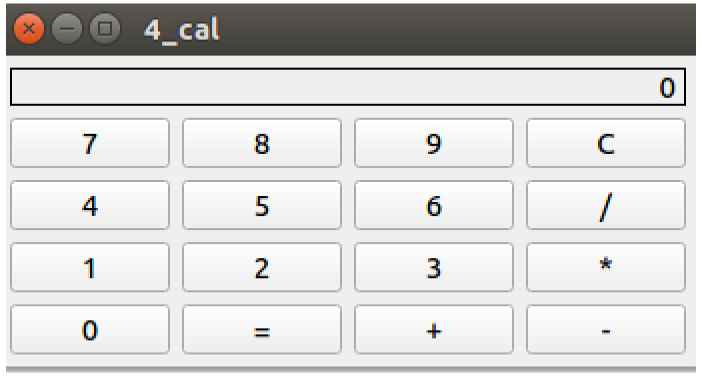
```
QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

CONFIG += c++11

DEFINES += QT_DEPRECATED_WARNINGS

SOURCES += ₩
    main.cpp ₩
    widget.cpp

HEADERS += ₩
    widget.h

TEMPLATE = app
TARGET = program
DESTDIR = exe

# Default rules for deployment.
qnx: target.path = /tmp/$${TARGET}/bin
else: unix:!android: target.path = /opt/$${TARGET}/bin
!isEmpty(target.path): INSTALLS += target

message(The $$TEMPLATE $$TARGET will be installed in $$DESTDIR)
```



1. 계산기 1차

widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

class QLabel;

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

private:
    QLabel* label;
    QString numberTemp;
    QString operate;

public slots:
    void setNum();
    void operation();
    void calculate();
    void clear();
};
#endif // WIDGET_H
```

widget.cpp 파일

```
#include "widget.h"

#include <QVBoxLayout>
#include <QGridLayout>
#include <QPushButton>
#include <QLabel>

Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    const char BtnChar[16][2] = {
        "7", "8", "9", "C",
        "4", "5", "6", "/",
        "1", "2", "3", "*",
        "0", "=", "+", "-"
    };

    label = new QLabel("0", this);
    label->setAlignment(Qt::AlignRight);
    label->setFrameShape(QFrame::Box);

    QPushButton* btn[16];
    QGridLayout* gridLayout = new QGridLayout();

    for(int i = 0; i < 16; i++)
    {
        btn[i] = new QPushButton(BtnChar[i], this);
        gridLayout->addWidget(btn[i], i/4, i%4);
    }

    connect(btn[0], SIGNAL(clicked()), SLOT(setNum())); // 7
    connect(btn[1], SIGNAL(clicked()), SLOT(setNum())); // 8
    connect(btn[2], SIGNAL(clicked()), SLOT(setNum())); // 9
    connect(btn[3], SIGNAL(clicked()), SLOT(operation())); // C
```

widget.h 파일

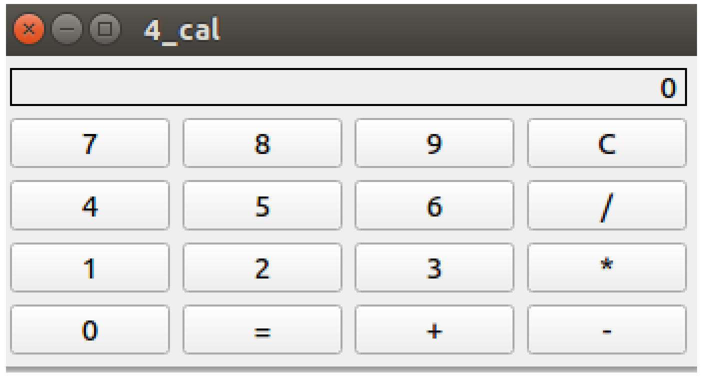
```
    connect(btn[4], SIGNAL(clicked()), SLOT(setNum())); // 4
    connect(btn[5], SIGNAL(clicked()), SLOT(setNum())); // 5
    connect(btn[6], SIGNAL(clicked()), SLOT(setNum())); // 6
    connect(btn[7], SIGNAL(clicked()), SLOT(operation())); // /

    connect(btn[8], SIGNAL(clicked()), SLOT(setNum())); // 1
    connect(btn[9], SIGNAL(clicked()), SLOT(setNum())); // 2
    connect(btn[10], SIGNAL(clicked()), SLOT(setNum())); // 3
    connect(btn[11], SIGNAL(clicked()), SLOT(operation())); // *

    connect(btn[12], SIGNAL(clicked()), SLOT(setNum())); // 0
    connect(btn[13], SIGNAL(clicked()), SLOT(calculate())); // =
    connect(btn[14], SIGNAL(clicked()), SLOT(operation())); // +
    connect(btn[15], SIGNAL(clicked()), SLOT(operation())); // -

    QVBoxLayout *vBoxLayout = new QVBoxLayout(this);
    vBoxLayout->setMargin(6);
    vBoxLayout->addWidget(label);
    vBoxLayout->addLayout(gridLayout);
    setLayout(vBoxLayout);
}

Widget::~Widget()
{
    delete label;
}

void Widget::setNum()
{
    // 문자로 취급
    QString result = (label->text()=="0")?
                        ((QPushButton*)sender())->text()
                        :label->text() + ((QPushButton*)sender())->text();
    label->setText(result);
```

widget.h 파일

```
  // 숫자로 취급
    //label->setText(QString::number(label->text().toFloat()*10 + ₩
                              ((QPushButton*)sender())->text().toFloat()));
}



void Widget::operation()
{
    numberTemp = label->text();
    operate = ((QPushButton*)sender())->text();
    label->setText("0");
}

void Widget::calculate()
{
    float result;
    switch(operate.at(0).toLatin1()){
        case '+':
            result = numberTemp.toFloat() + label->text().toFloat();
            break;
        case '-':
            result = numberTemp.toFloat() - label->text().toFloat();
            break;
        case '*':
            result = numberTemp.toFloat() * label->text().toFloat();
            break;
        case '/':
            if(label->text().toFloat() > 0)
            {
                result = numberTemp.toFloat() / label->text().toFloat();
            }
            else
            {
                label->setText("Err : Cannot Divide by Zero");
                result = 0;
            }
```

widget.cpp 파일

```
            break;
    }
    label->setText(QString::number(result));
}

void Widget::clear()
{
    numberTemp.setNum(0);
    label->setText("0");
}
```



2. 계산기 2차 (buttongroup)

작업내용

1. 필요한 slot 추가

2. buttongroup 생성 및 버튼 위젯을 buttongroup에 추가

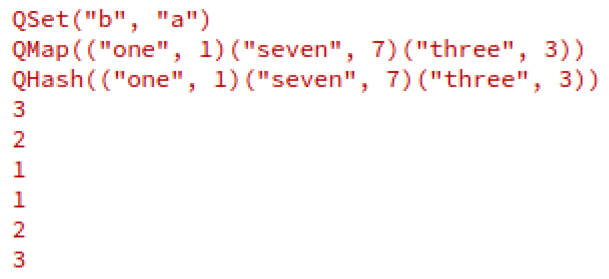
```
connect(buttonGroup, SIGNAL(buttonClicked(int)), SLOT(clickedGroup(int))); // 7
```

3. slot 구현

```
QPushButton* button = (QPushButton*)((QButtonGroup*)sender())->button(id);
```

widget.cpp 파일

```
void Widget::clickedGroup(int id)
{
    QPushButton* button = (QPushButton*)((QButtonGroup*)sender())->button(id);

    QString buttonText = button->text();

    switch (id) {
        case 0: case 1: case 2: // 7 8 9
        case 4: case 5: case 6: // 4 5 6
        case 8: case 9: case 10: // 1 2 3
        case 12:
            label->setText((label->text()=="0"?buttonText:label->text()+buttonText));
            break;
        case 3:
            clear();
            break;
        case 7: case 11: case 14: case 15:
            numberTemp = label->text();
            label->setText("0");
            operate = buttonText;
            break;
        case 13:
            calculate();
            break;

    }
}
```



```
QSet("b", "a")
QMap(("one", 1)("seven", 7)("three", 3))
QHash(("one", 1)("seven", 7)("three", 3))
3
2
1
1
2
3
```

3. DataType

widget.cpp 파일

```
// #1 QString
QString str = "12312.01";
QString str2 = "4";
qDebug() << str2.toInt();
qDebug() << str.toFloat();
qDebug("x=%.02f", str.toFloat());
qDebug("x=%.02f", str.toDouble());
qDebug() << QString("%1").arg(str.toShort());
qDebug() << QString("%1 %2").arg(str.toInt()).arg(str.toDouble());
qDebug() << "Types:" << QString("String") << QChar('x') << QRect(0, 10, 50, 40);

float floati = str.toFloat();
int inti = static_cast<int>(floati);
qDebug() << floati;
qDebug() << inti;


// #2 QByteArray
QByteArray ba;
ba.resize(5);
ba[0] = 0x3c;
ba[1] = 0xb8;
ba[2] = 0x64;
ba[3] = 0x18;
ba[4] = 0xca;
for (int i = 0; i < ba.size(); ++i) {
    if (ba.at(i) >= 'a' && ba.at(i) <= 'f')
        qDebug() << "Found character in range [a-f]" << "₩n" << i; // endl
}
```

widget.cpp 파일

```
// #3 QDataStream
QFile file("file.dat");
file.open(QIODevice::WriteOnly);
QDataStream out(&file);   // we will serialize the data into the file
out << QString("the answer is");   // serialize a string
out << (qint32)42;        // serialize an integer
file.close();

file.open(QIODevice::ReadOnly);
QDataStream in(&file);    // read the data serialized from the file
QString file_str;
qint32 a;
in >> file_str >> a;

qDebug() << file_str;
qDebug() << a;

// #4 QTextStream
QFile data("output.txt");
if (data.open(QFile::WriteOnly | QFile::Truncate)) {
    QTextStream out(&data);
    out << "Result: " << qSetFieldWidth(10) << left << 3.14 << 2.7 << " textstream";
}

// #5 QList
QList<QString> list = { "one", "two", "three" };
for (int i = 0; i < list.size(); ++i) {
    if (list.at(i) == "two")
        qDebug() << "Found two at position " << i;
}
```

widget.cpp 파일

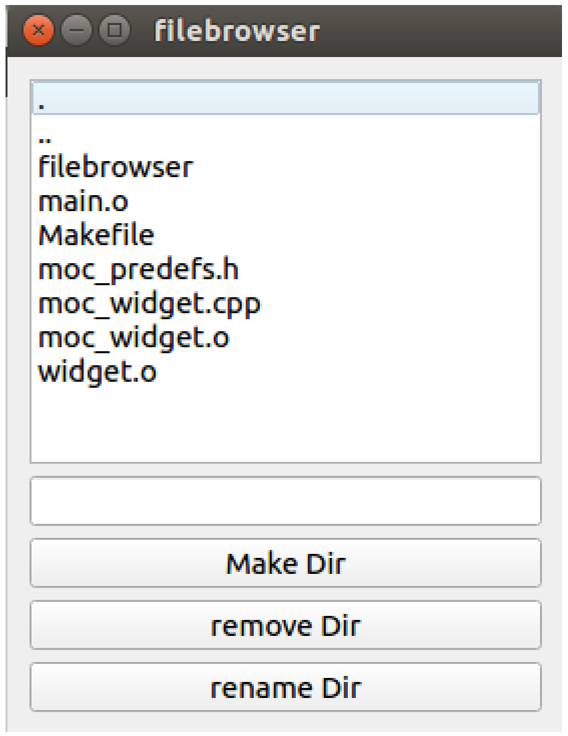
```
// #6 QLinkedList
QLinkedList<QString> linklist;
linklist << "one" << "two" << "three";
linklist.append("four");
while (!linklist.isEmpty())
    qDebug() << linklist.takeFirst();

// #7 QVector
QVector<int> vector(10);
int *vectordata = vector.data();
for (int i = 0; i < 10; ++i)
{
    vectordata[i] = 2 * i;
    qDebug() << vectordata[i];
}

// #8 QSet
QSet<QString> set;
set << "a" << "b" << "b";
qDebug() << set;

// #9 QMap
QMap<QString, int> map;
map["one"] = 1;
map["three"] = 3;
map["seven"] = 7;
qDebug() << map;

// #10 QHash
QHash<QString, int> hash;
hash["one"] = 1;
hash["three"] = 3;
hash["seven"] = 7;
qDebug() << hash;
```

widget.cpp 파일

```
// # 11 QStack
QStack<int> stack;
stack.push(1);
stack.push(2);
stack.push(3);
while (!stack.isEmpty())
 qDebug() << stack.pop();

// # 12 QStack
QQueue<int> queue;
queue.enqueue(1);
queue.enqueue(2);
queue.enqueue(3);
while (!queue.isEmpty())
    qDebug() << queue.dequeue();
```

-- 다음 예제

1. FileBrowser



```
.
..
filebrowser
main.o
Makefile
moc_predefs.h
moc_widget.cpp
moc_widget.o
widget.o
```

widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

class QDir;
class QListWidget;
class QLineEdit;

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

private:
    QDir *directory;
    QListWidget *dirListWidget;
    QLineEdit* filenameLineEdit;

    void refreshDir();

public slots:
    void selectItem();
    void changeDir();

    void makeDir();
    void removeDir();
    void renameDir();

};
#endif // WIDGET_H
```

widget.cpp 파일

```
#include "widget.h"

#include <QDir>
#include <QListWidget>
#include <QLineEdit>
#include <QFileInfo>
#include <QPushButton>
#include <QVBoxLayout>

Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    directory = new QDir(".");
    dirListWidget = new QListWidget(this);
    filenameLineEdit = new QLineEdit(this);

    QPushButton *makeDir = new QPushButton("Make Dir", this);
    QPushButton *removeDir = new QPushButton("remove Dir", this);
    QPushButton *renameDir = new QPushButton("rename Dir", this);

    QVBoxLayout *layout = new QVBoxLayout(this);
    layout->addWidget(dirListWidget);
    layout->addWidget(filenameLineEdit);
    layout->addWidget(makeDir);
    layout->addWidget(removeDir);
    layout->addWidget(renameDir);

    connect(dirListWidget, SIGNAL(itemClicked(QListWidgetItem*)),
SLOT(selectItem()));
    connect(dirListWidget, SIGNAL(itemDoubleClicked(QListWidgetItem*)),
SLOT(changeDir()));
    connect(makeDir, SIGNAL(clicked()), SLOT(makeDir()));
    connect(removeDir, SIGNAL(clicked()), SLOT(removeDir()));
    connect(renameDir, SIGNAL(clicked()), SLOT(renameDir()));

    refreshDir();
}
```

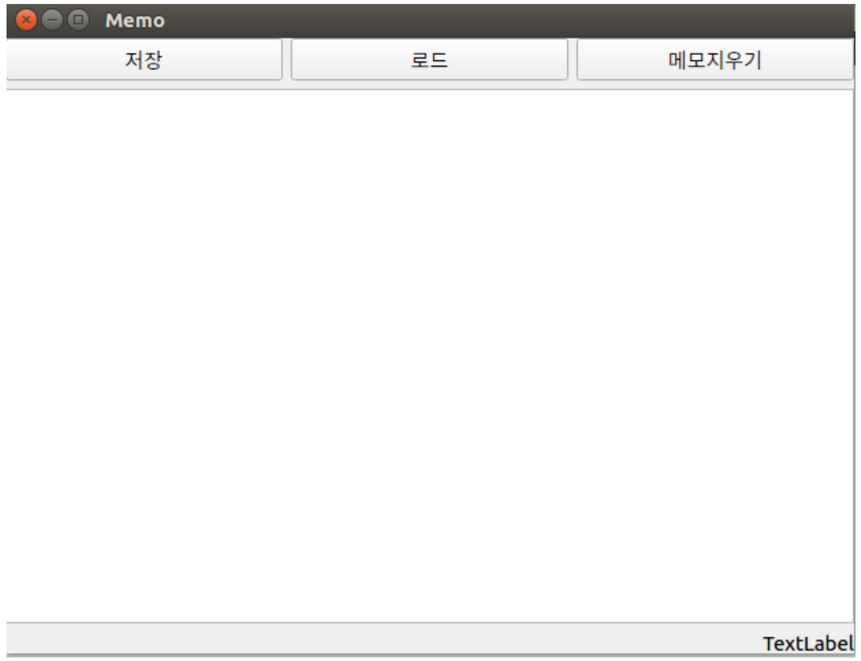
widget.cpp 파일

```
Widget::~Widget()
{
}

void Widget::refreshDir()
{
    dirListWidget->clear();
    for(int i=0; i<directory->entryList().count(); i++)
    {
        dirListWidget->addItem(directory->entryList().at(i));
    }
}

void Widget::selectItem()
{
    filenameLineEdit->setText(dirListWidget->currentItem()->text());
}

void Widget::changeDir()
{
    QFileInfo checkDir(dirListWidget->currentItem()->text());
    if(checkDir.isDir())
    {
        directory->cd(dirListWidget->currentItem()->text());
        refreshDir();
    }
}
```

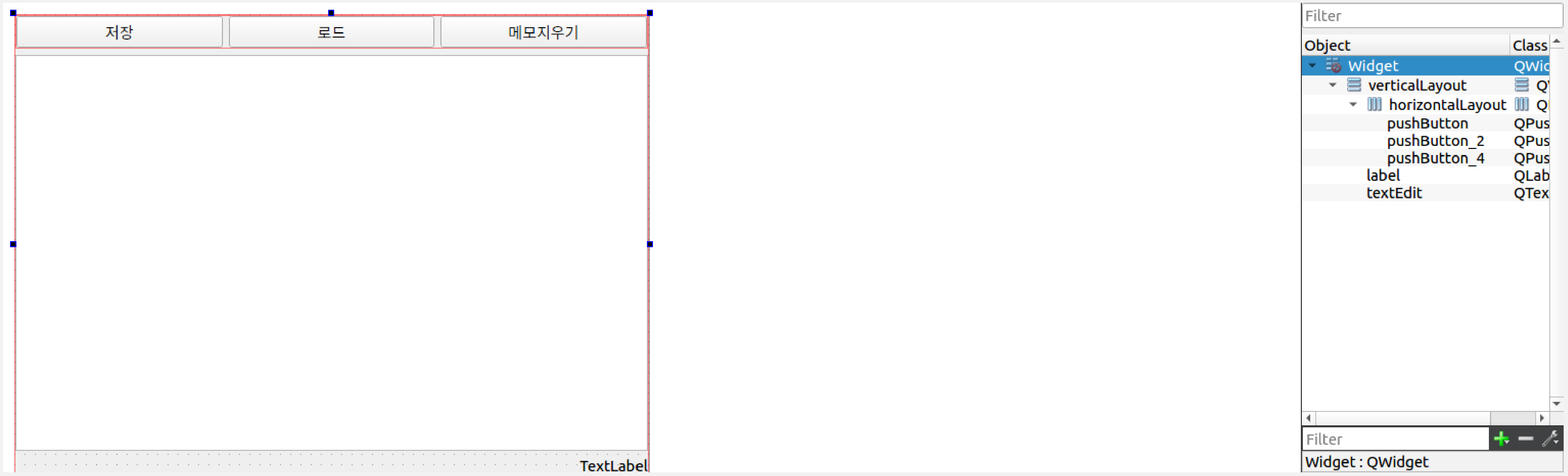
widget.cpp 파일

```
void Widget::makeDir()
{
    if(filenameLineEdit->text().length())
    {
        directory->mkdir(filenameLineEdit->text());
        directory->refresh();
        refreshDir();
    }
}

void Widget::removeDir()
{
    if(filenameLineEdit->text().length())
    {
        directory->rmdir(filenameLineEdit->text());
        directory->refresh();
        refreshDir();
    }
}

void Widget::renameDir()
{
    if(filenameLineEdit->text().length())
    {
        directory->rename(dirListWidget->currentItem()->text(), filenameLineEdit-
>text());
        directory->refresh();
        refreshDir();
    }
}
```

2. Memo



widget.ui 파일



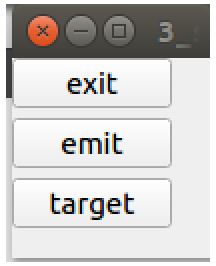
1. 시그널 + 슬롯 생성
   - 버튼 3개
   - textedit textchanged 슬롯

widget.cpp 파일

```
#include "widget.h"
#include "ui_widget.h"

#include <QFileDialog>
#include <QTextStream>

Widget::Widget(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Widget)
{
    ui->setupUi(this);
    this->setWindowTitle("Memo");
}

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


void Widget::on_pushButton_clicked()
{
    QString fileName = QFileDialog::getSaveFileName(this, tr("Save File"),
"./untitled.txt", tr("text(*.txt)"));

    QFile data(fileName);
    if (data.open(QFile::WriteOnly | QFile::Truncate)) {
        QTextStream out(&data);
        out << ui->textEdit->toPlainText();
    }
    data.close();

    this->setWindowTitle(QFileInfo(data).fileName());
}
```

widget.cpp 파일

```

void Widget::on_pushButton_2_clicked()
{
    QString fileName = QFileDialog::getOpenFileName(this, tr("Load File"),
"./untitled.txt", tr("text(*.txt)"));

    QFile data(fileName);
    if (data.open(QFile::ReadOnly)) {
        QTextStream in(&data);
        QString instr;
        in >> instr;
        ui->textEdit->setText(instr);
    }
    data.close();

    this->setWindowTitle(QFileInfo(data).fileName());
}

void Widget::on_pushButton_4_clicked()
{
    ui->textEdit->clear();
}

void Widget::on_textEdit_textChanged()
{
    ui->label->setText(QString("%1 %2").arg(QString::number((ui->textEdit-
>toPlainText().length()))).arg("count"));
}
```



3. Signal/Slot

widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

signals:
    void sigCustom(int);

public slots:
    void slotCustom(int);

    void slotExample();

    void emitSlot();
    void TargetSlot(int);
};
#endif // WIDGET_H
```

widget.cpp 파일

```
#include "widget.h"

#include <QDebug>
#include <QPushButton>
#include <QApplication>
#include <QAbstractButton>

int m_value = 0;
Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    this->resize(100,100);
    QPushButton *btn = new QPushButton("exit", this);

    // lambda
    connect(btn, &QPushButton::clicked, [btn] { ++m_value; btn-
>setText(QString::number(m_value)); });

    // 단순화 예시
    //QObject::connect(btn, SIGNAL(clicked()), this, SLOT(slotExample()));
    //->connect(btn, SIGNAL(clicked()), this, SLOT(slotExample()));
    //->->connect(btn, SIGNAL(clicked()), SLOT(slotExample()));

    // emit 예시
    QPushButton *btn1 = new QPushButton("emit", this);
    QPushButton *btn2 = new QPushButton("target", this);
    btn1->move(0, 30);
    btn2->move(0, 60);
    connect(btn1, SIGNAL(clicked()), this, SLOT(emitSlot()));
    connect(btn2, SIGNAL(clicked()), this, SLOT(TargetSlot(int))); // 작동 안함
    connect(this, SIGNAL(sigCustom(int)), this, SLOT(TargetSlot(int)));
    //connect(btn2, &QPushButton::clicked, qApp, QApplication::quit); // static 만 사용
가능
}
```

widget.cpp 파일

```
Widget::~Widget()
{
}

void Widget::slotCustom(int i)
{
    qDebug() << i;
}

void Widget::slotExample()
{
    qDebug() << "Empty Function";
}

void Widget::emitSlot()
{
    qDebug() << "CALL emitSlot";
    emit TargetSlot(100);
    emit sigCustom(101);
}

void Widget::TargetSlot(int i)
{
    qDebug() << "CALL TargetSlot";
    qDebug() << i;
}
```

4. Event Filter

widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

class QLabel;
class QTextEdit;

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

public:
    QLabel* label;

    QTextEdit *edit;

protected:
    void moveEvent(QMoveEvent*);

    bool eventFilter(QObject*, QEvent*);
};
#endif // WIDGET_H
```

widget.cpp 파일

```
#include "widget.h"

#include <QLabel>

#include "mytextedit.h"

#include <QTextEdit>
#include <QEvent>
#include <QKeyEvent>

Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    resize(100,250);
    label = new QLabel(this);
    label->setText("Power ON");
    label->resize(100,30);

    // 입력제한
    myTextEdit *customwidget = new myTextEdit(this);
    customwidget->resize(100,100);
    customwidget->move(0,40);



    // 입력제한 event-filter
    edit = new QTextEdit(this);
    edit->resize(100,100);
    edit->move(0,150);
    edit->installEventFilter(this);
}
```

widget.h 파일

```
Widget::~Widget()
{

}

void Widget::moveEvent(QMoveEvent*)
{
    label->setText(QString("X:%1, Y:%2").arg(pos().x()).arg(pos().y()));
}


bool Widget::eventFilter(QObject *target, QEvent *event)
{
    if(target == edit)
    {
        qDebug("in eventFilter edit");
        if(event->type() == QEvent::KeyPress)
        {
            if((static_cast<QKeyEvent*>(event))->key() == Qt::Key_1)
            {
                qDebug("pushed Key 1");
                return true;
            }
        }
    }
    return QWidget::eventFilter(target, event);
}
```

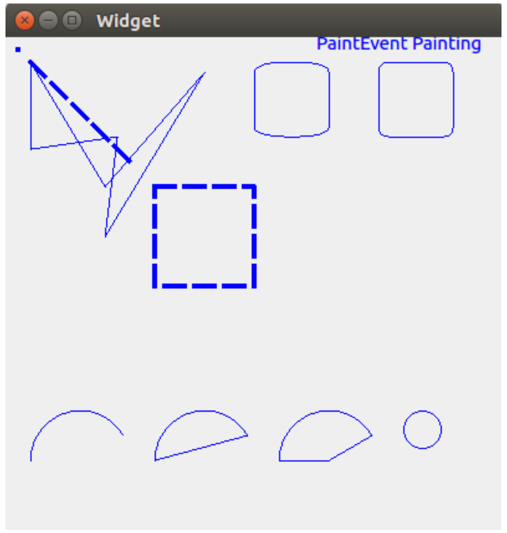
mytextedit.h 파일

```
#ifndef MYTEXTEDIT_H
#define MYTEXTEDIT_H

#include <QWidget>
#include <QTextEdit>

class myTextEdit : public QTextEdit
{
    Q_OBJECT
public:
    myTextEdit(QWidget *parent = nullptr);

protected:
    void keyPressEvent(QKeyEvent*);
};

#endif // MYTEXTEDIT_H
```

mytextedit.cpp 파일

```
#include "mytextedit.h"

#include <QTextEdit>

#include <QKeyEvent>

myTextEdit::myTextEdit(QWidget *parent) : QTextEdit(parent)
{
    installEventFilter(this);
}


void myTextEdit::keyPressEvent(QKeyEvent *e)
{
    qDebug("keyPressEvent(%x)", e->key());
    switch(e->key())
    {
        case Qt::Key_1:
            qDebug("pushed Key 1");
            return;
            break;
    }
    return QTextEdit::keyPressEvent(e);
}
```

5. Painter



widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

QT_BEGIN_NAMESPACE
namespace Ui { class Widget; }
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

private:
    Ui::Widget *ui;

protected:
    void paintEvent(QPaintEvent *event);
    void paint1();
    void paint2();
    void paint3();
};
#endif // WIDGET_H
```

widget.cpp 파일

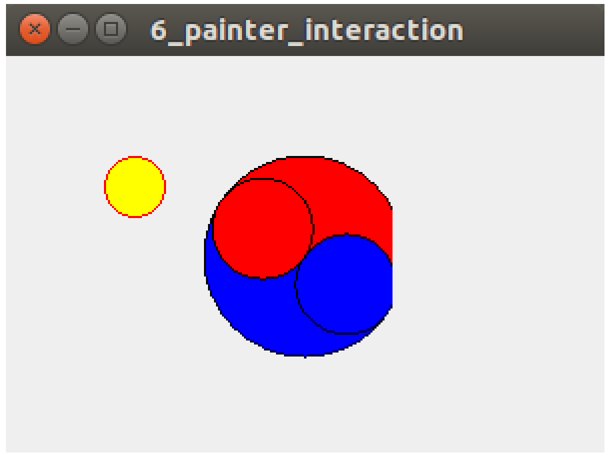
```
#include "widget.h"
#include "ui_widget.h"

#include <QPainter>

Widget::Widget(QWidget *parent)
    : QWidget(parent)
    , ui(new Ui::Widget)
{
    ui->setupUi(this);
    resize(400, 400);
}

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

void Widget::paintEvent(QPaintEvent *event)
{
    //paint1();
    //paint2();
    paint3();
}

void Widget::paint1()
{
    QPainter *painter = new QPainter(this);

    painter->setPen(QPen(Qt::blue, 4, Qt::DashLine));
    painter->drawPoint(10, 10);


    painter->drawLine(20, 20, 100, 100);

    painter->drawRect(120, 120, 80, 80);
```

widget. cpp 파일

```
    painter->setPen(QPen(Qt::blue, 1, Qt::SolidLine));

    painter->drawRoundRect(200, 20, 60, 60, 80);
    painter->drawRoundRect(300, 20, 60, 60);

    painter->drawArc(20, 300, 80, 80, 30 * 16, 150 * 16);
    painter->drawChord(120, 300, 80, 80, 30 * 16, 150 * 16);
    painter->drawPie(220, 300, 80, 80, 30 * 16, 150 * 16);

    painter->drawEllipse(320, 300, 30, 30);

    static const QPoint points[6] = {
        QPoint(20, 20),
        QPoint(20, 90),
        QPoint(90, 80),
        QPoint(80, 160),
        QPoint(160, 28),
        QPoint(80, 120),
    };

    painter->drawPolygon(points, 6);


    painter->drawText(250, 10, "PaintEvent Painting");
    delete painter;
}

void Widget::paint2()
{
    QPainterPath path;
    path.addRect(20, 20, 60, 60);

    path.moveTo(0, 0);
    path.cubicTo(99, 0, 50, 50, 99, 99);
    path.cubicTo(0, 99, 50, 50, 0, 0);
```

widget.cpp 파일

```
    QPainter *painter = new QPainter(this);

    painter->fillRect(0, 0, 100, 100, Qt::white);

    painter->setPen(QPen(QColor(79, 106, 25), 2, Qt::SolidLine, Qt::FlatCap,
Qt::MiterJoin));
    painter->setBrush(QColor(122,163,39));

    painter->drawPath(path);

    delete painter;
}


void Widget::paint3()
{
    QRectF target(10.0, 20.0, 80.0, 60.0);
    QRectF source(0.0, 0.0, 70.0, 40.0);
    QPixmap pixmap("../../../image/icon1.png");
    QPixmap pixmap2("../../../image/icon2.png");

    QPainter *painter = new QPainter(this);

    painter->drawPixmap(10, 10, pixmap2.width(), pixmap2.height(), pixmap2);
    painter->drawPixmap(target, pixmap, source);

    QImage img("../../../image/Cluster1.png");
    painter->drawImage(100,100, img, 20, 20, 100, 100);
    delete painter;
}
```



6. Painter Interaction

widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

    int clipWidth;

protected:
    void paintEvent(QPaintEvent *event);
    void resizeEvent(QResizeEvent *event);

    void timerEvent(QTimerEvent*);

    void mouseMoveEvent(QMouseEvent *event);
};
#endif // WIDGET_H
```

widget.cpp 파일

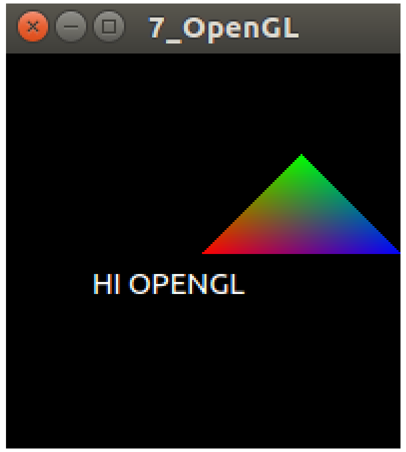
```
#include "widget.h"

#include <QPainter>
#include <QMouseEvent>

#define DEGREE 56.31

int iCircleSize = 1;

Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    resize(300, 300);

    clipWidth = 0;

    startTimer(20);

    setMouseTracking(true);
}

Widget::~Widget()
{
}

void Widget::timerEvent(QTimerEvent* event)
{
    if(clipWidth < 5000)
    {
        clipWidth++;
        update();
    }
}
```

widget.cpp 파일

```
// 태극무늬 각도 56.31
void Widget::paintEvent(QPaintEvent *event)
{
    QRect clipRect(0, 0, clipWidth%width()+1, height());

    QPainter painter(this);

    painter.setPen(Qt::red);
    painter.setBrush(Qt::yellow);
    painter.drawEllipse(50,50, 30 * iCircleSize, 30 * iCircleSize);

    painter.setClipping(true);
    painter.setClipRect(clipRect);

    painter.setPen(Qt::black);

    qreal diagonalLength = sqrt(pow(width(), 2) + pow(height(), 2));
    qint16 radius = width()/3.;
    qint16 xPoint = (width()-radius)/2., yPoint = (height()-radius)/2.;

    QPainterPath pathBigRed;
    pathBigRed.arcMoveTo(xPoint, yPoint, radius, radius, 180-DEGREE/2);
    pathBigRed.arcTo(xPoint, yPoint, radius, radius, -DEGREE/2, 180);
    painter.setBrush(Qt::red);
    painter.drawPath(pathBigRed);

    QPainterPath pathBigBlue;
    pathBigBlue.arcMoveTo(xPoint, yPoint, radius, radius, 180-DEGREE/2);
    pathBigBlue.arcTo(xPoint, yPoint, radius, radius, 180-DEGREE/2, 180);
    painter.setBrush(Qt::blue);
    painter.drawPath(pathBigBlue);
```

widget.cpp 파일

```
    painter.setBrush(Qt::red);
    painter.resetMatrix();
    painter.translate(width()/2, height()/2);
    painter.rotate(-DEGREE);
    painter.drawEllipse(-radius/4., -radius/2, radius/2., radius/2.);

    painter.setBrush(Qt::blue);
    painter.resetMatrix();
    painter.translate(width()/2, height()/2);
    painter.rotate(-DEGREE);
    painter.drawEllipse(-radius/4., 0, radius/2., radius/2.);
}

void Widget::resizeEvent(QResizeEvent *event)
{
    resize(this->size().width(), this->size().width()*2/3);
}


void Widget::mouseMoveEvent(QMouseEvent *event)
{
    QPoint position = event->pos();

    if(position.x() >= 50 && position.x() <= 100)
    {
        iCircleSize = 2;
    }
    else
    {
        iCircleSize = 1;
    }
}
```



HI OPENGL

7. OpenGL

.pro 파일

```
QT       += core gui opengl
```

widget.h 파일

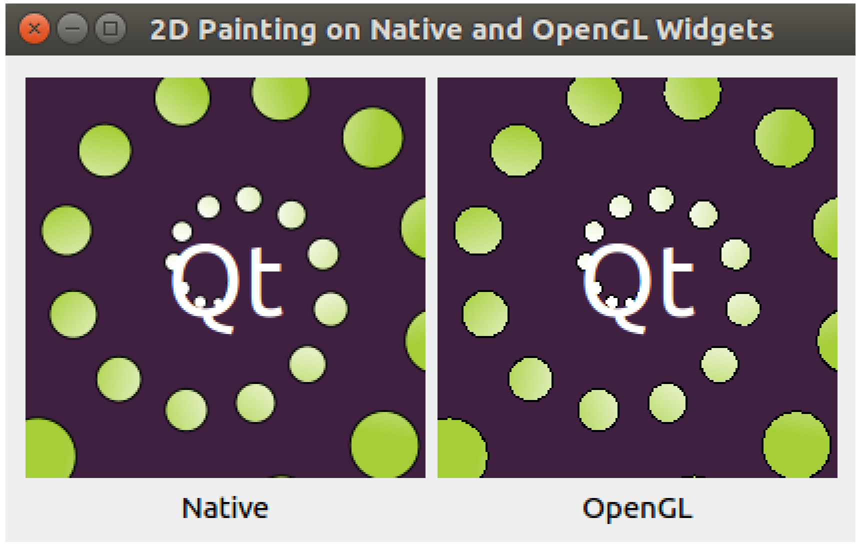
```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

#include <QtOpenGL>

class Widget : public QOpenGLWidget, protected QOpenGLFunctions
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

protected:
    void initializeGL();
    void paintGL();
    void resizeGL(int w, int h);
};
#endif // WIDGET_H
```

widget.cpp 파일

```
#include "widget.h"

Widget::Widget(QWidget *parent)
    : QOpenGLWidget(parent)
{
    resize(200, 200);
}

Widget::~Widget()
{
}

void Widget::initializeGL()
{
    initializeOpenGLFunctions();
    glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
}

void Widget::paintGL()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glBegin(GL_TRIANGLES);
        glColor3f(1.f, 0.f, 0.f);
        glVertex2f(0.0f, 0.0);
        glColor3f(0.f, 1.f, 0.f);
        glVertex2f(0.5f, 0.5);
        glColor3f(0.f, 0.f, 1.f);
        glVertex2f(1.0f, 0.0);
    glEnd();

    QPainter painter(this);
    painter.setPen(Qt::white);
    painter.setRenderHints(QPainter::Antialiasing | QPainter::TextAntialiasing);
    painter.drawText(45, 120, "HI OPENGL");
    painter.end();

    glFlush();
```

widget.cpp 파일

```
}


void Widget::resizeGL(int w, int h)
{
    qDebug("W: %d, H: %d₩n", w, h);
    glViewport(0, 0, (GLint)w/2, (GLint)h/2);
    glLoadIdentity();
    glOrtho(0, w, 0, h, -1, 1);

    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}
```

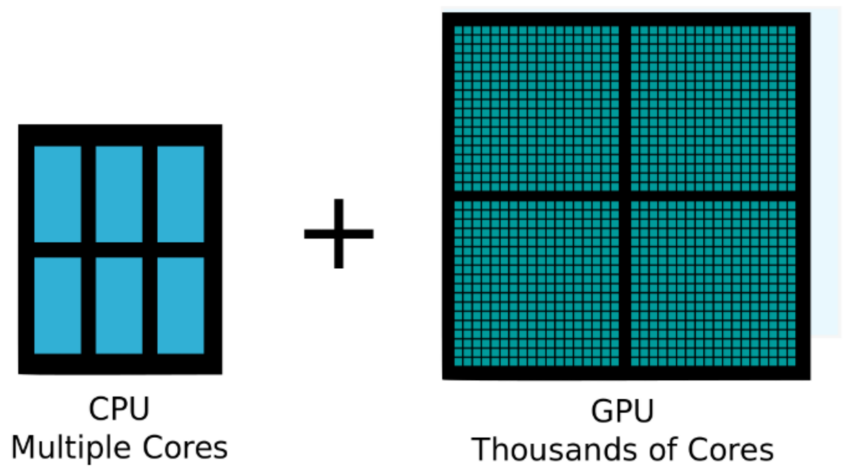


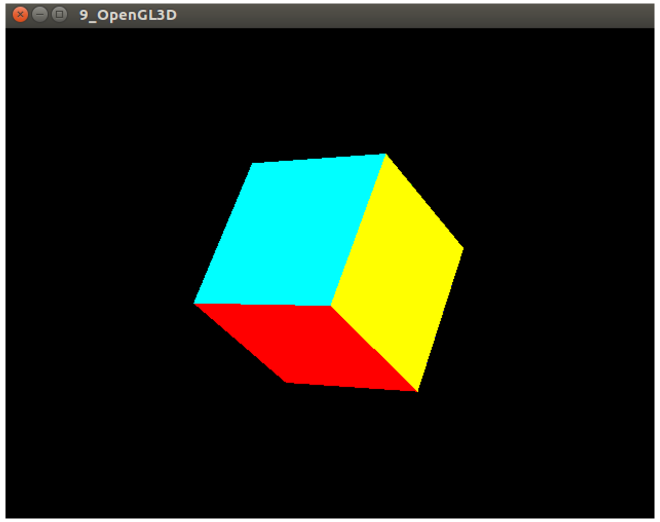
8. OpenGL Painting

위키 백과 발췌 –

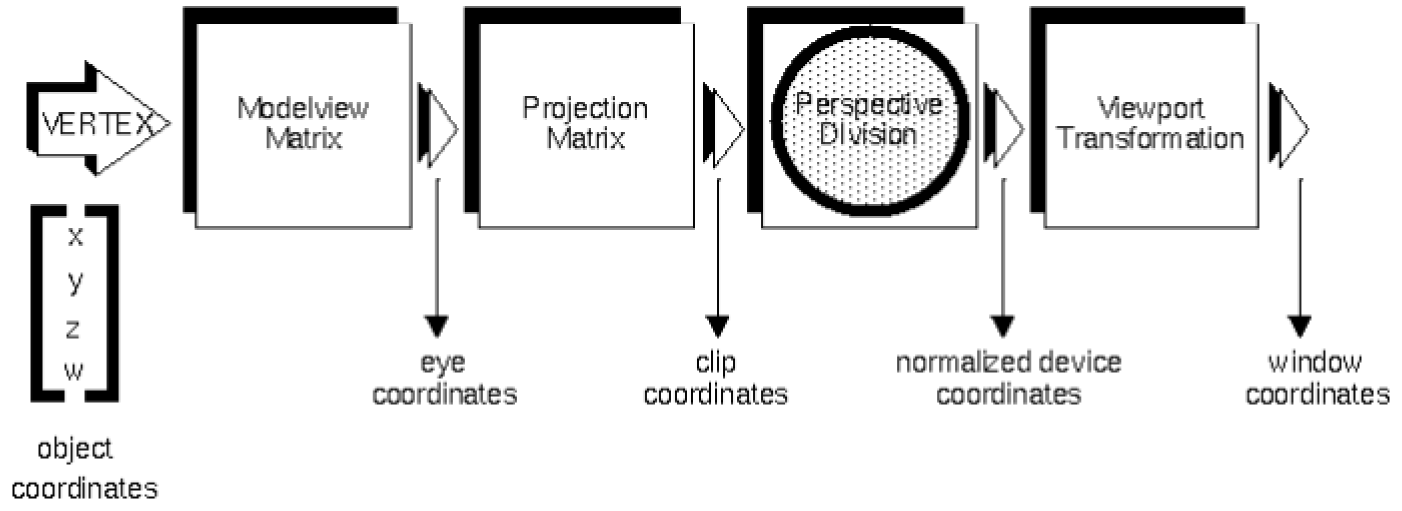
CPU로부터 별도로 가속을 수행하는 하드웨어를 **하드웨어 가속 장치**, 구체적으로 말해 그래픽 처리 장치, 부동 소수점 장치라고 한다. 다만 이러한 용어들은 오래 되어 비디오 카드, 그래픽 카드와 같은 용어로 치환되어 왔다.

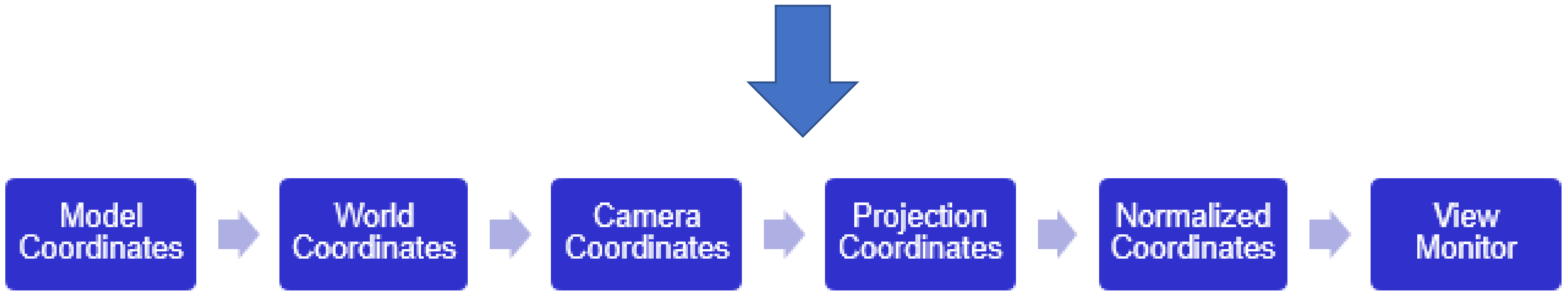
하드웨어 가속(Hardware acceleration)은 컴퓨팅에서 일부 기능을 CPU에서 구동하는 소프트웨어 방식보다 더 빠르게 수행할 수 있는 하드웨어의 사용을 말한다. 하드웨어 가속은 이를테면, 그래픽 처리 장치 (GPU)의 블리팅 가속 기능과 CPU의 복잡한 기능에 대한 함수가 있다.



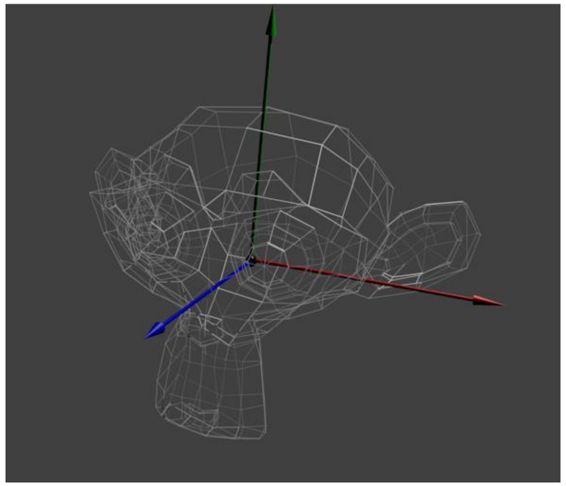


9. OpenGL 3D

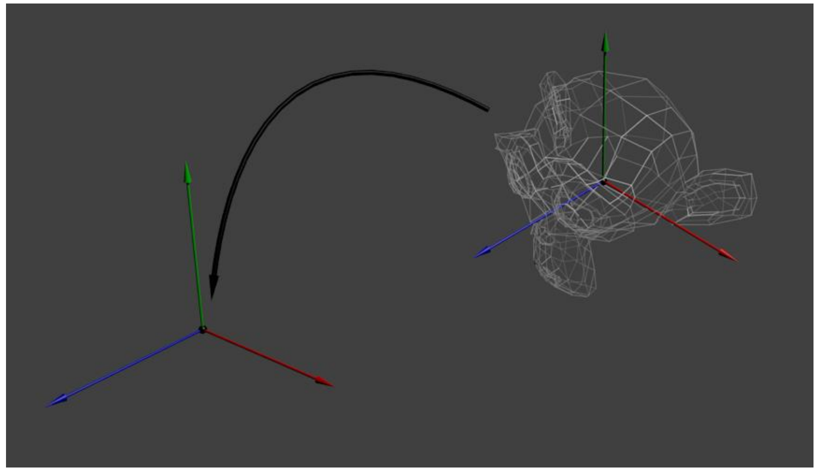

VERTEX
x
y
z
w
object coordinates
Modelview Matrix
eye coordinates
Projection Matrix
clip coordinates
Perspective Division
normalized device coordinates
Viewport Transformation
window coordinates


Model Coordinates
World Coordinates
Camera Coordinates
Projection Coordinates
Normalized Coordinates
View Monitor

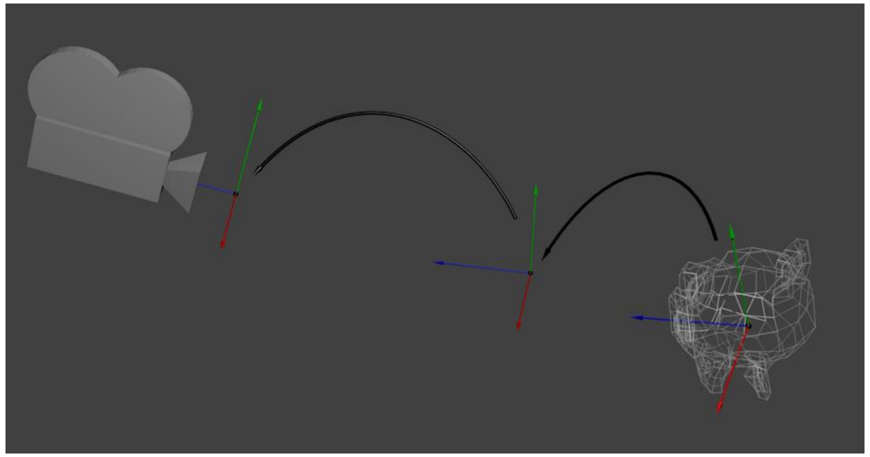
1.Model matrix



2.World matrix

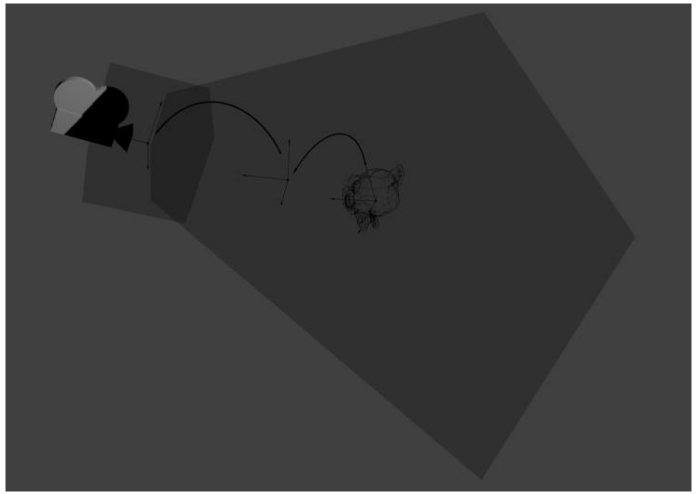


3.View matrix

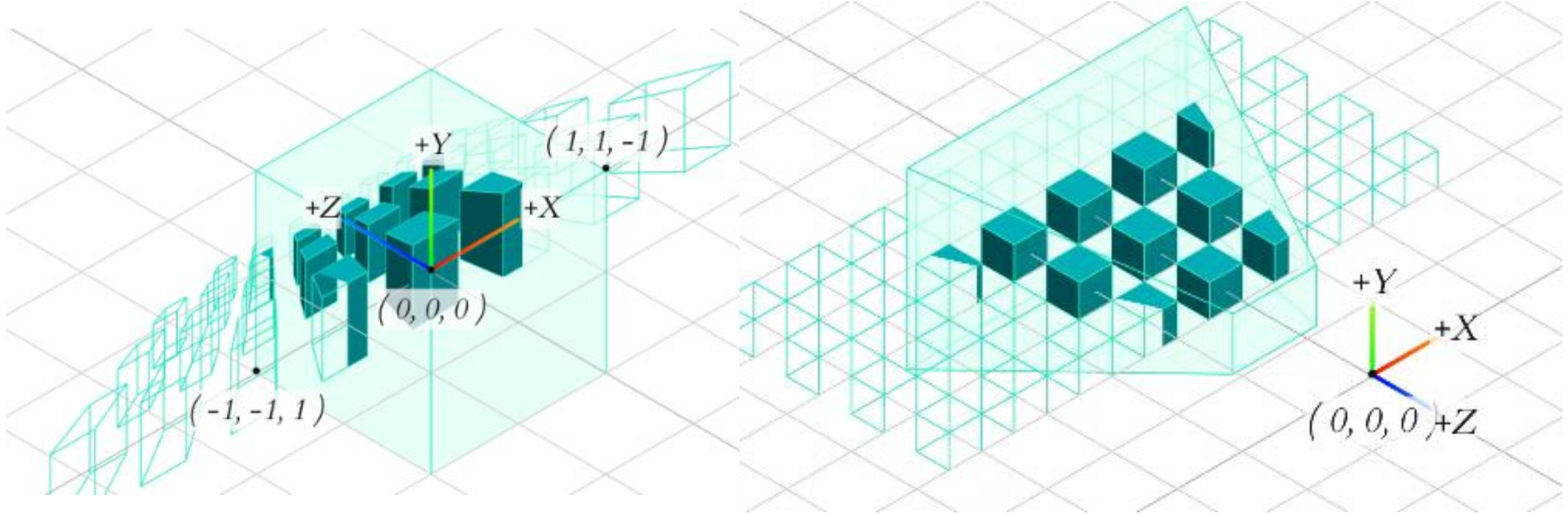


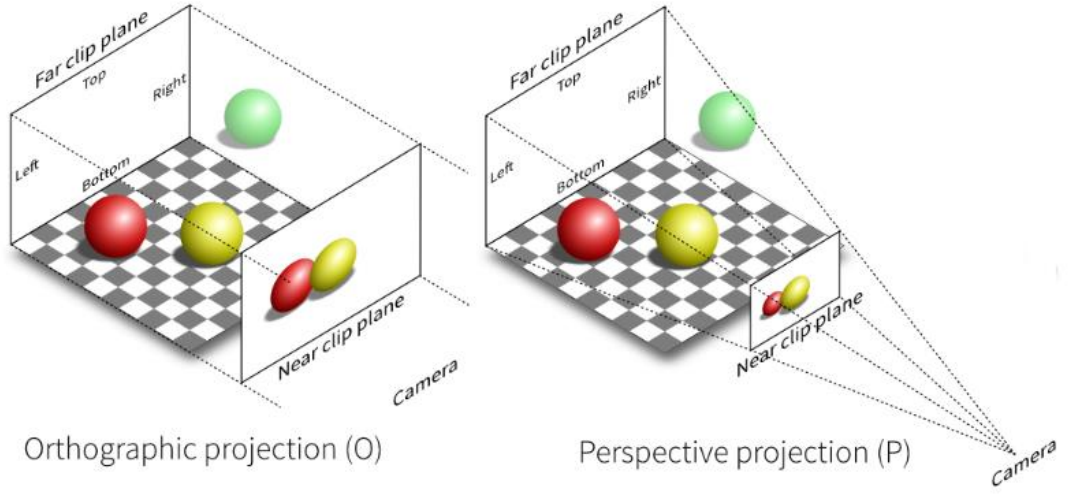
$$\begin{tikzcd}\boxed{\text{Model Coordinates}} \arrow[r] & \boxed{\text{World Coordinates}} \arrow[r] & \boxed{\text{Camera Coordinates}}\end{tikzcd}$$

4.Projection Matrix

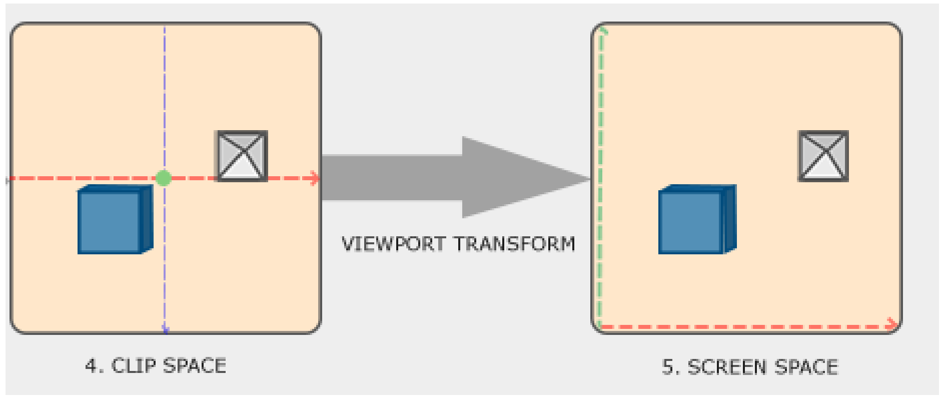


5.View Monitor

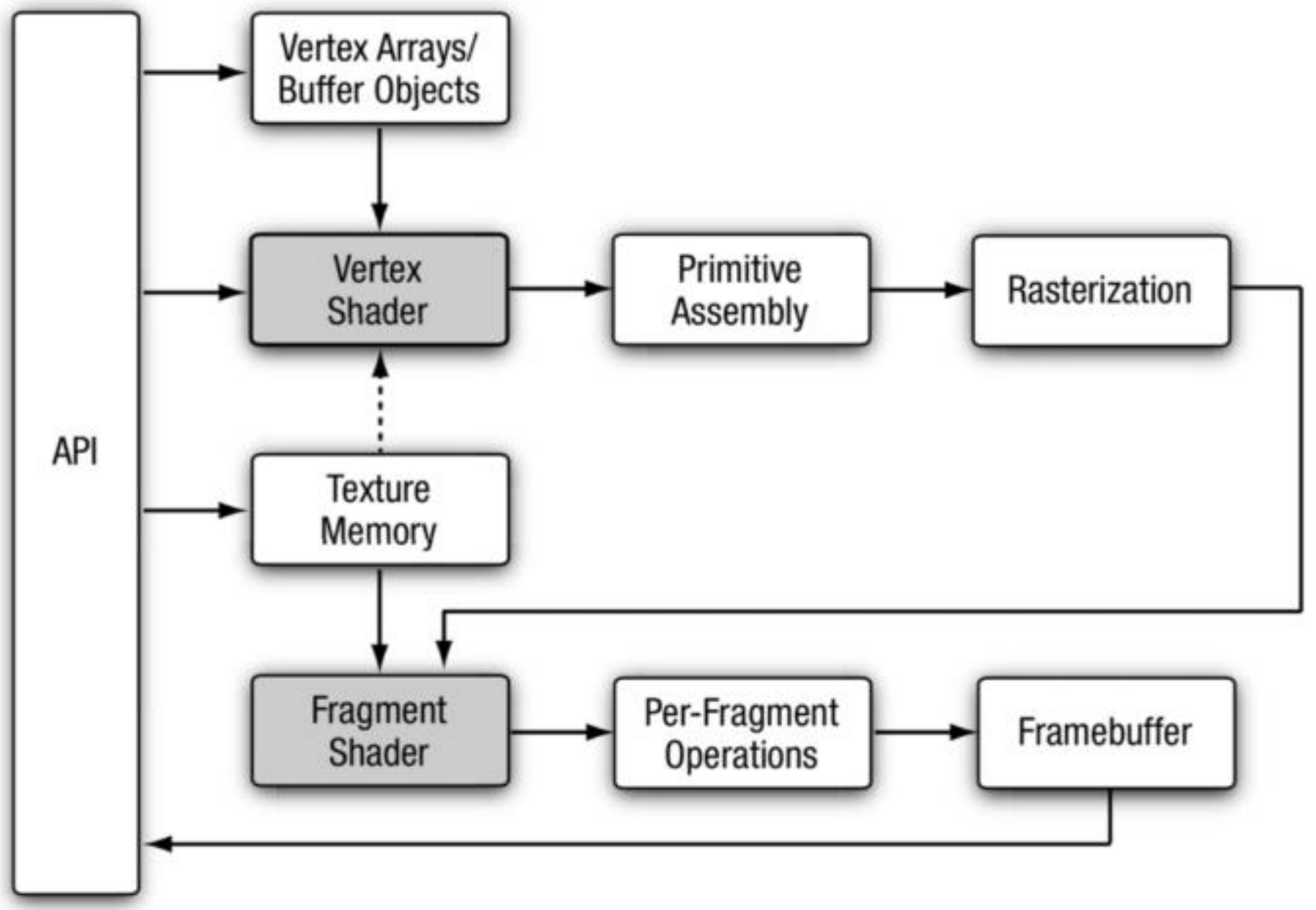


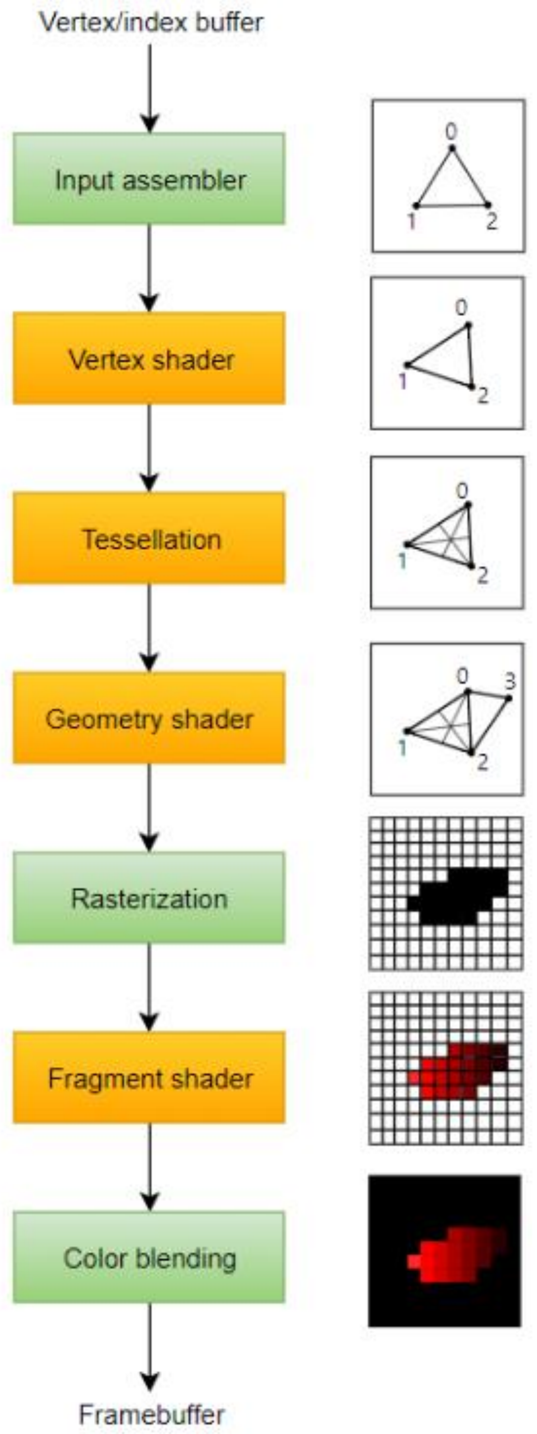


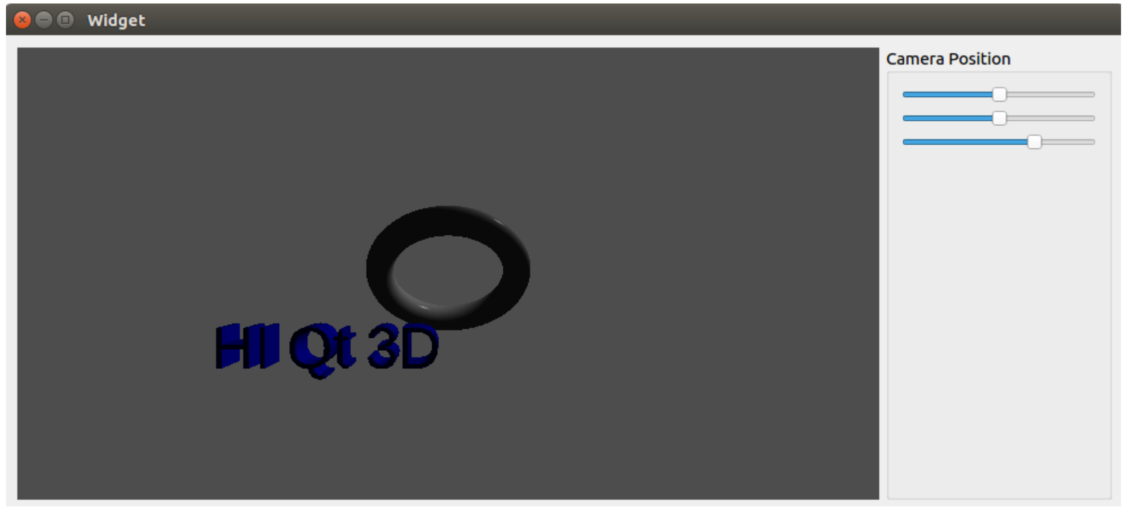
5.View Monitor



Graphics Pipeline







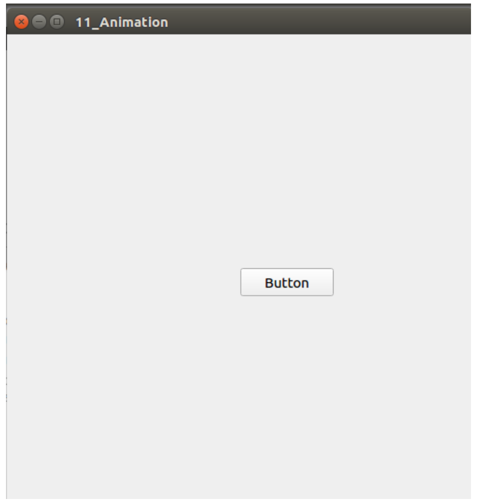
10. Qt3D

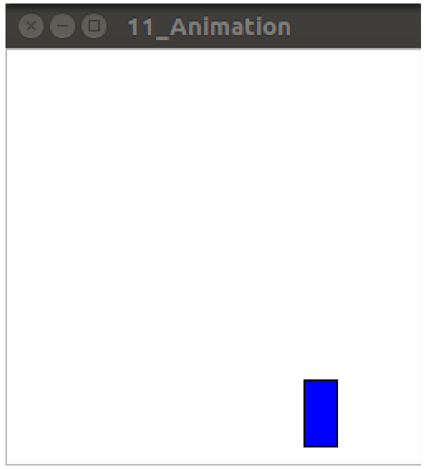
- Qt3DExtras::Qt3DWindow *m_view;

- Qt3DRender::QCamera *m_camera;

- Qt3DCore::QEntity *m_rootEntity;
  Qt3DRender::QMaterial* material
  Qt3DExtras::QTorusMesh* mesh

- Qt3DCore::QEntity *m_textEntity;





11. Animation

widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QPushButton>
#include <QPropertyAnimation>

class Widget : public QWidget
{
    Q_OBJECT
public:
    Widget(QWidget *parent = 0);
    ~Widget();

private:
    QPropertyAnimation *animation;

public slots:
    void btnClicked();

};

#endif // WIDGET_H
```

widget.cpp 파일

```
#include "widget.h"

#include <QPropertyAnimation>

#include <QState>
#include <QSignalTransition>
#include <QStateMachine>
#include <QSignalTransition>

Widget::Widget(QWidget *parent) : QWidget(parent)
{
    this->resize(500, 500);

    QPushButton *btn = new QPushButton("Button", this);
    connect(btn, &QPushButton::pressed,
            this, &Widget::btnClicked);
    btn->setGeometry(10, 10, 100, 30);

    animation = new QPropertyAnimation(btn, "geometry", this);

    animation->setDuration(3000); // 3초(단위 밀리세컨드)
    animation->setStartValue(QRect(10, 10, 100, 30)); // 시작 좌표
    animation->setEndValue(QRect(200, 150, 100, 30)); // 끝나는 좌표

    animation->setEasingCurve(QEasingCurve::OutInQuart);


    QStateMachine *machine = new QStateMachine;

    QState *state1 = new QState(machine); // state-1
    state1->assignProperty(btn, "geometry", QRect(10, 10, 100, 30));
    machine->setInitialState(state1);

    QState *state2 = new QState(machine); // state-2
    state2->assignProperty(btn, "geometry", QRect(250, 250, 100, 30));
```

widget.cpp 파일

```
    QSignalTransition *transition1  = state1->addTransition(btn,
                                        SIGNAL(clicked()), state2); // transition-1
    transition1->addAnimation(new QPropertyAnimation(btn, "geometry"));

    QSignalTransition *transition2  = state2->addTransition(btn,
                                        SIGNAL(clicked()), state1); // transition-2
    transition2->addAnimation(new QPropertyAnimation(btn, "geometry"));

    machine->start();
}

void Widget::btnClicked()
{
    //animation->start();
}

Widget::~Widget()
{
}
```

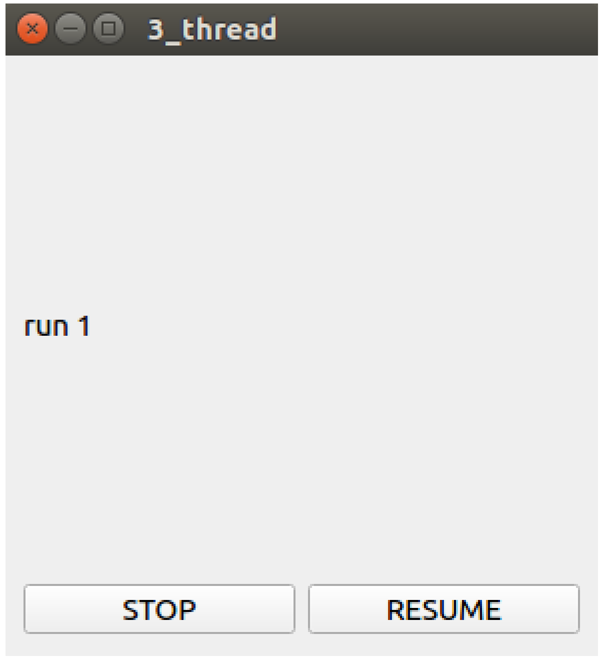
main.cpp 파일

```
#include "widget.h"

#include <QApplication>

#include <QtWidgets>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    Widget w;
    w.show();


    QGraphicsRectItem *rect = new QGraphicsRectItem(0, 0, 40, 20);
    rect->setBrush(QBrush(Qt::blue));

    QTimeLine *timer = new QTimeLine(5000);
    timer->setFrameRange(0, 100);

    QGraphicsItemAnimation *animation = new QGraphicsItemAnimation;
    animation->setItem(rect);
    animation->setTimeLine(timer);

    for(int i=0; i<200; ++i)
    {
        animation->setPosAt(i/200.0, QPointF(i,i));
    }
    animation->setRotationAt(80.0/200.0, 30);
    animation->setRotationAt(180.0/200.0, 90);

    QGraphicsScene *scene = new QGraphicsScene();
    scene->setSceneRect(0,0,250,250);
    scene->addItem(rect);
```

main.cpp 파일

```
    QGraphicsView *view = new QGraphicsView(scene);
    view->show();
    timer->start();

    return a.exec();
}
```

-- 다음 예제



3. thread

.pro 파일

```
QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

CONFIG += c++11
CONFIG += thread

DEFINES += QT_DEPRECATED_WARNINGS

SOURCES += ₩
    main.cpp ₩
    thread.cpp ₩
    widget.cpp

HEADERS += ₩
    thread.h ₩
    widget.h

# Default rules for deployment.
qnx: target.path = /tmp/$${TARGET}/bin
else: unix:!android: target.path = /opt/$${TARGET}/bin
!isEmpty(target.path): INSTALLS += target
```

thread.h 파일

```
#ifndef THREAD_H
#define THREAD_H

#include <QThread>
#include <QWaitCondition>
#include <QMutex>
#include <QLabel>

class Thread : public QThread
{
    Q_OBJECT

public:
    Thread(QObject* obj = nullptr);


public:
    enum {Stop = 0, Play};

private:
    QLabel* label;
    QWaitCondition waitcond;
    QMutex mutex;
    qint32 stopFlag;

protected:
    void run();

signals:
    void setLabeled(QString);

public slots:
    void stopThread();
    void resumeThread();
};

#endif // THREAD_H
```

thread.cpp 파일

```
#include "thread.h"

Thread::Thread(QObject* obj)
{
    label = (QLabel*)obj;
    stopFlag = Play;
}

void Thread::run()
{
    for(int count = 0;;) // while(true)
    {
        mutex.lock();
        if(stopFlag == Stop)
        {
            waitcond.wait(&mutex);
        }

        mutex.unlock();
        emit setLabeled(QString("run %1").arg(count++));
        sleep(1);
    }
}

void Thread::stopThread()
{
    stopFlag = Stop;
}

void Thread::resumeThread()
{
    mutex.lock();
    stopFlag = Play;
    waitcond.wakeAll();
    mutex.unlock();
}
```

widget.h 파일

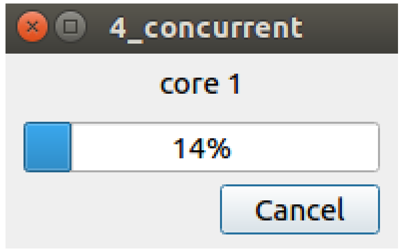
```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>
#include "thread.h"

#include <QThreadPool>
#include <QRunnable>
#include <QDebug>

class RunnableTask : public QRunnable
{
    void run()
    {
        qDebug() << "Runnable Thread" << QThread::currentThreadId();
    }

};

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

    Thread* th;
};
#endif // WIDGET_H
```

widget.cpp 파일

```
#include "widget.h"

#include <QLabel>
#include <QPushButton>
#include <QHBoxLayout>
#include <QVBoxLayout>
#include "thread.h"

Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    resize(300, 300);
    QLabel* label = new QLabel(this);
    label->resize(100,100);
    th = new Thread(label);

    QPushButton* stopbtn = new QPushButton("STOP", this);
    QPushButton* resumebtn = new QPushButton("RESUME", this);

    QHBoxLayout* hlayout = new QHBoxLayout();
    hlayout->addWidget(stopbtn);
    hlayout->addWidget(resumebtn);

    QVBoxLayout* vlayout = new QVBoxLayout();
    vlayout->addWidget(label);
    vlayout->addLayout(hlayout);

    connect(stopbtn, SIGNAL(clicked()), th, SLOT(stopThread()));
    connect(resumebtn, SIGNAL(clicked()), th, SLOT(resumeThread()));
    connect(th, SIGNAL(setLabeled(QString)), label, SLOT(setText(QString)));

    setLayout(vlayout);

    th->start();
```

widget.cpp 파일

```
    RunnableTask *runTh = new RunnableTask();
    QThreadPool::globalInstance()->start(runTh);
}

Widget::~Widget()
{
    th->terminate();
}
```



4. concurrent

.pro 파일

```
QT       += core gui concurrent
```

widget.cpp 파일

```
#include "widget.h"

#include <QtConcurrent>
#include <QFutureWatcher>
#include <QFuture>
#include <QThread>
#include <QDebug>

#include <QProgressDialog>

const int iter = 50;

static void spin(int &iter)
{
    const int work = 1000 * 1000 * 40;
    volatile int v = 0;
    for (int j = 0; j < work; j++)
    {
        ++v;
    }
    qDebug() << "iter" << iter << "in thread" << QThread::currentThreadId()
<< QThread::currentThread();

}

static void display(const QString &msg)
{
    qDebug() << __func__ << QThread::currentThreadId() << msg;
}
```

widget.cpp 파일

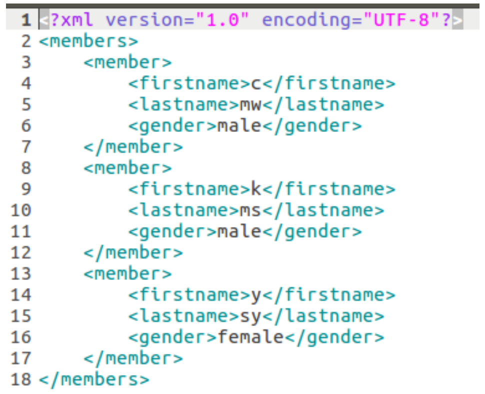
```
Widget::Widget(QWidget *parent)
    : QWidget(parent)
{
    QThreadPool::globalInstance()->setMaxThreadCount(3);

    qDebug() << "Widget Thread" << QThread::currentThreadId();

    QVector<int> vector;
    for(int i = 0; i<iter; ++i)
    {
        vector.append(i);
    }

    QProgressDialog dialog;
    dialog.setLabelText(QString("core %1").arg(QThread::idealThreadCount()));

    QFutureWatcher<void> watcher;

    connect(&watcher, SIGNAL(finished()), &dialog, SLOT(reset()));
    connect(&watcher, SIGNAL(progressRangeChanged(int, int)), &dialog,
SLOT(setRange(int, int)));
    connect(&watcher, SIGNAL(progressValueChanged(int)), &dialog,
SLOT(setValue(int)));
    connect(&dialog, SIGNAL(canceled()), &watcher, SLOT(cancel()));

    watcher.setFuture(QtConcurrent::map(vector, spin));
    dialog.exec();

    QFuture<void> future1 = QtConcurrent::run(display, QObject::tr("HI Concurrent"));
    watcher.setFuture(future1);

    watcher.waitForFinished();

    qDebug() << "Canceled?" << watcher.future().isCanceled();
}
```

widget.cpp 파일

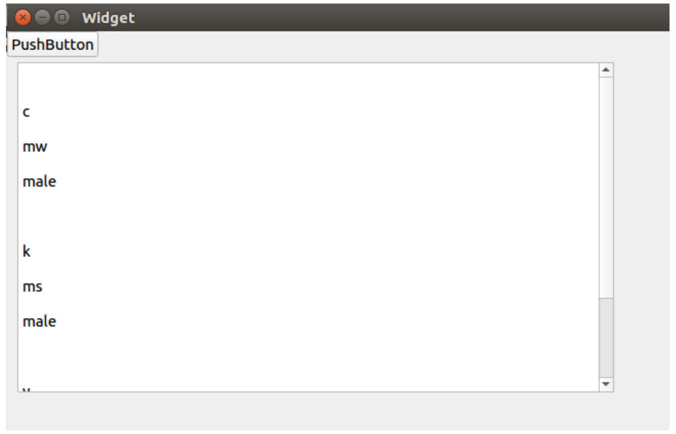
```
Widget::~Widget()
{
}
```

-- 다음 예제

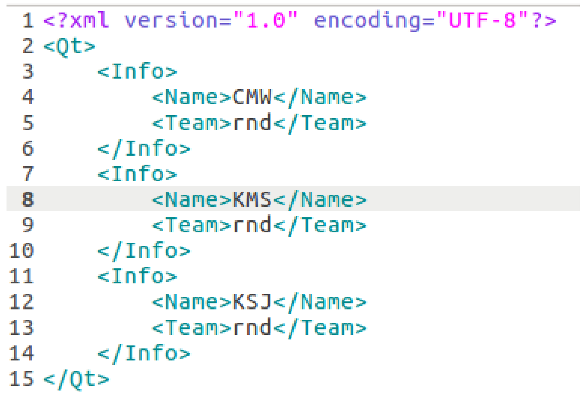
1. XML (SAX)



```xml
<?xml version="1.0" encoding="UTF-8"?>
<members>
    <member>
        <firstname>c</firstname>
        <lastname>mw</lastname>
        <gender>male</gender>
    </member>
    <member>
        <firstname>k</firstname>
        <lastname>ms</lastname>
        <gender>male</gender>
    </member>
    <member>
        <firstname>y</firstname>
        <lastname>sy</lastname>
        <gender>female</gender>
    </member>
</members>
```



```
c
mw
male

k
ms
male
```



```xml
<?xml version="1.0" encoding="UTF-8"?>
<Qt>
    <Info>
        <Name>CMW</Name>
        <Team>rnd</Team>
    </Info>
    <Info>
        <Name>KMS</Name>
        <Team>rnd</Team>
    </Info>
    <Info>
        <Name>KSJ</Name>
        <Team>rnd</Team>
    </Info>
</Qt>
```

1. XML(eXtensible Markup Language) = 다목적 Markup 언어 → HTML 태생 한계로 인해 발명
   - XML 문서들을 읽고 분석

2. SAX(Simple API for XML) 방식
   - 이벤트 중심의 인터페이스
   - 문서의 전체 구조 정보를 메모리 상으로 로드하지 않고 문서 내의 특정 엘리먼트만 처리

3. DOM(Document Object Model) 방식
   - W3C의 공식 표준 → 문서 구조
   - XML 문서를 트리 구조로 구성 → 메모리에 전부 로드 (속도 ↑, 데이터 수정 편리)
   - 메모리 사용량 ↑, 속도가 느림(트리 모델을 생성해야 하므로)

Qt 에서는 위 지원을 위해 XML 관련 모듈을 제공함
- SAX Parser, DOM Parser 제공
- XML 사용 예 : 국제화파일 (.ts), 리소스파일 (.qrc), ui디자인 파일(.ui)

.pro 파일

```
QT       += core gui xml
```

widget.h 파일

```
#ifndef WIDGET_H
#define WIDGET_H

#include <QWidget>

#include <QFile>

QT_BEGIN_NAMESPACE
namespace Ui { class Widget; }
QT_END_NAMESPACE

class Widget : public QWidget
{
    Q_OBJECT

public:
    Widget(QWidget *parent = nullptr);
    ~Widget();

private:
    Ui::Widget *ui;

    QFile           *mReadFile;

private slots:
    void readButtonClicked();
};
#endif // WIDGET_H
```

widget.cpp 파일

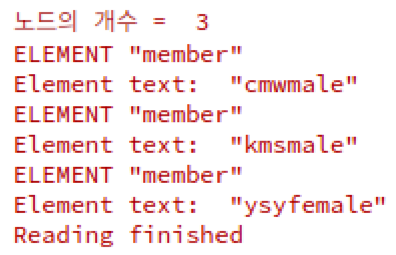
```
#include "widget.h"
#include "ui_widget.h"

#include <QFileDialog>
#include <QXmlStreamReader>
#include <QDebug>

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

    connect(ui->pushButton, &QPushButton::pressed,
            this,           &Widget::readButtonClicked);

    mReadFile = new QFile();
}

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

void Widget::readButtonClicked()
{
    QString fName = QFileDialog::getOpenFileName(this,
                                             "Open XML File",
                                             QDir::currentPath(),
                                             "XML Files (*.xml)");
    mReadFile->setFileName(fName);

    if(!QFile::exists(fName)) {
        ui->textEdit->setText("파일이 존재하지 않습니다. ");
        return;
    }
```

widget.cpp 파일

```
    if(!mReadFile->open(QIODevice::ReadOnly)) {
        ui->textEdit->setText("파일 Open 실패.");
        return;
    }

    QXmlStreamReader reader(mReadFile);

    QList<QString> members;


    QString inputData;
    while(!reader.atEnd())
    {
        reader.readNext();
        if(!reader.text().isEmpty()) {
            QString data = reader.text().toString();
            data.replace('₩n', "");
            data.replace('₩t', "");

            if(data.length() > 0)
                inputData.append(data).append("<br>");
        }
    }

    ui->textEdit->setText(inputData);
    reader.clear();
    mReadFile->close();
```

widget.cpp 파일

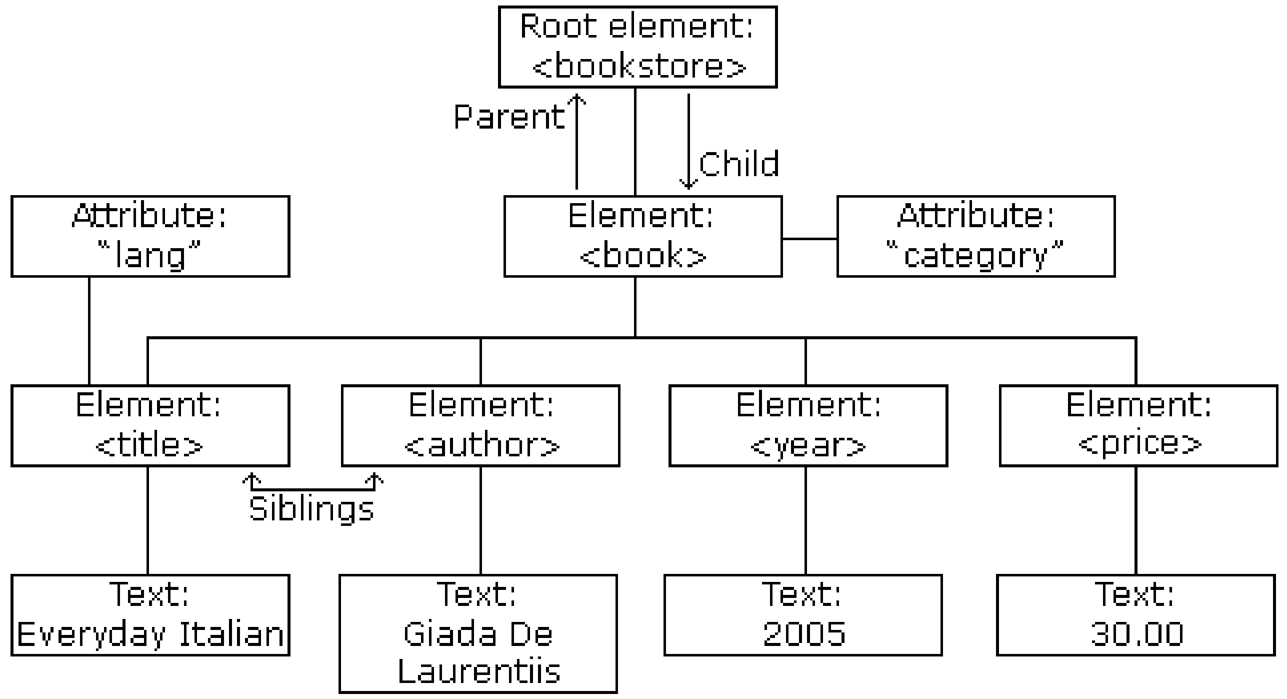
```
    QList<QString>    mOriData;

    mOriData.append("CMW");
    mOriData.append("rnd");

    mOriData.append("KMS");
    mOriData.append("intra");

    mOriData.append("KSJ");
    mOriData.append("operation");

    QFile writefile("output.xml");
    writefile.open(QIODevice::WriteOnly);

    QXmlStreamWriter xmlWriter(&writefile);
    xmlWriter.setAutoFormatting(true);
    xmlWriter.writeStartDocument();

    xmlWriter.writeStartElement("Qt");
    for(int i = 0 ; i < mOriData.count() ; i+=2)
    {
        xmlWriter.writeStartElement("Info");
        xmlWriter.writeTextElement("Name", mOriData.at(i));
        xmlWriter.writeTextElement("Team",  mOriData.at(1));
        xmlWriter.writeEndElement();
    }
    xmlWriter.writeEndElement();
    xmlWriter.writeEndDocument();

    writefile.close();
}
```



```
노드의 개수 =  3
ELEMENT "member"
Element text:  "cmwmale"
ELEMENT "member"
Element text:  "kmsmale"
ELEMENT "member"
Element text:  "ysyfemale"
Reading finished
```

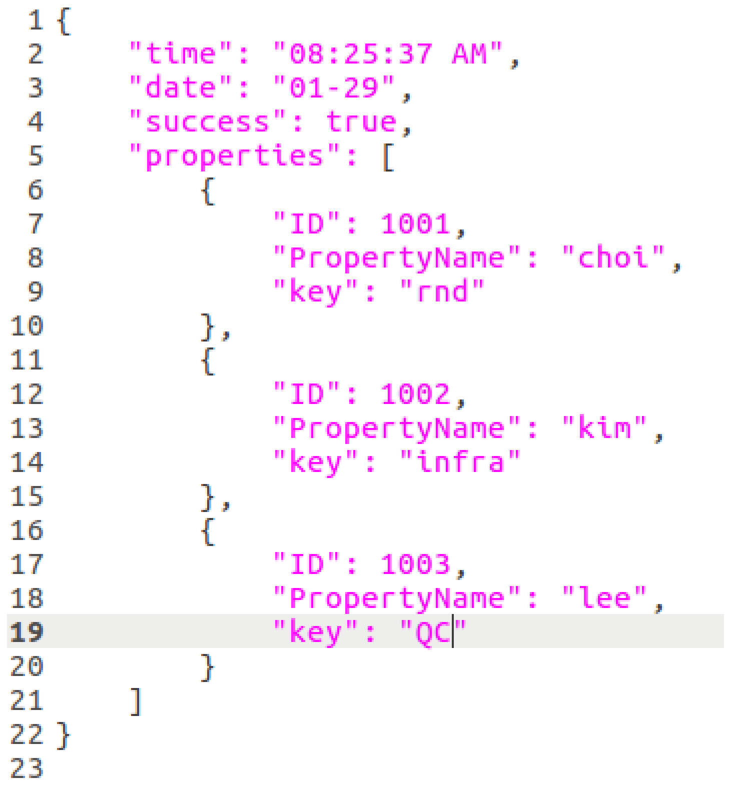
2. XML (DOM)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">XQuery Kick Start</title>
    <author>James McGovern</author>
    <author>Per Bothner</author>
    <author>Kurt Cagle</author>
    <author>James Linn</author>
    <author>Vaidyanathan Nagarajan</author>
    <year>2003</year>
    <price>49.99</price>
  </book>
  <book category="web" cover="paperback">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

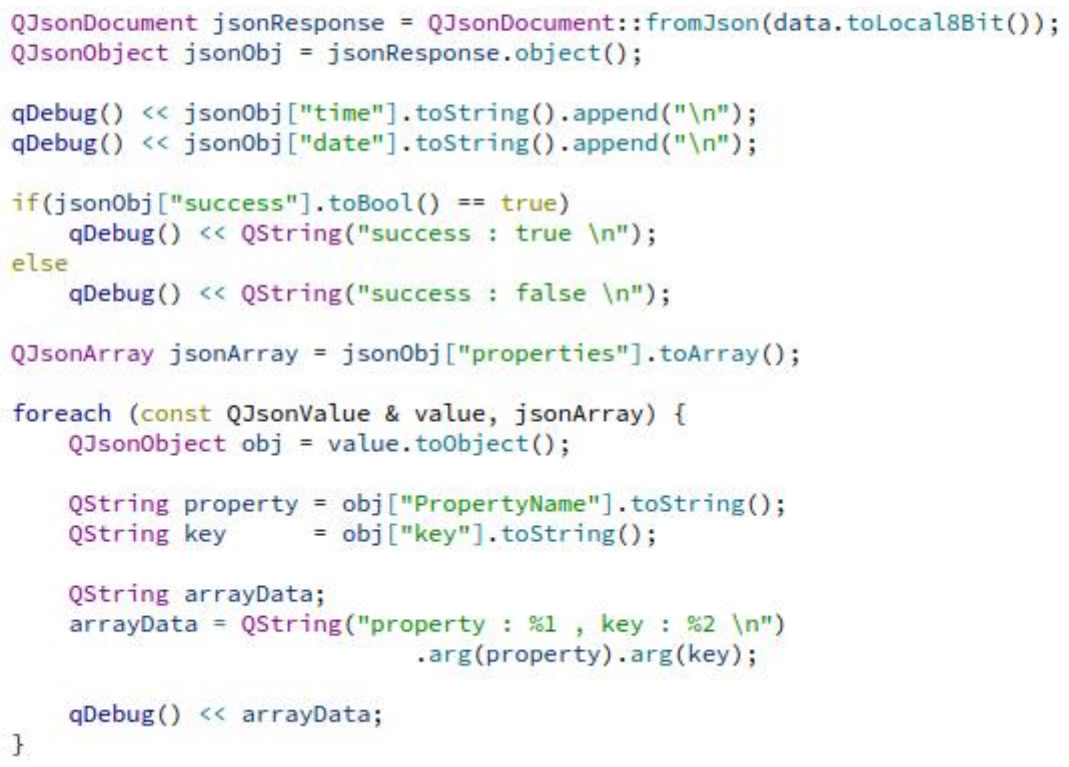


```
Widget::Widget(QWidget*)
"08:25:37 AM\n"
"01-29\n"
"success : true \n"
"property : choi , key : rnd \n"
"property : kim , key : infra \n"
"property : lee , key : QC \n"
```

3. Json



```json
{
    "time": "08:25:37 AM",
    "date": "01-29",
    "success": true,
    "properties": [
        {
            "ID": 1001,
            "PropertyName": "choi",
            "key": "rnd"
        },
        {
            "ID": 1002,
            "PropertyName": "kim",
            "key": "infra"
        },
        {
            "ID": 1003,
            "PropertyName": "lee",
            "key": "QC"
        }
    ]
}
```



```cpp
QJsonDocument jsonResponse = QJsonDocument::fromJson(data.toLocal8Bit());
QJsonObject jsonObj = jsonResponse.object();

qDebug() << jsonObj["time"].toString().append("\n");
qDebug() << jsonObj["date"].toString().append("\n");

if(jsonObj["success"].toBool() == true)
    qDebug() << QString("success : true \n");
else
    qDebug() << QString("success : false \n");

QJsonArray jsonArray = jsonObj["properties"].toArray();

foreach (const QJsonValue & value, jsonArray) {
    QJsonObject obj = value.toObject();

    QString property = obj["PropertyName"].toString();
    QString key      = obj["key"].toString();

    QString arrayData;
    arrayData = QString("property : %1 , key : %2 \n")
                        .arg(property).arg(key);

    qDebug() << arrayData;
}
```

widget.cpp 파일

```
Widget::~Widget()
{
}

void Widget::refreshDir()
{
    dirListWidget->clear();
    for(int i=0; i<directory->entryList().count(); i++)
    {
        dirListWidget->addItem(directory->entryList().at(i));
    }
}

void Widget::selectItem()
{
    filenameLineEdit->setText(dirListWidget->currentItem()->text());
}

void Widget::changeDir()
{
    QFileInfo checkDir(dirListWidget->currentItem()->text());
    if(checkDir.isDir())
    {
        directory->cd(dirListWidget->currentItem()->text());
        refreshDir();
    }
}
```

widget.cpp 파일

```
void Widget::makeDir()
{
    if(filenameLineEdit->text().length())
    {
        directory->mkdir(filenameLineEdit->text());
        directory->refresh();
        refreshDir();
    }
}

void Widget::removeDir()
{
    if(filenameLineEdit->text().length())
    {
        directory->rmdir(filenameLineEdit->text());
        directory->refresh();
        refreshDir();
    }
}

void Widget::renameDir()
{
    if(filenameLineEdit->text().length())
    {
        directory->rename(dirListWidget->currentItem()->text(), filenameLineEdit->text());
        directory->refresh();
        refreshDir();
    }
}
```