Smooks Dev Tools Reference Guide

Version: 1.1.0.CR1
Chapter 1. Introduction

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

<table>
<thead>
<tr>
<th>Feature</th>
<th>Benefit</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smooks Configuration File Wizard</td>
<td>Smooks tools allows to create/edit the Smooks configuration file for Java2Java data transformation.</td>
<td>Smooks Configuration File Wizard</td>
</tr>
<tr>
<td>Smooks Editor</td>
<td>Smooks Editor helps configure the created Smooks configuration file.</td>
<td>Smooks Editor</td>
</tr>
</tbody>
</table>

1.2. What is Smooks?

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

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.
Chapter 1. Introduction

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. How to install Smooks dev tools?

The Smooks tools were included by the JBoss Tools since 3.0.0 Beta1 version. You can download the JBoss Tools from JBoss download site [http://jboss.org/tools/download.html].

Smooks tools (JBoss Tools) run with the latest Eclipse and other required plug-ins (GEF, EMF, etc.).

You should download the latest IDE for Java EE developers from Eclipse site [http://www.eclipse.org/downloads/]. It contains many plug-ins (GEF, EMF, etc.) required by Smooks dev tools.

Now let's progress to more advanced topics.
Quick Start

This "Quick Start" will show how to use the Smooks tools to create/edit the Smooks configuration file for Java2Java data transformation.

This chapter gives you a brief understanding of the Smooks tools.

2.1. Download Smooks Java-to-Java Example

Visit smooks examples page [http://www.blogjava.net/Files/reloadcn/smooks_java_java.zip] to download the "java-to-java" example project.

Tip:

The example project is based on Maven, we suggest you to create a new eclipse Java project, copy the example code and .jar files from the example project and paste them into the eclipse Java project.

2.2. Mapping rules

At the begining of working with Smooks tools, we need to understand the Java-to-Java mapping rules of this example. The transform input Java class is example.sourcemodel.Order, its structure looks like this :

![Figure 2.1. Order class](image)

Let's describe the structure of example.sourcemodel.Order in short:
Chapter 2. Quick Start

- Order class contains a Header class and a List class.
- The Header class has customerName, customerNumber, priority as properties.
- In Order class the objects of the OrderItem class are stored in the List.
- OrderItem has price, productId, quantity as its properties.

Let's look at the transform target Java class: LineOrder.

On the screen below you can see its structure:

![LineOrder class structure](image)

**Figure 2.2. LineOrder class**

- LineOrder class contains 3 properties: customerId, customerName, priority and an array of lineItems.
- The Header class has customerName, customerNumber, priority as its properties.
- OrderItem objects are stored in a List.

Now it's the time to define the mapping rules to transform the input source java class to the target java class:

- Each instance of the Order class should be mapped with an instance of the LineOrder class.
- The value of customerName of the Header class should be assigned to the customerName property of a LineOrder instance.
- The value of customerNumber of the Header class should be assigned to the customerId property of a LineOrder instance.
• The value of priority of the Header class should be assigned to the priority property of LineOrder instance.

• For the orderItems List of the Order instance, it's necessary to map a new lineItems array instance of LineOrder, and each OrderItem instance in the orderItems List should be transformed to a LineItem instance, those instances will be put into the lineItems array.

• The value of productId of the OrderItem instance should be assigned to the 'productCode' of LineItem.

• The value of price of OrderItem should be assigned to the unitPrice of LineItem.

2.3. Smooks Configuration File Wizard

Select the Smooks Java-to-Java example project that you created and right-click, select the menu New > Other, then find Smooks > Smooks Configuration File. Click the Next button.

Figure 2.3. Selecting Smooks Configuration File Wizard

The first 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 2.4. Choosing the configuration file container and the file name

The second wizard page is allows you to select Smooks configuration file version. Select the appropriate one and click Finish to complete the wizard.

2.4. Smooks Editor

When all the wizard steps are passed, a new Smooks Configuration file is created and the Smooks Configuration Editor will be opened automatically:
Figure 2.5. Smooks Editor

On the bottom of this editor there are 5 tabs, they are: Overview, Reader, Message Filter, Graph, Source.

2.4.1. Adding the input data

At first, we need to add the Order class as the input data. To do this, click the Reader tab, switch to the Reader/Input editor page.

To choose the transform data you should open Reader tab and click Add button in the Input Data section.
Figure 2.6. Input Data section

After clicking the button the Data Type Selection wizard is opened, and there are many data types in the list: Java, XML etc.

Note

There will become more and more data types on the list as the development goes on.
Chapter 2. Quick Start

Figure 2.7. Data Type Wizard

Select the Java data type and click Next, you will see the JavaBean class selection page:

Figure 2.8. JavaBean Data Selection

Let's select the Order class from the example project as the source JavaBean class. After that click Finish and you should see the just added java class, being displayed in the Input Data Section.
Figure 2.9. Displaying the Structure of the Transformed Data

2.4.2. Adding the target Bean

After adding the input data, let's switch to the Message Filter page:
Figure 2.10. Displaying the Structure of the Transformed Data

Click the Add button, the Add Child wizard page will be shown, let's select the Java Binding v1.2 -> Bean v1.2 and click OK to add the Bean:
Then we can find on the Message Filter page the Bean v1.2 item added, click on it, a Bean v1.2 section should appear to the right of the page. The section represents a GUI for editing Bean v1.2 properties:

![Figure 2.11. Add Child wizard](image)
Figure 2.12. Bean v1.2 section

You should input `lineOrder` as the Bean's id in the "Bean Id" field. Then you should click Browse button near the Class textfield and select LineOrder as the Bean's class.

**Note**

When you enter a value for the Class text field, you can type 'Alt + /' and the content assist pop dialog will be shown, so it's will increase the efficiency. If 'Alt + /' doesn't work, try to use the 'Ctrl + space'('Key Assist' of Eclipse).

Let's click the Add Binding link at the bottom of this page, it will show a dialog to ask the binding of what Bean properties you will prefer to add. Select all and click OK, you will see a list of Binding elements added under the Bean item.

Then you should add an array of beans called `linelitems` and a Bean LinelItem. These two classes and Id values are:

Array Bean:

- `BeanId : linelitems;`
Chapter 2. Quick Start

• Class : example.trgmodel.LineItem[]

LineItem Bean :

• BeanId : lineItem;
• Class : example.trgmodel.LineItem

Note

In Smooks, to be an array of Beans, the class value should end with '[]'. This means that this bean is an array type.

To finish this work, use the Add Binding link to add all the bindings for the Bean.

2.5. Mapping

How to map the java class just added?

First, you should know what to transform:

a. Bind the "customerName" property of the Order#header to the "customerName" property of LineOrder.

b. Bind the "customerNumber" property of the Order#header to the "customerId" property of LineOrder.

c. Bind the "price" property of the OrderItem to the "unitPrice" property of LineItem.

d. Bind the "productId" property of the OrderItem to the "productCode" property of LineItem.

e. Bind the "quantity" property of the OrderItem to the "unitQuantity" property of LineItem.

Make sense? You should open Design tab and click Add button. In opened Add Child Wizard, please, select Add Smooks Resource>Java Binding>Bindings.

First of all, you should bind Order to LineOrder. To do this one Bindings Item should be added to the Resource List.
Figure 2.13. Mapping

Click on it and you should see **Bindings** Section to the right of **Message Filtering Resources**. Write `lineOrder` in the **Bean id** field.

Select `example.trgmodel.LineOrder` in the **Class** field and `example.srcmodel.Order` in the **Name** field. Your **Smooks Configuration Editor** should look like this:
Figure 2.14. Mapping

Now you should bind two values: "customerNumber" property of the Order#header to the "customerId" property of LineOrder. To do this right click your just created Bindings Item > Add Smooks Resource > Value.
Figure 2.15. Mapping Editor

Note:

Please, remember that the root nodes of the source data and target data *MUST* be connected.

This rule is not only for Java-to-Java, but also is relevant for XML-to-Java.

Do the same operation with the other properties in this and other classes.

It's also necessary to bind `orderItems` property of the `Order` to `lineltems` property of `LineOrder`. The difference from the previous steps is that the `orderItems` is presented by the list of `Order` and `lineltems` is the massive of `LinItem`. To do the binding, you should use `Wiring Item`. First you should create a new Binding item. This is a configuration that allows you to construct an object graph. Select `example.trgmodel.Lineltem[]` in the Class field and `orderItems` in the Name field, in bean Id write `lineltems`. Your Smooks Configuration Editor should look like this:
Figure 2.16. Creating Wiring Item

Then you should right-click `lineOrder` binding item >Add Smooks Resource>Wiring. Select `lineItem` in Bean id Ref.

The final step is to add new Wiring item to lineOrder Binding. In the Bean id Ref select `lineltems` and in Property select `lineltems`.

Now, you should save the file and the Smooks tools generate the correct configuration file content.

OK, let run the test class: `example.Main`. (Please, make sure that the Smooks engine loads the correct configuration file).

Success!
Smooks Design Page

In this section you will explore the details of Smooks Design Page usage.

Smooks Design Page is one tab of the Smooks Editor.

Figure 3.1. Design Page

This page consists of 3 sections: Message Filtering Resources, Input Data and the section devoted to the configurations of a selected in a Message Filtering Resources section Item.

You can bind classes using Bindings item on the Design page of the Smooks Editor.

Resource list item in Message Filtering Resources corresponds to the root element of the smooks-config.xml file. You can configure the next four properties using corresponding fields:

- **default-selector**: Defines the selector that will be applied to all resource-config elements in the smooks-config.xml, where a selector is not defined.
- **default-selector-namespace**: Defines the default selector namespace.
• **default-target-profile**: Defines the default target profile that will be applied to all resources in the smooks-config.xml.

• **default-condition-ref**: The condition is applied to resources where an empty "condition" element (i.e. `<condition/>`) is defined. The condition element does not reference a globally defined condition.

The configuration editor also edits the [Smooks parse type](#) and manages the [Import Files](#).

### 3.1. Selector generate dialog

Using [Selector generate dialog](#) you can allow the user to select a node for generation "selector" for smooks-config file. To open the dialog you should left click [Browse](#) button in [Default Selector](#) section.

![Selector generate dialog](image)

**Figure 3.2. Selector generate dialog**

When the dialog is opened you should click "Click to add input data" to set the selector. In the opened [Data Type Selection](#) Wizard you should choose the type of the file and the file itself you want to load to the [Selector generate dialog](#). After that your dialog should look nearly like that:
Figure 3.3. Selector generate dialog after adding Input Data

The final step is to choose the node you want to make a selector for `smooks-config.xml`.

### 3.2. Smooks Parse Type

If your config file doesn't contain `Global Parameters` item yet, you should add it by right click `Resource List>Add Smooks Resource>Global Parameters`. After this step you should add `Param` to just created `Global Parameters`. 
Figure 3.4. Adding Param to created Global Parameters

Then left click *Param* and in the Param section, you will see a set of text fields: *Name*, *Type* and *Text*. *Name* value is required. To set a parse type you should write SAX/DOM to the *Text* field.
Figure 3.5. Filling Param section field Name and Text Value for set a Parse Type

To apply changes to the `smooks-config.xml` click `Save` button (Ctrl+S).

3.3. Import Smooks Configuration page

Right click Resource List>Add Smooks Resource>Import Smooks Configuration, then you should see a new Import Smooks Configuration item in the list (see the figure below).
Figure 3.6. Import Data

Left click the added item and to the left of Message Filtering Resources Import Smooks Configuration section should appear. Click the Browse button and select the file you want to import (see File selection section).

Click Save and the <import> tag with the corresponding file path as a file property will be added to the smooks-config.xml.

3.4. File selection

Some attributes in Smooks elements should store path to a file as a value (e.g. attribute "mappingModel" of the element "edi:reader"). You can see a Browse button near the fields, that set such a value to the attribute (e.g. "Mapping Model" field set a value to the "mappingModel" attribute).
When you click *Browse*, the dialog prompts you click one of two buttons: *Browse File System* and *Browse WorkSpace*.

Using the buttons you can locate the file you need. If you click *Browse File System* button you specify the path of, or browse to, a file system directory.
Figure 3.9. Browse file system

If you click *Browse Workspace* button, you specify the path of, or browse to, a workspace relative working directory.

Figure 3.10. Browse Workspace
Smooks Source Editor Page

This section provides information about Smooks Source Editor Page.

4.1. XML Source Editor

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

![Figure 4.1. Graphical Editor](image)

4.2. 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.
4.3. 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 4.3. 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 4.4. Smooks Configuration File Validator Settings

We hope, our guide will help you to get started with JBoss Smooks Tools. Besides, for additional information you are welcome on [JBoss forum](http://www.jboss.com/index.html?module=bb&op=viewforum&f=201).