Technische Universität Braunschweig

Institut für Flugführung

GUI-Erstellung in Qt

Erstellung einer graphischen Benutzeroberfläche in Qt

Prof. Dr.-Ing. Peter Hecker, Dipl.-Ing. Paul Frost, 30. Mai 2017

Agenda

04. April	Kick-Off
11. April	Projektmanagement
18. April	Prozessmodelle
25. April	Versionsverwaltung
02. Mai	Einführung Arduino/Funduino
09. Mai	Entwicklungsumgebungen und Debugging
16. Mai	Dokumentation und Testing
23. Mai	Dateieingabe und -ausgabe
30. Mai	**GUI-Erstellung mit Qt**
06. Juni	Exkursionswoche
13. Juni	Bibliotheken
20. Juni	Netzwerke
27. Juni	Projektarbeit
04. Juli	Projektarbeit
11. Juli	Vorbereitung der Abgabe

Teil I

Wiederholung

Teil II

GUI-Erstellung

Qt GUI/Widgets Module

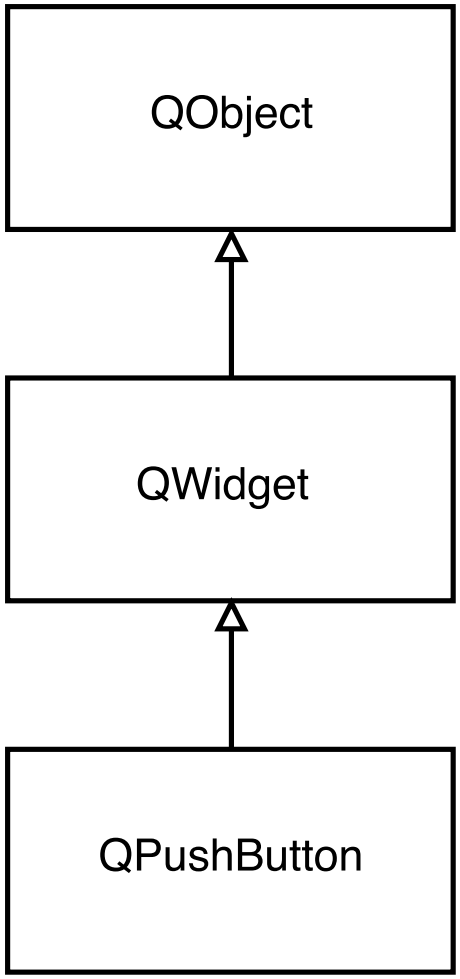
- Qt besitzt eigene Module für GUIs
  - Qt GUI
    Zentrales Modul für graphische Elemente
  - Qt Widgets
    High-Level Objekte, wie z. B. Fenster, Slider und Buttons
- Diese Module enthalten Klassen für eine plattformübergreifende GUI-Programmierung
- Module werden automatisch bei der Erstellung einer Qt-Widgets-Applikation eingebunden

Bei der Konsolenapplikation werden die Module nicht geladen!

Qt Widgets

- Elementare Bausteine für alle GUI-Elemente
- Jedes darstellbare GUI-Element ist von der Klasse `QWidget` abgeleitet
- `QWidget` Klassen sind auch von der `QObjekt` Klasse abgeleitet

→ Verwendung von Signals und Slots möglich



Widget Geometrie

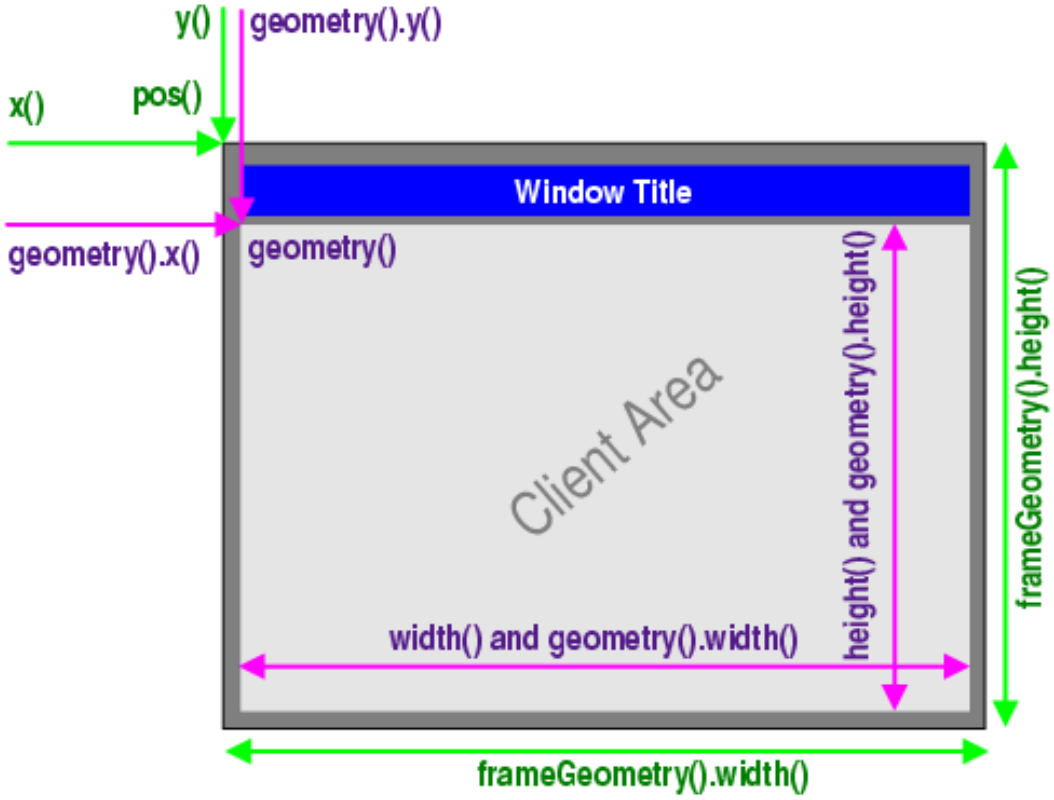


Abbildung 1: Übersicht der Geometriefunktionen für eine `QWidget`-Klasse[1]

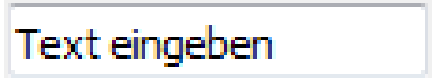
[1] http://doc.qt.io/qt-5/application-windows.html

Widget Übersicht

- `QPushButton`

Widget Übersicht

- `QPushButton`
- `QLineEdit`



Widget Übersicht

- `QPushButton`
- `QLineEdit`
- `QComboBox`

Widget Übersicht

- QPushButton
- QLineEdit
- QComboBox
- QSpinBox

Widget Übersicht

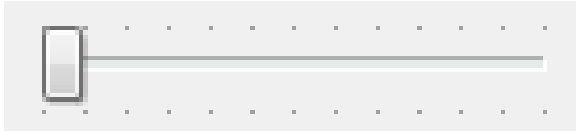
- QPushButton
- QLineEdit
- QComboBox
- QSpinBox
- QCheckBox

Widget Übersicht

- QPushButton
- QLineEdit
- QComboBox
- QSpinBox
- QCheckBox
- QRadioButton

Widget Übersicht

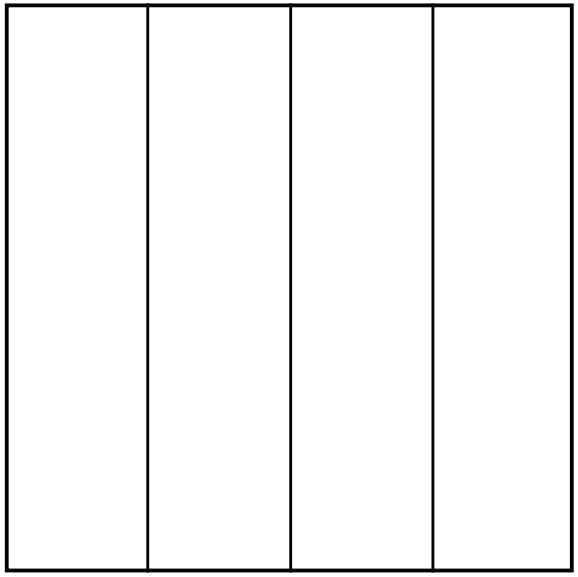
- QPushButton
- QLineEdit
- QComboBox
- QSpinBox
- QCheckBox
- QRadioButton
- QSlider



Layout Übersicht

- Layouts positionieren die enthaltenen Widgets
- Ein `QWidget`-Objekt kann ein Layout beinhalten
- Neben Widgets können auch Layouts hinzugefügt werden
- Erst Widgets erstellen und anschließend in ein Layout hinzufügen

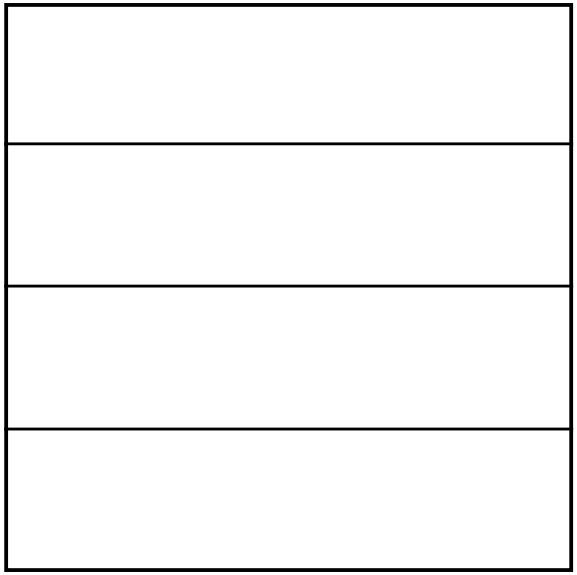
- `QHBoxLayout`



Layout Übersicht

- Layouts positionieren die enthaltenen Widgets
- Ein `QWidget`-Objekt kann ein Layout beinhalten
- Neben Widgets können auch Layouts hinzugefügt werden
- Erst Widgets erstellen und anschließend in ein Layout hinzufügen

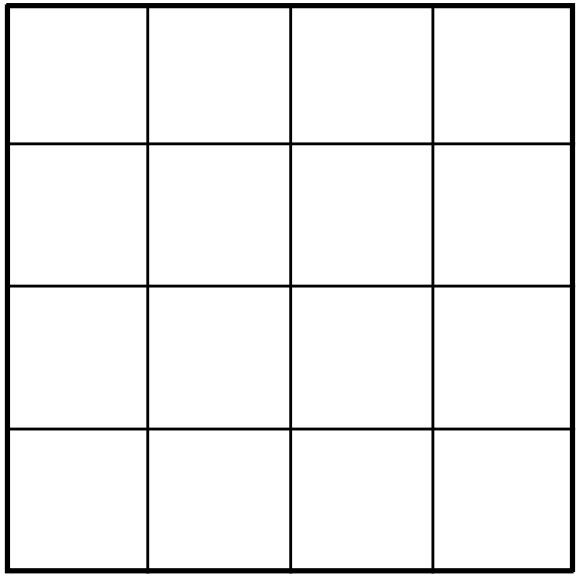
- `QHBoxLayout`
- `QVBoxLayout`



Layout Übersicht

- Layouts positionieren die enthaltenen Widgets
- Ein `QWidget`-Objekt kann ein Layout beinhalten
- Neben Widgets können auch Layouts hinzugefügt werden
- Erst Widgets erstellen und anschließend in ein Layout hinzufügen

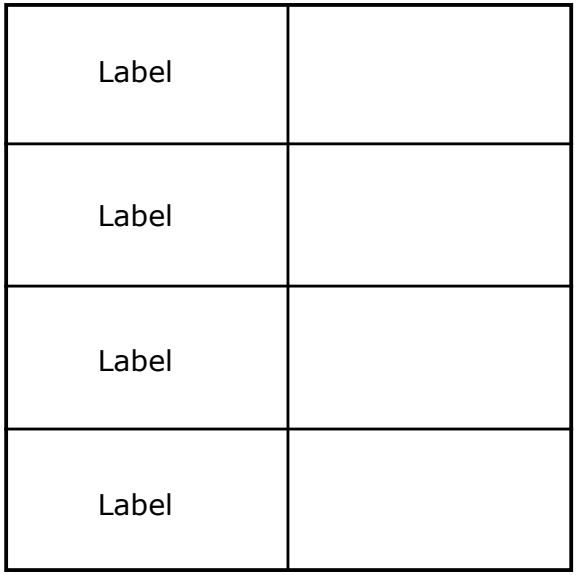
- `QHBoxLayout`
- `QVBoxLayout`
- `QGridLayout`



Layout Übersicht

- Layouts positionieren die enthaltenen Widgets
- Ein `QWidget`-Objekt kann ein Layout beinhalten
- Neben Widgets können auch Layouts hinzugefügt werden
- Erst Widgets erstellen und anschließend in ein Layout hinzufügen

- `QHBoxLayout`
- `QVBoxLayout`
- `QGridLayout`
- `QFormLayout`



Graphische Benutzeroberfläche erstellen

1. Qt-Projekt erstellen (Qt-Widgets-Anwendung)
2. GUI-Klasse auswählen
3. Layout im Designer erstellen
4. GUI in Quelltext einbinden

Hello World

Skizze

Anforderungen

- Der Benutzer soll eine Schaltfläche zur Verfügung haben mit der Aufschrift „Klick mich“
- Bei einem Linksklick auf die Schaltfläche soll in einem Label die Zeichenfolge „Hello World“ angezeigt werden.
- Schaltfläche und Label sollen wie in Abbildung 2 angeordnet sein

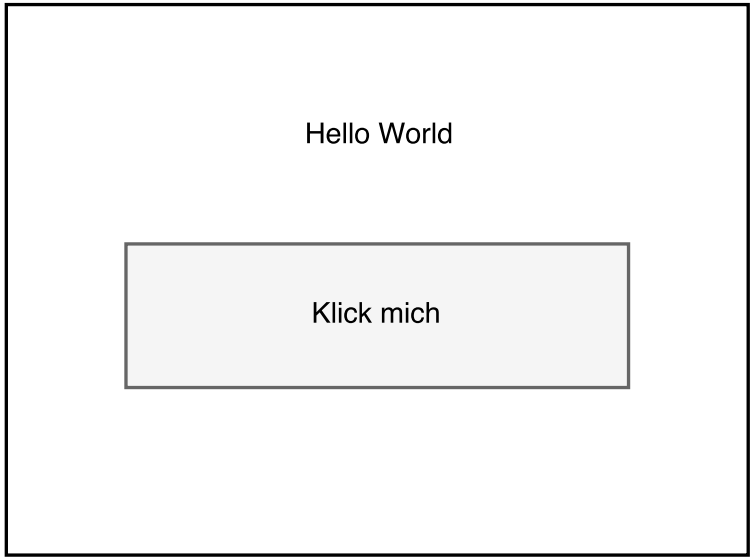
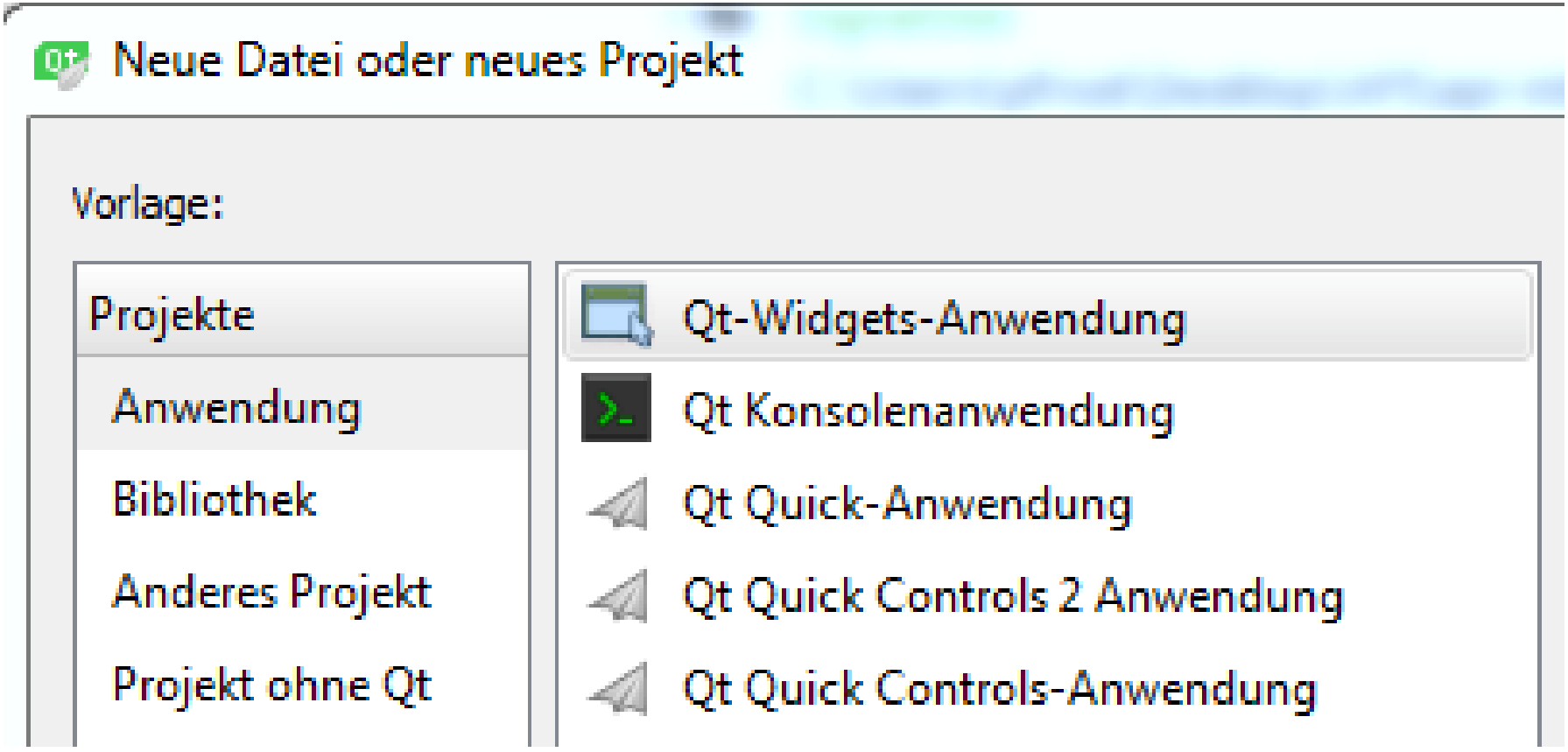


Abbildung 2: Skizze des „Hello World“-Fensters

Hello World

Projekt erstellen



Hello World

Klasse erstellen

Parameter der Klasse

Geben Sie Informationen bezüglich der Klassen ein, für die Sie Quelltexte generieren wollen.

Klassenname:	Widget
Basisklasse:	QWidget
Header-Datei:	widget.h
Quelldatei:	widget.cpp
Form-Datei generieren:	☑
Form-Datei:	widget.ui

Hello World — Quelltext

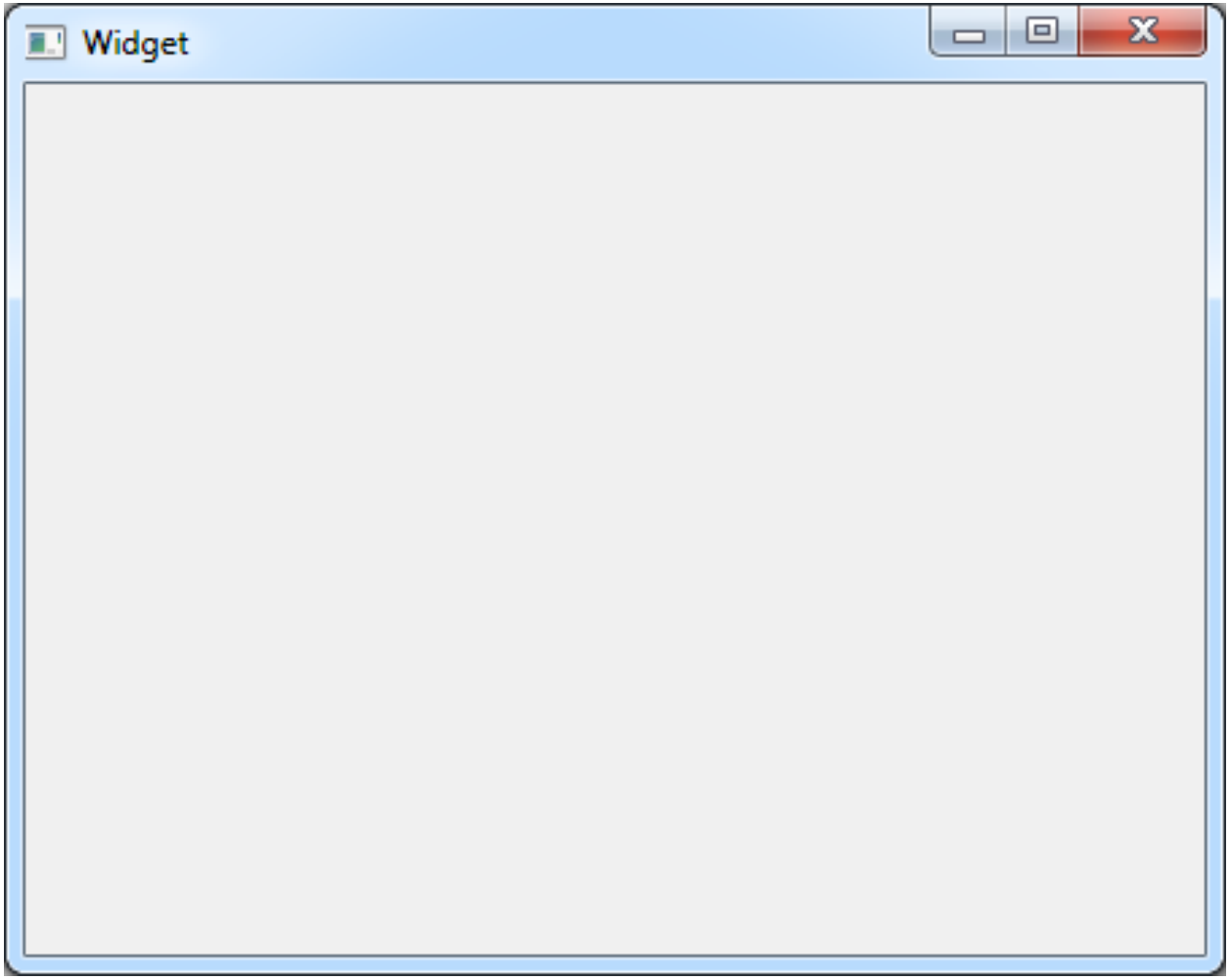
- Quelltext wird automatisch angelegt
- Zugriff auf GUI-Elemente über das Objekt `ui` möglich

Listing 1: code/QtHelloWorldGUI/widget.h

```
1  #include <QWidget>
2
3  namespace Ui {
4  class Widget;
5  }
6
7  class Widget : public QWidget
8  {
9    Q_OBJECT
10
11 private:
12   Ui::Widget *ui;
```

Hello World

Programm starten



Hello World

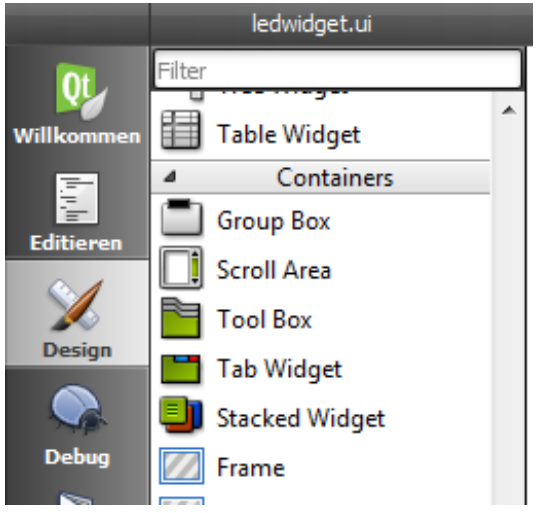
Benötigte Elemente

Elemente

`QWidget`	Fensterelement
`QPushbutton`	Schaltfläche in Qt
`QLabel`	Element für Beschriftungen

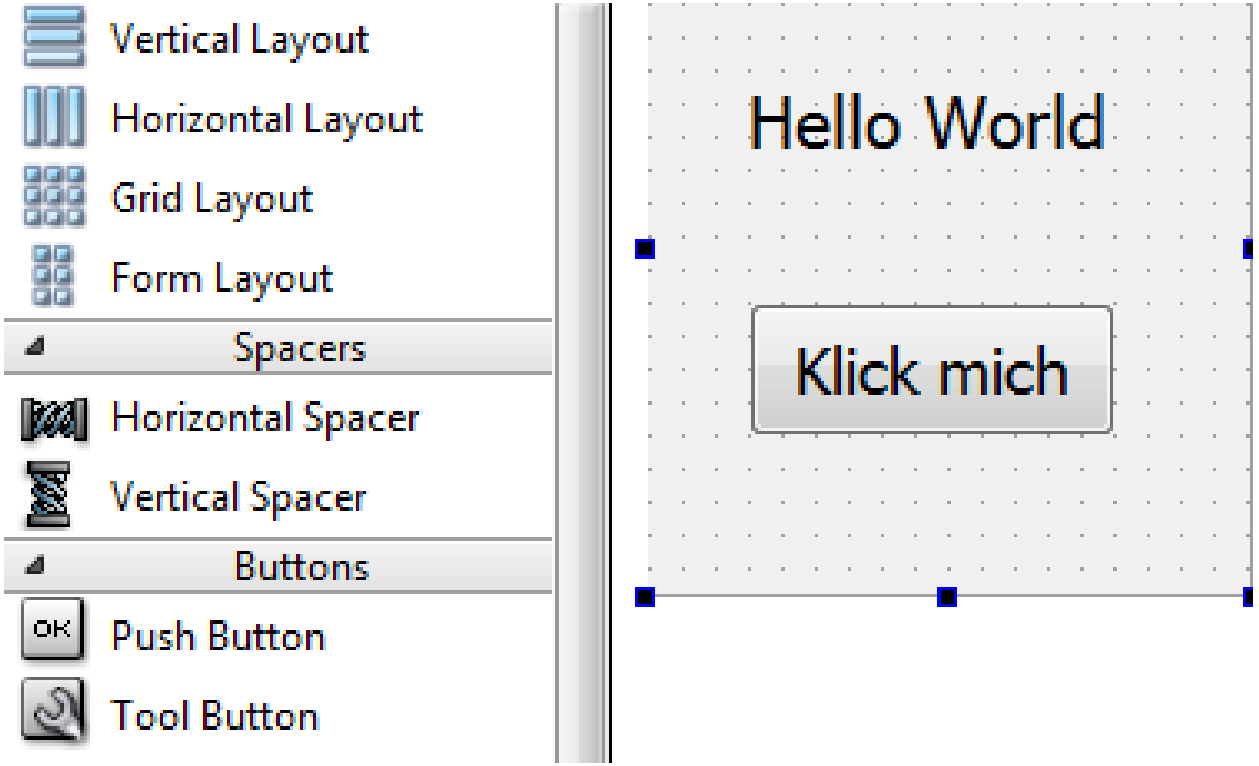
Qt Designer

Elemente können über den Qt Designer hinzugefügt werden



Hello World

Layout im Designer erstellen



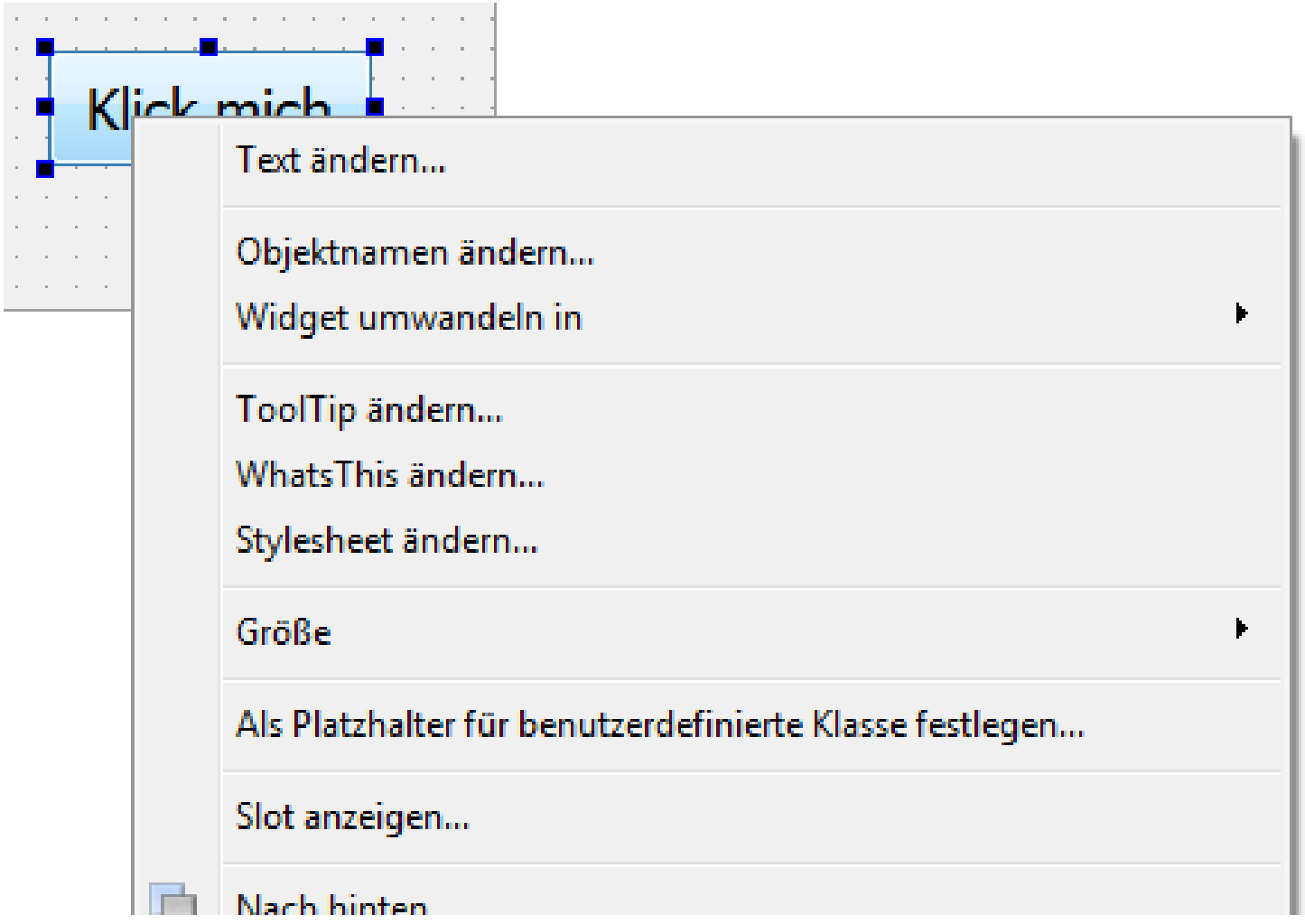
Hello World

Objektnamen anpassen

Eigenschaft	Wert
QObject	
objectName	hwButton
QWidget	
QAbstractButton	
text	Klick mich
icon	
iconSize	16 x 16
shortcut	
checkable	☐

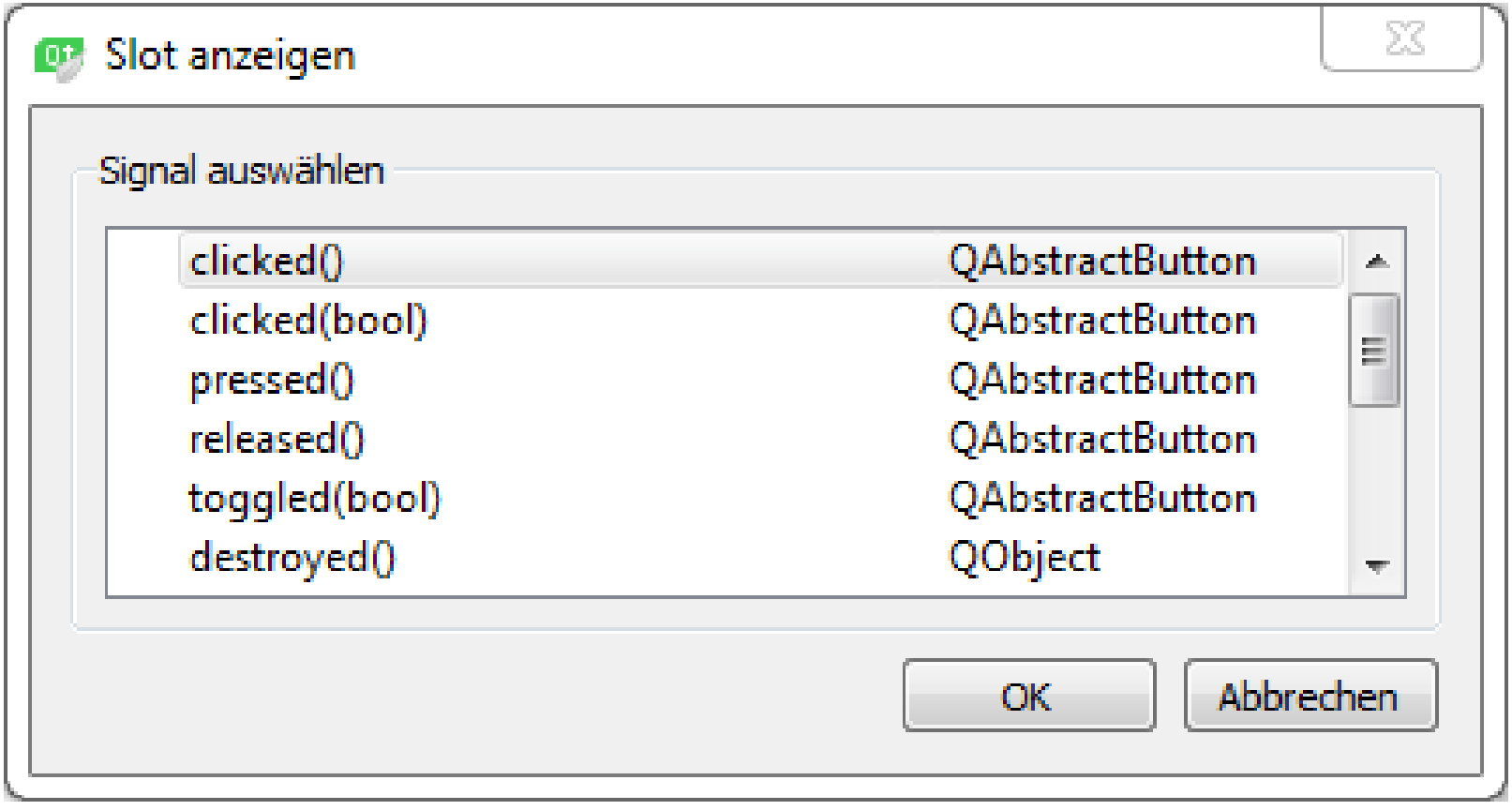
Hello World

Slot erstellen



Hello World

Slot erstellen



Hello World

Quelltext

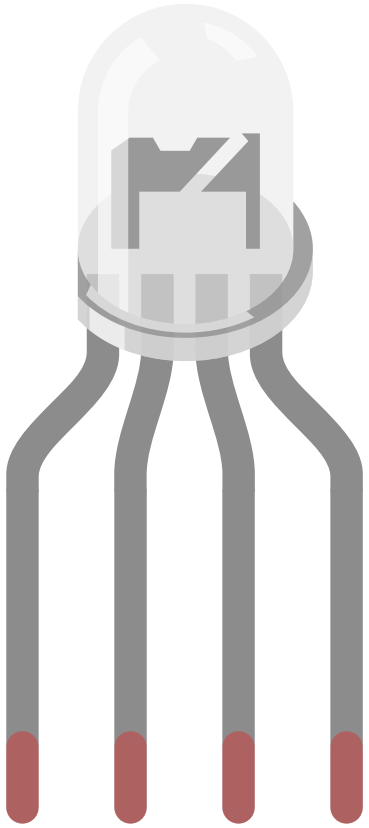
- Slot wird automatisch über den Dialog „Slot anzeigen“ angelegt
- Über den richtigen Namen wird der Slot automatisch mit dem Signal verbunden

  on_<*ObjektnameDesElements*>_clicked()

Listing 2: code/QtHelloWorldGUI/widget.cpp

```
1 void Widget::on_hwButton_clicked()
2 {
3   ui->hwLabel->setText("Hallo World");
4 }
```

LED-Farbwahl



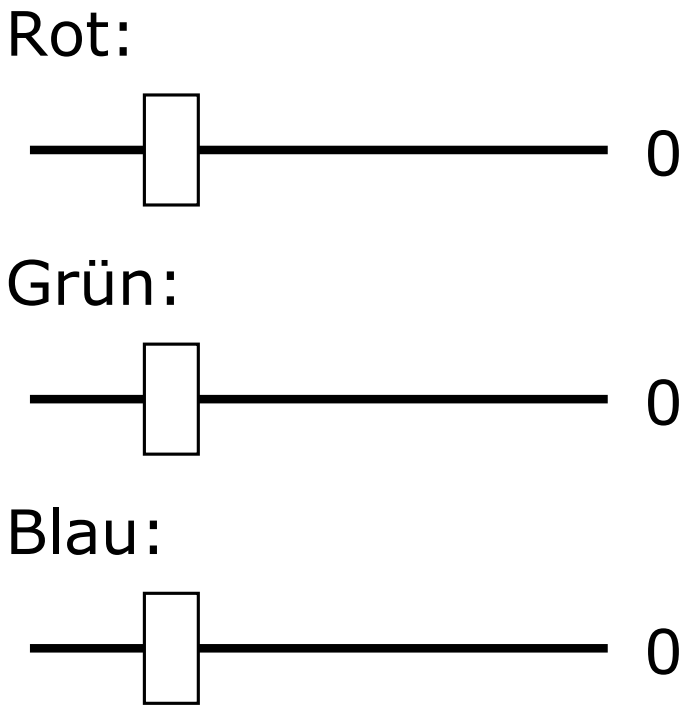
LED-Farbwahl

Anforderung

Anforderungen

- Steuerung der Farbe einer LED mittels Schiebereglern (QSlider)
- Jede Farbe der LED (rot, grün und blau) soll einen eigenen Slider erhalten
- Wertebereich: 0-100
- Der Wert soll rechts daneben dargestellt werden

Skizze



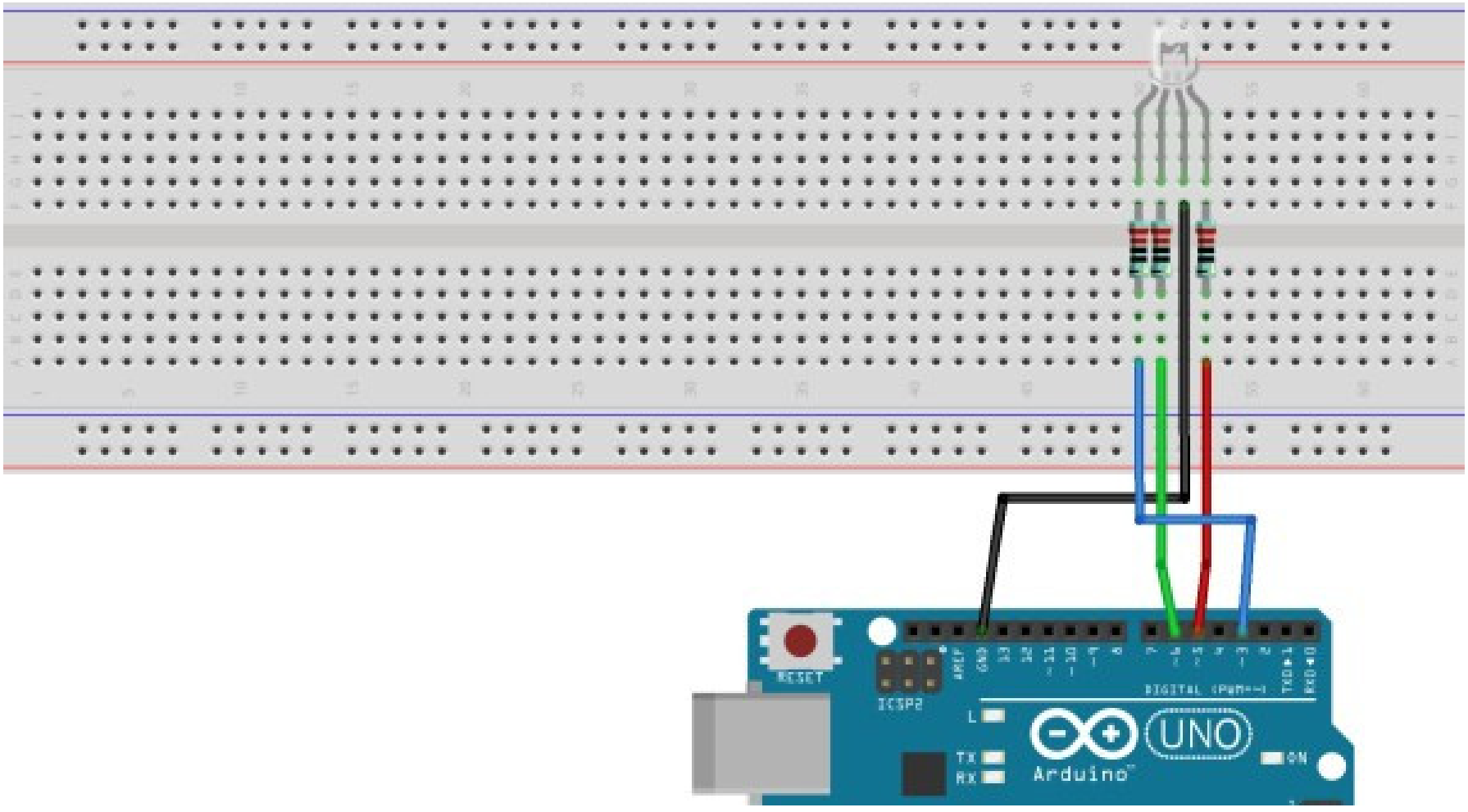
LED-Farbwahl

Qt-Designer

Praxisdemonstration Qt-Designer

LED-Farbwahl

Arduino Aufbau



LED-Farbwahl Arduino-Programmierung

Praxisdemonstration Arduino Programmierung

LED-Farbwahl Übertragungsprotokoll

- Übertragung erfolgt seriell
- 0-100 Sind reserviert für den Farbwertbereich
- 101-103 Wählen die Farbe aus

  101 Rot
  102 Grün
  103 Blau

- 2 Bytes werden für eine Farbeinstellung gesendet

	Byte 0	Byte 1	
Wert	101	0	→ 0 % Rot
Wert	101	100	→ 100 % Rot
Wert	102	50	→ 50 % Grün
Wert	103	20	→ 20 % Blau

LED-Farbwahl Qt-Schnittstelle

Praxisdemonstration Qt-Schnittstelle