Кабелион графика

Создано системой Doxygen 1.9.7

| 1 Иерархический список классов | 1 |
|--|---|
| 1.1 Иерархия классов | 1 |
| 2 Алфавитный указатель классов | 3 |
| 2.1 Классы | 3 |
| 3 Список файлов | 5 |
| 3.1 Файлы | 5 |
| 4 Классы | 7 |
| 4.1 Класс AddComand | 7 |
| 4.1.1 Подробное описание | 3 |
| 4.1.2 Конструктор(ы) | 3 |
| 4.1.2.1 AddComand() | 3 |
| 4.2 Класс AddPinComand | 3 |
| 4.2.1 Подробное описание | 9 |
| 4.2.2 Конструктор(ы) | 9 |
| 4.2.2.1 AddPinComand() | 9 |
| 4.3 Класс AddWhireCommand |) |
| 4.3.1 Подробное описание |) |
| 4.3.2 Конструктор(ы) |) |
| 4.3.2.1 AddWhireCommand() |) |
| 4.4 Класс Chain | 1 |
| 4.4.1 Подробное описание | 2 |
| 4.4.2 Конструктор(ы) | 2 |
| 4.4.2.1 Chain() | 2 |
| 4.4.3 Методы | 2 |
| 4.4.3.1 AddPin() | 2 |
| 4.4.3.2 moveToChain() | 2 |
| 4.4.3.3 RemovePin() | 2 |
| 4.5 Класс ChainTable | 3 |
| 4.5.1 Подробное описание | 4 |
| 4.5.2 Методы | 4 |
| 4.5.2.1 CellChange | 4 |
| 4.5.2.2 CellClck | 4 |
| 4.6 Класс CustomColliderLineRecoursive | 5 |
| 4.6.1 Подробное описание | 3 |
| 4.6.2 Конструктор(ы) | |
| 4.6.2.1 CustomColliderLineRecoursive() | |
| 4.6.3 Методы | |
| 4.6.3.1 boundingRect() | |
| 4.6.3.2 color() | |
| 4.6.3.3 FixColliding | |
| 4.6.3.4 JumpFrom | |

| $4.6.3.5 	ext{ paint}() \dots \dots$ | . 18 |
|---|----------|
| $4.6.3.6 \text{ setFixWay}() \dots \dots$ | . 18 |
| 4.6.3.7 shape() | . 18 |
| 4.7 Класс Dot | . 19 |
| 4.7.1 Подробное описание | . 20 |
| 4.7.2 Конструктор(ы) | . 20 |
| 4.7.2.1 Dot() | . 20 |
| 4.7.3 Методы | . 21 |
| 4.7.3.1 boundingRect() | . 21 |
| 4.7.3.2 paint() | . 21 |
| $4.7.3.3 \text{ setBig}() \dots \dots$ | . 21 |
| 4.7.3.4 setColor() | . 21 |
| 4.7.3.5 setPin() | . 21 |
| 4.7.3.6 setTriangle() | . 22 |
| 4.7.3.7 shape() | . 22 |
| 4.7.3.8 type() | . 22 |
| 4.7.3.9 x() | . 22 |
| 4.7.3.10 y() | . 22 |
| 4.8 Класс GItemFrame | . 23 |
| 4.8.1 Подробное описание | . 23 |
| 4.8.2 Методы | . 23 |
| 4.8.2.1 boundingRect() | . 23 |
| 4.8.2.2 paint() | . 23 |
| 4.8.2.3 shape() | . 24 |
| 4.9 Класс GView | . 24 |
| 4.9.1 Подробное описание | . 25 |
| 4.9.2 Конструктор(ы) | . 25 |
| 4.9.2.1 GView() | . 25 |
| 4.9.3 Методы | . 25 |
| 4.9.3.1 dragEnterEvent() | . 25 |
| $4.9.3.2 \operatorname{dragMoveEvent}() \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots$ | . 25 |
| 4.9.3.3 dropEvent() | . 25 |
| 4.9.3.4 GScene() | . 26 |
| 4.9.3.5 mouseMoveEvent() | . 26 |
| 4.9.3.6 mousePressEvent() | . 26 |
| 4.9.3.7 mouseReleaseEvent() | . 26 |
| 4.9.4 Данные класса | . 27 |
| 4.9.4.1 lastselected | . 27 |
| 4.10 Класс MainWindow | . 27 |
| 4.11 Класс minimap | . 28 |
| 4.11.1 Подробное описание | . 28 |
| 4.11.2 Конструктор(ы) | . 28 |
| 4.11.2.1 minimap() | . 28 |

| 4.11.3 Методы | 29 |
|---|--------|
| $4.11.3.1 \ mouseDoubleClickEvent() \ \dots \dots \dots \dots \dots \dots \dots \dots$ | 29 |
| 4.11.3.2 mouseMoveEvent() | 29 |
| 4.11.3.3 mousePressEvent() | 29 |
| 4.11.3.4 mouseReleaseEvent() | 29 |
| 4.12 Kласс MYGraphicsScene | 30 |
| 4.12.1 Подробное описание | 30 |
| 4.12.2 Конструктор(ы) | 30 |
| 4.12.2.1 MYGraphicsScene() | 30 |
| 4.13 Класс NewPinWhire | 31 |
| 4.13.1 Подробное описание | 32 |
| 4.13.2 Конструктор(ы) | 33 |
| 4.13.2.1 NewPinWhire() | 33 |
| 4.13.3 Методы | 33 |
| 4.13.3.1 color() | 33 |
| 4.13.3.2 hasConection() | 33 |
| 4.14 Kласс NewWhire | 34 |
| 4.14.1 Подробное описание | 36 |
| 4.14.2 Конструктор(ы) | 36 |
| 4.14.2.1 NewWhire() | 36 |
| 4.14.3 Методы | 36 |
| $4.14.3.1 \text{ AddComandW}() \dots \dots$ | 36 |
| $4.14.3.2 \operatorname{color}() \dots \dots$ | 36 |
| 4.15 Класс Pin | 37 |
| 4.15.1 Подробное описание | 39 |
| 4.15.2 Конструктор(ы) | 39 |
| 4.15.2.1 Pin() | 39 |
| 4.15.3 Методы | 39 |
| 4.15.3.1 ContextMenu() | 39 |
| $4.15.3.2 \operatorname{coredot}() \dots \dots$ | 39 |
| $4.15.3.3 \det() [1/2] \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots$ | 40 |
| $4.15.3.4 dot() [2/2] \dots $ | 40 |
| $4.15.3.5 \; \mathrm{EmitUpd}() \; \ldots \; $ | 40 |
| 4.15.3.6 getpinWhire() | 40 |
| $4.15.3.7 \text{ index}() \dots \dots$ | 41 |
| 4.15.3.8 name() [1/2] | 41 |
| 4.15.3.9 name() [2/2] | 41 |
| 4.15.3.10 pinWhire() | 41 |
| 4.15.3.11 PinWUpd() | 41 |
| 4.15.3.12 x() | 43 |
| 4.16 Класс PinTemplate | 43 |
| 4.16.1 Подробное описание | 45 |
| 4.16.2 Kohempykton(ki) | 15 |

| 4.16.2.1 PinTemplate() | 45 |
|--|----|
| 4.16.3 Методы | 46 |
| 4.16.3.1 dragEnterEvent() | 46 |
| 4.16.3.2 dragMoveEvent() | 46 |
| 4.16.3.3 dropEvent() | 46 |
| 4.17 Класс Port | 46 |
| 4.17.1 Подробное описание | 48 |
| 4.17.2 Конструктор(ы) | 48 |
| 4.17.2.1 Port() | 48 |
| 4.17.3 Методы | 48 |
| 4.17.3.1 addPin() | 48 |
| 4.17.3.2 ContextMenu() | 49 |
| 4.17.3.3 name() [1/2] | 49 |
| 4.17.3.4 name() [2/2] | 49 |
| $4.17.3.5 \; \mathrm{pins}() \;\; \ldots \; \ldots \;$ | 49 |
| 4.17.3.6 proxy [1/2] | 50 |
| 4.17.3.7 proxy [2/2] | 50 |
| 4.17.3.8 x() | 50 |
| 4.17.3.9 y() | 50 |
| 4.18 Класс PortTemplate | 51 |
| 4.18.1 Подробное описание | 52 |
| 4.18.2 Конструктор(ы) | 52 |
| 4.18.2.1 PortTemplate() | 52 |
| 4.18.3 Методы | 53 |
| 4.18.3.1 addPinn() | 53 |
| 4.19 Класс PortTwmplateObject | 53 |
| 4.19.1 Подробное описание | 54 |
| 4.19.2 Методы | 54 |
| 4.19.2.1 dragEnterEvent() | 54 |
| 4.19.2.2 dragMoveEvent() | 54 |
| 4.19.2.3 dropEvent() | 55 |
| 4.19.2.4 mousePressEvent() | 55 |
| 4.20 Kласс ProxyRectPort | 55 |
| 4.20.1 Подробное описание | 57 |
| 4.20.2 Конструктор(ы) | 57 |
| 4.20.2.1 ProxyRectPort() | 57 |
| 4.20.3 Методы | 57 |
| 4.20.3.1 bottom() | 57 |
| 4.20.3.2 boundingRect() | 57 |
| 4.20.3.3 center() | 57 |
| 4.20.3.4 ColiderCheck() | 57 |
| $4.20.3.5 \; \mathrm{color}() \; [1/2] \;\; \ldots \; \ldots \;$ | 58 |
| $4.20.3.6 \; \mathrm{color}() \; [2/2] \;\; \ldots \;\;$ | 58 |
| | |

| $4.20.3.7 \text{ geometry}() [1/2] \dots \dots \dots \dots \dots \dots \dots \dots \dots$ | 58 |
|--|----|
| $4.20.3.8 \; \text{geometry}() \; [2/2] \; \dots \; \dots$ | 58 |
| 4.20.3.9 getport() | 59 |
| $4.20.3.10 \operatorname{left}()$ | 59 |
| 4.20.3.11 paint() | 59 |
| 4.20.3.12 ProxyColider() | 59 |
| $4.20.3.13 \; \mathrm{right}() \;\; \ldots \;\;$ | 59 |
| 4.20.3.14 setconnector() | 60 |
| $4.20.3.15 \; \mathrm{shape}() \; \ldots \; $ | 60 |
| 4.20.3.16 top() | 60 |
| 4.20.3.17 type() | 60 |
| 4.20.3.18 Update() | 60 |
| 4.20.3.19 XX() [1/2] | 61 |
| $4.20.3.20~{ m XX}()~{ m [2/2]}$ | 61 |
| $4.20.3.21 \; { m YY}() \; { m [1/2]} \; \ldots \; $ | 61 |
| 4.20.3.22 YY() [2/2] | 61 |
| 4.21 Класс RemovePinCommand | 62 |
| 4.21.1 Подробное описание | 62 |
| 4.21.2 Конструктор(ы) | 62 |
| 4.21.2.1 RemovePinCommand() | 62 |
| 4.22 Kласс RemovePortComand | 63 |
| 4.22.1 Подробное описание | 63 |
| 4.22.2 Конструктор(ы) | 64 |
| 4.22.2.1 RemovePortComand() | 64 |
| 4.23 Класс SaveTemplates | 64 |
| 4.23.1 Подробное описание | 64 |
| 4.23.2 Методы | 64 |
| 4.23.2.1 Del() | 64 |
| $4.23.2.2 \; \mathrm{Load}() \; \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots \ldots$ | 65 |
| 4.23.2.3 Save() | 65 |
| 4.24 Класс View | 65 |
| 4.24.1 Подробное описание | 67 |
| 4.24.2 Методы | 67 |
| 4.24.2.1 backGroundColor() [1/2] | 67 |
| 4.24.2.2 backGroundColor() [2/2] | 67 |
| 4.24.2.3 ContextMenu() | 67 |
| 4.24.2.4 GScene() [1/2] | 68 |
| $4.24.2.5 \; \mathrm{GScene}() \; \mathrm{[2/2]} \; \ldots \; $ | 68 |
| w. The state of th | 68 |
| 4.24.2.7 view() | 68 |
| 4.25 Kласс WhireRemoveComand | 69 |
| 4.25.1 Подробное описание | 69 |
| $4.25.2\ m Kohctpyktop(ы)$ | 69 |
| | |

Предметный указатель

| 4.25.2.1 WhireRemoveComand() | 69 |
|---|----|
| 5 Файлы | 71 |
| 5.1 AddComand.h | 71 |
| 5.2 AddPinComand.h | 71 |
| 5.3 AddWhireCommand.h | 71 |
| 5.4 chain.h | 72 |
| 5.5 chaintable.h | 72 |
| $5.6\ Custom Collider Line Recoursive. h \qquad . $ | 72 |
| 5.7 Dot.h | 73 |
| 5.8 GItemFrame.h | 74 |
| 5.9 GView.h | 74 |
| 5.10 mainwindow.h | 75 |
| 5.11 minimap.h | 75 |
| 5.12 mygraphicsscene.h | 76 |
| 5.13 NewPinWhire.h | 76 |
| 5.14 NewWhire.h | 76 |
| 5.15 pin.h | 76 |
| 5.16 PinTemplate.h | 77 |
| 5.17 port.h | 78 |
| 5.18 PortTemplate.h | 78 |
| 5.19 PortTwmplateObject.h | 79 |
| 5.20 proxyrectport.h | 79 |
| $5.21 \; Remove Pin Command. h \; . \; . \; . \; . \; . \; . \; . \; . \; . $ | 80 |
| 5.22 RemovePortComand.h | 80 |
| 5.23 SaveTemplates.h | 80 |
| 5.24 view.h | 81 |
| 5.25 WhireRemoveComand.h | 81 |
| Предметный указатель | 83 |

Иерархический список классов

1.1 Иерархия классов

| DayagaM | TTTOOOD | |
|---------|---------|--|

| QFrame | |
|------------------------------|----|
| View | 65 |
| QGraphicsItem | |
| GItemFrame | 23 |
| ProxyRectPort | 55 |
| QGraphicsObject | |
| CustomColliderLineRecoursive | 15 |
| NewPinWhire | 31 |
| NewWhire | 34 |
| Dot | 19 |
| QGraphicsScene | |
| MYGraphicsScene | 30 |
| QGraphicsView | |
| GView | 24 |
| minimap | 28 |
| QLineEdit | |
| Pin | 37 |
| PinTemplate | 43 |
| QMainWindow | |
| MainWindow | 27 |
| QObject | |
| Chain | 11 |
| ProxyRectPort | 55 |
| SaveTemplates | 64 |
| QTableWidget | |
| ChainTable | 13 |
| QUndoCommand | |
| AddComand | 7 |
| AddPinComand | 8 |
| AddWhireCommand | 10 |
| RemovePinCommand | 62 |
| RemovePortComand | 63 |
| WhireRemoveComand | 69 |
| QWidget | |
| Port | 46 |
| PortTemplate | 51 |
| PortTwmplateObject | 53 |
| | |

| перархический список классов | Иерархический | список | классов |
|------------------------------|---------------|--------|---------|
|------------------------------|---------------|--------|---------|

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

| AddComand | |
|--|----|
| Класс наследник от QUndoCommand на добавление нового разъема на сцену | 7 |
| AddPinComand | |
| Класс наследник от QUndoCommand на добавление нового контакта в разъем | 8 |
| AddWhireCommand | |
| Класс наследник от QUndoCommand на добавление нового разъема на сцену | 10 |
| Chain | |
| Класс Цепи Содержит соедененные в цепь контакты | 11 |
| ChainTable | |
| Табличный вид цепей | 13 |
| CustomColliderLineRecoursive | |
| Линия с колизионной корекцией | 15 |
| Dot | |
| Класс Точки дляпостроения линий | 19 |
| GItemFrame | |
| Красная рамка | 23 |
| GView | |
| Hаследование QGraphicsView | 24 |
| MainWindow | 27 |
| minimap | |
| Мини карта | 28 |
| MYGraphicsScene | |
| Hаследник от QGraphicsScene | 30 |
| NewPinWhire | |
| Горизонтальный провод идущий от контакта | 31 |
| NewWhire | |
| Вертикальный провод соеденяющий 2 контакта | 34 |
| Pin | |
| Контакт | 37 |
| PinTemplate | |
| Контакт из шаблонов (библиотеки) | 43 |
| Port | |
| Разъем на сцене | 46 |
| PortTemplate | |
| Шаблон Разъема; перегрузка разъема (убрано перемещение итд) | 51 |

| PortTwmplateObject | |
|--|----|
| Объект для шаблона разъема | 53 |
| ProxyRectPort | |
| Прослойка для перемещения разъема по сцене | 55 |
| RemovePinCommand | |
| Undo Redo Удаление контакта | 62 |
| RemovePortComand | |
| Undo Redo Удаление разъема | 63 |
| SaveTemplates | |
| Static сохранение шаблонов | 64 |
| View | |
| "Дисплей" с вложеным отображением | 65 |
| WhireRemoveComand | |
| Команда удаления провода Undo Redo | 69 |

Список файлов

3.1 Файлы

Полный список документированных файлов.

| AddComand.h | 1 |
|--------------------------------|---|
| AddPinComand.h | 1 |
| AddWhireCommand.h | 1 |
| chain.h | 2 |
| chaintable.h 75 | 2 |
| CustomColliderLineRecoursive.h | 2 |
| Dot.h | 3 |
| GItemFrame.h | 4 |
| GView.h | 4 |
| mainwindow.h | 5 |
| minimap.h | 5 |
| mygraphicsscene.h | 6 |
| NewPinWhire.h | 6 |
| NewWhire.h | 6 |
| pin.h | 6 |
| PinTemplate.h | 7 |
| port.h | 8 |
| PortTemplate.h | 8 |
| PortTwmplateObject.h | 9 |
| proxyrectport.h | 9 |
| RemovePinCommand.h | 0 |
| RemovePortComand.h | 0 |
| SaveTemplates.h | 0 |
| view.h | 1 |
| White Personal h | 1 |

6 Список файлов

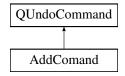
Классы

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

• \sim 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()

```
\label{eq:AddComand} \begin{split} AddComand :: & AddComand \; ( \\ & View * v, \\ & \text{int } x, \\ & \text{int } y, \\ & QString \; name = \text{```'} \; ) \end{split}
```

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 ста | |
|------|---|--|
| x | Координата х разъема на сцене | |
| У | Координата у разъема на сцене | |
| name | Имя раъема | |

Объявления и описания членов классов находятся в файлах:

- AddComand.h
- AddComand.cpp

4.2 Kласс 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 р.

• ~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()

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

Аргументы

| р 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,.

• \sim 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()

```
\label{eq:AddWhireCommand:AddWhireCommand (AddPinCommand * p11, $$AddPinCommand * p22 )$}
```

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

4.4 Класс Chain

Аргументы

| p11 | Первый контакт соедененный проводом | |
|-----|-------------------------------------|--|
| p22 | Второй контакт соедененный проводом | |

Объявления и описания членов классов находятся в файлах:

- AddWhireCommand.h
- $\bullet \ \, {\rm AddWhireCommand.cpp}$

4.4 Класс Chain

Класс Цепи Содержит соедененные 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 соедененные 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 ( \label{eq:pin*p} {\sf Pin}*{\tt p}\;)
```

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

Аргументы

```
р Контакт
```

4.4.3.2 moveToChain()

```
void Chain::moveToChain ( \frac{\text{Chain} * \text{chain}}{\text{Chain}}
```

Соединение данной цепи и chain в одну

Аргументы

chain

4.4.3.3 RemovePin()

void Chain::RemovePin (

4.5 Класс ChainTable

$$Pin * p$$
)

Удаляет контакт из цепи

Аргументы

р Удаляемый контакт

Объявления и описания членов классов находятся в файлах:

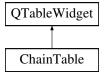
- chain.h
- · chain.cpp
- main.cpp

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)

Оброботка клика на +.

Открытые члены

• ChainTable ()

Конструктор

• \sim 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

```
\label{eq:condition} \begin{tabular}{ll} \be
```

Пересчет цепей по обновлению в таблице

Аргументы

| row | строка | |
|--------|---------|--|
| column | колонка | |

$4.5.2.2 \quad \mathrm{CellClck}$

```
void Chain
Table::CellClck ( \label{eq:condition} \text{int row,} \label{eq:column} \text{int colum )} \quad [\text{slot}]
```

Оброботка клика на +.

Аргументы

| row | строка | |
|-------|---------|--|
| colum | колонка | |

Объявления и описания членов классов находятся в файлах:

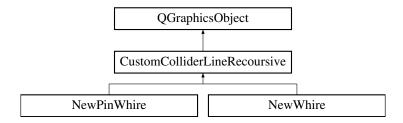
- chaintable.h
- chaintable.cpp

4.6 Kласс 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

 $\bullet \ CustomColliderLineRecoursive \ (bool\ Vertical_f_Horiz_t, Dot\ *d1, Dot\ *d2, CustomColliderLineRecoursive \ *parent=nullptr)$

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

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

 $\bullet \ \, \mathbf{ProxyRectPort} * \mathbf{lastcolide} = \! \mathbf{nullptr}$

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()

```
\label{eq:customColliderLineRecoursive} CustomColliderLineRecoursive ( $$bool Vertical_f_Horiz_t$, $$Dot * d1$, $$Dot * d2$, $$CustomColliderLineRecoursive * parent = nullptr )
```

Просто QGraphicsItem.

Аргументы

| Vertical_f_Horiz← | Горизонтальная или вертикальная линия false - Вертикальная true - |
|-------------------|---|
| _t | 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
cm. 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
{\bf void}\ {\bf CustomColliderLineRecoursive::} {\bf FixColliding}\ (
             int times = 0) [slot]
Пересчет коллизии
Аргументы
 times
         чЧто бы не уйти в вечный цикл при неудачной коллизии
4.6.3.4 JumpFrom
{\bf void}\ {\bf CustomColliderLineRecoursive::JumpFrom}\ (
             QGraphicsItem * itm ) [slot]
уход от обойденных ранее объектов
```

Создано системой Doxygen

Аргументы

itm

```
4.6.3.5 paint()
```

```
\label{eq:constraint} $$ void CustomColliderLineRecoursive::paint ($$ QPainter* painter, $$ const QStyleOptionGraphicsItem* option, $$ QWidget* widget ) [override]
```

Отрисовка

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

Аргументы

| painter | |
|---------|--|
| option | |
| widget | |

4.6.3.6 setFixWay()

```
void Custom
Collider
LineRecoursive::setFixWay ( {\bf ColisionFixWay~fw~})
```

Наработка на будущее...

Аргументы



4.6.3.7 shape()

 $\label{eq:QPainterPath} \ensuremath{\mathbf{CustomColliderLineRecoursive::shape}} \ensuremath{ \ (\)\ const} \quad \ensuremath{[\mathrm{override}]}$

Форма для отрисовки

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

Возвращает

Объявления и описания членов классов находятся в файлах:

- $\bullet \ Custom Collider Line Recoursive. h$
- $\bullet \ {\bf Custom Collider Line Recoursive.cpp}$

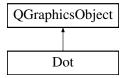
4.7 Класс Dot

4.7 Kласс 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)

 ${\bf QGraphics Item.}$

• int type () const override

для работы через GView

- int x ()
- int y ()
- void x (int x)

Устанавливает координату по оси х.

• void y (int y)

Устанавливает координату по оси у.

• 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 ()
    • \simDot ()
        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
    \bullet \  \, Custom Collider Line Recoursive * H dot
    • 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 )
\label{eq:QGraphicsItem.} QGraphicsItem.
см. https://doc.qt.io/qt-6/qgraphicsitem.html
Аргументы
 parent
```

4.7 Класс Dot 21

```
4.7.3
        Методы
4.7.3.1 boundingRect()
QRectF Dot::boundingRect ( ) const [override]
cm. https://doc.qt.io/qt-6/qgraphicsitem.html
4.7.3.2 paint()
void Dot::paint (
               QPainter * painter,
               const\ QStyleOptionGraphicsItem\ *\ option,
               QWidget * widget ) [override]
{\bf cm.}\ \ 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 \operatorname{setColor}()
{\bf void\ Dot::setColor\ (}
               QColor cl )
Устанавливает Цвет
Аргументы
      цвет
4.7.3.5 \operatorname{setPin}()
void Dot::setPin (
               Pin * p)
Закрепление за контактом
```

Аргументы \mathbf{p} 4.7.3.6 setTriangle() ${\bf void\ Dot::} {\bf setTriangle\ (}$ bool bl) Делает треугольной или круглой Аргументы bl 4.7.3.7 shape() $\label{eq:QPainterPath} \mbox{ QPainterPath Dot::shape () const } \mbox{ [override]}$ $\mathbf{cm.}\ \ \mathbf{https://doc.qt.io/qt\text{-}6/qgraphicsitem.html}$ 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 х

4.7.3.10 y()

int Dot::y()

Возвращает

Координату у

Объявления и описания членов классов находятся в файлах:

- Dot.h
- Dot.cpp

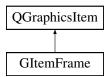
4.8 Класс GItemFrame 23

4.8 Класс GItemFrame

Красная рамка

```
#include <GItemFrame.h>
```

Граф наследования:GItemFrame:



Открытые члены

- ~GItemFrame () Деструктор
- QRectF boundingRect () const override
- QPainterPath shape () const override

4.8.1 Подробное описание

Красная рамка

```
4.8.2 Методы
```

```
4.8.2.1 boundingRect()
```

```
\label{eq:quantum} \mbox{QRectF GItemFrame::boundingRect ( ) const } \mbox{ [override]}
```

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

```
4.8.2.2 paint()
```

```
\label{eq:const_point} $$ void GItemFrame::paint ($$ QPainter * painter, $$ const QStyleOptionGraphicsItem * option, $$ QWidget * widget ) [override]
```

 $\mathbf{cm.}\ \ \mathbf{https://doc.qt.io/qt\text{-}6/qgraphicsitem.html}$

4.8.2.3 shape()

QPainterPath GItemFrame::shape () const [override]

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

Объявления и описания членов классов находятся в файлах:

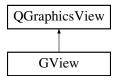
- GItemFrame.h
- GItemFrame.cpp

4.9 Класс GView

Hаследование QGraphicsView.

#include <GView.h>

Граф наследования:GView:



Открытые члены

• GView (QObject *parent)

QGraphicsView.

• \sim GView ()

Деструктор

• MYGraphicsScene * GScene ()

Сцена в данном отображении

- void mouseMoveEvent (QMouseEvent *event) override
 - Перегруз
- void mouse Release Event (QMouse
Event *event) override $\Pi eperpy 3$
- void mousePressEvent (QMouseEvent *event) override
- void dropEvent (QDropEvent *event) override

Перегруз

- void dragEnterEvent (QDragEnterEvent *event) override
 Перегруз
- void dragMoveEvent (QDragMoveEvent *event) override Перегруз

Открытые атрибуты

• int last selected = 0 Перегруз 4.9 Класс GView 25

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 )
{\bf QGraphics View.}
Аргументы
 parent
4.9.3 Методы
4.9.3.1 dragEnterEvent()
void \operatorname{GView}::\operatorname{dragEnterEvent} (
                QDragEnterEvent * event ) [override]
Перегруз
Аргументы
 event
4.9.3.2 dragMoveEvent()
void \ensuremath{\operatorname{GView}}\xspace:\ensuremath{\operatorname{dragMoveEvent}} (
                QDragMoveEvent * event ) [override]
Перегруз
Аргументы
 {\it event}
4.9.3.3 dropEvent()
void GView::dropEvent (
```

```
\label{eq:QDropEvent} \begin{tabular}{ll} QDropEvent * event \end{tabular} \begin{tabular}{ll} (event) & (event) \end{ta
Перегруз
 Аргументы
        event
4.9.3.4 GScene()
MYGraphicsScene * GView::GScene ( )
Сцена в данном отображении
Возвращает
4.9.3.5 mouseMoveEvent()
void GView::mouseMoveEvent ( \,
                                                                         \label{eq:QMouseEvent * event ) [override]} 
Перегруз
 Аргументы
       event
4.9.3.6 mousePressEvent()
void GView::mousePressEvent (
                                                                         \label{eq:QMouseEvent * event } \text{QMouseEvent * event }) \quad [\text{override}]
Перегруз
 Аргументы
       event
4.9.3.7 mouseReleaseEvent()
void GView::mouseReleaseEvent (
                                                                         \label{eq:QMouseEvent * event } \mbox{QMouseEvent * event )} \quad \mbox{[override]}
Перегруз
```

4.10 Класс MainWindow 27

Аргументы

event

4.9.4 Данные класса

4.9.4.1 lastselected

int GView::last selected = 0

Перегруз

Аргументы

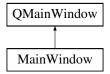
event

Объявления и описания членов классов находятся в файлах:

- GView.h
- GView.cpp

4.10 Класс MainWindow

Граф наследования: Main
Window:



Открытые слоты

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

Открытые члены

 $\bullet \ {\it MainWindow} \ ({\it QWidget *parent=nullptr})$

Объявления и описания членов классов находятся в файлах:

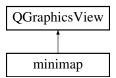
- mainwindow.h
- mainwindow.cpp

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

• \sim 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()

```
\label{eq:view} \begin{aligned} & \text{minimap::minimap (} \\ & & \text{View * v )} \end{aligned}
```

крадем сцену у v

4.11 Класс minimap

Аргументы

```
v Основное отображение
```

4.11.3 Методы

```
4.11.3.1 mouseDoubleClickEvent()
```

```
void minimap::mouseDoubleClickEvent ( \label{eq:QMouseEvent * event } \mbox{$\mathbb{Q}$ MouseEvent * event ) } \mbox{ [override]}
```

Перегруз

Аргументы

event

4.11.3.2 mouseMoveEvent()

```
void minimap::mouseMoveEvent ( \label{eq:QMouseEvent} Q MouseEvent * event \; ) \quad [override]
```

Перегруз

Аргументы

event

4.11.3.3 mousePressEvent()

```
void minimap::mousePressEvent ( \label{eq:QMouseEvent * event } \text{$\mathbb{Q}$ (override) }
```

Перегруз

Аргументы

event

4.11.3.4 mouseReleaseEvent()

```
void minimap::mouseReleaseEvent ( {\bf QMouseEvent * event } ) \quad [{\bf override}]
```

Перегруз

Аргументы

| event | |
|--------|--|
| CVCIII | |

Объявления и описания членов классов находятся в файлах:

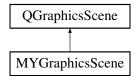
- minimap.h
- minimap.cpp

4.12 Класс MYGraphicsScene

Hаследник от QGraphicsScene.

#include <mygraphicsscene.h>

Граф наследования:MYGraphicsScene:



Открытые члены

• MYGraphicsScene (QObject *parent=nullptr) Стандартный конструктор

Открытые атрибуты

• View * Mview

Основное отображение

• int $_{\text{scale}} = 100$ Zoom.

4.12.1 Подробное описание

Hаследник 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()

```
\label{eq:main_scale} \begin{split} \mbox{MYGraphicsScene}: \mbox{MYGraphicsScene} \ ( \\ \mbox{QObject} * \mbox{parent} = \mbox{nullptr} \ ) \ \ \ \ \mbox{[explicit]} \end{split}
```

Стандартный конструктор

Аргументы

parent

Объявления и описания членов классов находятся в файлах:

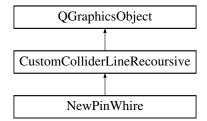
- · mygraphicsscene.h
- mygraphicsscene.cpp

4.13 Kласс 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

 $\bullet \ \, CustomColliderLineRecoursive \ (bool\ Vertical_f_Horiz_t, Dot\ *d1, Dot\ *d2, CustomColliderLineRecoursive \ *parent=nullptr) \\$

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 ( \frac{\text{Pin}*p,}{\text{QThread}*th} \label{eq:pinwhire}
```

Конструктор

Жрет контакт и поток в котором лежит этот контакт

По идее таки пошустрее в несколько потоков

Аргументы

| р | 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 - ecть; false - нет
```

Объявления и описания членов классов находятся в файлах:

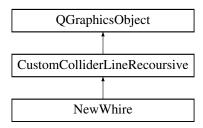
- NewPinWhire.h
- NewPinWhire.cpp

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

• \sim 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

 $\bullet \ CustomColliderLineRecoursive \ (bool\ Vertical_f_Horiz_t, Dot\ *d1, Dot\ *d2, CustomColliderLineRecoursive \ *parent=nullptr)$

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 4.14 Класс NewWhire 35

Открытые атрибуты

• Pin * p1 = nullptr

Контакты соедененные проводом

- Pin * p2 = nullptr
- Chain * chain = nullptr

Цепь в которую входит данный провод и соедененные 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 Jump Derection = 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
- $\bullet \ \, {\bf CustomColliderLineRecoursive} * {\bf Parent} \\$

родительская линия

Зб

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 (  \begin{aligned} & \text{Pin * pp1,} \\ & \text{Pin * pp2,} \\ & \text{AddWhireCommand * comm} \end{aligned} \right)
```

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 (  \begin{aligned} & \text{Pin}*\text{p1}, \\ & \text{Pin}*\text{p2} \ ) \end{aligned} \text{ [static]}
```

static для создания команды AddWhireCommand => провод

Аргументы

| p1 | Первый Контакт |
|----|-----------------|
| p2 | Второрй Контакт |

```
4.14.3.2 color()
```

 $\label{eq:QColor} \mbox{QColor NewWhire::color ()} \quad \mbox{[override], [virtual]}$

Цвет провода

4.15 Класс Pin 37

Возвращает

Переопределяет метод предка CustomColliderLineRecoursive.

Объявления и описания членов классов находятся в файлах:

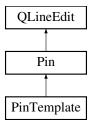
- NewWhire.h
- $\bullet \ \ {\rm NewWhire.cpp}$

4.15 Класс Pin

Контакт

#include <pin.h>

Граф наследования:Ріп:



Открытые слоты

- void Update (bool dotupd)
 - Обновление
- void Remove From
Chain ()

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 конткта

• qreal x ()

координата х контакта на сцене

• qreal y ()

координата у контакта на сцене

• void pinWhire (bool show=true)

Отображение провода контакта

• void PinWUpd (bool upd=true)

Пересчет расположения внутриней 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

4.15 Класс Pin 39

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()
```

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()
```

```
\ensuremath{\mathsf{Dot}} * \ensuremath{\mathsf{Pin}} :: \ensuremath{\mathsf{coredot}} ( )
```

Внутренняя точка провода конткта

Возвращает

```
4.15.3.3 \quad dot() [1/2]
Dot * Pin::dot (
              bool\ recalc = false\ )\quad [virtual]
Внешняя точка провода контакта
Аргументы
 recalc
          пересчитывать ли провод (коллизии)
Возвращает
4.15.3.4 \quad 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()
{\color{red}{\rm NewPinWhire}}*{\color{blue}{\rm Pin::getpinWhire}} ( )
Возвращает провод от данного контакта
```

Возвращает

4.15 Класс Pin 41

```
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\;)
```

Классы 42 Пересчет расположения внутриней точки провода

Аргументы

4.15.3.12 x()

qreal Pin::x ()

координата х контакта на сцене

Возвращает

Объявления и описания членов классов находятся в файлах:

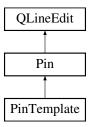
- pin.h
- pin.cpp

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 ${\tt dragMoveEvent}$ (QDragMoveEvent *event) override
 - отключение перемещение итд
- ∼PinTemplate ()
 - Деструктор

Открытые члены унаследованные от Pin

```
 - Pin (Port *port, bool bl=true, QLineEdit *parent=nullptr)
    Создает контакт
• \simPin ()
    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 х контакта на сцене
• qreal y ()
    координата у контакта на сцене
• void pinWhire (bool show=true)
    Отображение провода контакта
• void PinWUpd (bool upd=true)
    Пересчет расположения внутриней точки провода
```

Дополнительные унаследованные члены

Открытые слоты унаследованные от Ріп

```
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 show Context
Menu (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

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

4.16.3 Методы

Переопределяет метод предка Ріп.

```
4.16.3.2 dragMoveEvent()
```

Переопределяет метод предка Ріп.

```
4.16.3.3 dropEvent()
```

```
void PinTemplate::dropEvent ( {\bf QDropEvent * event } ) \quad [{\bf override}], \, [{\bf virtual}]
```

отключение перемещение итд

Переопределяет метод предка Pin.

Объявления и описания членов классов находятся в файлах:

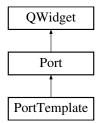
- PinTemplate.h
- PinTemplate.cpp

4.17 Класс Port

Разъем на сцене

#include <port.h>

Граф наследования:Port:



4.17 Класс Port 47

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 х на сцене

• qreal y ()

Координата у 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 \operatorname{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()
```

Добавление нового контакта

Аргументы

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

4.17 Класс Port 49

Возвращает Созданный контакт 4.17.3.2 ContextMenu() $\label{eq:QMenu*Port::ContextMenu} \mbox{QMenu*Port::ContextMenu} \ (\) \quad \mbox{[protected]}, \mbox{[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 $4.17.3.5 \quad pins()$ $\operatorname{QVector} < \operatorname{Pin} * > \operatorname{Port::pins}$ () 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()
{\it qreal~Port::x} ( )
Координата х на сцене
Возвращает
4.17.3.9 y()
qreal Port::y ( )
Координата у на сцене
Возвращает
Объявления и описания членов классов находятся в файлах:
```

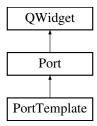
port.h main.cpp port.cpp

4.18 Kласс 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 х на сцене

• qreal y ()

Координата у 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 *> ports Vector
 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()

 ${\bf PortTemplate::} {\bf PortTemplate}~($

PortTwmplateObject * o)

Новый Шаблон hfp|tvf.

Аргументы

```
о Объект для Drag and Drop
```

4.18.3 Методы

4.18.3.1 addPinn()

```
Pin * PortTemplate::addPinn (  \label{eq:QString name} \ = \ \cdots,  int index = -1 )
```

Добавление контакта (шаблона контакта)

Аргументы

| name | |
|-------|--|
| index | |

Возвращает

Добавленный контакт

Объявления и описания членов классов находятся в файлах:

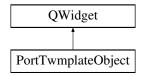
- PortTemplate.h
- $\bullet \ \operatorname{PortTemplate.cpp}$

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 drop Event (QDrop
Event *event) override

Перетягивание на сцену

- void dragMoveEvent (QDragMoveEvent *event) override Перетягивание на сцену
- void mousePressEvent (QMouseEvent *event) override Перетягивание на сцену
- PortTwmplateObject ()

конструктор

• \sim 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()

```
\label{lem:condition} \mbox{void PortTwmplateObject::dragEnterEvent (} $$ \mbox{QDragEnterEvent * event ) [override]} $$
```

Перетягивание на сцену

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

4.19.2.2 dragMoveEvent()

```
\label{local_potential} \begin{tabular}{ll} void PortTwmplateObject::dragMoveEvent ( \\ QDragMoveEvent * event ) & [override] \end{tabular}
```

Перетягивание на сцену

 $c_{\rm M.}$ https://doc.qt.io/qt-6/qgraphicsitem.html

```
4.19.2.3 dropEvent()
```

```
\label{eq:condition} \begin{tabular}{ll} void PortTwmplateObject::dropEvent ( & & \\ QDropEvent * event ) & [override] \\ \end{tabular}
```

Перетягивание на сцену

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

```
4.19.2.4 mousePressEvent()
```

```
\label{eq:condition} \begin{tabular}{ll} void PortTwmplateObject::mousePressEvent ( \\ QMouseEvent * event ) & [override] \end{tabular}
```

Перетягивание на сцену

```
cm. https://doc.qt.io/qt-6/qgraphicsitem.html
```

Объявления и описания членов классов находятся в файлах:

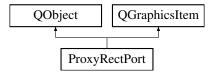
- PortTwmplateObject.h
- PortTwmplateObject.cpp

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

координаты центра

```
    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 х;
• qreal YY ()
    координата по у;
• void XX (greal x)
    установка координаты по х;
• void YY (qreal y)
    установка координаты по х;
• qreal left ()
    Левая сторона
• qreal right ()
    Правая сторона
• qreal bottom ()
    Низ
• qreal top ()
    Bepx
• QPointF center ()
```

4.20.1 Подробное описание

```
Прослойка для перемещения разъема по сцене
```

```
4.20.2 Конструктор(ы)
```

```
4.20.2.1 ProxyRectPort()
```

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()

```
\label{eq:const_proxyRectPort::boundingRect} \ \text{QRectF ProxyRectPort::boundingRect ( ) const} \quad [\text{override}]
```

```
cm. 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()

```
\label{eq:condition} \mbox{void ProxyRectPort::ColiderCheck (} \\ \mbox{bool upd} = \mbox{true )}
```

Проверка коллизий с проводами

| Аргументы |
|---|
| upd |
| |
| |
| 4.20.3.5 color() [1/2] |
| |
| QColor ProxyRectPort::color () |
| Цвет прокси (Шапка над разъемом) |
| |
| Возвращает |
| |
| |
| 4.00.9.6. 1. () to (a) |
| $4.20.3.6 \text{color}() \ [2/2]$ |
| void ProxyRectPort::color (|
| QColor c) |
| установка цвета прокси (Шапка над разъемом) |
| |
| Возвращает |
| |
| |
| 4.20.3.7 geometry() [1/2] |
| 1.20.8.1 geometry () [1/2] |
| QRectF ProxyRectPort::geometry () |
| костыль аля bounding rect |
| |
| Возвращает |
| |
| |
| 420.2.0 |
| 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
      х левого края
4.20.3.11 paint()
void ProxyRectPort::paint (
               \label{eq:QPainter} \mbox{QPainter} * \mbox{painter},
               const\ QStyleOptionGraphicsItem\ *\ option,
               {\bf QWidget * widget }) \quad [{\bf override}]
cm. https://doc.qt.io/qt-6/qgraphicsitem.html
4.20.3.12 ProxyColider()
{\bf void\ ProxyRectPort::ProxyColider\ (}
               int xx = 0,
               int yy = 0)
Проверка коллизий разъемами (рекурия)
Аргументы
```

| XX | направление смещения в случае рекурсивного наложения |
|----|--|
| уу | направление смещения в случае рекурсивного наложения |
| | |

```
4.20.3.13 right()
{\tt qreal\ ProxyRectPort::right\ (\ )\quad [inline]}
Правая сторона
```

```
Возвращает
      х правого края
4.20.3.14 setconnector()
void ProxyRectPort::setconnector (
              Port * port )
Установка вложенного разъема
Аргументы
 port
4.20.3.15 \text{ shape}()
QPainterPath ProxyRectPort::shape ( ) const [override]
{\bf cm.}\ \ https://doc.qt.io/qt-6/qgraphicsitem.html
4.20.3.16 \text{ top()}
qreal ProxyRectPort::top ( ) [inline]
Bepx
Возвращает
     у верхнего края
4.20.3.17 type()
int\ ProxyRectPort::type\ (\ )\ const\quad [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 \quad XX() \ [1/2]
```

qreal ProxyRectPort::XX () $\,$

координата по х;

Возвращает

```
4.20.3.20 XX()[2/2]
```

```
void ProxyRectPort::XX ( {\tt qreal}~{\tt x}~)
```

установка координаты по х;

Возвращает

4.20.3.21 YY() [1/2]

qreal ProxyRectPort::YY () $\,$

координата по у;

Возвращает

 $4.20.3.22 \quad YY() [2/2]$

void ProxyRectPort::YY ($qreal\ y\)$

установка координаты по х;

Возвращает

Объявления и описания членов классов находятся в файлах:

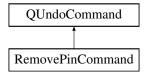
- \bullet proxyrectport.h
- $\bullet \ \ proxyrectport.cpp$

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()

```
Remove
PinCommand::Remove
PinCommand ( \label{eq:pin*} \mbox{Pin}*\mbox{ p })
```

создание команды в стэк

Аргументы

р удаляемый контакт

Объявления и описания членов классов находятся в файлах:

- RemovePinCommand.h
- RemovePinCommand.cpp

4.22 Kласс 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()

Объявления и описания членов классов находятся в файлах:

 $\bullet \ \ Remove Port Com and .h$

удаляемый разъем

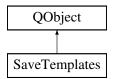
• RemovePortComand.cpp

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
```

4.24 Класс View 65

Аргументы

path

4.23.2.2 Load()

 $\label{eq:QList} \mbox{QList} < \mbox{QPair} < \mbox{QStringList, QString} > > \mbox{SaveTemplates::Load} \ (\) \quad [\mbox{static}]$

выгрузка листа шаблонов

Возвращает

4.23.2.3 Save()

сохранение шаблона

Аргументы

О

Объявления и описания членов классов находятся в файлах:

- SaveTemplates.h
- $\bullet \ \ Save Templates.cpp$

4.24 Класс View

"Дисплей" с вложеным отображением

#include <view.h>

Граф наследования:View:



66 Классы

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

4.24 Класс View 67

```
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 ( \label{eq:QColor} \mbox{QColor color })
```

установка цвета фона

Аргументы

color

4.24.2.3 ContextMenu()

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

Добовление Разъема контекстное меню

Возвращает

68 Классы

```
4.24.2.4 GScene() [1/2]
{\bf MYGraphicsScene} * {\bf View::GScene} \ ( \ )
Сцена вложенная в отображение
Возвращает
4.24.2.5 GScene() [2/2]
void View::GScene (
             {\bf 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:
```

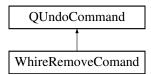
view.hview.cpp

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

• \sim WhireRemoveComand ()

Деструктор

• void undo () override

возвращение кабеля между р1 и р2 и запись в команду wc

• void redo () override

Удаление кабеля

Открытые атрибуты

• AddPinComand * p1

контакты р1 и р2

- 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()

Создание команды на удаление провода

70 Классы

Аргументы

w удаляемый провод

Объявления и описания членов классов находятся в файлах:

- $\bullet \ \ Whire Remove Com and .h$
- $\bullet \ \ Whire Remove Com and .cpp$

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:
            AddComand(View* v, int x, int y,QString name="");
00029 \\ 00033
            ~AddComand();
void undo() override;
void redo() override;
00037
00041
            Port* p;
int xx, yy;
View* v;
00045
00049
00053
            QString name;
00057
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:
             AddPinComand(Port*p,QString name="",bool rea=true); ~AddPinComand();
00023
00027
00031
             void undo() override;
            void redo() override;
QString name = "";
Pin* pn = nullptr;
AddComand* prt;
00035
00039
00043
00047
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~{\rm class}~{\bf AddWhireCommand:}~{\rm public}~{\rm QUndoCommand}
00014 {
00015 public:
00021
          AddWhireCommand(AddPinComand* p11, AddPinComand* p22);
00025
           ~AddWhireCommand();
00029
          void undo() override;
00033
          void redo() override;
AddPinComand* p1, *p2;
00037
          NewWhire* whire;
00041
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:
            Chain(); ~Chain();
00023
00027
            void AddPin(Pin* p);
void RemovePin(Pin* p);
static QVector<Chain*> chains;
00032
00037
00041
            \label{eq:qvector}  \text{QVector} < \text{Pin*} > \text{pins}; 
00045
            void moveToChain(Chain* chain);
00050
00054
            void Dots();
            QColor color;
00058
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 {
          \mathbf{Q}_{-}\mathbf{OBJECT}
00014
00015
00016~\mathrm{public}:
00020
          ChainTable();
00024
          ~ChainTable();
          View* view=nullptr;
void AdddPortSL();
00028
00032
00036
          void AddChain();
00037 public slots:
          void UpdateTable();
00041
00047
          void CellChange(int row, int column);
00053
          void CellClck(int row, int colum);
00054 protected:
00058
          {\bf QMetaObject::} Connection \ {\bf m\_connection};
00059 }:
00060 #endif
```

5.6 CustomColliderLineRecoursive.h

```
00001 #pragma once 00002 #ifndef CUSTOMCOLLIDERLINERECOURSIVE H 00003 #define CUSTOMCOLLIDERLINERECOURSIVE H 00004 00005 #include<br/>
<
```

5.7 Dot.h 73

```
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
                           Custom \\ Collider \\ Line \\ Recoursive \\ (bool\ Vertical\_f\_Horiz\_t, Dot^*\ d1, Dot^*\ d2,\ Custom \\ Collider \\ Line \\ Recoursive^*
                         parent=nullptr);
00038
00046
00054
00064
00069
00073
00077
00081
                          int JumpDerection = 1;
00082 public slots:
00088
                           void FixColliding(int times=0);
                          void ClearInside();
void setVertical();
00092
00096
00100
                          void setHorizontal();
                          void JumpFrom(QGraphicsItem* itm);
00105
00106 protected:
00110
                           Dot* d1, *d2;
00114
                           ColisionFixWay fixway;
00118
                            \label{eq:quantum quantum qu
00122
                          ProxyRectPort* lastcolide=nullptr;
                          int lastleft, lastbottom;
00123
00127
                          CustomColliderLineRecoursive* Parent;
00128
00129 };
00130
00131 \#endif
```

5.7 Dot.h

```
00001~\#\mathrm{pragma} once
00002 #ifndef DOT_H
00003 #define DOT_H
00004 \#include<Qt\overline{M}ath>
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~\mathrm{public};
00029
           Dot(QGraphicsObject* parent = nullptr);
00034
          int type() const override:
00038
           QColor cl;
00042
           bool triangle;
00046
          bool bg;
          int x();
int y();
void x(int x);
00050
00054
00058
00062
          void y(int y);
00067
           void setColor(QColor cl);
00072 \\ 00077
           void setTriangle(bool bl);
           void setBig(bool bl);
          Pin* pn();
CustomColliderLineRecoursive*Vdot, *Hdot;
00081
00085
00086
           void EmitIs inMove(bool moving)
00087
00088
              if (_pin != nullptr)
00089
00090
                  00091
00092
              emit Is_inMove(moving);
00093
```

```
00098
           void setPin(Pin* p);
00099
            void Emit_Moving()
00100
00101
                int \mathbf{x} = pos().x();
               int y = pos().y();
int mx, my;
mx = x % 25;
if (mx < 0)
00102
00103
00104
00105
               mx += 25;

my = y \% 25;

if (my < 0)

my += 25;

if (mx > 12)
00106
00107
00108
00109
00110
00111
                   \mathbf{x} = \mathbf{x} - \mathbf{m}\mathbf{x} + 25;
                else
00112
00113
               y = y - my + 25; else
00114
00115
00116
00117
                   \mathbf{y} = \mathbf{y} - \mathbf{m} \mathbf{y};
00118
                setPos(x, y);
00119
               emit moving(this);
00120
             Dot();
00124
00128
           QVector<NewWhire*>whires;
00132
            void WhPl();
00133
            void WhMin();
            void paint(QPainter* painter, const QStyleOptionGraphicsItem* option, QWidget* widget) override;
00139
00145 \\ 00151
            QRectF boundingRect() const override;//
           QPainterPath shape() const override;//
00152
           int whrscount()
00153
           {
00154
               return Linecounter;
           }
00155
00156
00157 \ \mathrm{signals};
           void Is_inMove(bool moving);
void moving(Dot* d);
00161
00165
00166 public slots:
00170
           void VerticalDot(Dot* d);
00174 void
00175 private:
           void HorizontalDot(Dot* d);
00179
           \quad \text{int Line counter} = 0; \\
           Pin* _pin;
00183
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:
          GItemFrame();
00017
00021
           ~GItemFrame();
00027
          QRectF boundingRect() const override;//
          QPainterPath shape() const override;//void paint(QPainter* painter, const QStyleOptionGraphicsItem* option, QWidget* widget) override;
00033
00039
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>
```

5.10 mainwindow.h

```
00009 \ \# include < QMimeData >
00016 class GView: public QGraphicsView
00017 {
00018
00019 public:
            GView(QObject *parent);
00024
00028
             ~GView();
00033
            MYGraphicsScene* GScene();
            void mouseMoveEvent(QMouseEvent* event) override;
00038
            void mouseReleaseEvent(QMouseEvent* event) override; void mousePressEvent(QMouseEvent* event) override; void dropEvent(QDropEvent* event)override;
00043
00048
00053
            void dragEnterEvent(QDragEnterEvent* event) override; void dragMoveEvent(QDragMoveEvent* event) override;
00058
00063
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:
          MainWindow(QWidget *parent = nullptr);
00018
           MainWindow();
00019
00020 public slots:
00021
          void AddTemp(Port* p=new Port());
00022 public slots:
          void hidemMap();
00023
00024 private:
00025
          Ui::MainWindow *ui;
00026
          View* v;
minimap* minima;
00027
00028
          QScrollArea* TempList;
00029 };
00030 #endif // MAINWINDOW H
```

5.11 minimap.h

```
00001~\#\mathrm{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 {
         Q_OBJECT
00015
00016
00017 public:
         minimap(View *v);
void mouseMoveEvent(QMouseEvent* event)override;
00022
00027
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
          \simminimap();
00057
          View*
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:
          View* Mview;
int _scale = 100;
00019
00023
          explīcit MYGraphicsScene(QObject *parent = nullptr);
00028
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 "Custom Collider Line Recoursive.h"
00008 class Pin;
00015 class NewPinWhire: public CustomColliderLineRecoursive
00016 {
           Q_OBJECT
00017
00018 public:
          NewPinWhire(Pin* p, QThread* th);
~NewPinWhire();
00030
00034
          bool hasConection();
QColor color() override;
00039
00044
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~{\rm class}~{\rm NewWhire}~:{\rm public}~{\rm CustomColliderLineRecoursive}
00016 {
00017
            /Q_OBJECT
00018 public:
           Pin *p1 = nullptr, *p2 = nullptr;
Chain* chain = nullptr;
NewWhire(Pin*pp1,Pin*pp2,AddWhireCommand* comm);
00022
00026
00033
            ~NewWhire();
00037
           AddWhireCommand* command;
static void AddComandW(Pin* p1,Pin* p2);
00041
00047
           QColor color() override;
00052
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>
```

5.16 PinTemplate.h 77

```
00008 \#include<QLineEdit>
00009 #Include Objects
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 {
          Q OBJECT
00022
00023
00024 public:
00031
          {\it explicit} \ {\it Pin(Port* port,bool bl=true,QLineEdit *parent = nullptr);}
00035 \\ 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();
          Port* parCon;
virtual Dot* dot(bool recalc = false);
virtual void dot(Dot* d);
00075
00081
00086
00087 private:
00092
          void mousePressEvent(QMouseEvent* event) override;
00096
          virtual void dragEnterEvent(QDragEnterEvent* event) override;
00100
          virtual void dropEvent(QDropEvent* event) override;
00104
          {\tt virtual\ void\ dragMoveEvent}({\tt QDragMoveEvent*}\ event)\ override;
00105 public:
00110 Dot* coredot();
          qreal x();
qreal y();
00115
00119
00124
          void pinWhire(bool show=true);
          void PinWUpd(bool upd = true);
Chain* chain;
00129
00133 \\ 00137
          AddPinComand* command=nullptr;
00138 protected:
          virtual QMenu* ContextMenu();
Ui::Pin *ui;
00143
00144
00148
          bool upd = false;
00149 public slots:
          void RemoveFromChain();
00153
          void Remove();
00157
00158 protected slots:
00162
          virtual void showContextMenu(const QPoint& pos);
00166
          void ChangeColor();
00167 private:
00168
          bool updaterbl;
Dot* d;
00172
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 {
           Q_OBJECT
00015
00016
00017 public:
00022
            PinTemplate(Port* p);
00027
            virtual void dragEnterEvent(QDragEnterEvent* event) override;
           virtual void dropEvent(QDropEvent* event) override;
virtual void dragMoveEvent(QDragMoveEvent* event) override;
00031
00035
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 \ {\rm class} \ {\rm 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();
           explicit Port(AddComand* com, QWidget *parent = nullptr);
00042
           Port();
virtual "Port();
QString name();
void name(QString str);
00046
00050
00055
00060
00064
           static QVector<Port*> portsVector;
           QVector<Pin*>pins();
AddComand* adcom;
00069
00073
           void Remove();
virtual Pin* addPin(QString name = "", int index = -1,bool bl=true);
00077
00085
00086
00087 private slots:
00091
           void showContextMenu(const QPoint& pos);
           void on_pushButton_clicked();
Pin* addPinSl(QString name="");
void RemoveSL();
00092
00098 \\ 00102
00103 public slots:
           void Update(bool updF=false);
           ProxyRectPort* proxy();
void proxy(ProxyRectPort*prox);
00112
00117
00118
00119
00120\ \mathrm{protected}\colon
           Ui::Port *ui;
00121
00126
           virtual QMenu* ContextMenu();
           ProxyRectPort* _proxy;
QSpacerItem* spacer;
00130
00134
00135 };
00136
00137 \# endif // PORT H
```

5.18 PortTemplate.h

```
00001~\#\mathrm{pragma} once
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:
            PortTemplate(PortTwmplateObject* o);
Pin* addPinn(QString name = "", int index = -1);
00028
00035
             PortTemplate();
00039
00040 private:
            PortTwmplateObject* object;
00044
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<qwidget.h>
00006 #include"qpushbutton.h"
00007 #include"qlineedit.h"
00008 class PortTemplate;
00013 class PortTwmplateObject: public QWidget
00014 {
            Q_OBJECT
00015
00016
00017 public:
00021
            PortTemplate* templ;
00025
            QPushButton* editpb;
00029
             QPushButton* delpb;
00033
            {\it QLineEdit*} \ {\it name};
00040
            {\bf void} \,\, {\bf dragEnterEvent}({\bf QDragEnterEvent*} \,\, {\bf event}) \,\, {\bf override};
            void dropEvent(QDropEvent* event) override;
void dragMoveEvent(QDragMoveEvent* event) override;
00047
00054
00061
            void mousePressEvent(QMouseEvent* event) override;
00062
00063 public slots:
00067 void editPush();
00071 void delPush();
00072 public:
            PortTwmplateObject();
00076
08000
             \simPortTwmplateObject();
00081
00082 \ \};
00083 #endif
```

5.20 proxyrectport.h

```
00001 \#pragma once
00002
00003 #ifndef PROXYRECTPORT
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 \\ 00029
           void EmitMove();
           QPainterPath shape()const override;
           QRectF boundingRect() const override;
00035
00041
           void paint (QPainter *painter, const QStyleOptionGraphicsItem *option, QWidget *widget) override;
00046
           QRectF geometry();
00051
           void geometry(QRectF rec);
00056
           void setconnector(Port* port);
           Port *getport();
QColor color();
00061
00066
           void color(QColor c);
00071
00075
           ~ProxyRectPort();
00081
           void Update(bool upd=false, int i=0);
           int type() const override;
void ColiderCheck(bool upd=true);
00086 \\ 00091
           void ProxyColider(int xx=0,int yy=0); qreal XX();
00097
00102
           qreal YY();
void XX(qreal x);
00107
00112
00117
           void YY(qreal y);
           qreal left() {return mapToScene(boundingRect().left(), 0).x(); }
qreal right() { return mapToScene(boundingRect().right(), 0).x(); }
qreal bottom() { return mapToScene(0, boundingRect().bottom()).y(); }
qreal top() { return mapToScene(0, boundingRect().top()).y(); }
00122
00127
00132
00137
           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 \ {\rm class} \ {\bf RemovePinCommand} \ : {\bf public} \ {\bf QUndoCommand}
00012~\{
00013
00014 public:
00019
          RemovePinCommand(Pin*p);
00023
           ~RemovePinCommand();
00027
          void undo() override;
00031
          void redo() override;
          int pos = 0;
QString name = "";
00035
00039
          AddPinComand* pinc;
00043
00047
          Port* prt;
          Pin* pin;
00051
00052
00053 };
00054 #endif
```

5.22 RemovePortComand.h

```
00001 #pragma once
00001 #pragma once
00002 #ifndef REMOVECOMAND_H
00003 #define REMOVECOMAND_H
00004 \#include<QUndoCommand>
00005 class Port;
00006 class AddComand;
00007 class View;
80000
00012 class RemovePortComand : public QUndoCommand
00013 {
00014 public:
00018
         RemovePortComand(Port* p);
00022
          ~RemovePortComand();
00026
         void undo() override;
00030
         void redo() override;
QString name;
00034
00038
          AddComand* addcom;
00042
          View* v;
00046
         int xx, yy;
00047 };
00048 #endif
```

5.23 SaveTemplates.h

```
\begin{array}{c} 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: \\ \end{array}
```

5.24 view.h 81

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

5.24 view.h

```
00001 #pragma once 00002 #ifndef VIEW
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 {
          Q_OBJECT
00017
00018 public:
00022
          View();
          MYGraphicsScene* GScene();
void GScene(MYGraphicsScene* scene);
00027
00032
00037
          QColor backGroundColor();
          void backGroundColor(QColor color);
GView* view();
QUndoStack* stack;
00042
00047
00051
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 \text{ signals:}
          void SceneChanged();
00078
00079 public slots:
          void stckUndo();
00083
00087
          void stckRedo();
00091
          void AdddPortSL();
00095
          void saveImage();
          void AdddPortSL(int x, int y, QString name);
00099
00100
00101 public:
00102
          AddComand* AdddPort();
AddComand* AdddPort(int x, int y, QString name);
00106
00110
00111
          void stackPush(QUndoCommand* com)
00115
00116
00117
             stack->push(com);
00119 #if QT_CONFIG(wheelevent)
              void wheelEvent(QWheelEvent* e);
00124
00125~\#\mathrm{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:
         WhireRemoveComand(NewWhire* w); ~WhireRemoveComand();
00018
00022
00026
         void undo() override;
00030
         void redo() override;
00034
         AddPinComand* p1, * p2;
```

82

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

Предметный указатель

| AddComand, 7 | View, 67 |
|----------------------------------|---|
| AddComand, 8 | $\operatorname{coredot}$ |
| AddComandW | Pin, 39 |
| NewWhire, 36 | CustomColliderLineRecoursive, 15 |
| AddPin | boundingRect, 17 |
| Chain, 12 | color, 17 |
| addPin | CustomColliderLineRecoursive, 16 |
| Port, 48 | FixColliding, 17 |
| AddPinComand, 8 | JumpFrom, 17 |
| AddPinComand, 9 | paint, 18 |
| addPinn | setFixWay, 18 |
| PortTemplate, 53 | shape, 18 |
| AddWhireCommand, 10 | |
| AddWhireCommand, 10 | Del |
| | SaveTemplates, 64 |
| backGroundColor | Dot, 19 |
| View, 67 | boundingRect, 21 |
| bottom | Dot, 20 |
| ProxyRectPort, 57 | paint, 21 |
| boundingRect | setBig, 21 |
| CustomColliderLineRecoursive, 17 | setColor, 21 |
| Dot, 21 | setPin, 21 |
| GItemFrame, 23 | setTriangle, 22 |
| ProxyRectPort, 57 | shape, 22 |
| | type, 22 |
| CellChange | x, 22 |
| ChainTable, 14 | y, 22 |
| CellClck | dot |
| ChainTable, 14 | Pin, 39, 40 |
| center | dragEnterEvent |
| ProxyRectPort, 57 | GView, 25 |
| Chain, 11 | PinTemplate, 46 |
| AddPin, 12 | PortTwmplateObject, 54 |
| Chain, 12 | ${ m drag}{ m Move}{ m Event}$ |
| moveToChain, 12 | GView, 25 |
| RemovePin, 12 | PinTemplate, 46 |
| ChainTable, 13 | PortTwmplateObject, 54 |
| CellChange, 14 | dropEvent |
| CellClck, 14 | GView, 25 |
| ColiderCheck | PinTemplate, 46 |
| ProxyRectPort, 57 | PortTwmplateObject, 54 |
| color | |
| CustomColliderLineRecoursive, 17 | $\operatorname{Emit}\operatorname{Upd}$ |
| NewPinWhire, 33 | Pin, 40 |
| NewWhire, 36 | Fr. G. 111.11 |
| ProxyRectPort, 58 | FixColliding |
| ContextMenu | CustomColliderLineRecoursive, 17 |
| Pin, 39 | goometry |
| Port, 49 | geometry ProxyRectPort. 58 |
| | |

| getpinWhire | MYGraphicsScene, 30 |
|----------------------------------|----------------------------------|
| Pin, 40 | MYGraphicsScene, 30 |
| getport | |
| ProxyRectPort, 59 | name |
| GItemFrame, 23 | Pin, 41 |
| bounding Rect, 23 | Port, 49 |
| paint, 23 | NewPinWhire, 31 |
| shape, 23 | color, 33 |
| GScene | hasConection, 33 |
| GView, 26 | NewPinWhire, 33 |
| View, 67, 68 | NewWhire, 34 |
| GView, 24 | AddComandW, 36 |
| dragEnterEvent, 25 | color, 36 |
| dragMoveEvent, 25 | NewWhire, 36 |
| dropEvent, 25 | |
| GScene, 26 | paint |
| GView, 25 | CustomColliderLineRecoursive, 18 |
| lastselected, 27 | Dot, 21 |
| mouseMoveEvent, 26 | GItemFrame, 23 |
| mousePressEvent, 26 | ProxyRectPort, 59 |
| mouseReleaseEvent, 26 | Pin, 37 |
| mousertereusez vent, 20 | ContextMenu, 39 |
| hasConection | coredot, 39 |
| NewPinWhire, 33 | dot, 39, 40 |
| 110111 11111111110, 00 | $\operatorname{EmitUpd}$, 40 |
| index | getpinWhire, 40 |
| Pin, 40 | index, 40 |
| | name, 41 |
| JumpFrom | Pin, 39 |
| CustomColliderLineRecoursive, 17 | pinWhire, 41 |
| , | PinWUpd, 41 |
| lastselected | x, 43 |
| GView, 27 | pins |
| left | Port, 49 |
| ProxyRectPort, 59 | PinTemplate, 43 |
| Load | dragEnterEvent, 46 |
| SaveTemplates, 65 | dragMoveEvent, 46 |
| | dropEvent, 46 |
| MainWindow, 27 | PinTemplate, 45 |
| minimap, 28 | pinWhire |
| minimap, 28 | Pin, 41 |
| mouseDoubleClickEvent, 29 | PinWUpd |
| mouseMoveEvent, 29 | Pin, 41 |
| mousePressEvent, 29 | Port, 46 |
| mouseReleaseEvent, 29 | addPin, 48 |
| mouseDoubleClickEvent | ContextMenu, 49 |
| minimap, 29 | |
| mouseMoveEvent | name, 49 |
| GView, 26 | pins, 49 |
| minimap, 29 | Port, 48 |
| mousePressEvent | proxy, 49, 50 |
| GView, 26 | x, 50 |
| minimap, 29 | y, 50 |
| PortTwmplateObject, 55 | PortTemplate, 51 |
| mouseReleaseEvent | addPinn, 53 |
| GView, 26 | PortTemplate, 52 |
| minimap, 29 | PortTwmplateObject, 53 |
| moveToChain | dragEnterEvent, 54 |
| Chain, 12 | dragMoveEvent, 54 |
| , | dropEvent, 54 |

| mousePressEvent, 55 | ProxyRectPort, 60 |
|---|---------------------------|
| proxy | |
| Port, 49, 50 | top |
| ProxyColider | ProxyRectPort, 60 |
| ProxyRectPort, 59 | type |
| ProxyRectPort, 55 | Dot, 22 |
| bottom, 57 | ProxyRectPort, 60 |
| bounding Rect, 57 | II., J. 4. |
| center, 57 | Update |
| ColiderCheck, 57 | ProxyRectPort, 60 |
| color, 58 | View, 65 |
| geometry, 58 | backGroundColor, 67 |
| getport, 59 | ContextMenu, 67 |
| left, 59 | GScene, 67, 68 |
| paint, 59 | scale, 68 |
| ProxyColider, 59 | view, 68 |
| ProxyRectPort, 57 | view, 08 |
| right, 59 | |
| setconnector, 60 | View, 68 |
| shape, 60 | WhireRemoveComand, 69 |
| top, 60 | WhireRemoveComand, 69 |
| type, 60 | vvimentemove containa, vo |
| Update, 60 | X |
| XX, 61 | Dot, 22 |
| YY, 61 | Pin, 43 |
| | Port, 50 |
| RemovePin | XX |
| Chain, 12 | ProxyRectPort, 61 |
| RemovePinCommand, 62 | , |
| RemovePinCommand, 62 | у |
| RemovePortComand, 63 | Dot, 22 |
| RemovePortComand, 64 | Port, 50 |
| right | YY |
| ProxyRectPort, 59 | ProxyRectPort, 61 |
| Corre | |
| Save Save Topopleton 65 | |
| SaveTemplates, 65 | |
| SaveTemplates, 64 | |
| Del, 64 | |
| Load, 65 | |
| Save, 65 | |
| scale | |
| View, 68 setBig | |
| | |
| Dot, 21 setColor | |
| | |
| Dot, 21 | |
| setconnector ProxyRectPort, 60 | |
| setFixWay | |
| CustomColliderLineRecoursive, 18 | |
| | |
| setPin Dot 21 | |
| Dot, 21 | |
| setTriangle | |
| Dot, 22 | |
| shape Custom Collidar Lina Paganggira 18 | |
| CustomColliderLineRecoursive, 18 | |
| Dot, 22 | |
| GItemFrame, 23 | |