Кабелион графика

Создано системой Doxygen 1.9.7

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

Глава 3

Список файлов

3.1 Файлы

Полный список документированных файлов.

Глава 4

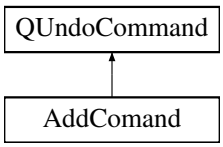
Классы

4.1 Класс AddComand

Класс наследник от QUndoCommand на добавление нового разъема на сцену

#include <AddComand.h>

Граф наследования:AddComand:



Открытые члены

- AddComand (View *v, int x, int y, QString name="")
  Конструктор
- ~AddComand ()
  Деструктор
- void undo () override
  Запоминает координаты и имя, и удаляет разъем
- void redo () override
  Создает разъем в переданных или запомненных координатах, с переданным или запомненным именем

Открытые атрибуты

- Port * p
  Создаваемый разъем
- int xx
  Координаты
- int yy
- View * v
  Отображение в котором работает команда
- QString name
  Имя создоваемого порта

4.1.1 Подробное описание

Класс наследник от QUndoCommand на добавление нового разъема на сцену

Класс ¨сам в себе¨, жрет ы конструкторе там же пушится в стэк....

4.1.2 Конструктор(ы)

4.1.2.1 AddComand()

```
AddComand::AddComand (
                View * v,
                int x,
                int y,
                QString name = "" )
```

Конструктор

Сам в себе высерает себя же в стэк из v и добавляет на сцену, из view, Port

Аргументы

v	Отображение в котором нахоодится нужная сцена и нужный Undo стэк
x	Координата x разъема на сцене
y	Координата y разъема на сцене
name	Имя раъема

Объявления и описания членов классов находятся в файлах:

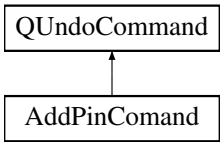
- AddComand.h
- AddComand.cpp

4.2 Класс AddPinComand

Класс наследник от QUndoCommand на добавление нового контакта в разъем

#include <AddPinComand.h>

Граф наследования:AddPinComand:



Открытые члены

- AddPinComand (Port *p, QString name="", bool rea=true)

  Сам в себе высерает себя же в стэк в котором лежит команда создания разъема p.
- ~AddPinComand ()

  Деструктор
- void undo () override

  Удаляет контакт
- void redo () override

  Создает контакт

Открытые атрибуты

- QString name = ""

  Имя контакта
- Pin * pn = nullptr

  Созданный контакт для удаления в дальнейшем
- AddComand * prt

  Команда создания разъема для вытягивания разъема так как он может быть перезаписан а команда останется
- bool real

4.2.1 Подробное описание

Класс наследник от QUndoCommand на добавление нового контакта в разъем

Класс "сам в себе", жрет ы конструкторе там же пушится в стэк....

4.2.2 Конструктор(ы)

4.2.2.1 AddPinComand()

```
AddPinComand::AddPinComand (
            Port * p,
            QString name = "",
            bool rea = true )
```

Сам в себе высерает себя же в стэк в котором лежит команда создания разъема p.

Аргументы

p	Разъем в который добовляем крнтакт
name	имя создоваемого контакта
rea	false если шаблон порта, true если порт на сцене

Объявления и описания членов классов находятся в файлах:

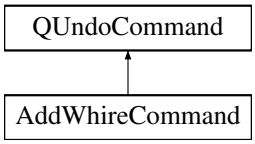
- AddPinComand.h
- AddPinComand.cpp

4.3 Класс AddWhireCommand

Класс наследник от QUndoCommand на добавление нового разъема на сцену

#include <AddWhireCommand.h>

Граф наследования:AddWhireCommand:



Открытые члены

- AddWhireCommand (AddPinComand *p11, AddPinComand *p22)

  Сам в себе высерает себя же в стэк из контакта p11 и добавляет на сцену, из view в разъеме контакта p11 провод,.
- ~AddWhireCommand ()

  Деструктор
- void undo () override

  Удаляет провод
- void redo () override

  Создает провод

Открытые атрибуты

- AddPinComand * p1

  Контакты соедененные проводом
- AddPinComand * p2
- NewWhire * whire

  Созданный провод

4.3.1 Подробное описание

Класс наследник от QUndoCommand на добавление нового разъема на сцену

Класс ¨сам в себе¨, жрет ы конструкторе там же пушится в стэк....

4.3.2 Конструктор(ы)

4.3.2.1 AddWhireCommand()

```
AddWhireCommand::AddWhireCommand (
            AddPinComand * p11,
            AddPinComand * p22 )
```

Сам в себе высерает себя же в стэк из контакта p11 и добавляет на сцену, из view в разъеме контакта p11 провод,.

Аргументы

p11	Первый контакт соеденненный проводом
p22	Второй контакт соеденненный проводом

Объявления и описания членов классов находятся в файлах:

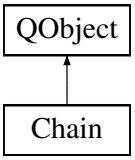
- AddWhireCommand.h
- AddWhireCommand.cpp

4.4 Класс Chain

Класс Цепи Содержит соеденненные в цепь контакты

#include <chain.h>

Граф наследования:Chain:



Открытые члены

- Chain ()

  Новая цепь при соединении 2 контактов без цепей
- ~Chain ()

  Деструктоор
- void AddPin (Pin *p)

  Записывает контакт в цепь при соединении его с контактом входящим в данную цепь
- void RemovePin (Pin *p)

  Удаляет контакт из цепи
- void moveToChain (Chain *chain)

  Соединение данной цепи и chain в одну
- void Dots ()

  Пересчет соединений >2 линий в 1 точке, в цепи

Открытые атрибуты

- QVector< Pin * > pins

  Контакты входящие в цепь
- QColor color

  Цвет проводов в цепи

Статические открытые данные

- static QVector< Chain * > chains

  статический вектор содержащий указатели на все цепи для получения ВСЕХ цепей из вне

4.4.1 Подробное описание

Класс Цепи Содержит соеденненные в цепь контакты

4.4.2 Конструктор(ы)

4.4.2.1 Chain()

Chain::Chain ( )

Новая цепь при соединении 2 контактов без цепей

Сам себя пишет в статический вектор chains

4.4.3 Методы

4.4.3.1 AddPin()

void Chain::AddPin (
Pin * p)

Записывает контакт в цепь при соединении его с контактом входящим в данную цепь

Аргументы

p	Контакт

4.4.3.2 moveToChain()

void Chain::moveToChain (
Chain * chain)

Соединение данной цепи и chain в одну

Аргументы

chain	

4.4.3.3 RemovePin()

void Chain::RemovePin (

Pin * p)

Удаляет контакт из цепи

Аргументы

p	Удаляемый контакт

Объявления и описания членов классов находятся в файлах:

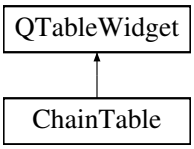
- chain.h
- chain.cpp
- main.cpp

4.5 Класс ChainTable

Табличный вид цепей

#include <chaintable.h>

Граф наследования:ChainTable:



Открытые слоты

- void UpdateTable ()

  Обновление таблицы исходя из отображения
- void CellChange (int row, int column)

  Пересчет цепей по обновлению в таблице
- void CellClck (int row, int colum)

  Обработка клика на +.

Открытые члены

- ChainTable ()

  Конструктор
- ~ChainTable ()

  Деструктор
- void AdddPortSL ()

  Добовление порта в отображеение
- void AddChain ()

  Добовление Цепм в список и в отображение

Открытые атрибуты

- View * view =nullptr
  Отображение в котором выводятся цепи

Защищенные данные

- QMetaObject::Connection m_connection
  связка для обновлений то нужна то мешает тч перезапись

4.5.1 Подробное описание

Табличный вид цепей

4.5.2 Методы

4.5.2.1 CellChange

void ChainTable::CellChange (
int row,
int column) [slot]

Пересчет цепей по обновлению в таблице

Аргументы

row	строка
column	колонка

4.5.2.2 CellClck

void ChainTable::CellClck (
int row,
int colum) [slot]

Обработка клика на +.

Аргументы

row	строка
colum	колонка

Объявления и описания членов классов находятся в файлах:

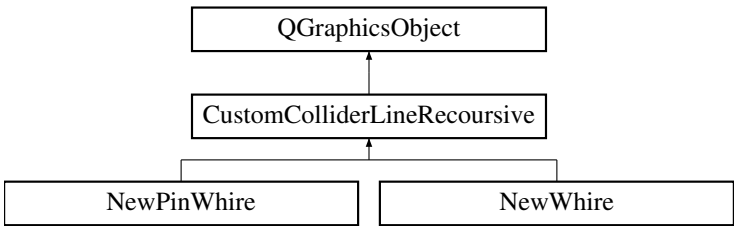
- chaintable.h
- chaintable.cpp

4.6 Класс CustomColliderLineRecoursive

Линия с колизионной корекцией

#include <CustomColliderLineRecoursive.h>

Граф наследования:CustomColliderLineRecoursive:



Открытые слоты

- void FixColliding (int times=0)

  Пересчет коллизии
- void ClearInside ()

  Очистка дочерних линий (выравнивание)
- void setVertical ()

  Смена направления линии на вертикаль
- void setHorizontal ()

  Смена направления линии на горизонталь
- void JumpFrom (QGraphicsItem *itm)

  уход от обойденных ранее объектов

Открытые члены

- virtual QColor color ()

  Цвет хранится в цепях не у линий нет прямого доступа в цепь
- CustomColliderLineRecoursive (bool Vertical_f_Horiz_t, Dot *d1, Dot *d2, CustomColliderLineRecoursive *parent=nullptr)

  Просто QGraphicsItem.
- ~CustomColliderLineRecoursive ()

  Удаляет линию и все дочерние (коррекционные линии)
- QRectF boundingRect () const override

  Линия без учета коррекции даже если не видна
- QPainterPath shape () const override

  Форма для отрисовки
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override

  Отрисовка
- void setFixWay (ColisionFixWay fw)

  Наработка на будущее...

Открытые атрибуты

- QVector< QGraphicsItem * > itsmine

  Уже обойденные данной линией объекты
- bool Vertical_f_Horizontal_t

  Горизонтальная или вертикальная линия false - Вертикальная true - горизонтальная
- int JumpDerection = 1

  Направление отступления при движении обойденных ранее объектов

Защищенные данные

- Dot * d1

  Точки соеденяемые линией
- Dot * d2
- ColisionFixWay fixway

  На будущее
- QVector< QGraphicsObject * > inside

  Дочерние линии
- ProxyRectPort * lastcolide =nullptr

  последний обойденный разъем
- int lastleft
- int lastbottom
- CustomColliderLineRecoursive * Parent

  родительская линия

4.6.1 Подробное описание

Линия с колизионной корекцией

4.6.2 Конструктор(ы)

4.6.2.1 CustomColliderLineRecoursive()

```
CustomColliderLineRecoursive::CustomColliderLineRecoursive (
        bool Vertical_f_Horiz_t,
        Dot * d1,
        Dot * d2,
        CustomColliderLineRecoursive * parent = nullptr )
```

Просто QGraphicsItem.

Аргументы

Vertical_f_Horiz↩ _t	Горизонтальная или вертикальная линия false - Вертикальная true - горизонтальная
d1	крайняя точка 1
d2	крайняя точка 2
parent	Родительская линия НУЖНА ДЛЯ КОРРЕКЦИИ КОЛЛИЗИИ РЕКУРСИЕЙ

4.6.3 Методы

4.6.3.1 boundingRect()

QRectF CustomColliderLineRecoursive::boundingRect ( ) const [override]

Линия без учета коррекции даже если не видна

см. https://doc.qt.io/qt-6/qgraphicsitem.html

Возвращает

Зона необходимая к пересечению для отображения и взаимодействия

4.6.3.2 color()

QColor CustomColliderLineRecoursive::color ( ) [virtual]

Цвет хранится в цепях не у линий нет прямого доступа в цепь

Возвращает

Переопределяется в NewPinWhire и NewWhire.

4.6.3.3 FixColliding

void CustomColliderLineRecoursive::FixColliding (
int times = 0) [slot]

Пересчет коллизии

Аргументы

times	чЧто бы не уйти в вечный цикл при неудачной коллизии

4.6.3.4 JumpFrom

void CustomColliderLineRecoursive::JumpFrom (
QGraphicsItem * itm) [slot]

уход от обойденных ранее объектов

Аргументы

itm	

4.6.3.5 paint()

void CustomColliderLineRecoursive::paint (
QPainter * painter,
const QStyleOptionGraphicsItem * option,
QWidget * widget) [override]

Отрисовка

см. https://doc.qt.io/qt-6/qgraphicsitem.html

Аргументы

painter	
option	
widget	

4.6.3.6 setFixWay()

void CustomColliderLineRecoursive::setFixWay (
ColisionFixWay fw)

Наработка на будущее...

Аргументы

fw	

4.6.3.7 shape()

QPainterPath CustomColliderLineRecoursive::shape ( ) const [override]

Форма для отрисовки

см. https://doc.qt.io/qt-6/qgraphicsitem.html

Возвращает

Объявления и описания членов классов находятся в файлах:

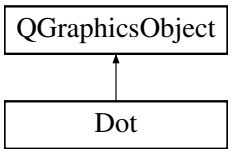
- CustomColliderLineRecoursive.h
- CustomColliderLineRecoursive.cpp

4.7 Класс Dot

Класс Точки дляпостроения линий

#include <Dot.h>

Граф наследования:Dot:



Открытые слоты

- void VerticalDot (Dot *d)
  Слот на обработку смещения другой связанной точки
- void HorizontalDot (Dot *d)
  Слот на обработку смещения другой связанной точки

Сигналы

- void Is_inMove (bool moving)
  Сигнал о смещении точки
- void moving (Dot *d)
  Сигнал о перемещении точки

Открытые члены

- Dot (QGraphicsObject *parent=nullptr)
  QGraphicsItem.
- int type () const override
  для работы через GView
- int x ()
- int y ()
- void x (int x)
  Устанавливает координату по оси x.
- void y (int y)
  Устанавливает координату по оси y.
- void setColor (QColor cl)
  Устанавливает Цвет
- void setTriangle (bool bl)
  Делает треугольной или круглой
- void setBig (bool bl)
  Делает большой или маленькой
- Pin * pn ()
  Контакт за которым данная точка ВОЗМОЖНО закрепленна
- void EmitIs_inMove (bool moving)
- void setPin (Pin *p)

Закрепление за контактом
- void Emit_Moving ()
- ~Dot ()

  Деструктор
- void WhPl ()

  Счетчик проводов когда их 0 удаление
- void WhMin ()
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override
- QRectF boundingRect () const override
- QPainterPath shape () const override
- int whrscount ()

Открытые атрибуты

- QColor cl

  Цвет Линии
- bool triangle

  треугольная ли точка
- bool bg

  Большая ли точка
- CustomColliderLineRecoursive * Vdot

  Линии использующие данную точку
- CustomColliderLineRecoursive * Hdot
- QVector< NewWhire * > whires

  провода использующие данную точку

4.7.1 Подробное описание

Класс Точки дляпостроения линий

4.7.2 Конструктор(ы)

4.7.2.1 Dot()

Dot::Dot (
QGraphicsObject * parent = nullptr)

QGraphicsItem.

см. https://doc.qt.io/qt-6/qgraphicsitem.html

Аргументы

parent	

4.7.3 Методы

4.7.3.1 boundingRect()

QRectF Dot::boundingRect ( ) const [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.7.3.2 paint()

void Dot::paint (
QPainter * painter,
const QStyleOptionGraphicsItem * option,
QWidget * widget) [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.7.3.3 setBig()

void Dot::setBig (
bool bl)

Делает большой или маленькой

Аргументы

bl	

4.7.3.4 setColor()

void Dot::setColor (
QColor cl)

Устанавливает Цвет

Аргументы

cl	цвет

4.7.3.5 setPin()

void Dot::setPin (
Pin * p)

Закрепление за контактом

Аргументы

p	

4.7.3.6 setTriangle()

void Dot::setTriangle (
bool bl)

Делает треугольной или круглой

Аргументы

bl	

4.7.3.7 shape()

QPainterPath Dot::shape ( ) const [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.7.3.8 type()

int Dot::type ( ) const [override]

для работы через GView

Возвращает

4.7.3.9 x()

int Dot::x ( )

Возвращает

Координату x

4.7.3.10 y()

int Dot::y ( )

Возвращает

Координату y

Объявления и описания членов классов находятся в файлах:

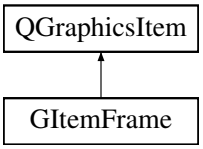
- Dot.h
- Dot.cpp

4.8 Класс GItemFrame

Красная рамка

#include <GItemFrame.h>

Граф наследования:GItemFrame:



Открытые члены

- GItemFrame ()
  QGraphicsItem.
- ~GItemFrame ()
  Деструктор
- QRectF boundingRect () const override
- QPainterPath shape () const override
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override

4.8.1 Подробное описание

Красная рамка

4.8.2 Методы

4.8.2.1 boundingRect()

QRectF GItemFrame::boundingRect ( ) const [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.8.2.2 paint()

void GItemFrame::paint (
QPainter * painter,
const QStyleOptionGraphicsItem * option,
QWidget * widget) [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.8.2.3 shape()

QPainterPath GItemFrame::shape ( ) const [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

Объявления и описания членов классов находятся в файлах:

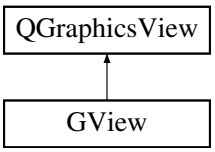
- GItemFrame.h
- GItemFrame.cpp

4.9 Класс GView

Наследование QGraphicsView.

#include <GView.h>

Граф наследования:GView:



Открытые члены

- GView (QObject *parent)
  QGraphicsView.
- ~GView ()
  Деструктор
- MYGraphicsScene * GScene ()
  Сцена в данном отображении
- void mouseMoveEvent (QMouseEvent *event) override
  Перегруз
- void mouseReleaseEvent (QMouseEvent *event) override
  Перегруз
- void mousePressEvent (QMouseEvent *event) override
  Перегруз
- void dropEvent (QDropEvent *event) override
  Перегруз
- void dragEnterEvent (QDragEnterEvent *event) override
  Перегруз
- void dragMoveEvent (QDragMoveEvent *event) override
  Перегруз

Открытые атрибуты

- int lastselected = 0
  Перегруз

4.9.1 Подробное описание

Наследование QGraphicsView.

Нужно для того что бы обрабатывать клики в том числе с EVENT FILTER

4.9.2 Конструктор(ы)

4.9.2.1 GView()

GView::GView (
QObject * parent)

QGraphicsView.

Аргументы

parent	

4.9.3 Методы

4.9.3.1 dragEnterEvent()

void GView::dragEnterEvent (
QDragEnterEvent * event) [override]

Перегруз

Аргументы

event	

4.9.3.2 dragMoveEvent()

void GView::dragMoveEvent (
QDragMoveEvent * event) [override]

Перегруз

Аргументы

event	

4.9.3.3 dropEvent()

void GView::dropEvent (

QDropEvent * event) [override]

Перегруз

Аргументы

event	

4.9.3.4 GScene()

MYGraphicsScene * GView::GScene ( )

Сцена в данном отображении

Возвращает

4.9.3.5 mouseMoveEvent()

void GView::mouseMoveEvent (
QMouseEvent * event) [override]

Перегруз

Аргументы

event	

4.9.3.6 mousePressEvent()

void GView::mousePressEvent (
QMouseEvent * event) [override]

Перегруз

Аргументы

event	

4.9.3.7 mouseReleaseEvent()

void GView::mouseReleaseEvent (
QMouseEvent * event) [override]

Перегруз

Аргументы

event	

4.9.4 Данные класса

4.9.4.1 lastselected

int GView::lastselected = 0

Перегруз

Аргументы

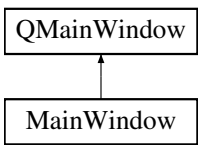
event	

Объявления и описания членов классов находятся в файлах:

- GView.h
- GView.cpp

4.10 Класс MainWindow

Граф наследования:MainWindow:



Открытые слоты

- void AddTemp (Port *p=new Port())
- void hidemMap ()

Открытые члены

- MainWindow (QWidget *parent=nullptr)

Объявления и описания членов классов находятся в файлах:

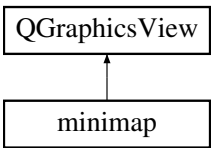
- mainwindow.h
- mainwindow.cpp

4.11 Класс minimap

Мини карта

#include <minimap.h>

Граф наследования:minimap:



Открытые слоты

- void rescale ()
  Пересчет маштаба

Открытые члены

- minimap (View *v)
  крадем сцену у v
- void mouseMoveEvent (QMouseEvent *event) override
  Перегруз
- void mouseReleaseEvent (QMouseEvent *event) override
  Перегруз
- void mousePressEvent (QMouseEvent *event) override
  Перегруз
- void mouseDoubleClickEvent (QMouseEvent *event) override
  Перегруз
- ~minimap ()
  Деструктор

Открытые атрибуты

- View * v
  Основное отображение
- float sclX
- float sclY

4.11.1 Подробное описание

Мини карта

Выводит ту же сцену что и основное отображение, но в другом маштабе

4.11.2 Конструктор(ы)

4.11.2.1 minimap()

minimap::minimap (
View * v)

крадем сцену у v

Аргументы

v	Основное отображение

4.11.3 Методы

4.11.3.1 mouseDoubleClickEvent()

void minimap::mouseDoubleClickEvent (
QMouseEvent * event) [override]

Перегруз

Аргументы

event	

4.11.3.2 mouseMoveEvent()

void minimap::mouseMoveEvent (
QMouseEvent * event) [override]

Перегруз

Аргументы

event	

4.11.3.3 mousePressEvent()

void minimap::mousePressEvent (
QMouseEvent * event) [override]

Перегруз

Аргументы

event	

4.11.3.4 mouseReleaseEvent()

void minimap::mouseReleaseEvent (
QMouseEvent * event) [override]

Перегруз

Аргументы

event	

Объявления и описания членов классов находятся в файлах:

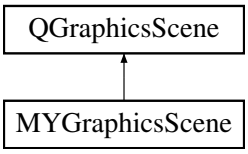
- minimap.h
- minimap.cpp

4.12 Класс MYGraphicsScene

Наследник от QGraphicsScene.

#include <mygraphicsscene.h>

Граф наследования:MYGraphicsScene:



Открытые члены

- MYGraphicsScene (QObject *parent=nullptr)
  Стандартный конструктор

Открытые атрибуты

- View * Mview
  Основное отображение
- int _scale = 100
  Zoom.

4.12.1 Подробное описание

Наследник от QGraphicsScene.

Нужен для пересчета zoom in/out; для прямого доступа к основному отображению через Upcast scene() в graphicsitem; для клеточек

4.12.2 Конструктор(ы)

4.12.2.1 MYGraphicsScene()

MYGraphicsScene::MYGraphicsScene (
QObject * parent = nullptr) [explicit]

Стандартный конструктор

Аргументы

parent	

Объявления и описания членов классов находятся в файлах:

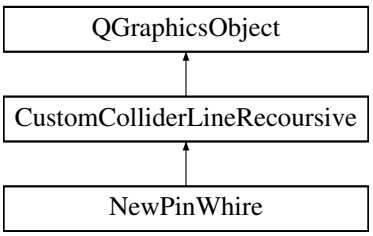
- mygraphicsscene.h
- mygraphicsscene.cpp

4.13 Класс NewPinWhire

Горизонтальный провод идущий от контакта

#include <NewPinWhire.h>

Граф наследования:NewPinWhire:



Открытые члены

- NewPinWhire (Pin *p, QThread *th)
  Конструктор
- ~NewPinWhire ()
  Деструктор
- bool hasConection ()
  наличие других связей
- QColor color () override
  Цвет провода

Открытые члены унаследованные от CustomColliderLineRecoursive

- virtual QColor color ()
  Цвет хранится в цепях не у линий нет прямого доступа в цепь
- CustomColliderLineRecoursive (bool Vertical_f_Horiz_t, Dot *d1, Dot *d2, CustomColliderLineRecoursive *parent=nullptr)
  Просто QGraphicsItem.
- ~CustomColliderLineRecoursive ()
  Удаляет линию и все дочерние (коррекционные линии)
- QRectF boundingRect () const override
  Линия без учета коррекции даже если не видна
- QPainterPath shape () const override
  Форма для отрисовки
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override
  Отрисовка
- void setFixWay (ColisionFixWay fw)
  Наработка на будущее...

Дополнительные унаследованные члены

Открытые слоты унаследованные от CustomColliderLineRecoursive

- void FixColliding (int times=0)

  Пересчет коллизии
- void ClearInside ()

  Очистка дочерних линий (выравнивание)
- void setVertical ()

  Смена направления линии на вертикаль
- void setHorizontal ()

  Смена направления линии на горизонталь
- void JumpFrom (QGraphicsItem *itm)

  уход от обойденных ранее объектов

Открытые атрибуты унаследованные от CustomColliderLineRecoursive

- QVector< QGraphicsItem * > itsmine

  Уже обойденные данной линией объекты
- bool Vertical_f_Horizontal_t

  Горизонтальная или вертикальная линия false - Вертикальная true - горизонтальная
- int JumpDerection = 1

  Направление отступления при движении обойденных ранее объектов

Защищенные данные унаследованные от CustomColliderLineRecoursive

- Dot * d1

  Точки соеденяемые линией
- Dot * d2
- ColisionFixWay fixway

  На будущее
- QVector< QGraphicsObject * > inside

  Дочерние линии
- ProxyRectPort * lastcolide =nullptr

  последний обойденный разъем
- int lastleft
- int lastbottom
- CustomColliderLineRecoursive * Parent

  родительская линия

4.13.1 Подробное описание

Горизонтальный провод идущий от контакта

Всегда один у каждого контакта

4.13.2 Конструктор(ы)

4.13.2.1 NewPinWhire()

```
NewPinWhire::NewPinWhire (
            Pin * p,
            QThread * th )
```

Конструктор

Жрет контакт и поток в котором лежит этот контакт

По идее таки пошустрее в несколько потоков

Аргументы

p	Контакт к которому приложен провод
th	Поток в который он перемещен

4.13.3 Методы

4.13.3.1 color()

```
QColor NewPinWhire::color ( )   [override], [virtual]
```

Цвет провода

Возвращает

Переопределяет метод предка CustomColliderLineRecoursive.

4.13.3.2 hasConection()

```
bool NewPinWhire::hasConection ( )
```

наличие других связей

Возвращает

true - есть; false - нет

Объявления и описания членов классов находятся в файлах:

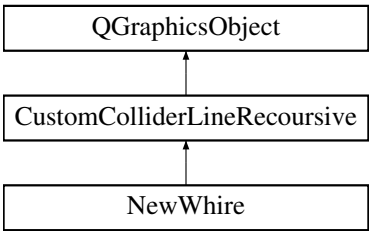
- NewPinWhire.h
- NewPinWhire.cpp

4.14 Класс NewWhire

Вертикальный провод соеденяющий 2 контакта

#include <NewWhire.h>

Граф наследования:NewWhire:



Открытые члены

- NewWhire (Pin *pp1, Pin *pp2, AddWhireCommand *comm)
  Добавляет провод на сцену
- ~NewWhire ()
  Деструктор
- QColor color () override
  Цвет провода

Открытые члены унаследованные от CustomColliderLineRecoursive

- virtual QColor color ()
  Цвет хранится в цепях не у линий нет прямого доступа в цепь
- CustomColliderLineRecoursive (bool Vertical_f_Horiz_t, Dot *d1, Dot *d2, CustomColliderLineRecoursive *parent=nullptr)
  Просто QGraphicsItem.
- ~CustomColliderLineRecoursive ()
  Удаляет линию и все дочерние (коррекционные линии)
- QRectF boundingRect () const override
  Линия без учета коррекции даже если не видна
- QPainterPath shape () const override
  Форма для отрисовки
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override
  Отрисовка
- void setFixWay (ColisionFixWay fw)
  Наработка на будущее...

Открытые статические члены

- static void AddComandW (Pin *p1, Pin *p2)
  static для создания команды AddWhireCommand => провод

Открытые атрибуты

- Pin * p1 = nullptr

  Контакты соеденненные проводом
- Pin * p2 = nullptr
- Chain * chain = nullptr

  Цепь в которую входит данный провод и соеденненные им контакты
- AddWhireCommand * command

  Команда Undo создания провода

Открытые атрибуты унаследованные от CustomColliderLineRecoursive

- QVector< QGraphicsItem * > itsmine

  Уже обойденные данной линией объекты
- bool Vertical_f_Horizontal_t

  Горизонтальная или вертикальная линия false - Вертикальная true - горизонтальная
- int JumpDerection = 1

  Направление отступления при движении обойденных ранее объектов

Дополнительные унаследованные члены

Открытые слоты унаследованные от CustomColliderLineRecoursive

- void FixColliding (int times=0)

  Пересчет коллизии
- void ClearInside ()

  Очистка дочерних линий (выравнивание)
- void setVertical ()

  Смена направления линии на вертикаль
- void setHorizontal ()

  Смена направления линии на горизонталь
- void JumpFrom (QGraphicsItem *itm)

  уход от обойденных ранее объектов

Защищенные данные унаследованные от CustomColliderLineRecoursive

- Dot * d1

  Точки соеденяемые линией
- Dot * d2
- ColisionFixWay fixway

  На будущее
- QVector< QGraphicsObject * > inside

  Дочерние линии
- ProxyRectPort * lastcolide =nullptr

  последний обойденный разъем
- int lastleft
- int lastbottom
- CustomColliderLineRecoursive * Parent

  родительская линия

4.14.1 Подробное описание

Вертикальный провод соеденяющий 2 контакта

соеденяет 2 внешние точки NewPinWhire

4.14.2 Конструктор(ы)

4.14.2.1 NewWhire()

```
NewWhire::NewWhire (
                Pin * pp1,
                Pin * pp2,
                AddWhireCommand * comm )
```

Добавляет провод на сцену

Аргументы

pp1	Первый Контакт
pp2	Второрй Контакт
comm	Команда Undo в которой создается провод (передается из com как this)

4.14.3 Методы

4.14.3.1 AddComandW()

```
void NewWhire::AddComandW (
                Pin * p1,
                Pin * p2 )   [static]
```

static для создания команды AddWhireCommand => провод

Аргументы

p1	Первый Контакт
p2	Второрй Контакт

4.14.3.2 color()

```
QColor NewWhire::color ( )   [override], [virtual]
```

Цвет провода

Возвращает

Переопределяет метод предка CustomColliderLineRecoursive.

Объявления и описания членов классов находятся в файлах:

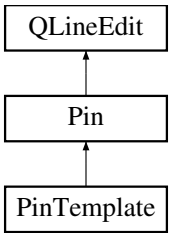
- NewWhire.h
- NewWhire.cpp

4.15 Класс Pin

Контакт

#include <pin.h>

Граф наследования:Pin:



Открытые слоты

- void Update (bool dotupd)
  Обновление
- void RemoveFromChain ()
  удаление из цепи
- void Remove ()
  удаление

Сигналы

- void updSignal (bool upddots)
  Сигнал обновления

Открытые члены

- Pin (Port *port, bool bl=true, QLineEdit *parent=nullptr)
  Создает контакт
- ~Pin ()
  Деструктор
- int index ()
  положение контакта по высоте в разъеме
- const QString name ()
  Имя контакта
- void name (QString name)
  Установка имени контакта
- void EmitUpd (bool dotold=false)
  вызов сигнала обновления
- NewPinWhire * getpinWhire ()
  Возвращает провод от данного контакта
- virtual Dot * dot (bool recalc=false)
  Внешняя точка провода контакта
- virtual void dot (Dot *d)
  установка внешней точки провода
- Dot * coredot ()
  Внутренняя точка провода контката
- qreal x ()
  координата x контакта на сцене
- qreal y ()
  координата y контакта на сцене
- void pinWhire (bool show=true)
  Отображение провода контакта
- void PinWUpd (bool upd=true)
  Пересчет расположения внутрнней точки провода

Открытые атрибуты

- Port * parCon
  Разъем в котором лежим контакт
- Chain * chain
  цепь к которой пренадлежит контакт
- AddPinComand * command =nullptr
  Команда добавления контакта Undo.

Защищенные слоты

- virtual void showContextMenu (const QPoint &pos)
  Демонстрация контекстного меню
- void ChangeColor ()
  Вызов colorDialog для смены цвета цепи в которую входит контакт

Защищенные члены

- virtual QMenu * ContextMenu ()
  Контекстное меню для удаления... по правой кнопке мыши

Защищенные данные

- Ui::Pin * ui
- bool upd = false

  Обновление при движении

4.15.1 Подробное описание

Контакт

4.15.2 Конструктор(ы)

4.15.2.1 Pin()

Pin::Pin (
Port * port,
bool bl = true,
QLineEdit * parent = nullptr) [explicit]

Создает контакт

Аргументы

port	Разъем в котором находится контакт
bl	Делать ли провод к контакту
parent	

4.15.3 Методы

4.15.3.1 ContextMenu()

QMenu * Pin::ContextMenu ( ) [protected], [virtual]

Контекстное меню для удаления... по правой кнопке мыши

Возвращает

4.15.3.2 coredot()

Dot * Pin::coredot ( )

Внутренняя точка провода контката

Возвращает

4.15.3.3 dot() [1/2]

Dot * Pin::dot (
bool recalc = false) [virtual]

Внешняя точка провода контакта

Аргументы

recalc	пересчитывать ли провод (коллизии)

Возвращает

4.15.3.4 dot() [2/2]

void Pin::dot (
Dot * d) [virtual]

установка внешней точки провода

Аргументы

d	

4.15.3.5 EmitUpd()

void Pin::EmitUpd (
bool dotold = false)

вызов сигнала обновления

Аргументы

dotold	

4.15.3.6 getpinWhire()

NewPinWhire * Pin::getpinWhire ( )

Возвращает провод от данного контакта

Возвращает

4.15.3.7 index()

int Pin::index ( )

положение контакта по высоте в разъеме

Возвращает

4.15.3.8 name() [1/2]

const QString Pin::name ( )

Имя контакта

Возвращает

4.15.3.9 name() [2/2]

void Pin::name (
QString name)

Установка имени контакта

Аргументы

name	

4.15.3.10 pinWhire()

void Pin::pinWhire (
bool show = true)

Отображение провода контакта

Аргументы

show	

4.15.3.11 PinWUpd()

void Pin::PinWUpd (
bool upd = true)

Пересчет расположения внутрнней точки провода

Аргументы

upd	

4.15.3.12 x()

qreal Pin::x ( )

координата x контакта на сцене

Возвращает

Объявления и описания членов классов находятся в файлах:

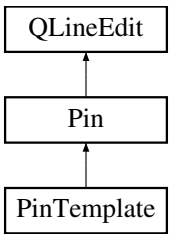
- pin.h
- pin.cpp

4.16 Класс PinTemplate

Контакт из шаблонов (библиотеки)

#include <PinTemplate.h>

Граф наследования:PinTemplate:



Открытые члены

- PinTemplate (Port *p)
  Конструктор Pin.
- virtual void dragEnterEvent (QDragEnterEvent *event) override
  отключение перемещение итд
- virtual void dropEvent (QDropEvent *event) override
  отключение перемещение итд
- virtual void dragMoveEvent (QDragMoveEvent *event) override
  отключение перемещение итд
- ~PinTemplate ()
  Деструктор

Открытые члены унаследованные от Pin

- Pin (Port *port, bool bl=true, QLineEdit *parent=nullptr)
  Создает контакт
- ~Pin ()
  Деструктор
- int index ()
  положение контакта по высоте в разъеме
- const QString name ()
  Имя контакта
- void name (QString name)
  Установка имени контакта
- void EmitUpd (bool dotold=false)
  вызов сигнала обновления
- NewPinWhire * getpinWhire ()
  Возвращает провод от данного контакта
- virtual Dot * dot (bool recalc=false)
  Внешняя точка провода контакта
- virtual void dot (Dot *d)
  установка внешней точки провода
- Dot * coredot ()
  Внутренняя точка провода конткта
- qreal x ()
  координата x контакта на сцене
- qreal y ()
  координата y контакта на сцене
- void pinWhire (bool show=true)
  Отображение провода контакта
- void PinWUpd (bool upd=true)
  Пересчет расположения внутрнней точки провода

Дополнительные унаследованные члены

Открытые слоты унаследованные от Pin

- void Update (bool dotupd)
  Обновление
- void RemoveFromChain ()
  удаление из цепи
- void Remove ()
  удаление

Сигналы унаследованные от Pin

- void updSignal (bool upddots)
  Сигнал обновления

Открытые атрибуты унаследованные от Pin

- Port * parCon

  Разъем в котором лежим контакт
- Chain * chain

  цепь к которой пренадлежит контакт
- AddPinComand * command =nullptr

  Команда добавления контакта Undo.

Защищенные слоты унаследованные от Pin

- virtual void showContextMenu (const QPoint &pos)

  Демонстрация контекстного меню
- void ChangeColor ()

  Вызов colorDialog для смены цвета цепи в которую входит контакт

Защищенные члены унаследованные от Pin

- virtual QMenu * ContextMenu ()

  Контекстное меню для удаления... по правой кнопке мыши

Защищенные данные унаследованные от Pin

- Ui::Pin * ui
- bool upd = false

  Обновление при движении

4.16.1 Подробное описание

Контакт из шаблонов (библиотеки)

отключены провода итд

4.16.2 Конструктор(ы)

4.16.2.1 PinTemplate()

PinTemplate::PinTemplate (
Port * p)

Конструктор Pin.

Аргументы

p	разъем в котором лежит контакт (в среднем шаблонный разъем)

4.16.3 Методы

4.16.3.1 dragEnterEvent()

void PinTemplate::dragEnterEvent (
QDragEnterEvent * event) [override], [virtual]

отключение перемещение итд

Аргументы

event	

Переопределяет метод предка Pin.

4.16.3.2 dragMoveEvent()

void PinTemplate::dragMoveEvent (
QDragMoveEvent * event) [override], [virtual]

отключение перемещение итд

Переопределяет метод предка Pin.

4.16.3.3 dropEvent()

void PinTemplate::dropEvent (
QDropEvent * event) [override], [virtual]

отключение перемещение итд

Переопределяет метод предка Pin.

Объявления и описания членов классов находятся в файлах:

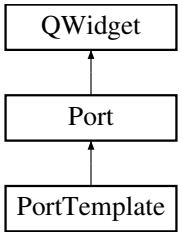
- PinTemplate.h
- PinTemplate.cpp

4.17 Класс Port

Разъем на сцене

#include <port.h>

Граф наследования:Port:



Открытые слоты

- void Update (bool updF=false)

  Пересчет позиции и проводов
- ProxyRectPort * proxy ()

  Прокси в который вложен разъем
- void proxy (ProxyRectPort *prox)

  Передача прокси в который вкладывается разъем

Открытые члены

- qreal x ()

  Координата x на сцене
- qreal y ()

  Координата y на сцене
- Port (AddComand *com, QWidget *parent=nullptr)

  Создает разъем
- Port ()

  Создает разъем
- virtual ~Port ()

  Деструктор
- QString name ()

  Имя разъема
- void name (QString str)

  Установка имени разъема
- QVector< Pin * > pins ()

  Список всех контактов разъема
- void Remove ()

  Удаление разъема
- virtual Pin * addPin (QString name="", int index=-1, bool bl=true)

  Добавление нового контакта

Открытые атрибуты

- AddComand * adcom

  Команда Undo по которой создан данный разъем

Статические открытые данные

- static QVector< Port * > portsVector

  Статический вектор для доступа ко всем разъемам из вне

Защищенные члены

- virtual QMenu * ContextMenu ()

  Контекстное меню. Удаление; добавление контакта...

Защищенные данные

- Ui::Port * ui
- ProxyRectPort * _proxy

  прокси в который вложен разъем
- QSpacerItem * spacer

  смещение для выравнивания контактов

4.17.1 Подробное описание

Разъем на сцене

Вкладывается в прокси

4.17.2 Конструктор(ы)

4.17.2.1 Port()

```
Port::Port (
            AddComand * com,
            QWidget * parent = nullptr )    [explicit]
```

Создает разъем

Аргументы

com	Undo команда в которой созданданный разъем
parent	

4.17.3 Методы

4.17.3.1 addPin()

```
Pin * Port::addPin (
            QString name = "",
            int index = -1,
            bool bl = true )    [virtual]
```

Добавление нового контакта

Аргументы

name	имя нового контакта
index	место в списке контактов в разъеме
bl	выравнивание в высоту по клеточкам

Возвращает

Созданный контакт

4.17.3.2 ContextMenu()

QMenu * Port::ContextMenu ( ) [protected], [virtual]

Контекстное меню. Удаление; добавление контакта...

Возвращает

4.17.3.3 name() [1/2]

QString Port::name ( )

Имя разъема

Возвращает

4.17.3.4 name() [2/2]

void Port::name (
QString str)

Установка имени разъема

Аргументы

str	

4.17.3.5 pins()

QVector< Pin * > Port::pins ( )

Список всех контактов разъема

Возвращает

4.17.3.6 proxy [1/2]

ProxyRectPort * Port::proxy ( ) [slot]

Прокси в который вложен разъем

Возвращает

4.17.3.7 proxy [2/2]

void Port::proxy (
ProxyRectPort * prox) [slot]

Передача прокси в который вкладывается разъем

Аргументы

prox	

4.17.3.8 x()

qreal Port::x ( )

Координата x на сцене

Возвращает

4.17.3.9 y()

qreal Port::y ( )

Координата y на сцене

Возвращает

Объявления и описания членов классов находятся в файлах:

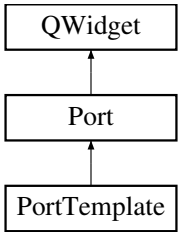
- port.h
- main.cpp
- port.cpp

4.18 Класс PortTemplate

Шаблон Разъема; перегрузка разъема (убрано перемещение итд)

#include <PortTemplate.h>

Граф наследования:PortTemplate:



Открытые члены

- PortTemplate (PortTwmplateObject *o)
  Новый Шаблон hfp]tvf.
- Pin * addPinn (QString name="", int index=-1)
  Добавление контакта (шаблона контакта)
- ~PortTemplate ()
  Деструктор

Открытые члены унаследованные от Port

- qreal x ()
  Координата x на сцене
- qreal y ()
  Координата y на сцене
- Port (AddComand *com, QWidget *parent=nullptr)
  Создает разъем
- Port ()
  Создает разъем
- virtual ~Port ()
  Деструктор
- QString name ()
  Имя разъема
- void name (QString str)
  Установка имени разъема
- QVector< Pin * > pins ()
  Список всех контактов разъема
- void Remove ()
  Удаление разъема
- virtual Pin * addPin (QString name="", int index=-1, bool bl=true)
  Добавление нового контакта

Дополнительные унаследованные члены

Открытые слоты унаследованные от Port

- void Update (bool updF=false)

  Пересчет позиции и проводов
- ProxyRectPort * proxy ()

  Прокси в который вложен разъем
- void proxy (ProxyRectPort *prox)

  Передача прокси в который вкладывается разъем

Открытые атрибуты унаследованные от Port

- AddComand * adcom

  Команда Undo по которой создан данный разъем

Статические открытые данные унаследованные от Port

- static QVector< Port * > portsVector

  Статический вектор для доступа ко всем разъемам из вне

Защищенные члены унаследованные от Port

- virtual QMenu * ContextMenu ()

  Контекстное меню. Удаление; добавление контакта...

Защищенные данные унаследованные от Port

- Ui::Port * ui
- ProxyRectPort * _proxy

  прокси в который вложен разъем
- QSpacerItem * spacer

  смещение для выравнивания контактов

4.18.1 Подробное описание

Шаблон Разъема; перегрузка разъема (убрано перемещение итд)

4.18.2 Конструктор(ы)

4.18.2.1 PortTemplate()

```
PortTemplate::PortTemplate (
            PortTwmplateObject * o )
```

Новый Шаблон hfp]tvf.

Аргументы

o	Объект для Drag and Drop

4.18.3 Методы

4.18.3.1 addPinn()

```
Pin * PortTemplate::addPinn (
                QString name = "",
                int index = -1 )
```

Добавление контакта (шаблона контакта)

Аргументы

name	
index	

Возвращает

Добавленный контакт

Объявления и описания членов классов находятся в файлах:

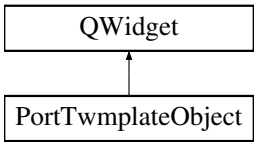
- PortTemplate.h
- PortTemplate.cpp

4.19 Класс PortTwmplateObject

Объект для шаблона разъема

#include <PortTwmplateObject.h>

Граф наследования:PortTwmplateObject:



Открытые слоты

- void editPush ()

  слот на изменение
- void delPush ()

  удаление

Открытые члены

- void dragEnterEvent (QDragEnterEvent *event) override
  Перетягивание на сцену
- void dropEvent (QDropEvent *event) override
  Перетягивание на сцену
- void dragMoveEvent (QDragMoveEvent *event) override
  Перетягивание на сцену
- void mousePressEvent (QMouseEvent *event) override
  Перетягивание на сцену
- PortTwmplateObject ()
  конструктор
- ~PortTwmplateObject ()
  деструктор

Открытые атрибуты

- PortTemplate * templ
  Шаблон разъема
- QPushButton * editpb
  кнопка изменения
- QPushButton * delpb
  кнопка удаления
- QLineEdit * name
  строка имени

4.19.1 Подробное описание

Объект для шаблона разъема

4.19.2 Методы

4.19.2.1 dragEnterEvent()

void PortTwmplateObject::dragEnterEvent (
QDragEnterEvent * event) [override]

Перетягивание на сцену

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.19.2.2 dragMoveEvent()

void PortTwmplateObject::dragMoveEvent (
QDragMoveEvent * event) [override]

Перетягивание на сцену

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.19.2.3 dropEvent()

void PortTwmplateObject::dropEvent (
QDropEvent * event) [override]

Перетягивание на сцену

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.19.2.4 mousePressEvent()

void PortTwmplateObject::mousePressEvent (
QMouseEvent * event) [override]

Перетягивание на сцену

см. https://doc.qt.io/qt-6/qgraphicsitem.html

Объявления и описания членов классов находятся в файлах:

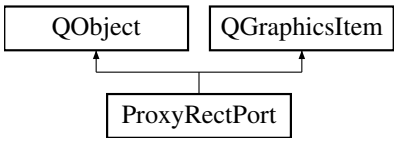
- PortTwmplateObject.h
- PortTwmplateObject.cpp

4.20 Класс ProxyRectPort

Прослойка для перемещения разъема по сцене

#include <proxyrectport.h>

Граф наследования:ProxyRectPort:



Сигналы

- void strartMove ()
  сигнал отметка начало перемещения

Открытые члены

- ProxyRectPort (Port *port)

  Создает прослойку вкладывает в нее разъем
- void EmitMove ()

  Сигна о перемещении
- QPainterPath shape () const override
- QRectF boundingRect () const override
- void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override
- QRectF geometry ()

  костыль аля bounding rect
- void geometry (QRectF rec)

  установка костыл аля bounding rect
- void setconnector (Port *port)

  Установка вложенного разъема
- Port * getport ()

  Вложенный порт
- QColor color ()

  Цвет прокси (Шапка над разъемом)
- void color (QColor c)

  установка цвета прокси (Шапка над разъемом)
- ~ProxyRectPort ()

  удаление прокси и вложенного разъема
- void Update (bool upd=false, int i=0)

  пересчет местоположения от колизий и обновление вложенного разъема
- int type () const override

  Нужен для обновления через GView.
- void ColiderCheck (bool upd=true)

  Проверка коллизий с проводами
- void ProxyColider (int xx=0, int yy=0)

  Проверка коллизий разъемами (рекурия)
- qreal XX ()

  координата по x;
- qreal YY ()

  координата по y;
- void XX (qreal x)

  установка координаты по x;
- void YY (qreal y)

  установка координаты по x;
- qreal left ()

  Левая сторона
- qreal right ()

  Правая сторона
- qreal bottom ()

  Низ
- qreal top ()

  Верх
- QPointF center ()

  координаты центра

4.20.1 Подробное описание

Прослойка для перемещения разъема по сцене

4.20.2 Конструктор(ы)

4.20.2.1 ProxyRectPort()

ProxyRectPort::ProxyRectPort (
Port * port)

Создает прослойку вкладывает в нее разъем

Аргументы

port	Вложеный разъем

4.20.3 Методы

4.20.3.1 bottom()

qreal ProxyRectPort::bottom ( ) [inline]

Низ

Возвращает

у нижнего края

4.20.3.2 boundingRect()

QRectF ProxyRectPort::boundingRect ( ) const [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.20.3.3 center()

QPointF ProxyRectPort::center ( ) [inline]

координаты центра

Возвращает

4.20.3.4 ColiderCheck()

void ProxyRectPort::ColiderCheck (
bool upd = true)

Проверка коллизий с проводами

Аргументы

upd	

4.20.3.5 color() [1/2]

QColor ProxyRectPort::color ( )

Цвет прокси (Шапка над разъемом)

Возвращает

4.20.3.6 color() [2/2]

void ProxyRectPort::color (
QColor c)

установка цвета прокси (Шапка над разъемом)

Возвращает

4.20.3.7 geometry() [1/2]

QRectF ProxyRectPort::geometry ( )

костыль аля bounding rect

Возвращает

4.20.3.8 geometry() [2/2]

void ProxyRectPort::geometry (
QRectF rec)

установка костыл аля bounding rect

Аргументы

rec	

4.20.3.9 getport()

Port * ProxyRectPort::getport ( )

Вложенный порт

Возвращает

4.20.3.10 left()

qreal ProxyRectPort::left ( ) [inline]

Левая сторона

Возвращает

x левого края

4.20.3.11 paint()

void ProxyRectPort::paint (
QPainter * painter,
const QStyleOptionGraphicsItem * option,
QWidget * widget) [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.20.3.12 ProxyColider()

void ProxyRectPort::ProxyColider (
int xx = 0,
int yy = 0)

Проверка коллизий разъемами (рекурия)

Аргументы

xx	направление смещения в случае рекурсивного наложения
yy	направление смещения в случае рекурсивного наложения

4.20.3.13 right()

qreal ProxyRectPort::right ( ) [inline]

Правая сторона

Возвращает

x правого края

4.20.3.14 setconnector()

void ProxyRectPort::setconnector (
Port * port)

Установка вложенного разъема

Аргументы

port	

4.20.3.15 shape()

QPainterPath ProxyRectPort::shape ( ) const [override]

см. https://doc.qt.io/qt-6/qgraphicsitem.html

4.20.3.16 top()

qreal ProxyRectPort::top ( ) [inline]

Верх

Возвращает

y верхнего края

4.20.3.17 type()

int ProxyRectPort::type ( ) const [override]

Нужен для обновления через GView.

Возвращает

4.20.3.18 Update()

void ProxyRectPort::Update (
bool upd = false,
int i = 0)

пересчет местоположения от колизий и обновление вложенного разъема

Аргументы

upd	
i	

4.20.3.19 XX() [1/2]

qreal ProxyRectPort::XX ( )

координата по x;

Возвращает

4.20.3.20 XX() [2/2]

void ProxyRectPort::XX (
qreal x)

установка координаты по x;

Возвращает

4.20.3.21 YY() [1/2]

qreal ProxyRectPort::YY ( )

координата по y;

Возвращает

4.20.3.22 YY() [2/2]

void ProxyRectPort::YY (
qreal y)

установка координаты по x;

Возвращает

Объявления и описания членов классов находятся в файлах:

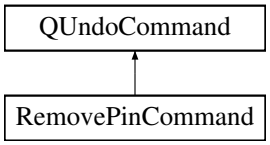
- proxyrectport.h
- proxyrectport.cpp

4.21 Класс RemovePinCommand

Undo Redo Удаление контакта

#include <RemovePinCommand.h>

Граф наследования:RemovePinCommand:



Открытые члены

- RemovePinCommand (Pin *p)
  создание команды в стэк
- ~RemovePinCommand ()
  Деструктор
- void undo () override
  Создает контакт снова и закладывает его в ту же команду создания для сохранения цепочки
- void redo () override
  Удаляет контакт запоминая его разъем и позицию в нем

Открытые атрибуты

- int pos = 0
  позиция удаляемого контакта в разъеме
- QString name = ""
  имя удаляемого контакта
- AddPinComand * pinc
  Команда создания удаляемого контакта (для перезаписи при откате
- Port * prt
  Разъем в котором находится контакт
- Pin * pin
  Удаляемый контакт

4.21.1 Подробное описание

Undo Redo Удаление контакта

4.21.2 Конструктор(ы)

4.21.2.1 RemovePinCommand()

RemovePinCommand::RemovePinCommand (
Pin * p)

создание команды в стэк

Аргументы

p	удаляемый контакт

Объявления и описания членов классов находятся в файлах:

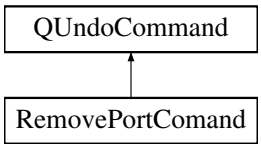
- RemovePinCommand.h
- RemovePinCommand.cpp

4.22 Класс RemovePortComand

Undo Redo Удаление разъема

#include <RemovePortComand.h>

Граф наследования:RemovePortComand:



Открытые члены

- RemovePortComand (Port *p)
- ~RemovePortComand ()
  Деструктор
- void undo () override
  Создает разъем снова и закладывает его в ту же команду создания для сохранения цепочки
- void redo () override
  Удаляет разъем

Открытые атрибуты

- QString name
  имя разъема
- AddComand * addcom
  Команда создания удаляемого разъема (для перезаписи при откате)
- View * v
  отображение в котором лежит разъем
- int xx
  координаты разъема
- int yy

4.22.1 Подробное описание

Undo Redo Удаление разъема

4.22.2 Конструктор(ы)

4.22.2.1 RemovePortComand()

RemovePortComand::RemovePortComand (
Port * p)

создание команды в стэк

Аргументы

p	удаляемый разъем

Объявления и описания членов классов находятся в файлах:

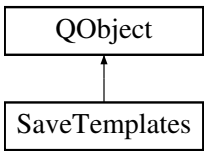
- RemovePortComand.h
- RemovePortComand.cpp

4.23 Класс SaveTemplates

static сохранение шаблонов

#include <SaveTemplates.h>

Граф наследования:SaveTemplates:



Открытые статические члены

- static void Save (PortTwmplateObject *o)
  сохранение шаблона
- static QList< QPair< QStringList, QString > > Load ()
  выгрузка листа шаблонов
- static void Del (QString path)
  удаление шаблона

4.23.1 Подробное описание

static сохранение шаблонов

4.23.2 Методы

4.23.2.1 Del()

void SaveTemplates::Del (
QString path) [static]

удаление шаблона

Аргументы

path	

4.23.2.2 Load()

QList< QPair< QStringList, QString > > SaveTemplates::Load ( ) [static]

выгрузка листа шаблонов

Возвращает

4.23.2.3 Save()

void SaveTemplates::Save (
PortTwmplateObject * o) [static]

сохранение шаблона

Аргументы

o	

Объявления и описания членов классов находятся в файлах:

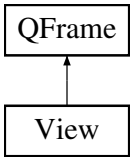
- SaveTemplates.h
- SaveTemplates.cpp

4.24 Класс View

¨Дисплей¨ с вложеным отображением

#include <view.h>

Граф наследования:View:



Открытые слоты

- void stckUndo ()
  Undo Redo.
- void stckRedo ()
  Undo Redo.
- void AdddPortSL ()
  Добовление разъема
- void saveImage ()
  Сохранение скриншота
- void AdddPortSL (int x, int y, QString name)
  Добовление разъема

Сигналы

- void SceneChanged ()
  Сигнал на изменение сцены

Открытые члены

- View ()
  Создает Frame с вложенным View.
- MYGraphicsScene * GScene ()
  Сцена вложенная в отображение
- void GScene (MYGraphicsScene *scene)
  установка сцены в отображение
- QColor backGroundColor ()
  цвет фона
- void backGroundColor (QColor color)
  установка цвета фона
- GView * view ()
  Отображение в Frame.
- int scale ()
  Маштаб
- AddComand * AdddPort ()
  Добовление разъема
- AddComand * AdddPort (int x, int y, QString name)
  Добовление разъема
- void stackPush (QUndoCommand *com)
  Undo Redo.

Открытые атрибуты

- QUndoStack * stack
  Undo|Redo stack.

Защищенные слоты

- virtual void showContextMenu (const QPoint &pos)
  Добовление Разъема контекстное меню

Защищенные члены

- virtual QMenu * ContextMenu ()
  Добовление Разъема контекстное меню

Защищенные данные

- GView * graphicsview
  отображение
- int _scale

4.24.1 Подробное описание

¨Дисплей¨ с вложеным отображением

4.24.2 Методы

4.24.2.1 backGroundColor() [1/2]

QColor View::backGroundColor ( )

цвет фона

Возвращает

4.24.2.2 backGroundColor() [2/2]

void View::backGroundColor (
QColor color)

установка цвета фона

Аргументы

color	

4.24.2.3 ContextMenu()

QMenu * View::ContextMenu ( ) [protected], [virtual]

Добовление Разъема контекстное меню

Возвращает

4.24.2.4 GScene() [1/2]

MYGraphicsScene * View::GScene ( )

Сцена вложенная в отображение

Возвращает

4.24.2.5 GScene() [2/2]

void View::GScene (
MYGraphicsScene * scene)

установка сцены в отображение

Аргументы

scene	

4.24.2.6 scale()

int View::scale ( )

Маштаб

Возвращает

4.24.2.7 view()

GView * View::view ( )

Отображение в Frame.

Возвращает

Объявления и описания членов классов находятся в файлах:

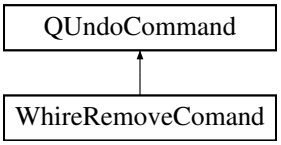
- view.h
- view.cpp

4.25 Класс WhireRemoveComand

Команда удаления провода Undo Redo.

#include <WhireRemoveComand.h>

Граф наследования:WhireRemoveComand:



Открытые члены

- WhireRemoveComand (NewWhire *w)
  Создание команды на удаление провода
- ~WhireRemoveComand ()
  Деструктор
- void undo () override
  возвращение кабеля между p1 и p2 и запись в команду wc
- void redo () override
  Удаление кабеля

Открытые атрибуты

- AddPinComand * p1
  контакты p1 и p2
- AddPinComand * p2
- AddWhireCommand * wc
  Команда добовления удаляемого кабеля

4.25.1 Подробное описание

Команда удаления провода Undo Redo.

4.25.2 Конструктор(ы)

4.25.2.1 WhireRemoveComand()

WhireRemoveComand::WhireRemoveComand (
NewWhire * w)

Создание команды на удаление провода

Аргументы

w	удаляемый провод

Объявления и описания членов классов находятся в файлах:

- WhireRemoveComand.h
- WhireRemoveComand.cpp

Глава 5

Файлы

5.1 AddComand.h

```
00001 #pragma once
00002 #ifndef ADDCOMAND_H
00003 #define ADDCOMAND_H
00004 #include <QUndoCommand>
00005 #include<QDebug>
00006 #include<qgraphicsproxywidget.h>
00007 class View;
00008 class Port;
00009
00016 class AddComand  : public QUndoCommand
00017 {
00018 public:
00029     AddComand(View* v, int x, int y,QString name="");
00033     ~AddComand();
00037     void undo() override;
00041     void redo() override;
00045     Port* p;
00049     int xx, yy;
00053     View* v;
00057     QString name;
00058 };
00059 #endif
```

5.2 AddPinComand.h

```
00001 #pragma once
00002 #ifndef ADDPINCOMAND_H
00003 #define ADDPINCOMAND_H
00004 #include <QUndoCommand>
00005 class Port;
00006 class Pin;
00007 class AddComand;
00014 class AddPinComand  : public QUndoCommand
00015 {
00016 public:
00023     AddPinComand(Port*p,QString name="",bool rea=true);
00027     ~AddPinComand();
00031     void undo() override;
00035     void redo() override;
00039     QString name = "";
00043     Pin* pn = nullptr;
00047     AddComand* prt;
00048     bool real;
00049 };
00050 #endif
```

5.3 AddWhireCommand.h

```
00001 #pragma once
00002 #ifndef ADDWHIRECOMMAND_H
```

```
00003 #define ADDWHIRECOMAND_H
00004 #include<QUndoCommand>
00005 class AddPinComand;
00006 class NewWhire;
00013 class AddWhireCommand: public QUndoCommand
00014 {
00015 public:
00021     AddWhireCommand(AddPinComand* p11, AddPinComand* p22);
00025     ~AddWhireCommand();
00029     void undo() override;
00033     void redo() override;
00037     AddPinComand* p1, *p2;
00041     NewWhire* whire;
00042 };
00043 #endif
```

5.4 chain.h

```
00001 #pragma once
00002 #include<qobject.h>
00003 #include"qcolor.h"
00004 class Pin;
00005 #ifndef CHAIN_H
00006 #define CHAIN_H
00013 class Chain :public QObject
00014 {
00015     Q_OBJECT
00016 public:
00023     Chain();
00027     ~Chain();
00032     void AddPin(Pin* p);
00037     void RemovePin(Pin* p);
00041     static QVector<Chain*> chains;
00045     QVector<Pin*> pins;
00050     void moveToChain(Chain* chain);
00054     void Dots();
00058     QColor color;
00059 };
00060 #endif
```

5.5 chaintable.h

```
00001 #pragma once
00002 #ifndef CHAINTABLE_H
00003 #define CHAINTABLE_H
00004 class View;
00005 #include<qobject.h>
00006 #include<qtablewidget.h>
00007 #include<qinputdialog.h>
00008 #include"qheaderview.h"
00012 class ChainTable  : public QTableWidget
00013 {
00014     Q_OBJECT
00015
00016 public:
00020     ChainTable();
00024     ~ChainTable();
00028     View* view=nullptr;
00032     void AdddPortSL();
00036     void AddChain();
00037 public slots:
00041     void UpdateTable();
00047     void CellChange(int row, int column);
00053     void CellClck(int row, int colum);
00054 protected:
00058     QMetaObject::Connection m_connection;
00059 };
00060 #endif
```

5.6 CustomColliderLineRecoursive.h

```
00001 #pragma once
00002 #ifndef CUSTOMCOLLIDERLINERECOURSIVE_H
00003 #define CUSTOMCOLLIDERLINERECOURSIVE_H
00004
00005 #include<qobject.h>
```

```
00006 #include<qgraphicsobject>
00007 #include"qgraphicsscene.h"
00008 class Dot;
00009 class ProxyRectPort;
00010
00011 enum ColisionFixWay
00012 {
00013     none, run, flow
00014 };
00018 class CustomColliderLineRecoursive  : public QGraphicsObject
00019 {
00020     Q_OBJECT
00021 public:
00026     virtual QColor color();
00034     CustomColliderLineRecoursive(bool Vertical_f_Horiz_t,Dot* d1,Dot* d2, CustomColliderLineRecoursive*
        parent=nullptr);
00038     ~CustomColliderLineRecoursive();
00046     QRectF boundingRect() const override;//
00054     QPainterPath shape() const override;//
00064     void paint(QPainter* painter, const QStyleOptionGraphicsItem* option, QWidget* widget) override;
00069     void setFixWay(ColisionFixWay fw);
00073     QVector<QGraphicsItem*>itsmine;
00077     bool Vertical_f_Horizontal_t;
00081     int JumpDerection = 1;
00082 public slots:
00088     void FixColliding(int times=0);
00092     void ClearInside();
00096     void setVertical();
00100     void setHorizontal();
00105     void JumpFrom(QGraphicsItem* itm);
00106 protected:
00110     Dot* d1, *d2;
00114     ColisionFixWay fixway;
00118     QVector<QGraphicsObject*> inside;
00122     ProxyRectPort* lastcolide=nullptr;
00123     int lastleft, lastbottom;
00127     CustomColliderLineRecoursive* Parent;
00128
00129 };
00130
00131 #endif
```

5.7 Dot.h

```
00001 #pragma once
00002 #ifndef DOT_H
00003 #define DOT_H
00004 #include<QtMath>
00005 #include<QWidget>
00006 #include<QStyleOptionGraphicsItem>
00007 #include<QGraphicsObject>
00008 #include<QPainter>
00009 class CustomColliderLineRecoursive;
00010 class NewWhire;
00011 class Pin;
00012 #include"pin.h"
00013
00017 class Dot  : public QGraphicsObject
00018 {
00019     Q_OBJECT
00020
00021 public:
00029     Dot(QGraphicsObject* parent = nullptr);
00034     int type() const override;
00038     QColor cl;
00042     bool triangle;
00046     bool bg;
00050     int x();
00054     int y();
00058     void x(int x);
00062     void y(int y);
00067     void setColor(QColor cl);
00072     void setTriangle(bool bl);
00077     void setBig(bool bl);
00081     Pin* pn();
00085     CustomColliderLineRecoursive*Vdot, *Hdot;
00086     void EmitIs_inMove(bool moving)
00087     {
00088         if (_pin != nullptr)
00089         {
00090             _pin->Update(false);
00091         }
00092         emit Is_inMove(moving);
00093     }
```

```
00098     void setPin(Pin* p);
00099     void Emit_Moving()
00100     {
00101         int x = pos().x();
00102         int y = pos().y();
00103         int mx, my;
00104         mx = x % 25;
00105         if (mx < 0)
00106             mx += 25;
00107         my = y % 25;
00108         if (my < 0)
00109             my += 25;
00110         if (mx > 12)
00111             x = x - mx + 25;
00112         else
00113             x = x - mx;
00114         if (y % 25 > 12)
00115             y = y - my + 25;
00116         else
00117             y = y - my;
00118         setPos(x, y);
00119         emit moving(this);
00120     }
00124     ~Dot();
00128     QVector<NewWhire*>whires;
00132     void WhPl();
00133     void WhMin();
00139     void paint(QPainter* painter, const QStyleOptionGraphicsItem* option, QWidget* widget) override;
00145     QRectF boundingRect() const override;//
00151     QPainterPath shape() const override;//
00152     int whrscount()
00153     {
00154         return Linecounter;
00155     }
00156
00157 signals:
00161     void Is_inMove(bool moving);
00165     void moving(Dot* d);
00166 public slots:
00170     void VerticalDot(Dot* d);
00174     void HorizontalDot(Dot* d);
00175 private:
00179     int Linecounter = 0;
00183     Pin* _pin;
00184
00185 };
00186 #endif
```

5.8 GItemFrame.h

```
00001 #pragma once
00002 #ifndef GITEMFRAME_H
00003 #define GITEMFRAME_H
00004 #include<QPen>
00005 #include<QPainter>
00006 #include<qgraphicsitem.h>
00010 class GItemFrame  : public  QGraphicsItem
00011 {
00012
00013 public:
00017     GItemFrame();
00021     ~GItemFrame();
00027     QRectF boundingRect() const override;//
00033     QPainterPath shape() const override;//
00039     void paint(QPainter* painter, const QStyleOptionGraphicsItem* option, QWidget* widget) override;
00040
00041
00042 };
00043 #endif
```

5.9 GView.h

```
00001 #pragma once
00002 #ifndef GVIEW_H
00003 #define GVIEW_H
00004 #include<qobject.h>
00005 class MYGraphicsScene;
00006 #include"qgraphicsview.h"
00007 #include<qdrag.h>
00008 #include<qevent.h>
```

```
00009 #include<QMimeData>
00016 class GView  : public QGraphicsView
00017 {
00018
00019 public:
00024     GView(QObject *parent);
00028     ~GView();
00033     MYGraphicsScene* GScene();
00038     void mouseMoveEvent(QMouseEvent* event) override;
00043     void mouseReleaseEvent(QMouseEvent* event) override;
00048     void mousePressEvent(QMouseEvent* event) override;
00053     void dropEvent(QDropEvent* event)override;
00058     void dragEnterEvent(QDragEnterEvent* event) override;
00063     void dragMoveEvent(QDragMoveEvent* event) override;
00068     int lastselected = 0;
00069 };
00070 #endif
```

5.10 mainwindow.h

```
00001 #pragma once
00002 #ifndef MAINWINDOW_H
00003 #define MAINWINDOW_H
00004 #include"ui_mainwindow.h"
00005 #include<qscrollarea.h>
00006 #include"port.h"
00007 class View;
00008 class minimap;
00009 QT_BEGIN_NAMESPACE
00010 namespace Ui { class MainWindow; }
00011 QT_END_NAMESPACE
00012
00013 class MainWindow : public QMainWindow
00014 {
00015     Q_OBJECT
00016
00017 public:
00018     MainWindow(QWidget *parent = nullptr);
00019     ~MainWindow();
00020 public slots:
00021     void AddTemp(Port* p=new Port());
00022 public slots:
00023     void hidemMap();
00024 private:
00025     Ui::MainWindow *ui;
00026     View* v;
00027     minimap* minima;
00028     QScrollArea* TempList;
00029 };
00030 #endif // MAINWINDOW_H
```

5.11 minimap.h

```
00001 #pragma once
00002 #ifndef MINIMAP_H
00003 #define MINIMAP_H
00004 #include"qevent.h"
00005 class View;
00006 #include<qgraphicsview.h>
00013 class minimap  : public QGraphicsView
00014 {
00015     Q_OBJECT
00016
00017 public:
00022     minimap(View *v);
00027     void mouseMoveEvent(QMouseEvent* event)override;
00032     void mouseReleaseEvent(QMouseEvent* event)override;
00037     void mousePressEvent(QMouseEvent* event)override;
00042     void mouseDoubleClickEvent(QMouseEvent* event) override;
00043 #if QT_CONFIG(wheelevent)
00048     void wheelEvent(QWheelEvent* e);
00049 #endif
00053     ~minimap();
00057     View* v;
00058     float sclX, sclY;
00059 public slots:
00063     void rescale();
00064 };
00065 #endif
```

5.12 mygraphicsscene.h

```
00001 #pragma once
00002 #ifndef MYGRAPHICSSCENE_H
00003 #define MYGRAPHICSSCENE_H
00004 #include<QGraphicsScene>
00005 #include<QPainter>
00006 #include"view.h"
00013 class MYGraphicsScene : public QGraphicsScene
00014 {
00015 public:
00019     View* Mview;
00023     int _scale = 100;
00028     explicit MYGraphicsScene(QObject *parent = nullptr);
00029 private:
00033     int lastselected = 0;
00039     void drawBackground(QPainter* painter, const QRectF& rect) override;
00040 };
00041
00042 #endif // MYGRAPHICSSCENE_H
```

5.13 NewPinWhire.h

```
00001 #pragma once
00002 #ifndef NEWPINWHIRE_H
00003 #define NEWPINWHIRE_H
00004 #include<qthread.h>
00005 #include<qobject.h>
00006 class CustomColliderLineRecoursive;
00007 #include"CustomColliderLineRecoursive.h"
00008 class Pin;
00015 class NewPinWhire  : public CustomColliderLineRecoursive
00016 {
00017     Q_OBJECT
00018 public:
00030     NewPinWhire(Pin* p, QThread* th);
00034     ~NewPinWhire();
00039     bool hasConection();
00044     QColor color() override;
00045 private:
00049     Pin* pin;
00050 };
00051 #endif
```

5.14 NewWhire.h

```
00001 #ifndef NEWWHIRE_H
00002 #define NEWWHIRE_H
00003 #pragma once
00004 #include"CustomColliderLineRecoursive.h"
00005 class Pin;
00006 class AddWhireCommand;
00007 class Chain;
00015 class NewWhire  : public CustomColliderLineRecoursive
00016 {
00017     //Q_OBJECT
00018 public:
00022     Pin *p1 = nullptr, *p2 = nullptr;
00026     Chain* chain = nullptr;
00033     NewWhire(Pin*pp1,Pin*pp2,AddWhireCommand* comm);
00037     ~NewWhire();
00041     AddWhireCommand* command;
00047     static void AddComandW(Pin* p1,Pin* p2);
00052     QColor color() override;
00053 };
00054 #endif
```

5.15 pin.h

```
00001 #pragma once
00002 #ifndef PIN_H
00003 #define PIN_H
00004 #include<QThread>
00005 #include"ui_pin.h"
00006 #include<qgraphicsproxywidget.h>
00007 #include<QColorDialog>
```

```
00008 #include<QLineEdit>
00009 class Port;
00010 class NewPinWhire;
00011 class AddPinComand;
00012 class Chain;
00013 class Dot;
00014 namespace Ui {
00015 class Pin;
00016 }
00020 class Pin : public QLineEdit
00021 {
00022     Q_OBJECT
00023
00024 public:
00031     explicit Pin(Port* port,bool bl=true,QLineEdit *parent = nullptr);
00035     ~Pin();
00040     int index();
00045     const QString name();
00050     void name(QString name);
00051 public slots:
00055     void Update(bool dotupd);
00056 signals:
00060     void updSignal(bool upddots);
00061 public:
00066     void EmitUpd(bool dotold=false);
00071     NewPinWhire* getpinWhire();
00075     Port* parCon;
00081     virtual Dot* dot(bool recalc = false);
00086     virtual void dot(Dot* d);
00087 private:
00092     void mousePressEvent(QMouseEvent* event) override;
00096     virtual void dragEnterEvent(QDragEnterEvent* event) override;
00100     virtual void dropEvent(QDropEvent* event) override;
00104     virtual void dragMoveEvent(QDragMoveEvent* event) override;
00105 public:
00110     Dot* coredot();
00115     qreal x();
00119     qreal y();
00124     void pinWhire(bool show=true);
00129     void PinWUpd(bool upd = true);
00133     Chain* chain;
00137     AddPinComand* command=nullptr;
00138 protected:
00143     virtual QMenu* ContextMenu();
00144     Ui::Pin *ui;
00148     bool upd = false;
00149 public slots:
00153     void RemoveFromChain();
00157     void Remove();
00158 protected slots:
00162     virtual void showContextMenu(const QPoint& pos);
00166     void ChangeColor();
00167 private:
00168     bool updaterbl;
00172     Dot* d;
00176     Dot* cored;
00180     NewPinWhire* pinW;
00184     QThread* thread;
00188     QString _name;
00189
00190 };
00191
00192 #endif // PIN_H
```

5.16 PinTemplate.h

```
00001 #pragma once
00002 #ifndef PINTEMPLATE_H
00003 #define PINTEMPLATE_H
00004 #include"pin.h"
00005 class Port;
00006 #include<qobject.h>
00013 class PinTemplate  : public Pin
00014 {
00015     Q_OBJECT
00016
00017 public:
00022     PinTemplate(Port* p);
00027     virtual void dragEnterEvent(QDragEnterEvent* event) override;
00031     virtual void dropEvent(QDropEvent* event) override;
00035     virtual void dragMoveEvent(QDragMoveEvent* event) override;
00039     ~PinTemplate();
00040 };
00041 #endif
```

5.17 port.h

```
00001 #pragma once
00002 #ifndef PORT_H
00003 #define PORT_H
00004 #include"ui_port.h"
00005 #include"qobject.h"
00006 #include<QMenu>
00007 #include <QSpacerItem>
00008 #include<qwidget.h>
00009 class AddComand;
00010 class Pin;
00011 class ProxyRectPort;
00012 namespace Ui {
00013 class Port;
00014 }
00015
00022 class Port : public QWidget
00023 {
00024     Q_OBJECT
00025
00026 public:
00031     qreal x();
00036     qreal y();
00042     explicit Port(AddComand* com, QWidget *parent = nullptr);
00046     Port();
00050     virtual ~Port();
00055     QString name();
00060     void name(QString str);
00064     static QVector<Port*> portsVector;
00069     QVector<Pin*>pins();
00073     AddComand* adcom;
00077     void Remove();
00085     virtual Pin* addPin(QString name = "", int index = -1,bool bl=true);
00086
00087 private slots:
00091     void showContextMenu(const QPoint& pos);
00092     void on_pushButton_clicked();
00098     Pin* addPinSl(QString name="");
00102     void RemoveSL();
00103 public slots:
00107     void Update(bool updF=false);
00112     ProxyRectPort* proxy();
00117     void proxy(ProxyRectPort*prox);
00118
00119
00120 protected:
00121     Ui::Port *ui;
00126     virtual QMenu* ContextMenu();
00130     ProxyRectPort* _proxy;
00134     QSpacerItem* spacer;
00135 };
00136
00137 #endif // PORT_H
```

5.18 PortTemplate.h

```
00001 #pragma once
00002 #ifndef PORTTEMPLATE_H
00003 #define PORTTEMPLATE_H
00004 #include"qobject.h"
00005 #include"ui_port.h"
00006 #include <qopenglfunctions.h>
00007 #include"port.h"
00008 class PortTwmplateObject;
00009 class SaveTemplates;
00010 class Pin;
00011
00012 namespace Ui {
00013     class Port;
00014 }
00015
00019 class PortTemplate  : public Port
00020 {
00021     Q_OBJECT
00022
00023 public:
00028     PortTemplate(PortTwmplateObject* o);
00035     Pin* addPinn(QString name = "", int index = -1);
00039     ~PortTemplate();
00040 private:
00044     PortTwmplateObject* object;
00045 private slots:
```

```
00049     Pin* addPinSl(QString name = "");
00053     void RemoveSL();
00058     void closeEvent(QCloseEvent* event) override;
00063     virtual QMenu* ContextMenu();
00064 };
00065 #endif
```

5.19 PortTwmplateObject.h

```
00001 #pragma once
00002 #ifndef PORTTWMPLATEOBJECT_H
00003 #define PORTTWMPLATEOBJECT_H
00004 #include<qobject.h>
00005 #include<qwidget.h>
00006 #include"qpushbutton.h"
00007 #include"qlineedit.h"
00008 class PortTemplate;
00009
00013 class PortTwmplateObject  : public QWidget
00014 {
00015     Q_OBJECT
00016
00017 public:
00021     PortTemplate* templ;
00025     QPushButton* editpb;
00029     QPushButton* delpb;
00033     QLineEdit* name;
00040     void dragEnterEvent(QDragEnterEvent* event) override;
00047     void dropEvent(QDropEvent* event) override;
00054     void dragMoveEvent(QDragMoveEvent* event) override;
00061     void mousePressEvent(QMouseEvent* event) override;
00062
00063 public slots:
00067     void editPush();
00071     void delPush();
00072 public:
00076     PortTwmplateObject();
00080     ~PortTwmplateObject();
00081
00082 };
00083 #endif
```

5.20 proxyrectport.h

```
00001 #pragma once
00002
00003 #ifndef PROXYRECTPORT_H
00004 #define PROXYRECTPORT_H
00005 #include<qgraphicsitem.h>
00006 #include<QPainter>
00007 class Port;
00011 class ProxyRectPort: public QObject, public QGraphicsItem
00012 {
00013     Q_OBJECT
00014 public:
00019     ProxyRectPort(Port*port);
00023     void EmitMove();
00029     QPainterPath shape()const override;
00035     QRectF boundingRect() const override;
00041     void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override;
00046     QRectF geometry();
00051     void geometry(QRectF rec);
00056     void setconnector(Port* port);
00061     Port *getport();
00066     QColor color();
00071     void color(QColor c);
00075     ~ProxyRectPort();
00081     void Update(bool upd=false, int i=0);
00086     int type() const override;
00091     void ColiderCheck(bool upd=true);
00097     void ProxyColider(int xx=0,int yy=0);
00102     qreal XX();
00107     qreal YY();
00112     void XX(qreal x);
00117     void YY(qreal y);
00122     qreal left() {return mapToScene(boundingRect().left(), 0).x(); }
00127     qreal right() { return mapToScene(boundingRect().right(), 0).x(); }
00132     qreal bottom() { return mapToScene(0, boundingRect().bottom()).y(); }
00137     qreal top() { return mapToScene(0, boundingRect().top()).y(); }
00142     QPointF center() { return mapToScene(boundingRect().center()); }
```

```
00143 signals:
00147         void strartMove();
00148 private:
00152     Port* port;
00156     QColor colo;
00160     QRectF rect;
00161
00162 };
00163
00164 #endif // PROXYRECTPORT_H
```

5.21 RemovePinCommand.h

```
00001 #pragma once
00002 #ifndef REMOVEPINCOMMAND_H
00003 #define REMOVEPINCOMMAND_H
00004 #include<QUndoCommand>
00005 class Pin;
00006 class Port;
00007 class AddPinComand;
00011 class RemovePinCommand  : public QUndoCommand
00012 {
00013
00014 public:
00019     RemovePinCommand(Pin*p);
00023     ~RemovePinCommand();
00027     void undo() override;
00031     void redo() override;
00035     int pos = 0;
00039     QString name = "";
00043     AddPinComand* pinc;
00047     Port* prt;
00051     Pin* pin;
00052
00053 };
00054 #endif
```

5.22 RemovePortComand.h

```
00001 #pragma once
00002 #ifndef REMOVECOMAND_H
00003 #define REMOVECOMAND_H
00004 #include<QUndoCommand>
00005 class Port;
00006 class AddComand;
00007 class View;
00008
00012 class RemovePortComand  : public QUndoCommand
00013 {
00014 public:
00018     RemovePortComand(Port* p);
00022     ~RemovePortComand();
00026     void undo() override;
00030     void redo() override;
00034     QString name;
00038     AddComand* addcom;
00042     View* v;
00046     int xx, yy;
00047 };
00048 #endif
```

5.23 SaveTemplates.h

```
00001 #pragma once
00002 #ifndef SAVETEMPLATES_H
00003 #define SAVETEMPLATES_H
00004 #include <QObject>
00005 #include <QFile>
00006 #include <QTextStream>
00007 #include<qsettings.h>
00008 class PortTwmplateObject;
00012 class SaveTemplates  : public QObject
00013 {
00014     Q_OBJECT
00015
00016 public:
```

```
00021     static void Save(PortTwmplateObject* o);
00026     static QList<QPair<QStringList, QString» Load();
00031     static void Del(QString path);
00032 };
00033 #endif
```

5.24 view.h

```
00001 #pragma once
00002 #ifndef VIEW_H
00003 #define VIEW_H
00004 #include<qshortcut.h>
00005 #include<qundostack.h>
00006 #include<qscrollbar.h>
00007 #include<QMenu>
00008 #include<qframe.h>
00009 class GView;
00010 class MYGraphicsScene;
00011 class AddComand;
00015 class View : public QFrame
00016 {
00017     Q_OBJECT
00018 public:
00022     View();
00027     MYGraphicsScene* GScene();
00032     void GScene(MYGraphicsScene* scene);
00037     QColor backGroundColor();
00042     void backGroundColor(QColor color);
00047     GView* view();
00051     QUndoStack* stack;
00056     int scale();
00057
00058 protected:
00062     GView* graphicsview;
00063     int _scale;
00068     virtual QMenu* ContextMenu();
00069 protected slots:
00073     virtual void showContextMenu(const QPoint& pos);
00074 signals:
00078     void SceneChanged();
00079 public slots:
00083     void stckUndo();
00087     void stckRedo();
00091     void AdddPortSL();
00095     void saveImage();
00099     void AdddPortSL(int x, int y, QString name);
00100
00101 public:
00102
00106     AddComand* AdddPort();
00110     AddComand* AdddPort(int x, int y, QString name);
00111
00115     void stackPush(QUndoCommand* com)
00116     {
00117         stack->push(com);
00118     }
00119 #if QT_CONFIG(wheelevent)
00124         void wheelEvent(QWheelEvent* e);
00125 #endif
00126 };
00127
00128 #endif // VIEW_H
```

5.25 WhireRemoveComand.h

```
00001 #pragma once
00002 #ifndef WHIREREMOVECOMAND_H
00003 #define WHIREREMOVECOMAND_H
00004 #include<QUndoCommand>
00005 class NewWhire;
00006 class AddPinComand;
00007 class AddWhireCommand;
00011 class WhireRemoveComand  : public QUndoCommand
00012 {
00013 public:
00018     WhireRemoveComand(NewWhire* w);
00022     ~WhireRemoveComand();
00026     void undo() override;
00030     void redo() override;
00034     AddPinComand* p1, * p2;
```

```
00038     AddWhireCommand* wc;
00039 };
00040 #endif
```

Предметный указатель