1. QMainWindow
   1. 菜单栏 只能有一个
      1. QMenuBar\* bar = menuBar() 创建
      2. setMenuBar(bar) ;设置到窗口中
      3. QMenu\* menu = bar->addMenu(“文件”) 添加菜单
      4. QAction \* newAction = menu->addAction(“新建”); 添加菜单项
      5. 添加分割线fileMenu->addSeparator();
   2. 工具栏 可以有多个
      1. QToolBar\* toolbar = new QToolBar();
      2. addToolBar( 默认停靠范围 Qt::LeftToolBarArea ,toolbar) 添加到窗口中
      3. 设置 只允许左右停靠、 setAllowedAreas
      4. 设置浮动、 setFloatable
      5. 设置移动 setMovable
   3. 状态栏 只能有一个
      1. QStatusBar \* stBar = statusBar();
      2. setStatusBar(stBar);
      3. stBar->addWidget(label1); 左侧信息添加
      4. stBar->addPermanentWidget(label2); 右侧信息添加
   4. 铆接部件 可以有多个
      1. QDockWidget \* dockWidget = new QDockWidget;
      2. addDockWidget(Qt::BottomDockWidgetArea,dock); 添加到窗口中
      3. dock->setAllowedAreas(Qt::LeftDockWidgetArea | Qt::RightDockWidgetArea); 设置后期停靠范围
   5. 核心部件 只能有一个
      1. setCentralWidget(edit);
2. 资源文件添加
   1. 将资源文件放入到项目下
   2. 右键项目->添加新文件 -> Qt ->Qt Recourse File
   3. 给资源文件起名 res 生成 res.qrc
   4. 用编辑的方式打开 res.qrc
   5. 添加前置 - 添加文件
   6. 使用 “ : + 前缀名 + 文件名 ”
3. 对话框创建
   1. 分类：模态对话框 和非模态对话框
      1. 模态 不可以对其他窗口进行操作
      2. 非模态 可以对其他窗口进行操作
      3. 模态对话框创建
         1. QDialog dlg(this);
         2. dlg.exec();
      4. 非模态对话框创建
         1. QDialog \* dlg2 = new QDialog(this);
         2. dlg2->show();
         3. 设置属性 dlg2->setAttribute(Qt::WA\_DeleteOnClose);
   2. 标准对话框 QMessageBox
      1. 利用静态成员函数可以弹出 警告、询问、信息、错误对话框
      2. 参数1 父窗口 参数2 标题 参数3 提示信息 参数4 按键类型 参数5 默认关联回车按键
      3. QMessageBox::Save == QMessageBox::question(this,"询问","question" , QMessageBox::Save | QMessageBox::Cancel ,QMessageBox::Cancel)
      4. 返回值代表用户选择按钮
   3. 其他常用标准对话框
      1. 颜色对话框
         1. QColor color = QColorDialog::getColor(Qt::red);
      2. 文件对话框
         1. QString fileName = QFileDialog::getOpenFileName(this,"打开文件","C:\\Users\\zhangtao\\Desktop","(\*.doc)");
      3. 字体对话框
         1. QFont font = QFontDialog::getFont(&flag,QFont("华文彩云",36));
         2. qDebug() << "字体" << font.family().toUtf8().data() << "字号"<< font.pointSize()<< "是否加粗"<<font.bold() << "是否倾斜" << font.italic();
4. 界面布局
   1. 设计登陆窗口
   2. 有用户名 密码、编辑框、登陆和退出按钮
   3. 灵活运用弹簧进行界面布局
5. 常用控件
   1. 按钮组
      1. QPushButton 最常用按钮，里面可以加图片
      2. QToolButton 工具按钮 用于显示图片
         1. toolButtonStyle 设置显示图片和文件
         2. autoRaise透明凸起效果
      3. QRadioButton 单选按钮
         1. 默认选中 setChecked(true)
      4. QCheckBox 复选按钮
         1. stateChanged 信号
         2. 0 未选中
         3. 1 半选中
         4. 2 选中
   2. QListWidget
      1. 创建项 QListWidgetItem \* item = new QListWidgetItem("锄禾日当午");
      2. 将项加载到 控件中
      3. ui->listWidget->addItem(item);
      4. 设置对齐方式 item->setTextAlignment(Qt::AlignHCenter);
      5. ui->listWidget->addItems(list); 可以一次性放入所有内容
   3. QTreeWidget 树控件
      1. 设置头
         1. ui->treeWidget->setHeaderLabels(QStringList()<<"英雄"<<"英雄简介");
      2. 创建根项目
         1. QTreeWidgetItem \* liItem = new QTreeWidgetItem(QStringList()<<"力量");
      3. 将根项目添加到树控件中
         1. ui->treeWidget->addTopLevelItem(liItem);
      4. 挂载子项目
         1. QTreeWidgetItem \* l1 = new QTreeWidgetItem(heroL1);
         2. liItem->addChild(l1);
   4. QTableWidget 表格控件使用
      1. 设置列 ui->tableWidget->setColumnCount(3);
      2. 设置水平表头 ui->tableWidget->setHorizontalHeaderLabels(QStringList()<<"姓名"<<"性别"<<"年龄");
      3. 设置行数 ui->tableWidget->setRowCount(5);
      4. 设置正文
      5. ui->tableWidget->setItem(行,列 , new QTableWidgetItem(“字符串”));
   5. 其他常用控件
      1. 栈控件
         1. ui->stackedWidget->setCurrentIndex(0);
         2. 切换栈显示内容
      2. 下拉框
         1. ui->comboBox->addItem("奔驰");
         2. 设置选中
         3. ui->comboBox->setCurrentIndex(2);
         4. ui->comboBox->setCurrentText("拖拉机");
      3. Qlabel显示图片
         1. ui->label\_img->setPixmap(QPixmap(":/Image/butterfly.png"));
      4. QLabel显示gif图片
         1. QMovie \* movie =new QMovie(":/Image/mario.gif");
         2. ui->label\_movie->setMovie(movie);
         3. movie->start();