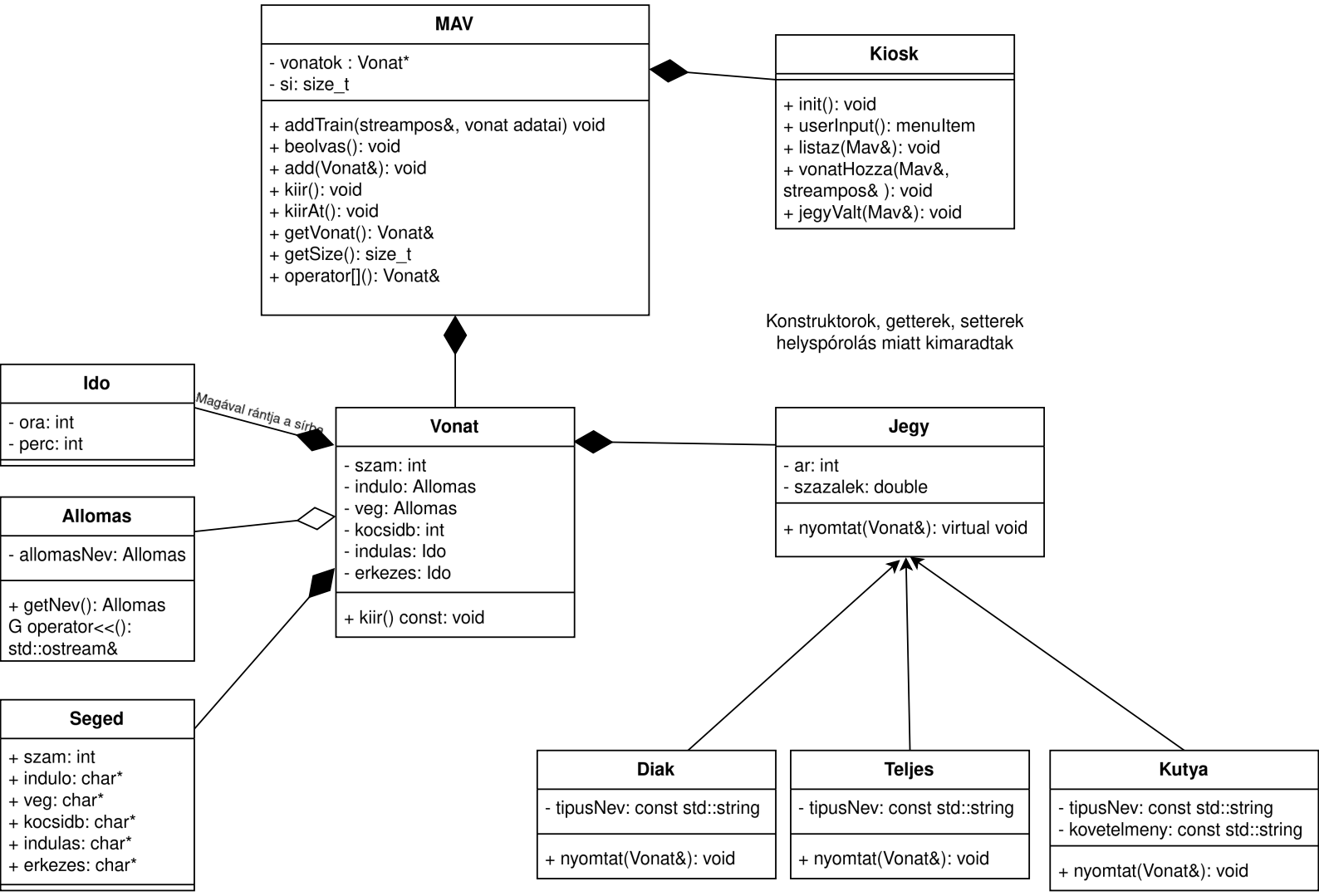
Házi feladat

1.0

Github

Hujber Hunor
YJTXDO

Generated by Doxygen 1.10.0


MAV
- vonatok : Vonat*
- si: size_t
+ addTrain(streampos&, vonat adatai) void
+ beolvas(): void
+ add(Vonat&): void
+ kiir(): void
+ kiirAt(): void
+ getVonat(): Vonat&
+ getSize(): size_t
+ operator[](): Vonat&
Kiosk
+ init(): void
+ userInput(): menuItem
+ listaz(Mav&): void
+ vonatHozza(Mav&, streampos&): void
+ jegyValt(Mav&): void
Konstruktorok, getterek, setterek helyspórolás miatt kimaradtak
Ido
- ora: int
- perc: int
Magával rántja a sírba
Vonat
- szam: int
- indulo: Allomas
- veg: Allomas
- kocsidb: int
- indulas: Ido
- erkezes: Ido
+ kiir() const: void
Jegy
- ar: int
- szazalek: double
+ nyomtat(Vonat&): virtual void
Allomas
- allomasNev: Allomas
+ getNev(): Allomas
G operator<<(): std::ostream&
Seged
+ szam: int
+ indulo: char*
+ veg: char*
+ kocsidb: char*
+ indulas: char*
+ erkezes: char*
Diak
- tipusNev: const std::string
+ nyomtat(Vonat&): void
Teljes
- tipusNev: const std::string
+ nyomtat(Vonat&): void
Kutya
- tipusNev: const std::string
- kovetelmeny: const std::string
+ nyomtat(Vonat&): void

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

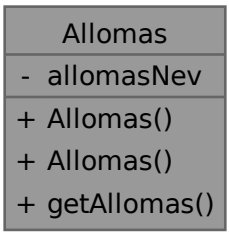
Chapter 4

Data Structure Documentation

4.1 Allomas Class Reference

`#include <allomas.h>`

Collaboration diagram for Allomas:



Public Member Functions

- Allomas (const char *allomas)

  Allomas nevét tartalmazo string.
- Allomas (char *allomas)
- std::string getAllomas () const

Private Attributes

- std::string allomasNev

4.1.1 Detailed Description

STRINGGEL ÚJRAÍRVA

Definition at line 12 of file allomas.h.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 Allomas() [1/2]

```
Allomas::Allomas (
            const char * allomas )  [inline]
```

Allomas nevét tartalmazo string.

Allomas konstrukor

Parameters

allomas	itt egy char* egyértelműen. a másik verzióban const, hogy fel lehessen közvetlen tölteni értsd: "Álloms_neve" mint paraméter.

Definition at line 22 of file allomas.h.

```
00022 : allomasNev(allomas) {}
```

4.1.2.2 Allomas() [2/2]

```
Allomas::Allomas (
            char * allomas )  [inline]
```

Definition at line 23 of file allomas.h.

```
00023 : allomasNev(allomas) {}
```

4.1.3 Member Function Documentation

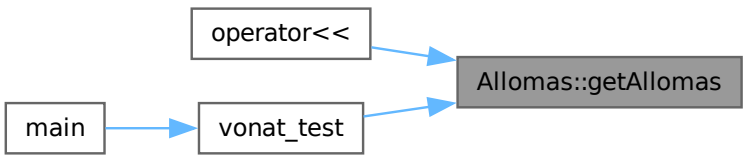
4.1.3.1 getAllomas()

```
std::string Allomas::getAllomas ( ) const  [inline]
```

Definition at line 27 of file allomas.h.

```
00027 { return allomasNev; }
```

Here is the caller graph for this function:



4.1.4 Field Documentation

4.1.4.1 allomasNev

```
std::string Allomas::allomasNev  [private]
```

Definition at line 14 of file allomas.h.

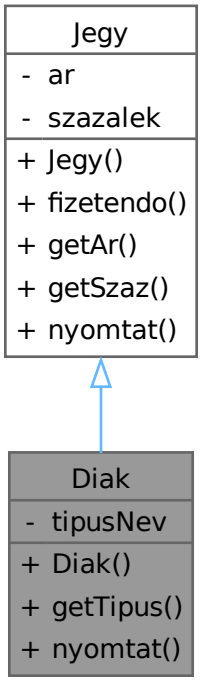
The documentation for this class was generated from the following file:

- allomas.h

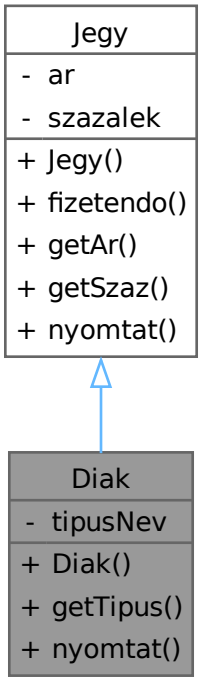
4.2 Diak Class Reference

```
#include <jegy.h>
```

Inheritance diagram for Diak:



Collaboration diagram for Diak:



Public Member Functions

- Diak (int ar=1200, const double szazalek=0.7)
- std::string getTipus ()

  Getterek.
- void nyomtat (Vonat &v)

Public Member Functions inherited from Jegy

- Jegy (int ar=1200, double szazalek=0.7)

  Kedvezmény százaléka, pontosabban mennyi az amit fizet százalék.
- virtual int fizetendo ()

  Fizetendő összeg, ár és a kedvezmény szorzata.
- int getAr ()

  Getterek.
- double getSzaz ()

Private Attributes

- const std::string tipusNev = "Diák"

4.2.1 Detailed Description

Definition at line 48 of file jegy.h.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 Diak()

```
Diak::Diak (
            int ar = 1200,
            const double szazalek = 0.7 )  [inline]
```

Konstruktor

Parameters

ar	megadja a jegy árát
szazalek	a kedvezmény értékét adja meg Meghívja a anya-class konstruktorát.

Definition at line 59 of file jegy.h.

```
00059 : Jegy(ar, szazalek) {}
```

4.2.3 Member Function Documentation

4.2.3.1 getTipus()

```
std::string Diak::getTipus ( )  [inline]
```

Getterek.

Definition at line 62 of file jegy.h.

```
00062 { return tipusNev; }
```

Here is the caller graph for this function:

4.2.3.2 nyomtat()

```
void Diak::nyomtat (
            Vonat & v )  [inline], [virtual]
```

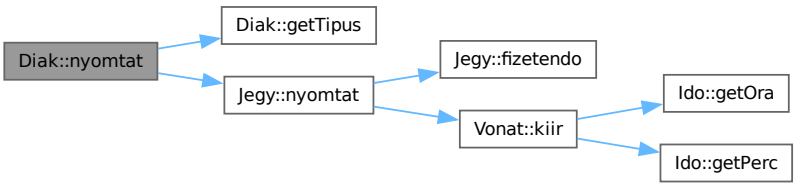
megörökölt nyomtat fv overwrite-ja meghívja az "eredei", főosztály nyomtatját

Reimplemented from Jegy.

Definition at line 66 of file jegy.h.

```
00066                               {
00067     std::cout « "### Jegy adatai ###\n";
00068     std::cout « "Jegy típusa: " « getTipus() « std::endl;
00069     Jegy::nyomtat(v);
00070   }
```

Here is the call graph for this function:



Here is the caller graph for this function:

4.2.4 Field Documentation

4.2.4.1 tipusNev

```
const std::string Diak::tipusNev = "Diák"  [private]
```

Definition at line 49 of file jegy.h.

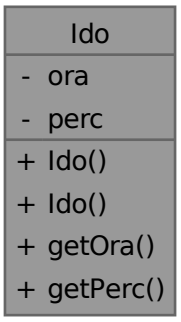
The documentation for this class was generated from the following file:

- jegy.h

4.3 Ido Class Reference

`#include <ido.h>`

Collaboration diagram for Ido:



Public Member Functions

- Ido (const char *ido)
- Ido (int ora, int perc)
- int getOra () const
- int getPerc () const

Private Attributes

- int ora
- int perc

4.3.1 Detailed Description

Definition at line 12 of file ido.h.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Ido() [1/2]

```
Ido::Ido (
            const char * ido )  [inline]
```

Konstruktor

Parameters

ido	mivel beolvasáskor char* típusokba olvasok bele char*-ot kap paraméterként és azt kezeli megfelelően konvertálva.

Definition at line 21 of file ido.h.

```
00021                         {
00022     ora = (ido[0] - '0') * 10 + (ido[1] - '0');
00023     perc = (ido[2] - '0') * 10 + (ido[3] - '0');
00024   }
```

4.3.2.2 Ido() [2/2]

```
Ido::Ido (
            int ora,
            int perc )  [inline]
```

Ido Konstruktor

Parameters

ora	(int)
perc	(int) Főképp tesztekhez volt használva.

Definition at line 32 of file ido.h.

```
00032 : ora(ora), perc(perc) {}
```

4.3.3 Member Function Documentation

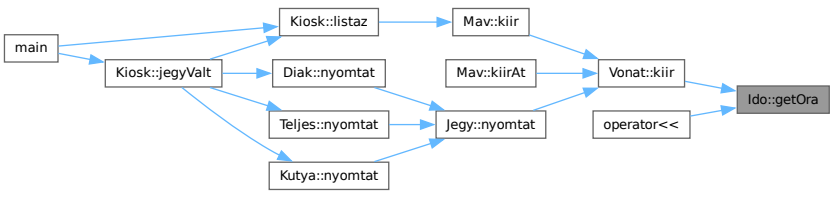
4.3.3.1 getOra()

```
int Ido::getOra ( ) const  [inline]
```

Definition at line 35 of file ido.h.

```
00035 { return ora; };
```

Here is the caller graph for this function:



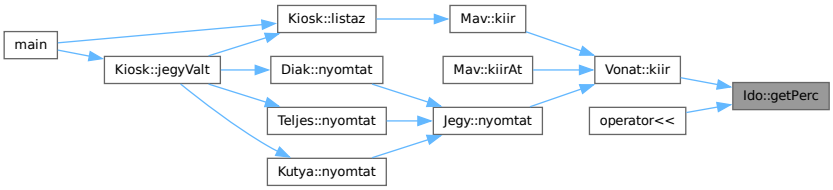
4.3.3.2 getPerc()

```
int Ido::getPerc ( ) const  [inline]
```

Definition at line 36 of file ido.h.

```
00036 {return perc; };
```

Here is the caller graph for this function:



4.3.4 Field Documentation

4.3.4.1 ora

```
int Ido::ora  [private]
```

Definition at line 13 of file ido.h.

4.3.4.2 perc

```
int Ido::perc  [private]
```

Definition at line 14 of file ido.h.

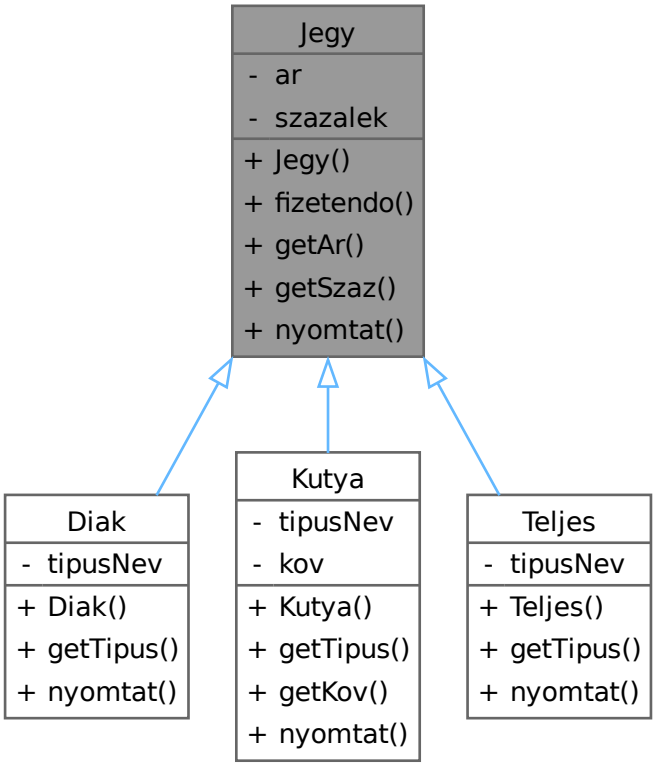
The documentation for this class was generated from the following file:

- ido.h

4.4 Jegy Class Reference

```
#include <jegy.h>
```

Inheritance diagram for Jegy:



Collaboration diagram for Jegy:

Public Member Functions

- Jegy (int ar=1200, double szazalek=0.7)

  Kedvezmény százaléka, pontosabban mennyi az amit fizet százalék.
- virtual int fizetendo ()

  Fizetendő összeg, ár és a kedvezmény szorzata.
- int getAr ()

  Getterek.
- double getSzaz ()
- virtual void nyomtat (Vonat &v)

Private Attributes

- int ar
- double szazalek

  A jegy ára.

4.4.1 Detailed Description

FILE JEGY_H

Definition at line 11 of file jegy.h.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 Jegy()

```
Jegy::Jegy (
            int ar = 1200,
            double szazalek = 0.7 )  [inline]
```

Kedvezmény százaléka, pontosabban mennyi az amit fizet százalék.

Anya osztály Konstruktora

Parameters

ar	megadja a jegy árát
szazalek	a kedvezmény értékét adja meg

Definition at line 21 of file jegy.h.

```
00021 : ar(ar), szazalek(szazalek)  {}
```

4.4.3 Member Function Documentation

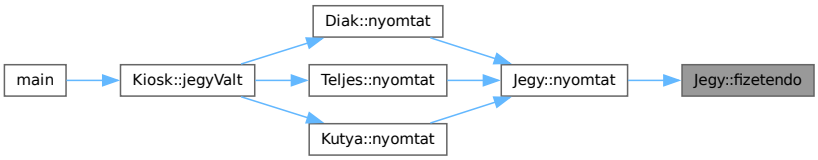
4.4.3.1 fizetendo()

```
virtual int Jegy::fizetendo ( )  [inline], [virtual]
```

Fizetendő összeg, ár és a kedvezmény szorzata.

Definition at line 24 of file jegy.h.
```
00024 { return ar * szazalek; }
```

Here is the caller graph for this function:



4.4.3.2 getAr()

```
int Jegy::getAr ( )  [inline]
```

Getterek.

Definition at line 27 of file jegy.h.
```
00027 { return ar; }
```

4.4.3.3 getSzaz()

```
double Jegy::getSzaz ( )  [inline]
```

Definition at line 28 of file jegy.h.
```
00028 { return szazalek; }
```

4.4.3.4 nyomtat()

```
virtual void Jegy::nyomtat (
            Vonat & v )  [inline], [virtual]
```

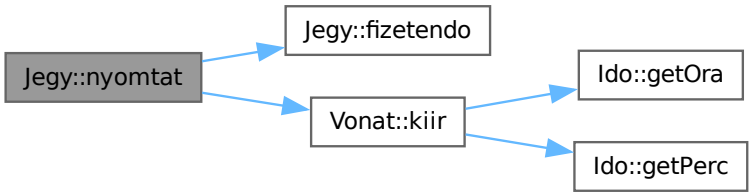
Parameters

Vonat&	megadott vonatra vonatkozik Kíírja a jegy adatinak egy felét. Azt a felét ami minden leszármazottál azonos. A felső pár so. virtual mivel a leszármazottak megöröklik és felülírják.

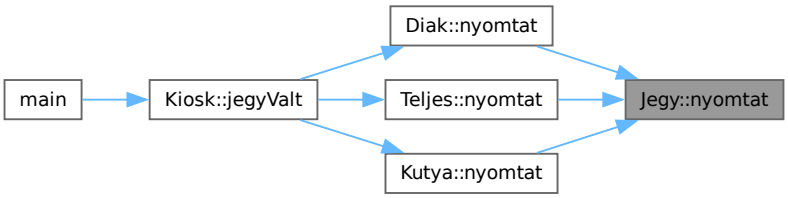
Reimplemented in Diak, Teljes, and Kutya.

Definition at line 36 of file jegy.h.
```
00036                                {
00037     std::cout « "Fizetendő: " « fizetendo() « "JMF\n";
00038     std::cout « "### Vonatod adatai ###" « std::endl;
00039     v.kiir();
00040     std::cout « std::endl;
00041   }
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.4 Field Documentation

4.4.4.1 ar

```
int Jegy::ar  [private]
```

Definition at line 12 of file jegy.h.

4.4.4.2 szazalek

```
double Jegy::szazalek  [private]
```

A jegy ára.

Definition at line 13 of file jegy.h.

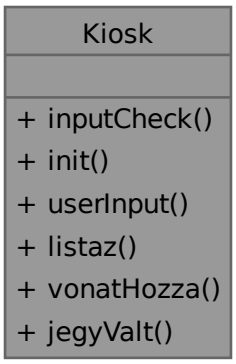
The documentation for this class was generated from the following file:

- jegy.h

4.5 Kiosk Class Reference

`#include <kiosk.h>`

Collaboration diagram for Kiosk:



Public Member Functions

- bool inputCheck ()

  Hibakezelés. Érvényes input esetén.
- void init ()

  Inicializálja a menüt, ergo kiírja a lehetőségeket.
- menuItem userInput ()
- void listaz (Mav &mav)

  Liszázza a vonatokat amiket beolvasott a fájlból.
- void vonatHozza (Mav &mav, std::streampos currPos)

  Hozzá ad a user vonatokat a fájlhoz.
- void jegyValt (Mav &mav)

  jegyet vált a kiválasztott vonatra

4.5.1 Detailed Description

Definition at line 23 of file kiosk.h.

4.5.2 Member Function Documentation

4.5.2.1 init()

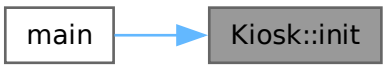
```
void Kiosk::init ( )
```

Inicializálja a menüt, ergo kiírja a lehetőségeket.

Definition at line 27 of file kiosk.cpp.

```
00027                {
00028   std::cout « "### Dönts ###" « std::endl;
00029   std::cout « "1. Vonat hozzáadása "     « std::endl;
00030   std::cout « "2. Vonatok listázása "    « std::endl;
00031   std::cout « "3. Jegy nyomtatása"       « std::endl;
00032   std::cout « "4. Kilépés"               « std::endl;
00033   std::cout « "5. Teszt"                 « std::endl;
00034 }
```

Here is the caller graph for this function:



4.5.2.2 inputCheck()

```
bool Kiosk::inputCheck ( )
```

Hibakezelés. Érvényes input esetén.

Returns

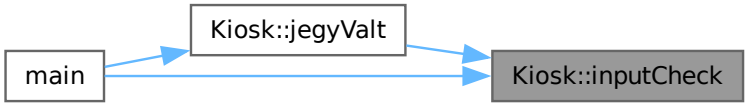
true, különben

false

Definition at line 15 of file kiosk.cpp.

```
00015                      {
00016   // Kb innen lopva
     https://stackoverflow.com/questions/12721911/c-how-to-verify-if-the-data-input-is-of-the-correct-datatype
00017   if (std::cin.fail()) {
00018         std::cout « "Nem sikerült érvényes inputot megadni... PRÓBÁLD ÚJRA" « std::endl;
00019         std::cin.clear();
00020         std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
00021         return false;
00022   }
00023   else return true;
00024 }
```

Here is the caller graph for this function:



4.5.2.3 jegyValt()

```
void Kiosk::jegyValt (
            Mav & mav )
```

jegyet vált a kiválasztott vonatra

Jegy vásárlás, nyomtatás

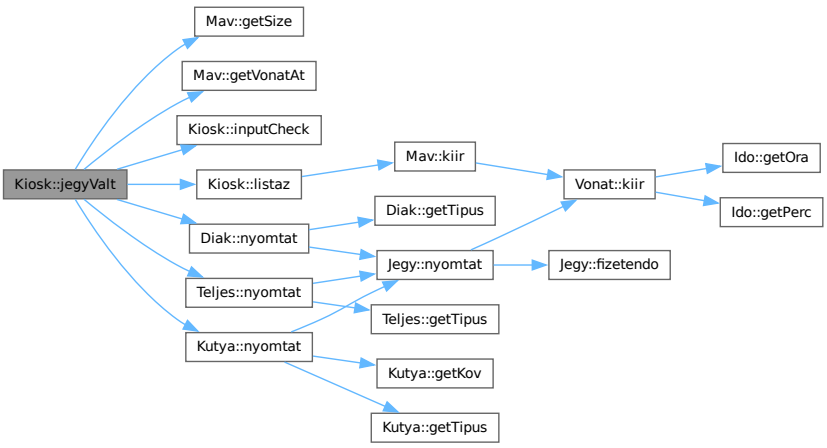
Parameters

mav	mav refereancia, ez az osztály tartalmazza a vonatok* tömböt

Definition at line 76 of file kiosk.cpp.

```
00076                                {
00077   int idx, buf;
00078   listaz(mav);
00079   std::cout « "Valasz vonatot: ";
00080   std::cin » idx;
00081   // Hibakezelés
00082   if(size_t(idx) > mav.getSize()){ std::cout « "Túlindexelés\n"; return; }
00083
00084   if(!(Kiosk::inputCheck())) return; // Érvényes inputot szűrök.
00085
00086   std::cout « std::endl; // szép kiírás miatt.
00087   // end if Hibakezelés
00088
00089
00090   std::cout « "1.Teljes, 2.Diak, 3.Kutya\n ";
00091   std::cin » buf;
00092   // Hibakezelés
00093   if(buf > 3){ std::cout « "Túlindexelés"; return; }
00094
00095   if(!(Kiosk::inputCheck())) return;
00096
00097   std::cout « std::endl;
00098   // end of Hibakezelés
00099
00100   switch (buf) {
00101     case 1: {
00102       Teljes t;
00103       t.nyomtat(mav.getVonatAt(idx));
00104       break;
00105     }
00106
00107     case 2: {
00108       Diak d;
00109       d.nyomtat(mav.getVonatAt(idx));
00110       break;
00111     }
00112
00113     case 3: {
00114       Kutya k;
00115       k.nyomtat(mav.getVonatAt(idx));
00116       break;
00117     }
00118
00119   } // end of switch
00120
00121 } // end of jegyValt
```

Here is the call graph for this function:



Here is the caller graph for this function:

4.5.2.4 listaz()

```
void Kiosk::listaz (
            Mav & mav )
```

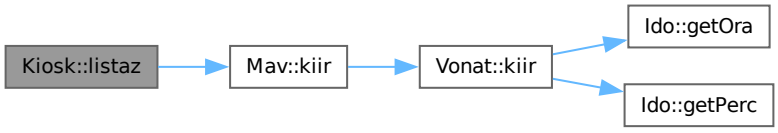
Liszázza a vonatokat amiket beolvasott a fájlból.

Liszázza az összes vonatot.

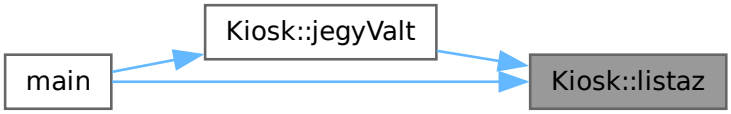
Definition at line 47 of file kiosk.cpp.

```
00047                                {
00048   mav.kiir();
00049 }
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.2.5 userInput()

```
menuItem Kiosk::userInput ( )
```

Felhasználótól bemenetet kér.

Returns

menuItem fent említett enum class elemt ad vissza

Definition at line 39 of file kiosk.cpp.

```
00039                           {
00040     int valasz;
00041     std::cout « "Választott lehetőség: ";
00042     std::cin » valasz;
00043     return static_cast<menuItem>(valasz);
00044 }
```

Here is the caller graph for this function:

4.5.2.6 vonatHozza()

```
void Kiosk::vonatHozza (
            Mav & mav,
            std::streampos currPos )
```

Hozzá ad a user vonatokat a fájlhoz.

User hozzáadhat vonatot a file-hoz

Parameters

mav	a mav osztályra refereancia, ezt töltöm fel (Mav)
currpos	a kurzor éppenleges helye, ahova írni kell, hogy ne legyen felülírás (streampos)

Definition at line 57 of file kiosk.cpp.

```
00057                                                   {
00058   Seged s;
00059
00060   std::cin » s.szam;
00061   // if(*(s.szam) == '\n') std::cout « "FASZ";
00062   std::cin » s.indulo;
00063   std::cin » s.veg;
00064   std::cin » s.kocsidb;
00065   std::cin » s.indulas;
00066   std::cin » s.erkezes;
00067
00068   mav.addTrain(currPos, atoi(s.szam), Allomas(s.indulo), Allomas(s.veg),
00069                atoi(s.kocsidb), Ido(s.indulas), Ido(s.erkezes));
00070 } // End of vonatHozza
```

Here is the call graph for this function:

Here is the caller graph for this function:

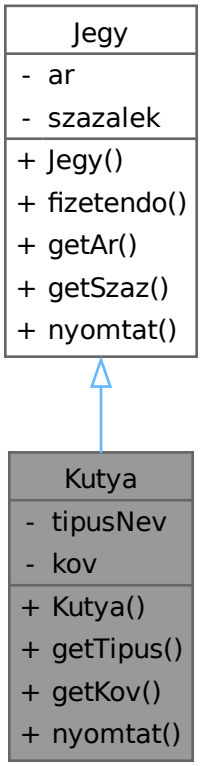
The documentation for this class was generated from the following files:

- kiosk.h
- kiosk.cpp

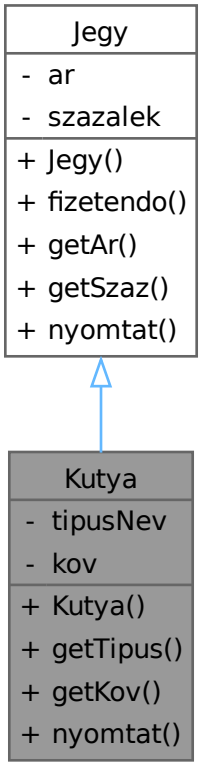
4.6 Kutya Class Reference

`#include <jegy.h>`

Inheritance diagram for Kutya:



Collaboration diagram for Kutya:



Public Member Functions

- Kutya (int ar=1200, const double szazalek=0.2)
- std::string getTipus ()
- std::string getKov ()
- void nyomtat (Vonat &v)

Public Member Functions inherited from Jegy

- Jegy (int ar=1200, double szazalek=0.7)

  Kedvezmény százaléka, pontosabban mennyi az amit fizet százalék.
- virtual int fizetendo ()

  Fizetendő összeg, ár és a kedvezmény szorzata.
- int getAr ()

  Getterek.
- double getSzaz ()

Private Attributes

- const std::string tipusNev = "Kutya"
- const std::string kov = "Szájkosár"

4.6.1 Detailed Description

Definition at line 101 of file jegy.h.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 Kutya()

```
Kutya::Kutya (
            int ar = 1200,
            const double szazalek = 0.2 )  [inline]
```

Konstruktor

Parameters

ar	megadja a jegy árát
szazalek	a kedvezmény értékét adja meg Meghívja a anya-class konstruktorát.

Definition at line 112 of file jegy.h.

```
00112 : Jegy(ar, szazalek) {}
```

4.6.3 Member Function Documentation

4.6.3.1 getKov()

```
std::string Kutya::getKov ( )  [inline]
```

Definition at line 116 of file jegy.h.

```
00116 { return kov; }
```

Here is the caller graph for this function:

4.6.3.2 getTipus()

```
std::string Kutya::getTipus ( )  [inline]
```

Definition at line 115 of file jegy.h.

```
00115 { return tipusNev; }
```

Here is the caller graph for this function:

4.6.3.3 nyomtat()

```
void Kutya::nyomtat (
            Vonat & v )  [inline], [virtual]
```

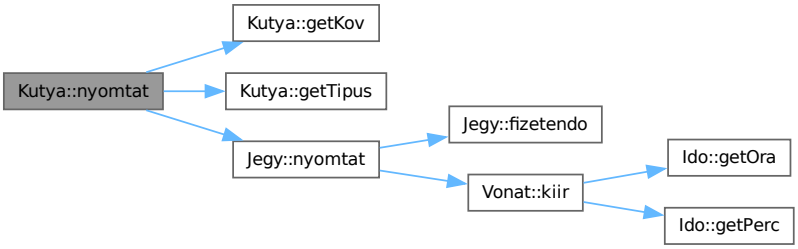
megörökölt nyomtat fv overwrite-ja meghívja az "eredei", főosztály nyomtatját

Reimplemented from Jegy.

Definition at line 120 of file jegy.h.

```
00120                           {
00121     std::cout « "### Jegy adatai ###\n";
00122     std::cout « "Jegy típusa: " « getTipus() « std::endl;
00123     std::cout « "Követelmény: " « getKov() « std::endl;
00124     Jegy::nyomtat(v);
00125   }
```

Here is the call graph for this function:



Here is the caller graph for this function:

4.6.4 Field Documentation

4.6.4.1 kov

```
const std::string Kutya::kov = "Szájkosár"  [private]
```

Definition at line 103 of file jegy.h.

4.6.4.2 tipusNev

```
const std::string Kutya::tipusNev = "Kutya"  [private]
```

Definition at line 102 of file jegy.h.

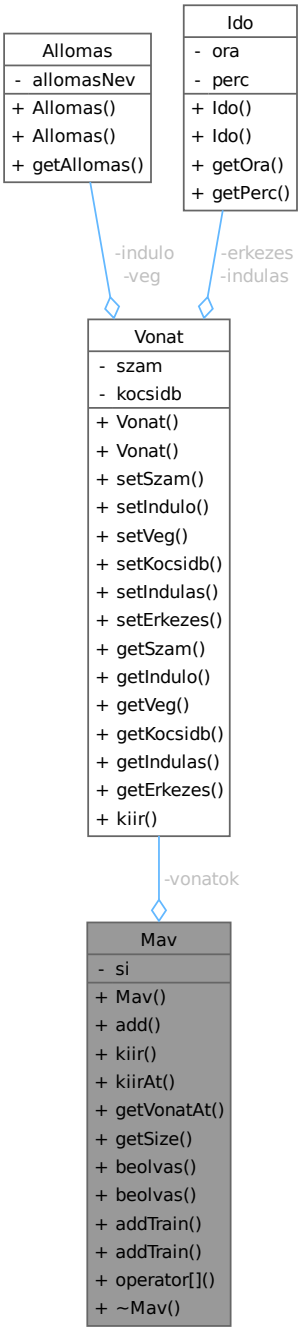
The documentation for this class was generated from the following file:

- jegy.h

4.7 Mav Class Reference

```
#include <mav.h>
```

Collaboration diagram for Mav:



Public Member Functions

- Mav (size_t si=0)
- void add (Vonat &vonat)
- void kiir ()
- void kiirAt (int i)
- Vonat & getVonatAt (int i)

- size_t getSize ()
- void beolvas ()
- void beolvas (const char *file)
- void addTrain (std::streampos &currPos, int szam, Allomas indulo, Allomas veg, int kocsidb, Ido indulas, Ido erkezes)
- void addTrain (std::streampos &currPos, const char *test, int szam, Allomas indulo, Allomas veg, int kocsidb, Ido indulas, Ido erkezes)
- Vonat & operator[ ] (int idx)

  op[] overload
- ~Mav ()

  Destruktor.

Private Attributes

- Vonat * vonatok
- size_t si

4.7.1 Detailed Description

Definition at line 16 of file mav.h.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 Mav()

```
Mav::Mav (
            size_t si = 0 )  [inline]
```

Definition at line 22 of file mav.h.

```
00022 : vonatok(nullptr), si(si) {}
```

4.7.2.2 ~Mav()

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

Destruktor.

Definition at line 95 of file mav.h.

```
00095         {
00096     delete[] vonatok;
00097   }
```

4.7.3 Member Function Documentation

4.7.3.1 add()

```
void Mav::add (
            Vonat & vonat )  [inline]
```

Elem hozzáfűzése Vonatok kollekcióhoz

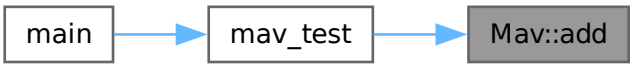
Parameters

vonat	vonat referencia amit hozzáad

Definition at line 28 of file mav.h.

```
00028                          {
00029     Vonat* temp = new Vonat[si + 1];  // Lefoglalok helyet
00030
00031     for (size_t i = 0; i < si; ++i) { // Átmásolom a tömb elemeit
00032       temp[i] = vonatok[i];
00033     }
00034     temp[si++] = vonat;               // Beleteszem a vonatot
00035
00036     delete[] vonatok;                   // "Régi" tömböt törlöm
00037     vonatok = temp;                     //
00038   } // End of feltolt
```

Here is the caller graph for this function:



4.7.3.2 addTrain() [1/2]

```
void Mav::addTrain (
              std::streampos & currPos,
              const char * test,
              int szam,
              Allomas indulo,
              Allomas veg,
              int kocsidb,
              Ido indulas,
              Ido erkezes )
```

addTrain() teszt miatti overwriteja Csak abban különbözik, hogy a fájl amit megnyit nem a hard coded hanem a test file

Definition at line 105 of file mav.cpp.

```
00106                                                                   {
00107   // Inicalizálás
00108   const char* vonatok = test;
00109   std::ofstream file (vonatok, std::ios::app); // Hozzáfűzésesen nyitom meg.
00110
00111   if(file.is_open()){
00112     file.seekp(currPos); // Megfelelő helyre ugrok
00113
00114     file « szam « " " « indulo « " " « veg « " " « kocsidb « " "
00115          « indulas « " " « erkezes « std::endl;
00116   }
00117
00118   file.close();
00119 } // END OF addTrain
```

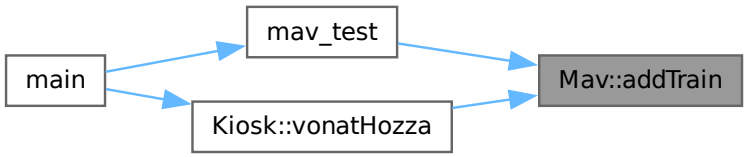
4.7.3.3 addTrain() [2/2]

```
void Mav::addTrain (
              std::streampos & currPos,
              int szam,
              Allomas indulo,
              Allomas veg,
              int kocsidb,
              Ido indulas,
              Ido erkezes )
```

Definition at line 77 of file mav.cpp.

```
00078                                                                    {
00079   // Inicalizálás
00080   const char* vonatok = "./vonatok.txt";
00081   std::ofstream file (vonatok, std::ios::app); // Hozzáfűzésesen nyitom meg.
00082
00083   // HIBAKEZELÉS IDE
00084   if (!file.is_open()) {
00085     std::cout « "Megynitással van baj " « vonatok « std::endl;
00086     return;
00087   }
00088   // end of HIBAKEZELÉS
00089
00090   if(file.is_open()){
00091     file.seekp(currPos); // Megfelelő helyre ugrok
00092
00093     file « szam « " " « indulo « " " « veg « " " « kocsidb « " "
00094          « indulas « " " « erkezes « std::endl;
00095   }
00096
00097   file.close();
00098 } // END OF addTrain
```

Here is the caller graph for this function:



4.7.3.4 beolvas() [1/2]

```
void Mav::beolvas ( )
```

Beolvaásás a file-ból egy segéd strukturába, onnan pedig egy Vonat objektum feltöltés

Parameters

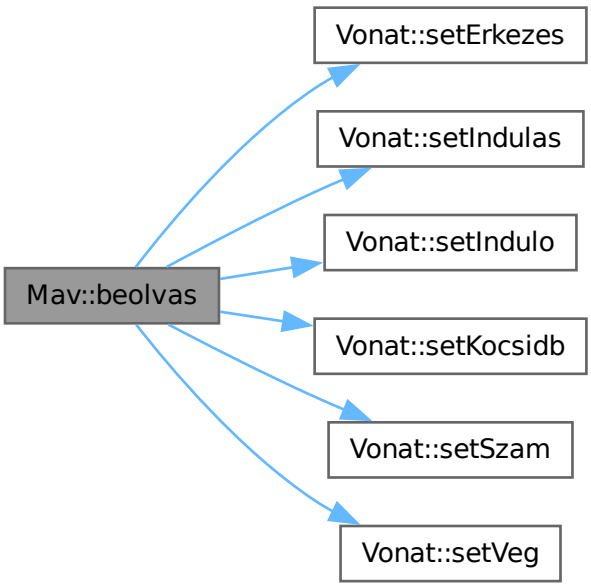
Mav&	mav class feltöltéséhez szükséges

Definition at line 9 of file mav.cpp.

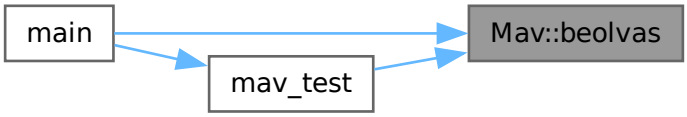
```
00009                 {
```

```
00010   // Inicalizálás
00011   const char* vonatok_file = "./vonatok.txt";
00012   std::ifstream file(vonatok_file);
00013   Vonat v;
00014   Seged seged;
00015
00016   if (!file.is_open()) {
00017     std::cout « "Megynitással van baj " « vonatok_file « std::endl;
00018     return;
00019   }
00020
00021   // Kiürítem a vonatok kollekciót, hogy ne legyen ráolvasás
00022   delete[] vonatok;
00023   vonatok = nullptr;
00024   si = 0;
00025
00026   while (file » seged.szam » seged.indulo » seged.veg » seged.kocsidb » seged.indulas » seged.erkezes)
      {
00027     v.setSzam(std::atoi(seged.szam));
00028     v.setIndulo(seged.indulo);
00029     v.setVeg(seged.veg);
00030     v.setKocsidb(std::atoi(seged.kocsidb));
00031     v.setIndulas(seged.indulas);
00032     v.setErkezes(seged.erkezes);
00033
00034     // feltöltöm a MAV "mgmt" class Vonat* tömbjét
00035     this->add(v);
00036   }
00037   file.close();
00038 } // END OF BEOLVAS
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.7.3.5 beolvas() [2/2]

```
void Mav::beolvas (
            const char * test )
```

Mav beolvas overwrite Csak a teszt miatt kell, meg kell tudnom neki adni egy teszt vonatok.txt-t

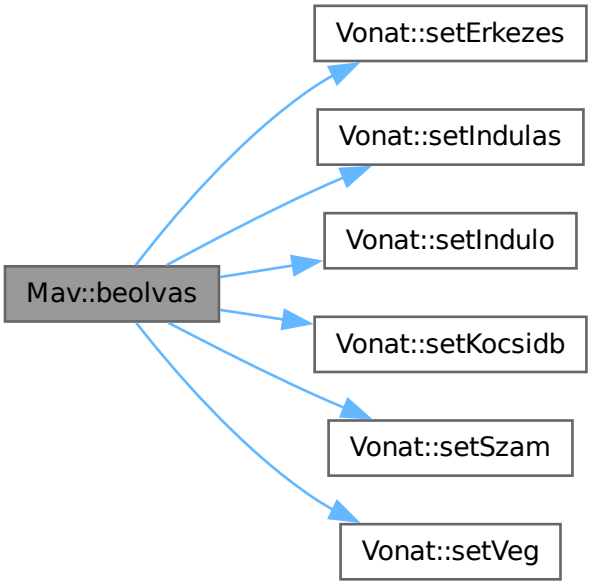
Parameters

file	file path (const char*)

Definition at line 46 of file mav.cpp.

```
00046                                {
00047   // Inicalizálás
00048   std::ifstream file(test);
00049   Vonat v;
00050   Seged seged;
00051
00052   // Kiürítem a vonatok kollekciót, hogy ne legyen ráolvasás
00053   delete[] vonatok;
00054   vonatok = nullptr;
00055   si = 0;
00056
00057   while (file » seged.szam » seged.indulo » seged.veg » seged.kocsidb » seged.indulas » seged.erkezes)
      {
00058     v.setSzam(std::atoi(seged.szam));
00059     v.setIndulo(seged.indulo);
00060     v.setVeg(seged.veg);
00061     v.setKocsidb(std::atoi(seged.kocsidb));
00062     v.setIndulas(seged.indulas);
00063     v.setErkezes(seged.erkezes);
00064
00065     // feltöltöm a MAV "mgmt" class Vonat* tömbjét
00066     this->add(v);
00067   }
00068   file.close();
00069 } // END OF BEOLVAS
```

Here is the call graph for this function:



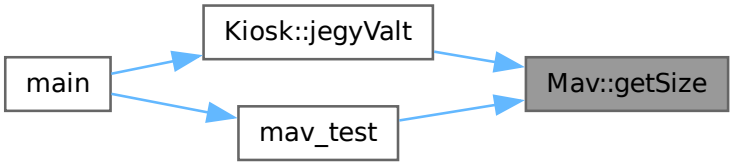
4.7.3.6 getSize()

```
size_t Mav::getSize ( )  [inline]
```

Definition at line 57 of file mav.h.

```
00057 { return static_cast<int>(si); }
```

Here is the caller graph for this function:



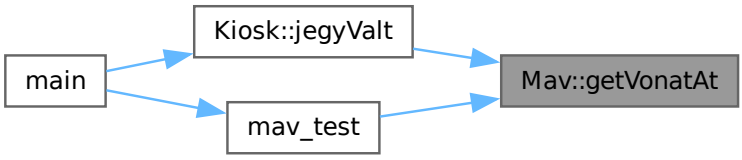
4.7.3.7 getVonatAt()

```
Vonat & Mav::getVonatAt (
            int i )  [inline]
```

Definition at line 56 of file mav.h.

```
00056 { return vonatok[i-1]; }
```

Here is the caller graph for this function:



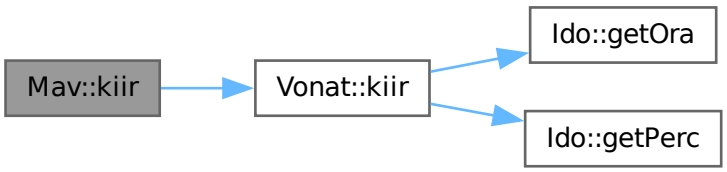
4.7.3.8 kiir()

```
void Mav::kiir ( )  [inline]
```

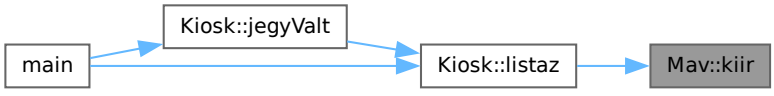
Definition at line 41 of file mav.h.

```
00041                {
00042     for (size_t i = 0; i < si; ++i) {
00043       std::cout « "--- " « i+1 « ".vonat ---" « std::endl;
00044         vonatok[i].kiir();
00045         std::cout « std::endl;
00046     }
00047   } // end of kiir
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.7.3.9 kiirAt()

```
void Mav::kiirAt (
            int i )  [inline]
```

Definition at line 50 of file mav.h.

```
00050                       {
00051     // HIBAEKEZELÉS
00052     std::cout « i+1 «".vonat" « std::endl;
00053     vonatok[i].kiir();
00054   } // end of kiirAt
```

Here is the call graph for this function:

4.7.3.10 operator[]()

```
Vonat & Mav::operator[] (
            int idx )  [inline]
```

op[] overload

Definition at line 92 of file mav.h.

```
00092 { return vonatok[idx]; }
```

4.7.4 Field Documentation

4.7.4.1 si

```
size_t Mav::si  [private]
```

Definition at line 18 of file mav.h.

4.7.4.2 vonatok

```
Vonat* Mav::vonatok  [private]
```

Definition at line 17 of file mav.h.

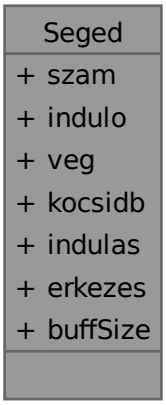
The documentation for this class was generated from the following files:

- mav.h
- mav.cpp

4.8 Seged Struct Reference

`#include <seged.h>`

Collaboration diagram for Seged:



Data Fields

- char szam [buffSize]
- char indulo [buffSize]
- char veg [buffSize]
- char kocsidb [buffSize]
- char indulas [buffSize]
- char erkezes [buffSize]

Static Public Attributes

- static const int buffSize = 25

4.8.1 Detailed Description

Magyarország leghosszabb településneve 15 karakter 25 karakterbe bele kell férnia Egyszerű segédstruktúra a beolvasáshoz segédkezett mind midle man

Definition at line 21 of file seged.h.

4.8.2 Field Documentation

4.8.2.1 buffSize

```
const int Seged::buffSize = 25  [static]
```

Definition at line 22 of file seged.h.

4.8.2.2 erkezes

```
char Seged::erkezes[buffSize]
```

Definition at line 28 of file seged.h.

4.8.2.3 indulas

```
char Seged::indulas[buffSize]
```

Definition at line 27 of file seged.h.

4.8.2.4 indulo

```
char Seged::indulo[buffSize]
```

Definition at line 24 of file seged.h.

4.8.2.5 kocsidb

```
char Seged::kocsidb[buffSize]
```

Definition at line 26 of file seged.h.

4.8.2.6 szam

```
char Seged::szam[buffSize]
```

Definition at line 23 of file seged.h.

4.8.2.7 veg

```
char Seged::veg[buffSize]
```

Definition at line 25 of file seged.h.

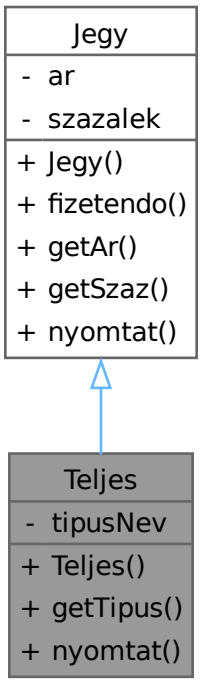
The documentation for this struct was generated from the following file:

- seged.h

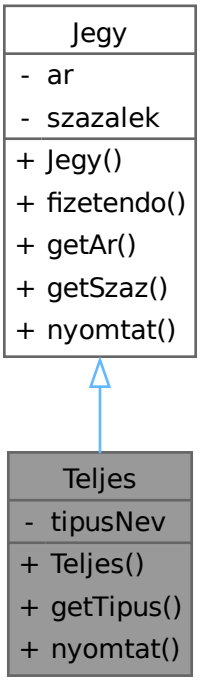
4.9 Teljes Class Reference

`#include <jegy.h>`

Inheritance diagram for Teljes:



Collaboration diagram for Teljes:



Public Member Functions

- Teljes (int ar=1200, const double szazalek=1.0)
- std::string getTipus ()
- void nyomtat (Vonat &v)

Public Member Functions inherited from Jegy

- Jegy (int ar=1200, double szazalek=0.7)

  Kedvezmény százaléka, pontosabban mennyi az amit fizet százalék.
- virtual int fizetendo ()

  Fizetendő összeg, ár és a kedvezmény szorzata.
- int getAr ()

  Getterek.
- double getSzaz ()

Private Attributes

- const std::string tipusNev = "Teljes"

4.9.1 Detailed Description

Definition at line 75 of file jegy.h.

4.9.2 Constructor & Destructor Documentation

4.9.2.1 Teljes()

```
Teljes::Teljes (
            int ar = 1200,
            const double szazalek = 1.0 )  [inline]
```

Konstruktor

Parameters

ar	megadja a jegy árát
szazalek	a kedvezmény értékét adja meg Meghívja a anya-class konstruktorát.

Definition at line 86 of file jegy.h.

```
00086 : Jegy(ar, szazalek) {}
```

4.9.3 Member Function Documentation

4.9.3.1 getTipus()

```
std::string Teljes::getTipus ( )  [inline]
```

Definition at line 88 of file jegy.h.

```
00088 { return tipusNev; }
```

Here is the caller graph for this function:

4.9.3.2 nyomtat()

```
void Teljes::nyomtat (
            Vonat & v )  [inline], [virtual]
```

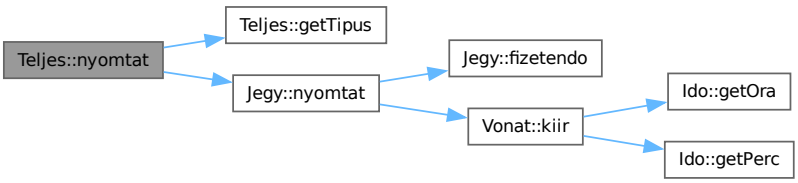
megörökölt nyomtat fv overwrite-ja meghívja az "eredei", főosztály nyomtatját

Reimplemented from Jegy.

Definition at line 92 of file jegy.h.

```
00092                           {
00093     std::cout « "### Jegy adatai ###\n";
00094     std::cout « "Jegy típusa: " « getTipus() « std::endl;
00095     Jegy::nyomtat(v);
00096   }
```

Here is the call graph for this function:



Here is the caller graph for this function:

4.9.4 Field Documentation

4.9.4.1 tipusNev

```
const std::string Teljes::tipusNev = "Teljes"  [private]
```

Definition at line 76 of file jegy.h.

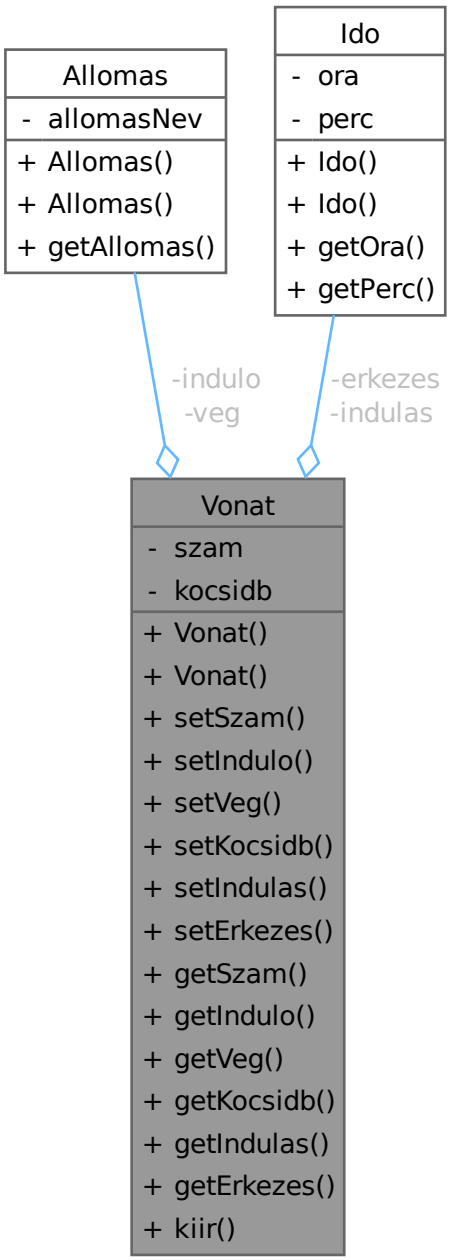
The documentation for this class was generated from the following file:

- jegy.h

4.10 Vonat Class Reference

```
#include <vonat.h>
```

Collaboration diagram for Vonat:



Public Member Functions

- Vonat ()

  Érkezési idp.
- Vonat (int vszam, Allomas indulop, Allomas vegp, int kocsidb, Ido indulasp, Ido erkezesp)

  Paraméteres Konstruktor.
- void setSzam (int szam)

Setterek.

- void setIndulo (const char *indulo)
- void setVeg (const char *veg)
- void setKocsidb (int kocsidb)
- void setIndulas (const char *ido)
- void setErkezes (const char *ido)
- int getSzam () const

  Getterek.
- Allomas getIndulo () const
- Allomas getVeg () const
- int getKocsidb () const
- Ido getIndulas () const
- Ido getErkezes () const
- void kiir () const

Private Attributes

- int szam
- Allomas indulo

  Vonatszám.
- Allomas veg

  Kiinduló állomas.
- int kocsidb

  Végállomas.
- Ido indulas

  Kocsik darabszáma, basically semmit sem csinál.
- Ido erkezes

  Indulási idő

4.10.1 Detailed Description

Definition at line 15 of file vonat.h.

4.10.2 Constructor & Destructor Documentation

4.10.2.1 Vonat() [1/2]

```
Vonat::Vonat ( )  [inline]
```

Érkezési idp.

Paraméter nélküli Konstruktor

Definition at line 26 of file vonat.h.

```
00026 : szam(0), indulo(""), veg(""), kocsidb(0), indulas(0,0), erkezes(0,0) {}
```

4.10.2.2 Vonat() [2/2]

```
Vonat::Vonat (
            int vszam,
            Allomas indulop,
            Allomas vegp,
            int kocsidb,
            Ido indulasp,
            Ido erkezesp )  [inline]
```

Paraméteres Konstruktor.

Definition at line 29 of file vonat.h.

```
00030         : szam(vszam), indulo(indulop), veg(vegp), kocsidb(kocsidb), indulas(indulasp),
      erkezes(erkezesp) {}
```

4.10.3 Member Function Documentation

4.10.3.1 getErkezes()

```
Ido Vonat::getErkezes ( ) const  [inline]
```

Definition at line 46 of file vonat.h.

```
00046 { return erkezes; }
```

4.10.3.2 getIndulas()

```
Ido Vonat::getIndulas ( ) const  [inline]
```

Definition at line 45 of file vonat.h.

```
00045 { return indulas; }
```

4.10.3.3 getIndulo()

```
Allomas Vonat::getIndulo ( ) const  [inline]
```

Definition at line 42 of file vonat.h.

```
00042 { return indulo; }
```

Here is the caller graph for this function:

4.10.3.4 getKocsidb()

```
int Vonat::getKocsidb ( ) const  [inline]
```

Definition at line 44 of file vonat.h.

```
00044 { return kocsidb; }
```

Here is the caller graph for this function:

4.10.3.5 getSzam()

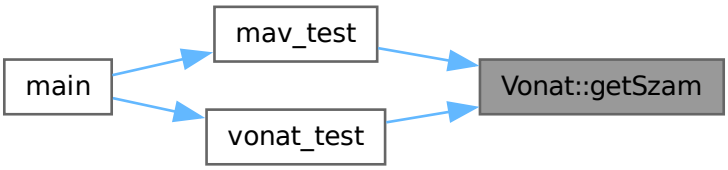
```
int Vonat::getSzam ( ) const  [inline]
```

Getterek.

Definition at line 41 of file vonat.h.

```
00041 { return szam; }
```

Here is the caller graph for this function:



4.10.3.6 getVeg()

```
Allomas Vonat::getVeg ( ) const  [inline]
```

Definition at line 43 of file vonat.h.

```
00043 { return veg; }
```

4.10.3.7 kiir()

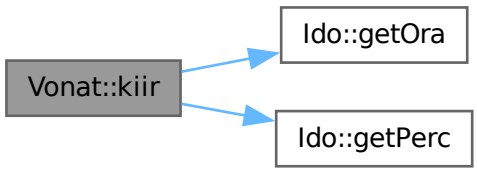
```
void Vonat::kiir ( ) const  [inline]
```

Kiírja a vonat adatait Főképp a jegyváltáskor van meghívva + listázás.

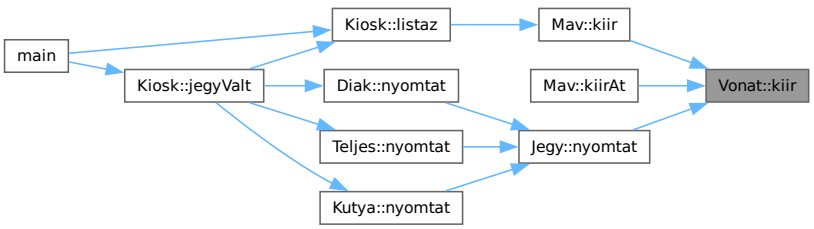
Definition at line 52 of file vonat.h.

```
00052                 {
00053     std::cout « "Vonat szama: " « szam « std::endl;
00054     std::cout « "Indulasi allomas: " « indulo « std::endl;
00055     std::cout « "Erkezesi allomas: " « veg « std::endl;
00056     std::cout « "Kocsi darabszam: " « kocsidb « std::endl;
00057
00058     std::cout « "Indulas idopontja: ";
00059     std::cout « std::setw(2) « std::setfill('0') «indulas.getOra();
00060     std::cout « ":";
00061     std::cout « std::setw(2) « std::setfill('0') « indulas.getPerc() « std::endl; // Nagyon ronda,
      tudom...
00062
00063     std::cout « "Érkezés idopontja: ";
00064     std::cout « std::setw(2) « std::setfill('0') « erkezes.getOra();
00065     std::cout « ":";
00066     std::cout « std::setw(2) « std::setfill('0') « erkezes.getPerc() « std::endl; // Nagyon ronda,
      tudom...
00067
00068   }
```

Here is the call graph for this function:



Here is the caller graph for this function:

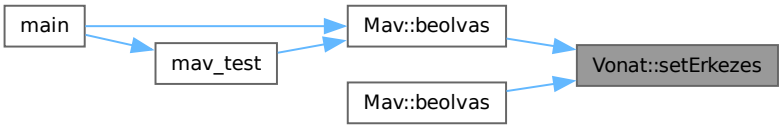


4.10.3.8 setErkezes()

```
void Vonat::setErkezes (
            const char * ido )  [inline]
```

Definition at line 38 of file vonat.h.
```
00038 { this->erkezes = Ido(ido); }
```

Here is the caller graph for this function:

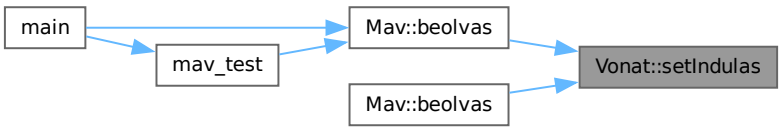


4.10.3.9 setIndulas()

```
void Vonat::setIndulas (
            const char * ido )  [inline]
```

Definition at line 37 of file vonat.h.
```
00037 { this->indulas = Ido(ido); }
```

Here is the caller graph for this function:

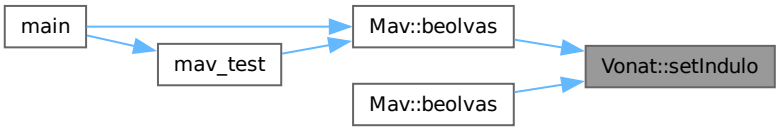


4.10.3.10 setIndulo()

```
void Vonat::setIndulo (
            const char * indulo )  [inline]
```

Definition at line 34 of file vonat.h.
```
00034 { this-> indulo = Allomas(indulo); }
```

Here is the caller graph for this function:



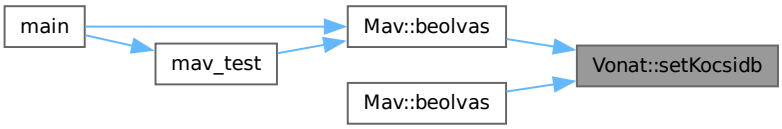
4.10.3.11 setKocsidb()

```
void Vonat::setKocsidb (
            int kocsidb )  [inline]
```

Definition at line 36 of file vonat.h.

```
00036 { this->kocsidb = kocsidb; }
```

Here is the caller graph for this function:



4.10.3.12 setSzam()

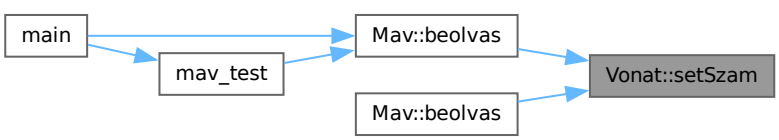
```
void Vonat::setSzam (
            int szam )  [inline]
```

Setterek.

Definition at line 33 of file vonat.h.

```
00033 { this->szam = szam; }
```

Here is the caller graph for this function:



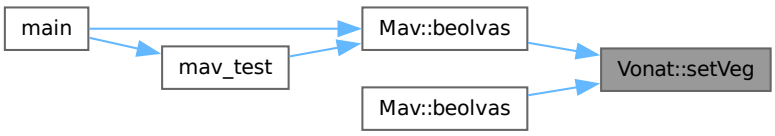
4.10.3.13 setVeg()

```
void Vonat::setVeg (
            const char * veg )  [inline]
```

Definition at line 35 of file vonat.h.

```
00035 { this->veg = Allomas(veg); }
```

Here is the caller graph for this function:



4.10.4 Field Documentation

4.10.4.1 erkezes

```
Ido Vonat::erkezes  [private]
```

Indulási idő

Definition at line 22 of file vonat.h.

4.10.4.2 indulas

```
Ido Vonat::indulas  [private]
```

Kocsik darabszáma, basically semmit sem csinál.

Definition at line 21 of file vonat.h.

4.10.4.3 indulo

```
Allomas Vonat::indulo  [private]
```

Vonatszám.

Definition at line 18 of file vonat.h.

4.10.4.4 kocsidb

```
int Vonat::kocsidb  [private]
```

Végállomas.

Definition at line 20 of file vonat.h.

4.10.4.5 szam

```
int Vonat::szam  [private]
```

Definition at line 17 of file vonat.h.

4.10.4.6 veg

```
Allomas Vonat::veg  [private]
```

Kiinduló állomas.

Definition at line 19 of file vonat.h.

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

- vonat.h

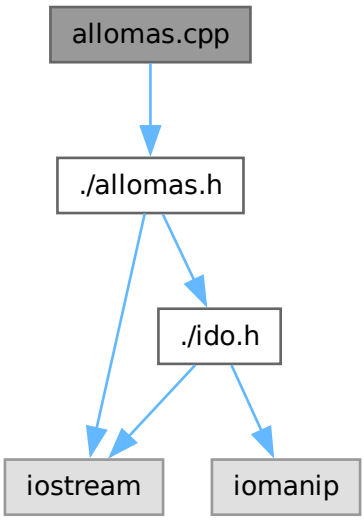
Chapter 5

File Documentation

5.1 allomas.cpp File Reference

```
#include "./allomas.h"
```

Include dependency graph for allomas.cpp:



Functions

- std::ostream & operator<< (std::ostream &os, const Allomas &allomas)

  Allomas op<< overload.

5.1.1 Function Documentation

5.1.1.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & os,
            const Allomas & allomas )
```

Allomas op<< overload.

Definition at line 4 of file allomas.cpp.

```
00004 { return os « allomas.getAllomas(); }
```

Here is the call graph for this function:

5.2 allomas.cpp

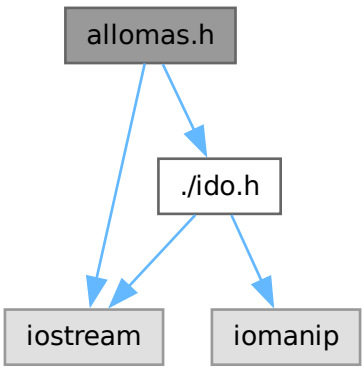
Go to the documentation of this file.

```
00001 #include "./allomas.h"
00002
00003 // Másképp nem ette meg a compiler.
00004 std::ostream& operator«(std::ostream& os, const Allomas& allomas) { return os « allomas.getAllomas();
      }
```

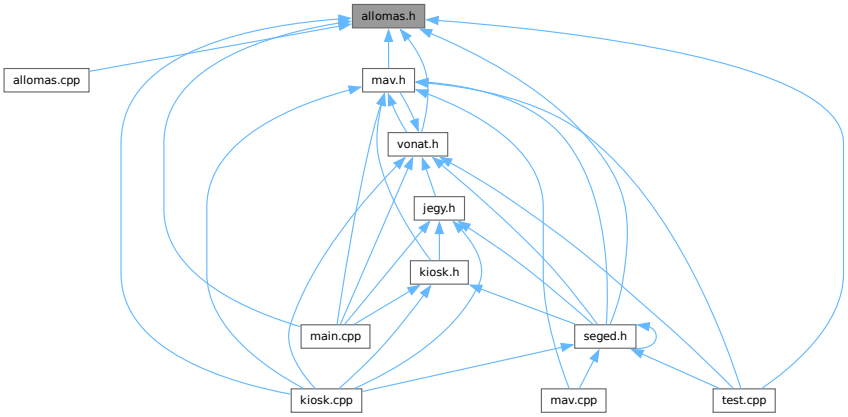
5.3 allomas.h File Reference

```
#include <iostream>
#include "./ido.h"
```

Include dependency graph for allomas.h:



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



Data Structures

- class Allomas

Functions

- std::ostream & operator<< (std::ostream &os, const Allomas &allomas)

  Allomas op<< overload.

5.3.1 Function Documentation

5.3.1.1 operator<<()

```
std::ostream & operator<< (
            std::ostream & os,
            const Allomas & allomas )
```

Allomas op<< overload.

Definition at line 4 of file allomas.cpp.

```
00004 { return os « allomas.getAllomas(); }
```

Here is the call graph for this function:

5.4 allomas.h

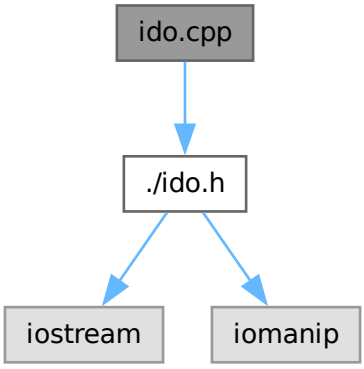
Go to the documentation of this file.

```
00001 #ifndef ALLOMASH_H
00002 #define ALLOMASH_H
00003
00004 /* file ALLOMASH_H */
00005
00006 // STD::STRING-GEL MEGY A CUCCOKS
00007
00008 #include <iostream>
00009 #include "./ido.h"
00010
00011 /** STRINGGEL ÚJRAÍRVA */
00012 class Allomas{
00013 private:
00014   std::string allomasNev; /// Allomas nevét tartalmazo string
00015 public:
00016   /**
00017   * Allomas konstrukor
00018   * @param allomas itt egy char* egyértelműen.
00019   * a másik verzióban const, hogy fel lehessen közvetlen tölteni
00020   * értsd: "Álloms_neve" mint paraméter.
00021   */
00022   Allomas(const char* allomas) : allomasNev(allomas) {}
00023   Allomas(char* allomas) : allomasNev(allomas) {}
00024   // Allomas(const std::string& allomas) : allomasNev(allomas) {}
00025
00026   // Getterek
00027   std::string getAllomas() const { return allomasNev; }
00028
00029 }; // End of ALLOMAS
00030 /// Allomas op« overload
00031 std::ostream& operator«(std::ostream& os, const Allomas& allomas);
00032
00033 #endif // !ALLOMASH_H
```

5.5 ido.cpp File Reference

`#include "./ido.h"`

Include dependency graph for ido.cpp:



Functions

- std::ostream & operator<< (std::ostream &os, const Ido &ido)

  Ido class op<<() overwrireja, a fájlba beíráskor hívódik meg. (addTrain)

5.5.1 Function Documentation

5.5.1.1 operator<<()

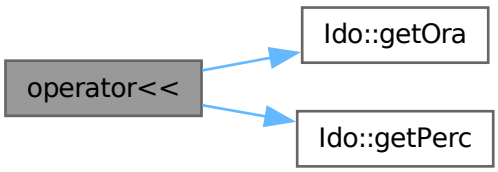
```
std::ostream & operator<< (
            std::ostream & os,
            const Ido & ido )
```

Ido class op<<() overwrrireja, a fájlba beíráskor hívódik meg. (addTrain)

Definition at line 3 of file ido.cpp.

```
00003                                                   {
00004   os « std::setw(2) « std::setfill('0') « ido.getOra();
00005   os « std::setw(2) « std::setfill('0') « ido.getPerc();
00006   return os;
00007 } // operator«()
```

Here is the call graph for this function:



5.6 ido.cpp

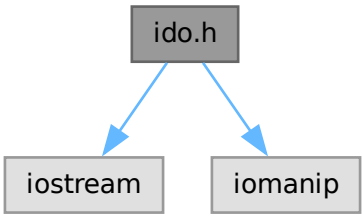
Go to the documentation of this file.

```
00001 #include "./ido.h"
00002
00003 std::ostream& operator«(std::ostream& os, const Ido& ido){
00004   os « std::setw(2) « std::setfill('0') « ido.getOra();
00005   os « std::setw(2) « std::setfill('0') « ido.getPerc();
00006   return os;
00007 } // operator«()
00008
```

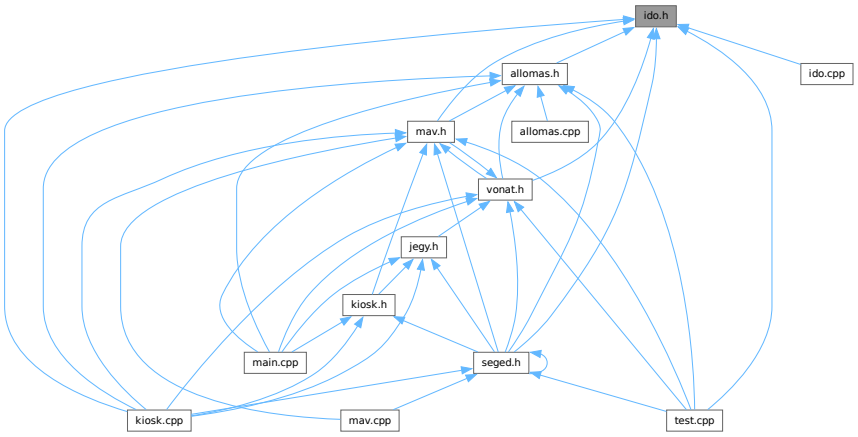
5.7 ido.h File Reference

```
#include <iostream>
#include <iomanip>
```

Include dependency graph for ido.h:



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



Data Structures

- class Ido

Functions

- std::ostream & operator<< (std::ostream &os, const Ido &ido)

  Ido class op<<() overwrireja, a fájlba beíráskor hívódik meg. (addTrain)

5.7.1 Function Documentation

5.7.1.1 operator<<()

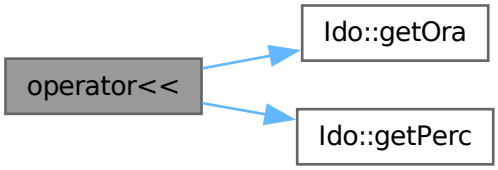
```
std::ostream & operator<< (
            std::ostream & os,
            const Ido & ido )
```

Ido class op<<() overwrireja, a fájlba beíráskor hívódik meg. (addTrain)

Definition at line 3 of file ido.cpp.

```
00003                                                   {
00004   os « std::setw(2) « std::setfill('0') « ido.getOra();
00005   os « std::setw(2) « std::setfill('0') « ido.getPerc();
00006   return os;
00007 } // operator«()
```

Here is the call graph for this function:



5.8 ido.h

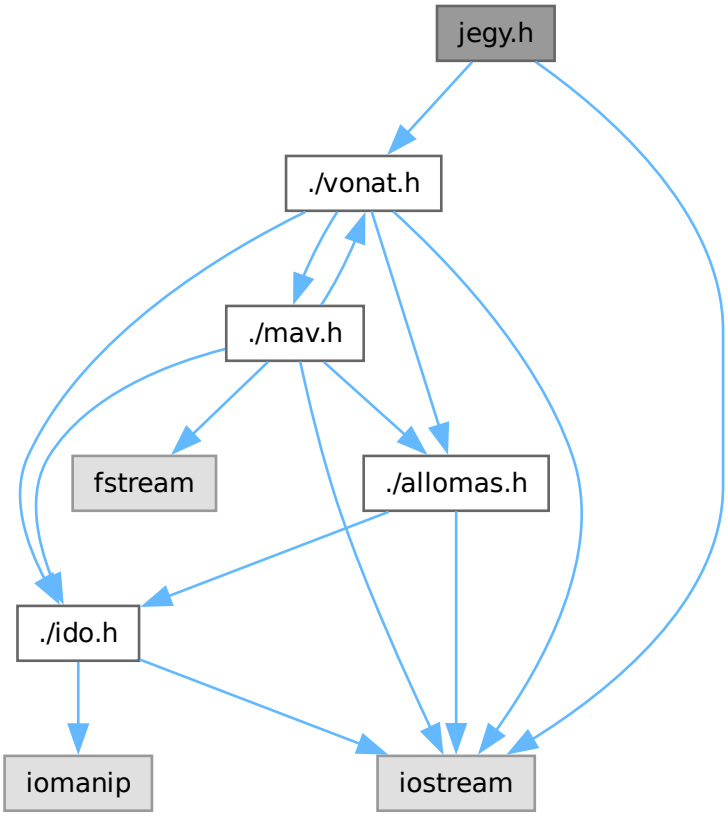
Go to the documentation of this file.

```
00001 #ifndef IDO_H
00002 #define IDO_H
00003
00004 /*
00005 * FILE IDO_H
00006 */
00007
00008 #include <iostream>
00009 #include <iomanip>
00010
00011
00012 class Ido{
00013   int ora;
00014   int perc;
00015 public:
00016   /**
00017   * Konstruktor
00018   * @param ido mivel beolvasáskor char* típusokba olvasok bele
00019   * char*-ot kap paraméterként és azt kezeli megfelelően konvertálva.
00020   */
00021   Ido(const char* ido) {
00022     ora = (ido[0] - '0') * 10 + (ido[1] - '0');
00023     perc = (ido[2] - '0') * 10 + (ido[3] - '0');
00024   }
00025
00026   /**
00027   * Ido Konstruktor
00028   * @param ora (int)
00029   * @param perc (int)
00030   * Főképp tesztekhez volt használva.
00031   */
00032   Ido(int ora, int perc) : ora(ora), perc(perc) {}
00033
00034   // Getterek
00035   int getOra() const { return ora; };
00036   int getPerc() const {return perc; };
00037
00038 }; // END OF IDO
00039
00040 /// Ido class op«() overwrireja, a fájlba beíráskor hívódik meg. (addTrain)
00041 std::ostream& operator«(std::ostream& os, const Ido& ido);
00042
00043
00044
00045 #endif // !IDO_H
```

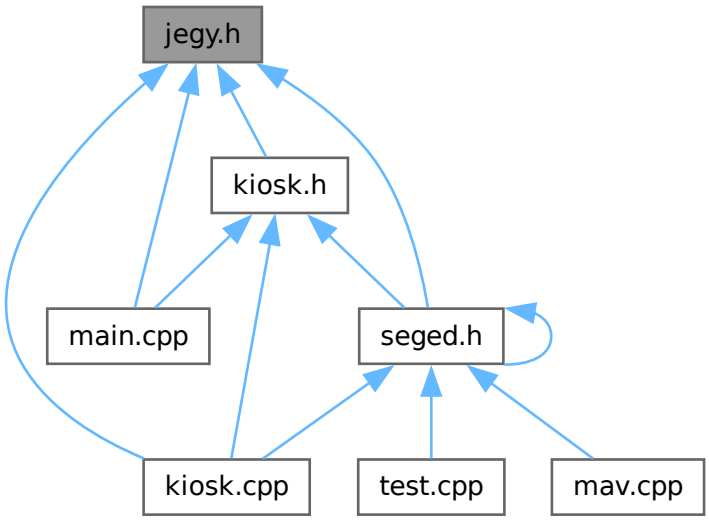
5.9 jegy.h File Reference

```
#include <iostream>
#include "./vonat.h"
```

Include dependency graph for jegy.h:



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



Data Structures

- class Jegy
- class Diak
- class Teljes
- class Kutya

5.10 jegy.h

Go to the documentation of this file.

```
00001 #ifndef JEGY_H
00002 #define JEGY_H
00003
00004 /**
00005 * FILE JEGY_H
00006 */
00007
00008 #include <iostream>
00009 #include "./vonat.h"
00010
00011 class Jegy {
00012   int ar;           /// A jegy ára
00013   double szazalek;  /// Kedvezmény százaléka, pontosabban mennyi az amit fizet százalék.
00014
00015 public:
00016   /**
00017    * Anya osztály Konstruktora
00018    * @param ar megadja a jegy árát
00019    * @param szazalek a kedvezmény értékét adja meg
00020    */
00021   Jegy(int ar = 1200, double szazalek = 0.7) : ar(ar), szazalek(szazalek)  {}
00022
00023   /// Fizetendő összeg, ár és a kedvezmény szorzata
00024   virtual int fizetendo(){ return ar * szazalek; }
00025
00026   /// Getterek
00027   int getAr() { return ar; }
```

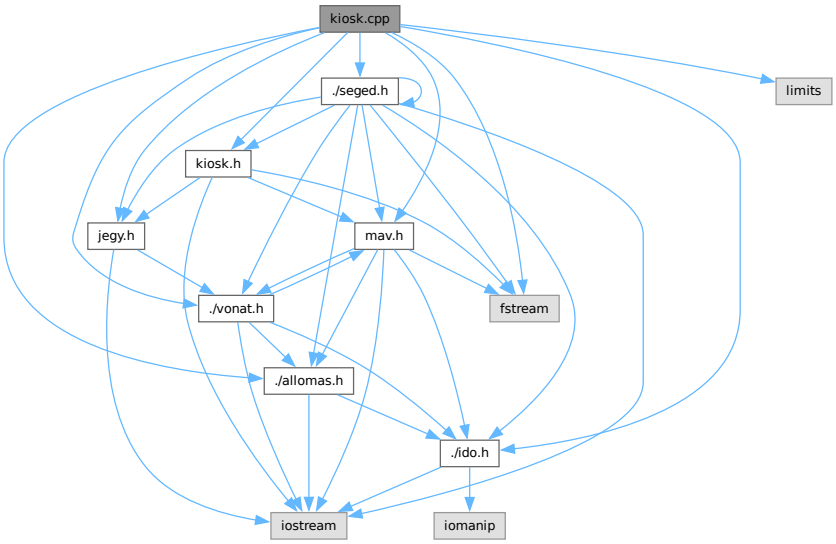
```
00028   double getSzaz() { return szazalek; }
00029
00030   /**
00031   * @param Vonat& megadott vonatra vonatkozik
00032   * Kiírja a jegy adatinak egy felét. Azt a felét ami minden leszármazottál
00033   * azonos. A felső pár so.
00034   * virtual mivel a leszármazottak megöröklik és felülírják.
00035   */
00036   virtual void nyomtat(Vonat& v) {
00037     std::cout « "Fizetendő: " « fizetendo() « "JMF\n";
00038     std::cout « "### Vonatod adatai ###" « std::endl;
00039     v.kiir();
00040     std::cout « std::endl;
00041   }
00042   // virtual void abstract() = 0 {}
00043
00044 }; // END OF JEGY
00045
00046
00047
00048 class Diak : public Jegy {
00049   const std::string tipusNev = "Diák";
00050   // const double szazalek = 0.7;
00051
00052 public:
00053    /**
00054    * Konstruktor
00055    * @param ar megadja a jegy árát
00056    * @param szazalek a kedvezmény értékét adja meg
00057    * Meghívja a anya-class konstruktorát.
00058   */
00059   Diak(int ar = 1200, const double szazalek = 0.7) : Jegy(ar, szazalek) {}
00060
00061   /// Getterek
00062   std::string getTipus() { return tipusNev; }
00063
00064   /// megörökölt nyomtat fv overwrite-ja
00065   /// meghívja az "eredei", főosztály nyomtatját
00066   void nyomtat(Vonat &v) {
00067     std::cout « "### Jegy adatai ###\n";
00068     std::cout « "Jegy típusa: " « getTipus() « std::endl;
00069     Jegy::nyomtat(v);
00070   }
00071
00072 }; // end of diak
00073
00074
00075 class Teljes : public Jegy {
00076   const std::string tipusNev = "Teljes";
00077   // const double szazalek = 1.0;
00078
00079 public:
00080   /**
00081    * Konstruktor
00082    * @param ar megadja a jegy árát
00083    * @param szazalek a kedvezmény értékét adja meg
00084    * Meghívja a anya-class konstruktorát.
00085   */
00086   Teljes(int ar = 1200, const double szazalek = 1.0) : Jegy(ar, szazalek) {}
00087
00088   std::string getTipus() { return tipusNev; }
00089
00090   /// megörökölt nyomtat fv overwrite-ja
00091   /// meghívja az "eredei", főosztály nyomtatját
00092   void nyomtat(Vonat &v) {
00093     std::cout « "### Jegy adatai ###\n";
00094     std::cout « "Jegy típusa: " « getTipus() « std::endl;
00095     Jegy::nyomtat(v);
00096   }
00097
00098 }; // end of Teljes
00099
00100
00101 class Kutya : public Jegy {
00102   const std::string tipusNev = "Kutya";
00103   const std::string kov = "Szájkosár";
00104
00105 public:
00106   /**
00107    * Konstruktor
00108    * @param ar megadja a jegy árát
00109    * @param szazalek a kedvezmény értékét adja meg
00110    * Meghívja a anya-class konstruktorát.
00111   */
00112   Kutya(int ar = 1200, const double szazalek = 0.2) : Jegy(ar, szazalek) {}
00113
00114   // Getterek
```

```
00115   std::string getTipus() { return tipusNev; }
00116   std::string getKov() { return kov; }
00117
00118   /// megörökölt nyomtat fv overwrite-ja
00119   /// meghívja az "eredei", főosztály nyomtatját
00120   void nyomtat(Vonat &v) {
00121     std::cout « "### Jegy adatai ###\n";
00122     std::cout « "Jegy típusa: " « getTipus() « std::endl;
00123     std::cout « "Követelmény: " « getKov() « std::endl;
00124     Jegy::nyomtat(v);
00125   }
00126
00127 }; // end of Kutya
00128
00129
00130 #endif // !JEGY_H
```

5.11 kiosk.cpp File Reference

```
#include "kiosk.h"
#include "jegy.h"
#include "./seged.h"
#include "./mav.h"
#include "./vonat.h"
#include "allomas.h"
#include "ido.h"
#include <fstream>
#include <limits>
```

Include dependency graph for kiosk.cpp:



5.12 kiosk.cpp

Go to the documentation of this file.

```
00001
00002 #include "kiosk.h"
00003 #include "jegy.h"
00004 #include "./seged.h"
00005
```

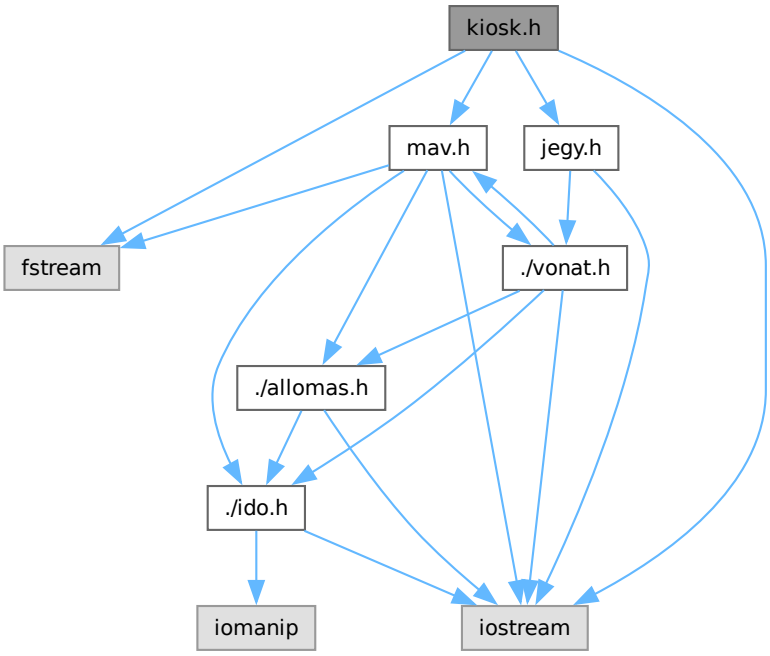
```
00006 #include "./mav.h"
00007 #include "./vonat.h"
00008 #include "allomas.h"
00009 #include "ido.h"
00010
00011 #include <fstream>
00012 #include <limits>
00013
00014 /// Hibakezelés. Érvényes input esetén @return true, különben @return false
00015 bool Kiosk::inputCheck(){
00016   // Kb innen lopva
      https://stackoverflow.com/questions/12721911/c-how-to-verify-if-the-data-input-is-of-the-correct-datatype
00017   if (std::cin.fail()) {
00018         std::cout « "Nem sikerült érvényes inputot megadni... PRÓBÁLD ÚJRA" « std::endl;
00019         std::cin.clear();
00020         std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
00021         return false;
00022   }
00023   else return true;
00024 }
00025
00026 /// Inicializálja a menüt, ergo kiírja a lehetőségeket
00027 void Kiosk::init() {
00028   std::cout « "### Dönts ###" « std::endl;
00029   std::cout « "1. Vonat hozzáadása "     « std::endl;
00030   std::cout « "2. Vonatok listázása "    « std::endl;
00031   std::cout « "3. Jegy nyomtatása"       « std::endl;
00032   std::cout « "4. Kilépés"               « std::endl;
00033   std::cout « "5. Teszt"                 « std::endl;
00034 }
00035
00036 /// Felhasználótól bemenetet kér.
00037 /// @return menuItem fent említett enum class elemt ad vissza
00038 // User input az input streamről enum-má konvertálva
00039 menuItem Kiosk::userInput() {
00040    int valasz;
00041    std::cout « "Választott lehetőség: ";
00042    std::cin » valasz;
00043    return static_cast<menuItem>(valasz);
00044 }
00045
00046 /// Liszázza az összes vonatot
00047 void Kiosk::listaz(Mav& mav){
00048   mav.kiir();
00049 }
00050
00051 /**
00052 * User hozzáadhat vonatot a file-hoz
00053 * @param mav a mav osztályra refereancia, ezt töltöm fel (Mav)
00054 * @param currpos a kurzor éppenleges helye, ahova írni kell, hogy
00055 * ne legyen felülírás (streampos)
00056 */
00057 void Kiosk::vonatHozza(Mav& mav, std::streampos currPos){
00058   Seged s;
00059
00060   std::cin » s.szam;
00061   // if(*(s.szam) == '\n') std::cout « "FASZ";
00062   std::cin » s.indulo;
00063   std::cin » s.veg;
00064   std::cin » s.kocsidb;
00065   std::cin » s.indulas;
00066   std::cin » s.erkezes;
00067
00068   mav.addTrain(currPos, atoi(s.szam), Allomas(s.indulo), Allomas(s.veg),
00069                  atoi(s.kocsidb), Ido(s.indulas), Ido(s.erkezes));
00070 } // End of vonatHozza
00071
00072 /**
00073 * Jegy vásárlás, nyomtatás
00074 * @param mav mav refereancia, ez az osztály tartalmazza a vonatok* tömböt
00075 */
00076 void Kiosk::jegyValt(Mav& mav){
00077   int idx, buf;
00078   listaz(mav);
00079   std::cout « "Valasz vonatot: ";
00080   std::cin » idx;
00081   // Hibakezelés
00082   if(size_t(idx) > mav.getSize()){ std::cout « "Túlindexelés\n"; return; }
00083
00084   if(!(Kiosk::inputCheck())) return; // Érvényes inputot szűrök.
00085
00086   std::cout « std::endl; // szép kiírás miatt.
00087   // end if Hibakezelés
00088
00089
00090   std::cout « "1.Teljes, 2.Diak, 3.Kutya\n ";
00091   std::cin » buf;
```

```
00092   // Hibakezelés
00093   if(buf > 3){ std::cout « "Túlindexelés"; return; }
00094
00095   if(!(Kiosk::inputCheck())) return;
00096
00097   std::cout « std::endl;
00098   // end of Hibakezelés
00099
00100   switch (buf) {
00101     case 1: {
00102       Teljes t;
00103       t.nyomtat(mav.getVonatAt(idx));
00104       break;
00105     }
00106
00107     case 2: {
00108       Diak d;
00109       d.nyomtat(mav.getVonatAt(idx));
00110       break;
00111     }
00112
00113     case 3: {
00114       Kutya k;
00115       k.nyomtat(mav.getVonatAt(idx));
00116       break;
00117     }
00118
00119   } // end of switch
00120
00121 } // end of jegyValt
00122
00123
00124
00125
00126
00127
00128
```

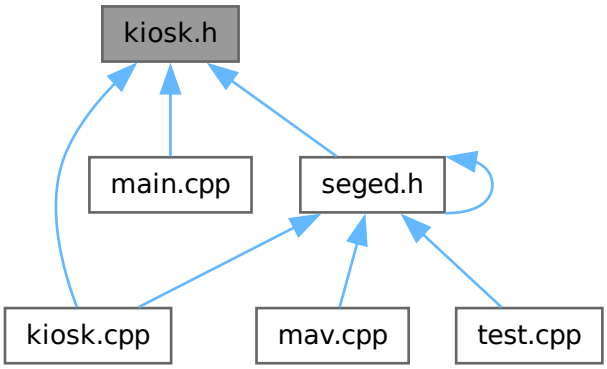
5.13 kiosk.h File Reference

```
#include <iostream>
#include "mav.h"
#include "jegy.h"
#include <fstream>
```

Include dependency graph for kiosk.h:



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



Data Structures

- class Kiosk

Enumerations

- enum menuItem { add , list , nyomtat , kilep }

5.13.1 Enumeration Type Documentation

5.13.1.1 menuItem

```
enum menuItem
```

Segéd "class" A menü kezelését könnyíti meg.

Enumerator

add	
list	
nyomtat	
kilep	

Definition at line 16 of file kiosk.h.

```
00016             {
00017   add,
00018   list,
00019   nyomtat,
00020   kilep
00021 };
```

5.14 kiosk.h

Go to the documentation of this file.

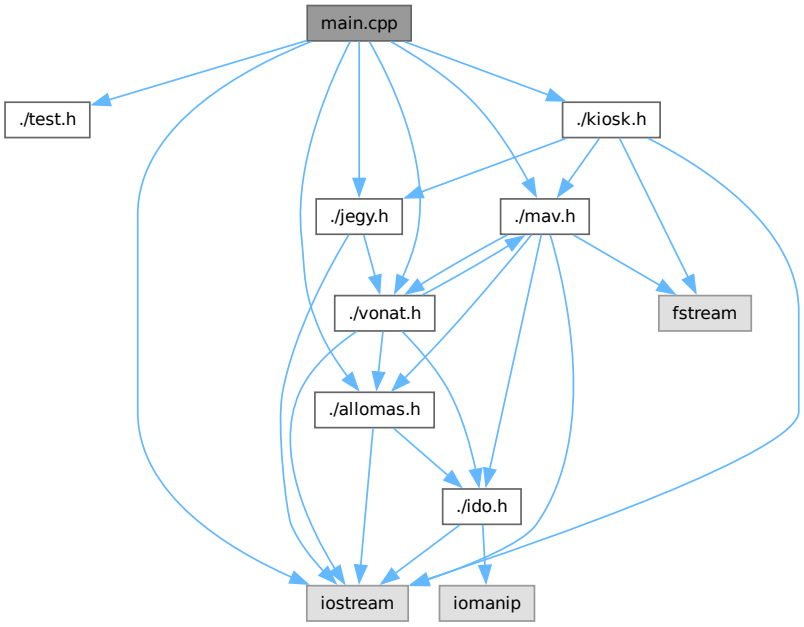
```
00001 #ifndef KIOSK_H
00002 #define KIOSK_H
00003
00004 /*
00005 * FILE KIOSK_H
00006 */
00007
00008 #include <iostream>
00009
00010 #include "mav.h"
00011 #include "jegy.h"
00012 #include <fstream>
00013
00014 /// Segéd "class"
00015 /// A menü kezelését könnyíti meg.
00016 enum menuItem{
00017   add,
00018   list,
00019   nyomtat,
00020   kilep
00021 };
00022
00023 class Kiosk {
00024 public:
00025
00026   /// Hibakezelés. Érvényes input esetén @return true, különben @return false
00027   bool inputCheck();
00028   /// Inicializálja a menüt, ergo kiírja a lehetőségeket
00029   void init();
00030   /// Felhasználótól bemenetet kér.
00031   /// @return menuItem fent említett enum class elemt ad vissza
00032   menuItem userInput();
00033
```

```
00034   /// Liszázza a vonatokat amiket beolvasott a fájlból
00035   void listaz(Mav& mav);
00036   /// Hozzá ad a user vonatokat a fájlhoz
00037   void vonatHozza(Mav& mav, std::streampos currPos);
00038   /// jegyet vált a kiválasztott vonatra
00039   void jegyValt(Mav& mav);
00040
00041 }; // end of Kiosk
00042
00043
00044
00045
00046
00047
00048 #endif // !KIOSK_H
```

5.15 main.cpp File Reference

```
#include "./test.h"
#include "./mav.h"
#include "./vonat.h"
#include "./allomas.h"
#include "./jegy.h"
#include "./kiosk.h"
#include <iostream>
```

Include dependency graph for main.cpp:



Functions

- int main ()

5.15.1 Function Documentation

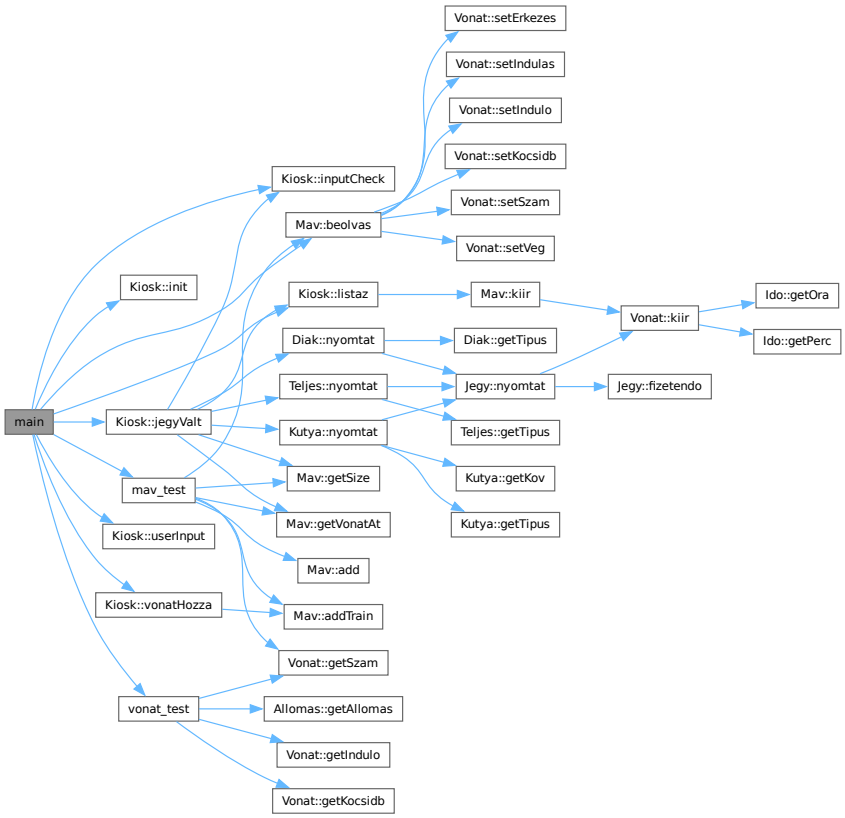
5.15.1.1 main()

```
int main ( )
```

Definition at line 11 of file main.cpp.

```
00011           {
00012   // Itt tárolon hol vagyok a file-ban
00013   // Mind a beolvasás mind pedig az íráshoz KELL!
00014   std::streampos currPos;
00015   Mav mav;
00016   mav.beolvas();
00017
00018   Kiosk k; // kiosk inicalizálása
00019   k.init(); // Kiírja a lehetőségeket
00020
00021   int choice = 0;
00022   while (choice != 4) {
00023     choice = k.userInput();
00024     switch (choice) {
00025
00026       // Vonat hozzáadása
00027       case 1: {
00028         std::cout « "Beolvas: " « std::endl;
00029         std::cout « "Szám Indulo Vég kocsidb Indulás erkezes" « std::endl;
00030         k.vonatHozza(mav, currPos);
00031         // system("clear"); // Tábla törlés, nem ette meg a JPORTA
00032         std::cout « "\n" « "Vonat hozzáadva\n\n";
00033         mav.beolvas();
00034         k.init(); // Kiírja a lehetőségeket
00035         break;
00036       }
00037
00038       // Vonatok listázása
00039       case 2: {
00040         std::cout « std::endl;
00041         mav.beolvas();
00042         k.listaz(mav);
00043         std::cout « std::endl;
00044         k.init(); // Kiírja a lehetőségeket
00045         break;
00046       }
00047
00048       // Jegy vásárlás
00049       case 3: {
00050         std::cout « std::endl;
00051         k.jegyValt(mav);
00052         std::cout « std::endl;
00053         k.init(); // Kiírja a lehetőségeket
00054         break;
00055       }
00056
00057       // Kilépés
00058       case 4: {
00059         std::cout « "Kilépés" « std::endl;
00060         break;
00061       }
00062
00063       // TEST case
00064       case 5:{
00065         std::cout « "Teszt futtatása" « std::endl;
00066         mav_test();
00067         vonat_test();
00068         k.init();
00069       }
00070
00071       default: {
00072         // std::cout « "Érvénytelen lehetőség" « std::endl;
00073         // Hibakezelés
00074         if(!(k.inputCheck())) { std::cout « std::endl; k.init(); }
00075       }
00076
00077     } // end of switch
00078   } // end of while
00079
00080
00081   return 0;
00082 } // end of main
```

Here is the call graph for this function:



5.16 main.cpp

Go to the documentation of this file.

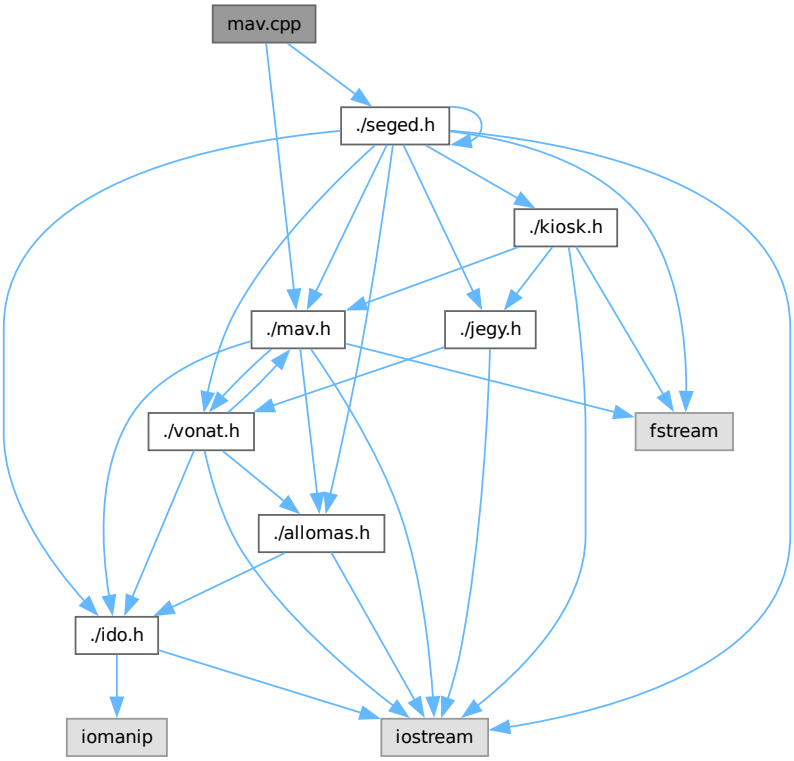
```
00001 #include "./test.h"
00002
00003 #include "./mav.h"
00004 #include "./vonat.h"
00005 #include "./allomas.h"
00006 #include "./jegy.h"
00007 #include "./kiosk.h"
00008
00009 #include <iostream>
00010
00011 int main(){
00012   // Itt tárolon hol vagyok a file-ban
00013   // Mind a beolvasás mind pedig az íráshoz KELL!
00014   std::streampos currPos;
00015   Mav mav;
00016   mav.beolvas();
00017
00018   Kiosk k; // kiosk inicalizálása
00019   k.init(); // Kiírja a lehetőségeket
00020
00021   int choice = 0;
00022   while (choice != 4) {
00023     choice = k.userInput();
00024     switch (choice) {
00025
00026       // Vonat hozzáadása
00027       case 1: {
00028         std::cout << "Beolvas: " << std::endl;
00029         std::cout << "Szám Indulo Vég kocsidb Indulás erkezes" << std::endl;
00030         k.vonatHozza(mav, currPos);
00031         // system("clear"); // Tábla törlés, nem ette meg a JPORTA
```

```
00032         std::cout « "\n" « "Vonat hozzáadva\n\n";
00033         mav.beolvas();
00034         k.init(); // Kiírja a lehetőségeket
00035         break;
00036       }
00037
00038       // Vonatok listázása
00039       case 2: {
00040         std::cout « std::endl;
00041         mav.beolvas();
00042         k.listaz(mav);
00043         std::cout « std::endl;
00044         k.init(); // Kiírja a lehetőségeket
00045         break;
00046       }
00047
00048       // Jegy vásárlás
00049       case 3: {
00050         std::cout « std::endl;
00051         k.jegyValt(mav);
00052         std::cout « std::endl;
00053         k.init(); // Kiírja a lehetőségeket
00054         break;
00055       }
00056
00057       // Kilépés
00058       case 4: {
00059         std::cout « "Kilépés" « std::endl;
00060         break;
00061       }
00062
00063       // TEST case
00064       case 5:{
00065         std::cout « "Teszt futtatása" « std::endl;
00066         mav_test();
00067         vonat_test();
00068         k.init();
00069       }
00070
00071       default: {
00072         // std::cout « "Érvénytelen lehetőség" « std::endl;
00073         // Hibakezelés
00074         if(!(k.inputCheck())) { std::cout « std::endl; k.init(); }
00075       }
00076
00077     } // end of switch
00078   } // end of while
00079
00080
00081   return 0;
00082 } // end of main
```

5.17 mav.cpp File Reference

```
#include "./mav.h"
#include "./seged.h"
```

Include dependency graph for mav.cpp:



5.18 mav.cpp

Go to the documentation of this file.

```
00001 #include "./mav.h"
00002 #include "./seged.h"
00003
00004
00005 /**
00006  * Beolvaásás a file-ból egy segéd strukturába, onnan pedig egy Vonat objektum feltöltés
00007  * @param Mav& mav class feltöltéséhez szükséges
00008  */
00009 void Mav::beolvas() {
00010   // Inicalizálás
00011   const char* vonatok_file = "./vonatok.txt";
00012   std::ifstream file(vonatok_file);
00013   Vonat v;
00014   Seged seged;
00015
00016   if (!file.is_open()) {
00017     std::cout « "Megynitással van baj " « vonatok_file « std::endl;
00018     return;
00019   }
00020
00021   // Kiürítem a vonatok kollekciót, hogy ne legyen ráolvasás
00022   delete[] vonatok;
00023   vonatok = nullptr;
00024   si = 0;
00025
00026   while (file » seged.szam » seged.indulo » seged.veg » seged.kocsidb » seged.indulas » seged.erkezes)
      {
00027     v.setSzam(std::atoi(seged.szam));
00028     v.setIndulo(seged.indulo);
00029     v.setVeg(seged.veg);
00030     v.setKocsidb(std::atoi(seged.kocsidb));
```

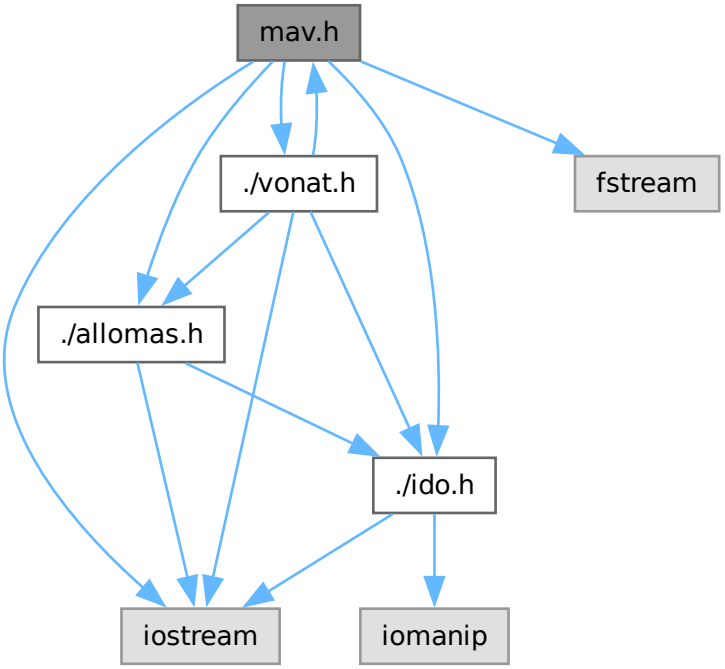
```
00031     v.setIndulas(seged.indulas);
00032     v.setErkezes(seged.erkezes);
00033
00034     // feltöltöm a MAV "mgmt" class Vonat* tömbjét
00035     this->add(v);
00036   }
00037   file.close();
00038 } // END OF BEOLVAS
00039
00040 /**
00041  * Mav beolvas overwrite
00042  * Csak a teszt miatt kell, meg kell tudnom neki adni egy
00043  * teszt vonatok.txt-t
00044  * @param file file path (const char*)
00045  */
00046 void Mav::beolvas(const char* test) {
00047   // Inicalizálás
00048   std::ifstream file(test);
00049   Vonat v;
00050   Seged seged;
00051
00052   // Kiürítem a vonatok kollekciót, hogy ne legyen ráolvasás
00053   delete[] vonatok;
00054   vonatok = nullptr;
00055   si = 0;
00056
00057   while (file » seged.szam » seged.indulo » seged.veg » seged.kocsidb » seged.indulas » seged.erkezes)
    {
00058     v.setSzam(std::atoi(seged.szam));
00059     v.setIndulo(seged.indulo);
00060     v.setVeg(seged.veg);
00061     v.setKocsidb(std::atoi(seged.kocsidb));
00062     v.setIndulas(seged.indulas);
00063     v.setErkezes(seged.erkezes);
00064
00065     // feltöltöm a MAV "mgmt" class Vonat* tömbjét
00066     this->add(v);
00067   }
00068   file.close();
00069 } // END OF BEOLVAS
00070
00071
00072 /*
00073  * Vonatok hozzáadása
00074  * @param currPos a streamben elfoglalt helyem
00075  * @param minden-más a vonat class feltöltéséhez szükséges adatok
00076 */
00077 void Mav::addTrain(std::streampos& currPos, int szam, Allomas indulo,
00078             Allomas veg, int kocsidb, Ido indulas, Ido erkezes){
00079   // Inicalizálás
00080   const char* vonatok = "./vonatok.txt";
00081   std::ofstream file (vonatok, std::ios::app); // Hozzáfűzésesen nyitom meg.
00082
00083   // HIBAKEZELÉS IDE
00084   if (!file.is_open()) {
00085     std::cout « "Megynitással van baj " « vonatok « std::endl;
00086     return;
00087   }
00088   // end of HIBAKEZELÉS
00089
00090   if(file.is_open()){
00091     file.seekp(currPos); // Megfelelő helyre ugrok
00092
00093     file « szam « " " « indulo « " " « veg « " " « kocsidb « " "
00094          « indulas « " " « erkezes « std::endl;
00095   }
00096
00097   file.close();
00098 } // END OF addTrain
00099
00100 /**
00101 * addTrain() teszt miatti overwriteja
00102 * Csak abban különbözik, hogy a fájl amit megnyit nem a hard coded
00103 * hanem a test file
00104   */
00105 void Mav::addTrain(std::streampos& currPos, const char* test, int szam, Allomas indulo,
00106             Allomas veg, int kocsidb, Ido indulas, Ido erkezes){
00107   // Inicalizálás
00108   const char* vonatok = test;
00109   std::ofstream file (vonatok, std::ios::app); // Hozzáfűzésesen nyitom meg.
00110
00111   if(file.is_open()){
00112     file.seekp(currPos); // Megfelelő helyre ugrok
00113
00114     file « szam « " " « indulo « " " « veg « " " « kocsidb « " "
00115          « indulas « " " « erkezes « std::endl;
00116   }
```

```
00117
00118   file.close();
00119 } // END OF addTrain
00120
00121
```

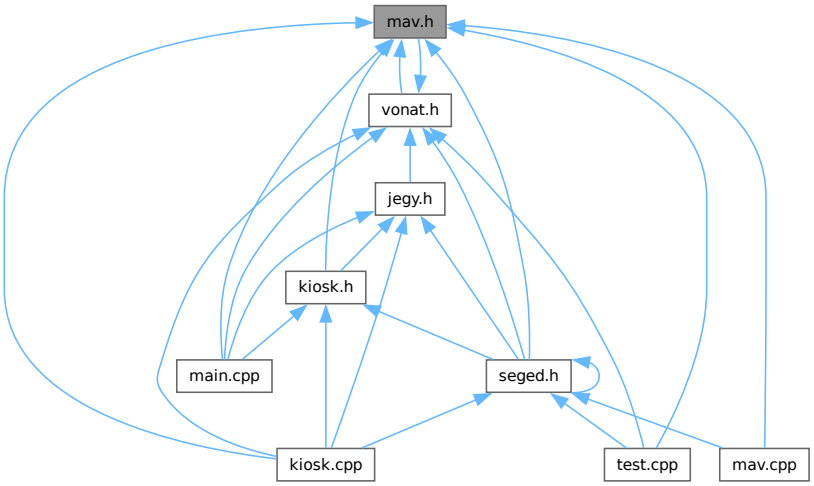
5.19 mav.h File Reference

```
#include <iostream>
#include <fstream>
#include "./vonat.h"
#include "./allomas.h"
#include "./ido.h"
```

Include dependency graph for mav.h:



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



Data Structures

- class Mav

5.20 mav.h

Go to the documentation of this file.

```
00001 #ifndef MAV_H
00002 #define MAV_H
00003
00004 /*
00005  * FILE MAV_H
00006  */
00007
00008 #include <iostream>
00009 #include <fstream>
00010
00011 #include "./vonat.h"
00012 #include "./allomas.h"
00013 #include "./ido.h"
00014
00015
00016 class Mav {
00017   Vonat* vonatok; // Vonatok kollekció
00018   size_t si;      // Vonatok kollekció mérete db számban
00019
00020 public:
00021   // Konstruktor
00022   Mav(size_t si = 0) : vonatok(nullptr), si(si) {}
00023
00024   /**
00025    * Elem hozzáfűzése Vonatok kollekcióhoz
00026    * @param vonat vonat referencia amit hozzáad
00027    */
00028   void add(Vonat& vonat){
00029     Vonat* temp = new Vonat[si + 1];  // Lefoglalok helyet
00030
00031     for (size_t i = 0; i < si; ++i) { // Átmásolom a tömb elemeit
00032       temp[i] = vonatok[i];
00033     }
00034     temp[si++] = vonat;               // Beleteszem a vonatot
00035
00036     delete[] vonatok;                   // "Régi" tömböt törlöm
00037     vonatok = temp;                     //
```

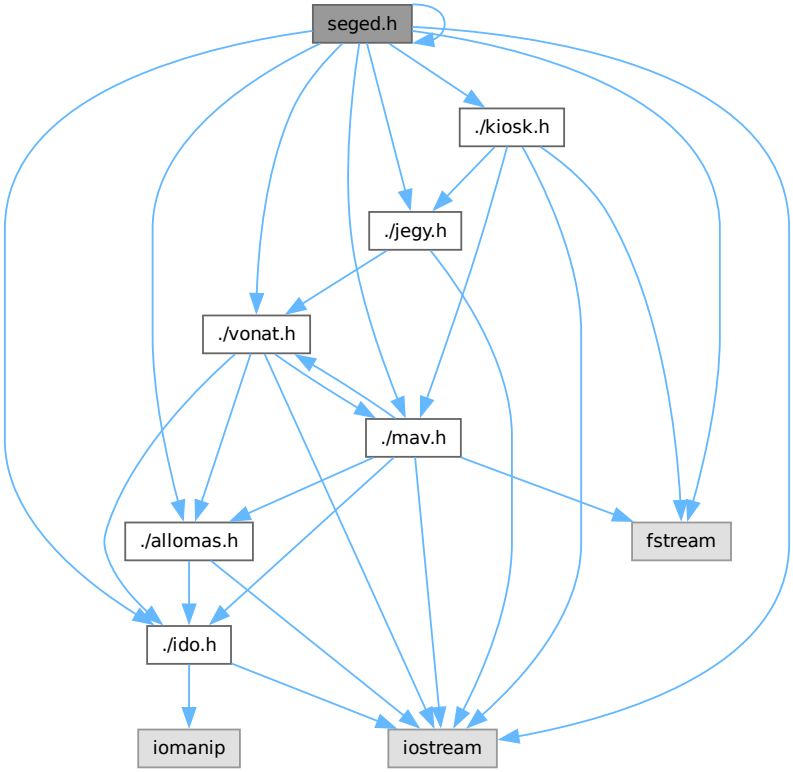
```
00038   } // End of feltolt
00039
00040   // Vonatok tömb kiírása
00041   void kiir() {
00042     for (size_t i = 0; i < si; ++i) {
00043       std::cout « "--- " « i+1 « ".vonat ---" « std::endl;
00044         vonatok[i].kiir();
00045         std::cout « std::endl;
00046     }
00047   } // end of kiir
00048
00049   // i.edik elem kiírása
00050   void kiirAt(int i){
00051     // HIBAEKEZELÉS
00052     std::cout « i+1 «".vonat" « std::endl;
00053     vonatok[i].kiir();
00054   } // end of kiirAt
00055
00056   Vonat& getVonatAt(int i){ return vonatok[i-1]; }
00057   size_t getSize() { return static_cast<int>(si); }
00058
00059
00060   /**
00061   * Beolvaásás a file-ból egy segéd strukturába, onnan pedig egy Vonat objektum feltöltés
00062   * @param Mav& mav class feltöltéséhez szükséges
00063   */
00064   void beolvas();
00065
00066   /**
00067   * Mav beolvas overwrite
00068   * Csak a teszt miatt kell, meg kell tudnom neki adni egy
00069   * teszt vonatok.txt-t
00070   * @param file file path (const char*)
00071   */
00072   void beolvas(const char* file);
00073
00074 /*
00075   * Vonatok hozzáadása
00076   * @param currPos a streamben elfoglalt helyem
00077   * @param minden-más a vonat class feltöltéséhez szükséges adatok
00078 */
00079   void addTrain(std::streampos& currPos, int szam, Allomas indulo,
00080                 Allomas veg, int kocsidb, Ido indulas, Ido erkezes);
00081
00082
00083   /**
00084 * addTrain() teszt miatti overwriteja
00085 * Csak abban különbözik, hogy a fájl amit megnyit nem a hard coded
00086 * hanem a test file
00087   */
00088   void addTrain(std::streampos& currPos, const char* test, int szam, Allomas indulo,
00089                     Allomas veg, int kocsidb, Ido indulas, Ido erkezes);
00090
00091   /// op[] overload
00092   Vonat& operator[](int idx){ return vonatok[idx]; }
00093
00094   /// Destruktor
00095   ~Mav(){
00096     delete[] vonatok;
00097   }
00098
00099 }; // END OF MAV
00100
00101
00102
00103
00104 #endif // !MAV_H
00105
```

5.21 seged.h File Reference

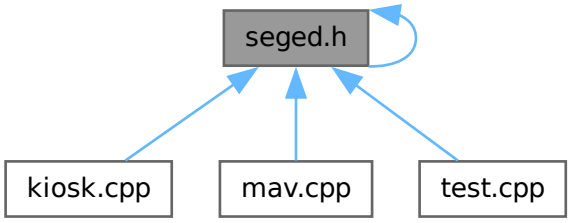
```
#include <iostream>
#include <fstream>
#include "./allomas.h"
#include "./ido.h"
#include "./jegy.h"
#include "./kiosk.h"
#include "./mav.h"
```

```
#include "./seged.h"
#include "./vonat.h"
```

Include dependency graph for seged.h:



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



Data Structures

- struct Seged

5.22 seged.h

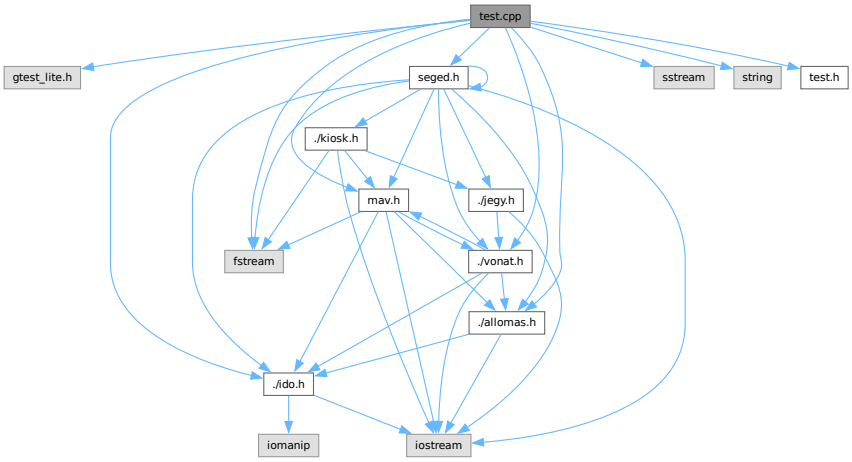
Go to the documentation of this file.

```
00001 #ifndef SEGED_H
00002 #define SEGED_H
00003
00004 #include <iostream>
00005 #include <fstream>
00006
00007 #include "./allomas.h"
00008 #include "./ido.h"
00009 #include "./jegy.h"
00010 #include "./kiosk.h"
00011 #include "./mav.h"
00012 #include "./seged.h"
00013 #include "./vonat.h"
00014
00015
00016 /**
00017 * Magyarország leghosszabb településneve 15 karakter
00018 * 25 karakterbe bele kell férnia
00019 * Egyszerű segédstruktúra a beolvasáshoz segédkezett mind midle man
00020 */
00021 struct Seged {
00022    static const int buffSize = 25;
00023    char szam[buffSize];
00024    char indulo[buffSize];
00025    char veg[buffSize];
00026    char kocsidb[buffSize];
00027    char indulas[buffSize];
00028    char erkezes[buffSize];
00029 };
00030
00031
00032
00033 #endif // !SEGED_H
00034
```

5.23 test.cpp File Reference

```
#include "gtest_lite.h"
#include "mav.h"
#include "vonat.h"
#include "allomas.h"
#include "ido.h"
#include "seged.h"
#include <sstream>
#include <fstream>
#include <string>
#include "test.h"
```

Include dependency graph for test.cpp:



Functions

- void mav_test ()
- void vonat_test ()

Variables

- Mav mav

5.23.1 Function Documentation

5.23.1.1 mav_test()

```
void mav_test ( )
```

TEST FILE MAV TEST Mav def ctor test

Mav ctor test

Mav add() test default ctor

addTrain metódust tesztel a mav classnak

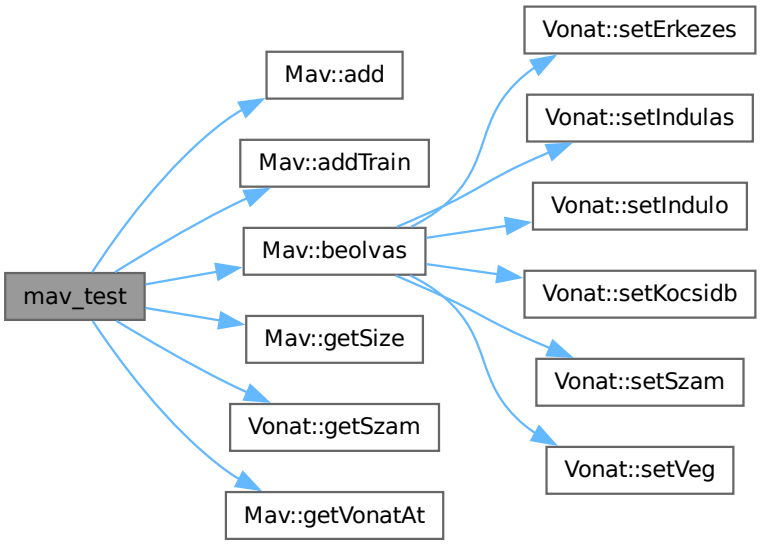
Beolvas() teszt Mav class beolvas() metódust tesztel

Definition at line 21 of file test.cpp.

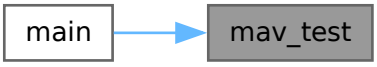
```
00021                 {
00022   /// Mav def ctor test
00023   TEST(Mav, default ctor){
00024     Mav mav;
00025     // EXPECT_EQ(0, mav.getSize());
00026   } ENDM;
00027
00028   /// Mav ctor test
00029   TEST(Mav, ctor){
00030     Mav mav(10);
```

```
00031     EXPECT_EQ(int(10), int(mav.getSize()));
00032   } ENDM;
00033
00034   /// Mav add() test default ctor
00035   TEST(Mav, default add()){
00036     Mav mav;
00037     Vonat v; // default ctor
00038
00039     mav.add(v);
00040     EXPECT_EQ(size_t(1) , mav.getSize()); // növelte a méretét
00041     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 0); // 1 mivel a getvonat i-1 el számol
00042   } ENDM;
00043
00044   // Mav add test parraméteres ctor
00045   TEST(Mav, add()){
00046     Mav mav;
00047     Vonat vonat1(1, "Budapest", "Szeged", 10, Ido(8, 0), Ido(10, 0));
00048     Vonat vonat2(2, "Szeged", "Debrecen", 8, Ido(12, 0), Ido(14, 0));
00049     mav.add(vonat1);
00050     mav.add(vonat2);
00051
00052     // EXPECT_EQ(mav.getSize(), 2);
00053     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 1);
00054     EXPECT_EQ(mav.getVonatAt(2).getSzam(), 2);
00055   } ENDM;
00056
00057   // Nagyon sok helyről összeollózott teszt.
00058   /// addTrain metódust tesztel a mav classnak
00059   TEST(Mav, AddTrain()) {
00060     // Csinálok egy test file-t
00061     const char* testFile = "./test_vonatok.txt";
00062     std::ofstream file(testFile);
00063     file.close();
00064
00065     // Inicializálás
00066     Mav mav;
00067     std::streampos currPos = 0; // Nullától olvasson be.
00068     Allomas indulo("Szeged");
00069     Allomas veg("Debrecen");
00070     Ido indulas(12, 0);
00071     Ido erkezes(14, 0);
00072
00073     // Tényleges fv meghívása
00074     mav.addTrain(currPos, testFile, 2, indulo, veg, 8, indulas, erkezes);
00075
00076     std::ifstream beolvasott("./test_vonatok.txt");
00077     std::stringstream buffer;
00078     buffer « beolvasott.rdbuf(); // ???, de működik
00079     beolvasott.close();
00080
00081     std::string elvart = "2 Szeged Debrecen 8 1200 1400\n";
00082     EXPECT_EQ(buffer.str(),elvart);
00083
00084     // Megnyilkolom a teszt file-t
00085     std::remove("./test_vonatok.txt");
00086   } ENDM;
00087
00088   /// Beolvas() teszt
00089   /// Mav class beolvas() metódust tesztel
00090   TEST(Mav, Beolvas()) {
00091     // Mindenhonnan összeollózott, nincs konkrét forrás
00092     const char* test = "./test_vonatok.txt";
00093     std::ofstream file(test);
00094     file « "1 Budapest Szeged 10 0800 1000\n"
00095          « "2 Szeged Debrecen 8 1200 1400\n";
00096     file.close();
00097
00098     Mav mav;
00099     mav.beolvas(test);
00100     EXPECT_EQ(mav.getSize(),size_t(2));
00101     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 1);
00102     EXPECT_EQ(mav.getVonatAt(2).getSzam(), 2);
00103
00104     std::remove("./test_vonatok.txt");
00105   } ENDM;
00106
00107 } // end of mav test
```

Here is the call graph for this function:



Here is the caller graph for this function:



5.23.1.2 vonat_test()

```
void vonat_test ( )
```

Vonat def ctor test

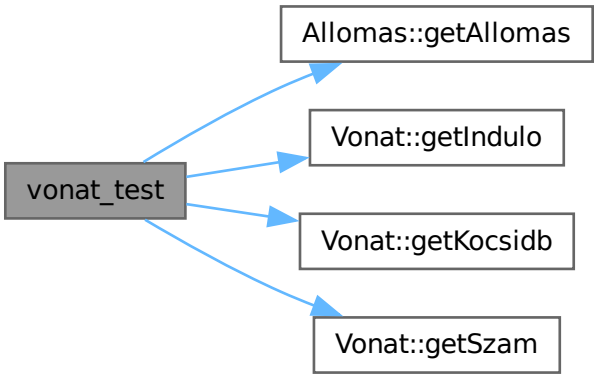
Vonat ctor test

Definition at line 110 of file test.cpp.

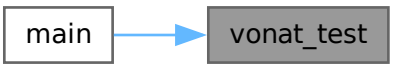
```
00110                 {
00111
00112   /// Vonat def ctor test
00113   TEST(Vonat, default ctor){
00114     Vonat v;
00115
00116     EXPECT_EQ(0, v.getSzam());
00117     EXPECT_EQ("", std::string(v.getIndulo().getAllomas()));
00118
00119
```

```
00120
00121   } ENDM;
00122
00123   /// Vonat ctor test
00124   TEST(Vonat, ctor){
00125     Vonat v(1, "Budapest", "Szeged", 10, Ido(8, 0), Ido(10, 0));
00126
00127     EXPECT_EQ(1, v.getSzam());
00128     EXPECT_EQ(10, v.getKocsidb());
00129     EXPECT_EQ("Budapest", std::string(v.getIndulo().getAllomas()));
00130
00131   } ENDM;
00132
00133 } // end of vonat test
```

Here is the call graph for this function:



Here is the caller graph for this function:



5.23.2 Variable Documentation

5.23.2.1 mav

```
Mav mav
```

Definition at line 13 of file test.cpp.

5.24 test.cpp

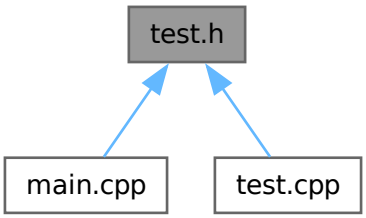
Go to the documentation of this file.

```
00001 #include "gtest_lite.h"
00002 #include "mav.h"
00003 #include "vonat.h"
00004 #include "allomas.h"
00005 #include "ido.h"
00006 #include "seged.h"
00007
00008 #include <sstream>
00009 #include <fstream>
00010 #include <string>
00011
00012 #include "test.h"
00013 Mav mav;
00014 /**
00015 * TEST FILE
00016 */
00017
00018 /**
00019 * MAV TEST
00020 */
00021 void mav_test(){
00022   /// Mav def ctor test
00023   TEST(Mav, default ctor){
00024     Mav mav;
00025     // EXPECT_EQ(0, mav.getSize());
00026   } ENDM;
00027
00028   /// Mav ctor test
00029   TEST(Mav, ctor){
00030     Mav mav(10);
00031     EXPECT_EQ(int(10), int(mav.getSize()));
00032   } ENDM;
00033
00034   /// Mav add() test default ctor
00035   TEST(Mav, default add()){
00036     Mav mav;
00037     Vonat v; // default ctor
00038
00039     mav.add(v);
00040     EXPECT_EQ(size_t(1) , mav.getSize()); // növelte a méretét
00041     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 0); // 1 mivel a getvonat i-1 el számol
00042   } ENDM;
00043
00044   // Mav add test parraméteres ctor
00045   TEST(Mav, add()){
00046     Mav mav;
00047     Vonat vonat1(1, "Budapest", "Szeged", 10, Ido(8, 0), Ido(10, 0));
00048     Vonat vonat2(2, "Szeged", "Debrecen", 8, Ido(12, 0), Ido(14, 0));
00049     mav.add(vonat1);
00050     mav.add(vonat2);
00051
00052     // EXPECT_EQ(mav.getSize(), 2);
00053     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 1);
00054     EXPECT_EQ(mav.getVonatAt(2).getSzam(), 2);
00055   } ENDM;
00056
00057   // Nagyon sok helyről összeollózott teszt.
00058   /// addTrain metódust tesztel a mav classnak
00059   TEST(Mav, AddTrain()) {
00060     // Csinálok egy test file-t
00061     const char* testFile = "./test_vonatok.txt";
00062     std::ofstream file(testFile);
00063     file.close();
00064
00065     // Inicializálás
00066     Mav mav;
00067     std::streampos currPos = 0; // Nullától olvasson be.
00068     Allomas indulo("Szeged");
00069     Allomas veg("Debrecen");
00070     Ido indulas(12, 0);
00071     Ido erkezes(14, 0);
00072
00073     // Tényleges fv meghívása
00074     mav.addTrain(currPos, testFile, 2, indulo, veg, 8, indulas, erkezes);
00075
00076     std::ifstream beolvasott("./test_vonatok.txt");
00077     std::stringstream buffer;
00078     buffer « beolvasott.rdbuf(); // ???, de működik
00079     beolvasott.close();
00080
00081     std::string elvart = "2 Szeged Debrecen 8 1200 1400\n";
00082     EXPECT_EQ(buffer.str(),elvart);
```

```
00083
00084     // Megnyilkolom a teszt file-t
00085     std::remove("./test_vonatok.txt");
00086   } ENDM;
00087
00088   /// Beolvas() teszt
00089   /// Mav class beolvas() metódust tesztel
00090   TEST(Mav, Beolvas()) {
00091     // Mindenhonnan összeollózott, nincs konkrét forrás
00092     const char* test = "./test_vonatok.txt";
00093     std::ofstream file(test);
00094     file « "1 Budapest Szeged 10 0800 1000\n"
00095          « "2 Szeged Debrecen 8 1200 1400\n";
00096     file.close();
00097
00098     Mav mav;
00099     mav.beolvas(test);
00100     EXPECT_EQ(mav.getSize(),size_t(2));
00101     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 1);
00102     EXPECT_EQ(mav.getVonatAt(2).getSzam(), 2);
00103
00104     std::remove("./test_vonatok.txt");
00105   } ENDM;
00106
00107 } // end of mav test
00108
00109
00110 void vonat_test(){
00111
00112   /// Vonat def ctor test
00113   TEST(Vonat, default ctor){
00114     Vonat v;
00115
00116     EXPECT_EQ(0, v.getSzam());
00117     EXPECT_EQ("", std::string(v.getIndulo().getAllomas()));
00118
00119
00120
00121   } ENDM;
00122
00123   /// Vonat ctor test
00124   TEST(Vonat, ctor){
00125     Vonat v(1, "Budapest", "Szeged", 10, Ido(8, 0), Ido(10, 0));
00126
00127     EXPECT_EQ(1, v.getSzam());
00128     EXPECT_EQ(10, v.getKocsidb());
00129     EXPECT_EQ("Budapest", std::string(v.getIndulo().getAllomas()));
00130
00131   } ENDM;
00132
00133 } // end of vonat test
```

5.25 test.h File Reference

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



Functions

- void mav_test ()
- void vonat_test ()

5.25.1 Function Documentation

5.25.1.1 mav_test()

```
void mav_test ( )
```

TEST FILE MAV TEST Mav def ctor test

Mav ctor test

Mav add() test default ctor

addTrain metódust tesztel a mav classnak

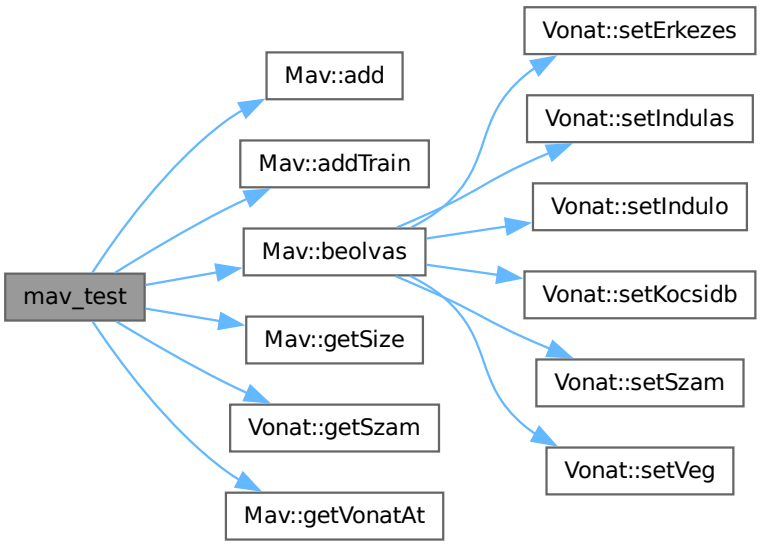
Beolvas() teszt Mav class beolvas() metódust tesztel

Definition at line 21 of file test.cpp.

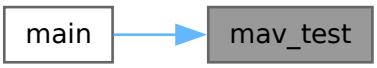
```
00021                {
00022   /// Mav def ctor test
00023   TEST(Mav, default ctor){
00024     Mav mav;
00025     // EXPECT_EQ(0, mav.getSize());
00026   } ENDM;
00027
00028   /// Mav ctor test
00029   TEST(Mav, ctor){
00030     Mav mav(10);
00031     EXPECT_EQ(int(10), int(mav.getSize()));
00032   } ENDM;
00033
00034   /// Mav add() test default ctor
00035   TEST(Mav, default add()){
00036     Mav mav;
00037     Vonat v; // default ctor
00038
00039     mav.add(v);
00040     EXPECT_EQ(size_t(1) , mav.getSize()); // növelte a méretét
00041     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 0); // 1 mivel a getvonat i-1 el számol
00042   } ENDM;
00043
00044   // Mav add test parraméteres ctor
00045   TEST(Mav, add()){
00046     Mav mav;
00047     Vonat vonat1(1, "Budapest", "Szeged", 10, Ido(8, 0), Ido(10, 0));
00048     Vonat vonat2(2, "Szeged", "Debrecen", 8, Ido(12, 0), Ido(14, 0));
00049     mav.add(vonat1);
00050     mav.add(vonat2);
00051
00052     // EXPECT_EQ(mav.getSize(), 2);
00053     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 1);
00054     EXPECT_EQ(mav.getVonatAt(2).getSzam(), 2);
00055   } ENDM;
00056
00057   // Nagyon sok helyről összeollózott teszt.
00058   /// addTrain metódust tesztel a mav classnak
00059   TEST(Mav, AddTrain()) {
00060     // Csinálok egy test file-t
00061     const char* testFile = "./test_vonatok.txt";
00062     std::ofstream file(testFile);
00063     file.close();
00064
00065     // Inicializálás
00066     Mav mav;
00067     std::streampos currPos = 0; // Nullától olvasson be.
00068     Allomas indulo("Szeged");
00069     Allomas veg("Debrecen");
00070     Ido indulas(12, 0);
00071     Ido erkezes(14, 0);
```

```
00072
00073     // Tényleges fv meghívása
00074     mav.addTrain(currPos, testFile, 2, indulo, veg, 8, indulas, erkezes);
00075
00076     std::ifstream beolvasott("./test_vonatok.txt");
00077     std::stringstream buffer;
00078     buffer « beolvasott.rdbuf(); // ???, de működik
00079     beolvasott.close();
00080
00081     std::string elvart = "2 Szeged Debrecen 8 1200 1400\n";
00082     EXPECT_EQ(buffer.str(),elvart);
00083
00084     // Megnyilkolom a teszt file-t
00085     std::remove("./test_vonatok.txt");
00086   } ENDM;
00087
00088   /// Beolvas() teszt
00089   /// Mav class beolvas() metódust tesztel
00090   TEST(Mav, Beolvas()) {
00091     // Mindenhonnan összeollózott, nincs konkrét forrás
00092     const char* test = "./test_vonatok.txt";
00093     std::ofstream file(test);
00094     file « "1 Budapest Szeged 10 0800 1000\n"
00095          « "2 Szeged Debrecen 8 1200 1400\n";
00096     file.close();
00097
00098     Mav mav;
00099     mav.beolvas(test);
00100     EXPECT_EQ(mav.getSize(),size_t(2));
00101     EXPECT_EQ(mav.getVonatAt(1).getSzam(), 1);
00102     EXPECT_EQ(mav.getVonatAt(2).getSzam(), 2);
00103
00104     std::remove("./test_vonatok.txt");
00105   } ENDM;
00106
00107 } // end of mav test
```

Here is the call graph for this function:



Here is the caller graph for this function:



5.25.1.2 vonat_test()

```
void vonat_test ( )
```

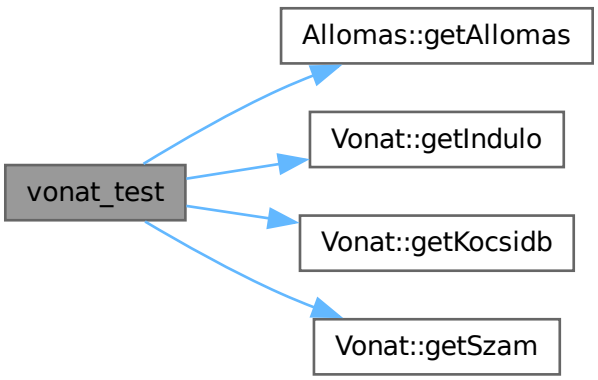
Vonat def ctor test

Vonat ctor test

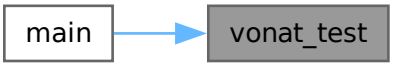
Definition at line 110 of file test.cpp.

```
00110                   {
00111
00112   /// Vonat def ctor test
00113   TEST(Vonat, default ctor){
00114     Vonat v;
00115
00116     EXPECT_EQ(0, v.getSzam());
00117     EXPECT_EQ("", std::string(v.getIndulo().getAllomas()));
00118
00119
00120
00121   } ENDM;
00122
00123   /// Vonat ctor test
00124   TEST(Vonat, ctor){
00125     Vonat v(1, "Budapest", "Szeged", 10, Ido(8, 0), Ido(10, 0));
00126
00127     EXPECT_EQ(1, v.getSzam());
00128     EXPECT_EQ(10, v.getKocsidb());
00129     EXPECT_EQ("Budapest", std::string(v.getIndulo().getAllomas()));
00130
00131   } ENDM;
00132
00133 } // end of vonat test
```

Here is the call graph for this function:



Here is the caller graph for this function:



5.26 test.h

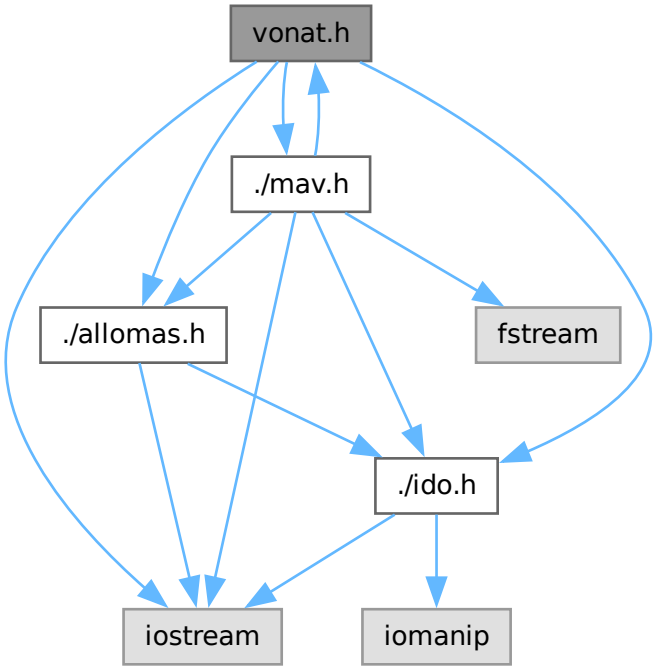
Go to the documentation of this file.

```
00001 #ifndef TEST_H
00002 #define TEST_H
00003
00004   void mav_test();
00005   void vonat_test();
00006
00007
00008 #endif // !TEST_H
```

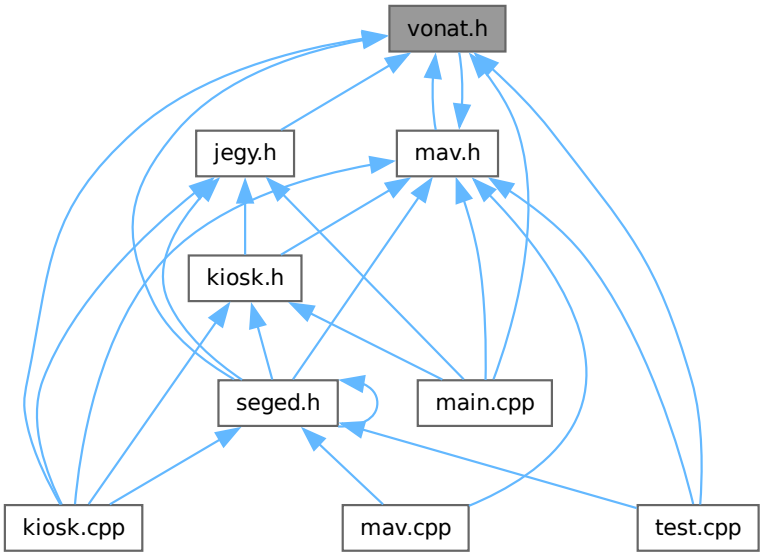
5.27 vonat.h File Reference

```
#include <iostream>
#include "./mav.h"
#include "./allomas.h"
#include "./ido.h"
```

Include dependency graph for vonat.h:



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



Data Structures

- class Vonat

5.28 vonat.h

Go to the documentation of this file.

```
00001 #ifndef VONAT_H
00002 #define VONAT_H
00003
00004 /*
00005  * file VONAT_H
00006 */
00007 #include <iostream>
00008
00009 #include "./mav.h"
00010 #include "./allomas.h"
00011 #include "./ido.h"
00012
00013 class Mav; // Nem ette meg a headerből, nem értem miért.
00014
00015 class Vonat {
00016 private:
00017   int szam;       /// Vonatszám
00018   Allomas indulo; /// Kiinduló állomas
00019   Allomas veg;    /// Végállomas
00020   int kocsidb;    /// Kocsik darabszáma, basically semmit sem csinál
00021   Ido indulas;    /// Indulási idő
00022   Ido erkezes;    /// Érkezési idp
00023
00024 public:
00025   /// Paraméter nélküli Konstruktor
00026   Vonat() : szam(0), indulo(""), veg(""), kocsidb(0), indulas(0,0), erkezes(0,0) {}
00027
00028   /// Paraméteres Konstruktor
00029   Vonat(int vszam, Allomas indulop, Allomas vegp, int kocsidb, Ido indulasp, Ido erkezesp)
00030         : szam(vszam), indulo(indulop), veg(vegp), kocsidb(kocsidb), indulas(indulasp),
      erkezes(erkezesp) {}
00031
00032   /// Setterek
00033   void setSzam(int szam)              { this->szam = szam; }
00034   void setIndulo(const char* indulo)  { this-> indulo = Allomas(indulo); }
00035   void setVeg(const char* veg)        { this->veg = Allomas(veg); }
00036   void setKocsidb(int kocsidb )       { this->kocsidb = kocsidb; }
00037   void setIndulas(const char* ido)    { this->indulas = Ido(ido); }
00038   void setErkezes(const char* ido)    { this->erkezes = Ido(ido); }
00039
00040   /// Getterek
00041   int getSzam() const       { return szam; }
00042   Allomas getIndulo() const { return indulo; }
00043   Allomas getVeg() const    { return veg; }
00044   int getKocsidb() const    { return kocsidb; }
00045   Ido getIndulas() const    { return indulas; }
00046   Ido getErkezes() const    { return erkezes; }
00047
00048   /**
00049   * Kiírja a vonat adatait
00050   * Főképp a jegyváltáskor van meghívva + listázás.
00051   */
00052   void kiir() const {
00053     std::cout « "Vonat szama: " « szam « std::endl;
00054     std::cout « "Indulasi allomas: " « indulo « std::endl;
00055     std::cout « "Erkezesi allomas: " « veg « std::endl;
00056     std::cout « "Kocsi darabszam: " « kocsidb « std::endl;
00057
00058     std::cout « "Indulas idopontja: ";
00059     std::cout « std::setw(2) « std::setfill('0') «indulas.getOra();
00060     std::cout « ":";
00061     std::cout « std::setw(2) « std::setfill('0') « indulas.getPerc() « std::endl; // Nagyon ronda,
      tudom...
00062
00063     std::cout « "Érkezés idopontja: ";
00064     std::cout « std::setw(2) « std::setfill('0') « erkezes.getOra();
00065     std::cout « ":";
00066     std::cout « std::setw(2) « std::setfill('0') « erkezes.getPerc() « std::endl; // Nagyon ronda,
      tudom...
00067
00068   }
00069
00070
```

```
00071 }; // end of VONAt
00072
00073
00074 #endif // !VONAT_H
```