Bajnok Vencel
X748Q2

TELEFONKÖZPONT SPECIFIKÁCIÓ

A VÁLASZTOTT FELADAT:

TELEFONKÖZPONT

Készítsen egyszerű telefonhálózatot szimuláló előfizető és központ osztályokat!
A központtól egy előfizető vagy egy másik központ kapcsolást kérhet. Az egyes előfizetőket 4 jegyű hívószámmal lehet elérni, melyből az eső jegy a körzetszám (1-9), a maradék 3 jegy pedig az előfizetői szám. Például az 1999 az 1-es körzetben a 999-es előfizető. Minden körzetben pontosan egy központ van. Ehhez kapcsolódnak a körzet előfizetői és az idegen körzetek központjai. Két központ között több kapcsolat is lehet. A központok külső kapcsolata a létrehozásakor megadott maximumot nem haladhatja meg. Hasonlóan a központok létrehozásakor adható meg, hogy az adott központ hány kapcsolási kérést tud egyszerre kiszolgálni (kapcsolási tábla mérete). Ha egy előfizető kapcsolást kér, meg kell próbálni a kapcsolatot fölépíteni. Ezt tárolni kell a korlátos méretű kapcsolótáblában. Távolsági hívásoknál minden érintett központban egy-egy kapcsolás keletkezik. Ha valamelyik központ nem tudja teljesíteni a kapcsolást, mert nincs szabad hely a kapcsolótáblában, nincs szabad távolsági vonal, vagy foglalt az előfizető, akkor kivételt dob, minek hatására törlődik az addig felépített kapcsolási sor. Ellenkező esetben a hívó egy kapcsolás-objektumot kap, amelyen keresztül üzenetet küldhet. A hívó megszakíthatja a hívást, ekkor törlődik a kapcsolás, és erről értesül a hívott fél is.
Demonstrálja a működést külön modulként fordított tesztprogrammal! A megoldáshoz felhasználhat STL tárolót is!

A PROGRAM CÉLJA:

Telefonhálózat modellezése, telefonközpontok és előfizetők kezelése, kapcsolása, kapcsolások kezelése.

A PROGRAM HASZNÁLATA:

A felhasználónak a menüből kell kiválasztania, hogy mit szeretne csinálni, a lehetőségei:

- Új központ létrehozása

  //A program ki fogja írni, hogy milyen adatokat kell megadnia a felhasználónak ebben és a következő menüpont esetében is.

- Új előfizető létrehozása
- Hívás – új kapcsolat létrehozása / kapcsolat megszűntetése

  //2 számot kell megadnia, előbb a sajátját, majd azét, akit hív. A program tájékoztatja majd a felhasználót, hogy sikerült-e létrehoznia a kapcsolatot (a központ/-ok tudják-e kapcsolni). Ha már létre lett hozva korábban a kapcsolat, akkor azt írja ki. A kapcsolat során üzenetküldésre és a kapcsolat befejezésére van lehetősége a felhasználónak. A hívás befejezése (= a kapcsolat megszűntetése) esetén a hívott fél kap üzenetet a megszakításról.

- Előfizetői bejelentkezés - üzenetek lekérdezése

  //Itt a saját telefonszámát kell megadnia az előfizetőnek.

- Adott központ kapcsolatainak lekérdezése
- Adatok importálása .txt file-okból
- A központok és a személyek exportálása *.txt file-ba

Az adatimportálás során az előfizetőket és a központokat külön .txt file-ban kell megadni, illetve ekkor még nincsenek kapcsolatok köztük. Nem lehet olyan személyt importálni, akinek a telefonszáma olyan központhoz tartozik, ami még nem létezik. Az adatok sorrendje, formátuma a következő: (példa)

központok:

1;12;3	//<körzet száma>;<maximum belső kapcsolatszám>;<maximum távolsági kapcsolatszám>
3;24;1	
4;4;30	

előfizetők:

Pél Dániel	//név
3758	//hívószám
45	//üzenet tárhely (ennyi darab üzenetet tud fogadni)

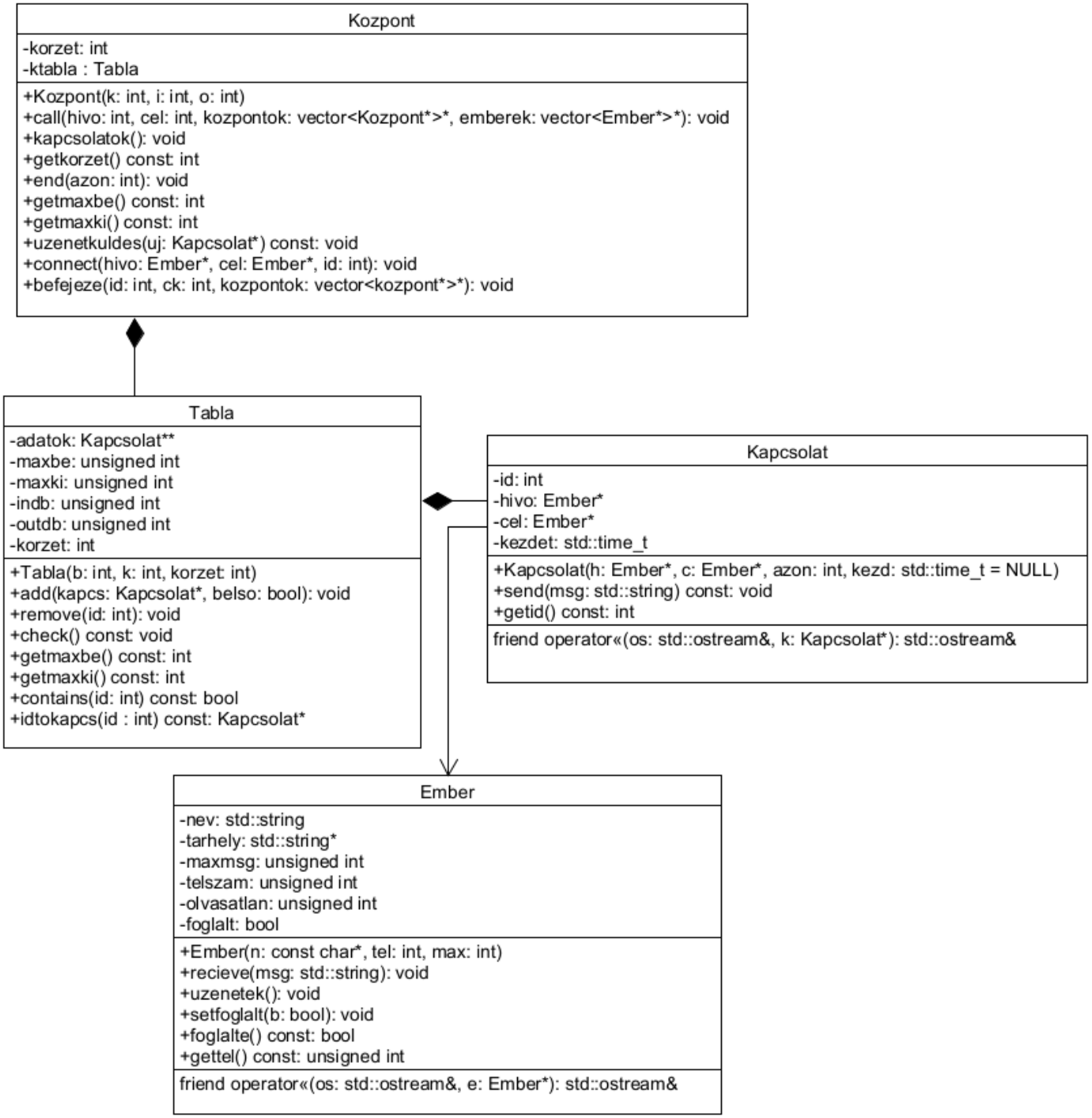
Az üzenetek megtekintése során az ügyfélnek lehetősége van törölni az adott üzenetet, illetve az eltárolt üzenetek bármelyikét megtekinteni.

A PROGRAM MŰKÖDÉSE:

A programnak el kell tudnia tárolni az ügyfelek és a központok adatait, továbbá a fennálló kapcsolatokat objektumokként. Az ügyfélnek/előfizetőnek lehetősége kell legyen üzenetet küldeni a létrehozott kapcsolat objektumon keresztül. Továbbá az ügyfél véges darabszámú üzeneteit is el kell tudnia tárolni. A működés helyessége parancssoros batch alkalmazásként (is) ellenőrizhető legyen, azaz a tesztprogram ne tételezzen fel semmilyen speciális be/kiviteli eszközt, a szabványos be/kimenetet ill. a hibakimenetet úgy kezelje, hogy az átirányítható legyen fájlba.

MEGVALÓSÍTÁSI TERV:

A tervezett osztályok UML diagrammja jobb oldalt látható.



Külön modul lesz a teszteléshez, mindegyik osztálynak egyesével, lesz egy main modul, ami vezérli a programot és egy hozzá tartozó vezérlő modul.

A központokat és az embereket vektorokban fogom tárolni, dinamikusan, a kapcsolatokat pedig tömbökben.

A program nem tartalmaz memóriaszivárgást, ezt a gtest-lite.h-val és a memtrace.h-val ellenőriztem.

TESZTELÉS DOKUMENTÁCIÓJA:

Ha tesztelni szeretnénk a programot, akkor a main.cpp-ben és a kozpont.h-ban definiálnunk kell a TESZT makrót.

Ezután a *bool test_main()* függvényben fog meghívódni szinte az összes metódus, a JPORTA-n lefuttatott tesztesetekhez hasonlóan, itt egy kicsit nagyobb lefedettséggel, mivel a JPORTA-n nem írattam file-t.

```
Kozpont* k1 = new Kozpont(1, 10, 10); …     // létrehozunk pár példa központot és embert
Ember* e1 = new Ember("Jani", 1023, 5); …

vector<Kozpont*> kptok = { k1, k2 };          // a központok és az emberek tárolása, mint a main()-ben
vector<Ember*> mbrek = { e1, e2, e3 };

dimport(&kptok, &mbrek, "kozpontok.txt", "emberek.txt");    // file-okból is beolvasunk pár központot és embert

k1->call(1023, 1111, &kptok, &mbrek);
// Felhívjuk a 1023-as számmal a 1111-est. A tesztelő ne fejezze be a kapcsolatot, automatikusan fog küldeni egy üzenetet!

k1->call(1023, 1111, &kptok, &mbrek);
// Kipróbáljuk a már létező kapcsolathoz való csatlakozást, fejezze be a kapcsolatot! A hívott félnek be fog telni a tárhelye.

k1->call(1023, 2000, &kptok, &mbrek);
// Újabb hívás, még ne zárjuk be, hogy legyen nyitott kapcsolat az 1-es központban!

k1->kapcsolatok();          // Kiírja a k1  központban tárolt 1 darab kapcsolat és a központ adatait.

k1->call(1023, 2000, &kptok, &mbrek);       // Újra csatlakozunk az előbbi kapcsolatunkhoz.

e1->uzenetek(); …           // Itt megnézzük a 3 példa ember üzeneteit, lehet törölni is akár őket.}

dexport(&kptok, &mbrek);            //Exportáljuk az összes központot és embert egy-egy .txt file-ba

for (size_t i = 0; i < kptok.size();i++){ delete kptok[i]; }
// Felszabadítjuk a dinamikusan lefoglalt központokat, és ugyanígy az embereket is ezután, ugyanez megtörténne a main() függvényben is

return true;
//Igazzal térünk vissza, ez esetben nem dobott semmilyen kezeletlen hibából fakadó kivételt a program és sikeres volt a teszt

catch (...){return false;}
// Ha másmilyen kivételt dobna a tesztelés, mint amilyet vártunk, akkor azt itt kapjuk el. Egy hamis visszatéréssel jelezzük, hogy hiba van valószínűleg a programban.
```

A tesztelés során az összes függvényt egy-egy *try{…}catch(){…}* blokkban hívtuk meg így figyeltünk az esetleges dobott kivételekre.

A standard inputról való ember és központ hozzáadását nem teszteltük csak a programban, de azt a JPORTA-n igen.

Az, hogy a teszt mennyi osztályt és tagfüggvényt érint, az az alábbi dokumentációban a *teszt_main()* függvény hívási gráfjában is látszódik.

TELEFONKÖZPONT DOKUMENTÁCIÓ

1. Osztálymutató **1**

1.1. Osztálylista . . . 1

2. Fájlmutató **2**

2.1. Fájllista . . . 2

3. Osztályok dokumentációja **2**

3.1. cp struktúrareferencia . . . 2

3.1.1. Részletes leírás . . . 3

3.1.2. Tagfüggvények dokumentációja . . . 3

3.2. Ember osztályreferencia . . . 4

3.2.1. Részletes leírás . . . 5

3.2.2. Konstruktorok és destruktorok dokumentációja . . . 5

3.2.3. Tagfüggvények dokumentációja . . . 6

3.2.4. Barát és kapcsolódó függvények dokumentációja . . . 7

3.2.5. Adattagok dokumentációja . . . 7

3.3. Kapcsolat osztályreferencia . . . 8

3.3.1. Részletes leírás . . . 10

3.3.2. Konstruktorok és destruktorok dokumentációja . . . 10

3.3.3. Tagfüggvények dokumentációja . . . 11

3.3.4. Barát és kapcsolódó függvények dokumentációja . . . 12

3.3.5. Adattagok dokumentációja . . . 12

3.4. Kozpont osztályreferencia . . . 13

3.4.1. Részletes leírás . . . 15

3.4.2. Konstruktorok és destruktorok dokumentációja . . . 15

3.4.3. Tagfüggvények dokumentációja . . . 16

3.4.4. Adattagok dokumentációja . . . 20

3.5. Tabla osztályreferencia . . . 20

3.5.1. Részletes leírás . . . 22

3.5.2. Konstruktorok és destruktorok dokumentációja . . . 22

3.5.3. Tagfüggvények dokumentációja . . . 23

3.5.4. Adattagok dokumentációja . . . 25

4. Fájlok dokumentációja **26**

4.1. telkozp/control.hpp fájlreferencia . . . 26

4.1.1. Részletes leírás . . . 27

4.1.2. Függvények dokumentációja . . . 28

4.1.3. Változók dokumentációja . . . 31

4.2. telkozp/ember.cpp fájlreferencia . . . 31

4.2.1. Részletes leírás . . . 31

4.3. telkozp/ember.h fájlreferencia . . . 32

4.3.1. Részletes leírás . . . 33

4.4. telkozp/kapcsolat.cpp fájlreferencia . . . 33

4.4.1. Részletes leírás . . . 33
4.5. telkozp/kapcsolat.h fájlreferencia . . . 33
4.5.1. Részletes leírás . . . 35
4.6. telkozp/kapcstabla.cpp fájlreferencia . . . 35
4.6.1. Részletes leírás . . . 35
4.7. telkozp/kapcstabla.h fájlreferencia . . . 36
4.7.1. Részletes leírás . . . 37
4.8. telkozp/kozpont.cpp fájlreferencia . . . 37
4.8.1. Részletes leírás . . . 37
4.9. telkozp/kozpont.h fájlreferencia . . . 38
4.9.1. Részletes leírás . . . 39
4.10. telkozp/test.cpp fájlreferencia . . . 39
4.10.1. Részletes leírás . . . 40
4.10.2. Függvények dokumentációja . . . 40
4.11. telkozp/test.h fájlreferencia . . . 41
4.11.1. Részletes leírás . . . 42
4.11.2. Függvények dokumentációja . . . 42

Tárgymutató **43**

1. Osztálymutató

1.1. Osztálylista

Az összes osztály, struktúra, unió és interfész listája rövid leírásokkal:

cp
Cross-platform getline **2**

Ember
Egy előfizető személy adatait tárolja: A nevét; az üzeneteit egy véges tömbben, melyek közül számon tartja, hogy mennyi olvasatlan; a telefonszámát; és a foglaltsági státuszát **4**

Kapcsolat
Két emberre mutató pointert tárol, akik között fennáll a kapcsolat, továbbá az azonosítóját és egy időpillanatot, amikor létrejött. Lehet rajta keresztül a hívó személynek üzenetet küldeni a hívott részére **8**

Kozpont
Kapcsolatok tárolására és kezelésére szolgál. Tartalmaz egy Tabla objektumot, melyben a Kapcsolat objektumok tárolva vannak, továbbá a körzetszámát **13**

Tabla
Kapcsolat objektumokra mutató pointereket tárol egy véges tömbben. Ezen kívül tartalmazza a hozzá tartozó központ körzetszámát és számon tartja a maximális és az aktuális kapcsolatok számát **20**

2. Fájlmutató

2.1. Fájllista

Az összes fájl listája rövid leírásokkal:

telkozp/control.hpp
: ez tartalmazza a main() által meghívott vezérlő függvényeket 26

telkozp/ember.cpp
: tartalmazza az Ember osztály függvényeinek a definícióit 31

telkozp/ember.h
: tartalmazza az Ember osztály függvényeinek deklarációit 32

telkozp/kapcsolat.cpp
: tartalmazza a Kapcsolat osztály függvényeinek a megvalósítását 33

telkozp/kapcsolat.h
: tartalmazza a Kapcsolat osztály függvényeinek deklarációit 33

telkozp/kapcstabla.cpp
: tartalmazza a Tabla osztály függvényeinek a megvalósítását 35

telkozp/kapcstabla.h
: tartalmazza a Tabla osztály függvényeinek deklarációit 36

telkozp/kozpont.cpp
: tartalmazza a Kozpont osztály függvényeinek a megvalósítását 37

telkozp/kozpont.h
: tartalmazza a Kozpont osztály függvényeinek a deklarációit, valamint a getline függvény cross-platform megvalósítását 38

telkozp/test.cpp
: előre megadott paraméterekkel minden függvényt kipróbál 39

telkozp/test.h
: tartalmazza a teszt függvény deklarációját 41

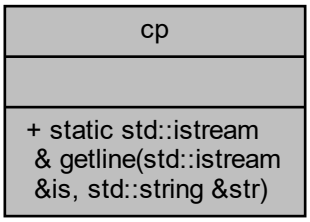
3. Osztályok dokumentációja

3.1. cp struktúrareferencia

cross-platform getline

```
#include <kozpont.h>
```

A cp osztály együttműködési diagramja:



Statikus publikus tagfüggvények

- static std::istream & getline (std::istream &is, std::string &str)

  // NOLINT(clang-diagnostic-documentation-unknown-command) Beolvas egy adott bemeneti folyamról egy sort egy adott string-be, majd ha szükség van rá, a stringből kiszedi a \r karaktert

3.1.1. Részletes leírás

cross-platform getline

Definíció a(z) kozpont.h fájl 24. sorában.

3.1.2. Tagfüggvények dokumentációja

3.1.2.1. getline()

```
static std::istream& cp::getline (
        std::istream & is,
        std::string & str )  [inline], [static]
```

// NOLINT(clang-diagnostic-documentation-unknown-command) Beolvas egy adott bemeneti folyamról egy sort egy adott string-be, majd ha szükség van rá, a stringből kiszedi a \r karaktert

Paraméterek

is	bemeneti folyam
str	cél string

Visszatérési érték

bemeneti folyam

Definíció a(z) kozpont.h fájl 32. sorában.

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

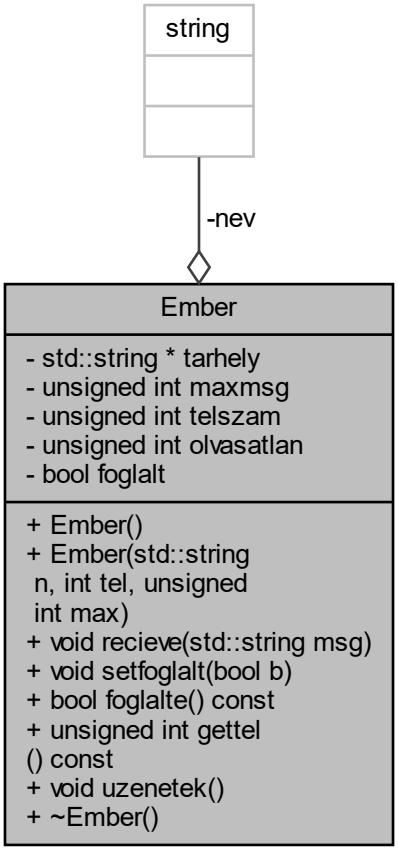
- telkozp/kozpont.h

3.2. Ember osztályreferencia

Egy előfizető személy adatait tárolja: A nevét; az üzeneteit egy véges tömbben, melyek közül számon tartja, hogy mennyi olvasatlan; a telefonszámát; és a foglaltsági státuszát.

`#include <ember.h>`

Az Ember osztály együttműködési diagramja:



Publikus tagfüggvények

- Ember ()

  Mindent 0-val vagy NULL-lal inicializál.
- Ember (std::string n, int tel, unsigned int max)

  Lefoglalja a megadott méretű üzenettárhelyet és beállítja a többi tagváltozót is.

- void recieve (std::string msg)

  Üzenet fogadása a fennálló kapcsolaton keresztül. Const char∗ kivételt dob, ha sikertelen volt.
- void setfoglalt (bool b)

  Beállítja a foglalt változó értékét, ha kapcsolatba kerül az adott ember valakivel
- bool foglalte () const
- unsigned int gettel () const
- void uzenetek ()

  Az üzenetek kezelése / olvasása, törlése.
- ∼Ember ()

  Felszabadítja az üzenetek tárhelyét.

Privát attribútumok

- std::string nev
- std::string ∗ tarhely
- unsigned int maxmsg
- unsigned int telszam
- unsigned int olvasatlan
- bool foglalt

Barátok

- std::ostream & operator<< (std::ostream &os, const Ember ∗e)

3.2.1. Részletes leírás

Egy előfizető személy adatait tárolja: A nevét; az üzeneteit egy véges tömbben, melyek közül számon tartja, hogy mennyi olvasatlan; a telefonszámát; és a foglaltsági státuszát.

Definíció a(z) ember.h fájl 11. sorában.

3.2.2. Konstruktorok és destruktorok dokumentációja

3.2.2.1. Ember() [1/2] Ember::Ember ( ) [inline]

Mindent 0-val vagy NULL-lal inicializál.

Definíció a(z) ember.h fájl 24. sorában.

3.2.2.2. Ember() [2/2] Ember::Ember (

```
std::string n,
int tel,
unsigned int max = 10 )
```

Lefoglalja a megadott méretű üzenettárhelyet és beállítja a többi tagváltozót is.

Konstruktor

Paraméterek

n	név
tel	telefonszám - négyjegyű
max	üzenettárhely mérete, alapértelmezetten 10

Definíció a(z) ember.cpp fájl 12. sorában.

3.2.2.3. ∼Ember() `Ember::∼Ember ( ) [inline]`

Felszabadítja az üzenetek tárhelyét.

Definíció a(z) ember.h fájl 65. sorában.

3.2.3. Tagfüggvények dokumentációja

3.2.3.1. foglalte() `bool Ember::foglalte ( ) const [inline]`

Visszatérési érték

foglalt-e

Definíció a(z) ember.h fájl 50. sorában.

3.2.3.2. gettel() `unsigned int Ember::gettel ( ) const [inline]`

Visszatérési érték

Telefonszám

Definíció a(z) ember.h fájl 55. sorában.

3.2.3.3. recieve() `void Ember::recieve ( std::string msg )`

Üzenet fogadása a fennálló kapcsolaton keresztül. Const char∗ kivételt dob, ha sikertelen volt.

Paraméterek

msg	A fogadott üzenet

Definíció a(z) ember.cpp fájl 25. sorában.

3.2.3.4. setfoglalt() `void Ember::setfoglalt ( bool b ) [inline]`

Beállítja a foglalt változó értékét, ha kapcsolatba kerül az adott ember valakivel

Paraméterek

b	Foglalt legyen-e, vagy se?

Definíció a(z) ember.h fájl 45. sorában.

3.2.3.5. uzenetek() `void Ember::uzenetek ( )`

Az üzenetek kezelése / olvasása, törlése.

Definíció a(z) ember.cpp fájl 41. sorában.

3.2.4. Barát és kapcsolódó függvények dokumentációja

3.2.4.1. operator<< `std::ostream& operator<< ( std::ostream & os, const Ember * e ) [friend]`

Definíció a(z) ember.h fájl 67. sorában.

3.2.5. Adattagok dokumentációja

3.2.5.1. foglalt `bool Ember::foglalt [private]`

Definíció a(z) ember.h fájl 17. sorában.

3.2.5.2. maxmsg `unsigned int Ember::maxmsg [private]`

Definíció a(z) ember.h fájl 14. sorában.

3.2.5.3. nev `std::string Ember::nev [private]`

Definíció a(z) ember.h fájl 12. sorában.

3.2.5.4. olvasatlan `unsigned int Ember::olvasatlan [private]`

Definíció a(z) ember.h fájl 16. sorában.

3.2.5.5. tarhely `std::string* Ember::tarhely [private]`

Definíció a(z) ember.h fájl 13. sorában.

3.2.5.6. telszam `unsigned int Ember::telszam [private]`

Definíció a(z) ember.h fájl 15. sorában.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

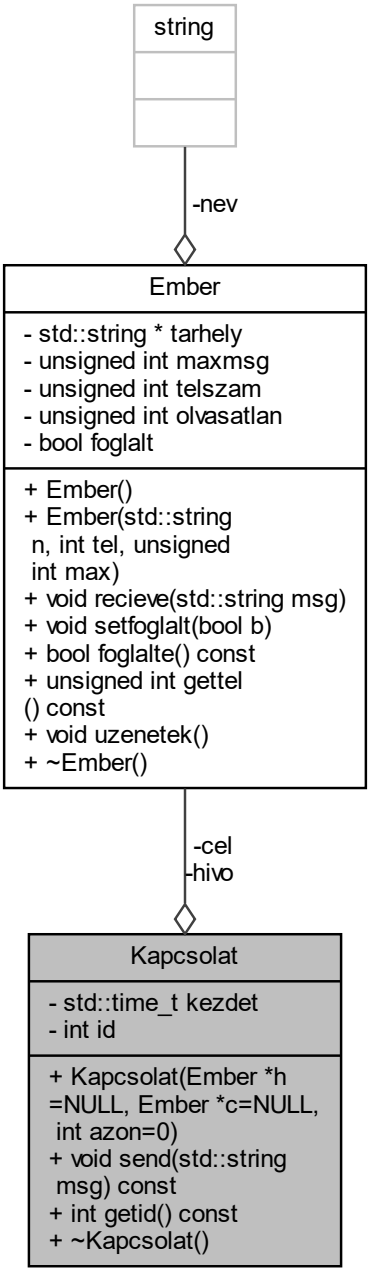
- telkozp/ember.h
- telkozp/ember.cpp

3.3. Kapcsolat osztályreferencia

Két emberre mutató pointert tárol, akik között fennáll a kapcsolat, továbbá az azonosítóját és egy időpillanatot, amikor létrejött. Lehet rajta keresztül a hívó személynek üzenetet küldeni a hívott részére.

```
#include <kapcsolat.h>
```

A Kapcsolat osztály együttműködési diagramja:



Publikus tagfüggvények

- Kapcsolat (Ember ∗h=NULL, Ember ∗c=NULL, int azon=0)

  azonosítószám a könnyebb kezelésért, a hívó és a foogadó fél telefonszámainak az összefűzése (8 jegyű)
- void send (std::string msg) const

  Üzenet küldés a hívott félnek, const char∗ kivételt dob, ha sikertelen volt.
- int getid () const

Id igénylés.

- ~Kapcsolat ()

  Üzenetet küld a hívott félnek és kiírja az eltelt időt, az ember pointereket nem kell felszabadítani, majd a vektorból szabadítjuk fel őket a főprogramban.

Privát attribútumok

- Ember * hivo
- Ember * cel
- std::time_t kezdet
- int id

  kapcsolat létrehozásának időpontja

Barátok

- std::ostream & operator<< (std::ostream &os, Kapcsolat *k)

  << *operátor, hogy a kapcsolatokat ki lehessen írni egy os-re*

3.3.1. Részletes leírás

Két emberre mutató pointert tárol, akik között fennáll a kapcsolat, továbbá az azonosítóját és egy időpillanatot, amikor létrejött. Lehet rajta keresztül a hívó személynek üzenetet küldeni a hívott részére.

Definíció a(z) kapcsolat.h fájl 19. sorában.

3.3.2. Konstruktorok és destruktorok dokumentációja

3.3.2.1. Kapcsolat()

```
Kapcsolat::Kapcsolat (
        Ember * h = NULL,
        Ember * c = NULL,
        int azon = 0 )  [inline]
```

azonosítószám a könnyebb kezelésért, a hívó és a foogadó fél telefonszámainak az összefűzése (8 jegyű)

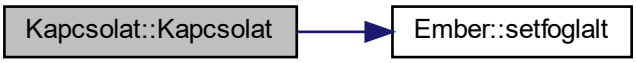
A telefonközpont adja neki az emberek referenciáit, beállítja a kezdőidőt.

Paraméterek

h	A hívó fél
c	A hívott fél
azon	A kapcsolat leendő azonosítója

Definíció a(z) kapcsolat.h fájl 34. sorában.

A függvény hívási gráfja:

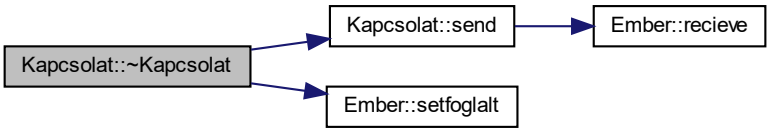


3.3.2.2. ∼Kapcsolat() `Kapcsolat::∼Kapcsolat ( ) [inline]`

Üzenetet küld a hívott félnek és kiírja az eltelt időt, az ember pointereket nem kell felszabadítani, majd a vektorból szabadítjuk fel őket a főprogramban.

Definíció a(z) kapcsolat.h fájl 65. sorában.

A függvény hívási gráfja:



3.3.3. Tagfüggvények dokumentációja

3.3.3.1. getid() `int Kapcsolat::getid ( ) const [inline]`

Id igénylés.

Visszatérési érték

a kapcsolat azonosítószáma

Definíció a(z) kapcsolat.h fájl 60. sorában.

3.3.3.2. send() `void Kapcsolat::send ( std::string msg ) const`

Üzenet küldés a hívott félnek, const char∗ kivételt dob, ha sikertelen volt.

Paraméterek

msg	Az elküldendő üzenet

Definíció a(z) kapcsolat.cpp fájl 10. sorában.

A függvény hívási gráfja:

3.3.4. Barát és kapcsolódó függvények dokumentációja

3.3.4.1. operator<< `std::ostream& operator<< ( std::ostream & os, Kapcsolat * k ) [friend]`

<< operátor, hogy a kapcsolatokat ki lehessen írni egy os-re

Paraméterek

os	kimeneti folyam
k	egy kapcsolatra mutató pointer

Visszatérési érték

kimeneti folyam

Definíció a(z) kapcsolat.h fájl 101. sorában.

3.3.5. Adattagok dokumentációja

3.3.5.1. cel `Ember* Kapcsolat::cel [private]`

Definíció a(z) kapcsolat.h fájl 21. sorában.

3.3.5.2. hivo `Ember* Kapcsolat::hivo [private]`

Definíció a(z) kapcsolat.h fájl 20. sorában.

3.3.5.3. id `int Kapcsolat::id [private]`

kapcsolat létrehozásának időpontja

Definíció a(z) kapcsolat.h fájl 23. sorában.

3.3.5.4. kezdet `std::time_t Kapcsolat::kezdet [private]`

Definíció a(z) kapcsolat.h fájl 22. sorában.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

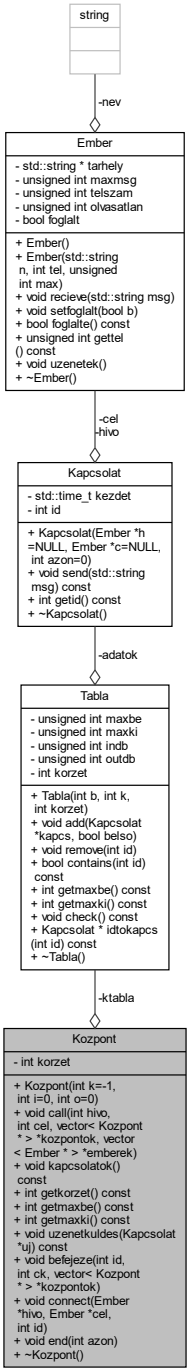
- telkozp/kapcsolat.h
- telkozp/kapcsolat.cpp

3.4. Kozpont osztályreferencia

Kapcsolatok tárolására és kezelésére szolgál. Tartalmaz egy Tabla objektumot, melyben a Kapcsolat objektumok tárolva vannak, továbbá a körzetszámát.

`#include <kozpont.h>`

A Kozpont osztály együttműködési diagramja:



Publikus tagfüggvények

- Kozpont (int k=-1, int i=0, int o=0)

  Létrehoz egy Tabla objektumot a megadott paraméterekkel és beállítja a körzetszámot.
- void call (int hivo, int cel, vector< Kozpont * > *kozpontok, vector< Ember * > *emberek)

  Híváskezdeményezés, const char kivételt dob, ha sikertelen volt.*
- void kapcsolatok () const

Vele lehet megtekinteni az éppen fennálló kapcsolatait a központnak és létrehozni új kapcsolatot

- int getkorzet () const
- int getmaxbe () const
- int getmaxki () const
- void uzenetkuldes (Kapcsolat ∗uj) const

  Gondoskodik a kapcsolaton belüli üzenetküldés vezérléséről
- void befejeze (int id, int ck, vector< Kozpont ∗ > ∗kozpontok)

  Gondoskodik a kapcsolat befejezésének az eldöntéséről
- void connect (Ember ∗hivo, Ember ∗cel, int id)

  Kapcsol egy távolsági hívást, const char∗ kivételt dob, ha sikertelen volt
- void end (int azon)

  Egy kapcsolat megszűntetése és törlése a kapcsolótáblából, a kapcsolatot es a pointerét mindig a hívó ügyfél központja törölje. Const char∗ kivételt dob, ha sikertelen volt.
- ∼Kozpont ()

Privát attribútumok

- int korzet

  körzetszám, 1-9
- Tabla ktabla

  itt tárolja a kapcsolataira mutató pointereket

3.4.1. Részletes leírás

Kapcsolatok tárolására és kezelésére szolgál. Tartalmaz egy Tabla objektumot, melyben a Kapcsolat objektumok tárolva vannak, továbbá a körzetszámát.

Definíció a(z) kozpont.h fájl 47. sorában.

3.4.2. Konstruktorok és destruktorok dokumentációja

3.4.2.1. Kozpont()

```
Kozpont::Kozpont (
        int k = -1,
        int i = 0,
        int o = 0 )  [inline]
```

Létrehoz egy Tabla objektumot a megadott paraméterekkel és beállítja a körzetszámot.

Paraméterek

k	körzetszám
i	belső kapcsolatok max száma
o	külső kapcsolatok max száma

Definíció a(z) kozpont.h fájl 60. sorában.

3.4.2.2. ~Kozpont() `Kozpont::~Kozpont ( ) [inline]`

Definíció a(z) kozpont.h fájl 124. sorában.

3.4.3. Tagfüggvények dokumentációja

3.4.3.1. befejeze()

```
void Kozpont::befejeze (
        int id,
        int ck,
        vector< Kozpont * > * kozpontok )
```

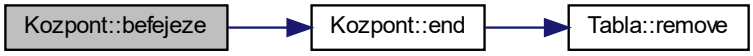
Gondoskodik a kapcsolat befejezésének az eldöntéséről

Paraméterek

id	a kapcsolat azonosítója
ck	a hívott fél körzete
kozpontok	a központokat tároló vektorra mutató pointer

Definíció a(z) kozpont.cpp fájl 14. sorában.

A függvény hívási gráfja:



3.4.3.2. call()

```
void Kozpont::call (
        int hivo,
        int cel,
        vector< Kozpont * > * kozpontok,
        vector< Ember * > * emberek )
```

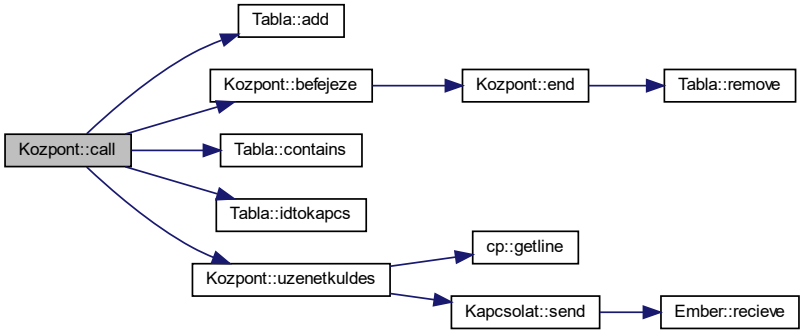
Híváskezdeményezés, const char* kivételt dob, ha sikertelen volt.

Paraméterek

hivo	A hívó telefonszáma
cel	A hívott telefonszáma
kozpontok	A központok tárolójának a pointere
emberek	Az emberek tárolójának a pointere

Definíció a(z) kozpont.cpp fájl 55. sorában.

A függvény hívási gráfja:



3.4.3.3. connect()

```
void Kozpont::connect (
        Ember * hivo,
        Ember * cel,
        int id )
```

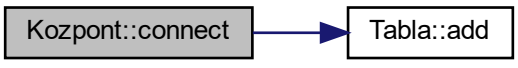
Kapcsol egy távolsági hívást, const char* kivételt dob, ha sikertelen volt

Paraméterek

hivo	a hívó emberre mutató pointer
cel	a hívott emberre mutató pointer
id	a kapcsolat leendő azonosítója

Definíció a(z) kozpont.cpp fájl 242. sorában.

A függvény hívási gráfja:



3.4.3.4. end() `void Kozpont::end ( int azon )`

Egy kapcsolat megszűntetése és törlése a kapcsolótáblából, a kapcsolatot es a pointerét mindig a hívó ügyfél központja törölje. Const char∗ kivételt dob, ha sikertelen volt.

Paraméterek

azon	A kapcsolat azonosítószáma

Definíció a(z) kozpont.cpp fájl 184. sorában.

A függvény hívási gráfja:

3.4.3.5. getkorzet() `int Kozpont::getkorzet ( ) const [inline]`

Visszatérési érték

Körzetszám

Definíció a(z) kozpont.h fájl 80. sorában.

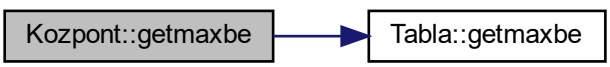
3.4.3.6. getmaxbe() `int Kozpont::getmaxbe ( ) const [inline]`

Visszatérési érték

A belső kapcsolatok maximális számma.

Definíció a(z) kozpont.h fájl 85. sorában.

A függvény hívási gráfja:



3.4.3.7. getmaxki() `int Kozpont::getmaxki ( ) const [inline]`

Visszatérési érték

A maximális külső kapcsolatok száma.

Definíció a(z) kozpont.h fájl 90. sorában.

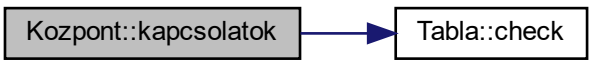
A függvény hívási gráfja:

3.4.3.8. kapcsolatok() `void Kozpont::kapcsolatok ( ) const`

Vele lehet megtekinteni az éppen fennálló kapcsolatait a központnak és létrehozni új kapcsolatot

Definíció a(z) kozpont.cpp fájl 174. sorában.

A függvény hívási gráfja:



3.4.3.9. uzenetkuldes() `void Kozpont::uzenetkuldes ( Kapcsolat * uj ) const`

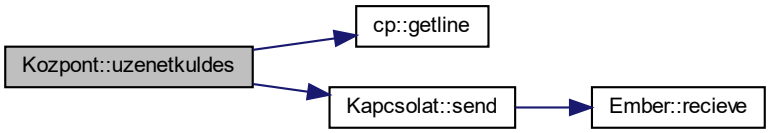
Gondoskodik a kapcsolaton belüli üzenetküldés vezérléséről

Paraméterek

uj	A kapcsolatra mutató pointer

Definíció a(z) kozpont.cpp fájl 200. sorában.

A függvény hívási gráfja:



3.4.4. Adattagok dokumentációja

3.4.4.1. korzet `int Kozpont::korzet [private]`

körzetszám, 1-9

Definíció a(z) kozpont.h fájl 48. sorában.

3.4.4.2. ktabla `Tabla Kozpont::ktabla [private]`

itt tárolja a kapcsolataira mutató pointereket

Definíció a(z) kozpont.h fájl 49. sorában.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

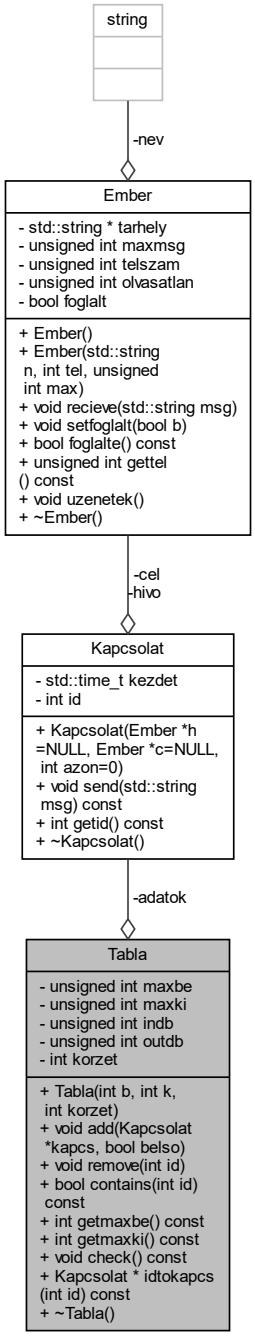
- telkozp/kozpont.h
- telkozp/kozpont.cpp

3.5. Tabla osztályreferencia

Kapcsolat objektumokra mutató pointereket tárol egy véges tömbben. Ezen kívül tartalmazza a hozzá tartozó központ körzetszámát és számon tartja a maximális és az aktuális kapcsolatok számát.

```
#include <kapcstabla.h>
```

A Tabla osztály együttműködési diagramja:



Publikus tagfüggvények

- Tabla (int b, int k, int korzet)

  Létrehozza a Kapcsolat -ok pointereinek a tömbjét és inicializálja NULL pointerekkel.
- void add (Kapcsolat ∗kapcs, bool belso)

  Hozzáad egy új kapcsolatot a táblába, ha lehet, const char∗ kivételt dob, ha sikertelen volt.
- void remove (int id)

Megszűntet egy kapcsolatot, const char∗ kivételt dob, ha sikertelen volt.

- bool contains (int id) const

  Megmondja, hogy a tábla tartalmazza-e az adott azonosítójú kapcsolatot
- int getmaxbe () const
- int getmaxki () const
- void check () const

  A tábla tartalmának megtekintése
- Kapcsolat ∗ idtokapcs (int id) const

  Egy megadott id alapjan megkeresi es visszater az adott kapcsolat pointerevel, vagy NULL-lal
- ∼Tabla ()

  Felszabadítja a Kapcsolat pointereket, majd törli a tömböt.

Privát attribútumok

- Kapcsolat ∗∗ adatok

  kapcsolatok tömbje
- unsigned int maxbe

  tároló mérete belső kapcsolóknak
- unsigned int maxki

  tároló mérete külső kapcsolóknak
- unsigned int indb

  belső kapcsolatok darabszáma
- unsigned int outdb

  külső kapcsolatok darabszáma
- int korzet

  a központjának a körzetszáma

3.5.1. Részletes leírás

Kapcsolat objektumokra mutató pointereket tárol egy véges tömbben. Ezen kívül tartalmazza a hozzá tartozó központ körzetszámát és számon tartja a maximális és az aktuális kapcsolatok számát.

Definíció a(z) kapcstabla.h fájl 14. sorában.

3.5.2. Konstruktorok és destruktorok dokumentációja

3.5.2.1. Tabla()

```
Tabla::Tabla (
        int b,
        int k,
        int korzet )  [inline]
```

Létrehozza a Kapcsolat -ok pointereinek a tömbjét és inicializálja NULL pointerekkel.

Paraméterek

b	a létrehozandó tábla belső maximális kapcolatszáma
k	a létrehozandó tábla külső maximális kapcolatszáma

Definíció a(z) kapcstabla.h fájl 29. sorában.

3.5.2.2. ~Tabla() `Tabla::~Tabla ( ) [inline]`

Felszabadítja a Kapcsolat pointereket, majd törli a tömböt.

Definíció a(z) kapcstabla.h fájl 83. sorában.

3.5.3. Tagfüggvények dokumentációja

3.5.3.1. add() `void Tabla::add ( Kapcsolat * kapcs, bool belso )`

Hozzáad egy új kapcsolatot a táblába, ha lehet, const char* kivételt dob, ha sikertelen volt.

Paraméterek

kapcs	a táblához adandó kapcsolat pointere, a kapcsolatot a központ hozza létre
belso	megmondja, hogy belső kapcsolatról van-szó

Definíció a(z) kapcstabla.cpp fájl 11. sorában.

3.5.3.2. check() `void Tabla::check ( ) const`

A tábla tartalmának megtekintése

Definíció a(z) kapcstabla.cpp fájl 93. sorában.

3.5.3.3. contains() `bool Tabla::contains ( int id ) const`

Megmondja, hogy a tábla tartalmazza-e az adott azonosítójú kapcsolatot

Paraméterek

id	a kapcsolat azonosítója

Visszatérési érték

tartalmazza-e

Definíció a(z) kapcstabla.cpp fájl 109. sorában.

3.5.3.4. getmaxbe() `int Tabla::getmaxbe ( ) const [inline]`

Visszatérési érték

A maximális belső kapcsolatok száma.

Definíció a(z) kapcstabla.h fájl 61. sorában.

3.5.3.5. getmaxki() `int Tabla::getmaxki ( ) const [inline]`

Visszatérési érték

A maximális külső kapcsolatok száma.

Definíció a(z) kapcstabla.h fájl 66. sorában.

3.5.3.6. idtokapcs() `Kapcsolat * Tabla::idtokapcs ( int id ) const`

Egy megadott id alapjan megkeresi es visszater az adott kapcsolat pointerevel, vagy NULL-lal

Paraméterek

id	a kapcsolat azonosítója

Visszatérési érték

a keresett kapcsolatra mutató pointer

Definíció a(z) kapcstabla.cpp fájl 124. sorában.

3.5.3.7. remove() `void Tabla::remove ( int id )`

Megszűntet egy kapcsolatot, const char* kivételt dob, ha sikertelen volt.

Paraméterek

id	a kapcsolat azonosítója

Definíció a(z) kapcstabla.cpp fájl 59. sorában.

3.5.4. Adattagok dokumentációja

3.5.4.1. adatok `Kapcsolat** Tabla::adatok [private]`

kapcsolatok tömbje

Definíció a(z) kapcstabla.h fájl 15. sorában.

3.5.4.2. indb `unsigned int Tabla::indb [private]`

belső kapcsolatok darabszáma

Definíció a(z) kapcstabla.h fájl 18. sorában.

3.5.4.3. korzet `int Tabla::korzet [private]`

a központjának a körzetszáma

Definíció a(z) kapcstabla.h fájl 20. sorában.

3.5.4.4. maxbe `unsigned int Tabla::maxbe [private]`

tároló mérete belső kapcsolóknak

Definíció a(z) kapcstabla.h fájl 16. sorában.

3.5.4.5. maxki `unsigned int Tabla::maxki [private]`

tároló mérete külső kapcsolóknak

Definíció a(z) kapcstabla.h fájl 17. sorában.

3.5.4.6. outdb `unsigned int Tabla::outdb [private]`

külső kapcsolatok darabszáma

Definíció a(z) kapcstabla.h fájl 19. sorában.

Ez a dokumentáció az osztályról a következő fájlok alapján készült:

- telkozp/kapcstabla.h
- telkozp/kapcstabla.cpp

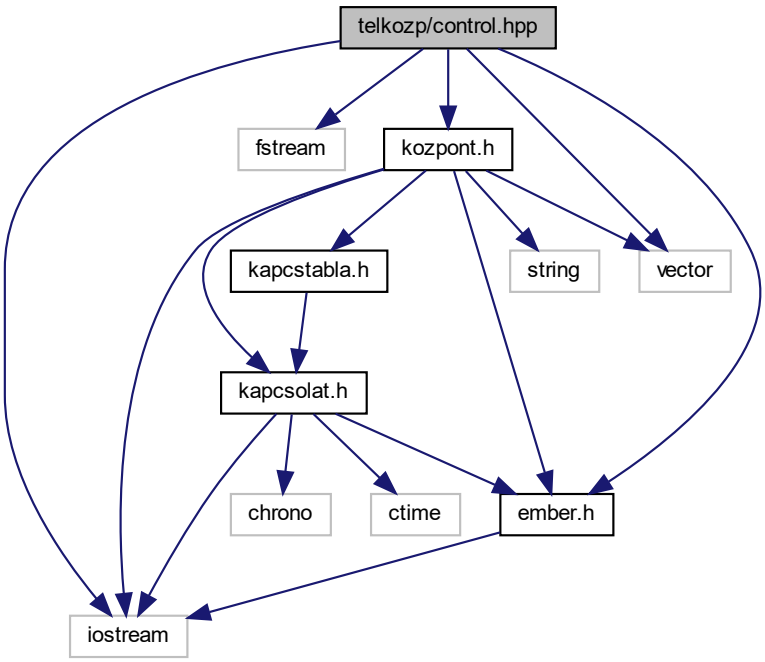
4. Fájlok dokumentációja

4.1. telkozp/control.hpp fájlreferencia

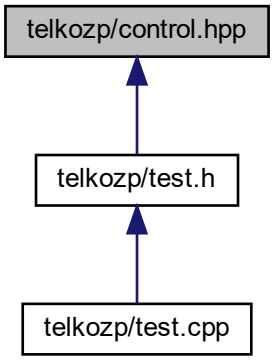
: ez tartalmazza a main() által meghívott vezérlő függvényeket

```
#include <iostream>
#include <fstream>
#include "kozpont.h"
#include "ember.h"
#include <vector>
```

A control.hpp definíciós fájl függési gráfja:



Ez az ábra azt mutatja, hogy mely fájlok ágyazzák be közvetve vagy közvetlenül ezt a fájlt:



Függvények

- void cerror (const char ∗exc)

  A vezérlő szekcióban egységes hibakiíró függvény.
- void addnkp (vector< Kozpont ∗ > ∗kozpontok)

  Új központ hozzáadása a standard inputról.
- void addnmbr (vector< Ember ∗ > ∗emberek, vector< Kozpont ∗ > ∗kozpontok)

  Új ember hozzáadása a standard inputról.
- void maincall (vector< Kozpont ∗ > ∗kozpontok, vector< Ember ∗ > ∗emberek)

  Hívást / kapcsolatkezelést vezérlő függvény.
- void login (vector< Ember ∗ > ∗emberek)

  Az üzenetek megtekintését vezérlő függvény.
- void kozpinfo (vector< Kozpont ∗ > ∗kozpontok)

  Egy adott központtól való adatigénylést vezérlő függvény
- void dimport (vector< Kozpont ∗ > ∗kozpontok, vector< Ember ∗ > ∗emberek, string kpath="", string epath="")

  Központok és emberek file-ból való importálása
- void dexport (vector< Kozpont ∗ > ∗kozpontok, vector< Ember ∗ > ∗emberek)

  A központok és az emberek exportálása 1-1 .txt file-ba.

Változók

- static string errstr = "Rosszul adta meg a parametereket, vagy nem adott meg semmit :("

  Egységes hibaüzenet a hasonló típusú hibákhoz.

4.1.1. Részletes leírás

: ez tartalmazza a main() által meghívott vezérlő függvényeket

4.1.2. Függvények dokumentációja

4.1.2.1. addnkp() void addnkp (vector< Kozpont * > * kozpontok) [inline]

Új központ hozzáadása a standard inputról.

Paraméterek

kozpontok	a központokat tároló vektorra mutató pointer

Definíció a(z) control.hpp fájl 39. sorában.

4.1.2.2. addnmbr() void addnmbr (vector< Ember * > * emberek, vector< Kozpont * > * kozpontok) [inline]

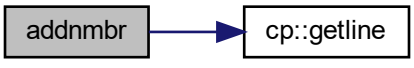
Új ember hozzáadása a standard inputról.

Paraméterek

emberek	az embereket tároló vektorra mutató pointer
kozpontok	a központokat tároló vektorra mutató pointer

Definíció a(z) control.hpp fájl 81. sorában.

A függvény hívási gráfja:



4.1.2.3. cerror() void cerror (const char * exc) [inline]

A vezérlő szekcióban egységes hibakiíró függvény.

Paraméterek

exc	const char* kivétel

Definíció a(z) control.hpp fájl 24. sorában.

4.1.2.4. dexport()

```
void dexport (
        vector< Kozpont * > * kozpontok,
        vector< Ember * > * emberek )  [inline]
```

A központok és az emberek exportálása 1-1 .txt file-ba.

Paraméterek

kozpontok	A központokat tartalmazó vektorra mutató pointer.
emberek	Az embereket tartalmazó vektorra mutató pointer.

Definíció a(z) control.hpp fájl 325. sorában.

4.1.2.5. dimport()

```
void dimport (
        vector< Kozpont * > * kozpontok,
        vector< Ember * > * emberek,
        string kpath = "",
        string epath = "" )  [inline]
```

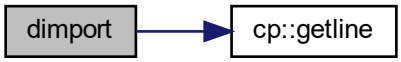
Központok és emberek file-ból való importálása

Paraméterek

kozpontok	a központokat tároló vektorra mutató pointer
emberek	az embereket tároló vektorra mutató pointer
kpath	a központokat tartalmazó file neve és elérési útvonala, ha nincs megadva, akkor a standard inputon keresztül kell majd megadni
epath	az embereket tartalmazó file neve és elérési útvonala, ha nincs megadva, akkor a standard inputon keresztül kell majd megadni

Definíció a(z) control.hpp fájl 208. sorában.

A függvény hívási gráfja:



4.1.2.6. kozpinfo() void kozpinfo (vector< Kozpont * > * kozpontok) [inline]

Egy adott központtól való adatigénylést vezérlő függvény

Paraméterek

kozpontok	

Definíció a(z) control.hpp fájl 183. sorában.

4.1.2.7. login() void login (vector< Ember * > * emberek) [inline]

Az üzenetek megtekintését vezérlő függvény.

Paraméterek

emberek	az embereket tároló vektorra mutató pointer

Definíció a(z) control.hpp fájl 160. sorában.

4.1.2.8. maincall() void maincall (vector< Kozpont * > * kozpontok, vector< Ember * > * emberek) [inline]

Hívást / kapcsolatkezelést vezérlő függvény.

Paraméterek

kozpontok	a központokat tároló vektorra mutató pointer
emberek	az embereket tároló vektorra mutató pointer

Definíció a(z) control.hpp fájl 134. sorában.

4.1.3. Változók dokumentációja

4.1.3.1. errstr `string errstr = "Rosszul adta meg a parametereket, vagy nem adott meg semmit :(." [static]`

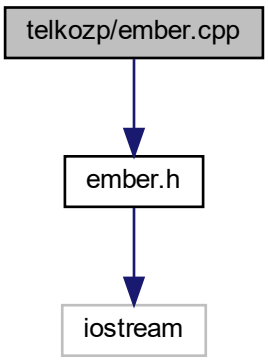
Egységes hibaüzenet a hasonló típusú hibákhoz.

Definíció a(z) control.hpp fájl 33. sorában.

4.2. telkozp/ember.cpp fájlreferencia

: tartalmazza az Ember osztály függvényeinek a definícióit

`#include "ember.h"`
Az ember.cpp definíciós fájl függési gráfja:



4.2.1. Részletes leírás

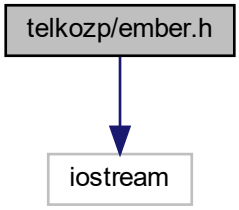
: tartalmazza az Ember osztály függvényeinek a definícióit

4.3. telkozp/ember.h fájlreferencia

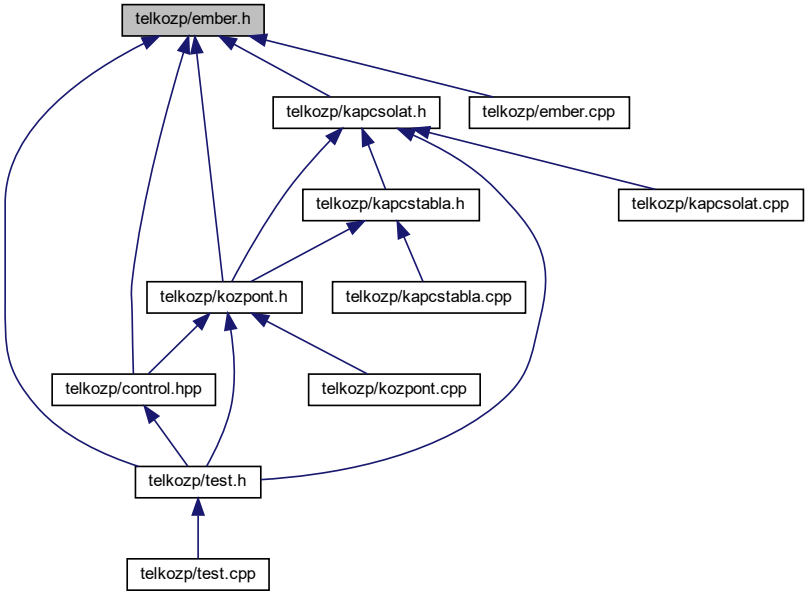
: tartalmazza az Ember osztály függvényeinek deklarációit

```
#include <iostream>
```

Az ember.h definíciós fájl függési gráfja:



Ez az ábra azt mutatja, hogy mely fájlok ágyazzák be közvetve vagy közvetlenül ezt a fájlt:



Osztályok

- class Ember

  Egy előfizető személy adatait tárolja: A nevét; az üzeneteit egy véges tömbben, melyek közül számon tartja, hogy mennyi olvasatlan; a telefonszámát; és a foglaltsági státuszát.

4.3.1. Részletes leírás

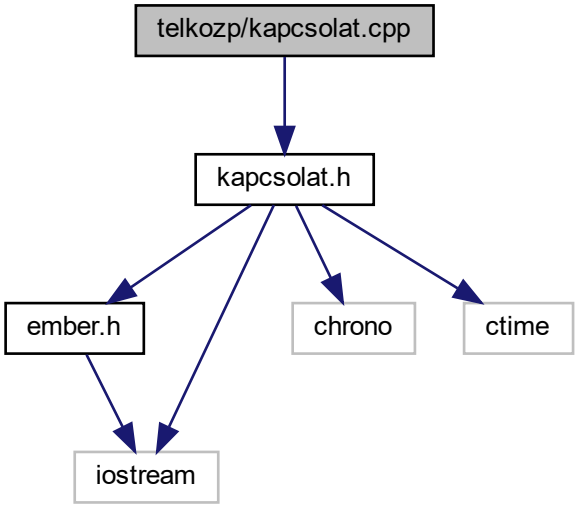
: tartalmazza az Ember osztály függvényeinek deklarációit

4.4. telkozp/kapcsolat.cpp fájlreferencia

: tartalmazza a Kapcsolat osztály függvényeinek a megvalósítását

```
#include "kapcsolat.h"
```

A kapcsolat.cpp definíciós fájl függési gráfja:



4.4.1. Részletes leírás

: tartalmazza a Kapcsolat osztály függvényeinek a megvalósítását

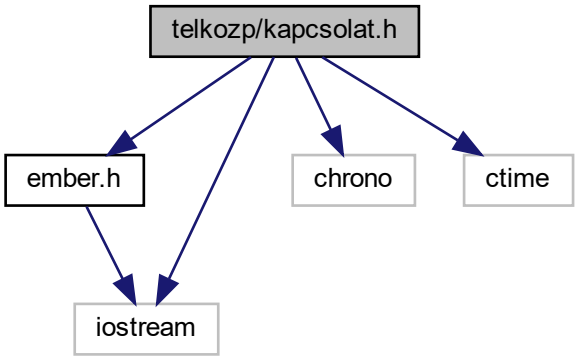
4.5. telkozp/kapcsolat.h fájlreferencia

: tartalmazza a Kapcsolat osztály függvényeinek deklarációit

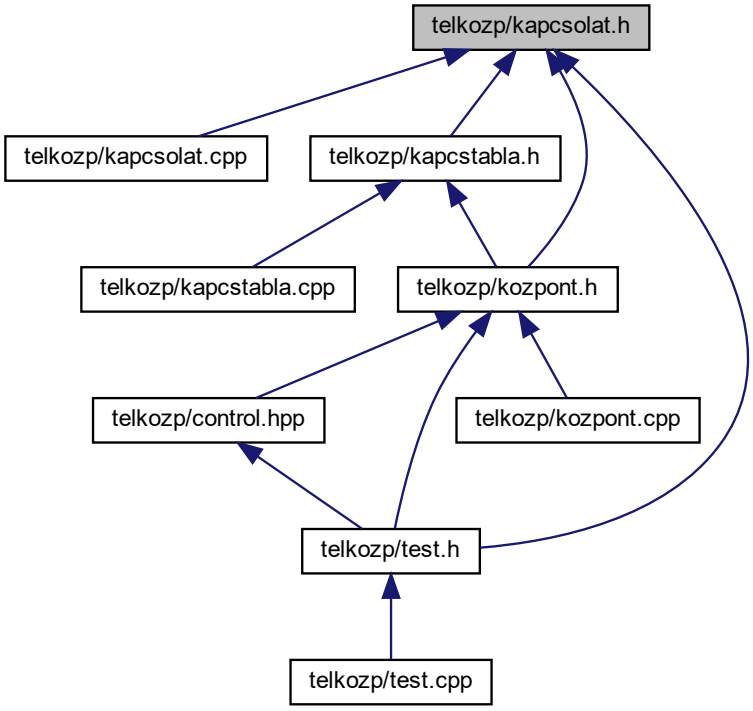
```
#include "ember.h"
#include <chrono>
#include <ctime>
```

```
#include <iostream>
```

A kapcsolat.h definíciós fájl függési gráfja:



Ez az ábra azt mutatja, hogy mely fájlok ágyazzák be közvetve vagy közvetlenül ezt a fájlt:



Osztályok

- class Kapcsolat

Két emberre mutató pointert tárol, akik között fennáll a kapcsolat, továbbá az azonosítóját és egy időpillanatot, amikor létrejött. Lehet rajta keresztül a hívó személynek üzenetet küldeni a hívott részére.

4.5.1. Részletes leírás

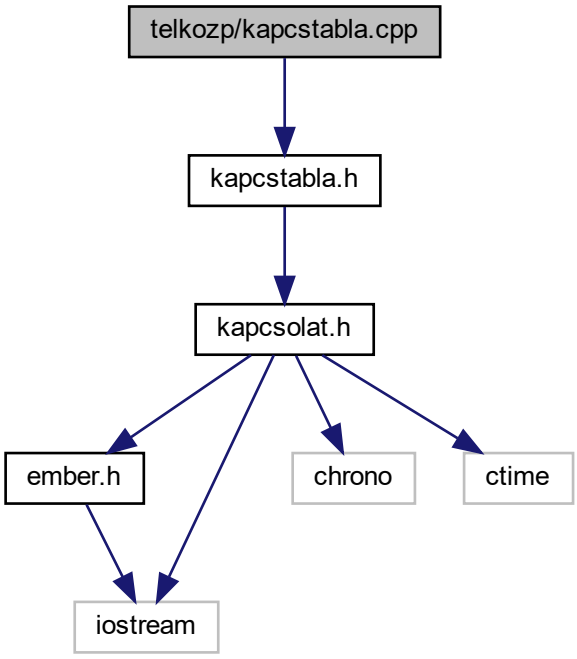
: tartalmazza a Kapcsolat osztály függvényeinek deklarációit

4.6. telkozp/kapcstabla.cpp fájlreferencia

: tartalmazza a Tabla osztály függvényeinek a megvalósítását

```
#include "kapcstabla.h"
```

A kapcstabla.cpp definíciós fájl függési gráfja:



4.6.1. Részletes leírás

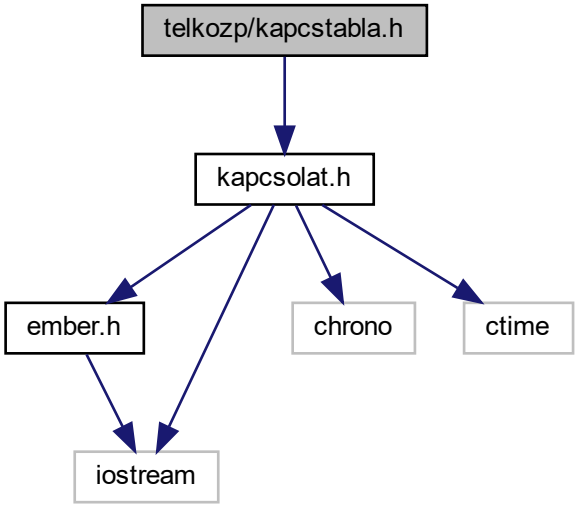
: tartalmazza a Tabla osztály függvényeinek a megvalósítását

4.7. telkozp/kapcstabla.h fájlreferencia

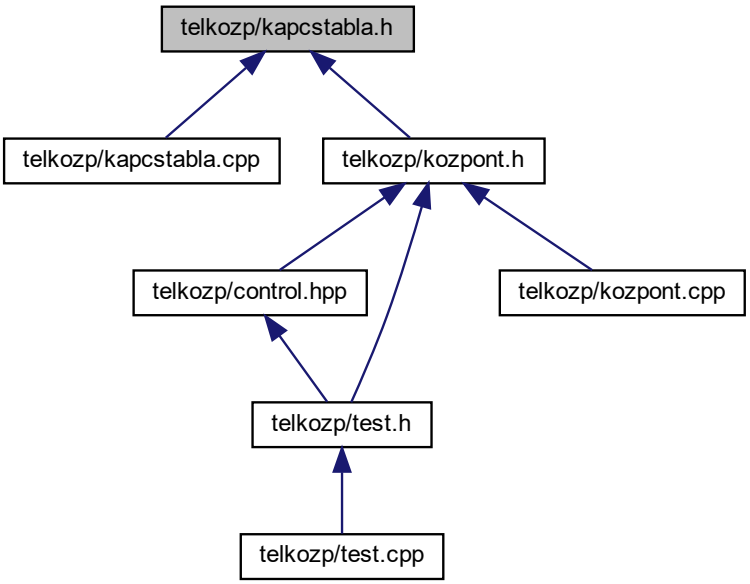
: tartalmazza a Tabla osztály függvényeinek deklarációit

```
#include "kapcsolat.h"
```

A kapcstabla.h definíciós fájl függési gráfja:



Ez az ábra azt mutatja, hogy mely fájlok ágyazzák be közvetve vagy közvetlenül ezt a fájlt:



Osztályok

- class Tabla

  Kapcsolat objektumokra mutató pointereket tárol egy véges tömbben. Ezen kívül tartalmazza a hozzá tartozó központ körzetszámát és számon tartja a maximális és az aktuális kapcsolatok számát.

4.7.1. Részletes leírás

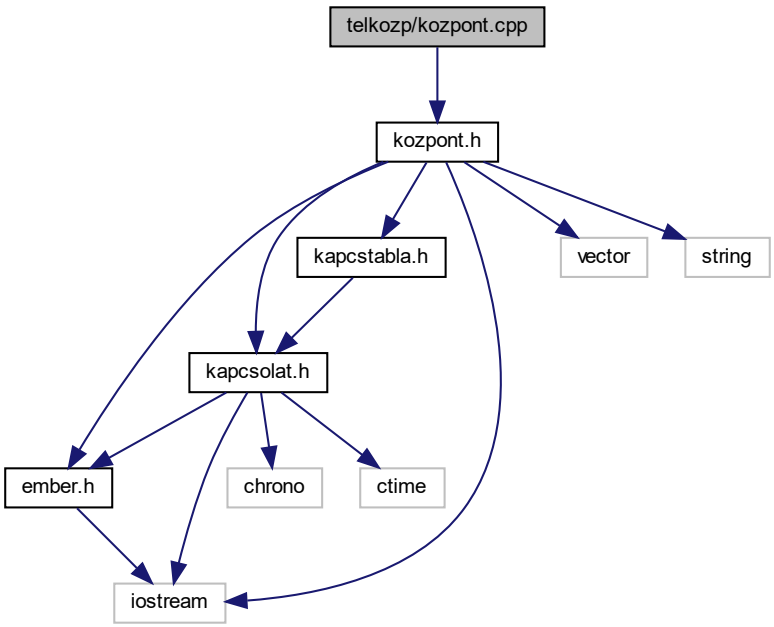
: tartalmazza a Tabla osztály függvényeinek deklarációit

4.8. telkozp/kozpont.cpp fájlreferencia

: tartalmazza a Kozpont osztály függvényeinek a megvalósítását

```
#include "kozpont.h"
```

A kozpont.cpp definíciós fájl függési gráfja:



4.8.1. Részletes leírás

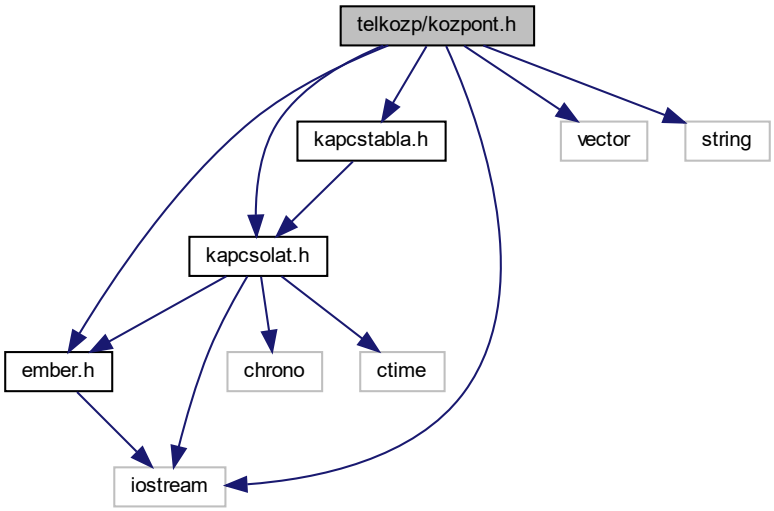
: tartalmazza a Kozpont osztály függvényeinek a megvalósítását

4.9. telkozp/kozpont.h fájlreferencia

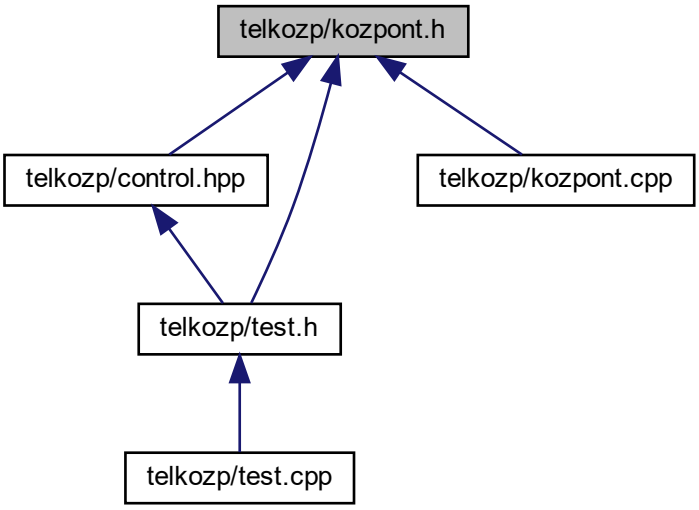
: tartalmazza a Kozpont osztály függvényeinek a deklarációit, valamint a getline függvény cross-platform megvalósítását

```
#include "kapcsolat.h"
#include "kapcstabla.h"
#include "ember.h"
#include <iostream>
#include <vector>
#include <string>
```

A kozpont.h definíciós fájl függési gráfja:



Ez az ábra azt mutatja, hogy mely fájlok ágyazzák be közvetve vagy közvetlenül ezt a fájlt:



Osztályok

- struct cp

  cross-platform getline
- class Kozpont

  Kapcsolatok tárolására és kezelésére szolgál. Tartalmaz egy Tabla objektumot, melyben a Kapcsolat objektumok tárolva vannak, továbbá a körzetszámát.

4.9.1. Részletes leírás

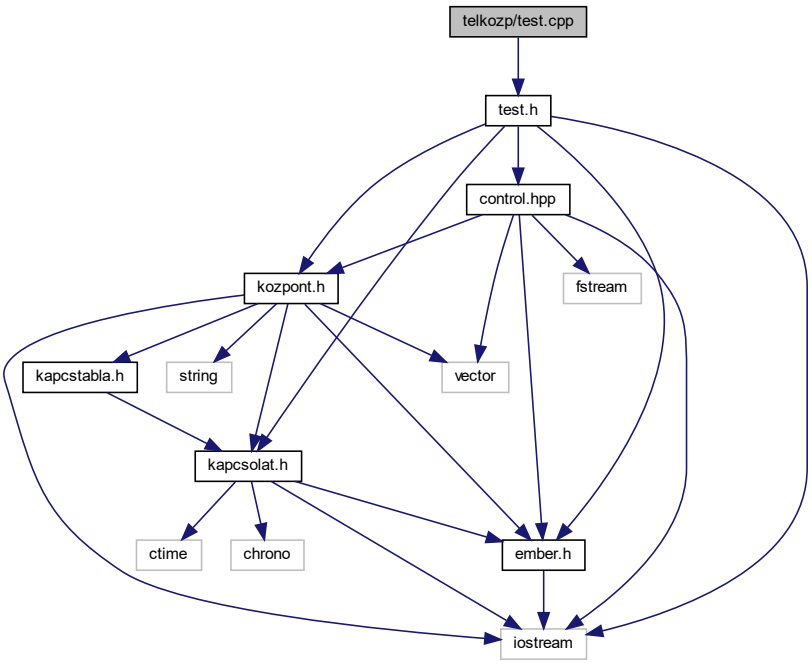
: tartalmazza a Kozpont osztály függvényeinek a deklarációit, valamint a getline függvény cross-platform megvalósítását

4.10. telkozp/test.cpp fájlreferencia

: előre megadott paraméterekkel minden függvényt kipróbál

```
#include "test.h"
```

A test.cpp definíciós fájl függési gráfja:



Függvények

- bool test_main ()

  A tesztprogram.

4.10.1. Részletes leírás

: előre megadott paraméterekkel minden függvényt kipróbál

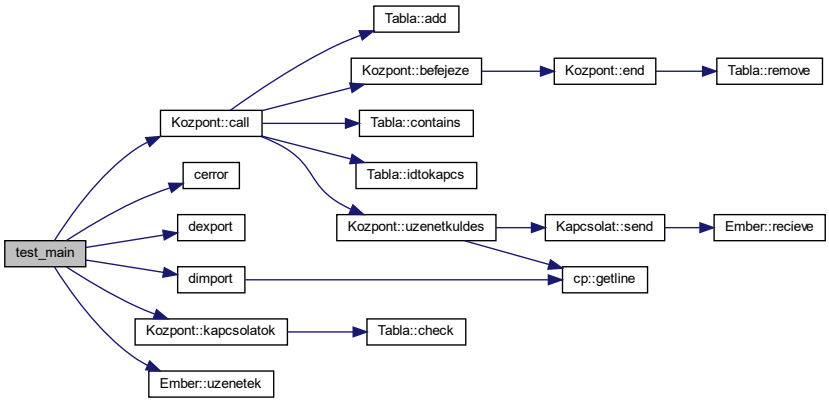
4.10.2. Függvények dokumentációja

4.10.2.1. test_main() `bool test_main ( )`

A tesztprogram.

Definíció a(z) test.cpp fájl 6. sorában.

A függvény hívási gráfja:

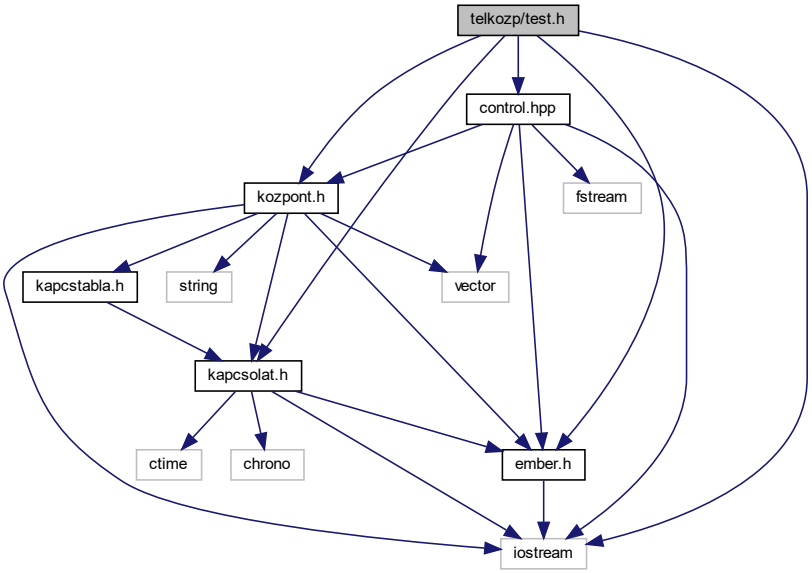


4.11. telkozp/test.h fájlreferencia

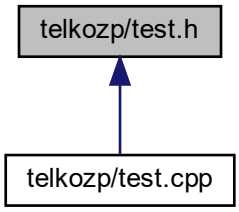
: tartalmazza a teszt függvény deklarációját

```
#include <iostream>
#include "kozpont.h"
#include "ember.h"
#include "kapcsolat.h"
#include "control.hpp"
```

A test.h definíciós fájl függési gráfja:



Ez az ábra azt mutatja, hogy mely fájlok ágyazzák be közvetve vagy közvetlenül ezt a fájlt:



Függvények

- bool test_main ()
  A tesztprogram.

4.11.1. Részletes leírás

: tartalmazza a teszt függvény deklarációját

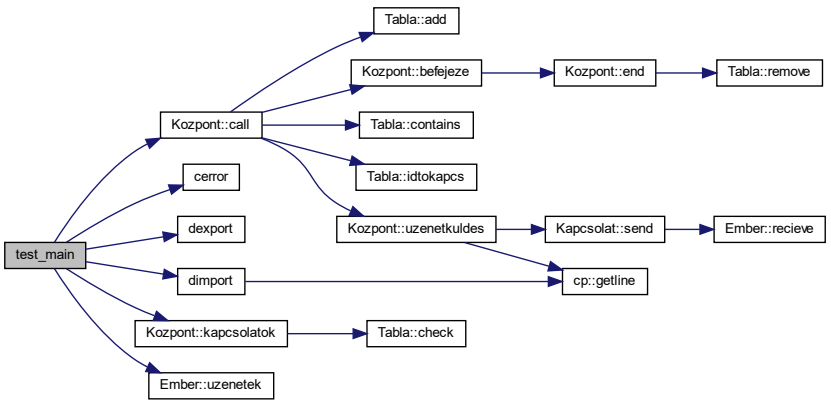
4.11.2. Függvények dokumentációja

4.11.2.1. test_main() `bool test_main ( )`

A tesztprogram.

Definíció a(z) test.cpp fájl 6. sorában.

A függvény hívási gráfja:



Tárgymutató

~Ember
  Ember, 6
~Kapcsolat
  Kapcsolat, 11
~Kozpont
  Kozpont, 15
~Tabla
  Tabla, 23

adatok
  Tabla, 25
add
  Tabla, 23
addnkp
  control.hpp, 28
addnmbr
  control.hpp, 28

befejeze
  Kozpont, 16

call
  Kozpont, 16
cel
  Kapcsolat, 12
cerror
  control.hpp, 28
check
  Tabla, 23
connect
  Kozpont, 17
contains
  Tabla, 23
control.hpp
  addnkp, 28
  addnmbr, 28
  cerror, 28
  dexport, 29
  dimport, 29
  errstr, 31
  kozpinfo, 30
  login, 30
  maincall, 30
cp, 2
  getline, 3

dexport
  control.hpp, 29
dimport
  control.hpp, 29

Ember, 4
  ~Ember, 6
  Ember, 5
  foglalt, 7
  foglalte, 6
  gettel, 6
  maxmsg, 7
  nev, 8
  olvasatlan, 8
  operator<<, 7
  recieve, 6
  setfoglalt, 7
  tarhely, 8
  telszam, 8
  uzenetek, 7
end
  Kozpont, 17
errstr
  control.hpp, 31

foglalt
  Ember, 7
foglalte
  Ember, 6

getid
  Kapcsolat, 11
getkorzet
  Kozpont, 18
getline
  cp, 3
getmaxbe
  Kozpont, 18
  Tabla, 24
getmaxki
  Kozpont, 18
  Tabla, 24
gettel
  Ember, 6

hivo
  Kapcsolat, 12

id
  Kapcsolat, 13
idtokapcs
  Tabla, 24
indb
  Tabla, 25

Kapcsolat, 8
  ~Kapcsolat, 11
  cel, 12
  getid, 11
  hivo, 12
  id, 13
  Kapcsolat, 10
  kezdet, 13
  operator<<, 12
  send, 11
kapcsolatok

Kozpont, 19
kezdet
Kapcsolat, 13
korzet
Kozpont, 20
Tabla, 25
kozpinfo
control.hpp, 30
Kozpont, 13
∼Kozpont, 15
befejeze, 16
call, 16
connect, 17
end, 17
getkorzet, 18
getmaxbe, 18
getmaxki, 18
kapcsolatok, 19
korzet, 20
Kozpont, 15
ktabla, 20
uzenetkuldes, 19
ktabla
Kozpont, 20

login
control.hpp, 30

maincall
control.hpp, 30
maxbe
Tabla, 25
maxki
Tabla, 25
maxmsg
Ember, 7

nev
Ember, 8

olvasatlan
Ember, 8
operator<<
Ember, 7
Kapcsolat, 12
outdb
Tabla, 25

recieve
Ember, 6
remove
Tabla, 24

send
Kapcsolat, 11
setfoglalt
Ember, 7

Tabla, 20
∼Tabla, 23
adatok, 25
add, 23
check, 23
contains, 23
getmaxbe, 24
getmaxki, 24
idtokapcs, 24
indb, 25
korzet, 25
maxbe, 25
maxki, 25
outdb, 25
remove, 24
Tabla, 22
tarhely
Ember, 8
telkozp/control.hpp, 26
telkozp/ember.cpp, 31
telkozp/ember.h, 32
telkozp/kapcsolat.cpp, 33
telkozp/kapcsolat.h, 33
telkozp/kapcstabla.cpp, 35
telkozp/kapcstabla.h, 36
telkozp/kozpont.cpp, 37
telkozp/kozpont.h, 38
telkozp/test.cpp, 39
telkozp/test.h, 41
telszam
Ember, 8
test.cpp
test_main, 40
test.h
test_main, 42
test_main
test.cpp, 40
test.h, 42

uzenetek
Ember, 7
uzenetkuldes
Kozpont, 19