Export excel in grafana

# 1. Purpose

Allowing Grafana to export to excel-readable format (.xlsx). The excel file will contain the data points of the current view graph. This will facilitate data manipulation in the future using excel features (bar charts, pie charts, pivot table, or VBA).

Furthermore, an additional feature is added so as to print all panels (within a dashboard) as PDF file for ease of monthly reporting.

# 2. Installation & Manual

Step 1: Download and install Grafana from Source Code ( <https://github.com/grafana/grafana> ). Follow the instruction to install all dependencies (Go 1.5, NodeJS V4+, Godep).

Grafana requires correct version of dependencies to be installed.

**\*Note:** PhantomJS installing from Grafana is currently faulty 🡺 solution install PhantomJS separately

Step 2: Replace the entire **github.com/grafana/grafana/public** folder with **public** folder (provided)

Step 3: Building from Grafana Source Code

**Building the Backend**

go run build.go build

**Building the Frontend**

npm install

npm run build

Only need to do this once before running Grafana

Step 4: Start Grafana server and use as per normal.

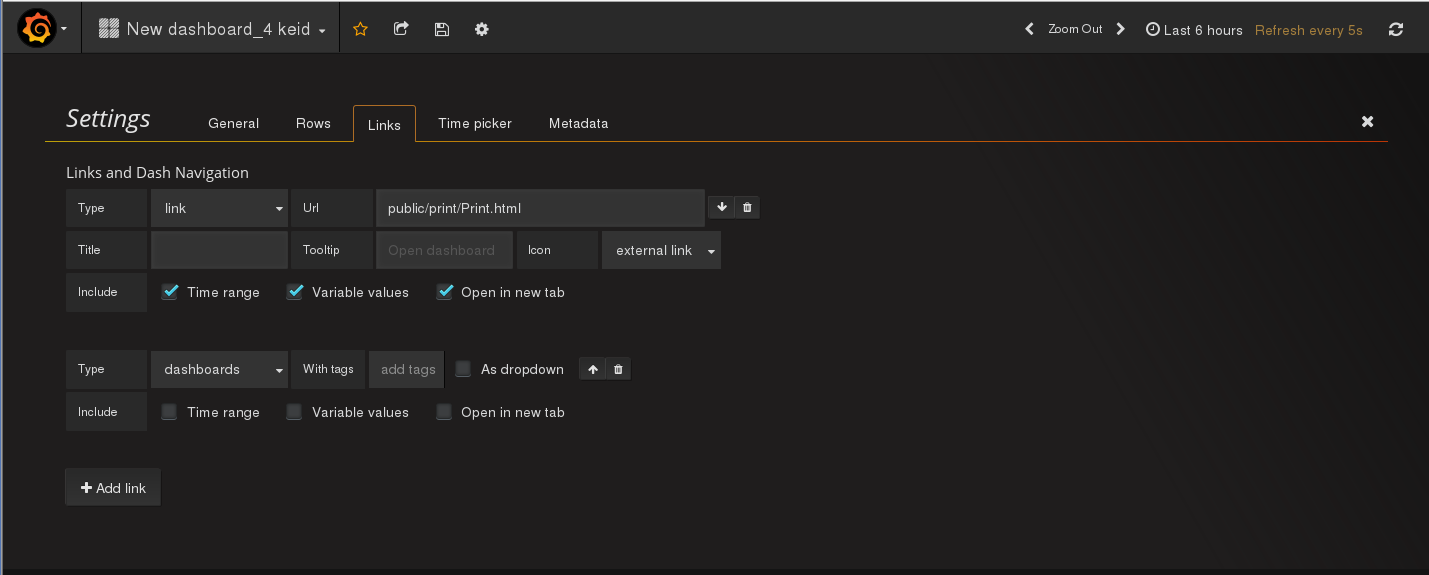
**Running**

./bin/grafana-server

\*Note: Grafana server will have to be started again after reboot

Step 5: To add the function Print To PDF. Login to Grafana 🡪 Select Dashboard 🡪 Setting 🡪 Links

* **Type:** Choose link
* **URL:** public/print/Print.html
* **Include:** Time range, Variable values (optional), Open in new tab
* **Title:** Print(Or other title that you want)

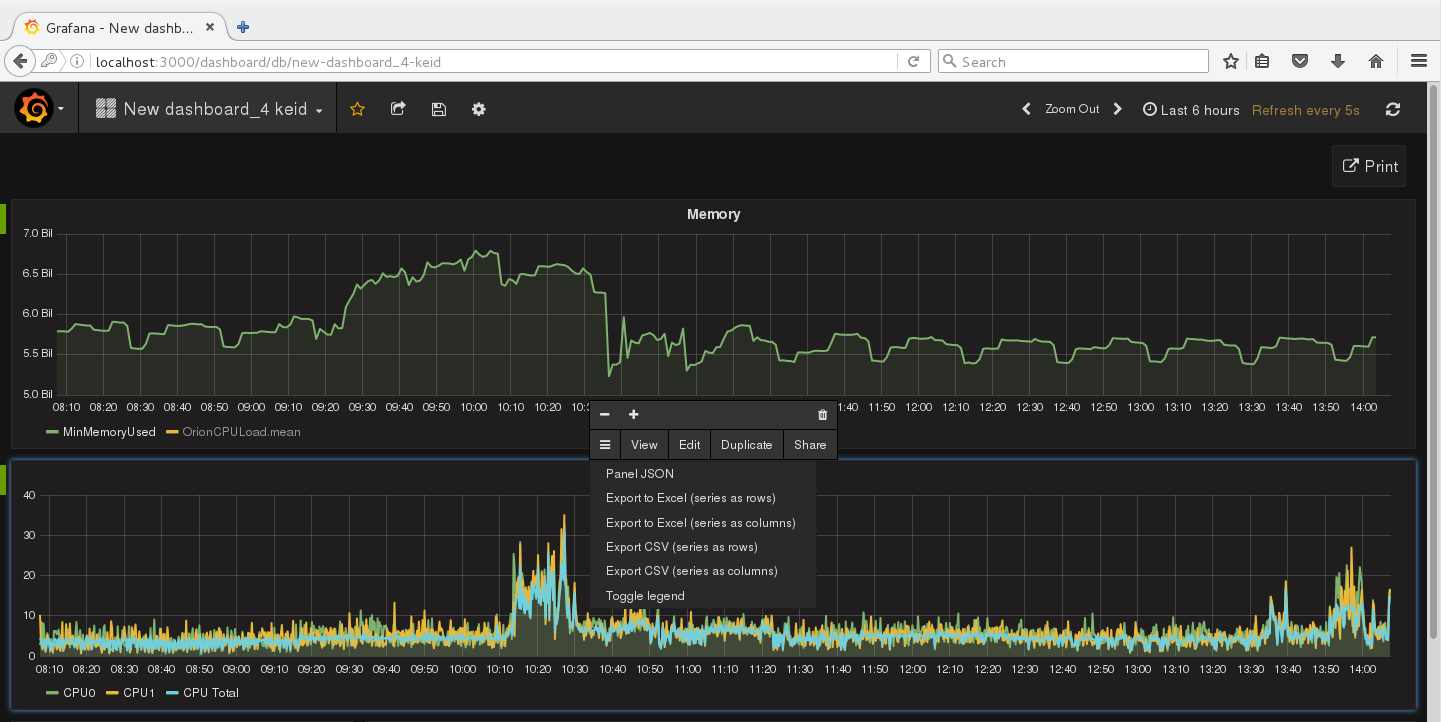


Step 6: Save Dashboard. Print button will appear on the right corner of the dashboard.



Step 7: Press Print button will open new tab with all panels view. Use Print function (Ctr+P) in Chrome and save as PDF

Step 8: To export data to Excel file. Navigate to a specific panel that you want to extract, dropdown button 🡪 choose Export to Excel (series as rows) or Export to Excel (series as column) to export



# 3. Source Code

1. Adding buttons to Grafana console

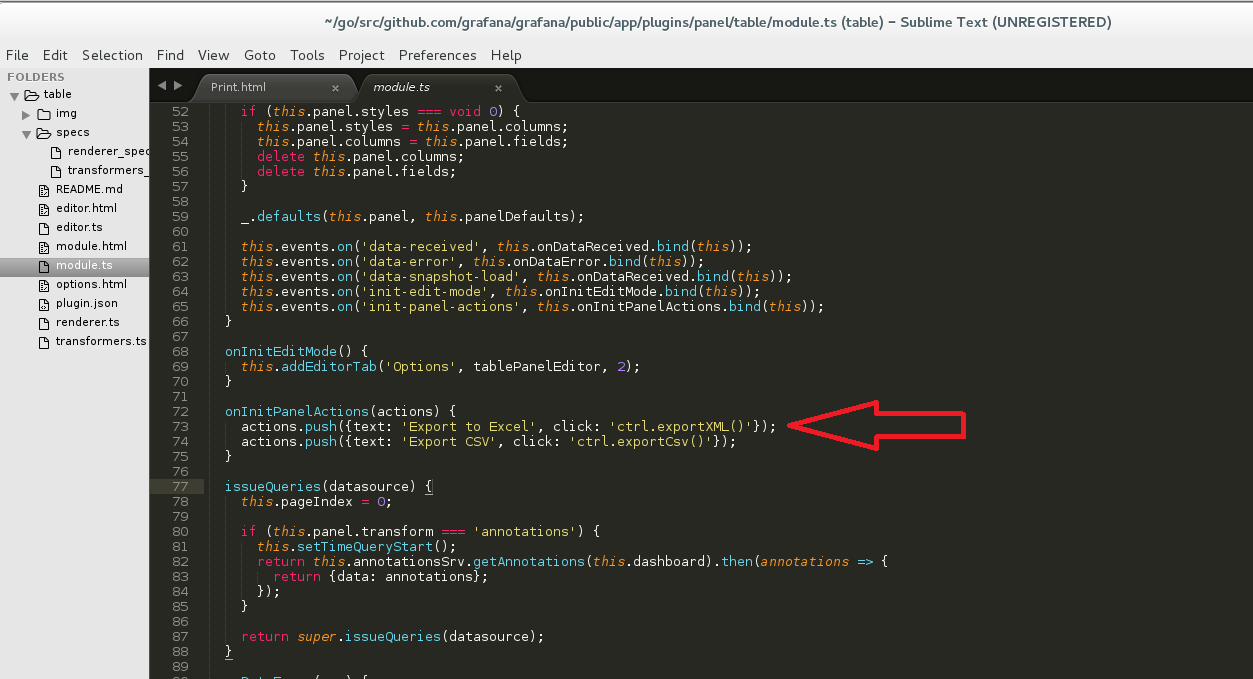
Depend on what type of panel, you need to update the button accordingly

For example:

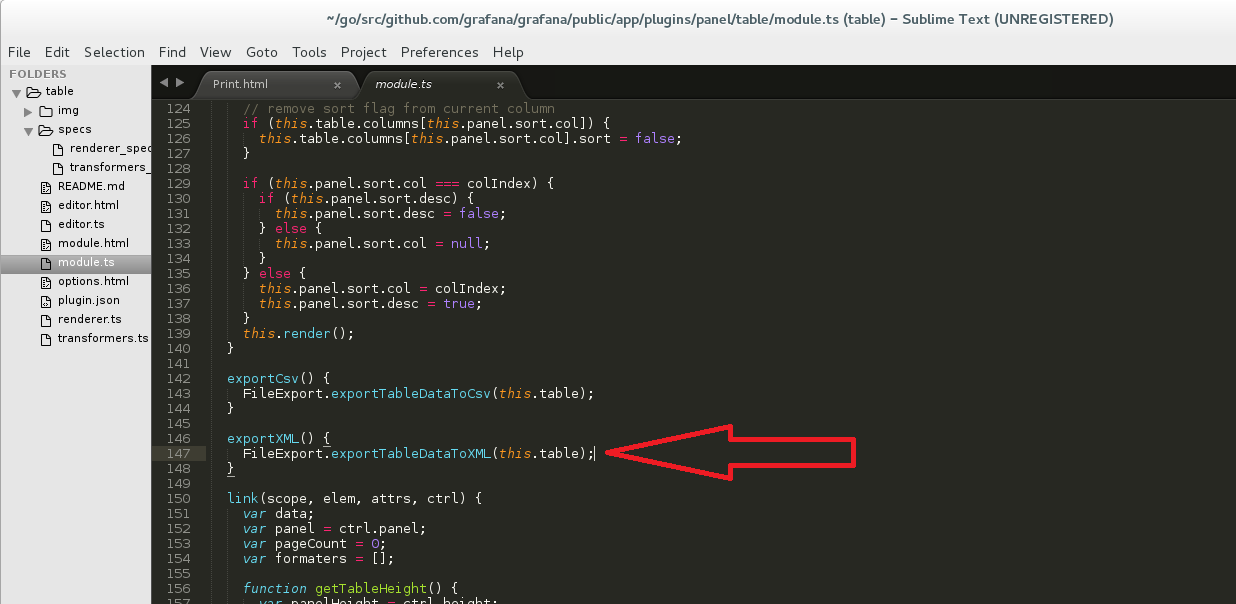
Update ./public/app/plugis/panel/graph/module.ts for graph

Or Update ./public/app/plugis/panel/table/module.ts for table

Add actions.push({text: ‘Your button name’, click: ‘ buttonFunc()’});



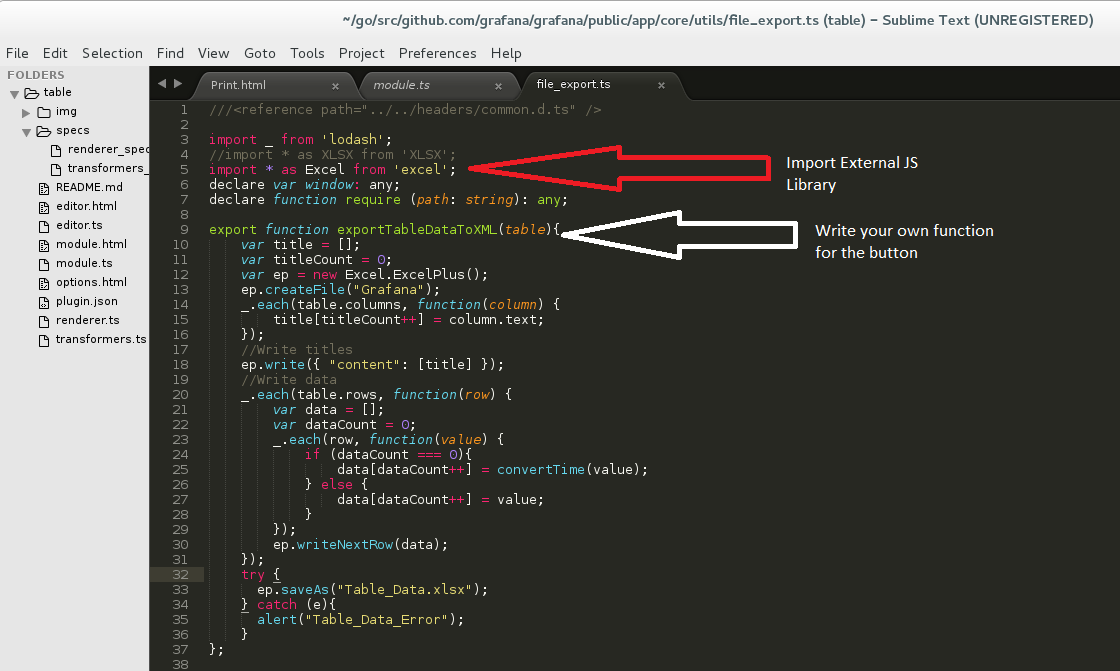
Add buttonFunc(){ FileExport.buttonFunc(…);}



1. Implement function of button

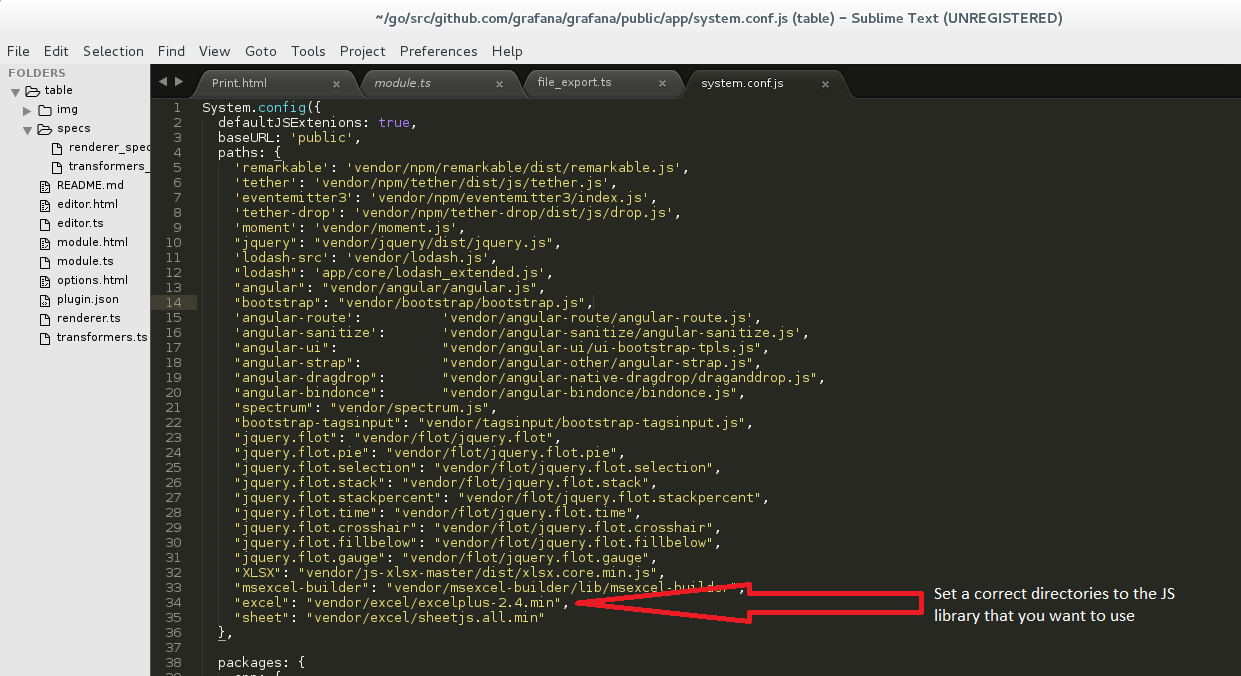
Navigate to ./public/app/core/utils/file\_export.ts

Import necessary files and then write your own function



1. Set directories for external library (if you want to use external Javascript library)

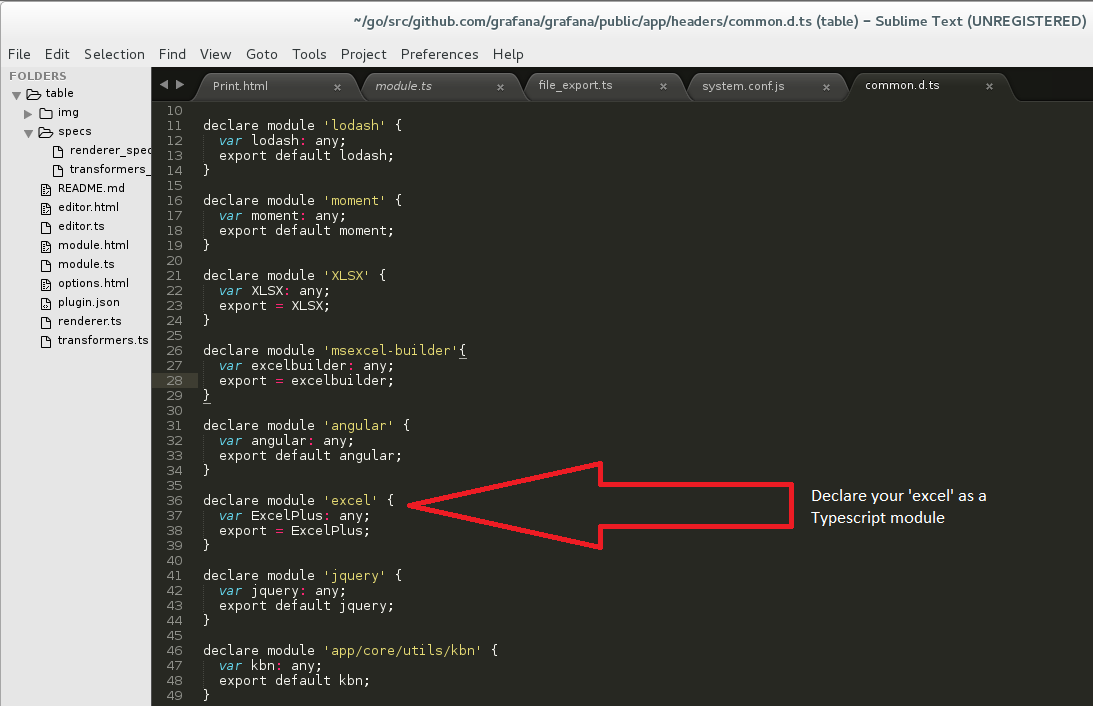
Navigate to ./public/app/system.conf.js and then update the correct path to your JS library. In this case: path to “excel” module is vendor/excel/excelplus-2.4.min



1. Declare the newly added Javascript library as a module for Typescript

Since you are using JS library for Typescript (Grafana), you will need to declare your JS library as a MODULE

Navigate to ./public/app/headers/common.d.ts and declare ‘excel’ module

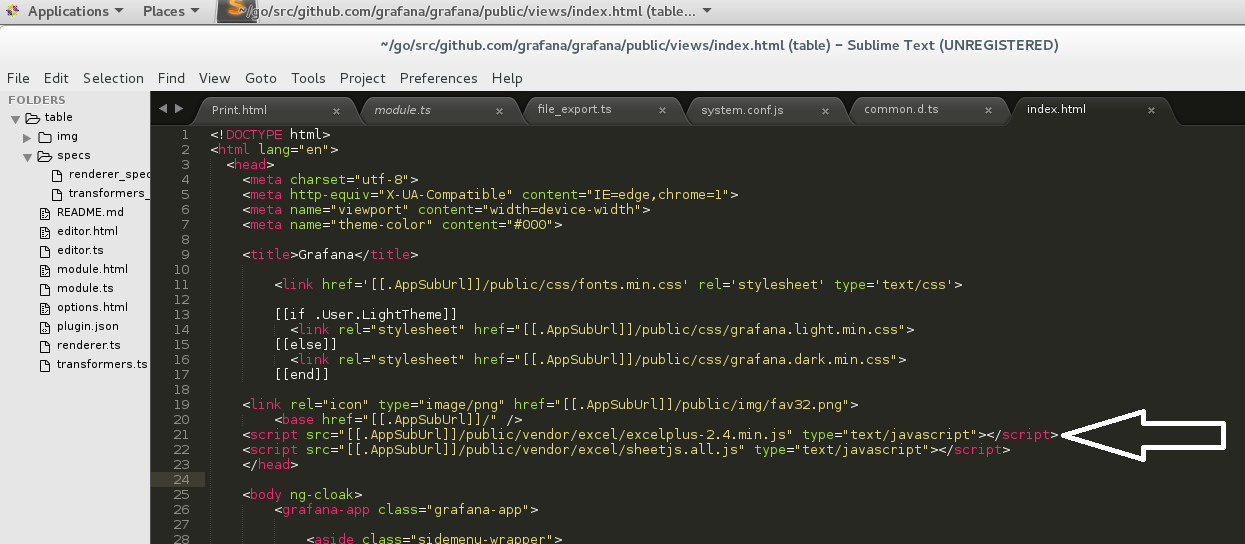


1. Place the external library in correct directories

Put your external JS library to correct directory that you have set in step 3. For example ./public/vendor/excel/

1. Import Javascript ( \*.js ) file from external library to the require page that uses the function

Since I am using my external JS function in Grafana Homepage. I will edit the ./public/views/index.html to include the library <script src = “…/excelplus.js”>



# 4. Author

Name: Nguyen Luong Chuong Thien

Last updated: July 5, 2016

Email: [terryn@ncs.com.sg](mailto:terryn@ncs.com.sg) or [nl.chuongthien@u.nus.edu](mailto:nl.chuongthien@u.nus.edu)