What is Spexy?

Spexy is an easy to use APEX debug log Visualization tool that Provides an easy to use interface to identify bottlenecks

What does Spexy provide?

It provides an insights based on tracing an entire user flow (use case) for recursion or optimization options

When do you use Spexy?

1. Single user testing
2. Scenario tracing for apex

Setup

Setup docker

Create a folder called Spexy
Get the Spexy yml for docker from Git / or the yml file provided
https://github.com/mgangadhar/spexy

Save it in the Spexy folder

Navigate to the Spexy folder where the yml exists

Execute the command: docker-compose pull

Once it is done Execute command: docker-compose up -d

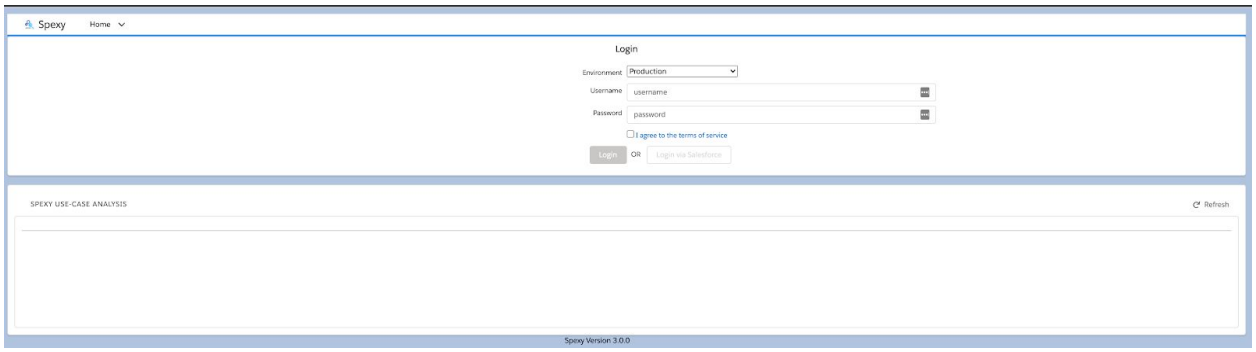
To shut down docker use docker-componse down

Once it is done you should be able to navigate to:

http://localhost:8001/

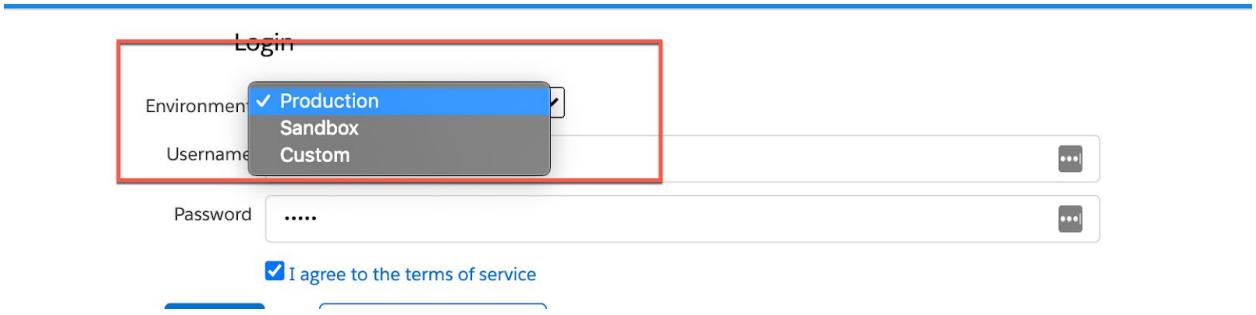
NOTE: **Do not change the port**

You should be able to see Spexy start page:



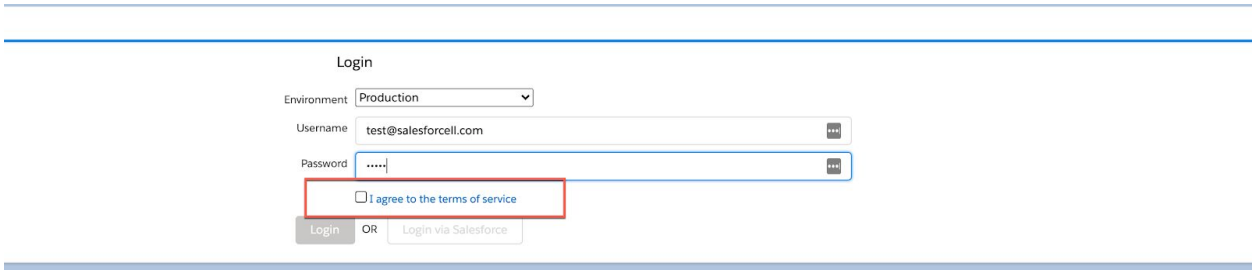
Spexy Login options

- Choose the right Environment
  Incase of mydomain choose Custom



- Username and password:
  This is the standard way to login using your salesforce username and password

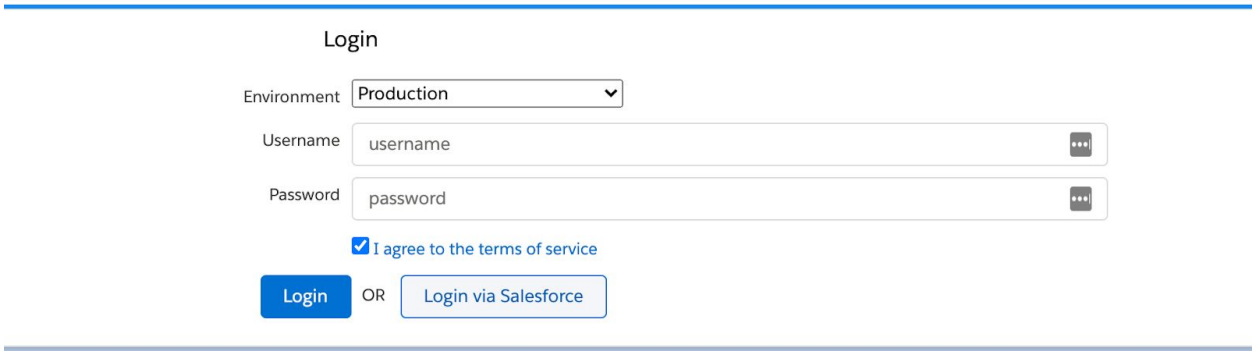
Note: You may need to include password + Security token if your org has security token enabled



Ensure to agree to terms and conditions

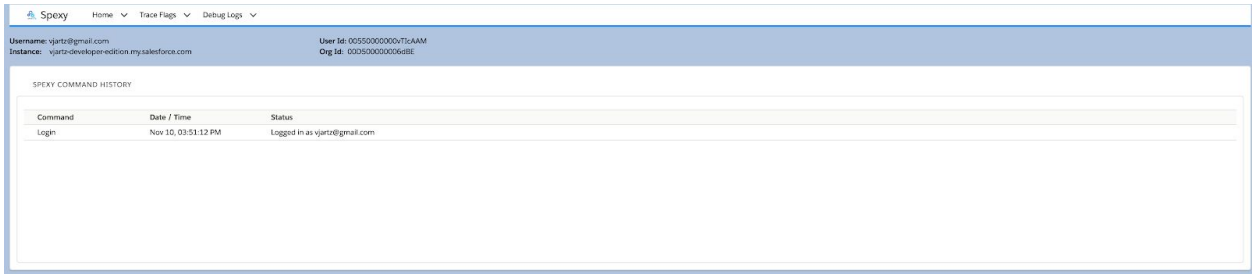
Login via salesforce authentication:

- If you would like to use salesforce to authenticate using connected app
  - Leave the username and password blank
  - Agree to terms and conditions
  - Click on Login via Salesforce



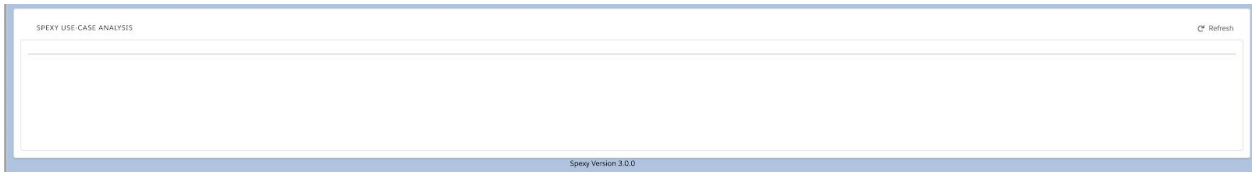
Using Spexy

Command History: This where all commands issues will be displayed



Use Case Analysis section:

This is where you will be able to see the use cases once analysis is completed by Spexy

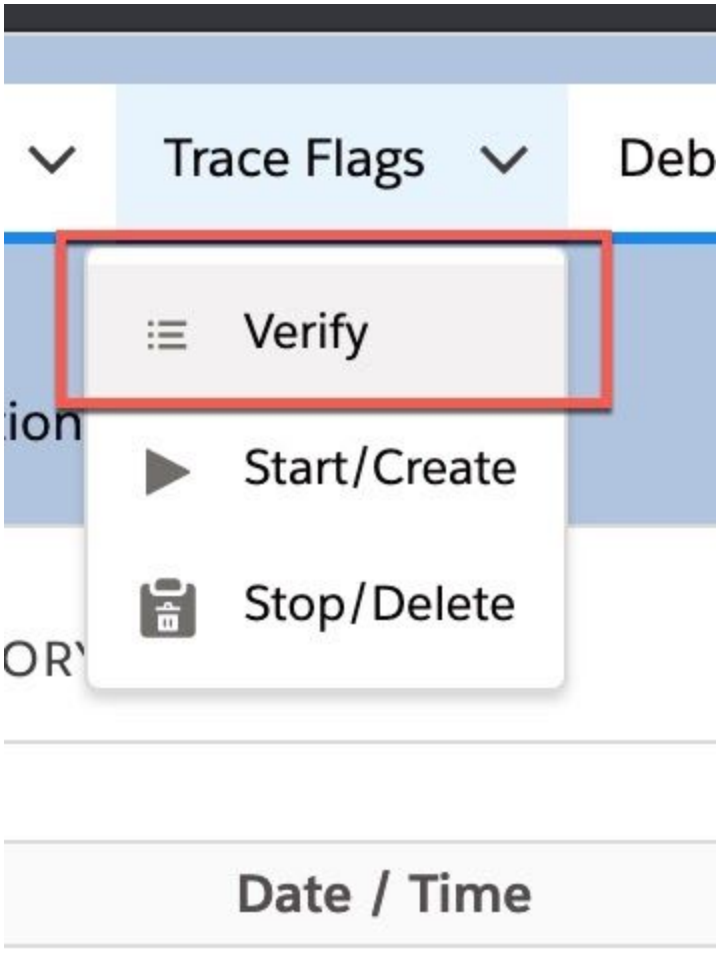


Analyzing a use case

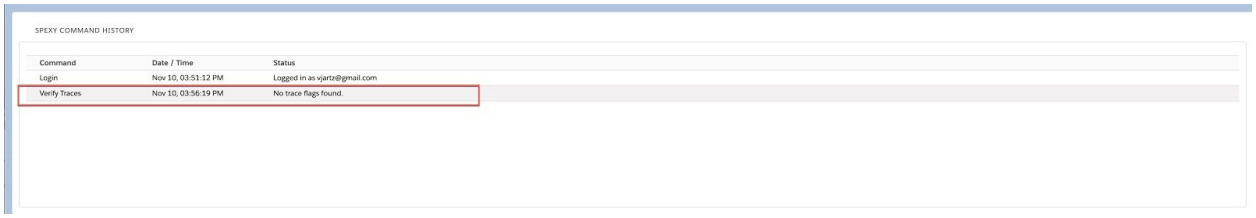
Step1

Verity that Trace flags are set up for the user you would like to trace

In Order to very go to Trace Flags Click on verify



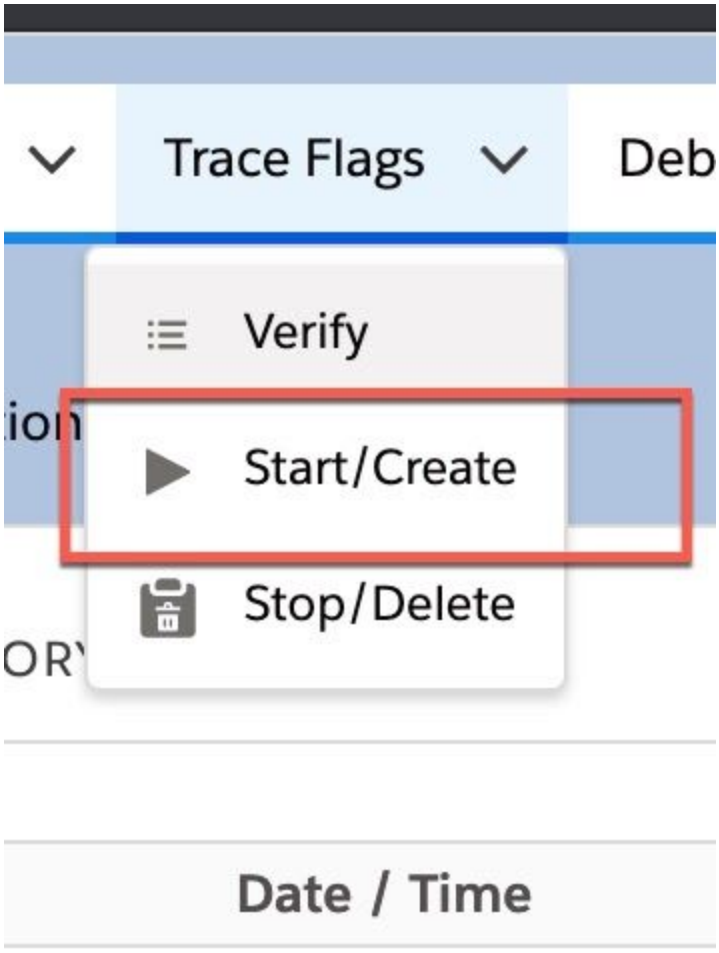
Command history will show if the flags are setup or no:



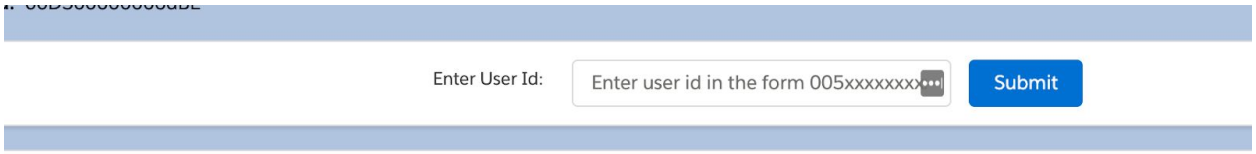
Setting up trace flags:

Click on Trace Flags:

Click on Start /Create

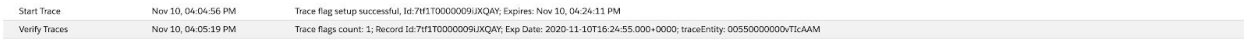


If you would like to trace a specific user please put in the userid for the user: eg 00550000002SaTN



If you want to trace as logged in user you can leave it blank and click submit

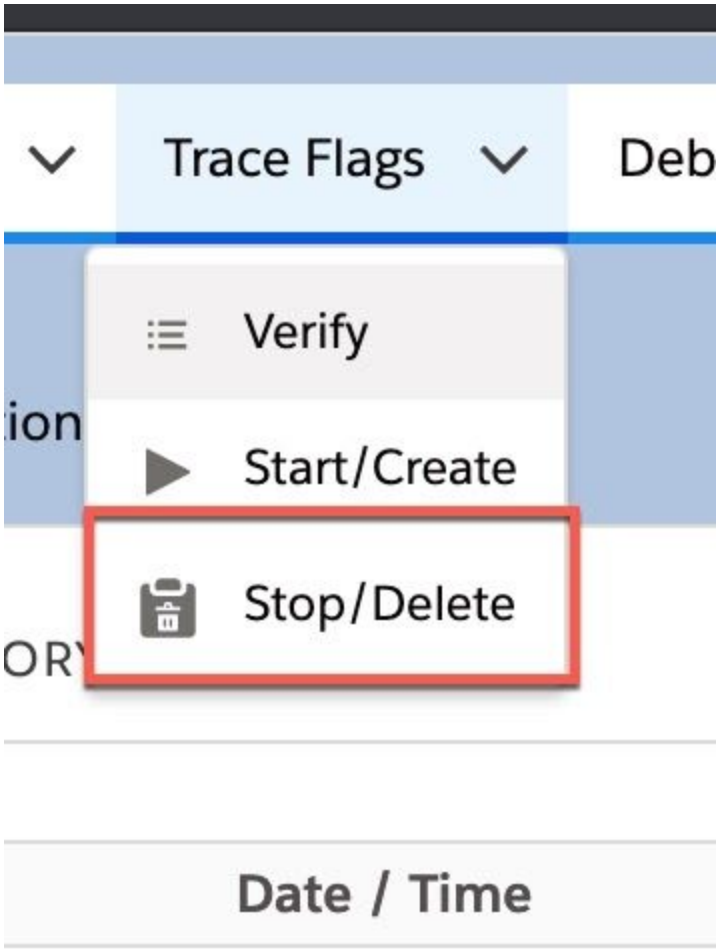
Click Verify Trace again by clicking Trace Flags → Verify:



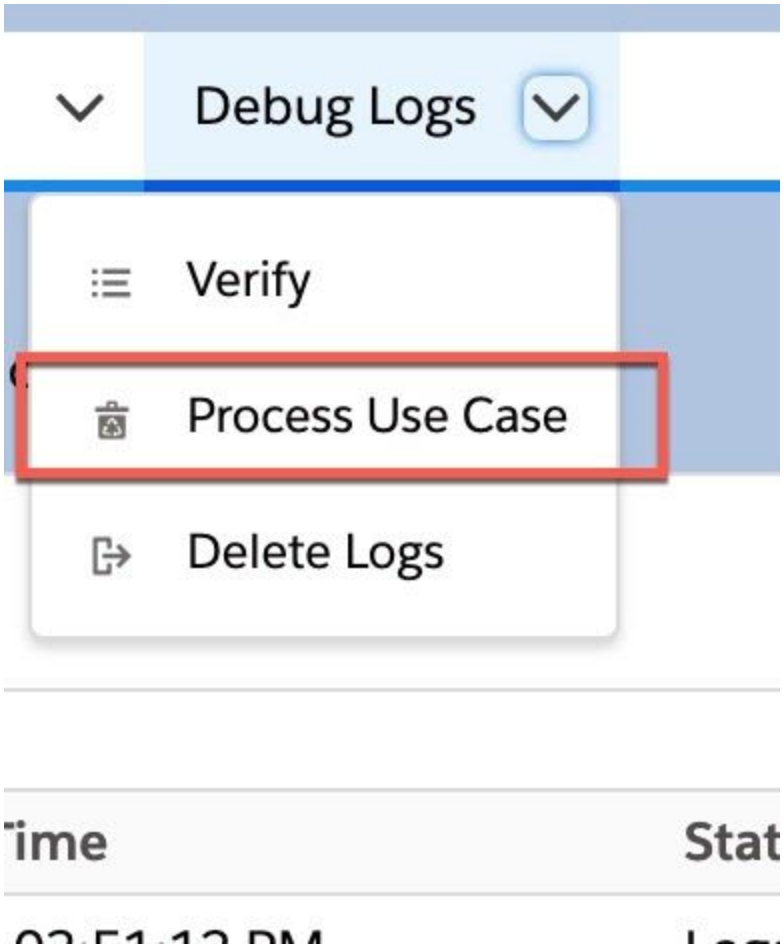
Now login to the org and perform the steps of the scenario you would like to trace

Once you have completed performing the steps for the scenario

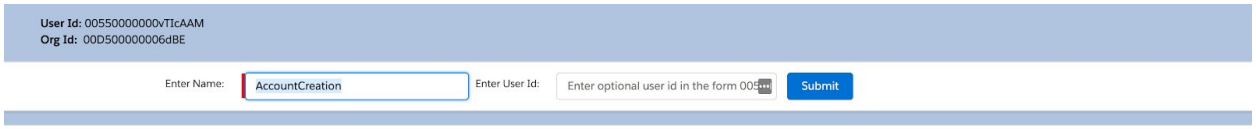
Click on Trace flags-- Stop:



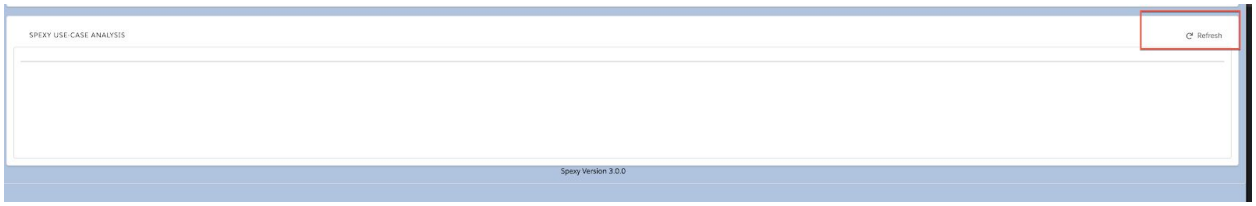
To process the use case click on Debug logs-- Process use case



Give your use case a name and User id you traced as(this is optional if it is logged in user please leave blank)



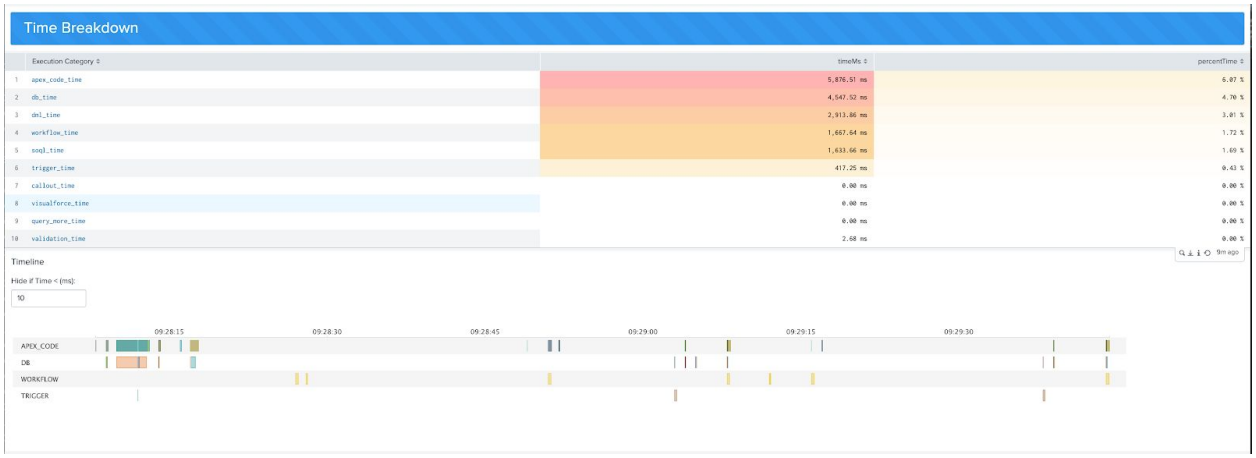
Now click the refresh button on the usecase analysis section:



Analyzing in splunk views:

Tree view:
This provides the entire stack tree so you can visualize how the various methods are being called:

Spexy- Timeline



Spexy - Call Hierarchy (Call Tree)

Introduction

WHAT DOES IT DO?

It displays the call tree for all calls and operations (Method, DML, SOQL, Callout, etc.) with each child function displaying time taken and a percentage of the time taken from the top level calling function (the function shown at the root).

WHAT CAN IT DO?

With this visualization, one can essentially see and “walk through” from a top level to identify the root cause of performance bottlenecks. A developer can drill down from the top level calling function of

choice and walk through all the child functions and calls to identify the specific performance culprits. It is similar to the Java call hierarchy tool in Java IDE's.

Filters

As inputs, this section takes in Method ID, Depth, Time, Events, Line/Page to aid in readability and visualization of APEX call trees.

Method Id
: The method ID's for the top level element. Default is none, you must select a method or code unit in a drill down in order to display a call hierarchy. This field is automatically changed upon selecting a specific method or code unit in their respective drill downs.

Depth
: The maximum desired depth to be displayed for the call hierarchy.

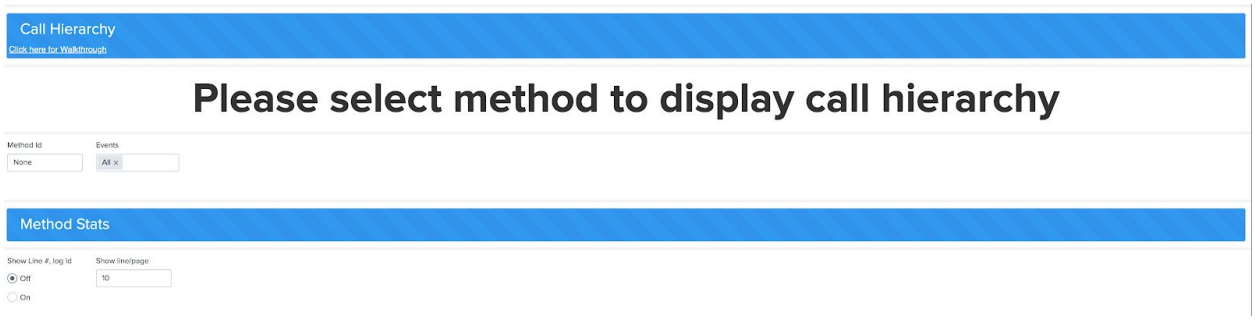
Time
: The minimum time a function must have spent in order to be shown in the visualization.

Percent Time
: The minimum percentage of time a function must have spent in order to be shown in the visualization.

Events
: The type of events (Method, DML, SOQL, Callout, etc.) to be shown.

Show Fields
: Toggle between displaying additional fields or just the time and function name

Export
: Allows you to export the current visualization as an SVG (can then be easily converted to PDF, png, etc.)



Usage

The call hierarchy will not be displayed until a code unit/method/etc is selected. By default, it will display the call tree starting from the first method/code unit called as the root and all of its immediate children functions. Clicking upon any of the nodes will expand them to reveal its children functions. The header above will display the name of the top level function.

Nodes that have children and are expandable are outlined in blue.

FLOW

Identify Method/Code Unit of interest (In stats sections)
Click to drill down
Click on row for the specific call in drill down
View call hierarchy for the specific call
Pinpoint child functions as root cause

After identifying a method or code unit that could be a potential performance culprit in the method or code unit stats section, the user can drill down to see specific information on that method or code unit. In the drill down, the user can click on the specific method or code unit of interest to render the call hierarchy for that particular code unit/method. Finally, the user can view the culprit's child calls to see which lower level function is taking the most time and then go about making the appropriate optimizations.

Help

Docker commands

Quick docker commands
To stop the service:
docker-compose down

To get the latest build ensure to shut down the service before executing this command
docker-compose pull

To start the service

docker-compose up -d

Post questions here