Splunk4Ninjas - Dashboard Studio

Lab Guide

Overview

This lab guide contains the hands-on exercises for the Splunk4Ninjas - Dashboard Studio workshop. Before proceeding with these exercises, please ensure that you have a copy of the workshop slide deck, which will help to put into context the tasks you are carrying out.

Download the workshop slide deck: https://splk.it/S4NDS-Attendee

Prerequisites

In order to complete these exercises, you will need your own Splunk instance. Splunk's hands-on workshops are delivered via the Splunk Show portal and you will need a splunk.com account in order to access this.

If you don't already have a Splunk.com account, please create one here before proceeding with the rest of the workshop.

Prior to beginning this workshop, you should have either:

- Taken *Splunk4Rookies - Dashboard Studio*, or
- Have a basic understanding of - or have built a dashboard in - Dashboard Studio

You should already understand:

- How to navigate the editing UI and high level understanding of the source code
- How to add visualizations, searches, and inputs

⚠️ Troubleshooting Connectivity

If you experience connectivity issues with accessing either your workshop environment or the event page, please try the following troubleshooting steps. If you still experience issues please reach out to the team running your workshop.

- **Use Google Chrome** (if you're not already)
- If the event page (i.e. *https://show.splunk.com/event/<eventID>*) didn't load when you clicked on the link, try **refreshing the page**
- **Disconnect from VPN** (if you're using one)
- **Clear your browser cache and restart your browser** (if using Google Chrome, go to: Settings > Privacy and security > Clear browsing data)
- **Try using private browsing mode** (e.g. Incognito in Google Chrome) to rule out any cache issues
- **Try using another computer** such as your personal computer - all you need is a web browser! Cloud platforms like AWS can often be blocked on corporate laptops.

Table of Contents

Exercise 1 – Set up your dashboard

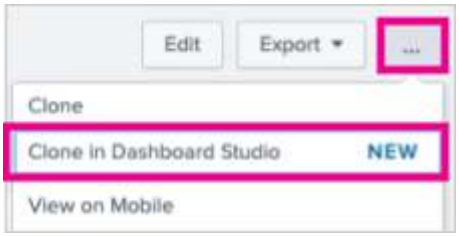
Description

Convert a Classic dashboard.

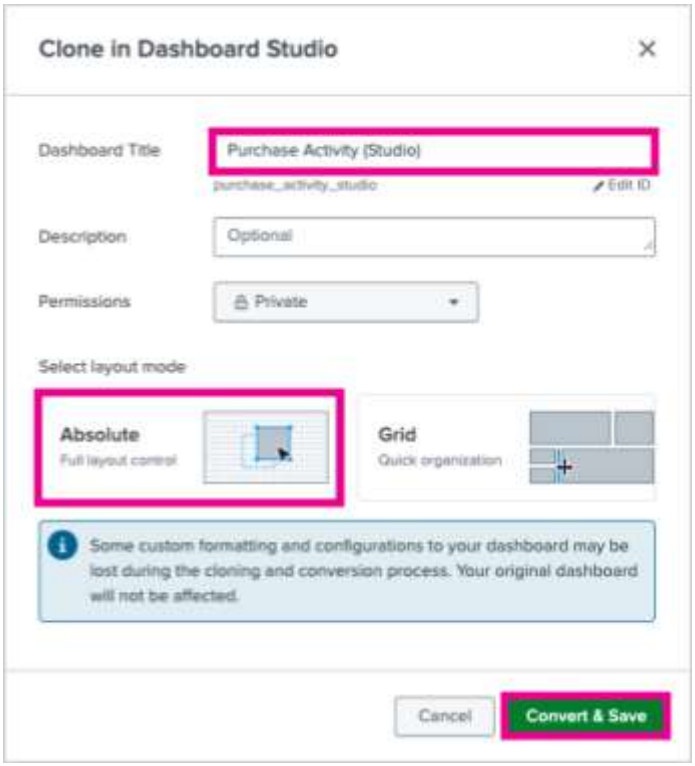
Steps

Convert a Classic dashboard

1. Log in to your Splunk environment using the **participant_x** user credentials provided, ensuring you replace ‘x’ with your unique User ID from Splunk Show. (e.g. If your User ID was ‘5’ then your username would be ‘participant_5’).
2. From the Dashboards tab, open the **Purchase Activity (Classic) dashboard**
3. In the top right corner, under the three dots menu, select **Clone in Dashboard Studio**



4. Provide a new title for your dashboard, such as “*Purchase Activity (Studio)*”
    a. (Optional) Specify a description
    b. Leave the Permissions as ‘Private’
5. Select **Absolute** layout
6. Select **Convert & Save**



Note: Visualizations that extend beyond the canvas borders will not display in View mode. We will be rearranging these visualizations to all fit with the canvas by the end of this workshop.

Exercise 2 – Configure product purchases single values

Description

Configure the 4 product purchases single values to have space for images (which we will add later) and to only take up the left two thirds of the canvas.

Steps

Resize product single values

1. Multi-select the single values

   On Mac: **CMD + click**

   On Mac + Windows: **Click and drag** to highlight the objects you want to select
2. In the right side configuration panel, bulk update all four single values to:
   a. width: **140**
   b. height: **140**
3. Deselect the four single values and select just one of them

   Under **Color & Style**, change the **Background** color to **transparent**

Rearrange product single values

4. Add a rectangle:
   o. Set the **width to 250** and the **height to 140**
   p. Change the fill color to **white**
   q. Change the stroke color to **transparent**
5. Reposition the rectangle so that it's behind one of the single values with the single value on the left side of the rectangle, with empty space on the right
6. Use the layering icon to adjust which objects are in front or behind others

7. Clone the white rectangle three times, to place behind the other three single values

Reposition product single values

8. Reposition your single values (and their white backgrounds) to take up the left two thirds of the canvas, and to have room for a text label above

Add custom label and divider

9. Add a Markdown component by selecting the 'M' ( ) icon in the toolbar

   In the right side configuration panel, add the following Markdown syntax:

   # Product sales
   Electronics sold by count and region

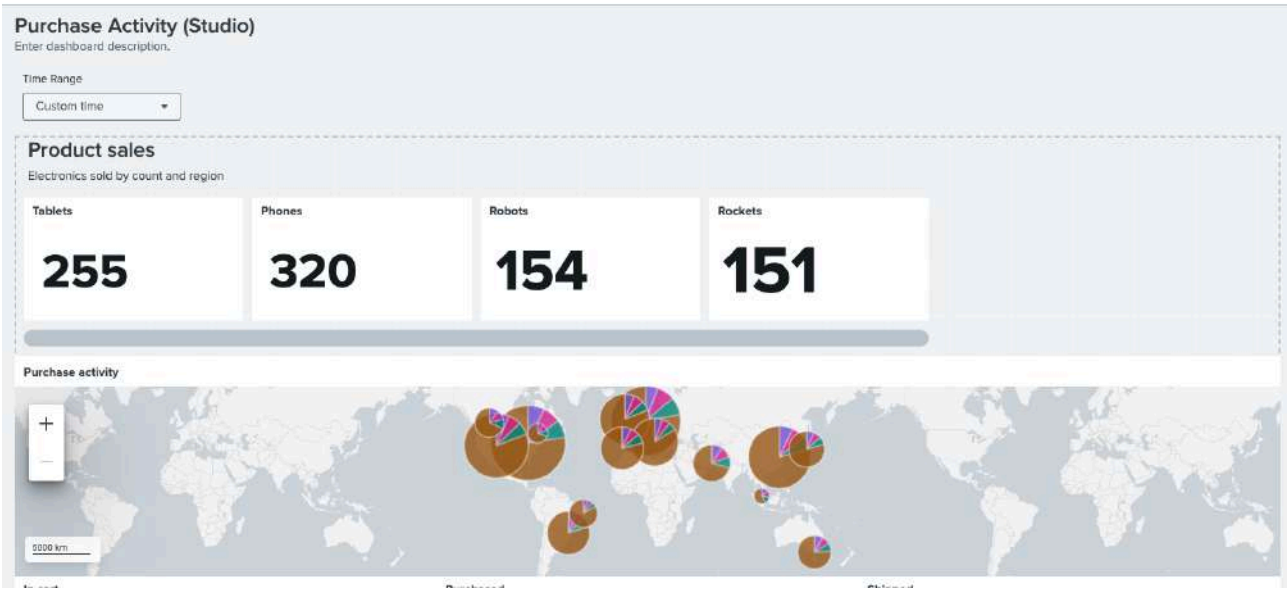
   Position the markdown above the single values. Note that the actual Markdown component won't update until you click outside of the configuration panel.

10. Add a rectangle to serve as a divider between the single values and map:

    o. Set the stroke color to **transparent**

    p. Set the radius to **10**

11. Position the rectangle to take up the space between the single values and the map. The width should be the same as the single values



Update the single values to use one search

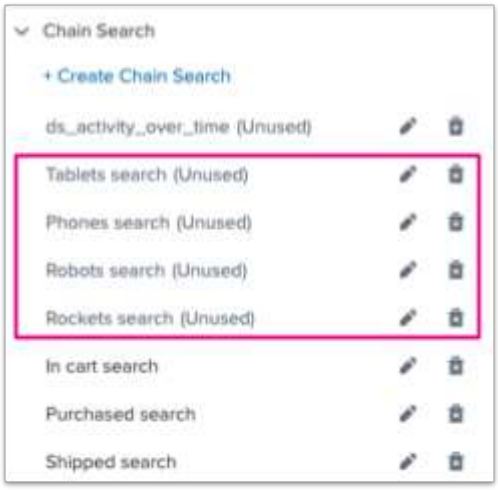
12. Select the Tablets single value and under **Data sources** click on the trash can icon ( ) next to "Tablets search"
13. Select **"+ Setup primary data source"** and select **ds_purchases_by_product**
14. Under **Selected data field,** update the field to be **Tablets**

    Repeat the steps above for the other single values - Phones, Robots and Rockets - so that all four product KPIs now use just the ds_purchases_by_products search.

Delete the unused chain searches

15. Select the Data source icon the toolbar:
16. Under **Chain Search**, notice that the Tablets search, Phones search, Robots search, and Rockets search all say "(Unused)"

    Clean up your dashboard by deleting these four chain searches. To do this click on the trash can icon ( ) next to each chain search.



Exercise 3 – Customize the map and add a link graph

Description

Learn how to customize the map, and how to configure a link graph to update based on what's selected in the map.

Steps

Resize and reposition the map

1. Resize the width to match the width of the divider. Resize the height to fill roughly half of the remaining vertical space (~300 px).
2. Move the map down to make room for a custom text label. You can move the other visualizations out of the way, off canvas for now - we'll add them back in later.
3. Add a Markdown component by selecting the ‘M’ ( ) icon in the toolbar. In the right side configuration panel, add the following Markdown syntax:

   # Purchase activity
   Breakdown by location. Click on a map area to investigate activity below.

   Position the markdown above the map
4. Remove the "Purchase activity" title on the map visualization

5. Under **Data configurations**, under **Bubble Size,** deselect all fields except **products_purchased (number)**

6. Under **Data configurations,** under **Additional tooltip fields,** select **credit_cards (number)** and **customers (number)**

Add a custom tile server

7. Under **Color & style**,

    a. Set **Base layer tile server** to https://tile.openstreetmap.org/{z}/{x}/{y}.png

    b. Set **Base layer tile server type** to Raster

    Your map should now look like this in View mode:

Set tokens on click

8. In Edit mode, select the map. Under **Interactions** click on **+ Add Interaction** and on the dropdown select **Set Tokens**

9. Specify the following 4 predefined tokens:

    a. Token name: **mapBoundsEast**
    Token value: **row._geo_bounds_east.value**

b. Token name: **mapBoundsWest**
Token value: **row._geo_bounds_west.value**

c. Token name: **mapBoundsSouth**
Token value: r**ow._geo_bounds_south.value**

d. Token name: **mapBoundsNorth**
Token value: **row._geo_bounds_north.value**

Alternatively, copy and paste this directly into the source code, at the same level as "options":

```
"eventHandlers": [
        {
                "type": "drilldown.setToken",
                "options": {
                        "tokens": [
                                {
                                        "token": "mapBoundsNorth",
                                        "key": "row._geo_bounds_north.value"
                                },
                                {
                                        "token": "mapBoundsEast",
                                        "key": "row._geo_bounds_east.value"
                                },
                                {
                                        "token": "mapBoundsSouth",
                                        "key": "row._geo_bounds_south.value"
                                },
                                {
                                        "token": "mapBoundsWest",
                                        "key": "row._geo_bounds_west.value"
                                }
                        ]
                }
        }
],
```

Add a link graph

10. Click on the Add Chart icon ( ) in the menu bar and select to add a **Link Graph**. Position and size the link graph it to fill the space beneath the map

11. Create a new search for your link graph:

    a. Name: **Activity in a highlighted area**

    b. SPL:

```
index=purchases status=purchased
| iplocation ip_address
| where (lat>=$mapBoundsSouth$ and lat<=$mapBoundsNorth$)
  and (lon>=$mapBoundsWest$ and lon<=$mapBoundsEast$)
| table customer_id ip_address credit_card
  billing_address status
```

    c. Click on **Apply & Close**

12. Under **Visibility**, check the "**When data is unavailable, hide element**" box

13. Click on **Save** and go to View mode for your dashboard

Notice that your link graph is not displayed because the tokens needed for its search are not set. Click on a bubble on the map to see the link graph appear.

Exercise 4 – Set up the conversion funnel

Description

Set up a conversion funnel to see an overview and detailed view of orders at each stage.

Steps

Create a section for the conversion funnel

1. Add a rectangle to fill the empty space on the right side of the dashboard canvas
2. Set the stroke color to **transparent**
3. In the right side configuration panel, add the following Markdown syntax:

    a. **# Order pipeline**
    Order flow from online store to customer

Place the order status KPIs

4. Towards the top of the conversion funnel, place the "**In cart**" single value KPI. Leave space between the markdown and the single value for a dropdown input that we will add later.

    In the middle, place the "**Purchased**" single value KPI

At the bottom, place the "**Shipped**" single value KPI

Note: These are the 3 single values that were originally at the bottom of the dashboard

5. Add a line shape, and position it vertically between "In cart" and "Purchased"

   In the right side configuration panel set the following:

   a. Under **Stroke,** increase **thickness** to **3**

   b. Under **Arrows,** select **Ending Point**

      i. You may need to select **Starting Point** instead, depending on how your arrow is oriented.

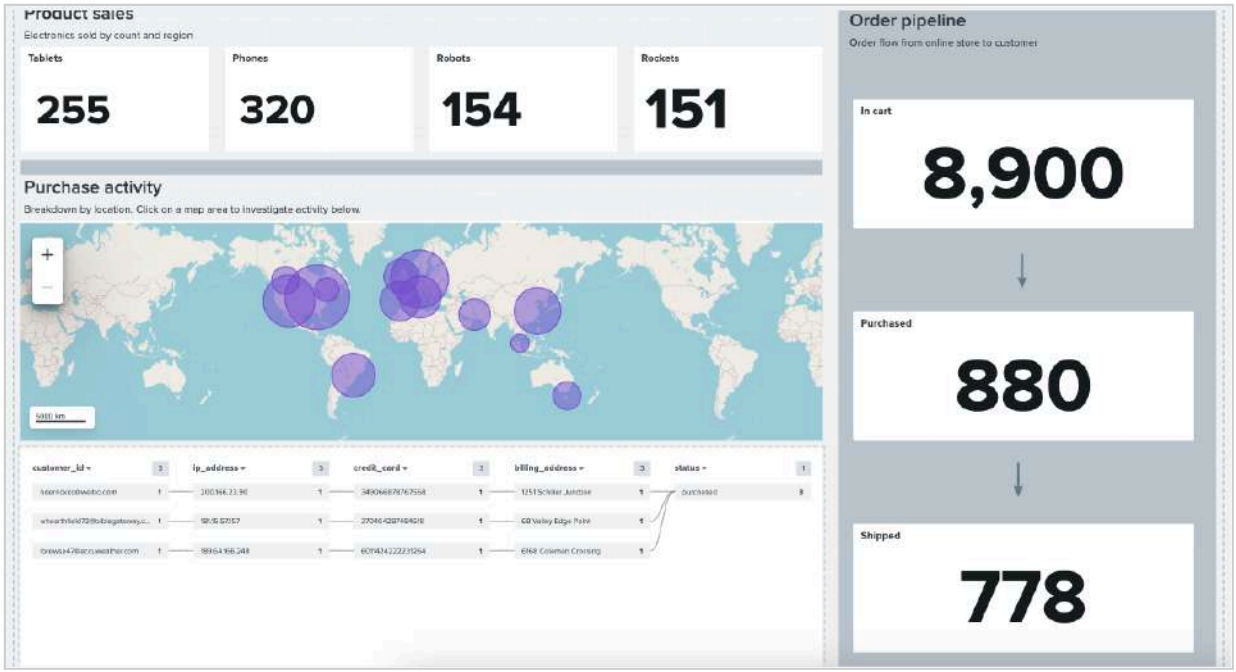
   c. Under **Coloring,** select **Color Value** to be **gray**

Set up an alternative "details" view

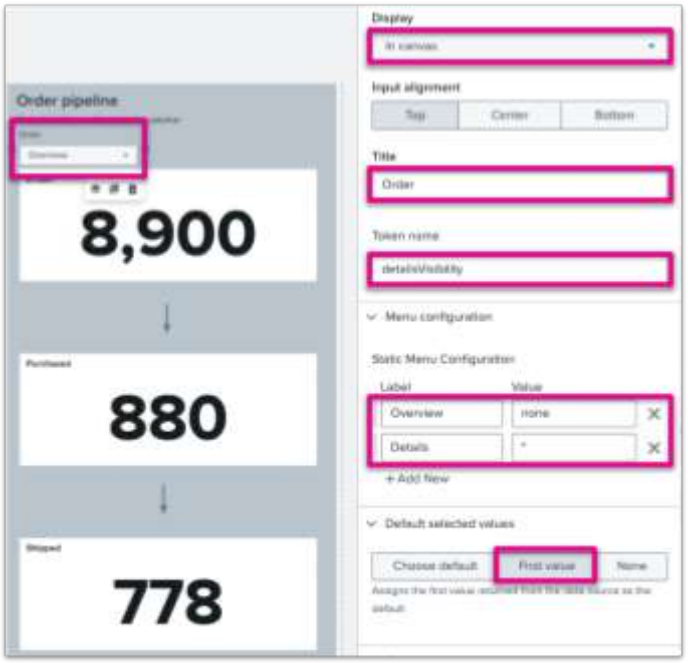
6. Copy and paste the arrow:

   a. Use keyboard shortcuts **CMD+C** → **CMD+V** or **CTRL+C** → **CTRL+V** to duplicate the arrow

   b. Place the duplicated arrow in between the "Purchased" and "Shipped" KPIs



Set up an alternative "Details" view



7. Add a dropdown input
8. In the configuration panel, under **Display,** select "**In canvas**"
9. Position the dropdown between the Markdown title and the "In cart" KPI
    a. Name: **Order**
    b. Token name: **detailsVisibility**
    c. Menu item 1:
        i. Label: **Overview**
        ii. Value: **none**
    d. Menu item 2:
        i. Label: **Details**
        ii. Value: *****
    e. Default selected values: **First value**
10. Place the "Purchased orders details" table over the "Purchased" single value
11. Place the "Shipped orders" chart over the "Shipped" single value
12. Add **`$detailsVisibility$`** to the first pipe of each search:
    a. The "**Purchased orders details**" table search should look like this:
    `index=purchases $detailsVisibility$`
    b. The "**Shipped orders**" table search should look like this:
    `index=purchases status=shipped $detailsVisibility$`
13. For both tables, select the table and under **Visibility** check "**When no data is available, hide element**"

    The result should be "No search results returned". This is because the Orders dropdown is set to "Overview" which means $detailsVisibility$ is set to "none".

    Select **Save** and go to View mode for your dashboard to select between Orders: Overview and Orders: Details to see the "Purchased orders details" table and "Shipped orders" line chart reappear and disappear.

Set up an alternative "details" view

14. Navigate to the Data source overview panel. Select the pencil icon next to **ds_order_pipeline** and check the box that says "**Access search results or metadata**"

    Click on **Apply & Close**

15. Select the "Purchased orders details" table and update the title to **$ds_order_pipeline:result.purchased$ purchased**

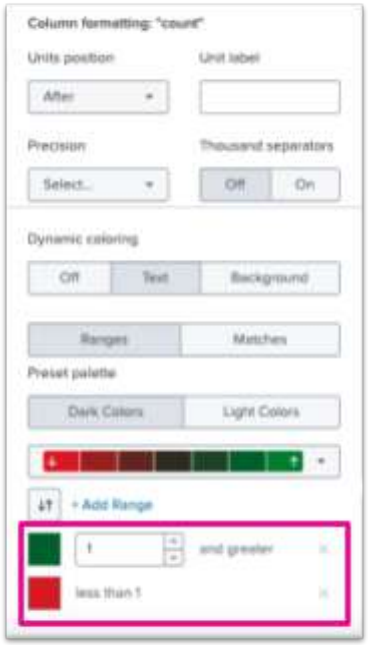
16. Select the "Shipped orders" line chart and update the title to **$ds_order_pipeline:result.shipped$ shipped**

Exercise 5 – Color table columns by hidden values

Description

Color table columns based on values in hidden columns using Dynamic Options Syntax (DOS). You do not need to create a new table, you will use the table that is already in your dashboard from the conversion.

Steps



Apply coloring to the 'count' column

1. Select the "Purchased orders details" table, navigate to **Color and style,** and under **Column-specific formatting,** add the "count" column
2. Select the pencil icon next to "count - number" and under **Dynamic coloring**, select "text"
3. Configure two ranges:
    a. **Less than 1** is **red**
    b. **1 or greater** is **green**

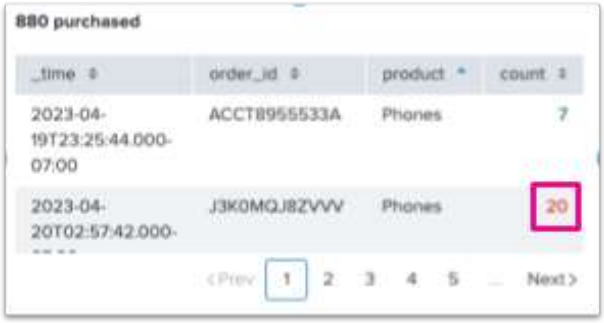
Update the DOS to use _has_stock

4. Open the **Code** section at the bottom of the configuration panel or find the table's source code in the full source code editor

    In source, under "options" > "columnFormat" > "count" > "rowColors", replace `seriesByName(\"count\")` with `seriesByName(\"_has_stock\")`

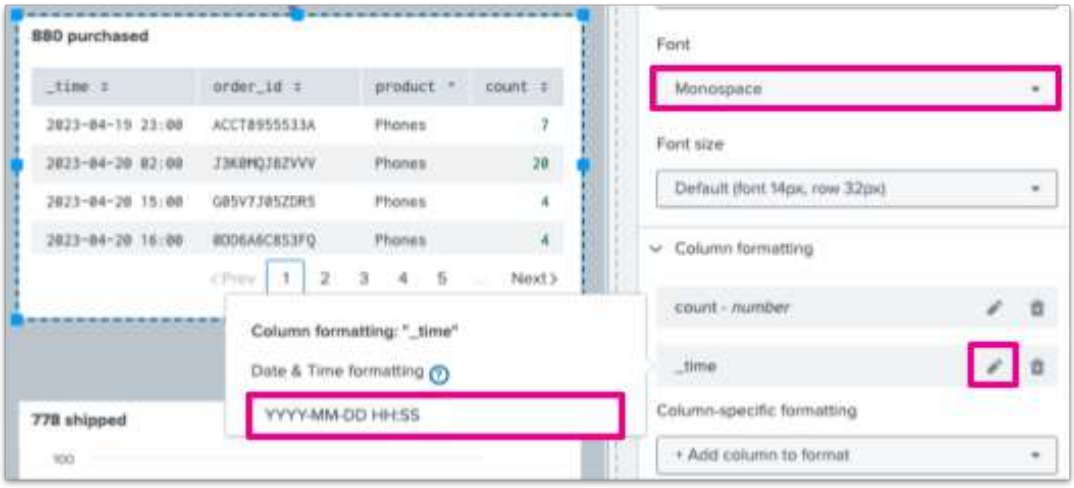
```
"options": {
    "columnFormat": {
        "count": {
            "data": "> table | seriesByName(\"count\") | formatByType(countColumnFormatEditorConfig)",
            "rowColors": "> table | seriesByName(\"_has_stock\") | rangeValue(countRowColorsEditorConfig)"
        },
```

    Your table should now have values in count that are red:



Finish formatting the table

5. Under **Color and style** change **Font** to "Monospace"
6. Under **Column-specific formatting**, select _time and apply the format: **YYYY-MM-DD HH:SS**



Note: As a best practice, configure as much in the UI as possible first before modifying the source code.

Exercise 6 – Add secondary data sources to a line chart

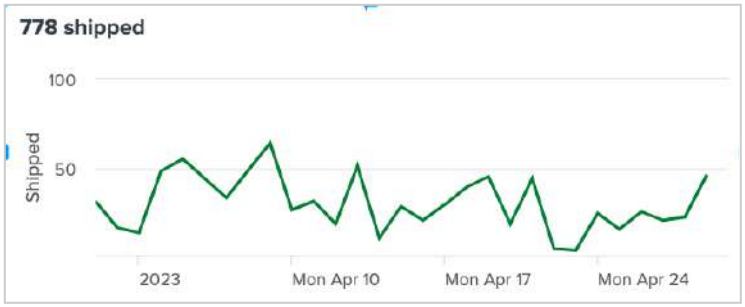
Description

Add annotations and overlay target thresholds to the "Shipped orders" line chart using Dynamic Options Syntax (DOS)

Steps

Beautify the "Shipped orders" line chart

1. Under **Data display**, change Null value display to "**Connect**"
2. Under **Color and style**, add a series to **Series color by field name**. Select the field name **count** and apply the color **#118832**
3. Under **Legend** change **Legend display** to "**Off**"
4. Under **X-axis grid and labels**, deselect **Axis title**, and change **Number of time label parts** to 1
5. Under **Y-axis grid and labels**, specify **Axis label text** as "**Shipped**"



Add an annotation data source

6. Select the line chart. Under **Data sources**, select **+ Set up Annotation Data Source**
7. Create a new chain search:
    a. Name: **Order events over time**
    b. Parent search: **ds_orders_over_time**
    c. Chain SPL:

```
| where count>60 | eval label="Big sale"
```

Click on **Apply & close**

Configure the chart annotation

8. Under **Data configurations,** select **Annotation x** to be "**_time (time)**"
9. Under **Data configurations,** select **Annotation labels** to be "**label (string)**"

Now in **View** mode, you can hover on the annotation to see the label

Create a search for "target" orders

10. Navigate to the **Data Overview**
11. Create a new chain search:
    a. Name: **Orders target**
    b. Parent search: **ds_orders_over_time**
    c. Chain SPL:

```
| eval target=50
```

12. Open the code editor and copy the data source ID
13. Click on **Apply & Close**

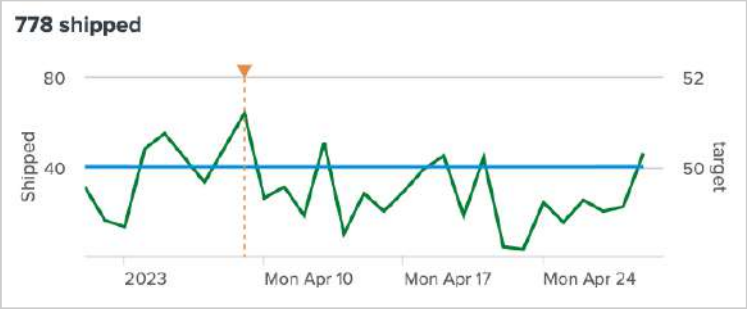
Use DOS to add a secondary data source

14. Select the line chart and open the code editor
15. Under **"dataSources"** add the ID you just copied from the last step:

```
"secondary": "ds_<your ID>"
```

16. Under **"options"** add:

```
"y2": "> secondary | frameBySeriesNames('target')",
```



Note: Learn more about how to write and modify DOS by visiting our documentation.

Exercise 7 – Color map bubbles

Description

Color map bubbles based on anomalous behavior using Dynamic Options Syntax (DOS)

Steps

Search for abnormal activity

1. Update the map data source to create a new field to designate abnormal activity by appending:

```
| fillnull
| eval abnormal_activity=if(customers > credit_cards, 1, 0)
```

2. Click on **Apply & close**

3. Select the map and open the **Code** section in the configuration panel or find the map code in the full source code editor. Under **"context"** create a section called **thresholdConfig**:

```
"context": {
        "thresholdConfig": [
                {
                        "to": 1,
                        "value": "green"
                },
                {
                        "from": 1,
                        "value": "red"
                }
        ]
}
```

4. Under **"options"** under **"layers"** add the following:

```
"dataColors": "> primary | seriesByName('abnormal_activity') | rangeValue(thresholdConfig)",
```

Exercise 8 – Add custom SVGs

Description

Add custom SVGs to your dashboard, which dynamically respond to search results

Steps

Create the inventory data source

1. Create a new search that will return what % of product is left in inventory:
    a. Name**: Inventory totals**
    b. SPL:

```
| makeresults count=1
| eval inventory_tablets=400
| eval inventory_phones=400
| eval inventory_robots=250
| eval inventory_rockets=250
| eval current_tablets=
  (inventory_tablets-$ds_purchases_by_product:result.Tablets$)/inventory_tablets*100
| eval
current_phones=(inventory_phones-$ds_purchases_by_product:result.Phones$)/inventory_phones*100
| eval
current_robots=(inventory_robots-$ds_purchases_by_product:result.Robots$)/inventory_robots*100
| eval current_rockets=
  (inventory_rockets-$ds_purchases_by_product:result.Rockets$)/inventory_rockets*100
| table _time current_tablets current_phones current_robots current_rockets
```

    c. Check the box that says **"Access search results or metadata"**

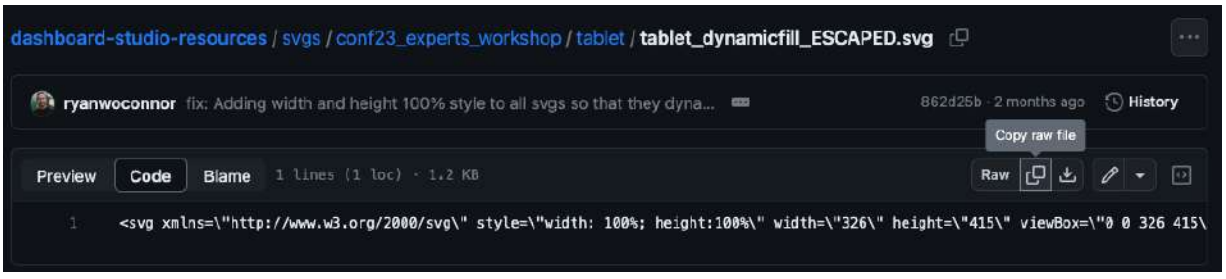
    Click on **Apply & Close**

2. Edit the **ds_purchases_by_product** data source and check the **"Access search results or metadata"** box

    Click on **Apply & Close**

Copy the SVG definition

3. Copy the raw file for the Tablet SVG here



Add an SVG to your dashboard

4. Add a **Choropleth SVG** and place it next to a single value KPI
5. When prompted to select a data source, select "**Cancel**"
6. Open the Code editor and under **"options"**, add:

   `"svg": "<copied file>”`

7. **Repeat these steps** for the remaining products:
   a. Github links: Phone, Robot and Rocket

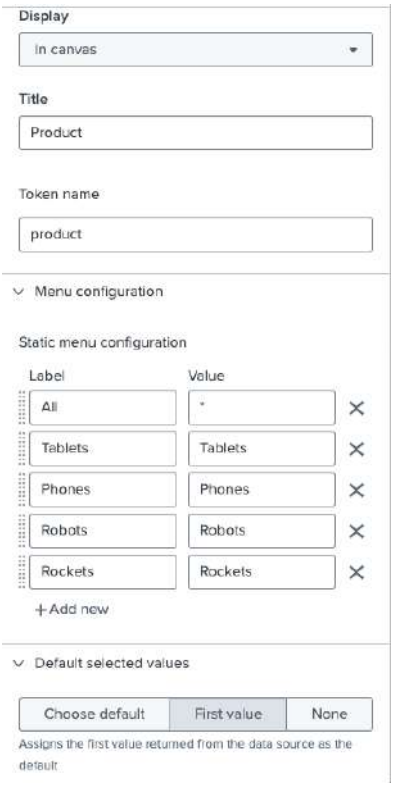
Note: You can find many resources and free tools online for creating custom SVGs.

Exercise 9 – Configure additional interactions

Description

Configure a "Set token on click" interaction and Input to share the same token. Set up multiple interactions on one visualization.

Steps



Filter the map by product

1. Add a dropdown input
2. Under **Display**, select **In canvas**
3. Reposition and resize the dropdown to be next to the Purchase activity custom map label
    a. Title: **Product**
    b. Token name: **product**
    c. Menu items - Label (value)
        i. **All (*)**
        ii. **Tablets**
        iii. **Phones**
        iv. **Robots**
        v. **Rockets**
    d. Default selected values: **First value**
4. Select the map
5. Edit the **Purchase activity search**
6. Add **product=$product$** to the first pipe
7. Select **Apply & close**
8. Select different products from the dropdown to see the map update

Configure interactions to set the product filter

1. Select the **Tablets** single value
2. Under Interactions, select **+ Add interaction**
3. Under **On click**, select **Set tokens**
    a. Token name: **product**
    b. Token value: **name**
4. Select **Apply**

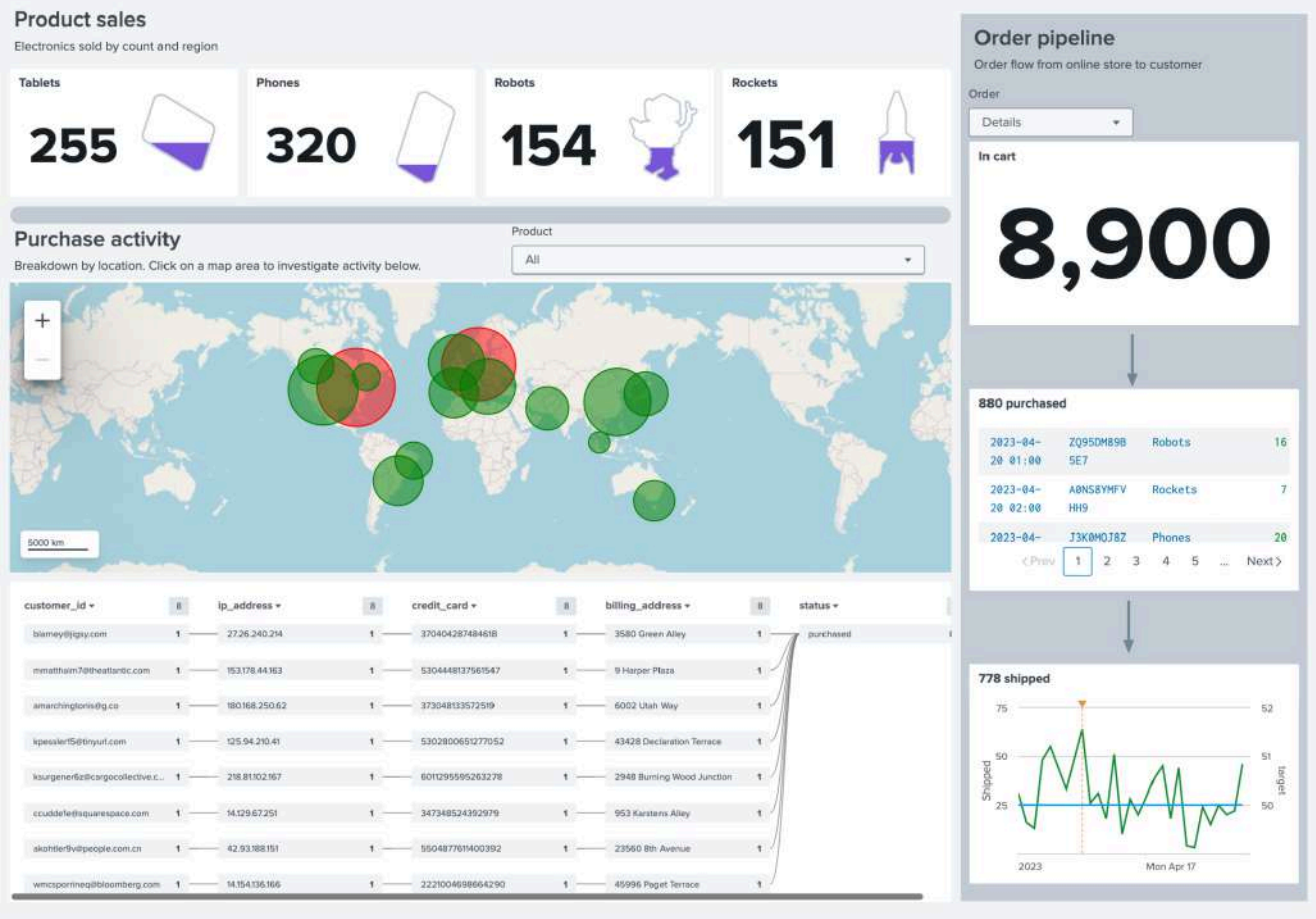
5. Repeat for Phones, Robots, and Rockets
6. Save your dashboard and go to View mode. Select one of the product KPIs and notice the Product dropdown and map update.

Note: There is a known bug where some charts without drilldowns in Classic are converting with "Link to Search" interactions. You can delete these interactions.

Set up additional interactions

1. Back in Edit mode, select the **Tablets KPIs**
2. Under **Interactions**, select **+ Add interaction**
3. Under **On click**, select **Link to custom URL**
    a. **https://www.google.com/search?q=$product$**
    b. Check the box to **Open in a new tab**
4. Select **Apply**
5. Save your dashboard and go to View mode. Select the Tablets KPI to see the Product dropdown and map update, and a redirection modal prompting navigation to Google.

Your dashboard should now look like this!



Exercise 10 – Design tips for finish touches

Description

Learn best practices for designing a dashboard that effectively communicates insights.

Steps

1. Check that objects on your dashboard are aligned when possible
2. Check that objects are spaced evenly
3. Check for consistency:
    a. Use the same color palette when possible
    b. Use the same casing in your labels

Note: Check out our resources on Dashboard Design Best Practices and Visualization Choices and Configurations

---