2017

布局管理

目录
CONTENTE

1、 布局管理系统

- 1.1 简述
- 1.2 Qt布局类
- 1.3 布局管理器常见属性
- 1.4 其它因素

1.1、 简述

- Qt的布局系统提供了一个简单的和强有力的方式，来自动排列窗口子控件布局。
- 所有QWidget子类可以使用布局来管理他们的子控件。QWidget::setLayout()函数可以为一个控件布局。
- 当通过这种方式布局以后，它负责以下任务：

- 定位子部件；
- 感知窗口默认大小；
- 感知窗口最小大小；
- 改变大小处理；
- 当内容改变时自动更新：
  - 字体大小，文本或子部件的其他内容随之改变；
  - 隐藏或显示子部件；
  - 移除一个子部件。

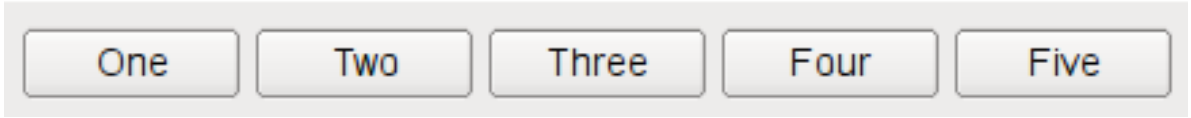
1.2、 Qt布局类

- QLayout类是布局管理器的基类，它是一个抽象基类。该类继承自QObject和QLayoutItem类，而QLayoutItem类提供了一个供QLayout操作的抽象项目。
- Qlayout子类：
  - QBoxLayout（基本布局管理器）
    - ✓ QHBoxLayout（水平布局管理器）
    - ✓ QVBoxLayout（垂直布局管理器）
  - QGridLayout（栅格布局管理器）
  - QFormLayout（窗体布局管理器）
  - QStackedLayout（栈布局管理器）

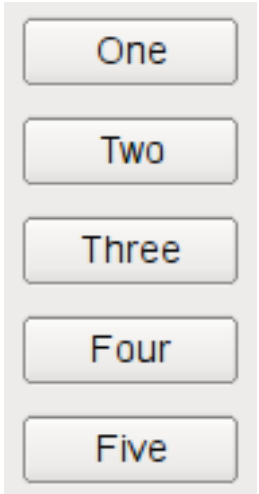
1.2、 Qt布局类

QBoxLayout（基本布局管理器）

- QHBoxLayout（水平）：把子窗口从左到右排列在一个水平行上。



- QVBoxLayout（垂直）：把子窗口从上到下排列在一个垂直列上

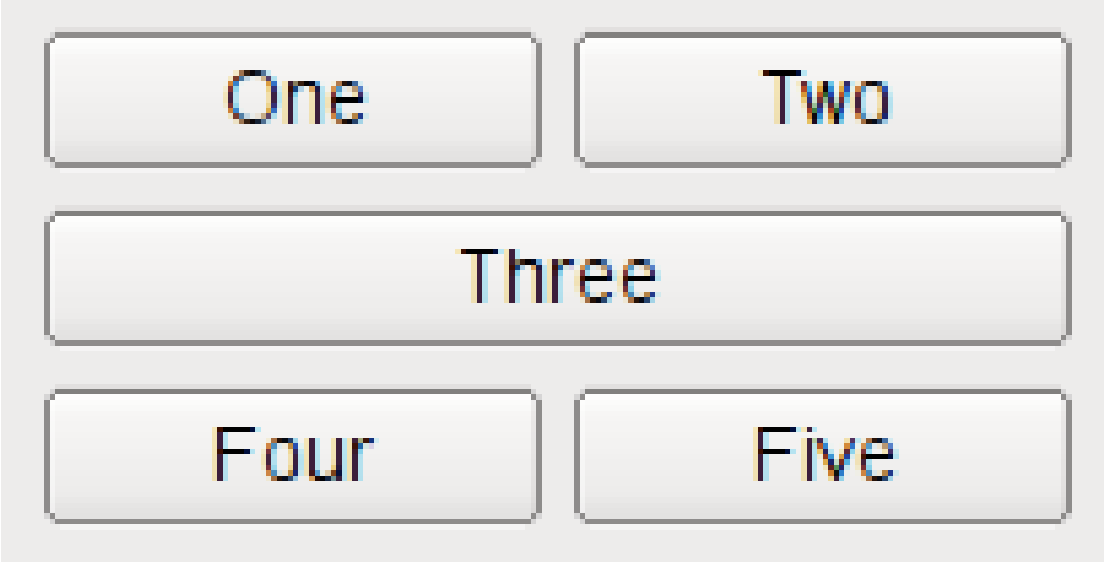


1.2、 Qt布局类

```
QWidget *window = new QWidget;

QPushButton *button1 = new QPushButton("One");
QPushButton *button2 = new QPushButton("Two");
QPushButton *button3 = new QPushButton("Three");
QPushButton *button4 = new QPushButton("Four");
QPushButton *button5 = new QPushButton("Five");

QHBoxLayout *layout = new QHBoxLayout;

layout->addWidget(button1);
layout->addWidget(button2);
layout->addWidget(button3);
layout->addWidget(button4);
layout->addWidget(button5);

window->setLayout(layout);

window->show();
```

1.2、 Qt布局类

QGridLayout（**栅格布局管理器**）：把子窗口排列在一个二维的网格中，窗口可占据多个单元格。



1.2、 Qt布局类

```
QGridLayout *layout = new QGridLayout;

layout->addWidget(button1, 0, 0);
layout->addWidget(button2, 0, 1);
layout->addWidget(button3, 1, 0, 1, 2);
layout->addWidget(button4, 2, 0);
layout->addWidget(button5, 2, 1);
```

QGridLayout不同之处需要指定子控件的行和列的位置。

1.2、 Qt布局类

QFormLayout（**窗体布局管理器**）：把子窗口按照标签-输入框的形式排列在两列。

QFormLayout将在一行上添加两个控件，通常是QLabel和QLineEdit。在同一行中添加QLabel和QLineEdit，将把QLineEdit设置为QLabel的伙伴。

1.2、 Qt布局类

```
QWidget *window = new QWidget;

QPushButton *button1 = new QPushButton("One");
QLineEdit *lineEdit1 = new QLineEdit();
QPushButton *button2 = new QPushButton("Two");
QLineEdit *lineEdit2 = new QLineEdit();
QPushButton *button3 = new QPushButton("Three");
QLineEdit *lineEdit3 = new QLineEdit();

QFormLayout *layout = new QFormLayout;

layout->addRow(button1, lineEdit1);
layout->addRow(button2, lineEdit2);
layout->addRow(button3, lineEdit3);

window->setLayout(layout);
window->show();
```

1.2、 Qt布局类

注意：

- 当使用布局的时候，构建子控件的时候不需要指定parent，布局将会自动的指定parent（使用QWidget::setParent()），使它们成为安装了该布局的界面的子控件。
- 也可以在布局中使用addLayout()来添加布局，内部的布局就会变成它的子布局（综合布局）。

1.3、布局管理器常见属性

属性	说明
layoutName	现在所使用的布局管理器的名称
layoutLeftMargin	设置布局管理器到界面左边界的距离
layoutTopMargin	设置布局管理器到界面上边界的距离
layoutRightMargin	设置布局管理器到界面右边界的距离
layoutBottomMargin	设置布局管理器到界面下边界的距离
layoutSpacing	布局管理器中各个子部件间的距离
layoutStretch	伸缩因子
layoutSizeConstraint	设置大小约束条件

1.3、布局管理器常见属性

```
layout->setSpacing(50); // 设置部件间的间隔

layout->setContentsMargins(0, 0, 50, 100); // 设置布局管理器到边界的距离，四个参数顺序是左，上，右，下

setLayout(layout);
```

1.4、 其它因素

当添加一个控件到一个布局中， 所有的控件将最初根据它们的QWidget::sizePolicy()和QWidget::sizeHint()而被分配到一定空间中。

1.4、 其它因素

部件大小：凡是继承自QWidget的类都有这两个属性：

- **大小提示（sizeHint）：保存了部件的建议大小，对于不同的部件，默认拥有不同的sizeHint，程序中使用sizeHint()函数来获取sizeHint的值；**
- **最小大小提示（minimumSizeHint）：保存了一个建议的最小大小。程序中可以使用minimumSizeHint()函数来获取minimumSizeHint的值 。**

如果使用minimumSize()函数设置了部件的最小大小，那么最小大小提示将会被忽略。

伸缩因子：它是用来设置部件间的比例的。在使用布局管理器的addWidget()函数添加部件的同时，在第二个参数中指定伸缩因子。

1.4、 其它因素

大小策略（sizePolicy）属性：

常量	描述
QSizePolicy::Fixed	只能使用 sizeHint()提供的值，无法伸缩
QSizePolicy::Minimum	sizeHint()提供的大小是最小的，部件可以被拉伸
QSizePolicy::Maximum	sizeHint()提供的是最大大小，部件可以被压缩
QSizePolicy::Preferred	sizeHint()提供的大小是最佳大小，部件可以被压缩或拉伸
QSizePolicy::Expanding	sizeHint()提供的是合适的大小，部件可以被压缩，不过它更倾向于被拉伸来获得更多的空间
QSizePolicy::MinimumExpanding	sizeHint()提供的大小是最小的，部件倾向于被拉伸来获取更多的空间
QSizePolicy::Ignored	sizeHint()的值被忽略，部件将尽可能的被拉伸来获取更多的空间

1.4、 其它因素

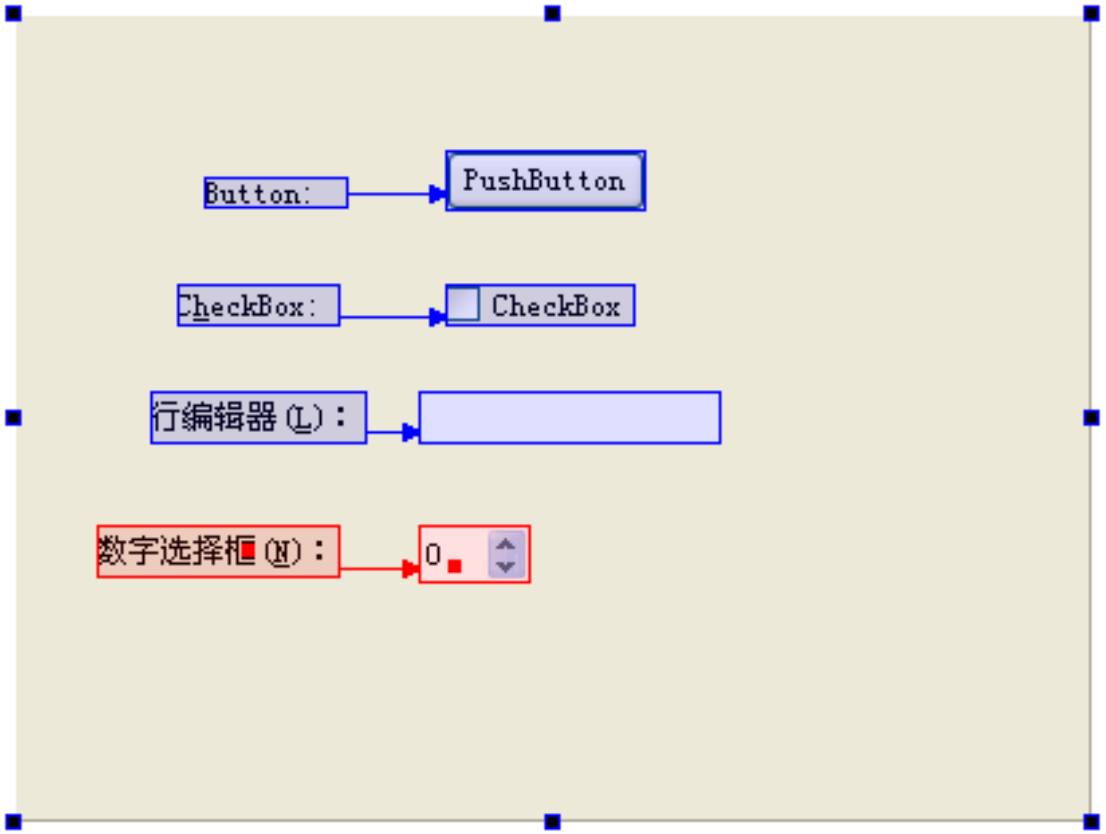
QWidget部件大小相关属性：

geometry	[(0, 0), 400 x 300]
X	0
Y	0
宽度	400
高度	300
sizePolicy	[Preferred, Preferred, 0, 0]
水平策略	Preferred
垂直策略	Preferred
水平伸展	0
垂直伸展	0
minimumSize	0 x 0
宽度	0
高度	0
maximumSize	16777215 x 16777215
宽度	16777215
高度	16777215
sizeIncrement	0 x 0
宽度	0
高度	0
baseSize	0 x 0
宽度	0
高度	0

- 高度与宽度属性，是现在界面的大小；
- sizePolicy属性可以设置大小策略以及伸缩因子；
- minimumSize属性用来设置最小大小；
- maximumSize属性设置最大大小；
- sizeIncrement属性和baseSize属性是设置窗口改变大小的，一般不用设置。

2、 设置伙伴

伙伴：QLable中提供助记符来设置焦点到对应的部件上。助记符即加速键。



3、 设置Tab键顺序

当程序启动时，焦点会在Tab键顺序为1的部件上。