Computer Vision

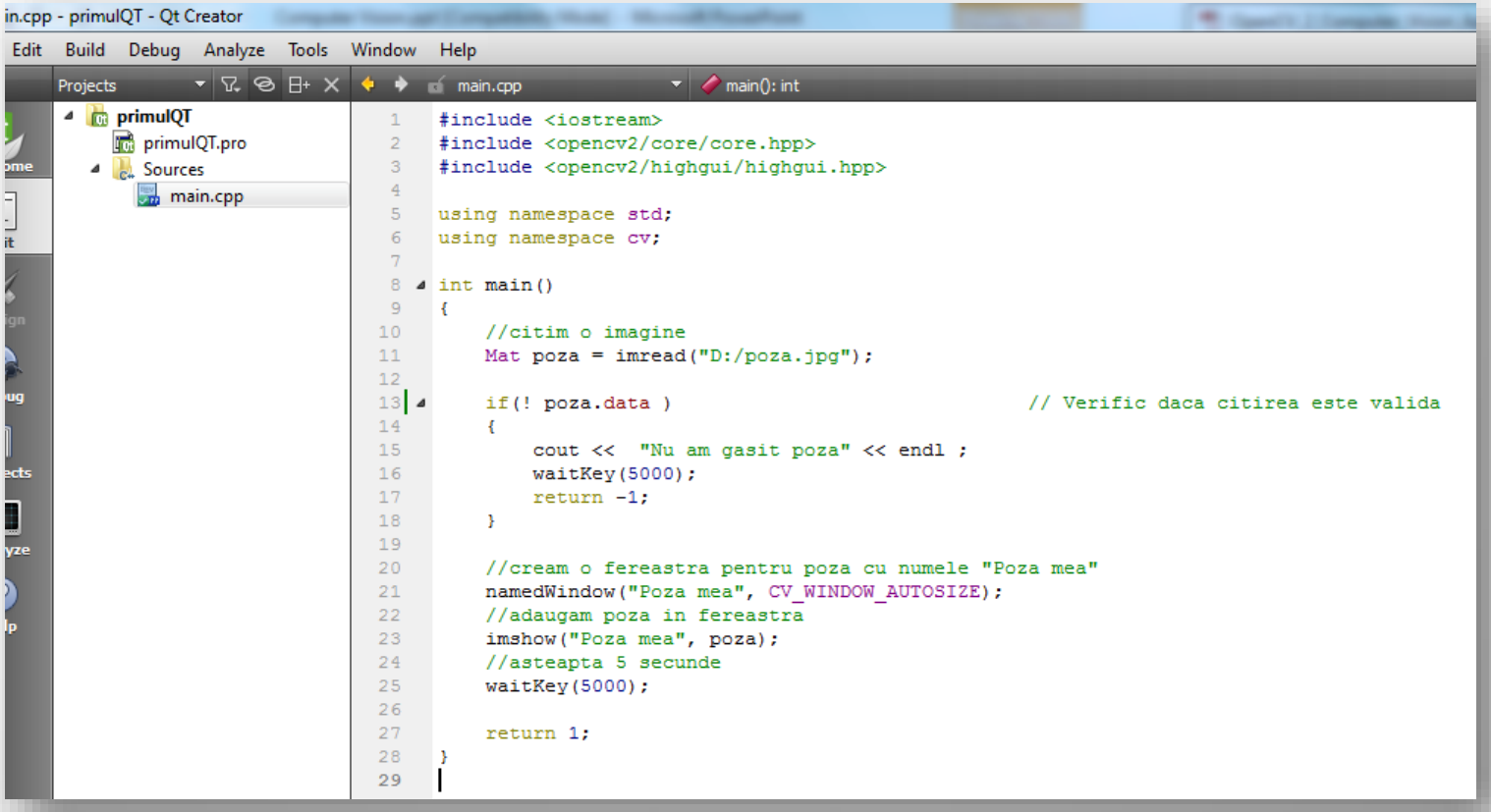
Catalin Stoean
catalin.stoean@inf.ucv.ro
http://inf.ucv.ro/~cstoean

Obiective

- Cele mai uzuale module din OpenCV
- Sa intelegem cateva din obiectele si metodele cele mai utilizate in OpenCV
- Oglindirea unei imagini
- Rularea unui fisier video

Proiect OpenCV folosind QT

- Ne amintim aplicatia de data trecuta.



Module OpenCV

- **opencv_core** – contine functionalitati de baza ale librariei, structuri de baza si functii aritmetice
- **opencv_imgproc** – contine principalele functii de procesare ale imaginilor
- **opencv_highgui** – contine functii pentru citirea si scrierea fisierelor cu imagini si video si alte functii pentru interfata
- **opencv_features2d** – contine functii pentru detectarea de puncte de interes

Module OpenCV

- **opencv_calib3d** – contine functii pentru calibrarea camerei, estimare geometrica pentru doua perspective, functii stereo
- **opencv_video** – contine functii despre estimarea miscarii, urmarire de caracteristici, extragere de informatii din prim plan.
- **opencv_objdetect** – contine functii pentru detectare de obiecte (de exemplu, a feței)

Stocarea unei imagini in OpenCV

- Obiectele din cadrul OpenCV utilizate pana acum sunt de tip cv.
  - Acesta este motivul pentru care am adaugat **using namespace cv;**
- Stocarea unei imagini se face intr-un obiect de tip **Mat**.
  - `Mat poza; //daca avem in cadrul programului using namespace cv`
  - `cv::Mat poza; //altfel`
- Definirea creeaza o imagine de marime 0 x 0.
- Pentru a vedea dimensiunea imaginii, putem apela metoda **size()** a lui Mat:

```
cout<<"Inaltimea este de "<<poza.size().height<<
      ", iar lungimea de "<<poza.size().width<<"."<<endl;
```

- De adaugat aceasta bucata de cod la programul cu afisarea pozei.

Citirea unei imagini

- Se utilizeaza metoda **imread** cu argumentul dat de un string ce contine calea catre fisierul cu poza.
  - `Poza = imread("D:/yes.jpg");`
  - Poza este citita din fisier, decodata si i se aloca memorie
- Pentru a verifica daca a fost buna calea sau formatul este acceptat
  - `if(! poza.data )`
- Pointerul **data** este 0 atunci cand nu s-a citit imaginea.

Afisarea imaginii

- Este utilizat modulul highgui al OpenCV pentru a afisa imaginile.
- `namedWindow("poza mea", CV_WINDOW_AUTOSIZE);`
- `imshow("poza mea", poza);`
- **namedWindow** declara fereastra in care se doreste afisarea pozei
- **imshow** specifica faptul ca poza ar trebui sa apara in acea fereastra.

Mat si IplImage

- Odata cu OpenCV 2, o noua interfata C++ a fost introdusa.
- In interfata anterioara, imaginile erau reprezentate folosind structura IplImage.
- Se poate face insa convertire intre ImpIImage si Mat:
- `IplImage* iplImage = cvLoadImage("c:\\img.jpg");`
- `Mat poza(iplImage);`
- Pentru multitudinea de constructori Mat vedeti: http://docs.opencv.org/modules/core/doc/basic_structures.html#mat-mat

Oglindirea unei imagini

- Pentru a nu modifica poza initiala, se declara o noua variabila de tip **Mat**.
- `Mat rezultat;`
- Se apeleaza apoi metoda flip
- `flip(poza, rezultat, 1)`
- Apoi se afiseaza cea de a doua imagine:
- `namedWindow("poza oglindita", CV_WINDOW_AUTOSIZE);`
- `imshow("poza oglindita", rezultat);`
- `waitKey(0);//pentru a astepta atingerea //unei taste`

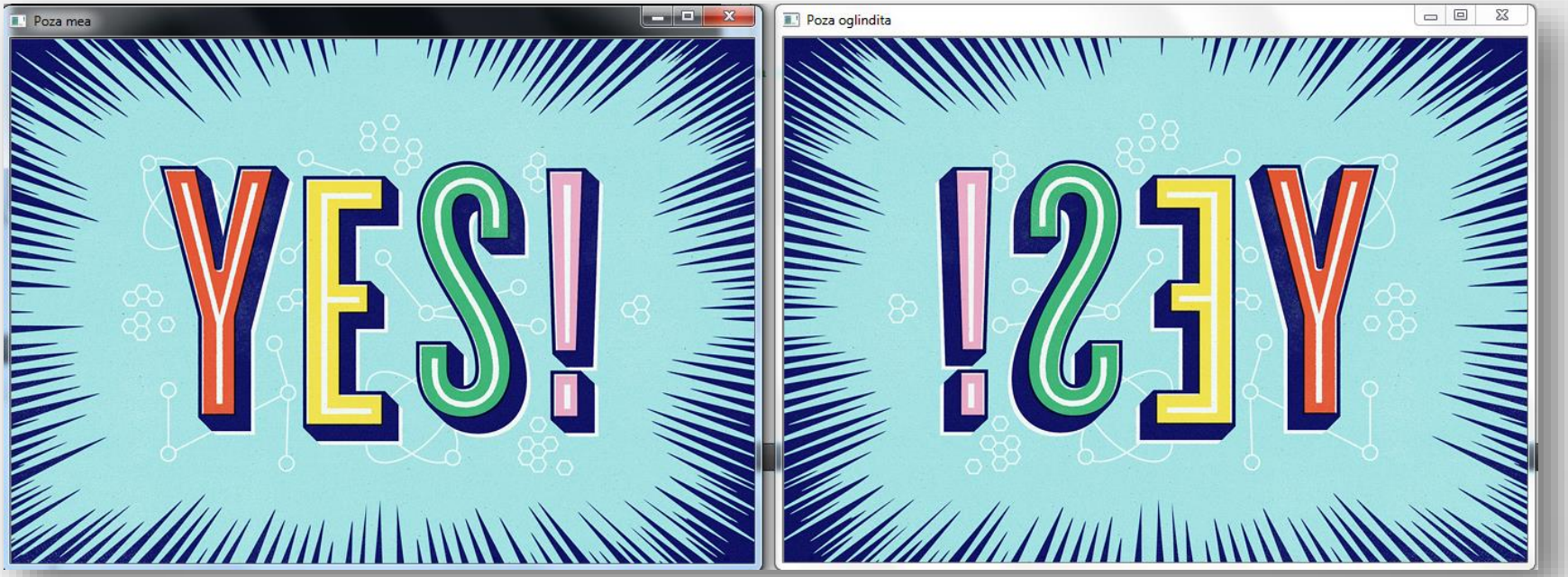
Oglindirea unei imagini

- Cel de-l treilea argument de la metoda **flip** poate fi
  - Pozitiv: oglindire orizontala
  - Zero: oglindire verticala
  - Negativ: si orizontala, si verticala
- **waitKey** este tot o metoda a modulului highgui.
- Imaginea poate fi salvata folosind metoda **imwrite**.

Oglindirea unei imagini

- `imwrite("D:\oglindita.jpg", rezultat);`
- Primul argument este calea catre fisier si numele, cel de-al doilea variabila de tip Mat care contine poza de salvat.
- Poza poate fi salvata si in alte formate: bmp, png etc. si in functie de extensie se stabileste codecul folosit.

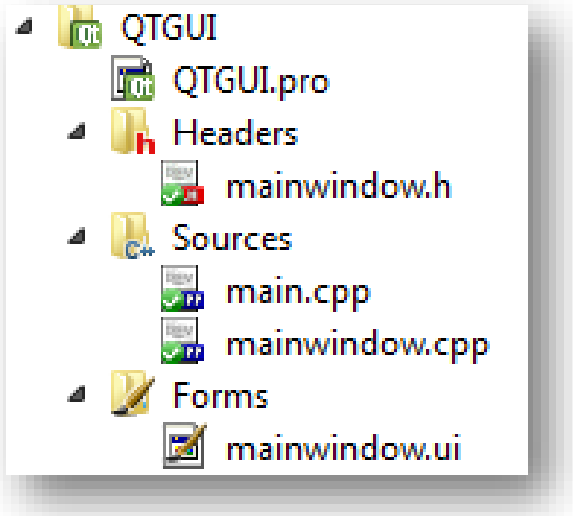
Rezultatul



- Modificati programul pentru a afisa 4 poze concomitent: originala, oglindita in sus, in jos si in ambele directii.
  - Definiti o metoda pe care o apelati de 4 ori pentru a face afisarea unei imagini.

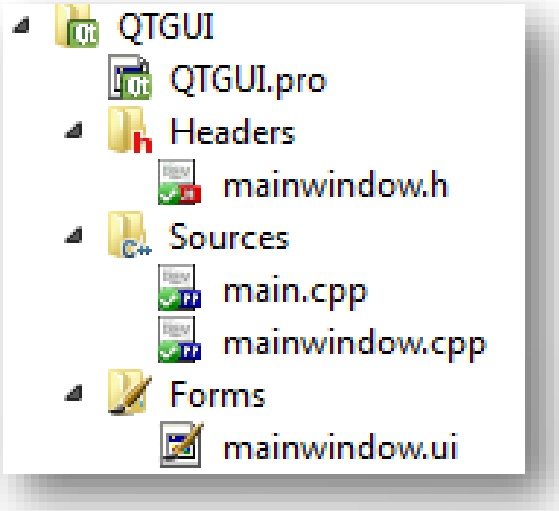
Interfata in Qt pentru oglindirea unei imagini

- Cream un proiect in Qt de tip **QT Widgets Application**.
- In cadrul acestuia se vor crea urmatoarele fisiere:



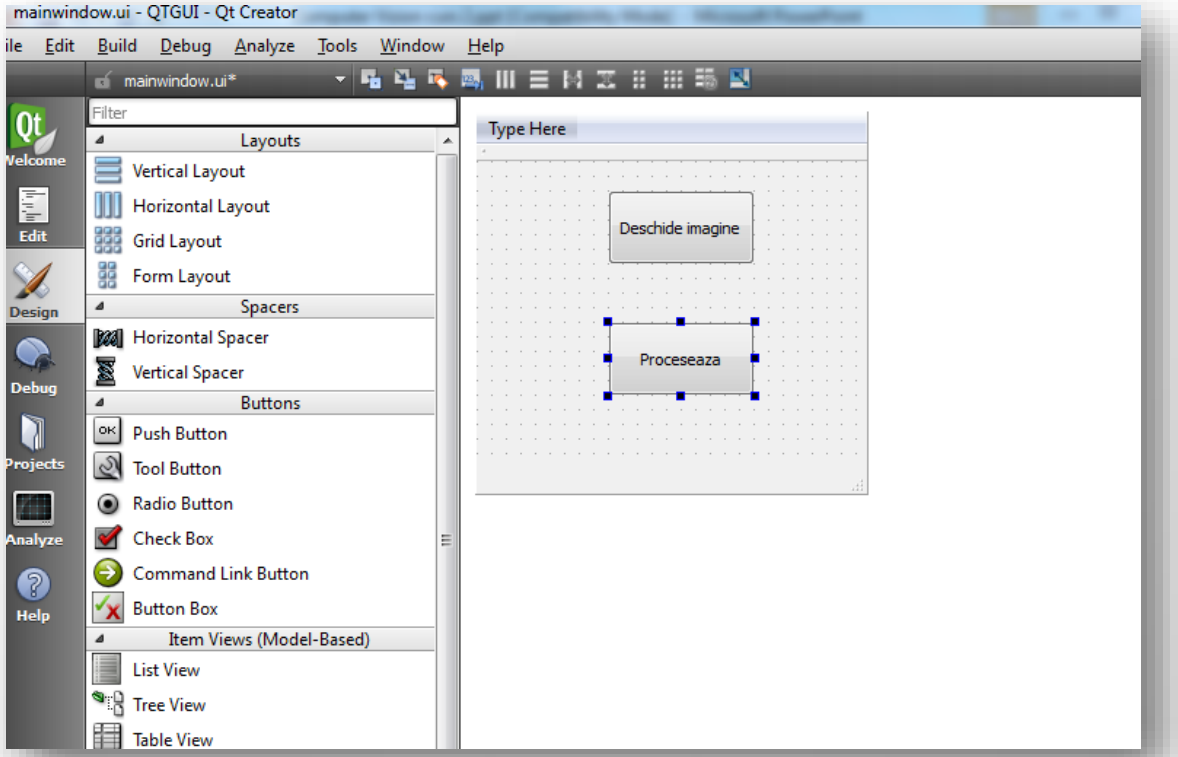
Interfata in Qt pentru oglindirea unei imagini

- In fisierul proiectului (cel cu extensia **.pro**) nu uitam sa adaugam librariile OpenCV.
- Fisierul **mainwindow.cpp** defineste clasa care contine componentele ferestrei GUI.
- Fisierul **mainwindow.ui** descrie aspectul ferestrei GUI.
- Fisierul **mainwindow.h** este fisierul header al clasei mainwindow.



Interfata in Qt pentru oglindirea unei imagini

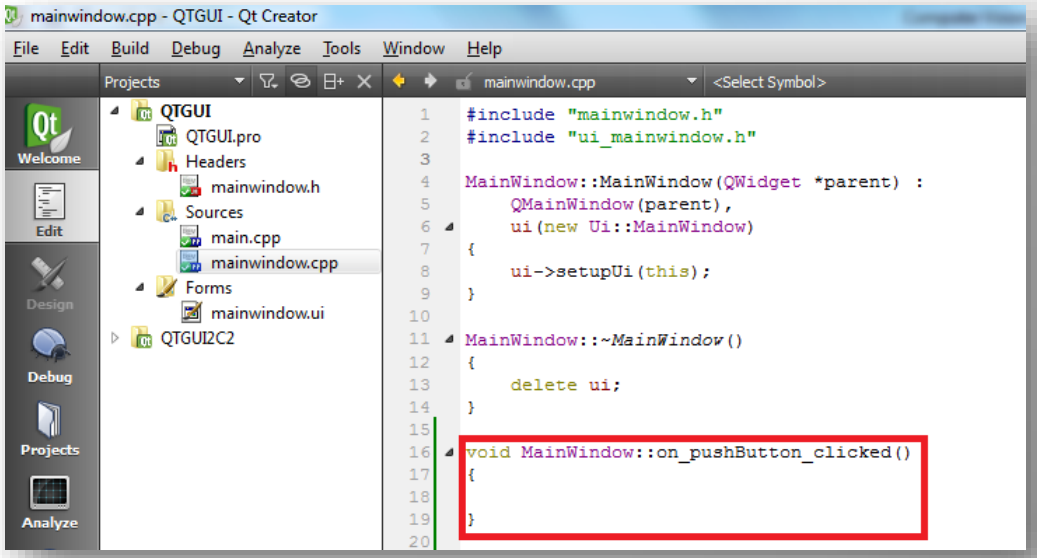
- Dam dublu-click pe fisierul **mainwindow.ui**.
- Adaugam doua butoane carora le editam etichetele.



Interfata in Qt pentru oglindirea unei imagini

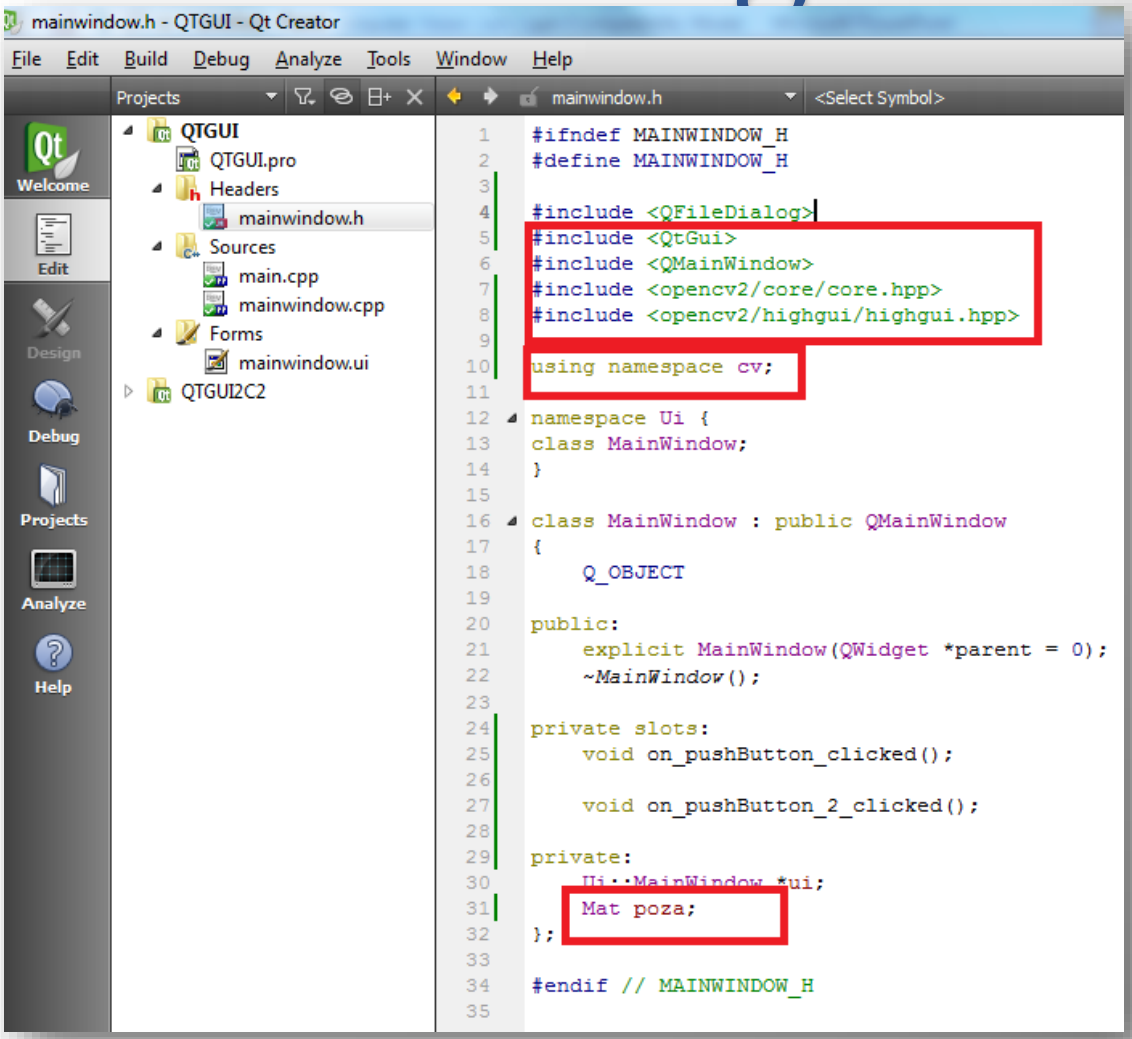
- Click-dreapta pe primul buton **-> Go to slot... -> clicked()**.
- Acesta face sa apara in mainwindow.cpp metoda **on_pushButton_clicked()**.

- Procedam apoi la fel si pentru al doilea buton.



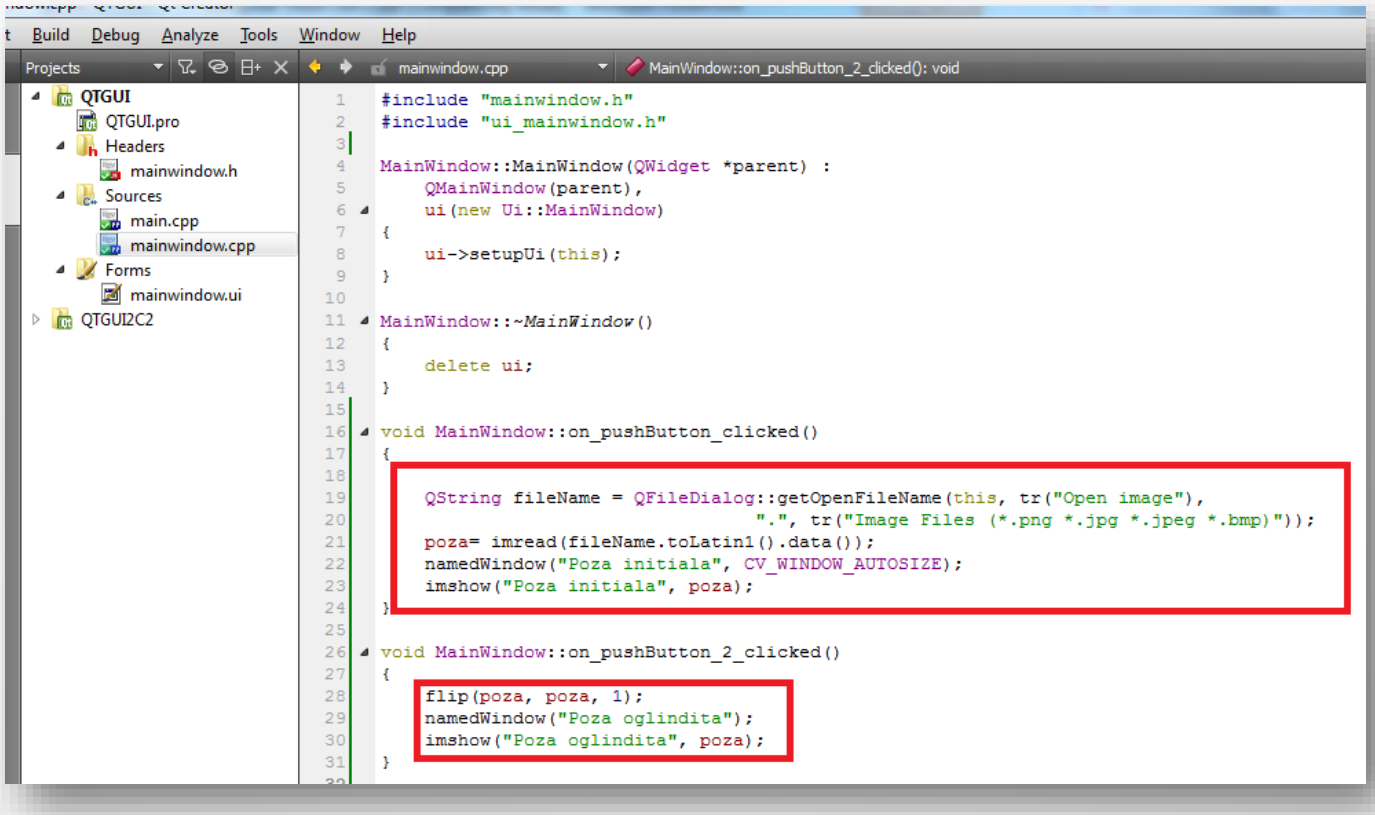
Interfata in Qt pentru oglindirea unei imagini

- In fisierul header, adaugam o variabila de tip Mat si includem modulele care ne sunt necesare.



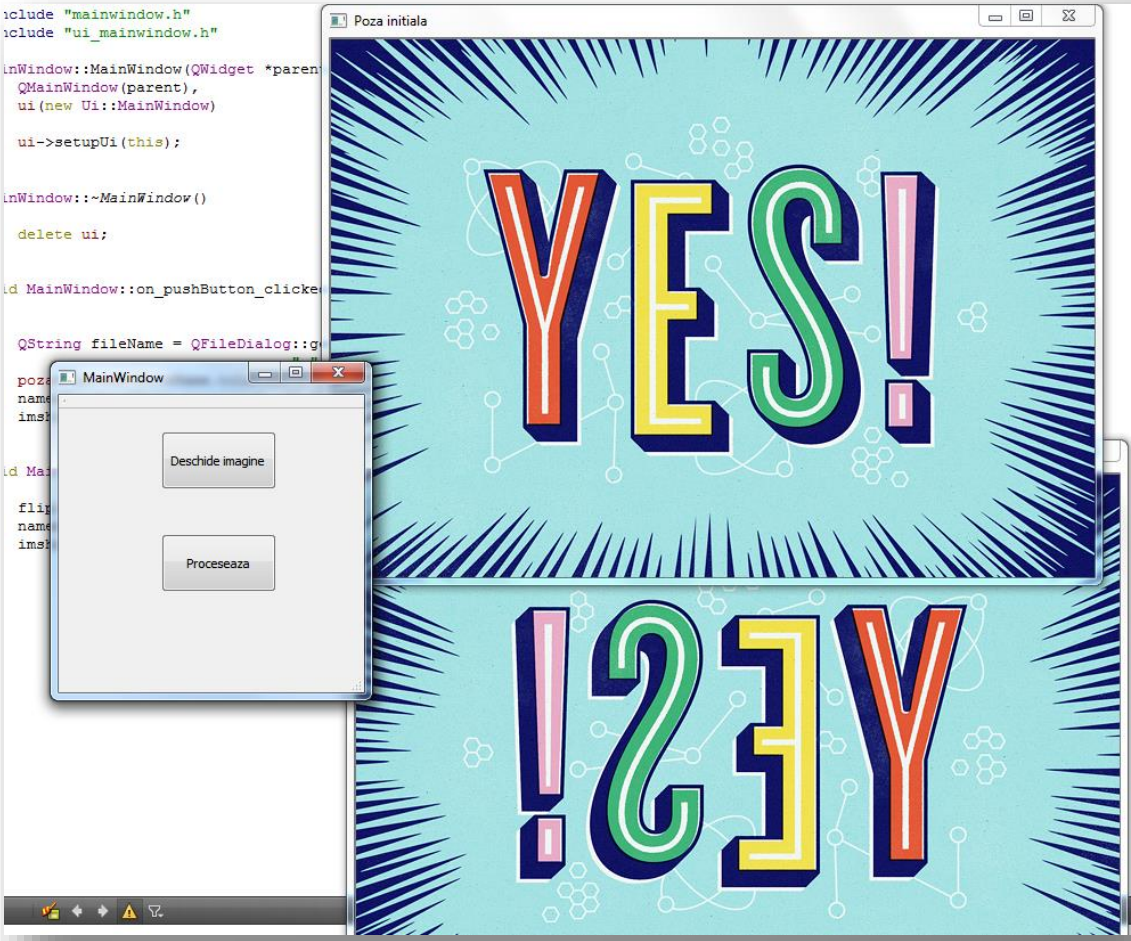
Interfata in Qt pentru oglindirea unei imagini

- In mainwindow.cpp adaugam ce trebuie sa faca aplicatia atunci cand sunt apasate butoanele.



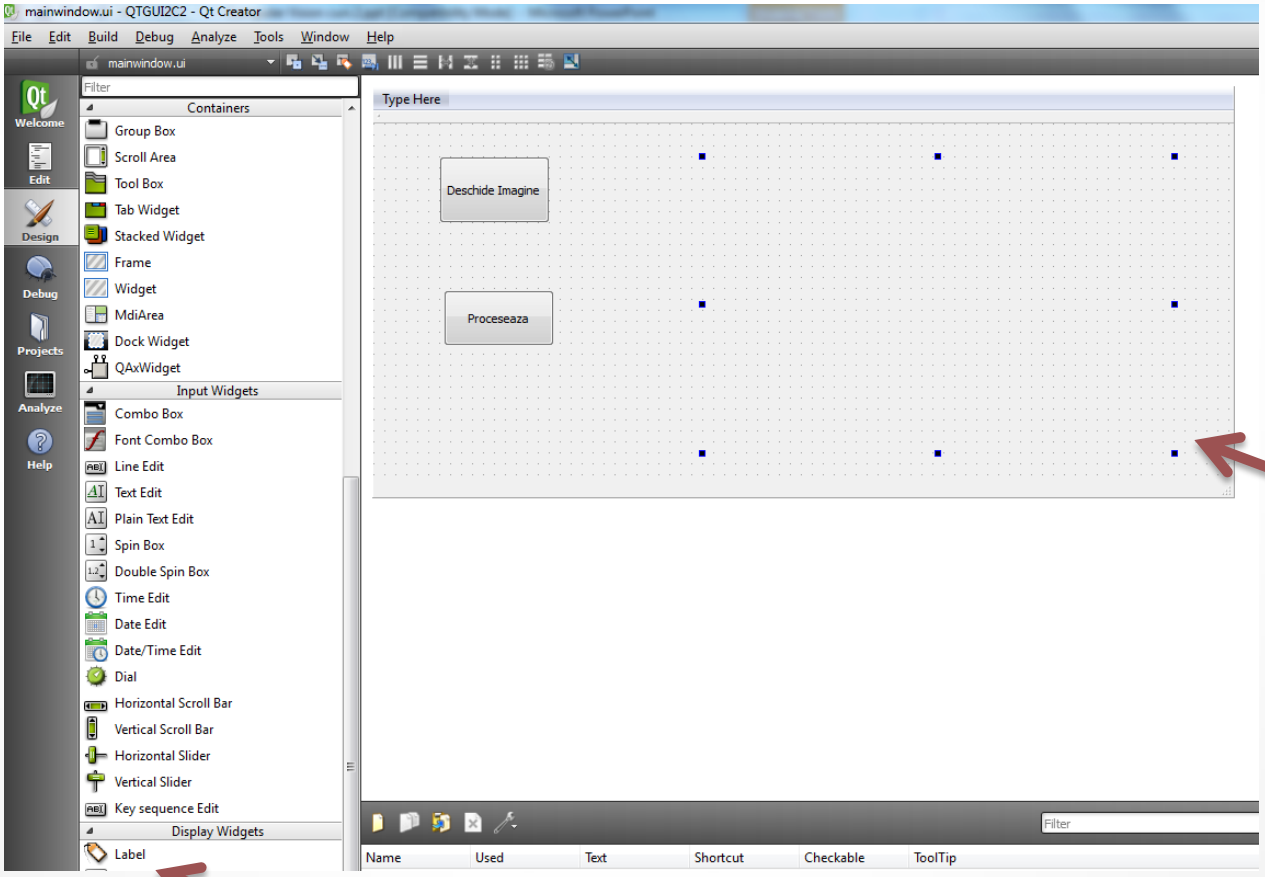
Interfata in Qt pentru oglindirea unei imagini

- Primul buton deschide o fereastra de dialog care ne permite sa selectam fisierul.
- Dupa deschiderea pozei, al doilea buton deschide cea de a doua fereastra cu poza oglindita.



O noua interfata in Qt pentru oglindirea unei imagini

- Cream un nou proiect in care adaugam si un label.



O noua interfata in Qt pentru oglindirea unei imagini

- La acel label stergem textul prestabilit.
- Intentia este sa adaugam poza chiar in locul etichetei, in aceeasi fereastra cu butoanele.
- Accesul catre eticheta se face prin intermediul atributului **label** al lui **ui**, adica **ui->label**.
- Pentru a adauga imaginea in fereastra Qt, trebuie sa facem o corespondenta intre QImage* si Mat.
  - Este necesara inversarea celor 3 canale de culori din BGR in RGB.
  - Pentru aceasta se foloseste metoda cvtColor

*Pentru mai multe detalii despre QImage, vedeti documentatia: http://qt-project.org/doc/qt-4.8/qimage.html

O noua interfata in Qt pentru oglindirea unei imagini

- In plus, dezactivam butonul **Proceseaza** pana cand imaginea nu este incarcata.
- Acest lucru il facem adaugand in constructorul MainWindow urmatoarea line de cod:
- `ui->pushButton_2->setEnabled(false);`
- In plus, la evenimentul pentru apasarea primului butonul, adaugam
- ```
  if (poza.data)
      ui->pushButton_2->setEnabled(true);
  ```
  ```

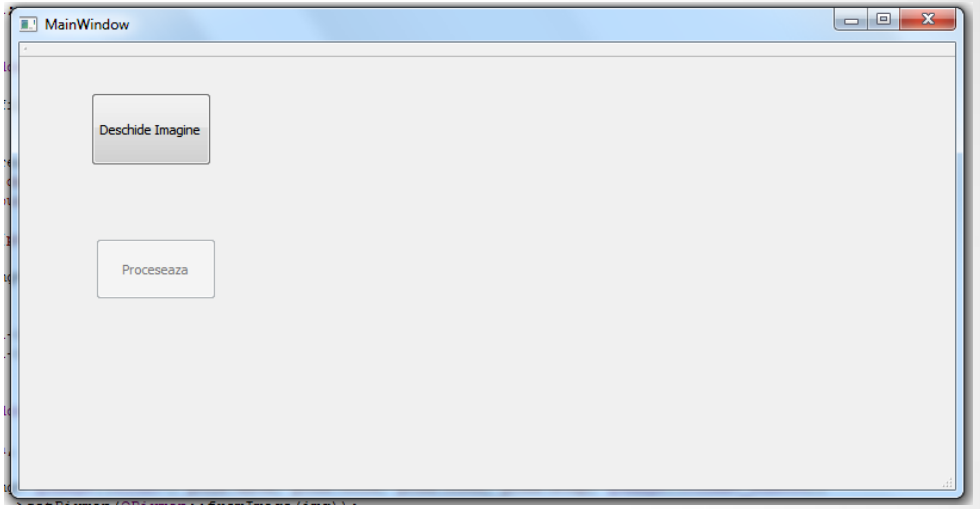
O noua interfata in Qt pentru oglindirea unei imagini

- Atributul **`step`** al obiectului poza contine lungimea in numarul de bytes a unei linii.
- In final, **mainwindow.cpp** va arata astfel:

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

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    ui->pushButton_2->setEnabled(false);
}

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

void MainWindow::on_pushButton_clicked()
{
    QString fileName = QFileDialog::getOpenFileName(this,
                       tr("Open image"), ".",
                       tr("Image Files (*.png *.jpg *.jpeg *.bmp)"));
    poza= imread(fileName.toLatin1().data());
    if (poza.data)
        ui->pushButton_2->setEnabled(true);

    cvtColor(poza, poza, CV_BGR2RGB);

    QImage img= QImage((uchar*) poza.data, poza.cols,
                       poza.rows, poza.step, QImage::Format_RGB888);

    ui->label->setPixmap(QPixmap::fromImage(img));
    ui->label->resize(ui->label->pixmap()->size());
}

void MainWindow::on_pushButton_2_clicked()
{
    flip(poza, poza, 1);

    QImage img= QImage((uchar*) poza.data, poza.cols, poza.rows, poza.step, QImage::Format_RGB888);
    ui->label->setPixmap(QPixmap::fromImage(img));
}
```

O noua interfata in Qt pentru oglindirea unei imagini

- In prima faza, butonul **Proceseaza** este dezactivat, iar dupa incarcarea pozei acesta este activat.
- La apasarea sa, se obtine oglindirea pozei.



Exercitii

1. (1p) Modificati proiectul anetrior astfel incat in afara de butonul de incarcare a pozei sa existe inca 3 alte butoane:
   - Unul pentru oglindire orizontala
   - Unul pentru oglindire verticala
   - Unul pentru ambele oglindiri

   Termen: 22 oct

2. (1p) Pornind de la proiectul rezolvat, realizati o aplicatie care sa contina butonul de deschidere a unei imagini si unul care sa inverseze canalele de culori R, G si B.
   - Termen: 22 oct

Cum functioneaza GUI-ul Qt

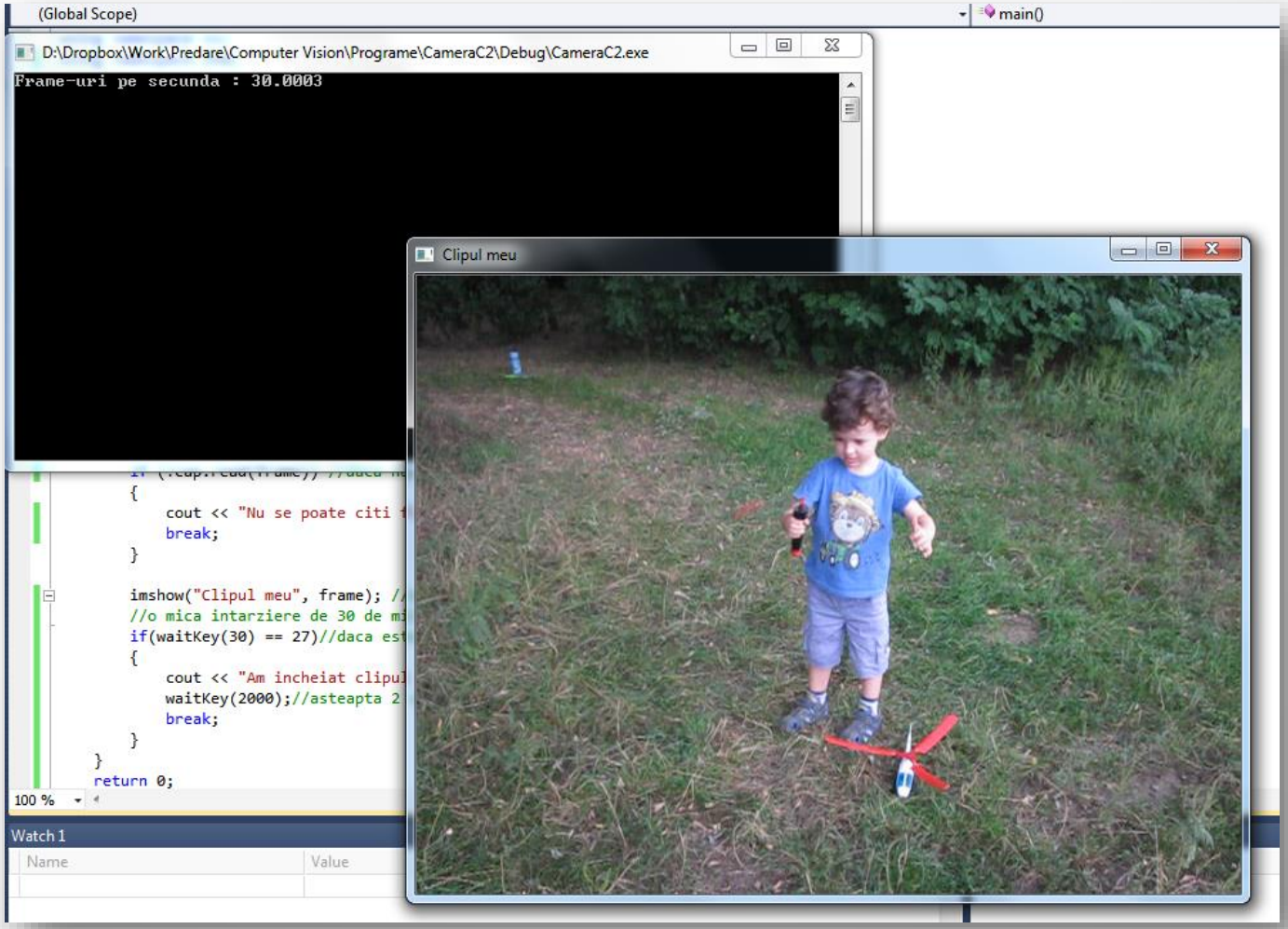
- Obiectele comunica prin semnale si slot-uri.
- Cand are loc un eveniment, un semnal este emis.
- Un slot este o metoda speciala care este apelata cand un semnal la care este conectata este emis.
- Slot-urile sunt definite in fisierul header.
- Accesul catre toate componentele in clasa principala MainWindow se face prin intermediul variabilei **ui** care este declarata in definitia clasei **MainWindow**.

Rularea unui fisier video

- Citim frame-urile din fisier si le redam poza cu poza.
- Dupa fiecare afisare, adaugam o mica intarziere (delay).
  - Daca aceasta este mai mica, filmul merge in ffw

```
using namespace cv;
using namespace std;

int main()
{
    VideoCapture cap("D:/film.avi");

    if ( !cap.isOpened() )
    {
         cout << "Fisierul video nu poate fi rulat" << endl;
         return -1;
    }
     double fps = cap.get(CV_CAP_PROP_FPS); //luam frame-urile pe secunda din clip
     cout << "Frame-uri pe secunda : " << fps << endl;

    namedWindow("Clipul meu",CV_WINDOW_AUTOSIZE);

    while(1)
    {
        Mat frame;

        if (!cap.read(frame)) //daca nu se pot citi frame-urile
        {
            cout << "Nu se poate citi fisierul video" << endl;
            break;
        }

        imshow("Clipul meu", frame); //afisam frame-ul in fereastra "Clipul meu"
        //o mica intarziere de 30 de milisecunde intre frame-uri este necesara. De asemenea, stopam clipul si cu ESC
        if(waitKey(30) == 27)//daca este apasat ESC
        {
            cout << "Am incheiat clipul cu ESC" << endl;
            waitKey(2000);//asteapta 2 secunde
            break;
        }
    }
    return 0;
}
```

Rularea clipului



Exercitiu

- (1p) Creati o interfata in QT care sa contina o fereastra in care se poate porni un fisier video de la un buton **Play** si se poate opri de la alt buton **Stop**.
  - Termen: 22 oct