

dWell

Создано системой Doxygen 1.8.20

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 adminDialog	7
4.1.1 Подробное описание	9
4.1.2 Конструктор(ы)	9
4.1.2.1 adminDialog()	9
4.1.3 Методы	9
4.1.3.1 on_tableWidget_doubleClicked	9
4.1.4 Данные класса	9
4.1.4.1 ui	9
4.2 Шаблон класса book< entry >	10
4.2.1 Подробное описание	11
4.2.2 Методы	11
4.2.2.1 loadFromFile()	11
4.2.2.2 operator[]()	12
4.2.2.3 read()	12
4.2.2.4 saveToFile()	13
4.2.2.5 write()	13
4.3 Класс bookEntry	13
4.3.1 Подробное описание	15
4.3.2 Документация по друзьям класса и функциям, относящимся к классу	15
4.3.2.1 operator<<	15
4.3.2.2 operator>>	15
4.4 Класс commandantDialog	15
4.4.1 Подробное описание	17
4.4.2 Конструктор(ы)	17
4.4.2.1 commandantDialog()	17
4.5 Класс doc	18
4.5.1 Подробное описание	18
4.5.2 Методы	18
4.5.2.1 generate()	18
4.6 Класс dormitory	19
4.6.1 Подробное описание	19
4.6.2 Конструктор(ы)	19
4.6.2.1 dormitory()	19
4.6.3 Методы	20

4.6.3.1	getDormCfg()	20
4.7	Класс habitant	20
4.7.1	Подробное описание	21
4.7.2	Конструктор(ы)	22
4.7.2.1	habitant()	22
4.8	Структура habitant::habitantData	22
4.8.1	Подробное описание	22
4.9	Класс habitantEditDialog	23
4.9.1	Подробное описание	24
4.9.2	Конструктор(ы)	24
4.9.2.1	habitantEditDialog()	24
4.10	Класс habitantsbook	24
4.11	Класс initSetupDialog	25
4.11.1	Подробное описание	26
4.11.2	Конструктор(ы)	26
4.11.2.1	initSetupDialog()	26
4.12	Класс MainWindow	26
4.12.1	Подробное описание	27
4.12.2	Конструктор(ы)	27
4.12.2.1	MainWindow()	27
4.13	Класс rbook	28
4.13.1	Подробное описание	29
4.13.2	Методы	29
4.13.2.1	availableForCheckin()	29
4.13.2.2	checkin()	29
4.13.2.3	checkout()	30
4.13.2.4	getHabitantBySid()	30
4.13.2.5	getRbook()	30
4.14	Класс relocationDialog	31
4.14.1	Подробное описание	32
4.14.2	Конструктор(ы)	32
4.14.2.1	relocationDialog()	32
4.15	Класс room	32
4.15.1	Подробное описание	34
4.15.2	Методы	34
4.15.2.1	availableForCheckin()	34
4.15.2.2	checkin()	34
4.15.2.3	checkout()	35
4.15.2.4	operator[]()	35
4.15.2.5	read()	35
4.15.2.6	write()	36
4.16	Класс studentDialog	36
4.16.1	Подробное описание	37

4.16.2 Конструктор(ы)	37
4.16.2.1 studentDialog()	37
4.17 Класс tools	38
4.17.1 Подробное описание	38
4.18 Класс ubook	39
4.18.1 Подробное описание	40
4.18.2 Методы	40
4.18.2.1 findUser()	40
4.18.2.2 findUserByName()	41
4.18.2.3 getUbook()	41
4.19 Класс user	41
4.19.1 Подробное описание	43
4.19.2 Конструктор(ы)	43
4.19.2.1 user()	43
4.20 Класс userEditDialog	43
4.20.1 Подробное описание	45
4.20.2 Конструктор(ы)	45
4.20.2.1 userEditDialog()	45
4.21 Класс writeReadableItem	45
4.21.1 Подробное описание	46
5 Файлы	47
5.1 Файл adminDialog.cpp	47
5.1.1 Подробное описание	48
5.2 Файл adminDialog.h	48
5.2.1 Подробное описание	49
5.3 Файл book.cpp	49
5.3.1 Подробное описание	50
5.4 Файл book.h	50
5.4.1 Подробное описание	51
5.5 Файл bookentry.h	51
5.5.1 Подробное описание	52
5.5.2 Функции	52
5.5.2.1 operator<<()	52
5.5.2.2 operator>>()	53
5.6 Файл commandantDialog.cpp	53
5.6.1 Подробное описание	54
5.7 Файл commandantDialog.h	54
5.7.1 Подробное описание	55
5.8 Файл config.h	55
5.8.1 Подробное описание	56
5.8.2 Переменные	56
5.8.2.1 applicationName	56

5.9	Файл doc.cpp	57
5.9.1	Подробное описание	57
5.10	Файл doc.h	57
5.10.1	Подробное описание	58
5.11	Файл dormitory.cpp	59
5.11.1	Подробное описание	59
5.12	Файл dormitory.h	59
5.12.1	Подробное описание	60
5.13	Файл habitant.cpp	61
5.13.1	Подробное описание	61
5.14	Файл habitant.h	61
5.14.1	Подробное описание	62
5.15	Файл habitanteditdialog.cpp	63
5.15.1	Подробное описание	63
5.16	Файл habitanteditdialog.h	64
5.16.1	Подробное описание	64
5.17	Файл initsetupdialog.cpp	65
5.17.1	Подробное описание	65
5.18	Файл initsetupdialog.h	65
5.18.1	Подробное описание	66
5.19	Файл main.cpp	67
5.19.1	Подробное описание	67
5.20	Файл mainwindow.cpp	68
5.20.1	Подробное описание	68
5.21	Файл mainwindow.h	68
5.21.1	Подробное описание	69
5.22	Файл rbook.cpp	70
5.22.1	Подробное описание	70
5.23	Файл rbook.h	70
5.23.1	Подробное описание	71
5.24	Файл relocationdialog.cpp	72
5.24.1	Подробное описание	72
5.25	Файл relocationdialog.h	73
5.25.1	Подробное описание	73
5.26	Файл room.cpp	74
5.26.1	Подробное описание	74
5.27	Файл room.h	74
5.27.1	Подробное описание	75
5.28	Файл studentdialog.cpp	76
5.28.1	Подробное описание	76
5.29	Файл studentdialog.h	77
5.29.1	Подробное описание	77
5.30	Файл tools.cpp	78

5.30.1 Подробное описание	78
5.31 Файл tools.h	78
5.31.1 Подробное описание	80
5.32 Файл ubook.cpp	80
5.32.1 Подробное описание	80
5.33 Файл ubook.h	81
5.33.1 Подробное описание	82
5.34 Файл user.cpp	82
5.34.1 Подробное описание	83
5.35 Файл user.h	83
5.35.1 Подробное описание	84
5.36 Файл usereditdialog.cpp	84
5.36.1 Подробное описание	85
5.37 Файл writereadableitem.h	85
5.37.1 Подробное описание	86
Предметный указатель	87

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

doc	18
dormitory	19
habitant	20
habitant::habitantData	22
habitantsbook	24
QDialog	
adminDialog	7
commandantDialog	15
habitantEditDialog	23
initSetupDialog	25
relocationDialog	31
studentDialog	36
userEditDialog	43
QMainWindow	
MainWindow	26
tools	38
writeReadableItem	45
book< entry >	10
book< room >	10
rbook	28
book< user >	10
ubook	39
bookEntry	13
room	32
user	41

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

adminDialog	Класс диалога панели администрирования. Для функционирования необходима коллекция пользователей. *m_ubook указывает на эту коллекцию	7
book< entry >	Шаблонный класс-контейнер с функциями сохранения в файл. Используется для хранения пользователей и комнат в системе. Наследуется от класса writeReadableItem , переопределяя его методы	10
bookEntry	Класс шаблонного элемента контейнера системы. Наследуется от класса writeReadableItem , определяя операторы >> и << над методами writeReadableItem::write и writeReadableItem::read	13
commandantDialog	Класс диалога панели коменданта. Для функционирования необходима коллекция комнат	15
doc	Класс справки	18
dormitory	Класс, предоставляющий сведения об общежитии	19
habitant	Класс проживающего	20
habitant::habitantData	Структура, хранящая информацию о проживающем. Используется для перемещения этой информации внутри программы	22
habitantEditDialog	Класс диалога редактирования проживающего (его данных)	23
habitantsbook	24
initSetupDialog	Класс диалога начальной настройки системы	25
MainWindow	Класс диалога входа в систему	26
rbook	Класс контейнер комнат. Наследуется от класса book . Добавляет необходимый функционал для доступа к контейнеру комнат	28
relocationDialog	Класс диалога переселения проживающего	31

room	Класс комнаты. Наследуется от класса bookEntry . Содержит контейнер проживающих в комнате	32
studentDialog	Класс диалога студента	36
tools	Класс вспомогательных инструментов	38
ubook	Класс контейнер пользователей. Наследуется от класса <code>book</code> . Добавляет необходимый функционал для доступа к контейнеру <code>book</code>	39
user	Класс комнаты. Наследуется от класса bookEntry	41
userEditDialog	Класс диалога редактирования пользователя	43
writeReadableItem	Абстрактный класс объекта, способного записываться и считываться из потока . .	45

Глава 3

Список файлов

3.1 Файлы

Полный список документированных файлов.

admindialog.cpp	Файл реализации класса adminDialog	47
admindialog.h	Заголовчный файл класса adminDialog	48
book.cpp	Файл реализации класса book	49
book.h	Заголовчный файл класса book	50
bookentry.h	Заголовчный файл класса bookEntry	51
commandantdialog.cpp	Файл реализации класса commandantDialog	53
commandantdialog.h	Заголовчный файл класса commandantDialog	54
config.h	Файл конфигурации	55
doc.cpp	Файл реализации класса doc	57
doc.h	Заголовчный файл класса doc	57
dormitory.cpp	Файл реализации класса dormitory	59
dormitory.h	Заголовчный файл класса dormitory	59
habitant.cpp	Файл реализации класса habitant	61
habitant.h	Заголовчный файл класса habitant	61
habitanteditdialog.cpp	Файл реализации класса habitantEditDialog	63
habitanteditdialog.h	Заголовчный файл класса habitantEditDialog	64
habitantsbook.h		??
initsetupdialog.cpp	Файл реализации класса initSetupDialog	65

initsetupdialog.h	Заголовчный файл класса initSetupDialog	65
main.cpp	Файл, содержащий точку входа в программу	67
mainwindow.cpp	Файл реализации класса MainWindow	68
mainwindow.h	Заголовчный файл класса MainWindow	68
rbook.cpp	Файл реализации класса rbook	70
rbook.h	Заголовчный файл класса rbook	70
relocationdialog.cpp	Файл реализации класса relocationdialog	72
relocationdialog.h	Заголовчный файл класса relocationDialog	73
room.cpp	Файл реализации класса room	74
room.h	Заголовчный файл класса room	74
studentdialog.cpp	Файл реализации класса studentDialog	76
studentdialog.h	Заголовчный файл класса studentDialog	77
tools.cpp	Файл реализации класса tools	78
tools.h	Заголовчный файл класса tools	78
ubook.cpp	Файл реализации класса ubook	80
ubook.h	Заголовчный файл класса ubook	81
user.cpp	Файл реализации класса user	82
user.h	Заголовчный файл класса user	83
usereditdialog.cpp	Файл реализации класса userEditDialog	84
usereditdialog.h	??
writereadableitem.h	Заголовчный файл класса writeReadableItem	85

Глава 4

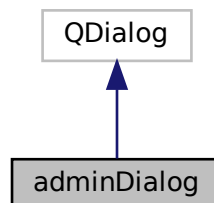
Классы

4.1 Класс adminDialog

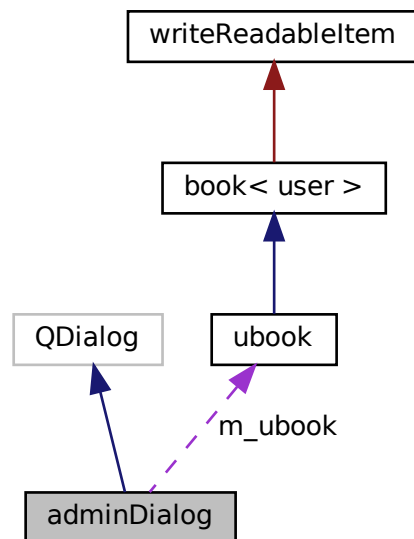
Класс диалога панели администрирования. Для функционирования необходима коллекция пользователей. *m_ubook указывает на эту коллекцию.

```
#include <admindialog.h>
```

Граф наследования:adminDialog:



Граф связей класса adminDialog:



Открытые члены

- [adminDialog](#) (QWidget *parent)
Для использования слотов и/или сигналов необходим макрос Q_OBJECT внутри класса.
- [~adminDialog](#) ()
Деструктор класса [adminDialog](#).

Закрытые слоты

- void [on_addButton_clicked](#) ()
Слот, обрабатывающий нажатие на кнопку "Добавить"(пользователя)
- void [on_removeButton_clicked](#) ()
Слот, обрабатывающий нажатие на кнопку "Удалить"(пользователя)
- void [updateTable](#) ()
Слот, выполняющий обновление (генерацию) таблицы пользователей
- void [on_tableWidget_doubleClicked](#) (const QModelIndex &index)
Обработчик двойного нажатия на строку пользователя в таблице
- void [on_logoutButton_clicked](#) ()

Закрытые данные

- [ubook * m_ubook](#)
указатель на контейнер пользователей
- [Ui::adminDialog * ui](#)
Указатель на сгенерированный интерфейс.

4.1.1 Подробное описание

Класс диалога панели администрирования. Для функционирования необходима коллекция пользователей. *m_ubook указывает на эту коллекцию.

4.1.2 Конструктор(ы)

4.1.2.1 adminDialog()

```
adminDialog::adminDialog (  
    QWidget * parent ) [explicit]
```

Для использования слотов и/или сигналов необходим макрос Q_OBJECT внутри класса.

Конструктор с необязательным указанием родительского объекта.

Аргументы

parent	Указатель на родительский объект.
--------	-----------------------------------

4.1.3 Методы

4.1.3.1 on_tableWidget_doubleClicked

```
void adminDialog::on_tableWidget_doubleClicked (  
    const QModelIndex & index ) [private], [slot]
```

Обработчик двойного нажатия на строку пользователя в таблице

Аргументы

index	Индекс пользователя в таблице.
-------	--------------------------------

4.1.4 Данные класса

4.1.4.1 ui

```
Ui::adminDialog* adminDialog::ui [private]
```

Указатель на сгенерированный интерфейс.

Указатель на объект UI-класса, сгенерированного на основе UI-файла [adminDialog.ui](#).

Через этот указатель можно обратиться к элементам главного окна, созданного в Qt Designer.

Объявления и описания членов классов находятся в файлах:

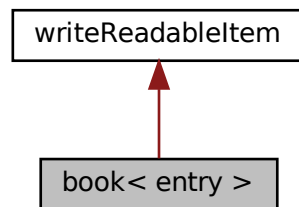
- [admindialog.h](#)
- [admindialog.cpp](#)

4.2 Шаблон класса `book< entry >`

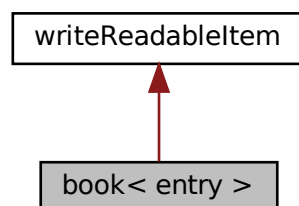
Шаблонный класс-контейнер с функциями сохранения в файл. Используется для хранения пользователей и комнат в системе. Наследуется от класса [writeReadableItem](#), переопределяя его методы.

```
#include <book.h>
```

Граф наследования: `book< entry >`:



Граф связей класса `book< entry >`:



Открытые члены

- `book ()`
Конструктор класса по умолчанию.
- `void loadFromFile (const QString &filename)`
Метод, позволяющий загрузить контейнер из файла.
- `void saveToFile (const QString &filename) const`
Константный метод, позволяющий сохранить контейнер в файл.
- `const entry & operator[] (uint idx) const`
Оператор [].
- `uint size () const`
Определяет размер контейнера.

Защищенные данные

- `QVector< entry > mEntries`
Контейнер, хранящий шаблонные элементы

Закрытые члены

- `void write (QDataStream &ost) const override`
Определение базового метода записи в поток
- `void read (QDataStream &ist) override`
Определение базового метода считывания из потока

4.2.1 Подробное описание

```
template<typename entry>
class book< entry >
```

Шаблонный класс-контейнер с функциями сохранения в файл. Используется для хранения пользователей и комнат в системе. Наследуется от класса `writeReadableItem`, переопределяя его методы.

Параметры шаблона

entry	объект, хранящийся в контейнере.
-------	----------------------------------

4.2.2 Методы

4.2.2.1 loadFromFile()

```
template<typename entry >
void book< entry >::loadFromFile (
    const QString & filename )
```

Метод, позволяющий загрузить контейнер из файла.

Аргументы

filename	Имя файла.
----------	------------

4.2.2.2 operator[]()

```
template<typename entry >
const entry & book< entry >::operator[] (
    uint idx ) const
```

Оператор [].

Аргументы

idx	Индекс читаемого элемента.
-----	----------------------------

Возвращает

Константная ссылка на читаемый элемент.

Возвращает ссылку на элемент с индексом `idx`. Слово `const` после списка параметров означает, что это константная версия метода, она не может изменять данные класса. Результат также имеет квалификатор `const`, т. е. по этой ссылке элемент нельзя изменить.

Можно было бы определить метод, позволяющий изменить заметку. Его объявление выглядело бы так:

```
Note &operator[] (uint idx);
```

В данном случае делать этого нельзя, поскольку возвращённая ссылка на элемент неподконтрольна данному классу и он не имеет возможности узнать, было ли изменение и когда это произошло, а значит не может уведомить присоединённые виды о том, что данные изменились.

4.2.2.3 read()

```
template<typename entry >
void book< entry >::read (
    QDataStream & ist ) [override], [private], [virtual]
```

Определение базового метода считывания из потока

Переопределение наследуемого метода `writeReadableItem::read`.

Аргументы

ist	Ссылка на поток.
-----	------------------

Замещает `writeReadableItem`.

4.2.2.4 saveToFile()

```
template<typename entry >
void book< entry >::saveToFile (
    const QString & filename ) const
```

Константный метод, позволяющий сохранить контейнер в файл.

Аргументы

filename	Имя файла.
----------	------------

4.2.2.5 write()

```
template<typename entry >
void book< entry >::write (
    QDataStream & ost ) const    [override], [private], [virtual]
```

Определение базового метода записи в поток

Переопределение наследуемого метода [writeReadableItem::write](#).

Аргументы

ost	Ссылка на поток.
-----	------------------

Замещает [writeReadableItem](#).

Объявления и описания членов классов находятся в файлах:

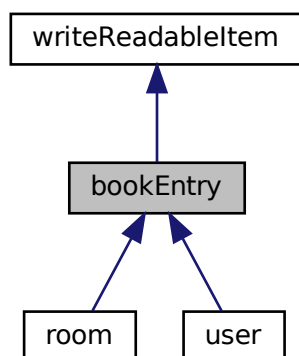
- [book.h](#)
- [book.cpp](#)

4.3 Класс bookEntry

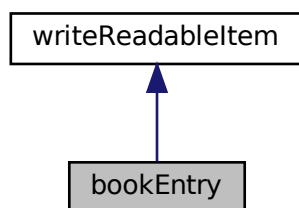
Класс шаблонного элемента контейнера системы. Наследуется от класса [writeReadableItem](#), определяя операторы >> и << над методами [writeReadableItem::write](#) и [writeReadableItem::read](#).

```
#include <bookentry.h>
```

Граф наследования:bookEntry:



Граф связей класса bookEntry:



Открытые члены

- `bookEntry ()`
Конструктор класса по умолчанию

Друзья

- `QDataStream & operator<< (QDataStream &ost, const bookEntry &e)`
Дружественное определение оператора <<.
- `QDataStream & operator>> (QDataStream &ist, bookEntry &e)`
Дружественное определение оператора >>.

Дополнительные унаследованные члены

4.3.1 Подробное описание

Класс шаблонного элемента контейнера системы. Наследуется от класса [writeReadableItem](#), определяя операторы `>>` и `<<` над методами [writeReadableItem::write](#) и [writeReadableItem::read](#).

4.3.2 Документация по друзьям класса и функциям, относящимся к классу

4.3.2.1 `operator<<`

```
QDataStream& operator<< (  
    QDataStream & ost,  
    const bookEntry & e ) [friend]
```

Дружественное определение оператора `<<`.

Аргументы

ost	поток, в который происходит запись.
e	элемент, записываемый в поток.

4.3.2.2 `operator>>`

```
QDataStream& operator>> (  
    QDataStream & ist,  
    bookEntry & e ) [friend]
```

Дружественное определение оператора `>>`.

Аргументы

ost	поток, из которого происходит чтение.
e	элемент, считываемый из потока.

Объявления и описания членов класса находятся в файле:

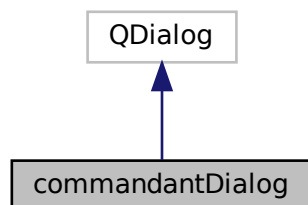
- [bookentry.h](#)

4.4 Класс `commandantDialog`

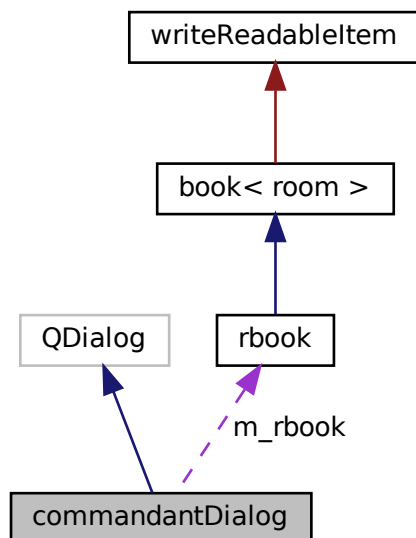
Класс диалога панели коменданта. Для функционирования необходима коллекция комнат.

```
#include <commandantdialog.h>
```

Граф наследования:commandantDialog:



Граф связей класса commandantDialog:



Открытые члены

- `commandantDialog` (`QWidget *parent=nullptr`)
Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.
- `~commandantDialog` ()
Деструктор

Закрытые слоты

- void `updateTable` ()
Обновляет таблицу проживающих в интерфейсе.
- void `updateSumLabel` ()
Обновляет статистику о заполненности общежития.
- void `on_checkinButton_clicked` ()
Обработчик кнопки "Заселить".
- void `on_pushButton_clicked` ()
Обработчик кнопки "Выйти".
- void `on_checkoutButton_clicked` ()
Обработчик кнопки "Выселить".
- void `on_relocButton_clicked` ()
Обработчик кнопки "Переселить".
- void `on_giveDocButton_clicked` ()
Обработчик кнопки "Выдать справку".

Закрытые данные

- `rbook * m_rbook`
Указатель на контейнер комнат.
- `Ui::commandantDialog * ui`
Указатель на сгенерированный интерфейс.

4.4.1 Подробное описание

Класс диалога панели коменданта. Для функционирования необходима коллекция комнат.

4.4.2 Конструктор(ы)

4.4.2.1 `commandantDialog()`

```
commandantDialog::commandantDialog (
    QWidget * parent = nullptr ) [explicit]
```

Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.

Конструктор с необязательным указанием родительского объекта.

Аргументы

parent	Указатель на родительский объект.
--------	-----------------------------------

Объявления и описания членов классов находятся в файлах:

- [commandantdialog.h](#)
- [commandantdialog.cpp](#)

4.5 Класс doc

Класс справки.

```
#include <doc.h>
```

Открытые члены

- [doc](#) ()
Конструктор по умолчанию

Открытые статические члены

- static void [generate](#) (const [habitant](#) *h, QPrinter *printer)
Статический метод, выполняющий сохранение PDF-файла справки.

4.5.1 Подробное описание

Класс справки.

4.5.2 Методы

4.5.2.1 generate()

```
void doc::generate (  
    const habitant * h,  
    QPrinter * printer ) [static]
```

Статический метод, выполняющий сохранение PDF-файла справки.

Аргументы

h	Константный указатель на проживающего, для которого генерируется справка.
printer	Содержит настройки печати PDF-файла.

Объявления и описания членов классов находятся в файлах:

- [doc.h](#)
- [doc.cpp](#)

4.6 Класс dormitory

Класс, предоставляющий сведения об общежитии.

```
#include <dormitory.h>
```

Открытые члены

- `dormitory ()`
Конструктор по умолчанию.
- `dormitory (QString &name, uint &roomCapacity, uint &dormCapacity)`
Дополнительный конструктор.
- `const QString name ()`
Возвращает наименование комнаты
- `uint roomCapacity ()`
Возвращает емкость одной комнаты
- `uint capacity ()`
Возвращает количество комнат в общежитии.

Открытые статические члены

- `static dormitory * getDormCfg ()`

Закрытые члены

- `void load (QDataStream &ost)`
- `void save (QDataStream &ist)`
- `void loadFromFile (const QString &filename)`
- `void saveToFile (const QString &filename)`

Закрытые данные

- `uint mRoomCapacity`
Количество мест в комнате.
- `uint mDormCapacity`
Количество комнат в общежитии.
- `QString mDormName`
Наименование общежития.

4.6.1 Подробное описание

Класс, предоставляющий сведения об общежитии.

4.6.2 Конструктор(ы)

4.6.2.1 dormitory()

```
dormitory::dormitory (  
    QString & name,  
    uint & roomCapacity,  
    uint & dormCapacity )
```

Дополнительный конструктор.

Аргументы

name	Наименование общежития.
roomCapacity	Количество мест в одной комнате.
dormCapacity	Количество комнат в общежитии.

4.6.3 Методы

4.6.3.1 getDormCfg()

```
dormitory * dormitory::getDormCfg ( ) [static]
```

Статический метод, возвращающий указатель на созданный объект класса dormitory с загруженными данными

Объявления и описания членов классов находятся в файлах:

- [dormitory.h](#)
- [dormitory.cpp](#)

4.7 Класс habitant

Класс проживающего.

```
#include <habitant.h>
```

Классы

- struct [habitantData](#)

Структура, хранящая информацию о проживающем. Используется для перемещения этой информации внутри программы.

Открытые члены

- `habitant ()`
Конструктор класса.
- `habitant (habitantData *hd)`
Дополнительный конструктор.
- `bool setData (habitantData *hd)`
Устанавливает поля данных, равным полям в структуре .
- `habitantData * getData () const`
Возвращает указатель на данные проживающего.
- `const QString fname () const`
Возвращает имя проживающего.
- `const QString lname () const`
Возвращает фамилию проживающего.
- `const QString patronymic () const`
Возвращает отчество проживающего.
- `const QDate birthDate () const`
Возвращает дату рождения проживающего.
- `uint studentID () const`
Возвращает номер студенческого билета проживающего.
- `uint numOfCourse () const`
Возвращает номер курса обучения проживающего.
- `QString group () const`
Возвращает наименование группы, в которой обучается проживающий.
- `uint roomNumber () const`
Возвращает номер комнаты проживающего.

Закрытые данные

- `QString mFname`
Поле имени.
- `QString mLname`
Поле фамилии.
- `QString mPatronymic`
Поле отчества.
- `QDate mBirthDate`
Поле даты рождения.
- `uint mStudentID`
Поле номера студенческого билета.
- `uint mNumOfCourse`
Поле номера курса обучения.
- `QString mGroup`
Поле номера группы обучения.
- `uint mRoomNumber`
Поле номера комнаты проживания.

4.7.1 Подробное описание

Класс проживающего.

4.7.2 Конструктор(ы)

4.7.2.1 `habitant()`

```
habitant::habitant (
    habitantData * hd )
```

Дополнительный конструктор.

Аргументы

hd	Указатель на структуру данных.
----	--------------------------------

Объявления и описания членов классов находятся в файлах:

- [habitant.h](#)
- [habitant.cpp](#)

4.8 Структура `habitant::habitantData`

Структура, хранящая информацию о проживающем. Используется для перемещения этой информации внутри программы.

```
#include <habitant.h>
```

Открытые атрибуты

- `QString fname`
- `QString lname`
- `QString patronymic`
- `QDate birthDate`
- `uint studentID`
- `uint numOfCourse`
- `QString group`
- `uint roomNumber`

4.8.1 Подробное описание

Структура, хранящая информацию о проживающем. Используется для перемещения этой информации внутри программы.

Объявления и описания членов структуры находятся в файле:

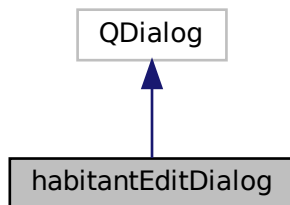
- [habitant.h](#)

4.9 Класс habitantEditDialog

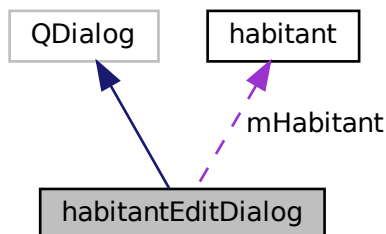
Класс диалога редактирования проживающего (его данных).

```
#include <habitanteditdialog.h>
```

Граф наследования: habitantEditDialog:



Граф связей класса habitantEditDialog:



Открытые члены

- `habitantEditDialog` (`QWidget *parent`, `QStringList availRooms`)
Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.
- `~habitantEditDialog` ()
Деструктор.
- `void setHabitant` (`habitant *h`)
Устанавливает редактируемого проживающего.

Закрытые слоты

- `void accept` ()
Обрабатывает подтверждение диалога.

Закрытые данные

- [habitant](#) * [mHabitant](#)
Указатель на редактируемого проживающего.
- [Ui::habitantEditDialog](#) * [ui](#)
Указатель на сгенерированный интерфейс.

4.9.1 Подробное описание

Класс диалога редактирования проживающего (его данных).

4.9.2 Конструктор(ы)

4.9.2.1 `habitantEditDialog()`

```
habitantEditDialog::habitantEditDialog (
    QWidget * parent,
    QStringList availRooms )
```

Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.

Конструктор с необязательным указанием родительского объекта.

Аргументы

<code>parent</code>	Указатель на родительский объект.
<code>availRooms</code>	Доступный для заселения комнаты.

Объявления и описания членов классов находятся в файлах:

- [habitanteditdialog.h](#)
- [habitanteditdialog.cpp](#)

4.10 Класс `habitantsbook`

Объявления и описания членов класса находятся в файле:

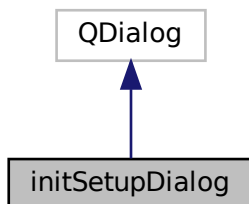
- [habitantsbook.h](#)

4.11 Класс `initSetupDialog`

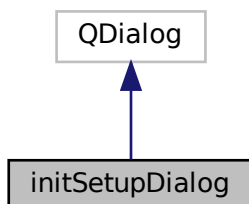
Класс диалога начальной настройки системы.

```
#include <initsetupdialog.h>
```

Граф наследования:`initSetupDialog`:



Граф связей класса `initSetupDialog`:



Открытые слоты

- `void accept ()`
Обрабатывает подтверждение диалога.

Открытые члены

- `initSetupDialog (QWidget *parent=nullptr)`
Конструктор с необязательным указанием родительского объекта.
- `~initSetupDialog ()`
Деструктор.

Закрытые данные

- `Ui::initSetupDialog * ui`
Указатель на сгенерированный интерфейс.

4.11.1 Подробное описание

Класс диалога начальной настройки системы.

4.11.2 Конструктор(ы)

4.11.2.1 `initSetupDialog()`

```
initSetupDialog::initSetupDialog (
    QWidget * parent = nullptr ) [explicit]
```

Конструктор с необязательным указанием родительского объекта.

Аргументы

parent	Указатель на родительский объект.
--------	-----------------------------------

Объявления и описания членов классов находятся в файлах:

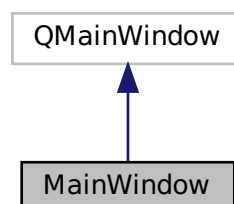
- [initsetupdialog.h](#)
- [initsetupdialog.cpp](#)

4.12 Класс MainWindow

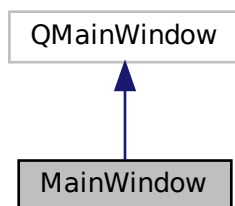
Класс диалога входа в систему.

```
#include <mainwindow.h>
```

Граф наследования:MainWindow:



Граф связей класса MainWindow:



Открытые члены

- `MainWindow (QWidget *parent=nullptr)`
Конструктор с необязательным указанием родительского объекта.
- `~MainWindow ()`
Деструктор.

Закрытые слоты

- `void about_qt ()`
Обрабатывает нажатие на кнопку "О Qt" в меню.
- `void about_dwell ()`
Обрабатывает нажатие на кнопку "О программе" в меню.
- `void about_autors ()`
Обрабатывает нажатие на кнопку "Об авторах" в меню.
- `void on_loginButton_clicked ()`
Обрабатывает нажатие на кнопку "Войти".

Закрытые данные

- `Ui::MainWindow * ui`
Указатель на сгенерированный интерфейс.

4.12.1 Подробное описание

Класс диалога входа в систему.

4.12.2 Конструктор(ы)

4.12.2.1 MainWindow()

`MainWindow::MainWindow (`
`QWidget * parent = nullptr)`

Конструктор с необязательным указанием родительского объекта.

Аргументы

parent	Указатель на родительский объект.
--------	-----------------------------------

Объявления и описания членов классов находятся в файлах:

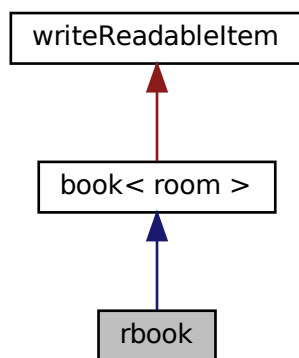
- [mainwindow.h](#)
- [mainwindow.cpp](#)

4.13 Класс rbook

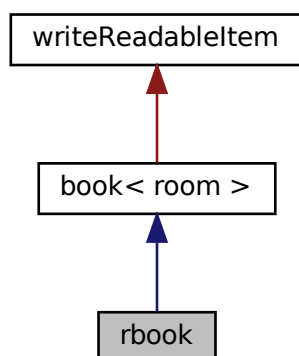
Класс контейнер комнат. Наследуется от класса book. Добавляет необходимый функционал для доступа к контейнеру комнат.

```
#include <rbook.h>
```

Граф наследования:rbook:



Граф связей класса rbook:



Открытые члены

- `rbook ()`
Конструктор по умолчанию
- `void checkin (uint rn, habitant *h)`
Производит заселение проживающего в комнату.
- `void checkout (uint rn, uint sid)`
Производит выселение проживающего из комнаты.
- `uint fullness () const`
Возвращает сумму заполненных мест во всех комнатах.
- `void setCapacity (uint &cap)`
Устанавливает количество комнат в контейнере равным cap.
- `const habitant * getHabitantBySid (uint &sid) const`
- `QStringList availRooms () const`
Возвращает список доступных для заселения комнат.
- `QStringList availRooms (uint &excludeRoomNumber) const`
Возвращает список доступных для заселения комнат, исключая .
- `bool availableForCheckin () const`
- `void touchFile (uint &dormCap, uint &roomCap)`
Генерирует файл данных контейнера.

Открытые статические члены

- `static rbook * getRbook ()`

Дополнительные унаследованные члены

4.13.1 Подробное описание

Класс контейнер комнат. Наследуется от класса `book`. Добавляет необходимый функционал для доступа к контейнеру комнат.

4.13.2 Методы

4.13.2.1 `availableForCheckin()`

```
bool rbook::availableForCheckin ( ) const
```

Возвращает логическое значение, которое отражает доступно ли заселение в общежитие.

4.13.2.2 `checkin()`

```
void rbook::checkin (
    uint rn,
    habitant * h )
```

Производит заселение проживающего в комнату.

Аргументы

rn	Номер комнаты.
h	Указатель на данные проживающего.

4.13.2.3 checkout()

```
void rbook::checkout (
    uint rn,
    uint sid )
```

Производит выселение проживающего из комнаты.

Аргументы

rn	Номер комнаты.
sid	Номер студенческого билета выселяемого.

4.13.2.4 getHabitantBySid()

```
const habitant * rbook::getHabitantBySid (
    uint & sid ) const
```

Возвращает указатель на объект проживающего с номером студенческого билета, равным sid

4.13.2.5 getRbook()

```
rbook * rbook::getRbook ( ) [static]
```

Статический метод, возвращающий указатель на созданный объект класса rbook с загруженными контейнером.

Объявления и описания членов классов находятся в файлах:

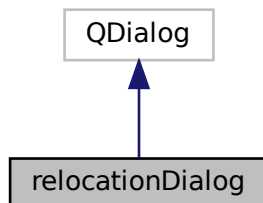
- [rbook.h](#)
- [rbook.cpp](#)

4.14 Класс relocationDialog

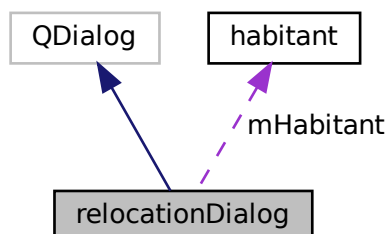
Класс диалога переселения проживающего.

```
#include <relocationdialog.h>
```

Граф наследования:relocationDialog:



Граф связей класса relocationDialog:



Открытые члены

- `relocationDialog` (`QWidget *parent`, `QStringList availRooms`)
Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.
- `~relocationDialog` ()
Деструктор.
- `void setHabitant` (`habitant *h`)
Устанавливает переселяемого проживающего.

Закрытые слоты

- `void accept` ()
Обрабатывает подтверждение диалога.

Закрытые данные

- [habitant](#) * [mHabitant](#)
Указатель на переселяемого проживающего.
- [Ui::relocationDialog](#) * [ui](#)
Указатель на сгенерированный интерфейс.

4.14.1 Подробное описание

Класс диалога переселения проживающего.

4.14.2 Конструктор(ы)

4.14.2.1 relocationDialog()

```
relocationDialog::relocationDialog (
    QWidget * parent,
    QStringList availRooms )
```

Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.

Конструктор с необязательным указанием родительского объекта.

Аргументы

<code>parent</code>	Указатель на родительский объект.
<code>availRooms</code>	Список доступных для переселения комнат

Объявления и описания членов классов находятся в файлах:

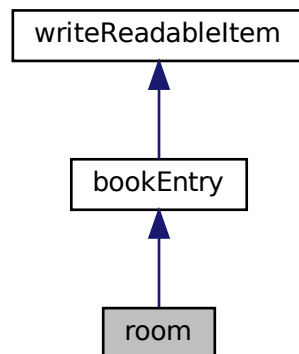
- [relocationdialog.h](#)
- [relocationdialog.cpp](#)

4.15 Класс room

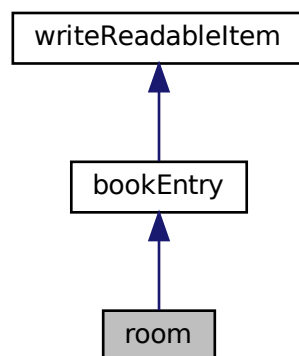
Класс комнаты. Наследуется от класса [bookEntry](#). Содержит контейнер проживающих в комнате.

```
#include <room.h>
```


Граф наследования:room:



Граф связей класса room:



Открытые члены

- `room ()`
Конструктор по умолчанию
- `void checkin (habitant h)`
Производит заселение проживающего в комнату.
- `bool checkout (uint sid)`
Производит выселение проживающего из комнаты.
- `uint number () const`
Возвращает номер комнаты проживающего.
- `uint capacity () const`

- Возвращает количество мест в комнате.
- `uint size () const`
- Возвращает количество занятых мест.
- `uint freeSlots () const`
- Возвращает количество свободных мест.
- `const habitant & operator[] (uint idx) const`
- Оператор `[]`.
- `void setCapacity (uint &n)`
- Устанавливает количество мест в комнате равным `n`.
- `void setNumber (uint &n)`
- Устанавливает номер комнаты равным `n`.
- `void clear ()`
- Освобождает все места в комнате.
- `bool isEmpty () const`
- Возвращает логическое значение, которое отражает пуста ли комната.
- `bool availableForCheckin () const`
- `QVector< habitant >::iterator findBySid (uint sid)`
- Возвращает итератор на проживающего с номером студ. билета `sid`.

Защищенные члены

- `void write (QDataStream &ost) const override`
- Определение базового метода записи комнаты в поток
- `void read (QDataStream &ist) override`
- Определение базового метода считывания комнаты из потока

Закрытые данные

- `uint mNumber`
- Поле номера комнаты.
- `QVector< habitant > mHabitants`
- Контейнер проживающих.

4.15.1 Подробное описание

Класс комнаты. Наследуется от класса `bookEntry`. Содержит контейнер проживающих в комнате.

4.15.2 Методы

4.15.2.1 `availableForCheckin()`

```
bool room::availableForCheckin ( ) const
```

Возвращает логическое значение, которое отражает доступна ли комната для заселения.

4.15.2.2 `checkin()`

```
void room::checkin (
    habitant h )
```

Производит заселение проживающего в комнату.

Аргументы

h	Указатель на данные проживающего.
---	-----------------------------------

4.15.2.3 checkout()

```
bool room::checkout (
    uint sid )
```

Производит выселение проживающего из комнаты.

Аргументы

sid	Номер студенческого билета выселяемого.
-----	---

4.15.2.4 operator[]()

```
const habitant & room::operator[] (
    uint idx ) const
```

Оператор [].

Аргументы

idx	Индекс читаемого проживающего.
-----	--------------------------------

Возвращает

Константная ссылка на читаемого проживающего.

4.15.2.5 read()

```
void room::read (
    QDataStream & ist ) [override], [protected], [virtual]
```

Определение базового метода считывания комнаты из потока

Аргументы

ist	Ссылка на поток.
-----	------------------

Замещает [writeReadableItem](#).

4.15.2.6 write()

```
void room::write (
    QDataStream & ost ) const    [override], [protected], [virtual]
```

Определение базового метода записи комнаты в поток

Аргументы

ost	Ссылка на поток.
-----	------------------

Замещает [writeReadableItem](#).

Объявления и описания членов классов находятся в файлах:

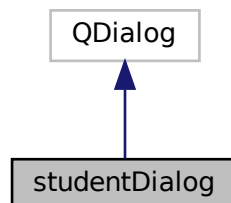
- [room.h](#)
- [room.cpp](#)

4.16 Класс studentDialog

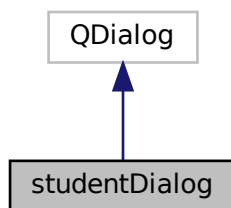
Класс диалога студента.

```
#include <studentdialog.h>
```

Граф наследования:studentDialog:



Граф связей класса studentDialog:



Открытые члены

- `studentDialog` (`QWidget *parent=nullptr`)
Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.
- `~studentDialog` ()
Деструктор.

Закрытые данные

- `Ui::studentDialog * ui`
Указатель на сгенерированный интерфейс.

4.16.1 Подробное описание

Класс диалога студента.

4.16.2 Конструктор(ы)

4.16.2.1 studentDialog()

```
studentDialog::studentDialog (
    QWidget * parent = nullptr ) [explicit]
```

Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.

Конструктор с необязательным указанием родительского объекта.

Аргументы

parent	Указатель на родительский объект.
--------	-----------------------------------

Объявления и описания членов классов находятся в файлах:

- [studentdialog.h](#)
- [studentdialog.cpp](#)

4.17 Класс tools

Класс вспомогательных инструментов.

```
#include <tools.h>
```

Открытые члены

- [tools](#) ()
Конструктор.

Открытые статические члены

- static QString [userTypeToStr](#) (const [user::utype](#) &)
Преобразовывает тип пользователя в строку с названием типа.
- static void [initSystem](#) (QWidget *)
Производит начальную инициализацию системы
- static QDialog * [getUserInterface](#) (QWidget *, const [user::utype](#) &type)
Возвращает указатель на интерфейс пользователя с типом type.

4.17.1 Подробное описание

Класс вспомогательных инструментов.

Объявления и описания членов классов находятся в файлах:

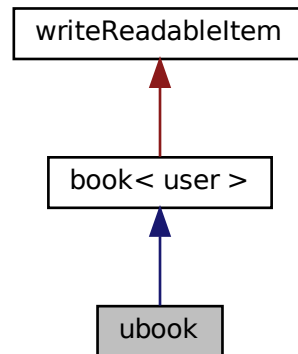
- [tools.h](#)
- [tools.cpp](#)

4.18 Класс ubook

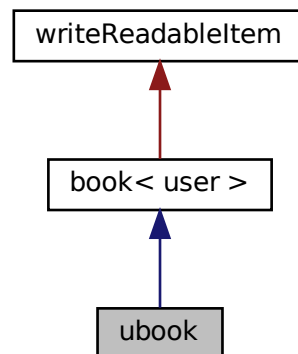
Класс контейнер пользователей. Наследуется от класса book. Добавляет необходимый функционал для доступа к контейнеру book.

```
#include <ubook.h>
```

Граф наследования:ubook:



Граф связей класса ubook:



Открытые члены

- `ubook ()`
Конструктор по умолчанию
- `user::utype findUser (const QString &name, const QString &passwd) const`

- Ищет пользователя с указанным именем и паролем.
- `user::utype findUserByName (const QString &name) const`
Ищет пользователя с указанным именем.
- `bool insert (user &user)`
Создаёт в контейнере запись о пользователе `user`.
- `bool remove (const uint &idx)`
Извлекает из контейнера запись о пользователе, который соответствует позиции `idx`.
- `void touchFile ()`
Генерирует файл данных контейнера.

Открытые статические члены

- `static ubook * getUbook ()`

Дополнительные унаследованные члены

4.18.1 Подробное описание

Класс контейнер пользователей. Наследуется от класса `book`. Добавляет необходимый функционал для доступа к контейнеру `book`.

4.18.2 Методы

4.18.2.1 findUser()

```
user::utype ubook::findUser (
    const QString & name,
    const QString & passwd ) const
```

Ищет пользователя с указанным именем и паролем.

Аргументы

name	Имя пользователя.
passwd	Пароль пользователя.

Возвращает

Тип пользователя.

4.18.2.2 findUserByName()

```
user::utype ubook::findUserByName (
    const QString & name ) const
```

Ищет пользователя с указанным именем.

Аргументы

name	Имя пользователя.
------	-------------------

Возвращает

Тип пользователя.

4.18.2.3 getUbook()

```
ubook * ubook::getUbook ( ) [static]
```

Статический метод, возвращающий указатель на созданный объект класса ubook с загруженными контейнером.

Объявления и описания членов классов находятся в файлах:

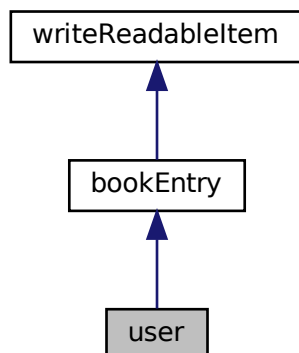
- [ubook.h](#)
- [ubook.cpp](#)

4.19 Класс user

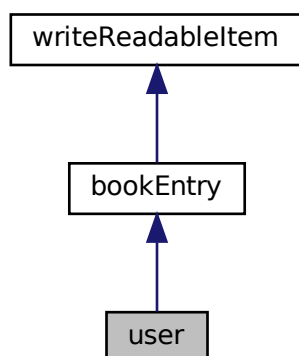
Класс комнаты. Наследуется от класса [bookEntry](#).

```
#include <user.h>
```

Граф наследования:user:



Граф связей класса user:



Открытые типы

- enum `utype` { ADMIN, COMMANDANT, STUDENT, UNKNOWN }
Перечислитель, содержащий возможные типы пользователей.

Открытые члены

- `user ()`
Конструктор по умолчанию.
- `user (QString name, QString pass, utype type)`
Дополнительный конструктор. Устанавливает соответствующие поля объекта класса user.
- `const QString name () const`
Возвращает имя пользователя.
- `const QString passwd () const`
Возвращает пароль пользователя.
- `utype type () const`
Возвращает тип пользователя.
- `bool setData (QString &name, QString &pass, utype &type)`
Устанавливает поля объекта пользователя.

Защищенные члены

- `void write (QDataStream &ost) const override`
Определение виртуального метода для записи в поток.
- `void read (QDataStream &ist) override`
Определение виртуального метода для считывания из потока.

Закрытые данные

- [QString mName](#)
Поле имени пользователя.
- [QString mPasswd](#)
Поле пароля пользователя.
- [utype mType](#)
Поле типа пользователя.

4.19.1 Подробное описание

Класс комнаты. Наследуется от класса [bookEntry](#).

4.19.2 Конструктор(ы)

4.19.2.1 user()

```
user::user (  
    QString name,  
    QString pass,  
    utype type )
```

Дополнительный конструктор. Устанавливает соответствующие поля объекта класса user.

Аргументы

name	Имя пользователя
pass	Пароль пользователя.
type	Тип пользователя.

Объявления и описания членов классов находятся в файлах:

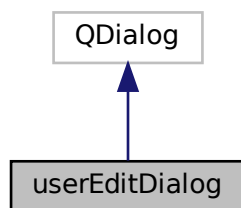
- [user.h](#)
- [user.cpp](#)

4.20 Класс userEditDialog

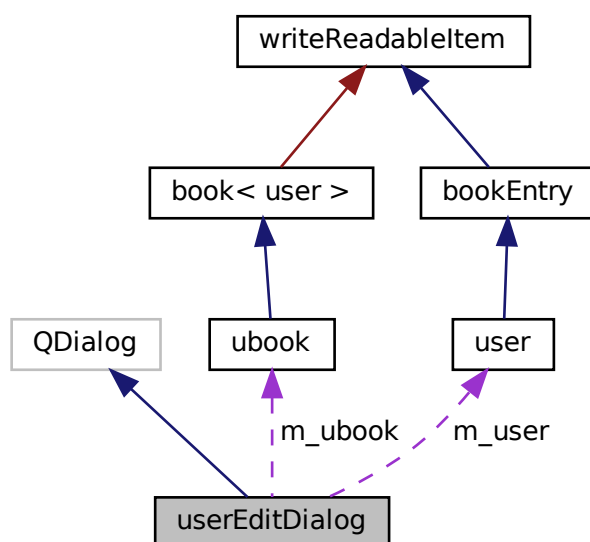
Класс диалога редактирования пользователя.

```
#include <usereditdialog.h>
```

Граф наследования: userEditDialog:



Граф связей класса userEditDialog:



Открытые члены

- `userEditDialog` (`QWidget *parent`)
Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.
- `~userEditDialog` ()
Деструктор.
- `void setUser` (`user *u`)
Устанавливает нового пользователя для редактирования.
- `void setUserForEdit` (`user *u`)
Устанавливает существующего пользователя для редактирования.

Закрытые слоты

- void `accept` ()
Обрабатывает подтверждение диалога.
- void `on_usernameEdit_textChanged` (const QString &username)
- void `on_passwdEdit_textChanged` (const QString &passwd)

Закрытые данные

- bool `editMode`
Режим редактирования: новый пользователь или существующий.
- `ubook * m_ubook`
Указатель на контейнер пользователей.
- `user * m_user`
Указатель на редактируемого пользователя.
- `Ui::userEditDialog * ui`
Указатель на сгенерированный интерфейс.

4.20.1 Подробное описание

Класс диалога редактирования пользователя.

4.20.2 Конструктор(ы)

4.20.2.1 userEditDialog()

```
userEditDialog::userEditDialog (
    QWidget * parent ) [explicit]
```

Для использования слотов и/или сигналов необходим макрос `Q_OBJECT` внутри класса.

Конструктор с необязательным указанием родительского объекта.

Аргументы

parent	Указатель на родительский объект.
--------	-----------------------------------

Объявления и описания членов классов находятся в файлах:

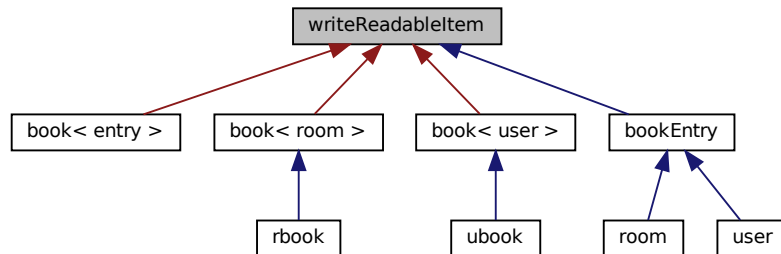
- `usereditdialog.h`
- `usereditdialog.cpp`

4.21 Класс writeReadableItem

Абстрактный класс объекта, способного записываться и считываться из потока.

```
#include <writereadableitem.h>
```

Граф наследования:writeReadableItem:



Открытые члены

- [writeReadableItem \(\)](#)
Конструктор по умолчанию.

Защищенные члены

- virtual void [write](#) (QDataStream &ost) const =0
Определение виртуального метода для записи в поток.
- virtual void [read](#) (QDataStream &ist)=0
Определение виртуального метода для считывания из потока.

4.21.1 Подробное описание

Абстрактный класс объекта, способного записываться и считываться из потока.

Объявления и описания членов класса находятся в файле:

- [writereadableitem.h](#)

Глава 5

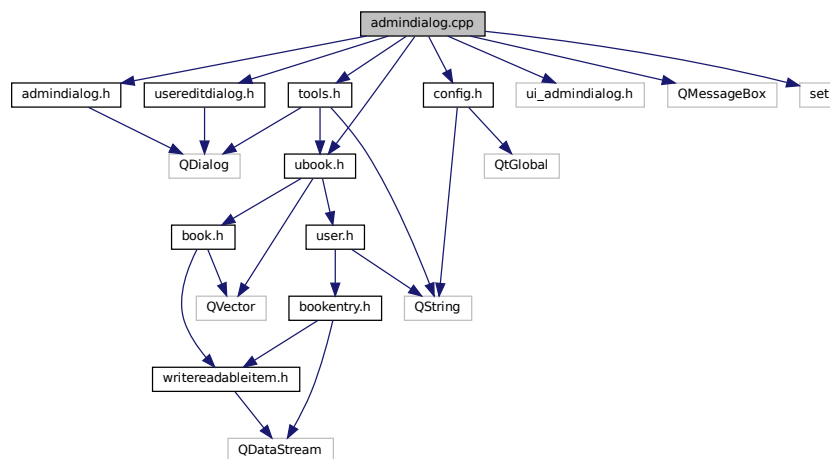
Файлы

5.1 Файл admindialog.cpp

Файл реализации класса `adminDialog`.

```
#include "admindialog.h"  
#include "ui_admindialog.h"  
#include "ubook.h"  
#include "usereditdialog.h"  
#include "tools.h"  
#include "config.h"  
#include <QMessageBox>  
#include <set>
```

Граф включаемых заголовочных файлов для `admindialog.cpp`:



Макросы

- `#define UNAME_COLUMN 0`
- `#define HTYPE_COLUMN 1`

5.1.1 Подробное описание

Файл реализации класса `adminDialog`.

Автор

Кашапов Ярослав

Дата

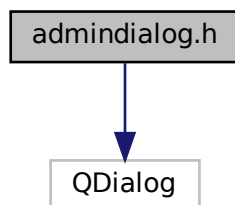
2021

5.2 Файл `admindialog.h`

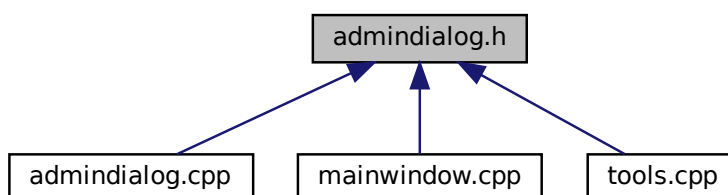
Заголовочный файл класса `adminDialog`.

```
#include <QDialog>
```

Граф включаемых заголовочных файлов для `admindialog.h`:



Граф файлов, в которые включается этот файл:



Классы

- class `adminDialog`

Класс диалога панели администрирования. Для функционирования необходима коллекция пользователей. `*m_ubook` указывает на эту коллекцию.

5.2.1 Подробное описание

Заголовчный файл класса `adminDialog`.

Автор

Кашапов Ярослав

Дата

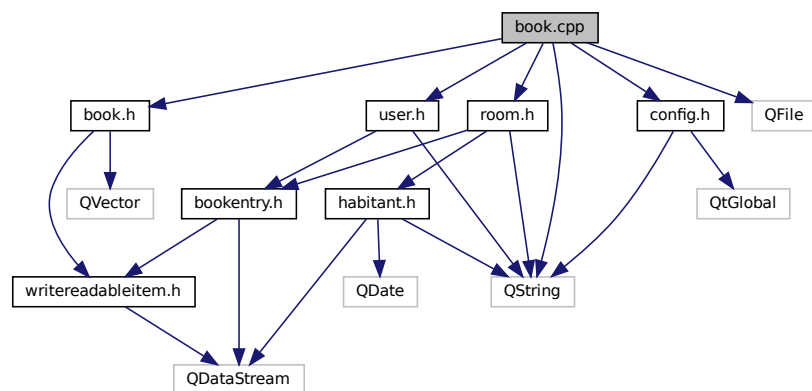
2021

5.3 Файл book.cpp

Файл реализации класса `book`.

```
#include "book.h"  
#include "config.h"  
#include "user.h"  
#include "room.h"  
#include <QFile>  
#include <QString>
```

Граф включаемых заголовчных файлов для `book.cpp`:



5.3.1 Подробное описание

Файл реализации класса book.

Автор

Кашапов Ярослав

Дата

2021

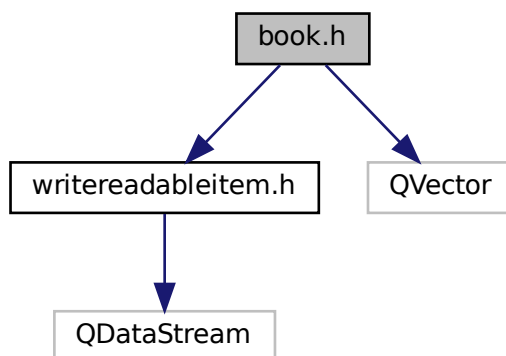
5.4 Файл book.h

Заголовочный файл класса book.

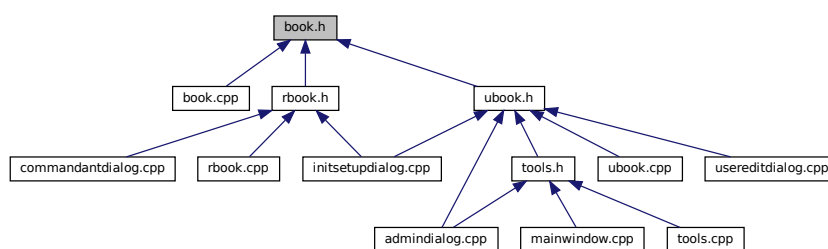
```
#include "writereadableitem.h"
```

```
#include <QVector>
```

Граф включаемых заголовочных файлов для book.h:



Граф файлов, в которые включается этот файл:



Классы

- class `book<entry>`

Шаблонный класс-контейнер с функциями сохранения в файл. Используется для хранения пользователей и комнат в системе. Наследуется от класса `writeReadableItem`, переопределяя его методы.

5.4.1 Подробное описание

Заголовчный файл класса `book`.

Автор

Кашапов Ярослав

Дата

2021

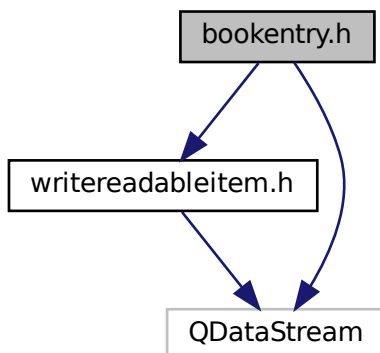
5.5 Файл bookentry.h

Заголовчный файл класса `bookEntry`.

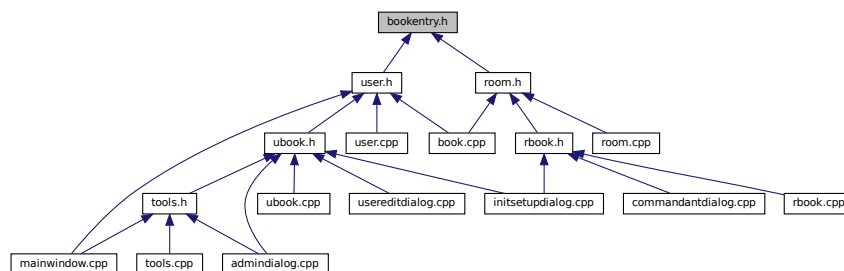
```
#include "writereadableitem.h"
```

```
#include <QDataStream>
```

Граф включаемых заголовчных файлов для `bookentry.h`:



Граф файлов, в которые включается этот файл:



Классы

- class [bookEntry](#)

Класс шаблонного элемента контейнера системы. Наследуется от класса [writeReadableItem](#), определяя операторы `>>` и `<<` над методами [writeReadableItem::write](#) и [writeReadableItem::read](#).

Функции

- `QDataStream & operator<< (QDataStream &ost, const bookEntry &e)`
Реализация оператора `<<`.
- `QDataStream & operator>> (QDataStream &ist, bookEntry &e)`
Реализация оператора `>>`

5.5.1 Подробное описание

Заголовчный файл класса [bookEntry](#).

Автор

Кашапов Ярослав

Дата

2021

5.5.2 Функции

5.5.2.1 operator<<()

```
QDataStream& operator<< (
    QDataStream & ost,
    const bookEntry & e ) [inline]
```

Реализация оператора `<<`.

Дружественное определение оператора `<<`.

Аргументы

ost	поток, в который происходит запись.
e	элемент, записываемый в поток.

5.5.2.2 operator>>()

```
QDataStream& operator>> (
    QDataStream & ist,
    bookEntry & e ) [inline]
```

Реализация оператора >>

Дружественное определение оператора >>.

Аргументы

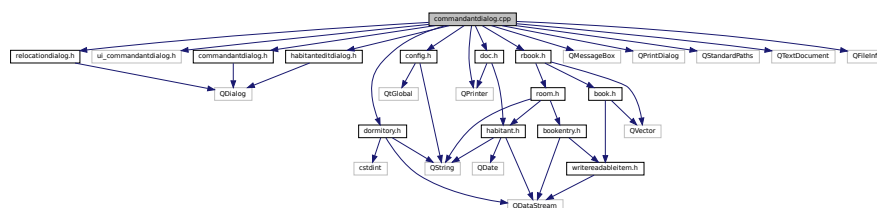
ost	поток, из которого происходит чтение.
e	элемент, считываемый из потока.

5.6 Файл commandantdialog.cpp

Файл реализации класса `commandantDialog`.

```
#include "commandantdialog.h"
#include "ui_commandantdialog.h"
#include "habitanteditdialog.h"
#include "relocationdialog.h"
#include "doc.h"
#include "dormitory.h"
#include "config.h"
#include "rbook.h"
#include <QMessageBox>
#include <QPrinter>
#include <QPrintDialog>
#include <QStandardPaths>
#include <QTextDocument>
#include <QFileInfo>
```

Граф включаемых заголовочных файлов для commandantdialog.cpp:



Макросы

- `#define ROOM_COLUMN 0`
- `#define SID_COLUMN 1`
- `#define NAME_COLUMN 2`
- `#define BDATE_COLUMN 3`

5.6.1 Подробное описание

Файл реализации класса `commandantDialog`.

Автор

Кашапов Ярослав

Дата

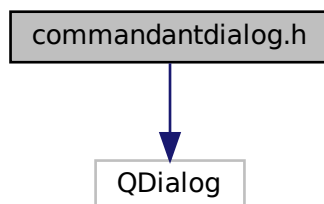
2021

5.7 Файл `commandantdialog.h`

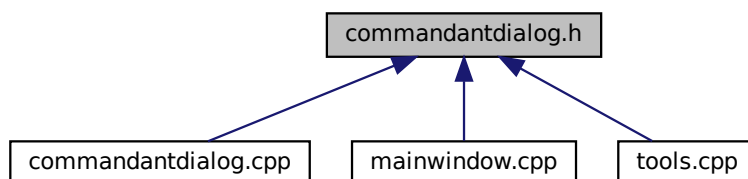
Заголовочный файл класса `commandantDialog`.

```
#include <QDialog>
```

Граф включаемых заголовочных файлов для `commandantdialog.h`:



Граф файлов, в которые включается этот файл:



Классы

- class `commandantDialog`

Класс диалога панели коменданта. Для функционирования необходима коллекция комнат.

5.7.1 Подробное описание

Заголовочный файл класса `commandantDialog`.

Автор

Кашапов Ярослав

Дата

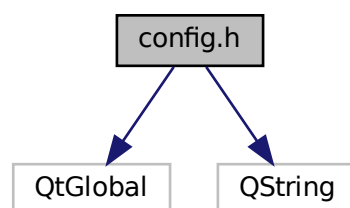
2021

5.8 Файл config.h

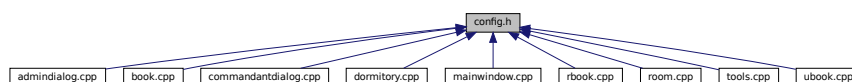
Файл конфигурации.

```
#include <QtGlobal>
#include <QString>
```

Граф включаемых заголовочных файлов для config.h:



Граф файлов, в которые включается этот файл:



Переменные

- `const char config::applicationName [] = QT_TRANSLATE_NOOP("Config", "dWell")`
Название приложения.
- `const char config::applicationVersion [] = "20210104"`
Версия приложения.
- `const QString config::fileUsers = "users.dwl"`
название файла, в котором записаны пользователи.
- `const QString config::fileRooms = "rooms.dwl"`
название файла, в котором записаны комнаты общежития.
- `const QString config::dormConf = "dorm.dwl"`
название файла, в котором записаны сведения об общежитии.

5.8.1 Подробное описание

Файл конфигурации.

Автор

Кашапов Ярослав

Дата

2021

5.8.2 Переменные

5.8.2.1 applicationName

```
const char config::applicationName[] = QT_TRANSLATE_NOOP("Config", "dWell")
```

Название приложения.

Заметки

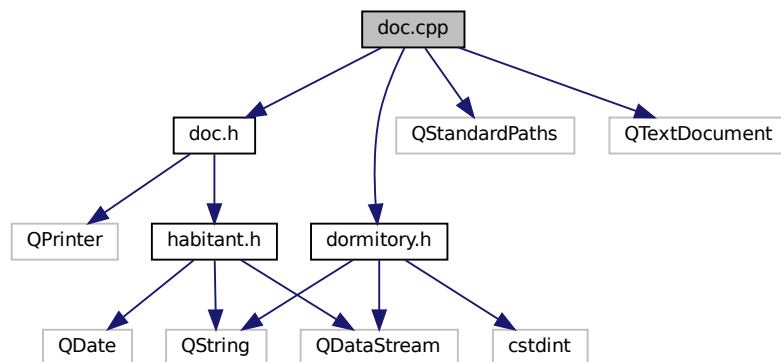
`QT_TRANSLATE_NOOP()` используется, чтобы строки отображались в файле строк для перевода на другие языки в контексте `Config`.

5.9 Файл doc.cpp

Файл реализации класса doc.

```
#include "doc.h"  
#include "dormitory.h"  
#include <QStandardPaths>  
#include <QTextDocument>
```

Граф включаемых заголовочных файлов для doc.cpp:



5.9.1 Подробное описание

Файл реализации класса doc.

Автор

Кашапов Ярослав

Дата

2021

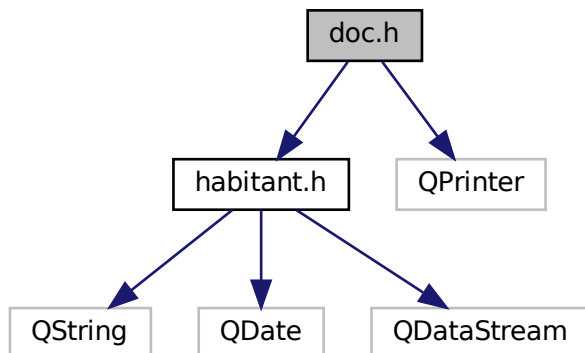
5.10 Файл doc.h

Заголовочный файл класса doc.

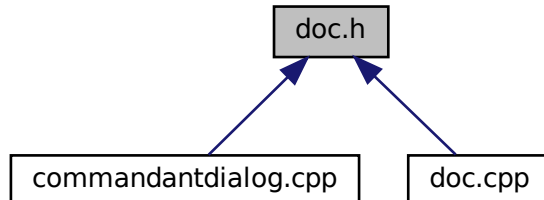
```
#include "habitant.h"
```

```
#include <QPrinter>
```

Граф включаемых заголовочных файлов для doc.h:



Граф файлов, в которые включается этот файл:



Классы

- class [doc](#)

Класс справки.

5.10.1 Подробное описание

Заголовочный файл класса doc.

Автор

Кашапов Ярослав

Дата

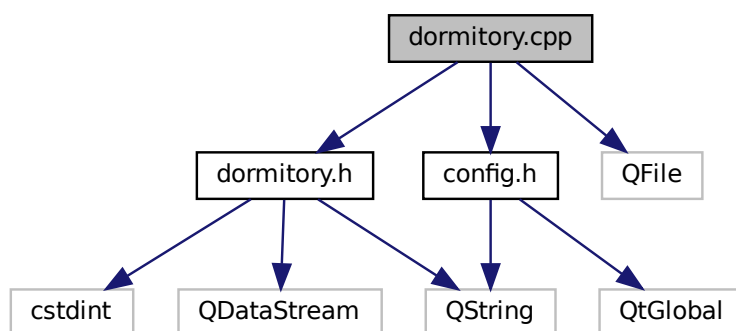
2021

5.11 Файл dormitory.cpp

Файл реализации класса dormitory.

```
#include "dormitory.h"  
#include "config.h"  
#include <QFile>
```

Граф включаемых заголовочных файлов для dormitory.cpp:



5.11.1 Подробное описание

Файл реализации класса dormitory.

Автор

Кашапов Ярослав

Дата

2021

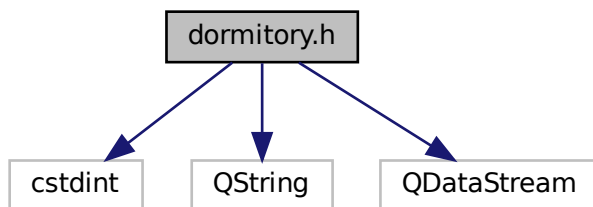
5.12 Файл dormitory.h

Заголовочный файл класса dormitory.

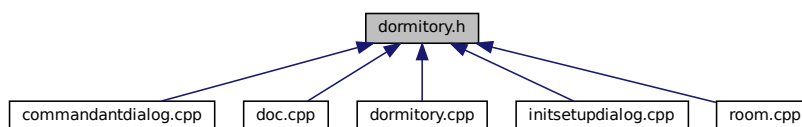
```
#include <cstdint>  
#include <QString>
```

```
#include <QDataStream>
```

Граф включаемых заголовочных файлов для dormitory.h:



Граф файлов, в которые включается этот файл:



Классы

- class `dormitory`

Класс, предоставляющий сведения об общежитии.

5.12.1 Подробное описание

Заголовочный файл класса `dormitory`.

Автор

Кашапов Ярослав

Дата

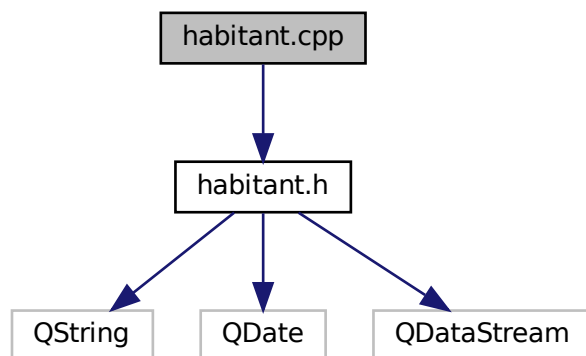
2021

5.13 Файл habitant.cpp

Файл реализации класса habitant.

```
#include "habitant.h"
```

Граф включаемых заголовочных файлов для habitant.cpp:



5.13.1 Подробное описание

Файл реализации класса habitant.

Автор

Кашапов Ярослав

Дата

2021

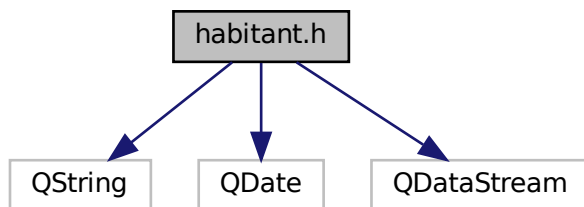
5.14 Файл habitant.h

Заголовочный файл класса habitant.

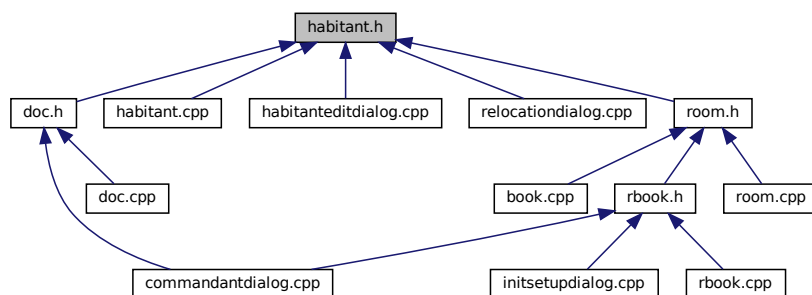
```
#include <QString>
#include <QDate>
```

```
#include <QDataStream>
```

Граф включаемых заголовочных файлов для `habitant.h`:



Граф файлов, в которые включается этот файл:



Классы

- class `habitant`

Класс проживающего.

- struct `habitant::habitantData`

Структура, хранящая информацию о проживающем. Используется для перемещения этой информации внутри программы.

Функции

- `QDataStream & operator<< (QDataStream &ost, const habitant &h)`
- `QDataStream & operator>> (QDataStream &ist, habitant &h)`

5.14.1 Подробное описание

Заголовочный файл класса `habitant`.

Автор

Кашапов Ярослав

Дата

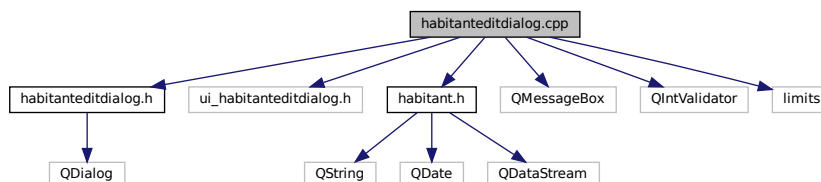
2021

5.15 Файл habitanteditdialog.cpp

Файл реализации класса [habitantEditDialog](#).

```
#include "habitanteditdialog.h"  
#include "ui_habitanteditdialog.h"  
#include "habitant.h"  
#include <QMessageBox>  
#include <QIntValidator>  
#include <limits>
```

Граф включаемых заголовочных файлов для habitanteditdialog.cpp:



5.15.1 Подробное описание

Файл реализации класса [habitantEditDialog](#).

Автор

Кашапов Ярослав

Дата

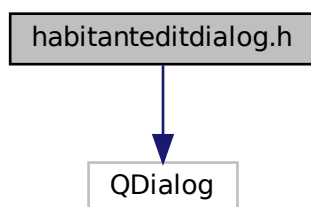
2021

5.16 Файл habitanteditdialog.h

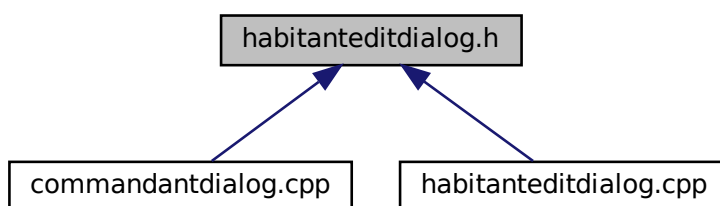
Заголовчный файл класса `habitantEditDialog`.

```
#include <QDialog>
```

Граф включаемых заголовчных файлов для `habitanteditdialog.h`:



Граф файлов, в которые включается этот файл:



Классы

- class `habitantEditDialog`
Класс диалога редактирования проживающего (его данных).

5.16.1 Подробное описание

Заголовчный файл класса `habitantEditDialog`.

Автор

Кашапов Ярослав

Дата

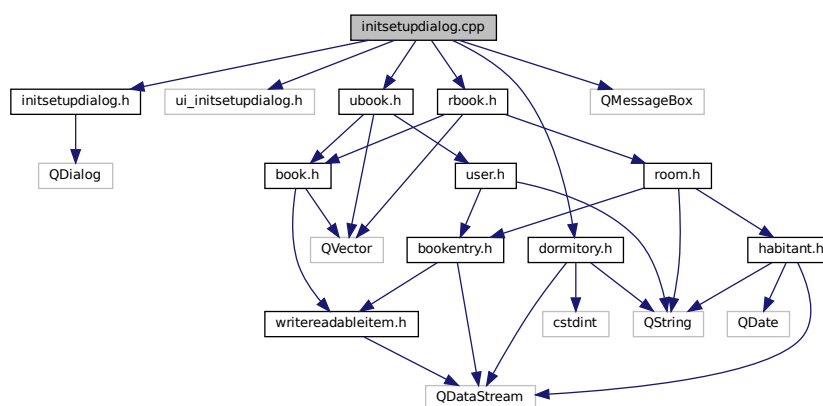
2021

5.17 Файл initsetupdialog.cpp

Файл реализации класса [initSetupDialog](#).

```
#include "initsetupdialog.h"
#include "ui_initsetupdialog.h"
#include "ubook.h"
#include "dormitory.h"
#include "rbook.h"
#include <QMessageBox>
```

Граф включаемых заголовочных файлов для initsetupdialog.cpp:



5.17.1 Подробное описание

Файл реализации класса [initSetupDialog](#).

Автор

Кашапов Ярослав

Дата

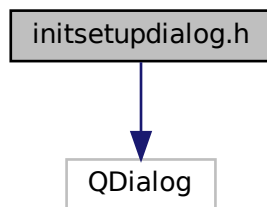
2021

5.18 Файл initsetupdialog.h

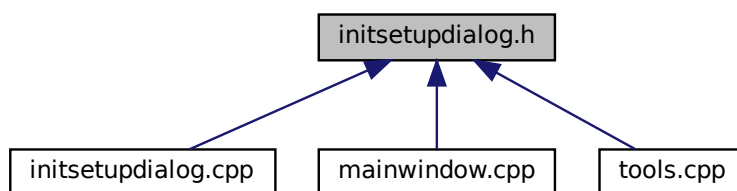
Заголовочный файл класса [initSetupDialog](#).

```
#include <QDialog>
```

Граф включаемых заголовочных файлов для `initsetupdialog.h`:



Граф файлов, в которые включается этот файл:



Классы

- class [initSetupDialog](#)

Класс диалога начальной настройки системы.

5.18.1 Подробное описание

Заголовочный файл класса [initSetupDialog](#).

Автор

Кашапов Ярослав

Дата

2021

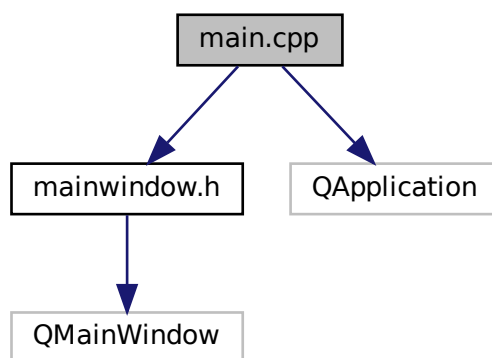
5.19 Файл main.cpp

Файл, содержащий точку входа в программу.

```
#include "mainwindow.h"
```

```
#include <QApplication>
```

Граф включаемых заголовочных файлов для main.cpp:



Функции

- `int main (int argc, char *argv[])`

5.19.1 Подробное описание

Файл, содержащий точку входа в программу.

Автор

Кашапов Ярослав

Дата

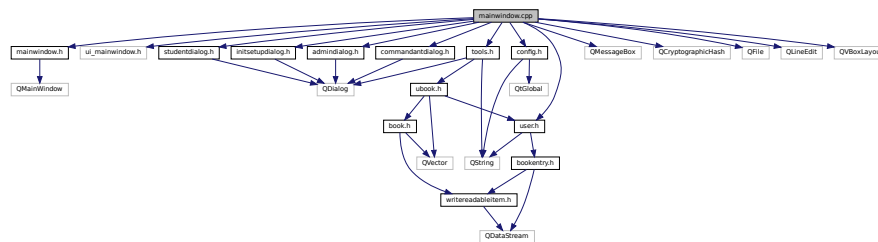
2021

5.20 Файл mainwindow.cpp

Файл реализации класса [MainWindow](#).

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "admindialog.h"
#include "commandantdialog.h"
#include "studentdialog.h"
#include "initsetupdialog.h"
#include "config.h"
#include "user.h"
#include "tools.h"
#include <QMessageBox>
#include <QCryptographicHash>
#include <QFile>
#include <QLineEdit>
#include <QVBoxLayout>
```

Граф включаемых заголовочных файлов для mainwindow.cpp:



5.20.1 Подробное описание

Файл реализации класса [MainWindow](#).

Автор

Кашапов Ярослав

Дата

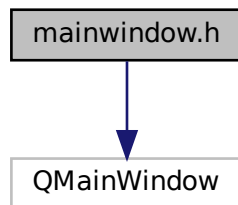
2021

5.21 Файл mainwindow.h

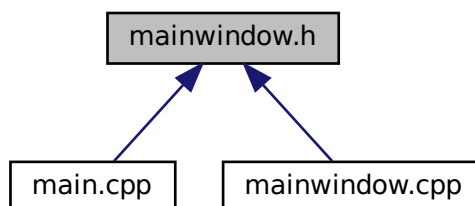
Заголовочный файл класса [MainWindow](#).

```
#include <QMainWindow>
```

Граф включаемых заголовочных файлов для/mainwindow.h:



Граф файлов, в которые включается этот файл:



Классы

- class [MainWindow](#)

Класс диалога входа в систему.

5.21.1 Подробное описание

Заголовочный файл класса [MainWindow](#).

Автор

Кашапов Ярослав

Дата

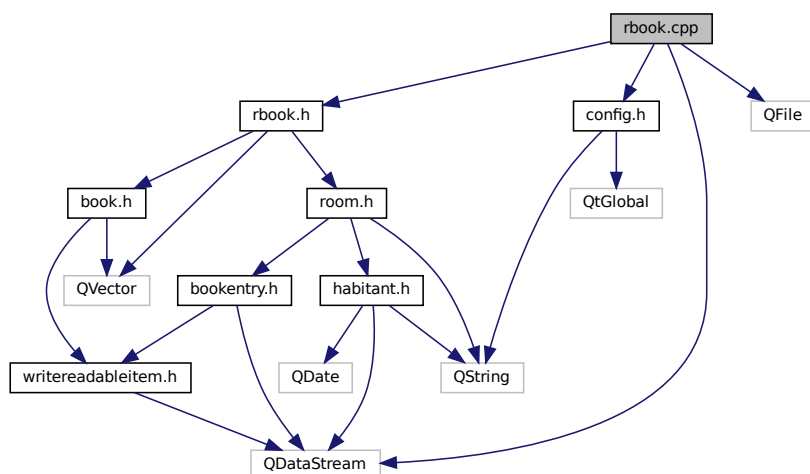
2021

5.22 Файл rbook.cpp

Файл реализации класса rbook.

```
#include "rbook.h"
#include "config.h"
#include <QFile>
#include <QDataStream>
```

Граф включаемых заголовочных файлов для rbook.cpp:



5.22.1 Подробное описание

Файл реализации класса rbook.

Автор

Кашапов Ярослав

Дата

2021

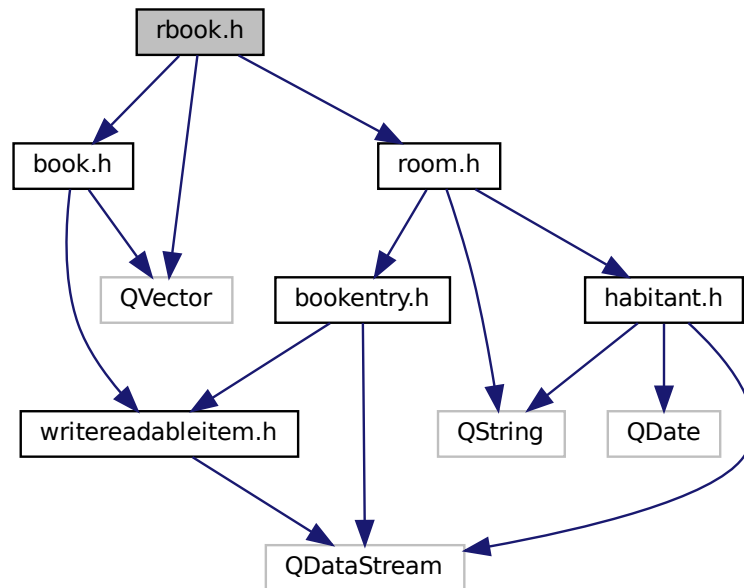
5.23 Файл rbook.h

Заголовочный файл класса rbook.

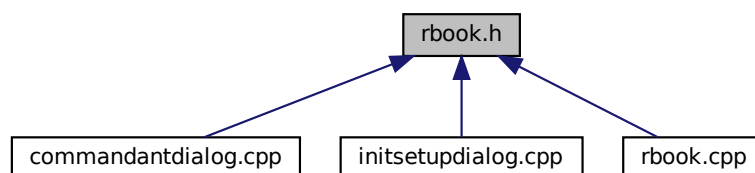
```
#include <QVector>
#include "room.h"
```

```
#include "book.h"
```

Граф включаемых заголовочных файлов для rbook.h:



Граф файлов, в которые включается этот файл:



Классы

- class `rbook`

Класс контейнер комнат. Наследуется от класса `book`. Добавляет необходимый функционал для доступа к контейнеру комнат.

5.23.1 Подробное описание

Заголовочный файл класса `rbook`.

Автор

Кашапов Ярослав

Дата

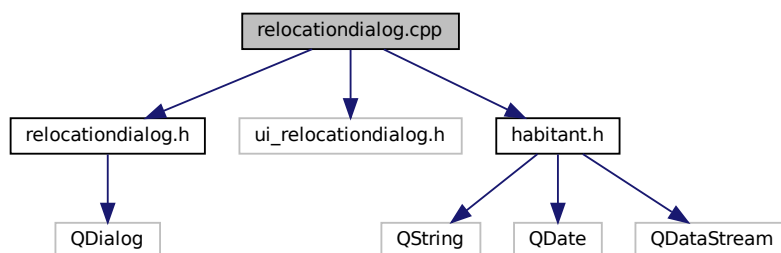
2021

5.24 Файл relocationdialog.cpp

Файл реализации класса relocationdialog.

```
#include "relocationdialog.h"  
#include "ui_relocationdialog.h"  
#include "habitant.h"
```

Граф включаемых заголовочных файлов для relocationdialog.cpp:



5.24.1 Подробное описание

Файл реализации класса relocationdialog.

Автор

Кашапов Ярослав

Дата

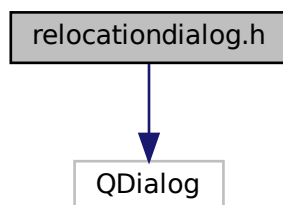
2021

5.25 Файл relocationdialog.h

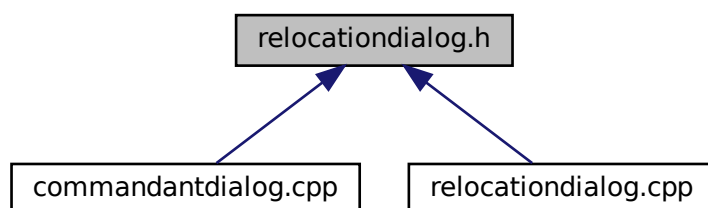
Заголовочный файл класса [relocationDialog](#).

```
#include <QDialog>
```

Граф включаемых заголовочных файлов для relocationdialog.h:



Граф файлов, в которые включается этот файл:



Классы

- class [relocationDialog](#)

Класс диалога переселения проживающего.

5.25.1 Подробное описание

Заголовочный файл класса [relocationDialog](#).

Автор

Кашапов Ярослав

Дата

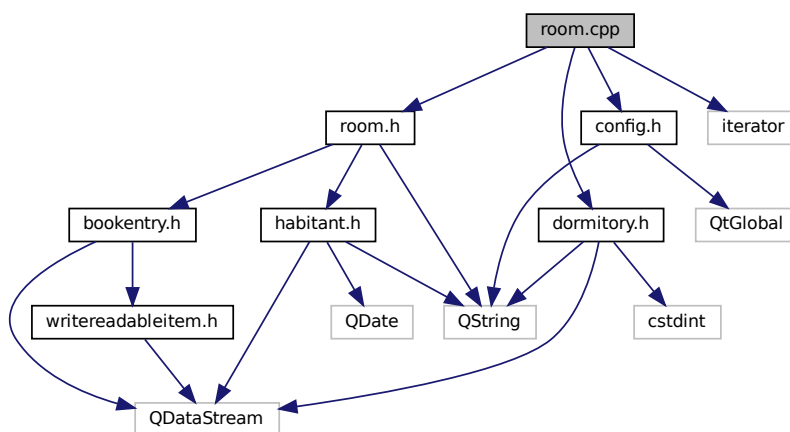
2021

5.26 Файл room.cpp

Файл реализации класса room.

```
#include "room.h"
#include "dormitory.h"
#include "config.h"
#include <iterator>
```

Граф включаемых заголовочных файлов для room.cpp:



5.26.1 Подробное описание

Файл реализации класса room.

Автор

Кашапов Ярослав

Дата

2021

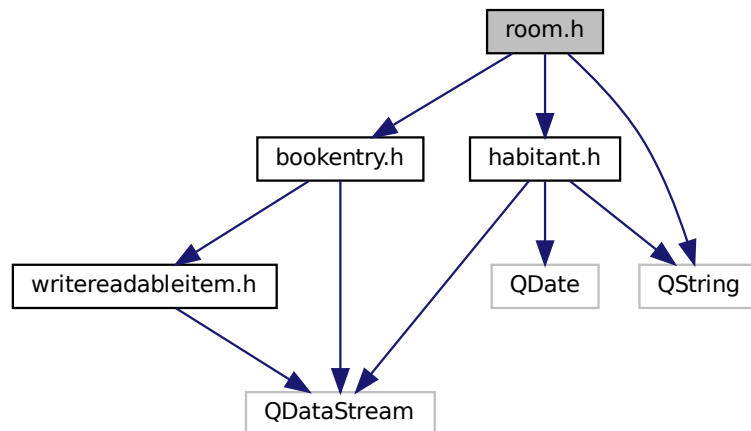
5.27 Файл room.h

Заголовочный файл класса room.

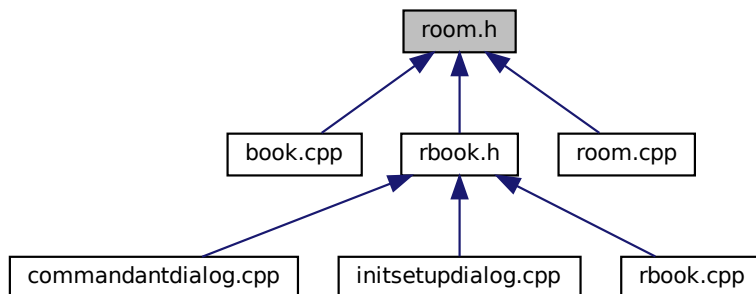
```
#include "bookentry.h"
#include "habitant.h"
```

```
#include <QString>
```

Граф включаемых заголовочных файлов для room.h:



Граф файлов, в которые включается этот файл:



Классы

- class [room](#)

Класс комнаты. Наследуется от класса [bookEntry](#). Содержит контейнер проживающих в комнате.

5.27.1 Подробное описание

Заголовочный файл класса room.

Автор

Кашапов Ярослав

Дата

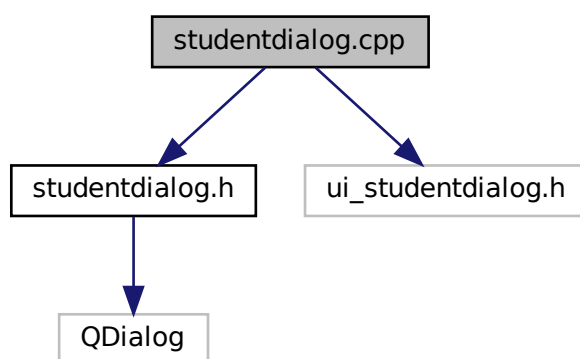
2021

5.28 Файл studentdialog.cpp

Файл реализации класса `studentDialog`.

```
#include "studentdialog.h"  
#include "ui_studentdialog.h"
```

Граф включаемых заголовочных файлов для studentdialog.cpp:



5.28.1 Подробное описание

Файл реализации класса `studentDialog`.

Автор

Кашапов Ярослав

Дата

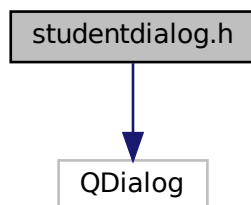
2021

5.29 Файл studentdialog.h

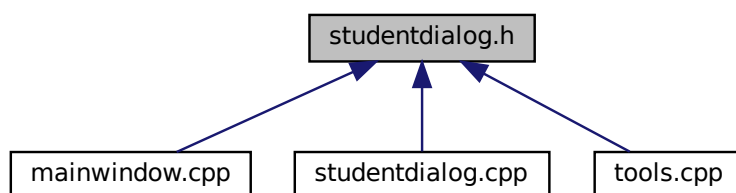
Заголовочный файл класса `studentDialog`.

```
#include <QDialog>
```

Граф включаемых заголовочных файлов для studentdialog.h:



Граф файлов, в которые включается этот файл:



Классы

- class `studentDialog`
Класс диалога студента.

5.29.1 Подробное описание

Заголовочный файл класса `studentDialog`.

Автор

Кашапов Ярослав

Дата

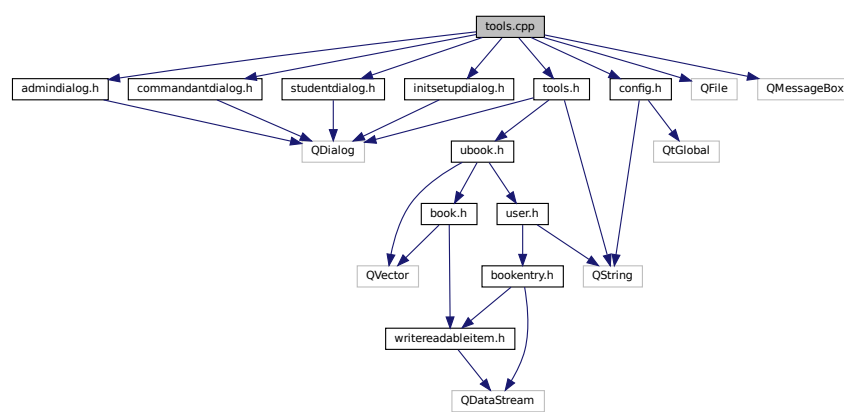
2021

5.30 Файл tools.cpp

Файл реализации класса tools.

```
#include "tools.h"
#include "config.h"
#include "initsetupdialog.h"
#include "adminialog.h"
#include "commandantdialog.h"
#include "studentdialog.h"
#include <QFile>
#include <QMessageBox>
```

Граф включаемых заголовочных файлов для tools.cpp:



5.30.1 Подробное описание

Файл реализации класса tools.

Автор

Кашапов Ярослав

Дата

2021

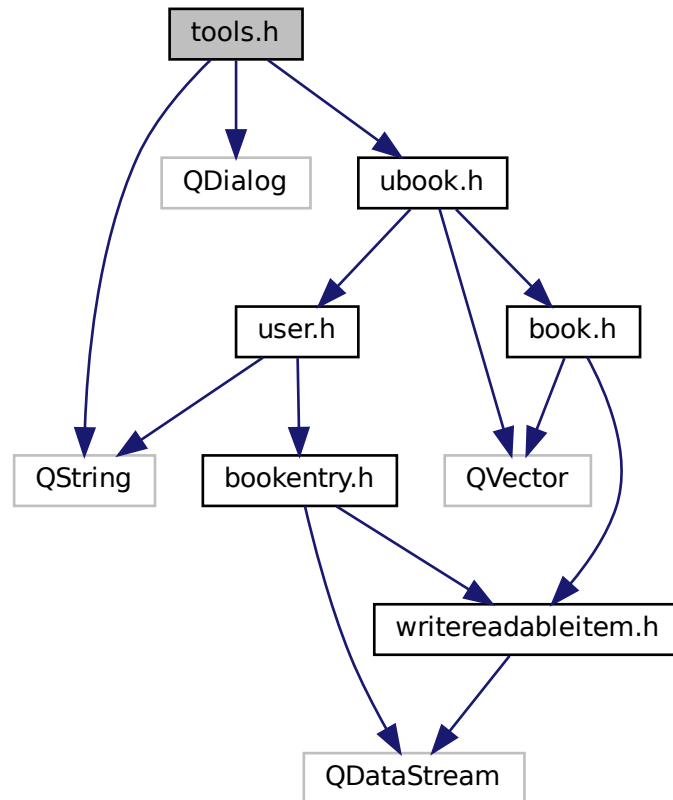
5.31 Файл tools.h

Заголовочный файл класса tools.

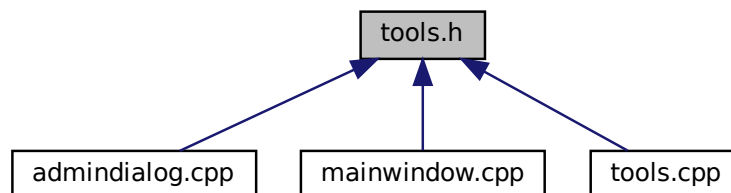
```
#include <QString>
#include <QDialog>
```

```
#include "ubook.h"
```

Граф включаемых заголовочных файлов для tools.h:



Граф файлов, в которые включается этот файл:



Классы

- class [tools](#)

Класс вспомогательных инструментов.

5.31.1 Подробное описание

Заголовочный файл класса tools.

Автор

Кашапов Ярослав

Дата

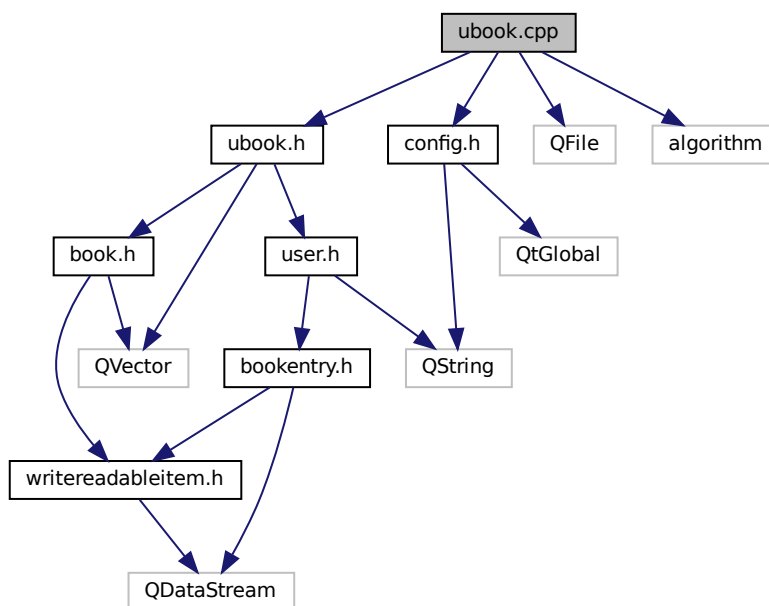
2021

5.32 Файл ubook.cpp

Файл реализации класса ubook.

```
#include "ubook.h"  
#include "config.h"  
#include <QFile>  
#include <algorithm>
```

Граф включаемых заголовочных файлов для ubook.cpp:



5.32.1 Подробное описание

Файл реализации класса ubook.

Автор

Кашапов Ярослав

Дата

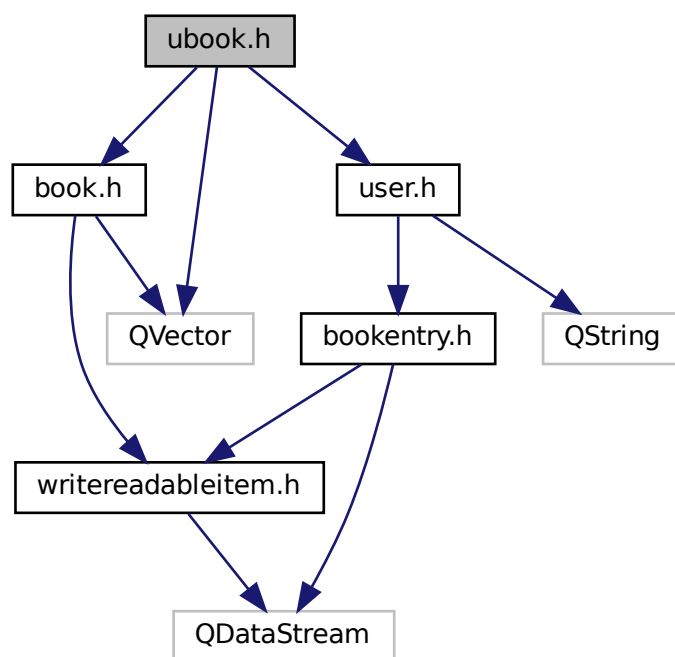
2021

5.33 Файл ubook.h

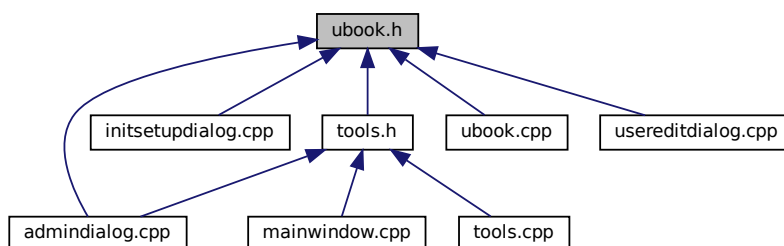
Заголовочный файл класса ubook.

```
#include "book.h"  
#include "user.h"  
#include <QVector>
```

Граф включаемых заголовочных файлов для ubook.h:



Граф файлов, в которые включается этот файл:



Классы

- class `ubook`

Класс контейнер пользователей. Наследуется от класса `book`. Добавляет необходимый функционал для доступа к контейнеру `book`.

5.33.1 Подробное описание

Заголовочный файл класса `ubook`.

Автор

Кашапов Ярослав

Дата

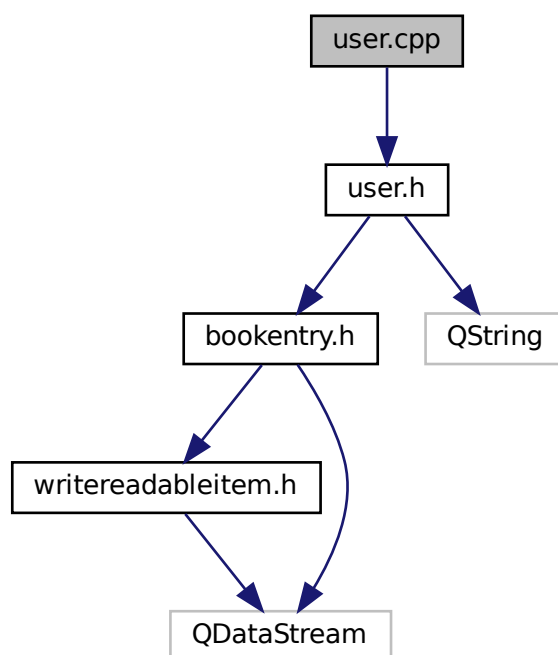
2021

5.34 Файл `user.cpp`

Файл реализации класса `user`.

```
#include "user.h"
```

Граф включаемых заголовочных файлов для `user.cpp`:



5.34.1 Подробное описание

Файл реализации класса user.

Автор

Кашапов Ярослав

Дата

2021

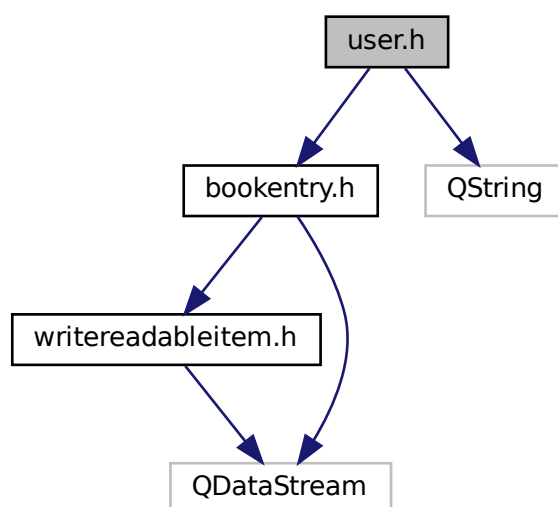
5.35 Файл user.h

Заголовочный файл класса user.

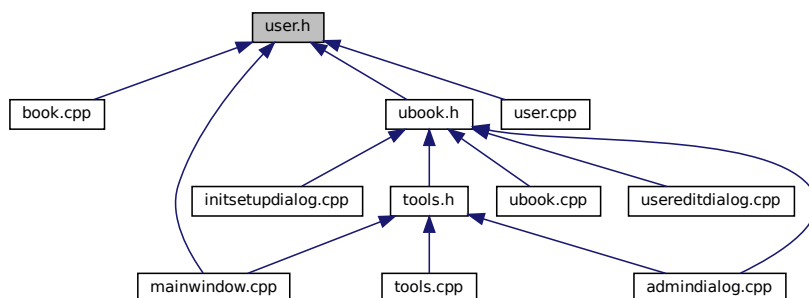
```
#include "bookentry.h"
```

```
#include <QString>
```

Граф включаемых заголовочных файлов для user.h:



Граф файлов, в которые включается этот файл:



Классы

- class [user](#)

Класс комнаты. Наследуется от класса [bookEntry](#).

5.35.1 Подробное описание

Заголовчный файл класса `user`.

Автор

Кашапов Ярослав

Дата

2021

5.36 Файл `usereditdialog.cpp`

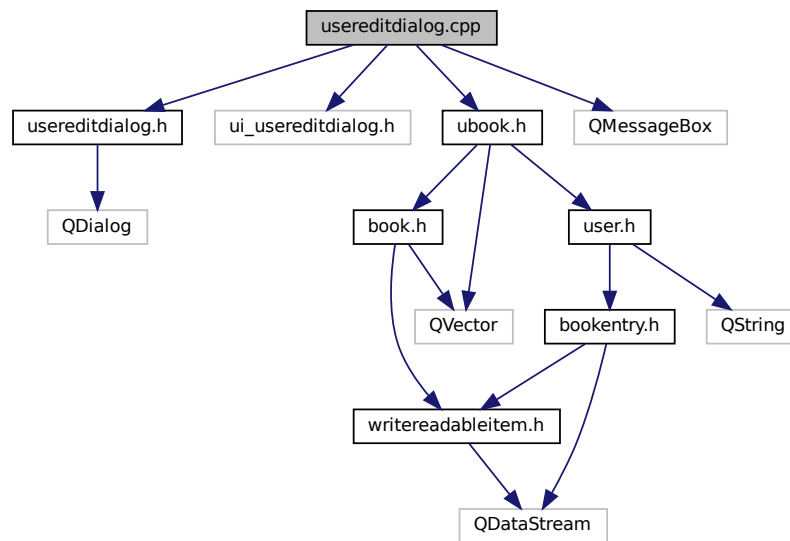
Файл реализации класса [userEditDialog](#).

```

#include "usereditdialog.h"
#include "ui_usereditdialog.h"
#include "ubook.h"
  
```

```
#include <QMessageBox>
```

Граф включаемых заголовочных файлов для `usereditdialog.cpp`:



5.36.1 Подробное описание

Файл реализации класса [userEditDialog](#).

Автор

Кашапов Ярослав

Дата

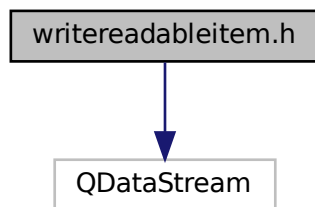
2021

5.37 Файл writereadableitem.h

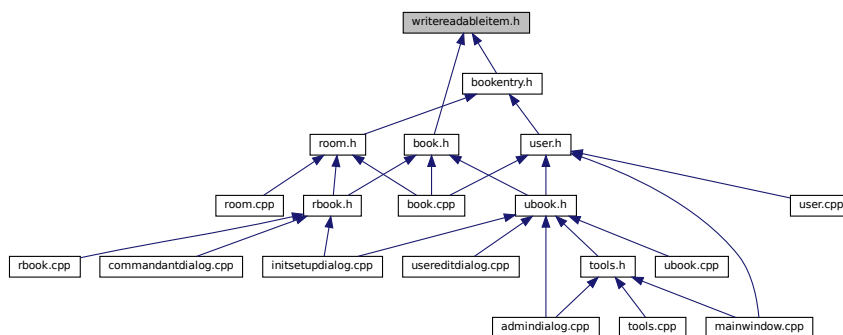
Заголовочный файл класса [writeReadableItem](#).

```
#include <QDataStream>
```

Граф включаемых заголовочных файлов для `writereadableitem.h`:



Граф файлов, в которые включается этот файл:



Классы

- class [writeReadableItem](#)

Абстрактный класс объекта, способного записываться и считываться из потока.

5.37.1 Подробное описание

Заголовочный файл класса [writeReadableItem](#).

Автор

Кашапов Ярослав

Дата

2021

Предметный указатель

adminDialog, 7
 adminDialog, 9
 on_tableWidget_doubleClicked, 9
 ui, 9
admindialog.cpp, 47
admindialog.h, 48
applicationName
 config.h, 56
availableForCheckin
 rbook, 29
 room, 34

book< entry >, 10
 loadFromFile, 11
 operator[], 12
 read, 12
 saveToFile, 13
 write, 13
book.cpp, 49
book.h, 50
bookEntry, 13
 operator<<, 15
 operator>>, 15
bookentry.h, 51
 operator<<, 52
 operator>>, 53

checkin
 rbook, 29
 room, 34
checkout
 rbook, 30
 room, 35
commandantDialog, 15
 commandantDialog, 17
commandantdialog.cpp, 53
commandantdialog.h, 54
config.h, 55
 applicationName, 56

doc, 18
 generate, 18
doc.cpp, 57
doc.h, 57
dormitory, 19
 dormitory, 19
 getDormCfg, 20
dormitory.cpp, 59
dormitory.h, 59

findUser
 ubook, 40
findUserByName
 ubook, 40

generate
 doc, 18
getDormCfg
 dormitory, 20
getHabitantBySid
 rbook, 30
getRbook
 rbook, 30
getUbook
 ubook, 41

habitant, 20
 habitant, 22
habitant.cpp, 61
habitant.h, 61
habitant::habitantData, 22
habitantEditDialog, 23
 habitantEditDialog, 24
habitanteditdialog.cpp, 63
habitanteditdialog.h, 64
habitantsbook, 24

initSetupDialog, 25
 initSetupDialog, 26
initsetupdialog.cpp, 65
initsetupdialog.h, 65

loadFromFile
 book< entry >, 11

main.cpp, 67
MainWindow, 26
 MainWindow, 27
mainwindow.cpp, 68
mainwindow.h, 68

on_tableWidget_doubleClicked
 adminDialog, 9
operator<<
 bookEntry, 15
 bookentry.h, 52
operator>>
 bookEntry, 15
 bookentry.h, 53
operator[]

- book< entry >, 12
- room, 35
- rbook, 28
 - availableForCheckin, 29
 - checkin, 29
 - checkout, 30
 - getHabitantBySid, 30
 - getRbook, 30
- rbook.cpp, 70
- rbook.h, 70
- read
 - book< entry >, 12
 - room, 35
- relocationDialog, 31
 - relocationDialog, 32
- relocationdialog.cpp, 72
- relocationdialog.h, 73
- room, 32
 - availableForCheckin, 34
 - checkin, 34
 - checkout, 35
 - operator[], 35
 - read, 35
 - write, 36
- room.cpp, 74
- room.h, 74
- saveToFile
 - book< entry >, 13
- studentDialog, 36
 - studentDialog, 37
- studentdialog.cpp, 76
- studentdialog.h, 77
- tools, 38
- tools.cpp, 78
- tools.h, 78
- ubook, 39
 - findUser, 40
 - findUserByName, 40
 - getUbook, 41
- ubook.cpp, 80
- ubook.h, 81
- ui
 - adminDialog, 9
- user, 41
 - user, 43
- user.cpp, 82
- user.h, 83
- userEditDialog, 43
 - userEditDialog, 45
- usereditdialog.cpp, 84
- write
 - book< entry >, 13
 - room, 36
- writeReadableItem, 45
- writereadableitem.h, 85