

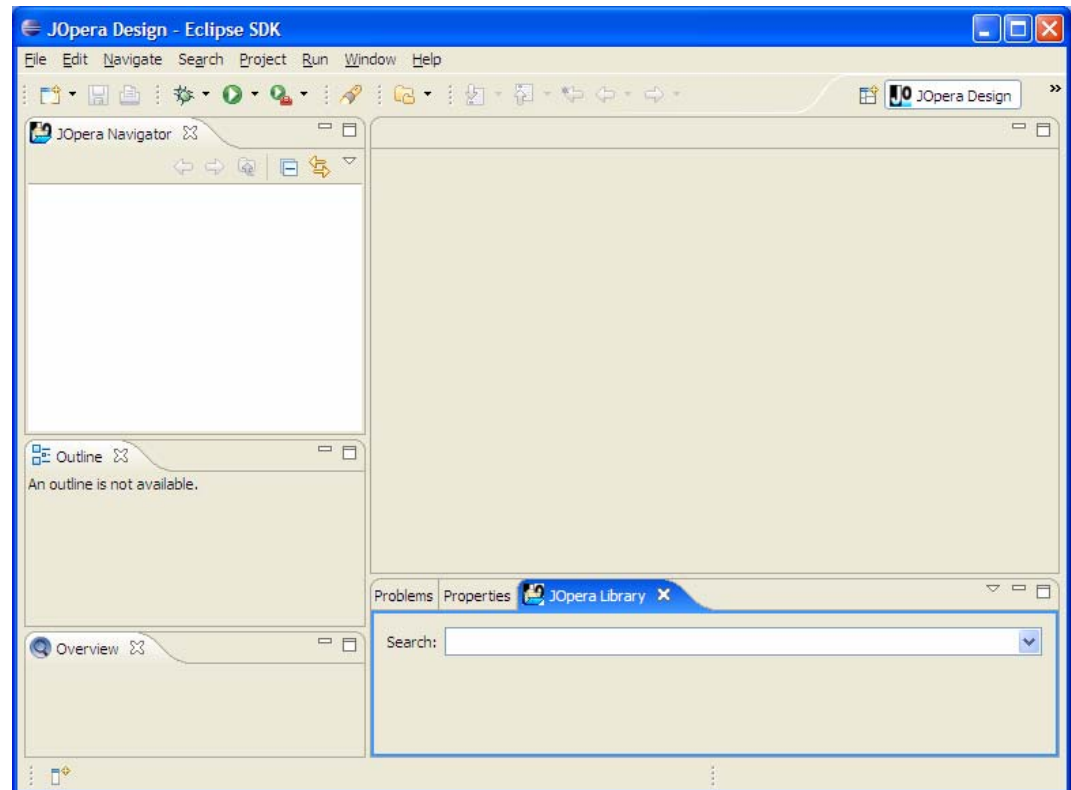


JOpera Overview

- JOpera is a visual Workflow Management Tool
- It is primarily used for Scientific Workflows and Web service composition
- It is built on the Eclipse platform (so that you can easily extend it writing Eclipse plugins)
- It features process development, execution and monitoring tools
- It was recently extended to support streaming

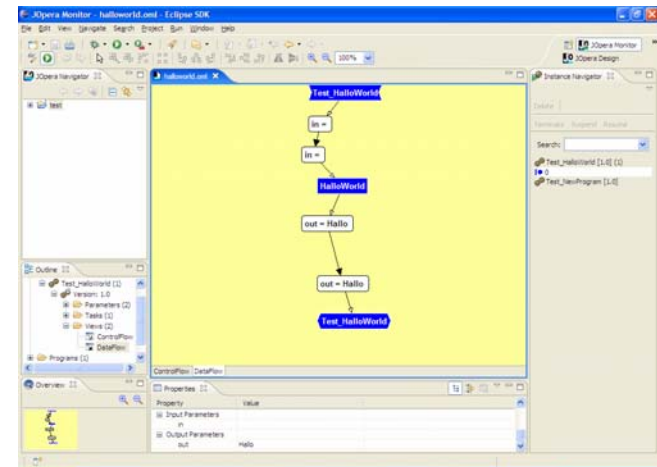
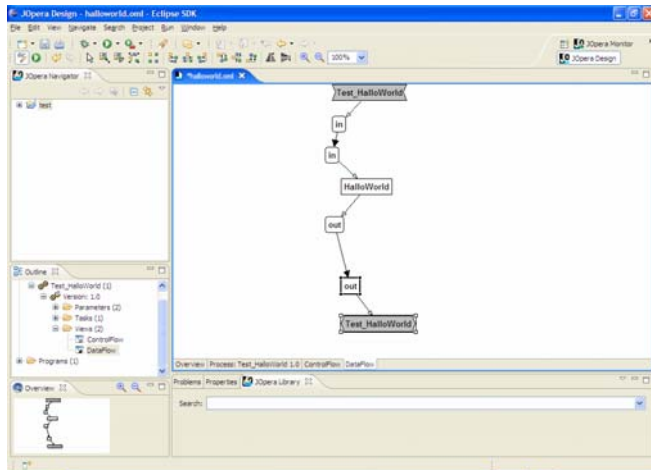
JOpera Visual Components

- JOpera Navigator
- Outline
- Overview
- JOperaLibrary
- JOpera Editor
- JOpera Monitor



JOpera Editor/Monitor

- Two different modes:
 - One for editing processes, one for monitoring execution
- Monitoring mode also has
 - Instance Navigator
 - Kernel Memory Inspector





Building Blocks

- Process:
 - Tasks
 - Control Flow
 - Data Flow
 - Input- / Outputparameters
 - Constants

- Tasks
 - Processes
 - Programs

- Programs/Components
 - Adapter
 - Input- / Outputparameters



Data Flow

- Connects Input / Output Parameters and Constants
- Input Parameters of the Process connect with Input parameters of Tasks
- Output Parameters of Tasks with Input Parameters of Tasks or with Output Parameters of the Process
- Constants can only connect with Input parameters of tasks or Output parameters of Processes
- Watch out for data types!

Control Flow

- Mostly inferred from the Data Flow
- Exceptions are loops or more advanced workflow patterns
- Conditions can be used to determine when a task gets executed:
- In the Process View: (you can only depend on input of processes and output of tasks)

The screenshot shows a configuration window for a task named 'NewProgram'. On the left, a tree view under 'Tasks' shows a folder 'Activities (1)' containing 'NewProgram', and a folder 'SubProcesses (0)'. On the right, the configuration fields are:

- Name:** NewProgram
- Invoke:** {sq}NewProgram[1,0] (with a 'Browse' button)
- Activator:** (empty field)
- Condition:** PROC.in > 100
- Time Out:** 0
- Description:** (empty text area)



Components

- Web Services
- SQL
- UNIX
- Java
- Java Snippets

- Grid Services
- XML (XQuery/XPath/XSLT)
- SSH
- And many more

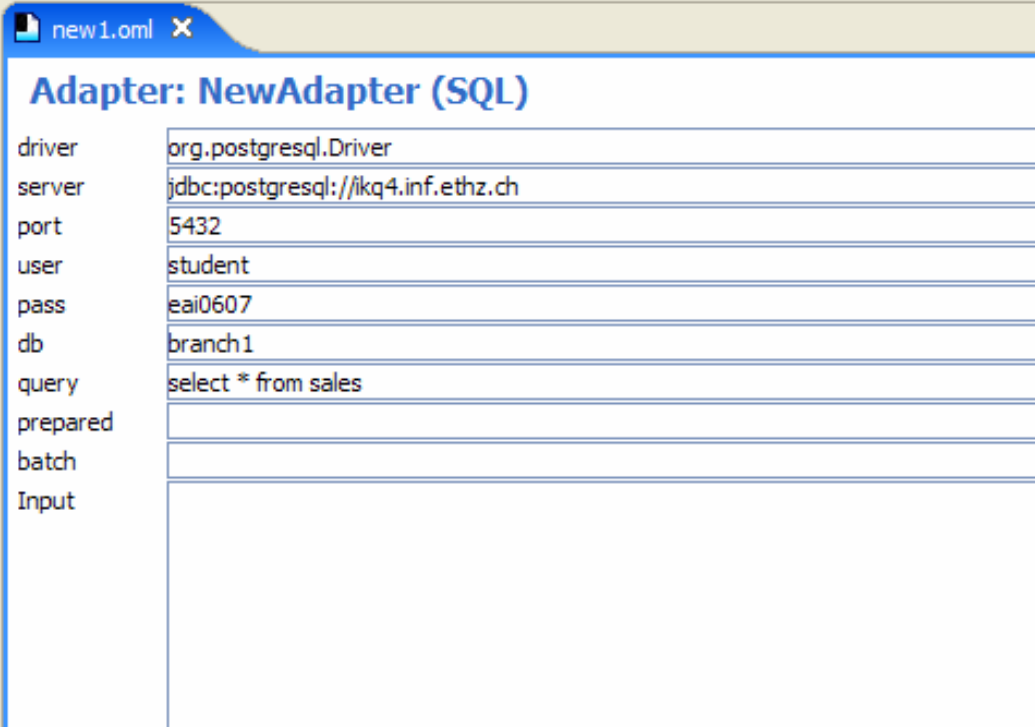
JavaSnippet Component

- Use the Parameters as variable names
- Watch out for types
- Use as Java, but without imports (or use fully qualified names)



SQL Component

- Use Systemparams to handle result and query

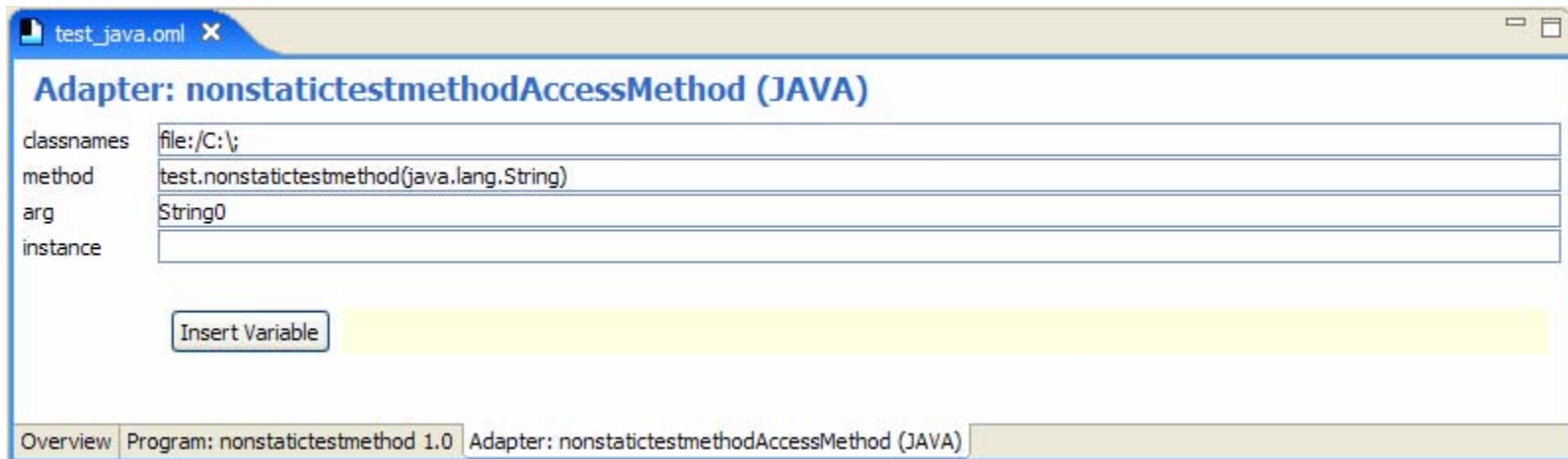


The screenshot shows a web browser window with a tab titled 'new1.oml'. The main content is a form titled 'Adapter: NewAdapter (SQL)'. The form contains several input fields, each with a label on the left and a text input area on the right. The fields are: driver (org.postgresql.Driver), server (jdbc:postgresql://ikq4.inf.ethz.ch), port (5432), user (student), pass (eai0607), db (branch1), query (select * from sales), prepared (empty), batch (empty), and Input (empty).

Label	Value
driver	org.postgresql.Driver
server	jdbc:postgresql://ikq4.inf.ethz.ch
port	5432
user	student
pass	eai0607
db	branch1
query	select * from sales
prepared	
batch	
Input	

Java Component

- Use the Java Import Wizard (File -> Import -> JOpera -> Java Import Wizard)



Web Service Component

- Use the WSDL Import Wizard

CurrencyExchangeService.oaml

Adapter: getRate_INVOKE (INVOKE)

message

```
<soapEnv:Envelope xmlns:soapEnv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:soapEnc="http://schemas.xmlsoap.org/soap/encoding"
<soapEnv:Body>
  <bns:getRate xmlns:bns="urn:xmethods-CurrencyExchange" soapEnv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <country1 xsi:type="xsd:string"%country1%</country1>
    <country2 xsi:type="xsd:string"%country2%</country2>
  </bns:getRate>
</soapEnv:Body>
</soapEnv:Envelope>
```

destination

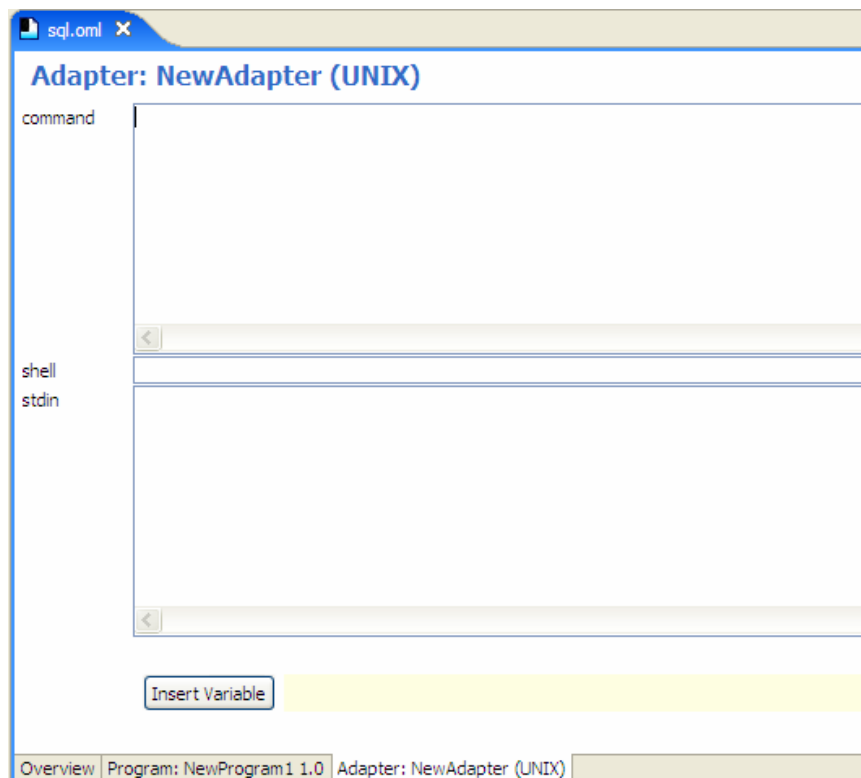
SOAPAction

timeout

Overview Program: getRate 1.0 Adapter: getRate_INVOKE (INVOKE) Adapter: getRate_WSIF (WSIF)

Unix Component

- Also a SSH Component available





Help/Problems

- Online Help System contained in the JOpera installation (Help -> Help Contents -> JOpera)
- Problems view gives good pointers. Also use the Memory Inspector
- Help is not complete
- Send mail to help@jopera.org if there are problems
- Do not forget to include:
 - Comprehensive error description
 - OML Files you need
 - Error messages (including stack traces)



Shortcuts

- Create new project:
 - File -> New -> Project... -> JOpera -> JOpera Project
- Create new OML file:
 - File -> New -> Other... -> JOpera -> OML File
- Start process:
 - Run -> Run... -> JOpera Process -> Browse... (select process) -> Run
- Create new program:
 - Wizard for Java / WebServices
 - Outline -> Programs -> New Program -> Add Parameters & Adapter