

My Project

Создано системой Doxygen 1.8.14

Оглавление

1	Иерархический список классов	1
1.1	Иерархия классов	1
2	Алфавитный указатель классов	3
2.1	Классы	3
3	Классы	5
3.1	Класс CBomberBuilder	5
3.1.1	Подробное описание	6
3.2	Класс CBomberPrinter	7
3.2.1	Подробное описание	8
3.3	Класс CBuilding	8
3.3.1	Подробное описание	10
3.3.2	Методы	10
3.3.2.1	addBuilding()	10
3.3.2.2	clean()	10
3.3.2.3	decreaseProtection()	11
3.3.2.4	getAttack()	11
3.3.2.5	getCosts()	11
3.3.2.6	getFreePeople()	11
3.3.2.7	getIncome()	11
3.3.2.8	heal()	12
3.3.2.9	increaseAttack()	12
3.3.2.10	increaseIncome()	12

3.3.2.11	increaseMaxProtection()	12
3.3.2.12	increasePeople()	12
3.4	Класс CBuildingFactory	13
3.4.1	Подробное описание	13
3.5	Класс CBuildingsGroup	13
3.5.1	Подробное описание	16
3.5.2	Методы	16
3.5.2.1	addBuilding()	16
3.5.2.2	decreaseProtection()	16
3.5.2.3	getFreePeople()	17
3.5.2.4	heal()	17
3.5.2.5	increaseAttack()	17
3.5.2.6	increaseIncome()	17
3.5.2.7	increaseMaxProtection()	17
3.5.2.8	increasePeople()	18
3.6	Класс CEconomyPlanet	18
3.6.1	Подробное описание	19
3.6.2	Методы	20
3.6.2.1	heal()	20
3.6.2.2	upgradeCityAttack()	20
3.6.2.3	upgradeCityIncome()	20
3.6.2.4	upgradeCityMaxProtection()	21
3.6.2.5	upgradeCityPeople()	21
3.7	Класс CFactory	21
3.7.1	Подробное описание	24
3.7.2	Методы	24
3.7.2.1	getIncome()	24
3.7.2.2	increaseIncome()	24
3.8	Класс CFactoryPrinter	24
3.8.1	Подробное описание	25

3.9	Класс CFighterBuilder	25
3.9.1	Подробное описание	26
3.10	Класс CFighterPrinter	27
3.10.1	Подробное описание	28
3.11	Класс CFortification	28
3.11.1	Подробное описание	31
3.11.2	Методы	31
3.11.2.1	getAttack()	31
3.11.2.2	getCosts()	31
3.11.2.3	getIncome()	31
3.11.2.4	increaseAttack()	32
3.12	Класс CFortificationPrinter	32
3.12.1	Подробное описание	32
3.13	Класс CHouse	33
3.13.1	Подробное описание	34
3.13.2	Методы	35
3.13.2.1	getFreePeople()	35
3.13.2.2	increasePeople()	35
3.14	Класс CHousePrinter	35
3.14.1	Подробное описание	36
3.15	Класс CInputReader	36
3.16	Класс CPlanet	37
3.17	Класс CShip	38
3.17.1	Подробное описание	40
3.17.2	Методы	40
3.17.2.1	addShip()	40
3.18	Класс CShipsGroup	40
3.18.1	Подробное описание	43
3.19	Класс CShipyards	43
3.19.1	Подробное описание	44
3.20	Класс CWarPlanet	44
3.20.1	Подробное описание	45
3.20.2	Методы	45
3.20.2.1	attackCity()	46
3.20.2.2	attackFleet()	47
3.21	Класс IPlanet	47
3.21.1	Подробное описание	49
3.22	Класс IShipBuilder	49
3.22.1	Подробное описание	50
3.23	Класс IShipPrinter	50
3.23.1	Подробное описание	51
3.24	Класс Printer	52

Алфавитный указатель	53
----------------------	----

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

CBuilding	8
CBuildingsGroup	13
CFactory	21
CFortification	28
CHouse	33
CBuildingFactory	13
CFactoryPrinter	24
CFortificationPrinter	32
CHousePrinter	35
CInputReader	36
CShip	38
CShipsGroup	40
CShipyards	43
IPlanet	47
CEconomyPlanet	18
CPlanet	37
CWarPlanet	44
IShipBuilder	49
CBomberBuilder	5
CFighterBuilder	25
IShipPrinter	50
CBomberPrinter	7
CFighterPrinter	27
Printer	52

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

CBomberBuilder	5
CBomberPrinter	7
CBuilding	
Здание	8
CBuildingFactory	
Фабрика для создания строений различного типа	13
CBuildingsGroup	
Класс описывающий район - группу зданий	13
CEconomyPlanet	18
CFactory	
Завод	21
CFactoryPrinter	
Класс необходимый для вывода объекта класса Завод в консоль	24
CFighterBuilder	25
CFighterPrinter	27
CFortification	
Укреплений	28
CFortificationPrinter	
Класс необходимый для вывода объекта класса Укрепление в консоль	32
CHouse	
Жилой дом	33
CHousePrinter	
Класс необходимый для вывода объекта класса Дом в консоль	35
CInputReader	36
CPlanet	37
CShip	38
CShipsGroup	40
CShipyards	43
CWarPlanet	44
IPlanet	47
IShipBuilder	49
IShipPrinter	50
Printer	52

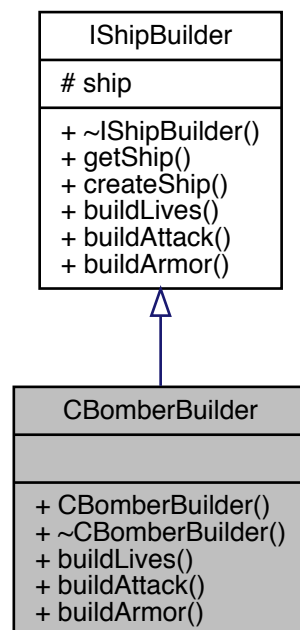
Глава 3

Классы

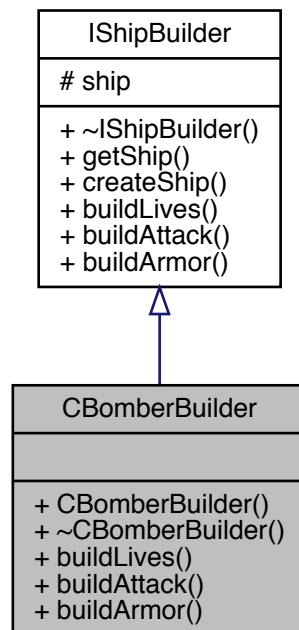
3.1 Класс CBomberBuilder

```
#include <ships.h>
```

Граф наследования:CBomberBuilder:



Граф связей класса CBomberBuilder:



Открытые члены

- `void buildLives (double lives) override`
- `void buildAttack (double attack) override`
- `void buildArmor (double armor) override`

Дополнительные унаследованные члены

3.1.1 Подробное описание

Конструктор бомбардировщика

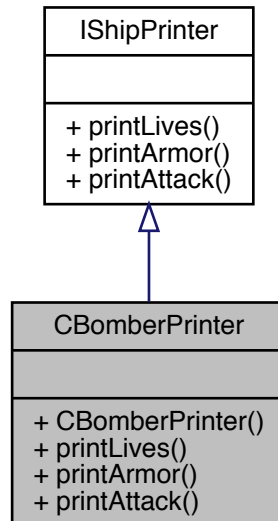
Объявления и описания членов классов находятся в файлах:

- `ships.h`
- `ships.cpp`

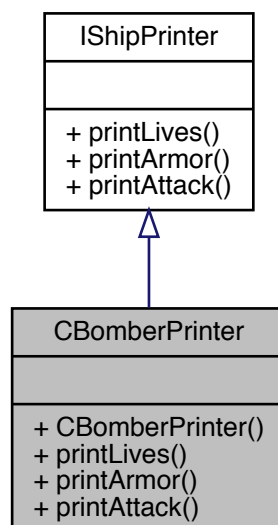
3.2 Класс CBomberPrinter

```
#include <ships.h>
```

Граф наследования:CBomberPrinter:



Граф связей класса CBomberPrinter:



Открытые члены

- void printLives (int money, int maxLives) override
- void printArmor (int money, int maxArmor) override
- void printAttack (int money, int maxAttack) override

3.2.1 Подробное описание

Класс, печатающий информацию о бомбардировщике

Объявления и описания членов классов находятся в файлах:

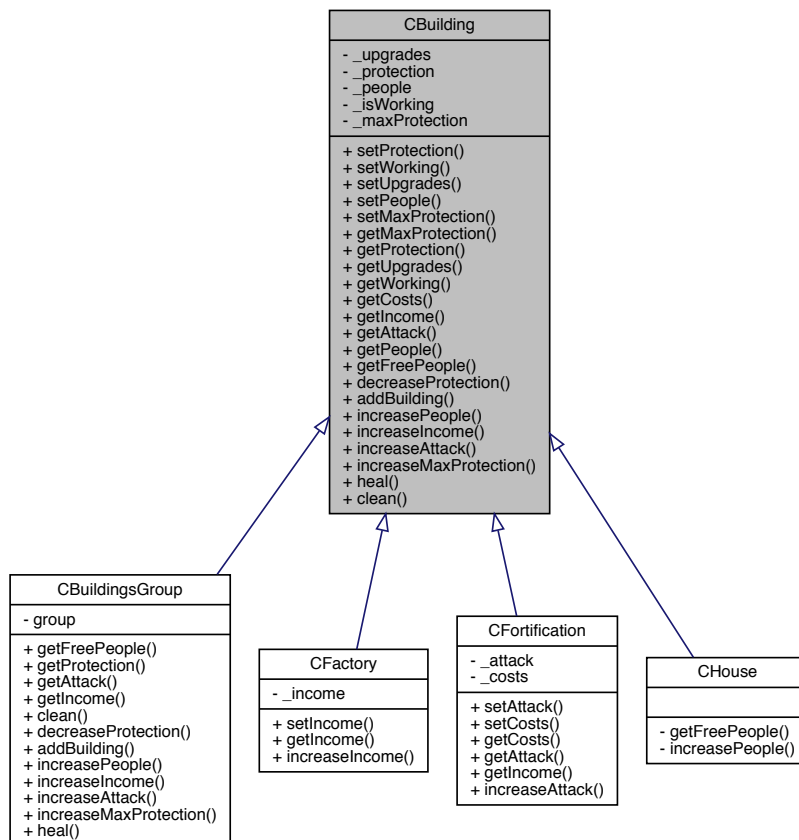
- ships.h
- ships.cpp

3.3 Класс CBuilding

Здание

```
#include <buildings.h>
```

Граф наследования: CBuilding:



Граф связей класса CBuilding:

CBuilding
- _upgrades - _protection - _people - _isWorking - _maxProtection
+ setProtection() + setWorking() + setUpgrades() + setPeople() + setMaxProtection() + getMaxProtection() + getProtection() + getUpgrades() + getWorking() + getCosts() + getIncome() + getAttack() + getPeople() + getFreePeople() + decreaseProtection() + addBuilding() + increasePeople() + increaseIncome() + increaseAttack() + increaseMaxProtection() + heal() + clean()

Открытые члены

- void [setProtection](#) (double protection)
Максимальное значение очков защиты
- void setWorking (bool isWorking)
- void setUpgrades (int upgrades)
- void setPeople (int people)
- void setMaxProtection (double protection)
- double getMaxProtection () const
- virtual double getProtection () const
- int getUpgrades () const
- bool getWorking () const
- virtual int [getCosts](#) () const
- virtual int [getIncome](#) () const
- virtual double [getAttack](#) () const
- virtual int getPeople () const
- virtual int [getFreePeople](#) () const
- virtual void [decreaseProtection](#) (double value)
- virtual void [addBuilding](#) (std::shared_ptr< [CBuilding](#) > building_ptr)
- virtual void [increasePeople](#) (int value)

- virtual void `increaseIncome` (int value)
- virtual void `increaseAttack` (double value)
- virtual void `increaseMaxProtection` (double value)
- virtual void `heal` ()
- virtual void `clean` ()

Закрытые данные

- int `_upgrades`
- double `_protection`
Количество сделанных улучшений
- int `_people`
Очки защиты
- bool `_isWorking`
Количество людей живущих или работающих здесь
- double `_maxProtection`
Эксплуатируется ли здание

3.3.1 Подробное описание

Здание

3.3.2 Методы

3.3.2.1 `addBuilding()`

```
void CBuilding::addBuilding (  
    std::shared_ptr< CBuilding > building_ptr ) [virtual]
```

Функция ничего не делает и используется только для корректной работы классов-наследников. При её вызове из этого класса программа завершается с ошибкой

Переопределяется в `CBuildingsGroup`.

3.3.2.2 `clean()`

```
void CBuilding::clean ( ) [virtual]
```

Функция используется только для корректной работы классов-наследников

Переопределяется в `CBuildingsGroup`.

3.3.2.3 decreaseProtection()

```
void CBuilding::decreaseProtection (
    double value ) [virtual]
```

Уменьшает количество очков защиты на value;

Переопределяется в [CBuildingsGroup](#).

3.3.2.4 getAttack()

```
double CBuilding::getAttack ( ) const [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяется в [CBuildingsGroup](#) и [CFortification](#).

3.3.2.5 getCosts()

```
int CBuilding::getCosts ( ) const [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяется в [CFortification](#).

3.3.2.6 getFreePeople()

```
int CBuilding::getFreePeople ( ) const [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяется в [CBuildingsGroup](#) и [CHouse](#).

3.3.2.7 getIncome()

```
int CBuilding::getIncome ( ) const [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяется в [CBuildingsGroup](#), [CFortification](#) и [CFactory](#).

3.3.2.8 heal()

```
void CBuilding::heal ( ) [virtual]
```

Делает значение `_protection` равным `_maxProtection`;

Переопределяется в [CBuildingsGroup](#).

3.3.2.9 increaseAttack()

```
void CBuilding::increaseAttack (
    double value ) [virtual]
```

Функция используется только для корректной работы классов-наследников

Переопределяется в [CBuildingsGroup](#) и [CFortification](#).

3.3.2.10 increaseIncome()

```
void CBuilding::increaseIncome (
    int value ) [virtual]
```

Функция используется только для корректной работы классов-наследников

Переопределяется в [CBuildingsGroup](#) и [CFactory](#).

3.3.2.11 increaseMaxProtection()

```
void CBuilding::increaseMaxProtection (
    double value ) [virtual]
```

Увеличивает максимально возможное количество очков защиты на `value`

Переопределяется в [CBuildingsGroup](#).

3.3.2.12 increasePeople()

```
void CBuilding::increasePeople (
    int value ) [virtual]
```

Функция используется только для корректной работы классов-наследников

Переопределяется в [CBuildingsGroup](#) и [CHouse](#).

Объявления и описания членов классов находятся в файлах:

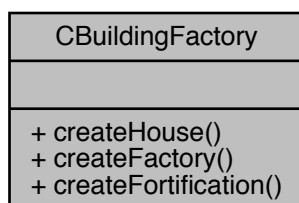
- `buildings.h`
- `buildings.cpp`

3.4 Класс CBuildingFactory

Фабрика для создания строений различного типа

```
#include <buildings.h>
```

Граф связей класса CBuildingFactory:



Открытые члены

- virtual std::shared_ptr< [CHouse](#) > createHouse ()
- virtual std::shared_ptr< [CFactory](#) > [createFactory](#) ()
Создает дом
- virtual std::shared_ptr< [CFortification](#) > [createFortification](#) ()
Создает завод

3.4.1 Подробное описание

Фабрика для создания строений различного типа

Объявления и описания членов классов находятся в файлах:

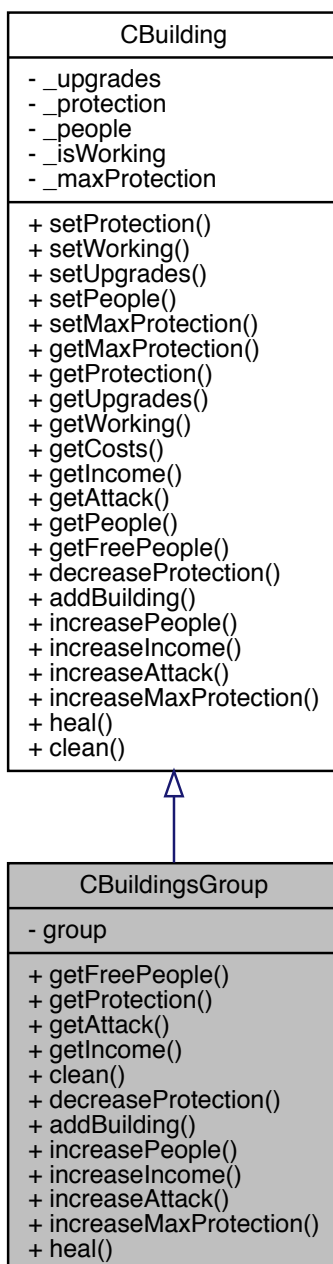
- buildings.h
- buildings.cpp

3.5 Класс CBuildingsGroup

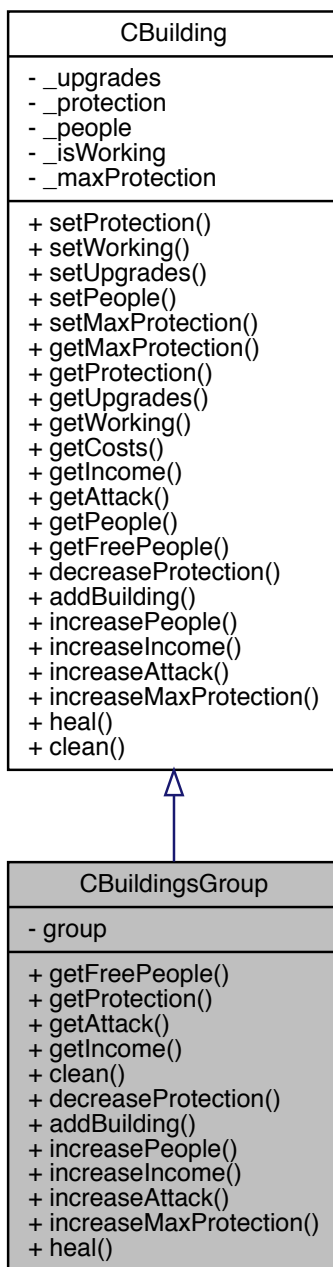
Класс описывающий район - группу зданий.

```
#include <buildings.h>
```

Граф наследования: CBuildingsGroup:



Граф связей класса CBuildingsGroup:



Открытые члены

- int [getFreePeople](#) () const override
- double [getProtection](#) () const override
Возвращает количество людей, которые нигде не работают
- double [getAttack](#) () const override
Возвращает суммарное значение очков защиты

- `int getIncome () const override`
Возвращает суммарное значение очков атаки
- `void clean () override`
Возвращает доход с региона
- `void decreaseProtection (double value) override`
Удаляет уничтоженные противником здания
- `void addBuilding (std::shared_ptr< CBuilding > building_ptr) override`
- `void increasePeople (int value) override`
- `void increaseIncome (int value) override`
- `void increaseAttack (double value) override`
- `void increaseMaxProtection (double value) override`
- `void heal () override`

Закрытые данные

- `std::vector< std::shared_ptr< CBuilding > > group`

3.5.1 Подробное описание

Класс описывающий район - группу зданий.

Тут применяется паттерн Компоновщик;

3.5.2 Методы

3.5.2.1 `addBuilding()`

```
void CBuildingsGroup::addBuilding (
    std::shared_ptr< CBuilding > building_ptr ) [override], [virtual]
```

Функция ничего не делает и используется только для корректной работы классов-наследников. При её вызове из этого класса программа завершается с ошибкой

Переопределяет метод предка [CBuilding](#).

3.5.2.2 `decreaseProtection()`

```
void CBuildingsGroup::decreaseProtection (
    double value ) [override], [virtual]
```

Удаляет уничтоженные противником здания

Уменьшает суммарное значение очков защиты на value

Переопределяет метод предка [CBuilding](#).

3.5.2.3 getFreePeople()

```
int CBuildingsGroup::getFreePeople ( ) const [override], [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяет метод предка [CBuilding](#).

3.5.2.4 heal()

```
void CBuildingsGroup::heal ( ) [override], [virtual]
```

Делает значение `_protection` равным `_maxProtection`;

Переопределяет метод предка [CBuilding](#).

3.5.2.5 increaseAttack()

```
void CBuildingsGroup::increaseAttack (
    double value ) [override], [virtual]
```

Увеличивает очки каждого укрепления района на `value`

Переопределяет метод предка [CBuilding](#).

3.5.2.6 increaseIncome()

```
void CBuildingsGroup::increaseIncome (
    int value ) [override], [virtual]
```

Увеличивает доход с каждого завода в районе на `value`

Переопределяет метод предка [CBuilding](#).

3.5.2.7 increaseMaxProtection()

```
void CBuildingsGroup::increaseMaxProtection (
    double value ) [override], [virtual]
```

Увеличивает максимально-возможное число очков защиты на `value` для каждого здания района

Переопределяет метод предка [CBuilding](#).

3.5.2.8 increasePeople()

```
void CBuildingsGroup::increasePeople (
    int value )    [override], [virtual]
```

Увеличивает число людей, живущих в каждом жилом доме района на value

Переопределяет метод предка [CBuilding](#).

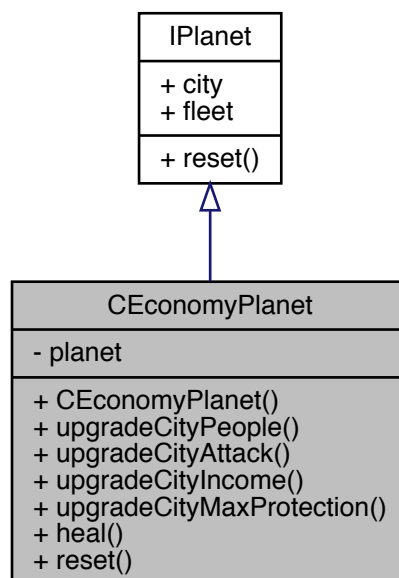
Объявления и описания членов классов находятся в файлах:

- buildings.h
- buildings.cpp

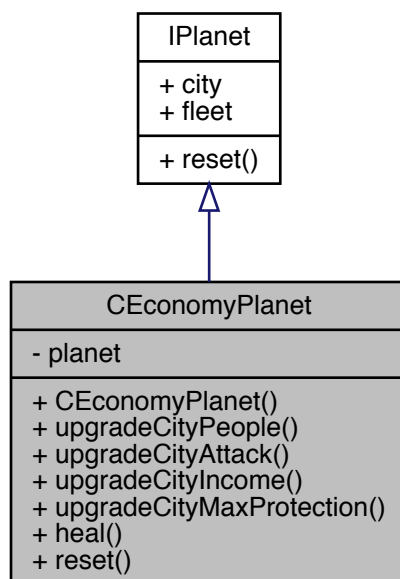
3.6 Класс CEconomyPlanet

```
#include <planet.h>
```

Граф наследования:CEconomyPlanet:



Граф связей класса CEconomyPlanet:



Открытые члены

- `CEconomyPlanet` (`std::shared_ptr< IPlanet > planet_ptr`)
Указатель на планету
- `void upgradeCityPeople` (`int regionNum, int points`)
- `void upgradeCityAttack` (`int regionNum, int points`)
- `void upgradeCityIncome` (`int regionNum, int points`)
- `void upgradeCityMaxProtection` (`int regionNum, int points`)
- `void heal` (`int regionNum`)
- `void reset` () `override`
Vector эскадрилий

Закрытые данные

- `std::shared_ptr< IPlanet > planet`

Дополнительные унаследованные члены

3.6.1 Подробное описание

Экономическая сущность планеты

3.6.2 Методы

3.6.2.1 heal()

```
void CEconomyPlanet::heal (  
    int regionNum )
```

Восстанавливает все еще не уничтоженные здания в регионе

Аргументы

regionNum	номер региона
-----------	---------------

3.6.2.2 upgradeCityAttack()

```
void CEconomyPlanet::upgradeCityAttack (  
    int regionNum,  
    int points )
```

Увеличивает количество очков атаки для данного региона

Аргументы

regionNum	номер региона
points	- на сколько мощнее становится укрепление

3.6.2.3 upgradeCityIncome()

```
void CEconomyPlanet::upgradeCityIncome (  
    int regionNum,  
    int points )
```

Увеличивает доход данного региона

Аргументы

regionNum	номер региона
points	- на сколько больше галактионов приносит один завод

3.6.2.4 upgradeCityMaxProtection()

```
void CEconomyPlanet::upgradeCityMaxProtection (
    int regionNum,
    int points )
```

Увеличивает максимально-возможное количество очков защиты для зданий в данном регионе

Аргументы

regionNum	номер региона
points	- на сколько увеличивается это количество

3.6.2.5 upgradeCityPeople()

```
void CEconomyPlanet::upgradeCityPeople (
    int regionNum,
    int points )
```

Увеличивает количество людей, проживающих в регионе

Аргументы

regionNum	номер региона
points	- на сколько людей становится больше в каждом доме

Объявления и описания членов классов находятся в файлах:

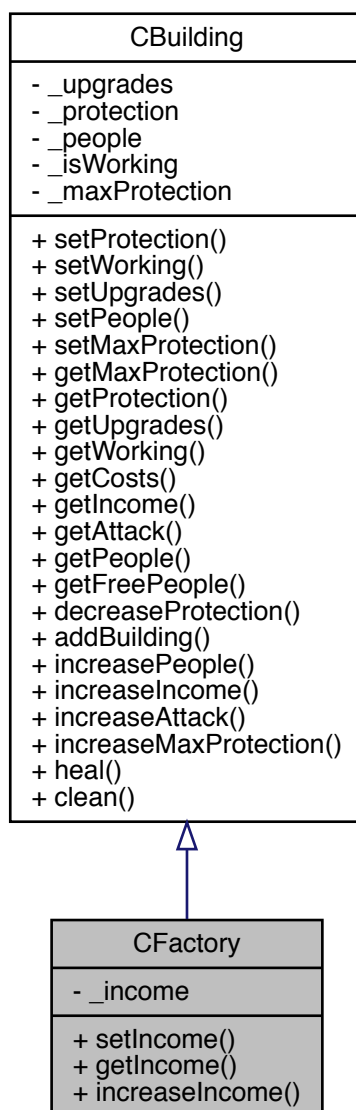
- planet.h
- planet.cpp

3.7 Класс CFactory

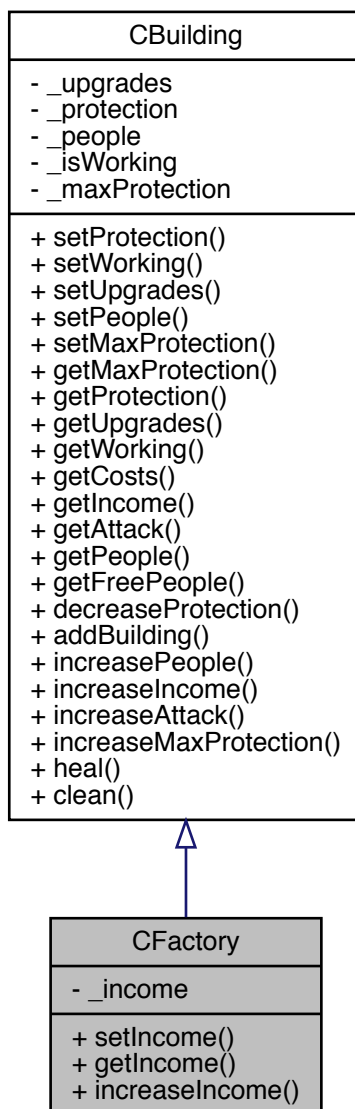
Завод

```
#include <buildings.h>
```

Граф наследования: CFactory:



Граф связей класса CFactory:



Открытые члены

- void `setIncome` (int income)
Доход, получаемый с завода
- int `getIncome` () const override
- void `increaseIncome` (int value) override

Закрытые данные

- int `_income`

3.7.1 Подробное описание

Завод

3.7.2 Методы

3.7.2.1 `getIncome()`

```
int CFactory::getIncome ( ) const [override], [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяет метод предка [CBuilding](#).

3.7.2.2 `increaseIncome()`

```
void CFactory::increaseIncome (
    int value ) [override], [virtual]
```

Функция используется только для корректной работы классов-наследников

Переопределяет метод предка [CBuilding](#).

Объявления и описания членов классов находятся в файлах:

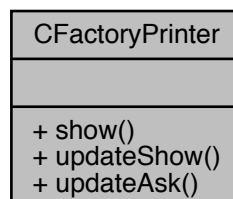
- `buildings.h`
- `buildings.cpp`

3.8 Класс `CFactoryPrinter`

Класс необходимый для вывода объекта класса Завод в консоль

```
#include <buildings.h>
```

Граф связей класса `CFactoryPrinter`:



Открытые статические члены

- static void show (CFactory const &factory)
- static void updateShow (CFactory const &factory)
Выводит основную информацию
- static void updateAsk (CFactory const &factory)
Выводит информацию по возможным улучшениям

3.8.1 Подробное описание

Класс необходимый для вывода объекта класса Завод в консоль

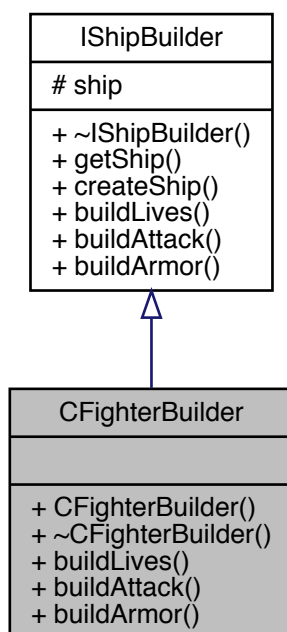
Объявления и описания членов классов находятся в файлах:

- buildings.h
- buildings.cpp

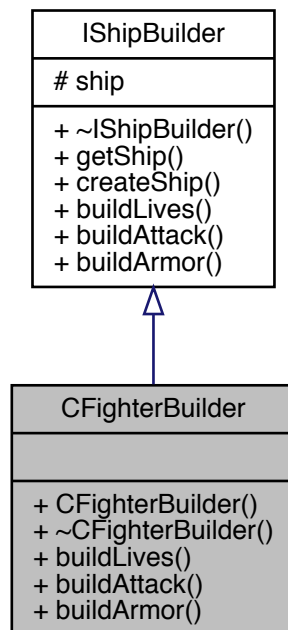
3.9 Класс CFighterBuilder

```
#include <ships.h>
```

Граф наследования:CFighterBuilder:



Граф связей класса CFighterBuilder:



Открытые члены

- void buildLives (double lives) override
- void buildAttack (double attack) override
- void buildArmor (double armor) override

Дополнительные унаследованные члены

3.9.1 Подробное описание

Конструктор истребителя

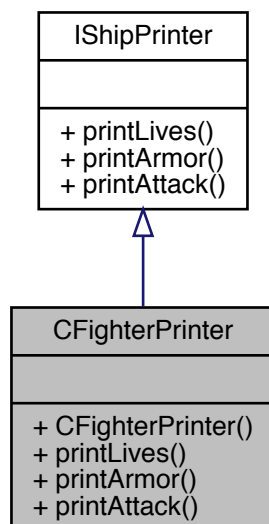
Объявления и описания членов классов находятся в файлах:

- ships.h
- ships.cpp

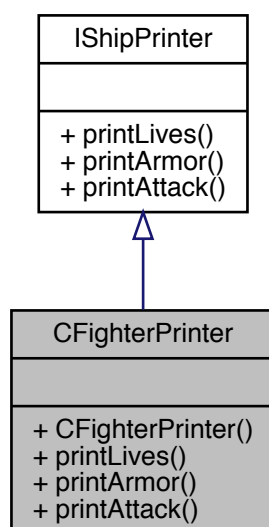
3.10 Класс CFighterPrinter

```
#include <ships.h>
```

Граф наследования:CFighterPrinter:



Граф связей класса CFighterPrinter:



Открытые члены

- void printLives (int money, int maxLives) override
- void printArmor (int money, int maxArmor) override
- void printAttack (int money, int maxAttack) override

3.10.1 Подробное описание

Класс, печатающий информацию об истребителе

Объявления и описания членов классов находятся в файлах:

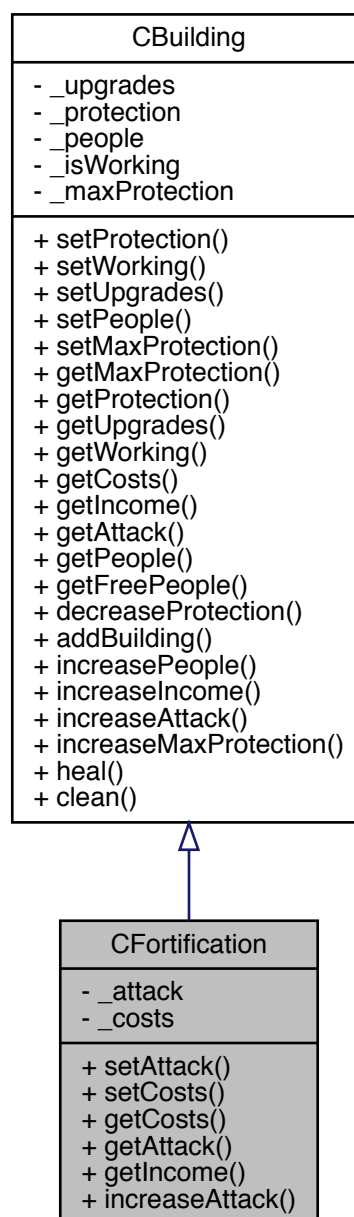
- ships.h
- ships.cpp

3.11 Класс CFortification

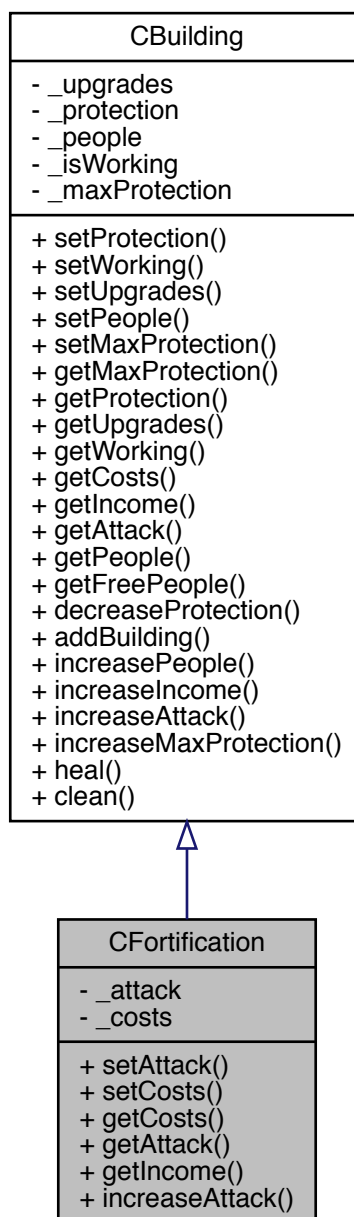
Укреплений

```
#include <buildings.h>
```

Граф наследования: CFortification:



Граф связей класса CFortification:



Открытые члены

- void [setAttack](#) (double attack)
Расходы на содержание
- void [setCosts](#) (int costs)
- int [getCosts](#) () const override
- double [getAttack](#) () const override
- int [getIncome](#) () const override
- void [increaseAttack](#) (double value) override

Закрытые данные

- double `_attack`
- int `_costs`
Очки атаки

3.11.1 Подробное описание

Укреплений

3.11.2 Методы

3.11.2.1 `getAttack()`

```
double CFortification::getAttack ( ) const [override], [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяет метод предка [CBuilding](#).

3.11.2.2 `getCosts()`

```
int CFortification::getCosts ( ) const [override], [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяет метод предка [CBuilding](#).

3.11.2.3 `getIncome()`

```
int CFortification::getIncome ( ) const [override], [virtual]
```

Функция возвращает 0 и используется только для корректной работы классов-наследников

Переопределяет метод предка [CBuilding](#).

3.11.2.4 increaseAttack()

```
void CFortification::increaseAttack (
    double value ) [override], [virtual]
```

Функция используется только для корректной работы классов-наследников

Переопределяет метод предка [CBuilding](#).

Объявления и описания членов классов находятся в файлах:

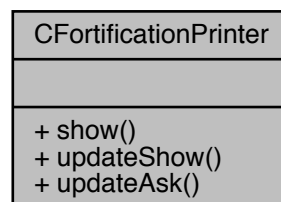
- buildings.h
- buildings.cpp

3.12 Класс CFortificationPrinter

Класс необходимый для вывода объекта класса Укрепление в консоль

```
#include <buildings.h>
```

Граф связей класса CFortificationPrinter:



Открытые статические члены

- static void show ([CFortification](#) const &fort)
- static void [updateShow](#) ([CFortification](#) const &fort)
Выводит основную информацию
- static void [updateAsk](#) ([CFortification](#) const &fort)
Выводит информацию по возможным улучшениям

3.12.1 Подробное описание

Класс необходимый для вывода объекта класса Укрепление в консоль

Объявления и описания членов классов находятся в файлах:

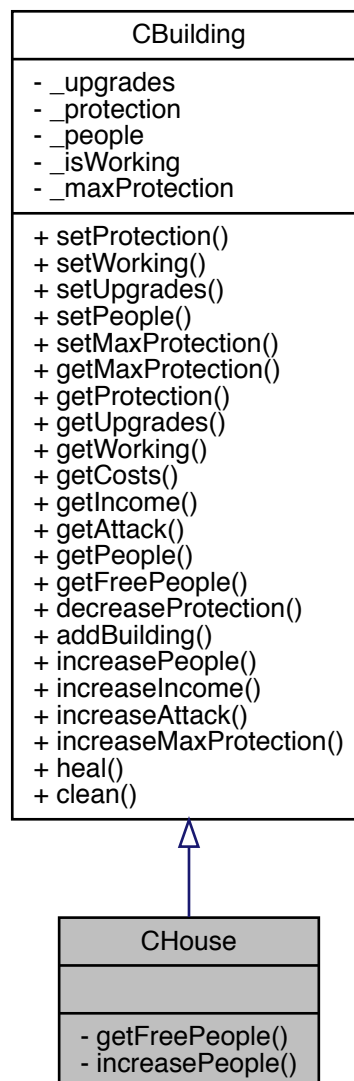
- buildings.h
- buildings.cpp

3.13 Класс CHouse

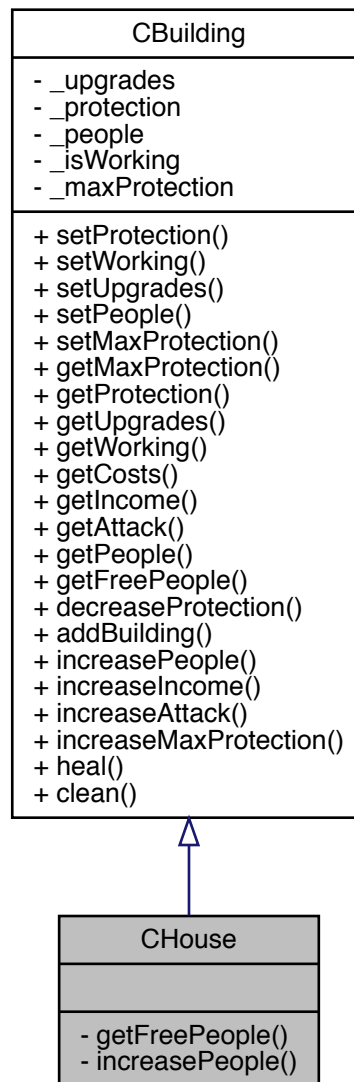
Жилой дом

```
#include <buildings.h>
```

Граф наследования:CHouse:



Граф связей класса CHouse:



Закрытые члены

- virtual int [getFreePeople](#) () const override
- virtual void [increasePeople](#) (int value) override

Дополнительные унаследованные члены

3.13.1 Подробное описание

Жилой дом

3.13.2 Методы

3.13.2.1 getFreePeople()

```
int CHouse::getFreePeople ( ) const [override], [private], [virtual]
```

Возвращает число людей живущих в доме

Переопределяет метод предка [CBuilding](#).

3.13.2.2 increasePeople()

```
void CHouse::increasePeople (
    int value ) [override], [private], [virtual]
```

Увеличивает число людей живущих в доме на value

Переопределяет метод предка [CBuilding](#).

Объявления и описания членов классов находятся в файлах:

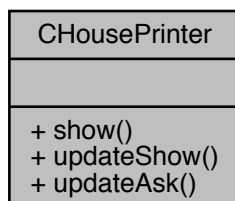
- buildings.h
- buildings.cpp

3.14 Класс CHousePrinter

Класс необходимый для вывода объекта класса Дом в консоль

```
#include <buildings.h>
```

Граф связей класса CHousePrinter:



Открытые статические члены

- static void show (CHouse const &house)
- static void updateShow (CHouse const &house)
Выводит основную информацию
- static void updateAsk (CHouse const &house)
Выводит информацию по возможным улучшениям

3.14.1 Подробное описание

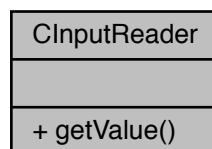
Класс необходимый для вывода объекта класса Дом в консоль

Объявления и описания членов классов находятся в файлах:

- buildings.h
- buildings.cpp

3.15 Класс CInputReader

Граф связей класса CInputReader:



Открытые статические члены

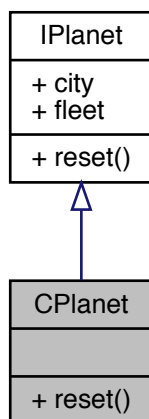
- static int getValue (int minValue, int maxValue)

Объявления и описания членов классов находятся в файлах:

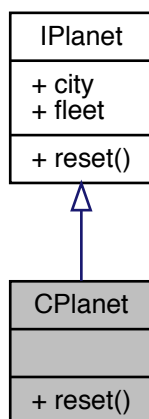
- ships.h
- ships.cpp

3.16 Класс CPlanet

Граф наследования:CPlanet:



Граф связей класса CPlanet:



Открытые члены

- virtual void `reset` () override
Vector эскадрилий

Дополнительные унаследованные члены

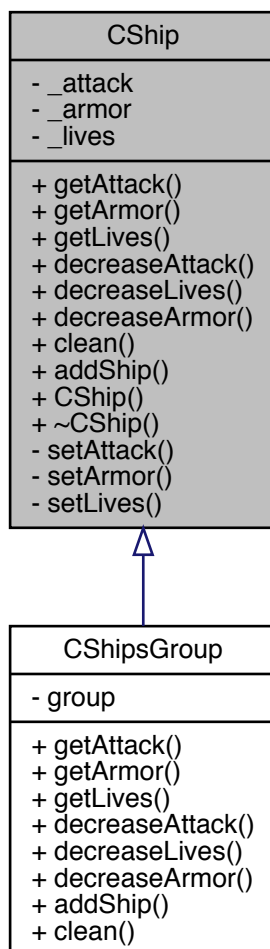
Объявления и описания членов классов находятся в файлах:

- planet.h
- planet.cpp

3.17 Класс CShip

```
#include <ships.h>
```

Граф наследования:CShip:



Граф связей класса CShip:

CShip
- _attack - _armor - _lives
+ getAttack() + getArmor() + getLives() + decreaseAttack() + decreaseLives() + decreaseArmor() + clean() + addShip() + CShip() + ~CShip() - setAttack() - setArmor() - setLives()

Открытые члены

- virtual double getAttack () const
- virtual double getArmor () const
- virtual double getLives () const
- virtual void decreaseAttack (double value)
- virtual void [decreaseLives](#) (double value)
Уменьшает очки атаки на value.
- virtual void [decreaseArmor](#) (double value)
Уменьшает жизни на value.
- virtual void [clean](#) ()
Уменьшает броню на value.
- virtual void [addShip](#) (std::shared_ptr< [CShip](#) >)
Ничего не делает, нужен для работы классов-наследников

Закрытые члены

- void [setAttack](#) (double attack)
Жизни
- void setArmor (double armor)
- void setLives (double lives)

Закрытые данные

- double _attack
Атака
- double [_armor](#)
Броня
- double [_lives](#)

Друзья

- class CFighterBuilder
- class CBomberBuilder

3.17.1 Подробное описание

Космический корбаль

3.17.2 Методы

3.17.2.1 addShip()

```
void CShip::addShip (  
    std::shared_ptr< CShip > ) [virtual]
```

Ничего не делает, нужен для работы классов-наследников

Ничего не делает, нужен для работы классов-наследников. В случае вызова Программа завершается с ошибкой.

Переопределяется в [CShipsGroup](#).

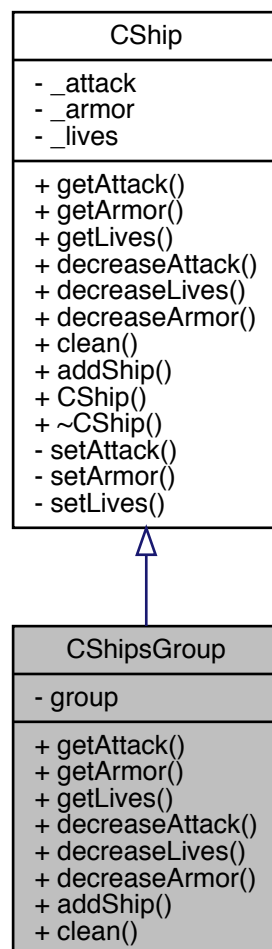
Объявления и описания членов классов находятся в файлах:

- ships.h
- ships.cpp

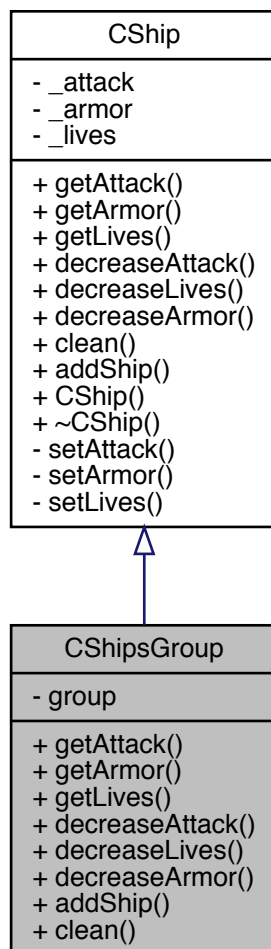
3.18 Класс CShipsGroup

```
#include <ships.h>
```

Граф наследования:CShipsGroup:



Граф связей класса CShipsGroup:



Открытые члены

- `double getAttack () const override`
- `double getArmor () const override`
Возвращает суммарное значение атаки
- `double getLives () const override`
Возвращает суммарное значение брони
- `void decreaseAttack (double value) override`
Возвращает суммарное значение жизней
- `void decreaseLives (double value) override`
Уменьшает суммарное значение атаки на value.
- `void decreaseArmor (double value) override`
Уменьшает суммарное значение жизни на value.
- `void addShip (std::shared_ptr< CShip > ship_ptr) override`
Уменьшает суммарное значение брони на value.
- `void clean () override`
Добавляет новый корабль

Закрытые данные

- `std::vector< std::shared_ptr< CShip > > group`

3.18.1 Подробное описание

Класс описывающий эскадрилью

Объявления и описания членов классов находятся в файлах:

- `ships.h`
- `ships.cpp`

3.19 Класс CShipyards

```
#include <ships.h>
```

Граф связей класса CShipyards:

CShipyards
- shipBuilder - shipPrinter
+ CShipyards() + ~CShipyards() + setShipBuilder() + setShipPrinter() + getShip() + constructShip()

Открытые члены

- `void setShipBuilder (std::shared_ptr< IShipBuilder > builder)`
- `void setShipPrinter (std::shared_ptr< IShipPrinter > printer)`
- `std::shared_ptr< CShip > getShip ()`
- `int constructShip (int money, int maxLives, int maxArmor, int maxAttack, int livesPrice, int armorPrice, int attackPrice)`

Закрытые данные

- `std::shared_ptr< IShipBuilder > shipBuilder`
- `std::shared_ptr< IShipPrinter > shipPrinter`

3.19.1 Подробное описание

Директор, который использует переданного уме конструктора для создания нужного космического корабля

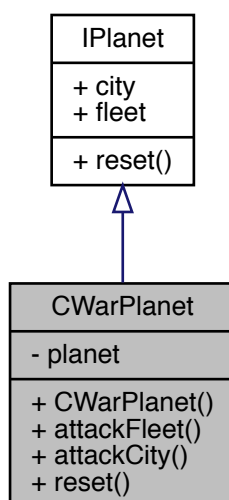
Объявления и описания членов классов находятся в файлах:

- ships.h
- ships.cpp

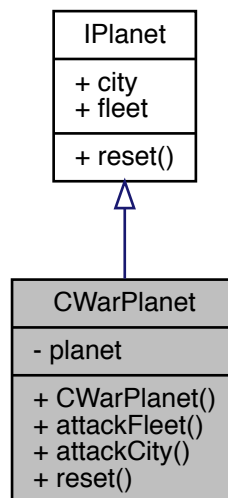
3.20 Класс CWarPlanet

```
#include <planet.h>
```

Граф наследования: CWarPlanet:



Граф связей класса CWarPlanet:



Открытые члены

- `CWarPlanet` (`std::shared_ptr< IPlanet > planet_ptr`)
Указатель на планету
- `void attackFleet` (`IPlanet &otherPlanet`, `int mySquadronNum`, `int otherSquadronNum`)
- `void attackCity` (`IPlanet &otherPlanet`, `int mySquadronNum`, `int regionNum`)
- `void reset` () override
Vector эскадрилий

Закрытые данные

- `std::shared_ptr< IPlanet > planet`

Дополнительные унаследованные члены

3.20.1 Подробное описание

Военная сущность планеты

3.20.2 Методы

3.20.2.1 attackCity()

```
void CWarPlanet::attackCity (  
    IPlanet & otherPlanet,  
    int mySquadronNum,  
    int regionNum )
```

Атакует город противника

Аргументы

otherPlanet	ссылка на атакуемую планету
mySquadronNum	номер эскадрильи, которая наносит удар
regionNum	номер района, по которому наносят удар

3.20.2.2 attackFleet()

```
void CWarPlanet::attackFleet (  
    IPланet & otherPlanet,  
    int mySquadronNum,  
    int otherSquadronNum )
```

Атакует флот противника

Аргументы

otherPlanet	ссылка на атакуемую планету
mySquadronNum	номер эскадрильи, которая наносит удар
otherSquadronNum	номер эскадрильи, по которой наносят удар

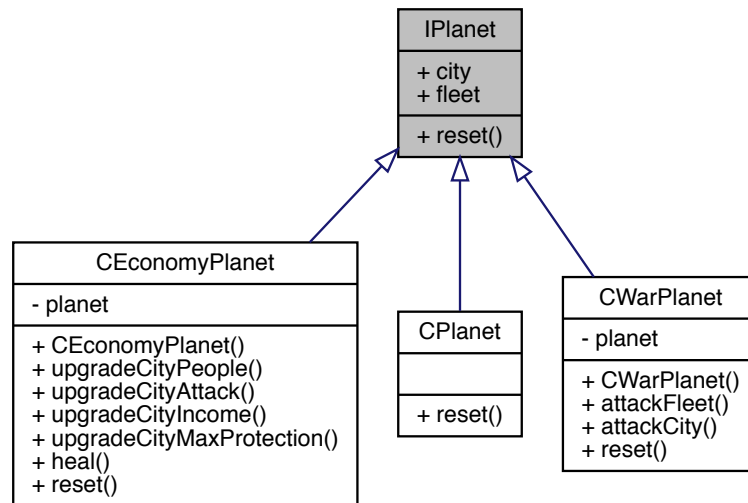
Объявления и описания членов классов находятся в файлах:

- planet.h
- planet.cpp

3.21 Класс IPланet

```
#include <planet.h>
```

Граф наследования: IPlanet:



Граф связей класса IPlanet:



Открытые члены

- virtual void `reset` ()=0
Vector эскадрилий

Открытые атрибуты

- std::vector< `CBuildingsGroup` > `city`
- std::vector< `CShipsGroup` > `fleet`
Vector районов города

Друзья

- class CWarPlanet
- class CEconomyPlanet

3.21.1 Подробное описание

Интерфейс необходимый для использования паттерна Декоратор

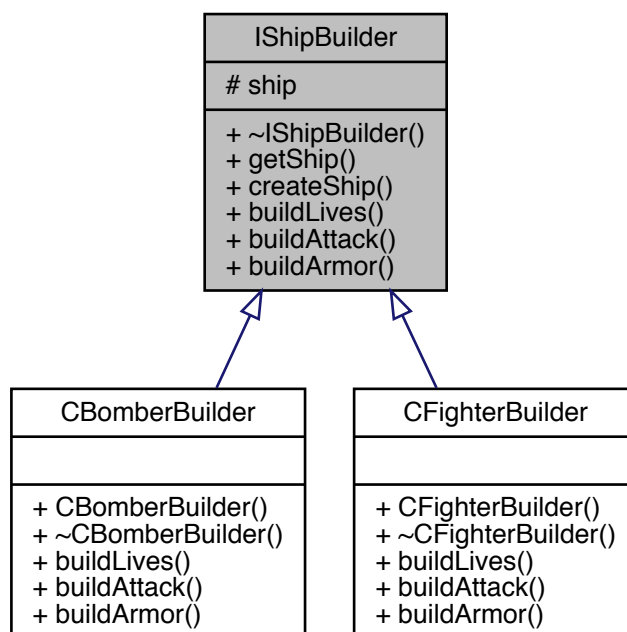
Объявления и описания членов класса находятся в файле:

- planet.h

3.22 Класс IShipBuilder

```
#include <ships.h>
```

Граф наследования:IShipBuilder:



Граф связей класса IShipBuilder:

IShipBuilder
ship
+ ~IShipBuilder() + getShip() + createShip() + buildLives() + buildAttack() + buildArmor()

Открытые члены

- `std::shared_ptr< CShip > getShip ()`
- `void createShip ()`
- `virtual void buildLives (double lives)=0`
- `virtual void buildAttack (double attack)=0`
- `virtual void buildArmor (double armor)=0`

Защищенные данные

- `std::shared_ptr< CShip > ship`

3.22.1 Подробное описание

Интерфейс для конструктора корабля

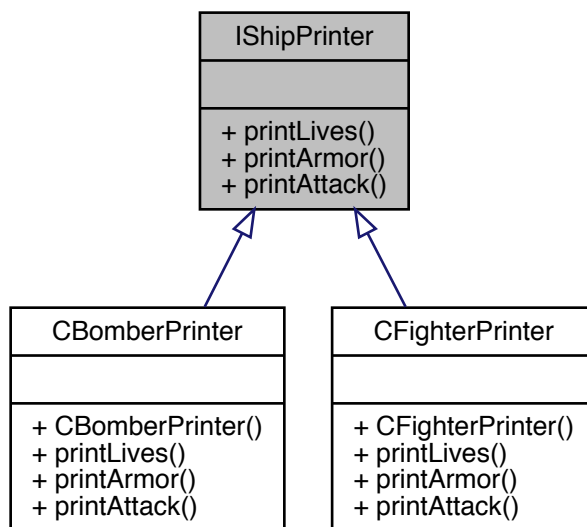
Объявления и описания членов классов находятся в файлах:

- `ships.h`
- `ships.cpp`

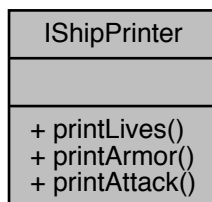
3.23 Класс IShipPrinter

```
#include <ships.h>
```


Граф наследования: IShipPrinter:



Граф связей класса IShipPrinter:



Открытые члены

- virtual void printLives (int money, int maxLives)=0
- virtual void printArmor (int money, int maxArmor)=0
- virtual void printAttack (int money, int maxAttack)=0

3.23.1 Подробное описание

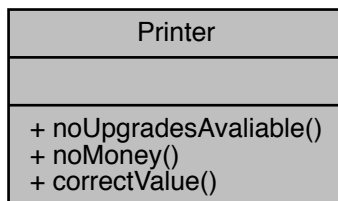
Интерфейс для класса, выводящего информацию о классе [CShip](#) в консоль

Объявления и описания членов класса находятся в файле:

- ships.h

3.24 Класс Printer

Граф связей класса Printer:



Открытые статические члены

- static void noUpgradesAvaliable ()
- static void noMoney ()
- static void correctValue ()

Объявления и описания членов класса находятся в файле:

- game.h

Предметный указатель

- addBuilding
 - CBuilding, 10
 - CBuildingsGroup, 16
- addShip
 - CShip, 40
- attackCity
 - CWarPlanet, 45
- attackFleet
 - CWarPlanet, 47
- CBomberBuilder, 5
- CBomberPrinter, 7
- CBuilding, 8
 - addBuilding, 10
 - clean, 10
 - decreaseProtection, 10
 - getAttack, 11
 - getCosts, 11
 - getFreePeople, 11
 - getIncome, 11
 - heal, 11
 - increaseAttack, 12
 - increaseIncome, 12
 - increaseMaxProtection, 12
 - increasePeople, 12
- CBuildingFactory, 13
- CBuildingsGroup, 13
 - addBuilding, 16
 - decreaseProtection, 16
 - getFreePeople, 16
 - heal, 17
 - increaseAttack, 17
 - increaseIncome, 17
 - increaseMaxProtection, 17
 - increasePeople, 17
- CEconomyPlanet, 18
 - heal, 20
 - upgradeCityAttack, 20
 - upgradeCityIncome, 20
 - upgradeCityMaxProtection, 20
 - upgradeCityPeople, 21
- CFactory, 21
 - getIncome, 24
 - increaseIncome, 24
- CFactoryPrinter, 24
- CFighterBuilder, 25
- CFighterPrinter, 27
- CFortification, 28
 - getAttack, 31
 - getCosts, 31
 - getIncome, 31
 - increaseAttack, 31
- CFortificationPrinter, 32
- CHouse, 33
 - getFreePeople, 35
 - increasePeople, 35
- CHousePrinter, 35
- CInputReader, 36
- CPlanet, 37
- CShip, 38
 - addShip, 40
- CShipsGroup, 40
- CShipyards, 43
- CWarPlanet, 44
 - attackCity, 45
 - attackFleet, 47
- clean
 - CBuilding, 10
- decreaseProtection
 - CBuilding, 10
 - CBuildingsGroup, 16
- getAttack
 - CBuilding, 11
 - CFortification, 31
- getCosts
 - CBuilding, 11
 - CFortification, 31
- getFreePeople
 - CBuilding, 11
 - CBuildingsGroup, 16
 - CHouse, 35
- getIncome
 - CBuilding, 11
 - CFactory, 24
 - CFortification, 31
- heal
 - CBuilding, 11
 - CBuildingsGroup, 17
 - CEconomyPlanet, 20
- IPlanet, 47
- IShipBuilder, 49
- IShipPrinter, 50
- increaseAttack
 - CBuilding, 12
 - CBuildingsGroup, 17

- CFortification, [31](#)
- increaseIncome
 - CBuilding, [12](#)
 - CBuildingsGroup, [17](#)
 - CFactory, [24](#)
- increaseMaxProtection
 - CBuilding, [12](#)
 - CBuildingsGroup, [17](#)
- increasePeople
 - CBuilding, [12](#)
 - CBuildingsGroup, [17](#)
 - CHouse, [35](#)
- Printer, [52](#)
- upgradeCityAttack
 - CEconomyPlanet, [20](#)
- upgradeCityIncome
 - CEconomyPlanet, [20](#)
- upgradeCityMaxProtection
 - CEconomyPlanet, [20](#)
- upgradeCityPeople
 - CEconomyPlanet, [21](#)