Version 2.0: Written by Gregor N. Purdy.


# ElkinsEngine

INHERITS FROM RandomEngine

CLASS DESCRIPTION

The ElkinsEngine class implements a random number generator with a cycle length of 8.8 trillion.

Upon creation of an ElkinsEngine, the seeds are set using the system clock. Three calls are made to the system clock function, and for each the microseconds are used as the seed value. Thus, the relationships between the seeds are dependant upon system load.

The Elkins algorithm generates short integers (16 bits), but only 15 of those bits are pseudo-random. The same method is used in ElkinsEngine as in StandardEngine, i.e. 16 iterations of the algorithm are used, and one result is used to fill out the non-pseudo-random bit in each of the other results. This means the unit for ElkinsEngine is 15 short integers, or 30 bytes.

The algorithm used by the ElkinsEngine class is that given in the article:

"A Higly Random Random-Number Generator" by T.A. Elkins
Computer Language, 1989 December (Volume 6, Number 12), Page 59.
Published by:

Miller Freeman Publications
500 Howard Street
San Francisco, CA 94105
415-397-1881

This class is part of Version 2.0 of Random, distributed 1992 May 29.

Contemporary Design Studios
2339 Woodchip Way #1B
Ypsilanti, MI 48197
313-572-1779

Gregor N. Purdy
Software Designer
University of Michigan
gregor@umich.edu



Comments, suggestions, and bug reports are welcome. Feel free to drop us an email or letter with your comments.

(ii) You must credit Contemporary Design Studios and include the copyright information at the top of this document in your documentation and in your "Info Panel," if any, if your work is ever used by anyone other than yourself.

(iii) Contemporary Design Studios retains the right to make improvements to this work and to change the distribution conditions and terms in future versions, including commercial derivative works.

(iv) You may NOT in any way change the Random Classes and redistribute them, even under a new name, but you MAY, and in fact are encouraged to, report bugs and make suggestions for enhancements to Contemporary Design Studios.

(iv) You MAY make subclasses of Random that do whatever you want. Of course, these subclasses will be governed by your own rules, but the Random Classes will still be governed by these rules.

(v) You MAY distribute the source code of the Random Classes with your work, but if you do all the documentation for the Random Classes, including this notice, must be distributed with it. Nobody should ever see the source code to the Random Classes without also seeing these documentation files, so they can use it, too.

METHOD TYPES

| | |
|---|---|
| Creating and freeing instances | + alloc<br>- free |
| | Generation of random bits - makeRandom:<br>+ unit |
| Archiving | - read:<br>- write: |

CLASS METHODS

**alloc**

+ **alloc**

Returns a new uninitialized instance.

**unit**

+ (int)**unit**

Returns the number of bytes each invocation of **makeRandom:** will produce. For ElkinsEngine, this is 30.

INSTANCE METHODS

**free**

- **free**

Frees the memory occupied by the ElkinsEngine instance and returns **nil**.

**makeRandom:**

- **makeRandom:**(uchar *)*storage*

Performs 16 iterations of the Elkins algorithm, and stuffs the results into *storage* (note that only 30 bytes of pseudo-random bits are generated).

**read:**

- **read:**(NXTypedStream *)*stream*

Unarchives an ElkinsEngine from *stream*.

See also: - **write:**

**write:**

- **write:**(NXTypedStream *)*stream*

Archives an ElkinsEngine to *stream*.

See also: - **read:**