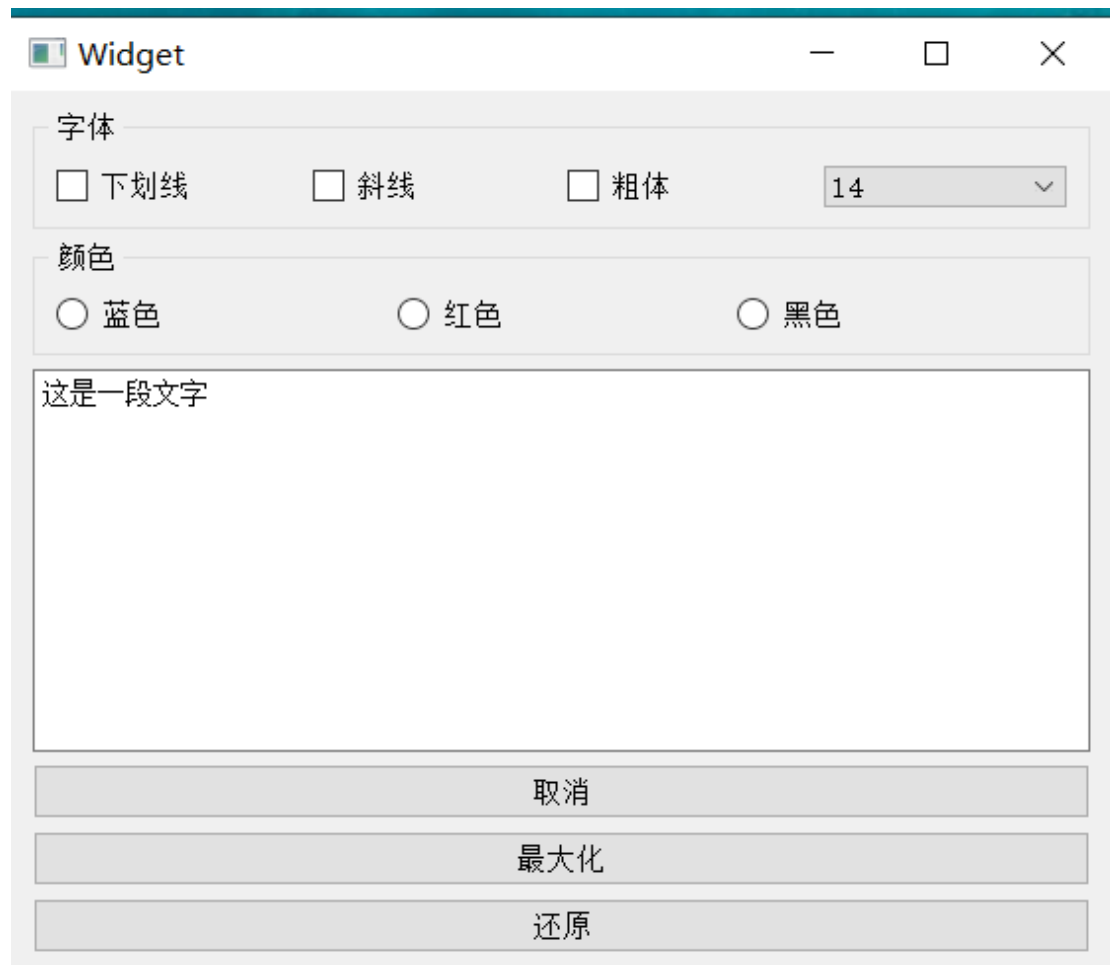


QT实战之字体编辑器

字体编辑器案例



本案例能学到

- 1. QT基本布局和基本控件
- 2. QT如何触发事件
- 3. QT如何编写事件代码
- 4. QTUI设计器的使用
- 5 QFont设置字体风格
- 6 QPalte设置字体颜色

UI布局

1. 界面组件的层次关系

为了将界面上的各个组件的分布设计得更加美观，经常使用一些容器类，如 `QGroupBox`、`QtabWidget`、`QFrame` 等。例如，将 3 个 `CheckBox` 组件放置在一个 `GroupBox` 组件里，该 `GroupBox` 组件就是这 3 个 `CheckBox` 的容器，移动这个 `GroupBox` 就会同时移动其中的 3 个 `CheckBox`。

图2-7显示的是设计图2-6界面的前期阶段。在窗体上放置了2个 `GroupBox` 组件，在 `groupBox1` 里放置 3 个 `CheckBox` 组件，在 `groupBox2` 里放置 3 个 `RadioButton` 组件。图 2-7 右侧 Object Inspector 里显示了界面上各组件之间的层次关系。



图 2-7 界面组件的放置及层次关系

UI布局

2. 布局管理

Qt 为界面设计提供了丰富的布局管理功能，在 UI 设计器中，组件面板里有 Layouts 和 Spacers 两个组件面板，在窗体上方的工具栏里有布局管理的按钮（如图 2-8 所示）。

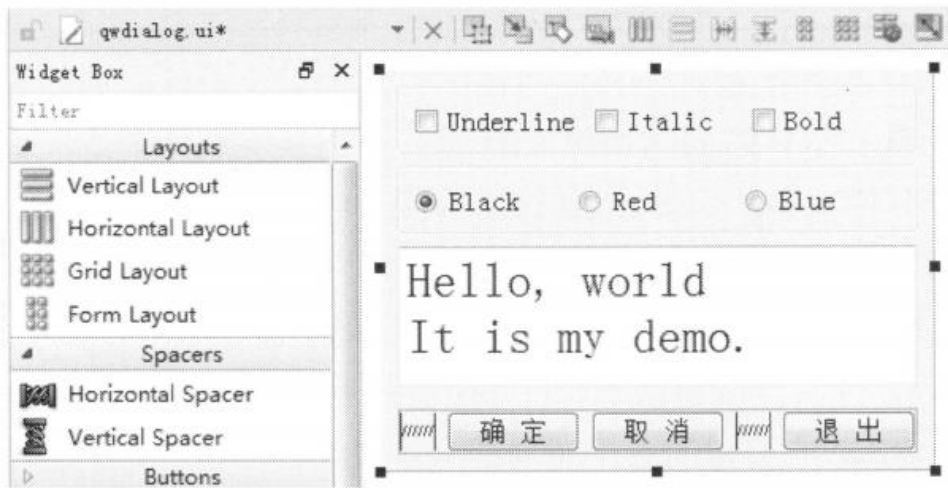








图 2-8 用于布局可视化设计的组件面板和工具栏

布局组件	功能
 Vertical Layout	垂直方向布局，组件自动在垂直方向上分布
 Horizontal Layout	水平方向布局，组件自动在水平方向上分布
 Grid Layout	网格状布局，网状布局大小改变时，每个网格的大小都改变
布局组件	功能
 Form Layout	窗体布局，与网格状布局类似，但是只有最右侧的一列网格会改变大小
 Horizontal Spacer	一个用于水平分隔的空格
 Vertical Spacer	一个用于垂直分隔的空格

信号和槽

信号与槽（Signal & Slot）是 Qt 编程的基础，也是 Qt 的一大创新。因为有了信号与槽的编程机制，在 Qt 中处理界面各个组件的交互操作时变得更加直观和简单。

信号（Signal）就是在特定情况下被发射的事件，例如 `PushButton` 最常见的信号就是鼠标单击时发射的 `clicked()` 信号，一个 `ComboBox` 最常见的信号是选择的列表项变化时发射的 `CurrentIndexChanged()` 信号。GUI 程序设计的主要内容就是对界面上各组件的信号的响应，只需要知道什么情况下发射哪些信号，合理地去响应和处理这些信号就可以了。

槽（Slot）就是对信号响应的函数。槽就是一个函数，与一般的 C++ 函数是一样的，可以定义在类的任何部分（`public`、`private` 或 `protected`），可以具有任何参数，也可以被直接调用。槽函数与一般的函数不同的是：槽函数可以与一个信号关联，当信号被发射时，关联的槽函数被自动执行。

信号与槽关联是用 `QObject::connect()` 函数实现的，其基本格式是：

```
QObject::connect(sender, SIGNAL(signal()), receiver, SLOT(slot()));
```

`connect()` 是 `QObject` 类的一个静态函数，而 `QObject` 是所有 Qt 类的基类，在实际调用时可以忽略前面的限定符，所以可以直接写为：

```
connect(sender, SIGNAL(signal()), receiver, SLOT(slot()));
```

信号和槽

(1) 一个信号可以连接多个槽，例如：

```
connect(spinNum, SIGNAL(valueChanged(int)), this, SLOT(addFun(int)));  
connect(spinNum, SIGNAL(valueChanged(int)), this, SLOT(updateStatus(int)));
```

这是当一个对象 `spinNum` 的数值发生变化时，所在窗体有两个槽进行响应，一个 `addFun()` 用于计算，一个 `updateStatus()` 用于更新状态。

当一个信号与多个槽函数关联时，槽函数按照建立连接时的顺序依次执行。

当信号和槽函数带有参数时，在 `connect()` 函数里，要写明参数的类型，但可以不写参数名称。

(2) 多个信号可以连接同一个槽，例如在本项目的设计中，让三个选择颜色的 `RadioButton` 的 `clicked()` 信号关联到相同的一个自定义槽函数 `setTextFontColor()`。

```
connect(ui->rBtnBlue, SIGNAL(clicked()), this, SLOT(setTextFontColor()));  
connect(ui->rBtnRed, SIGNAL(clicked()), this, SLOT(setTextFontColor()));  
connect(ui->rBtnBlack, SIGNAL(clicked()), this, SLOT(setTextFontColor()));
```

这样，当任何一个 `RadioButton` 被单击时，都会执行 `setTextFontColor()` 函数。

(3) 一个信号可以连接另外一个信号，例如：

```
connect(spinNum, SIGNAL(valueChanged(int)), this, SIGNAL(refreshInfo(int)));
```

这样，当一个信号发射时，也会发射另外一个信号，实现某些特殊的功能。

UI设计说明

组件设计

对象	类
Widget	QWidget
verticalLayout_2	QVBoxLayout
pbclose	QPushButton
pbok	QPushButton
pbresize	QPushButton
groupBox	QGroupBox
checkBoxct	QCheckBox
checkBoxxh	QCheckBox
checkBoxxx	QCheckBox
comboBox	QComboBox
groupBox_2	QGroupBox
radioButtonhes	QRadioButton
radioButtonhs	QRadioButton
radioButtonls	QRadioButton
plainTextEdit	QPlainTextEdit

属性	值
QObject	
QWidget	
Layout	
layoutName	verticalLayout

按钮信号和槽

字体

☐ 下划线

☐ 斜线

☐ 粗体

颜色

☐ 蓝色

☐ 红色

☐ 黑色

这是一段文字

取消

最大化

还原

发送者	信号	接收者	槽
pbresize	clicked()	Widget	showNormal()
pbok	clicked()	Widget	showFullScreen()
pbclose	clicked()	Widget	close()

案例说明

```
4  #include <QWidget>
5
6  namespace Ui {
7  class Widget;
8  }
9
10 class Widget : public QWidget
11 {
12     Q_OBJECT
13
14 public:
15     explicit Widget(QWidget *parent = nullptr);
16     ~Widget();
17
18 private slots:
19     void on_checkBoxxh_clicked(bool checked);
20
21     void on_checkBoxxx_clicked(bool checked);
22
23     void on_checkBoxct_clicked(bool checked);
24
25     void on_radoi_clicked();
26
27     void init();
28
29     void on_comboBox_currentIndexChanged(int index);
30
31 private:
32     Ui::Widget *ui;
33 };
34
```



Widget.h
定义的槽函数

案例说明

```
1 #include "widget.h"
2 #include "ui_widget.h"
3
4 Widget::Widget(QWidget *parent) :
5     QWidget(parent),
6     ui(new Ui::Widget)
7 {
8     ui->setupUi(this);
9     QObject::connect(ui->radioButtons, SIGNAL(clicked()), this, SLOT(on_radoi_clicked()));
10    QObject::connect(ui->radioButtons, SIGNAL(clicked()), this, SLOT(on_radoi_clicked()));
11    QObject::connect(ui->radioButtons, SIGNAL(clicked()), this, SLOT(on_radoi_clicked()));
12    init();
13 }
14
15 ~Widget()
16 {
17     delete ui;
18 }
19
20 void Widget::init()
21 {
22     ui->comboBox->addItem("14");
23     ui->comboBox->addItem("16");
24     ui->comboBox->addItem("18");
25 }
```

Widget.h
信号和槽的关联

单选框和函数关联

Combobox的下拉列表初始化

案例说明

```
void Widget::on_checkBoxxh_clicked(bool checked)
{
    QFont font = ui->plainTextEdit->font();
    font.setUnderline(checked);
    ui->plainTextEdit->setFont(font);
}
```

设置字体下滑线

```
void Widget::on_checkBoxxx_clicked(bool checked)
{
    QFont font = ui->plainTextEdit->font();
    font.setItalic(checked);
    ui->plainTextEdit->setFont(font);
}
```

设置字体斜线

```
void Widget::on_checkBoxct_clicked(bool checked)
{
    QFont font = ui->plainTextEdit->font();
    font.setBold(checked);
    ui->plainTextEdit->setFont(font);
}
```

设置字体粗体

案例说明

```
void Widget::on_radoi_clicked()
{
    QPalette palette = ui->plainTextEdit->palette();
    if(ui->radioButtons->isChecked()) {
        palette.setColor(QPalette::Text, Qt::red);
    } else if(ui->radioButtons->isChecked()) {
        palette.setColor(QPalette::Text, Qt::blue);
    } else {
        palette.setColor(QPalette::Text, Qt::black);
    }
    ui->plainTextEdit->setPalette(palette);
}
```

设置字体颜色

```
void Widget::on_comboBox_currentIndexChanged(int index)
{
    QFont font = ui->plainTextEdit->font();
    if(index == 1) {
        font.setPixelSize(14);
    } else if(index == 2) {
        font.setPixelSize(16);
    } else if(index == 3) {
        font.setPixelSize(18);
    } else {
        font.setPixelSize(14);
    }
    ui->plainTextEdit->setFont(font);
}
```

设置字体大小