Search: Go

Not logged in

Reference <random> uniform_real_distribution register log in

C++

Information
Tutorials
Reference
Articles
Forum

Reference

C library:
Containers:
Input/Output:
Multi-threading:
Other:
<algorithm>
<bitset>
<chrono>
<codecvt>
<complex>
<exception>
<functional>
<initializer_list>
<iterator>
<limits>
<locale>
<memory>
<new>
<numeric>
<random>
<ratio>
<regex>
<stdexcept>
<string>
<system_error>
<tuple>
<typeindex>
<typeinfo>
<type_traits>
<utility>
<valarray>

<random>

distributions:
bernoulli_distribution
binomial_distribution
cauchy_distribution
chi_squared_distribution
discrete_distribution
exponential_distribution
extreme_value_distribution
fisher_f_distribution
gamma_distribution
geometric_distribution
lognormal_distribution
negative_binomial_distribution
normal_distribution
piecewise_constant_distribution
piecewise_linear_distribution
poisson_distribution
student_t_distribution
uniform_int_distribution
uniform_real_distribution
weibull_distribution
generators:
default_random_engine
discard_block_engine
independent_bits_engine
knuth_b
linear_congruential_engine
mersenne_twister_engine
minstd_rand
minstd_rand0
mt19937
mt19937_64
random_device
ranlux24
ranlux24_base
ranlux48
ranlux48_base
shuffle_order_engine
subtract_with_carry_engine
other:
generate_canonical
seed_seq

uniform_real_distribution

uniform_real_distribution::(constructor)
member functions:
uniform_real_distribution::a
uniform_real_distribution::b
uniform_real_distribution::max

class template

std::uniform_real_distribution

<random>

```
template <class RealType = double> class uniform_real_distribution;
```

Uniform real distribution

Random number distribution that produces floating-point values according to a *uniform distribution*, which is described by the following *probability density function*:

$$p(x|a,b) = \frac{1}{b-a} \quad , \; a \le x < b$$

This distribution (also know as rectangular distribution) produces random numbers in a range `[a,b)` where all intervals of the same length within it are equally probable.

The distribution parameters, a and b, are set on construction.

To produce a random value following this distribution, call its member function operator().

Template parameters

`RealType`
A floating-point type. Aliased as member type `result_type`.
By default, this is `double`.

Member types

The following aliases are member types of `uniform_real_distribution`:

member type	definition	notes
`result_type`	The first template parameter (`RealType`)	The type of the numbers generated (defaults to `double`)
`param_type`	*not specified*	The type returned by member param.

Member functions

(constructor)	Construct uniform real distribution (public member function)
operator()	Generate random number (public member function)
reset	Reset distribution (public member function)
param	Distribution parameters (public member function)
min	Minimum value (public member function)
max	Upper bound of range (public member function)

Distribution parameters:

a	Lower bound of range (public member function)
b	Upper bound of range (public member function)

Non-member functions

operator<<	Insert into output stream (function template)
operator>>	Extract from input stream (function template)
relational operators	Relational operators (function template)

Example

```cpp
// uniform_real_distribution
#include <iostream>
#include <random>

int main()
{
  const int nrolls=10000;  // number of experiments
  const int nstars=95;     // maximum number of stars to distribute
  const int nintervals=10; // number of intervals

  std::default_random_engine generator;
  std::uniform_real_distribution<double> distribution(0.0,1.0);

  int p[nintervals]={};

  for (int i=0; i<nrolls; ++i) {
    double number = distribution(generator);
    ++p[int(nintervals*number)];
  }

  std::cout << "uniform_real_distribution (0.0,1.0):" << std::endl;
  std::cout << std::fixed; std::cout.precision(1);

  for (int i=0; i<nintervals; ++i) {
    std::cout << float(i)/nintervals << "-" << float(i+1)/nintervals << ": ";
    std::cout << std::string(p[i]*nstars/nrolls,'*') << std::endl;
  }

```

uniform_real_distribution::min
uniform_real_distribution::operator()
uniform_real_distribution::param
uniform_real_distribution::reset
non-member functions:
operator<<
operator>>
relational operators

```
  return 0;
}
```

Possible output:

```
uniform_real_distribution (0.0,1.0):
0.0-0.1: *********
0.1-0.2: *********
0.2-0.3: *********
0.3-0.4: *********
0.4-0.5: *********
0.5-0.6: *********
0.6-0.7: *********
0.7-0.8: *********
0.8-0.9: *********
0.9-1.0: *********
```

See also

uniform_int_distribution	Uniform discrete distribution (class template)
generate_canonical	Generate canonical numbers (function template)

Home page | Privacy policy

Spotted an error? contact us