Unidade 2

Interface gráfica
GUI

GUI

- **G**raphical **u**ser **i**nterface

  (Interface gráfica do usuário)
- Tipos de interface:
  - Interface gráfica
  - Linha de comando

GUI

- Permite alcançar o objetivo por meio de manipulando de objetos gráficos, chamados widgets, o que inclui:
  - Janelas
  - Botões
  - Menus
  - Ícones
  - Barras de rolagem e etc..

GUI: componentes

- **Window** – janela
  - uma área da tela que é controlada pelo programa
  - normalmente é retangular, mas podem existir outras formas
  - podem conter outras janelas

GUI: componentes

- **Control** – elemento de controle
  - usado para controlar o programa
  - geralmente geram **eventos**
  - eventos estão ligados com objetos de tal forma que quando um evento ocorre o método correspondente do objeto será chamado.
  - GUI costuma proporcionar o mecanismo para conectar eventos com métodos

GUI: componentes

- **Widget**
  - Elementos de controle que podem ser visíveis ou não ao usuário
  - Podem ser manipulados pelo usuário:
    - **Frame**
    - **Label**
    - **Button**
    - **Message boxes**
    - **Text box**
    - **Check button**
    - **Radio Button**
    - **Listbox**
    - **Menu/MenuButton**
    - **Scale/Scrollbar**

GUI: componentes

- **Frame**
  - é um **widget**
  - normalmente é usado para agrupar outros widgets
  - frequentemente representa a “janela” propriamente dita, e os outros frames podem ser embutidos nele

GUI: componentes

- **Layout (esboço, arranjo, esquema, design)**
  - Os componentes estão posicionados dentro de um **Frame** de acordo com um **Layout** especifico.
  - Layout pode ser definido de varias formas:
    - usando coordenadas em pixeis
    - determinando a posição relativa com outros componentes (esquerda, topo, etc)
    - usando um padrão: tabela, “grade”
  - Sistema de coordenadas em pixeis é fácil de utilizar, mas pode gerar dificuldades para janelas que mudam de tamanho.

GUI: componentes

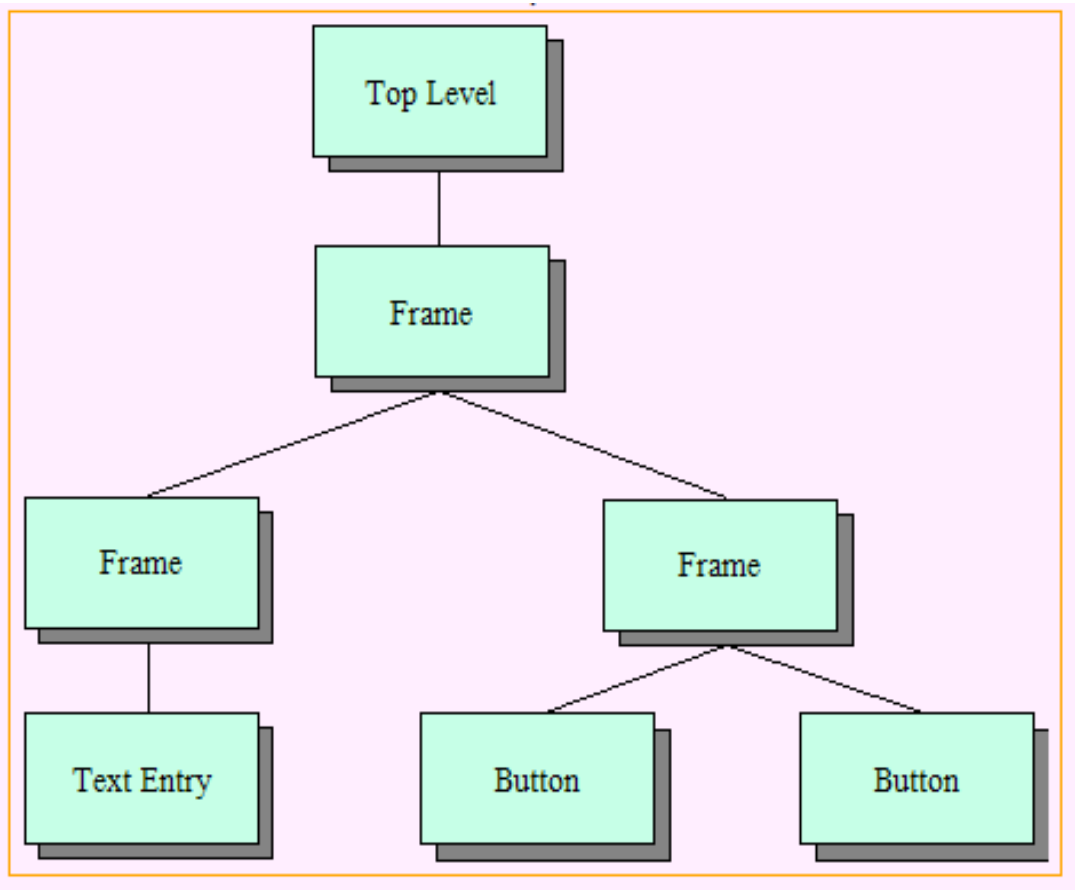
- **Child**
  - Aplicativos tendem a conter uma hierarquia de widgets/componentes de controle.
  - O Frame principal geralmente vai ser composto por **Frames** auxiliares, que por sua vez também podem conter alguns outros **Frames**.
  - Essa hierarquia pode ser representada como uma arvore, onde cada componente tem um único “**pai**” e vários “**filhos**”.
  - Normalmente essa estrutura pode ser visualizada em processo de desenvolvimento, e algumas ações podem ser aplicados tanto para **componente-pai** como para todos seus “**filhos**”.

GUI: componentes

- **The Containment tree** (a arvore de hierarquia)
  - É um conceito importante para desenvolvimento de GUI
  - Os **widgets** pertencem a uma arvore de hierarquia, onde o componente-pai (a raiz) fica responsável pela interface
  - Vários componentes-filhos pode ter seu próprios filhos
  - Os componentes-filhos recebem **Eventos** e caso eles não consigam processar esse tipo de evento eles passam essa tarefa para seu componete-pai, e assim por diante subindo até o topo da hierarquia.
  - Se o **Evento** for recebido no nível alto, dependendo do comando, é bem provável que esse comando vai ser repassado para componentes-filhos. O mesmo comando recebido pelo componente-filho pode ser processado por ele mesmo.
    (Exemplo: comando redraw)

GUI: componentes

- O conceito de **Eventos** e **Comandos** é fundamental para entender o funcionamento de GUI de ponto de vista de programação:
  - **Eventos** subindo pela arvore
  - **Comandos** descendo pela arvore
- É preciso sempre especificar quem é o componete-pai de um **widget** para encaixar ele em uma hierarquia



QT

- **Qt** é um framework de desenvolvimento de aplicativos multi-plataforma (cross-platform) tanto para PC como para mobile.
- Plataformas:
  - Linux
  - OS X
  - Windows
  - Android
  - iOS, BlackBerry, Sailfish OS e etc.

http://www.qt.io/developers/

QT

- QT não é uma linguagem de programação
- QT é um framework escrito em C++
- Adiciona algumas características para linguagem C++ (signals e slots)
- Na fase de compilação gera arquivos de C++ padrão

QT

- Compatível com vários compiladores C++ como Clang, GCC, ICC, MinGW and MSVC.
- O projeto começou a ser desenvolvido em 1990
- Hoje em dia se tornou um grande projeto com a participação de vários membros
- Distribuição (licença):
  - Comercial
  - **GPL** - General Public License (Licença Pública Geral)
  - **LGPL** - Lesser General Public License

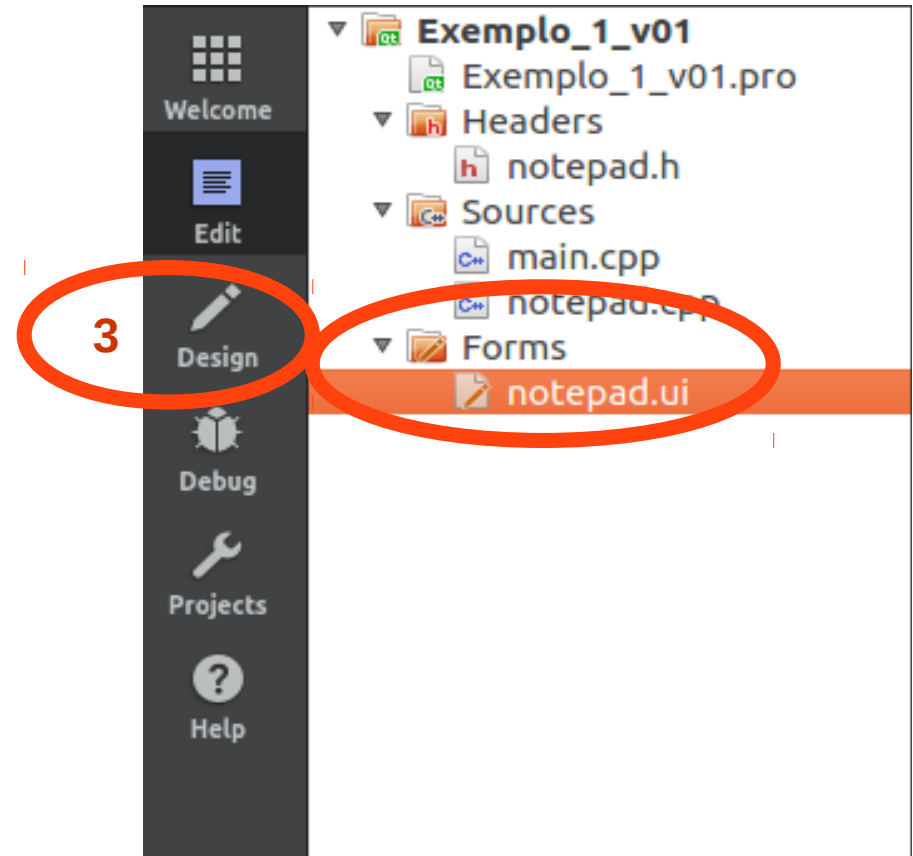
QT

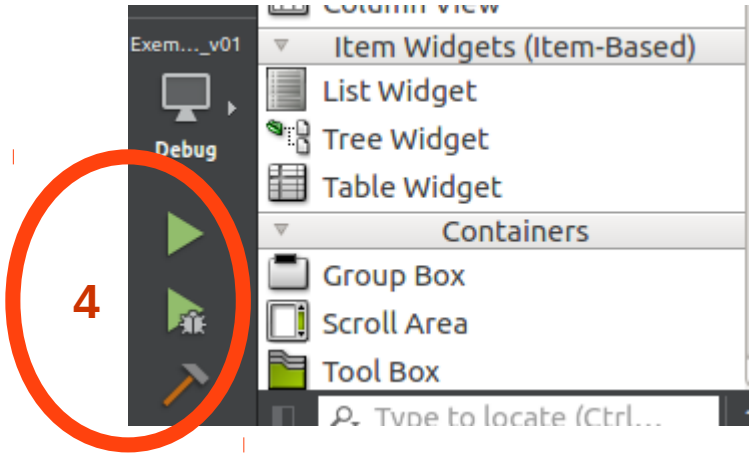
- **QT Creator** – IDE (Integrated Development Environment)
- Permite trabalhar com vários módulos:
  - Redes
  - Bancos de dados
  - OpenGL
  - Web
  - Sensores
  - Protocolos de comunicação (Bluetoth, NFC)
  - XML e JSON
  - Geração de PDF etc.

Exemplo 1: Notepad (editor de texto)

http://doc.qt.io/qt-5/gettingstartedqt.html

1. New Project → Qt Widgets Application
2. Class Name: Notepad
3. Troca entre modo edição de código e desenvolvimento visual
   (**Edit mode** e ***.UI**)
4. Compilação e execução

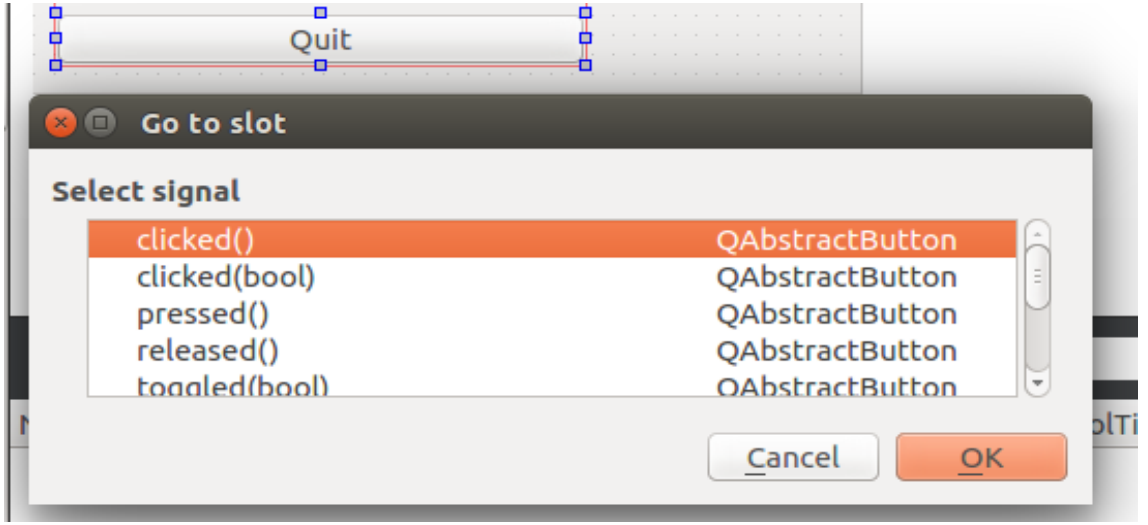




signals and slots

- Um **sinal** é emitido quando um evento especifico acontece
- **Slot** é a função que é chamada em resposta para um sinal especifico
- Existe um conjunto de **sianis** e **slots** predefinidos para **widgets** em QT
- Os sinais e slots predefinidos podem ser usados diretamente do QT Designer.
- Para usar o Qt Designer para boton “Quit” temos que chamar o menu com o clique direito do mouse e no menu escolher a opção

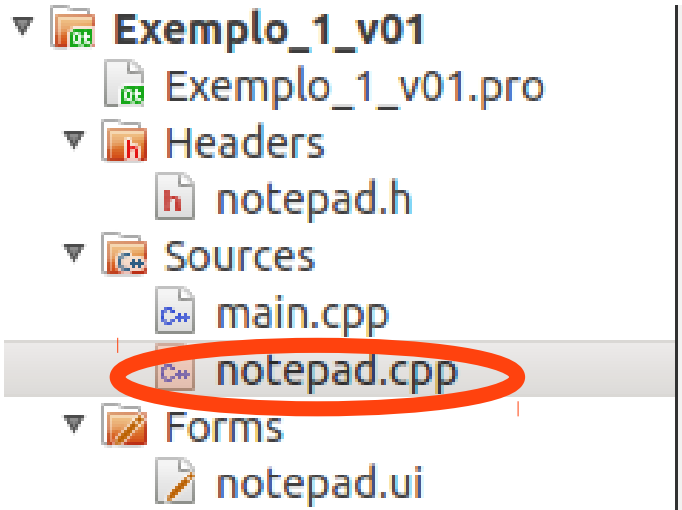
Go to slot -> clicked()



signals and slots

Para boton **Quit** funcionar precisamos adicionar:

qApp->quit();




```
1  #include "notepad.h"
2  #include "ui_notepad.h"
3
4  Notepad::Notepad(QWidget *parent) :
5      QMainWindow(parent),
6      ui(new Ui::Notepad)
7  {
8      ui->setupUi(this);
9  }
10
11 Notepad::~Notepad()
12 {
13     delete ui;
14 }
15
16 void Notepad::on_quitButton_clicked()
17 {
18     qApp->quit();
19 }
20
```

Exemplo 2: Calculadora GUI

- Desenvolvimento de aplicativo com interface gráfica (GUI)
- Aplicativo Calculadora
- Operações aritméticas básicas
  - Soma
  - Subtração
  - Multiplicação
  - Divisão

Exemplo 3: TextFinder

- Aplicativo simples que permite a busca pelo texto definido pelo usuário.

Exercicio: Aplicativo simples

- Criar um aplicativo simples com vários widgets e estudar seus atributos