

# Програмування GUI

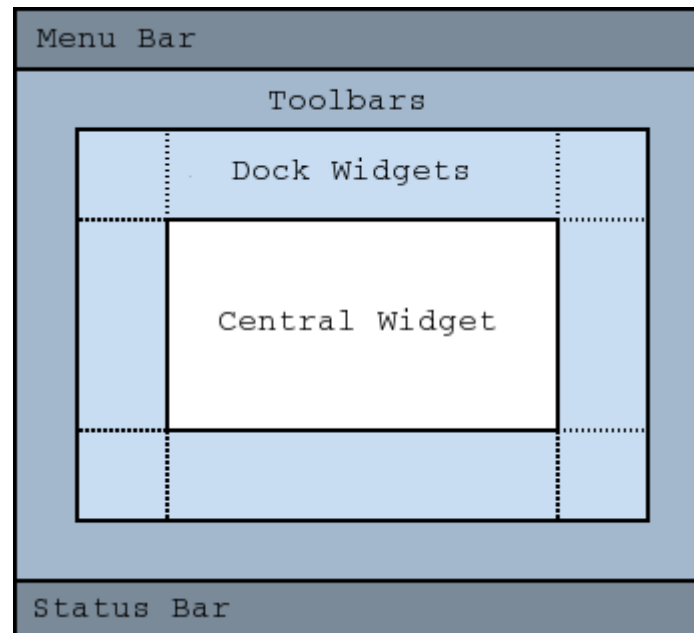
На основі мови C++ та фреймворку Qt

## Лекція 11

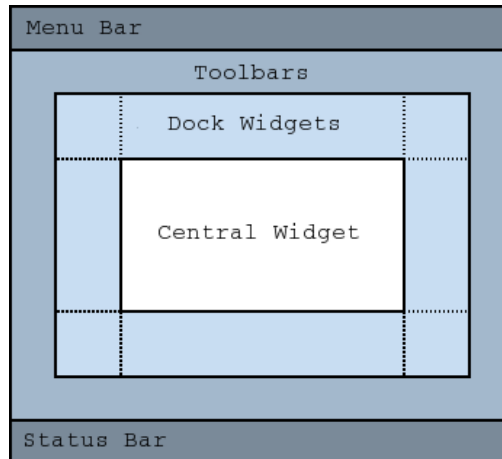
# Клас QMainWindow



Клас `QMainWindow` призначений для створення програм, що містять меню, панелі інструментів, рядок стану, які мають підтримку клавіш швидкого доступу. Успадкований від `QWidget`.



# Клас QMainWindow



**MenuBar** – область розміщення меню верхнього рівня, тобто головне меню.

**ToolBar** – область для розміщення панелей інструментів

**Dock Widgets** – область для розміщення вікон, створених на базі класу QDocWidget

**Central Widget** – область для розміщення головного дочірнього вікна програми

**StatusBar** – область для розміщення рядка стану програми

# Клас QMainWindow



## Конструктор:

`QMainWindow (QWidget *parent=0,  
Qt::WindowFlags flags=0),`


`parent` – покажчик на батьківський віджет, `flags` – параметри, що задають вигляд вікна (зазвичай `flags=0`).



У додатках, створених на основі класу `QMainWindow`, використовується клас `QAction`, який дозволяє суттєво прискорити розробку програми.

Об'єкти класу `QAction` спрощують обробку подій, що дублюються, пов'язаних з вибором пунктів меню, натискань кнопок панелей інструментів, введення комбінацій швидких клавіш і т.д.

# Клас QAction



 Клас `QAction` поєднує такі елементи інтерфейсу користувача:

1. Текст для пункту меню
-  2. Текст для підказки для кнопки на панелі інструментів
3. Комбінація швидких клавіш
-  4. Кнопки, що розміщуються на кнопках панелі інструментів та пунктах меню
5. І т.д.



# Клас QAction

Конструктор класа QAction:

```
QAction (const QIcon &icon, const  
QString &text, QObject *parent),
```

icon – посилання на кнопку, яка розміщуватиметься в пунктах меню та на кнопках панелі інструментів

text – задає текст для пункту меню і для підказки.

parent – покажчик на батьківський об'єкт (наприклад, головне вікно програми).

```
QAction *openAction=new QAction  
(QIcon ("open.png"), "&Open...", this);
```

```
QAction *saveAction = new QAction  
(QIcon ("save.png"),  
tr("&Сохранить..."), this);
```

# Методи класу QAction



1. Метод `void setShortcut (const QKeySequence *shortcut)` додає комбінацію швидких клавіш в об'єкт класу `QAction`  
`shortcut` – посилання на об'єкт класу `QKeySequence`, який задає клавіші комбінації

Конструктор класу `QKeySequence`:

`QKeySequence (const QString &key)`,  
де рядок `key` містить комбінацію клавіш, об'єднаних символом "+" («Ctrl+Q”).


Максимальна кількість клавіш у комбінації – 4 (наприклад, “Ctrl+A+B+C”).


`openAction–`

`>setShortcut (QKeySequence ("Ctrl+A")) ;`


# Методи класу QAction



 2. Метод `void setToolTip (const QString &tip)` – задає текст спливаючої підказки для кнопки на панелі керування.

 `tip` - задає текст підказки.

```
openAction->setToolTip(tr("Открыть  
файл"));
```

 3. Метод `void setStatusTip (const QString &status)` – дозволяє задати текст, який буде відображатися в рядку стану (як підказка) при наведенні курсору миші на відповідний пункт меню або кнопку на панелі інструментів.


```
openAction->setStatusTip("Open file  
from disk");
```






# Методи класу QAction




 При виборі пункту меню, натисканні кнопки на панелі інструментів або наборі комбінації швидких клавіш, пов'язаний з ними об'єкт класу `QAction` буде надсилати сигнал `triggered()`. Цей сигнал потрібно з'єднати з одним із слотом програми, який оброблятиме дані події.


```
connect (openAction, SIGNAL(triggered()),  
this, SLOT(open()));
```





# Меню



1. Створити меню верхнього рівня в головному вікні можна за допомогою функції класу `QMainWindow` `QMenuBar * menuBar()`, яка повертає покажчик на створене меню верхнього рівня.



2. Додати пункт меню до меню верхнього рівня можна за допомогою методу класу `QMenuBar` `QMenuBar * addMenu (const QString &title)`, який повертає покажчик на об'єкт класу `QMenuBar` – класу спливаючого (контекстного) меню; `title` – задає текст пункту меню верхнього рівня.



# Меню




3. Додати пункт у спливаюче меню можна за допомогою методу `QAction * (const QAction &action) .`


Пункт меню буде взято з об'єкту класу `QAction`.



```
QMenu *fileMenu = menuBar() -  
>addMenu("&File");  
fileMenu->addAction(openAction);  
fileMenu->addAction(saveAction);
```

# Панель інструментів



 1. Додати до панелі інструментів можна за допомогою методу класу `QMainWindow` `QToolBar` \* `addToolBar (const QString &title)`, який повертає покажчик на створену панель інструментів. `title` – задає ім'я панелі інструментів.

 Якщо навести курсор миші на панель інструментів або меню верхнього рівня і натиснути праву кнопку миші, то з'явиться віконце з ім'ям панелі інструментів і «галочкою», якщо панель видима. Знявши галочку, можна сховати панель інструментів, поставивши знову відобразити.



# Панель інструментів



2. Додати кнопку до панелі інструментів можна за допомогою методу класу `QToolBar`

```
void addAction (const QAction &action) .
```



```
QToolBar *fileToolBar;
```

```
fileToolBar=addToolBar("File Tool  
Bar");
```






```
fileToolBar->addAction(openAction);
```




# Рядок стану



 1. Створити рядок стану в головному вікні програми можна за допомогою методу класу QMainWindow `QStatusBar * statusBar()`, який повертає покажчик на створений рядок стану. У рядок стану можна додати віджет (наприклад, клас `QLabel`), який відображатиме інформацію у рядку стану.

```
QLabel statusLabel=new QLabel  
(tr("Ready"), this);  
statusBar()->addWidget(statusLabel);
```





# Методи класу QAction



За допомогою методу класу QAction void `setEnabled (bool)` можна зробити пункти меню та кнопки панелі інструментів доступними (`true`) або недоступними (`false`).


```
closeAction->setEnabled(false);  
// недоступен
```


За допомогою методу класу QAction void `setCheckable (bool)` можна дозволити кнопкам панелі інструментів і пунктам меню бути обраними (біля пунктів з'явиться галочка, якщо не використовується картинка), тобто працювати у режимі перемикача.

```
aboutAction->setCheckable(true);
```

# Ресурси



 Ресурси являють собою дані, які компілюються разом з текстом програми і поміщаються у файл \*.exe, що виконується. Як ресурси можна використовувати текст, картинки, звукові файли (\*.wav) та інших.


 Іконки для кнопок на панелі інструментів можна також використовувати як ресурси. У цьому випадку вони інтегруються у виконуваний файл програми, і для коректної роботи програми не потрібна наявність файлів іконок на жорсткому диску (у папці з файлом програми, що виконується).







# Ресурси



 Для використання ресурсів потрібно використовувати файл ресурсів. У Qt такий файл має розширення “qrc” (“\*.qrc”).

 Формат файлу заснований на XML, в якому перераховуються файли на диску і опціонально надає їм ім'я ресурсу, яке додаток має використовувати для доступу до ресурсу.

 Файл ресурсів є текстовим файлом та задає шлях до ресурсів. Можна створити вручну або засобами Visual Studio.



# Ресурси

A small icon representing an XML document, showing a tree structure with nodes and edges.

<!DOCTYPE RCC>

<RCC version="1.0">

<qresource prefix="/icons">

A small icon representing a clipboard, showing a sheet of paper with a checkmark.

<file> resources/open.png</file>

<file> resources/close.png</file>

<file> resources/save.png</file>

A small icon representing a computer monitor, showing a screen with a checkmark.

<file> resources/exit.png</file>

</qresource>

</RCC>

A small icon representing a skull and crossbones, indicating a warning or error.

# Ресурси



Далі такий файл разом із ресурсами компілюється компілятором ресурсів у файл із розширенням “\*.crr”, підключається до проекту програми. Якщо картинка є ресурсом, то шлях до неї має починатися з “:” та містити префікс ресурсу.

```
openAction=new QAction  
(QIcon(":/icons/resources/open.png"),  
tr("&Открыть..."),this);
```