

# **QTestLib framework: Criando unit tests inteligentes para projetos baseados em Qt**

**Bruno Abinader**  
22 de Julho de 2010



# CONTEÚDO (1/2)

## Introdução

- Quem somos?
- O que é o Qt?
- O que são unit tests?

## Ferramentas de testes do Qt

- Framework QTestLib
- QTest namespace
- Benchmarks

# CONTEÚDO (2/2)

## Escrevendo unit tests

- Criando uma classe de teste
- Comparando valores
- Utilizando diferentes valores de entrada
- Simulação de eventos

## Tópicos avançados

- Eventos em widgets baseados no QGraphicsView
- Acessando membros privados

# INTRODUÇÃO

Quem somos?



# INTRODUÇÃO

O que é o Qt?



**Code less.  
Create more.  
Deploy everywhere.**

Framework multiplataforma para desenvolvimento de aplicações



# INTRODUÇÃO

O que são unit tests?

## Características:

- Teste de blocos individuais e independentes de código
- Validação do comportamento esperado
- Mensura a qualidade do código gerado
- Testes podem ser automatizados

## Benefícios:

- Simplifica a integração de código
- Evita bugs de regressão
- Detecção rápida de bottlenecks



# FERRAMENTAS DE TESTES DO QT

## Framework QTestLib

### Características:

- Leve: ~6000 LOC e ~60 símbolos exportados
- Auto-contido: *QT = core testlib*
- Testes são binários independentes
- Suporte a *Data Driven Testing*
- Separação entre lógica e dados
- Simulação de eventos de teclado e mouse
- Introspecção de sinais/slots
- Suporte a benchmarking



# FERRAMENTAS DE TESTES DO QT

## QTest namespace

### Conteúdo:

- **Métodos públicos:** *Benchmarking, Data Driven Testing, Simulação de Eventos, Sleep/Wait do Loop de Eventos, Facilitadores de Output*
- **Macros:** *QBENCHMARK, QCOMPARE, QTEST, QVERIFY* e outros
- **Tipos:** *KeyAction, MouseAction, TestFailMode* e outros
- **Classes:** *QSignalSpy* e *QTouchEventSequence*



# FERRAMENTAS DE TESTES DO QT

## Benchmarks

**Macros: QBENCHMARK / QBENCHMARK\_ONCE**

### Backends:

- **Walltime** (padrão – repete o código várias vezes)
- **Tick Counter** (menos repetições, problemas com frequency scaling)
- **Valgrind/Callgrind** (resultados exatos, não leva em conta I/O)
- **Event Counter** (provê o número de eventos gerados)



# ESCREVENDO UNIT TESTS

## Criando uma classe de teste

### O que é necessário:

- Herdar de QObject
- Implementar cada teste em separado como *private slots*
- 4 *slots* especiais tratados como não-teste:

**initTestCase()**

**cleanupTestCase()**

**init()**

**cleanup()**



# ESCREVENDO UNIT TESTS

## Comparando valores

### Exemplo:

```
class MinhaClasse: public QObject
{
    Q_OBJECT
private slots:
    void initTestCase()
    { qDebug("chamado antes de começar a executar os testes"); }
    void primeiroTeste()
    { QVERIFY(1 == 1); }
    void segundoTest()
    { QString valor("teste"); QCOMPARE(QString("teste"), valor); }
    void cleanupTestCase()
    { qDebug("chamado depois da execução de todos os testes"); }
};
```



# ESCREVENDO UNIT TESTS

Utilizando diferentes valores de entrada

## Exemplo:

```
class MinhaClasse: public QObject
{
    Q_OBJECT
private slots:
    void meuTeste_data()
    { QTest::addColumn<QString>("texto");
      QTest::newRow("texto igual") << "fulano";
      QTest::newRow("texto diferente") << "sicrano"; }
    void meuTeste()
    { QFETCH(QString, texto);
      QCOMPARE(texto, "fulano");
    }
};
```



# ESCREVENDO UNIT TESTS

## Simulação de eventos

### Exemplo:

```
class MinhaClasse: public QObject
{
    Q_OBJECT
private slots:
    void testeEventoTeclado()
    { QlineEdit campoTexto;
      QTest::keyClicks(&campoTexto, "olá FISL");
      QCOMPARE(campoTexto.text(), "olá FISL"); }
    void testeEventoMouse()
    { QPushButton botao;
      QsignalSpy espiaoCliqueBotao(&botao, SIGNAL(clicked()));
      QTest::MouseEvent(&botao, Qt::LeftButton);
      QVERIFY(espiaoCliqueBotao.count() == 1); }
};
```



# TÓPICOS AVANÇADOS

## Eventos em itens do QGraphicsWidget

### Exemplo:

```
class MinhaClasse: public QObject
{
    Q_OBJECT
private slots:
    void testeQGV()
    { QGraphicsWidget widget;
      scene.addItem(&widget);
      ...
      QtTest::mouseClick(view.viewport(), Qt::LeftButton, 0,
                          widget.geometry().center().mapToScene()); }
};
```



# TÓPICOS AVANÇADOS

## Acessando membros privados

### Método 1: Friend class

```
class myWidget : public QWidget
...
friend class myWidgetTest;
...
```

### Método 2: Include hack

```
#define private public
#define protected public
#include "mywidget.h"
#undef protected
#undef private
```



# Obrigado!

## Perguntas?

Contato:

[bruno.abinader@openbossa.org](mailto:bruno.abinader@openbossa.org) ([abinader @ irc.freenode.org](irc://irc.freenode.org))

Próximas palestras:

“Plasma Animations Overview” (23/07 - 13:00)

“Desenvolvendo aplicações para dispositivos Nokia usando Linux” (23/7 - 13:00)

“Gallium3D: Understand the Linux Graphical Infrastructure” (24/07 - 17:00)

