Writing SECONDO-Algebras for Java-only Programmers

Thomas Behr

January 11, 2005

Abstract

This article describes how algebras written in Java can be used in the SECONDO-frame using the `JavaWrapper` tool.

1 What is JavaWrapper

`JavaWrapper` is a little tool writen in Java to make Java classes useable in SECONDO. It takes some classes and methods and creates an algebra from them using Jni for calling the Java code.

2 Requirements to the Java Classes and Methods

A Java class which should be used as an algebra type have to implement the `AlgebraType` interface. Additionally a constructor without any parameter must exist. From other classes only `static` methods can be wrapped as a SECONDO-operator. Static methods can be used as operator if the return type, and all parameter types can be an algebra type or are elements of the `StandardAlgebra` of SECONDO. E.g. methods using characters as parameter can't be wrapped because `char` is not a type of the StandardAlgebra. For static methods also the declaring class must fulfil this condition.

For the reason of persistent storing of the objects, all used classes have to implement the `Serializable` interface.

2.1 The AlgebraType Interface

The interface `AlgebraType` extends the interfaces `Serializable` and `Comparable`. For the `Serializable` interface no method must be implemented, but all internal used classes have also to implement this interface.

The methods to implement are:

`Object copy()`	Returns a depth copy of this object.
`compareTo(o)`	Returns -1 if `this` is smaller than o, 1 if `this` is greater than o and 0 in the case of equality. This method is inherited from the `Comparable` interface.
`int getHashValue()`	Computes a hash value for this object and returns it.
`boolean loadFrom(type, instance)`	Reads the value of this object from instance, possible using the typeInfo. The return value is `true` if the list can be readed successfully.
`ListExpr toListExpr(typeInfo)`	Returns the nested list representation of this object.
`boolean checkType(ListExpr LE)`	Checks whether LE describes this type.

3 The JavaWrapper Tool

3.1 Starting the Tool

Because the JavaWrapper is written in Java you need a Java interpreter to start it. First, you should create a new directory for your algebra in the `Algebras` directory of SECONDO. In this directory create a further directory and copy your Java classes into it. The tool should (not must) be started in the new created algebra directory. The classpath must be set correct. This means, the paths to SECONDO's `Javagui`, the path to the `wrapper` directory `$SECONDO_BUILD_DIR/Tools/Jni` and the path to your Java-Classes must be included.

A possible call is:

```
export CP=$SECONDO_BUILD_DIR/Tools/Jni:$SECONDO_BUILD_DIR/Javagui:.
java -classpath $CP wrapper.JavaWrapper
```

3.2 Warning

Please be carefully in filling out all fields, the $\boxed{\texttt{Back}}$ button is out of function in the current version of JavaWrapper.

3.3 Selection the Algebra Name and the Classes

If JavaWrapper has been started, you have first to input the name of the algebra. After that, you can include the classes by typing in the name of the class and pressing the $\boxed{\texttt{add}}$ button. You can also remove a class by selecting it and pressing the $\boxed{\texttt{remove}}$ button.

If all classes are selected, press the $\boxed{\texttt{Next}}$ button.

3.4 Type Descriptions

If you have selected classes implementing the `AlgebraType` interface, you have now to enter the type description by filling out the table in the screen. This descriprions are used in SECONDO's type descriptions, e.g. when `list type constructors` is called. If all type descriptions are correct, press the $\boxed{\texttt{Next}}$ button.

3.5 Selecting Methods

At the left side of this panel, all methods are listened which can be used as operators in SECONDO. Select all desired methods and press the $\boxed{\texttt{add}}$ button. If you are ready, press the $\boxed{\texttt{Next}}$ button.

3.6 Operator Specifications

Like for type constructors, you have to enter the descriptions for the operators. Please fill out this table carefully because this descriptions is the only source for an user of SECONDO to get informations about an operator. After that, press the $\boxed{\texttt{Next}}$ button.

3.7 Creating the Algebra

Now all required information is available. Please select a filename and press the $\boxed{\texttt{Save}}$ button. If the algebra is writted, you can finish JavaWrapper by pressing the $\boxed{\texttt{X}}$ at the top of the window.

4 Include the Algebra into Secondo

The new created algebra must be included into SECONDO. First copy the makefile from the `JBBoxAlgebra` into your algebra directory and change the `MODNAME` as well as the directory of the target `alljava` to the correct values. Include makefiles to compile (and clean) your java files in the directories of them. Remember to set the classpath correct. After that, insert an entry into the `AlgebraList.i.cfg` file located the `Algebras/Management` directory of SECONDO. Insert also two entries in the file `makefile.algebras` in SECONDO's home directory. The entries have the form

```
JNI_ALGEBRA_DIRS := <AlgebraDirName>
JNIALGEBRAS      := <AlgebraName>
```

if this is the first algebra using JNI, and the form

```
JNI_ALGEBRA_DIRS += <AlgebraDirName>
JNIALGEBRAS      += <AlgebraName>
```

if an algebra using jni already exists.

Now you can enter:

```
make alg=auto
```

in the home directory of SECONDOto include your new Algebra.