Gép Konfigurátor

Prog2HF

Generated by Andris Borbás with the help of Doxygen and a lot of RedBull

Nagy házi feladat specifikáció

A Feladat:

Számítógép konfigurátor

Készítsen C++ programot számítógép konfigok nyilvántartására! A program alkatrészekből építse fel a konfigokat. Elegendő 3-4 konfigot kezelnie, de legyen bővíthető. A programmal minimálisan a következő feladatokat kell ellátni:

- új alkatrészek bevitele az adatbázisba
- új konfig bevitele az adatbázisba
- adatbázis kiírása fájlba
- adatbázis beolvasása fájlból
- konfig törlése
- kiválasztott konfig alkatrészeinek listázása

Egyszerű felhasználói felületet tervezzen! A feladat lényege az objektumorientált megközelítés, ill. modellezés és nem a felhasználói felület szépsége. Használjon heterogén adatszerkezetet! A megoldáshoz ne használjon STL tárolót!

A program célja

A feladat egy olyan program készítése, amely számítógépeknek a konfigjait és az azok összerakásához szükséges alkatrészeket tárolja, jeleníti meg és módosítja. A program tárolhasson akárhány konfigot és alkatrészt, amiket bővíteni is lehessen. A program külön fájlban tárolja a konfigokat és alkatrészeket, amiket tudjon beolvasni és módosítani, bővíteni. Ezek mellett extra funkció lehet az összár kijelzése. A számítógépek csak PC k lehetnek. Az alkatrészek típusai:

- Alkatrész – gyártó, típus, ár
  - CPU – órajel, magok száma, socket
  - GPU – órajel, vram mérete
  - RAM – órajel, ram mérete
  - Alaplap – chipset, socket, form factor
  - ház – form factor
  - táp – teljesítmény
  - háttértár – méret, írási/olvasási sebesség
    - ssd – form factor, flash cellek száma (slc, mlc, tlc)
    - hdd – rpm

A program használata

A felhasználó a programot indítása után command-line (vagy a felhasználói felület(talán)) segítségével tudja használni. A programnak indítási paraméterként meg lehet adni a fájlok helyét.

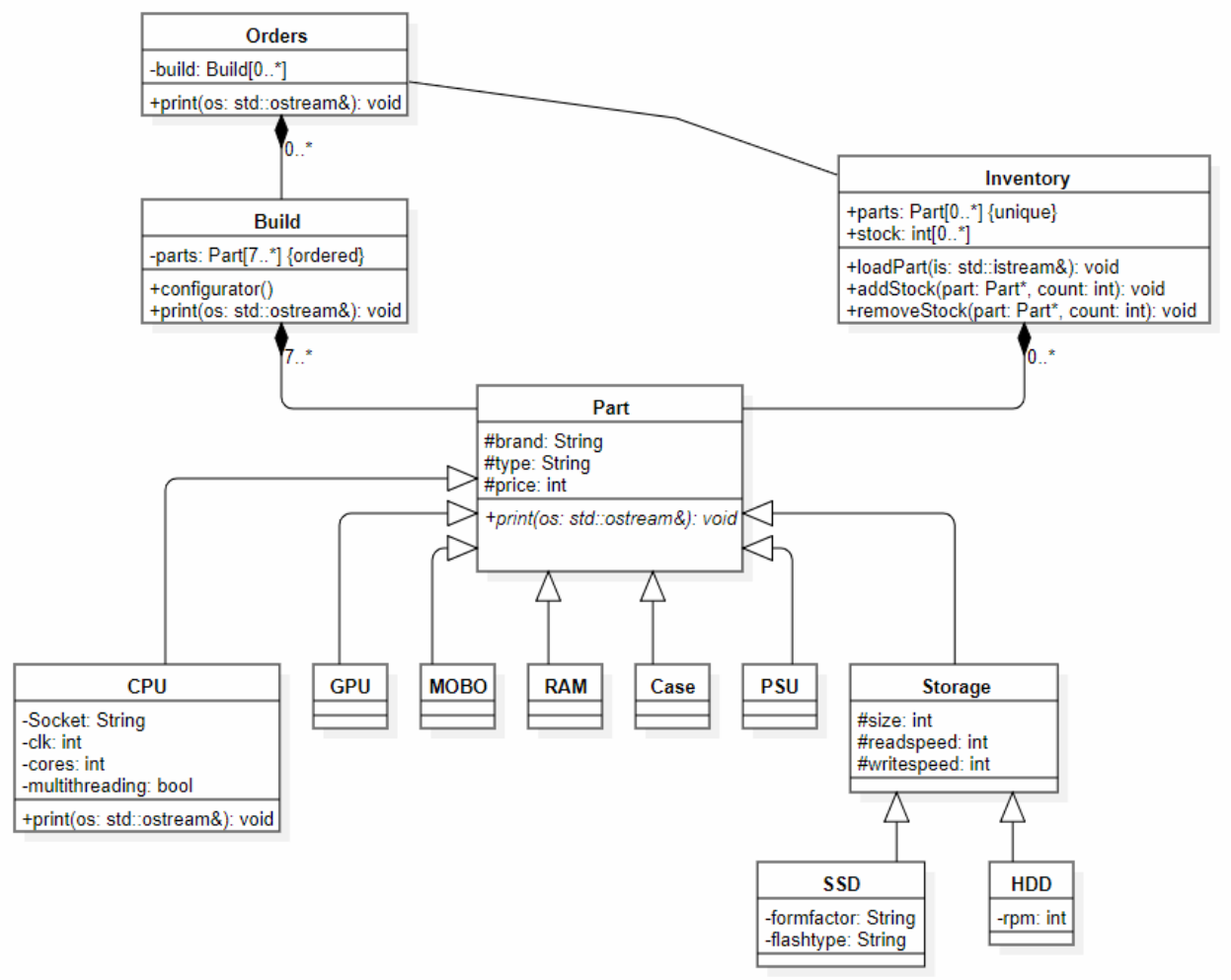
A command-line ban a menüpontok kiválasztásával lehet navigálni a különböző funkciók között. A program a futás végén a megadott fájlba menti az új konfigokat vagy beírja a változtatásokat.

A fájlban a paraméterek szó végi : után következnek.

```
Brand:
```

A futás eredménye

A program indítása után a főmenüben lehet választani a funkciók közül majd funkciónak megfelelően a program kiírja nekünk, amit választottunk vagy be lehet neki írni új konfigot. Ha a konfigban olyan alkatrész is van, amit a program még nem ismer akkor azt is eltárolja.



Használati utasítás:

A programot a menüpontok sorszámának beírásával lehet vezérelni, általában mindenhol megjelenik hogy mit kell nyomni a tovább vagy visszalépéshez. Ez az "1" szokott lenni, ahol nem az ott ki van írva.
Ha új konfigot akarunk felvenni akkor a megjelenő utasításokat kell követni, mindig kiválaszta az odaillő alkatrészt.
Ha új alkatrészt szeretnénk felvenni akkor ki kell választani az alkatrész típusát és utána megadni a paraméterket, vigyázva arra hogy ne legyen a szavakban szóköz és azokat az adatokat amiket számmal kell megadni csak számot írjunk be.

1 Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

2 Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

3 File Index

3.1 File List

Here is a list of all files with brief descriptions:

4 Class Documentation

4.1 Build Class Reference

Egy gépkonfigot tárol.

```
#include <Builds.h>
```

Public Member Functions

- Build (size_t capacity=7)
- ~Build ()
- template<typename T >
  void push_back (T *part)
  végére beszúrás
- int get_price ()
- void print (std::ostream &os) const
  konfig kiírása
- void load (std::fstream &is, Inventory &inventory, TempInput &tmp)
  konfig betöltése
- void save (std::ostream &os) const
  konfig mentése
- const Part * operator[ ] (int idx) const
- Part * operator[ ] (int idx)

4.1.1 Detailed Description

Egy gépkonfigot tárol.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Build()

```
Build::Build (
            size_t capacity = 7 )  [inline]
```

4.1.2.2 ~Build()

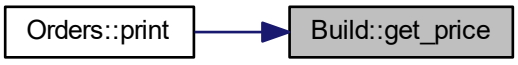
```
Build::~Build ( )  [inline]
```

4.1.3 Member Function Documentation

4.1.3.1 get_price()

```
int Build::get_price ( )  [inline]
```

Here is the caller graph for this function:

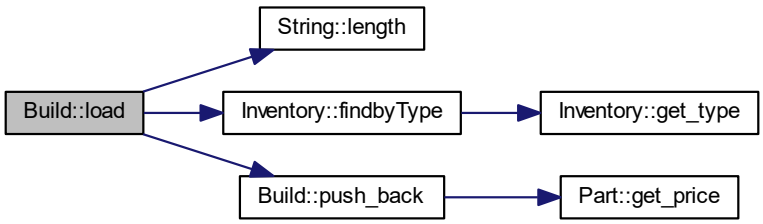


4.1.3.2 load()

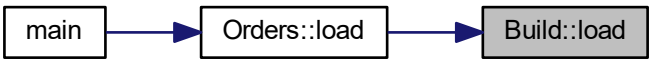
```
void Build::load (
            std::fstream & is,
            Inventory & inventory,
            TempInput & tmp )
```

konfig betöltése

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.3 operator[]() [1/2]

```
const Part* Build::operator[] (
            int idx ) const  [inline]
```

4.1.3.4 operator[]() [2/2]

```
Part* Build::operator[] (
            int idx )  [inline]
```

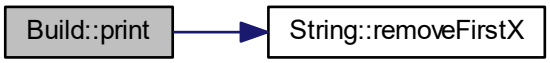
4.1.3.5 print()

```
void Build::print (
            std::ostream & os ) const
```

konfig kiírása

class neve

class szó levétele a class neve elől Here is the call graph for this function:



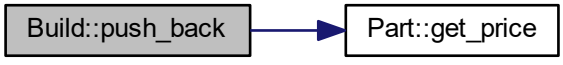
Here is the caller graph for this function:

4.1.3.6 push_back()

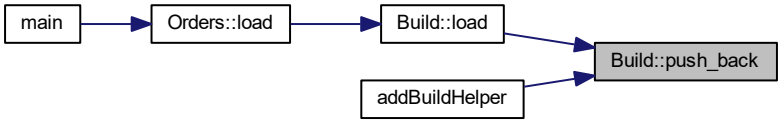
```
template<typename T >
void Build::push_back (
            T * part )  [inline]
```

végére beszúrás

Here is the call graph for this function:



Here is the caller graph for this function:



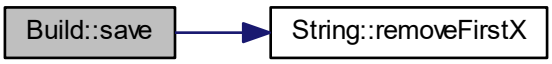
4.1.3.7 save()

```
void Build::save (
            std::ostream & os ) const
```

konfig mentése

class neve

class szó levétele a class neve elől Here is the call graph for this function:



The documentation for this class was generated from the following files:

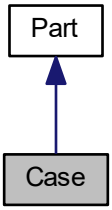
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Builds.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Builds.cpp

4.2 Case Class Reference

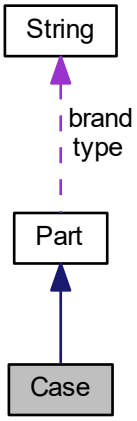
Ház.

```
#include <Parts.h>
```

Inheritance diagram for Case:



Collaboration diagram for Case:



Public Member Functions

- Case (String brand, String type, int price, String formfactor)
- Case (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.2.1 Detailed Description

Ház.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Case() [1/2]

```
Case::Case (
            String brand,
            String type,
            int price,
            String formfactor )  [inline], [explicit]
```

4.2.2.2 Case() [2/2]

```
Case::Case (
            TempInput & tmp )  [inline], [explicit]
```

4.2.3 Member Function Documentation

4.2.3.1 print() [1/4]

```
void Case::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

Here is the caller graph for this function:

4.2.3.2 print() [2/4]

```
void Case::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.2.3.3 print() [3/4]

```
void Case::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.2.3.4 print() [4/4]

```
void Case::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.3 CompatibilityList Class Reference

`#include <Compatibility.h>`

Public Member Functions

- CompatibilityList ()
- CompatibilityList (String &)
- ~CompatibilityList ()
- int get_length () const
- String * get_listp () const
- void addItems (String &)
- bool operator== (String &rhs)
- bool operator== (const char *rhs)

4.3.1 Constructor & Destructor Documentation

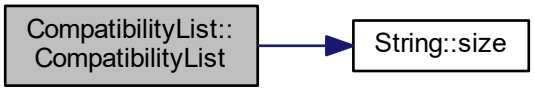
4.3.1.1 CompatibilityList() [1/2]

`CompatibilityList::CompatibilityList ( )  [inline], [explicit]`

4.3.1.2 CompatibilityList() [2/2]

```
CompatibilityList::CompatibilityList (
            String & slist )  [explicit]
```

Here is the call graph for this function:



4.3.1.3 ~CompatibilityList()

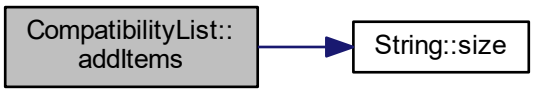
```
CompatibilityList::~CompatibilityList ( )  [inline]
```

4.3.2 Member Function Documentation

4.3.2.1 addItems()

```
void CompatibilityList::addItems (
            String & slist )
```

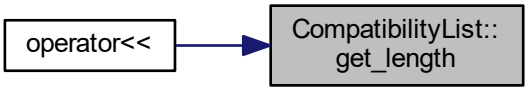
Here is the call graph for this function:



4.3.2.2 get_length()

```
int CompatibilityList::get_length ( ) const  [inline]
```

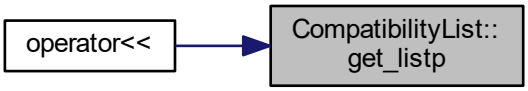
Here is the caller graph for this function:



4.3.2.3 get_listp()

```
String* CompatibilityList::get_listp ( ) const  [inline]
```

Here is the caller graph for this function:



4.3.2.4 operator==() [1/2]

```
bool CompatibilityList::operator== (
            String & rhs )  [inline]
```

4.3.2.5 operator==() [2/2]

```
bool CompatibilityList::operator== (
            const char * rhs )  [inline]
```

The documentation for this class was generated from the following files:

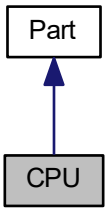
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Compatibility.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Compatibility.cpp

4.4 CPU Class Reference

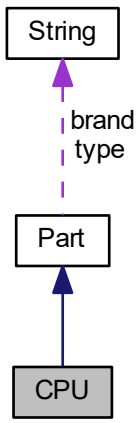
Processzor.

```
#include <Parts.h>
```

Inheritance diagram for CPU:



Collaboration diagram for CPU:



Public Member Functions

- CPU (String brand, String type, int price, int clk, int cores, String socket, bool multithreading)
- CPU (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.4.1 Detailed Description

Processzor.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 CPU() [1/2]

```
CPU::CPU (
            String brand,
            String type,
            int price,
            int clk,
            int cores,
            String socket,
            bool multithreading )  [inline], [explicit]
```

4.4.2.2 CPU() [2/2]

```
CPU::CPU (
            TempInput & tmp )  [inline], [explicit]
```

4.4.3 Member Function Documentation

4.4.3.1 print() [1/4]

```
void CPU::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

Here is the caller graph for this function:

4.4.3.2 print() [2/4]

```
void CPU::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.4.3.3 print() [3/4]

```
void CPU::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.4.3.4 print() [4/4]

```
void CPU::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

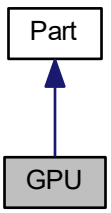
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.5 GPU Class Reference

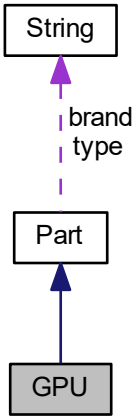
Videókártya.

```
#include <Parts.h>
```

Inheritance diagram for GPU:



Collaboration diagram for GPU:



Public Member Functions

- GPU (String brand, String type, int price, int clk, int vram)
- GPU (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.5.1 Detailed Description

Videókártya.

4.5.2 Constructor & Destructor Documentation

4.5.2.1 GPU() [1/2]

```
GPU::GPU (
            String brand,
            String type,
            int price,
            int clk,
            int vram )  [inline], [explicit]
```

4.5.2.2 GPU() [2/2]

```
GPU::GPU (
            TempInput & tmp )  [inline], [explicit]
```

4.5.3 Member Function Documentation

4.5.3.1 print() [1/4]

```
void GPU::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

Here is the caller graph for this function:

4.5.3.2 print() [2/4]

```
void GPU::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.5.3.3 print() [3/4]

```
void GPU::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.5.3.4 print() [4/4]

```
void GPU::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

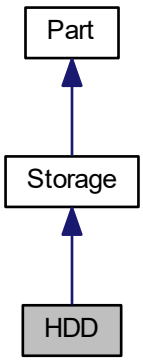
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.6 HDD Class Reference

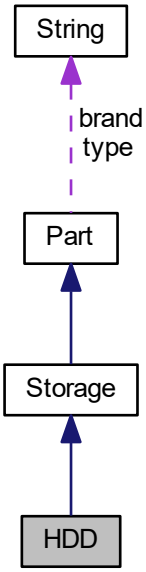
Merevlemez.

`#include <Parts.h>`

Inheritance diagram for HDD:



Collaboration diagram for HDD:



Public Member Functions

- HDD (String brand, String type, int price, int size, int readspeed, int writespeed, int rpm)
- HDD (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.6.1 Detailed Description

Merevlemez.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 HDD() [1/2]

```
HDD::HDD (
            String brand,
            String type,
            int price,
            int size,
            int readspeed,
            int writespeed,
            int rpm )  [inline], [explicit]
```

4.6.2.2 HDD() [2/2]

```
HDD::HDD (
            TempInput & tmp )  [inline], [explicit]
```

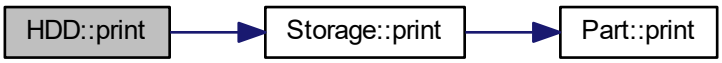
4.6.3 Member Function Documentation

4.6.3.1 print() [1/4]

```
void HDD::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:



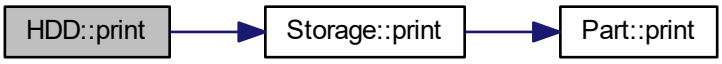
Here is the caller graph for this function:

4.6.3.2 print() [2/4]

```
void HDD::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:



4.6.3.3 print() [3/4]

```
void HDD::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:

4.6.3.4 print() [4/4]

```
void HDD::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.7 Inventory Class Reference

Alkatrész tároló

```
#include <Inventory.h>
```

Public Member Functions

- Inventory (size_t capacity=1)
- ~Inventory ()
- int get_size ()
- String get_type (int i)
- void loadPart (std::fstream &is, TempInput &tmp, enumPart)

  Betölt egy alkatrészt fájlból.
- void loadPart (std::istream &is, TempInput &tmp, enumPart)

  Betölt egy alkatrészt terminalból.
- void save (std::ostream &os)

  Raktár mentése egy streamre.
- void print (std::ostream &os, const String &test="-1")

  Raktár kiírása egy streamre.
- void remove (int idx)

  Egy alkatrész kitörlése a raktárból.
- int findbyType (const String &s0) const

  Megkeres egy alkatrészt a típusa alapján és visszaadja az indexét.
- const String & findbyIndex (int idx) const

  Megkeres egy alkatrészt index alapján és visszaadja a típusát.
- template<typename T >
  void push_back (T *part, String type)

  Egy alkatrész hozzáadása a raktárhoz.
- const Part * operator[ ] (int idx) const
- Part * operator[ ] (int idx)

4.7.1 Detailed Description

Alkatrész tároló

4.7.2 Constructor & Destructor Documentation

4.7.2.1 Inventory()

```
Inventory::Inventory (
            size_t capacity = 1 )  [inline]
```

4.7.2.2 ~Inventory()

```
Inventory::~Inventory ( )  [inline]
```

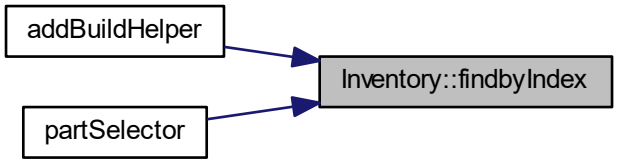
4.7.3 Member Function Documentation

4.7.3.1 findbyIndex()

```
const String & Inventory::findbyIndex (
            int idx ) const
```

Megkeres egy alkatrészt index alapján és visszaadja a típusát.

Here is the caller graph for this function:



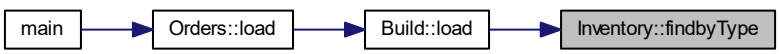
4.7.3.2 findbyType()

```
int Inventory::findbyType (
            const String & s0 ) const
```

Megkeres egy alkatrészt a típusa alapján és visszaadja az indexét.

Here is the call graph for this function:

Here is the caller graph for this function:



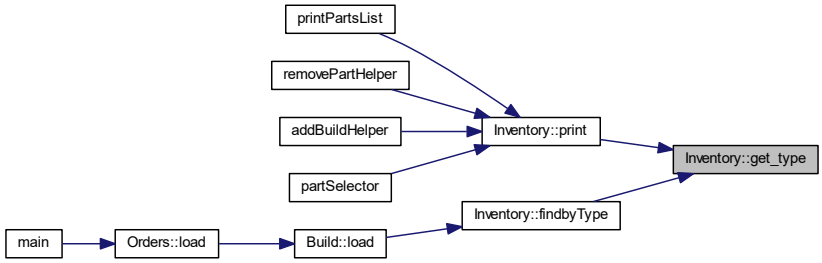
4.7.3.3 get_size()

```
int Inventory::get_size ( )  [inline]
```

4.7.3.4 get_type()

```
String Inventory::get_type (
            int i )  [inline]
```

Here is the caller graph for this function:

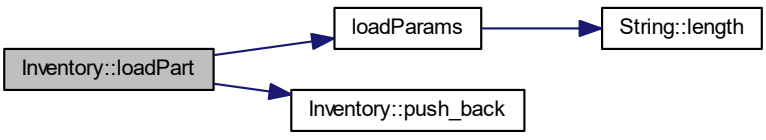


4.7.3.5 loadPart() [1/2]

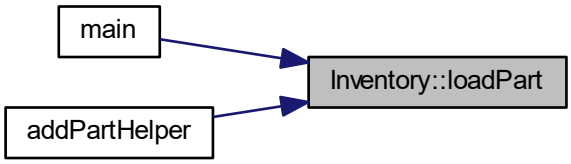
```
void Inventory::loadPart (
            std::fstream & is,
            TempInput & tmp,
            enumPart e )
```

Betölt egy alkatrészt fájlból.

Here is the call graph for this function:



Here is the caller graph for this function:

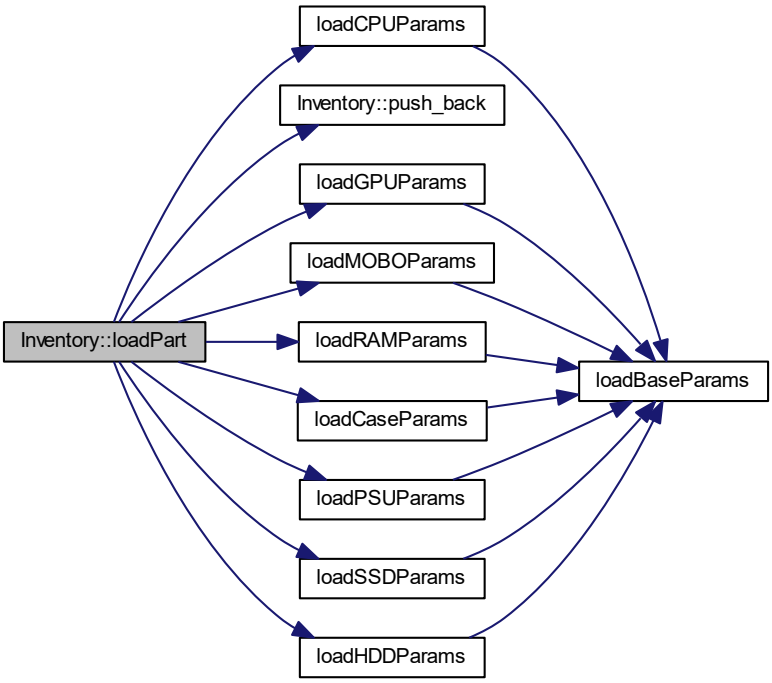


4.7.3.6 loadPart() [2/2]

```
void Inventory::loadPart (
            std::istream & is,
            TempInput & tmp,
            enumPart e )
```

Betölt egy alkatrészt terminalból.

Here is the call graph for this function:



4.7.3.7 operator[]() [1/2]

```
const Part* Inventory::operator[] (
            int idx ) const  [inline]
```

4.7.3.8 operator[]() [2/2]

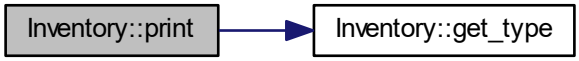
```
Part* Inventory::operator[] (
            int idx )  [inline]
```

4.7.3.9 print()

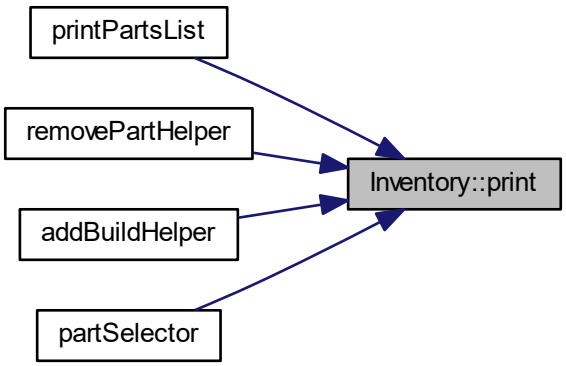
```
void Inventory::print (
            std::ostream & os,
            const String & test = "-1" )
```

Raktár kiírása egy streamre.

Here is the call graph for this function:



Here is the caller graph for this function:

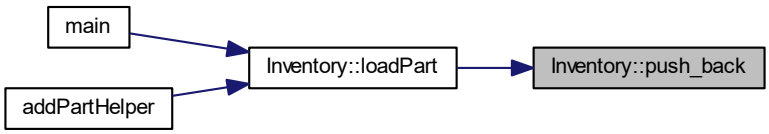


4.7.3.10 push_back()

```
template<typename T >
void Inventory::push_back (
             T * part,
             String type )
```

Egy alkatrész hozzáadása a raktárhoz.

Here is the caller graph for this function:



4.7.3.11 remove()

```
void Inventory::remove (
             int idx )
```

Egy alkatrész kitörlése a raktárból.

Here is the caller graph for this function:

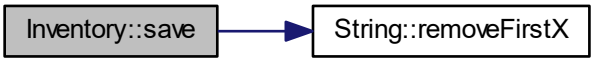
4.7.3.12 save()

```
void Inventory::save (
            std::ostream & os )
```

Raktár mentése egy streamre.

class neve

class szó levétele a class neve elől Here is the call graph for this function:



The documentation for this class was generated from the following files:

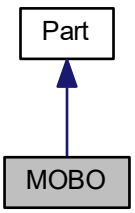
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Inventory.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Inventory.cpp

4.8 MOBO Class Reference

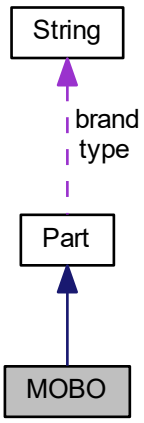
Alaplap.

```
#include <Parts.h>
```

Inheritance diagram for MOBO:



Collaboration diagram for MOBO:



Public Member Functions

- MOBO (String brand, String type, int price, String socket, String chipset, String formfactor)
- MOBO (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.8.1 Detailed Description

Alaplap.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 MOBO() [1/2]

```
MOBO::MOBO (
            String brand,
            String type,
            int price,
            String socket,
            String chipset,
            String formfactor )  [inline], [explicit]
```

4.8.2.2 MOBO() [2/2]

```
MOBO::MOBO (
            TempInput & tmp )  [inline], [explicit]
```

4.8.3 Member Function Documentation

4.8.3.1 print() [1/4]

```
void MOBO::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

Here is the caller graph for this function:

4.8.3.2 print() [2/4]

```
void MOBO::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.8.3.3 print() [3/4]

```
void MOBO::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.8.3.4 print() [4/4]

```
void MOBO::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.9 Orders Class Reference

A megrendelt konfigokat tárolja.

`#include <Builds.h>`

Public Member Functions

- Orders (size_t capacity=1)
- ~Orders ()
- int get_size ()
- void push_back (Build *build)
  végére beszúrás
- void load (std::fstream &is, Inventory &inventory, TempInput &tmp)
  megrendelések betöltése
- void save (std::ostream &os) const
  megrendelések mentése
- void complete (int idx)
  megrendelés teljesített állapotba tétele
- void remove (int idx)
  megrendelés törlése
- void print (std::ostream &os) const
- void print (simple_ostream &tos) const
- const Build * operator[ ] (int idx) const
- Build * operator[ ] (int idx)

4.9.1 Detailed Description

A megrendelt konfigokat tárolja.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 Orders()

```
Orders::Orders (
            size_t capacity = 1 )  [inline]
```

4.9.2.2 ~Orders()

```
Orders::~Orders ( )  [inline]
```

4.9.3 Member Function Documentation

4.9.3.1 complete()

```
void Orders::complete (
            int idx )
```

megrendelés teljesített állapotba tétele

4.9.3.2 get_size()

```
int Orders::get_size ( )  [inline]
```

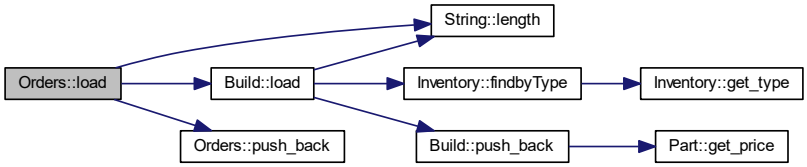
Here is the caller graph for this function:

4.9.3.3 load()

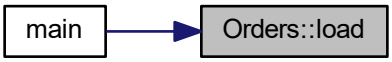
```
void Orders::load (
            std::fstream & is,
            Inventory & inventory,
            TempInput & tmp )
```

megrendelések betöltése

Here is the call graph for this function:



Here is the caller graph for this function:



4.9.3.4 operator[]() [1/2]

```
const Build* Orders::operator[] (
            int idx ) const  [inline]
```

4.9.3.5 operator[]() [2/2]

```
Build* Orders::operator[] (
            int idx )  [inline]
```

4.9.3.6 print() [1/2]

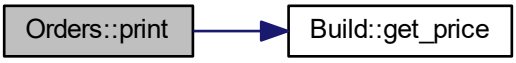
```
void Orders::print (
            std::ostream & os ) const
```

Here is the caller graph for this function:

4.9.3.7 print() [2/2]

```
void Orders::print (
            simple_ostream & tos ) const
```

Here is the call graph for this function:

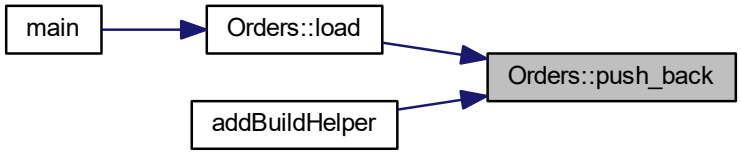


4.9.3.8 push_back()

```
void Orders::push_back (
            Build * build )  [inline]
```

végére beszúrás

Here is the caller graph for this function:



4.9.3.9 remove()

```
void Orders::remove (
            int idx )
```

megrendelés törlése

4.9.3.10 save()

```
void Orders::save (
            std::ostream & os ) const
```

megrendelések mentése

The documentation for this class was generated from the following files:

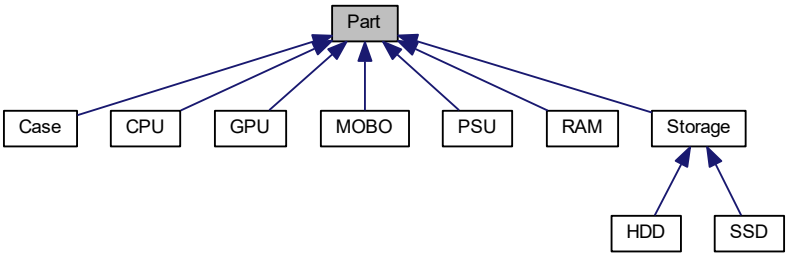
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Builds.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Builds.cpp

4.10 Part Class Reference

Alap alkatrész típus.

`#include <Parts.h>`

Inheritance diagram for Part:



Collaboration diagram for Part:

Public Member Functions

- Part (String brand="", String type="", int price=0)
- virtual ~Part ()
- virtual int get_price ()
- virtual String get_type ()
- virtual void print (std::ostream &os) const
- virtual void print (utos_ostream &tos) const
- virtual void print (simple_ostream &tos) const
- virtual void print (typ_ostream &tos) const

Protected Attributes

- String brand
  Gyártó
- String type
  Típus.
- int price
  Ár.

4.10.1 Detailed Description

Alap alkatrész típus.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 Part()

```
Part::Part (
            String brand = "",
            String type = "",
            int price = 0 )  [inline]
```

4.10.2.2 ~Part()

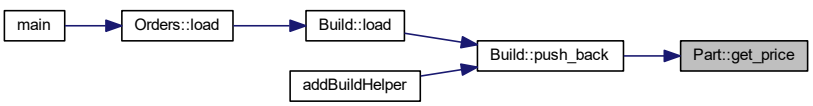
```
virtual Part::~Part ( )  [inline], [virtual]
```

4.10.3 Member Function Documentation

4.10.3.1 get_price()

```
virtual int Part::get_price ( )  [inline], [virtual]
```

Here is the caller graph for this function:



4.10.3.2 get_type()

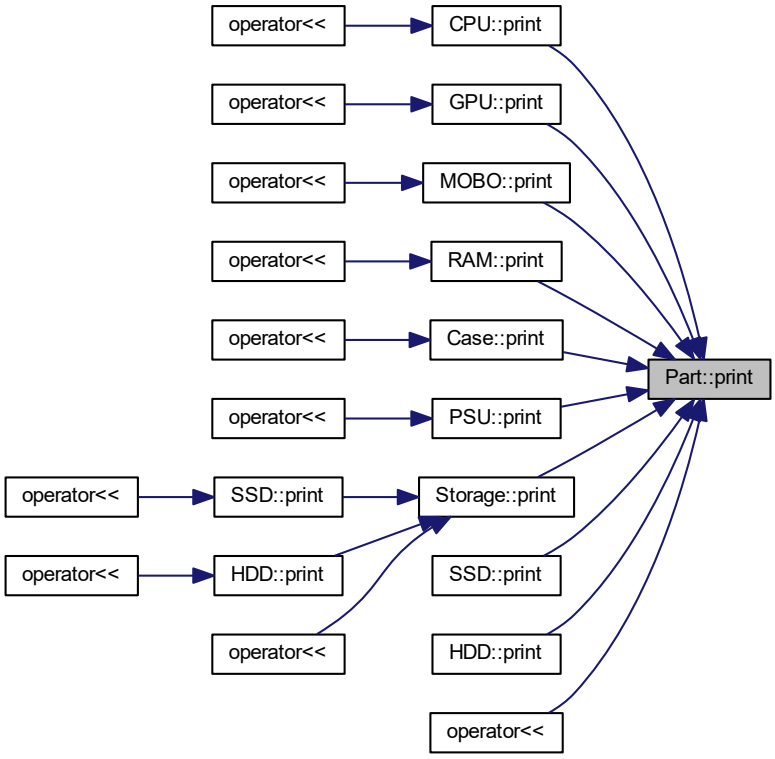
```
virtual String Part::get_type ( )  [inline], [virtual]
```

4.10.3.3 print() [1/4]

```
void Part::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented in HDD, SSD, Storage, PSU, Case, RAM, MOBO, GPU, and CPU.

Here is the caller graph for this function:



4.10.3.4 print() [2/4]

```
void Part::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented in HDD, SSD, Storage, PSU, Case, RAM, MOBO, GPU, and CPU.

4.10.3.5 print() [3/4]

```
void Part::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented in HDD, SSD, Storage, PSU, Case, RAM, MOBO, GPU, and CPU.

4.10.3.6 print() [4/4]

```
void Part::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented in HDD, SSD, Storage, PSU, Case, RAM, MOBO, GPU, and CPU.

4.10.4 Member Data Documentation

4.10.4.1 brand

```
String Part::brand  [protected]
```

Gyártó

4.10.4.2 price

```
int Part::price  [protected]
```

Ár.

4.10.4.3 type

```
String Part::type  [protected]
```

Típus.

The documentation for this class was generated from the following files:

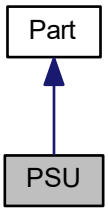
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.11 PSU Class Reference

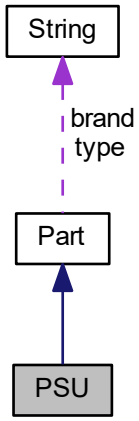
Táp.

`#include <Parts.h>`

Inheritance diagram for PSU:



Collaboration diagram for PSU:



Public Member Functions

- PSU (String brand, String type, int price, int wattage)
- PSU (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.11.1 Detailed Description

Táp.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 PSU() [1/2]

```
PSU::PSU (
            String brand,
            String type,
            int price,
            int wattage )  [inline], [explicit]
```

4.11.2.2 PSU() [2/2]

```
PSU::PSU (
            TempInput & tmp )  [inline], [explicit]
```

4.11.3 Member Function Documentation

4.11.3.1 print() [1/4]

```
void PSU::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

Here is the caller graph for this function:

4.11.3.2 print() [2/4]

```
void PSU::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.11.3.3 print() [3/4]

```
void PSU::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.11.3.4 print() [4/4]

```
void PSU::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

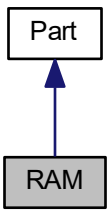
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.12 RAM Class Reference

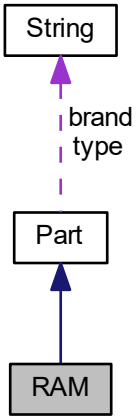
Memória.

```
#include <Parts.h>
```

Inheritance diagram for RAM:



Collaboration diagram for RAM:



Public Member Functions

- RAM (String brand, String type, int price, int clk, int size)
- RAM (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.12.1 Detailed Description

Memória.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 RAM() [1/2]

```
RAM::RAM (
            String brand,
            String type,
            int price,
            int clk,
            int size )  [inline], [explicit]
```

4.12.2.2 RAM() [2/2]

```
RAM::RAM (
            TempInput & tmp )  [inline], [explicit]
```

4.12.3 Member Function Documentation

4.12.3.1 print() [1/4]

```
void RAM::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

Here is the caller graph for this function:

4.12.3.2 print() [2/4]

```
void RAM::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.12.3.3 print() [3/4]

```
void RAM::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

4.12.3.4 print() [4/4]

```
void RAM::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.13 simple_ostream Struct Reference

csak paraméter stream manipulator

```
#include <schtring.hpp>
```

Public Attributes

- std::ostream & os

4.13.1 Detailed Description

csak paraméter stream manipulator

4.13.2 Member Data Documentation

4.13.2.1 os

```
std::ostream& simple_ostream::os
```

The documentation for this struct was generated from the following file:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp

4.14 simple_t Struct Reference

csak paraméter toggle

```
#include <schtring.hpp>
```

4.14.1 Detailed Description

csak paraméter toggle

The documentation for this struct was generated from the following file:

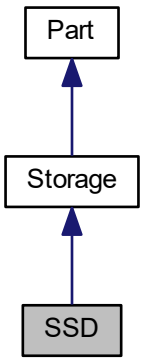
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp

4.15 SSD Class Reference

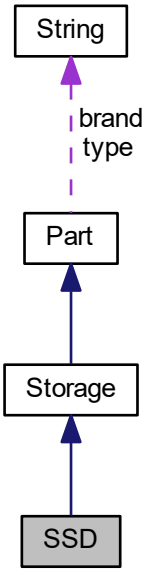
SSD.

```
#include <Parts.h>
```

Inheritance diagram for SSD:



Collaboration diagram for SSD:



Public Member Functions

- SSD (String brand, String type, int price, int size, int readspeed, int writespeed, String formfactor, String flashtype)
- SSD (TempInput &tmp)
- void print (std::ostream &os) const
- void print (utos_ostream &tos) const
- void print (simple_ostream &tos) const
- void print (typ_ostream &tos) const

Additional Inherited Members

4.15.1 Detailed Description

SSD.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 SSD() [1/2]

```
SSD::SSD (
            String brand,
            String type,
            int price,
            int size,
            int readspeed,
            int writespeed,
            String formfactor,
            String flashtype )  [inline], [explicit]
```

4.15.2.2 SSD() [2/2]

```
SSD::SSD (
            TempInput & tmp )  [inline], [explicit]
```

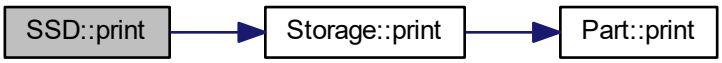
4.15.3 Member Function Documentation

4.15.3.1 print() [1/4]

```
void SSD::print (
            std::ostream & os ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:



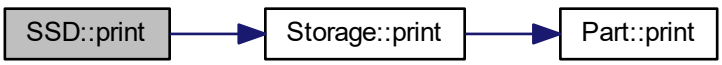
Here is the caller graph for this function:

4.15.3.2 print() [2/4]

```
void SSD::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:



4.15.3.3 print() [3/4]

```
void SSD::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:

4.15.3.4 print() [4/4]

```
void SSD::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Storage.

Here is the call graph for this function:

The documentation for this class was generated from the following files:

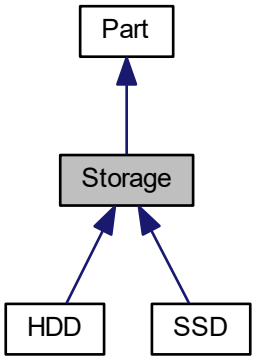
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.16 Storage Class Reference

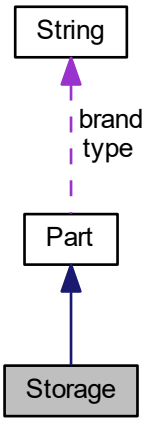
Tárhely alap.

`#include <Parts.h>`

Inheritance diagram for Storage:



Collaboration diagram for Storage:



Public Member Functions

- Storage (String brand, String type, int price, int size, int readspeed, int writespeed)
- virtual void print (std::ostream &os) const
- virtual void print (utos_ostream &tos) const
- virtual void print (simple_ostream &tos) const
- virtual void print (typ_ostream &tos) const

Protected Attributes

- int size
  Méret.
- int readspeed
  Olvasási sebesség.
- int writespeed
  Írási sebesség.

4.16.1 Detailed Description

Tárhely alap.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 Storage()

```
Storage::Storage (
            String brand,
            String type,
            int price,
            int size,
            int readspeed,
            int writespeed )  [inline], [explicit]
```

4.16.3 Member Function Documentation

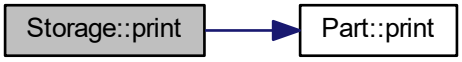
4.16.3.1 print() [1/4]

```
void Storage::print (
            std::ostream & os ) const  [virtual]
```

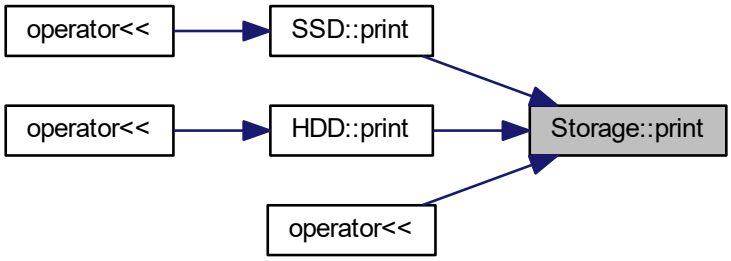
Reimplemented from Part.

Reimplemented in HDD, and SSD.

Here is the call graph for this function:



Here is the caller graph for this function:



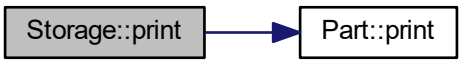
4.16.3.2 print() [2/4]

```
void Storage::print (
            utos_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Reimplemented in HDD, and SSD.

Here is the call graph for this function:



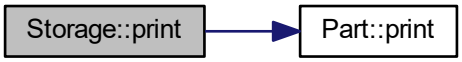
4.16.3.3 print() [3/4]

```
void Storage::print (
            simple_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Reimplemented in HDD, and SSD.

Here is the call graph for this function:



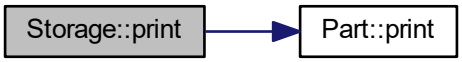
4.16.3.4 print() [4/4]

```
void Storage::print (
            typ_ostream & tos ) const  [virtual]
```

Reimplemented from Part.

Reimplemented in HDD, and SSD.

Here is the call graph for this function:



4.16.4 Member Data Documentation

4.16.4.1 readspeed

```
int Storage::readspeed  [protected]
```

Olvasási sebesség.

4.16.4.2 size

```
int Storage::size  [protected]
```

Méret.

4.16.4.3 writespeed

```
int Storage::writespeed  [protected]
```

Írási sebesség.

The documentation for this class was generated from the following files:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp

4.17 String Class Reference

```
#include <schtring.hpp>
```

Public Member Functions

- size_t size () const
  hossz lezáró nulla nélkül
- size_t length () const
  Visszaadja a string hosszát.
- String ()
  Default konstruktor.
- String (char ch)
  Konstruktor: egy char karakterre.
- String (const char *p)
  Konstruktor: egy karakter tömbre.
- String (const String &s1)
  Konstruktor: egy másik Stringre.
- const char * c_str () const
  C-stringet ad vissza.
- ~String ()
  Destruktor.
- String & operator= (const String &rhs_s)
  Egyenlőség operator.
- String & operator+= (const String &rhs_s)
  Pluszegyenlő operator.
- String operator+ (const String &rhs_s) const
  string + string
- String operator+ (char rhs_c)
  string + karakter
- bool operator== (String &rhs_s) const
  hasonlító operator stringgel
- bool operator== (const String &rhs_s) const
- bool operator== (const char *rhs_s)
  hasonlító operator char tömbbel
- bool operator== (const char *rhs_s) const
- String operator-- (int a)
  kitörli az utolsó karaktert a stringből
- char & operator[ ] (unsigned int idx)

index operator

- const char & operator[ ] (unsigned int idx) const

  index operator

- void erase ()

  törli a stringben lévő karaktereket

- void removeFirstX (int x)

  törli az első x karaktert a stringből

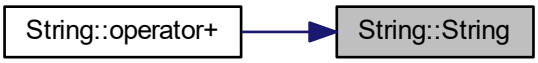
4.17.1 Constructor & Destructor Documentation

4.17.1.1 String() [1/4]

```
String::String ( )  [inline]
```

Default konstruktor.

Here is the caller graph for this function:



4.17.1.2 String() [2/4]

```
String::String (
            char ch )
```

Konstruktor: egy char karakterre.

4.17.1.3 String() [3/4]

```
String::String (
            const char * p )
```

Konstruktor: egy karakter tömbre.

4.17.1.4 String() [4/4]

```
String::String (
            const String & s1 )
```

Konstruktor: egy másik Stringre.

4.17.1.5 ~String()

```
String::~String ( )  [inline]
```

Destruktor.

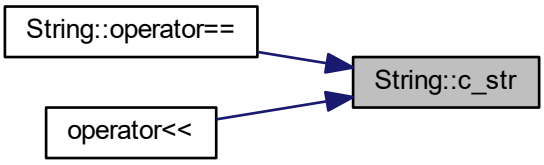
4.17.2 Member Function Documentation

4.17.2.1 c_str()

```
const char* String::c_str ( ) const  [inline]
```

C-stringet ad vissza.

Here is the caller graph for this function:



4.17.2.2 erase()

```
void String::erase ( )  [inline]
```

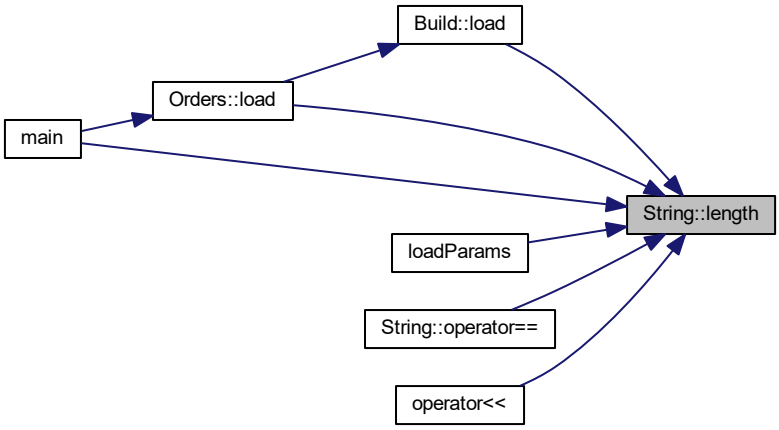
törli a stringben lévő karaktereket

4.17.2.3 length()

```
size_t String::length ( ) const  [inline]
```

Visszaadja a string hosszát.

Here is the caller graph for this function:



4.17.2.4 operator+() [1/2]

```
String String::operator+ (
            const String & rhs_s ) const
```

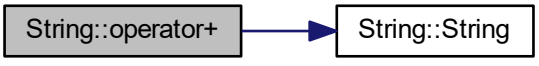
string + string

4.17.2.5 operator+() [2/2]

```
String String::operator+ (
            char rhs_c )  [inline]
```

string + karakter

Here is the call graph for this function:



4.17.2.6 operator+=()

```
String& String::operator+= (
            const String & rhs_s )  [inline]
```

Pluszegyenlő operator.

4.17.2.7 operator--()

```
String String::operator-- (
            int a )
```

kitörli az utolsó karaktert a stringből

4.17.2.8 operator=()

```
String & String::operator= (
            const String & rhs_s )
```

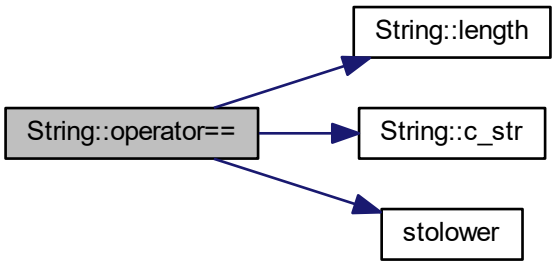
Egyenlőség operator.

4.17.2.9 operator==() [1/4]

```
bool String::operator== (
            String & rhs_s ) const
```

hasonlító operator stringgel

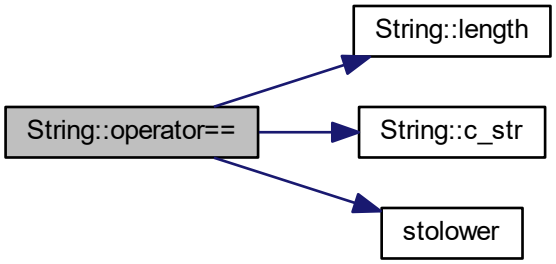
Here is the call graph for this function:



4.17.2.10 operator==() [2/4]

```
bool String::operator== (
            const String & rhs_s ) const
```

Here is the call graph for this function:

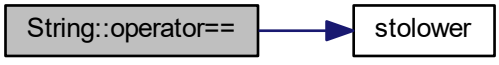


4.17.2.11 operator==() [3/4]

```
bool String::operator== (
            const char * rhs_s )
```

hasonlító operator char tömbbel

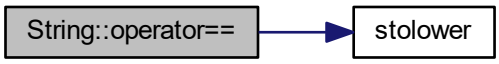
Here is the call graph for this function:



4.17.2.12 operator==() [4/4]

```
bool String::operator== (
            const char * rhs_s ) const
```

Here is the call graph for this function:



4.17.2.13 operator[]() [1/2]

```
char & String::operator[] (
            unsigned int idx )
```

index operator

4.17.2.14 operator[]() [2/2]

```
const char & String::operator[] (
            unsigned int idx ) const
```

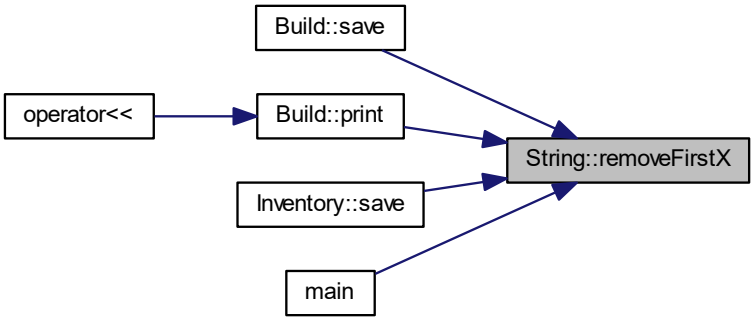
index operator

4.17.2.15 removeFirstX()

```
void String::removeFirstX (
            int x )
```

törli az első x karaktert a stringből

Here is the caller graph for this function:

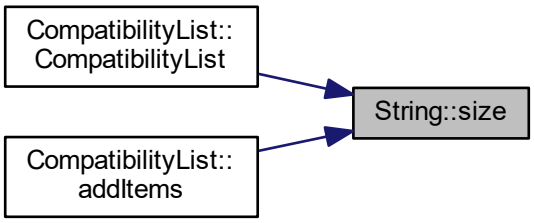


4.17.2.16 size()

```
size_t String::size ( ) const  [inline]
```

hossz lezáró nulla nélkül

Visszaadja a string hosszát Here is the caller graph for this function:



The documentation for this class was generated from the following files:

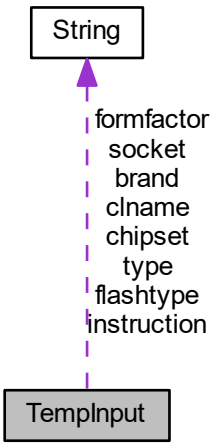
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.cpp

4.18 TempInput Struct Reference

Lehetséges inputokat tárolja adatokkal való konstruáláshoz.

```
#include <Parts.h>
```

Collaboration diagram for TempInput:



Public Attributes

- String instruction
  Mihez tartozik a változó
- String clname
  Kompatibilitás lista neve.
- String brand
  Gyártó
- String type
  Típus.
- int price
  Ár.
- String socket
  Foglalat.
- int clk
  Órajel.
- int cores
  Magok száma.
- bool multithreading
  Multithreading support.
- String chipset
  Chipset.
- String formfactor
  Méret szabvány.
- int size
  Memória méret.
- int wattage
  Teljesítmény.
- int readspeed
  Olvasási sebesség.
- int writespeed
  Írási sebesség.
- String flashtype
  Flash csip típusa.
- int rpm
  Fordulatszám.

4.18.1 Detailed Description

Lehetséges inputokat tárolja adatokkal való konstruáláshoz.

4.18.2 Member Data Documentation

4.18.2.1 brand

```
String TempInput::brand
```

Gyártó

4.18.2.2 chipset

```
String TempInput::chipset
```

Chipset.

4.18.2.3 clk

```
int TempInput::clk
```

Órajel.

4.18.2.4 clname

```
String TempInput::clname
```

Kompatibilitás lista neve.

4.18.2.5 cores

```
int TempInput::cores
```

Magok száma.

4.18.2.6 flashtype

```
String TempInput::flashtype
```

Flash csip típusa.

4.18.2.7 formfactor

```
String TempInput::formfactor
```

Méret szabvány.

4.18.2.8 instruction

```
String TempInput::instruction
```

Mihez tartozik a változó

4.18.2.9 multithreading

```
bool TempInput::multithreading
```

Multithreading support.

4.18.2.10 price

```
int TempInput::price
```

Ár.

4.18.2.11 readspeed

```
int TempInput::readspeed
```

Olvasási sebesség.

4.18.2.12 rpm

```
int TempInput::rpm
```

Fordulatszám.

4.18.2.13 size

```
int TempInput::size
```

Memória méret.

4.18.2.14 socket

```
String TempInput::socket
```

Foglalat.

4.18.2.15 type

```
String TempInput::type
```

Típus.

4.18.2.16 wattage

```
int TempInput::wattage
```

Teljesítmény.

4.18.2.17 writespeed

```
int TempInput::writespeed
```

Írási sebesség.

The documentation for this struct was generated from the following file:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h

4.19 typ_ostream Struct Reference

csak típus stream manipulator

```
#include <schtring.hpp>
```

Public Attributes

- std::ostream & os

4.19.1 Detailed Description

csak típus stream manipulator

4.19.2 Member Data Documentation

4.19.2.1 os

```
std::ostream& typ_ostream::os
```

The documentation for this struct was generated from the following file:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp

4.20 typ_t Struct Reference

csak típus toggle

```
#include <schtring.hpp>
```

4.20.1 Detailed Description

csak típus toggle

The documentation for this struct was generated from the following file:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp

4.21 utos_ostream Struct Reference

szóközösítő stream manipulator

`#include <schtring.hpp>`

Public Attributes

- std::ostream & os

4.21.1 Detailed Description

szóközösítő stream manipulator

4.21.2 Member Data Documentation

4.21.2.1 os

`std::ostream& utos_ostream::os`

The documentation for this struct was generated from the following file:

- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp

4.22 utos_t Struct Reference

szóközösítő toggle

`#include <schtring.hpp>`

4.22.1 Detailed Description

szóközösítő toggle

The documentation for this struct was generated from the following file:

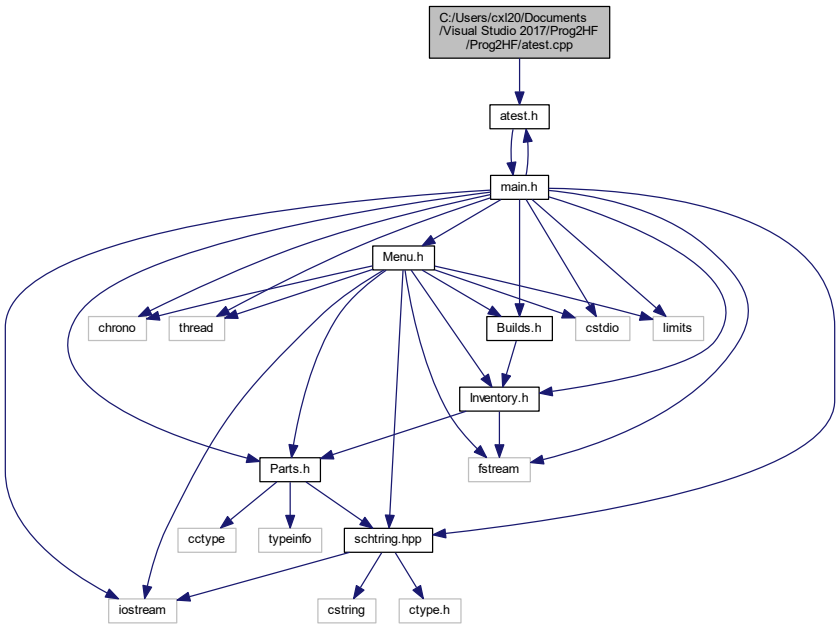
- C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp

5 File Documentation

5.1 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/atest.cpp File Reference

```
#include "atest.h"
```

Include dependency graph for atest.cpp:



Functions

- void test1 (std::fstream &partsFile, const char filename[52])

  Test the if the parts file could be opened.
- bool test3 (String test1, String test2)

  Test the non case sensitive String compare.
- bool test4 (String asd, const char *test)

  Test the string shortener.
- bool test5 (String asd, const char *test)

  Test the sring first x character removal.

5.1.1 Function Documentation

5.1.1.1 test1()

```
void test1 (
            std::fstream & partsFile,
            const char filename[52] )
```

Test the if the parts file could be opened.

5.1.1.2 test3()

```
bool test3 (
            String test1,
            String test2 )
```

Test the non case sensitive String compare.

Here is the call graph for this function:

5.1.1.3 test4()

```
bool test4 (
            String asd,
            const char * test )
```

Test the string shortener.

5.1.1.4 test5()

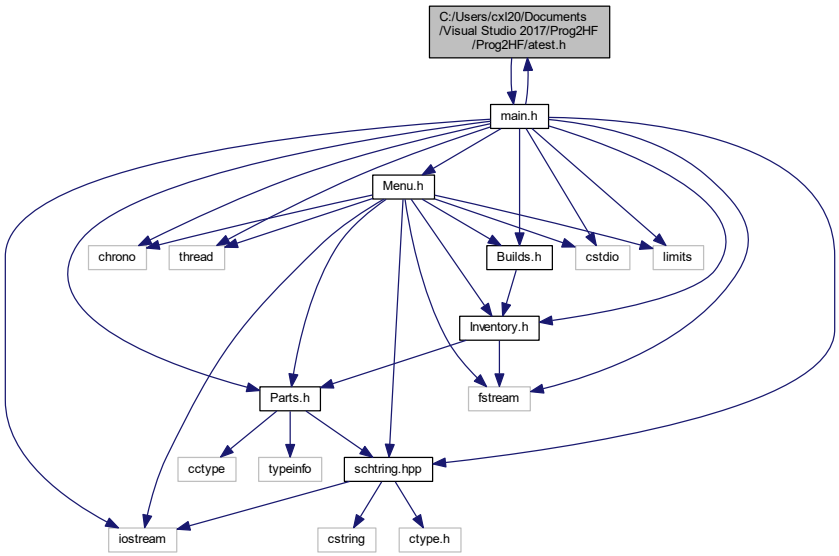
```
bool test5 (
            String asd,
            const char * test )
```

Test the sring first x character removal.

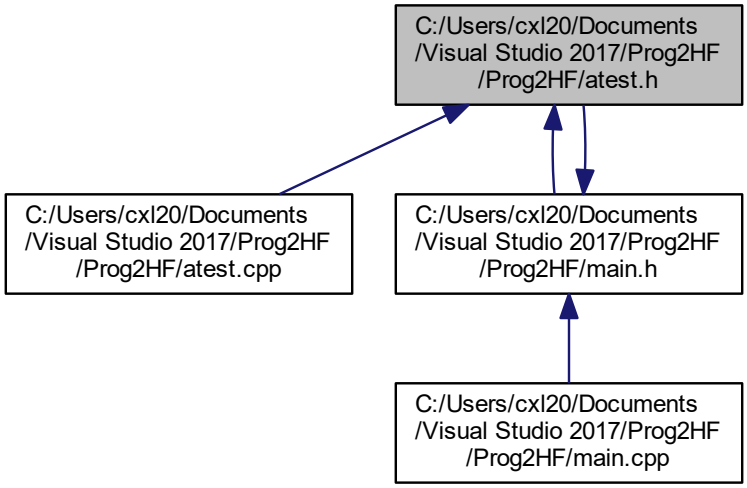
5.2 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/atest.h File Reference

```
#include "main.h"
```

Include dependency graph for atest.h:



This graph shows which files directly or indirectly include this file:



Functions

- void test1 (std::fstream &partsFile, const char partsfilename[52])

Test the if the parts file could be opened.

- bool test3 (String test1, String test2)

  Test the non case sensitive String compare.

- bool test4 (String asd, const char *test)

  Test the string shortener.

- bool test5 (String asd, const char *test)

  Test the sring first x character removal.

5.2.1 Function Documentation

5.2.1.1 test1()

```
void test1 (
            std::fstream & partsFile,
            const char partsfilename[52] )
```

Test the if the parts file could be opened.

5.2.1.2 test3()

```
bool test3 (
            String test1,
            String test2 )
```

Test the non case sensitive String compare.

Here is the call graph for this function:

5.2.1.3 test4()

```
bool test4 (
            String asd,
            const char * test )
```

Test the string shortener.

5.2.1.4 test5()

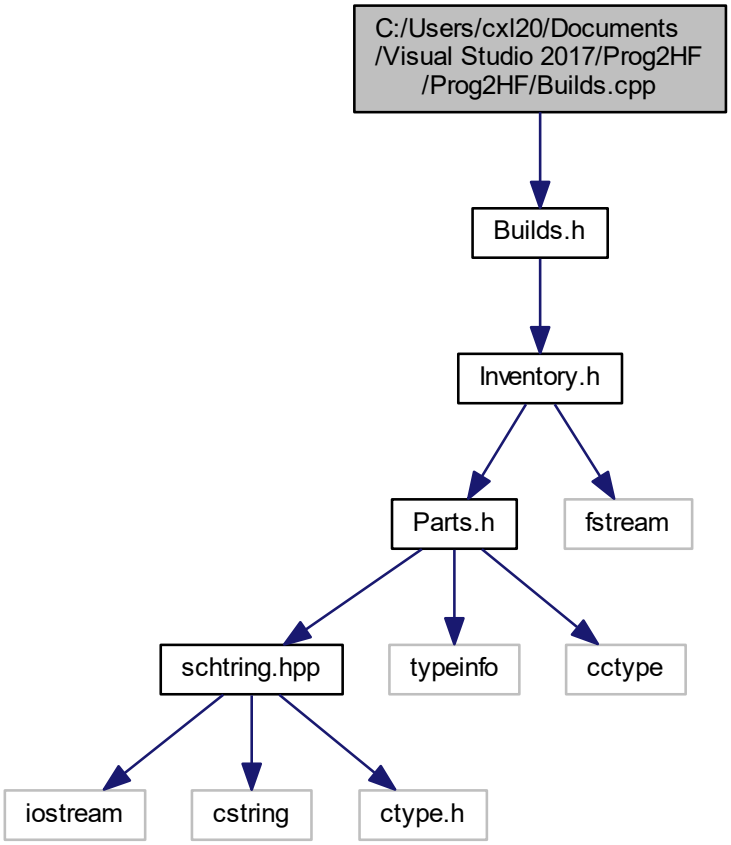
```
bool test5 (
            String asd,
            const char * test )
```

Test the sring first x character removal.

5.3 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Builds.cpp File Reference

```
#include "Builds.h"
```

Include dependency graph for Builds.cpp:



Functions

- std::ostream & operator<< (std::ostream &os, const Build &b)
- std::ostream & operator<< (std::ostream &os, const Orders &o)
- std::ostream & operator<< (simple_ostream tos, const Orders &o)

5.3.1 Function Documentation

5.3.1.1 operator<<() [1/3]

```
std::ostream& operator<< (
            std::ostream & os,
            const Build & b )
```

Here is the call graph for this function:

5.3.1.2 operator<<() [2/3]

```
std::ostream& operator<< (
            std::ostream & os,
            const Orders & o )
```

Here is the call graph for this function:

5.3.1.3 operator<<() [3/3]

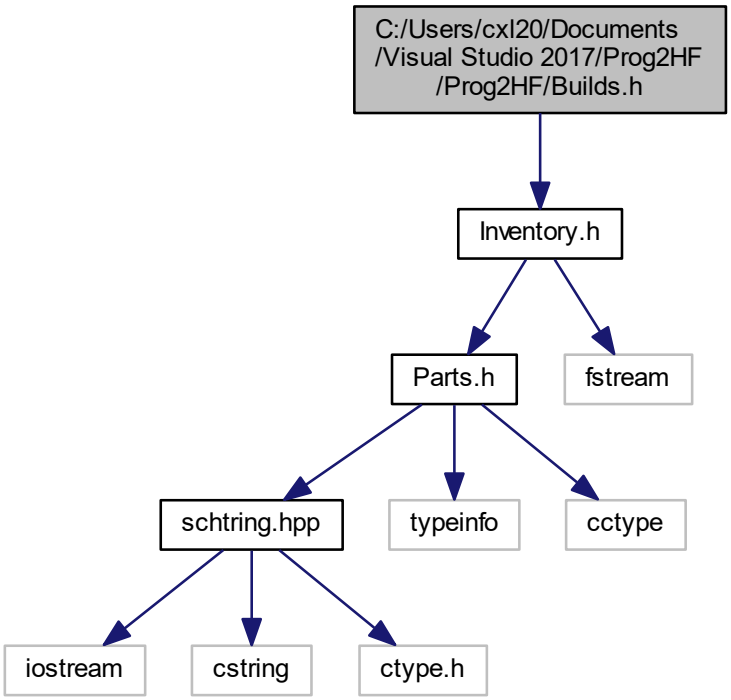
```
std::ostream& operator<< (
            simple_ostream tos,
            const Orders & o )
```

Here is the call graph for this function:

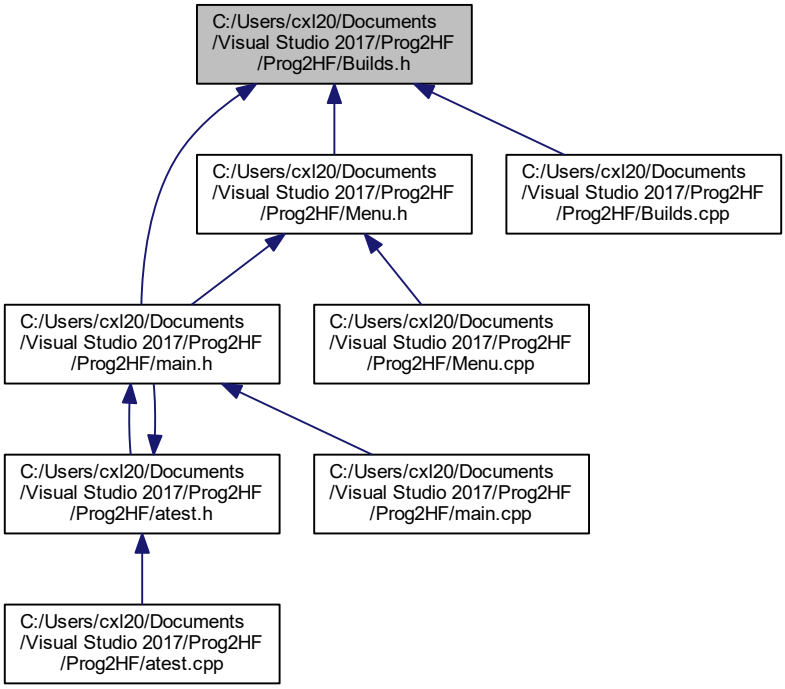
5.4 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Builds.h File Reference

```
#include "Inventory.h"
```

Include dependency graph for Builds.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Build

  Egy gépkonfigot tárol.
- class Orders

  A megrendelt konfigokat tárolja.

Functions

- std::ostream & operator<< (std::ostream &os, const Build &b)
- std::ostream & operator<< (std::ostream &os, const Orders &o)
- std::ostream & operator<< (simple_ostream tos, const Orders &o)

5.4.1 Function Documentation

5.4.1.1 operator<<() [1/3]

```
std::ostream& operator<< (
            std::ostream & os,
            const Build & b )
```

Here is the call graph for this function:

5.4.1.2 operator<<() [2/3]

```
std::ostream& operator<< (
            std::ostream & os,
            const Orders & o )
```

Here is the call graph for this function:

5.4.1.3 operator<<() [3/3]

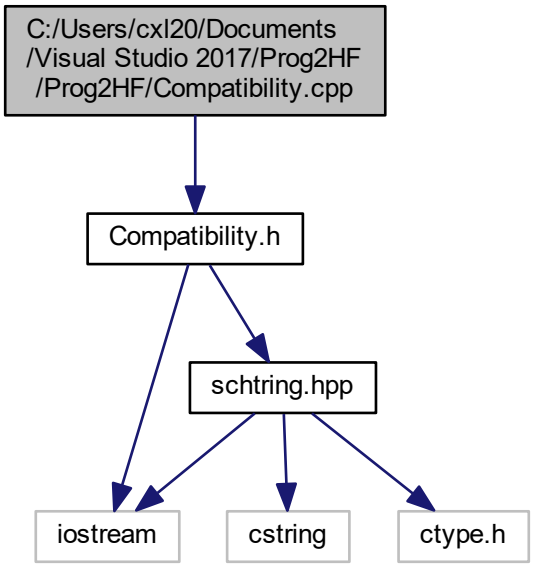
```
std::ostream& operator<< (
            simple_ostream tos,
            const Orders & o )
```

Here is the call graph for this function:

5.5 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Compatibility.cpp File Reference

```
#include "Compatibility.h"
```

Include dependency graph for Compatibility.cpp:



Functions

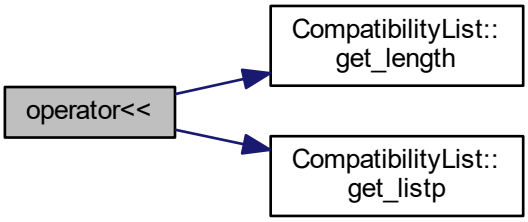
- std::ostream & operator<< (std::ostream &os, const CompatibilityList &cl)

5.5.1 Function Documentation

5.5.1.1 operator<<()

```
std::ostream& operator<< (
            std::ostream & os,
            const CompatibilityList & cl )
```

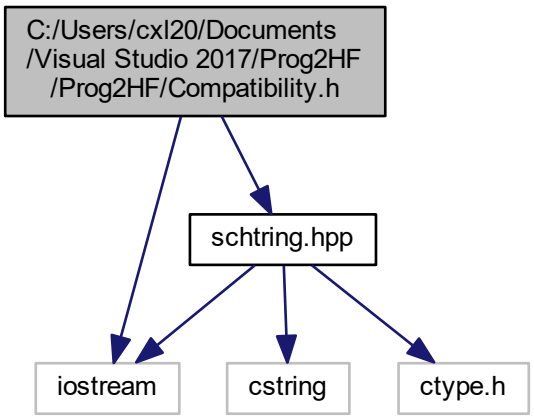
Here is the call graph for this function:



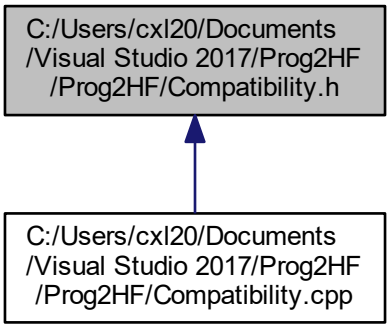
5.6 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Compatibility.h File Reference

```
#include <iostream>
#include "schtring.hpp"
```

Include dependency graph for Compatibility.h:



This graph shows which files directly or indirectly include this file:



Classes

- class CompatibilityList

Functions

- std::ostream & operator<< (std::ostream &os, const CompatibilityList &cl)
- template<typename T1 = String, typename T2 = String>
  bool compatible (T1 is, T2 with, CompatibilityList cl)

5.6.1 Function Documentation

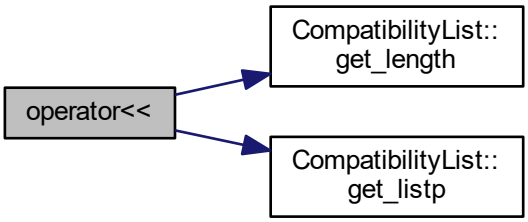
5.6.1.1 compatible()

```
template<typename T1 = String, typename T2 = String>
bool compatible (
            T1 is,
            T2 with,
            CompatibilityList cl )
```

5.6.1.2 operator<<()

```
std::ostream& operator<< (
            std::ostream & os,
            const CompatibilityList & cl )
```

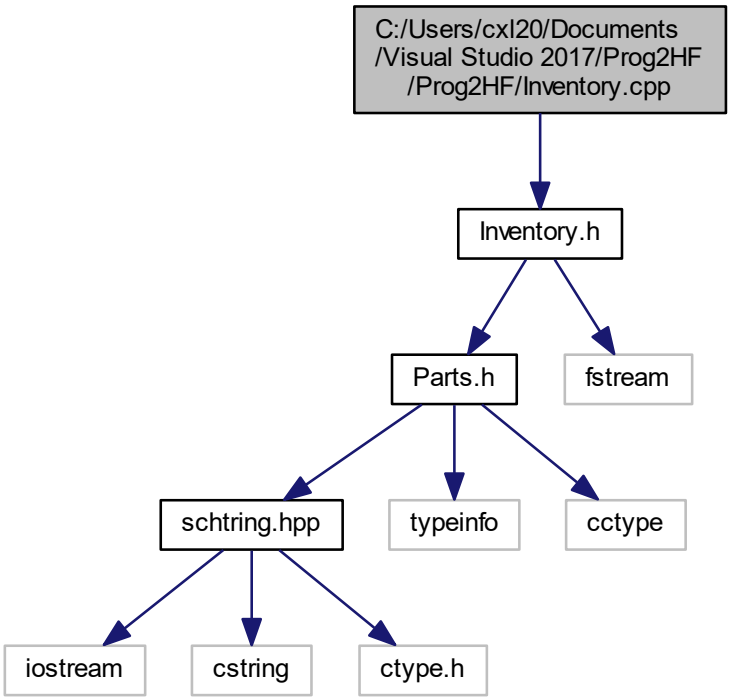
Here is the call graph for this function:



5.7 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Inventory.cpp File Reference

```
#include "Inventory.h"
```

Include dependency graph for Inventory.cpp:



Functions

- void loadParams (std::fstream &is, TempInput &tmp, int const params)

  Jelölők alapján betölti az alkatrész paramétereit fájlból.
- void loadBaseParams (std::istream &is, TempInput &tmp)

  Segítség kiírásával betölti az alkatrész paramétereit console ból.
- void loadCPUParams (std::istream &is, TempInput &tmp)
- void loadGPUParams (std::istream &is, TempInput &tmp)
- void loadMOBOParams (std::istream &is, TempInput &tmp)
- void loadRAMParams (std::istream &is, TempInput &tmp)
- void loadCaseParams (std::istream &is, TempInput &tmp)
- void loadPSUParams (std::istream &is, TempInput &tmp)
- void loadSSDParams (std::istream &is, TempInput &tmp)
- void loadHDDParams (std::istream &is, TempInput &tmp)

5.7.1 Function Documentation

5.7.1.1 loadBaseParams()

```
void loadBaseParams (
            std::istream & is,
            TempInput & tmp )
```

Segítség kiírásával betölti az alkatrész paramétereit console ból.

5.7.1.2 loadCaseParams()

```
void loadCaseParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.7.1.3 loadCPUParams()

```
void loadCPUParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.7.1.4 loadGPUParams()

```
void loadGPUParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.7.1.5 loadHDDParams()

```
void loadHDDParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.7.1.6 loadMOBOParams()

```
void loadMOBOParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.7.1.7 loadParams()

```
void loadParams (
            std::fstream & is,
            TempInput & tmp,
            int const params )
```

Jelölők alapján betölti az alkatrész paramétereit fájlból.

Here is the call graph for this function:

5.7.1.8 loadPSUParams()

```
void loadPSUParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.7.1.9 loadRAMParams()

```
void loadRAMParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.7.1.10 loadSSDParams()

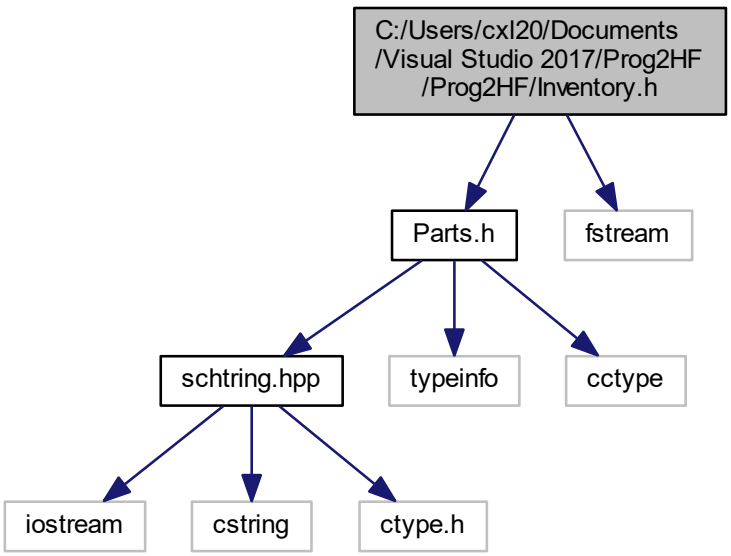
```
void loadSSDParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

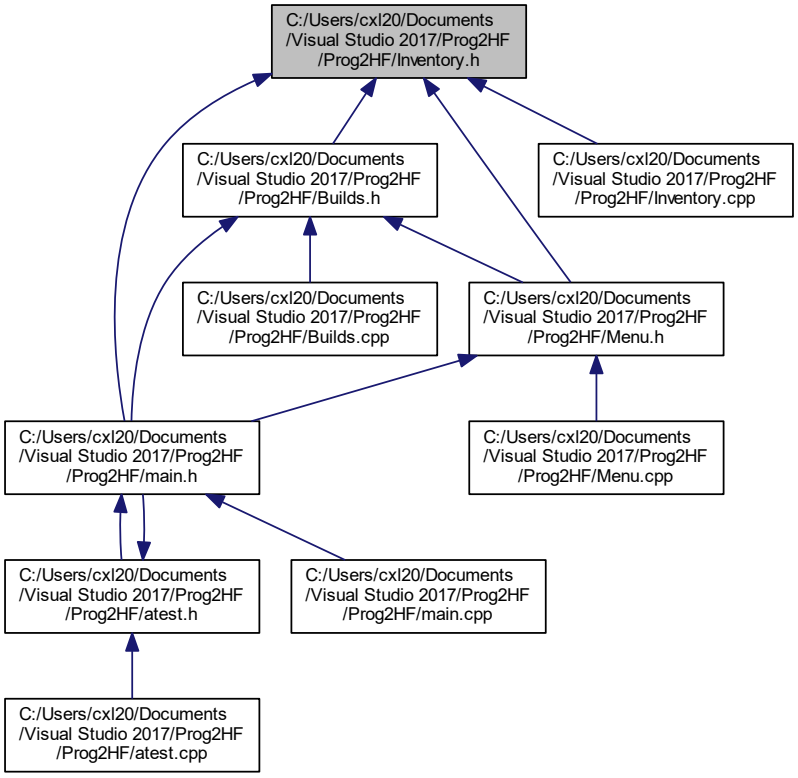
5.8 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Inventory.h File Reference

```
#include "Parts.h"
#include <fstream>
```

Include dependency graph for Inventory.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Inventory

  Alkatrész tároló

Functions

- void loadParams (std::fstream &is, TempInput &tmp, int const params)

  Jelölők alapján betölti az alkatrész paramétereit fájlból.
- void loadBaseParams (std::istream &is, TempInput &tmp)

  Segítség kiírásával betölti az alkatrész paramétereit console ból.
- void loadCPUParams (std::istream &is, TempInput &tmp)
- void loadGPUParams (std::istream &is, TempInput &tmp)
- void loadMOBOParams (std::istream &is, TempInput &tmp)
- void loadRAMParams (std::istream &is, TempInput &tmp)
- void loadCaseParams (std::istream &is, TempInput &tmp)
- void loadPSUParams (std::istream &is, TempInput &tmp)
- void loadSSDParams (std::istream &is, TempInput &tmp)
- void loadHDDParams (std::istream &is, TempInput &tmp)

5.8.1 Function Documentation

5.8.1.1 loadBaseParams()

```
void loadBaseParams (
            std::istream & is,
            TempInput & tmp )
```

Segítség kiírásával betölti az alkatrész paramétereit console ból.

5.8.1.2 loadCaseParams()

```
void loadCaseParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.8.1.3 loadCPUParams()

```
void loadCPUParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.8.1.4 loadGPUParams()

```
void loadGPUParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.8.1.5 loadHDDParams()

```
void loadHDDParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.8.1.6 loadMOBOParams()

```
void loadMOBOParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.8.1.7 loadParams()

```
void loadParams (
            std::fstream & is,
            TempInput & tmp,
            int const params )
```

Jelölők alapján betölti az alkatrész paramétereit fájlból.

Here is the call graph for this function:

5.8.1.8 loadPSUParams()

```
void loadPSUParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.8.1.9 loadRAMParams()

```
void loadRAMParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.8.1.10 loadSSDParams()

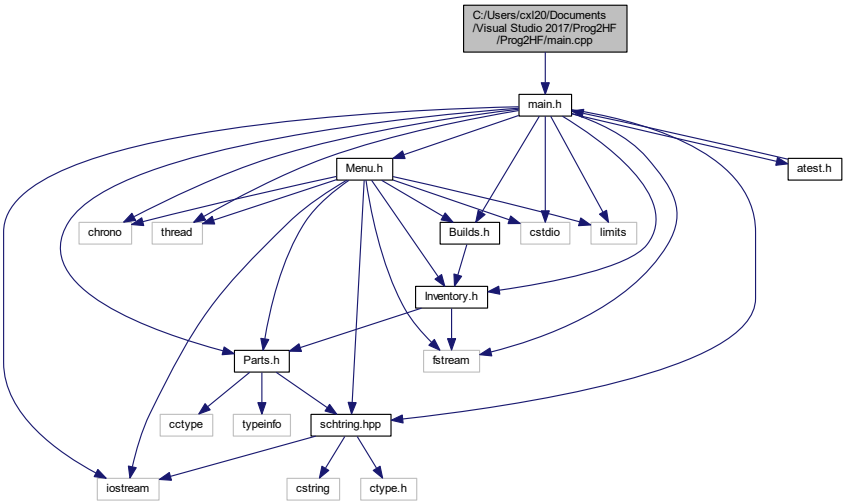
```
void loadSSDParams (
            std::istream & is,
            TempInput & tmp )
```

Here is the call graph for this function:

5.9 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/main.cpp File Reference

```
#include "main.h"
```

Include dependency graph for main.cpp:



Functions

- int main (int argc, char **argv)

  entrypoint

- template<typename T >
  void save (std::fstream &tempFile, std::fstream &origFile, T &classwithsavefunc, std::streampos &pos, const char *filename, const char *tempfilename)

  elmenti a program módosításait

5.9.1 Function Documentation

5.9.1.1 main()

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

entrypoint

Alapértelmezett fájl nevek

fájl nevek beállítása indítási parancsból

Alkatrészek

Megrendelések

első 6 sor átmásolása

pozíció mentése kiíráshoz

Alkatrész típusa betöltéshez

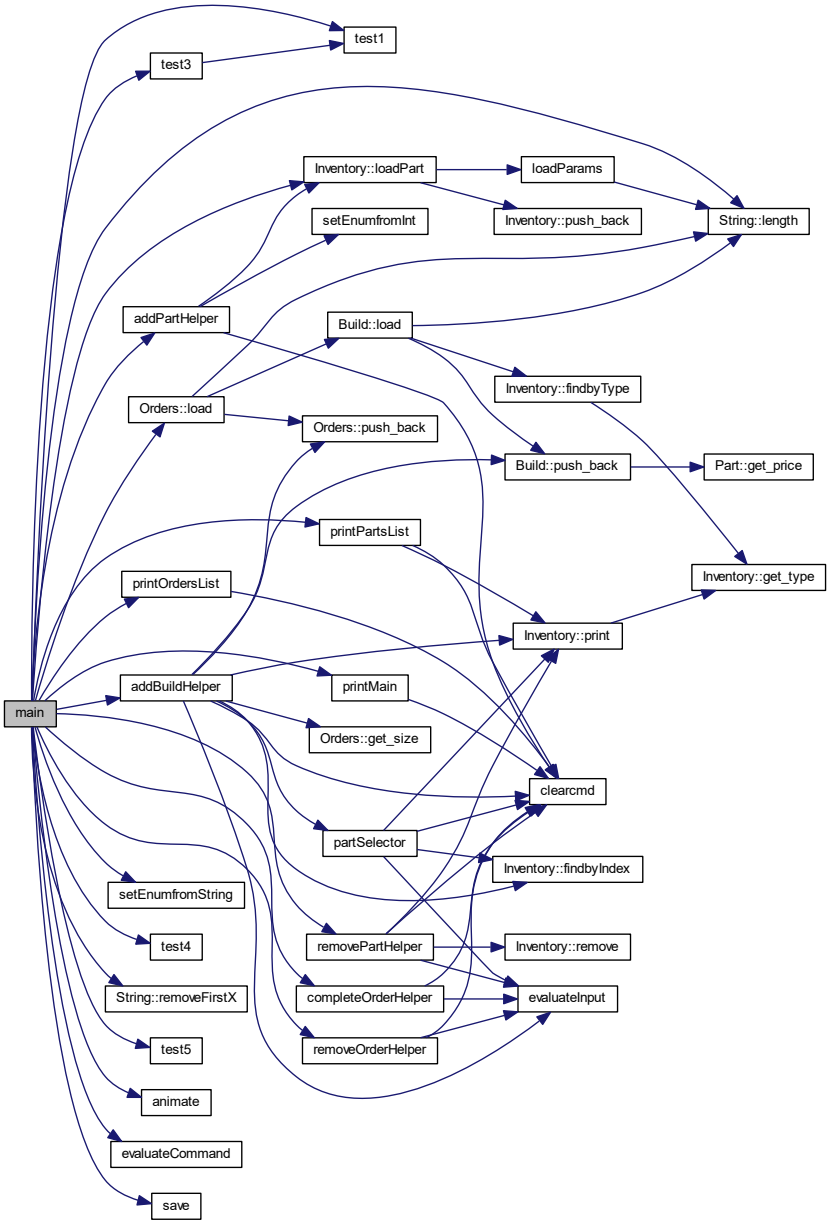
Menüpontok közti váltás

alkatrészek betöltése fájlból

megrendelések betöltése fájlból

main menu loop

mentés Here is the call graph for this function:



5.9.1.2 save()

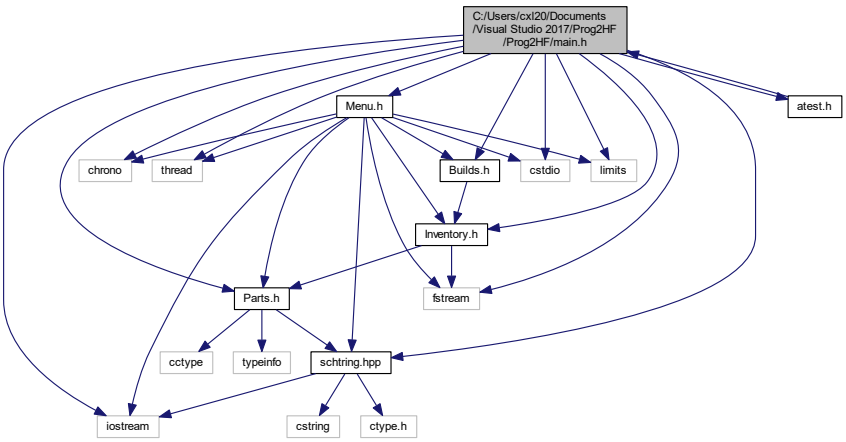
```
template<typename T >
void save (
            std::fstream & tempFile,
            std::fstream & origFile,
            T & classwithsavefunc,
            std::streampos & pos,
            const char * filename,
            const char * tempfilename )
```

elmenti a program módosításait

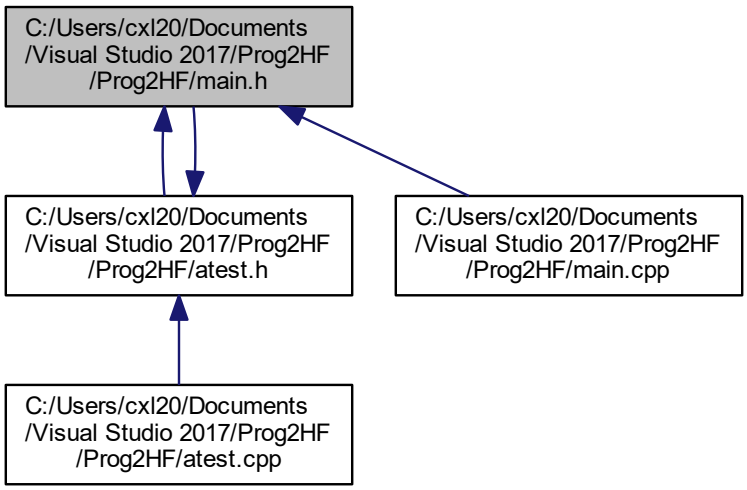
5.10 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/main.h File Reference

```
#include <chrono>
#include <thread>
#include "schtring.hpp"
#include <cstdio>
#include <iostream>
#include <fstream>
#include <limits>
#include "Parts.h"
#include "Inventory.h"
#include "Builds.h"
#include "Menu.h"
#include "atest.h"
```

Include dependency graph for main.h:



This graph shows which files directly or indirectly include this file:



Functions

- int main (int argc, char **argv)

  entrypoint

- template<typename T >
  void save (std::fstream &, std::fstream &, T &, std::streampos &, const char *, const char *)

  elmenti a program módosításait

5.10.1 Function Documentation

5.10.1.1 main()

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

entrypoint

Alapértelmezett fájl nevek

fájl nevek beállítása indítási parancsból

Alkatrészek

Megrendelések

első 6 sor átmásolása

pozíció mentése kiíráshoz

Alkatrész típusa betöltéshez

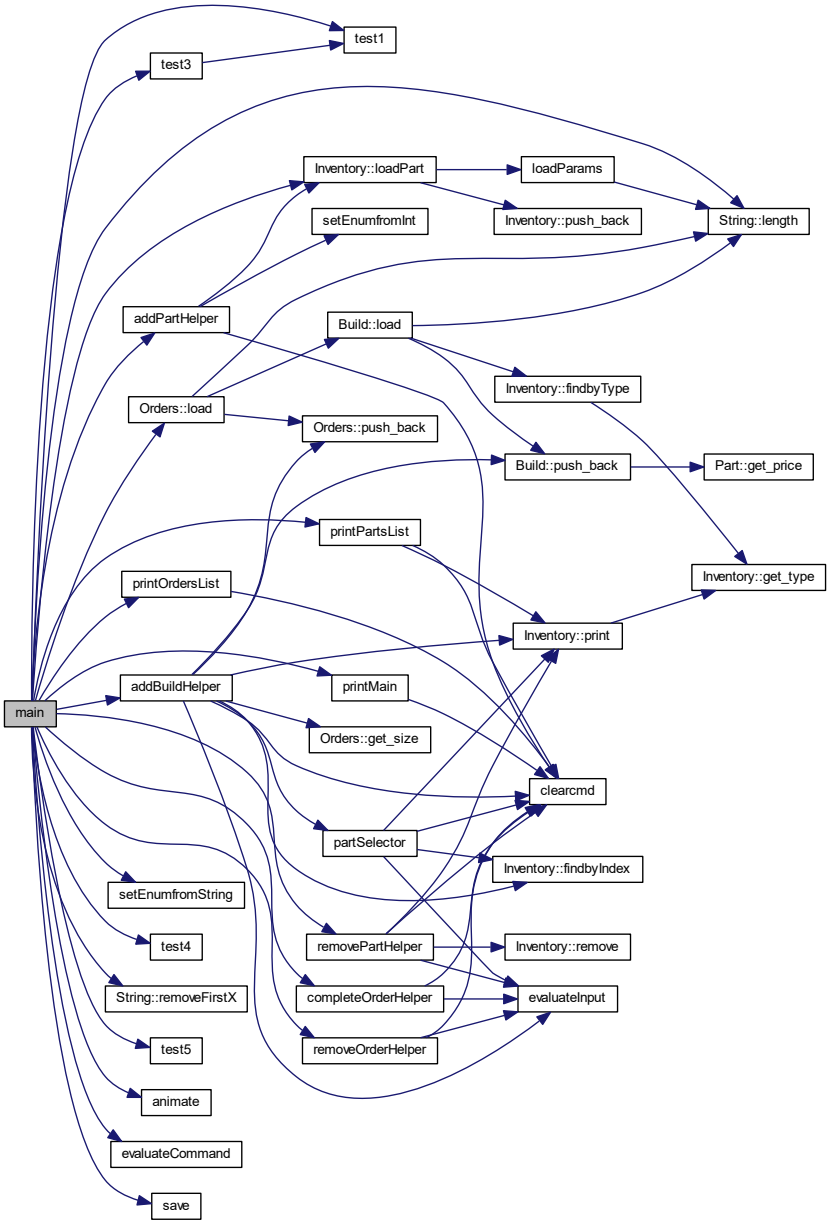
Menüpontok közti váltás

alkatrészek betöltése fájlból

megrendelések betöltése fájlból

main menu loop

mentés Here is the call graph for this function:



5.10.1.2 save()

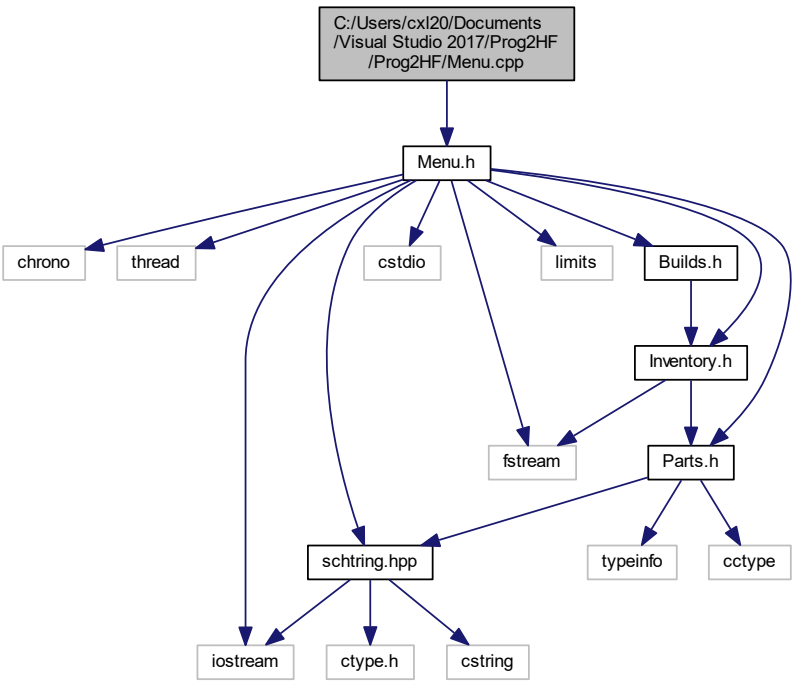
```
template<typename T >
void save (
            std::fstream & ,
            std::fstream & ,
            T & ,
            std::streampos & ,
            const char * ,
            const char *  )
```

elmenti a program módosításait

5.11 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Menu.cpp File Reference

```
#include "Menu.h"
```

Include dependency graph for Menu.cpp:



Functions

- void printMain ()
  kiírja a főmenüt
- void printPartsList (Inventory &inventory)
  kiírja az összes betölttött alkatrészt
- void printOrdersList (Orders &orders)
  kiírja a megrendeléseket
- int addPartHelper (Inventory &inventory, TempInput &tmp, enum enumPart &eP)
  új alkatrészt tölt be console inputról.
- void removePartHelper (Inventory &inventory)
  törli a kiválasztott alkatrészt
- void addBuildHelper (Orders &orders, Inventory &inventory)
  egy configot lehet csinálni vele
- int partSelector (Inventory &inventory, const char *type)
  konfighoz választ alkatrészt
- void completeOrderHelper (Orders &orders)
  megrendelést lehet teljesítetté tenni
- void removeOrderHelper (Orders &orders)
  megrendelést lehet vele törölni
- void animate (char c)

csinál egy sor animációt

- template<typename T >
  int evaluateInput (T &classwithsize)

  átalakítja a beírt számot indexelővé
- void evaluateCommand (enum enumMenu &eM)

  bemenet alapján vált a menük között
- void setEnumfromInt (int a, enumPart &eP)

  beállítja a part loadert

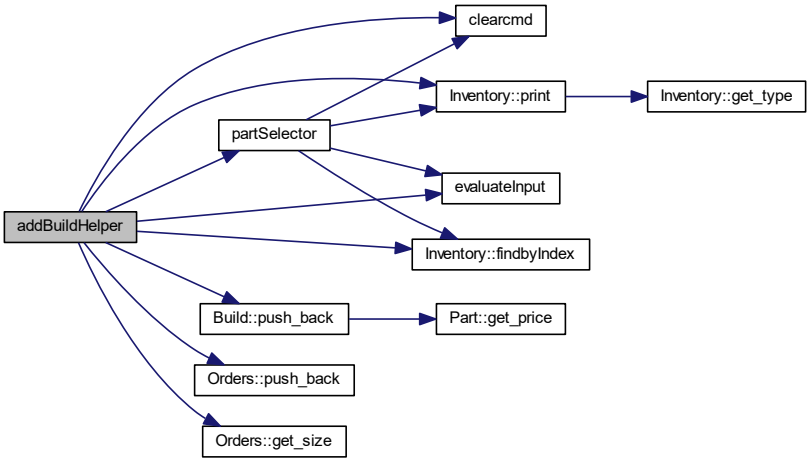
5.11.1 Function Documentation

5.11.1.1 addBuildHelper()

```
void addBuildHelper (
            Orders & orders,
            Inventory & inventory )
```

egy configot lehet csinálni vele

Here is the call graph for this function:

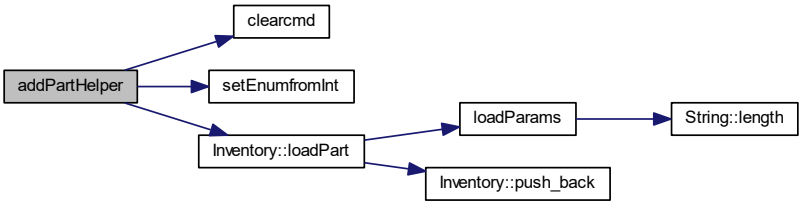


5.11.1.2 addPartHelper()

```
int addPartHelper (
            Inventory & inventory,
            TempInput & tmp,
            enum enumPart & eP )
```

új alkatrészt tölt be console inputról.

Here is the call graph for this function:



5.11.1.3 animate()

```
void animate (
            char c )
```

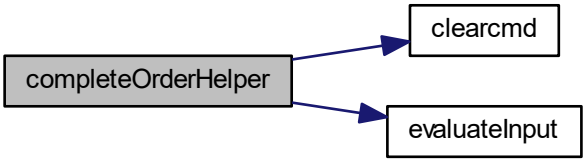
csinál egy sor animációt

5.11.1.4 completeOrderHelper()

```
void completeOrderHelper (
            Orders & orders )
```

megrendelést lehet teljesítetté tenni

Here is the call graph for this function:



5.11.1.5 evaluateCommand()

```
void evaluateCommand (
            enum enumMenu & eM )
```

bemenet alapján vált a menük között

5.11.1.6 evaluateInput()

```
template<typename T >
int evaluateInput (
            T & classwithsize )
```

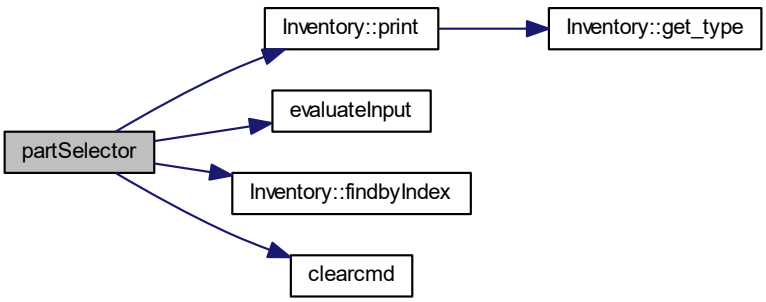
átalakítja a beírt számot indexelővé

5.11.1.7 partSelector()

```
int partSelector (
            Inventory & inventory,
            const char * type )
```

konfighoz választ alkatrészt

Here is the call graph for this function:



5.11.1.8 printMain()

```
void printMain ( )
```

kiírja a főmenüt

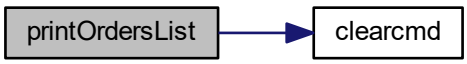
Here is the call graph for this function:

5.11.1.9 printOrdersList()

```
void printOrdersList (
            Orders & orders )
```

kiírja a megrendeléseket

Here is the call graph for this function:

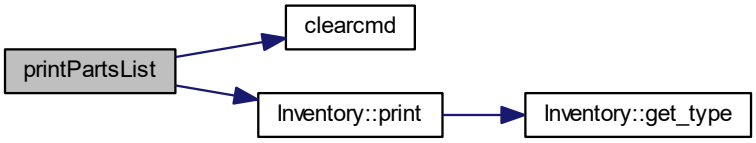


5.11.1.10 printPartsList()

```
void printPartsList (
            Inventory & inventory )
```

kiírja az összes betölttött alkatrészt

Here is the call graph for this function:

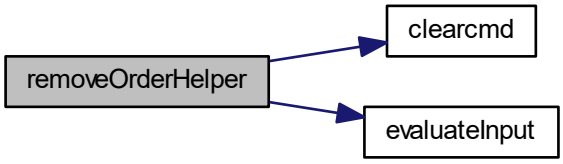


5.11.1.11 removeOrderHelper()

```
void removeOrderHelper (
            Orders & orders )
```

megrendelést lehet vele törölni

Here is the call graph for this function:

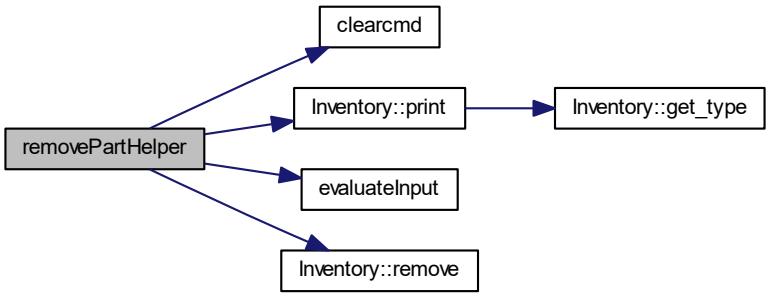


5.11.1.12 removePartHelper()

```
void removePartHelper (
            Inventory & inventory )
```

törli a kiválasztott alkatrészt

Here is the call graph for this function:



5.11.1.13 setEnumfromInt()

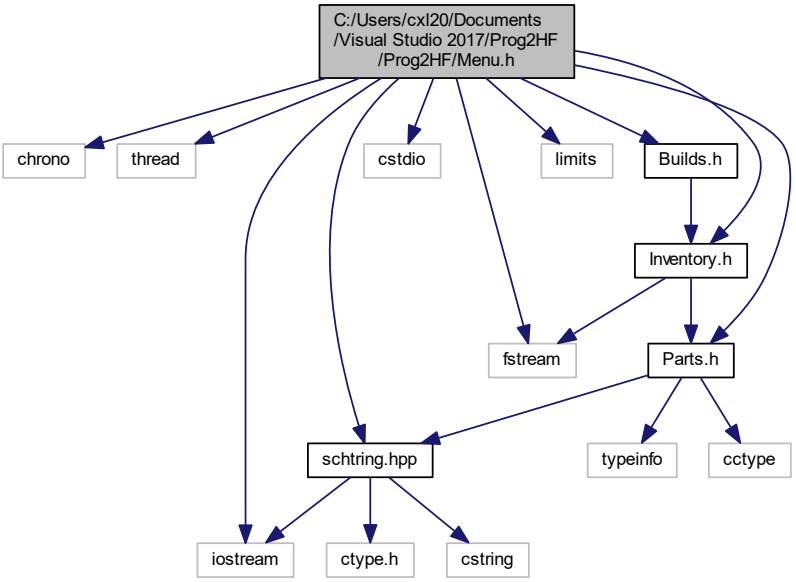
```
void setEnumfromInt (
            int a,
            enumPart & eP )
```

beállítja a part loadert

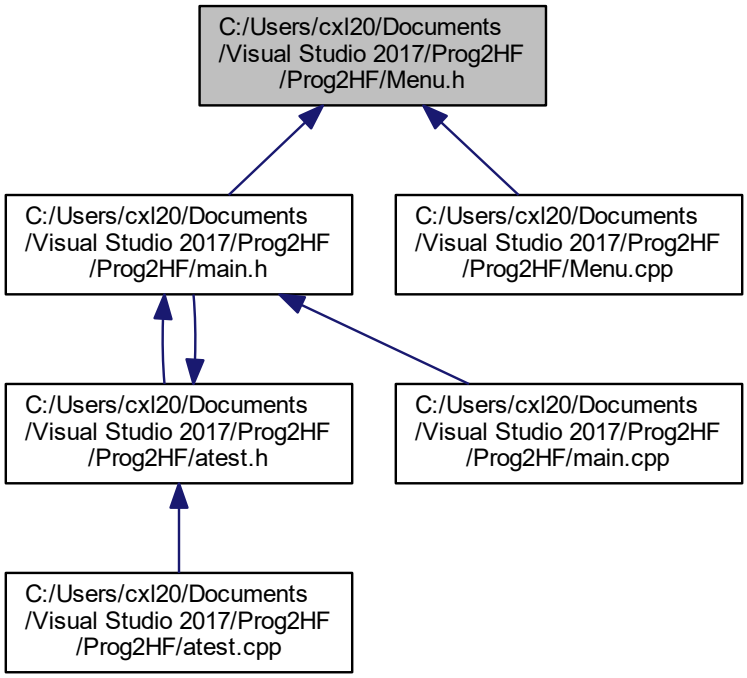
5.12 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Menu.h File Reference

```
#include <chrono>
#include <thread>
#include "schtring.hpp"
#include <cstdio>
#include <iostream>
#include <fstream>
#include <limits>
#include "Parts.h"
#include "Inventory.h"
#include "Builds.h"
```

Include dependency graph for Menu.h:



This graph shows which files directly or indirectly include this file:



Enumerations

- enum enumMenu {
  eMain = 1, ePartsList = 11, ePartsAdd = 12, ePartsRemove = 13,
  eBuildsList = 21, eBuildsAdd = 22, eBuildsComplete = 23, eBuildsRemove = 24,
  eExit = 9 }

  menü almenüi

Functions

- void printOrdersList (Orders &orders)

  kiírja a megrendeléseket
- void animate (char c='∼')

  csinál egy sor animációt
- void printMain ()

  kiírja a főmenüt
- void evaluateCommand (enum enumMenu &)

  bemenet alapján vált a menük között
- void printPartsList (Inventory &)

  kiírja az összes betölttött alkatrészt
- void setEnumfromInt (int a, enumPart &eP)

  beállítja a part loadert
- int addPartHelper (Inventory &, TempInput &, enumPart &)

  új alkatrészt tölt be console inputról.
- void removePartHelper (Inventory &)

  törli a kiválasztott alkatrészt
- void addBuildHelper (Orders &orders, Inventory &inventory)

  egy configot lehet csinálni vele
- int partSelector (Inventory &inventory, const char *type)

  konfighoz választ alkatrészt
- void completeOrderHelper (Orders &orders)

  megrendelést lehet teljesítetté tenni
- void removeOrderHelper (Orders &orders)

  megrendelést lehet vele törölni
- template<typename T >
  int evaluateInput (T &)

  átalakítja a beírt számot indexelővé
- std::fstream & GotoLine (std::fstream &file, unsigned int n)

  n edik sorra ugrik egy file streamben
- void clearcmd ()

  kitörli a terminált

5.12.1 Enumeration Type Documentation

5.12.1.1 enumMenu

```
enum enumMenu
```

menü almenüi

Enumerator

eMain	
ePartsList	
ePartsAdd	
ePartsRemove	
eBuildsList	
eBuildsAdd	
eBuildsComplete	
eBuildsRemove	
eExit	

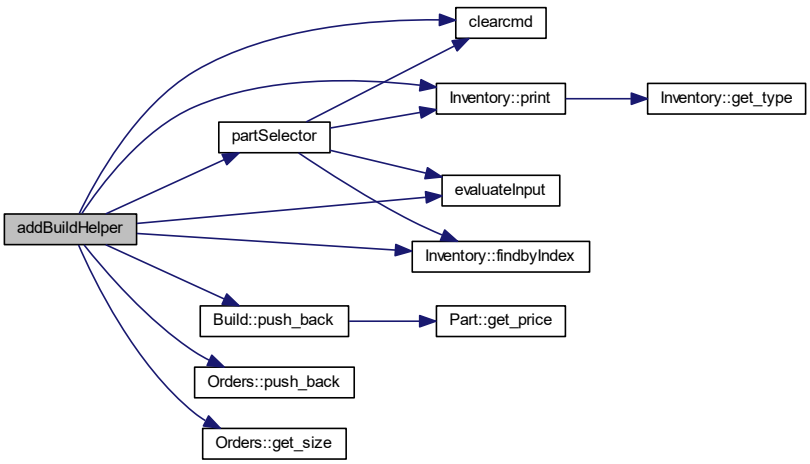
5.12.2 Function Documentation

5.12.2.1 addBuildHelper()

```
void addBuildHelper (
            Orders & orders,
            Inventory & inventory )
```

egy configot lehet csinálni vele

Here is the call graph for this function:

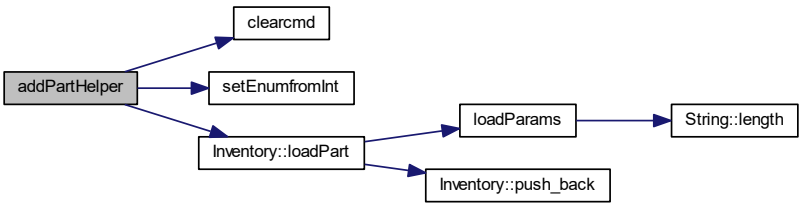


5.12.2.2 addPartHelper()

```
int addPartHelper (
            Inventory & ,
            TempInput & ,
            enumPart &  )
```

új alkatrészt tölt be console inputtról.

Here is the call graph for this function:



5.12.2.3 animate()

```
void animate (
            char c = '~' )
```

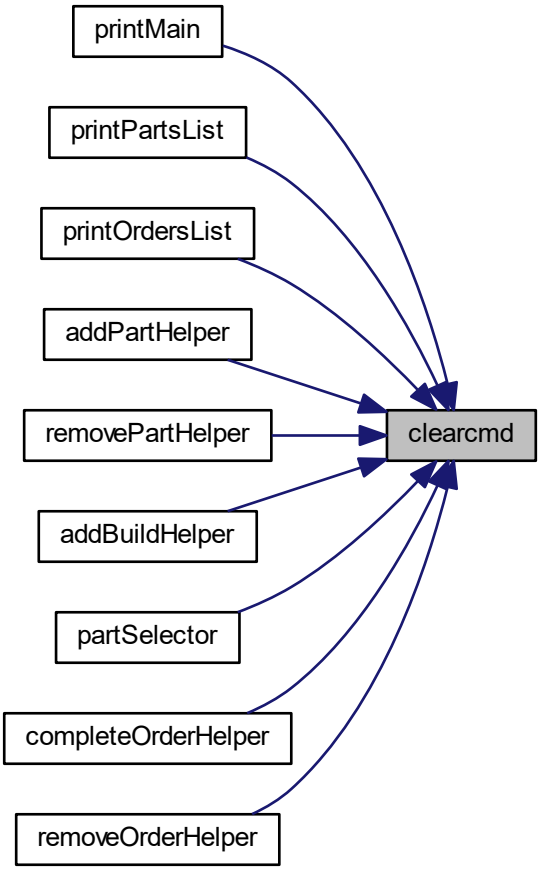
csinál egy sor animációt

5.12.2.4 clearcmd()

```
void clearcmd ( )  [inline]
```

kitörli a terminált

Here is the caller graph for this function:

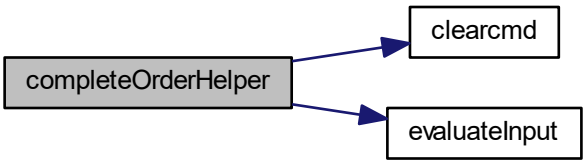


5.12.2.5 completeOrderHelper()

```
void completeOrderHelper (
            Orders & orders )
```

megrendelést lehet teljesítetté tenni

Here is the call graph for this function:



5.12.2.6 evaluateCommand()

```
void evaluateCommand (
              enum enumMenu &  )
```

bemenet alapján vált a menük között

5.12.2.7 evaluateInput()

```
template<typename T >
int evaluateInput (
              T &  )
```

átalakítja a beírt számot indexelővé

5.12.2.8 GotoLine()

```
std::fstream& GotoLine (
              std::fstream & file,
              unsigned int n )  [inline]
```

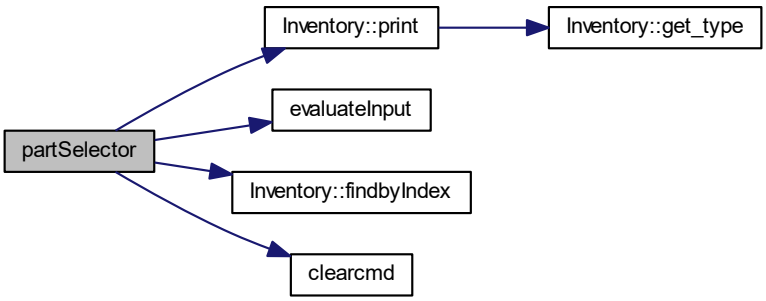
n edik sorra ugrik egy file streamben

5.12.2.9 partSelector()

```
int partSelector (
            Inventory & inventory,
            const char * type )
```

konfighoz választ alkatrészt

Here is the call graph for this function:



5.12.2.10 printMain()

```
void printMain ( )
```

kiírja a főmenüt

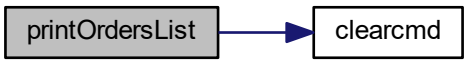
Here is the call graph for this function:

5.12.2.11 printOrdersList()

```
void printOrdersList (
            Orders & orders )
```

kiírja a megrendeléseket

Here is the call graph for this function:

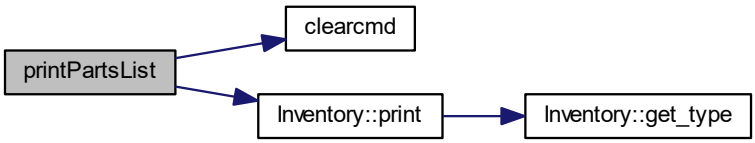


5.12.2.12 printPartsList()

```
void printPartsList (
            Inventory &  )
```

kiírja az összes betölttött alkatrészt

Here is the call graph for this function:

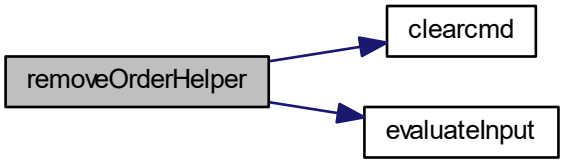


5.12.2.13 removeOrderHelper()

```
void removeOrderHelper (
            Orders & orders )
```

megrendelést lehet vele törölni

Here is the call graph for this function:

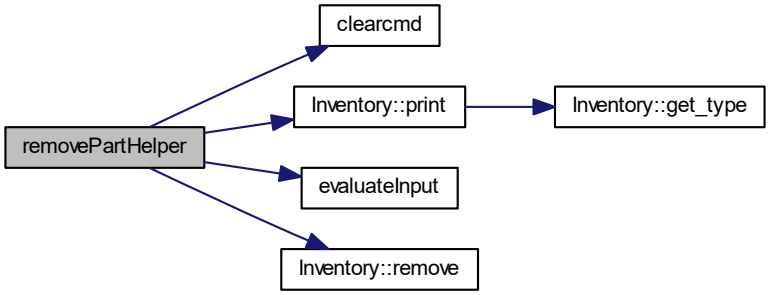


5.12.2.14 removePartHelper()

```
void removePartHelper (
            Inventory &  )
```

törli a kiválasztott alkatrészt

Here is the call graph for this function:



5.12.2.15 setEnumfromInt()

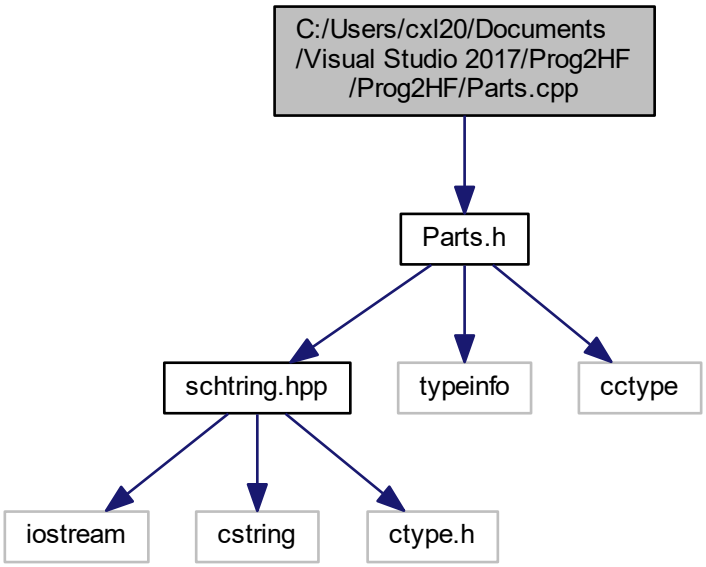
```
void setEnumfromInt (
            int a,
            enumPart & eP )
```

beállítja a part loadert

5.13 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.cpp File Reference

```
#include "Parts.h"
```

Include dependency graph for Parts.cpp:



Functions

- std::ostream & operator<< (std::ostream &os, const Part &p)
- std::ostream & operator<< (utos_ostream tos, const Part &p)
- std::ostream & operator<< (simple_ostream tos, const Part &p)
- std::ostream & operator<< (typ_ostream tos, const Part &p)
- std::ostream & operator<< (std::ostream &os, const CPU &p)
- std::ostream & operator<< (utos_ostream tos, const CPU &p)
- std::ostream & operator<< (simple_ostream tos, const CPU &p)
- std::ostream & operator<< (typ_ostream tos, const CPU &p)
- std::ostream & operator<< (std::ostream &os, const GPU &p)
- std::ostream & operator<< (utos_ostream tos, const GPU &p)
- std::ostream & operator<< (simple_ostream tos, const GPU &p)
- std::ostream & operator<< (typ_ostream tos, const GPU &p)
- std::ostream & operator<< (std::ostream &os, const MOBO &p)
- std::ostream & operator<< (utos_ostream tos, const MOBO &p)
- std::ostream & operator<< (simple_ostream tos, const MOBO &p)
- std::ostream & operator<< (typ_ostream tos, const MOBO &p)
- std::ostream & operator<< (std::ostream &os, const RAM &p)
- std::ostream & operator<< (utos_ostream tos, const RAM &p)
- std::ostream & operator<< (simple_ostream tos, const RAM &p)
- std::ostream & operator<< (typ_ostream tos, const RAM &p)
- std::ostream & operator<< (std::ostream &os, const Case &p)
- std::ostream & operator<< (utos_ostream tos, const Case &p)

- std::ostream & operator<< (simple_ostream tos, const Case &p)
- std::ostream & operator<< (typ_ostream tos, const Case &p)
- std::ostream & operator<< (std::ostream &os, const PSU &p)
- std::ostream & operator<< (utos_ostream tos, const PSU &p)
- std::ostream & operator<< (simple_ostream tos, const PSU &p)
- std::ostream & operator<< (typ_ostream tos, const PSU &p)
- std::ostream & operator<< (std::ostream &os, const Storage &p)
- std::ostream & operator<< (utos_ostream tos, const Storage &p)
- std::ostream & operator<< (simple_ostream tos, const Storage &p)
- std::ostream & operator<< (typ_ostream tos, const Storage &p)
- std::ostream & operator<< (std::ostream &os, const SSD &p)
- std::ostream & operator<< (utos_ostream tos, const SSD &p)
- std::ostream & operator<< (simple_ostream tos, const SSD &p)
- std::ostream & operator<< (typ_ostream tos, const SSD &p)
- std::ostream & operator<< (std::ostream &os, const HDD &p)
- std::ostream & operator<< (utos_ostream tos, const HDD &p)
- std::ostream & operator<< (simple_ostream tos, const HDD &p)
- std::ostream & operator<< (typ_ostream tos, const HDD &p)
- void setEnumfromString (String s0, enumPart &e)

  Enumot állít be gy stringből.

5.13.1 Function Documentation

5.13.1.1 operator<<() [1/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const Part & p )
```

Here is the call graph for this function:

5.13.1.2 operator<<() [2/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const Part & p )
```

Here is the call graph for this function:

5.13.1.3 operator<<() [3/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const Part & p )
```

Here is the call graph for this function:

5.13.1.4 operator<<() [4/40]

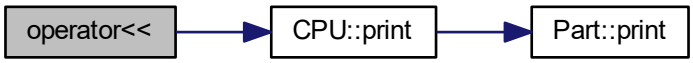
```
std::ostream& operator<< (
            typ_ostream tos,
            const Part & p )
```

Here is the call graph for this function:

5.13.1.5 operator<<() [5/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const CPU & p )
```

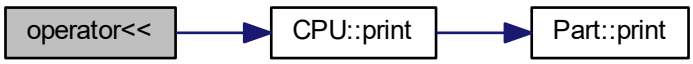
Here is the call graph for this function:



5.13.1.6 operator<<() [6/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const CPU & p )
```

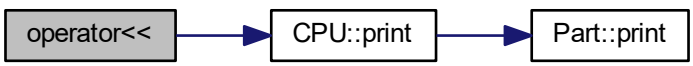
Here is the call graph for this function:



5.13.1.7 operator<<() [7/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const CPU & p )
```

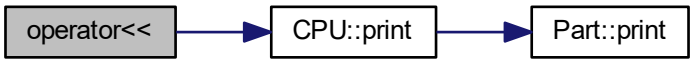
Here is the call graph for this function:



5.13.1.8 operator<<() [8/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const CPU & p )
```

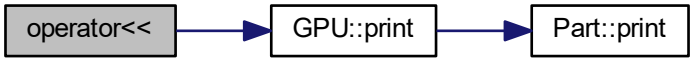
Here is the call graph for this function:



5.13.1.9 operator<<() [9/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const GPU & p )
```

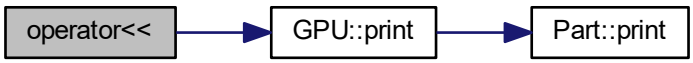
Here is the call graph for this function:



5.13.1.10 operator<<() [10/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const GPU & p )
```

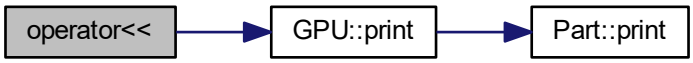
Here is the call graph for this function:



5.13.1.11 operator<<() [11/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const GPU & p )
```

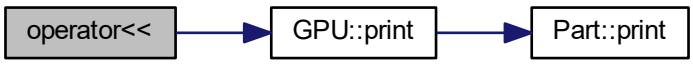
Here is the call graph for this function:



5.13.1.12 operator<<() [12/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const GPU & p )
```

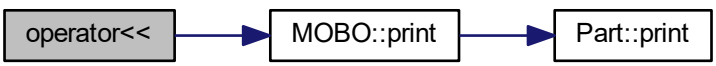
Here is the call graph for this function:



5.13.1.13 operator<<() [13/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const MOBO & p )
```

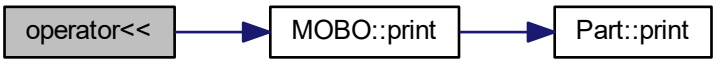
Here is the call graph for this function:



5.13.1.14 operator<<() [14/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const MOBO & p )
```

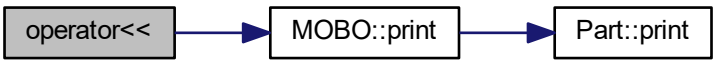
Here is the call graph for this function:



5.13.1.15 operator<<() [15/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const MOBO & p )
```

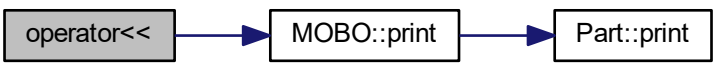
Here is the call graph for this function:



5.13.1.16 operator<<() [16/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const MOBO & p )
```

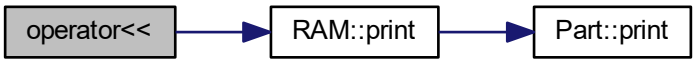
Here is the call graph for this function:



5.13.1.17 operator<<() [17/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const RAM & p )
```

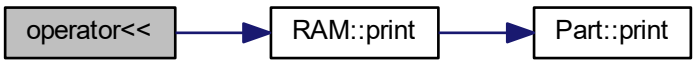
Here is the call graph for this function:



5.13.1.18 operator<<() [18/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const RAM & p )
```

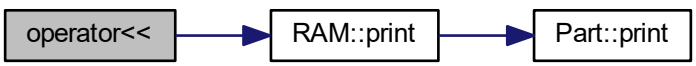
Here is the call graph for this function:



5.13.1.19 operator<<() [19/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const RAM & p )
```

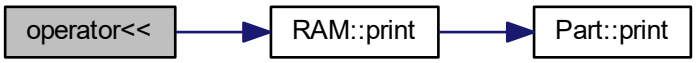
Here is the call graph for this function:



5.13.1.20 operator<<() [20/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const RAM & p )
```

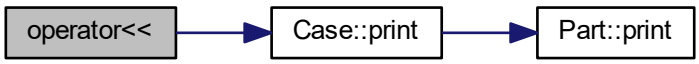
Here is the call graph for this function:



5.13.1.21 operator<<() [21/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const Case & p )
```

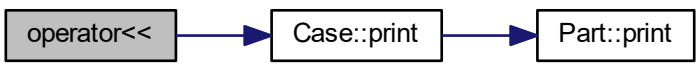
Here is the call graph for this function:



5.13.1.22 operator<<() [22/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const Case & p )
```

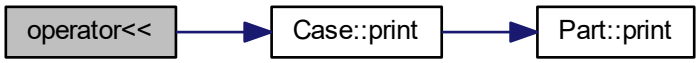
Here is the call graph for this function:



5.13.1.23 operator<<() [23/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const Case & p )
```

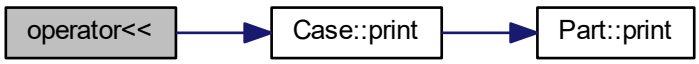
Here is the call graph for this function:



5.13.1.24 operator<<() [24/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const Case & p )
```

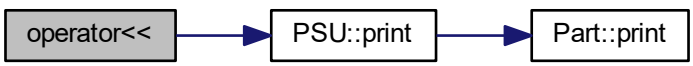
Here is the call graph for this function:



5.13.1.25 operator<<() [25/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const PSU & p )
```

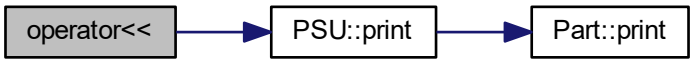
Here is the call graph for this function:



5.13.1.26 operator<<() [26/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const PSU & p )
```

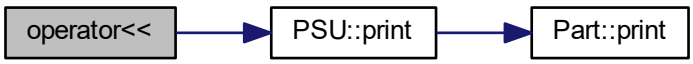
Here is the call graph for this function:



5.13.1.27 operator<<() [27/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const PSU & p )
```

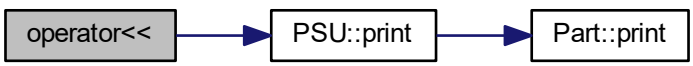
Here is the call graph for this function:



5.13.1.28 operator<<() [28/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const PSU & p )
```

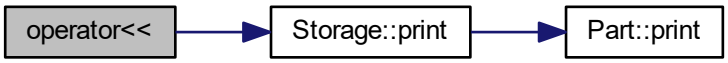
Here is the call graph for this function:



5.13.1.29 operator<<() [29/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const Storage & p )
```

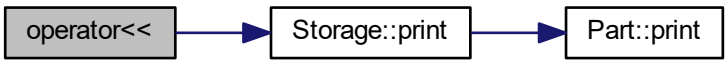
Here is the call graph for this function:



5.13.1.30 operator<<() [30/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const Storage & p )
```

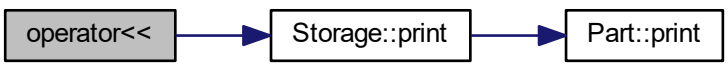
Here is the call graph for this function:



5.13.1.31 operator<<() [31/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const Storage & p )
```

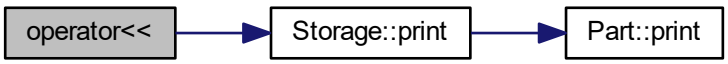
Here is the call graph for this function:



5.13.1.32 operator<<() [32/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const Storage & p )
```

Here is the call graph for this function:



5.13.1.33 operator<<() [33/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const SSD & p )
```

Here is the call graph for this function:

5.13.1.34 operator<<() [34/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const SSD & p )
```

Here is the call graph for this function:

5.13.1.35 operator<<() [35/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const SSD & p )
```

Here is the call graph for this function:

5.13.1.36 operator<<() [36/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const SSD & p )
```

Here is the call graph for this function:

5.13.1.37 operator<<() [37/40]

```
std::ostream& operator<< (
            std::ostream & os,
            const HDD & p )
```

Here is the call graph for this function:

5.13.1.38 operator<<() [38/40]

```
std::ostream& operator<< (
            utos_ostream tos,
            const HDD & p )
```

Here is the call graph for this function:

5.13.1.39 operator<<() [39/40]

```
std::ostream& operator<< (
            simple_ostream tos,
            const HDD & p )
```

Here is the call graph for this function:

5.13.1.40 operator<<() [40/40]

```
std::ostream& operator<< (
            typ_ostream tos,
            const HDD & p )
```

Here is the call graph for this function:

5.13.1.41 setEnumfromString()

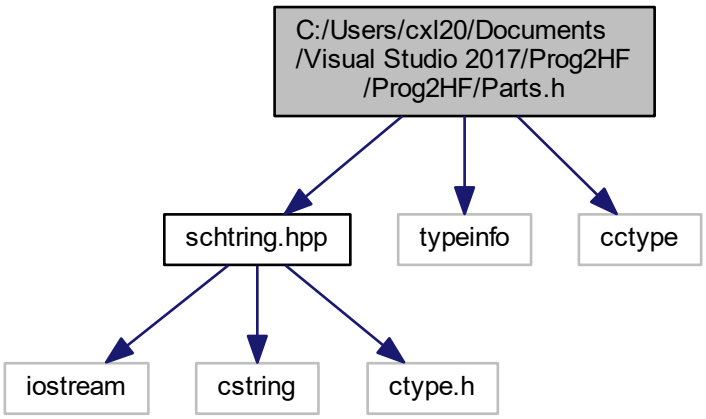
```
void setEnumfromString (
            String s0,
            enumPart & e )
```

Enumot állít be gy stringből.

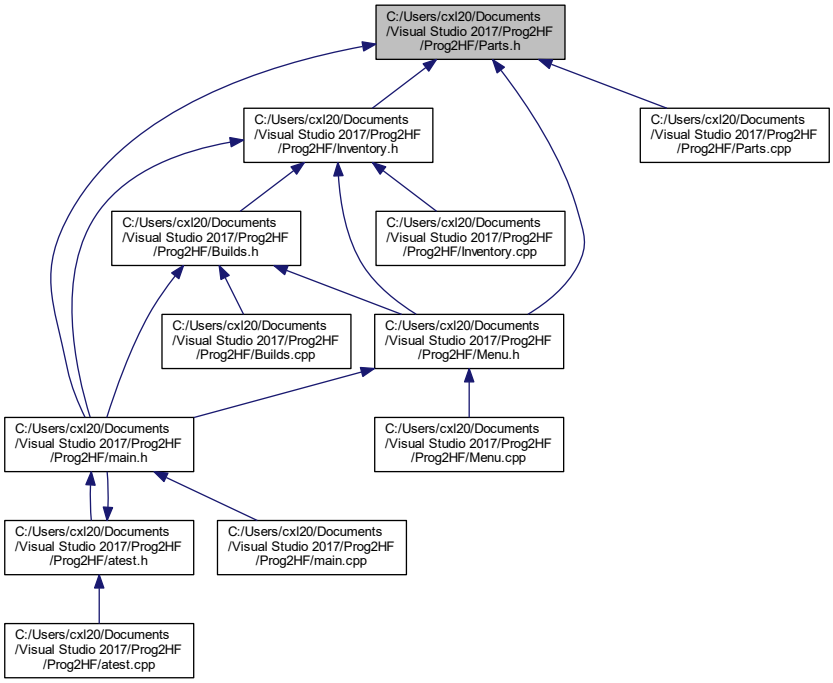
5.14 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/Parts.h File Reference

```
#include "schtring.hpp"
#include <typeinfo>
#include <cctype>
```

Include dependency graph for Parts.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct TempInput

  Lehetséges inputokat tárolja adatokkal való konstruáláshoz.
- class Part

  Alap alkatrész típus.
- class CPU

  Processzor.
- class GPU

  Videókártya.
- class MOBO

  Alaplap.
- class RAM

  Memória.
- class Case

  Ház.
- class PSU

  Táp.
- class Storage

  Tárhely alap.
- class SSD

  SSD.
- class HDD

  Merevlemez.

Enumerations

- enum enumPart {
  eInvalid = 0, eCPU = 1, eGPU = 2, eMOBO = 3,
  eRAM = 4, eCase = 5, ePSU = 6, eSSD = 7,
  eHDD = 8 }

  Betöltéshez segítő

Functions

- std::ostream & operator<< (std::ostream &, const Part &)
- std::ostream & operator<< (utos_ostream, const Part &)
- std::ostream & operator<< (simple_ostream, const Part &)
- std::ostream & operator<< (typ_ostream, const Part &)
- std::ostream & operator<< (std::ostream &, const CPU &)
- std::ostream & operator<< (utos_ostream, const CPU &)
- std::ostream & operator<< (simple_ostream, const CPU &)
- std::ostream & operator<< (typ_ostream, const CPU &)
- std::ostream & operator<< (std::ostream &, const GPU &)
- std::ostream & operator<< (utos_ostream, const GPU &)
- std::ostream & operator<< (simple_ostream, const GPU &)
- std::ostream & operator<< (typ_ostream, const GPU &)
- std::ostream & operator<< (std::ostream &, const MOBO &)
- std::ostream & operator<< (utos_ostream, const MOBO &)
- std::ostream & operator<< (simple_ostream, const MOBO &)
- std::ostream & operator<< (typ_ostream, const MOBO &)
- std::ostream & operator<< (std::ostream &, const RAM &)
- std::ostream & operator<< (utos_ostream, const RAM &)
- std::ostream & operator<< (simple_ostream, const RAM &)
- std::ostream & operator<< (typ_ostream, const RAM &)
- std::ostream & operator<< (std::ostream &, const Case &)
- std::ostream & operator<< (utos_ostream, const Case &)
- std::ostream & operator<< (simple_ostream, const Case &)
- std::ostream & operator<< (typ_ostream, const Case &)
- std::ostream & operator<< (std::ostream &, const PSU &)
- std::ostream & operator<< (utos_ostream, const PSU &)
- std::ostream & operator<< (simple_ostream, const PSU &)
- std::ostream & operator<< (typ_ostream, const PSU &)
- std::ostream & operator<< (std::ostream &, const Storage &)
- std::ostream & operator<< (utos_ostream, const Storage &)
- std::ostream & operator<< (simple_ostream, const Storage &)
- std::ostream & operator<< (typ_ostream, const Storage &)
- std::ostream & operator<< (std::ostream &, const SSD &)
- std::ostream & operator<< (utos_ostream, const SSD &)
- std::ostream & operator<< (simple_ostream, const SSD &)
- std::ostream & operator<< (typ_ostream, const SSD &)
- std::ostream & operator<< (std::ostream &, const HDD &)
- std::ostream & operator<< (utos_ostream, const HDD &)
- std::ostream & operator<< (simple_ostream, const HDD &)
- std::ostream & operator<< (typ_ostream, const HDD &)
- void setEnumfromString (String inst, enumPart &)

  Enumot állít be gy stringből.

5.14.1 Enumeration Type Documentation

5.14.1.1 enumPart

```
enum enumPart
```

Betöltéshez segítő

Enumerator

eInvalid	
eCPU	
eGPU	
eMOBO	
eRAM	
eCase	
ePSU	
eSSD	
eHDD	

5.14.2 Function Documentation

5.14.2.1 operator<<() [1/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const Part &  )
```

Here is the call graph for this function:

5.14.2.2 operator<<() [2/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const Part &  )
```

Here is the call graph for this function:

5.14.2.3 operator<<() [3/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const Part &  )
```

Here is the call graph for this function:

5.14.2.4 operator<<() [4/40]

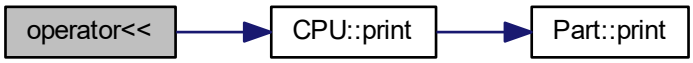
```
std::ostream& operator<< (
            typ_ostream ,
            const Part &  )
```

Here is the call graph for this function:

5.14.2.5 operator<<() [5/40]

```
std::ostream& operator<< (
             std::ostream & ,
             const CPU &  )
```

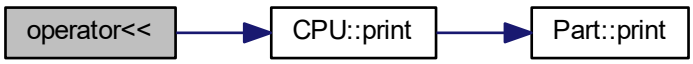
Here is the call graph for this function:



5.14.2.6 operator<<() [6/40]

```
std::ostream& operator<< (
             utos_ostream ,
             const CPU &  )
```

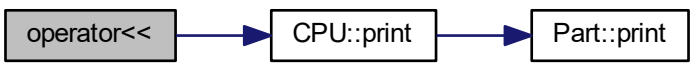
Here is the call graph for this function:



5.14.2.7 operator<<() [7/40]

```
std::ostream& operator<< (
             simple_ostream ,
             const CPU &  )
```

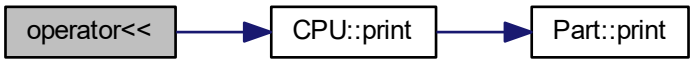
Here is the call graph for this function:



5.14.2.8 operator<<() [8/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const CPU &  )
```

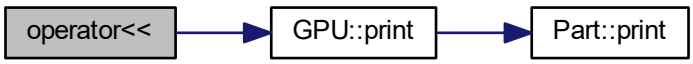
Here is the call graph for this function:



5.14.2.9 operator<<() [9/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const GPU &  )
```

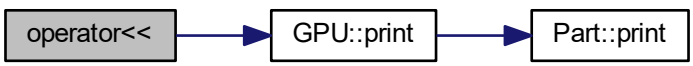
Here is the call graph for this function:



5.14.2.10 operator<<() [10/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const GPU &  )
```

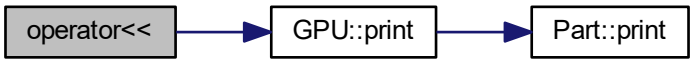
Here is the call graph for this function:



5.14.2.11 operator<<() [11/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const GPU &  )
```

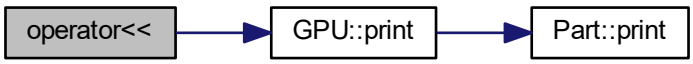
Here is the call graph for this function:



5.14.2.12 operator<<() [12/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const GPU &  )
```

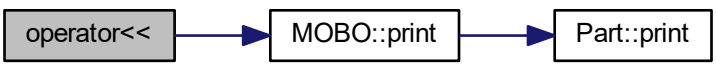
Here is the call graph for this function:



5.14.2.13 operator<<() [13/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const MOBO &  )
```

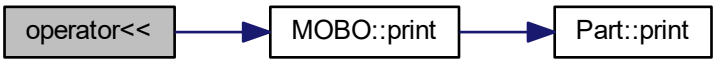
Here is the call graph for this function:



5.14.2.14 operator<<() [14/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const MOBO &  )
```

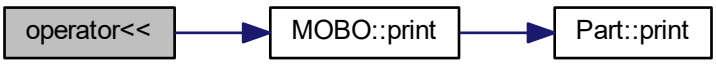
Here is the call graph for this function:



5.14.2.15 operator<<() [15/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const MOBO &  )
```

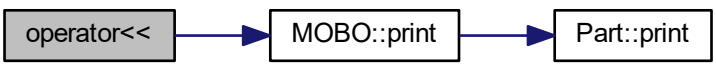
Here is the call graph for this function:



5.14.2.16 operator<<() [16/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const MOBO &  )
```

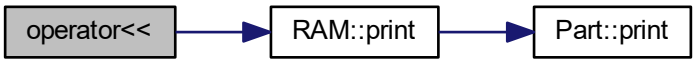
Here is the call graph for this function:



5.14.2.17 operator<<() [17/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const RAM &  )
```

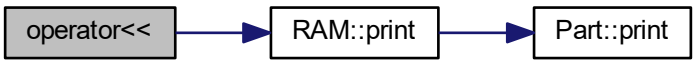
Here is the call graph for this function:



5.14.2.18 operator<<() [18/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const RAM &  )
```

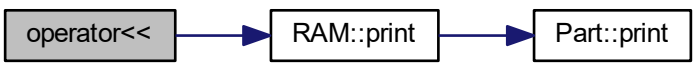
Here is the call graph for this function:



5.14.2.19 operator<<() [19/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const RAM &  )
```

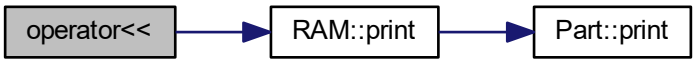
Here is the call graph for this function:



5.14.2.20 operator<<() [20/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const RAM &  )
```

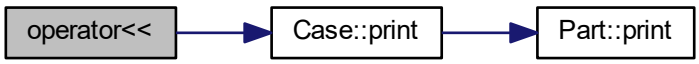
Here is the call graph for this function:



5.14.2.21 operator<<() [21/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const Case &  )
```

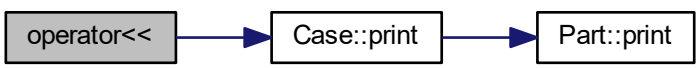
Here is the call graph for this function:



5.14.2.22 operator<<() [22/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const Case &  )
```

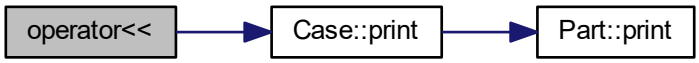
Here is the call graph for this function:



5.14.2.23 operator<<() [23/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const Case &  )
```

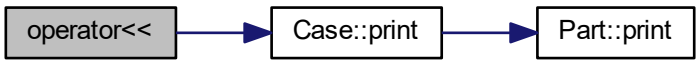
Here is the call graph for this function:



5.14.2.24 operator<<() [24/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const Case &  )
```

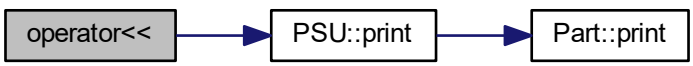
Here is the call graph for this function:



5.14.2.25 operator<<() [25/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const PSU &  )
```

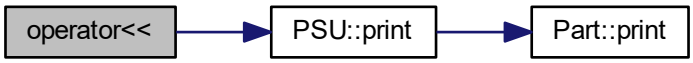
Here is the call graph for this function:



5.14.2.26 operator<<() [26/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const PSU &  )
```

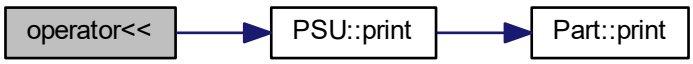
Here is the call graph for this function:



5.14.2.27 operator<<() [27/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const PSU &  )
```

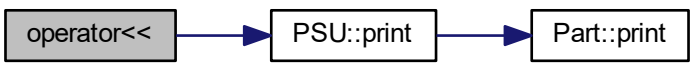
Here is the call graph for this function:



5.14.2.28 operator<<() [28/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const PSU &  )
```

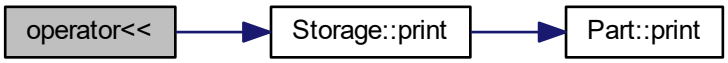
Here is the call graph for this function:



5.14.2.29 operator<<() [29/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const Storage &  )
```

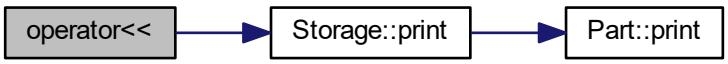
Here is the call graph for this function:



5.14.2.30 operator<<() [30/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const Storage &  )
```

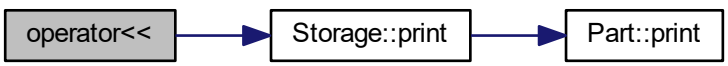
Here is the call graph for this function:



5.14.2.31 operator<<() [31/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const Storage &  )
```

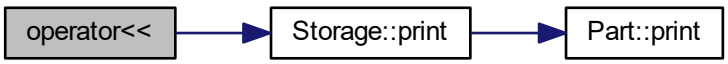
Here is the call graph for this function:



5.14.2.32 operator<<() [32/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const Storage &  )
```

Here is the call graph for this function:



5.14.2.33 operator<<() [33/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const SSD &  )
```

Here is the call graph for this function:

5.14.2.34 operator<<() [34/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const SSD &  )
```

Here is the call graph for this function:

5.14.2.35 operator<<() [35/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const SSD &  )
```

Here is the call graph for this function:

5.14.2.36 operator<<() [36/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const SSD &  )
```

Here is the call graph for this function:

5.14.2.37 operator<<() [37/40]

```
std::ostream& operator<< (
            std::ostream & ,
            const HDD &  )
```

Here is the call graph for this function:

5.14.2.38 operator<<() [38/40]

```
std::ostream& operator<< (
            utos_ostream ,
            const HDD &  )
```

Here is the call graph for this function:

5.14.2.39 operator<<() [39/40]

```
std::ostream& operator<< (
            simple_ostream ,
            const HDD &  )
```

Here is the call graph for this function:

5.14.2.40 operator<<() [40/40]

```
std::ostream& operator<< (
            typ_ostream ,
            const HDD &  )
```

Here is the call graph for this function:

5.14.2.41 setEnumfromString()

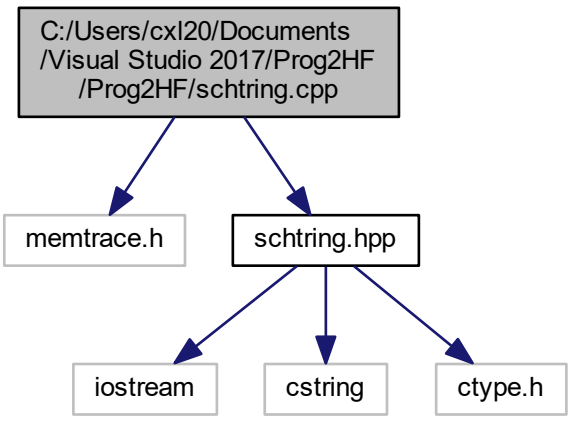
```
void setEnumfromString (
            String inst,
            enumPart &  )
```

Enumot állít be gy stringből.

5.15 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.cpp File Reference

```
#include "memtrace.h"
#include "schtring.hpp"
```

Include dependency graph for schtring.cpp:



Functions

- char * stolower (char *s)

  char tömb kisbetűsítése
- std::ostream & operator<< (std::ostream &os, const String &s0)

  inserter operator
- std::istream & operator>> (std::istream &is, String &s0)

  extractor operator
- std::ostream & operator<< (utos_ostream tos, const String &s0)

  alsóvonást szóközzé alakító kiírás

5.15.1 Function Documentation

5.15.1.1 operator<<() [1/2]

```
std::ostream& operator<< (
            std::ostream & os,
            const String & s0 )
```

inserter operator

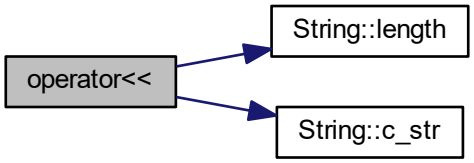
Here is the call graph for this function:

5.15.1.2 operator<<() [2/2]

```
std::ostream& operator<< (
            utos_ostream tos,
            const String & s0 )
```

alsóvonást szóközzé alakító kiírás

Here is the call graph for this function:



5.15.1.3 operator>>()

```
std::istream& operator>> (
            std::istream & is,
            String & s0 )
```

extractor operator

5.15.1.4 stolower()

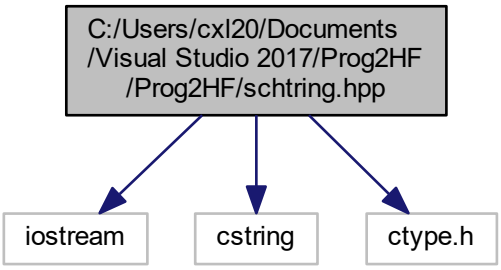
```
char* stolower (
            char * s )
```

char tömb kisbetűsítése

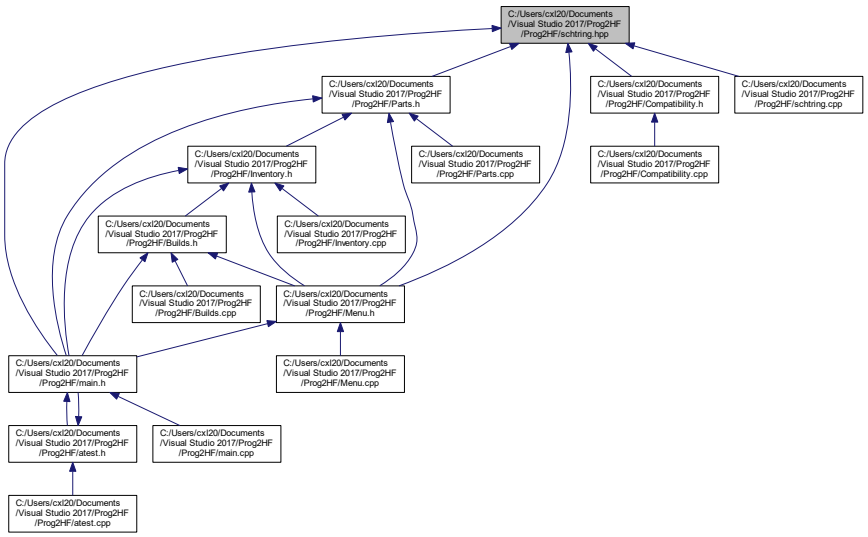
5.16 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/schtring.hpp File Reference

```
#include <iostream>
#include <cstring>
#include <ctype.h>
```

Include dependency graph for schtring.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- struct utos_t
  szóközösítő toggle
- struct simple_t
  csak paraméter toggle
- struct typ_t
  csak típus toggle
- struct utos_ostream
  szóközösítő stream manipulator
- struct simple_ostream
  csak paraméter stream manipulator
- struct typ_ostream
  csak típus stream manipulator
- class String

Functions

- utos_ostream operator<< (std::ostream &os, utos_t)
  szóközösítő stream manipulator
- simple_ostream operator<< (std::ostream &os, simple_t)
  csak paraméter stream manipulator
- typ_ostream operator<< (std::ostream &os, typ_t)
  csak típus stream manipulator
- template<typename T >
  std::ostream & operator<< (utos_ostream tos, const T &v)
  szóközösítő ostream
- template<typename T >
  std::ostream & operator<< (simple_ostream tos, const T &v)
  csak paraméter ostream
- template<typename T >
  std::ostream & operator<< (typ_ostream tos, const T &v)
  csak paraméter ostream
- char * stolower (char *s)
  char tömb kisbetűsítése
- String operator+ (char ch, const String &str)
  karakter + string
- std::ostream & operator<< (std::ostream &os, const String &s0)
  inserter operator
- std::istream & operator>> (std::istream &is, String &s0)
  extractor operator
- std::ostream & operator<< (utos_ostream tos, const String &s0)
  alsóvonást szóközzé alakító kiírás

Variables

- constexpr utos_t utos
  szóközösítő toggle
- constexpr simple_t simple
  csak paraméter toggle
- constexpr typ_t typ
  csak típus toggle

5.16.1 Function Documentation

5.16.1.1 operator+()

```
String operator+ (
            char ch,
            const String & str )  [inline]
```

karakter + string

5.16.1.2 operator<<() [1/8]

```
utos_ostream operator<< (
            std::ostream & os,
            utos_t  )  [inline]
```

szóközösítő stream manipulator

5.16.1.3 operator<<() [2/8]

```
simple_ostream operator<< (
            std::ostream & os,
            simple_t  )  [inline]
```

csak paraméter stream manipulator

5.16.1.4 operator<<() [3/8]

```
typ_ostream operator<< (
            std::ostream & os,
            typ_t  )  [inline]
```

csak típus stream manipulator

5.16.1.5 operator<<() [4/8]

```
template<typename T >
std::ostream& operator<< (
            utos_ostream tos,
            const T & v )
```

szóközösítő ostream

5.16.1.6 operator<<() [5/8]

```
template<typename T >
std::ostream& operator<< (
            simple_ostream tos,
            const T & v )
```

csak paraméter ostream

5.16.1.7 operator<<() [6/8]

```
template<typename T >
std::ostream& operator<< (
            typ_ostream tos,
            const T & v )
```

csak paraméter ostream

5.16.1.8 operator<<() [7/8]

```
std::ostream& operator<< (
            std::ostream & os,
            const String & s0 )
```

inserter operator

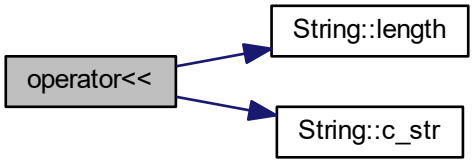
Here is the call graph for this function:

5.16.1.9 operator<<() [8/8]

```
std::ostream& operator<< (
            utos_ostream tos,
            const String & s0 )
```

alsóvonást szóközzé alakító kiírás

Here is the call graph for this function:



5.16.1.10 operator>>()

```
std::istream& operator>> (
            std::istream & is,
            String & s0 )
```

extractor operator

5.16.1.11 stolower()

```
char* stolower (
            char * s )
```

char tömb kisbetűsítése

5.16.2 Variable Documentation

5.16.2.1 simple

```
constexpr simple_t simple
```

csak paraméter toggle

5.16.2.2 typ

```
constexpr typ_t typ
```

csak típus toggle

5.16.2.3 utos

```
constexpr utos_t utos
```

szóközösítő toggle

5.17 C:/Users/cxl20/Documents/Visual Studio 2017/Prog2HF/Prog2HF/SFML_test.cpp File Reference

Index