Міністерство освіти і науки України

Запорізький національний технічний університет

Кафедра програмних засобів

ЗВІТ

З лабораторної роботи №5

з дисципліни «Якість програмного забезпечення та тестування» на тему:

«ПРОВЕДЕННЯ ТЕСТІВ В QT»

Виконав:

ст. гр. КНТ-415 І.О. Лимарєв

Прийняв:

к.т.н., доцент Т. І. Каплієнко

2018

**Мета роботи:** Вивчити можливості спеціального модулю QtTestLib та реалізувати тести із застосуванням мови С++ та Qt.

***Код програми:***

***test\_event.h***

#ifndef TEST\_EVENT\_H

#define TEST\_EVENT\_H

#include <QtTest>

#include "event.h"

class Test\_Event : public QObject {

Q\_OBJECT

private slots:

void NameOfTask\_data();

void StartDate\_data();

void FinishDate\_data();

void StartTime\_data();

void FinishTime\_data();

void RepeatOfTask\_data();

void RemindOfTask\_data();

void RemindComplete\_data();

void RemindDate\_data();

void RemindTime\_data();

void DescriptionOfTask\_data();

void Color\_data();

void NameOfTask();

void StartDate();

void FinishDate();

void StartTime();

void FinishTime();

void RepeatOfTask();

void RemindOfTask();

void RemindComplete();

void RemindDate();

void RemindTime();

void DescriptionOfTask();

void Color();

};

#endif // TEST\_EVENT\_H

***test\_event.cpp***

#include "test\_event.h"

void Test\_Event::NameOfTask\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QString>("result");

QTest::newRow("NameOfTask\_test1") << "Task" << "Task";

QTest::newRow("NameOfTask\_test2") << "Задача" << "Задача";

QTest::newRow("NameOfTask\_test3") << "ІІІїїїїііЇ" << "ІІІїїїїііЇ";

}

void Test\_Event::DescriptionOfTask\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QString>("result");

QTest::newRow("DescriptionOfTask\_test1") << "Description" << "Description";

QTest::newRow("DescriptionOfTask\_test2") << "Описание" << "Описание";

QTest::newRow("DescriptionOfTask\_test3") << "ІІІїїїїііЇ" << "ІІІїїїїііЇ";

}

void Test\_Event::StartDate\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QDate>("result");

QTest::newRow("StartDate\_test1") << QString("10.07.1993") << QDate(1993, 7, 10);

QTest::newRow("StartDate\_test2") << QString("07.11.2018") << QDate::fromString("07.11.2018", "dd.MM.yyyy");

QTest::newRow("StartDate\_test3") << QString("01.01.2019") << QDate::fromString("01.01.2019", "dd.MM.yyyy");

}

void Test\_Event::FinishDate\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QDate>("result");

QTest::newRow("FinishDate\_test1") << QString("10.07.1993") << QDate(1993, 7, 10);

QTest::newRow("FinishDate\_test2") << QString("07.11.2018") << QDate::fromString("07.11.2018", "dd.MM.yyyy");

QTest::newRow("FinishDate\_test3") << QString("01.01.2019") << QDate::fromString("01.01.2019", "dd.MM.yyyy");

}

void Test\_Event::RemindDate\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QDate>("result");

QTest::newRow("FinishDate\_test1") << QString("10.07.1993") << QDate(1993, 7, 10);

QTest::newRow("FinishDate\_test2") << QString("07.11.2018") << QDate::fromString("07.11.2018", "dd.MM.yyyy");

QTest::newRow("FinishDate\_test3") << QString("01.01.2019") << QDate::fromString("01.01.2019", "dd.MM.yyyy");

}

void Test\_Event::StartTime\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QTime>("result");

QTest::newRow("StartTime\_test1") << QString("00:00") << QTime(0, 0);

QTest::newRow("StartTime\_test2") << QString("20:10") << QTime::fromString("20:10", "hh:mm");

QTest::newRow("StartTime\_test3") << QString("13:15") << QTime::fromString("13:15", "hh:mm");

}

void Test\_Event::FinishTime\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QTime>("result");

QTest::newRow("FinishTime\_test1") << QString("00:00") << QTime(0, 0);

QTest::newRow("FinishTime\_test2") << QString("20:10") << QTime::fromString("20:10", "hh:mm");

QTest::newRow("FinishTime\_test3") << QString("13:15") << QTime::fromString("13:15", "hh:mm");

}

void Test\_Event::RemindTime\_data()

{

QTest::addColumn<QString>("arg");

QTest::addColumn<QTime>("result");

QTest::newRow("FinishTime\_test1") << QString("00:00") << QTime(0, 0);

QTest::newRow("FinishTime\_test2") << QString("20:10") << QTime::fromString("20:10", "hh:mm");

QTest::newRow("FinishTime\_test3") << QString("13:15") << QTime::fromString("13:15", "hh:mm");

}

void Test\_Event::RepeatOfTask\_data(){

QTest::addColumn<short>("arg");

QTest::addColumn<short>("result");

QTest::newRow("RepeatOfTask\_test1") << short(1) << short(1);

QTest::newRow("RepeatOfTask\_test2") << short(3) << short(3);

QTest::newRow("RepeatOfTask\_test3") << short(5) << short(5);

}

void Test\_Event::RemindOfTask\_data(){

QTest::addColumn<short>("arg");

QTest::addColumn<short>("result");

QTest::newRow("RemindOfTask\_test1") << short(1) << short(1);

QTest::newRow("RemindOfTask\_test2") << short(3) << short(3);

QTest::newRow("RemindOfTask\_test3") << short(5) << short(5);

}

void Test\_Event::RemindComplete\_data()

{

QTest::addColumn<short>("arg");

QTest::addColumn<short>("result");

QTest::newRow("RemindComplete\_test1") << short(1) << short(1);

QTest::newRow("RemindComplete\_test2") << short(3) << short(3);

QTest::newRow("RemindComplete\_test3") << short(5) << short(5);

}

void Test\_Event::Color\_data()

{

QTest::addColumn<QColor>("arg");

QTest::addColumn<QColor>("result");

QTest::newRow("Color\_test1") << QColor(255, 0, 0, 127) << QColor(255, 0, 0, 127);

QTest::newRow("Color\_test2") << QColor(0, 0, 0, 127) << QColor(0, 0, 0, 127);

QTest::newRow("Color\_test3") << QColor(255, 255, 255, 127) << QColor(255, 255, 255, 127);

}

void Test\_Event::StartDate(){

Event myEvent;

QFETCH(QString, arg);

QFETCH(QDate, result);

myEvent.SetStartDate(arg);

QCOMPARE(myEvent.GetStartDate(), result);

}

void Test\_Event::FinishDate(){

Event myEvent;

QFETCH(QString, arg);

QFETCH(QDate, result);

myEvent.SetFinishDate(arg);

QCOMPARE(myEvent.GetFinishDate(), result);

}

void Test\_Event::RemindDate(){

Event myEvent;

QFETCH(QString, arg);

QFETCH(QDate, result);

myEvent.SetRemindDate(arg);

QCOMPARE(myEvent.GetRemindDate(), result);

}

void Test\_Event::StartTime(){

Event myEvent;

QFETCH(QString, arg);

QFETCH(QTime, result);

myEvent.SetStartTime(arg);

QCOMPARE(myEvent.GetStartTime(), result);

}

void Test\_Event::FinishTime(){

Event myEvent;

QFETCH(QString, arg);

QFETCH(QTime, result);

myEvent.SetFinishTime(arg);

QCOMPARE(myEvent.GetFinishTime(), result);

}

void Test\_Event::RemindTime()

{

Event myEvent;

QFETCH(QString, arg);

QFETCH(QTime, result);

myEvent.SetRemindTime(arg);

QCOMPARE(myEvent.GetRemindTime(), result);

}

void Test\_Event::RepeatOfTask(){

Event myEvent;

QFETCH(short, arg);

QFETCH(short, result);

myEvent.SetRepeatOfTask(arg);

QCOMPARE(myEvent.GetRepeatOfTask(), result);

}

void Test\_Event::RemindOfTask(){

Event myEvent;

QFETCH(short, arg);

QFETCH(short, result);

myEvent.SetRemindOfTask(arg);

QCOMPARE(myEvent.GetRemindOfTask(), result);

}

void Test\_Event::RemindComplete(){

Event myEvent;

QFETCH(short, arg);

QFETCH(short, result);

myEvent.SetRemindComplete(arg);

QCOMPARE(myEvent.GetRemindComplete(), result);

}

void Test\_Event::NameOfTask(){

Event myEvent;

QFETCH(QString, arg);

QFETCH(QString, result);

myEvent.SetNameOfTask(arg);

QCOMPARE(myEvent.GetNameOfTask(), result);

}

void Test\_Event::DescriptionOfTask(){

Event myEvent;

QFETCH(QString, arg);

QFETCH(QString, result);

myEvent.SetDescriptionOfTask(arg);

QCOMPARE(myEvent.GetDescriptionOfTask(), result);

}

void Test\_Event::Color(){

Event myEvent;

myEvent.SetColor(QColor(255, 0, 0, 127));

QCOMPARE(myEvent.GetColor(), QColor(255, 0, 0, 127));

}

QTEST\_MAIN(Test\_Event)

***guitest.h***

#ifndef GUITEST\_H

#define GUITEST\_H

#include <QtWidgets>

#include <QRadioButton>

#include <QtTest>

#include <QtGui>

class TestGui: public QObject

{

Q\_OBJECT

private slots:

void testQRadioButton();

};

#endif // GUITEST\_H

***guitest.cpp***

#include "guitest.h"

void Test\_QDateEdit::edit(){

QDateEdit date;

QTest::keyClicks(&date, "88");

QCOMPARE(date.date(), QDate(1993, 07, 01));

}

QTEST\_MAIN(Test\_QDateEdit)

**Мета роботи:** Вивчив можливості спеціального модулю QtTestLib та реалізувати тести із застосуванням мови С++ та Qt.