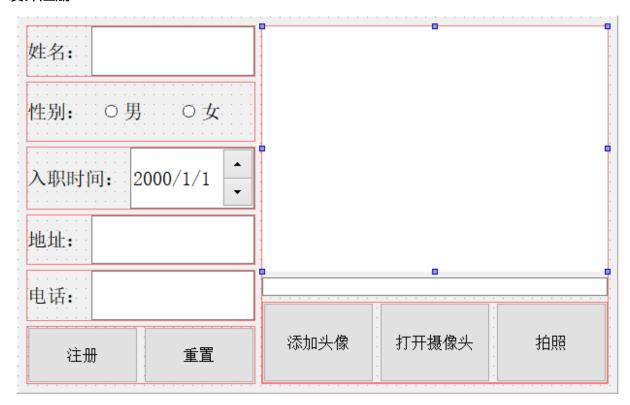
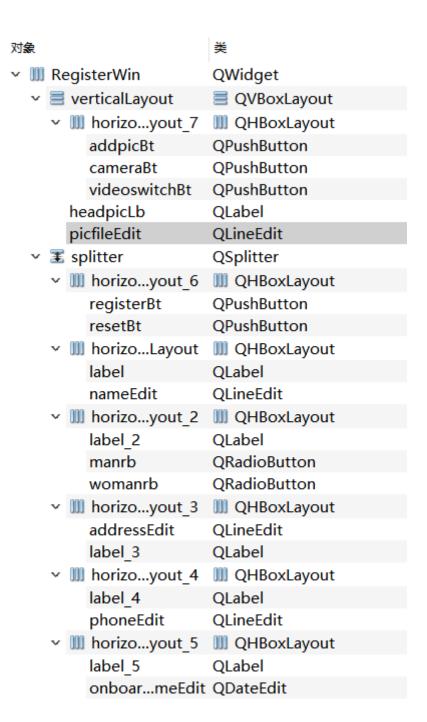
# 10.员工信息注册

# 设计注册界面

- 1. 在客户端工程文件中新建Qt设计器界面类
- 2. 设计注册ui





- 3. 对其中的按钮设置功能,右键->转到槽->QAbstractButton/clicked()
- 4. 重置:

```
void RegisterWin::on_resetBt_clicked()
{
    //清空数据
    ui → nameEdit → clear();
    ui → onboardingtimeEdit → setDate(QDate::currentDate()); //重置为当天日期
    ui → addressEdit → clear();
    ui → phoneEdit → clear();
    ui → picfileEdit → clear();
}
```

#### 5. 添加头像:

#### 6. 打开摄像头,使用摄像头捕获数据的操作需要在定时器中进行。

```
/*在.h文件中创建定时器函数、flag和摄像头捕获*/
void timerEvent(QTimerEvent *e);// 定时器函数
int timerid; // 定时器flag
cv::VideoCapture cap;
cv::Mat image; // 全局变量在多个函数中处理
/* 定时器函数 */
void RegisterWin::timerEvent(QTimerEvent *e)
{
   // 获取摄像头数据并且显示在界面上
   if(cap.isOpened()){
       cap>>image;
       if(image.data == nullptr) return;
       // Mat→ QImage
       cv::Mat rgbimage;
       cv::cvtColor(image, rgbimage,cv::COLOR_BGR2RGB);
       QImage qimg(rgbimage.data, rgbimage.cols, rgbimage.rows,
rgbimage.step1(), QImage::Format_RGB888);
       QPixmap mmp = QPixmap::fromImage(qimg);
       mmp = mmp.scaledToWidth(ui-headpicLb-width());
       ui→headpicLb→setPixmap(mmp);
   }
}
/* 按钮函数 */
void RegisterWin::on_videoswitchBt_clicked()
```

```
{
   if(ui→videoswitchBt→text() == "打开摄像头"){
       // 打开摄像头
       if(cap.open(0)){
          ui→videoswitchBt→setText("关闭摄像头");
          // 启动定时器事件
          timerid = startTimer(100);
       }
   }
   else{
       ui→videoswitchBt→setText("打开摄像头");
       // 关闭定时器事件
       killTimer(timerid);
       // 关闭摄像头
       cap.release();
   }
}
```

### 7. 拍照

```
void RegisterWin::on_cameraBt_clicked()
{
    // 保存数据
    QString headfile = QString("./data/%1.jpg").arg(ui→nameEdit-
>text().toUtf8().Base64Encoding);
    ui→picfileEdit→setText(headfile);//显示地址
    cv::imwrite(headfile.toUtf8().data(),image);
    ui→videoswitchBt→setText("打开摄像头");
    // 关闭定时器事件
    killTimer(timerid);
    // 关闭摄像头
    cap.release();
}
```

### 1. 注册

```
void RegisterWin::on_registerBt_clicked()
{

    // 1.通过照片结合faceObject模块得到faceID
    QFaceObject faceobj;// 步骤10 需要将图片添加到face.db库中
    cv::Mat image = cv::imread(ui→picfileEdit→text().toUtf8().data());
    int faceID = faceobj.face_register(image);
```

```
//把头像保存在固定路径下
   OString headfile = OString("./data/%1.jpg").arg(ui→nameEdit-
>text().toUtf8().Base64Encoding);
   cv::imwrite(headfile.toUtf8().data(),image);
   // 2.把个人信息存储到数据库employee
   QSqlTableModel model;
   model.setTable("employee");//设置表名
   QSqlRecord record = model.record();
   //设置数据
   record.setValue("name",ui→nameEdit→text());
   record.setValue("sex",ui→manrb→isChecked()?"男":"女");
   record.setValue("onboardingTime",ui→onboardingtimeEdit→text());
   record.setValue("address",ui→addressEdit→text());
   record.setValue("phone",ui→phoneEdit→text());
   record.setValue("faceID",faceID);
   record.setValue("headfile", headfile);
   // 把记录插入到数据库表格中
   bool ret = model.insertRecord(0,record);
   // 3.提示注册成功
   if(ret)
   {
       // 提交
       model.submitAll();
       QMessageBox::information(nullptr, "注册提示","注册成功");
   }else
   {
       QMessageBox::information(nullptr, "注册提示","注册失败");
   }
}
```

9. 当工程中有多个ui界面时,需要在 main 函数中添加头文件,并显示 #include "registerwin.h"
RegisterWin ww; www.show();`

10. 注册后,注册成功的照片应该存入 face.db 中,方便再次存入后得到引得faceID,在QFaceObject生成函数中添加 this→fengineptr→Load("./face.db");