


Програмування GUI


На основі мови C++ та фреймворку Qt

Лекція 6






Layouts



Лейаути - класи, що представляють собою менеджери компонування і призначені для управління автоматичним розміщенням графічних елементів графічного інтерфейсу користувача (віджетів) на поверхні вікна. Лейаути є невидимими контейнерами, які після зміни розмірів вікна автоматично приводять у відповідність координати і розміри віджетів, розташованих на ньому.



В Qt існують кілька класів-лейаутов: `QLayout` (абстрактний клас), успадковані від нього `QBoxLayout`, `QGridLayout`, `QStackedLayout`. Від класу `QBoxLayout` успадковані класи: `QHBoxLayout` і `QVBoxLayout`.



Клас QVBoxLayout



Дозволяє лінійно (по горизонталі або вертикалі) розміщувати віджети або інші лейаути.



Конструктор:


```
QVBoxLayout (Direction dir, QWidget *  
parent = 0),
```

```
(Enum Direction {LeftToRight,  
RightToLeft, TopToBottom, BottomToTop})
```


де `dir` задає розміщення віджетів і дочірніх лейаутов.




Клас QVBoxLayout

Додати віджет в лейаут можна методом

```
void addWidget ( QWidget * widget, int  
stretch=0, Qt::Alignment alignment=0),
```



де `widget` вказує на віджет, що додається, `stretch` задає фактор розтягування віджета, `alignment` задає розташування (вирівнювання) віджета в відведеної йому комірці лейаута, (наприклад, `Qt::AlignLeft`, `Qt::AlignHCenter`, за замовчуванням - розташування по центру).



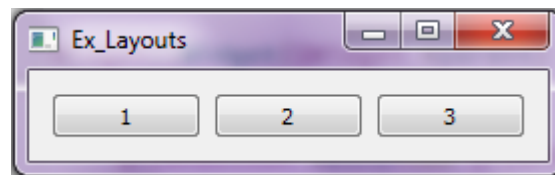
Клас QVBoxLayout



```
QPushButton *btn1=new QPushButton ("1",this);  
QPushButton *btn2=new QPushButton ("2",this);  
QPushButton *btn3=new QPushButton ("3",this);
```

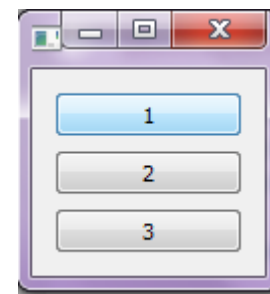
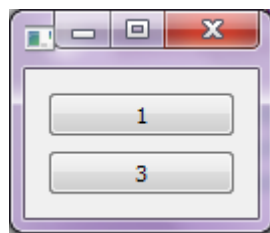
```
QBoxLayout *layout=new QBoxLayout  
(QBoxLayout::LeftToRight, this);
```

```
layout->addWidget(btn1);  
layout->addWidget(btn2);  
layout->addWidget(btn3);
```



```
layout->setDirection(QBoxLayout::TopToBottom);
```

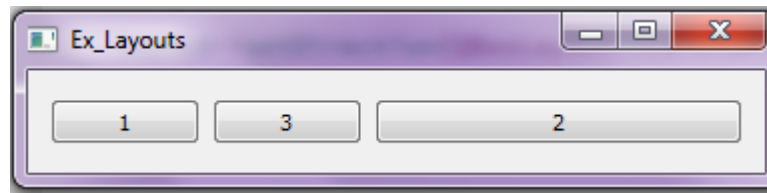
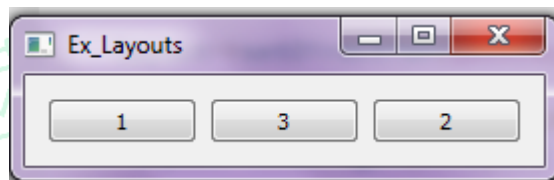
```
delete btn2;
```



Клас QVBoxLayout

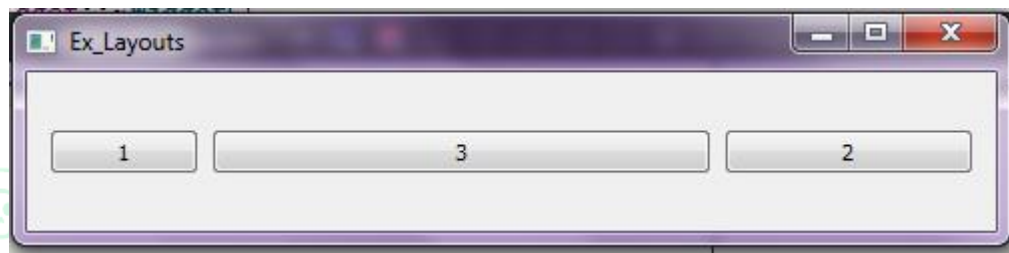


```
layout->setDirection(QBoxLayout::LeftToRight);  
btn2=new QPushButton ("2",this);  
layout->addWidget(btn2,1);
```



btn2 - розтягується по горизонталі, не розтягується по вертикалі

```
layout->setStretchFactor(btn3,2);
```



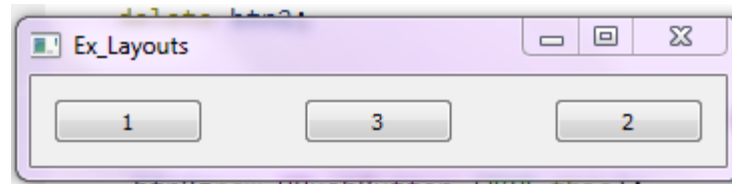
btn1 - не розтягується , btn3 – розтягується в 2 рази більше в порівнянні з btn2

Клас QVBoxLayout



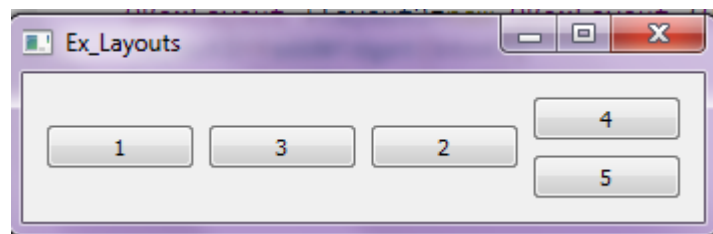
За допомогою методу `void setSpacing (int spacing)` можна задати мінімальну відстань між віджетами в пікселях.

```
layout->setSpacing(50);
```



За допомогою методу `void addLayout (QLayout *layout, int stretch=0)` можна додати дочірній лейаут в вихідний лейаут.

```
QPushButton *btn4=new QPushButton ("4",this);
QPushButton *btn5=new QPushButton ("5",this);
QVBoxLayout *layout2=new QVBoxLayout
(QVBoxLayout::TopToBottom); // без this, оскільки це
дочірній лейаут
layout2->addWidget(btn4);
layout2->addWidget(btn5);
layout->addLayout(layout2);
```



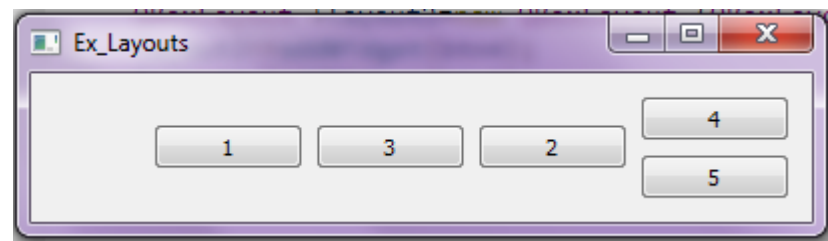
Клас QVBoxLayout



За допомогою методів `insertLayout` и `insertWidget` можна вставити лейаут і віджет в довільну комірку лейаута.

За допомогою методу `void insertSpacing(int index, int size)` можна вставити «шматок простору, що не розтягується» (розпірку) між віджетами, `index` задає номер віджета, перед яким слід вставка розпірки, `size`- ширину або висоту розпірки, в залежності від кількості елементів в лейауте.

```
layout->insertSpacing(0, 50)
```

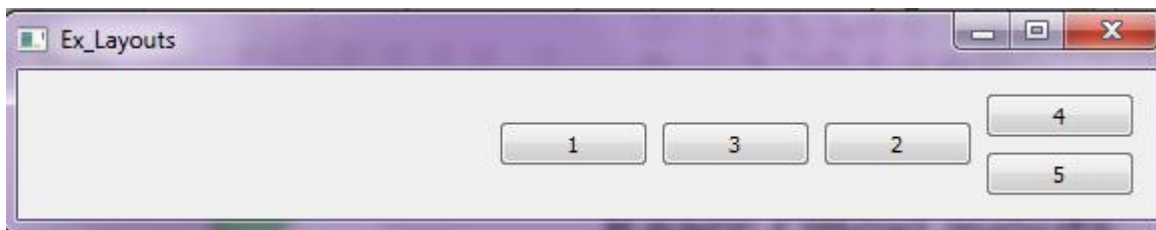


Клас QVBoxLayout



За допомогою методу `void insertStretch (int index, int stretch = 0)` можна вставити «шматок простору, що розтягується» (пружину) між віджетами, `index` задає номер віджета, перед яким буде вставлена пружина, `stretch` — фактор розтягування.

```
layout->insertStretch(1, 5);
```

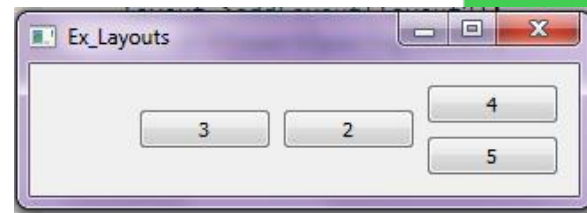


Клас QVBoxLayout

Qt

```
layout->removeWidget(btn1);
```

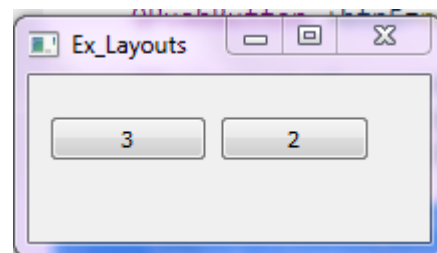
```
layout->removeItem(layout2);
```



Задати ширину кромки лейаута можна за допомогою методу

```
void setContentsMargins (int left,  
int top, int right, int bottom)
```


```
layout->setContentsMargins  
(10, 20, 30, 40);
```




Обмежити можливі розміри віджетів можна зокрема за допомогою таких методів класа QWidget:

- 1) setMinimum Width,
- 2) setMinimumHeight,
- 3) setMaximum Width,
- 4) setMaximumHeight,
- 5) setFixedWidth,
- 6) setFixedHeight.


Клас QVBoxLayout

A small icon showing a network of nodes connected by lines.

При додаванні лейаута в віджет головного вікна, він автоматично розміщується в центрі віджета і займає все його простір.

A small icon of a clipboard with a checkmark.

Для розміщення лейаута в довільному місці головного вікна і завдання потрібних розмірів лейаута необхідно буде створити додаткове вікно (дочірнє для головного), наприклад, на базі класу `QWidget`, розмістити в цьому вікні лейаут, після чого за допомогою методу `setGeometry ()` задати потрібне розташування і розміри дочірнього вікна і отже лейаута. Так робиться в програмі `Qt Designer`.

A small icon of a skull and crossbones.

Клас QVBoxLayout



Класи `QHBoxLayout` і `QVBoxLayout` повністю аналогічні класу `QBoxLayout`, за винятком того, що клас `QHBoxLayout` за замовчуванням розміщує віджети і лейаути тільки по горизонталі (зліва направо), а клас `QVBoxLayout` - тільки по вертикалі (згори вниз).



Клас QGridLayouts



Дозволяє розміщувати віджети і інші лейаути в «сітці», що складається з комірок, позиції яких задаються номерами рядків і стовпців (індексація з 0).

Конструктори:

`QGridLayout (QWidget *parent)` для головного лейаута

`QGridLayout ()` <--- для дочірнього лейаута .

Додати віджет в осередок лейаута можна за допомогою методу

`void addWidget (QWidget * widget, int row, int column, Qt:: : Alignment alignment=0)`, де `row` вказує номер рядка, а `column` - номер стовпця комірки.

Клас QGridLayouts

```
QGridLayout *grid=new QGridLayout (this);  
QPushButton *btn1=new QPushButton ("1",this);  
QPushButton *btn2=new QPushButton («2",this);  
QPushButton *btn3=new QPushButton («3",this);  
QPushButton *btn4=new QPushButton («4",this);  
QPushButton *btn5=new QPushButton («5",this);  
grid->addWidget (btn1,0,0);  
grid->addWidget (btn2,0,1);  
grid->addWidget (btn3,1,0);  
grid->addWidget (btn4,1,1);
```

