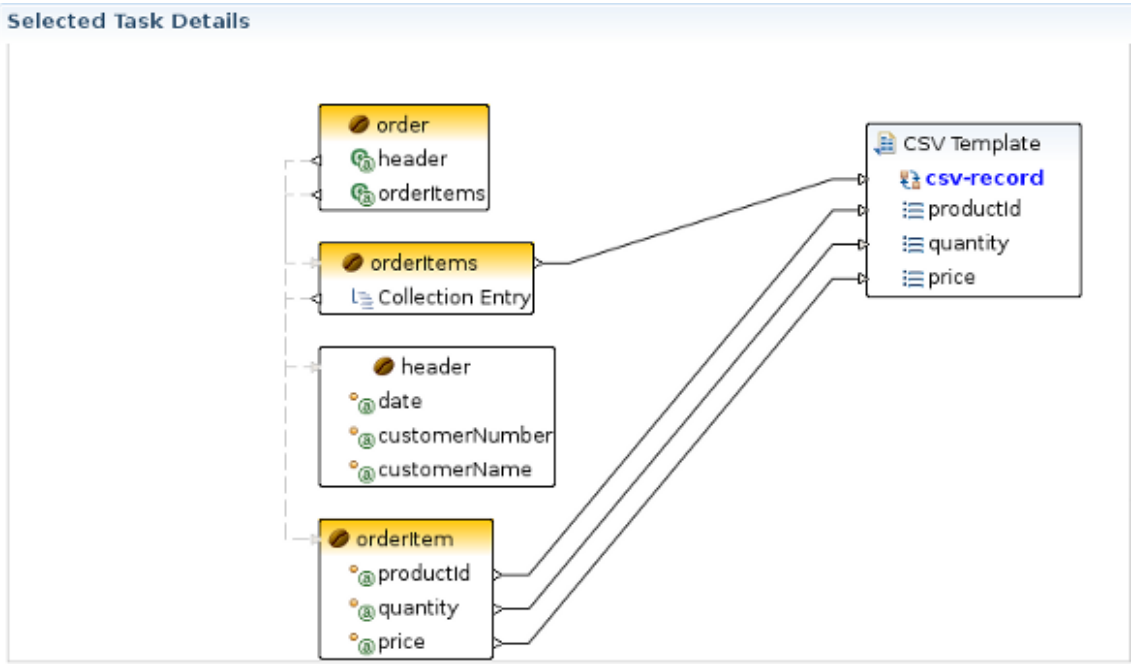

Figure 2.12. Relations between input and output models

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.

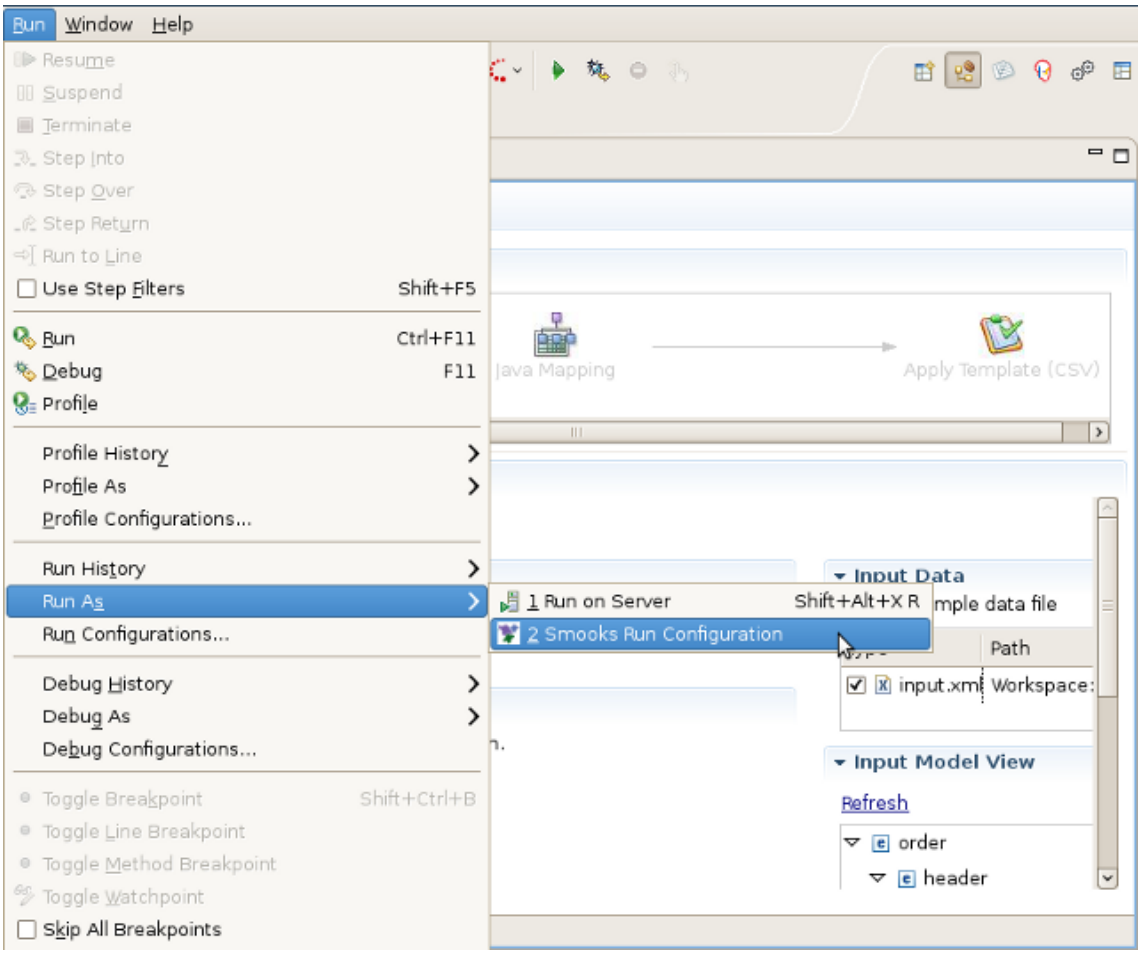

Figure 2.13. Smooks Configuration testing

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")
   |            > productId = 222L
   |            > quantity = 7I
   |            > price = 5.2D
   |--
```

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.

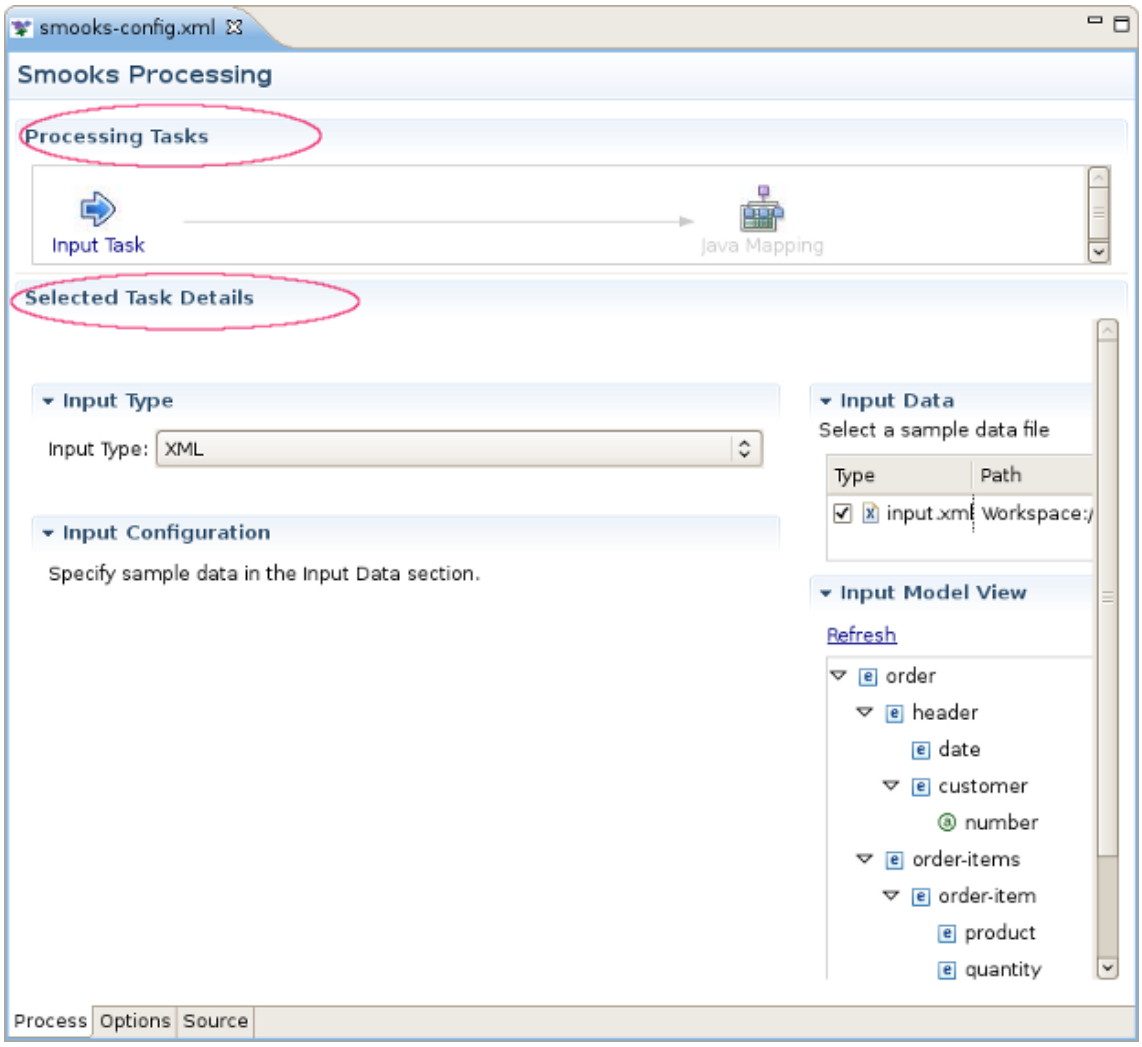

Figure 3.1. Two Sections of the Process tab.

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.

Table 3.1. Process Tab. Processing Task section.

Option	Description	Default
Add Task	Select one of the following tasks according to the necessary type of Source and Result types of the files: • *Input* - 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	Choose one of the following methods: • System • Simple • Amharic(EZ+) • Cedilla • Cyrillic • Inuktitut • IPA • Multipress • SCIM Bridge Input Method • SCIM Input Method • Thai-Lio • Tigrigna-Eritrean(EZ+) • Tigrigna-Ethiopian(EZ+) • Vietnamese • X input Method	System

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.

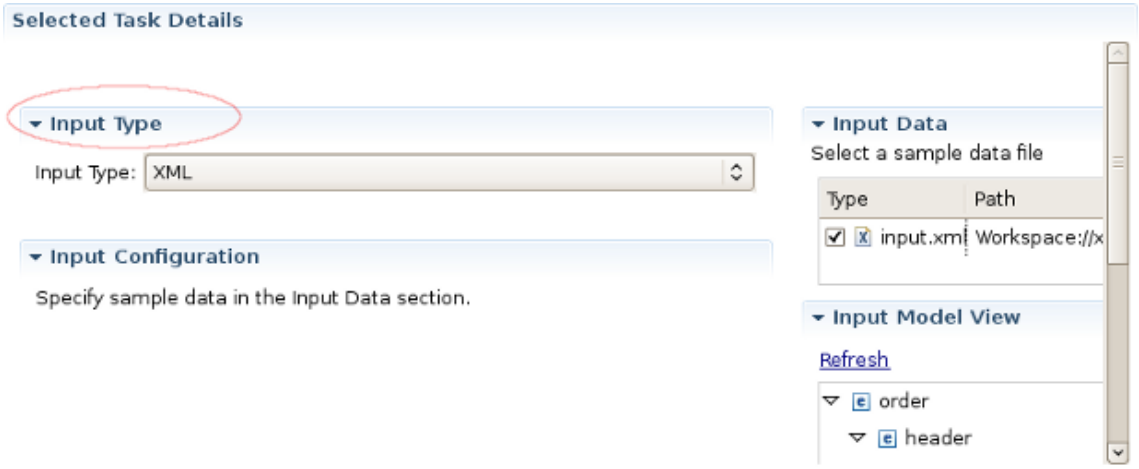

Figure 3.2. Selected Task Details Section for Input XML Task.

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:

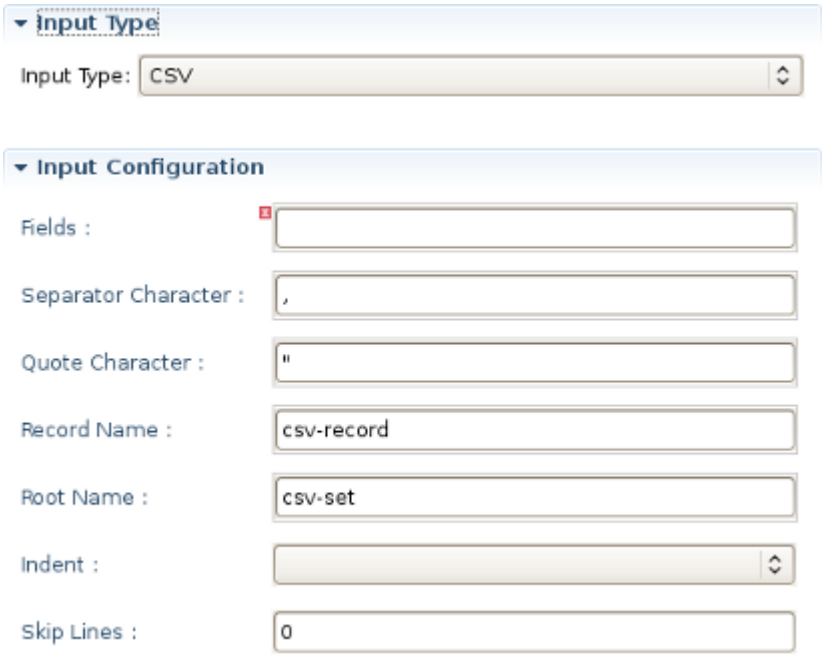

Figure 3.3. Selected Task Details Section for Input CSV Task.

- EDI:

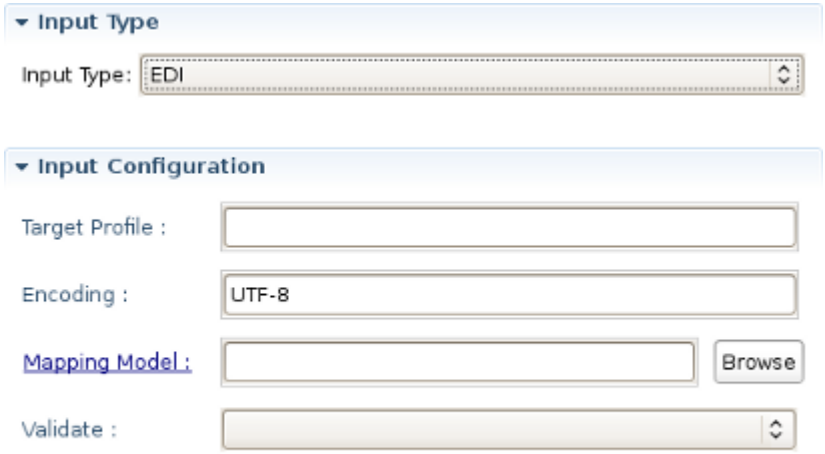

Figure 3.4. Selected Task Details Section for Input EDI Task.

- JSON:

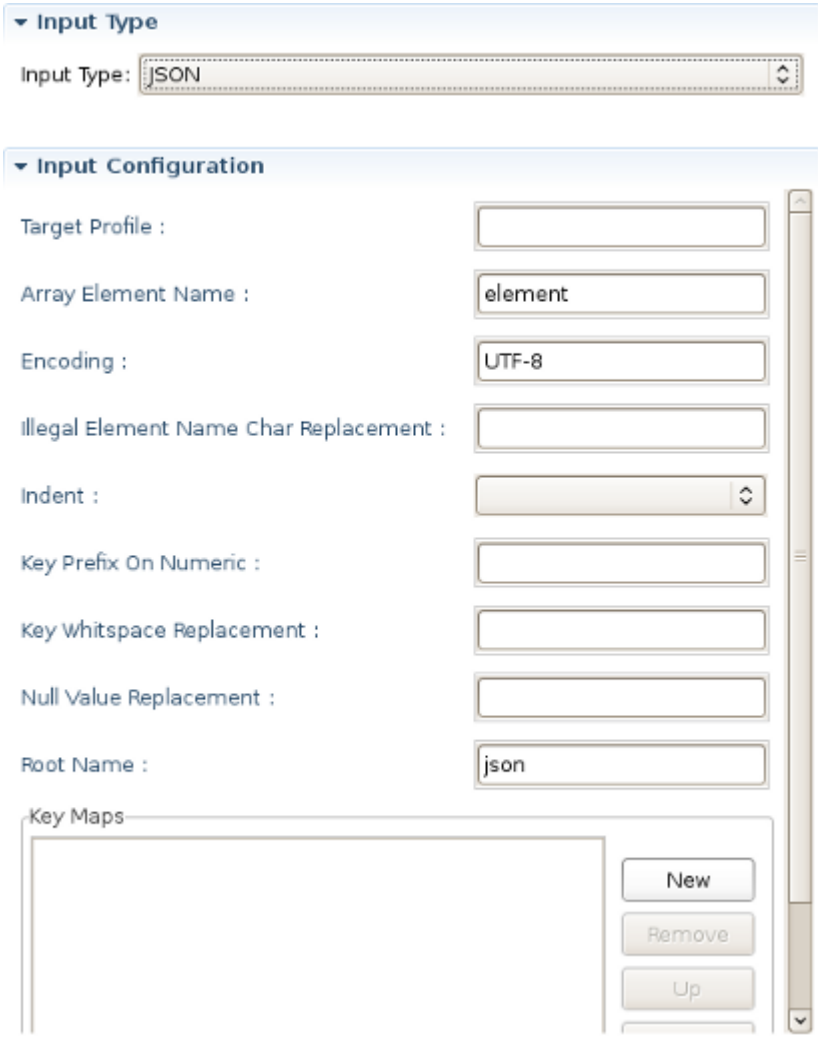

Figure 3.5. Selected Task Details Section for Input JSON Task.

- Custom:

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. Options for Input Task.

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

Option	Description	Default
	• CSV • EDI • JSON • Custom	

Input configuration	• *No Input* - no info required
	• *XML* - no info required
	• *Java* - no info required
	• *XSD/WSDL* - no info required
	• *CSV*
	  • *Fields* - Comma separated list of CSV record field names
	  • *Separator Character* - Field separator character.
	  • *Quote Character* - Quote character.
	  • *Record Name* - Name of csv record element.
	  • *Root Name* - Name of csv root element.
	  • *indent* - 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.
	  • *Skip Lines* - Number of lines to skip before processing starts.
	• *EDI*
	  • *Target Profile* - Defines the output transformation profile
	  • *Encoding* - The character encoding.
	  • *Mapping Model* - Defines the EDI Mapping Model configuration for processing the EDI message stream to a stream of SAX events that can be processed by Smooks.
		• *CSV*
		  • not defined
		  • ','
		  • ""
		  • csv-record
		  • csv-set
		  • true
		  • 0
		• *EDI*
		  • not defined
		  • UTF-8
		  • not defined
		  • true
		• *JSON*
		  • not defined
		  • element
		  • UTF-8
		  • not defined
		  • false
		  • not defined
		  • not defined
		  • ""(an empty string)

Option	Description	Default

	• *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.
	• *JSON*
	  • *Target Profile* - Defines the output transformation profile
	  • *Array Element Name* - The element name of an array element.
	  • *Encoding* - encoding: The default encoding of any JSON message InputStream processed by this Reader.
	  • *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.
	  • *Indent* - 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.
	  • *Key Prefix on Numeric* - 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.
	  • *Key Whitespace Replacement* - The replacement character for whitespaces in a JSON map key. By default this not defined, so that the reader doesn't search for whitespaces.
	  • *Null Value Replacement* - The replacement string for JSON NULL values.
	  • *Root Name* - The element name of the document root.
		• 'json'
		  • not defined
		• *Custom*
		  • no defaults

Option	Description	Default

	• *Key Maps* - Defines a JSON element name mapping The "from" key will be replaced with the "to" key or the contents of the element.
	• *Custom*
	  • *Target Profile* - Defines the output transformation profile
	  • *Class* - Custom reader class.
	  • *Handlers* - Set a handler on the reader instance e.g. an EntityResolver, ErrorHandler etc.
	  • *Features* - Reader Features List
	  • *Parametres* - Resource Parameters	

Input Data	You should select a data file using *Add* and *Delete* 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 *Refresh*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.

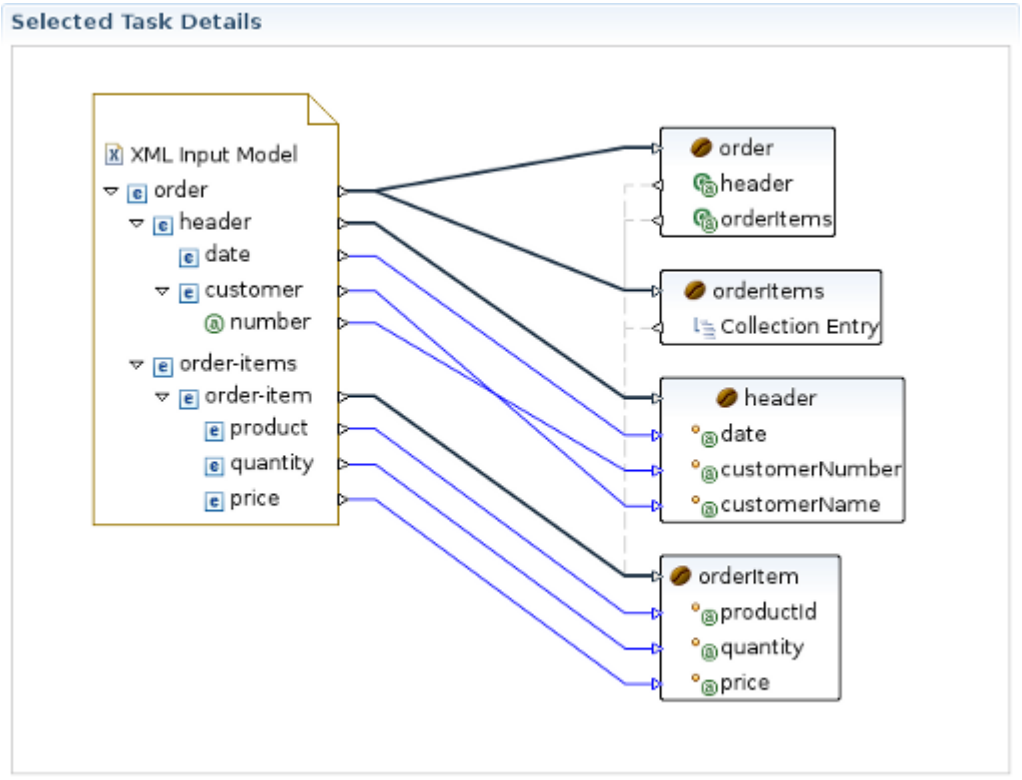

Figure 3.7. Selected Task Details Section for Mapping Task.

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.

Table 3.3. Popup menu. Selected Task Details section.

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. • *Expression Binding* - adds expression based binding to selected java binding element. • *Value Binding* - adds Value binding (<jb:value>) to the selected java binding element.	

Option	Description	Default
	• *Bean Binding* - adds Wiring binding (<jb:wiring>) to the selected java binding element.	
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 *Properties view* to the current perspective. The just opened *Properties view* will automatically reflect the properties of the selected diagram element.	
Input Methods	Choose one of the following methods: • System • Simple • Amharic(EZ+) • Cedilla • Cyrillic • Inuktitut • IPA • Multipress • SCIM Bridge Input Method • SCIM Input Method • Thai-Lio • Tigrigna-Eritrean(EZ+) • Tigrigna-Ethiopian(EZ+) • Vietnamese • X input Method	System

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*.

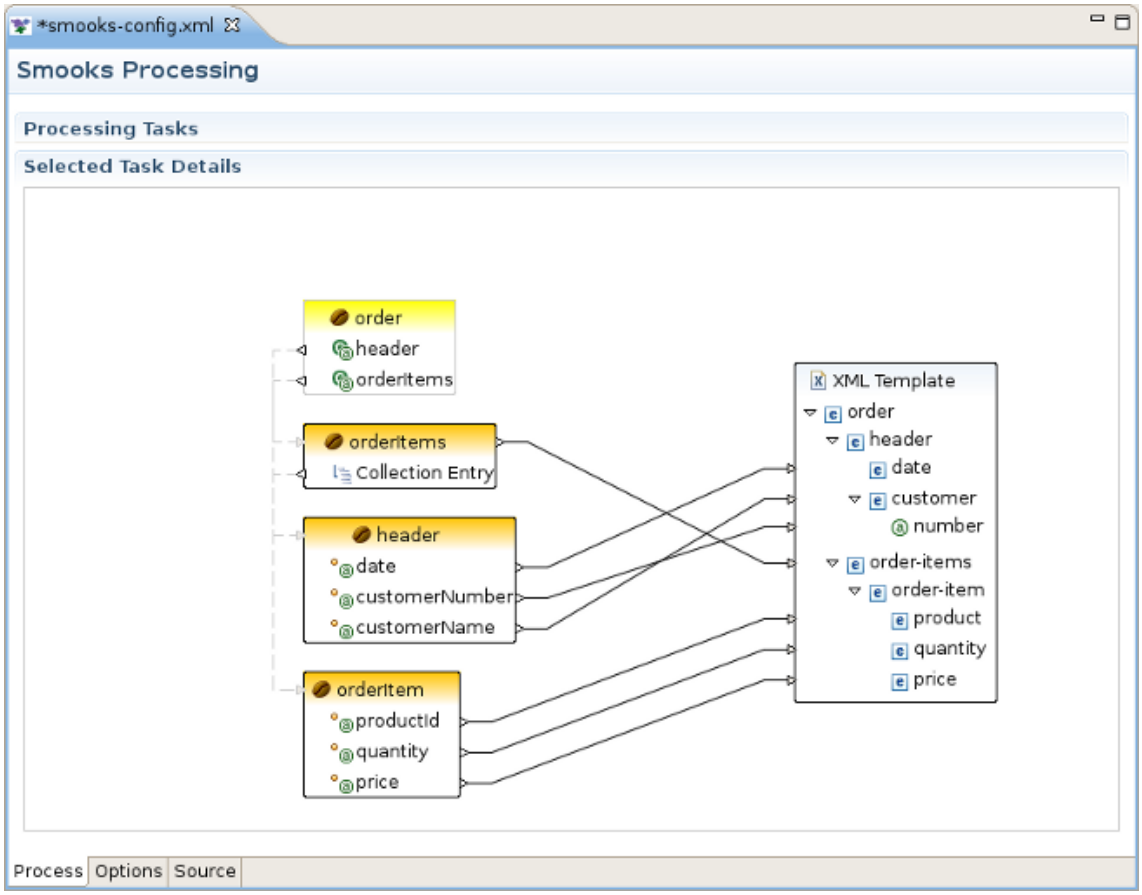

Figure 3.8. Selected Task Details Section for Template Task.

Popup menu similar to the one in *Selected Task Details section for Java Mapping Task* 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.

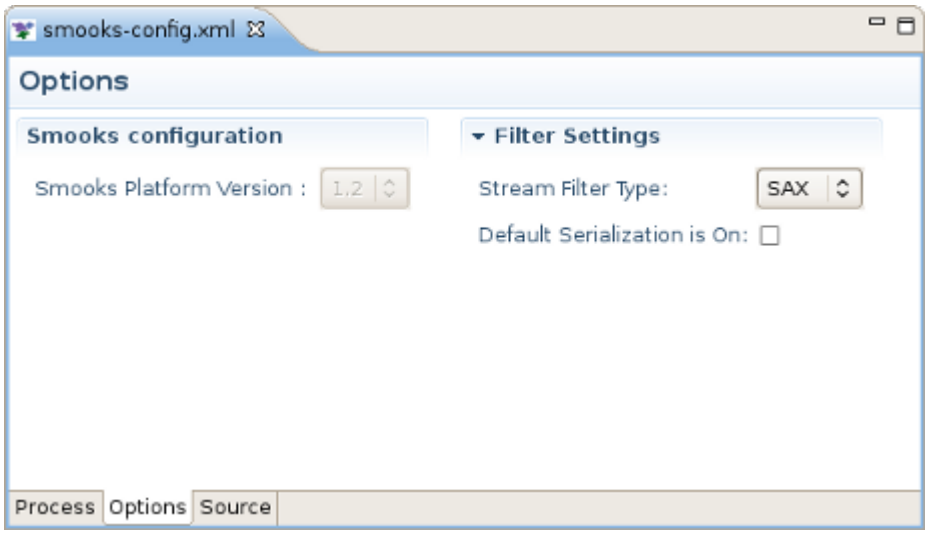

Figure 3.9. Options tab of the Smooks Configuration File editor

3.2.1. Smooks Configuration section

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

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:

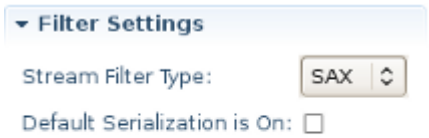

Figure 3.11. Filter Settings section of Options tab

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.

Table 3.4. Options Tab. Filter Settings section.

Option	Description	Default
Stream Filter Type	Determines the type of processing model that will be used. Please refer to *Filtering Process Selection section* [http://www.smooks.org/mediawiki/index.php?title=V1.2:Smooks_v1.2_User_Guide#Filtering_Process_Selection_.28DOM_or_SAX.3F.29] of the official Smooks User Guide for more information about these models: • *SAX* • *DOM*	SAX
Default Serialization is On	Defines whether default serialization should be switched on. Default serialization being turned on leads to locating StreamResult/DOMResult to the Result objects provided to the Smooks.filterSource method and to serialization all the events to that Result.	false

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.

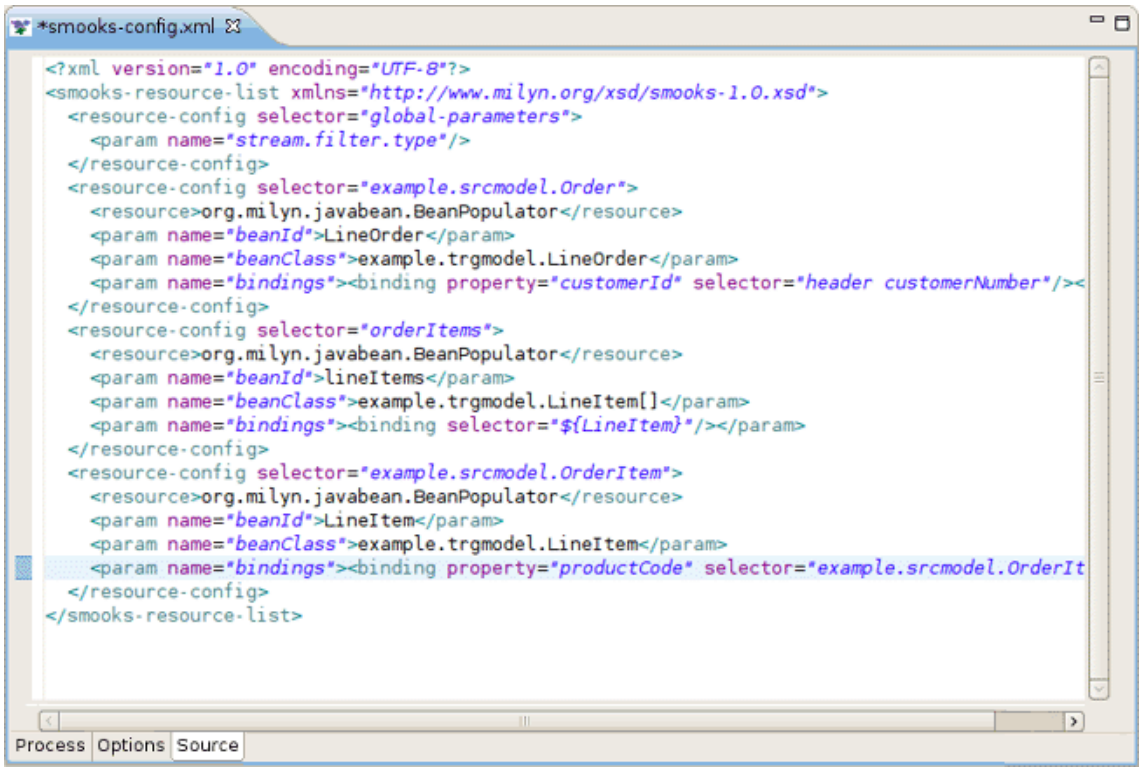

Figure 3.12. Graphical Editor

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.

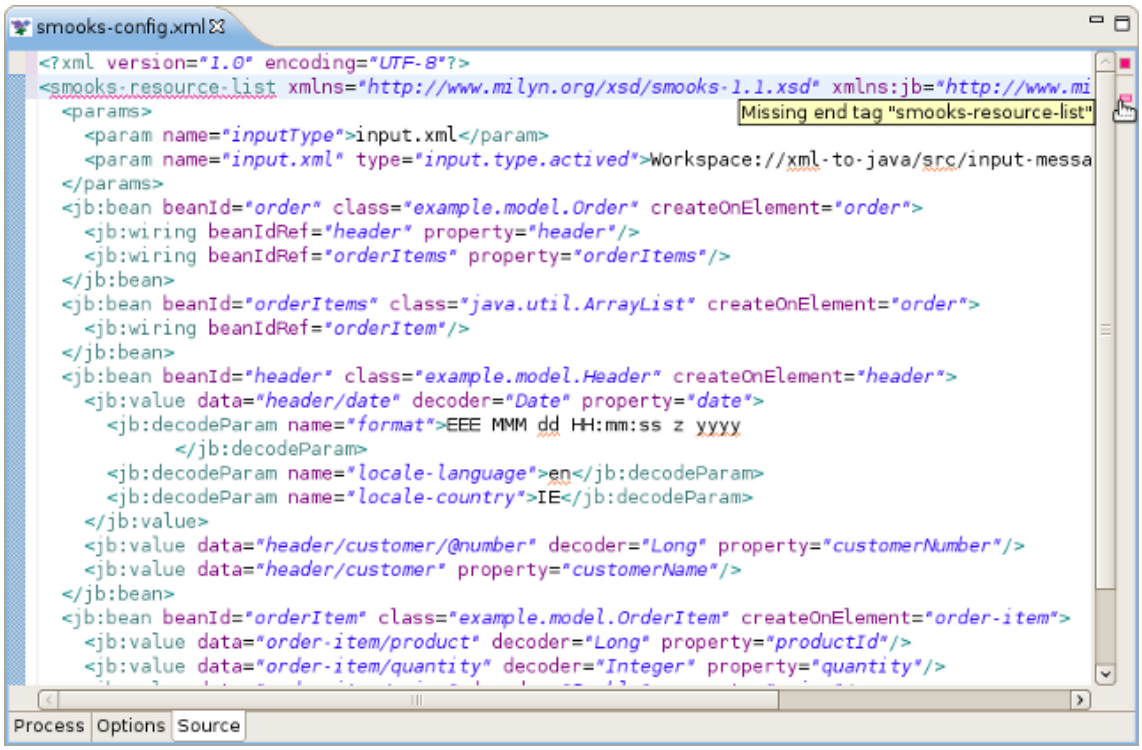

Figure 3.13. Graphical Editor

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*:

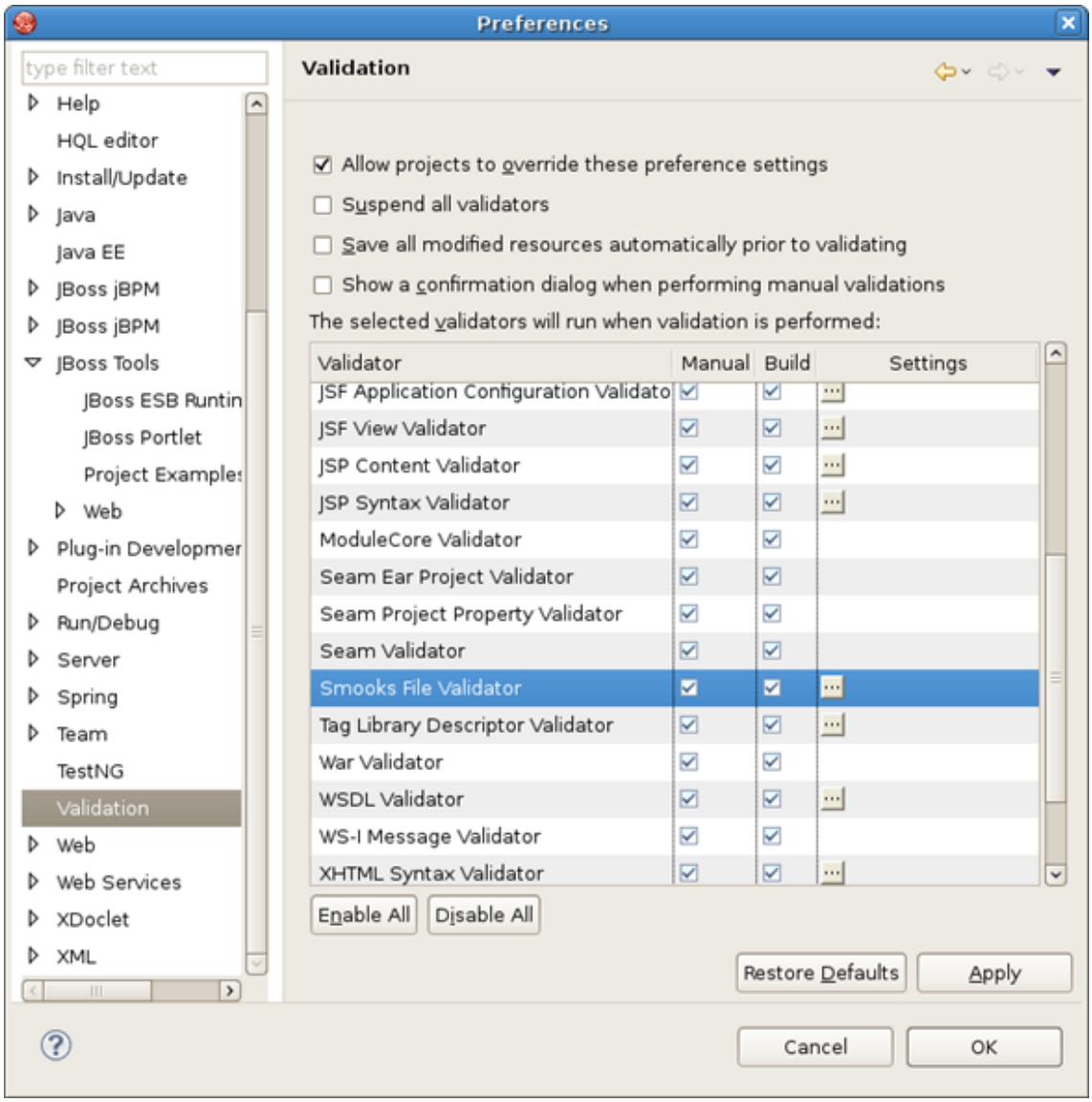

Figure 3.14. Validation: Smooks Configuration File Validator

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:

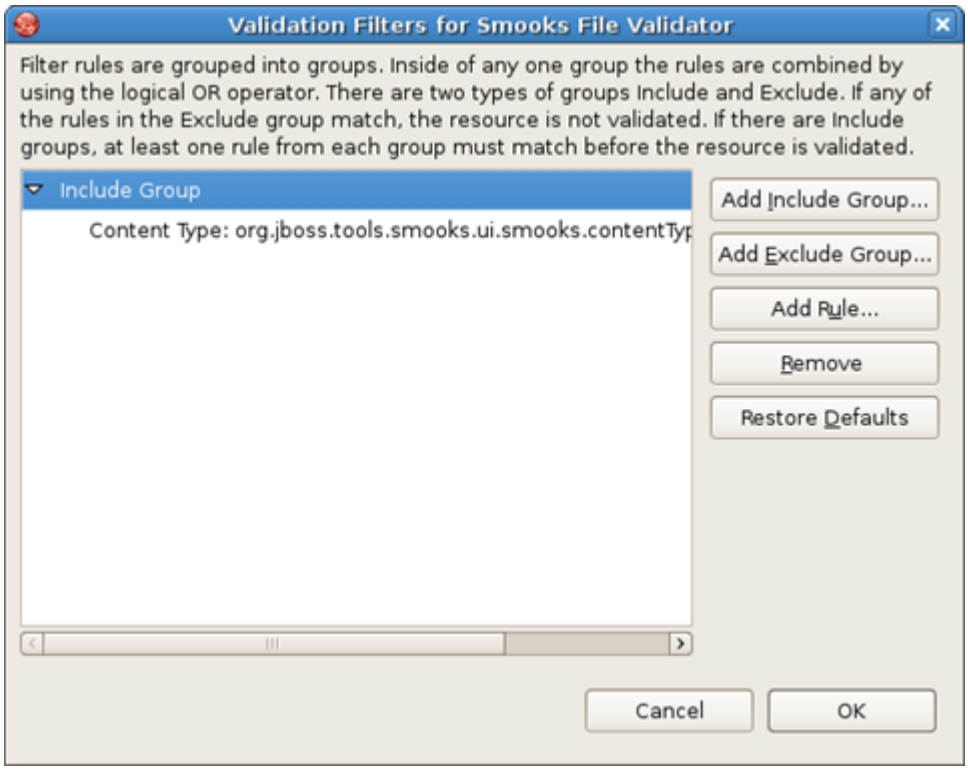

Figure 3.15. Smooks Configuration File Validator Settings

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.

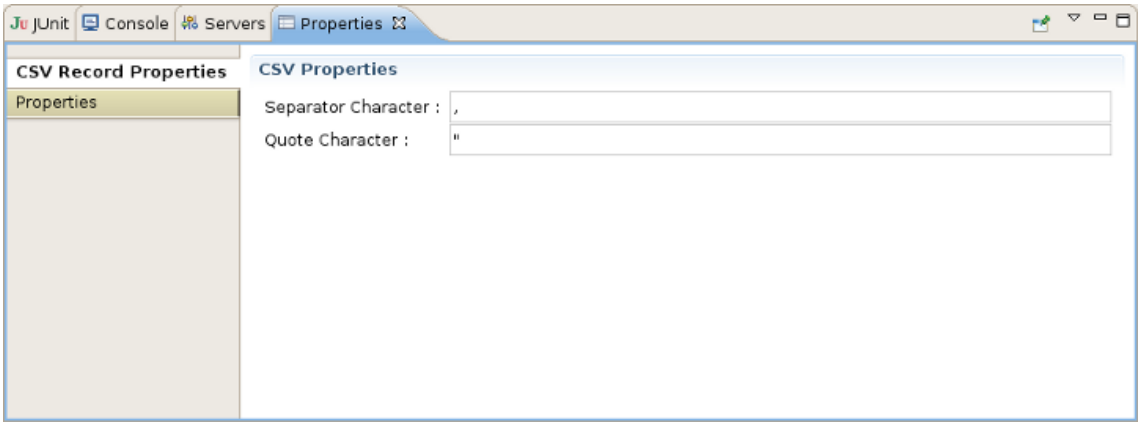

Figure 3.16. Properties View

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.

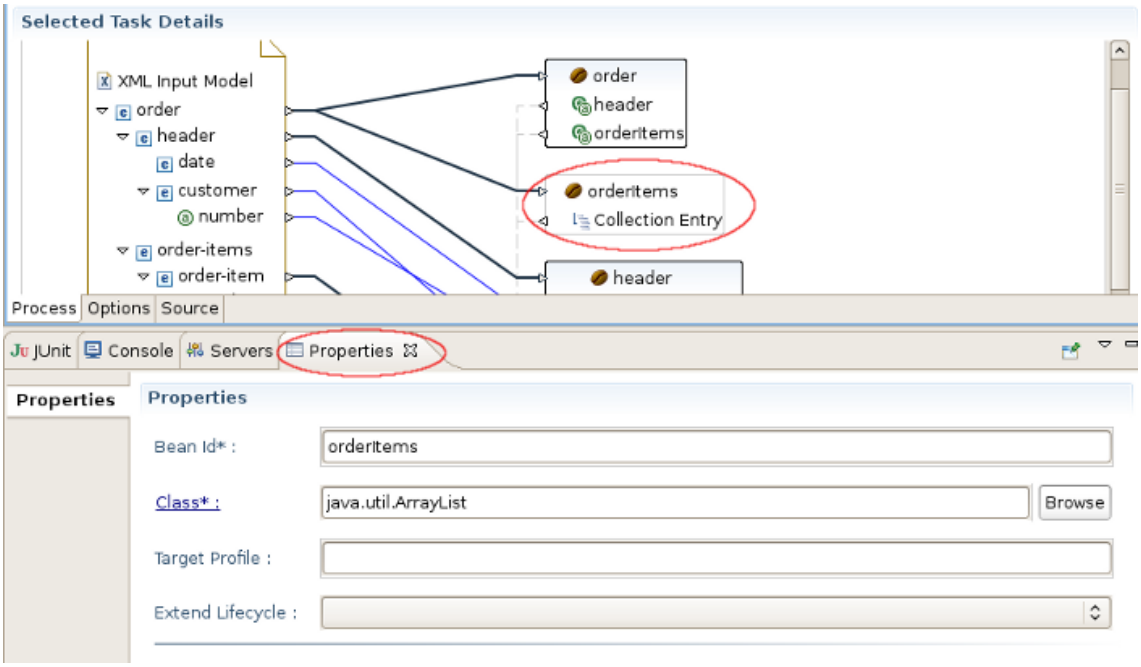

Figure 3.17. Synchronization between Properties View and the canvas

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:

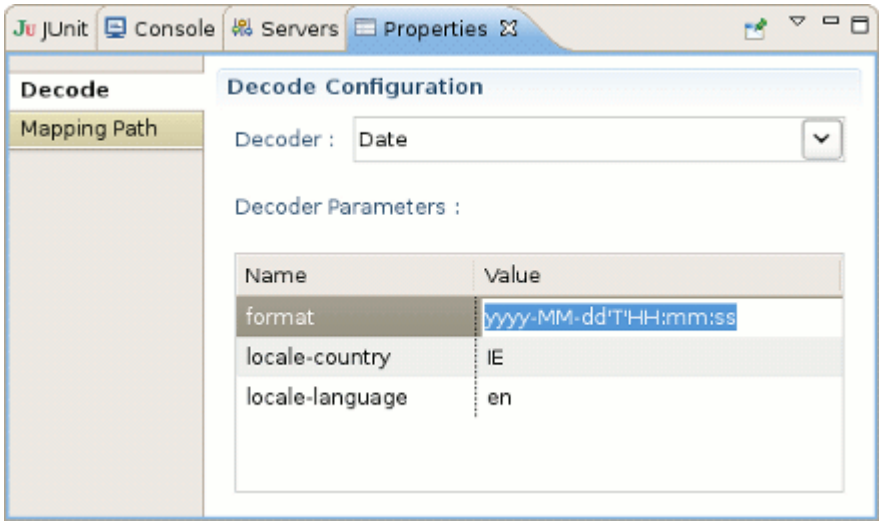

Figure 3.18. Decode Configuration tab in Properties View

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

Table 3.5. Decode Configuration tab in Properties View

Option	Description	
Decoder	Select from the dropdown list the type of decoder you need.	
Decoder Parameters	For most of decoders *Decoder Parametres table* 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: • format - Date format string. • locale country - ISO Country Code. Upper case two-letter code defined by ISO-3166. • locale language - ISO Language Code. Lower case two-letter code defined by ISO-639.	

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

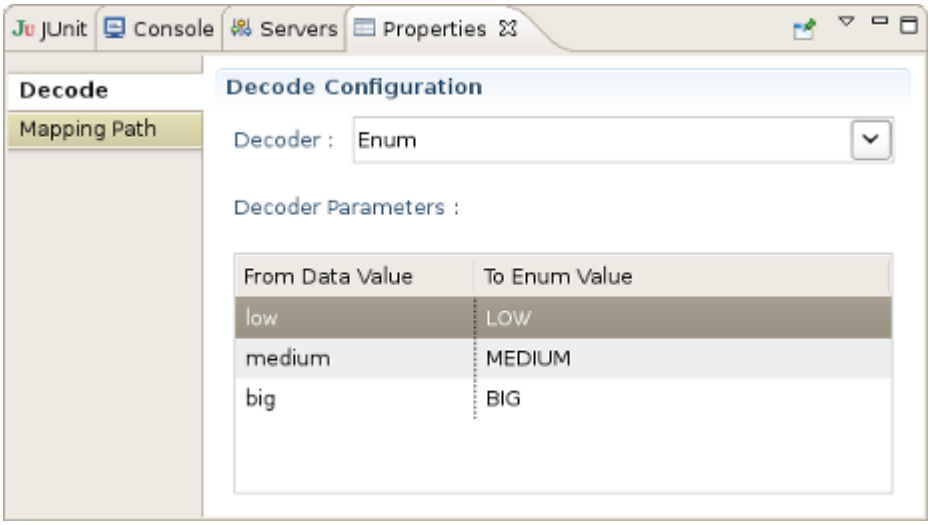

Figure 3.19. EnumDecoder in Properties View

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 *Smooks Technology Documentation* [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 *Apply Template configuration* 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:

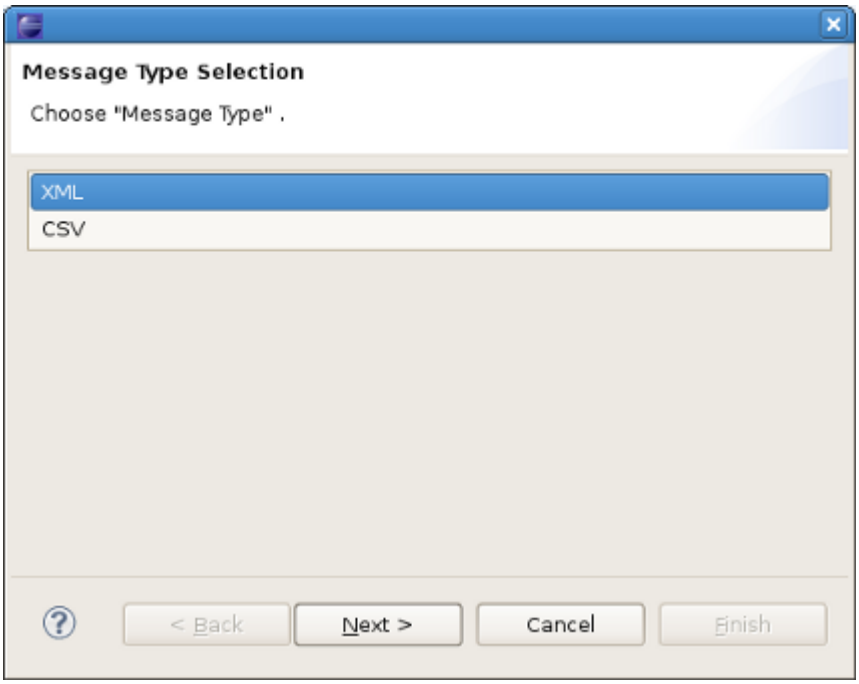

Figure 3.20. The first page of Apply Template Wizard

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 :

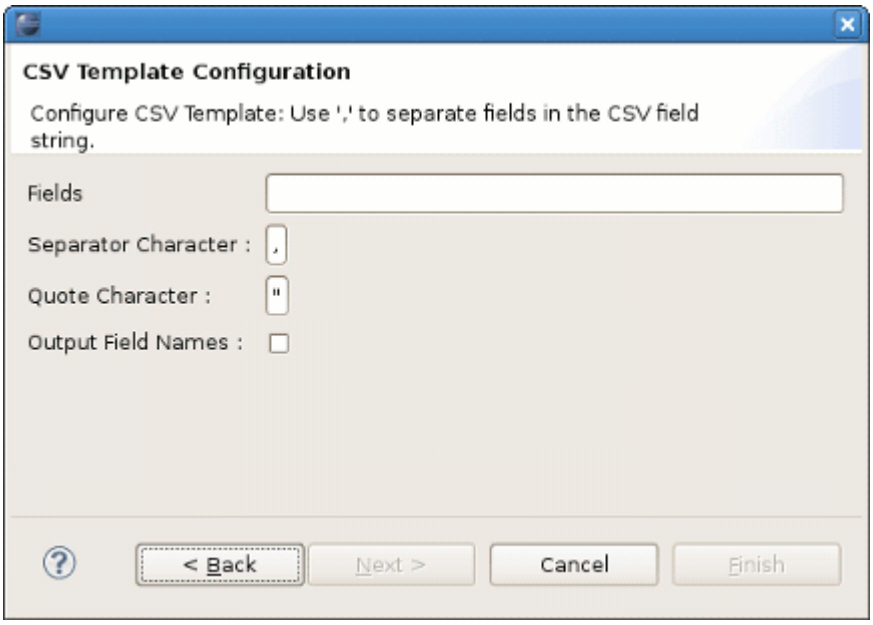

Figure 3.21. CSV:The second page of Apply Template Wizard

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.	"
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:

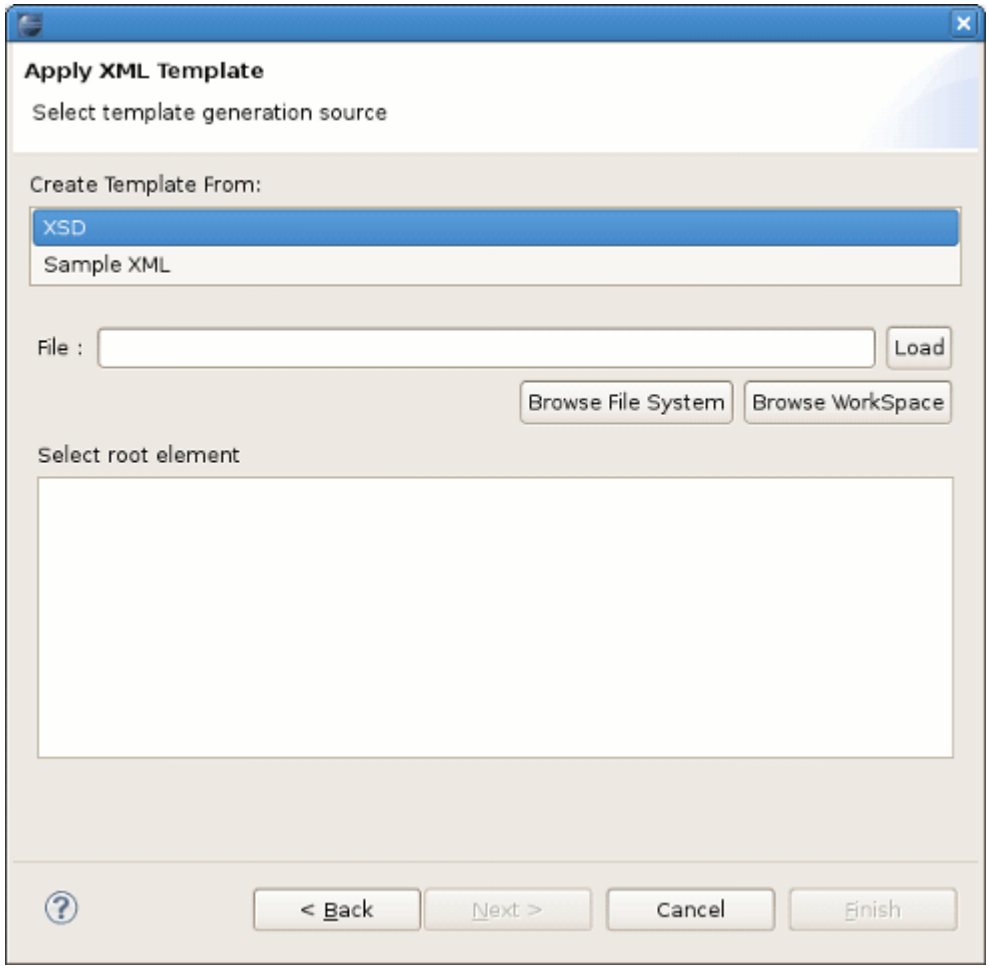

Figure 3.22. Apply Template Wizard. Second Page Options if XML output format is selected.

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:

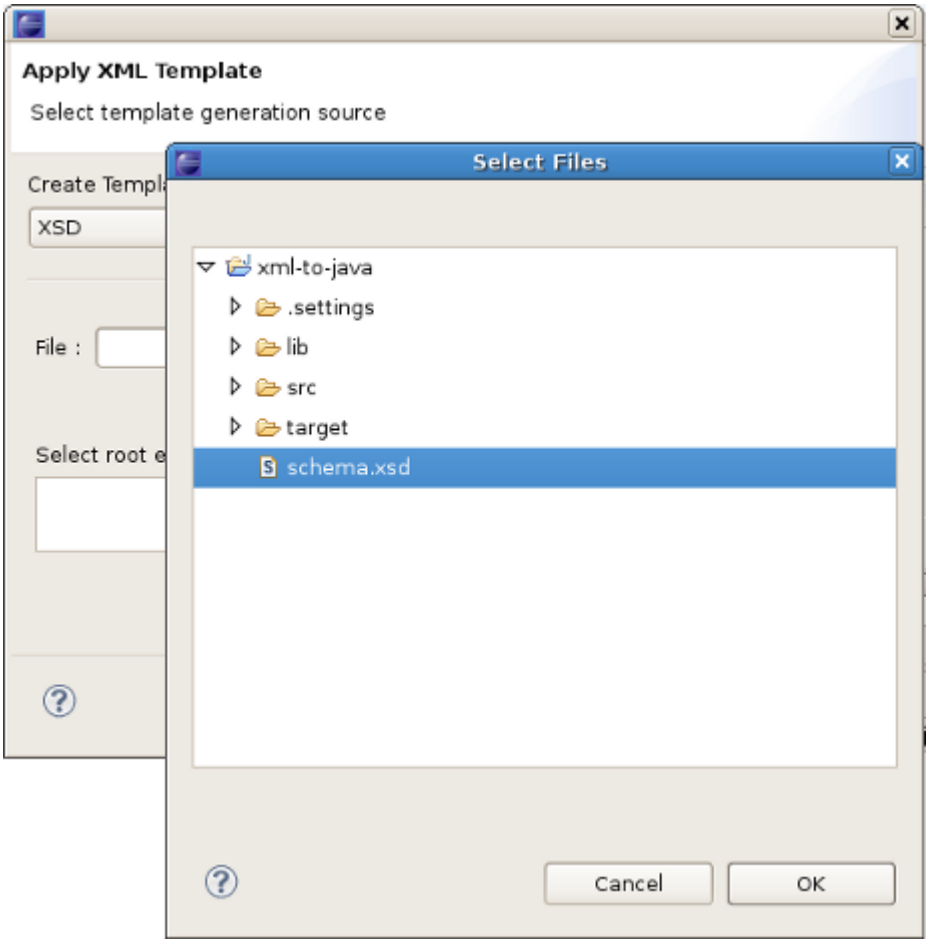

Figure 3.23. Browse Workspace

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:

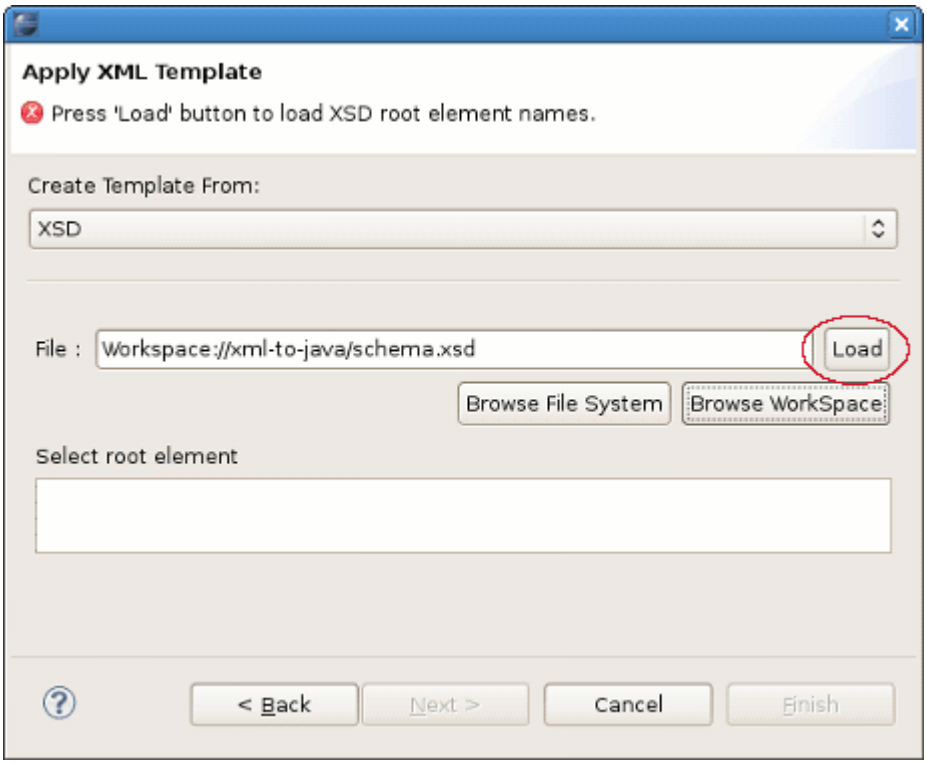

Figure 3.24. Load Button

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

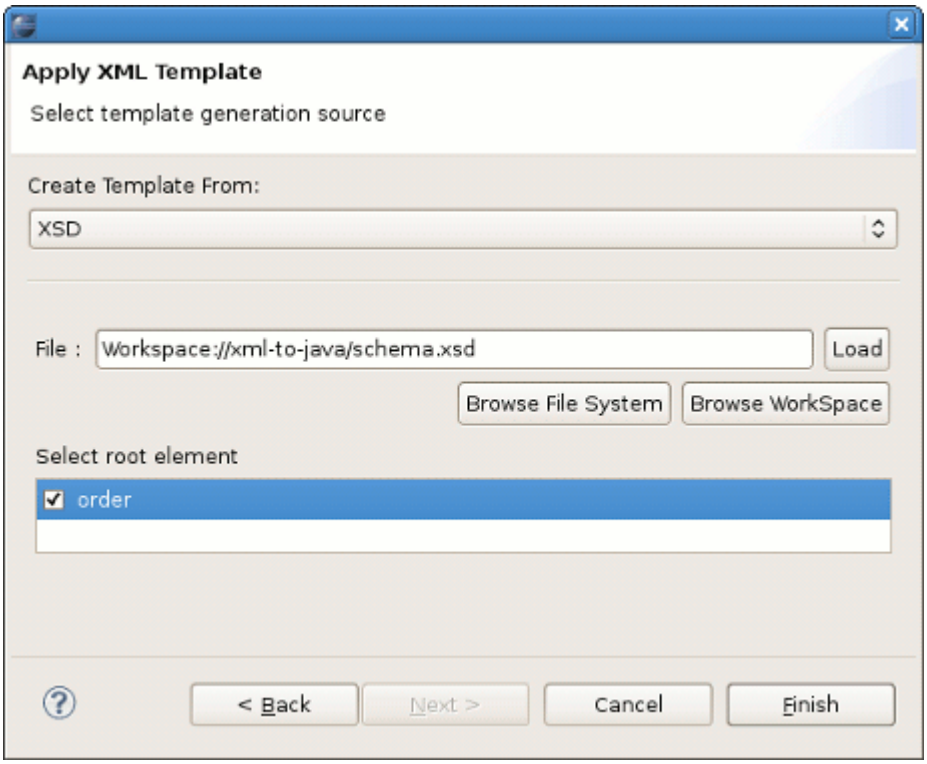

Figure 3.25. Load Button

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

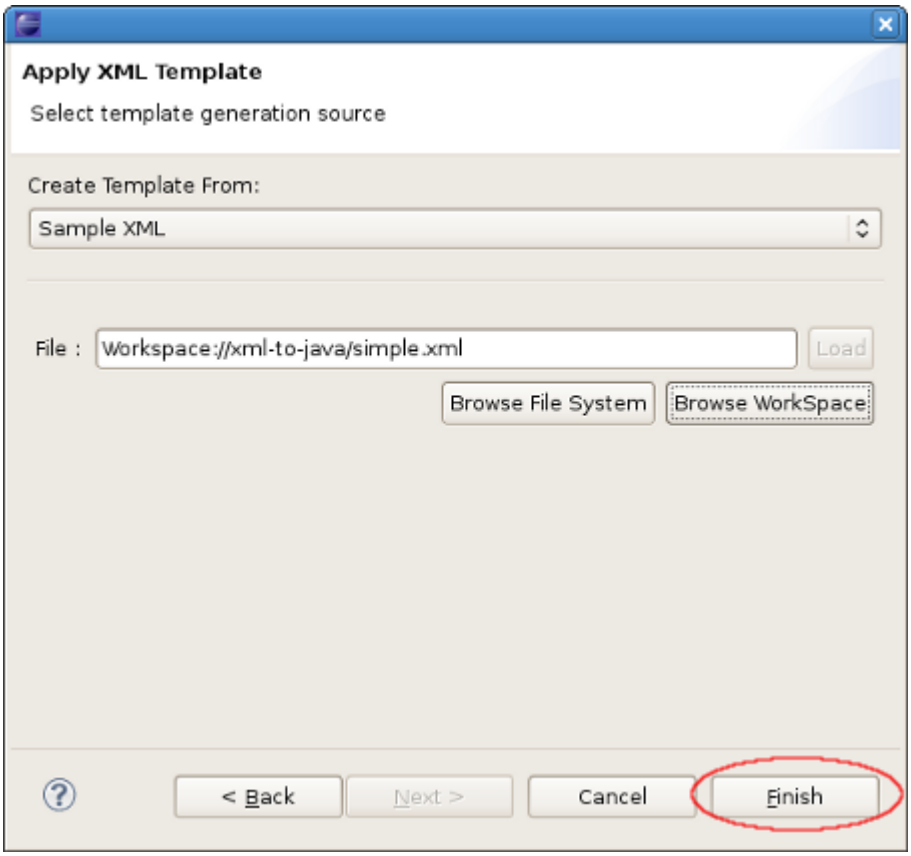

Figure 3.26. Load Button

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 *http://docs.jboss.org/tools* in the corresponding release directory.

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

For more information about Smooks technology please visit *Smooks Technology Home Page* [http://www.smooks.org/mediawiki/index.php?title=Main_Page]

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