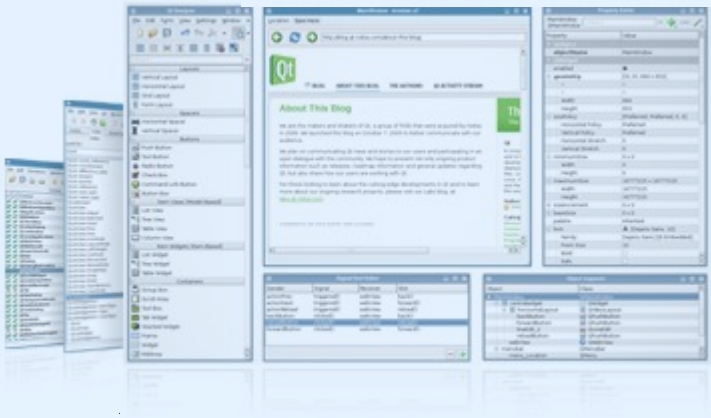
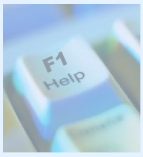
Revista Qt

revistaqt@gmail.com

Edição 1 – setembro/outubro - 2010





Dicas

Tutoriais

QT + PHP

Aplicações híbridas: Desktop/Web

A primeira revista brasileira para programadores e estudantes de Qt

Edição 1 – Setembro/Outubro – 2010

Editorial

Olá, caríssimo(a) leitor(a).

Esta é a edição inaugural da primeira revista brasileira sobre esta fabulosa ferramenta de programação chamada **Qt**. A motivação para criar a **Revista Qt** vem do meu entusiasmo com o framework e da dificuldade em se obter informação em língua portuguesa disponível sobre o assunto. A cada dois meses (começaremos como uma publicação bimestral), a **Revista Qt** trará tutoriais, macetes, novidades, da forma mais simples e bem humorada que for possível.

Agradecimentos especiais a Oliver Widder, do Geek and Poke, por ter permitido a publicação de suas charges na **Revista Qt**.

O sucesso deste projeto só será possível com a sua participação, criticando, sugerindo, perguntando. O email da revista é revistaqt@gmail.com. Nosso site é o revistaqt.blogspot.com e o Twitter é **http://twitter.com/revistaqt**.

Um grande abraço e boa leitura

André Luiz de Oliveira Vasconcelos

Editor

Exemplo 1: Documentação Desatualizada de Sistema

http://geekandpoke.typepad.com/

3. Apresentando o Qt
O que é, quem criou e para que serve o Qt

4. Instalação do Qt SDK
A instalação do ambiente de desenvolvimento em Qt nas plataformas Linux e Windows®

10. Alô, Qt Creator!
Primeiros passos na utilização da IDE Qt Creator

20. Qt + PHP – parte 1
Aplicativos híbridos: Desktop/Web

24. Espaço do Leitor
Apresentação do espaço para colaboração do leitor

Editor: André Luiz de O. Vasconcelos
Charges: Oliver Widder

Apresentando o Qt

Qt é um *framework* multiplataforma para desenvolvimento de aplicações com interfaces gráficas de usuário (GUI). Por ser multiplataforma, as aplicações escritas utilizando o Qt podem ser compiladas em diferentes plataformas, como Linux, Windows® e Mac OS®.

O Qt começou a ser desenvolvido em 1991 na Noruega por Haavard Nord e Eirik Chamb-Eng, em uma empresa chamada Quasar Technologies. Três anos mais tarde, eles fundariam a Troll Tech que passaria depois a chamar-se Trolltech.

Em 2008, a empresa finlandesa do ramo de celulares e outros dispositivos móveis, Nokia, comprou a Trolltech como parte de uma estratégia para desenvolvimento de aplicações independentes de plataformas e para expandir seus serviços de Internet.

Code less.
Create more.
Deploy everywhere.

Entre as aplicações mais famosas desenvolvidas em Qt, podemos citar KDE, Skype, VLC Media Player, Google Earth, e Virtual Box. Nomes como Google, Cannon,AMD, GE, Pfizer, Wolksvagen, Samsung, Hitachi, Siemens estão entre os usuários mais ilustres do Qt

Atualmente, o Qt está disponível sob licença comercial - paga ou LGPL - grátis.

A licença comercial tem custo, mas permite o desenvolvimento de aplicações proprietárias com restrições de licença.

A licença LGPL não tem custo, mas qualquer mudança feita no código do Qt deverá ser compartilhada e as aplicações também devem ser distribuídas sob a licença LGPL. Ou seja, desde que disponibilize sua aplicação sob a licença LGPL, você pode usar a versão gratuita do Qt.

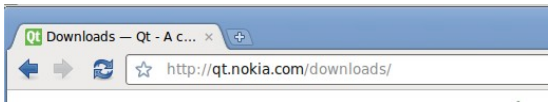
O Qt possui uma rica biblioteca de classes, facilitando o criação de aplicações para as mais diversas finalidades, como comunicação, renderização de imagens, multimídia, jogos, etc.

O *slogan* na logomarca apresentada nesta página, sintetiza a motivação do Qt: *Codifique menos, crie mais, distribua por toda a parte*.

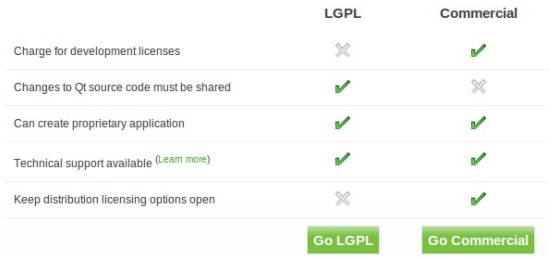
Instalação do Qt

O primeiro passo para a instalação do Qt, é fazer o download do mesmo, de acordo com a plataforma e a licença desejadas. Veremos aqui a instalação da versão LGPL. Acesse a página de Downloads da Nokia pelo endereço:

http://qt.nokia.com/downloads



Na página de Downloads, clique no botão Go LGPL para selecionar a opção gratuita do Qt.



Na próxima tela, teremos as opções disponíveis para download sob a licença LGPL. Pode-se fazer download apenas das bibliotecas, apenas das ferramentas de desenvolvimento ou de todo o ambiente de desenvolvimento do Qt.

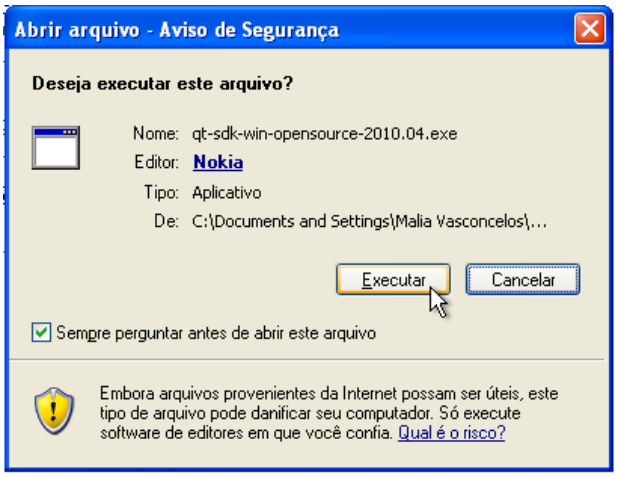
Neste tutorial de instalação veremos apenas a instalação do ambiente de desenvolvimento (Qt SDK completo) no Windows® e no Linux.

Instalação no Windows®

Clique no link “Qt SDK for Windows* (293 MB)” para fazer o download do arquivo de instalação.

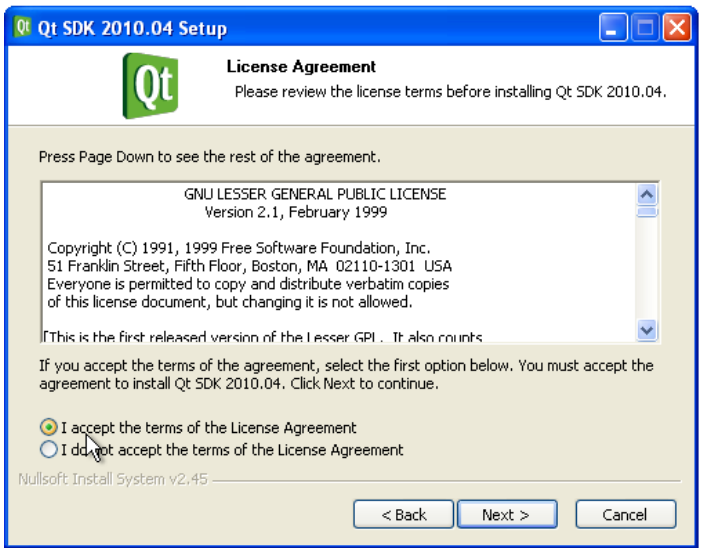
Dependendo da velocidade de sua conexão com a Internet, este pode ser um bom momento para um cafezinho, pois estamos falando de um download de 293MB.

Após concluído o café, ou o download, execute o arquivo para iniciar a instalação.

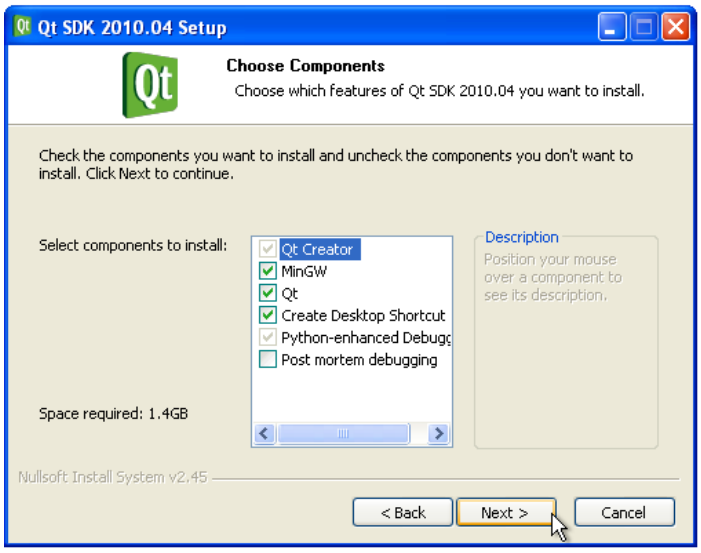


Após a descompactação de dados, teremos a tela de “boas vindas” do instalador. A tela pode diferir de acordo com a versão do Qt SDK disponível no momento em que você fez o download, neste caso: 2010.04.

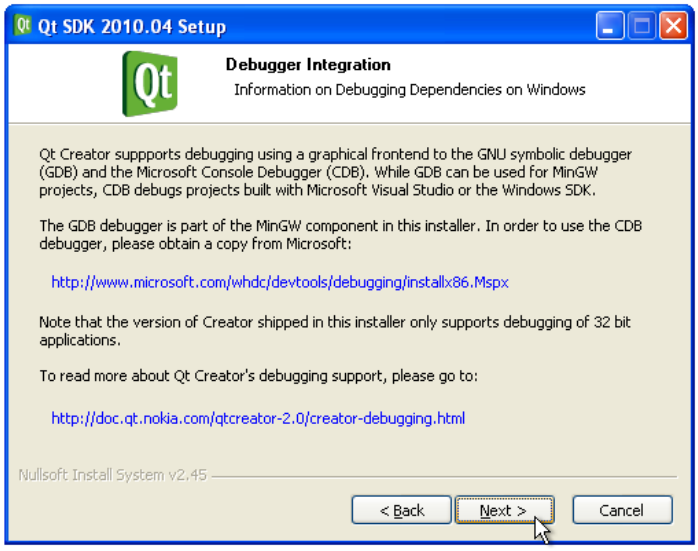
A próxima tela do instalador apresenta a licença GNU LGPL e aguarda que o usuário marque a opção "**I accept the terms of the License Agreement**", indicando que concorda com os termos da Licença. Clique no botão **Next** para continuar a instalação.



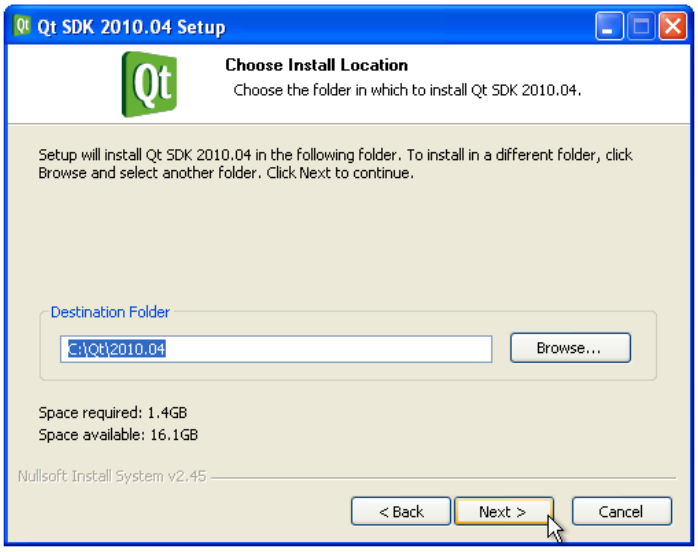
A tela seguinte do instalador nos apresenta os componentes do Qt SDK a serem instalados. Para prosseguir apenas clique no botão **Next**.



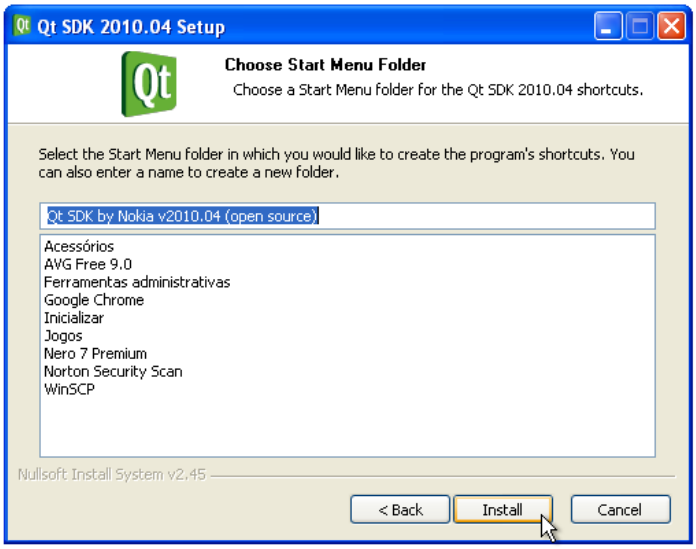
Neste ponto, o instalador apresenta informações sobre as dependências para instalação do GNU Debugger no Windows®. Para o escopo deste tutorial, vamos ignorar estas dependências, clicando no botão **Next**.



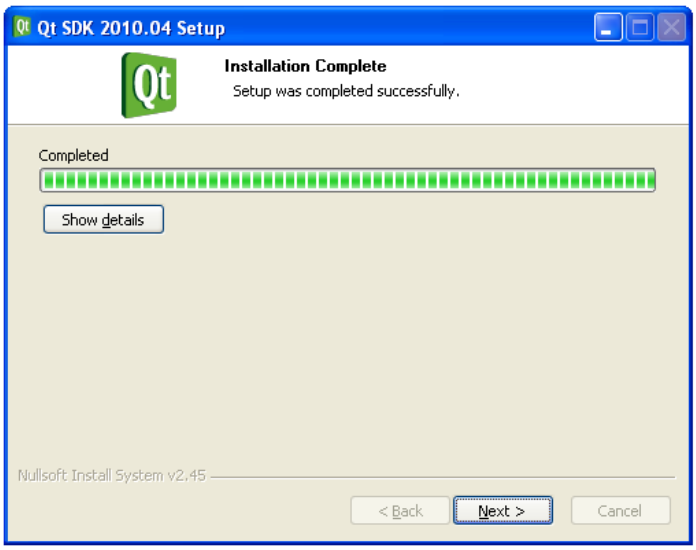
O próximo passo é informar ao instalador o diretório onde será instalado o Qt SDK. O diretório sugerido pelo instalador é C:\Qt\2010.04, onde 2010.04 corresponde à versão do Qt SDK. Clique no botão **Next** para continuar.



Na tela seguinte, o usuário pode informar um nome para a pasta correspondente ao Qt SDK no menu principal do Windows®. Para continuar apenas clique no botão **Install**.



Ao final da instalação, o botão **Next** será habilitado. Clique neste botão para continuar.



Pronto! Agora o Qt SDK está instalado. Como o escopo deste tutorial é apenas a instalação do Qt SDK, desmarque a opção **Run Qt Creator**, que vem marcada por *default*, para que o Qt **Creator** não seja executado após a finalização do instalador. Em outra matéria desta edição, veremos a criação de um "Alô, mundo!" usando o **Qt Creator**.

Instalação no Linux utilizando o binário de instalação

Na tela apresentada na página 4, clique no link "Qt SDK for Linux/X11 32-bit** (371 MB)", caso tenha uma máquina com arquitetura de 32 bits ou no link "Qt SDK for Linux/X11 64-bit** (469 MB)", caso tenha uma máquina com arquitetura de 64 bits.

Após a conclusão do download, abra um terminal, vá até a pasta onde salvou o arquivo e dê permissão de execução, executando o comando **chmod u+x**, como segue:

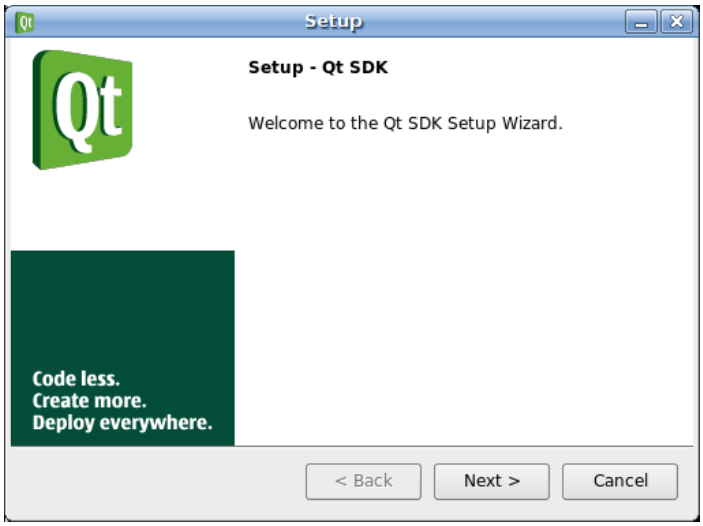
```
chmod u+x qt-sdk-linux-x86-opensource-2010.04.bin
```

Depois de ter dado permissão de execução para o arquivo, basta executá-lo para iniciar a instalação:

```
./qt-sdk-linux-x86-opensource-2010.04.bin
```

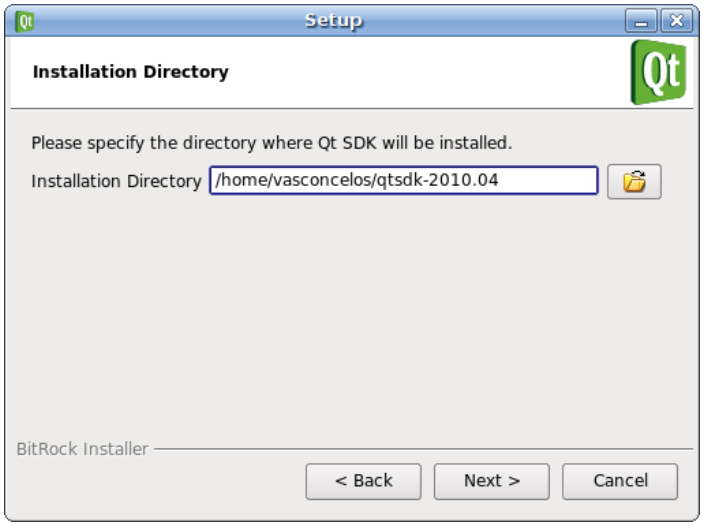
Tem dúvidas, críticas, sugestões?
Envie um e-mail para revistaqt@gmail.com.

A tela de “boas vindas” do programa de instalação será exibida.

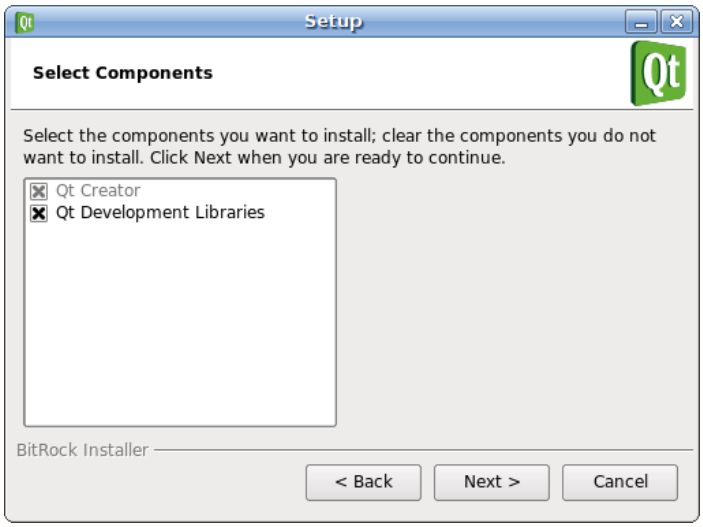


A próxima tela do instalador apresenta a licença GNU LGPL e aguarda que o usuário marque a opção “I accept the terms of the License Agreement”, indicando que concorda com os termos da Licença. Clique no botão **Next** para continuar a instalação.

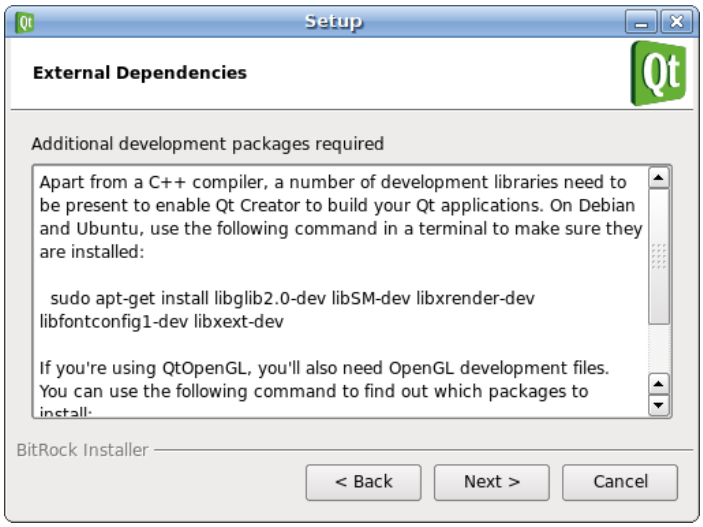
O próximo passo é informar ao instalador o diretório onde será instalado o Qt SDK. O diretório sugerido pelo instalador é /home/usuario/qtsdk-2010.04, onde **usuario** é o nome do usuário que executou o instalador e 2010.04 corresponde à versão do Qt SDK. Clique no botão **Next** para continuar.



A tela seguinte do instalador nos apresenta os componentes do Qt SDK a serem instalados. Você pode prosseguir com a instalação clicando no botão **Next**.



Prosseguindo, o instalador apresenta uma tela com as dependências . Como estamos instalando o Qt SDK para desenvolvimento em C++, precisamos do compilador desta linguagem, bem como de algumas bilbiotecas de desenvolvimento instaladas para que o Qt Creator possa “construir” as aplicações.



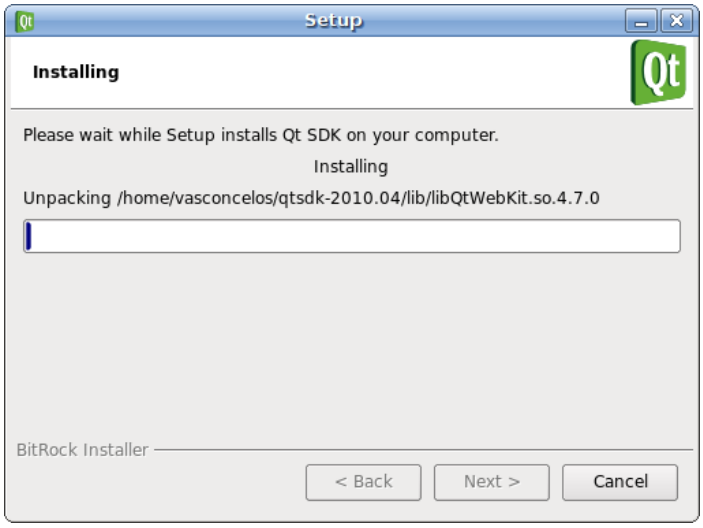
Nesta tela, o instalador inclusive mostra os comandos a serem utilizados para instalação das bibliotecas em sistemas Debian ou Ubuntu:

```
sudo apt-get install libglib2.0-dev libSM-dev libxrender-dev libfontconfig1-dev libxext-dev
```

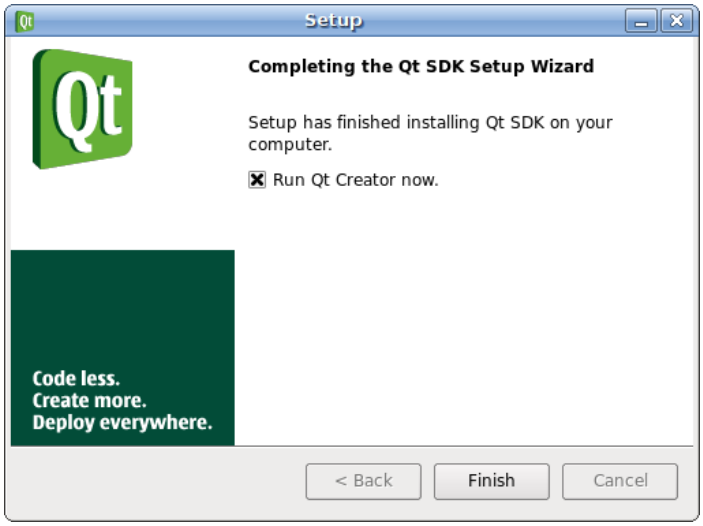
Você pode clicar no botão **Next** agora para continuar e instalar as bibliotecas e aplicativos adicionais necessários depois.

Neste ponto, o instalador está pronto para iniciar o processo de criação de diretórios e cópia dos arquivos do Qt SDK. Clique no botão **Next**.

O instalador procede neste ponto a criação dos diretórios e cópia dos arquivos do Qt SDK.



Pronto! Agora o Qt SDK está instalado. Como o escopo deste tutorial é apenas a instalação do Qt SDK, desmarque a opção **Run Qt Creator**, que vem marcada por *default*, para que o Qt **Creator** não seja executado após a finalização do instalador. Em outra matéria desta edição, veremos a criação de um “Alô, mundo!” usando o **Qt Creator**.



Se tudo correu bem no acompanhamento deste tutorial, agora você está com o Qt SDK instalado em sua máquina. Usuários de distribuições derivadas do Debian ou Ubuntu podem utilizar o aplicativo Synaptic para instalar os pacotes do Qt4 ao invés de utilizar o arquivo binário disponível no endereço http://qt.nokia.com/downloads. Neste caso recomendo a instalação dos seguintes pacotes:

libqt4-assistant
libqt4-dbg
libqt4-designer
libqt4-gui
libqt4-multimedia
libqt4-opengl
libqt4-phonon
libqt4-script
libqt4-sql
libqt4-sql-odbc
libqt4-sql-sqlite
libqt4-svg
libqt4-webkit
libqt4-xmlpatterns
qt4-designer
qt4-doc
qtcreator
libqt4-core
libqt4-dbus
libqt4-dev
libqt4-help
libqt4-network
libqt4-opengl-dev
libqt4-qt3support
libqt4-scripttools
libqt4-sql-mysql
libqt4-sql-psql
libqt4-sql-sqlite2
libqt4-test
libqt4-xml
qt4-demos
qt4-dev-tools
qt4-qmake
qtcreator-doc

Baixar o arquivo diretamente no site da Nokia, garante que estejamos usando as versões mais atuais disponíveis, enquanto a instalação dos pacotes é feita com as versões disponíveis nos repositórios do Debian ou Ubuntu.
Se você vai trabalhar com Linux, existem ainda outros pacotes que devem ser instalados, relativos ao desenvolvimento em C/C++, a saber:

build-essential
xlibs-static-dev libxclass-dev
libxext-dev

por André Vasconcelos

Se ao tentar executar um programa compilado em Qt, você obtem a seguinte mensagem:

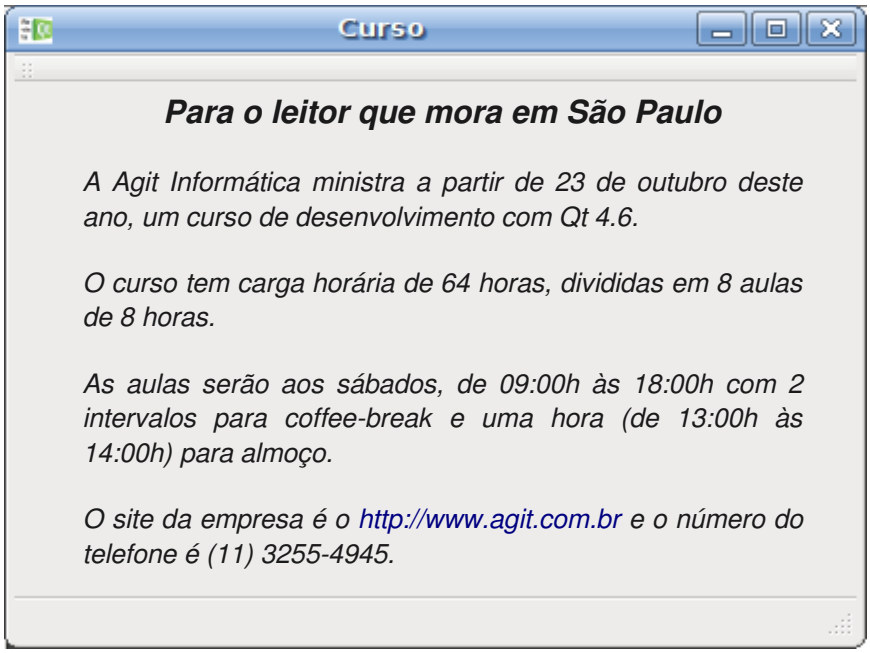
symbol lookup error: /usr/local/lib/libQtGui.so.4:
undefined symbol: _ZNK17QVariantAnimation10metaObjectEv

Inclua a linha a seguir no arquivo /etc/profile

export LD_LIBRARY_PATH=/home/caminho_para_diretorio_qt/qt/lib

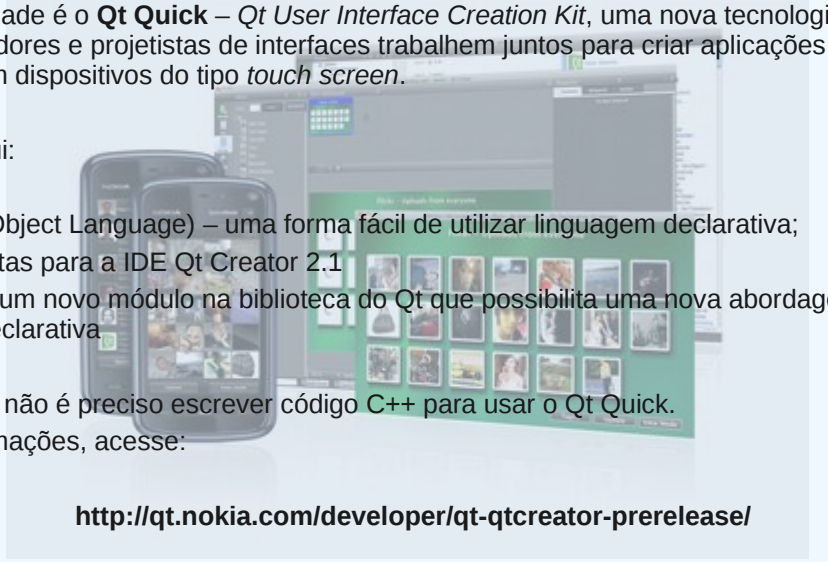
No meu caso, como instalei o Qt SDK no diretório /home/vasconcelos/qtsdk-2010.04ficou assim:

export LD_LIBRARY_PATH=/home/vasconcelos/qtsdk-2010.04/qt/lib



No dia 26 de agosto de 2010, a Nokia disponibilizou uma versão Release Candidate do Qt 4.7: a próxima versão do Qt. A versão final deve ser liberada no mês de setembro.

A principal novidade é o **Qt Quick** – *Qt User Interface Creation Kit*, uma nova tecnologia que permite que desenvolvedores e projetistas de interfaces trabalhem juntos para criar aplicações animadas e compatíveis com dispositivos do tipo *touch screen*.



O Qt Quick inclui:

QML (Qt Meta-Object Language) – uma forma fácil de utilizar linguagem declarativa;

Novas ferramentas para a IDE Qt Creator 2.1

QtDeclarative – um novo módulo na biblioteca do Qt que possibilita uma nova abordagem de programação declarativa

O detalhe é que não é preciso escrever código C++ para usar o Qt Quick.

Para mais informações, acesse:

http://qt.nokia.com/developer/qt-qtcreator-prerelease/

Alô, Qt Creator!

Qt Creator é uma IDE (*Integrated Development Environment*) multiplataforma para desenvolvedores Qt.

Disponível nas plataformas Windows, Linux/X11 e Mac OS, o Qt Creator permite criar aplicações tanto para desktop como para dispositivos móveis.

O Qt Creator é composto por:
* Editor de código C++ and JavaScript
* Editor visual integrado para desenho de interfaces gráficas de usuário
* Ferramentas de montagem e gerenciamento de projeto
* Depuradores gdb e CDB
* Controle de versão
* Simulador para interfaces de dispositivos móveis
* Suporte para aplicações desktop e móveis

Editor de Código
O editor de código disponível no Qt Creator provê:
- suporte para edição de C++ e QML (Javascript)
- ajuda sensível ao contexto
- *code completion*, auxiliando na digitação de nomes de classes, métodos, enumerações, etc.

Controle de Versão
O Qt Creator se integra com os mais populares sistemas de controle de versões: Git, Subversion, Perforce, CVS e Mercurial.

Editor visual integrado de interfaces gráficas de usuário
O Qt Creator provê 2 editores visuais integrados: Qt Designer para montar interfaces gráficas de usuário a partir de *widgets* do Qt, e Qt Quick Designer* para desenvolvimento de interfaces gráficas animadas com a linguagem QML.

*O Qt Quick Designer está disponível como *Preview* no Qt Creator 2.0.1.

Montagem e gerenciamento de projeto
Quer você importe um projeto existente ou crie um do zero, Qt Creator gera todos os arquivos necessários. Suporte para cross-make e CMake estão incluidos.

Suporte para aplicações *desktop* e móveis
Qt Creator provê suporte para montar e executar aplicações Qt para desktops e dispositivos móveis. Parâmetros permitem-lhe rapidamente alterar entre alvos de montagem.

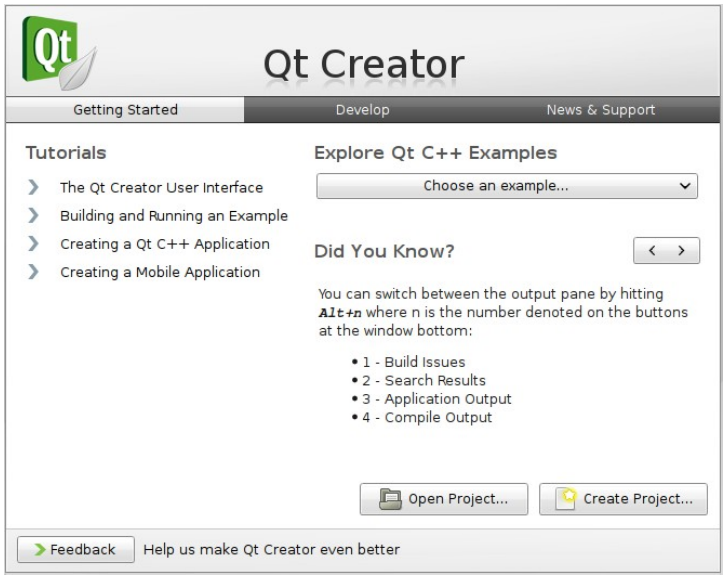
Qt Simulator
Disponível como parte do Nokia Qt SDK, o Qt Simulator permite testar aplicações para dispositivos móveis em ambiente similar àquele do dispositivo alvo.

Tela de “boas vindas” do Qt Creator

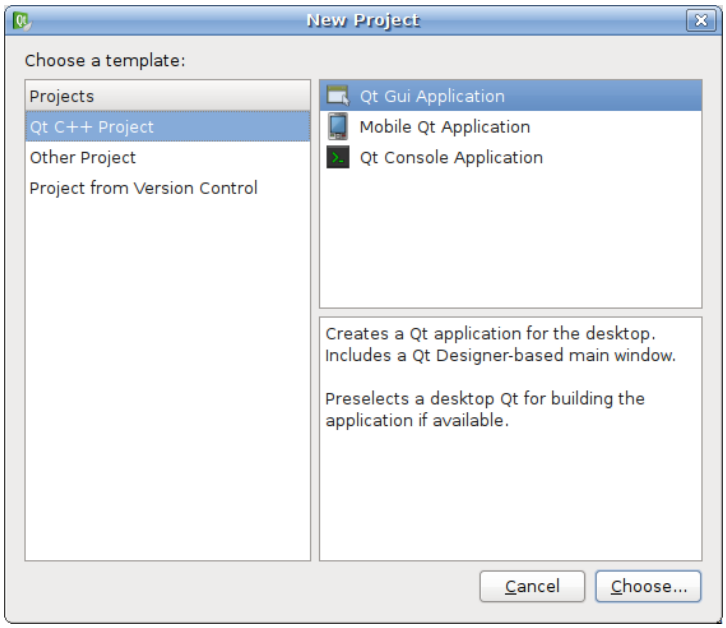
Na tela inicial do Qt Creator - mostrada na próxima figura – existe uma janela de “boas vindas” que permite ao usuário:

Acesso a tutoriais sobre a criação de aplicações usando o Qt Creator
Selecionar um exemplo de aplicação Qt para servir como base para um projeto nosso
Visualizar dicas sobre a utilização do Qt Creator
Abrir um projeto existente
Criar um projeto novo
Enviar à equipe do Qt Creator informações sobre a utilização da ferramenta que possam auxiliar em sua melhoria

Neste tutorial veremos como criar a nossa primeira aplicação em Qt utilizando o Qt Creator. Para isso, clique na opção “*Create Project...*” na tela de “boas vindas”.

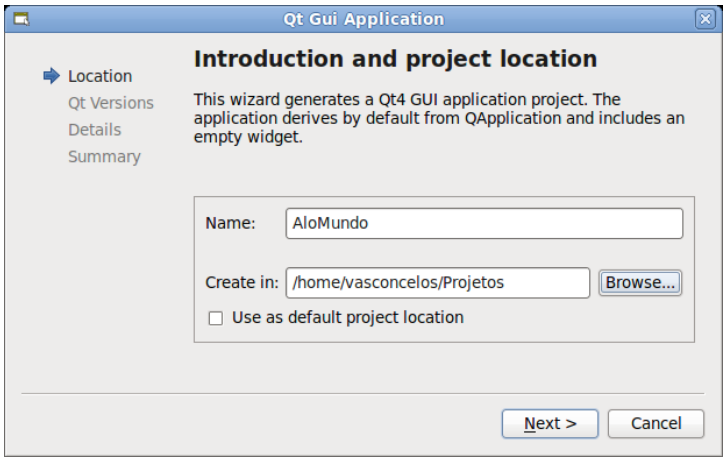


O primeiro passo na criação do projeto é selecionar o tipo de aplicação. Para este nosso “Alô, Mundo!”, vamos usar as opões *default*, que são Qt C++ Project → Qt Gui Application.
Para prosseguir, clique no botão “*Choose...*”.

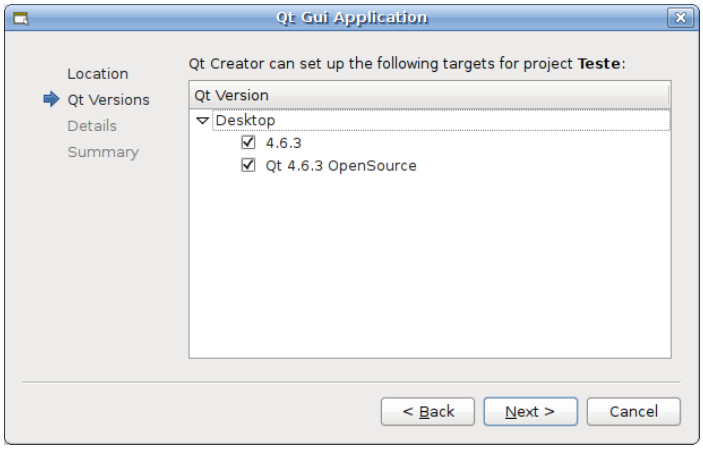


A tela seguinte serve para informar o nome do projeto e sua localização (diretório).
Preencha o campo *Name* com **AloMundo**, que será o nome do nosso projeto.
Preencha o campo *Create in* com o diretório no qual será criado a diretório do seu projeto. No meu caso, tenho um diretório chamado **Projetos** em /home/vasconcelos. Assim, o projeto será criado em /home/vasconcelos/Projetos/AloMundo.
Você pode usar o botão “*Browse*” para selecionar um diretório (ou pasta) em seu sistema de arquivos.
Se desejar que esta localização seja utilizada como base para todos os seus projetos, clique na opção “*Use as default project location*”.

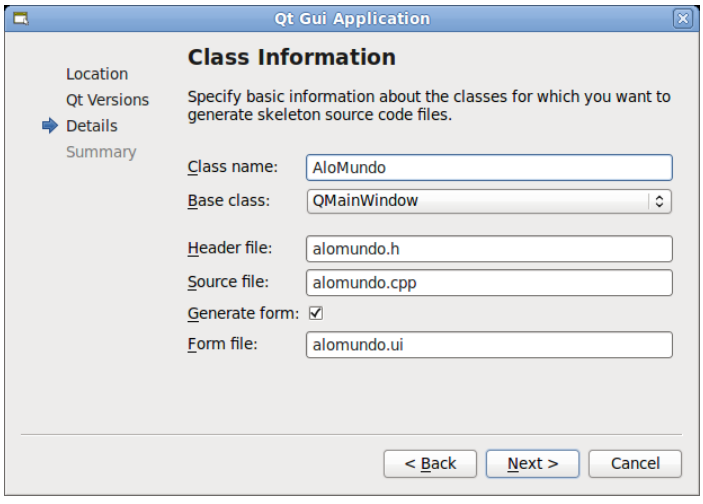
Para prosseguir, clique no botão “*Next...*”.

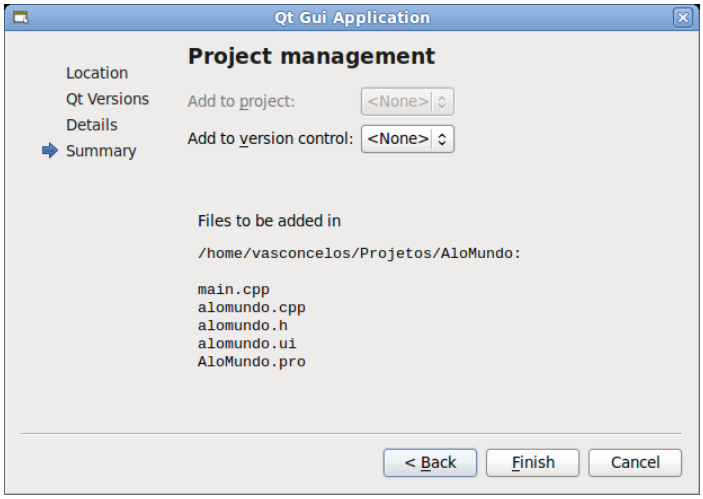


A próxima tela apresenta os produtos Qt instalados em seu sistema. Para prosseguir, clique no botão “*Next...*”.



Em seguida, você deve informar alguns dados relacionados à classe principal da sua aplicação.
Preencha o campo *Class name* com o texto “**AloMundo**”. Isto será o nome da classe principal da nossa aplicação.
O campo *Base class* indica qual a classe pai da classe principal da nossa aplicação. Deixe o valor sugerido pelo Qt Creator (QmainWindow).
Os nomes dos arquivos de cabeçalho, fonte e de formulário são sugeridos a partir do nome da classe principal.
Observe nesta tela, a opção “*Generate form*” que determina que o Qt Creator deve criar um formulário GUI para a aplicação.
Para prosseguir, clique no botão “*Next...*”.





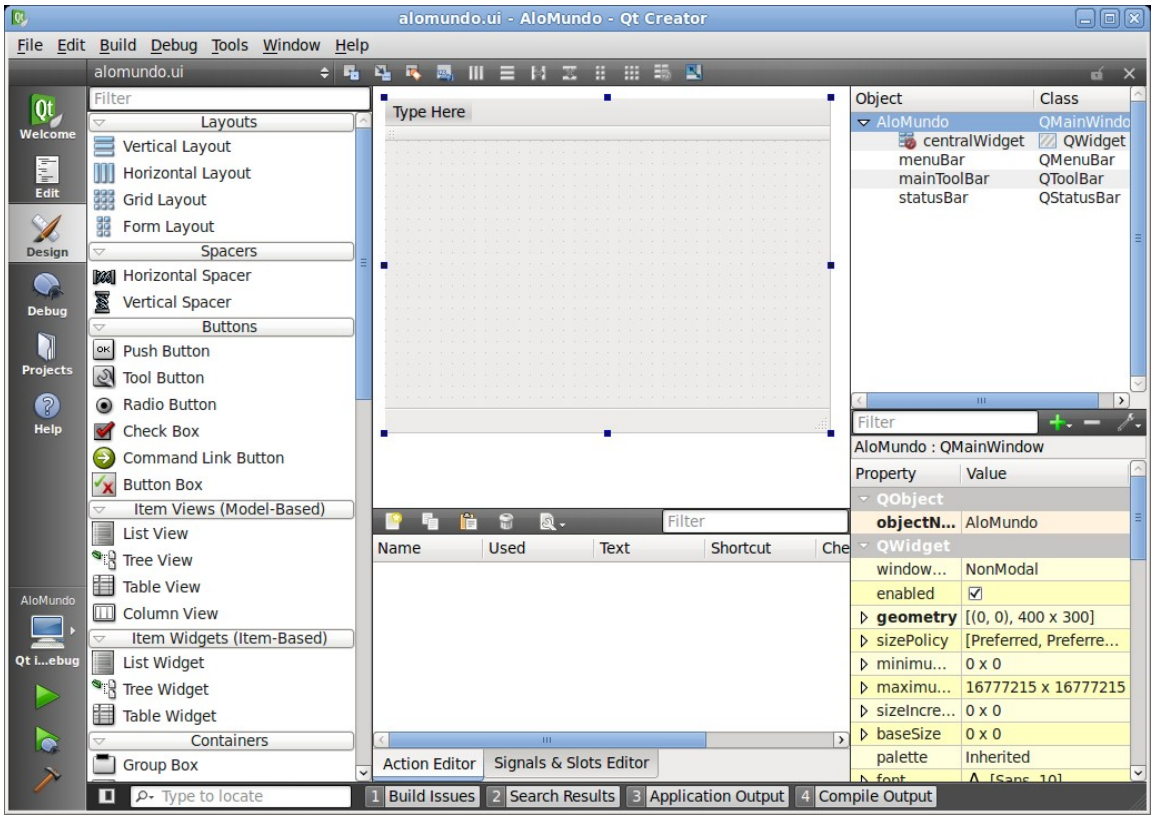
A última tela do *wizard* de geração do projeto apresenta os nomes dos arquivos a serem gerados. Neste ponto é possível ainda selecionar um sistema de controle de versão.
Para prosseguir, clique no botão "*Finish...*".

Pronto! Você acaba de criar o seu primeiro projeto usando o Qt Creator.

O Qt Creator criou o diretório AloMundo e dentro dele os seguintes arquivos:

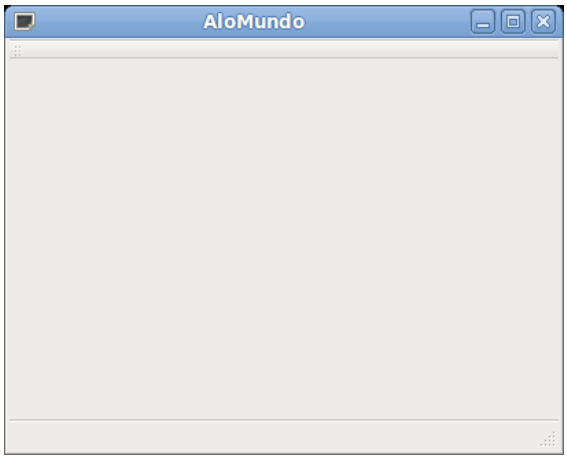
main.cpp – contém a função main da aplicação
alomundo.cpp – contém o código da classe principal da aplicação, que no caso é a classe AloMundo.
alomundo.h -refere-se cabeçalho da classe AloMundo.
alomundo.ui – XML com definição do formulário da aplicação.
AloMundo.pro – este é o arquivo de definições do projeto.

Em matérias futuras da Revista Qt veremos mais detalhadamente os elementos que compõem um projeto Qt, por ora basta saber que o Qt Creator faz o trabalho de criar tais elementos para você.



Apesar de não fazer muita coisa, o projeto que você acabou de criar já pode ser executado pelo Qt Creator. Para isso você pode teclar a combinação CTRL – R , selecionar no menu a opção Build → Run ou clicar no botão Run mostrado a seguir.

Como não colocamos nada em nosso projeto, o resultado de sua execução será apenas uma janela sem conteúdo tendo como título o nome do projeto.



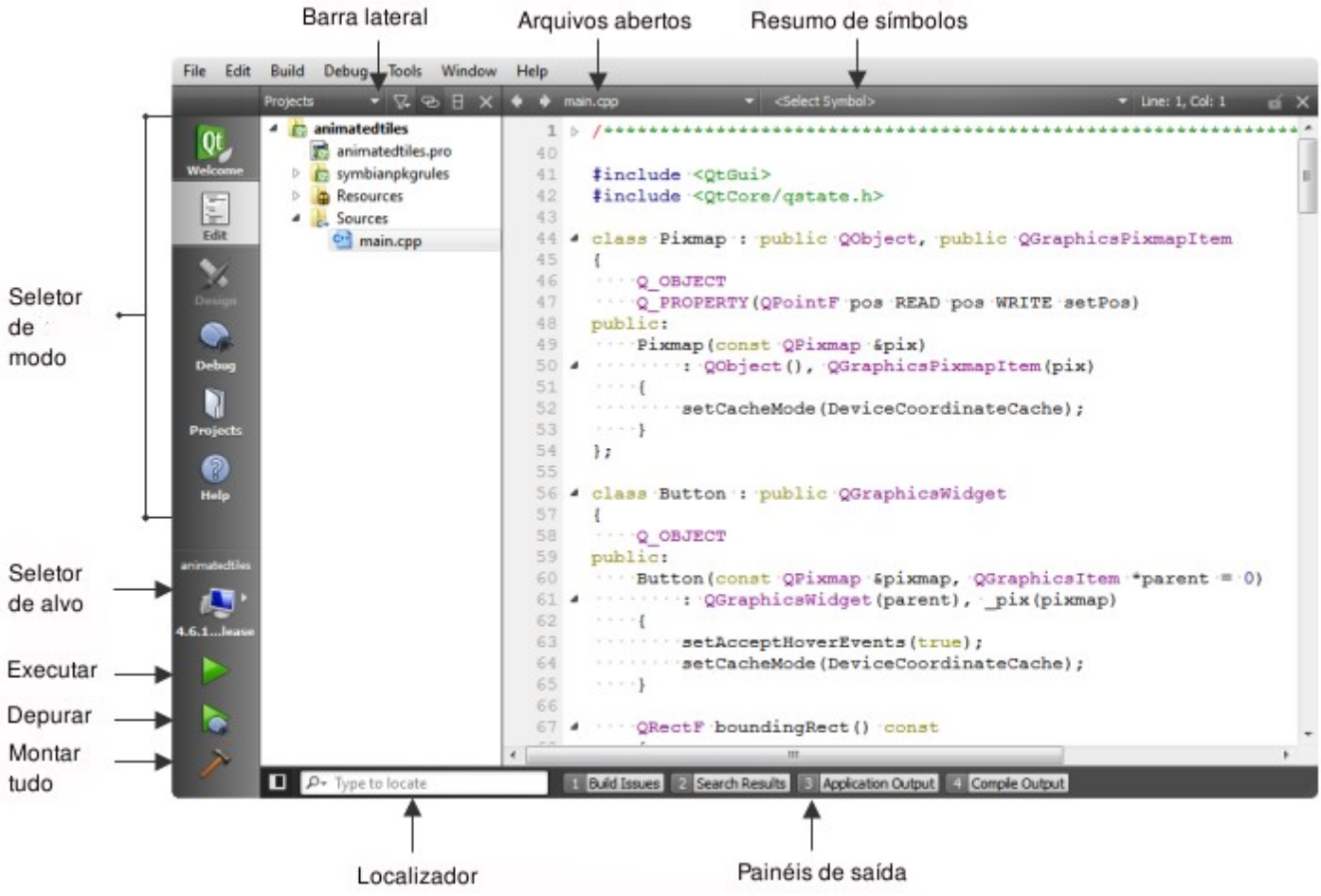
Mesmo não parecendo grande coisa, note que a janela de nossa pequena aplicação já apresenta as funcionalidades básicas de uma aplicação gráfica comum. Pode ser minimizada, maximizada, movida, redimensionada e fechada.

Nota ainda que no topo da parte interna da janela existem uma barra de ferramentas vazia, que pode ser movida para outras posições da janela.

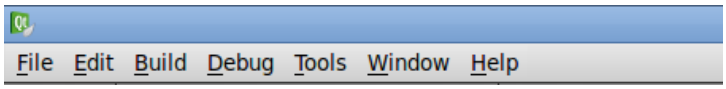
Estas funcionalidades estão na classe que utilizamos como base para a nossa classe principal: QMainWindow. Desta forma não precisamos nos preocupar com aspectos básicos de nossas aplicações.

Clique no botão fechar para encerrar a execução do AloMundo.

A seguir vamos conhecer os elementos principais do Qt Creator, antes de fazer qualquer alteração em nosso projeto. Mas atenção: este é apenas o primeiro contato com o Qt Creator e alguns elementos ficarão mais fáceis de entender quando você os estiver utilizando na prática.



Vamos começar pelo Menu Principal do Qt Creator, que fica na parte superior da tela.



Muitas das opções acessíveis pelo Menu, estão disponíveis em forma de atalhos do teclado, menus especiais de contexto, acessíveis pelo botão direito do mouse e/ou através de botões disponíveis.

Barra lateral

Do lado esquerdo da tela, na barra lateral, encontramos um navegador para o conteúdo de nossos projetos.
Logo acima do navegador, temos um seletor para o tipo de visualização desejada, a saber:

Projects – este é o *default* e mostra os projetos abertos na sessão com seus respectivos arquivos, separados por tipo.

File System – mostra o conteúdo do diretório selecionado.

Bookmarks – mostra todos os *bookmarks* da sessão atual. *Bookmarks* são marcas que colocamos em determinadas linhas do código.

Open Documents – mostra os arquivos abertos no editor.

Seletor de modo

O Seletor de modo, permite que você selecione rapidamente entre tarefas como edição de arquivos, desenho de interfaces, configuração de montagem e execução dos projetos e depuração de suas aplicações.

Para mudar o modo, clique no ícone ou use o atalho correspondente no teclado.

Os modos possíveis são:

Welcome – este modo exibe a tela de boas vindas, que você já conheceu no começo deste artigo.
Atalho: Ctrl + 1

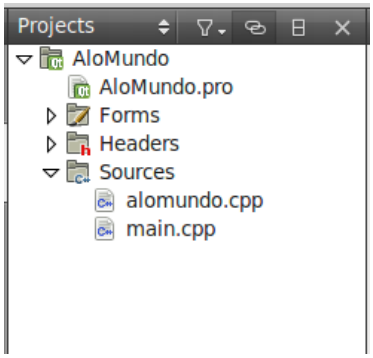
Edit – modo para edição de projetos e arquivos-fonte.
Atalho: Ctrl + 2

Design – modo para desenho de interfaces de usuário.
Atalho: Ctrl + 3

Debug – modo para inspecionar o estado de seu programa enquanto depura.
Atalho: Ctrl + 4

Projects – modo para configurar a montagem e execução de projetos.
Atalho: Ctrl + 5

Help – este modo exibe a documentação do Qt.
Atalho: Ctrl + 6



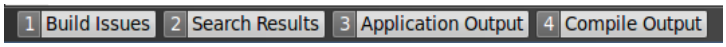
Os painéis de saída permitem que você selecione um item a ser visualizado, dentre os seguintes:

Build Issues – lista de erros e advertências encontradas durante o processo de montagem do projeto.

Search Results – mostra o resultado de pesquisas feitas nos arquivos do projeto

Application Output – mostra o *status* do programa que está sendo executado e as saídas de *debug*.

Compile Output – mostra todas as saídas geradas durante o processo de compilação de um projeto, inclusive os eventuais erros e advertências.



Localizador

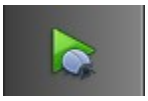
O localizador como o nome diz, localiza elementos nos projetos abertos na sessão.

Seletor de alvo

Quando estiver com mais de um projeto aberto no Qt Creator, use este controle para selecionar o projeto a ser alvo da montagem/execução.
Use-o ainda para selecionar o modo de montagem do projeto: debug ou release (veremos no futuro do que se trata isso).

Depurar

Executa a aplicação em modo de *debug*. Atalho: F5



Montar tudo

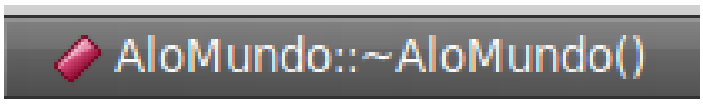
Executa o procedimento de montagem da aplicação.
Atalho: Ctrl – Shift - B

Arquivos abertos

Com este controle você pode selecionar entre os arquivos abertos para edição.

Resumo de símbolos

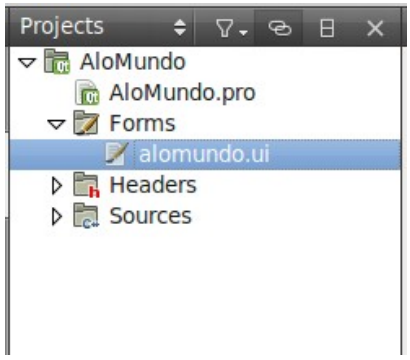
Permite selecionar um dos símbolos relacionados ao arquivo que está aberto no editor. Com o código-fonte de uma classe por exemplo, os nomes dos métodos da mesma serão mostrados neste seletor. Para códigos muito grandes pode ser bastante útil, permitindo que se mova o cursor diretamente para um determinado ponto.



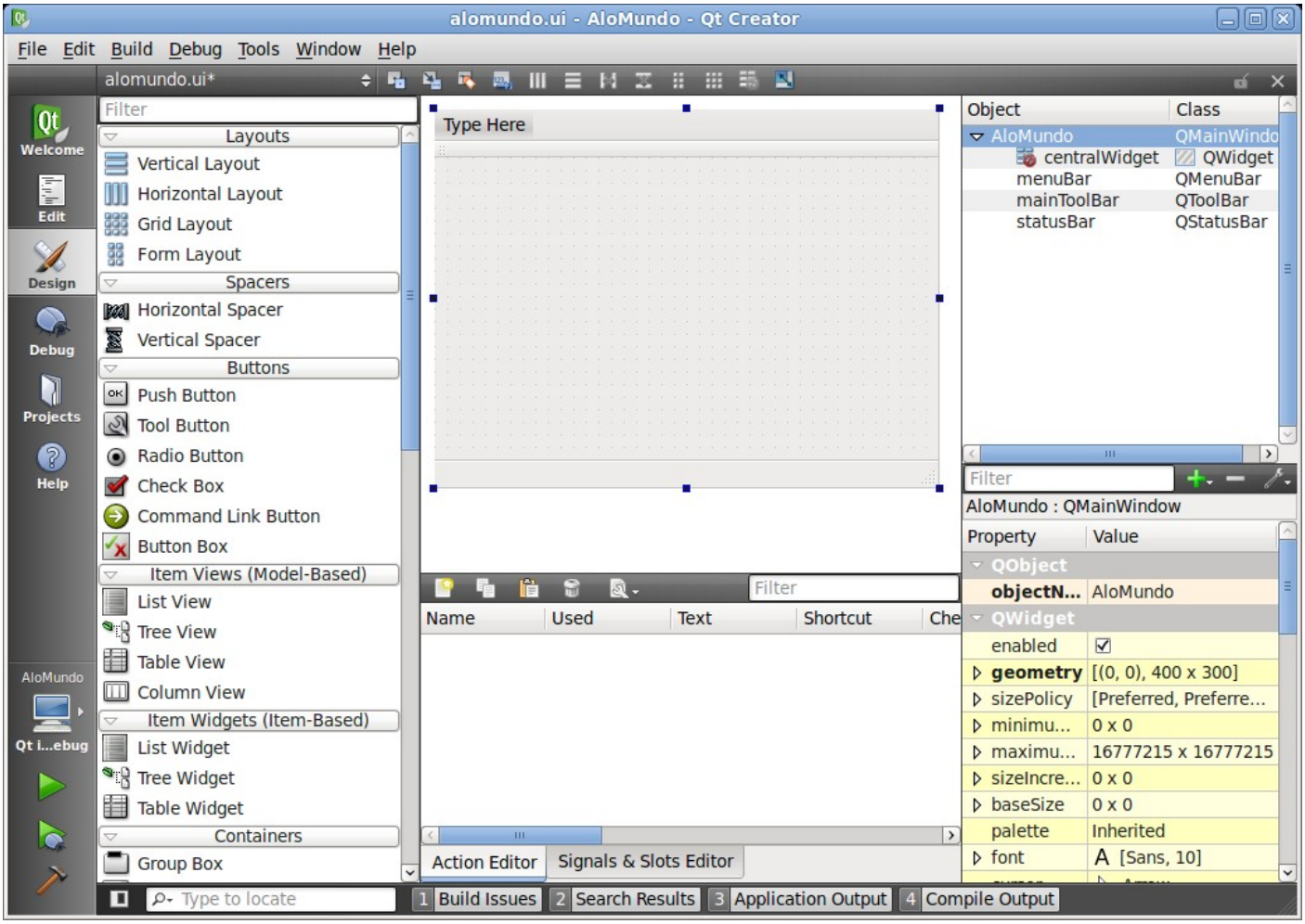
Editor de interface gráfica

No começo deste artigo, foi mencionado que o Qt Creator possui um editor visual integrado para desenho de interfaces gráficas de usuário. Este é um dos pontos mais fortes na utilização do Qt Creator por possibilitar ao programador a criação de interfaces de modo muito simples e sem escrever código.

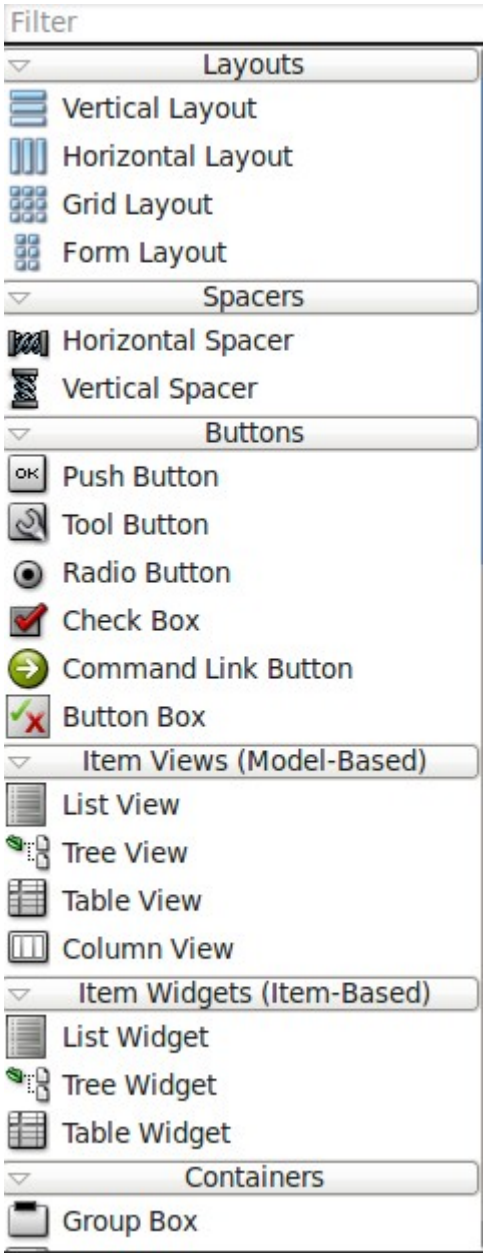
Dê um duplo clique no arquivo alomundo.ui na barra lateral do Qt Creator.



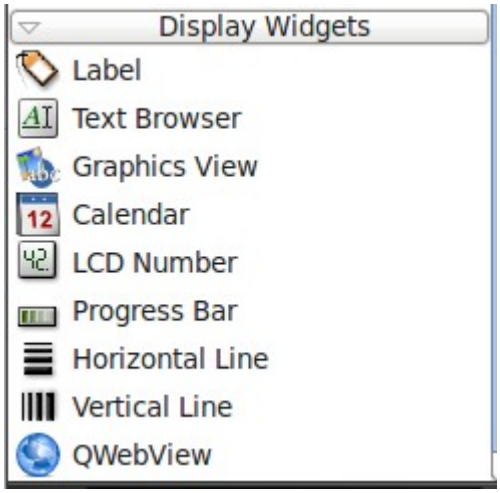
Ao invés de abrir o texto do arquivo alomundo.ui, que é um XML, o Qt Creator “entende” que se trata de um formulário e abre o seu editor interno de interfaces gráficas.



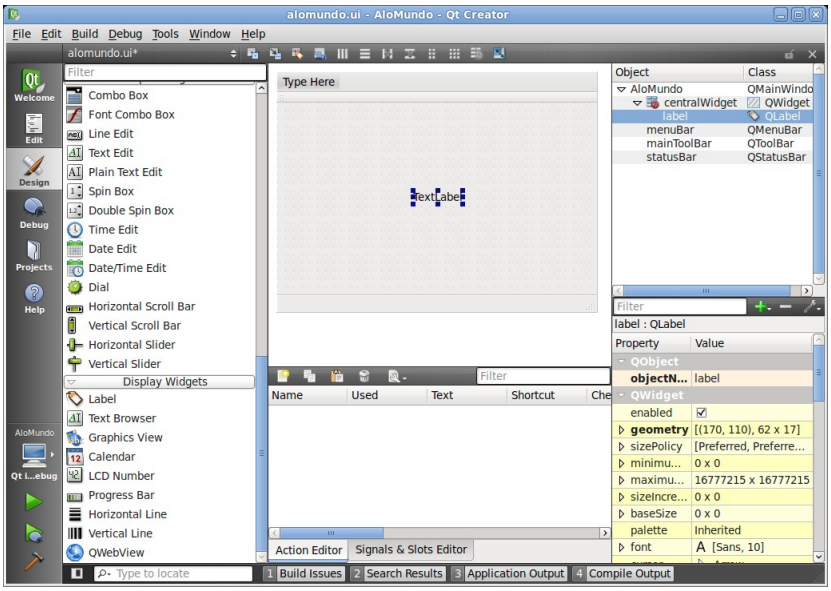
Observe que no lugar da barra lateral com o conteúdo do projeto, temos agora uma paleta de componentes. Estes são os componentes padrão do Qt disponíveis no Qt Creator.



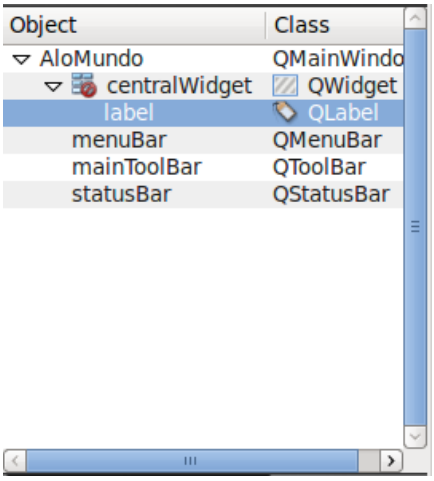
Futuramente veremos cada um destes componentes detalhadamente aqui na Revista Qt. No momento, para concluir o nosso AloMundo, vamos colocar o texto “Alô, mundo!” no centro da janela da nossa aplicação em negrito e com o tamanho de 12px. Veja como é simples:
Procure na paleta de componentes pelo componente **Label** que fica no grupo **Display Widgets.**



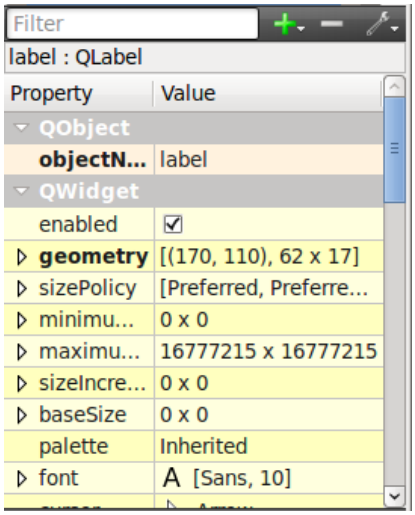
Clique no componente **Label** e arraste-o para o centro da janela. No futuro veremos como fazer para que nossas aplicações tenham um *layout* consistente independente do tamanho da janela ter sido alterado pelo usuário durante a execução do programa. Por enquanto não vamos nos preocupar se o texto “Alô, Mundo!” vai ficar fora da posição que o colocamos quando o usuário maximizar a janela, por exemplo.



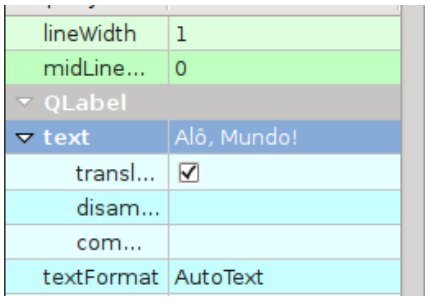
Dê um clique no componente TextLabel para selecioná-lo. Outra forma de selecionar um componente no editor de interfaces é através do **Object Browser** (Navegador de Objetos) que fica na parte superior direita da tela.



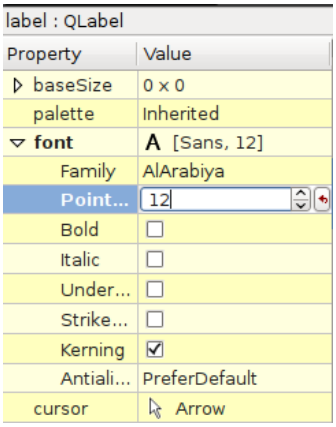
Com um componente selecionado, temos acesso às propriedades do mesmo através do **Property Editor** (Editor de propriedades) localizado logo abaixo do **Object Browser**.



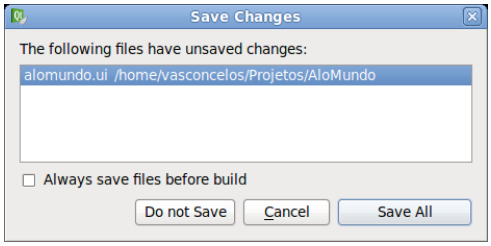
Para alterar o texto exibido pelo **Label**, localize no **Property Editor**, a propriedade **Value** e altere o conteúdo para “Alô, Mundo!”.



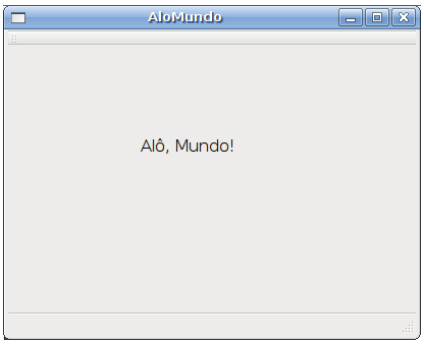
Localize no **Property Editor** a propriedade **font** e altere a propriedade **Point Size** da mesma para 12, como mostra a próxima figura.



Usando o mouse, ajuste o componente **Label** para um tamanho que permita a exibição adequada do texto e execute a aplicação, teclando **CTRL – R** ou clicando no botão **Run**. Como você alterou um arquivo do projeto – o formulário – e não o salvou, o Qt Creator irá emitir um aviso, como mostra a figura a seguir. Se quiser que o Qt Creator salve automaticamente os arquivos alterados quando você mandar executar, marque a opção “**Always save files before build**” e clique no botão “**Save all**”.



Pronto. Concluímos nosso AloMundo com o Qt Creator. Mais uma vez: este artigo é para aqueles que estão tendo o primeiro contato com o Qt Creator, Em outros artigos em próximas edições da Revista Qt, abordaremos tópicos mais avançados desta IDE.



por André Vasconcelos

Qt + PHP – parte 1

Uma abordagem interessante na utilização do Qt é o desenvolvimento de aplicações híbridas: *Desktop* / WEB. Com esta abordagem, você desenvolve uma aplicação cuja interface com o usuário não requer o uso de um navegador (*browser*).

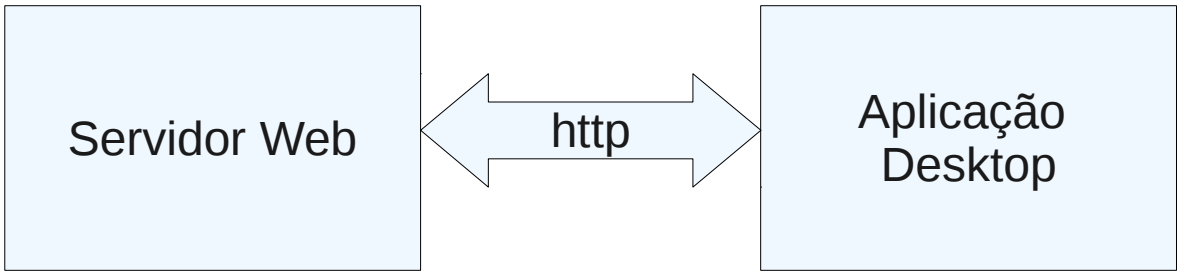
Você deve estar pensando: “Mas espere um pouco? Isso não vai na contramão da tal de *cloud computing*?” Não exatamente. Na verdade, esta abordagem apenas reúne o melhor de dois mundos: a riqueza e consistência da *interface desktop* com a facilidade de distribuição e comunicação da Internet.

Esta abordagem não é nenhuma novidade. Quando utiliza um programa como o Pidgin ou o Amsn para conversar com seus contatos, você está usando uma aplicação *desktop* que utiliza o acesso à Internet como meio de comunicação de dados. Embora existam páginas da Web que permitem acesso online para os serviços de bate-papo, as aplicações possuem recursos mais interessantes, como chamar a atenção do usuário “chacoalhando” a imagem na tela dele, ou habilitar a *webcam* durante uma conversa.

Com nossas aplicações *desktop* conversando com servidores Web, podemos fazer coisas como:

verificar se existem versões mais recentes disponíveis para o usuário
veicular publicidade nas versões grátis dos aplicativos
obter informações online, como cotações de moedas, clima, fuso horário, consulta de CEPs, etc.

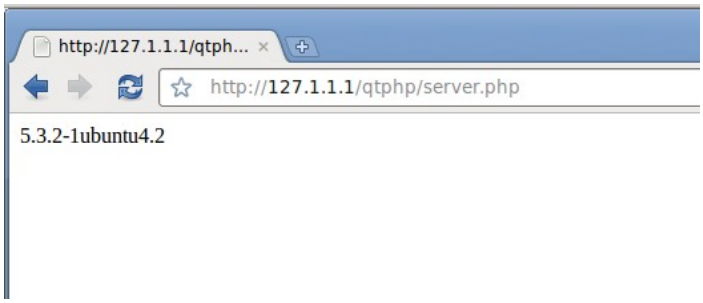
Nesta primeira parte veremos o básico sobre a intercomunicação de uma aplicação em Qt com uma aplicação para Web, escrita em PHP. Para acompanhar este tutorial você deve ter instalado em sua máquina, tanto o Qt como o PHP e o Apache. Se não tiver o servidor Web instalado em sua máquina, uma busca no Google por “instalação de php e apache” lhe trará ótimos tutoriais sobre o assunto.



Vamos começar pela criação da aplicação Web, que retornará a versão do PHP instalada no servidor. É uma aplicação muito simples, mas já serve para esta primeira parte do tutorial. Crie um arquivo server.php com o conteúdo abaixo e salve em um diretório do seu servidor Web:

```
<?php
        echo phpversion();
```

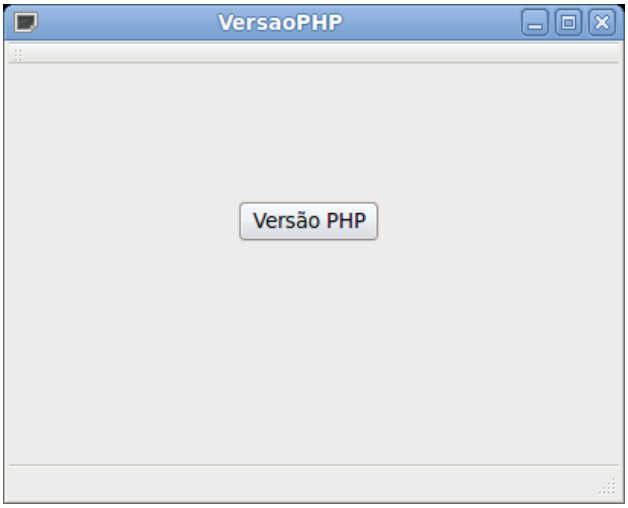
No meu caso, salvei o arquivo em um diretório chamado qtphp no *document root* do Apache instalado em minha máquina, então para acessar o *script* pelo *browser*, tenho que usar o endereço **http://127.1.1.1/qtphp/server.php**.



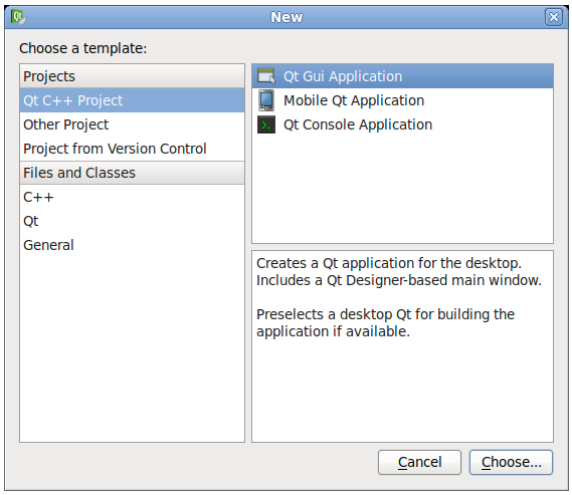
Como resultado, o nosso *script* retorna uma *string* com a versão do PHP instalada no servidor, que meu caso é a "5.3.2-1 ubuntu4.2". O retorno pode ser diferente para você.

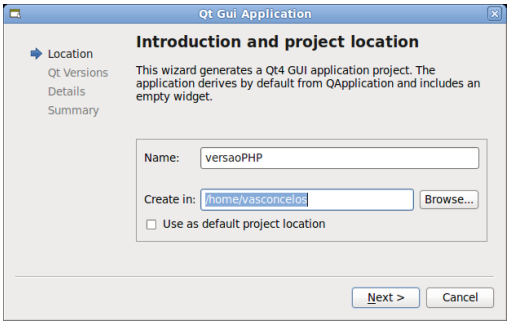
Ok, agora vamos criar a aplicação em Qt que irá acessar este script e mostrar a versão do PHP instalada no servidor.

O objetivo desta pequena aplicação é mostrar a versão do PHP instalada no servidor quando o usuário clicar no botão **Versão PHP**. Veja como deverá ficar a tela do programa:



Usando o Qt Creator, crie uma aplicação chamada **versaoPHP**.





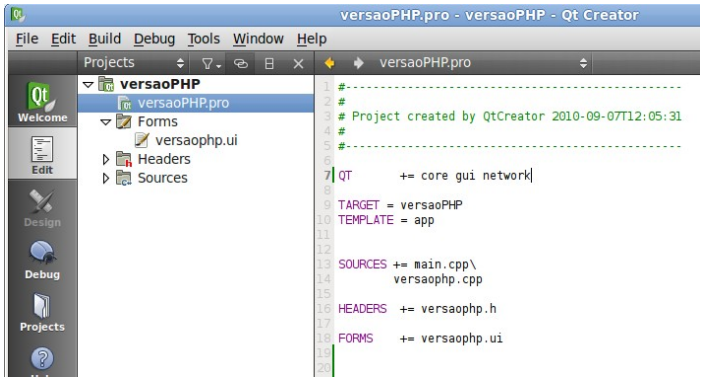
Com o projeto criado, altere no arquivo **versaoPHP.pro**, a linha que contém:

QT += core gui

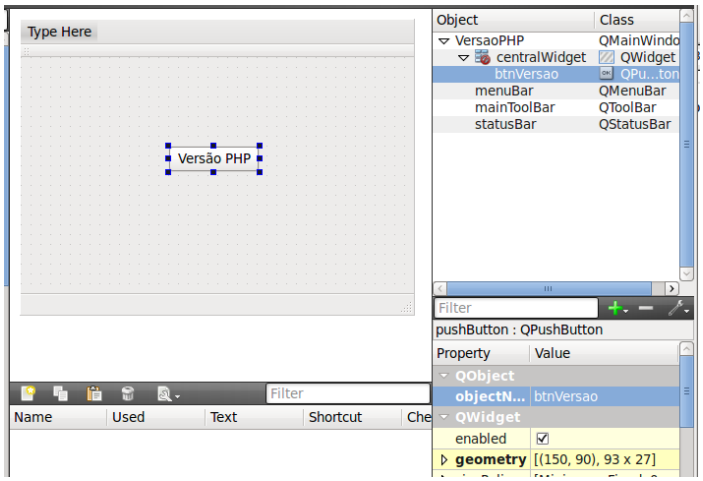
Para:

QT += core gui network

Isto inclui o módulo **QtNetwork** no projeto.



Inclua no formulário – arquivo **versaophp.ui** – um botão, coloque nele o texto "Versão PHP" e mude seu nome para **btnVersao**.



Agora é a hora de programar um pouco. Altere o conteúdo do arquivo *header* – versaophp.h, que passará a ter o seguinte conteúdo:

```
#ifndef VERSAOPHP_H
#define VERSAOPHP_H

#include <QMainWindow>
#include <QHttp>

namespace Ui {
    class VersaoPHP;
}

class VersaoPHP : public QMainWindow
{
    Q_OBJECT

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

private:
    Ui::VersaoPHP *ui;
    QHttp * http;

private slots:
    void perguntaVersaoPHP();
    void mostraVersaoPHP();
};

#endif // VERSAOPHP_H
```

As alterações foram:

Acrescentamos o **#include** para a classe **QHttp**;
Incluímos um atributo privado chamado **http** que é um ponteiro para um objeto **QHttp**;
Declaramos dois **slots** privados, chamados **perguntaVersaoPHP** e **mostraVersaoPHP**.

Se você não conhece como funciona o mecanismo de **sinais** e **slots** do Qt pode se sentir um pouco confuso agora. Em futuras edições da **Revista Qt**, o assunto será abordado mais detalhadamente. Por ora basta entender o sinal como um tipo de evento, uma ação executada em, ou por um determinado objeto. No caso desta nossa aplicação, por exemplo, queremos que ela execute algo quando o usuário clicar no botão. Então vamos associar o sinal **clicked** do botão ao slot **perguntaVersaoPHP**, que nada mais é do que um método da nossa classe que pode ser relacionado a um sinal.

Vamos alterar o conteúdo do arquivo versaophp.cpp para implementar as funcionalidades da nossa aplicação:

```
#include "versaophp.h"
#include "ui_versaophp.h"
#include <QMessageBox>

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


    connect(ui->btnVersao, SIGNAL(clicked()), this, SLOT(perguntaVersaoPHP()));
}

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

void VersaoPHP::perguntaVersaoPHP()
{
    this->http = new QHttp();
    this->http->setHost("127.1.1.1");
    connect(this->http, SIGNAL(done(bool)), this, SLOT(mostraVersaoPHP()));
    this->http->get("/qtphp/server.php");
}

void VersaoPHP::mostraVersaoPHP()
{
    QMessageBox::information(this, "Info", this->http->readAll());
}
```

Acrescentamos o **#include** para a classe QMessageBox, que será utilizada para exibir ao usuário a versão do PHP.

Observe que no método construtor foi acrescentada uma linha para conectar o sinal **clicked** do botão **btnVersao** com o slot **perguntaVersaoPHP**, pela instrução **connect**. Além disso, foram implementados os códigos dos slots **perguntaVersaoPHP** e **mostraVersaoPHP**.

Vejamos o código do slot **perguntaVersaoPHP**:

```
void VersaoPHP::perguntaVersaoPHP()
{
    this->http = new QHttp();
    this->http->setHost("127.1.1.1");
    connect(this->http, SIGNAL(done(bool)), this, SLOT(mostraVersaoPHP()));
    this->http->get("/qtphp/server.php");
}
```

Primeiro instanciamos um objeto do tipo QHttp que será utilizado para envio de uma requisição ao servidor Web. Note que estamos usando um atributo da nossa classe, chamado http, que foi declarado no arquivo header.
Com o objeto instanciado, *setamos* o atributo **host** com o endereço do servidor Web. No meu caso, como estou utilizando a minha máquina como servidor, coloquei o endereço local – 127.1.1.1.
Lá vem os sinais e slots de novo. Precisamos indicar à nossa aplicação, qual método – um slot no caso – será executado quando o objeto QHttp obtiver a resposta do servidor. O sinal emitido pelo objeto QHttp quando este recebe a resposta do servidor é o **done(bool)**. Nesta aplicação, conectamos o sinal **done(bool)** do objeto **http** ao slot **mostraVersaoPHP**.
A última instrução do método perguntaVersaoPHP contém a requisição ao servidor, propriamente dita. Como eu coloque o script criado no começo do artigo – server.php em um diretório chamado qtphp no meu servidor Web, então a requisição ficou como "/qtphp/server.php".

Trocando em miúdos, a requisição feita pelo objeto **http** nesta aplicação será para a seguinte URL:

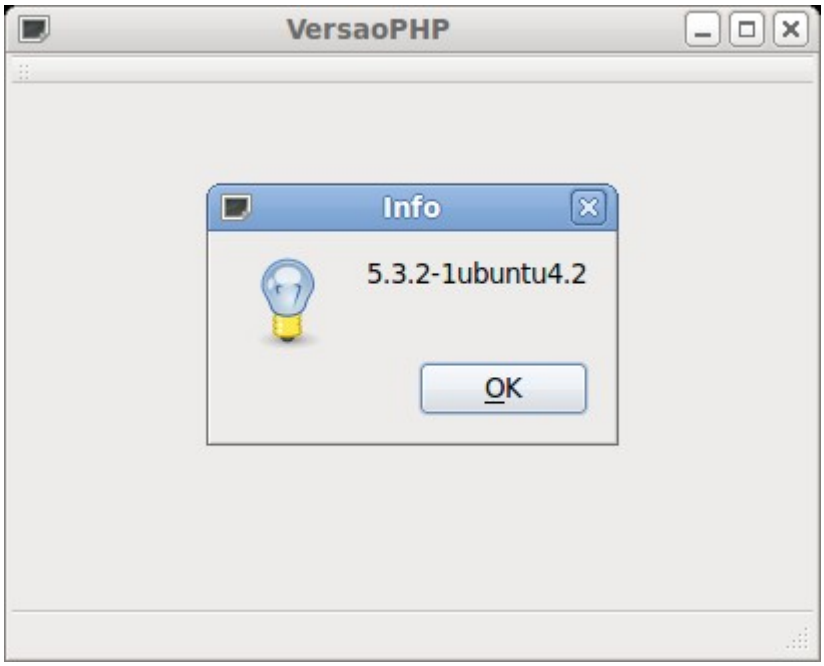
http://127.1.1.1/qtphp/server.php

```
void VersaoPHP::mostraVersaoPHP()
{
    QMessageBox::information(this, "Info", this->http->readAll());
}
```

Em seguida, temos a implementação do slot **mostraVersaoPHP**, que como vimos será executado quando o objeto **http** obtiver a resposta à sua requisição.

A única instrução presente neste método serve para exibir o resultado da requisição para o usuário. Utilizamos aqui uma das caixas de diálogo padrão do Qt, através da classe **QmessageBox**.

A resposta enviada pelo servidor ao objeto **http** pode ser "lida" através do método **readAll()**. Este método retorna uma string, que neste programa, estamos simplesmente exibindo para o usuário.



A figura acima mostra o resultado da execução desta aplicação, quando clicamos no botão **Versão PHP**. Claro que o resultado depende da versão de PHP que você tenha instalada no servidor Web utilizado.

Esta foi apenas uma introdução ao assunto. Em edições futuras, veremos mais detalhes sobre a classe QHttp, bem como aplicações mais complexas utilizando-a. Se estiver impaciente, você pode recorrer à referência online sobre o assunto, disponível pelo endereço:

http://doc.trolltech.com/4.6/qhttp.html

Até a próxima.

por André Vasconcelos

Espaço do Leitor

Caríssimo(a) leitor(a),

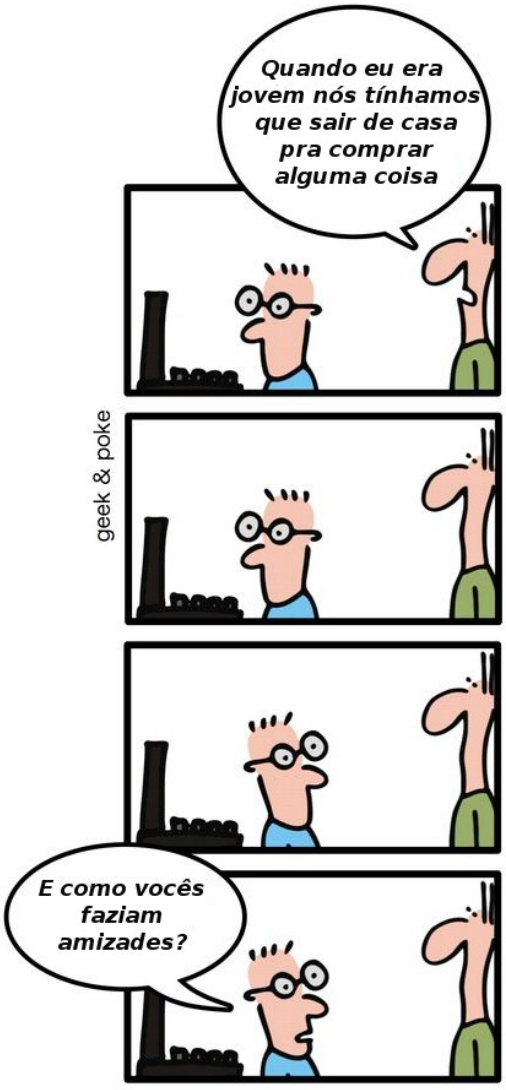
Da sua participação, depende o sucesso da Revista Qt. Mande e-mails para revistaqt@gmail.com com suas dúvidas, críticas e sugestões.

Publicidade sobre cursos, palestras, seminários, ou quaisquer outros eventos relacionados ao Qt serão publicados gratuitamente em nossa revista.

Conto com você.

Um grande abraço

André Luiz de O. Vasconcelos



Graças a Deus, até fazer compras é uma atividade social hoje em dia