Gra_RPG

1.0

Wygenerowano przez Doxygen 1.8.20

Rozdział 1

Indeks hierarchiczny

1.1 Hierarchia klas

Ta lista dziedziczenia posortowana jest z grubsza, choć nie całkowicie, alfabetycznie:

Rozdział 2

Indeks klas

2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

Rozdział 3

Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

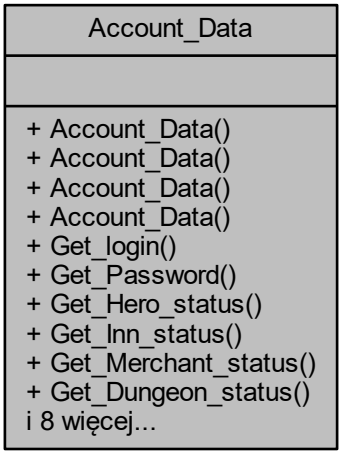
Rozdział 4

Dokumentacja klas

4.1 Dokumentacja klasy Account_Data

`#include <Account_Data.h>`

Diagram współpracy dla Account_Data:



Metody publiczne

- Account_Data ()
- Account_Data (std::string &v_Login, std::string &v_Password)
- Account_Data (std::string &v_Login, std::string &v_Password, std::shared_ptr< Hero > &v_Hero_status, std::shared_ptr< Options > &v_Settings)
- Account_Data (std::string v_Login, std::string v_Password, std::shared_ptr< Options > v_Settings, std↩ ::shared_ptr< Hero > v_Hero_status, std::shared_ptr< Inn > v_Inn_status, std::shared_ptr< Merchant > v_Merchant_status, std::shared_ptr< Dungeon > v_Dungeon_status, bool v_lost)

- std::string Get_login ()
- std::string Get_Password ()
- std::shared_ptr< Hero > Get_Hero_status ()
- std::shared_ptr< Inn > Get_Inn_status ()
- std::shared_ptr< Merchant > Get_Merchant_status ()
- std::shared_ptr< Dungeon > Get_Dungeon_status ()
- std::shared_ptr< Options > Get_Settings ()
- bool Get_Lost ()
- void Set_Settings (Options v_Settings)
- void Set_Inn_status (std::shared_ptr< Inn > v_Inn_status)
- void Set_Hero_status (std::shared_ptr< Hero > v_Hero_status)
- void Set_Merchant_status (std::shared_ptr< Merchant > v_Merchant_status)
- void Set_Dungeon_status (std::shared_ptr< Dungeon > v_Dungeon_status)
- void Lost ()

4.1.1 Opis szczegółowy

Klasa Danych_Użytkowników

4.1.2 Dokumentacja konstruktora i destruktora

4.1.2.1 Account_Data() [1/4]

```
Account_Data::Account_Data ( )
```

Konstruktor domyślny

4.1.2.2 Account_Data() [2/4]

```
Account_Data::Account_Data (
            std::string & v_Login,
            std::string & v_Password )
```

Konstruktor dwu-argumentowy.

4.1.2.3 Account_Data() [3/4]

```
Account_Data::Account_Data (
            std::string & v_Login,
            std::string & v_Password,
            std::shared_ptr< Hero > & v_Hero_status,
            std::shared_ptr< Options > & v_Settings )
```

Konstruktor wielo-argumentowy, tworzący nowy obiekt danych użytkownika.

4.1.2.4 Account_Data() [4/4]

```
Account_Data::Account_Data (
            std::string v_Login,
            std::string v_Password,
            std::shared_ptr< Options > v_Settings,
            std::shared_ptr< Hero > v_Hero_status,
            std::shared_ptr< Inn > v_Inn_status,
            std::shared_ptr< Merchant > v_Merchant_status,
            std::shared_ptr< Dungeon > v_Dungeon_status,
            bool v_lost )
```

Konstruktor wielo-argumentowy, tworzący konkretny obiekt danych użytkownika.

4.1.3 Dokumentacja funkcji składowych

4.1.3.1 Get_Dungeon_status()

```
std::shared_ptr< Dungeon > Account_Data::Get_Dungeon_status ( )
```

Funkcja zwracająca wskaźnik na loch. (shared_ptr<Dungeon>)

4.1.3.2 Get_Hero_status()

```
std::shared_ptr< Hero > Account_Data::Get_Hero_status ( )
```

Funkcja zwracająca wskaźnik na bohatera. (shared_ptr<Hero>)

4.1.3.3 Get_Inn_status()

```
std::shared_ptr< Inn > Account_Data::Get_Inn_status ( )
```

Funkcja zwracająca wskaźnik na karczme. (shared_ptr<Inn>)

4.1.3.4 Get_login()

```
std::string Account_Data::Get_login ( )
```

Funkcja zwracająca login. (string)

4.1.3.5 Get_Lost()

```
bool Account_Data::Get_Lost ( )
```

Funkcja zwracająca informacje o polu lost. (bool)

4.1.3.6 Get_Merchant_status()

```
std::shared_ptr< Merchant > Account_Data::Get_Merchant_status ( )
```

Funkcja zwracająca wskaźnik na kupca. (shared_ptr<Merchant>)

4.1.3.7 Get_Password()

```
std::string Account_Data::Get_Password ( )
```

Funkcja zwracająca hasło. (string)

4.1.3.8 Get_Settings()

```
std::shared_ptr< Options > Account_Data::Get_Settings ( )
```

Funkcja zwracająca wskaźnik na ustawienia. (shared_ptr<Options>)

4.1.3.9 Lost()

```
void Account_Data::Lost ( )
```

Funkcja ustawiająca pole lost na true, porażka w trybe Hardcore.

4.1.3.10 Set_Dungeon_status()

```
void Account_Data::Set_Dungeon_status (
            std::shared_ptr< Dungeon > v_Dungeon_status )
```

Funkcja ustawiająca loch.

4.1.3.11 Set_Hero_status()

```
void Account_Data::Set_Hero_status (
            std::shared_ptr< Hero > v_Hero_status )
```

Funkcja ustawiająca bohatera.

4.1.3.12 Set_Inn_status()

```
void Account_Data::Set_Inn_status (
            std::shared_ptr< Inn > v_Inn_status )
```

Funkcja ustawiająca karczme.

4.1.3.13 Set_Merchant_status()

```
void Account_Data::Set_Merchant_status (
            std::shared_ptr< Merchant > v_Merchant_status )
```

Funkcja ustawiająca kupca.

4.1.3.14 Set_Settings()

```
void Account_Data::Set_Settings (
            Options v_Settings )
```

Funkcja ustawiająca ustawienia.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Account_Data.h
- Account_Data.cpp

4.2 Dokumentacja klasy Armor

```
#include <Armor.h>
```

Diagram dziedziczenia dla Armor

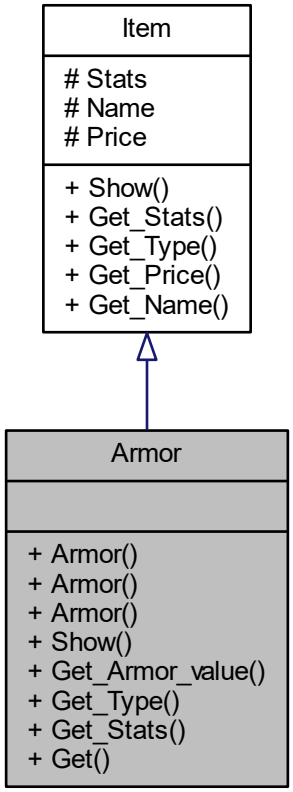
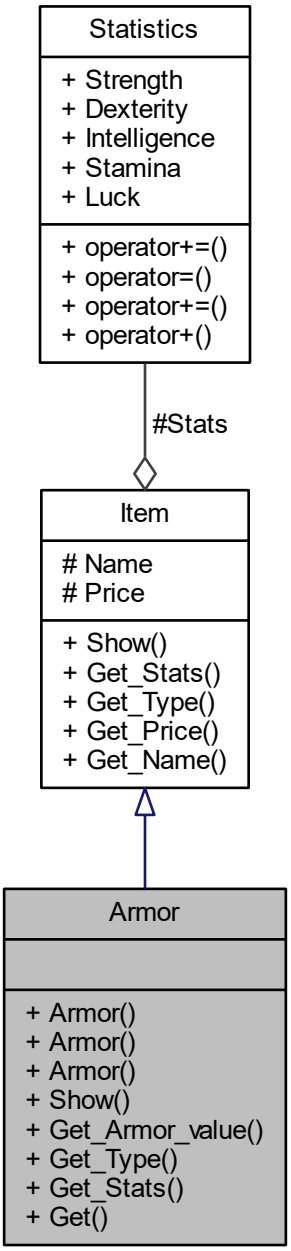


Diagram współpracy dla Armor:



Metody publiczne

- Armor (std::string v_Type, std::string &v_Profession)
- Armor (std::string &v_Profession, int &v_Level, std::string v_Type, std::string &v_Name, int v_max=2)
- Armor (Statistics &v_Stats, std::string &v_Name, int &v_Price, int &v_Armor_value, std::string &v_Type)
- void Show ()
- int Get_Armor_value ()

- std::string Get_Type ()
- Statistics Get_Stats ()
- Armor & Get ()

Przyjaciele

- std::ostream & operator<< (std::ostream &s, Armor &x)

Dodatkowe Dziedziczone Składowe

4.2.1 Opis szczegółowy

Klasa Pancerz (pochodna po Przedmiocie)

4.2.2 Dokumentacja konstruktora i destruktora

4.2.2.1 Armor() [1/3]

```
Armor::Armor (
            std::string v_Type,
            std::string & v_Profession )
```

Konstruktor dwu-argumentowy.

4.2.2.2 Armor() [2/3]

```
Armor::Armor (
            std::string & v_Profession,
            int & v_Level,
            std::string v_Type,
            std::string & v_Name,
            int v_max = 2 )
```

Konstruktor wielo-argumentowy, tworzący losowy pancerz.

4.2.2.3 Armor() [3/3]

```
Armor::Armor (
            Statistics & v_Stats,
            std::string & v_Name,
            int & v_Price,
            int & v_Armor_value,
            std::string & v_Type )
```

Konstruktor wielo-argumentowy, tworzący konkretny pancerz.

4.2.3 Dokumentacja funkcji składowych

4.2.3.1 Get()

```
Armor & Armor::Get ( )
```

Funkcja zwracająca referencje do pancerza. (Armor)

4.2.3.2 Get_Armor_value()

```
int Armor::Get_Armor_value ( )
```

Funkcja zwracająca ilość pancerza. (int)

4.2.3.3 Get_Stats()

```
Statistics Armor::Get_Stats ( )  [virtual]
```

Funkcja zwracająca statyski przedmiotu. (Statistics)

Implementuje Item.

4.2.3.4 Get_Type()

```
std::string Armor::Get_Type ( )  [virtual]
```

Funkcja zwracająca typ pancerza. (string)

Implementuje Item.

4.2.3.5 Show()

```
void Armor::Show ( )  [virtual]
```

Funkcja wypisująca pancerza.

Implementuje Item.

4.2.4 Dokumentacja przyjaciół i funkcji związanych

4.2.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & s,
            Armor & x )  [friend]
```

Operator wypisania dla klasy pancerz.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Armor.h
- Armor.cpp

4.3 Dokumentacja klasy Character

```
#include <Character.h>
```

Diagram dziedziczenia dla Character

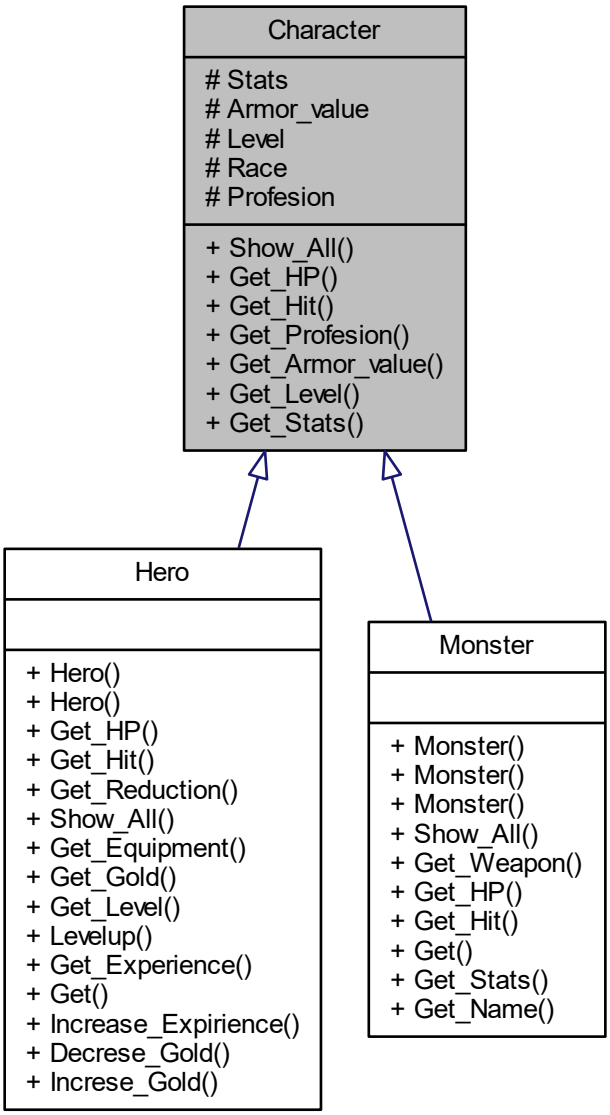
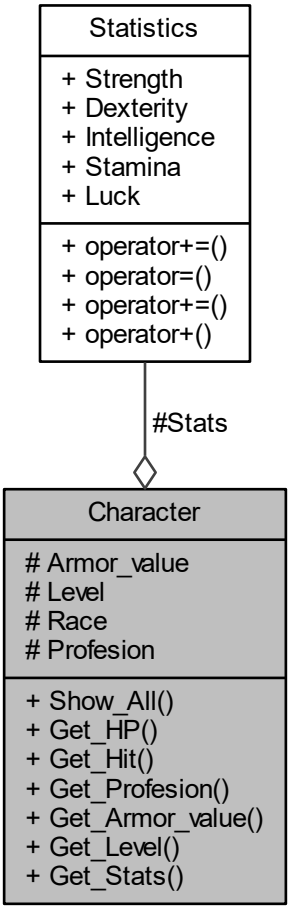


Diagram współpracy dla Character:



Metody publiczne

- virtual void Show_All ()=0
- virtual int Get_HP ()=0
- virtual int Get_Hit ()=0
- std::string Get_Profesion ()
- int Get_Armor_value ()
- virtual int Get_Level ()
- Statistics Get_Stats ()

Atrybuty chronione

- Statistics Stats
- int Armor_value
- int Level
- std::string Race
- std::string Profesion

4.3.1 Opis szczegółowy

Klasa abstrakcyja Postać

4.3.2 Dokumentacja funkcji składowych

4.3.2.1 Get_Armor_value()

```
int Character::Get_Armor_value ( )
```

Funkcja zwracająca wartość pancerza. (int)

4.3.2.2 Get_Hit()

```
virtual int Character::Get_Hit ( )  [pure virtual]
```

Funkcja zwracająca zadane obrażnia. (int)

Implementowany w Monster i Hero.

4.3.2.3 Get_HP()

```
virtual int Character::Get_HP ( )  [pure virtual]
```

Funkcja zwracająca ilość zdrowia bohatera. (int)

Implementowany w Monster i Hero.

4.3.2.4 Get_Level()

```
int Character::Get_Level ( )  [virtual]
```

Funkcja zwracająca poziom. (int)

Reimplementowana w Hero.

4.3.2.5 Get_Profesion()

```
std::string Character::Get_Profesion ( )
```

Funkcja zwracająca profesje. (string)

4.3.2.6 Get_Stats()

```
Statistics Character::Get_Stats ( )
```

Funkcja zwracająca Statystyki. (Statistics)

4.3.2.7 Show_All()

```
virtual void Character::Show_All ( )  [pure virtual]
```

Funkcja wyświetlająca wszystkie informacje o bohaterze. (int)

Implementowany w Monster i Hero.

4.3.3 Dokumentacja atrybutów składowych

4.3.3.1 Armor_value

```
int Character::Armor_value  [protected]
```

Wartość pancerza

4.3.3.2 Level

```
int Character::Level  [protected]
```

Poziom

4.3.3.3 Profesion

```
std::string Character::Profesion  [protected]
```

Profesja

4.3.3.4 Race

```
std::string Character::Race  [protected]
```

Rasa

4.3.3.5 Stats

```
Statistics Character::Stats  [protected]
```

Statystyki postaci

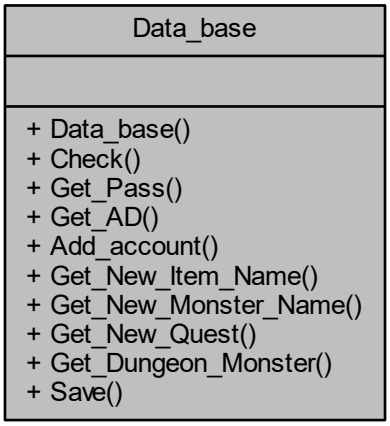
Dokumentacja dla tej klasy została wygenerowana z plików:

- Character.h
- Character.cpp

4.4 Dokumentacja klasy Data_base

```
#include <Data_base.h>
```

Diagram współpracy dla Data_base:



Metody publiczne

- Data_base (std::string v_file)
- bool Check (std::string &t_login)
- std::string Get_Pass (std::string &t_login)
- Account_Data & Get_AD (std::string &t_login, std::string &t_password)
- void Add_account (Account_Data &t_AD)
- std::string Get_New_Item_Name (std::string &v_Proffesion, std::string &v_Type)
- std::string Get_New_Monster_Name (std::string &v_Proffesion)
- std::pair< std::string, std::string > Get_New_Quest ()
- std::shared_ptr< Monster > Get_Dungeon_Monster (int v_Floor, int v_Progress)
- void Save (std::string v_file)

4.4.1 Opis szczegółowy

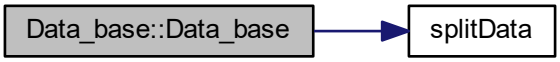
Klasa Bazy Danych

4.4.2 Dokumentacja konstruktora i destruktora

4.4.2.1 Data_base()

```
Data_base::Data_base (
            std::string v_file )
```

Baza opisów Zadań Konstruktor jedno-argumentowy, inicjalizujący baze z plików. Oto graf wywołań dla tej funkcji:



4.4.3 Dokumentacja funkcji składowych

4.4.3.1 Add_account()

```
void Data_base::Add_account (
            Account_Data & t_AD )
```

Funkcja dodająca nowe konto uzytkownika do bazy.

4.4.3.2 Check()

```
bool Data_base::Check (
            std::string & t_login )
```

Funkcja sprawdzająca czy podany login jest zajęty. (bool)

4.4.3.3 Get_AD()

```
Account_Data & Data_base::Get_AD (
            std::string & t_login,
            std::string & t_password )
```

Funkcja zwracajca obiekt danych użytkownika dla podanego loginu i hasła. (Account_Data)

4.4.3.4 Get_Dungeon_Monster()

```
std::shared_ptr< Monster > Data_base::Get_Dungeon_Monster (
            int v_Floor,
            int v_Progress )
```

Funkcja zwracająca wskaźnik na potwora z podanego piętra i progresu. (shared_ptr<Monster>)

4.4.3.5 Get_New_Item_Name()

```
std::string Data_base::Get_New_Item_Name (
            std::string & v_Proffesion,
            std::string & v_Type )
```

Funkcja zwracająca losowa nazwę przedmiotu z bazy. (string)

4.4.3.6 Get_New_Monster_Name()

```
std::string Data_base::Get_New_Monster_Name (
            std::string & v_Proffesion )
```

Funkcja zwracająca losowa nazwę potwora z bazy. (string)

4.4.3.7 Get_New_Quest()

```
std::pair< std::string, std::string > Data_base::Get_New_Quest ( )
```

Funkcja zwracająca krótki i długi opis losowego zadania z bazy. (pair<string,string>)

4.4.3.8 Get_Pass()

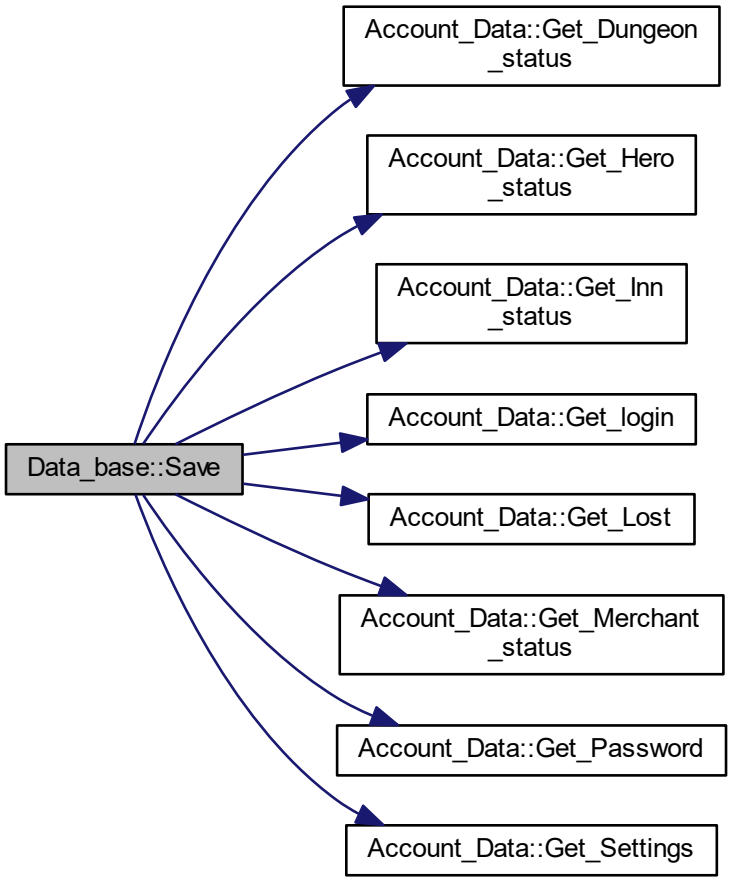
```
std::string Data_base::Get_Pass (
            std::string & t_login )
```

Funkcja zwracajca hasło dla konkretnego loginu. (string)

4.4.3.9 Save()

```
void Data_base::Save (
            std::string v_file )
```

Funkcja zapisująca bazę do plików. Oto graf wywołań dla tej funkcji:



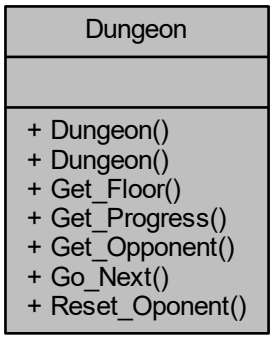
Dokumentacja dla tej klasy została wygenerowana z plików:

- Data_base.h
- Data_base.cpp

4.5 Dokumentacja klasy Dungeon

```
#include <Dungeon.h>
```

Diagram współpracy dla Dungeon:



Metody publiczne

- Dungeon ()
- Dungeon (int v_Floor, int v_Progress, std::shared_ptr< Monster > v_Opponent)
- int Get_Floor ()
- int Get_Progress ()
- std::shared_ptr< Monster > Get_Opponent ()
- void Go_Next ()
- void Reset_Oponent (std::shared_ptr< Monster > v_Oponent)

4.5.1 Opis szczegółowy

Klasa Loch

4.5.2 Dokumentacja konstruktora i destruktora

4.5.2.1 Dungeon() [1/2]

```
Dungeon::Dungeon ( )
```

Konstruktor domyślny

4.5.2.2 Dungeon() [2/2]

```
Dungeon::Dungeon (
            int v_Floor,
            int v_Progress,
            std::shared_ptr< Monster > v_Opponent )
```

Konstruktor wielo-argumentowy, tworzący konkretny loch.

4.5.3 Dokumentacja funkcji składowych

4.5.3.1 Get_Floor()

```
int Dungeon::Get_Floor ( )
```

Funkcja zwracająca piętro. (int)

4.5.3.2 Get_Opponent()

```
std::shared_ptr< Monster > Dungeon::Get_Opponent ( )
```

Funkcja zwracająca wskaźnik na przeciwnika. (shared_ptr<Monster>)

4.5.3.3 Get_Progress()

```
int Dungeon::Get_Progress ( )
```

Funkcja zwracająca progress. (int)

4.5.3.4 Go_Next()

```
void Dungeon::Go_Next ( )
```

Funkcja przestawiająca loch na następny poziom.

4.5.3.5 Reset_Oponent()

```
void Dungeon::Reset_Oponent (
            std::shared_ptr< Monster > v_Oponent )
```

Funkcja resetująca przeciwnika.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Dungeon.h
- Dungeon.cpp

4.6 Dokumentacja klasy Equipment

`#include <Equipment.h>`

Diagram współpracy dla Equipment:

Metody publiczne

- Equipment ()
- Equipment (std::string &v_Profession)
- Equipment (std::vector< std::shared_ptr< Item >> v_Bag, std::shared_ptr< Weapon > v_e_Weapon, std::shared_ptr< Armor > v_e_Helm, std::shared_ptr< Armor > v_e_Armor, std::shared_ptr< Armor > v_e_Gloves)
- void Show_Bag ()
- void Show_Equiped ()
- void Equip (int ch)
- int Get_Bag_Size ()
- bool Bag_full ()
- bool Add_Item (std::shared_ptr< Item > v_Item)
- void Remove_Item (int ch)
- std::shared_ptr< Item > Get_Item (int ch)
- std::shared_ptr< Armor > Get_Armor (int ch)
- std::shared_ptr< Weapon > Get_Weapon ()
- Statistics Get_Equiped_Stats ()
- int Get_Armor_Value ()

4.6.1 Opis szczegółowy

Klasa Ekwipunek

4.6.2 Dokumentacja konstruktora i destruktora

4.6.2.1 Equipment() [1/3]

```
Equipment::Equipment ( )
```

Konstruktor domyślny.

4.6.2.2 Equipment() [2/3]

```
Equipment::Equipment (
            std::string & v_Profession )
```

Konstruktor jedno-argumentowy.

4.6.2.3 Equipment() [3/3]

```
Equipment::Equipment (
            std::vector< std::shared_ptr< Item >> v_Bag,
            std::shared_ptr< Weapon > v_e_Weapon,
            std::shared_ptr< Armor > v_e_Helm,
            std::shared_ptr< Armor > v_e_Armor,
            std::shared_ptr< Armor > v_e_Gloves )
```

Konstruktor wielo-argumentowy, tworzący konkretny ekwipunek.

4.6.3 Dokumentacja funkcji składowych

4.6.3.1 Add_Item()

```
bool Equipment::Add_Item (
            std::shared_ptr< Item > v_Item )
```

Funkcja dodająca przedmiot do plecaka i informująca czy się powiodło. (bool)

4.6.3.2 Bag_full()

```
bool Equipment::Bag_full ( )
```

Funkcja zakładająca informacje czy plecak jest pełny. (bool)

4.6.3.3 Equip()

```
void Equipment::Equip (
            int ch )
```

Funkcja zakładająca konkretny przedmiot z plecaka.

4.6.3.4 Get_Armor()

```
std::shared_ptr< Armor > Equipment::Get_Armor (
            int ch )
```

Funkcja zwracająca wskaźnik na wybrany pancerz. (shared_ptr<Armor>)

4.6.3.5 Get_Armor_Value()

```
int Equipment::Get_Armor_Value ( )
```

Funkcja zwracająca wartość założonego pancerza. (int)

4.6.3.6 Get_Bag_Size()

```
int Equipment::Get_Bag_Size ( )
```

Funkcja zwracająca rozmiar plecaka. (int)

4.6.3.7 Get_Equiped_Stats()

```
Statistics Equipment::Get_Equiped_Stats ( )
```

Funkcja zwracająca statyskyki założonych przedmiotów. (Statistics)

4.6.3.8 Get_Item()

```
std::shared_ptr< Item > Equipment::Get_Item (
            int ch )
```

Funkcja zwracająca wskaźnik na wybrany przedmiot. (shared_ptr<Item>)

4.6.3.9 Get_Weapon()

```
std::shared_ptr< Weapon > Equipment::Get_Weapon ( )
```

Funkcja zwracająca wskaźnik na broń. (shared_ptr<Weapon>)

4.6.3.10 Remove_Item()

```
void Equipment::Remove_Item (
            int ch )
```

Funkcja usuwająca przedmiot z plecaka.

4.6.3.11 Show_Bag()

```
void Equipment::Show_Bag ( )
```

Funkcja wypisująca zawartość plecaka.

4.6.3.12 Show_Equiped()

```
void Equipment::Show_Equiped ( )
```

Funkcja wypisująca założone przedmioty.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Equipment.h
- Equipment.cpp

4.7 Dokumentacja klasy Hero

```
#include <Hero.h>
```

Diagram dziedziczenia dla Hero

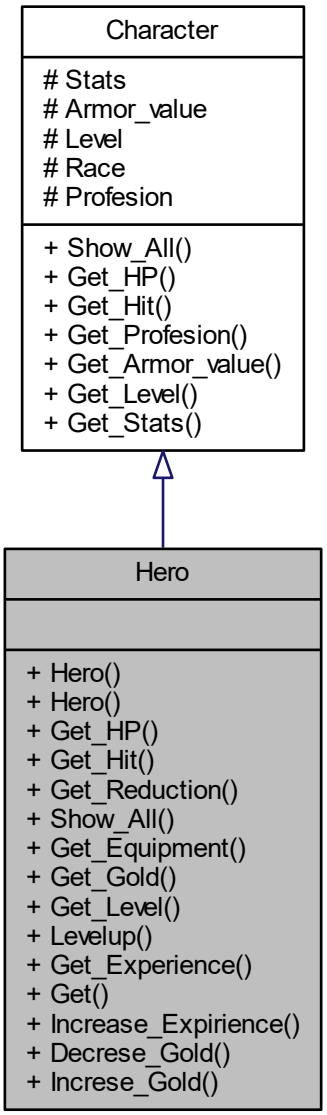
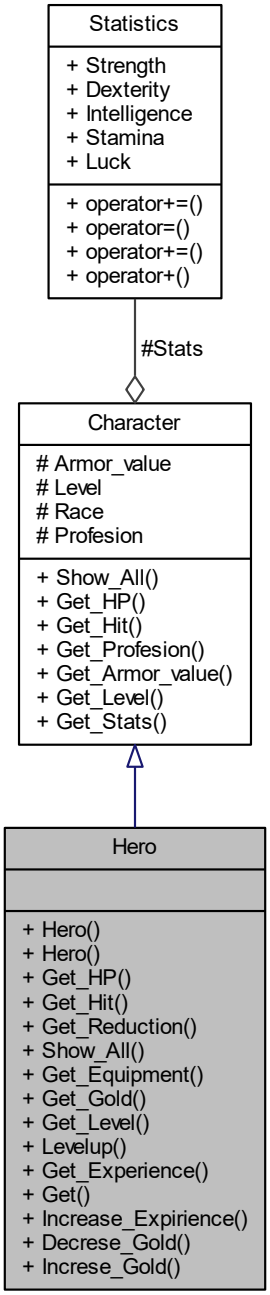


Diagram współpracy dla Hero:



Metody publiczne

- Hero (std::string &v_Race, std::string &v_Profesion)
- Hero (Statistics &v_Stats, int &v_Armor_Value, int &v_Level, std::string &v_Race, std::string &v_Proffesion, std::shared_ptr< Equipment > v_Hero_Equipment, int &v_Gold, int &v_Expirience)
- int Get_HP ()
- int Get_Hit ()

- int Get_Reduction ()
- void Show_All ()
- std::shared_ptr< Equipment > Get_Equipment ()
- int Get_Gold ()
- int Get_Level ()
- void Levelup ()
- int Get_Experience ()
- Hero & Get ()
- void Increase_Expirience (int v_Expirience)
- void Decrese_Gold (int v_gold)
- void Increse_Gold (int v_gold)

Przyjaciele

- std::ostream & operator<< (std::ostream &s, Hero &x)

Dodatkowe Dziedziczone Składowe

4.7.1 Opis szczegółowy

Klasa Bohater (pochodna po Postaci)

4.7.2 Dokumentacja konstruktora i destruktora

4.7.2.1 Hero() [1/2]

```
Hero::Hero (
            std::string & v_Race,
            std::string & v_Profesion )
```

Doświadczenie Konstruktor dwu-argumentowy.

4.7.2.2 Hero() [2/2]

```
Hero::Hero (
            Statistics & v_Stats,
            int & v_Armor_Value,
            int & v_Level,
            std::string & v_Race,
            std::string & v_Proffesion,
            std::shared_ptr< Equipment > v_Hero_Equipment,
            int & v_Gold,
            int & v_Expirience )
```

Konstruktor wielo-argumentowy, tworzący konkretnego Bohatera.

4.7.3 Dokumentacja funkcji składowych

4.7.3.1 Decrese_Gold()

```
void Hero::Decrese_Gold (
            int v_gold )
```

Funkcja zmniejszająca ilość złota o podaną wartość.

4.7.3.2 Get()

```
Hero & Hero::Get ( )
```

Funkcja zwracająca referencej do bohatera. (Hero)

4.7.3.3 Get_Equipment()

```
std::shared_ptr< Equipment > Hero::Get_Equipment ( )
```

Funkcja zwracająca wskaźnik na ekwipunek. (shared_ptr<Equipment>)

4.7.3.4 Get_Experience()

```
int Hero::Get_Experience ( )
```

Funkcja zwracająca ilość doświadczenia. (int)

4.7.3.5 Get_Gold()

```
int Hero::Get_Gold ( )
```

Funkcja zwracająca ilość złota. (int)

4.7.3.6 Get_Hit()

```
int Hero::Get_Hit ( )  [virtual]
```

Funkcja zwracająca zadane obrażnia. (int)

Implementuje Character.

4.7.3.7 Get_HP()

```
int Hero::Get_HP ( )  [virtual]
```

Funkcja zwracająca ilość zdrowia bohatera. (int)

Implementuje Character.

4.7.3.8 Get_Level()

```
int Hero::Get_Level ( )  [virtual]
```

Funkcja zwracająca poziom. (int)

Reimplementowana z Character.

4.7.3.9 Get_Reduction()

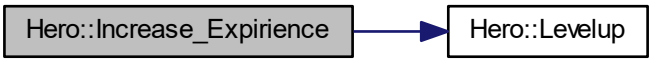
```
int Hero::Get_Reduction ( )
```

Funkcja zwracająca redukcje obrażeń. (int)

4.7.3.10 Increase_Expirience()

```
void Hero::Increase_Expirience (
            int v_Expirience )
```

Funkcja zwiększająca ilość doświadczenia o podaną wartość. Oto graf wywołań dla tej funkcji:



4.7.3.11 Increse_Gold()

```
void Hero::Increse_Gold (
            int v_gold )
```

Funkcja zwiększająca ilość złota o podaną wartość.

4.7.3.12 Levelup()

```
void Hero::Levelup ( )
```

Funkcja przeprowadzająca proces podnoszenia poziomu o ile jest on możliwy.

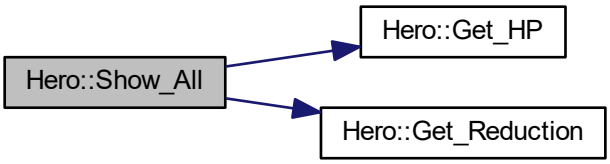
4.7.3.13 Show_All()

```
void Hero::Show_All ( )  [virtual]
```

Funkcja Wyświetlająca wszystkie informacje o bohaterze. (int)

Implementuje Character.

Oto graf wywołań dla tej funkcji:



4.7.4 Dokumentacja przyjaciół i funkcji związanych

4.7.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & s,
            Hero & x )  [friend]
```

Operator wypisania dla klasy Bohater.

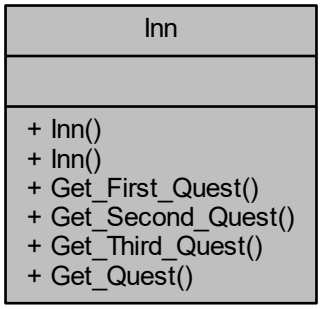
Dokumentacja dla tej klasy została wygenerowana z plików:

- Hero.h
- Hero.cpp

4.8 Dokumentacja klasy Inn

`#include <Inn.h>`

Diagram współpracy dla Inn:



Metody publiczne

- Inn ()
- Inn (std::shared_ptr< Quest > v_First_Quest, std::shared_ptr< Quest > v_Second_Quest, std::shared_ptr< Quest > v_Third_Quest)
- std::shared_ptr< Quest > Get_First_Quest ()
- std::shared_ptr< Quest > Get_Second_Quest ()
- std::shared_ptr< Quest > Get_Third_Quest ()
- std::shared_ptr< Quest > Get_Quest (int v)

4.8.1 Opis szczegółowy

Klasa Karczma

4.8.2 Dokumentacja konstruktora i destruktora

4.8.2.1 Inn() [1/2]

`Inn::Inn ( )`

Konstruktor bez-argumentowy.

4.8.2.2 Inn() [2/2]

```
Inn::Inn (
            std::shared_ptr< Quest > v_First_Quest,
            std::shared_ptr< Quest > v_Second_Quest,
            std::shared_ptr< Quest > v_Third_Quest )
```

Konstruktor wielo-argumentowy, tworzący konkretną Karczmę.

4.8.3 Dokumentacja funkcji składowych

4.8.3.1 Get_First_Quest()

```
std::shared_ptr< Quest > Inn::Get_First_Quest ( )
```

Funkcaj zwracająca wskaźnik na pierwsze zadanie. (shared_ptr<Quest>)

4.8.3.2 Get_Quest()

```
std::shared_ptr< Quest > Inn::Get_Quest (
            int v )
```

Funkcaj zwracająca wskaźnik na wybrane zadanie. (shared_ptr<Quest>)

4.8.3.3 Get_Second_Quest()

```
std::shared_ptr< Quest > Inn::Get_Second_Quest ( )
```

Funkcaj zwracająca wskaźnik na drugie zadanie. (shared_ptr<Quest>)

4.8.3.4 Get_Third_Quest()

```
std::shared_ptr< Quest > Inn::Get_Third_Quest ( )
```

Funkcaj zwracająca wskaźnik na trzecie zadanie. (shared_ptr<Quest>)

Dokumentacja dla tej klasy została wygenerowana z plików:

- Inn.h
- Inn.cpp

4.9 Dokumentacja klasy Item

`#include <Item.h>`

Diagram dziedziczenia dla Item

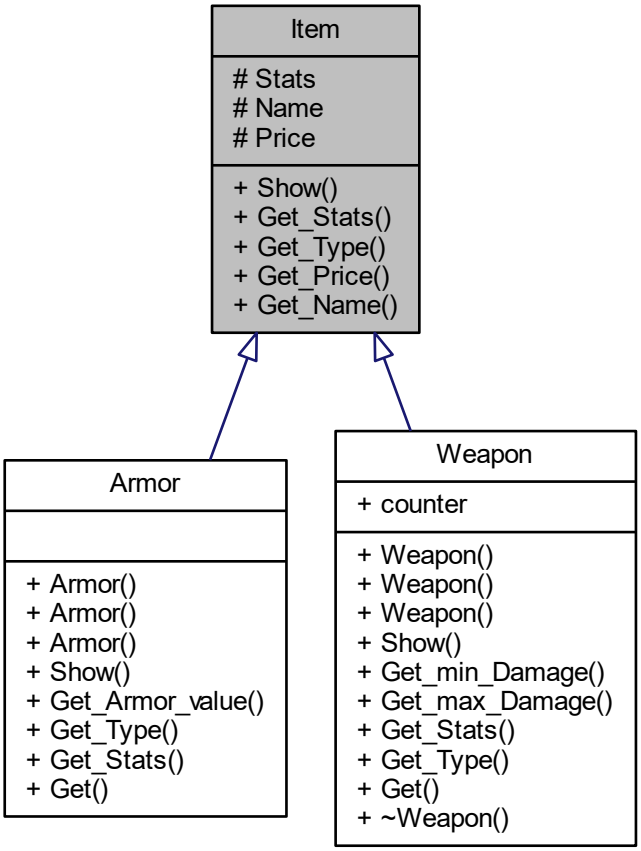
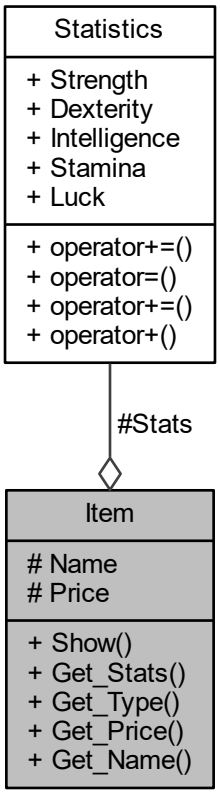


Diagram współpracy dla Item:



Metody publiczne

- virtual void Show ()=0
- virtual Statistics Get_Stats ()=0
- virtual std::string Get_Type ()=0
- int Get_Price ()
- std::string Get_Name ()

Atrybuty chronione

- Statistics Stats
- std::string Name
- int Price

4.9.1 Opis szczegółowy

Klasa abstrakcyjna Przedmiot

4.9.2 Dokumentacja funkcji składowych

4.9.2.1 Get_Name()

```
std::string Item::Get_Name ( )
```

Funkcaj zwracająca nazwę przedmiotu. (string)

4.9.2.2 Get_Price()

```
int Item::Get_Price ( )
```

Funkcaj zwracająca cenę przedmiotu. (int)

4.9.2.3 Get_Stats()

```
virtual Statistics Item::Get_Stats ( )  [pure virtual]
```

Funkcja zwracająca statyski przedmiotu. (Statistics)

Implementowany w Weapon i Armor.

4.9.2.4 Get_Type()

```
virtual std::string Item::Get_Type ( )  [pure virtual]
```

Funkcja zwracająca typ przedmiotu. (string)

Implementowany w Weapon i Armor.

4.9.2.5 Show()

```
virtual void Item::Show ( )  [pure virtual]
```

Funkcja wypisująca przedmiot.

Implementowany w Weapon i Armor.

4.9.3 Dokumentacja atrybutów składowych

4.9.3.1 Name

```
std::string Item::Name  [protected]
```

Nazwa przedmiotu

4.9.3.2 Price

```
int Item::Price  [protected]
```

Cena przedmiotu

4.9.3.3 Stats

```
Statistics Item::Stats  [protected]
```

Statystyki przedmiotu

Dokumentacja dla tej klasy została wygenerowana z plików:

- Item.h
- Item.cpp

4.10 Dokumentacja klasy Merchant

```
#include <Merchant.h>
```

Diagram współpracy dla Merchant:

Metody publiczne

- Merchant ()
- Merchant (std::string v_Proffesion)
- Merchant (std::shared_ptr< Item > v_First_Item, std::shared_ptr< Item > v_Second_Item, std::shared_ptr< Item > v_Third_Item)
- std::shared_ptr< Item > Get_Item (int ch)
- bool Buy_Item (int ch, std::shared_ptr< Hero > v_Hero)
- bool Sell_Item (int ch, std::shared_ptr< Hero > v_Hero)
- void Add_New_Item (int ch, int v_Level, std::string &v_Proffesion, std::string &v_Name, std::string &v_Type)
- void Show_All ()

4.10.1 Opis szczegółowy

Klasa Kupiec

4.10.2 Dokumentacja konstruktora i destruktora

4.10.2.1 Merchant() [1/3]

```
Merchant::Merchant ( )
```

Konstruktor bez-argumentowy.

4.10.2.2 Merchant() [2/3]

```
Merchant::Merchant (
            std::string v_Proffesion )
```

Konstruktor jedno-argumentowy.

4.10.2.3 Merchant() [3/3]

```
Merchant::Merchant (
            std::shared_ptr< Item > v_First_Item,
            std::shared_ptr< Item > v_Second_Item,
            std::shared_ptr< Item > v_Third_Item )
```

Konstruktor wielo-argumentowy, tworzący konkretnego kupca.

4.10.3 Dokumentacja funkcji składowych

4.10.3.1 Add_New_Item()

```
void Merchant::Add_New_Item (
             int ch,
             int v_Level,
             std::string & v_Proffesion,
             std::string & v_Name,
             std::string & v_Type )
```

Funkcja dodająca nowy losowy przedmiot.

4.10.3.2 Buy_Item()

```
bool Merchant::Buy_Item (
             int ch,
             std::shared_ptr< Hero > v_Hero )
```

Funkcja zakupująca konkretny przedmiot. (bool)

4.10.3.3 Get_Item()

```
std::shared_ptr< Item > Merchant::Get_Item (
             int ch )
```

Funkcja zwracająca konkretny przedmiot. (shared_ptr<Item>)

4.10.3.4 Sell_Item()

```
bool Merchant::Sell_Item (
             int ch,
             std::shared_ptr< Hero > v_Hero )
```

Funkcja sprzedająca konkretny przedmiot. (bool)

4.10.3.5 Show_All()

```
void Merchant::Show_All ( )
```

Funkcja wyświetlająca zawartość kupca.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Merchant.h
- Merchant.cpp

4.11 Dokumentacja klasy Monster

`#include <Monster.h>`

Diagram dziedziczenia dla Monster

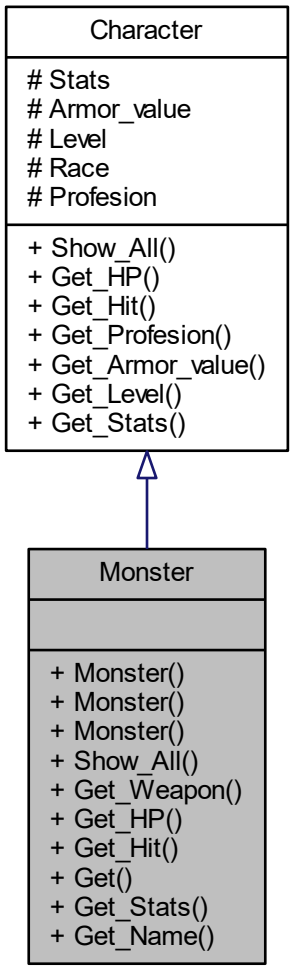
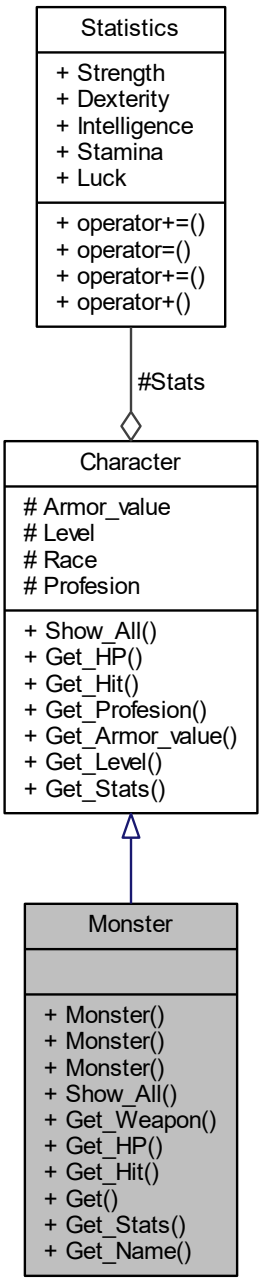


Diagram współpracy dla Monster:



Metody publiczne

- Monster (std::string &v_Race, std::string &v_Profesion)
- Monster (Statistics &v_Stats, int &v_Armor_Value, std::string &v_Name, int &v_Level, std::string &v_Race, std::string &v_Proffesion, std::shared_ptr< Weapon > v_Monster_Weapon)
- Monster (std::string &v_Proffesion, std::string &v_Race, std::string &v_Monster_Name, int &v_Level, std↩ ::string &v_Weapon_Name)

- void Show_All ()
- std::shared_ptr< Weapon > Get_Weapon ()
- int Get_HP ()
- int Get_Hit ()
- Monster & Get ()
- Statistics Get_Stats ()
- std::string Get_Name ()

Przyjaciele

- std::ostream & operator<< (std::ostream &s, Monster &x)

Dodatkowe Dziedziczone Składowe

4.11.1 Opis szczegółowy

Klasa Potwór (pochodna po Postaci)

4.11.2 Dokumentacja konstruktora i destruktora

4.11.2.1 Monster() [1/3]

```
Monster::Monster (
            std::string & v_Race,
            std::string & v_Profesion )
```

Nazwa Konstruktor dwu-argumentowy.

4.11.2.2 Monster() [2/3]

```
Monster::Monster (
            Statistics & v_Stats,
            int & v_Armor_Value,
            std::string & v_Name,
            int & v_Level,
            std::string & v_Race,
            std::string & v_Proffesion,
            std::shared_ptr< Weapon > v_Monster_Weapon )
```

Konstruktor wielo-argumentowy, tworzący konkretnego potwora

4.11.2.3 Monster() [3/3]

```
Monster::Monster (
            std::string & v_Proffesion,
            std::string & v_Race,
            std::string & v_Monster_Name,
            int & v_Level,
            std::string & v_Weapon_Name )
```

Konstruktor wielo-argumentowy, tworzący losowego potwora

4.11.3 Dokumentacja funkcji składowych

4.11.3.1 Get()

```
Monster & Monster::Get ( )
```

Funkcja zracająca referencje do Potwora. (Monster)

4.11.3.2 Get_Hit()

```
int Monster::Get_Hit ( )  [virtual]
```

Funkcja zwracająca zadane obrażnia. (int)

Implementuje Character.

4.11.3.3 Get_HP()

```
int Monster::Get_HP ( )  [virtual]
```

Funkcja zwracająca ilość zdrowia potwora. (int)

Implementuje Character.

4.11.3.4 Get_Name()

```
std::string Monster::Get_Name ( )
```

Funkcja zracająca nazwe potwora. (string)

4.11.3.5 Get_Stats()

```
Statistics Monster::Get_Stats ( )
```

Funkcja zracająca statystki potwora. (Statistics)

4.11.3.6 Get_Weapon()

```
std::shared_ptr< Weapon > Monster::Get_Weapon ( )
```

Funkcja zwracająca wskaźnik na broń potwora. (shared_ptr<Weapon>)

4.11.3.7 Show_All()

```
void Monster::Show_All ( )  [virtual]
```

Funkcja wypisująca wszystkie informacje o potworze.

Implementuje Character.

Oto graf wywołań dla tej funkcji:

4.11.4 Dokumentacja przyjaciół i funkcji związanych

4.11.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & s,
            Monster & x )  [friend]
```

Operator wypisania dla klasy Potwór.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Monster.h
- Monster.cpp

4.12 Dokumentacja struktury Options

`#include <Struct.h>`

Diagram współpracy dla Options:

Metody publiczne

- Options Get ()

Atrybuty publiczne

- bool FastGame
- bool Hardcore

Przyjaciele

- std::ostream & operator<< (std::ostream &s, Options x)

4.12.1 Opis szczegółowy

Struktura Ustawień

4.12.2 Dokumentacja funkcji składowych

4.12.2.1 Get()

`Options Options::Get ( )`

Funkcja zwracająca Ustawienia. (Options)

4.12.3 Dokumentacja przyjaciół i funkcji związanych

4.12.3.1 operator<<

```
std::ostream& operator<< (
            std::ostream & s,
            Options x )  [friend]
```

Operator wypisania dla Ustawień.

4.12.4 Dokumentacja atrybutów składowych

4.12.4.1 FastGame

```
bool Options::FastGame
```

Szybka gra

4.12.4.2 Hardcore

```
bool Options::Hardcore
```

Poziom Hardcore

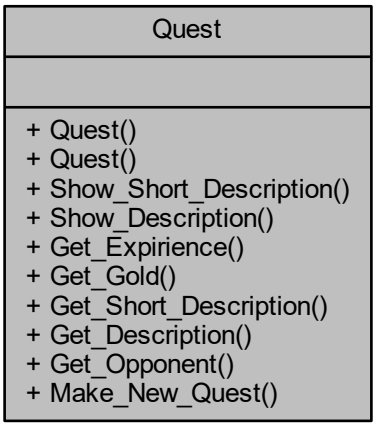
Dokumentacja dla tej struktury została wygenerowana z plików:

- Struct.h
- Struct.cpp

4.13 Dokumentacja klasy Quest

`#include <Quest.h>`

Diagram współpracy dla Quest:



Metody publiczne

- Quest ()
- Quest (std::string &v_Short_Description, std::string v_Description, int &v_Expirience, int &v_Gold, std::shared_ptr< Monster > v_Opponent)
- void Show_Short_Description ()
- void Show_Description ()
- int Get_Expirience ()
- int Get_Gold ()
- std::string Get_Short_Description ()
- std::string Get_Description ()
- std::shared_ptr< Monster > Get_Opponent ()
- void Make_New_Quest (std::string &v_Short_Description, std::string &v_Description, int &v_Level, std::string &v_Proffesion, std::string &v_Monster_Name, std::string &v_Weapon_Name)

4.13.1 Opis szczegółowy

Klasa Zadanie

4.13.2 Dokumentacja konstruktora i destruktora

4.13.2.1 Quest() [1/2]

```
Quest::Quest ( )
```

Konstruktor domyślny.

4.13.2.2 Quest() [2/2]

```
Quest::Quest (
            std::string & v_Short_Description,
            std::string v_Description,
            int & v_Expirience,
            int & v_Gold,
            std::shared_ptr< Monster > v_Opponent )
```

Konstruktor wielo-argumentowy, tworzący konkretne zadanie

4.13.3 Dokumentacja funkcji składowych

4.13.3.1 Get_Description()

```
std::string Quest::Get_Description ( )
```

Funkcja zwracająca opis. (string)

4.13.3.2 Get_Expirience()

```
int Quest::Get_Expirience ( )
```

Funkcja zwracająca doświadczenie. (int)

4.13.3.3 Get_Gold()

```
int Quest::Get_Gold ( )
```

Funkcja zwracająca złoto. (int)

4.13.3.4 Get_Opponent()

```
std::shared_ptr< Monster > Quest::Get_Opponent ( )
```

Funkcja zwracająca przeciwnika. (shared_ptr<Monster>)

4.13.3.5 Get_Short_Description()

```
std::string Quest::Get_Short_Description ( )
```

Funkcja zwracająca krótki opis. (string)

4.13.3.6 Make_New_Quest()

```
void Quest::Make_New_Quest (
            std::string & v_Short_Description,
            std::string & v_Description,
            int & v_Level,
            std::string & v_Proffesion,
            std::string & v_Monster_Name,
            std::string & v_Weapon_Name )
```

Funkcja generująca nowe zadanie z otrzymanych danych. Oto graf wywołań dla tej funkcji:

4.13.3.7 Show_Description()

```
void Quest::Show_Description ( )
```

Funkcja wypisująca opis.

4.13.3.8 Show_Short_Description()

```
void Quest::Show_Short_Description ( )
```

Funkcja wypisująca krótki opis.

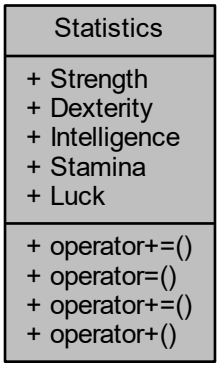
Dokumentacja dla tej klasy została wygenerowana z plików:

- Quest.h
- Quest.cpp

4.14 Dokumentacja struktury Statistics

`#include <Struct.h>`

Diagram współpracy dla Statistics:



Metody publiczne

- Statistics operator+= (Statistics x)
- Statistics operator= (int x)
- Statistics operator+= (int x)
- Statistics operator+ (Statistics x)

Atrybuty publiczne

- int Strength = 0
- int Dexterity = 0
- int Intelligence = 0
- int Stamina = 0
- int Luck = 0

Przyjaciele

- std::ostream & operator<< (std::ostream &s, Statistics &x)

4.14.1 Opis szczegółowy

Struktura Statystyk

4.14.2 Dokumentacja funkcji składowych

4.14.2.1 operator+()

```
Statistics Statistics::operator+ (
            Statistics x )
```

Operator dodawania dla Statystyk. (Staticstic)

4.14.2.2 operator+=() [1/2]

```
Statistics Statistics::operator+= (
            int x )
```

Operator "+=" dla Statystyk. (int)

4.14.2.3 operator+=() [2/2]

```
Statistics Statistics::operator+= (
            Statistics x )
```

Operator "+=" dla Statystyk.

4.14.2.4 operator=()

```
Statistics Statistics::operator= (
            int x )
```

Operator dodawania dla Statystyk. (int)

4.14.3 Dokumentacja przyjaciół i funkcji związanych

4.14.3.1 operator<<

```
std::ostream& operator<< (
            std::ostream & s,
            Statistics & x )  [friend]
```

Operator wypisania dla Statystyk.

4.14.4 Dokumentacja atrybutów składowych

4.14.4.1 Dexterity

```
int Statistics::Dexterity = 0
```

Zręczność

4.14.4.2 Intelligence

```
int Statistics::Intelligence = 0
```

Inteligencja

4.14.4.3 Luck

```
int Statistics::Luck = 0
```

Szczęście

4.14.4.4 Stamina

```
int Statistics::Stamina = 0
```

Wytrzymałość

4.14.4.5 Strength

```
int Statistics::Strength = 0
```

Siła

Dokumentacja dla tej struktury została wygenerowana z plików:

- Struct.h
- Struct.cpp

4.15 Dokumentacja klasy Weapon

```
#include <Weapon.h>
```

Diagram dziedziczenia dla Weapon

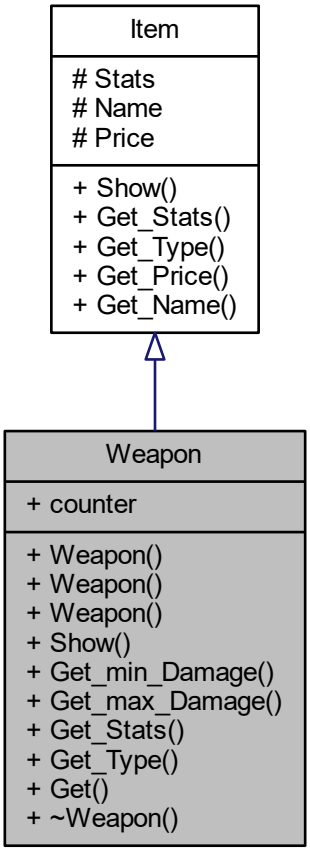
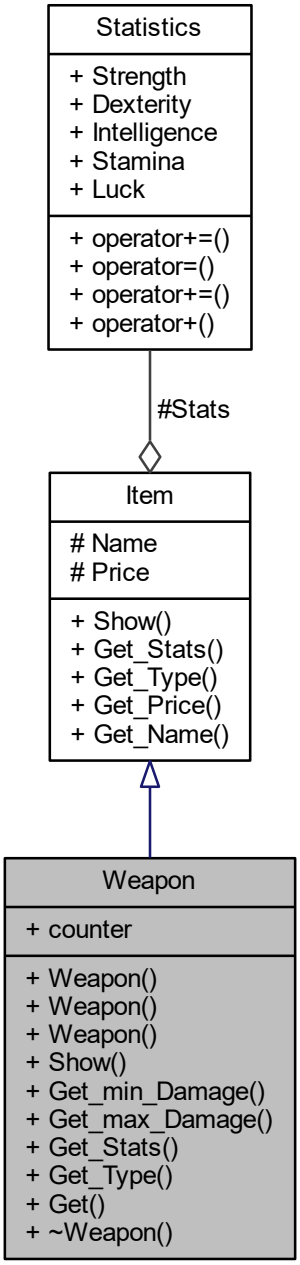


Diagram współpracy dla Weapon:



Metody publiczne

- Weapon (std::string &v_Profession)
- Weapon (std::string &v_Profession, int &v_Level, std::string &v_Name, int v_max=1)
- Weapon (Statistics &v_Stats, std::string &v_Name, int &v_Price, int &v_min_Damage, int &v_max_Damage)
- void Show ()
- int Get_min_Damage ()

- int Get_max_Damage ()
- Statistics Get_Stats ()
- std::string Get_Type ()
- Weapon & Get ()
- ~Weapon ()

Statyczne atrybuty publiczne

- static int **counter** = 0

Przyjaciele

- std::ostream & operator<< (std::ostream &s, Weapon &x)

Dodatkowe Dziedziczone Składowe

4.15.1 Opis szczegółowy

Klasa Broń (pochodna po Przedmiocie)

4.15.2 Dokumentacja konstruktora i destruktora

4.15.2.1 Weapon() [1/3]

```
Weapon::Weapon (
            std::string & v_Profession )
```

Konstruktor jedno-argumentowy.

4.15.2.2 Weapon() [2/3]

```
Weapon::Weapon (
            std::string & v_Profession,
            int & v_Level,
            std::string & v_Name,
            int v_max = 1 )
```

Konstruktor wielo-argumentowy, tworzący losową broń.

4.15.2.3 Weapon() [3/3]

```
Weapon::Weapon (
            Statistics & v_Stats,
            std::string & v_Name,
            int & v_Price,
            int & v_min_Damage,
            int & v_max_Damage )
```

Konstruktor wielo-argumentowy, tworzący konkretną broń.

4.15.2.4 ~Weapon()

```
Weapon::~Weapon ( )
```

Destruktor klasy Broń

4.15.3 Dokumentacja funkcji składowych

4.15.3.1 Get()

```
Weapon & Weapon::Get ( )
```

Funkcja zwracająca referencje do broni. (Weapon)

4.15.3.2 Get_max_Damage()

```
int Weapon::Get_max_Damage ( )
```

Funkcja zwracająca maksymalną wartość obrażeń. (int)

4.15.3.3 Get_min_Damage()

```
int Weapon::Get_min_Damage ( )
```

Funkcja zwracająca minimalna wartość obrażeń. (int)

4.15.3.4 Get_Stats()

```
Statistics Weapon::Get_Stats ( )  [virtual]
```

Funkcja zwracająca statyski przedmiotu. (Statistics)

Implementuje Item.

4.15.3.5 Get_Type()

```
std::string Weapon::Get_Type ( )  [virtual]
```

Funkcja zwracająca typ przedmiotu tu "broń". (string)

Implementuje Item.

4.15.3.6 Show()

```
void Weapon::Show ( )  [virtual]
```

Funkcja wypisująca pancerza.

Implementuje Item.

4.15.4 Dokumentacja przyjaciół i funkcji związanych

4.15.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & s,
            Weapon & x )  [friend]
```

Operator wypisania dla klasy Broń.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Weapon.h
- Weapon.cpp

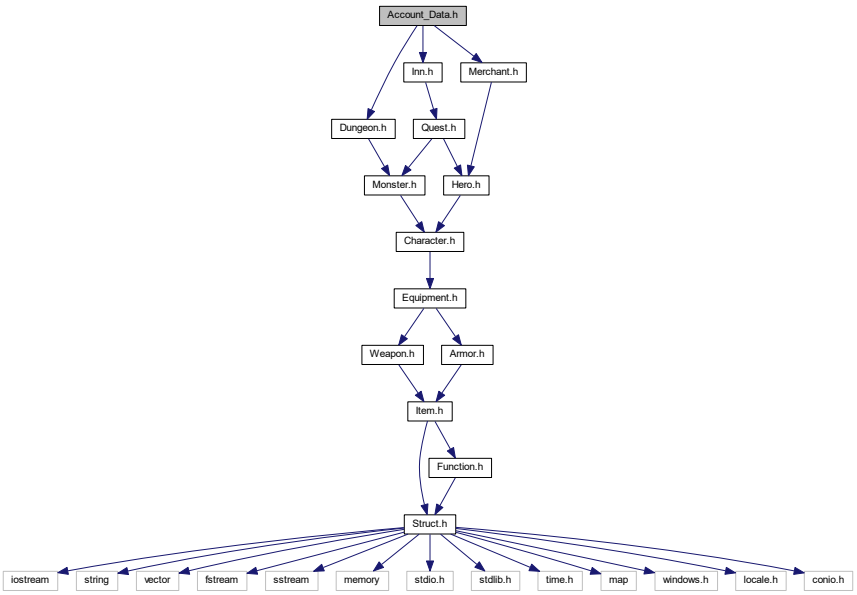
Rozdział 5

Dokumentacja plików

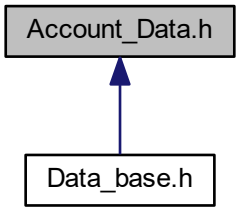
5.1 Dokumentacja pliku Account_Data.h

```
#include "Inn.h"
#include "Dungeon.h"
#include "Merchant.h"
```

Wykres zależności załączania dla Account_Data.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



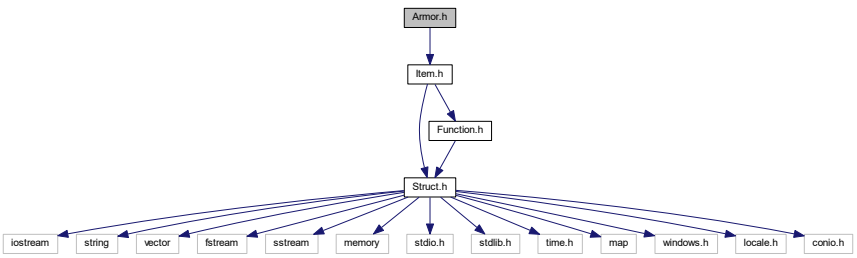
Komponenty

- class Account_Data

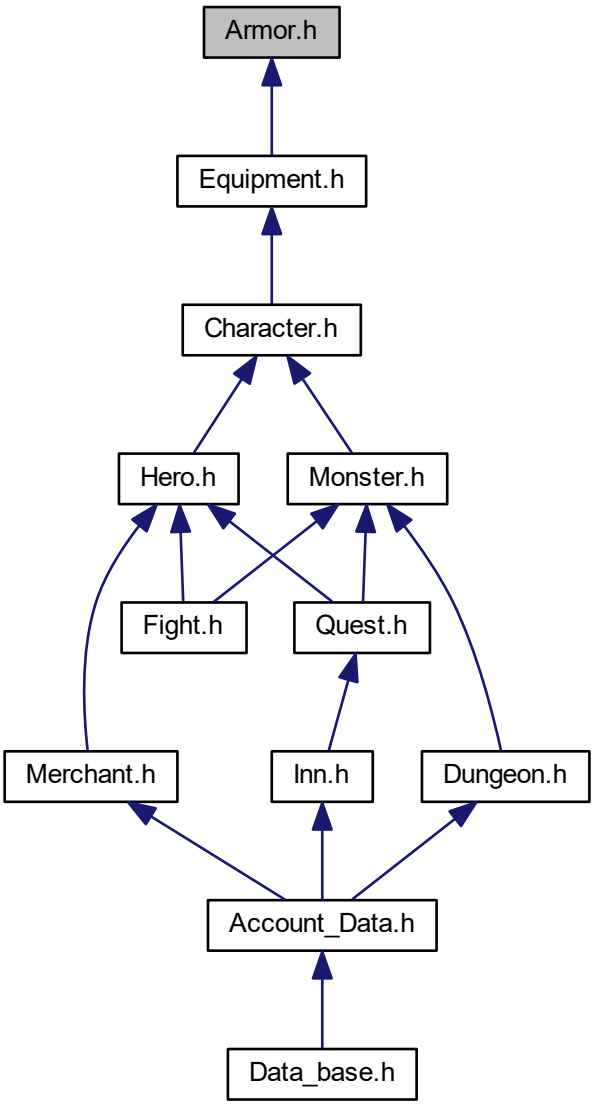
5.2 Dokumentacja pliku Armor.h

```
#include "Item.h"
```

Wykres zależności załączania dla Armor.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



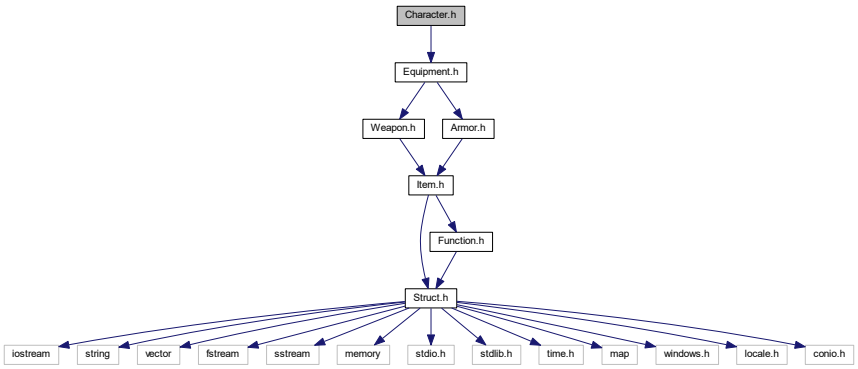
Komponenty

- class Armor

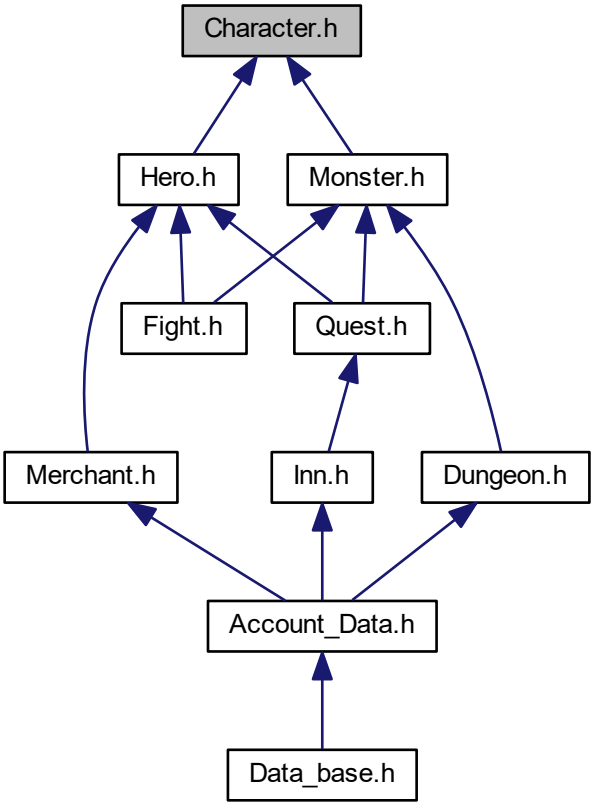
5.3 Dokumentacja pliku Character.h

```
#include "Equipment.h"
```

Wykres zależności załączania dla Character.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



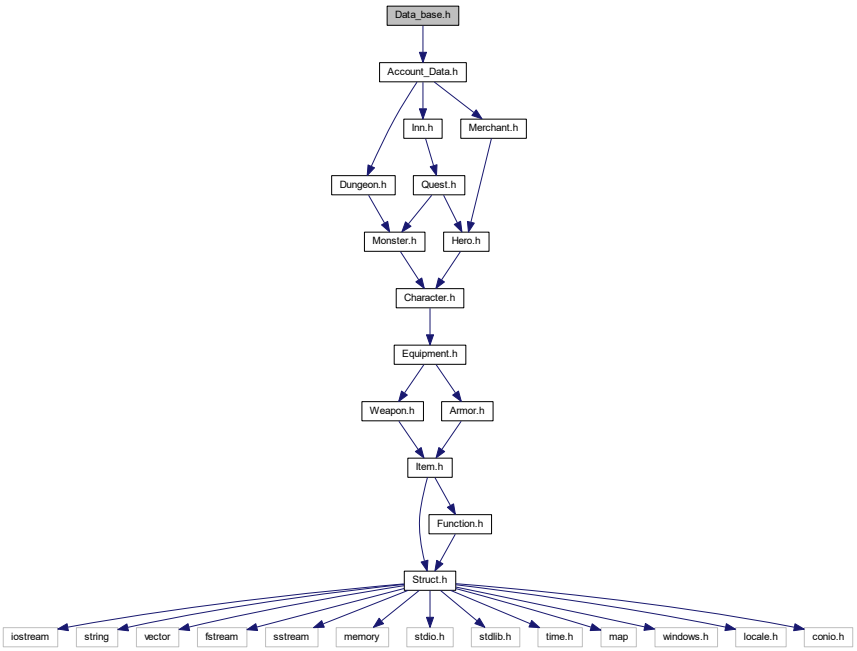
Komponenty

- class Character

5.4 Dokumentacja pliku Data_base.h

```
#include "Account_Data.h"
```

Wykres zależności załączania dla Data_base.h:



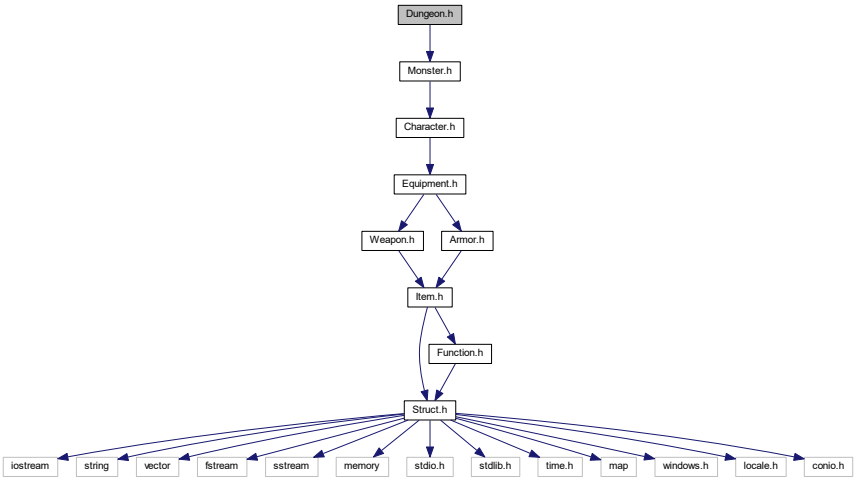
Komponenty

- class Data_base

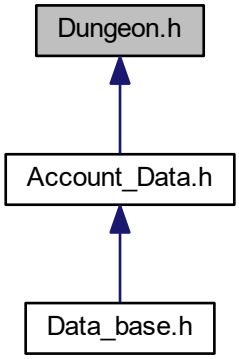
5.5 Dokumentacja pliku Dungeon.h

```
#include "Monster.h"
```

Wykres zależności załączania dla Dungeon.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



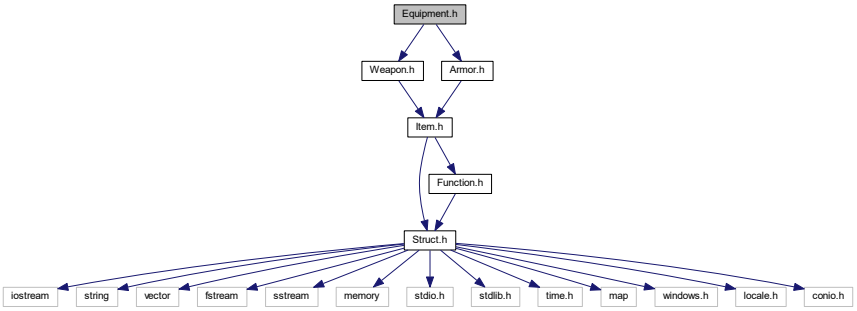
Komponenty

- class Dungeon

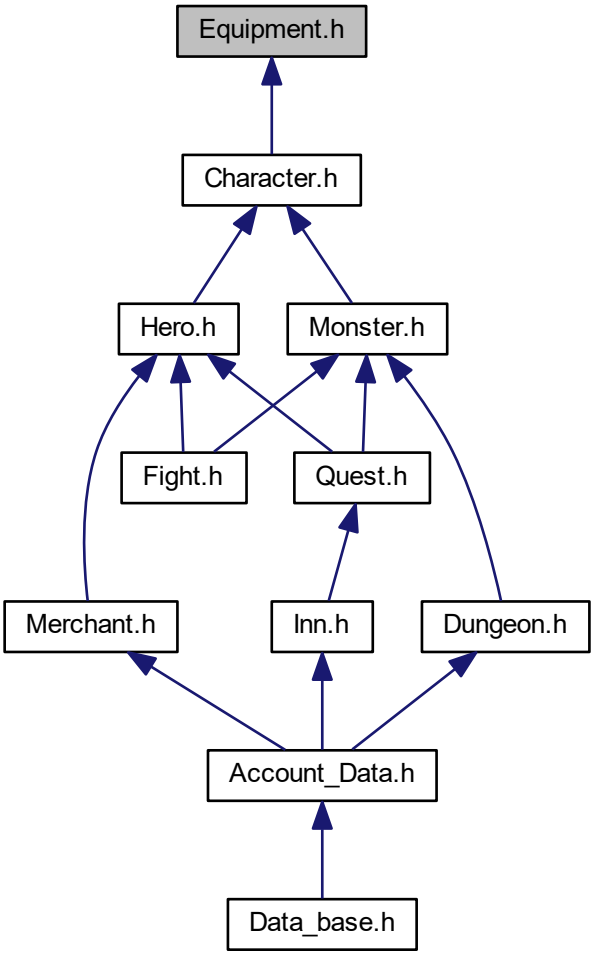
5.6 Dokumentacja pliku Equipment.h

```
#include "Weapon.h"
#include "Armor.h"
```

Wykres zależności załączania dla Equipment.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



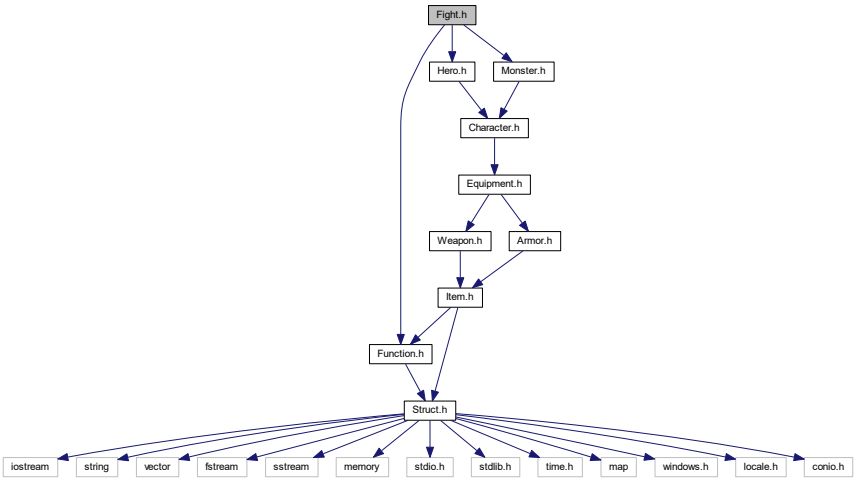
Komponenty

- class Equipment

5.7 Dokumentacja pliku Fight.h

```
#include "Function.h"
#include "Hero.h"
```

```
#include "Monster.h"
```
Wykres zależności załączania dla Fight.h:



Funkcje

- bool Fight (std::shared_ptr< Hero > v_Hero, std::shared_ptr< Monster > v_Monster, Options v_Settings)

5.7.1 Dokumentacja funkcji

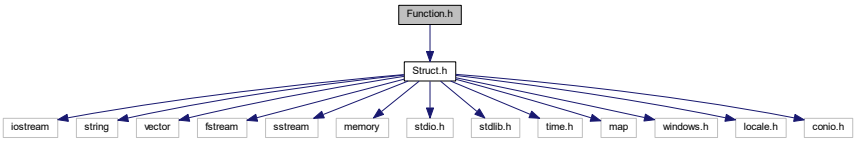
5.7.1.1 Fight()

```
bool Fight (
            std::shared_ptr< Hero > v_Hero,
            std::shared_ptr< Monster > v_Monster,
            Options v_Settings )
```

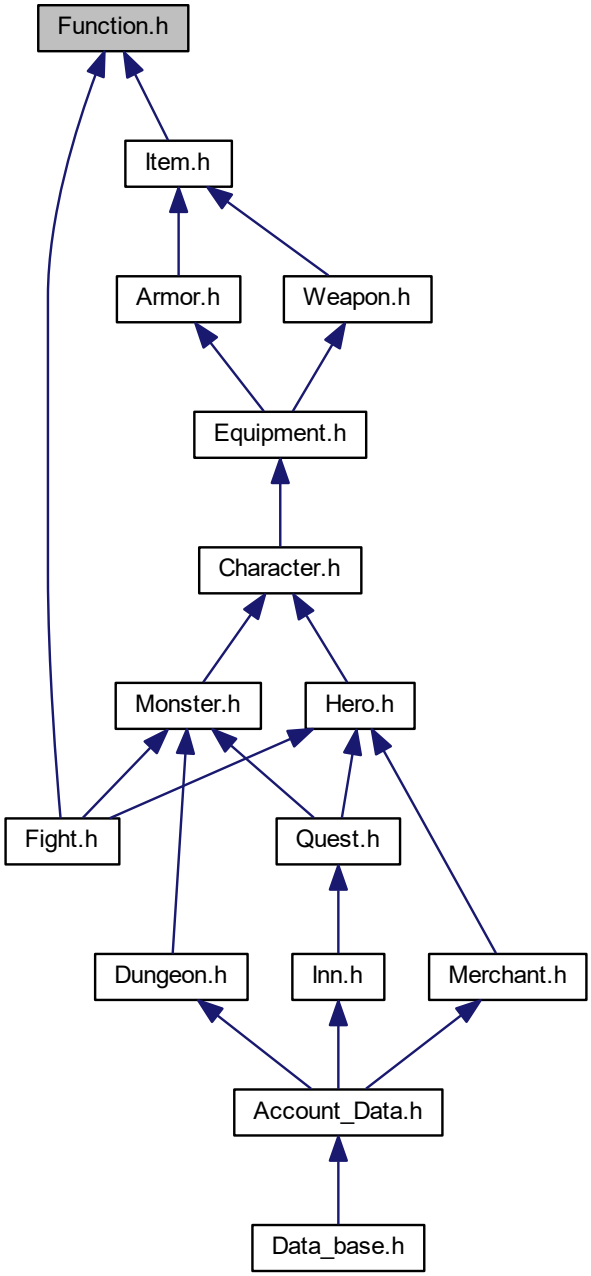
Funkcja odpowiedzialna z przeprowadzenie walki pomiędzy bohaterem i potworem. (bool)

5.8 Dokumentacja pliku Function.h

```
#include "Struct.h"
```
Wykres zależności załączania dla Function.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Funkcje

- std::string Random_Type ()
- std::string Random_Proffesion ()
- std::string Random_Race ()
- std::string Connect_Stats (Statistics v_Stats)
- std::vector< int > splitData (const std::string &s, char t_char)

5.8.1 Dokumentacja funkcji

5.8.1.1 Connect_Stats()

```
std::string Connect_Stats (
            Statistics v_Stats )
```

Funkcja łącząca statystyki w formę mozliwą do zapisu w bazie danych. (string)

5.8.1.2 Random_Proffesion()

```
std::string Random_Proffesion ( )
```

Funkcja generująca losową profesje. (string)

5.8.1.3 Random_Race()

```
std::string Random_Race ( )
```

Funkcja generująca losową rasę. (string)

5.8.1.4 Random_Type()

```
std::string Random_Type ( )
```

Funkcja generująca losowy typ (Przedmiotu). (string)

5.8.1.5 splitData()

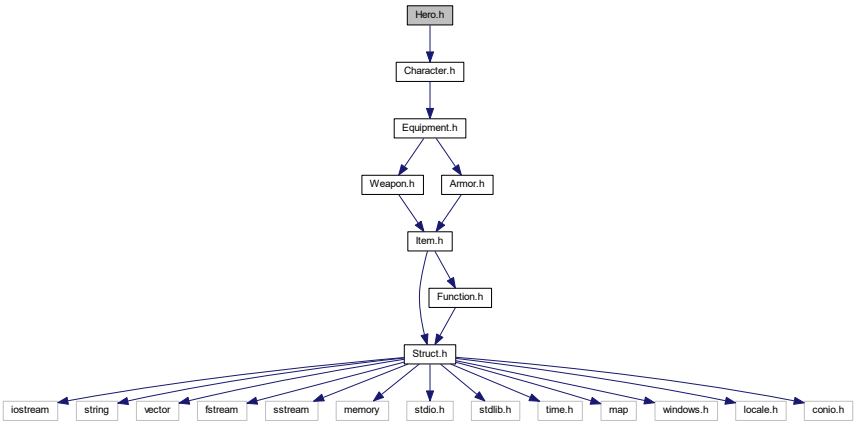
```
std::vector<int> splitData (
            const std::string & s,
            char t_char )
```

Funkcja rozdzielająca łańcuch z formy przechwywanej bazie danych. (vector<int>)

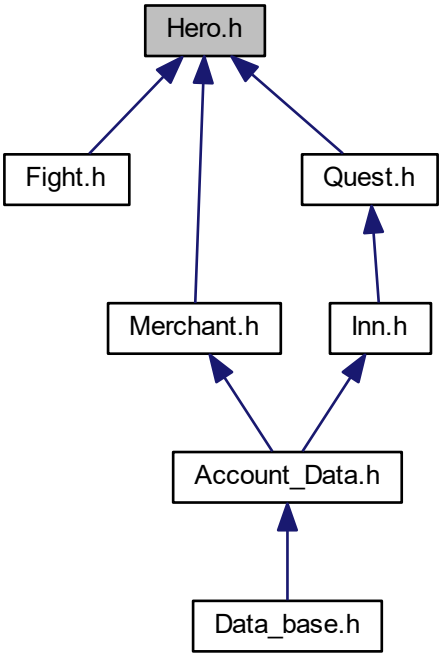
5.9 Dokumentacja pliku Hero.h

```
#include "Character.h"
```

Wykres zależności załączania dla Hero.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



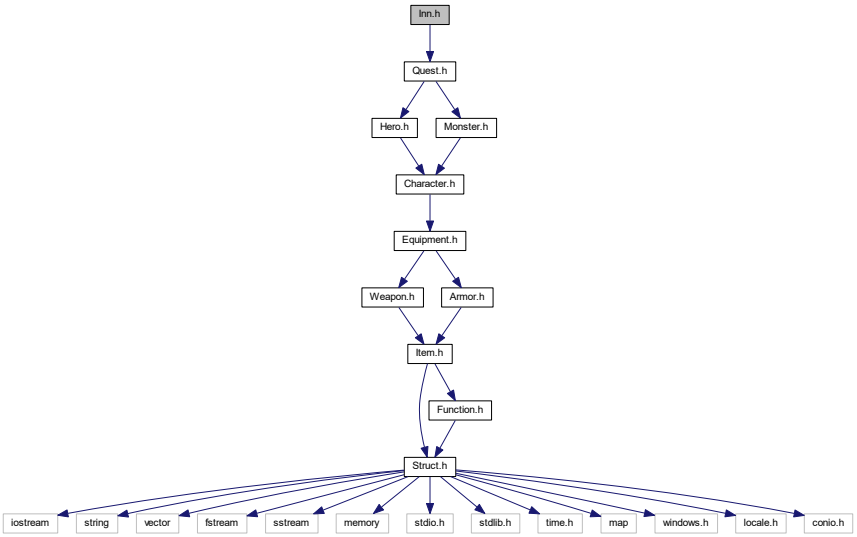
Komponenty

- class Hero

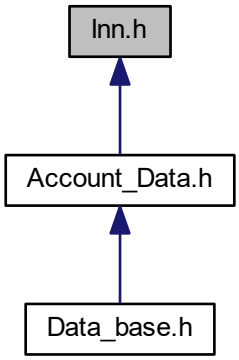
5.10 Dokumentacja pliku Inn.h

```
#include "Quest.h"
```

Wykres zależności załączania dla Inn.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



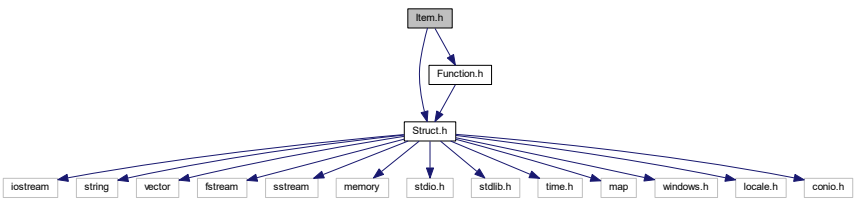
Komponenty

- class Inn

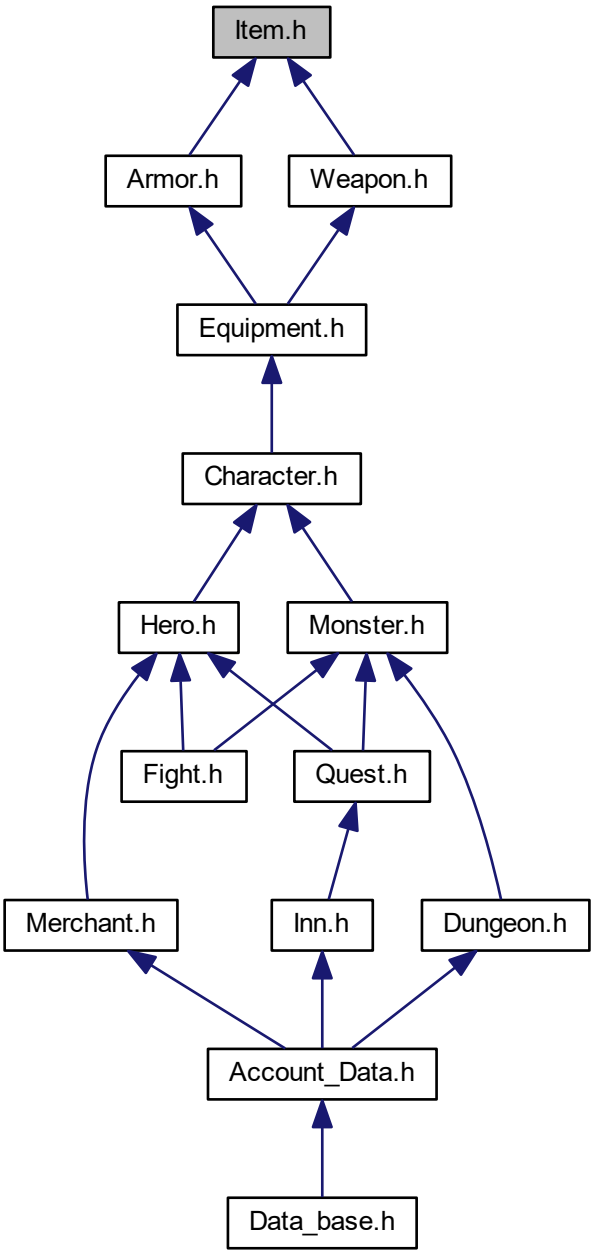
5.11 Dokumentacja pliku Item.h

```
#include "Struct.h"
#include "Function.h"
```

Wykres zależności załączania dla Item.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



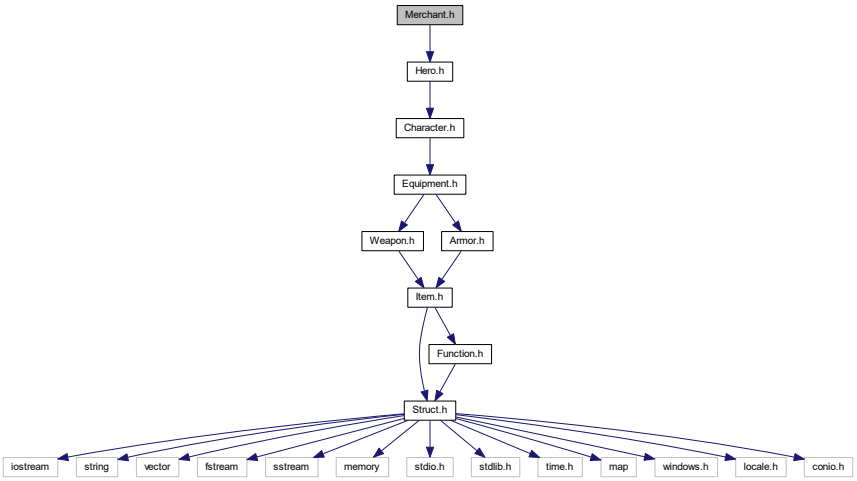
Komponenty

- class Item

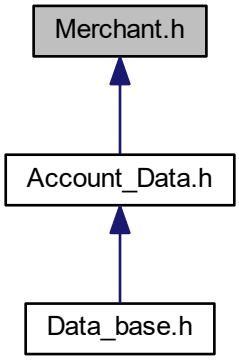
5.12 Dokumentacja pliku Merchant.h

```
#include "Hero.h"
```

Wykres zależności załączania dla Merchant.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



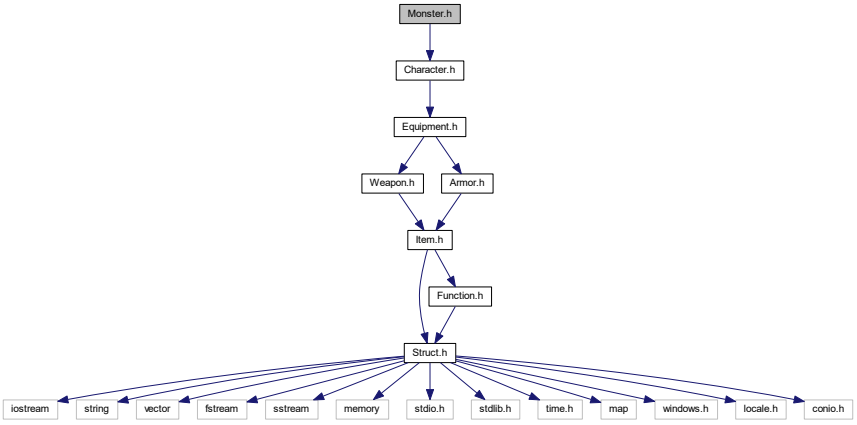
Komponenty

- class Merchant

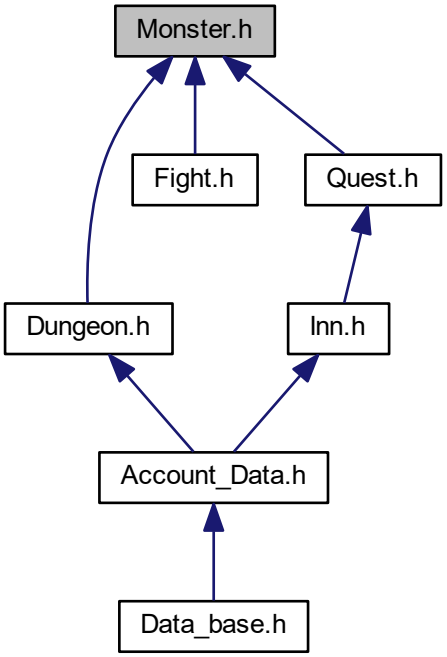
5.13 Dokumentacja pliku Monster.h

```
#include "Character.h"
```

Wykres zależności załączania dla Monster.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



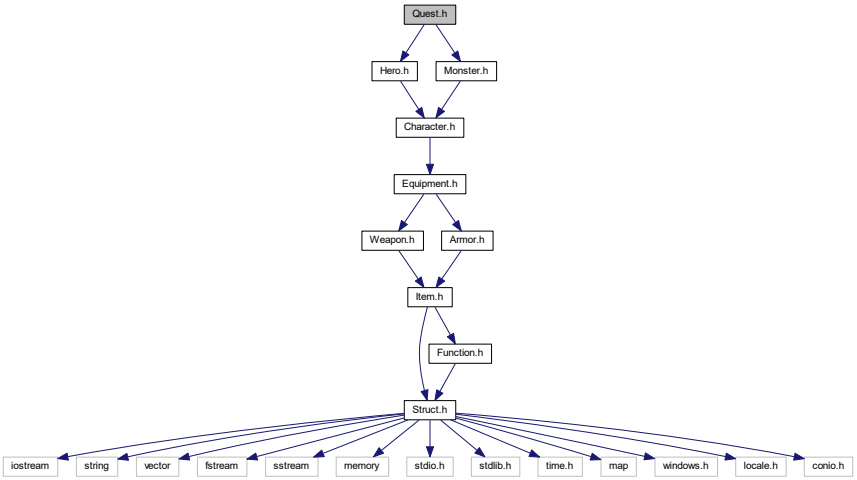
Komponenty

- class Monster

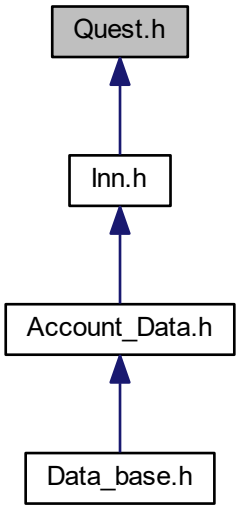
5.14 Dokumentacja pliku Quest.h

```
#include "Hero.h"
#include "Monster.h"
```

Wykres zależności załączania dla Quest.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



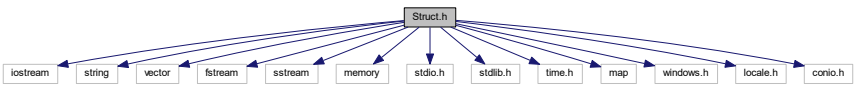
Komponenty

- class Quest

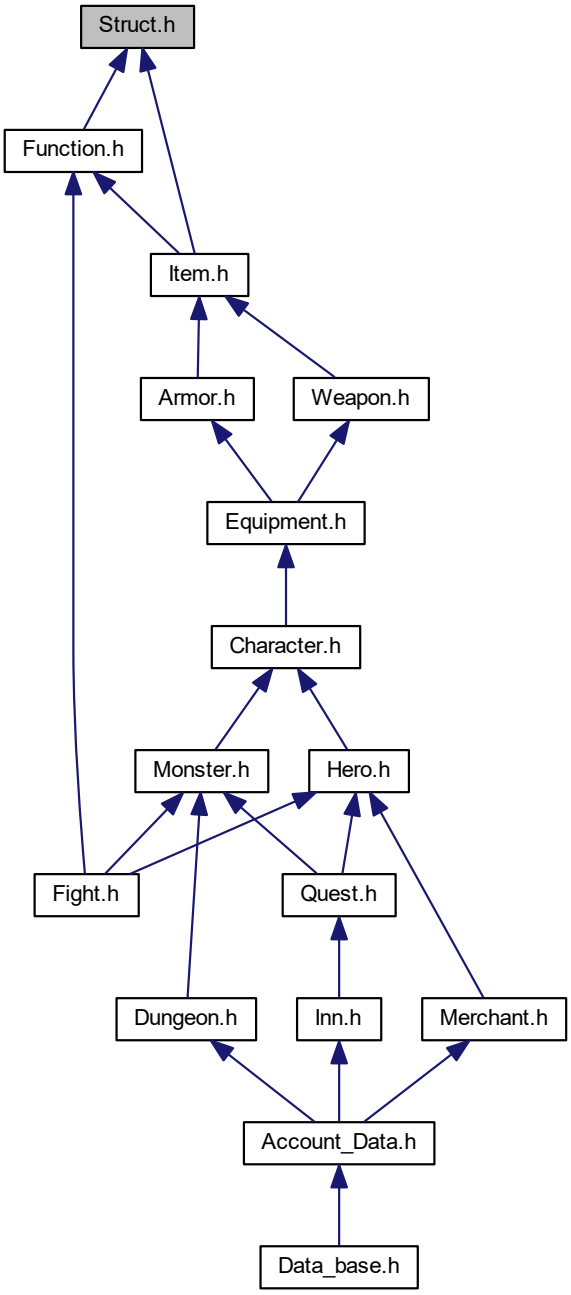
5.15 Dokumentacja pliku Struct.h

```
#include <iostream>
#include <string>
#include <vector>
#include <fstream>
#include <sstream>
#include <memory>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <map>
#include <windows.h>
#include <locale.h>
#include <conio.h>
```

Wykres zależności załączania dla Struct.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



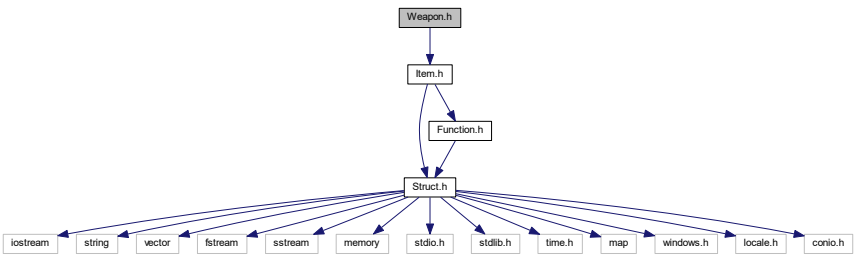
Komponenty

- struct Options
- struct Statistics

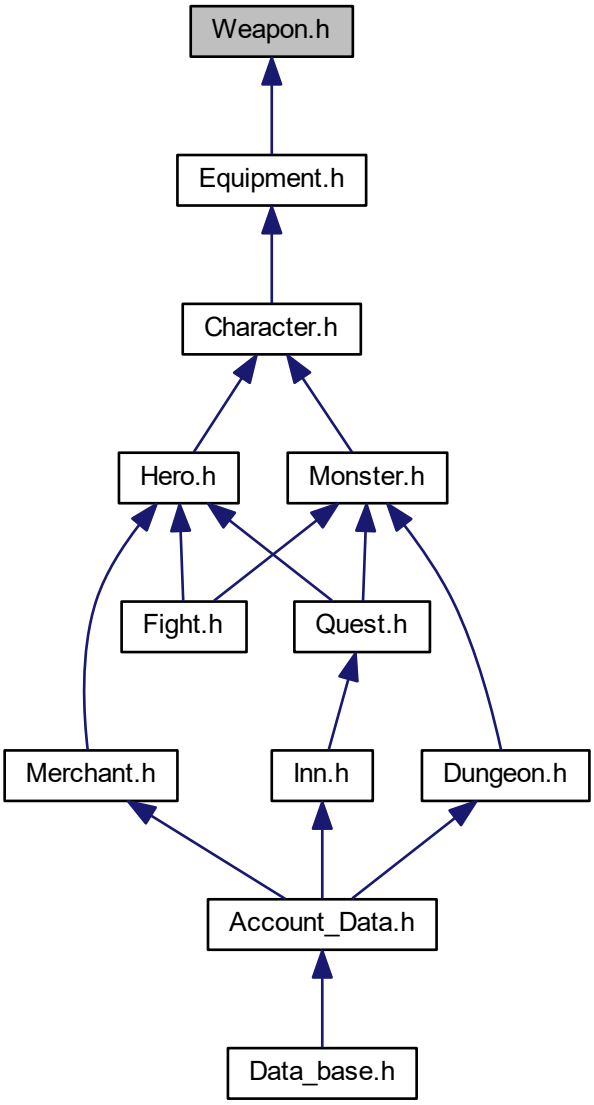
5.16 Dokumentacja pliku Weapon.h

```
#include "Item.h"
```

Wykres zależności załączania dla Weapon.h:



Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:



Komponenty

- class Weapon

Indeks