

SquareEquationSolver

Создано системой Doxygen 1.8.14

## Содержание

1	Иерархический список классов	1
1.1	Иерархия классов . . . . .	1
2	Алфавитный указатель классов	2
2.1	Классы . . . . .	2
3	Список файлов	2
3.1	Файлы . . . . .	2
4	Классы	3
4.1	Класс Console . . . . .	3
4.1.1	Подробное описание . . . . .	3
4.1.2	Конструктор(ы) . . . . .	3
4.1.3	Методы . . . . .	4
4.2	Класс GetterApp . . . . .	6
4.2.1	Подробное описание . . . . .	7
4.2.2	Конструктор(ы) . . . . .	7
4.2.3	Методы . . . . .	7
4.2.4	Данные класса . . . . .	8
4.3	Класс HelpApp . . . . .	8
4.3.1	Подробное описание . . . . .	9
4.3.2	Конструктор(ы) . . . . .	9
4.3.3	Методы . . . . .	9
4.4	Класс IApp . . . . .	10
4.4.1	Подробное описание . . . . .	11
4.4.2	Конструктор(ы) . . . . .	11
4.4.3	Методы . . . . .	11
4.4.4	Данные класса . . . . .	12
4.5	Класс NullOutputStream . . . . .	13
4.5.1	Конструктор(ы) . . . . .	13
4.5.2	Методы . . . . .	13
4.6	Класс SetterApp . . . . .	14
4.6.1	Подробное описание . . . . .	15
4.6.2	Конструктор(ы) . . . . .	15
4.6.3	Методы . . . . .	15
4.6.4	Данные класса . . . . .	16
4.7	Класс SolverApp . . . . .	16
4.7.1	Конструктор(ы) . . . . .	17
4.7.2	Методы . . . . .	17
4.7.3	Данные класса . . . . .	18

5	Файлы	19
5.1	Файл include/app.h . . . . .	19
5.2	Файл include/console.h . . . . .	20
5.2.1	Перечисления . . . . .	20
5.3	Файл include/getterapp.h . . . . .	21
5.4	Файл include/helpapp.h . . . . .	22
5.5	Файл include/setterapp.h . . . . .	23
5.6	Файл include/solverapp.h . . . . .	24
5.7	Файл src/console.cpp . . . . .	25
5.8	Файл src/getterapp.cpp . . . . .	26
5.9	Файл src/helpapp.cpp . . . . .	26
5.10	Файл src/main.cpp . . . . .	27
5.10.1	Функции . . . . .	28
5.11	Файл src/setterapp.cpp . . . . .	29
5.12	Файл src/solverapp.cpp . . . . .	29
5.12.1	Макросы . . . . .	30
5.12.2	Функции . . . . .	30
	Алфавитный указатель	31

## 1 Иерархический список классов

### 1.1 Иерархия классов

Иерархия классов.

Console	3
IApp	10
GetterApp	6
HelpApp	8
SetterApp	14
SolverApp	16
ostream	
NullOutputStream	13

## 2 Алфавитный указатель классов

### 2.1 Классы

Классы с их кратким описанием.

<a href="#">Console</a>	
Командный интерпретатор	3
<a href="#">GetterApp</a>	6
<a href="#">HelpApp</a>	8
<a href="#">IApp</a>	
Интерфейс приложения	10
<a href="#">NullOutputStream</a>	13
<a href="#">SetterApp</a>	14
<a href="#">SolverApp</a>	16

## 3 Список файлов

### 3.1 Файлы

Полный список файлов.

<a href="#">include/app.h</a>	19
<a href="#">include/console.h</a>	20
<a href="#">include/getterapp.h</a>	21
<a href="#">include/helpapp.h</a>	22
<a href="#">include/setterapp.h</a>	23
<a href="#">include/solverapp.h</a>	24
<a href="#">src/console.cpp</a>	25
<a href="#">src/getterapp.cpp</a>	26
<a href="#">src/helpapp.cpp</a>	26
<a href="#">src/main.cpp</a>	27
<a href="#">src/setterapp.cpp</a>	29
<a href="#">src/solverapp.cpp</a>	29

## 4 Классы

### 4.1 Класс Console

Командный интерпретатор

```
#include <console.h>
```

Открытые члены

- [Console](#) (std::istream &in, std::ostream &out)
- [~Console](#) ()
- template<class App , class... Args>  
bool [emplaceApp](#) (const std::string &appName, Args &&...args)
- std::ostream & [output](#) () const
- std::ostream & [log](#) ([Verbosity](#) verbosity, const char \*prompt="") const
- std::ostream & [debug](#) () const
- std::ostream & [info](#) () const
- std::ostream & [error](#) () const
- std::istream & [input](#) () const
- [Verbosity](#) [getVerbosity](#) () const
- int [exec](#) (int argc, char \*argv[])
- std::string [getVariable](#) (const std::string &name, const std::string &defaultValue="") const
- void [setVariable](#) (const std::string &name, const std::string &value)
- const std::map< std::string, [IApp](#) \* > & [getApps](#) () const
- const std::map< std::string, std::string > & [getAllVariables](#) () const

#### 4.1.1 Подробное описание

Командный интерпретатор

Главный класс командного интерпретатора, содержащий указатели на приложения (см. [IApp](#)), переменные, потоки ввода-вывода.

#### 4.1.2 Конструктор(ы)

##### 4.1.2.1 Console()

```
Console::Console (
    std::istream & in,
    std::ostream & out ) [inline]
```

##### 4.1.2.2 ~Console()

```
Console::~Console ( )
```

### 4.1.3 Методы

#### 4.1.3.1 debug()

```
std::ostream & Console::debug ( ) const
```

Эквивалентно `log(VERB_DEBUG, "# [DEBUG] ")` (см. [log](#))

#### 4.1.3.2 emplaceApp()

```
template<class App , class... Args>
bool Console::emplaceApp (
    const std::string & appName,
    Args &&... args ) [inline]
```

Создаёт приложение класса App

Аргументы

in	appName	команда, связанная с приложением
in	...args	аргументы конструктора

Возвращает

`true`, если приложение успешно добавлено, и `false`, если

#### 4.1.3.3 error()

```
std::ostream & Console::error ( ) const
```

Эквивалентно `log(VERB_ERROR, "# [ERROR] ")` (см. [log](#))

#### 4.1.3.4 exec()

```
int Console::exec (
    int argc,
    char * argv[] )
```

Исполняет основной цикл командного интерпретатора

#### 4.1.3.5 getAllVariables()

```
const std::map< std::string, std::string > & Console::getAllVariables ( ) const
```

Возвращает все установленные переменные в виде отображения "название" -> "значение"

## 4.1.3.6 getAppс()

```
const std::map< std::string, IApp * > & Console::getApps ( ) const
```

Возвращает все установленные приложения в виде отображения "команда" -> "приложение"

## 4.1.3.7 getVariable()

```
std::string Console::getVariable (
    const std::string & name,
    const std::string & defaultValue = "" ) const
```

Возвращает значение переменной

Аргументы

in	name	название переменной
in	defaultValue	если переменная с именем name не установлена, вызов вернёт defaultValue

## 4.1.3.8 getVerbosity()

```
Verbosity Console::getVerbosity ( ) const
```

Возвращает уровень важности, исходя из значения переменной verbosity (см. [getVariable](#))

## 4.1.3.9 info()

```
std::ostream & Console::info ( ) const
```

Эквивалентно `log(VERB_INFO, "# [INFO ] ")` (см. [log](#))

## 4.1.3.10 input()

```
std::istream & Console::input ( ) const
```

Возвращает поток ввода

## 4.1.3.11 log()

```
std::ostream & Console::log (
    Verbosity verbosity,
    const char * prompt = "" ) const
```

Возвращает поток логирования

Аргументы

in	verbosity	уровень важности
in	prompt	заголовок

#### 4.1.3.12 output()

```
std::ostream & Console::output ( ) const
```

Возвращает поток вывода

#### 4.1.3.13 setVariable()

```
void Console::setVariable (
    const std::string & name,
    const std::string & value )
```

Устанавливает значение переменной name равным value

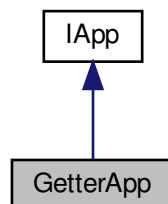
Объявления и описания членов классов находятся в файлах:

- `include/console.h`
- `src/console.cpp`

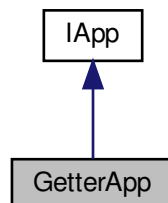
## 4.2 Класс GetterApp

```
#include <getterapp.h>
```

Граф наследования:GetterApp:



Граф связей класса GetterApp:





## Открытые члены

- [GetterApp](#) (const [Console](#) \*parent)
- virtual int [exec](#) (const std::vector< std::string > &args)  
Запуск приложения
- virtual const char \* [getStatusCodeDescription](#) (int statusCode)
- virtual const char \* [getHelp](#) ()

## Статические открытые данные

- static constexpr int [STATUS\\_BAD\\_ARGUMENTS](#) = 1

## 4.2.1 Подробное описание

Приложение, позволяющее узнать значение переменных (см. [Console](#))

## 4.2.2 Конструктор(ы)

## 4.2.2.1 GetterApp()

```
GetterApp::GetterApp (
    const Console * parent ) [inline]
```

## 4.2.3 Методы

## 4.2.3.1 exec()

```
int GetterApp::exec (
    const std::vector< std::string > & args ) [virtual]
```

Запуск приложения

Аргументы

in	args	аргументы, разобранные командным интерпретатором
----	------	--

Возвращает

Статус. Если статус равен [STATUS\\_OK](#), приложение считается успешно завершённым. Если в результате работы приложения произошла ошибка, приложение возвращает иной код. Подробнее см. описания конкретных приложений.

Замещает [IApp](#).

#### 4.2.3.2 getHelp()

```
const char * GetterApp::getHelp ( ) [virtual]
```

Возвращает текстовое описание приложения.

Замещает [IApp](#).

#### 4.2.3.3 getStatusCodeDescription()

```
const char * GetterApp::getStatusCodeDescription (
    int statusCode ) [virtual]
```

По коду ошибки возвращает текстовое описание.

Аргументы

in	statusCode	код, который вернуло приложение
----	------------	---------------------------------

Замещает [IApp](#).

#### 4.2.4 Данные класса

##### 4.2.4.1 STATUS\_BAD\_ARGUMENTS

```
constexpr int GetterApp::STATUS_BAD_ARGUMENTS = 1 [static]
```

Аргументы не соответствуют требуемому формату

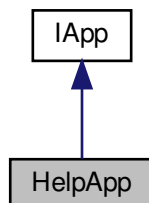
Объявления и описания членов классов находятся в файлах:

- [include/getterapp.h](#)
- [src/getterapp.cpp](#)

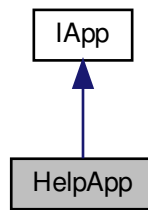
#### 4.3 Класс HelpApp

```
#include <helpapp.h>
```

Граф наследования:HelpApp:



Граф связей класса HelpApp:



Открытые члены

- `HelpApp` (const `Console` \*parent)
- `~HelpApp` ()
- int `exec` (const std::vector< std::string > &args)  
Запуск приложения
- const char \* `getStatusCodeDescription` (int statusCode)
- const char \* `getHelp` ()

Дополнительные унаследованные члены

#### 4.3.1 Подробное описание

Приложение, выводящее на экран справочную информацию

#### 4.3.2 Конструктор(ы)

##### 4.3.2.1 HelpApp()

```
HelpApp::HelpApp (
    const Console * parent ) [inline], [explicit]
```

##### 4.3.2.2 ~HelpApp()

```
HelpApp::~HelpApp ( )
```

#### 4.3.3 Методы

##### 4.3.3.1 exec()

```
int HelpApp::exec (
    const std::vector< std::string > & args ) [virtual]
```

Запуск приложения

## Аргументы

in	args	аргументы, разобранные командным интерпретатором
----	------	--

## Возвращает

Статус. Если статус равен `STATUS_OK`, приложение считается успешно завершённым. Если в результате работы приложения произошла ошибка, приложение возвращает иной код. Подробнее см. описания конкретных приложений.

Замещает [IApp](#).

4.3.3.2 `getHelp()`

```
const char * HelpApp::getHelp ( ) [virtual]
```

Возвращает текстовое описание приложения.

Замещает [IApp](#).

4.3.3.3 `getStatusCodeDescription()`

```
const char * HelpApp::getStatusCodeDescription (
    int statusCode ) [virtual]
```

По коду ошибки возвращает текстовое описание.

## Аргументы

in	statusCode	код, который вернуло приложение
----	------------	---------------------------------

Замещает [IApp](#).

Объявления и описания членов классов находятся в файлах:

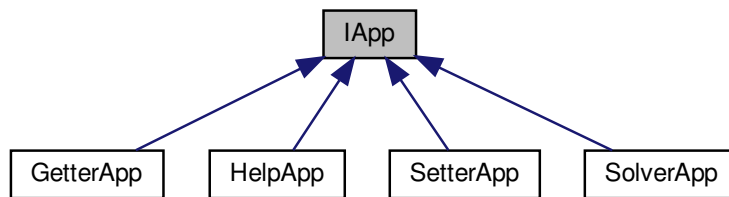
- `include/helpapp.h`
- `src/helpapp.cpp`

4.4 Класс `IApp`

## Интерфейс приложения

```
#include <app.h>
```

Граф наследования:IApp:



Открытые члены

- virtual [~IApp](#) ()
- virtual int [exec](#) (const std::vector< std::string > &args)=0  
Запуск приложения
- virtual const char \* [getStatusCodeDescription](#) (int statusCode)=0
- virtual const char \* [getHelp](#) ()=0

Статические открытые данные

- static constexpr int [STATUS\\_OK](#) = 0

#### 4.4.1 Подробное описание

Интерфейс приложения

Проект представляет собой командный интерпретатор, который читает из входного потока команды. За реализацию команд отвечают приложения — объекты классов, унаследованных от [IApp](#).

См. также

[Console](#)

#### 4.4.2 Конструктор(ы)

##### 4.4.2.1 ~IApp()

```
virtual IApp::~IApp ( ) [inline], [virtual]
```

#### 4.4.3 Методы

##### 4.4.3.1 exec()

```
virtual int IApp::exec (
    const std::vector< std::string > & args ) [pure virtual]
```

Запуск приложения

## Аргументы

in	args	аргументы, разобранные командным интерпретатором
----	------	--

## Возвращает

Статус. Если статус равен `STATUS_OK`, приложение считается успешно завершённым. Если в результате работы приложения произошла ошибка, приложение возвращает иной код. Подробнее см. описания конкретных приложений.

Замещается в [SolverApp](#), [GetterApp](#), [HelpApp](#) и [SetterApp](#).

4.4.3.2 `getHelp()`

```
virtual const char* IApp::getHelp ( ) [pure virtual]
```

Возвращает текстовое описание приложения.

Замещается в [SolverApp](#), [GetterApp](#), [HelpApp](#) и [SetterApp](#).

4.4.3.3 `getStatusCodeDescription()`

```
virtual const char* IApp::getStatusCodeDescription (
    int statusCode ) [pure virtual]
```

По коду ошибки возвращает текстовое описание.

## Аргументы

in	statusCode	код, который вернуло приложение
----	------------	---------------------------------

Замещается в [SolverApp](#), [GetterApp](#), [HelpApp](#) и [SetterApp](#).

## 4.4.4 Данные класса

4.4.4.1 `STATUS_OK`

```
constexpr int IApp::STATUS_OK = 0 [static]
```

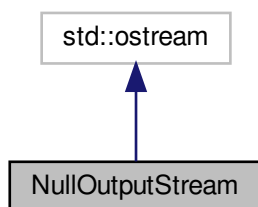
Статус, соответствующий успешному завершению приложения.

Объявления и описания членов класса находятся в файле:

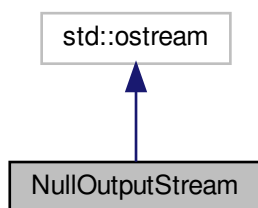
- `include/app.h`

4.5 Класс `NullOutputStream`

Граф наследования: `NullOutputStream`:



Граф связей класса `NullOutputStream`:



Открытые члены

- [NullOutputStream](#) ()
- `template<class T >`  
[NullOutputStream](#) & `operator<<` (T &&)

## 4.5.1 Конструктор(ы)

4.5.1.1 `NullOutputStream()`

`NullOutputStream::NullOutputStream ( )` [inline]

## 4.5.2 Методы

## 4.5.2.1 operator&lt;&lt;()

```
template<class T >
NullOutputStream& NullOutputStream::operator<< (
    T && ) [inline]
```

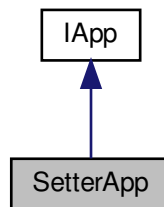
Объявления и описания членов класса находятся в файле:

- [src/console.cpp](#)

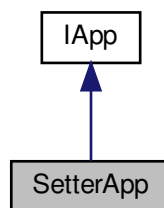
## 4.6 Класс SetterApp

```
#include <setterapp.h>
```

Граф наследования:SetterApp:



Граф связей класса SetterApp:



Открытые члены

- [SetterApp](#) ([Console](#) \*parent)
- virtual int [exec](#) (const std::vector< std::string > &args)  
Запуск приложения
- virtual const char \* [getStatusCodeDescription](#) (int statusCode)
- virtual const char \* [getHelp](#) ()



Статические открытые данные

- static constexpr int [STATUS\\_BAD\\_ARGUMENTS](#) = 1

#### 4.6.1 Подробное описание

Приложение, позволяющее устанавливать значения переменных (см. [Console](#))

#### 4.6.2 Конструктор(ы)

##### 4.6.2.1 SetterApp()

```
SetterApp::SetterApp (
    Console * parent ) [inline], [explicit]
```

#### 4.6.3 Методы

##### 4.6.3.1 exec()

```
int SetterApp::exec (
    const std::vector< std::string > & args ) [virtual]
```

Запуск приложения

Аргументы

in	args	аргументы, разобранные командным интерпретатором
----	------	--

Возвращает

Статус. Если статус равен [STATUS\\_OK](#), приложение считается успешно завершённым. Если в результате работы приложения произошла ошибка, приложение возвращает иной код. Подробнее см. описания конкретных приложений.

Замещает [IApp](#).

##### 4.6.3.2 getHelp()

```
const char * SetterApp::getHelp ( ) [virtual]
```

Возвращает текстовое описание приложения.

Замещает [IApp](#).

#### 4.6.3.3 `getStatusCodeDescription()`

```
const char * SetterApp::getStatusCodeDescription (
    int statusCode ) [virtual]
```

По коду ошибки возвращает текстовое описание.

Аргументы

in	statusCode	код, который вернуло приложение
----	------------	---------------------------------

Замещает [IApp](#).

#### 4.6.4 Данные класса

##### 4.6.4.1 `STATUS_BAD_ARGUMENTS`

```
constexpr int SetterApp::STATUS_BAD_ARGUMENTS = 1 [static]
```

Аргументы не соответствуют требуемому формату

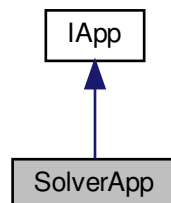
Объявления и описания членов классов находятся в файлах:

- `include/setterapp.h`
- `src/setterapp.cpp`

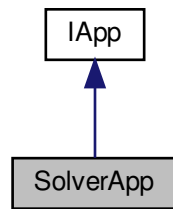
## 4.7 Класс SolverApp

```
#include <solverapp.h>
```

Граф наследования: SolverApp:



Граф связей класса SolverApp:



Открытые члены

- [SolverApp](#) (const [Console](#) \*parent)
- virtual int [exec](#) (const std::vector< std::string > &args)  
Запуск приложения
- virtual const char \* [getStatusCodeDescription](#) (int statusCode)
- virtual const char \* [getHelp](#) ()

Статические открытые данные

- static constexpr int [STATUS\\_BAD\\_ARGUMENTS](#) = 1
- static constexpr int [STATUS\\_BAD\\_FIELD](#) = 2
- static constexpr int [STATUS\\_PARSE\\_ERROR](#) = 3

#### 4.7.1 Конструктор(ы)

##### 4.7.1.1 SolverApp()

```
SolverApp::SolverApp (
    const Console * parent ) [inline]
```

#### 4.7.2 Методы

##### 4.7.2.1 exec()

```
int SolverApp::exec (
    const std::vector< std::string > & args ) [virtual]
```

Запуск приложения

## Аргументы

in	args	аргументы, разобранные командным интерпретатором
----	------	--

## Возвращает

Статус. Если статус равен `STATUS_OK`, приложение считается успешно завершённым. Если в результате работы приложения произошла ошибка, приложение возвращает иной код. Подробнее см. описания конкретных приложений.

Замещает [IApp](#).

4.7.2.2 `getHelp()`

```
const char * SolverApp::getHelp ( ) [virtual]
```

Возвращает текстовое описание приложения.

Замещает [IApp](#).

4.7.2.3 `getStatusCodeDescription()`

```
const char * SolverApp::getStatusCodeDescription (
    int statusCode ) [virtual]
```

По коду ошибки возвращает текстовое описание.

## Аргументы

in	statusCode	код, который вернуло приложение
----	------------	---------------------------------

Замещает [IApp](#).

## 4.7.3 Данные класса

4.7.3.1 `STATUS_BAD_ARGUMENTS`

```
constexpr int SolverApp::STATUS_BAD_ARGUMENTS = 1 [static]
```

Аргументы не соответствуют требуемому формату

4.7.3.2 `STATUS_BAD_FIELD`

```
constexpr int SolverApp::STATUS_BAD_FIELD = 2 [static]
```

Значение переменной `field` некорректно (см. [Console](#))

#### 4.7.3.3 STATUS\_PARSE\_ERROR

```
constexpr int SolverApp::STATUS_PARSE_ERROR = 3 [static]
```

Не удалось обработать входные данные

Объявления и описания членов классов находятся в файлах:

- [include/solverapp.h](#)
- [src/solverapp.cpp](#)

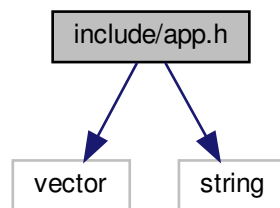
## 5 Файлы

### 5.1 Файл include/app.h

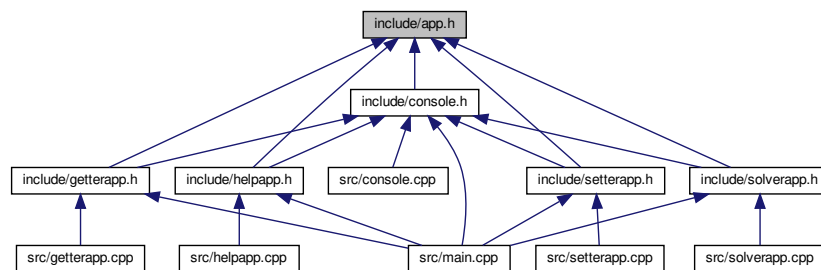
```
#include <vector>
```

```
#include <string>
```

Граф включаемых заголовочных файлов для app.h:



Граф файлов, в которые включается этот файл:



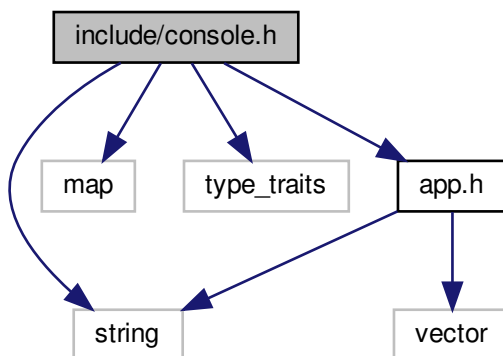
Классы

- [class IApp](#)  
Интерфейс приложения

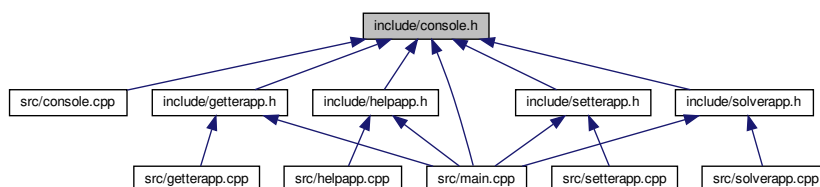
## 5.2 Файл include/console.h

```
#include <string>
#include <map>
#include <type_traits>
#include "app.h"
```

Граф включаемых заголовочных файлов для console.h:



Граф файлов, в которые включается этот файл:



### Классы

- class [Console](#)  
Командный интерпретатор

### Перечисления

- enum [Verbosity](#) { [VERB\\_DEBUG](#), [VERB\\_INFO](#), [VERB\\_ERROR](#) }  
Уровень вывода

#### 5.2.1 Перечисления

## 5.2.1.1 Verbosity

enum [Verbosity](#)

Уровень вывода

При выводе отладочной информации сообщениям присваивается уровень важности. Сообщения с низким уровнем важности будут проигнорированы. За минимальный допустимый уровень важности отвечает переменная `verbosity` (см. [Console::getVariable\(\)](#))

Элементы перечислений

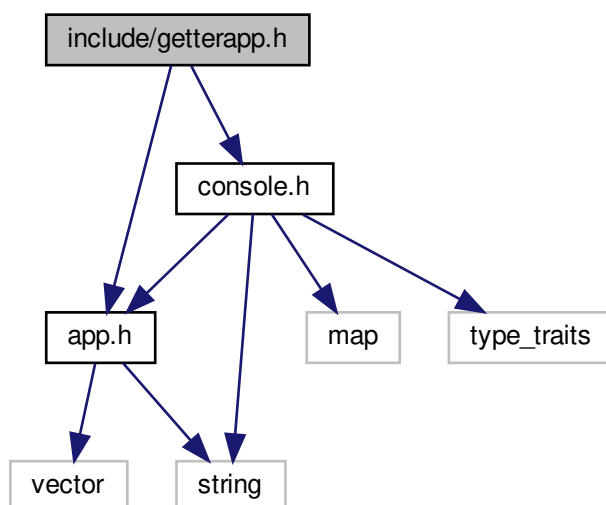
VERB_DEBUG	
VERB_INFO	
VERB_ERROR	

## 5.3 Файл include/getterapp.h

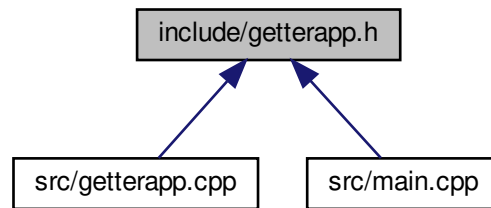
```
#include <app.h>
```

```
#include <console.h>
```

Граф включаемых заголовочных файлов для `getterapp.h`:



Граф файлов, в которые включается этот файл:



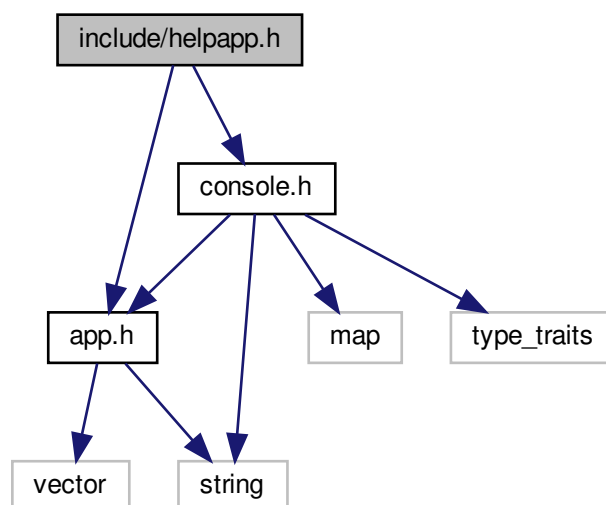
Классы

- class [GetterApp](#)

## 5.4 Файл include/helpapp.h

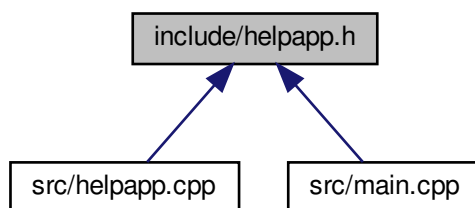
```
#include <app.h>  
#include <console.h>
```

Граф включаемых заголовочных файлов для helpapp.h:





Граф файлов, в которые включается этот файл:



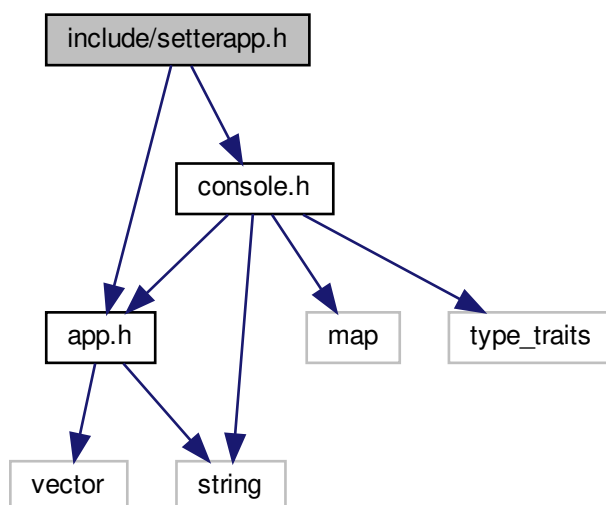
Классы

- class [HelpApp](#)

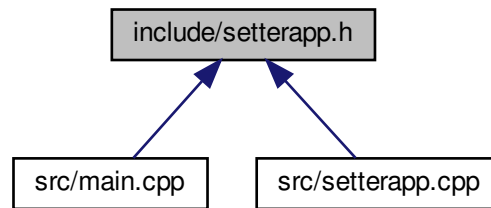
## 5.5 Файл include/setterapp.h

```
#include <app.h>  
#include <console.h>
```

Граф включаемых заголовочных файлов для setterapp.h:



Граф файлов, в которые включается этот файл:



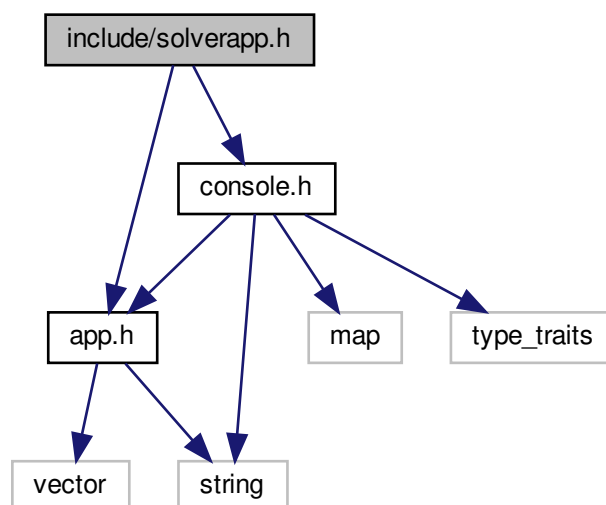
Классы

- class [SetterApp](#)

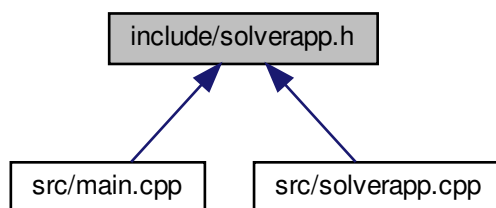
## 5.6 Файл `include/solverapp.h`

```
#include <app.h>  
#include <console.h>
```

Граф включаемых заголовочных файлов для `solverapp.h`:



Граф файлов, в которые включается этот файл:



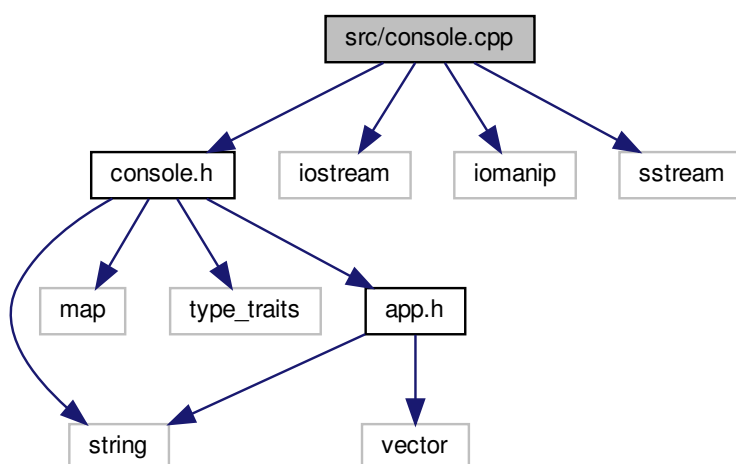
Классы

- class [SolverApp](#)

## 5.7 Файл src/console.cpp

```
#include <console.h>
#include <iostream>
#include <iomanip>
#include <sstream>
```

Граф включаемых заголовочных файлов для console.cpp:



Классы

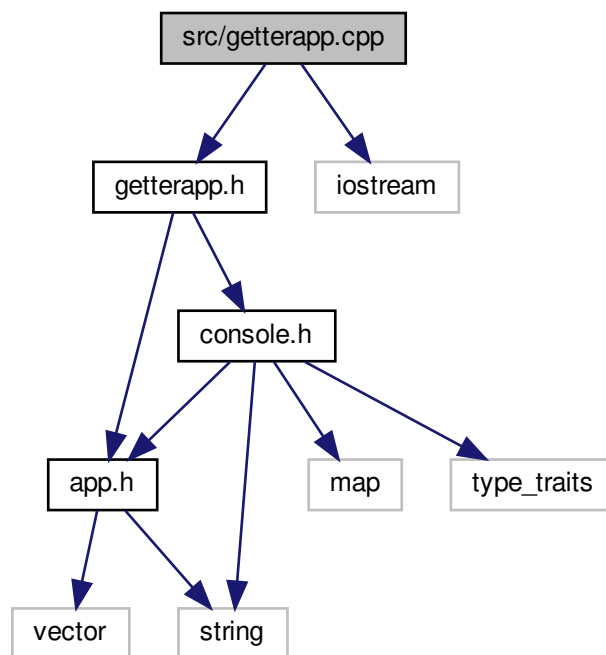
- class [NullOutputStream](#)

## 5.8 Файл src/getterapp.cpp

```
#include <getterapp.h>
```

```
#include <iostream>
```

Граф включаемых заголовочных файлов для getterapp.cpp:

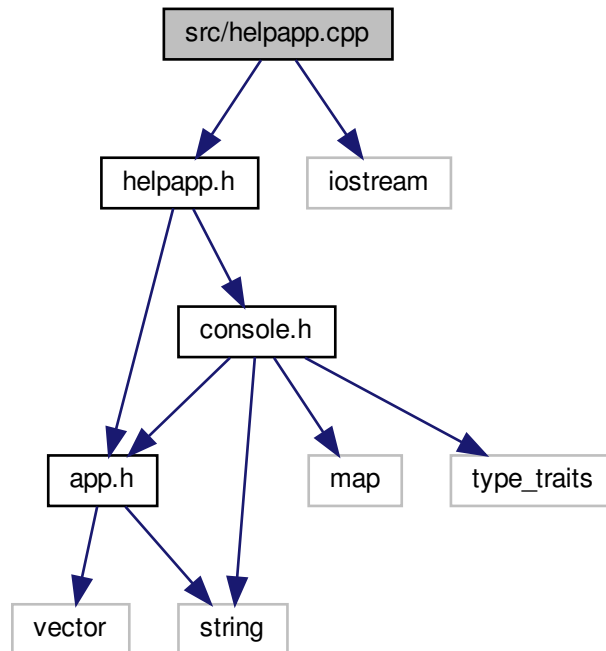


## 5.9 Файл src/helpapp.cpp

```
#include <helpapp.h>
```

```
#include <iostream>
```

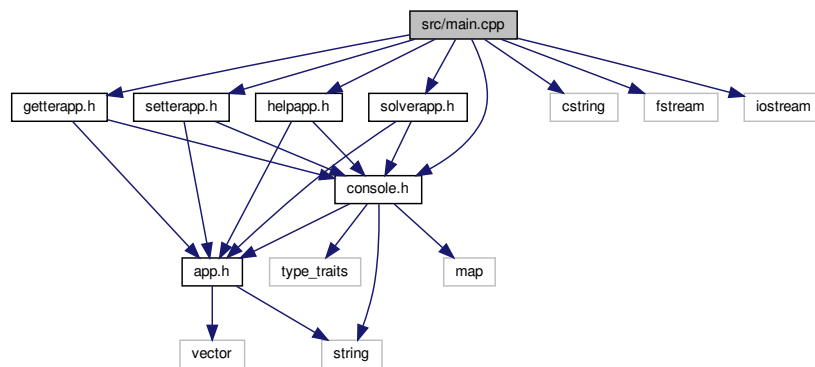
Граф включаемых заголовочных файлов для helpapp.cpp:



## 5.10 Файл src/main.cpp

```
#include <console.h>
#include <helpapp.h>
#include <setterapp.h>
#include <getterapp.h>
#include <solverapp.h>
#include <cstring>
#include <fstream>
#include <iostream>
```

Граф включаемых заголовочных файлов для main.cpp:



Функции

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

### 5.10.1 Функции

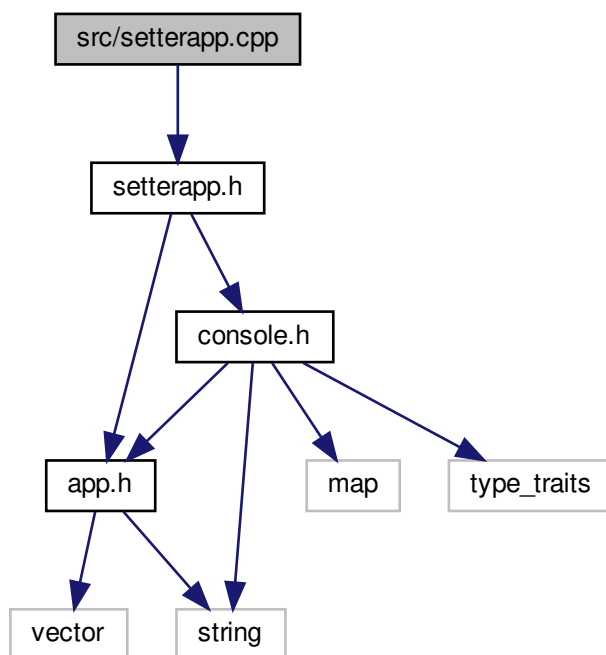
#### 5.10.1.1 main()

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

## 5.11 Файл src/setterapp.cpp

```
#include <setterapp.h>
```

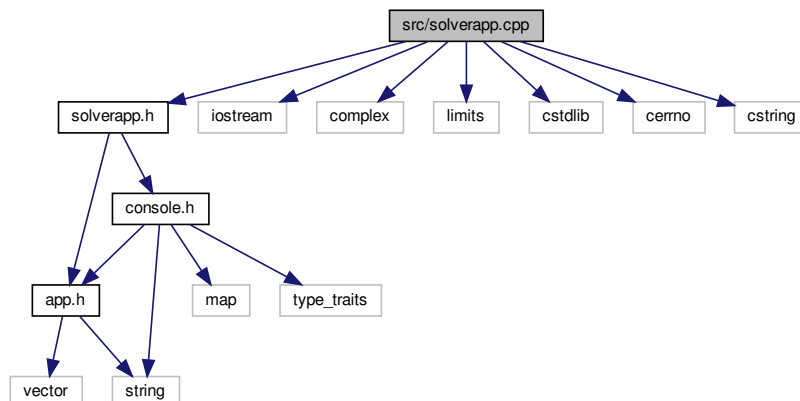
Граф включаемых заголовочных файлов для setterapp.cpp:



## 5.12 Файл src/solverapp.cpp

```
#include <solverapp.h>
#include <iostream>
#include <complex>
#include <limits>
#include <cstdlib>
#include <cerrno>
#include <cstring>
```

Граф включаемых заголовочных файлов для solverapp.cpp:



Макросы

- `#define OPT_PTR(type, x) static type default_##x##_var; if ( x == nullptr ) x = &default_##x##_var;`

Функции

- `template<>`  
`std::complex< double > SolverApp::parse< std::complex< double > > (const std::string &input,`  
`bool *ok) const`
- `std::ostream & operator<< (std::ostream &out, const std::complex< double > &val)`

## 5.12.1 Макросы

### 5.12.1.1 OPT\_PTR

```

#define OPT_PTR(
    type,
    x ) static type default_##x##_var; if ( x == nullptr ) x = &default_##x##_var;

```

## 5.12.2 Функции

### 5.12.2.1 operator<<()

```

std::ostream& operator<< (
    std::ostream & out,
    const std::complex< double > & val ) [inline]

```

### 5.12.2.2 SolverApp::parse< std::complex< double > >()

```

template<>
std::complex<double> SolverApp::parse< std::complex< double > > (
    const std::string & input,
    bool * ok ) const

```



## Предметный указатель

- ~Console
  - Console, 3
- ~HelpApp
  - HelpApp, 9
- ~IApp
  - IApp, 11
- Console, 3
  - ~Console, 3
  - Console, 3
  - debug, 4
  - emplaceApp, 4
  - error, 4
  - exec, 4
  - getAllVariables, 4
  - getApps, 4
  - getVariable, 5
  - getVerbosity, 5
  - info, 5
  - input, 5
  - log, 5
  - output, 6
  - setVariable, 6
- console.h
  - Verbosity, 20
- debug
  - Console, 4
- emplaceApp
  - Console, 4
- error
  - Console, 4
- exec
  - Console, 4
  - GetterApp, 7
  - HelpApp, 9
  - IApp, 11
  - SetterApp, 15
  - SolverApp, 17
- getAllVariables
  - Console, 4
- getApps
  - Console, 4
- getHelp
  - GetterApp, 7
  - HelpApp, 10
  - IApp, 12
  - SetterApp, 15
  - SolverApp, 18
- getStatusCodeDescription
  - GetterApp, 8
  - HelpApp, 10
  - IApp, 12
  - SetterApp, 15
  - SolverApp, 18
- getVariable
  - Console, 5
- getVerbosity
  - Console, 5
- GetterApp, 6
  - exec, 7
  - getHelp, 7
  - getStatusCodeDescription, 8
  - GetterApp, 7
  - STATUS\_BAD\_ARGUMENTS, 8
- HelpApp, 8
  - ~HelpApp, 9
  - exec, 9
  - getHelp, 10
  - getStatusCodeDescription, 10
  - HelpApp, 9
- IApp, 10
  - ~IApp, 11
  - exec, 11
  - getHelp, 12
  - getStatusCodeDescription, 12
  - STATUS\_OK, 12
- include/app.h, 19
- include/console.h, 20
- include/getterapp.h, 21
- include/helpapp.h, 22
- include/setterapp.h, 23
- include/solverapp.h, 24
- info
  - Console, 5
- input
  - Console, 5
- log
  - Console, 5
- main
  - main.cpp, 28
- main.cpp
  - main, 28
- NullOutputStream, 13
  - NullOutputStream, 13
  - operator<<, 13
- OPT\_PTR
  - solverapp.cpp, 30
- operator<<
  - NullOutputStream, 13
  - solverapp.cpp, 30
- output
  - Console, 6
- STATUS\_BAD\_ARGUMENTS

- GetterApp, [8](#)
- SetterApp, [16](#)
- SolverApp, [18](#)
- STATUS\_BAD\_FIELD
  - SolverApp, [18](#)
- STATUS\_OK
  - IApp, [12](#)
- STATUS\_PARSE\_ERROR
  - SolverApp, [18](#)
- setVariable
  - Console, [6](#)
- SetterApp, [14](#)
  - exec, [15](#)
  - getHelp, [15](#)
  - getStatusCodeDescription, [15](#)
  - STATUS\_BAD\_ARGUMENTS, [16](#)
  - SetterApp, [15](#)
- SolverApp, [16](#)
  - exec, [17](#)
  - getHelp, [18](#)
  - getStatusCodeDescription, [18](#)
  - STATUS\_BAD\_ARGUMENTS, [18](#)
  - STATUS\_BAD\_FIELD, [18](#)
  - STATUS\_PARSE\_ERROR, [18](#)
  - SolverApp, [17](#)
- SolverApp::parse< std::complex< double > >
  - solverapp.cpp, [30](#)
- solverapp.cpp
  - OPT\_PTR, [30](#)
  - operator<<, [30](#)
  - SolverApp::parse< std::complex< double > >, [30](#)
- src/console.cpp, [25](#)
- src/getterapp.cpp, [26](#)
- src/helpapp.cpp, [26](#)
- src/main.cpp, [27](#)
- src/setterapp.cpp, [29](#)
- src/solverapp.cpp, [29](#)
- Verbosity
  - console.h, [20](#)