Unity Random

Random Numbers for Unity3d

Homepage: http://tucanosoftware.com/projects/unityrandom

FAQ

What is this?

In Unity3d there is already a Random number generator based on the **platform-specific** random generator. Here we present an alternative Random library for **Unity3d** designed to generate uniform Pseudo-Random deviates. The library use a fast PRNG (**Mersenne-Twister**) to generate: **Floating Number** in range [0-1] and in range [n-m], **Vector2** and **Vector3** data types. The library comes with some special functions designed specifically for a game design framework: **Shuffle Bag**, **Dice**, **Random Color**.

Which kind of transformations I can apply to the random uniform deviates?

The uniform deviates can be transformed with the distributions: **Standard Normal Distribution** and **Power-Law**. In addition is possible to generate floating random deviates coming from other distributions: **Poisson**, **Exponential** and **Gamma**.

How I can test the random numbers?

The library add a window to the **Unity3D** editor that allow you to **Test** and **Visualize** your random numbers, Vector2 and Vector3. With the **SAVE** button, you can write the sample of random number to a txt file. This is useful if you need to analyze in deep the distribution of your random numbers with a statistical software.

Usage (C#)

Initialization Initialization with a seed: `UnityRandom urand = new UnityRandom(int seed);`

Numbers A Random number in range [0-1]: @float val = urand.Value()

Transformations A Random number in range [0-1] with a Transformation: `float val = urand.Value(UnityRandom.Normalization.STDNORMAL, 5.0f)`

Vectors A random point in a disk with R=1: `Vector2 pos = urand.PointInADisk()`

Colors A random color in the range of visible light (rainbow): `Color col = urand.Rainbow()`

Dice A 2D6 dice roll `DiceRoll roll = urand.RollDice(2,DiceRoll.DiceType.D6)`

Documentation

Initialization

- Initialization without a seed: `UnityRandom urand = new UnityRandom();`
- Initialization with a seed: `UnityRandom urand = new UnityRandom(int seed);`

Numbers

Generation of uniform deviates in any range.

Available Transformations

- `UnityRandom.Normalization.STDNORMAL` with parameter: `float temperature`
- `UnityRandom.Normalization.POWERLAW` with parameter: `float power`

Value

- A Random number in range [0-1]: `float val = urand.Value()`
- A Random number in range [0-1] with a Transformation: `float val = urand.Value(UnityRandom.Normalization.STDNORMAL, 5.0f)`

Range

- A Random number in range [1-100]: `float val = urand.Range(1,100)`
- A Random number in range [m-n] with a Transformation: `float val = urand.Value(0,100,UnityRandom.Normalization.POWERLAW, 5.0f)`

Poisson

- The Poisson distribution (pronounced [pwasɔ̃]) is a discrete probability distribution that expresses the probability of a given number of events occurring in a fixed interval of time and/or space if these events occur with a known average rate and independently of the time since the last event. Example: `float val = urand.Poisson(5.0f)`

Exponential

- The Exponential distribution describes the time between events in a Poisson process, i.e. a process in which events occur continuously and independently at a constant average rate. Example: `float val = urand.Exponential(5.0f)`

Gamma

- The gamma distribution, like the lognormal distribution, is an alternative to consider for ecological variables that seem to be highly skewed. Example: `float val = urand.Gamma(5.0f)`

Vector2

generation of Unity Vector2 Objects.

Square

- A random **Vector2** point in a square with L=1: `Vector2 pos = urand.PointInASquare()`
- A random **Vector2** point in a square with L=1 **normalized**: `Vector2 pos = urand.PointInASquare(UnityRandom.Normalization.STDNORMAL, 5.0f)`

Disk/Circle

- A random **Vector2** point in a circle (in the circle **perimeter**) with R=1: `Vector2 pos = urand.PointInACircle()`
- A random **Vector2** point in a circle (in the circle **perimeter**) with R=1 **normalized**: `Vector2 pos = urand.PointInACircle(UnityRandom.Normalization.STDNORMAL, 5.0f)`
- A random **Vector2** point in a disk (in the circle **area**) with R=1: `Vector2 pos = urand.PointInADisk()`
- A random **Vector2** point in a circle (in the circle **area**) with R=1 **normalized**: `Vector2 pos = urand.PointInADisk(UnityRandom.Normalization.STDNORMAL, 5.0f)`

Vector3

generation of Unity Vector3

Cube

- A random **Vector3** point inside a cube with L=1: `Vector3 pos = urand.PointInACube()`
- A random **Vector3** point inside a cube with L=1 **normalized**: `Vector3 pos = urand.PointInACube(UnityRandom.Normalization.STDNORMAL, 5.0f)`
- A random **Vector3** point in the surface of a cube with L=1: `Vector3 pos =`

`urand.PointOnACube()`
- A random **Vector3** point inside a cube with L=1 **normalized**: `Vector3 pos = urand.PointOnACube(UnityRandom.Normalization.STDNORMAL, 5.0f)`

Sphere

- A random **Vector3** point inside a sphere (in the sphere **volume**) with R=1: `Vector3 pos = urand.PointInASphere()`
- A random **Vector3** point in the sphere surface (in the sphere **surface**) with R=1: `Vector3 pos = urand.PointOnASphere()`

Color

- A random **Color** in the range of visible light (rainbow): `Color col = urand.Rainbow()`
- A random **Color** in the range of visible light (rainbow) **normalized**: `Color col = urand.Rainbow(UnityRandom.Normalization.STDNORMAL, 5.0f)`

Dice

- A **Dice Roll**: `DiceRoll roll = urand.RollDice(n,type)`

Dice types:

- DiceRoll.DiceType.D2
- DiceRoll.DiceType.D3
- DiceRoll.DiceType.D4
- DiceRoll.DiceType.D6
- DiceRoll.DiceType.D8
- DiceRoll.DiceType.D10
- DiceRoll.DiceType.D12
- DiceRoll.DiceType.D20
- DiceRoll.DiceType.D30
- DiceRoll.DiceType.D100

Example 2D6:

```
DiceRoll roll = urand.RollDice(2,DiceRoll.DiceType.D6)
```

Dev Notes

- **Clone and test**
  - Fork the main project or clone: git://github.com/tucano/UnityRandom.git
  - Create an empty project in Unity
  - cd ./Assets
  - clone the repo there!