Technical University of Moldova

Faculty of Computers, Informatics and Microelectronics

Department of Automatics and Information Technologies

Laboratory work #2

on Interactive Development Environments for Sowtware Products
GUI Calculator

Executed:
st.gr TI-141(fr) Iulian Maşaev

Verified:
assistant lecturer Irina Cojanu

Chişinău, 2017

Prerequisities

- IDE: QTCreator
- Programming Language: C/C++
- Framework: QtWidgets

Objectives

- Create GUI calculator
- Operations: $+, -, *, /, x^2, \sqrt{x}$,invert sign, with floating point support
- Divide project in two modules - GUI and Core module

Task realization

For this task I used Qt Creator IDE. At the beggining I designed window using graphical drag & drop staff from **Design** tab. I added buttons for digits, required operations, and additionally an 'C' button, to erase current operand and also for getting full grid 4 by 5 buttons. Of course an Text Edit box(with disabled editing). Figure 1 shows screenshot of finished window. Red boxes are grid auto alignments.

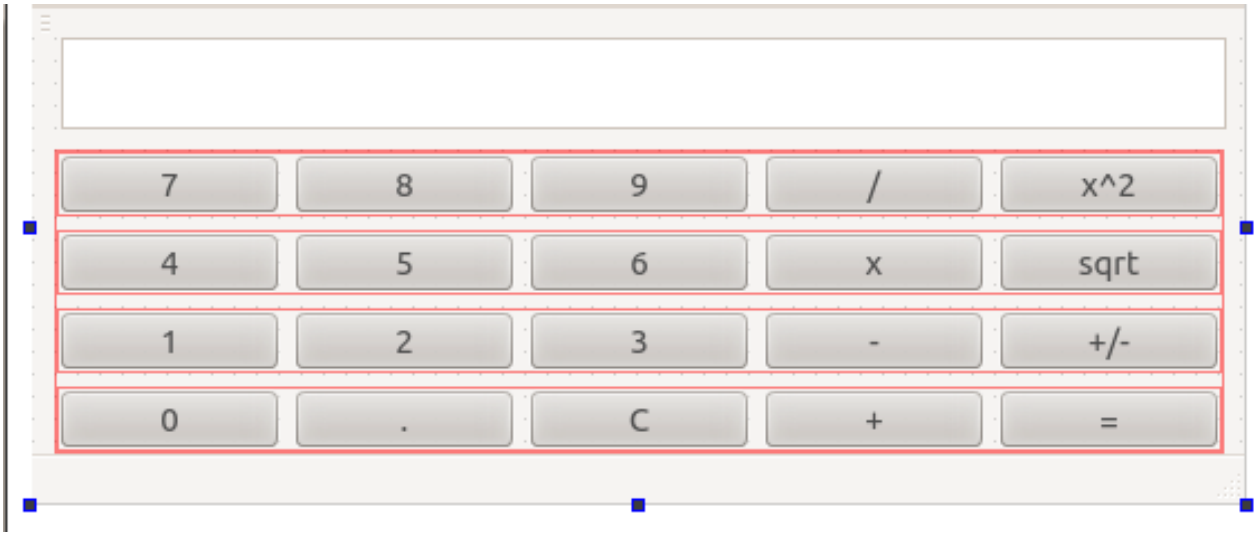

Figure 1: Design

After that, I started adding slots, and adding connections from `clicked()` signal to window. All digits I connected to the same slot, `digitClicked`

```
void MainWindow::digitClicked(){
 QPushButton *clickedButton = qobject_cast<QPushButton *>(sender());
 int digitValue = clickedButton->text().toInt();
 if(ui->summator->toPlainText() == "0" && digitValue==0.0)
     return;
 if (waitingForOperand) {
     ui->summator->clear();
```

```
        waitingForOperand = false;
    }
    ui->summator->setText(ui->summator->toPlainText() + QString::number(digitValue));
}
```

We get necessary digit from `text().toInt();` of the sender, because we know that sender will be a button, and it's text field is equal to value of the digit. `ui->summator` actually is display of our calculator, I'm too lazy to find a prettier way to access it. Of course it can be declared as a field of MainWindow class, but ui is already a member, so it will become redundant.

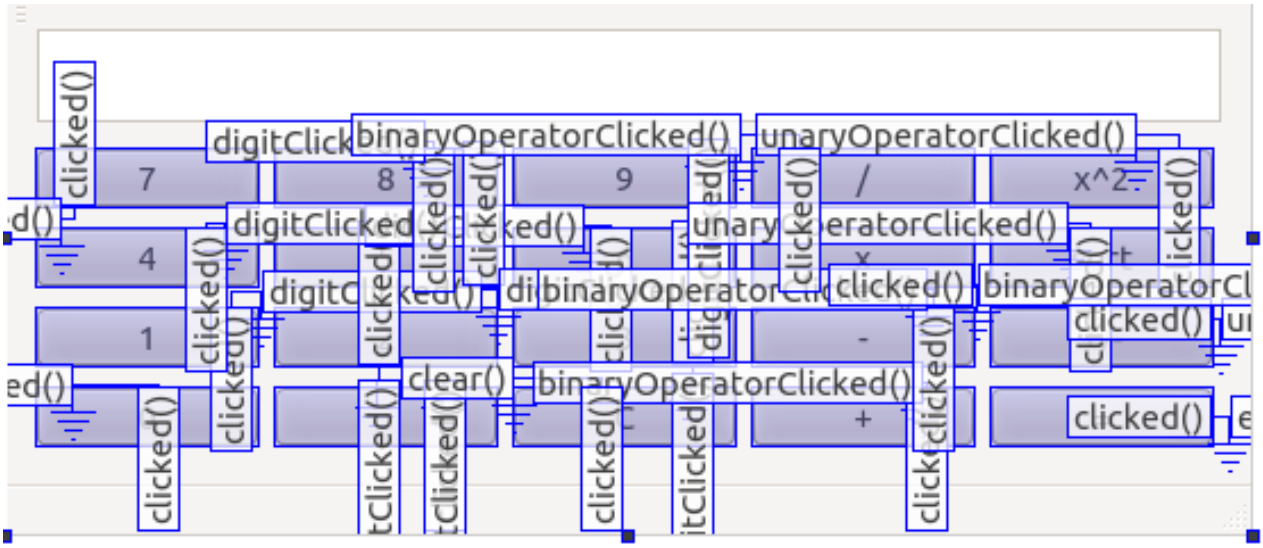

Figure 2: Connected signals

After connecting the signals(see Figure 2, it is messy because there are lot of signals and not so much space) and testing input, I proceeded with simplest operations, unary, because you need just one operand.

```
void MainWindow::unaryOperatorClicked(){
    QPushButton *clickedButton = qobject_cast<QPushButton *>(sender());
    QString clickedOperator = clickedButton->text();
    double operand = ui->summator->toPlainText().toDouble();
    double result = 0.0;

    if (clickedOperator == tr("sqrt")) {
        result = sqrt(operand);
    }
    else if (clickedOperator == tr("x^2")){
        result = pow(operand, 2.0);
    }
    else if (clickedOperator == tr("+/-")){
        result = -operand;
    }
    ui->summator->setText(QString::number(result));
```

```
    waitingForOperand = true;
}
```

There is no check, but in case of division by zero in UI will be displayed *Inf*, in case of square root from negative - *NaN*. Also it would look clearer with switch-case construct, but from some strange reason Qt seems to accept `switch` only on integers.

In case of binary operation, we store curent value, store operation, and wait next operand. When clicking on equal - the operand and operation from the class atributes will be used:

```
void MainWindow::binaryOperatorClicked(){
    QPushButton *clickedButton = qobject_cast<QPushButton *>(sender());
    QString clickedOperator = clickedButton->text();
    double operand = ui->summator->toPlainText().toDouble();
    ui->summator->setText("0");
    sumInMemory=operand;
    pendingOperator=clickedOperator;
    waitingForOperand = true;
}

void MainWindow::equalClicked(){
    double operand = ui->summator->toPlainText().toDouble();
    double result = 0.0;
    if (pendingOperator=="/") result = sumInMemory/operand;
    else if (pendingOperator=="-") result = sumInMemory-operand;
    else if (pendingOperator=="+") result = sumInMemory+operand;
    else if (pendingOperator=="x") result = sumInMemory*operand;

    ui->summator->setText(QString::number(result));
    sumInMemory=0.0;
    waitingForOperand = true;
}
```

For consistent conversion to double, point will be appendend to display value only in case if it isn't already present:

```
if (!ui->summator->toPlainText().contains("."))
    ui->summator->setText(ui->summator->toPlainText() + tr("."));
```

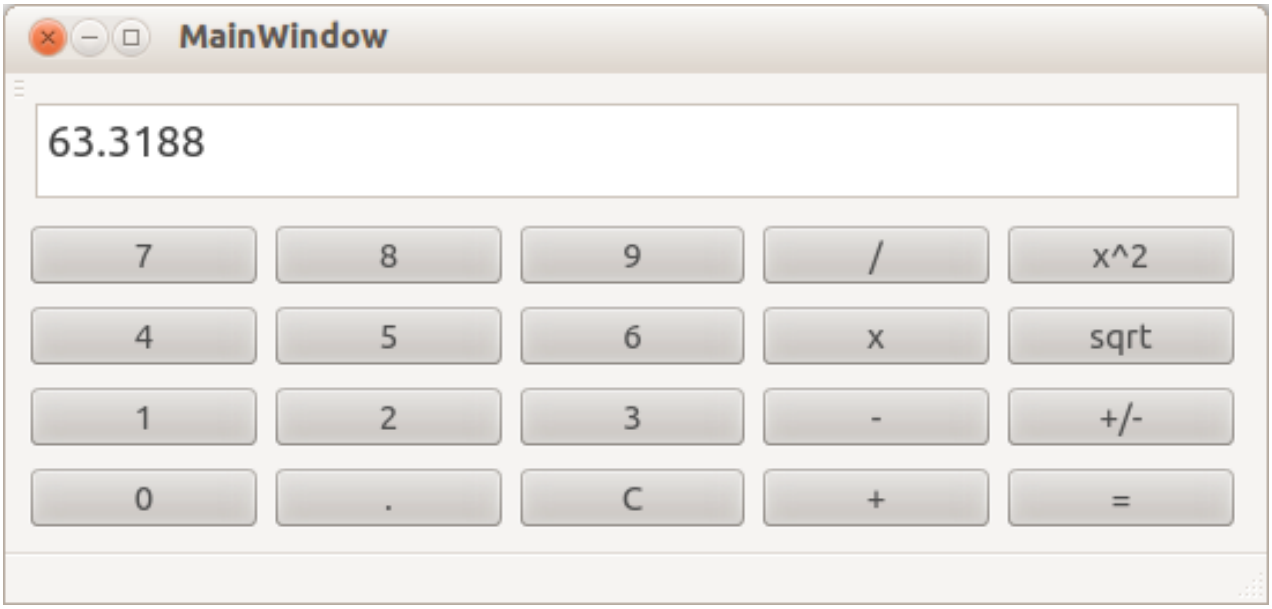

Figure 3: Working application