

Програмування GUI

На основі мови C++ та фреймворку Qt

Лекція 9

Основи графіки в Qt



Растрові зображення

Растрові зображення є набором чисел, що задають колір пікселів. Qt підтримує такі формати зображень: BMP, GIF, PNG, JPEG (JPG), MNG і т.д.

BMP - BitMap (двовимірний масив бітів). BMP-формат не підтримує стиснення зображень та займає багато пам'яті.

GIF – Graphic Interchange Format (формат обміну графічними даними). Основні переваги – стиснення інформації без втрат, за рахунок застосування алгоритму стиснення LZW (Lempel-Ziv-Welch) та підтримка анімації. Основний недолік – ліцензійні відрахування використання алгоритму LZW.

Основи графіки в Qt



Растрові зображення



PNG – Portable Network Graphics (перенесена мережева графіка). Розроблено як безкоштовну альтернативу GIF. Підтримується стиск без втрат.

MNG – багато PNG. Зберігає серію зображень у форматі PNG. Є безкоштовною альтернативою для анімованих файлів GIF.

JPEG – Joint Photographic Experts Group. Характеризується дуже високим ступенем стиснення, але з втратою інформації.



Основи графіки в Qt



Способи представлення растрових зображень



1. Контекстно залежний (залежить від відеосистеми комп'ютера)
2. Контекстно-незалежний (не залежить від відеосистеми комп'ютера)



У разі контекстно-незалежного способу дані зображення поміщаються у звичайний масив, завдяки чому можна швидко зчитувати та записувати значення окремих пікселів зображення. Контекстно-залежний спосіб дозволяє швидше відображати зображення на екрані, проте операція зміни значень пікселів працює повільніше.



Клас QImage

Призначений для створення об'єкта контекстно-незалежного представлення растрових зображень.

Клас успадкований від класу QPainterDevice, що дозволяє малювати на об'єктах цього класу за допомогою об'єкта класу QPainter.

Конструктори:

```
QImage (int width, int height, Format format);
```

```
QImage (const QSize &size, Format format);
```


`width, height` – задають ширину та висоту зображення в пікселях,

`format` – задає формат зображення :


Клас QImage




Format:

 QImage::Format_Invalid – зображення недійсне (порожне)

 QImage::Format_Mono – чорно-біле зображення

 QImage::Format_RGB32 – 32 бита на піксель, біти непрозорі

 QImage::Format_ARGB32_Premultiplied – 4 байта на піксель+ прозорість

 `QImage img (300,200, QImage::Format_RGB32);`



Клас QImage

2. `QImage (const QString &filename, const char* format=0) .`

Завантажує зображення з файлу з ім'ям `filename` і створює на її основі об'єкт растрового зображення.

Параметр `format` задає формат файлу із зображенням ("jpg", "png", "bmp"). Якщо формат не заданий, конструктор спробує самостійно розпізнати графічний формат, зчитуючи заголовок файлу. Після завантаження зображення об'єкт зображення прийме розміри зображення.

```
QImage img1 ("image.jpg"); // текущий каталог
QImage img2 ("image.jpg", "jpg");
QImage img3 ("image", "jpg");
QImage img4 ("image");
QImage img5 ("image.jpg", "bmp"); // ошибка
```

Клас QImage

3. `QImage (const QImage &image) .`



Створює об'єкт растрового зображення на основі іншого об'єкта.



```
QImage img6 (img5) ;
```

4. `QImage ()` – конструктор за замовчанням.




```
QImage img7 ;
```



Методи класа QImage




 1. `bool load (const QString &fileName, const char *format=0)` – завантажує зображення з файлу.

`fileName` – ім'я файлу, `format` – формат файла.

У разі успішного завантаження повертає `true`, інакше `false`.

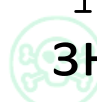
 `QImage image;`

`image.load("image.png");`

 2. `bool save (const QString &fileName, const char *format=0, int quality=-1)` – дозволяє зберегти зображення у файл.

`fileName` – ім'я файлу, `format` – формат файла,

`quality` – якість зображення, `quality[0,100]`. Чим більше

 значення – тим вища якість. `quality=-1` – налаштування за замовчуванням.

`image.save("image.jpg", "jpg", 0);`

`//сохранили как jpg`



Методи класа QImage



3. `QRgb pixel (int x, int y)` – зчитує значення пікселя зображення з координатами `x`, `y`.

4. `void setPixel (int x, int y, uint index_of_rgb)`

– дозволяє змінити колір пікселя з координатами `x`, `y`.

`index_of_rgb` – задає значення кольору у форматі `QRgb` або індекс кольору з палітри кольорів, якщо зображення має 8 біт на піксель.

```
QRgb color=img.pixel(0,0);  
color=color+10; //увеличили интенсивность синего цвета  
image.setPixel(0,0,color);
```

5. `int width()`, `int height()` – повертають ширину та висоту об'єкта зображення.

```
int width=image.width();  
int height=image.height();
```

Методи класа QImage



6. `void fill (uint pixel)` – дозволяє "залити" зображення кольором `pixel`.

```
QImage image1 (100,100,QImage::Format_RGB32);  
image1.fill(QColor(Qt::red).rgb());
```

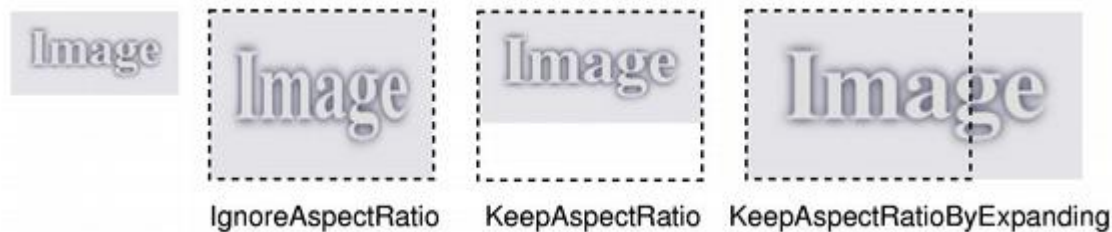
7. `QImage scaled (const QSize &size, Qt::AspectRatioMode aspectMode=Qt::IgnoreAspectRatio, Qt::TransformationMode transformationMode=Qt::FastTransformation)` – дозволяє масштабувати зображення (стискати або розтягувати) та повертає об'єкт відмасштабованого зображення, не змінюючи вихідний об'єкт. Параметр `size` – задає нові розміри зображення, `aspectMode` – задає тип масштабування, `transformationMode` – оптимизує процес масштабування за швидкістю або за пам'яттю, яка виділяється.

Методи класа QImage



aspectMode:

- Qt::IgnoreAspectRatio – масштабування без дотримання пропорцій
- Qt::KeepAspectRatio – масштабування з дотриманням пропорцій зображення поміститься в задану параметром size область
- Qt::KeepAspectRatioByExpanding – масштабування з дотриманням пропорцій, проте зображення може не поміститися в область, задану параметром size



Методи класа QImage



transformationMode:

- Qt::FastTransformation
- Qt::SmoothTransformation



```
void Widget::metod()
```

```
{
```



```
    QImage img("1.bmp");
```


```
    img=img.scaled(size());
```



```
}
```



Методи класа QImage



 8. `void invertPixels (InvertMode mode=InvertRgb)` – інвертує значення всіх пікселів зображення. Параметр `mode` вказує режим інвертування.
`mode`:

-  `QImage::InvertRgb` – інвертуються лише колірні інтенсивності (`r, g, b`) пікселів, прозорість не змінюється
-  `QImage::InvertRgba` – інвертуються колірні інтенсивності та прозорість.

```
QImage image(100,100, QImage::Format_RGB32);  
image.fill(QColor(Qt::black).rgb());  
// чорний квадрат  
image.invertPixels(); //теперь белый квадрат
```



Методи класа QImage




9. `QImage mirrored (bool horizontal=false, bool vertical=true)` – здійснює дзеркальне відображення зображення по горизонталі та/або вертикалі та повертає об'єкт із відображеним зображенням, не змінюючи вихідний. Для відображення по горизонталі параметр `horizontal` повинен мати значення `true`, по вертикалі - `vertical=true`.


```
QImage img ("image.jpg");  
QImage img2=img.mirrored(true,false);
```




Методи класа QImage



 Відобразити об'єкт зображення на поверхні віджету можна за допомогою методу `void drawImage (const QPoint &p, const QImage &image)` класа `QPainter`.


 `p` – задає координати верхнього лівого кута зображення на поверхні віджету, `image` – задає зображення.



```
void Widget::paintEvent(QPaintEvent *)  
{  
    QPainter painter;  
    painter.begin(this);  
    painter.drawImage(QPoint(100, 100), img);  
    painter.end();  
}
```



Клас QPixmap



Призначений для створення об'єкта контекстно-залежного представлення зображення. Клас успадкований від класу `QPaintDevice`, тому на об'єктах класу `QPixmap` можна малювати за допомогою малювальника `QPainter`.



Конструктори:

1. `QPixmap()` — за замовчанням
2. `QPixmap (const QSize &size);`
3. `QPixmap (const QPixmap &pixmap);`
4. `QPixmap (const QString &filename, const char* format=0, Qt::ImageConversionFlags flags=Qt::AutoColor)`

Клас QPixmap



flags:

`Qt::AutoColor` – автоматичний вибір кольорової гами

`Qt::ColorOnly` – перетворення до кольорового зображення

`Qt::MonoOnly` – перетворення до чорно-білого зображення

```
QPixmap pxm;
```

```
QPixmap pxm2 ( QSize ( 100 , 100 ) );
```

```
QPixmap pxm3 ( pxm2 );
```

```
QPixmap pxm4 ( "image.jpg" );
```

Методи класа QPixmap



1. `bool load(const QString &fileName, const char *format = 0, Qt::ImageConversionFlags flags = Qt::AutoColor)` – завантаження зображення в об'єкт класу QPixmap.

`pxm.load("image.bmp", "bmp", Qt::MonoOnly);`

2. `void fill(const QColor &color = Qt::white)` – заливка зображення.

`pxm.fill(Qt::blue);`

3. `int width(), int height()` – Розміри зображення.

4. `bool save(const QString &fileName, const char *format = nullptr, int quality = -1)` – збереження зображення.

`pxm.save("3.jpg", "jpg");`


Методи класа QPixmap

5. QPixmap scaled(const QSize &size, Qt::AspectRatioMode aspectRatioMode = Qt::IgnoreAspectRatio, Qt::TransformationMode transformMode = Qt::FastTransformation), QPixmap scaled(int width, int height, Qt::AspectRatioMode aspectRatioMode = Qt::IgnoreAspectRatio, Qt::TransformationMode transformMode = Qt::FastTransformation) – масштабує за аналогією з класом QImage


6. QImage toImage() – дозволяє перетворити об'єкт класу QPixmap в об'єкт класу QImage.

```
QImage image;  
image=pxm.toImage();
```


Методи класа QPixmap



```
7. QPixmap fromImage(const QImage &img,  
Qt::ImageConversionFlags
```



flags=Qt::AutoColor) – статичний метод, який виконує перетворення з QImage на QPixmap.



img – посилання на об'єкт класа QImage, flags – контролює перетворення зображення (If the image needs to be modified to fit in a lower-resolution result (e.g. converting from 32-bit to 8-bit))

```
QPixmap pixmap;
```






```
pixmap=QPixmap::fromImage(image);
```



Методи класа QPixmap



 8. QPixmap grabWidget(QWidget *widget, int x=0, int y=0, int w=-1, int h=-1) – статичний метод, який повертає копію зображення клієнтської області widget віджету. x, y и w, h – координати та розміри прямокутної області віджету, що копіюватиметься. За промовчанням копіюється вся клієнтська область.


`QPixmap pxm_grab;`
`pxm_grab=QPixmap::grabWidget(this);`
 `pxm_grab.save("1.bmp", "bmp");`

`pxm_grab=QWidget::grab();`



Методи класа QPixmap



9. `void drawPixmap (const QPoint &p, const QPixmap &pm)` – відображає зображення класу QPixmap на поверхні віджету.

```
void Widget::paintEvent(QPaintEvent *)  
{  
    QPainter painter(this);  
    painter.begin(this);  
    painter.drawPixmap(QPoint(100,100), pxm);  
    painter.end();  
}
```

