

```
/*
**
** Copyright (C) 2017 The Qt Company Ltd.
** Contact: https://www.qt.io/licensing/
**
** This file is part of the documentation of the Qt Toolkit.
**
** $QT_BEGIN_LICENSE:BSD$
** Commercial License Usage
** Licensees holding valid commercial Qt licenses may use this file in
** accordance with the commercial license agreement provided with the
** Software or, alternatively, in accordance with the terms contained in
** a written agreement between you and The Qt Company. For licensing terms
** and conditions see https://www.qt.io/terms-conditions. For further
** information use the contact form at https://www.qt.io/contact-us.
**
** BSD License Usage
** Alternatively, you may use this file under the terms of the BSD license
** as follows:
**
** *Redistribution and use in source and binary forms, with or without
** modification, are permitted provided that the following conditions are
** met:
**
** * Redistributions of source code must retain the above copyright
** notice, this list of conditions and the following disclaimer.
** * Redistributions in binary form must reproduce the above copyright
** notice, this list of conditions and the following disclaimer in
** the documentation and/or other materials provided with the
** distribution.
** * Neither the name of The Qt Company Ltd nor the names of its
** contributors may be used to endorse or promote products derived
** from this software without specific prior written permission.
**
**
** THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
** "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
** LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
** A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
** OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
** SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
** LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
** DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
** THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
** (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
** OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE."
**
** $QT_END_LICENSE$
**
**/
#define QT_PRINTSUPPORT_LIB
#include <QFile>
#include <QFileDialog>
#include <QTextStream>
#include <QMessageBox>
#ifdef QT_PRINTSUPPORT_LIB
#include <QPrintSupport/QtPrintSupportGlobal.h>
#ifdef QT_CONFIG(printer)
#include <QPrintDialog>
#endif // QT_CONFIG(printdialog)
#include <QPrinter>
#endif // QT_CONFIG(printer)
#include <QFont>
#include <QFontDialog>

#include <QPrintDialog>
#include <QPrinter>
```

```

#include "notepad.h"
#include "ui_notepad.h"

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

    connect(ui->actionNew, &QAction::triggered, this, &Notepad::newDocument);
    connect(ui->actionOpen, &QAction::triggered, this, &Notepad::open);
    connect(ui->actionSave, &QAction::triggered, this, &Notepad::save);
    connect(ui->actionSave_as, &QAction::triggered, this, &Notepad::saveAs);
    connect(ui->actionPrint, &QAction::triggered, this, &Notepad::print);
    connect(ui->actionExit, &QAction::triggered, this, &Notepad::exit);
    connect(ui->actionCopy, &QAction::triggered, this, &Notepad::copy);
    connect(ui->actionCut, &QAction::triggered, this, &Notepad::cut);
    connect(ui->actionPaste, &QAction::triggered, this, &Notepad::paste);
    connect(ui->actionUndo, &QAction::triggered, this, &Notepad::undo);
    connect(ui->actionRedo, &QAction::triggered, this, &Notepad::redo);
    connect(ui->actionFont, &QAction::triggered, this, &Notepad::selectFont);
    connect(ui->actionBold, &QAction::triggered, this, &Notepad::setFontBold);
    connect(ui->actionUnderline, &QAction::triggered, this, &Notepad::setFontUnderline);
    connect(ui->actionItalic, &QAction::triggered, this, &Notepad::setFontItalic);
    connect(ui->actionAbout, &QAction::triggered, this, &Notepad::about);

    // Disable menu actions for unavailable features
    #if !defined(QT_PRINTSUPPORT_LIB) || !QT_CONFIG(printer)
        ui->actionPrint->setEnabled(false);
    #endif

    #if !QT_CONFIG(dipboard)
        ui->actionCut->setEnabled(false);
        ui->actionCopy->setEnabled(false);
        ui->actionPaste->setEnabled(false);
    #endif
}

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

void Notepad::newDocument()
{
    currentFile.clear();
    ui->textEdit->setText(QString());
}

void Notepad::open()
{
    QString fileName = QFileDialog::getOpenFileName(this, "Open the file");
    QFile file(fileName);
    currentFile = fileName;
    if (!file.open(QIODevice::ReadOnly | QFile::Text)) {
        QMessageBox::warning(this, "Warning", "Cannot open file: " + file.errorString());
        return;
    }
    setWindowTitle(fileName);
    QTextStream in(&file);
    QString text = in.readAll();
    ui->textEdit->setText(text);
    file.close();
}

void Notepad::save()
{
    QString fileName;

```

```

// If we don't have a filename from before, get one.
if (currentFile.isEmpty()) {
    fileName = QFileDialog::getSaveFileName(this, "Save");
    currentFile = fileName;
} else {
    fileName = currentFile;
}
QFile file(fileName);
if (file.open(QIODevice::WriteOnly | QFile::Text) {
    QMessageBox::warning(this, "Warning", "Cannot save file: " + file.errorString());
    return;
}
setWindowTitle(fileName);
QTextStream out(&file);
QString text = ui->textEdit->toPlainText();
out << text;
file.close();
}

void Notepad::saveAs()
{
    QString fileName = QFileDialog::getSaveFileName(this, "Save as");
    QFile file(fileName);

    if (file.open(QFile::WriteOnly | QFile::Text)) {
        QMessageBox::warning(this, "Warning", "Cannot save file: " + file.errorString());
        return;
    }
    currentFile = fileName;
    setWindowTitle(fileName);
    QTextStream out(&file);
    QString text = ui->textEdit->toPlainText();
    out << text;
    file.close();
}

void Notepad::print()
{
    #if defined(QT_PRINTSUPPORT_LIB) && QT_CONFIG(printer)
    QPrinter printDev;
    #if QT_CONFIG(printdialog)
    QPrintDialog dialog(&printDev, this);
    if (dialog.exec() == QDialog::Rejected)
        return;
    #endif // QT_CONFIG(printdialog)
    ui->textEdit->print(&printDev);
    #endif // QT_CONFIG(printer)
}

void Notepad::exit()
{
    QApplication::quit();
}

void Notepad::copy()
{
    #if QT_CONFIG(dipboard)
    ui->textEdit->copy();
    #endif
}

void Notepad::cut()
{
    #if QT_CONFIG(dipboard)
    ui->textEdit->cut();
    #endif
}

void Notepad::paste()
{

```

```

#if QT_CONFIG(dipboard)
    ui->textEdit->paste();
#endif
}

void Notepad::undo()
{
    ui->textEdit->undo();
}

void Notepad::redo()
{
    ui->textEdit->redo();
}

void Notepad::selectFont()
{
    bool fontSelected;
    QFont font = QFontDialog::getFont(&fontSelected, this);
    if (fontSelected)
        ui->textEdit->setFont(font);
}

void Notepad::setFontUnderline(bool underline)
{
    ui->textEdit->setFontUnderline(underline);
}

void Notepad::setFontItalic(bool italic)
{
    ui->textEdit->setFontItalic(italic);
}

void Notepad::setFontBold(bool bold)
{
    bold ? ui->textEdit->setFontWeight(QFont::Bold) :
        ui->textEdit->setFontWeight(QFont::Normal);
}

void Notepad::about()
{
    QMessageBox::about(this, tr("About MDI"),
        tr("The <b>Notepad</b> example demonstrates how to code a basic "
            "text editor using QtWidgets"));
}

void Notepad::on_actionPrint_2_triggered()
{
    QPrinter printer;
    printer.setPrinterName("My printer");
    QPrintDialog dialog(&printer, this);
    if(dialog.exec() == QDialog::Rejected) return;
    ui->textEdit->print(&printer);
}
}

```