

Nuero4j Studio- Debug Plugin.

NEURO4J.ORG



PDF version available online at <http://static.neuro4j.org/download/doc/studio/StudioDebugPlugin.pdf>

ABOUT THIS GUIDE

This guide describes how to use Debug Plugin for Neuro4j Flows .

Online html version available at <http://neuro4j.org/docs/wf/flowdebugtool>

CONTENTS

ABOUT THIS GUIDE	1
OVERVIEW	1
INSTALLATION STEPS	1
CREATING NEW REMOTE DEBUG CONFIGURATION	2
CREATING NEW LOCAL DEBUG CONFIGURATION	5
PERSPECTIVE OVERVIEW	10
ADDING/REMOVING BREAKPOINTS	13
REMOTE DEBUGGING	16

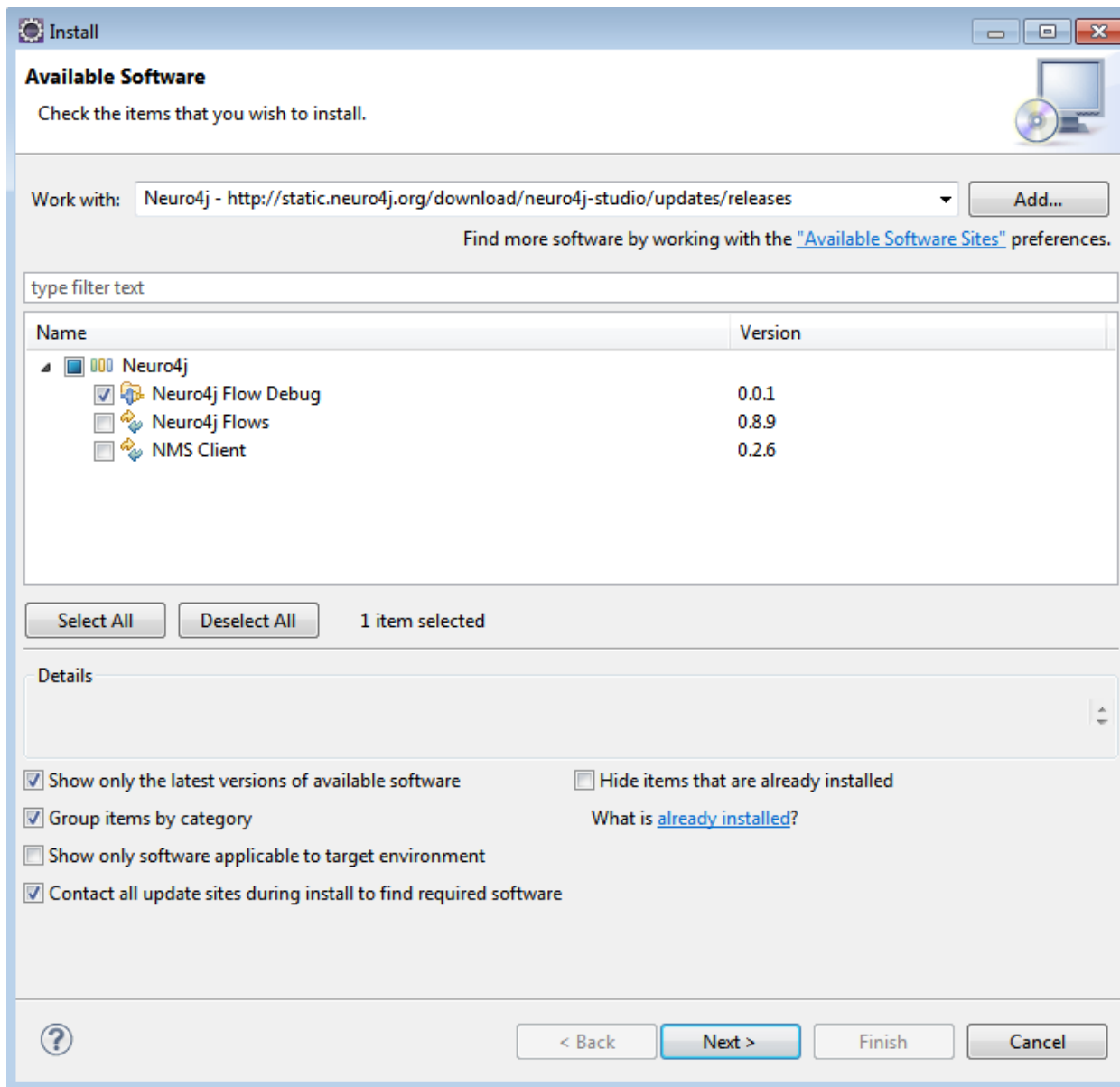
OVERVIEW

Neuro4j Flow Debug Tool is eclipse plugin which allows the developer to track the execution of a flow step-by-step in Neuro4j Studio in order to locate errors precisely.

Using the Debug Tool, developers can easily check the storefront behavior of flows or monitor the status of the flow context at each step using a special view.

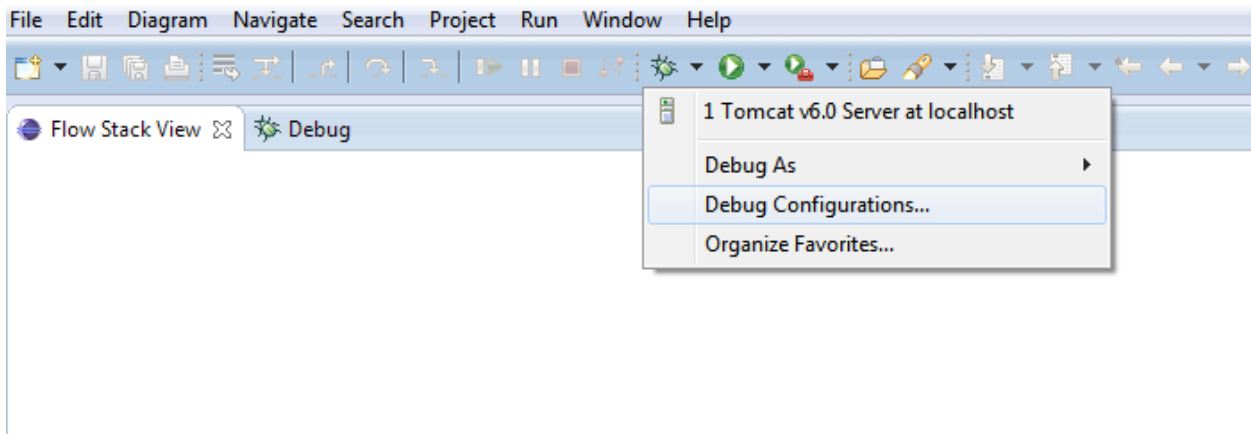
INSTALLATION STEPS

Neuro4j Debug Tool is part of Neuro4j Studio (from version 0.8.9) and can be downloaded from [Download Page](#) or installed using update site.

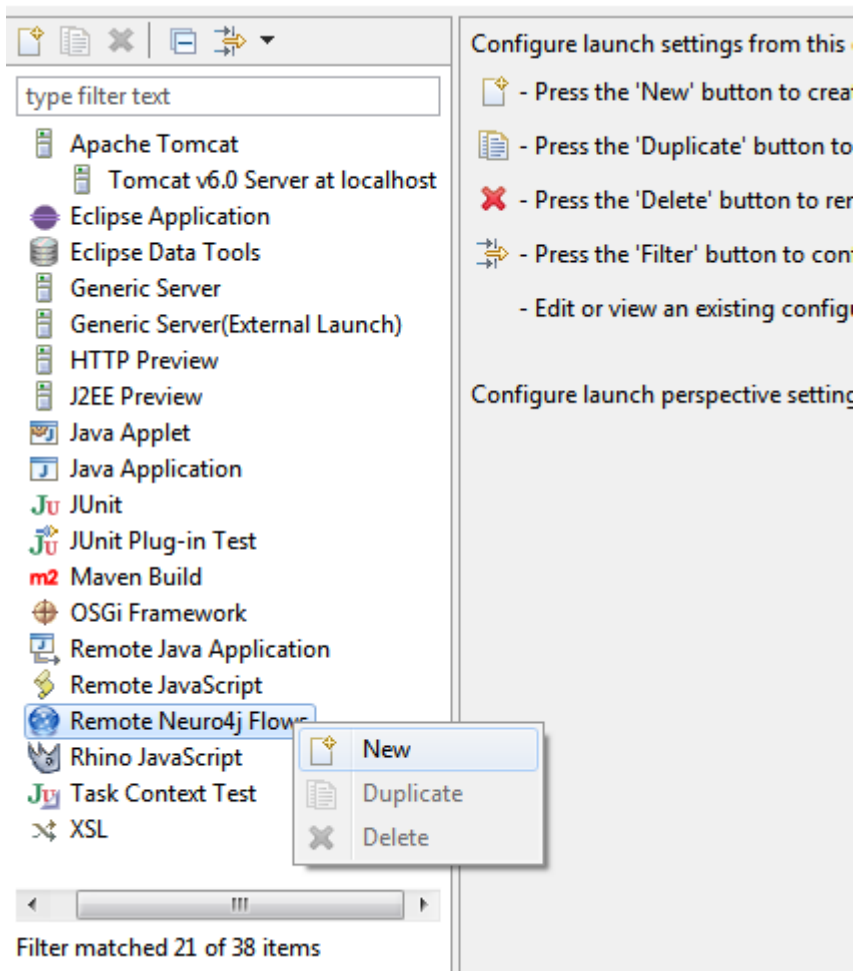


CREATING NEW REMOTE DEBUG CONFIGURATION

- 1) Setup and Run container (ex. Tomcat) in debug mode:
 - a. In file catalina.*
set JPDA_OPTS=-Xdebug -Xnoagent -Xrunjdpw:transport=dt_socket,server=y,address=8888,suspend=n
- 2) Click Debug Configuration...



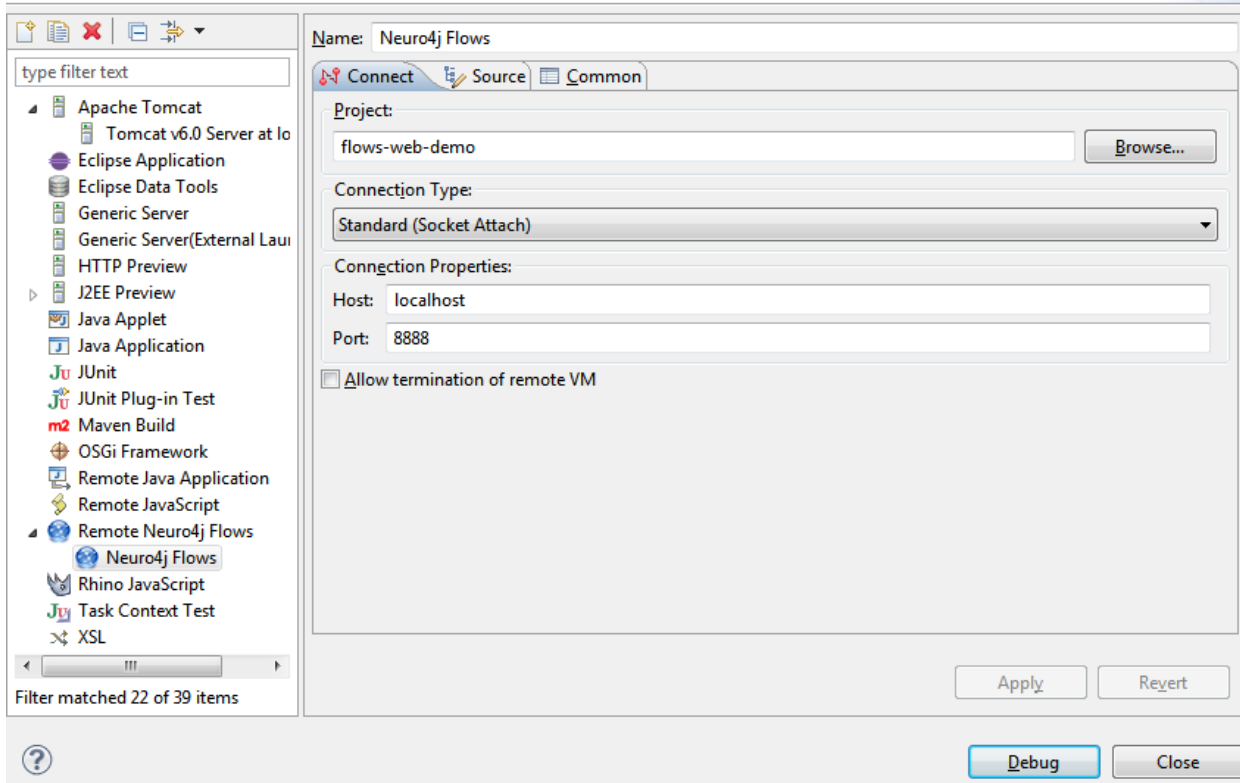
3) Select Remote Neuro4j Flows and click New



4) Set up port and host. Click debug.

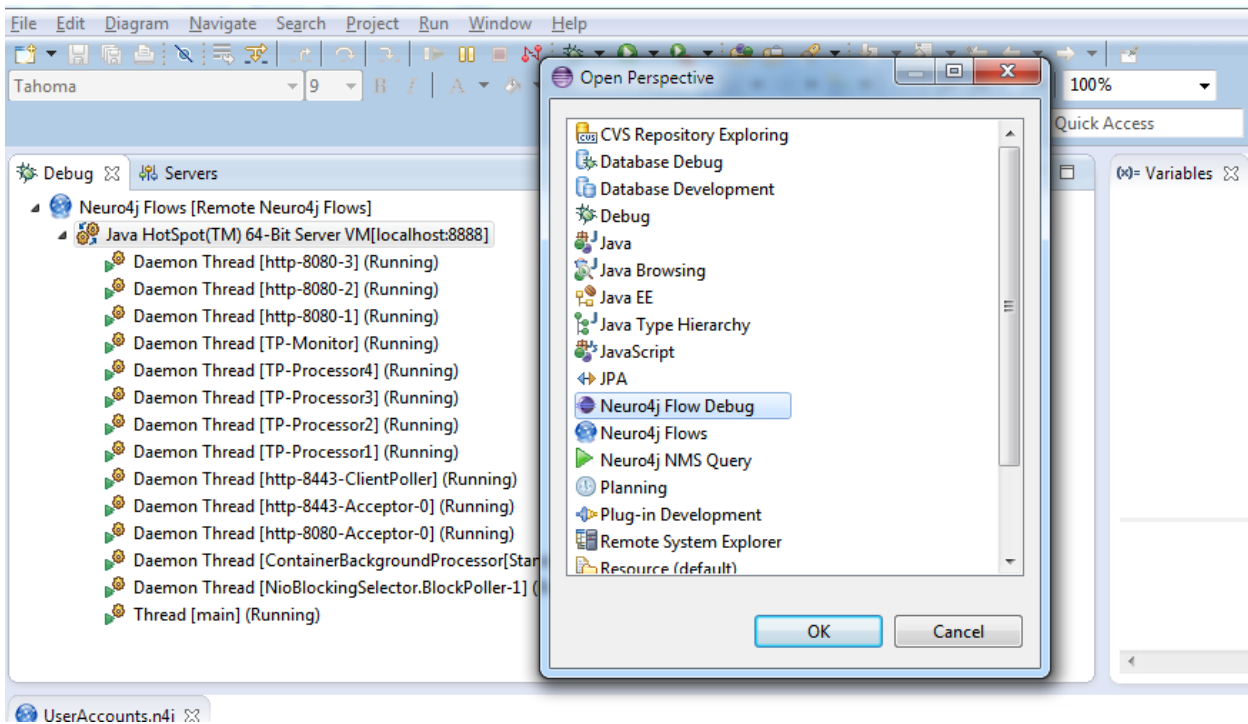
Create, manage, and run configurations

Flow debug description

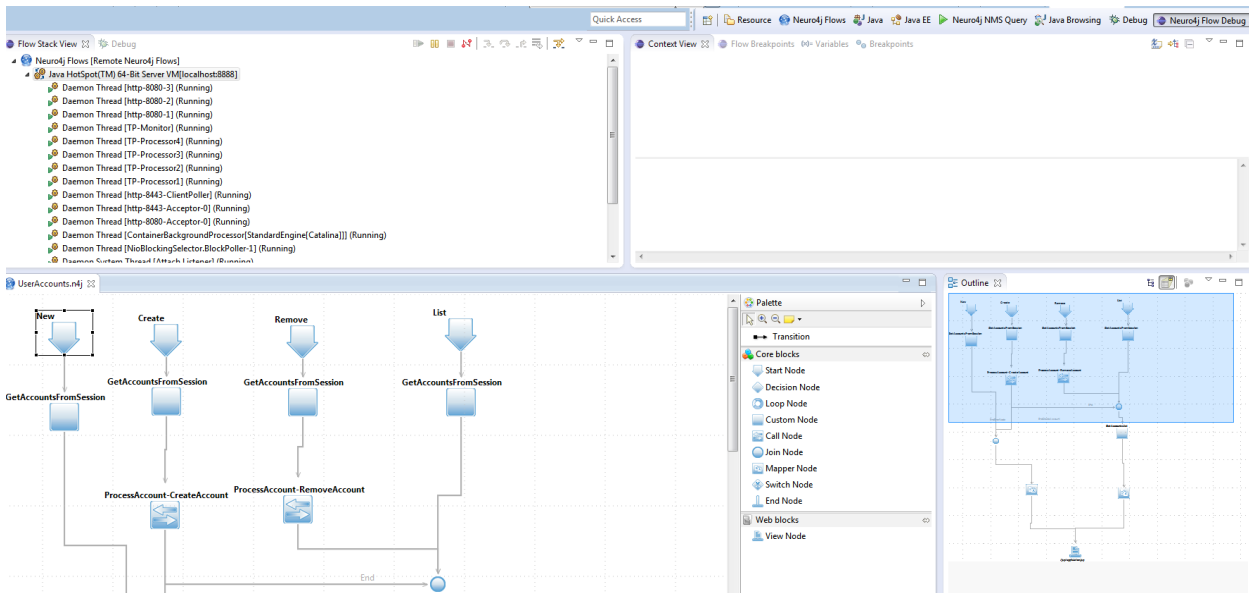



The screenshot shows the 'Run Configurations' dialog in Eclipse. The 'Name' field is 'Neuro4j Flows'. The 'Project' is 'flows-web-demo'. The 'Connection Type' is 'Standard (Socket Attach)'. The 'Host' is 'localhost' and the 'Port' is '8888'. There is a checkbox for 'Allow termination of remote VM' which is currently unchecked. The 'Apply' and 'Revert' buttons are visible at the bottom right. A 'Debug' button is also present at the bottom center.

5) Switch to “Neuro4j Flow Debug” perspective;

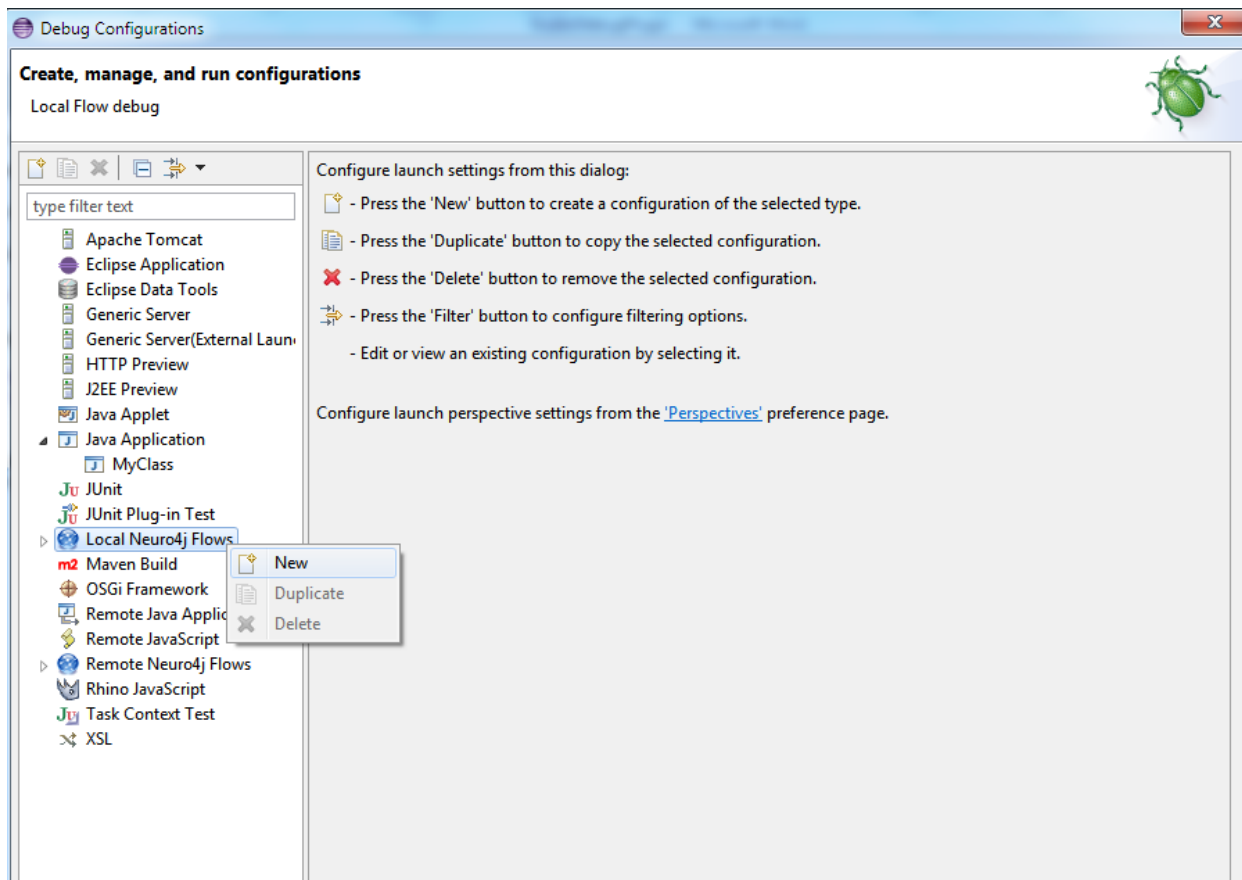


The screenshot shows the Eclipse IDE with the 'Open Perspective' dialog open. The dialog lists various perspectives, and 'Neuro4j Flow Debug' is selected. The background shows the 'Servers' view with a list of running threads for the 'Neuro4j Flows' configuration, including 'Java HotSpot(TM) 64-Bit Server VM[localhost:8888]' and several 'Daemon Thread' instances.

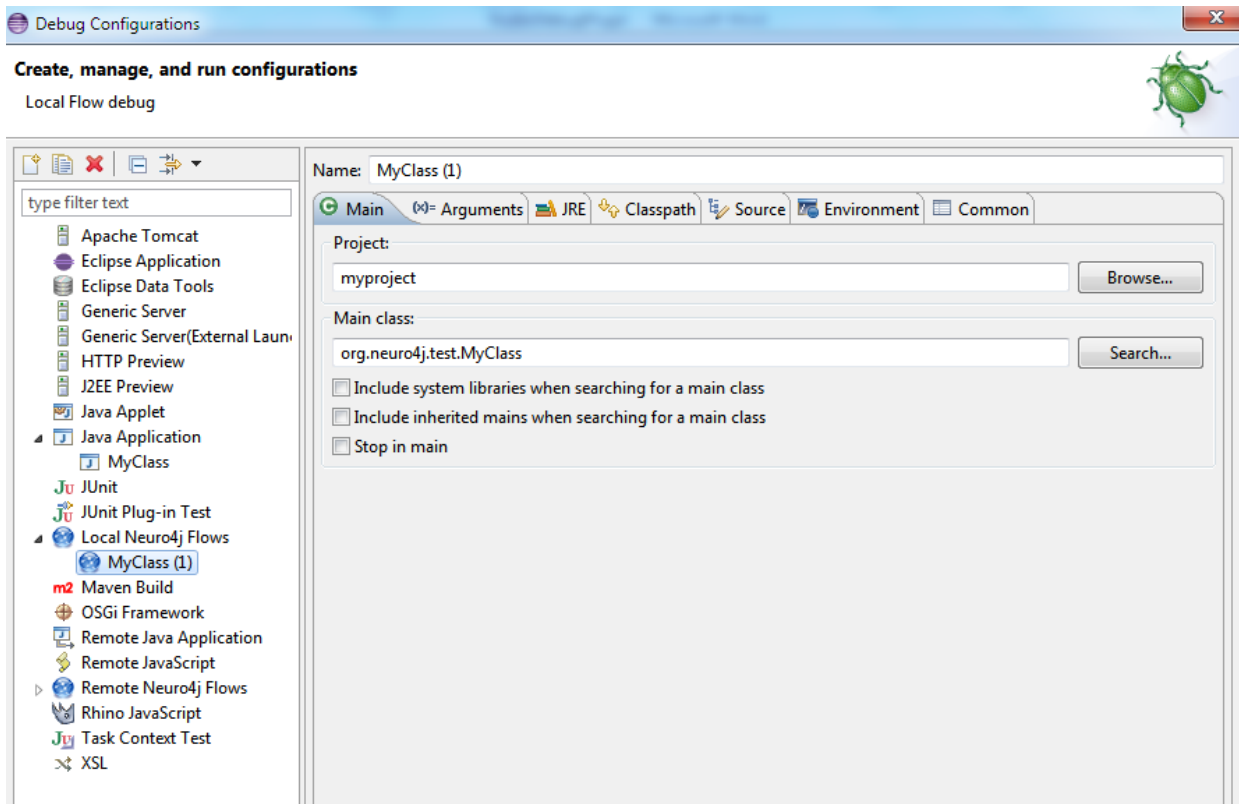


CREATING NEW LOCAL DEBUG CONFIGURATION

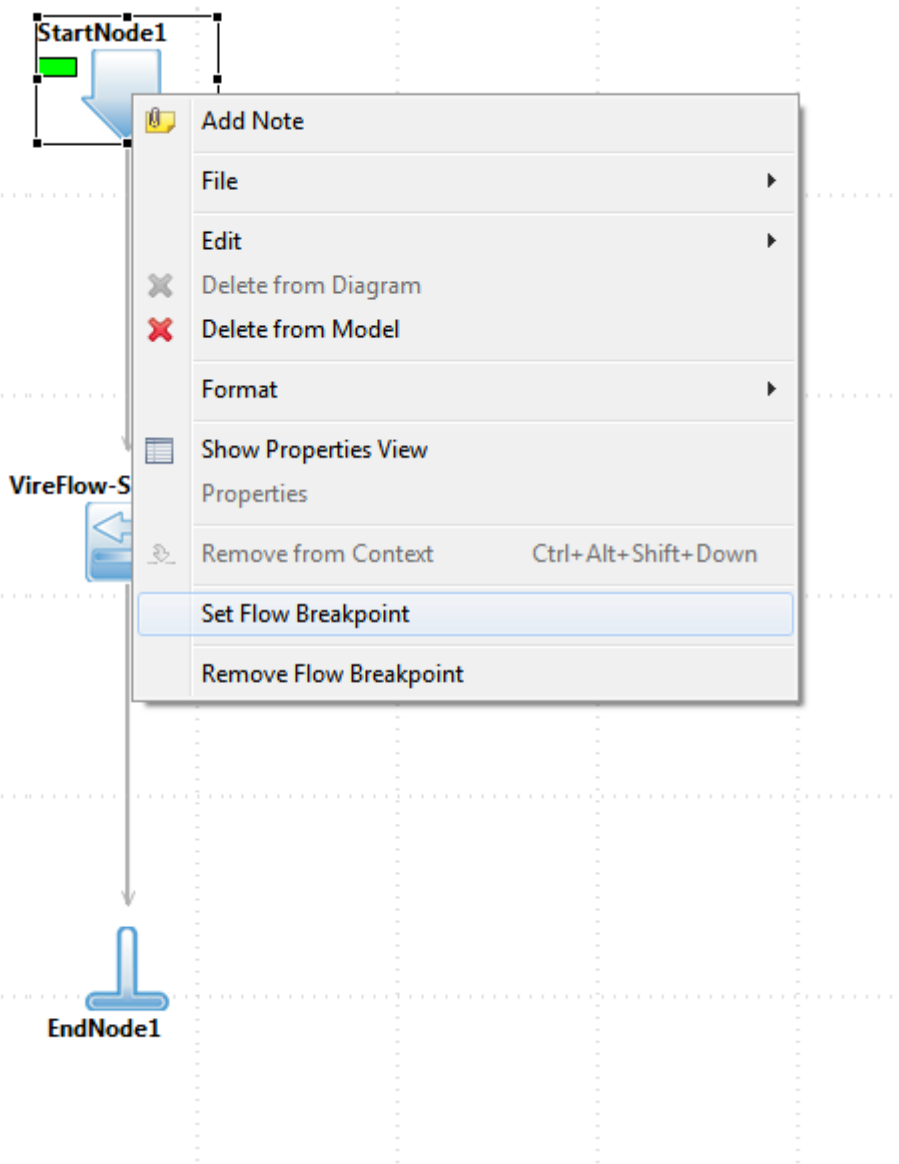
To debug your application, select a Java file with main method which will execute flow, right-click on it and select Debug As → Local Neuro4j Flows.



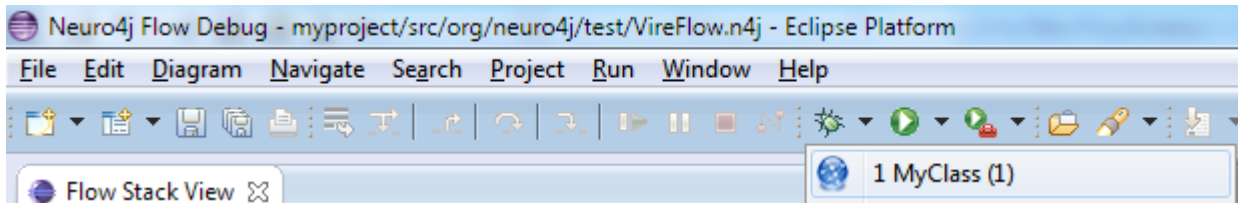
Select project and class which contains main method.



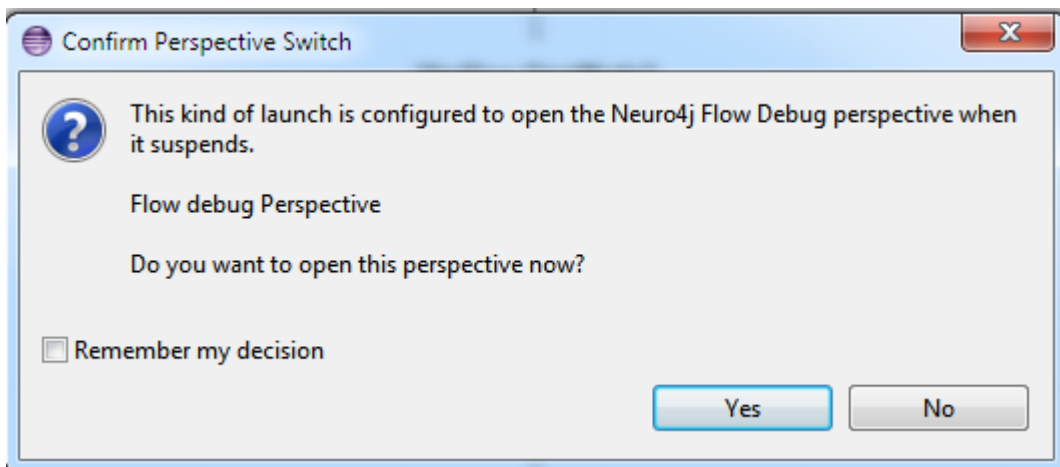
Add breakpoints:



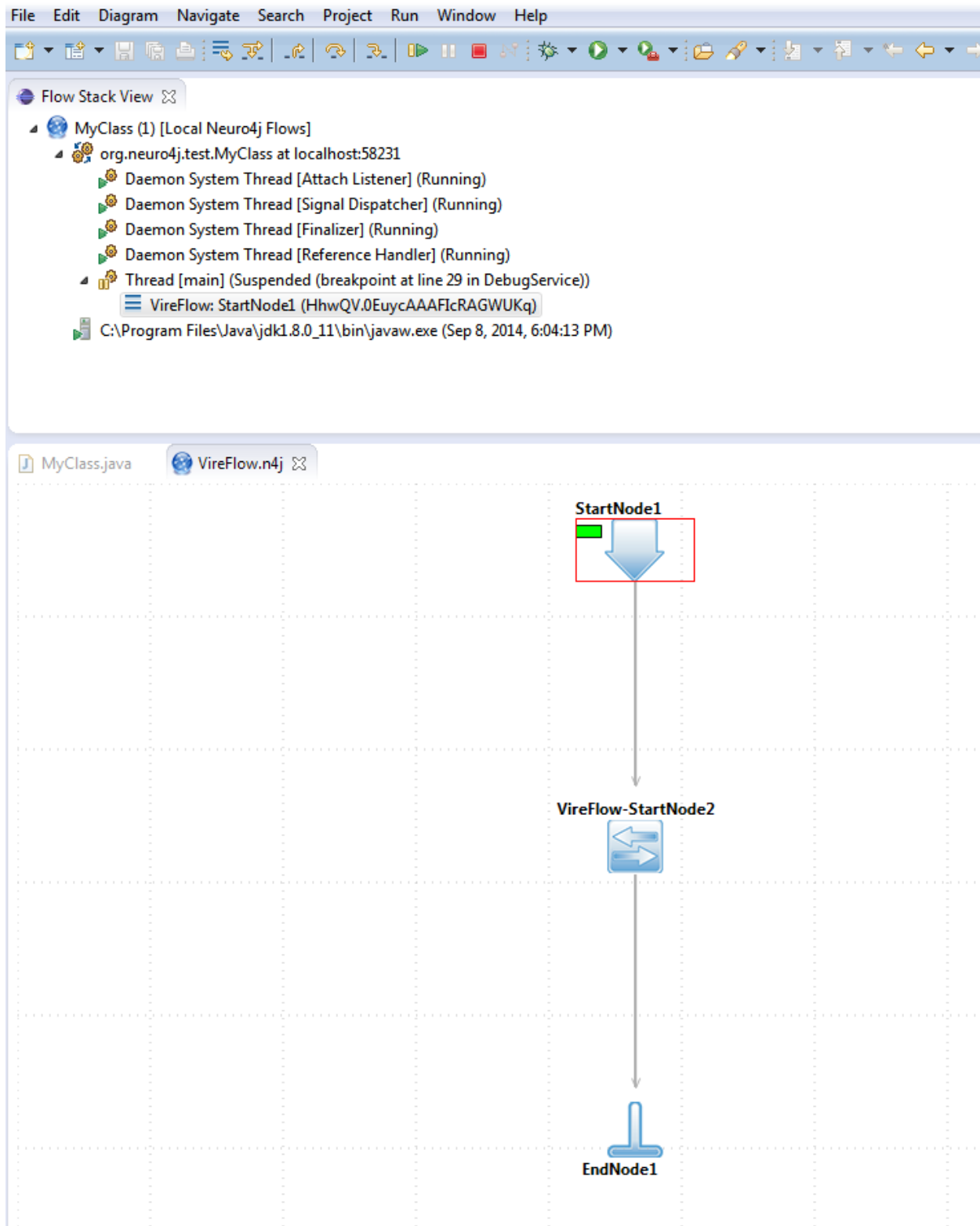
Run debug configuration:



If you start the debugger, Eclipse asks you if you want to switch to the Debug perspective once a stop point is reached. Answer Yes in the corresponding dialog.



Afterwards Eclipse opens this perspective, which looks similar to the following screenshot.



The screenshot displays the Eclipse IDE interface. At the top, the menu bar includes File, Edit, Diagram, Navigate, Search, Project, Run, Window, and Help. Below the menu bar is a toolbar with various icons for file operations, navigation, and execution. The main workspace is divided into two sections. The upper section, titled "Flow Stack View", shows a tree structure of threads and processes. The lower section, titled "MyClass.java" and "VireFlow.n4j", displays a flow diagram on a grid background. The flow diagram consists of three nodes connected by arrows: "StartNode1" (a blue arrow pointing down), "VireFlow-StartNode2" (a blue double-headed arrow), and "EndNode1" (a blue T-shaped terminal node).

Flow Stack View

- MyClass (1) [Local Neuro4j Flows]
 - org.neuro4j.test.MyClass at localhost:58231
 - Daemon System Thread [Attach Listener] (Running)
 - Daemon System Thread [Signal Dispatcher] (Running)
 - Daemon System Thread [Finalizer] (Running)
 - Daemon System Thread [Reference Handler] (Running)
 - Thread [main] (Suspended (breakpoint at line 29 in DebugService))
 - VireFlow: StartNode1 (HhwQV.0EuycAAAFicRAGWUKq)
 - C:\Program Files\Java\jdk1.8.0_11\bin\javaw.exe (Sep 8, 2014, 6:04:13 PM)

MyClass.java VireFlow.n4j

StartNode1

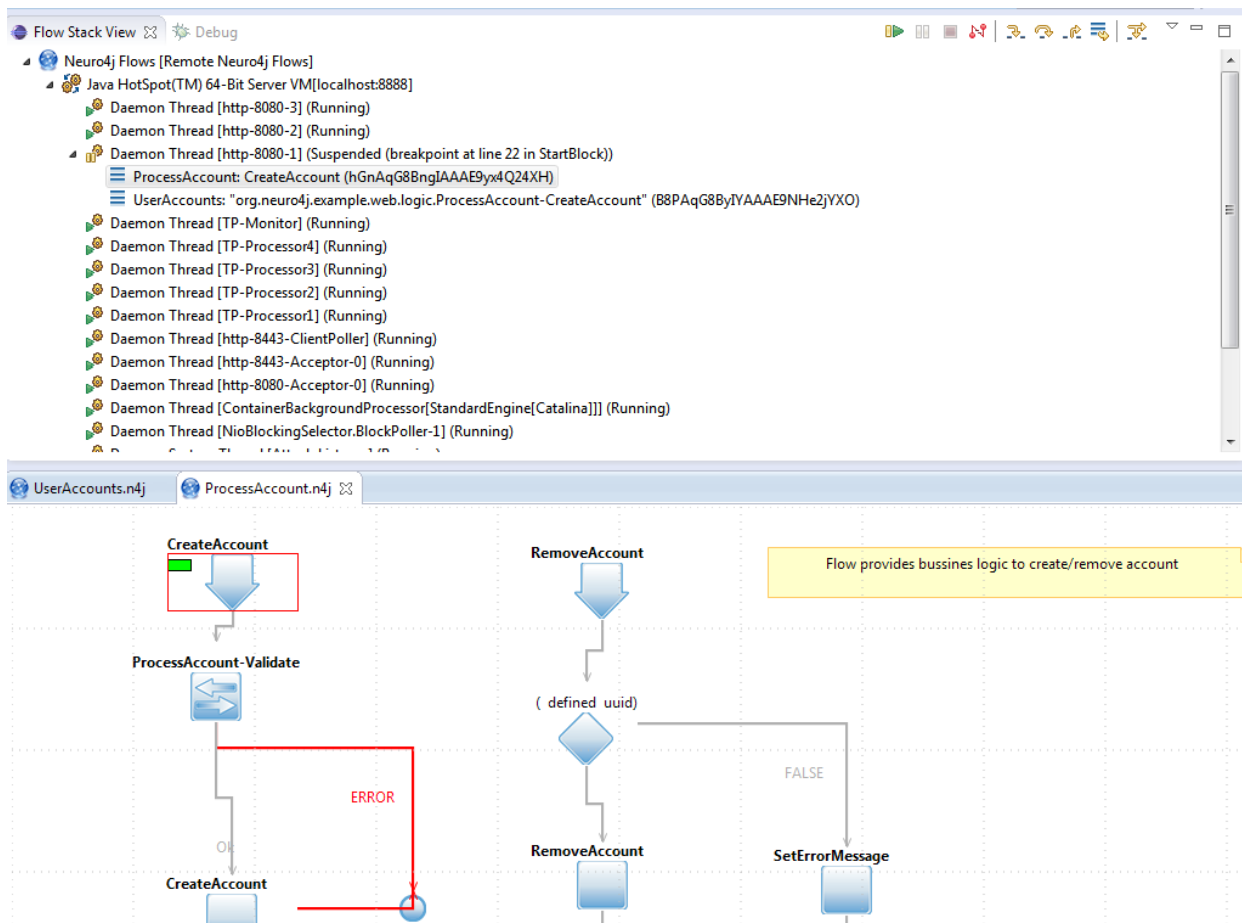
VireFlow-StartNode2

EndNode1

PERSPECTIVE OVERVIEW

Neuro4j Flow Debug perspective provides following views

FLOW STACK VIEW



This view displays Call Node's stacks. In this case flow **UserAccounts** calls **CreateAccount**.

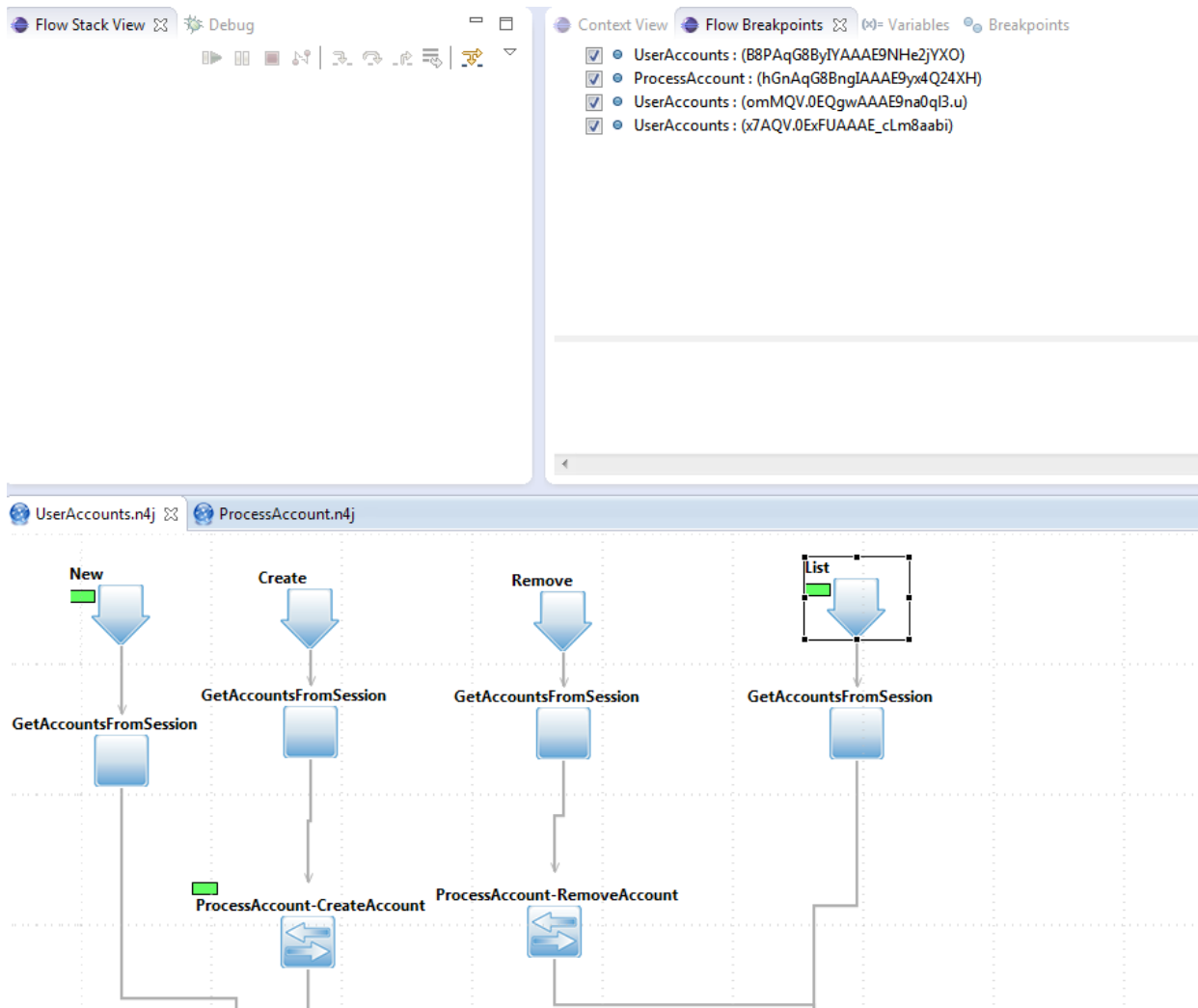
CONTEXT VIEW

Name	Value
accountList	ArrayList<E> (id=3546)
elementData	Object[5] (id=3548)
[0]	Account (id=3549)
[1]	Account (id=3550)
[2]	Account (id=3551)
[3]	Account (id=3552)
[4]	Account (id=3553)
modCount	1
size	5
RESPONSE	ResponseFacade (id=3531)
ACTION_STACK	LinkedHashSet<E> (id=3532)
CURRENT_NODE	Connected (id=3547)
REQUEST	RequestFacade (id=3534)

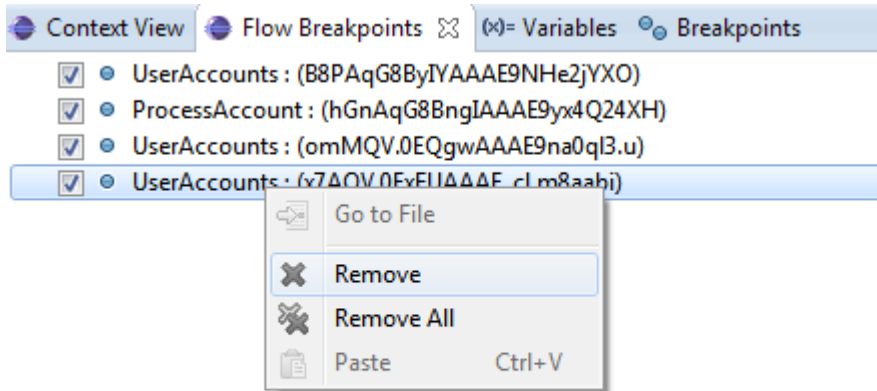
This view displays all objects in Context.

FLOW BREAKPOINTS

View displays list of all breakpoints.



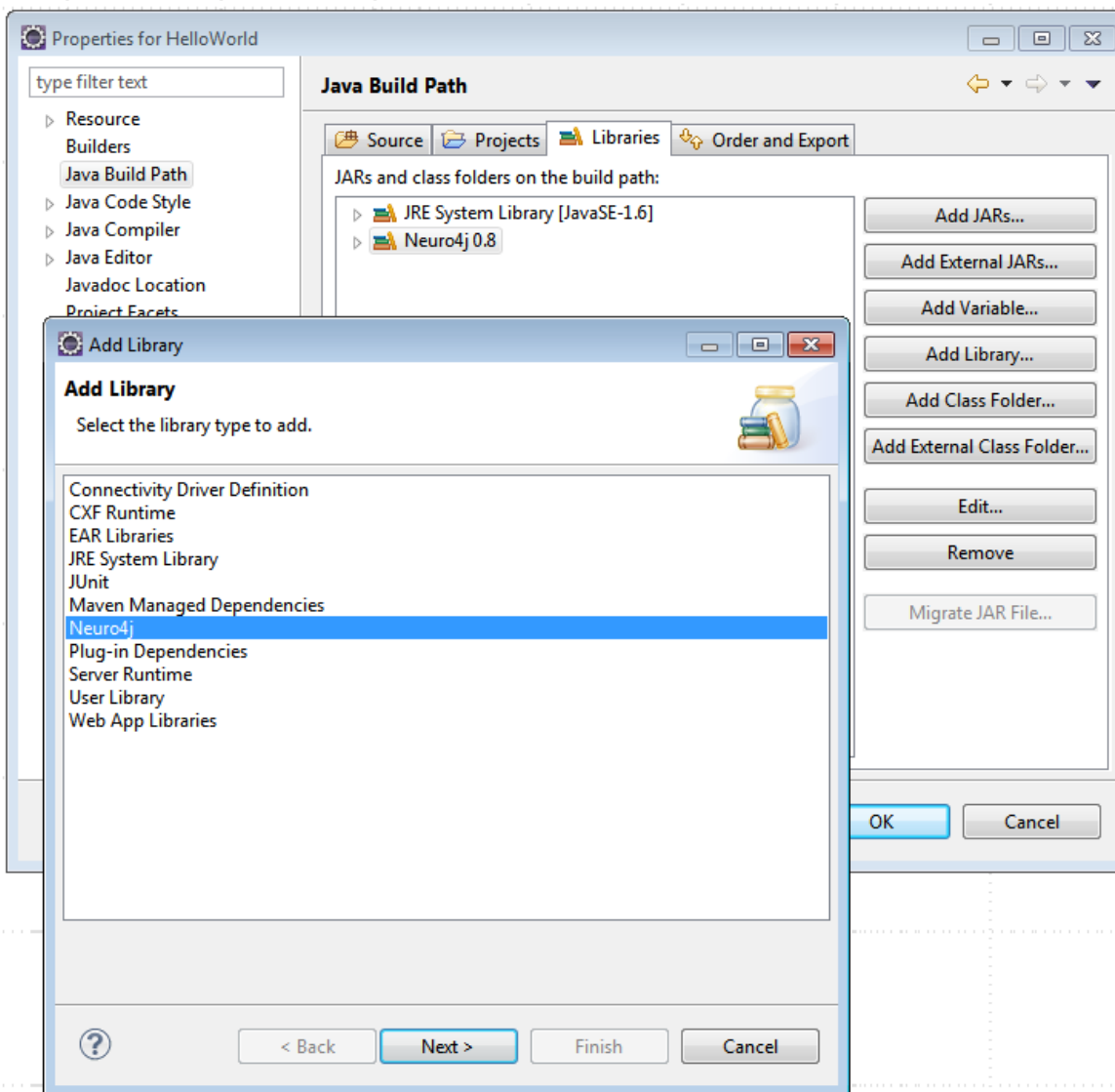
Developer can remove breakpoint using right-click.



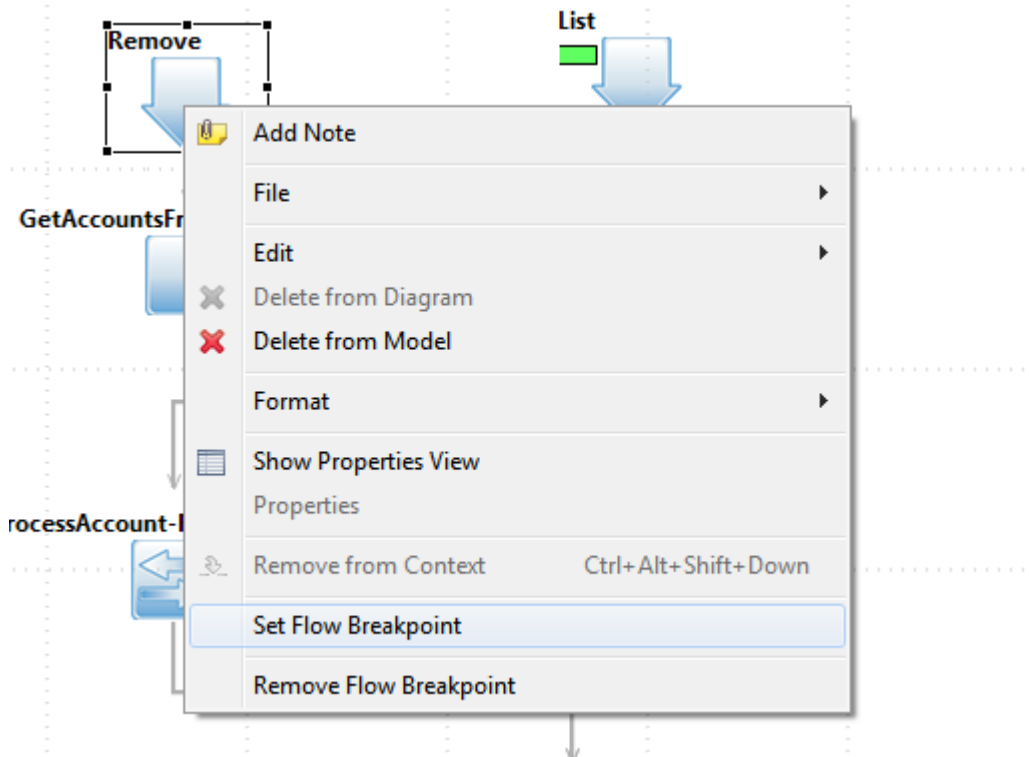
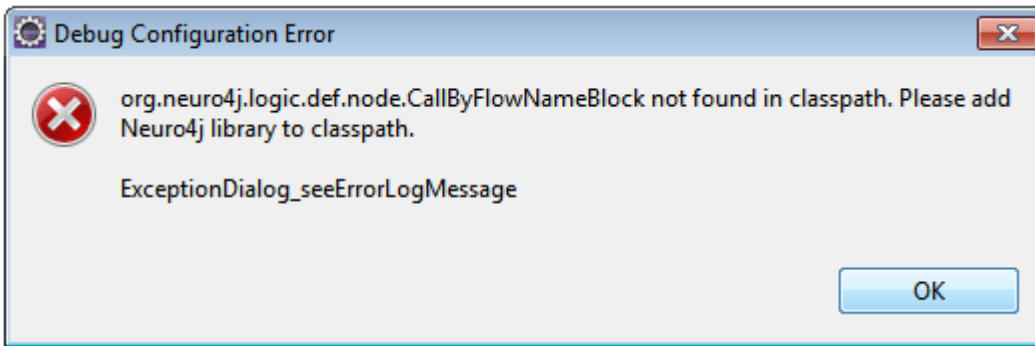
ADDING/REMOVING BREAKPOINTS

Breakpoint can be added by right-click on node in editor.

Before adding breakpoint make sure Neuro4j library had been added to classpath!

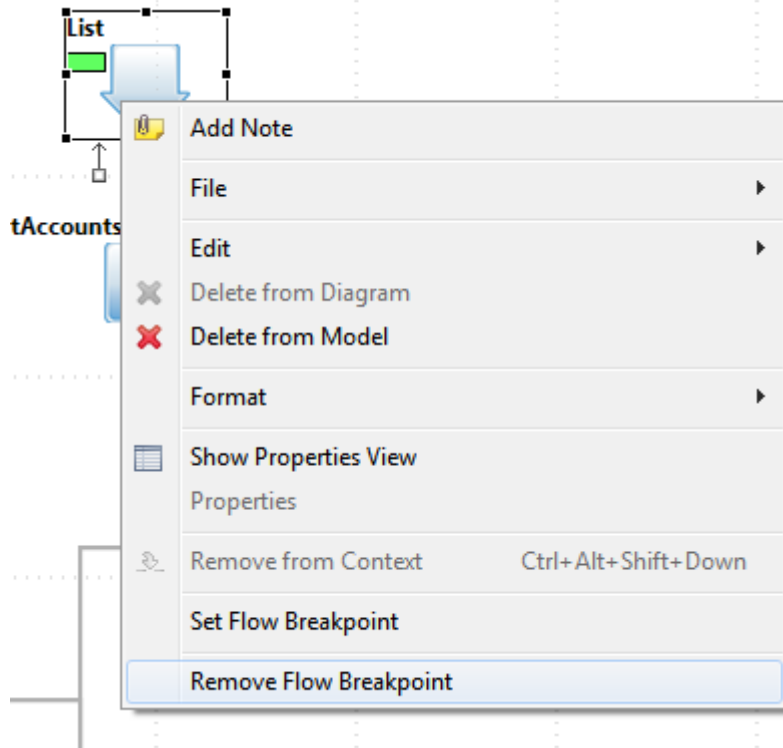


If library is not in classpath developer will receive error message like:



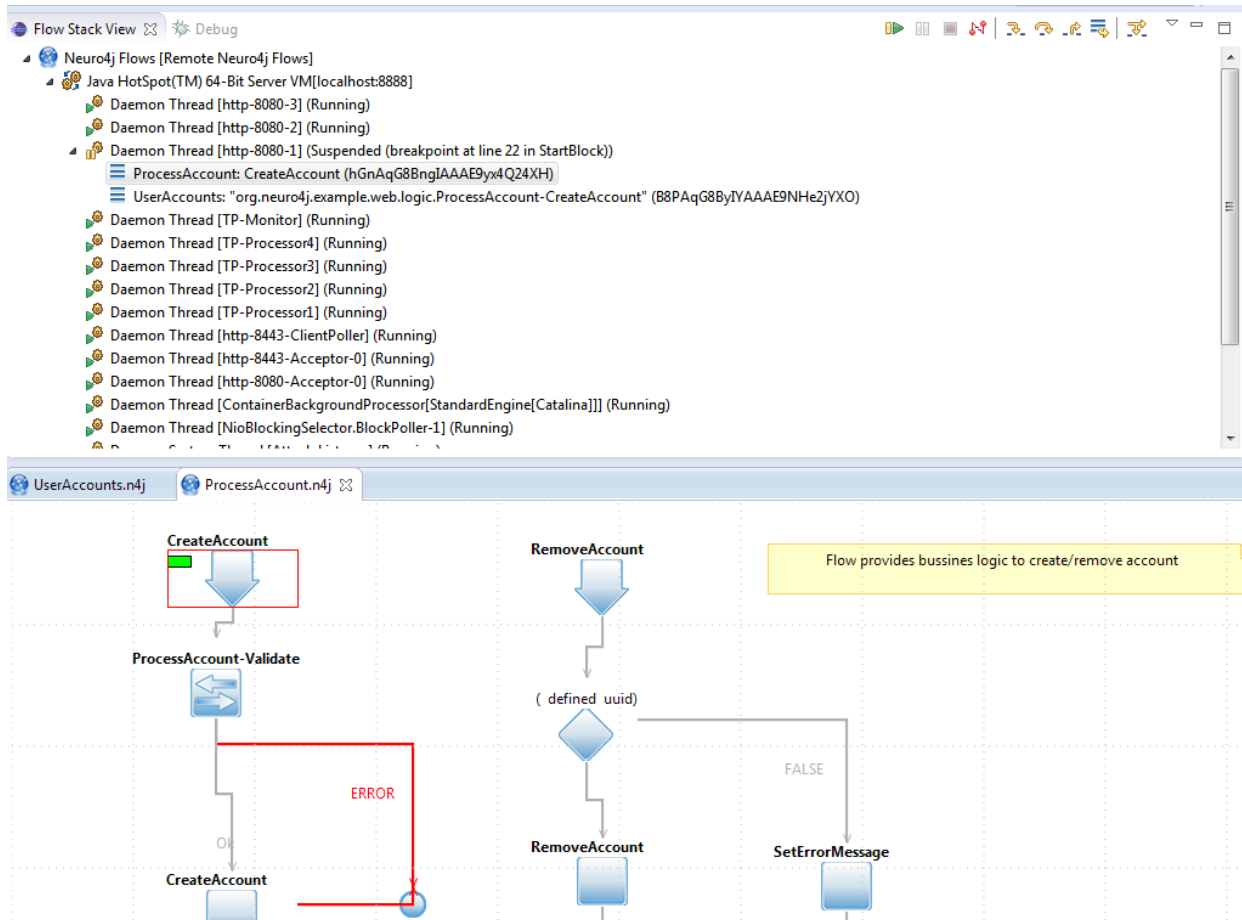
Green bar indicates what node has breakpoint.

Breakpoint can be removed by right-click -> Remove Flow Breakpoint;




REMOTE DEBUGGING

When you launch your flow application for debugging, Studio switches to the Flow Debug perspective automatically.



The screenshot shows the 'Flow Stack View' and 'Debug' perspective in Studio. The 'Flow Stack View' displays a tree of threads, with 'ProcessAccount: CreateAccount' selected. Below it, the 'Flow Diagram' shows the execution flow. A red rectangle highlights the 'CreateAccount' node, indicating where the processor was suspended. A red arrow points from the 'ProcessAccount-Validate' node to the 'CreateAccount' node, labeled 'ERROR'. A yellow callout box states 'Flow provides bussines logic to create/remove account'. The flow diagram includes nodes for 'CreateAccount', 'ProcessAccount-Validate', 'RemoveAccount', and 'SetErrorMessage', connected by arrows. A decision diamond labeled '(defined uuid)' branches into 'TRUE' and 'FALSE' paths.

Red rectangle around node indicates place where Processor was suspended.

To resume execution click  or F8 key, execution will be suspended at next breakpoint.

To make <Step Over> press F6, Processor will suspend execution at next Node.