CENTRO UNIVERSITARIO DE CIENCIAS EXÁCTAS E INGENIERÍAS

Temporizador

Aldo Alexandro Vargas Meza

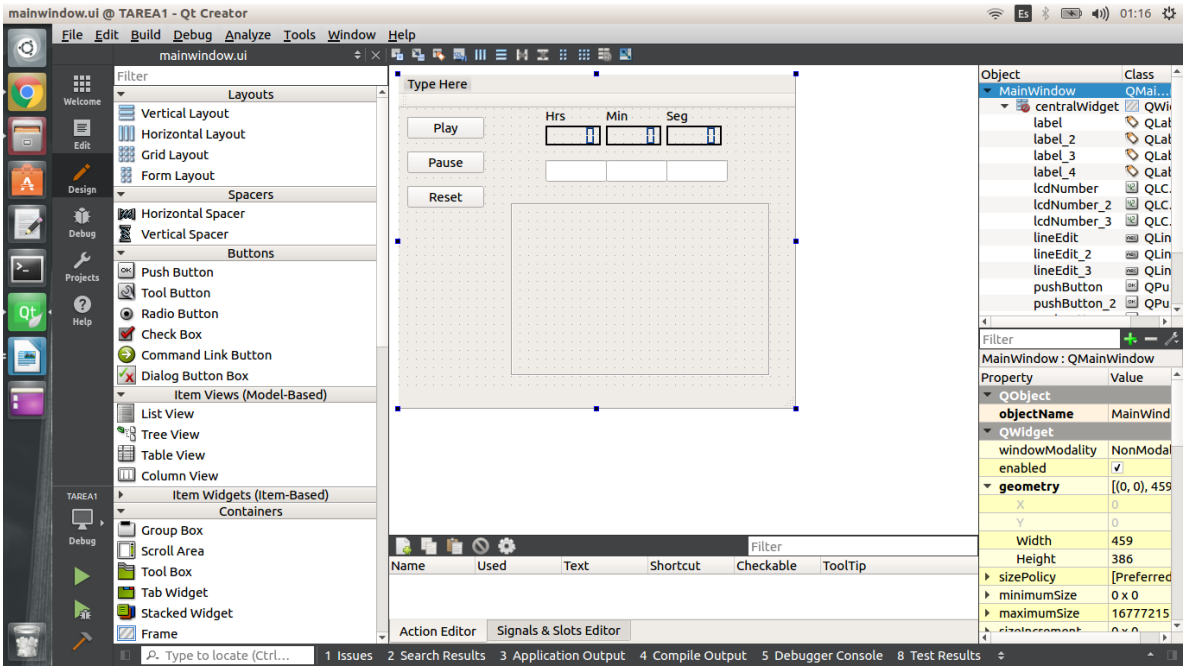
El objetivo de esta tarea, es la de replicar el funcionamiento de un temporizador. Dado un valor en horas, minutos o segundos, el contador desciende hasta el valor cero, despues emite una notificación o alarma al usuario.

Las funciones básicas son el botón de inicio, un botón de pausa, y el reset.

El temporizador diseñado aquí, tendrá como alerta al usuario una serie de imagenes, las cuales serán cargadas y procesadas a travéz de QT y librerías Open Cv.

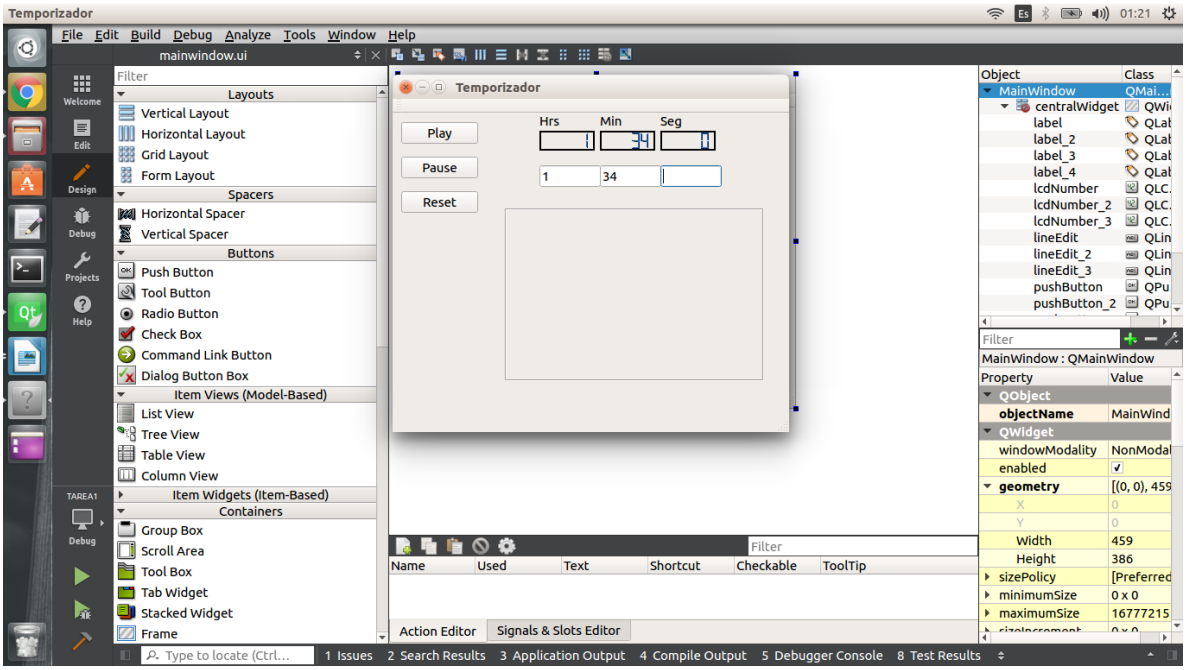
Introducción:

La interfaz gráfica a continuación, está dividida en diferentes elementos, los cuales cumplen con una función específica, ya sea mostrar al usuario, u obtener datos del mismo.

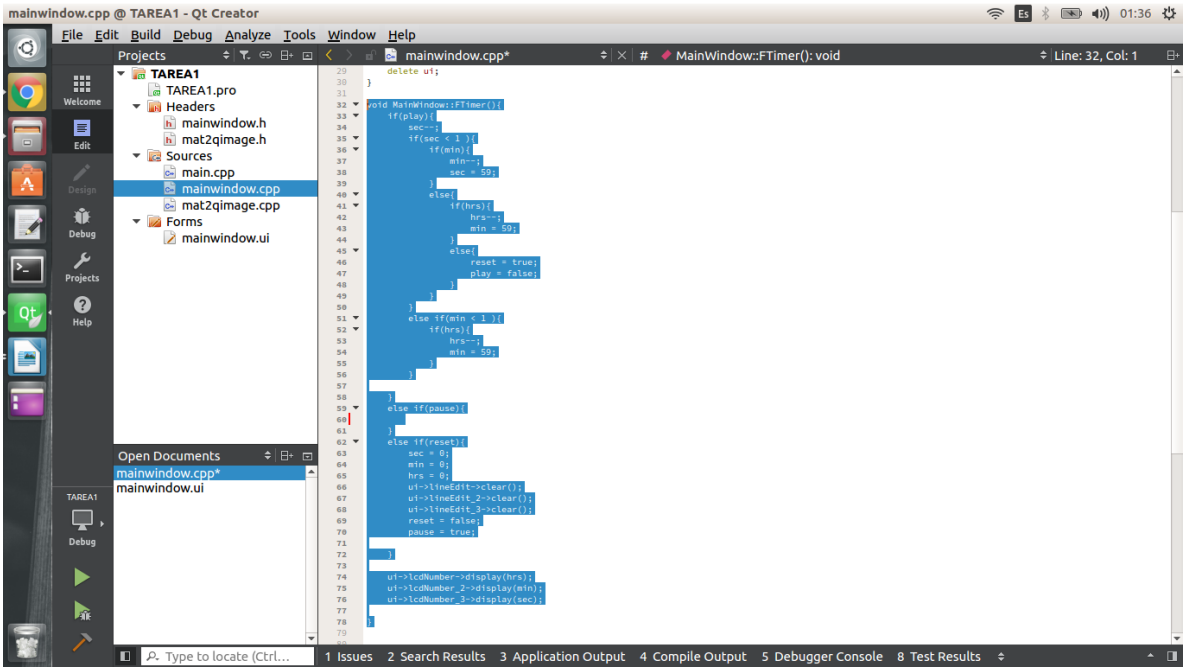


Los elementos LCD muestran valores que los lineEdit recibe por medio del teclado. Los botones play, pause y reset están programados para ejecutar las funciones principales.

En los lineEdit, se programó la función para cargar en la LCD correspondiente, el valor ingresado tan pronto la edición del lineEdit termine.

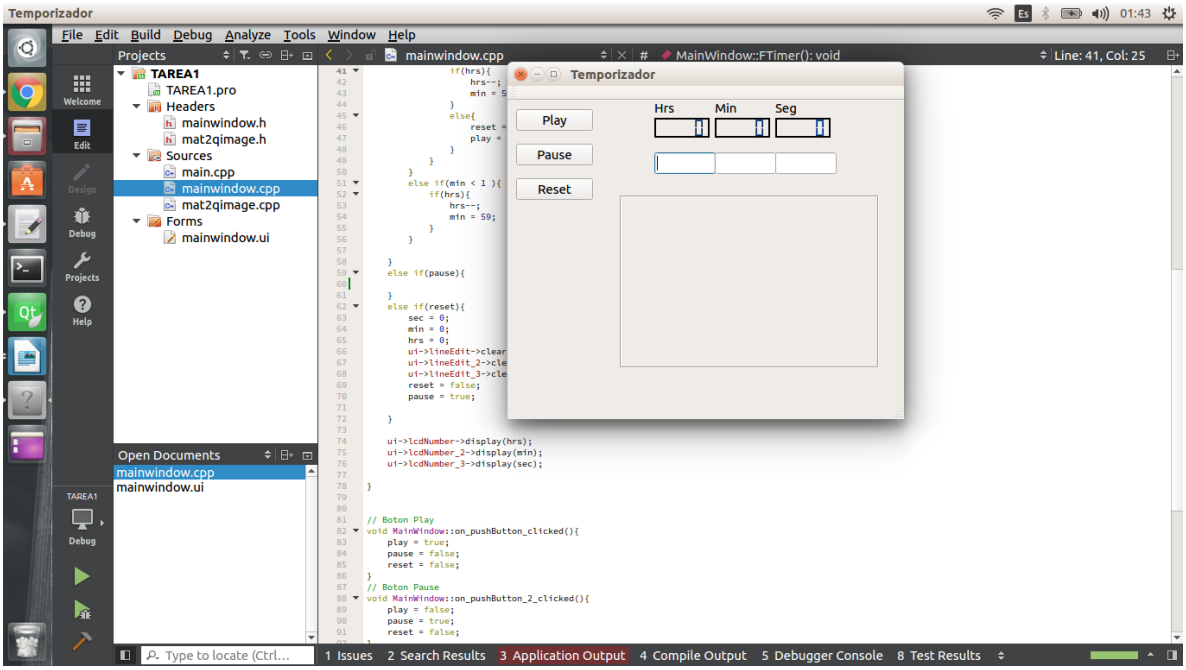


Al iniciar, el contador comienza a retroceder correctamente descontando minutos y horas, todo realizado desde la siguiente sección del código

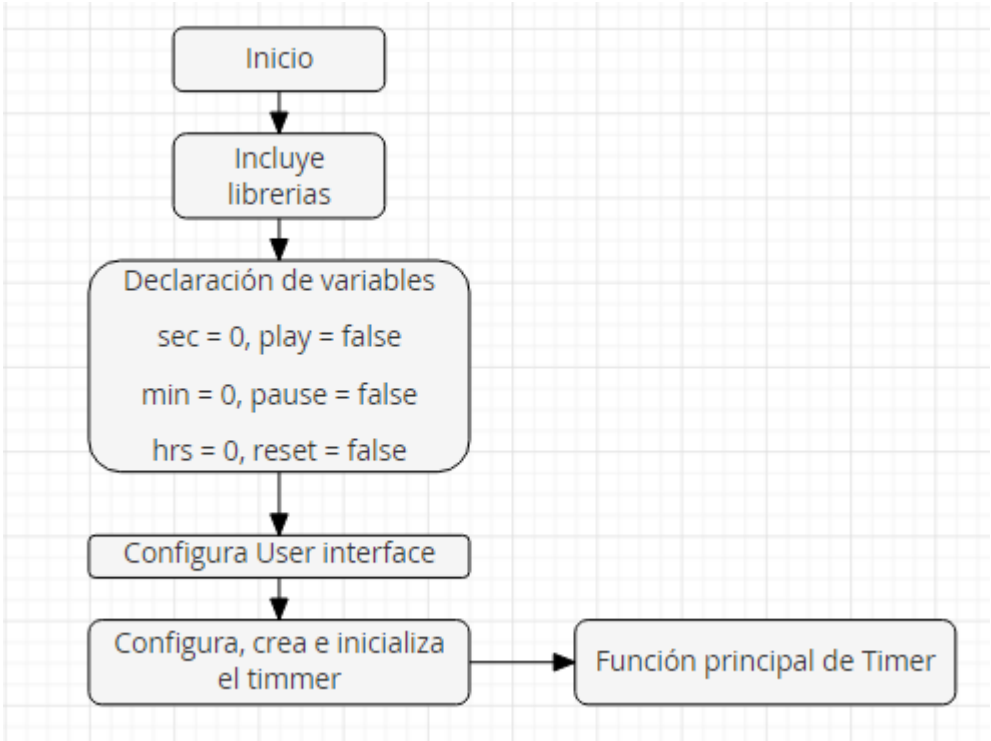


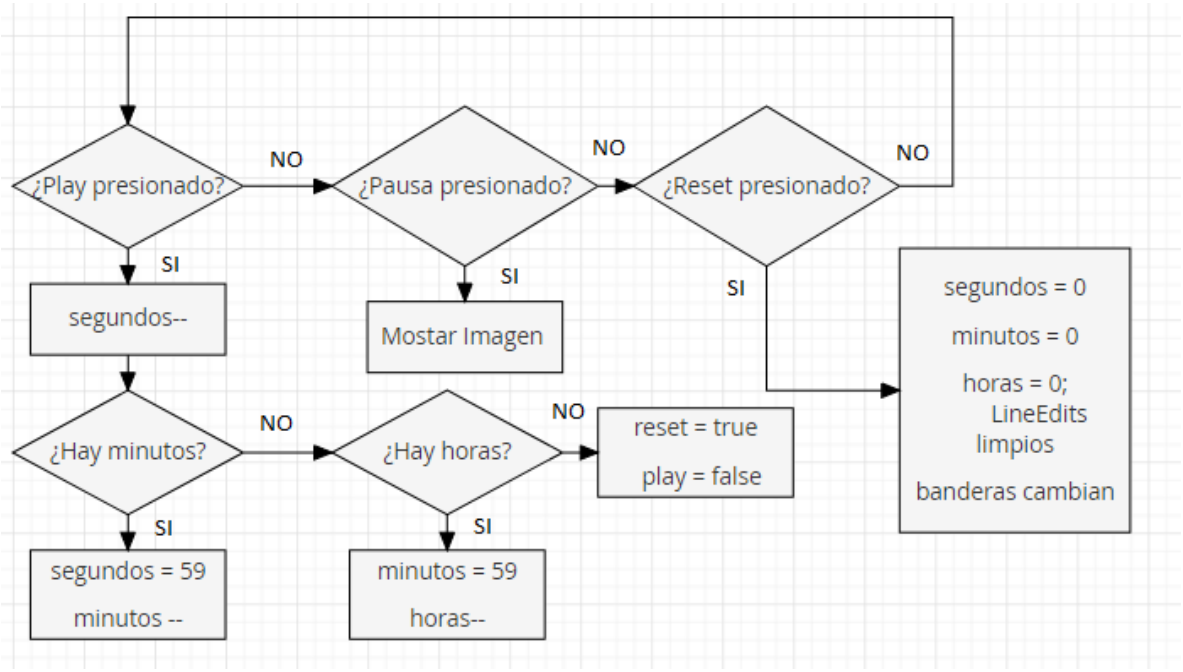
De manera sencilla, el código maneja banderas las cuales cambian de valor con los botones. Estas banderas permiten que una función previamente definida para ejecutarse cada segundo reste las variables necesarias y haga conversiones de datos con respecto de los valores actuales, después los presente en un display LCD.

De esta forma al llegar a cero el programa muestra los lineEdit vacíos y propiamente mantiene los valores listos para iniciar de nuevo.



Podemos revisar el comportamiento del contador en el siguiente diagrama de flujo:




¿Play presionado?
NO
¿Pausa presionado?
NO
¿Reset presionado?
NO
SI
segundos--
SI
Mostar Imagen
SI
segundos = 0
minutos = 0
horas = 0;
LineEdits
limpios
banderas cambian
¿Hay minutos?
NO
¿Hay horas?
NO
reset = true
play = false
SI
segundos = 59
minutos --
SI
minutos = 59
horas--

A continuación, encontramos los códigos utilizados para programar la interfaz

```
#-------------------------------------------------

#

# Project created by QtCreator 2017-09-09T10:09:35

#

#-------------------------------------------------


QT       += core gui


greaterThan(QT_MAJOR_VERSION, 4): QT += widgets


# Salida del programa

TARGET = TAREA1

TEMPLATE = app


# The following define makes your compiler emit warnings if you use

# any feature of Qt which as been marked as deprecated (the exact warnings

# depend on your compiler). Please consult the documentation of the

# deprecated API in order to know how to port your code away from it.

DEFINES += QT_DEPRECATED_WARNINGS


# You can also make your code fail to compile if you use deprecated APIs.

# In order to do so, uncomment the following line.

# You can also select to disable deprecated APIs only up to a certain version of Qt.

#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs deprecated
before Qt 6.0.0
```

```
# Configuracion de estructura del programa y archivos
SOURCES += \
        main.cpp \
        mainwindow.cpp \
    mat2qimage.cpp

HEADERS += \
        mainwindow.h \
    mat2qimage.h

FORMS += \
        mainwindow.ui



# Configuracion de rutas para OPEN CV
INCLUDEPATH += /usr/local/include/opencv2
LIBS += -L/usr/local/lib -lopencv_core -lopencv_imgcodecs -lopencv_highgui -opencv_videoio
CONFIG += link_pkgconfig
PKGCONFIG += opencv



#-------------------------------------------------
#
# Main.h
#
```

```
#-------------------------------------------------



#ifndef MAINWINDOW_H

#define MAINWINDOW_H


#include <QMainWindow>


// Librerias


#include<mat2qimage.h>

#include<opencv2/core/core.hpp>

#include<opencv2/ml/ml.hpp>

#include<opencv/cv.h>

#include<opencv2/imgproc/imgproc.hpp>

#include<opencv2/highgui/highgui.hpp>

#include<opencv2/video/background_segm.hpp>

#include<opencv2/videoio.hpp>

#include<opencv2/imgcodecs.hpp>


namespace Ui {

class MainWindow;

}


// Clase Principal

class MainWindow : public QMainWindow
```

```
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = 0);
    ~MainWindow();

public slots:
            // Funcion para conteo
    void FTimer();



private slots:
            // Funciones de elementos graficos
    void on_pushButton_clicked();

    void on_pushButton_2_clicked();

    void on_pushButton_3_clicked();

    void on_lineEdit_editingFinished();

    void on_lineEdit_2_editingFinished();

    void on_lineEdit_3_editingFinished();
```

```
private:
   Ui::MainWindow *ui;
};

#endif // MAINWINDOW_H
```

```
#-------------------------------------------------
#
# Main.cpp
#
#-------------------------------------------------
```

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

/* Librería QTimer*/
#include <QTimer>

/* Variables para tiempo */
int sec = 0;
int min = 0;
int hrs = 0;
```

```
/* Banderas estado de Temporizador */
bool play = false;
bool pause = false;
bool reset = false;



MainWindow::MainWindow(QWidget *parent):QMainWindow(parent),ui(new Ui::MainWindow){
   ui->setupUi(this);

   // Creacion de puntero del tipo QTimer que almacena la configuracion actual
   QTimer *cronometro = new QTimer(this);
   // Funcion para conectar el timer
   connect(cronometro, SIGNAL(timeout()), this, SLOT(FTimer()));
   // Inicialización de timmer acelerado, para efectos de demostracion
   cronometro -> start(10);

}

// Destructor
MainWindow::~MainWindow(){
   delete ui;
}

// Funcion de conteo
void MainWindow::FTimer(){
```

```
  if(play){                                          // Si play es presionado
    sec--;                                           // Comienzan a descender los segundos
uno a uno
    if(sec < 1 ){                          // Si segundos llega a cero
      if(min){                             // Si hay algun minuto
        min--;                             // se resta 1
        sec = 59;                          // Y se convierten los segundos al minuto
      }

      else{
                                           // Si no hay mas minutos
        if(hrs){                   // Revisa si hay alguna hora
          hrs--;                           // Si si, resta 1
          min = 59;                // y convierte minutos
        }

        else{                              // Si no hay hora
          reset = true; // El conteo ha terminado
          play = false;
          // MOSTRAR IMAGEN VERDE
        }
      }
    }
    else if(min < 1 ){                     // Si los minutos llegan a cero
      if(hrs){                             // Revisa si hay hora
```

```
            hrs--;                          // Si si, resta 1
            min = 59;                       // y convierte los minutos
        }
    }

}
else if(pause){                                     // Boton de pausa
        // MOSTRAR IMAGEN AMARILLA
}
else if(reset){                                 // Si se presiona reset
    sec = 0;                                    // Las variables regresan a su estado inicial
    min = 0;
    hrs = 0;
    ui->lineEdit->clear();              // Los elementos graficos regresan a su estado inicial
    ui->lineEdit_2->clear();
    ui->lineEdit_3->clear();
    reset = false;                              // Las banderas cambian
    pause = true;

}

// Cada ciclo imprime los valores en los LCD
ui->lcdNumber->display(hrs);
ui->lcdNumber_2->display(min);
ui->lcdNumber_3->display(sec);
```

```
}



// Boton Play
void MainWindow::on_pushButton_clicked(){
    play = true;
    pause = false;
    reset = false;
}
// Boton Pause
void MainWindow::on_pushButton_2_clicked(){
    play = false;
    pause = true;
    reset = false;
}
// Boton Reset
void MainWindow::on_pushButton_3_clicked(){
    play = false;
    pause = false;
    reset = true;
}

// LineEdit Hrs
void MainWindow::on_lineEdit_editingFinished(){
    hrs = ui->lineEdit->text().toInt();
    ui->lcdNumber->display(hrs);
```

```
}
// LineEdit Min
void MainWindow::on_lineEdit_2_editingFinished(){
    min = ui->lineEdit_2->text().toInt();
    ui->lcdNumber_2->display(min);
}
// LineEdit Sec
void MainWindow::on_lineEdit_3_editingFinished(){
    sec = ui->lineEdit_3->text().toInt();
    ui->lcdNumber_3->display(sec);
}
```