GUI Programming with PyQt5

Sompob Shanokprasith

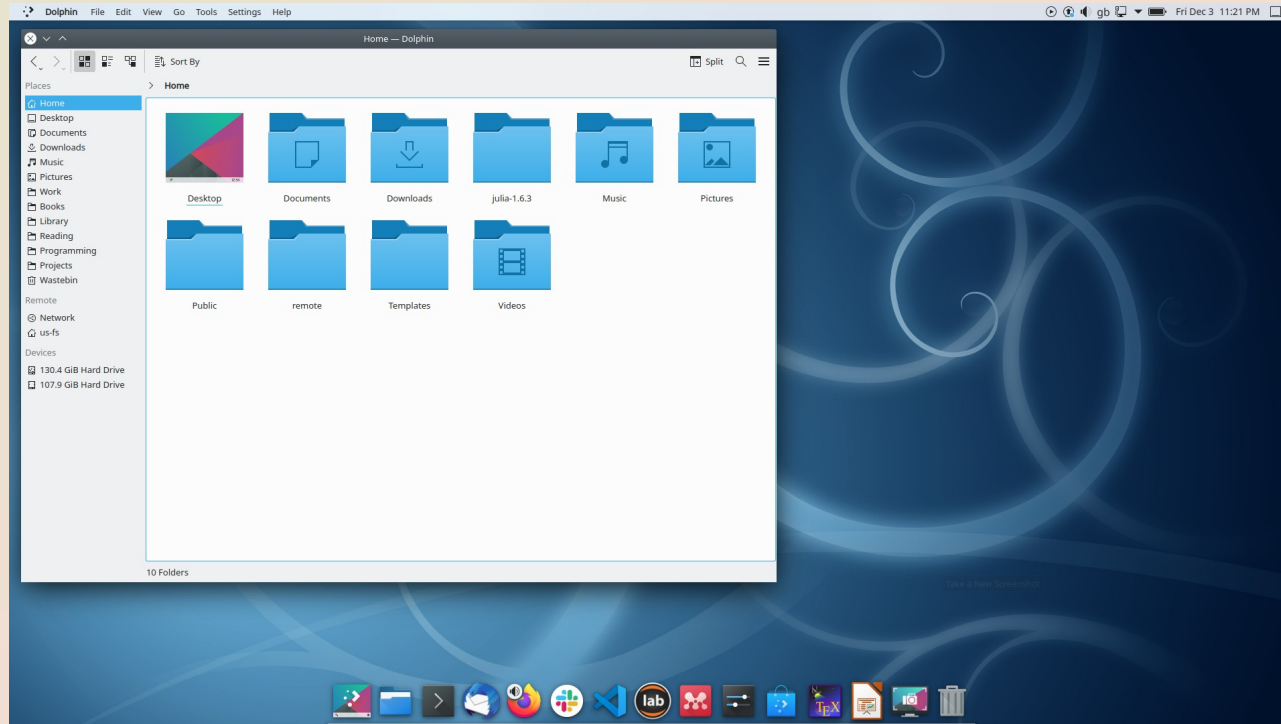Away Day

14 December 2021

Outline

- GUI

- PyQt5

- Model-view-controller

- pyqtgraph

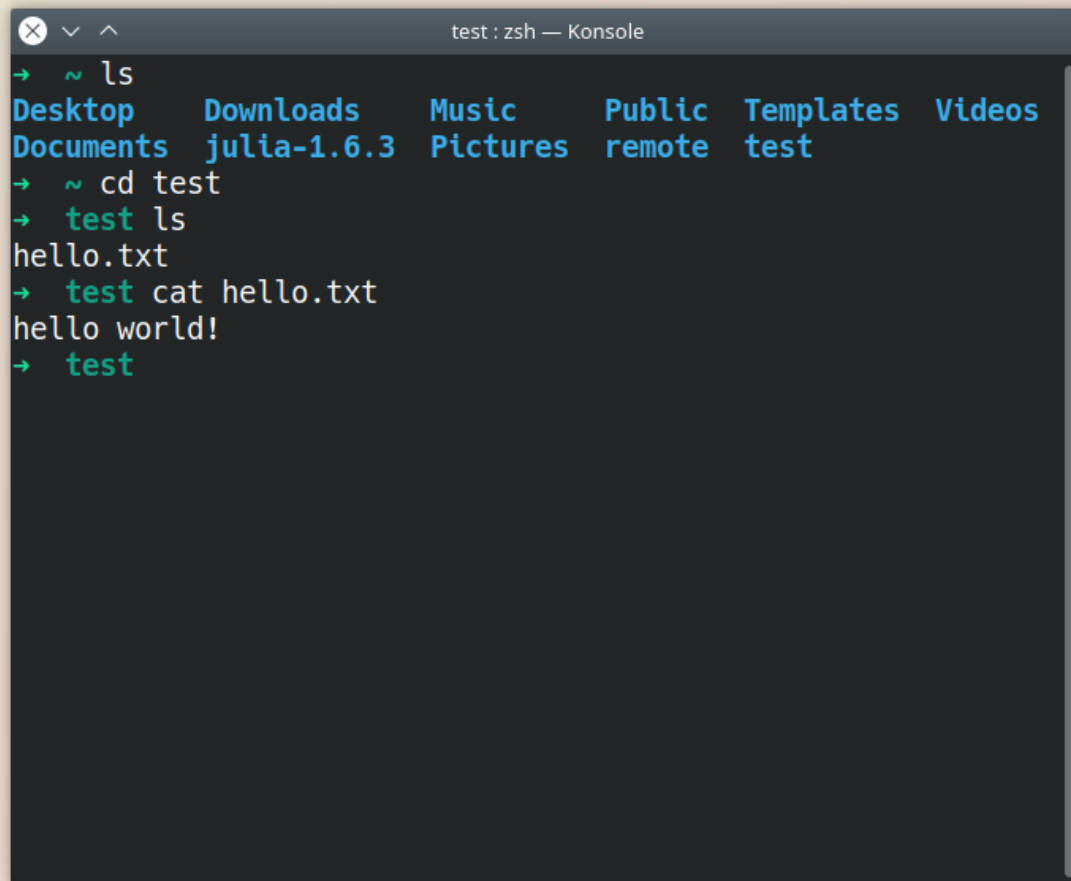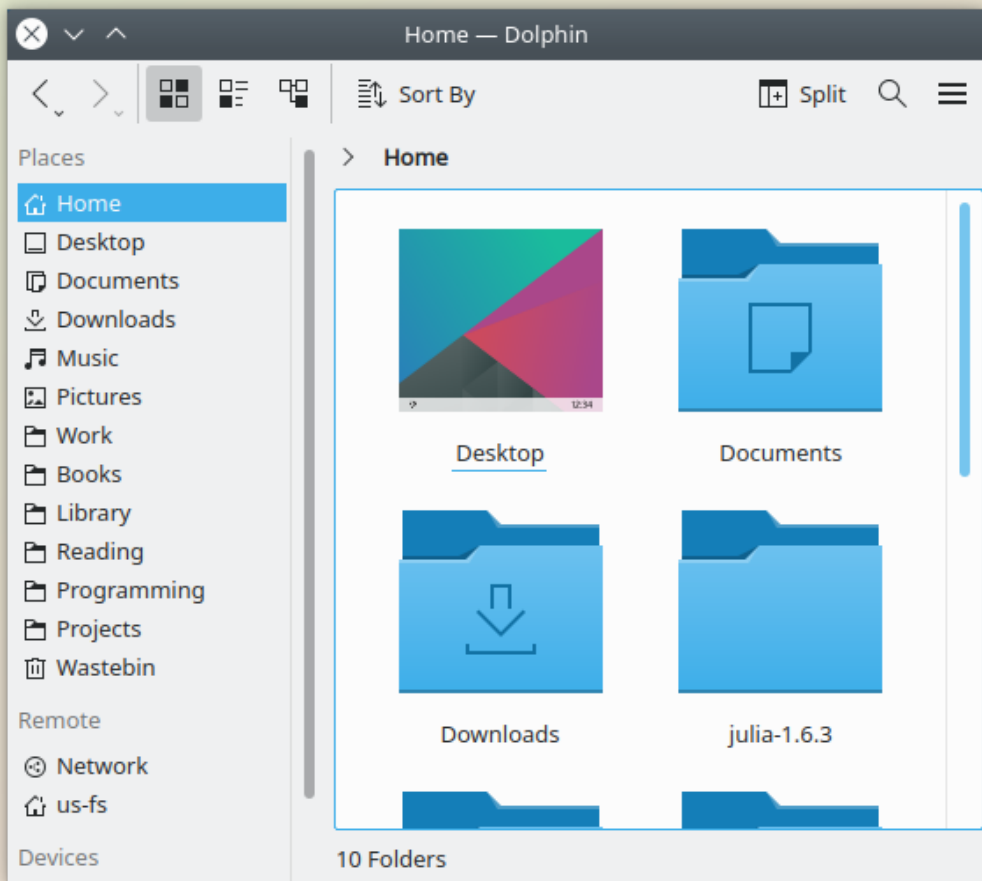- Examples

GUI

- Graphical user interface
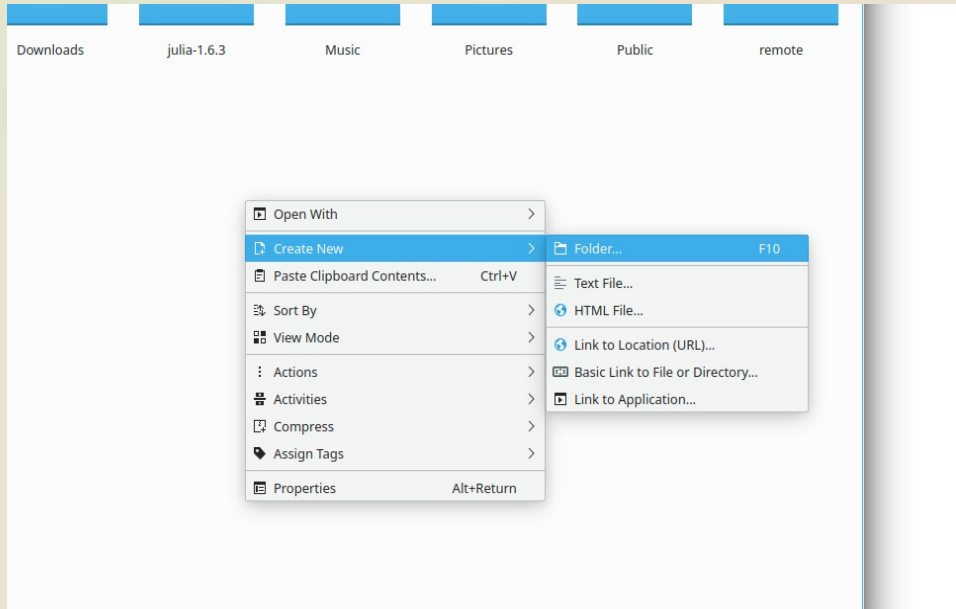
GUI

Pros

- Good looking

- Easy to use

Cons

- Restrictive

- Longer development

Example: GUI vs command line
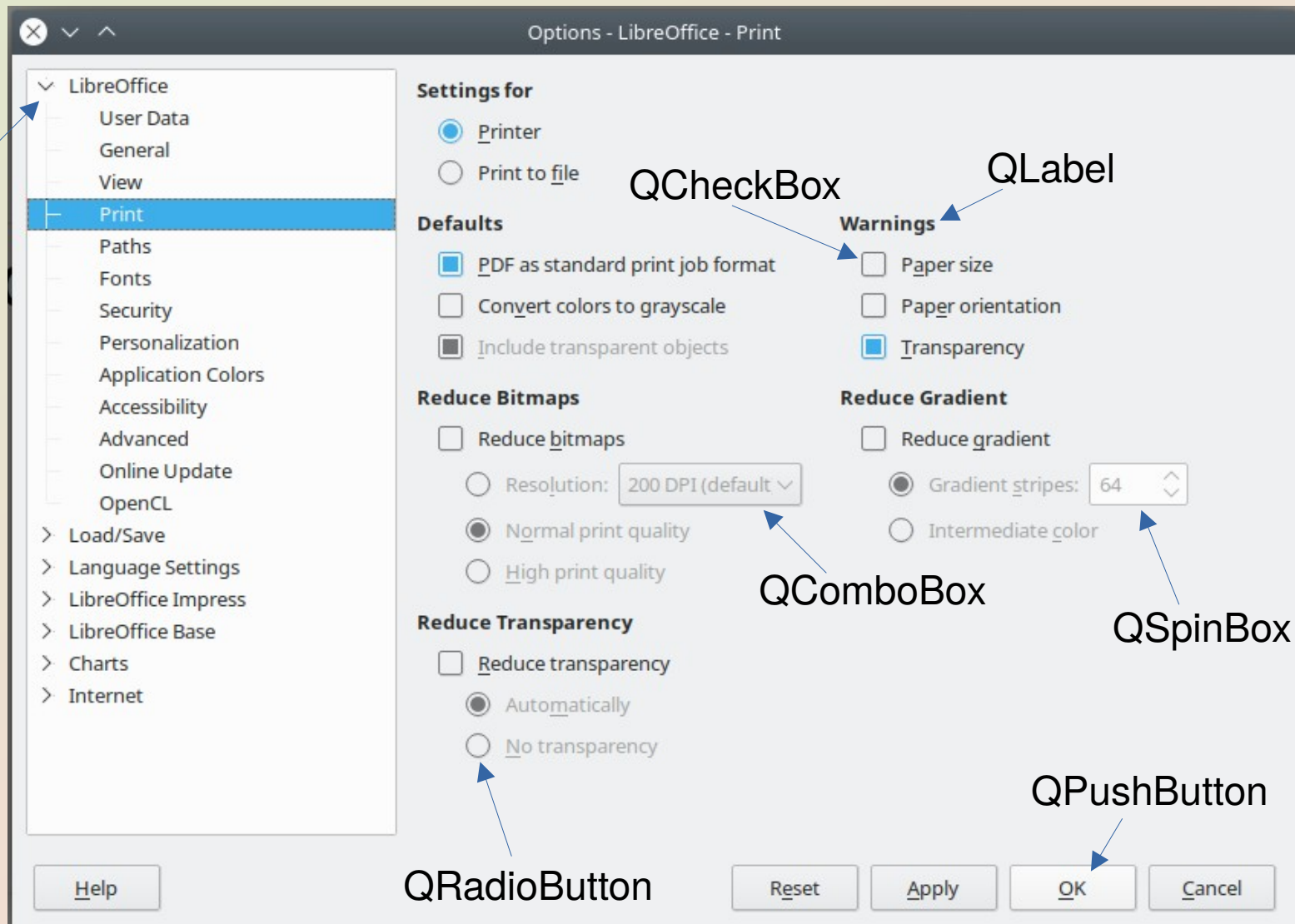
File system navigation

Create folders

PyQt5

- GUI library (Riverbank Computing)

- Python wrapper of Qt 5 (C++, The Qt Company)

- Good documentation for Qt

- PyQt6 (Qt 6) is now available

- Alternatives:

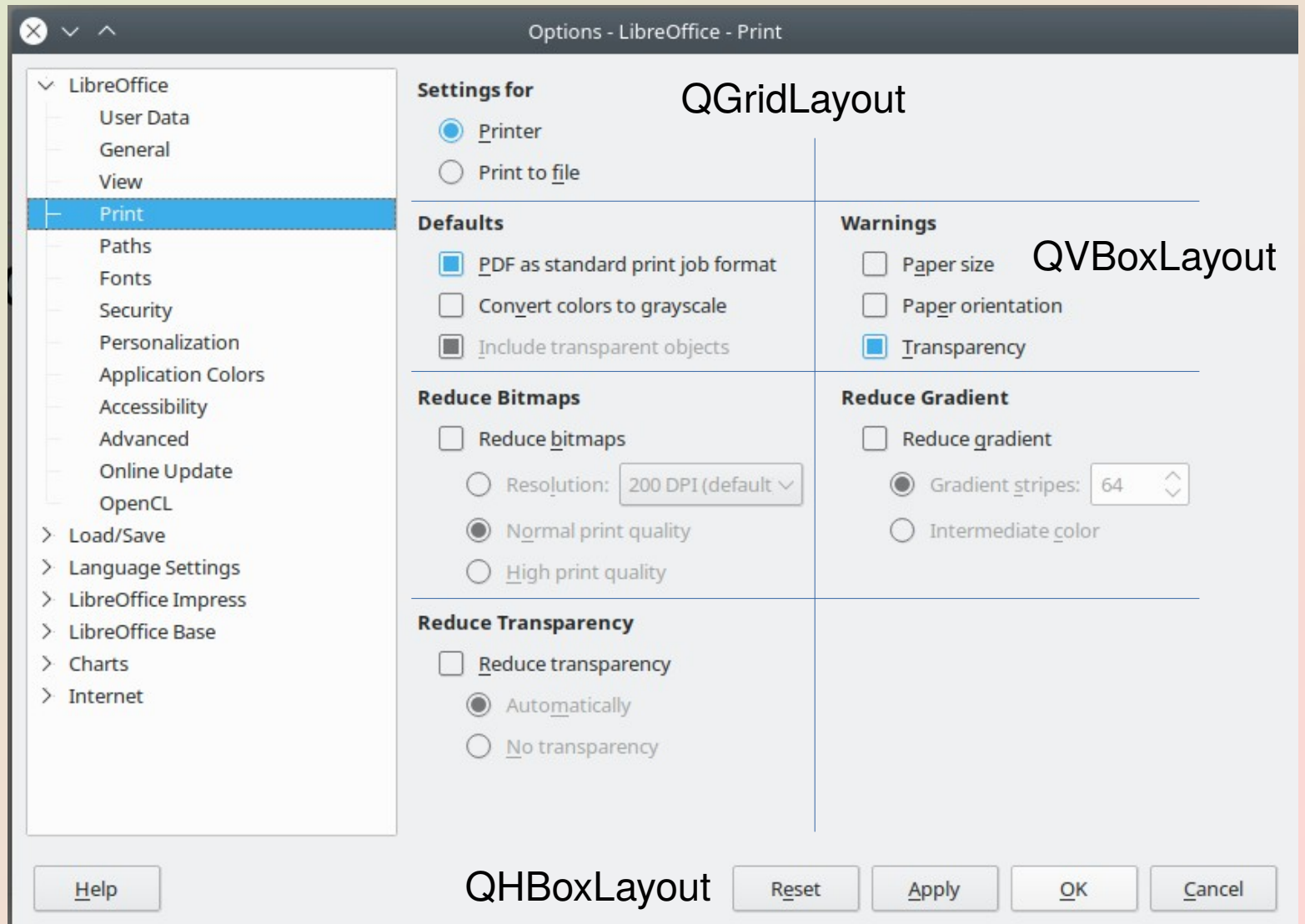   PySide2 (Qt 5), tkinter (Tcl/Tk), wxPython (wxWidgets)

Widgets

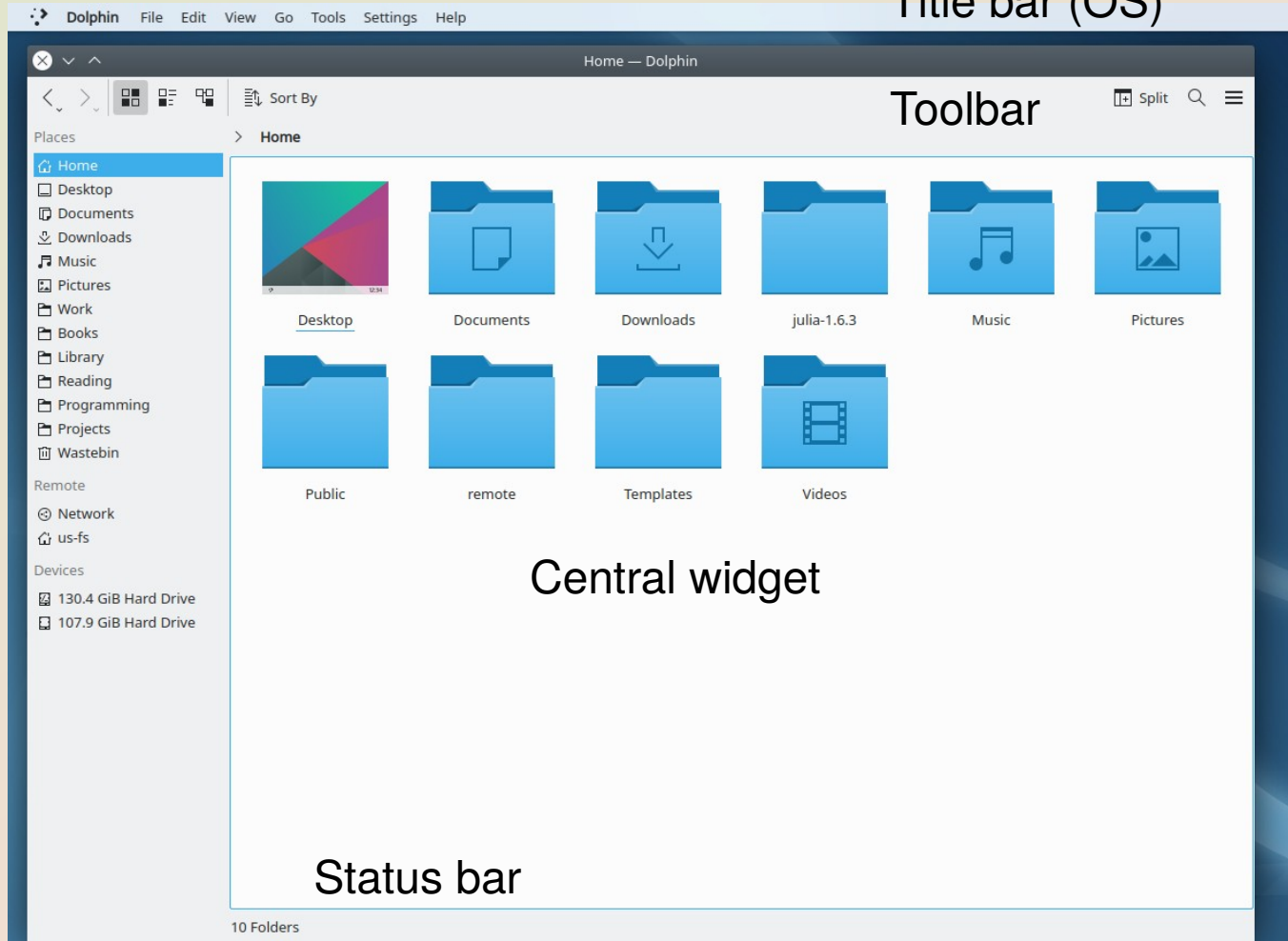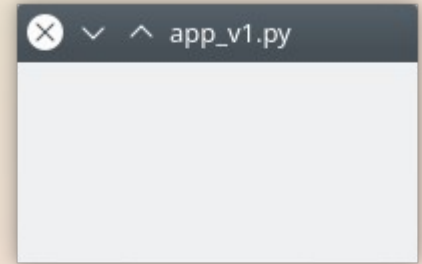Options - LibreOffice - Print

- ∨ LibreOffice
  - User Data
  - General
  - View
  - Print
  - Paths
  - Fonts
  - Security
  - Personalization
  - Application Colors
  - Accessibility
  - Advanced
  - Online Update
  - OpenCL
- › Load/Save
- › Language Settings
- › LibreOffice Impress
- › LibreOffice Base
- › Charts
- › Internet

QTreeWidget
QTabWidget

Settings for
- ● Printer
- ○ Print to file

Defaults
- ☑ PDF as standard print job format
- ☐ Convert colors to grayscale
- ☑ Include transparent objects

Warnings
- ☐ Paper size
- ☐ Paper orientation
- ☑ Transparency

QCheckBox   QLabel

Reduce Bitmaps
- ☐ Reduce bitmaps
  - ○ Resolution: 200 DPI (default ∨)
  - ● Normal print quality
  - ○ High print quality

Reduce Gradient
- ☐ Reduce gradient
  - ● Gradient stripes: 64
  - ○ Intermediate color

QComboBox   QSpinBox

Reduce Transparency
- ☐ Reduce transparency
  - ● Automatically
  - ○ No transparency

QRadioButton

Help   Reset   Apply   OK   Cancel

QPushButton

Layouts

QGridLayout

QVBoxLayout

QHBoxLayout

Main window

Menu bar

Title bar (OS)

Dolphin   File   Edit   View   Go   Tools   Settings   Help

Home — Dolphin

Toolbar

Split

Places

> Home

- Home
- Desktop
- Documents
- Downloads
- Music
- Pictures
- Work
- Books
- Library
- Reading
- Programming
- Projects
- Wastebin

Remote

- Network
- us-fs

Devices

- 130.4 GiB Hard Drive
- 107.9 GiB Hard Drive

Desktop   Documents   Downloads   julia-1.6.3   Music   Pictures

Public   remote   Templates   Videos

Central widget

Status bar

10 Folders

Getting started

- Installation: `pip install PyQt5`

- Code:

```python
import sys
from PyQt5.QtWidgets import *

class MainWindow(QMainWindow): # any QWidget
    pass # add code here

app = QApplication(sys.argv)
m = MainWindow()
m.show()
sys.exit(app.exec_())
```

```python
class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        main_widget = QWidget()
        self.setCentralWidget(main_widget)
        vbox = QVBoxLayout()
        main_widget.setLayout(vbox)

        label = QLabel('Hello World')
        vbox.addWidget(label)

        self.setGeometry(100, 100, 300, 100)
        self.setWindowTitle('First Program')
```
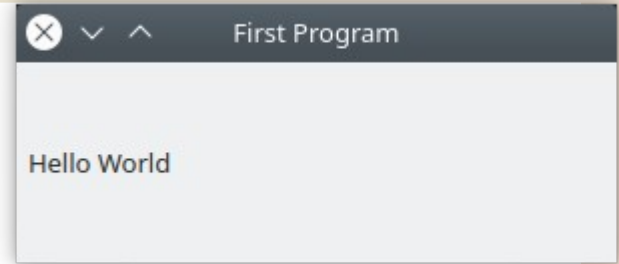
Contents

Properties

Public Functions

Reimplemented Public
Functions

Reimplemented Protected
Functions

Detailed Description

QGridLayout Class

The QGridLayout class lays out widgets in a grid. More...

Header:	#include <QGridLayout>
qmake:	QT += widgets
Inherits:	QLayout

› List of all members, including inherited members

| void	**addLayout**(QLayout *layout, int row, int column, Qt::Alignment alignment = Qt::Alignment())
void	**addLayout**(QLayout *layout, int row, int column, int rowSpan, int columnSpan, Qt::Alignment alignment = Qt::Alignment())
void	**addWidget**(QWidget *widget, int row, int column, Qt::Alignment alignment = Qt::Alignment())
void	**addWidget**(QWidget *widget, int fromRow, int fromColumn, int rowSpan, int columnSpan, Qt::Alignment alignment = Qt::Alignment())
QRect	**cellRect**(int row, int column) const

Signals and slots

- How can objects communicate with each other?
  - Button clicked
  - Item selected
  - Text edited
- Object 1 emits signal when its state changes
- Object 2 has a slot (function)
- Connect signal to slot

```python
# inside __init__ definition
    self.label = QLabel('Click button to change text')
    vbox.addWidget(self.label)
    button = QPushButton('Click me')
    vbox.addWidget(button)

    # connect predefined signal
    button.clicked.connect(self.on_button_clicked)

def on_button_clicked(self): # slot
    self.label.setText('Button clicked')
```
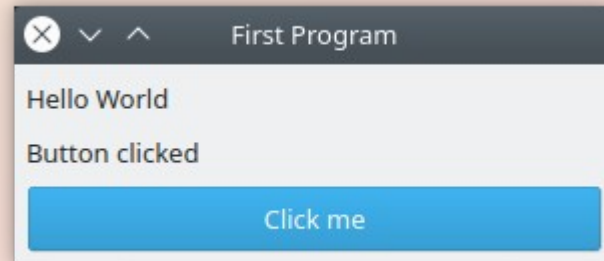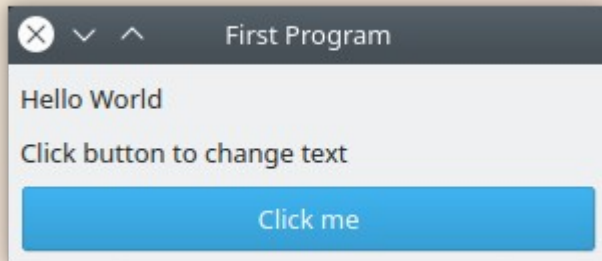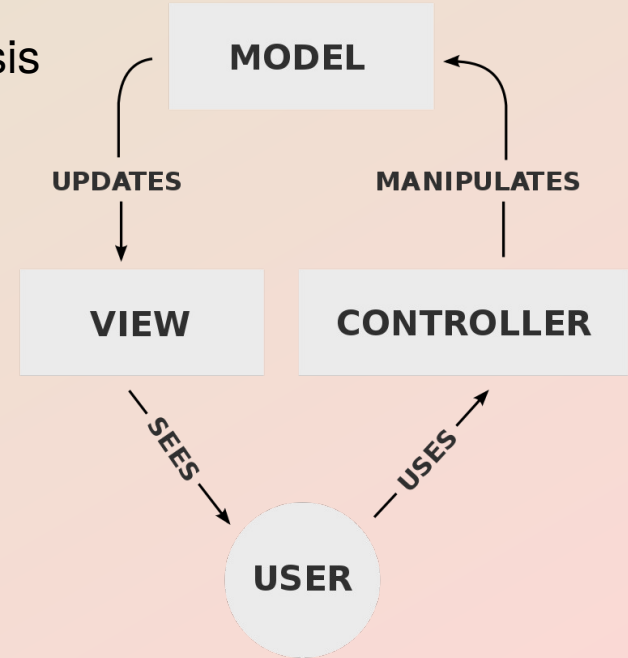
Model-view-controller

- For physicists - simulation, data acquisition/visualisation/analysis

- Larger program easily leads to spaghetti code

- Need design pattern

- Independent components

  - Engine (Model)

    - Perform calculation

    - Load data

  - GUI (View)

    - Plot results (View)

    - Adjust plotting parameters (Controller)

   - Adjust model parameters (Controller)

Example: https://codeshare.phy.cam.ac.uk/amop-mbqd/kagome/experiment-control/timing/qcontrol3/qcontrol3_gui_qt
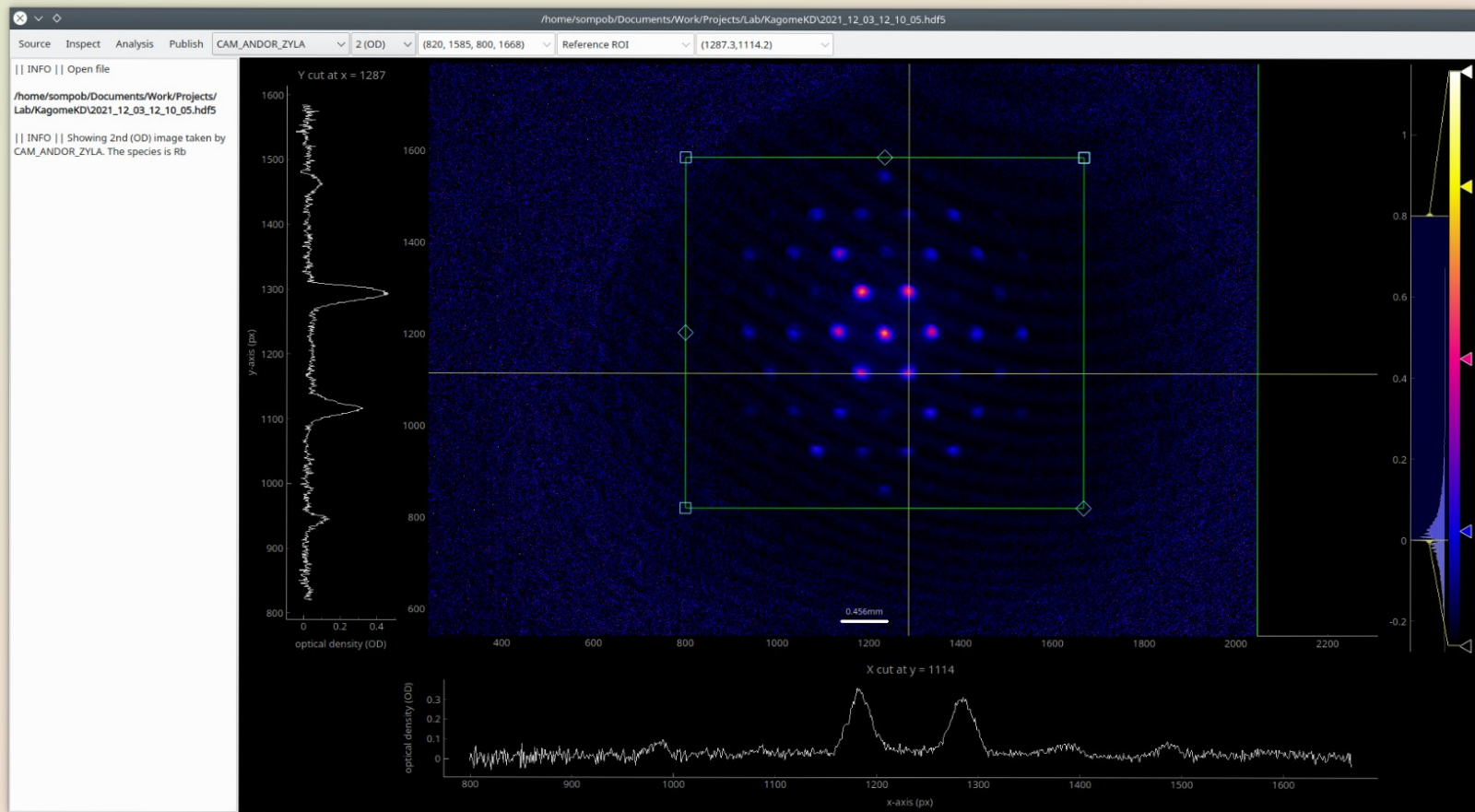
pyqtgraph

- Plotting library compatible with PyQt5

- Fast and interactive

  - Data update

  - Zoom in/out

  - ROI (region of interest)

- Need to be careful with changes between versions

- Poor documentation, but many examples
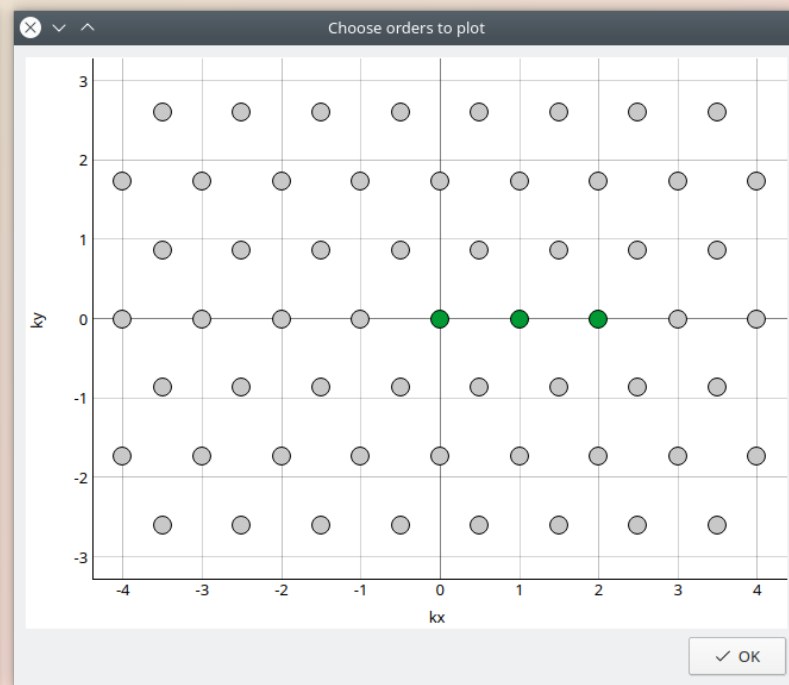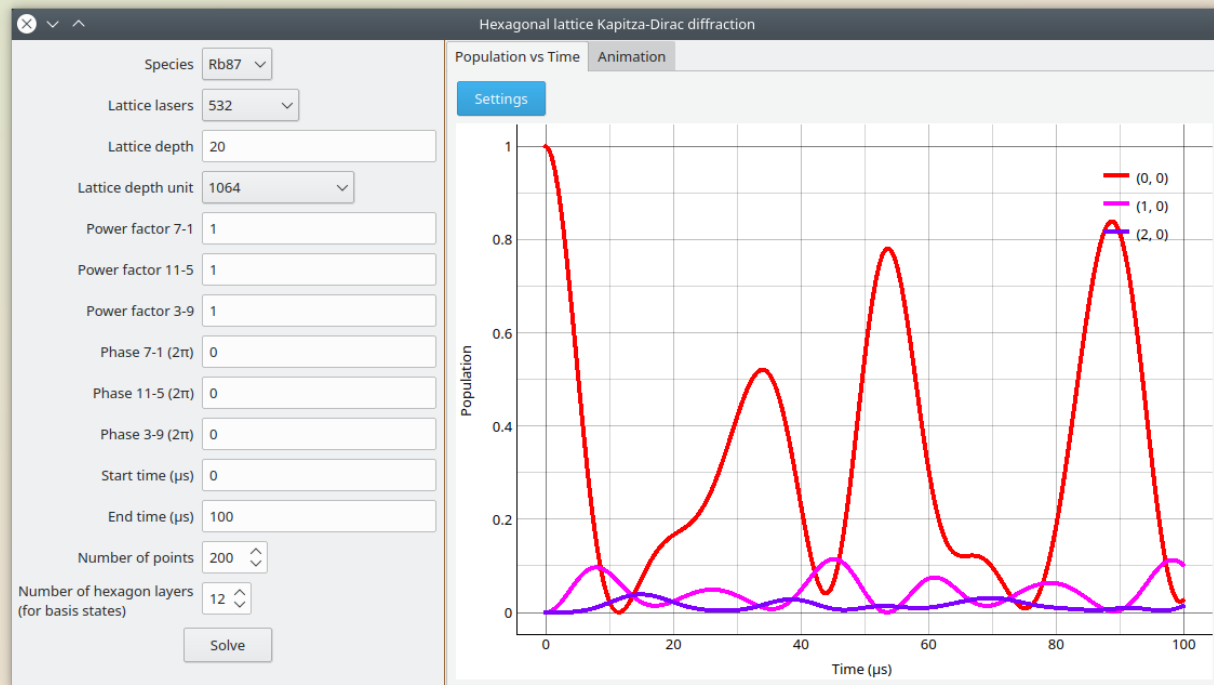
  - `python3 -m pyqtgraph.examples`
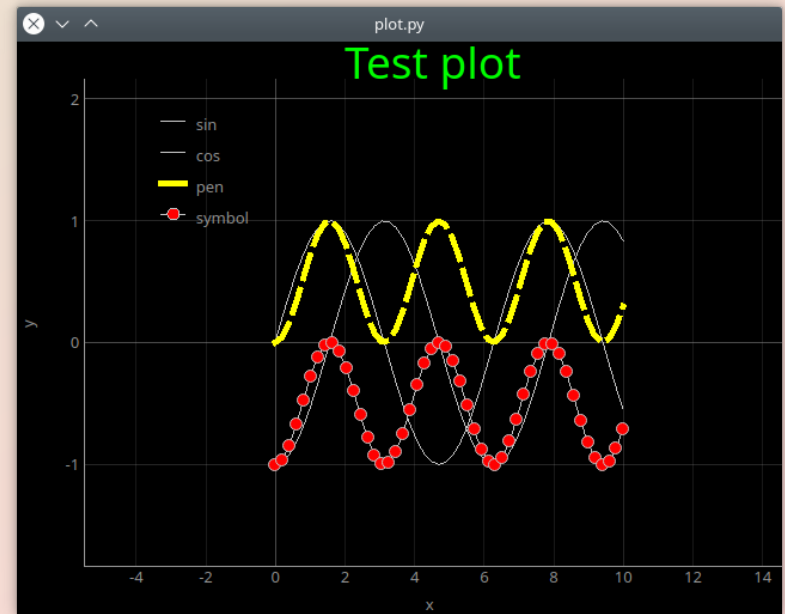
Example – beamprofiler
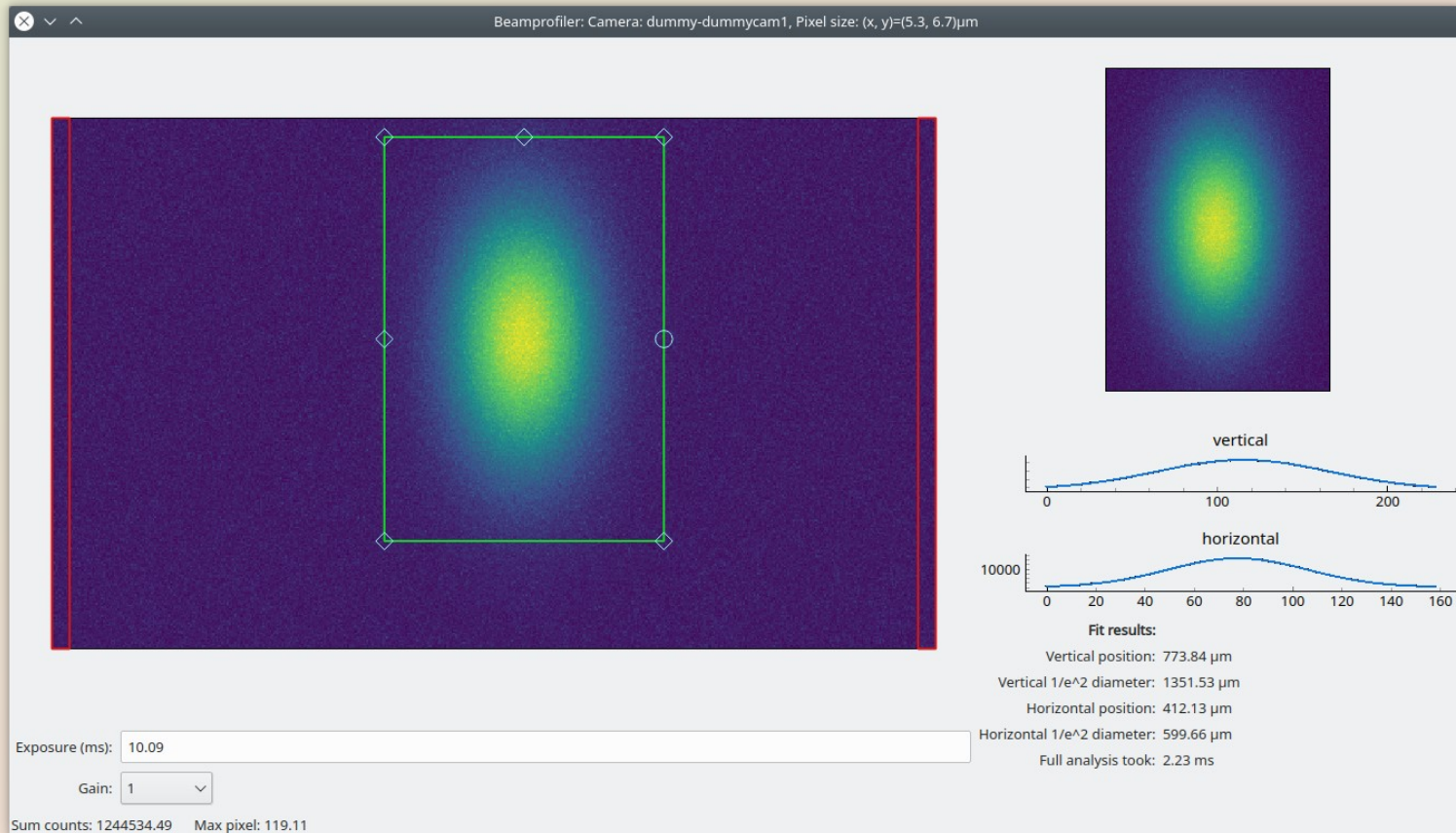
Example – Analysis GUI

Example – Kapitza-Dirac diffraction

Example – Kapitza-Dirac diffraction