

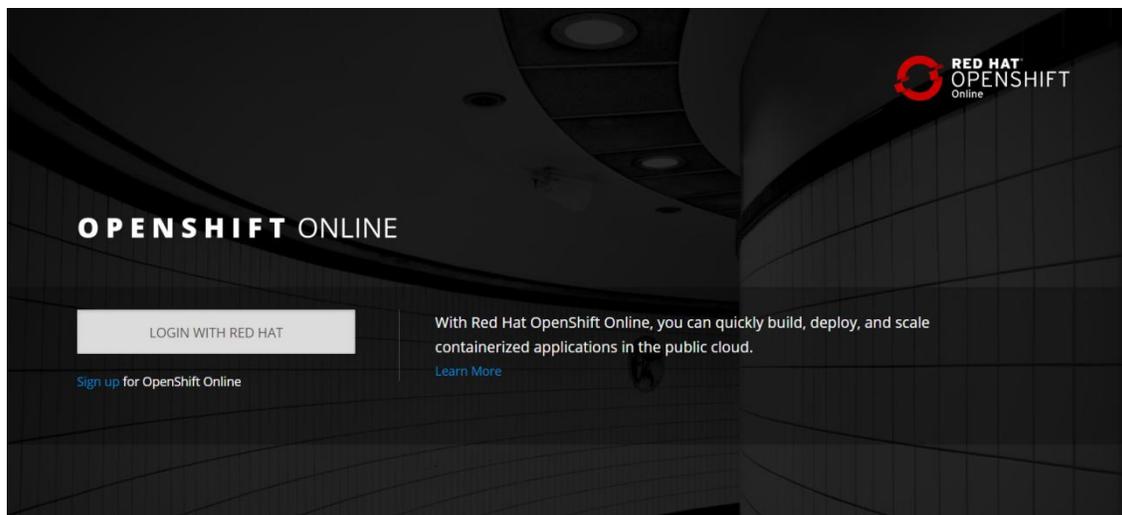
# RTView DataServer in Red Hat OpenShift - Quick Start Guide

---

*This Quick Start Guide shows you how to deploy and run RTView DataServer in Red Hat OpenShift Container Platform. It will also show you how you can push third-party data to RTView DataServer and use that data to build custom displays in RTView Cloud.*

## Get a Red Hat Open Shift Account

- Log in to your Open Shift account (<https://manage.openshift.com/>)
- If you do not have an account, sign-up for an account by clicking the 'Sign Up' link in the Open Shift online page.



- Go to <https://learn.openshift.com/introduction/> and get yourself acquainted with the following topics.
  - Getting Started with OpenShift for Developers
  - Deploying Applications from Images

The screenshot shows the OpenShift Interactive Learning Portal. At the top left is the OpenShift logo. At the top right, there is a "Report an Issue" link and a red button that says "SIGN UP TO OPENSHIFT ONLINE FOR FREE". The main header area features the Red Hat OpenShift logo and the title "Interactive Learning Portal". Below the title is a paragraph: "Our Interactive Learning Scenarios provide you with a pre-configured OpenShift instance, accessible from your browser without any downloads or configuration. Use it to experiment, learn OpenShift and see how we can help solve real-world problems." Below this are six scenario cards arranged in a 2x3 grid. Each card has a title and a red "START SCENARIO" button at the bottom.

Getting Started with OpenShift for Developers START SCENARIO	Logging in to an OpenShift Cluster START SCENARIO	Deploying Applications From Images START SCENARIO
Deploying Applications From Source START SCENARIO	Using the CLI to Manage Resource Objects START SCENARIO	Connecting to a Database Using Port Forwarding START SCENARIO

## OpenShift Playground

- Go to <https://learn.openshift.com/> and click on 'OpenShift Playgrounds' to select a provisioned system. (e.g. OpenShift 3.7 Playground).

This screenshot shows a portion of the OpenShift Interactive Learning Portal. It features the Red Hat OpenShift logo and the title "Interactive Learning Portal". Below the title is the same introductory paragraph as seen in the first screenshot. Below the text are two cards for "OpenShift 3.6 Playground" and "OpenShift 3.7 Playground", each with a red "START SCENARIO" button at the bottom.

OpenShift 3.6 Playground START SCENARIO	OpenShift 3.7 Playground START SCENARIO
--	--

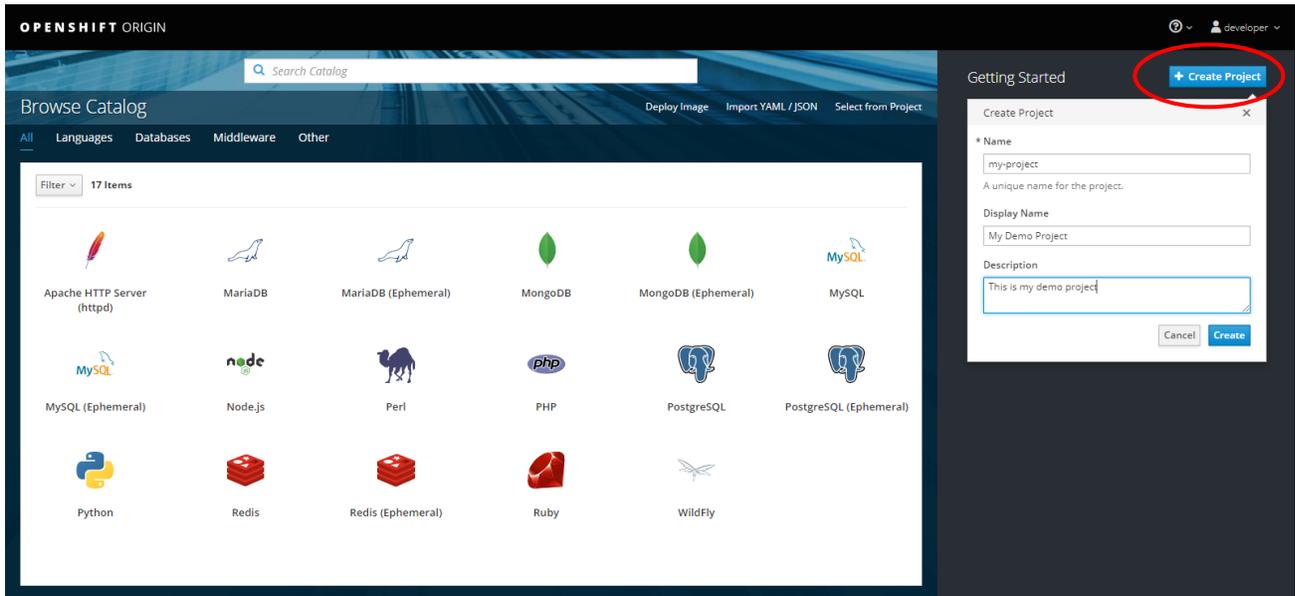
- Login to the Cluster using the '**Terminal**' as 'developer' and create a new project (e.g. my-project). You can also use the 'Dashboard' to complete this step, which is described below.

The screenshot shows the OpenShift Playground interface. On the left, there is a sidebar with the title 'OpenShift Playground (OpenShift 3.7)'. Below the title, there is a section titled 'OpenShift Playground' with introductory text. A section titled 'Logging in to the Cluster' provides instructions on how to log in as a cluster admin or a distinct user. A code block shows the command `oc login -u developer -p developer`. Below this, there is a list of credentials: Username: developer. On the right, there is a 'Terminal' tab showing the execution of the `.launch.sh` script, which starts OpenShift and logs in as the 'developer' user. The terminal output shows 'Login successful.' and a message: 'You don't have any projects. You can try to create a new project, by running `oc new-project <projectname>`'. The 'Dashboard' tab is also visible but not active. The top right corner has a 'Report an Issue' link and a 'SIGN UP TO OPENSIFT ONLINE FOR FREE' button. The bottom right corner says 'Powered by KataCoda'.

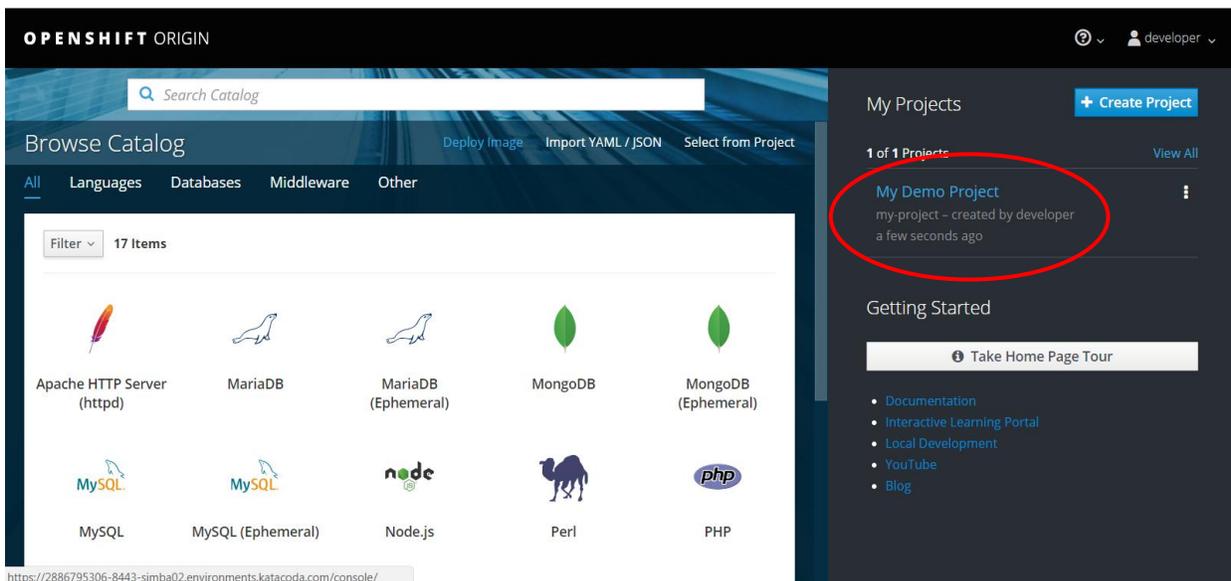
- Click on the '**Dashboard**' to login to Open Shift Origin, the web console. (login: developer/developer).

The screenshot shows the OpenShift Origin login page. The page has a dark background with the 'OPENSIFT ORIGIN' logo in the top right corner. Below the logo, there is a login form with two input fields: 'Username' with the value 'developer' and 'Password' with masked characters. A blue 'Log In' button is positioned below the password field. To the right of the login form, there is a message: 'Welcome to OpenShift Origin.'.

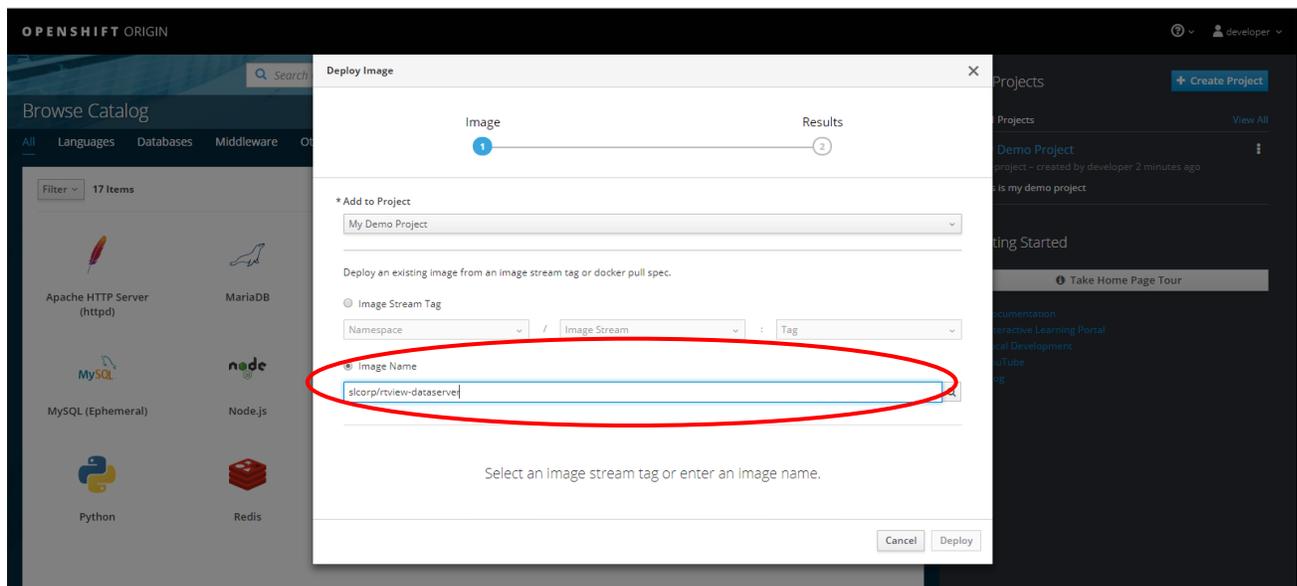
- Click **'Create Project'** to create a new project by filling in your project details (e.g. my-project).



- After entering the details of your project, click the **'Create'** button to create your new project. You can see the new project as shown below.



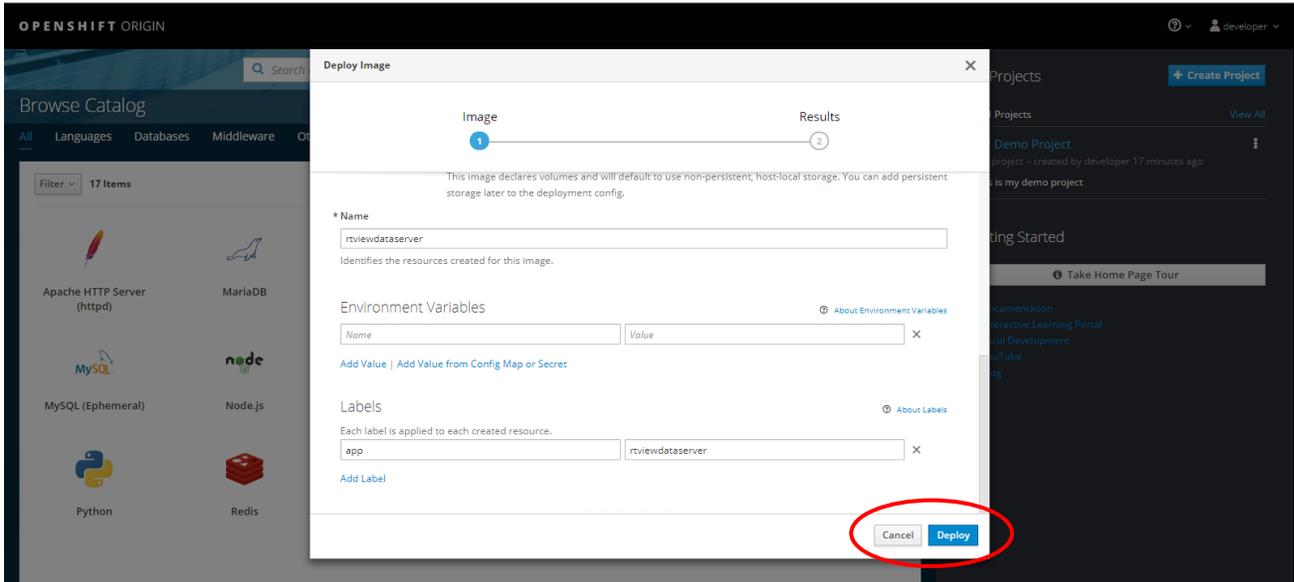
- Click '**Deploy Image**' link.
- Enter the details in the '**Deploy Image**' dialog as shown below.
  - Select your project (e.g my-project).
  - Enter 'Image Name' as [registry.connect.redhat.com/slcorp/rtview-dataserver](https://registry.connect.redhat.com/slcorp/rtview-dataserver).
  - After selecting the image name , click on the magnifying class to confirm the image metadata.
- About Red Hat Connect Container Registry
  - You can download the RTView DataServer image in the Red Hat Container catalog. <https://access.redhat.com/containers/?type=products#/search/>
  - In order to use this RTView DataServer image, you need:
    - Red Hat Registry Authorization
    - Accept RedHat subscription agreement and Red Hat Connect Certified Container Partner's (SL Corp) terms.
    - Review the following product documentation located at <https://access.redhat.com/containers/?tab=images&platform=openshift#/registry.connect.redhat.com/slcorp/rtview-dataserver>



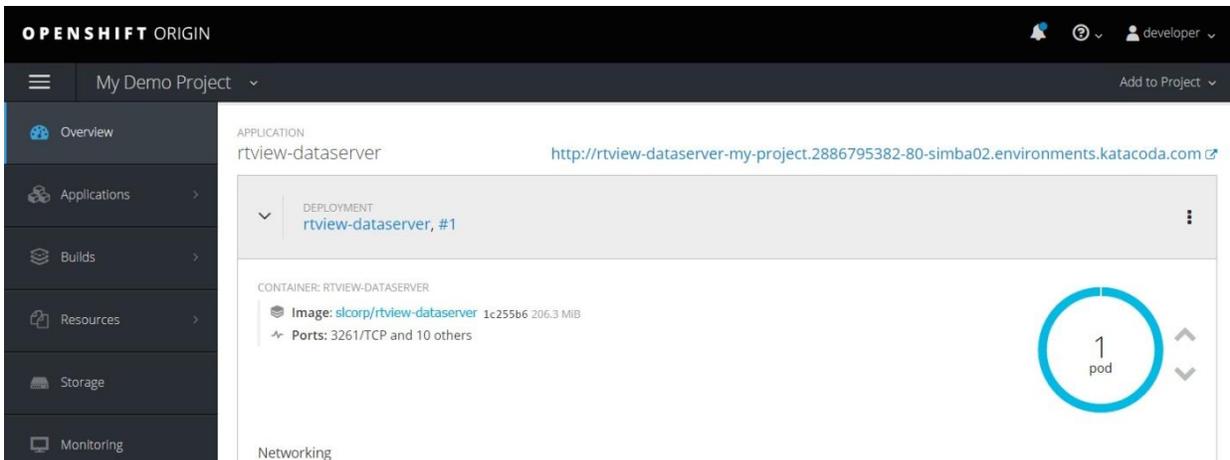
- After finding the image, enter the **'Name'** as 'rtview-dataserver'.

Note: If the image name is xxx/rtview-dataserver, then the name will be rtview-dataserver.

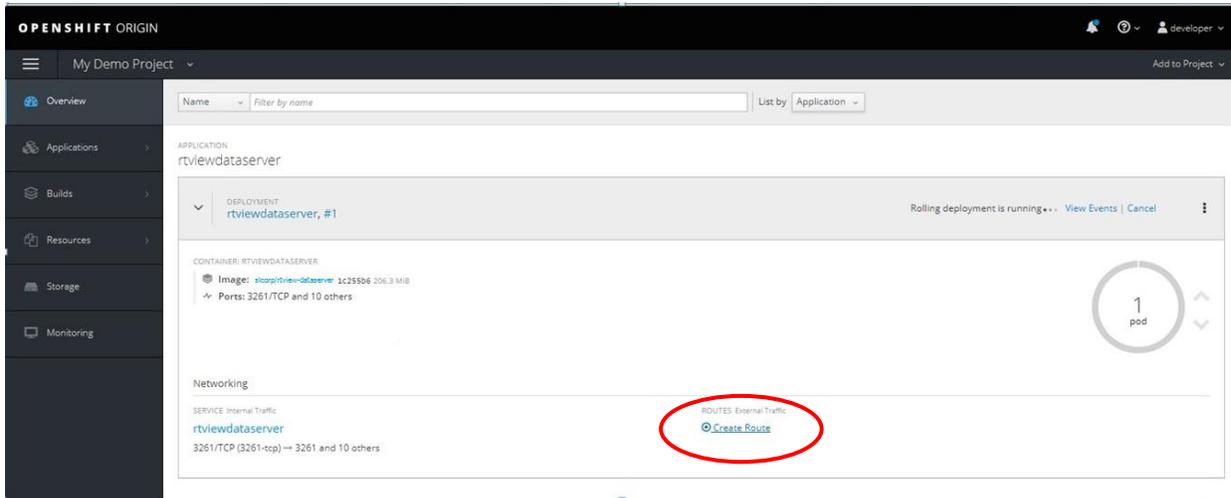
- Click the **'Deploy'** button to deploy the RTView DataServer image.



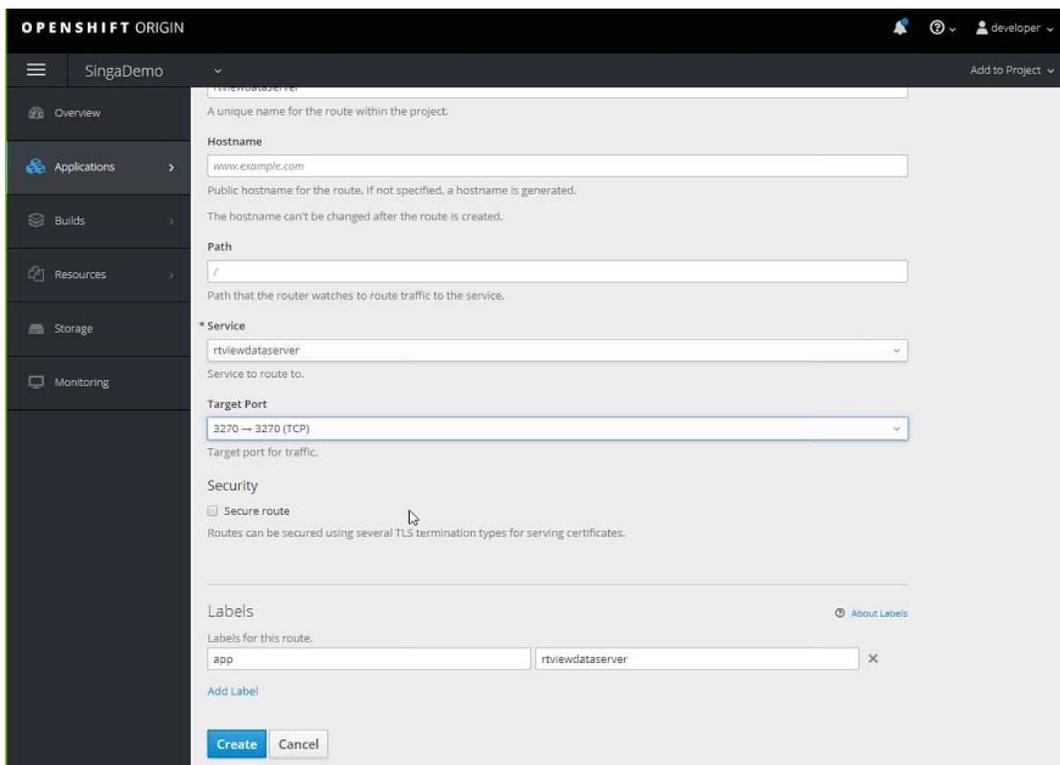
- RTView DataServer image is now deployed in OpenShift. You can see the newly deployed image in your project page.



- Create a Route for this image, by clicking **'Create Route'**. This step is needed so that the RTView DataServer can be accessed from outside OpenShift.



- Enter **'Service'** name as 'rtview-dataserver' and **'Select Target Port'** as 3270 from the **Target Port** drop down. Click **'Create'** button to create the new route.



- You can now see the newly created Route for your RTView DataServer (highlighted in red).

Note: Your newly created route will work only if the pod is running. You need to give some time for OpenShift to deploy and run your route. The pod is ready when its color turn blue as seen in the screenshot below.

OPENSIFT ORIGIN

My Demo Project

Overview

Applications

Builds

Resources

Storage

Monitoring

APPLICATION rtview-dataserver <http://rtview-dataserver-my-project.2886795382-80-simba02.environments.katacoda.com>

DEPLOYMENT rtview-dataserver, #1

CONTAINER: RTVIEW-DATASERVER

Image: slcorp/rtview-dataserver 1c255b6 206.3 MIB

Ports: 3261/TCP and 10 others

1 pod

Networking

SERVICE Internal Traffic

rtview-dataserver

3261/TCP (3261-tcp) → 3261 and 10 others

ROUTES External Traffic

<http://rtview-dataserver-my-project.2886795382-80-simba02.environments.katacoda.com>

Route rtview-dataserver, target port 3270-tcp

## Testing

- You can test by opening the above web link in a browser and run a `rtvquery` on the URL

<YOUR\_ROUTE\_URL>/[rtvquery/cache/RTViewDs/Tables?fmt=json](#)

(e.g. <http://rtview-dataserver-my-project.2886795382-80-simba02.environments.katacoda.com/rtvquery/cache/RTViewDs/Tables?fmt=json>)

You can view the contents of your RTView DataServer cache with the above URL (i.e. assuming your RTView DataServer is running and its Caches are populated with data).

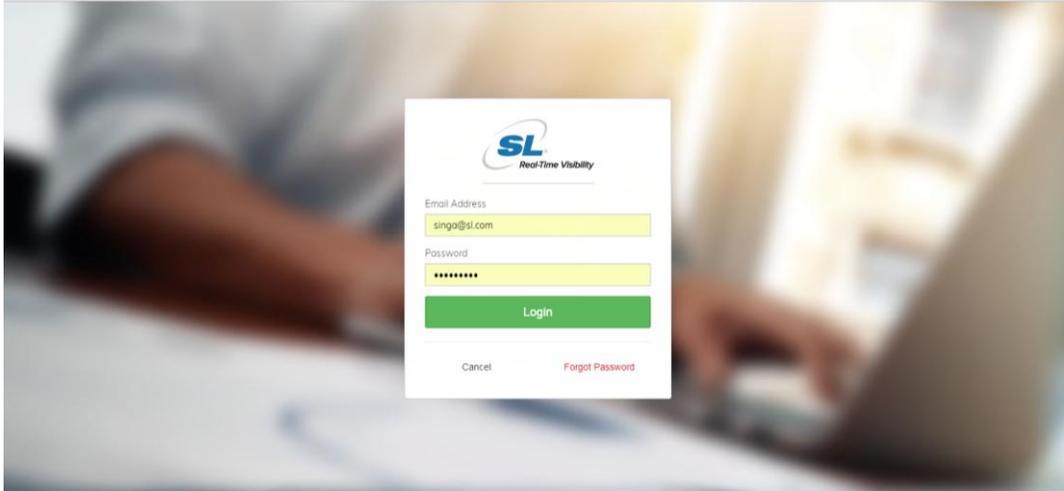
- Your route URL is pointing to the RTView DataServer running on port 3270.



```
{
  "metadata": [
    { "name": "Table", "type": "string" },
    { "name": "Rows", "type": "int" },
    { "name": "Columns", "type": "int" },
    { "name": "Memory", "type": "int" }
  ],
  "data": [
    { "Table": "JmxStatsTotals.current", "Rows": 1, "Columns": 4, "Memory": 441, "%Full": "" },
    { "Table": "JmxStatsTotals.current_condensed", "Rows": 1, "Columns": 4, "Memory": 441, "%Full": "" },
    { "Table": "JmxStatsTotals.history", "Rows": 22, "Columns": 4, "Memory": 1029, "%Full": 17.09 },
    { "Table": "JmxStatsTotals.history_combo", "Rows": 0, "Columns": 0, "Memory": 0, "%Full": 0.0 },
    { "Table": "JmxStatsTotals.history_condensed", "Rows": 4, "Columns": 4, "Memory": 525, "%Full": 0.02 },
    { "Table": "RTViewDs.CacheDefs", "Rows": 11, "Columns": 12, "Memory": 8330, "%Full": "" },
    { "Table": "RTViewDs.CacheObjectProperties", "Rows": 11, "Columns": 32, "Memory": 16222, "%Full": "" },
    { "Table": "RTViewDs.Tables", "Rows": 11, "Columns": 5, "Memory": 1591, "%Full": "" },
    { "Table": "RtvAlertMapByCI.current", "Rows": 0, "Columns": 5, "Memory": 464, "%Full": "" },
    { "Table": "RtvAlertSourceStats.current", "Rows": 0, "Columns": 0, "Memory": 0, "%Full": 0.0 },
    { "Table": "RtvAlertStatsByCI.current", "Rows": 0, "Columns": 5, "Memory": 477, "%Full": "" },
    { "Table": "RtvAlertStatsByCIAndAlertGroup.current", "Rows": 0, "Columns": 0, "Memory": 0, "%Full": 0.0 },
    { "Table": "RtvAlertStatsByCategoryIndex.current", "Rows": 0, "Columns": 7, "Memory": 673, "%Full": "" },
    { "Table": "RtvAlertStatsByPackageIndex.current", "Rows": 0, "Columns": 6, "Memory": 583, "%Full": "" },
    { "Table": "RtvAlertTable.current", "Rows": 0, "Columns": 21, "Memory": 1952, "%Full": "" },
    { "Table": "RtvAlertTable.history", "Rows": 0, "Columns": 11, "Memory": 1026, "%Full": 0.0 },
    { "Table": "RtvCacheMapByCI.current", "Rows": 0, "Columns": 5, "Memory": 477, "%Full": "" },
    { "Table": "RtvCacheMapByCIType.current", "Rows": 0, "Columns": 0, "Memory": 0, "%Full": 0.0 },
    { "Table": "RtvDataServerConnections.current", "Rows": 0, "Columns": 0, "Memory": 0, "%Full": 0.0 }
  ],
  "queryStatus": 0,
  "queryStatusText": "OK"
}
```

## Get a RTView Cloud Account

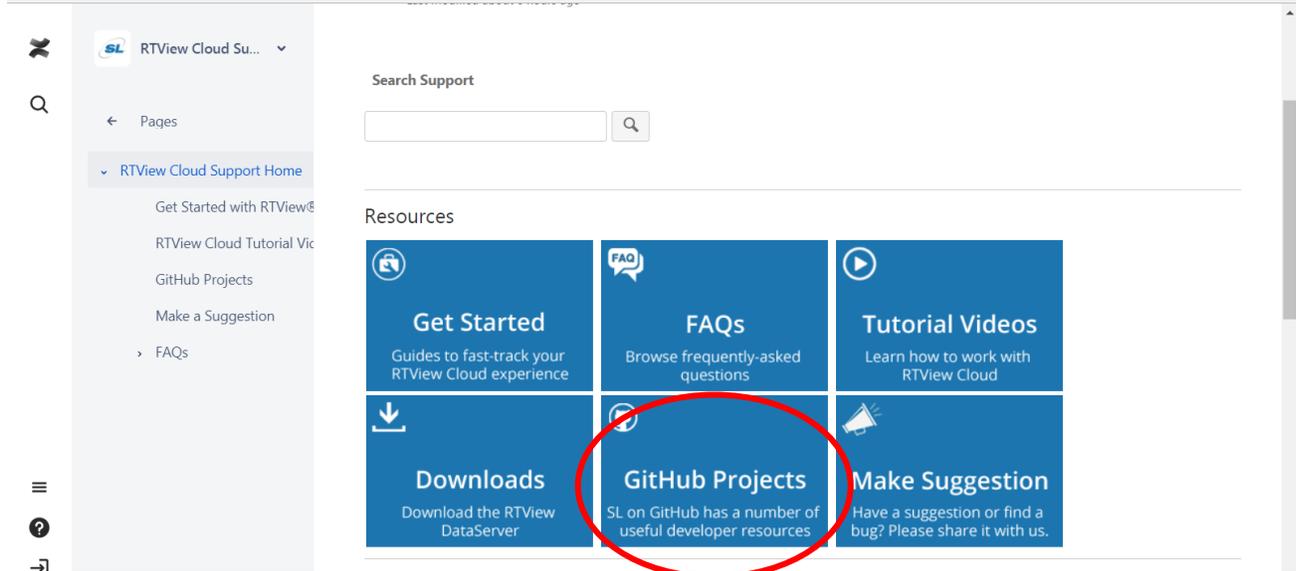
- Log in to your RTView Cloud account (<https://rtviewcloud.sl.com>).
- If you do not have a RTView Cloud account, sign-up for an account at <http://rtviewcloud.sl.com/register> and follow the instructions in your email to activate your account.



- Click on the '?' icon at upper right to go to the 'RTView Cloud Support' page.

A screenshot of the RTView Cloud dashboard. The top navigation bar includes 'Design', 'View', 'Publish', and 'Data'. The user profile 'singa/SingaNarayan...' is visible in the top right, with a red circle highlighting a question mark icon. The main content area is divided into several sections: 'Account information' (Username: singa, Email: singa@sl.com), 'Organizations' (listing various demo and private orgs), 'Custom Views Summary' (table with 2 My Custom Views and 0 Other users), 'Dashboards' (table with 0 My Dashboards and 0 Other users), and 'Components Summary'. A 'Favorites' section at the bottom states 'You do not currently have any favorites. Please visit Custom, Components, or Dashboards to create some favorites.'

- Click the '**GitHub Projects**' section in the 'RTView Cloud Support Page'.



- While you are in the <https://github.com/slcorp> page, select one of the IoT Projects listed.

You are welcome to choose any of the RTView IoT projects listed in our Github page. Each IoT project has its own README file that describes all the steps necessary to run the project. These projects are RTView-Artik, RTView-ClearBlade, RTView-LightStreamer and RTView-PubNub.

For this example, we'll be using 'RTView ClearBlade' IoT project. Click on the **RTView-ClearBlade** link.

SL Corporation  
SL Corporation Open Source Repository  
240 Tamal Vista Blvd. Cort... http://www.sl.com info@sl.com

Repositories 10 People 0

Pinned repositories

- RTView-MySQL-DB  
RTView MySQL DB  
★ 2 🍴 1
- RTView-ClearBlade**  
RTView Integration with ClearBlade  
JavaScript ★ 1
- RTView-InfluxDB  
RTView InfluxDB  
Shell ★ 1
- RTView-R-Analytics  
RTView - R Analytics Repository  
★ 1 🍴 1
- RTView-Splunk-Integration  
RTView Splunk Integration  
★ 1
- RTView-PubNub  
RTView Integration with PubNub  
JavaScript

Search repositories... Type: All Language: All

- Click on the **'Clone or download'** button to get a copy of the project to your computer.

slcorp / RTView-ClearBlade Watch 4 Star 1 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Insights

Join GitHub today  
GitHub is home to over 20 million developers working together to host and review code, manage projects, and build software together.  
Sign up

RTView Integration with ClearBlade  
rtview clearblade clearblade-iot-server custom-displays

102 commits 4 branches 0 releases 3 contributors

Branch: master New pull request Find file **Clone or download**

Mehran-S Merge pull request #28 from slcorp/mehran-testing Latest commit fbfaaa a day ago

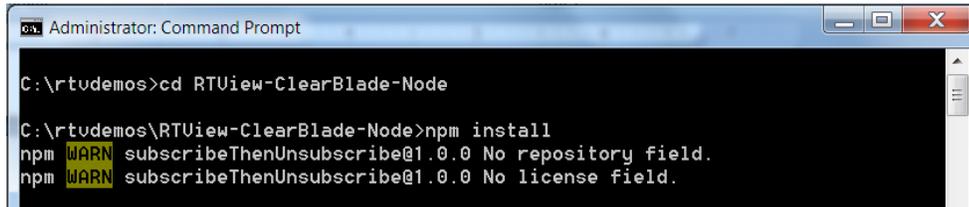
- Images ms - images for the README 2 months ago
- RTView-ClearBlade-Displays Addition of a sub-model instance for the map and updated displays a day ago
- RTView-ClearBlade-Node updated for non-local data servers a day ago
- COPYRIGHT changed to 2018 3 months ago
- LICENSE Initial commit 5 months ago
- README.md Update README.md 2 months ago

## Run the RTView ClearBlade Connector

- To install the connector program:

```
cd RTView-ClearBlade-Node
```

```
npm install
```



```
Administrator: Command Prompt
C:\rtvdemos>cd RTView-ClearBlade-Node
C:\rtvdemos\RTView-ClearBlade-Node>npm install
npm WARN subscribeThenUnsubscribe@1.0.0 No repository field.
npm WARN subscribeThenUnsubscribe@1.0.0 No license field.
```

- To start the program:

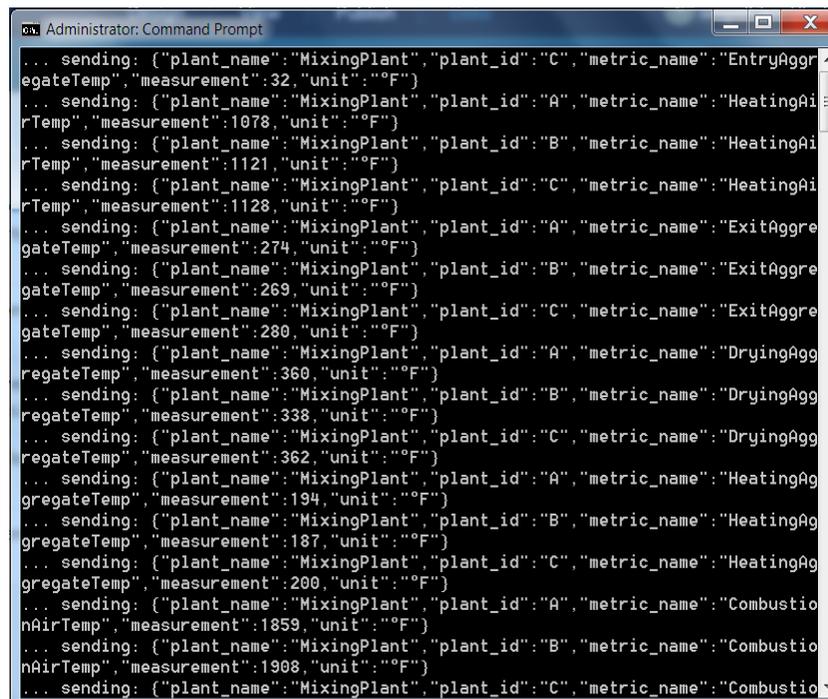
```
node rtview_clearblade_feed.js http://<YOUR_ROUTE_URL>/rtvpost
```

e.g.

```
node rtview_clearblade_feed.js http://rtview-dataserver-my-project.2886795382-80-simba02.environments.katacoda.com/rtvpost
```

Note that your route URL might be different from the above example depending on your project and environment settings in OpenShift. Please get the correct route URL from OpenShift and use it here.

You will see messages printed in the console showing successful subscriptions made.

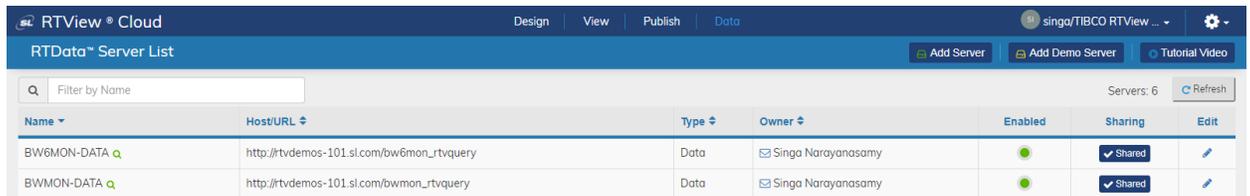


```
Administrator: Command Prompt
... sending: {"plant_name":"MixingPlant","plant_id":"C","metric_name":"EntryAggr
egateTemp","measurement":32,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"A","metric_name":"HeatingAi
rTemp","measurement":1078,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"B","metric_name":"HeatingAi
rTemp","measurement":1121,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"C","metric_name":"HeatingAi
rTemp","measurement":1128,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"A","metric_name":"ExitAggre
gateTemp","measurement":274,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"B","metric_name":"ExitAggre
gateTemp","measurement":269,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"C","metric_name":"ExitAggre
gateTemp","measurement":280,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"A","metric_name":"DryingAgg
regateTemp","measurement":360,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"B","metric_name":"DryingAgg
regateTemp","measurement":338,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"C","metric_name":"DryingAgg
regateTemp","measurement":362,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"A","metric_name":"HeatingAg
gregateTemp","measurement":194,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"B","metric_name":"HeatingAg
gregateTemp","measurement":187,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"C","metric_name":"HeatingAg
gregateTemp","measurement":200,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"A","metric_name":"Combustio
nAirTemp","measurement":1859,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"B","metric_name":"Combustio
nAirTemp","measurement":1908,"unit":"°F"}
... sending: {"plant_name":"MixingPlant","plant_id":"C","metric_name":"Combustio
```

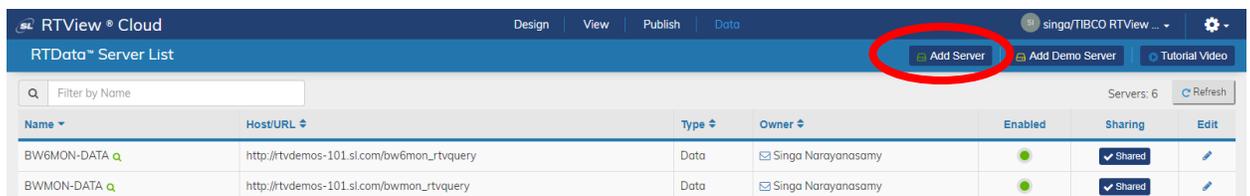
## Setup RTView DataServer in RTView Cloud

- Login to RTView Cloud and click on the top menu 'Data'.

This will take you to the **RTData Server List** page.



- Click on **Add Server** button. This will bring up a dialog to enter name, URL and type of the data server.



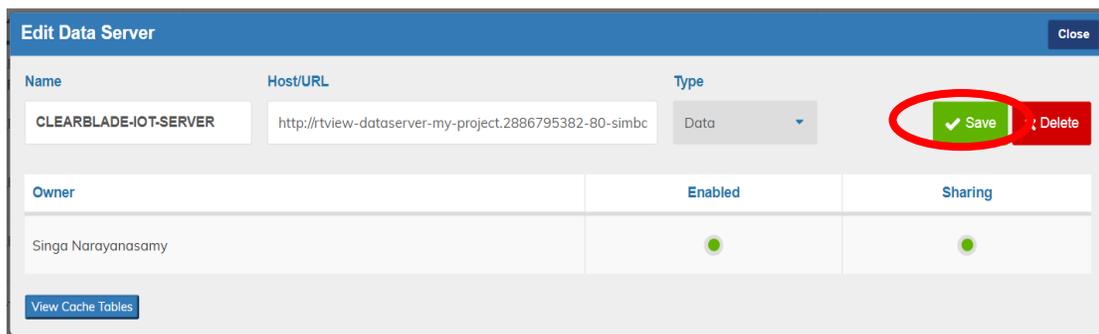
- For Name, type: CLEARBLADE-IOT-SERVER
- For Host/URL, type: <YOUR\_ROUTE\_URL>/rtvquery

(e.g. <http://rtview-dataserver-my-project.2886795382-80-simba02.environments.katacoda.com/rtvquery>)

Note that your route URL might be different from the above example depending on your project and environment settings in OpenShift. Please get the correct route URL from OpenShift and use it here.

- For Type, select 'Data'.

- After filling in the details, click on **Save Added Servers** button.



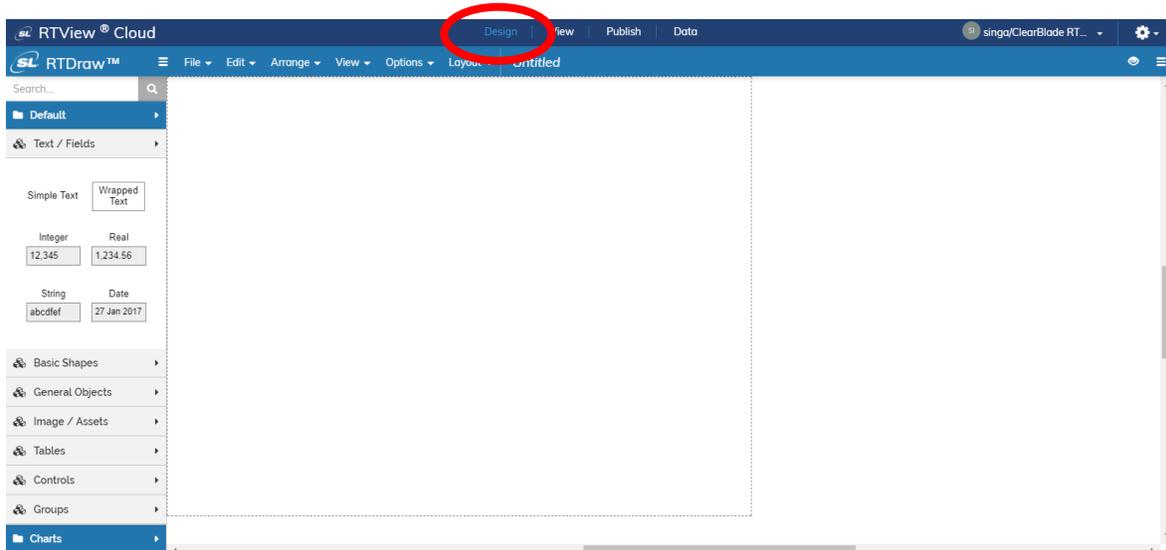
- To test your RTView DataServer connection, click the green magnifying glass next to the CLEARBLADE-IOT-SERVER. This will bring up the **RTView DataServer - Cache Tables** dialog.
  - You should see "**Connected**" under Connection Status.  
You should see all the available Caches listed under **CacheTable**.

CacheTable	Rows	Cols	Memory
ClearBladeCache	30	8	7,401
JmxStatsTotals	1	4	441
RtvAlertMapByCI	0	5	464
RtvAlertSourceStats	0	0	0
RtvAlertStatsByCI	0	5	477
RtvAlertStatsByCIAndAlertGroup	0	0	0
RtvAlertStatsByCategory/Index	0	7	673
RtvAlertStatsByDestination/Device	0	0	0

- Close the dialog.

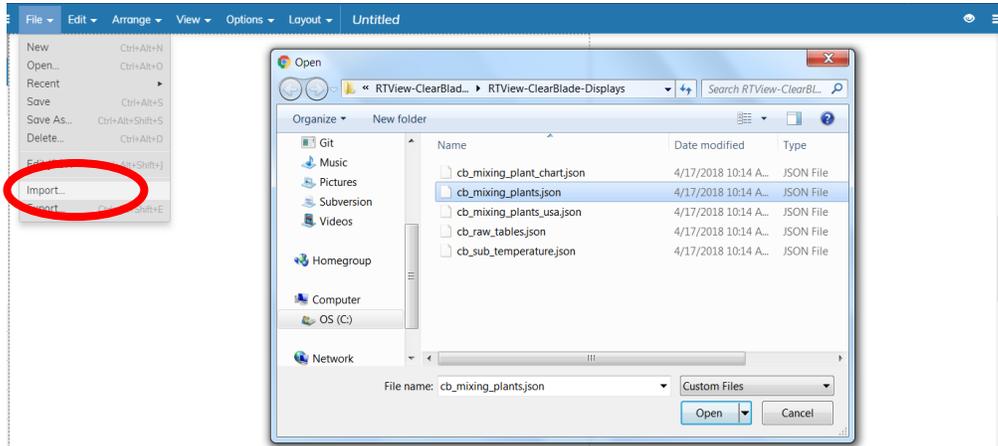
## Import and View RTView ClearBlade Displays in RTView Cloud

- Login to RTView Cloud and click on the top menu '**Design**' This will take you to **RTDraw** page.

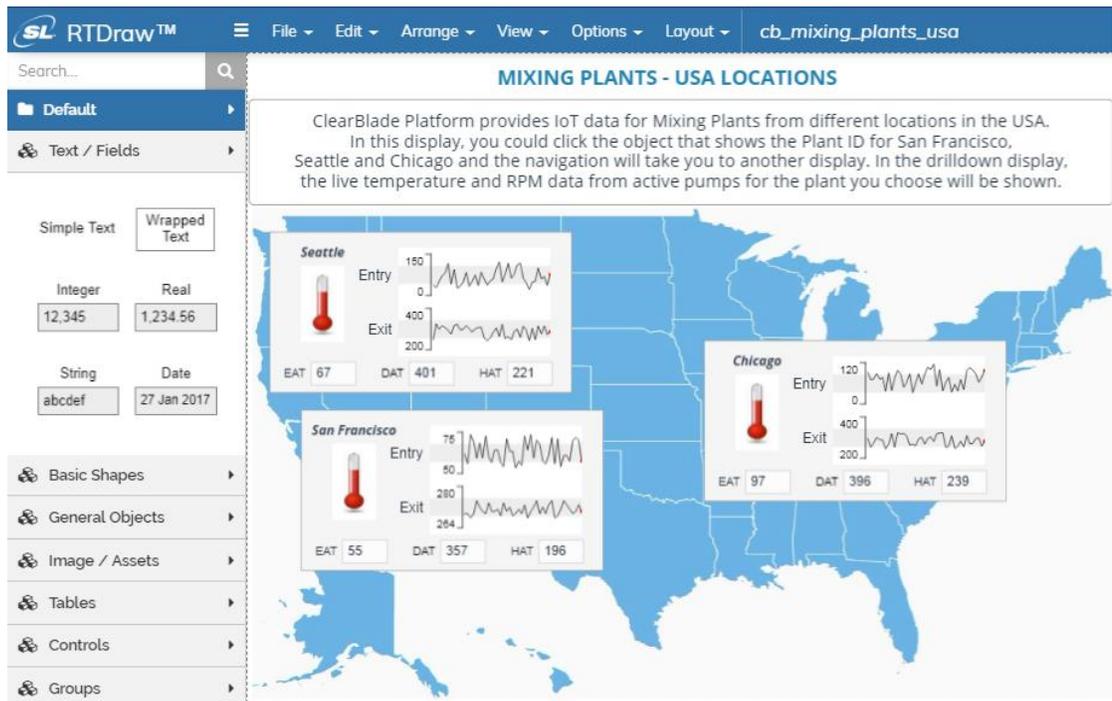


- Click 'File' menu and select 'Import...'

Navigate to 'RTView-ClearBlade-Displays' and choose one or more of the displays to import.



- After completing the import, click 'File' menu and select 'Open'. Choose one of the displays to open (e.g. cb\_mixing\_plants\_usa.json).
- You will see the display filled with data from RTView ClearBlade connector program.
- You can repeat this step to see data in other displays as well.



## **CONGRATULATIONS!**

**You've successfully completed the following tasks:**

- **Deployed and ran the RTView DataServer in Red Hat OpenShift.**
- **Send third-party data to RTView DataServer in OpenShift.**
- **Viewed pre-built displays in RTView Cloud using this data.**