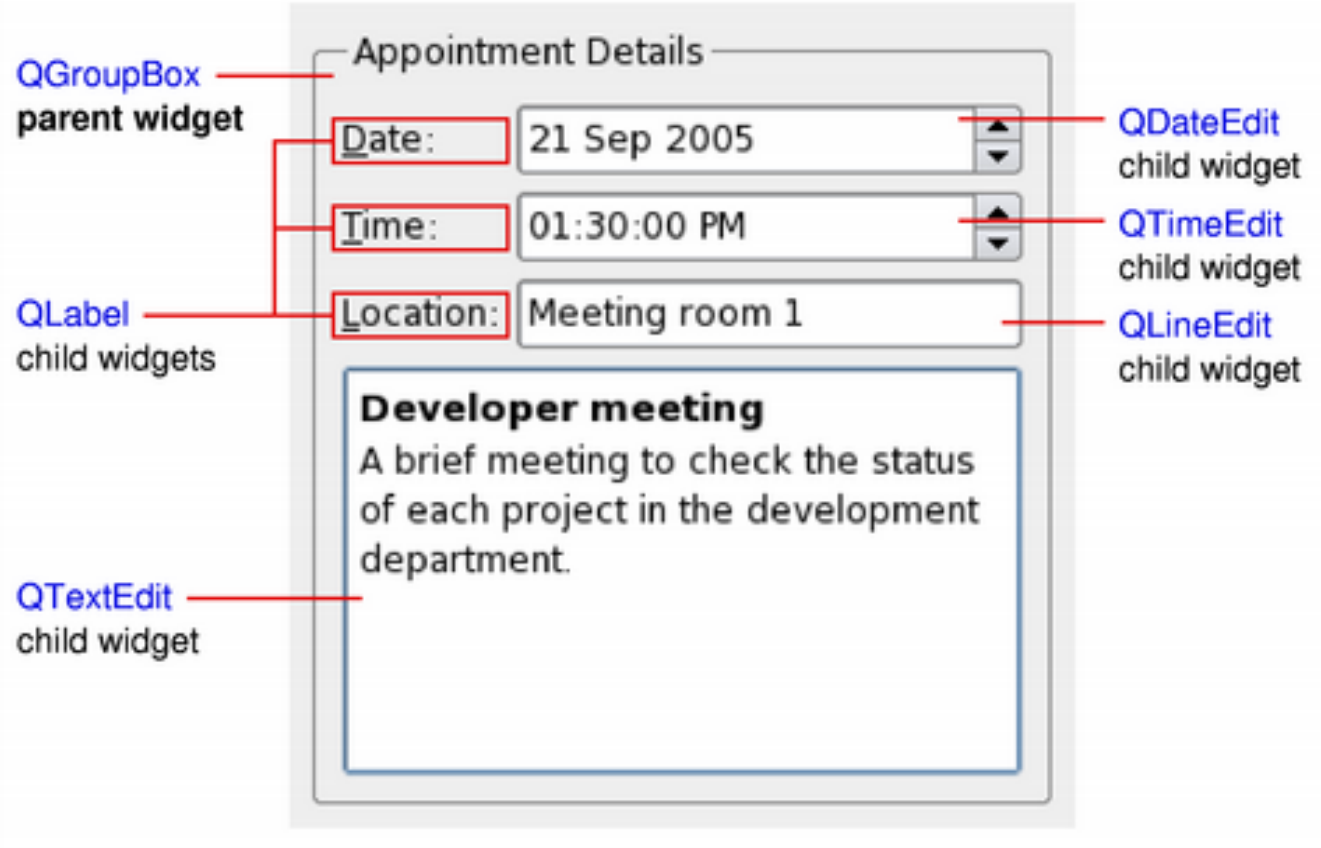
Tampereen yliopisto
Tampere University

User interface

Qt Creator

UI and component layout



Component layout

- Three choices:
  - Absolute coordinates
  - Manual layout
  - Layout managers
- In Qt: also QML

Absolute coordinates

- Programmer sets the coordinates and size of the components
- User cannot resize the window
- Changing fonts may truncate text
- The components might have inappropriate sizes for some styles
- Program maintenance becomes painful

Manual layout

- Programmer still sets the positions of the components, but their sizes are calculated based on the window size
- Requires overriding resizeEvent() function of the dialog
- Result is typically better than with absolute coordinates, but implementation is still hard

Layout managers in Qt

A simple way to organize UI components automatically

- Describe the layout of the components in UI
- Automatically adjust the layout based in response to font changes, content changes, window resizing
- Ensure a regular order of the components, and keep the UI usable

Layout managers in Qt

- Rough sizes:
  - Minimum size
  - Maximum size
  - Preferred size
- Widget can be scalable, or it can have a preferred size
- Layout manager calculates the size and position of the widget whenever the layout or the size of the window changes – Resize event

Layout managers in Qt

- The easiest way to pretty layout is provided by:
  - `QHBoxLayout`
  - `QVBoxLayout`
  - `QGridLayout`, ja
  - `QFormLayout`

  which are possible to compose inside of each other

QHBoxLayout

For horizontal layout

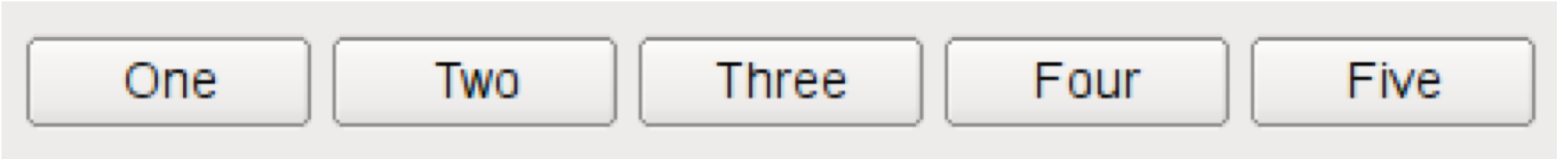


Fig: Qt documentation

QVBoxLayout

For vertical layout

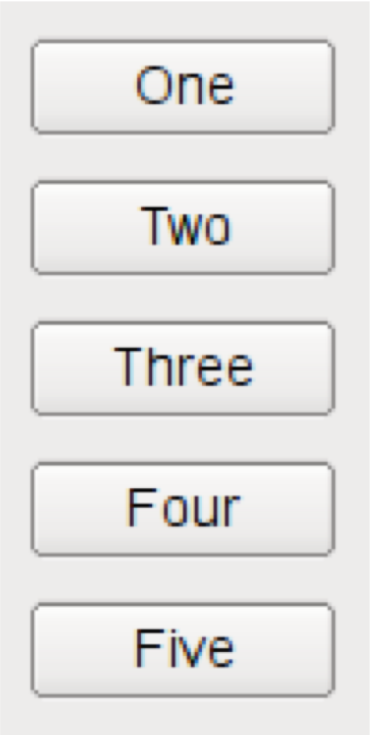


Fig: Qt documentation

QGridLayout

For grid layout

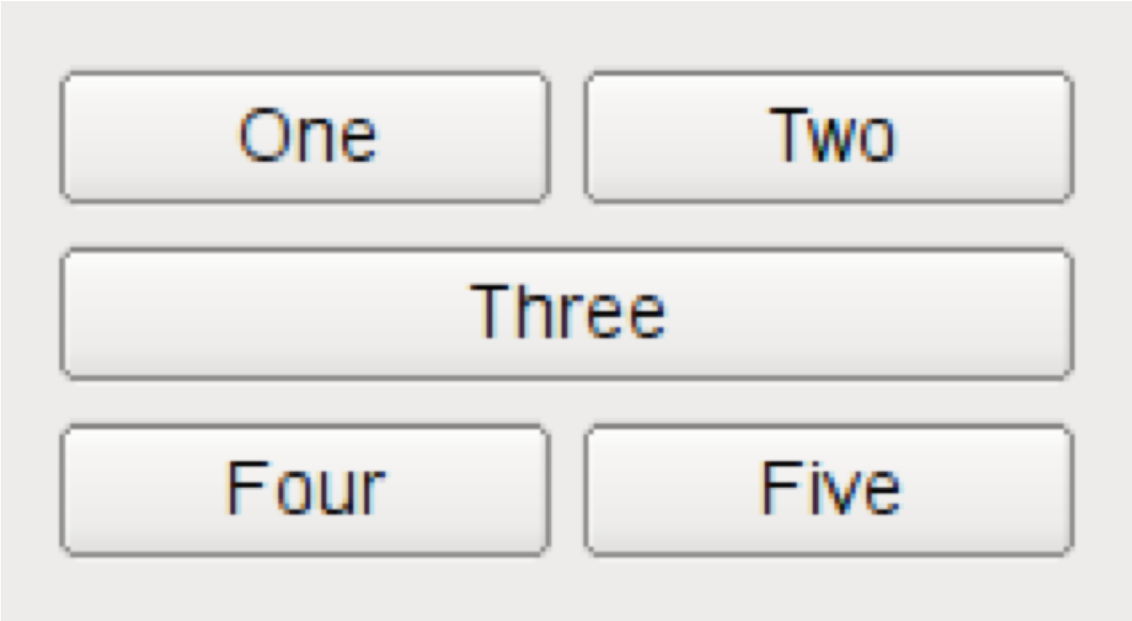


Fig: Qt documentation

QFormLayout

For two columns

Fig: Qt documentation

Nested layout

A dialog window typically consists of several nested layouts

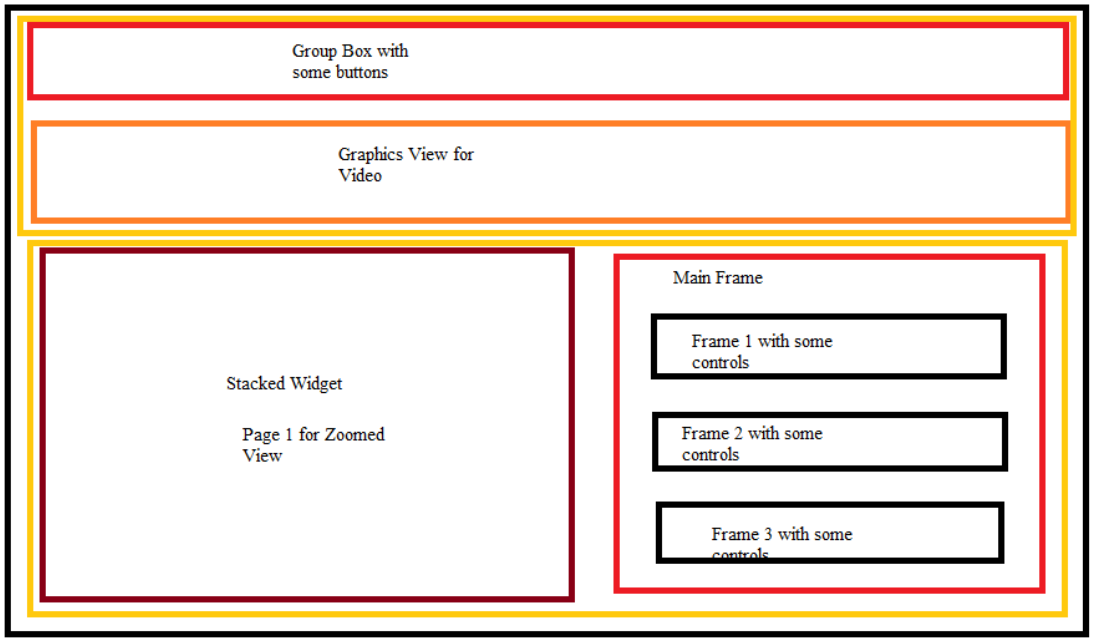


Fig: Stackoverflow

Parent

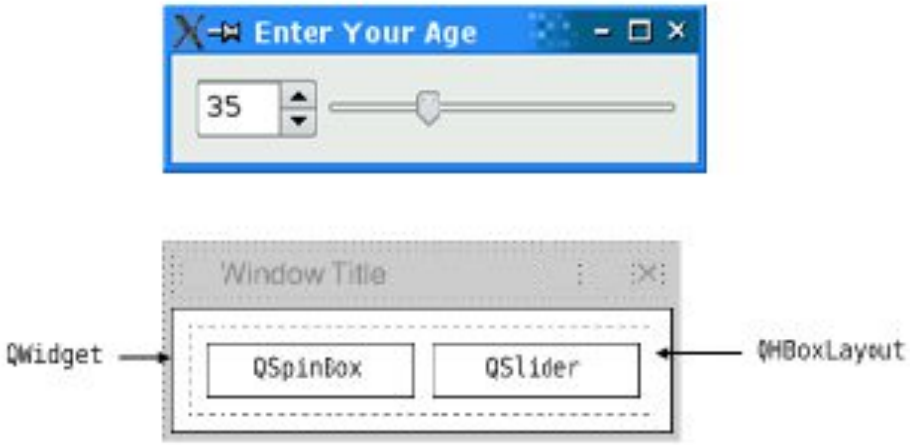
- Layout sets a parent widget (reparent) for a widget automatically
- Parent widget is not itself a layout but a widget, which has a layout set in (setLayout)

Example

- Three widgets:
  - parent: QWidget (main window)
  - children: QSpinBox and QSlider
- Signals and slots synchronize widgets

```
#include <QApplication>
#include <QHBoxLayout>
#include <QSlider>
#include <QSpinBox>

int main(int argc, char *arch[]) {
    QApplication app(archc, argv);
    QWidget *window = new QWidget;
    window->setWindowTitle("Enter your age");
    QSpinBox *spinBox = new QSpinBox;
    QSlider *slider = new QSlider(QT::Horizontal);
    spinBox->setRange(0, 130);
    slider->setRange(0, 130);              // to be
continued
```



```
    QObject::connect(spinBox,
SIGNAL(valueChanged(int)),
                        slider, SLOT(setValue(int)));
    QObject::connect(slider,
SIGNAL(valueChanged(int)),
                        spinBox, SLOT(setValue(int)));
    spinBox->setValue(35);
    QHBoxLayout *layout = new QHBoxLayout;
    layout->addWidget(spinBox);
    layout->addWidget(slider);
    window->setLayout(layout);
    window->show();
    return app.exec();
}
```

Using several windows

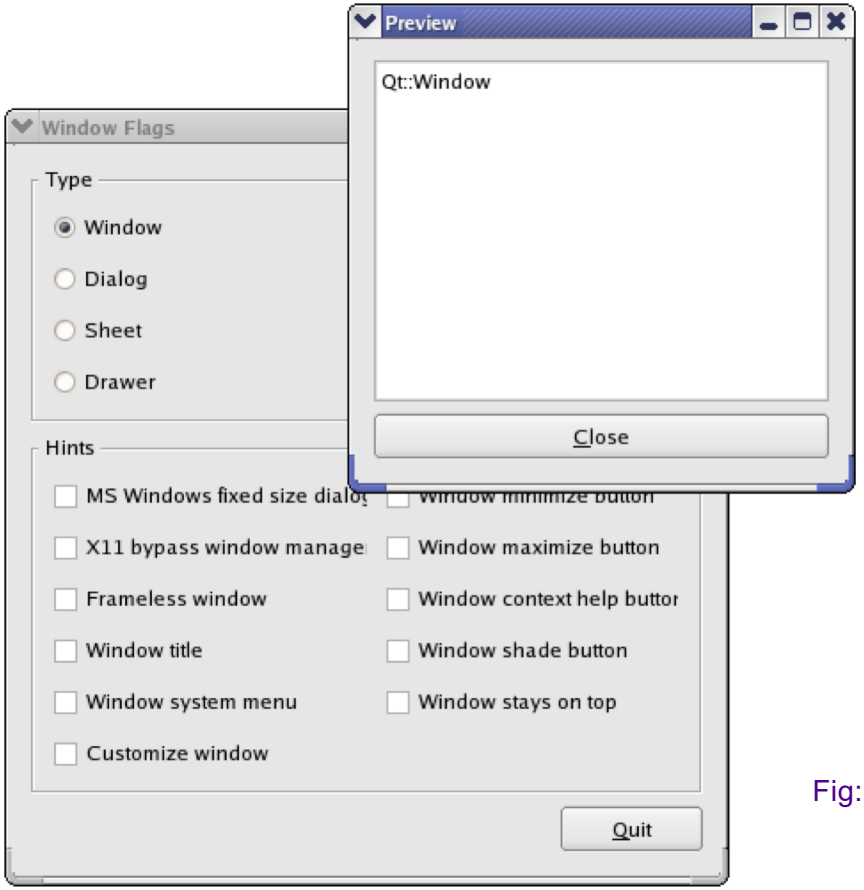


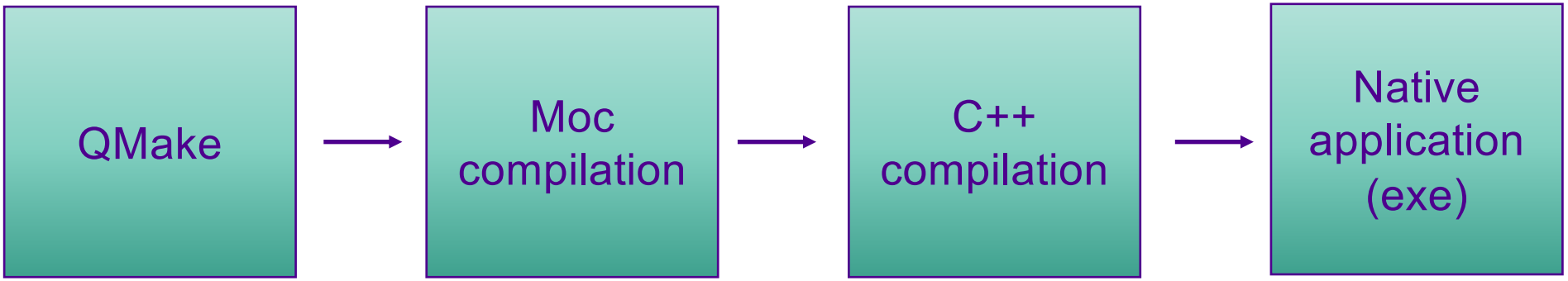
Fig: Qt documentation

Tampereen yliopisto
Tampere University

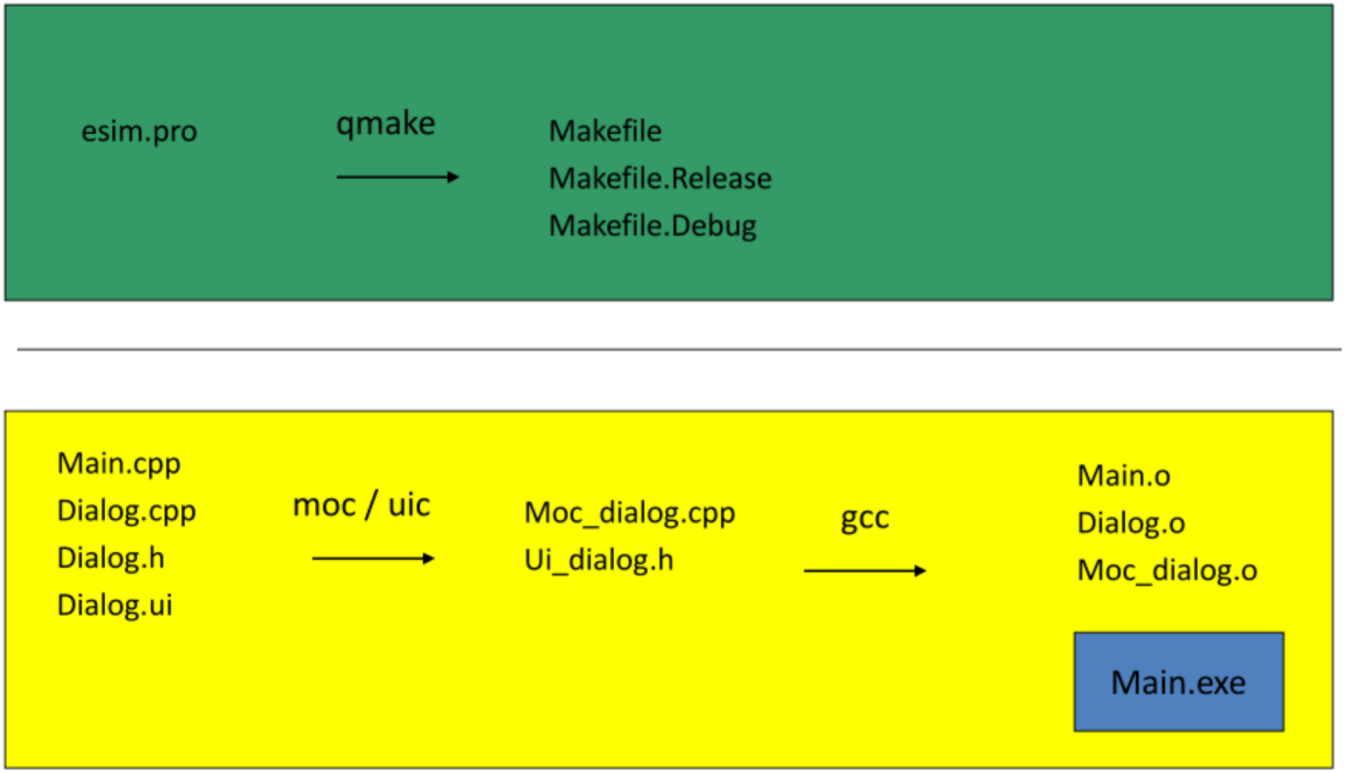
Qt under the hood

Code less. Create more.
Deploy everywhere.

Qt: compilation process



Qt: compilation process



Qt: compilation process

- Files needed in compilation are created by qmake based on information given in the project file (.pro)
- UI files (.ui) are compiled with uic (User Interface Compiler) and meta files with moc (Meta Object Compiler)
- moc generates temporary code accepted by C++ compiler

UI compiler (uic)

User Interface

- Qt Designer generates an XML file
- uic reads the file and compiles it to the corresponding C++ header file

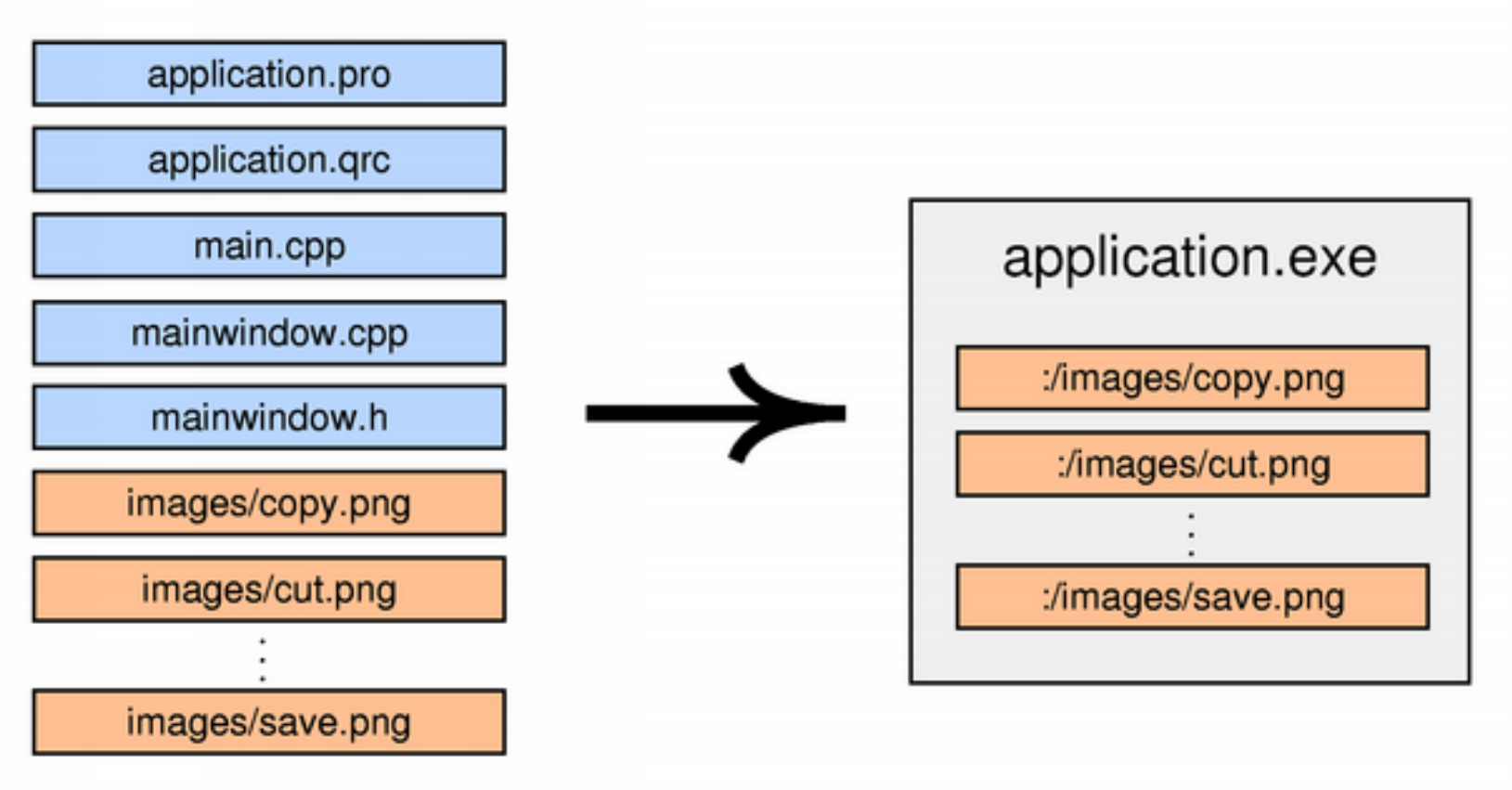
Tampereen yliopisto
Tampere University

Meta Object Compiler (moc)

Meta Object System

- `QObject` class tells that an object exploits meta object system
- `Q_OBJECT` macro sets meta object properties on (e.g. signals and slots)
- moc adds the code, needed in implementing the properties, in `QObject` subclass

Resources


application.pro
application.qrc
main.cpp
mainwindow.cpp
mainwindow.h
images/copy.png
images/cut.png
images/save.png
application.exe
:/images/copy.png
:/images/cut.png
:/images/save.png