

BEGINNER'S DOCUMENT FOR CAPE

Author : Tejaswini Ranadive
tejaswinir@wpi.edu

Editor : Venkatesh Raghavan
venky@wpi.edu

This document gives an overview of the following:

- 1) How to write a basic operator
- 2) How to integrate it in Eclipse
- 3) Details of the xml files and bat files required to run the operator
- 4) How to run the operator or the individual processes inside the console

Overview:

In order to run an operator in CAPE, we require the following components:

1. Stream Generator
2. Application
3. Query Processor
4. Distribution Manager

CAPE is a **multi-thread** application and usually follows **Round Robin Scheduling**. It means that fixed and equal time slices are assigned to each of the above processes.

For example, a Query Process will be allotted time 't', Stream Generator will be allotted time 't' and so on..

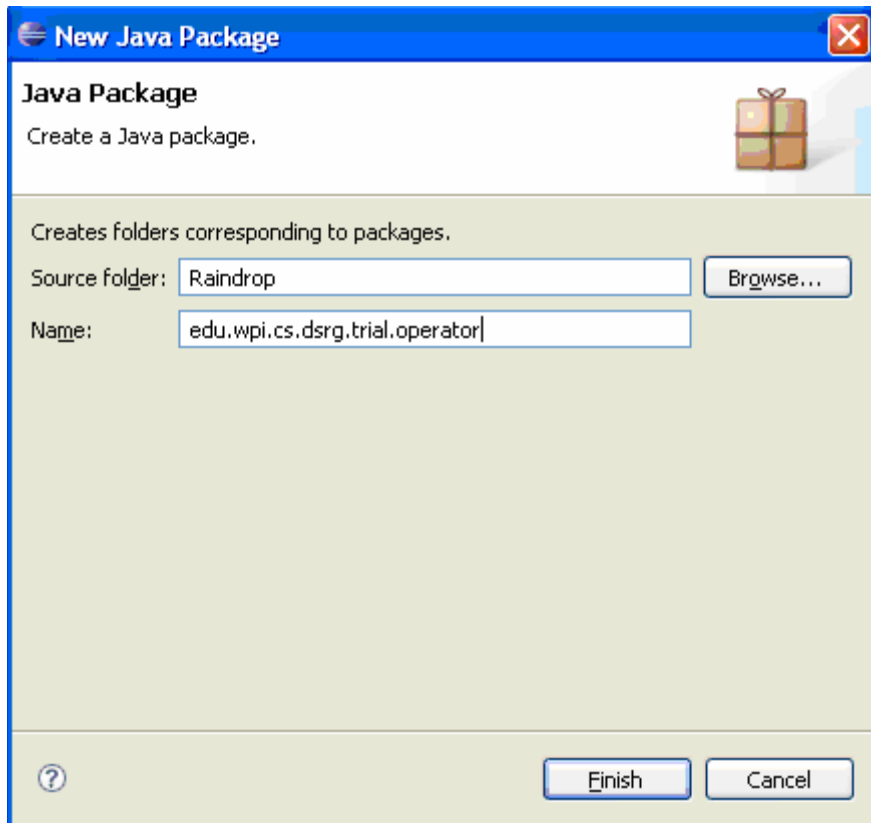
Now let us learn how to create an Operator from Scratch.

For this, we will need to

- Create a Package and add the java code
- Create all the required xml files and bat files

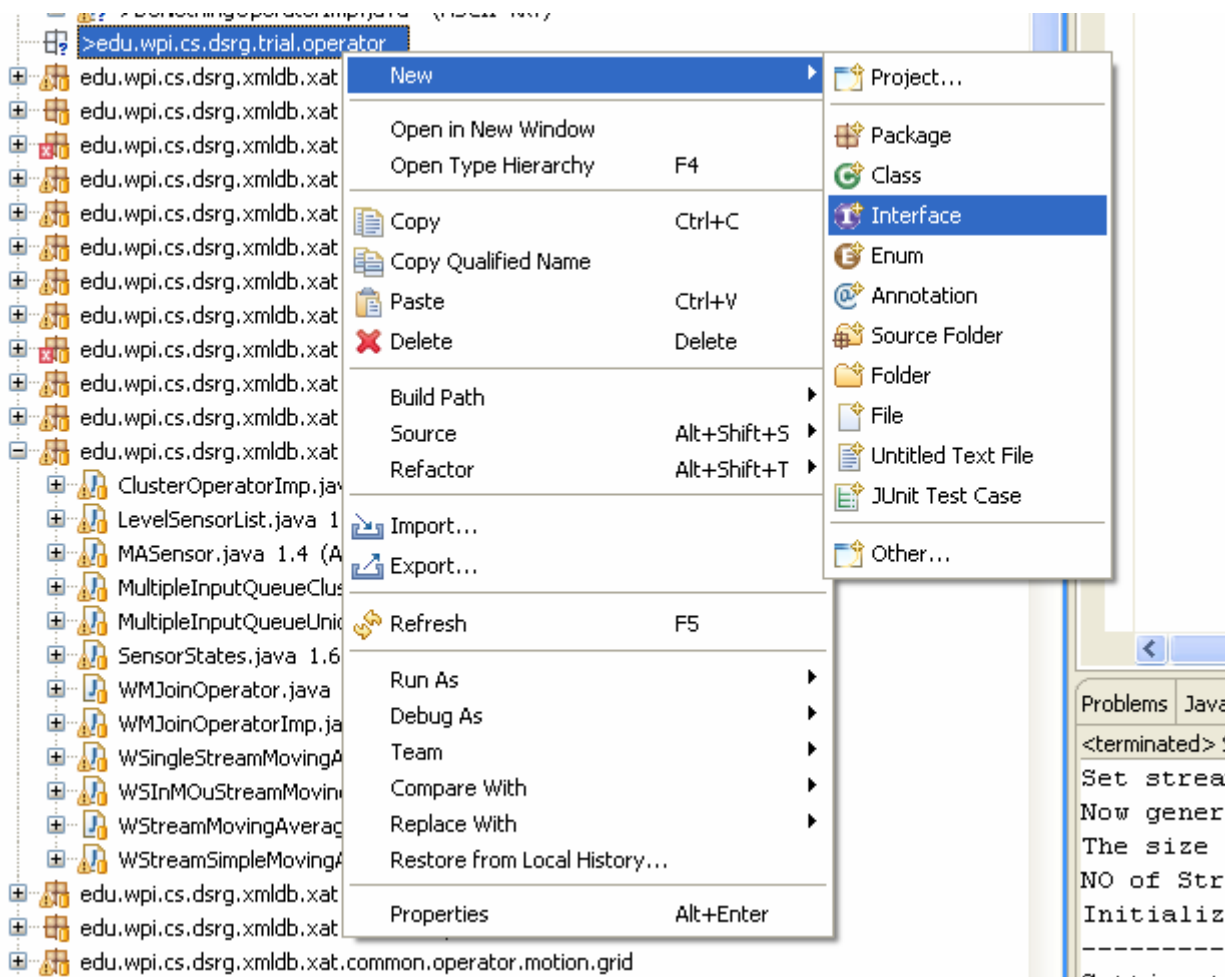
CREATING A PACKAGE:

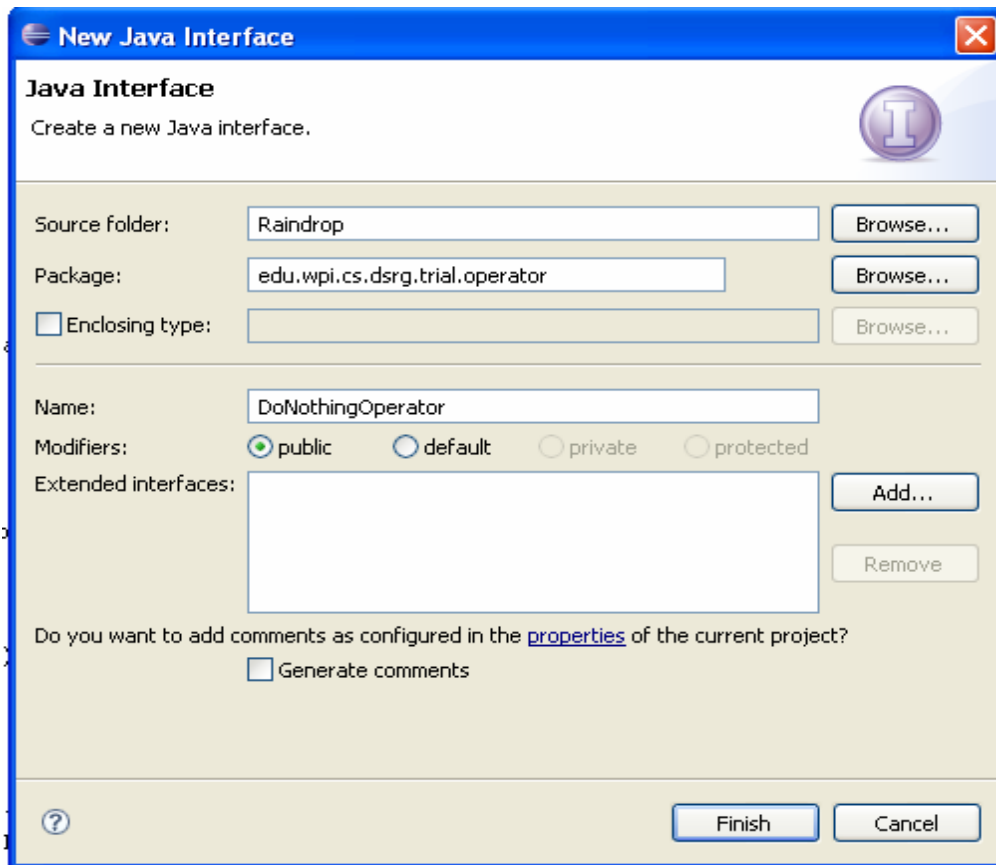
Go to Raindrop >> New >> Package



Now that we have created a package, we need to add a class and interface file to it, which will contain the code.

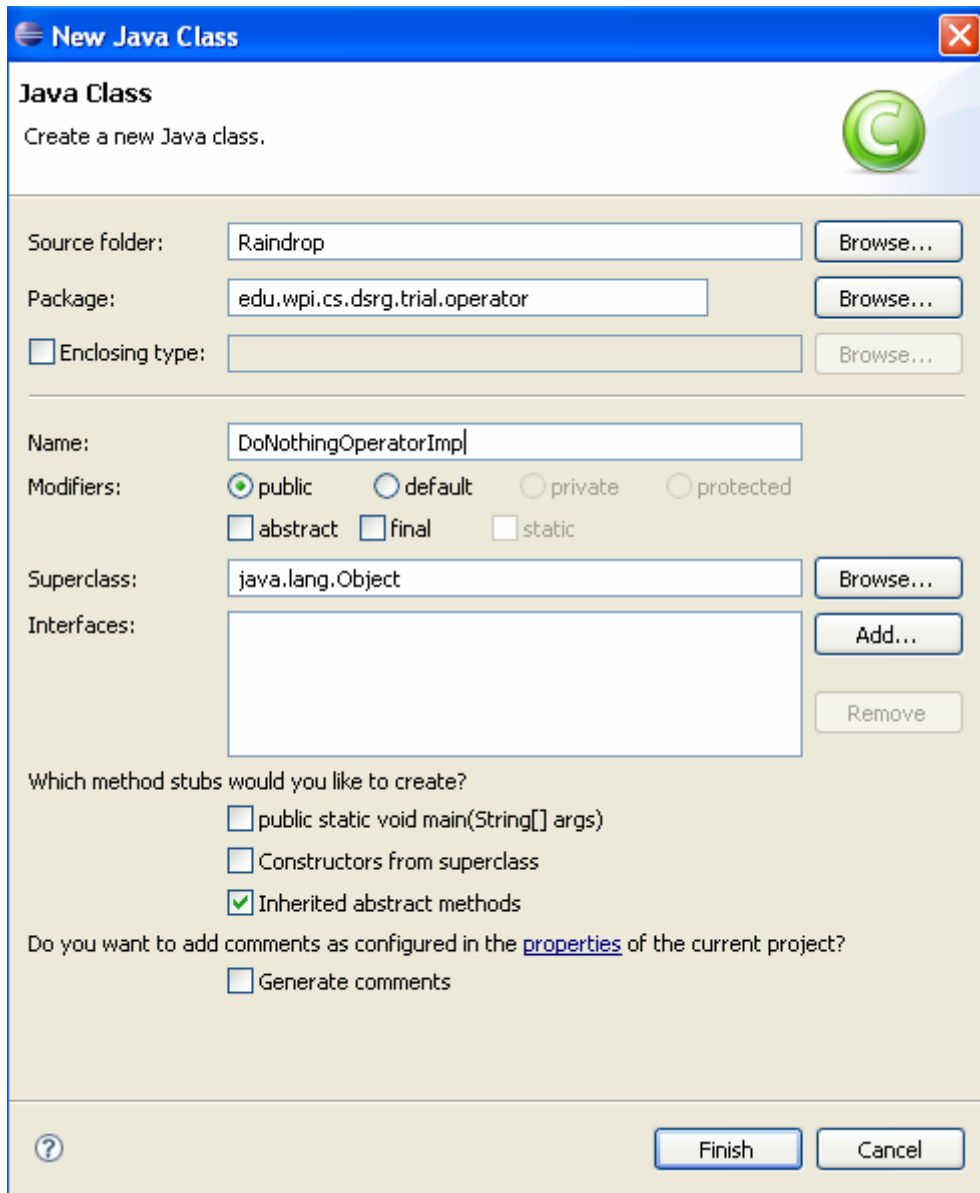
This is how we add an INTERFACE:





In this file, we write the interface, eg : operatorName.java

In a similar way, we create a class file as follows:



This file should contain the class file like `operatorNameImp.java`, which implements the operator.

Now let us understand the basic idea of the code.

The code that you need to execute in your operator needs to be in the **run()** method. Before **run()**, the **init()** method is called, which sets up the Output Queue Schema.

The run method first DEQUEUES the tuples from the input queue. You can send them one by one or in bulk to another method where your functionality is coded. Finally, the output tuples should be sent to the Output Queue.

Now we are done with adding the java code.

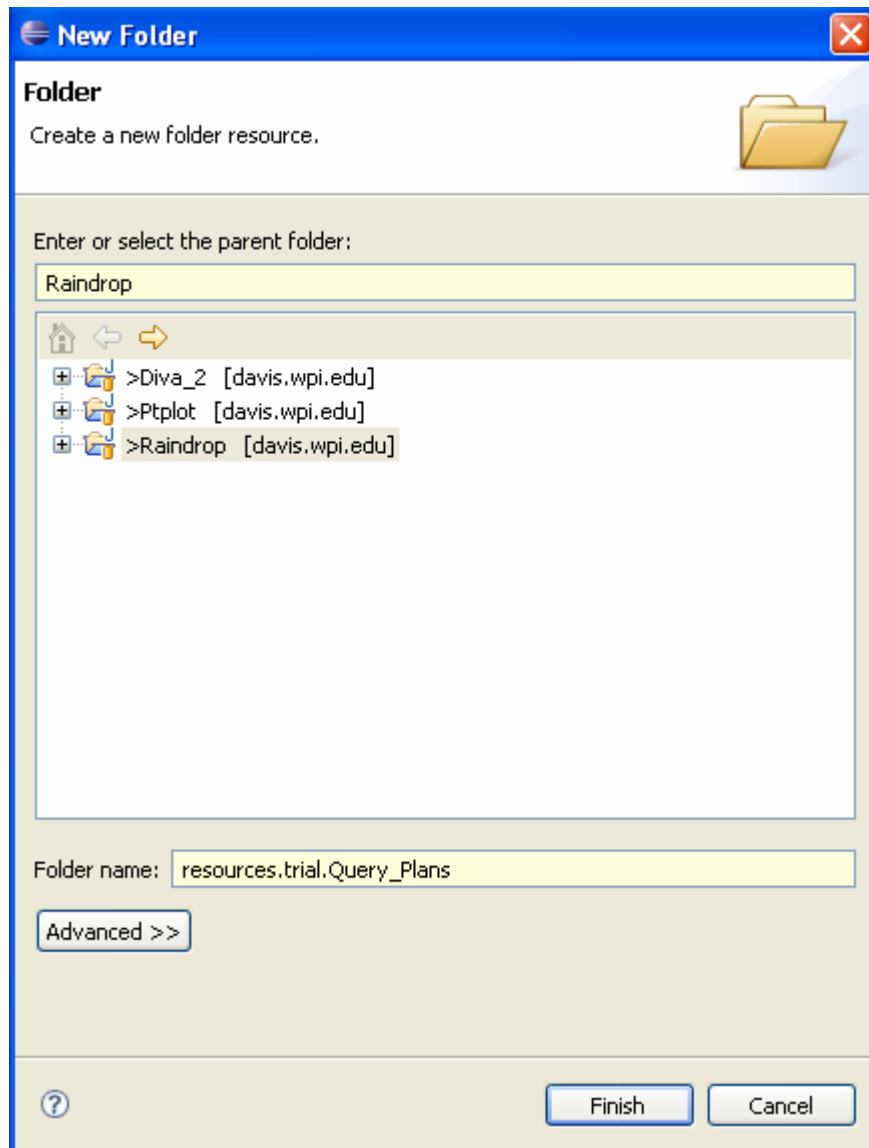
CREATING XML AND BAT FILES:

Next we need to add all the xml files and bat files required.

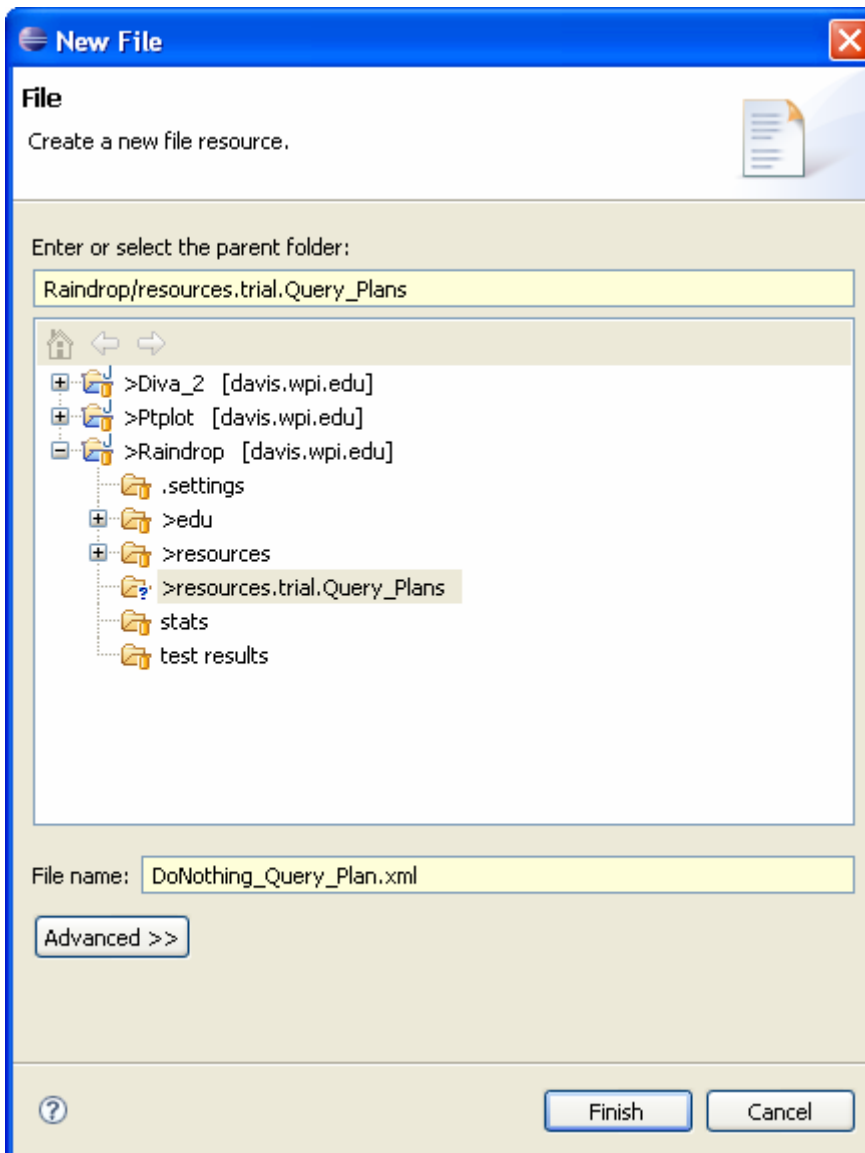
So we create separate folders for Query Plans, Stream Files, System Files and bat files as follows.

The following is a snapshot for creating the folder resources.trial.Query_Plans and a file DoNothing_Query_Plan.xml

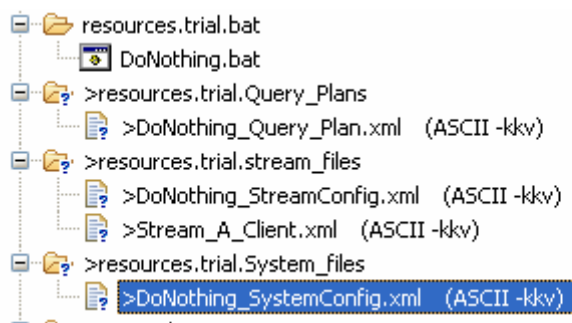
Go to Raindrop >> New >> Folder



In this folder, create a file as follows:



Similarly, create folders and files as follows:



Now we are done with creating all the required files and folders.

We aim to read the input queue and put it in the output queue without doing anything to it. So in this case, we have only 1 child, i.e. the input data stream. Also there is only 1 input queue and 1 output queue.

Now let us look at all the required xml files in detail

❖ QUERY PLAN

Query Plan represents the physical execution of a database query, and is generally depicted using a *Direct Acyclic Graph (DAG)*.

Let us have a look at a sample Query Plan.

```
<queryplan>
  <operator root="true" id="0"
className="edu.wpi.cs.dsrg.tejaswini.operator.DoNothingOperatorImp"
numberOfOutputQueue = "1">
  <classVariables/>
  - <properties/>
  <schema />
  <parents/>
  <children>
    <child type="stream" name="A" />
  </children>
</operator>
</queryplan>
```

We need to specify the className, the input queue name, number of output queues and so on.

❖ STREAM FILES

Next what we need is the Stream files

There are 2 stream file:

1) StreamConfig.xml

It specifies the name, schema and format of the data file.

Eg:

```
<streams>
  <!--Note the stream name has to be unique for each stream-->
  <stream name="A">
    <files>
      <file name="resources\firestream\data\txt\testdataA.txt"/>
    </files>
  </stream>
</streams>
```

```

        <repeat_file value="true"/>
    </files>
    <!--Gives the format of the file-->
    <delimiter attribute=" " record="\n"/>
    <schema>
        <table name="A"/>
        <attribute name="a1" type="int"/>
    </schema>
    <inter_arrival>
        <distribution value="poisson" seed="0">
            <interval start_time="0" mean="1000"/>
        </distribution>
    </inter_arrival>
</stream>
<total_time value="-1"/>
</streams>

```

The stream is 'A', the input file is **testdataA.txt**. Rest all details, we can just follow without explanation for the time being.

2) Stream_Client.xml

It provides the following information:

For each server, the ip address and the port number
 For each stream, which server it is coming from

Eg:

```

<client_config>
    <servers>
        <server name="Stream1" ip_address="localhost" port="15000"/>
    </servers>
    <streams>
        <stream name="A" server="Stream1"/>
    </streams>
</client_config>

```

❖ SYSTEM CONFIG FILE

Now comes the System Config file

It gives a general idea about what kind of scheduling to use (eg Round Robin), Location of stream config file, query plan xml file, output file name, format, and many such details.

❖ BATCH FILE

Batch file initiates all the processes, i.e. the Stream Generator, Application, Query Processor, and the Distribution Manager.

Eg:

```
cd C:\Documents and Settings\tejaswinir\workspace\Raindrop
```

```
START /min "Stream Generator" java -classpath
.;xercesImpl.jar;xmlParserAPIs.jar;xercesSamples.jar
edu.wpi.cs.dsrq.xmlldb.xat.component.streamgenerators.server.XATStreamGenerator 15000
resources\tejaswini\stream_files\doNothing_streamConfig.xml
```

```
START /min "Application" java -classpath
.;..\Diva_2;xercesImpl.jar;xmlParserAPIs.jar;xercesSamples.jar
edu.wpi.cs.dsrq.xmlldb.xat.component.application.RaindropApplication 16001 -o f
```

```
START "Query Processor" java -classpath .;..\Ptplot;..\Diva_2;jxl.jar;jakarta-regexp-
1.3.jar;xercesImpl.jar;xmlParserAPIs.jar;xercesSamples.jar
edu.wpi.cs.dsrq.xmlldb.xat.component.executioncontroller.DistributedExperimentSetup
8001
```

```
START "QM Test Standard Run" java -classpath .;..\Ptplot;..\Diva_2;jxl.jar;jakarta-
regexp-1.3.jar;xercesImpl.jar;xmlParserAPIs.jar;xercesSamples.jar
edu.wpi.cs.dsrq.xmlldb.xat.component.executioncontroller.DistributedExperimentSetup
resources\tejaswini\system_files\doNothing_SystemConfig.xml
```

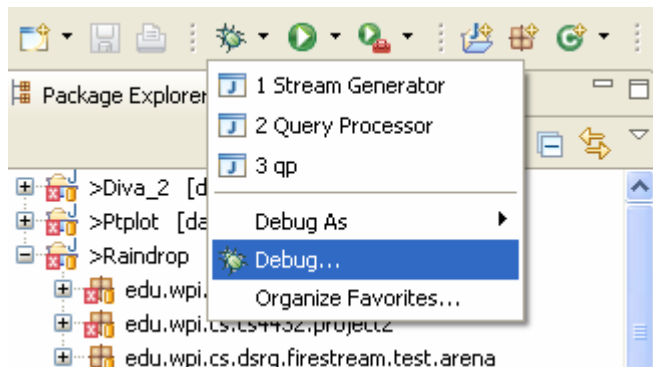
Pause

Now, if we do not want to run the whole application in a single go, we can check if each process is working separately.

RUNNING THE OPERATOR INSIDE THE CONSOLE:

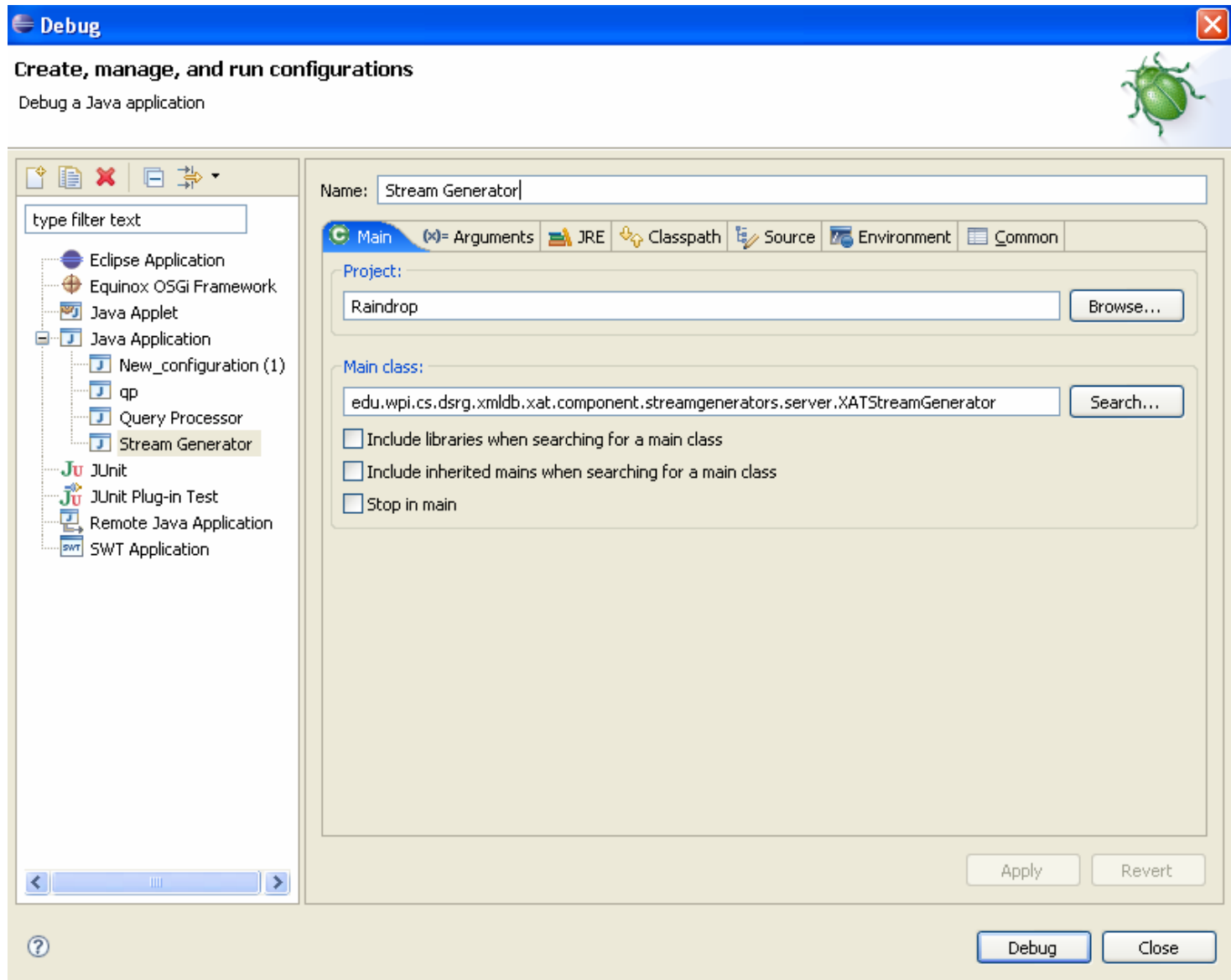
Now we will try to run and debug each thread separately inside the console.

1) **STREAM GENERATOR:**



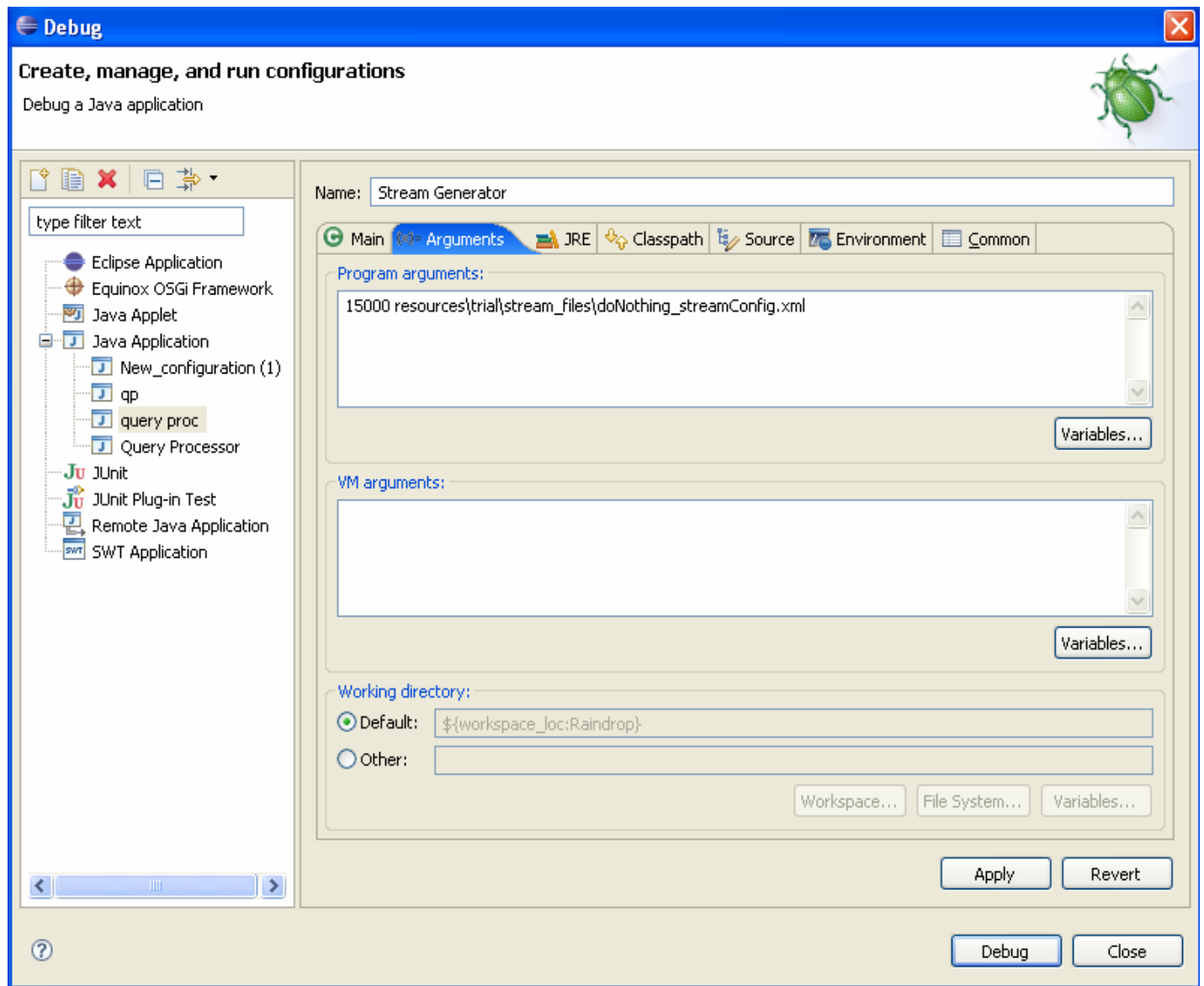
Then the window shown below appears.

Name the Project as Raindrop and the Main Class as shown below.

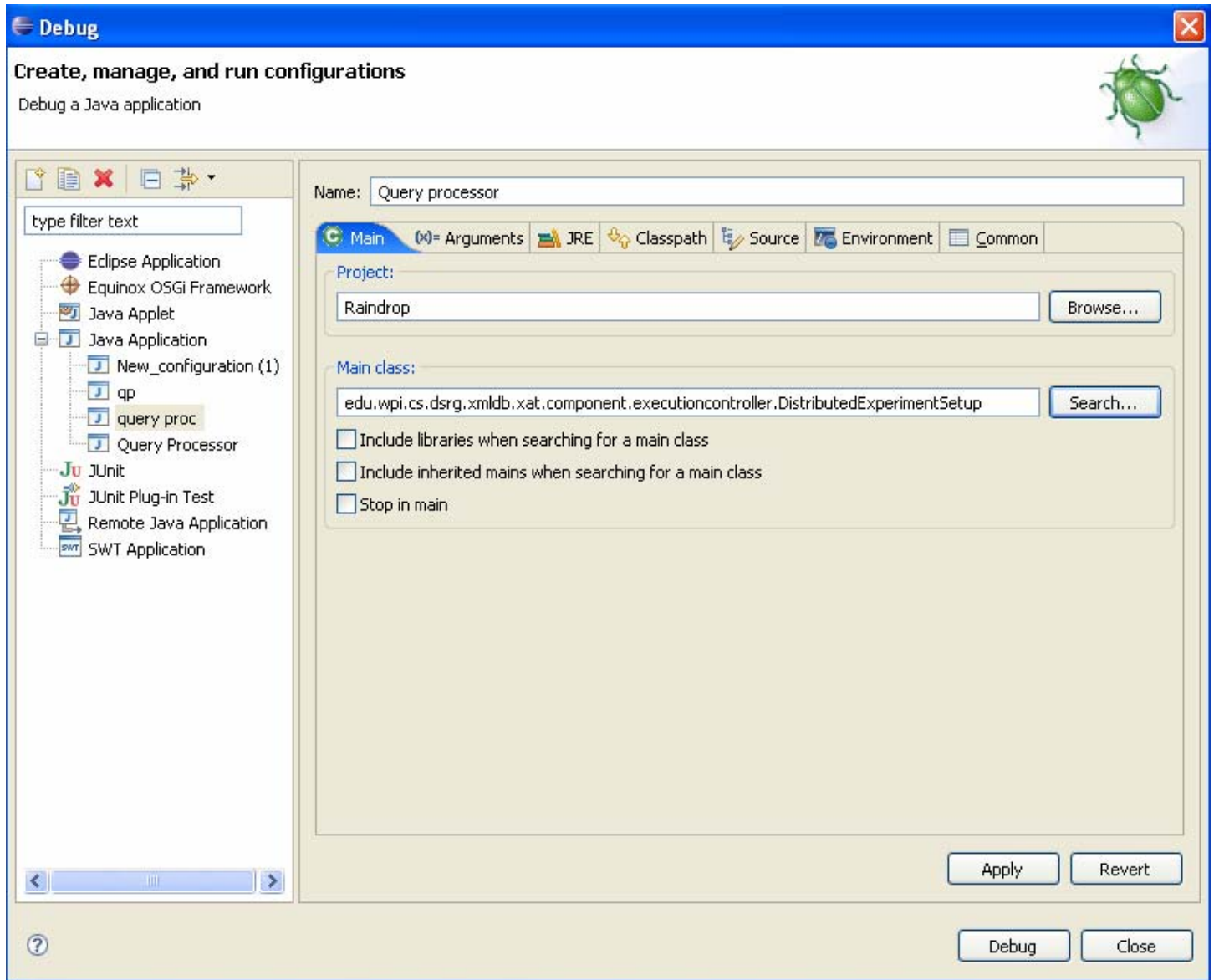


Next click on the Arguments tab and enter the variable as follows, which is specified in the bat file.

The following is the snapshot of the same.



Similarly we can debug for Query Processor as well in the following way



Using this we can check if the individual processes are running successfully, before running the whole application.

After you are assured that the individual processes are running fine, your operator is ready to run.