+++ title = “Development with local Grafana” +++

Development with local Grafana

This guide allows you to setup a development environment where you run Grafana and your plugin locally. With this, you will be able to see your changes as you add them.

Run Grafana in your host

If you have git, Go and the required version of NodeJS in your system, you can clone and run Grafana locally:

1. Download and set up Grafana. You can find instructions on how to do it in the developer-guide.
2. Grafana will look for plugins, by default, on its `data/plugins` directory. You can create a symbolic link to your plugin repository to detect new changes:

   ```
   ln -s <plugin-path>/dist data/plugins/<plugin-name>
   ```

3. (Optional) If the step above doesn't work for you (e.g. you are running on Windows), you can also modify the default path in the Grafana configuration (that can be found at `conf/custom.ini`) and point to the directory with your plugin:

   ```
   [paths]
   plugins = <path-to-your-plugin-parent-directory>
   ```

Run Grafana with docker-compose

Another possibility is to run Grafana with docker-compose so it runs in a container. For doing so, create the docker-compose file in your plugin directory:

NOTE: Some plugins already include a docker-compose file so you can skip this step.

```
version: '3.7'

services:
  grafana:
    # Change latest with your target version, if needed
    image: grafana/grafana:latest
    ports:
      - 3000:3000/tcp
    volumes:
      # Use your plugin folder (e.g. redshift-datasource)
      - ./dist:/var/lib/grafana/plugins/<plugin-folder>
```

```
    - ./provisioning:/etc/grafana/provisioning
  environment:
    - TERM=linux
    - GF_LOG_LEVEL=debug
    - GF_DATAPROXY_LOGGING=true
    - GF_DEFAULT_APP_MODE=development
```

Run your plugin

Finally start your plugin in development mode. Go to your plugin root directory and follow these steps:

1. Build your plugin backend and start the frontend in watch mode:

   ```
   mage -v
   yarn watch
   ```

2. Start Grafana backend and frontend:

   2.1 For a local copy of Grafana, go to the directory with Grafana source code and run:

   ```
   make run
   ```

   ```
   yarn start
   ```

   2.2 Or with docker-compose, in your plugin directory, run:

   ```
   docker-compose up
   ```

After this, you should be able to see your plugin listed in Grafana and test your changes. Note that any change in the fronted will require you to refresh your browser while changes in the backend may require to rebuild your plugin binaries and reload the plugin (`mage && mage reloadPlugin` for local development or `docker-compose up` again if you are using docker-compose).