[Overview 2](#_Toc381111680)

[What is π 2](#_Toc381111681)

[Monte Carlo method 2](#_Toc381111682)

[Pre-requisites 3](#_Toc381111683)

[Configuration 3](#_Toc381111684)

[Design 3](#_Toc381111685)

[DAG 4](#_Toc381111686)

[Operators 4](#_Toc381111687)

[RandomEventGenerator : 4](#_Toc381111688)

[PiCalculateOperator: 4](#_Toc381111689)

[Console: 4](#_Toc381111690)

[Functionality 5](#_Toc381111691)

[Launching the application 5](#_Toc381111692)

[Monitoring the application 6](#_Toc381111693)

[Operations 11](#_Toc381111694)

[Stopping the application 14](#_Toc381111695)

[Conclusions 16](#_Toc381111696)

Application ‘Pi Calculator’

# Overview

This ‘π’ calculator is one of the simplest applications, to understand the DataTorrent platform. This application computes value of ‘π’ using – ‘Monte Carlo π estimation formula’.

## What is π

‘π’ is a numerical constant that represents the ratio of a circle's circumference to its diameter. It is an irrational number i.e. it has an infinite number of non-repeating decimal digits after the decimal point.

Another way to define ‘π’ is – it is the area of a unit circle. To understand this we need to know that the area of a circle is (π \* r2) and that a unit circle is simply a circle whose radius is equal to 1.

Consider a circle of radius ‘*r*’ inscribed within a square with a side *2r*. The diameter of the circle is equal to the width of the square. The square, hence, has length and width equal to the diameter of the circle (or twice the radius, *2r*).

Area of the circle:



Area of the surrounding square:



If we compare the area of the circle with the area of the square, we can form a ratio:



Rearranging to solve for pi:



In other words, pi is equal to the ratio of the area of an inscribed circle to the area of its outer square, multiplied by four. Our estimate for pi will use the equation above.

## Monte Carlo method

A Monte Carlo method is a mathematical model that relies on chance or repeated random behavior in order to determine a solution to a problem.

Monte Carlo methods can be thought of as statistical simulation methods that utilize sequences of random numbers to perform the simulation.

The name "Monte Carlo'' was coined by [Nicholas Constantine Metropolis](http://scienceworld.wolfram.com/biography/Metropolis.html) (1915-1999) and inspired by [Stanslaw Ulam](http://www-groups.dcs.st-and.ac.uk/~history/Mathematicians/Ulam.html) (1909-1986), because of the similarity of statistical simulation to games of chance, and because Monte Carlo is a center for gambling and games of chance.

In a typical process one computes the number of points in a circle C that lies inside square S.  The ratio of the number of points that fall inside C to the total number of points tried is an estimation of the ratio of the two areas. The quality of the estimate depends on the number of points used, with more points leading to a more accurate value.

* C is the circle with area Ac
* S is a square with area As
* π = 4 \* (Ac/As)

A simple Monte Carlo simulation to approximate the value of ‘π’ involves randomly selecting points (x, y) in the unit square and determining the ratio m/n, where

* ‘m’ is number of points that satisfy xi2 + yi2 <= 1,
* ‘n’ is the total number of points and
* 1 is the base.

The accuracy tends to increase with the size of sample. The advantage of Monte Carlo simulation is that we do not need any analytic study of the object. This is why Monte Carlo is used in many applied areas.

# Pre-requisites

There are no special pre-requisites needed to run the ‘Pi’ demo application on the DataTorrent platform.

# Configuration

There is no particular configuration that needs to be done to run the ‘Pi’ demo application on the DataTorrent platform.

# Algorithm

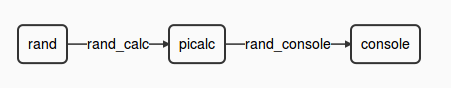
The ‘Pi’ application written for DataTorrent platform uses this algorithm:

1. Generates random distribution of points.
2. Checks whether the point falls inside the circle or not.
3. Counts the number of points that are being plotted.
4. Gets the ratio of the number of points in the circle to the number of point that fall in the square.
5. Calculates the instantaneous value of ‘π’ after each such point.
6. It then proceeds to calculate the average value of ‘π’
7. The application outputs the final value of ‘π’ on the console.

# Design

The DAG for the application is shown below followed by a description of the operators.

## DAG



This view can be seen in the ‘logicalDAG’ widget in the console.

## Operators

The ‘Pi’ application involves multiple computational steps and the DataTorrent platform provides operators to perform each of these. Specifically, this application consists of the following operators –

|  |  |  |
| --- | --- | --- |
| **Type of Operator** | **Name of the operator in DAG** | **Library** |
| [RandomEventGenerator](#RandomEventGenerator) | Rand | lib/testbench/randomeventgenerator |
| [PiCalculateOperator](#PiCalculateOperator) | Calc | custom |
| [ConsoleOutputOperator](#ConsoleOutputOperator) | Console | lib/io/consoleoutputoperator |

### [RandomEventGenerator](https://github.com/DataTorrent/Malhar/library/src/main/java/com/datatorrent/lib/testbench/RandomEventGenerator.java) :

This is the first operator in the DAG and performs the role of simulating an input received by the DAG.

The name of the instance of this operator is ‘rand. It generates a random input between a range of 0 and 30,000

### [PiCalculateOperator](https://github.com/DataTorrent/Malhar/demos/src/main/java/com/datatorrent/demos/pi/PiCalculateOperator.java):

This operator calculates value of ‘π’ using the Monte Carlo technique. It receives the input from the RandomEventGenerator ‘rand’ and computed the areas ‘Ac’ and ‘As’ as a part of processing this input data. Once computed, it takes the ratio of the two computed areas – (Ac/As) and multiplies it by a constant ‘4’ as per the Monte Carlo technique.

The value of ‘π’ thus computed is emitted on its output port ‘output’.

### [Console](https://github.com/DataTorrent/Malhar/blob/master/library/src/main/java/com/datatorrent/lib/io/ConsoleOutputOperator.java):

The console operator displays the value of ‘π’ received on its input port.

#### Streams

|  |  |
| --- | --- |
| **Stream** | **Description** |
| Rand\_calc | Sends the generated data to the PiCalculatorOperator as input |
| Rand\_console | Sends the computed value of ‘π’ from the PiCalculatorOperator’s output port to the console operator’s input port. |

# Functionality

The detailed functionality of each operator is covered above. Note that in this application, we do use operators which are stateful as well as some which are stateless.

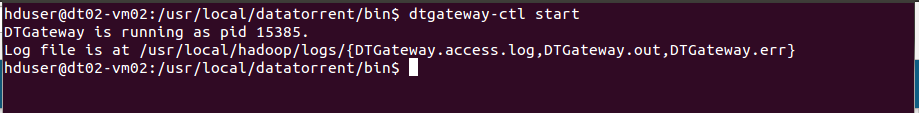
|  |  |  |
| --- | --- | --- |
| **Type of Operator** | **Statefull** | **Partitionable** |
| RandomEventGenerator | - | Y |
| PiCalculatorOperator | Y | Y |
| Console (Please refer the note below) | - | Y |

Note: The Console operator is partitionable and can be partitioned based on a key. In the current demo application, it is not being partitioned.

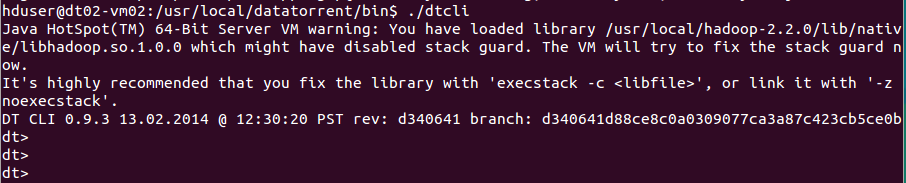
# Launching the application

In order to be able to launch sample Demo Applications, please make sure you have completed following steps,

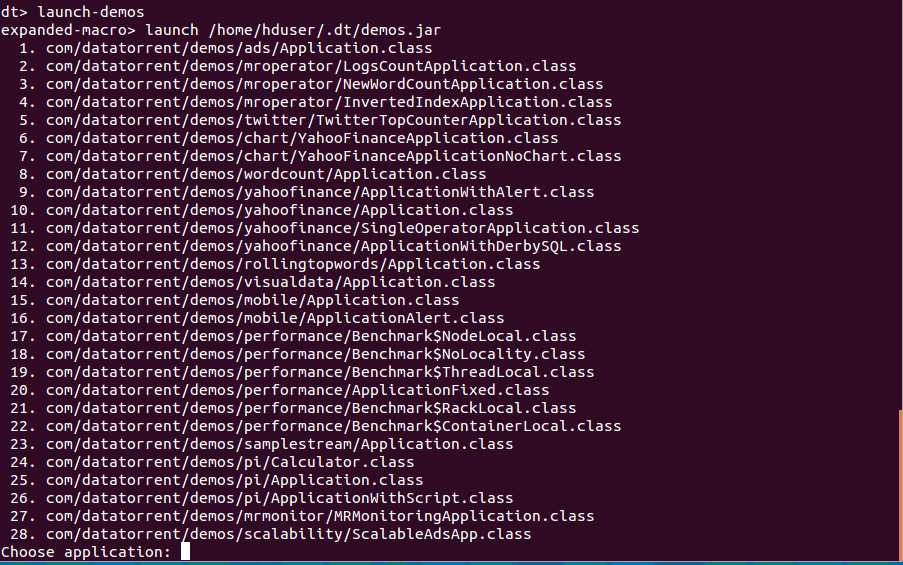
1. In case, the user has not started the ‘dtGateway’, start the DataTorrent Gateway process. This can be done from the command line as –



1. Launch DataTorrent Command Line Interface (dtcli)



1. Launch demo application Jar. (Here, we assume that the Hadoop services are up and running and the demos are being launched in a cluster mode.)



The user can now select and launch the application.

The jar is in ~/.dt/demos.jar and can also be launched as "launch ~/.dt/demos.jar

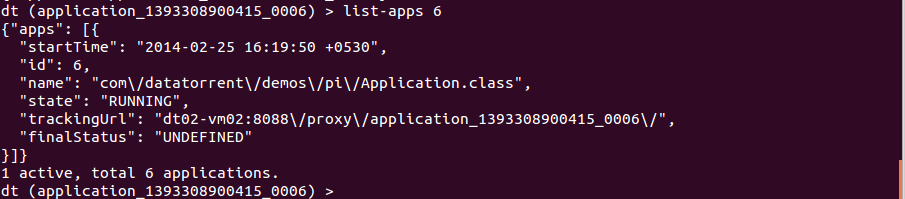
Please refer to DataTorrent Quick Start Guide for additional details.

# Monitoring the application

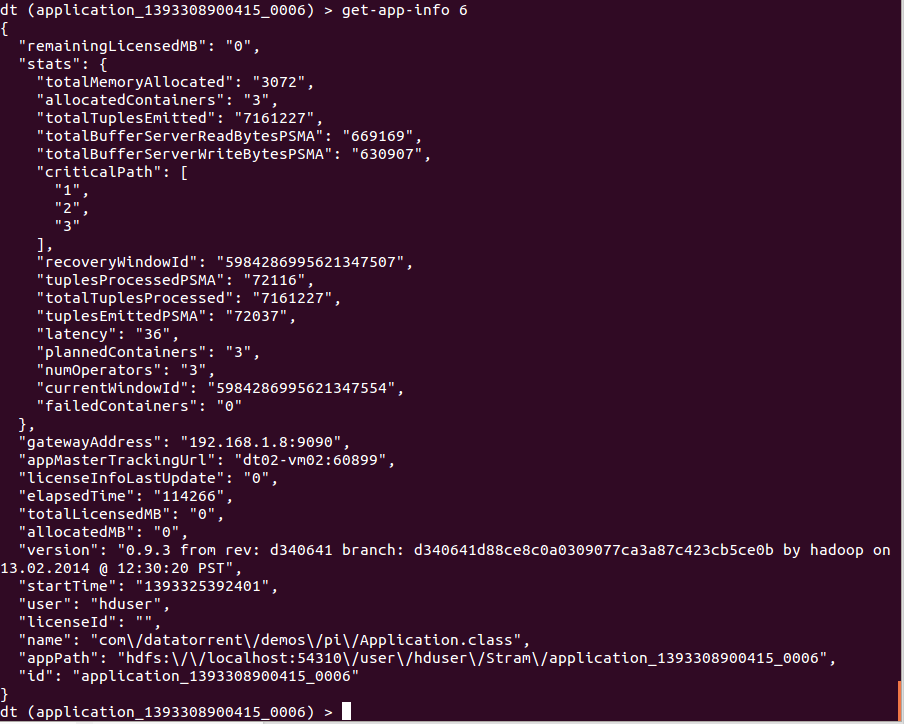
Once launched, the application can be monitored/managed from either the command line viz. ‘dtcli’ or the GUI viz. DataTorrent Gateway Console. The console and cli get data from the same webservices layer. Following are some of the sample operations that are typically performed while monitoring the application –

1. List the application -

#### Using ‘dtcli’



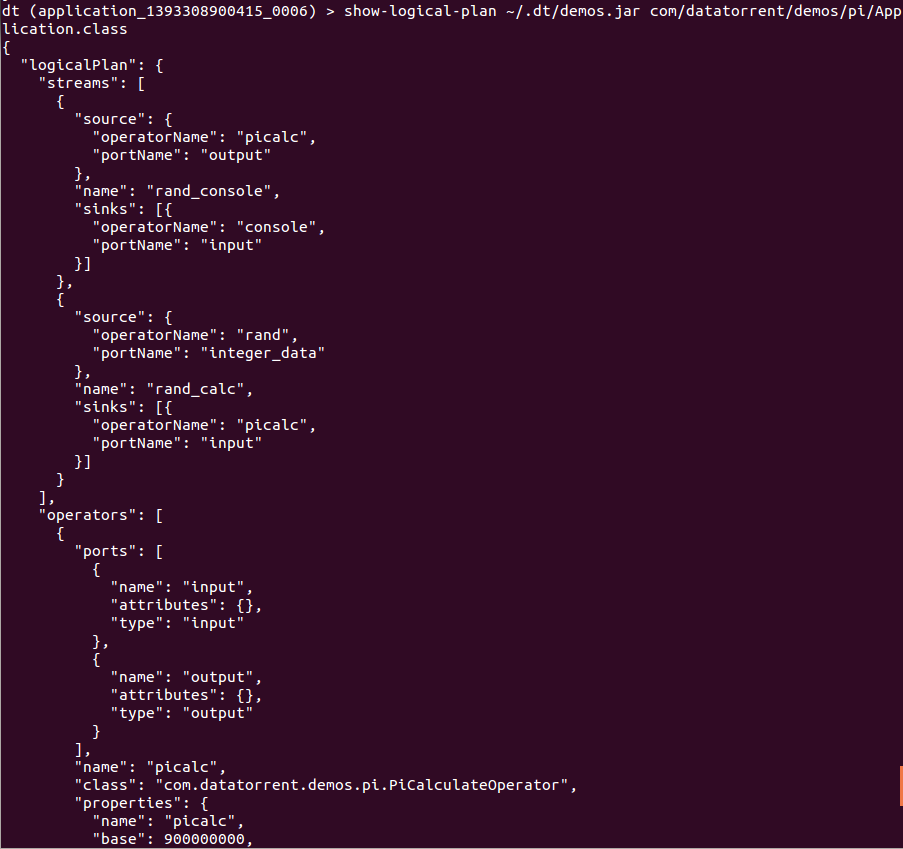
1. Get detailed information about a given application –



1. Connect to the application –

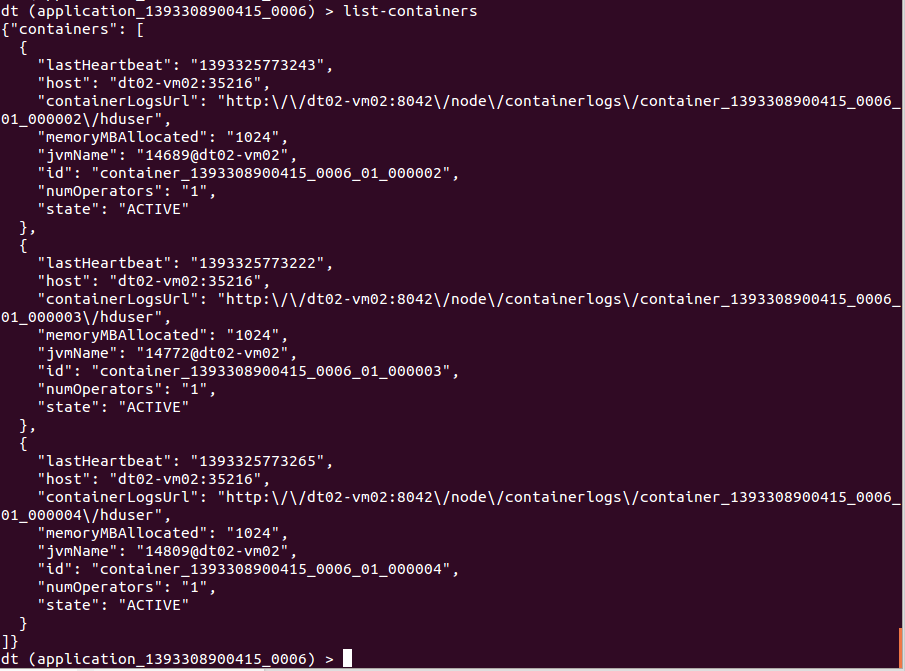


1. Show logical plan



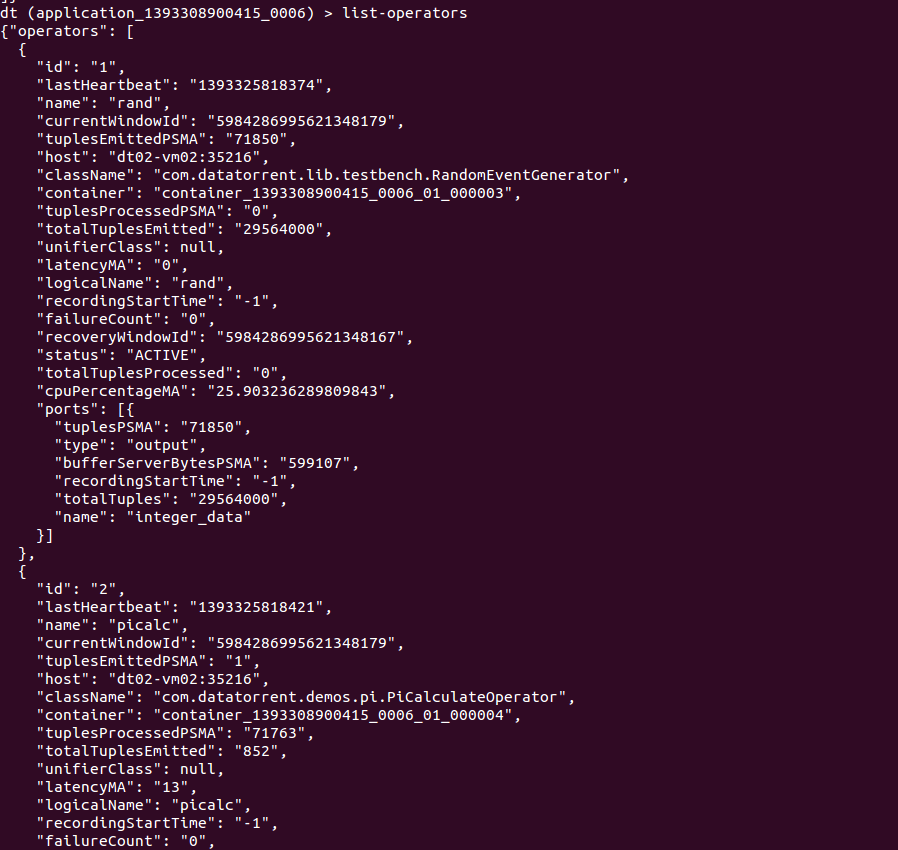
1. List the containers :

This lists the containers for the application that you are connected to, at this point in time.

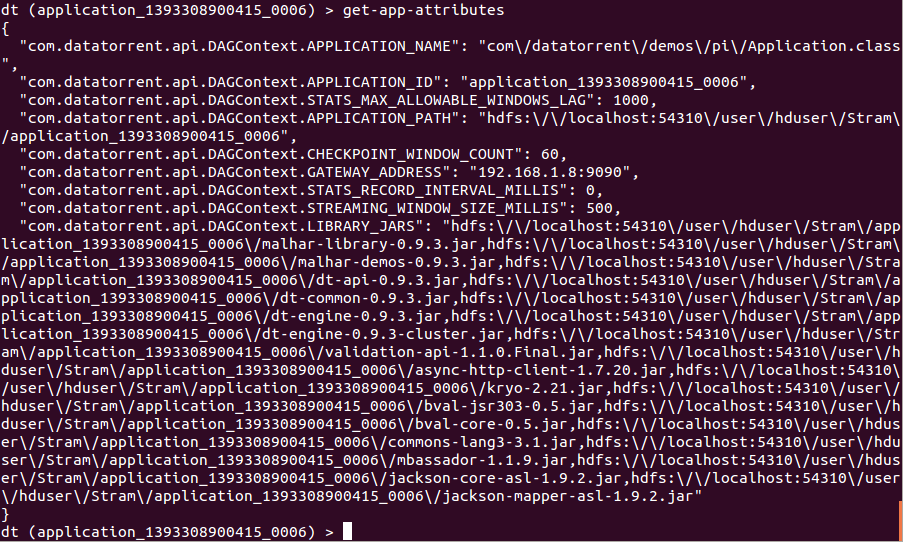


1. List all the operators –

(Note that the screenshot below captures only the first 2. Others are displayed in the same manned in order)



1. Get application attributes :

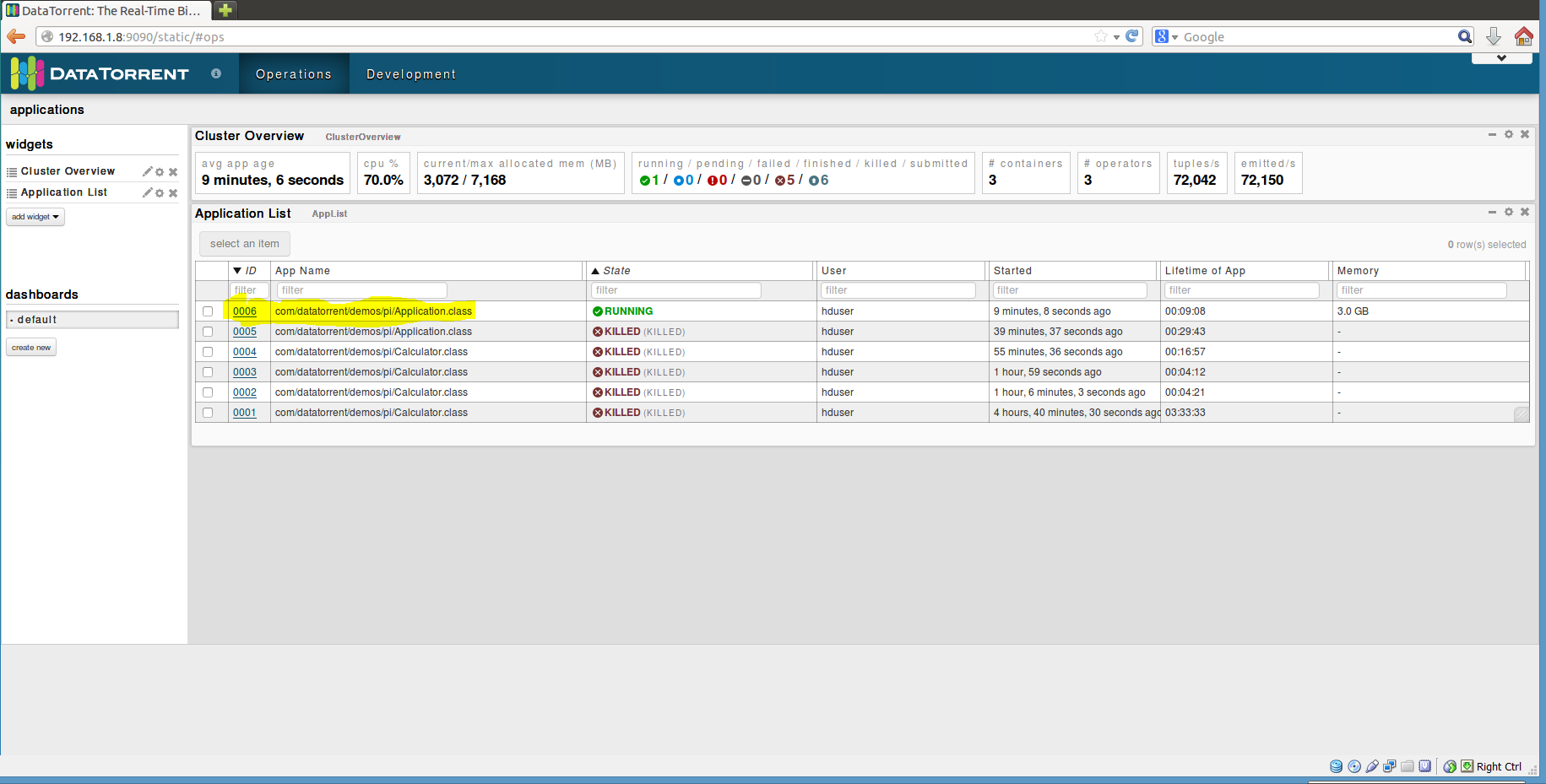


1. Similarly the user can use other dtcli to monitor/manage the application.

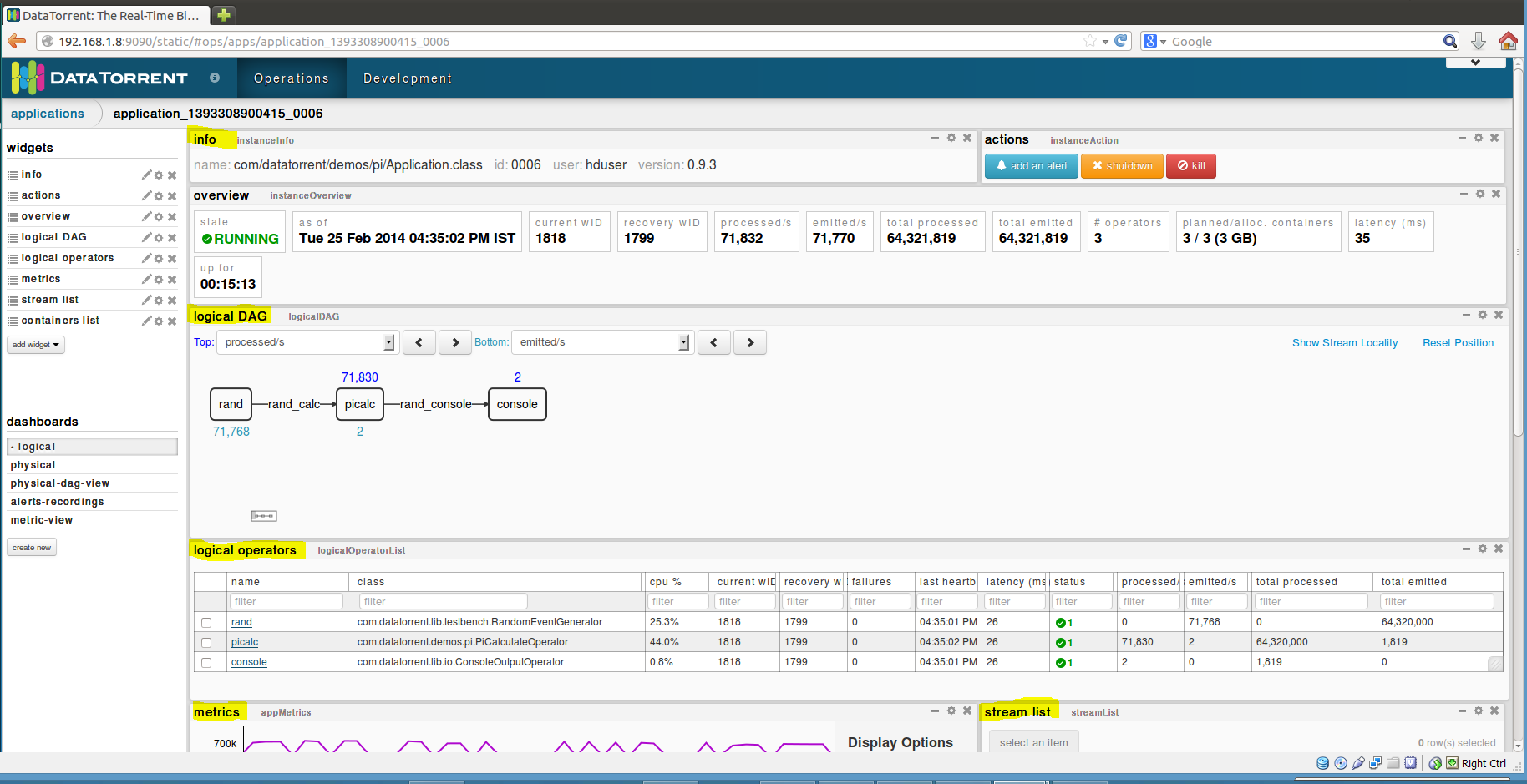
#### Using ‘Gateway Console’

# Operations

1. On the Gateway console, you will see the apps as below –



1. In order to get detailed about the application, click on ‘ID’ for the application and you will get to the screen below –

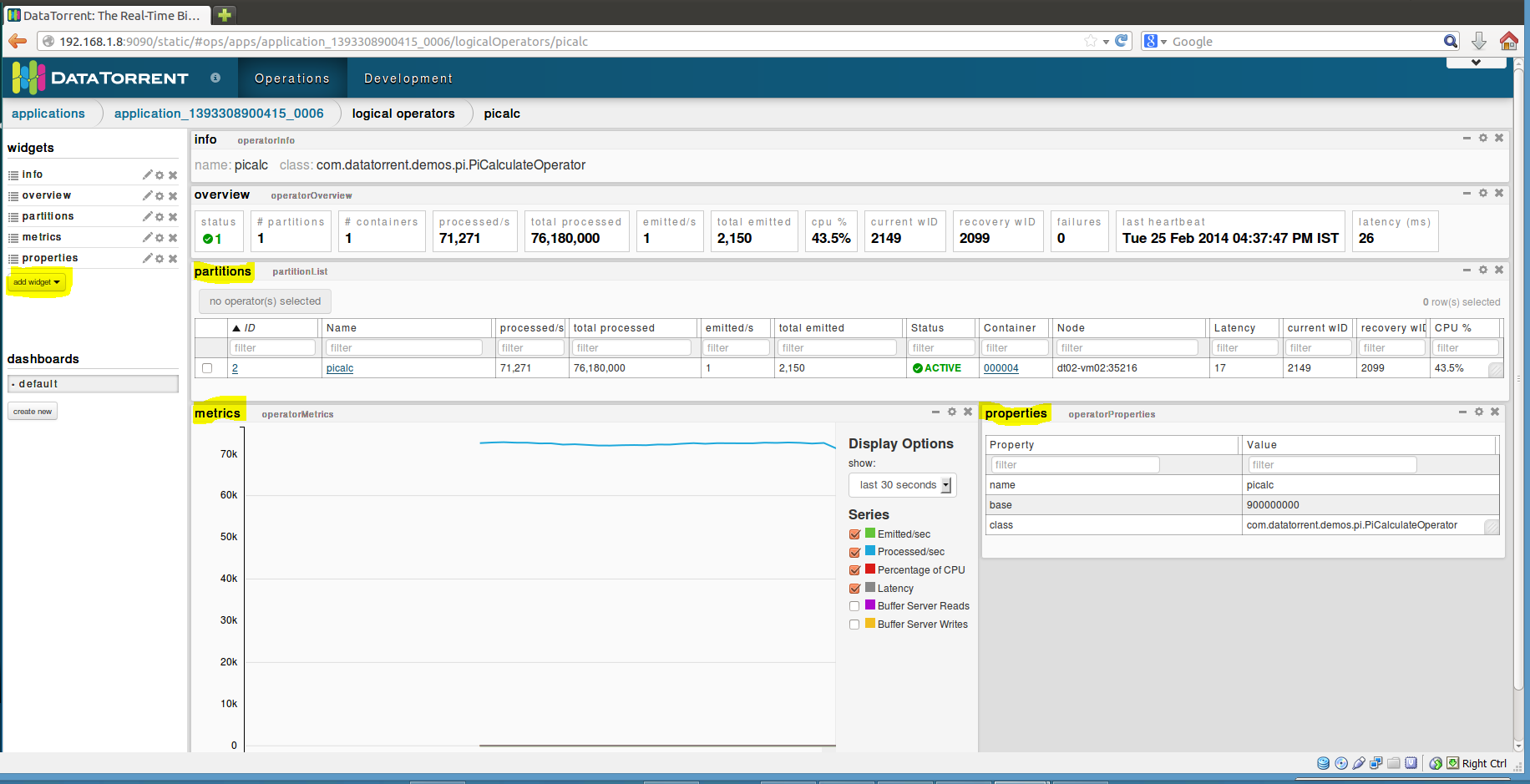


As you can see, it shows you a lot of details about the application, such as –

* Overview
* Logical DAG
* Logical operators
* Streams
* Various mertices, etc.

Additionally, you can select additional widgets as well as rearrange the widgets as per your wish.

1. Clicking on any of the operators will give you the details for that particular operator. For example, the screenshot below is for the PiCalculatorOperator.

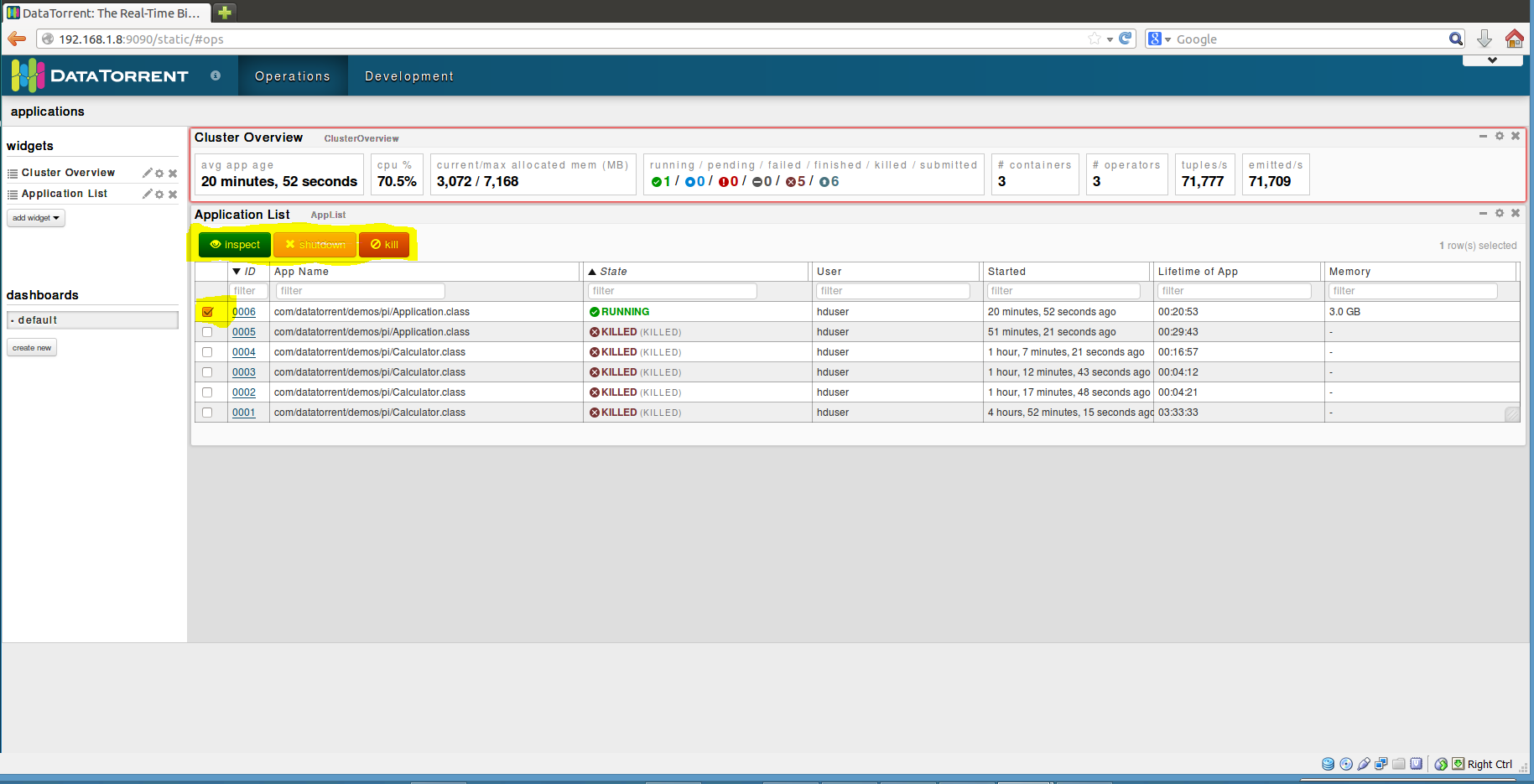


You can see detailed information about –

* partitions
* Metrices
* Properties

You can add more widgets here as well

1. Selecting an application from the list of applications also enables the operations to control the application viz. inspect, shutdown, kill. The user can click on any of the buttons to perform the corresponding operation.



# Stopping the application

From dtcli

The user can use the ‘shutdown-app’ command to shut down the application from the dtcli.

Since this is a platform for streaming applications, in most of the cases, the application would not be shutting down The user will have to kill the application. In order to achieve this, a kill-app command is provided which can be used.

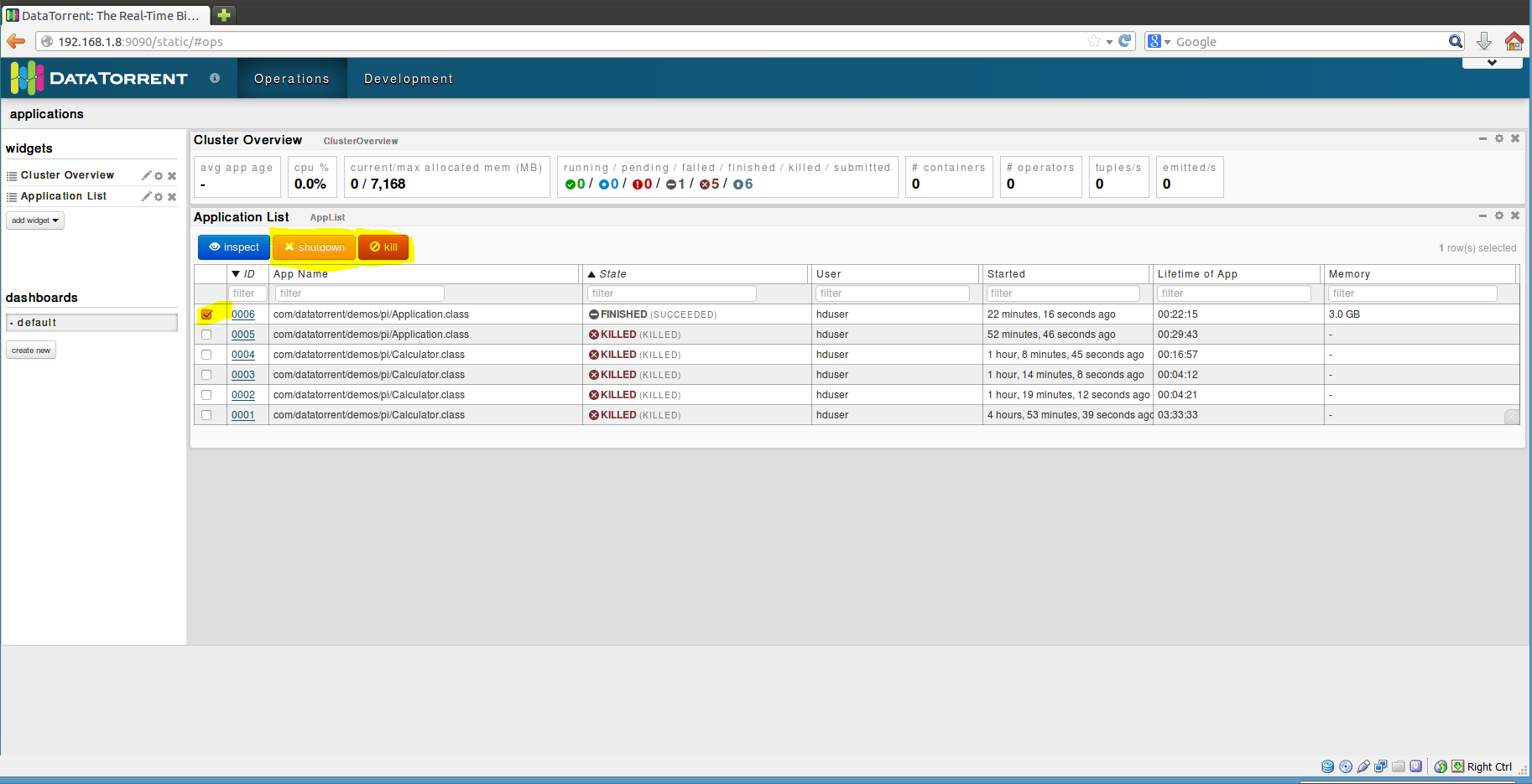


On a successful shutdown, the user will be able to see the change in state as below –

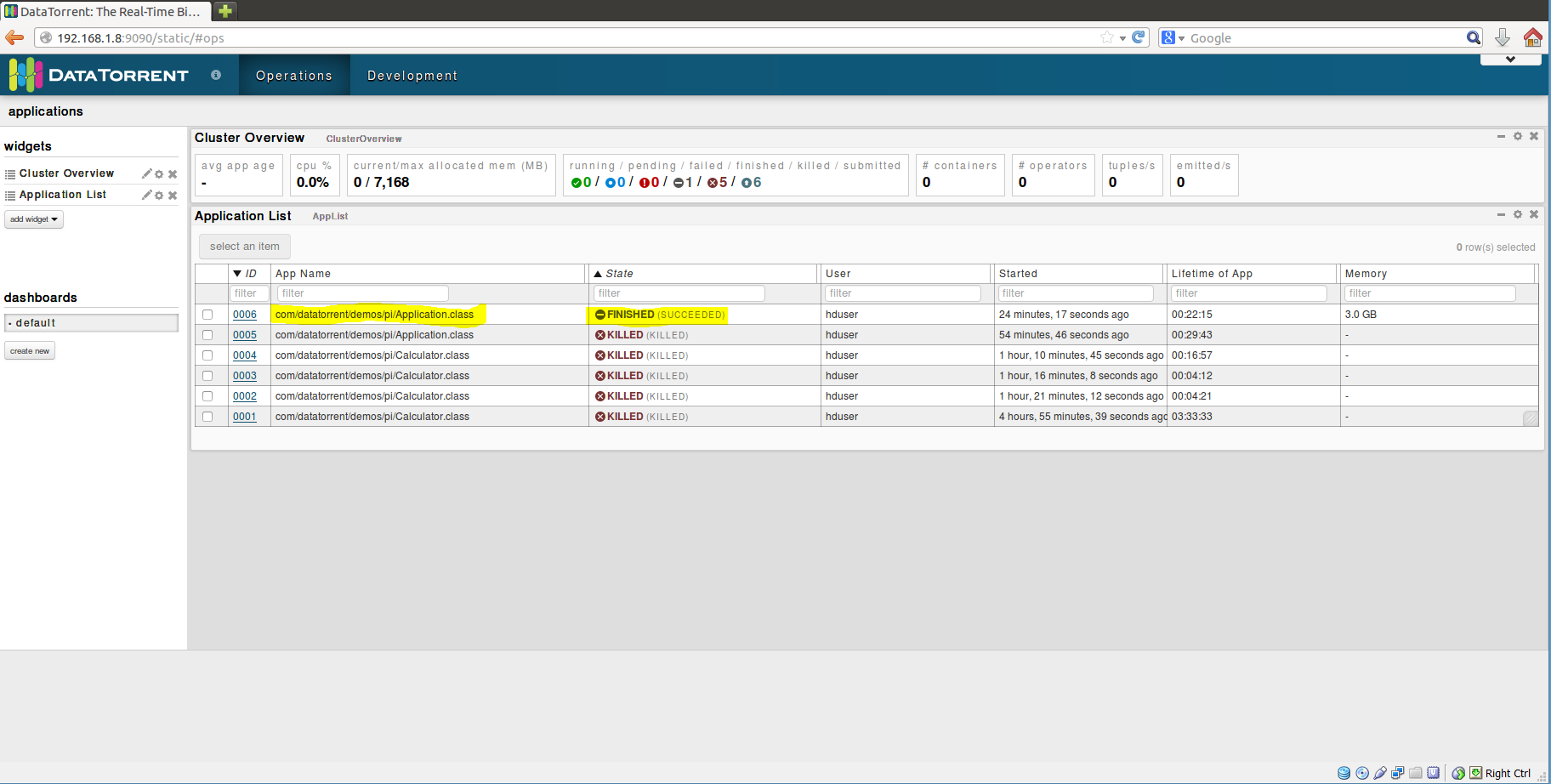


From DT Gateway Console

An application can be either shutdown or killed from the DT Gateway console. You can do so when you select an application from the list of applications. Once the user does this, the ‘shutdown’ and ‘kill buttons get activated along with the ‘inspect’ button. The user can then shutdown or kill the application using these buttons.

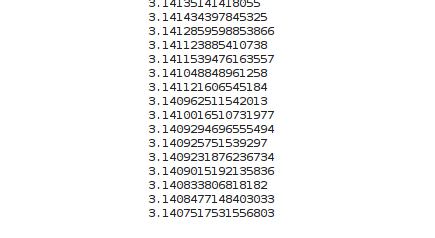


After the application has been shutdown, you will see the change in state as shown in the screen shot below –



# Conclusions

The ‘Pi’ application computes the value of ‘π’ using the Monte Carlo estimation technique. At the end, the user should be able to see the output as –



Console operator is one way to output results for demo purpose. Results can be sent to any external system such as databases, websockets, files, message bus, etc.