

JBoss Community



Lab 3

Goals

- Main Objectives
 - Create a BPM process
 - Build and test

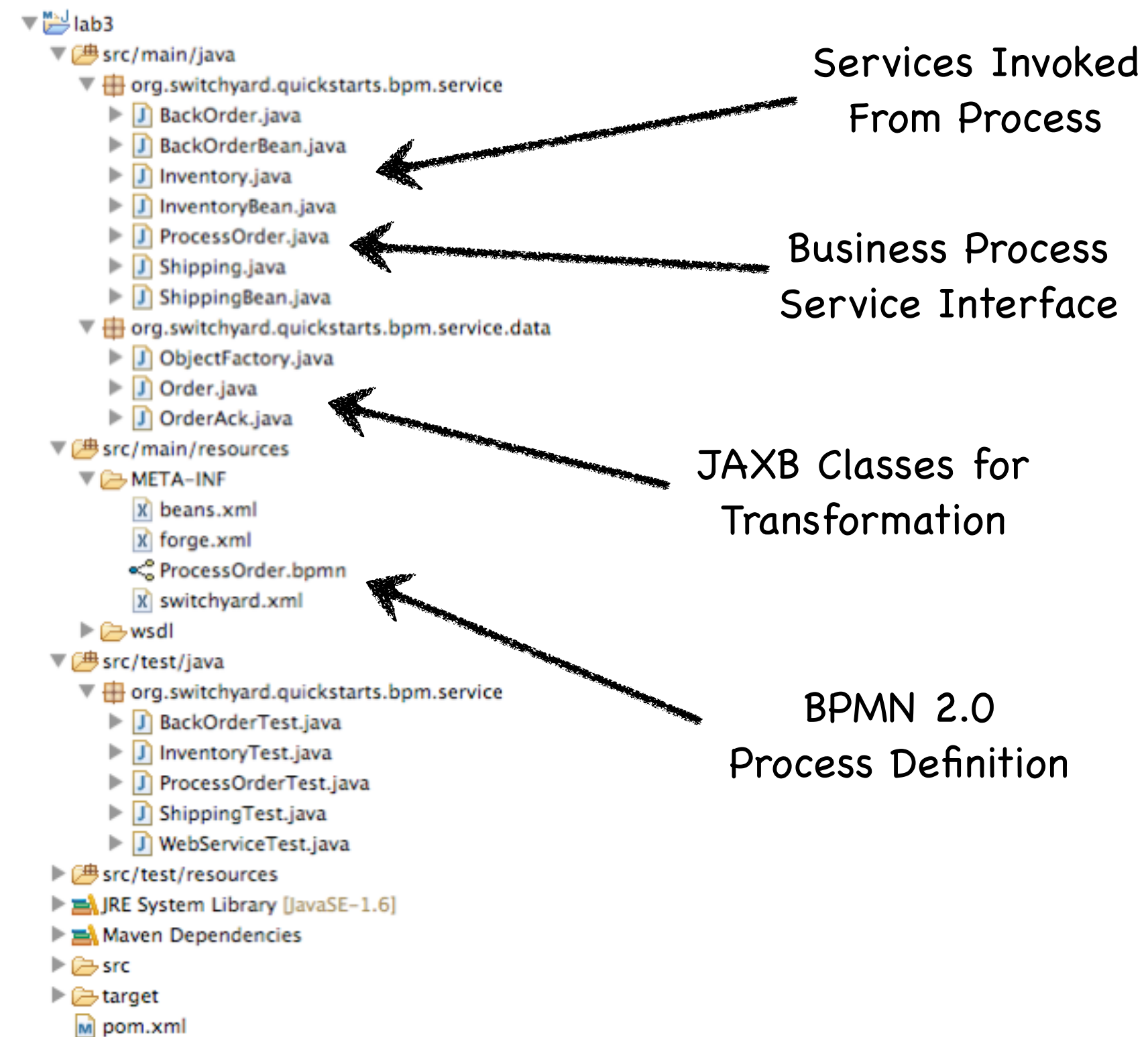
Creating a Business Process Service



Import Lab3 App

- File -> Import
- Maven -> Existing Maven Projects
- Next
- Root Directory : workshop/labs/lab3
- Select lab3
- Next
- [Resolve All Later]
- Finish

What's Inside



Hands On!

- Option 1 - Clean Hands
 - Copy in existing business process definition

```
% cd workshop/labs/lab3  
% cp -R etc/main src/
```

- Option 2 - Dirty Hands
 - Create BPMN 2 definition from scratch

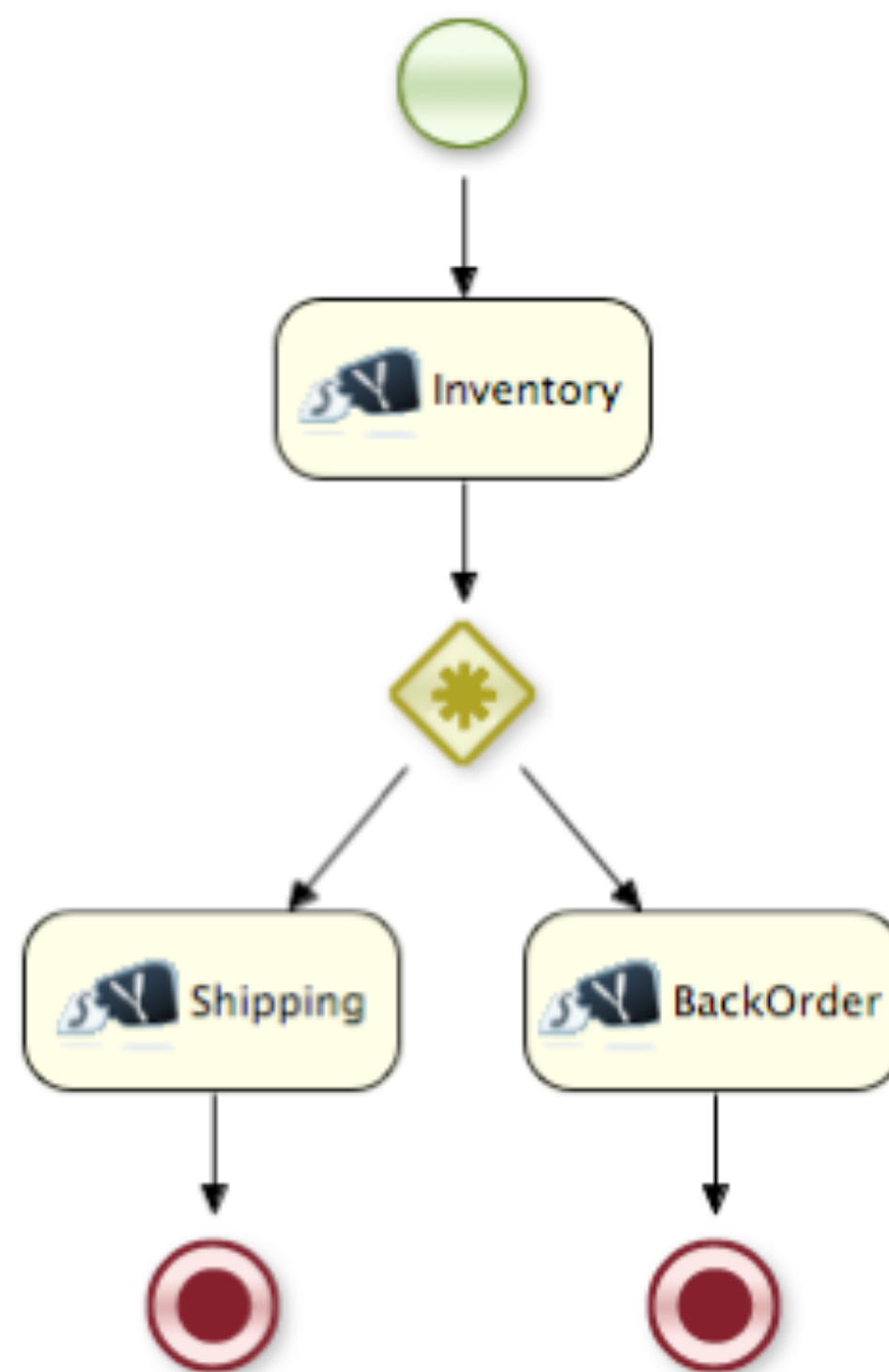
ProcessOrder.bpmn

(before)

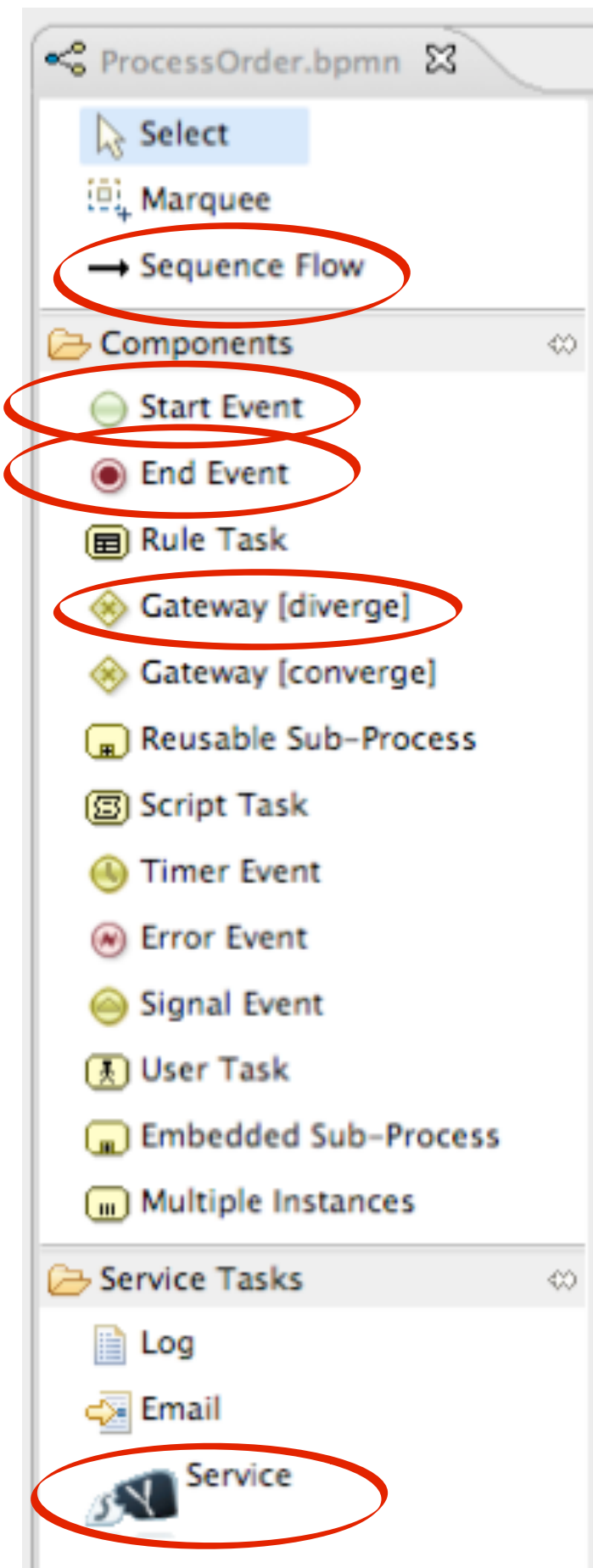


ProcessBuilder.bpmn

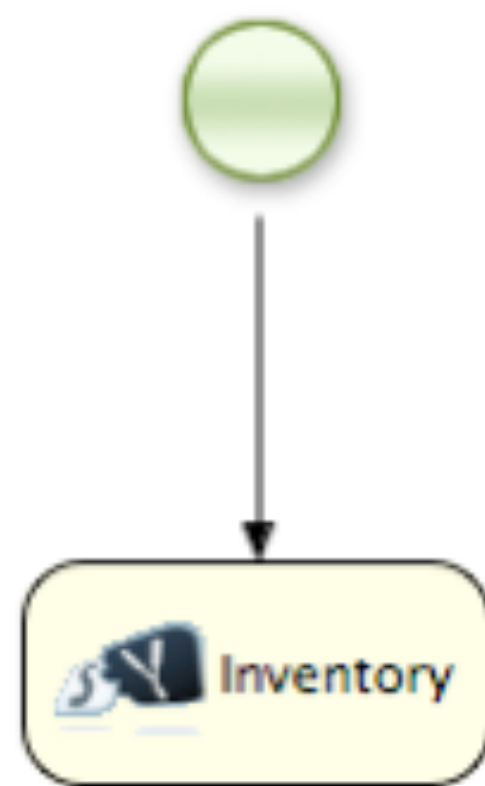
(after)



Widgets We Will Use



Invoke Inventory Service



Property	Value
Id	8
InputMessageVariable	
MetaData	{height=48, width=100, UniqueId=_8, y=145, x=259}
Name	Inventory
On Entry Actions	
On Exit Actions	
OutputMessageVariable	itemAvailable
Parameter Mapping	{}
Result Mapping	{}
ServiceName	Inventory
ServiceOperationName	checkAvailability
Timers	
Wait for completion	true

Maps Result to
Process Variable

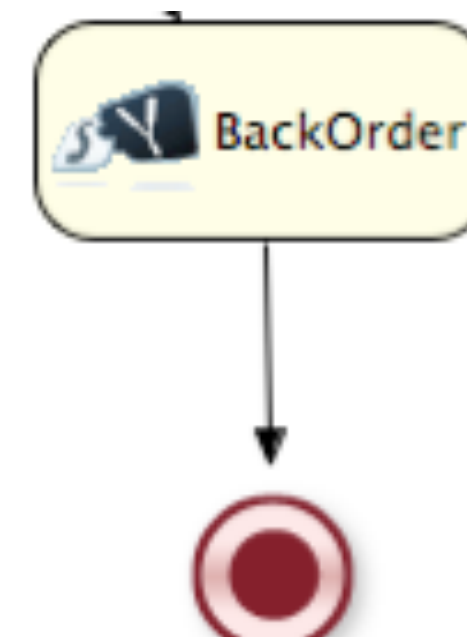
Service Invocation
Details

Invoke Inventory Service

- Properties Sheet for Inventory Service
 - Name : Inventory
 - OutputMessageVariable : itemAvailable
 - ServiceName : Inventory
 - ServiceOperationName : checkAvailability

Shipping and BackOrder Services

Property	Value
Id	9
InputMessageVariable	
MetaData	{height=48, width=100}
Name	Shipping
On Entry Actions	
On Exit Actions	
OutputMessageVariable	
Parameter Mapping	{}
Result Mapping	{}
ServiceName	Shipping
ServiceOperationName	ship
Timers	
Wait for completion	true

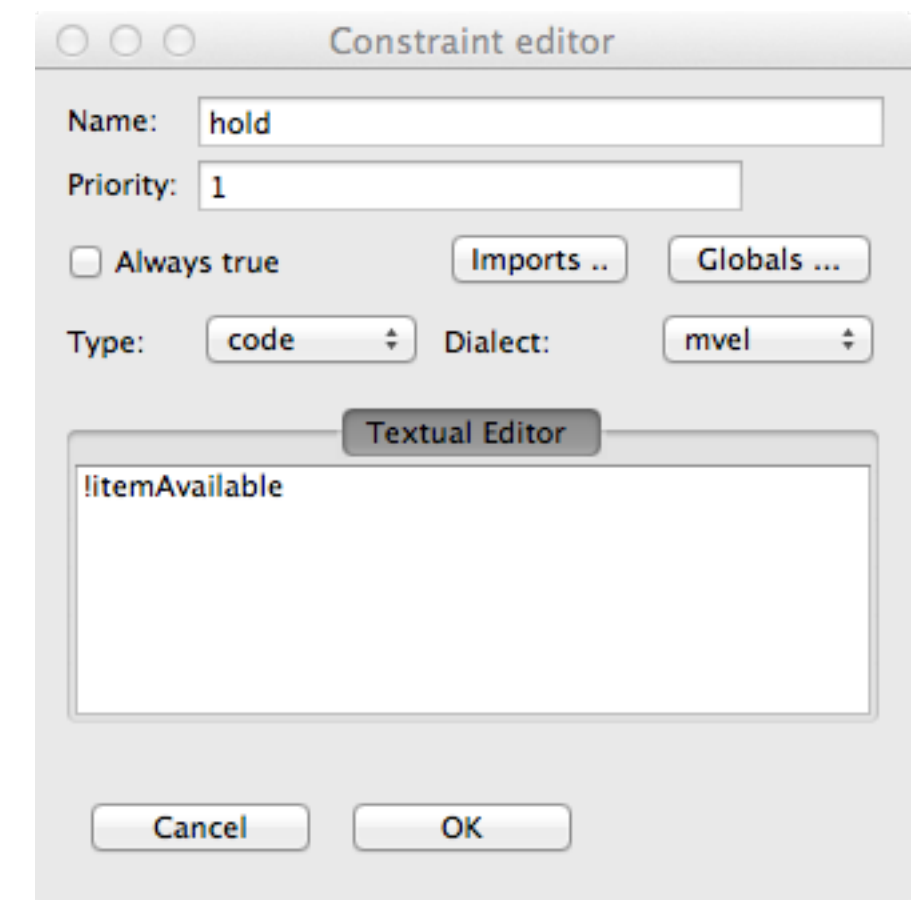
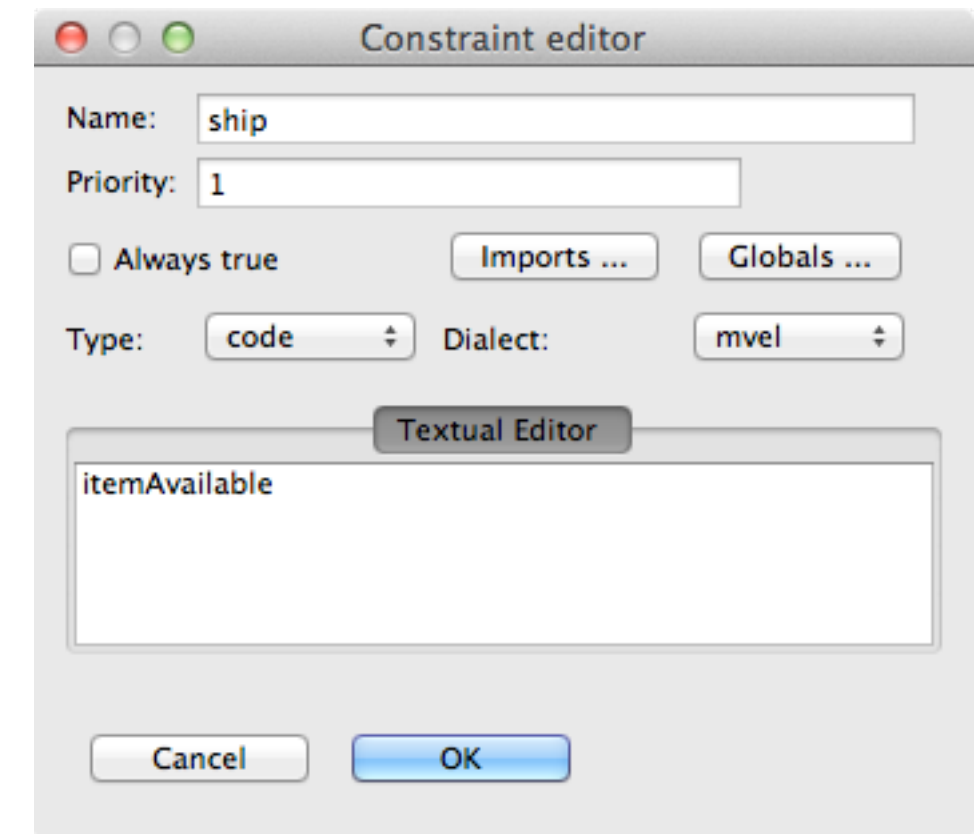
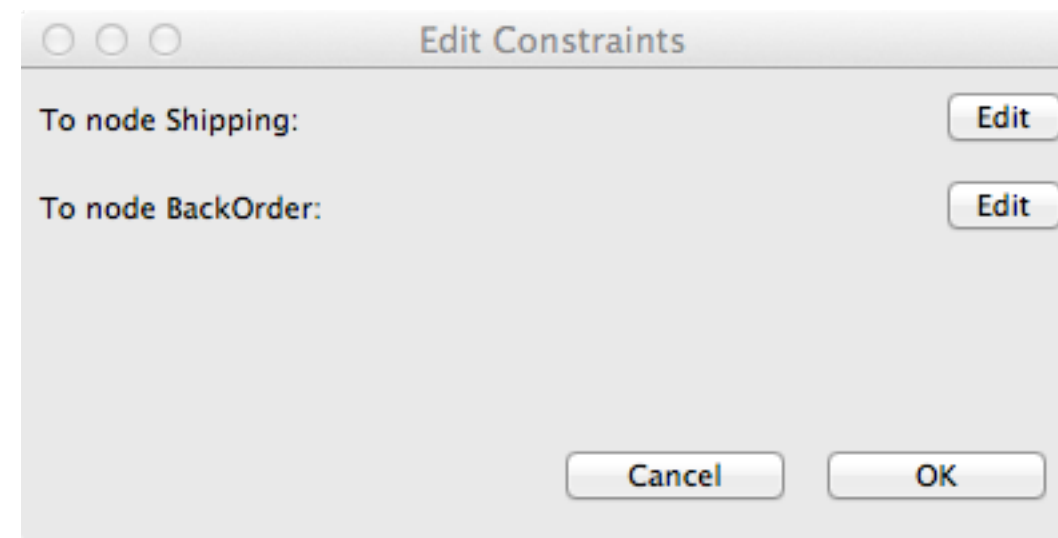
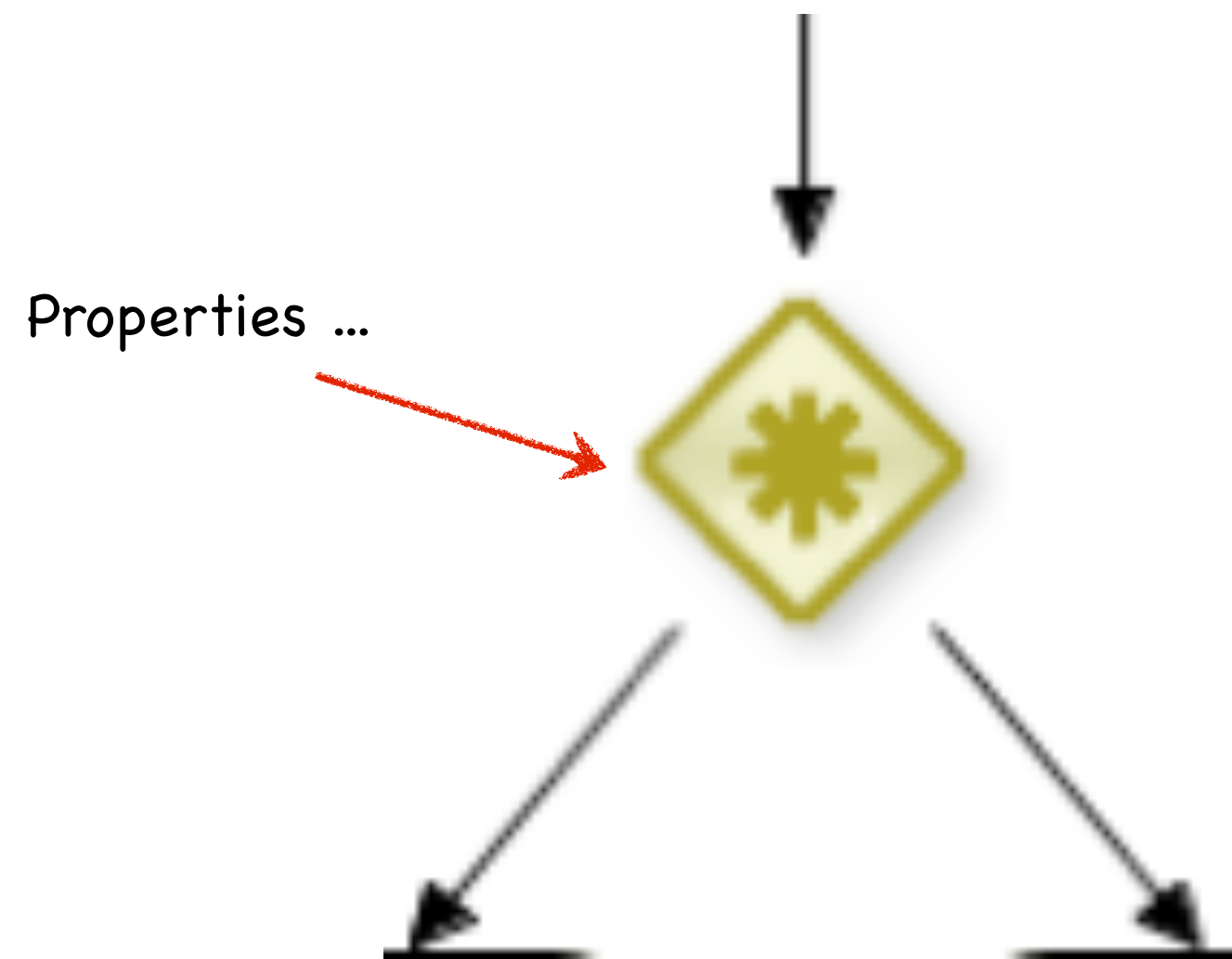


Property	Value
Id	10
InputMessageVariable	
MetaData	{height=48, width=100}
Name	BackOrder
On Entry Actions	
On Exit Actions	
OutputMessageVariable	
Parameter Mapping	{}
Result Mapping	{}
ServiceName	BackOrder
ServiceOperationName	hold
Timers	
Wait for completion	true

Shipping and BackOrder Services

- Properties Sheet for Shipping Service
 - Name : Shipping
 - ServiceName : Shipping
 - ServiceOperationName : ship
- Properties Sheet for BackOrder Service
 - Name : BackOrder
 - ServiceName : BackOrder
 - ServiceOperationName : hold

Define Gateway



Configure Gateway

- Gateway Properties
 - Type : XOR
 - Constraints ...
- To node Shipping
 - name : ship
 - Type : code
 - Textual Editor : itemAvailable
- To node BackOrder
 - name : hold
 - Type : code
 - Textual Editor : !itemAvailable

Note the "!"



Testing the Service

```
% cd workshop/labs/lab3
% mvn test
```

If your AS instance is running, please shut it down

```
#####
Thanks for your order, it has been shipped!
#####
```

(snip ...)

```
#####
Insufficient quantity on hand - order has been placed on hold.
#####
```

Bonus Points



Deploy to AS7

- Start the SwitchYard AS7 runtime

```
% cd switchyard-as7-0.2  
% bin/standalone.sh
```

- Copy application to AS7 deployment directory

```
% cd labs/lab3  
% cp target/lab3-1.0.0.jar ../../switchyard-as7-0.2/standalone/deployments/
```

Create new SoapUI Project

New soapUI Project

Creates a new soapUI Project in this workspace

Project Name: test

Initial WSDL/WADL: http://localhost:18001/ProcessOrder?wsdl **Browse...**

Create Requests: Create sample requests for all operations?

Create TestSuite: Creates a TestSuite for the imported WSDL or WADL

Create MockService: Creates a Web Service Simulation of the imported WSDL

Add REST Service: Opens dialog to create REST Service

Relative Paths: Stores all file paths in project relatively to project file (requires save)

Create Web TestCase: Creates a TestCase with a Web Recording session for functional web testing

OK **Cancel**

Send a Test Message

The screenshot displays the soapUI 3.6.1 interface. On the left, a project tree shows a 'test' project containing a 'ProcessOrderSOAP' service with a 'submitOrder' operation and a 'Request 1' message. The main window shows the details of 'Request 1' at the URL 'http://localhost:18001/ProcessOrder'. The request is a SOAP message with the following XML body:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:submitOrder>
      <orderId>TEST1</orderId>
      <itemId>COWBELL</itemId>
      <quantity>100</quantity>
    </urn:submitOrder>
  </soapenv:Body>
</soapenv:Envelope>
```

Below the request, the response is shown as a SOAP message with the following XML body:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Header/>
  <SOAP-ENV:Body>
    <ns2:submitOrderResponse xmlns:ns2="urn:switchyard-quickstart:bpm-service">
      <orderId>TEST1</orderId>
      <accepted>true</accepted>
      <status>Thanks for your order, it has been shipped!</status>
    </ns2:submitOrderResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The response time is 215ms (377 bytes). At the bottom, there are buttons for 'soapUI log', 'http log', 'jetty log', 'error log', 'wsrm log', and 'memory log'. A 'Request Properties' table is also visible at the bottom left.

Property	Value
Name	Request 1

JBoss Community