

**MODUL PEMBUATAN GRAPHICAL USER INTERFACE (GUI) ARDUINO
MENGUNAKAN PYTHON DAN PYQT5**



Disusun oleh:

Muhammad Husni Muttaqin, S. Pd.

Diky Zakaria, S.Pd., M.T.

Galura Muhammad Suranegara, S.Pd., M.T.

PROGRAM STUDI MEKATRONIKA DAN KECERDASAN BUATAN

UPI KAMPUS DAERAH PURWAKARTA

AGUSTUS 2022

DAFTAR ISI

DAFTAR ISI	2
DAFTAR GAMBAR	4
DAFTAR TABEL.....	5
BAB 1 PENDAHULUAN.....	6
A. Latar Belakang	6
B. Graphical User Interface (GUI).....	8
C. Sekilas Tentang Qt.....	9
D. PyQt.....	9
E. GUI yang Akan Dibuat pada Modul ini	10
BAB 2 Software yang Dibutuhkan	11
A. Thonny IDE	11
B. Notepad++	13
C. Arduino IDE & drivernya.....	14
BAB 3 Membuat GUI Sederhana Menggunakan PyQt5	16
A. Program minimal untuk menjalankan PyQt5	16
B. Membuat Tampilan Windows	19
1. Program Window default.....	20
2. Program Window Fullscreen	21
3. Program window dengan lebar dan tinggi fix tidak dapat diubah ubah	21
4. Program Window hanya title dan tombol close	23
5. Program Window hanya judul	24
6. Program Window polos tanpa judul dan tombol tombol	25
C. Menampilkan Teks	25

D. Membuat Tombol / Button	28
E. Memasukkan Gambar	30
F. Membuat Slider	33
G. Membuat Gauge	35
H. Membuat Rectangle	38
I. Menggabungkan Seluruh Fungsi	38
BAB 4 INTEGRASI DENGAN ARDUINO	40
A. Apa itu Arduino	40
B. Komunikasi Serial pada Arduino	40
C. Parsing data	40
D. I/O digital dan analog	43
E. Mengendalikan LED via PyQt5	43
References	52



DAFTAR GAMBAR

Gambar 1. Bahasa pemrograman populer digunakan di dunia.	6
Gambar 2. List peringkat bahasa pemrograman paling populer di dunia berdasarkan 4 industri programming besar.....	7
Gambar 3. GUI Windows 10.	9
Gambar 4. GUI yang akan dibuat.....	10
Gambar 5. Tampilan laman thonny.org	11
Gambar 6. Tampilan awal Thonny IDE.	12
Gambar 7. Tampilan web notepad++ download.....	13
Gambar 8. Tampilan awal notepad++.	14
Gambar 9. Mendownload Arduino IDE.....	14
Gambar 10. Tampilan awal Arduino IDE.	15
Gambar 11. Ilustrasi python dan QML.....	16
Gambar 12. File main.py dan main.qml disimpan dalam 1 folder "Project".	17
Gambar 13. Program dasar untuk membuat windows.	19
Gambar 14. Properties dalam membuat windows.....	19
Gambar 15. Window default.....	21
Gambar 16. Window dengan ukuran yang fixed.	22
Gambar 17. Window hanya title dan tombol close	23
Gambar 18. Window hanya title.	24
Gambar 19. Window tanpa title dan tombol close.	25
Gambar 20. Tampilan menambahkan teks dalam window.....	28
Gambar 21. File gambar diletakkan di folder yang sama.	31
Gambar 22. Memasukkan gambar pada window.	32
Gambar 23. Tampilan slider dalam sebuah window.....	35
Gambar 24. Gauge and circular gauge pada window.	38
Gambar 25. GUI sederhana menggunakan seluruh fungsi.	39

DAFTAR TABEL

Tabel 1. 7 properties dasar membuat Window.....	20
Tabel 2. Properties dalam membuat teks.....	26
Tabel 3. Properties dasar membuat tombol / button.....	29
Tabel 4. Properties dasar memasukkan gambar.....	31
Tabel 5. Properties dasar slider.	33
Tabel 6. Properties gauge / circular gauge.....	35

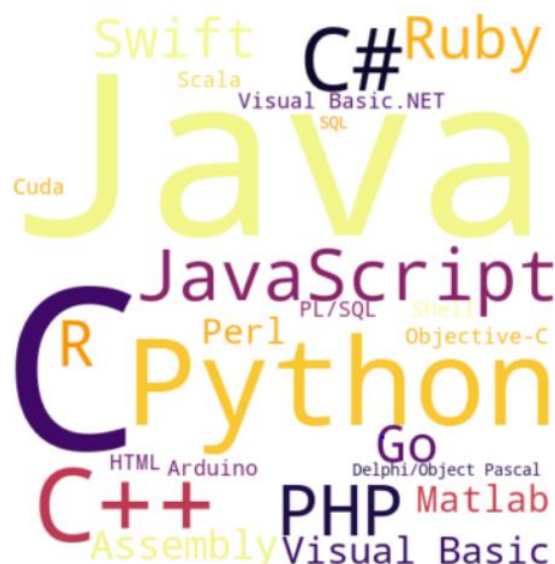


BAB 1 PENDAHULUAN

A. Latar Belakang

Ilmu dasar seperti matematika, statistik dan pemrograman merupakan beberapa cabang ilmu yang dibutuhkan di dunia industri [1]. Untuk menyelesaikan persoalan matematika dan statistik [2], kita bisa menggunakan bahasa pemrograman yang telah ada. Tentunya kita harus menguasai dulu algoritma pemrograman sebelum dapat mengaplikasikan matematika dan statistik ke dalam bahasa pemrograman. Salah satu bahasa pemrograman yang sedang naik daun saat ini adalah python.

Python merupakan bahasa pemrograman tingkat tinggi yang dibuat oleh Guido van Rossum pada tahun 1980 [3][4]. Karena merupakan bahasa pemrograman tingkat tinggi (bukan bahasa mesin), python lebih mudah dipahami bagi pemula [5]. Saat ini, Python adalah bahasa pemrograman computer yang sangat luas digunakan di berbagai macam aplikasi bidang ilmu seperti otomasi, big data, data science, computer graphic, kalkulus, artificial intelligence dan web development [6]. Seiring berjalannya waktu, saat ini python merupakan bahasa pemrograman yang paling diminati di dunia setelah Bahasa C dan Java [7]. Gambar 1 merupakan bahasa pemrograman yang populer digunakan di dunia.



Gambar 1. Bahasa pemrograman populer digunakan di dunia.

Pemeringkatan lebih detail terkait popularitas bahasa pemrograman di dunia berdasarkan data dari TIOBE index [8], IEEE Spectrum [9], Github [10] dan Stack Overflow [11] terlihat pada gambar 2.

TIOBE	IEEE Spectrum	Github	Stack Overflow
Java	Python	JavaScript	JavaScript
C	C	Python	SQL
C++	Java	Java	Java
Python	C++	Ruby	C#
C#	C#	PHP	Python
Visual Basic.NET	R	C++	PHP
PHP	JavaScript	CSS	C++
JavaScript	PHP	C#	C
Delphi/Object Pascal	Go	Go	TypeScript
Ruby	Swift	C	Ruby
SQL	Arduino	TypeScript	Swift
Visual Basic	Ruby	Shell	Objective-C
R	Assembly	Swift	Visual Basic.NET
PL/SQL	Matlab	Scala	Assembly
Assembly	Scala	Objective-C	R
Swift	HTML	–	Perl
Perl	Shell	–	VBA
Go	Perl	–	Matlab
Matlab	Visual Basic	–	Go
Objective-C	Cuda	–	Scala

Gambar 2. List peringkat bahasa pemrograman paling populer di dunia berdasarkan 4 industri programming besar.

Pada gambar 2, terlihat Python menduduki peringkat top 5 di semua survey yang dilakukan 4 industri programming besar dunia. Hal ini bukan terjadi secara tiba-tiba. Python memiliki beberapa keuntungan bagi penggunaannya yaitu [7]:

1. Bahasa yang digunakan sederhana dan mudah dipahami
2. Cross platform. Python dapat digunakan secara sama di seluruh platform sistem operasi seperti windows, linux dan MacOS.
3. Gratis dan open source. Berbeda dengan bahasa pemrograman lain yang memerlukan lisensi untuk menggunakan fitur-fiturnya seperti MATLAB, python ini gratis dan open source.
4. Digunakan oleh perusahaan besar dunia. Perusahaan besar seperti Youtube, Google, Facebook, dan IBM menggunakan python. Jika kita menguasai python, maka kita memiliki kesempatan besar untuk berkarir di perusahaan besar tersebut.

Berdasarkan latar belakang diatas, maka menjadi penting bagi para guru dan siswa SMK untuk dikenalkan dengan Bahasa pemrograman python sehingga bisa mengikuti tren industri saat ini. Program pengabdian kepada masyarakat ini akan

menargetkan siswa dan guru yang berasal dari SMK di Kabupaten Purwakarta, Jawa Barat. Bagi siswa, pengenalan pemrograman python ini dapat menambah kompetensi siswa di bidang pemrograman dan dapat mengikuti tren bahasa pemrograman terkini. Bagi guru, kegiatan pengabdian ini dapat menarik minat guru untuk terus update materi dan beradaptasi dengan perkembangan materi berkaitan dengan pemrograman.

Rencana pelatihan python ini akan dilakukan secara daring dan simulasi. Topik pembahasan yang akan disampaikan sebagai berikut:

1. Pengenalan dasar bahasa pemrograman python dan PyQt5
2. Pengenalan Graphical User Interface (GUI) sederhana menggunakan python dan PyQt5
3. Praktek membuat GUI sederhana menggunakan python dan PyQt5. Software yang digunakan adalah **Thonny IDE** dan **notepad++**.
4. Integrasi GUI yang telah dibuat dengan hardware Arduino Uno (on off LED Arduino Uno dengan GUI yang telah dibuat.

B. Graphical User Interface (GUI)

GUI adalah singkatan dari Graphical User Interface. GUI diciptakan untuk mempermudah interaksi pengguna dalam menggunakan perangkat [12]. Sebelum GUI diciptakan pengguna harus menggunakan CLI (command line interface) yang merupakan perintah perintah berupa teks untuk menggunakan sebuah perangkat. GUI mempermudah pengguna menampilkan data secara visual untuk pengguna dalam mengoperasikan program karena dapat menghubungkan pengguna dengan bahasa pemrograman pada sebuah program di perangkat. GUI pada umumnya terdiri dari tampilan *widget layouts*, *forms*, *hypertext*, *toolbars*, *windows*, dan *menu systems* [13]. Berikut adalah contoh GUI:



Gambar 3. GUI Windows 10.

Pada gambar 3, terlihat tampilan antar muka dari Windows 10 yang mana setiap icon yang ada mewakili sebuah fungsi tersendiri saat kita klik. Hal ini memudahkan kita dalam memilih fungsi karena setiap fungsi disertai gambar dan teks.

C. Sekilas Tentang Qt

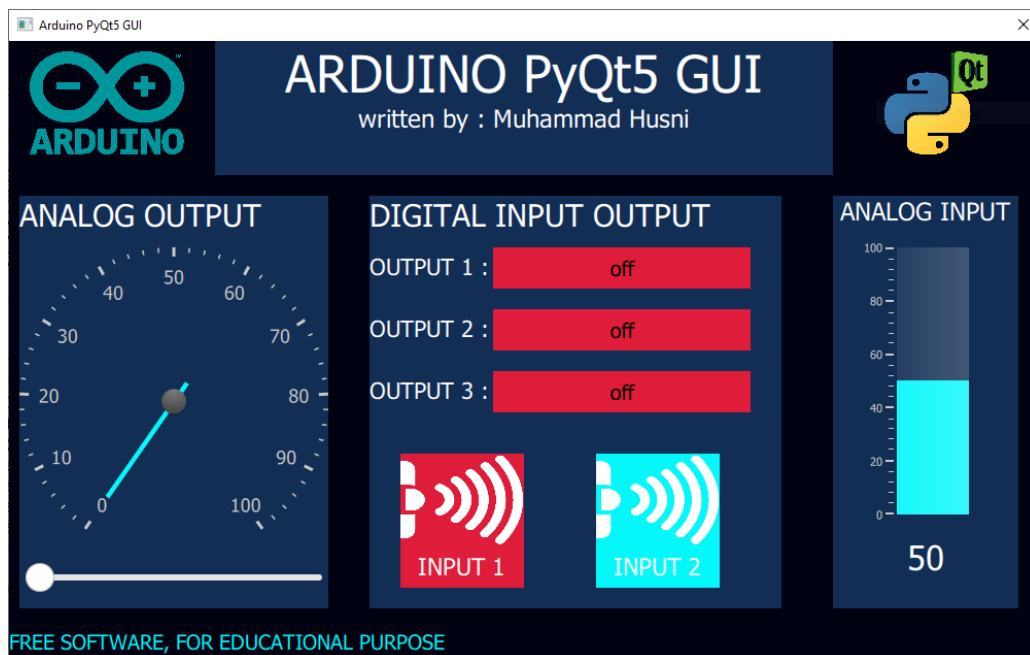
Qt adalah toolkit yang gratis dan open source untuk membuat GUI lintas platform mulai dari Windows, macOS, Linux dan Android dengan satu codebase [14]. Qt juga support untuk multimedia, database dan vector graphics. Qt didirikan oleh Eirik Chambe-Eng dan Haavard Nord pada tahun 1991, mendirikan perusahaan Qt Trolltech pertama pada tahun 1994. Qt saat ini dikembangkan oleh The Qt Company dan terus diperbarui secara berkala, menambahkan fitur dan memperluas dukungan seluler dan lintas platform.

D. PyQt

Seperti yang telah dijelaskan, bahwa codebase Qt dapat digunakan untuk membuat GUI lintas platform. PyQt merupakan binding / kombinasi antara python dan Qt. PyQt merupakan perangkat lunak gratis yang dikembangkan oleh Riverbank computing asal Inggris. PyQt mempunyai lisensi GNU yang terdiri dari GPL (General Public License) dan lisensi komersial [15]. Contoh software yang menggunakan PyQt sebagai GUI-nya adalah dropbox, Orange, Ninja IDE, Openshot (aplikasi editing video). PyQt yang kita gunakan adalah **PyQt5**.

E. GUI yang Akan Dibuat pada Modul ini

Berikut adalah GUI yang akan dibuat pada modul ini.



Gambar 4. GUI yang akan dibuat.

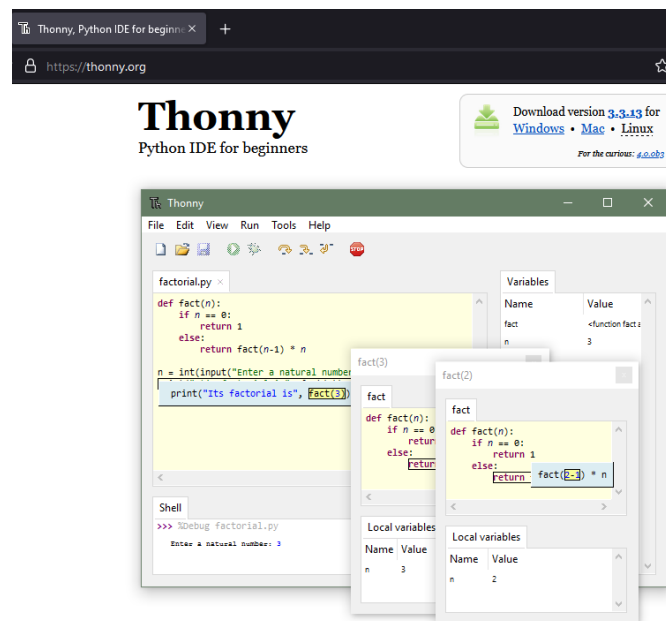
Seperti terlihat pada gambar 4, GUI yang dibuat terdiri dari windows, slider, gambar, teks, button dan komunikasi antara PyQt dengan Arduino.

BAB 2 Software yang Dibutuhkan

Untuk membuat GUI Arduino sederhana berbasis python dan PyQt5 kita membutuhkan 3 software yaitu:

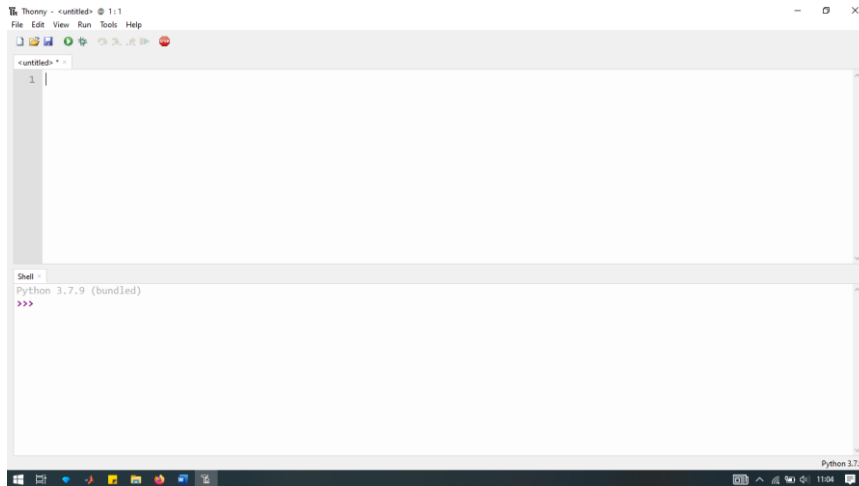
A. Thonny IDE

Thonny adalah software yang dikembangkan oleh University of Tartu di Estonia yang diperuntukkan untuk programmer python pemula. Thonny merupakan software open source alias gratis dan memiliki lisensi GPL v3 [16]. Thonny sudah memiliki Python 3.7 didalamnya yang akan memudahkan pengguna untuk mengeksekusi program dengan cepat. Untuk menginstal Thonny IDE, kita dapat mengakses link <https://thonny.org/>, lalu memilih system operasi sesuai dengan yang kita gunakan. Ada 3 opsi system operasi yaitu Windows, MacOS, dan Linux. Tampilan laman thonny.org sebagai berikut:



Gambar 5. Tampilan laman thonny.org

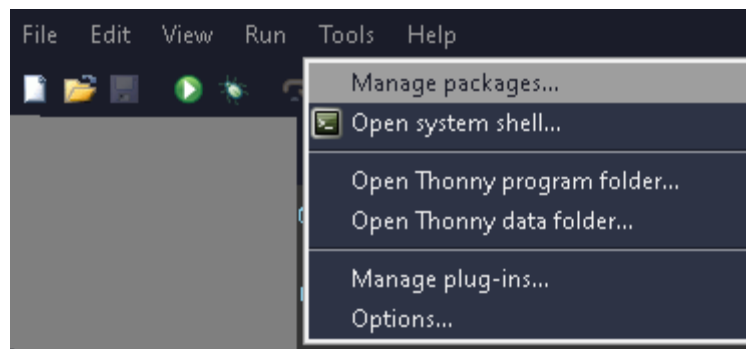
Berdasarkan gambar 5, klik download version sesuai system operasi kita dan lakukan penginstalan sampai selesai. Setelah selesai proses instal, tampilan Thonny IDE seperti berikut:



Gambar 6. Tampilan awal Thonny IDE.

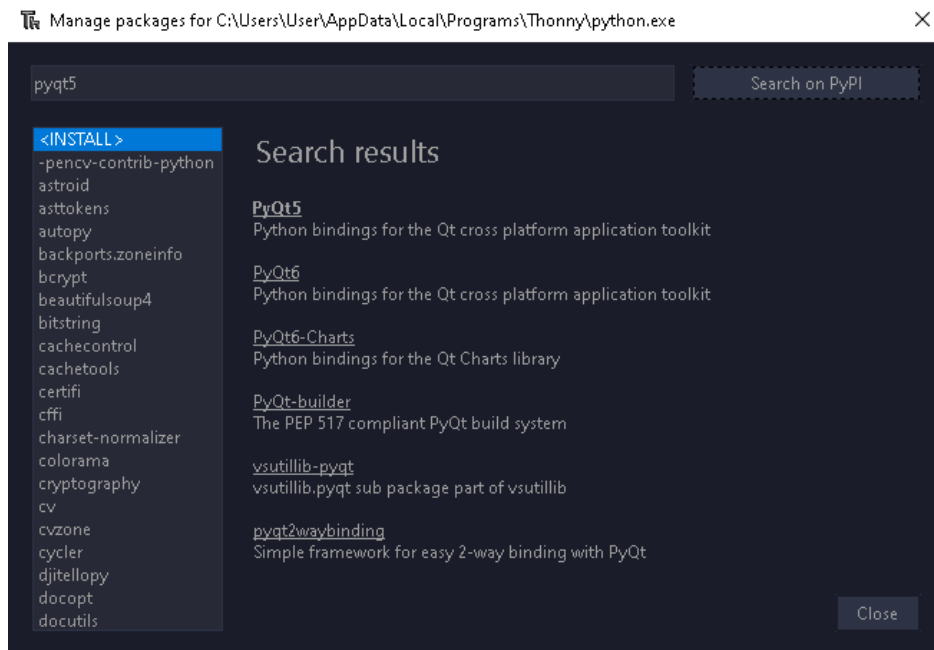
Berdasarkan gambar 6, kolom atas adalah untuk kita menulis coding dan kolom bawah (shell) adalah hasil running coding akan muncul disitu. Untuk kebutuhan membuat GUI, kita harus menginstal PyQt5 di Thonny IDE. Caranya adalah:

- Pastikan laptop/computer kita sudah terkoneksi ke internet. Klik tab “tools” lalu klik “manage packages”



- Ketik “PyQt5” lalu klik search on PyPi

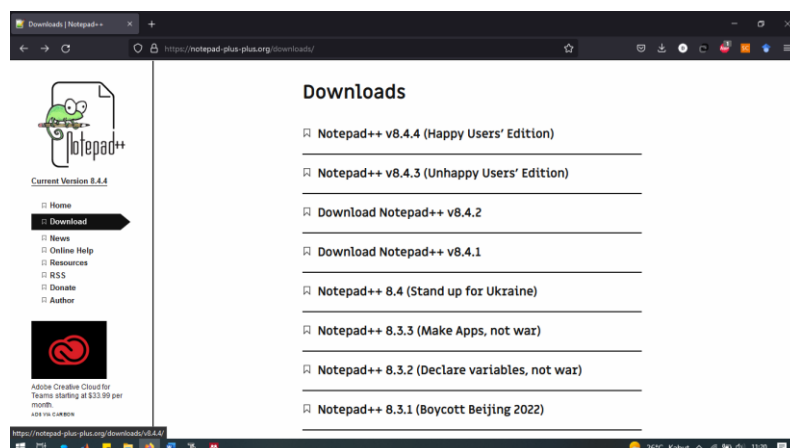




- Klik pada PyQt5 dan klik install.

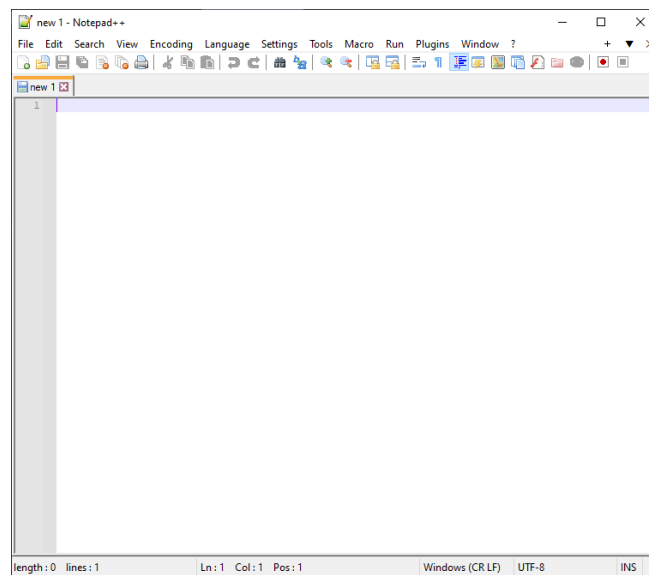
B. Notepad++

Notepad++ adalah source code editor gratis sebagai pengganti notepad yang mendukung beberapa bahasa pemrograman. Notepad++ ditulis dalam Bahasa C++ dan menggunakan Win32 API murni dan STL yang memastikan kecepatan eksekusi yang lebih tinggi, ukuran program yang lebih kecil dan penggunaan CPU yang lebih ringan [17]. Cara instal Notepad++ adalah dengan mengakses link <https://notepad-plus-plus.org/downloads/> lalu download installer yang paling atas (artinya paling baru) dan lakukan penginstalan.



Gambar 7. Tampilan web notepad++ download.

Berdasarkan gambar 7, klik yang paling atas lalu download dan instal. Setelah instal, tampilannya sebagai berikut:

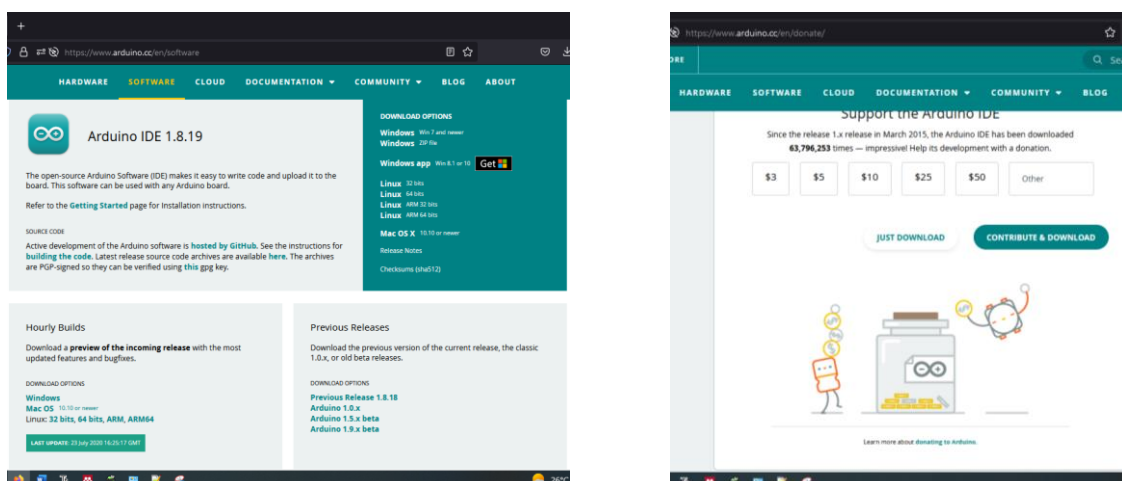


Gambar 8. Tampilan awal notepad++.

Notepad++ nantinya akan digunakan untuk menulis code dalam ekstensi qml.

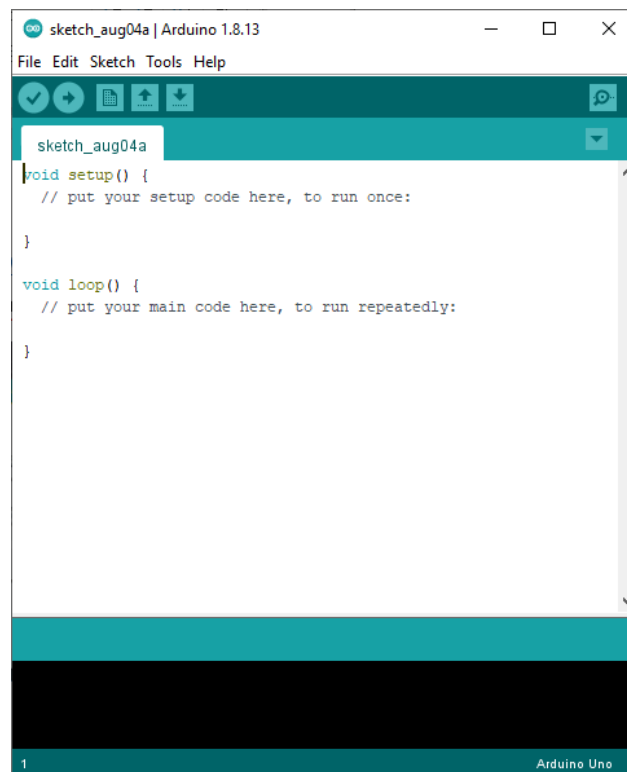
C. Arduino IDE & drivernya

Software terakhir adalah instalasi Arduino IDE dengan mengakses link <https://www.arduino.cc/en/software>, lalu klik pilihan system operasi kita (Windows, MacOS atau Linux), klik “Just Download” seperti gambar 9 berikut.



Gambar 9. Mendownload Arduino IDE.

Lakukan penginstalan dan tampilannya setelah instalasi selesai sebagai berikut:



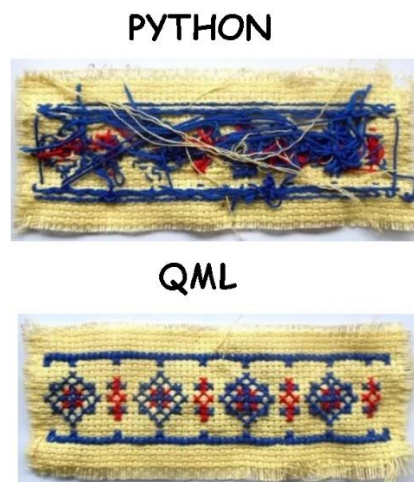
Gambar 10. Tampilan awal Arduino IDE.

BAB 3 Membuat GUI Sederhana Menggunakan PyQt5

Pada bab sebelumnya kita telah melakukan instalasi software yang dibutuhkan untuk membuat GUI sederhana menggunakan PyQt5 yaitu Thonny IDE, Notepad++ dan Arduino IDE. Selanjutnya kita akan membahas bagaimana step by step dalam membuat sebuah GUI.

A. Program minimal untuk menjalankan PyQt5

Untuk menjalankan program PyQt5, paling tidak dibutuhkan dua buah file yaitu file python dan file QML dalam satu buah folder. File Python berfungsi sebagai program utama yang berfungsi sebagai induk program dan file QML sebagai program yang menampilkan tampilan GUI. File python di-run di Thonny IDE, dan file QML dibuat di Notepad++. Adapun perumpamaan hubungan Python dan QML pada gambar 11.

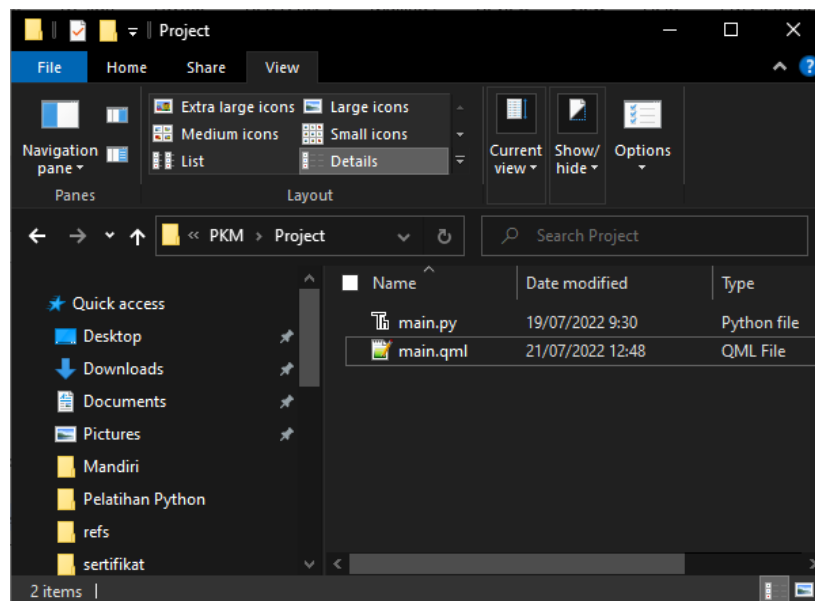


Gambar 11. Ilustrasi python dan QML.

Python bertindak sebagai base, sebagai wadah. Sedang QML bertindak sebagai alat untuk merapikan dan mengaplikasikan wadah tersebut agar dapat membentuk sebuah fungsi dengan tampilan yang menarik.

Langkah selanjutnya adalah membuat sebuah folder misalkan kita beri nama folder "Project" dan kita isi folder tersebut dengan file "main.py" dan "main.qml". File main.py kita buat di Thonny IDE dan save di folder "Project", file "main.qml" kita buat di

Notepad++ dan kita save di folder “Project” dengan mengubah ekstensinya menjadi QML. Hasilnya, folder project akan berisi file seperti gambar 12.



Gambar 12. File main.py dan main.qml disimpan dalam 1 folder "Project".

Penamaan folder dan filenya **bebas** disesuaikan dengan keinginan kita asalkan dalam satu folder tersebut file .py dan file .qml digunakan penamaan yang sama. Selanjutnya, isi dari file main.py adalah sebagai berikut:

```
##### PROGRAM MEMANGGIL WINDOWS PYQT5 #####
##### memanggil library PyQt5 #####
#-----#
from PyQt5.QtCore import *
from PyQt5.QtGui import *
from PyQt5.QtQml import *
from PyQt5.QtWidgets import *
from PyQt5.QtQuick import *
import sys
#-----#

##### mengisi class table dengan instruksi pyqt5#####
#-----#
class table(QObject):
    def __init__(self, parent = None):
        super().__init__(parent)
```

```

self.app = QApplication(sys.argv)
self.engine = QQmlApplicationEngine(self)
self.engine.rootContext().setContextProperty("backend", self)
self.engine.load(QUrl("main.qml"))
sys.exit(self.app.exec_())

#-----#
##### memanggil class table di mainloop#####
#-----#
if __name__ == "__main__":
    main = table()

#-----#

```

Sedangkan pada file main.qml, isikan program berikut:

```

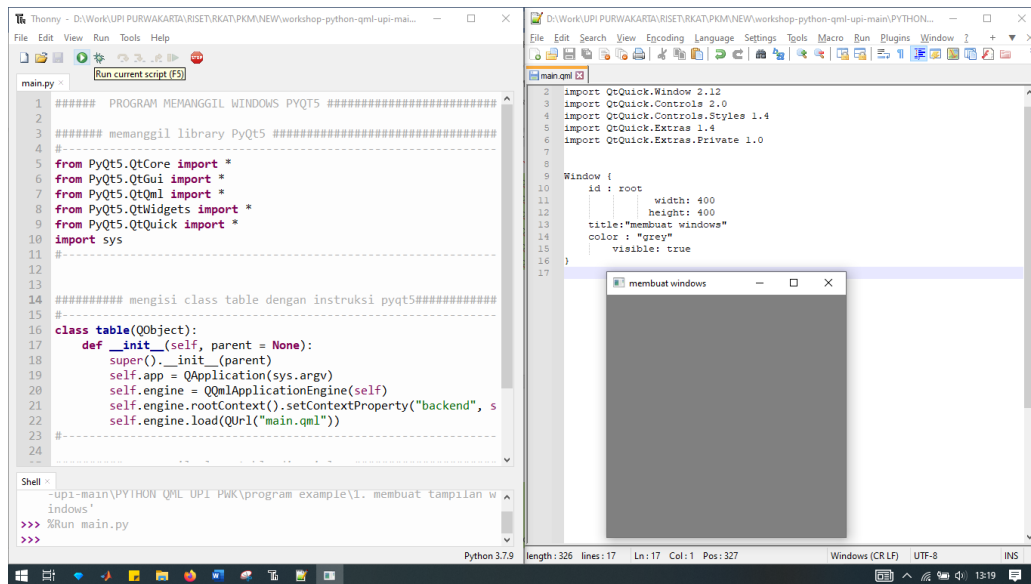
import QtQuick 2.12
import QtQuick.Window 2.12
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    height: 400
    title:"membuat windows"
    color : "grey"
    visible: true
}

```

Saat kita klik tombol Run pada Thonny IDE, maka akan muncul tampilan Windows seperti gambar 13.



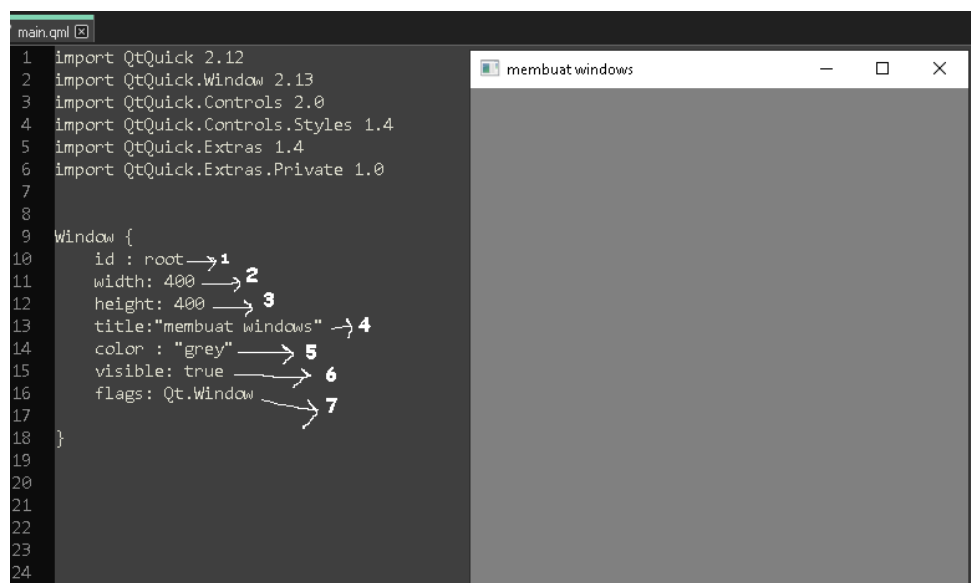


Gambar 13. Program dasar untuk membuat windows.

Jika sudah berhasil mengikuti setiap langkahnya, maka kita telah berhasil mengkomunikasikan file .py dan .qml nya. Selanjutnya kita akan menginjak ke materi lainnya.

B. Membuat Tampilan Windows

Seperti yang sudah dijelaskan sebelumnya, kita sudah berhasil membuat windows. Namun perlu dijelaskan properties apa saja yang ada dalam file QML kita untuk membuat windows sebagai berikut:



Gambar 14. Properties dalam membuat windows.

Berikut ini adalah tampilan windows yang dibuat dengan instruksi dasar. Terdiri dari 7 *properties* dasar untuk membuat sebuah Window yaitu id, weight, height, title, color, visible, dan flags. Adapun penjelasan dari tiap bagian dasar pada tabel 1.

Tabel 1. 7 properties dasar membuat Window.

Properties	Fungsi	Contoh pengisian
Id	Sebagai kode unik komponen qml	root, window, menu
Width	Untuk mengatur lebar windows dalam satuan pixel	100, 200, 300....dst
Height	Untuk mengatur tinggi windows dalam satuan pixel	100,200,300... dst
Title	Untuk menampilkan judul pada windows	"membuat windows", "program saya", atau apapun kalimatnya diiringi dengan petik dua
Color	Untuk menampilkan warna dasar pada windows	"red", "green", "blue", atau dapat juga menggunakan kode warna heksadesimal seperti "#68F3F8"
Visible	Untuk menampilkan keseluruhan komponen pada window	True
Flags	Untuk menampilkan jenis window yang dipilih	Qt.Window,

Berikut adalah beberapa contoh program membuat window. Coding dilakukan pada file QML kita.

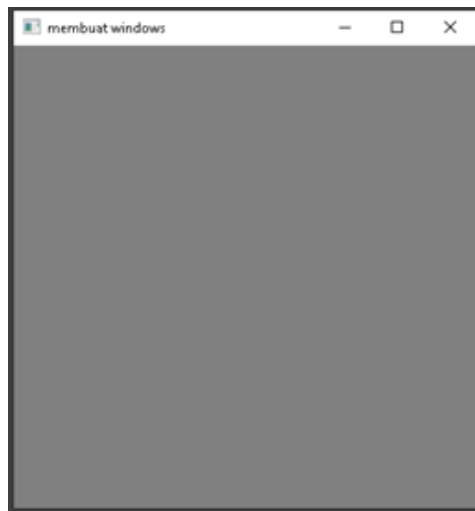
1. Program Window default

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    height: 400
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Window
}
```

Hasilnya:





Gambar 15. Window default.

2. Program Window Fullscreen

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    height: 400
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Window

    Component.onCompleted: {
        root.showFullScreen();
    }
}
```

Hasilnya akan menampilkan sebuah window secara fullscreen.

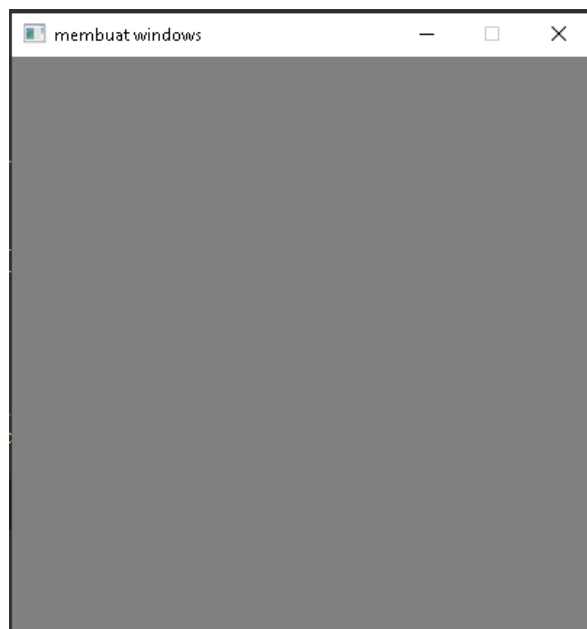
3. Program window dengan lebar dan tinggi fix tidak dapat diubah ubah

```
import QtQuick 2.12
```

```
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Window
}
```

Hasilnya seperti berikut:



Gambar 16. Window dengan ukuran yang fixed.

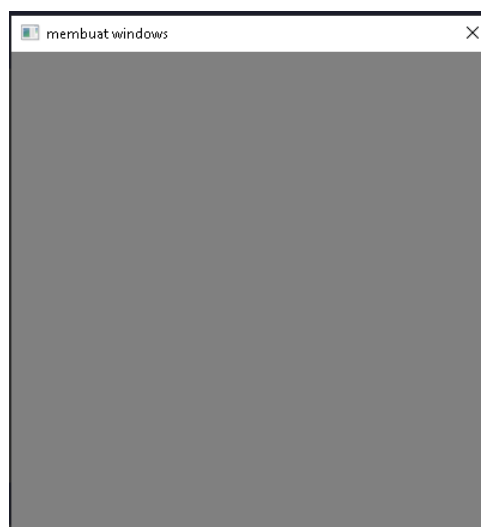
Berdasarkan gambar 16, window yang muncul ukurannya sudah fixed sesuai ukuran yang kita tentukan pada coding. Sedangkan pada window default (gambar 15), ukurannya dapat kita ubah dengan menarik ujung window-nya.

4. Program Window hanya title dan tombol close

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Dialog
}
```

Hasil:



Gambar 17. Window hanya title dan tombol close

5. Program Window hanya judul

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.WindowActive
}
```

Hasilnya:



Gambar 18. Window hanya title.

6. Program Window polos tanpa judul dan tombol tombol

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.FramelessWindowHint
}
```

Hasilnya:



Gambar 19. Window tanpa title dan tombol close.

C. Menampilkan Teks

Salah satu komponen yang penting dalam sebuah GUI adalah teks. Dengan adanya teks, dapat mempermudah user untuk memahami fungsi dari GUI tersebut. Coding untuk menampilkan teks terdiri dari 8 *properties* dasar yang diletakkan dalam sebuah

Window yaitu id, x, y, text, color, font.family, font.pixelSize, dan font.bold. Ini adalah program dasar untuk menampilkan teks pada QML.

```
Text{
    id : text1
    x:100
    y:200
    text:"Hello World"
    color: "#00FF00"
    font.family : "Comic Sans MS"
    font.pixelSize: 35
    font.bold : true
}
```

Adapun penjelasan dari tiap komponen dasar pada table 2:

Tabel 2. Properties dalam membuat teks.

Properties	Fungsi	Contoh pengisian
Id	Sebagai kode unik komponen qml	root, window, menu
X	Untuk mengatur posisi horizontal	100, 200, 300....dst
Y	Untuk mengatur posisi vertical	100,200,300... dst
Text	Untuk menampilkan text	"hello world", "nama saya husni", atau apapun kalimatnya diiringi dengan petik dua
Color	Untuk menampilkan warna dasar pada tulisan	"red", "green", "blue", atau dapat juga menggunakan kode warna heksadesimal seperti "#68F3F8"
font.family	Jenis font yang dibutuhkan	"Times New Roman", "Arial", "Comic Sans MS", dan font lainnya
font.pixelSize	Untuk mengatur besarnya tulisan	Angka nilai besarnya font
font.bold	Untuk mengatur tebalnya tulisan	"true" "false"

Contoh Program lengkap untuk memunculkan teks :

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
```

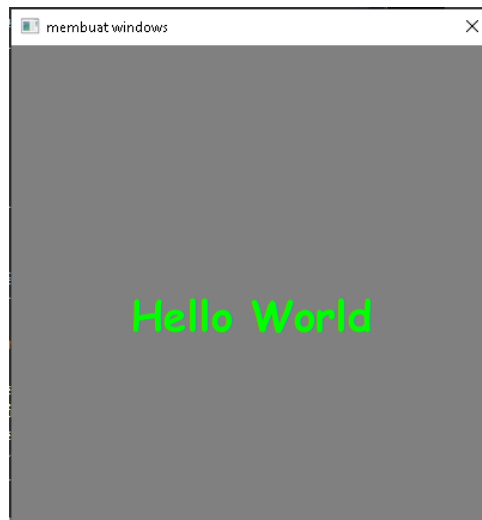
```
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Dialog

    Text{
        id : text1
        x:100
        y:200
        text:"Hello World"
        color: "#00FF00"
        font.family : "Comic Sans MS"
        font.pixelSize: 35
        font.bold : true
    }
}
```

Hasilnya sebagai berikut:





Gambar 20. Tampilan menambahkan teks dalam window.

D. Membuat Tombol / Button

Dalam sebuah GUI, tombol digunakan untuk mengaktifkan / menonaktifkan sebuah fungsi. Sebagai contoh, saat kita klik tombol, maka akan menuju ke menu selanjutnya. Atau dapat juga saat kita menekan tombol, maka akan menyalakan lampu LED Arduino. Pada QML, tombol atau button terdiri dari 6 *properties* dasar yaitu id, x, y, text, palette, dan onClicked. Ini adalah program dasar untuk menampilkan tombol pada QML.

```
Button {
    id: button1
    x :100
    y :200
    text: "button1"
    palette {
        button: "#00FF00"
        buttonText: "black"
    }
    onClicked:{
    }
}
```

Tabel 3. Properties dasar membuat tombol / button.

Properties	Fungsi	Contoh pengisian
Id	Sebagai kode unik komponen qml	root, window, menu
X	Untuk mengatur posisi horizontal	100, 200, 300....dst
Y	Untuk mengatur posisi vertical	100,200,300... dst
Text	Untuk menampilkan text	"hello world", "nama saya husni", atau apapun kalimatnya diiringi dengan petik dua
Palette	Untuk mengatur warna tombol dan warna tulisan	{ button : "green" buttonText:"red" }
OnClicked	Untuk menaruh program yang berjalan ketika tombol di klik	*penjelasan detail ada pada integrasi python dan qml"

Berikut adalah contoh program lengkap untuk memunculkan tombol:

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0
```

```
Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Dialog
```

```
Button {
```

```

        id: button1
        x :100
        y :200
        text: "button1"

        palette {
            button: "#00FF00"
            buttonText: "black"
        }

        onClicked:{
        }
    }
}




```

Hasil:



E. Memasukkan Gambar

Gambar dalam GUI dapat berfungsi sebagai ikon, header atau untuk memperindah GUI kita. Agar dapat memasukkan gambar pada GUI kita, maka gambar yang ingin kita gunakan harus kita letakkan pada folder yang sama seperti gambar berikut:

4. memasukkan gambar				
Name	Date modified	Type	Size	
 arduino.png	6/15/2022 10:51 AM	PNG File	91 KB	
 main.py	6/13/2022 1:48 PM	PY File	2 KB	
 main.qml	6/15/2022 10:55 AM	QML File	1 KB	

Gambar 21. File gambar diletakkan di folder yang sama.

Gambar harus didalam folder yang sama dengan file .py dan .qml. Terdapat 5 *properties* dasar untuk membuat masukkan gambar yaitu id, x, y, width, height, dan source. Ini adalah program dasar untuk menampilkan gambar pada QML.

```
Image{
    x:50
    y:0
    width : 250
    height : 250
    source:"arduino.png"
}
```

Tabel 4 menjelaskan properties dasar untuk memasukkan gambar.

Tabel 4. Properties dasar memasukkan gambar.

Properties	Fungsi	Contoh pengisian
Id	Sebagai kode unik komponen qml	root, window, menu
X	Untuk mengatur posisi horizontal	100, 200, 300....dst
Y	Untuk mengatur posisi vertical	100,200,300... dst
width	Untuk mengatur lebar windows dalam satuan pixel	100,200,300... dst
height	Untuk mengatur tinggi windows dalam satuan pixel	100,200,300... dst
Source	Untuk memanggil nama file gambar. Pada umumnya format .png atau .jpg	"Arduino.png"

Contoh Program lengkap untuk memunculkan gambar :

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0
```

```
Window {
```

```

id : root
width: 400
maximumWidth : width
minimumWidth : width
height: 400
maximumHeight : height
minimumHeight : height
title:"membuat windows"
color : "grey"
visible: true
flags: Qt.Dialog

Image{
x:50
y:0
width : 250
height : 250
source:"arduino.png"
}
}

```

Hasilnya sebagai berikut:



Gambar 22. Memasukkan gambar pada window.

F. Membuat Slider

Slider berfungsi untuk menampilkan sebuah besaran / angka yang nantinya akan dikomunikasikan datanya dengan perangkat lain seperti Arduino. Terdiri dari 11 *properties* dasar untuk membuat masukkan slider yaitu id, x, y, width, height, value, from, to, stepSize, orientation, dan onValueChanged. Ini adalah program dasar untuk menampilkan slider pada QML.

```
Slider {
    id: slider1
    x:0
    y:150
    height: 20
    width: 300
    value: 0
    from:10
    to: 255
    stepSize: 5
    orientation: Qt.Horizontal
    onValueChanged: {
    }
}
```

Penjelasan properties slider sebagai berikut:

Tabel 5. Properties dasar slider.

Properties	Fungsi	Contoh pengisian
Id	Sebagai kode unik komponen qml	root, window, menu
X	Untuk mengatur posisi horizontal	100, 200, 300....dst
Y	Untuk mengatur posisi vertical	100,200,300... dst
width	Untuk mengatur lebar windows dalam satuan pixel	100,200,300... dst
height	Untuk mengatur tinggi windows dalam satuan pixel	100,200,300... dst
value	Untuk mengatur nilai pertama pada slider	100,200,300... dst
From	Untuk mengatur nilai terendah pada slider	100,200,300... dst
To	Untuk mengatur nilai tertinggi pada slider	100,200,300... dst
stepSize	Untuk mengatur nilai langkah pada slider	100,200,300... dst
orientation	Untuk menentukan orientasi slider	Qt.Horizontal Qt.Vertical

onValueChanged	Untuk meletakkan program apabila slider berubah nilai	*penjelasan detail ada pada integrasi python dan qml
----------------	---	--

Contoh Program lengkap untuk memunculkan slider dengan nilai slider berupa teks:

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Dialog

    Slider {
        id: slider1
        x:0
        y:150
        height: 20
        width: 300
        value: 0
        from:10
        to: 255
        stepSize: 5
        orientation: Qt.Horizontal
        onValueChanged: {
        }
    }

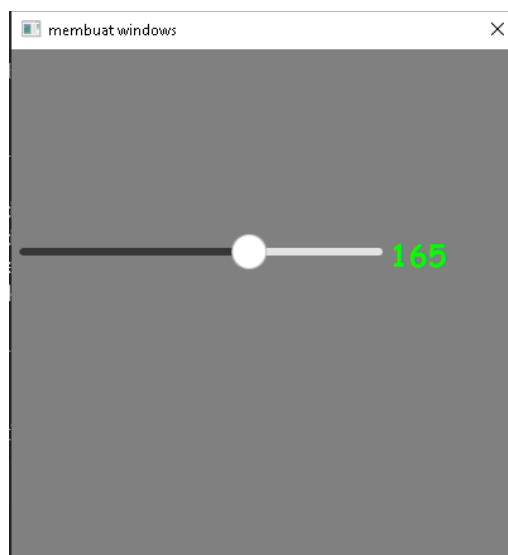
    Text {
```

```

x:300
y:145
text : slider1.value
color : "#00FF00"
font.pixelSize:24
font.bold : true
font.family : "Comic Sans MS"
    }
}

```

Hasilnya sebagai berikut:



Gambar 23. Tampilan slider dalam sebuah window.

G. Membuat Gauge

Fungsi gauge dalam sebuah GUI sebenarnya mirip dengan slider hanya dengan tampilan yang berbeda. Ada dua jenis gauge yaitu `CircularGauge` dan `Gauge`. Terdiri dari 9 *properties* dasar untuk membuat masukkan `CircularGauge` yaitu `id`, `x`, `y`, `width`, `height`, `value`, `minimumValue`, `maximumValue`, dan `style`. Penjelasan *properties* tersebut seperti berikut:

Tabel 6. Properties gauge / circular gauge.

Properties	Fungsi	Contoh pengisian
Id	Sebagai kode unik komponen qml	root, window, menu
X	Untuk mengatur posisi horizontal	100, 200, 300....dst
Y	Untuk mengatur posisi vertical	100,200,300... dst

width	Untuk mengatur lebar windows dalam satuan pixel	100,200,300... dst
height	Untuk mengatur tinggi windows dalam satuan pixel	100,200,300... dst
value	Untuk mengatur nilai pertama pada Gauge	100,200,300... dst
minimumValue	Untuk mengatur nilai paling kecil pada Gauge	100,200,300... dst
maximumValue	Untuk mengatur nilai paling besar pada Gauge	100,200,300... dst
Style	Untuk mengatur step pada Gauge	CircularGaugeStyle { labelStepSize: 10 }

Ini adalah program dasar untuk menampilkan CircularGauge pada QML.

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 400
    maximumWidth : width
    minimumWidth : width
    height: 400
    maximumHeight : height
    minimumHeight : height
    title:"membuat windows"
    color : "grey"
    visible: true
    flags: Qt.Dialog

    CircularGauge {
        id : gauge1
        x: 10
        y: 70
        height : 250
        width : 250
        value: 0
        minimumValue: 0
    }
}
```

```

        maximumValue: 100

        style: CircularGaugeStyle {
            labelStepSize: 10
        }

    }

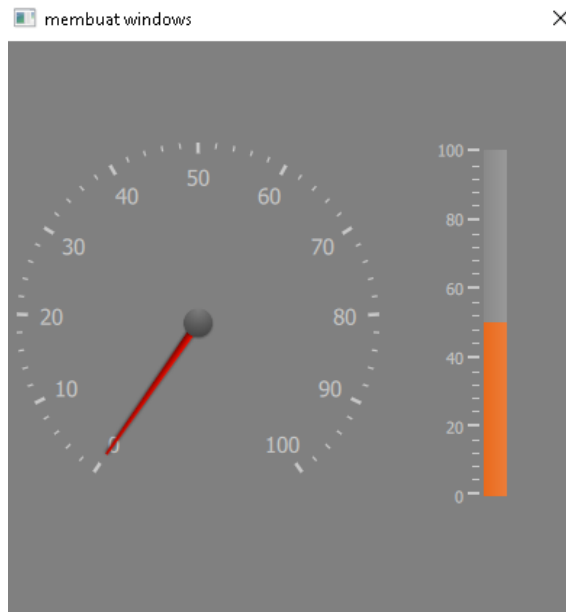
    Gauge {
        id : gauge2
        x: 300
        y: 70
        height : 250
        width : 250
        minimumValue: 0
        value: 50
        maximumValue: 100
        tickmarkStepSize: 20

        style: GaugeStyle {

            valueBar: Rectangle {
                color: "#e85d08"
                implicitWidth: 16
            }
        }
    }
}

```

Hasilnya sebagai berikut:



Gambar 24. Gauge and circular gauge pada window.

H. Membuat Rectangle

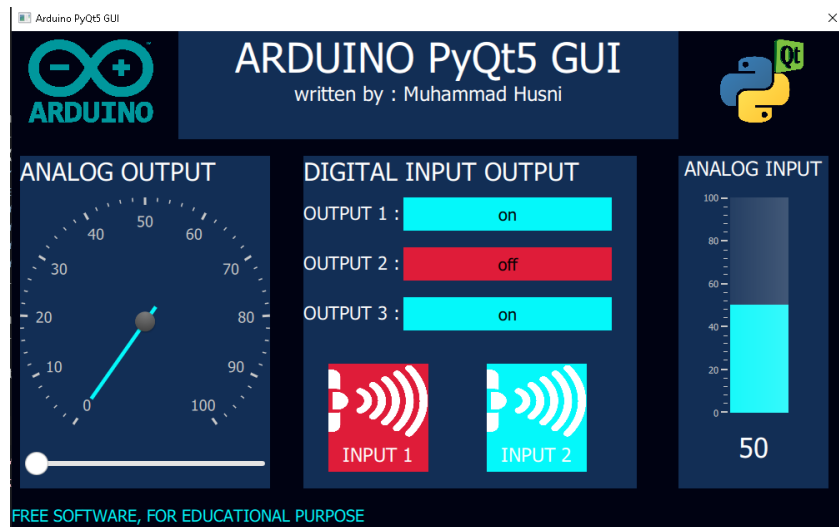
Rectangle adalah perintah untuk membuat kotak. Kotak pada GUI berfungsi sebagai pemisah antara satu komponen dengan komponen lain. Program untuk membuat rectangle sebagai berikut:

```
Rectangle{
  x: 200
  y:0
  width : 600
  height : 130
  color : "#122e55"

}
```

I. Menggabungkan Seluruh Fungsi

Setelah kita mengetahui seluruh fungsi untuk membuat GUI sederhana, tugas selanjutnya adalah menggabungkan keseluruhan fungsi tersebut sesuai keinginan dan kreatifitas kita. Sebuah GUI tidak hanya harus mudah dipahami penggunaannya oleh user tapi juga harus menarik tampilannya. Sebagai contoh, penggabungan seluruh fungsi dalam sebuah GUI adalah sebagai berikut:



Gambar 25. GUI sederhana menggunakan seluruh fungsi.

BAB 4 INTEGRASI DENGAN ARDUINO

Sebuah GUI tidak akan berfungsi jika setiap komponen GUI tersebut tidak memiliki fungsi apa-apa. Salah satu untuk menambahkan fungsi komponen GUI tersebut adalah dengan mengkomunikasikannya dengan system lain contohnya Arduino. Pada bab ini kita akan membuat proyek sederhana untuk mengontrol on off LED melalui GUI kita.

A. Apa itu Arduino

Arduino adalah mikrokontroller / pengendali mikro papan tunggal(single board) yang bersifat sumber terbuka dan menjadi salah satu proyek Open Source Hardware yang paling populer saat ini.

B. Komunikasi Serial pada Arduino

Komunikasi serial adalah sebuah komunikasi yang terjadi dengan mengirimkan data per-bit secara berurutan dan bergantian. Dengan adanya komunikasi serial, maka Arduino tak hanya bisa mengolah data dari pin input dan outputnya saja, tetapi juga bisa dikomunikasikan secara dua arah dengan perangkat komputer untuk menampilkan hasil pengolahan datanya. Bahkan Arduino pun bisa melakukan komunikasi serial dengan perangkat Android melalui koneksi bluetooth. Di Arduino IDE sendiri sudah disediakan fitur untuk komunikasi dua arah melalui serial monitor yang bisa digunakan untuk berbagai keperluan. Dengan adanya fitur ini, maka komputer bisa mengirim dan menerima data dari Arduino. Komunikasi serial Arduino dengan PC memungkinkanmu mengontrol Arduino melalui komputer atau memantau sesuatu yang terjadi padanya.

C. Parsing data

Parsing data adalah sebuah metode untuk men-deskripsi data. Fungsi parsing data adalah memisah-misahkan data yang diterima kedalam variable variable yang akan digunakan. Pada kasus ini ada dua bagian proses parsing data yaitu : parsing data Arduino dari python dan parsing data python dari Arduino.



Parsing data Arduino dari python:

Python to Arduino

* (val 1) | (val 2) | (val 3) | (val 4)

Pada kasus ini ada 4 buah jenis data dari python yaitu data led 1, led 2, led 3, dan led 4. Untuk memisahkan data data tersebut dibutuhkan sebuah program. Adapun programnya adalah sebagai berikut :

```
//program untuk menerima data serial
while (Serial.available()>0){
  delay(10);
  c = Serial.read();
  myString += c;
  data_buffer = myString;
}
//memisah misahkan data (parsing) serial yang diterima
if (myString.length()>0){
  Index1 = myString.indexOf('*');
  Index2 = myString.indexOf('|', Index1+1);
  Index3 = myString.indexOf('|', Index2+1);
  Index4 = myString.indexOf('|', Index3+1);
  Index5 = myString.indexOf('|', Index4+1);

  secondValue = myString.substring(Index1+1, Index2);
  thirdValue = myString.substring(Index2+1, Index3);
  fourthValue = myString.substring(Index3+1, Index4);
  firstValue = myString.substring(Index4+1, Index5);
  myString="";
}
```

Parsing data Arduino dari python

Arduino to Python

(val 1) : (val 2) : (val 3)

Pada kasus ini ada 3 buah jenis data dari arduino yaitu data dari potensiometer, tombol 1 dan tombol 2. Untuk memisahkan data data tersebut dibutuhkan sebuah program. Adapun programnya adalah sebagai berikut :

```
#-----#
#####MEMBACA DATA SERIAL#####
def serial_read(num):
    global ser_bytes
    global decoded_bytes
    global serial_data
    global analog
    global data
    global input1_color
    global input2_color

    while True:
        try:
            ser_bytes = ser.readline()
            serial_data = (ser_bytes.decode('utf-8')[:-2])

            print(serial_data)

        except:
            serial_data = serial_data

    data = serial_data.split(":")

    analog = int(data[0])
    print(analog)
    if (data[1] == "0"):
```

```

    input1_color = "#df1c39"
else:
    input1_color = "#04f8fa"

if (data[2] == "0"):
    input2_color = "#df1c39"
else:
    input2_color = "#04f8fa"

#-----#

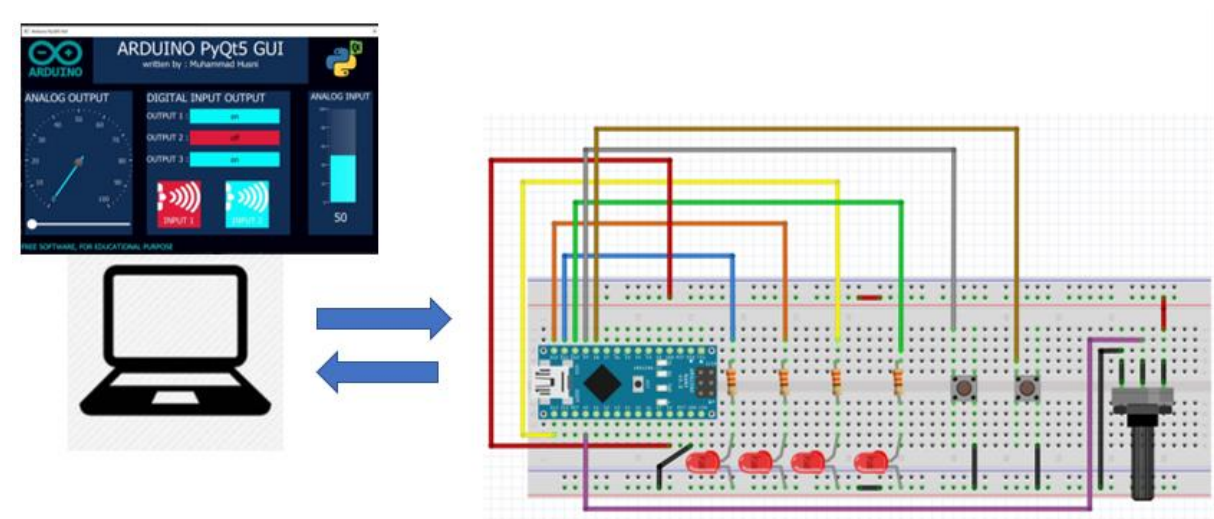
```

D. I/O digital dan analog

Komponen digital (tombol dan push button)

Komponen analog (slider dan potensiometer)

E. Mengendalikan LED via PyQt5



Program lengkap silahkan akses:

<https://github.com/muhammadhusni777/workshop-python-qml-upi/tree/main/PYTHON%20QML%20UPI%20PWK>

Program python :

```
##### PROGRAM MEMANGGIL WINDOWS PYQT5 #####
##### WRITTEN BY : MUHAMMAD HUSNI #####
##### FOR EDUCATIONAL PURPOSE #####
#####

##### memanggil library PyQt5 #####
#-----#

from PyQt5.QtCore import *
from PyQt5.QtGui import *
from PyQt5.QtQml import *
from PyQt5.QtWidgets import *
from PyQt5.QtQuick import *

import sys

import time

#-----#

#####

#-----deklarasi variabel-----#

analog = 110
```

```
input1_color = "#df1c39"

input2_color = "#df1c39"


button1_status = "0"

button2_status = "0"

button3_status = "0"


analog_output = "0"


#####

#-----mengaktifkan komunikasi serial-----#

import sys

import serial

import threading


serial_data = ""


transmit_time = 0

transmit_time_prev = 0


data_send = ""
```

```
print ("select your arduino port:")

def serial_ports():

    if sys.platform.startswith('win'):

        ports = ['COM%s' % (i + 1) for i in range(256)]

    elif sys.platform.startswith('linux') or sys.platform.startswith('cygwin'):

        # this excludes your current terminal "/dev/tty"
        ports = glob.glob('/dev/tty[A-Za-z]*')

    elif sys.platform.startswith('darwin'):

        ports = glob.glob('/dev/tty.*')

    else:

        raise EnvironmentError('Unsupported platform')

    result = []

    for port in ports:

        try:

            s = serial.Serial(port)

            s.close()

            result.append(port)

        except (OSError, serial.SerialException):

            pass

    return result
```

```

print(str(serial_ports()))

port = input("write port : ")

ser = serial.Serial(port, 9600, timeout=3)

##### mengisi class table dengan instruksi pyqt5#####
#-----#

class table(QObject):

    global analog

    def __init__(self, parent = None):

        super().__init__(parent)

        self.app = QApplication(sys.argv)

        self.engine = QQmlApplicationEngine(self)

        self.engine.rootContext().setContextProperty("backend", self)

        self.engine.load(QUrl("main.qml"))

        sys.exit(self.app.exec_())

#####TOMBOL QML KE PYTHON#####

@pyqtSlot(str)

def button1(self, message):

```

```
global button1_status

print(message)

button1_status = message


@pyqtSlot(str)
def button2(self, message):

    global button2_status

    print(message)

    button2_status = message


@pyqtSlot(str)
def button3(self, message):

    print(message)

    global button3_status

    print(message)

    button3_status = message


#####SLIDER QML KE PYTHON#####

@pyqtSlot(str)
```



```

def analog_output(self, message):

    global analog_output

    analog_output=message

#####KIRIM DATA ANALOG KE GAUGE#####

@pyqtSlot(result=float)

def get_analog(self): return analog

#####KIRIM DATA WARNA STATUS BUTTON#####

@pyqtSlot(result=str)

def get_input1_color(self): return input1_color

@pyqtSlot(result=str)

def get_input2_color(self): return input2_color

#-----#

#####MEMBACA DATA SERIAL#####

def serial_read(num):

    global ser_bytes

    global decoded_bytes

    global serial_data

    global analog

    global data

```

```
global input1_color

global input2_color


while True:

    try:

        ser_bytes = ser.readline()

        serial_data = (ser_bytes.decode('utf-8')[:-2])


        print(serial_data)


    except:

        serial_data = serial_data


    data = serial_data.split(":")


    analog = int(data[0])

    #print(analog)

    if (data[1] == "0"):

        input1_color = "#df1c39"

    else:

        input1_color = "#04f8fa"


    if (data[2] == "0"):
```

```

        input2_color = "#df1c39"

    else:

        input2_color = "#04f8fa"

#-----#

def serial_write(num):

    global transmit_time

    global transmit_time_prev

    global data_send

    while True:

        transmit_time = time.time() - transmit_time_prev

        data_send = (str("*") + str(button1_status) + str("|")

        + str(button2_status) + str("|")

        + str(button3_status) + str("|")

        + str(analog_output) + str("|")

        )

        if (transmit_time > 0.5):

            #print(data_send)

            ser.write(data_send.encode())

            transmit_time_prev = time.time()

```

```
##### memanggil class table di mainloop#####
#-----#

if __name__ == "__main__":

    t1 = threading.Thread(target=serial_read, args=(10,))

    t1.start()

    t2 = threading.Thread(target=serial_write, args=(10,))

    t2.start()

    main = table()

#-----#
```

Program QML :

```
import QtQuick 2.12
import QtQuick.Window 2.13
import QtQuick.Controls 2.0
import QtQuick.Controls.Styles 1.4
import QtQuick.Extras 1.4
import QtQuick.Extras.Private 1.0

Window {
    id : root
    width: 1000
```

```
maximumWidth : width
minimumWidth : width
height: 600
maximumHeight : height
minimumHeight : height
title:"Arduino PyQt5 GUI"
color : "#000212"
visible: true
flags: Qt.Dialog
```

```
Image{
x:20
y:10
width : 150
height : 100
source : "arduino.png"

}
```

```
Image{
x:850
y:10
width : 100
height : 100
source : "pyqt.png"

}
```

```
Rectangle{
x: 200
y:0
width : 600
height : 130
```

```
color : "#122e55"
}
```

```
Text{
anchors.horizontalCenter: parent.horizontalCenter
y:0
text : "ARDUINO PyQt5 GUI"
color : "white"
font.pixelSize : 50
}
```

```
Text{
anchors.horizontalCenter: parent.horizontalCenter
y:60
text : "written by : Muhammad Husni"
color : "white"//"#04f8fa"
font.pixelSize : 24
}
```

```
Rectangle{
x: 10
y: 150
width: 300
height:400
color: "#122e55"
```

```
Text{
x:0
y:0
text:"ANALOG OUTPUT"
font.pixelSize:30
color: "white"
```

```

    }

    CircularGauge {
        id : gauge1
        x: 0
        y: 50
        height : 300
        width : 300
        value: slider1.value
        minimumValue: 0
        maximumValue: 255

        style: CircularGaugeStyle {
            labelStepSize: 51

            needle: Rectangle {
                y: outerRadius * 0.15
                implicitWidth: outerRadius * 0.03
                implicitHeight: outerRadius * 0.9
                antialiasing: true
                color: "#04f8fa"
            }
        }
    }
}

Slider {
    id: slider1
    x:0
    y:360
    height: 20
    width: 300

```

```

        value: 0
        from:0
        to: 255
        stepSize: 1
        orientation: Qt.Horizontal
        onValueChanged: {
            backend.analog_output(value)
        }
    }
}
}

```

```

Rectangle{
    x: 350
    y: 150
    width: 400
    height:400
    color: "#122e55"

    Text{
        x:0
        y:0
        text:"DIGITAL INPUT OUTPUT"
        font.pixelSize:30
        color: "white"
    }
}

```



```
Text{
    x:0
    y:55
    text:"OUTPUT 1 :"
    font.pixelSize:22
    color: "white"
}

Button {
id: button1
x :120
y :50
width : 250
text: "off"
font.pixelSize : 20

Rectangle{
    id:button1_color
    width : parent.width
    height: parent.height
    color:"#df1c39"
}

palette {
button: "transparent"
    buttonText: "black"
}

onClicked:{
    if(button1.text == "on"){
        text = "off";
```

```

        button1_color.color = "#df1c39"
        backend.button1("0")

    }else
        if(button1.text == "off"){
            text = "on";
            button1_color.color = "#04f8fa"
            backend.button1("1")
        }
    }

}

```

```

Text{
    x:0
    y:115
    text:"OUTPUT 2 :"
    font.pixelSize:22
    color: "white"
}

```

```

Button {
    id: button2
    x :120
    y :110
    width : 250
    text: "off"
    font.pixelSize : 20
}

```

```

Rectangle{

```

```

        id:button2_color
        width : parent.width
        height: parent.height
        color:"#df1c39"
    }

    palette {
        button: "transparent"
            buttonText: "black"
    }

    onClicked:{
        if(button2.text == "on"){
            text = "off";
            button2_color.color = "#df1c39"
            backend.button2("0")

        }else
            if(button2.text == "off"){
                text = "on";
                button2_color.color = "#04f8fa"
                backend.button2("1")
            }
    }
}

```

```

Text{
    x:0

```

```

        y:175
        text:"OUTPUT 3 : "
        font.pixelSize:22
        color: "white"
    }

    Button {
        id: button3
        x :120
        y :170
        width : 250
        text: "off"
        font.pixelSize : 20

        Rectangle{
            id:button3_color
            width : parent.width
            height: parent.height
            color:"#df1c39"
        }

        palette {
            button: "transparent"
            buttonText: "black"
        }

        onClicked:{
            if(button3.text == "on"){
                text = "off";
                button3_color.color = "#df1c39"
                backend.button3("0")
            }
        }
    }
}

```

```

        }else
            if(button3.text == "off"){
                text = "on";
                button3_color.color = "#04f8fa"
                backend.button3("1")
            }
    }

}

Rectangle{
    id:sensor1_color
    x: 30
    y:250
    width : 120
    height: 130
    color: backend.get_input1_color()

    Image{
        width : parent.width
        height : 90
        source:"sensor.png"
    }
    Text {
        anchors.horizontalCenter: parent.horizontalCenter
        y: 95
        text : "INPUT 1"
        color : "white"
        font.pixelSize : 23
    }
}

```

```
    }

    Rectangle{
        id:sensor2_color
        x: 220
        y:250
        width : 120
        height: 130
        color:backend.get_input2_color()

        Image{
            width : parent.width
            height : 90
            source:"sensor.png"
        }

        Text {
            anchors.horizontalCenter: parent.horizontalCenter
            y: 95
            text : "INPUT 2"
            color : "white"
            font.pixelSize : 23
        }
    }

}
```

```

Rectangle{
    x: 800
    y: 150
    width: 180
    height:400
    color: "#122e55"

    Text{
        anchors.horizontalCenter: parent.horizontalCenter
        y:0
        text:"ANALOG INPUT"
        font.pixelSize:24
        color: "white"
    }
}

```

```

Gauge {
    id : gauge2
    x: 30
    y: 45
    height : 270
    width : 150
    minimumValue: 0
    tickmarkStepSize: 205
    value: pot_val.text
    maximumValue: 1023
}

```

```

style: GaugeStyle {

    valueBar: Rectangle {
        antialiasing: true
        color: "#04f8fa"
    }
}

```

```

        implicitWidth: 70
    }

}

}

Text{
    id : pot_val
    anchors.horizontalCenter: parent.horizontalCenter
    y:330
    font.pixelSize:33
    color : "white"
}

}

Text{
    x:0
    y:570
    text:"FREE SOFTWARE, FOR EDUCATIONAL PURPOSE"
    font.pixelSize:20

    color : "#04f8fa"
}

Timer{
    id:tmgauge
    interval: 50

```



```

        repeat: true
        running: true
        onTriggered: {
            pot_val.text = backend.get_analog()
            sensor1_color.color = backend.get_input1_color()
            sensor2_color.color = backend.get_input2_color()
        }
    }
}

```

Program Arduino :

```

/////////////////////////////////////////////////////////////////

///// PROGRAM FIRMWARE ARDUINO PYTHON ///////////////////////////////////

///// written by : muhammad husni  ///////////////////////////////////

///// FOR EDUCATIONAL PURPOSE      ///////////////////////////////////

/////////////////////////////////////////////////////////////////

/////////////////////////////////DEKLARASI VARIABEL/////////////////////////////////

String myString;

char c;

String data_buffer;

int Index1,Index2,Index3,Index4,Index5,Index6, Index7, Index8, Index9;

String secondValue, thirdValue, fourthValue, fifthValue, sixthValue, seventhValue,
eighthValue, firstValue;

```

```
unsigned long time_send;

unsigned long prev_time_send;


int led1 = 13;

int led2 = 12;

int led3 = 11;

int led_pwm = 10;


int led1_status;

int led2_status;

int led3_status;

int led_pwm_val;


int button1=9;

int button2=8;


int pot = A0;


void setup() {

    //inisialisasi baud rate serial

    Serial.begin(9600);
```

```

//inisialisasi pin input dan output

pinMode(led1, OUTPUT);

pinMode(led2, OUTPUT);

pinMode(led3, OUTPUT);

pinMode(led_pwm, OUTPUT);

pinMode(button1, INPUT_PULLUP);

pinMode(button2, INPUT_PULLUP);

}

void loop() {

    //program untuk menerima data serial

    while (Serial.available()>0){

        delay(10);

        c = Serial.read();

        myString += c;

        data_buffer = myString;

    }

    //memisah misahkan data (parsing) serial yang diterima

    if (myString.length()>0){

        Index1 = myString.indexOf('*');

        Index2 = myString.indexOf('|', Index1+1);

        Index3 = myString.indexOf('|', Index2+1);

        Index4 = myString.indexOf('|', Index3+1);

        Index5 = myString.indexOf('|', Index4+1);

```

```

secondValue = myString.substring(Index1+1, Index2);
thirdValue = myString.substring(Index2+1, Index3);
fourthValue = myString.substring(Index3+1, Index4);
firstValue = myString.substring(Index4+1, Index5);
myString="";
}

//mengirim data serial dari arduino ke python
time_send = millis() - prev_time_send;
if (time_send > 500){
    Serial.print(analogRead(pot));
    Serial.print(":");
    Serial.print(!digitalRead(button1));
    Serial.print(":");
    Serial.print(!digitalRead(button2));

    Serial.println("");
    prev_time_send = millis();
}

//mengolah data serial yang sudah diparsing
led1_status = fourthValue.toInt();
led2_status = thirdValue.toInt();

```

```

led3_status = secondValue.toInt();

led_pwm_val = firstValue.toInt();

//mentransfer data ke pin output

digitalWrite(led1, led1_status);

digitalWrite(led2, led2_status);

digitalWrite(led3, led3_status);

analogWrite(led_pwm, led_pwm_val);

}

```

References

- [1] A. Sitio, A. Sindar, M. Marbun, and D. Tiara, "Pengenalalan Data Scientist Pada Peserta PKBM AL HABIB Melalui Belajar Dasar Coding Python," *J. Pengabd. Pada Masy.*, vol. 7, no. 1, pp. 194–200, 2022, doi: 10.30653/002.202271.44.
- [2] J. Olsen, "How to (and why) use Python to teach introductory Calculus," *Pedagog. inspirationskonferensen-Genombrottet*, pp. 1–2, 2018, [Online]. Available: <https://130.235.140.198/pige/article/view/21266>.
- [3] M. Eroglu S., Toprak S., Urgan O, MD, Ozge E. Onur, MD, Arzu Denizbasi, MD, Haldun Akoglu, MD, Cigdem Ozpolat, MD, Ebru Akoglu, *Learning Python 5th*, vol. 33. 2012.
- [4] J. V. Guttag, *Introduction to computation and programming using python*, vol. 1. MIT Press, 2013.
- [5] C. Ozgur, T. Colliau, G. Rogers, and Z. Hughes, "MatLab vs. Python vs. R," *J. Data Sci.*, vol. 15, no. 3, pp. 355–372, 2021, doi: 10.6339/jds.201707_15(3).0001.
- [6] C. Sotomayor-Beltran, G. W. Z. Segura, and A. Roman-Gonzalez, "Why should Python be a compulsory introductory programming course in Lima (Peru) universities?," in *IEEE ICA-ACCA 2018 - IEEE International Conference on Automation/23rd Congress of the Chilean Association of Automatic Control:*

- Towards an Industry 4.0 - Proceedings*, 2019, pp. 1–4, doi: 10.1109/ICA-ACCA.2018.8609808.
- [7] A. Rawat, “A Review on Python Programming,” *Int. J. Res. Eng. Sci. Manag.*, vol. 3, no. 12, pp. 8–11, 2020, [Online]. Available: <https://www.ijresm.com>.
 - [8] Tiobe.com, “TIOBE index for February 2018,” 2018. <https://www.tiobe.com/tiobe-index/>. (accessed Mar. 17, 2022).
 - [9] N. Diakopoulos and S. Cass, “Interactive: The Top Programming Languages 2017,” 2017. <https://spectrum.ieee.org/static/interactive-the-top-programming-languages-2017> (accessed Mar. 17, 2022).
 - [10] Octoverse.github.com, “The State of Octoverse 2017,” 2017. <https://octoverse.github.com/> (accessed Mar. 17, 2022).
 - [11] Insights.stackoverflow.com, “Developer Survey Results 2017,” 2017. <https://insights.stackoverflow.com/survey/2017#technology> (accessed Mar. 17, 2022).
 - [12] A. Oulasvirta, N. R. Dayama, M. Shiripour, M. John, and A. Karrenbauer, “Combinatorial Optimization of Graphical User Interface Designs,” *Proc. IEEE*, vol. 108, no. 3, pp. 434–464, 2020, doi: 10.1109/JPROC.2020.2969687.
 - [13] Q. M. Ilyas, M. Ahmad, N. Zaman, M. A. Alshamari, and I. Ahmed, “Localized Text-Free User Interfaces,” *IEEE Access*, vol. 10, pp. 2357–2371, 2022, doi: 10.1109/ACCESS.2021.3139525.
 - [14] M. Fitzpatrick, “Create GUI Applications with Python & Qt5 (PyQt5 Edition): The hands-on guide to making apps with Python,” p. 825, 2020.
 - [15] B. A. Meier, *Qt5 Python GUI Programming Cookbook: Building responsive and powerful cross-platform applications with PyQt*. 2018.
 - [16] A. Annamaa, “Thonny , a Python IDE for Learning Programming Categories and Subject Descriptors,” *ITiCSE '15 Proc. 2015 ACM Conf. Innov. Technol. Comput. Sci. Educ.*, vol. 13, no. 4, p. 343, 2015.
 - [17] D. Ho, “What is Notepad++,” 2022. <https://notepad-plus-plus.org/> (accessed Aug. 04, 2022).

