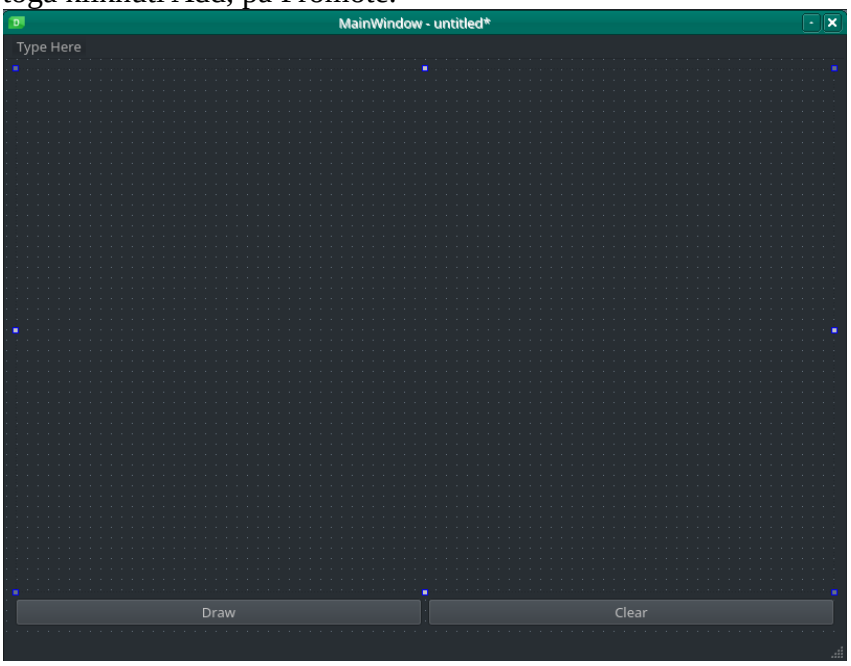
Primer 1: Na MainWindow dodati u grid layout dva tastera i widget.
Dodati Widget promovisati desni klik → promote to → Promoted class name: PlotWidget, Header file: pyqtgraph. Nakon toga kliknuti Add, pa Promote.



Konvertovati ui u py fajl (u ovom slučaju naziv fajla p1_uic.py)

```
import sys
from PySide6.QtWidgets import QApplication, QMainWindow
from p1_uic import Ui_MainWindow
from numpy.random import uniform

class MainWindow(QMainWindow):
      def __init__(self):
            super(MainWindow, self).__init__()
            self.ui = Ui_MainWindow()
            self.ui.setupUi(self)

            self.ui.pushButton.clicked.connect(self.draw)
            self.ui.pushButton_2.clicked.connect(self.clear)

      def draw(self):
            x = uniform(-1,1,1000)
            y = uniform(-1,1,1000)
            self.ui.widget.plot(x,y, pen=None, symbolSize=1, symbolPen='g',
            symbol='x') ## setting pen=None disables line drawing)

      def clear(self):
            self.ui.widget.clear()

if __name__ == "__main__":
      app = QApplication(sys.argv)
```

```
    window = MainWindow()
    window.show()

    sys.exit(app.exec())
```

Primer 2: Identičan izgled aplikacije kao i u prethodnom primeru samo izmeniti imena tasterima Start i Stop za početak i kraj komunikacije.

```
import sys
from PySide6 import QtCore, QtGui, QtSerialPort
from PySide6.QtWidgets import QApplication, QGraphicsScene, QGraphicsView,
QMainWindow
from p1_uic import Ui_MainWindow
import numpy as np

class MainWindow(QMainWindow):
    def __init__(self):
        super(MainWindow, self).__init__()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.x = np.zeros(100)
        self.y = np.zeros(100)
        self.ui.pushButton.clicked.connect(self.start_com)
        self.ui.pushButton_2.clicked.connect(self.close_com)
        self.serial = QtSerialPort.QSerialPort('/dev/ttyACM1',
        baudRate=QtSerialPort.QserialPort.Baud115200,
        readyRead=self.receive)

    def receive(self):
        while self.serial.canReadLine():
            self.ui.widget.clear()
            text = self.serial.readLine().data().decode()
            text = text.rstrip('\n')
            self.x = np.arange(len(self.y))
            self.y = np.roll(self.y, -1)
            self.y[-1] = (int(text))

            self.ui.widget.plot(self.x,self.y, pen='g', symbolSize=1,
            symbolPen='g', symbol='x')
            #QApplication.processEvents()

    def start_com(self):
        if not self.serial.isOpen():
            self.ui.pushButton.setDisabled(True)
            self.ui.pushButton_2.setDisabled(False)
            self.serial.open(QtCore.QIODevice.ReadWrite)
```

```
    def close_com(self):
        self.serial.close()
        self.ui.pushButton_2.setDisabled(True)
        self.ui.pushButton.setDisabled(False)


if __name__ == "__main__":
    app = QApplication(sys.argv)

    window = MainWindow()
    window.show()

    sys.exit(app.exec())
```

Primer 3: Identičan primer kao prošli samo korišćenjem pyserial-a

```
import sys
from PySide6 import QtSerialPort
from PySide6.QtCore import QIODevice, QRunnable, QObject, QThread,
Signal
from PySide6.QtWidgets import QApplication, QMainWindow
from p1_uic import Ui_MainWindow
import numpy as np
import serial

class MainWindow(QMainWindow):
    def __init__(self):
        super(MainWindow, self).__init__()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        self.x = np.zeros(100)

        self.ui.pushButton.clicked.connect(self.start_com)
        self.ui.pushButton_2.clicked.connect(self.close_com)
        self.th = Worker()

        self.th.signalData.connect(self.draw)

    def draw(self,y):
        self.ui.widget.clear()
        x = np.arange(len(y))
        self.ui.widget.plot(x,y, pen='g', symbolSize=1,
symbolPen='g', symbol='x')



    def start_com(self):
        self.th.start()
        self.th.running = True
        self.ui.pushButton.setDisabled(True)
```

```
        self.ui.pushButton_2.setDisabled(False)

    def close_com(self):
        self.th.running = False
        self.th.serial.close()
        self.ui.pushButton_2.setDisabled(True)
        self.ui.pushButton.setDisabled(False)

class Worker(QThread):
    signalData = Signal(np.ndarray)
    def __init__(self):
        super().__init__()
        self.running = True

    def run(self):
        self.serial = serial.Serial('/dev/ttyACM1', 115200,
                                    timeout=0.1)
        y = np.zeros(100)
        while self.running:
            try:
                text = self.serial.readline().decode()
                text = text.rstrip('\n')

                y = np.roll(y, -1)
                y[-1] = (int(text))
                self.signalData.emit(y)
            except:
                print(text)


if __name__ == "__main__":
    app = QApplication(sys.argv)

    window = MainWindow()
    window.show()

    sys.exit(app.exec())
```

Kod za ploču:

```
  char txt[5];
  while (1)
  {
        uint8_t random_number = rand()&0x00FF;
        sprintf(txt, "%hu\n",random_number);
        HAL_UART_Transmit(&huart2, txt, strlen(txt), HAL_MAX_DELAY);
        HAL_Delay(10);
  }
```

Primer 4: Crtanje dva grafika

Kontroler kod:

```
  char txt[10];
  while (1)
  {
       for(uint16_t i=0;i<100;i++){
            float random_number = (rand()/1073741823.5)-1;
            float sin_val = sin(i*2*3.14*0.01);
            sprintf(txt, "%.3f,%.3f\n",random_number,sin_val);
            HAL_UART_Transmit(&huart2, txt, strlen(txt),
HAL_MAX_DELAY);
            HAL_Delay(10);
       }
  }
```

PySide code:

```
import sys
from PySide6.QtCore import QThread, Signal
from PySide6.QtWidgets import QApplication, QMainWindow
from p1_uic import Ui_MainWindow
import numpy as np
import serial

class MainWindow(QMainWindow):
    def __init__(self):
        super(MainWindow, self).__init__()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)

        self.ui.pushButton.clicked.connect(self.start_com)
        self.ui.pushButton_2.clicked.connect(self.close_com)
        self.th = Worker()
        self.th.signalData.connect(self.draw)

    def draw(self,y1,y2):
        self.ui.widget.clear()
        x = np.arange(len(y1))
        self.ui.widget.plot(x,y1, pen='g', symbolSize=1,
symbolPen='g', symbol='x')
        self.ui.widget.plot(x,y2, pen='b', symbolSize=1,
symbolPen='g', symbol='o')

    def start_com(self):
        self.th.start()
        self.th.running = True
        self.ui.pushButton.setDisabled(True)
        self.ui.pushButton_2.setDisabled(False)

    def close_com(self):
        self.th.running = False
```

```
        self.th.serial.close()
        self.ui.pushButton_2.setDisabled(True)
        self.ui.pushButton.setDisabled(False)

class Worker(QThread):
    signalData = Signal(np.ndarray,np.ndarray)
    def __init__(self):
        super().__init__()
        self.running = True

    def run(self):
        self.serial = serial.Serial('/dev/ttyACM1', 115200,
timeout=0.1)
        y1 = np.zeros(100)
        y2 = np.zeros(100)
        while self.running:
            try:
                text = self.serial.readline().decode()
                text = text.rstrip('\n').split(',')
                if len(text) == 2:
                    y1 = np.roll(y1, -1)
                    y1[-1] = (float(text[0]))
                    y2 = np.roll(y2, -1)
                    y2[-1] = (float(text[1]))
                    self.signalData.emit(y1,y2)
            except:
                print(text)


if __name__ == "__main__":
    app = QApplication(sys.argv)

    window = MainWindow()
    window.show()

    sys.exit(app.exec())
```