Reference Manual

Generated by Doxygen 1.8.13

Contents

1 Test List

Member TEST_CASE ("deposit")

deposit() member function to check if depositing 100 into the checking will change the checking account to be 100. Calls BankAccount member function deposit A portfolio with one saving and one checking bank account.

```
portfolio.cpp:68: passed: ss.str() == "The amount in the checking account is
100\n" "The amount in the saving account is 0\n" for: "The amount in the
checking account is 100
The amount in the saving account is 0
"
==
"The amount in the checking account is 100
The amount in the saving account is 0
"
Passed 1 test case with 1 assertion.
```

Member TEST_CASE ("output")

withdraw() deposit() transfer() print_balance() member function to check checking if the output matches when all function are called Calls BankAccount all of member functions A portfolio with one saving and one checking bank account.

```
portfolio.cpp:175: passed: ss.str() == "The amount in the checking account is
300\n" "The amount in the saving account is 290\n" for: "The amount in the
checking account is 300
The amount in the saving account is 290
"
==
"The amount in the checking account is 300
The amount in the saving account is 290
"
Passed 1 test case with 1 assertion.
```

Member TEST_CASE ("print_balance")

print_balance() member function to check if multiple deposits, transfers, and withdraws are made the function will print the total of both accounts Calls BankAccount member function get_balance A portfolio with one saving and one checking bank account.

```
portfolio.cpp:149: passed: ss.str() == "The amount in the checking account is
200\n" "The amount in the saving account is 200\n" for: "The amount in the
checking account is 200
The amount in the saving account is 200
```

```
"
==
"The amount in the checking account is 200
The amount in the saving account is 200
"
Passed 1 test case with 1 assertion.
```

Member TEST_CASE ("transfer")

transfer() member function to check if depositing 200 into the saving after adding 100 and transfering 100 from saving to checking will change the checking account to be 200 and saving to 100; Calls BankAccount member function withdraw and deposit A portfolio with one saving and one checking bank account.

```
portfolio.cpp:120: passed: ss.str() == "The amount in the checking account is
200\n" "The amount in the saving account is 100\n" for: "The amount in the
checking account is 200
The amount in the saving account is 100
"
==
"The amount in the checking account is 200
The amount in the saving account is 100
"
Passed 1 test case with 1 assertion.
```

Member TEST_CASE ("withdraw")

withdraw() member function to check if withdrawing 100 into the checking after adding 200 will change the checking account to be 100. Calls BankAccount member function withdraw A portfolio with one saving and one checking bank account.

```
portfolio.cpp:93: passed: ss.str() == "The amount in the checking account is
0\n" "The amount in the saving account is 100\n" for: "The amount in the
checking account is 0
The amount in the saving account is 100
"
==
"The amount in the checking account is 0
The amount in the saving account is 100
"
Passed 1 test case with 1 assertion.
```

2 Class Index

2.1 Class List

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

3 File Index

3.1 File List

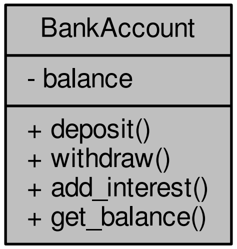
Here is a list of all files with brief descriptions:

4 Class Documentation

4.1 BankAccount Class Reference

```
#include <bankAccount.h>
```

Collaboration diagram for BankAccount:



Public Member Functions

- void deposit (double amount)
- void withdraw (double amount)
- void add_interest (double rate)
- double get_balance () const

Private Attributes

- double balance

4.1.1 Detailed Description

A bank account whose balance can be changed by deposits and withdrawals.

Definition at line 4 of file bankAccount.h.

4.1.2 Member Function Documentation

4.1.2.1 add_interest()

```
void BankAccount::add_interest (
            double rate )
```

Adds interest to this account.

Parameters

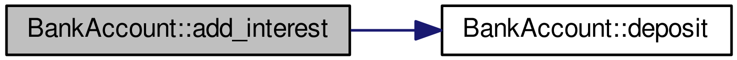
rate	the interest rate in percent

Definition at line 53 of file bankAccount.h.

```
{
   double amount = balance * rate / 100;
```

```
    deposit(amount);
}
```

Here is the call graph for this function:



4.1.2.2 deposit()

```
void BankAccount::deposit (
             double amount )
```

Makes a deposit into this account.

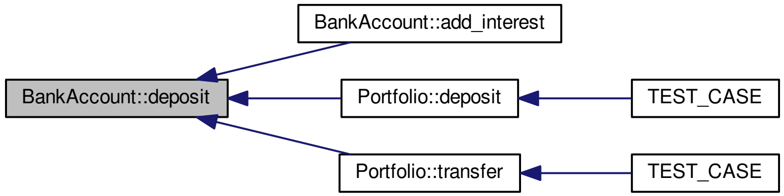
Parameters

amount	the amount of the deposit

Definition at line 35 of file bankAccount.h.

```
{
    balance = balance + amount;
}
```

Here is the caller graph for this function:



4.1.2.3 get_balance()

`double BankAccount::get_balance ( ) const`

Gets the current balance of this bank account.

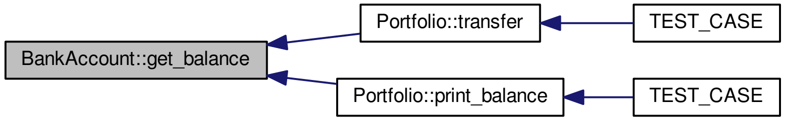
Returns

the current balance

Definition at line 59 of file bankAccount.h.

```
{
    return balance;
}
```

Here is the caller graph for this function:



4.1.2.4 withdraw()

```
void BankAccount::withdraw (
            double amount )
```

Makes a withdrawal from this account, or charges a penalty if sufficient funds are not available.

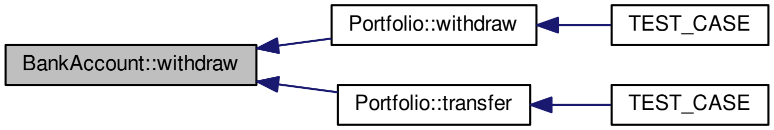
Parameters

amount	the amount of the withdrawal

Definition at line 40 of file bankAccount.h.

```
{
    const double PENALTY = 10;
    if (amount > balance)
    {
        balance = balance - PENALTY;
    }
    else
    {
        balance = balance - amount;
    }
}
```

Here is the caller graph for this function:



4.1.3 Member Data Documentation

4.1.3.1 balance

`double BankAccount::balance [private]`

Definition at line 32 of file bankAccount.h.

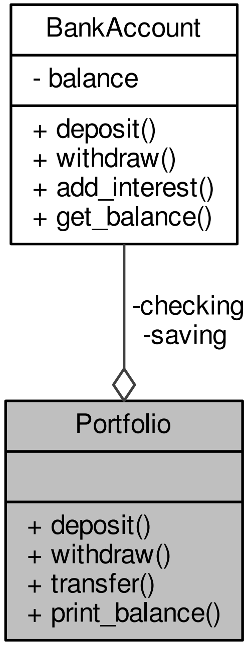
The documentation for this class was generated from the following file:

- bankAccount.h

4.2 Portfolio Class Reference

```
#include <portfolio.h>
```

Collaboration diagram for Portfolio:



Public Member Functions

- void deposit (double amount, std::string acount)
- void withdraw (double amount, std::string acount)
- void transfer (double amount, std::string acount)
- void print_balance () const

Private Attributes

- BankAccount checking
- BankAccount saving

4.2.1 Detailed Description

A portfolio with one saving and one checking bank account. Money can be transfer or deposited or withdraw in each account

Definition at line 5 of file portfolio.h.

4.2.2 Member Function Documentation

4.2.2.1 deposit()

```
void Portfolio::deposit (
            double amount,
            std::string acount )
```

Makes a deposit from the saving or checking account. Calls BankAccount member function deposit

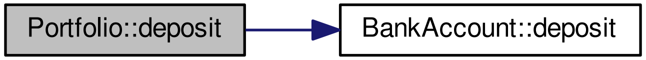
Parameters

amount	the amount of the deposit

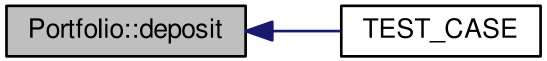
Definition at line 6 of file portfolio.cpp.

```
{
    if (account == "S")
        saving.deposit(amount);
    if (account == "C")
        checking.deposit(amount);
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.2 print_balance()

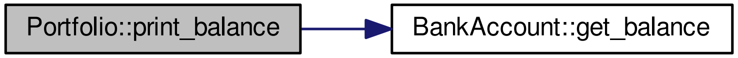
`void Portfolio::print_balance ( ) const`

Prints the current balances of both bank accounts. Calls BankAccount member function get_balance

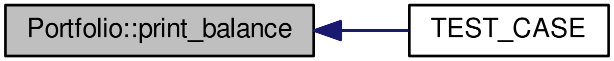
Definition at line 36 of file portfolio.cpp.

```
{
    double balance = checking.get_balance();
    std::cout << "The amount in the checking account is "
    << balance << std::endl;
    balance = saving.get_balance();
    std::cout << "The amount in the saving account is "
    << balance << std::endl;
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.3 transfer()

```
void Portfolio::transfer (
             double amount,
             std::string acount )
```

Transfers an amount from saving or checking to the opposing account Calls BankAccount member function deposit and withdraw

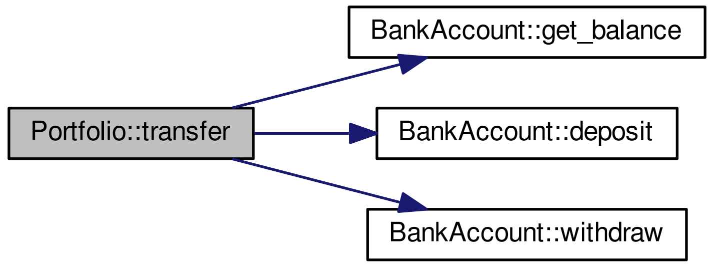
Parameters

amount	the amount of the transfer

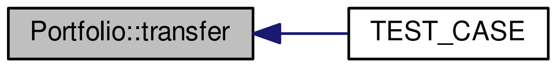
Definition at line 20 of file portfolio.cpp.

```
{
    if (account == "S")
    {
        if (amount <= saving.get_balance())
            checking.deposit(amount);
        saving.withdraw(amount);
    }
    if (account == "C")
    {
        if (amount <= checking.get_balance())
            saving.deposit(amount);
        checking.withdraw(amount);
    }
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.2.4 withdraw()

```
void Portfolio::withdraw (
            double amount,
            std::string acount )
```

Makes a withdraw from the saving or checking account. Calls BankAccount member function withdraw

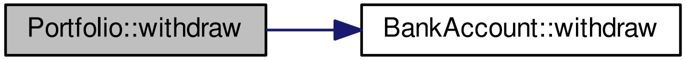
Parameters

amount	the amount of the withdraw

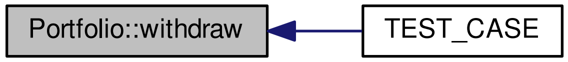
Definition at line 13 of file portfolio.cpp.

```
{
    if (account == "S")
        saving.withdraw(amount);
    if (account == "C")
        checking.withdraw(amount);
}
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.2.3 Member Data Documentation

4.2.3.1 checking

`BankAccount Portfolio::checking [private]`

Definition at line 36 of file portfolio.h.

4.2.3.2 saving

`BankAccount Portfolio::saving [private]`

Definition at line 37 of file portfolio.h.

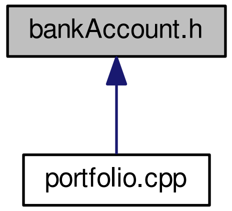
The documentation for this class was generated from the following files:

- portfolio.h
- portfolio.cpp

5 File Documentation

5.1 bankAccount.h File Reference

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



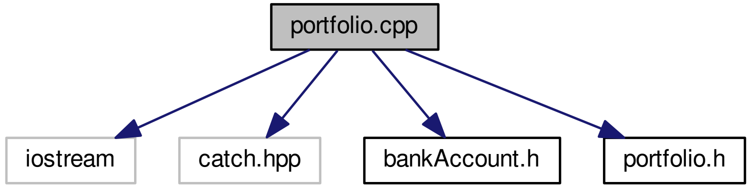
Classes

- class BankAccount

5.2 portfolio.cpp File Reference

```
#include <iostream>
#include "catch.hpp"
#include "bankAccount.h"
```

```
#include "portfolio.h"
```
Include dependency graph for portfolio.cpp:



Functions

- TEST_CASE ("deposit")
- TEST_CASE ("withdraw")
- TEST_CASE ("transfer")
- TEST_CASE ("print_balance")
- TEST_CASE ("output")

5.2.1 Function Documentation

5.2.1.1 TEST_CASE() [1/5]

```
TEST_CASE (
             "deposit" )
```

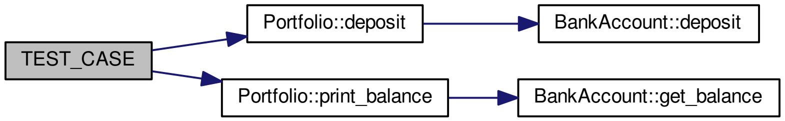
Test deposit() member function to check if depositing 100 into the checking will change the checking account to be 100. Calls BankAccount member function deposit A portfolio with one saving and one checking bank account.

```
portfolio.cpp:68: passed: ss.str() == "The amount in the checking account is
100\n" "The amount in the saving account is 0\n" for: "The amount in the
checking account is 100
The amount in the saving account is 0
"
==
"The amount in the checking account is 100
The amount in the saving account is 0
"
Passed 1 test case with 1 assertion.
```

Definition at line 54 of file portfolio.cpp.

```
55 {
56     Portfolio account1;
57     account1.deposit(100,"C");
58     std::streambuf *b = std::cout.rdbuf();
59     std::stringstream ss;
60     std::streambuf *sb = ss.rdbuf();
61     std::cout.rdbuf(sb);
62     // Now all output will be redirected into ss
63     account1.print_balance();
64     // set output back to the terminal
65     std::cout.rdbuf(b);
66     CHECK(ss.str() ==
67     "The amount in the checking account is 100\n"
68     "The amount in the saving account is 0\n");
69 }
```

Here is the call graph for this function:



5.2.1.2 TEST_CASE() [2/5]

```
TEST_CASE (
             "withdraw"  )
```

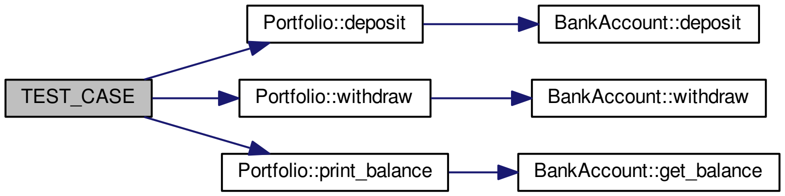
Test withdraw() member function to check if withdrawing 100 into the checking after adding 200 will change the checking account to be 100. Calls BankAccount member function withdraw A portfolio with one saving and one checking bank account.

```
portfolio.cpp:93: passed: ss.str() == "The amount in the checking account is
0\n" "The amount in the saving account is 100\n" for: "The amount in the
checking account is 0
The amount in the saving account is 100
"
==
"The amount in the checking account is 0
The amount in the saving account is 100
"
Passed 1 test case with 1 assertion.
```

Definition at line 78 of file portfolio.cpp.

```
{
    Portfolio account1;
    account1.deposit(200,"S");
    account1.withdraw(100,"S");
    std::streambuf *b = std::cout.rdbuf();
    std::stringstream ss;
    std::streambuf *sb = ss.rdbuf();
    std::cout.rdbuf(sb);
    // Now all output will be redirected into ss
    account1.print_balance();
    // set output back to the terminal
    std::cout.rdbuf(b);
    CHECK(ss.str() ==
    "The amount in the checking account is 0\n"
    "The amount in the saving account is 100\n");
}
```

Here is the call graph for this function:



5.2.1.3 TEST_CASE() [3/5]

```
TEST_CASE (
             "transfer"  )
```

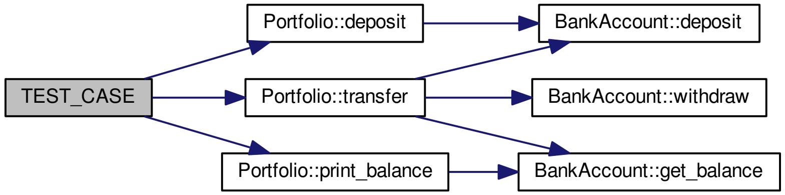
Test transfer() member function to check if depositing 200 into the saving after adding 100 and transfering 100 from saving to checking will change the checking account to be 200 and saving to 100; Calls BankAccount member function withdraw and deposit A portfolio with one saving and one checking bank account.

```
portfolio.cpp:120: passed: ss.str() == "The amount in the checking account is
200\n" "The amount in the saving account is 100\n" for: "The amount in the
checking account is 200
The amount in the saving account is 100
"
==
"The amount in the checking account is 200
The amount in the saving account is 100
"
Passed 1 test case with 1 assertion.
```

Definition at line 104 of file portfolio.cpp.

```
{
    Portfolio account1;
    account1.deposit(200,"S");
    account1.deposit(100,"C");
    account1.transfer(100,"S");
    std::streambuf *b = std::cout.rdbuf();
    std::stringstream ss;
    std::streambuf *sb = ss.rdbuf();
    std::cout.rdbuf(sb);
    // Now all output will be redirected into ss
    account1.print_balance();
    // set output back to the terminal
    std::cout.rdbuf(b);
    CHECK(ss.str() ==
    "The amount in the checking account is 200\n"
    "The amount in the saving account is 100\n");
}
```

Here is the call graph for this function:



5.2.1.4 TEST_CASE() [4/5]

```
TEST_CASE (
            "print_balance" )
```

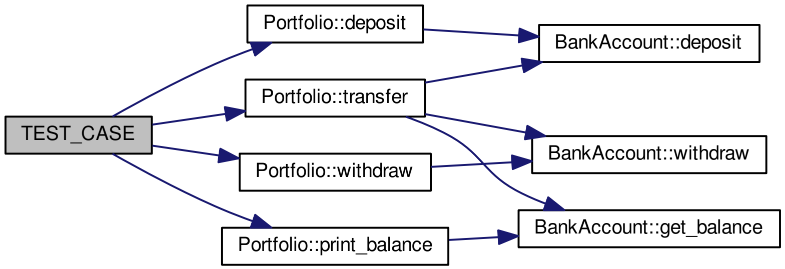
Test print_balance() member function to check if multiple deposits, transfers, and withdraws are made the function will print the total of both accounts Calls BankAccount member function get_balance A portfolio with one saving and one checking bank account.

```
portfolio.cpp:149: passed: ss.str() == "The amount in the checking account is
200\n" "The amount in the saving account is 200\n" for: "The amount in the
checking account is 200
The amount in the saving account is 200
"
==
"The amount in the checking account is 200
The amount in the saving account is 200
"
Passed 1 test case with 1 assertion.
```

Definition at line 130 of file portfolio.cpp.

```
{
    Portfolio account1;
    account1.deposit(500,"S");
    account1.withdraw(100,"S");
    account1.withdraw(100,"S");
    account1.deposit(100,"C");
    account1.transfer(200,"S");
    account1.transfer(100,"C");
    std::streambuf *b = std::cout.rdbuf();
    std::stringstream ss;
    std::streambuf *sb = ss.rdbuf();
    std::cout.rdbuf(sb);
    // Now all output will be redirected into ss
    account1.print_balance();
    // set output back to the terminal
    std::cout.rdbuf(b);
    CHECK(ss.str() ==
    "The amount in the checking account is 200\n"
    "The amount in the saving account is 200\n");
}
```

Here is the call graph for this function:



5.2.1.5 TEST_CASE() [5/5]

```
TEST_CASE (
             "output"  )
```

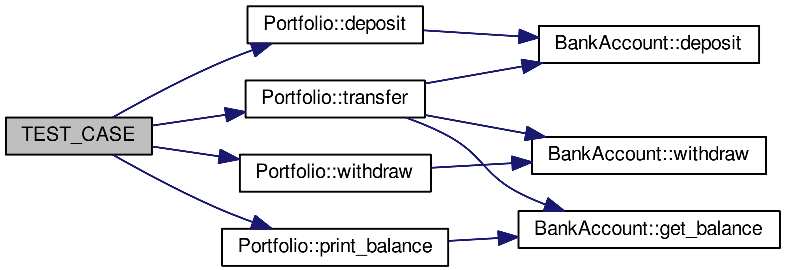
Test withdraw() deposit() transfer() print_balance() member function to check checking if the output matches when all function are called Calls BankAccount all of member functions A portfolio with one saving and one checking bank account.

```
portfolio.cpp:175: passed: ss.str() == "The amount in the checking account is
300\n" "The amount in the saving account is 290\n" for: "The amount in the
checking account is 300
The amount in the saving account is 290
"
==
"The amount in the checking account is 300
The amount in the saving account is 290
"
Passed 1 test case with 1 assertion.
```

Definition at line 158 of file portfolio.cpp.

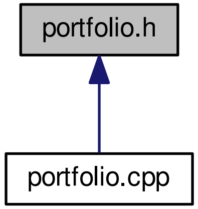
```
{
    Portfolio account1;
    account1.deposit(500,"C");
    account1.deposit(300,"S");
    account1.withdraw(200,"C");
    account1.transfer(500,"S");
    std::streambuf *b = std::cout.rdbuf();
    std::stringstream ss;
    std::streambuf *sb = ss.rdbuf();
    std::cout.rdbuf(sb);
    // Now all output will be redirected into ss
    account1.print_balance();
    // set output back to the terminal
    std::cout.rdbuf(b);
    CHECK(ss.str() ==
    "The amount in the checking account is 300\n"
    "The amount in the saving account is 290\n");
}
```

Here is the call graph for this function:



5.3 portfolio.h File Reference

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



Classes

- class Portfolio

Index