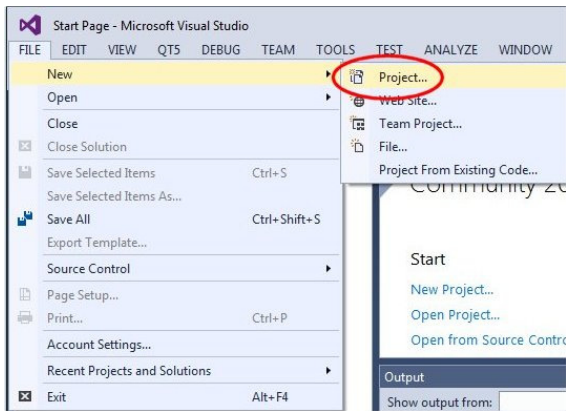


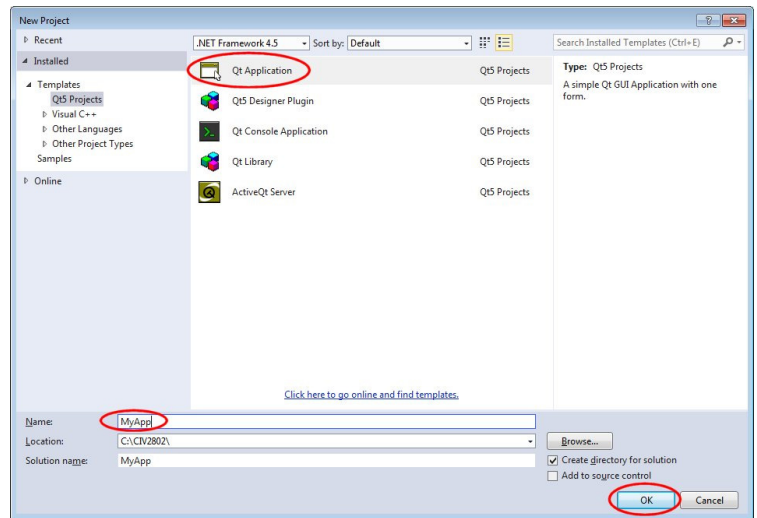
CIV 2802 – Sistemas Gráficos para Engenharia – PUC-Rio

Roteiro para criação de um programa simples para desenho 2D usando Visual Studio 2013 (Community), Qt 5.4 e OpenGL.

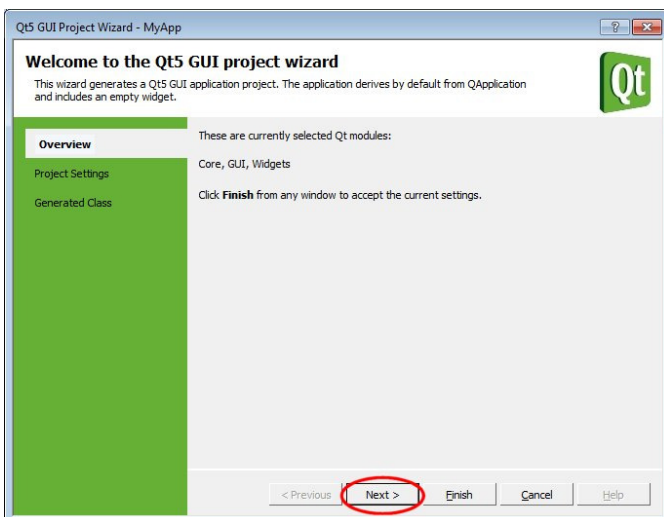
1.



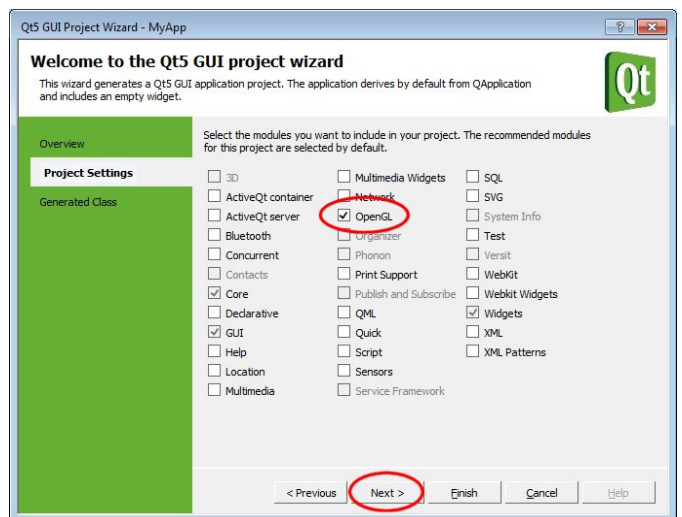
2.



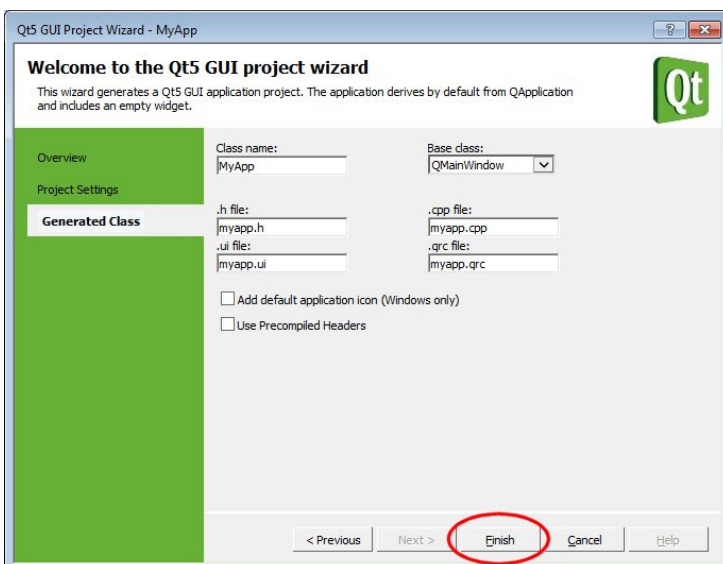
3.



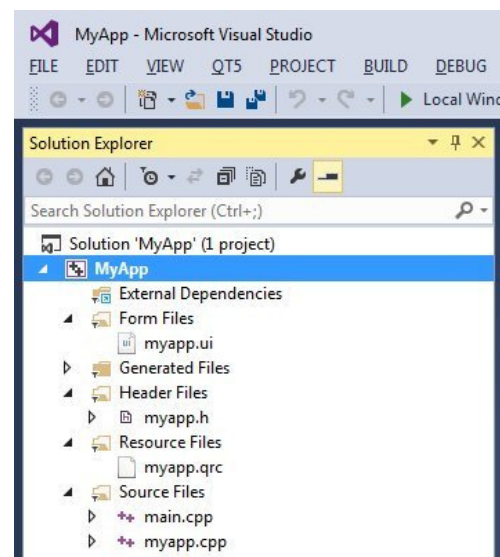
4.



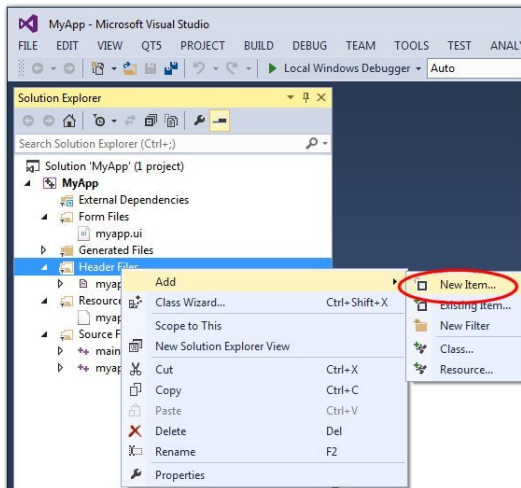
5.



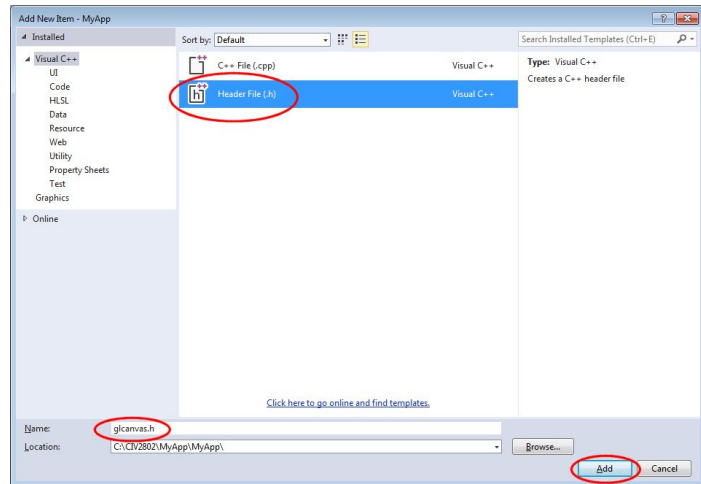
6.



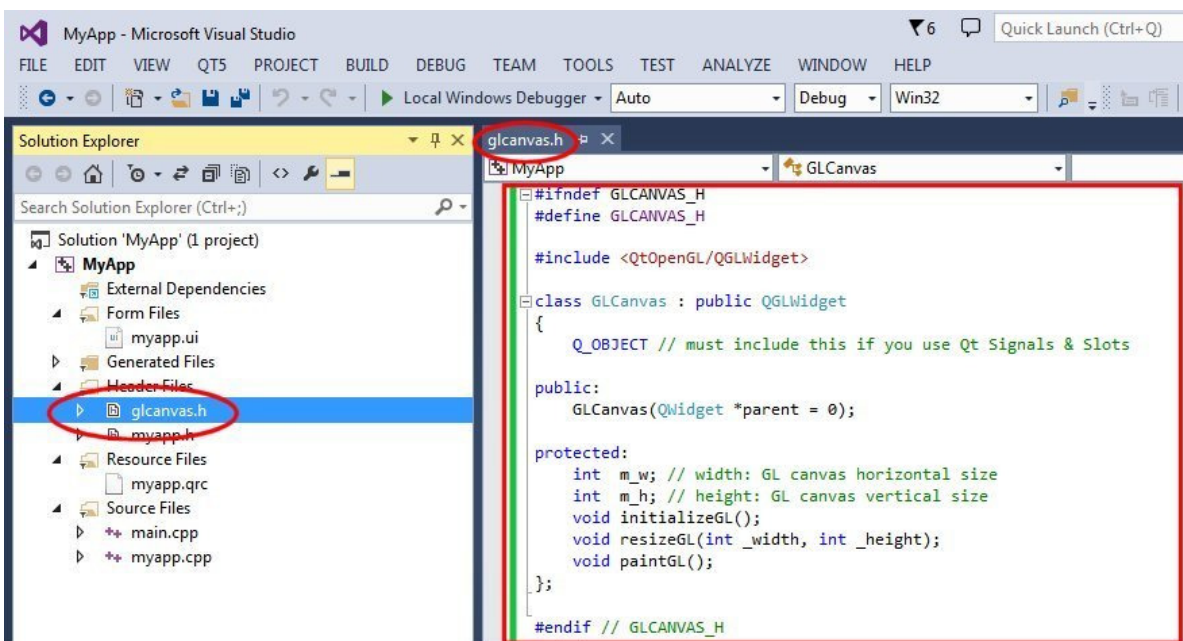
7.



8.



9.



glcanvas.h

```
#ifndef GLCANVAS_H
#define GLCANVAS_H
```

```
#include <QtOpenGL/QGLWidget>
```

```
class GLCanvas : public QGLWidget
{
```

```
    Q_OBJECT // must include this if you use Qt Signals & Slots
```

```
public:
```

```
    GLCanvas(QWidget *parent = 0);
```

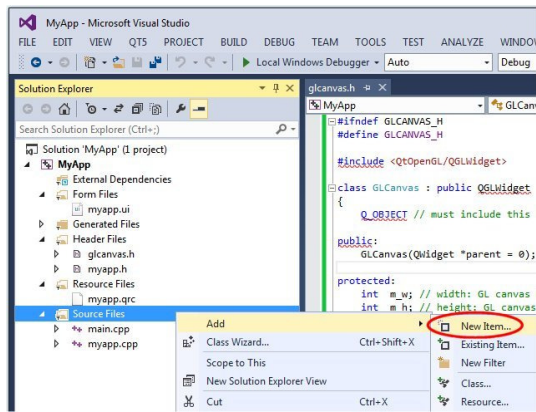
```
protected:
```

```
    int m_w; // width: GL canvas horizontal size
    int m_h; // height: GL canvas vertical size
    void initializeGL();
    void resizeGL(int _width, int _height);
    void paintGL();
```

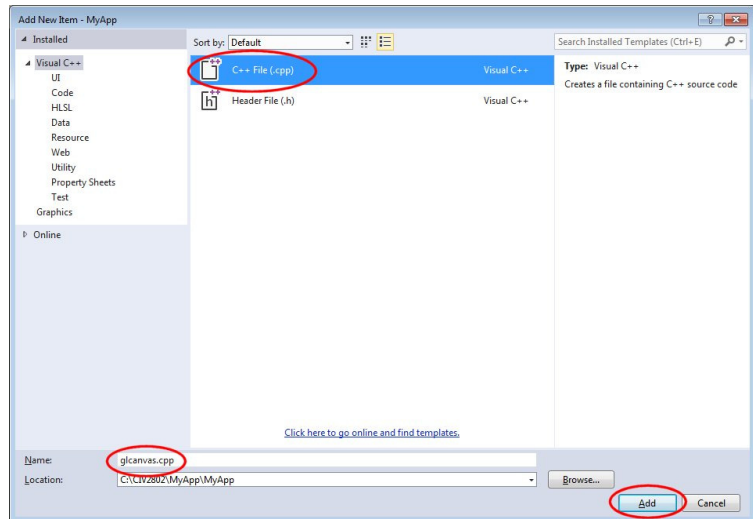
```
};
```

```
#endif // GLCANVAS_H
```

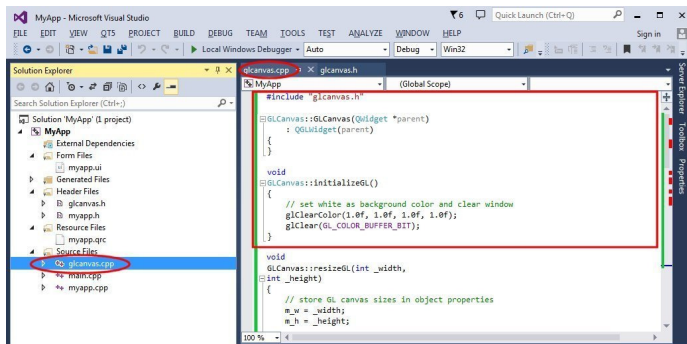
10.



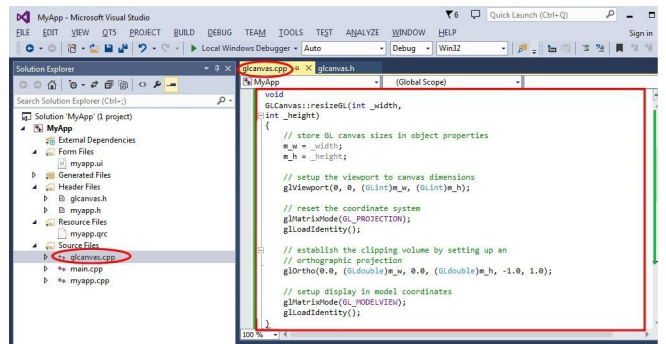
11.



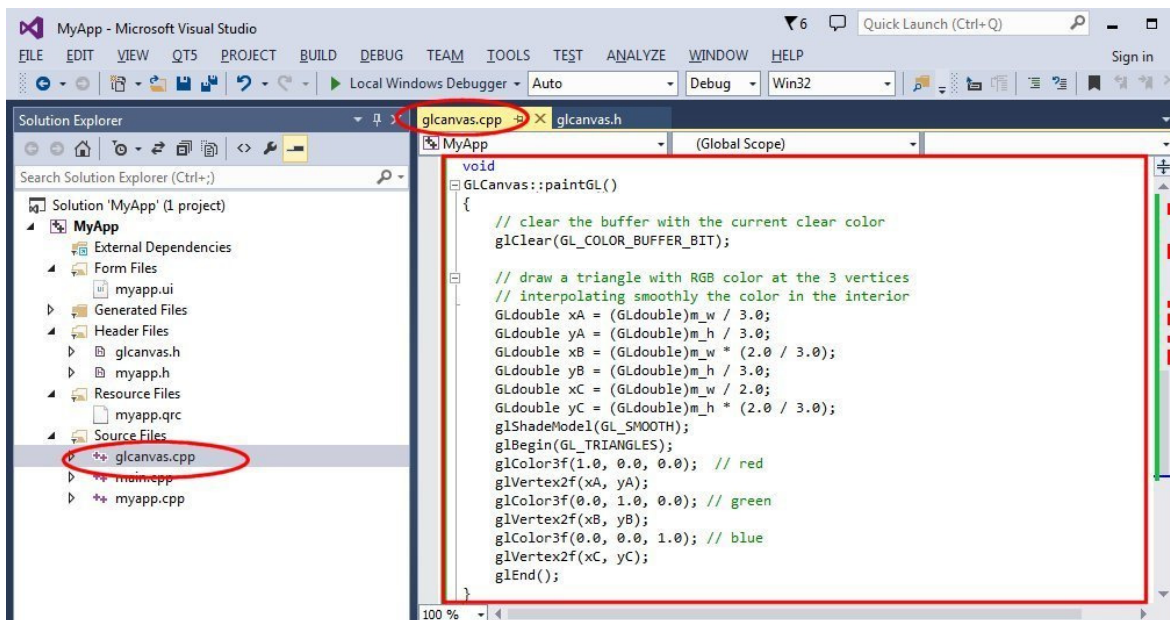
12.



13.



14.



glcanvas.cpp

```
#include "glcanvas.h"

GLCanvas::GLCanvas(QWidget *parent)
    : QGLWidget(parent)
{
}

void
GLCanvas::initializeGL()
{
    // set white as background color and clear window
    glClearColor(1.0f, 1.0f, 1.0f, 1.0f);
    glClear(GL_COLOR_BUFFER_BIT);
}

void
GLCanvas::resizeGL(int _width,
int _height)
{
    // store GL canvas sizes in object properties
    m_w = _width;
    m_h = _height;

    // setup the viewport to canvas dimensions
    glViewport(0, 0, (GLint)m_w, (GLint)m_h);

    // reset the coordinate system
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

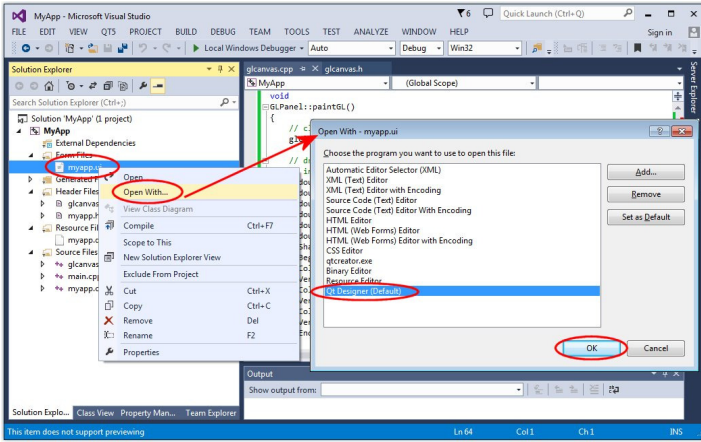
    // establish the clipping volume by setting up an
    // orthographic projection
    glOrtho(0.0, (GLdouble)m_w, 0.0, (GLdouble)m_h, -1.0, 1.0);

    // setup display in model coordinates
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}

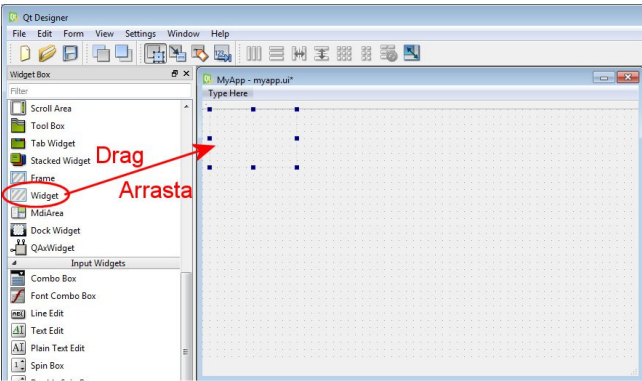
void
GLCanvas::paintGL()
{
    // clear the buffer with the current clear color
    glClear(GL_COLOR_BUFFER_BIT);

    // draw a triangle with RGB color at the 3 vertices
    // interpolating smoothly the color in the interior
    GLdouble xA = (GLdouble)m_w / 3.0;
    GLdouble yA = (GLdouble)m_h / 3.0;
    GLdouble xB = (GLdouble)m_w * (2.0 / 3.0);
    GLdouble yB = (GLdouble)m_h / 3.0;
    GLdouble xC = (GLdouble)m_w / 2.0;
    GLdouble yC = (GLdouble)m_h * (2.0 / 3.0);
    glShadeModel(GL_SMOOTH);
    glBegin(GL_TRIANGLES);
    glColor3f(1.0, 0.0, 0.0); // red
    glVertex2f(xA, yA);
    glColor3f(0.0, 1.0, 0.0); // green
    glVertex2f(xB, yB);
    glColor3f(0.0, 0.0, 1.0); // blue
    glVertex2f(xC, yC);
    glEnd();
}
```

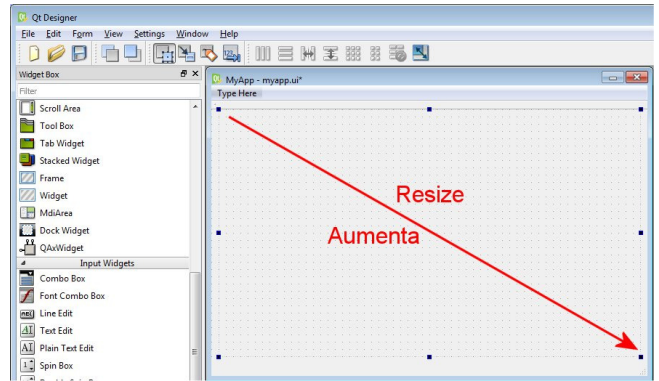
15.



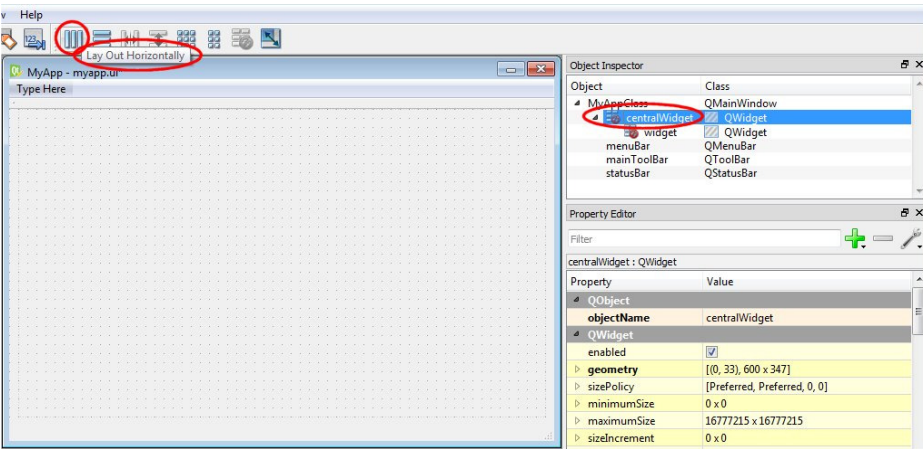
16.



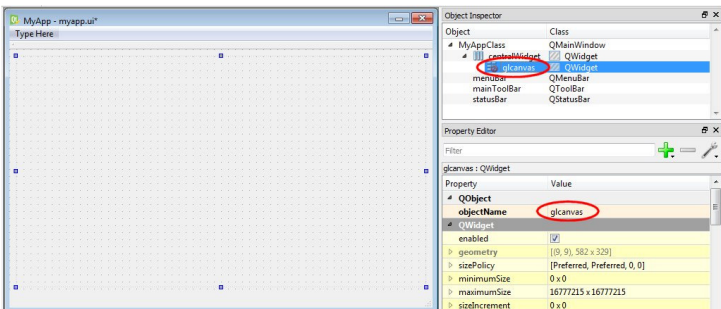
17.



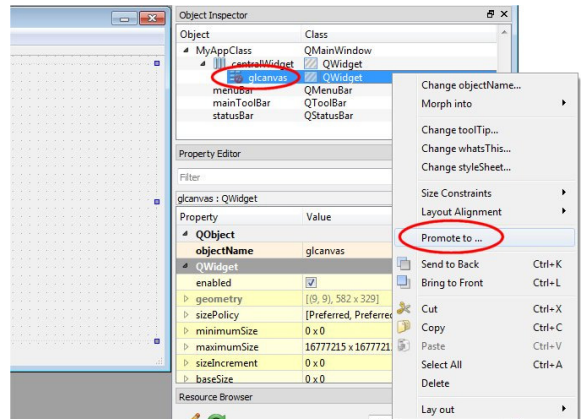
18.



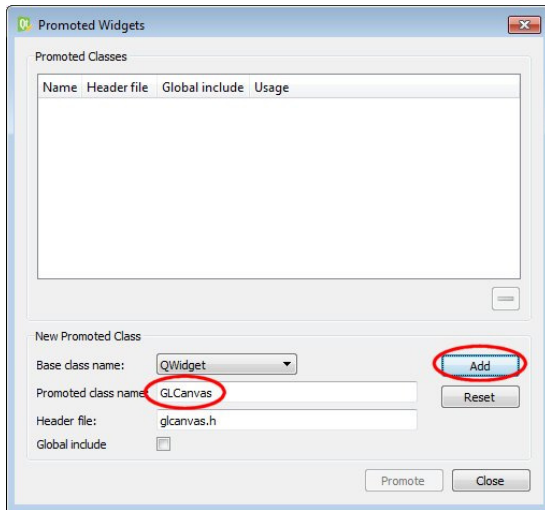
19.



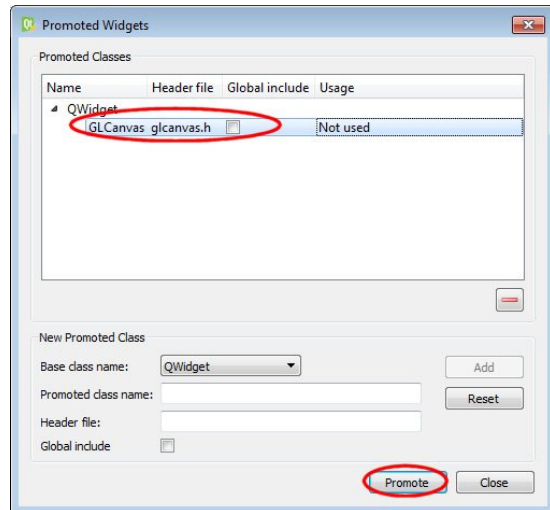
20.



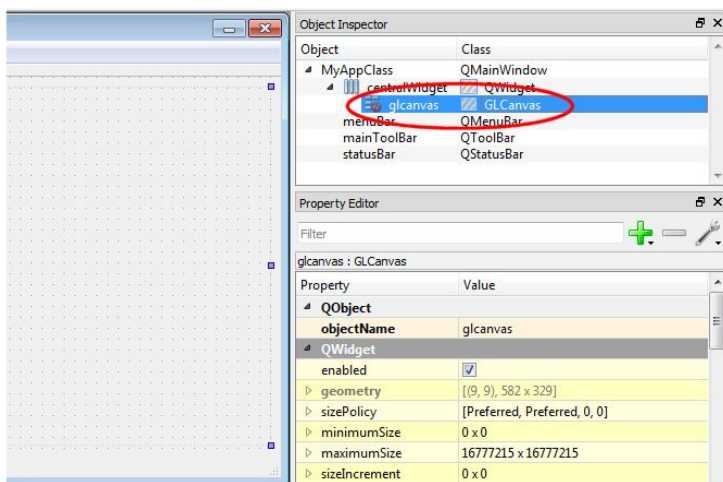
21.



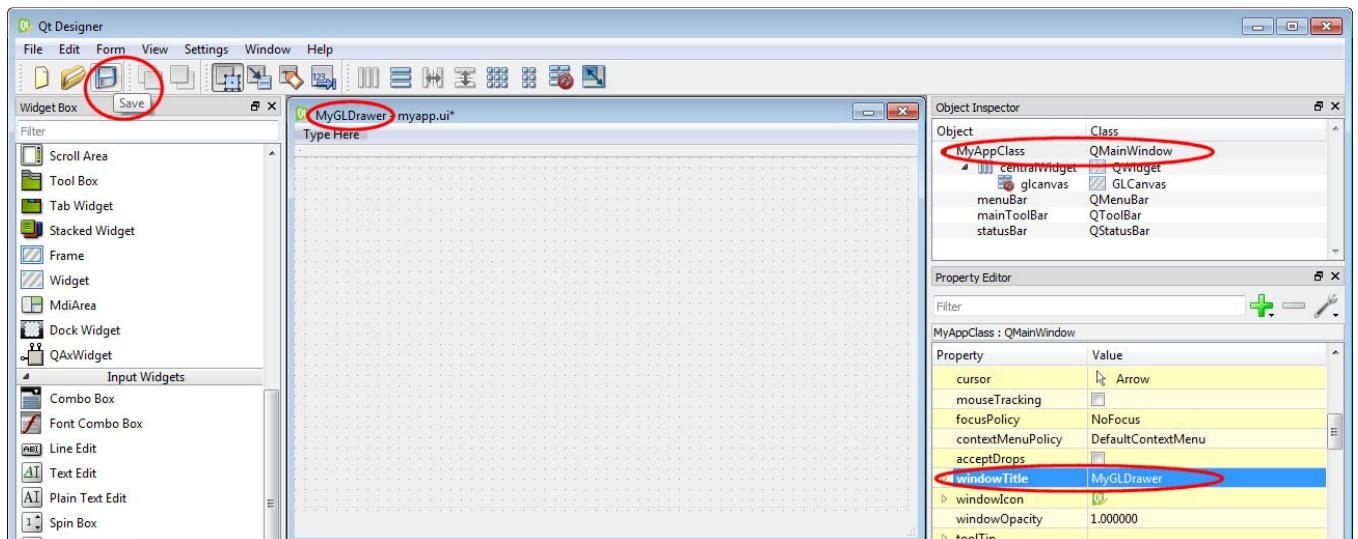
22.



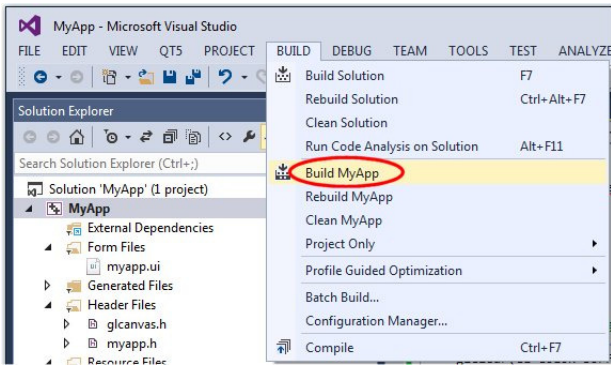
23.



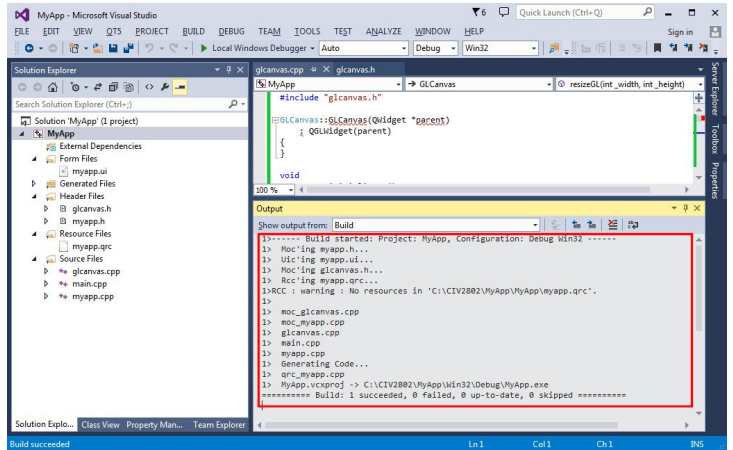
24.



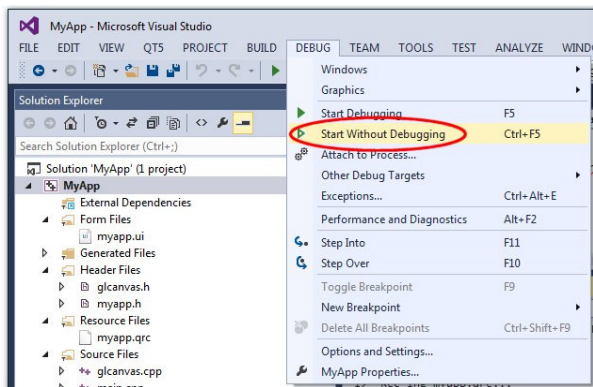
25.



26.



27.



28.

