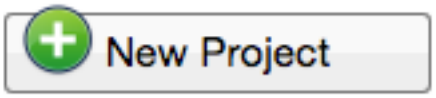
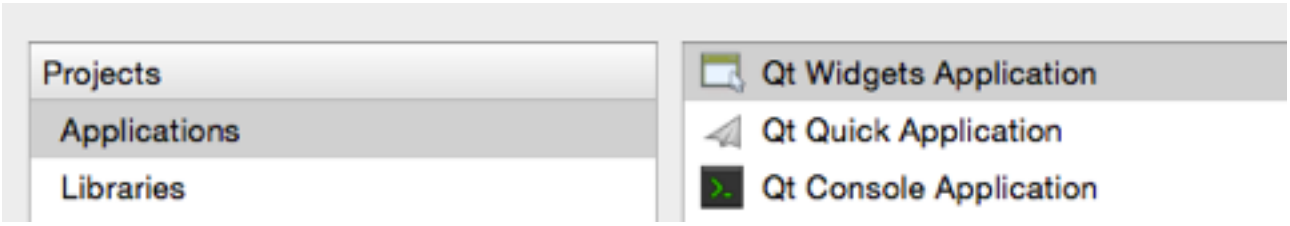
Introducao ao QT Creator

Novo projeto



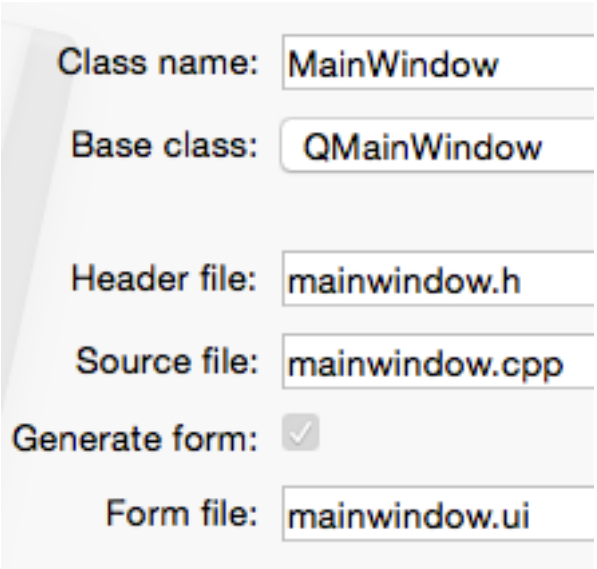
Defina o template (modelo/estilo/tipo) do projeto. No nosso caso escolha Qt Widget Application e click no botão Choose…



Defina o nome do seu projeto (No nosso exemplo: Pessoa). Click no botão Continue.

Escolha os kits contendo as bibliotecas que você precisará:

Especifique as informações básicas sobre as classes para as quais você quer gerar o esqueleto dos arquivos de código fonte.



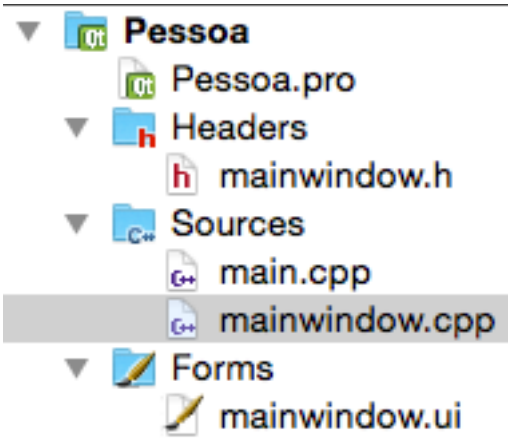
- Class name: MainWindow - Nome da classe que refere-se à sua janela principal;
- Base class: QMainWindow - Super classe (pai) que é herdada pela MainWindow
- Header file: mainwindow.h - Arquivo que contém a definição da classe MainWindow;
- Source file: mainwindow.cpp - Arquivo que contém os métodos da classe MainWindow;
- Generate form: (Deixe marcado para que ele crie automaticamente um form para você.
- Form file: mainwindow.ui - Arquivo XML usado para definir a configuração do seu form;

Click em Continue.

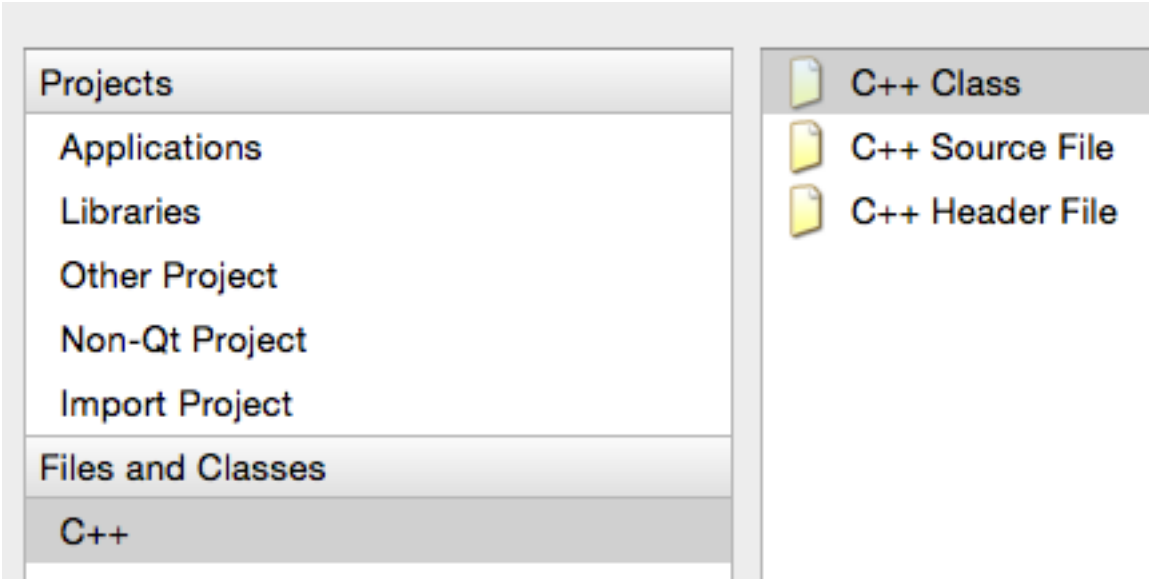
Desta forma seu projeto conterá os seguintes arquivos:

```
Pessoa.pro
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
```

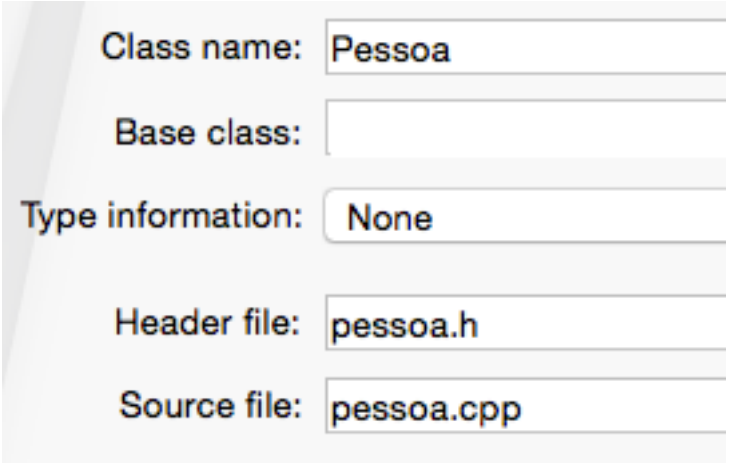
Seu projeto será mostrado na IDE desta forma:



Crie uma novo arquivo para definição da classe Pessoa:



Click em Choose…
Em seguida denomine sua classe como Pessoa:



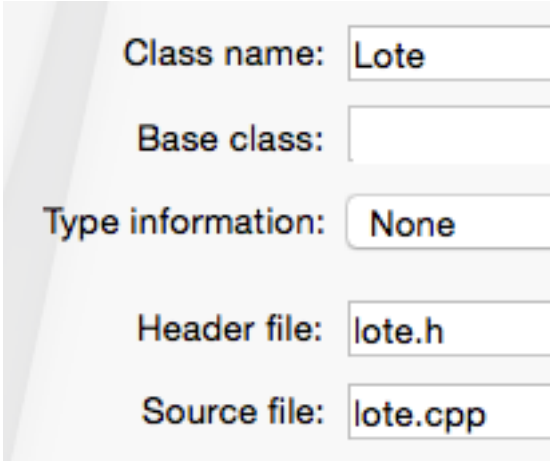
Observe que a IDE criará para você os arquivos pessoa.h e pessoa.cpp.
Click sobre o arquivo pessoa.h e defina-o:

```
#ifndef PESSOA_H
#define PESSOA_H
#include <QString>
class Pessoa
{
public:
    Pessoa();
    void setNome(QString nome);
    QString getNome() const;
    void setEMail(QString eMail);
    QString getEMail()const;
    void setCpf(QString cpf);
    QString getCpf() const;
private:
    QString cpf;
    QString nome;
    QString eMail;
};

#endif // PESSOA_H
```

Click sobre o código pessoa.cpp e defina-o:

```cpp
#include "pessoa.h"

Pessoa::Pessoa()
{
}
void Pessoa::setNome(QString nome){
    this->nome=nome;
}

QString Pessoa::getNome() const{
    return this->nome;
}
void Pessoa::setEMail(QString eMail){
    this->eMail=eMail;
}

QString Pessoa::getEMail()const{
    return this->eMail;
}
void Pessoa::setCpf(QString cpf){
    this->cpf=cpf;
}

QString Pessoa::getCpf() const{
    return this->cpf;
}
```

Defina a segunda classe Lote:



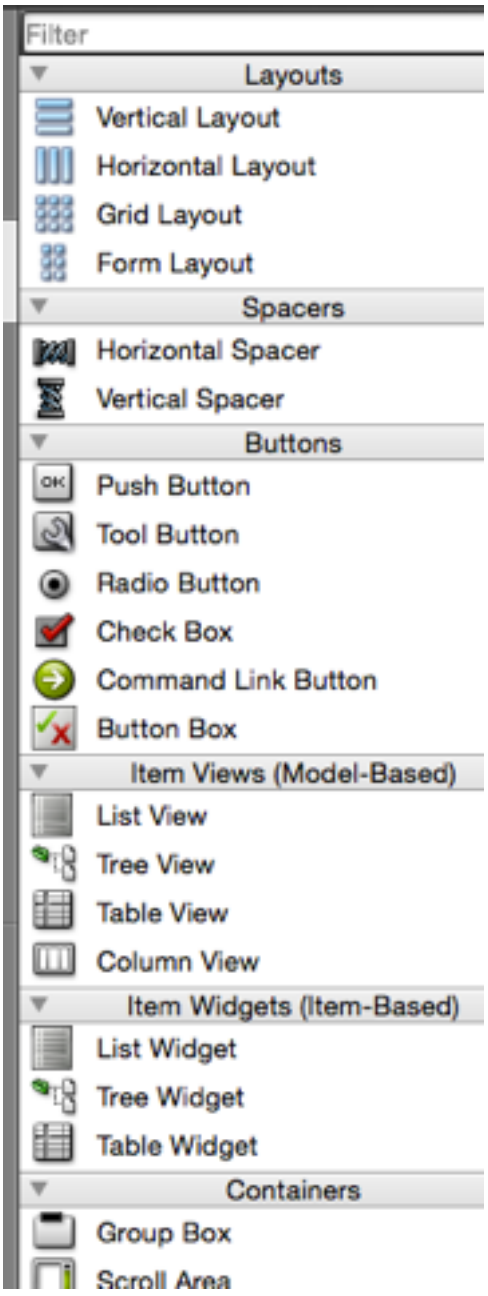
Defina seus arquivos lote.h e lote.cpp conforme código abaixo respectivamente:

```h
#ifndef LOTE_H
#define LOTE_H
#include "pessoa.h"
class Lote
{
public:
    Lote();
    void setFrente(float frente);
    float getFrente()const;
    void setComprimento(float comprimento);
    float getComprimento() const;
    float getArea() const;
    void setProprietario(Pessoa *proprietario);
    Pessoa* getProprietario() const;
private:
    float frente;
    float comprimento;
    Pessoa *proprietario;
};

```

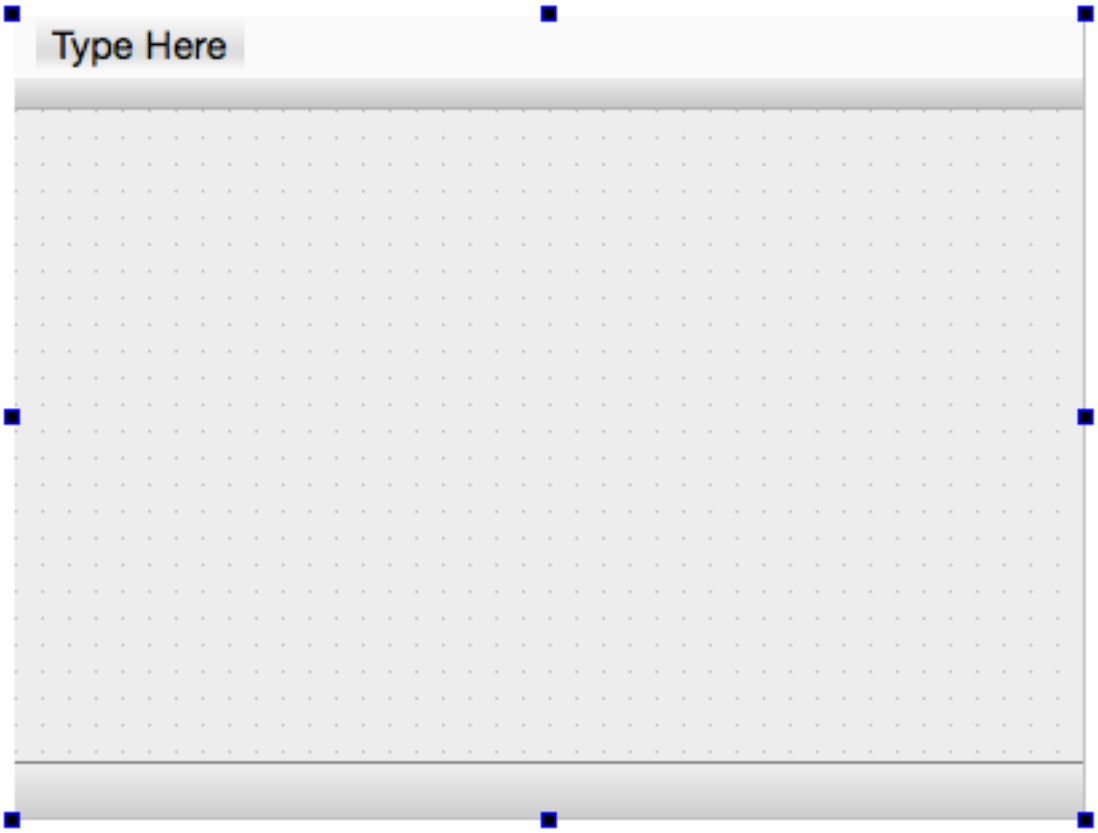
```cpp
#include "lote.h"

Lote::Lote()
{
}
void Lote::setFrente(float frente){
    this->frente=frente;
}

float Lote::getFrente()const{
    return this->frente;
}
void Lote::setComprimento(float comprimento){
    this->comprimento=comprimento;
}

float Lote::getComprimento() const{
    return this->comprimento;
}
float Lote::getArea() const{
    return this->comprimento*this->frente;
}
void Lote::setProprietario(Pessoa *proprietario){
    this->proprietario=proprietario;
}

Pessoa* Lote::getProprietario() const{
    return this->proprietario;
}
```

Para testar o que foi feito até agora, salve todos os arquivos e em seguida click sobre o botão:

Note que caso tudo esteja correto aparecerá uma janela que corresponde ao seu software, porém apenas com as formas de interação padrão das janelas. Por isso vamos agora à elaboração de nossa interface. Para isso, click duas vezes em mainwindow.ui.
Observe a aba Widget Box situada a esquerda da sua IDE:



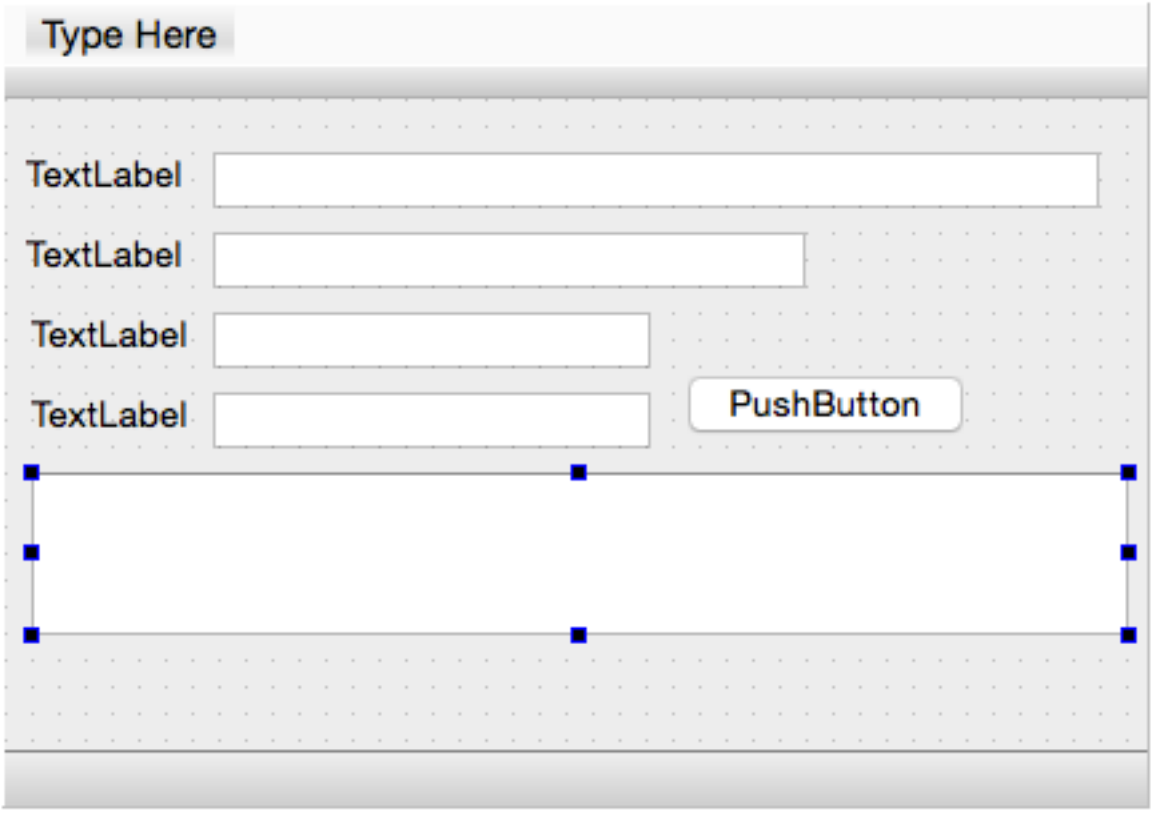
Esta aba contém todos os componentes que você poderá incrementar na sua janela (form). Segue sua form, situada no meio da sua IDE:



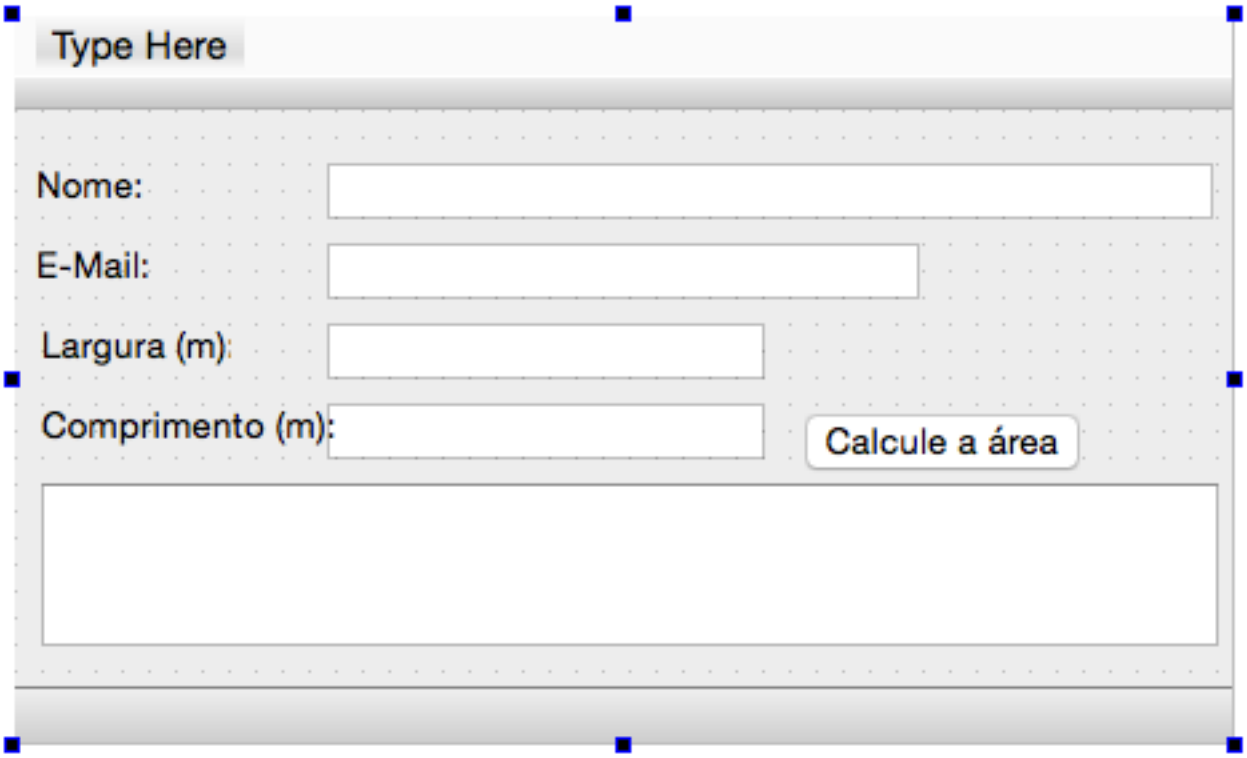
Assim vamos incrementar nossa janela com os seguintes componentes:

- 4 TextLabels
- 4 Line Edit
- 1 PushButton
- 1 Text Browser

Segue o layout:



Agora vamos personalizar os textos da janela para que fique intuitivo:



Agora vamos definir e vincular as responsabilidades de cada elemento da janela.

Primeiramente vamos definir dois objetos, sendo uma Pessoa e um Lote. Isto deve ser feito no arquivo que define sua janela principal (mainwindow.h). Definindo os seguintes atributos:
Pessoa *p;
Lote *l;

Em seguida instancie os respectivos objetos no construtor da classe MainWindow:
this->p = new Pessoa();
this->l = new Lote();

Agora vamos definir os eventos sobre os quais responderemos. Para isso você deve dar um click com o botão direito sobre o componente da janela e em seguida selecionar go to slot…
Assim aparecerá uma janela para que você escolha qual evento você irá responder.
Logo abaixo definimos quais são estes slots e qual será nossa reação diante dos respectivos eventos:

```
void MainWindow::on_lineEdit_returnPressed()
{
    this->p->setNome(ui->lineEdit->text());
    ui->lineEdit_2->setFocus();
    ui->lineEdit_2->selectAll();

}

void MainWindow::on_lineEdit_2_returnPressed()
{
```

```
    this->p->setEMail(ui->lineEdit_2->text());
    ui->lineEdit_3->setFocus();
    ui->lineEdit_3->selectAll();
}

void MainWindow::on_lineEdit_3_returnPressed()
{
    this->l->setFrente(ui->lineEdit_3->text().toFloat());
    ui->lineEdit_4->setFocus();
    ui->lineEdit_4->selectAll();
}
void MainWindow::on_lineEdit_4_returnPressed()
{
    this->l->setComprimento(ui->lineEdit_4->text().toFloat());
    ui->pushButton->setFocus();
    this->on_pushButton_pressed();

}

void MainWindow::on_pushButton_pressed()
{
    QString resposta;
    resposta = "Proprietário: "+p->getNome()+"\n"+p->getEMail()+"\nLote: \n - Frente:
"+QString::number(l->getFrente())+
            "\n - Comprimento: "+QString::number(l->getComprimento())+"\nAREA:
"+QString::number(l->getArea());
    ui->textBrowser->clear();
    ui->textBrowser->setText(resposta);
    ui->lineEdit->setFocus();
    ui->lineEdit->selectAll();
}

void MainWindow::on_pushButton_clicked()
{
    this->on_pushButton_pressed();
}
```