



<b>ID</b>	CS001142	<b>Creation date</b>	October 14, 2008
<b>Platform</b>	S60 3rd Edition, FP2	<b>Tested on devices</b>	Nokia 6220 Classic
<b>Category</b>	Open C/C++	<b>Subcategory</b>	Files/Data

**Keywords (APIs, classes, methods, functions):** srand, rand, time, vector, iterator, ostream\_iterator, random\_shuffle()

## Overview

This code snippet shows how to use the `random_shuffle` algorithm to randomize a sequence of elements. The `random_shuffle()` function randomly re-orders the elements in the given range. It works only with sequences that provide random-access iterators, such as vectors and arrays. The function takes two random-access iterators (first, last) as parameters and an optional random-number generation functor. The header file `<algorithm>` must be included in the application in order to use the `random_shuffle` algorithm.

**Note:** In order to use this code, you need to install the [Open C/C++ plug-in](#).

This snippet can be self-signed.

## MMP file

The following libraries are required:

```
STATICLIBRARY  libcrtd0.lib

LIBRARY  libstdcpp.lib
LIBRARY  libc.lib
LIBRARY  euser.lib
```

## Header file

```
#ifndef EXAMPLECARDS_H_
#define EXAMPLECARDS_H_
```

## CS001142\_-\_Shuffling\_data\_using\_STL\_random\_shuffle\_algorithm

```
#include <vector>

typedef enum
{
    Ace, _2, _3, _4, _5, _6, _7, _8, _9,
    _10, Jack, Queen, King } ERank;

typedef enum
{
    Club, Diamond,
    Heart, Spade
} ESuit;

class Card
{
public:
    Card(ESuit suit, ERank rank);
    void GetValues(ESuit &suit, ERank &rank);
private:
    ERank m_rank;
    ESuit m_suit;
};

class Deck
{
public:
    Deck(void);
    ~Deck();
    void Shuffle(bool aSeed = false);
    void PrintDeck();
private:
    std::vector<Card> m_cards;
};

#endif /*EXAMPLECARDS_H_*/
```

## Source file

```
#include <algorithm>
#include <iostream>
#include <ctime>
#include "ExampleCards.h"

using namespace std;

Card::Card(ESuit suit, ERank rank)
{
    m_rank = rank;
    m_suit = suit;
}

void Card::GetValues(ESuit &suit, ERank &rank)
{
    rank = m_rank;
    suit = m_suit;
}

Deck::Deck(void)
{
    int count = 0;
```

Header file

## CS001142\_-\_Shuffling\_data\_using\_STL\_random\_shuffle\_algorithm

```
        for (int s = int(Club); s <= int(Spade); ++s)
        {
            for (int c = int(Ace); c <= int(King); ++c)
            {
                m_cards.push_back(Card(ESuit(s), ERank(c)));
                ++count;
            }
        }
    }

Deck::~Deck(void)
{
    m_cards.clear();
}

void Deck::PrintDeck()
{
    const char suits[] =
    {'C', 'D', 'H', 'S'};

    const char ranks[] =
    {'A', '2', '3', '4', '5',
     '6', '7', '8', '9',
     '0', 'J', 'Q', 'K'};

    for(int i=0; i < (int)m_cards.size(); i++)
    {
        ESuit s; ERank r;
        m_cards[i].GetValues(s,r);

        cout << '(' << suits[(int)s] <<
                '-' << ranks[(int)r] << ')';
    }
    cout << endl;
}

void Deck::Shuffle(bool aSeed)
{
    static bool seeded;
    if (!seeded && aSeed)
    {
        srand ( unsigned ( time (NULL) ) );
        seeded = true;
    }

    random_shuffle( m_cards.begin(), m_cards.end() );
}
```

### The main application:

```
#include <algorithm> //random_shuffle
#include <iterator> //iterator, ostream_iterator
#include <iostream> //cout
#include <vector> //vector
#include <cstdlib> //srand, rand
#include <ctime> //time

#include "ExampleCards.h"

//random-number generation functor
class ExampleRnd
{
```

## CS001142\_-\_Shuffling\_data\_using\_STL\_random\_shuffle\_algorithm

```
public:
    size_t operator( )(size_t n) const
    { return(rand( )%n ); }
};

int main()
{
    //set the random number seed for the rand() operator
    srand ( unsigned ( time (NULL) ) );

    //-- using random_shuffle with a vector
    vector<int> v;
    vector<int>::iterator vi;

    for(int index = 0; index<10; index++)
        v.push_back(index);

    random_shuffle(v.begin( ), v.end( ));

    vi = v.begin();
    while(vi != v.end())
        cout << *vi++ << " ";
    cout << endl;

    ExampleRnd rnd; //use own functor to shuffle
    random_shuffle(v.begin( ), v.end( ), rnd);

    vi = v.begin();
    while(vi != v.end())
        cout << *vi++ << " ";
    cout << endl;

    //-- using random_shuffle with an array
    string s[5];
    ostream_iterator<string> si(cout, " ");

    s[0] = "abc";
    s[1] = "def";
    s[2] = "ghi";
    s[3] = "jkl";
    s[4] = "mno";

    random_shuffle( s, s+5 ); //end must be one element past
    copy( s, s+5, si );
    cout << endl;

    //-- using a class that internally calls random_shuffle
    Deck d;

    cout << "An ordered deck:" << endl;
    d.PrintDeck();

    d.Shuffle();

    cout << "A shuffled deck:" << endl;
    d.PrintDeck();

    //getchar();

    return 0;
}
```

## Postconditions

The vector containing integer values from 1 to 10 has been shuffled two times with and without its own functor. After that, the array containing strings has been shuffled and finally the class implementing a card deck is also used to demonstrate the random shuffle algorithm.