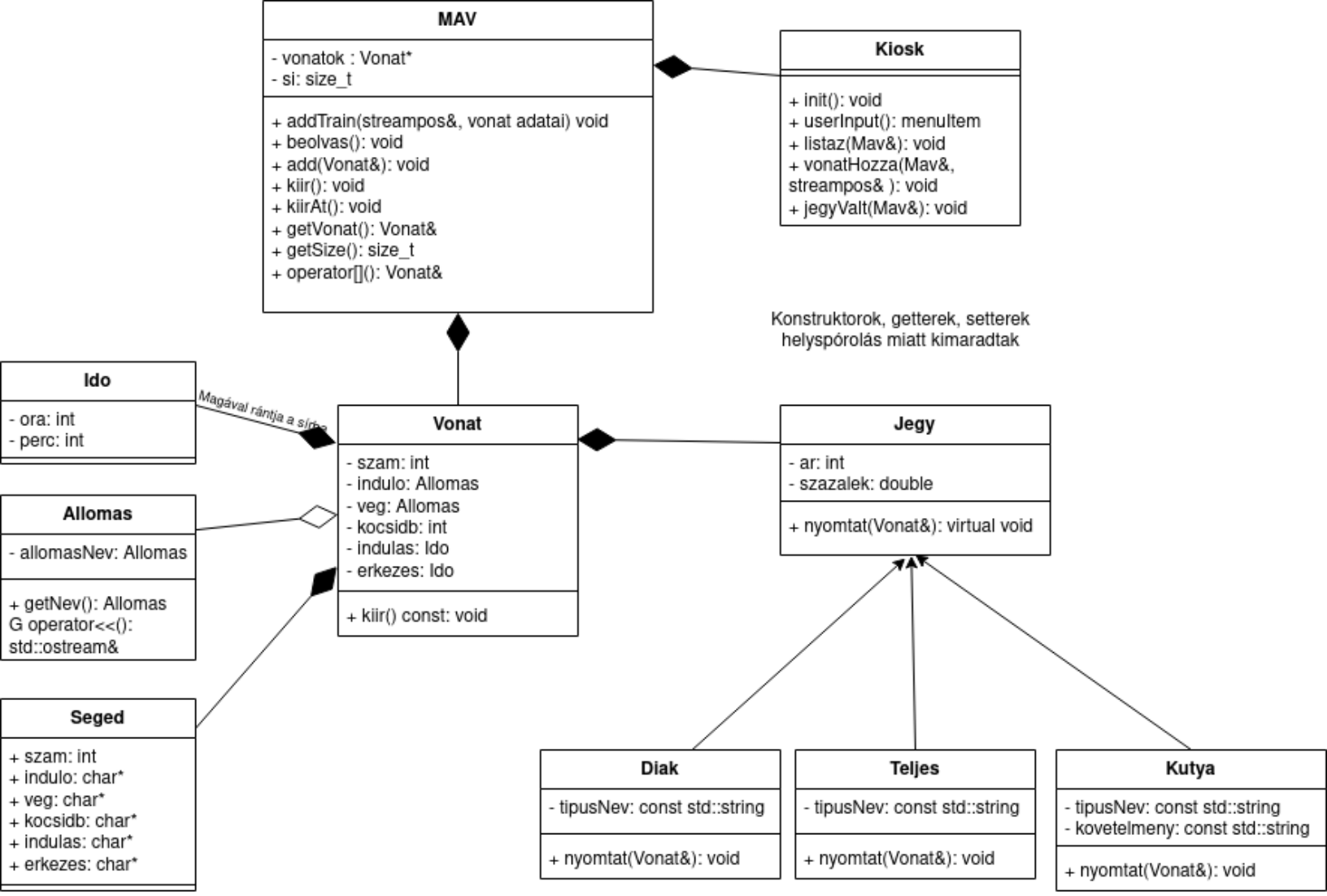
Házi feladat

1.0

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 sínbe
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:

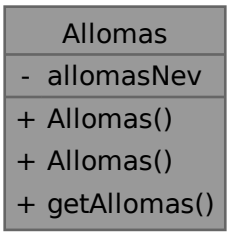
Chapter 4

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.

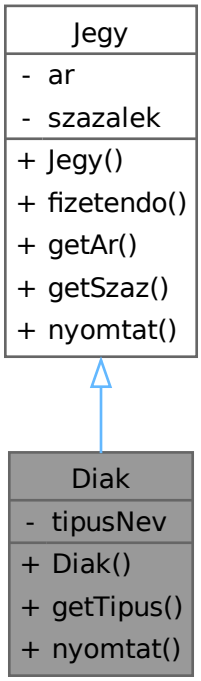
The documentation for this class was generated from the following file:

- allomas.h

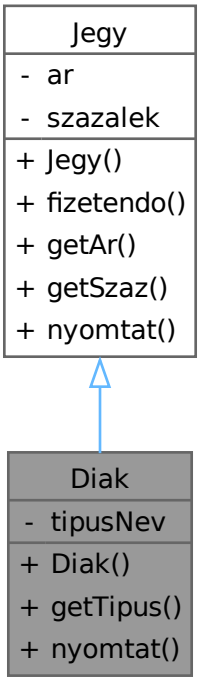
4.2 Diak Class Reference

```
#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]
```

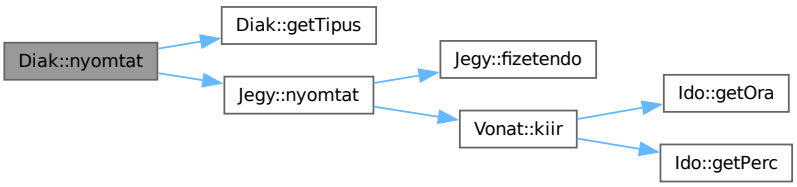
megörökölt nyomtat fv overwrite-ja meghívja az "eredei", főosztály nyomtatját

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.

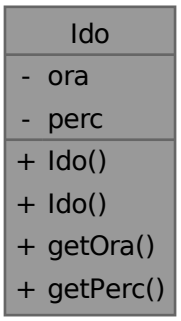
The documentation for this class was generated from the following file:

- 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 22 of file ido.h.

```
00022                         {
00023     ora = (ido[0] - '0') * 10 + (ido[1] - '0');
00024     perc = (ido[2] - '0') * 10 + (ido[3] - '0');
00025   }
```

4.3.2.2 Ido() [2/2]

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

Definition at line 27 of file ido.h.

```
00027 : ora(ora), perc(perc) {}
```

4.3.3 Member Function Documentation

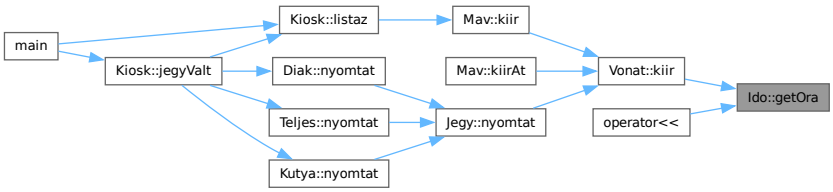
4.3.3.1 getOra()

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

Definition at line 36 of file ido.h.

```
00036 { return ora; };
```

Here is the caller graph for this function:



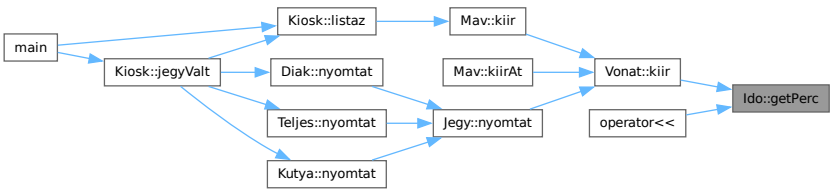
4.3.3.2 getPerc()

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

Definition at line 37 of file ido.h.

```
00037 {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.

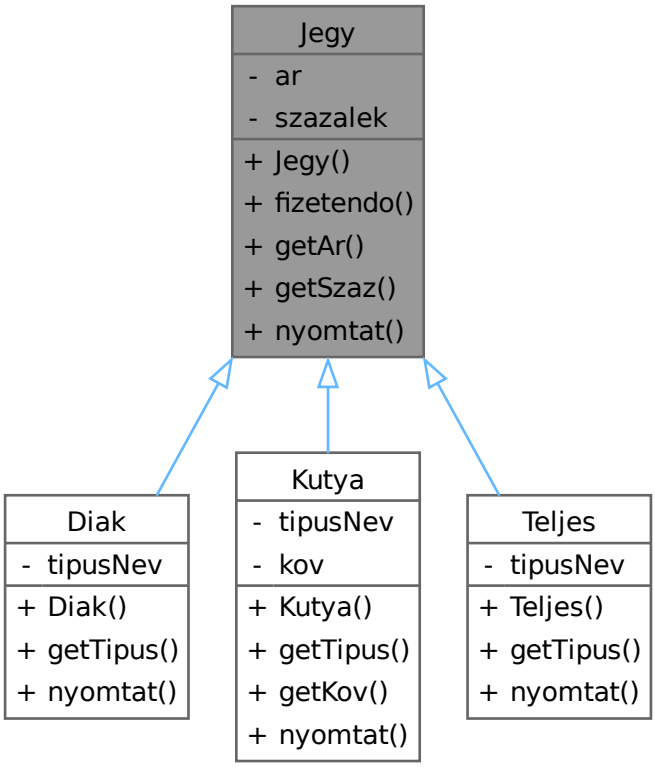
The documentation for this class was generated from the following file:

- 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

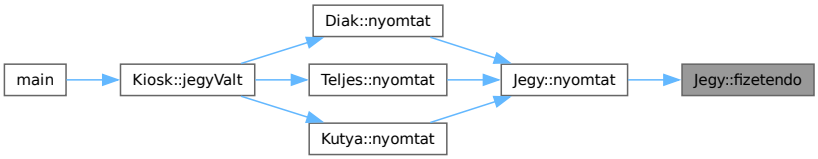
4.4.3.1 fizetendo()

```
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

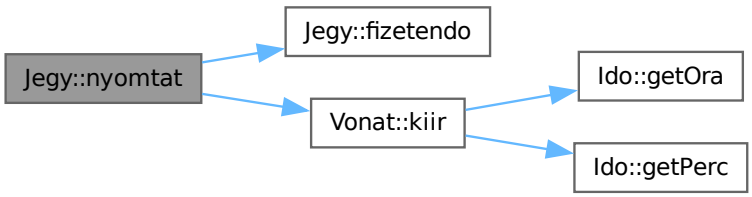
Vonat&	megadott vonatra vonatkozik Kíírja a jegy adatinak egy felét. Azt a felét ami minden leszármazottál azonos. A felső pár so. virtual mivel a leszármazottak megöröklik és felülírják.

Reimplemented in Diak, Teljes, and Kutya.

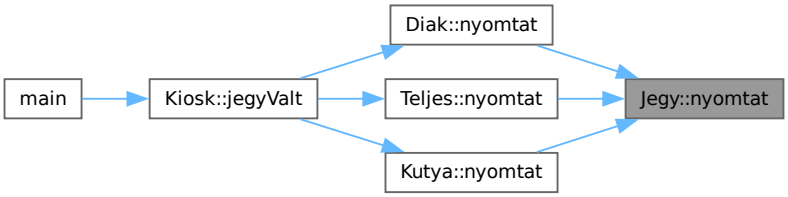
Definition at line 36 of file jegy.h.

```
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.

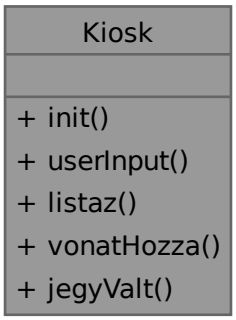
The documentation for this class was generated from the following file:

- jegy.h

4.5 Kiosk Class Reference

```
#include <kiosk.h>
```

Collaboration diagram for Kiosk:



Public Member Functions

- 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()

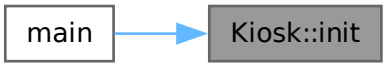
```
void Kiosk::init ( )
```

Inicializálja a menüt, ergo kiírja a lehetőségeket.

Definition at line 13 of file kiosk.cpp.

```
00013                {
00014   std::cout « "### Dönts ###" « std::endl;
00015   std::cout « "1. Vonat hozzáadása "     « std::endl;
00016   std::cout « "2. Vonatok listázása "    « std::endl;
00017   std::cout « "3. Jegy nyomtatása"       « std::endl;
00018   std::cout « "4. Kilépés"               « std::endl;
00019 }
```

Here is the caller graph for this function:



4.5.2.2 jegyValt()

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

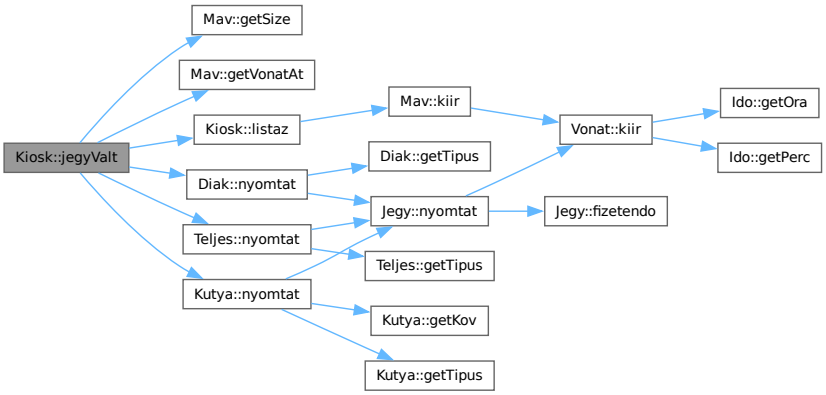
jegyet vált a kiválasztott vonatra

Definition at line 53 of file kiosk.cpp.

```
00053                              {
00054   int idx, buf;
00055   listaz(mav);
00056   std::cout « "Valasz vonatot: ";
00057   std::cin » idx;
00058   // Hibakezelés
00059   if(size_t(idx) > mav.getSize()){ std::cout « "Túlindexelés\n"; return; }
00060   std::cout « std::endl;
00061
00062   std::cout « "1.Teljes, 2.Diak, 3.Kutya\n ";
00063   std::cin » buf;
00064   // Hibakezelés
00065   if(buf > 3){ std::cout « "Túlindexelés"; return; }
00066   std::cout « std::endl;
00067
00068   switch (buf) {
00069     case 1: {
00070       Teljes t;
00071       t.nyomtat(mav.getVonatAt(idx));
00072       break;
00073     }
00074
00075     case 2: {
00076       Diak d;
00077       d.nyomtat(mav.getVonatAt(idx));
00078       break;
00079     }
00080
00081     case 3: {
00082       Kutya k;
```

```
00083         k.nyomtat(mav.getVonatAt(idx));
00084         break;
00085       }
00086
00087     } // end of switch
00088
00089 } // end of jegyValt
```

Here is the call graph for this function:



Here is the caller graph for this function:

4.5.2.3 listaz()

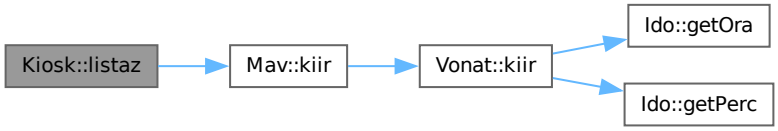
```
void Kiosk::listaz (
            Mav & mav )
```

Liszázza a vonatokat amiket beolvasott a fájlból.

Definition at line 30 of file kiosk.cpp.

```
00030                           {
00031   mav.kiir();
00032 }
```

Here is the call graph for this function:



Here is the caller graph for this function:

4.5.2.4 userInput()

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

Felhasználótól bemenetet kér.

Returns

menuItem fent említett enum class elemt ad vissza

Definition at line 22 of file kiosk.cpp.

```
00022                               {
00023     int valasz;
00024     std::cout « "Választott lehetőség: ";
00025     std::cin » valasz;
00026     return static_cast<menuItem>(valasz);
00027 }
```

Here is the caller graph for this function:

4.5.2.5 vonatHozza()

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

Hozzá ad a user vonatokat a fájlhoz.

Definition at line 35 of file kiosk.cpp.

```
00035                                                   {
00036   Seged s;
00037
00038   std::cin » s.szam;
00039   // if(*(s.szam) == '\n') std::cout « "FASZ";
00040   std::cin » s.indulo;
00041   std::cin » s.veg;
00042   std::cin » s.kocsidb;
00043   std::cin » s.indulas;
00044   std::cin » s.erkezes;
00045
00046   mav.addTrain(currPos, atoi(s.szam), Allomas(s.indulo), Allomas(s.veg),
00047                    atoi(s.kocsidb), Ido(s.indulas), Ido(s.erkezes));
00048 } // End of vonatHozza
```

Here is the call graph for this function:

Here is the caller graph for this function:

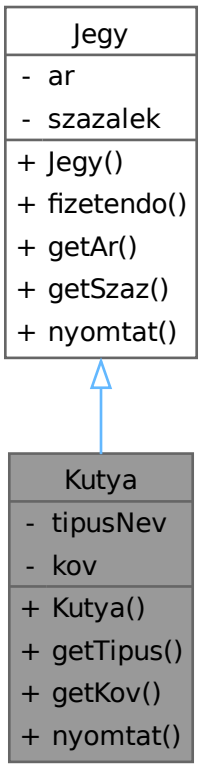
The documentation for this class was generated from the following files:

- kiosk.h
- kiosk.cpp

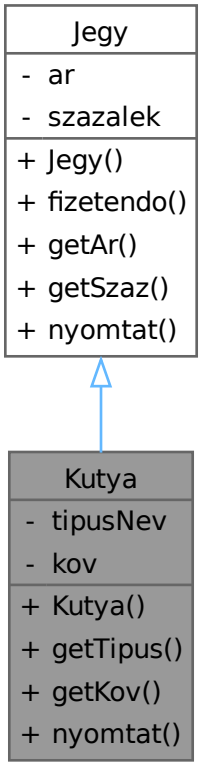
4.6 Kutya Class Reference

`#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]
```

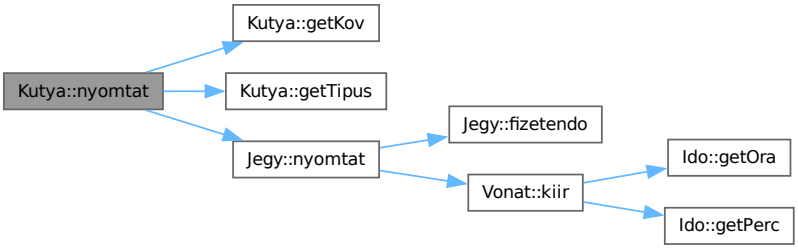
megörökölt nyomtat fv overwrite-ja meghívja az "eredei", főosztály nyomtatját

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.

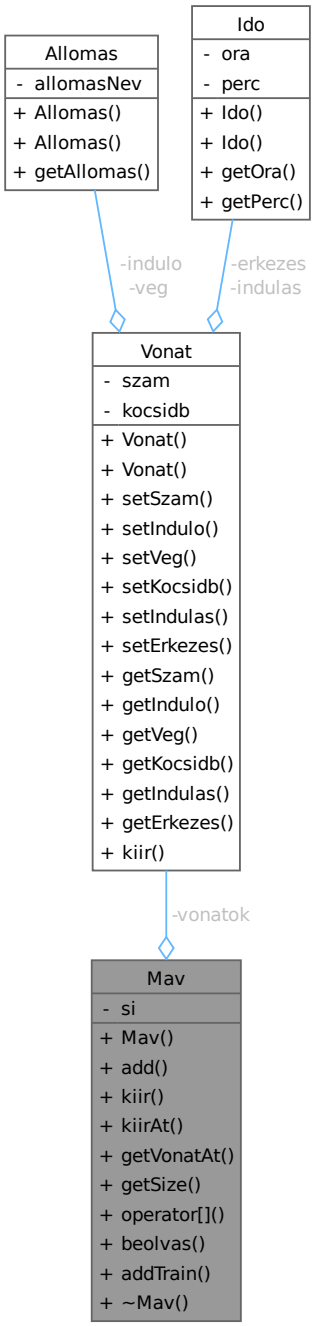
The documentation for this class was generated from the following file:

- 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 ()
- Vonat & operator[ ] (int idx)
- void beolvas ()
- void addTrain (std::streampos &currPos, int szam, Allomas indulo, Allomas veg, int kocsidb, Ido indulas, Ido erkezes)
- ~Mav ()

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]
```

Definition at line 70 of file mav.h.

```
00070         {
00071     delete[] vonatok;
00072   }
```

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
```

4.7.3.2 addTrain()

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

Definition at line 45 of file mav.cpp.

```
00046                                                                    {
00047   // Inicalizálás
00048   const char* vonatok = "./vonatok.txt";
00049   std::ofstream file (vonatok, std::ios::app); // Hozzáfűzésesen nyitom meg.
00050
00051   // HIBAKEZELÉS IDE
00052   if (!file.is_open()) {
00053     std::cout « "Megynitással van baj " « vonatok « std::endl;
00054     return;
00055   }
00056
00057   if(file.is_open()){
00058     file.seekp(currPos); // Megfelelő helyre ugrok
00059
00060     file « szam « " " « indulo « " " « veg « " " « kocsidb « " "
00061          « indulas « " " « erkezes « std::endl;
00062   }
00063   file.close();
00064 } // END OF addTrain
```

Here is the caller graph for this function:

4.7.3.3 beolvas()

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

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

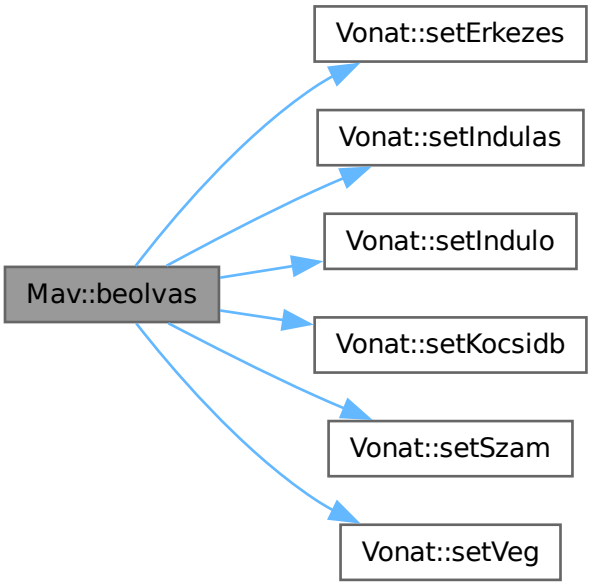
Parameters

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:

4.7.3.4 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:

4.7.3.5 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:

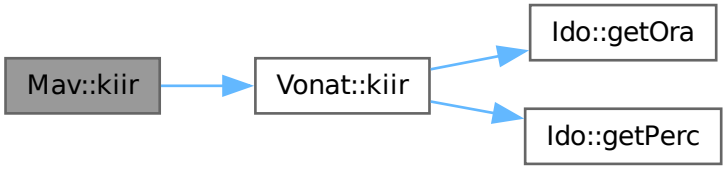
4.7.3.6 kiir()

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

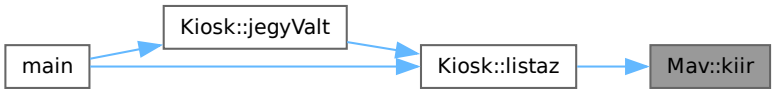
Definition at line 41 of file mav.h.

```
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:



4.7.3.7 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:

4.7.3.8 operator[]()

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

Definition at line 59 of file mav.h.

```
00059 { 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.

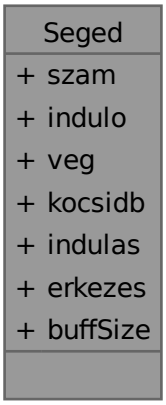
The documentation for this class was generated from the following files:

- 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

Definition at line 20 of file seged.h.

4.8.2 Field Documentation

4.8.2.1 buffSize

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

Definition at line 21 of file seged.h.

4.8.2.2 erkezes

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

Definition at line 27 of file seged.h.

4.8.2.3 indulas

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

Definition at line 26 of file seged.h.

4.8.2.4 indulo

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

Definition at line 23 of file seged.h.

4.8.2.5 kocsidb

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

Definition at line 25 of file seged.h.

4.8.2.6 szam

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

Definition at line 22 of file seged.h.

4.8.2.7 veg

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

Definition at line 24 of file seged.h.

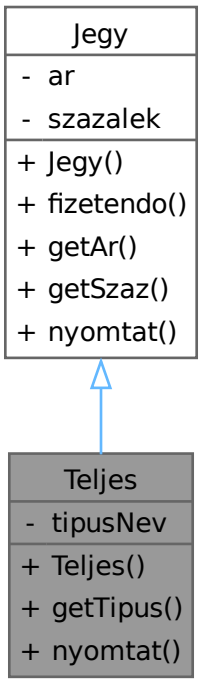
The documentation for this struct was generated from the following file:

- seged.h

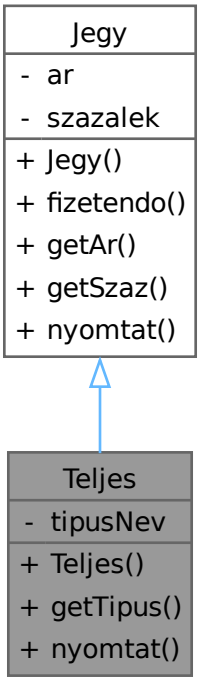
4.9 Teljes Class Reference

`#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]
```

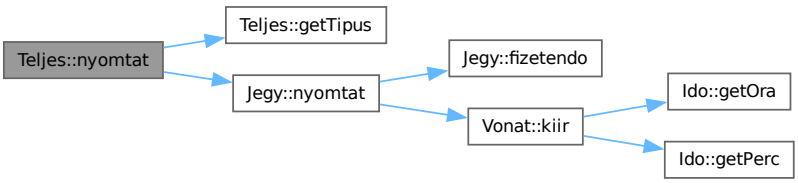
megörökölt nyomtat fv overwrite-ja meghívja az "eredei", főosztály nyomtatját

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.

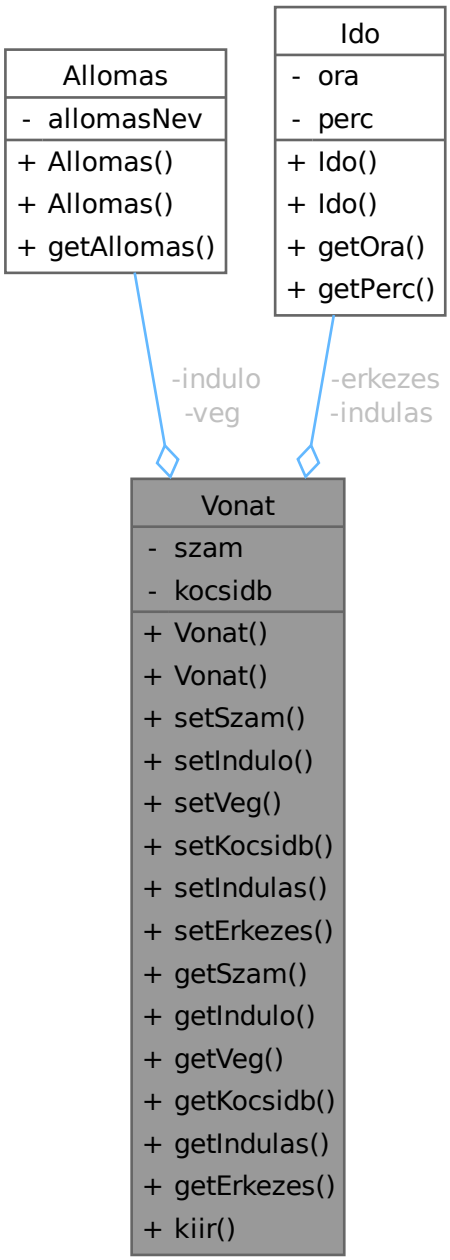
The documentation for this class was generated from the following file:

- 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; }
```

4.10.3.4 getKocsidb()

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

Definition at line 44 of file vonat.h.

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

4.10.3.5 getSzam()

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

Getterek.

Definition at line 41 of file vonat.h.

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

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()

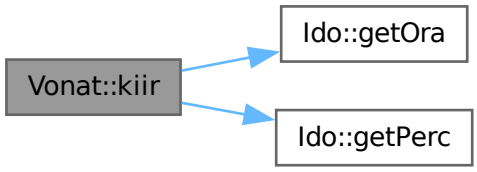
```
void Vonat::kiir ( ) const  [inline]
```

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

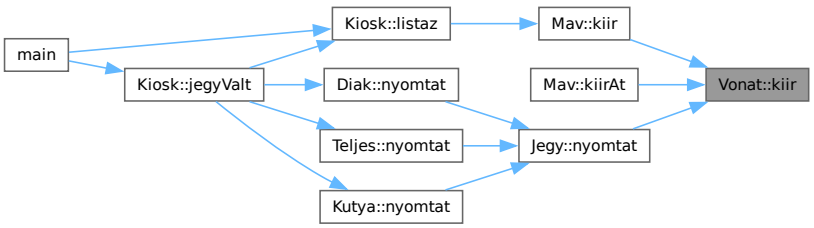
Definition at line 52 of file vonat.h.

```
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     std::cout « "Indulas idopontja: " « std::setw(2) « std::setfill('0')
00058               «indulas.getOra() « ":" « indulas.getPerc() « std::endl; // Nagyon ronda, tudom...
00059     std::cout « "Erkezes idopontja: " « std::setw(2) « std::setfill('0')
00060               «erkezes.getOra() « ":" « erkezes.getPerc() « std::endl;
00061   }
```

Here is the call graph for this function:



Here is the caller graph for this function:



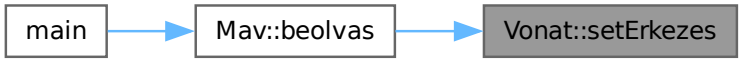
4.10.3.8 setErkezes()

```
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:

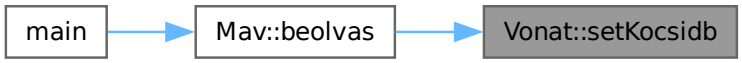
4.10.3.11 setKocsidb()

```
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:

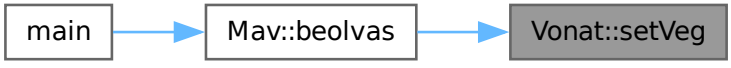
4.10.3.13 setVeg()

```
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

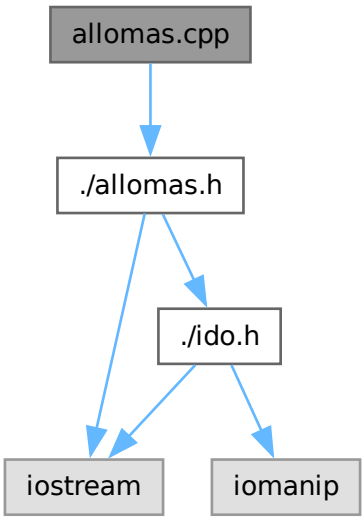
Chapter 5

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

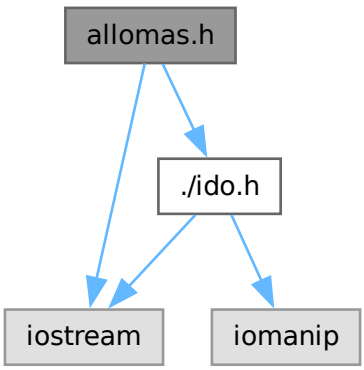
Go to the documentation of this file.

```
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();
      }
```

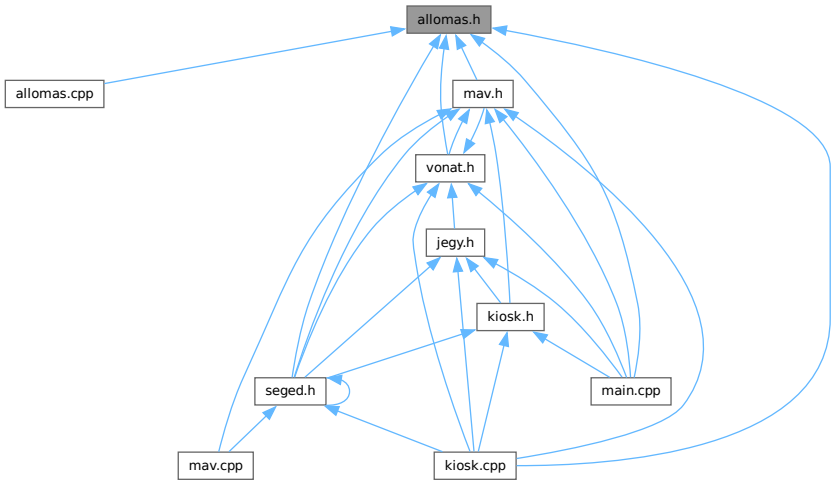
5.3 allomas.h File Reference

```
#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

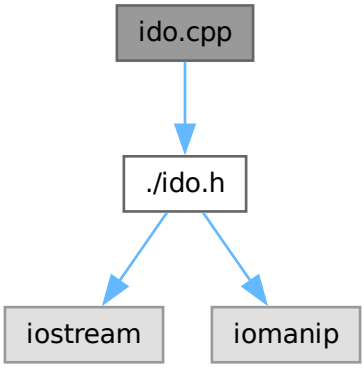
Go to the documentation of this file.

```
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<<()

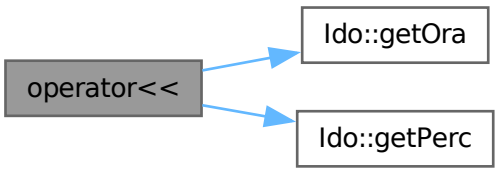
```
std::ostream & operator<< (
            std::ostream & os,
            const Ido & ido )
```

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

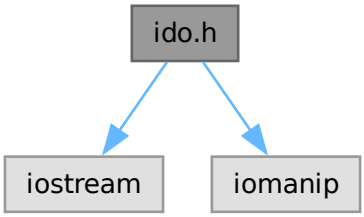
Go to the documentation of this file.

```
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«()
```

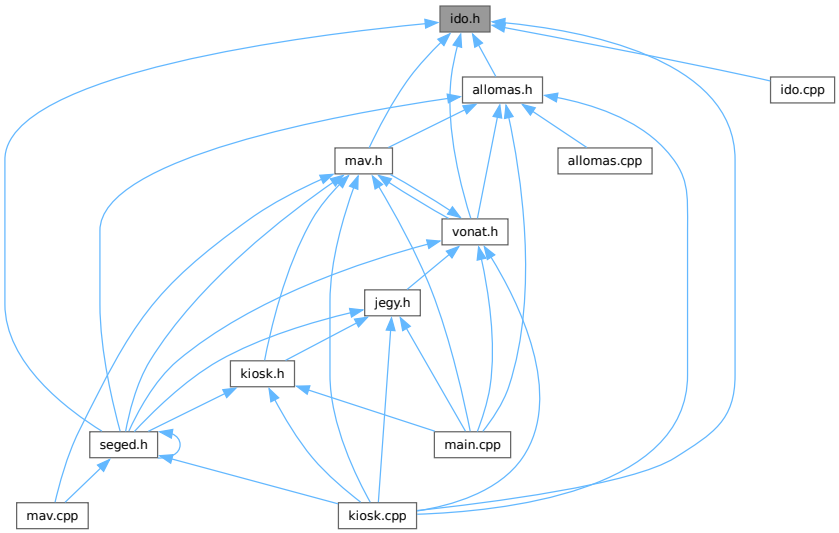
5.7 ido.h File Reference

```
#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<<()

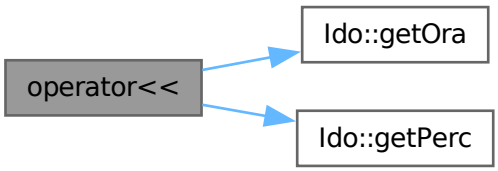
```
std::ostream & operator<< (
            std::ostream & os,
            const Ido & ido )
```

Ido class op<<() 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:



5.8 ido.h

Go to the documentation of this file.

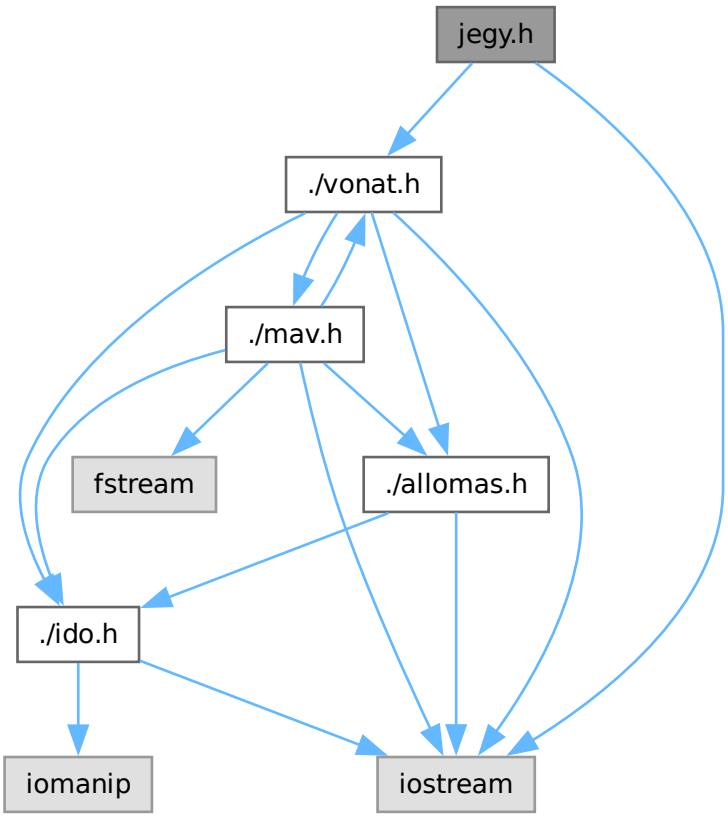
```
00001 #ifndef IDO_H
00002 #define IDO_H
00003
00004 /*
00005 * FILE IDO_H
00006 */
00007
00008 #include <iostream>
00009 #include <iomanip>
00010
00011
00012 class Ido{
00013   int ora;
00014   int perc;
00015 public:
00016   // Ido(int ora = 0, int perc = 0) : ora(ora), perc(perc) {}
00017   /**
00018   * Konstruktor
00019   * @param ido mivel beolvasáskor char* típusokba olvasok bele
00020   * char*-ot kap paraméterként és azt kezeli megfelelően konvertálva.
00021   */
00022   Ido(const char* ido) {
00023     ora = (ido[0] - '0') * 10 + (ido[1] - '0');
00024     perc = (ido[2] - '0') * 10 + (ido[3] - '0');
00025   }
00026
00027   Ido(int ora, int perc) : ora(ora), perc(perc) {}
00028
00029   // Setter
00030   // void setIdo(char* ido){
00031   //   ora = (ido[0] - '0') * 10 + (ido[1] - '0');
```

```
00032   //    perc = (ido[2] - '0') * 10 + (ido[3] - '0');
00033   // }
00034
00035   // Gettere
00036   int getOra() const { return ora; };
00037   int getPerc() const {return perc; };
00038
00039
00040
00041 }; // END OF IDO
00042
00043 /// Ido class op«() overwrireja, a fájlba beíráskor hívódik meg. (addTrain)
00044 std::ostream& operator«(std::ostream& os, const Ido& ido);
00045
00046
00047
00048 #endif // !IDO_H
```

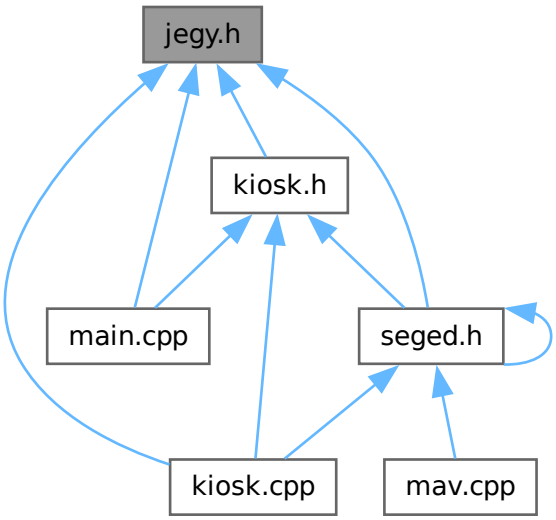
5.9 jegy.h File Reference

```
#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; }
```

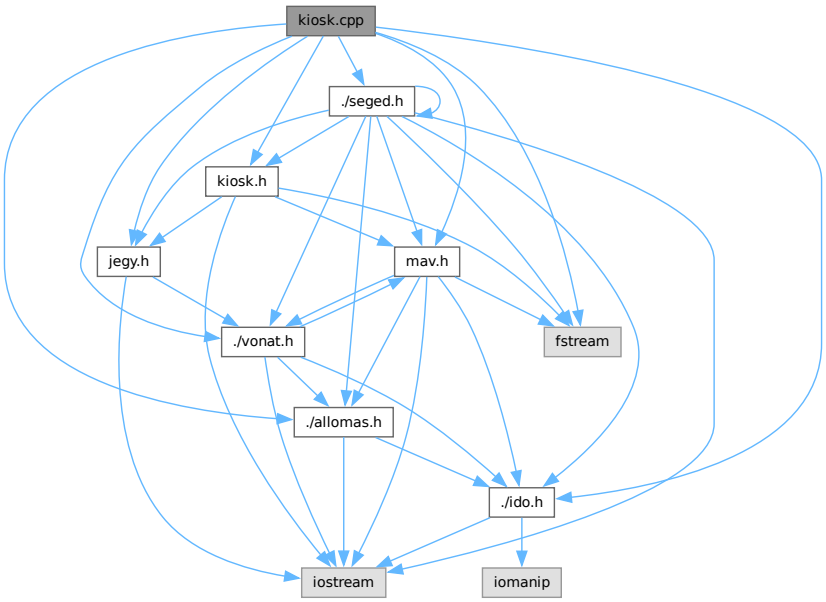
```
00028   double getSzaz() { return szazalek; }
00029
00030   /**
00031   * @param Vonat& megadott vonatra vonatkozik
00032   * Kiírja a jegy adatinak egy felét. Azt a felét ami minden leszármazottál
00033   * azonos. A felső pár so.
00034   * virtual mivel a leszármazottak megöröklik és felülírják.
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 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"
```

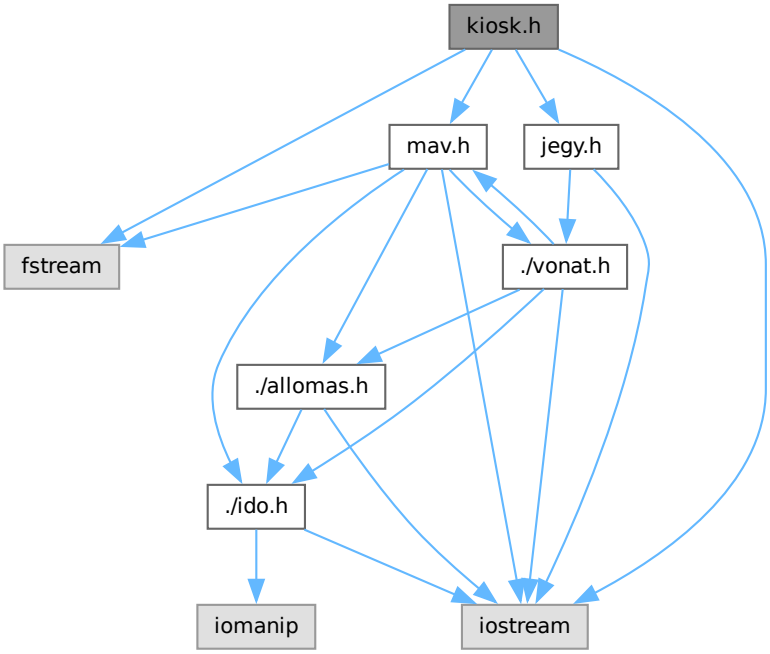
```
00004 #include "./seged.h"
00005
00006 #include "./mav.h"
00007 #include "./vonat.h"
00008 #include "allomas.h"
00009 #include "ido.h"
00010
00011 #include <fstream>
00012
00013 void Kiosk::init() {
00014   std::cout « "### Dönts ###" « std::endl;
00015   std::cout « "1. Vonat hozzáadása "     « std::endl;
00016   std::cout « "2. Vonatok listázása "    « std::endl;
00017   std::cout « "3. Jegy nyomtatása"       « std::endl;
00018   std::cout « "4. Kilépés"               « std::endl;
00019 }
00020
00021 // User input az input streamről enum-má konvertálva
00022 menuItem Kiosk::userInput() {
00023    int valasz;
00024    std::cout « "Választott lehetőség: ";
00025    std::cin » valasz;
00026    return static_cast<menuItem>(valasz);
00027 }
00028
00029 // Liszázza az összes vonatot
00030 void Kiosk::listaz(Mav& mav){
00031   mav.kiir();
00032 }
00033
00034
00035 void Kiosk::vonatHozza(Mav& mav, std::streampos currPos){
00036   Seged s;
00037
00038   std::cin » s.szam;
00039   // if(*(s.szam) == '\n') std::cout « "FASZ";
00040   std::cin » s.indulo;
00041   std::cin » s.veg;
00042   std::cin » s.kocsidb;
00043   std::cin » s.indulas;
00044   std::cin » s.erkezes;
00045
00046   mav.addTrain(currPos, atoi(s.szam), Allomas(s.indulo), Allomas(s.veg),
00047                     atoi(s.kocsidb), Ido(s.indulas), Ido(s.erkezes));
00048 } // End of vonatHozza
00049
00050
00051
00052 // Jegy vásárlás
00053 void Kiosk::jegyValt(Mav& mav){
00054   int idx, buf;
00055   listaz(mav);
00056   std::cout « "Valasz vonatot: ";
00057   std::cin » idx;
00058   // Hibakezelés
00059   if(size_t(idx) > mav.getSize()){ std::cout « "Túlindexelés\n"; return; }
00060   std::cout « std::endl;
00061
00062   std::cout « "1.Teljes, 2.Diak, 3.Kutya\n ";
00063   std::cin » buf;
00064   // Hibakezelés
00065   if(buf > 3){ std::cout « "Túlindexelés"; return; }
00066   std::cout « std::endl;
00067
00068   switch (buf) {
00069     case 1: {
00070       Teljes t;
00071       t.nyomtat(mav.getVonatAt(idx));
00072       break;
00073     }
00074
00075     case 2: {
00076       Diak d;
00077       d.nyomtat(mav.getVonatAt(idx));
00078       break;
00079     }
00080
00081     case 3: {
00082       Kutya k;
00083       k.nyomtat(mav.getVonatAt(idx));
00084       break;
00085     }
00086
00087   } // end of switch
00088
00089 } // end of jegyValt
00090
```

```
00091
00092
00093
00094
00095
00096
```

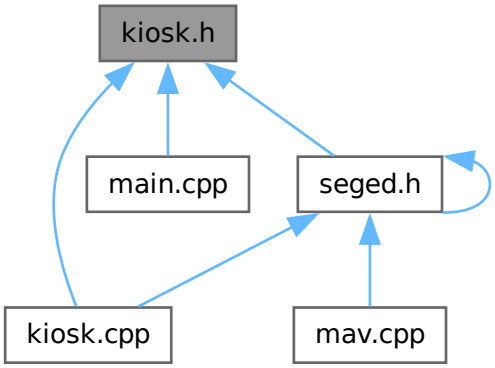
5.13 kiosk.h File Reference

```
#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

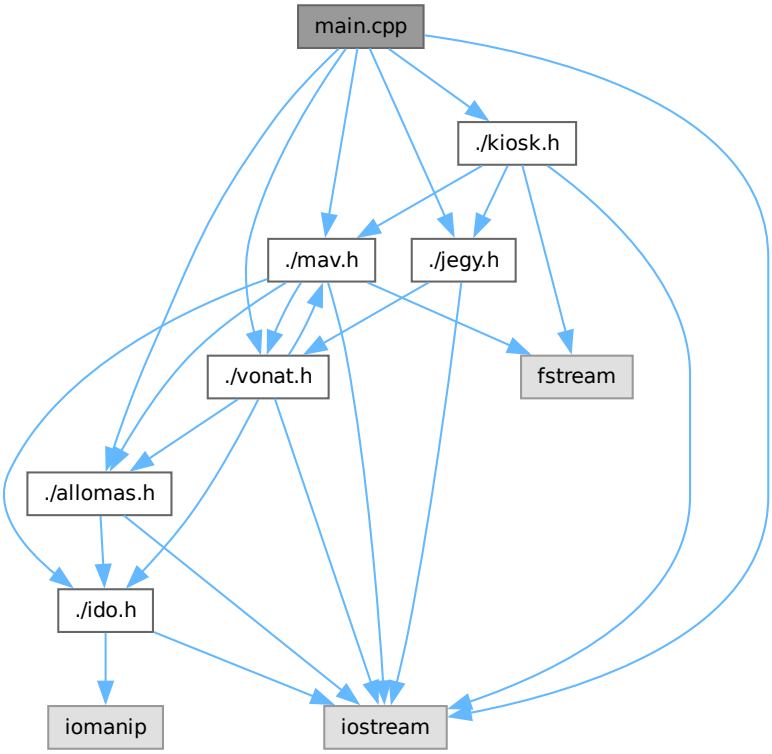
Go to the documentation of this file.

```
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   /// Inicializálja a menüt, ergo kiírja a lehetőségeket
00027   void init();
00028   /// Felhasználótól bemenetet kér.
00029   /// @return menuItem fent említett enum class elemt ad vissza
00030   menuItem userInput();
00031
00032   /// Liszázza a vonatokat amiket beolvasott a fájlból
00033   void listaz(Mav& mav);
00034   /// Hozzá ad a user vonatokat a fájlhoz
00035   void vonatHozza(Mav& mav, std::streampos currPos);
00036   /// jegyet vált a kiválasztott vonatra
00037   void jegyValt(Mav& mav);
00038
00039 }; // end of Kiosk
00040
00041
00042
00043
00044
00045
00046 #endif // !KIOSK_H
```

5.15 main.cpp File Reference

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

Include dependency graph for main.cpp:



Functions

- int main ()

5.15.1 Function Documentation

5.15.1.1 main()

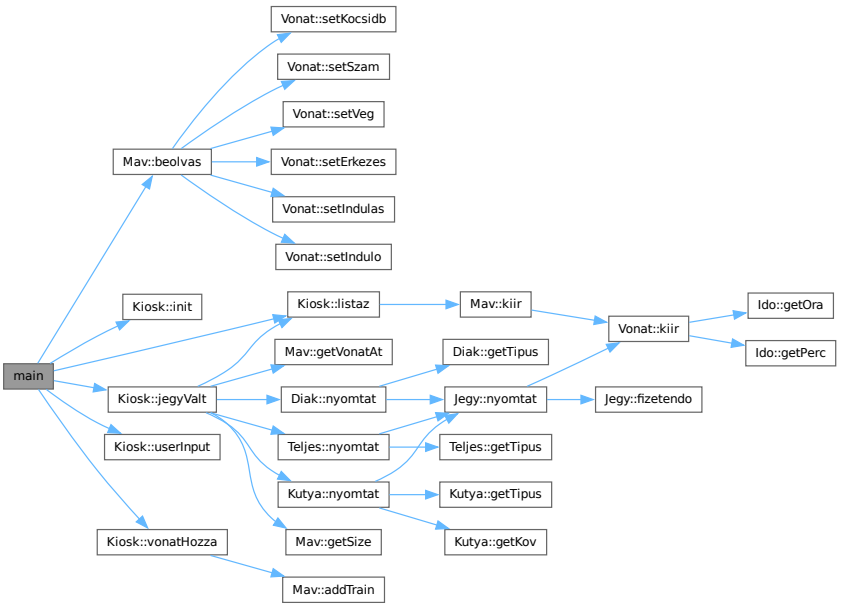
```
int main ( )
```

Definition at line 10 of file main.cpp.

```
00010         {
00011   // Itt tárolon hol vagyok a file-ban
00012   // Mind a beolvasás mind pedig az íráshoz KELL!
00013
00014   /* FÁJL BEOLVASÁS */
00015   std::streampos currPos;
00016   Mav mav;
00017
00018   mav.beolvas();
00019
00020   /* KIOSK TEST */
00021   Kiosk k;
00022   k.init(); // Kiírja a lehetőségeket
00023
00024
00025 int choice = 0;
00026   while (choice != 4) {
```

```
00027     choice = k.userInput(); // Get user input once per loop iteration
00028     switch (choice) {
00029       // Vonat hozzáadása
00030       case 1: {
00031         std::cout « "Beolvas: " « std::endl;
00032         std::cout « "Szám Indulo Vég kocsidb Indulás erkezes" « std::endl;
00033         k.vonatHozza(mav, currPos);
00034         // system("clear"); // Tábla törlés
00035         std::cout « "\n" « "Vonat hozzáadva\n\n";
00036         mav.beolvas();
00037         k.init(); // Kiírja a lehetőségeket
00038         break;
00039       }
00040       // Vonatok listázása
00041       case 2: {
00042         std::cout « std::endl;
00043         mav.beolvas();
00044         k.listaz(mav);
00045         std::cout « std::endl;
00046         k.init(); // Kiírja a lehetőségeket
00047         break;
00048       }
00049       // Jegy vásárlás
00050       case 3: {
00051         std::cout « std::endl;
00052         k.jegyValt(mav);
00053         std::cout « std::endl;
00054         k.init(); // Kiírja a lehetőségeket
00055         break;
00056       }
00057       // Kilépés
00058       case 4: {
00059         std::cout « "Kilépés" « std::endl;
00060         break;
00061       }
00062       default: {
00063         std::cout « "Érvénytelen lehetőség" « std::endl;
00064       }
00065     } // end of switch
00066   } // end of while
00067
00068   return 0;
00069 } // end of main
```

Here is the call graph for this function:



5.16 main.cpp

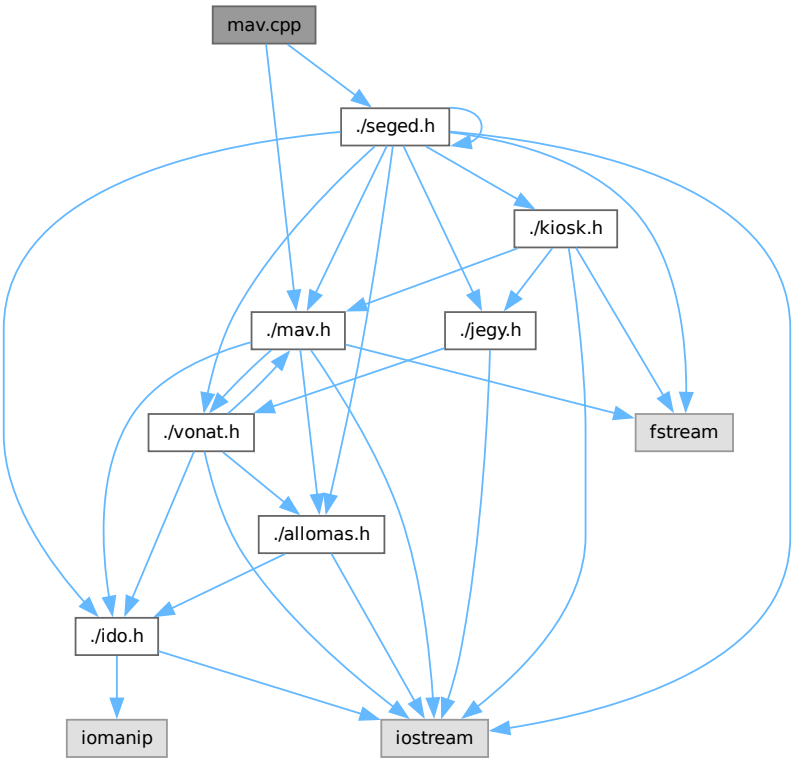
Go to the documentation of this file.

```
00001 #include <iostream>
00002
00003 #include "./mav.h"
00004 #include "./vonat.h"
00005 #include "./allomas.h"
00006 #include "./jegy.h"
00007 #include "./kiosk.h"
00008
00009
00010 int main(){
00011   // Itt tárolon hol vagyok a file-ban
00012   // Mind a beolvasás mind pedig az íráshoz KELL!
00013
00014   /* FÁJL BEOLVASÁS */
00015   std::streampos currPos;
00016   Mav mav;
00017
00018   mav.beolvas();
00019
00020   /* KIOSK TEST */
00021   Kiosk k;
00022   k.init(); // Kiírja a lehetőségeket
00023
00024
00025 int choice = 0;
00026   while (choice != 4) {
00027     choice = k.userInput(); // Get user input once per loop iteration
00028     switch (choice) {
00029       // Vonat hozzáadása
00030       case 1: {
00031         std::cout « "Beolvas: " « std::endl;
00032         std::cout « "Szám Indulo Vég kocsidb Indulás erkezes" « std::endl;
00033         k.vonatHozza(mav, currPos);
00034         // system("clear"); // Tábla törlés
00035         std::cout « "\n" « "Vonat hozzáadva\n\n";
00036         mav.beolvas();
00037         k.init(); // Kiírja a lehetőségeket
00038         break;
00039       }
00040       // Vonatok listázása
00041       case 2: {
00042         std::cout « std::endl;
00043         mav.beolvas();
00044         k.listaz(mav);
00045         std::cout « std::endl;
00046         k.init(); // Kiírja a lehetőségeket
00047         break;
00048       }
00049       // Jegy vásárlás
00050       case 3: {
00051         std::cout « std::endl;
00052         k.jegyValt(mav);
00053         std::cout « std::endl;
00054         k.init(); // Kiírja a lehetőségeket
00055         break;
00056       }
00057       // Kilépés
00058       case 4: {
00059         std::cout « "Kilépés" « std::endl;
00060         break;
00061       }
00062       default: {
00063         std::cout « "Érvénytelen lehetőség" « std::endl;
00064       }
00065     } // end of switch
00066   } // end of while
00067
00068   return 0;
00069 } // 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.

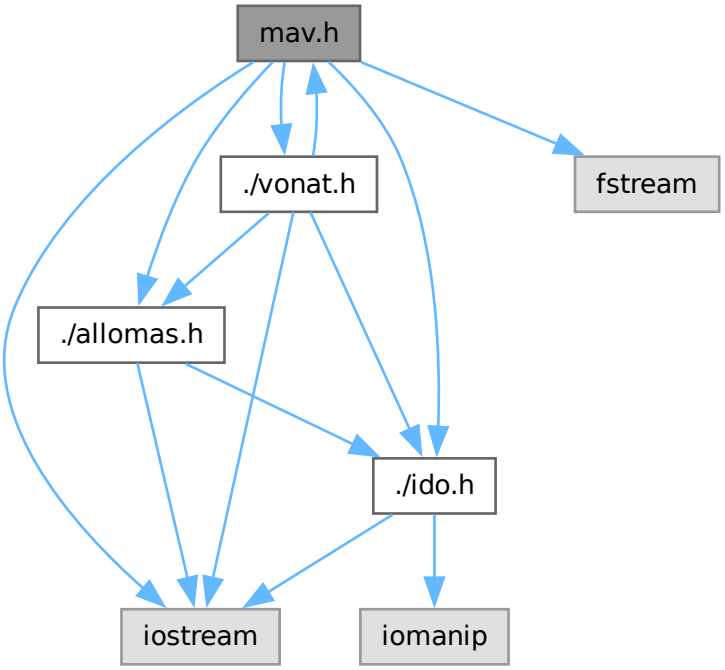
```
00001 #include "./mav.h"
00002 #include "./seged.h"
00003
00004
00005 /**
00006  * Beolvaásás a file-ból egy segéd strukturába, onnan pedig egy Vonat objektum feltöltés
00007  * @param Mav& mav class feltöltéséhez szükséges
00008  */
00009 void Mav::beolvas() {
00010   // Inicalizálás
00011   const 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  * Vonatok hozzáadása
00042  * @param currPos a streamben elfoglalt helyem
00043  * @param minden-más a vonat class feltöltéséhez szükséges adatok
00044 */
00045 void Mav::addTrain(std::streampos& currPos, int szam, Allomas indulo,
00046             Allomas veg, int kocsidb, Ido indulas, Ido erkezes){
00047   // Inicalizálás
00048   const char* vonatok = "./vonatok.txt";
00049   std::ofstream file (vonatok, std::ios::app); // Hozzáfűzésesen nyitom meg.
00050
00051   // HIBAKEZELÉS IDE
00052   if (!file.is_open()) {
00053     std::cout « "Megynitással van baj " « vonatok « std::endl;
00054     return;
00055   }
00056
00057   if(file.is_open()){
00058     file.seekp(currPos); // Megfelelő helyre ugrok
00059
00060     file « szam « " " « indulo « " " « veg « " " « kocsidb « " "
00061          « indulas « " " « erkezes « std::endl;
00062   }
00063   file.close();
00064 } // END OF addTrain
00065
00066
```

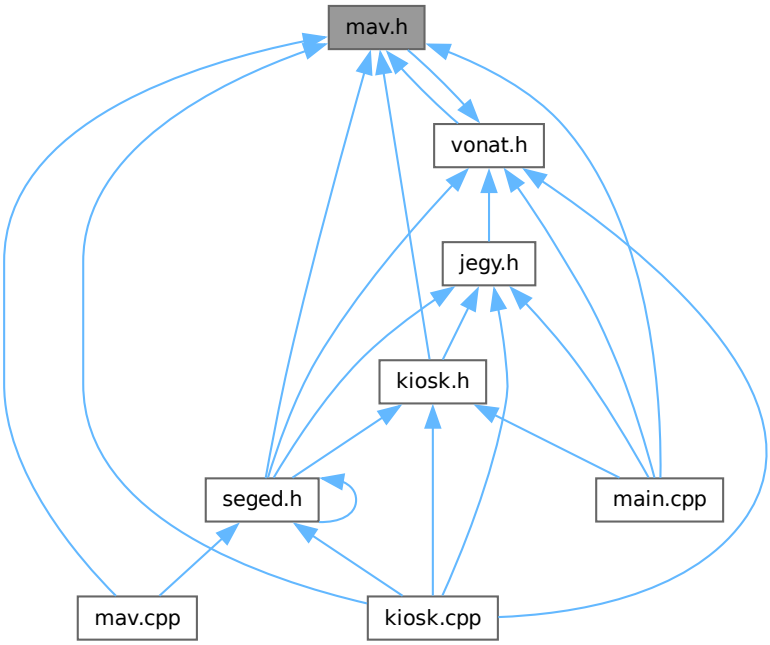
5.19 mav.h File Reference

```
#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.

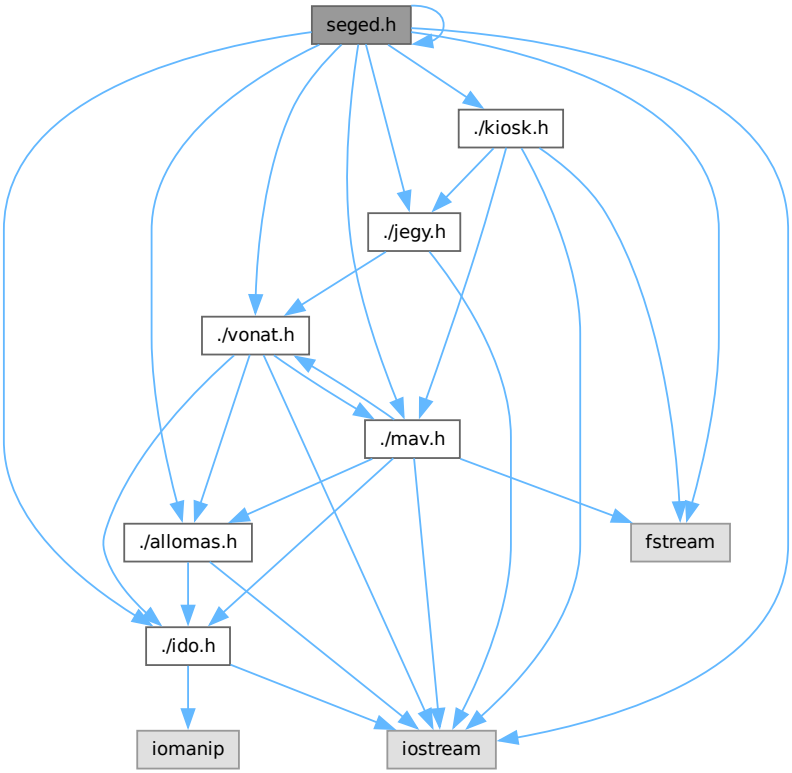
```
00001 #ifndef MAV_H
00002 #define MAV_H
00003
00004 /*
00005  * FILE MAV_H
00006  */
00007
00008 #include <iostream>
00009 #include <fstream>
00010
00011 #include "./vonat.h"
00012 #include "./allomas.h"
00013 #include "./ido.h"
00014
00015
00016 class Mav {
00017   Vonat* vonatok; // Vonatok kollekció
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   Vonat& operator[](int idx){ return vonatok[idx]; }
00060
00061   /*
00062   * @param fazs geci
00063   * @param fasz geic
00064   */
00065   void beolvas();
00066   void addTrain(std::streampos& currPos, int szam, Allomas indulo,
00067                 Allomas veg, int kocsidb, Ido indulas, Ido erkezes);
00068
00069   // Destruktor
00070   ~Mav(){
00071     delete[] vonatok;
00072   }
00073
00074
00075 }; // END OF MAV
00076
00077
00078
00079
00080 #endif // !MAV_H
00081
```

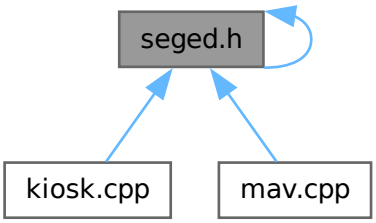
5.21 seged.h File Reference

```
#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

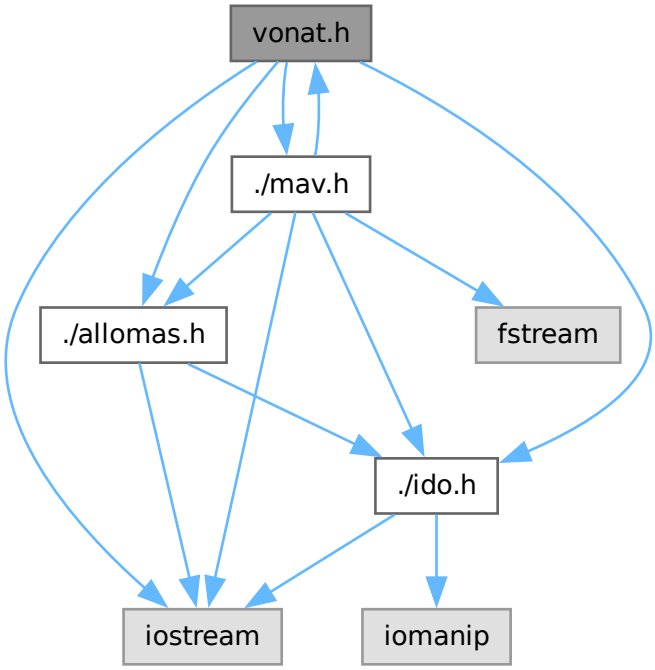
Go to the documentation of this file.

```
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 */
00020 struct Seged {
00021    static const int buffSize = 25;
00022    char szam[buffSize];
00023    char indulo[buffSize];
00024    char veg[buffSize];
00025    char kocsidb[buffSize];
00026    char indulas[buffSize];
00027    char erkezes[buffSize];
00028 };
00029
00030
00031
00032 #endif // !SEGED_H
00033
```

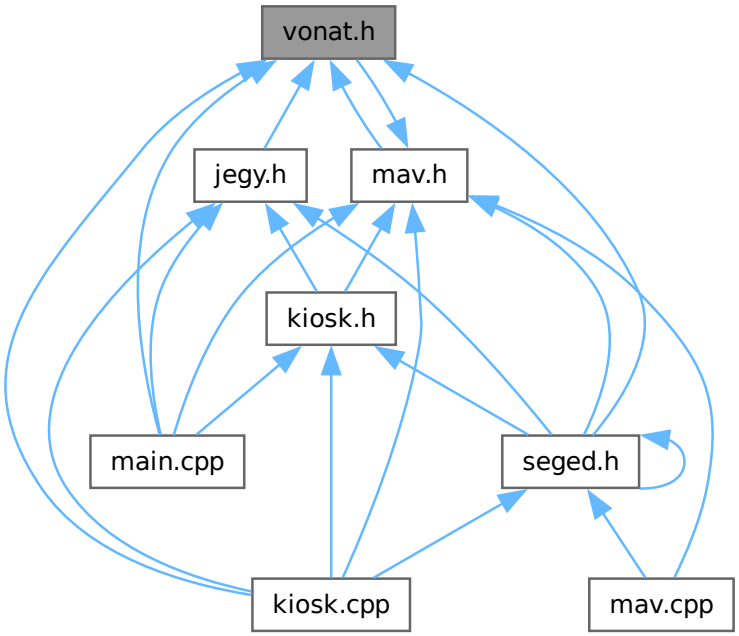
5.23 vonat.h File Reference

```
#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

5.24 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     std::cout « "Indulas idopontja: " « std::setw(2) « std::setfill('0')
00058               «indulas.getOra() « ":" « indulas.getPerc() « std::endl; // Nagyon ronda, tudom...
00059     std::cout « "Erkezes idopontja: " « std::setw(2) « std::setfill('0')
00060               «erkezes.getOra() « ":" « erkezes.getPerc() « std::endl;
00061   }
00062
00063 }; // end of VONAt
00064
00065
00066 #endif // !VONAT_H
```