1. Qt简介
   1. 发展史
      1. 1991 奇趣科技
   2. 版本
      1. 商业版
      2. 开源版
   3. 优点
      1. 跨平台
      2. 接口简单 容易上手
      3. 一定程度上简化了内存回收机制
      4. 。。。
   4. 成功案例
      1. Linux 桌面环境 KDE
      2. WPS
      3. 谷歌地图
      4. VLC
      5. 虚拟机软件
      6. …
2. Qt项目创建
   1. 项目名称 不能有空格和中文
   2. 项目路径不能有中文路径
   3. 创建窗口三大基类
      1. QWidget
      2. QMainWindow
      3. QDialog
   4. Main函数中
      1. QApplication a 应用程序对象 在Qt中 有且仅有一个
      2. MyWidget w 窗口对象
      3. w.show() 函数显示窗口
      4. return a.exec()进入消息循环机制，阻塞功能
   5. .pro文件
      1. QT += core gui //Qt包含的模块
      2. greaterThan(QT\_MAJOR\_VERSION, 4): QT += widgets //大于4版本 加入 widgets 模块
      3. TARGET = 01\_QtFirst //生成.exe程序的名称
      4. TEMPLATE = app //模板 应用程序模板
      5. SOURCES += main.cpp\ //源文件
      6. mywidget.cpp
      7. HEADERS += mywidget.h //头文件
3. QPushButton按钮
   1. QPushButton \* btn = new QPushButton; 创建按钮对象
   2. btn - >setParent(this) 设置父亲
   3. 设置显示文本 setText
   4. 移动 move
   5. 重置窗口大小 resize
   6. 设置固定窗口大小 setFixedSize
   7. 设置窗口标题 setWindowTitle
4. Qt中的对象树
   1. 一定程度上简化了内存回收机制
   2. 当创建的对象 指定的父亲是由QObject或者Object派生的类时候，这个对象被加载到对象树上，当窗口关闭掉时候，树上的对象也都会被释放掉
5. Qt中的坐标系
   1. x以右侧为正
   2. y以下侧为正
   3. 左上角是 0,0点
6. Qt中信号和槽基本使用
   1. 需求：点击按钮关闭窗口
   2. 连接 connect ( 信号的发送者，发送的信号，信号的接受者，处理的槽函数)

connect( myBtn , &MyPushButton::clicked ,this, &MyWidget::close);

* 1. 测试

1. 自定义信号和槽
   1. 自定义信号 写在 signals下
      1. 返回值是void
      2. 只需要声明 不需要实现
      3. 可以有参数 可以发生重载
   2. 自定义槽函数 写在 public 或者全局函数 或者 public slot 或者lambda
      1. 返回值是void
      2. 需要声明 也需要有实现
      3. 可以有参数 可以发生重载
   3. 当信号和槽发生重载时候，需要利用函数指针明确指出函数地址
      1. void(Teacher:: \*teacherSignal)(QString) = &Teacher::hungry;
   4. 将QString 转为 char \*
      1. .toUtf8()转为 QByteArray数据类型
      2. .data() 转为 char \* 类型
2. 信号和槽拓展
   1. 信号是可以连接信号
   2. 可以断开信号和槽 disconnect
   3. 一个信号可以响应多个槽函数
   4. 多个信号可以连接同一个槽函数
   5. 信号和槽函数的参数类型 必须一一对应,信号的参数个数 可以多余槽函数的参数个数，反之不可以 , 参数类型要一一对应
3. Qt4版本信号和槽写法
   1. 优势 ：参数直观
   2. 劣势 ：参数类型不做匹配检测
   3. Qt4本质 SIGNAL("hungry(int)")SLOT("treat(QString)")
4. Lambda表达式
   1. []()mutable -> type {}组成
   2. []中可以加 = & a &a … 推荐 使用 =
   3. () 形参列表
   4. {} 实现体
   5. mutable可以修改按值传递进来的拷贝
   6. -> type 代表lambda表达式返回值类型