Tango Workshop @ ICALEPCS 2021
PyTango and jupyTango

October 2021

Anton Joubert
NRF/SARAO

Nicolas Leclercq
ESRF

Acknowledgements

Sergi Rubio (ALBA) https://github.com/sergirubio

Vincent Michel https://github.com/vxgmichel

Karoo Team (SARAO)

Agenda

Introduction

Docker compose environment

Simple Tango device servers

~~API: Low-level vs. High-level~~

ITango for easy client access

JupyTango

~~Events and polling~~

~~Miscellaneous~~

~~How to test?~~

Additional resources

~~Strikethrough~~ items: in slide deck, but won't be covered today

Introduction

What is PyTango?

Python library

Binding over the C++ tango library

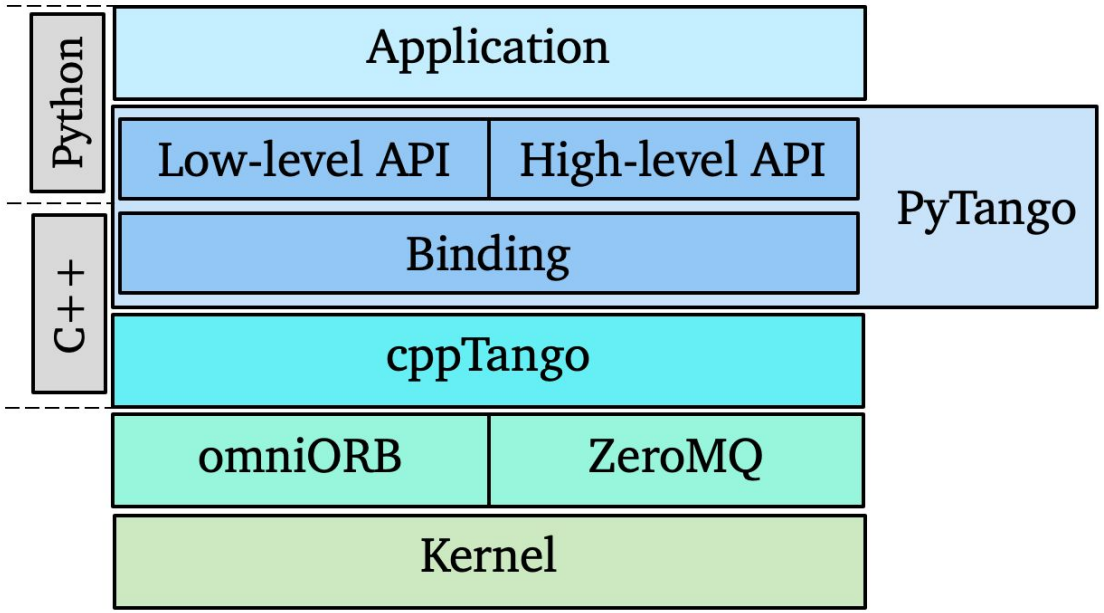
... using boost-python (future: pybind11)

Does not use omniorb Python library

Relies on numpy

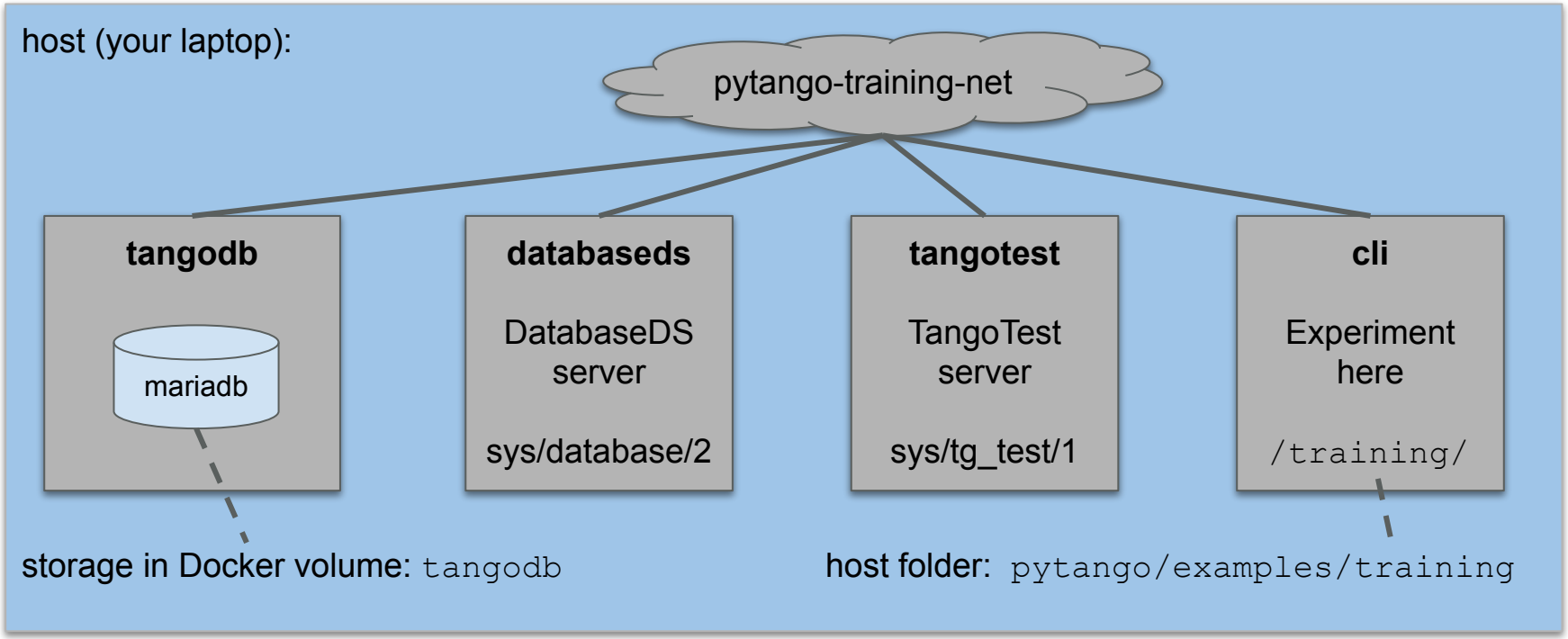
Multi OS: Linux, Windows, MacOS (sort-of)

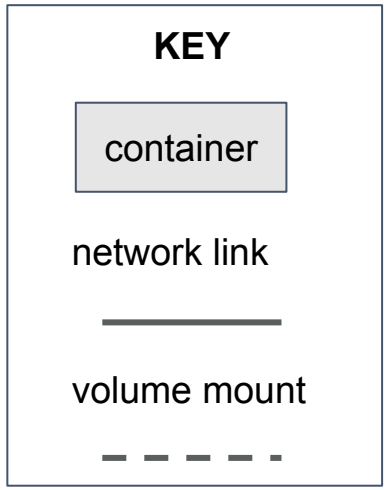
Works on Python 2.7, 3.5+



Docker compose environment

Docker compose setup





Repo URL: https://gitlab.com/tango-controls/pytango/-/tree/develop/examples/training

Start the Docker compose services

New Docker network required (once off):

```
➜  training git:(develop) ✗ docker network create pytango-training-net
3a55881054809b74546982482dcca9f90aecf4271f3abbf58fb43b8f7bca2311
```

Start services:

```
➜  training git:(develop) ✗ docker-compose up
Starting tangodb ... done
Starting databaseds ... done
Starting ipython    ... done
Starting tangotest  ... done
Attaching to tangodb, databaseds, tangotest, ipython
databaseds     | wait-for-it.sh: waiting 30 seconds for tangodb:3306
tangodb        | 2021-06-30 11:18:58+00:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:10.5.10+maria~focal star
ted.
databaseds     | wait-for-it.sh: tangodb:3306 is available after 0 seconds
tangodb        | 2021-06-30 11:18:58+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
databaseds     | main(): arrived
tangodb        | 2021-06-30 11:18:58+00:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:10.5.10+maria~focal star
ted.
tangodb        | Warning: World-writable config file '/etc/mysql/conf.d/sql_mode.cnf' is ignored
databaseds     | main(): export DataBase as named servant (name=database)
tangodb        | 2021-06-30 11:18:58 0 [Note] mysqld (mysqld 10.5.10-MariaDB-1:10.5.10+maria~focal) starting as process 1 ...
databaseds     | Ready to accept request
```

Simple Tango device servers

Trivial *PowerSupply* device

A device server with a single device:

```python
#!/usr/bin/env python3

"""
Trivial power supply device with no external connection or behaviour.
"""

from time import sleep
from tango.server import Device, attribute, command


class PowerSupply(Device):

    @attribute(dtype=float)
    def voltage(self):
        return 1.5

    @command
    def calibrate(self):
        sleep(0.1)


if __name__ == '__main__':
    PowerSupply.run_server()
```

File: training/server/ps0a.py

Try to run it

```
[→  training git:(add-training-examples) ✗ docker-compose exec cli bash
[tango@7ee8862308bd:/training$ cd server/
[tango@7ee8862308bd:/training/server$ ./ps0a.py --help
 usage :  PowerSupply instance_name [-v[trace level]] [-file=<file_name> | -nodb [-dlist <device name list>] ]
[tango@7ee8862308bd:/training/server$ ./ps0a.py test
 The device server PowerSupply/test is not defined in database. Exiting!
 tango@7ee8862308bd:/training/server$
```

```
[tango@7ee8862308bd:/training/server$ tango_admin --help

Usage:
 --help                  Prints this help
 --ping-database         [max_time (s)] Ping database
 --check-device <dev>    Check if the device is defined in DB
 --add-server <exec/inst> <class> <dev list (comma separated)>   Add a server in DB
 --delete-server <exec/inst> [--with-properties]   Delete a server from DB
 --check-server <exec/inst>   Check if a device server is defined in DB
 --server-list  Display list of server names
 --server-instance-list <exec>   Display list of server instances for the given server name
 --add-property <dev> <prop_name> <prop_value (comma separated for array)>    Add a device property in DB
 --delete-property <dev> <prop_name>   Delete a device property from DB
 --tac-enabled Check if the TAC (Tango Access Control) is enabled
 --ping-device <dev> [max_time (s)] Check if the device is running
 --ping-network [max_time (s)] [-v] Ping network
```

Register (once-off) and run it

```
tango@7ee8862308bd:/training/server$ tango_admin --add-server PowerSupply/test PowerSupply train/ps/1
tango@7ee8862308bd:/training/server$ ./ps0a.py test
Ready to accept request
```

Start another shell and connect to the device as client:

```
→  training git:(add-training-examples) ✗ docker-compose exec cli ipython3
Python 3.7.3 (default, Jan 22 2021, 20:04:44)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.21.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import tango

In [2]: dp = tango.DeviceProxy("train/ps/1")

In [3]: dp.ping()
Out[3]: 936

In [4]: dp.voltage
Out[4]: 1.5

In [5]: dp.calibrate()

In [6]:
```

Less trivial *PowerSupply* device

Device connects to external hardware via TCP.

Need to install `gevent` in container to run simulator:

```
docker-compose exec cli pip install gevent
```

Configuration via properties (can be overridden in the Tango Database)

```python
from time import sleep
from socket import create_connection

from tango.server import Device, attribute, command, device_property


def connect(host, port):...


def write_readline(conn, msg):...


class PowerSupply(Device):

    host = device_property(str, default_value='localhost')
    port = device_property(int, default_value=45000)

    def init_device(self):
        super().init_device()
        self.conn = connect(self.host, self.port)

    @attribute(dtype=float)
    def voltage(self):
        return float(write_readline(self.conn, b'VOL?\n'))

    @command
    def calibrate(self):
        write_readline(self.conn, b'CALIB 1\n')
        while int(write_readline(self.conn, b'stat?\n')):
            sleep(0.1)
```

File: training/server/ps1.py

ITango for easy client access

Connect to device

```
[→  training git:(add-training-examples) ✗ docker-compose exec cli itango3
ITango 9.3.3 -- An interactive Tango client.

Running on top of Python 3.7.3, IPython 7.21 and PyTango 9.3.3

help       -> ITango's help system.
object?    -> Details about 'object'. ?object also works, ?? prints more.

IPython profile: tango

hint: Try typing: mydev = Device("<tab>

[In [1]: # Device is an alias for tango.DeviceProxy

[In [2]: dev = Device("sys/tg_test/1")

[In [3]: # or can use class name (limits <tab> search space)

[In [4]: dev = TangoTest("sys/tg_test/1")

[In [5]: dev.ping()
Out[5]: 563
```

Commands and attributes

```
[In [6]: # send a command (low-level way)

[In [7]: dev.command_inout('DevShort', 1234)
 Out[7]: 1234

[In [8]: # send a command (high-level way)

[In [9]: dev.DevShort(1235)
 Out[9]: 1235

[In [10]: # read an attribute

[In [11]: dev.long_spectrum
 Out[11]:
 array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int32)

[In [12]: # write to it

[In [13]: dev.long_spectrum = (1, 2, 3, 4)

[In [14]: dev.long_spectrum
 Out[14]: array([1, 2, 3, 4], dtype=int32)
```

Built-in event monitor: `mon` command

```
In [15]: dev.poll_attribute('State', 3000)

In [16]: mon -a sys/tg_test/1/State
'sys/tg_test/1/State' is now being monitored. Type 'mon' to see all events

In [17]: dev.SwitchStates()

In [18]: mon
ID   Device                         Attribute          Value               Quality      Time
---- ------------------------------ ------------------ ------------------- ------------ ----------------
   0 sys/tg_test/1                  state              RUNNING             ATTR_VALID   21:04:55.149842
   1 sys/tg_test/1                  state              RUNNING             ATTR_VALID   21:04:58.149351
   2 sys/tg_test/1                  state              FAULT               ATTR_VALID   21:05:04.151308

In [19]: dev.SwitchStates()

In [20]: mon
ID   Device                         Attribute          Value               Quality      Time
---- ------------------------------ ------------------ ------------------- ------------ ----------------
   0 sys/tg_test/1                  state              RUNNING             ATTR_VALID   21:04:55.149842
   1 sys/tg_test/1                  state              RUNNING             ATTR_VALID   21:04:58.149351
   2 sys/tg_test/1                  state              FAULT               ATTR_VALID   21:05:04.151308
   3 sys/tg_test/1                  state              RUNNING             ATTR_VALID   21:05:16.148329

In [21]: mon -d sys/tg_test/1/State
Stopped monitoring 'sys/tg_test/1/State'
```

Run `mon?` for more details

End of ITango demo

More info: https://itango.readthedocs.io

It can also be used from a Jupyter notebook

jupyTango - Nicolas Leclercq

Build docker image

```
[➜  tango-src git clone git@gitlab.com:tango-controls/jupyTango.git
Cloning into 'jupyTango'...
remote: Enumerating objects: 212, done.
remote: Counting objects: 100% (125/125), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 212 (delta 66), reused 113 (delta 55), pack-reused 87
Receiving objects: 100% (212/212), 4.89 MiB | 949.00 KiB/s, done.
Resolving deltas: 100% (105/105), done.
[➜  tango-src cd jupyTango/docker
[➜  docker git:(develop) docker build -t jupytango:1.0.0 .
[+] Building 264.5s (15/15) FINISHED
 => [internal] load build definition from Dockerfile                                                                0.0s
 => => transferring dockerfile: 1.78kB                                                                              0.0s
 => [internal] load .dockerignore                                                                                   0.0s
 => => transferring context: 2B                                                                                     0.0s
 => [internal] load metadata for artefact.skao.int/ska-tango-images-tango-itango:9.3.4                              0.0s
 => [ 1/11] FROM artefact.skao.int/ska-tango-images-tango-itango:9.3.4                                              0.2s
 => [ 2/11] RUN DEBIAN_FRONTEND=noninteractive apt-get update && apt-get install -y procps git                     31.7s
 => [ 3/11] RUN python3 -m pip install opencv-python jupyterlab ipywidgets jupyter_bokeh                          218.5s
 => [ 4/11] RUN cp -Rf $HOME/.ipython/profile_default $HOME/.ipython/profile_jupytango                              0.3s
 => [ 5/11] RUN echo "config = get_config()" > $HOME/.ipython/profile_jupytango/ipython_config.py                   0.3s
 => [ 6/11] RUN echo "config.InteractiveShellApp.extensions = ['jupytango']" >> $HOME/.ipython/profile_jupytango/  0.4s
 => [ 7/11] RUN python -m ipykernel install --user --name jupyTango --display-name "jupyTango" --profile jupytango  1.0s
 => [ 8/11] RUN git clone -b master https://gitlab.com/tango-controls/jupyTango.git $HOME/jupyTango                 7.3s
 => [ 9/11] RUN cp $HOME/jupyTango/resources/logo/* $HOME/.local/share/jupyter/kernels/jupytango                    0.4s
 => [10/11] RUN export PYTHONPATH=$HOME/jupyTango                                                                   0.3s
 => [11/11] RUN export JUPYTER_CONTEXT=LAB                                                                          0.3s
 => exporting to image                                                                                              3.6s
 => => exporting layers                                                                                             3.5s
 => => writing image sha256:bb3e658008440fa9b2809129ee1d0d7e6896dfa923e9cd824c19f059bfb2ce5e                        0.0s
 => => naming to docker.io/library/jupytango:1.0.0                                                                  0.0s
```

Readme: https://gitlab.com/tango-controls/jupyTango/-/tree/develop/#giving-jupytango-a-try-using-docker

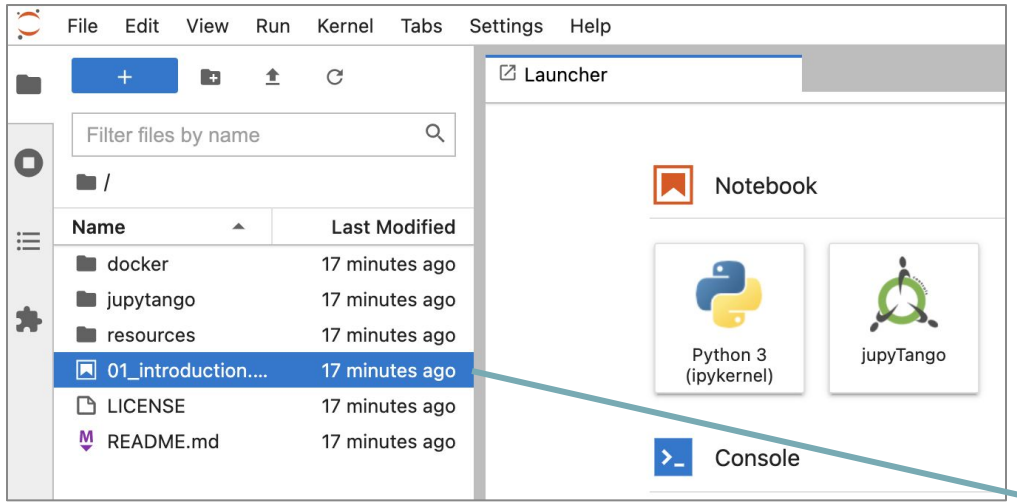
Run docker-compose

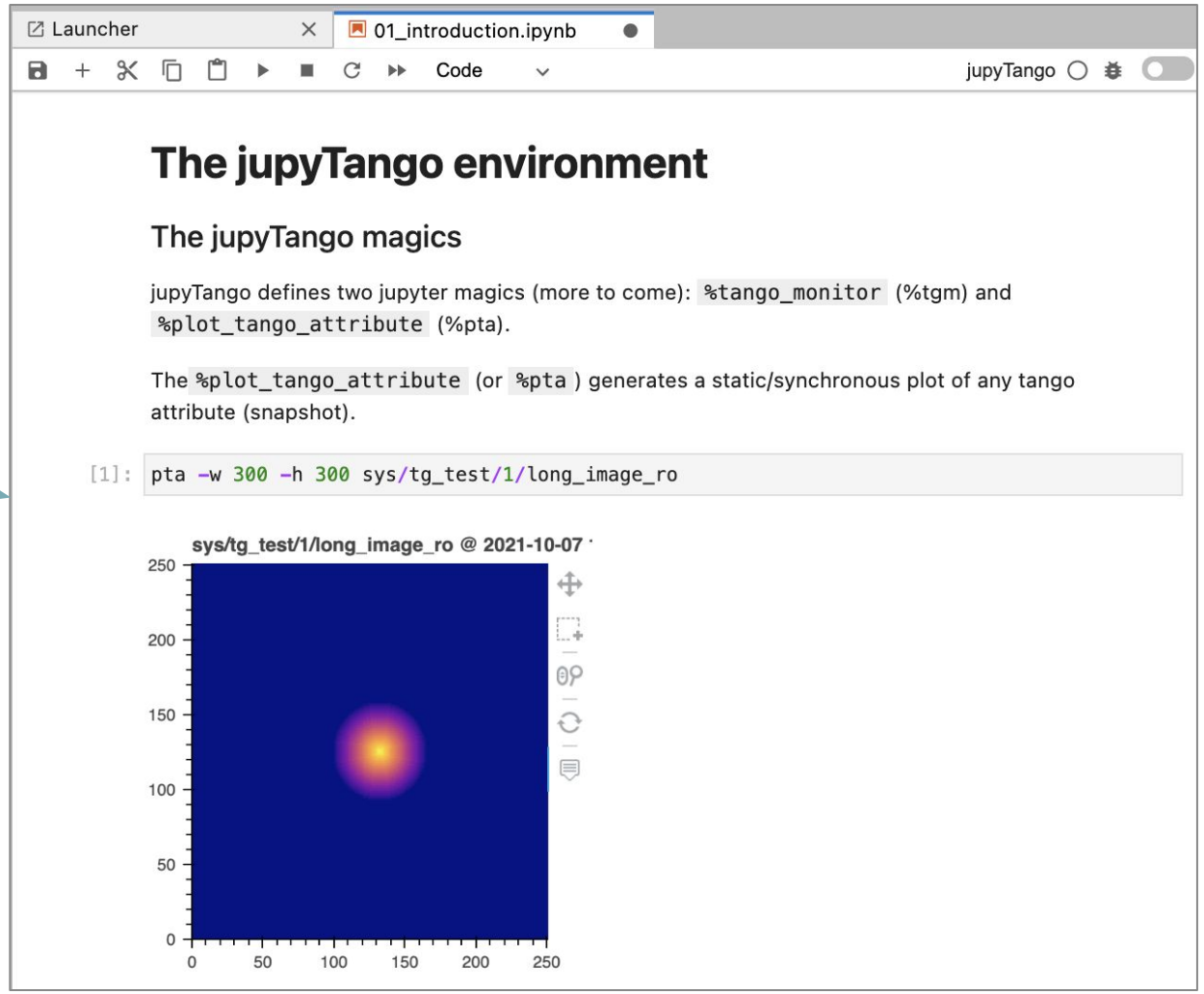
```
[→  docker git:(develop) docker network create jupytango-net
e9690e04aac610379c75c8200766df4dd8f285e31deb5715529224728d7835e2
[→  docker git:(develop) docker-compose up
Starting tangodb ... done
Creating databaseds ... done
Creating tangotest  ... done
Creating jupytango  ... done
Attaching to tangodb, databaseds, tangotest, jupytango
databaseds  | wait-for-it.sh: waiting 30 seconds for tangodb:3306
tangodb     | 2021-10-07 10:00:30+00:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:10.5.11+maria~focal started.
tangotest   | Can't build connection to TANGO database server, exiting
tangodb     | 2021-10-07 10:00:30+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
tangodb     | 2021-10-07 10:00:30+00:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:10.5.11+maria~focal started.
tangodb     | 2021-10-07 10:00:30+00:00 [Note] [Entrypoint]: Initializing database files
```

```
jupytango   | [I 2021-10-07 10:00:32.741 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
jupytango   | [W 2021-10-07 10:00:32.745 ServerApp] No web browser found: could not locate runnable browser.
jupytango   | [C 2021-10-07 10:00:32.746 ServerApp]
jupytango   |
jupytango   |     To access the server, open this file in a browser:
jupytango   |         file:///home/tango/.local/share/jupyter/runtime/jpserver-1-open.html
jupytango   |     Or copy and paste one of these URLs:
jupytango   |         http://046c6790decb:8888/lab?token=afd379cd2d2a3a1d48b6e7010940a96d4c57d43b84f696d1
jupytango   |     or http://127.0.0.1:8888/lab?token=afd379cd2d2a3a1d48b6e7010940a96d4c57d43b84f696d1
```

Open URL with token in your browser

Connect to Jupyter notebook





NOTE:
Dynamic updates to plots (`tgm` and `tango_monitor`) don't work with docker-compose under MacOS!

Additional resources

Useful links

Examples from this presentation

https://gitlab.com/tango-controls/pytango/-/tree/develop/examples/training

PyTango documentation

https://pytango.readthedocs.io

General Tango documentation

https://tango-controls.readthedocs.io

Tango community forum

https://www.tango-controls.org/community/forum/

SKAO Tango Dockerfiles:

https://gitlab.com/ska-telescope/ska-tango-images

SKAO artefact repository, for Docker images:

https://artefact.skao.int

Thanks!

We recognise and acknowledge the indigenous peoples and cultures that have traditionally lived on the lands on which our facilities are located.

SKAO

www.skao.int