•10.2 绘图设备

绘图设备是指继承 QPainterDevice 的子类。Qt 一共提供了四个这样的类,分别

是 QPixmap、QBitmap、QImage 和 QPicture。其中, 4

- QPixmap 专门为图像在屏幕上的显示做了优化→
- QBitmap 是 QPixmap 的一个子类,它的色深限定为 1,可以使用 QPixmap 的 isQBitmap()函数来确定这个 QPixmap 是不是一个 QBitmap。↓
- QImage 专门为图像的像素级访问做了优化。 →
- QPicture 则可以记录和重现 QPainter 的各条命令。→

举例:

下面我们来看同一个图像文件在 QPixmap 和 QBitmap 下的不同表现: ₽

```
void PaintWidget::paintEvent (QPaintEvent *) +

{+

QPixmap pixmap(":/Image/butterfly.png");+

QPixmap pixmap1(":/Image/butterfly1.png");+
```

```
QBitmap bitmap(":/Image/butterfly.png");

QBitmap bitmap1(":/Image/butterfly1.png");

QPainter painter(this);

painter drawPixmap(0, 0, pixmap);

painter drawPixmap(200, 0, pixmap1);

painter drawPixmap(0, 130, bitmap);

painter drawPixmap(200, 130, bitmap1);

painter drawPixmap(200, 130, bitmap1);

}
```



● QPicture 则可以记录和重现 QPainter 的各条命令。J →

QPixmap

```
#include "widget.h"
#include "ui_widget.h"
#include<QPainter>
#include<QPixmap>
Widget::Widget(QWidget *parent)
    : QWidget(parent)
   , ui(new Ui::Widget)
{
   ui->setupUi(this);
   QPixmap pix(300,300);//绘图设备
   QPainter painter(&pix);//声明画家
   painter.setPen(QPen(Qt::green));//填充绿色
    painter.drawEllipse(QPoint(150,150),100,100);//画一个圆
   pix.save("D:/QPixmap/pix.png");
}
Widget::~Widget()
   delete ui;
}
```

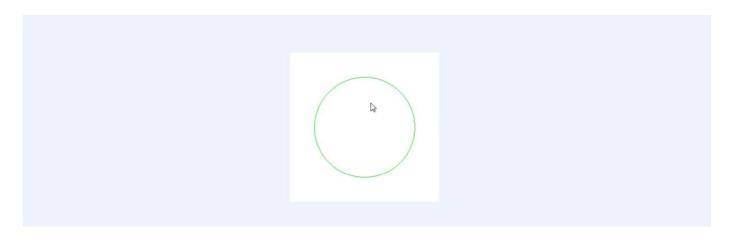
此时, 窗口中并不会画出我们需要的, 但本地可以查看到



填充颜色

```
QPixmap pix(300,300);//绘图设备
    pix.fill(Qt::white);
    QPainter painter(&pix);//声明画家
    painter.setPen(QPen(Qt::green));//填充绿色
    painter.drawEllipse(QPoint(150,150),100,100);//画一个圆
    pix.save("D:/QPixmap/pix.png");
```

本地看到新的图片



Qlmage

```
QImage
                                                                                                                        The image is invalid.
                                           QImage::Format Invalid
QImage
QImage::Format
QImage::Format_A2BGR30_Premult***
                                           QImage::Format_Mono
                                                                                                               1
                                                                                                                         The image is stored using 1-bit per pixel. Bytes
                                                                                                                        first.
QImage::Format_A2RGB30_Premult...
QImage::Format_Alpha8
                                                                                                                         The image is stored using 1-bit per pixel. Bytes
                                           QImage::Format_MonoLSB
                                                                                                               2
QImage::Format_ARGB32
QImage::Format_ARGB32_Premulti...
QImage::Format_ARGB4444_Premul...
                                                                                                                        first.
                                                                                                                        The image is stored using 8-bit indexes into a co
                                           QImage::Format_Indexed8
                                                                                                               3
QImage::Format_ARGB8666_Premul...
QImage::Format_ARGB8555_Premul...
                                                                                                                        The image is stored using a 32-bit RGB format (Ox
QImage::Format_ARGB8565_Premul...
QImage::Format_BGR30
                                           QImage::Format_RGB32
                                                                                                               4
QImage::Format_Grayscale8
QImage::Format_Indexed8
                                           QImage::Format_ARGB32
                                                                                                                        The image is stored using a 32-bit ARGB format (C
QImage::Format_Invalid
QImage::Format_Mono
                                                                                                                         The image is stored using a premultiplied 32-bit
                                           QImage::Format_ARGB32_Premultiplied
                                                                                                              6
```

接下来一样的步骤

```
QImage img(300,300,QImage::Format_ARGB32);//绘图设备
img.fill(Qt::white);
    QPainter painter2(&img);
```

```
painter2.setPen(QPen(Qt::cyan));
painter2.drawEllipse(QPoint(150,150),100,100);
img.save("D:/QPixmap/img.png");
```

本地查看



QImage比QPixmap多了一个功能:对像素点进行访问

如何证明这件事呢?

我们在窗口中画一个图片

重写绘图事件的函数

widget.cpp

```
void Widget::paintEvent(QPaintEvent *)
{
    QPainter painter(this);
    //利用QImage修改像素
    QImage img;
    img.load(":/Image/Luffy.png");
    painter.drawImage(0,0,img);
}
```

```
delete 💷 v
          29
          30
                }
          31
              void Widget
          32
                {
                     QPainte
          33
                     //利用Q:
          34
erfly1.png
          35
                     QImage
n.png
          36
                     img.loa
ne.jpg
          37
                     painter
/Q.png
          38
          39
```

现在使用QImage修改像素点

```
void Widget::paintEvent(QPaintEvent *)
{
         QPainter painter(this);
```

```
//利用QImage img;
img.load(":/Image/Luffy.png");

for(int i=20;i<220;i++)
{
    for(int j=10;j<50;j++)
    {
        QRgb value=qRgb(0,255,0);
        img.setPixel(i,j,value);//改变像素点的颜色为绿色
    }
}

painter.drawImage(0,0,img);//绘图
}
```

运行



QPicture

用于记录和重现绘图的指令

可以看到提示指定绘图设备,我们一开始可以指定,也可以不指定如果不指定,那我们先创建。

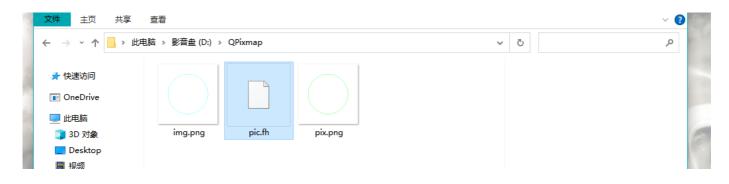
QPainter painter;

然后使用begin(0;指定绘图设备,end();结束

```
QPicture pic;//绘图设备,用于记录和重现绘图指令
QPainter painter;//一开始不指定设备
painter.begin(&pic);//用 .begin() 指定绘图设备
painter.setPen(QPen(Qt::cyan));
painter.drawEllipse(QPoint(150,150),100,100);
painter.end();//结束画画
pic.save("D:/QPixmap/pic.fh");
```

后缀名是我们自定义的

本地得到文件, 虽然打不开, 但它确确实实记录了我们画图的步骤



现在, 我们来重现绘图指令

```
QPainter painter(this);//重现
      QPicture pic;
      pic.load("D:/QPixmap/pic.fh");
      painter.drawPicture(0,0,pic);
54
   //
              painter.end();//结束画画
55
              pic.save("D:/QPixmap/pic
    1//
56
57
            QPainter painter(this);//1
58
            QPicture pic;
            pic.load("D:/QPixmap/pic.f
59
60
            painter.drawPicture(0,0,pi
61
62
63
```

总结:

```
QPaintDevice绘图设备
1.1
     QPixmap QImage QBitmap(黑白色) QPicture QWidget
     QPixmap 对不同平台做了显示的优化
1.2
1.2.1
       QPixmap pix( 300,300)
1.2.2
       pix.fill(填充颜色)
      利用画家 往pix上画画 QPainter painter( & pix)
1.2.3
1.2.4 保存 pix.save("路径")
     Qimage 可以对像素进行访问
1.3
       使用和QPixmap差不多 QImage img(300,300,QImage::Format_RGB32);
1.3.1
1.3.2
       其他流程和QPixmap一样
1.3.3
       可以对像素进行修改 img.setPixel(i,j,value);
1.4
   QPicture 记录和重现 绘图指令
1.4.1
     QPicture pic
1.4.2
       painter.begin(&pic);
1.4.3
     保存 pic.save( 任意后缀名 )
       重现 利用画家可以重现painter.drawPicture(0,0,pic);
1.4.4
```