Table of Contents

Tracker DQM offline tools

Introduction

The plots produced by the DQM modules are visualized using a dedicated DQM GUI, that permits to navigate easily the numerous plots needed to properly monitor the detector status and the quality of data reconstruction. A quick introduction to the DQM GUI functionalities and usage can be found in this twiki, or in the Tracker DQM Tutorial slides. In summary, the DQM GUI, is organized in different workspaces, each relative to a different DPG (i.e SiStrip, Pixel) and POG (i.e Tracking), that can be inspected (and also modified) in a user-friendly web interface.
The hierarchy structure of the different workspaces reflect the structure of the relative DQM files, plus some "special" folders meant to give a quick summary view of the full content created at the GUI level to ease data monitoring.
In particular each workspace contains a "QuikCollection" folder with the fundamental plots needed to asses the status of that particular subsystem (i.e SiStrip, Pixel...) and a "Layout" folder with a more extended expert-view. Also "special" workspaces are created to summarize the status of the full CMS detector, as an example the "Summary" workspace contains basic plots showing quality test results on the different CMS subsytems, or the "Shift" workspaces where central shifters can check a basic collection of plots from each subsystem.

The DQM GUI is the main tool for data monitoring and certification, used extensively by the Tracker Offline shift crew and central DQM shifters. The Tracker DQM team is in charge of maintaining the content of SiStrip, Pixel and Tracking workspaces as well as the "Shift" and "Summary" worksapaces for the Online and Offline flavour of the GUI, while the Central DQM team is in charge of the software implementation of the GUI.
In particular, for what concerns the GUI development, the Tracker DQM is responsible to decide the content of the "QuickCollection" and "Layout" folder for the three workspaces above, plus the "Shift" workspace for the Offline and Online DQM GUI, and to provide a proper description of each plot.

Some useful links

- The github link to the DQM GUI code: https://github.com/dmwm/deployment.
- You need to have your GRID certificate installed in your browser (instructions).
- offline and online DQM Graphical User Interface (GUI) (GRID certificate needed, instructions to get the certificate)
- For more detailed and complete documentation to develop DQM applications please refer to the central DQM page here
- If interested also in Online DQM development, please refer to this page.
- Documentation to run DQM packages can be found here DQMOffline a basic guide for beginners can be found here: BasicInstructionsforTrackerDQMDevelopers
- If you need any support to develop Tracker specific DQM modules, please email hn-cms-trk-dqm@cernNOSPAMPLEASE.ch.
- If you find any problem on running standard DQM configurations, please email the central DQM hn-cms-dqmDevel@cernNOSPAMPLEASE.ch.

Goal of this page:

This twiki page collects a set of basic instructions to perform the standard operation of development/maintainace of the DQM GUI relevant for the Tracker DQM team. In particular the content of the "Quick Collection" and "Layout" folder for the SiStrip, Pixel and Tracking workspaces are decided and modified, if needed, by the Tracker DQM team, the same applies to the "Shift" workspace. Moreover descriptions of plots shown in the twiki, have to be provided by Tracker DQM too.

The modifications to the GUI usually performed by the Tracker DQM are moving (or removing) plots in the "special" workspaces and folders, adding decriptions and customization to the plots. This page contains instructions to perform and test these standard developments.

1. The GUI and the layouts

Very detailed informations can be found in Sandbox.DQMGuiForUsers and won't therefore be rewritten here. We strongly recommend newcomers to go through the whole twiki. On this twiki, one can find information about what is the DQM GUI, what are the different flavors of GUI, how the release cycle works, and how to set up your own DQM GUI.

2. Test of the online DQM GUI

In order to be able to reproduce the way the data is fed in the online flavor or the DQM GUI, a dedicated machine has been set up for the tracker group: fu-c2f11-21-01. All the information related to this machine (how to access it, how to feed a run in the GUI, etc) is described thoroughly in the DQMOnlineTesting twiki. More tracker related information can be found here.

3. How to...

This section will give straightforward instructions to be able to manipulate the GUI, improving the layouts and is basically a "ready to use" summary of what can be found in the twiki referred above. If it is the first time you play with those tools, or if you would like to learn more about what you are doing, we strongly recommend to read the first twiki referenced above : Sandbox.DQMGuiForUsers

The following two steps are mandatory to set up your own DQM GUI and to be able to perform and test new changes:

Create your own new GUI

Create your own new GUI Hide 'Create your own new GUI'
Go on lxplus and download the code in a temporary directory:

```
ssh <username>@lxplus.cern.ch  #login to lxplus
/bin/bash
export PATH=$PATH:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin
mkdir -p /tmp/$USER/testGui
cd /tmp/$USER/testGui     #we will work in tmp to avoid issue with non-closed GUI (more secu
git clone git://github.com/dmwm/deployment.git
$PWD/deployment/Deploy -A slc6_amd64_gcc493 -r "comp=comp" -R comp@<LATESTTAG> -t MYDEV -s "
source current/apps/dqmgui/128/etc/profile.d/env.sh
```

Choose your favorite flavor:

```
$PWD/current/config/dqmgui/manage -f dev start "I did read documentation"
$PWD/current/config/dqmgui/manage -f online start "I did read documentation"
$PWD/current/config/dqmgui/manage -f offline start "I did read documentation"
$PWD/current/config/dqmgui/manage -f relval start "I did read documentation"
```

Now you can browse to your private DQM GUI in your favorite browser, by filling in your host name in the following address: (If you don't know the host name, type the command **`hostname`** to find out.)

```
hostname:8060/dqm/dev
hostname:8070/dqm/online-dev
hostname:8080/dqm/offline
hostname:8081/dqm/relval
```

Important note: if you are not at P5, you will need to tunnel to lxplus (see section Tunneling to lxplus).
Since you will be working in a /tmp/ directory, this means all you do is linked to a specific UI. Therefore, if you plan to make a long work don't forget to write down the UI you're working on (e.g. lxplus054)... and also to make quick backups since /tmp/ folder is meant to be cleaned frequently.

Feed the GUI with a DQM file

Feed the GUI with a DQM file Hide 'Feed the GUI with a DQM file'

Step1: Find data

According to the flavor you want to run on, find the ROOT file you are interested in (StreamExpress, ZeroBias, ...)

```
Offline: https://cmsweb.cern.ch/dqm/offline/data/browse/ROOT/OfflineData
Relval: https://cmsweb.cern.ch/dqm/relval/data/browse/ROOT/
Online: https://cmsweb.cern.ch/dqm/online/data/browse/Original
```

Step 2: Download the file to lxplus

To download directly the file on lxplus, you can write:

```
grid-proxy-init -rfc -bits 1024
myproxy=`grid-proxy-info -path`
wget --certificate=$myproxy --private-key=$myproxy --ca-certificate=$myproxy https://cmsweb
```

Due to certain convention, the file name should be version 1 (DQM_V0001), please change the version (name) of your file before uploading it to the GUI (see Step 3).

Step 3: Upload your DQM file to your GUI (according to the flavor you want to run on)

```
source current/apps/dqmgui/128/etc/profile.d/env.sh

visDQMUpload http://<your_hostname>.cern.ch:8060/dqm/dev FILENAME.root
visDQMUpload http://<your_hostname>.cern.ch:8070/dqm/online-dev FILENAME.root
visDQMUpload http://<your_hostname>.cern.ch:8080/dqm/offline FILENAME.root
visDQMUpload http://<your_hostname>.cern.ch:8081/dqm/relval FILENAME.root
```

After some time it should be uploaded to your GUI, you can select the uploaded run like in the production GUI (for example, in the dev and online flavors of the GUI, you will need to click on the **`Live`** button to be able to chose your run). Same applies if you want to switch between different files you uploaded.
Note: you can upload several files for a single runs (e.g. Pixel and SiStrip). It will still appear as a single run in the GUI but will contain both files.

Now that your own DQM GUI is woking, you can go ahead editing it. The next point will present an overview of the DQM GUI from the tracker point of view. Then, a list of "How to's" follows with specific instructions for tasks you could be asked to do.

Edit the GUI - the structure

Edit the layouts Hide 'Edit the layouts'
Once your custom GUI installed, go in the dqmgui folder: **`cd deployment/dqmgui`**.
There, you will find several files :
- **`server-conf-*.py`**, where you can find which layouts are loaded in which GUI flavor. This is particularly important if you wish to create a new GUI file, or to understand if the layouts you just changed will be seen in a specific flavor of the GUI.
- **`workspaces-*.py`**, this file shows the Layouts for the Quick Collection, i.e. what will be displayed when the workspace is opened for the first time.
- **`layouts`**, the folder containing all the configuration files for each workspace layout.
- `=style`, the folder containing all the customisation for the plots in the DQM GUI, such as logarithmic axes, grid option, drawing lines...

The dqmgui folder is a **development folder**. It means that the change you will make there will **not** be directly applied to your test server. To apply those changes, you will have to re-deploy, using the same commands as before:

```
$PWD/deployment/Deploy -A slc6_amd64_gcc493 -r "comp=comp" -R comp@<LATESTTAG> -t MYDEV -s "
source current/apps/dqmgui/128/etc/profile.d/env.sh
```

Add new plots to the QuickCollection

Add new plots to the QuickCollection Hide 'Add new plots to the QuickCollection'
To add a plot in the quick collection:

1. Open the **`workspace-.py`** you want to update
2. Add the QuickCollection path (get inspiration from other lines). Please always start by adding plots in layouts before putting them in Quickcollection. The plots in QuickCollection should **always** be in layouts, if possible with the same index.

Plots in QuickCollection are the most important plots a shifter will have to look at. They need to be properly documented with a good description and a specific description in the related twiki:

- **`for SiStrip`** : https://twiki.cern.ch/twiki/bin/viewauth/CMS/SiStripOfflineDQMHistoDescription
- **`for Pixel`** : https://twiki.cern.ch/twiki/bin/view/CMS/PixelOfflineDQMHistoDescription
- **`for Tracking`** : https://twiki.cern.ch/twiki/bin/view/CMS/TrackingOfflineDQMHistoDescription

Edit the layouts (add new plots, add a reference, add a description...)

Edit the layouts (add new plots, add a reference, add a description...) Hide 'Edit the layouts (add new plots, add a reference, add a description...)'
This section explains how to play with the layouts in the dqmgui/layouts folder (github link).

The folder structure

There are different python files for each workspaces and flavors of the DQM GUI. They follow this nomenclature: `=T0 = offline; shift = shift workspace; overview = feedback for collisions.`
Here is the list of interest for the tracker community:

For SiStrip:

- sistrip-layouts.py for online flavor of the GUI

- sistrip_T0_layouts.py for the offline GUI
- shift_sistrip_layout.py for what is in the Shift/SiStrip workspace of the online GUI
- shift_sistrip_T0_layout.py for what is in the Shift/SiStrip workspace of the offline GUI
- sistrip_overview_layouts.py for the layouts of the tracking feedback collection in both offline and online GUI
- sistriplas-layout.py for the sistripLAS workspace of the online GUI (not used)

For Pixel:

- pixel-layouts.py for online flavor of the GUI
- pixellumi-layouts.py for the layouts of the workspace of PixelLumi... which only exist in "Everything" (not used)
- pixel_T0_layouts.py for offline flavor of the GUI
- shift_beampixel_layout.py for what is in the Shift/BeamPixel workspace of the online GU
- shift_pixel_layout.py for online flavor of the GUI
- shift_pixellumi_layout.py for what is in the Shift/PixelLumi workspace of the online GU
- shift_pixel_T0_layout.py for what is in the Shift/Pixel workspace of the offline GUI

For Tracking:

- shift_tracking_T0_layout.py for what is in the Shift/Tracking workspace of the offline GUI
- tracking_T0_layouts.py for the tracking worskpace in the offline GUI

Note: You can see at the top of the file where is the layout folder located in case you have any doubts.

The python files structure

At the top of each python file, you can find the information of where is this layout applied, then follows the definition of each layout with:

- a title for the "layout object" (a "layout object" can contain several plots)
- then for each plot in this object, you have the following options:
  - **`path`**, the path in DQM GUI of this histogram
  - **`description`**, the description that will be shown when the "Describe" button is hit in the GUI.
    It is *mandatory* to add a description to new plots. If you created a new plot and want to add it to the layouts, it's **your** role to provide a good description to it. Furtheremore, if the plot is added to the layouts, then its description should also figure here.
  - **`reference`**, to automatically plot the reference on this plot. Most of the time this option is set to "no".

Therefore, to add a new layout, just create a new layout object. You can get inspiration by looking at the other lines in the code.
Always test your code, even if it looks trivial, a typo is really easy to make and can be deadly to the GUI.

Update the documentation

Plots in Layouts are (after the QuickCollection ones) the most important plots a shifter will have to look at. They need to be properly documented with a good description and a specific description in the related twiki:

- **`for SiStrip`** : https://twiki.cern.ch/twiki/bin/viewauth/CMS/SiStripOfflineDQMHistoDescription
- **`for Pixel`** : https://twiki.cern.ch/twiki/bin/view/CMS/PixelOfflineDQMHistoDescription
- **`for Tracking`** : https://twiki.cern.ch/twiki/bin/view/CMS/TrackingOfflineDQMHistoDescription

If you have doubts about what to write there, contact the person who asked you to make this plot.

Edit the style (log scale, colz option, lines, etc.)

Edit the style of some plots in the GUI (log scale, colz option, etc.) Hide 'Edit the style of some plots in the GUI (log scale, colz option, etc.)'
This section explains how to play with the style of the plots, in the dqmgui/style folder (github link)

The folder structure

Here, there are less files, since those C++ files are both for online and offline flavor of the GUI.

For SiStrip:

- `SiStripRenderPlugin.cc`, for the SiStrip workspace
- `SiStripLASRenderPlugin.cc`, for the SiStripLAS workspace **(not used)**

For Pixel:

- `SiPixelRenderPlugin.cc`, for the Pixel workspace
- `BeamPixelRenderPlugin.cc`, for the BeamPixel workspace

The C++ files structure

1. The first function (`applies`) shows in which folders this RenderPlugin is applied. One has to be careful to do not alter other folders (like BeamPixel with Pixel) with too evasive keywords.
2. Then there are several `preDraw` functions for each kind of histogram (TH2F, TH1F, ...). This is where you should add your customization for new plots, such as **colz option**. %BR A common mistake is to add the plot in the wrong category (e.g. TH2F instead of a TProfile2D). Therefore be careful and **test your changes**!
3. Then there are `postDraw` functions. This is where you add things that has to come after drawing the histograms, like adding an "error message" when something happens, or to change to a logarithmic axis if there are more than N entries, etc. This is also where you can add lines to be drawn on the plot to show expected ranges.

As always, what you want to implement is probably already implemented for an other plot, so get inspiration from there!
When adding a new plot (through the `name.find` function), please be careful to:

- avoid duplicating functions. Mayeb there is already a line that is changing your plot, but it just misses the (e.g) colz option.
- be smart, don't create 5 lines if you have 5 new plots with almost the same name and that requires the same options.

Test your changes

Test your changes Hide 'Test your changes'
Testing your changes is the most important step. Please be sure to **always** check what you did before sending any pull request.

1. You should already have a DQM instance running if you followed the two first steps of this How to section. If not, please refer to those two steps first.
2. Then, re-deploy to copy your changes to your running GUI:

```
cd /tmp/$USER/testGui
$PWD/deployment/Deploy -A slc6_amd64_gcc493 -r "comp=comp" -R comp@<LATESTTAG> -t MYDI
source current/apps/dqmgui/128/etc/profile.d/env.sh
```

3. And restart your GUI (depending on the flavour you're working on):

```
$PWD/current/config/dqmgui/manage -f dev restart "I did read documentation"
$PWD/current/config/dqmgui/manage -f online restart "I did read documentation"
$PWD/current/config/dqmgui/manage -f offline restart "I did read documentation"
$PWD/current/config/dqmgui/manage -f relval restart "I did read documentation"
```

4. Have a (good) look at your changes:

```
hostname:8060/dqm/dev
hostname:8070/dqm/online-dev
hostname:8080/dqm/offline
hostname:8081/dqm/relval
```

Send a Pull Request and close your GUI

Send a Pull Request and close your GUI Hide 'Send a Pull Request and close your GUI'

Make the Pull Request

- Make sure you have forked the *deployment* repo to your account. You can do this here by clicking on the *Fork* button.
- Create a separate branch in git (strongly recommended).

```
cd /tmp/$USER/testGui/deployment

git checkout -b MyNewBranchName
```

- Commit your changes. Please use a smart and logical message.

```
git status

git add modifiedFile1 modifiedFile2 ...

git commit -m "smart and logical mesage"
```

- Add your private repo as a remote and push the branch to your private repo.

```
git remote add my-deployment git@github.com:<my_github_account>/deployment.git

git push my-deployment <my_branch_name>
```

- Before making any further request, make sure that you have thoroughly tested your development!
  - Restart the DQM GUI and make sure your modified layouts and render plugins compile without any problems.
  - Please keep your code clean: Make sure you have no trailing spaces or trailing newlines.
  - If you need to modify things, you can simply make commits to your current branch and push it again.
- If you are changing anything to the **shift layouts** for your subsystem: **update your subsystem's shift instructions**
  - Please understand that out of respect for our shifters, we will not even consider your change if it's not documented first.
  - Off course, the instructions should not appear for the shifter until the updated layout is on production. For that reason, please place the new instructions between `<!--` and `-->` in the Twiki source, so that we can review them, without the shifter seeing them.
  - If you have questions, don't hesitate to contact us.

- After successful testing (and, if applicable, update of the instructions), go to Github and create a pull request for the branch you just pushed. Please give the pull request a smart and logical title and description.
- We will see that you created a pull request, but please send an email to cms-dqm-coreTeam@cernNOSPAMPLEASE.ch nonetheless.
    - ♦ Let us know if this is a patch for the Online DQM GUI, the Offline DQM GUI, the Relval DQM GUI or all of them.
    - ♦ Mention which pull request you're emailing about.
    - ♦ Don't hesitate to mention any other useful information.
- We will pick this up, test if the patch process works and install it on the *Online Test* system and the *Online Playback* system.
    - ♦ If the modification can be tested on the Online flavor, you can test it now.
        - ◊ Depending on what you want to test, we have to DQM GUI test servers looking at different data:
            - · On the *Online Test* system, you will find exactly the same data as on the production system.
            - · On the *Online Playback* system, you will find data produced with the latest CMSSW developments that are currently being tested.
        - ◊ If testing is successful, we will propagate it to the DQM GUI Online Production systems.
        - ◊ After that we will also accept the pull request so that it will be merged in the next official release (for Offline and Relval).
    - ♦ If the modification can only be tested on the Offline or Relval flavor:
        - ◊ We will accept the pull request so that it can be tested during the testing period of the next official release.
        - ◊ You test during the official testing period and let us know if everything works as expected.

Close your DQM GUI (depending of the flavor)

This is a very important step do not overload the lxplus machine for nothing.

```
$PWD/current/config/dqmgui/manage -f dev stop "I did read documentation"
$PWD/current/config/dqmgui/manage -f online stop "I did read documentation"
$PWD/current/config/dqmgui/manage -f offline stop "I did read documentation"
$PWD/current/config/dqmgui/manage -f relval stop "I did read documentation"
```

4. Additional information

In this section, we present tutorials for experts or for more technical issues (like tunneling).

Tunneling to lxplus

Tunneling to lxplus Hide 'Tunneling to lxplus'
To ease things, we strongly recommend to use a browser add-on to play with proxy settings. In this section we present a way to do it on firefox with the add-on **`FoxyProxy`**

1. Install the add-on
    - ♦ Open the add-on page on firefox, for this write **`about:addons`** in the address bar.
    - ♦ Search for "FoxyProxy" and install the "FoxyProxy Standard" version of this add-on.
    - ♦ Restart your browser.
2. Configure your proxy

- Once installed, you will see a "fox head" icon on the right of your address bar.
- Right-click on it and click on option.
- Then click on "Add new proxy":
- select "Manual Proxy Configuration"
- Host or IP Address: localhost
- Port: 11080
- Select SOCKS Proxy
- Switch tab and go in "General" : there you can choose your settings name (set it to lxplus for example)
- Switch tab and go in "URL patters": Click on "add New Pattern"
- choose a pattern name (for example lxplus)
- in URL pattern, set expression to tell when this proxy should be active: set it to **`lxplus`** and click on OK.
- to avoid bugs in some cases, add again a new pattern. Name it "NOT google" and set the URL pattern to **`google`** . But this time select the blacklist option and then press OK.
- eventually, since tunneling to lxplus is also used when working on vocms machine, add a new pattern. Name it "vocms" and set the URL pattern to **`vocms`** and then press OK.
- then close the windows and right-click again on the "fox head" icon and select "Use proxies based on their pre-defined patterns and priorities"

3. Start the tunneling
   - Once FoxyProxy set, you won't have to touch it anymore. When you will connect to lxplus or vocms it will automatically try to use your tunnel options. All you have to do is to open the tunnel, with the following command:

```
ssh -D 11080 <username>@lxplus.cern.ch
```

And you can now navigate through your personal GUI, or the vocms machines, ...

Play with our own online GUI (experts only)

Create our own online GUI (expert only) Hide 'Create our own online GUI (expert only)'
In order to be able to reproduce the way the data is fed in the online flavor or the DQM GUI, a dedicated machine has been set up. Here are the different steps to be able to connect to this machine and use it. If one wants to know more about how to set up such a machine, please refer to the Create our own GUI. Also, please note that the central DQM has made its own general twiki, which englobe this one : DQMOnlineTesting.

1. Working on the right machine
   Tunnel to lxplus, then to cmsuser, then connect to the dedicated machine for tracker (fu-c2f11-21-01).

```
ssh -tC username@lxplus.cern.ch ssh -tC username@cmsusr.cern.ch ssh -C username@fu-c
```

2. Set up CMSSW and the proxy (just do it the first time)
   On the test machine, edit your .bashrc and add the following

```
prod_machines=fu-c2f11-21-01
localhost=$(hostname)

case "${prod_machines[@]}" in
*$localhost*)
    # Getting cmssw
    source /opt/offline/cmsset_default.sh
    export SCRAM_ARCH=slc6_amd64_gcc491
    # Proxy settings
    export http_proxy="http://cmsproxy.cms:3128"
    export https_proxy="https://cmsproxy.cms:3128/"
```

```
    export NO_PROXY=".cms"
    ;;
esac
```

3. Set up github (just do it the first time)
   Follow the usual steps to set up github:

```
git config --global user.name <name> <last name>
git config --global user.email <email adress>
git config --global user.github <username>
```

   Register your ssh key.
   However, you will need to follow some additional steps (those instructions originate from the DQMOnline twiki. Please note this guide only works if github access is via ssh (--ssh option))

   - Login to cmsusr and edit your .ssh/config file adding:

```
Host github.com
    User git
    ProxyCommand ssh cmsusr.cms nc %h %p
```

   - remove the folder `~/.cmsgit-cache/`
   - copy your `.ssh` folder contents from lxplus to cmsusr
   - Make sure that you set the ssh settings for 'forward agent' is `true`
   - login to the machine you want to reach (here fu-c2f11-21-01)
   - type `kinit` and insert the password
   - from inside any project, after `cmsenv` git and the cms-git-tools should work with the --ssh option

4. CMSSW and addition of DQM/Integration
   On the machine fu-c2f11-21-01, write

```
kinit <username>
scram list
scram project -n <CMSSW_version>_official <CMSSW_version>
cd <CMSSW_version>/src
cmsenv
git-cms-addpkg DQM/Integration --ssh 2>&1 | cat;  #It is possible that a lot of "kil
scram buid -j8
```

   Do a quick check:

```
cat /etc/dqm_run_config
```

   This should give the following output:

```
[host]
type = userarea
```

5. Online configuration
   Now that we have the DQM/Integration package, we can configure two things:
   - The source of the input:
     in DQM/Integration/python/config/inputsource_cfi.py, under 'runInputDir', change `'/tmp'` by `'/fff/BU0/output/lookarea'`
   - The destination of the output:
     in DQM/Integration/python/config/environment_cfi.py, change userarea `'collectorHost'` by `'fu-c2f11-21-01'`

Then, don't forget to build the project:

```
scram b -j8
```

6. Installing an Online DQM GUI (already done)
   This step shouldn't be needed since one GUI have been set once for all. However, if the expert user would need to set up a whole new GUI, some information are given in installing an Online DQM GUI.
7. Feed the GUI
   Always start to check if the run you want to look is available:

   ```
   ls -l /fff/BU0/output/lookarea/runXXXXXX/
   ```

   If you need collisions, check it in the WBM Run Summary.

   Create the `./upload` directory fom where you will launch `cmsRun`

   ```
   cd DQM/Integration/python/clients/
   mkdir ./upload
   ```

   Run the typical configuration of your subsystem:

   ```
   cmsRun subsystem_config.py runNumber=abcxyz
   #E.g cmsRun sistrip_dqm_sourceclient-live_cfg.py runNumber=261259
   ```

   This step should take a few minutes. It will "freeze" automatically after the last LS is reached.
8. Check the DQM GUI
   Tunnel to P5 from outside:

   ```
   ssh -tCXgL 22223:localhost:22224 username@lxplus.cern.ch ssh -tCXgL 22224:localhost:
   ```

   Now you can use your browser with port 22223.
   The address is: fu-c2f11-21-01:8070/dqm/online-dev

Installing our own DQM GUI (experts only)

Installing our own DQM GUI (experts only) Hide 'Installing our own DQM GUI (experts only)'
This section is still to be done. If needed, please contact hugo.ruben.delannoy@cernNOSPAMPLEASE.ch. In any case you will need special rights to do this.

This topic: Sandbox > TrackerDQMOfflineTools
Topic revision: r1 - 2017-05-11 - HugoRubenDelannoy