

C++ Software transactional Memory

Zoltan Fuzesi

C00197361 IT Carlow

Software Engineering 2017/2018

STM library with client application

Code to develop the library

Contents

1 C++ Software Transactional Memory	2
1.1 Object Based Software Transactional Memory.	2
1.1.1 Installation of the Shared library on Linux platform.	2
1.1.2 Step 1: Download the archive file.	2
1.1.3 Step 2: Unzip in to the target destination.	2
1.1.4 Step 3: Copy the shared library.	2
1.1.5 Step 4: Achieve the required class hierarchy.	2
1.1.6 Step 5: Create an executable file.	3
1.1.7 Step 6: Transactional Environment.	3
1.1.8 Step 7: Run the application.	3
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	4
3.1 Class List	4
4 File Index	4
4.1 File List	4
5 Class Documentation	6
5.1 AIB Class Reference	6
5.1.1 Detailed Description	9
5.1.2 Constructor & Destructor Documentation	9
5.1.3 Member Function Documentation	11
5.1.4 Member Data Documentation	18
5.2 BANK Class Reference	19
5.2.1 Detailed Description	21
5.2.2 Constructor & Destructor Documentation	21
5.2.3 Member Function Documentation	22
5.3 BOA Class Reference	24

5.3.1	Detailed Description	27
5.3.2	Constructor & Destructor Documentation	27
5.3.3	Member Function Documentation	29
5.3.4	Member Data Documentation	36
5.4	BOI Class Reference	37
5.4.1	Detailed Description	39
5.4.2	Constructor & Destructor Documentation	39
5.4.3	Member Function Documentation	41
5.4.4	Member Data Documentation	48
5.5	CARLOW_W Class Reference	49
5.5.1	Detailed Description	51
5.5.2	Constructor & Destructor Documentation	52
5.5.3	Member Function Documentation	53
5.5.4	Member Data Documentation	62
5.6	CARPHONE_WAREHOUSE Class Reference	64
5.6.1	Detailed Description	66
5.6.2	Constructor & Destructor Documentation	67
5.6.3	Member Function Documentation	68
5.6.4	Member Data Documentation	77
5.7	DUNDALK_W Class Reference	79
5.7.1	Detailed Description	81
5.7.2	Constructor & Destructor Documentation	82
5.7.3	Member Function Documentation	83
5.7.4	Member Data Documentation	92
5.8	KILKENNY_W Class Reference	94
5.8.1	Detailed Description	96
5.8.2	Constructor & Destructor Documentation	97
5.8.3	Member Function Documentation	98
5.8.4	Member Data Documentation	107
5.9	OSTM Class Reference	109

5.9.1	Detailed Description	111
5.9.2	Constructor & Destructor Documentation	111
5.9.3	Member Function Documentation	113
5.9.4	Member Data Documentation	119
5.10	SLIGO_W Class Reference	120
5.10.1	Detailed Description	123
5.10.2	Constructor & Destructor Documentation	124
5.10.3	Member Function Documentation	125
5.10.4	Member Data Documentation	134
5.11	SWBPLC Class Reference	136
5.11.1	Detailed Description	138
5.11.2	Constructor & Destructor Documentation	138
5.11.3	Member Function Documentation	140
5.11.4	Member Data Documentation	146
5.12	TALLAGH_W Class Reference	147
5.12.1	Detailed Description	150
5.12.2	Constructor & Destructor Documentation	151
5.12.3	Member Function Documentation	152
5.12.4	Member Data Documentation	161
5.13	TM Class Reference	163
5.13.1	Detailed Description	164
5.13.2	Constructor & Destructor Documentation	164
5.13.3	Member Function Documentation	164
5.13.4	Member Data Documentation	168
5.14	TX Class Reference	170
5.14.1	Detailed Description	172
5.14.2	Constructor & Destructor Documentation	172
5.14.3	Member Function Documentation	172
5.14.4	Friends And Related Function Documentation	181
5.14.5	Member Data Documentation	181

5.15	ULSTER Class Reference	183
5.15.1	Detailed Description	186
5.15.2	Constructor & Destructor Documentation	186
5.15.3	Member Function Documentation	188
5.15.4	Member Data Documentation	194
5.16	UNBL Class Reference	195
5.16.1	Detailed Description	198
5.16.2	Constructor & Destructor Documentation	198
5.16.3	Member Function Documentation	200
5.16.4	Member Data Documentation	207
5.17	WAREHOUSE Class Reference	208
5.17.1	Detailed Description	210
5.17.2	Constructor & Destructor Documentation	210
5.17.3	Member Function Documentation	211
6	File Documentation	214
6.1	AIB.cpp File Reference	214
6.2	AIB.cpp	215
6.3	AIB.h File Reference	216
6.4	AIB.h	217
6.5	BANK.cpp File Reference	219
6.6	BANK.cpp	219
6.7	BANK.h File Reference	219
6.8	BANK.h	220
6.9	BOA.cpp File Reference	221
6.10	BOA.cpp	221
6.11	BOA.h File Reference	223
6.12	BOA.h	224
6.13	BOI.cpp File Reference	225
6.14	BOI.cpp	225
6.15	BOI.h File Reference	226

6.16 BOI.h	227
6.17 CARLOW_W.cpp File Reference	230
6.18 CARLOW_W.cpp	230
6.19 CARLOW_W.h File Reference	232
6.20 CARLOW_W.h	233
6.21 CARPHONE_WAREHOUSE.cpp File Reference	235
6.22 CARPHONE_WAREHOUSE.cpp	235
6.23 CARPHONE_WAREHOUSE.h File Reference	237
6.24 CARPHONE_WAREHOUSE.h	238
6.25 DUNDALK_W.cpp File Reference	239
6.26 DUNDALK_W.cpp	240
6.27 DUNDALK_W.h File Reference	242
6.28 DUNDALK_W.h	243
6.29 KILKENNY_W.cpp File Reference	244
6.30 KILKENNY_W.cpp	245
6.31 KILKENNY_W.h File Reference	247
6.32 KILKENNY_W.h	248
6.33 main.cpp File Reference	249
6.33.1 Function Documentation	250
6.33.2 Variable Documentation	266
6.34 main.cpp	266
6.35 OSTM.cpp File Reference	274
6.36 OSTM.cpp	275
6.37 OSTM.h File Reference	276
6.38 OSTM.h	277
6.39 SLIGO_W.cpp File Reference	277
6.40 SLIGO_W.cpp	278
6.41 SLIGO_W.h File Reference	280
6.42 SLIGO_W.h	281
6.43 SWBPLC.cpp File Reference	282

6.44 SWBPLC.cpp	282
6.45 SWBPLC.h File Reference	284
6.46 SWBPLC.h	285
6.47 TALLAGH_W.cpp File Reference	286
6.48 TALLAGH_W.cpp	286
6.49 TALLAGH_W.h File Reference	288
6.50 TALLAGH_W.h	289
6.51 TM.cpp File Reference	290
6.52 TM.cpp	291
6.53 TM.h File Reference	292
6.54 TM.h	293
6.55 TX.cpp File Reference	294
6.56 TX.cpp	294
6.57 TX.h File Reference	298
6.58 TX.h	299
6.59 ULSTER.cpp File Reference	300
6.60 ULSTER.cpp	300
6.61 ULSTER.h File Reference	302
6.62 ULSTER.h	303
6.63 UNBL.cpp File Reference	304
6.64 UNBL.cpp	304
6.65 UNBL.h File Reference	305
6.66 UNBL.h	306
6.67 WAREHOUSE.cpp File Reference	307
6.68 WAREHOUSE.cpp	308
6.69 WAREHOUSE.h File Reference	308
6.70 WAREHOUSE.h	309

1 C++ Software Transactional Memory

File: [TM.h](#) Author: Zoltan Fuzesi C00197361, IT Carlow, Software Engineering,

Supervisor : Joe Kehoe,

C++ Software Transactional Memory,

Created on December 18, 2017, 2:09 PM Transaction Manager class fields and methods declarations

1.1 Object Based Software Transactional Memory.

[OSTM](#) is a polymorphic solution to store and manage shared memory spaces within c++ programming context. You can store and managed any kind of object in transactional environment as a shared and protected memory space, if your class inherited from the [OSTM](#) base class, and follows the required steps.

1.1.1 Installation of the Shared library on Linux platform.

Download the zip file from the provided (Windows, Linux, MAC OSX)link in the web-site, that contains the libostm.↔ so, [TM.h](#), [TX.h](#), [OSTM.h](#) files.Unzip the archive file to the desired destination possibly where you program is stored. Copy the library (Shared, Static) to the destination directory. Implement the inheritance from the base class. Create an executable, and run the application.

1.1.2 Step 1: Download the archive file.

Go to the website [Tutorial](#) and download the library to the required operating system platform. (Linux, Windows, Mac OSX)

1.1.3 Step 2: Unzip in to the target destination.

Unzip the downloaded rar file. You can find the Shared, Static library and the *.h files in the unzipped folder. Copy the *.h files to the same folder where is the other C++ files are stored.

1.1.4 Step 3: Copy the shared library.

The Shared library is a libostm.so file, that you need copy to the operating system directory where the other shared library are stored. It will be different destination folder on different platforms. (Linux, Windows, Mac OS) [More Information](#)

1.1.5 Step 4: Achieve the required class hierarchy.

To achieve the required class hierachy between the [OSTM](#) library and your own class structure, you need to implement few steps to inherite from the [OSTM](#) base class. Go to website [Tutorial](#) for more details. Details and instruction of class hierarchy requirements can be found on the web-site. www.serversite.info/ostm

1.1.6 Step 5: Create an executable file.

You can create an executable file using the provided Makefile as you linking together the library (libostm.so), and the *.h files with your own files.

1.1.7 Step 6: Transactional Environment.

Now your application use transactional environment, that guarantees the consistency between object transactions.

1.1.8 Step 7: Run the application.

Go to the directory where the executable was created, and used the following line in the terminal to run the application : ./EXECUTABLE_NAME

Abbreviation for bank names used in the test cases:

BOA - Bank of America

ULSTER - Ulster Bank

UNBL - United National Bank Limited

SWBPLC - Scottish Windows Bank PLC

AIB - Allied Irish Bank

BOI - Bank of Ireland

2 Hierarchical Index

2.1 Class Hierarchy

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

OSTM	109
BANK	19
AIB	6
BOA	24
BOI	37
SWBPLC	136
ULSTER	183
UNBL	195
WAREHOUSE	208
CARLOW_W	49
CARPHONE_WAREHOUSE	64
DUNDALK_W	79

KILKENNY_W	94
SLIGO_W	120
TALLAGH_W	147
TM	163
TX	170

3 Class Index

3.1 Class List

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

AIB	6
BANK	19
BOA	24
BOI	37
CARLOW_W	49
CARPHONE_WAREHOUSE	64
DUNDALK_W	79
KILKENNY_W	94
OSTM	109
SLIGO_W	120
SWBPLC	136
TALLAGH_W	147
TM	163
TX	170
ULSTER	183
UNBL	195
WAREHOUSE	208

4 File Index

4.1 File List

Here is a list of all files with brief descriptions:

AIB.cpp	214
AIB.h	216
BANK.cpp	219
BANK.h	219
BOA.cpp	221
BOA.h	223
BOI.cpp	225
BOI.h	226
CARLOW_W.cpp	230
CARLOW_W.h	232
CARPHONE_WAREHOUSE.cpp	235
CARPHONE_WAREHOUSE.h	237
DUNDALK_W.cpp	239
DUNDALK_W.h	242
KILKENNY_W.cpp	244
KILKENNY_W.h	247
main.cpp	249
OSTM.cpp	274
OSTM.h	276
SLIGO_W.cpp	277
SLIGO_W.h	280
SWBPLC.cpp	282
SWBPLC.h	284
TALLAGH_W.cpp	286
TALLAGH_W.h	288
TM.cpp	290
TM.h	292
TX.cpp	294
TX.h	298
ULSTER.cpp	300
ULSTER.h	302
UNBL.cpp	304

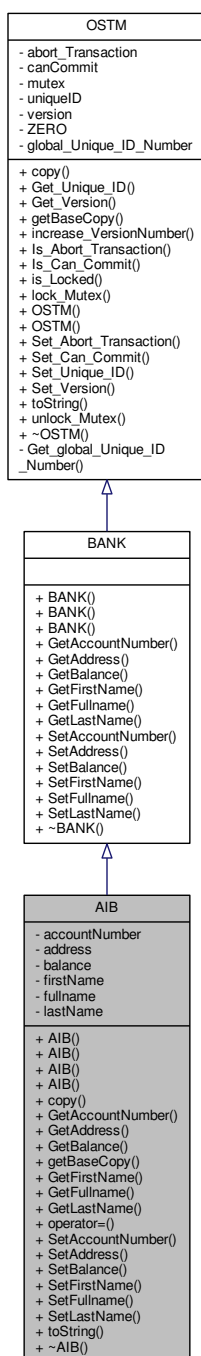
UNBL.h	305
WAREHOUSE.cpp	307
WAREHOUSE.h	308

5 Class Documentation

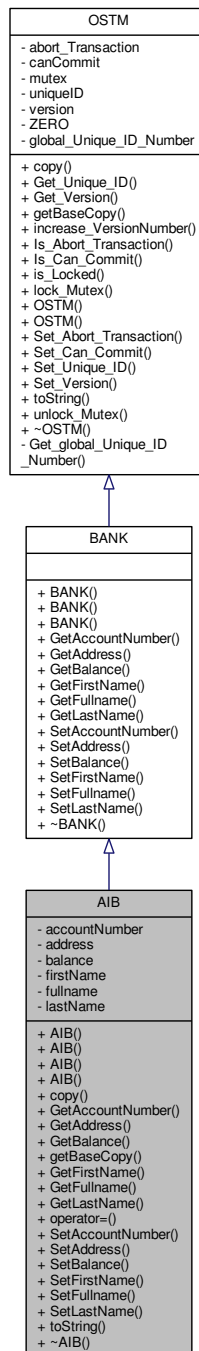
5.1 AIB Class Reference

```
#include <AIB.h>
```

Inheritance diagram for AIB:



Collaboration diagram for AIB:



Public Member Functions

- **AIB** ()
- **AIB** (int `accountNumber`, double `balance`, std::string `firstName`, std::string `lastName`, std::string `address`)
- **AIB** (std::shared_ptr< **BANK** > `obj`, int `_version`, int `_unique_id`)
- **AIB** (const **AIB** &`orig`)
- virtual void `copy` (std::shared_ptr< **OSTM** > `to`, std::shared_ptr< **OSTM** > `from`)

copy function, make deep copy of the object/pointer

- virtual int [GetAccountNumber](#) () const
- virtual std::string [GetAddress](#) () const
- virtual double [GetBalance](#) () const
- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)

getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object

- virtual std::string [GetFirstName](#) () const
- virtual std::string [GetFullname](#) () const
- virtual std::string [GetLastName](#) () const
- [AIB operator=](#) (const [AIB](#) &orig)
- virtual void [SetAccountNumber](#) (int [accountNumber](#))
- virtual void [SetAddress](#) (std::string [address](#))
- virtual void [SetBalance](#) (double [balance](#))
- virtual void [SetFirstName](#) (std::string [firstName](#))
- virtual void [SetFullname](#) (std::string [fullname](#))
- virtual void [SetLastName](#) (std::string [lastName](#))
- virtual void [toString](#) ()

_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type

- virtual [~AIB](#) ()

Private Attributes

- int [accountNumber](#)
- std::string [address](#)
- double [balance](#)
- std::string [firstName](#)
- std::string [fullname](#)
- std::string [lastName](#)

5.1.1 Detailed Description

Inherit from [BANK](#)

Definition at line 18 of file [AIB.h](#).

5.1.2 Constructor & Destructor Documentation

5.1.2.1 [AIB::AIB](#) () [inline]

Constructor

Definition at line 23 of file [AIB.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

Referenced by [AIB\(\)](#), and [getBaseCopy\(\)](#).

```
00023         : BANK()
00024     {
00025         this->accountNumber = 0;
00026         this->balance = 50;
00027         this->firstName = "Joe";
00028         this->lastName = "Blog";
00029         this->address = "High street, Carlow";
00030         this->fullname = firstName + " " + lastName;
00031     };
00032 }
```

5.1.2.2 `AIB::AIB (int accountNumber, double balance, std::string firstName, std::string lastName, std::string address)`
`[inline]`

Custom constructor

Definition at line 36 of file [AIB.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

```
00036                                     :
00037         BANK()
00038     {
00039         this->accountNumber = accountNumber;
00039         this->balance = balance;
00040         this->firstName = firstName;
00041         this->lastName = lastName;
00042         this->address = address;
00043         this->fullname = firstName + " " + lastName;
00044     };
```

5.1.2.3 `AIB::AIB (std::shared_ptr< BANK > obj, int _version, int _unique_id)` `[inline]`

Custom constructor, used by the library for deep copying

Definition at line 48 of file [AIB.h](#).

References [accountNumber](#), [address](#), [AIB\(\)](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

```
00048                                     : BANK(_version, _unique_id)
00049     {
00050
00051         this->accountNumber = obj->GetAccountNumber();
00052         this->balance = obj->GetBalance();
00053         this->firstName = obj->GetFirstName();
00054         this->lastName = obj->GetLastName();
00055         this->address = obj->GetAddress();
00056         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00057
00058     };
```

Here is the call graph for this function:



5.1.2.4 `AIB::AIB (const AIB & orig)`

Copy constructor

Definition at line 14 of file [AIB.cpp](#).

```
00014     {
00015 }
```


5.1.2.5 AIB::~~AIB () [virtual]

de-constructor

Definition at line 17 of file [AIB.cpp](#).

Referenced by [operator=\(\)](#).

```
00017         {
00018     }
```

5.1.3 Member Function Documentation

5.1.3.1 void AIB::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 37 of file [AIB.cpp](#).

References [OSTM::Set_Unique_ID\(\)](#).

Referenced by [operator=\(\)](#).

```
00037         {
00038
00039     std::shared_ptr<AIB> objTO = std::dynamic_pointer_cast<AIB>(to);
00040     std::shared_ptr<AIB> objFROM = std::dynamic_pointer_cast<AIB>(from);
00041     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00042     objTO->Set_Version(objFROM->Get_Version());
00043     objTO->Set_AccountNumber(objFROM->Get_AccountNumber());
00044     objTO->Set_Balance(objFROM->Get_Balance());
00045 }
```

Here is the call graph for this function:



5.1.3.2 int AIB::GetAccountNumber () const [virtual]

Reimplemented from [BANK](#).

Definition at line 81 of file [AIB.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00081                                     {
00082     return accountNumber;
00083 }
```

5.1.3.3 std::string AIB::GetAddress () const [virtual]

Reimplemented from [BANK](#).

Definition at line 65 of file [AIB.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00065                                     {
00066     return address;
00067 }
```

5.1.3.4 double AIB::GetBalance () const [virtual]

Reimplemented from [BANK](#).

Definition at line 73 of file [AIB.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00073                                     {
00074     return balance;
00075 }
```

5.1.3.5 std::shared_ptr< OSTM > AIB::getBaseCopy (std::shared_ptr< OSTM > object) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a std::shared_ptr<BANK> return type

Reimplemented from [OSTM](#).

Definition at line 24 of file [AIB.cpp](#).

References [AIB\(\)](#).

Referenced by [operator=\(\)](#).

```
00025 {
00026
00027     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00028     std::shared_ptr<BANK> obj(new AIB(objTO, object->Get_Version(), object->Get_Unique_ID()));
00029     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00030     return ostm_obj;
00031 }
```

Here is the call graph for this function:



5.1.3.6 std::string AIB::GetFirstName () const [virtual]

Reimplemented from [BANK](#).

Definition at line 97 of file [AIB.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00097                                     {
00098     return firstName;
00099 }
```

5.1.3.7 std::string AIB::GetFullname () const [virtual]

Reimplemented from [BANK](#).

Definition at line 105 of file [AIB.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00105                                     {
00106     return fullname;
00107 }
```

5.1.3.8 `std::string AIB::GetLastName () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 89 of file [AIB.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00089                                     {  
00090     return lastName;  
00091 }
```

5.1.3.9 `AIB AIB::operator= (const AIB & orig) [inline]`

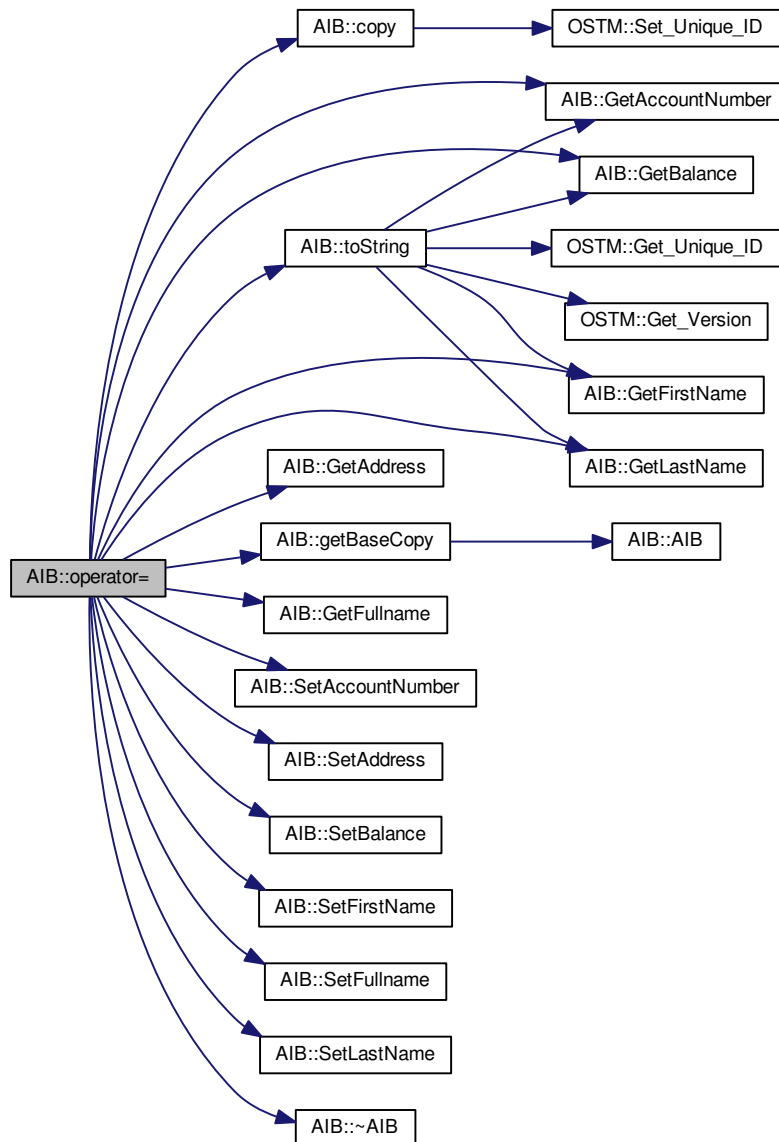
Operator

Definition at line 66 of file [AIB.h](#).

References [accountNumber](#), [address](#), [balance](#), [copy\(\)](#), [firstName](#), [fullName](#), [GetAccountNumber\(\)](#), [GetAddress\(\)](#), [GetBalance\(\)](#), [getBaseCopy\(\)](#), [GetFirstName\(\)](#), [GetFullName\(\)](#), [GetLastName\(\)](#), [lastName](#), [SetAccountNumber\(\)](#), [SetAddress\(\)](#), [SetBalance\(\)](#), [SetFirstName\(\)](#), [SetFullName\(\)](#), [SetLastName\(\)](#), [toString\(\)](#), and [~AIB\(\)](#).

```
00066 {};
```

Here is the call graph for this function:



5.1.3.10 void AIB::SetAccountNumber (int *accountNumber*) [virtual]

Reimplemented from [BANK](#).

Definition at line 77 of file [AIB.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#).

```

00077      {
00078          this->accountNumber = accountNumber;
00079      }

```

5.1.3.11 void AIB::SetAddress (std::string *address*) [virtual]

Reimplemented from [BANK](#).

Definition at line 61 of file [AIB.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00061                                     {  
00062     this->address = address;  
00063 }
```

5.1.3.12 void AIB::SetBalance (double *balance*) [virtual]

Reimplemented from [BANK](#).

Definition at line 69 of file [AIB.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#).

```
00069                                     {  
00070     this->balance = balance;  
00071 }
```

5.1.3.13 void AIB::SetFirstName (std::string *firstName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 93 of file [AIB.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#).

```
00093                                     {  
00094     this->firstName = firstName;  
00095 }
```

5.1.3.14 void AIB::SetFullname (std::string *fullname*) [virtual]

Reimplemented from [BANK](#).

Definition at line 101 of file [AIB.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00101                                     {  
00102     this->fullname = fullname;  
00103 }
```

5.1.3.15 void AIB::SetLastName (std::string *lastName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 85 of file [AIB.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#).

```
00085         {
00086         this->lastName = lastName;
00087     }
```

5.1.3.16 void AIB::toString () [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

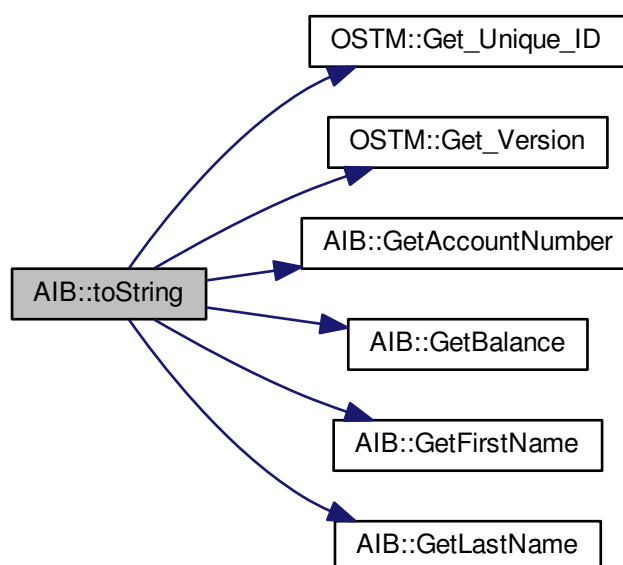
Definition at line 56 of file [AIB.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetAccountNumber\(\)](#), [GetBalance\(\)](#), [GetFirstName\(\)](#), and [GetLastName\(\)](#).

Referenced by [operator=\(\)](#).

```
00057 {
00058     std::cout << "\nAIB BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account : "
    << this->GetAccountNumber() << "\nDouble value : " << this->
    GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
    this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00059 }
```

Here is the call graph for this function:



5.1.4 Member Data Documentation

5.1.4.1 `int AIB::accountNumber` `[private]`

Definition at line 100 of file [AIB.h](#).

Referenced by [AIB\(\)](#), [GetAccountNumber\(\)](#), [operator=\(\)](#), and [SetAccountNumber\(\)](#).

5.1.4.2 `std::string AIB::address` `[private]`

Definition at line 102 of file [AIB.h](#).

Referenced by [AIB\(\)](#), [GetAddress\(\)](#), [operator=\(\)](#), and [SetAddress\(\)](#).

5.1.4.3 `double AIB::balance` `[private]`

Definition at line 101 of file [AIB.h](#).

Referenced by [AIB\(\)](#), [GetBalance\(\)](#), [operator=\(\)](#), and [SetBalance\(\)](#).

5.1.4.4 `std::string AIB::firstName` `[private]`

Definition at line 98 of file [AIB.h](#).

Referenced by [AIB\(\)](#), [GetFirstName\(\)](#), [operator=\(\)](#), and [SetFirstName\(\)](#).

5.1.4.5 `std::string AIB::fullName` `[private]`

Definition at line 97 of file [AIB.h](#).

Referenced by [AIB\(\)](#), [GetFullName\(\)](#), [operator=\(\)](#), and [SetFullName\(\)](#).

5.1.4.6 `std::string AIB::lastName` `[private]`

Definition at line 99 of file [AIB.h](#).

Referenced by [AIB\(\)](#), [GetLastName\(\)](#), [operator=\(\)](#), and [SetLastName\(\)](#).

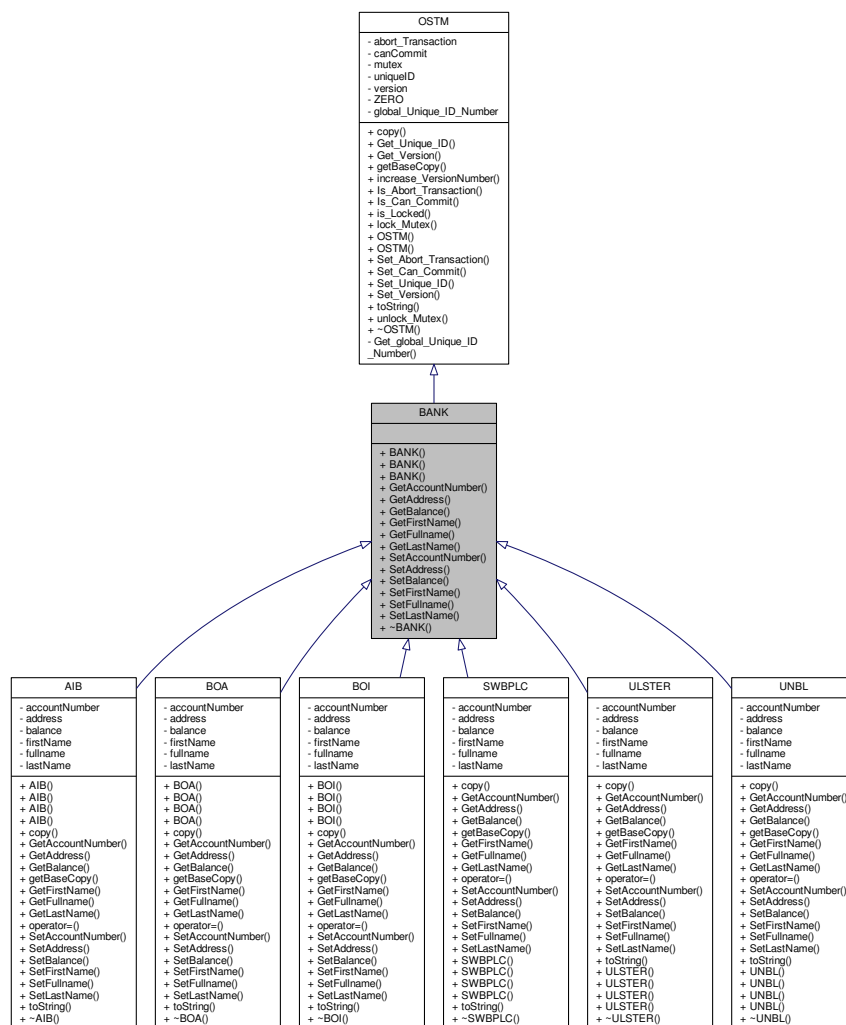
The documentation for this class was generated from the following files:

- [AIB.h](#)
- [AIB.cpp](#)

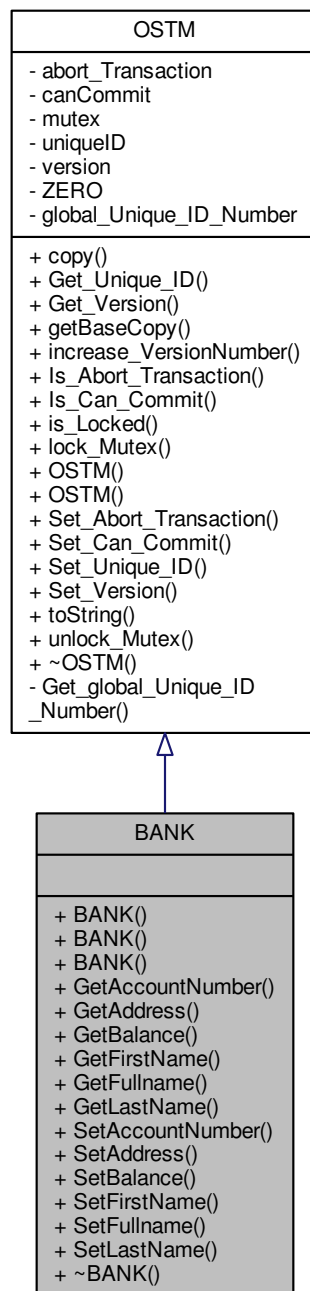
5.2 BANK Class Reference

```
#include <BANK.h>
```

Inheritance diagram for BANK:



Collaboration diagram for BANK:



Public Member Functions

- [BANK](#) ()
- [BANK](#) (int _version, int _unique_id)
- [BANK](#) (const [BANK](#) &orig)
- virtual int [GetAccountNumber](#) () const
- virtual std::string [GetAddress](#) () const

- virtual double [GetBalance](#) () const
- virtual std::string [GetFirstName](#) () const
- virtual std::string [GetFullname](#) () const
- virtual std::string [GetLastName](#) () const
- virtual void [SetAccountNumber](#) (int accountNumber)
- virtual void [SetAddress](#) (std::string address)
- virtual void [SetBalance](#) (double balance)
- virtual void [SetFirstName](#) (std::string firstName)
- virtual void [SetFullname](#) (std::string fullname)
- virtual void [SetLastName](#) (std::string lastName)
- virtual [~BANK](#) ()

5.2.1 Detailed Description

[BANK](#) inherit from the [OSTM](#) library. It is declares the common functions in the child classes as a virtual function.

Definition at line 16 of file [BANK.h](#).

5.2.2 Constructor & Destructor Documentation

5.2.2.1 [BANK::BANK](#) () [\[inline\]](#)

Constructor

Definition at line 23 of file [BANK.h](#).

Referenced by [BANK\(\)](#).

```
00023         : OSTM() {
00024
00025     };
```

5.2.2.2 [BANK::BANK](#) (int *_version*, int *_unique_id*) [\[inline\]](#)

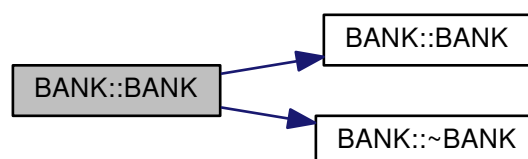
Custom Constructor

Definition at line 29 of file [BANK.h](#).

References [BANK\(\)](#), and [~BANK\(\)](#).

```
00029                                     : OSTM(_version, _unique_id){
00030
00031     };
```

Here is the call graph for this function:



5.2.2.3 `BANK::BANK (const BANK & orig)`

Copy constructor

Definition at line 11 of file [BANK.cpp](#).

```
00011             {
00012 }
```

5.2.2.4 `BANK::~BANK () [virtual]`

de-constructor

Definition at line 14 of file [BANK.cpp](#).

Referenced by [BANK\(\)](#).

```
00014             {
00015 }
```

5.2.3 Member Function Documentation

5.2.3.1 `virtual int BANK::GetAccountNumber () const [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 49 of file [BANK.h](#).

```
00049 {};
```

5.2.3.2 `virtual std::string BANK::GetAddress () const [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 45 of file [BANK.h](#).

```
00045 {};
```

5.2.3.3 `virtual double BANK::GetBalance () const [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 47 of file [BANK.h](#).

```
00047 {};
```

5.2.3.4 `virtual std::string BANK::GetFirstName () const [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 53 of file [BANK.h](#).

```
00053 {};
```

5.2.3.5 `virtual std::string BANK::GetFullName () const [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 55 of file [BANK.h](#).

```
00055 {};
```

5.2.3.6 `virtual std::string BANK::GetLastName () const [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 51 of file [BANK.h](#).

```
00051 {};
```

5.2.3.7 `virtual void BANK::SetAccountNumber (int accountNumber) [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 48 of file [BANK.h](#).

```
00048 {};
```

5.2.3.8 `virtual void BANK::SetAddress (std::string address) [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 44 of file [BANK.h](#).

```
00044 {};
```

5.2.3.9 `virtual void BANK::SetBalance (double balance) [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 46 of file [BANK.h](#).

Referenced by [_complex_transfer_\(\)](#), [_nesting_\(\)](#), [_six_account_transfer_\(\)](#), and [_two_account_transfer_\(\)](#).

```
00046 {};
```

5.2.3.10 `virtual void BANK::SetFirstName (std::string firstName) [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 52 of file [BANK.h](#).

```
00052 {};
```

5.2.3.11 `virtual void BANK::SetFullName (std::string fullName) [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 54 of file [BANK.h](#).

```
00054 {};
```

5.2.3.12 `virtual void BANK::SetLastName (std::string lastName) [inline],[virtual]`

Reimplemented in [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 50 of file [BANK.h](#).

```
00050 {};
```

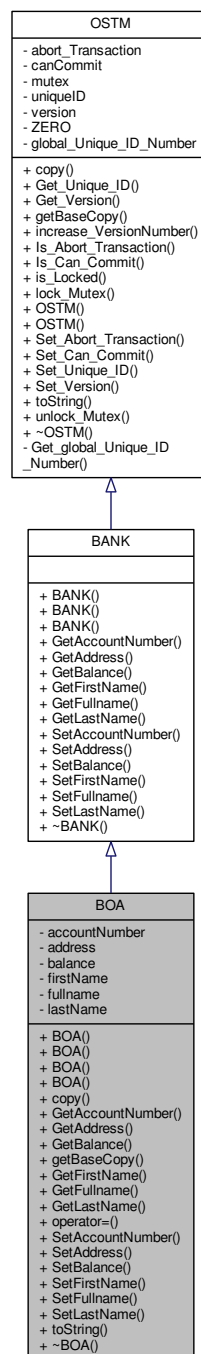
The documentation for this class was generated from the following files:

- [BANK.h](#)
- [BANK.cpp](#)

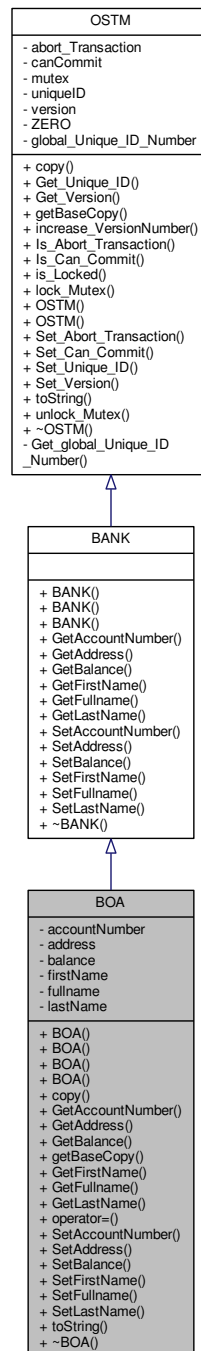
5.3 BOA Class Reference

```
#include <BOA.h>
```

Inheritance diagram for BOA:



Collaboration diagram for BOA:



Public Member Functions

- `BOA ()`
- `BOA (int accountNumber, double balance, std::string firstName, std::string lastName, std::string address)`
- `BOA (std::shared_ptr< BANK > obj, int _version, int _unique_id)`
- `BOA (const BOA &orig)`
- `virtual void copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from)`

copy function, make deep copy of the object/pointer

- virtual int [GetAccountNumber](#) () const
- virtual std::string [GetAddress](#) () const
- virtual double [GetBalance](#) () const
- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)

getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object

- virtual std::string [GetFirstName](#) () const
- virtual std::string [GetFullname](#) () const
- virtual std::string [GetLastName](#) () const
- [BOA operator=](#) (const [BOA](#) &orig)
- virtual void [SetAccountNumber](#) (int [accountNumber](#))
- virtual void [SetAddress](#) (std::string [address](#))
- virtual void [SetBalance](#) (double [balance](#))
- virtual void [SetFirstName](#) (std::string [firstName](#))
- virtual void [SetFullname](#) (std::string [fullname](#))
- virtual void [SetLastName](#) (std::string [lastName](#))
- virtual void [toString](#) ()

_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type

- virtual [~BOA](#) ()

Private Attributes

- int [accountNumber](#)
- std::string [address](#)
- double [balance](#)
- std::string [firstName](#)
- std::string [fullname](#)
- std::string [lastName](#)

5.3.1 Detailed Description

Inherit from [BANK](#)

Definition at line 18 of file [BOA.h](#).

5.3.2 Constructor & Destructor Documentation

5.3.2.1 [BOA::BOA](#) () [inline]

Constructor

Definition at line 24 of file [BOA.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

Referenced by [BOA\(\)](#), and [getBaseCopy\(\)](#).

```
00024         : BANK() {
00025             this->accountNumber = 0;
00026             this->balance = 50;
00027             this->firstName = "Joe";
00028             this->lastName = "Blog";
00029             this->address = "High street, Carlow";
00030             this->fullname = firstName + " " + lastName;
00031         };
```

5.3.2.2 BOA::BOA (int *accountNumber*, double *balance*, std::string *firstName*, std::string *lastName*, std::string *address*) [inline]

Custom constructor

Definition at line 35 of file [BOA.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

```
00035                                     :
00036     BANK() {
00037         this->accountNumber = accountNumber;
00038         this->balance = balance;
00039         this->firstName = firstName;
00040         this->lastName = lastName;
00041         this->address = address;
00042         this->fullname = firstName + " " + lastName;
00042     };
```

5.3.2.3 BOA::BOA (std::shared_ptr< BANK > *obj*, int *_version*, int *_unique_id*) [inline]

Custom constructor, used by the library for deep copying

Definition at line 46 of file [BOA.h](#).

References [accountNumber](#), [address](#), [balance](#), [BOA\(\)](#), [firstName](#), [fullname](#), and [lastName](#).

```
00046                                     : BANK(_version, _unique_id) {
00047
00048     this->accountNumber = obj->GetAccountNumber();
00049     this->balance = obj->GetBalance();
00050     this->firstName = obj->GetFirstName();
00051     this->lastName = obj->GetLastName();
00052     this->address = obj->GetAddress();
00053     this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054 };
```

Here is the call graph for this function:



5.3.2.4 BOA::BOA (const BOA & *orig*)

Copy constructor

Definition at line 12 of file [BOA.cpp](#).

```
00012     {
00013 }
```

5.3.2.5 BOA::~BOA () [virtual]

de-constructor

Definition at line 15 of file BOA.cpp.

Referenced by [operator=\(\)](#).

```
00015         {
00016     }
```

5.3.3 Member Function Documentation

5.3.3.1 void BOA::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 34 of file BOA.cpp.

References [OSTM::Set_Unique_ID\(\)](#).

Referenced by [operator=\(\)](#).

```
00034                                     {
00035
00036     std::shared_ptr<BOA> objTO = std::dynamic_pointer_cast<BOA>(to);
00037     std::shared_ptr<BOA> objFROM = std::dynamic_pointer_cast<BOA>(from);
00038     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00039     objTO->Set_Version(objFROM->Get_Version());
00040     objTO->Set_AccountNumber(objFROM->Get_AccountNumber());
00041     objTO->Set_Balance(objFROM->Get_Balance());
00042
00043 }
```

Here is the call graph for this function:



5.3.3.2 `int BOA::GetAccountNumber () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 80 of file [BOA.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00080                                     {
00081     return accountNumber;
00082 }
```

5.3.3.3 `std::string BOA::GetAddress () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 64 of file [BOA.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00064                                     {
00065     return address;
00066 }
```

5.3.3.4 `double BOA::GetBalance () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 72 of file [BOA.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00072                                     {
00073     return balance;
00074 }
```

5.3.3.5 `std::shared_ptr< OSTM > BOA::getBaseCopy (std::shared_ptr< OSTM > object) [virtual]`

`getBaseCopy` function, make deep copy of the object/pointer and Return a new `std::shared_ptr<BANK>` type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a <code>std::shared_ptr<BANK></code> return type

Reimplemented from [OSTM](#).

Definition at line 22 of file [BOA.cpp](#).

References [BOA\(\)](#).

Referenced by [operator=\(\)](#).

```
00023 {
00024     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00025     std::shared_ptr<BANK> obj(new BOA(objTO, object->Get_Version(), object->Get_Unique_ID()));
00026     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00027     return ostm_obj;
00028 }
```

Here is the call graph for this function:



5.3.3.6 `std::string BOA::GetFirstName () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 96 of file [BOA.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00096                                     {
00097     return firstName;
00098 }
```

5.3.3.7 `std::string BOA::GetFullname () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 104 of file [BOA.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00104                                     {
00105     return fullname;
00106 }
```

5.3.3.8 `std::string BOA::GetLastName () const` `[virtual]`

Reimplemented from [BANK](#).

Definition at line 88 of file [BOA.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00088                                     {  
00089     return lastName;  
00090 }
```

5.3.3.9 `BOA BOA::operator= (const BOA & orig)` `[inline]`

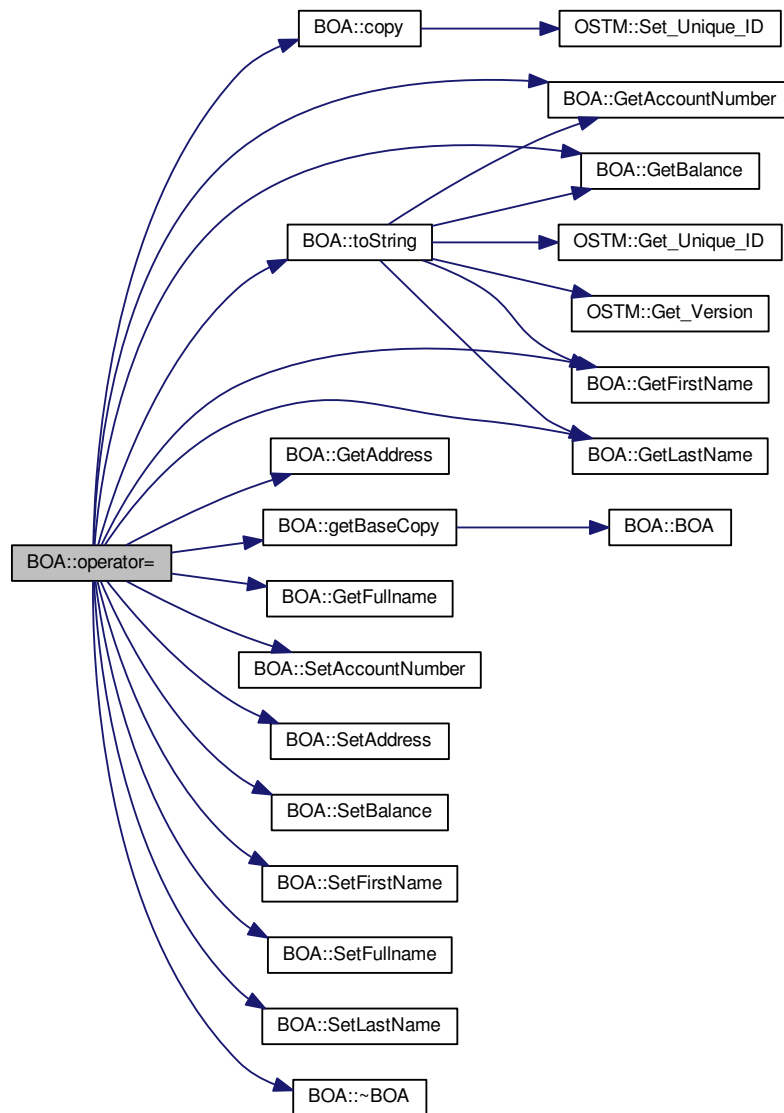
Operator

Definition at line 64 of file [BOA.h](#).

References [accountNumber](#), [address](#), [balance](#), [copy\(\)](#), [firstName](#), [fullname](#), [GetAccountNumber\(\)](#), [GetAddress\(\)](#), [GetBalance\(\)](#), [getBaseCopy\(\)](#), [GetFirstName\(\)](#), [GetFullname\(\)](#), [GetLastName\(\)](#), [lastName](#), [SetAccountNumber\(\)](#), [SetAddress\(\)](#), [SetBalance\(\)](#), [SetFirstName\(\)](#), [SetFullname\(\)](#), [SetLastName\(\)](#), [toString\(\)](#), and [~BOA\(\)](#).

```
00064                                     {  
00065     };
```

Here is the call graph for this function:



5.3.3.10 void BOA::SetAccountNumber (int *accountNumber*) [virtual]

Reimplemented from [BANK](#).

Definition at line 76 of file [BOA.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#).

```

00076      {
00077          this->accountNumber = accountNumber;
00078      }
  
```

5.3.3.11 void BOA::SetAddress (std::string *address*) [virtual]

Reimplemented from [BANK](#).

Definition at line 60 of file [BOA.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00060                                     {
00061     this->address = address;
00062 }
```

5.3.3.12 void BOA::SetBalance (double *balance*) [virtual]

Reimplemented from [BANK](#).

Definition at line 68 of file [BOA.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#).

```
00068                                     {
00069     this->balance = balance;
00070 }
```

5.3.3.13 void BOA::SetFirstName (std::string *firstName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 92 of file [BOA.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#).

```
00092                                     {
00093     this->firstName = firstName;
00094 }
```

5.3.3.14 void BOA::SetFullname (std::string *fullname*) [virtual]

Reimplemented from [BANK](#).

Definition at line 100 of file [BOA.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00100                                     {
00101     this->fullname = fullname;
00102 }
```


5.3.3.15 void BOA::SetLastName (std::string *lastName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 84 of file [BOA.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#).

```
00084 {
00085     this->lastName = lastName;
00086 }
```

5.3.3.16 void BOA::toString () [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

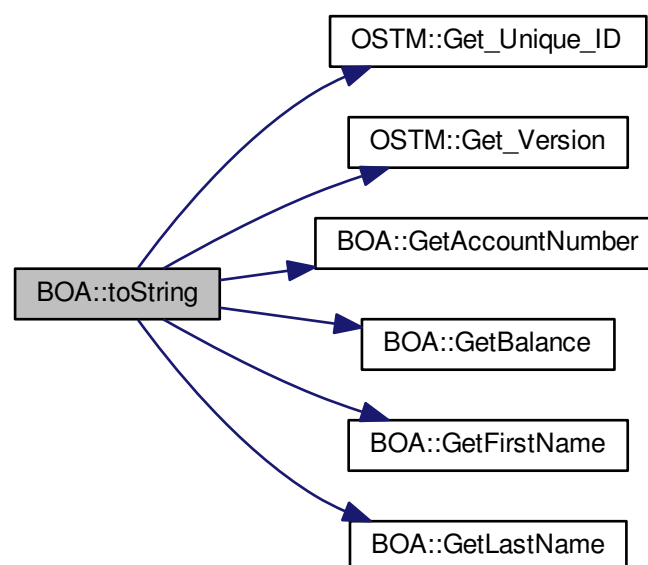
Definition at line 54 of file [BOA.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetAccountNumber\(\)](#), [GetBalance\(\)](#), [GetFirstName\(\)](#), and [GetLastName\(\)](#).

Referenced by [operator=\(\)](#).

```
00055 {
00056     // std::cout << "\nUnique ID : " << this->GetUniqueID() << "\nInt value : " << this->GetV_int() <<
    "\nDouble value : " << this->GetV_double() << "\nFloat value : " << this->GetV_float() << "\nString value : " <<
    this->GetV_string() << "\nVersion number : " << this->GetVersion() << "\nLoad Counter : "<<
    this->GetLoadCounter() << "\nWrite Counter : "<< this->GetWriteCounter() << std::endl;
00057     std::cout << "\nBOA BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account
    : " << this->GetAccountNumber() << "\nDouble value : " << this->
    GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
    this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00058 }
```

Here is the call graph for this function:



5.3.4 Member Data Documentation

5.3.4.1 `int BOA::accountNumber` `[private]`

Definition at line 98 of file [BOA.h](#).

Referenced by [BOA\(\)](#), [GetAccountNumber\(\)](#), [operator=\(\)](#), and [SetAccountNumber\(\)](#).

5.3.4.2 `std::string BOA::address` `[private]`

Definition at line 100 of file [BOA.h](#).

Referenced by [BOA\(\)](#), [GetAddress\(\)](#), [operator=\(\)](#), and [SetAddress\(\)](#).

5.3.4.3 `double BOA::balance` `[private]`

Definition at line 99 of file [BOA.h](#).

Referenced by [BOA\(\)](#), [GetBalance\(\)](#), [operator=\(\)](#), and [SetBalance\(\)](#).

5.3.4.4 `std::string BOA::firstName` `[private]`

Definition at line 96 of file [BOA.h](#).

Referenced by [BOA\(\)](#), [GetFirstName\(\)](#), [operator=\(\)](#), and [SetFirstName\(\)](#).

5.3.4.5 `std::string BOA::fullname` `[private]`

Definition at line 95 of file [BOA.h](#).

Referenced by [BOA\(\)](#), [GetFullname\(\)](#), [operator=\(\)](#), and [SetFullname\(\)](#).

5.3.4.6 `std::string BOA::lastName` `[private]`

Definition at line 97 of file [BOA.h](#).

Referenced by [BOA\(\)](#), [GetLastName\(\)](#), [operator=\(\)](#), and [SetLastName\(\)](#).

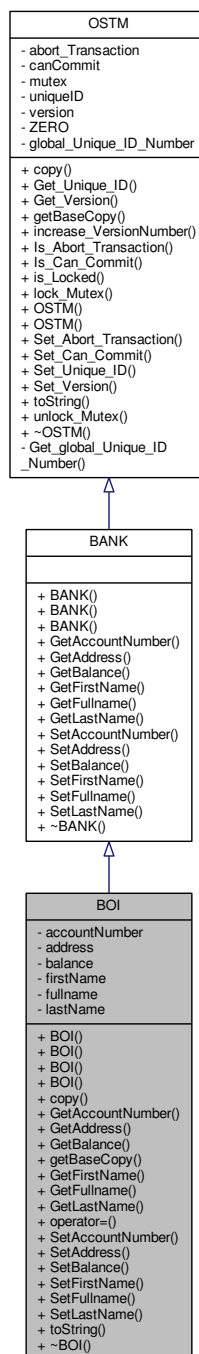
The documentation for this class was generated from the following files:

- [BOA.h](#)
- [BOA.cpp](#)

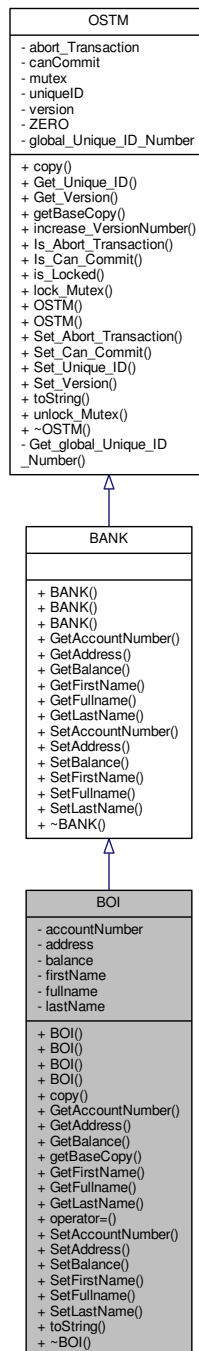
5.4 BOI Class Reference

```
#include <BOI.h>
```

Inheritance diagram for BOI:



Collaboration diagram for BOI:



Public Member Functions

- **BOI** ()
- **BOI** (int **accountNumber**, double **balance**, std::string **firstName**, std::string **lastName**, std::string **address**)
- **BOI** (std::shared_ptr< **BOI** > obj, int **_version**, int **_unique_id**)
- **BOI** (const **BOI** &orig)
- virtual void **copy** (std::shared_ptr< **OSTM** > to, std::shared_ptr< **OSTM** > from)

copy function, make deep copy of the object/pointer

- virtual int [GetAccountNumber](#) () const
- virtual std::string [GetAddress](#) () const
- virtual double [GetBalance](#) () const
- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)

getBaseCopy function, make deep copy of the object/pointer and Return a new BANK type object*

- virtual std::string [GetFirstName](#) () const
- virtual std::string [GetFullname](#) () const
- virtual std::string [GetLastName](#) () const
- [BOI operator=](#) (const [BOI](#) &orig)
- virtual void [SetAccountNumber](#) (int [accountNumber](#))
- virtual void [SetAddress](#) (std::string [address](#))
- virtual void [SetBalance](#) (double [balance](#))
- virtual void [SetFirstName](#) (std::string [firstName](#))
- virtual void [SetFullname](#) (std::string [fullname](#))
- virtual void [SetLastName](#) (std::string [lastName](#))
- virtual void [toString](#) ()

_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type

- virtual [~BOI](#) ()

Private Attributes

- int [accountNumber](#)
- std::string [address](#)
- double [balance](#)
- std::string [firstName](#)
- std::string [fullname](#)
- std::string [lastName](#)

5.4.1 Detailed Description

Inherit from [BANK](#)

Definition at line 19 of file [BOI.h](#).

5.4.2 Constructor & Destructor Documentation

5.4.2.1 [BOI::BOI](#)() [inline]

Constructor

Definition at line 24 of file [BOI.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

Referenced by [BOI\(\)](#), and [getBaseCopy\(\)](#).

```

00024         : BANK()
00025     {
00026         this->accountNumber = 0;
00027         this->balance = 50;
00028         this->firstName = "Joe";
00029         this->lastName = "Blog";
00030         this->address = "High street, Carlow";
00031         this->fullname = firstName + " " + lastName;
00032     }
00033 
```

5.4.2.2 `BOI::BOI(int accountNumber, double balance, std::string firstName, std::string lastName, std::string address)` `[inline]`

Custom constructor

Definition at line 37 of file [BOI.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

```
00037                                     :
00038     BANK()
00039     {
00039         this->accountNumber = accountNumber;
00040         this->balance = balance;
00041         this->firstName = firstName;
00042         this->lastName = lastName;
00043         this->address = address;
00044         this->fullname = firstName + " " + lastName;
00045     };
```

5.4.2.3 `BOI::BOI(std::shared_ptr< BOI > obj, int _version, int _unique_id)` `[inline]`

Custom constructor, used by the library for deep copying

Definition at line 49 of file [BOI.h](#).

References [accountNumber](#), [address](#), [balance](#), [BOI\(\)](#), [firstName](#), [fullname](#), and [lastName](#).

```
00049                                     : BANK(_version, _unique_id)
00050     {
00051         this->accountNumber = obj->GetAccountNumber();
00052         this->balance = obj->GetBalance();
00053         this->firstName = obj->GetFirstName();
00054         this->lastName = obj->GetLastName();
00055         this->address = obj->GetAddress();
00056         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00057     };
```

Here is the call graph for this function:



5.4.2.4 `BOI::BOI(const BOI & orig)`

Copy constructor

Definition at line 15 of file [BOI.cpp](#).

```
00015     {
00016 }
```

5.4.2.5 BOI::~BOI() [virtual]

de-constructor

Definition at line 12 of file [BOI.cpp](#).

Referenced by [operator=\(\)](#).

```
00012         {
00013     }
```

5.4.3 Member Function Documentation

5.4.3.1 void BOI::copy (std::shared_ptr< OSTM > *to*, std::shared_ptr< OSTM > *from*) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 35 of file [BOI.cpp](#).

References [OSTM::Set_Unique_ID\(\)](#).

Referenced by [operator=\(\)](#).

```
00035                                     {
00036
00037     std::shared_ptr<BOI> objTO = std::dynamic_pointer_cast<BOI>(to);
00038     std::shared_ptr<BOI> objFROM = std::dynamic_pointer_cast<BOI>(from);
00039     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00040     objTO->Set_Version(objFROM->Get_Version());
00041     objTO->Set_AccountNumber(objFROM->Get_AccountNumber());
00042     objTO->Set_Balance(objFROM->Get_Balance());
00043 }
```

Here is the call graph for this function:



5.4.3.2 int BOI::GetAccountNumber () const [virtual]

Reimplemented from [BANK](#).

Definition at line 78 of file [BOI.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00078                                     {
00079     return accountNumber;
00080 }
```

5.4.3.3 std::string BOI::GetAddress () const [virtual]

Reimplemented from [BANK](#).

Definition at line 62 of file [BOI.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00062                                     {
00063     return address;
00064 }
```

5.4.3.4 double BOI::GetBalance () const [virtual]

Reimplemented from [BANK](#).

Definition at line 70 of file [BOI.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00070                                     {
00071     return balance;
00072 }
```

5.4.3.5 std::shared_ptr< OSTM > BOI::getBaseCopy (std::shared_ptr< OSTM > object) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new [BANK*](#) type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a BANK* return type

Reimplemented from [OSTM](#).

Definition at line 22 of file [BOI.cpp](#).

References [BOI\(\)](#).

Referenced by [operator=\(\)](#).

```
00023 {
00024
00025     std::shared_ptr<BOI> objTO = std::dynamic_pointer_cast<BOI>(object);
00026     std::shared_ptr<BOI> obj(new BOI(objTO,object->Get_Version(),object->Get_Unique_ID()));
00027     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00028     return ostm_obj;
00029 }
```

Here is the call graph for this function:



5.4.3.6 std::string BOI::GetFirstName () const [virtual]

Reimplemented from [BANK](#).

Definition at line 94 of file [BOI.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00094                                     {
00095     return firstName;
00096 }
```

5.4.3.7 std::string BOI::GetFullname () const [virtual]

Reimplemented from [BANK](#).

Definition at line 102 of file [BOI.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00102                                     {
00103     return fullname;
00104 }
```

5.4.3.8 `std::string BOI::GetLastName () const` `[virtual]`

Reimplemented from [BANK](#).

Definition at line 86 of file [BOI.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00086                                     {  
00087     return lastName;  
00088 }
```

5.4.3.9 `BOI BOI::operator= (const BOI & orig)` `[inline]`

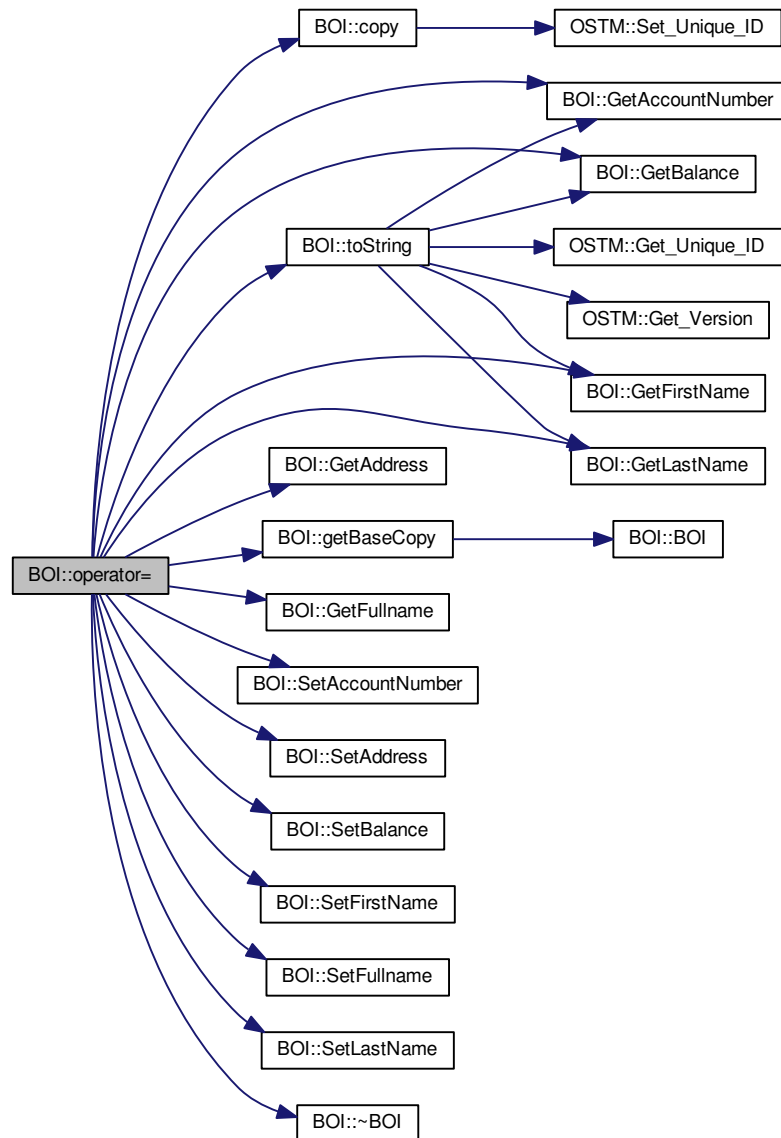
Operator

Definition at line 65 of file [BOI.h](#).

References [accountNumber](#), [address](#), [balance](#), [copy\(\)](#), [firstName](#), [fullname](#), [GetAccountNumber\(\)](#), [GetAddress\(\)](#), [GetBalance\(\)](#), [getBaseCopy\(\)](#), [GetFirstName\(\)](#), [GetFullname\(\)](#), [GetLastName\(\)](#), [lastName](#), [SetAccountNumber\(\)](#), [SetAddress\(\)](#), [SetBalance\(\)](#), [SetFirstName\(\)](#), [SetFullname\(\)](#), [SetLastName\(\)](#), [toString\(\)](#), and [~BOI\(\)](#).

```
00065 {};
```

Here is the call graph for this function:



5.4.3.10 void BOI::SetAccountNumber (int *accountNumber*) [virtual]

Reimplemented from [BANK](#).

Definition at line 74 of file [BOI.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#).

```

00074      {
00075          this->accountNumber = accountNumber;
00076      }

```

5.4.3.11 void BOI::SetAddress (std::string *address*) [virtual]

Reimplemented from [BANK](#).

Definition at line 58 of file [BOI.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00058                                     {  
00059     this->address = address;  
00060 }
```

5.4.3.12 void BOI::SetBalance (double *balance*) [virtual]

Reimplemented from [BANK](#).

Definition at line 66 of file [BOI.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#).

```
00066                                     {  
00067     this->balance = balance;  
00068 }
```

5.4.3.13 void BOI::SetFirstName (std::string *firstName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 90 of file [BOI.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#).

```
00090                                     {  
00091     this->firstName = firstName;  
00092 }
```

5.4.3.14 void BOI::SetFullname (std::string *fullname*) [virtual]

Reimplemented from [BANK](#).

Definition at line 98 of file [BOI.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00098                                     {  
00099     this->fullname = fullname;  
00100 }
```

5.4.3.15 void BOI::SetLastName (std::string *lastName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 82 of file [BOI.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#).

```
00082         {
00083     this->lastName = lastName;
00084 }
```

5.4.3.16 void BOI::toString () [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

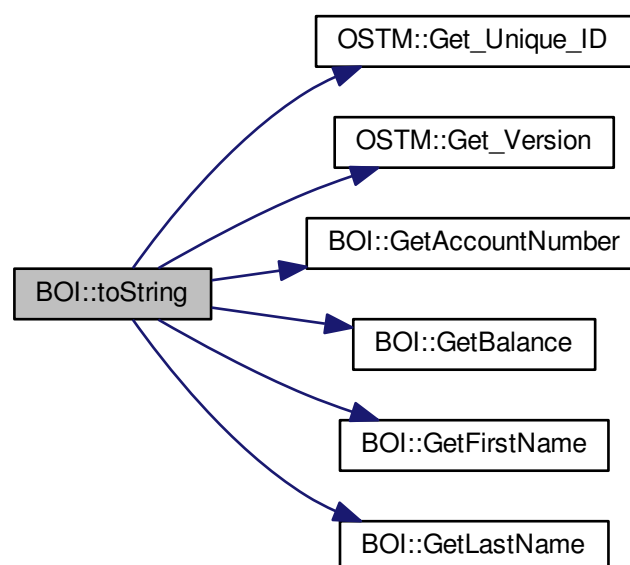
Definition at line 54 of file [BOI.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetAccountNumber\(\)](#), [GetBalance\(\)](#), [GetFirstName\(\)](#), and [GetLastName\(\)](#).

Referenced by [operator=\(\)](#).

```
00055 {
00056     std::cout << "\nBOI BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account : "
    << this->GetAccountNumber() << "\nDouble value : " << this->
    GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
    this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00057 }
```

Here is the call graph for this function:



5.4.4 Member Data Documentation

5.4.4.1 `int BOI::accountNumber` `[private]`

Definition at line 99 of file [BOI.h](#).

Referenced by [BOI\(\)](#), [GetAccountNumber\(\)](#), [operator=\(\)](#), and [SetAccountNumber\(\)](#).

5.4.4.2 `std::string BOI::address` `[private]`

Definition at line 101 of file [BOI.h](#).

Referenced by [BOI\(\)](#), [GetAddress\(\)](#), [operator=\(\)](#), and [SetAddress\(\)](#).

5.4.4.3 `double BOI::balance` `[private]`

Definition at line 100 of file [BOI.h](#).

Referenced by [BOI\(\)](#), [GetBalance\(\)](#), [operator=\(\)](#), and [SetBalance\(\)](#).

5.4.4.4 `std::string BOI::firstName` `[private]`

Definition at line 97 of file [BOI.h](#).

Referenced by [BOI\(\)](#), [GetFirstName\(\)](#), [operator=\(\)](#), and [SetFirstName\(\)](#).

5.4.4.5 `std::string BOI::fullName` `[private]`

Definition at line 96 of file [BOI.h](#).

Referenced by [BOI\(\)](#), [GetFullname\(\)](#), [operator=\(\)](#), and [SetFullname\(\)](#).

5.4.4.6 `std::string BOI::lastName` `[private]`

Definition at line 98 of file [BOI.h](#).

Referenced by [BOI\(\)](#), [GetLastName\(\)](#), [operator=\(\)](#), and [SetLastName\(\)](#).

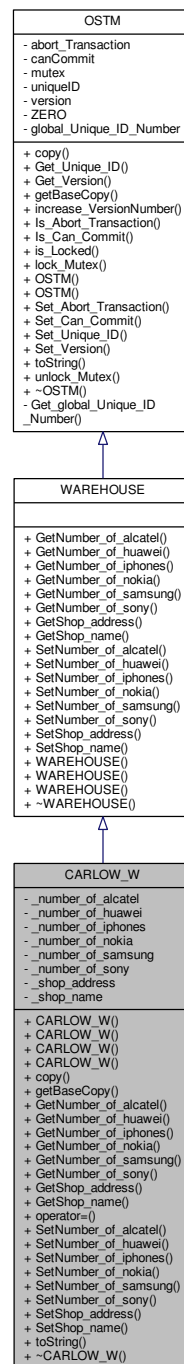
The documentation for this class was generated from the following files:

- [BOI.h](#)
- [BOI.cpp](#)

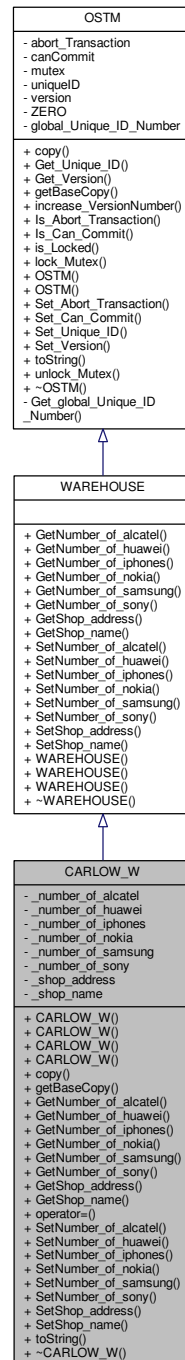
5.5 CARLOW_W Class Reference

```
#include <CARLOW_W.h>
```

Inheritance diagram for CARLOW_W:



Collaboration diagram for CARLOW_W:



Public Member Functions

- [CARLOW_W \(\)](#)
- [CARLOW_W \(std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel\)](#)
- [CARLOW_W \(std::shared_ptr< \[WAREHOUSE\]\(#\) > obj, int _version, int _unique_id\)](#)
- [CARLOW_W \(const \[CARLOW_W\]\(#\) &orig\)](#)

- virtual void [copy](#) (std::shared_ptr< [OSTM](#) > to, std::shared_ptr< [OSTM](#) > from)
copy function, make deep copy of the object/pointer
- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new BANK type object*
- virtual int [GetNumber_of_alcatel](#) ()
- virtual int [GetNumber_of_huawei](#) ()
- virtual int [GetNumber_of_iphones](#) ()
- virtual int [GetNumber_of_nokia](#) ()
- virtual int [GetNumber_of_samsung](#) ()
- virtual int [GetNumber_of_sony](#) ()
- virtual std::string [GetShop_address](#) ()
- virtual std::string [GetShop_name](#) ()
- [CARLOW_W](#) operator= (const [CARLOW_W](#) &orig)
- virtual void [SetNumber_of_alcatel](#) (int _number_of_alcatel)
- virtual void [SetNumber_of_huawei](#) (int _number_of_huawei)
- virtual void [SetNumber_of_iphones](#) (int _number_of_iphones)
- virtual void [SetNumber_of_nokia](#) (int _number_of_nokia)
- virtual void [SetNumber_of_samsung](#) (int _number_of_samsung)
- virtual void [SetNumber_of_sony](#) (int _number_of_sony)
- virtual void [SetShop_address](#) (std::string _shop_address)
- virtual void [SetShop_name](#) (std::string _shop_name)
- virtual void [toString](#) ()
_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type
- virtual [~CARLOW_W](#) ()

Private Attributes

- int [_number_of_alcatel](#)
- int [_number_of_huawei](#)
- int [_number_of_iphones](#)
- int [_number_of_nokia](#)
- int [_number_of_samsung](#)
- int [_number_of_sony](#)
- std::string [_shop_address](#)
- std::string [_shop_name](#)

5.5.1 Detailed Description

Inherit from [WAREHOUSE](#)

Definition at line 19 of file [CARLOW_W.h](#).

5.5.2 Constructor & Destructor Documentation

5.5.2.1 CARLOW_W::CARLOW_W() [inline]

Constructor

Definition at line 24 of file [CARLOW_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

Referenced by [CARLOW_W\(\)](#), and [getBaseCopy\(\)](#).

```

00024         : WAREHOUSE() {
00025
00026         this->_shop_address = "Carlow potato street";
00027         this->_shop_name = "CARLOW C_WAREHOUSE";
00028         this->_number_of_iphones = 200;
00029         this->_number_of_samsung = 200;
00030         this->_number_of_sony = 200;
00031         this->_number_of_huawei = 200;
00032         this->_number_of_nokia = 200;
00033         this->_number_of_alcatel = 200;
00034     };

```

5.5.2.2 CARLOW_W::CARLOW_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel) [inline]

Custom constructor

Definition at line 38 of file [CARLOW_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

```

00038         : WAREHOUSE() {
00039         /*
00040         * copy over values
00041         */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };

```

5.5.2.3 CARLOW_W::CARLOW_W (std::shared_ptr< WAREHOUSE > obj, int _version, int _unique_id) [inline]

Custom constructor, used by the library for deep copying

Definition at line 55 of file [CARLOW_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), and [CARLOW_W\(\)](#).

```

00055                                     :
00056     WAREHOUSE(_version, _unique_id) {
00057         /*
00058          * copy over values
00059          */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }

```

Here is the call graph for this function:



5.5.2.4 CARLOW_W::CARLOW_W (const CARLOW_W & orig)

Copy constructor

Definition at line 17 of file [CARLOW_W.cpp](#).

```

00017                                     {
00018 }

```

5.5.2.5 CARLOW_W::~~CARLOW_W () [virtual]

de-constructor

Definition at line 14 of file [CARLOW_W.cpp](#).

Referenced by [operator=\(\)](#).

```

00014                                     {
00015 }

```

5.5.3 Member Function Documentation

5.5.3.1 void CARLOW_W::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 37 of file [CARLOW_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```

00037                                     {
00038
00039     std::shared_ptr<CARLOW_W> objTO = std::dynamic_pointer_cast<CARLOW_W>(to);
00040     std::shared_ptr<CARLOW_W> objFROM = std::dynamic_pointer_cast<CARLOW_W>(from);
00041     objTO->_shop_address = objFROM->GetShop_address();
00042     objTO->_shop_name = objFROM->GetShop_name();
00043     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00044     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00045     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00046     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00047     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00048     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00049     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00050     objTO->Set_Version(objFROM->Get_Version());
00051
00052
00053 }
```

5.5.3.2 std::shared_ptr<OSTM> CARLOW_W::getBaseCopy (std::shared_ptr<OSTM> *object*) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new BANK* type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a BANK* return type

Reimplemented from [OSTM](#).

Definition at line 24 of file [CARLOW_W.cpp](#).

References [CARLOW_W\(\)](#).

Referenced by [operator=\(\)](#).

```

00025 {
00026
00027     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00028     std::shared_ptr<WAREHOUSE> obj(new CARLOW_W(objTO, object->Get_Version(), object->Get_Unique_ID(
00029 ))) ;
00029     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00030     return ostm_obj;
00031 }
```

Here is the call graph for this function:



5.5.3.3 `int CARLOW_W::GetNumber_of_alcatel () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 75 of file [CARLOW_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00075                                     {  
00076     return _number_of_alcatel;  
00077 }
```

5.5.3.4 `int CARLOW_W::GetNumber_of_huawei () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 91 of file [CARLOW_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00091                                     {  
00092     return _number_of_huawei;  
00093 }
```

5.5.3.5 `int CARLOW_W::GetNumber_of_iphones () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 115 of file [CARLOW_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00115                                     {  
00116     return _number_of_iphones;  
00117 }
```

5.5.3.6 int CARLOW_W::GetNumber_of_nokia () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 83 of file [CARLOW_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00083                                     {  
00084     return _number_of_nokia;  
00085 }
```

5.5.3.7 int CARLOW_W::GetNumber_of_samsung () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 107 of file [CARLOW_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00107                                     {  
00108     return _number_of_samsung;  
00109 }
```

5.5.3.8 int CARLOW_W::GetNumber_of_sony () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 99 of file [CARLOW_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00099                                     {  
00100     return _number_of_sony;  
00101 }
```

5.5.3.9 std::string CARLOW_W::GetShop_address () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 131 of file [CARLOW_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00131                                     {  
00132     return _shop_address;  
00133 }
```

5.5.3.10 std::string CARLOW_W::GetShop_name () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 123 of file [CARLOW_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00123                                     {
00124     return _shop_name;
00125 }
```

5.5.3.11 CARLOW_W CARLOW_W::operator= (const CARLOW_W & orig) [inline]

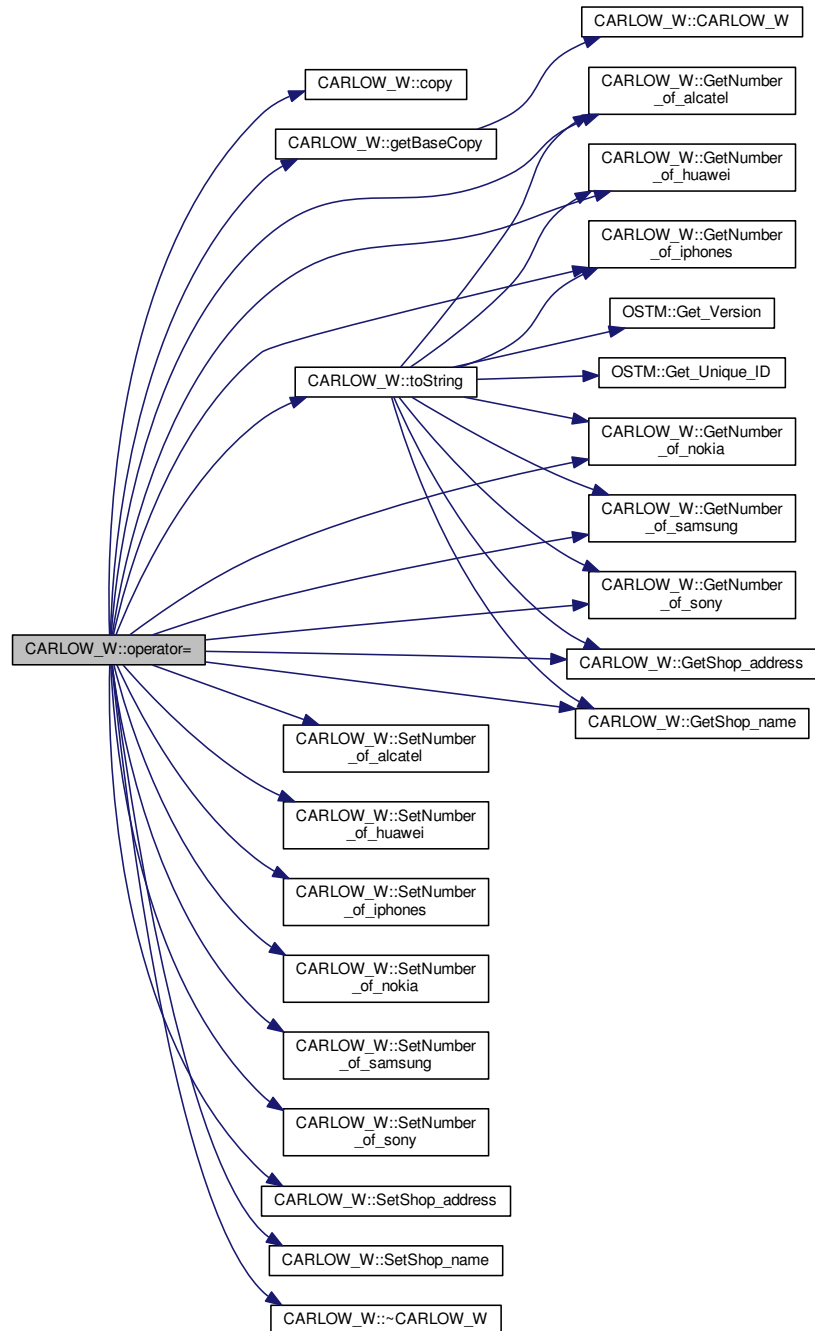
Operator

Definition at line 75 of file [CARLOW_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), [copy\(\)](#), [getBaseCopy\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), [GetShop_name\(\)](#), [SetNumber_of_alcatel\(\)](#), [SetNumber_of_huawei\(\)](#), [SetNumber_of_iphones\(\)](#), [SetNumber_of_nokia\(\)](#), [SetNumber_of_samsung\(\)](#), [SetNumber_of_sony\(\)](#), [SetShop_address\(\)](#), [SetShop_name\(\)](#), [toString\(\)](#), and [~CARLOW_W\(\)](#).

```
00075 {};
```

Here is the call graph for this function:



5.5.3.12 void CARLOW_W::SetNumber_of_alcatel (int _number_of_alcatel) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 71 of file [CARLOW_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#).


```
00071                                     {
00072     this->_number_of_alcatel = _number_of_alcatel;
00073 }
```

5.5.3.13 void CARLOW_W::SetNumber_of_huawei (int *_number_of_huawei*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 87 of file [CARLOW_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#).

```
00087                                     {
00088     this->_number_of_huawei = _number_of_huawei;
00089 }
```

5.5.3.14 void CARLOW_W::SetNumber_of_iphones (int *_number_of_iphones*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 111 of file [CARLOW_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#).

```
00111                                     {
00112     this->_number_of_iphones = _number_of_iphones;
00113 }
```

5.5.3.15 void CARLOW_W::SetNumber_of_nokia (int *_number_of_nokia*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 79 of file [CARLOW_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#).

```
00079                                     {
00080     this->_number_of_nokia = _number_of_nokia;
00081 }
```

5.5.3.16 void CARLOW_W::SetNumber_of_samsung (int *_number_of_samsung*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 103 of file [CARLOW_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#).

```
00103                                     {
00104     this->_number_of_samsung = _number_of_samsung;
00105 }
```

5.5.3.17 void CARLOW_W::SetNumber_of_sony (int *_number_of_sony*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 95 of file [CARLOW_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#).

```
00095                                     {  
00096     this->_number_of_sony = _number_of_sony;  
00097 }
```

5.5.3.18 void CARLOW_W::SetShop_address (std::string *_shop_address*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 127 of file [CARLOW_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```
00127                                     {  
00128     this->_shop_address = _shop_address;  
00129 }
```

5.5.3.19 void CARLOW_W::SetShop_name (std::string *_shop_name*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 119 of file [CARLOW_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#).

```
00119                                     {  
00120     this->_shop_name = _shop_name;  
00121 }
```

5.5.3.20 void CARLOW_W::toString () [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

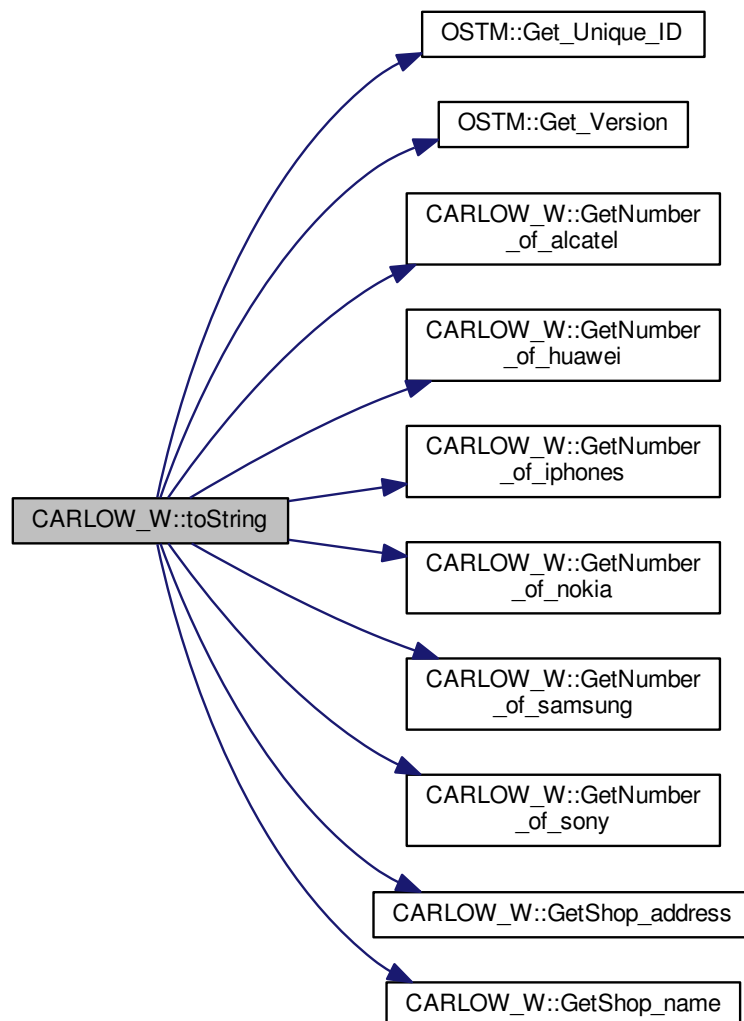
Definition at line 64 of file [CARLOW_W.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), and [GetShop_name\(\)](#).

Referenced by [operator=\(\)](#).

```
00065 {
00066     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
    Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
    << this->GetShop_address() << "\nNo. Iphones : " << this->
    GetNumber_of_iphones() << "\nNo. Samsung : " << this->
    GetNumber_of_samsung() << "\nNo. Sony : " << this->
    GetNumber_of_sony() << "\nNo. Huawei : " << this->
    GetNumber_of_huawei() << "\nNo. Nokia : " << this->
    GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
    GetNumber_of_alcatel() << "\nVersion number : " << this->
    Get_Version() << std::endl;
00067 }
```

Here is the call graph for this function:



5.5.4 Member Data Documentation

5.5.4.1 `int CARLOW_W::_number_of_alcatel` [private]

Definition at line 116 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [GetNumber_of_alcatel\(\)](#), [operator=\(\)](#), and [SetNumber_of_alcatel\(\)](#).

5.5.4.2 `int CARLOW_W::_number_of_huawei` [private]

Definition at line 114 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [GetNumber_of_huawei\(\)](#), [operator=\(\)](#), and [SetNumber_of_huawei\(\)](#).

5.5.4.3 int CARLOW_W::_number_of_iphones [private]

Definition at line 111 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [GetNumber_of_iphones\(\)](#), [operator=\(\)](#), and [SetNumber_of_iphones\(\)](#).

5.5.4.4 int CARLOW_W::_number_of_nokia [private]

Definition at line 115 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [GetNumber_of_nokia\(\)](#), [operator=\(\)](#), and [SetNumber_of_nokia\(\)](#).

5.5.4.5 int CARLOW_W::_number_of_samsung [private]

Definition at line 112 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [GetNumber_of_samsung\(\)](#), [operator=\(\)](#), and [SetNumber_of_samsung\(\)](#).

5.5.4.6 int CARLOW_W::_number_of_sony [private]

Definition at line 113 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [GetNumber_of_sony\(\)](#), [operator=\(\)](#), and [SetNumber_of_sony\(\)](#).

5.5.4.7 std::string CARLOW_W::_shop_address [private]

Definition at line 109 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [copy\(\)](#), [GetShop_address\(\)](#), [operator=\(\)](#), and [SetShop_address\(\)](#).

5.5.4.8 std::string CARLOW_W::_shop_name [private]

Definition at line 110 of file [CARLOW_W.h](#).

Referenced by [CARLOW_W\(\)](#), [GetShop_name\(\)](#), [operator=\(\)](#), and [SetShop_name\(\)](#).

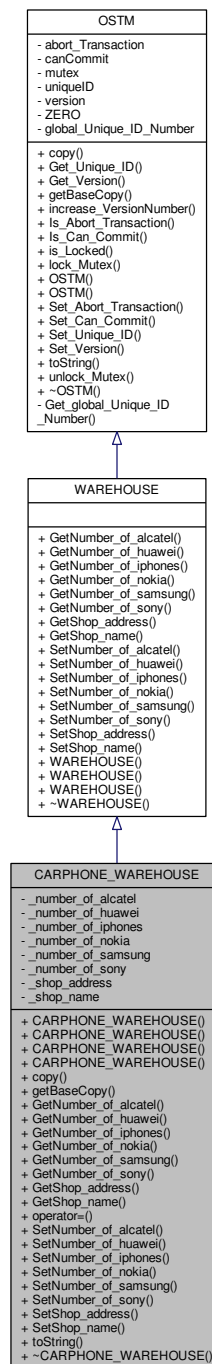
The documentation for this class was generated from the following files:

- [CARLOW_W.h](#)
- [CARLOW_W.cpp](#)

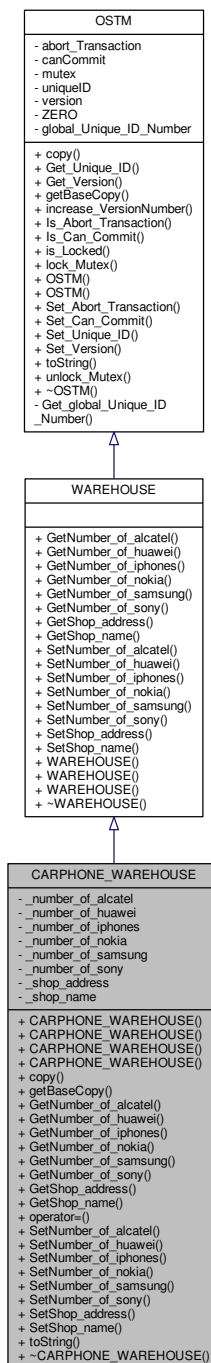
5.6 CARPHONE_WAREHOUSE Class Reference

```
#include <CARPHONE_WAREHOUSE.h>
```

Inheritance diagram for CARPHONE_WAREHOUSE:



Collaboration diagram for CARPHONE_WAREHOUSE:



Public Member Functions

- [CARPHONE_WAREHOUSE](#) ()
- [CARPHONE_WAREHOUSE](#) (std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel)
- [CARPHONE_WAREHOUSE](#) (std::shared_ptr< [WAREHOUSE](#) > obj, int _version, int _unique_id)
- [CARPHONE_WAREHOUSE](#) (const [CARPHONE_WAREHOUSE](#) &orig)

- virtual void `copy` (std::shared_ptr< `OSTM` > to, std::shared_ptr< `OSTM` > from)
copy function, make deep copy of the object/pointer
- virtual std::shared_ptr< `OSTM` > `getBaseCopy` (std::shared_ptr< `OSTM` > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new BANK type object*
- virtual int `GetNumber_of_alcatel` ()
- virtual int `GetNumber_of_huawei` ()
- virtual int `GetNumber_of_iphones` ()
- virtual int `GetNumber_of_nokia` ()
- virtual int `GetNumber_of_samsung` ()
- virtual int `GetNumber_of_sony` ()
- virtual std::string `GetShop_address` ()
- virtual std::string `GetShop_name` ()
- `CARPHONE_WAREHOUSE` operator= (const `CARPHONE_WAREHOUSE` &orig)
- virtual void `SetNumber_of_alcatel` (int `_number_of_alcatel`)
- virtual void `SetNumber_of_huawei` (int `_number_of_huawei`)
- virtual void `SetNumber_of_iphones` (int `_number_of_iphones`)
- virtual void `SetNumber_of_nokia` (int `_number_of_nokia`)
- virtual void `SetNumber_of_samsung` (int `_number_of_samsung`)
- virtual void `SetNumber_of_sony` (int `_number_of_sony`)
- virtual void `SetShop_address` (std::string `_shop_address`)
- virtual void `SetShop_name` (std::string `_shop_name`)
- virtual void `toString` ()
_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type
- virtual `~CARPHONE_WAREHOUSE` ()

Private Attributes

- int `_number_of_alcatel`
- int `_number_of_huawei`
- int `_number_of_iphones`
- int `_number_of_nokia`
- int `_number_of_samsung`
- int `_number_of_sony`
- std::string `_shop_address`
- std::string `_shop_name`

5.6.1 Detailed Description

Inherit from `WAREHOUSE`

Definition at line 19 of file `CARPHONE_WAREHOUSE.h`.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 CARPHONE_WAREHOUSE::CARPHONE_WAREHOUSE () [inline]

Constructor

Definition at line 24 of file [CARPHONE_WAREHOUSE.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), and [getBaseCopy\(\)](#).

```

00024             : WAREHOUSE() {
00025
00026         this->_shop_address = "DUBLIN XII";
00027         this->_shop_name = "DISTRIBUTION CENTER";
00028         this->_number_of_iphones = 10000;
00029         this->_number_of_samsung = 10000;
00030         this->_number_of_sony = 10000;
00031         this->_number_of_huawei = 10000;
00032         this->_number_of_nokia = 10000;
00033         this->_number_of_alcatel = 10000;
00034     };

```

5.6.2.2 CARPHONE_WAREHOUSE::CARPHONE_WAREHOUSE (std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel) [inline]

Custom constructor

Definition at line 38 of file [CARPHONE_WAREHOUSE.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

```

00038             : WAREHOUSE() {
00039
00039         /*
00040         * copy over values
00041
00041         */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };

```

5.6.2.3 CARPHONE_WAREHOUSE::CARPHONE_WAREHOUSE (std::shared_ptr< WAREHOUSE > obj, int _version, int _unique_id) [inline]

Custom constructor, used by the library for deep copying

Definition at line 55 of file [CARPHONE_WAREHOUSE.h](#).

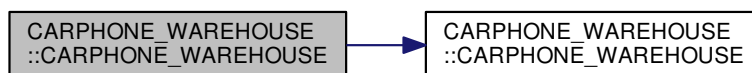
References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), and [CARPHONE_WAREHOUSE\(\)](#).

```

00055                                     :
00056     WAREHOUSE(_version, _unique_id){
00057         /*
00058          * copy over values
00059
00060          */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }

```

Here is the call graph for this function:



5.6.2.4 CARPHONE_WAREHOUSE::CARPHONE_WAREHOUSE (const CARPHONE_WAREHOUSE & orig)

Copy constructor

Definition at line 11 of file [CARPHONE_WAREHOUSE.cpp](#).

```

00011                                     {
00012 }

```

5.6.2.5 CARPHONE_WAREHOUSE::~CARPHONE_WAREHOUSE () [virtual]

de-constructor

Definition at line 14 of file [CARPHONE_WAREHOUSE.cpp](#).

Referenced by [operator=\(\)](#).

```

00014                                     {
00015 }

```

5.6.3 Member Function Documentation

5.6.3.1 void CARPHONE_WAREHOUSE::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 34 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```

00034
00035
00036     std::shared_ptr<CARPHONE_WAREHOUSE> objTO = std::dynamic_pointer_cast<
00037     CARPHONE\_WAREHOUSE>(to);
00037     std::shared_ptr<CARPHONE_WAREHOUSE> objFROM = std::dynamic_pointer_cast<
00038     CARPHONE\_WAREHOUSE>(from);
00038     objTO->_shop_address = objFROM->GetShop_address();
00039     objTO->_shop_name = objFROM->GetShop_name();
00040     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00041     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00042     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00043     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00044     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00045     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00046     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00047     objTO->Set_Version(objFROM->Get_Version());
00048
00049 }
```

5.6.3.2 std::shared_ptr< OSTM > CARPHONE_WAREHOUSE::getBaseCopy (std::shared_ptr< OSTM > object) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new BANK* type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a BANK* return type

Reimplemented from [OSTM](#).

Definition at line 21 of file [CARPHONE_WAREHOUSE.cpp](#).

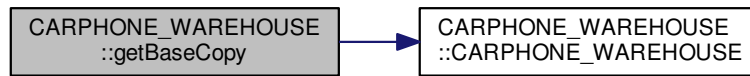
References [CARPHONE_WAREHOUSE\(\)](#).

Referenced by [operator=\(\)](#).

```

00022 {
00023
00024     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00025     std::shared_ptr<WAREHOUSE> obj(new CARPHONE\_WAREHOUSE(objTO, object->Get_Version(),
00026     object->Get_Unique_ID()));
00026     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00027     return ostm_obj;
00028 }
```

Here is the call graph for this function:



5.6.3.3 int CARPHONE_WAREHOUSE::GetNumber_of_alcatel () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 71 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```

00071                                     {
00072     return _number_of_alcatel;
00073 }
  
```

5.6.3.4 int CARPHONE_WAREHOUSE::GetNumber_of_huawei () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 87 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```

00087                                     {
00088     return _number_of_huawei;
00089 }
  
```

5.6.3.5 int CARPHONE_WAREHOUSE::GetNumber_of_iphones () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 111 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```

00111                                     {
00112     return _number_of_iphones;
00113 }
  
```

5.6.3.6 int CARPHONE_WAREHOUSE::GetNumber_of_nokia () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 79 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00079                                     {
00080     return _number_of_nokia;
00081 }
```

5.6.3.7 int CARPHONE_WAREHOUSE::GetNumber_of_samsung () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 103 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00103                                     {
00104     return _number_of_samsung;
00105 }
```

5.6.3.8 int CARPHONE_WAREHOUSE::GetNumber_of_sony () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 95 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00095                                     {
00096     return _number_of_sony;
00097 }
```

5.6.3.9 std::string CARPHONE_WAREHOUSE::GetShop_address () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 127 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00127                                     {
00128     return _shop_address;
00129 }
```

5.6.3.10 `std::string CARPHONE_WAREHOUSE::GetShop_name () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 119 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00119                                     {
00120     return _shop_name;
00121 }
```

5.6.3.11 `CARPHONE_WAREHOUSE CARPHONE_WAREHOUSE::operator= (const CARPHONE_WAREHOUSE & orig) [inline]`

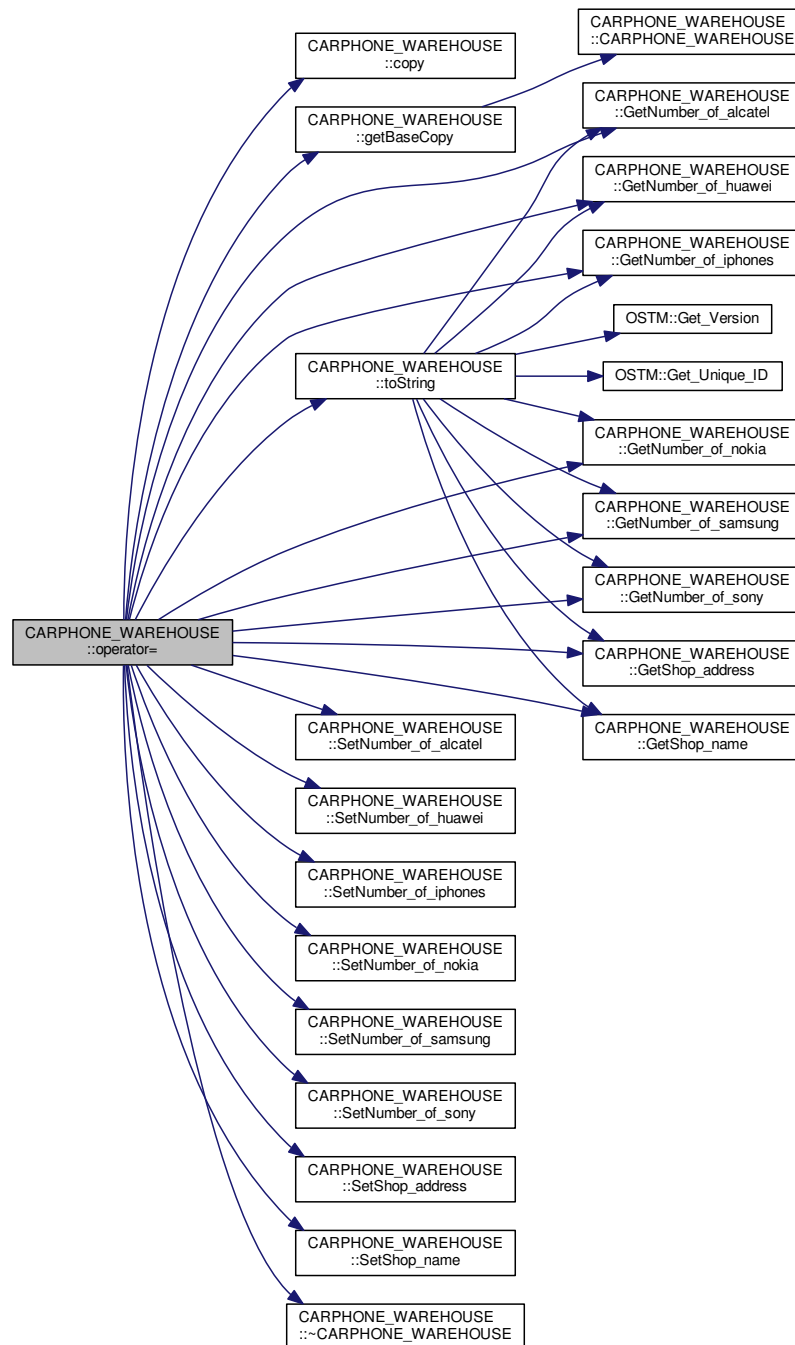
Operator

Definition at line 75 of file [CARPHONE_WAREHOUSE.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), [copy\(\)](#), [getBaseCopy\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), [GetShop_name\(\)](#), [SetNumber_of_alcatel\(\)](#), [SetNumber_of_huawei\(\)](#), [SetNumber_of_iphones\(\)](#), [SetNumber_of_nokia\(\)](#), [SetNumber_of_samsung\(\)](#), [SetNumber_of_sony\(\)](#), [SetShop_address\(\)](#), [SetShop_name\(\)](#), [toString\(\)](#), and [~CARPHONE_WAREHOUSE\(\)](#).

```
00075 {};
```

Here is the call graph for this function:



5.6.3.12 void CARPHONE_WAREHOUSE::SetNumber_of_alcatel (int *number_of_alcatel*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 67 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#).

```

00067                                     {
00068     this->_number_of_alcatel = _number_of_alcatel;
00069 }

```

5.6.3.13 void CARPHONE_WAREHOUSE::SetNumber_of_huawei (int _number_of_huawei) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 83 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#).

```

00083                                     {
00084     this->_number_of_huawei = _number_of_huawei;
00085 }

```

5.6.3.14 void CARPHONE_WAREHOUSE::SetNumber_of_iphones (int _number_of_iphones) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 107 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#).

```

00107                                     {
00108     this->_number_of_iphones = _number_of_iphones;
00109 }

```

5.6.3.15 void CARPHONE_WAREHOUSE::SetNumber_of_nokia (int _number_of_nokia) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 75 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#).

```

00075                                     {
00076     this->_number_of_nokia = _number_of_nokia;
00077 }

```

5.6.3.16 void CARPHONE_WAREHOUSE::SetNumber_of_samsung (int _number_of_samsung) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 99 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#).

```

00099                                     {
00100     this->_number_of_samsung = _number_of_samsung;
00101 }

```


5.6.3.17 void CARPHONE_WAREHOUSE::SetNumber_of_sony (int *_number_of_sony*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 91 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#).

```
00091                                     {  
00092     this->_number_of_sony = _number_of_sony;  
00093 }
```

5.6.3.18 void CARPHONE_WAREHOUSE::SetShop_address (std::string *_shop_address*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 123 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```
00123                                     {  
00124     this->_shop_address = _shop_address;  
00125 }
```

5.6.3.19 void CARPHONE_WAREHOUSE::SetShop_name (std::string *_shop_name*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 115 of file [CARPHONE_WAREHOUSE.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#).

```
00115                                     {  
00116     this->_shop_name = _shop_name;  
00117 }
```

5.6.3.20 void CARPHONE_WAREHOUSE::toString() [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

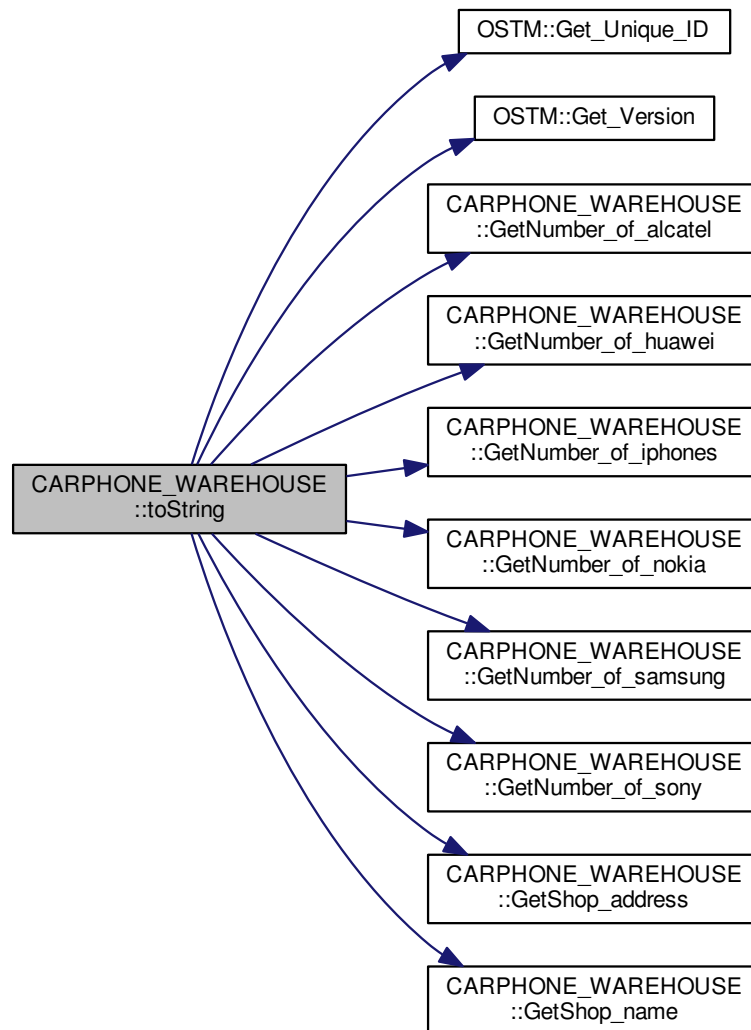
Definition at line 60 of file [CARPHONE_WAREHOUSE.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), and [GetShop_name\(\)](#).

Referenced by [operator=\(\)](#).

```
00061 {
00062     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
    Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
    << this->GetShop_address() << "\nNo. Iphones : " << this->
    GetNumber_of_iphones() << "\nNo. Samsung : " << this->
    GetNumber_of_samsung() << "\nNo. Sony : " << this->
    GetNumber_of_sony() << "\nNo. Huawei : " << this->
    GetNumber_of_huawei() << "\nNo. Nokia : " << this->
    GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
    GetNumber_of_alcatel() << "\nVersion number : " << this->
    Get_Version() << std::endl;
00063 }
```

Here is the call graph for this function:



5.6.4 Member Data Documentation

5.6.4.1 `int CARPHONE_WAREHOUSE::_number_of_alcatel` [private]

Definition at line 118 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [GetNumber_of_alcatel\(\)](#), [operator=\(\)](#), and [SetNumber_of_alcatel\(\)](#).

5.6.4.2 `int CARPHONE_WAREHOUSE::_number_of_huawei` [private]

Definition at line 116 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [GetNumber_of_huawei\(\)](#), [operator=\(\)](#), and [SetNumber_of_huawei\(\)](#).

5.6.4.3 `int CARPHONE_WAREHOUSE::_number_of_iphones` [private]

Definition at line 113 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [GetNumber_of_iphones\(\)](#), [operator=\(\)](#), and [SetNumber_of_iphones\(\)](#).

5.6.4.4 `int CARPHONE_WAREHOUSE::_number_of_nokia` [private]

Definition at line 117 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [GetNumber_of_nokia\(\)](#), [operator=\(\)](#), and [SetNumber_of_nokia\(\)](#).

5.6.4.5 `int CARPHONE_WAREHOUSE::_number_of_samsung` [private]

Definition at line 114 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [GetNumber_of_samsung\(\)](#), [operator=\(\)](#), and [SetNumber_of_samsung\(\)](#).

5.6.4.6 `int CARPHONE_WAREHOUSE::_number_of_sony` [private]

Definition at line 115 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [GetNumber_of_sony\(\)](#), [operator=\(\)](#), and [SetNumber_of_sony\(\)](#).

5.6.4.7 `std::string CARPHONE_WAREHOUSE::_shop_address` [private]

Definition at line 111 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [copy\(\)](#), [GetShop_address\(\)](#), [operator=\(\)](#), and [SetShop_address\(\)](#).

5.6.4.8 `std::string CARPHONE_WAREHOUSE::_shop_name` [private]

Definition at line 112 of file [CARPHONE_WAREHOUSE.h](#).

Referenced by [CARPHONE_WAREHOUSE\(\)](#), [GetShop_name\(\)](#), [operator=\(\)](#), and [SetShop_name\(\)](#).

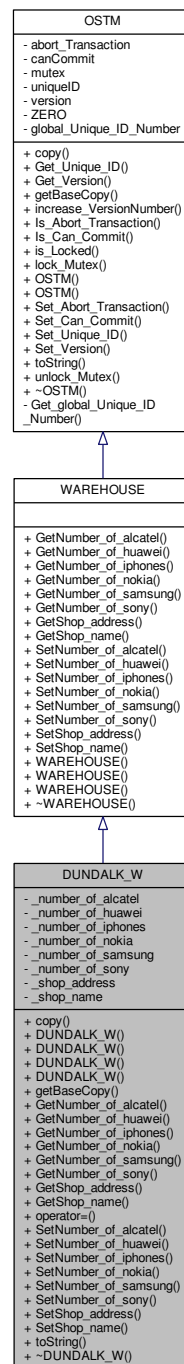
The documentation for this class was generated from the following files:

- [CARPHONE_WAREHOUSE.h](#)
- [CARPHONE_WAREHOUSE.cpp](#)

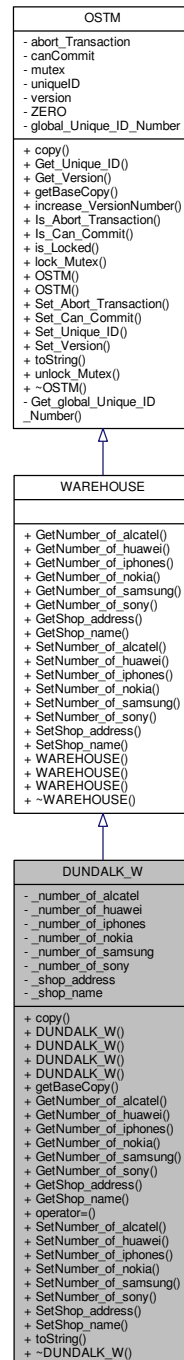
5.7 DUNDALK_W Class Reference

```
#include <DUNDALK_W.h>
```

Inheritance diagram for DUNDALK_W:



Collaboration diagram for DUNDALK_W:



Public Member Functions

- virtual void `copy` (std::shared_ptr< `OSTM` > to, std::shared_ptr< `OSTM` > from)
copy function, make deep copy of the object/pointer
- `DUNDALK_W` ()
- `DUNDALK_W` (std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel)

- [DUNDALK_W](#) (std::shared_ptr< [WAREHOUSE](#) > obj, int _version, int _unique_id)
- [DUNDALK_W](#) (const [DUNDALK_W](#) &orig)
- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new BANK type object*
- virtual int [GetNumber_of_alcatel](#) ()
- virtual int [GetNumber_of_huawei](#) ()
- virtual int [GetNumber_of_iphones](#) ()
- virtual int [GetNumber_of_nokia](#) ()
- virtual int [GetNumber_of_samsung](#) ()
- virtual int [GetNumber_of_sony](#) ()
- virtual std::string [GetShop_address](#) ()
- virtual std::string [GetShop_name](#) ()
- [DUNDALK_W](#) operator= (const [DUNDALK_W](#) &orig)
- virtual void [SetNumber_of_alcatel](#) (int _number_of_alcatel)
- virtual void [SetNumber_of_huawei](#) (int _number_of_huawei)
- virtual void [SetNumber_of_iphones](#) (int _number_of_iphones)
- virtual void [SetNumber_of_nokia](#) (int _number_of_nokia)
- virtual void [SetNumber_of_samsung](#) (int _number_of_samsung)
- virtual void [SetNumber_of_sony](#) (int _number_of_sony)
- virtual void [SetShop_address](#) (std::string _shop_address)
- virtual void [SetShop_name](#) (std::string _shop_name)
- virtual void [toString](#) ()
_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type
- virtual [~DUNDALK_W](#) ()

Private Attributes

- int [_number_of_alcatel](#)
- int [_number_of_huawei](#)
- int [_number_of_iphones](#)
- int [_number_of_nokia](#)
- int [_number_of_samsung](#)
- int [_number_of_sony](#)
- std::string [_shop_address](#)
- std::string [_shop_name](#)

5.7.1 Detailed Description

Inherit from [WAREHOUSE](#)

Definition at line 19 of file [DUNDALK_W.h](#).

5.7.2 Constructor & Destructor Documentation

5.7.2.1 DUNDALK_W::DUNDALK_W() [inline]

Constructor

Definition at line 24 of file [DUNDALK_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

Referenced by [DUNDALK_W\(\)](#), and [getBaseCopy\(\)](#).

```

00024         : WAREHOUSE() {
00025
00026     this->_shop_address = "Dundalk Busy Street";
00027     this->_shop_name = "DUNDALK D_WAREHOUSE";
00028     this->_number_of_iphones = 200;
00029     this->_number_of_samsung = 200;
00030     this->_number_of_sony = 200;
00031     this->_number_of_huawei = 200;
00032     this->_number_of_nokia = 200;
00033     this->_number_of_alcatel = 200;
00034 };

```

5.7.2.2 DUNDALK_W::DUNDALK_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel) [inline]

Custom constructor

Definition at line 38 of file [DUNDALK_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

```

00038         : WAREHOUSE() {
00039     /*
00040     * copy over values
00041     */
00042     this->_shop_address = address;
00043     this->_shop_name = shop_name;
00044     this->_number_of_iphones = iphone;
00045     this->_number_of_samsung = samsung;
00046     this->_number_of_sony = sony;
00047     this->_number_of_huawei = huawei;
00048     this->_number_of_nokia = nokia;
00049     this->_number_of_alcatel = alcatel;
00050
00051 };

```


5.7.2.3 DUNDALK_W::DUNDALK_W (std::shared_ptr< WAREHOUSE > obj, int _version, int _unique_id) [inline]

Custom constructor, used by the library for deep copying

Definition at line 55 of file [DUNDALK_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), and [DUNDALK_W\(\)](#).

```

00055                                     :
00056     WAREHOUSE(_version, _unique_id){
00057         /*
00058          * copy over values
00059          */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }

```

Here is the call graph for this function:



5.7.2.4 DUNDALK_W::DUNDALK_W (const DUNDALK_W & orig)

Copy constructor

Definition at line 15 of file [DUNDALK_W.cpp](#).

```

00015                                     {
00016 }

```

5.7.2.5 DUNDALK_W::~~DUNDALK_W () [virtual]

de-constructor

Definition at line 12 of file [DUNDALK_W.cpp](#).

Referenced by [operator=\(\)](#).

```

00012                                     {
00013 }

```

5.7.3 Member Function Documentation

5.7.3.1 void DUNDALK_W::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 35 of file [DUNDALK_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```

00035                                     {
00036
00037     std::shared_ptr<DUNDALK_W> objTO = std::dynamic_pointer_cast<DUNDALK_W>(to);
00038     std::shared_ptr<DUNDALK_W> objFROM = std::dynamic_pointer_cast<DUNDALK_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
```

5.7.3.2 std::shared_ptr< OSTM > DUNDALK_W::getBaseCopy (std::shared_ptr< OSTM > *object*) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new BANK* type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a BANK* return type

Reimplemented from [OSTM](#).

Definition at line 22 of file [DUNDALK_W.cpp](#).

References [DUNDALK_W\(\)](#).

Referenced by [operator=\(\)](#).

```

00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new DUNDALK\_W(objTO, object->Get_Version(), object->
00027         Get_Unique_ID()));
00028     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00029     return ostm_obj;
00029 }
```

Here is the call graph for this function:



5.7.3.3 `int DUNDALK_W::GetNumber_of_alcatel () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 73 of file [DUNDALK_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00073                                     {  
00074     return _number_of_alcatel;  
00075 }
```

5.7.3.4 `int DUNDALK_W::GetNumber_of_huawei () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 89 of file [DUNDALK_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00089                                     {  
00090     return _number_of_huawei;  
00091 }
```

5.7.3.5 `int DUNDALK_W::GetNumber_of_iphones () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 113 of file [DUNDALK_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00113                                     {  
00114     return _number_of_iphones;  
00115 }
```

5.7.3.6 int DUNDALK_W::GetNumber_of_nokia () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 81 of file [DUNDALK_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00081                                     {  
00082     return _number_of_nokia;  
00083 }
```

5.7.3.7 int DUNDALK_W::GetNumber_of_samsung () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 105 of file [DUNDALK_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00105                                     {  
00106     return _number_of_samsung;  
00107 }
```

5.7.3.8 int DUNDALK_W::GetNumber_of_sony () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 97 of file [DUNDALK_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00097                                     {  
00098     return _number_of_sony;  
00099 }
```

5.7.3.9 std::string DUNDALK_W::GetShop_address () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 129 of file [DUNDALK_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00129                                     {  
00130     return _shop_address;  
00131 }
```

5.7.3.10 std::string DUNDALK_W::GetShop_name() [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 121 of file [DUNDALK_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00121                                     {  
00122     return _shop_name;  
00123 }
```

5.7.3.11 DUNDALK_W DUNDALK_W::operator= (const DUNDALK_W & orig) [inline]

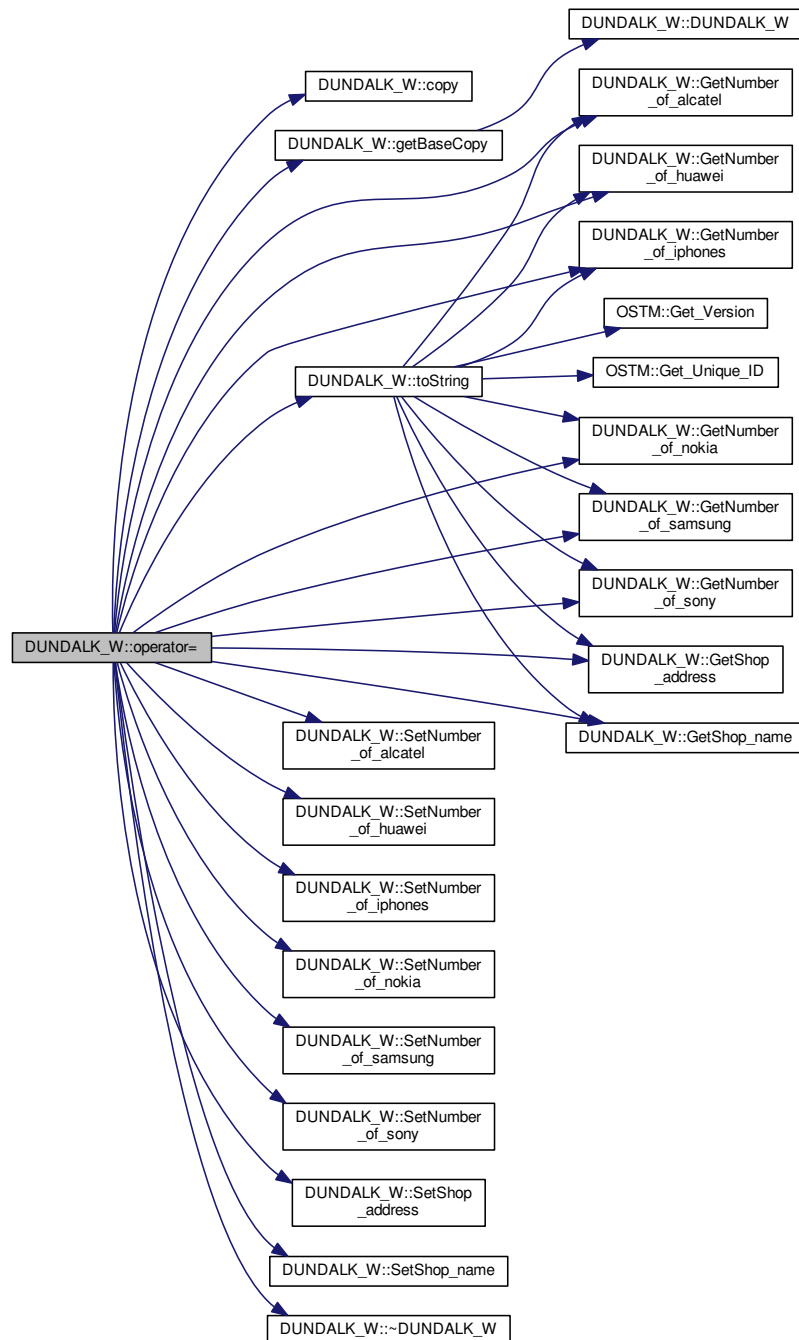
Operator

Definition at line 75 of file [DUNDALK_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), [copy\(\)](#), [getBaseCopy\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), [GetShop_name\(\)](#), [SetNumber_of_alcatel\(\)](#), [SetNumber_of_huawei\(\)](#), [SetNumber_of_iphones\(\)](#), [SetNumber_of_nokia\(\)](#), [SetNumber_of_samsung\(\)](#), [SetNumber_of_sony\(\)](#), [SetShop_address\(\)](#), [SetShop_name\(\)](#), [toString\(\)](#), and [~DUNDALK_W\(\)](#).

```
00075 {};
```

Here is the call graph for this function:



5.7.3.12 `void DUNDALK_W::SetNumber_of_alcatel (int number_of_alcatel)` [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 69 of file [DUNDALK_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#).

```
00069                                     {
00070     this->_number_of_alcatel = _number_of_alcatel;
00071 }
```

5.7.3.13 void DUNDALK_W::SetNumber_of_huawei (int *_number_of_huawei*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 85 of file [DUNDALK_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#).

```
00085                                     {
00086     this->_number_of_huawei = _number_of_huawei;
00087 }
```

5.7.3.14 void DUNDALK_W::SetNumber_of_iphones (int *_number_of_iphones*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 109 of file [DUNDALK_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#).

```
00109                                     {
00110     this->_number_of_iphones = _number_of_iphones;
00111 }
```

5.7.3.15 void DUNDALK_W::SetNumber_of_nokia (int *_number_of_nokia*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 77 of file [DUNDALK_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#).

```
00077                                     {
00078     this->_number_of_nokia = _number_of_nokia;
00079 }
```

5.7.3.16 void DUNDALK_W::SetNumber_of_samsung (int *_number_of_samsung*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 101 of file [DUNDALK_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#).

```
00101                                     {
00102     this->_number_of_samsung = _number_of_samsung;
00103 }
```

5.7.3.17 void DUNDALK_W::SetNumber_of_sony (int *_number_of_sony*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 93 of file [DUNDALK_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#).

```
00093                                     {  
00094     this->_number_of_sony = _number_of_sony;  
00095 }
```

5.7.3.18 void DUNDALK_W::SetShop_address (std::string *_shop_address*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 125 of file [DUNDALK_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```
00125                                     {  
00126     this->_shop_address = _shop_address;  
00127 }
```

5.7.3.19 void DUNDALK_W::SetShop_name (std::string *_shop_name*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 117 of file [DUNDALK_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#).

```
00117                                     {  
00118     this->_shop_name = _shop_name;  
00119 }
```


5.7.3.20 void DUNDALK_W::toString () [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

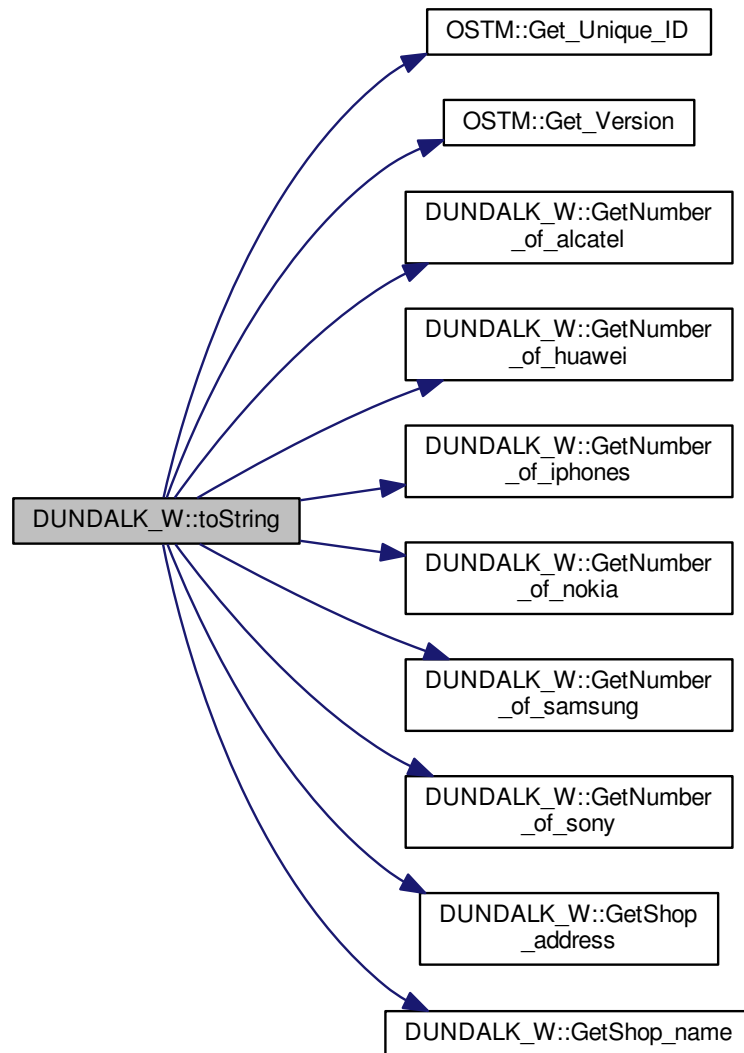
Definition at line 62 of file [DUNDALK_W.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), and [GetShop_name\(\)](#).

Referenced by [operator=\(\)](#).

```
00063 {
00064     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
    Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
    << this->GetShop_address() << "\nNo. Iphones : " << this->
    GetNumber_of_iphones() << "\nNo. Samsung : " << this->
    GetNumber_of_samsung() << "\nNo. Sony : " << this->
    GetNumber_of_sony() << "\nNo. Huawei : " << this->
    GetNumber_of_huawei() << "\nNo. Nokia : " << this->
    GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
    GetNumber_of_alcatel() << "\nVersion number : " << this->
    Get_Version() << std::endl;
00065 }
```

Here is the call graph for this function:



5.7.4 Member Data Documentation

5.7.4.1 `int DUNDALK_W::_number_of_alcatel` [private]

Definition at line 116 of file [DUNDALK_W.h](#).

Referenced by [DUNDALK_W\(\)](#), [GetNumber_of_alcatel\(\)](#), [operator=\(\)](#), and [SetNumber_of_alcatel\(\)](#).

5.7.4.2 `int DUNDALK_W::_number_of_huawei` [private]

Definition at line 114 of file [DUNDALK_W.h](#).

Referenced by [DUNDALK_W\(\)](#), [GetNumber_of_huawei\(\)](#), [operator=\(\)](#), and [SetNumber_of_huawei\(\)](#).

5.7.4.3 `int DUNDALK_W::_number_of_iphones` [private]

Definition at line 111 of file [DUNDALK_W.h](#).

Referenced by [DUNDALK_W\(\)](#), [GetNumber_of_iphones\(\)](#), [operator=\(\)](#), and [SetNumber_of_iphones\(\)](#).

5.7.4.4 `int DUNDALK_W::_number_of_nokia` [private]

Definition at line 115 of file [DUNDALK_W.h](#).

Referenced by [DUNDALK_W\(\)](#), [GetNumber_of_nokia\(\)](#), [operator=\(\)](#), and [SetNumber_of_nokia\(\)](#).

5.7.4.5 `int DUNDALK_W::_number_of_samsung` [private]

Definition at line 112 of file [DUNDALK_W.h](#).

Referenced by [DUNDALK_W\(\)](#), [GetNumber_of_samsung\(\)](#), [operator=\(\)](#), and [SetNumber_of_samsung\(\)](#).

5.7.4.6 `int DUNDALK_W::_number_of_sony` [private]

Definition at line 113 of file [DUNDALK_W.h](#).

Referenced by [DUNDALK_W\(\)](#), [GetNumber_of_sony\(\)](#), [operator=\(\)](#), and [SetNumber_of_sony\(\)](#).

5.7.4.7 `std::string DUNDALK_W::_shop_address` [private]

Definition at line 109 of file [DUNDALK_W.h](#).

Referenced by [copy\(\)](#), [DUNDALK_W\(\)](#), [GetShop_address\(\)](#), [operator=\(\)](#), and [SetShop_address\(\)](#).

5.7.4.8 `std::string DUNDALK_W::_shop_name` [private]

Definition at line 110 of file [DUNDALK_W.h](#).

Referenced by [DUNDALK_W\(\)](#), [GetShop_name\(\)](#), [operator=\(\)](#), and [SetShop_name\(\)](#).

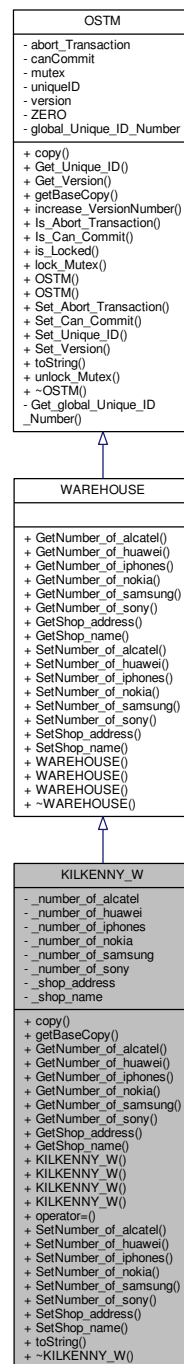
The documentation for this class was generated from the following files:

- [DUNDALK_W.h](#)
- [DUNDALK_W.cpp](#)

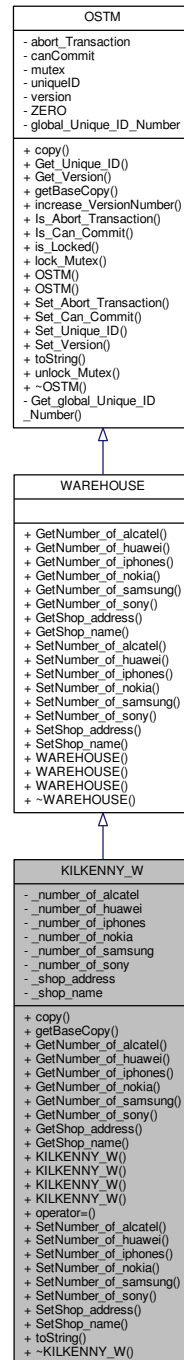
5.8 KILKENNY_W Class Reference

```
#include <KILKENNY_W.h>
```

Inheritance diagram for KILKENNY_W:



Collaboration diagram for KILKENNY_W:



Public Member Functions

- virtual void [copy](#) (std::shared_ptr< [OSTM](#) > to, std::shared_ptr< [OSTM](#) > from)
copy function, make deep copy of the object/pointer
- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new BANK type object*
- virtual int [GetNumber_of_alcatel](#) ()

- virtual int [GetNumber_of_huawei](#) ()
- virtual int [GetNumber_of_iphones](#) ()
- virtual int [GetNumber_of_nokia](#) ()
- virtual int [GetNumber_of_samsung](#) ()
- virtual int [GetNumber_of_sony](#) ()
- virtual std::string [GetShop_address](#) ()
- virtual std::string [GetShop_name](#) ()
- [KILKENNY_W](#) ()
- [KILKENNY_W](#) (std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel)
- [KILKENNY_W](#) (std::shared_ptr< [WAREHOUSE](#) > obj, int _version, int _unique_id)
- [KILKENNY_W](#) (const [KILKENNY_W](#) &orig)
- [KILKENNY_W](#) operator= (const [KILKENNY_W](#) &orig)
- virtual void [SetNumber_of_alcatel](#) (int _number_of_alcatel)
- virtual void [SetNumber_of_huawei](#) (int _number_of_huawei)
- virtual void [SetNumber_of_iphones](#) (int _number_of_iphones)
- virtual void [SetNumber_of_nokia](#) (int _number_of_nokia)
- virtual void [SetNumber_of_samsung](#) (int _number_of_samsung)
- virtual void [SetNumber_of_sony](#) (int _number_of_sony)
- virtual void [SetShop_address](#) (std::string _shop_address)
- virtual void [SetShop_name](#) (std::string _shop_name)
- virtual void [toString](#) ()
- *_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type*
- virtual [~KILKENNY_W](#) ()

Private Attributes

- int [_number_of_alcatel](#)
- int [_number_of_huawei](#)
- int [_number_of_iphones](#)
- int [_number_of_nokia](#)
- int [_number_of_samsung](#)
- int [_number_of_sony](#)
- std::string [_shop_address](#)
- std::string [_shop_name](#)

5.8.1 Detailed Description

Inherit from [WAREHOUSE](#)

Definition at line 19 of file [KILKENNY_W.h](#).

5.8.2 Constructor & Destructor Documentation

5.8.2.1 KILKENNY_W::KILKENNY_W() [inline]

Constructor

Definition at line 24 of file [KILKENNY_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

Referenced by [getBaseCopy\(\)](#), and [KILKENNY_W\(\)](#).

```

00024         : WAREHOUSE() {
00025
00026     this->_shop_address = "Kilkenny High Street";
00027     this->_shop_name = "KILKENNY K_WAREHOUSE";
00028     this->_number_of_iphones = 200;
00029     this->_number_of_samsung = 200;
00030     this->_number_of_sony = 200;
00031     this->_number_of_huawei = 200;
00032     this->_number_of_nokia = 200;
00033     this->_number_of_alcatel = 200;
00034 };

```

5.8.2.2 KILKENNY_W::KILKENNY_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel) [inline]

Custom constructor

Definition at line 38 of file [KILKENNY_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

```

00038         : WAREHOUSE() {
00039     /*
00040     * copy over values
00041     */
00042     this->_shop_address = address;
00043     this->_shop_name = shop_name;
00044     this->_number_of_iphones = iphone;
00045     this->_number_of_samsung = samsung;
00046     this->_number_of_sony = sony;
00047     this->_number_of_huawei = huawei;
00048     this->_number_of_nokia = nokia;
00049     this->_number_of_alcatel = alcatel;
00050
00051 };

```

5.8.2.3 KILKENNY_W::KILKENNY_W (std::shared_ptr< WAREHOUSE > obj, int _version, int _unique_id) [inline]

Custom constructor, used by the library for deep copying

Definition at line 55 of file [KILKENNY_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), and [KILKENNY_W\(\)](#).

```

00055                                     :
00056     WAREHOUSE(_version, _unique_id){
00057         /*
00058          * copy over values
00059          */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }

```

Here is the call graph for this function:



5.8.2.4 KILKENNY_W::KILKENNY_W (const KILKENNY_W & orig)

Copy constructor

Definition at line 15 of file [KILKENNY_W.cpp](#).

```

00015                                     {
00016 }

```

5.8.2.5 KILKENNY_W::~~KILKENNY_W () [virtual]

de-constructor

Definition at line 12 of file [KILKENNY_W.cpp](#).

Referenced by [operator=\(\)](#).

```

00012                                     {
00013 }

```

5.8.3 Member Function Documentation

5.8.3.1 void KILKENNY_W::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 35 of file [KILKENNY_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```

00035                                     {
00036
00037     std::shared_ptr<KILKENNY_W> objTO = std::dynamic_pointer_cast<KILKENNY_W>(to);
00038     std::shared_ptr<KILKENNY_W> objFROM = std::dynamic_pointer_cast<KILKENNY_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
```

5.8.3.2 std::shared_ptr<OSTM> KILKENNY_W::getBaseCopy (std::shared_ptr<OSTM> *object*) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new BANK* type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a BANK* return type

Reimplemented from [OSTM](#).

Definition at line 22 of file [KILKENNY_W.cpp](#).

References [KILKENNY_W\(\)](#).

Referenced by [operator=\(\)](#).

```

00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new KILKENNY\_W(objTO, object->Get_Version(), object->
00027         Get_Unique_ID()));
00028     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00029     return ostm_obj;
00029 }
```

Here is the call graph for this function:



5.8.3.3 int KILKENNY_W::GetNumber_of_alcatel () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 73 of file [KILKENNY_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00073                                     {  
00074     return _number_of_alcatel;  
00075 }
```

5.8.3.4 int KILKENNY_W::GetNumber_of_huawei () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 89 of file [KILKENNY_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00089                                     {  
00090     return _number_of_huawei;  
00091 }
```

5.8.3.5 int KILKENNY_W::GetNumber_of_iphones () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 113 of file [KILKENNY_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00113                                     {  
00114     return _number_of_iphones;  
00115 }
```

5.8.3.6 int KILKENNY_W::GetNumber_of_nokia () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 81 of file [KILKENNY_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00081                                     {  
00082     return _number_of_nokia;  
00083 }
```

5.8.3.7 int KILKENNY_W::GetNumber_of_samsung () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 105 of file [KILKENNY_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00105                                     {  
00106     return _number_of_samsung;  
00107 }
```

5.8.3.8 int KILKENNY_W::GetNumber_of_sony () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 97 of file [KILKENNY_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00097                                     {  
00098     return _number_of_sony;  
00099 }
```

5.8.3.9 std::string KILKENNY_W::GetShop_address () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 129 of file [KILKENNY_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00129                                     {  
00130     return _shop_address;  
00131 }
```

5.8.3.10 `std::string KILKENNY_W::GetShop_name () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 121 of file [KILKENNY_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00121                                     {
00122     return _shop_name;
00123 }
```

5.8.3.11 `KILKENNY_W KILKENNY_W::operator= (const KILKENNY_W & orig) [inline]`

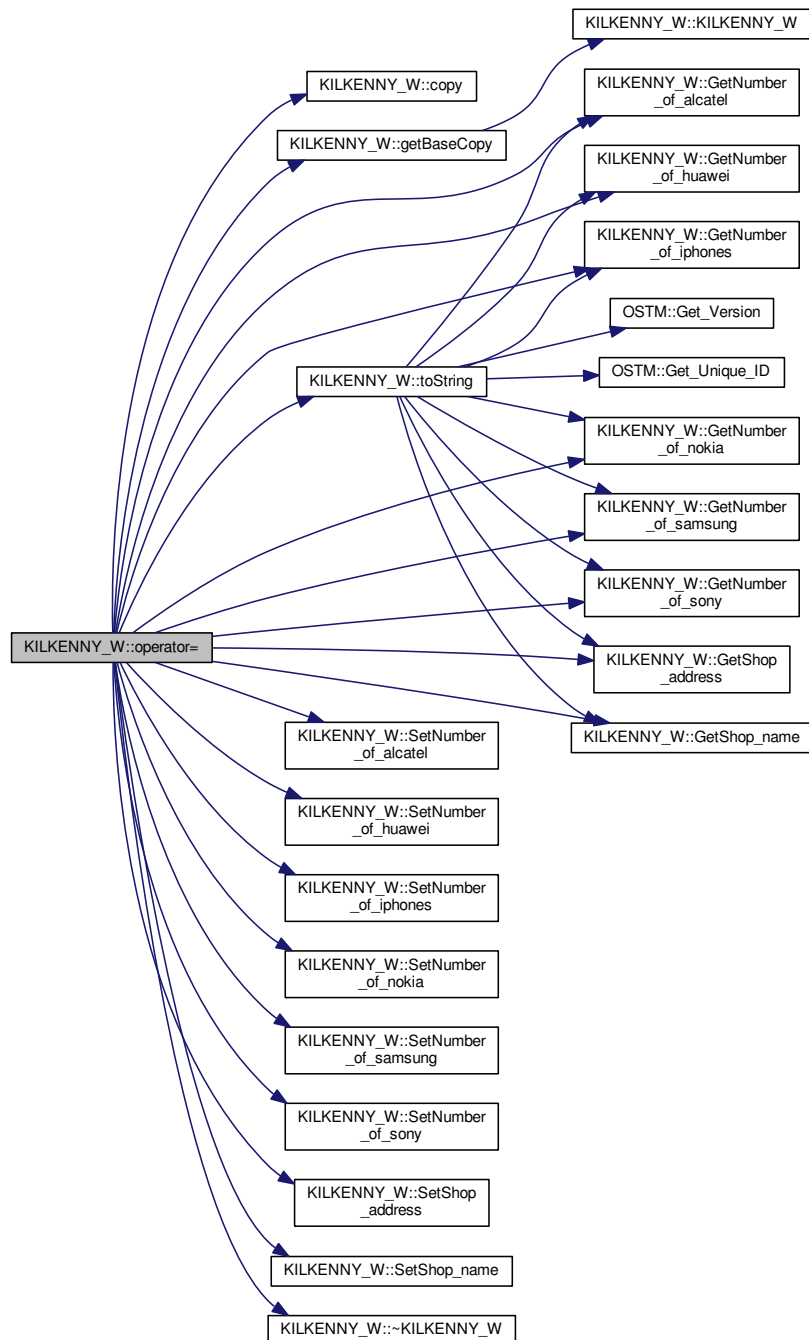
Operator

Definition at line 75 of file [KILKENNY_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), [copy\(\)](#), [getBaseCopy\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), [GetShop_name\(\)](#), [SetNumber_of_alcatel\(\)](#), [SetNumber_of_huawei\(\)](#), [SetNumber_of_iphones\(\)](#), [SetNumber_of_nokia\(\)](#), [SetNumber_of_samsung\(\)](#), [SetNumber_of_sony\(\)](#), [SetShop_address\(\)](#), [SetShop_name\(\)](#), [toString\(\)](#), and [~KILKENNY_W\(\)](#).

```
00075 {};
```

Here is the call graph for this function:



5.8.3.12 `void KILKENNY_W::SetNumber_of_alcatel (int _number_of_alcatel)` `[virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 69 of file [KILKENNY_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#).

```

00069                                     {
00070     this->_number_of_alcatel = _number_of_alcatel;
00071 }

```

5.8.3.13 void KILKENNY_W::SetNumber_of_huawei (int *_number_of_huawei*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 85 of file [KILKENNY_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#).

```

00085                                     {
00086     this->_number_of_huawei = _number_of_huawei;
00087 }

```

5.8.3.14 void KILKENNY_W::SetNumber_of_iphones (int *_number_of_iphones*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 109 of file [KILKENNY_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#).

```

00109                                     {
00110     this->_number_of_iphones = _number_of_iphones;
00111 }

```

5.8.3.15 void KILKENNY_W::SetNumber_of_nokia (int *_number_of_nokia*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 77 of file [KILKENNY_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#).

```

00077                                     {
00078     this->_number_of_nokia = _number_of_nokia;
00079 }

```

5.8.3.16 void KILKENNY_W::SetNumber_of_samsung (int *_number_of_samsung*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 101 of file [KILKENNY_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#).

```

00101                                     {
00102     this->_number_of_samsung = _number_of_samsung;
00103 }

```

5.8.3.17 void KILKENNY_W::SetNumber_of_sony (int *_number_of_sony*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 93 of file [KILKENNY_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#).

```
00093                                     {
00094     this->_number_of_sony = _number_of_sony;
00095 }
```

5.8.3.18 void KILKENNY_W::SetShop_address (std::string *_shop_address*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 125 of file [KILKENNY_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```
00125                                     {
00126     this->_shop_address = _shop_address;
00127 }
```

5.8.3.19 void KILKENNY_W::SetShop_name (std::string *_shop_name*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 117 of file [KILKENNY_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#).

```
00117                                     {
00118     this->_shop_name = _shop_name;
00119 }
```

5.8.3.20 void KILKENNY_W::toString () [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

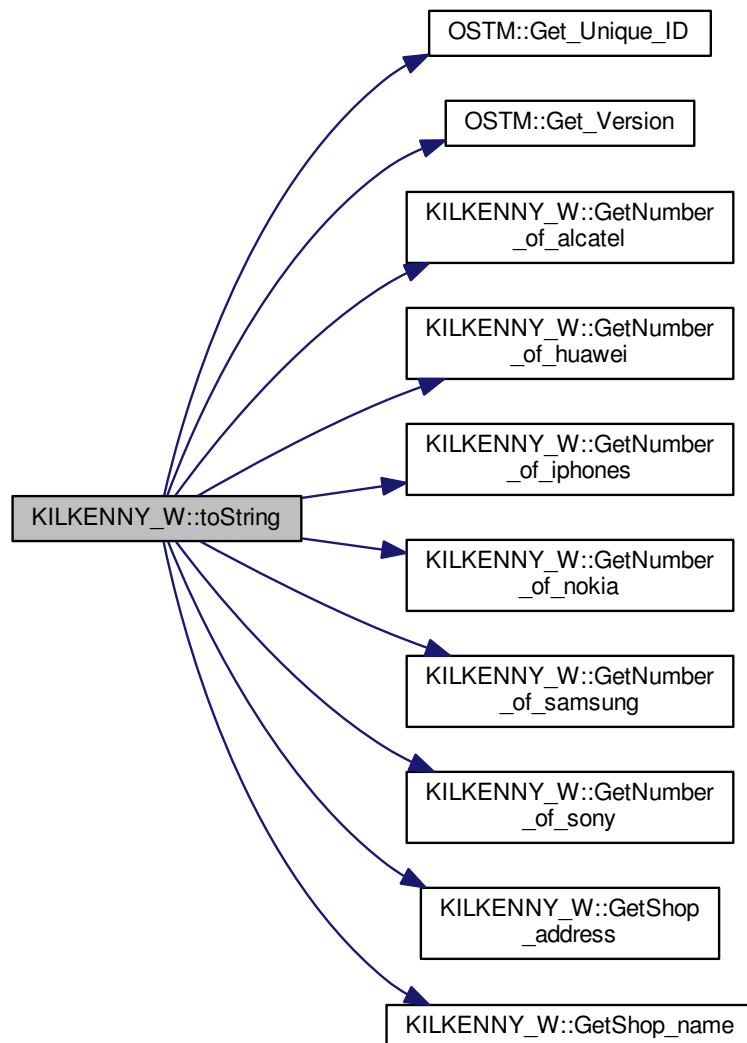
Definition at line 62 of file [KILKENNY_W.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), and [GetShop_name\(\)](#).

Referenced by [operator=\(\)](#).

```
00063 {
00064     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
        Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
        << this->GetShop_address() << "\nNo. Iphones : " << this->
        GetNumber_of_iphones() << "\nNo. Samsung : " << this->
        GetNumber_of_samsung() << "\nNo. Sony : " << this->
        GetNumber_of_sony() << "\nNo. Huawei : " << this->
        GetNumber_of_huawei() << "\nNo. Nokia : " << this->
        GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
        GetNumber_of_alcatel() << "\nVersion number : " << this->
        Get_Version() << std::endl;
00065 }
```


Here is the call graph for this function:



5.8.4 Member Data Documentation

5.8.4.1 `int KILKENNY_W::_number_of_alcatel` `[private]`

Definition at line 116 of file [KILKENNY_W.h](#).

Referenced by [GetNumber_of_alcatel\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetNumber_of_alcatel\(\)](#).

5.8.4.2 `int KILKENNY_W::_number_of_huawei` `[private]`

Definition at line 114 of file [KILKENNY_W.h](#).

Referenced by [GetNumber_of_huawei\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetNumber_of_huawei\(\)](#).

5.8.4.3 `int KILKENNY_W::_number_of_iphones [private]`

Definition at line 111 of file [KILKENNY_W.h](#).

Referenced by [GetNumber_of_iphones\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetNumber_of_iphones\(\)](#).

5.8.4.4 `int KILKENNY_W::_number_of_nokia [private]`

Definition at line 115 of file [KILKENNY_W.h](#).

Referenced by [GetNumber_of_nokia\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetNumber_of_nokia\(\)](#).

5.8.4.5 `int KILKENNY_W::_number_of_samsung [private]`

Definition at line 112 of file [KILKENNY_W.h](#).

Referenced by [GetNumber_of_samsung\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetNumber_of_samsung\(\)](#).

5.8.4.6 `int KILKENNY_W::_number_of_sony [private]`

Definition at line 113 of file [KILKENNY_W.h](#).

Referenced by [GetNumber_of_sony\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetNumber_of_sony\(\)](#).

5.8.4.7 `std::string KILKENNY_W::_shop_address [private]`

Definition at line 109 of file [KILKENNY_W.h](#).

Referenced by [copy\(\)](#), [GetShop_address\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetShop_address\(\)](#).

5.8.4.8 `std::string KILKENNY_W::_shop_name [private]`

Definition at line 110 of file [KILKENNY_W.h](#).

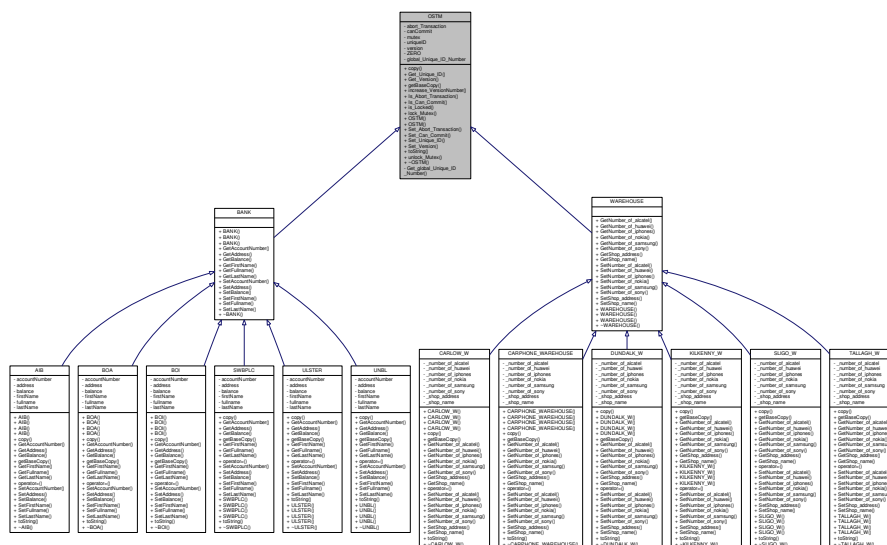
Referenced by [GetShop_name\(\)](#), [KILKENNY_W\(\)](#), [operator=\(\)](#), and [SetShop_name\(\)](#).

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

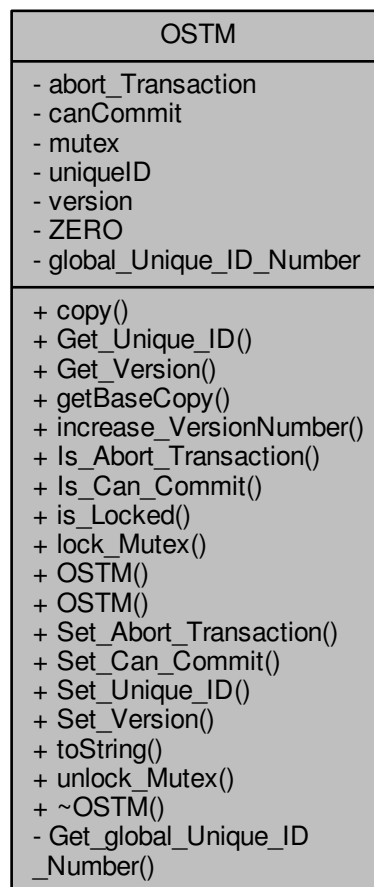
- [KILKENNY_W.h](#)
- [KILKENNY_W.cpp](#)

```
#include <OSTM.h>
```

Inheritance diagram for OSTM:



Collaboration diagram for OSTM:



Public Member Functions

- virtual void `copy` (std::shared_ptr< `OSTM` > from, std::shared_ptr< `OSTM` > to)
`OSTM` required virtual method for deep copy.
- int `Get_Unique_ID` () const
getter for unique id
- int `Get_Version` () const
getter for version number
- virtual std::shared_ptr< `OSTM` > `getBaseCopy` (std::shared_ptr< `OSTM` > object)
`OSTM` required virtual method for returning a pointer that is copy of the original pointer.
- void `increase_VersionNumber` ()
commit time increase version number to child object
- bool `Is_Abort_Transaction` () const
NOT USED YET.
- bool `Is_Can_Commit` () const
NOT USED YET.

- bool [is_Locked](#) ()
object unique lock, try locks mutex return boolean value depends on the lock state
- void [lock_Mutex](#) ()
object unique lock, locks mutex
- [OSTM](#) ()
OSTM Constructor.
- [OSTM](#) (int [_version_number_](#), int [_unique_id_](#))
OSTM Custom Constructor.
- void [Set_Abort_Transaction](#) (bool [abortTransaction](#))
NOT USED YET.
- void [Set_Can_Commit](#) (bool [canCommit](#))
NOT USED YET.
- void [Set_Unique_ID](#) (int [uniqueID](#))
setter for unique id
- void [Set_Version](#) (int [version](#))
setter for version number
- virtual void [toString](#) ()
OSTM required virtual method for display object.
- void [unlock_Mutex](#) ()
object unique lock, unlocks mutex
- virtual [~OSTM](#) ()
De-constructor.

Private Member Functions

- int [Get_global_Unique_ID_Number](#) ()

Private Attributes

- bool [abort_Transaction](#)
- bool [canCommit](#)
- std::mutex [mutex](#)
Object built in lock.
- int [uniqueID](#)
- int [version](#)
- const int [ZERO](#) = 0
Meaningful display for value 0.

Static Private Attributes

- static int [global_Unique_ID_Number](#) = 0
Unique object number increase at object creation.

5.9.1 Detailed Description

Definition at line 17 of file [OSTM.h](#).

5.9.2 Constructor & Destructor Documentation

5.9.2.1 [OSTM::OSTM](#) ()

[OSTM](#) Constructor.

Default constructor.

Parameters

<i>version</i>	indicates the version number of the inherited child pointer
<i>uniqueID</i>	is a unique identifier assigned to every object registered in OSTM library
<i>canCommit</i>	NOT USED YET
<i>abort_Transaction</i>	NOT USED YET

Definition at line 20 of file [OSTM.cpp](#).

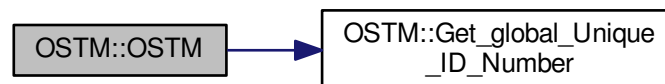
References [abort_Transaction](#), [canCommit](#), [Get_global_Unique_ID_Number\(\)](#), [uniqueID](#), [version](#), and [ZERO](#).

```

00021 {
00022     this->version = ZERO;
00023     this->uniqueID = Get_global_Unique_ID_Number(); //
++global_Unique_ID_Number;
00024     this->canCommit = true;
00025     this->abort_Transaction = false;
00026 }

```

Here is the call graph for this function:



5.9.2.2 OSTM::OSTM (int _version_number_, int _unique_id_)

[OSTM](#) Custom Constructor.

Custom Constructor Used for copy object.

Parameters

<i>version</i>	indicates the version number of the inherited child pointer
<i>uniqueID</i>	is a unique identifier assigned to every object registered in OSTM library
<i>canCommit</i>	NOT USED YET
<i>abort_Transaction</i>	NOT USED YET

Definition at line 36 of file [OSTM.cpp](#).

References [abort_Transaction](#), [canCommit](#), [uniqueID](#), and [version](#).

```

00037 {
00038     // std::cout << "OSTM COPY CONSTRUCTOR" << global_Unique_ID_Number << std::endl;
00039     this->uniqueID = _unique_id_;
00040     this->version = _version_number_;
00041     this->canCommit = true;
00042     this->abort_Transaction = false;
00043 }

```

5.9.2.3 OSTM::~~OSTM () [virtual]

De-constructor.

De-constructor

Definition at line 48 of file [OSTM.cpp](#).

```
00048         {
00049         //std::cout << "[OSTM DELETE]" << std::endl;
00050     }
```

5.9.3 Member Function Documentation**5.9.3.1 virtual void OSTM::copy (std::shared_ptr< OSTM > from, std::shared_ptr< OSTM > to) [inline], [virtual]**

[OSTM](#) required virtual method for deep copy.

Reimplemented in [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [CARPHONE_WAREHOUSE](#), [DUNDALK_W](#), [KILKE](#), [NNY_W](#), [AIB](#), [BOI](#), [BOA](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 34 of file [OSTM.h](#).

```
00034 {};
```

5.9.3.2 int OSTM::Get_global_Unique_ID_Number () [private]

Returning global_Unique_ID_Number to the constructor

If global_Unique_ID_Number equals to 10000000 then reset back to ZERO, to make sure the value of global_Unique_ID_Number never exceed the MAX_INT value

Definition at line 56 of file [OSTM.cpp](#).

References [global_Unique_ID_Number](#).

Referenced by [OSTM\(\)](#).

```
00056         {
00057         if(global_Unique_ID_Number > 10000000)
00058             global_Unique_ID_Number = 0;
00059         return ++global_Unique_ID_Number;
00060     }
```

5.9.3.3 int OSTM::Get_Unique_ID () const

getter for unique id

Parameters

<i>uniqueID</i>	int
-----------------	-----

Definition at line 73 of file [OSTM.cpp](#).

References [uniqueID](#).

Referenced by [toString\(\)](#), [ULSTER::toString\(\)](#), [UNBL::toString\(\)](#), [SWBPLC::toString\(\)](#), [BOA::toString\(\)](#), [BOI::toString\(\)](#), [AIB::toString\(\)](#), [KILKENNY_W::toString\(\)](#), [CARLOW_W::toString\(\)](#), [DUNDALK_W::toString\(\)](#), [SLIGO_W::toString\(\)](#), [TALLAGH_W::toString\(\)](#), and [CARPHONE_WAREHOUSE::toString\(\)](#).

```
00074 {
00075     return uniqueID;
00076 }
```

5.9.3.4 int OSTM::Get_Version () const

getter for version number

Parameters

<i>version</i>	int
----------------	-----

Definition at line 89 of file [OSTM.cpp](#).

References [version](#).

Referenced by [toString\(\)](#), [ULSTER::toString\(\)](#), [UNBL::toString\(\)](#), [SWBPLC::toString\(\)](#), [BOA::toString\(\)](#), [BOI::toString\(\)](#), [AIB::toString\(\)](#), [KILKENNY_W::toString\(\)](#), [CARLOW_W::toString\(\)](#), [DUNDALK_W::toString\(\)](#), [SLIGO_W::toString\(\)](#), [TALLAGH_W::toString\(\)](#), and [CARPHONE_WAREHOUSE::toString\(\)](#).

```
00090 {
00091     return version;
00092 }
```

5.9.3.5 virtual std::shared_ptr<OSTM> OSTM::getBaseCopy (std::shared_ptr<OSTM> *object*) [inline], [virtual]

[OSTM](#) required virtual method for returning a pointer that is copy of the original pointer.

Reimplemented in [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [CARPHONE_WAREHOUSE](#), [DUNDALK_W](#), [KILKENNY_W](#), [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 38 of file [OSTM.h](#).

```
00038 {};//std::cout << "[OSTM GETBASECOPY]" << std::endl;;
```

5.9.3.6 void OSTM::increase_VersionNumber ()

commit time increase version number to child object

Parameters

<i>version</i>	int
----------------	-----

Definition at line 97 of file [OSTM.cpp](#).

References [version](#).

Referenced by [toString\(\)](#).

```
00098 {
00099     this->version += 1;
00100 }
```

5.9.3.7 bool OSTM::Is_Abort_Transaction () const

NOT USED YET.

Parameters

<i>abort_Transaction</i>	boolean
--------------------------	---------

Definition at line 126 of file [OSTM.cpp](#).

References [abort_Transaction](#).

Referenced by [toString\(\)](#).

```
00126                                     {
00127     return abort_Transaction;
00128 }
```

5.9.3.8 bool OSTM::Is_Can_Commit () const

NOT USED YET.

Parameters

<i>canCommit</i>	boolean
------------------	---------

Definition at line 112 of file [OSTM.cpp](#).

References [canCommit](#).

Referenced by [toString\(\)](#).

```
00112                                     {
00113     return canCommit;
00114 }
```

5.9.3.9 bool OSTM::is_Locked ()

object unique lock, try locks mutex return boolean value depends on the lock state

Parameters

<i>mutex</i>	std::mutex
--------------	------------

Definition at line 147 of file [OSTM.cpp](#).

References [mutex](#).

Referenced by [toString\(\)](#).

```
00147         {
00148     return this->mutex.try_lock();
00149 }
```

5.9.3.10 void OSTM::lock_Mutex ()

object unique lock, locks mutex

Parameters

<i>mutex</i>	std::mutex
--------------	------------

Definition at line 133 of file [OSTM.cpp](#).

References [mutex](#).

Referenced by [toString\(\)](#).

```
00133     {
00134     this->mutex.lock();
00135 }
```

5.9.3.11 void OSTM::Set_Abort_Transaction (bool *abortTransaction*)

NOT USED YET.

Parameters

<i>abort_Transaction</i>	boolean
--------------------------	---------

Definition at line 119 of file [OSTM.cpp](#).

References [abort_Transaction](#).

Referenced by [toString\(\)](#).

```
00119     {
00120     this->abort_Transaction = abortTransaction;
00121 }
```

5.9.3.12 void OSTM::Set_Can_Commit (bool *canCommit*)

NOT USED YET.

Parameters

<i>canCommit</i>	boolean
------------------	---------

Definition at line 105 of file [OSTM.cpp](#).

References [canCommit](#).

Referenced by [toString\(\)](#).

```
00105                                     {
00106     this->canCommit = canCommit;
00107 }
```

5.9.3.13 void OSTM::Set_Unique_ID (int *uniqueID*)

setter for unique id

Parameters

<i>uniqueID</i>	int
-----------------	-----

Definition at line 66 of file [OSTM.cpp](#).

References [uniqueID](#).

Referenced by [ULSTER::copy\(\)](#), [UNBL::copy\(\)](#), [SWBPLC::copy\(\)](#), [BOA::copy\(\)](#), [AIB::copy\(\)](#), [BOI::copy\(\)](#), and [toString\(\)](#).

```
00066                                     {
00067     this->uniqueID = uniqueID;
00068 }
```

5.9.3.14 void OSTM::Set_Version (int *version*)

setter for version number

Parameters

<i>version</i>	int
----------------	-----

Definition at line 81 of file [OSTM.cpp](#).

References [version](#).

Referenced by [toString\(\)](#).

```
00082 {
00083     this->version = version;
00084 }
```

5.9.3.15 virtual void OSTM::toString () [inline],[virtual]

OSTM required virtual method for display object.

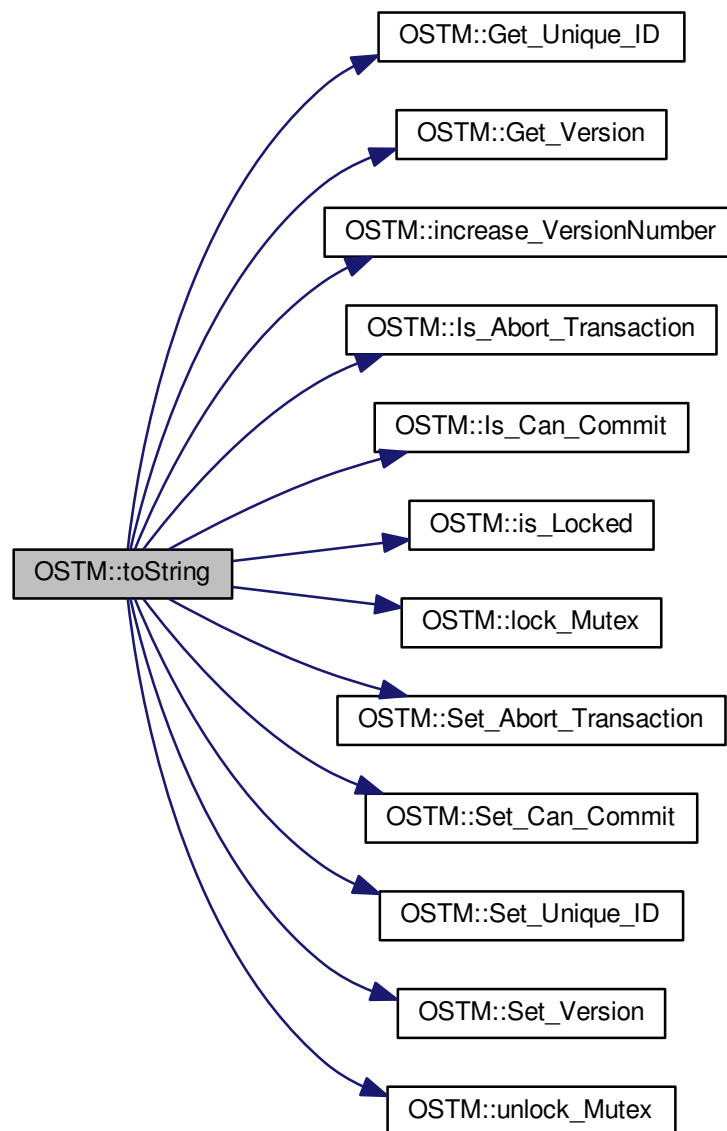
Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), [KILKEANNY_W](#), [AIB](#), [BOA](#), [BOI](#), [SWBPLC](#), [ULSTER](#), and [UNBL](#).

Definition at line 42 of file [OSTM.h](#).

References [canCommit](#), [Get_Unique_ID\(\)](#), [Get_Version\(\)](#), [increase_VersionNumber\(\)](#), [Is_Abort_Transaction\(\)](#), [Is_Can_Commit\(\)](#), [is_Locked\(\)](#), [lock_Mutex\(\)](#), [Set_Abort_Transaction\(\)](#), [Set_Can_Commit\(\)](#), [Set_Unique_ID\(\)](#), [Set_Version\(\)](#), [uniqueID](#), [unlock_Mutex\(\)](#), and [version](#).

```
00042 {};
```

Here is the call graph for this function:



5.9.3.16 void OSTM::unlock_Mutex ()

object unique lock, unlocks mutex

Parameters

<i>mutex</i>	std::mutex
--------------	------------

Definition at line 140 of file [OSTM.cpp](#).

References [mutex](#).

Referenced by [toString\(\)](#).

```
00140     {
00141         this->mutex.unlock();
00142     }
```

5.9.4 Member Data Documentation**5.9.4.1 bool OSTM::abort_Transaction [private]**

Definition at line 108 of file [OSTM.h](#).

Referenced by [Is_Abort_Transaction\(\)](#), [OSTM\(\)](#), and [Set_Abort_Transaction\(\)](#).

5.9.4.2 bool OSTM::canCommit [private]

Definition at line 104 of file [OSTM.h](#).

Referenced by [Is_Can_Commit\(\)](#), [OSTM\(\)](#), [Set_Can_Commit\(\)](#), and [toString\(\)](#).

5.9.4.3 int OSTM::global_Unique_ID_Number = 0 [static], [private]

Unique object number increase at object creation.

Definition at line 112 of file [OSTM.h](#).

Referenced by [Get_global_Unique_ID_Number\(\)](#).

5.9.4.4 std::mutex OSTM::mutex [private]

Object built in lock.

Definition at line 120 of file [OSTM.h](#).

Referenced by [is_Locked\(\)](#), [lock_Mutex\(\)](#), and [unlock_Mutex\(\)](#).

5.9.4.5 int OSTM::uniqueID [private]

Definition at line 100 of file [OSTM.h](#).

Referenced by [Get_Unique_ID\(\)](#), [OSTM\(\)](#), [Set_Unique_ID\(\)](#), and [toString\(\)](#).

5.9.4.6 `int OSTM::version` `[private]`

Definition at line 96 of file [OSTM.h](#).

Referenced by [Get_Version\(\)](#), [increase_VersionNumber\(\)](#), [OSTM\(\)](#), [Set_Version\(\)](#), and [toString\(\)](#).

5.9.4.7 `const int OSTM::ZERO = 0` `[private]`

Meaningful display for value 0.

Definition at line 116 of file [OSTM.h](#).

Referenced by [OSTM\(\)](#).

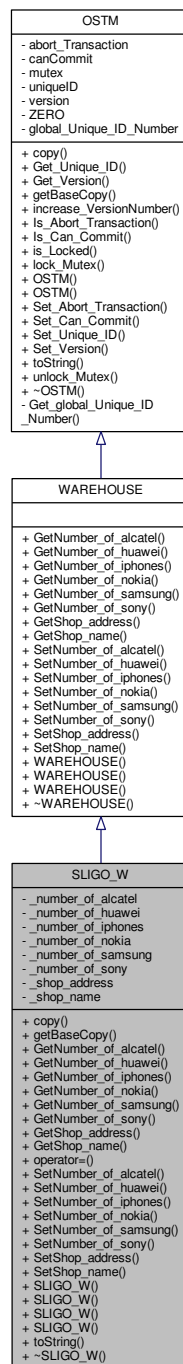
The documentation for this class was generated from the following files:

- [OSTM.h](#)
- [OSTM.cpp](#)

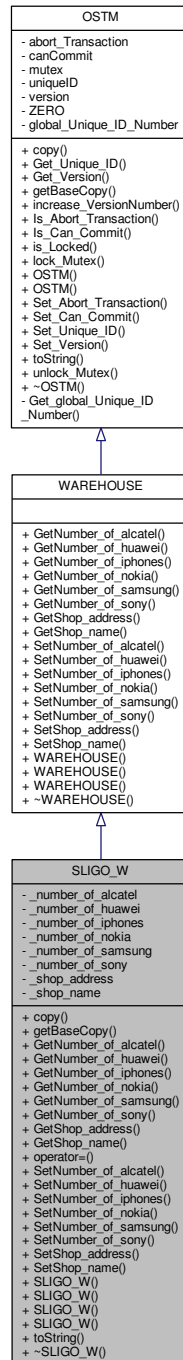
5.10 SLIGO_W Class Reference

```
#include <SLIGO_W.h>
```

Inheritance diagram for SLIGO_W:



Collaboration diagram for SLIGO_W:



Public Member Functions

- virtual void `copy` (std::shared_ptr< `OSTM` > to, std::shared_ptr< `OSTM` > from)
copy function, make deep copy of the object/pointer
- virtual std::shared_ptr< `OSTM` > `getBaseCopy` (std::shared_ptr< `OSTM` > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new BANK type object*
- virtual int `GetNumber_of_alcatel` ()

- virtual int [GetNumber_of_huawei](#) ()
- virtual int [GetNumber_of_iphones](#) ()
- virtual int [GetNumber_of_nokia](#) ()
- virtual int [GetNumber_of_samsung](#) ()
- virtual int [GetNumber_of_sony](#) ()
- virtual std::string [GetShop_address](#) ()
- virtual std::string [GetShop_name](#) ()
- [SLIGO_W](#) operator= (const [SLIGO_W](#) &orig)
- virtual void [SetNumber_of_alcatel](#) (int [_number_of_alcatel](#))
- virtual void [SetNumber_of_huawei](#) (int [_number_of_huawei](#))
- virtual void [SetNumber_of_iphones](#) (int [_number_of_iphones](#))
- virtual void [SetNumber_of_nokia](#) (int [_number_of_nokia](#))
- virtual void [SetNumber_of_samsung](#) (int [_number_of_samsung](#))
- virtual void [SetNumber_of_sony](#) (int [_number_of_sony](#))
- virtual void [SetShop_address](#) (std::string [_shop_address](#))
- virtual void [SetShop_name](#) (std::string [_shop_name](#))
- [SLIGO_W](#) ()
- [SLIGO_W](#) (std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel)
- [SLIGO_W](#) (std::shared_ptr< [WAREHOUSE](#) > obj, int [_version](#), int [_unique_id](#))
- [SLIGO_W](#) (const [SLIGO_W](#) &orig)
- virtual void [toString](#) ()
- *_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type*
- virtual [~SLIGO_W](#) ()

Private Attributes

- int [_number_of_alcatel](#)
- int [_number_of_huawei](#)
- int [_number_of_iphones](#)
- int [_number_of_nokia](#)
- int [_number_of_samsung](#)
- int [_number_of_sony](#)
- std::string [_shop_address](#)
- std::string [_shop_name](#)

5.10.1 Detailed Description

Inherit from [WAREHOUSE](#)

Definition at line 19 of file [SLIGO_W.h](#).

5.10.2 Constructor & Destructor Documentation

5.10.2.1 SLIGO_W::SLIGO_W() [inline]

Constructor

Definition at line 24 of file [SLIGO_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

Referenced by [getBaseCopy\(\)](#), and [SLIGO_W\(\)](#).

```
00024         : WAREHOUSE() {
00025
00026         this->_shop_address = "Sligo River Street";
00027         this->_shop_name = "SLIGO S_WAREHOUSE";
00028         this->_number_of_iphones = 200;
00029         this->_number_of_samsung = 200;
00030         this->_number_of_sony = 200;
00031         this->_number_of_huawei = 200;
00032         this->_number_of_nokia = 200;
00033         this->_number_of_alcatel = 200;
00034     };
```

5.10.2.2 SLIGO_W::SLIGO_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel) [inline]

Custom constructor

Definition at line 38 of file [SLIGO_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

```
00038         : WAREHOUSE() {
00039         /*
00040         * copy over values
00041         */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };
```

5.10.2.3 SLIGO_W::SLIGO_W (std::shared_ptr< WAREHOUSE > obj, int _version, int _unique_id) [inline]

Custom constructor, used by the library for deep copying

Definition at line 55 of file [SLIGO_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), and [SLIGO_W\(\)](#).

```

00055                                     :
00056     WAREHOUSE(_version, _unique_id){
00057         /*
00058             * copy over values
00059
00060             */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }

```

Here is the call graph for this function:



5.10.2.4 SLIGO_W::SLIGO_W (const SLIGO_W & orig)

Copy constructor

Definition at line 15 of file [SLIGO_W.cpp](#).

```

00015                                     {
00016 }

```

5.10.2.5 SLIGO_W::~~SLIGO_W () [virtual]

de-constructor

Definition at line 12 of file [SLIGO_W.cpp](#).

Referenced by [operator=\(\)](#).

```

00012                                     {
00013 }

```

5.10.3 Member Function Documentation

5.10.3.1 void SLIGO_W::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 35 of file [SLIGO_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```

00035                                     {
00036
00037     std::shared_ptr<SLIGO_W> objTO = std::dynamic_pointer_cast<SLIGO_W>(to);
00038     std::shared_ptr<SLIGO_W> objFROM = std::dynamic_pointer_cast<SLIGO_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
```

5.10.3.2 std::shared_ptr<OSTM> SLIGO_W::getBaseCopy (std::shared_ptr<OSTM> *object*) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new BANK* type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a BANK* return type

Reimplemented from [OSTM](#).

Definition at line 22 of file [SLIGO_W.cpp](#).

References [SLIGO_W\(\)](#).

Referenced by [operator=\(\)](#).

```

00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new SLIGO\_W(objTO, object->Get_Version(), object->Get_Unique_ID()));
00027     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00028
00029     return ostm_obj;
00029 }
```

Here is the call graph for this function:



5.10.3.3 `int SLIGO_W::GetNumber_of_alcatel () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 73 of file [SLIGO_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00073 {  
00074     return _number_of_alcatel;  
00075 }
```

5.10.3.4 `int SLIGO_W::GetNumber_of_huawei () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 89 of file [SLIGO_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00089 {  
00090     return _number_of_huawei;  
00091 }
```

5.10.3.5 `int SLIGO_W::GetNumber_of_iphones () [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 113 of file [SLIGO_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00113 {  
00114     return _number_of_iphones;  
00115 }
```

5.10.3.6 int SLIGO_W::GetNumber_of_nokia () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 81 of file [SLIGO_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00081                                     {  
00082     return _number_of_nokia;  
00083 }
```

5.10.3.7 int SLIGO_W::GetNumber_of_samsung () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 105 of file [SLIGO_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00105                                     {  
00106     return _number_of_samsung;  
00107 }
```

5.10.3.8 int SLIGO_W::GetNumber_of_sony () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 97 of file [SLIGO_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00097                                     {  
00098     return _number_of_sony;  
00099 }
```

5.10.3.9 std::string SLIGO_W::GetShop_address () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 129 of file [SLIGO_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00129                                     {  
00130     return _shop_address;  
00131 }
```

5.10.3.10 `std::string SLIGO_W::GetShop_name ()` [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 121 of file [SLIGO_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00121                                     {
00122     return _shop_name;
00123 }
```

5.10.3.11 `SLIGO_W SLIGO_W::operator= (const SLIGO_W & orig)` [inline]

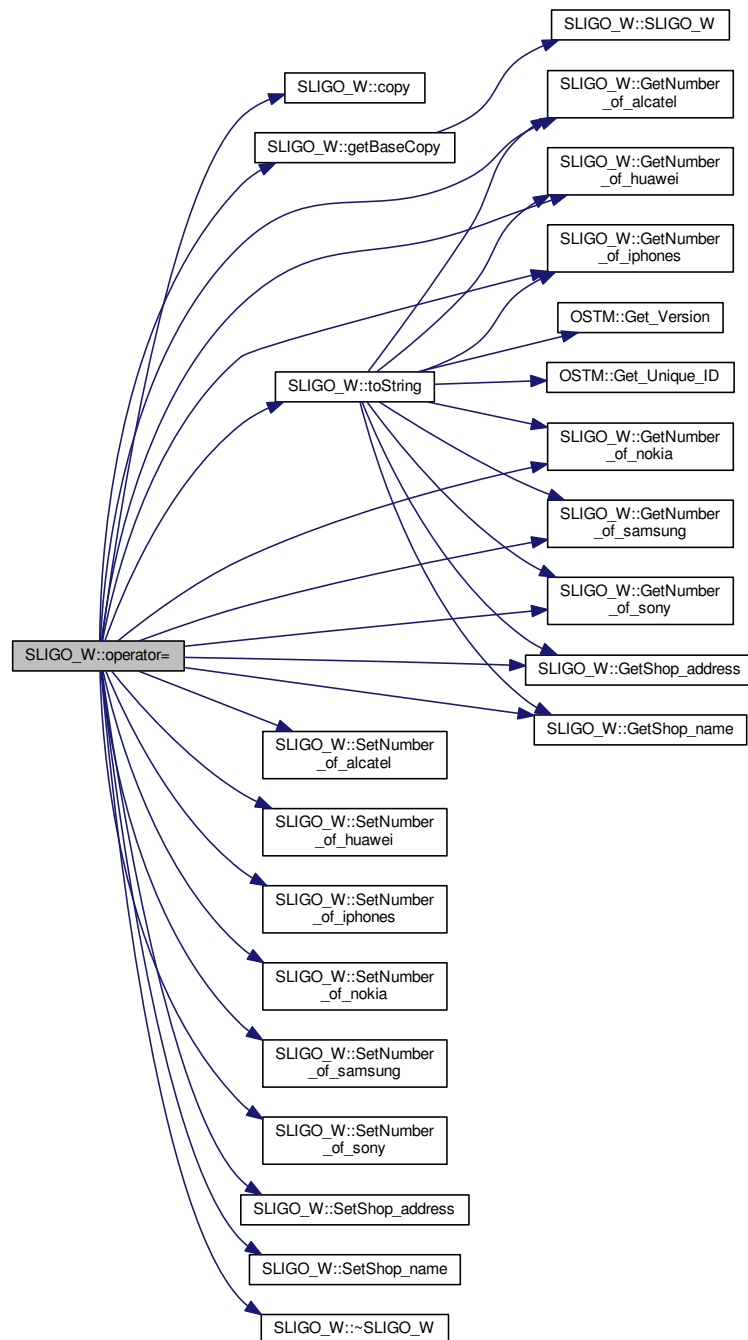
Operator

Definition at line 75 of file [SLIGO_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), [copy\(\)](#), [getBaseCopy\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), [GetShop_name\(\)](#), [SetNumber_of_alcatel\(\)](#), [SetNumber_of_huawei\(\)](#), [SetNumber_of_iphones\(\)](#), [SetNumber_of_nokia\(\)](#), [SetNumber_of_samsung\(\)](#), [SetNumber_of_sony\(\)](#), [SetShop_address\(\)](#), [SetShop_name\(\)](#), [toString\(\)](#), and [~SLIGO_W\(\)](#).

```
00075 {};
```

Here is the call graph for this function:



5.10.3.12 `void SLIGO_W::SetNumber_of_alcatel (int _number_of_alcatel) [virtual]`

Reimplemented from [WAREHOUSE](#).

Definition at line 69 of file [SLIGO_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#).


```

00069                                     {
00070         this->_number_of_alcatel = _number_of_alcatel;
00071     }

```

5.10.3.13 void SLIGO_W::SetNumber_of_huawei (int *_number_of_huawei*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 85 of file [SLIGO_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#).

```

00085                                     {
00086         this->_number_of_huawei = _number_of_huawei;
00087     }

```

5.10.3.14 void SLIGO_W::SetNumber_of_iphones (int *_number_of_iphones*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 109 of file [SLIGO_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#).

```

00109                                     {
00110         this->_number_of_iphones = _number_of_iphones;
00111     }

```

5.10.3.15 void SLIGO_W::SetNumber_of_nokia (int *_number_of_nokia*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 77 of file [SLIGO_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#).

```

00077                                     {
00078         this->_number_of_nokia = _number_of_nokia;
00079     }

```

5.10.3.16 void SLIGO_W::SetNumber_of_samsung (int *_number_of_samsung*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 101 of file [SLIGO_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#).

```

00101                                     {
00102         this->_number_of_samsung = _number_of_samsung;
00103     }

```

5.10.3.17 void SLIGO_W::SetNumber_of_sony (int *_number_of_sony*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 93 of file [SLIGO_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#).

```
00093                                     {  
00094     this->_number_of_sony = _number_of_sony;  
00095 }
```

5.10.3.18 void SLIGO_W::SetShop_address (std::string *_shop_address*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 125 of file [SLIGO_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```
00125                                     {  
00126     this->_shop_address = _shop_address;  
00127 }
```

5.10.3.19 void SLIGO_W::SetShop_name (std::string *_shop_name*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 117 of file [SLIGO_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#).

```
00117                                     {  
00118     this->_shop_name = _shop_name;  
00119 }
```

5.10.3.20 void SLIGO_W::toString() [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

toString function, displays the object values in formatted way

Reimplemented from [OSTM](#).

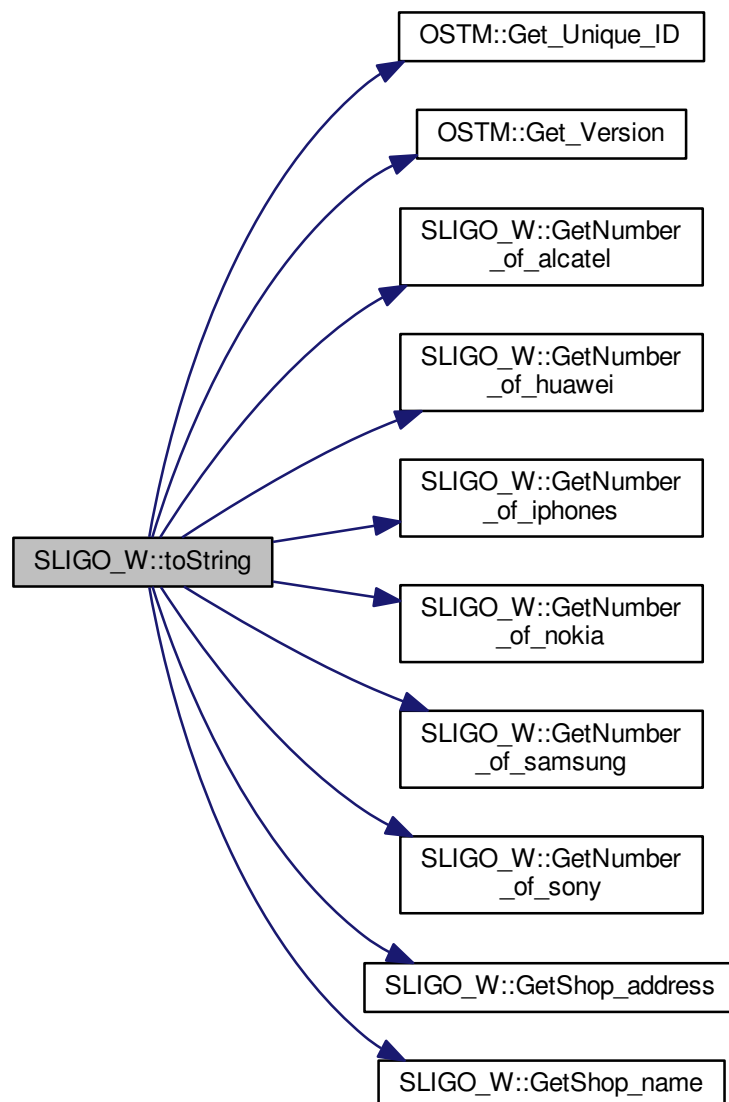
Definition at line 62 of file [SLIGO_W.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), and [GetShop_name\(\)](#).

Referenced by [operator=\(\)](#).

```
00063 {
00064     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
        Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
        << this->GetShop_address() << "\nNo. Iphones : " << this->
        GetNumber_of_iphones() << "\nNo. Samsung : " << this->
        GetNumber_of_samsung() << "\nNo. Sony : " << this->
        GetNumber_of_sony() << "\nNo. Huawei : " << this->
        GetNumber_of_huawei() << "\nNo. Nokia : " << this->
        GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
        GetNumber_of_alcatel() << "\nVersion number : " << this->
        Get_Version() << std::endl;
00065 }
```

Here is the call graph for this function:



5.10.4 Member Data Documentation

5.10.4.1 `int SLIGO_W::_number_of_alcatel` [private]

Definition at line 117 of file [SLIGO_W.h](#).

Referenced by [GetNumber_of_alcatel\(\)](#), [operator=\(\)](#), [SetNumber_of_alcatel\(\)](#), and [SLIGO_W\(\)](#).

5.10.4.2 `int SLIGO_W::_number_of_huawei` [private]

Definition at line 115 of file [SLIGO_W.h](#).

Referenced by [GetNumber_of_huawei\(\)](#), [operator=\(\)](#), [SetNumber_of_huawei\(\)](#), and [SLIGO_W\(\)](#).

5.10.4.3 `int SLIGO_W::_number_of_iphones` `[private]`

Definition at line 112 of file [SLIGO_W.h](#).

Referenced by [GetNumber_of_iphones\(\)](#), [operator=\(\)](#), [SetNumber_of_iphones\(\)](#), and [SLIGO_W\(\)](#).

5.10.4.4 `int SLIGO_W::_number_of_nokia` `[private]`

Definition at line 116 of file [SLIGO_W.h](#).

Referenced by [GetNumber_of_nokia\(\)](#), [operator=\(\)](#), [SetNumber_of_nokia\(\)](#), and [SLIGO_W\(\)](#).

5.10.4.5 `int SLIGO_W::_number_of_samsung` `[private]`

Definition at line 113 of file [SLIGO_W.h](#).

Referenced by [GetNumber_of_samsung\(\)](#), [operator=\(\)](#), [SetNumber_of_samsung\(\)](#), and [SLIGO_W\(\)](#).

5.10.4.6 `int SLIGO_W::_number_of_sony` `[private]`

Definition at line 114 of file [SLIGO_W.h](#).

Referenced by [GetNumber_of_sony\(\)](#), [operator=\(\)](#), [SetNumber_of_sony\(\)](#), and [SLIGO_W\(\)](#).

5.10.4.7 `std::string SLIGO_W::_shop_address` `[private]`

Definition at line 110 of file [SLIGO_W.h](#).

Referenced by [copy\(\)](#), [GetShop_address\(\)](#), [operator=\(\)](#), [SetShop_address\(\)](#), and [SLIGO_W\(\)](#).

5.10.4.8 `std::string SLIGO_W::_shop_name` `[private]`

Definition at line 111 of file [SLIGO_W.h](#).

Referenced by [GetShop_name\(\)](#), [operator=\(\)](#), [SetShop_name\(\)](#), and [SLIGO_W\(\)](#).

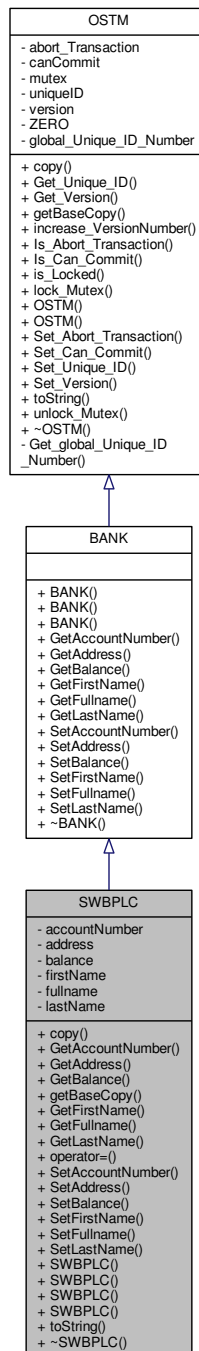
The documentation for this class was generated from the following files:

- [SLIGO_W.h](#)
- [SLIGO_W.cpp](#)

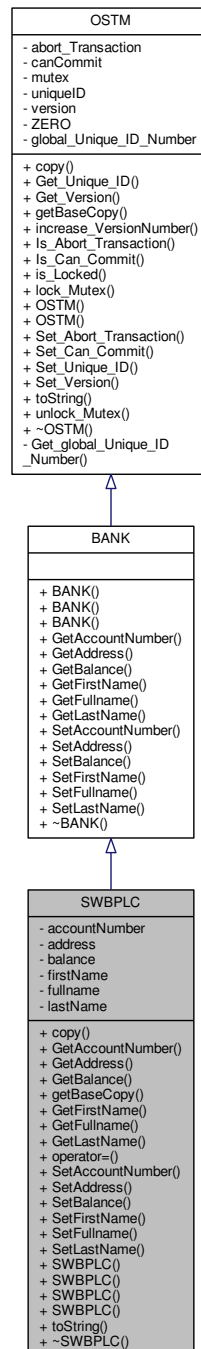
5.11 SWBPLC Class Reference

```
#include <SWBPLC.h>
```

Inheritance diagram for SWBPLC:



Collaboration diagram for SWBPLC:



Public Member Functions

- virtual void [copy](#) (std::shared_ptr< [OSTM](#) > to, std::shared_ptr< [OSTM](#) > from)
copy function, make deep copy of the object/pointer
- virtual int [GetAccountNumber](#) () const
- virtual std::string [GetAddress](#) () const
- virtual double [GetBalance](#) () const

- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object
- virtual std::string [GetFirstName](#) () const
- virtual std::string [GetFullName](#) () const
- virtual std::string [GetLastName](#) () const
- [SWBPLC operator=](#) (const [SWBPLC](#) &orig)
- virtual void [SetAccountNumber](#) (int [accountNumber](#))
- virtual void [SetAddress](#) (std::string [address](#))
- virtual void [SetBalance](#) (double [balance](#))
- virtual void [SetFirstName](#) (std::string [firstName](#))
- virtual void [SetFullName](#) (std::string [fullname](#))
- virtual void [SetLastName](#) (std::string [lastName](#))
- [SWBPLC](#) ()
- [SWBPLC](#) (int [accountNumber](#), double [balance](#), std::string [firstName](#), std::string [lastName](#), std::string [address](#))
- [SWBPLC](#) (std::shared_ptr< [BANK](#) > obj, int _version, int _unique_id)
- [SWBPLC](#) (const [SWBPLC](#) &orig)
- virtual void [toString](#) ()
_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type
- virtual [~SWBPLC](#) ()

Private Attributes

- int [accountNumber](#)
- std::string [address](#)
- double [balance](#)
- std::string [firstName](#)
- std::string [fullname](#)
- std::string [lastName](#)

5.11.1 Detailed Description

Inherit from [BANK](#)

Definition at line 19 of file [SWBPLC.h](#).

5.11.2 Constructor & Destructor Documentation

5.11.2.1 [SWBPLC::SWBPLC](#)() [inline]

Constructor

Definition at line 24 of file [SWBPLC.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

Referenced by [getBaseCopy\(\)](#), and [SWBPLC\(\)](#).

```

00024         : BANK() {
00025             this->accountNumber = 0;
00026             this->balance = 50;
00027             this->firstName = "Joe";
00028             this->lastName = "Blog";
00029             this->address = "High street, Carlow";
00030             this->fullname = firstName + " " + lastName;
00031         };

```


5.11.2.2 SWBPLC::SWBPLC (int *accountNumber*, double *balance*, std::string *firstName*, std::string *lastName*, std::string *address*) [inline]

Custom constructor

Definition at line 35 of file [SWBPLC.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

```
00035
00036     BANK() {
00037         this->accountNumber = accountNumber;
00038         this->balance = balance;
00039         this->firstName = firstName;
00040         this->lastName = lastName;
00041         this->address = address;
00042         this->fullname = firstName + " " + lastName;
00043     };
```

5.11.2.3 SWBPLC::SWBPLC (std::shared_ptr< BANK > *obj*, int *_version*, int *_unique_id*) [inline]

Custom constructor, used by the library for deep copying

Definition at line 46 of file [SWBPLC.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), [lastName](#), and [SWBPLC\(\)](#).

```
00046
00047     : BANK(_version, _unique_id) {
00048         this->accountNumber = obj->GetAccountNumber();
00049         this->balance = obj->GetBalance();
00050         this->firstName = obj->GetFirstName();
00051         this->lastName = obj->GetLastName();
00052         this->address = obj->GetAddress();
00053         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054     };
00055 }
```

Here is the call graph for this function:



5.11.2.4 SWBPLC::SWBPLC (const SWBPLC & *orig*)

Copy constructor

Definition at line 12 of file [SWBPLC.cpp](#).

```
00012     {
00013 }
```

5.11.2.5 SWBPLC::~~SWBPLC () [virtual]

de-constructor

Definition at line 15 of file [SWBPLC.cpp](#).

Referenced by [operator=\(\)](#).

```
00015         {
00016     }
```

5.11.3 Member Function Documentation

5.11.3.1 void SWBPLC::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 34 of file [SWBPLC.cpp](#).

References [OSTM::Set_Unique_ID\(\)](#).

Referenced by [operator=\(\)](#).

```
00034         {
00035
00036     std::shared_ptr<SWBPLC> objTO = std::dynamic_pointer_cast<SWBPLC>(to);
00037     std::shared_ptr<SWBPLC> objFROM = std::dynamic_pointer_cast<SWBPLC>(from);
00038     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00039     objTO->Set_Version(objFROM->Get_Version());
00040     objTO->Set_AccountNumber(objFROM->Get_AccountNumber());
00041     objTO->Set_Balance(objFROM->Get_Balance());
00042
00043
00044 }
```

Here is the call graph for this function:



5.11.3.2 int SWBPLC::GetAccountNumber () const [virtual]

Reimplemented from [BANK](#).

Definition at line 80 of file [SWBPLC.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00080                                     {
00081     return accountNumber;
00082 }
```

5.11.3.3 std::string SWBPLC::GetAddress () const [virtual]

Reimplemented from [BANK](#).

Definition at line 64 of file [SWBPLC.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00064                                     {
00065     return address;
00066 }
```

5.11.3.4 double SWBPLC::GetBalance () const [virtual]

Reimplemented from [BANK](#).

Definition at line 72 of file [SWBPLC.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00072                                     {
00073     return balance;
00074 }
```

5.11.3.5 std::shared_ptr< OSTM > SWBPLC::getBaseCopy (std::shared_ptr< OSTM > object) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a std::shared_ptr<BANK> return type

Reimplemented from [OSTM](#).

Definition at line 22 of file [SWBPLC.cpp](#).

References [SWBPLC\(\)](#).

Referenced by [operator=\(\)](#).

```
00023 {
00024     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00025     std::shared_ptr<BANK> obj(new SWBPLC(objTO, object->Get_Version(), object->Get_Unique_ID()));
00026     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00027     return ostm_obj;
00028 }
```

Here is the call graph for this function:



5.11.3.6 std::string SWBPLC::GetFirstName () const [virtual]

Reimplemented from [BANK](#).

Definition at line 96 of file [SWBPLC.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00096                                     {
00097     return firstName;
00098 }
```

5.11.3.7 std::string SWBPLC::GetFullname () const [virtual]

Reimplemented from [BANK](#).

Definition at line 104 of file [SWBPLC.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00104                                     {
00105     return fullname;
00106 }
```

5.11.3.8 `std::string SWBPLC::GetLastName () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 88 of file [SWBPLC.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00088                                     {
00089     return lastName;
00090 }
```

5.11.3.9 `SWBPLC SWBPLC::operator= (const SWBPLC & orig) [inline]`

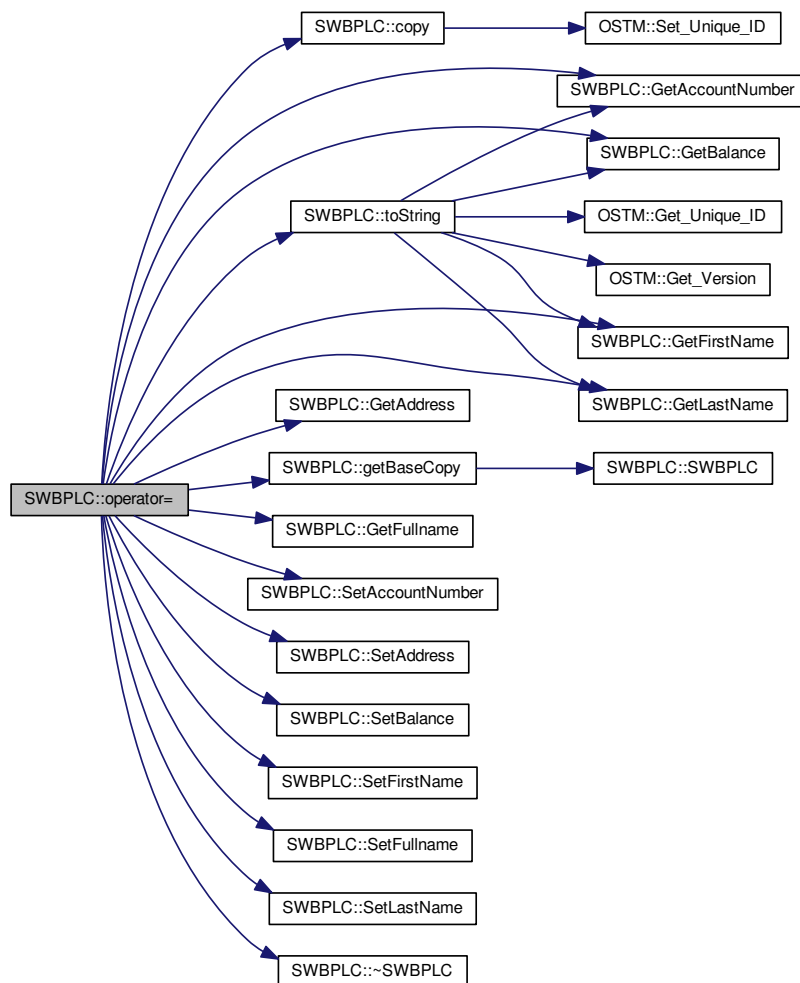
Operator

Definition at line 63 of file [SWBPLC.h](#).

References [accountNumber](#), [address](#), [balance](#), [copy\(\)](#), [firstName](#), [fullname](#), [GetAccountNumber\(\)](#), [GetAddress\(\)](#), [GetBalance\(\)](#), [getBaseCopy\(\)](#), [GetFirstName\(\)](#), [GetFullname\(\)](#), [GetLastName\(\)](#), [lastName](#), [SetAccountNumber\(\)](#), [SetAddress\(\)](#), [SetBalance\(\)](#), [SetFirstName\(\)](#), [SetFullname\(\)](#), [SetLastName\(\)](#), [toString\(\)](#), and [~SWBPLC\(\)](#).

```
00063 {};
```

Here is the call graph for this function:



5.11.3.10 void SWBPLC::SetAccountNumber (int *accountNumber*) [virtual]

Reimplemented from [BANK](#).

Definition at line 76 of file [SWBPLC.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#).

```
00076                                     {
00077     this->accountNumber = accountNumber;
00078 }
```

5.11.3.11 void SWBPLC::SetAddress (std::string *address*) [virtual]

Reimplemented from [BANK](#).

Definition at line 60 of file [SWBPLC.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00060                                     {
00061     this->address = address;
00062 }
```

5.11.3.12 void SWBPLC::SetBalance (double *balance*) [virtual]

Reimplemented from [BANK](#).

Definition at line 68 of file [SWBPLC.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#).

```
00068                                     {
00069     this->balance = balance;
00070 }
```

5.11.3.13 void SWBPLC::SetFirstName (std::string *firstName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 92 of file [SWBPLC.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#).

```
00092                                     {
00093     this->firstName = firstName;
00094 }
```

5.11.3.14 void SWBPLC::SetFullName (std::string *fullname*) [virtual]

Reimplemented from [BANK](#).

Definition at line 100 of file [SWBPLC.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00100      {
00101      this->fullname = fullname;
00102  }
```

5.11.3.15 void SWBPLC::SetLastName (std::string *lastName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 84 of file [SWBPLC.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#).

```
00084      {
00085      this->lastName = lastName;
00086  }
```

5.11.3.16 void SWBPLC::toString () [virtual]

[_cast](#), is use to cast bak the std::shared_ptr<OSTM> to the required type

toString function, displays the object values in formatted way

Reimplemented from [OSTM](#).

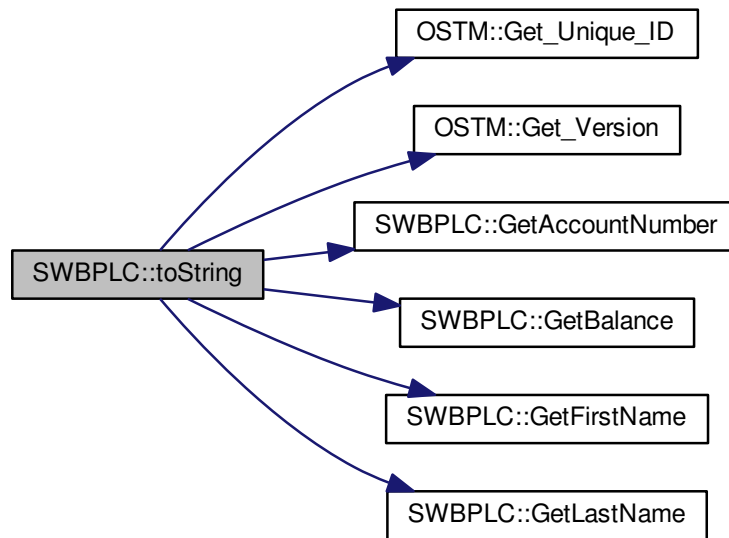
Definition at line 55 of file [SWBPLC.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetAccountNumber\(\)](#), [GetBalance\(\)](#), [GetFirstName\(\)](#), and [GetLastName\(\)](#).

Referenced by [operator=\(\)](#).

```
00056 {
00057     std::cout << "\nSWBPLC BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt
    account : " << this->GetAccountNumber() << "\nDouble value : " << this->
    GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
    this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00058 }
```

Here is the call graph for this function:



5.11.4 Member Data Documentation

5.11.4.1 `int SWBPLC::accountNumber` [private]

Definition at line 96 of file [SWBPLC.h](#).

Referenced by [GetAccountNumber\(\)](#), [operator=\(\)](#), [SetAccountNumber\(\)](#), and [SWBPLC\(\)](#).

5.11.4.2 `std::string SWBPLC::address` [private]

Definition at line 98 of file [SWBPLC.h](#).

Referenced by [GetAddress\(\)](#), [operator=\(\)](#), [SetAddress\(\)](#), and [SWBPLC\(\)](#).

5.11.4.3 `double SWBPLC::balance` [private]

Definition at line 97 of file [SWBPLC.h](#).

Referenced by [GetBalance\(\)](#), [operator=\(\)](#), [SetBalance\(\)](#), and [SWBPLC\(\)](#).

5.11.4.4 `std::string SWBPLC::firstName` [private]

Definition at line 94 of file [SWBPLC.h](#).

Referenced by [GetFirstName\(\)](#), [operator=\(\)](#), [SetFirstName\(\)](#), and [SWBPLC\(\)](#).

5.11.4.5 `std::string SWBPLC::fullName` [private]

Definition at line 93 of file [SWBPLC.h](#).

Referenced by [GetFullName\(\)](#), [operator=\(\)](#), [SetFullName\(\)](#), and [SWBPLC\(\)](#).

5.11.4.6 `std::string SWBPLC::lastName` [private]

Definition at line 95 of file [SWBPLC.h](#).

Referenced by [GetLastName\(\)](#), [operator=\(\)](#), [SetLastName\(\)](#), and [SWBPLC\(\)](#).

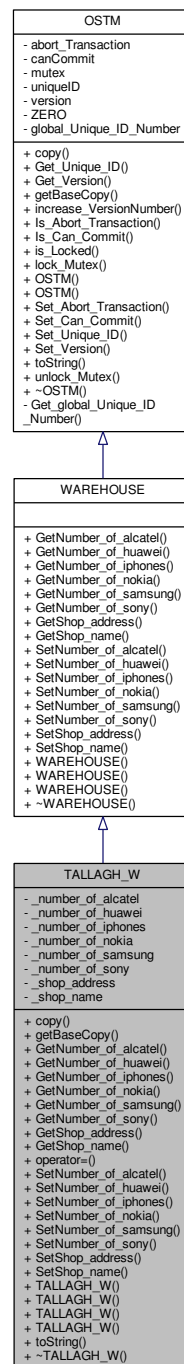
The documentation for this class was generated from the following files:

- [SWBPLC.h](#)
- [SWBPLC.cpp](#)

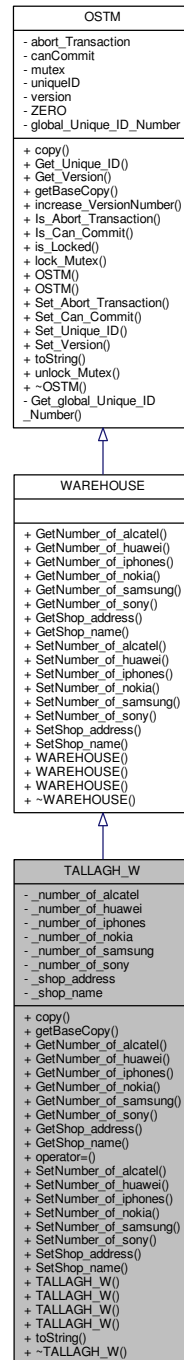
5.12 TALLAGH_W Class Reference

```
#include <TALLAGH_W.h>
```

Inheritance diagram for TALLAGH_W:



Collaboration diagram for TALLAGH_W:



Public Member Functions

- virtual void [copy](#) (std::shared_ptr< [OSTM](#) > to, std::shared_ptr< [OSTM](#) > from)
copy function, make deep copy of the object/pointer
- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new BANK type object*
- virtual int [GetNumber_of_alcatel](#) ()

- virtual int [GetNumber_of_huawei](#) ()
- virtual int [GetNumber_of_iphones](#) ()
- virtual int [GetNumber_of_nokia](#) ()
- virtual int [GetNumber_of_samsung](#) ()
- virtual int [GetNumber_of_sony](#) ()
- virtual std::string [GetShop_address](#) ()
- virtual std::string [GetShop_name](#) ()
- [TALLAGH_W](#) operator= (const [TALLAGH_W](#) &orig)
- virtual void [SetNumber_of_alcatel](#) (int [_number_of_alcatel](#))
- virtual void [SetNumber_of_huawei](#) (int [_number_of_huawei](#))
- virtual void [SetNumber_of_iphones](#) (int [_number_of_iphones](#))
- virtual void [SetNumber_of_nokia](#) (int [_number_of_nokia](#))
- virtual void [SetNumber_of_samsung](#) (int [_number_of_samsung](#))
- virtual void [SetNumber_of_sony](#) (int [_number_of_sony](#))
- virtual void [SetShop_address](#) (std::string [_shop_address](#))
- virtual void [SetShop_name](#) (std::string [_shop_name](#))
- [TALLAGH_W](#) ()
- [TALLAGH_W](#) (std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel)
- [TALLAGH_W](#) (std::shared_ptr< [WAREHOUSE](#) > obj, int [_version](#), int [_unique_id](#))
- [TALLAGH_W](#) (const [TALLAGH_W](#) &orig)
- virtual void [toString](#) ()
- *_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type*
- virtual [~TALLAGH_W](#) ()

Private Attributes

- int [_number_of_alcatel](#)
- int [_number_of_huawei](#)
- int [_number_of_iphones](#)
- int [_number_of_nokia](#)
- int [_number_of_samsung](#)
- int [_number_of_sony](#)
- std::string [_shop_address](#)
- std::string [_shop_name](#)

5.12.1 Detailed Description

Inherit from [WAREHOUSE](#)

Definition at line 19 of file [TALLAGH_W.h](#).

5.12.2 Constructor & Destructor Documentation

5.12.2.1 TALLAGH_W::TALLAGH_W() [inline]

Constructor

Definition at line 24 of file [TALLAGH_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

Referenced by [getBaseCopy\(\)](#), and [TALLAGH_W\(\)](#).

```

00024         : WAREHOUSE() {
00025
00026     this->_shop_address = "Tallagh Low street";
00027     this->_shop_name = "TALLAGH T_WAREHOUSE";
00028     this->_number_of_iphones = 200;
00029     this->_number_of_samsung = 200;
00030     this->_number_of_sony = 200;
00031     this->_number_of_huawei = 200;
00032     this->_number_of_nokia = 200;
00033     this->_number_of_alcatel = 200;
00034 };

```

5.12.2.2 TALLAGH_W::TALLAGH_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int huawei, int nokia, int alcatel) [inline]

Custom constructor

Definition at line 38 of file [TALLAGH_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), and [_shop_name](#).

```

00038         : WAREHOUSE() {
00039
00039     /*
00040     * copy over values
00041     */
00042     this->_shop_address = address;
00043     this->_shop_name = shop_name;
00044     this->_number_of_iphones = iphone;
00045     this->_number_of_samsung = samsung;
00046     this->_number_of_sony = sony;
00047     this->_number_of_huawei = huawei;
00048     this->_number_of_nokia = nokia;
00049     this->_number_of_alcatel = alcatel;
00050
00051 };

```

5.12.2.3 TALLAGH_W::TALLAGH_W (std::shared_ptr< WAREHOUSE > obj, int _version, int _unique_id) [inline]

Custom constructor, used by the library for deep copying

Definition at line 55 of file [TALLAGH_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), and [TALLAGH_W\(\)](#).

```

00055                                     :
00056     WAREHOUSE(_version, _unique_id){
00057         /*
00058             * copy over values
00059             */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }

```

Here is the call graph for this function:



5.12.2.4 TALLAGH_W::TALLAGH_W (const TALLAGH_W & orig)

Copy constructor

Definition at line 15 of file [TALLAGH_W.cpp](#).

```

00015                                     {
00016 }

```

5.12.2.5 TALLAGH_W::~~TALLAGH_W () [virtual]

de-constructor

Definition at line 12 of file [TALLAGH_W.cpp](#).

Referenced by [operator=\(\)](#).

```

00012                                     {
00013 }

```

5.12.3 Member Function Documentation

5.12.3.1 void TALLAGH_W::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a BANK* type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 35 of file [TALLAGH_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```

00035                                     {
00036
00037     std::shared_ptr<TALLAGH_W> objTO = std::dynamic_pointer_cast<TALLAGH_W>(to);
00038     std::shared_ptr<TALLAGH_W> objFROM = std::dynamic_pointer_cast<TALLAGH_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
```

5.12.3.2 std::shared_ptr<OSTM> TALLAGH_W::getBaseCopy (std::shared_ptr<OSTM> *object*) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new BANK* type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a BANK* return type

Reimplemented from [OSTM](#).

Definition at line 22 of file [TALLAGH_W.cpp](#).

References [TALLAGH_W\(\)](#).

Referenced by [operator=\(\)](#).

```

00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new TALLAGH\_W(objTO, object->Get_Version(), object->
    Get_Unique_ID()));
00027     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00028
00028     return ostm_obj;
00029 }
```

Here is the call graph for this function:



5.12.3.3 int TALLAGH_W::GetNumber_of_alcatel () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 71 of file [TALLAGH_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```

00071                                     {
00072     return _number_of_alcatel;
00073 }
```

5.12.3.4 int TALLAGH_W::GetNumber_of_huawei () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 87 of file [TALLAGH_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```

00087                                     {
00088     return _number_of_huawei;
00089 }
```

5.12.3.5 int TALLAGH_W::GetNumber_of_iphones () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 111 of file [TALLAGH_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```

00111                                     {
00112     return _number_of_iphones;
00113 }
```


5.12.3.6 int TALLAGH_W::GetNumber_of_nokia () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 79 of file [TALLAGH_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00079                                     {  
00080     return _number_of_nokia;  
00081 }
```

5.12.3.7 int TALLAGH_W::GetNumber_of_samsung () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 103 of file [TALLAGH_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00103                                     {  
00104     return _number_of_samsung;  
00105 }
```

5.12.3.8 int TALLAGH_W::GetNumber_of_sony () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 95 of file [TALLAGH_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00095                                     {  
00096     return _number_of_sony;  
00097 }
```

5.12.3.9 std::string TALLAGH_W::GetShop_address () [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 127 of file [TALLAGH_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00127                                     {  
00128     return _shop_address;  
00129 }
```

5.12.3.10 `std::string TALLAGH_W::GetShop_name ()` [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 119 of file [TALLAGH_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00119                                     {
00120     return _shop_name;
00121 }
```

5.12.3.11 `TALLAGH_W TALLAGH_W::operator= (const TALLAGH_W & orig)` [inline]

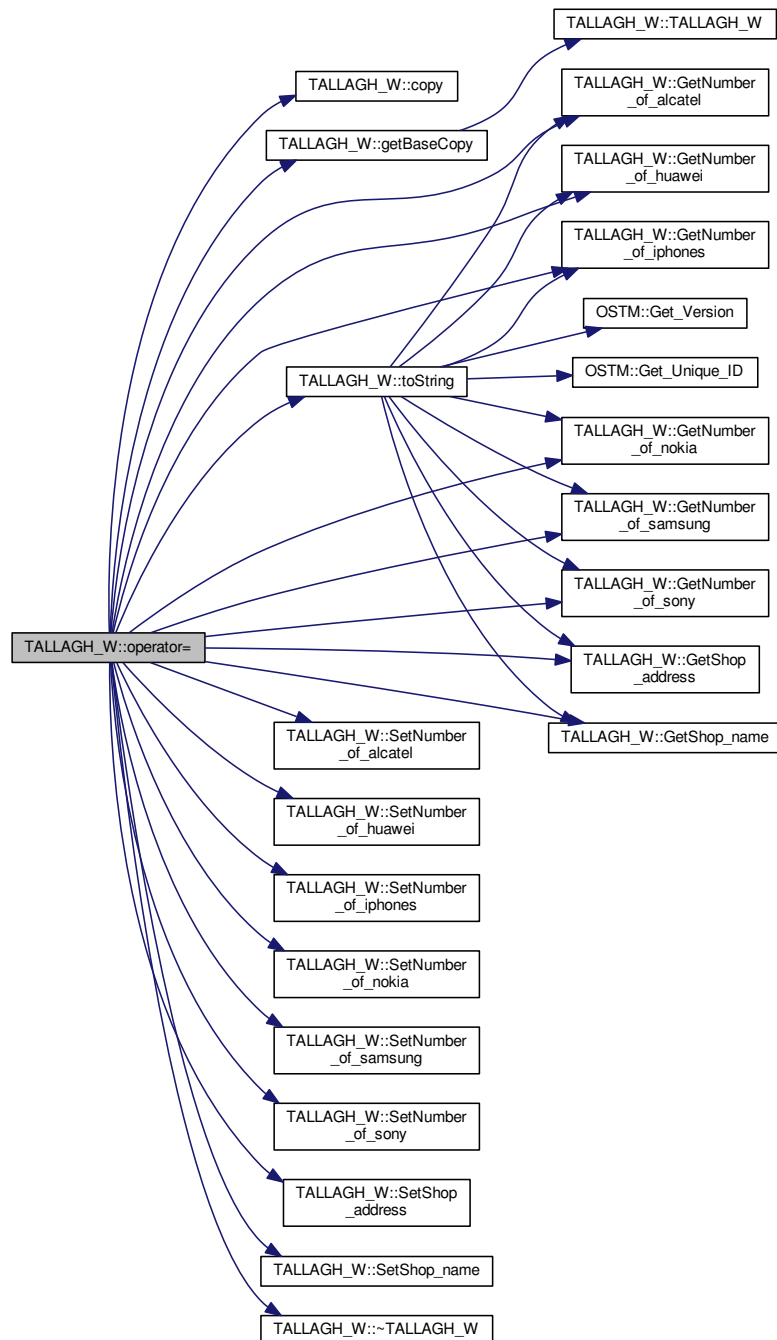
Operator

Definition at line 75 of file [TALLAGH_W.h](#).

References [_number_of_alcatel](#), [_number_of_huawei](#), [_number_of_iphones](#), [_number_of_nokia](#), [_number_of_samsung](#), [_number_of_sony](#), [_shop_address](#), [_shop_name](#), [copy\(\)](#), [getBaseCopy\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), [GetShop_name\(\)](#), [SetNumber_of_alcatel\(\)](#), [SetNumber_of_huawei\(\)](#), [SetNumber_of_iphones\(\)](#), [SetNumber_of_nokia\(\)](#), [SetNumber_of_samsung\(\)](#), [SetNumber_of_sony\(\)](#), [SetShop_address\(\)](#), [SetShop_name\(\)](#), [toString\(\)](#), and [~TALLAGH_W\(\)](#).

```
00075 {};
```

Here is the call graph for this function:



5.12.3.12 `void TALLAGH_W::SetNumber_of_alcatel (int number_of_alcatel)` [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 67 of file [TALLAGH_W.cpp](#).

References [_number_of_alcatel](#).

Referenced by [operator=\(\)](#).

```

00067                                     {
00068     this->_number_of_alcatel = _number_of_alcatel;
00069 }

```

5.12.3.13 void TALLAGH_W::SetNumber_of_huawei (int _number_of_huawei) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 83 of file [TALLAGH_W.cpp](#).

References [_number_of_huawei](#).

Referenced by [operator=\(\)](#).

```

00083                                     {
00084     this->_number_of_huawei = _number_of_huawei;
00085 }

```

5.12.3.14 void TALLAGH_W::SetNumber_of_iphones (int _number_of_iphones) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 107 of file [TALLAGH_W.cpp](#).

References [_number_of_iphones](#).

Referenced by [operator=\(\)](#).

```

00107                                     {
00108     this->_number_of_iphones = _number_of_iphones;
00109 }

```

5.12.3.15 void TALLAGH_W::SetNumber_of_nokia (int _number_of_nokia) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 75 of file [TALLAGH_W.cpp](#).

References [_number_of_nokia](#).

Referenced by [operator=\(\)](#).

```

00075                                     {
00076     this->_number_of_nokia = _number_of_nokia;
00077 }

```

5.12.3.16 void TALLAGH_W::SetNumber_of_samsung (int _number_of_samsung) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 99 of file [TALLAGH_W.cpp](#).

References [_number_of_samsung](#).

Referenced by [operator=\(\)](#).

```

00099                                     {
00100     this->_number_of_samsung = _number_of_samsung;
00101 }

```

5.12.3.17 void TALLAGH_W::SetNumber_of_sony (int *_number_of_sony*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 91 of file [TALLAGH_W.cpp](#).

References [_number_of_sony](#).

Referenced by [operator=\(\)](#).

```
00091                                     {
00092     this->_number_of_sony = _number_of_sony;
00093 }
```

5.12.3.18 void TALLAGH_W::SetShop_address (std::string *_shop_address*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 123 of file [TALLAGH_W.cpp](#).

References [_shop_address](#).

Referenced by [operator=\(\)](#).

```
00123                                     {
00124     this->_shop_address = _shop_address;
00125 }
```

5.12.3.19 void TALLAGH_W::SetShop_name (std::string *_shop_name*) [virtual]

Reimplemented from [WAREHOUSE](#).

Definition at line 115 of file [TALLAGH_W.cpp](#).

References [_shop_name](#).

Referenced by [operator=\(\)](#).

```
00115                                     {
00116     this->_shop_name = _shop_name;
00117 }
```

5.12.3.20 void TALLAGH_W::toString() [virtual]

_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type

toString function, displays the object values in formatted way

Reimplemented from [OSTM](#).

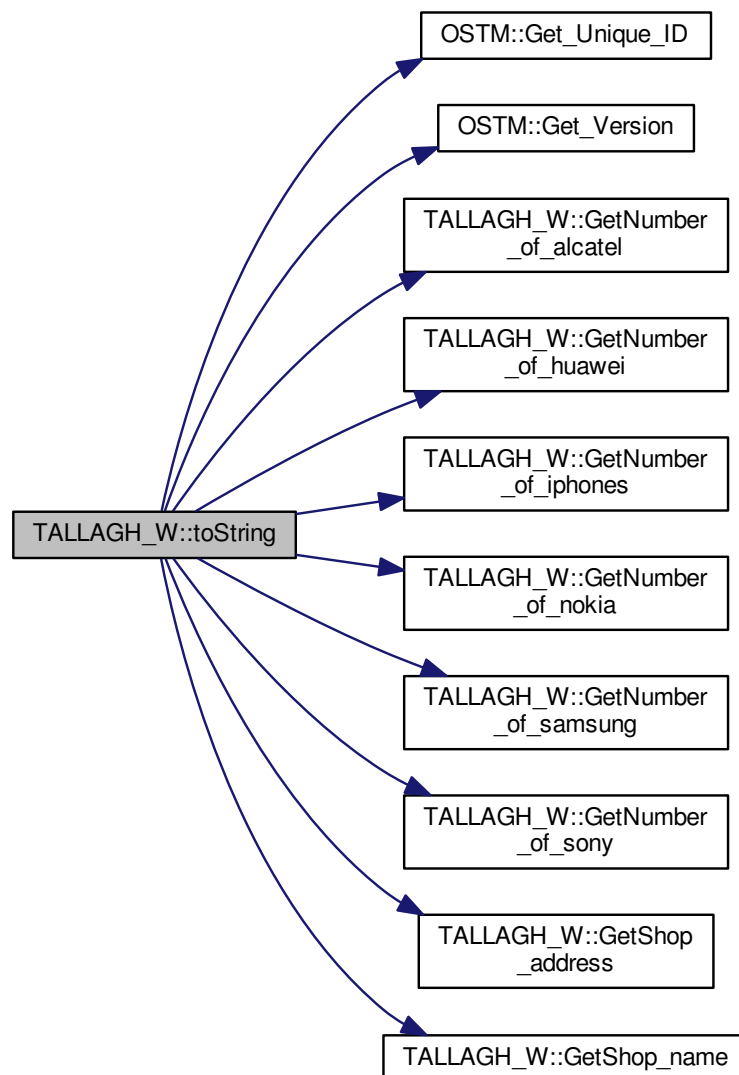
Definition at line 62 of file [TALLAGH_W.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetNumber_of_alcatel\(\)](#), [GetNumber_of_huawei\(\)](#), [GetNumber_of_iphones\(\)](#), [GetNumber_of_nokia\(\)](#), [GetNumber_of_samsung\(\)](#), [GetNumber_of_sony\(\)](#), [GetShop_address\(\)](#), and [GetShop_name\(\)](#).

Referenced by [operator=\(\)](#).

```
00063 {
00064     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
    Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
    << this->GetShop_address() << "\nNo. Iphones : " << this->
    GetNumber_of_iphones() << "\nNo. Samsung : " << this->
    GetNumber_of_samsung() << "\nNo. Sony : " << this->
    GetNumber_of_sony() << "\nNo. Huawei : " << this->
    GetNumber_of_huawei() << "\nNo. Nokia : " << this->
    GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
    GetNumber_of_alcatel() << "\nVersion number : " << this->
    Get_Version() << std::endl;
00065 }
```

Here is the call graph for this function:



5.12.4 Member Data Documentation

5.12.4.1 `int TALLAGH_W::_number_of_alcatel` [private]

Definition at line 117 of file [TALLAGH_W.h](#).

Referenced by [GetNumber_of_alcatel\(\)](#), [operator=\(\)](#), [SetNumber_of_alcatel\(\)](#), and [TALLAGH_W\(\)](#).

5.12.4.2 `int TALLAGH_W::_number_of_huawei` [private]

Definition at line 115 of file [TALLAGH_W.h](#).

Referenced by [GetNumber_of_huawei\(\)](#), [operator=\(\)](#), [SetNumber_of_huawei\(\)](#), and [TALLAGH_W\(\)](#).

5.12.4.3 `int TALLAGH_W::_number_of_iphones` `[private]`

Definition at line 112 of file [TALLAGH_W.h](#).

Referenced by [GetNumber_of_iphones\(\)](#), [operator=\(\)](#), [SetNumber_of_iphones\(\)](#), and [TALLAGH_W\(\)](#).

5.12.4.4 `int TALLAGH_W::_number_of_nokia` `[private]`

Definition at line 116 of file [TALLAGH_W.h](#).

Referenced by [GetNumber_of_nokia\(\)](#), [operator=\(\)](#), [SetNumber_of_nokia\(\)](#), and [TALLAGH_W\(\)](#).

5.12.4.5 `int TALLAGH_W::_number_of_samsung` `[private]`

Definition at line 113 of file [TALLAGH_W.h](#).

Referenced by [GetNumber_of_samsung\(\)](#), [operator=\(\)](#), [SetNumber_of_samsung\(\)](#), and [TALLAGH_W\(\)](#).

5.12.4.6 `int TALLAGH_W::_number_of_sony` `[private]`

Definition at line 114 of file [TALLAGH_W.h](#).

Referenced by [GetNumber_of_sony\(\)](#), [operator=\(\)](#), [SetNumber_of_sony\(\)](#), and [TALLAGH_W\(\)](#).

5.12.4.7 `std::string TALLAGH_W::_shop_address` `[private]`

Definition at line 110 of file [TALLAGH_W.h](#).

Referenced by [copy\(\)](#), [GetShop_address\(\)](#), [operator=\(\)](#), [SetShop_address\(\)](#), and [TALLAGH_W\(\)](#).

5.12.4.8 `std::string TALLAGH_W::_shop_name` `[private]`

Definition at line 111 of file [TALLAGH_W.h](#).

Referenced by [GetShop_name\(\)](#), [operator=\(\)](#), [SetShop_name\(\)](#), and [TALLAGH_W\(\)](#).

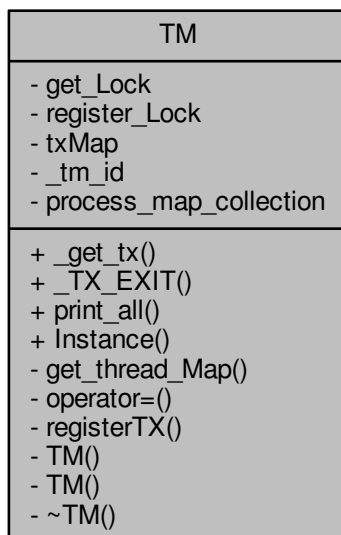
The documentation for this class was generated from the following files:

- [TALLAGH_W.h](#)
- [TALLAGH_W.cpp](#)

5.13 TM Class Reference

```
#include <TM.h>
```

Collaboration diagram for TM:



Public Member Functions

- `std::shared_ptr< TX > const _get_tx ()`
_get_tx std::shared_ptr<TX>, returning a shared pointer with the transaction
- `void _TX_EXIT ()`
_TX_EXIT void, the thread calls the ostm_exit function in the transaction, and clear all elements from the shared global collection associated with the main process
- `void print_all ()`
ONLY FOR TESTING print_all void, print out all object key from txMAP collection.

Static Public Member Functions

- `static TM & Instance ()`
Scott Meyer's Singleton creation, what is thread safe.

Private Member Functions

- `std::map< std::thread::id, int > get_thread_Map ()`
get_thread_Map returning and map to insert to the process_map_collection as an inner value
- `TM & operator= (const TM &)=delete`
[TM](#) copy operator, prevent from copying the Transaction Manager.
- `void registerTX ()`
registerTX void, register transaction into txMap
- `TM ()=default`
[TM](#) constructor, prevent from multiple instantiation.
- `TM (const TM &)=delete`
[TM](#) copy constructor, prevent from copying the Transaction Manager.
- `~TM ()=default`
[TM](#) de-constructor, prevent from deletion.

Private Attributes

- `std::mutex get_Lock`
- `std::mutex register_Lock`
- `std::map< std::thread::id, std::shared_ptr< TX > > txMap`

Static Private Attributes

- `static int _tm_id`
- `static std::map< int, std::map< std::thread::id, int > > process_map_collection`
STATIC GLOBAL MAP Collection to store all process associated keys to find when deleting transactions.

5.13.1 Detailed Description

Definition at line 80 of file [TM.h](#).

5.13.2 Constructor & Destructor Documentation

5.13.2.1 `TM::TM () [private], [default]`

[TM](#) constructor, prevent from multiple instantiation.

5.13.2.2 `TM::~~TM () [private], [default]`

[TM](#) de-constructor, prevent from deletion.

5.13.2.3 `TM::TM (const TM &) [private], [delete]`

[TM](#) copy constructor, prevent from copying the Transaction Manager.

5.13.3 Member Function Documentation

5.13.3.1 `std::shared_ptr< TX > const TM::get_tx ()`

`_get_tx std::shared_ptr<TX>`, returning a shared pointer with the transaction

`_get_tx std::shared_ptr<TX>`, return a shared_ptr with the Transaction object, if [TX](#) not exists then create one, else increasing the nesting level std::mutex, protect shared collection from critical section

Parameters

<code>guard</code>	<code>std::lock_guard<std::mutex></code> , locks the <code>register_Lock</code> mutex, unlock automatically when goes out of the scope
--------------------	--

Definition at line 78 of file [TM.cpp](#).

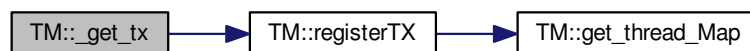
References [get_Lock](#), [registerTX\(\)](#), and [txMap](#).

Referenced by [_complex_transfer_\(\)](#), [_complex_warehouse_transfer_\(\)](#), [_nested_warehouse_transfer_\(\)](#), [_nesting_\(\)](#), [_six_account_transfer_\(\)](#), [_two_account_transfer_\(\)](#), [_warehouse_transfer_\(\)](#), and [main\(\)](#).

```

00079 {
00080     std::lock_guard<std::mutex> guard(get\_Lock);
00081
00082     std::map<std::thread::id, std::shared_ptr<TX>>::iterator it = txMap.find(std::this_thread::get_id(
00083 ));
00084     if(it == txMap.end())
00085     {
00086         registerTX();
00087         it = txMap.find(std::this_thread::get_id());
00088     } else {
00089         it->second->_increase_tx_nesting();
00090     }
00091     //it = txMap.find(std::this_thread::get_id());
00092
00093     return it->second;
00094 }
00095
00096 }
```

Here is the call graph for this function:



5.13.3.2 void TM::_TX_EXIT ()

`_TX_EXIT` void, the thread calls the `ostm_exit` function in the transaction, and clear all elements from the shared global collection associated with the main process

`_TX_EXIT` void, the thread calls the `ostm_exit` function in the transaction, and clear all elements from the shared global collection associated with the main process tx [TX](#), local object to function in transaction

Definition at line 101 of file [TM.cpp](#).

References [TX::ostm_exit\(\)](#), [process_map_collection](#), and [txMap](#).

Referenced by [main\(\)](#).

```

00101         {
00102             TX tx(std::this_thread::get_id());
00103             int ppid = getpid();
00104             std::map<int, std::map< std::thread::id, int >>::iterator process_map_collection_Iterator =
TM::process_map_collection.find(ppid);
00105             if (process_map_collection_Iterator != TM::process_map_collection.end()) {
00106                 for (auto current = process_map_collection_Iterator->second.begin(); current !=
process_map_collection_Iterator->second.end(); ++current) {
00108                     /*
00109                      * Delete all transaction associated with the actual main process
00110                      */
00111                     txMap.erase(current->first);
00112                 }
00113                 TM::process_map_collection.erase(ppid);
00114             }
00115         }
00116         tx.ostm_exit();
00117     }

```

Here is the call graph for this function:



5.13.3.3 `std::map< std::thread::id, int > TM::get_thread_Map ()` [private]

`get_thread_Map` returning and map to insert to the `process_map_collection` as an inner value

`get_thread_Map` `std::map`, returning a map to store all unique ID from all objects from all transactions within the main process

Parameters

<i>thread_Map</i>	<code>std::map< int, int ></code> ,
-------------------	---

Definition at line 133 of file `TM.cpp`.

Referenced by `registerTX()`.

```

00133         {
00134             std::map< std::thread::id, int > thread_Map;
00135             return thread_Map;
00136     }

```

5.13.3.4 `TM & TM::Instance ()` [static]

Scott Meyer's Singleton creation, what is thread safe.

Instance `TM`, return the same singleton object to any process.

Parameters

<code>_instance</code>	TM, static class reference to the instance of the Transaction Manager class
<code>_instance</code>	ppid, assigning the process id whoever created the Singleton instance

Definition at line 28 of file [TM.cpp](#).

References [_tm_id](#).

Referenced by [main\(\)](#).

```

00028     {
00029         static TM _instance;
00030         _instance._tm_id = getpid();
00031     }
00032     return _instance;
00033 }
```

5.13.3.5 TM& TM::operator= (const TM &) [private],[delete]

TM copy operator, prevent from copying the Transaction Manager.

5.13.3.6 void TM::print_all ()

ONLY FOR TESTING print_all void, print out all object key from txMAP collection.

ONLY FOR TESTING print_all void, prints all object in the txMap

Definition at line 121 of file [TM.cpp](#).

References [get_Lock](#), and [txMap](#).

Referenced by [main\(\)](#).

```

00121     {
00122         get_Lock.lock();
00123         for (auto current = txMap.begin(); current != txMap.end(); ++current) {
00124             std::cout << "KEY : " << current->first << std::endl;
00125         }
00126         get_Lock.unlock();
00127     }
```

5.13.3.7 void TM::registerTX() [private]

registerTX void, register transaction into txMap

registerTX void, register a new TX Transaction object into ythe txMap/Transaction Map to manage all the transactions within the shared library

Parameters

<code>txMap</code>	std::map, collection to store all transaction created by the Transaction Manager
<code>register_Lock</code>	std::mutex, used by the lock_guard to protect shared map from race conditions
<code>guard</code>	std::lock_guard, locks the register_Lock mutex, unlock automatically when goes out of the scope

Definition at line 43 of file [TM.cpp](#).

References [get_thread_Map\(\)](#), [process_map_collection](#), [register_Lock](#), and [txMap](#).

Referenced by [_get_tx\(\)](#).

```

00044 {
00045     std::lock_guard<std::mutex> guard(register_Lock);
00046     int ppid = getpid();
00047     std::map<int, std::map< std::thread::id, int >>::iterator process_map_collection_Iterator =
TM::process_map_collection.find(ppid);
00048     if (process_map_collection_Iterator == TM::process_map_collection.end()) {
00049         /*
00050          * Register main process/application to the global map
00051          */
00052         std::map< std::thread::id, int >map = get_thread_Map();
00053         TM::process_map_collection.insert({ppid, map});
00054     }
00055     std::map<std::thread::id, std::shared_ptr < TX>>::iterator it = txMap.find(
std::this_thread::get_id());
00057     if (it == txMap.end()) {
00058         std::shared_ptr<TX> _transaction_object(new TX(std::this_thread::get_id()));
00059         txMap.insert({std::this_thread::get_id(), _transaction_object});
00060         /*
00061          * Get the map if registered first time
00062          */
00063         process_map_collection_Iterator = TM::process_map_collection.find(ppid);
00064         /*
00065          * Insert to the GLOBAL MAP as a helper to clean up at end of main process
00066          */
00067         process_map_collection_Iterator->second.insert({std::this_thread::get_id(), 1});
00068     }
00069 }
00070
00071 }

```

Here is the call graph for this function:



5.13.4 Member Data Documentation

5.13.4.1 int TM::tm_id [static], [private]

Parameters

<code>_tm_id</code>	pid_t, process id determine the actual process between process in the shared OSTM library
---------------------	---

Definition at line 126 of file [TM.h](#).

Referenced by [Instance\(\)](#).

5.13.4.2 `std::mutex TM::get_Lock` [private]

Parameters

<i>register_Lock</i>	std::mutex, used in the <code>_get_tx</code> function
----------------------	---

Definition at line 122 of file [TM.h](#).

Referenced by [_get_tx\(\)](#), and [print_all\(\)](#).

5.13.4.3 `std::map< int, std::map< std::thread::id, int > > TM::process_map_collection` [static], [private]

STATIC GLOBAL MAP Collection to store all process associated keys to find when deleting transactions.

Parameters

<i>process_map_collection</i>	std::map
<i>static</i>	Global std::map process_map_collection store all transactional objects/pointers

Definition at line 106 of file [TM.h](#).

Referenced by [_TX_EXIT\(\)](#), and [registerTX\(\)](#).

5.13.4.4 `std::mutex TM::register_Lock` [private]

Parameters

<i>register_Lock</i>	std::mutex, used in the <code>registerTX</code> function
----------------------	--

Definition at line 118 of file [TM.h](#).

Referenced by [registerTX\(\)](#).

5.13.4.5 `std::map<std::thread::id, std::shared_ptr<TX> > TM::txMap` [private]

Parameters

<i>txMap</i>	std::map, store all transactional objects created with Transaction Manager
--------------	--

Definition at line 101 of file [TM.h](#).

Referenced by [_get_tx\(\)](#), [_TX_EXIT\(\)](#), [print_all\(\)](#), and [registerTX\(\)](#).

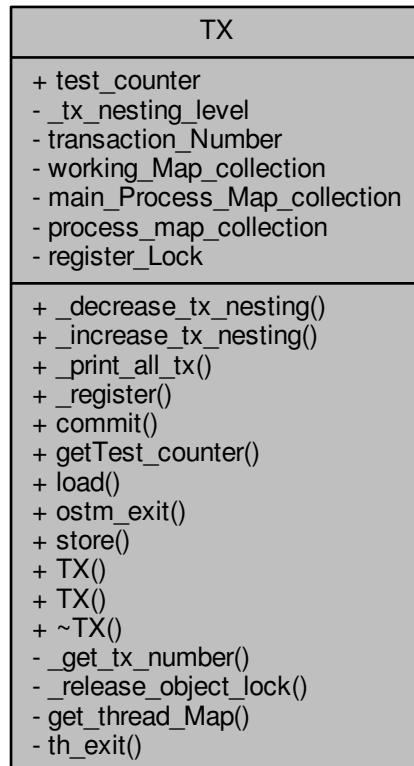
The documentation for this class was generated from the following files:

- [TM.h](#)
- [TM.cpp](#)

5.14 TX Class Reference

```
#include <TX.h>
```

Collaboration diagram for TX:



Public Member Functions

- void [_decrease_tx_nesting](#) ()
Remove TX nesting level by one.
- void [_increase_tx_nesting](#) ()
Add TX nesting level by one.
- void [_print_all_tx](#) ()
- void [_register](#) (std::shared_ptr< [OSTM](#) > object)
Register OSTM pointer into STM library.
- bool [commit](#) ()
Commit transactional changes.
- int [getTest_counter](#) ()
getTest_counter TESTING ONLY!!! returning the value of the test_counter stored, number of rollbacks
- std::shared_ptr< [OSTM](#) > [load](#) (std::shared_ptr< [OSTM](#) > object)
load std::shared_ptr<OSTM>, returning an std::shared_ptr<OSTM> copy of the original pointer, to work with during transaction life time

- void `ostm_exit` ()
Delete all map entries associated with the main process.
- void `store` (std::shared_ptr< `OSTM` > object)
Store transactional changes.
- `TX` (std::thread::id id)
Constructor.
- `TX` (const `TX` &orig)
Default copy constructor.
- `~TX` ()
De-constructor.

Static Public Attributes

- static int `test_counter` = 0

Private Member Functions

- const std::thread::id `_get_tx_number` () const
_get_tx_number returning the transaction unique identifier
- void `_release_object_lock` ()
_release_object_lock void, is get called from commit function, with the purpose to release the locks on all the objects participating in the transaction
- std::map< int, int > `get_thread_Map` ()
get_thread_Map returning and map to insert to the process_map_collection as an inner value
- void `th_exit` ()
Clean up all associated values by the thread delete from working_Map_collection, it is an automated function.

Private Attributes

- int `_tx_nesting_level`
- std::thread::id `transaction_Number`
Returning the transaction number.
- std::map< int, std::shared_ptr< `OSTM` > > `working_Map_collection`
MAP Collection to store OSTM parent based pointers to make invisible changes during isolated transaction.*

Static Private Attributes

- static std::map< int, std::shared_ptr< `OSTM` > > `main_Process_Map_collection`
STATIC GLOBAL MAP Collection to store OSTM parent based pointers to control/lock and compare objects version number within transactions.*
- static std::map< int, std::map< int, int > > `process_map_collection`
STATIC GLOBAL MAP Collection to store all process associated keys to find when deleting transactions.
- static std::mutex `register_Lock`

Friends

- class `TM`

5.14.1 Detailed Description

Definition at line 26 of file [TX.h](#).

5.14.2 Constructor & Destructor Documentation

5.14.2.1 TX::TX (std::thread::id id)

Constructor.

Parameters

<i>transaction_Number</i>	int, to store associated thread
<i>_tx_nesting_level</i>	int, to store and indicate nesting level of transactions within transaction

Definition at line 31 of file [TX.cpp](#).

References [_tx_nesting_level](#), and [transaction_Number](#).

```
00031         {
00032     this->transaction_Number = id;
00033     this->_tx_nesting_level = 0;
00034 }
```

5.14.2.2 TX::~TX ()

De-constructor.

Definition at line 38 of file [TX.cpp](#).

```
00038     {
00039
00040 }
```

5.14.2.3 TX::TX (const TX & orig)

Default copy constructor.

Definition at line 44 of file [TX.cpp](#).

```
00044         {
00045
00046 }
```

5.14.3 Member Function Documentation

5.14.3.1 void TX::_decrease_tx_nesting ()

Remove [TX](#) nesting level by one.

[_decrease_tx_nesting](#) decrease the value stored in [_tx_nesting_level](#) by one, when outer transactions committing

Parameters

<code>_tx_nesting_level</code>	int
--------------------------------	-----

Definition at line 316 of file [TX.cpp](#).

References [_tx_nesting_level](#).

Referenced by [commit\(\)](#).

```
00316                                     {
00317     // std::cout << "[this->_tx_nesting_level] = " << this->_tx_nesting_level << std::endl;
00318     this->_tx_nesting_level -= 1;
00319 ;
00320 }
```

5.14.3.2 const std::thread::id TX::_get_tx_number () const [private]

[_get_tx_number](#) returning the transaction unique identifier

[_get_tx_number](#) std::thread::id, returning the thread id that has assigned the given transaction

Parameters

<code>transaction_Number</code>	int
---------------------------------	-----

Definition at line 331 of file [TX.cpp](#).

References [transaction_Number](#).

```
00331                                     {
00332     return transaction_Number;
00333 }
```

5.14.3.3 void TX::_increase_tx_nesting ()

Add [TX](#) nesting level by one.

[_increase_tx_nesting](#) increase the value stored in [_tx_nesting_level](#) by one, indicate that the transaction nested

Parameters

<code>_tx_nesting_level</code>	int
--------------------------------	-----

Definition at line 307 of file [TX.cpp](#).

References [_tx_nesting_level](#).

```
00307                                     {
00308
00309     this->_tx_nesting_level += 1;
00310     // std::cout << "[this->_tx_nesting_level] = " << this->_tx_nesting_level << std::endl;
00311 }
```

5.14.3.4 void TX::_print_all_tx ()

ONLY FOR TESTING CHECK THE MAP AFTER THREAD EXIT AND ALL SHOULD BE DELETED!!!!!!

Definition at line 346 of file TX.cpp.

References [process_map_collection](#), and [working_Map_collection](#).

```

00346         {
00347
00348         std::cout << "[PRINTALLTHREAD]" << std::endl;
00349         std::map< int, std::shared_ptr<OSTM> >::iterator it;
00350         /*
00351          * All registered thread id in the TX global
00352          */
00353         int ppid = getpid();
00354         std::map<int, std::map< int, int >::iterator process_map_collection_Iterator =
TX::process_map_collection.find(ppid);
00355         if (process_map_collection_Iterator != TX::process_map_collection.end()) {
00356
00357             for (auto current = process_map_collection_Iterator->second.begin(); current !=
process_map_collection_Iterator->second.end(); ++current) {
00358                 it = working_Map_collection.find(current->first);
00359                 if(it != working_Map_collection.end()){
00360                     std::cout << "[Unique number ] : " <<it->second->Get_Unique_ID() << std::endl;
00361                 }
00362             }
00363         }
00364     }
00365 }
00366 }
00367 }
```

5.14.3.5 void TX::_register (std::shared_ptr< OSTM > object)

Register [OSTM](#) pointer into STM library.

register void, receives an std::shared_ptr<OSTM> that point to the original memory space to protect from reca conditions

Parameters

<i>working_Map_collection</i>	std::map, store all the std::shared_ptr<OSTM> pointer in the transaction
<i>main_Process_Map_collection</i>	std::map, store all std::shared_ptr<OSTM> from all transaction, used to lock and compare the objects
<i>process_map_collection</i>	std::map, store all std::shared_ptr<OSTM> unique ID from all transaction, used to delete all pointers used by the main process, from all transaction before the program exit.
<i>std::lock_guard</i>	use register_Lock(mutex) shared lock between all transaction
<i>ppid</i>	int, store main process number

Definition at line 104 of file TX.cpp.

References [get_thread_Map\(\)](#), [main_Process_Map_collection](#), [process_map_collection](#), [register_Lock](#), and [working_Map_collection](#).

```

00104         {
00105         /*
00106          * MUST USE SHARED LOCK TO PROTECT SHARED GLOBAL MAP/COLLECTION
00107          */
00108         std::lock_guard<std::mutex> guard(TX::register_Lock);
```

```

00109
00110     /*
00111     * Check for null pointer !
00112
00112     * Null pointer can cause segmentation fault!!!
00113
00113     */
00114     if(object == nullptr){
00115         throw std::runtime_error(std::string("[RUNTIME ERROR : NULL POINTER IN REGISTER FUNCTION]") );
00116     }
00117
00118     int ppid = getpid();
00119     std::map<int, std::map< int, int >>::iterator process_map_collection_Iterator =
TX::process_map_collection.find(ppid);
00120     if (process_map_collection_Iterator == TX::process_map_collection.end()) {
00121         /*
00122         * Register main process/application to the global map
00123
00123         */
00124         std::map< int, int >map = get_thread_Map();
00125         TX::process_map_collection.insert({ppid, map});
00126         /*
00127         * Get the map if registered first time
00128
00128         */
00129         process_map_collection_Iterator = TX::process_map_collection.find(ppid);
00130     }
00131     std::map<int, std::shared_ptr<OSTM>::iterator main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(object->Get_Unique_ID());
00132     if (main_Process_Map_collection_Iterator == TX::main_Process_Map_collection
.end()) {
00133         /*
00134         * Insert to the GLOBAL MAP
00135
00135         */
00136         TX::main_Process_Map_collection.insert({object->Get_Unique_ID(),
object});
00137         /*
00138         * Insert to the GLOBAL MAP as a helper to clean up at end of main process
00139
00139         */
00140         process_map_collection_Iterator->second.insert({object->Get_Unique_ID(), 1});
00141     }
00142
00143
00144     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator
= working_Map_collection.find(object->Get_Unique_ID());
00145     if (working_Map_collection_Object_Shared_Pointer_Iterator ==
working_Map_collection.end()) {
00146
00147         working_Map_collection.insert({object->Get_Unique_ID(), object->getBaseCopy(
object)});
00148     }
00149
00150 }

```

Here is the call graph for this function:



5.14.3.6 void TX::_release_object_lock() [private]

_release_object_lock void, is get called from commit function, with the purpose to release the locks on all the objects participating in the transaction

Release the locks in objects with transaction associated collection

Parameters

<i>working_Map_collection</i>	std::map, store all the std::shared_ptr<OSTM> pointer in the transaction
<i>main_Process_Map_collection</i>	std::map, store all std::shared_ptr<OSTM> from all transaction, used to release the lock on object

Definition at line 286 of file [TX.cpp](#).

References [main_Process_Map_collection](#), and [working_Map_collection](#).

Referenced by [commit\(\)](#).

```

00286             {
00287
00288         std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00289         std::map<int, std::shared_ptr<OSTM>>::iterator main_Process_Map_collection_Iterator;
00290         for (working_Map_collection_Object_Shared_Pointer_Iterator =
00291             working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
00292             != working_Map_collection.end();
00293             working_Map_collection_Object_Shared_Pointer_Iterator++) {
00291             main_Process_Map_collection_Iterator =
00292             TX::main_Process_Map_collection.find((
00293             working_Map_collection_Object_Shared_Pointer_Iterator->second)->Get_Unique_ID());
00293             if (main_Process_Map_collection_Iterator !=
00294             TX::main_Process_Map_collection.end()) {
00294                 /*
00295                 * Release object lock
00296
00297                 */
00297                 (main_Process_Map_collection_Iterator->second->unlock_Mutex());
00298             }
00299         }
00300     }
00301 }

```

5.14.3.7 bool TX::commit ()

Commit transactional changes.

commit bool, returns boolean value TRUE/FALSE depends on the action taken within the function

Parameters

<i>working_Map_collection</i>	std::map, store all the std::shared_ptr<OSTM> pointer in the transaction
<i>main_Process_Map_collection</i>	std::map, store all std::shared_ptr<OSTM> from all transaction, used to lock and compare the objects
<i>can_Commit</i>	bool, helps to make decision that the transaction can commit or rollback

Definition at line 202 of file [TX.cpp](#).

References [_decrease_tx_nesting\(\)](#), [_release_object_lock\(\)](#), [_tx_nesting_level](#), [main_Process_Map_collection](#), [test_counter](#), [th_exit\(\)](#), and [working_Map_collection](#).

```

00202             {
00203
00204         bool can_Commit = true;
00205
00206         /*
00207         * Dealing with nested transactions first
00208
00209         */

```

```

00209     if (this->_tx_nesting_level > 0) {
00210         _decrease_tx_nesting();
00211         return true;
00212     }
00213
00214     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00215
00216     std::map<int, std::shared_ptr<OSTM>>::iterator main_Process_Map_collection_Iterator;
00217     for (working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
!= working_Map_collection.end();
working_Map_collection_Object_Shared_Pointer_Iterator++) {
00218
00219         main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Unique_ID());
00220         /*
00221          * Throws runtime error if object can not find
00222          */
00223         if(main_Process_Map_collection_Iterator ==
TX::main_Process_Map_collection.end())
00224         {
00225             throw std::runtime_error(std::string("[RUNTIME ERROR : CAN'T FIND OBJECT COMMIT FUNCTION]"));
00226         }
00227
00228         /*
00229          * Busy wait WHILE object locked by other thread
00230          */
00231         while(!(main_Process_Map_collection_Iterator->second->is_Locked()));
00232
00233         if (main_Process_Map_collection_Iterator->second->Get_Version() >
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Version()) {
00234
00235             working_Map_collection_Object_Shared_Pointer_Iterator->second->Set_Can_Commit(false);
00236             can_Commit = false;
00237             break;
00238         } else {
00239
00240             working_Map_collection_Object_Shared_Pointer_Iterator->second->Set_Can_Commit(true);
00241         }
00242     }
00243     if (!can_Commit) {
00244         TX::test_counter += 1;
00245         for (working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
!= working_Map_collection.end();
working_Map_collection_Object_Shared_Pointer_Iterator++) {
00246
00247             main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Unique_ID());
00248             (working_Map_collection_Object_Shared_Pointer_Iterator->second->copy(
working_Map_collection_Object_Shared_Pointer_Iterator->second, main_Process_Map_collection_Iterator->second);
00249
00250         }
00251         _release_object_lock();
00252
00253         return false;
00254     } else {
00255         /*
00256          * Commit changes
00257          */
00258         for (working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
!= working_Map_collection.end();
working_Map_collection_Object_Shared_Pointer_Iterator++) {
00260
00261             main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find((
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Unique_ID());
00262             if (main_Process_Map_collection_Iterator !=
TX::main_Process_Map_collection.end()) {
00263
00264                 (main_Process_Map_collection_Iterator->second->copy(
main_Process_Map_collection_Iterator->second, working_Map_collection_Object_Shared_Pointer_Iterator->second);
00265                 main_Process_Map_collection_Iterator->second->increase_VersionNumber();
00266
00267             } else {
00268                 throw std::runtime_error(std::string("[RUNTIME ERROR : CAN'T FIND OBJECT COMMIT
00269 FUNCTION]"));
00270

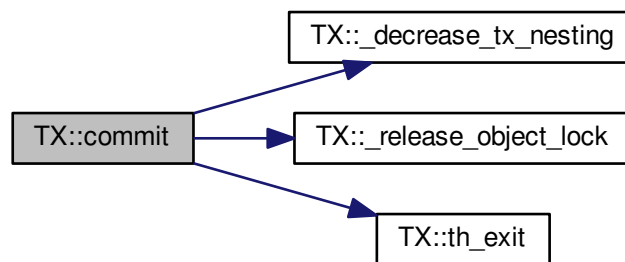
```

```

00271         }
00272     }
00273
00274
00275     _release_object_lock();
00276     this->th_exit();
00277     return true;
00278 }
00279 } //Commit finish

```

Here is the call graph for this function:



5.14.3.8 `std::map< int, int > TX::get_thread_Map ()` [private]

`get_thread_Map` returning and map to insert to the `process_map_collection` as an inner value

`get_thread_Map` `std::map`, returning a map to store all unique ID from all objects from all transactions within the main process

Parameters

<i>thread_Map</i>	<code>std::map< int, int ></code> ,
-------------------	---

Definition at line 338 of file [TX.cpp](#).

Referenced by [_register\(\)](#).

```

00338
00339     std::map< int, int > thread_Map;
00340     return thread_Map;
00341 }

```

5.14.3.9 `int TX::getTest_counter ()`

`getTest_counter` TESTING ONLY!!! returning the value of the `test_counter` stored, number of rollbacks

Definition at line 324 of file [TX.cpp](#).

References [test_counter](#).

```

00324
00325     return TX::test_counter;
00326 }

```


5.14.3.10 `std::shared_ptr<OSTM> TX::load (std::shared_ptr<OSTM> object)`

load `std::shared_ptr<OSTM>`, returning an `std::shared_ptr<OSTM>` copy of the original pointer, to work with during transaction life time

Register [OSTM](#) pointer into STM library

Parameters

<i>working_Map_collection</i>	<code>std::map</code> , store all the <code>std::shared_ptr<OSTM></code> pointer in the transaction
-------------------------------	---

Definition at line 155 of file [TX.cpp](#).

References [working_Map_collection](#).

```

00155                                     {
00156
00157     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00158     /*
00159     * Check for null pointer !
00160
00161     * Null pointer can cause segmentation fault!!!
00162
00163     */
00164     if(object == nullptr){
00165         throw std::runtime_error(std::string("[RUNTIME ERROR : NULL POINTER IN LOAD FUNCTION]") );
00166     }
00167     working_Map_collection_Object_Shared_Pointer_Iterator =
00168     working_Map_collection.find(object->Get_Unique_ID());
00169     if (working_Map_collection_Object_Shared_Pointer_Iterator !=
00170     working_Map_collection.end()) {
00171         return working_Map_collection_Object_Shared_Pointer_Iterator->second->getBaseCopy (
00172         working_Map_collection_Object_Shared_Pointer_Iterator->second);
00173     } else { throw std::runtime_error(std::string("[RUNTIME ERROR : NO OBJECT FOUND LOAD FUNCTION]") );}
00174 }

```

5.14.3.11 `void TX::ostm_exit ()`

Delete all map entries associated with the main process.

`ostm_exit` void, clear all elements from the shared global collections associated with the main process

Parameters

<i>main_Process_Map_collection</i>	<code>std::map</code> , store all <code>std::shared_ptr<OSTM></code> from all transaction shared between multiple processes
<i>process_map_collection</i>	<code>std::map</code> , store all unique id from all transaction within main process DO NOT CALL THIS METHOD EXPLICITLY!!!!!! WILL DELETE ALL PROCESS ASSOCIATED ELEMENTS!!!!

Definition at line 72 of file [TX.cpp](#).

References [main_Process_Map_collection](#), and [process_map_collection](#).

Referenced by [TM::_TX_EXIT\(\)](#).

```

00072         {
00073             std::map<int, std::shared_ptr<OSTM>>::iterator main_Process_Map_collection_Iterator;
00074
00075             int ppid = getpid();
00076             std::map<int, std::map< int, int >>::iterator process_map_collection_Iterator =
TX::process_map_collection.find(ppid);
00077             if (process_map_collection_Iterator != TX::process_map_collection.end()) {
00078
00079                 for (auto current = process_map_collection_Iterator->second.begin(); current !=
process_map_collection_Iterator->second.end(); ++current) {
00080                     main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(current->first);
00081
00082                     if (main_Process_Map_collection_Iterator !=
TX::main_Process_Map_collection.end()){
00083                         /*
00084                         * Delete element from shared main_Process_Map_collection by object unique key value,
shared_ptr will destroy automatically
00085
00086                         */
TX::main_Process_Map_collection.erase(
main_Process_Map_collection_Iterator->first);
00087                     }
00088                 }
00089                 /*
00090                 * Delete from Process_map_collection, Main process exits delete association with library
00091
00092                 */
TX::process_map_collection.erase(process_map_collection_Iterator->first);
00093             }
00094         }

```

5.14.3.12 void TX::store (std::shared_ptr< OSTM > object)

Store transactional changes.

store void, receive an std::shared_ptr<OSTM> object to store the changes within the transaction, depends the user action

Parameters

<i>working_Map_collection</i>	std::map, store all the std::shared_ptr<OSTM> pointer in the transaction
-------------------------------	--

Definition at line 178 of file TX.cpp.

References [working_Map_collection](#).

```

00178         {
00179             /*
00180             * Check for null pointer !
00181
00182             * Null pointer can cause segmentation fault!!!
00183
00184             */
00185             if(object == nullptr){
00186                 throw std::runtime_error(std::string("[RUNTIME ERROR : NULL POINTER IN STORE FUNCTION]") );
00187             }
00188             std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00189             working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.find(object->Get_Unique_ID());
00190             if (working_Map_collection_Object_Shared_Pointer_Iterator !=
working_Map_collection.end()) {
00191
00192                 working_Map_collection_Object_Shared_Pointer_Iterator->second = object;
00193             } else { std::cout << "[ERROR STORE]" << std::endl; }
00194         }
00195     }

```

5.14.3.13 void TX::th_exit() [private]

Clean up all associated values by the thread delete from working_Map_collection, it is an automated function.

th_exit void, delete all std::shared_ptr<OSTM> elements from working_Map_collection, that store pointers to working objects

Parameters

<i>working_Map_collection</i>	std::map, store std::shared_ptr<OSTM> transaction pointers
-------------------------------	--

Definition at line 52 of file TX.cpp.

References [_tx_nesting_level](#), and [working_Map_collection](#).

Referenced by [commit\(\)](#).

```

00052         {
00053
00054         if (this->_tx_nesting_level > 0) {
00055             /*
00056              * Active nested transactions running in background, do not delete anything yet
00057              */
00058         } else {
00059             /*
00060              * Remove all elements map entries from transaction and clear the map
00061              */
00062             working_Map_collection.clear();
00063         }
00064     }

```

5.14.4 Friends And Related Function Documentation

5.14.4.1 friend class TM [friend]

Only [TM](#) Transaction Manager can create instance of [TX](#) Transaction

Definition at line 72 of file TX.h.

5.14.5 Member Data Documentation

5.14.5.1 int TX::_tx_nesting_level [private]

Parameters

<i>_tx_nesting_level</i>	int
--------------------------	-----

Definition at line 102 of file TX.h.

Referenced by [_decrease_tx_nesting\(\)](#), [_increase_tx_nesting\(\)](#), [commit\(\)](#), [th_exit\(\)](#), and [TX\(\)](#).

5.14.5.2 `std::map< int, std::shared_ptr< OSTM > > TX::main_Process_Map_collection` `[static], [private]`

STATIC GLOBAL MAP Collection to store OSTM* parent based pointers to control/lock and compare objects version number within transactions.

Parameters

<i>main_Process_Map_collection</i>	<code>std::map</code>
<i>static</i>	Global <code>std::map</code> <code>main_Process_Map_collection</code> store all transactional objects/pointers

Definition at line 108 of file [TX.h](#).

Referenced by [_register\(\)](#), [_release_object_lock\(\)](#), [commit\(\)](#), and [ostm_exit\(\)](#).

5.14.5.3 `std::map< int, std::map< int, int > > TX::process_map_collection` `[static], [private]`

STATIC GLOBAL MAP Collection to store all process associated keys to find when deleting transactions.

Parameters

<i>process_map_collection</i>	<code>std::map</code>
<i>static</i>	Global <code>std::map</code> <code>process_map_collection</code> store all transactional objects/pointers

Definition at line 113 of file [TX.h](#).

Referenced by [_print_all_tx\(\)](#), [_register\(\)](#), and [ostm_exit\(\)](#).

5.14.5.4 `std::mutex TX::register_Lock` `[static], [private]`

Parameters

<i>register_Lock</i>	<code>std::mutex</code> to control shared access on MAIN MAP
<i>static</i>	shared <code>std::mutex</code> <code>register_Lock</code> to protect writes into shared global collection

Definition at line 122 of file [TX.h](#).

Referenced by [_register\(\)](#).

5.14.5.5 `int TX::test_counter = 0` `[static]`

Parameters

<i>test_counter</i>	<code>int</code> ONLY FOR TESTING!!!
<i>static</i>	Global counter for rollback

Definition at line 80 of file [TX.h](#).

Referenced by [commit\(\)](#), and [getTest_counter\(\)](#).

5.14.5.6 `std::thread::id TX::transaction_Number` [private]

Returning the transaction number.

Parameters

<i>transaction_Number</i>	<code>std::thread::id</code> NOT USED YET
---------------------------	---

Definition at line 98 of file [TX.h](#).

Referenced by [_get_tx_number\(\)](#), and [TX\(\)](#).

5.14.5.7 `std::map< int, std::shared_ptr<OSTM> > TX::working_Map_collection` [private]

MAP Collection to store OSTM* parent based pointers to make invisible changes during isolated transaction.

Parameters

<i>working_Map_collection</i>	<code>std::map</code>
-------------------------------	-----------------------

Definition at line 92 of file [TX.h](#).

Referenced by [_print_all_tx\(\)](#), [_register\(\)](#), [_release_object_lock\(\)](#), [commit\(\)](#), [load\(\)](#), [store\(\)](#), and [th_exit\(\)](#).

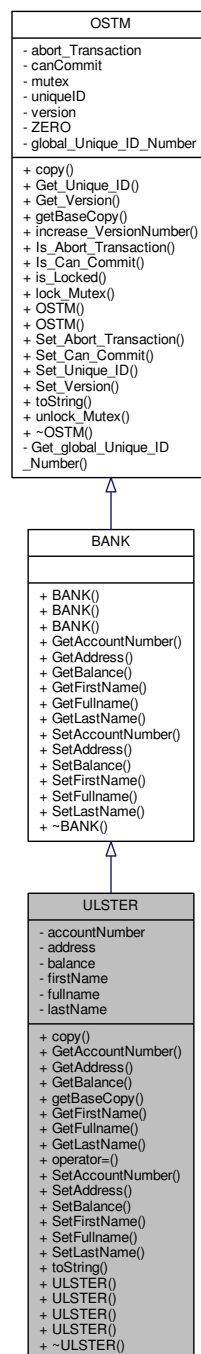
The documentation for this class was generated from the following files:

- [TX.h](#)
- [TX.cpp](#)

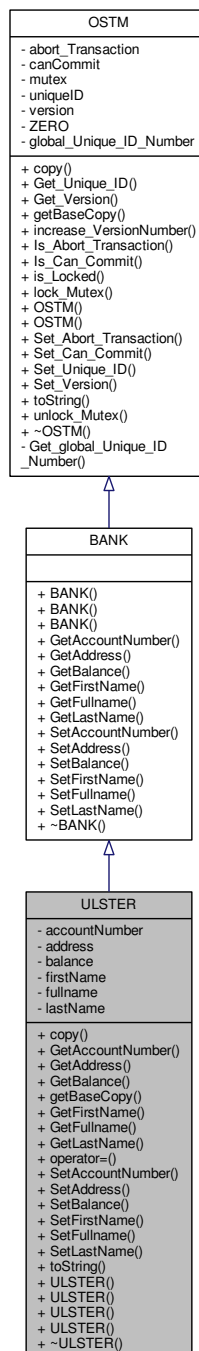
5.15 ULSTER Class Reference

```
#include <ULSTER.h>
```

Inheritance diagram for ULSTER:



Collaboration diagram for ULSTER:



Public Member Functions

- virtual void [copy](#) (std::shared_ptr< [OSTM](#) > to, std::shared_ptr< [OSTM](#) > from)
copy function, make deep copy of the object/pointer
- virtual int [GetAccountNumber](#) () const
- virtual std::string [GetAddress](#) () const
- virtual double [GetBalance](#) () const

- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object
- virtual std::string [GetFirstName](#) () const
- virtual std::string [GetFullName](#) () const
- virtual std::string [GetLastName](#) () const
- [ULSTER](#) operator= (const [ULSTER](#) &orig)
- virtual void [SetAccountNumber](#) (int [accountNumber](#))
- virtual void [SetAddress](#) (std::string [address](#))
- virtual void [SetBalance](#) (double [balance](#))
- virtual void [SetFirstName](#) (std::string [firstName](#))
- virtual void [SetFullName](#) (std::string [fullname](#))
- virtual void [SetLastName](#) (std::string [lastName](#))
- virtual void [toString](#) ()
_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type
- [ULSTER](#) ()
- [ULSTER](#) (int [accountNumber](#), double [balance](#), std::string [firstName](#), std::string [lastName](#), std::string [address](#))
- [ULSTER](#) (std::shared_ptr< [BANK](#) > obj, int _version, int _unique_id)
- [ULSTER](#) (const [ULSTER](#) &orig)
- virtual ~[ULSTER](#) ()

Private Attributes

- int [accountNumber](#)
- std::string [address](#)
- double [balance](#)
- std::string [firstName](#)
- std::string [fullname](#)
- std::string [lastName](#)

5.15.1 Detailed Description

Inherit from [BANK](#)

Definition at line 19 of file [ULSTER.h](#).

5.15.2 Constructor & Destructor Documentation

5.15.2.1 [ULSTER::ULSTER](#) () [inline]

Constructor

Definition at line 24 of file [ULSTER.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

Referenced by [getBaseCopy\(\)](#), and [ULSTER\(\)](#).

```

00024         : BANK() {
00025             this->accountNumber = 0;
00026             this->balance = 50;
00027             this->firstName = "Joe";
00028             this->lastName = "Blog";
00029             this->address = "High street, Carlow";
00030             this->fullname = firstName + " " + lastName;
00031         };

```


5.15.2.2 ULSTER::ULSTER (int *accountNumber*, double *balance*, std::string *firstName*, std::string *lastName*, std::string *address*) [inline]

Custom constructor

Definition at line 35 of file [ULSTER.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

```

00035                                     :
00036     BANK() {
00037         this->accountNumber = accountNumber;
00038         this->balance = balance;
00039         this->firstName = firstName;
00040         this->lastName = lastName;
00041         this->address = address;
00042         this->fullname = firstName + " " + lastName;
00043     };

```

5.15.2.3 ULSTER::ULSTER (std::shared_ptr< BANK > *obj*, int *_version*, int *_unique_id*) [inline]

Custom constructor, used by the library for deep copying

Definition at line 46 of file [ULSTER.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), [lastName](#), and [ULSTER\(\)](#).

```

00046                                     : BANK(_version, _unique_id) {
00047
00048         this->accountNumber = obj->GetAccountNumber();
00049         this->balance = obj->GetBalance();
00050         this->firstName = obj->GetFirstName();
00051         this->lastName = obj->GetLastName();
00052         this->address = obj->GetAddress();
00053         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054     };

```

Here is the call graph for this function:



5.15.2.4 ULSTER::ULSTER (const ULSTER & *orig*)

Copy constructor

Definition at line 15 of file [ULSTER.cpp](#).

```

00015     {
00016 }

```

5.15.2.5 ULSTER::~~ULSTER() [virtual]

de-constructor

Definition at line 18 of file [ULSTER.cpp](#).

Referenced by [operator=\(\)](#).

```
00018         {
00019     }
```

5.15.3 Member Function Documentation

5.15.3.1 void ULSTER::copy (std::shared_ptr< OSTM > *to*, std::shared_ptr< OSTM > *from*) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 37 of file [ULSTER.cpp](#).

References [OSTM::Set_Unique_ID\(\)](#).

Referenced by [operator=\(\)](#).

```
00037         {
00038
00039     std::shared_ptr<ULSTER> objTO = std::dynamic_pointer_cast<ULSTER>(to);
00040     std::shared_ptr<ULSTER> objFROM = std::dynamic_pointer_cast<ULSTER>(from);
00041     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00042     objTO->Set_Version(objFROM->Get_Version());
00043     objTO->Set_AccountNumber(objFROM->Get_AccountNumber());
00044     objTO->Set_Balance(objFROM->Get_Balance());
00045
00046
00047 }
```

Here is the call graph for this function:



5.15.3.2 int ULSTER::GetAccountNumber () const [virtual]

Reimplemented from [BANK](#).

Definition at line 83 of file [ULSTER.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00083                                     {
00084     return accountNumber;
00085 }
```

5.15.3.3 std::string ULSTER::GetAddress () const [virtual]

Reimplemented from [BANK](#).

Definition at line 67 of file [ULSTER.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00067                                     {
00068     return address;
00069 }
```

5.15.3.4 double ULSTER::GetBalance () const [virtual]

Reimplemented from [BANK](#).

Definition at line 75 of file [ULSTER.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00075                                     {
00076     return balance;
00077 }
```

5.15.3.5 std::shared_ptr< OSTM > ULSTER::getBaseCopy (std::shared_ptr< OSTM > object) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a std::shared_ptr<BANK> return type

Reimplemented from [OSTM](#).

Definition at line 25 of file [ULSTER.cpp](#).

References [ULSTER\(\)](#).

Referenced by [operator=\(\)](#).

```
00026 {
00027     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00028     std::shared_ptr<BANK> obj(new ULSTER(objTO, object->Get_Version(), object->Get_Unique_ID()));
00029     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00030     return ostm_obj;
00031 }
```

Here is the call graph for this function:



5.15.3.6 std::string ULSTER::GetFirstName () const [virtual]

Reimplemented from [BANK](#).

Definition at line 99 of file [ULSTER.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00099                                     {
00100     return firstName;
00101 }
```

5.15.3.7 std::string ULSTER::GetFullname () const [virtual]

Reimplemented from [BANK](#).

Definition at line 107 of file [ULSTER.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00107                                     {
00108     return fullname;
00109 }
```

5.15.3.8 `std::string ULSTER::GetLastName () const` [virtual]

Reimplemented from [BANK](#).

Definition at line 91 of file [ULSTER.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00091                                     {
00092     return lastName;
00093 }
```

5.15.3.9 `ULSTER ULSTER::operator= (const ULSTER & orig)` [inline]

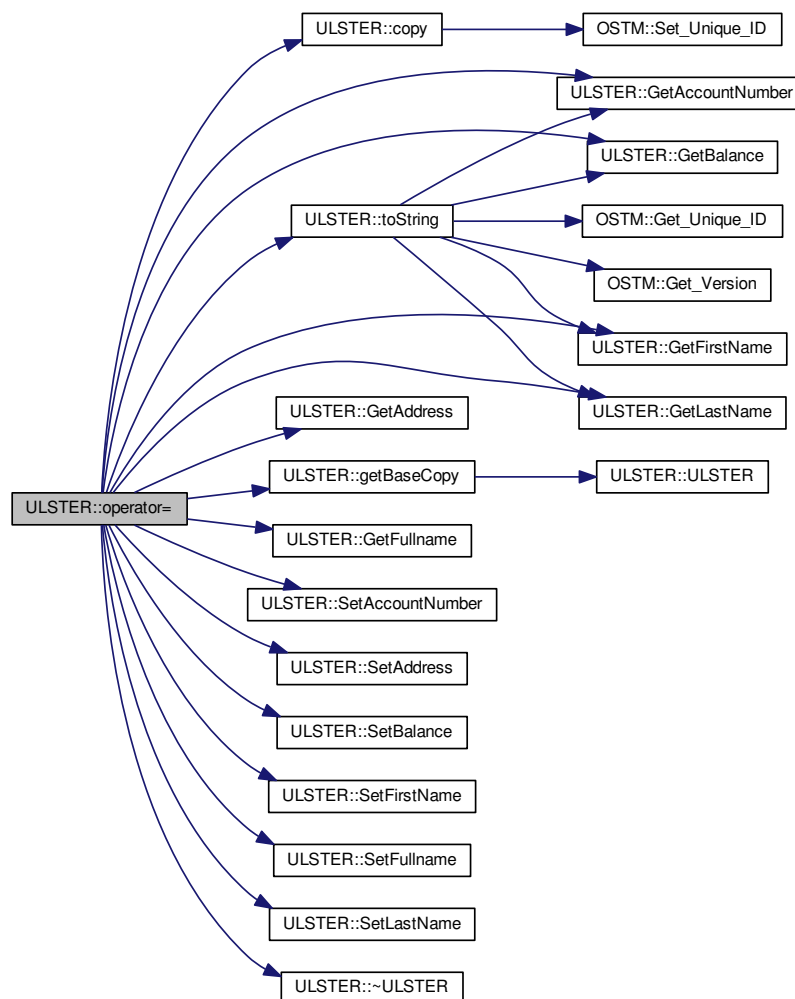
Operator

Definition at line 62 of file [ULSTER.h](#).

References [accountNumber](#), [address](#), [balance](#), [copy\(\)](#), [firstName](#), [fullname](#), [GetAccountNumber\(\)](#), [GetAddress\(\)](#), [GetBalance\(\)](#), [getBaseCopy\(\)](#), [GetFirstName\(\)](#), [GetFullname\(\)](#), [GetLastName\(\)](#), [lastName](#), [SetAccountNumber\(\)](#), [SetAddress\(\)](#), [SetBalance\(\)](#), [SetFirstName\(\)](#), [SetFullname\(\)](#), [SetLastName\(\)](#), [toString\(\)](#), and [~ULSTER\(\)](#).

```
00062 {};
```

Here is the call graph for this function:



5.15.3.10 void ULSTER::SetAccountNumber (int *accountNumber*) [virtual]

Reimplemented from [BANK](#).

Definition at line 79 of file [ULSTER.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#).

```
00079                                     {
00080     this->accountNumber = accountNumber;
00081 }
```

5.15.3.11 void ULSTER::SetAddress (std::string *address*) [virtual]

Reimplemented from [BANK](#).

Definition at line 63 of file [ULSTER.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00063                                     {
00064     this->address = address;
00065 }
```

5.15.3.12 void ULSTER::SetBalance (double *balance*) [virtual]

Reimplemented from [BANK](#).

Definition at line 71 of file [ULSTER.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#).

```
00071                                     {
00072     this->balance = balance;
00073 }
```

5.15.3.13 void ULSTER::SetFirstName (std::string *firstName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 95 of file [ULSTER.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#).

```
00095                                     {
00096     this->firstName = firstName;
00097 }
```

5.15.3.14 void ULSTER::SetFullName (std::string *fullName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 103 of file [ULSTER.cpp](#).

References [fullName](#).

Referenced by [operator=\(\)](#).

```
00103         {
00104             this->fullName = fullName;
00105     }
```

5.15.3.15 void ULSTER::SetLastName (std::string *lastName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 87 of file [ULSTER.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#).

```
00087         {
00088             this->lastName = lastName;
00089     }
```

5.15.3.16 void ULSTER::toString () [virtual]

[_cast](#), is use to cast bak the std::shared_ptr<OSTM> to the required type

toString function, displays the object values in formatted way

Reimplemented from [OSTM](#).

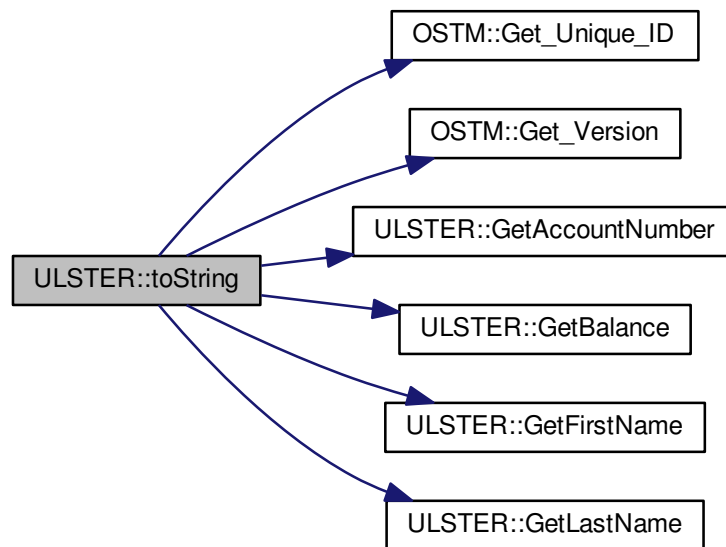
Definition at line 58 of file [ULSTER.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetAccountNumber\(\)](#), [GetBalance\(\)](#), [GetFirstName\(\)](#), and [GetLastName\(\)](#).

Referenced by [operator=\(\)](#).

```
00059 {
00060     std::cout << "\nULSTER BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account
: " << this->GetAccountNumber() << "\nDouble value : " << this->
GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00061 }
```

Here is the call graph for this function:



5.15.4 Member Data Documentation

5.15.4.1 `int ULSTER::accountNumber` [private]

Definition at line 95 of file [ULSTER.h](#).

Referenced by [GetAccountNumber\(\)](#), [operator=\(\)](#), [SetAccountNumber\(\)](#), and [ULSTER\(\)](#).

5.15.4.2 `std::string ULSTER::address` [private]

Definition at line 97 of file [ULSTER.h](#).

Referenced by [GetAddress\(\)](#), [operator=\(\)](#), [SetAddress\(\)](#), and [ULSTER\(\)](#).

5.15.4.3 `double ULSTER::balance` [private]

Definition at line 96 of file [ULSTER.h](#).

Referenced by [GetBalance\(\)](#), [operator=\(\)](#), [SetBalance\(\)](#), and [ULSTER\(\)](#).

5.15.4.4 `std::string ULSTER::firstName` [private]

Definition at line 93 of file [ULSTER.h](#).

Referenced by [GetFirstName\(\)](#), [operator=\(\)](#), [SetFirstName\(\)](#), and [ULSTER\(\)](#).

5.15.4.5 `std::string ULSTER::fullname` [private]

Definition at line 92 of file [ULSTER.h](#).

Referenced by [GetFullname\(\)](#), [operator=\(\)](#), [SetFullname\(\)](#), and [ULSTER\(\)](#).

5.15.4.6 `std::string ULSTER::lastName` [private]

Definition at line 94 of file [ULSTER.h](#).

Referenced by [GetLastName\(\)](#), [operator=\(\)](#), [SetLastName\(\)](#), and [ULSTER\(\)](#).

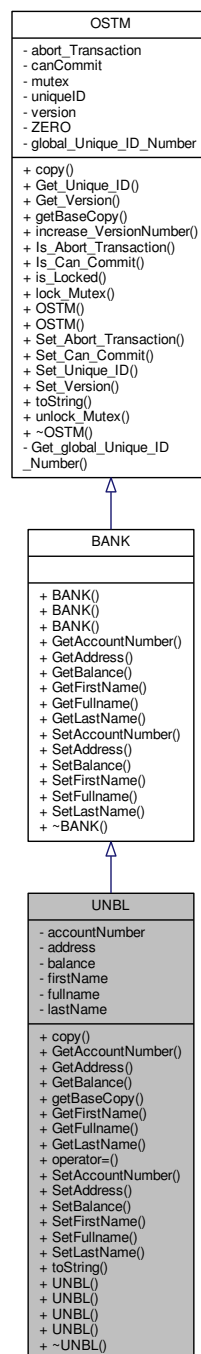
The documentation for this class was generated from the following files:

- [ULSTER.h](#)
- [ULSTER.cpp](#)

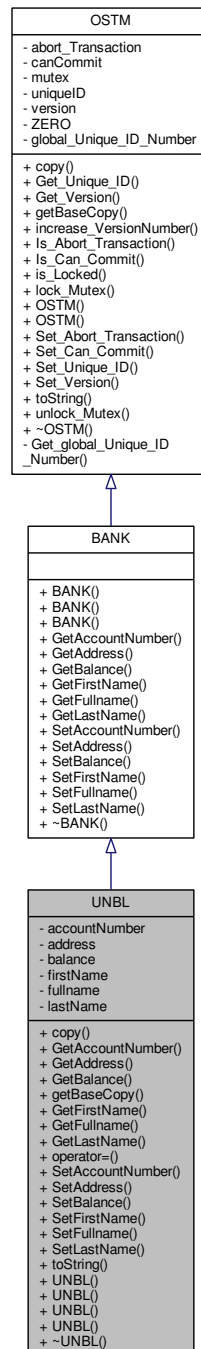
5.16 UNBL Class Reference

```
#include <UNBL.h>
```

Inheritance diagram for UNBL:



Collaboration diagram for UNBL:



Public Member Functions

- virtual void [copy](#) (std::shared_ptr< [OSTM](#) > to, std::shared_ptr< [OSTM](#) > from)
copy function, make deep copy of the object/pointer
- virtual int [GetAccountNumber](#) () const
- virtual std::string [GetAddress](#) () const
- virtual double [GetBalance](#) () const

- virtual std::shared_ptr< [OSTM](#) > [getBaseCopy](#) (std::shared_ptr< [OSTM](#) > object)
getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object
- virtual std::string [GetFirstName](#) () const
- virtual std::string [GetFullName](#) () const
- virtual std::string [GetLastName](#) () const
- [UNBL operator=](#) (const [UNBL](#) &orig)
- virtual void [SetAccountNumber](#) (int [accountNumber](#))
- virtual void [SetAddress](#) (std::string [address](#))
- virtual void [SetBalance](#) (double [balance](#))
- virtual void [SetFirstName](#) (std::string [firstName](#))
- virtual void [SetFullName](#) (std::string [fullname](#))
- virtual void [SetLastName](#) (std::string [lastName](#))
- virtual void [toString](#) ()
_cast, is use to cast bak the std::shared_ptr<OSTM> to the required type
- [UNBL](#) ()
- [UNBL](#) (int [accountNumber](#), double [balance](#), std::string [firstName](#), std::string [lastName](#), std::string [address](#))
- [UNBL](#) (std::shared_ptr< [BANK](#) > obj, int _version, int _unique_id)
- [UNBL](#) (const [UNBL](#) &orig)
- virtual [~UNBL](#) ()

Private Attributes

- int [accountNumber](#)
- std::string [address](#)
- double [balance](#)
- std::string [firstName](#)
- std::string [fullname](#)
- std::string [lastName](#)

5.16.1 Detailed Description

Inherit from [BANK](#)

Definition at line 19 of file [UNBL.h](#).

5.16.2 Constructor & Destructor Documentation

5.16.2.1 [UNBL::UNBL](#) () [inline]

Constructor

Definition at line 24 of file [UNBL.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

Referenced by [getBaseCopy\(\)](#), and [UNBL\(\)](#).

```

00024         : BANK() {
00025             this->accountNumber = 0;
00026             this->balance = 50;
00027             this->firstName = "Joe";
00028             this->lastName = "Blog";
00029             this->address = "High street, Carlow";
00030             this->fullname = firstName + " " + lastName;
00031         };

```

5.16.2.2 UNBL::UNBL (int *accountNumber*, double *balance*, std::string *firstName*, std::string *lastName*, std::string *address*)
[inline]

Custom constructor

Definition at line 35 of file [UNBL.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), and [lastName](#).

```
00035                                     :
00036     BANK() {
00037         this->accountNumber = accountNumber;
00038         this->balance = balance;
00039         this->firstName = firstName;
00040         this->lastName = lastName;
00041         this->address = address;
00042         this->fullname = firstName + " " + lastName;
00043     };
```

5.16.2.3 UNBL::UNBL (std::shared_ptr< BANK > *obj*, int *_version*, int *_unique_id*) **[inline]**

Custom constructor, used by the library for deep copying

Definition at line 46 of file [UNBL.h](#).

References [accountNumber](#), [address](#), [balance](#), [firstName](#), [fullname](#), [lastName](#), and [UNBL\(\)](#).

```
00046                                     : BANK(_version, _unique_id) {
00047
00048         this->accountNumber = obj->GetAccountNumber();
00049         this->balance = obj->GetBalance();
00050         this->firstName = obj->GetFirstName();
00051         this->lastName = obj->GetLastName();
00052         this->address = obj->GetAddress();
00053         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054     };
```

Here is the call graph for this function:



5.16.2.4 UNBL::UNBL (const UNBL & *orig*)

Copy constructor

Definition at line 11 of file [UNBL.cpp](#).

```
00011     {
00012 }
```

5.16.2.5 UNBL::~~UNBL () [virtual]

de-constructor

Definition at line 14 of file [UNBL.cpp](#).

Referenced by [operator=\(\)](#).

```
00014         {
00015     }
```

5.16.3 Member Function Documentation

5.16.3.1 void UNBL::copy (std::shared_ptr< OSTM > to, std::shared_ptr< OSTM > from) [virtual]

copy function, make deep copy of the object/pointer

Parameters

<i>objTO</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>
<i>objFROM</i>	is a std::shared_ptr<BANK> type object casted back from std::shared_ptr<OSTM>

Reimplemented from [OSTM](#).

Definition at line 33 of file [UNBL.cpp](#).

References [OSTM::Set_Unique_ID\(\)](#).

Referenced by [operator=\(\)](#).

```
00033                                     {
00034
00035     std::shared_ptr<UNBL> objTO = std::dynamic_pointer_cast<UNBL>(to);
00036     std::shared_ptr<UNBL> objFROM = std::dynamic_pointer_cast<UNBL>(from);
00037     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00038     objTO->Set_Version(objFROM->Get_Version());
00039     objTO->Set_AccountNumber(objFROM->Get_AccountNumber());
00040     objTO->Set_Balance(objFROM->Get_Balance());
00041
00042 }
```

Here is the call graph for this function:



5.16.3.2 int UNBL::GetAccountNumber () const [virtual]

Reimplemented from [BANK](#).

Definition at line 78 of file [UNBL.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00078                                     {
00079     return accountNumber;
00080 }
```

5.16.3.3 std::string UNBL::GetAddress () const [virtual]

Reimplemented from [BANK](#).

Definition at line 62 of file [UNBL.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00062                                     {
00063     return address;
00064 }
```

5.16.3.4 double UNBL::GetBalance () const [virtual]

Reimplemented from [BANK](#).

Definition at line 70 of file [UNBL.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00070                                     {
00071     return balance;
00072 }
```

5.16.3.5 std::shared_ptr< OSTM > UNBL::getBaseCopy (std::shared_ptr< OSTM > object) [virtual]

getBaseCopy function, make deep copy of the object/pointer and Return a new std::shared_ptr<BANK> type object

Parameters

<i>objTO</i>	is a BANK type pointer for casting
<i>obj</i>	is a std::shared_ptr<BANK> return type

Reimplemented from [OSTM](#).

Definition at line 21 of file [UNBL.cpp](#).

References [UNBL\(\)](#).

Referenced by [operator=\(\)](#).

```
00022 {
00023     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00024     std::shared_ptr<BANK> obj(new UNBL(objTO,object->Get_Version(),object->Get_Unique_ID()));
00025     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);

00026     return ostm_obj;
00027 }
```

Here is the call graph for this function:



5.16.3.6 `std::string UNBL::GetFirstName () const` [virtual]

Reimplemented from [BANK](#).

Definition at line 94 of file [UNBL.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00094                                     {
00095     return firstName;
00096 }
```

5.16.3.7 `std::string UNBL::GetFullname () const` [virtual]

Reimplemented from [BANK](#).

Definition at line 102 of file [UNBL.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00102                                     {
00103     return fullname;
00104 }
```


5.16.3.8 `std::string UNBL::GetLastName () const [virtual]`

Reimplemented from [BANK](#).

Definition at line 86 of file [UNBL.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#), and [toString\(\)](#).

```
00086                                     {  
00087     return lastName;  
00088 }
```

5.16.3.9 `UNBL UNBL::operator=(const UNBL & orig) [inline]`

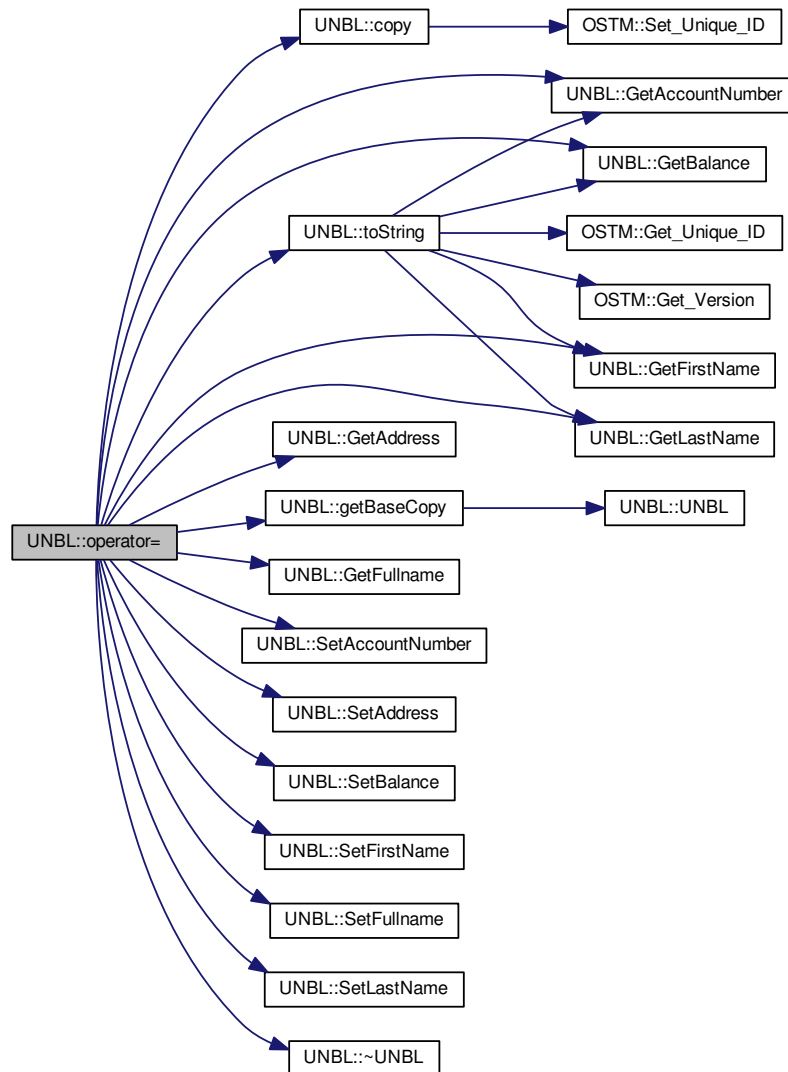
Operator

Definition at line 62 of file [UNBL.h](#).

References [accountNumber](#), [address](#), [balance](#), [copy\(\)](#), [firstName](#), [fullName](#), [GetAccountNumber\(\)](#), [GetAddress\(\)](#), [GetBalance\(\)](#), [getBaseCopy\(\)](#), [GetFirstName\(\)](#), [GetFullName\(\)](#), [GetLastName\(\)](#), [lastName](#), [SetAccountNumber\(\)](#), [SetAddress\(\)](#), [SetBalance\(\)](#), [SetFirstName\(\)](#), [SetFullName\(\)](#), [SetLastName\(\)](#), [toString\(\)](#), and [~UNBL\(\)](#).

```
00062 {};
```

Here is the call graph for this function:



5.16.3.10 void UNBL::SetAccountNumber (int *accountNumber*) [virtual]

Reimplemented from [BANK](#).

Definition at line 74 of file [UNBL.cpp](#).

References [accountNumber](#).

Referenced by [operator=\(\)](#).

```

00074      {
00075          this->accountNumber = accountNumber;
00076      }

```

5.16.3.11 void UNBL::SetAddress (std::string *address*) [virtual]

Reimplemented from [BANK](#).

Definition at line 58 of file [UNBL.cpp](#).

References [address](#).

Referenced by [operator=\(\)](#).

```
00058                                     {
00059     this->address = address;
00060 }
```

5.16.3.12 void UNBL::SetBalance (double *balance*) [virtual]

Reimplemented from [BANK](#).

Definition at line 66 of file [UNBL.cpp](#).

References [balance](#).

Referenced by [operator=\(\)](#).

```
00066                                     {
00067     this->balance = balance;
00068 }
```

5.16.3.13 void UNBL::SetFirstName (std::string *firstName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 90 of file [UNBL.cpp](#).

References [firstName](#).

Referenced by [operator=\(\)](#).

```
00090                                     {
00091     this->firstName = firstName;
00092 }
```

5.16.3.14 void UNBL::SetFullName (std::string *fullname*) [virtual]

Reimplemented from [BANK](#).

Definition at line 98 of file [UNBL.cpp](#).

References [fullname](#).

Referenced by [operator=\(\)](#).

```
00098                                     {
00099     this->fullname = fullname;
00100 }
```

5.16.3.15 void UNBL::SetLastName (std::string *lastName*) [virtual]

Reimplemented from [BANK](#).

Definition at line 82 of file [UNBL.cpp](#).

References [lastName](#).

Referenced by [operator=\(\)](#).

```
00082                                     {
00083     this->lastName = lastName;
00084 }
```

5.16.3.16 void UNBL::toString () [virtual]

`_cast`, is use to cast bak the `std::shared_ptr<OSTM>` to the required type

`toString` function, displays the object values in formatted way

Reimplemented from [OSTM](#).

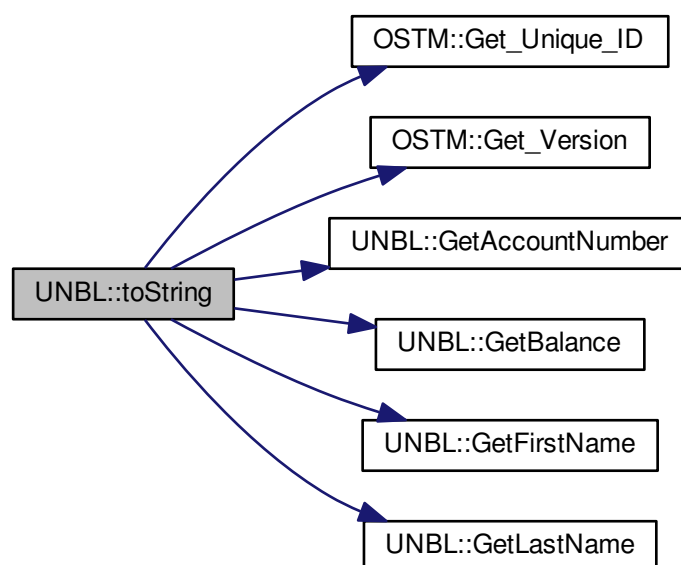
Definition at line 53 of file [UNBL.cpp](#).

References [OSTM::Get_Unique_ID\(\)](#), [OSTM::Get_Version\(\)](#), [GetAccountNumber\(\)](#), [GetBalance\(\)](#), [GetFirstName\(\)](#), and [GetLastName\(\)](#).

Referenced by [operator=\(\)](#).

```
00054 {
00055     std::cout << "\nUNBL BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account : "
00056     << this->GetAccountNumber() << "\nDouble value : " << this->
00057     GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
00058     this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00059 }
```

Here is the call graph for this function:



5.16.4 Member Data Documentation

5.16.4.1 `int UNBL::accountNumber` `[private]`

Definition at line 95 of file [UNBL.h](#).

Referenced by [GetAccountNumber\(\)](#), [operator=\(\)](#), [SetAccountNumber\(\)](#), and [UNBL\(\)](#).

5.16.4.2 `std::string UNBL::address` `[private]`

Definition at line 97 of file [UNBL.h](#).

Referenced by [GetAddress\(\)](#), [operator=\(\)](#), [SetAddress\(\)](#), and [UNBL\(\)](#).

5.16.4.3 `double UNBL::balance` `[private]`

Definition at line 96 of file [UNBL.h](#).

Referenced by [GetBalance\(\)](#), [operator=\(\)](#), [SetBalance\(\)](#), and [UNBL\(\)](#).

5.16.4.4 `std::string UNBL::firstName` `[private]`

Definition at line 93 of file [UNBL.h](#).

Referenced by [GetFirstName\(\)](#), [operator=\(\)](#), [SetFirstName\(\)](#), and [UNBL\(\)](#).

5.16.4.5 `std::string UNBL::fullName` `[private]`

Definition at line 92 of file [UNBL.h](#).

Referenced by [GetFullName\(\)](#), [operator=\(\)](#), [SetFullName\(\)](#), and [UNBL\(\)](#).

5.16.4.6 `std::string UNBL::lastName` `[private]`

Definition at line 94 of file [UNBL.h](#).

Referenced by [GetLastName\(\)](#), [operator=\(\)](#), [SetLastName\(\)](#), and [UNBL\(\)](#).

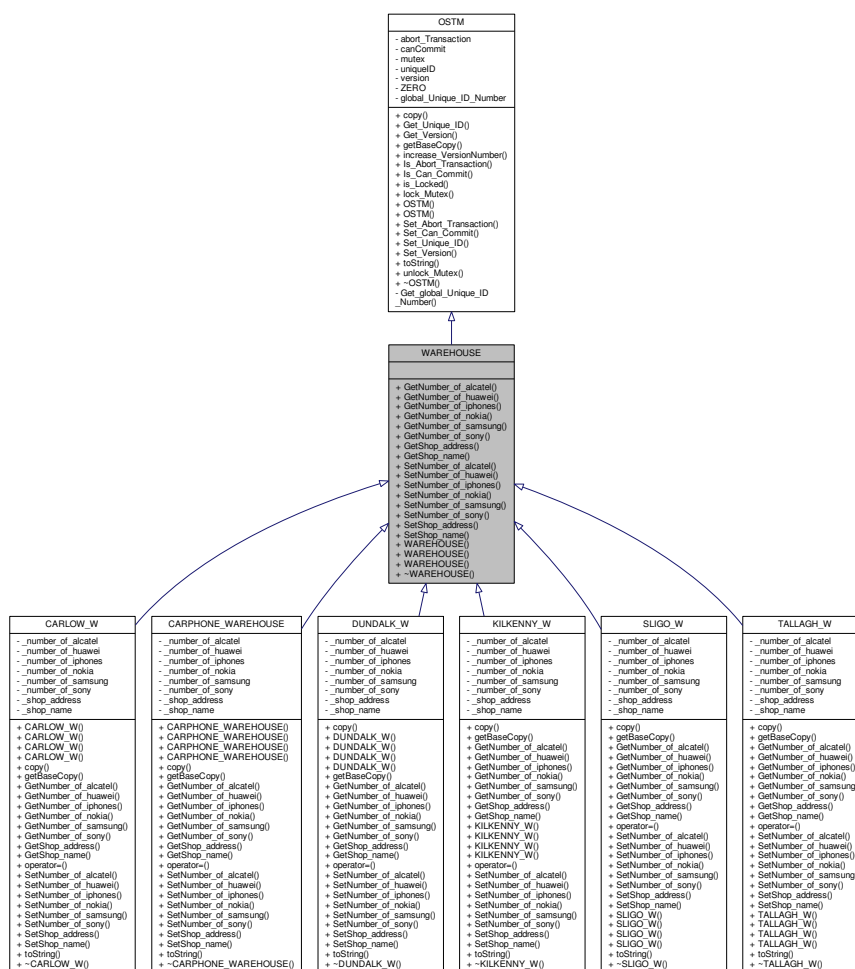
The documentation for this class was generated from the following files:

- [UNBL.h](#)
- [UNBL.cpp](#)

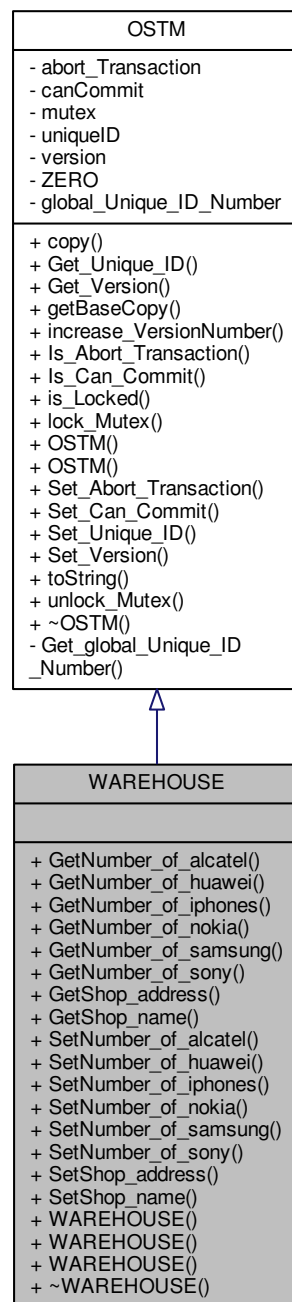
5.17 WAREHOUSE Class Reference

```
#include <WAREHOUSE.h>
```

Inheritance diagram for WAREHOUSE:



Collaboration diagram for WAREHOUSE:



Public Member Functions

- virtual int [GetNumber_of_alcatel](#) ()
- virtual int [GetNumber_of_huawei](#) ()
- virtual int [GetNumber_of_iphones](#) ()
- virtual int [GetNumber_of_nokia](#) ()
- virtual int [GetNumber_of_samsung](#) ()

- virtual int [GetNumber_of_sony](#) ()
- virtual std::string [GetShop_address](#) ()
- virtual std::string [GetShop_name](#) ()
- virtual void [SetNumber_of_alcatel](#) (int _number_of_alcatel)
- virtual void [SetNumber_of_huawei](#) (int _number_of_huawei)
- virtual void [SetNumber_of_iphones](#) (int _number_of_iphones)
- virtual void [SetNumber_of_nokia](#) (int _number_of_nokia)
- virtual void [SetNumber_of_samsung](#) (int _number_of_samsung)
- virtual void [SetNumber_of_sony](#) (int _number_of_sony)
- virtual void [SetShop_address](#) (std::string _shop_address)
- virtual void [SetShop_name](#) (std::string _shop_name)
- [WAREHOUSE](#) ()
- [WAREHOUSE](#) (int _version, int _unique_id)
- [WAREHOUSE](#) (const [WAREHOUSE](#) &orig)
- virtual [~WAREHOUSE](#) ()

5.17.1 Detailed Description

[WAREHOUSE](#) inherit from [OSTM](#) library

Definition at line 16 of file [WAREHOUSE.h](#).

5.17.2 Constructor & Destructor Documentation

5.17.2.1 [WAREHOUSE::WAREHOUSE](#) () [inline]

Constructor

Definition at line 21 of file [WAREHOUSE.h](#).

Referenced by [WAREHOUSE\(\)](#).

```
00021             :OSTM() {
00022
00023     };
```

5.17.2.2 [WAREHOUSE::WAREHOUSE](#) (int _version, int _unique_id) [inline]

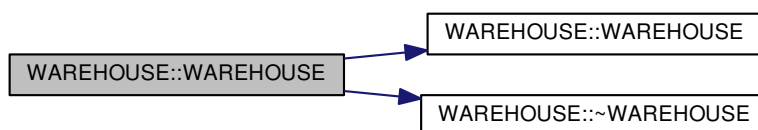
Custom Constructor

Definition at line 27 of file [WAREHOUSE.h](#).

References [WAREHOUSE\(\)](#), and [~WAREHOUSE\(\)](#).

```
00027             : OSTM(_version, _unique_id) {
00028
00029     };
```

Here is the call graph for this function:



5.17.2.3 WAREHOUSE::WAREHOUSE (const WAREHOUSE & orig)

Copy constructor

Definition at line 12 of file [WAREHOUSE.cpp](#).

```
00012                                     {
00013 }
```

5.17.2.4 WAREHOUSE::~WAREHOUSE () [virtual]

de-constructor

Definition at line 15 of file [WAREHOUSE.cpp](#).

Referenced by [WAREHOUSE\(\)](#).

```
00015                                     {
00016 }
```

5.17.3 Member Function Documentation

5.17.3.1 virtual int WAREHOUSE::GetNumber_of_alcatel () [inline],[virtual]

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 44 of file [WAREHOUSE.h](#).

```
00044 {};
```

5.17.3.2 virtual int WAREHOUSE::GetNumber_of_huawei () [inline],[virtual]

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 48 of file [WAREHOUSE.h](#).

```
00048 {};
```

5.17.3.3 virtual int WAREHOUSE::GetNumber_of_iphones () [inline],[virtual]

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 54 of file [WAREHOUSE.h](#).

```
00054 {};
```

5.17.3.4 `virtual int WAREHOUSE::GetNumber_of_nokia () [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 46 of file [WAREHOUSE.h](#).

```
00046 {};
```

5.17.3.5 `virtual int WAREHOUSE::GetNumber_of_samsung () [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 52 of file [WAREHOUSE.h](#).

```
00052 {};
```

5.17.3.6 `virtual int WAREHOUSE::GetNumber_of_sony () [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 50 of file [WAREHOUSE.h](#).

```
00050 {};
```

5.17.3.7 `virtual std::string WAREHOUSE::GetShop_address () [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 58 of file [WAREHOUSE.h](#).

```
00058 {};
```

5.17.3.8 `virtual std::string WAREHOUSE::GetShop_name () [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 56 of file [WAREHOUSE.h](#).

```
00056 {};
```

5.17.3.9 `virtual void WAREHOUSE::SetNumber_of_alcatel (int _number_of_alcatel) [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 43 of file [WAREHOUSE.h](#).

```
00043 {};
```

5.17.3.10 `virtual void WAREHOUSE::SetNumber_of_huawei (int _number_of_huawei) [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 47 of file [WAREHOUSE.h](#).

```
00047 {};
```

5.17.3.11 `virtual void WAREHOUSE::SetNumber_of_iphones (int _number_of_iphones) [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 53 of file [WAREHOUSE.h](#).

```
00053 {};
```

5.17.3.12 `virtual void WAREHOUSE::SetNumber_of_nokia (int _number_of_nokia) [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 45 of file [WAREHOUSE.h](#).

Referenced by [_complex_warehouse_transfer_\(\)](#), [_nested_warehouse_transfer_\(\)](#), and [_warehouse_transfer_\(\)](#).

```
00045 {};
```

5.17.3.13 `virtual void WAREHOUSE::SetNumber_of_samsung (int _number_of_samsung) [inline],[virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 51 of file [WAREHOUSE.h](#).

```
00051 {};
```

5.17.3.14 `virtual void WAREHOUSE::SetNumber_of_sony (int number_of_sony) [inline], [virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 49 of file [WAREHOUSE.h](#).

```
00049 {};
```

5.17.3.15 `virtual void WAREHOUSE::SetShop_address (std::string shop_address) [inline], [virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 57 of file [WAREHOUSE.h](#).

```
00057 {};
```

5.17.3.16 `virtual void WAREHOUSE::SetShop_name (std::string shop_name) [inline], [virtual]`

Reimplemented in [CARPHONE_WAREHOUSE](#), [SLIGO_W](#), [TALLAGH_W](#), [CARLOW_W](#), [DUNDALK_W](#), and [KILKENNY_W](#).

Definition at line 55 of file [WAREHOUSE.h](#).

```
00055 {};
```

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

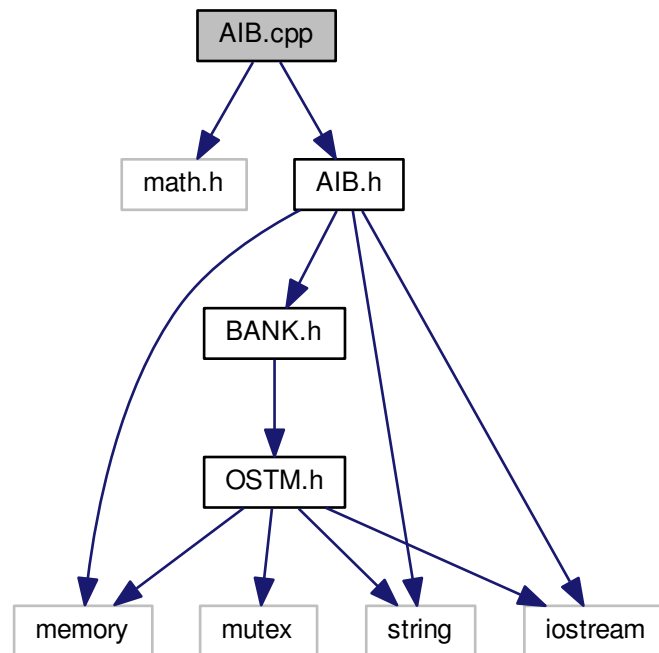
- [WAREHOUSE.h](#)
- [WAREHOUSE.cpp](#)

6 File Documentation

6.1 AIB.cpp File Reference

```
#include <math.h>
#include "AIB.h"
```

Include dependency graph for AIB.cpp:



6.2 AIB.cpp

```

00001 /*
00002  * File:   AIB.cpp
00003  * Author: Zoltan Fuzesi
00004  * IT Carlow : C00197361
00005  *
00006  * Created on January 17, 2018, 8:02 PM
00007  */
00008
00009 #include <math.h>
00010
00011 #include "AIB.h"
00012
00013
00014 AIB::AIB(const AIB& orig) {
00015 }
00016
00017 AIB::~AIB() {
00018 }
00019
00024 std::shared_ptr<OSTM> AIB::getBaseCopy(std::shared_ptr<OSTM> object)
00025 {
00026
00027     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00028     std::shared_ptr<BANK> obj(new AIB(objTO, object->Get_Version(), object->Get_Unique_ID()));
00029     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00030     return ostm_obj;
00031 }
00032
00037 void AIB::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00038
00039     std::shared_ptr<AIB> objTO = std::dynamic_pointer_cast<AIB>(to);
00040     std::shared_ptr<AIB> objFROM = std::dynamic_pointer_cast<AIB>(from);
00041     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00042     objTO->Set_Version(objFROM->Get_Version());
00043     objTO->SetAccountNumber(objFROM->GetAccountNumber());
00044     objTO->SetBalance(objFROM->GetBalance());
00045 }

```

```

00049 //std::shared_ptr<AIB> AIB::_cast(std::shared_ptr<OSTM> _object){
00050 //
00051 //     return std::static_pointer_cast<AIB>(_object);
00052 //}
00056 void AIB::toString()
00057 {
00058     std::cout << "\nAIB BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account :
        " << this->GetAccountNumber() << "\nDouble value : " << this->
        GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
        this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00059 }
00060
00061 void AIB::SetAddress(std::string address) {
00062     this->address = address;
00063 }
00064
00065 std::string AIB::GetAddress() const {
00066     return address;
00067 }
00068
00069 void AIB::SetBalance(double balance) {
00070     this->balance = balance;
00071 }
00072
00073 double AIB::GetBalance() const {
00074     return balance;
00075 }
00076
00077 void AIB::SetAccountNumber(int accountNumber) {
00078     this->accountNumber = accountNumber;
00079 }
00080
00081 int AIB::GetAccountNumber() const {
00082     return accountNumber;
00083 }
00084
00085 void AIB::SetLastName(std::string lastName) {
00086     this->lastName = lastName;
00087 }
00088
00089 std::string AIB::GetLastName() const {
00090     return lastName;
00091 }
00092
00093 void AIB::SetFirstName(std::string firstName) {
00094     this->firstName = firstName;
00095 }
00096
00097 std::string AIB::GetFirstName() const {
00098     return firstName;
00099 }
00100
00101 void AIB::SetFullname(std::string fullname) {
00102     this->fullname = fullname;
00103 }
00104
00105 std::string AIB::GetFullname() const {
00106     return fullname;
00107 }

```

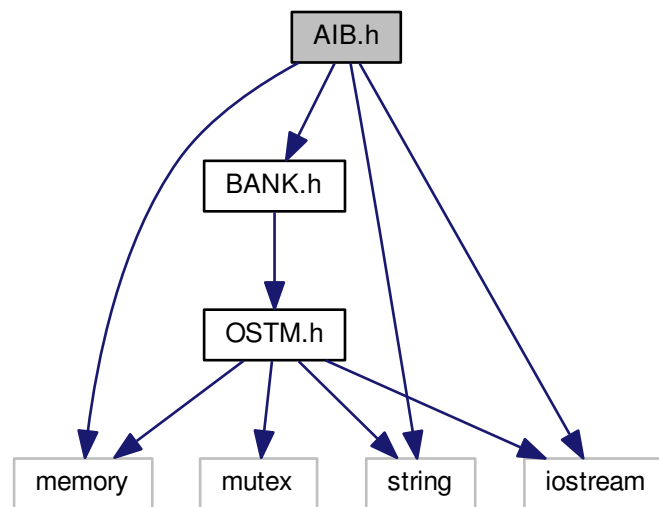
6.3 AIB.h File Reference

```

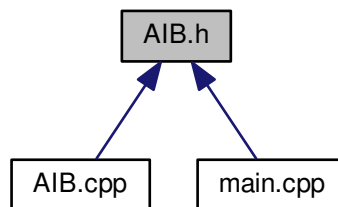
#include "BANK.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for AIB.h:



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



Classes

- class [AIB](#)

6.4 AIB.h

```

00001 /*
00002  * File:   AIB.h
00003  * Author: Zoltan Fuzesi
00004  * IT Carlow : C00197361
00005  *
00006  * Created on January 17, 2018, 8:02 PM
00007  */
  
```

```

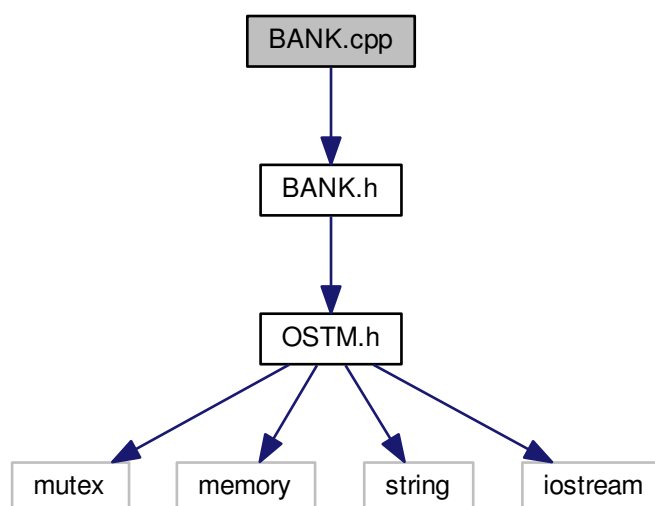
00008
00009 #ifndef AIB_H
00010 #define AIB_H
00011 #include "BANK.h"
00012 #include <string>
00013 #include <memory>
00014 #include <iostream>
00018 class AIB : public BANK {
00019 public:
00023     AIB(): BANK()
00024     {
00025         this->accountNumber = 0;
00026         this->balance = 50;
00027         this->firstName = "Joe";
00028         this->lastName = "Blog";
00029         this->address = "High street, Carlow";
00030         this->fullname = firstName + " " + lastName;
00031
00032     };
00036     AIB(int accountNumber, double balance, std::string
firstName, std::string lastName, std::string address):
BANK()
00037     {
00038         this->accountNumber = accountNumber;
00039         this->balance = balance;
00040         this->firstName = firstName;
00041         this->lastName = lastName;
00042         this->address = address;
00043         this->fullname = firstName + " " + lastName;
00044     };
00048     AIB(std::shared_ptr<BANK> obj, int _version, int _unique_id): BANK(_version, _unique_id)
00049     {
00050
00051         this->accountNumber = obj->GetAccountNumber();
00052         this->balance = obj->GetBalance();
00053         this->firstName = obj->GetFirstName();
00054         this->lastName = obj->GetLastName();
00055         this->address = obj->GetAddress();
00056         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00057
00058     };
00062     AIB(const AIB& orig);
00066     AIB operator=(const AIB& orig){};
00070     virtual ~AIB();
00071
00072     /*
00073     * Implement OSTM virtual methods
00074     */
00075     // virtual std::shared_ptr<AIB> _cast(std::shared_ptr<OSTM> _object);
00076     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00077     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00078     virtual void toString();
00079
00080     /*
00081     * Implement BANK virtual methods
00082     */
00083     virtual void SetAddress(std::string address);
00084     virtual std::string GetAddress() const;
00085     virtual void SetBalance(double balance);
00086     virtual double GetBalance() const;
00087     virtual void SetAccountNumber(int accountNumber);
00088     virtual int GetAccountNumber() const;
00089     virtual void SetLastName(std::string lastName);
00090     virtual std::string GetLastName() const;
00091     virtual void SetFirstName(std::string firstName);
00092     virtual std::string GetFirstName() const;
00093     virtual void SetFullname(std::string fullname);
00094     virtual std::string GetFullname() const;
00095
00096 private:
00097     std::string fullname;
00098     std::string firstName;
00099     std::string lastName;
00100     int accountNumber;
00101     double balance;
00102     std::string address;
00103
00104 };
00105
00106
00107 #endif /* AIB_H */

```


6.5 BANK.cpp File Reference

```
#include "BANK.h"
```

Include dependency graph for BANK.cpp:



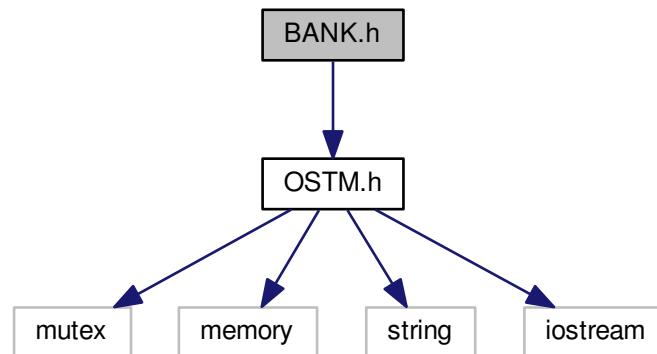
6.6 BANK.cpp

```
00001 /*
00002  * File:   BANK.cpp
00003  * Author: Zoltan Fuzesi
00004  * IT Carlow : C00197361
00005  *
00006  * Created on January 17, 2018, 8:02 PM
00007  */
00008
00009 #include "BANK.h"
00010
00011 BANK::BANK(const BANK& orig) {
00012 }
00013
00014 BANK::~BANK() {
00015 }
00016
```

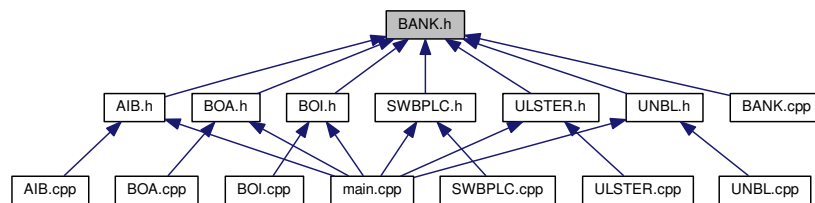
6.7 BANK.h File Reference

```
#include "OSTM.h"
```

Include dependency graph for BANK.h:



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



Classes

- class [BANK](#)

6.8 BANK.h

```

00001 /*
00002  * File:   BANK.h
00003  * Author: Zoltan Fuzesi
00004  * IT Carlow : C00197361
00005  *
00006  * Created on January 17, 2018, 8:02 PM
00007  */
00008
00009 #ifndef BANK_H
00010 #define BANK_H
00011 #include "OSTM.h"
00016 class BANK : public OSTM {
00017
00018
00019 public:
00023     BANK(): OSTM() {
00024
00025     };
00029     BANK(int _version, int _unique_id) : OSTM(_version, _unique_id){
  
```

```

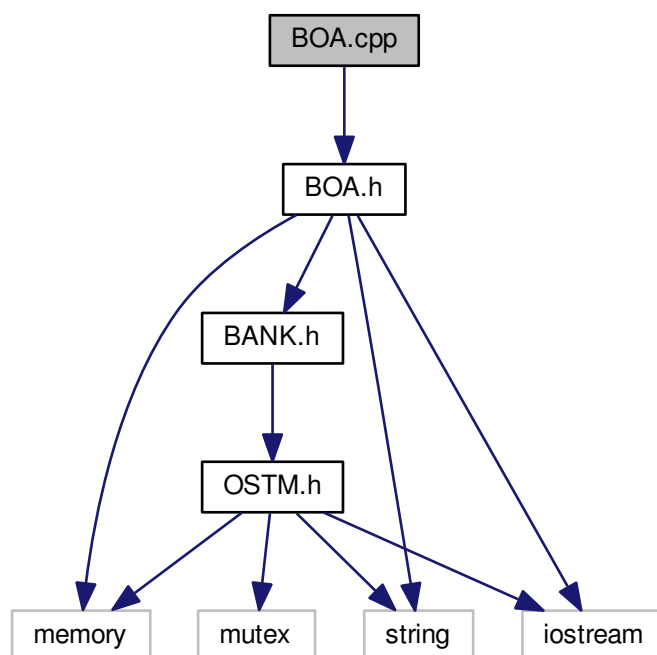
00030
00031     };
00035     BANK(const BANK& orig);
00039     virtual ~BANK();
00040
00041     /*
00042      * Bank specific virtual functions
00043      */
00044     virtual void SetAddress(std::string address){};
00045     virtual std::string GetAddress() const{};
00046     virtual void SetBalance(double balance){};
00047     virtual double GetBalance() const{};
00048     virtual void SetAccountNumber(int accountNumber){};
00049     virtual int GetAccountNumber() const{};
00050     virtual void SetLastName(std::string lastName){};
00051     virtual std::string GetLastName() const{};
00052     virtual void SetFirstName(std::string firstName){};
00053     virtual std::string GetFirstName() const{};
00054     virtual void SetFullname(std::string fullname){};
00055     virtual std::string GetFullname() const{};
00056
00057 private:
00058
00059 };
00060
00061 #endif /* BANK_H */
00062

```

6.9 BOA.cpp File Reference

```
#include "BOA.h"
```

Include dependency graph for BOA.cpp:



6.10 BOA.cpp

```
00001 /*
```

```

00002  * File:    BOA.cpp
00003  * Author:  Zoltan Fuzesi
00004  * IT Carlow : C00197361
00005  *
00006  * Created on January 17, 2018, 8:02 PM
00007  */
00008
00009 #include "BOA.h"
00010
00011
00012 BOA::BOA(const BOA& orig) {
00013 }
00014
00015 BOA::~BOA() {
00016 }
00022 std::shared_ptr<OSTM> BOA::getBaseCopy(std::shared_ptr<OSTM> object)
00023 {
00024     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00025     std::shared_ptr<BANK> obj(new BOA(objTO, object->Get_Version(), object->Get_Unique_ID()));
00026     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00027     return ostm_obj;
00028 }
00034 void BOA::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00035
00036     std::shared_ptr<BOA> objTO = std::dynamic_pointer_cast<BOA>(to);
00037     std::shared_ptr<BOA> objFROM = std::dynamic_pointer_cast<BOA>(from);
00038     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00039     objTO->Set_Version(objFROM->Get_Version());
00040     objTO->SetAccountNumber(objFROM->GetAccountNumber());
00041     objTO->SetBalance(objFROM->GetBalance());
00042 }
00043
00047 //std::shared_ptr<BOA> BOA::_cast(std::shared_ptr<OSTM> _object){
00048 //
00049 //     return std::static_pointer_cast<BOA>(_object);
00050 //}
00054 void BOA::toString()
00055 {
00056     // std::cout << "\nUnique ID : " << this->GetUniqueID() << "\nInt value : " << this->GetV_int() <<
00057     // "\nDouble value : " << this->GetV_double() << "\nFloat value : " << this->GetV_float() << "\nString value : " <<
00058     // this->GetV_string() << "\nVersion number : " << this->GetVersion() << "\nLoad Counter : " <<
00059     // this->GetLoadCounter() << "\nWrite Counter : " << this->GetWriteCounter() << std::endl;
00060     std::cout << "\nBOA BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account
00061     : " << this->GetAccountNumber() << "\nDouble value : " << this->
00062     GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
00063     this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00064 }
00065
00066 void BOA::SetAddress(std::string address) {
00067     this->address = address;
00068 }
00069
00070 std::string BOA::GetAddress() const {
00071     return address;
00072 }
00073
00074 void BOA::SetBalance(double balance) {
00075     this->balance = balance;
00076 }
00077
00078 double BOA::GetBalance() const {
00079     return balance;
00080 }
00081
00082 void BOA::SetAccountNumber(int accountNumber) {
00083     this->accountNumber = accountNumber;
00084 }
00085
00086 int BOA::GetAccountNumber() const {
00087     return accountNumber;
00088 }
00089
00090 void BOA::SetLastName(std::string lastName) {
00091     this->lastName = lastName;
00092 }
00093
00094 std::string BOA::GetLastName() const {
00095     return lastName;
00096 }
00097
00098 void BOA::SetFirstName(std::string firstName) {
00099     this->firstName = firstName;
00100 }
00101
00102 std::string BOA::GetFirstName() const {
00103     return firstName;
00104 }

```

```

00099
00100 void BOA::SetFullname(std::string fullname) {
00101     this->fullname = fullname;
00102 }
00103
00104 std::string BOA::GetFullname() const {
00105     return fullname;
00106 }
00107

```

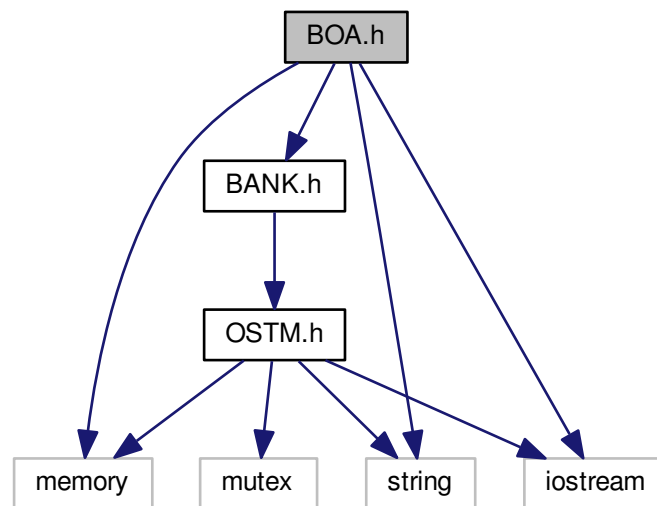
6.11 BOA.h File Reference

```

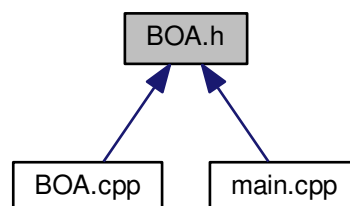
#include "BANK.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for BOA.h:



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



Classes

- class BOA

6.12 BOA.h

```

00001 /*
00002  * File:    BOA.h
00003  * Author:  Zoltan Fuzesi
00004  * IT Carlow : C00197361
00005  *
00006  * Created on January 17, 2018, 8:02 PM
00007  */
00008
00009 #ifndef BOA_H
00010 #define BOA_H
00011 #include "BANK.h"
00012 #include <string>
00013 #include <memory>
00014 #include <iostream>
00018 class BOA : public BANK {
00019 public:
00020
00024     BOA() : BANK() {
00025         this->accountNumber = 0;
00026         this->balance = 50;
00027         this->firstName = "Joe";
00028         this->lastName = "Blog";
00029         this->address = "High street, Carlow";
00030         this->fullname = firstName + " " + lastName;
00031     };
00035     BOA(int accountNumber, double balance, std::string
firstName, std::string lastName, std::string address) :
BANK() {
00036         this->accountNumber = accountNumber;
00037         this->balance = balance;
00038         this->firstName = firstName;
00039         this->lastName = lastName;
00040         this->address = address;
00041         this->fullname = firstName + " " + lastName;
00042     };
00046     BOA(std::shared_ptr<BANK> obj, int _version, int _unique_id) : BANK(_version, _unique_id) {
00047
00048         this->accountNumber = obj->GetAccountNumber();
00049         this->balance = obj->GetBalance();
00050         this->firstName = obj->GetFirstName();
00051         this->lastName = obj->GetLastName();
00052         this->address = obj->GetAddress();
00053         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054     };
00055
00056     BOA(const BOA& orig);
00064     BOA operator=(const BOA& orig) {
00065     };
00069     virtual ~BOA();
00070
00071     /*
00072     * Implement OSTM virtual methods
00073     */
00074     //virtual std::shared_ptr<BOA> _cast(std::shared_ptr<OSTM> _object);
00075     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00076     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00077     virtual void toString();
00078
00079     /*
00080     * Implement BANK virtual methods
00081     */
00082     virtual void SetAddress(std::string address);
00083     virtual std::string GetAddress() const;
00084     virtual void SetBalance(double balance);
00085     virtual double GetBalance() const;
00086     virtual void SetAccountNumber(int accountNumber);
00087     virtual int GetAccountNumber() const;
00088     virtual void SetLastName(std::string lastName);
00089     virtual std::string GetLastName() const;
00090     virtual void SetFirstName(std::string firstName);
00091     virtual std::string GetFirstName() const;
00092     virtual void SetFullname(std::string fullname);
00093     virtual std::string GetFullname() const;
00094 private:
00095     std::string fullname;

```

```

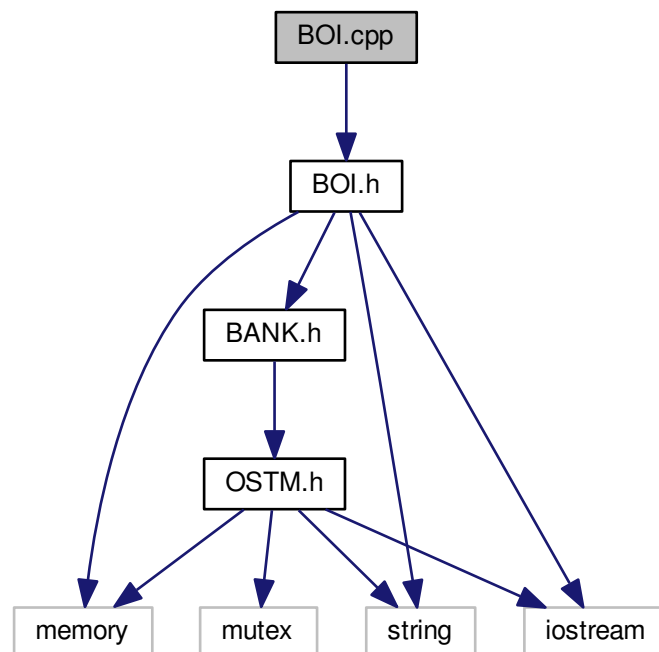
00096     std::string firstName;
00097     std::string lastName;
00098     int accountNumber;
00099     double balance;
00100     std::string address;
00101
00102 };
00103
00104 #endif /* BOA_H */
00105

```

6.13 BOI.cpp File Reference

```
#include "BOI.h"
```

Include dependency graph for BOI.cpp:



6.14 BOI.cpp

```

00001
00002 /*
00003  * File:   BOI.cpp
00004  * Author: Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #include "BOI.h"
00011
00012 BOI::~BOI() {
00013 }
00014
00015 BOI::BOI(const BOI& orig) {
00016 }

```

```

00022 std::shared_ptr<OSTM> BOI::getBaseCopy(std::shared_ptr<OSTM> object)
00023 {
00024
00025     std::shared_ptr<BOI> objTO = std::dynamic_pointer_cast<BOI>(object);
00026     std::shared_ptr<BOI> obj(new BOI(objTO,object->Get_Version(),object->Get_Unique_ID()));
00027     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00028     return ostm_obj;
00029 }
00035 void BOI::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00036
00037     std::shared_ptr<BOI> objTO = std::dynamic_pointer_cast<BOI>(to);
00038     std::shared_ptr<BOI> objFROM = std::dynamic_pointer_cast<BOI>(from);
00039     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00040     objTO->Set_Version(objFROM->Get_Version());
00041     objTO->SetAccountNumber(objFROM->GetAccountNumber());
00042     objTO->SetBalance(objFROM->GetBalance());
00043 }
00047 //std::shared_ptr<BOI> BOI::_cast(std::shared_ptr<OSTM> _object){
00048 //
00049 //     return std::static_pointer_cast<BOI>(_object);
00050 //}
00054 void BOI::toString()
00055 {
00056     std::cout << "\nBOI BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account :
" << this->GetAccountNumber() << "\nDouble value : " << this->
GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00057 }
00058 void BOI::SetAddress(std::string address) {
00059     this->address = address;
00060 }
00061
00062 std::string BOI::GetAddress() const {
00063     return address;
00064 }
00065
00066 void BOI::SetBalance(double balance) {
00067     this->balance = balance;
00068 }
00069
00070 double BOI::GetBalance() const {
00071     return balance;
00072 }
00073
00074 void BOI::SetAccountNumber(int accountNumber) {
00075     this->accountNumber = accountNumber;
00076 }
00077
00078 int BOI::GetAccountNumber() const {
00079     return accountNumber;
00080 }
00081
00082 void BOI::SetLastName(std::string lastName) {
00083     this->lastName = lastName;
00084 }
00085
00086 std::string BOI::GetLastName() const {
00087     return lastName;
00088 }
00089
00090 void BOI::SetFirstName(std::string firstName) {
00091     this->firstName = firstName;
00092 }
00093
00094 std::string BOI::GetFirstName() const {
00095     return firstName;
00096 }
00097
00098 void BOI::SetFullname(std::string fullname) {
00099     this->fullname = fullname;
00100 }
00101
00102 std::string BOI::GetFullname() const {
00103     return fullname;
00104 }
00105

```

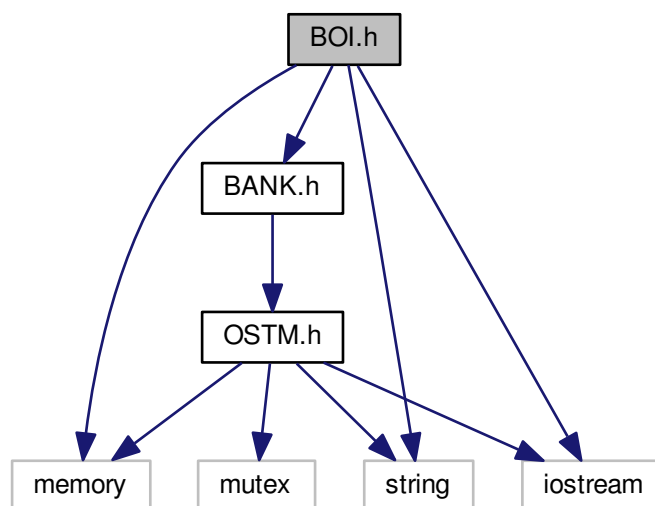
6.15 BOI.h File Reference

```
#include "BANK.h"
```

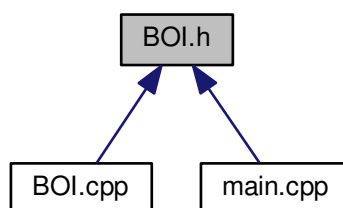


```
#include <string>
#include <memory>
#include <iostream>
```

Include dependency graph for BOI.h:



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



Classes

- class [BOI](#)

6.16 BOI.h

```
00001
00002 /*
00003  * File:   BOI.h
```

```

00004  * Author: Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef BOI_H
00011 #define BOI_H
00012 #include "BANK.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class BOI: public BANK {
00020 public:
00024     BOI(): BANK()
00025     {
00026         this->accountNumber = 0;
00027         this->balance = 50;
00028         this->firstName = "Joe";
00029         this->lastName = "Blog";
00030         this->address = "High street, Carlow";
00031         this->fullname = firstName + " " + lastName;
00032     }
00033
00037     BOI(int accountNumber, double balance, std::string
firstName, std::string lastName, std::string address):
BANK()
00038     {
00039         this->accountNumber = accountNumber;
00040         this->balance = balance;
00041         this->firstName = firstName;
00042         this->lastName = lastName;
00043         this->address = address;
00044         this->fullname = firstName + " " + lastName;
00045     };
00049     BOI(std::shared_ptr<BOI> obj, int _version, int _unique_id): BANK(_version, _unique_id)
00050     {
00051         this->accountNumber = obj->GetAccountNumber();
00052         this->balance = obj->GetBalance();
00053         this->firstName = obj->GetFirstName();
00054         this->lastName = obj->GetLastName();
00055         this->address = obj->GetAddress();
00056         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00057     };
00061     BOI(const BOI& orig);
00065     BOI operator=(const BOI& orig){};
00069     virtual ~BOI();
00070
00071     /*
00072     * Implement OSTM virtual methods
00073     */
00074     // virtual std::shared_ptr<BOI> _cast(std::shared_ptr<OSTM> _object);
00075     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00076     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00077     virtual void toString();
00078
00079     /*
00080     * Implement BANK virtual methods
00081     */
00082     virtual void SetAddress(std::string address);
00083     virtual std::string GetAddress() const;
00084     virtual void SetBalance(double balance);
00085     virtual double GetBalance() const;
00086     virtual void SetAccountNumber(int accountNumber);
00087     virtual int GetAccountNumber() const;
00088     virtual void SetLastName(std::string lastName);
00089     virtual std::string GetLastName() const;
00090     virtual void SetFirstName(std::string firstName);
00091     virtual std::string GetFirstName() const;
00092     virtual void SetFullname(std::string fullname);
00093     virtual std::string GetFullname() const;
00094
00095 private:
00096     std::string fullname;
00097     std::string firstName;
00098     std::string lastName;
00099     int accountNumber;
00100     double balance;
00101     std::string address;
00102
00103 };
00104
00105 #endif /* BOI_H */
00106
00107
00108     //virtual std::string get_class();
00109

```

```

00110 //
00111 //virtual bool get(std::shared_ptr<OSTM> object);
00112
00114 // * To change this license header, choose License Headers in Project Properties.
00115 // * To change this template file, choose Tools | Templates
00116 // * and open the template in the editor.
00117 // */
00118 //
00120 // * File: BOI.h
00121 // * Author: zoltan
00122 // *
00123 // * Created on January 19, 2018, 6:37 PM
00124 // */
00125 //
00126 //ifndef BOI_H
00127 //define BOI_H
00128 //include "OSTM.h"
00129 //include <string>
00130 //include <memory>
00131 //include <iostream>
00132 //
00133 //class BOI: public OSTM {
00134 //public:
00135 //    BOI(): OSTM()
00136 //    {
00137 //        this->accountNumber = 0;
00138 //        this->balance = 50;
00139 //        this->firstName = "Joe";
00140 //        this->lastName = "Blog";
00141 //        this->address = "High street, Carlow";
00142 //        this->fullname = firstName + " " + lastName;
00143 //    }
00144 //
00145 //    BOI(int accountNumber, double balance, std::string firstName, std::string lastName, std::string
address): OSTM()
00146 //    {
00147 //        this->accountNumber = accountNumber;
00148 //        this->balance = balance;
00149 //        this->firstName = firstName;
00150 //        this->lastName = lastName;
00151 //        this->address = address;
00152 //        this->fullname = firstName + " " + lastName;
00153 //    };
00154 //
00155 //    BOI(OSTM& obj, int _version, int _unique_id): OSTM(_version, _unique_id)
00156 //    {
00157 //        this->accountNumber = obj.GetAccountNumber();
00158 //        this->balance = obj.GetBalance();
00159 //        this->firstName = obj.GetFirstName();
00160 //        this->lastName = obj.GetLastName();
00161 //        this->address = obj.GetAddress();
00162 //        this->fullname = obj.GetFirstName() + " " + obj.GetLastName();
00163 //    };
00164 //
00165 //    BOI(std::shared_ptr<OSTM> obj, int _version, int _unique_id): OSTM(_version, _unique_id)
00166 //    {
00167 //        this->accountNumber = obj->GetAccountNumber();
00168 //        this->balance = obj->GetBalance();
00169 //        this->firstName = obj->GetFirstName();
00170 //        this->lastName = obj->GetLastName();
00171 //        this->address = obj->GetAddress();
00172 //        this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00173 //    };
00174 //
00175 //    BOI(const BOI& orig);
00176 //    BOI operator=(const BOI& orig){};
00177 //    virtual ~BOI();
00178 //
00179 //    virtual std::shared_ptr<BOI> _cast(std::shared_ptr<OSTM> _object);
00180 //    virtual std::shared_ptr<BOI> getBaseCopy(OSTM& object);
00181 //    virtual std::shared_ptr<BOI> getBaseCopy(std::shared_ptr<OSTM> object);
00182 //    virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00183 //    virtual void toString();
00184 //    virtual void SetAddress(std::string address);
00185 //    virtual std::string GetAddress() const;
00186 //    virtual void SetBalance(double balance);
00187 //    virtual double GetBalance() const;
00188 //    virtual void SetAccountNumber(int accountNumber);
00189 //    virtual int GetAccountNumber() const;
00190 //    virtual void SetLastName(std::string lastName);
00191 //    virtual std::string GetLastName() const;
00192 //    virtual void SetFirstName(std::string firstName);
00193 //    virtual std::string GetFirstName() const;
00194 //    virtual void SetFullname(std::string fullname);
00195 //    virtual std::string GetFullname() const;
00196 //

```

```

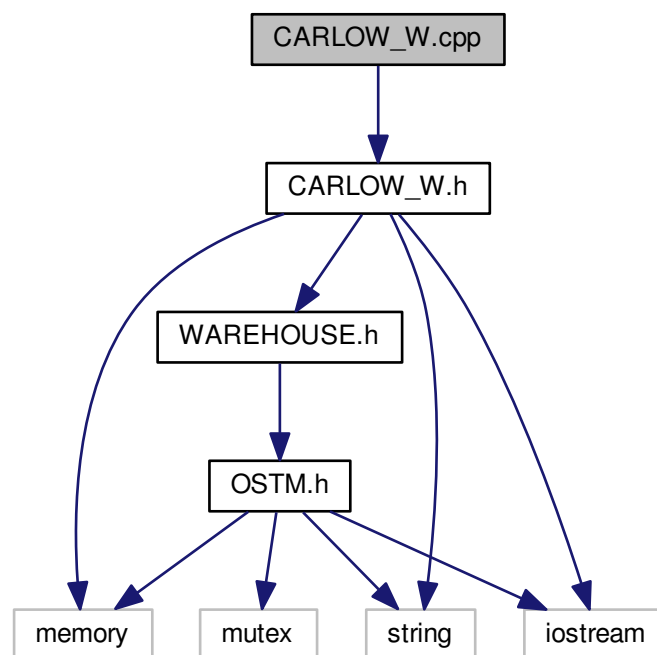
00210 //private:
00211 //     std::string fullname;
00212 //     std::string firstName;
00213 //     std::string lastName;
00214 //     int accountNumber;
00215 //     double balance;
00216 //     std::string address;
00217 //
00218 //};
00219 //
00220 //endif /* BOI_H */
00221 //
00222 //
00223 //     //virtual std::string get_class();
00224 //
00225 //     //
00226 //     //virtual bool get(std::shared_ptr<OSTM> object);
00227 //

```

6.17 CARLOW_W.cpp File Reference

```
#include "CARLOW_W.h"
```

Include dependency graph for CARLOW_W.cpp:



6.18 CARLOW_W.cpp

```

00001
00002 /*
00003  * File:    CARLOW_W.cpp
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */

```

```

00009 #include "CARLOW_W.h"
00010
00011
00012 //CARLOW_W::CARLOW_W() {
00013 //}
00014 CARLOW_W::~CARLOW_W() {
00015 }
00016
00017 CARLOW_W::CARLOW_W(const CARLOW_W& orig) {
00018 }
00024 std::shared_ptr<OSTM> CARLOW_W::getBaseCopy(std::shared_ptr<OSTM> object)
00025 {
00026
00027     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00028     std::shared_ptr<WAREHOUSE> obj(new CARLOW_W(objTO, object->Get_Version(), object->Get_Unique_ID(
00029 )));
00029     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00030     return ostm_obj;
00031 }
00037 void CARLOW_W::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00038
00039     std::shared_ptr<CARLOW_W> objTO = std::dynamic_pointer_cast<CARLOW_W>(to);
00040     std::shared_ptr<CARLOW_W> objFROM = std::dynamic_pointer_cast<CARLOW_W>(from);
00041     objTO->_shop_address = objFROM->GetShop_address();
00042     objTO->_shop_name = objFROM->GetShop_name();
00043     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00044     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00045     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00046     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00047     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00048     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00049     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00050     objTO->Set_Version(objFROM->Get_Version());
00051
00052
00053 }
00057 //std::shared_ptr<CARLOW_W> CARLOW_W::_cast(std::shared_ptr<OSTM> _object){
00058 //
00059 //     return std::static_pointer_cast<CARLOW_W>(_object);
00060 //}
00064 void CARLOW_W::toString()
00065 {
00066     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
<< this->GetShop_address() << "\nNo. iPhones : " << this->
GetNumber_of_iphones() << "\nNo. Samsung : " << this->
GetNumber_of_samsung() << "\nNo. Sony : " << this->
GetNumber_of_sony() << "\nNo. Huawei : " << this->
GetNumber_of_huawei() << "\nNo. Nokia : " << this->
GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
GetNumber_of_alcatel() << "\nVersion number : " << this->
Get_Version() << std::endl;
00067 }
00068
00069
00070
00071 void CARLOW_W::SetNumber_of_alcatel(int
_number_of_alcatel) {
00072     this->_number_of_alcatel = _number_of_alcatel;
00073 }
00074
00075 int CARLOW_W::GetNumber_of_alcatel(){
00076     return _number_of_alcatel;
00077 }
00078
00079 void CARLOW_W::SetNumber_of_nokia(int
_number_of_nokia) {
00080     this->_number_of_nokia = _number_of_nokia;
00081 }
00082
00083 int CARLOW_W::GetNumber_of_nokia(){
00084     return _number_of_nokia;
00085 }
00086
00087 void CARLOW_W::SetNumber_of_huawei(int
_number_of_huawei) {
00088     this->_number_of_huawei = _number_of_huawei;
00089 }
00090
00091 int CARLOW_W::GetNumber_of_huawei(){
00092     return _number_of_huawei;
00093 }
00094
00095 void CARLOW_W::SetNumber_of_sony(int _number_of_sony) {
00096     this->_number_of_sony = _number_of_sony;
00097 }
00098

```

```

00099 int CARLOW_W::GetNumber_of_sony() {
00100     return _number_of_sony;
00101 }
00102
00103 void CARLOW_W::SetNumber_of_samsung(int
    _number_of_samsung) {
00104     this->_number_of_samsung = _number_of_samsung;
00105 }
00106
00107 int CARLOW_W::GetNumber_of_samsung() {
00108     return _number_of_samsung;
00109 }
00110
00111 void CARLOW_W::SetNumber_of_iphones(int
    _number_of_iphones) {
00112     this->_number_of_iphones = _number_of_iphones;
00113 }
00114
00115 int CARLOW_W::GetNumber_of_iphones() {
00116     return _number_of_iphones;
00117 }
00118
00119 void CARLOW_W::SetShop_name(std::string _shop_name) {
00120     this->_shop_name = _shop_name;
00121 }
00122
00123 std::string CARLOW_W::GetShop_name() {
00124     return _shop_name;
00125 }
00126
00127 void CARLOW_W::SetShop_address(std::string
    _shop_address) {
00128     this->_shop_address = _shop_address;
00129 }
00130
00131 std::string CARLOW_W::GetShop_address() {
00132     return _shop_address;
00133 }
00134
00135
00136

```

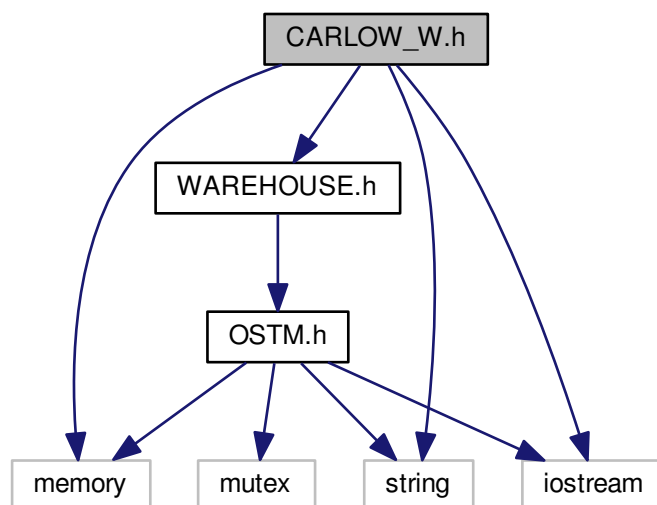
6.19 CARLOW_W.h File Reference

```

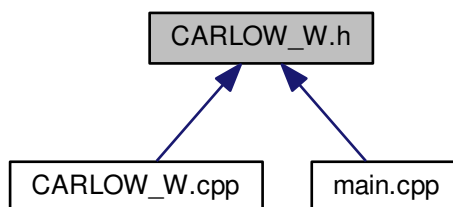
#include "WAREHOUSE.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for CARLOW_W.h:



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



Classes

- class [CARLOW_W](#)

6.20 CARLOW_W.h

```

00001
00002 /*
00003  * File:   CARLOW_W.h
00004  * Author: Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
  
```

```

00008  */
00009
00010 #ifndef CARLOW_W_H
00011 #define CARLOW_W_H
00012 #include "WAREHOUSE.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class CARLOW_W :public WAREHOUSE {
00020 public:
00024     CARLOW_W() : WAREHOUSE() {
00025
00026         this->_shop_address = "Carlow potato street";
00027         this->_shop_name = "CARLOW C_WAREHOUSE";
00028         this->_number_of_iphones = 200;
00029         this->_number_of_samsung = 200;
00030         this->_number_of_sony = 200;
00031         this->_number_of_huawei = 200;
00032         this->_number_of_nokia = 200;
00033         this->_number_of_alcatel = 200;
00034     };
00038     CARLOW_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int
huawei, int nokia, int alcatel): WAREHOUSE() {
00039         /*
00040         * copy over values
00041         */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };
00055     CARLOW_W(std::shared_ptr<WAREHOUSE> obj, int _version, int _unique_id):
WAREHOUSE(_version, _unique_id) {
00056         /*
00057         * copy over values
00058         */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }
00071     CARLOW_W(const CARLOW_W& orig);
00075     CARLOW_W operator=(const CARLOW_W& orig){};
00079     virtual ~CARLOW_W();
00080     /*
00081     * Implement OSTM virtual methods
00082     */
00083     // virtual std::shared_ptr<CARLOW_W> _cast(std::shared_ptr<OSTM> _object);
00084     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00085     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00086     virtual void toString();
00087     /*
00088     * Implement Warehouse methods
00089     */
00090     virtual void SetNumber_of_alcatel(int _number_of_alcatel);
00091     virtual int GetNumber_of_alcatel();
00092     virtual void SetNumber_of_nokia(int _number_of_nokia);
00093     virtual int GetNumber_of_nokia();
00094     virtual void SetNumber_of_huawei(int _number_of_huawei);
00095     virtual int GetNumber_of_huawei();
00096     virtual void SetNumber_of_sony(int _number_of_sony);
00097     virtual int GetNumber_of_sony();
00098     virtual void SetNumber_of_samsung(int _number_of_samsung);
00099     virtual int GetNumber_of_samsung();
00100     virtual void SetNumber_of_iphones(int _number_of_iphones);
00101     virtual int GetNumber_of_iphones();
00102     virtual void SetShop_name(std::string _shop_name);
00103     virtual std::string GetShop_name();
00104     virtual void SetShop_address(std::string _shop_address);
00105     virtual std::string GetShop_address();
00106
00107 private:
00109     std::string _shop_address;
00110     std::string _shop_name;
00111     int _number_of_iphones;
00112     int _number_of_samsung;
00113     int _number_of_sony;

```



```

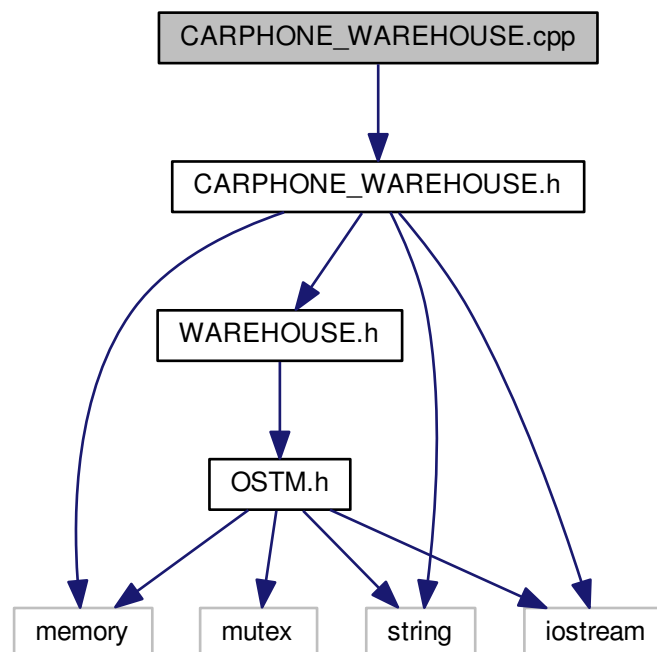
00114     int _number_of_huawei;
00115     int _number_of_nokia;
00116     int _number_of_alcatel;
00117
00118 };
00119
00120 #endif /* CARLOW_W_H */
00121

```

6.21 CARPHONE_WAREHOUSE.cpp File Reference

```
#include "CARPHONE_WAREHOUSE.h"
```

Include dependency graph for CARPHONE_WAREHOUSE.cpp:



6.22 CARPHONE_WAREHOUSE.cpp

```

00001
00002 /*
00003  * File:    CARPHONE_WAREHOUSE.cpp
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009 #include "CARPHONE_WAREHOUSE.h"
00010
00011 CARPHONE_WAREHOUSE::CARPHONE_WAREHOUSE(const
00012     CARPHONE_WAREHOUSE& orig) {
00013 }
00014 CARPHONE_WAREHOUSE::~CARPHONE_WAREHOUSE() {
00015 }
00021 std::shared_ptr<OSTM> CARPHONE_WAREHOUSE::getBaseCopy(std::shared_ptr<OSTM>
    object)

```

```

00022 {
00023
00024     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00025     std::shared_ptr<WAREHOUSE> obj(new CARPHONE_WAREHOUSE(objTO, object->Get_Version(),
00026         object->Get_Unique_ID()));
00026     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00027     return ostm_obj;
00028 }
00034 void CARPHONE_WAREHOUSE::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from)
00035 {
00036     std::shared_ptr<CARPHONE_WAREHOUSE> objTO = std::dynamic_pointer_cast<
00037 CARPHONE_WAREHOUSE>(to);
00037     std::shared_ptr<CARPHONE_WAREHOUSE> objFROM = std::dynamic_pointer_cast<
00038 CARPHONE_WAREHOUSE>(from);
00038     objTO->_shop_address = objFROM->GetShop_address();
00039     objTO->_shop_name = objFROM->GetShop_name();
00040     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00041     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00042     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00043     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00044     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00045     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00046     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00047     objTO->Set_Version(objFROM->Get_Version());
00048
00049 }
00053 //std::shared_ptr<CARPHONE_WAREHOUSE> CARPHONE_WAREHOUSE::_cast(std::shared_ptr<OSTM> _object){
00054 //
00055 //     return std::static_pointer_cast<CARPHONE_WAREHOUSE>(_object);
00056 //}
00060 void CARPHONE_WAREHOUSE::toString()
00061 {
00062     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
00063 Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address :
00064 " << this->GetShop_address() << "\nNo. Iphones : " << this->
00065 GetNumber_of_iphones() << "\nNo. Samsung : " << this->
00066 GetNumber_of_samsung() << "\nNo. Sony : " << this->
00067 GetNumber_of_sony() << "\nNo. Huawei : " << this->
00068 GetNumber_of_huawei() << "\nNo. Nokia : " << this->
00069 GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
00070 GetNumber_of_alcatel() << "\nVersion number : " << this->
00071 Get_Version() << std::endl;
00072 }
00073
00074 void CARPHONE_WAREHOUSE::SetNumber_of_alcatel(int
00075 _number_of_alcatel) {
00076     this->_number_of_alcatel = _number_of_alcatel;
00077 }
00078
00079 int CARPHONE_WAREHOUSE::GetNumber_of_alcatel() {
00080     return _number_of_alcatel;
00081 }
00082
00083 void CARPHONE_WAREHOUSE::SetNumber_of_nokia(int
00084 _number_of_nokia) {
00085     this->_number_of_nokia = _number_of_nokia;
00086 }
00087
00088 int CARPHONE_WAREHOUSE::GetNumber_of_nokia() {
00089     return _number_of_nokia;
00090 }
00091
00092 void CARPHONE_WAREHOUSE::SetNumber_of_huawei(int
00093 _number_of_huawei) {
00094     this->_number_of_huawei = _number_of_huawei;
00095 }
00096
00097 int CARPHONE_WAREHOUSE::GetNumber_of_huawei() {
00098     return _number_of_huawei;
00099 }
00100
00101 void CARPHONE_WAREHOUSE::SetNumber_of_samsung(int
00102 _number_of_samsung) {
00103     this->_number_of_samsung = _number_of_samsung;
00104 }
00105

```

```

00102
00103 int CARPHONE_WAREHOUSE::GetNumber_of_samsung() {
00104     return _number_of_samsung;
00105 }
00106
00107 void CARPHONE_WAREHOUSE::SetNumber_of_iphones(int
00108     _number_of_iphones) {
00109     this->_number_of_iphones = _number_of_iphones;
00110 }
00111 int CARPHONE_WAREHOUSE::GetNumber_of_iphones() {
00112     return _number_of_iphones;
00113 }
00114
00115 void CARPHONE_WAREHOUSE::SetShop_name(std::string
00116     _shop_name) {
00117     this->_shop_name = _shop_name;
00118 }
00119 std::string CARPHONE_WAREHOUSE::GetShop_name() {
00120     return _shop_name;
00121 }
00122
00123 void CARPHONE_WAREHOUSE::SetShop_address(std::string
00124     _shop_address) {
00125     this->_shop_address = _shop_address;
00126 }
00127 std::string CARPHONE_WAREHOUSE::GetShop_address() {
00128     return _shop_address;
00129 }
00130
00131
00132

```

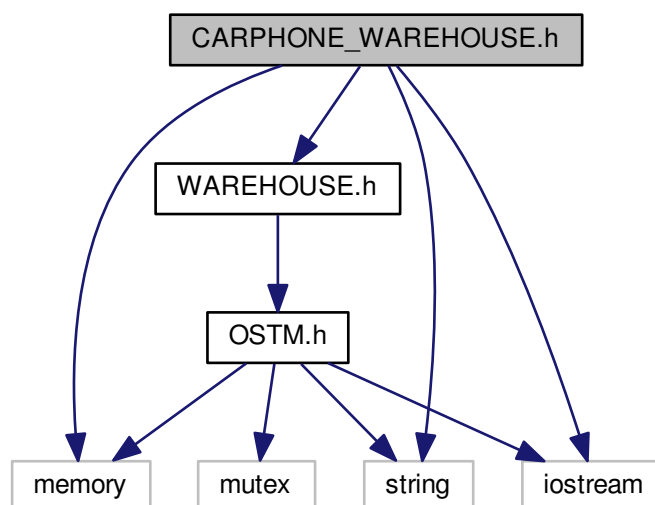
6.23 CARPHONE_WAREHOUSE.h File Reference

```

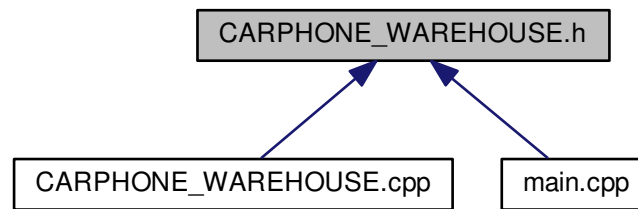
#include "WAREHOUSE.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for CARPHONE_WAREHOUSE.h:



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



Classes

- class [CARPHONE_WAREHOUSE](#)

6.24 CARPHONE_WAREHOUSE.h

```

00001
00002 /*
00003  * File:    CARPHONE_WAREHOUSE.h
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef CARPHONE_WAREHOUSE_H
00011 #define CARPHONE_WAREHOUSE_H
00012 #include "WAREHOUSE.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class CARPHONE_WAREHOUSE :public WAREHOUSE {
00020 public:
00024     CARPHONE_WAREHOUSE() : WAREHOUSE() {
00025
00026         this->_shop_address = "DUBLIN XII";
00027         this->_shop_name = "DISTRIBUTION CENTER";
00028         this->_number_of_iphones = 10000;
00029         this->_number_of_samsung = 10000;
00030         this->_number_of_sony = 10000;
00031         this->_number_of_huawei = 10000;
00032         this->_number_of_nokia = 10000;
00033         this->_number_of_alcatel = 10000;
00034     };
00038     CARPHONE_WAREHOUSE(std::string address, std::string shop_name, int iphone, int
samsung, int sony, int huawei, int nokia, int alcatel): WAREHOUSE(){
00039         /*
00040          * copy over values
00041          */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };
00055     CARPHONE_WAREHOUSE(std::shared_ptr<WAREHOUSE> obj, int _version, int _unique_id):
WAREHOUSE(_version, _unique_id){
00056         /*
00057          * copy over values
00058          */

```

```

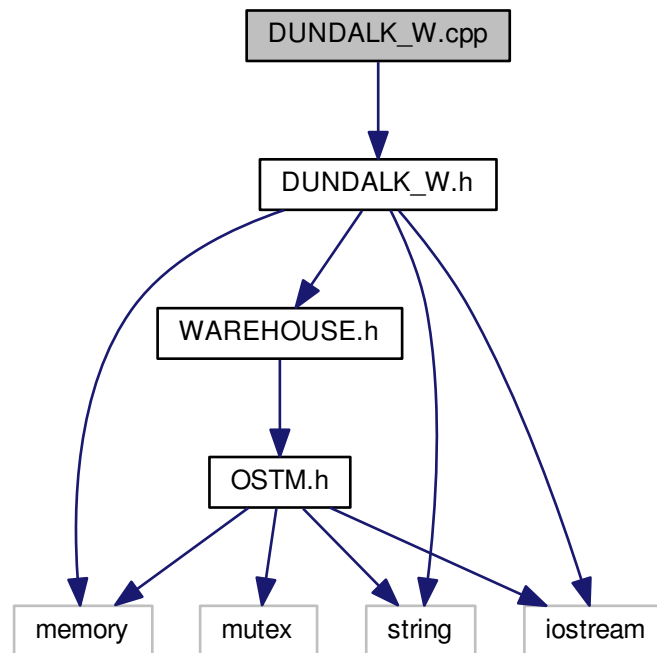
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }
00071     CARPHONE_WAREHOUSE(const CARPHONE_WAREHOUSE& orig);
00075     CARPHONE_WAREHOUSE operator=(const
CARPHONE_WAREHOUSE& orig) {};
00079     virtual ~CARPHONE_WAREHOUSE();
00080     /*
00081     * Implement OSTM virtual methods
00082     */
00083     //virtual std::shared_ptr<CARPHONE_WAREHOUSE> _cast(std::shared_ptr<OSTM> _object);
00084     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00085     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00086
00087
00088
00089     virtual void toString();
00090     /*
00091     * Implement Warehouse methods
00092     */
00093     virtual void SetNumber_of_alcatel(int _number_of_alcatel);
00094     virtual int GetNumber_of_alcatel();
00095     virtual void SetNumber_of_nokia(int _number_of_nokia);
00096     virtual int GetNumber_of_nokia();
00097     virtual void SetNumber_of_huawei(int _number_of_huawei);
00098     virtual int GetNumber_of_huawei();
00099     virtual void SetNumber_of_sony(int _number_of_sony);
00100     virtual int GetNumber_of_sony();
00101     virtual void SetNumber_of_samsung(int _number_of_samsung);
00102     virtual int GetNumber_of_samsung();
00103     virtual void SetNumber_of_iphones(int _number_of_iphones);
00104     virtual int GetNumber_of_iphones();
00105     virtual void SetShop_name(std::string _shop_name);
00106     virtual std::string GetShop_name();
00107     virtual void SetShop_address(std::string _shop_address);
00108     virtual std::string GetShop_address();
00109
00110 private:
00111     std::string _shop_address;
00112     std::string _shop_name;
00113     int _number_of_iphones;
00114     int _number_of_samsung;
00115     int _number_of_sony;
00116     int _number_of_huawei;
00117     int _number_of_nokia;
00118     int _number_of_alcatel;
00119
00120 };
00121
00122 #endif /* CARPHONE_WAREHOUSE_H */
00123

```

6.25 DUNDALK_W.cpp File Reference

```
#include "DUNDALK_W.h"
```

Include dependency graph for DUNDALK_W.cpp:



6.26 DUNDALK_W.cpp

```

00001
00002 /*
00003  * File:    DUNDALK_W.cpp
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #include "DUNDALK_W.h"
00011
00012 DUNDALK_W::~DUNDALK_W() {
00013 }
00014
00015 DUNDALK_W::DUNDALK_W(const DUNDALK_W& orig) {
00016 }
00022 std::shared_ptr<OSTM> DUNDALK_W::getBaseCopy(std::shared_ptr<OSTM> object)
00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new DUNDALK_W(objTO, object->Get_Version(),object->
00027         Get_Unique_ID()));
00028     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00029     return ostm_obj;
00035 void DUNDALK_W::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00036
00037     std::shared_ptr<DUNDALK_W> objTO = std::dynamic_pointer_cast<DUNDALK_W>(to);
00038     std::shared_ptr<DUNDALK_W> objFROM = std::dynamic_pointer_cast<DUNDALK_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();

```

```

00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
00055 //std::shared_ptr<DUNDALK_W> DUNDALK_W::_cast(std::shared_ptr<OSTM> _object) {
00056 //
00057 //     return std::static_pointer_cast<DUNDALK_W>(_object);
00058 //}
00062 void DUNDALK_W::toString()
00063 {
00064     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
        Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
        << this->GetShop_address() << "\nNo. Iphones : " << this->
        GetNumber_of_iphones() << "\nNo. Samsung : " << this->
        GetNumber_of_samsung() << "\nNo. Sony : " << this->
        GetNumber_of_sony() << "\nNo. Huawei : " << this->
        GetNumber_of_huawei() << "\nNo. Nokia : " << this->
        GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
        GetNumber_of_alcatel() << "\nVersion number : " << this->
        Get_Version() << std::endl;
00065 }
00066
00067
00068
00069 void DUNDALK_W::SetNumber_of_alcatel(int
    _number_of_alcatel) {
00070     this->_number_of_alcatel = _number_of_alcatel;
00071 }
00072
00073 int DUNDALK_W::GetNumber_of_alcatel() {
00074     return _number_of_alcatel;
00075 }
00076
00077 void DUNDALK_W::SetNumber_of_nokia(int
    _number_of_nokia) {
00078     this->_number_of_nokia = _number_of_nokia;
00079 }
00080
00081 int DUNDALK_W::GetNumber_of_nokia() {
00082     return _number_of_nokia;
00083 }
00084
00085 void DUNDALK_W::SetNumber_of_huawei(int
    _number_of_huawei) {
00086     this->_number_of_huawei = _number_of_huawei;
00087 }
00088
00089 int DUNDALK_W::GetNumber_of_huawei() {
00090     return _number_of_huawei;
00091 }
00092
00093 void DUNDALK_W::SetNumber_of_sony(int
    _number_of_sony) {
00094     this->_number_of_sony = _number_of_sony;
00095 }
00096
00097 int DUNDALK_W::GetNumber_of_sony() {
00098     return _number_of_sony;
00099 }
00100
00101 void DUNDALK_W::SetNumber_of_samsung(int
    _number_of_samsung) {
00102     this->_number_of_samsung = _number_of_samsung;
00103 }
00104
00105 int DUNDALK_W::GetNumber_of_samsung() {
00106     return _number_of_samsung;
00107 }
00108
00109 void DUNDALK_W::SetNumber_of_iphones(int
    _number_of_iphones) {
00110     this->_number_of_iphones = _number_of_iphones;
00111 }
00112
00113 int DUNDALK_W::GetNumber_of_iphones() {
00114     return _number_of_iphones;
00115 }
00116
00117 void DUNDALK_W::SetShop_name(std::string _shop_name) {
00118     this->_shop_name = _shop_name;
00119 }
00120
00121 std::string DUNDALK_W::GetShop_name() {
00122     return _shop_name;

```

```

00123 }
00124
00125 void DUNDALK_W::SetShop_address(std::string
    _shop_address) {
00126     this->_shop_address = _shop_address;
00127 }
00128
00129 std::string DUNDALK_W::GetShop_address() {
00130     return _shop_address;
00131 }
00132
00133
00134

```

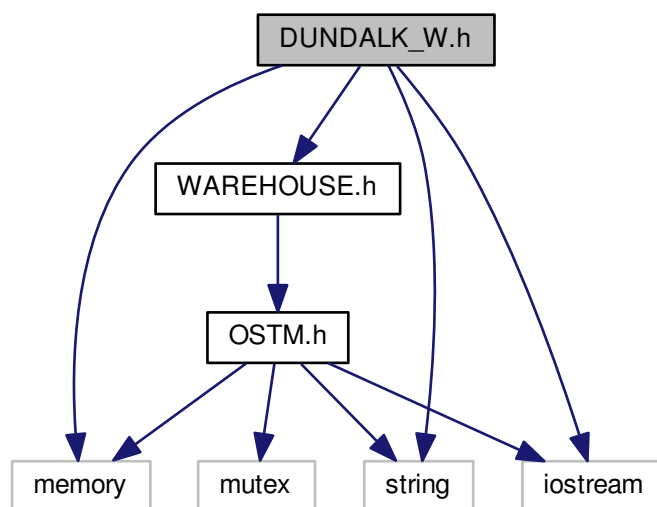
6.27 DUNDALK_W.h File Reference

```

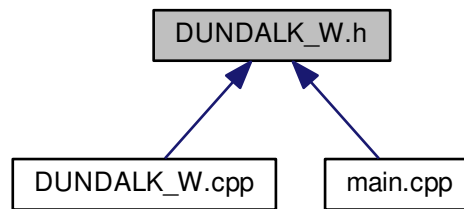
#include "WAREHOUSE.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for DUNDALK_W.h:



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



Classes

- class [DUNDALK_W](#)

6.28 DUNDALK_W.h

```

00001
00002 /*
00003  * File:    DUNDALK_W.h
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef DUNDALK_W_H
00011 #define DUNDALK_W_H
00012 #include "WAREHOUSE.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class DUNDALK_W :public WAREHOUSE {
00020 public:
00024     DUNDALK_W() : WAREHOUSE(){
00025
00026         this->_shop_address = "Dundalk Busy Street";
00027         this->_shop_name = "DUNDALK D_WAREHOUSE";
00028         this->_number_of_iphones = 200;
00029         this->_number_of_samsung = 200;
00030         this->_number_of_sony = 200;
00031         this->_number_of_huawei = 200;
00032         this->_number_of_nokia = 200;
00033         this->_number_of_alcatel = 200;
00034     };
00038     DUNDALK_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int
00039     huawei, int nokia, int alcatel): WAREHOUSE(){
00039         /*
00040         * copy over values
00041         */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };
00055     DUNDALK_W(std::shared_ptr<WAREHOUSE> obj, int _version, int _unique_id):
00056     WAREHOUSE(_version, _unique_id){
00056         /*
00057         * copy over values
00058         */

```

```

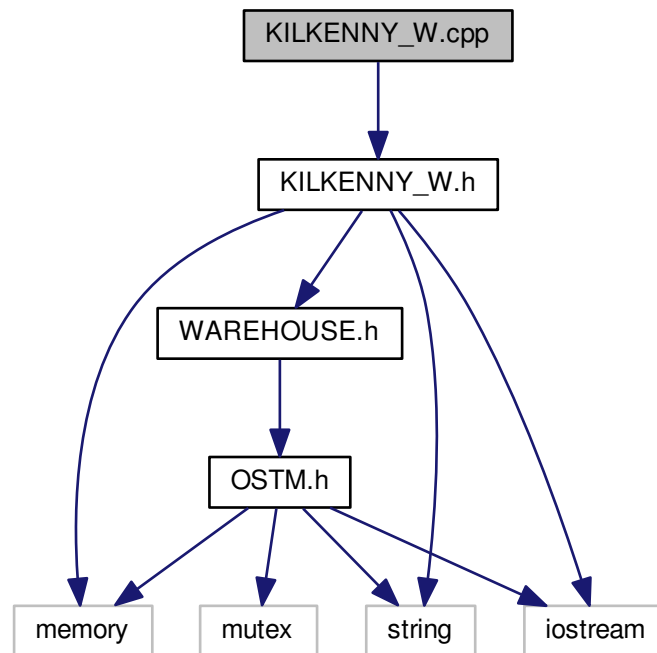
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }
00071     DUNDALK_W(const DUNDALK_W& orig);
00075     DUNDALK_W operator=(const DUNDALK_W& orig){};
00079     virtual ~DUNDALK_W();
00080     /*
00081     * Implement OSTM virtual methods
00082     */
00083     //virtual std::shared_ptr<DUNDALK_W> _cast(std::shared_ptr<OSTM> _object);
00084     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00085     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00086     virtual void toString();
00087     /*
00088     * Implement Warehouse methods
00089     */
00090     virtual void SetNumber_of_alcatel(int _number_of_alcatel);
00091     virtual int GetNumber_of_alcatel();
00092     virtual void SetNumber_of_nokia(int _number_of_nokia);
00093     virtual int GetNumber_of_nokia();
00094     virtual void SetNumber_of_huawei(int _number_of_huawei);
00095     virtual int GetNumber_of_huawei();
00096     virtual void SetNumber_of_sony(int _number_of_sony);
00097     virtual int GetNumber_of_sony();
00098     virtual void SetNumber_of_samsung(int _number_of_samsung);
00099     virtual int GetNumber_of_samsung();
00100     virtual void SetNumber_of_iphones(int _number_of_iphones);
00101     virtual int GetNumber_of_iphones();
00102     virtual void SetShop_name(std::string _shop_name);
00103     virtual std::string GetShop_name();
00104     virtual void SetShop_address(std::string _shop_address);
00105     virtual std::string GetShop_address();
00106
00107
00108 private:
00109     std::string _shop_address;
00110     std::string _shop_name;
00111     int _number_of_iphones;
00112     int _number_of_samsung;
00113     int _number_of_sony;
00114     int _number_of_huawei;
00115     int _number_of_nokia;
00116     int _number_of_alcatel;
00117
00118 };
00119
00120 #endif /* DUNDALK_W_H */
00121

```

6.29 KILKENNY_W.cpp File Reference

```
#include "KILKENNY_W.h"
```

Include dependency graph for KILKENNY_W.cpp:



6.30 KILKENNY_W.cpp

```

00001
00002 /*
00003  * File:    KILKENNY_W.cpp
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #include "KILKENNY_W.h"
00011
00012 KILKENNY_W::~KILKENNY_W() {
00013 }
00014
00015 KILKENNY_W::KILKENNY_W(const KILKENNY_W& orig) {
00016 }
00022 std::shared_ptr<OSTM> KILKENNY_W::getBaseCopy(std::shared_ptr<OSTM> object)
00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new KILKENNY_W(objTO, object->Get_Version(),object->
00027         Get_Unique_ID()));
00028     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00029     return ostm_obj;
00035 void KILKENNY_W::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00036
00037     std::shared_ptr<KILKENNY_W> objTO = std::dynamic_pointer_cast<KILKENNY_W>(to);
00038     std::shared_ptr<KILKENNY_W> objFROM = std::dynamic_pointer_cast<KILKENNY_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();

```

```

00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
00055 //std::shared_ptr<KILKENNY_W> KILKENNY_W::_cast(std::shared_ptr<OSTM> _object){
00056 //
00057 //     return static_cast<std::shared_ptr<KILKENNY_W>>(_object);
00058 //}
00062 void KILKENNY_W::toString()
00063 {
00064     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
        Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
        << this->GetShop_address() << "\nNo. Iphones : " << this->
        GetNumber_of_iphones() << "\nNo. Samsung : " << this->
        GetNumber_of_samsung() << "\nNo. Sony : " << this->
        GetNumber_of_sony() << "\nNo. Huawei : " << this->
        GetNumber_of_huawei() << "\nNo. Nokia : " << this->
        GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
        GetNumber_of_alcatel() << "\nVersion number : " << this->
        Get_Version() << std::endl;
00065 }
00066
00067
00068
00069 void KILKENNY_W::SetNumber_of_alcatel(int
    _number_of_alcatel) {
00070     this->_number_of_alcatel = _number_of_alcatel;
00071 }
00072
00073 int KILKENNY_W::GetNumber_of_alcatel(){
00074     return _number_of_alcatel;
00075 }
00076
00077 void KILKENNY_W::SetNumber_of_nokia(int
    _number_of_nokia) {
00078     this->_number_of_nokia = _number_of_nokia;
00079 }
00080
00081 int KILKENNY_W::GetNumber_of_nokia(){
00082     return _number_of_nokia;
00083 }
00084
00085 void KILKENNY_W::SetNumber_of_huawei(int
    _number_of_huawei) {
00086     this->_number_of_huawei = _number_of_huawei;
00087 }
00088
00089 int KILKENNY_W::GetNumber_of_huawei(){
00090     return _number_of_huawei;
00091 }
00092
00093 void KILKENNY_W::SetNumber_of_sony(int
    _number_of_sony) {
00094     this->_number_of_sony = _number_of_sony;
00095 }
00096
00097 int KILKENNY_W::GetNumber_of_sony(){
00098     return _number_of_sony;
00099 }
00100
00101 void KILKENNY_W::SetNumber_of_samsung(int
    _number_of_samsung) {
00102     this->_number_of_samsung = _number_of_samsung;
00103 }
00104
00105 int KILKENNY_W::GetNumber_of_samsung(){
00106     return _number_of_samsung;
00107 }
00108
00109 void KILKENNY_W::SetNumber_of_iphones(int
    _number_of_iphones) {
00110     this->_number_of_iphones = _number_of_iphones;
00111 }
00112
00113 int KILKENNY_W::GetNumber_of_iphones(){
00114     return _number_of_iphones;
00115 }
00116
00117 void KILKENNY_W::SetShop_name(std::string _shop_name) {
00118     this->_shop_name = _shop_name;
00119 }
00120
00121 std::string KILKENNY_W::GetShop_name(){
00122     return _shop_name;

```

```

00123 }
00124
00125 void KILKENNY_W::SetShop_address(std::string
    _shop_address) {
00126     this->_shop_address = _shop_address;
00127 }
00128
00129 std::string KILKENNY_W::GetShop_address() {
00130     return _shop_address;
00131 }
00132

```

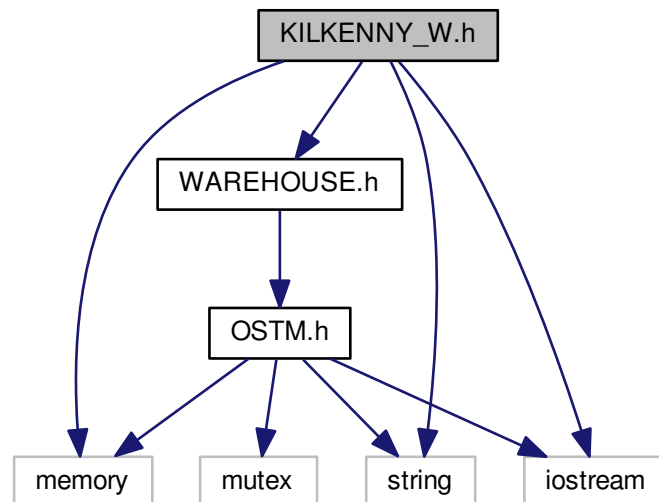
6.31 KILKENNY_W.h File Reference

```

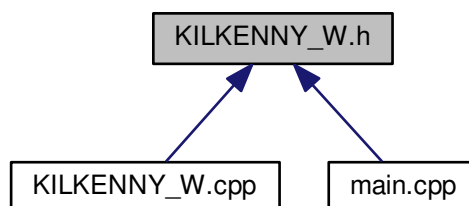
#include "WAREHOUSE.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for KILKENNY_W.h:



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



Classes

- class `KILKENNY_W`

6.32 KILKENNY_W.h

```

00001
00002 /*
00003  * File:    KILKENNY_W.h
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef KILKENNY_W_H
00011 #define KILKENNY_W_H
00012 #include "WAREHOUSE.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class KILKENNY_W : public WAREHOUSE {
00020 public:
00024     KILKENNY_W(): WAREHOUSE() {
00025
00026         this->_shop_address = "Kilkenny High Street";
00027         this->_shop_name = "KILKENNY K_WAREHOUSE";
00028         this->_number_of_iphones = 200;
00029         this->_number_of_samsung = 200;
00030         this->_number_of_sony = 200;
00031         this->_number_of_huawei = 200;
00032         this->_number_of_nokia = 200;
00033         this->_number_of_alcatel = 200;
00034     };
00038     KILKENNY_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int
huawei, int nokia, int alcatel): WAREHOUSE(){
00039         /*
00040          * copy over values
00041          */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };
00055     KILKENNY_W(std::shared_ptr<WAREHOUSE> obj, int _version, int _unique_id):
WAREHOUSE(_version, _unique_id){
00056         /*
00057          * copy over values
00058          */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }
00071     KILKENNY_W(const KILKENNY_W& orig);
00075     KILKENNY_W operator=(const KILKENNY_W& orig){};
00079     virtual ~KILKENNY_W();
00080     /*
00081      * Implement OSTM virtual methods
00082      */
00083     //virtual std::shared_ptr<KILKENNY_W> _cast(std::shared_ptr<OSTM> _object);
00084     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00085     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00086     virtual void toString();
00087     /*
00088      * Implement Warehouse methods
00089      */
00090     virtual void SetNumber_of_alcatel(int _number_of_alcatel);
00091     virtual int GetNumber_of_alcatel();
00092     virtual void SetNumber_of_nokia(int _number_of_nokia);
00093     virtual int GetNumber_of_nokia();
00094     virtual void SetNumber_of_huawei(int _number_of_huawei);
00095     virtual int GetNumber_of_huawei();

```

```

00096     virtual void SetNumber_of_sony(int _number_of_sony);
00097     virtual int GetNumber_of_sony();
00098     virtual void SetNumber_of_samsung(int _number_of_samsung);
00099     virtual int GetNumber_of_samsung();
00100     virtual void SetNumber_of_iphones(int _number_of_iphones);
00101     virtual int GetNumber_of_iphones();
00102     virtual void SetShop_name(std::string _shop_name);
00103     virtual std::string GetShop_name();
00104     virtual void SetShop_address(std::string _shop_address);
00105     virtual std::string GetShop_address();
00106
00107
00108 private:
00109     std::string _shop_address;
00110     std::string _shop_name;
00111     int _number_of_iphones;
00112     int _number_of_samsung;
00113     int _number_of_sony;
00114     int _number_of_huawei;
00115     int _number_of_nokia;
00116     int _number_of_alcatel;
00117
00118 };
00119
00120 #endif /* KILKENNY_W_H */
00121

```

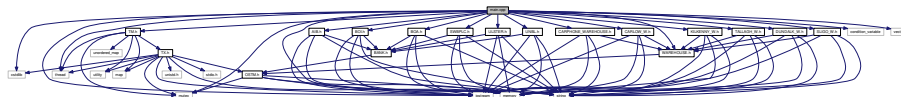
6.33 main.cpp File Reference

```

#include <cstdlib>
#include <iostream>
#include <thread>
#include "TM.h"
#include "AIB.h"
#include "BOI.h"
#include "BOA.h"
#include "SWBPLC.h"
#include "ULSTER.h"
#include "UNBL.h"
#include "WAREHOUSE.h"
#include "CARPHONE_WAREHOUSE.h"
#include "CARLOW_W.h"
#include "KILKENNY_W.h"
#include "TALLAGH_W.h"
#include "DUNDALK_W.h"
#include "SLIGO_W.h"
#include <mutex>
#include <memory>
#include <condition_variable>
#include <vector>

```

Include dependency graph for main.cpp:



Functions

- void [_complex_transfer_](#) (std::shared_ptr< [OSTM](#) > _from_, std::shared_ptr< [OSTM](#) > _from_two_, std::vector< std::shared_ptr< [OSTM](#) >> _customer_vec, [TM](#) &_tm, double _amount)

complex_transfer function, takes two `std::shared_ptr<OSTM>` pointer, a vector of `std::shared_ptr<OSTM>` pointers, the Transaction manager, and the amount to use in the transaction, and transfer the `_amount` value from booth single objects to the objects to the vector collection

- void `_complex_warehouse_transfer_` (`std::shared_ptr<OSTM>` `_to_`, `std::shared_ptr<OSTM>` `_to_two`, `std::shared_ptr<OSTM>` `_to_three`, `std::vector<std::shared_ptr<OSTM>>` `_warehouse_vec`, `std::shared_ptr<OSTM>` `_from_`, `TM & tm`, double `_amount`)
- void `_nested_warehouse_transfer_` (`std::shared_ptr<OSTM>` `_to_`, `std::shared_ptr<OSTM>` `_to_two`, `std::shared_ptr<OSTM>` `_to_three`, `std::shared_ptr<OSTM>` `_from_`, `TM & tm`, double `_amount`)

nested_warehouse_transfer function, takes three `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from one account to the another account

- void `_nesting_` (`std::shared_ptr<OSTM>` `_to_`, `std::shared_ptr<OSTM>` `_from_`, `TM & tm`, double `_amount`)

nesting function, takes two `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from one account to the another account This function create nested transactions inside the transaction, and call other function to nesting the transaction as well

- void `_six_account_transfer_` (`std::shared_ptr<OSTM>` `_to_`, `std::shared_ptr<OSTM>` `_from_one_`, `std::shared_ptr<OSTM>` `_from_two_`, `std::shared_ptr<OSTM>` `_from_three_`, `std::shared_ptr<OSTM>` `_from_four_`, `std::shared_ptr<OSTM>` `_from_five_`, `TM & tm`, double `_amount`)

six_account_transfer function, takes six `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from five account to one account

- void `_two_account_transfer_` (`std::shared_ptr<OSTM>` `_to_`, `std::shared_ptr<OSTM>` `_from_`, `TM & tm`, double `_amount`)

two_account_transfer function, takes two `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from one account to the another account

- void `_warehouse_transfer_` (`std::shared_ptr<OSTM>` `_to_`, `std::shared_ptr<OSTM>` `_from_`, `TM & tm`, double `_amount`)

warehouse_transfer function, takes two `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from one account to the another account

- int `main` (void)

Variables

- static int `vector_number` = 600

6.33.1 Function Documentation

- 6.33.1.1 void `_complex_transfer_` (`std::shared_ptr<OSTM>` `_from_`, `std::shared_ptr<OSTM>` `_from_two_`, `std::vector<std::shared_ptr<OSTM>>` `_customer_vec`, `TM & tm`, double `_amount`)

complex_transfer function, takes two `std::shared_ptr<OSTM>` pointer, a vector of `std::shared_ptr<OSTM>` pointers, the Transaction manager, and the amount to use in the transaction, and transfer the `_amount` value from booth single objects to the objects to the vector collection

Parameters

<code>std::shared_ptr<TX></code>	tx, Transaction Object
<code>std::shared_ptr<BANK></code>	type, <code>FROM & FROM_TWO & TO</code>
<code>std::shared_ptr<OSTM></code>	type, <code>FROM_OSTM_ONE & FROM_OSTM_TWO & TO_OSTM</code>

Register the two single account

Declare required pointers

Register customers accounts from the collection (vector)

From `std::shared_ptr<OSTM>` to `std::shared_ptr<BANK>` to access the virtual methods

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

Commit changes

Definition at line 294 of file [main.cpp](#).

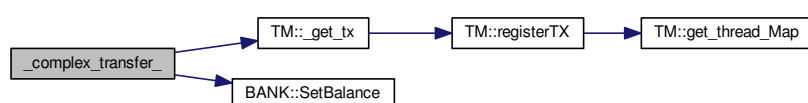
References [TM::_get_tx\(\)](#), and [BANK::SetBalance\(\)](#).

```

00294
00295     std::shared_ptr<TX> tx = _tm._get_tx();
00299     tx->_register(_from_);
00300     tx->_register(_from_two_);
00304     std::shared_ptr<OSTM> _FROM_OSTM_ONE_, _FROM_OSTM_TWO_, _TO_OSTM_;
00305     std::shared_ptr<BANK> _FROM_, _FROM_TWO_, _TO_;
00306
00307     bool done = false;
00308     try {
00309         while (!done) {
00310             // for (int i = 0; i < vector_number; ++i) {
00311             for (auto&& obj : _customer_vec) {
00315                 // auto&& obj = _customer_vec.at(i);
00316                 tx->_register(obj);
00320                 _FROM_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_));
00321                 _FROM_TWO_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_two_));
00322                 _TO_ = std::dynamic_pointer_cast<BANK> (tx->load(obj));
00326                 _FROM->SetBalance(_FROM->GetBalance() - _amount);
00327                 _FROM_TWO->SetBalance(_FROM_TWO->GetBalance() - _amount);
00328                 _TO->SetBalance(_TO->GetBalance() + (_amount * 2));
00332                 _FROM_OSTM_ONE_ = std::dynamic_pointer_cast<OSTM> (_FROM_);
00333                 _FROM_OSTM_TWO_ = std::dynamic_pointer_cast<OSTM> (_FROM_TWO_);
00334                 _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_);
00338                 tx->store(_FROM_OSTM_ONE_);
00339                 tx->store(_FROM_OSTM_TWO_);
00340                 tx->store(_TO_OSTM_);
00341             }
00345             done = tx->commit();
00346         }
00347     } catch (std::runtime_error& e) {
00348         std::cout << e.what() << std::endl;
00349     }
00350 }

```

Here is the call graph for this function:



```
6.33.1.2 void _complex_warehouse_transfer_ ( std::shared_ptr< OSTM > _to_, std::shared_ptr< OSTM >
      _to_two, std::shared_ptr< OSTM > _to_three, std::vector< std::shared_ptr< OSTM >> _warehouse_vec,
      std::shared_ptr< OSTM > _from_, TM & _tm, double _amount )
```

Register the two single account

Declare required pointers

Register customers accounts from the collection (vector)

From std::shared_ptr<OSTM> to std::shared_ptr<BANK> to access the virtual methods

Make changes with the objects

From std::shared_ptr<WAREHOUSE> to std::shared_ptr<OSTM> to store the memory spaces

Store changes

NESTED [WAREHOUSE TEST](#) [_to_two](#)

Make changes with the objects

From std::shared_ptr<BANK> to std::shared_ptr<OSTM> to store the memory spaces

Store changes

Commit changes

Definition at line 518 of file [main.cpp](#).

References [TM::get_tx\(\)](#), [_nested_warehouse_transfer\(\)](#), [_warehouse_transfer\(\)](#), and [WAREHOUSE::SetNumber_of_nokia\(\)](#).

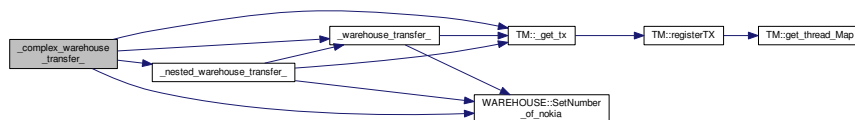
```
00518
{
00519     std::shared_ptr<TX> tx = _tm._get_tx();
00523     tx->_register(_to_);
00524     tx->_register(_to_two);
00525     tx->_register(_to_three);
00526     tx->_register(_from_);
00530     std::shared_ptr<WAREHOUSE> _TO_SHOP_, _TO_SHOP_TWO_, _TO_SHOP_VEC_, _FROM_DIST_;
00531     std::shared_ptr<OSTM> _TO_OSTM_, _TO_OSTM_TWO_, _TO_OSTM_VEC_, _FROM_OSTM_;
00532
00533     bool done = false;
00534     try {
00535         while (!done) {
00536
00537             // for (int i = 0; i < vector_number; ++i) {
00538             for (auto&& obj : _warehouse_vec) {
00542                 //auto&& obj = _warehouse_vec.at(i);
00543                 tx->_register(obj);
00547                 _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_));
00548                 _TO_SHOP_TWO_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_two));
00549                 _TO_SHOP_VEC_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(obj));
00550                 _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_from_));
00551
00555                 _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00556                 _TO_SHOP_TWO->SetNumber_of_nokia(_TO_SHOP_TWO->GetNumber_of_nokia() + _amount);
00557                 _TO_SHOP_VEC->SetNumber_of_nokia(_TO_SHOP_VEC->GetNumber_of_nokia() + _amount);
00558                 _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - (_amount * 3));
00559
00560                 _TO_SHOP_->SetNumber_of_samsung(_TO_SHOP_->GetNumber_of_samsung() + _amount);
00561                 _TO_SHOP_TWO->SetNumber_of_samsung(_TO_SHOP_TWO->GetNumber_of_samsung() + _amount);
00562                 _TO_SHOP_VEC->SetNumber_of_samsung(_TO_SHOP_VEC->GetNumber_of_samsung() + _amount);
00563                 _FROM_DIST_->SetNumber_of_samsung(_FROM_DIST_->GetNumber_of_samsung() - (_amount * 3));
00564
00565                 _TO_SHOP_->SetNumber_of_iphones(_TO_SHOP_->GetNumber_of_iphones() + _amount);
00566                 _TO_SHOP_TWO->SetNumber_of_iphones(_TO_SHOP_TWO->GetNumber_of_iphones() + _amount);
00567                 _TO_SHOP_VEC->SetNumber_of_iphones(_TO_SHOP_VEC->GetNumber_of_iphones() + _amount);
```

```

00568         _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - (_amount * 3));
00569
00570         _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00571         _TO_SHOP_TWO->SetNumber_of_sony(_TO_SHOP_TWO->GetNumber_of_sony() + _amount);
00572         _TO_SHOP_VEC->SetNumber_of_sony(_TO_SHOP_VEC->GetNumber_of_sony() + _amount);
00573         _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - (_amount * 3));
00574
00575         _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00576         _TO_OSTM_TWO = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_TWO);
00577         _TO_OSTM_VEC = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_VEC);
00578         _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00579         tx->store(_TO_OSTM_);
00580         tx->store(_TO_SHOP_TWO);
00581         tx->store(_TO_SHOP_VEC);
00582         tx->store(_FROM_OSTM_);
00583
00584     }
00585     std::shared_ptr<TX> txTwo = _tm._get_tx();
00586     bool nestedDone = false;
00587     while (!nestedDone) {
00588         _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_to_two));
00589         _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_from_));
00590         _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00591         _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - _amount);
00592
00593         _TO_SHOP_->SetNumber_of_samsung(_TO_SHOP_->GetNumber_of_samsung() + _amount);
00594         _FROM_DIST_->SetNumber_of_samsung(_FROM_DIST_->GetNumber_of_samsung() - _amount);
00595
00596         _TO_SHOP_->SetNumber_of_iphones(_TO_SHOP_->GetNumber_of_iphones() + _amount);
00597         _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - _amount);
00598
00599         _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00600         _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - _amount);
00601         _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00602         _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00603         txTwo->store(_TO_OSTM_);
00604         txTwo->store(_FROM_OSTM_);
00605
00606         /*
00607          * NESTED TRANSACTION TEST _to_three
00608
00609          */
00610         _warehouse_transfer_(_to_three, _from_, _tm, _amount);
00611         _nested_warehouse_transfer_(_to_, _to_two, _to_three, _from_,
00612         _tm, _amount);
00613
00614         nestedDone = tx->commit();
00615     }
00616     done = tx->commit();
00617 }
00618 } catch (std::runtime_error& e) {
00619     std::cout << e.what() << std::endl;
00620 }
00621 }
00622 }

```

Here is the call graph for this function:



6.33.1.3 void _nested_warehouse_transfer_ (std::shared_ptr< OSTM > _to_, std::shared_ptr< OSTM > _to_two, std::shared_ptr< OSTM > _to_three, std::shared_ptr< OSTM > _from_, TM & _tm, double _amount)

nested_warehouse_transfer function, takes three std::shared_ptr<OSTM> pointer, the Transaction manager, and the amount to use in the transaction and transfer the _amount value from one account to the another account

Parameters

<code>std::shared_ptr<TX></code>	tx, Transaction Object
<code>std::shared_ptr<WAREHOUSE></code>	type, <i>TO_SHOP</i> & <i>FROM_DIST</i>
<code>std::shared_ptr<OSTM></code>	type, <i>TO_OSTM</i> & <i>FROM_OSTM</i>

Register the two single account

Declare required pointers

From `std::shared_ptr<OSTM>` to `std::shared_ptr<BANK>` to access the virtual methods

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

NESTED [WAREHOUSE TEST](#) `_to_two`

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

Commit changes

Definition at line 419 of file [main.cpp](#).

References [TM::get_tx\(\)](#), [_warehouse_transfer_\(\)](#), and [WAREHOUSE::SetNumber_of_nokia\(\)](#).

Referenced by [_complex_warehouse_transfer_\(\)](#).

```

00419
00420     std::shared_ptr<TX> tx = _tm._get_tx();
00424     tx->_register(_to_);
00425     tx->_register(_to_two);
00426     tx->_register(_to_three);
00427     tx->_register(_from_);
00431     std::shared_ptr<WAREHOUSE> _TO_SHOP_, _FROM_DIST_;
00432     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00433
00434     bool done = false;
00435     try {
00436         while (!done) {
00440             _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_));
00441             _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_from_));
00445             _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00446             _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - _amount);
00447
00448             _TO_SHOP_->SetNumber_of_samsung(_TO_SHOP_->GetNumber_of_samsung() + _amount);
00449             _FROM_DIST_->SetNumber_of_samsung(_FROM_DIST_->GetNumber_of_samsung() - _amount);
00450
00451             _TO_SHOP_->SetNumber_of_iphones(_TO_SHOP_->GetNumber_of_iphones() + _amount);
00452             _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - _amount);
00453
00454             _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00455             _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - _amount);
00459             _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00460             _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00464             tx->store(_TO_OSTM_);
00465             tx->store(_FROM_OSTM_);
00466
00470             std::shared_ptr<TX> txTwo = _tm._get_tx();
00471             bool nestedDone = false;
00472             while (!nestedDone) {

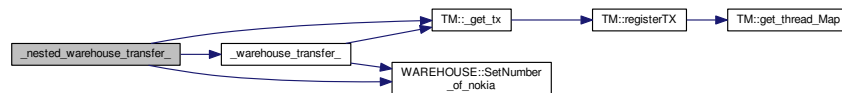
```

```

00473         _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_to_two));
00474         _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_from_));
00478         _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00479         _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - _amount);
00480
00481         _TO_SHOP_->SetNumber_of_samsung(_TO_SHOP_->GetNumber_of_samsung() + _amount);
00482         _FROM_DIST_->SetNumber_of_samsung(_FROM_DIST_->GetNumber_of_samsung() - _amount);
00483
00484         _TO_SHOP_->SetNumber_of_iphones(_TO_SHOP_->GetNumber_of_iphones() + _amount);
00485         _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - _amount);
00486
00487         _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00488         _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - _amount);
00492         _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00493         _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00497         txTwo->store(_TO_OSTM_);
00498         txTwo->store(_FROM_OSTM_);
00499
00500         /*
00501         * NESTED TRANSACTION TEST _to_three
00502
00503         */
00503         _warehouse_transfer_(_to_three, _from_, _tm, _amount);
00504
00505         nestedDone = tx->commit();
00507     }
00511     done = tx->commit();
00512 }
00513 } catch (std::runtime_error& e) {
00514     std::cout << e.what() << std::endl;
00515 }
00516 }

```

Here is the call graph for this function:



6.33.1.4 void _nesting_ (std::shared_ptr< OSTM > _to_, std::shared_ptr< OSTM > _from_, TM & _tm, double _amount)

nesting function, takes two `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from one account to the another account This function create nested transactions inside the transaction, and call other function to nesting the transaction as well

Parameters

<code>std::shared_ptr< TX></code>	tx, Transaction Object
<code>std::shared_ptr< BANK></code>	type, <i>TO_BANK</i> & <i>FROM_BANK</i>
<code>std::shared_ptr< OSTM></code>	type, <i>TO_OSTM</i> & <i>FROM_OSTM</i>

Register the two single account

Declare required pointers

From `std::shared_ptr<OSTM>` to `std::shared_ptr<BANK>` to access the virtual methods

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

NESTED TRANSACTION

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

NESTED TRANSACTION IN THE NESTED TRANSACTION *two_account_transfer* function call

Commit changes

Definition at line 206 of file `main.cpp`.

References `TM::get_tx()`, `_two_account_transfer_()`, and `BANK::SetBalance()`.

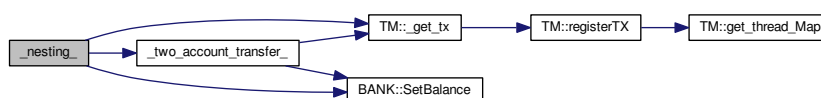
Referenced by `main()`.

```

00206
00207     std::shared_ptr<TX> tx = _tm._get_tx();
00211     tx->_register(_to_);
00212     tx->_register(_from_);
00216     std::shared_ptr<BANK> _TO_BANK_, _FROM_BANK_;
00217     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00218
00219
00220     bool done = false;
00221     try {
00222         while (!done) {
00226             _TO_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_to_));
00227             _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_));
00231             _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00232             _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00236             _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00237             _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00241             tx->store(_TO_OSTM_);
00242             tx->store(_FROM_OSTM_);
00243
00247             std::shared_ptr<TX> txTwo = _tm._get_tx();
00248
00249             bool nestedDone = false;
00250             while (!nestedDone) {
00251                 _TO_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_to_));
00252                 _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_from_));
00256                 _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00257                 _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00261                 _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00262                 _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00266                 txTwo->store(_TO_OSTM_);
00267                 txTwo->store(_FROM_OSTM_);
00272                 _two_account_transfer_(_to_, _from_, _tm, _amount);
00273
00274                 nestedDone = txTwo->commit();
00275             }
00276
00280             done = tx->commit();
00281         }
00282     } catch (std::runtime_error& e) {
00283         std::cout << e.what() << std::endl;
00284     }
00285 }

```

Here is the call graph for this function:



6.33.1.5 `void _six_account_transfer(std::shared_ptr< OSTM > _to_, std::shared_ptr< OSTM > _from_one_, std::shared_ptr< OSTM > _from_two_, std::shared_ptr< OSTM > _from_three_, std::shared_ptr< OSTM > _from_four_, std::shared_ptr< OSTM > _from_five_, TM & _tm, double _amount)`

`_six_account_transfer` function, takes six `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from five account to one account

Parameters

<code>std::shared_ptr<TX></code>	tx, Transaction Object
<code>std::shared_ptr<BANK></code>	type, <code>TO & FROM_ONE & FROM_TWO & FROM_THREE & FROM_FOUR & FROM_FIVE</code>
<code>std::shared_ptr<OSTM></code>	type, <code>_TO_OSTM & _FROM_ONE_OSTM & _FROM_TWO_OSTM & _FROM_THREE_OSTM & _FROM_FOUR_OSTM & _FROM_FIVE_OSTM</code>

Register the two single account

Required pointers to use in transaction

From `std::shared_ptr<OSTM>` to `std::shared_ptr<BANK>` to access the virtual methods

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

Commit changes

Definition at line 51 of file [main.cpp](#).

References [TM::get_tx\(\)](#), and [BANK::SetBalance\(\)](#).

```

00051
00052         {
00053             std::shared_ptr<TX> tx = _tm._get_tx();
00054             tx->_register(_to_);
00055             tx->_register(_from_one_);
00056             tx->_register(_from_two_);
00057             tx->_register(_from_three_);
00058             tx->_register(_from_four_);
00059             tx->_register(_from_five_);
00060             tx->_register(_from_five_);
00061             tx->_register(_from_five_);
00062
00063             std::shared_ptr<OSTM> _TO_OSTM, _FROM_ONE_OSTM, _FROM_TWO_OSTM, _FROM_THREE_OSTM, _FROM_FOUR_OSTM,
00064             _FROM_FIVE_OSTM;
00065             std::shared_ptr<BANK> _TO_, _FROM_ONE_, _FROM_TWO_, _FROM_THREE_, _FROM_FOUR_, _FROM_FIVE_;
00066             try {
00067                 bool done = false;
00068                 while (!done) {
00069                     _TO_ = std::dynamic_pointer_cast<BANK> (tx->load(_to_));
00070                     _FROM_ONE_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_one_));
00071                     _FROM_TWO_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_two_));
00072                     _FROM_THREE_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_three_));
00073                     _FROM_FOUR_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_four_));
00074                     _FROM_FIVE_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_five_));
00075                     _TO_->SetBalance(_TO_->GetBalance() + (_amount * 5));
00076                     _FROM_ONE_->SetBalance(_FROM_ONE_->GetBalance() - _amount);
00077                     _FROM_TWO_->SetBalance(_FROM_TWO_->GetBalance() - _amount);
00078                     _FROM_THREE_->SetBalance(_FROM_THREE_->GetBalance() - _amount);
00079                     _FROM_FOUR_->SetBalance(_FROM_FOUR_->GetBalance() - _amount);
00080                     _FROM_FIVE_->SetBalance(_FROM_FIVE_->GetBalance() - _amount);
00081                     _TO_OSTM = std::dynamic_pointer_cast<OSTM> (_TO_);
00082                     _FROM_ONE_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_ONE_);
00083                     _FROM_TWO_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_TWO_);
00084                     _FROM_THREE_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_THREE_);
00085                     _FROM_FOUR_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_FOUR_);
00086                     _FROM_FIVE_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_FIVE_);
00087
00088                 }
00089             }
00090             catch (...) {
00091                 // ...
00092             }
00093         }

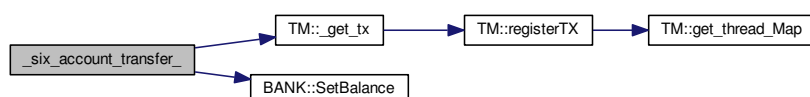
```

```

00101         tx->store(_TO_OSTM);
00102         tx->store(_FROM_ONE_OSTM);
00103         tx->store(_FROM_TWO_OSTM);
00104         tx->store(_FROM_THREE_OSTM);
00105         tx->store(_FROM_FOUR_OSTM);
00106         tx->store(_FROM_FIVE_OSTM);
00110         done = tx->commit();
00111     }
00112     } catch (std::runtime_error& e) {
00113         std::cout << e.what() << std::endl;
00114     }
00115 }

```

Here is the call graph for this function:



6.33.1.6 `void _two_account_transfer_ (std::shared_ptr< OSTM > _to_, std::shared_ptr< OSTM > _from_, TM & _tm, double _amount)`

two_account_transfer function, takes two `std::shared_ptr<OSTM>` pointer, the Transaction manager, and the amount to use in the transaction and transfer the `_amount` value from one account to the another account

Parameters

<code>std::shared_ptr<TX></code>	tx, Transaction Object
<code>std::shared_ptr<BANK></code>	type, <i>TO_BANK</i> & <i>FROM_BANK</i>
<code>std::shared_ptr<OSTM></code>	type, <i>TO_OSTM</i> & <i>FROM_OSTM</i>

Register the two single account

Declare required pointers

From `std::shared_ptr<OSTM>` to `std::shared_ptr<BANK>` to access the virtual methods

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

NESTED TRANSACTION

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

Commit changes

Commit changes

Definition at line 123 of file [main.cpp](#).

References [TM::_get_tx\(\)](#), and [BANK::SetBalance\(\)](#).

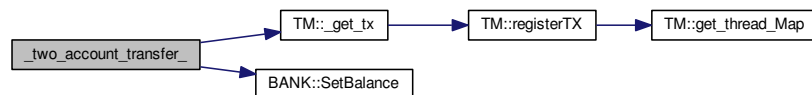
Referenced by [_nesting_\(\)](#).


```

00123 {
00124     std::shared_ptr<TX> tx = _tm._get_tx();
00128     tx->_register(_to_);
00129     tx->_register(_from_);
00133     std::shared_ptr<BANK> _TO_BANK_, _FROM_BANK_;
00134     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00135
00136     bool done = false;
00137     try {
00138         while (!done) {
00142             _TO_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_to_));
00143             _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_));
00147             _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00148             _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00152             _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00153             _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00157             tx->store(_TO_OSTM_);
00158             tx->store(_FROM_OSTM_);
00159
00163             std::shared_ptr<TX> txTwo = _tm._get_tx();
00164
00165             bool nestedDone = false;
00166             while (!nestedDone) {
00167                 _TO_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_to_));
00168                 _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_from_));
00172                 _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00173                 _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00177                 _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00178                 _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00182                 txTwo->store(_TO_OSTM_);
00183                 txTwo->store(_FROM_OSTM_);
00187                 nestedDone = txTwo->commit();
00188             }
00192             done = tx->commit();
00193         }
00194     } catch (std::runtime_error& e) {
00195         std::cout << e.what() << std::endl;
00196     }
00197 }

```

Here is the call graph for this function:



6.33.1.7 void _warehouse_transfer_ (std::shared_ptr< OSTM > _to_, std::shared_ptr< OSTM > _from_, TM & _tm, double _amount)

warehouse_transfer function, takes two std::shared_ptr<OSTM> pointer, the Transaction manager, and the amount to use in the transaction and transfer the _amount value from one account to the another account

Parameters

<i>std::shared_ptr<TX></i>	tx, Transaction Object
<i>std::shared_ptr< WAREHOUSE></i>	type, <i>TO_SHOP</i> & <i>FROM_DIST</i>
<i>std::shared_ptr<OSTM></i>	type, <i>TO_OSTM</i> & <i>FROM_OSTM</i>

Register the two single account

Declare required pointers

From `std::shared_ptr<OSTM>` to `std::shared_ptr<BANK>` to access the virtual methods

Make changes with the objects

From `std::shared_ptr<BANK>` to `std::shared_ptr<OSTM>` to store the memory spaces

Store changes

Commit changes

Definition at line 358 of file `main.cpp`.

References `TM::get_tx()`, and `WAREHOUSE::SetNumber_of_nokia()`.

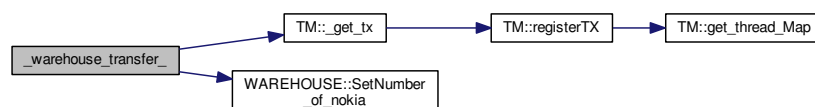
Referenced by `_complex_warehouse_transfer_()`, and `_nested_warehouse_transfer_()`.

```

00358
00359     std::shared_ptr<TX> tx = _tm._get_tx();
00363     tx->_register(_to_);
00364     tx->_register(_from_);
00368     std::shared_ptr<WAREHOUSE> _TO_SHOP_, _FROM_DIST_;
00369     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00370
00371     bool done = false;
00372     try {
00373         while (!done) {
00377             _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_));
00378             _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_from_));
00382             _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00383             _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - _amount);
00384
00385             _TO_SHOP_->SetNumber_of_samsung(_TO_SHOP_->GetNumber_of_samsung() + _amount);
00386             _FROM_DIST_->SetNumber_of_samsung(_FROM_DIST_->GetNumber_of_samsung() - _amount);
00387
00388             _TO_SHOP_->SetNumber_of_iphones(_TO_SHOP_->GetNumber_of_iphones() + _amount);
00389             _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - _amount);
00390
00391             _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00392             _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - _amount);
00396             _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00397             _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00401             tx->store(_TO_OSTM_);
00402             tx->store(_FROM_OSTM_);
00406             done = tx->commit();
00407         }
00408     } catch (std::runtime_error& e) {
00409         std::cout << e.what() << std::endl;
00410     }
00411 }

```

Here is the call graph for this function:



6.33.1.8 int main (void)

main method to run test Get the Transaction Manager

```
TM& tm = TM::Instance();
```

Create vector to store std::shared_ptr<OSTM> pointers. All object will have unique ID by default

```
std::vector<std::shared_ptr<OSTM>> _customer_vec(vector_number);
std::vector<std::shared_ptr<OSTM>> _warehouse_vec(vector_number);
```

Create objects type of **BANK**. All object will have unique ID by default

```
std::shared_ptr<OSTM> aib_ptr = new AIB(100, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny");
std::shared_ptr<OSTM> boi_ptr = new BOI(200, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny");
std::shared_ptr<OSTM> boa_ptr = new BOA(300, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny");
std::shared_ptr<OSTM> swplc_ptr = new SWBPLC(400, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny");
std::shared_ptr<OSTM> ulster_ptr = new ULSTER(500, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny");
std::shared_ptr<OSTM> unbl_ptr = new UNBL(600, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny");
```

Create objects type of **WAREHOUSE**. All object will have unique ID by default

```
std::shared_ptr<OSTM> w_dist = new CARPHONE_WAREHOUSE();
std::shared_ptr<OSTM> c_shop = new CARLOW_W();
std::shared_ptr<OSTM> k_shop = new KILKENNY_W();
std::shared_ptr<OSTM> t_shop = new TALLAGH_W();
std::shared_ptr<OSTM> d_shop = new DUNDALK_W();
std::shared_ptr<OSTM> s_shop = new SLIGO_W();
```

Create vector of std::shared_ptr<OSTM> **BANK** pointers

vector_number is 100 at the moment

```
for(int i=0;i<vector_number;++i)
```

Create vector of std::shared_ptr<OSTM> **WAREHOUSE** pointers

vector_number is 100 at the moment

```
for(int i=0;i<vector_number;++i)
```

Display **WAREHOUSE** objects before transaction

```
w_dist->toString();
c_shop->toString();
k_shop->toString();
t_shop->toString();
d_shop->toString();
s_shop->toString();
```

Display **BANK** objects before transaction

```
aib_ptr->toString();
boi_ptr->toString();
boa_ptr->toString();
swplc_ptr->toString();
ulster_ptr->toString();
unbl_ptr->toString();
```

Parameters

<i>transferAmount</i>	in the transaction, control the value in the transaction between objetcs
-----------------------	--

Parameters

<i>threadArraySize</i>	control number of threads The logic in the IF ELSE statement distribute the threads between three different thread creating option. If the threadArraySize is divisible with three, the threads will be distributed between function. However, you can creates any number of threads, but to follow the correct output should increase the IF ELSE statement to distribute the threads in equal number.
------------------------	---

Creating threadsⁿ -> threadArraySize
for (int i = 0; i < threadArraySize; ++i)

TEST 1 : Nested transaction Test

```
thArray[i] = std::thread(nesting, aib_ptr, boi_ptr, std::ref(tm), transferAmount);
```

TEST 2 : Three different type of function call where the objects are participating in multiple type of transactions

```
thArray[i] = std::thread(two_account_transfer, aib_ptr, boi_ptr, std::ref(tm), transferAmount);
thArray[i] = std::thread(six_account_transfer, boi_ptr, boa_ptr, swplc_ptr, ulster_ptr, aib_ptr, unbl_ptr, std::ref(tm),
transferAmount)
thArray[i] = std::thread(complex_transfer, aib_ptr, boi_ptr, std::ref(_customer_vec), std::ref(tm), transferAmount);
```

TEST 3 : Testing [WAREHOUSE](#) type pointers within transactions

```
thArray[i] = std::thread(phone_transfer, c_shop, w_dist, std::ref(tm), transferAmount);
```

TEST 4 : Testing [WAREHOUSE](#) type pointers within nested transactions

```
thArray[i] = std::thread(nested_warehouse_transfer, c_shop, d_shop, k_shop, w_dist, std::ref(tm), transferAmount);
```

TEST 5 : Testing [WAREHOUSE](#) type pointers within mixed and nested transactions

```
thArray[i] = std::thread(warehouse_transfer, c_shop, w_dist, std::ref(tm), transferAmount);
thArray[i] = std::thread(nested_warehouse_transfer, c_shop, d_shop, k_shop, w_dist, std::ref(tm), transferAmount);
thArray[i] = std::thread(complex_warehouse_transfer, d_shop, c_shop, std::ref(_warehouse_vec), w_dist, std::ref(tm), transferAmount);
```

Display objects after all transactions are finished

Uncomment the required corresponding TEST to display results

Extra tx to call and display ROLLBACK value

```
std::shared_ptr<TX> tx = tm._get_tx();
```

Display the number of ROLLBACK by all the threads

```
std::cout << "Rollback counter is : " << tx->getTest_counter() << std::endl;
```

Display object from vector

Clean up Transaction Manager from all main process associated transactions

```
tm._TX_EXIT();
```

Display all Transactions associated with the main process. It should be empty after _TX_EXIT() function call!!!

```
tm.print_all();
```

Definition at line 649 of file [main.cpp](#).

References [TM::_get_tx\(\)](#), [_nesting\(\)](#), [TM::_TX_EXIT\(\)](#), [TM::Instance\(\)](#), [TM::print_all\(\)](#), and [vector_number](#).

```

00649         {
00654             TM& tm = TM::Instance();
00655
00661
00662             std::vector<std::shared_ptr<OSTM>>_customer_vec; //(vector_number);
00663             std::vector<std::shared_ptr<OSTM>>_warehouse_vec; //(vector_number);
00664
00674             std::shared_ptr<OSTM> aib_ptr(new AIB(100, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny")
);
00675             std::shared_ptr<OSTM> boi_ptr(new BOI(200, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny")
);
00676             std::shared_ptr<OSTM> boa_ptr(new BOA(300, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny")
);
00677             std::shared_ptr<OSTM> swplc_ptr(new SWBPLC(400, 500, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00678             std::shared_ptr<OSTM> ulster_ptr(new ULSTER(500, 500, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00679             std::shared_ptr<OSTM> unbl_ptr(new UNBL(600, 500, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00680
00690
00691             std::shared_ptr<OSTM> w_dist(new CARPHONE_WAREHOUSE());
00692             std::shared_ptr<OSTM> c_shop(new CARLOW_W());
00693             std::shared_ptr<OSTM> k_shop(new KILKENNY_W());
00694             std::shared_ptr<OSTM> t_shop(new TALLAGH_W());
00695             std::shared_ptr<OSTM> d_shop(new DUNDALK_W());
00696             std::shared_ptr<OSTM> s_shop(new SLIGO_W());
00697
00703             for (int i = 0; i < vector_number; ++i) {
00704                 if (i % 5 == 0) {
00705                     std::shared_ptr<OSTM> sharedptr(new CARLOW_W());
00706                     _warehouse_vec.push_back(std::move(sharedptr));
00707                 } else if (i % 4 == 0) {
00708                     std::shared_ptr<OSTM> sharedptr(new KILKENNY_W());
00709                     _warehouse_vec.push_back(std::move(sharedptr));
00710                 } else if (i % 3 == 0) {
00711                     std::shared_ptr<OSTM> sharedptr(new TALLAGH_W());
00712                     _warehouse_vec.push_back(std::move(sharedptr));
00713                 } else if (i % 2 == 0) {
00714                     std::shared_ptr<OSTM> sharedptr(new DUNDALK_W());
00715                     _warehouse_vec.push_back(std::move(sharedptr));
00716                 } else if (i % 1 == 0) {
00717                     std::shared_ptr<OSTM> sharedptr(new SLIGO_W());
00718                     _warehouse_vec.push_back(std::move(sharedptr));
00719                 }
00720             }
00721
00727             for (int i = 0; i < vector_number; ++i) {
00728                 if (i % 6 == 0) {
00729                     std::shared_ptr<OSTM> sharedptr(new AIB(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00730                     _customer_vec.push_back(std::move(sharedptr));
00731                 } else if (i % 5 == 0) {
00732                     std::shared_ptr<OSTM> sharedptr(new BOI(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00733                     _customer_vec.push_back(std::move(sharedptr));
00734                 } else if (i % 4 == 0) {
00735                     std::shared_ptr<OSTM> sharedptr(new BOA(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00736                     _customer_vec.push_back(std::move(sharedptr));
00737                 } else if (i % 3 == 0) {
00738                     std::shared_ptr<OSTM> sharedptr(new SWBPLC(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00739                     _customer_vec.push_back(std::move(sharedptr));
00740                 } else if (i % 2 == 0) {
00741                     std::shared_ptr<OSTM> sharedptr(new ULSTER(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00742                     _customer_vec.push_back(std::move(sharedptr));
00743                 } else if (i % 1 == 0) {
00744                     std::shared_ptr<OSTM> sharedptr(new UNBL(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00745                     _customer_vec.push_back(std::move(sharedptr));
00746                 }
00747             }
00748
00758             // w_dist->toString();
00759             // c_shop->toString();
00760             // k_shop->toString();
00761             // t_shop->toString();
00762             // d_shop->toString();
00763             // s_shop->toString();
00764
00774
00775             /*
00776             * TEST 1 : object requirements
00777
00777             */

```



```

00883         //          thArray[i] = std::thread(_warehouse_transfer_, t_shop, w_dist, std::ref(tm),
transferAmount);
00884
00889         //          if (i % 3 == 0)
00890         //          thArray[i] = std::thread(_nested_warehouse_transfer_, c_shop, d_shop, k_shop, w_dist,
std::ref(tm), transferAmount);
00891         //          else if (i % 2 == 0)
00892         //          thArray[i] = std::thread(_nested_warehouse_transfer_, k_shop, s_shop, t_shop, w_dist,
std::ref(tm), transferAmount);
00893         //          else if (i % 1 == 0)
00894         //          thArray[i] = std::thread(_nested_warehouse_transfer_, t_shop, c_shop, s_shop, w_dist,
std::ref(tm), transferAmount);
00895
00903
00904         //          if (i % 3 == 0)
00905         //          thArray[i] = std::thread(_warehouse_transfer_, c_shop, w_dist, std::ref(tm),
transferAmount);
00906         //          else if (i % 2 == 0)
00907         //          thArray[i] = std::thread(_nested_warehouse_transfer_, k_shop, s_shop, t_shop, w_dist,
std::ref(tm), transferAmount);
00908         //          else if (i % 1 == 0)
00909         //          thArray[i] = std::thread(_complex_warehouse_transfer_, d_shop, s_shop, c_shop,
std::ref(_warehouse_vec), w_dist, std::ref(tm), transferAmount);
00910
00911
00912     }
00913     /*
00914     * Join threads^n -> threadArraySize<br>
00915
00916     * thArray[i].join();
00917
00918     */
00917     for (int i = 0; i < threadArraySize; ++i) {
00918         thArray[i].join();
00919     }
00920
00921
00922     std::cout << "\nMain process print " << std::endl;
00927
00928     /*
00929     * TEST 1 : object requirements
00930
00931     */
00931     aib_ptr->toString();
00932     boi_ptr->toString();
00933
00934     /*
00935     * TEST 2 : object requirements
00936
00937     */
00937     // aib_ptr->toString();
00938     // boi_ptr->toString();
00939     // boa_ptr->toString();
00940     // swplc_ptr->toString();
00941     // ulster_ptr->toString();
00942     // unbl_ptr->toString();
00943     // for(int i=0; i<vector_number; ++i){
00944     //     _customer_vec[i]->toString();
00945     // }
00946
00947     /*
00948     * TEST 3 : object requirements
00949
00950     */
00950     // w_dist->toString();
00951     // c_shop->toString();
00952     // k_shop->toString();
00953     // t_shop->toString();
00954
00955     /*
00956     * TEST 4 : objects requirements
00957
00958     */
00958     // w_dist->toString();
00959     // c_shop->toString();
00960     // k_shop->toString();
00961     // t_shop->toString();
00962     // d_shop->toString();
00963     // s_shop->toString();
00964
00965     /*
00966     * TEST 5 : objects requirements
00967
00968     */
00968     // w_dist->toString();
00969     // c_shop->toString();
00970     // k_shop->toString();

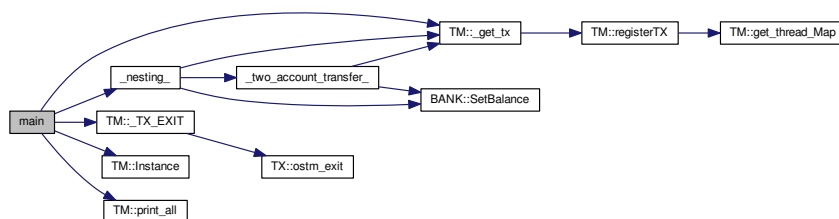
```

```

00971 //      t_shop->toString();
00972 //      d_shop->toString();
00973 //      s_shop->toString();
00974
00975 //      for(auto&& elem: _warehouse_vec){
00976 //          elem->toString(); // virtual dispatch
00977 //      }
00978 //
00979
00980 /* TEST 5 FINISH */
00981
00982
00983 std::cout << "\nMAIN PROCESS EXIT !!!! " << std::endl;
00984 std::shared_ptr<TX> tx = tm._get_tx();
00985
00986
00987 std::cout << "Rollback counter is : " << tx->getTest_counter() << std::endl;
00988 //      std::cout << "[vector_number]" << std::endl;
00989 //      for (int i = 0; i < vector_number; ++i) {
00990 //          //_customer_vec[i]->toString();
00991 //          auto&& os = _customer_vec.at(i);
00992 //          os->toString();
00993 //      }
00994 //      std::cout << "[_warehouse_vec]" << std::endl;
00995 //      for(auto&& elem: _warehouse_vec){
00996 //          elem->toString(); // virtual dispatch
00997 //      }
00998 //      //_customer_vec[10]->toString();
00999
01000 tm._TX_EXIT();
01001 std::cout << "\nPRINT ALL FROM TM !!!! SHOULD BE EMPTY AFTER _TX_EXIT() !" << std::endl;
01002 tm.print_all();
01003
01004 return 0;
01005 }

```

Here is the call graph for this function:



6.33.2 Variable Documentation

6.33.2.1 int vector_number = 600 [static]

Parameters

vector_number	control the size of the vector to store std::shared_ptr<OSTM> pointer
----------------------	---

Definition at line 43 of file [main.cpp](#).

Referenced by [main\(\)](#).

6.34 main.cpp

```
00001 /*
```



```

00002  * To change this license header, choose License Headers in Project Properties.
00003  * To change this template file, choose Tools | Templates
00004  * and open the template in the editor.
00005  */
00006
00007  /*
00008  * File:    main.cpp
00009  * Author:  zoltan
00010  *
00011  * Created on November 27, 2017, 9:26 PM
00012  */
00013
00014 #include <cstdlib>
00015 #include <iostream>
00016 #include <thread>
00017 // #include <unistd.h> // used for pid_t
00018
00019 // STM library requirement
00020 #include "TM.h"
00021 #include "AIB.h" // Bank Account
00022 #include "BOI.h" // Bank Account
00023 #include "BOA.h" // Bank Account
00024 #include "SWBPLC.h" // Bank Account
00025 #include "ULSTER.h" // Bank Account
00026 #include "UNBL.h" // Bank Account
00027 #include "WAREHOUSE.h" // WAREHOUSE
00028 #include "CARPHONE_WAREHOUSE.h" // WAREHOUSE
00029 #include "CARLOW_W.h" // WAREHOUSE
00030 #include "KILKENNY_W.h" // WAREHOUSE
00031 #include "TALLAGH_W.h" // WAREHOUSE
00032 #include "DUNDALK_W.h" // WAREHOUSE
00033 #include "SLIGO_W.h" // WAREHOUSE
00034 #include <mutex>
00035 #include <memory>
00036 #include <condition_variable>
00037 #include <vector>
00038
00039
00043 static int vector_number = 600;
00044
00051 void _six_account_transfer(std::shared_ptr<OSTM> _to_, std::shared_ptr<OSTM>
    _from_one_, std::shared_ptr<OSTM> _from_two_, std::shared_ptr<OSTM> _from_three_, std::shared_ptr<OSTM> _from_four_
    , std::shared_ptr<OSTM> _from_five_, TM& _tm, double _amount) {
00052     std::shared_ptr<TX> tx = _tm._get_tx();
00056     tx->_register(_to_);
00057     tx->_register(_from_one_);
00058     tx->_register(_from_two_);
00059     tx->_register(_from_three_);
00060     tx->_register(_from_four_);
00061     tx->_register(_from_five_);
00062
00066     std::shared_ptr<OSTM> _TO_OSTM, _FROM_ONE_OSTM, _FROM_TWO_OSTM, _FROM_THREE_OSTM, _FROM_FOUR_OSTM,
    _FROM_FIVE_OSTM;
00067     std::shared_ptr<BANK> _TO_, _FROM_ONE_, _FROM_TWO_, _FROM_THREE_, _FROM_FOUR_, _FROM_FIVE_;
00068     try {
00069         bool done = false;
00070         while (!done) {
00074             _TO_ = std::dynamic_pointer_cast<BANK> (tx->load(_to_));
00075             _FROM_ONE_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_one_));
00076             _FROM_TWO_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_two_));
00077             _FROM_THREE_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_three_));
00078             _FROM_FOUR_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_four_));
00079             _FROM_FIVE_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_five_));
00083             _TO_->SetBalance(_TO_->GetBalance() + (_amount * 5));
00084             _FROM_ONE_->SetBalance(_FROM_ONE_->GetBalance() - _amount);
00085             _FROM_TWO_->SetBalance(_FROM_TWO_->GetBalance() - _amount);
00086             _FROM_THREE_->SetBalance(_FROM_THREE_->GetBalance() - _amount);
00087             _FROM_FOUR_->SetBalance(_FROM_FOUR_->GetBalance() - _amount);
00088             _FROM_FIVE_->SetBalance(_FROM_FIVE_->GetBalance() - _amount);
00092             _TO_OSTM = std::dynamic_pointer_cast<OSTM> (_TO_);
00093             _FROM_ONE_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_ONE_);
00094             _FROM_TWO_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_TWO_);
00095             _FROM_THREE_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_THREE_);
00096             _FROM_FOUR_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_FOUR_);
00097             _FROM_FIVE_OSTM = std::dynamic_pointer_cast<OSTM> (_FROM_FIVE_);
00101             tx->store(_TO_OSTM);
00102             tx->store(_FROM_ONE_OSTM);
00103             tx->store(_FROM_TWO_OSTM);
00104             tx->store(_FROM_THREE_OSTM);
00105             tx->store(_FROM_FOUR_OSTM);
00106             tx->store(_FROM_FIVE_OSTM);
00110             done = tx->commit();
00111         }
00112     } catch (std::runtime_error& e) {
00113         std::cout << e.what() << std::endl;
00114     }
00115 }

```

```

00116
00123 void _two_account_transfer(std::shared_ptr<OSTM> _to_, std::shared_ptr<OSTM> _from_,
    TM& _tm, double _amount) {
00124     std::shared_ptr<TX> tx = _tm._get_tx();
00128     tx->_register(_to_);
00129     tx->_register(_from_);
00133     std::shared_ptr<BANK> _TO_BANK_, _FROM_BANK_;
00134     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00135
00136     bool done = false;
00137     try {
00138         while (!done) {
00142             _TO_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_to_));
00143             _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_));
00147             _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00148             _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00152             _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00153             _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00157             tx->store(_TO_OSTM_);
00158             tx->store(_FROM_OSTM_);
00159
00163             std::shared_ptr<TX> txTwo = _tm._get_tx();
00164
00165             bool nestedDone = false;
00166             while (!nestedDone) {
00167                 _TO_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_to_));
00168                 _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_from_));
00172                 _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00173                 _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00177                 _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00178                 _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00182                 txTwo->store(_TO_OSTM_);
00183                 txTwo->store(_FROM_OSTM_);
00187                 nestedDone = txTwo->commit();
00188             }
00192             done = tx->commit();
00193         }
00194     } catch (std::runtime_error& e) {
00195         std::cout << e.what() << std::endl;
00196     }
00197 }
00198
00206 void _nesting_(std::shared_ptr<OSTM> _to_, std::shared_ptr<OSTM> _from_,
    TM& _tm, double _amount) {
00207     std::shared_ptr<TX> tx = _tm._get_tx();
00211     tx->_register(_to_);
00212     tx->_register(_from_);
00216     std::shared_ptr<BANK> _TO_BANK_, _FROM_BANK_;
00217     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00218
00219
00220     bool done = false;
00221     try {
00222         while (!done) {
00226             _TO_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_to_));
00227             _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_));
00231             _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00232             _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00236             _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00237             _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00241             tx->store(_TO_OSTM_);
00242             tx->store(_FROM_OSTM_);
00243
00247             std::shared_ptr<TX> txTwo = _tm._get_tx();
00248
00249             bool nestedDone = false;
00250             while (!nestedDone) {
00251                 _TO_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_to_));
00252                 _FROM_BANK_ = std::dynamic_pointer_cast<BANK> (txTwo->load(_from_));
00256                 _TO_BANK_->SetBalance(_TO_BANK_->GetBalance() + _amount);
00257                 _FROM_BANK_->SetBalance(_FROM_BANK_->GetBalance() - _amount);
00261                 _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_BANK_);
00262                 _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_BANK_);
00266                 txTwo->store(_TO_OSTM_);
00267                 txTwo->store(_FROM_OSTM_);
00272                 _two_account_transfer(_to_, _from_, _tm, _amount);
00273
00274                 nestedDone = txTwo->commit();
00275             }
00276
00280             done = tx->commit();
00281         }
00282     } catch (std::runtime_error& e) {
00283         std::cout << e.what() << std::endl;
00284     }
00285 }
00286

```

```

00294 void _complex_transfer_(std::shared_ptr<OSTM> _from_, std::shared_ptr<OSTM> _from_two_,
std::vector<std::shared_ptr<OSTM>> _customer_vec, TM& _tm, double _amount) {
00295     std::shared_ptr<TX> tx = _tm._get_tx();
00299     tx->_register(_from_);
00300     tx->_register(_from_two_);
00304     std::shared_ptr<OSTM> _FROM_OSTM_ONE_, _FROM_OSTM_TWO_, _TO_OSTM_;
00305     std::shared_ptr<BANK> _FROM_, _FROM_TWO_, _TO_;
00306
00307     bool done = false;
00308     try {
00309         while (!done) {
00310             // for (int i = 0; i < vector_number; ++i) {
00311             for (auto&& obj : _customer_vec) {
00312                 // auto&& obj = _customer_vec.at(i);
00315                 tx->_register(obj);
00316                 _FROM_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_));
00320                 _FROM_TWO_ = std::dynamic_pointer_cast<BANK> (tx->load(_from_two_));
00321                 _TO_ = std::dynamic_pointer_cast<BANK> (tx->load(obj));
00322                 _FROM_->SetBalance(_FROM_->GetBalance() - _amount);
00326                 _FROM_TWO_->SetBalance(_FROM_TWO_->GetBalance() - _amount);
00327                 _TO_->SetBalance(_TO_->GetBalance() + (_amount * 2));
00328                 _FROM_OSTM_ONE_ = std::dynamic_pointer_cast<OSTM> (_FROM_);
00332                 _FROM_OSTM_TWO_ = std::dynamic_pointer_cast<OSTM> (_FROM_TWO_);
00333                 _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_);
00334                 tx->store(_FROM_OSTM_ONE_);
00338                 tx->store(_FROM_OSTM_TWO_);
00339                 tx->store(_TO_OSTM_);
00340             }
00341             done = tx->commit();
00345         }
00346     } catch (std::runtime_error& e) {
00347         std::cout << e.what() << std::endl;
00348     }
00349 }
00350 }
00351
00358 void _warehouse_transfer_(std::shared_ptr<OSTM> _to_, std::shared_ptr<OSTM> _from_,
TM& _tm, double _amount) {
00359     std::shared_ptr<TX> tx = _tm._get_tx();
00363     tx->_register(_to_);
00364     tx->_register(_from_);
00368     std::shared_ptr<WAREHOUSE> _TO_SHOP_, _FROM_DIST_;
00369     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00370
00371     bool done = false;
00372     try {
00373         while (!done) {
00377             _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_));
00378             _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_from_));
00382             _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00383             _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - _amount);
00384
00385             _TO_SHOP_->SetNumber_of_samsung(_TO_SHOP_->GetNumber_of_samsung() + _amount);
00386             _FROM_DIST_->SetNumber_of_samsung(_FROM_DIST_->GetNumber_of_samsung() - _amount);
00387
00388             _TO_SHOP_->SetNumber_of_iphones(_TO_SHOP_->GetNumber_of_iphones() + _amount);
00389             _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - _amount);
00390
00391             _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00392             _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - _amount);
00396             _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00397             _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00401             tx->store(_TO_OSTM_);
00402             tx->store(_FROM_OSTM_);
00406             done = tx->commit();
00407         }
00408     } catch (std::runtime_error& e) {
00409         std::cout << e.what() << std::endl;
00410     }
00411 }
00412
00419 void _nested_warehouse_transfer_(std::shared_ptr<OSTM> _to_,
std::shared_ptr<OSTM> _to_two_, std::shared_ptr<OSTM> _to_three_, std::shared_ptr<OSTM> _from_, TM& _tm, double _amount) {
00420     std::shared_ptr<TX> tx = _tm._get_tx();
00424     tx->_register(_to_);
00425     tx->_register(_to_two_);
00426     tx->_register(_to_three_);
00427     tx->_register(_from_);
00431     std::shared_ptr<WAREHOUSE> _TO_SHOP_, _FROM_DIST_;
00432     std::shared_ptr<OSTM> _TO_OSTM_, _FROM_OSTM_;
00433
00434     bool done = false;
00435     try {
00436         while (!done) {
00440             _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_));
00441             _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_from_));
00445             _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00446             _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - _amount);

```

```

00447
00448     _TO_SHOP_>SetNumber_of_samsung(_TO_SHOP_>GetNumber_of_samsung() + _amount);
00449     _FROM_DIST_>SetNumber_of_samsung(_FROM_DIST_>GetNumber_of_samsung() - _amount);
00450
00451     _TO_SHOP_>SetNumber_of_iphones(_TO_SHOP_>GetNumber_of_iphones() + _amount);
00452     _FROM_DIST_>SetNumber_of_iphones(_FROM_DIST_>GetNumber_of_iphones() - _amount);
00453
00454     _TO_SHOP_>SetNumber_of_sony(_TO_SHOP_>GetNumber_of_sony() + _amount);
00455     _FROM_DIST_>SetNumber_of_sony(_FROM_DIST_>GetNumber_of_sony() - _amount);
00459     _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00460     _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00464     tx->store(_TO_OSTM_);
00465     tx->store(_FROM_OSTM_);
00466
00470     std::shared_ptr<TX> txTwo = _tm._get_tx();
00471     bool nestedDone = false;
00472     while (!nestedDone) {
00473         _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_to_two));
00474         _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_from_));
00478         _TO_SHOP_>SetNumber_of_nokia(_TO_SHOP_>GetNumber_of_nokia() + _amount);
00479         _FROM_DIST_>SetNumber_of_nokia(_FROM_DIST_>GetNumber_of_nokia() - _amount);
00480
00481         _TO_SHOP_>SetNumber_of_samsung(_TO_SHOP_>GetNumber_of_samsung() + _amount);
00482         _FROM_DIST_>SetNumber_of_samsung(_FROM_DIST_>GetNumber_of_samsung() - _amount);
00483
00484         _TO_SHOP_>SetNumber_of_iphones(_TO_SHOP_>GetNumber_of_iphones() + _amount);
00485         _FROM_DIST_>SetNumber_of_iphones(_FROM_DIST_>GetNumber_of_iphones() - _amount);
00486
00487         _TO_SHOP_>SetNumber_of_sony(_TO_SHOP_>GetNumber_of_sony() + _amount);
00488         _FROM_DIST_>SetNumber_of_sony(_FROM_DIST_>GetNumber_of_sony() - _amount);
00492         _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00493         _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00497         txTwo->store(_TO_OSTM_);
00498         txTwo->store(_FROM_OSTM_);
00499
00500         /*
00501          * NESTED TRANSACTION TEST _to_three
00502          */
00503         _warehouse_transfer_(_to_three, _from_, _tm, _amount);
00504
00505         nestedDone = tx->commit();
00506     }
00507     done = tx->commit();
00511 }
00512 } catch (std::runtime_error& e) {
00513     std::cout << e.what() << std::endl;
00514 }
00515 }
00516 }
00517
00518 void _complex_warehouse_transfer_(std::shared_ptr<OSTM> _to_,
std::shared_ptr<OSTM> _to_two, std::shared_ptr<OSTM> _to_three, std::vector<std::shared_ptr<OSTM>> _warehouse_vec,
std::shared_ptr<OSTM> _from_, TM& _tm, double _amount) {
00519     std::shared_ptr<TX> tx = _tm._get_tx();
00523     tx->_register(_to_);
00524     tx->_register(_to_two);
00525     tx->_register(_to_three);
00526     tx->_register(_from_);
00530     std::shared_ptr<WAREHOUSE> _TO_SHOP_, _TO_SHOP_TWO_, _TO_SHOP_VEC_, _FROM_DIST_;
00531     std::shared_ptr<OSTM> _TO_OSTM_, _TO_OSTM_TWO_, _TO_OSTM_VEC_, _FROM_OSTM_;
00532
00533     bool done = false;
00534     try {
00535         while (!done) {
00536
00537             // for (int i = 0; i < vector_number; ++i) {
00538             for (auto&& obj : _warehouse_vec) {
00542                 //auto&& obj = _warehouse_vec.at(i);
00543                 tx->_register(obj);
00547                 _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_));
00548                 _TO_SHOP_TWO_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_to_two));
00549                 _TO_SHOP_VEC_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(obj));
00550                 _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (tx->load(_from_));
00551
00555                 _TO_SHOP_>SetNumber_of_nokia(_TO_SHOP_>GetNumber_of_nokia() + _amount);
00556                 _TO_SHOP_TWO_>SetNumber_of_nokia(_TO_SHOP_TWO_>GetNumber_of_nokia() + _amount);
00557                 _TO_SHOP_VEC_>SetNumber_of_nokia(_TO_SHOP_VEC_>GetNumber_of_nokia() + _amount);
00558                 _FROM_DIST_>SetNumber_of_nokia(_FROM_DIST_>GetNumber_of_nokia() - (_amount * 3));
00559
00560                 _TO_SHOP_>SetNumber_of_samsung(_TO_SHOP_>GetNumber_of_samsung() + _amount);
00561                 _TO_SHOP_TWO_>SetNumber_of_samsung(_TO_SHOP_TWO_>GetNumber_of_samsung() + _amount);
00562                 _TO_SHOP_VEC_>SetNumber_of_samsung(_TO_SHOP_VEC_>GetNumber_of_samsung() + _amount);
00563                 _FROM_DIST_>SetNumber_of_samsung(_FROM_DIST_>GetNumber_of_samsung() - (_amount * 3));
00564
00565                 _TO_SHOP_>SetNumber_of_iphones(_TO_SHOP_>GetNumber_of_iphones() + _amount);
00566                 _TO_SHOP_TWO_>SetNumber_of_iphones(_TO_SHOP_TWO_>GetNumber_of_iphones() + _amount);
00567                 _TO_SHOP_VEC_>SetNumber_of_iphones(_TO_SHOP_VEC_>GetNumber_of_iphones() + _amount);

```

```

00568         _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - (_amount * 3));
00569
00570         _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00571         _TO_SHOP_TWO->SetNumber_of_sony(_TO_SHOP_TWO->GetNumber_of_sony() + _amount);
00572         _TO_SHOP_VEC->SetNumber_of_sony(_TO_SHOP_VEC->GetNumber_of_sony() + _amount);
00573         _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - (_amount * 3));
00574
00575         _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00576         _TO_OSTM_TWO = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_TWO);
00577         _TO_OSTM_VEC = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_VEC);
00578         _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00579         tx->store(_TO_OSTM_);
00580         tx->store(_TO_SHOP_TWO);
00581         tx->store(_TO_SHOP_VEC);
00582         tx->store(_FROM_OSTM_);
00583
00584
00585
00586
00587
00588
00589
00590
00591
00592     }
00593     std::shared_ptr<TX> txTwo = _tm._get_tx();
00594     bool nestedDone = false;
00595     while (!nestedDone) {
00596         _TO_SHOP_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_to_two));
00597         _FROM_DIST_ = std::dynamic_pointer_cast<WAREHOUSE> (txTwo->load(_from));
00598         _TO_SHOP_->SetNumber_of_nokia(_TO_SHOP_->GetNumber_of_nokia() + _amount);
00599         _FROM_DIST_->SetNumber_of_nokia(_FROM_DIST_->GetNumber_of_nokia() - _amount);
00600
00601         _TO_SHOP_->SetNumber_of_samsung(_TO_SHOP_->GetNumber_of_samsung() + _amount);
00602         _FROM_DIST_->SetNumber_of_samsung(_FROM_DIST_->GetNumber_of_samsung() - _amount);
00603
00604         _TO_SHOP_->SetNumber_of_iphones(_TO_SHOP_->GetNumber_of_iphones() + _amount);
00605         _FROM_DIST_->SetNumber_of_iphones(_FROM_DIST_->GetNumber_of_iphones() - _amount);
00606
00607         _TO_SHOP_->SetNumber_of_sony(_TO_SHOP_->GetNumber_of_sony() + _amount);
00608         _FROM_DIST_->SetNumber_of_sony(_FROM_DIST_->GetNumber_of_sony() - _amount);
00609         _TO_OSTM_ = std::dynamic_pointer_cast<OSTM> (_TO_SHOP_);
00610         _FROM_OSTM_ = std::dynamic_pointer_cast<OSTM> (_FROM_DIST_);
00611         txTwo->store(_TO_OSTM_);
00612         txTwo->store(_FROM_OSTM_);
00613
00614         /*
00615          * NESTED TRANSACTION TEST _to_three
00616          */
00617         _warehouse_transfer_(_to_three, _from, _tm, _amount);
00618         _nested_warehouse_transfer_(_to, _to_two, _to_three, _from,
00619         _tm, _amount);
00620
00621         nestedDone = tx->commit();
00622     }
00623     done = tx->commit();
00624 }
00625 }
00626 }
00627 }
00628 }
00629 }
00630
00631 int main(void) {
00632     TM& tm = TM::Instance();
00633
00634     std::vector<std::shared_ptr<OSTM>>_customer_vec; //(vector_number);
00635     std::vector<std::shared_ptr<OSTM>>_warehouse_vec; //(vector_number);
00636
00637     std::shared_ptr<OSTM> aib_ptr(new AIB(100, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny"));
00638 );
00639     std::shared_ptr<OSTM> boi_ptr(new BOI(200, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny"));
00640 );
00641     std::shared_ptr<OSTM> boa_ptr(new BOA(300, 500, "Joe", "Blog", "High street, Kilkenny, Co.Kilkenny"));
00642 );
00643     std::shared_ptr<OSTM> swplc_ptr(new SWBPLC(400, 500, "Joe", "Blog", "High street, Kilkenny,
00644 Co.Kilkenny"));
00645     std::shared_ptr<OSTM> ulster_ptr(new ULSTER(500, 500, "Joe", "Blog", "High street, Kilkenny,
00646 Co.Kilkenny"));
00647     std::shared_ptr<OSTM> unbl_ptr(new UNBL(600, 500, "Joe", "Blog", "High street, Kilkenny,
00648 Co.Kilkenny"));
00649
00650     std::shared_ptr<OSTM> w_dist(new CARPHONE_WAREHOUSE());
00651     std::shared_ptr<OSTM> c_shop(new CARLOW_W());
00652     std::shared_ptr<OSTM> k_shop(new KILKENNY_W());
00653     std::shared_ptr<OSTM> t_shop(new TALLAGH_W());
00654     std::shared_ptr<OSTM> d_shop(new DUNDALK_W());
00655     std::shared_ptr<OSTM> s_shop(new SLIGO_W());
00656
00657     for (int i = 0; i < vector_number; ++i) {
00658         if (i % 5 == 0) {
00659             std::shared_ptr<OSTM> sharedptr(new CARLOW_W());

```

```

00706         _warehouse_vec.push_back(std::move(sharedptr));
00707     } else if (i % 4 == 0) {
00708         std::shared_ptr<OSTM> sharedptr(new KILKENNY_W());
00709         _warehouse_vec.push_back(std::move(sharedptr));
00710     } else if (i % 3 == 0) {
00711         std::shared_ptr<OSTM> sharedptr(new TALLAGH_W());
00712         _warehouse_vec.push_back(std::move(sharedptr));
00713     } else if (i % 2 == 0) {
00714         std::shared_ptr<OSTM> sharedptr(new DUNDALK_W());
00715         _warehouse_vec.push_back(std::move(sharedptr));
00716     } else if (i % 1 == 0) {
00717         std::shared_ptr<OSTM> sharedptr(new SLIGO_W());
00718         _warehouse_vec.push_back(std::move(sharedptr));
00719     }
00720 }
00721
00722 for (int i = 0; i < vector_number; ++i) {
00723     if (i % 6 == 0) {
00724         std::shared_ptr<OSTM> sharedptr(new AIB(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00730         _customer_vec.push_back(std::move(sharedptr));
00731     } else if (i % 5 == 0) {
00732         std::shared_ptr<OSTM> sharedptr(new BOI(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00733         _customer_vec.push_back(std::move(sharedptr));
00734     } else if (i % 4 == 0) {
00735         std::shared_ptr<OSTM> sharedptr(new BOA(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00736         _customer_vec.push_back(std::move(sharedptr));
00737     } else if (i % 3 == 0) {
00738         std::shared_ptr<OSTM> sharedptr(new SWBPLC(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00739         _customer_vec.push_back(std::move(sharedptr));
00740     } else if (i % 2 == 0) {
00741         std::shared_ptr<OSTM> sharedptr(new ULSTER(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00742         _customer_vec.push_back(std::move(sharedptr));
00743     } else if (i % 1 == 0) {
00744         std::shared_ptr<OSTM> sharedptr(new UNBL(i, 50, "Joe", "Blog", "High street, Kilkenny,
Co.Kilkenny"));
00745         _customer_vec.push_back(std::move(sharedptr));
00746     }
00747 }
00748
00758 // w_dist->toString();
00759 // c_shop->toString();
00760 // k_shop->toString();
00761 // t_shop->toString();
00762 // d_shop->toString();
00763 // s_shop->toString();
00764
00775 /*
00776  * TEST 1 : object requirements
00777  */
00778 aib_ptr->toString();
00779 boi_ptr->toString();
00780
00781 /*
00782  * TEST 2 : object requirements
00783  */
00784 // aib_ptr->toString();
00785 // boi_ptr->toString();
00786 // boa_ptr->toString();
00787 // swplc_ptr->toString();
00788 // ulster_ptr->toString();
00789 // unbl_ptr->toString();
00790 // for(int i=0; i<vector_number; ++i){
00791 //     _customer_vec[i]->toString();
00792 // }
00793
00794 /*
00795  * TEST 3 : object requirements
00796  */
00797 // w_dist->toString();
00798 // c_shop->toString();
00799 // k_shop->toString();
00800 // t_shop->toString();
00801
00802 /*
00803  * TEST 4 : objects requirements
00804  */
00805 // w_dist->toString();
00806 // c_shop->toString();
00807 // k_shop->toString();
00808 // t_shop->toString();
00809 // d_shop->toString();
00810 // s_shop->toString();

```

```

00811
00812
00813     /*
00814     * TEST 5 : objects requirements
00815     */
00816     //         w_dist->toString();
00817     //         c_shop->toString();
00818     //         k_shop->toString();
00819     //         t_shop->toString();
00820     //         d_shop->toString();
00821     //         s_shop->toString();
00822
00823     //         for(auto&& elem: _warehouse_vec){
00824     //             elem->toString(); // virtual dispatch
00825     //
00826     //         }
00827
00828
00829
00833     int transferAmount = 1;
00840     int threadArraySize = 99;
00841     std::thread thArray[threadArraySize];
00842
00847     for (int i = 0; i < threadArraySize; ++i) {
00848
00853         if (i % 3 == 0)
00854             thArray[i] = std::thread(_nesting_, aib_ptr, boi_ptr, std::ref(tm), transferAmount);
00855         else if (i % 2 == 0)
00856             thArray[i] = std::thread(_nesting_, aib_ptr, boi_ptr, std::ref(tm), transferAmount);
00857         else if (i % 1 == 0)
00858             thArray[i] = std::thread(_nesting_, aib_ptr, boi_ptr, std::ref(tm), transferAmount);
00859
00866         //         if (i % 3 == 0)
00867         //             thArray[i] = std::thread(_two_account_transfer_, aib_ptr, boi_ptr, std::ref(tm),
transferAmount);
00868         //         else if (i % 2 == 0)
00869         //             thArray[i] = std::thread(_six_account_transfer_, boi_ptr, boa_ptr, swplc_ptr, ulster_ptr,
aib_ptr, unbl_ptr, std::ref(tm), transferAmount);
00870         //         else if (i % 1 == 0)
00871         //             thArray[i] = std::thread(_complex_transfer_, aib_ptr, boi_ptr, std::ref(_customer_vec),
std::ref(tm), transferAmount);
00872
00873
00878         //         if (i % 3 == 0)
00879         //             thArray[i] = std::thread(_warehouse_transfer_, c_shop, w_dist, std::ref(tm),
transferAmount);
00880         //         else if (i % 2 == 0)
00881         //             thArray[i] = std::thread(_warehouse_transfer_, k_shop, w_dist, std::ref(tm),
transferAmount);
00882         //         else if (i % 1 == 0)
00883         //             thArray[i] = std::thread(_warehouse_transfer_, t_shop, w_dist, std::ref(tm),
transferAmount);
00884
00889         //         if (i % 3 == 0)
00890         //             thArray[i] = std::thread(_nested_warehouse_transfer_, c_shop, d_shop, k_shop, w_dist,
std::ref(tm), transferAmount);
00891         //         else if (i % 2 == 0)
00892         //             thArray[i] = std::thread(_nested_warehouse_transfer_, k_shop, s_shop, t_shop, w_dist,
std::ref(tm), transferAmount);
00893         //         else if (i % 1 == 0)
00894         //             thArray[i] = std::thread(_nested_warehouse_transfer_, t_shop, c_shop, s_shop, w_dist,
std::ref(tm), transferAmount);
00895
00904         //         if (i % 3 == 0)
00905         //             thArray[i] = std::thread(_warehouse_transfer_, c_shop, w_dist, std::ref(tm),
transferAmount);
00906         //         else if (i % 2 == 0)
00907         //             thArray[i] = std::thread(_nested_warehouse_transfer_, k_shop, s_shop, t_shop, w_dist,
std::ref(tm), transferAmount);
00908         //         else if (i % 1 == 0)
00909         //             thArray[i] = std::thread(_complex_warehouse_transfer_, d_shop, s_shop, c_shop,
std::ref(_warehouse_vec), w_dist, std::ref(tm), transferAmount);
00910
00911
00912     }
00913     /*
00914     * Join threads^n -> threadArraySize<br>
00915     * thArray[i].join();
00916     */
00917     for (int i = 0; i < threadArraySize; ++i) {
00918         thArray[i].join();
00919     }
00920
00921
00922     std::cout << "\nMain process print " << std::endl;
00928     /*
00929     * TEST 1 : object requirements

```

```

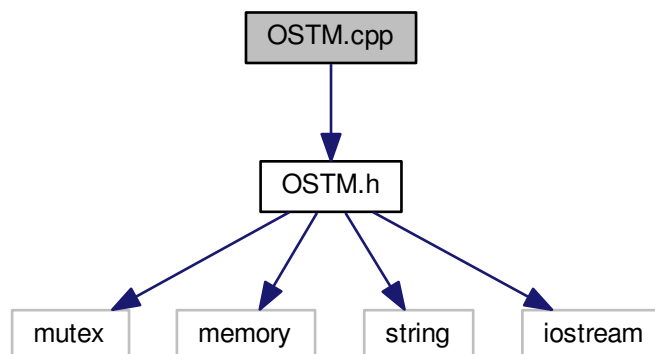
00930      */
00931      aib_ptr->toString();
00932      boi_ptr->toString();
00933
00934      /*
00935       * TEST 2 : object requirements
00936       */
00937      //      aib_ptr->toString();
00938      //      boi_ptr->toString();
00939      //      boa_ptr->toString();
00940      //      swplc_ptr->toString();
00941      //      ulster_ptr->toString();
00942      //      unbl_ptr->toString();
00943      //      for(int i=0; i<vector_number; ++i){
00944      //          _customer_vec[i]->toString();
00945      //      }
00946
00947      /*
00948       * TEST 3 : object requirements
00949       */
00950      //          w_dist->toString();
00951      //          c_shop->toString();
00952      //          k_shop->toString();
00953      //          t_shop->toString();
00954
00955      /*
00956       * TEST 4 : objects requirements
00957       */
00958      //          w_dist->toString();
00959      //          c_shop->toString();
00960      //          k_shop->toString();
00961      //          t_shop->toString();
00962      //          d_shop->toString();
00963      //          s_shop->toString();
00964
00965      /*
00966       * TEST 5 : objects requirements
00967       */
00968      //          w_dist->toString();
00969      //          c_shop->toString();
00970      //          k_shop->toString();
00971      //          t_shop->toString();
00972      //          d_shop->toString();
00973      //          s_shop->toString();
00974
00975      //          for(auto&& elem: _warehouse_vec){
00976      //              elem->toString(); // virtual dispatch
00977      //          }
00978
00979      /* TEST 5 FINISH */
00980
00981
00982
00983      std::cout << "\nMAIN PROCESS EXIT !!!! " << std::endl;
00984      std::shared_ptr<TX> tx = tm._get_tx();
00985
00986      std::cout << "Rollback counter is : " << tx->getTest_counter() << std::endl;
00987      //      std::cout << "[vector_number]" << std::endl;
00988      //      for (int i = 0; i < vector_number; ++i) {
00989      //          _customer_vec[i]->toString();
00990      //          auto&& os = _customer_vec.at(i);
00991      //          os->toString();
00992      //      }
00993      //      std::cout << "[_warehouse_vec]" << std::endl;
00994      //      for(auto&& elem: _warehouse_vec){
00995      //          elem->toString(); // virtual dispatch
00996      //      }
00997      //      _customer_vec[10]->toString();
00998
00999      tm._TX_EXIT();
01000      std::cout << "\nPRINT ALL FROM TM !!!! SHOULD BE EMPTY AFTER _TX_EXIT() !!" << std::endl;
01001      tm.print_all();
01002
01003      return 0;
01004 }

```

6.35 OSTM.cpp File Reference

```
#include "OSTM.h"
```


Include dependency graph for OSTM.cpp:



6.36 OSTM.cpp

```

00001 /*
00002  * File:   OSTM.cpp
00003  * Author: Zoltan Fuzesi
00004  *
00005  * Created on December 18, 2017, 2:09 PM
00006  * OSTM cpp file methods implementations
00007  */
00008
00009 #include "OSTM.h"
00010
00011 int OSTM::global_Unique_ID_Number = 0;
00012
00020 OSTM::OSTM()
00021 {
00022     this->version = ZERO;
00023     this->uniqueID = Get_global_Unique_ID_Number(); //
00024     ++global_Unique_ID_Number;
00025     this->canCommit = true;
00026     this->abort_Transaction = false;
00027 }
00028
00036 OSTM::OSTM(int _version_number_, int _unique_id_)
00037 {
00038     // std::cout << "OSTM COPY CONSTRUCTOR" << global_Unique_ID_Number << std::endl;
00039     this->uniqueID = _unique_id_;
00040     this->version = _version_number_;
00041     this->canCommit = true;
00042     this->abort_Transaction = false;
00043 }
00044
00048 OSTM::~OSTM() {
00049     //std::cout << "[OSTM DELETE]" << std::endl;
00050 }
00056 int OSTM::Get_global_Unique_ID_Number() {
00057     if(global_Unique_ID_Number > 10000000)
00058         global_Unique_ID_Number = 0;
00059     return ++global_Unique_ID_Number;
00060 }
00061
00066 void OSTM::Set_Unique_ID(int uniqueID) {
00067     this->uniqueID = uniqueID;
00068 }
00073 int OSTM::Get_Unique_ID() const
00074 {
00075     return uniqueID;
00076 }
00081 void OSTM::Set_Version(int version)
00082 {

```


6.38 OSTM.h

```

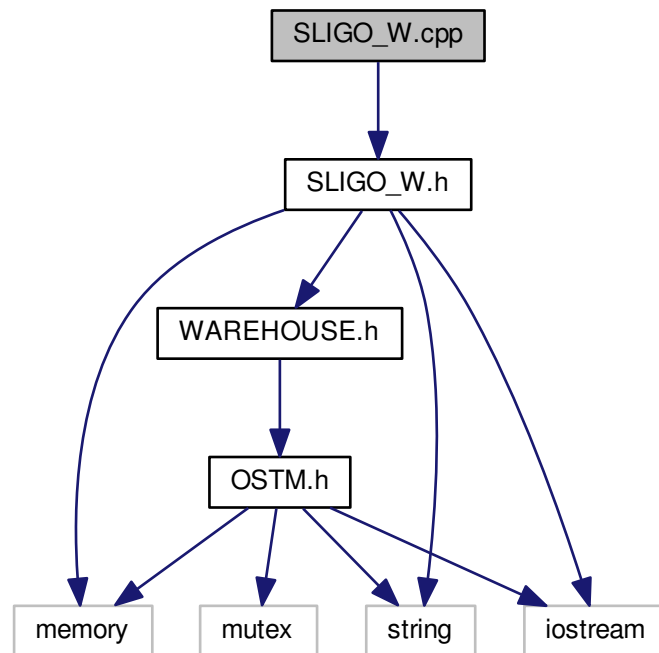
00001 /*
00002  * File:    OSTM.h
00003  * Author:  Zoltan FUzesi
00004  *
00005  * Created on December 18, 2017, 2:09 PM
00006  * OSTM header file fields and methods declarations
00007  */
00008
00009 #ifndef OSTM_H
00010 #define OSTM_H
00011 #include <mutex>
00012 #include <memory>
00013 #include <string>
00014 #include <iostream>
00015 #include <string>
00016
00017 class OSTM {
00018 public:
00019     OSTM();
00020     OSTM(int _version_number_, int _unique_id_);
00021     virtual ~OSTM();
00022     virtual void copy(std::shared_ptr<OSTM> from, std::shared_ptr<OSTM> to){};
00023     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object){}; //std::cout <<
00024     "[OSTM GETBASECOPY]" << std::endl;};
00025     virtual void toString(){};
00026     void Set_Unique_ID(int uniqueID);
00027     int Get_Unique_ID() const;
00028     void Set_Version(int version);
00029     int Get_Version() const;
00030     void increase_VersionNumber();
00031     bool Is_Can_Commit() const;
00032     void Set_Can_Commit(bool canCommit);
00033     void Set_Abort_Transaction(bool abortTransaction);
00034     bool Is_Abort_Transaction() const;
00035     void lock_Mutex();
00036     void unlock_Mutex();
00037     bool is_Locked();
00038
00039 private:
00040     /*
00041      * \brief Object version number
00042      */
00043     int version;
00044     /*
00045      * \brief Object unique identifier
00046      */
00047     int uniqueID;
00048     /*
00049      * \brief Boolean value to check any other thread failed to commit
00050      */
00051     bool canCommit;
00052     /*
00053      * \brief Abort the transaction
00054      */
00055     bool abort_Transaction;
00056     static int global_Unique_ID_Number;
00057     const int ZERO = 0;
00058     std::mutex mutex;
00059     int Get_global_Unique_ID_Number();
00060
00061 };
00062
00063 #endif /* OSTM_H */

```

6.39 SLIGO_W.cpp File Reference

```
#include "SLIGO_W.h"
```

Include dependency graph for SLIGO_W.cpp:



6.40 SLIGO_W.cpp

```

00001
00002 /*
00003  * File:    SLIGO_W.cpp
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #include "SLIGO_W.h"
00011
00012 SLIGO_W::~SLIGO_W() {
00013 }
00014
00015 SLIGO_W::SLIGO_W(const SLIGO_W& orig) {
00016 }
00022 std::shared_ptr<OSTM> SLIGO_W::getBaseCopy(std::shared_ptr<OSTM> object)
00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new SLIGO_W(objTO, object->Get_Version(), object->Get_Unique_ID()));
00027     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00028
00029     return ostm_obj;
00030 }
00035 void SLIGO_W::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00036
00037     std::shared_ptr<SLIGO_W> objTO = std::dynamic_pointer_cast<SLIGO_W>(to);
00038     std::shared_ptr<SLIGO_W> objFROM = std::dynamic_pointer_cast<SLIGO_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
  
```

```

00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
00052 //std::shared_ptr<SLIGO_W> SLIGO_W::_cast(std::shared_ptr<OSTM> _object){
00053 //
00054 //     return static_cast<std::shared_ptr<SLIGO_W>>(_object);
00055 //}
00056 void SLIGO_W::toString()
00057 {
00058     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
    Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
    << this->GetShop_address() << "\nNo. Iphones : " << this->
    GetNumber_of_iphones() << "\nNo. Samsung : " << this->
    GetNumber_of_samsung() << "\nNo. Sony : " << this->
    GetNumber_of_sony() << "\nNo. Huawei : " << this->
    GetNumber_of_huawei() << "\nNo. Nokia : " << this->
    GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
    GetNumber_of_alcatel() << "\nVersion number : " << this->
    Get_Version() << std::endl;
00059 }
00060
00061 void SLIGO_W::SetNumber_of_alcatel(int
    _number_of_alcatel) {
00062     this->_number_of_alcatel = _number_of_alcatel;
00063 }
00064
00065 int SLIGO_W::GetNumber_of_alcatel() {
00066     return _number_of_alcatel;
00067 }
00068
00069 void SLIGO_W::SetNumber_of_nokia(int _number_of_nokia) {
00070     this->_number_of_nokia = _number_of_nokia;
00071 }
00072
00073 int SLIGO_W::GetNumber_of_nokia() {
00074     return _number_of_nokia;
00075 }
00076
00077 void SLIGO_W::SetNumber_of_huawei(int
    _number_of_huawei) {
00078     this->_number_of_huawei = _number_of_huawei;
00079 }
00080
00081 int SLIGO_W::GetNumber_of_huawei() {
00082     return _number_of_huawei;
00083 }
00084
00085 void SLIGO_W::SetNumber_of_sony(int
    _number_of_sony) {
00086     this->_number_of_sony = _number_of_sony;
00087 }
00088
00089 int SLIGO_W::GetNumber_of_sony() {
00090     return _number_of_sony;
00091 }
00092
00093 void SLIGO_W::SetNumber_of_samsung(int
    _number_of_samsung) {
00094     this->_number_of_samsung = _number_of_samsung;
00095 }
00096
00097 int SLIGO_W::GetNumber_of_samsung() {
00098     return _number_of_samsung;
00099 }
00100
00101 void SLIGO_W::SetNumber_of_iphones(int
    _number_of_iphones) {
00102     this->_number_of_iphones = _number_of_iphones;
00103 }
00104
00105 int SLIGO_W::GetNumber_of_iphones() {
00106     return _number_of_iphones;
00107 }
00108
00109 void SLIGO_W::SetShop_name(std::string _shop_name) {
00110     this->_shop_name = _shop_name;
00111 }
00112
00113 std::string SLIGO_W::GetShop_name() {
00114     return _shop_name;
00115 }
00116
00117 }

```

```

00124
00125 void SLIGO_W::SetShop_address(std::string _shop_address) {
00126     this->_shop_address = _shop_address;
00127 }
00128
00129 std::string SLIGO_W::GetShop_address(){
00130     return _shop_address;
00131 }
00132
00133
00134

```

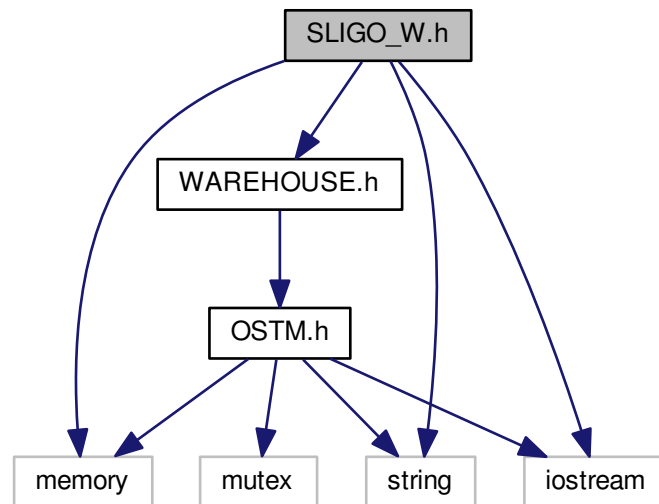
6.41 SLIGO_W.h File Reference

```

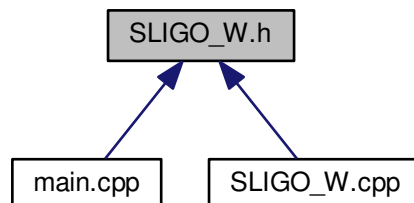
#include "WAREHOUSE.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for SLIGO_W.h:



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



Classes

- class [SLIGO_W](#)

6.42 SLIGO_W.h

```

00001
00002 /*
00003  * File:    SLIGO_W.h
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef SLIGO_W_H
00011 #define SLIGO_W_H
00012 #include "WAREHOUSE.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class SLIGO_W :public WAREHOUSE {
00020 public:
00024     SLIGO_W() : WAREHOUSE() {
00025
00026         this->_shop_address = "Sligo River Street";
00027         this->_shop_name = "SLIGO S_WAREHOUSE";
00028         this->_number_of_iphones = 200;
00029         this->_number_of_samsung = 200;
00030         this->_number_of_sony = 200;
00031         this->_number_of_huawei = 200;
00032         this->_number_of_nokia = 200;
00033         this->_number_of_alcatel = 200;
00034     };
00038     SLIGO_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int
00039     huawei, int nokia, int alcatel): WAREHOUSE() {
00040         /*
00041          * copy over values
00042          */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };
00055     SLIGO_W(std::shared_ptr<WAREHOUSE> obj, int _version, int _unique_id):
00056     WAREHOUSE(_version, _unique_id) {
00057         /*
00058          * copy over values
00059          */
00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }
00071     SLIGO_W(const SLIGO_W& orig);
00075     SLIGO_W operator=(const SLIGO_W& orig) {};
00079     virtual ~SLIGO_W();
00080
00081     /*
00082      * Implement OSTM virtual methods
00083      */
00084     // virtual std::shared_ptr<SLIGO_W> _cast(std::shared_ptr<OSTM> _object);
00085     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00086     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00087     virtual void toString();
00088     /*
00089      * Implement Warehouse methods
00090      */
00091     virtual void SetNumber_of_alcatel(int _number_of_alcatel);
00092     virtual int GetNumber_of_alcatel();
00093     virtual void SetNumber_of_nokia(int _number_of_nokia);
00094     virtual int GetNumber_of_nokia();
00095     virtual void SetNumber_of_huawei(int _number_of_huawei);

```

```

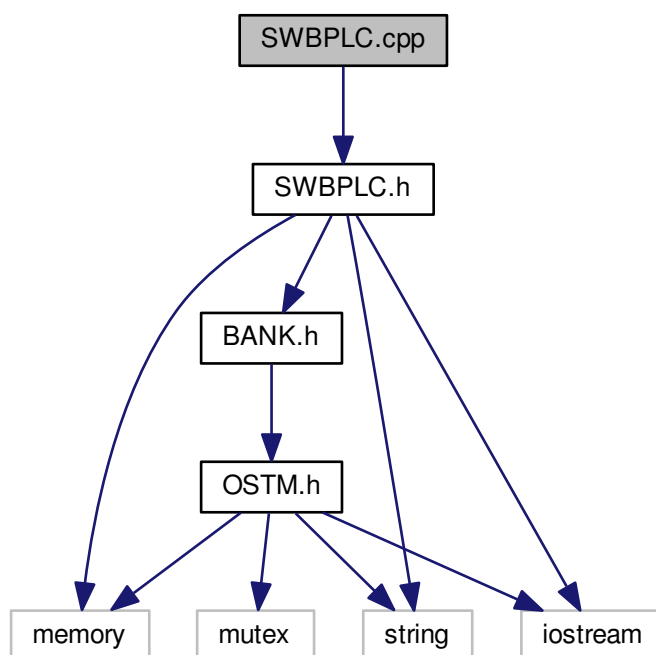
00096     virtual int GetNumber_of_huawei();
00097     virtual void SetNumber_of_sony(int _number_of_sony);
00098     virtual int GetNumber_of_sony();
00099     virtual void SetNumber_of_samsung(int _number_of_samsung);
00100     virtual int GetNumber_of_samsung();
00101     virtual void SetNumber_of_iphones(int _number_of_iphones);
00102     virtual int GetNumber_of_iphones();
00103     virtual void SetShop_name(std::string _shop_name);
00104     virtual std::string GetShop_name();
00105     virtual void SetShop_address(std::string _shop_address);
00106     virtual std::string GetShop_address();
00107
00108
00109 private:
00110     std::string _shop_address;
00111     std::string _shop_name;
00112     int _number_of_iphones;
00113     int _number_of_samsung;
00114     int _number_of_sony;
00115     int _number_of_huawei;
00116     int _number_of_nokia;
00117     int _number_of_alcatel;
00118
00119 };
00120
00121 #endif /* SLIGO_W_H */
00122

```

6.43 SWBPLC.cpp File Reference

```
#include "SWBPLC.h"
```

Include dependency graph for SWBPLC.cpp:



6.44 SWBPLC.cpp

```
00001
```



```

00002  /*
00003  * File:   SWBPLC.cpp
00004  * Author: Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010  #include "SWBPLC.h"
00011
00012  SWBPLC::SWBPLC(const SWBPLC& orig) {
00013  }
00014
00015  SWBPLC::~SWBPLC() {
00016  }
00022  std::shared_ptr<OSTM> SWBPLC::getBaseCopy(std::shared_ptr<OSTM> object)
00023  {
00024      std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00025      std::shared_ptr<BANK> obj(new SWBPLC(objTO, object->Get_Version(), object->Get_Unique_ID()));
00026      std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00027
00028      return ostm_obj;
00029  }
00034  void SWBPLC::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00035
00036      std::shared_ptr<SWBPLC> objTO = std::dynamic_pointer_cast<SWBPLC>(to);
00037      std::shared_ptr<SWBPLC> objFROM = std::dynamic_pointer_cast<SWBPLC>(from);
00038      objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00039      objTO->Set_Version(objFROM->Get_Version());
00040      objTO->SetAccountNumber(objFROM->GetAccountNumber());
00041      objTO->SetBalance(objFROM->GetBalance());
00042
00043  }
00044
00048  //std::shared_ptr<SWBPLC> SWBPLC::_cast(std::shared_ptr<OSTM> _object){
00049  //
00050  //    return static_cast<std::shared_ptr<SWBPLC>>(_object);
00051  //}
00055  void SWBPLC::toString()
00056  {
00057      std::cout << "\nSWBPLC BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt
account : " << this->GetAccountNumber() << "\nDouble value : " << this->
GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00058  }
00059
00060  void SWBPLC::SetAddress(std::string address) {
00061      this->address = address;
00062  }
00063
00064  std::string SWBPLC::GetAddress() const {
00065      return address;
00066  }
00067
00068  void SWBPLC::SetBalance(double balance) {
00069      this->balance = balance;
00070  }
00071
00072  double SWBPLC::GetBalance() const {
00073      return balance;
00074  }
00075
00076  void SWBPLC::SetAccountNumber(int accountNumber) {
00077      this->accountNumber = accountNumber;
00078  }
00079
00080  int SWBPLC::GetAccountNumber() const {
00081      return accountNumber;
00082  }
00083
00084  void SWBPLC::SetLastName(std::string lastName) {
00085      this->lastName = lastName;
00086  }
00087
00088  std::string SWBPLC::GetLastName() const {
00089      return lastName;
00090  }
00091
00092  void SWBPLC::SetFirstName(std::string firstName) {
00093      this->firstName = firstName;
00094  }
00095
00096  std::string SWBPLC::GetFirstName() const {
00097      return firstName;
00098  }
00099
00100  void SWBPLC::SetFullname(std::string fullname) {

```

```

00101     this->fullname = fullname;
00102 }
00103
00104 std::string SWBPLC::GetFullname() const {
00105     return fullname;
00106 }
00107

```

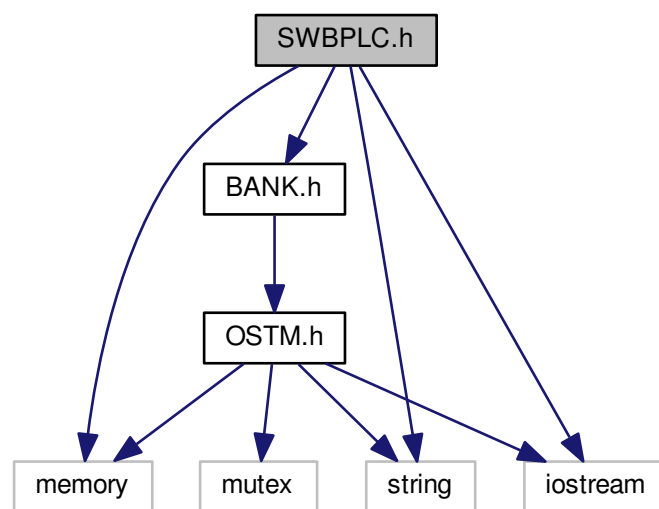
6.45 SWBPLC.h File Reference

```

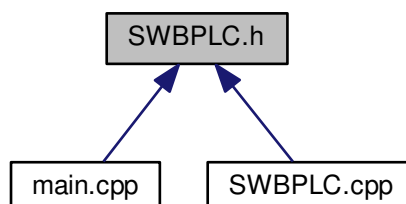
#include "BANK.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for SWBPLC.h:



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



Classes

- class [SWBPLC](#)

6.46 SWBPLC.h

```

00001
00002 /*
00003  * File:    SWBPLC.h
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef SWBPLC_H
00011 #define SWBPLC_H
00012 #include "BANK.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class SWBPLC : public BANK {
00020 public:
00024     SWBPLC() : BANK() {
00025         this->accountNumber = 0;
00026         this->balance = 50;
00027         this->firstName = "Joe";
00028         this->lastName = "Blog";
00029         this->address = "High street, Carlow";
00030         this->fullname = firstName + " " + lastName;
00031     };
00035     SWBPLC(int accountNumber, double balance, std::string
firstName, std::string lastName, std::string address) :
BANK() {
00036         this->accountNumber = accountNumber;
00037         this->balance = balance;
00038         this->firstName = firstName;
00039         this->lastName = lastName;
00040         this->address = address;
00041         this->fullname = firstName + " " + lastName;
00042     };
00046     SWBPLC(std::shared_ptr<BANK> obj, int _version, int _unique_id) : BANK(_version, _unique_id)
{
00047
00048         this->accountNumber = obj->GetAccountNumber();
00049         this->balance = obj->GetBalance();
00050         this->firstName = obj->GetFirstName();
00051         this->lastName = obj->GetLastName();
00052         this->address = obj->GetAddress();
00053         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054
00055     };
00059     SWBPLC(const SWBPLC& orig);
00063     SWBPLC operator=(const SWBPLC& orig) {};
00067     virtual ~SWBPLC();
00068
00069     /*
00070      * Implement OSTM virtual methods
00071      */
00072     //virtual std::shared_ptr<SWBPLC> _cast(std::shared_ptr<OSTM> _object);
00073     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00074     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00075     virtual void toString();
00076
00077     /*
00078      * Implement BANK virtual methods
00079      */
00080     virtual void SetAddress(std::string address);
00081     virtual std::string GetAddress() const;
00082     virtual void SetBalance(double balance);
00083     virtual double GetBalance() const;
00084     virtual void SetAccountNumber(int accountNumber);
00085     virtual int GetAccountNumber() const;
00086     virtual void SetLastName(std::string lastName);
00087     virtual std::string GetLastName() const;
00088     virtual void SetFirstName(std::string firstName);
00089     virtual std::string GetFirstName() const;
00090     virtual void SetFullname(std::string fullname);
00091     virtual std::string GetFullname() const;
00092 private:
00093     std::string fullname;
00094     std::string firstName;

```

```

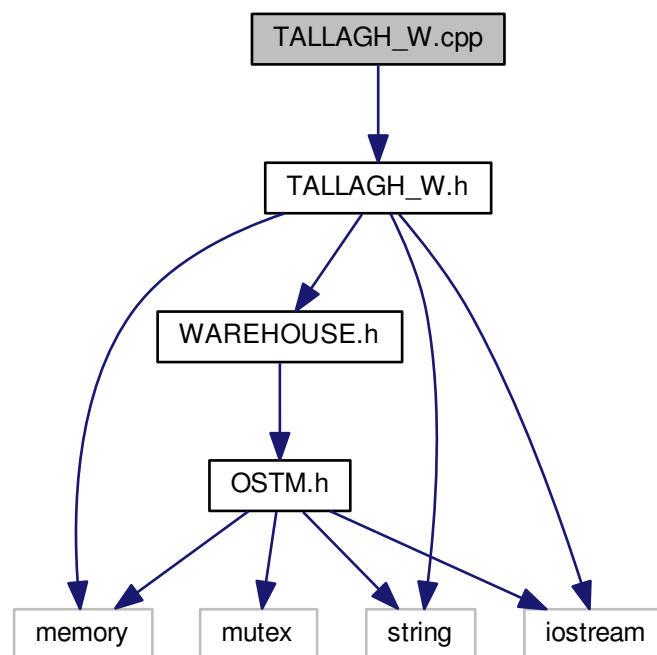
00095     std::string lastName;
00096     int accountNumber;
00097     double balance;
00098     std::string address;
00099
00100 };
00101
00102 #endif /* SWBPLC_H */
00103

```

6.47 TALLAGH_W.cpp File Reference

```
#include "TALLAGH_W.h"
```

Include dependency graph for TALLAGH_W.cpp:



6.48 TALLAGH_W.cpp

```

00001
00002 /*
00003  * File:    TALLAGH_W.cpp
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #include "TALLAGH_W.h"
00011
00012 TALLAGH_W::~TALLAGH_W() {
00013 }
00014
00015 TALLAGH_W::TALLAGH_W(const TALLAGH_W& orig) {
00016 }
00022 std::shared_ptr<OSTM> TALLAGH_W::getBaseCopy(std::shared_ptr<OSTM> object)

```

```

00023 {
00024
00025     std::shared_ptr<WAREHOUSE> objTO = std::dynamic_pointer_cast<WAREHOUSE>(object);
00026     std::shared_ptr<WAREHOUSE> obj(new TALLAGH_W(objTO, object->Get_Version(), object->
    Get_Unique_ID()));
00027     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00028     return ostm_obj;
00029 }
00035 void TALLAGH_W::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00036
00037     std::shared_ptr<TALLAGH_W> objTO = std::dynamic_pointer_cast<TALLAGH_W>(to);
00038     std::shared_ptr<TALLAGH_W> objFROM = std::dynamic_pointer_cast<TALLAGH_W>(from);
00039     objTO->_shop_address = objFROM->GetShop_address();
00040     objTO->_shop_name = objFROM->GetShop_name();
00041     objTO->_number_of_iphones = objFROM->GetNumber_of_iphones();
00042     objTO->_number_of_samsung = objFROM->GetNumber_of_samsung();
00043     objTO->_number_of_sony = objFROM->GetNumber_of_sony();
00044     objTO->_number_of_huawei = objFROM->GetNumber_of_huawei();
00045     objTO->_number_of_nokia = objFROM->GetNumber_of_nokia();
00046     objTO->_number_of_alcatel = objFROM->GetNumber_of_alcatel();
00047     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00048     objTO->Set_Version(objFROM->Get_Version());
00049
00050
00051 }
00055 //std::shared_ptr<TALLAGH_W> TALLAGH_W::_cast(std::shared_ptr<OSTM> _object){
00056 //
00057 //     return static_cast<std::shared_ptr<TALLAGH_W>>(_object);
00058 //}
00062 void TALLAGH_W::toString()
00063 {
00064     std::cout << "\n" << this->GetShop_name() << "\nUnique ID : " << this->
    Get_Unique_ID() << "\nShop Name : " << this->GetShop_name() << "\nShop Address : "
    << this->GetShop_address() << "\nNo. Iphones : " << this->
    GetNumber_of_iphones() << "\nNo. Samsung : " << this->
    GetNumber_of_samsung() << "\nNo. Sony : " << this->
    GetNumber_of_sony() << "\nNo. Huawei : " << this->
    GetNumber_of_huawei() << "\nNo. Nokia : " << this->
    GetNumber_of_nokia() << "\nNo. Alcatel : " << this->
    GetNumber_of_alcatel() << "\nVersion number : " << this->
    Get_Version() << std::endl;
00065 }
00066
00067 void TALLAGH_W::SetNumber_of_alcatel(int
    _number_of_alcatel) {
00068     this->_number_of_alcatel = _number_of_alcatel;
00069 }
00070
00071 int TALLAGH_W::GetNumber_of_alcatel() {
00072     return _number_of_alcatel;
00073 }
00074
00075 void TALLAGH_W::SetNumber_of_nokia(int
    _number_of_nokia) {
00076     this->_number_of_nokia = _number_of_nokia;
00077 }
00078
00079 int TALLAGH_W::GetNumber_of_nokia() {
00080     return _number_of_nokia;
00081 }
00082
00083 void TALLAGH_W::SetNumber_of_huawei(int
    _number_of_huawei) {
00084     this->_number_of_huawei = _number_of_huawei;
00085 }
00086
00087 int TALLAGH_W::GetNumber_of_huawei() {
00088     return _number_of_huawei;
00089 }
00090
00091 void TALLAGH_W::SetNumber_of_sony(int
    _number_of_sony) {
00092     this->_number_of_sony = _number_of_sony;
00093 }
00094
00095 int TALLAGH_W::GetNumber_of_sony() {
00096     return _number_of_sony;
00097 }
00098
00099 void TALLAGH_W::SetNumber_of_samsung(int
    _number_of_samsung) {
00100     this->_number_of_samsung = _number_of_samsung;
00101 }
00102
00103 int TALLAGH_W::GetNumber_of_samsung() {
00104     return _number_of_samsung;

```

```

00105 }
00106
00107 void TALLAGH_W::SetNumber_of_iphones(int
    _number_of_iphones) {
00108     this->_number_of_iphones = _number_of_iphones;
00109 }
00110
00111 int TALLAGH_W::GetNumber_of_iphones() {
00112     return _number_of_iphones;
00113 }
00114
00115 void TALLAGH_W::SetShop_name(std::string _shop_name) {
00116     this->_shop_name = _shop_name;
00117 }
00118
00119 std::string TALLAGH_W::GetShop_name() {
00120     return _shop_name;
00121 }
00122
00123 void TALLAGH_W::SetShop_address(std::string
    _shop_address) {
00124     this->_shop_address = _shop_address;
00125 }
00126
00127 std::string TALLAGH_W::GetShop_address() {
00128     return _shop_address;
00129 }

```

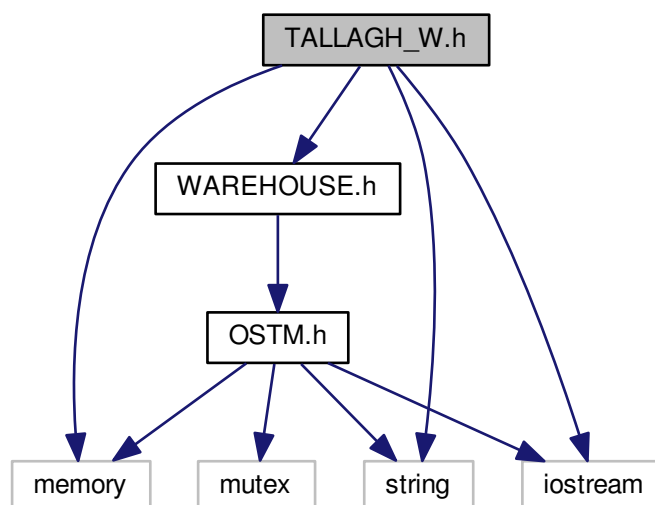
6.49 TALLAGH_W.h File Reference

```

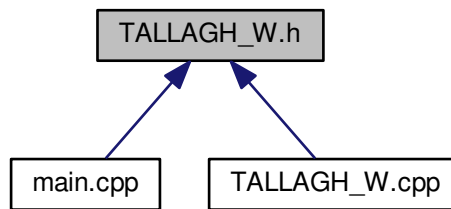
#include "WAREHOUSE.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for TALLAGH_W.h:



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



Classes

- class [TALLAGH_W](#)

6.50 TALLAGH_W.h

```

00001
00002 /*
00003  * File:    TALLAGH_W.h
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef TALLAGH_W_H
00011 #define TALLAGH_W_H
00012 #include "WAREHOUSE.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class TALLAGH_W :public WAREHOUSE {
00020 public:
00024     TALLAGH_W() : WAREHOUSE(){
00025
00026         this->_shop_address = "Tallagh Low street";
00027         this->_shop_name = "TALLAGH T_WAREHOUSE";
00028         this->_number_of_iphones = 200;
00029         this->_number_of_samsung = 200;
00030         this->_number_of_sony = 200;
00031         this->_number_of_huawei = 200;
00032         this->_number_of_nokia = 200;
00033         this->_number_of_alcatel = 200;
00034     };
00038     TALLAGH_W(std::string address, std::string shop_name, int iphone, int samsung, int sony, int
00039     huawei, int nokia, int alcatel): WAREHOUSE(){
00039         /*
00040         * copy over values
00041         */
00042         this->_shop_address = address;
00043         this->_shop_name = shop_name;
00044         this->_number_of_iphones = iphone;
00045         this->_number_of_samsung = samsung;
00046         this->_number_of_sony = sony;
00047         this->_number_of_huawei = huawei;
00048         this->_number_of_nokia = nokia;
00049         this->_number_of_alcatel = alcatel;
00050
00051     };
00055     TALLAGH_W(std::shared_ptr<WAREHOUSE> obj, int _version, int _unique_id):
00056     WAREHOUSE(_version, _unique_id){
00056         /*
00057         * copy over values
00058         */

```

```

00059         this->_shop_address = obj->GetShop_address();
00060         this->_shop_name = obj->GetShop_name();
00061         this->_number_of_iphones = obj->GetNumber_of_iphones();
00062         this->_number_of_samsung = obj->GetNumber_of_samsung();
00063         this->_number_of_sony = obj->GetNumber_of_sony();
00064         this->_number_of_huawei = obj->GetNumber_of_huawei();
00065         this->_number_of_nokia = obj->GetNumber_of_nokia();
00066         this->_number_of_alcatel = obj->GetNumber_of_alcatel();
00067     }
00071     TALLAGH_W(const TALLAGH_W& orig);
00075     TALLAGH_W operator=(const TALLAGH_W& orig){};
00079     virtual ~TALLAGH_W();
00080
00081     /*
00082     * Implement OSTM virtual methods
00083     */
00084     //virtual std::shared_ptr<TALLAGH_W> _cast(std::shared_ptr<OSTM> _object);
00085     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00086     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00087     virtual void toString();
00088     /*
00089     * Implement Warehouse methods
00090     */
00091     virtual void SetNumber_of_alcatel(int _number_of_alcatel);
00092     virtual int GetNumber_of_alcatel();
00093     virtual void SetNumber_of_nokia(int _number_of_nokia);
00094     virtual int GetNumber_of_nokia();
00095     virtual void SetNumber_of_huawei(int _number_of_huawei);
00096     virtual int GetNumber_of_huawei();
00097     virtual void SetNumber_of_sony(int _number_of_sony);
00098     virtual int GetNumber_of_sony();
00099     virtual void SetNumber_of_samsung(int _number_of_samsung);
00100     virtual int GetNumber_of_samsung();
00101     virtual void SetNumber_of_iphones(int _number_of_iphones);
00102     virtual int GetNumber_of_iphones();
00103     virtual void SetShop_name(std::string _shop_name);
00104     virtual std::string GetShop_name();
00105     virtual void SetShop_address(std::string _shop_address);
00106     virtual std::string GetShop_address();
00107
00108 private:
00109     std::string _shop_address;
00110     std::string _shop_name;
00111     int _number_of_iphones;
00112     int _number_of_samsung;
00113     int _number_of_sony;
00114     int _number_of_huawei;
00115     int _number_of_nokia;
00116     int _number_of_alcatel;
00117
00118 };
00119
00120
00121 #endif /* TALLAGH_W_H */
00122

```

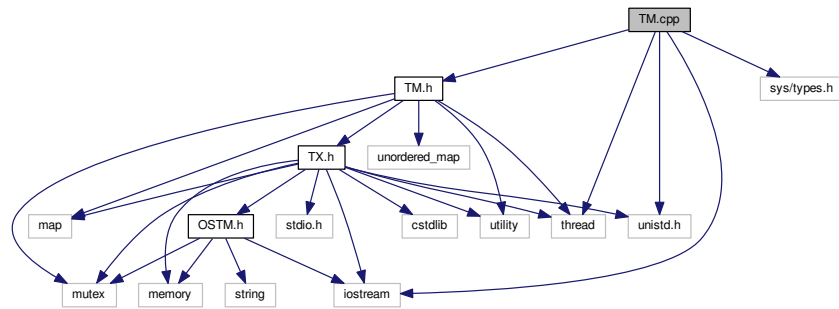
6.51 TM.cpp File Reference

```

#include "TM.h"
#include <thread>
#include <unistd.h>
#include <sys/types.h>
#include <iostream>

```


Include dependency graph for TM.cpp:



6.52 TM.cpp

```

00001 /*
00002  * File:    TM.cpp
00003  * Author:  Zoltan Fuzesi
00004  *
00005  * Created on December 18, 2017, 2:09 PM
00006  * Transaction Manager class methods implementation
00007  */
00008 #include "TM.h"
00009 #include <thread>
00010 #include <unistd.h>
00011 // #include <process.h>
00012 #include <sys/types.h>
00013 #include <iostream>
00014
00015 int TM::_tm_id;
00016 std::map<int, std::map< std::thread::id, int >> TM::process_map_collection;
00017 TM& TM::Instance() {
00018     static TM _instance;
00019     _instance._tm_id = getpid();
00020     return _instance;
00021 }
00022
00023 //TM Transaction manager checking the Process ID existence in the map
00024 //If not in the map then register
00025 void TM::registerTX()
00026 {
00027     std::lock_guard<std::mutex> guard(register_Lock);
00028     int ppid = getpid();
00029     std::map<int, std::map< std::thread::id, int >>::iterator process_map_collection_Iterator =
00030     TM::process_map_collection.find(ppid);
00031     if (process_map_collection_Iterator == TM::process_map_collection.end()) {
00032         /*
00033          * Register main process/application to the global map
00034          */
00035         std::map< std::thread::id, int >map = get_thread_Map();
00036         TM::process_map_collection.insert({ppid, map});
00037     }
00038     std::map<std::thread::id, std::shared_ptr < TX>::iterator it = txMap.find(
00039     std::this_thread::get_id());
00040     if (it == txMap.end()) {
00041         std::shared_ptr<TX> _transaction_object(new TX(std::this_thread::get_id()));
00042         txMap.insert({std::this_thread::get_id(), _transaction_object});
00043         /*
00044          * Get the map if registered first time
00045          */
00046         process_map_collection_Iterator = TM::process_map_collection.find(ppid);
00047         /*
00048          * Insert to the GLOBAL MAP as a helper to clean up at end of main process
00049          */
00050         process_map_collection_Iterator->second.insert({std::this_thread::get_id(), 1});
00051     }
00052 }
00053
00054 }
00055
00056 }
00057
00058 }
00059
00060 }
00061
00062 }
00063
00064 }
00065
00066 }
00067
00068 }
00069
00070 }
00071
00072

```

```

00078 std::shared_ptr<TX>const TM::_get_tx()
00079 {
00080     std::lock_guard<std::mutex> guard(get_Lock);
00081
00082     std::map<std::thread::id, std::shared_ptr<TX>>::iterator it = txMap.find(std::this_thread::get_id(
00083 ));
00084     if(it == txMap.end())
00085     {
00086         registerTX();
00087         it = txMap.find(std::this_thread::get_id());
00088     } else {
00089         it->second->_increase_tx_nesting();
00090     }
00091     //it = txMap.find(std::this_thread::get_id());
00092
00093     return it->second;
00094 }
00095
00096 }
00101 void TM::_TX_EXIT(){
00102     TX tx(std::this_thread::get_id());
00103     int ppid = getpid();
00104     std::map<int, std::map< std::thread::id, int >>::iterator process_map_collection_Iterator =
TM::process_map_collection.find(ppid);
00105     if (process_map_collection_Iterator != TM::process_map_collection.end()) {
00106
00107         for (auto current = process_map_collection_Iterator->second.begin(); current !=
process_map_collection_Iterator->second.end(); ++current) {
00108             /*
00109              * Delete all transaction associated with the actual main process
00110              */
00111             txMap.erase(current->first);
00112         }
00113         TM::process_map_collection.erase(ppid);
00114
00115     }
00116     tx.ostm_exit();
00117 }
00121 void TM::print_all(){
00122     get_Lock.lock();
00123     for (auto current = txMap.begin(); current != txMap.end(); ++current) {
00124         std::cout << "KEY : " << current->first << std::endl;
00125     }
00126     get_Lock.unlock();
00127 }
00128
00133 std::map< std::thread::id, int > TM::get_thread_Map() {
00134     std::map< std::thread::id, int > thread_Map;
00135     return thread_Map;
00136 }

```

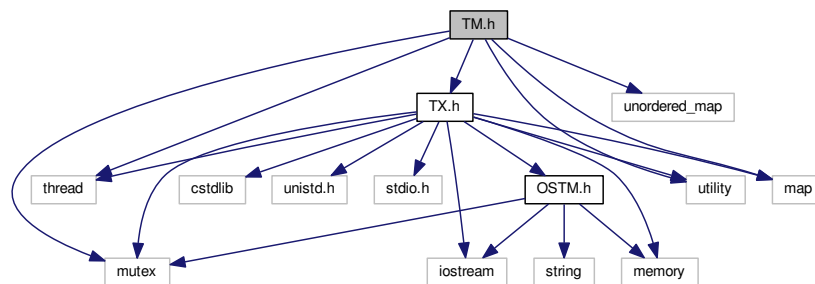
6.53 TM.h File Reference

```

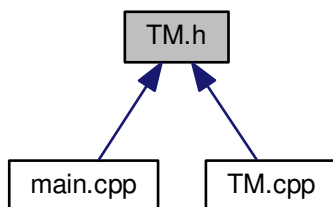
#include <thread>
#include <mutex>
#include <unordered_map>
#include <utility>
#include <map>
#include "TX.h"

```

Include dependency graph for TM.h:



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



Classes

- class [TM](#)

6.54 TM.h

```

00001
00067 #ifndef TM_H
00068 #define TM_H
00069
00070 #include <thread>
00071 // #include <unistd.h> // used for pid_t
00072 // #include <io.h>
00073 #include <mutex>
00074 #include <unordered_map>
00075 #include <utility>
00076 // #include <process.h>
00077 #include <map>
00078 #include "TX.h"
00079
00080 class TM {
00081 private:
00085     TM() = default;
00089     ~TM() = default;
00093     TM(const TM&) = delete;
00097     TM& operator=(const TM&) = delete;
00101     std::map<std::thread::id, std::shared_ptr<TX>> txMap;
00106     static std::map<int, std::map<std::thread::id, int >>

```

```

    process_map_collection;
00110     std::map< std::thread::id, int > get_thread_Map();
00114     void registerTX();
00118     std::mutex register_Lock;
00122     std::mutex get_Lock;
00126     static int _tm_id;
00127
00128
00129 public:
00130
00134     static TM& Instance();
00138     std::shared_ptr<TX>const _get_tx();
00142     void _TX_EXIT();
00146     void print_all();
00147
00148
00149 };
00150
00151
00152 #endif // TM_H

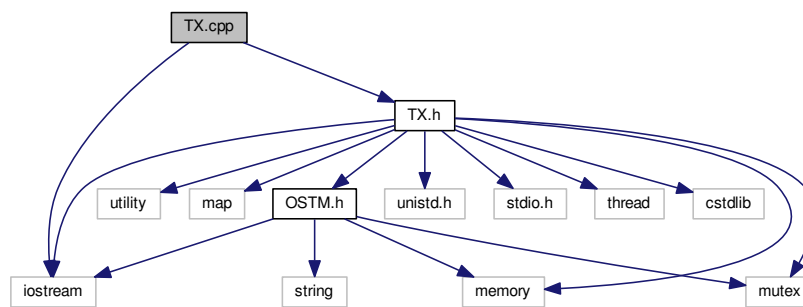
```

6.55 TX.cpp File Reference

```

#include "TX.h"
#include <iostream>
Include dependency graph for TX.cpp:

```



6.56 TX.cpp

```

00001 /*
00002  * File:   TX.cpp
00003  * Author: Zoltan Fuzesi
00004  *
00005  * Created on December 18, 2017, 2:09 PM
00006  * TX cpp file methods implementations
00007  */
00008 #include "TX.h"
00009 #include <iostream>
00013 std::map<int, std::shared_ptr<OSTM> >TX::main_Process_Map_collection;
00017 std::map<int, std::map< int, int >> TX::process_map_collection;
00021 std::mutex TX::register_Lock;
00025 int TX::test_counter = 0;
00031 TX::TX(std::thread::id id) {
00032     this->transaction_Number = id;
00033     this->_tx_nesting_level = 0;
00034 }
00038 TX::~TX() {
00039
00040 }
00044 TX::TX(const TX& orig) {
00045
00046 }
00047
00052 void TX::th_exit() {

```

```

00053
00054     if (this->_tx_nesting_level > 0) {
00055         /*
00056          * Active nested transactions running in background, do not delete anything yet
00057          */
00058     } else {
00059         /*
00060          * Remove all elements map entries from transaction and clear the map
00061          */
00062         working_Map_collection.clear();
00063     }
00064 }
00065
00072 void TX::ostm_exit() {
00073     std::map<int, std::shared_ptr<OSTM>>::iterator main_Process_Map_collection_Iterator;
00074
00075     int ppid = getpid();
00076     std::map<int, std::map< int, int >::iterator process_map_collection_Iterator =
TX::process_map_collection.find(ppid);
00077     if (process_map_collection_Iterator != TX::process_map_collection.end()) {
00078
00079         for (auto current = process_map_collection_Iterator->second.begin(); current !=
process_map_collection_Iterator->second.end(); ++current) {
00080             main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(current->first);
00081
00082             if (main_Process_Map_collection_Iterator !=
TX::main_Process_Map_collection.end()) {
00083                 /*
00084                  * Delete element from shared main_Process_Map_collection by object unique key value,
shared_ptr will destroy automatically
00085                  */
00086                 TX::main_Process_Map_collection.erase(
main_Process_Map_collection_Iterator->first);
00087             }
00088         }
00089         /*
00090          * Delete from Process_map_collection, Main process exits delete association with library
00091          */
00092         TX::process_map_collection.erase(process_map_collection_Iterator->first);
00093     }
00094 }
00095
00104 void TX::_register(std::shared_ptr<OSTM> object) {
00105     /*
00106      * MUST USE SHARED LOCK TO PROTECT SHARED GLOBAL MAP/COLLECTION
00107      */
00108     std::lock_guard<std::mutex> guard(TX::register_Lock);
00109
00110     /*
00111      * Check for null pointer !
00112      * Null pointer can cause segmentation fault!!!
00113      */
00114     if(object == nullptr){
00115         throw std::runtime_error(std::string("[RUNTIME ERROR : NULL POINTER IN REGISTER FUNCTION]") );
00116     }
00117
00118     int ppid = getpid();
00119     std::map<int, std::map< int, int >::iterator process_map_collection_Iterator =
TX::process_map_collection.find(ppid);
00120     if (process_map_collection_Iterator == TX::process_map_collection.end()) {
00121         /*
00122          * Register main process/application to the global map
00123          */
00124         std::map< int, int >map = get_thread_Map();
00125         TX::process_map_collection.insert({ppid, map});
00126         /*
00127          * Get the map if registered first time
00128          */
00129         process_map_collection_Iterator = TX::process_map_collection.find(ppid);
00130     }
00131     std::map<int, std::shared_ptr<OSTM>>::iterator main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(object->Get_Unique_ID());
00132     if (main_Process_Map_collection_Iterator == TX::main_Process_Map_collection
.end()) {
00133         /*
00134          * Insert to the GLOBAL MAP
00135          */
00136         TX::main_Process_Map_collection.insert({object->Get_Unique_ID(),
object});
00137         /*
00138          * Insert to the GLOBAL MAP as a helper to clean up at end of main process
00139          */
00140         process_map_collection_Iterator->second.insert({object->Get_Unique_ID(), 1});
00141     }
00142
00143

```

```

00144     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator
= working_Map_collection.find(object->Get_Unique_ID());
00145     if (working_Map_collection_Object_Shared_Pointer_Iterator ==
working_Map_collection.end()) {
00146         working_Map_collection.insert({object->Get_Unique_ID(), object->getBaseCopy(
object)});
00147     }
00148 }
00149 }
00150 }
00155 std::shared_ptr<OSTM> TX::load(std::shared_ptr<OSTM> object) {
00156     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00157     /*
00158     * Check for null pointer !
00159     * Null pointer can cause segmentation fault!!!
00160     */
00161     if(object == nullptr){
00162         throw std::runtime_error(std::string("[RUNTIME ERROR : NULL POINTER IN LOAD FUNCTION]") );
00163     }
00164 }
00165 }
00166     working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.find(object->Get_Unique_ID());
00167 }
00168     if (working_Map_collection_Object_Shared_Pointer_Iterator !=
working_Map_collection.end()) {
00169         return working_Map_collection_Object_Shared_Pointer_Iterator->second->getBaseCopy(
working_Map_collection_Object_Shared_Pointer_Iterator->second);
00170     } else { throw std::runtime_error(std::string("[RUNTIME ERROR : NO OBJECT FOUND LOAD FUNCTION]") );}
00171 }
00172 void TX::store(std::shared_ptr<OSTM> object) {
00173     /*
00174     * Check for null pointer !
00175     * Null pointer can cause segmentation fault!!!
00176     */
00177     if(object == nullptr){
00178         throw std::runtime_error(std::string("[RUNTIME ERROR : NULL POINTER IN STORE FUNCTION]") );
00179     }
00180     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00181     working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.find(object->Get_Unique_ID());
00182     if (working_Map_collection_Object_Shared_Pointer_Iterator !=
working_Map_collection.end()) {
00183         working_Map_collection_Object_Shared_Pointer_Iterator->second = object;
00184     } else { std::cout << "[ERROR STORE]" << std::endl; }
00185 }
00186 bool TX::commit() {
00187     bool can_Commit = true;
00188     /*
00189     * Dealing with nested transactions first
00190     */
00191     if (this->_tx_nesting_level > 0) {
00192         _decrease_tx_nesting();
00193         return true;
00194     }
00195     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00196     std::map<int, std::shared_ptr<OSTM>>::iterator main_Process_Map_collection_Iterator;
00197     for (working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
!= working_Map_collection.end();
working_Map_collection_Object_Shared_Pointer_Iterator++) {
00198         main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Unique_ID());
00199         /*
00200         * Throws runtime error if object can not find
00201         */
00202         if(main_Process_Map_collection_Iterator ==
TX::main_Process_Map_collection.end())
00203         {
00204             throw std::runtime_error(std::string("[RUNTIME ERROR : CAN'T FIND OBJECT COMMIT FUNCTION]")
);
00205         }
00206     }
00207     /*
00208     * Busy wait WHILE object locked by other thread

```

```

00230         */
00231         while (! (main_Process_Map_collection_Iterator->second->is_Locked()));
00232
00233         if (main_Process_Map_collection_Iterator->second->Get_Version() >
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Version()) {
00234
00235             working_Map_collection_Object_Shared_Pointer_Iterator->second->Set_Can_Commit(false);
00236             can_Commit = false;
00237             break;
00238         } else {
00239
00240             working_Map_collection_Object_Shared_Pointer_Iterator->second->Set_Can_Commit(true);
00241         }
00242     }
00243     if (!can_Commit) {
00244         TX::test_counter += 1;
00245         for (working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
!= working_Map_collection.end();
working_Map_collection_Object_Shared_Pointer_Iterator++) {
00246
00247             main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find(
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Unique_ID());
00248             (working_Map_collection_Object_Shared_Pointer_Iterator->second->copy(
working_Map_collection_Object_Shared_Pointer_Iterator->second, main_Process_Map_collection_Iterator->second);
00249
00250             }
00251
00252             _release_object_lock();
00253
00254             return false;
00255         } else {
00256             /*
00257              * Commit changes
00258              */
00259             for (working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
!= working_Map_collection.end();
working_Map_collection_Object_Shared_Pointer_Iterator++) {
00260
00261                 main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find((
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Unique_ID());
00262                 if (main_Process_Map_collection_Iterator !=
TX::main_Process_Map_collection.end()) {
00263
00264                     (main_Process_Map_collection_Iterator->second->copy(
main_Process_Map_collection_Iterator->second, working_Map_collection_Object_Shared_Pointer_Iterator->second);
00265                     main_Process_Map_collection_Iterator->second->increase_VersionNumber();
00266
00267                 } else {
00268                     throw std::runtime_error(std::string("[RUNTIME ERROR : CAN'T FIND OBJECT COMMIT
00269 FUNCTION]"));
00270
00271                 }
00272             }
00273
00274             _release_object_lock();
00275             this->th_exit();
00276             return true;
00277         }
00278     }
00279 } //Commit finish
00280
00286 void TX::_release_object_lock(){
00287
00288     std::map< int, std::shared_ptr<OSTM> >::iterator working_Map_collection_Object_Shared_Pointer_Iterator;
00289     std::map<int, std::shared_ptr<OSTM>>::iterator main_Process_Map_collection_Iterator;
00290     for (working_Map_collection_Object_Shared_Pointer_Iterator =
working_Map_collection.begin(); working_Map_collection_Object_Shared_Pointer_Iterator
!= working_Map_collection.end();
working_Map_collection_Object_Shared_Pointer_Iterator++) {
00291
00292         main_Process_Map_collection_Iterator =
TX::main_Process_Map_collection.find((
working_Map_collection_Object_Shared_Pointer_Iterator->second->Get_Unique_ID());
00293         if (main_Process_Map_collection_Iterator !=
TX::main_Process_Map_collection.end()) {
00294             /*
00295              * Release object lock
00296              */
00297             (main_Process_Map_collection_Iterator->second->unlock_Mutex();
00298
00299         }
00300     }

```

```

00301 }
00302
00307 void TX::_increase_tx_nesting() {
00308     this->_tx_nesting_level += 1;
00309     // std::cout << "[this->_tx_nesting_level] = " << this->_tx_nesting_level << std::endl;
00310 }
00311
00316 void TX::_decrease_tx_nesting() {
00317     // std::cout << "[this->_tx_nesting_level] = " << this->_tx_nesting_level << std::endl;
00318     this->_tx_nesting_level -= 1;
00319 }
00320 }
00324 int TX::getTest_counter() {
00325     return TX::test_counter;
00326 }
00331 const std::thread::id TX::_get_tx_number() const {
00332     return transaction_Number;
00333 }
00338 std::map< int, int > TX::get_thread_Map() {
00339     std::map< int, int > thread_Map;
00340     return thread_Map;
00341 }
00342
00346 void TX::_print_all_tx() {
00347     std::cout << "[PRINTALLTHREAD]" << std::endl;
00348     std::map< int, std::shared_ptr<OSTM> >::iterator it;
00349     /*
00350     * All registered thread id in the TX global
00351     */
00352     int ppid = getpid();
00353     std::map<int, std::map< int, int >::iterator process_map_collection_Iterator =
00354     TX::process_map_collection.find(ppid);
00355     if (process_map_collection_Iterator != TX::process_map_collection.end()) {
00356         for (auto current = process_map_collection_Iterator->second.begin(); current !=
00357 process_map_collection_Iterator->second.end(); ++current) {
00358             it = working_Map_collection.find(current->first);
00359             if(it != working_Map_collection.end()){
00360                 std::cout << "[Unique number ] : " <<it->second->Get_Unique_ID() << std::endl;
00361             }
00362         }
00363     }
00364 }
00365 }
00366 }
00367 }

```

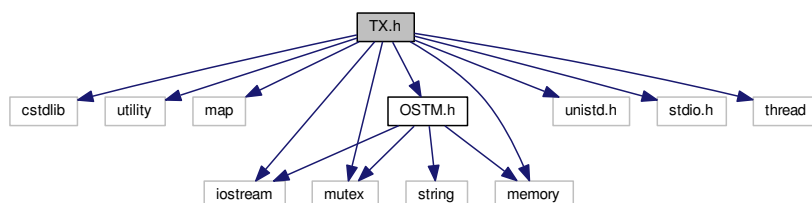
6.57 TX.h File Reference

```

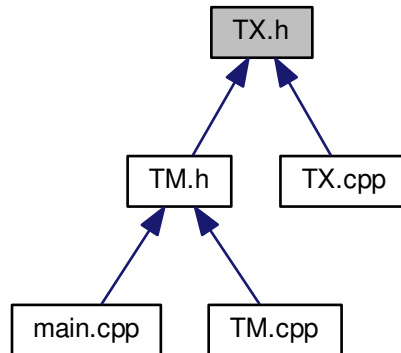
#include <cstdlib>
#include <utility>
#include <map>
#include <iostream>
#include <mutex>
#include <unistd.h>
#include <memory>
#include <stdio.h>
#include <thread>
#include "OSTM.h"

```

Include dependency graph for TX.h:



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



Classes

- class [TX](#)

6.58 TX.h

```

00001 /*
00002  * File:   TX.h
00003  * Author: Zoltan Fuzesi
00004  *
00005  * Created on December 18, 2017, 2:09 PM
00006  * Transaction class fields and methods declarations
00007  */
00008
00009 #ifndef TX_H
00010 #define TX_H
00011 #include <cstdlib>
00012 #include <utility>
00013 #include <map>
00014 #include <iostream>
00015 #include <mutex>
00016 #include <unistd.h>
00017 // #include <io.h>
00018 #include <memory>
00019 // #include <process.h>
00020 #include <stdio.h>
00021 #include <thread>
00022 #include "OSTM.h"
00023
00024 class TM;
00025
00026 class TX {
00027 public:
00031     TX(std::thread::id id);
00035     ~TX();
00039     TX(const TX& orig);
00043     void ostm_exit();
00044
00048     void _register(std::shared_ptr<OSTM> object);
00052     std::shared_ptr<OSTM> load(std::shared_ptr<OSTM> object);
00056     void store(std::shared_ptr<OSTM> object);
00060     bool commit();
00064     void _increase_tx_nesting();
00068     void _decrease_tx_nesting();
00072     friend class TM;
00073     /*
00074      * \brief ONLY FOR TESTING!!! returning the number of rollback happened during transactions
  
```

```

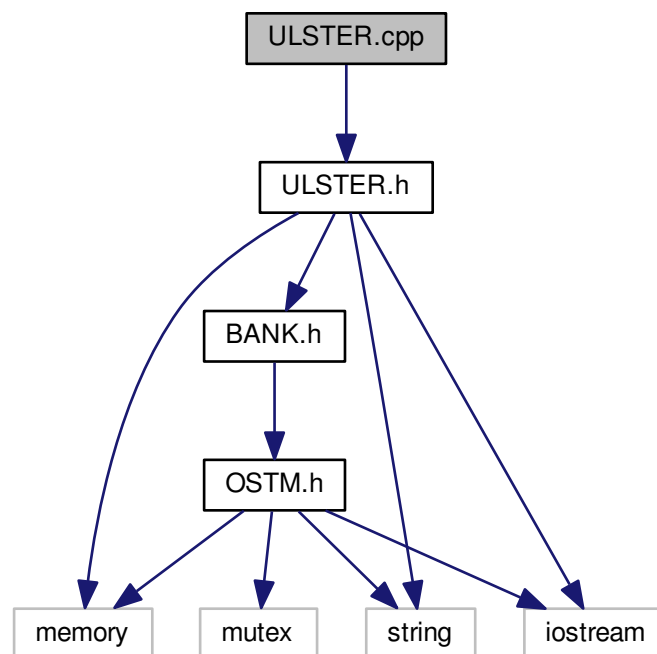
00075     */
00076     int getTest_counter();
00080     static int test_counter;
00081     /*
00082     * TESTING ONLY
00083     */
00084     void _print_all_tx() ;
00085
00086
00087 private:
00092     std::map< int, std::shared_ptr<OSTM> > working_Map_collection;
00098     std::thread::id transaction_Number;
00102     int _tx_nesting_level;
00103
00108     static std::map<int, std::shared_ptr<OSTM> >main_Process_Map_collection;
00113     static std::map<int, std::map< int, int >> process_map_collection;
00114     //static std::map<pid_t, std::map< int, std::pair<ppid, int> >> process_map_collection;
00118     std::map< int , int > get_thread_Map();
00122     static std::mutex register_Lock;
00126     const std::thread::id _get_tx_number() const;
00127
00131     void _release_object_lock();
00135     void th_exit();
00136
00137
00138
00139 };
00140 #endif // _TX_H_

```

6.59 ULSTER.cpp File Reference

```
#include "ULSTER.h"
```

Include dependency graph for ULSTER.cpp:



6.60 ULSTER.cpp

```
00001
```

```

00002  /*
00003  * File:    ULSTER.cpp
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010  #include "ULSTER.h"
00011
00012  //ULSTER::ULSTER() {
00013  //}
00014
00015  ULSTER::ULSTER(const ULSTER& orig) {
00016  }
00017
00018  ULSTER::~ULSTER() {
00019  }
00020
00021  std::shared_ptr<OSTM> ULSTER::getBaseCopy(std::shared_ptr<OSTM> object)
00022  {
00023      std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);
00024      std::shared_ptr<BANK> obj(new ULSTER(objTO, object->Get_Version(), object->Get_Unique_ID()));
00025      std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);
00026
00027      return ostm_obj;
00028  }
00029
00030  void ULSTER::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00031
00032      std::shared_ptr<ULSTER> objTO = std::dynamic_pointer_cast<ULSTER>(to);
00033      std::shared_ptr<ULSTER> objFROM = std::dynamic_pointer_cast<ULSTER>(from);
00034      objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00035      objTO->Set_Version(objFROM->Get_Version());
00036      objTO->Set_AccountNumber(objFROM->Get_AccountNumber());
00037      objTO->Set_Balance(objFROM->Get_Balance());
00038  }
00039
00040  //std::shared_ptr<ULSTER> ULSTER::_cast(std::shared_ptr<OSTM> _object){
00041  //
00042  //    return static_cast<std::shared_ptr<ULSTER>>(_object);
00043  //}
00044
00045  void ULSTER::toString()
00046  {
00047      std::cout << "\nULSTER BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account
00048      : " << this->Get_AccountNumber() << "\nDouble value : " << this->
00049      Get_Balance() << "\nFirst name: " << this->Get_FirstName() << "\nLast name : " <<
00050      this->Get_LastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00051  }
00052
00053  void ULSTER::SetAddress(std::string address) {
00054      this->address = address;
00055  }
00056
00057  std::string ULSTER::GetAddress() const {
00058      return address;
00059  }
00060
00061  void ULSTER::SetBalance(double balance) {
00062      this->balance = balance;
00063  }
00064
00065  double ULSTER::GetBalance() const {
00066      return balance;
00067  }
00068
00069  void ULSTER::SetAccountNumber(int accountNumber) {
00070      this->accountNumber = accountNumber;
00071  }
00072
00073  int ULSTER::GetAccountNumber() const {
00074      return accountNumber;
00075  }
00076
00077  void ULSTER::SetLastName(std::string lastName) {
00078      this->lastName = lastName;
00079  }
00080
00081  std::string ULSTER::GetLastName() const {
00082      return lastName;
00083  }
00084
00085  void ULSTER::SetFirstName(std::string firstName) {
00086      this->firstName = firstName;
00087  }
00088
00089  std::string ULSTER::GetFirstName() const {
00090      return firstName;
00091  }

```

```

00101 }
00102
00103 void ULSTER::SetFullname(std::string fullname) {
00104     this->fullname = fullname;
00105 }
00106
00107 std::string ULSTER::GetFullname() const {
00108     return fullname;
00109 }
00110

```

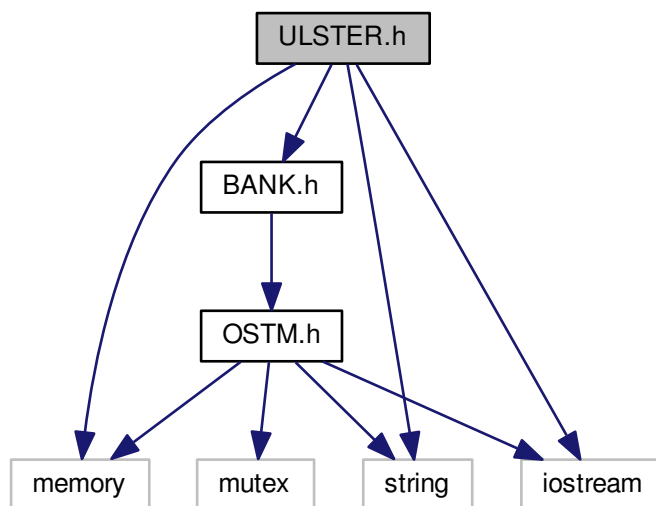
6.61 ULSTER.h File Reference

```

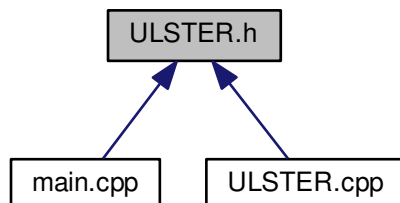
#include "BANK.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for ULSTER.h:



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



Classes

- class [ULSTER](#)

6.62 ULSTER.h

```

00001
00002 /*
00003  * File:    ULSTER.h
00004  * Author:  Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef ULSTER_H
00011 #define ULSTER_H
00012 #include "BANK.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class ULSTER : public BANK {
00020 public:
00024     ULSTER() : BANK() {
00025         this->accountNumber = 0;
00026         this->balance = 50;
00027         this->firstName = "Joe";
00028         this->lastName = "Blog";
00029         this->address = "High street, Carlow";
00030         this->fullname = firstName + " " + lastName;
00031     };
00035     ULSTER(int accountNumber, double balance, std::string
firstName, std::string lastName, std::string address) :
BANK() {
00036         this->accountNumber = accountNumber;
00037         this->balance = balance;
00038         this->firstName = firstName;
00039         this->lastName = lastName;
00040         this->address = address;
00041         this->fullname = firstName + " " + lastName;
00042     };
00046     ULSTER(std::shared_ptr<BANK> obj, int _version, int _unique_id) : BANK(_version, _unique_id)
{
00047
00048         this->accountNumber = obj->GetAccountNumber();
00049         this->balance = obj->GetBalance();
00050         this->firstName = obj->GetFirstName();
00051         this->lastName = obj->GetLastName();
00052         this->address = obj->GetAddress();
00053         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054     };
00058     ULSTER(const ULSTER& orig);
00062     ULSTER operator=(const ULSTER& orig) {};
00066     virtual ~ULSTER();
00067
00068     /*
00069     * Implement OSTM virtual methods
00070     */
00071     //virtual std::shared_ptr<ULSTER> _cast(std::shared_ptr<OSTM> _object);
00072     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00073     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00074     virtual void toString();
00075
00076     /*
00077     * Implement BANK virtual methods
00078     */
00079     virtual void SetAddress(std::string address);
00080     virtual std::string GetAddress() const;
00081     virtual void SetBalance(double balance);
00082     virtual double GetBalance() const;
00083     virtual void SetAccountNumber(int accountNumber);
00084     virtual int GetAccountNumber() const;
00085     virtual void SetLastName(std::string lastName);
00086     virtual std::string GetLastName() const;
00087     virtual void SetFirstName(std::string firstName);
00088     virtual std::string GetFirstName() const;
00089     virtual void SetFullname(std::string fullname);
00090     virtual std::string GetFullname() const;
00091 private:
00092     std::string fullname;
00093     std::string firstName;
00094     std::string lastName;

```

```

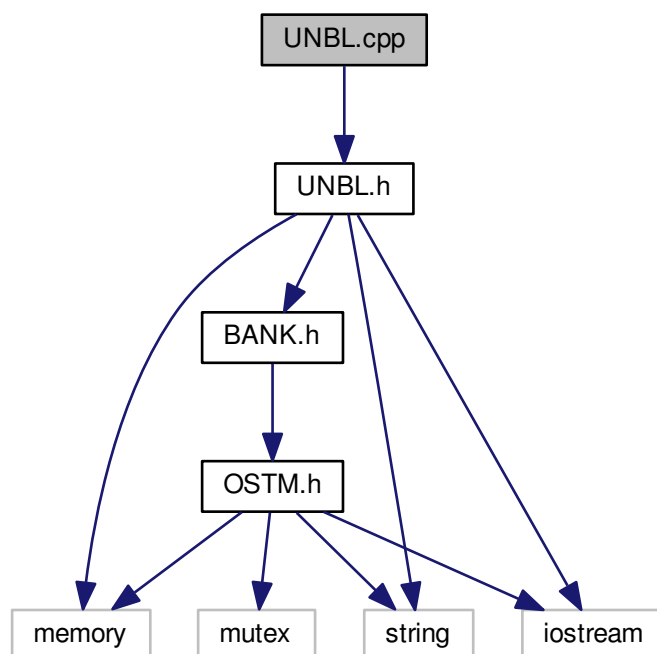
00095     int  accountNumber;
00096     double balance;
00097     std::string address;
00098
00099 };
00100
00101 #endif /* ULSTER_H */
00102

```

6.63 UNBL.cpp File Reference

```
#include "UNBL.h"
```

Include dependency graph for UNBL.cpp:



6.64 UNBL.cpp

```

00001 /*
00002  * File:    UNBL.cpp
00003  * Author:  Zoltan Fuzesi
00004  * IT Carlow : C00197361
00005  *
00006  * Created on January 17, 2018, 8:02 PM
00007  */
00008
00009 #include "UNBL.h"
00010
00011 UNBL::UNBL(const UNBL& orig) {
00012 }
00013
00014 UNBL::~UNBL() {
00015 }
00021 std::shared_ptr<OSTM> UNBL::getBaseCopy(std::shared_ptr<OSTM> object)
00022 {
00023     std::shared_ptr<BANK> objTO = std::dynamic_pointer_cast<BANK>(object);

```

```

00024     std::shared_ptr<BANK> obj(new UNBL(objTO,object->Get_Version(),object->Get_Unique_ID()));
00025     std::shared_ptr<OSTM> ostm_obj = std::dynamic_pointer_cast<OSTM>(obj);

00026     return ostm_obj;
00027 }
00033 void UNBL::copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from){
00034
00035     std::shared_ptr<UNBL> objTO = std::dynamic_pointer_cast<UNBL>(to);
00036     std::shared_ptr<UNBL> objFROM = std::dynamic_pointer_cast<UNBL>(from);
00037     objTO->Set_Unique_ID(objFROM->Get_Unique_ID());
00038     objTO->Set_Version(objFROM->Get_Version());
00039     objTO->SetAccountNumber(objFROM->GetAccountNumber());
00040     objTO->SetBalance(objFROM->GetBalance());
00041
00042 }
00046 //std::shared_ptr<UNBL> UNBL::_cast(std::shared_ptr<OSTM> _object){
00047 //
00048 //     return static_cast<std::shared_ptr<UNBL>>(_object);
00049 //}
00053 void UNBL::toString()
00054 {
00055     std::cout << "\nUNBL BANK" << "\nUnique ID : " << this->Get_Unique_ID() << "\nInt account : "
00056     << this->GetAccountNumber() << "\nDouble value : " << this->
00057     GetBalance() << "\nFirst name: " << this->GetFirstName() << "\nLast name : " <<
00058     this->GetLastName() << "\nVersion number : " << this->Get_Version() << std::endl;
00059 }
00060
00061 void UNBL::SetAddress(std::string address) {
00062     this->address = address;
00063 }
00064
00065 std::string UNBL::GetAddress() const {
00066     return address;
00067 }
00068
00069 void UNBL::SetBalance(double balance) {
00070     this->balance = balance;
00071 }
00072
00073 double UNBL::GetBalance() const {
00074     return balance;
00075 }
00076
00077 void UNBL::SetAccountNumber(int accountNumber) {
00078     this->accountNumber = accountNumber;
00079 }
00080
00081 int UNBL::GetAccountNumber() const {
00082     return accountNumber;
00083 }
00084
00085 void UNBL::SetLastName(std::string lastName) {
00086     this->lastName = lastName;
00087 }
00088
00089 std::string UNBL::GetLastName() const {
00090     return lastName;
00091 }
00092
00093 void UNBL::SetFirstName(std::string firstName) {
00094     this->firstName = firstName;
00095 }
00096
00097 std::string UNBL::GetFirstName() const {
00098     return firstName;
00099 }
00100
00101 void UNBL::SetFullname(std::string fullname) {
00102     this->fullname = fullname;
00103 }
00104
00105 std::string UNBL::GetFullname() const {
00106     return fullname;
00107 }

```

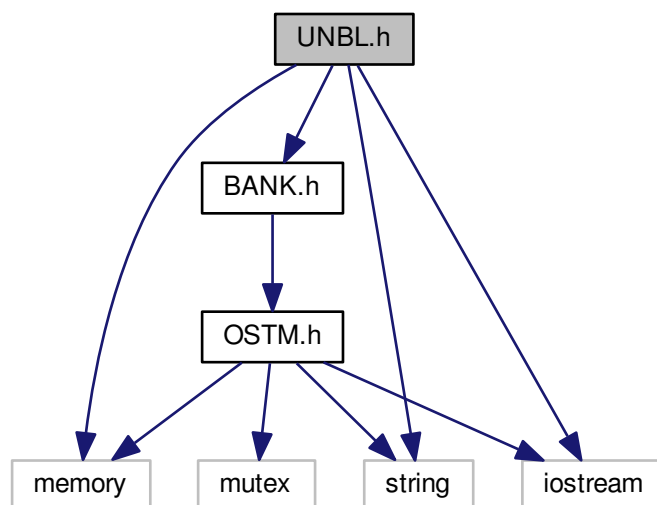
6.65 UNBL.h File Reference

```

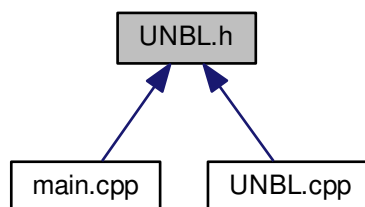
#include "BANK.h"
#include <string>
#include <memory>
#include <iostream>

```

Include dependency graph for UNBL.h:



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



Classes

- class [UNBL](#)

6.66 UNBL.h

```

00001
00002 /*
00003  * File:   UNBL.h
00004  * Author: Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
  
```



```

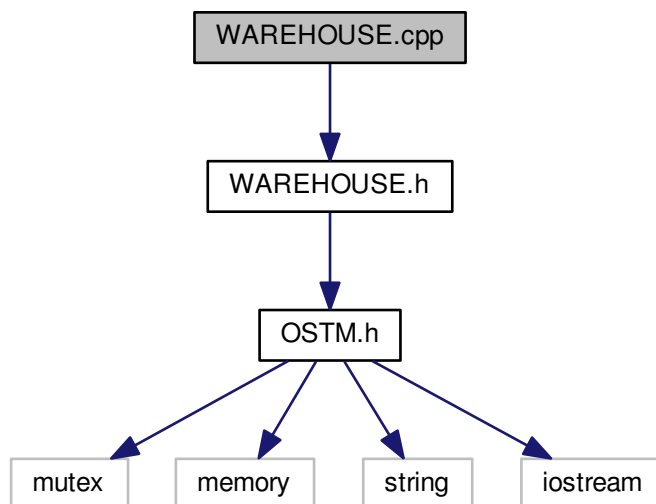
00008  */
00009
00010 #ifndef UNBL_H
00011 #define UNBL_H
00012 #include "BANK.h"
00013 #include <string>
00014 #include <memory>
00015 #include <iostream>
00019 class UNBL : public BANK {
00020 public:
00024     UNBL() : BANK() {
00025         this->accountNumber = 0;
00026         this->balance = 50;
00027         this->firstName = "Joe";
00028         this->lastName = "Blog";
00029         this->address = "High street, Carlow";
00030         this->fullname = firstName + " " + lastName;
00031     };
00035     UNBL(int accountNumber, double balance, std::string
        firstName, std::string lastName, std::string address) :
        BANK() {
00036         this->accountNumber = accountNumber;
00037         this->balance = balance;
00038         this->firstName = firstName;
00039         this->lastName = lastName;
00040         this->address = address;
00041         this->fullname = firstName + " " + lastName;
00042     };
00046     UNBL(std::shared_ptr<BANK> obj, int _version, int _unique_id) : BANK(_version, _unique_id) {
00047
00048         this->accountNumber = obj->GetAccountNumber();
00049         this->balance = obj->GetBalance();
00050         this->firstName = obj->GetFirstName();
00051         this->lastName = obj->GetLastName();
00052         this->address = obj->GetAddress();
00053         this->fullname = obj->GetFirstName() + " " + obj->GetLastName();
00054     };
00058     UNBL(const UNBL& orig);
00062     UNBL operator=(const UNBL& orig) {};
00066     virtual ~UNBL();
00067
00068     /*
00069     * Implement OSTM virtual methods
00070     */
00071     //virtual std::shared_ptr<UNBL> _cast(std::shared_ptr<OSTM> _object);
00072     virtual void copy(std::shared_ptr<OSTM> to, std::shared_ptr<OSTM> from);
00073     virtual std::shared_ptr<OSTM> getBaseCopy(std::shared_ptr<OSTM> object);
00074     virtual void toString();
00075
00076     /*
00077     * Implement BANK virtual methods
00078     */
00079     virtual void SetAddress(std::string address);
00080     virtual std::string GetAddress() const;
00081     virtual void SetBalance(double balance);
00082     virtual double GetBalance() const;
00083     virtual void SetAccountNumber(int accountNumber);
00084     virtual int GetAccountNumber() const;
00085     virtual void SetLastName(std::string lastName);
00086     virtual std::string GetLastName() const;
00087     virtual void SetFirstName(std::string firstName);
00088     virtual std::string GetFirstName() const;
00089     virtual void SetFullname(std::string fullname);
00090     virtual std::string GetFullname() const;
00091 private:
00092     std::string fullname;
00093     std::string firstName;
00094     std::string lastName;
00095     int accountNumber;
00096     double balance;
00097     std::string address;
00098
00099 };
00100
00101 #endif /* UNBL_H */
00102

```

6.67 WAREHOUSE.cpp File Reference

```
#include "WAREHOUSE.h"
```

Include dependency graph for WAREHOUSE.cpp:



6.68 WAREHOUSE.cpp

```

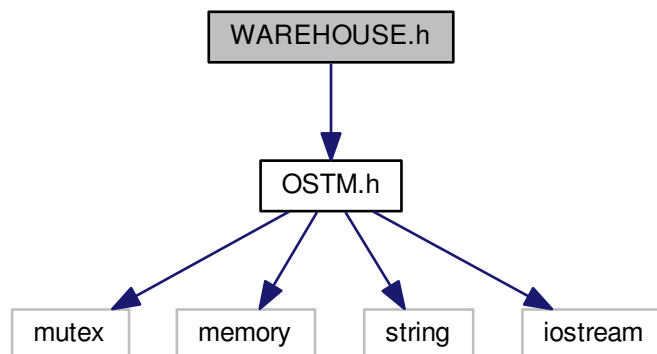
00001
00002 /*
00003  * File:   WAREHOUSE.cpp
00004  * Author: Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #include "WAREHOUSE.h"
00011
00012 WAREHOUSE::WAREHOUSE(const WAREHOUSE& orig) {
00013 }
00014
00015 WAREHOUSE::~WAREHOUSE() {
00016 }
00017

```

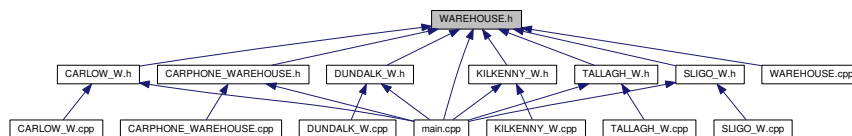
6.69 WAREHOUSE.h File Reference

```
#include "OSTM.h"
```

Include dependency graph for WAREHOUSE.h:



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



Classes

- class [WAREHOUSE](#)

6.70 WAREHOUSE.h

```

00001
00002 /*
00003  * File:   WAREHOUSE.h
00004  * Author: Zoltan Fuzesi
00005  * IT Carlow : C00197361
00006  *
00007  * Created on January 17, 2018, 8:02 PM
00008  */
00009
00010 #ifndef WAREHOUSE_H
00011 #define WAREHOUSE_H
00012 #include "OSTM.h"
00016 class WAREHOUSE : public OSTM {
00017 public:
00021     WAREHOUSE():OSTM(){
00022     };
00023     WAREHOUSE(int _version, int _unique_id) : OSTM(_version, _unique_id){
00024     };
00029     WAREHOUSE(const WAREHOUSE& orig);
00037     virtual ~WAREHOUSE();
00038
00039     /*
  
```

```
00040      * WAREHOUSE BASE METHODS
00041      */
00042
00043      virtual void SetNumber_of_alcatel(int _number_of_alcatel){};
00044      virtual int GetNumber_of_alcatel(){};
00045      virtual void SetNumber_of_nokia(int _number_of_nokia){};
00046      virtual int GetNumber_of_nokia(){};
00047      virtual void SetNumber_of_huawei(int _number_of_huawei){};
00048      virtual int GetNumber_of_huawei(){};
00049      virtual void SetNumber_of_sony(int _number_of_sony){};
00050      virtual int GetNumber_of_sony(){};
00051      virtual void SetNumber_of_samsung(int _number_of_samsung){};
00052      virtual int GetNumber_of_samsung(){};
00053      virtual void SetNumber_of_iphones(int _number_of_iphones){};
00054      virtual int GetNumber_of_iphones(){};
00055      virtual void SetShop_name(std::string _shop_name){};
00056      virtual std::string GetShop_name(){};
00057      virtual void SetShop_address(std::string _shop_address){};
00058      virtual std::string GetShop_address(){};
00059
00060 private:
00061
00062 };
00063
00064 #endif /* WAREHOUSE_H */
00065
```