25 绘制事件

25.1 处理绘制事件

当有下列情况之一发生时，将触发窗口的绘制事件，届时窗口的paintEvent虚函数会被调用：

- 窗口创建后首次显示
- 窗口由隐藏变为可见
- 窗口由最小化变为正常或最大化
- 窗口因大小改变需要呈现更多内容
- 人为调用窗口的update或repaint方法

程序设计者可以通过覆盖基类的paintEvent虚函数，自己决定在窗口中绘制的内容。

25.2 画家类

Qt提供了名为QPainter的画家类，用于实现二维图形图像的绘制和渲染。在paintEvent虚函数的覆盖版本中，可以借助QPainter类的对象，完成在窗口中绘制图形或渲染图像的操作。

25.3 案例

25.3.1 创建项目

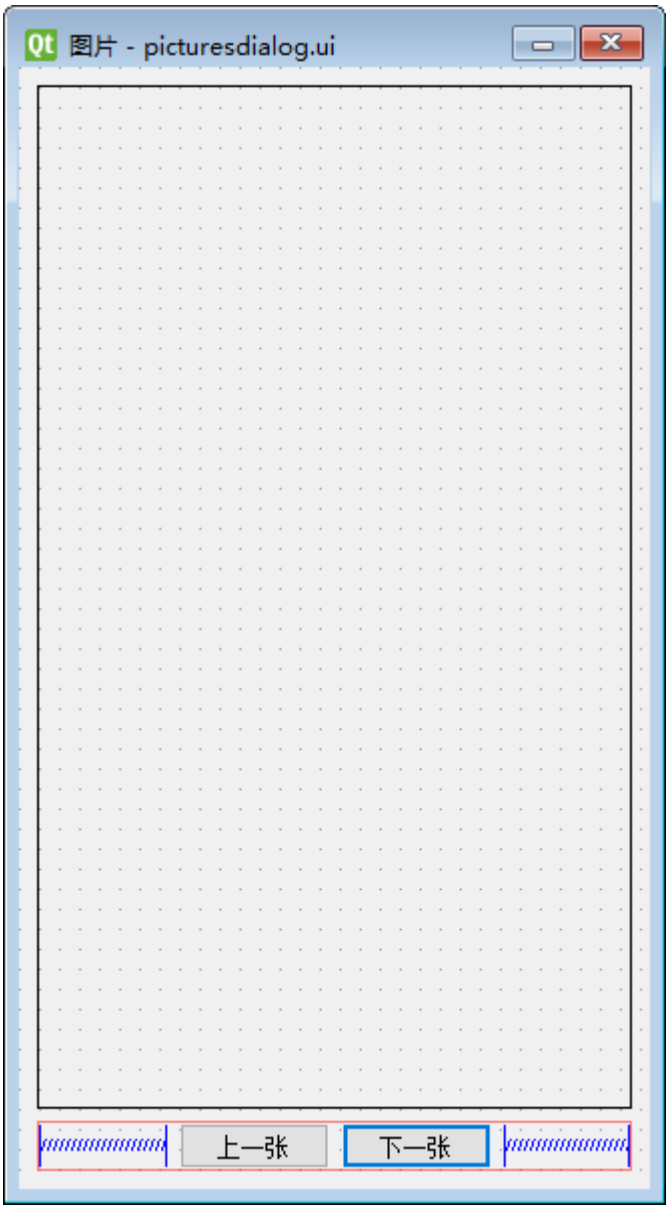
通过QtCreator，在C:\Users\Minwei\Projects\Qt路径下，创建名为Pictures的项目。

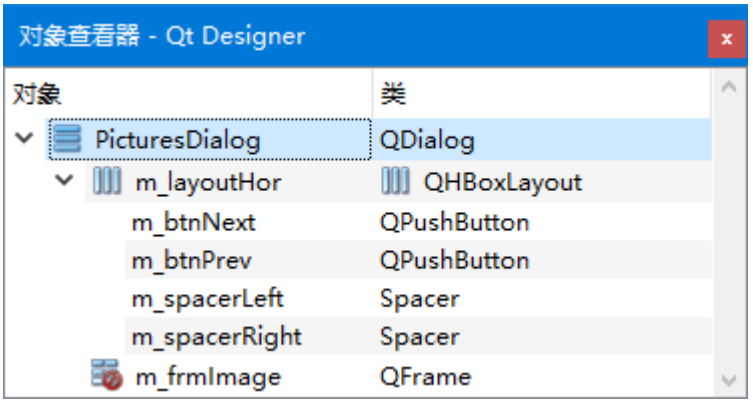
25.3.2 添加资源

C:\Users\Minwei\Projects\Qt\Pictures\Pictures.qrc：

```
<RCC>
    <qresource prefix="/">
        <file>images/0.jpg</file>
        <file>images/1.jpg</file>
        <file>images/2.jpg</file>
        <file>images/3.jpg</file>
        <file>images/4.jpg</file>
        <file>images/5.jpg</file>
        <file>images/6.jpg</file>
        <file>images/7.jpg</file>
    </qresource>
</RCC>
```

25.3.3 设计界面





C:\Users\Minwei\Projects\Qt\Pictures\picturesdialog.ui：

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>PicturesDialog</class>
 <widget class="QDialog" name="PicturesDialog">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
```

```
    <width>315</width>
    <height>560</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>图片</string>
  </property>
  <layout class="QVBoxLayout" name="m_layoutVer">
   <item>
    <widget class="QFrame" name="m_frmImage">
     <property name="sizePolicy">
      <sizepolicy hsizetype="Preferred" vsizetype="Expanding">
       <horstretch>0</horstretch>
       <verstretch>0</verstretch>
      </sizepolicy>
     </property>
     <property name="frameShape">
      <enum>QFrame::Box</enum>
     </property>
     <property name="frameShadow">
      <enum>QFrame::Plain</enum>
     </property>
     <property name="lineWidth">
      <number>1</number>
     </property>
    </widget>
   </item>
   <item>
    <layout class="QHBoxLayout" name="m_layoutHor">
     <item>
      <spacer name="m_spacerLeft">
       <property name="orientation">
        <enum>Qt::Horizontal</enum>
       </property>
       <property name="sizeHint" stdset="0">
        <size>
         <width>40</width>
         <height>20</height>
        </size>
       </property>
      </spacer>
     </item>
     <item>
      <widget class="QPushButton" name="m_btnPrev">
       <property name="text">
        <string>上一张</string>
       </property>
      </widget>
     </item>
     <item>
      <widget class="QPushButton" name="m_btnNext">
       <property name="text">
        <string>下一张</string>
       </property>
       <property name="default">
        <bool>true</bool>
```

```
       </property>
      </widget>
     </item>
     <item>
      <spacer name="m_spacerRight">
       <property name="orientation">
        <enum>Qt::Horizontal</enum>
       </property>
       <property name="sizeHint" stdset="0">
        <size>
         <width>40</width>
         <height>20</height>
        </size>
       </property>
      </spacer>
     </item>
    </layout>
   </item>
  </layout>
 </widget>
 <resources/>
 <connections/>
</ui>
```

25.3.4 实现功能

C:\Users\Minwei\Projects\Qt\Pictures\picturesdialog.h：

```
#ifndef PICTURESDIALOG_H
#define PICTURESDIALOG_H

#include <QDialog>

QT_BEGIN_NAMESPACE
namespace Ui { class PicturesDialog; }
QT_END_NAMESPACE

class PicturesDialog : public QDialog
{
    Q_OBJECT

public:
    PicturesDialog(QWidget *parent = nullptr);
    ~PicturesDialog();

protected:
    void paintEvent(QPaintEvent*);

private slots:
    void on_m_btnPrev_clicked();
    void on_m_btnNext_clicked();

private:
    void enableButtons(void);

```

```
private:
    Ui::PicturesDialog *ui;
    int m_imageIndex;
};

#endif // PICTURESDIALOG_H
```

C:\Users\Minwei\Projects\Qt\Pictures\picturesdialog.cpp:

```
#include <QPainter>

#include "picturesdialog.h"
#include "ui_picturesdialog.h"

PicturesDialog::PicturesDialog(QWidget *parent)
    : QDialog(parent)
    , ui(new Ui::PicturesDialog)
    , m_imageIndex(0)
{
    ui->setupUi(this);

    enableButtons();
}

PicturesDialog::~PicturesDialog()
{
    delete ui;
}

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

    QRect frameRect = ui->m_frmImage->frameRect();
    frameRect.translate(ui->m_frmImage->pos());

    QImage image(":/images/" + QString::number(m_imageIndex) + ".jpg");
    painter.drawImage(frameRect, image);
}

void PicturesDialog::on_m_btnPrev_clicked()
{
    --m_imageIndex;

    enableButtons();
    update();
}

void PicturesDialog::on_m_btnNext_clicked()
{
    ++m_imageIndex;

    enableButtons();
    update();
}
```

```
47
48  void PicturesDialog::enableButtons(void)
49  {
50      ui->m_btnPrev->setEnabled(m_imageIndex != 0);
51      ui->m_btnNext->setEnabled(m_imageIndex != 7);
52  }
```

25.3.5 测试验证

运行效果如图所示：